ABSTRACT

HU, JIE. Radio Frequency Front-End Measurement and Modeling. (Under the direction of Dr. Michael B. Steer.)

New nonlinear characterization and behavioral modeling strategies are developed to capture the complex nonlinear behavior of RF circuits. Two novel absolute phase calibrated nonlinear vector network analyzer (NVNA) systems are developed using common bench-top instruments and without relying on a multi-harmonic generator as in existing systems. The 1-channel VSA-based NVNA demonstrates for the first time a single receiver channel implementation for calibrated nonlinear network measurement. This enables future self-tuning and self-healing RF front-end architectures that require an accurate and low overhead on-chip self-characterization method. The hybrid NVNA system does not require a multi-harmonic generator, and thus overcomes the limitation of existing systems which limit the narrowest tone-spacing usable in multi-tone testing. Also developed is an adaptive reduced-order procedure for developing grey box models with an expanded range of model architectures beyond the Wiener and Hammerstein family. The stochastic model extraction combines a modified genetic algorithm and the Nelder-Mead simplex algorithm to replace the commonly used human-in-the-loop ad-hoc deterministic approach in which a skilled operator must guide model fitting. The procedure is ideal for the extraction of multiple distinct model candidates given incomplete observations, noisy data, and inexact model structure. This modeling strategy was successful in replicating the magnitude and phase asymmetry behavior in a multistage RF circuit. As part of the modeling strategy, a more general algorithmic formulation of the direct calculation method for Volterra circuit analysis is developed. The new formulation can handle compression effects and arbitrary degree of device nonlinearity, and is suitable for implementation in a circuit simulator.

© Copyright 2011 by Jie Hu

All Rights Reserved

Radio Frequency Front-End Measurement and Modeling

by
Jie Hu

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

Electrical Engineering

Raleigh, North Carolina

2011

APPROVED BY:

Dr. Huaiyu Dai	Dr. Mohammed A. Zikry
Dr. David Schurig Co-Chair of Advisory Committee	Dr. Michael B. Steer Chair of Advisory Committee

ACKNOWLEDGEMENTS

I would like to thank several people for their support during my completion of this dissertation: Dr. Michael Steer, for guiding me through my publications and presentations, and for providing me with professional contacts in the Army Research Laboratory; Dr. Kevin Gard, for his guidance and continuous support; Dr. Mohammed Zikry, Dr. Huaiyu Dai, and Dr. David Schurig, for serving on my Ph.D. advisory committee.

I would also like to thank my graduate student colleagues and friends for their critical thinking and professionalism: Dr. Greg Mazzaro, Chris Saunders, Rob Harris, Glen Garner, Vrinda Haridasan, Justin Lowry, Dr. Frank Hart, Dr. Nikhil Kriplani, Dr. Zhiping Feng, Dr. Xueming Yang, Dr. Mingsheng Li, Pochih Lin, Miaofu Ding, Yue Chen, Dr. Gautham Krishnamurthy and Dr. Prem Swaroop. These people provided me with a good working environment. Their wisdom and creativity have been invaluable to me.

Finally, but most importantly, I am grateful to my parents for their unconditional support. I owe a lot to my beloved wife Danqiong Hou for her patience and support while she continues her own graduate studies. I thank my son for the new chapter he started in my life.

TABLE OF CONTENTS

List of Figures . . . . . **vii**

List of Tables . . . . . **xiv**

Chapter 1 Introduction **1**
1.1 Overview 1
1.1.1 Communication System Analysis and Design 1
1.1.2 RF Front-End Behavioral Models in Link-Level Design Flow 6
1.2 Motivation & Research Objectives 8
1.2.1 Nonlinear Characterization: Absolute Phase Measurement 8
1.2.2 Nonlinear Modeling: Grey-Box Model Identification 10
1.3 Original Contributions 12
1.3.1 Novel 1-Channel Receiver Based NVNA 12
1.3.2 Novel 4-Port VNA-based NVNA 13
1.3.3 Grey-Box Model Identification Methodology 13
1.3.4 Direct Calculation Algorithm for Volterra Circuit Analysis 14
1.4 Dissertation Outline 15
1.5 Publications 16
1.5.1 Journal papers 16
1.5.2 Conference papers 17

Chapter 2 Prior Research in Nonlinear Metrology and Modeling **18**
2.1 Introduction 18
2.2 Review Nonlinear Metrology 20
2.2.1 Absolute Phase Signal Measurement 22
2.2.2 Signal Alignment 30
2.2.3 Time-Invariant Phase Network Measurement 31
2.2.4 Time-Invariant Phase Relation Definitions 37
2.2.5 Current Research 42
2.3 Review Nonlinear RF Behavioral Modeling 44
2.3.1 Black-Box Behavioral Models 46
2.3.2 Grey-Box Behavioral Models 50
2.3.3 Current Research 55
2.4 Summary 57

Chapter 3 Nonlinear Metrology **59**
3.1 Introduction 59
3.1.1 Dependence on Multi-Harmonic Generator 59
3.1.2 Implementation Using Common Benchtop Hardware 60
3.1.3 Proposed NVNA Setups 61
3.2 Novel 1-Channel VSA Based NVNA 61

3.2.1 Mimic Synchronous Multi-Channel Receiver . . . 62
3.2.2 Proposed NVNA Setup . . . 76
3.2.3 Theory of Absolute Phase Calibration . . . 78
3.2.4 Experimental Results . . . 81
3.2.5 Summary . . . 84
3.3 Novel 4-Port VNA Based NVNA . . . 85
3.3.1 Proposed NVNA Setup . . . 86
3.3.2 Theory of Absolute Phase Calibration . . . 92
3.3.3 Experimental Results . . . 98
3.3.4 Dynamic Range . . . 106
3.3.5 Measurement Time . . . 107
3.3.6 Summary . . . 108
3.4 Conclusion . . . 108

Chapter 4 Grey-Box Model Identification . . . 110
4.1 Introduction . . . 110
4.1.1 Stochastic Optimization and Multiple Optimum Solutions . . . 111
4.1.2 Proposed Identification Methodology . . . 112
4.1.3 Uniqueness of Model Parameters and Structures . . . 113
4.2 Grey-Box Model Identification Methodology with Minimal User Intervention . . . 115
4.2.1 Hybrid Genetic Algorithm Optimizer . . . 117
4.2.2 Identification Algorithm . . . 119
4.2.3 Experiment Design . . . 121
4.3 Case-Study: PA with Long-Term Memory . . . 124
4.3.1 Model Structure Simplification and Parametrization . . . 124
4.3.2 Identification Algorithm and Experiment Design . . . 126
4.3.3 Optimization Settings and Hybrid GA Performance . . . 128
4.3.4 Model Fitting Result . . . 131
4.4 Discussion . . . 133
4.4.1 Model Parameter and Structure Uniqueness . . . 137
4.4.2 Comparison to Circuit Design . . . 137
4.5 Conclusions . . . 139

Chapter 5 Direct Calculation Method for Volterra Circuit Analysis . . . 141
5.1 Introduction . . . 141
5.1.1 NLTF Method . . . 141
5.1.2 Direct Calculation Method . . . 142
5.1.3 Proposed Improvement to Direct Calculation Method . . . 143
5.2 Improved Direct Calculation Method . . . 144
5.2.1 General Formulation of Circuit Equations . . . 144
5.2.2 Determining the Nonlinear Current Source . . . 146
5.2.3 Circuit Analysis . . . 163
5.3 Circuit Analysis Examples . . . 165
5.3.1 Differential Pair as Single-Balanced Mixer . . . 165

5.3.2 RF Receiver Front-End . . . 172
5.4 Conclusions . . . 182

Chapter 6 Modeling Cosite Inteference . . . 183
6.1 Introduction . . . 183
6.1.1 Cosite Inteference and the Near-Far Problem . . . 185
6.1.2 Cosite Simulation in RF Design . . . 186
6.1.3 Simulation Approaches and Behavioral Models . . . 187
6.2 Behavioral Modeling for Cosite Interference Simulation . . . 188
6.2.1 Modeling Antenna-to-Antenna Coupling . . . 189
6.2.2 Modeling Nonlinear RF Front-End Circuits . . . 189
6.3 Case-Study . . . 201
6.3.1 Measurement Setup . . . 202
6.3.2 Modeling Results . . . 216
6.4 Conclusions . . . 216

Chapter 7 Conclusions and Future Work . . . 218
7.1 Summary of Research and Original Contributions . . . 218
7.2 Future Research . . . 220

References . . . 222

Appendices . . . 231
Appendix A Symbolic Volterra Circuit Simulator Implementing Direct Calculation Method . . . 232
A.1 User Guide . . . 232
A.2 Generate Nonlinear Current Source . . . 233
A.2.1 Main Script . . . 233
A.2.2 Functions . . . 235
A.3 Define Circuit . . . 261
A.3.1 Differential Pair as Single-Balanced Mixer . . . 261
A.3.2 RF Front-End Behavioral Model . . . 262
A.4 Circuit Analysis . . . 264
A.4.1 Main Script . . . 264
A.4.2 Functions . . . 264
Appendix B 4-Port VNA-Based NVNA: Measurement and Calibration . . . 289
B.1 User Guide . . . 289
B.2 Calibration Measurement and DUT Measurement . . . 291
B.2.1 Open Instrument Connection . . . 291
B.2.2 Close Instrument Connection . . . 302
B.2.3 SOLT Calibration Measurement . . . 303
B.2.4 Power Calibration Measurement . . . 308
B.2.5 DUT Measurement . . . 315
B.3 DUT Measurement Correction . . . 323

B.3.1 Correction Script 323
B.3.2 Correction Functions 325
Appendix C Grey-Box Model ID: Hybrid Genetic Optimizer 342
C.1 User Guide 342
C.2 Hybrid Genetic Optimizer 343
C.2.1 Main Script 343
C.2.2 Functions 346
C.3 Example Usage: PA Model ID 359
C.3.1 Main Script: Part 1 359
C.3.2 Main Script: Part 2 364

LIST OF FIGURES

Figure 1.1 Communication system hierarchy. 3
Figure 1.2 Communication system simulation hierarchy. 4
Figure 1.3 RF front-end behavioral model in link-level simulation. 5
Figure 1.4 RF front-end behavioral model in link-level design flow. 7

Figure 2.1 Characterizing the incident and scattered port waves using multiple synchronous receiver channels. 21
Figure 2.2 For (a) linear systems, the excitation can be used as the phase reference since the response is at the same frequency as the excitation, and their relative phase is the phase of the linear transfer function. For (b) nonlinear systems, the response contains additional spectral components that are not present in the excitation, and requires an alternative phase reference definition. 22
Figure 2.3 Harmonic phase reference generator showing equally spaced tones in the output spectrum. 23
Figure 2.4 NVNA setup using a broadband receiver after [1]. 24
Figure 2.5 NVNA setup using a narrowband receiver. 29
Figure 2.6 Relative phase measurement system that synchronously captures the input and the output using a 2-channel MTA, then generates the phase reference from the input using a software reference nonlinearity. 34
Figure 2.7 Relative phase measurement system that generates the phase reference from the input using a hardware reference nonlinearity, then measure the relative phase using either: (a) cancelation method; or (b) synchronous measurement method. 35
Figure 2.8 Relative phase measurement system that generates the phase reference from a phase locked signal generator, then measure the relative phase using either: (a) cancelation method; or (b) synchronous measurement method. 35
Figure 2.9 High level view of the identification algorithm using a deterministic optimizer for the linear and nonlinear model parameters: (a) brute force approach estimating all parameters simultaneously; and (b) partitioned parameter approach alternately optimizing linear and nonlinear parameters in separate iteration steps. 54

Figure 3.1 Block diagram of a 1-channel VSA. 62
Figure 3.2 Sequentially sample DUT signals using VSA, to mimic a synchronous multi-channel receiver. 63
Figure 3.3 The excitation waveform sequence during sequential sampling. 64
Figure 3.4 Spectrogram of a dynamic time-frequency two-tone signal to reduce measurement time, where the magnitude is plotted versus envelope frequency. 68

