Abstract

LUNSFORD II, PHILIP J. The Frequency Domain Behavioral Modeling and Simulation of Nonlinear Analog Circuits and Systems. (Under the direction of Michael B. Steer.)

A new technique for the frequency–domain behavioral modeling and simulation of nonautonomous nonlinear analog subsystems is presented. This technique extracts values of the Volterra nonlinear transfer functions and stores these values in binary files. Using these files, the modeled substem can be simulated for an arbitrary periodic input expressed as a finite sum of sines and cosines. Furthermore, the extraction can be based on any circuit simulator that is capable of steady state simulation. Thus a large system can be divided into smaller subsystems, each of which is characterized by circuit level simulations or lab measurements. The total system can then be simulated using the subsystem characterization stored as tables in binary files.

Using known ideal nonlinear circuits, the method extraction technique was tested for nonlinearities up to seventh order. The Volterra nonlinear transfer functions of an equalizing circuit were extracted from the results of transient simulations which utilized the shooting method. The resulting tables were then used to simulate an arbitrary waveform and the results closely matched the equivalent time domain circuit simulation.

The Frequency Domain Behavioral Modeling and Simulation of Nonlinear Analog Circuits and Systems

by

Philip J. Lunsford, II

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Department of Electrical and Computer Engineering

Raleigh, NC

1993

Approved By:

Co-Chairman, Advisory Committee	Co-Chairman, Advisory Committee

Acknowledgements

I wish to thank the following people for the support they have given me during my time at North Carolina State University.

To Dr. Michael Steer for his guidance, support, and technical direction.

To IBM for supporting me and this research.

To everyone in Daniels 333 and 336 for their outlook, work ethic, and sense of humor.

To my wife, Anne, for providing moral support and the correct perspective, and to Jean Elizabeth Lunsford, my daughter, for giving me incentive to no longer be a graduate student.

Table of Contents

List of Figures

List of Tables

List of Symbols

!	factorial
$()^*$	complex conjugate
$()^\dagger_k$	complex conjugate for $n_k < 0$, no operation otherwise
$()^\ddagger_k$	complex conjugate for $n_k \geq 0$, no operation otherwise
$A, a, b, c, \hat{c}, d$	Coefficients, real or complex numbers
B	memoryless bivariate function
C	intermediate complex variable
C_i	capacitance value of the ith component
E	error
$E(\sigma, x)$	intermediate function used to define calculation of the output of a bivariate system
$F()$	an analytic function
$F(\rho, z)$	intermediate function used to define calculation of the output of a bivariate system
G	a linear transfer function
$H_n()$	nth order Volterra nonlinear transfer function
$H_{m;n}()$	m, nth order Volterra nonlinear transfer function of a bivariate system
I	current
$I_{\omega k}$	Phasor of the current corresponding to frequency ω_k at the interface to a linear subcircuit
$I'_{\omega k}$	Phasor of the current corresponding to frequency ω_k at the interface to a nonlinear subcircuit
L	inductance
P	a vector of parameters that characterize a model
R_i	resistance value of the ith resistor
R	the frequency domain representation of the response to a system
T	subcomponent of the output
U	intermediate complex variable
V	voltage
V_t	thermal voltage
$V_{\omega k}$	Phasor of the voltage corresponding to frequency ω_k at the interface to a linear subcircuit
$V'_{\omega k}$	Phasor of the voltage corresponding to frequency ω_k at the interface to a nonlinear subcircuit
W	intermediate complex variable
X_k	phasor representation of the kth component of the input of a univariate system or of the input of the first port in a bivariate system
$Y(f)$	frequency domain representation of the output

$Y_n(f)$	nth order frequency domain representation of the output
Y_q	phasor representation of the qth component of the output
Z_k	frequency domain representation (phasor) of the kth component of the input to the second port in a bivariate system
β	intermediate real variable
,	phasor used in intermediate calculation that always has a length of 1
Λ	phasor used in intermediate calculation that always has a length of 1
Φ	subcomponent of the output
α	indexing variable that represents the number of +/- pairs of frequencies in the corresponding FIPD, or a real scaling variable
$\bar{f}_i$	ith input component frequency to the first port of a bivariate system
$\delta()$	impulse function
ϵ_k	neumann factor, ϵ_k=1 for k=0, ϵ_k=2 otherwise
η	indexing variable used to index other indexing variables
$\hat{f}_i$	ith input component frequency to the second port of a bivariate system
λ	time delay variable associated with the second port of a bivariate system
ν	indexing variable corresponding to the number of different values of the scaling variable α
ω_k	the output radian frequency of the kth output component
ϕ_k	phase shift of the kth component of the input of a univariate system or the first port of a bivariate system
π	the ratio of the circumference to the diameter of a circle = 3.141592...
τ	time delay or dummy time variable
θ_k	phase shift of the kth component of the input to the second port of a bivariate system
ζ	indexing variable used to index other indexing variables
e	base for natural logrithm = 2.7128281...
f_i	frequency for the ith component
g	impulse response function corresponding to the transfer function G
$g(V_{RF}, V_{LO})$	real bivariate polynomial that characterizes a ring diode mixer
$h_n()$	nth order Volterra kernel
$h_{m;n}()$	m, nth order Volterra kernel of a bivariate system
i, k, l, m, p, q, s, μ	indexing variables
j	square root of -1
m	indexing variable representing the order of the calculation corresponding to the second port of a bivariate system
n	indexing variable representing the order of the calculation
n_i	indexing variable usually represents the order of the ith

	subcomponent being calculated
p_i	ith component of P
r	the time domain representation of the response to a system
t	time
$x(t)$	time domain representation of the input for a single–input system
x_0	value of input used for Taylor expansion
$x_k(t)$	time domain representation of the k input component
y	output
$y(t)$	time domain representation of the output
$y_n(t)$	time domain representation of the nth order output component
$z(t)$	time domain representation of the input to the second port in a bivariate system
$z_k(t)$	time domain representation of the kth component of the input to the second port in a bivariate system
$\text{Im}\{\}$	Imaginary part of the expression in braces
$\text{Re}\{\}$	Real part of the expression in braces

List of Abreviations

VLSI	Very Large Scale Integration
LAN	Local Area Network
MESFET	Metal–Semiconductor Field–Effect Transistor
RC	Resistor–Capacitor
Hz	Hertz
KVL	Kirchhoff's Voltage Law
KCL	Kirchhoff's Current Law
dB	decibel
IPD	Intermodulation Product Description
DC	Direct Current
IF	Intermodulation Frequency
LO	Local Oscillator
RF	Radio Frequency
AOM	Arithmetic Operation Method
GPSA	Generalized Power Series Analysis
MLE	Maximum Likelihood Estimator
IP	Intermodulation Product
FIPD	Frequency Intermodulation Product Description
ASTAP	A Statistical Analysis Program
MHz	Megaherz
FFT	Fast Fourier Transform
SVD	Singular Value Decomposition
GHz	Gigahertz
TL	Transmission Line

Chapter 1

Introduction

1.1 Motivations and Objectives of This Study

The design complexity of analog circuits and systems is forever increasing. With this complexity comes the problem of simulation of these large analog systems. VLSI circuit dimensions are decreasing, thus requiring more accurate simulations and chip sizes are increasing, thus allowing larger, more complex circuits on a single chip. With VLSI technology, design iterations are costly both in time and money, and accurate simulation capability is a necessity for designing these large chips. Furthermore, when these chips are put together to form a complex analog system, the system design becomes complex, and system simulation even more demanding.

Circuit simulators such as SPICE [1] are impractical for system level simulation because the cost of simulations is $O(N^3)$ where N is the number of nodes. Recent advances such as waveform relaxation [2] have resulted in $O(N^\alpha)$ where α is typically 1.2 to 1.5. However, no such speed up of global simulation has been achieved for analog circuit functions.

A standard approach to system design is to identify the important characteristics of the subsystems, and define figures of merit for each subsystem. These figures of merit can be budgeted for the total system. For example, for a Local Area Network (LAN) using a token ring protocol, each station or subsystem contributes to the overall jitter of the signal as it travels around the ring. In order to assure that the system jitter is below some maximum, the maximum jitter contribution of a single station is restricted to the total jitter budget divided by the number of stations. This approach assumes that mechanics of how jitter is added is well understood. As designs become more complex, the accumulation characteristics of such figures of merit become more complicated and harder to predict. Identifying and accurately defining and budgeting these becomes more difficult.

System simulators such as SABER [3,4] address the problem of system simulation by providing the user with a language (e.g. MAST) to write behavioral models of large circuits or small subsystems. It is up to the designer to write and verify these models. The complexity of writing accurate models and the overhead and possible mistakes in verifying these models can render this method of system simulation unreliable.

The objectives of this study are to develop new ways to simulate nonlinear circuits and systems and to develop new ways to automatically extract the subsytem models to be used. This study is restricted to steady–state simulation, meaning all of the signals can be expressed as the sum of periodic signals.

1.2 Overview

The organization of this thesis is divided into six chapters. Chapter one is an introductory chapter outlining the entire thesis and stating the motivation, objectives, and original contributions of this study. Chapter two gives a review of the state of the art in steady–state behavioral modeling and simulation. In particular, modeling and simulation techniques that can be used to analyze the steady–state behavior of analog circuits and systems are reviewed. Also, the current technology in behavioral model extraction is reviewed.

Chapter 3 is a presentation of new modeling and simulation techniques that are restricted to steady–state simulation as they are based in the frequency domain. Special attention is paid to bivariate, or two input, models both as a behavioral black box (as in a ring diode mixer) and as a circuit element model (as in a current source modeling the drain–to-source current in a MESFET controlled by both the gate voltage and the drain voltage). Chapter 4 is a detailed description of a new model extraction technique for Volterra behavioral models. This technique is applicable to black box model extraction by both laboratory measurements and by device level simulation. Chapter 5 is a review of verification for the techniques presented in chapters 4 and 5 and include simulation results for a ring diode mixer, a MESFET amplifier, simple RC filter, a memoryless 7th order device, and a nonlinear equalization circuit. Chapter 6 concludes this work by summarizing and giving suggestions for further work.

1.3 Summary of Original Contributions

1.3.1 Published Work

- P.J. Lunsford, G.W. Rhyne, and M.B. Steer, "Frequency domain bivariate generalized power series analysis of nonlinear analog circuits," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-38, pp. 815–818, June 1990.

  (sections 3.2 and 5.2)

- P.J. Lunsford and M.B. Steer, "The Relationship Between Bivariate Volterra Analysis and Power Series Analysis with Application to the Behavioral Modeling of Microwave Circuits," *Int. J. on Microwave and Millimeter Wave Computer Aided Engineering*, vol. 1, no. 3, July 1991, pp. 253–262.

  (sections 3.3 and 5.3)

1.3.2 Unpublished Work

- Volterra Behavioral Model Extraction (chapter 4)
- Frequency Domain Harmonic Balance using Volterra Behaviorals (section 3.4 and appendix A)

Chapter 2

Review of Steady–State Behavioral Modeling and Simulation

2.1 Introduction

The current state–of–the–art of steady–state modeling and simulation of analog circuits and systems will be discussed in this chapter. More specifically, simulation and modeling techniques that are applicable to steady–state modeling are reviewed. In this context, the term "steady–state" refers to input and output signals that can be expressed as a sum of periodic signals. In general, a steady–state signal is not required to be periodic. For example, a signal consisting of the sum of two sinusoids, one at 1 Hz and one at $\sqrt{2}$ Hz would be a steady–state signal, but it would not be a periodic signal. However, most techniques discussed here are specific to periodic signals.

First several definitions of terms are discussed and remarks are made giving overall advantages and disadvantages of the techniques described by these terms. Secondly modeling and simulation techniques are reviewed, and lastly behavioral model extraction is reviewed.

2.2 Definitions & Issues

Several terms merit discussion. Here macromodeling versus behavioral modeling, circuit versus system simulation, and time versus frequency domain simulation are discussed.

Two terms that are sometimes used interchangeably but usually carry different meaning are "macromodel" and "behavioral model". A macromodel is a simplified circuit that represents the behavior of a more complex circuit. Macromodeling is commonly used for the simulation of complex circuits based on operational amplifiers [5]. A behavioral model, on the other hand, is usually defined as an input–output characteristic described by an equation, algorithm, or table, but a macromodel is also a behavioral model.

The terms "circuit simulation" and "system simulation" are also terms that carry different meaning. Circuit level simulation is a transistor level simulation where branches of a circuit topology are described to a simulator and the voltage-current constitutive relationship for each branch is known. The simulator then solves Kirchhoff's voltage and current laws (KVL and KCL) to predict the performance of the circuit. A system level simulator usually has an input where the branches describing the topology have direction. In other words, the connections to the primitive blocks in the system are either inputs or outputs. An output of one block connects to the input(s) of another block(s). No two outputs are connected together. In a system simulator, the effects of loading on a signal are not taken into account. Thus in a system simulator, the performance of a subsystem is dependent only on the signals of the input, not what is connected to the output. However, a circuit simulator can be used as a pseudosystem simulator by creating devices with the voltage-current relationship described by a behavioral model. Thus through the use of behavioral modeling, a circuit simulation can perform a mix of circuit and system simulation.

Other types of simulation (or design) techniques used to analyze high level systems are signal flow graphs [6, pp. 542–558], and spread sheet techniques that use defined figures of merit [7]. Signal flow analysis is confined to linear systems, and bugeting defined figures of merit requires a simplification and often linearization of the characteristics of the subsystems.

2.2.1 Time Domain vs Frequency Domain

Simulators capable of steady–state simulation can be divided into two types, time domain and frequency domain simulators. A time domain simulator is the most intuitive. The signals are expressed as values at points in time. On the other hand, a frequency domain steady-state simulator expresses signals as coefficients of a Fourier series or set of Fourier series. Usually the phasor form of these coefficients is used, thus each signal is a vector of complex numbers. Each complex number has an associated frequency. The magnitude and phase of the phasor corresponds to the magnitude and phase of the frequency component.

Compared to time–domain simulators, frequency domain simulators can be more efficient, especially for the case of a linear circuit or system. Moreover, nonperiodic steady–state signals can be more easily represented in the frequency domain than in the time domain. Circuits which have steady–state responses can typically be simulated several orders of magnitude faster. The simulation dynamic range (defined as the ratio of the magnitude of the largest sinusoidal signal component present in the circuit to the smallest component simultaneously present) exceeds 200dB for a frequency domain simulation [8] and approaches 400 dB for some specialized forms. On the other hand, the dynamic range for time–domain simulation can be as little as 60 dB. [9]. To put this in perspective, the accurate simulation of a 16 bit digital–to–analog converter requires that the dynamic range of the simulator be at least 100 dB. But time domain simulation is more intuitive and easier to integrate physical based semiconductor models. The transients related to non-periodic affects (e.g. start up) can also be analyzed with a time domain simulator and no such information is available with frequency domain simulators.

2.3 Modeling Techniques

For a simulator to be useful, it must be supplied with a description of the circuit or system that is to be simulated. Techniques of modeling are usually heavily dependent on the simulator that will be used. For instance, a frequency domain description of a component is more easily used in a frequency domain simulator.

2.3.1 Time Domain Modeling

Time domain modeling is reasonably intuitive since the time domain is the domain of physical reality. Other domains are just mathematical representations that enable analysis from different perspectives. For a macromodel used in circuit simulation, an input–output characteristic or current–versus–voltage characteristic can be described by a time domain equation or table, so that $y(t) = F(x(t))$ where F is some analytic function. A behavioral model for an idealized subsystem can then be the idealized function. For example, an ideal gain stage can be modeled by simply setting the output equal to the input multiplied by the gain. This simplified model neglects skew rate limitations and other distortion factors, but enables simple and efficient simulation. Behavior level time domain simulators such as SABER [3, 4] and PROFILE [10, 11] use specialized languages which enables the user to model arbitrary systems. Of course trade–offs between simplicity, efficiency, and accuracy of a model are made when creating the model. But each model must be coded by hand, and the resulting model is only as good as the understanding of the designer.

2.3.2 Generalized Power Series Modeling

Generalized power series analysis is a frequency domain nonlinear circuit simulation technique which utilizes generalized power series descriptions of nonlinear components. It was developed by Steer and Khan in 1983 [12] based on earlier work by Sea and Vacroux [13–15] and Heiter [16]. Examples of this simulation technique have been published for microwave circuit simulation [17–19]. The work described here forms one of the bases of the macromodeling techniques developed in this thesis.

Generalized power series requires that the constitutive relations (i.e. current versus voltage) of the nonlinear elements be described by a power series that can contain time delays. The steady–state input $x(t)$ is described by a series of sinusoids, or a vector of phasors. Each phasor X_k represents the amplitude and phase of the corresponding sinusoid.

$$x(t) = \sum_{k=0}^{K} \epsilon_k |X_k| \cos(\omega_k t + \phi_k) = \sum_{k=-K}^{K} X_k e^{j\omega_k t} \tag{2.1}$$

The general form of the output $y(t)$ (e.g. the current through a branch as a function of the voltage across the branch) is characterized by A, A_i, $a_{n,i}$, $b_{k,i}$ and $\tau_{k,n,i}$.

$$y(t) = A \sum_{i=1}^{I} A_i \sum_{n=0}^{\infty} a_{n,i} \left[\sum_{k=-K}^{K} b_{k,i} x_k(t - \tau_{k,n,i}) \right]^n \tag{2.2}$$

The resulting output phasor Y_q at frequency ω_q is given by

$$Y_q = \sum_{n=0}^{\infty} \sum_{\underbrace{n_1,\ldots,n_K}_{|n_1|+\cdots+|n_K|=n}} \mathrm{Re}\left\{\epsilon_n T\right\}_{\omega_q} \tag{2.3}$$

where the output frequency is given by

$$\omega_q = \sum_{k=1}^{K} n_k \omega_k \tag{2.4}$$

and T and Φ are defined by

$$T = \sum_{\alpha=0}^{\infty} \sum_{\underbrace{s_1,\ldots,s_K}_{s_1+\cdots+s_K=\alpha}} (n+2\alpha)!\; a_{n+2\alpha}\Phi \tag{2.5}$$

$$\Phi = \prod_{k=1}^{K} \frac{(b_k)^{n_k+2s_k}(X_k^{\dagger})^{|n_k|+s_k}(X_k^{\ddagger})^{s_k}(,_k^{\dagger})^{|n_k|+s_k}(,_k^{\ddagger})^{s_k}}{s_k!(|n_k|+s_k)!} \tag{2.6}$$

$$,_{k,n} = \mathrm{e}^{-j\omega_k\tau_{k,n}} \tag{2.7}$$

Using Kirchhoff's current and voltage laws, the harmonic balance system of equations can be solved using iterative techniques such as Newton-Raphson. The key to this technique is the efficiency of the nonlinear calculation. For a given set of frequencies, an IPD (Intermodulation Production Description) table is created during the initialization of the simulation. Each entry in this table is a valid set of n_k (2.4). This table is referred to during each iteration. The table can also be saved in a file to be used in subsequent simulations that use the same set of frequencies and that contain elements with the same maximum order of the power series. For example, figure 2.1 shows a simple spectrum of a mixer circuit.

A partial list of the IPD table for this spectrum is given in table 2.1. The n and n_1, n_2, and n_3 variables are used in (2.3). Note that zero Hz (DC) is not an input that is mixed with the other frequencies. The only DC input component is when $n = 0$, or the DC output value for zero input. This is because we can simplify the mixing calculations by changing the form of the polynomial. For instance, take the polynomial of x

$$y = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n \tag{2.8}$$

If the DC input to the input x is given by x_0, then (2.8) can be expressed in the form

$$y = b_0 + b_1(x-x_0) + b_2(x-x_0)^2 + \ldots + b_n(x-x_0)^n \tag{2.9}$$

where

$$b_i = a_i + a_{i+1}x_0 + a_{i+2}x_0^2 + \ldots + a_nx_0^n \tag{2.10}$$

This recalculation of the coefficients is inexpensive and reduces the number of IPD's needed to calculate the output.

2.3.3 Power Series Ratios

In 1989, C.R. Chang introduced a new technique to calculate power series in the frequency domain [20]. Termed the Arithmetic Operator Method or AOM, this technique maps the time domain arithmetic operation into frequency domain matrix

Output Frequency	n	n_1	n_2	n_3
f_1, IF	1	1	0	0
	2	0	1	-1
	4	2	-1	1
	5	-1	2	-2
	7	3	-2	2
	8	-2	3	-3
f_2, LO	1	0	1	0
	2	1	0	1
	4	-1	2	-1
	5	2	-1	2
	7	-2	3	-2
	8	3	-2	3
f_3, RF	1	0	0	1
	2	-1	1	0
	4	1	-1	2
	5	-2	2	-1
	7	2	-2	3
	8	-3	3	-2
f_4, DC	0	0	0	0
	3	1	-1	1
	6	2	-2	2

Table 2.1: Partial listing of IPD's when DC is not an input to the algebraic formula

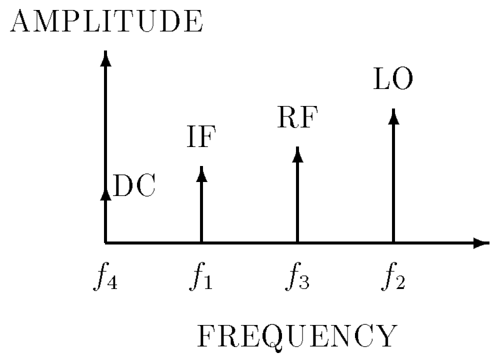

Figure 2.1: Simplified spectrum for a mixer circuit ($f_1 = 2f_2 - f_3$)

operations. Since, for example, multiplication in the time domain corresponds to a matrix multiplication in the frequency domain, division in the time domain can be mapped to matrix inversion and multiplication in the frequency domain. Thus saturating curves such as $\tanh(x)$ can be efficiently modeled by a ratio of power series.

$$y = \frac{a_0 + a_1 x + a_2 x^2 + a_3 x_3 \ldots}{b_0 + b_1 x + b_2 x^2 + b_3 x_3 \ldots} \tag{2.11}$$

This ability to allow division in the model algebra reduces the number of terms needed for many models, especially models that have a saturating behavior similiar to $\tanh(x)$. With GPSA, saturating curves such as $\tanh(x)$ require many terms. Thus AOM increases the efficiency and extends the modeling capability of GPSA.

2.3.4 Volterra Series

In the late 1800's and early 1900's Volterra developed functional series, now known as Volterra series [21, 22]. A summary of this work is given in [22]. This has provided a basis for modeling and simulation in a wide variety of disciplines. It has been used in the field of biology to model the thermal dependency of the heart rate [23], and the properties of the auditory system [24, 25]. In the field of nonlinear hydrodynamics, it has been used to study ship and platform motion and stabilization [26–32]. In addition to hydrodynamics, other mechanical engineering problems have been studied using Volterra series including elastomer properties [33] and structural mechanics [34–36]. It has been used to study the optical [37] and magnetic [38] properties of materials. The input–output properties of a laser diode have also been modeled [39].

Volterra series has been extensively used in circuit design, often tailored for a specific application. For example, oscillators [40, 34], power conversion systems [41], transconductance–capacitor filters [42], operational amplifiers [43, 44], diode mixers [45,46], hard-limiters [47], radio receivers [48], and traveling–wave tubes [49] have all been analyzed using Volterra series. However, the most extensive use of Volterra series has been in analyzing microwave amplifiers [50–55]. There has also been a vast amount of theoretical research on Volterra series. Most of this deals with convergence properties [56, 57], conditions where Volterra series exist [58–62], techniques to analytically derive Volterra descriptions for a given system [63], and relationships to other mathematical theory [64–68]. There has also been work in developing insight and techniques that will allow Volterra theory to be used in the design of nonlinear controls [69–71]. The frequency domain form of the Volterra Series forms the basis for behavioral modeling techniques developed in this thesis.

Time Domain Volterra Series

Time continuous time–domain expression for the output $y(t)$ of a function described by a Volterra series with input $x(t)$ is

$$y(t) = \sum_{n=0}^{\infty} y_n(t) \tag{2.12}$$

where

$$y_n(t) = \int_{\tau_n=-\infty}^{+\infty} \int_{\tau_{n-1}=-\infty}^{+\infty} \cdots \int_{\tau_1=-\infty}^{+\infty} h_n(\tau_1, \tau_2, \ldots \tau_n) \left[\prod_{i=1}^{n} x(t-\tau_i) \right] d\tau_1 \, d\tau_2 \ldots d\tau_n \tag{2.13}$$

The nth order Volterra kernel is $h_n(\tau_1, \tau_2, \ldots \tau_n)$ and is not unique unless some further restriction is applied. Requiring that h_n be symmetric can be done without loss of generality and results in a unique set of Volterra kernels for a given system. This restriction is usually applied and the following discussion assumes that the kernels are symmetric.

Frequency Domain Volterra Series

The n–fold Fourier transform of the Volterra kernel is known as the Volterra nonlinear transfer function, H_n.

$$\begin{aligned} H_n(f_1, f_2, \ldots f_m) = & \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} \\ & h_n(\tau_1, \tau_2, \ldots \tau_m) \\ & e^{-j2\pi(f_1\tau_1 + f_2\tau_2 + \ldots f_n\tau_n)} \\ & d\tau_1 \, d\tau_2 \ldots d\tau_n \end{aligned} \tag{2.14}$$

Using a Fourier transform, (2.13) can be used to give the frequency domain description of the nth order output $Y_n(f)$

$$\begin{aligned} Y_n(f) &= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \dots \int_{-\infty}^{+\infty} H_n(f_1, f_2, \dots f_m) \\ &\quad \delta(f - f_1 - f_2 - \dots f_n) \\ &\quad \left[\prod_{i=1}^{n} X(f_i) \right] df_1\, df_2 \dots df_n \end{aligned} \tag{2.15}$$

Figure 2.2 gives a graphical representation of the frequency domain calculation of the output. The output of each order calculation is summed to give the output of the system. The output of each order calculation is calculated from the frequency domain description of the input and the Volterra nonlinear transfer function corresponding to that order. The 0th order calculation is just the DC offset for zero input and is represented by the value of the 0th order Volterra nonlinear transfer function H_0.

2.4 Simulation

Three different simulation techiques can be used for steady–state simulation. They are shooting methods, harmonic balance, and the Volterra method of nonlinear currents.

2.4.1 Shooting Methods

Shooting methods [72–77] are restricted to periodic simulation and are based on time domain simulation. Since a time domain simulator can simulate a period of time equal to the period of the signal, the only remaining problem is to find the initial condition such that the initial state of the circuit is the same as the ending state of the circuit. In other words, there is a boundary condition that all signals at the end of the period simulation must be equal to the value of the signals at the beginning of the simulation. For a circuit simulator, the value of all of the signals can usually be uniquely determined by the value of all of the capacitor voltages and inductor currents. For the special case of a circuit with a transmission line, the problem becomes much more difficult because the state of a transmission line requires a continuous representation. Thus the initial state of the circuit must include a vector of time samples representing the continuous waves traveling in the transmission line. This makes shooting methods impractical for most circuits that contain transmission line models.

2.4.2 Harmonic Balance

Harmonic balance simulators [8, 78] work in the frequency domain and thus signals are expressed as vectors of phasors. Each phasor has a corresponding frequency, and this list of frequencies is usually part of the specification of the simulation. In other words, for the simulation to be accurate, one must know *apriori* the discrete values of the significant frequencies that will be present in the circuit.

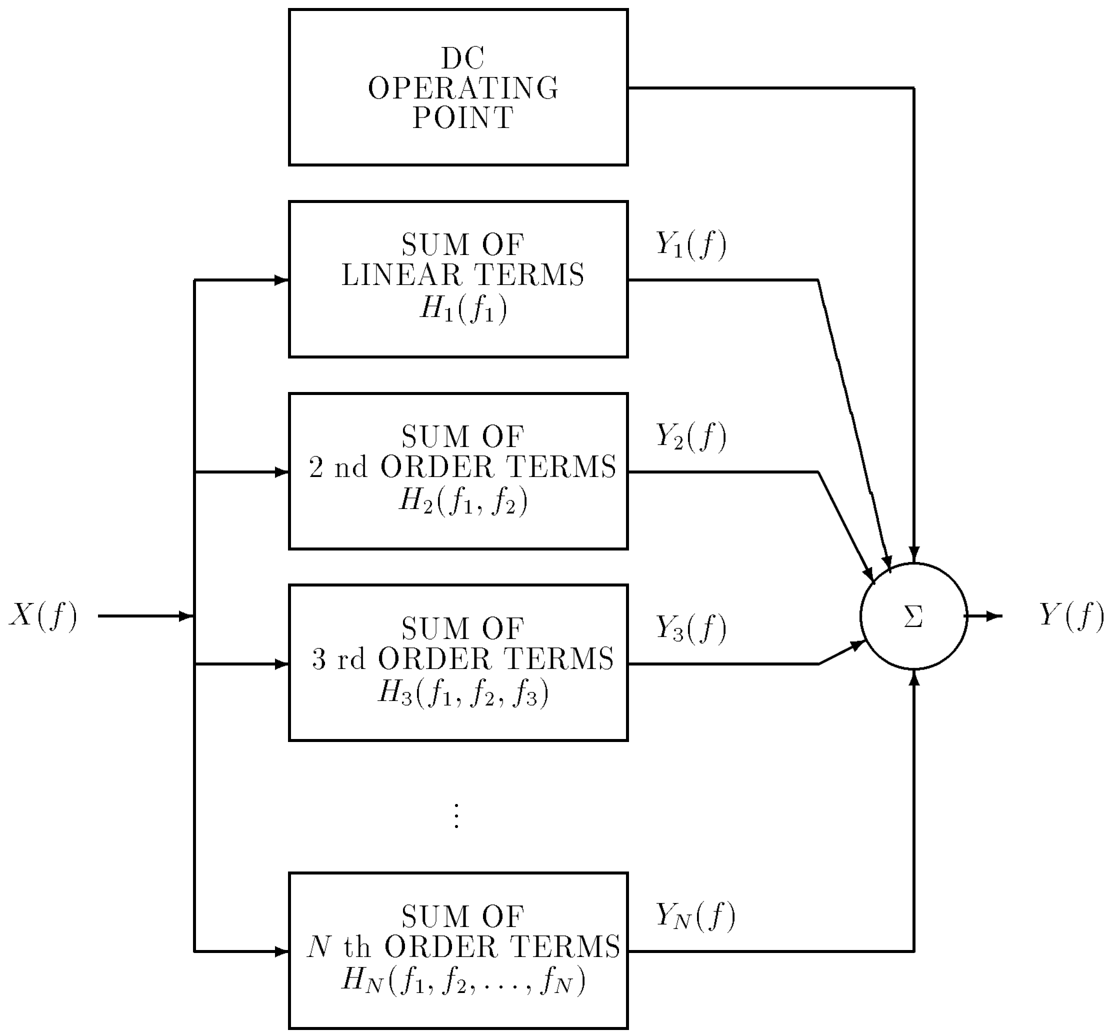
DC OPERATING POINT
SUM OF LINEAR TERMS $H_1(f_1)$
SUM OF 2 nd ORDER TERMS $H_2(f_1, f_2)$
SUM OF 3 rd ORDER TERMS $H_3(f_1, f_2, f_3)$
SUM OF N th ORDER TERMS $H_N(f_1, f_2, \ldots, f_N)$
$X(f)$
$Y_1(f)$
$Y_2(f)$
$Y_3(f)$
$Y_N(f)$
Σ
$Y(f)$

Figure 2.2: Illustration of nonlinear analysis using Volterra series.

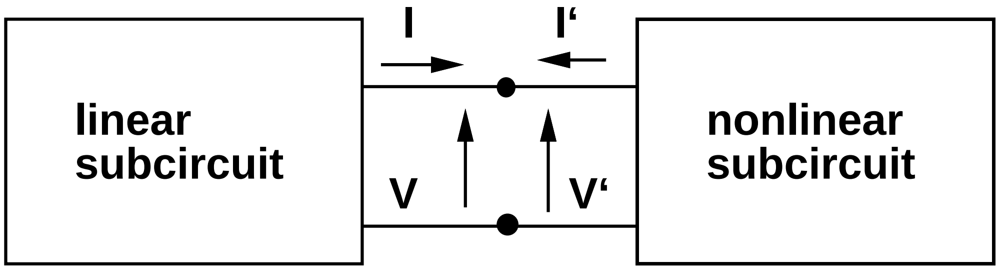

Figure 2.3: Linear–Nonlinear Partitioning of Circuit

The overall circuit solution technique of harmonic balance simulators can be seen in figure 2.3. The circuit is grouped into two subcircuits. One subcircuit contains only linear elements, and the other subcircuit contains only nonlinear elements. The error signal that is minimized during the solution iteration is defined by the voltages and currents at this interface.

$$E = \sum_{k=0}^{N} |I_{\omega k} + I'_{\omega k}|^2 + |V_{\omega k} - V'_{\omega k}|^2 \tag{2.16}$$

2.4.3 Method of Nonlinear Currents

The method of nonlinear currents is a simulation technique based on the mathematics of Volterra series that allows direct calculation of the frequency response of nonlinear curcuits where the consitutive relationships of the nonlinear elements are described by a power series [79, pp 190–207] [47, 80–87]. This method is particularly efficient when the nonlinearity is weak, and has been shown to be efficient in calculating the distortion in weakly nonlinear microwave circuits.

2.5 Behavioral Model Extraction

As analog circuits and systems get larger, the need for more complex simulations increases. One way to simplify the simulation is to create a behavioral model for portions of the sytem, and use this simplified model in simulation. However, the problem of creating the behavioral model is not simple. Even if the form or topology of the model has been chosen, the specific model parameters must be created or extracted in some way. The creation or extraction of a specific model depends on the simulator that is being used. This section will discuss several aspects of behavioral model extraction, using a written modeling language, using error minimization, using the maximum likelihood estimator, and extracting a general Volterra series model.

2.5.1 Hand Written in Modeling Language

Time domain circuit simulators that are tailored for behavioral simulation like SABER [4] and PROFILE [10,11] use a structured language to describe the model. The most common behavioral blocks are assumed to be linear and are easy to characterize. Moreover, many circuits are intentionally designed to be linear, therefore the assumption of linearity is as good as the design of the subcircuit being modeled. Furthermore, sometimes the model can be an idealization of the intended function of the intended circuit, thus no extraction is needed. Moreover, saturation effects can be simply modeled by using thresholding functions.

2.5.2 Error Minimization

Most model extraction can be thought of as an error minimization problem. Assuming that the topology of the model has been determined (e.g. the form of the equation used to model the input–output behavior) then the known behavior of the circuit can be compared to the modeled behavior of the circuit. The difference in the behavior can be thought of as the error, e, of the model, and the parameters, P, of the model can be varied until the error is as small as possible.

$$e = f(P) \tag{2.17}$$

Linear Least Square Fit

If the relationship between the parameters $P = [p_1, p_2 \dots p_N]$ and the behavior of the circuit can be expressed in the matrix form of (2.18), then the parameters P can be directly calculated [88, pp. 194–199]. This assumes that N is less than or equal to M, and the error to be minimized is the L_2 norm of the difference between the left and right side of equation 2.18. This is a common method for extracting parameters but is not always applicable because equation (2.18) cannot be formulated or because the resulting error is inappropriate.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ a_{31} & a_{32} & \dots & a_{3N} \\ \vdots & \vdots & \vdots & \vdots \\ a_{M1} & a_{M2} & \dots & a_{MN} \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_M \end{bmatrix} \tag{2.18}$$

Nonlinear Minimization with derivative information

Where the error minimization problem cannot be set up as a linear problem, nonlinear minimization must be used. Several methods of nonlinear minimization use derivative information. These methods only find a local minimum in the parameter space, but are relatively fast and efficient. The major difficulty is calculating the derivative information.

Nonlinear Minimization without derivative information – Simulated Annealing

There are also nonlinear minimization techniques which do not require derivative information. The main advantage of these techniques is that they tend to find the global, instead of local, minimum of the error function. These are usually termed "annealing" techniques because the algorithm has an analogy in the physics of a metal being annealed [89]. The error value is analogous to the total energy of the metal being annealed. As a metal cools, each molecule has a probability of changing its energy level. The change of the state usually results in a decrease in energy. But in some cases, the energy can increase, and the probability of an increase in energy is greater at higher temperatures. Thus the cooling schedule used is an important factor in the efficiency and the quality of the resulting solution. A fast cooling schedule is quicker but less likely to find the global minimum.

Conventional simulated annealing is not directly applicable to a continuous parameter space, the technique called tree annealing developed by Bilbro *et al* [90] is very efficient. The history of previous points in the parameter space is used to bias the random choices of possible better solutions. This information is stored in a multi–dimensional binary tree, and thus the term tree annealing. As with all nonlinear minimization algorithms, the user must specify a range of the valid parameter space. Tree annealing is efficient at finding a global minimum, especially if an appropriate cooling schedule is used and the calculation of the error function is not too expensive.

2.5.3 Maximum Likelihood Estimator

A recent development in parameter extraction that deserves mention is the use of the maximum likelihood estimator (MLE) [91–93]. This technique is based on the mathematics of Volterra series and has the advantage of minimizing the effect of gaussian white noise present in the system. Noise is always a factor that can affect the accuracy of the extracted model. MLE assumes that random noise is added to the input and output of the system due to the measurement technique. The system is first characterized by measuring the noise characteristics of the input and output with no signal applied. Thus MLE is applicable only to models based on measurements taken in a laboratory, not simulated measurements derived by simulation. After the zero–input noise is characterized, the output is measured with an input stimulus. The problem is then formulated, "what is the most likely true input (the intended input minus the noise added) and true output (the measured output minus the noise added) for this system with the known intended input and measured output". This formulation has the advantage of minimizing the effects of noise coupled into the system, but is expensive because the true input and output are now unknown. Thus both the parameter space, and the input–output space must be solved simultaneously. This translates into a larger number of unknowns. Thus MLE is especially useful in parameter extraction where the model has a small number of unknown parameters and where measurement noise is significant.

2.5.4 Volterra Series Model Extraction

General behavioral models based on Volterra series can be used to accurately represent many different sytems [56]. The general Volterra model is a behavioral model where the input–output characteristics are described by a Volterra series as given in (2.13) or (2.15). The problem of extracting, storing, and using these models presents difficulties. Since a general Volterra model consists of continuous functions, either the time–domain kernels, or the frequency–domain nonlinear transfer functions, the range and method of storage must be determined. Typically, discrete points are selected. Since the time domain kernels are integrated over time, the equivalent of the entire kernels must be saved. For the case of the frequency–domain nonlinear transfer function, a limited set of points can be saved if the simulation frequencies are known in advance. The amount of data needed to store the model is heavily dependent on the order of the model assumed. The higher the order, the more data needed. Both the Volterra kernels and the nonlinear transfer functions have n inputs (where n is the order of the kernel or nonlinear transfer function). Thus a small increase in the order of the model can vastly increase the amount of data that needs to be stored. For most applications, the model must be extracted by exciting the system with several different inputs and measuring the responses. If there is a system of equations already describing the system, the Volterra series can be calcualted directly [63], but in that case, it would usually be more efficient to base the behavioral model directly on the known set of equations.

White Noise Probing for the Development of Functional Models

The idea of using noise as a probing input to extract Volterra models of a system was first suggested by N. Wiener in the 1940's [94]. To simplify the procedure, Wiener developed G-Functionals which are based on Volterra functionals (i.e. Volterra series) but unlike Volterra functionals, G-Functionals are orthogonal with respect to gaussian white noise. Unlike G-Functionals responses, Volterra functional responses are homogeneous of degree n, where n is the order of the functional. This extraction procedure was further developed and demonstrated on various circuits in the 1960's by Lee and Schetzen [95, 96] used in the 1960's for various circuits and systems. In the 1980's this method gained interest again [97–100] and gaussian noise has been used to extract Volterra based models for electrical systems [101] structural components [102, 103] vertebrate photoreceptors [104] and other biological and physiological systems [105] .

The method is appealing because of the simplicity of the measurement. Only a noise generator, variable delays, multipliers, and an averager are needed to extract the models. These procedures, however, are time consuming and certain points are difficult to obtain, namely $h_n(\tau_1, \tau_2 \ldots \tau_n)$ where $\tau_i = \tau_j$ for all $i \neq j$.

Two–tone probing

A second way to extract a Volterra representation of a system is to use two–tone probing [106, 107]. This is very simple and straightforward, but it requires that the system be of order two, and thus is only applicable to a small set of systems. Since the system is of order two, when the system is excited by the sum of two sinusoids the output spectrum has a limited number of discrete components, and each component characterizes a single point in one of the Volterra nonlinear transfer functions.

Quick Method

In order to maximize the information derived from a single steady–state measurement, Boyd *et al* developed the so called quick method for measuring second order Volterra kernels [108]. The procedure uses an input of the sum of several sinusoids. The relative frequencies are very carefully chosen so that most of the output frequencies are associated with only one point on a Volterra nonlinear transfer function. The method requires very careful control of the complex input signal, but the output measurement and the calculation of the Volterra nonlinear transfer functions is straightforward. The system is not, however, applicable to extraction based on shooting method circuit simulation because of the extremely long periods of the input signal. Work by Chua and Liao published in 1989 [109] extended this work to higher order measurements. Although the theory presented is for any arbitrary order, the authors show data for third order extraction and report that this method is practical only up to fourth order.

This method is good for finding the general shapes of the nonlinear transfer functions, but it is limited since there are several points that cannot be measured directly from a single measurement. For instance, if the system is of order 3 or higher, all points on the first order transfer function cannot be extracted from a single measurement. The output frequencies associated with the first order transfer functions will always have components associated with the third order transfer function. For example, if the input spectrum contains a 4 Hz sinusoid, then in general, the output will contain a 4 Hz sinusoid. Part of the sinusoid will be assiciated with the point $H_1(4)$, but part will also be associated with the point $H3(-4, 4, 4)$.

In order to separate the different orders associated with the same output frequencies, several measurements can be taken using the same input frequencies and the same relative amplitudes at each frequency, but the total amplitude of the signal is varied. Since the nth order response y_n is homogeneous of degree n, then $x(t) \to y_n(t)$ implies $\alpha x(t) \to \alpha^n y_n(t)$. Thus the measurement is taken m times, each time with the input x multiplied by a scalar α_i. If r_i is the measured response for $\alpha_i x(t)$, then the nth order response y_n can be calculated by solving (2.19) in the least square error sense. This method of separation previously has been used for second order separation and will be used for higher order as part of the extraction method proposed in chapter 4.

$$\begin{bmatrix} \alpha_1 & \alpha_1^2 & \dots & \alpha_1^N \\ \alpha_2 & \alpha_2^2 & \dots & \alpha_2^N \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_m & \alpha_m^2 & \dots & \alpha_m^N \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ \vdots \\ r_m \end{bmatrix} \qquad (2.19)$$

2.6 Conclusion

The current state of the art behavioral simulation techniques are usually based on an idealized time domain model. Volterra series techniques offer the generality needed to model arbitrary non-autonomous causal systems, but techniques for model extraction and efficient simulation are lacking. Most system level simulation today is done using time domain simulators that require the equivalent of simplified macromodels. Although the modeling languages for these simulators provide general modeling capability, model synthesis and extraction is specific to the subsystem being modeled and requires extensive knowledge of the subsystem.

Some of the work described in this chapter forms the basis of the behavioral modeling technique discussed in later chapters. These are:

- Generalized Power Series indexing techniques
- Error Minimization with tree annealing
- Frequency domain Volterra models
- Two tone probing for Volterra model extraction
- Quick Method for Volterra model extraction

Chapter 3

Behavioral Modeling and Simulation

3.1 Introduction

In order to simulate large circuits efficiently, the circuit must be partitioned into smaller subcircuits, and then each subcircuit characterized or modeled. The behavioral model of a subcircuit needs to have sufficient generality so that the subcircuit can be modeled to the desired accuracy. Volterra techniques provide a mathematical base that will enable a large class of input-output behaviors to be modeled. It also provides a built–in measure of complexity, the order of the model, that can be increased to improve the accuracy of the model, or decreased to improve the computation speed of the model.

This chapter discusses the relationship of power series modeling to the more general case of Volterra series modeling. This relationship is important for several reasons. First, since the present state of the art includes circuit simulation based on power series modeling, this relationship gives us a foundation to extend this simulation technique to the more general case of models based on Volterra series. Secondly, since techniques already exist to extract power series models with order dependent delays, a way to translate these existing power series models into Volterra models is provided. Lastly, by understanding the relationship between power series and Volterra series, we can gain insight into the limitations of the present power series analysis.

This chapter first introduces power series analysis for the general case of a two–input one–output system. This work gives the basis for using two–input models in a power series based simulator. Second, the relationship between power series analysis and Volterra series analysis will be shown. The univariate (single input) case of the output described by a power series is derived first to an alternate form. Then the univariate Volterra description will also be shown in that same form, thus providing a means to derive the direct equation for the Volterra nonlinear transfer function given that the power series description is known. The same is then done for the bivariate (two–input) case. First deriving a form for the output of a bivariate power series, and then by deriving the same form for a bivariate Volterra series, a direct equation for the bivariate Volterra nonlinear transfer function as a function of the bivariate power series description will be given. Using the derived relationship, the differences and limitations of power series and Volterra series modeling is discussed. Although the univariate relationship has previously been established [110], the method used did not lend itself to the bivariate case.

The second section of this chapter shows how the indexing scheme for power series simulation can be modified so that Volterra descriptions can be used. First the simpler case of a univariate description is given, and then the more complicated case of a bivariate description. This modification provides the basis of a general Volterra simulator derived from an existing power series simulator, thus allowing a practical implementation of Volterra simulator.

3.2 Bivariate Generalized Power Series

This section presents the development of an algorithm for determining the steady–state frequency domain description of the output of a system described by a bivariate (i.e. two independent inputs) power series. This power series can have complex coefficients and frequency–dependent time delays. Since the development is restricted to the steady–state response of the system, we can restrict the inputs to be represented as the sums of sinusoids.