Figure 3.5 Composite envelope spectrum of a dynamic time-frequency two-tone signal using: (a) equal number of periods for each tone-spacing segment; and (b) uniform time duration for each tone-spacing segment. The magnitude is plotted versus envelope frequency. 70
Figure 3.6 Compare S_{21} and linear transmission response measured using the novel 1-channel VSA method: (a) magnitude; and (b) phase. Both are plotted versus frequency deviation from the center frequency of the 2-tone. . . . 72
Figure 3.7 Envelope spectrum of time-domain record captured by the VSA at several tone-spacings for: (a) the undistorted signal $x(t)$; and (b) the distorted signal $y(t)$. The magnitude is plotted versus envelope frequency. 73
Figure 3.8 Compare predetermined (solid blue line) and estimated a_1: (a) magnitude; and (b) a_1 phase. Both are plotted versus frequency deviation from the center frequency of the 2-tone. 74
Figure 3.9 Compare predetermined (solid blue line) and estimated a_3: (a) magnitude; and (b) phase. Both are plotted versus frequency deviation from the center frequency of the 2-tone. 75
Figure 3.10 Proposed NVNA setup using a 1-channel VSA. 76
Figure 3.11 For transmission response measurement of well-matched DUTs, a simplified version of the proposed 1-channel VSA-based NVNA, which is the same as Fig. 3.2. 82
Figure 3.12 Compare the linear transmission response measured at 2-tone input frequencies f_1 and f_2, using the proposed 1-channel VSA-based NVNA, a commercial NVNA system, and a VNA. The frequency deviation are in dB scale referenced to 1 kHz. 83
Figure 3.13 Comparison the nonlinear transmission response at frequencies $2f_1 - f_2$ and $2f_2 - f_1$ for several input power levels, using the proposed 1-channel VSA-based NVNA and a commercial NVNA system. The frequency deviation are in dB scale referenced to 1 kHz. 84
Figure 3.14 Connection diagrams of the proposed hybrid NVNA: (a) for DUT measurement; and (b) for phase calibration measurement. The proposed hybrid NVNA system combines a sampling oscilloscope-based NVNA, and a four-port VNA-based NVNA, with phase-locked signal generators for multi-tone testing. 87
Figure 3.15 Power waves at the fundamental frequency f_1 for a two-tone test at 2 GHz with 200 kHz tone-spacing following the first phase correction described in (2.1): (a) amplitude and (b) phase. 100
Figure 3.16 Power waves at the IM3 frequency $2f_1 - f_2$ for a two-tone test at 2 GHz with 200 kHz tone-spacing following the first phase correction described in (2.1): (a) amplitude and (b) phase. 101
Figure 3.17 Phases of power waves at port 1 and port 2 for the two-tone test following the final phase correction step as described in (3.34)–(3.36): (a) at the fundamental, f_1; and (b) at the IM3, $2f_1 - f_2$. 102

Figure 3.18 Phases of power waves for the two-tone test after applying the time-invariant phase definition (3.18): (a) at the fundamental, f_1; and (b) at the IM3, $2f_1 - f_2$. 104
Figure 3.19 Power waves at the IM3 frequency $2f_1 - f_2$ for a two-tone test at 2 GHz with 200 Hz tone-spacing after applying the time-invariant phase definition (3.18): (a) amplitude and (b) phase. 105

Figure 4.1 Models containing one memoryless nonlinearity can be separated into disjoint parts: (a) basic Wiener-Hammerstein model; (b) Wiener-Hammerstein model with a linear feedback network; and (c) Wiener-Hammerstein model with a nonlinear feedback network. 114
Figure 4.2 SISO model containing 1 memoryless nonlinearity. 115
Figure 4.3 Proposed general approach for the extraction of nonlinear equivalent circuit models in the frequency domain. 116
Figure 4.4 Proposed hybrid optimization method. The settings for GA are the number of generations, the population size, and the selection of individuals used in local search. The settings for local search are the relative error tolerance defined as the error difference between the iterations relative to the lowest error achieved (4.3), and the maximum number of consecutive iterations such that the result is below the relative error tolerance. In this way, the absolute error tolerance need not be set, but will be changing adaptively. 118
Figure 4.5 Higher level identification algorithm: (a) estimating all parameters simultaneously; and (b) partitioned approach alternately optimizing linear and nonlinear parameters in separate iteration steps. The settings of the identification algorithm are the relative error tolerance defined as the error difference between the iterations relative to the lowest error achieved (4.3), the maximum number of consecutive iterations such that the result is below the relative error tolerance, and the selecting of individuals from each population between the iteration steps. In this way, the absolute error tolerance need not be set, but will be changing adaptively. 120
Figure 4.6 For narrow-band circuits, a less computation intensive variation of the identification algorithm. Initially, assuming constant linear parameters in the carrier harmonic bands, a 1st-pass estimate results from stochastic optimizer. Then, the linear and the nonlinear parameters are alternately estimated in separate iteration steps. 122
Figure 4.7 Multi-stage PA with 2-stages shown. 125
Figure 4.8 Model after simplification of the structure and parameter set. Structural simplification assumptions: last stage nonlinear, no dominant feedback or any linear feedback can be approximately transformed so unilateral device model can be used. Parameter simplification assumptions: active device is a BJT, only transmission measurements will be modeled, all linear memory is now carried by $Z(w)$. 126

Figure 4.9 Across the generations: (a) error for the best individual in each GA generation and for the 5 individuals selected for local searches; and (b) corresponding values of an estimated model parameter for the 5 individuals. There is correlation between the error plateaus in GA, and the error level after the proposed Hybrid-GA method. 130
Figure 4.10 Across the generations, the number of function evaluations for the 5 individuals selected for local searches. 131
Figure 4.11 Error and the number of function evaluations versus index to the final solutions of hybrid GA method, with the index sorted in order of increasing error. Compared to hybrid GA, the standard GA requires anywhere between the same to seven times the number of function evaluations. . . 132
Figure 4.12 Values of an estimated model parameter with the first forty lowest errors. 133
Figure 4.13 Modeled versus measured response: (a) fundamental power; (b) fundamental phase. 134
Figure 4.14 Modeled versus measured response: (a) IM3 power; and (b) IM3 phase. . 135
Figure 4.15 Estimated linear parameter at baseband frequency: (a) magnitude; and (b) phase. The high-pass nature corresponds physically to the inductive nature nature of the bias network. 136
Figure 4.16 The different levels of element model abstraction for: (a) circuit design; and (b) RF model identification. 138

Figure 5.1 General nonlinear network. 144
Figure 5.2 Algorithm flow chart for the proposed direct calculation method of Volterra circuit analysis. 164
Figure 5.3 Circuit schematic of a bipolar differential pair. 166
Figure 5.4 Equivalent circuit of a bipolar differential pair used as single-balanced mixer. 167
Figure 5.5 Block diagram of a RF receiver front-end. 170
Figure 5.6 A frequency-domain filter model equivalent circuit, with element definitions similar to a y-parameter model. 173
Figure 5.7 A simple frequency-domain amplifier model equivalent circuit, with element definitions similar to a y-parameter model. The port impedance behavior is modeled by a parallel combination of linear frequency-dependent impedances, nonlinear resistances, nonlinear capacitances and nonlinear inductances. Analogously, the transfer characteristic is modeled as a parallel combination of linear frequency-dependent transconductances, nonlinear transconductances, nonlinear trans-capacitances and a nonlinear trans-inductances. (The latter two are similar to their 2-terminal counterpart, except that the generated nonlinear current is summed at terminals other than terminals of the controlling variable.) 173
Figure 5.8 A more general frequency-domain amplifier model based on Fig. 5.7. The arbitrary frequency weighting for the nonlinear coefficients is more suitable for modeling the complex nonlinear interactions in multi-stage amplifiers. 175

Figure 5.9 A RF receiver front-end equivalent circuit model. 176

Figure 6.1 In a cosite interference scenario, multiple co-located radios simultaneously operate in the same frequency band. The poor antenna-to-antenna isolation couples interference between the radios. 184
Figure 6.2 In model identification using a narrowband two-tone signal (centered at 2 GHz with 60 MHz tone-spacing), nonlinear response up to the third-order are sorted into bands for parametrization. Sorting is based on the magnitude of the S_{21} measurement of an amplifier. IM frequencies with frequency response variation below a user set threshold (2 dB here) are grouped into the same band. 193
Figure 6.3 In model identification using a broadband two-tone signal (f_1= 500 MHz and f_2= 900 MHz), nonlinear response up to the third-order are sorted into bands for parametrization. Sorting is based on the magnitude of the S_{21} measurement of an amplifier. IM frequencies with frequency response variation below an user set threshold (2 dB here) are grouped into the same band. 198
Figure 6.4 Block diagram of the cosite interference scenario under study. 202
Figure 6.5 Antenna-to-antenna coupling between each of the transmitter antennas and the receiver antenna from linear S-parameter measurements. 203
Figure 6.6 Linear S-parameter measurement: (a) transmission response between each of the transmitters and the receiver; and (b) reflection at the input of each transmitter and the output of the receiver. 205
Figure 6.7 Block diagram of the cosite interference measurement step for the case where a single tone is transmitted at 500 MHz, and another single tone is transmitted at 900 MHz. 206
Figure 6.8 Measurement results from the test setup shown in Fig. 6.7. The power of the incident 500 MHz CW signal ($P1$) at the measurement plane AA is swept from -12.25 dBm to -2.25 dBm in 2.5 dB steps. The power of the incident 900 MHz CW signal ($P2$) at the measurement plane BB is swept from -12.5 dBm to -2.5 dBm in 2.5 dB steps. The power of the transmitted IM responses at the measurement plane CC depends on three independent variables, namely the IM frequency, $P1$ and $P2$. Two of the independent variables, IM frequency and $P2$, are used as the coordinate axis. The third independent variable, $P1$, is used as a parameter. The power of the transmitted IM responses with order less than or equal to three are plotted using solid lines. The power of the incident CW signal at 500 MHz and 900 MHz are plotted using dashed lines. Lines with different parametric $P1$ values are differentiated by color. 207

Figure 6.9 Measurement results from the test setup shown in Fig. 6.7. The power of the incident 500 MHz CW signal ($P1$) at the measurement plane AA is swept from -12.25 dBm to -2.25 dBm in 2.5 dB steps. The power of the incident 900 MHz CW signal ($P2$) at the measurement plane BB is swept from -12.5 dBm to -2.5 dBm in 2.5 dB steps. The power of the transmitted IM responses at the measurement plane CC depends on three independent variables, namely the IM frequency, $P1$ and $P2$. Two of the independent variables, IM frequency and $P2$, are used as the coordinate axis. The third independent variable, $P1$, is used as a parameter. The power of the transmitted fifth-order IM responses are plotted using solid lines. Lines with different parametric $P1$ values are differentiated by color. 208
Figure 6.10 Block diagram of the cosite interference measurement step for the case where a single tone is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz. 209
Figure 6.11 Power spectrum of the linear responses at the receiver output centered at: (a) f_1; and (b) f_2. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz. 210
Figure 6.12 Power spectrum of the second-order IM responses at the receiver output centered at: (a) $2f_1$; and (b) $2f_2$. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz. 211
Figure 6.13 Power spectrum of the second-order IM responses at the receiver output centered at: (a) f_2-f_1; and (b) f_2+f_1. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz. 212
Figure 6.14 Power spectrum of the third-order IM responses at the receiver output centered at: (a) $2f_1 - f_2$; and (b) $2f_1 + f_2$. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz. . 213
Figure 6.15 Power spectrum of the third-order IM responses at the receiver output centered at: (a) $2f_2 - f_1$; and (b) $2f_2 + f_1$. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz. . 214
Figure 6.16 Power spectrum of the third-order IM responses at the receiver output centered at: (a) $3f_1$; and (b) $3f_2$. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz. 215
Figure 6.17 Compare simulated versus the measured IM response centered about $2f_2$. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz. 217

Figure B.1 Connection diagrams of linear SOLT calibration: connecting short-open-load at port-1. The loads represent the off-state output impedances of the signal generators making up the two-tone source. 292
Figure B.2 Connection diagrams of linear SOLT calibration: connecting short-open-load at port-2. 293
Figure B.3 Connection diagrams of linear SOLT calibration: connecting thru between port-1 and port-2. The loads represent the off-state output impedances of the signal generators making up the two-tone source. 294