A nonlinear element or system having the two multifrequency inputs, $x(t)$ and $z(t)$, (each having N components)

$$x(t) = \sum_{k=1}^{N} x_k(t) = \sum_{k=1}^{N} \mid X_k \mid \cos(\omega_k t + \phi_k) \tag{3.1}$$

and

$$z(t) = \sum_{k=1}^{N} z_k(t) = \sum_{k=1}^{N} \mid Z_k \mid \cos(\omega_k t + \theta_k) \tag{3.2}$$

can be represented by the bivariate generalized power series

$$y(t) = \sum_{\sigma=0}^{\infty} \sum_{\rho=0}^{\infty} a_{\sigma,\rho} E(\sigma, x) F(\rho, z) \tag{3.3}$$

with

$$E(\sigma, x) = \left(\sum_{k=1}^{N} b_k x_k(t - \tau_{k,\sigma}) \right)^{\sigma} \tag{3.4}$$

and

$$F(\rho, z) = \left(\sum_{k=1}^{N} d_k z_k(t - \lambda_{k,\rho}) \right)^{\rho} . \tag{3.5}$$

In these expressions, $a_{\sigma,\rho}$ is a complex coefficient, b_k and d_k are real, and $\tau_{k,\sigma}$ and $\lambda_{k,\rho}$ are time delays that depend on both the order of the power series and the index of the input frequency components. With these parameters a large variety of systems can be modeled. Our aim is to rewrite (3.3) in terms of phasors. The x input can be expressed as

$$\begin{aligned} x_k(t - \tau_{k,\sigma}) &= \mid X_k \mid \cos(\omega_k t + \phi_k - \omega_k \tau_{k,\sigma}) \\ &= \frac{1}{2} X_k , \, {}_{k,\sigma} \mathrm{e}^{j\omega_k t} + \frac{1}{2} X_k^* , \, {}^{*}_{k,\sigma} \mathrm{e}^{-j\omega_k t} \end{aligned} \tag{3.6}$$

where X_k is the phasor of the component of $x_k(t)$ at the radian frequency ω_k and

$$, \, {}_{k,\sigma} = \mathrm{e}^{-j\omega_k \tau_{k,\sigma}} \tag{3.7}$$

Similarly, for the other input $z(t)$, we write

$$\begin{aligned} z_k(t-\lambda_{k,\rho}) &= |\,Z_k\,|\cos(\omega_k t + \theta_k - \omega_k\lambda_{k,\rho}) \\ &= \frac{1}{2}Z_k\Upsilon_{k,\rho}\mathrm{e}^{j\omega_k t} + \frac{1}{2}Z_k^*\Upsilon_{k,\rho}^*\mathrm{e}^{-j\omega_k t} \end{aligned} \tag{3.8}$$

where Z_k is the phasor of the component of $z_k(t)$ at the radian frequency ω_k and

$$\Upsilon_{k,\rho} = \mathrm{e}^{-j\omega_k\lambda_{k,\rho}} \tag{3.9}$$

Using the multinomial expansion theorem [111], we write (3.4) as

$$\begin{aligned} E(\sigma,x) &= \sum_{\substack{\underbrace{l_1,\ldots,l_N,m_1,\ldots,m_N} \\ l_1+\cdots+l_N+m_1+\cdots+m_N=\sigma}} \left\{ \left[\exp\left(j\sum_{k=1}^{N}(l_k-m_k)\omega_k t\right)\right]\sigma! \right. \\ &\qquad \left. \times \prod_{k=1}^{N}\left(\frac{(\frac{1}{2}b_k)^{l_k+m_k}(X_k)^{l_k}(X_k^*)^{m_k}(\ _{k,\sigma})^{l_k}(\ _{k,\sigma}^{*})^{m_k}}{l_k!\ m_k!}\right) \right\} \end{aligned} \tag{3.10}$$

where the summation is over all combinations of the integers $l_1,\ldots,l_N,m_1,\ldots,m_n$ such that $\sum_{k=1}^{N} l_k+m_k=\sigma$. Similarly, we can expand F in (3.5), and so the product of E and F becomes

$$E(\sigma,x)\,F(\rho,z) = \sum_{\substack{\underbrace{\substack{l_1,\ldots,l_N,m_1,\ldots,m_N \\ i_1,\ldots,i_N,j_1,\ldots,j_N}} \\ l_1+\cdots+l_N+m_1+\cdots+m_N=\sigma \\ i_1+\cdots+i_N+j_1+\cdots+j_N=\rho}} \left[\exp\left(j\sum_{k=1}^{N}(l_k+i_k-m_k-j_k)\omega_k t\right)\right]\sigma!\rho!\Psi \tag{3.11}$$

where

$$\Psi=\prod_{k=1}^{N}\frac{(\frac{1}{2}b_k)^{l_k+m_k}(\frac{1}{2}d_k)^{i_k+j_k}(X_k)^{l_k}(X_k^*)^{m_k}(Z_k)^{i_k}(Z_k^*)^{j_k}(\ _{k,\sigma})^{l_k}(\ _{k,\sigma}^{*})^{m_k}(\Upsilon_{k,\rho})^{i_k}(\Upsilon_{k,\rho}^*)^{j_k}}{l_k!\ m_k!\ i_k!\ j_k!} \tag{3.12}$$

and the above summation is over all combinations of the nonnegative integers l, m, i, and j such that $\sum_{k=1}^{N} l_k+m_k=\sigma$ and $\sum_{k=1}^{N} i_k+j_k=\rho$. As with the single-variable power series, the frequency of each component is

$$\omega = \sum_{k=1}^{N} n_k\omega_k \tag{3.13}$$

where n_k is a set of integers, an intermodulation product description (IPD) where $n_k = l_k+i_k-m_k-j_k$. The intermodulation order is n, where $n=\sum_{k=1}^{N}\mid n_k\mid$.

Letting $(p_k + q_k)$ equal the larger of $(l_k + i_k)$ and $(m_k + j_k)$, and $(r_k + s_k)$ equal the smaller, we have $p_k + q_k - r_k - s_k = \mid n_k \mid$ where p_k, q_k, r_k, $s_k \geq 0$.

For a given set of n_k's specifying an individual intermodulation product (IP), the relevant components of $E(\sigma, x)F(\rho, z)$ can be written as the sum of two terms (for $n \neq 0$)

$$(\frac{1}{2}C)\mathrm{e}^{j\omega_q t} + (\frac{1}{2}C)^*\mathrm{e}^{-j\omega_q t} = \left\{ \begin{array}{ll} (\frac{1}{2}U'_q)\mathrm{e}^{j\omega_q t} + (\frac{1}{2}U'_q)^*\mathrm{e}^{-j\omega_q t} & \text{for } \omega_q \neq 0 \\ U'_q & \text{for } \omega_q = 0 \end{array} \right. \tag{3.14}$$

where

$$C = 2 \sum_{\substack{\underbrace{\substack{p_1, \ldots, p_N, r_1, \ldots, r_N \\ q_1, \ldots, q_N, s_1, \ldots, s_N}} \\ p_1 + \cdots + p_N + r_1 + \cdots + r_N = \sigma \\ q_1 + \cdots + q_N + s_1 + \cdots + s_N = \rho \\ p_k + q_k - r_k - s_k = \mid n_k \mid}} \left\{ \left(\frac{1}{2}\right)^{\sigma+\rho} \sigma!\, \rho!\, \Phi \right\} \tag{3.15}$$

and with

$$\Phi = \prod_{k=1}^{N} \frac{(b_k)^{p_k+r_k}(d_k)^{q_k+s_k}(X_k^\dagger)^{p_k}(X_k^\ddagger)^{r_k}(Z_k^\dagger)^{q_k}(Z_k^\ddagger)^{s_k}(,^\dagger_{k,\sigma})^{p_k}(,^\ddagger_{k,\sigma})^{r_k}(\Upsilon^\dagger_{k,\rho})^{q_k}(\Upsilon^\ddagger_{k,\rho})^{s_k}}{p_k!\; r_k!\; q_k!\; s_k!}. \tag{3.16}$$

Here we define

$$X_k^\dagger = \left\{ \begin{array}{ll} X_k & \text{for } n_k \geq 0 \\ X_k^* & \text{for } n_k < 0 \end{array} \right. \tag{3.17}$$

(where X_k^* denotes the complex conjugate of X_k) and

$$X_k^\ddagger = \left\{ \begin{array}{ll} X_k^* & \text{for } n_k \geq 0 \\ X_k & \text{for } n_k < 0 \end{array} \right. \tag{3.18}$$

($Z_k^\dagger$, $Z_k^\ddagger$, $,^\dagger_{k,\sigma}$, $,^\ddagger_{k,\sigma}$, $\Upsilon^\dagger_{k,\rho}$, and $\Upsilon^\ddagger_{k,\rho}$ are similarly defined.) Thus,

$$U'_q = \left\{ \begin{array}{ll} C & \text{for } \omega_q \neq 0 \\ \frac{1}{2}(C + C^*) = \mathrm{Re}(C) & \text{for } n \neq 0 \text{ and } \omega_q = 0 \end{array} \right. \tag{3.19}$$

Note that U'_q is the contribution to $E(\sigma, x)\,F(\rho, y)$ of one intermodulation product (IP). The two terms in (3.14) occur as for $n \neq 0$, p_k and q_k replace two sets of i_k, j_k, l_k, and m_k, one set corresponding to $(l_k + i_k) > (m_k + j_k)$ resulting in the $\mathrm{e}^{j\omega_q t}$ term and a set corresponding to $(l_k + i_k) < (m_k + j_k)$ resulting in the $\mathrm{e}^{-j\omega_q t}$ term. For $n = 0$ there is only one set corresponding to $(l_k + i_k) = (m_k + j_k)$. Thus, the U'_q expression is one-half that in (3.19) for the case $n = 0$. For (3.19) to hold we make the restriction that no IPD be equal to the negative of another IPD. If U_q is the component of Y due to a single intermodulation product then

$$U_q = \sum_{\sigma=0}^{\infty} \sum_{\rho=0}^{\infty} a_{\sigma,\rho} U'_q. \tag{3.20}$$

Using the Neumann factor, ϵ_n ($\epsilon_n = 1$, $n = 0$; $\epsilon_n = 2$, $n \neq 0$),

$$U_q = \mathrm{Re}\left\{\epsilon_n T\right\}_{\omega_q} \tag{3.21}$$

where Re$\{\ \}_{\omega_q}$ is defined such that it is ignored for $\omega_q \neq 0$ but for $\omega_q = 0$ the real part of the expression in brackets is taken. In (3.21)

$$T = \sum_{\alpha=0}^{\infty} \sum_{\substack{\underbrace{\substack{p_1,\ldots,p_N,r_1,\ldots,r_N \\ q_1,\ldots,q_N,s_1,\ldots,s_N}} \\ p_1+\cdots+p_N+r_1+\cdots+r_N=\sigma \\ q_1+\cdots+q_N+s_1+\cdots+s_N=\rho \\ p_k+q_k-r_k-s_k=\mid n_k \mid \\ \sigma+\rho=n+2\alpha}} \left\{ \left(\frac{\sigma!\ \rho! a_{\sigma,\rho}}{2^{(n+2\alpha)}} \right) \Phi \right\} \tag{3.22}$$

and Φ is given by (3.16). The phasor of the ω_q component of the output $y(t)$ is then given by

$$Y_q = \sum_{n=0}^{\infty} \sum_{\substack{\underbrace{n_1,\ldots,n_N} \\ \mid n_1 \mid + \cdots + \mid n_N \mid = n}} U_q. \tag{3.23}$$

We have thus derived an algebraic formula for the output of a bivariate power series having two multifrequency inputs. These formulas reduce to those presented in [12] for a single variable power series with the elimination of the appropriate variables and subsequent grouping of terms.

3.3 The Relationship between Volterra Series and Power Series

In this section an explicit relationship is developed between Volterra series and power series both for the univariate (single input) and bivariate (two–input) case. The strategy is to take both the equation for power series and the equation for Volterra series and derive a form for both equations that show the same structure. The relationship between the two can then be inferred by inspection. For clarity, the simpler univariate case is shown first, and then the more general bivariate case is presented in the same format.

3.3.1 Univariate Case

Generalized Power Series Analysis introduced by Steer and Khan [12] expresses the output $y(t)$ as a function of the input $x(t)$ by the equation

$$y(t) = A \sum_{i=1}^{I} A_i \sum_{n=0}^{\infty} a_{n,i} \left[\sum_{k=1}^{K} b_{k,i} x_k(t - \tau_{k,n,i}) \right]^n \tag{3.24}$$

Where the input is steady–state given by

$$x(t) = \sum_{k=1}^{K} x_k(t) \tag{3.25}$$

and $x_k(t)$ is a single frequency component of radian frequency ω_k. Thus $x_k(t)$ can be expressed as

$$x_k(t) = X_k e^{+j\omega_k t} \tag{3.26}$$

Note that we must choose the $x_k(t)$'s such that (3.25) yields a real $x(t)$. For the discussion here we will restrict A , A_i, $a_{n,i}$, $b_{k,i}$, and $\tau_{k,n,i}$ to be real. Now we wish to put (3.24) in a form which is close to a conventional Volterra series form. Thus we rewrite (3.24) as

$$y(t) = c_0 + \sum_{i=1}^{I} \sum_{n=1}^{\infty} \left[\sum_{k=1}^{K} \hat{c}_{k,i,n} \, x_k(t - \tau_{k,n,i}) \right]^n \tag{3.27}$$

where

$$c_0 \equiv A \sum_{i=1}^{I} a_{0,i} \tag{3.28}$$

and

$$\hat{c}_{k,i,n} \equiv [A \, A_i \, a_{n,i}]^{\frac{1}{n}} \, b_{k,i} \tag{3.29}$$

Changing the summation order then gives

$$y(t) = c_0 + \sum_{n=1}^{\infty} \sum_{i=1}^{I} \left[\sum_{k=1}^{K} c_{k,i,n} \;\; x_k(t) \right]^n \tag{3.30}$$

Where

$$c_{k,i,n} \equiv \hat{c}_{k,i,n} \, e^{-j\omega_k \, \tau_{k,n,i}} \tag{3.31}$$

Or

$$y(t) = \sum_{n=0}^{\infty} y_n(t) \tag{3.32}$$

Where

$$y_n(t) = \sum_{i=1}^{I} \left[\sum_{k=1}^{K} c_{k,i,n} x_k(t) \right]^n \tag{3.33}$$

and

$$y_0(t) = c_0 \tag{3.34}$$

By adding the new summation variable k_1 we can factor the term in brackets in (3.33)

$$\left[\sum_{k=1}^{K} c_{k,i,n} x_k(t) \right]^n = \left[\sum_{k_1=1}^{K} c_{k_1,i,n} x_{k_1}(t) \right] \left[\sum_{k=1}^{K} c_{k,i,n} x_k(t) \right]^{n-1} \tag{3.35}$$

And again with k_2 yields

$$\left[\sum_{k=1}^{K} c_{k,i,n} x_k(t)\right]^n = \left[\sum_{k_1}^{K} c_{k_1,i,n} x_{k_1=1}(t)\right]\left[\sum_{k_2=1}^{K} c_{k_2,i,n} x_{k_2}(t)\right]\left[\sum_{k=1}^{K} c_{k,i,n} x_k(t)\right]^{n-2} \tag{3.36}$$

Continuing this n times gives

$$\left[\sum_{k=1}^{K} c_{k,i,n} x_k(t)\right]^n = \left[\sum_{k_1=1}^{K} c_{k_1,i,n} x_{k_1}(t)\right]\left[\sum_{k_2=1}^{K} c_{k_2,i,n} x_{k_2}(t)\right]\cdots\left[\sum_{k_n=1}^{K} c_{k_n,i,n} x_{k_n}(t)\right] \tag{3.37}$$

Moving all the terms to the right

$$\left[\sum_{k=1}^{K} c_{k,i,n} x_k(t)\right]^n = \sum_{k_1=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_n=1}^{K} c_{k_1,i,n} x_{k_1}(t) c_{k_2,i,n} x_{k_2}(t)\cdots c_{k_n,i,n} x_{k_n}(t) \tag{3.38}$$

Or

$$\left[\sum_{k=1}^{K} c_{k,i,n} x_k(t)\right]^n = \sum_{k_1=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_n=1}^{K}\prod_{\zeta=1}^{n} c_{k_\zeta,i,n}\, x_{k_\zeta}(t) \tag{3.39}$$

Substituting (3.39) into (3.33) and changing the summation order gives

$$y_n(t) = \sum_{k_1=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_n=1}^{K}\sum_{i=1}^{I}\prod_{\zeta=1}^{n} c_{k_\zeta,i,n}\, x_{k_\zeta}(t) \tag{3.40}$$

or

$$y_n(t) = \sum_{k_1=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_n=1}^{K}\left(\left[\prod_{\eta=1}^{n} x_{k_\eta}(t)\right]\sum_{i=1}^{I}\prod_{\zeta=1}^{n} c_{k_\zeta,i,n}\right) \tag{3.41}$$

This is close to the form in which the output of Volterra series analysis is evaluated. We will now consider the output of Volterra series analysis and list it in a form corresponding to (3.41).

In Volterra series analysis the output of a nonlinear system is

$$y(t) = \sum_{n=0}^{\infty} y_n(t) \tag{3.42}$$

where

$$\begin{aligned} y_n(t) \;=\; & \int_{\tau_n=-\infty}^{+\infty}\int_{\tau_{n-1}=-\infty}^{+\infty}\cdots \\ & \int_{\tau_1=-\infty}^{+\infty} h_n(\tau_1,\tau_2\ldots\tau_n)\prod_{i=1}^{n} x(t-\tau_i) \\ & d\tau_1 d\tau_2\ldots d\tau_n \end{aligned} \tag{3.43}$$

This is the integral form of Volterra series analysis and is analogous to a multidimensional convolution. The $h_n()$ terms are known as Volterra kernels. For a linear system only the $h_0()$ and $h_1()$ kernels are non–zero, so that only $y_0(t)$ and $y_1(t)$ are non–zero. The y_0 term

$$y_0(t) \quad = \quad h_0 \tag{3.44}$$

is just a DC output for no input. Equation (3.43) is a more general form of the Volterra integral equation than is usually reported as most authors consider $1 \leq n \leq \infty$ and so neglect the DC offset. Now we want to put (3.43) in frequency domain form and so we will present a number of Fourier transforms which will be used during the development.

$$X(f) \equiv \int_{-\infty}^{+\infty} x(t) e^{-j2\pi ft} dt \tag{3.45}$$

$$X(f) e^{+j2\pi f\tau_i} = \int_{-\infty}^{+\infty} x(t-\tau_i) e^{+j2\pi f\tau_i} d\tau_i \tag{3.46}$$

$$x(t) = \int_{-\infty}^{+\infty} X(f) e^{+j2\pi ft} df \tag{3.47}$$

$$Y_n(f) \equiv \int_{-\infty}^{+\infty} y_n(t) e^{-j2\pi ft} dt \tag{3.48}$$

$$y_n(t) = \int_{-\infty}^{+\infty} Y_n(f) e^{+j2\pi ft} df \tag{3.49}$$

$$\begin{aligned} H_n(f_1, f_2 \ldots f_n) \quad \equiv \quad & \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} h_n(\tau_1, \tau_2 \ldots \tau_n) \\ & e^{-j2\pi(f_1\tau_1 + f_2\tau_2 + \ldots f_n\tau_n)} d\tau_1 d\tau_2 \ldots d\tau_n \end{aligned} \tag{3.50}$$

$$H_0 \equiv h_0 \tag{3.51}$$

and

$$\begin{aligned} h_n(\tau_1, \tau_2 \ldots \tau_n) \quad = \quad & \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} H_n(f_1, f_2 \ldots f_n) \\ & e^{+j2\pi(f_1\tau_1 + f_1\tau_2 + \ldots f_n\tau_n)} df_1 df_2 \ldots df_n \end{aligned} \tag{3.52}$$

Thus by substituting (3.52) into (3.43) and rearranging the terms and order of integration yields

$$\begin{aligned} y_n(t) \quad = \quad & \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} H_n(f_1, f_2 \ldots f_n) \\ & \left(\int_{-\infty}^{+\infty} x(t-\tau_1) e^{+j2\pi f_1\tau_1} d\tau_1 \right) \\ & \left(\int_{-\infty}^{+\infty} x(t-\tau_2) e^{+j2\pi f_2\tau_2} d\tau_2 \right) \\ & \vdots \\ & \left(\int_{-\infty}^{+\infty} x(t-\tau_n) e^{+j2\pi f_n\tau_n} d\tau_n \right) \\ & df_1 df_2 \ldots df_n \end{aligned} \tag{3.53}$$

Using (3.46), this reduces to

$$\begin{aligned} y_n(t) &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\cdots\int_{-\infty}^{+\infty} H_n(f_1, f_2 \ldots f_n) \\ &\left[\prod_{i=1}^{n} X(f_i)e^{+j2\pi f_i t}\right] df_1 df_2 \ldots df_n \end{aligned} \tag{3.54}$$

Substituting (3.54) into (3.48) and rearranging yields

$$\begin{aligned} Y_n(f) &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\cdots\int_{-\infty}^{+\infty} H_n(f_1, f_2 \ldots f_n) \\ &\left[\prod_{i=1}^{n} X(f_i)\right] \int_{-\infty}^{+\infty} e^{+j2\pi(f_1+f_2 \ldots f_n)t} e^{-j2\pi f t} dt \\ &df_1 df_2 \ldots df_n \end{aligned} \tag{3.55}$$

or

$$\begin{aligned} Y_n(f) &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\cdots\int_{-\infty}^{+\infty} H_n(f_1, f_2 \ldots f_n) \\ &\delta(f - f_1 - f_2 \ldots - f_n) \prod_{i=1}^{n} X(f_i) df_i \end{aligned} \tag{3.56}$$

Where $\delta(\cdot)$ is the impulse function. This is the form of the Volterra series used for frequency domain analysis. Note that the first order response, Y_1 reduces to

$$Y_1(f) = \int_{-\infty}^{+\infty} H_1(f_1)\delta(f - f_1)X(f_1)df_1 \tag{3.57}$$

This integral is trivial to evaluate because of the $\delta(f - f_1)$ in the integrand. Thus we get the more common form of the frequency domain equation used for linear analysis.

$$Y_1(f) = H_1(f)X(f) \tag{3.58}$$

Note that (3.58) is valid for any frequency domain description of $X(f)$. Thus in general, $Y_1(f)$ could be considered a continuous spectrum representing any arbitrary signal. Or alternately if $X(f)$ was a phasor representing a single tone input (i.e. an impulse in the frequency domain), then $Y_1(f)$ would also be a single phasor representing the single tone output. Furthermore, if $X(f)$ were a set of phasors, each phasor representing the phase and amplitude of a single tone at a different frequency, then $Y(f)$ would be a set of phasors, each corresponding to a frequency of the resulting sum of output tones.

For $n = 2$, (3.56) becomes

$$Y_2(f) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} H_1(f_1, f_2)\delta(f - f_1 - f_2)X(f_1)X(f_2)df_1 df_2 \tag{3.59}$$

Here the double integral is not trivial to evaluate. So for frequency–domain Volterra analysis to be useful for nonlinear circuits, a restriction must be applied to the input of the circuit. Therefore, we require the input of the system to be steady–state. Thus we can express $x(t)$ in the form of (3.25) where $x_k(t)$ is defined by (3.26), then from (3.47)

$$X(f) = \sum_{k=1}^{K} X_k \, \delta(f - \frac{\omega_k}{2\pi}) \qquad (3.60)$$

Note that the ω_k's are the radian frequencies of the input and f is the frequency of the output. We will use this convention for clarity. Thus

$$\begin{aligned} Y_n(f) &= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} H_n(f_1, f_2 \ldots f_n) \\ & \delta(f - f_1 - f_2 \ldots - f_n) \left[\prod_{i=1}^{n} \sum_{k=1}^{K} X_k \, \delta(f_i - \frac{\omega_k}{2\pi}) \right] \\ & df_1 df_2 \ldots df_n \end{aligned} \qquad (3.61)$$

Expanding the multiplication by introducing the summation variables $k_1, k_2 \ldots k_n$ yields

$$\begin{aligned} Y_n(f) &= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} H_n(f_1, f_2 \ldots f_n) \\ & \delta(f - f_1 - f_2 \ldots - f_n) \\ & \left[\sum_{k_1=1}^{K} X_{k_1} \, \delta(f_1 - \frac{\omega_{k_1}}{2\pi}) \right] \\ & \left[\sum_{k_2=1}^{K} X_{k_2} \, \delta(f_2 - \frac{\omega_{k_2}}{2\pi}) \right] \\ & \vdots \\ & \left[\sum_{k_n=1}^{K} X_{k_n} \, \delta(f_n - \frac{\omega_{k_n}}{2\pi}) \right] \\ & df_1 df_2 \ldots df_n \end{aligned} \qquad (3.62)$$

Changing the order of summation and integration

$$\begin{aligned} Y_n(f) &= \sum_{k_1=1}^{K} \sum_{k_2=1}^{K} \cdots \sum_{k_n=1}^{K} \Bigg[\\ & \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} H_n(f_1, f_2 \ldots f_n) \delta(f - f_1 - f_2 \ldots - f_n) \\ & X_{k_1} \, \delta(f_1 - \frac{\omega_{k_1}}{2\pi}) X_{k_2} \, \delta(f_2 - \frac{\omega_{k_2}}{2\pi}) \ldots X_{k_n} \, \delta(f_n - \frac{\omega_{k_n}}{2\pi}) \\ & df_1 df_2 \ldots df_n \Bigg] \end{aligned} \qquad (3.63)$$

Carrying out the integration is relatively simple since the only nonzero terms of the integrand occur when $f_i = \omega_{k_i}/2\pi$ thus

$$\begin{aligned} Y_n(f) \;=\; & \sum_{k_1=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_n=1}^{K}\Bigg[\\ & H_n(\frac{\omega_{k_1}}{2\pi},\frac{\omega_{k_2}}{2\pi},\ldots\frac{\omega_{k_n}}{2\pi})\delta(f-\frac{\omega_{k_1}}{2\pi}-\frac{\omega_{k_2}}{2\pi}-\ldots-\frac{\omega_{k_n}}{2\pi}) \\ & X_{k_1}\,X_{k_2}\,\ldots X_{k_n}\Bigg] \end{aligned} \tag{3.64}$$

Now using (3.49) this becomes

$$\begin{aligned} y_n(t) \;=\; & \sum_{k_1=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_n=1}^{K} H_n(\frac{\omega_{k_1}}{2\pi},\frac{\omega_{k_2}}{2\pi},\ldots\frac{\omega_{k_n}}{2\pi}) \\ & e^{j(\omega_{k_1}+\omega_{k_2}+\ldots\omega_{k_n})t}X_{k_1}\,X_{k_2}\,\ldots X_{k_n} \end{aligned} \tag{3.65}$$

and using (3.26) we have the desired form of the output of a nonlinear system described by Volterra nonlinear transfer functions.

$$y_n(t) = \sum_{k_1=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_n=1}^{K}\left[\prod_{\eta=1}^{n} x_{k_\eta}(t)\right] H_n(\frac{\omega_{k_1}}{2\pi},\frac{\omega_{k_2}}{2\pi},\ldots\frac{\omega_{k_n}}{2\pi}) \tag{3.66}$$

By comparing, the output of a nonlinear system described by generalized power series, (3.41) with (3.66) we see that univariate Volterra series analysis is identical to univariate generalized power series analysis when the nth order Volterra nonlinear transfer function is given by

$$H_n(\frac{\omega_{k_1}}{2\pi},\frac{\omega_{k_2}}{2\pi},\ldots\frac{\omega_{k_n}}{2\pi}) = \sum_{i=1}^{I}\prod_{\zeta=1}^{n} c_{k_\zeta,i,n} \tag{3.67}$$

That is using (3.29) and (3.31), then

$$H_0 = c_0\sum_{i=1}^{I} a_{0,i} \tag{3.68}$$

and for $n = 1$,

$$H_n(\frac{\omega_{k_1}}{2\pi},\frac{\omega_{k_2}}{2\pi},\ldots\frac{\omega_{k_n}}{2\pi}) = A\sum_{i=1}^{I} A_i\, a_{n,i}\prod_{\zeta=1}^{n} b_{k_\zeta,i}\, e^{-j\omega_{k_\zeta}\,\tau_{k_\zeta,n,i}} \tag{3.69}$$

Thus if we are now given a univariate generalized power series description, we can directly calculate a Volterra nonlinear transfer function.

3.3.2 Bivariate Case

Now that the relationship between univariate Volterra series and power series has been developed, the bivariate case is considered. First a form of bivariate generalized power series is derived, and then an equivalent form for bivariate Volterra series is also derived. As with the univariate case, a direct relationship between the two forms is demonstrated and a formula for the Volterra nonlinear transfer function as a function of the generalized power series description is derived.

Changing the subscripts to be consistent with bivariate Volterra analysis, (3.3) can be expressed as

$$y(t) = A\sum_{i=1}^{I} A_i \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} a_{m,n,i}\left[\sum_{k=1}^{K} b_{k,i}x_k(t-\tau_{k,n,i})\right]^n \left[\sum_{l=1}^{L} d_{l,i}z_l(t-\lambda_{l,m,i})\right]^m \tag{3.70}$$

The sum of the first port is given by $x(t)$ and is descibed by (3.25) and (3.26) as was the case for the univariate system. The input at the second port is given by $z(t)$ and is given by

$$z(t) = \sum_{l=1}^{L} z_k(t) \tag{3.71}$$

and $z_l(t)$ is a single frequency component of radian frequency ω_l. Thus $z_k(t)$ can be expressed as

$$z_l(t) = Z_l e^{+j\omega_l t} \tag{3.72}$$

As with $x_k(t)$, the values of $z_l(t)$ must be in complex conjugate pairs for $\omega_l \neq 0$ such that $z(t)$ is real. The output can be expressed as

$$y(t) = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} y_{m;n}(t) \tag{3.73}$$

Where

$$y_{m;n}(t) = \sum_{i=1}^{I} \hat{a}_{m,n,i}\left[\sum_{k=1}^{K} \hat{b}_{k,i,n}x_k(t)\right]^n \left[\sum_{l=1}^{L} \hat{d}_{l,i,n}z_l(t)\right]^m \tag{3.74}$$

$$\hat{a}_{m,n,i} = AA_i a_{m,n,i} \tag{3.75}$$

$$\hat{b}_{k,i,n} = b_{k,i}e^{-j\omega_k\tau_{k,n,i}} \tag{3.76}$$

$$\hat{d}_{l,i,n} = d_{l,i}e^{-j\omega_l\lambda_{l,n,i}} \tag{3.77}$$

As was done in (3.35), the exponentation in (3.74) can be factored to yield

$$y_{m;n}(t) = \tag{3.78}$$

$$\textstyle\sum_{i=1}^{I} \hat{a}_{m,n,i} \left[\sum_{k_1=1}^{K} \hat{b}_{k_1,i,n}x_{k_1}(t)\right]\left[\sum_{k=1}^{K} \hat{b}_{k,i,n}x_k(t)\right]^{n-1} \left[\sum_{l_1=1}^{L} \hat{d}_{l_1,i,n}z_{l_1}(t)\right]\left[\sum_{l=1}^{L} \hat{d}_{l,i,n}z_l(t)\right]^{m-1} \tag{3.79}$$

Continuing this process results in

$$y_{m;n}(t) = \sum_{i=1}^{I} \hat{a}_{m,n,i} \left[\sum_{k_1=1}^{K} \sum_{k_2=1}^{K} \cdots \sum_{k_n=1}^{K} \prod_{\zeta=1}^{n} \hat{b}_{k_\zeta,i,n} x_{k_\zeta}(t) \right] \left[\sum_{l_1=1}^{L} \sum_{l_2=1}^{L} \cdots \sum_{l_n=1}^{L} \prod_{\nu=1}^{m} \hat{d}_{l_\nu,i,n} z_{l_\nu}(t) \right] \tag{3.80}$$

or

$$\begin{aligned} y_{m;n}(t) &= \sum_{k_1=1}^{K} \sum_{k_2=1}^{K} \cdots \sum_{k_n=1}^{K} \sum_{l_1=1}^{L} \sum_{l_2=1}^{L} \cdots \sum_{l_n=1}^{L} \\ &\quad \left[\prod_{\zeta=1}^{n} x_{k_\zeta}(t) \right] \left[\prod_{\nu=1}^{m} z_{l_\nu}(t) \right] \sum_{i=1}^{I} \hat{a}_{m,n,i} \left[\prod_{\theta=1}^{n} \hat{b}_{k_\theta,i,n} \right] \left[\prod_{\mu=1}^{m} \hat{d}_{l_\mu,i,n} \right] \end{aligned} \tag{3.81}$$

The output of a system described by a bivariate Volterra series is given by

$$y(t) = \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} y_{m;n}(t) \tag{3.82}$$

where

$$\begin{aligned} y_{m;n}(t) &= \int_{\hat{\tau}_m=-\infty}^{+\infty} \int_{\hat{\tau}_{m-1}=-\infty}^{+\infty} \cdots \int_{\hat{\tau}_1=-\infty}^{+\infty} \\ &\quad \int_{\bar{\tau}_n=-\infty}^{+\infty} \int_{\bar{\tau}_{n-1}=-\infty}^{+\infty} \cdots \int_{\bar{\tau}_1=-\infty}^{+\infty} \\ &\quad h_{m;n}(\hat{\tau}_1, \hat{\tau}_2, \ldots \hat{\tau}_m; \bar{\tau}_1, \bar{\tau}_2, \ldots \bar{\tau}_n) \left[\prod_{h=1}^{m} z(t-\hat{\tau}_h) \right] \left[\prod_{i=1}^{n} x(t-\bar{\tau}_i) \right] \\ &\quad d\bar{\tau}_1 d\bar{\tau}_2 \ldots d\bar{\tau}_n d\hat{\tau}_1 d\hat{\tau}_2 \ldots d\hat{\tau}_m \end{aligned} \tag{3.83}$$

in the sense that

$$y_{0;0}(t) = h_{0;0} \tag{3.84}$$

The $y_{0;0}$ term is usually omitted but is kept here for generality. As in the univariate case, we will use the Fourier transform pairs of (3.45) through (3.47) and

$$Y_{m;n}(f) \equiv \int_{-\infty}^{+\infty} y_{m;n}(t) e^{-j2\pi ft} dt \tag{3.85}$$

$$y_{m;n}(t) = \int_{-\infty}^{+\infty} Y_{m;n}(f) e^{+j2\pi ft} df \tag{3.86}$$

$$Z(f) \equiv \int_{-\infty}^{+\infty} z(t) e^{-j2\pi ft} dt \tag{3.87}$$

$$z(t) = \int_{-\infty}^{+\infty} Z(f) e^{+j2\pi ft} df \tag{3.88}$$

$$\begin{aligned} H_{m;n}(\hat{f}_1, \hat{f}_2, \ldots \hat{f}_m; \bar{f}_1, \bar{f}_2, \ldots \bar{f}_n) \equiv & \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} \\ & h_{m;n}(\hat{\tau}_1, \hat{\tau}_2, \ldots \hat{\tau}_m, \bar{\tau}_1, \bar{\tau}_2, \ldots \bar{\tau}_n) \\ & e^{-j2\pi(\hat{f}_1\hat{\tau}_1 + \hat{f}_2\hat{\tau}_2 + \ldots \hat{f}_m\hat{\tau}_m + \bar{f}_1\bar{\tau}_1 + \bar{f}_2\bar{\tau}_2 + \ldots \bar{f}_n\bar{\tau}_n)} \\ & d\hat{\tau}_1 d\hat{\tau}_2 \ldots d\hat{\tau}_m d\bar{\tau}_1 d\bar{\tau}_2 \ldots d\bar{\tau}_n \end{aligned} \tag{3.89}$$

$$H_{0;0} \equiv h_{0;0} \tag{3.90}$$

and

$$\begin{aligned} h_{m;n}(\hat{\tau}_1, \hat{\tau}_2, \ldots \hat{\tau}_m, \bar{\tau}_1, \bar{\tau}_2, \ldots \bar{\tau}_n) = & \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} \\ & H_{m;n}(\hat{f}_1, \hat{f}_2, \ldots \hat{f}_m; \bar{f}_1, \bar{f}_2, \ldots \bar{f}_n) \\ & e^{+j2\pi(\hat{f}_1\hat{\tau}_1 + \hat{f}_2\hat{\tau}_2 + \ldots \hat{f}_m\hat{\tau}_m + \bar{f}_1\bar{\tau}_1 + \bar{f}_2\bar{\tau}_2 + \ldots \bar{f}_n\bar{\tau}_n)} \\ & d\hat{f}_1 d\hat{f}_2 \ldots d\hat{f}_m d\bar{f}_1 d\bar{f}_2 \ldots d\bar{f}_n \end{aligned} \tag{3.91}$$

By substituting (3.91) into (3.83) and rearranging the terms and order of integration

$$\begin{aligned} y_{m;n}(t) = & \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} H_{m;n}(\hat{f}_1, \hat{f}_2, \ldots \hat{f}_m; \bar{f}_1, \bar{f}_2, \ldots \bar{f}_n) \\ & \left(\int_{-\infty}^{+\infty} z(t - \hat{\tau}_1) e^{+j2\pi\hat{f}_1\hat{\tau}_1} d\hat{\tau}_1 \right) \\ & \left(\int_{-\infty}^{+\infty} z(t - \hat{\tau}_2) e^{+j2\pi\hat{f}_2\hat{\tau}_2} d\hat{\tau}_2 \right) \\ & \vdots \\ & \left(\int_{-\infty}^{+\infty} z(t - \hat{\tau}_m) e^{+j2\pi\hat{f}_m\hat{\tau}_m} d\hat{\tau}_m \right) \\ & \left(\int_{-\infty}^{+\infty} x(t - \bar{\tau}_1) e^{+j2\pi\bar{f}_1\bar{\tau}_1} d\bar{\tau}_1 \right) \\ & \left(\int_{-\infty}^{+\infty} x(t - \bar{\tau}_2) e^{+j2\pi\bar{f}_2\bar{\tau}_2} d\bar{\tau}_2 \right) \\ & \vdots \\ & \left(\int_{-\infty}^{+\infty} x(t - \bar{\tau}_n) e^{+j2\pi\bar{f}_n\bar{\tau}_n} d\bar{\tau}_n \right) \\ & d\hat{f}_1 d\hat{f}_2 \ldots d\hat{f}_m d\bar{f}_1 d\bar{f}_2 \ldots d\bar{f}_n \end{aligned} \tag{3.92}$$

Using (3.45) and (3.87) this reduces to

$$\begin{aligned} y_{m;n}(t) = & \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} H_{m;n}(\hat{f}_1, \hat{f}_2, \ldots \hat{f}_m; \bar{f}_1, \bar{f}_2, \ldots \bar{f}_n) \\ & \left[\prod_{h=1}^{m} Z(\hat{f}_h) e^{+j2\pi\hat{f}_h t} \right] \left[\prod_{i=1}^{n} X(\bar{f}_i) e^{+j2\pi\bar{f}_i t} \right] \\ & d\hat{f}_1 d\hat{f}_2 \ldots d\hat{f}_m d\bar{f}_1 d\bar{f}_2 \ldots d\bar{f}_n \end{aligned} \tag{3.93}$$

Substituting this value of $y_{m;n}(t)$ into (3.85) yields

$$\begin{aligned}
Y_{m;n}(t) &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\ldots\int_{-\infty}^{+\infty} H_{m;n}(\hat{f}_1,\hat{f}_2,\ldots\hat{f}_m;\bar{f}_1,\bar{f}_2,\ldots\bar{f}_n) \\
&\int_{-\infty}^{+\infty} e^{+j2\pi(\hat{f}_1+\hat{f}_2+\ldots\hat{f}_m+\bar{f}_1+\bar{f}_2+\ldots\bar{f}_n)t} e^{-j2\pi ft} dt \\
&\left[\prod_{h=1}^{m} Z(\hat{f}_h)\right]\left[\prod_{i=1}^{n} X(\bar{f}_i)\right] \\
&d\hat{f}_1 d\hat{f}_2\ldots d\hat{f}_m d\bar{f}_1 d\bar{f}_2\ldots d\bar{f}_n
\end{aligned} \tag{3.94}$$

or

$$\begin{aligned}
Y_{m;n}(t) &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\ldots\int_{-\infty}^{+\infty} H_{m;n}(\hat{f}_1,\hat{f}_2,\ldots\hat{f}_m;\bar{f}_1,\bar{f}_2,\ldots\bar{f}_n) \\
&\delta(f-\hat{f}_1-\hat{f}_2-\ldots\hat{f}_m-\bar{f}_1-\bar{f}_2-\ldots\bar{f}_n) \\
&\left[\prod_{h=1}^{m} Z(\hat{f}_h)\right]\left[\prod_{i=1}^{n} X(\bar{f}_i)\right] \\
&d\hat{f}_1 d\hat{f}_2\ldots d\hat{f}_m d\bar{f}_1 d\bar{f}_2\ldots d\bar{f}_n
\end{aligned} \tag{3.95}$$

Assuming $x(t)$ is in the form of (3.25) where $x_k(t)$ is defined by (3.26) and $z(t)$ is in the form of

$$z(t) = \sum_{l=1}^{L} z_l(t) \tag{3.96}$$

where $z_l(t)$ is defined by

$$z_l(t) = Z_l e^{+j\omega_l t} \tag{3.97}$$

then from (3.88)

$$Z(f) = \sum_{l=1}^{L} Z_l\,\delta(f-\frac{\omega_l}{2\pi}) \tag{3.98}$$

Substituting the values of $X(f)$ and $Z(f)$ given by (3.60) and (3.98) into (3.95) gives

$$\begin{aligned}
Y_{m;n}(t) &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\ldots\int_{-\infty}^{+\infty} H_{m;n}(\hat{f}_1,\hat{f}_2,\ldots\hat{f}_m;\bar{f}_1,\bar{f}_2,\ldots\bar{f}_n) \\
&\delta(f-\hat{f}_1-\hat{f}_2-\ldots\hat{f}_m-\bar{f}_1-\bar{f}_2-\ldots\bar{f}_n) \\
&\left[\prod_{h=1}^{m}\sum_{l=1}^{L} Z_l\,\delta(\hat{f}_h-\frac{\omega_l}{2\pi})\right]\left[\prod_{i=1}^{n}\sum_{k=1}^{K} X_k\,\delta(\bar{f}_i-\frac{\omega_k}{2\pi})\right] \\
&d\hat{f}_1 d\hat{f}_2\ldots d\hat{f}_m d\bar{f}_1 d\bar{f}_2\ldots d\bar{f}_n
\end{aligned} \tag{3.99}$$

Expanding the multiplication by introducing the summation variables $l_1, l_2\ldots l_m$ and $k_1, k_2\ldots k_n$ yields

$$Y_{m;n}(t) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\ldots\int_{-\infty}^{+\infty} H_{m;n}(\hat{f}_1,\hat{f}_2,\ldots\hat{f}_m;\bar{f}_1,\bar{f}_2,\ldots\bar{f}_n)$$

$$\begin{aligned}
&\delta(f - \hat{f}_1 - \hat{f}_2 - \ldots \hat{f}_m - \bar{f}_1 - \bar{f}_2 - \ldots \bar{f}_n) \\
&\left[\sum_{l_1=1}^{L} Z_{l_1}\, \delta(\hat{f}_1 - \frac{\omega_{l_1}}{2\pi})\right] \left[\sum_{l_2=1}^{L} Z_{l_2}\, \delta(\hat{f}_2 - \frac{\omega_{l_2}}{2\pi})\right] \ldots \left[\sum_{l_m=1}^{L} Z_{l_m}\, \delta(\hat{f}_m - \frac{\omega_{l_m}}{2\pi})\right] \\
&\left[\sum_{k_1=1}^{K} X_{k_1}\, \delta(\bar{f}_1 - \frac{\omega_{k_1}}{2\pi})\right] \left[\sum_{k_2=1}^{K} X_{k_2}\, \delta(\bar{f}_2 - \frac{\omega_{k_2}}{2\pi})\right] \ldots \left[\sum_{k_n=1}^{K} X_{k_n}\, \delta(\bar{f}_n - \frac{\omega_{k_n}}{2\pi})\right] \\
&d\hat{f}_1 d\hat{f}_2 \ldots d\hat{f}_m d\bar{f}_1 d\bar{f}_2 \ldots d\bar{f}_n \qquad (3.100)
\end{aligned}$$

Changing the order of summation and integration

$$\begin{aligned}
Y_{m;n}(t) \;=\; & \sum_{k_1=1}^{K} \sum_{k_2=1}^{K} \cdots \sum_{k_n=1}^{K} \sum_{l_1=1}^{L} \sum_{l_2=1}^{L} \cdots \sum_{l_m=1}^{L} \\
& \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} H_{m;n}(\hat{f}_1, \hat{f}_2, \ldots \hat{f}_m; \bar{f}_1, \bar{f}_2, \ldots \bar{f}_n) \\
& \delta(f - \hat{f}_1 - \hat{f}_2 - \ldots \hat{f}_m - \bar{f}_1 - \bar{f}_2 - \ldots \bar{f}_n) \\
& Z_{l_1}\, \delta(\hat{f}_1 - \frac{\omega_{l_1}}{2\pi}) Z_{l_2}\, \delta(\hat{f}_2 - \frac{\omega_{l_2}}{2\pi}) \ldots Z_{l_m}\, \delta(\hat{f}_m - \frac{\omega_{l_m}}{2\pi}) \\
& X_{k_1}\, \delta(\bar{f}_1 - \frac{\omega_{k_1}}{2\pi}) X_{k_2}\, \delta(\bar{f}_2 - \frac{\omega_{k_2}}{2\pi}) \ldots X_{k_n}\, \delta(\bar{f}_n - \frac{\omega_{k_n}}{2\pi}) \\
& d\hat{f}_1 d\hat{f}_2 \ldots d\hat{f}_m d\bar{f}_1 d\bar{f}_2 \ldots d\bar{f}_n \qquad (3.101)
\end{aligned}$$

Again carrying out the integration is relatively easy since the integrand is nonzero only when $\bar{f}_i = \omega_{k_i}/2\pi$ and $\hat{f}_i = \omega_{l_i}/2\pi$ thus

$$\begin{aligned}
Y_{m;n}(t) \;=\; & \sum_{k_1=1}^{K} \sum_{k_2=1}^{K} \cdots \sum_{k_n=1}^{K} \sum_{l_1=1}^{L} \sum_{l_2=1}^{L} \cdots \sum_{l_m=1}^{L} \\
& H_{m;n}(\frac{\omega_{l_1}}{2\pi}, \frac{\omega_{l_2}}{2\pi}, \ldots \frac{\omega_{l_m}}{2\pi}; \frac{\omega_{k_1}}{2\pi}, \frac{\omega_{k_2}}{2\pi}, \ldots \frac{\omega_{k_n}}{2\pi}) \\
& \delta(f - \frac{\omega_{l_1}}{2\pi} - \frac{\omega_{l_2}}{2\pi} - \ldots \frac{\omega_{l_m}}{2\pi} - \frac{\omega_{k_1}}{2\pi} - \frac{\omega_{k_2}}{2\pi} - \ldots \frac{\omega_{k_n}}{2\pi}) \\
& Z_{l_1}\, Z_{l_2}\, \ldots Z_{l_m} X_{k_1}\, X_{k_2}\, \ldots X_{k_n} \qquad (3.102)
\end{aligned}$$

Now substituting this value of $Y_{m;n}(f)$ into (3.86)

$$\begin{aligned}
y_{m;n}(t) \;=\; & \sum_{k_1=1}^{K} \sum_{k_2=1}^{K} \cdots \sum_{k_n=1}^{K} \sum_{l_1=1}^{L} \sum_{l_2=1}^{L} \cdots \sum_{l_m=1}^{L} \\
& H_{m;n}(\frac{\omega_{l_1}}{2\pi}, \frac{\omega_{l_2}}{2\pi}, \ldots \frac{\omega_{l_m}}{2\pi}; \frac{\omega_{k_1}}{2\pi}, \frac{\omega_{k_2}}{2\pi}, \ldots \frac{\omega_{k_n}}{2\pi}) e^{+j(\omega_{l_1} + \omega_{l_2} + \ldots \omega_{l_m} + \omega_{k_1} + \omega_{k_2} + \ldots \omega_{k_n})t} \\
& Z_{l_1}\, Z_{l_2}\, \ldots Z_{l_m} X_{k_1}\, X_{k_2}\, \ldots X_{k_n} \qquad (3.103)
\end{aligned}$$

and using (3.26) and (3.97)

$$y_{m;n}(t) \;=\; \sum_{k_1=1}^{K} \sum_{k_2=1}^{K} \cdots \sum_{k_n=1}^{K} \sum_{l_1=1}^{L} \sum_{l_2=1}^{L} \cdots \sum_{l_m=1}^{L}$$

$$\left[\prod_{\zeta=1}^{n} x_{k_\zeta}(t)\right]\left[\prod_{\nu=1}^{m} z_{l_\nu}(t)\right]$$
$$H_{m;n}(\frac{\omega_{l_1}}{2\pi},\frac{\omega_{l_2}}{2\pi},\dots\frac{\omega_{l_m}}{2\pi};\frac{\omega_{k_1}}{2\pi},\frac{\omega_{k_2}}{2\pi},\dots\frac{\omega_{k_n}}{2\pi}) \tag{3.104}$$

As we did in the univariate case by comparing (3.41) and (3.66), we now do the same in the bivariate case by comparing (3.104) and (3.81). Thus we observe that

$$H_{m;n}(\frac{\omega_{l_1}}{2\pi},\frac{\omega_{l_2}}{2\pi},\dots\frac{\omega_{l_m}}{2\pi};\frac{\omega_{k_1}}{2\pi},\frac{\omega_{k_2}}{2\pi},\dots\frac{\omega_{k_n}}{2\pi}) = \sum_{i=1}^{I}\hat{a}_{m,n,i}\left[\prod_{\theta=1}^{n}\hat{b}_{k_\theta,i,n}\right]\left[\prod_{\mu=1}^{m}\hat{d}_{l_\mu,i,n}\right] \tag{3.105}$$

or

$$\begin{aligned} H_{m;n}(\frac{\omega_{l_1}}{2\pi},\frac{\omega_{l_2}}{2\pi},\dots\frac{\omega_{l_m}}{2\pi};\frac{\omega_{k_1}}{2\pi},\frac{\omega_{k_2}}{2\pi},\dots\frac{\omega_{k_n}}{2\pi}) &= A\sum_{i=1}^{I}A_i a_{m,n,i} \\ &\left[\prod_{\theta=1}^{n} b_{k_\theta,i,n}e^{-j\omega_{k_\theta}\tau_{k_\theta,i,n}}\right] \\ &\left[\prod_{\mu=1}^{m} d_{l_\mu,i,n}e^{-j\omega_{l_\mu}\lambda_{l_\mu,i,m}}\right] \end{aligned} \tag{3.106}$$

$$H_{0;0} = A\sum_{i=1}^{I}A_i a_{0,0,i} \tag{3.107}$$

Chapter 5 will give an example of a bivariate Volterra description derived from a bivariate power series used to describe the current source of a MESFET as a function of the gate–to–source and drain–to–source voltages.