Figure B.4 Connection diagram of power calibration. The load represents the off-state output impedance of one of the signal generators making up the two-tone source. 295
Figure B.5 Connection diagram of phase calibration. The loads represent the off-state output impedances of the signal generators making up the two-tone source. 296
Figure B.6 Connection diagram of DUT measurement for the hybrid NVNA. 297

LIST OF TABLES

Table 5.1 For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the NLTF up to the fifth order. 149
Table 5.2 For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $f_2 - f_1$ up to the fifth order. 152
Table 5.3 For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $f_1 + f_2$ up to the fifth order. 152
Table 5.4 For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1$ up to the fifth order. 153
Table 5.5 For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at f_1 up to the fifth order. 154
Table 5.6 For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1 - f_2$ up to the fifth order. 155
Table 5.7 For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1 + f_2$ up to the fifth order. 156
Table 5.8 For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $3f_2$ up to the fifth order. 157
Table 5.9 For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $f_2 - f_1$ up to the fifth order. 160
Table 5.10 For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $f_1 + f_2$ up to the fifth order. 161
Table 5.11 For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1$ up to the fifth order. . . 161
Table 5.12 For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at f_1 up to the fifth order. . . . 161
Table 5.13 For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1 - f_2$ up to the fifth order. 162
Table 5.14 For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1 + f_2$ up to the fifth order. 162
Table 5.15 For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $3f_2$ up to the fifth order. . . 163
Table 5.16 The 2-tone response of a differential pair single-balanced mixer at $f_2 - f_1$ up to the fifth order. 169
Table 5.17 The 2-tone response of a differential pair single-balanced mixer at $f_1 + f_2$ up to the fifth order. 169
Table 5.18 The 2-tone response of a differential pair single-balanced mixer at f_1 up to the fifth order. 170
Table 5.19 The 2-tone response of a differential pair single-balanced mixer at $2f_2 - f_1$ up to the fifth order. 170
Table 5.20 The 2-tone response of a differential pair single-balanced mixer at $2f_2 + f_1$ up to the fifth order. 171

Table 5.21 The 2-tone response of a differential pair single-balanced mixer at $3f_1$ up to the fifth order. 171
Table 5.22 The 2-tone response of a RF receiver front-end model at $f_2 - f_1$ up to the third order. 179
Table 5.23 The 2-tone response of a RF receiver front-end model at $f_1 + f_2$ up to the third order. 179
Table 5.24 The 2-tone response of a RF receiver front-end model at f_1 up to the third order. 179
Table 5.25 The 2-tone response of a RF receiver front-end model at f_2 up to the third order. 180
Table 5.26 The 2-tone response of a RF receiver front-end model at $2f_1 - f_2$ up to the third order. 180
Table 5.27 The 2-tone response of a RF receiver front-end model at $2f_1 + f_2$ up to the third order. 181
Table 5.28 The 2-tone response of a RF receiver front-end model at $2f_2 - f_1$ up to the third order. 181
Table 5.29 The 2-tone response of a RF receiver front-end model at $2f_2 + f_1$ up to the third order. 181

Table 6.1 In model identification using a narrowband two-tone signal (centered at 2 GHz with 60 MHz tone-spacing), the contributing f_{IM} are listed for IM response being modeled at frequencies f_1, f_2, $2f_1 - f_2$, and $2f_2 - f_1$. . . . 195
Table 6.2 In model identification using a broadband two-tone signal (f_1= 500 MHz and f_2= 900 MHz), the contributing f_{IM} frequencies are listed for IM response being modeled at frequencies f_1, f_2, $2f_1 - f_2$, and $2f_2 - f_1$. . . . 200

Chapter 1

Introduction

This introductory chapter discusses the principal motivations and research objectives for the work conducted in this thesis. This is followed by a summary of the original contributions and the dissertation outline. A list of publications is given at the end.

1.1 Overview

The consumer demand for tether-less mobile communication devices continues to fuel the growth of the wireless communication market. The industry has responded to this demand by offering a wide variety of applications, while improving the quality of service and lowering the cost of successive generation of products. Commensurate with these advanced applications are the proliferation of complex standards, which dictate how the mobile devices will communicate with each other. Together with the need for short time-to-market, these demands can only be met through reduced prototyping, and the increased reliance on computer-aided design (CAD) tools.

1.1.1 Communication System Analysis and Design

To deal with the complexity in the analysis and design of complex communication systems, modern CAD tools have a hierarchical view of communication systems [2], [3]. A communication

network consists of communication nodes, which are interconnected via communication links as shown in Fig. 1.1. A digital communication link comprises a communication channel, an encoder, a decoder, a modulator, a demodulator, a transmitter, and a receiver.

Performance evaluation and tradeoff studies are similarly carried out in a hierarchical fashion, generally using three conceptual layers of abstraction as shown in Fig. 1.2. At the network level, packet flow is simulated in an event-driven simulator, to derive specifications for the communication nodes and the communication links. At the link level, bit error rate (BER) performance is evaluated to check specifications derived at the network level, and to establish specifications for the components in the link. At the implementation level inside the communication link, component performance is evaluate in circuit simulators such as Spice, or digital simulators supporting a hardware description language (HDL). The circuit-level description of the radio frequency (RF) front-end usually cannot be used in the performance tradeoff iteration loop at the link level, due to long simulation time. Typically a component's behavioral model, extracted from simulation or measurement data, needs to be provided to the link level.

For simulation efficiency, link-level entities usually have behavioral descriptions. For BER performance evaluation, the common platform for these behavioral descriptions is typically time-driven waveform simulation. However, the variety of implementation level entities makes the behavioral modeling job difficult. Further division at the link level is possible, according to the underlying theory for the type of entity at the implementation level, as shown in Fig. 1.3. The coder design is based on information theory, while modulator design is based on communication theory. Both have algorithmic behavioral descriptions at the link level, and are implemented as digital or mixed-signal circuits. The physical channel entails the electromagnetic interaction between the antenna and the environment. At the link level, the channel propagation model can have either deterministic or stochastic behavioral description. The transceiver design is based on circuit theory. In wireless networks, the role of RF front-end modeling is to supply transceiver behavioral models.

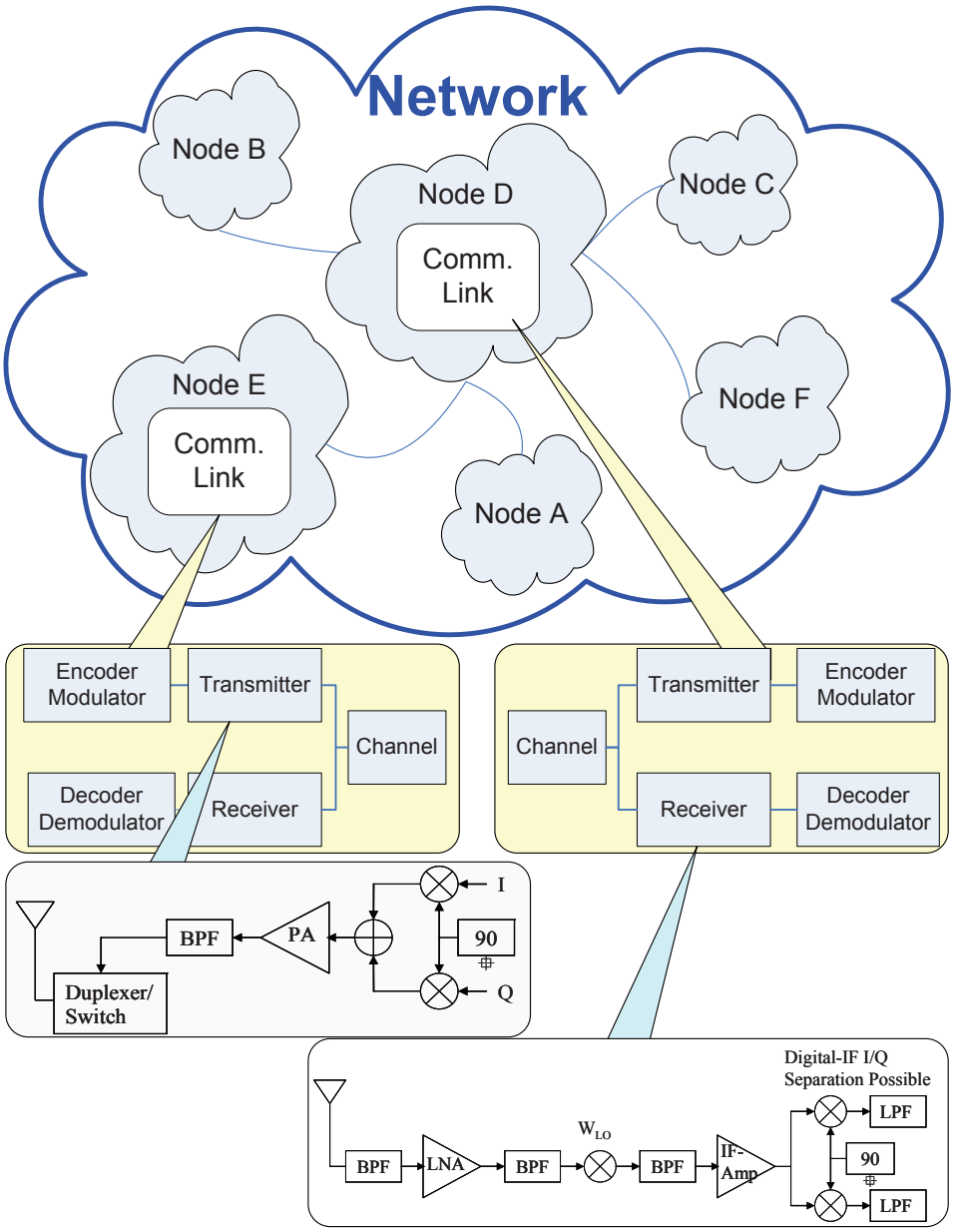


Figure 1.1: Communication system hierarchy.

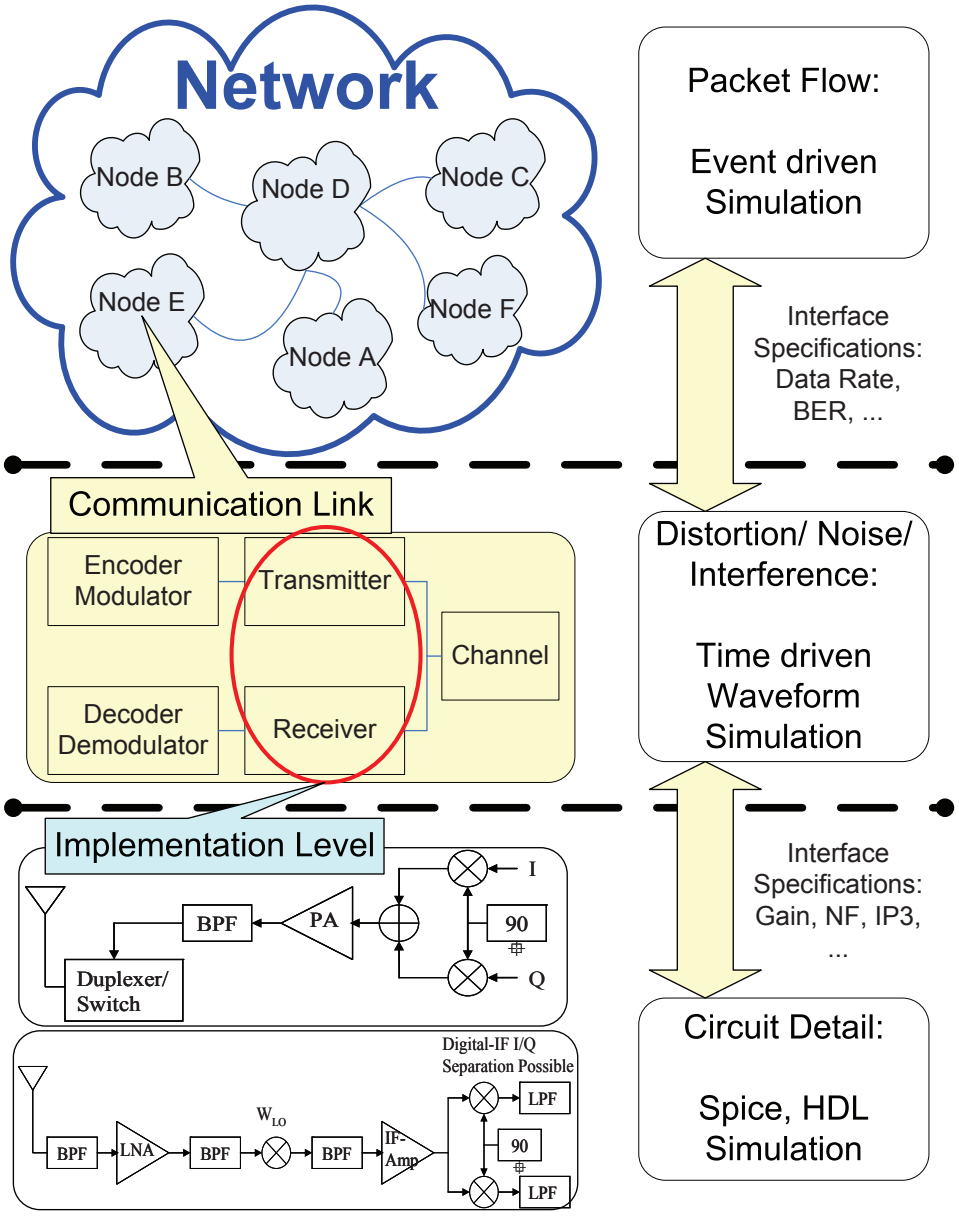


Figure 1.2: Communication system simulation hierarchy.