3.3.3 Discussion

In the previous two subsections the relationship between Volterra series analysis and generalized power series analysis was derived. Although Volterra series is more general than power series, the complexity of determining the nonlinear transfer functions used to characterize a Volterra system can be overwhelming. A power series, if it exists, is much easier to obtain. If the system can be described by a univariate or bivariate power series, we have provided a direct and straightforward means to calculate the Volterra nonlinear transfer function.

A generalized power series can contain frequency dependent delays, τ, and these delays are only calculated for a discrete set of input frequencies, thus the resulting Volterra nonlinear transfer functions are only defined for points defined by those input frequencies. However, as would be expected, these points are exactly the ones needed for any simulation that contains only the prescribed input frequencies.

Further insight to the system representation of these two kinds of analyses can be seen using graphical representations of the system model. In order to simplify the discussion, for now we restrict $A = A_i = I = 1$ in (3.24) and (3.70). The diagrams shown can be extended to arbitrary A, A_i and I, but the comparisons between Volterra and power series are more easily seen for the simpler case.

First let us look at equation (3.24) and at the effects of the parameters b_k and $\tau_{k,n}$. Here the k subscript indicates the input frequency ω_k, and the n subscript indicates the order of the calculation. The parameter b_k is the amplification of the signal at the input frequency ω_k, and $\tau_{k,n}$ is the time delay equivalent to a phase shift at the input frequency ω_k. Therefore for a given n, the effect of b_k and $\tau_{k,n}$ is equivalent to passing the input through a linear system characterized by amplitude $b(\omega)$ and delay $\tau_n(\omega)$. We define $G_{b,\tau,n}$ to be this linear system characterized by b_k and $\tau_{k,n}$.

In generalized power series analysis, the coefficients, a_n, can be complex. At first glance, this appears to be impossible. The left side of (3.24) is a time domain signal, and thus must be a pure real function. If the a_n on the right side are complex, how can the right side be guaranteed to be real? The answer lies in the form of $x_k(t)$ which is complex as given in (3.26), but $x(t)$ is real. Thus the chosen values of $x_k(t)$ must be in complex conjugate pairs so that (3.25) can remain real on the right side. In the calculation of the output of a generalized power series, the requirement of the complex conjugate pairs is used to reduce the complexity of calculating the output. We know that the right side of (3.24) expands to a sum of complex conjugate pairs, each pair corresponding to a single absolute output frequency, but one component of the pair representing the positive frequency, and the other component representing the negative frequency. Since we know *apriori* that these will be complex conjugate pairs, only the positive frequencies need to be calculated. Thus when the a_n coefficient is referred to as being complex, it is really implicitly a function of the output frequency. For the negative output frequencies, the complex conjugate of a_n is used. But in practice the phasors for the negative frequencies are inferred from the values calculated for the positive frequencies. Thus the negative frequency phasors are not directly calculated and in practice the coefficients a_n can be complex numbers.

Previously we defined $G_{b,\tau,n}$ to be the linear system characterized by b_k and $\tau_{k,n}$. Likewise we define G_{a_n} to be the linear system characterized by a_n. We have already stated that a is implicitly a function of output frequency because we allowed them to be complex. We can extend GPSA such that a is explicitly a function of the output frequency. Thus G_{a_n} can be any arbitrary linear system. Moreover, to simplify the calculation, we can restrict τ to be independent of n, or conversely we could let b be dependent on n, thus generalizing $G_{b,\tau,n}$ to be any arbitrary linear system.

From examination of (3.24), we can see that the system representation for univariate GPSA can be given as shown in Figure 3.1. Note that a_0 is just the DC output for the system when no inputs are applied. Figure 3.2 gives the standard system representation for Volterra series analysis. By comparing figures 3.1 and 3.2, we can easily see that GPSA is equivalent to Volterra analysis if we restrict the Volterra nonlinear transfer functions, H_n to be representable as shown in figure 3.3. The same representation can be applied to bivariate Volterra and GPSA analysis. Figure 3.4 shows the equivalent bivariate Volterra nonlinear transfer function that is represented by bivariate GPSA.

As previously stated, the equivalent linear systems (G) that we have used to characterize power series analysis can be any arbitrary linear system. Furthermore, these linear systems are not restricted to causal systems. For example, take the

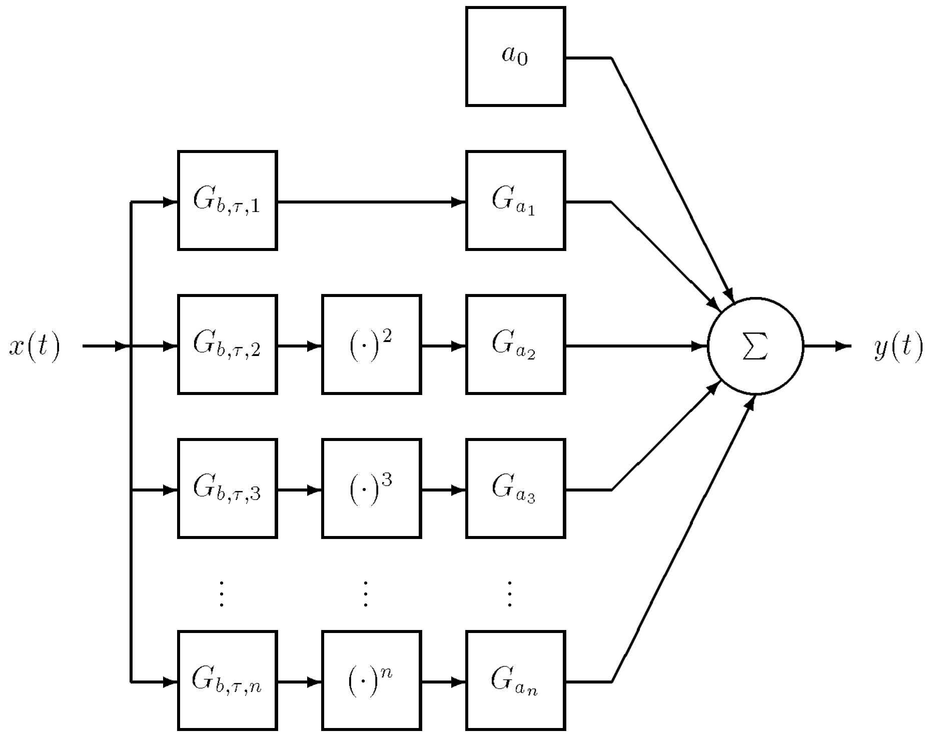

Figure 3.1: Univariate power series system representation.

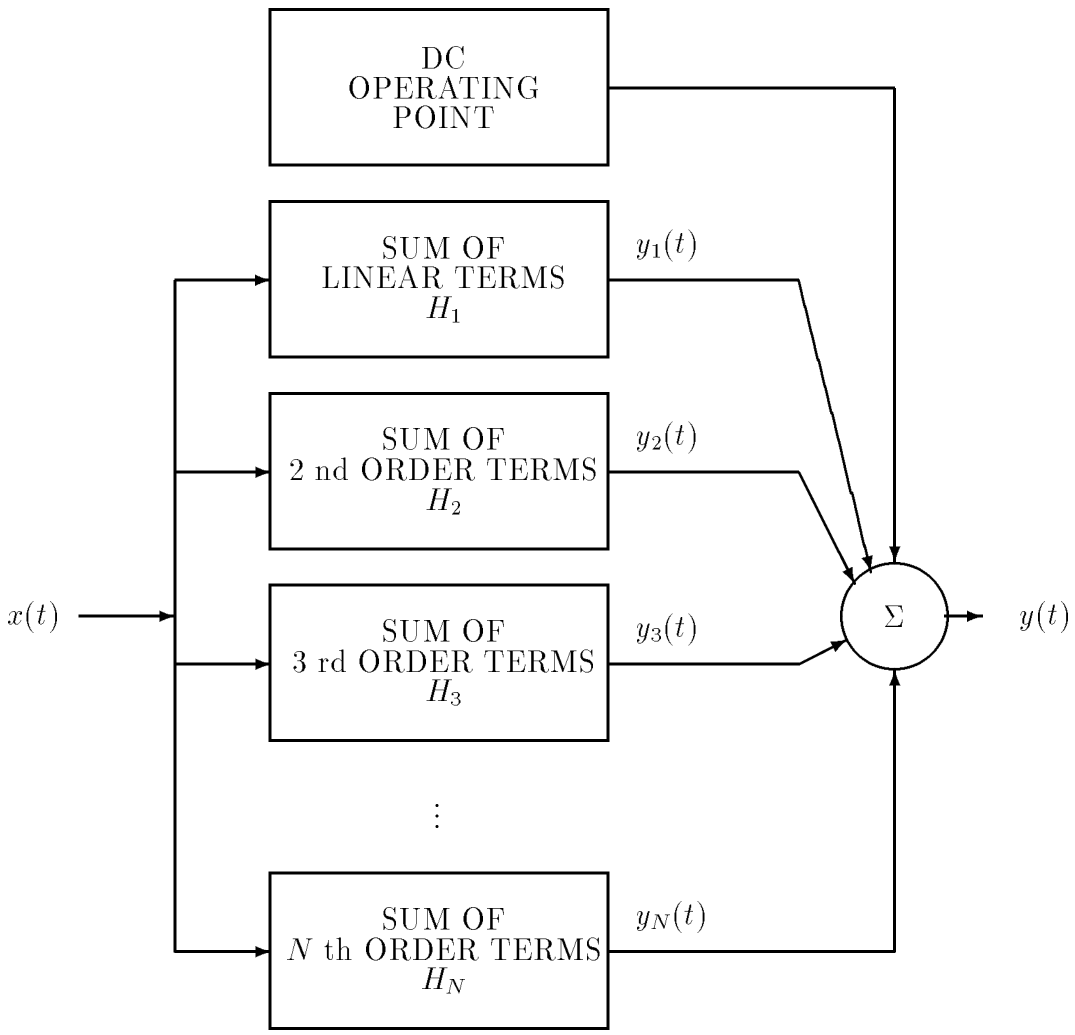
DC OPERATING POINT
SUM OF LINEAR TERMS H_1
SUM OF 2 nd ORDER TERMS H_2
SUM OF 3 rd ORDER TERMS H_3
SUM OF N th ORDER TERMS H_N
$y_1(t)$
$y_2(t)$
$y_3(t)$
$y_N(t)$
$x(t)$
Σ
$y(t)$

Figure 3.2: Illustration of nonlinear analysis using Volterra series.

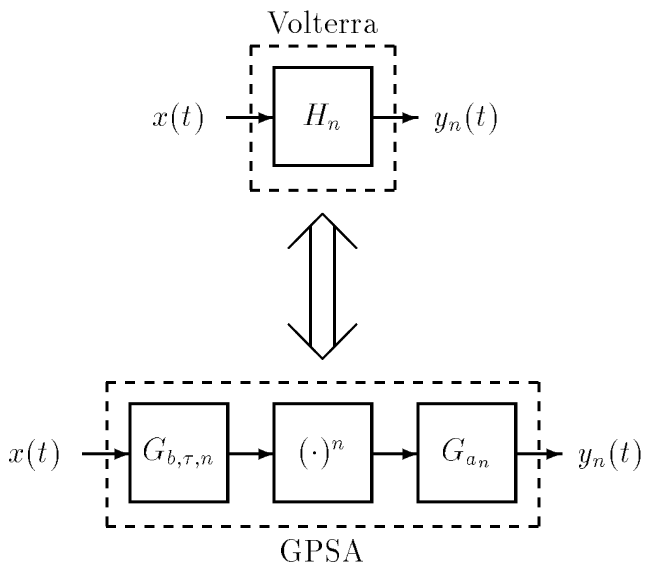

Figure 3.3: Equivalency of univariate Volterra analysis and generalized power series analysis (GPSA).

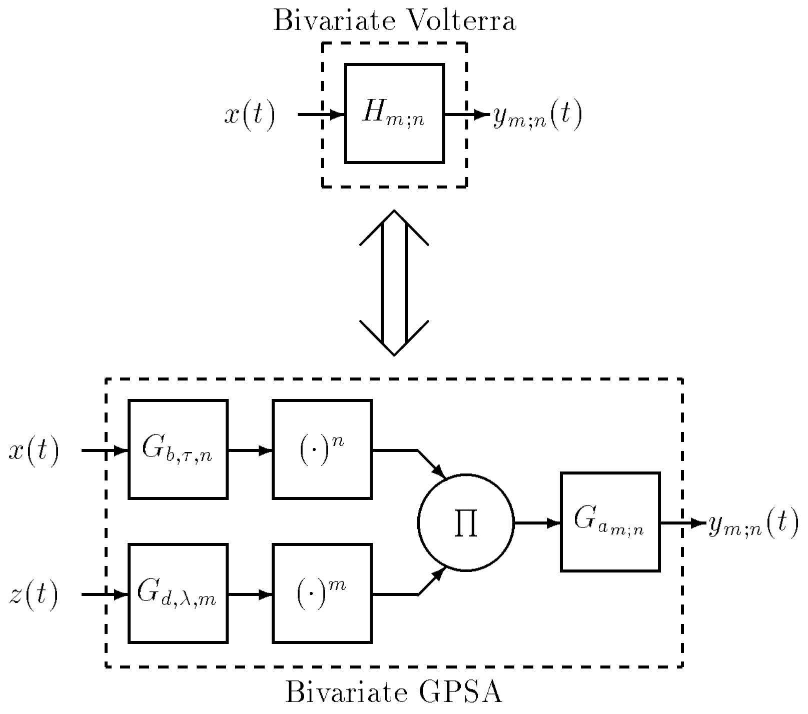

Figure 3.4: Equivalency of bivariate Volterra analysis and generalized power series analysis (GPSA).

simple case of G_a when a is an arbitrary complex constant. As we discussed before, a is implicitly a function of the output frequency. Thus even though we use a constant to implement the calculation of $y(t)$, the full function of G_a is

$$G_a(f) = \begin{cases} a & \text{if } f > 0 \\ \text{Re}\{a\} & \text{if } f = 0 \\ a^* & \text{if } f < 0 \end{cases} \tag{3.108}$$

We can also express this function using the signum function (sgn) defined by

$$\text{sgn}(x) \equiv \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x < 0 \end{cases} \tag{3.109}$$

Thus (3.108) is equivalent to

$$G_a(f) = \text{Re}\{a\} + j\,\text{Im}\{a\}\,\text{sgn}(f) \tag{3.110}$$

To determine causality we can take the inverse fourier transform of the transfer function. The inverse Fourier transform of (3.110) is given by

$$g(t) = \text{Re}\{a\}\,\delta(t) - \frac{\text{Im}\{a\}}{\pi\, t} \tag{3.111}$$

We can now see that the impulse response of the system described by (3.108) has nonzero values for $t < 0$ and thus the system is not causal. Of course it is impossible to build a system that is not causal, but the non-causal system here is only a section of a system model used to represent the steady–state response of nonlinear systems.

3.4 Volterra Harmonic Balance Behavioral Simulation

Given that one has the Volterra descriptions of several subcircuits, one would like to use those descriptions to simulate the entire system whose subcircuits have been characterized. If the Volterra description is independent of the input and output loading effects of the rest of the circuit, then the subcircuit can be modeled as a black box voltage or current source. For instance, assuming that the input nodes had high impedance, and the output nodes had low impedance, the subcircuit could be modeled as a voltage–controlled voltage source as shown in figure 3.5. The output voltage is described by Volterra nonlinear transfer functions. If the input and output loading effects were linear, then these loading effects could be taken into account by adding an input impedance across the two input terminals, and an output impedance in series with one of the output terminals. Thus a very general behavioral model can be added to a circuit simulator, allowing it to act both as a circuit and as a system simulator. The simulator, however, must be able to use the Volterra description of the input–output relationship of the voltage-controlled voltage source. The scheme presented here is to include this in a harmonic balance

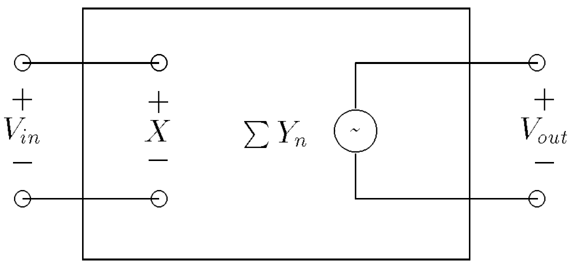

Figure 3.5: Equivalent Circuit model for Voltage controlled voltage source described by Volterra nonlinear transfer functions.

simulator, such as FREDA [20] since the Volterra description used is most useful in the frequency domain.

The idea of using a black box voltage–controlled voltage source can easily be extended to a current–controlled current source, a current–controlled voltage source, or a voltage–controlled current source. The scheme has several advantages. The simulator can be used for small circuits or large systems. Any mix of circuit components and behavioral black box models can easily be simulated. Furthermore, the calculation of the Volterra series description can use the same frequency indexing scheme that is used for GPSA after some modification. The scheme also can be extended to bivariate models, e.g. a voltage source controlled by two input ports (four terminals).

If the time domain form of Volterra series as given in (3.43) was used instead of the frequency domain form as given in (3.66) there would be several disadvantages. The first disadvantage would be the amount of data needed for a table based model. The time domain calculation requires n–fold convolution integrals using the Volterra kernels. Thus the entire continuous functions $h_n(t_1, t_2, \ldots t_n)$ must be stored. Not only must they be stored, but it also must be somehow measured or derived. In the scheme presented here, we have restricted the input to our simulation to be represented by a set of Fourier series, thus if we know which input frequencies will be used, we know the exact points on the nonlinear transfer functions that will be needed. Note we do not need to know the values of the phases or amplitudes of the input phasors, just a list of all of the possible input frequencies. So instead of storing entire continuous n dimensional kernel functions, we only have to store a defined set of points of the n dimensional nonlinear transfer functions. Furthermore, calculating the n–fold convolution integrals in the time domain would be very expensive in terms of computation time. The integral calculation can be simplified if the form of the kernel is known in advance, but such a restriction would limit either the accuracy of the model or limit the class of circuits that could be modeled.

3.4.1 Univariate Volterra and Power Series Simulation

In order to use a Volterra description of a subcircuit, one needs a simulator that can calculate the output of the system described by the Volterra nonlinear transfer functions H_0, $H_1(f_1)$, $H_2(f_1, f_2)$, $H_3(f_1, f_2, f_3)$, $\ldots$ $H_n(f_1, f_2, f_3 \ldots f_n)$. Here we present a modification to an already existing indexing scheme for GPSA to allow a GPSA simulator to use univariate Volterra nonlinear transfer functions as the characterization of a black box model.

Consider the case of a single input consisting of a finite sum of sinusoids

$$x(t) = \sum_{k=1}^{N} x_k(t) = \sum_{k=1}^{N} \mid X_k \mid \cos(\omega_k t + \phi_k) \tag{3.112}$$

where the output can be described as a power series

$$y(t) = \sum_{\ell=0}^{\infty} \left[a_\ell \left\{ \sum_{k=1}^{N} x_k(t) \right\}^{\ell} \right] \tag{3.113}$$

The input can be expressed as a sum of complex exponentials

$$\begin{aligned} x_k(t) &= \mid X_k \mid \cos(\omega_k t + \phi_k) \\ &= \frac{1}{2} X_k \mathrm{e}^{j\omega_k t} + \frac{1}{2} X_k^* \mathrm{e}^{-j\omega_k t} \end{aligned}$$

where X_k is the phasor of x_k. As discussed in subsection 2.3.2, the phasor of the component of the output at frequency ω_q is given by

$$Y_q = \sum_{n=0}^{\infty} \sum_{\substack{\underbrace{n_1, \ldots, n_N} \\ \mid n_1 \mid + \cdots + \mid n_N \mid = n}} U_q \tag{3.114}$$

where $\omega_q = \sum_{k=1}^{N} n_k \omega_k$, a set of n_k's defines an intermodulation product description (called an IPD), and n is the order of intermodulation. The second summation is over all possible combinations of $n_1, \ldots, n_N$ such that $|n_1| + \cdots + |n_N| = n$.

$$U_q = \mathrm{Re} \left\{ \epsilon_n T \right\}_{\omega_q} \tag{3.115}$$

where

$$T = \sum_{\alpha=0}^{\infty} \sum_{\substack{\underbrace{s_1, \ldots, s_N} \\ s_1 + \cdots + s_N = \alpha}} \left\{ \left(\frac{(n+2\alpha)!}{2^{(n+2\alpha)}} \right) a_{n+2\alpha} \Phi \right\} \tag{3.116}$$

and

$$\Phi = \prod_{k=1}^{N} \frac{(X_k^{\dagger})^{|n_k|} \mid X_k \mid^{2s_k}}{s_k!(\mid n_k \mid + s_k)!} \tag{3.117}$$

Where $s_k \geq 0$. In these expressions X_k is the phasor of x_k ,

$$X_k^\dagger = \begin{cases} X_k & n_k \geq 0 \\ X_k^* & n_k < 0 \end{cases}, \tag{3.118}$$

$$\epsilon_n = \begin{cases} 1 & n = 0 \\ 2 & n \neq 0 \end{cases}, \tag{3.119}$$

and $\mathrm{Re}\{\}_{\omega_q}$ is defined such that for $\omega_q \neq 0$ it is ignored and for $\omega_q = 0$ the real part of the expression in braces is taken.

Assuming that the series (3.113) is convergent, then the summation may be truncated and still represent an output arbitrarily close to the true value. Thus we can specify

$$a_\ell = \begin{cases} a_\ell & \ell \leq n_{max} \\ 0 & \ell > n_{max} \end{cases}, \tag{3.120}$$

Thus the infinite summation limits in (3.113), (3.114) and (3.116) can be replaced by n_{max}, n_{max}, and $n_{max}/2$. For the general case in which N components are considered as inputs, a set of integers, denoted n_k, are used to specify the frequency,

$$\omega = \sum_{k=1}^{N} n_k \omega_k$$

As before the set of n_k's is an IPD and the order of intermodulation is given by

$$n = \sum_{k=1}^{N} \mid n_k \mid .$$

IPD's up to a maximum order n_{max} are predetermined and stored in a database. In the evaluation of the algebraic formula, all intermodulation products of the same order are calculated and added to the total response for that frequency component until the desired fractional accuracy is obtained.

Although a device described by a generalized power series, (3.113), has constant Volterra nonlinear transfer functions

$$H_n(f_1, f_2 \ldots f_n) = a_n \tag{3.121}$$

the techniques used in evaluating generalized power series cannot be directly applied to evaluating the output of a system described by Volterra nonlinear transfer functions. In order to apply GPSA techniques to Volterra analysis, we need to rearrange (3.114) - (3.117). The value of n in (3.114) is the order of the IPD, but not necessarily the order of the corresponding Volterra nonlinear transfer function. As seen in (3.64) the μth order nonlinear transfer function H_μ is multiplied by μ phasor components of the input. By observing the form of (3.116) – (3.117) we can define μ to be

$$\mu = n + 2\alpha \tag{3.122}$$

Thus we will rewrite (3.116) as

$$T = \underbrace{\sum_{\mu=n}^{\infty}}_{\substack{\mu \text{ odd for } n \text{ odd} \\ \mu \text{ even for } n \text{ even}}} \underbrace{\sum_{s_1,\ldots,s_N}}_{s_1+\cdots+s_N=(\mu-n)/2} \frac{\mu!}{2^\mu} a_\mu \, \Phi \tag{3.123}$$

If we rearrange the summation order of (3.114) – (3.117) and use (3.123) instead of (3.116)

$$Y_q = \sum_{\mu=0}^{\infty} \sum_{\substack{n=0,1 \\ n \text{ odd for } \mu \text{ odd} \\ n \text{ even for } \mu \text{ even}}}^{\mu} \underbrace{\sum_{n_1,\ldots,n_N}}_{|\, n_1 \,| + \cdots + |\, n_N \,| = n} \overline{U_q} \tag{3.124}$$

$$\overline{U_q} = \mathrm{Re}\left\{ \epsilon_n \underbrace{\sum_{s_1,\ldots,s_N}}_{s_1+\cdots+s_N=(\mu-n)/2} a_\mu \frac{\mu!}{2^\mu} \prod_{k=1}^{N} \frac{(X_k^\dagger)^{|n_k|} \,|\, X_k \,|^{2s_k}}{s_k!(|\, n_k \,| + s_k)!} \right\}_{\omega_q} \tag{3.125}$$

We now have an equation that is in the form for both GPSA and Volterra analysis. Since a_μ is equivalent to the Volterra nonlinear transfer function, we can use (3.125) to calculate the response to a system characterized by a Volterra series if we interpret the argument of the nonlinear transfer function correctly. Thus

$$\overline{U_q} = \mathrm{Re}\left\{ \epsilon_n \underbrace{\sum_{s_1,\ldots,s_N}}_{s_1+\cdots+s_N=\frac{\mu-n}{2}} H_\mu(f_1, f_2 \ldots f_\mu) \frac{\mu!}{2^\mu} \prod_{k=1}^{N} \frac{(X_k^\dagger)^{|n_k|} \,|\, X_k \,|^{2s_k}}{s_k!(|\, n_k \,| + s_k)!} \right\}_{\omega_q} \tag{3.126}$$

Here the arguments of H_μ are determined by the two sets of numbers n_k and s_k. The value of s_i is the number of positive and negative pairs of $f = \pm\omega_i/2\pi$. Since these are pairs whose frequencies add to zero, they have no effect on the output frequency. The value of a given $|n_i|$ gives the number of additional times that $f = \pm\omega_i/2\pi$ appears in the argument. A positive value of n_i indicates that exactly $|n_i|$ values are $+\omega_i/2\pi$, and a negative value of n_i indicates that exactly $|n_i|$ values are $-\omega_i/2\pi$. Since, without loss of generality, we can restrict the nonlinear transfer functions to be symmetric [106], (see the discussion in section B.3) the order of the f_i's has no effect.

For example, suppose $N = 3$, $n_1 = 0$, $n_2 = -1$, $n_3 = 1$, $s_1 = 1$, $s_2 = 2$, and $s_3 = 0$. Thus

$$\mu = \sum_{i}^{N} |n_i| + 2s_i = 8 \tag{3.127}$$

Thus for our example the arguments of H would be

$$H_8(\underbrace{+\frac{\omega_1}{2\pi},-\frac{\omega_1}{2\pi},}_{s_1=1}\underbrace{+\frac{\omega_2}{2\pi},-\frac{\omega_2}{2\pi},+\frac{\omega_2}{2\pi},-\frac{\omega_2}{2\pi},}_{s_2=2}\ \underbrace{-\frac{\omega_2}{2\pi},}_{n_2=-1}\ \underbrace{+\frac{\omega_3}{2\pi}}_{n_3=+1}\ ) \tag{3.128}$$

Thus with a few modifications, a GPSA simulator can be turned into a general purpose Volterra simulator. Furthermore, the method of nonlinear currents could be used to solve the circuit equations. This is a direct calculation and does not require iteration, thus also not requiring neither the evaluation of a Jacobian matrix nor its inversion.

For the previous example with the spectrum shown in figure 2.1, tables 3.1 and 3.2 show the correspondence between the IPD and the arguments of a Volterra nonlinear transfer function. The arguments of the Volterra nonlinear transfer function is a set of frequencies, and the value of the Volterra nonlinear transfer function at that point corresponds to the intermodulation of those frequencies of the order of that Volterra nonlinear transfer function. Thus we term this list of frequencies a frequency intermodulation product description, or FIPD. The order of the FIPD is defined as the number of frequencies in the FIPD and is the same as the order of the corresponding Volterra nonlinear transfer function.

Table 3.1 gives the simpler case for $\alpha = 0$. For $\alpha = 0$, the order of the IPD is the same as the order of the FIPD. The absolute value of n_1 corresponds to the number of times frequency f_1 appears in the FIPD. The sign of n_1 is the same as the sign of the corresponding f_1 entries. Thus if $n_1 = -2$, then $-f_1$ would appear in the FIPD twice. The same relationship holds for n_2 and f_2, n_3 and f_3, and so on.

The case of $\alpha \neq 0$ is more complicated and is shown in table 3.2. For nonzero α, the order of the calculation (e.g. the order of the FIPD), is not equal to the order of the IPD. Instead the order of the corresponding FIPD is the order of the IPD plus 2α. For our example of three input frequencies, the value of α is the sum of $s_1 + s_2 + s_3$. The value of s_1 corresponds to the number of $+/-$ pairs of f_1 present in the FIPD. Thus if $s_1 = 2$, then we would find two $+/-$ pairs of f_1 (i.e. f_1, $-f_1$, f_1, $-f_1$) in the FIPD. These occurrences of f_1 are in addition to any occurrences associated with the value of n_1. The same relationship holds for s_2 and f_2, s_3 and f_3, and so on.

Table 3.1: Relationship between IPD and arguments to Volterra nonlinear transfer function for $\alpha = 0$

Output Frequency	n	IPD			FIPD (arguments of H_n)
		n_1	n_2	n_3	
f_1, IF	1	1	0	0	f_1
	2	0	1	-1	$f_2, -f_3$
	4	2	-1	1	$f_1, f_1, -f_2, f_3$
	5	-1	2	-2	$-f_1, f_2, f_2, -f_3, -f_3$
	7	3	-2	2	$f_1, f_1, f_1, -f_2, -f_2, f_3, f_3$
	8	-2	3	-3	$-f_1, -f_1, f_2, f_2, f_2, -f_3, -f_3, -f_3$
f_2, LO	1	0	1	0	f_2
	2	1	0	1	f_1, f_3
	4	-1	2	-1	$-f_1, f_2, f_2, -f_3$
	5	2	-1	2	$f_1, f_1, -f_2, f_3, f_3$
	7	-2	3	-2	$-f_1, -f_1, f_2, f_2, f_2, -f_3, -f_3$
	8	3	-2	3	$f_1, f_1, f_1, -f_2, -f_2, f_3, f_3, f_3$
f_3, RF	1	0	0	1	f_3
	2	-1	1	0	$-f_1, f_1$
	4	1	-1	2	$f_1, -f_2, f_3, f_3$
	5	-2	2	-1	$-f_1, -f_1, f_2, f_2, -f_3$
	7	2	-2	3	$f_1, f_1, -f_2, -f_2, f_3, f_3, f_3$
	8	-3	3	-2	$-f_1, -f_1, -f_1, f_2, f_2, f_2, -f_3, -f_3$

Table 3.2: Example of relationship between IPD and arguments to Volterra nonlinear transfer function for nonzero α

n_1	n_2	n_3	α	s_1	s_2	s_3	FIPD
0	1	-1	0	0	0	0	$f_2, -f_3$
0	1	-1	1	1	0	0	$f_2, -f_3, \underbrace{f_1, -f_1}_{s_1}$
0	1	-1	1	0	1	0	$f_2, -f_3, \underbrace{f_2, -f_2}_{s_2}$
0	1	-1	1	0	0	1	$f_2, -f_3, \underbrace{f_3, -f_3}_{s_3}$
0	1	-1	2	1	1	0	$f_2, -f_3, \underbrace{f_1, -f_1}_{s_1}, \underbrace{f_2, -f_2}_{s_2}$
0	1	-1	2	0	1	1	$f_2, -f_3, \underbrace{f_2, -f_2}_{s_2}, \underbrace{f_3, -f_3}_{s_3}$
0	1	-1	2	2	0	0	$f_2, -f_3, \underbrace{f_1, -f_1, f_1, -f_1}_{s_1}$
0	1	-1	2	0	2	0	$f_2, -f_3, \underbrace{f_2, -f_2, f_2, -f_2}_{s_2}$
0	1	-1	2	0	0	2	$f_2, -f_3, \underbrace{f_2, -f_3, f_3, -f_3}_{s_3}$

3.4.2 Bivariate Volterra and Power Series Simulation

In the previous section we discussed how a GPSA simulator can be modified to accept univariate Volterra nonlinear transfer functions as the subcircuit descriptions. Here we present the same for the bivariate case.

For the case of a dual input structure with each input, $x(t)$ and $z(t)$, consisting of a finite sum of sinusoids

$$x(t) = \sum_{k=1}^{N} x_k(t) = \sum_{k=1}^{N} |X_k| \cos(\omega_k t + \phi_k) \tag{3.129}$$

$$z(t) = \sum_{k=1}^{N} z_k(t) = \sum_{k=1}^{N} |Z_k| \cos(\omega_k t + \theta_k) \tag{3.130}$$

and where the output can be described as a power series

$$y(t) = \sum_{\sigma=0}^{\infty} \sum_{\rho=0}^{\infty} a_{\sigma,\rho} \left[x(t)\right]^{\sigma} \left[z(t)\right]^{\rho} \tag{3.131}$$

$$\omega = \sum_{k=1}^{N} n_k \omega_k \tag{3.132}$$

The phasor of the ω_q component of the output $y(t)$ is then given by

$$Y_q = \sum_{n=0}^{\infty} \sum_{\underbrace{n_1, \ldots, n_N}_{|n_1| + \cdots + |n_N| = n}} U_q \tag{3.133}$$

$$U_q = \text{Re}\left\{\epsilon_n T\right\}_{\omega_q} \tag{3.134}$$

The intermodulation order is n, where $n = \sum_{k=1}^{N} |n_k|$. where p_k, q_k, r_k, $s_k \geq 0$. ϵ_n is the Neumann factor, ($\epsilon_n = 1$, $n = 0$; $\epsilon_n = 2$, $n \neq 0$), and Re$\{\ \}_{\omega_q}$ is defined such that it is ignored for $\omega_q \neq 0$ but for $\omega_q = 0$ the real part of the expression in brackets is taken.

$$T = \sum_{\alpha=0}^{\infty} \sum_{\substack{\underbrace{\substack{p_1, \ldots, p_N, r_1, \ldots, r_N \\ q_1, \ldots, q_N, s_1, \ldots, s_N}}_{} \\ p_1 + \cdots + p_N + r_1 + \cdots + r_N = \sigma \\ q_1 + \cdots + q_N + s_1 + \cdots + s_N = \rho \\ p_k + q_k - r_k - s_k = |n_k| \\ \sigma + \rho = n + 2\alpha}} \left\{ \left(\frac{\sigma!\ \rho! a_{\sigma,\rho}}{2^{(n+2\alpha)}} \right) \Phi \right\} \tag{3.135}$$

$$\Phi = \prod_{k=1}^{N} \frac{(X_k^{\dagger})^{p_k} (X_k^{\ddagger})^{r_k} (Z_k^{\dagger})^{q_k} (Z_k^{\ddagger})^{s_k}}{p_k!\ r_k!\ q_k!\ s_k!}. \tag{3.136}$$

$$X_k^{\dagger} = \begin{cases} X_k & \text{for } n_k \geq 0 \\ X_k^* & \text{for } n_k < 0 \end{cases}$$

and

$$X_k^{\ddagger} = \begin{cases} X_k^* & \text{for } n_k \geq 0 \\ X_k & \text{for } n_k < 0 \end{cases}$$

($Z_k^{\dagger}$, and $Z_k^{\ddagger}$ are similarly defined.)

As with the univariate case, our aim is to rewrite the equations with the leftmost summation over the order of the response. For the bivariate case, the order is $\mu = \rho + \sigma$, thus (3.135) can be rewritten as

$$T = \underbrace{\sum_{\mu=n}^{\infty}}_{\substack{\mu \text{ odd for } n \text{ odd} \\ \mu \text{ even for } n \text{ even}}} \sum_{\substack{\underbrace{\substack{p_1,\ldots,p_N,r_1,\ldots,r_N \\ q_1,\ldots,q_N,s_1,\ldots,s_N}} \\ p_1+\cdots+p_N+r_1+\cdots+r_N=\sigma \\ q_1+\cdots+q_N+s_1+\cdots+s_N=\rho \\ p_k+q_k-r_k-s_k=|n_k| \\ \sigma+\rho=\mu}} \left\{ \left(\frac{\sigma!\ \rho! a_{\sigma,\rho}}{2^{\mu}} \right) \Phi \right\} \tag{3.137}$$

As in the univariate case, we now rearrange the order of summation

$$Y_q = \sum_{\mu=0}^{\infty} \underbrace{\sum_{n=0,1}^{\mu}}_{\substack{n \text{ odd for } \mu \text{ odd} \\ n \text{ even for } \mu \text{ even}}} \underbrace{\sum_{n_1,\ldots,n_N}}_{|n_1|+\cdots+|n_N|=n} \overline{U_q} \tag{3.138}$$

Where

$$\overline{U_q} = \text{Re} \left\{ \epsilon_n \sum_{\substack{\underbrace{\substack{p_1,\ldots,p_N,r_1,\ldots,r_N \\ q_1,\ldots,q_N,s_1,\ldots,s_N}} \\ p_1+\cdots+p_N+r_1+\cdots+r_N=\sigma \\ q_1+\cdots+q_N+s_1+\cdots+s_N=\rho \\ p_k+q_k-r_k-s_k=|n_k| \\ \sigma+\rho=\mu}} a_{\sigma,\rho} \frac{\sigma!\ \rho!}{2^{\mu}} \Phi \right\}_{\omega_q} \tag{3.139}$$

As in the univariate case, $a_{\sigma,\rho}$ can be replaced by the bivariate Volterra nonlinear transfer function $H_{\sigma,\rho}(\hat{f}_1, \hat{f}_2, \ldots \hat{f}_m; \bar{f}_1, \bar{f}_2, \ldots \bar{f}_n)$ if the values of the $\hat{f}_i$'s and $\bar{f}_i$'s are correctly interpreted as functions of the values of p_k, q_k, r_k, and s_k for all k. Thus

$$\overline{U_q} = \mathrm{Re}\left\{ \epsilon_n \sum_{\substack{\underbrace{\substack{p_1,\ldots,p_N,r_1,\ldots,r_N\\ q_1,\ldots,q_N,s_1,\ldots,s_N}}\\ p_1+\cdots+p_N+r_1+\cdots+r_N=\sigma \\ q_1+\cdots+q_N+s_1+\cdots+s_N=\rho \\ p_k+q_k-r_k-s_k=|n_k| \\ \sigma+\rho=\mu}} H_{\sigma,\rho}(\hat{f}_1,\hat{f}_2\ldots\hat{f}_\sigma;\bar{f}_1,\bar{f}_2\ldots\bar{f}_\rho)\frac{\sigma!\,\rho!}{2^\mu}\Phi \right\}_{\omega_q} \tag{3.140}$$

The roll of the index vectors in the bivariate case is different than that of the univariate case. The values of p_k and q_k represent the number of times $+\frac{\omega_k}{2\pi}$ occurs in the $\hat{f}$ and $\bar{f}$ arguments respectively. The values of r_k and s_k represent the number of times $-\frac{\omega_k}{2\pi}$ occurs in the $\hat{f}$ and $\bar{f}$ arguments respectively. For the example of

$$\begin{aligned} p_2 &= 3 \\ p_4 &= 1 \\ r_4 &= 2 \\ q_3 &= 1 \\ \text{all others} &= 0 \end{aligned}$$

would correspond to

$$H_{6,1}(\underbrace{+\frac{\omega_2}{2\pi},+\frac{\omega_2}{2\pi},+\frac{\omega_2}{2\pi}}_{s_2=3},\underbrace{+\frac{\omega_4}{2\pi}}_{s_4=1},\underbrace{-\frac{\omega_4}{2\pi},-\frac{\omega_4}{2\pi}}_{r_4=2};\underbrace{+\frac{\omega_3}{2\pi}}_{q_3=1}) \tag{3.141}$$

Thus we can modify the indexing scheme for bivariate GPSA to make it more general and to handle both bivariate GPSA models and bivariate Volterra models in the same simulation. Thus a simulator could handle both bivariate power series models and the more general bivariate Volterra series models.

3.5 Conclusion

In order to explore ways of general frequency domain modeling, this chapter has extensively reviewed two types of frequency domain analysis, generalized power series analysis and Volterra analysis. The relationship between these has been shown for both the single input, or univariate, system and the two input, or bivariate, system. By understanding this relationship, the indexing scheme for GPSA was modified to provide a basis for simulation of Volterra models, both univariate and bivariate. Since the Volterra model is more general, a Volterra simulation could include both

Volterra models and GPSA models. This kind of simulation would extend the modeling capability available, and provide a basis where complex subsystems could be simulated using Volterra descriptions and device level circuits could, in the same simulation, be modeled using power series techniques. Thus frequency domain circuit simulation could be generalized to include system level verification. However, this system level simulation is not possible without the ability to extract Volterra models. This extraction is examined in the next chapter.

Chapter 4

Volterra Behavioral Model Extraction

4.1 Introduction

The advantage of circuit and system simulation is the fact that a circuit can be designed and tested before it is built, thus giving the designer a fast and inexpensive way to test designs. But of course, for any simulation, the results are only as good as the models. For simulating large circuits or systems, the compute resource required can be prohibitive if transistor level SPICE–like simulations are used. In order to provide a practical means of simulating large systems, the Volterra harmonic balance technique proposed in section 3.4 can be used. But the work to date on extracting table–based Volterra behavioral models has been restricted to simple models and extraction techniques based on lab measurements. Thus a circuit can only be modeled after it is built. The technique proposed here is shown to be applicable to extraction by running circuit level simulations, thus allowing the designer to simulate the entire system before the subcircuits are built. Furthermore, if the subcircuit description can be simulated for various process variations, multiple behavioral models can be extracted to represent the performance variation of the manufactured subcircuits, and thus the manufactured system. This ability to predict not only the nominal performance, but also the performance variation due to manufacturing tolerances becomes increasingly important as systems become larger and more complex.

This chapter will present a procedure to extract the Volterra nonlinear transfer function $H_N(f_1, f_2 \dots f_n)$. This procedure requires only a circuit level description of the subcircuit to be modeled and a circuit simulator that can simulate the steady–state behavior of the circuit. This can be either a time–based simulator that uses shooting methods [72–75, 77, 112] or a frequency–based simulator that uses harmonic balance techniques [8, 78]. One also needs to know the possible discrete input frequencies and the range of input amplitudes. An additional parameter that must be specified by the user is the maximum order of the system which will be the highest order Volterra nonlinear transfer function to be extracted. The extraction process uses this order as the order of multiple polynomial curve fits, and assume $H_n = 0$ for all n greater than the specified maximum order. The error of this polynomial curve fit can be used to determine if the specified maximum order is sufficiently large.

4.2 Extraction procedure

A high–level flow chart of the overall extraction procedure is given in figure 4.1. The initialize portion of the program includes the determination of both how many simulations are to be run and exactly what the input stimulus of each simulation will be. In order to understand how this selection is determined, we must first review how the Volterra nonlinear transfer function is calculated in the last step.

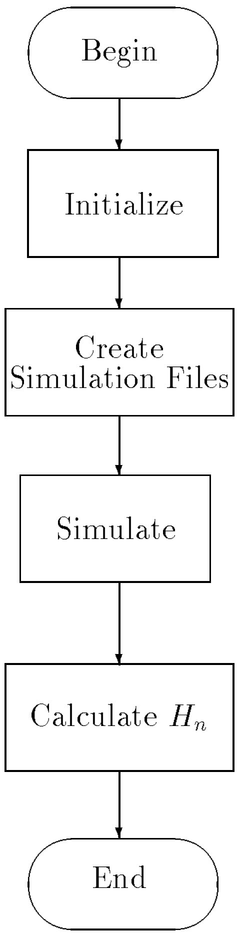

Figure 4.1: High level flow chart for the procedure to extract the nonlinear Volterra transfer functions using circuit simulation

4.2.1 Output Decomposition

The extraction procedure is based on the fact that the mapping of the nth order response, y_n is homogeneous of degree n with respect to the input x. That is, for the mapping $x \rightarrow y_n$, for any scalar α, then $\alpha x \rightarrow \alpha^n y_n$. This property can be seen by studying equation reqeq:volt.yn. As would be expected, this property also holds for the frequency domain description. Thus

$$\alpha X \rightarrow \alpha^n Y_n \tag{4.1}$$

Our procedure now is to pick several sets of "runs". Each run is defined by a single set of input frequencies and relative amplitudes of these frequencies. If a time–domain shooting method simulator is going to be used, then these frequencies must be commensurable. For each run, ν simulations are performed, where ν is greater than or equal to the assumed order, n , of the system. Each simulation has the same input frequencies, and the relative amplitudes and phases of the input frequencies are the same with respect to each other, but the total input signal amplitude is changed for each simulation. Thus the inputs of each simulation are only scalar multiples, α, of a reference input x. We will characterize each simulation amplitude by the scalar α_i so the input amplitudes for the simulations of a given run are $\alpha_1 x, \alpha_2 x, \ldots, \alpha_\nu x$. If we refer to the simulated responses of the system as r_i where $1 \leq i \leq \nu$, then

$$r_i = y_0 + \alpha_i y_1 + \alpha_i^2 y_2 + \ldots + \alpha_i^n y_n \tag{4.2}$$

Note here that r_i and α_i are known, and y_n is the unknown nth order system response as given in (3.43) for the known reference input x for a given run. Here y_0 is just the DC offset of the output. We can easily determine y_0 by simulating the system with a zero value input. In our procedure, this is the first simulation and the resulting DC output is subtracted from all subsequent simulations. For simplicity, we will drop the y_0 term and assume that this value has been subtracted from r_i.

We can also look at the frequency domain representation of (4.2)

$$R_{q,i} = \alpha_i Y_{q,1} + \alpha_i^2 Y_{q,2} + \ldots + \alpha_i^n Y_{q,n} \tag{4.3}$$

Here the capital letter indicates that the value is a complex number representing a phasor for a particular output frequency, ω_q. $R_{q,i}$ represents the measured output at frequency ω_q for the ith simulation of a given run. $Y_{q,j}$ is the jth order response of the output frequency ω_q of the system simulated with the reference input x. Since α is a real number, (4.3) can be easily separated into real and imaginary parts, or

$$\begin{aligned} \mathrm{Re}\{R_{q,i}\} &= \alpha_i \mathrm{Re}\{Y_{q,1}\} + \alpha_i^2 \mathrm{Re}\{Y_{q,2}\} + \ldots + \alpha_i^n \mathrm{Re}\{Y_{q,n}\} & (4.4) \\ \mathrm{Im}\{R_{q,i}\} &= \alpha_i \mathrm{Im}\{Y_{q,1}\} + \alpha_i^2 \mathrm{Im}\{Y_{q,2}\} + \ldots + \alpha_i^n \mathrm{Im}\{Y_{q,n}\} & (4.5) \end{aligned}$$

Using all ν simulations for a given run, we can express (4.3) in matrix form.

$$\begin{bmatrix} \mathrm{Re}\{R_{q,1}\} \\ \mathrm{Re}\{R_{q,2}\} \\ \mathrm{Re}\{R_{q,3}\} \\ \vdots \\ \mathrm{Re}\{R_{q,\nu}\} \end{bmatrix} = \begin{bmatrix} \alpha_1 & \alpha_1^2 & \alpha_1^3 & \ldots & \alpha_1^n \\ \alpha_2 & \alpha_2^2 & \alpha_2^3 & \ldots & \alpha_2^n \\ \alpha_3 & \alpha_3^2 & \alpha_3^3 & \ldots & \alpha_3^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_\nu & \alpha_\nu^2 & \alpha_\nu^3 & \ldots & \alpha_\nu^n \end{bmatrix} \begin{bmatrix} \mathrm{Re}\{Y_{q,1}\} \\ \mathrm{Re}\{Y_{q,2}\} \\ \mathrm{Re}\{Y_{q,3}\} \\ \vdots \\ \mathrm{Re}\{Y_{q,n}\} \end{bmatrix} \tag{4.6}$$

and

$$\begin{bmatrix} \mathrm{Im}\{R_{q,1}\} \\ \mathrm{Im}\{R_{q,2}\} \\ \mathrm{Im}\{R_{q,3}\} \\ \vdots \\ \mathrm{Im}\{R_{q,\nu}\} \end{bmatrix} = \begin{bmatrix} \alpha_1 & \alpha_1^2 & \alpha_1^3 & \dots & \alpha_1^n \\ \alpha_2 & \alpha_2^2 & \alpha_2^3 & \dots & \alpha_2^n \\ \alpha_3 & \alpha_3^2 & \alpha_3^3 & \dots & \alpha_3^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_\nu & \alpha_\nu^2 & \alpha_\nu^3 & \dots & \alpha_\nu^n \end{bmatrix} \begin{bmatrix} \mathrm{Im}\{Y_{q,1}\} \\ \mathrm{Im}\{Y_{q,2}\} \\ \mathrm{Im}\{Y_{q,3}\} \\ \vdots \\ \mathrm{Im}\{Y_{q,n}\} \end{bmatrix} \tag{4.7}$$

Thus for each run, the response at output frequency ω_q can be decomposed into the nth order responses by solving (4.6) and (4.7) for $Y_{q,i}$. This is equivalent to a polynomial curve fit of a real function. Of course the α–based matrix must not be singular, but we are free to choose any α. For small $n = \nu$ (≤ 3), the problem of choosing a set of α's such that the matrix is as well conditioned as possible has been solved in closed form [109], but for the general case the equations become too complicated to be practical. For our measurement procedure, we will use α_i's that alternate in sign and whose absolute values are equally spaced in the interval $0 < |\alpha_i| \leq 1$:

$$\alpha_i = -1^{(i+1)}\frac{i}{\nu} \tag{4.8}$$

For some sets of input frequencies, especially those with few distinct input frequencies, some output frequencies may have only even or odd order components. These cases are known *apriori* and thus we can reduce the size of the α matrix and simplify the decomposition. For example, for the case of a single input frequency (single–tone excitation), only the even order responses contribute to the first harmonic output frequency, and so (4.6) and (4.7) reduce to (assuming even n)

$$\begin{bmatrix} \mathrm{Re}\{R_{q,2}\} \\ \mathrm{Re}\{R_{q,4}\} \\ \mathrm{Re}\{R_{q,6}\} \\ \vdots \\ \mathrm{Re}\{R_{q,\nu}\} \end{bmatrix} = \begin{bmatrix} \alpha_1^2 & \alpha_1^4 & \alpha_1^6 & \dots & \alpha_1^n \\ \alpha_2^2 & \alpha_2^4 & \alpha_2^6 & \dots & \alpha_2^n \\ \alpha_3^2 & \alpha_3^4 & \alpha_3^6 & \dots & \alpha_3^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_\nu^2 & \alpha_\nu^4 & \alpha_\nu^6 & \dots & \alpha_\nu^n \end{bmatrix} \begin{bmatrix} \mathrm{Re}\{Y_{q,2}\} \\ \mathrm{Re}\{Y_{q,4}\} \\ \mathrm{Re}\{Y_{q,6}\} \\ \vdots \\ \mathrm{Re}\{Y_{q,n}\} \end{bmatrix} \tag{4.9}$$

and

$$\begin{bmatrix} \mathrm{Im}\{R_{q,2}\} \\ \mathrm{Im}\{R_{q,4}\} \\ \mathrm{Im}\{R_{q,6}\} \\ \vdots \\ \mathrm{Im}\{R_{q,\nu}\} \end{bmatrix} = \begin{bmatrix} \alpha_1^2 & \alpha_1^4 & \alpha_1^6 & \dots & \alpha_1^n \\ \alpha_2^2 & \alpha_2^4 & \alpha_2^6 & \dots & \alpha_2^n \\ \alpha_3^2 & \alpha_3^4 & \alpha_3^6 & \dots & \alpha_3^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_\nu^2 & \alpha_\nu^4 & \alpha_\nu^6 & \dots & \alpha_\nu^n \end{bmatrix} \begin{bmatrix} \mathrm{Im}\{Y_{q,2}\} \\ \mathrm{Im}\{Y_{q,4}\} \\ \mathrm{Im}\{Y_{q,6}\} \\ \vdots \\ \mathrm{Im}\{Y_{q,n}\} \end{bmatrix} \tag{4.10}$$

4.2.2 Transfer Function Calculation

Our goal is to calculate the Volterra nonlinear transfer function H_n for all n. Now that we know the ith order output response at frequency ω_q, $Y_{q,i}$, and the input X, the problem is to calculate the appropriate H's. In order to understand the problems of this calculation, we need to look at how $Y_{q,i}$ can be calculated from H_i and X. From (3.65) we know that the *nth* order steady–state response at an output frequency of ω_q of a system characterized by Volterra nonlinear transfer functions can be represented in the form

$$Y_{q,n} = \sum_j^J \beta_j H_n^{(j)} \prod_i^n X_{i,j} \tag{4.11}$$

Where the arguments to $H_n^{(j)}$ are the frequencies corresponding to the frequencies associated with $X_{i,j}$. We will refer to a given set of these frequencies as a Frequency Intermodulation Product Description or FIPD. Thus an FIPD is just the arguments to the Volterra nonlinear transfer function consisting of n frequencies. The sum of the frequencies in an FIPD will be the value of the output frequency.

$$\sum_1^n \omega_i = \omega_q \tag{4.12}$$

Note that the frequencies in an FIPD can be negative. Thus J is the number of FIPD's of order n that contribute to the ω_q. The value of β_j is known and a function of the FIPD and is equivalent to the $(n+2\alpha)!/(s_k!(|n_k|+s+k)!)$ term in (3.116). The product $\prod_i^n X_{i,j}$ is the set of input phasors that correspond to the appropriate components of an FIPD for the output frequency ω_q. Letting $W^{(j)} = \beta_j \prod_i^n X_{i,j}$ represent the known values of the equation, (4.11) becomes

$$Y_{q,n} = \sum_{j=1}^J H^{(j)} W^{(j)} \tag{4.13}$$

Letting an re subscript indicate the real part of the complex number and an im subscript represent the imaginary part of a complex number, we can express (4.13) as a real matrix equation.

$$\begin{bmatrix} W_{re}^{(1)} & -W_{im}^{(1)} & W_{re}^{(2)} & -W_{im}^{(2)} & \cdots & W_{re}^{(J)} & -W_{im}^{(J)} \\ W_{im}^{(1)} & W_{re}^{(1)} & W_{im}^{(2)} & W_{re}^{(2)} & \cdots & W_{im}^{(J)} & W_{re}^{(J)} \end{bmatrix} \begin{bmatrix} H_{re}^{(1)} \\ H_{im}^{(1)} \\ H_{re}^{(2)} \\ H_{im}^{(2)} \\ \vdots \\ H_{re}^{(J)} \\ H_{im}^{(J)} \end{bmatrix} = \begin{bmatrix} Y_{q,n,re} \\ Y_{q,n,im} \end{bmatrix} \tag{4.14}$$

For $J = 1$, the number of knowns equal the number of unknowns and we can solve for $H_{re}^{(1)}$ and $H_{im}^{(1)}$ as long as the resulting 2×2 matrix is not singular. For $J > 1$, we need either to know some of H, or we need to add some rows to the matrix, or a combination of both.

We will now examine when the number of unknowns in (4.14) is greater than the number of rows in the left matrix. For any first order system, the output at any frequency is due only to a single input frequency, thus the number of unknowns for a single output frequency is at most two, one for $\text{Im}\{H_1\}$ and one for $\text{Re}\{H_1\}$.

	Transfer Function	FIPD	Order	Output Frequency
(a)	H_0		0	0
(b)	$H_1(1)$	1	1	1
(c)	$H_1(2)$	2	1	2
(d)	$H_2(-1,1)$	-1 1	2	0
(e)	$H_2(-2,2)$	-2 2	2	0
(f)	$H_2(1,1)$	1 1	2	2
(g)	$H_2(-1,2)$	-1 2	2	1
(h)	$H_2(1,2)$	1 2	2	3
(i)	$H_2(2,2)$	2 2	2	4

Table 4.1: All FIPD's for a second order system with a discrete input spectrum of 1Hz and 2Hz.

The exception to this is the DC or zero output frequency point. It can have two components, one from H_0 that is not associated with any input frequency, and one from $H_1(0)$ if a DC input is present. However, we can easily determine the H_0 value by a single simulation with no inputs, but we must do this before we try to calculate $H_1(0)$. Thus the order that we determine the values of H is important.

For systems of order less than or equal to two, (4.14) can always be solved by using previously calculated values of H if the correct runs (i.e. sets of input frequencies) are simulated and the Volterra nonlinear transfer functions calculated in the correct order. As with the first order case above, the value of H_0 must be calculated before $H_1(0)$ can be found. The next question is what values of H_1 are needed before some value of H_2 can be calculated, and how can one in general always have the needed values of H_1 available when calcuating H_2. This can easily be accomplished by picking which simulations are to be performed and in which order. All of the single–tone inputs are simulated first, then the two–tone inputs are simulated. Since each FIPD is limited to at most two frequencies, for the case of a second order system, more than two tones are not required to find any value of the two transfer functions H_1 and H_2. For instance, suppose that we were trying to calculate the values of H for a second order system in order to be able to predict the response of that system for any phase and amplitude combination of 1 Hz and 2 Hz. The needed values of the Volterra nonlinear transfer functions would be as shown in table 4.1.

In this example the four different runs indicated in Table 4.2 are required. Here run (A) has no input, (B) has a single 1 Hz input, (C) has a single 2 Hz input, and (D) had both 1 Hz and 2 Hz inputs. Run (A) would determine H_0. Run (B) would have two output frequencies, 0 Hz and 2 Hz. The 0 Hz output component would be a sum of the effects of (a) and (f), but run (A) already determined the value of H_0, thus leaving only $H_2(-1,1)$ unknown. Run (C) would also have two output frequencies, 0 Hz and 4 Hz. As before, the 0 Hz output component would be a sum

	Input Frequencies	Output Frequencies
(A)	no input	0
(B)	1 Hz	0, 1, 2
(C)	2 Hz	0, 2, 4
(D)	1 Hz and 2 Hz	0, 1, 2, 3, 4

Table 4.2: List of simulation runs for extracting all Volterra nonlinear transfer functions points needed to calculate an output for a second order system with 1 Hz and 2 Hz input phasors

of two FIPD's, one from (a) and one from (e). And again H_0 is already known, leaving just $H_2(-2, 2)$. However, run (D) is more complicated. The output has all of the components due to all of the FIPD's. But careful examination indicates that at most only one value of H is now unknown for each output frequency. All of the values of H contributing to an output frequency of 0 Hz (a,d,e) are already known. For the output frequency of 1 Hz, only $H_2(-1, 2)$ is unknown. Both of the values of H for an output frequency of 2 Hz are known. And there is only one FIPD for each of the remaining output frequencies 3 Hz and 4 Hz. This method of measuring transfer functions for low order systems is referred to as the harmonic probing method and previously has been successfully used to measure systems of order less than or equal to two [107].

For assumed system order of a value greater than two, the problem becomes more complex. There are four situations where care must be taken when trying to solve (4.14), balanced DC FIPD, unbalanced DC FIPD, FIPD with an unbalanced DC FIPD subset, and other unique FIPD's with the same order and frequencies.

Balanced DC FIPD

When an FIPD is balanced with respect to zero (e.g. $H_4(-2, -1, 1, 2)$), the value of the nonlinear transfer function will be real. To be a balanced FIPD, each positive frequency must have a corresponding negative frequency in the FIPD. For this case we know that the value of H must be real. See Appendix B, section B.5 for further discussion. This case of a pure real transfer function is not a problem as then (4.14) will have the form

$$\begin{bmatrix} W_{re}^{(1)} & 0 \\ 0 & W_{re}^{(1)} \end{bmatrix} \begin{bmatrix} H_{re}^{(1)} \\ H_{im}^{(1)} \end{bmatrix} = \begin{bmatrix} Y_{q,n,re} \\ 0 \end{bmatrix} \tag{4.15}$$

We can see that $H_{im}^{(1)} = 0$. Roundoff errors or other inaccuracies in the simulation and Fourier transform process can cause $W_{im}^{(1)}$ and $Y_{q,n,im}$ to be very small nonzero numbers. Since for this case we know ahead of time by inspecting the FIPD that $H_{im}^{(1)} = 0$, we just set it to zero and calculate $H_{re}^{(1)} = Y_{q,3,re} \div W_{re}^{(1)}$.

Unbalanced DC FIPD

For orders greater than two, there can be unbalanced arguments to the nonlinear transfer function that add to zero. In these cases, the transfer function will not in general be real. Of course, since the output frequency is zero, the imaginary part of the measured output will be zero, ($Y_{q,n,im} = 0$). Furthermore, the value of $W^{(1)}$ can be complex.

For example, the third order DC response of a system with input of three sinusoids of frequencies 1 Hz, 2 Hz, and 3 Hz, the unknown values of H_3 are $H_3(-2,-1,3)$ and $H_3(-3,2,1)$. The other points contributing to the DC output of the system, $H_3(-2,1,1)$, $H_3(-1,-1,2)$ can be calculated from previous runs using simpler simulations with inputs of only two frequencies, 1 Hz and 2 Hz. $H_3(-2,-1,3)$ can be in general complex, but both $H^{(1)} = H_3(-2,-1,3)$ and $H^{(2)} = H_3(-3,1,2)$ contribute to the third order DC output. We also know that for this case $H_3(-2,-1,3) = H_3^*(-3,1,2)$ and $W^{(1)} = (W^{(2)})^*$. (See appendix B for more information). Thus it works out that $H^{(1)}W^{(1)} + H^{(2)}W^{(2)} = 2\text{Re}\{H^{(1)W^{(1)}}\}$. Therefore (4.14) will have the form

$$\begin{bmatrix} W_{re}^{(1)} & -W_{im}^{(1)} & W_{re}^{(1)} & W_{im}^{(1)} \\ W_{im}^{(1)} & W_{re}^{(1)} & -W_{im}^{(1)} & W_{re}^{(1)} \end{bmatrix} \begin{bmatrix} H_{re}^{(1)} \\ H_{im}^{(1)} \\ H_{re}^{(1)} \\ -H_{im}^{(1)} \end{bmatrix} = \begin{bmatrix} Y_{q,3,re} \\ 0 \end{bmatrix} \tag{4.16}$$

Which reduces to

$$\begin{bmatrix} 2W_{re}^{(1)} & -2W_{im}^{(1)} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} H_{re}^{(1)} \\ H_{im}^{(1)} \end{bmatrix} = \begin{bmatrix} Y_{q,3,re} \\ 0 \end{bmatrix} \tag{4.17}$$

Note that we cannot solve (4.17) for $H_{re}^{(1)}$ and $H_{im}^{(1)}$. We need more information. To solve this problem we can add two additional rows to the matrix equation. These two rows would represent an additional run using the same input frequencies. The additional run however, must be different from the first run. The two things we can change and still keep the same input frequencies are the relative amplitudes of the input signals and the relative phases of the input signals. Careful examination of (4.14) will show that system will still not be solvable if we change the relative amplitudes of the input signals. For example, let us assume that our original run had equal amplitudes of each of the three signals, 1 Hz, 2 Hz, and 3 Hz. If we double the amplitude of the 1 Hz signal and half the amplitude of the 2 Hz signal, then the response associated with $H_3(-2,-1,3)$ and $H_3(-3,1,2)$ will not change. This is because the W's will be the same. Thus the two rows we added to (4.14) are identical to the first two. If we double just the 1 Hz input signal and leave the 2 Hz and 3 Hz the same, then the two rows that we add will just be a copy of the first two rows multiplied by 2.

If, on the other hand, we change the relative phases of the input signals, say shift the 1 Hz signal by 90 degrees, then the value of $W^{(1)}$ will have the same amplitude but be shifted in phase by -90 degrees, and the value of $W^{(2)}$ will have the same amplitude but be shifted in phase by 90 degrees. Thus the matrix equation to be solved will be of the form

$$\begin{bmatrix} W_{re}^{(1)} & -W_{im}^{(1)} & W_{re}^{(1)} & W_{im}^{(1)} \\ W_{im}^{(1)} & W_{re}^{(1)} & -W_{im}^{(1)} & W_{re}^{(1)} \\ W_{im}^{(1)} & W_{re}^{(1)} & -W_{im}^{(1)} & W_{re}^{(1)} \\ -W_{re}^{(1)} & W_{im}^{(1)} & -W_{re}^{(1)} & -W_{im}^{(1)} \end{bmatrix} \begin{bmatrix} H_{re}^{(1)} \\ H_{im}^{(1)} \\ H_{re}^{(1)} \\ -H_{im}^{(1)} \end{bmatrix} = \begin{bmatrix} Y_{q,3,re} \\ 0 \\ Y'_{q,3,re} \\ 0 \end{bmatrix} \tag{4.18}$$

which reduces to

$$\begin{bmatrix} 2W_{re}^{(1)} & -2W_{im}^{(1)} \\ 0 & 0 \\ -2W_{re}^{(1)} & -2W_{im}^{(1)} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} H_{re}^{(1)} \\ H_{im}^{(1)} \end{bmatrix} = \begin{bmatrix} Y_{q,3,re} \\ 0 \\ Y'_{q,3,re} \\ 0 \end{bmatrix} \tag{4.19}$$

or

$$\begin{bmatrix} 2W_{re}^{(1)} & -2W_{im}^{(1)} \\ -2W_{re}^{(1)} & -2W_{im}^{(1)} \end{bmatrix} \begin{bmatrix} H_{re}^{(1)} \\ H_{im}^{(1)} \end{bmatrix} = \begin{bmatrix} Y_{q,3,re} \\ Y'_{q,3,re} \end{bmatrix} \tag{4.20}$$

which can be solved.

FIPD with Unbalanced DC FIPD subset

The case of an FIPD which contains a subset of an unbalanced DC FIPD is very similar to the case of an unbalanced DC FIPD. Each FIPD of this type will have a corresponding mate FIPD which is the same except that the frequencies of the unbalanced subset have been multiplied by -1. Since it takes at least 3 frequencies to make an unbalanced DC FIPD, these types of FIPD's must be at least order 4. For example, both $H_4(-2,-1,3,4)$ and $H_4(-3,1,2,4)$ contain the unbalanced DC FIPD $\pm(-2,-1,3)$. As before, the matrix equation cannot be solved without adding an additional run. The new run should be such that the resulting W phasor of the unbalanced DC FIPD is rotated by 90 degrees with respect to the other FIPD. In this example, shifting the phase of the 4 Hz input would not work since it has the same sign in both FIPD's.

Other unique FIPD's with same order and frequencies

Beginning with third order, there are some circumstances where there can be more than one FIPD of the same order, which contain the same frequency magnitudes, and have the same output frequency (i.e. add to the same positive frequency). For example, both $H^{(1)} = H(-1,3,3)$ and $H^{(2)} = H(1,1,3)$ contain only the frequencies 1 and 3, and both add to 5. For this example, (4.14) would be of the form

$$\begin{bmatrix} W_{re}^{(1)} & -W_{im}^{(1)} & W_{re}^{(2)} & -W_{im}^{(2)} \\ W_{im}^{(1)} & W_{re}^{(1)} & W_{im}^{(2)} & W_{re}^{(2)} \end{bmatrix} \begin{bmatrix} H_{re}^{(1)} \\ H_{im}^{(1)} \\ H_{re}^{(2)} \\ H_{im}^{(2)} \end{bmatrix} = \begin{bmatrix} Y_{q,3,re} \\ Y_{q,3,im} \end{bmatrix} \tag{4.21}$$

Here all four values of H are unknown and we have no additional information. As before we must simulate an additional run in order to calculate the transfer function values. For this case we choose to vary the relative amplitudes of the input frequency components, instead of the relative phases. For our example, let us add an additional run with the 3 Hz signal increased in amplitude by 1/3 and the 1 Hz signal decreased in amplitude by 1/3. Assuming the first run had both signals at equal amplitude, then the peak value of the second run will be the same, and the ratio of the 3 Hz to 1 Hz amplitude will be 1/2. The amplitude of $W^{(1)}$ will increase by a factor of $\frac{2}{3} \times \frac{4}{3} \times \frac{4}{3} = \frac{32}{27}$ and the amplitude of $W^{(2)}$ will decrease by a factor of $\frac{2}{3} \times \frac{2}{3} \times \frac{4}{3} = \frac{16}{27}$. Now (4.21) becomes

$$\begin{bmatrix} W^{(1)}_{re} & -W^{(1)}_{im} & W^{(2)}_{re} & -W^{(2)}_{im} \\ W^{(1)}_{im} & W^{(1)}_{re} & W^{(2)}_{im} & W^{(2)}_{re} \\ \frac{32}{27}W^{(1)}_{re} & -\frac{32}{27}W^{(1)}_{im} & \frac{16}{27}W^{(2)}_{re} & -\frac{16}{27}W^{(2)}_{im} \\ \frac{32}{27}W^{(1)}_{im} & \frac{32}{27}W^{(1)}_{re} & \frac{16}{27}W^{(2)}_{im} & \frac{16}{27}W^{(2)}_{re} \end{bmatrix} \begin{bmatrix} H^{(1)}_{re} \\ H^{(1)}_{im} \\ H^{(2)}_{re} \\ H^{(2)}_{im} \end{bmatrix} = \begin{bmatrix} Y_{q,3,re} \\ Y_{q,3,im} \\ Y'_{q,3,re} \\ Y'_{q,3,im} \end{bmatrix} \tag{4.22}$$

Note that if we have more runs than we have unknown FIPD's, then (4.14) will be underconstrained, the number of equations is greater than the number of unknowns. In our algorithm, we would actually add two more runs for the case of $H^{(1)} = H(-1,3,3)$ and $H^{(2)} = H(1,1,3)$, one case where the 3 Hz signal was increased and one case where the 1 Hz signal was increased. This poses no mathematical problem, since we solve (4.14) using least squares techniques.

Maximum Input Amplitude and Assumed Order of System

Since the set of extracted nonlinear transfer functions is intended to be used to simulate the subcircuit as a black box, it is important to know the maximum peak amplitude in the black box simulation. If the amplitude is very small, it may be sufficient to extract only the first order transfer function. For larger input amplitudes, and for increasingly nonlinear circuits, the extraction needs to include the higher order transfer functions. Furthermore, it is important to extract all of the transfer functions in the same procedure. In other words, one should not extract, say the first and second order functions, and then go back and start the procedure again extracting just the third and fourth order functions while keeping the same first and second order functions extracted in the first procedure. Remember that when we solve (4.6), we are essentially performing a polynomial least squares fit. In order to get the best fit and thus the most accurate behavioral model over the entire input range, we need to extract all transfer functions in the same procedure.

On the other hand, if one wanted to trade off accuracy in one amplitude range for accuracy in another amplitude range, then one could use the two procedure method. For example, suppose the behavioral model were to be used primarily in two modes, small signal (i.e. linear) simulation, and large signal (i.e. nonlinear) simulation. The small signal simulation would be much faster and would only be valid for small inputs to the circuit. Whereas the large signal simulation would accurately simulate the nonlinear effects. For this case, another possibility would be to use standard AC simulation to extract the first order transfer function. This would be much faster than the running of the extraction procedure and specifying order=1. Section 5.5 will show example results of this process.

Another important aspect of the extraction procedure concerns the peak amplitude of the test signal. We want to be sure that the test signals used in the extraction procedure have a peak absolute value amplitude equal to that specified by the user at the beginning of the extraction process. This is to ensure that the extracted model is valid for the range for which it will be used. Thus for each run, the peak input amplitude absolute value will equal that specified by the user. This is easily accomplished by searching the input amplitude waveform to find the peak (absolute value) and then scaling the input waveform accordingly. But we also want to maximize the energy to the circuit to get the maximum energy output and thus obtain a more accurate extraction. For any measurement or simulation system, there will be some unwanted noise injected into the measurement or simulation. Larger power input signals help to reduce the effects of this noise.

In order to maximize input energy, we adjust the relative phases of the input tones to minimize the absolute value of the input peak before we scale the signal as discussed above. In our procedure, we will use tree annealing [90] to accomplish this minimization.

4.2.3 Algorithm Flow Description

First we will examine the box labeled "Initialize" in figure 4.1. Figure 4.2 gives a more detailed description of this block. First the data from a file created by the user is read. A sample of this file is given in figure 4.3. The first line of the file is a description that is used to annotate other files. The second line specifies the maximum input value to the circuit. The value is the peak absolute voltage of the input. The maximum order specified on the next line is the assumed order of the system. The next line specifies the number of additional simulations to be run for each set of relative input phasors. As explained in section 4.2, the given set of input frequencies will be simulated n times, each with a different input amplitude, but with the same relative input phasors. The output is decomposed into the various mth order responses. The decomposition is basically a series of polynomial curve fits where the polynomial has the order equal to the assumed order of the system. The number of different amplitudes, n, must therefore be equal to or greater than the assumed order of the system. The number of extra simulations is just $m - n$, which must be a non–negative integer. A larger number will require more simulations and more extraction calculations. If no extra simulations are used, then the number of points for the polynomial would equal the order of the polynomial (the zeroth order component being known) and thus the polynomial would always pass through all of the points given. If the number of different input amplitude simulations is greater than the order of the system then there are two advantages:

1. The measurement noise will have a reduced effect on the calculation. Even though the circuit output voltage used is from a simulation as opposed to a lab measurement, there is still noise present in the values.

2. The value of the assumed order can be verified. Though this is not a rigorous verification, the goodness of fit of the oversampled polynomial fit is an

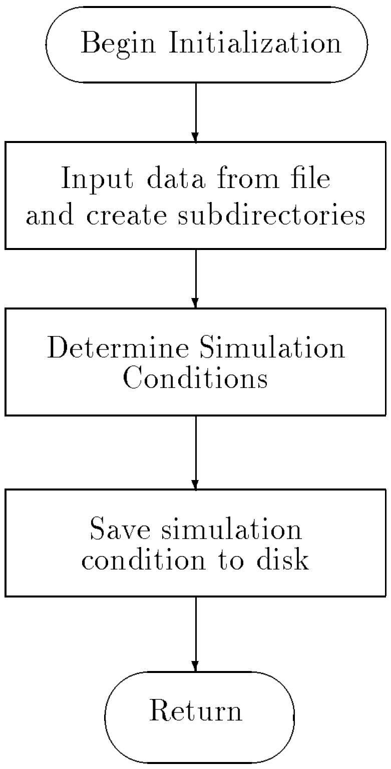
Begin Initialization
Input data from file and create subdirectories
Determine Simulation Conditions
Save simulation condition to disk
Return

Figure 4.2: High level flow chart for the procedure to initialize the extraction procedure. This flow chart represents the box labeled “Initialize” in figure 4.1

```
low pass filter with pole at 3MHz
max_input: 1.0
max_order: 3
xtra_runs: 1
fn: lp1
test_model_lib: /u/lunsford/asxlib/lowpass
runcontrol: /u/lunsford/asxlib/stdrun
acruncontrol: /u/lunsford/asxlib/acrun
loadlib: /u/lunsford/asxlib/libspi31.o
num_extract_runs: -1
tot_runs: 18
run_num: -1
calc_dc: yes
out_dc_offset: 0
num_freq: 4
freq[0]: 0
freq[1]: 1000
freq[2]: 2000
freq[3]: 3000
```

Figure 4.3: The input file for low–pass RC filter.

indication of the validity of the assumed order.

The next five lines of figure 4.3 are related to the simulator used, ASTAP [113]. The first is the file name containing the circuit description, the second is the name of the directory that contains this file and other files that contain needed subcircuit models. The fourth and fifth are file names that contain run control statements for the simulations. And the last of the five lines gives the name of a file that contains compiled fortran used to model the nonlinear characteristics of the circuit elements.

The next three lines, num_extract_runs, tot_runs, and run_num, are set by the program as it goes through the various steps and are not set by the user. These values are written to the file during the initialization phase of the program. The following two lines, calc_dc and out_dc_offset are used to calculate the value of H_0, which is just DC output voltage for the circuit when the input value is a constant value of zero. If calc_dc is set to "yes", then the value of out_dc_offset will be calculated by the program by running a simulation with an input voltage of zero volts. If the value of calc_dc is set to "no", then the program will take the value entered for out_dc_offset and use it as the value of H_0.

Lastly, the discrete values of the assumed input frequencies are given, first the number of frequencies, then each frequency in hertz.

The third block in figure 4.2 is for determining how many simulations are needed and what the input conditions will be for each simulation. This is a complicated procedure and is shown in more detail in figure 4.4. The list of basic runs created in the first block is just all the different possible combinations of the input frequencies in groups of $1, 2, \ldots n$ where n is the assumed order of the system. Table 4.3 gives an example of the basic runs that correspond to figure 4.3. Run 0 is used to determine the DC operation point, or H_0. For table 4.3, we have chosen the amplitudes of each input phasor to be the maximum allowed amplitude divided by the number of input phasors, thus guaranteeing that the peak amplitude will not be greater than the maximum specified in the "max_input:" field in the input file. Table 4.3 gives just a starting point for determining all of the inputs for the simulations. There are four adjustments that are made to this table. They are, (a) adjusting the relative phases of the input phase for the minimum peak value, (b) adding extra runs, (c) shifting the input in time such that the average value of the signal occurs at time equal to zero and (d) adjusting input amplitudes for exact peak value.

Table 4.4 is table 4.3 with the addition of the extra runs needed to calculate all of the nonlinear transfer functions. Run 8 has been duplicated with a 90 degree phase change in the 1 MHz signal because run 8 determines the H value at an unbalanced DC FIPD, $(-1, -1, 2)$. Run 9 is duplicated with different input phasor amplitudes because $H_3(1, 1, 3)$ and $H_3(-1, 3, 3)$ are uniquely determined only with two runs as they both have the same output frequency. This type of situation was discussed above on page 60 under the heading "Other unique FIPD's with same order and frequencies". Run 14 determines an H value at an unbalanced DC FIPD, $(-2, -1, 3)$, thus a second simulation was added that shifted one of the input phasors by 90 degrees. The actual algorithm used to determine the cases where additional runs are required is shown in figures 4.5 and 4.6.

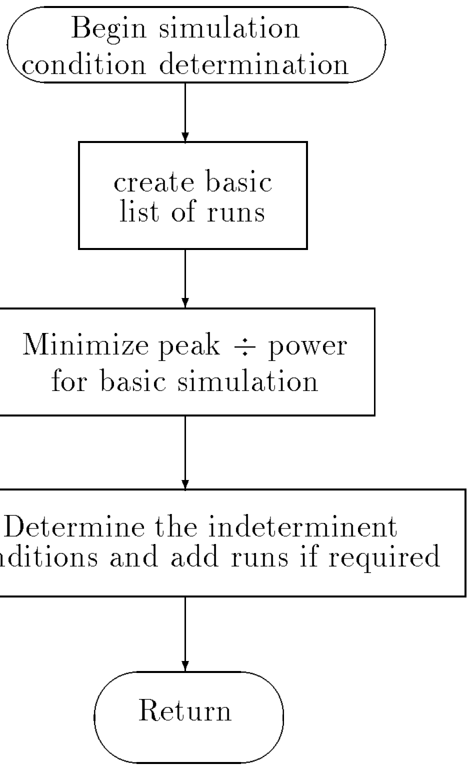
Begin simulation condition determination
create basic list of runs
Minimize peak ÷ power for basic simulation
Determine the indeterminent conditions and add runs if required
Return

Figure 4.4: Flow chart for determining simulation conditions. This represents the block labeled “Determine Simulation Conditions” in figure 4.2.

Run Number	Input Amplitudes							
	0 Hz		1 MHz		2 MHz		3 MHz	
	Real	Imag.	Real	Imag.	Real	Imag.	Real	Imag.
0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	0
3	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0	1
5	0.5	0	0	0.5	0	0	0	0
6	0.5	0	0	0	0	0.5	0	0
7	0.5	0	0	0	0	0	0	0.5
8	0	0	0	0.5	0	0.5	0	0
9	0	0	0	0.5	0	0	0	0.5
10	0	0	0	0	0	0.5	0	0.5
11	0.3333	0	0	0.3333	0	0.3333	0	0
12	0.3333	0	0	0.3333	0	0	0	0.3333
13	0.3333	0	0	0	0	0.3333	0	0.3333
14	0	0	0	0.3333	0	0.3333	0	0.3333

Table 4.3: Basic list of runs for third order and f =0, 1MHz, 2MHz, 3MHz. without peak minimization, without additional needed runs, without time shift, and without amplitude adjustment.

Run	Input Amplitudes							
	0 Hz		1 MHz		2 MHz		3 MHz	
Number	Real	Imag.	Real	Imag.	Real	Imag.	Real	Imag.
0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	0
3	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0	1
5	0.5	0	0	0.5	0	0	0	0
6	0.5	0	0	0	0	0.5	0	0
7	0.5	0	0	0	0	0	0	0.5
8	0	0	0.5	0.0000	0	0.5	0	0
9	0	0	-0.4330	-0.2501	0	0	0	0.5
10	0	0	0	0	0.0001	0.5	0	0.5
11	0.3333	0	0.2357	-0.2357	0	0.3333	0	0
12	0.3333	0	-0.2886	-0.1667	0	0	0	0.3333
13	0.3333	0	0	0	-0.1667	-0.2886	0	0.3333
14	0	0	0.2232	-0.2476	0.0525	-0.3292	0	0.3333

Table 4.4: Basic list of runs for third order and f =0, 1MHz, 2MHz, 3MHz. with peak minimization, without additional needed runs, without time shift, and without amplitude adjustment. A value of 0.0000 indicates a nonzero number whose absolute value is less than 0.0001.

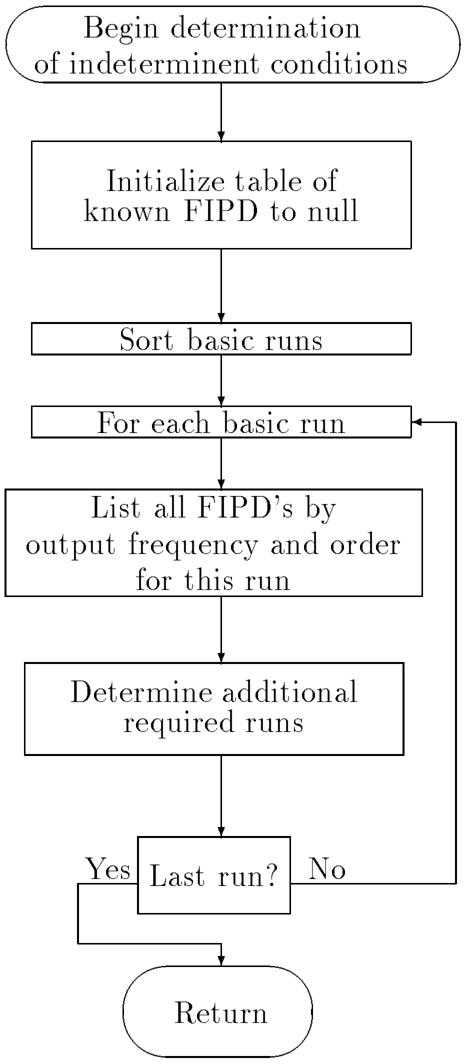
Begin determination of indeterminent conditions
Initialize table of known FIPD to null
Sort basic runs
For each basic run
List all FIPD's by output frequency and order for this run
Determine additional required runs
Last run?
Yes
No
Return

Figure 4.5: Flow chart for determining indeterminant conditions. This represents the block labeled "Determine the indeterminent conditions" in figure 4.4.

Run	Input Amplitudes							
	0 Hz		1 MHz		2 MHz		3 MHz	
Number	Real	Imag.	Real	Imag.	Real	Imag.	Real	Imag.
0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	0
3	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0	1
5	0.5	0	0	0.5	0	0	0	0
6	0.5	0	0	0	0	0.5	0	0
7	0.5	0	0	0	0	0	0	0.5
8a	0	0	0.5	0.0000	0	0.5	0	0
8b	0	0	0.3535	0.3535	0	0.5	0	0
9a	0	0	-0.4330	-0.2501	0	0	0	0.5
9b	0	0	-0.5773	-0.3335	0	0	0	0.3333
9c	0	0	0	-0.2886	-0.1667	0	0	0.6666
10	0	0	0	0	0.0001	0.5	0	0.5
11	0.3333	0	0.2357	-0.2357	0	0.3333	0	0
12	0.3333	0	-0.2886	-0.1667	0	0	0	0.3333
13	0.3333	0	0	0	-0.1667	-0.2886	0	0.3333
14a	0	0	0.2232	-0.2476	0.0525	-0.3292	0	0.3333
14b	0	0	0.2232	-0.2476	0.3292	0.0525	0	0.3333

Table 4.5: List of runs for third order and f =0, 1MHz, 2MHz, 3MHz. with peak minimization, with additional needed runs, without time shift, and without amplitude adjustment. A value of 0.0000 indicates a nonzero number whose absolute value is less than 0.0001.

Table 4.5 gives the values of the input phasors of table 4.4 after the phase has been adjusted to maximize the input power relative to the maximum peak amplitude. Since we will later adjust the overall amplitude, the step here is just a minimization of the peak voltage as a function of the relative phases of the input vectors. Tree annealing [90] is well suited for the phase adjustment and is used in this procedure. Note that the runs added in table 4.4 required a certain phase relationship to another run, e.g. the phase of the 1 MHz phasor in run 8b is the phase of the 1 MHz phasor in run 8a shifted by 90 degrees. Thus the same relative phase adjustment for maximizing power calculated for 8a is used in 8b. Table 4.6 gives table 4.5 with the input delayed such that the value at time equal zero is the average value of the signal. This is equivalent to shifting the time such that the value of all of the non–DC input phasors add to zero. This adjustment can help to speed the

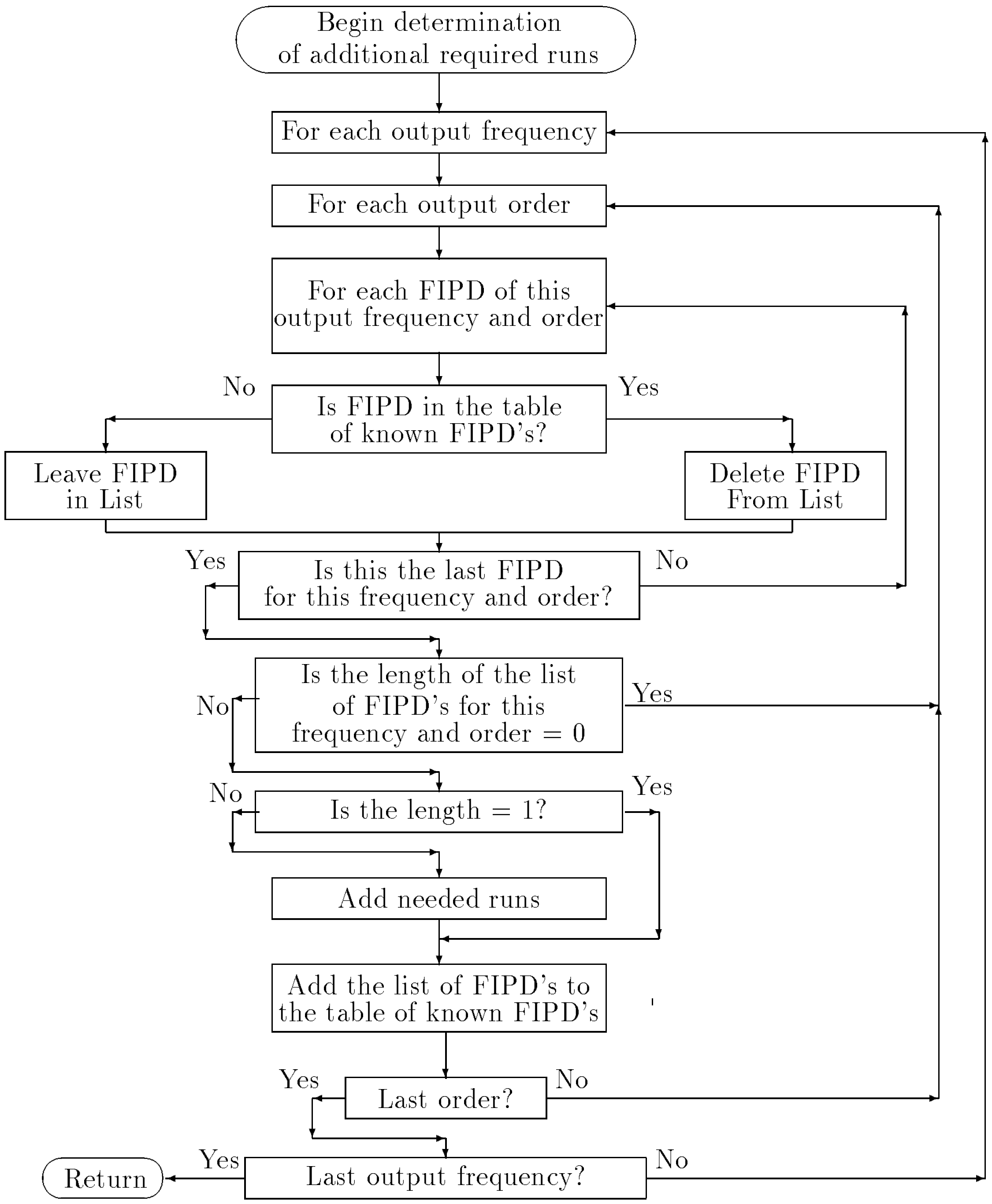
Begin determination of additional required runs
For each output frequency
For each output order
For each FIPD of this output frequency and order
No
Is FIPD in the table of known FIPD's?
Yes
Leave FIPD in List
Delete FIPD From List
Yes
Is this the last FIPD for this frequency and order?
No
No
Is the length of the list of FIPD's for this frequency and order = 0
Yes
No
Is the length = 1?
Yes
Add needed runs
Add the list of FIPD's to the table of known FIPD's
Yes
Last order?
No
Return
Yes
Last output frequency?
No

Figure 4.6: Flow chart for determining additional required runs. This represents the block labeled "Determine additional required runs" in figure 4.5.

Run Number	Input Amplitudes							
	0 Hz		1 MHz		2 MHz		3 MHz	
	Real	Imag.	Real	Imag.	Real	Imag.	Real	Imag.
0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	0
3	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0	1
5	0.5	0	0	0.5	0	0	0	0
6	0.5	0	0	0	0	0.5	0	0
7	0.5	0	0	0	0	0	0	0.5
8a	0	0	0.5	0.0000	0.0000	0.5	0	0
8b	0	0	0.3535	0.3535	0.0000	0.5	0	0
9a	0	0	-0.4330	-0.2501	0	0	0	0.5
9b	0	0	-0.5773	-0.3335	0	0	0	0.3333
9c	0	0	-0.2886	-0.1667	0	0	0	0.6666
10	0	0	0	0	0.0001	0.5	0	0.5
11	0.3333	0	0.2357	-0.2357	0	0.3333	0	0
12	0.3333	0	-0.2886	-0.1667	0	0	0	0.3333
13	0.3333	0	0	0	-0.1667	-0.2886	0	0.3333
14a	0	0	0.2232	-0.2476	0.0525	-0.3292	0	0.3333
14b	0	0	0.2232	-0.2476	0.3292	0.0525	0	0.3333

Table 4.6: List of runs for third order and f =0, 1MHz, 2MHz, 3MHz. with peak minimization, with additional needed runs, with time shift, and without amplitude adjustment. A value of 0.0000 indicates a nonzero number whose absolute value is less than 0.0001.

simulations if a shooting method simulator is used. If the circuit being simulated has widely varying time constants, and if the input frequencies are high compared to the inverse of the smallest time constant of the circuit, then this adjustment can significantly reduce the simulation time. This reduction in simulation time can be understood by looking at the example of a series resistor–capacitor pair in the circuit. Given that the pole associated with the RC pair is much lower in frequency than the input frequencies, then the voltage across that capacitor will remain relatively steady through the simulation. Since shooting methods first solve for the DC solution to the circuit using the input conditions at time equal zero, the value of the voltage across our example capacitor will often be the same for the DC solution as for the steady–state solution. Thus if we adjust the time such that the DC solution at time equal zero is the average, we can reduce the simulation time for some circuits if a shooting method simulator is used. This will be true for the case of a linear circuit where the input frequencies are much higher than the pole associated with the example resistor-capacitor pair, but for the general case of nonlinear circuits, this is not guaranteed. For memoryless circuits, this time shift has no effect on the simulation time.

Table 4.7 gives table 4.6 with the input phasor amplitudes adjusted so that the peak absolute value matches the value of the "max_input:" field in the input file. This adjustment is accomplished by calculating all of the time–domain values of the input at a sufficiently small time step and taking the largest absolute value. This value is then compared to the value given in the input file and the phasor amplitudes are adjusted accordingly.

Now that the "Initialize" block in figure 4.1 has been completed, the next step is to create the simulation files. These files depend on the simulator to be used and have the input conditions as described by the runs calculated in the previous step. Note that for each run, several simulations are performed, all with the same relative phases, but with different input amplitudes. The last simulation for each run is the one with largest input amplitude and the peak absolute value of the input for this simulation equals that specified in the input file.

After the simulation files are created, all of the simulation files are simulated. This section uses the majority of the computor resources and can take days or weeks depending on several factors. The biggest factor is the speed of the circuit simulator being used relative to the complexity of the circuit being modeled. The more complex the subcircuit, the longer it takes to simulate the circuit. A second factor is the value chosen for "xtra_runs" in the input file. This is a relatively small factor since the total number of runs varies linearly with the sum of the "xtra_runs" parameter summed with the "max_order" parameter. Thirdly, both the "max_order" and the "num_freq" parameters in the input file together determine how many points of H values will be calculated. This is an extremely nonlinear relationship and is the biggest factor in determining how many runs must be simulated. Lastly, for a shooting method type of circuit simulator, the actual values of the input frequencies can drastically effect the simulation time. Since shooting methods require that the input be periodic, the smallest period for two almost incomensurable frequencies can be very large. For example, if the input frequencies were chosen to be 2 Hz and 4 Hz, then the input signal is periodic in a time of 0.5 s. But if the input frequencies were chosen to be 2.01 and 4, then the input signal is periodic in a time of 100 s.

Run Number	Input Amplitudes							
	0 Hz		1 MHz		2 MHz		3 MHz	
	Real	Imag.	Real	Imag.	Real	Imag.	Real	Imag.
0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	0
3	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0	1
5	0.5	0	0	0.5	0	0	0	0
6	0.5	0	0	0	0	0.5	0	0
7	0.5	0	0	0	0	0	0	0.5
8a	0	0	0.5697	0.0000	0.0000	0.5697	0	0
8b	0	0	0.3535	0.3536	0.0000	0.5	0	0
9a	0	0	-0.5636	-0.3255	0	0	0	0.6508
9b	0	0	0	-0.8053	-0.4652	0	0	0.4650
9c	0	0	0	-0.3440	-0.1987	0	0	0.7945
10	0	0	0	0	0.0001	0.5250	0	0.5250
11	0.4713	0	0.3333	-0.3333	0	0.4713	0	0
12	0.3942	0	-0.3414	-0.1972	0	0	0	0.3942
13	0.3801	0	0	0	-0.1901	-0.3291	0	0.3801
14a	0	0	0.3373	-0.3742	0.0794	-0.4975	0.0000	0.5038
14b	0	0	0.2421	-0.2686	0.3571	0.0570	0.0000	0.3616

Table 4.7: Final list of runs for third order and f =0, 1MHz, 2MHz, 3MHz. Peak minimization, additional needed runs, time shift, and amplitude adjustment are all included. A value of 0.0000 indicates a nonzero number whose absolute value is less than 0.0001.