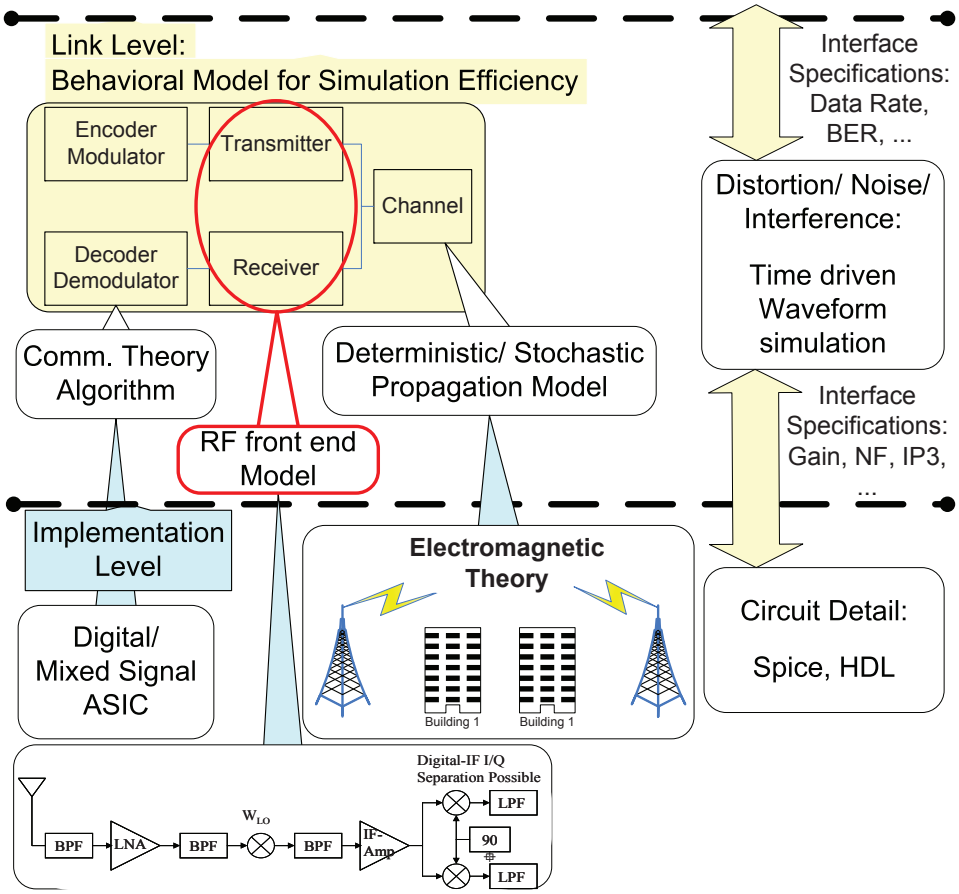


Figure 1.3: RF front-end behavioral model in link-level simulation.

1.1.2 RF Front-End Behavioral Models in Link-Level Design Flow

The large body of CAD techniques developed for link-level simulation can be categorized approximately as [2], [3]:

1. Formula-based approach: output computed from analytic expressions
    - This approach has questionable accuracy, but is computationally efficient.
    - This approach enables broad design space exploration in the early design stages; and provides insight into the dominate factors affecting system performance.
2. Simulation-based approach: waveform level simulation using detailed mathematical model or design-specific empirical model
    - This yields improved accuracy, however what can be done is limited by the available computation resource.
    - Detailed performance tradeoff analysis in the later stages of design.

Both analytical and empirical models are needed at different stages of development in system design. An analytical model is useful when no component design information except the specification is known. The empirical model is useful when circuit-level simulation data or component prototype measurement result are available.

A typical link-level design flow, shown in Fig. 1.4, is detailed below to clarify the use of the above CAD techniques:

1. Top-level concept specification: set information rate and BER.
2. Candidate system evaluation: formula-based link budget analysis to iterate on the candidate design topology.
3. Top-down system design: simulation-based performance tradeoff analysis using detailed mathematical model (ex. parametric behavioral model [4]), to set specifications for components inside the link.

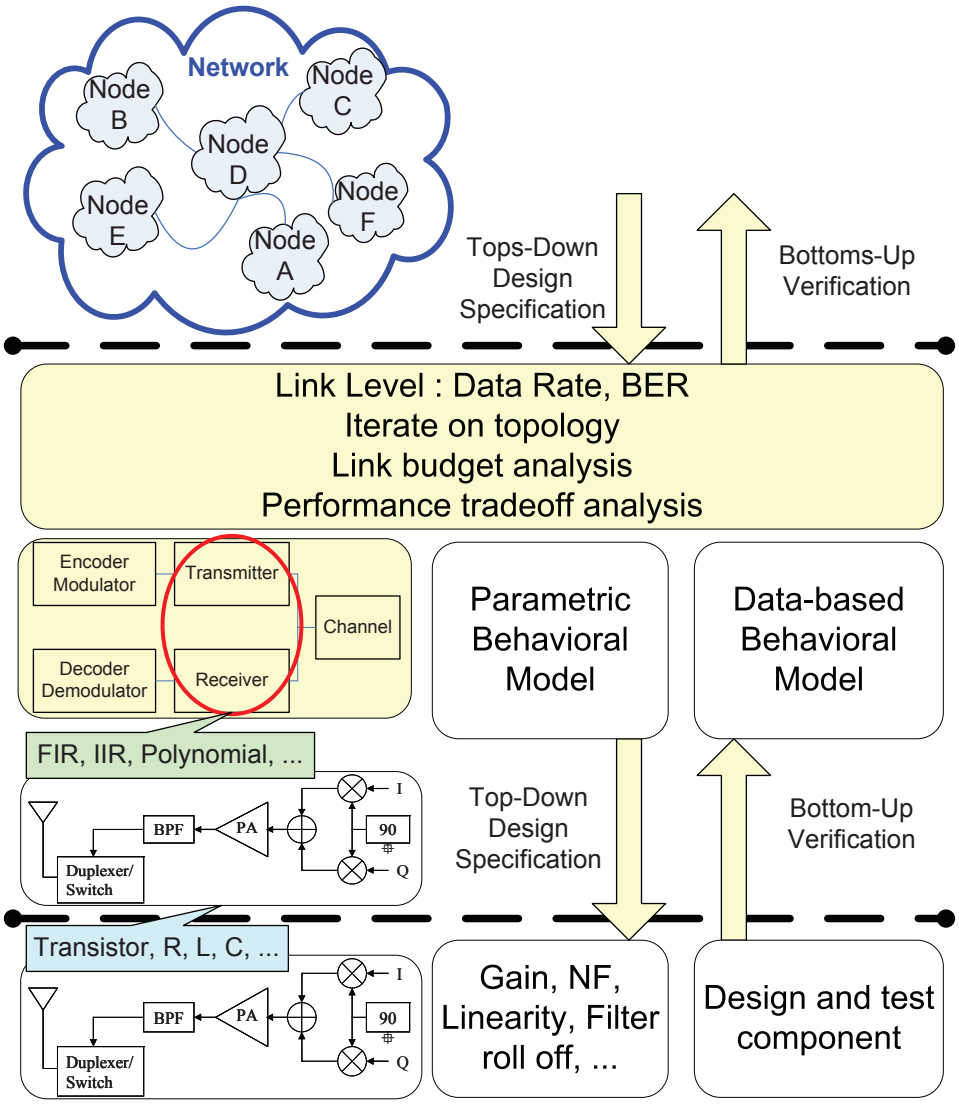


Figure 1.4: RF front-end behavioral model in link-level design flow.

4. Bottom-up verification: simulation-based performance tradeoff analysis using design specific empirical model (ex. data-based behavioral model [4]).

5. System prototype: validate simulation with measured performance.

Thus both analytical and empirical simulation models are critical to the success of a simulation-based rapid prototyping CAD environment. For RF front-end circuits, while mature filters models have been developed, challenges remain in developing high-fidelity models for nonlinear components such as amplifiers and mixers.

1.2 Motivation & Research Objectives

The current trend in mobile applications is toward ever higher data rate, which requires wide bandwidth RF front-end circuits. The narrow bandwidth assumption used in early behavioral models is no longer valid. Similarly in nonlinear characterization, approximating the communication signal as a single tone cannot excite the realistic behavior of the circuit.

The current generation of characterization methods and behavioral models have been developed for accurate representation of these wideband circuits in simulation.

1.2.1 Nonlinear Characterization: Absolute Phase Measurement

In nonlinear characterization, absolute phase measurement using a nonlinear vector network analyzer (NVNA) or a large signal network analyzer (LSNA) has emerged as the state of the art. For characterizing a high-frequency nonlinear device under test (DUT), multi-tone or quasi-periodic excitation is commonly used. These excitations can be designed to closely model realistic communication signals. In experimental characterization, the incident and scattered port waves at the measurement plane are synchronously sampled and vectorially corrected. The measured power is traceable to calibration standards, and the measured phase conforms to the consensus phase standard which is widely adopted by the microwave community and under consideration by standards bodies.

One issue with existing NVNA setups are their reliance on the multi-harmonic generator to provide a phase reference. Such a phase reference generator may not be readily available on the bench in practice, or more importantly, impose a significant limit on the minimum tone-spacing in multi-tone testing. The latter makes characterization at narrow tone-spacings (as low as a few hundred hertz), as often required for characterizing memory behavior, impossible. The multi-harmonic generator is driven at the fundamental and must generate harmonic tones with enough power up to the maximum bandwidth of the receiver to be measured with acceptable signal to noise ratio (SNR). The power of the fundamental driving signal cannot be simply increased, since the lower harmonic tones will saturate the receiver's front-end. A microstrip filter can be designed to shape the generator output power and phase spectrum to lower its peak to average ratio (PAR). Still it is difficult to characterize microwave components without placing an unacceptable lower limit on the fundamental frequency of the multi-harmonic generator. A commercially available multi-harmonic generator is driven at 10 MHz, which limits the narrowest tone-spacing that can be used in multi-tone testing. Other experimental types may exist, but are not readily available to the practicing engineer.

Another issue that hinders the wide adoption of the absolute phase characterization concept in everyday design practice is that a commercially available NVNA constitutes a major capital investment. The more readily available bench top instruments are the arbitrary waveform generator (AWG), the spectrum analyzer, the oscilloscope, the downconverter, and the analog to digital converter (ADC) evaluation board. It would be a lower cost and more practical solution to build an absolute phase measurement system using the available hardware as described above.