After the simulations are complete, the values of H can finally be calculated. This process is complicated but very similar to the algorithm used to determine the needed additional runs in the initialization phase. Figure 4.7 gives the overall flow of the calculation procedure. The first step deals with multiple runs with the same input frequencies. This step will be discussed later. For each run, all of the FIPD's and their corresponding output frequencies are calculated. This calculation is the same as performed in the initialization procedure. The next step is to take the time–domain output waveform and convert it to frequency–domain phasors using the FFT algorithm. Then each output phasor, Y_q, in a run is decomposed into the order responses $Y_{q,1}$, $Y_{q,2}$, ... $Y_{q,n}$. This procedure is just a polynomial least squares fit and is done with the SVD algorithm [114]. Then for each order response the H values are calculated if possible. The details of this block are shown in figure 4.8. For each FIPD that corresponds to a particular output frequency and order, first the previously calculated values of H are checked to see if the value of H at this FIPD is known. If it is known, then the response due to this FIPD is calculated and subtracted from $Y_{q,i}$. If not, then the unknown FIPD is added to a list. After all of the FIPD's have been checked, the length of the list is checked. If the length is zero, then all of the values of H contributing to the response at this order and at this output frequency have already been calculated. If there is only one FIPD in the list, then the value of H at this point can be found by solving (4.14). If there is more than one FIPD in the list, then there are more than two columns in the corresponding decomposition matrix equation (4.14) and we need information of more than one run. Therefore we save the current value of $Y_{q,i}$ and the set of unknown FIPD's to a linked list. This linked list is used in the box marked "Deal with multiple runs with the same input frequencies" in figure 4.7. Figure 4.9 gives the detailed algorithm flow for this procedure. Remember that the order of runs is important. We first simulated all of the runs that had a single frequency input, then a two frequency input, and so on. The additional run(s) that are added to the basic list during the "Add needed runs" block of figure 4.5 have the same input frequencies as one of the runs in the basic list. These additional run(s) are added adjacent to that run. Thus when a list of FIPD's is added to the linked list in figure 4.8 it is because there is more than one run with the same input frequencies. Thus the matrix in (4.14) has more than two columns. The information needed to formulate and solve (4.14) is contained in the linked list.

4.2.4 Summary

The extraction procedure consists of three major steps, initialization, simulation, and calculation. First the number of simulations and the conditions for these simulations must be determined. This determination is done by first determining all the different possible combinations of the input frequencies taken at most n at a time where n is the assumed order of the system. For each combination of input frequencies, at least n different simulations are required, each with different input amplitudes, keeping constant the relative phases and relative amplitudes of the components of the input frequencies. Furthermore, for systems of order three and higher, additional simulations with different relative phases and different relative amplitudes of the components of the input frequencies may be required. Secondly, the actual

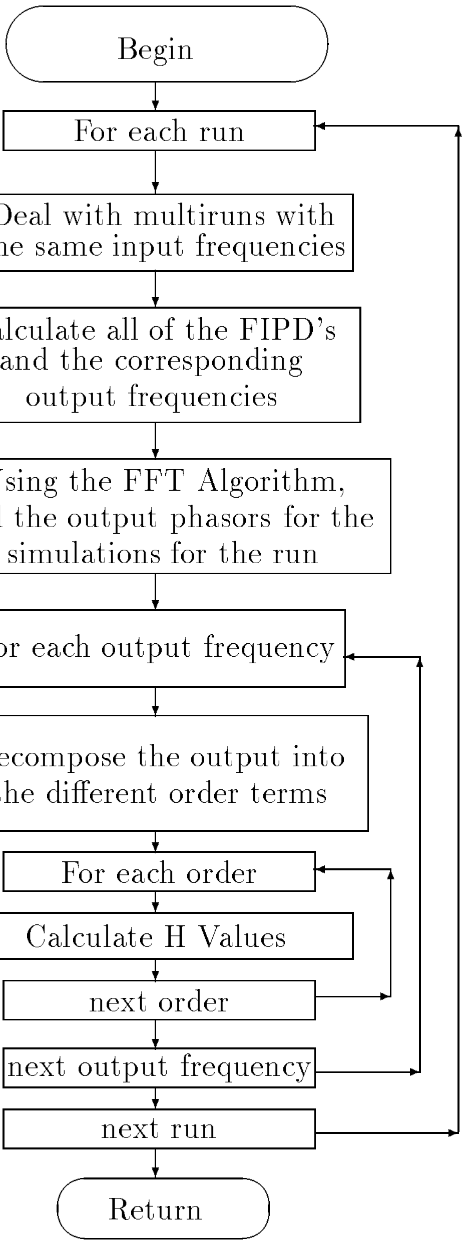
Begin
For each run
Deal with multiruns with the same input frequencies
Calculate all of the FIPD's and the corresponding output frequencies
Using the FFT Algorithm, find the output phasors for the simulations for the run
For each output frequency
Decompose the output into the different order terms
For each order
Calculate H Values
next order
next output frequency
next run
Return

Figure 4.7: High level flow chart for the procedure to go through all of the runs and calculate the transfer function once the simulations are finished.

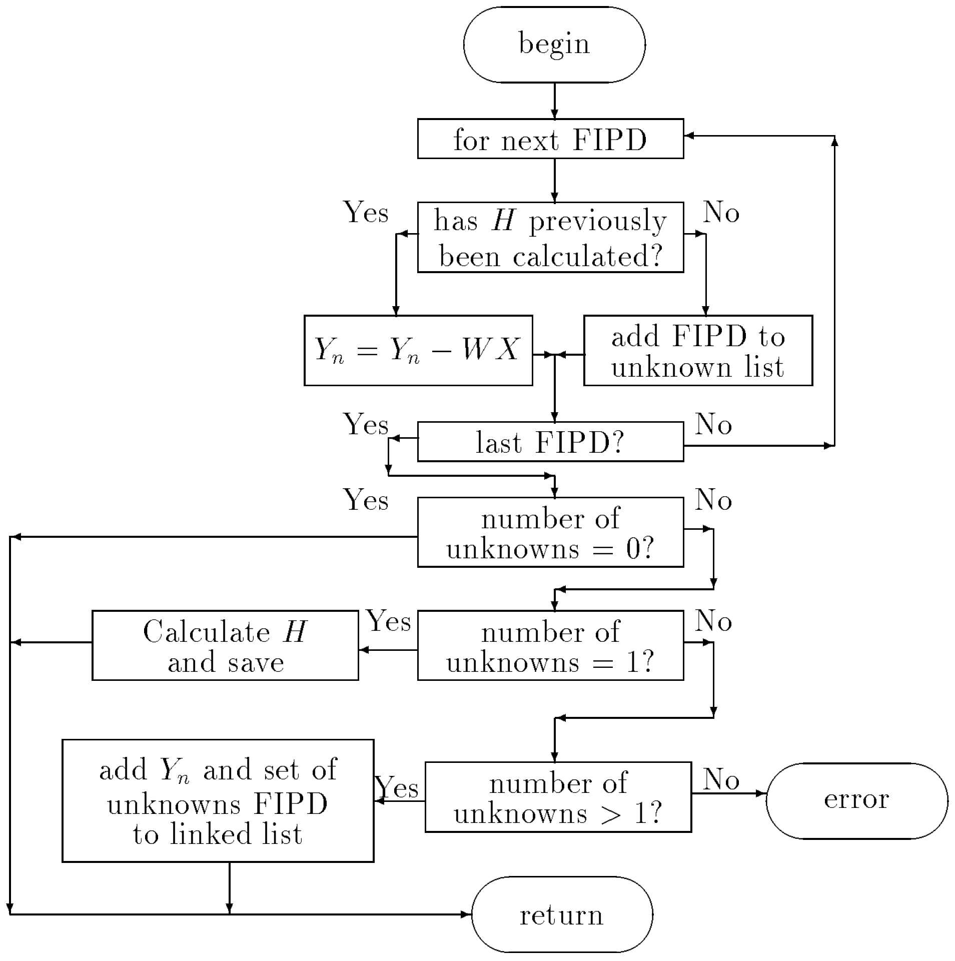

Figure 4.8: Flow chart for the procedure to calculate the nonlinear transfer functions. This represents the block labeled "Calculate H values" in figure 4.7.

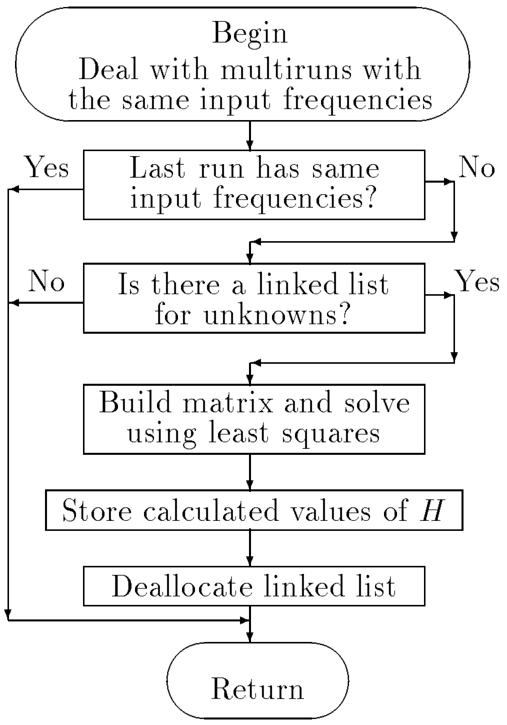
Begin
Deal with multiruns with the same input frequencies
Last run has same input frequencies?
Yes
No
Is there a linked list for unknowns?
No
Yes
Build matrix and solve using least squares
Store calculated values of H
Deallocate linked list
Return

Figure 4.9: Flow chart for the procedure used to deal with multiple runs with the same input frequencies.

simulations are run and the output values saved in files. And lastly, the results from the simulations are used to calculate the Volterra nonlinear transfer functions.

4.3 Conclusion

The extraction procedure described in this chapter provides an automatic way to first determine the needed simulations, perform the needed simulations, and then extract the Volterra nonlinear transfer function values from the simulation results. The compute time needed for the entire process is mainly dependent on the number of input frequencies and the assumed order of the system. The complexity of the system and the efficiency of the circuit simulator being used also effect the amount of required compute resource. Typically, the actual simulation is the part of the procedure that uses the majority of the needed compute resource.

The resulting calculated points of the Volterra nonlinear transfer function are stored in files and can be used to calculate the steady–state response of the system as long as two conditions are satisfied. First, the input amplitude must not be greater than the maximum amplitude specified for the extraction. And secondly, the input spectrum must be comprised soley of discrete frequency values that belong to the list of input frequencies specified for the extraction.

Chapter 5

Verification

5.1 Introduction

This chapter presents examples and measurements of the techniques proposed in chapters 3 and 4. Various examples verify the individual techniques developed in this thesis. A final example provides verification of the integration of the techniques in the analysis of a subsystem using a behavioral model developed in a systematic manner from circuit level simulations. First an example of bivariate generalized power series analysis as proposed in chapter 3 is given. The first example is a ring diode mixer that is modeled as a two input voltage controlled voltage source. Results of the bivariate power series analysis are compared to results from a time domain circuit simulator. Secondly, a MESFET amplifier with a bivariate series characterization of the current source in the equivalent circuit model is examined. By using the relationship between generalized power series and bivariate Volterra series, the circuit is analyzed using signal flow analysis for bivariate Volterra series. Comparison to lab measurements show good results.

Finally, examples of the extraction technique proposed in chapter 4 are given. Circuits with known Volterra nonlinear transfer functions are used to quantify the accuracy of the extraction technique. Lastly, a nonlinear equalization circuit is used as an example of a circuit that does not have a known Volterra series representation. The output predicted by the extracted nonlinear transfer functions is compared to that of the original circuit level simulation.

5.2 Bivariate Generalized Power Series – Ring Diode Mixer

An example of a circuit that can be modeled using bivariate generalized power series analysis is the diode ring mixer shown in Fig. 5.1. In order to calculate a behavioral model for this entire circuit, we need to find an approximation to the output voltage V_{IF} as a function of the two input voltages V_{LO} and V_{RF}. Moreover, this approximation needs to be in the form of (3.3). Thus the entire circuit is modeled as a two-input voltage controlled voltage source.

The current-voltage relationship for the individual diodes are modeled by the Shockley diode equation:

$$I_j = I_{0j}(e^{V_j/V_t} - 1) \tag{5.1}$$

Where I_j is the current through diode D_j, V_j is the voltage across D_j, $V_t = \eta kT/q$, I_{0j} is the saturation current for diode j, T is the junction temperature, k is Boltzmann's constant, and η is the ideality factor.

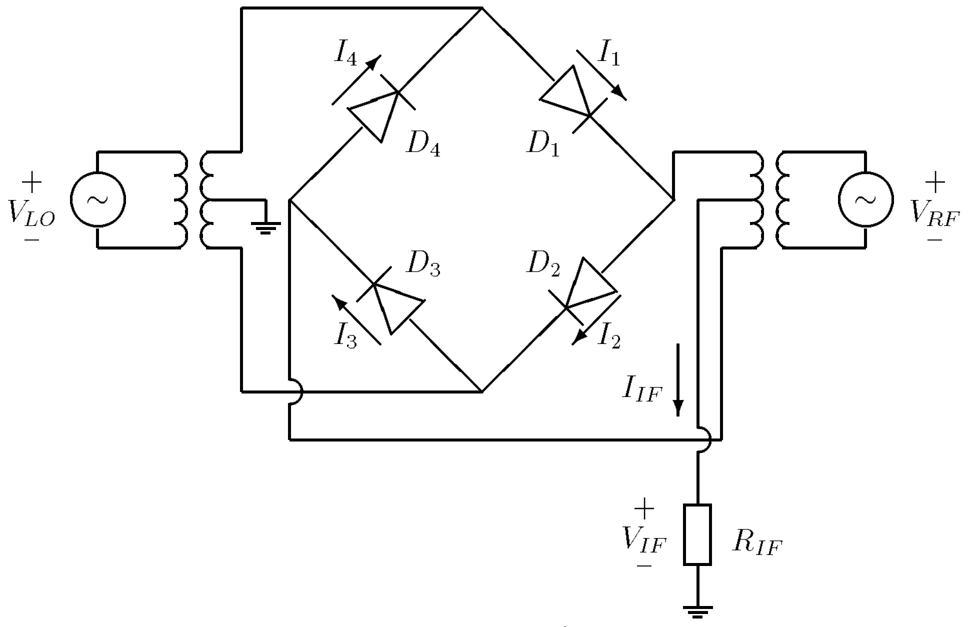

Figure 5.1: Ring Diode Mixer

Assuming that the transformers are ideal, the KCL and KVL equations for this circuit reduce to:

$$V_1(t) = \frac{1}{2}V_{LO}(t) - \frac{1}{2}V_{RF}(t) - V_{IF}(t) \tag{5.2}$$

$$V_2(t) = \frac{1}{2}V_{LO}(t) + \frac{1}{2}V_{RF}(t) + V_{IF}(t) \tag{5.3}$$

$$V_3(t) = \frac{1}{2}V_{RF}(t) - \frac{1}{2}V_{LO}(t) - V_{IF}(t) \tag{5.4}$$

$$V_4(t) = V_{IF}(t) - \frac{1}{2}V_{LO}(t) - \frac{1}{2}V_{RF}(t) \tag{5.5}$$

$$I_1(t) - I_2(t) + I_3(t) - I_4(t) - \frac{V_{IF}(t)}{R_{IF}} = 0 \tag{5.6}$$

The explicit notation of time (t) will be dropped for simplification.

Equations (5.2) – (5.6) were solved for V_{IF} by means of the computer-algebra program MAPLE [115] to perform the algebra and calculus operations. The MAPLE code used is included in appendix C. Although all of the operations are straightforward, some of the equations are so large that manual calculations are impractical. As an intermediate step, a function f having the desired solution at $f = 0$ was obtained:

$$\begin{aligned} f &= I_{01}\left(e^{\left(\frac{1}{2}V_{LO} - \frac{1}{2}V_{RF} - V_{IF}\right)/V_t} - 1\right) \\ &\quad -I_{02}\left(e^{\left(\frac{1}{2}V_{LO} + \frac{1}{2}V_{RF} + V_{IF}\right)/V_t} - 1\right) \end{aligned}$$

$$+I_{03}\left(e^{\left(\frac{1}{2}V_{RF} - \frac{1}{2}V_{LO} - V_{IF}\right)/V_t} - 1\right)$$
$$-I_{04}\left(e^{\left(V_{IF} - \frac{1}{2}V_{LO} - \frac{1}{2}V_{RF}\right)/V_t} - 1\right) - \frac{V_{IF}}{R_{IF}} \qquad (5.7)$$

Since we want a closed form solution for V_{IF} we now estimate f by a third degree Taylor series in V_{IF}.

$$f \approx f(V_{IF}=0) + f^{'}(V_{IF}=0)\,V_{IF} + \frac{1}{2}f^{''}(V_{IF}=0)\,V_{IF}^2 + \frac{1}{6}f^{'''}(V_{IF}=0)\,V_{IF}^3 \quad (5.8)$$

This third order polynomial in V_{IF} can be solved in closed form [111]. In general, of the three solutions to this third order equation, two are complex and one is real. Using the real solution, we get a very large equation which we will call g. The full solution of g is too large to be included here, but is given in appendix C for completeness. The function g has the form $V_{IF} = g(V_{RF}, V_{LO}, I_{01}, I_{02}, I_{03}, I_{04}, V_t)$ to which a bivariate power series of the form

$$g(V_{RF}, V_{LO}) = \sum_{\sigma=0}^{9}\sum_{\rho=0}^{9} a_{\sigma,\rho} V_{RF}^{\sigma} V_{LO}^{\rho} \qquad (5.9)$$

can be fitted using least square techniques over a given range. Thus (5.9) has the form of (3.3) with $b_k = 1$, $\tau_{k,\sigma} = 0$, $d_k = 1$, and $\lambda_{k,\rho} = 0$. For our investigation we used the voltage ranges $-0.6 \leq V_{LO} \leq 0.6$, and $-0.2 \leq V_{RF} \leq 0.2$. and the RMS fit error was less than 1 %. For the ring mixer of Fig. 1, $\eta = 1$, $I_{01} = 1.00$ pA, $I_{02} = 1.01$ pA, $I_{03} = 1.03$ pA, $I_{04} = 1.06$ pA, $V_t = 0.02569$ Volts, and the transformers are ideal and have 1:1 turns ratios. The solid lines in Fig. 5.2 give the steady-state conversion power gain of P_{IF}/P_{RF} for several LO powers. Since the power series (5.9) is valid only for a given range of the inputs, the macromodel is only valid within that range. The circles are simulation results using the time domain simulator ASTAP [113], and show good agreement.

The dashed lines in Fig. 5.2 demonstrate an important property of bivariate GPSA. Since the summation given by (3.114) contains powers of the phasor components as given by (3.16), any arbitrary input-output transfer characteristic is of the form

$$Y_q = \sum_i H_{i,j}(X_j) \qquad (5.10)$$

Where $H_{i,j}$ is an ith order nonlinear transfer function for the input phasor X_j (or set of X_j's). In general, $H_{i,j}$ is a function of the other input phasors. The dashed lines in Fig. 5.2 give (a) first order, H_1, and (b) third order, H_3, transfer characteristics for the IF phasor when $P_{LO} = 2.0$ dBm. In our example, the even-order transfer functions are zero. Note that in Fig. 5.2, the vertical axis is gain, not output amplitude; thus the lines represent the 0 th order and 2 nd order transfer characteristics for the gain, corresponding to the 1st and 3rd order transfer characteristics for output amplitude. For the range modeled, all higher order transfer functions are negligible. Adding (a) and (b) yields a value within 1 percent of the total characteristic (solid line) shown for $P_{LO} = 2$ dBm. Thus a simple behavioral model is obtained by using only lower order powers of the input. Here, the nonlinear RF to IF characteristics can be described by the sum of two components. Each component can be represented as a linear function when expressed in log-log form.

cut and paste same figure as mtt paper June 1990

Figure 5.2: Conversion Gain for different LO power levels. Solid lines are bivariate GPSA calculations. Circles are ASTAP (time domain simulation) calculations. Dashed lines are the first two terms of the decomposition of the solid line for $P_{LO} = 2.0$ dBm. (a) is the constant term: RF^0. (b) is the second order term: RF^2.

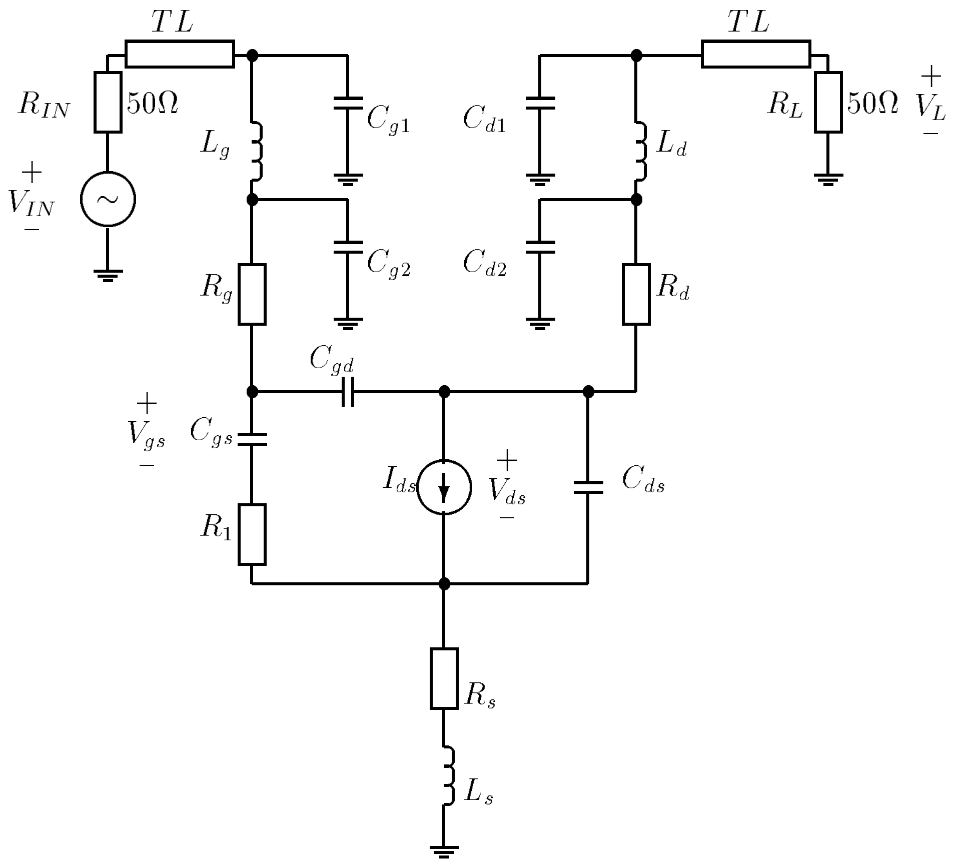

Figure 5.3: Full circuit for measurement of MESFET amplifier characteristics

5.3 Signal Flow Analysis for Bivariate Volterra Series

One form of Volterra series analysis of nonlinear circuits casts the nonlinear circuit into block diagram form, each block being described by linear transfer functions for linear subcircuits and by Volterra nonlinear transfer functions for nonlinear subcircuits [116]. Here we apply this technique to the Volterra series–based analysis of a MESFET amplifier and compare the simulated results to experimental results.

The MESFET amplifier used here was previously analyzed by Chang *et al.* [20]. The model schematic is shown in Fig. 5.3, and the measured parameter values are given in table 5.1. Using the substitution theorem [79, Section 2.2.1] the equivalent signal flow graph of the circuit is shown in Fig. 5.4. The input is the value of the source voltage V_{IN}. This signal passes through the linear systems G_{IN-GS} and G_{IN-DS}. Each of these systems is characterized by all of the linear components of the circuit. V_{GS} and V_{DS} are the inputs to the two-input nonlinear block B characterized by τ and table 5.2. Using (3.106) we can directly calculate the bivariate Volterra

Element	Value
C_{g1}	0.1386 pF
L_g	0.69414 nH
C_{g2}	0.30707 pF
R_g	2.9Ω
R_s	2.4Ω
L_s	0.00323 nH
R_d	5.3Ω
L_d	0.41143 nH
C_{d2}	0.09012 pF
C_{d1}	0.00341 pF
R_1	10Ω
τ	6.56 ps
C_{ds}	0.25050 pF
C_{gs}	0.50150 pF
C_{gd}	0.08637 pF

Table 5.1: Element values for MESFET circuit

Order of	Order of V_{ds} Term			
V_{gs} Term	0	1	2	3
0	0.0211092	0.00467136	0.00004985	0.00026358
1	0.0689287	0.00307239	−0.000616581	0.00244648
2	0.0552606	−0.0158083	0.00336844	0.00931433
3	−0.0241834	−0.0132546	0.00541917	0.0127004
4	−0.0328935	0.0236952	−0.00941775	−0.0136877
5	0.00865757	0.0340615	−0.0131895	−0.0366089
6	0.00689521	−0.00467676	0.0034243	−0.00190339
7	−0.00616748	−0.0241809	0.00856464	0.0253214
8	−0.0029458	−0.00895554	0.00252446	0.0105268

Order of	Order of V_{ds} Term			
V_{gs} Term	4	5	6	7
0	0.00006759	−0.000138752	−0.000008659	0.0000154853
1	−0.000132799	−0.000942032	0.0000306236	0.000105969
2	−0.00214949	−0.00326047	0.000315678	0.000336661
3	−0.00146277	−0.00362775	0.000207059	0.000290322
4	0.00546728	0.00528516	−0.000791546	−0.000575213
5	0.00563955	0.0110368	−0.000849452	−0.000954538
6	−0.00231212	−0.000264063	0.000324688	0.00010378
7	−0.00426292	−0.00775008	0.000655445	0.000683864
8	−0.001237	−0.00305257	0.00019882	0.000254013

Table 5.2: Coefficients for bivariate power series of I_{ds} about DC operating point

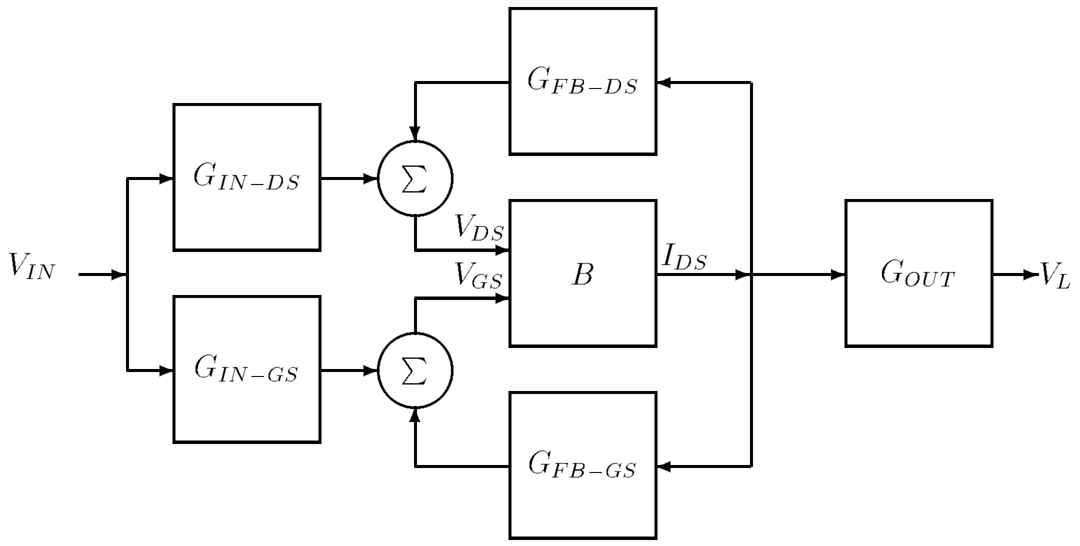

Figure 5.4: Nonlinear signal flow graph for MESFET amplifier. G_{IN-GS}, G_{IN-DS}, G_{FB-GS}, G_{FG-DS}, and G_{OUT} are linear systems. B is a bivariate nonlinear system.

nonlinear transfer function and thus can calculate the value I_{DS}. The value of the output voltage V_L is the output of the linear system G_{OUT} which has as its input I_{DS}. The two linear systems G_{FB-DS} and G_{FB-GS} provide the feedback to the two inputs of the bivariate block.

The aim now is to solve for the steady–state output of the system given a steady–state input signal V_{IN}. The two feedback paths do not allow a straightfoward closed form solution. However, the steady–state output of the system can be solved relatively easily by iteration for low input powers. For the first iteration, we assume that the feedback contributions are negligible and calculate the system response. Then using the new calculated value of I_{DS} we calculate a new V_{DS} and V_{GS} and use the new values for the next iteration. This process is continued until there is no appreciable change of any of the values from one iteration to the next.

Figure 5.5 gives the results of a two–tone test for the circuit using two equal-amplitude signals at input frequencies of 2.35 GHz and 2.4 GHz. The horizontal axis is the input power for one of the tones, and the vertical axis is the output power at the fundamental frequency of 2.35 GHz and the image frequency of 2.3 GHz. The points are experimentally measured values and the solid lines are the simulated values and show good agreement. These simulated values also agree closely with those given by Chang *et al.* [20]. Note that a linear simulation would predict no power at 2.3 GHz.

Another technique for solving nonlinear circuits, that is based on the work of Volterra, is the method of nonlinear currents [116], [79]. This technique has the advantage of a closed form solution to a nonlinear system which consists of (a) linear subsystems and (b) simple nonlinear systems described by univariate power series. Strictly speaking, the generalized form of the power series (which includes the time delays τ) cannot be used with this analysis. However, a pure delay can be lumped into the linear systems, and thus a generalized power series can be adapted to this analysis. This method has been used to analyze MESFET amplifiers [81], but since the nonlinear blocks are characterized by univariate power series, the V_{DS}–V_{GS} cross product terms of the bivariate power series are ignored. Thus the coefficients in table 5.2 that are neither in the first row nor the first column are assumed to be negligible. The work presented here can be used to extend the method of nonlinear currents to include more general nonlinearities.

5.4 New Volterra Extraction

5.4.1 Linear RC Filter

The single–pole low pass filter circuit shown in figure 5.6 was used to test the extraction procedure described in chapter 5. The circuit was simulated using the ASTAP circuit simulator [113]. This circuit is linear and thus the nonlinear transfer function is equal to zero for orders higher than 1. The input file used was that shown in figure 4.3. Note that we assumed an order of 3 and thus we expect to extract values equal to zero for H_2 and H_3. Figures 5.7 and 5.8 show the tables that were extracted using the procedure. These tables show the general form of the output files. This first line contains two numbers, the number of inputs and the number of outputs. Since the functions H_n are complex, the number of outputs is always two, one for

Cute and paste Same fig as IJMMWCAE article, July 1991

Figure 5.5: The results of a two tone test for the circuit using two equal-amplitude signals at input frequencies of 2.35 GHz and 2.4 GHz. The horizontal axis is the input power for one of the tones, and the vertical axis is the output power at the fundamental frequency of 2.35 GHz and the image frequency of 2.3 GHz. The solid lines are the simulated values and the points are experimentally measured values.

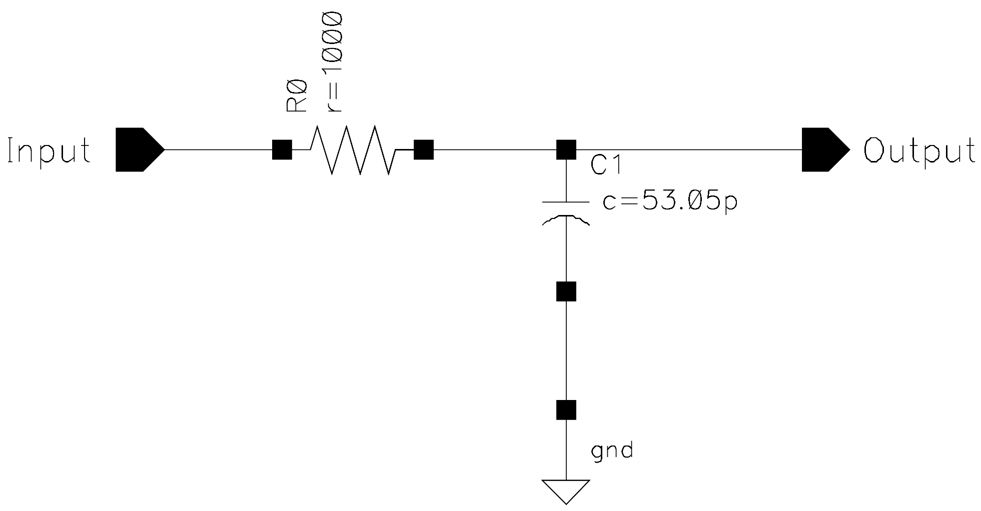

Figure 5.6: Schematic of simple RC filter used to test extraction technique.

number_of_inputs	number_of_output				
format:	type_of_format				
f_1	f_2	$\ldots$	f_n	$\mathrm{Re}\{H_n\}$	$\mathrm{Im}\{H_n\}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 5.3: General output format for nth order H's

the real part and one for the imaginary part. The number of inputs corresponds to the order of the nonlinear transfer function. The general form of the tables is shown in table 5.3.

In figure 5.7 the value of H_0 is equal to 0 as expected. The known linear characteristics are shown by the values of H_1 and show the expected roll off of a single pole at 3MHz. The point for H_2 in figure 5.7 and the points for H_3 in figure 5.8 are close to zero as expected. These values are not exactly equal to zero due to computational noise present in the simulation results.

5.4.2 Power Series Device, 7th order

Now that a linear circuit with memory has been used to test the extraction algorithm, we move onto the case of a nonlinear system with *apriori* known values of H. In this example we will use a memoryless sytem that can be described by the equation

$$y = x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x^1 + 1 \tag{5.11}$$

```
--------------------------------------------------------------------
0 2
Format: ascii
         +0          +0
--------------------------------------------------------------------
1 2
Format: ascii
      +0   +0.999817          +0
  +1e+06   +0.899975   +0.300012
  +2e+06   +0.692269   +0.461544
  +3e+06   +0.49996    +0.499993
--------------------------------------------------------------------
2 2
Format: ascii
  -3e+06   +3e+06 -1.01562e-07          +0
  -2e+06   +2e+06 -3.78306e-08          +0
  -2e+06   +3e+06 +6.53864e-07 +1.35933e-06
  -1e+06   +1e+06 -1.91459e-09          +0
  -1e+06   +2e+06 -2.07609e-06 +6.26003e-07
  -1e+06   +3e+06 -1.13393e-07 -2.99255e-07
      +0       +0 +2.78369e-08          +0
      +0   +1e+06 +9.48788e-07  -9.2115e-07
      +0   +2e+06 +1.64792e-07 +5.84656e-08
      +0   +3e+06 -8.97048e-07 -5.75013e-09
  +1e+06   +1e+06 -1.54333e-08 +3.31582e-08
  +1e+06   +2e+06 +5.14803e-07 -7.56933e-07
  +1e+06   +3e+06 -1.14537e-07  +3.2733e-07
  +2e+06   +2e+06 +5.75729e-09   +5.225e-08
  +2e+06   +3e+06 +3.02006e-07   +1.144e-06
  +3e+06   +3e+06 -2.74028e-08 +1.12958e-07
```

Figure 5.7: The results for H_0, H_1, and H_2 of a low–pass RC filter. Pole is at 3MHz. The last two columns are the real and imaginary part of the extracted transfer function. The beginning columns are the arguments in hertz to the transfer function.

```
3 2
Format: ascii
  -3e+06       +0   +3e+06 -0.000170968          +0
  -3e+06   +1e+06   +3e+06 +2.95559e-06 +5.03615e-06
  -3e+06   +2e+06   +2e+06 +1.60853e-05 -2.08472e-05
  -3e+06   +2e+06   +3e+06 +2.08938e-06 +3.12744e-06
  -3e+06   +3e+06   +3e+06 +1.19571e-06 +1.75821e-07
  -2e+06   -1e+06   +3e+06           -0  +1.3751e-05
  -2e+06       +0   +2e+06 -0.000168853           +0
  -2e+06       +0   +3e+06 -1.95233e-05 +9.20555e-06
  -2e+06   +1e+06   +2e+06 -0.00010614   -3.1055e-05
  -2e+06   +1e+06   +3e+06 +0.000408738   +0.0002841
  -2e+06   +2e+06   +2e+06 +1.31617e-06 -3.43648e-07
  -2e+06   +2e+06   +3e+06  +9.4762e-06 +4.53258e-07
  -2e+06   +3e+06   +3e+06 +2.57414e-05 +2.58141e-05
  -1e+06   -1e+06   +2e+06           -0 +7.18017e-06
  -1e+06   -1e+06   +3e+06 -5.91117e-06 -1.00723e-05
  -1e+06       +0   +1e+06 -0.000169207           +0
  -1e+06       +0   +2e+06 -0.00019892  +0.000616142
  -1e+06       +0   +3e+06 -3.37367e-05  +2.5736e-05
  -1e+06   +1e+06   +1e+06 +4.16386e-07 -3.29523e-07
  -1e+06   +1e+06   +2e+06 -7.65668e-05 -5.72107e-05
  -1e+06   +1e+06   +3e+06 +1.01919e-05 +1.59415e-06
  -1e+06   +2e+06   +2e+06 +7.87875e-06 +6.11655e-06
  -1e+06   +2e+06   +3e+06 +1.23486e-05 +4.00424e-06
  -1e+06   +3e+06   +3e+06 +5.42847e-06 +9.70645e-06
      +0       +0       +0 +1.23999e-05          +0
      +0       +0   +1e+06 -0.000106029 -3.70775e-05
      +0       +0   +2e+06 -8.27047e-05 -5.50806e-05
      +0       +0   +3e+06 -5.96541e-05 -5.90868e-05
      +0   +1e+06   +1e+06 +9.42825e-08 +2.04803e-06
      +0   +1e+06   +2e+06 -7.71757e-06 -1.48022e-05
      +0   +1e+06   +3e+06 -1.73463e-05   -3.431e-05
      +0   +2e+06   +2e+06 -2.02255e-06 +1.45229e-06
      +0   +2e+06   +3e+06 -2.10179e-05 -1.00984e-06
      +0   +3e+06   +3e+06 -4.61831e-06 +9.56671e-07
  +1e+06   +1e+06   +1e+06 +2.79563e-06 +1.35041e-06
```

Figure 5.8: Sample of results for H_3 of low–pass RC filter. Pole is at 3MHz. The first three columns represent the argument to H_3. The last two columns are the real and imaginary part of the value of the function.

Transfer Function indicating order	Number of Calculated points
H_0	1
H_1	4
H_2	16
H_3	32
H_4	60
H_5	87
H_6	131
H_7	179
Total	510

Table 5.4: Number of points calculated for each H for 7th order system

From (3.69) we know that the nonlinear transfer function for a memoryless system described by a polynomial is just the constant coefficient of that order. Thus for this system we know that

$$H_0 = 1 \tag{5.12}$$
$$H_1(f_a) = 1 \tag{5.13}$$
$$H_2(f_a, f_b) = 1 \tag{5.14}$$
$$H_3(f_a, f_b, f_c) = 1 \tag{5.15}$$
$$H_4(f_a, f_b, f_c, f_e) = 1 \tag{5.16}$$
$$H_5(f_a, f_b, f_c, f_e, f_f) = 1 \tag{5.17}$$
$$H_6(f_a, f_b, f_c, f_e, f_f, f_g) = 1 \tag{5.18}$$
$$H_7(f_a, f_b, f_c, f_e, f_f, f_g, f_h) = 1 \tag{5.19}$$

This polynomial characteristic was simulated using ASTAP with a voltage controlled voltage source with the specified polynomial characteristics. The number of transfer function points extracted for each order is indicated in table 5.4. Figure 5.9 is a scattergram of the extracted values of the transfer functions as functions of transfer function order.

5.5 Four Megabit Token Ring Equalizer Circuit

In order to test the extraction procedure, an existing circuit was chosen that exhibits linear behavior for small inputs, and nonlinear behavior for large inputs. This circuit will be used to test the theory and practicality of the development presented in this thesis. The circuit shown in figure 5.10 is an active equalizer circuit for the IBM 4M bit token ring local area network. Terminals IN_HI and IN_LO are the high and low inputs, and the output is the voltage across OUT_HI and OUT_LO. For long cable lengths, and thus low power input signals, both input transistors remain in the active region and the small signal, or linear model of the circuit accurately predicts

Cut and paste. from file 7ordho.ps

Figure 5.9: Extracted value of the nonlinear transfer function for a memoryless 7th order system with known characteristics. All values should be equal to (1,0). Different symbols represent the order of the nonlinear transfer function.

the performance of the circuit. But for short cable lengths, and thus large input signals, the input signal is strong enough to alternately turn off both transistors and the output signal is clipped. This behavior is shown in figures 5.11 and 5.12 for various input signal levels and at two different frequencies. Note also the frequency–dependent nonlinear affects of the zero crossing shift. The phase of the input signal is the same for all signals.

Since we will use the extraction algorithm described in section 4.2 to develop the transfer functions used in simulating a steady–state system, we must know the frequencies which will be used. Furthermore, we must be able to make the tradeoff between accuracy and speed of both the simulation and the extraction procedure. Let us examine a lower bound for the number of simulations needed for our extraction routine. If F is the number of input frequencies and N is the assumed order of the system, then the number of simulations needed, S, is

$$S \geq 1 + N \sum_{k=1}^{\min(N,F)} \binom{\mathrm{F}}{\mathrm{k}} \tag{5.20}$$

Here the 1 represents the single simulation that is needed to determine H_0. The value $\sum_{k=1}^{\min(N,K)} \binom{\mathrm{F}}{\mathrm{k}}$ is the sum of all of the possible combinations of the input frequencies starting with a single input frequency up to the maximum number of input frequencies. This maximum is the lesser of either the assumed order of the system, or the number of input frequencies. Since each run represents at least N simulations, this sum is multiplied by N. Table 5.5 shows the relationship between this lower bound for the number of required simulations as a function of the number of frequencies, and the assumed order of the system. This table represents a lower bound in the sense that it will be exactly the minimum number of simulations needed if and only if there are no unique FIPD's with the same order and frequency as discussed in section 4.2.2. Additional runs must be added either because of (a) unbalanced DC FIPD's, (b) FIPD's with unbalanced DC FIPD subsets or (c) other unique FIPD's with the same order and frequencies. The actual number of runs used can be greater than the minimum required as discussed here. This is denoted by the value of xtra_runs as described in figure 4.3 being greater than zero. The term xtra_runs is somewhat misleading because it really refers to the number of extra simulations that are done for each run. For a nth order system, each run must have at least n different simulations at different input amplitudes. The equivalent of a polynomial fit is performed on the varying output amplitudes in order to decompose the output into the 1st, 2nd, etc. order response. Increasing xtra_runs only allows for extra points on the polynomial fit for each output phasor. Addional runs added because (a), (b) and (c) listed above are required to separate out different components of an output frequency that are caused by the same set of input frequencies.

Table 5.6 shows the same relationship of table 5.5 for the special case of all input frequencies being the lowest possible harmonics of a fundamental. So, for instance, for 3 frequencies, the frequencies used would be the first harmonic (the fundamental), the second harmonic, and the third harmonic. The value of the fundamental does

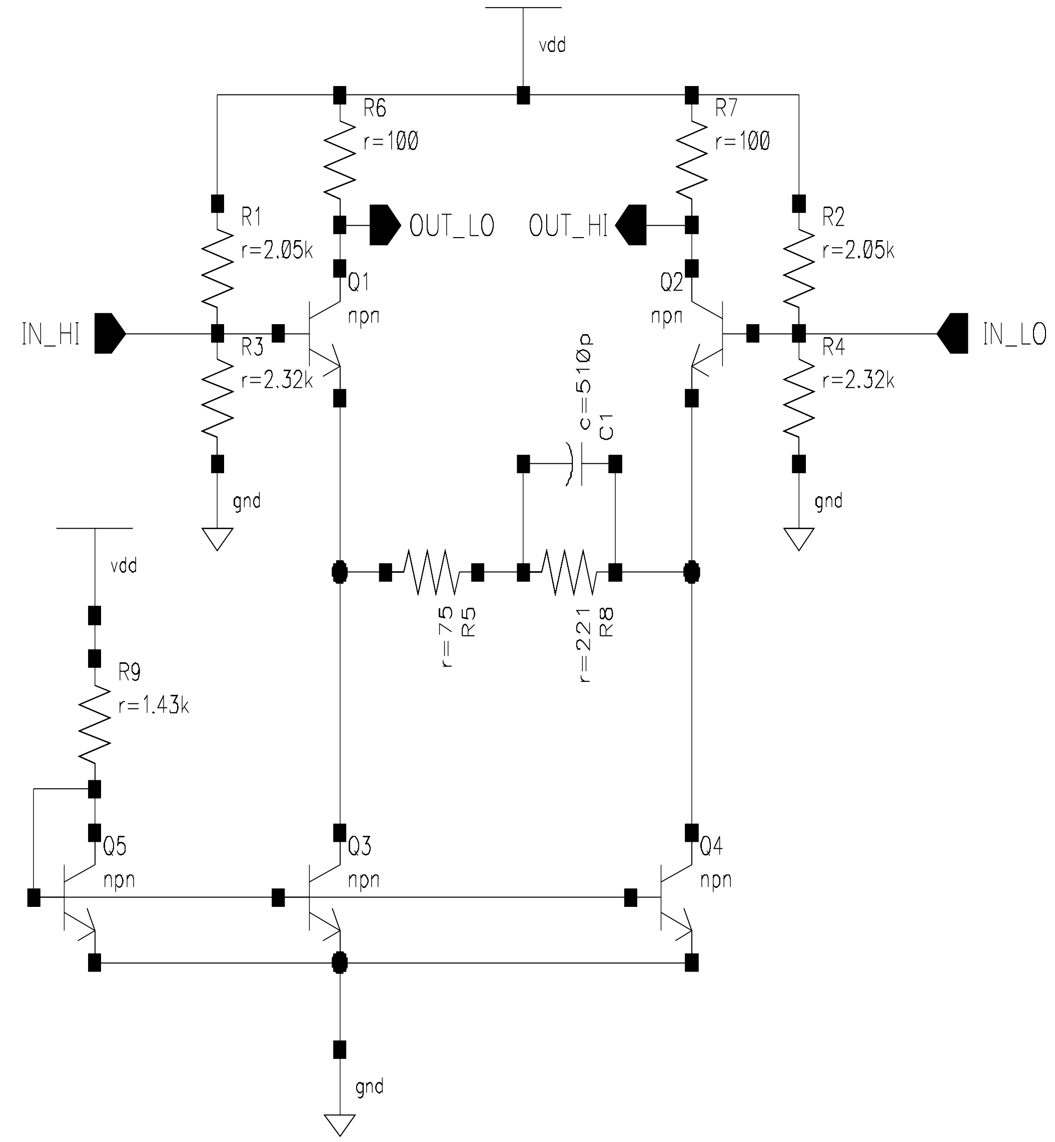

Figure 5.10: Schematic of equalizer circuit

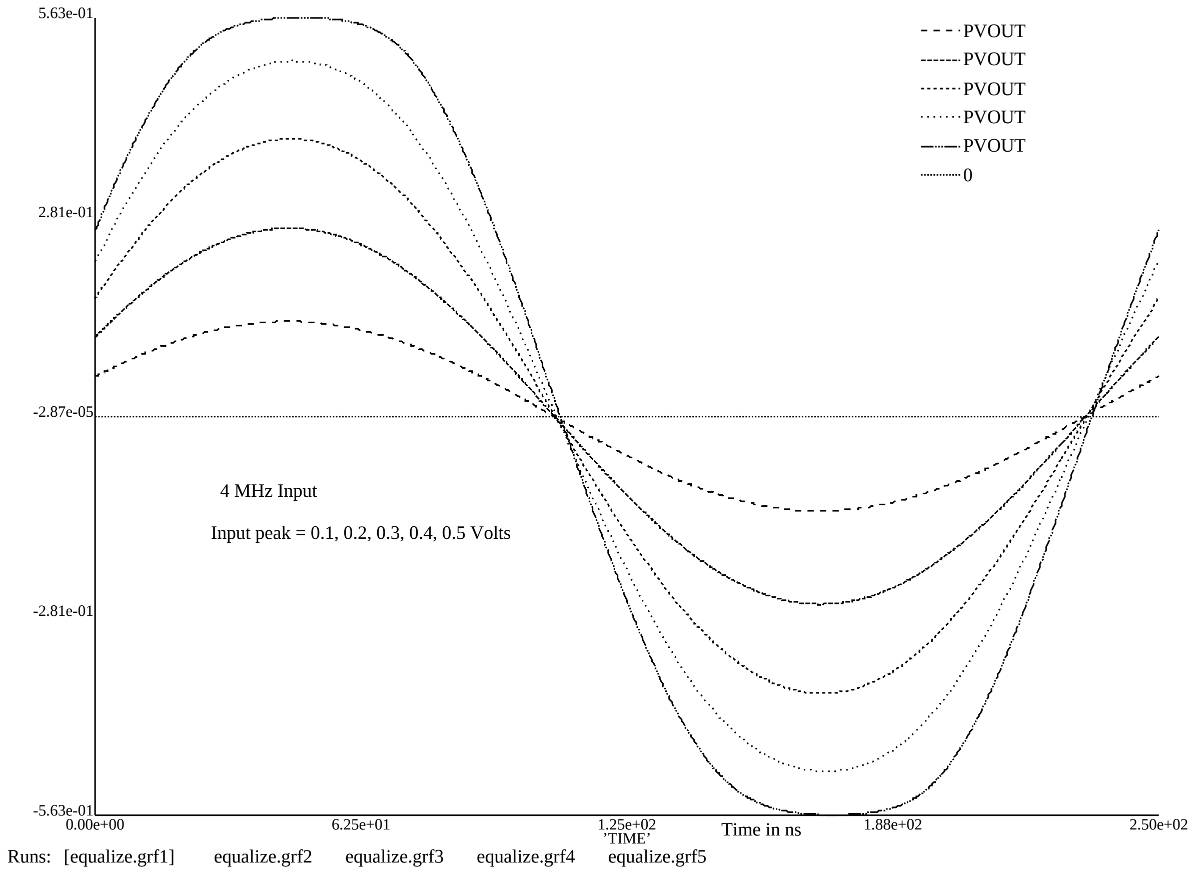

Figure 5.11: Output of the equalizer circuit shown in figure 5.10 with a single sinusoidal 4 MHz input signal of 100mV, 200mV, 300mV, 400 mV and 500mV. Note the nonlinear affect of the clipping and the zero crossing. The phase of the input is the same for all signals.