Typically a single receiver channel can be built rather easily, compared to the four or five synchronous channels required in existing NVNAs. In the foreseeable future, self-tuning and self-healing concepts will likely be applied to RF front-end architectures. This will require on-chip self-characterization employing accurate absolute phase measurements. A single receiver channel setup capable of absolute phase measurement will certainly win over existing multi-channel setups, due to the reduced power consumption and lower circuit real estate.

Current research explores other possible measurement configurations that can be implemented using common bench top instruments, while keeping in mind the essential elements in absolute phase measurement. One area of exploration is in possible implementations using only a single receiver channel. Another area of exploration is in possible implementations that do not require a multi-harmonic generator. Also being developed are quantitative theories of absolute phase calibration for existing and proposed NVNA systems.

1.2.2 Nonlinear Modeling: Grey-Box Model Identification

The fidelity and precision of simulatable models of the radio frequency (RF) front-ends of communication, radio and sensor systems range from highest for circuit-level models, to intermediate for grey-box models, to coarse for black-box models.

Circuit-level models incorporate compact models of individual components and these are extracted using well-developed procedures requiring access to the component terminals. However, simulating system behavior using circuit-level models and digitally-modulated signals can require excessive run times. Also circuit schematic and layout information are often not available due to intellectual property (IP) protection reasons. Thus model-order reduction is necessary to yield a much simpler model that can be used in system investigations with limited loss in accuracy.

Black-box models capture the behavior of a network at external ports but have little relationship to the underlying physics. The main attributes of the black-box models are IP encapsulation, and being reduced-order system-level models, they enable rapid computation of the behavior of a system with digitally-modulated communication signals.

Grey-box models provide similar, but usually much more precise simulation than can be obtained using black-box models. Developing the structure of a grey-box model of unknown RF architecture requires physical insight. Fortunately a finite number of RF front-end architectures are used so that a suite of grey-box models can be developed to capture the fundamental linear and nonlinear interactions of most systems.

The type of problem for which this work is targeted is extracting the grey-box models of the RF front-end of undocumented devices for co-site interference simulation, for example, where two or more radios are operated in close proximity and performance is compromised by broad-band noise and the unintended active scattering of intermodulation (IM) components, both effects interfering with radio reception. The model identification problem is one of extracting a large number of parameters, while considering several candidate grey-box models in parallel. The complexity of grey-box models can be adjusted in response to both the level of fidelity required and the measured data available to fit the model.

A typical parameter extraction procedure uses an algorithm that seeks to find the global solution having the minimum error between the modeled and measured responses. However the global solution may not be a physically or intuitively satisfying solution. One reason is that in most RF systems, frequency shaping by bandpass filters and matching networks prevent direct observation of nonlinear effects. Also there are rarely enough external measurements available to unambiguously identify all of the model parameters. As a result, the modeling exercise is not straight-forward and all too often must proceed either by choosing alternative starting points for the model fitting algorithm, or by modifying the error cost function.

Past work on RF model identification has focused on the Wiener and Hammerstein model family, and slight variations of such structures. In all cases the procedures were developed specific to the model, using an ad-hoc higher-level identification algorithm, and relying on deterministic optimization methods which require an initial solution. The models are simple enough that an initial solution close to the optimums be easily found. Typically only one solution, the global optimum if it can be found, is presented.

Current research explores general identification methodology for more complex nonlinear models than have been used previously. One area of exploration is in generalized higher-level identification algorithm compatible with stochastic optimization methods, so that multiple local optimum solutions are available for selection by the user. Another area of exploration is in general purpose nonlinear simulation compatible with the parameter extraction procedure, to

relate the model parameters to the measured data which typically consists of the amplitude and phase of discrete tone responses to swept-power and swept-frequency one-tone and multi-tone excitations. In this way, model analysis and response calculation are automated. The overall research goal is to minimize the need for a human in-the-loop, so that user input is directed first at establishing several candidate grey-box models, establishing the measurement protocol, and then choosing the best solution at the end of the procedure.

1.3 Original Contributions

To achieve the aforementioned objectives, a number of original research initiatives were undertaken. The author's contributions to the field of RF and microwave engineering are summarized below.

1.3.1 Novel 1-Channel Receiver Based NVNA

No previous 1-channel receiver implementations exist for linear or nonlinear characterization. Such an implementation will offer a hardware and power efficient solution for on-chip self-characterization.

This work presents a 1-channel VSA-based NVNA used together with an AWG and an equivalent-time sampling oscilloscope, that does not require a multi-harmonic generator to establish an absolute phase reference. The quantitative theory of absolute phase calibration for this type of NVNA is also presented.

To mimic a synchronous multi-channel receiver, the signals of a multi-port DUT are sequentially sampled, with the excitation waveform repeated for each sampling duration. To maintain the same timing relation between sequentially sampled waveforms as if they were synchronously sampled, excitation waveforms are designed to be on a DFT grid that is commensurate with the AWG and the VSA sampler time base. In the post processing step, the entire VSA sampled waveform is divided into segments corresponding to each sequentially sampled DUT signal. The start and end of each segment can be determined, since the excitation duration and the

switching interval are specified in terms of the smallest period in the excitation waveform.

Once the 1-channel VSA can be made to behave like a multi-channel receiver, concepts from absolute phase measurement using broadband receivers can be applied with minor modifications. Namely, a multi-tone signal generated by an AWG replaces the multi-harmonic generator, and a nose-to-nose calibrated oscilloscope provides the reference measurements for VSA phase error correction.

1.3.2 Novel 4-Port VNA-based NVNA

The previous 4-port VNA-based NVNA requires a multi-harmonic generator as an absolute phase reference, and thus is limited in the narrowest tone-spacing that can be used in multi-tone testing [5]. The idea of using a relative phase reference has not been explored. The more readily available phase-locked signal generators can be used as a relative phase reference, and can tune to tone-spacings on the order of 1 Hz

This work presents a 4-port VNA-based NVNA used together with phase-locked signal generators and an equivalent-time sampling oscilloscope. The multi-harmonic generator is not required, and the associated tone-spacing limitation in multi-tone testing is removed. The quantitative theory of absolute phase calibration for this type of NVNA is also presented.

The concept of using a relative phase reference that is phase-locked to the excitation enables optimum use of different instruments for different parts of the measurement. The high dynamic range heterodyne receiver-based VNA is used for data acquisition during a power sweep, while the relatively low dynamic range sampling oscilloscope is used for its absolute phase calibration ability. Using each instrument to its advantage, narrow tone-spacing, wide bandwidth, and high dynamic range is achieved.

1.3.3 Grey-Box Model Identification Methodology

Past work on RF model identification has focused on the Wiener and Hammerstein model family. In all cases extraction procedures were developed specific to the model, using ad-

hoc higher-level identification algorithms, and relying on deterministic optimization methods. Deterministic local search methods depends on the starting point of the search and are prone to getting trapped in local optimum solutions, but may suffice for simple models where an initial solution close to the global optimum can be found. Typically only one solution, the global optimum if it can be found, is presented.

This work presents a general identification methodology for more complex nonlinear models than have been used previously. A general-purpose symbolic Volterra circuit simulator is used to automate the model analysis and response calculation. A generalized higher-level identification algorithm is developed to be compatible with stochastic optimization methods. A hybrid genetic algorithm (GA) optimizer is developed to specifically track local optimum solutions, and present the user with a number of distinct modeling solutions with comparable error metrics. This is particularly suited to modeling incompletely-observed RF systems, since an initial solution close to the optimums can no longer be easily found for complex models.

The main result is a reorganization of the way models are extracted for RF devices. Instead of a skilled user being involved throughout the model fitting process, user input is directed first at establishing several candidate grey-box models, and then establishing a measurement regime. The approach can also be adapted to fit several grey-box structures simultaneously, and can efficiently use distributed computing resources.

1.3.4 Direct Calculation Algorithm for Volterra Circuit Analysis

Compared to the well known NLTF method of Volterra circuit analysis where the NLTF is determined first before being used to evaluate the response, the direct calculation method can easily handle nonlinear circuits with multiple inputs (such as mixers) which would otherwise require tensor manipulation to find the NLTFs.

In the original presentation of the direct calculation method [6], an expression for the nonlinear current sources was only presented up to third order, and not in an algorithmic form suitable for implementation in a circuit simulator. In addition, only the lowest-order contribu-

tion to a response frequency was considered, so compression effects were not being accounted for.

This work presents the direct calculation method in an algorithmic form to handle arbitrarily high degree of device nonlinearity. The expression for nonlinear current sources is determined by recursively finding the lower-order IM response due to the controlling nonlinear variables. For the same response frequency, contributions of different order are differentiated, so compression effects can now be handled. Using this method, a symbolic Volterra circuit simulator is coded in Matlab.

With this recursive implementation, determining parameter dependence in model identification is easier than using the NLTF method, which otherwise requires analysis of the response expression. More importantly, the recursive implementation also determines the dependence of the modeled terminal response on the inter-stage responses which are not observed. By evaluating the inter-stage responses at the multiple optimum solutions found in the identification process, a physically or intuitively satisfying solution can be selected by a skilled and knowledgeable engineer with experience acquired with the type of system being modeled.

1.4 Dissertation Outline

Chapter 2 of this dissertation presents a literature review of nonlinear characterization setups and RF system behavioral models. First the principles of absolute phase measurement are explained, and existing NVNA implementations are described and compared to early approaches to phase measurement. Then several existing behavioral modeling approaches that have been found to be successful in modeling RF systems are described.

Chapter 3 presents two novel NVNA setups that do not need a multi-harmonic generator, and are implemented using common bench top instruments. First a 1-channel VSA-based NVNA is used together with an AWG and an equivalent-time sampling oscilloscope, and it is shown that linear and nonlinear characterization is possible using only a single channel receiver. To mimic a synchronous multi-channel receiver, a solution combining sequential sampling, exci-

tation design, and post processing is presented. Then a 4-port VNA-based NVNA, used together with phase-locked signal generators and an equivalent-time sampling oscilloscope, shows that a relative phase reference is useful for the power sweep part of the absolute phase measurement. The quantitative theory of absolute phase calibration for both types of NVNAs are also presented.

Chapter 4 presents a general identification methodology for more complex grey-box models than have been treated previously. Essential to the methodology is the generalized higher-level identification algorithm that is compatible with stochastic optimization methods. A hybrid genetic algorithm (GA) optimizer is developed to track multiple optimum solutions, and present the user with a number of distinct modeling solutions with comparable error metrics. As a case study, the model extraction procedure is applied to a multi-stage solid state power amplifier exhibiting long-term memory effects.

Chapter 5 presents a symbolic Volterra circuit simulator implementing the direct calculation method. First the various methods of Volterra circuit analysis is briefly reviewed. Then the recursive algorithm of the proposed direct calculation method is developed to handle arbitrarily high degree of device nonlinearity and compression effects lacking in the original work. This enhanced direct calculation method is then applied in the analysis of some example circuits.

Chapter 6 applies the grey-box behavioral model identification methodology in cosite interference prediction.

Chapter 7 contains a summary of the research performed and lists the significant results of this work.

1.5 Publications

1.5.1 Journal papers

1. J. Hu, J. Q. Lowry, K. G. Gard, and M. B. Steer, "Nonlinear Radio Frequency Model Identification Using A Hybrid Genetic Optimizer for Minimal User Intervention," Submitted

to *IET Microwaves, Antennas, & Propagation*, August 2010.

2. J. Hu, K. G. Gard, and M. B. Steer, "Calibrated Nonlinear Vector Network Measurement Without Using a Multi-Harmonic Generator," Accepted to *IET Microwaves, Antennas, & Propagation*, November 2010.

3. C. Saunders, C. Christoffersen, M. B. Steer, and J. Hu, "Inverse Singular Value Method for Enforcing Passivity in Reduced-Order Models of Distributed Structures for Transient and Steady-State Simulation," *IEEE Transactions on Microwave Theory & Techniques*, Vol. 59, No. 4, August 2011.

1.5.2 Conference papers

1. J. Hu, K. G. Gard, N. B. Carvalho, and M. B. Steer, "Dynamic Time-Frequency Waveforms for VSA Characterization of PA Long-term Memory Effects," *71st ARFTG Conf. Dig. Digest*, Jun. 2007.

2. J. Hu, K. G. Gard, N. B. Carvalho, and M. B. Steer, "Time-Frequency Characterization of Long-Term Memory in Nonlinear Power Amplifiers," *2008 IEEE MTT-S International Microwave Symposium*, Jun. 2008, pp. 269-272.

3. J. Hu, K. G. Gard, and M. B. Steer, "Extraction of Behavioral Models for Power Amplifiers with Memory Effects using Vector Signal Analysis and a Dynamic Time-Frequency Signal," *Government Microcircuit Applications Conference*, Mar. 2008.

4. J. Hu, K. G. Gard, and M. B. Steer, "Topology Assisted Extraction of Volterra-Based Behavioral Models for Power Amplifiers with Memory Effects," *Government Microcircuit Applications Conference*, Mar. 2009.

Chapter 2

Prior Research in Nonlinear Metrology and Modeling

2.1 Introduction

The general goal of modeling is to replicate certain behavior, using representations for the external stimulus and the system under study. In the process, knowledge is gained about physical mechanisms underlying the behavior, and the system representation discovered enables prediction of responses to different sets of circumstances. Models that can replicate the minutest observed detail under the most general circumstances, are often the most complex.

A linear system has sinusoids as eigenfunctions, and is uniquely represented by a linear transfer function. There is generally no eigenfunction for nonlinear systems that yield useful decomposition for arbitrary excitation and response signals. Different analysis approaches such as phase-plane analysis, theory of differential equations, Wiener-Volterra theory, and computer simulation, are required to gain distinct view points and insights into the modeling problem [7].

Thus for each nonlinear system, the response to each specific excitation is often studied on a case by case basis, using analytic formulation or computer simulation. For weakly nonlinear systems, the Wiener-Volterra formulation offers an extraction procedure to uniquely determine

the Wiener G-kernel [8], but the problem is finding the right instruments for signal generation and measurement.

To manage complexity, models are often extracted under conditions corresponding to their intended application, which is specific to each layer of abstraction in the communication system hierarchy as discussed in Section 1.1. For example, at the circuit implementation level, device engineers develop transistor models primarily to replicate the current-voltage characteristic determined from device physics simulation and DC I-V curve tracer measurement. In link-level design, circuit engineers develop circuit subsystem models primarily to replicate the response to realistic communication signal excitation. Subsystem models are developed from circuit-level simulation and from Vector Signal Analyzer (VSA) measurement.

However complex excitations used to replicate realistic operation conditions also complicate the measurement setup and the model extraction problem. The distortion effects produced by a nonlinear system are best viewed using frequency-domain measurement setups, and realistic excitations need to be approximated by periodic signals. For analysis using analytical formulation of a model, complex excitations entangle many model parameters in the response expression, and parameter extraction is complicated by the web of interdependence. Thus the three elements of behavioral modeling — model formulation, excitation design and measurement setup — are intertwined and must be considered together in forming a modeling solution.

Historically a practical approach is taken by approximating the realistic excitation using the simplest signal that is compatible with simple measurement setups. Then to capture the dominant behavior, models are often devised as a simple extension to the linear system and with clear parameter dependence to ease the parameter extraction process. For example, a single tone is often used to approximate the communication signal in narrowband systems. The early generation of behavior models [9] [10] [11] that were devised to replicate the nonlinear response under this type of excitation are reviewed in [2]. For the most widely used power series model extracted from 1-tone measurements, performance prediction with a modulated signal is possible using a statistical analysis approach [12] [13] [14].

To capture the more realistic bandwidth-dependent behavior in modern systems with wider communication bandwidth and channel allocations, the complexity of the excitation signal and model formulation are gradually increased. Typically a 2-tone is used to simulate the blocking performance in a receiver design, or adjacent channel power ratio (ACPR) performance in transmitter design. To approximate the real modulation signal or the interaction between different communication channels in the same band, a multi-tone excitation is used [15] [16]. The current generation of nonlinear characterization concepts and methods are reviewed in Section 2.2, and the related behavioral modeling approaches are reviewed in Section 2.3.

To determine whether a further increase in model complexity is required at various stages of model development, one needs to compare the simulated behavior of the model with the measured behavior of the device. The eventual goal is to extract a behavioral model that closely approximates the DUT's behavior under realistic operating conditions, using the simplest excitation signal and measurement setup possible.

2.2 Review Nonlinear Metrology

In characterizing a high-frequency nonlinear circuit, the incident and scattered port waves at measurement planes are synchronously sampled using multiple receiver channels with a common time base. Typically 4 receiving channels are needed to characterize a 2-port device as in Fig. 2.1. Steady-state frequency-domain characterization is especially suited to characterizing a narrowband nonlinear system, where the output nonlinear component may be orders of magnitude smaller than the linear component and cannot be viewed in the time domain.

Under periodic discrete tone excitation, each spectral component in the DUT output is described by a complex number. The magnitude is an absolute quantity describing power, while the phase is a relative quantity, describing the position in time relative to the time origin. For periodic signal, the position is unique modulo a period. The phase is an important quantity since all manner of cancelation effects depend on the phase of the interacting spectral

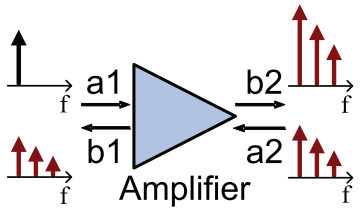


Figure 2.1: Characterizing the incident and scattered port waves using multiple synchronous receiver channels.

components. Phase is also more sensitive than magnitude in detecting phenomena such as the long-term memory effect or the IM distortion sweet-spot[1].

For any measurement equipment or setup, it is important for the measurement results to be repeatable within an acceptable degree of accuracy. It is also important that measurements made with different equipment be comparable. As an integral part of any measurement system, calibration ensures traceability to calibration standards. Power measurement has a well known calibration method, typically using commercially available power meters as the transfer standard. Phase measurement currently has no traceable standard.

For linear systems, the linear transfer function is the ratio of the input and output measurements under 1-tone excitation. The linear transfer function, along with criteria for stability and causality, uniquely characterize the linear system. For a linear DUT, the phase of the S-parameter is the relative phase of the scattered port waves relative to the incident wave.

Nonlinear systems generate additional frequency content as shown in Fig. 2.2. So except at the frequencies of the excitation tones, ratios cannot be used to characterize the nonlinear DUT. One approach adopted by the microwave community and under consideration at the National Institute of Standards and Technology (NIST) is the consensus phase standard [1], which ensures that everybody measures the same phase for a signal by adopting an arbitrary but agreed upon phase reference value. This approach known as the absolute phase measurement,

[1]The IM sweet-spot phenomenon appears a null when the amplitudes of IM products are plotted versus the power of the excitation. The null is due to the cancelation between nonlinear contributions of different order.

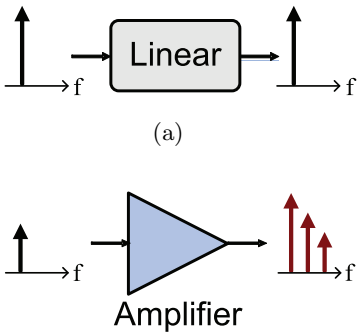


Figure 2.2: For (a) linear systems, the excitation can be used as the phase reference since the response is at the same frequency as the excitation, and their relative phase is the phase of the linear transfer function. For (b) nonlinear systems, the response contains additional spectral components that are not present in the excitation, and requires an alternative phase reference definition.

is described in Section 2.2.1, where port waves are characterized as signals. Another approach described in Section 2.2.3 is to find an alternative definition that relates the excitation and DUT outputs in an invariant and unambiguous way. This is more useful for DUT characterization as designers are often more interested in the phase changes caused by the DUT as a function of input power or tone-spacing.

2.2.1 Absolute Phase Signal Measurement

The consensus phase standard originates from the need to establish a phase reference for the output spectral components not present in the excitation, and the need to correct phase dispersion inherent in any receiver channel used to measure spectral components in the port waves.

For output spectral components not present in the excitation, the solution most commonly used is to create a phase reference signal by generating discrete tones on a harmonically spaced frequency grid that will contain all possible output spectral components. To ensure repeatable measurement, the reference signal must uphold a fixed but unknown relative phase relation between the tones. Such a multi-harmonic generator is shown in Fig. 2.3 and comprises a

nonlinear device, typically a step recovery diode followed by a power amplifier [17], driven at the fundamental frequency. Thus the only arbitrary phase value in the generator output is that of the fundamental tone, the consensus phase reference value [18].

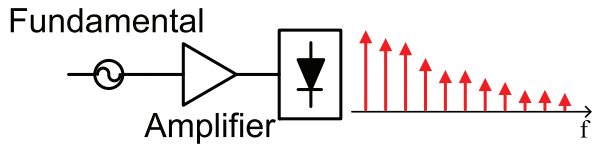


Figure 2.3: Harmonic phase reference generator showing equally spaced tones in the output spectrum.

To find the systematic phase dispersion error in a receiver channel, the solution is to characterize the unknown phase relation of a multi-harmonic generator using an error free oscilloscope, then compare the result with that of the practical receiver. To characterize a multi-harmonic generator, an electro-optical sampling process, which is frequency independent up to hundreds of gigahertz, can be used. The equivalent-time sampling oscilloscope can also be used, but must be corrected first using the nose-to-nose calibration method which finds the impulse response and correct the phase dispersion up to about 20 GHz [19]. The characterized multi-harmonic generator can now be used as the transfer standard for phase measurements.

Any measurement system capable of synchronous multi-port characterization of a DUT's signal spectral components, and with relative and absolute calibration in power and phase, are termed a nonlinear vector network analyzer (NVNA) or a large signal network analyzer (LSNA).

Using a Broadband Receiver

Fig. 2.4 shows the NVNA system described in Reference [1]. It can be generalized to use broadband receivers such as an oscilloscope-based or a sub-sampling mixer-based RF front-end,

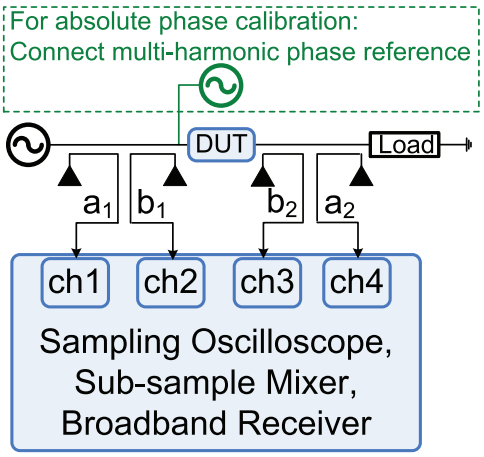


Figure 2.4: NVNA setup using a broadband receiver after [1].

which can characterize all spectral components in the signal at once.

Calibration of the NVNA requires the establishment of calibrated reference planes. Full calibration requires linear (or relative) calibration, power calibration, and absolute phase calibration. The calibration between the ideal (internal) measurement planes and the actual VNA measurement planes, uses an 8-term linear error model [1]:

$$\begin{bmatrix} a_1(f) \\ b_1(f) \\ a_2(f) \\ b_2(f) \end{bmatrix} = K(f) \begin{bmatrix} 1 & e_{12}(f) & 0 & 0 \\ e_{21}(f) & e_{22}(f) & 0 & 0 \\ 0 & 0 & e_{33}(f) & e_{34}(f) \\ 0 & 0 & e_{43}(f) & e_{44}(f) \end{bmatrix} \begin{bmatrix} A(f) \\ B(f) \\ C(f) \\ D(f) \end{bmatrix} \tag{2.1}$$

where the port waves at the DUT planes are denoted as a_1, b_1, a_2 and b_2, the raw receiver measurements are denoted A, B, C and D, and the e's are the coefficients of the error model. In the creation of an error model for a VNA, the actual receiver measurements and the port waves at the ideal measurement planes are assumed to be linearly related. Thus at each frequency, a matrix containing error terms $e_{ij}(f)$ relates the actual receiver measurements and the port waves at the ideal measurement planes. For absolute phase calibration of a NVNA, an additional

factor $K(f)$ is added to the error model to correct the absolute power and phase. In (2.1) the seven e's provide linear (or relative) calibration, the magnitude of K provides power calibration, and the phase of K provides phase calibration. This linear error model is determined at each intermodulation frequency of interest. Each term of the error model is a complex number, and in particular the error term $K(f)$ equals $|K(f)|e^{j\theta_K(f)}$.