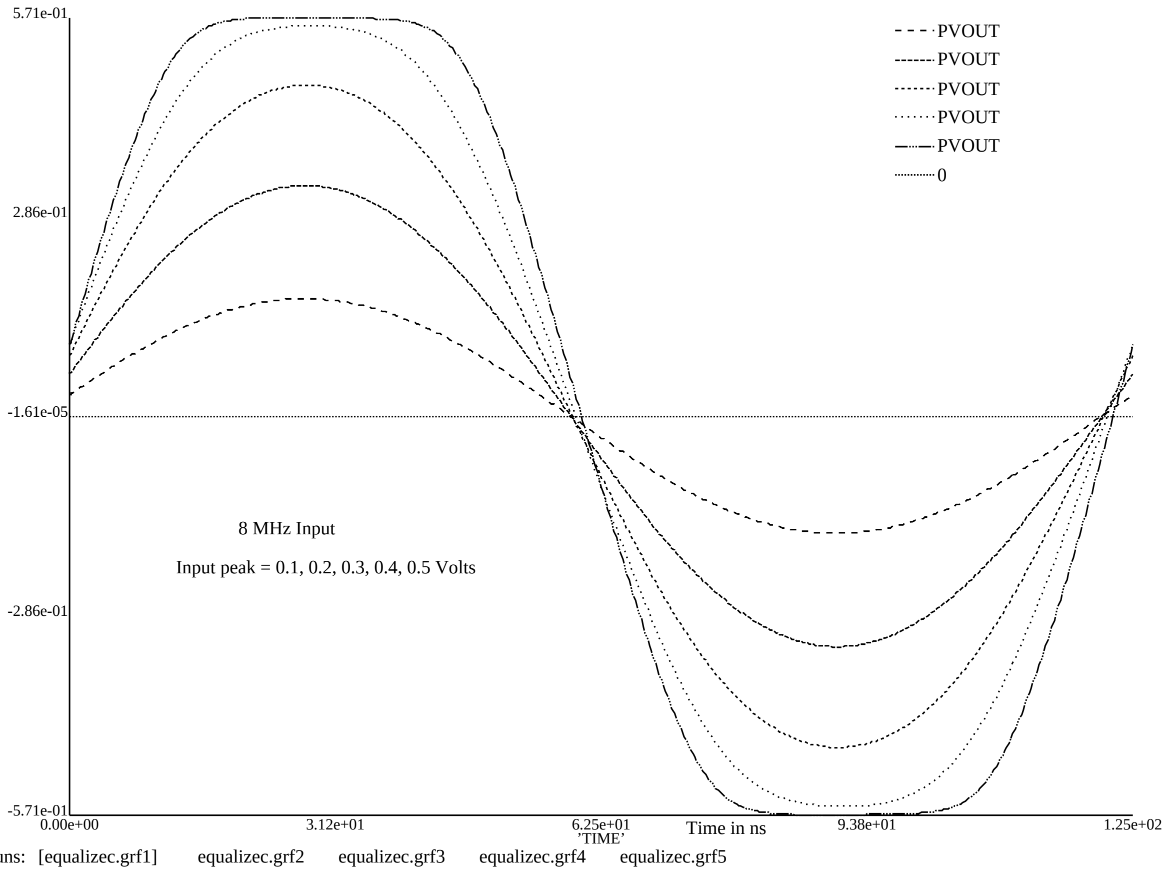

Figure 5.12: Output of the equalizer circuit shown in figure 5.10 with a single sinusoidal 8 MHz input signal of 100mV, 200mV, 300mV, 400 mV and 500mV. Note the nonlinear affect of the clipping and the zero crossing. The phase of the input is the same for all signals.

Number of	Assumed Order of the System							
Frequencies	1	2	3	4	5	6	7	8
1	2	3	4	5	6	7	8	9
2	3	7	10	13	16	19	22	25
3	4	13	22	29	36	43	50	57
4	5	21	43	61	76	91	106	121
5	6	31	76	121	156	187	218	249
6	7	43	124	225	311	379	442	505
7	8	57	190	393	596	757	890	1017
8	9	73	277	649	1091	1477	1779	2041

Table 5.5: The lower bound of the miminum number of simulations needed to extract the nonlinear transfer functions as a function of the number of frequencies and the order of the system. No assumption is made on the relationship between the input frequencies. This lower bound will be exactly the same as the minimum if there are no additional runs required because of multiple unique FIPD's with the same order and frequencies.

not affect this table as long as DC (zero hertz) is not considered an input frequency. Note the striking difference between table 5.6 and table 5.5. For example, the lower bound for the number of simulations needed for a third order system with four input frequencies is 43. By looking at the similar input shown in table 4.3 one can derive this number of simulations. Run 0 in table 4.3 represents a single simulation, and runs 1 through 14 represent 3 (because the system is 3rd order) runs each. Thus a total of $1 + 3 \times 14 = 43$ simulations are needed. Note that tables 4.3 through 4.7 correspond to a system of 4 input frequencies that include DC (zero hertz). This is slightly different from table 5.5 which represents input frequencies that are harmonics and do not include DC. When DC is an input frequency, fewer additional runs need to be added because DC mixed with any other frequency can produce only that other frequency, whereas a non-DC frequency, say f_1 mixed with another frequency, say f_2 produces two frequencies, $f_1 + f_2$ and $f_1 - f_2$. Thus the number of simulations represented in tables 4.5 through 4.7 $(1 + 3 \times (14 + 4) = 55)$ is slightly different from the number of simulations shown for third order and 4 input frequencies in table 5.5 (61). The additional simulations over the lower bound given in table 5.6 are due to: (a) unbalanced DC FIPD's, (b) FIPD's with unbalanced DC FIPD subsets or (c) other unique FIPD's with the same order and frequencies. As the assumed order of the system increases, and as the number of input frequencies increases, the number of additional runs needed increases dramatically.

Since the number of additional runs is related to the number of unique FIPD's that add to the same number, it can be conjectured that table 5.5 represents the worst case condition, or the upper bound, for the minimum number of simulations needed. Of course the number of actual simulations can be made arbitrarily high by chosing a large value for xtra_runs, as described in figure 4.3.

The circuit shown in figure 5.10 was extracted assuming a 5th order system and

Number of Frequencies	Assumed Order of the System							
	1	2	3	4	5	6	7	8
1	2	3	4	5	6	7	8	9
2	3	7	13	81	101	175	302	665
3	4	13	34	165	355	625	1261	3169
4	5	21	61	393	891	2131	4383	10209
5	6	31	100	825	2546	5587	12776	–
6	7	43	163	1445	5436	13273	30864	–
7	8	57	238	2345	11241	30787	–	–
8	9	73	337	3529	20556	65131	–	–

Table 5.6: The minimum number of simulations needed to extract the nonlinear transfer functions as a function of the number of frequencies and the order of the system. The frequencies consist of a fundamental and the lowest harmonics. A value of "-" indicates that the run time to calculate the number of simulations was over 24 hours on an IBM RS/6000 model 530.

a maximum input amplitude of 0.5 Volts. The number of extra simulations per run, xtra_runs, as described in figure 4.3 was set to 1. The input frequencies to be extracted were 2MHz, 4MHz, 6MHz, 8MHz, and 10MHz. A total of 509 simulation runs, each consisting of 6 simulations, resulted in a total of 3054 individual simulations. The total number of points extracted for each transfer function is shown in table 5.7. Figures 5.13 and 5.14 show the response of the equalizer circuit at 4MHz and 8MHz respectively when the input amplitude is 0.5 volts. In addition to the simulated response, the response as predicted by the extracted Volterra model is shown. The Volterra model is truncated to various orders to show the significance of the various order contributions. For the 4MHz sine waves in figure 5.13 the full fifth

Transfer Function	Number of Points
H_0	1
H_1	5
H_2	30
H_3	110
H_4	365
H_5	1001

Table 5.7: Number of points calculated for each nonlinear transfer function. The assumed order of the system equals 5 and the number of frequencies equals 5.

Frequency	Input		Output	
(MHz)	Real	Imag.	Real	Imag.
0	0	0	-0.0989567	0
2	-0.225079	0	-0.183574	0.0982065
4	0.15915494	0	0.189815	-0.0831388
6	-0.07502636	0	-0.110332	0.0315885
8	0	0	0.000159172	0.00230049
10	0.04501582	0	0.079895	-0.010912
12	0	0	0.00254697	0.00186382
14	0	0	-0.000943781	-0.00189096
16	0	0	0.000206277	-0.000143682
18	0	0	0.000110684	0.00111191
20	0	0	0.000420457	-0.000757654
22	0	0	0.000282121	8.05754e-05
24	0	0	-6.52212e-05	-0.000271398
26	0	0	-9.02686e-05	0.000133352
28	0	0	-3.43362e-05	-7.49608e-05
30	0	0	-8.13724e-06	-2.21758e-05
32	0	0	-3.3284e-05	5.19877e-05
34	0	0	1.86298e-05	-1.66032e-05
36	0	0	-2.2996e-06	4.43709e-06
38	0	0	-8.14703e-06	2.89457e-06
40	0	0	3.18382e-06	-2.25542e-06
42	0	0	2.58058e-07	-3.26546e-07
44	0	0	-8.20489e-07	4.53726e-07
46	0	0	3.38836e-08	-1.49648e-07
48	0	0	0	0
50	0	0	-2.44766e-08	4.77152e-09

Table 5.8: Input and output phasors as an example of the response predicted by the extracted Volterra model. The input is derived from a pulse train with truncated harmonics.

order Volterra model prediction is indistinguishable from the simulated response. The results in figure 5.14 are similar, but because of the more pronounced clipping, a slight difference can be distinguished between the fifth order Volterra model response and the simulated response. Figures 5.15 and 5.16 show the same curves as figures 5.13 and 5.14, but with an input amplitude of 0.1 volts. With this lower input amplitude, the circuit exhibits a linear response, and thus the various truncations of the Volterra model are indistinguishable, i.e. only the first order response (H_1) is significant. Figure 5.17 shows the various truncated Volterra models for an arbitrary input with non–zero amplitudes for all five valid input frequencies (2MHz, 4MHz, 6MHz, 8MHz, and 20MHz). The output phasors for the full fifth order Volterra model and the input phasors are given in table 5.9. Note that the output components are negligible at high frequencies. As in figures 5.13 and 5.14 the higher order Volterra model yields better results and the full fifth order model is almost indistinguishable from the simulated result. Figure 5.18 shows the effect increasing the input amplitude 10 percent over the maximum valid range for the model. Although the fifth order model closely matches the simulated result for most of the waveform, the area where the circuit clipps the output (and is thus highly nonlinear), the Volterra model differs from the simulated result.

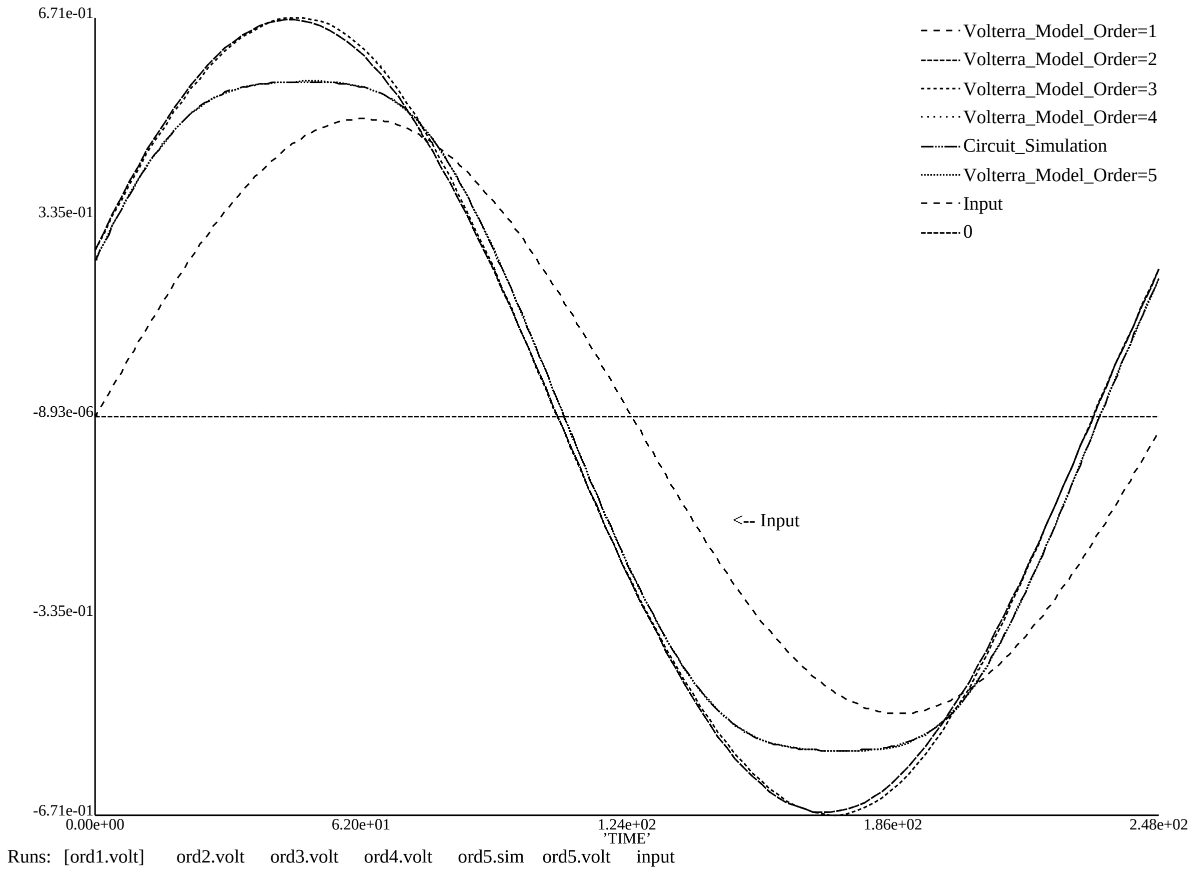

Figure 5.13: Output of the equalizer circuit shown in figure 5.10 with a single sinusoid 4 MHz input signal of 500mV. The Volterra model is truncated at various orders. The even order contributions are negligible, thus model order = 1 is indistinguishable from order=2, and model order = 3 is indistinguishable from order=4.

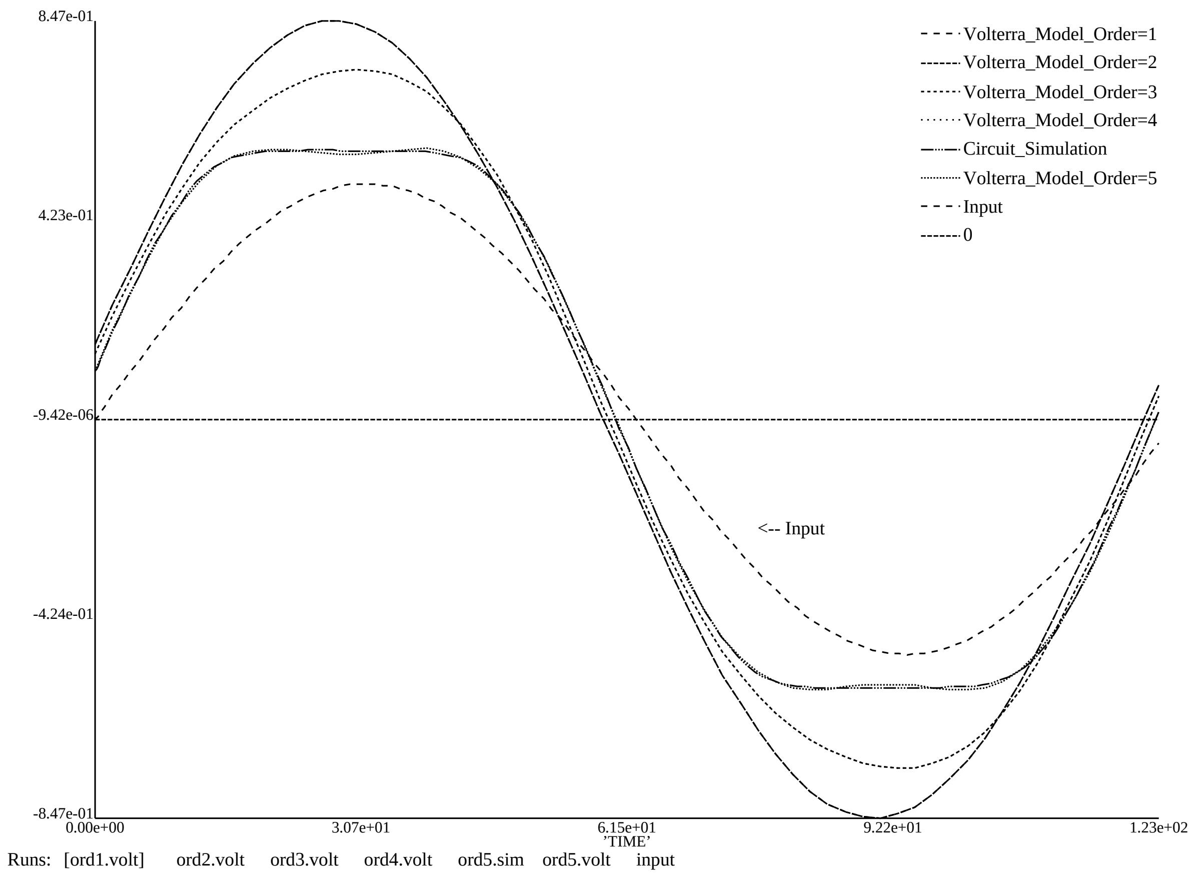

Figure 5.14: Output of the equalizer circuit shown in figure 5.10 with a single sinusoid 8 MHz input signal of 500mV. The Volterra model is truncated at various orders. The even order contributions are negligible, thus model order = 1 is indistinguishable from order=2, and model order = 3 is indistinguishable from order=4.

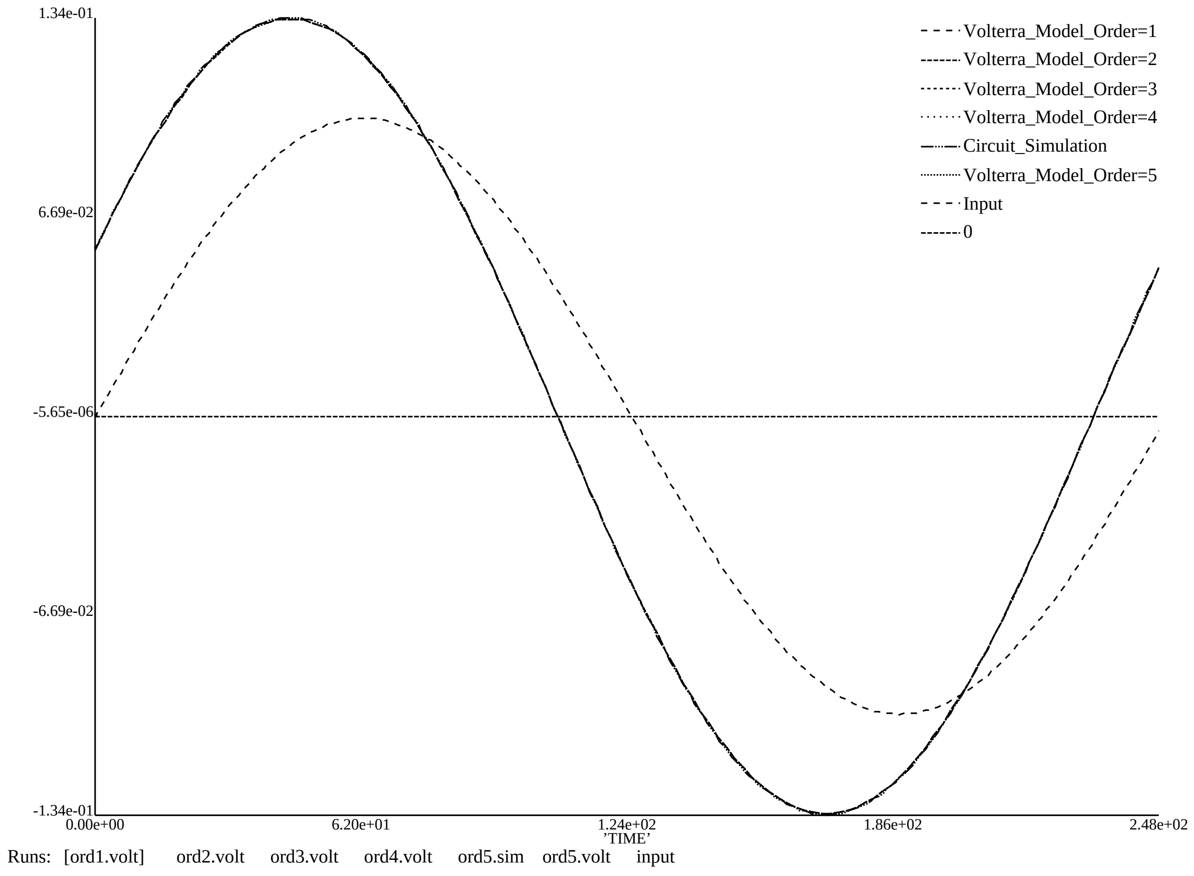

Figure 5.15: Output of the equalizer circuit shown in figure 5.10 with a single sinusoid 4 MHz input signal of 100mV. The Volterra model is truncated at various orders. Since the input signal is relatively low, only the first order contribution is significant, and thus the higher order models are indistinguishable from order=1.

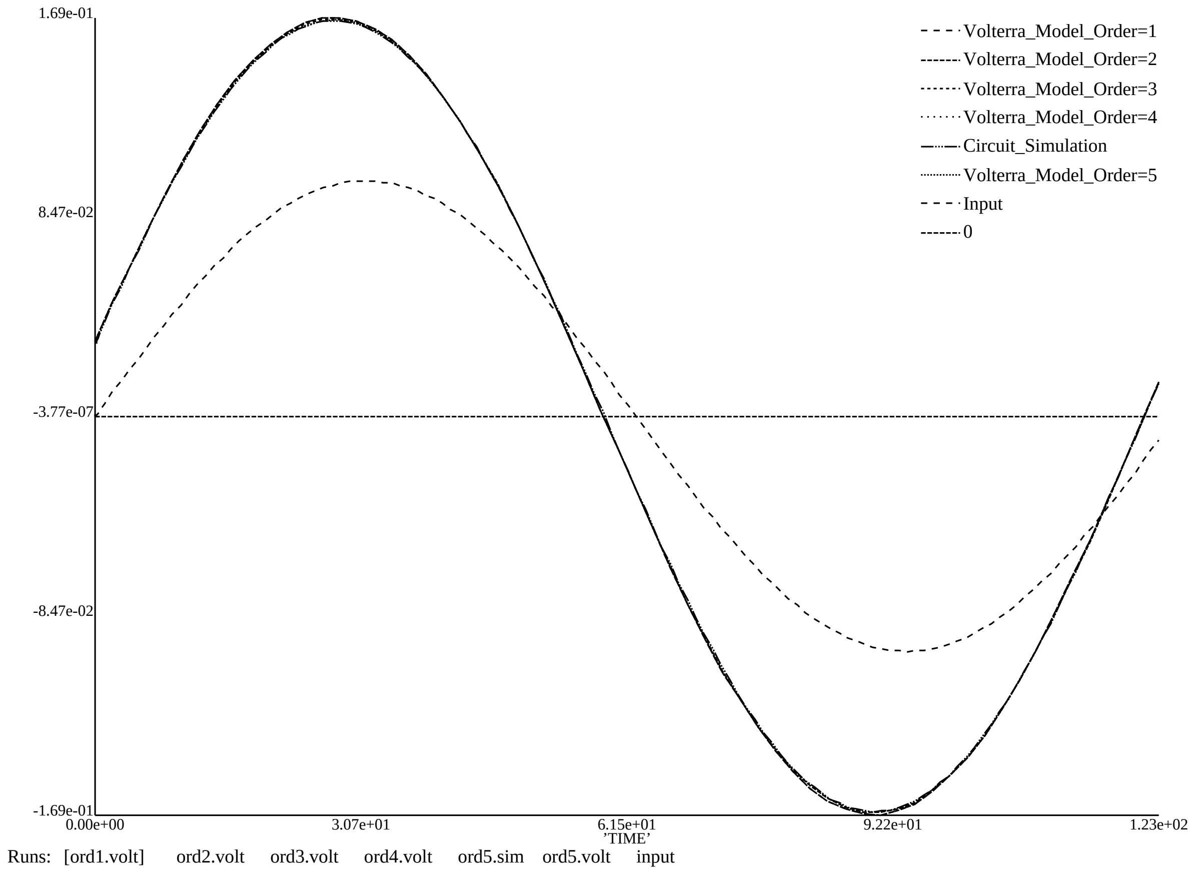

Figure 5.16: Output of the equalizer circuit shown in figure 5.10 with a single sinusoid 8 MHz input signal of 100mV. The Volterra model is truncated at various orders. Since the input signal is relatively low, only the first order contribution is significant, and thus the higher order models are indistinguishable from order=1.

Frequency	Input		Output	
(MHz)	Real	Imag.	Real	Imag.
0	0	0	-0.00384618	0
2	0.076699	0	0.0509467	-0.0313206
4	0.092379	0.056737	0.127815	0.0184941
6	0.060898	0.13464	0.134272	0.169927
8	0.037567	0.056706	0.0633615	0.0743186
10	0.073445	0.094801	0.132349	0.14164
12	0	0	-0.0100938	-0.0144004
14	0	0	-0.00250643	-0.0140619
16	0	0	-0.000236111	-0.011516
18	0	0	-0.000133463	-0.00812579
20	0	0	0.00386777	-0.00955325
22	0	0	0.00535749	-0.00760273
24	0	0	0.00463715	-0.00487514
26	0	0	0.00574414	-0.00363565
28	0	0	0.00321482	-0.00160746
30	0	0	0.00239918	0.000292326
32	0	0	0.00159212	0.000290928
34	0	0	0.000995979	0.000390799
36	0	0	0.000660513	0.000468547
38	0	0	0.000339444	0.000447438
40	0	0	0.000207854	0.000320117
42	0	0	5.34351e-05	0.000139171
44	0	0	5.94764e-05	9.18203e-05
46	0	0	2.15796e-05	2.31651e-05
48	0	0	2.04514e-06	7.53843e-06
50	0	0	1.13676e-06	3.14699e-06

Table 5.9: Input and output phasors as an example of the response predicted by the extracted Volterra model. The corresponding waveforms are shown in figure 5.17.

5.5.1 Problems

The DC or zero Hz output component is, in general, the most difficult to predict because in there are more distinct contributions to DC than to any other frequency. One problem with the extraction algorithm presented here is that previously calculated values of the Volterra nonlinear transfer function are used to subtract "known" contributions of the outputs of later simulations. Since there are so many different FIPD's, and thus points on the Volterra nonlinear transfer functions, that contribute to DC, the "known" contributions that are subtracted out are more sensitive to error. This error in the "known" contributions leads to error in the calculated values of the Volterra nonlinear transfer function, and these errors lead to larger errors later on in the extraction process. This effect can be seen in figure 5.19. The input in this case is a truncated Fourier series of a pulse train. The DC error of the fifth order Volterra model is about 0.15 Volts. If the DC component is removed from the output of the Volterra model and the simulated output, the higher order Volterra models give very good agreement as shown in figure 5.20. The output phasors for the fifth order Volterra model and the input phasors as shown in figure 5.19 are given in table 5.8.

5.5.2 Discussion

The computation needed in the extraction process is dominated by the simulations. The resource needed for simulations is determined by the number of simulations needed and the individual simulation times. The number of simulations needed is a function of the number of input frequencies and the assumed order of the system. On the other hand, the individual simulation times depend on the inputs to the simulator, the simulator itself, and the complexity of the circuit. In the equalizer example, the input frequencies were required to be commensurable because the simulator employed shooting methods and thus required a periodic input. Large differences in the values of the input frequencies would result in small time steps relative to the input period, and thus long run times and large output files. For our case, only simple harmonics were used as input frequencies, and thus, relative to the highest input frequency, the simulation period is kept to a minimum. If a frequency domain simulator were used in the extraction process, these would not be the dominating factors.

The extracted fifth order Volterra nonlinear transfer function of the equalizer circuit agrees well with simulated data with the exception of the DC component for some sets of input frequencies. This disagreement is due to the error accumulated during the extraction process.

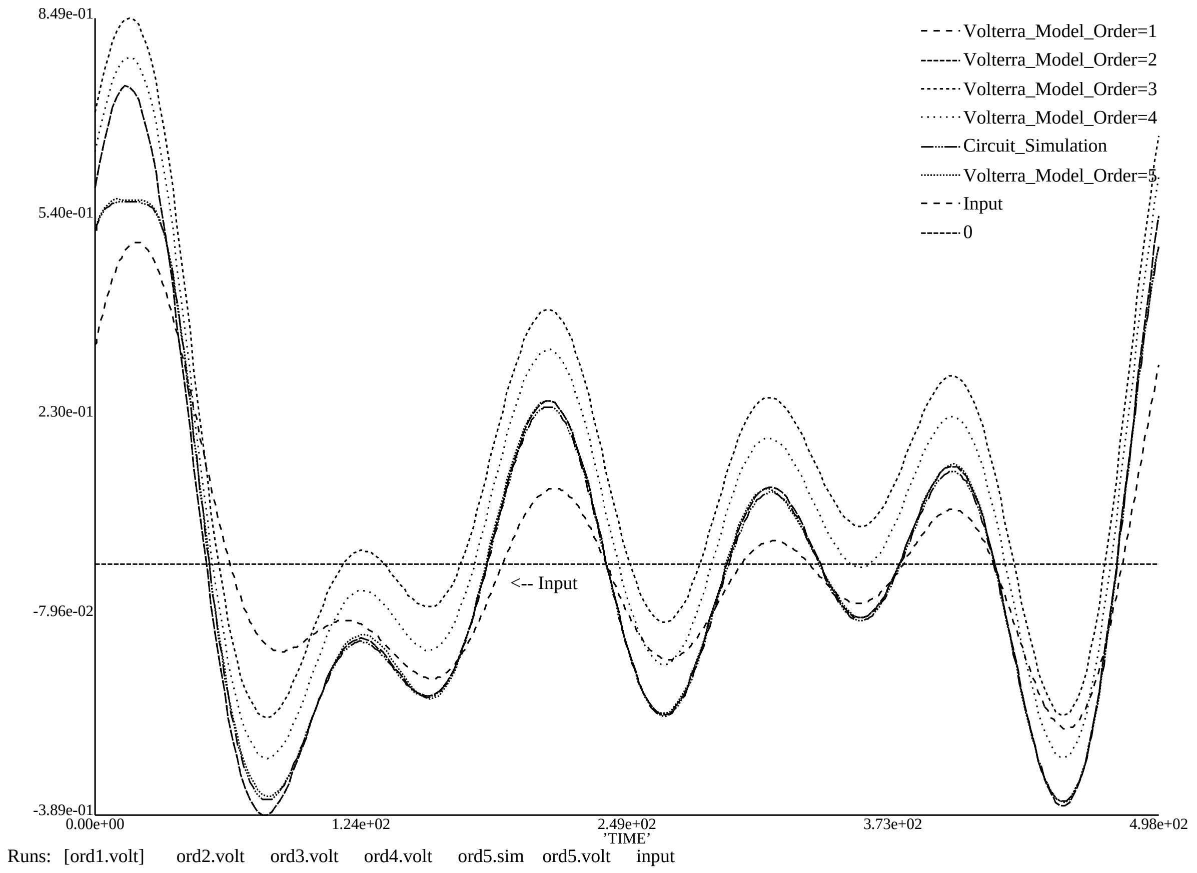

Figure 5.17: Sample output predicted by extracted Volterra model for arbitrary input with maximum allowable input amplitude. The peak value of the input is 0.5 volts. Higher order Volterra models more closely match the output as predicted by simulation. The full 5th order Volterra model output is almost indistinguishable from the simulated output.

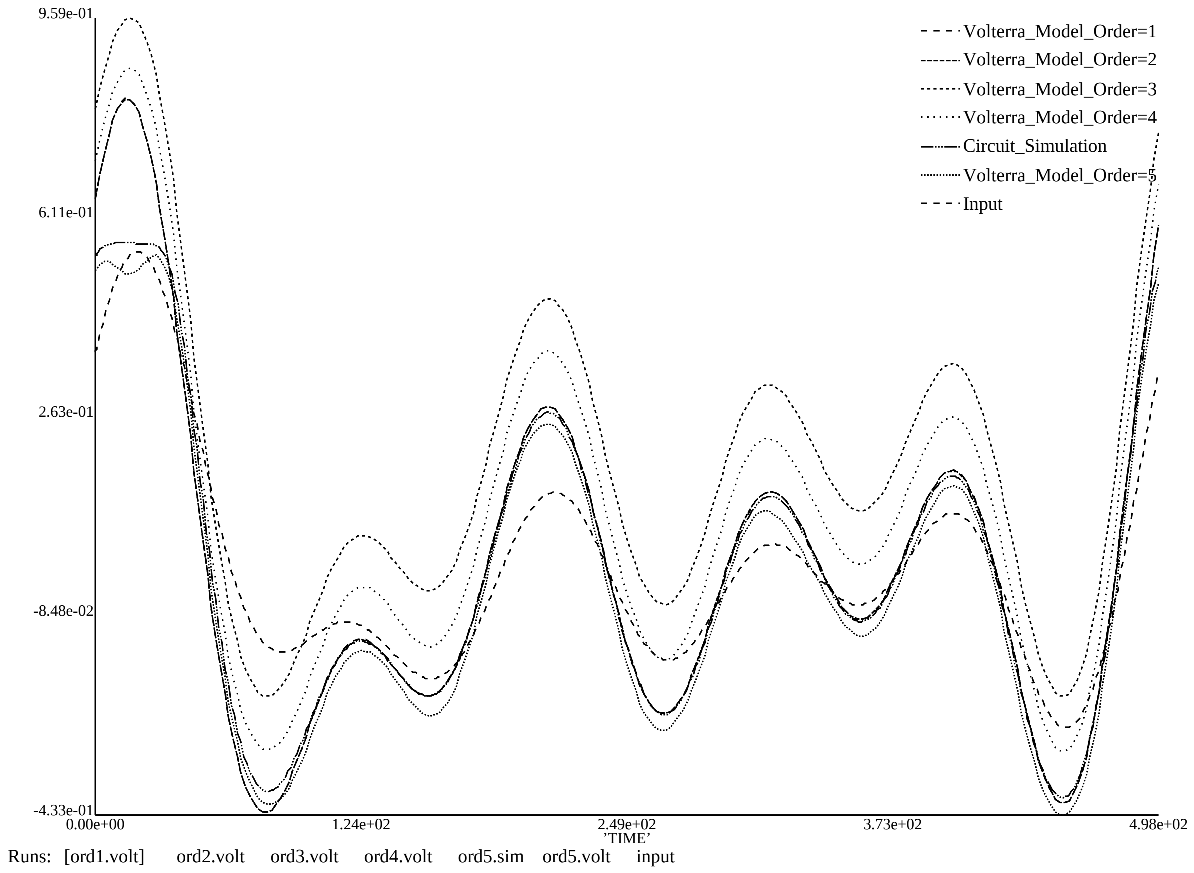

Figure 5.18: Sample output predicted by extracted Volterra model for arbitrary input with input amplitude 10 percent greater than maximum valid level. The peak value of the input is 0.55 volts. Higher order Volterra models more closely match the output as predicted by simulation.

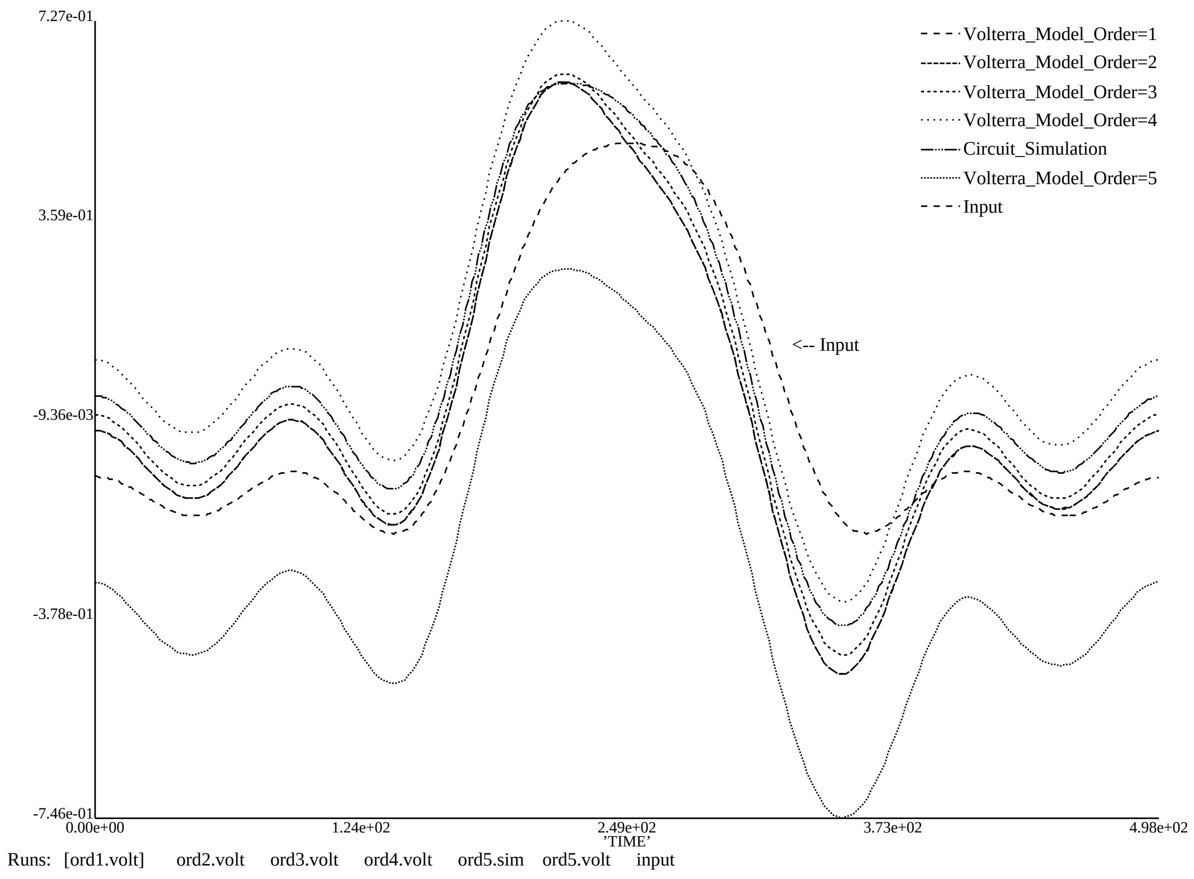

Figure 5.19: The characteristics of the equalizer with a pulse with truncated harmonics. The output predicted by the Volterra model closely matches the output predicted by time domain simulation if the Volterra model curve is shifted up by 0.15 volts. The phasors for the Volterra output are given in table 5.8.

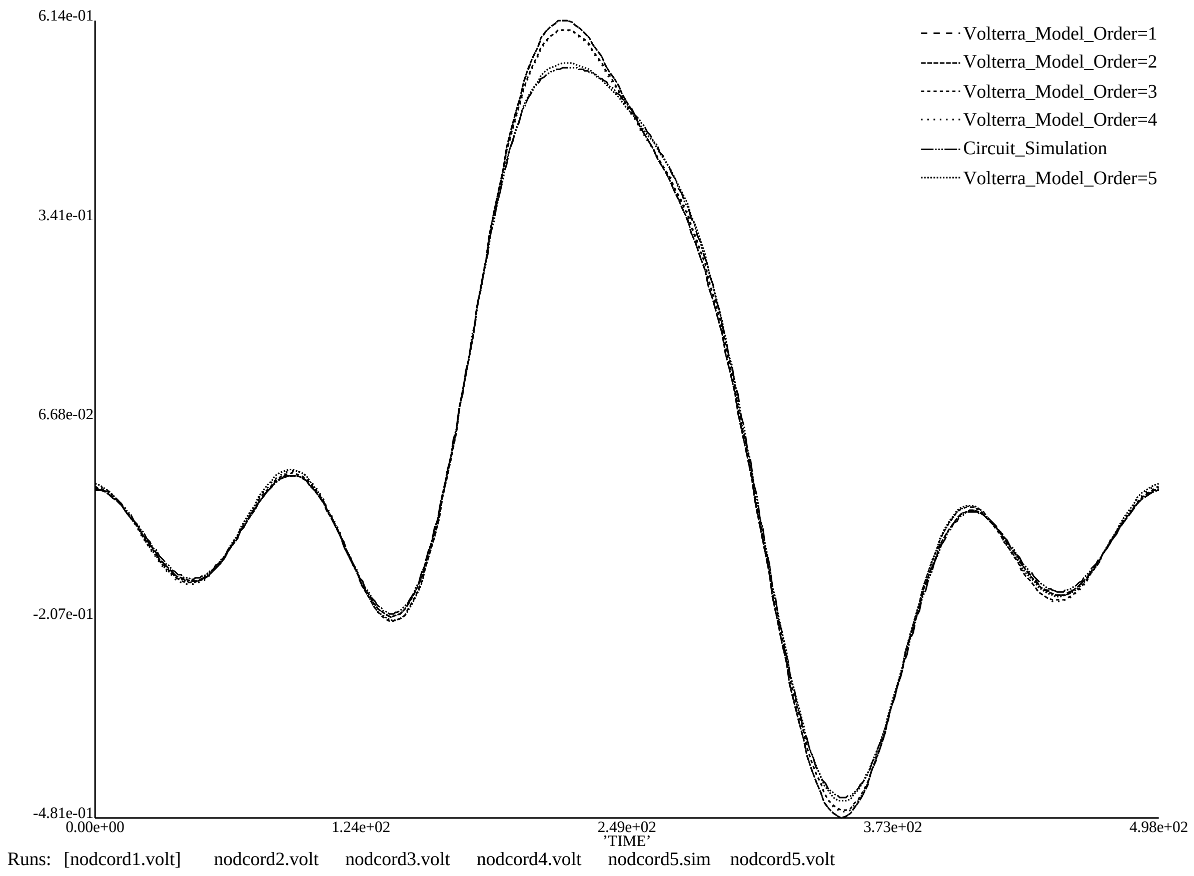

Figure 5.20: The characteristics of the equalizer with a pulse with truncated harmonics. This is figure 5.19 with the DC component removed for all signals. The higher order Volterra model more closely matches the simulated results.

Chapter 6

Conclusion

6.1 Summary

In this work, a new technique for the characterization of nonlinear analog circuits was developed. As a preliminary, the analysis of univariate and bivariate generalized power series and their relationship to univariate and bivariate Volterra series was examined both mathematically and experimentally.

The relationship between Volterra series analysis and generalized power series analysis was established for both the single input (univariate) and two input (bivariate) system representations. A general power series is equivalent to a Volterra system if the form of the Volterra nonlinear transfer function is restricted to a linear system followed by an ideal integer exponentiation $(\cdot)^n$ followed by a second linear system. A similar restriction applies to the bivariate case.

The experimentally derived bivariate power series characteristics of the drain current of a MESFET amplifier were used to calculate the bivariate Volterra nonlinear transfer function, and thus the output of a MESFET circuit can be represented by a nonlinear signal flow graph. If feedback is present in the flow graph, iterative techniques can be used to solve for the steady–state response entirely in the frequency domain. The predicted output calculated in this way matches experimental data closely.

A new general method to measure nonlinear transfer functions was presented. The method is applicable to extraction from circuit level simulation, and the results could be used in a table–based frequency–domain simulator. As an example of this extraction technique, several test circuits were examined including a simple linear circuit, a memoryless seventh order system, and a practical circuit whose transfer functions were unknown. Predicted values based on the extracted table show good agreement with the behavior predicted by circuit level simulations with the exception of the DC output value for some input conditions.

This new method of extracting Volterra nonlinear transfer functions lends to applications for large analog system simulation. The system is broken up into small subsystems, especially subsystems that are linear with some distortion. The extraction process is then applied to each subsystem. A harmonic balance type simulator could then be used to efficiently predict the behavior of the entire system. If a subsystem was used in different systems, or more than once in the same system, then the extraction procedure would only be performed once on the subsystem.

Furthermore, with a harmonic balance simulator, these black box characterizations of the subsystems could be mixed with device level elements. Thus a simulation would not necessarily be at the system level or the device level, but rather something in between. This ability greatly increases the flexibility of the simulator and allows the designer to seamlessly choose the level and detail of the simulation.

6.2 Suggestions for Further Work

Several aspects of this study can possibly be extended. Firstly, several mathematical treatments need to be further studied. The indexing scheme given in appendix A for determining the IPD's of a bivariate simulation is much faster than a brute force approach, but could possibly still be improved with further work. Secondly, a rigorous mathematical treatment of the best and necessary choices of the phase and amplitude for the test signals used in extraction could provide more insight and better extraction techniques.

Furthermore, practical applications using the extraction work could be investigated. The black–box extracted model can be integrated into a harmonic balance simulator as proposed in section 3.4. Ways to reduce the amount of data stored need to be investigated. Possibilities include:

1. Extract complex polynomial coefficient form for Volterra nonlinear transfer functions instead of table of values.

2. Investigate restricted forms of the Volterra nonlinear transfer functions and develop optimum ways to extract the necessary parameters.