Linear calibration determines the 7 terms in the 4×4 matrix in (2.1) using a modified version of one of many procedures developed for linear VNAs [1], [20]. After linear correction to the DUT plane using the linear error model, only the relative power and phase between the port waves at each frequency have the correct values. The relative nature of this calibration is indicated by the leading 1 in the matrix. For absolute calibration, one of the port waves must be corrected in absolute power and phase, so that the other port waves can be corrected accordingly.

Absolute power calibration determines the magnitude of the error term $K(f)$, $|K(f)|$, and absolute phase calibration determines the angle of $K(f)$, $\theta_K(f)$. Analytical descriptions for determining $|K(f)|$ using a power meter and $\theta_K(f)$ using a phase transfer standard while accounting for the effects of mismatch are given in [1]. In determining $\theta_K(f)$, which accounts for errors due to the receiver phase dispersion, a multi-harmonic generator output is characterized using two of the receivers at port 1.

The outline of the calibration and error correction process is as follows

1. Establish the linear error model (see (2.1)).

2. Establish the power error $|K(f)|$.

3. Establish the phase errors $\theta_K(f)$. $\theta_K(f)$ is the relative phase difference at each frequency f, between the original nose-to-nose calibrated oscilloscope measurement of the multi-harmonic generator output, and phase of the multi-harmonic generator output measured by receivers (CH1-CH4) at the DUT plane.

After determining the errors from calibration measurements, the DUT measurements are cor-

rected as follows:

1. Linear correction to the DUT plane using the linear error model.

2. Absolute power correction at the DUT planes using $|K(f)|$.

3. Absolute phase correction at the DUT planes uses $\theta_K(f)$.

4. Align different data sets to a common time reference.

What is not mentioned in [1] is that the original nose-to-nose calibrated oscilloscope measurement of the multi-harmonic generator output, and the current flawed receiver measurement, are performed at different times. Noting that the arbitrary phase of the fundamental tone in the multi-harmonic generator, the phase measured by the flawed receiver cannot be directly used to find $\theta_K(f)$, but must be first aligned to a common time origin as the characterization result of the multi-harmonic generator. The common time origin is the arbitrary but agreed upon phase of the fundamental tone in the multi-harmonic generator, also known as the consensus phase reference. The alignment process is briefly described in Section 2.2.2. Once the receiver phase error is determined, all DUT measurements can be corrected. Note that the multi-harmonic generator is only used to correct the receiver phase error, but not relied upon to establish phase reference for output spectral components not present in the excitation.

The corrected measurement phase can already be considered to be absolute in the sense that if the DUT has an exact mathematical representation such as a power series, the simulated DUT signals under the measured incident excitation will equal the measured scattered port waves. This capability is essential in design verification.

However the consensus phase standard requires that all parties measure the same phase for the same signal, and that different measurement data sets taken at different times can be compared. Due to imperfect phase lock between the excitation source and the measurement receiver, the phase of the excitation will slowly drift in time, and thus the phase of the measured data will also change in time. As long as the phase relation of the tones in the excitation is

maintained, this drift can be removed by applying the same alignment process [21] [22]. By agreeing upon the reference measurement data set, what is known as the 'absolute phase' signal measurement is achieved.

The alignment process is typically applied to incident excitations of the same type, such as 2-tones with the same tone-spacing, amplitude and phase profile. In typical DUT nonlinear characterization, the engineer is interested in the phase change as a function of input power or tone-spacing. For a power sweep with a fixed tone-spacing and phase profile, the alignment process can still be applied to excitations of different powers by scaling to a common amplitude. Since power sweep requires switching attenuators in the signal generators, an additional requirement is that the attenuator phase contribution be equal for all the excitation tones. Then for a chosen reference data set, the aligned phase over the power sweep is still absolute in the sense that it uniquely characterizes the DUT for the given type of excitation.

However for a sweep over tone-spacing with a fixed power and phase profile, the alignment process cannot be applied to excitations of different tone-spacings, and thus the absolute phase measured becomes less intuitive for design purposes. Section 2.2.3 describes invariant and unambiguous phase relation definitions between the excitation and DUT outputs, which overcomes the alignment limitations in handling data over tone-spacing sweeps.

One early NVNA used a 4-channel equivalent-time sampling oscilloscope [1], which is a broadband receiver. For a periodic signal, the sampling oscilloscope is triggered at regular intervals, and samples are taken after a programmed delay. The waveform is constructed from samples taken over many periods. While the acquisition time of each sample is on the order of microseconds, equivalent-time sampling allows measurement of repetitive signals with bandwidth of 20 GHz. The accuracy of the measured signal is affected by the timebase error and the impulse response of the sampling channel [23]. The dominant sources of the timebase error are the distortion in the oscilloscope's internal clock, the jitter in the trigger circuitry and the time delay generator, and the drift in timebase between different measurements [24]. A timebase correction software is available from National Institute of Standards (NIST) to correct

the sampled waveform. There is also jitter in the oscilloscope's internal clock and in the samplers of different channel, but these are minor factors. The impulse response of the sampling channels can be corrected using the nose-to-nose calibration procedure [17] after time base correction. The dynamic range is around 60 dB depending on the number of samples acquired. However the data acquisition speed was considered too slow for industrial applications, so alternative approaches was taken.

One mainstream variation of the modern NVNA commercially available uses a 4-channel sub-sampling mixer-based RF front-end [1] taken from the microwave transition analyzer (MTA). The MTA is essentially a synchronous 2-channel real time oscilloscope with a much faster data acquisition time [25]. A multi-harmonic generator with a 10 MHz fundamental is used for absolute phase calibration. The comb signal derived from another multi-harmonic generator drives the sampler which downconverts spectrums around each frequency grid point to baseband. The dynamic range is around 60 dB, and the bandwidth is around 20 GHz.

Using Narrowband Receiver

For a narrowband tuned receiver such as a super-heterodyne mixer-based RF front-end, only a select range of spectral components can be captured at once. Since different spectral components of the multi-harmonic generator outputs are measured at different times, the phase of the fundamental tone may be unknown and the alignment process cannot be used to correct the phase error in any of the receiving channels.

The solution is to concurrently measure, along with the DUT signals, the multi-harmonic generator output using an additional receiving channel for reference. The measured phase difference between the multi-harmonic generator output and DUT signals is added to the previously characterized phase of the multi-harmonic generator to arrive at the phase of the DUT signals. By agreeing upon the consensus phase reference value of the multi-harmonic generator fundamental tone, the 'absolute phase' measurement of the DUT signals is again achieved. Note that the multi-harmonic generator is relied upon to correct the receiver phase error, and to establish

phase reference for output spectral components not present in the excitation.

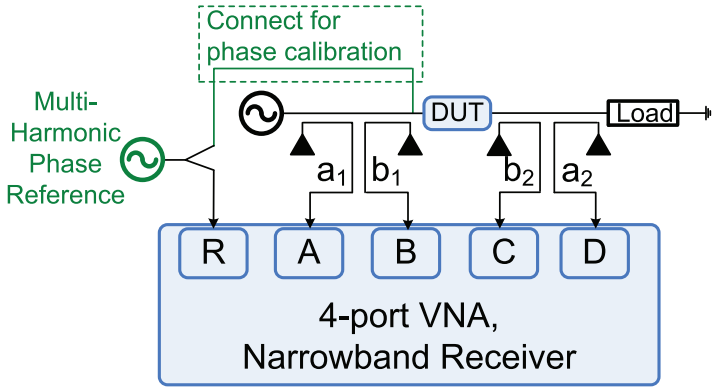


Figure 2.5: NVNA setup using a narrowband receiver.

Fig. 2.5 shows the NVNA setup in [5]. It can be generalized to use narrowband receivers. The outline of the calibration and error correction process is as follows

1. Establish the linear error model (see (2.1)).
2. Establish the power error $|K(f)|$.
3. Establish the phase errors $\theta_K(f)$. $\theta_K(f)$ is the relative phase difference at each frequency f, between phase measured by the reference receiver (R), and phase measured by each of the other receivers (A-D) at the DUT plane.

Since the reference receiver (R) serves as a phase transfer mechanism between the sequentially sampled spectral components, the relative phase error of the phase transfer mechanism as represented by $\theta_K(f)$ must be determined. This calibration step was not described in [5], but is obviously necessary since the reference receiver and the other receivers in a VNA give different measurements for the same signal. After determining the errors from calibration measurements, the DUT measurements are corrected as follows:

1. Linear correction to the DUT plane using the linear error model.

2. Absolute power correction at the DUT planes using $|K(f)|$.

3. Absolute phase correction at the DUT planes uses $\theta_K(f)$, with the phase difference between the concurrently measured multi-harmonic generator output and the DUT signals at the DUT plane, and with the previously characterized phase of the multi-harmonic generator output.

4. Align different data sets to a common time reference.

This variation of the modern NVNA uses a commercially available 4-port vector network analyzer (VNA) with a reference receiver architecture [5], which has narrowband tuned receivers with an instantaneous bandwidth of 250 kHz. Typically only one discrete tone is measured at a time. The reference receiver (R), which is the 5th channel normally used to sample the 1-tone excitation in S-parameter measurements, is used to concurrently measure the corresponding discrete tone in the multi-harmonic generator output. The dynamic range is around 85 dB, and an effective bandwidth of 20 GHz is achieved by sequentially measuring the spectral components of interest.

2.2.2 Signal Alignment

For example consider a sinusoid $\cos(2\pi ft + \phi)$, the time origin is at time 0. Sampling a period of this waveform starting at time 0, the result of a DFT will show a phase value of ϕ at the frequency f. If sampling is started at time t_0, the DFT result will show a phase value of $2\pi ft_0 + \phi$. The expression of the sampled waveform relative to the new time origin is $\cos(2\pi ft^{'} + (2\pi ft_0 + \phi))$ via a change of variable $t^{'} = t - t_0$. The goal of signal alignment is to find the time delay t_0 between the two sampled waveforms.

Assuming one sampled waveform is a time-delayed version of the other, the time delay can be found either in the time domain or the frequency domain. Time-domain alignment maximizes the cross-correlation between the two signals. Frequency-domain alignment, also known as phase detrending [26], minimizes the least squared error between the phase of the

spectral components in the two signals. Either signal can then be adjusted by this time delay in the time- or frequency-domain.

By the assumption, a distorted signal cannot be aligned to the undistorted version without incurring increasing error in the time delay estimate with increasing distortion. In a 1-channel receiver, the incident and scattered port waves are sequentially captured at different times, so the input cannot be aligned to the nonlinear DUT output. In a synchronous multi-channel receiver, the incident and scattered port waves are sampled concurrently. To compare measurements made at different times, the incident excitations are aligned to find the time delay, and all scattered port waves are adjusted by this time delay.

Signal alignment applies to general signals, and especially to periodic discrete tone signals with or without a fundamental tone present. This approach incurs lower error on the aligned phase than the other approach described in Section 2.2.4 [27].

2.2.3 Time-Invariant Phase Network Measurement

In DUT characterization, designers are most often interested in phase changes caused by the DUT as a function of input power or tone-spacing. As discussed in Section 2.2.1, alignment has trouble handling data over tone-spacing sweeps. Unlike the alignment process which is typically applied to incident excitations, the current approach seeks to find unambiguous phase relation between the excitation and DUT signals.

For linear DUT measurements under 1-tone excitation, the relation between the incident and scattered port waves are contained in the S-parameter as a ratio. For nonlinear DUT measurements under multi-tone excitation, time-invariant phase relation definitions were found [28] [29] [30] [31]. Briefly described in Section 2.2.4, these relations linearly transform the phases of the excitation and DUT signals into the corresponding time-invariant phase values. These linear transform can also be viewed as mathematically creating a phase reference from the tones in the incident excitation. Using this phase reference, the relative phase values computed are the time-invariant phase of the DUT signals.