The extraction examples given were based on a time–domain shooting method simulator. Integrating a harmonic balance simulator into the extraction system could improve the accuracy of the model and speed up the extraction process. This would also allow for incommensurable input frequencies. A harmonic balance simulator based approach could also be extended to perform the extraction process on a system consisting of subsystems that were previously characterized by the extraction process. Thus a complicated system described only by black box elements characterized by several sets of Volterra nonlinear transfer functions could be reduced to a single black box characterized by a single set of Volterra nonlinear transfer functions.

Solutions to the DC error problems could be investigated. One possible approach would be to form the problem as a large error minimization system. The existing extraction process would first be run to extract best guesses for all of the input values. Then these values (all calculated points on the Volterra nonlinear transfer functions) would be used as starting points in a many-input simulated annealing process. The existing simulation data would be used as the target behavior and thus be used to calculate the error function. Thus no new simulations would be needed. Different combinations of input sets, error functions, and annealing techniques could be investigated.

Lastly, an investigation of application of the algorithm used to choose the input conditions could be applied to analog test vector generation. As analog systems become more complex, the cost of testing in the manufacturing environment becomes more and more important.

References

[1] L.W. Nagel, "SPICE2: A Computer Program to Simulate Semiconductor Circuits," University of California at Berkeley, Memo No. ERL-M520," May, 1975.

[2] J.K. White and A. Sangiovanni-Vincentelli, "Relaxation techniques for the simulation of VLSI circuits," 1986, Kluwer Academic Publishers, Norwell, Massachusetts.

[3] M. Vlach, "Modeling and simulation with Saber," *Proceedings of the 3rd Annual IEEE ASIC Seminar and Exhibit*, 1990

[4] H.A. Mantooth and M. Vlach, "Beyond SPICE with Saber and MAST," *IEEE International Symposium on Circuits and Systems*, May 1992, pp. 77-80.

[5] G.R. Boyle, B.M. Cohn, D.O. Pederson and J.E. Solomon, "Macromodeling of Integrated Circuit Operational Amplifiers," *IEEE J. Solid–State Circuits,* vol. SC-9, no. 6, pp. 353–363, December 1974.

[6] L.O. Chua and P.M. Lin, "Computer–Aided Anlaysis of Electronic Circuits", 1975, Prentice-Hall, Inc. Englewood Cliffs, New Jersey.

[7] B. Epstein, "Microwave System Simulation," Workshop on Analog Circuit Engineering, Sheraton Imperial, Research Triangle Park, North Carolina, October 1 & 2, 1990. North Carolina State University in cooperation with the IEEE Circuits & Systems Society & the IEEE Microwaver Theory & Techniques Society.

[8] R.J. Gilmore and M.B. Steer, "Nonlinear Circuit Analysis Using the Method of Harmonic Balance–A Review of the Art. Part I. Introductory Concepts," *International Journal of Microwave and Millimeter–Wave Computer–Aided Engineering,* Vol. 1, No. 1, 1991, pp. 22–37.

[9] M.B. Steer, "Simulation of Nonlinear Microwave Circuits — an Historical Perspective and Comparisons," in *IEEE MTT-S International Microwave Symposium Digest*, June 1991, pp. 599-602.

[10] E. Filseth and J. Sanders, "Tech Talk," *Validator*, Valid Logic Systems, Vol. 3, No. 1, Spring 1991, pp. 10–15.

[11] J. Lee, "Simulating A successive Approsimation A/D Converter," *Australian Electronics Engineering*, March 1991, pp. 54-60.

[12] M.B. Steer and P.J. Khan, "An Algebraic Formula for the Output of a System with Large-Signal, Multifrequency Excitation," *Proceedings of the IEEE*, Vol. 71, No. 1, pp. 177-179, January 1983.

[13] R.G. Sea, "An Algebraic Formula for Amplitudes of Intermodulation Products Involving an Arbitrary Number of Frequencies," *Proceedings of the IEEE*, Vol. 56, pp. 1388-1389, August 1968.

[14] R.G. Sea, "On the Computation of Intermodulation Products for a Power Series Nonlinearity," *Proceedings of the IEEE*, pp. 337-338, March 1969.

[15] R.G. Sea and A.G. Vacroux, "On the computation of intermodulation products for a power series nonlinearity," *Proc. IEEE,* March 1969, pp. 337-338

[16] G.L. Heiter, "Characterisation of Nonlinearities in Microwave Devices and Systems," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-21, pp. 797-805, December 1973.

[17] G.W. Rhyne and M.B. Steer, "Simulation of Intermodulation Distortion in MESFET Circuits with Arbitrary Frequency Separation of Tones," *1986 IEEE MTT-S International Microwave Symposium Digest*, pp. 547-550, 1986.

[18] G.W. Rhyne and M.B. Steer, "Generalized Power Series Analysis of Intermodulation Distortion in a MESFET Amplifier: Simulation and Experiment," *IEEE Transactions on Microwave Theory and Techniques*, Vol. MTT-35, No. 12, pp. 1248-1255, December 1987.

[19] G.W. Rhyne, M.B. Steer, and B.D. Bates, "Frequency-Domain Nonlinear Circuit Analysis Using Generalized Power Series," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 36, No. 2, pp. 379-387, February 1988.

[20] C.R. Chang, M.B. Steer, and G.W. Rhyne, "Frequency-Domain Spectral Balance Using the Arithmetic Operator Method," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-37, November 1989.

[21] V.Volterra, "Sopra le Funzioni che Dipendono da Altre Funzioni, Nota 1," *Rend. Lincei Ser. 4*, Vol. 3, pp. 97-105, 1887.

[22] V.Volterra, *The Theory of Functionals and of Intergral and Integro-differential Equations.* Blackie & Sons, London 1930 (also Dover, New York 1959).

[23] A.K. Ahmed, "Modelling the Thermoregulatory Control System Using Kernel Method," *Proceedings of the Annual Conference on Engineering in Medicine and Biology*, Vol. 11, Pt. 1, Alliance for Engineering in Medicine and Biology, Bethesda, Maryland, pp. 308-309, 1989.

[24] J. Yang and C. Fan, "Modeling of Nonlinear and Active Cochlear Mechanics and its Volterra-Series Analysis," *Applied Mathematics and Computation*, Vol. 43, Elsevier Science Publishing Co., Inc., New York, pp. 241-263, 1991.

[25] Y. Shi and K.E. Hecox, "Nonlinear System Identification by m-Pulse Sequences: Application to Brainstem Auditory Evoked Responses," *IEEE Transactions on Biomedical Engineering*, Vol. 38, No. 9, pp. 834-845, September 1991.

[26] E.J. Powers and R.W. Miksad, "Applications of Frequency and Wavenumber Nonlinear Digital Signal Processing to Nonlinear Hydrodynamics Research," Applied Hydrodynamics Research Program Report, Office of Naval Research, Contract N00167-88-K-0049, December 18, 1989.

[27] Y.S. Cho, S.B. Kim, E.J. Powers, and R. W. Miksad, "Stabilization of Moored Vessels Using a Second-Order Volterra Filter and Feedforward Compensator," *Proceedings of the 9th Conference on Offshore Mechanics and Arctic Engineering*, Houston, Texas, pp. 165-170, February 18-23, 1990.

[28] Y.S. Cho, S.B. Kim, E.L. Hixson, and E.J. Powers, "Nonlinear Distortion Analysis Using Digital Higher-Order Coherence Spectra," *ICASSP 90, 1990 International Conference on Acoustics, Speech and Signal Processing, IEEE*, Vol. 2, pp. 1165-1168, April 1990.

[29] Y.S. Cho, S.B. Kim, E.J. Powers, R.W. Miksad, and F.J. Fischer, "Stabilization of Moored Vessels Using a Second-Order Volterra Filter and Feedforward Compensator," *Journal of Offshore Mechanics and Arctic Engineering*, Vol. 113, pp. 137-141, May 1991.

[30] S.B. Kim, E.J. Powers, R.W. Miksad, F.J. Fischer, and J.Y. Hong, "Spectral Estimation of Second-Order Wave Forces on a TLP Subject to NonGaussian Irregular Seas," *Proceedings of the 9th Conference on Offshore Mechanics and Arctic Engineering*, Houston, Texas, pp. 159-164, February 18-23, 1990.

[31] T. Kinoshita, S. Takase, and K. Takaiwa, "Probability Density Function of a Slow Drift Motion of a Moored Floating Structure in Random Seas," *Proceedings of the International Offshore Mechanics and Artic Engineering Symposium*, Vol. 2, The American Society of Mechanical Engineers (ASME), New York, pp. 97-105, 1990.

[32] M.G. Donley and P.D. Spanos, "Stochastic Response of a Tension Leg Platform to Viscous Drift Forces," *Journal of Offshore Mechanics and Artic Engineering*, Vol. 113, pp. 148-155, May 1991.

[33] D.K. Sung, "Application of Volterra Series to Modeling an Elastomer Force-Displacement Relation," *J. Korean Inst. Telemat. Electron.*, Vol. 26, No. 6, pp. 868-875, June 1989.

[34] D.M. Storer and G. R. Tomlinson, "Higher Order Frequency Response Functions and Their Relations to Practical Structures," *Proceedings of the International Modal Analysis Conference - IMAC*, Vol. 2, Union Coll, Graduate & Continuing Studies, Schenectady, NY, pp. 1206-1214, 1991.

[35] D.M. Storer and G.R. Tomlinson, "An Explanation of the Cause of the Distortion in the Transfer Function of a Duffing Oscillator Subject to Sine Excitation," *Proceedings of the International Modal Analysis Conference - IMAC*, Vol. 2, pp. 1197-1205, 1991.

[36] A. Frachebourg, "Identification of Nonlinearities - Application of the Volterra-Model to Discrete MDOF Systems; Possibilities and Limits," *Proceedings of the International Modal Analysis Conference - IMAC*, Vol. 2, Union Coll, Graduate & Continuing Studies, Schenectady, NY, pp. 1029-1035, 1991.

[37] G. Mazzarella and G. Panariello, "Reconstruction of One-Dimensional Dielectric Profiles," *J. Opt. Soc. Am. A.*, Vol. 8, No. 10, pp. 1622-1629, October 1991.

[38] R. Hermann, "Volterra Modeling of Digital Magnetic Saturation Recording Channels," *IEEE Transactions on Magnetics*, Vol. 26, No. 5, pp. 2125-2127, September 1990.

[39] T.K. Biswas and W.F. McGee, "Volterra Series Analysis of Semiconductor Laser Diode," *IEEE Photonics Technology Letters*, Vol. 3, No. 8, pp. 706-708, August 1991.

[40] Y. Hu, J.J. Obregon, and J.C. Mollier, "Nonlinear Analysis of Microwave FET Oscillators Using Volterra Series," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 37, No. 11, September 1989.

[41] R. Tymerski, "Volterra Series Modelling of Power Conversion Systems," *IEEE Power Electronics Specialists Conference*, San Antonio, TX, USA, pp. 786-791, June 1990.

[42] S. Szczepanski and R. Schaumann, "Effects of Weak Nonlinearities in Transconductance-Capacitor Filters," *ISCAS*, pp. 1055-1058, 1989.

[43] S. Narayanan, "Application of Volterra Series to Intermodulation Distortion Analysis of Transistor Feedback Amplifiers," *IEEE Trans. Circuits Theory*, Vol. CT-17, pp. 518-527, November 1970.

[44] A. Borys and J. Vlach, "A Cad-Oriented Nonlinear Model of the Operational Amplifier in the Frequency-Domain," *Scientia Electrica*, Vol. 33, No. 1, pp. 3-17, 1987.

[45] Y. Hu, J.C. Mollier, and J. Obregon, "A New Method of Third-Order Intermodulation Reduction in Nonlinear Microwave Systems," *IEEE Transactions on Microwave Theory and Techniques*, Vol. MTT-34, No. 2, pp. 245-250, February 1986.

[46] S.A. Maas, "Two-Tone Intermodulation in Diode Mixers," *IEEE Transactions on Microwave Theory and Techniques*, Vol. MTT-35, pp. 307-314, March 1987.

[47] S.A. Maas and A.M Crosmun, "Modeling the Gate I/V Characteristic of a GaAs MESFET for Volterra-Series Analysis," The Aerospace Corporation Laboratory Operations, El Segundo, CA 90245, September 30, 1989.

[48] Y.A. Volkov, "Analysis of Nonlinear Devices in a Radio-Receiving Channel in the Time Domain," *Scripta Technica, Inc.*, pp. 35-37, 1991.

[49] B.E. Zhelezovskii and A.P. Kozyrev, "Use of a Typical Radio Section to Investigate Devices with Amplitude-Phase Conversion," *Radioelectron Commun. Syst.*, Vol. 32, No. 1, pp. 72-74, 1989.

[50] R.G. Meyer, M.J.Shensa, and R. Eschenbach, "Cross Modulation and Intermodulation in Amplifiers at High Frequencies," *IEEE Journal of Solid-State Circuits*, Vol. SC-7, No. 1, pp. 16-23, February 1972.

[51] G. M. Lambrianou and C.S. Aitchison, "Power characterisation of a MESFET amplifier using small signal measurements and Volterra series ," *IEEE MTT-S Digest*, pp. 409-412, 1985.

[52] G.M. Lambrianou and C.S. Aitchison, "Optimization of Third-Order Intermodulation Product and Output Power from an X-Band MESFET Amplifier using Volterra Series Analysis," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-33, pp. 1395-1403, December 1985.

[53] B.S. Troitskiy, "Analysis of Nonlinear Amplifiers with Feedback Based on Power-Series Inversion," *Telecommunications and Radio Engineering*, Vol. 43, No. 8, pp. 82-83, August 1988.

[54] S. Narayanan, "Transistor Distortion Analysis Using Volterra Series Representation," *Bell Syst. Tech. J.*, pp. 991-1024, May-June 1967.

[55] R.A. Minasian, "Intermodulation Distortion Analysis of MESFET Amplifiers using the Volterra Series Representation," *IEEE Trans. on Microwave Theory and Techniques*, Vol. MTT-28, pp. 1-8, January 1980.

[56] I.W. Sandberg, "Series Expansions for Nonlinear Systems," *Circuits Systems Signal Processing, The Bell System Technical Journ.*, Vol. 61, pp. 159-199, February 1982.

[57] I.W. Sandberg, "On Volterra Expansions for Time-Varying Nonlinear Systems," *IEEE Transactions on Circuits and Systems*, Vol. CAS-30, No. 2, pp. 61-67, February 1983.

[58] V. Raman and R. Liu, "Existence of Volterra Series in the Design of Feedbacack Systems for Nonchaotic Motion," *Proceedings of the 28th Conference on Decision and Control*, Tampa, Florida, pp. 1060-1061, December 1989.

[59] A. Halme, J. Orava, and H. Blomberg, "Polynomial Operators in Non-Linear Systems Theory," *Int. J. Systems Sci.*, Vol. 2, No. 1, pp. 25-47, 1971.

[60] S. Boyd and L.O. Chua, "Fading Memory and the Problem of Approximating Nonlinear Operators with Volterra Series," *IEEE Transactions on Circuits and Systems*, Vol. CAS-32, No. 11, pp. 1150-1161, November 1985.

[61] D. Ball and I.W. Sandberg, "Systems Which Possess g- and h- Representations," *Circuits Systems Signal Process*, Vol. 8, No. 2, pp. 145-162, 1989.

[62] G. Palm and T. Poggio, "The Volterra Representation and the Wiener Expansion: Validity and Pitfalls," *SIAM J. Appl. Math.*, Vol. 33, No. 2, pp. 195-216, September 1977.

[63] H.K. Thapar and B.J. Leon, "Transform-Domain and Time-Domain Characterization of Nonlinear Systems with Volterra Series," *IEEE Transactions on Circuits and Systems*, Vol. CAS-31, No. 10, October 1984.

[64] Y.H. Ku, "Volterra-Wiener Functionals for the Analysis of Nonlinear Systems," *Journal of The Franklin Institute*, Vol. 281, No. 1, pp. 9-26, January 1966.

[65] V.H.N. Minh and G. Jacob, "Symbolic Calculus and Volterra Series," *IFAC Nonlinear Control Systems Design*, Capri, Italy, pp. 149-154, 1990.

[66] J.C. Peyton Jones and S.A. Billings, "Interpretation of Non-linear Frequency Response Functions," *Int. J. Control*, Vol. 52, No. 2, pp. 319-346, August 1990.

[67] J.C. Peyton Jones and S.A. Billings, "Describing Functions, Volterra Series, and the Analysis of Non-linear Systems in the Frequency Domain," *Int. J. Control*, Vol. 53, No. 4, pp. 871-887, 1991.

[68] M. Wright and J.K. Hammond, "The Implications of Using Linear Frequency Response Function Estimators on Nonlinear Systems - a Functional Series Expansion Approach," *Proceedings of the International Modal Analysis Conference - IMAC*, Vol. 2, pp. 1193-1196, 1991.

[69] S.A. Al-Baiyat and M.K. Sain, "A Volterra Method for Nonlinear Control Design," *IFAC Nonlinear Control Systems Design*, Capri, Italy, pp. 125-130, 1990.

[70] L.R. Hunt, D.A. Linebarger, and R.D. DeGroat, "Realizations of Nonlinear Systems," *Circuits Systems Signal Process*, Vol. 8, No. 4, pp. 487-506, 1989.

[71] G. Stefani and P. Zezza, "Necessary Conditions for Optimality Via Volterra Approximations," *IFAC Nonlinear Control Systems Design*, Capri, Italy, pp. 321-326, 1990.

[72] S.W. Director, "A method for quick determination of the periodic steady-state in nonlinear networks," *Allerton Conf. Circuit Syst. Theory*, Oct. 1971, pp. 131-139.

[73] T. J. Aprile and T. N. Trick, "Steady-State Analysis of Nonlinear Circuits with Periodic Inputs," *Proc. IEEE*, Vol. 60, pp. 108-114, January 1972.

[74] T.J. Aprille and T.N. Trick, "A computer algorithm to determine the steady-state response of nonlinear oscillators," *IEEE Trans. Circuit Theory*, Vol. CT-19, July 1972, pp. 354-360.

[75] F.R. Colon and T.N. Trick, "Fast Periodic Steady-State Analysis for Large-Signal Electronic Circuits," *IEEE J. Solid-State Circuits*, Vol. SC-8, pp. 260-269, August 1973.

[76] S.W. Director and K.W. Current, "Optimization of forced nonlinear periodic circuits," *IEEE Trans. Circuits Syst.*, Vol. CAS-23, June 1976, pp. 329–335.

[77] M.S. Nakhla and F.H. Branin, "Determining the Periodic Response of Non-linear Systems by a Gradient Method," *Circuit Theory and Applications*, Vol. 5, pp. 255-273, 1977.

[78] R.J. Gilmore and M.B. Steer, "Nonlinear Circuit Analysis Using the Method of Harmonic Balance–A Review of the Art. Part II. Advanced Concepts," *International Journal of Microwave and Millimeter–Wave Computer–Aided Engineering,* Vol. 1, No. 2, 1991, pp. 159–180.

[79] S.A. Maas, *Nonlinear Microwave Circuits*, Norwood, MA, Artech House, pp. 190-207, 1988.

[80] S.A. Maas, "A General-Purpose Computer Program for the Volterra-Series Analysis of Nonlinear Microwave Circuits," *1988 IEEE MTT-S Digest*, pp. 311-314, 1988.

[81] A.M. Crosmun and S.A. Maas, "Minimization of Intermodulation Distortion in GaAs MESFET Small-Signal Amplifiers," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-37, pp. 1411-1417, September 1989.

[82] V.D. Dmitriev and A.I. Silyutin, "A Method of Designing Broadband Non-linear Circuits Based on a Modification of the Nonlinear Current Method," *Radioelectron Commun. Syst.*, Vol. 29, No. 11, pp. 50-55, 1986.

[83] P. Wambacq, G. Gielen, and W. Sansen, "Symbolic Simulation of Harmonic Distortion in Analog Integrated Circuits with Weak Nonlinearities," *1990 IEEE International Symposium on Circuits and Systems*, Vol. 1, pp. 536-539, May 1990.

[84] A.G. Zharkoy, "Calculation of Balanced Currents of Nonlinear Elements at Arbitrary Frequency Selections," *Telecommunications and Radio Engineering*, Vol. 44, No. 2, pp. 133-134, February 1989.

[85] A.G. Zharkoy and V.I. Tuyev, "Calculation of Nonlinear Equivalent Current Sources in Multielectrode Active Elements," *Sov. J. Commun. Technol. Electron.*, Vol. 35, No. 2, pp. 61-69, 1990.

[86] J.R. Fayos and S.J. Nightingale, "Determination of the Transfer Function of a Nonlinear Circuit for Prediction of Intermodulation Characteristics," *IEEE - MTT S International Microwave Symposium Digest*, Baltimore, MD, USA, pp. 499-502, June 1986.

[87] A.I. Silyutin and V.D. Dmitriyev, "An Algorithm for Analyzing Nonlinear Distortions Based on a Modified Nonlinear-Current Method," *Radioelectron Commun. Syst.*, Vol. 44, No. 1, pp. 76-80, January 1989.

[88] E. Isaacson and H.B. Keller, "Analysis of Numerical Methods", 1966, John wiley & Sons, Inc.

[89] R.A. Rutenbar, "Simulated Annealing Algorithms: An Overview," *IEEE Circuits and Devices Magazine*, pp. 19-26, January 1989.

[90] G.L. Bilbro, M.B. Steer, R.J. Trew, C.R. Chang, and S.G. Skaggs, "Extraction of the Parameters of Equivalent Circuits of Microwave Transistors Using Tree Annealing," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 38, No. 11, pp. 1711-1718, November 1990.

[91] E. Van den Eijnde, J. Schoukens, and J. Renneboog, "Parameter Estimation in Rational Volterra Models," *IEEE ISCAS 1987*, pp. 110-114, 1987.

[92] E. Van den Eijnde and J. Schoukens, "Parameter Estimation in Strongly Nonlinear Circuits," *IEEE Instrumentation and Measurement Technology Conference*, San Jose, California, USA, pp. 378-383, February 1990.

[93] E. Van den Eijnde and J. Schoukens, "Parameter Estimation in Strongly Nonlinear Circuits," *IEEE Transactions on Instrumentation and Measurement*, Vol. 39, No. 6, pp. 853-859, December 1990.

[94] N. Wiener, "Extrapolation, Interpolation and Smoothing of Stationary Time Series", John Wiley, 1949.

[95] M. Schetzen, "Measurement of the Kernels of a Non-linear System of Finite Order," *Research Laboratory of Electronics, M.I.T.*, pp. 251-263, 1964.

[96] Y.W. Lee and M. Schetzen, "Measurement of the Wiener Kernels of a Non-linear System by Cross-correlation," *Research Laboratory of Electronics, M.I.T.*, pp. 237-254, 1965.

[97] M. Schetzen, "Nonlinear System Modelling Based on the Weiner Theory," *Proc. IEEE*, Vol. 69, pp. 1557-1573, December 1981.

[98] Y.H. Ku, "Theory of Nonlinear Systems," *Journal of the Franklin Institute*, Vol. 315, No. 1, pp. 1-26, January 1983.

[99] M.A. Shcherbakov, "Algorithm for Calculating Wiener Kernels of Nonlinear Systems in the Frequency Domain," *Kibern. Vychisl. Tekh. (Ukrainian SSR) Cybern. Comput. Technol. (USA)*, No. 79, pp. 79-90, 1988.

[100] K. Yoshine and N. Ishii, "Estimation of Kernel Function of Non-linear System by Gaussian Non-white Noise," *Int. J. Systems Sci.*, Vol. 22, No. 3, pp. 595-603, 1991.

[101] Y.S. Cho, W. T. Oh, S.B. Kim, and E.J. Powers, "Testing for Gaussianity in Nonlinear System Identification," *IEEE International Symposium on Circuits and Systems, Proceedings of the IEEE*, Vol. 2, pp. 1450-1453, 1990.

[102] S.J. Gifford and G.R. Tomlinson, "Recent Advances in the Application of Functional Series to Non-linear Structures," *Journal of Sound and Vibration*, pp. 289-317, 1989.

[103] F. Thouverez and L. Jezequel, "Identification of Non-linearity Joints from Dynamical Tests," *Proceedings of the International Modal Analysis Conference - IMAC*, Vol. 2, pp. 1583-1588, 1991.

[104] A.E.C. Pece, A.S. French, M.J. Korenberg, and J.E. Kuster, "Nonlinear Mechanisms for Gain Adaptation in Locust Photoreceptors," *Biophysical Journal*, Volume 57, pp. 733-743, April 1990.

[105] M.J. Korenberg and I.W. Hunter, "The Identification of Nonlinear Biological Systems: Wiener Kernel Approaches," *Annals of Biomedical Engineering*, Vol. 18, pp. 629-654, 1990.

[106] E. Bedrosian and S.O. Rice, "The Output Properties of Volterra Systems (Nonlinear Systems with Memory) Driven by Harmonic and Gaussian Inputs," *Proceedings of the IEEE*, Vol. 59, No. 12, pp. 1688-1707, December 1971.

[107] P. Zhang and Y. Song, "Discrete Frequency-Domain Modeling and Measuring of Mildly Nonlinear of Volterra Transfer Functions with a Set of Special Sinusoid Signals," *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pp. 544-547, June 1989.

[108] S. Boyd, Y.S. Tang, and L.O. Chua, "Measuring Volterra Kernels," *IEEE Transactions on Circuits and Systems*, Vol. CAS-30, No. 8, pp. 571-577, August 1983.

[109] L.O. Chua and Y. Liao, "Measuring Volterra Kernels (II)," *International Journal of Circuit Theory and Applications*, Vol. 17, John Wiley & Sons, Ltd., pp. 151-190, 1989.

[110] M.B. Steer, P.J. Khan, and R.S. Tucker, "Relationship Between Volterra Series and Generalized Power Series," *Proceedings of the IEEE*, Vol. 71, No. 12, pp. 1453-1454, December 1983.

[111] M. Abramowitz and I.A. Stegun, *Handbook of Mathematical Functions*, Dover Publications, Inc., New York, 1965.

[112] I.P. Norenko, Y.A. Yevstifeyev, and V.B. Manichev, "A steady-state analysis method for multiperiod electronic circuits," *Radiotekhnika*, No. 11, 1987, pp. 86-89.

[113] W.T. Weeks, A.J. Jimenez, G.W. Mahoney, D. Metha, H. Qassemzadeh, and T.R. Scott, "Algorithms for ASTAP – A Network-Analysis Program," *IEEE Trans. Circuit Theory*, Vol. CT-20, pp. 628-634, November 1973.

[114] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes in C*, CAmbridge University Press, 1988.

[115] Bruce W. Char, Keight O. Geddes, Gaston H. Gonnet, and Stephen M. Watt, *Maple User's Guide*, WATCOM Publications Limited, Waterloo, Ontario, 1985.

[116] S.L. Bussgang, L. Ehrman, and J.W. Graham, "Analysis of Nonlinear Systems with Multiple Inputs," *Proc. IEEE*, Vol. 62, No. 8, pp. 1088-1119, August 1974.

Other Useful References Not Cited in the Text

[117] G.W. Rhyne, M.B. Steer, and B.D. Bates, "Analysis of Nonlinear Circuits Driven by Multi-Tone Signals Using Generalized Power Series," *1987 IEEE International Symposium on Circuits and Systems*, pp. 903-906, 1987.

[118] H.W. Schussler and Y. Dong, "A New Method for Measuring the Performance of Weakly Nonlinear and Noisy Systems," *Frequenz*, Vol. 44, No. 3-4, pp. 82-87, March-April 1990.

[119] H.W. Wong-Lam, "Characterization and Measurement of Nonlinear Bit Shifts in Digital Magnetic Tape Recording," *Eighth International Conference on Video, Audio and Data Recording*, IEEE, pp. 84-92, April 1990.

[120] P. Zhang and Y. Song, "An Algorithm to Measure High Order Volterra Kernels," *1990 IEEE International Symposium on Circuits and Systems*, Vol. 1, New Orleans, LA, USA, pp. 184-187, May 1990.

[121] L.O.Chua and Y. Liao, "Measuring Volterra Kenels III: How to Estimate the Highest Significant Order," *International Journal of Circuit Theory and Applications*, Vol. 19, John Wiley & Sons, Ltd., pp. 189-209, 1991.

[122] S.O. Rice, "Volterra Systems With More Than One Input Port–Distortion in a Frequency Converter," *The Bell System Technical Journal*, Vol. 52, No. 8, American Telephone and Telegraph Company, pp. 1255-1271, October 1973.

[123] A. Prochazka, "Degradation of Performance Parameters in a Television Relay System-I," *IEEE Transactions on Broadcasting*, Vol. BC-25, No. 3, pp. 90-99, September 1979.

[124] A.A.M. Saleh, "Matrix Analysis of Mildly Nonlinear, Multiple-Input, Multiple-Output Systems With Memory," *The Bell System Technical Journal*, Vol. 61, No. 9, pp. 2221-2243, November 1982.

[125] A. Nassirharand and J.H. Taylor, "Frequency-Domain Modeling of Nonlinear Multivariable Systems," *IFAC Computer Aided Design in Control Systems*, Beijing, PRC, pp. 301-305, 1988.

[126] T. Narhi, "Multi-Frequency Analysis of Nonlinear Circuits Using One and Two-Dimensional Series Expansions," *Proceedings of the Nineteenth European Microwave Conference*, pp. 375-379, September 1989.

[127] J. Duvernoy, "Volterra-Wiener Partially Coherent Imaging Systems: Three Dimensional Objects and Generalized G Funtionals," *Optical Society of America*, Vol. 7, No. 5, pp. 809-819, May 1990.

[128] C.K. An, E.J. Powers, and C.P. Ritz, "A Digital Method of Modeling Two-Input Quadratic Systems with General Random Inputs," *IEEE Transactions on Signal Processing*, Vol. 39, No. 10, pp. 2320-2323, October 1991.

[129] S.R. Parker and F.A.Perry, "A Discrete ARMA Model for Nonlinear System Identification," *IEEE Transactions on Circuits and Systems*, Vol. CAS-28, No. 3, pp. 224-233, March 1981.

[130] M. Maqusi, "Analysis and Modeling of Intermodulation Distortion in Wide-Band Cable TV Channels," *IEEE Transactions on Communications*, Vol. Com-35, No. 5, pp. 568-572, May 1987.

[131] L.R. Hunt and P. Whitney, "Nonlinear Filter Design," *Analysis and Control of Nonlinear Systems*, Elsevier Science Publishers B.V., pp. 453-458, 1988.

[132] E. Biglieri, S. Barberis, and M. Catena, "Analysis and Compensation of Nonlinearities in Digital Transmission Systems," *IEEE Journal on Selected Areas in Communications*, Vol. 6, No. 1, pp. 42-51, January 1988.

[133] X.Y. Gao, W.M. Snelgrove, and D.A. Johns, "Nonlinear IIR Adaptive Filtering Using a Bilinear Structure," *1989 IEEE International Symposium on Circuits and Systems*, Vol. 3, IEEE, pp. 1740-1743, May 1989.

[134] G. Ramponi, "Bi-Impulse Response Design of Isotropic Quadratic Filters," *Proceedings of the IEEE*, Vol. 78, No. 4, pp. 665-677, April 1990.

[135] S.W. Nam and E.J. Powers, "On the Linearization of Volterra Nonlinear Systems Using Third-Order Inverses in the Digital Frequency-Domain," *1990 IEEE International Symposium on Circuits and Systems*, New Orleans, LA, pp. 407-410, May 1990.

[136] S.A Alshebeili and A. Venetsanopoulos, "A Volterra–Type Non–Gaussian Process: Parameter Estimation and Spectral Analysis," *1990 Gulf Digital Signal Processing Symposium*, pp. B3/1-B3/10, May 1990.

[137] B.G. Mertzios and A.N. Venetsanopoulos, "Fast Implementation of Nonlinear Volterra Digital Filters Via Systolic Arrays," *Communication, Control, and Signal Processing, Proceedings of the 1990 Bilkent International Conference on New Trends in Communication, Control and Signal Processing*, Ankara, Turkey, pp. 1135-1141, July 1990.

[138] B. Picinbono and P. Duvaut, "Geometrical Properties of Optimal Volterra Filters for Signal Detection," *IEEE Transactions on Information Theory*, Vol. 36, No. 5, pp. 1061-1068, September 1990.

[139] K.H. Kim, S.B. Kim, and E.J. Powers, "Fast RLS Algorithms for General Filters," *Signal Processing V: Theories and Applications, Proceedings of EUSIPCO 90, Fifth European Signal Processing Conference*, Vol. 1, Barcelona, Spain, pp. 181-184, September 1990.

[140] D.R. Korrai and D.C. Reddy, "A New Method of Designing Second Order Nonlinear Filters," *Signal Processing V: Theories and Applications, Proceedings of EUSIPCO 90, Fifth European Signal Processing Conference*, Vol. 1, Barcelona, Spain, pp. 521-523, September 1990.

[141] P. Chevalier and B. Picinbono, "Detection with a Second-Order Volterra Array Processor Mismatched to the Fourth-Order Moments of the Noise," *Signal Processing V: Theories and Applications, Proceedings of EUSIPCO 90, Fifth European Signal Processing Conference*, Vol. 1, Barcelona, Spain, pp. 661-664, September 1990.

[142] E.J. Powers, S.W. Nam, and S.B. Kim, "Adaptive Algorithms for the Frequency-Domain Identification of a Second- Order Volterra System with Random Input," *Fifth ASSP Workshop on Spectrum Estimation and Modeling*, IEEE, pp. 25-29, October 1990.

[143] G.B. Giannakis and A.V. Dandawate, "Linear and Nonlinear Adaptive Noise Cancelers," *IEEE*, pp. 1373-1376, 1990.

[144] V.J. Mathews, "Adaptive Polynomial Filters," *IEEE Signal Processing Magazine*, Vol. 8, No. 3, pp. 10-26, July 1991.

[145] M.J. Korenberg and L.D. Paarmann, "Orthogonal Approaches to Time-Series Analysis and System Identification," *IEEE Signal Processing Magazine*, Vol. 8, No. 3, pp. 29-43, July 1991.

[146] M. Morhac, "A Fast Algorithm of Nonlinear Volterra Filtering," *IEEE Transactions on Signal Processing*, Vol. 39, No. 10, pp. 2353-2356, October 1991.

[147] K. Deergha Rao and D.C. Reddy, "New Method of Designing Adaptive Nonlinear Filters," *IEEE Proceedings-F*, Vol. 138, No. 5, pp. 513-519, October 1991.

[148] H.T. Mouftah and C.H. Sauer, "Computer-Aided Modeling, Analysis, and Design of Communication Networks: Introduction and Issue Overview," *IEEE Journal on Selected Areas in Communications*, Vol. 6, No. 1, pp. 126-128, January 1988.

[149] K.S. Shanmugan, "An Update on Software Packages for Simulation of Communication Systems (Links)," *IEEE Journal on Selected Areas in Communications*, Vol. 6, No. 1, pp. 5-12, January 1988.

[150] G. Kompa and F. van Raay, "Automatic Large Signal Measurement System for Nonlinear Device Modeling and Model Verification," *Conference Proceedings of the 19th European Microwave Conference*, London, UK, pp. 587-594, September 1989.

[151] A.J. Poplawski, "Parametric Identification of Nonlinear Dynamic Systems in the Time Domain," *Advances in Modelling & Simulation*, AMSE Press, Vol. 17, No. 3, pp. 1-10, 1989.

[152] R. Jones and D.S. Broomhead, "Phase Spaces from Experimental Time Series," *European Conference on Circuit Theory and Design*, Brighton, UK, pp. 542-546, September 5-8, 1989.

[153] J.W. Bandler, Q.J. Zhang, S. Ye, and S.H. Chen, "Efficient Large-Signal FET Parameter Extraction Using Harmonics," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 37, No. 12, pp. 2099-2108, December 1989.

[154] G.G.E. Gielen, H.C.C. Walscharts, and W.M.C. Sansen, "Analog Circuit Design Optimization Based on Symbolic Simulation and Simulated Annealing," *IEEE Journal of Solid-State Circuits*, Vol. 25, No. 3, pp. 707-713, June 1990.

[155] R.J. Bowman and A.T. Davis, "An Intuitive Behavioral Modeling Language for Mixed Signal ASIC Simulation," *IEEE*, pp. 823-826, 1990.

[156] S.A. Maas, "Analysis and Optimization of Nonlinear Microwave Circuits by Volterra-Series Analysis," *Microwave Journal*, Vol. 33, No. 4, pp. 245-251, April 1990.

[157] S. Boyd, "Multitone Signals with Low Crest Factor," *IEEE Transactions on Circuits and Systems*, Vol. CAS-33, No. 10, pp. 1018-1022, October 1985.

[158] R. Deutsch, "On a Method of Wiener for Noise through Nonlinear Devices," pp. 186-192, 1955.

[159] R.E. Maurer and S. Narayanan, "Noise Loading Analysis of a Third-Order Nonlinear System with Memory," *IEEE Transactions on Communication Technology*, Vol. COM-16, No. 5, pp. 701-712, October 1968.

[160] W. Reiss, "Nonlinear Distortion Analysis of p-i-n Diode Attenuators Using Volterra Series Representation," *IEEE Transactions on Circuits and Systems*, Vol. CAS-31, No. 6, pp. 535-542, June 1984.

[161] Y.J. Liu, I. Oka, and E. Biglieri, "Error Probability for Digital Transmission Over Nonlinear Channels with Application to TCM," *IEEE Transactions on Information Theory*, Vol. 36, No. 5, pp. 1101-1110, September 1990.

[162] T. Larsen, "Theory of Weakly Nonlinear Noisy Systems," *International Journal of Circuit Theory and Applications*, Vol. 19, No. 5, pp. 431-452, September-October 1991.

[163] J.D. Blair, A. Correale, Jr., H.C. Cranford, D.A. Dombrowski, C.K. Erdelyi, C.R. Hoffman, J.L. Lamphere, K.W. Lang, J.K. Lee, J.M. Mullen, V.R. Norman, and S.F. Oakland, "A 16-Mbit/s Adapter Chip for the IBM Token-Ring Local Area Network," *IEEE Journal of Solid-State Circuits*, Vol. 24, No. 6, pp. 1647-1655, December 1989.

[164] L.A. Zadeh, "Frequency Analysis of Variable Networks," *Proceedings of the I.R.E.*, pp. 291-299, March 1950.

[165] T.B.M. Neill, "Improved Method of Analysing Nonlinear Electrical Networks," *Electronic Letters*, Vol. 5, No. 1, pp. 13-15, January 9, 1969.

[166] A. Mircea and H. Sinnreich, "Distortion Noise in Frequency-Dependent Nonlinear Networks," *Proc. IEE*, Vol. 116, No. 10, pp. 1644-1648, October 1969.

[167] B.J. Leon and D.J. Schaefer, "Volterra Series and Picard Iteration for Nonlinear Circuits and Systems," *IEEE Transactions on Circuits and Systems*, Vol. CAS-25, No. 9, pp. 789-793, September 1978.

[168] A. Javed, B.A. Syrett, and P.A. Goud, "Intermodulation Distortion Analysis of Reflection-Type IMPATT Amplifiers Using Volterra Series Representation," *IEEE Transactions on Microwave Theory and Techniques*, Vol. MTT-25, No. 9, pp. 729-734, September 1977.

[169] A. Mesquita, C. Vidallon, and S. Geronimi, "The Time Domain Solution of the Volterra Functional Equation: Application to Nonlinear Circuit Analysis," *Proceedings of the IEEE*, Vol. 67, No. 3, pp. 431-432, March 1979.

[170] L.O. Chua and C.Y. Ng, "Frequency-Domain Analysis of Nonlinear Systems: Formulation of Transfer Functions," *Electronic Circuits and Systems*, Vol. 3, No. 6, pp. 257-269, November 1979.

[171] J.Y. Hong, Y.C. Kim, and E.J. Powers, "On Modeling the Nonlinear Relationship Between Fluctuations with Nonlinear Transfer Functions," *Proceedings of the IEEE*, Vol. 68, No. 8, pp. 1026-1027, August 1980.

[172] A. Borys, "The Relationship Between Different Measures of Nonlinear Distortion in Single-Amplifier Active Filters," *IEEE Transactions on Circuits and Systems*, Vol. CAS-30, No. 11, pp. 842-846, November 1983.

[173] M. Schetzen, "Multilinear Theory of Nonlinear Networks," *Journal of the Franklin Institute*, Vol. 320, No. 5, pp. 221-247, November 1985.

[174] S.P. Gulin, "The Modified Method of Volterra Functional Series," *Radiotekhnika* No. 10, pp. 57-60, 1986.

[175] A. Borys, "The Response of a Nonlinear Circuit to an Excitation in the Form of Fourier Series," *Scientia Electrica*, Vol. 33, No. 3, pp. 3-8, 1987.

[176] K.V. Golovinskii and N.Y. Razumovskii, "An Adaptive Algorithm for Calculations on Nonlinear Devices Based on Volterra-Picard Series," *Radioelectronics and Communications Systems*, Vol. 31, No. 9, pp. 44-47, 1988.

[177] V.M. Bogachev and A.D. Sazonov, "Computer-Oriented Analysis of Periodic Modes in Nonlinear Networks by Means of Spectral Volterra-Picard Series," *Radiotekhnika*, No. 4, pp. 92-94, May 1988.

[178] J. Ladvanszky, "Maximum Power Transfer in Weakly Nonlinear Circuits," *IEEE ISCAS 1988*, pp. 2723-2726, 1988.

[179] S.T. Glad and K. Stahl, "Harmonic Balancing Using a Volterra Input Output Description," *IFAC Nonlinear Control Systems Design*, Capri, Italy, pp. 143-148, June 14-16, 1989.

[180] E. Van den Eijnde and J. Schoukens, "A Recursive Solution for Strongly Nonlinear Systems by Combining Volterra Principles with the Harmonic Balance Technique," *IEEE ISCAS 1989*, pp. 2159-2164, 1989.

[181] M.V. Nazarova and V.E. Porotikov, "Selection of the Optimal Structure of a Discrete Volterra Series for the Signal-Transformation Model in a Weakly Nonlinear System," *Izvestiya VUZ. Radioelektronika*, Vol. 32, No. 7, pp. 66-68, 1989.

[182] A.E. Sowa and J.S. Witkowski, "Response of a Selective H.F. Amplifier to a Pulse Train Input with an Uniform Frequency Spectrum," *1989 International Symposium on Electromagnetic Compatibility Pt. 1, IEEE*, Piscataway, NJ, USA, pp. 257-259, 1989.

[183] V.M. Bogachev and A.D. Sazonov, "Volterra-Picard Spectral Series in Nonlinear-Circuit Theory," *Radioelectron Commun. Syst.*, Vol. 32, No. 5, pp. 25-30, 1989.

[184] G.E. Nasr and B.J. Leon, "Forced Systems with Multiple Steady States," *The Twenty-Second Southeastern Symposium on System Theory, Proc. of the IEEE*, pp. 405-409, March 1990.

[185] D.A. Kabanov and A.N. Zaytsev, "Method of Analysis of Non-Linear Systems on the Basis of Generalization of the Volterra Functional Series Relative to Complex Envelopes," *International Journal of Circuit Theory and Application*, Vol. 19, pp. 9-17, 1991.

[186] H.K. Thapar and B.J. Leon, "Volterra Series Charaterization of Nonlinear Systems," *IEEE Symposium on Circuits and Systems*, pp. 951-954, 1981.

[187] W.X. Zhang and M.R. Raghuveer, "Estimation of Frequency Response and Intermodulation Distortion From Bispectra," *IEEE Workshop on Higher-Order Spectral Analysis*, Piscataway, NJ, pp. 30-35, 1989.

[188] N. Rozario and A. Papoulis, "The Identification of Certain Nonlinear Systems by Only Observing the Output," *IEEE Workshop on Higher-Order Spectral Analysis*, Piscataway, NJ, pp. 78-82, 1989.

[189] E.J. Ewen and D.D. Weiner, "Identification of Weakly Nonlinear Systems Using Input and Output Measurements," *IEEE Transactions on Circuits and Systems*, Vol. CAS-27, No. 12, pp. 1255-1261, December 1980.

[190] S.M. Biyiksiz, "Volterra-Tauberian Estimation for Nonlinear System Identification," *IEEE International Conference on Systems Engineering*, pp. 455-457, 1989.

[191] A. Krzyzak, "On Estimation of a Class of Nonlinear Systems by the Kernel Regression Estimate," *IEEE Transactions on Information Theory*, Vol. 36, No. 1, pp. 141-152, January 1990.

[192] S.W. Nam, S.B. Kim, and E.J. Powers, "On the Identification of a Third-Order Volterra Nonlinear System Using a Frequency-Domain Block RLS Adaptive Algorithm," *IEEE International Conference on Acoustics, Speech and Signal Processing*, Vol. 5, pp. 2407-2410, April 1990.

[193] R. Kannurpatti and G.W. Hart, "Memoryless Nonlinear System Identification With Unknown Model Order," *IEEE Transactions on Information Theory*, Vol. 37, No. 5, pp. 1440-1450, September 1991.

[194] V. Volterra, "Theory of Functionals and of Integral and Integro-Differential Equations," Dover, New York," p. 4, 1959.

[195] D.D. Weiner and J.E. Spina, "Sinusoidal Analysis and Modelling of Weakly Nonlinear Circuits," Van Nostrand Reinhold, New York, 1980.

[196] M. Schetzen, "The Volterra and Wiener Theories of Nonlinear Systems," John Wiley & Sons, New York, 1980.

[197] W. Rugh, "Nonlinear System Theory," The Johns Hopkins University Press, Baltimore, 1981.

[198] J.W. Graham and L. Ehrman, "Nonlinear System Modeling and Analysis with Applications to Communications Receivers," Rome Air Development Center, Rome, N.Y.," June 1973.

[199] R.B. Swerdlow, "Analysis of Intermodulation Noise in Frequency Converters by Volterra Series," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-26, pp. 305-313, April 1978.

[200] M. Lamnabhi, "Functional Analysis of Nonlinear Circuits: A Generating Power Series Approach," *IEE Proc.*, Vol. 133, Pt. H, pp. 375-384, October 1986.

[201] M. Lamnabhi, "A New Symbolic Calculus for the Response of Nonlinear Systems," *Systems & Control Letters*, Vol. 2, pp. 154-161, October 1982.

[202] M. Lamnabhi, "Functional Analysis of Nonlinear Circuits," pp. 1095-1103.

[203] M. Fliess, M. Lamnabhi, and F.Lamnabhi-Lagarrigue, "An Algebraic Approach to Nonlinear Functional Expansions," *IEEE Trans. Circuits and Systems*, Vol. CAS-30, pp. 554-570, August 1983.

[204] T. Brazil, S. El-Rabaie, E. Choo, V. Fusco, and C. Stewart, "Large-Signal FET Simulation using the Time Domain and Harmonic Balance Methods," *IEE Proc.*, Vol. 133, Pt. H, pp. 363-367, October 1986.

[205] D. Hente and R.H. Jansen, "Frequency Domain Continuation Method for the Analysis and Stability Investigation of Nonlinear Microwave Circuits," *IEE Proc.*, Vol. 133, Pt. H, pp. 351-362, October 1986.

[206] W.R. Curtice, "Nonlinear Analysis of GaAs MESFET Amplifiers, Mixers, and Distributed Amplifiers Using the Harmonic Balance Technique," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-35, pp. 441-447, April 1987.

[207] B.R. Epstein, S.M. Perlow, and D.L. Rhodes, "Large-Signal MESFET Characterization Using Harmonic Balance," *IEEE MTT-S Digest*, pp. 1045-1048, 1988.

[208] R. Gilmore, "Design of a Novel FET Frequency Doubler Using a Harmonic Balance Algorithm," *IEEE MTT-S International Microwave Symposium Digest*, pp. 585-588, 1986.

[209] R. Gilmore, "Nonlinear Circuit Design Using the Modified Harmonic Balance Algorithm," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-34, pp. 1294-1307, December 1986.

[210] K.S. Kundert, G.B. Sorkin, and A. Sangiovanni-Vincentelli, "Applying Harmonic Balance to Almost-Periodic Circuits," *IEEE Trans. on Microwave Theory and Tech.*, Vol. 36, pp. 366-377, February 1988.

[211] K.S. Kundert and A. Sangiovanni-Vincentelli, "Simulation of Nonlinear Circuits in the Frequency Domain," *IEEE Journal on Selected Areas in Communication*, Vol. SAC-2, pp. 521-535, January 1984.

[212] M.S. Nakhla and J. Vlach, "A Piecewise Harmonic Balance Technique for Determination of Periodic Response of Nonlinear Systems," *IEEE Trans. Circuits and Systems*, Vol. CAS-23, pp. 85-91, February 1976.

[213] V. Rizzoli, C. Cecchetti, A. Lipparini, and F. Mastri, "General-Purpose Harmonic Balance Analysis of Nonlinear Microwave Circuits Under Multitone Excitation," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-36, pp. 1650-1660, December 1988.

[214] Ulrich L. Rohde, "Harmonic Balance Method Handles Nonlinear Microwave CAD Problems," *Microwave Journal*, pp. 203-210, October 1987.

[215] G.W. Rhyne and M.B. Steer, “Generalized Power Series Analysis of Intermodulation Distortion in a MESFET Amplifier: Simulation and Experiment,” *1987 IEEE MTT-S International Microwave Symposium Digest*, pp. 115-118, December 1987.

[216] G.W. Rhyne and M.B. Steer, “A New Frequency Domain Approach to the Analysis of Nonlinear Microwave Circuits,” *1985 IEEE MTT-S Int. Microwave Symp. Digest*, pp. 401-404, 1985.

[217] L.O. Chua and Y.S. Tang, “Nonlinear Oscillation via Volterra Series,” *IEEE Trans. Circuits and Systems*, Vol. CAS-29, pp. 150-168, March 1986.

[218] T. Endo and L.O. Chua, “Quasi-Periodic Oscillation via Volterra Series,” *IEEE International Symposium on Circuits and Systems*, pp. 57-60, May 1986.

[219] Y.L. Kuo, “Frequency-Domain Analysis of Weakly Nonlinear Networks, “Canned” Volterra Analysis,” *IEEE Circuits and Systems Magazine*, Part 1 — pp. 2-8, August 1977; Part 2 — pp. 2-6, October 1977.

[220] L.O. Chua and C.Y. Ng, “Frequency Domain Analysis of Nonlinear Systems: General Theory,” *Electronic Circuits and Systems*, Vol. 3, No. 4, pp. 165-185, July 1979.

[221] J.W. Graham and L. Ehrman, “Nonlinear System Modeling and Analysis with Applications to Communications Receivers,” Rome Air Development Center, Rome, N.Y.,” June 1973.

[222] V. Krozer, K. Fricke, and H.L. Hartnagel, “A Novel Analytical Approach for the Nonlinear Microwave Circuits and Experimental Characterization of the Nonlinear Behavior of a New MESFET Device Structure,” *IEEE MTT-S International Microwave Symposium Digest*, pp. 351-354, June 1989.

[223] P.J. Lunsford, G.W. Rhyne, and M.B. Steer, “Frequency-Domain Bivariate Generalized Power Series Analysis of Nonlinear Analog Circuits,” *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-38, Vol. 38, No. 6, June 1990.

[224] V. Volterra, “Theory of Functionals and of Integral and Integro-Differential Equations,” Dover, New York,” 1959.

[225] I.W. Sandberg, “Expansions for Nonlinear Systems,” *The Bell System Technical Journ.*, Vol. 61, pp. 159-199, February 1982.

[226] C.L. Law and C.S. Aitchison, “Prediction of Wide-Band Power Performance of MESFET Distributed Amplifiers Using the Volterra Series Representation,” *IEEE Trans. Microwave Theory and Tech.*, Vol. MTT-34, pp. 1308-1317, December 1986.

[227] R.S. Tucker and C. Rauscher, “Modelling the 3rd-Order Intermodulation-Distortion Properties of a GaAs F.E.T.,” *Electron. Lett.*, Vol. 13, pp. 508-509, August 18, 1977.

[228] R.S. Tucker, "Third-Order Intermodulation Distortion and Gain Compression in GaAs FET's," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-27, pp. 400-408, May 1979.

[229] D.D. Weiner and J.E. Spina, "Sinusoidal Analysis and Modelling of Weakly Nonlinear Circuits," Van Nostrand Reinhold, New York," 1980.

[230] G.W. Rhyne and M.B. Steer, "A New Frequency Domain Approach to the Analysis of Nonlinear Microwave Circuits," *1985 IEEE MTT-S Int. Microwave Symp. Digest*, pp. 401-404, 1985.

[231] T.W. Crowe, R.J. Mattauch, and S.A. Maas , "A GaAs MESFET Mixer with Very Low Intermodulation ," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-35, pp. 425-429, April 1987.

[232] J.H. Haywood, Y.L. Chow, and H. Schutzman, "Intermodulation Distortion Analysis Using a Frequency-Domain Harmonic Balance Technique," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-36, pp. 1251-1257, August 1988.

[233] Y.D. Sverkunov, "Determination of Multidimensional Transfer Functions of a Nonlinear System," *Telecom. and Radio Engrg.*, Vol. 31/32, pp. 108-110, September 1977.

[234] R.A. Nobakht, P.W. Pate, and S.H. Ardalan, "CAPSIM: A Graphical Simulation Tool for Communication Systems," *Globecom '88*, pp. 1692-1696, 1988.

[235] B. Chanane and S.P. Banks, "Rational Expansion of Non-Linear Input-Output Maps," *Int. J. Control*, Vol. 51, No. 6, pp. 1241-1250, 1990.

[236] E. Van den Eijnde and J. Schoukens, "Steady-State Analysis of a Periodically Excited Nonlinear System," *IEEE Transactions on Circuits and Systems*, Vol. 37, No. 2, pp. 232-242, February 1990.

Appendix A
Algorithm for Bivariate GPSA Indexing

In order to calculate T in (3.22) the set of conditions on the second summation give restrictions that are less than obvious to implement:

$$\sigma = \sum_{i=0}^{N} p_i + r_i \tag{A.1}$$

$$\rho = \sum_{i=0}^{N} q_i + s_i \tag{A.2}$$

$$\text{for } 1 \leq i \leq N : p_i + q_i - r_i - s_i = |n_i| \tag{A.3}$$

$$\sigma + \rho = n + 2\alpha \tag{A.4}$$

At this point we know the value of n, α, and the IPD vector $\bar{n}$ which consists of integers such that $\sum_{k=1}^{N} |n_k| = n$. We also know that σ, ρ, p_i, q_i, r_i, and s_i are all non-negative integers. From (A.4) it follows that $0 \leq \sigma \leq (n + 2\alpha)$ and $0 \leq \rho \leq (n + 2\alpha)$ since both σ and ρ are nonnegative. We can therefore step through all possible σ and ρ by the algorithm given in figure A.1. Thus the only remaining task is to identify all sets of vectors for a given $\bar{n}$, σ, and ρ such that that all equations are satisfied.

In order to facilitate further explanation, a specialized ripple algorithm is defined. The purpose of the algorithm is to sequence through all possible sets of vectors of length N such that the sum of the components is a given constant S, i.e. $\sum_{i=1}^{N} v_i = S$. We further restrict all of the components of the vectors to be non-negative integers, i.e. $v_i \geq 0$. Since all the components of $\bar{v}$ are non-negative, the sum is equivalent to the L_1 norm, $||\bar{v}||_1 = \sum_i |v_i|$. It follows that $\max_i v_i \leq S$. Our algorithm first initializes $v_1 = S$ and all other $v_i = 0$. Since $\sum_{i=1}^{N} v_i = S$, then this initial vector is the only vector with $v_1 = S$. We then find all the possible vectors with $v_1 = S - 1$, then all possible vectors with $v_1 = S - 2$, and so on. The key to this algorithm is in recognizing that the problem of finding all valid vectors with $v_1 = S - j$ is equivalent to the problem of finding all possible vectors of length $N - 1$ with $||\bar{v}||_1 = j$. Thus our algorithm is recursive. Let us define the number of vectors to be $F_N(S)$. From

- initialize $\sigma = 0$ and $\rho = n + 2\alpha$.
- Increment σ and decrement ρ until $\sigma = n + 2\alpha$ and $\rho = 0$.

Figure A.1: Algorithm for stepping through ρ and σ

- Let $v_1 = S, S-1, S-2 \ldots 0$.
- for each value of v_1, define a new vector $\bar{v}'$ of length $N-1$ defined by $v'_i = v_{i+1}$, and a new constant value S' defined by $S' = S - v_1$.
    - let $v'_1 = S', S'-1, S'-2 \ldots 0$.
    - for each value of v'_1, define a new vector $\bar{v}''$ of length $N-2$ defined by $v''_i = v'_{i+1}$, and a new constant value S'' defined by $S'' = S' - v'_1$.
    - Continue the process until the vector defined is of length 1, and thus the vector has only one possible value for a given L_1 norm.

Figure A.2: Recursive ripple algorithm

our recursive algorithm we know that

$$F_N(S) = \sum_{K=0}^{S} F_{(N-1)}(K) \tag{A.5}$$

and

$$F_1(S) = 1 \tag{A.6}$$

Thus

$$F_N(S) = \sum_{K_N=0}^{S} \sum_{K_{(N-1)}=0}^{K_N} \sum_{K_{(N-2)}=0}^{K_{(N-1)}} \sum_{K_{(N-3)}=0}^{K_{(N-2)}} \cdots \sum_{K_3=0}^{K_4} \sum_{K_2=0}^{K_3} 1 \tag{A.7}$$

It follows that the number of vector combinations $F_N(S)$ is of order $\frac{S^{N-1}}{(N-1)!}$. In order to better illustrate this sequence, let us look at the example of $N = 4$ and $S = 3$. Here our algorithm produces the vectors shown in figure A.3. In order to more easily implement this ripple algorithm, we will use the non-recursive procedure shown in figure A.4 that takes a vector of length N and finds the next vector in the sequence. If the input vector is the last in the sequence, then the overflow flag is set and the first vector of the sequence is returned. The following algorithm will take any of the above vectors and produce the next vector.

The problem now is to find all sets of vectors $\bar{v}$, $\bar{q}$, $\bar{r}$, and $\bar{s}$ for a given ρ, σ, and $\bar{n}$. We now define a new vector $\bar{u}$ of length $2N$ that consists of the concatination of $\bar{p}$ and $\bar{r}$, i.e. $u_i = p_i$ for $i \leq N$ and $u_i = r_{(i-N)}$ for $i > N$. From (A.1) and (A.2) we know that the L_1 norm of u is equal to ρ, thus we can step through all possible values of $\bar{p}$ and $\bar{r}$ using our ripple algorithm on $\bar{u}$ with $||\bar{u}||_1 = \rho$.

Our only remaining task is to find all valid values of $\bar{q}$ and $\bar{s}$ for a given $\bar{n}$, ρ, σ, $\bar{p}$, and $\bar{r}$. By solving (A.3) for q_i and substituting into (A.2) we get:

$$\rho = \sum_{i=1}^{N} |n_i| + r_i + 2s_i - p_i \tag{A.8}$$

$$(3,0,0,0)$$
$$(2,1,0,0)$$
$$(2,0,1,0)$$
$$(2,0,0,1)$$
$$(1,2,0,0)$$
$$(1,1,1,0)$$
$$(1,1,0,1)$$
$$(1,0,2,0)$$
$$(1,0,1,1)$$
$$(1,0,0,2)$$
$$(0,3,0,0)$$
$$(0,2,1,0)$$
$$(0,2,0,1)$$
$$(0,1,2,0)$$
$$(0,1,1,1)$$
$$(0,1,0,2)$$
$$(0,0,3,0)$$
$$(0,0,2,1)$$
$$(0,0,1,2)$$
$$(0,0,0,3)$$

Figure A.3: Example of ripple sequence for $S = 3$ and $N = 4$

```
• Calculate $L_1$ norm $= S$.
• If $S = 0$ then
    – set the overflow flag
    – end
• Find $i_{max} = \max_{v_i \neq 0} i$.
• If $i_{max} = N$ then
    – Find $i_{next} = \max_{v_i \neq 0} i < N$
    – If $i_{next}$ does not exist
        * $v_1 = v_N$
        * If $N \neq 1$ then $v_N = 0$
        * set overflow flag
        * end
    – If $i_{next} = N - 1$ then
        * $v_{N-1} = v_{N-1} - 1$
        * $v_N = v_N + 1$
        * end
    – If $i_{next} \neq N - 1$ then
        * $v_{i_{next}} = v_{i_{next}} - 1$
        * $v_{i_{next}+1} = 1 + v_N$
        * $v_N = 0$
        * end
• If $i_{max} \neq N$ then
    – $v_{i_{max}} = v_{i_{max}} + 1$
    – $v_{i_{max}+1} = 1$
    – end
```

Figure A.4: Nonrecursive ripple algorithm

Rearranging for $||\bar{s}||_1$ yields

$$\sum_{i=1}^{N} s_i = \frac{\rho + \sum_{i=1}^{N} p_i - r_i - |n_i|}{2} \tag{A.9}$$

Thus using (A.9) we can determine $||\bar{s}||_1$ and ripple through all possible combinations of $\bar{s}$. If the right side of (A.9) yields a negative or a noninteger value, then we know that no solutions exist for the current set of $\bar{n}$, ρ, σ, $\bar{p}$, and $\bar{r}$.

Rearranging (A.3) for q_i yields

$$q_i = |n_i| + r_i + s_i - p_i \tag{A.10}$$

Thus for a given set of $\bar{n}$, ρ, σ, $\bar{p}$, $\bar{r}$, and $\bar{s}$, we can calculate $\bar{q}$. If any q_i is negative, then no solution exists for the current set of $\bar{n}$, ρ, σ, $\bar{p}$, $\bar{r}$, and $\bar{s}$.

The alternate way of finding all of the possible values $\bar{p}$, $\bar{r}$,$\bar{q}$, and $\bar{s}$ that fit our criterion is to go through all possible combinations with components less than or equal to $n + 2\alpha$. The number of possibilities is $(n + 2\alpha + 1)^N$.

Appendix B
Properties of Volterra Nonlinear Transfer Functions

B.1 Introduction

In order to gain a better understanding of Volterra nonlinear transfer functions, we will look at the basic formula and the implications of its form. Specifically, we will look at the basic forumla, the idea of kernel symmetry, and the relationship of transfer functions whose arguments are related.

B.2 Basic Formula

The output $y(t)$ of a system described by Volterra kernels $h_n(\tau_1, \tau_2 \ldots \tau_n)$ stimulated by an input $x(t)$ is given by

$$y(t) = \sum_{n=0}^{\infty} y_n(t) \tag{B.11}$$

where

$$\begin{aligned} y_n(t) \quad = \quad & \int_{\tau_n=-\infty}^{+\infty} \int_{\tau_{n-1}=-\infty}^{+\infty} \cdots \\ & \int_{\tau_1=-\infty}^{+\infty} h_n(\tau_1, \tau_2 \ldots \tau_n) \prod_{i=1}^{n} x(t-\tau_i) \\ & d\tau_1 d\tau_2 \ldots d\tau_n \end{aligned} \tag{B.12}$$

$$y_0(t) \quad = \quad h_0 \tag{B.13}$$

The nth order nonlinear transfer function $H_n(f_1, f_2 \ldots f_n)$ is defined to be the multi–dimensional Fourier transform of the nth order kernel $h_n(\tau_1, \tau_2 \ldots \tau_n)$.

$$\begin{aligned} H_n(f_1, f_2 \ldots f_n) \quad = \quad & \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} h_n(\tau_1, \tau_2 \ldots \tau_n) \\ & e^{-j2\pi(f_1\tau_1 + f_2\tau_2 + \ldots f_n\tau_n)} d\tau_1 d\tau_2 \ldots d\tau_n \end{aligned} \tag{B.14}$$

$$H_0 = h_0 \tag{B.15}$$

and

$$\begin{aligned} h_n(\tau_1, \tau_2 \ldots \tau_n) \quad = \quad & \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} H_n(f_1, f_2 \ldots f_n) \\ & e^{+j2\pi(f_1\tau_1 + f_1\tau_2 + \ldots f_n\tau_n)} df_1 df_2 \ldots df_n \end{aligned} \tag{B.16}$$

B.3 Symmetry of Kernels and Transfer Functions

The Volterra kernels, h_n are used to completely characterize the system. But there can be more than one set of kernels that are equivalent and predict the exact same system response. In order to simplify the output calculations, and to specify a unique set of kernels, we can require that all of the kernels be symmetric with respect to their input arguments.

From (B.12)

$$\begin{aligned} y_n(t) &= \int_{\tau_n=-\infty}^{+\infty} \int_{\tau_{n-1}=-\infty}^{+\infty} \cdots \\ &\quad \int_{\tau_2=-\infty}^{+\infty} \int_{\tau_1=-\infty}^{+\infty} h_n(\tau_1, \tau_2 \ldots \tau_n) x(t-\tau_1) x(t-\tau_2) \prod_{i=3}^{n} x(t-\tau_i) \\ &\quad d\tau_1 d\tau_2 \ldots d\tau_n \end{aligned} \tag{B.17}$$

Or by interchanging the dummy integration variables τ_1 and τ_2, i.e. $\tau_1 \to \tau_2$ and $\tau_2 \to \tau_3$

$$\begin{aligned} y_n(t) &= \int_{\tau_n=-\infty}^{+\infty} \int_{\tau_{n-1}=-\infty}^{+\infty} \cdots \\ &\quad \int_{\tau_1=-\infty}^{+\infty} \int_{\tau_2=-\infty}^{+\infty} h_n(\tau_2, \tau_1 \ldots \tau_n) x(t-\tau_2) x(t-\tau_1) \prod_{i=3}^{n} x(t-\tau_i) \\ &\quad d\tau_2 d\tau_1 \ldots d\tau_n \end{aligned} \tag{B.18}$$

and changing the order of integration

$$\begin{aligned} y_n(t) &= \int_{\tau_n=-\infty}^{+\infty} \int_{\tau_{n-1}=-\infty}^{+\infty} \cdots \\ &\quad \int_{\tau_2=-\infty}^{+\infty} \int_{\tau_1=-\infty}^{+\infty} h_n(\tau_2, \tau_1 \ldots \tau_n) x(t-\tau_2) x(t-\tau_1) \prod_{i=3}^{n} x(t-\tau_i) \\ &\quad d\tau_1 d\tau_2 \ldots d\tau_n \end{aligned} \tag{B.19}$$

Or

$$\begin{aligned} y_n(t) &= \int_{\tau_n=-\infty}^{+\infty} \int_{\tau_{n-1}=-\infty}^{+\infty} \cdots \\ &\quad \int_{\tau_2=-\infty}^{+\infty} \int_{\tau_1=-\infty}^{+\infty} h_n(\tau_2, \tau_1 \ldots \tau_n) \prod_{i=1}^{n} x(t-\tau_i) \\ &\quad d\tau_1 d\tau_2 \ldots d\tau_n \end{aligned} \tag{B.20}$$

Therefore we see that changing the order of the arguments of h_n yields the same output. Thus we can restrict the kernels to be symmetric, ($h_n(\tau_1 \ldots \tau_i, \tau_{i+1} \ldots \tau_j, \tau_{j+1} \ldots \tau_n)$ $= h_n(\tau_1 \ldots \tau_i, \tau_{j+1} \ldots \tau_j, \tau_{i+1} \ldots \tau_n)$) and thus uniquely specify the kernels. Note that symmetry implies that any two arguments of the function can be interchanged and the resulting value will be the same.

From (B.14) we can see that if we restrict h_n to be symmetric, then the nonlinear transfer functions $H_n(f_1, f_2 \ldots f_n)$ are also symmetric. This property simplifies the calculation of determining the steady–state frequency domain response of the system described by a set of nonlinear transfer functions.

B.4 Negative Argument of a Transfer Function

In extracting symmetric nonlinear transfer functions, it is useful to understand the relationship between the value of $H_n(f_1, f_2 \ldots f_n)$ and $H_n(-f_1, -f_2 \ldots - f_n)$. From (B.14)

$$\begin{aligned} H_n(-f_1, -f_2 \ldots - f_n) &= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} h_n(\tau_1, \tau_2 \ldots \tau_n) \\ &\quad e^{-j2\pi(-f_1\tau_1 - f_2\tau_2 \ldots - f_n\tau_n)} d\tau_1 d\tau_2 \ldots d\tau_n \qquad \text{(B.21)} \end{aligned}$$

Multiplying out the -1 in the exponent yields

$$\begin{aligned} H_n(-f_1, -f_2 \ldots - f_n) &= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} h_n(\tau_1, \tau_2 \ldots \tau_n) \\ &\quad e^{+j2\pi(f_1\tau_1 + f_2\tau_2 + \ldots f_n\tau_n)} d\tau_1 d\tau_2 \ldots d\tau_n \qquad \text{(B.22)} \end{aligned}$$

Taking the complex conjugate twice of the right hand side results in

$$\begin{aligned} H_n(-f_1, -f_2 \ldots - f_n) &= \left(\left[\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} h_n(\tau_1, \tau_2 \ldots \tau_n)\right.\right. \\ &\quad \left.\left. e^{+j2\pi(f_1\tau_1 + f_2\tau_2 + \ldots f_n\tau_n)} d\tau_1 d\tau_2 \ldots d\tau_n\right]^*\right)^* \qquad \text{(B.23)} \end{aligned}$$

Since h_n is a real function, and f and τ are real variables, we can take the complex conjugate of the integrand.

$$\begin{aligned} H_n(-f_1, -f_2 \ldots - f_n) &= \left(\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} h_n(\tau_1, \tau_2 \ldots \tau_n)\right. \\ &\quad \left. e^{-j2\pi(f_1\tau_1 + f_2\tau_2 + \ldots f_n\tau_n)} d\tau_1 d\tau_2 \ldots d\tau_n\right)^* \qquad \text{(B.24)} \end{aligned}$$

By using (B.14) we show

$$\begin{aligned} H_n(-f_1, -f_2 \ldots - f_n) &= (H_n(f_1, f_2 \ldots f_n))^* \\ &= H_n^*(f_1, f_2 \ldots f_n) \qquad \text{(B.25)} \end{aligned}$$

Since the value of a transfer function with all of the arguments multiplied by -1 is just the complex conjugate of the transfer function at the original point, we can reduce the number of transfer function points that need to be stored in order to characterize a system. For our case, we choose to store only those transfer function points where the sum of the frequencies is non–negative.

B.5 Balanced Transfer Function Arguments that Add to Zero

We know that the transfer functions are symmetric, and also that the value of the arguments to a transfer function multiplied by -1 is just the complex conjugate of the original transfer function. Using these two properties, we can show that the value of any transfer function whose arguments are a symmetric DC FIPD is a real number. A balanced argument to a transfer function is one where every frequency value has a corresponding negative pair, e.g. $H(f_a, f_b \ldots - f_b, -f_a)$. For an odd order balanced argument, there of course will be a single frequency input that does not have a matching negative pair. For a DC FIPD, the values of the frequencies sum to zero, thus all even balanced symmetric FIPD's are DC FIPD's. An odd symmetric FIPD is DC if and only if the frequency with the unmatching pair is equal to zero. Thus for a symmetric DC FIPD as the input to a transfer function

$$\begin{aligned} H(f_a, f_b \ldots - f_b, -f_a) &= H^*(-f_a, -f_b \ldots f_b, f_a) \\ & H^*(f_a, f_b \ldots - f_b, -f_a) \end{aligned} \tag{B.26}$$

Therefore for any balaned DC FIPD

$$\operatorname{Im}\{H(f_a, f_b \ldots - f_b, -f_a)\} = 0 \tag{B.27}$$

B.6 Conclusion

We have presented the basic equations of Volterra series and shown several properties that can be derived from its form. Without loss of generality, we can restrict all kernels, and thus all Volterra nonlinear transfer functions, to be symmetric with respect to their arguments. This simplifies some of the output calculations and guarantees a unique description of a system. We also have looked at properties relating to the signs ($+$ or $-$) of the nonlinear transfer functions. These properties help us in developing a process to characterize an unknown system by extracting their nonlinear transfer functions.

```
printlevel:=-1;
Digits:=20;
rIF:=1000;
i01:=0.00000000100;
i02:=0.00000000101;
i03:=0.00000000103;
i04:=0.00000000106;
vt:=(0.00008617349*(25+273.15));
e1:= i1 = i01*(exp(v1/vt)-1);
e2:= i2 = i02*(exp(v2/vt)-1);
e3:= i3 = i03*(exp(v3/vt)-1);
e4:= i4 = i04*(exp(v4/vt)-1);
e5:= v1 = (1/2)*vLO - (1/2)*vRF - vIF ;
e6:= v2 = (1/2)*vLO + (1/2)*vRF + vIF ;
e7:= v3 = (1/2)*vRF - (1/2)*vLO - vIF ;
e8:= v4 = vIF - (1/2)*vLO - (1/2)*vRF ;
e9:= iIF = vIF/rIF ;
f1:= i1 - i2 + i3 - i4 - iIF ;
f2:= subs(e5,e6,e7,e8,subs(e1,e2,e3,e4,e9,f1));
vIF1:= taylor(f2,vIF,4);
save rIF,i01,i02,i03,i04,vt,vIF1,‘VIF‘;
quit;
```

Figure C.5: MAPLE code for creating taylor series to solve KCL and KVL of a ring diode mixer.

Appendix C
MAPLE Code for Ring Diode Mixer

C.1 Formulation

The KVL and LCL equations for the circuit in figure 5.1 are given by (5.1)–(5.6). In order to solve these equations for V_{IF} we use the computer–algebra program MAPLE [115]. The MAPLE code given in figure C.5 defines the relationships given in (5.1)–(5.6) and then tells MAPLE to solve for function f2. The relationships of (5.1)–(5.6) are met when f2=0. A taylor series of f2 is then taken with respect to the MAPLE variable vIF (V_{IF} in (5.2)–(5.6)). This taylor series is saved in the file VIF. The resulting file VIF is shown in figure C.6 The file VIF is then copied to VIFt and the "O(vIF**4)" is removed from the last line. Thus when the code in figure C.7 reads the file VIFt, the higher order terms (4th order and higher) are

```
rIF := 1000;
i01 := Float(100,-11);
i02 := Float(101,-11);
i03 := Float(103,-11);
i04 := Float(106,-11);
vt := Float(256926260435,-13);
vIF1 := (Float(100,-11)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF
)+Float(4,-11)+Float(-101,-11)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF)+Float(-106,-11)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(103,-11)*exp(19.460836706744324355*vRF-19.
460836706744324355*vLO))+(Float(-41256973818297967632,-27)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)+Float(-38921673413488648709,-
27)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)+Float(-
10000000000000000000,-22)+Float(-39310890147623535196,-27)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)+Float(-40089323615893308170,-
27)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO))*vIF+(Float(
75744833065313455975,-26)*exp(19.460836706744324355*vLO-19.460836706744324355*
vRF)+Float(78017178057272859654,-26)*exp(19.460836706744324355*vRF-19.
460836706744324355*vLO)+Float(-80289523049232263334,-26)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)+Float(-76502281395966590535,-
26)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF))*vIF**2+(Float(-
10121863749957223487,-24)*exp(19.460836706744324355*vRF-19.460836706744324355*
vLO)+Float(-10416675315489958152,-24)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(-98270521844244888226,-25)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)+Float(-99253227062687337108,-
25)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF))*vIF**3+O(vIF**4);
```

Figure C.6: MAPLE output of taylor series expansion.

```
printlevel:=-1;
Digits:=20;
gc(1000);
maxm:=1;
maxn:=1;
read `VIFt`;
VIF:= solve(vIF1,vIF)[1];
save rIF,i01,i02,i03,i04,vt,VIF,`VIFclosed`;
quit;
```

Figure C.7: MAPLE code to create closed form equation of V_{IF}.

effectively truncated.

C.2 Solution

Since the MAPLE equation vIF1 is now a third order polynomial, it can be solved in closed form for the MAPLE variable vIF. This is easily accomplished by line 7 in figure C.7 and the result is saved in the file VIFclosed shown here for completeness. This file can be read into MAPLE and any values of vIF can be calculated for any given vLO and vRF. These calculated points are then used to fit a bivariate power series of the form given in (5.9).

```
--------------  begin file VIFclosed ----------------------------------------
rIF := 1000;
i01 := Float(100,-11);
i02 := Float(101,-11);
i03 := Float(103,-11);
i04 := Float(106,-11);
vt := Float(256926260435,-13);
VIF := (.16666666666666666667*(Float(75744833065313455978,-26)*exp(19.
46083670674432435 5*vLO-19.460836706744324355*vRF)+Float(78017178057272859655,-
26)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)+Float(-
80289523049232263339,-26)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)+Float(-76502281395966590536,-26)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF))/(Float(-10121863749957223487,-24)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(-10416675315489958152,-
24)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)+Float(-
98270521844244888230,-25)*exp(19.460836706744324355*vLO-19.460836706744324355*
vRF)+Float(-99253227062687337108,-25)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF))**2*(Float(-41256973818297967632,-27)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)+Float(-38921673413488648709,-
27)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)+Float(-
10000000000000000000,-22)+Float(-39310890147623535196,-27)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)+Float(-40089323615893308170,-
27)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO))-.
50000000000000000000*(Float(10000000000000000000,-28)*exp(19.460836706744324355
*vLO-19.460836706744324355*vRF)+Float(40000000000000000000,-30)+Float(-
10100000000000000000,-28)*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)+Float(-10600000000000000000,-28)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(10300000000000000000,-28)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO))/(Float(-10121863749957223487
,-24)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)+Float(-
10416675315489958152,-24)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)+Float(-98270521844244888230,-25)*exp(19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(-99253227062687337108,-25)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF))+Float(-37037037037037037037,
-21)*(Float(75744833065313455978,-26)*exp(19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(78017178057272859655,-26)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(-80289523049232263339,-
26)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)+Float(-
76502281395966590536,-26)*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF))**3/(Float(-10121863749957223487,-24)*exp(19.460836706744324355*vRF-19.
460836706744324355*vLO)+Float(-10416675315489958152,-24)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)+Float(-98270521844244888230,-
25)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)+Float(-
99253227062687337108,-25)*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF))**3+Float(11941715590486027987,-190)*(Float(52550011397152175827,503)*exp(
-19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355
*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF-19.
460836706744324355*vLO)+Float(26574461505079066841,498)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF)+Float(26080577645031842674,498)*exp(-19.
```

```
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)*exp(19.460836706744324355*vLO+19.460836706744324355
*vRF)**2+Float(11404101640068168948,499)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF)**2*exp(19.460836706744324355*vRF-19.460836706744324355*
vLO)**2+Float(80421882583752094722,497)*exp(19.460836706744324355*vLO-19.
460836706744324355*vRF)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF
)**3+Float(46480237387185462213,499)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO
)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)+Float(36866910405991707208,
507)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)+Float(95755765911700743143,
497)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)**3*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(82834539061264657556,
497)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF)**3*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(12421550871993125888,
498)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)**2+Float(85247195538777220412
,497)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)**3+Float(13238403108869299838
,511)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)+Float(
13370787139957992836,511)*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)+Float(14032707295401457828,511)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(13635555202135378832,511)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(90412328280280632285,
497)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)**3+Float(80398387520991106075
,497)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**3*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(22143886679744017374,
499)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)*exp(19.460836706744324355*vLO
+19.460836706744324355*vRF)**2+Float(25091532761426114289,498)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)**2*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF)+Float(51529652910629730461,503)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO
-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO+19.460836706744324355
*vRF)+Float(22563222032614302045,499)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF
)**2*exp(19.460836706744324355*vLO+19.460836706744324355*vRF)+Float(
53075542497948622374,503)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)+Float(50071237274644997721,
503)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)+Float(25320949169933827841,498)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO
-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO+19.460836706744324355
*vRF)**2+Float(13420103060064928758,498)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)**2*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)**2+Float(23937322254400513039,499)*exp(-19.460836706744324355*vLO-19.
```

```
460836706744324355*vRF)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO
)**2*exp(19.460836706744324355*vLO+19.460836706744324355*vRF)+Float(
86147595814229659060,497)*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)**3+Float(
25509714270462221276,503)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**2+Float(
85294649321019464429,497)*exp(19.460836706744324355*vLO-19.460836706744324355*
vRF)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)**3+Float(
27371695350231438844,498)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)**2*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)+Float(25566687231675171726,
498)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)*exp(19.460836706744324355*vLO
-19.460836706744324355*vRF)**2+Float(26333687848625426878,498)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)**2*exp(19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(24390619619018113835,499)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vRF-19.460836706744324355*vLO)*exp(19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(12561169103794328625,499)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vRF-19.460836706744324355*vLO)**2+Float(24360711418860305134,498)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vRF-19.460836706744324355*vLO)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF)+Float(-79723384515248745789,496)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vLO+19.460836706744324355*vRF)+Float(72358548768884447996,496)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vRF-19.460836706744324355*vLO)+Float(37972917718171458423,507)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)+Float(28961044932830794533,507)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO
+19.460836706744324355*vRF)+Float(35127905198161909698,507)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO
+19.460836706744324355*vRF)+Float(27862690703834496137,507)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)+Float(24549504452705673474,503)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO
+19.460836706744324355*vRF)**2+Float(36181742354106766988,507)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)+Float(27063255869533370550,503)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)**2+Float(15197380014257743666,507)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**2+Float(74529305231951860686
,496)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)**2+Float(13525578011570143756
,507)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**2+Float(
13797478953848633244,507)*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)**2+Float(-75962847509810592238,496)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF
)**2+Float(27851522300903732533,503)*exp(-19.460836706744324355*vLO-19.
```

```
460836706744324355*vRF)**2*exp(19.460836706744324355*vRF-19.460836706744324355*
vLO)+Float(25786687196442173826,503)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO
)**2+Float(24306425861478154234,503)*exp(19.460836706744324355*vLO-19.
460836706744324355*vRF)**2*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)+Float(14349285712474765510,507)*exp(19.460836706744324355*vRF-19.
460836706744324355*vLO)**2+Float(23417006074072360633,496)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)**3+Float(10749459553273794842
,499)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)**2+Float(25588394796303774045
,496)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)**3+Float(
27040312913498769450,503)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)**2*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)+Float(-
27890028906325338380,496)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)**3+Float(-24126564775127920530,496)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF)**3+Float(21963372200162512090,497)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)**4+Float(78837253782719434090
,497)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**3*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)+Float(25285989586286843678,
503)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF)**2*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(82740088128398613999,
497)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)**3+Float(19514171728395899533
,497)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**4+Float(
20306525352397403919,497)*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)**4+Float(24636192200583200914,497)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)**4+Float(11840106611173841668,499)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vLO-19.460836706744324355*vRF)**2+Float(93896430651279369523,497)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**3*exp(19.460836706744324355*
vLO+19.460836706744324355*vRF)+Float(92966763021068682671,497)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**3*exp(19.460836706744324355*
vLO-19.460836706744324355*vRF))**(1/2)/(Float(85898953831112912821,26)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(88400865107747269507,26
)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)+Float(
83397042554478556142,26)*exp(19.460836706744324355*vLO-19.460836706744324355*
vRF)+Float(84231012980023341699,26)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF))**2)**(1/3)+(.16666666666666666667*(Float(
75744833065313455978,-26)*exp(19.460836706744324355*vLO-19.460836706744324355*
vRF)+Float(78017178057272859655,-26)*exp(19.460836706744324355*vRF-19.
460836706744324355*vLO)+Float(-80289523049232263339,-26)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)+Float(-76502281395966590536,-
26)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF))/(Float(-
10121863749957223487,-24)*exp(19.460836706744324355*vRF-19.460836706744324355*
vLO)+Float(-10416675315489958152,-24)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(-98270521844244888230,-25)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)+Float(-99253227062687337108,-
25)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF))**2*(Float(-
41256973818297967632,-27)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)+Float(-38921673413488648709,-27)*exp(19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(-10000000000000000000,-22)+Float(-
```

```
393108901476235351 96,-27)*exp(19.460836706744324355*vLO+19.460836706744324355*
```

```
13370787139957992836,511)*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)+Float(14032707295401457828,511)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(13635555202135378832,511)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(90412328280280632285,
497)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)**3+Float(80398387520991106075
,497)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**3*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(22143886679744017374,
499)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)*exp(19.460836706744324355*vLO
+19.460836706744324355*vRF)**2+Float(25091532761426114289,498)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)**2*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF)+Float(51529652910629730461,503)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO
-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO+19.460836706744324355
*vRF)+Float(22563222032614302045,499)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF
)**2*exp(19.460836706744324355*vLO+19.460836706744324355*vRF)+Float(
53075542497948622374,503)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)+Float(50071237274644997721,
503)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)+Float(25320949169933827841,498)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO
-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO+19.460836706744324355
*vRF)**2+Float(13420103060064928758,498)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)**2*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)**2+Float(23937322254400513039,499)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO
)**2*exp(19.460836706744324355*vLO+19.460836706744324355*vRF)+Float(
86147595814229659060,497)*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)**3+Float(
25509714270462221276,503)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**2+Float(
85294649321019464429,497)*exp(19.460836706744324355*vLO-19.460836706744324355*
vRF)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)**3+Float(
27371695350231438844,498)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)**2*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)+Float(25566687231675171726,
498)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)*exp(19.460836706744324355*vLO
-19.460836706744324355*vRF)**2+Float(26333687848625426878,498)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)**2*exp(19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(24390619619018113835,499)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vRF-19.460836706744324355*vLO)*exp(19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(12561169103794328625,499)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
```

```
vRF-19.460836706744324355*vLO)**2+Float(24360711418860305134,498)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vRF-19.460836706744324355*vLO)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF)+Float(-79723384515248745789,496)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vLO+19.460836706744324355*vRF)+Float(72358548768884447996,496)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vRF-19.460836706744324355*vLO)+Float(37972917718171458423,507)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)+Float(28961044932830794533,507)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO
+19.460836706744324355*vRF)+Float(35127905198161909698,507)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO
+19.460836706744324355*vRF)+Float(27862690703834496137,507)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)+Float(24549504452705673474,503)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vLO
+19.460836706744324355*vRF)**2+Float(36181742354106766988,507)*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)+Float(27063255869533370550,503)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.460836706744324355*vRF
-19.460836706744324355*vLO)**2+Float(15197380014257743666,507)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**2+Float(74529305231951860686
,496)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)**2+Float(13525578011570143756
,507)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**2+Float(
13797478953848633244,507)*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)**2+Float(-75962847509810592238,496)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF
)**2+Float(27851522300903732533,503)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)**2*exp(19.460836706744324355*vRF-19.460836706744324355*
vLO)+Float(25786687196442173826,503)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO
)**2+Float(24306425861478154234,503)*exp(19.460836706744324355*vLO-19.
460836706744324355*vRF)**2*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)+Float(14349285712474765510,507)*exp(19.460836706744324355*vRF-19.
460836706744324355*vLO)**2+Float(23417006074072360633,496)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)**3+Float(10749459553273794842
,499)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)**2+Float(25588394796303774045
,496)*exp(19.460836706744324355*vRF-19.460836706744324355*vLO)**3+Float(
27040312913498769450,503)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)**2*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)+Float(-
27890028906325338380,496)*exp(-19.460836706744324355*vLO-19.460836706744324355*
vRF)**3+Float(-24126564775127920530,496)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF)**3+Float(21963372200162512090,497)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)**4+Float(78837253782719434090
,497)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**3*exp(19.
460836706744324355*vLO+19.460836706744324355*vRF)+Float(25285989586286843678,
503)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF)**2*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(82740088128398613999,
497)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)**3+Float(19514171728395899533
,497)*exp(19.460836706744324355*vLO-19.460836706744324355*vRF)**4+Float(
20306525352397403919,497)*exp(19.460836706744324355*vLO+19.460836706744324355*
vRF)**4+Float(24636192200583200914,497)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)**4+Float(11840106611173841668,499)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**2*exp(19.460836706744324355*
vLO-19.460836706744324355*vRF)**2+Float(93896430651279369523,497)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**3*exp(19.460836706744324355*
vLO+19.460836706744324355*vRF)+Float(92966763021068682671,497)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)**3*exp(19.460836706744324355*
vLO-19.460836706744324355*vRF))**(1/2)/(Float(85898953831112912821,26)*exp(19.
460836706744324355*vRF-19.460836706744324355*vLO)+Float(88400865107747269507,26
)*exp(-19.460836706744324355*vLO-19.460836706744324355*vRF)+Float(
83397042554478556142,26)*exp(19.460836706744324355*vLO-19.460836706744324355*
vRF)+Float(84231012980023341699,26)*exp(19.460836706744324355*vLO+19.
460836706744324355*vRF))**2)**(1/3)-.33333333333333333333*(Float(
75744833065313455978,-26)*exp(19.460836706744324355*vLO-19.460836706744324355*
vRF)+Float(78017178057272859655,-26)*exp(19.460836706744324355*vRF-19.
460836706744324355*vLO)+Float(-80289523049232263339,-26)*exp(-19.
460836706744324355*vLO-19.460836706744324355*vRF)+Float(-76502281395966590536,-
26)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF))/(Float(-
10121863749957223487,-24)*exp(19.460836706744324355*vRF-19.460836706744324355*
vLO)+Float(-10416675315489958152,-24)*exp(-19.460836706744324355*vLO-19.
460836706744324355*vRF)+Float(-98270521844244888230,-25)*exp(19.
460836706744324355*vLO-19.460836706744324355*vRF)+Float(-99253227062687337108,-
25)*exp(19.460836706744324355*vLO+19.460836706744324355*vRF));
--------------  end file VIFclosed ---------------------------------------
```

Appendix D
Instructions for Running Extraction Code

D.1 Overview

The source code for the "measure" program is in ecemw6:/u0/users/MBS/pjl/c/measure. The macro definitions that affect how the program is compiled are documented in the comments of the makefile. I would recommend to always define "SAFE", (e.g. -DSAFE). If this is defined, then the program checks for valid ranges of index variables and other "assertions". This can aid in finding bugs in the code. There are several stages for running the code

- Edit the source code to use your particular simulator. The code currently uses the IBM internal simulator AS/X (follow on of ASTAP) and expects the output of the simulator to be in rawspice format.
- Create *.status file that contains all of the information that the program needs.
- Run initialization to create directories and calculate the required simulation input.
- Create the input files for the simulations
- Run the simulations
- Extract the H values based on the simulation

D.2 Customizing for a Simulator

In order to customize the extraction code to use a particular simulator, there are files that might have to be modified.

- mk.c

  This code creates the input files for the simulations.
- run.c

  This code initiates the simulations.
- get_spect.c

  This code extracts a spectrum of a single simulation.
- extr.c

  This code does the actual extraction and calls the top function in get_spect.c. The file extr.c also calls get_vect.c if the "ffextr" option is used (see section D.7). The file get_vect.c is currently used by get_spect.c and is useful only for rawspice format output files.

D.3 Creating *.status file

All of the simulation steps are related to a single root name. For our examples, we will use the name “tmp”. So for our example, we need to create the file “tmp.status” in the correct format. the easiest way to do this is to first use the dflt option to create a sample file, e.g.

```
measure dflt tmp
```

which will will return the prompt

```
 This will over write file tmp.status if it exists
 Do you want to continue ?
 YOU MAY LOOSE DATA IF YOU ANSWER  y
 (y or n): y
```

The macro NO_ASK can be used when compiling to supress this prompt. The resulting tmp.status file should look like:

```
This is a dummy sample description
max_input: 1.1
max_order: 3
xtra_runs: 0
fn: the_name_of_this_file(without the .status)
test_model_lib: dummy_test_model_lib
runcontrol: dummy_runcontrol
acruncontrol: dummy_acruncontrol
loadlib: dummy_loadlib
num_extract_runs: 0
tot_runs: 0
run_num: -1
calc_dc: yes
out_dc_offset: 0
num_freq: 3
freq[0]: 0
freq[1]: 1
freq[2]: 3
```

The first line is just a comment line and contain anything. The fourth line (fn:) must be changed to

```
fn: tmp
```

The first line sets the maximum absolute value of the input signal that is to be used in the extraction. The resulting extracted Volterra nonlinear transfer functions should be used to predict output only when the peak input is less than or equal to this value. The next line is the assumed maximum order of nonlinearity, and the third line is the number of extra simulations per run. This must be greater than or equal to zero. A value greater than zero will allow a least square fit where the number of points is greater than the order of the fitting polynomial. For instance, with max_order set to 3 and xtra_runs set to 1, 4 points will be used to fit a third order polynomial. For these curve fits the 0th order coefficient (a_0) is known to be zero.

The next lines test_model_lib, runcontrol, acruncontrol, and loadlib are specific to the AS/X simulator. The line num_extract_runs should be left at 0. This line is updated by the program during during the extraction process to allow for multi-step extraction. The line tot_runs is updated during the init option and should not be manually changed in most circumstances. The run_num line is updated during the run option (simulations), and can be used to re-start this process at a particular run if this option was interupted for some reason (e.g. network outage, disk space problems, etc). The starting value of -1 corresponds to the AC simulation used with the ffextr option.

The next two lines, calc_dc and out_dc_offset are use to set the value of H_0. The calc_dc line should be set to either "yes" or "no". If it is set to "yes", then the value of out_dc_offset will be calculated with a simulation. If is set to "no", the the desired value of H_0 should be entered by the user. The remaining lines give the input frequencies to be used. If you have more than three input frequencies, just add more lines.

D.4 Initializing

The init option is used for initialization, e.g.

```
measure init tmp
```

The output to the screen from this step and all other steps is highly dependent on if the following macros are defined at compiliation time, VVERBOSE, VERBOSE, QUIET, VQUIET. For more information on these see the comments in the makefile. In our example, the following directories are created, tmp.xfrdir, tmp.inpdir, and tmp.rundir. Additionally, the following files are created, tmp.runs, tmp.xfrdir/tmp.H0, tmp.xfrdir/tmp.H1, tmp.xfrdir/tmp.H2, and tmp.xfrdir/tmp.H3. The file tmp.runs contains the input conditions for the simulations, and the tmp.xfrdir/tmp.H* files are the initial files where the extracted values will be stored.

D.5 Creating Simulation Files

The creation of simulation files is performed with the mk (stands for make) option, e.g.

```
measure mk tmp
```

Files are added to the tmp.inpdir and tmp.rundir directories. To create a file specific to a single run, the smk option can be used. For example, to create the files associated with run 2, the following command can be issued

```
measure smk tmp 3
```

D.6 Running Simulations

The actual simulation requires the most computer resource. To run all of the simulations, the following command is issued

```
measure run tmp
```

To perform the simulations for a single run, use the srun option. For instance, to perform the simulations created with the above smk command:

```
measure srun tmp 3
```

The run command can be used to start at a particular run number and continue to the end by editing the *.status file and changing the run_num line.

D.7 Extracting Values

To read the simulation results and calculate the Volterra nonlinear transfer functions, enter

```
measure extr tmp
```

The extr command can also be used to restart the extraction at a particular run number and continue to the end by editing the *.status file and changing the num_extract_runs line. Care must be taken when doing this because multiple runs with the same input frequency should not be broken up with this technique. The format of the *.xfrdir/*.H* files can be changed from ascii to binary by editing these files before the extraction procedure is run. To make this change, just change the word "ascii" to "binary".

The ffextr option can be used in place of the extr option if the linear response (H_1) is to be taken from a linear AC simulation.

D.8 Calculating from Tables

The extracted model output can be calculated with the calc option. The format for our example is

```
measure calc tmp
```

The file tmp.calcin must be created to give the input for the model calculation. Below is an example:

```
num_freq: 2
period: 5e-7
 freq[0]: 1 real[0]: -0.27 imag[0]:  0.123
 freq[1]: 3 real[1]:  0.19 imag[1]:  0.456
```

The output will be written to the two files tmp.calcout and tmp.rawout. The file *.calcout lists the output phasors in the form similar to the *.calcin file, and the *.rawout is the time domain representation in rawspice format. The spice waveform display program "nutmeg" can be used to display the waveforms.