As the last step of the absolute phase measurement, the time-invariant phase relations are applied in place of the alignment process. However for wideband multi-tone excitations with the number of tones much greater than two, the time-invariant phase definition amplifies the noise variance in the measured data, and creates phase uncertainties of $\pm 5^\circ$ [27]. For the more widely used 2-tone excitation, the amplified phase uncertainty is less than $\pm 0.5^\circ$, and the time-invariant phase approach is suitable.

For general excitation with a none constant phase profile, the aligned phase does not equal the time-invariant phase. The aligned phase contains the phase profile of the excitation, while the time-invariant phase will only contain the phase change caused by the DUT. They are equal only for excitation using constant zero phase profile. During a power sweep, an additional requirement is that the signal generators contribute equal phase to the excitation tones. For example using the 2-tone excitation with a given phase profile, the time-invariant phase for the two fundamental tones in the transmitted wave at frequencies f_1 and f_2, is equal to their relative phase referenced to the excitation. This relative phase is just the phase of the ratio normally computed for S-parameter. In circuit simulation using an exact representation of the DUT, by assigning the 2-tone excitation with zero phase, the two transmitted fundamental tones will have the time-invariant phase value. This correspondence between measurement using excitation with a given phase profile and simulation using excitation with an all zero phase profile, also holds for the IM tones.

(A) Relative Phase Measurement: Software Reference Nonlinearity

In the early applications of these definitions, the measurement system often lacks the elements required for absolute phase measurement, namely the full relative calibration of the test set and the absolute phase calibration of the receiver. The only requirement is that the incident excitation and DUT signals be synchronously sampled in multiple receiving channels, since the tones in the incident excitation are used to mathematically create a phase reference.

In [29], only a 1-channel vector signal analyzer (VSA) captures the DUT transmission

response under 2-tone excitation. A VSA has a narrowband tuned receiver such as those in a VNA, but with a broader instantaneous bandwidth around 45 MHz, and a comparable dynamic range of 75 dB. The phase reference is mathematically created from the fundamental tones at frequencies f_1 and f_2 in the output instead of the input. This simplification is only valid for DUTs in the weakly nonlinear region of operation, where the output fundamental tones is dominated by the linear response that is relatively constant over power and tone-spacing sweeps. This effectively only create a constant offset in the mathematically created phase reference value. In that application, the goal was to optimize a multi-stage amplifier design, where the phase strongly affect IM cancelation and the final output distortion level.

In [31], a 2-channel MTA synchronously captures the incident 2-tone excitation and the DUT transmission response as shown in Fig. 2.6. The test set is only calibrated with respect to the through response, and the receiver response is assumed to be frequency independent. This method can be applied to multi-tone excitation of correlated type which uses phase-locked signal generators or arbitrary waveform generators (AWG), uncorrelated type which uses frequency-locked signal generators, and incommensurate type which uses independent signal generators set to frequencies not on the receiver discrete fourier transform (DFT) grid. The computed time-invariant phase can be averaged to reduce the measurement noise, with the correlated case requiring the least amount of averaging, and the incommensurate case requiring the most due to spectral leakage. The reported phase uncertainty is $\pm 2.5^{\circ}$. Interestingly, an ensemble of correlated multi-tones with random phase can approximate the uncorrelated case [32]. This amounts to a sequence of 30 to 50 measurements in place of a uncorrelated one.

(B) Relative Phase Measurement: Hardware Reference Nonlinearity

In other applications hardware is used to create the phase reference. One approach, shown in Fig. 2.7, uses a reference nonlinearity such as a 'golden diode' or a mixer [33] [34] [35]. Another approach, shown in Fig. 2.8, which is referred to here as the signal injection approach, uses signal generators [36] [37] [38]. Once the phase reference signal is created, the relative phase of

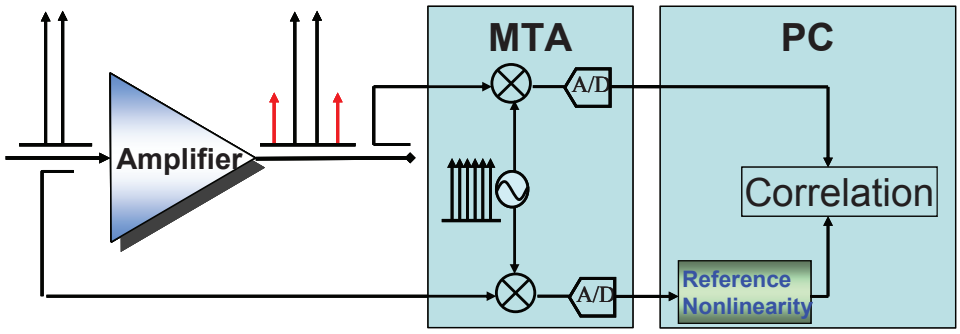


Figure 2.6: Relative phase measurement system that synchronously captures the input and the output using a 2-channel MTA, then generates the phase reference from the input using a software reference nonlinearity.

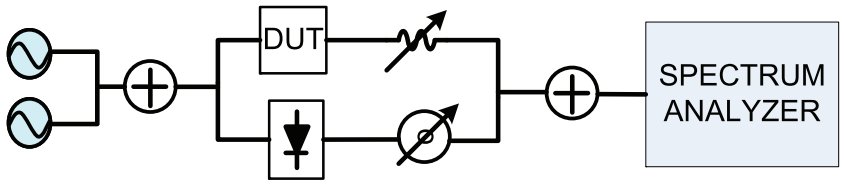
each spectral component can be measured using either a cancelation method which is suitable for a 1-channel magnitude only receiver such as a spectrum analyzer, or a synchronous 2-channel measurement method which is suitable for instruments such as VNA or MTA. Typically, only one spectral component of the DUT transmission response and the reference nonlinearity is considered for cancelation or synchronous measurement.

The hardware reference nonlinearity is the forerunner of the multi-harmonic generator used in absolute phase measurement, but the early applications lack the full relative calibration of the test set and the absolute phase calibration of the receiver. In the signal injection approach, the reference signal generator can only create the phase reference signal for one spectral component at a time, and has a phase value that vary randomly as the frequency is tuned. Thus, unlike the hardware reference nonlinearity, this idea cannot be adapted to absolute phase measurement.

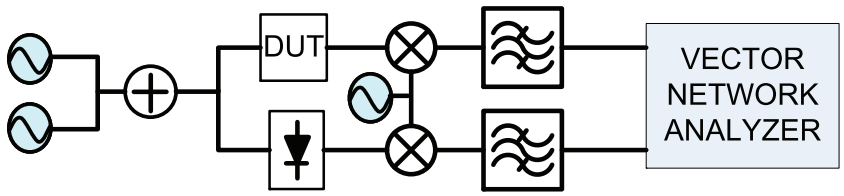
(B1) Reference Nonlinearity: Cancellation or Phase Measurement

The reference nonlinearity approach applies the same excitation to both the DUT and the reference nonlinearity. Thus the multi-tone excitation can be generated in a correlated, uncorrelated, or incommensurate way.

In [33] using 2-tone excitation, the same incident excitation is applied to the DUT and

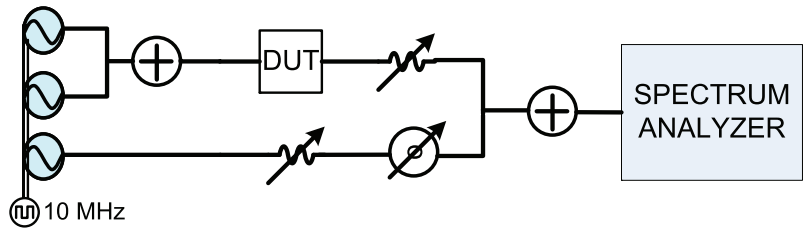


(a)

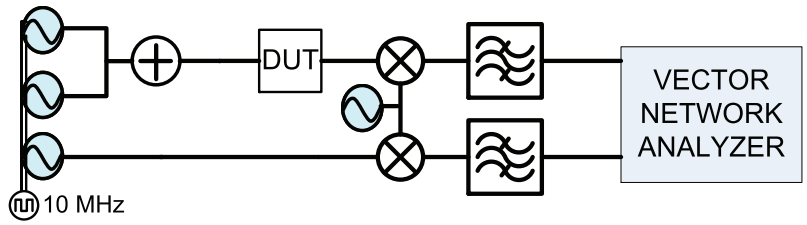


(b)

Figure 2.7: Relative phase measurement system that generates the phase reference from the input using a hardware reference nonlinearity, then measure the relative phase using either: (a) cancelation method; or (b) synchronous measurement method.



(a)



(b)

Figure 2.8: Relative phase measurement system that generates the phase reference from a phase locked signal generator, then measure the relative phase using either: (a) cancelation method; or (b) synchronous measurement method.

the hardware reference nonlinearity. The IM product of interest in the two branches of the measurement system are combined at RF frequencies. The magnitude of this vector addition is observed on a spectrum analyzer. The signal in the reference nonlinearity path is adjusted in magnitude and phase to achieve maximum cancellation. The phase shift value is the relative phase, which has an uncertainty determined by the degree of cancellation.

In [34] the cancellation is carried out at IF frequencies. First the excitation and the DUT output are coherently downconverted. Then the downconverted excitation is applied to the reference nonlinearity, and combined with the downconverted DUT output.

In [35] the signals in the two branches are not canceled, but synchronously measured using a VNA to obtain the relative phase. This technique was called spectral filtering, since the desired spectral component is selected via a filter. Suitable bandpass filters is used to satisfy the tone-spacing requirement. Since only one tone is selected by the filter, the full dynamic range of the VNA can be used, depending on the filter rejection characteristic.

(B2) Signal Injection: Cancellation or Phase Measurement

Here separate signal generators are used for excitation and generating the phase reference signal. Since time-invariant relative phase is only defined between correlated tones at the same frequency, these separate signal generators must be phase-locked. Thus the multi-tone excitation must be generated in a correlated way.

In [36] [37] the relative phase is measured using the cancellation method. In [37] the relative phase over a power sweep is measured using a wideband feed-forward cancellation system with 90 dB dynamic range. During the power sweep, the switching attenuator in the signal generator contributes systematic error to the measured relative phase and must be calibrated out. The phase error determined by the amount of cancellation is less than 0.5° typically and 2° in the worst case.

In [38] the signals in the two branches are not canceled, but synchronously measured using a VNA to obtain the relative phase. As in the spectral filtering method used with the reference

nonlinearity, the desired spectral component is selected via a filter.

2.2.4 Time-Invariant Phase Relation Definitions

Depending on the type of excitation, several types of linear transform definitions were previously found to yield time-invariant phase from the measured excitation and DUT signals [28] [29] [30] [31]. The essential point of the devised linear transform is to create another signal at the same frequency to cancel the frequency dependence, and thus canceling the time dependence.

Two definitions with low phase variance are described below. The fundamental alignment method is suitable for 1-tone excitation, and the time-zero cancelation is suitable for 2-tone excitation.

Fundamental Alignment for 1-Tone Excitation

Consider a single-tone signal at the 1st harmonic of frequency f, the phase of the waveform sampled at time t_0 is $\Phi_1(t_0) = 2\pi f_1 t_0 + \phi_1$ where the subscript indicates the harmonic. Consider another signal at the kth harmonic of frequency f, the phase of the concurrently sampled waveform is $\Phi_k(t_0) = 2\pi f_k t_0 + \phi_k$. The time-invariant phase $\phi_k^{'}$ for the harmonic signal is defined as

$$\begin{aligned} \phi_k^{'} &= \Phi_k(t_0) - k\Phi_1(t_0) \\ &= (2\pi f_k t_0 + \phi_k) - k(2\pi f_1 t_0 + \phi_1) \\ &= \phi_k - k\phi_1 \end{aligned} \tag{2.2}$$

which is invariant to the sampling time t_0.

When applied to absolute phase measurement data, the single-tone signal sampled at time t_0 is the incident excitation $\cos[2\pi f_1 t + (2\pi f_1 t_0 + \phi_{i,1})]$, and the concurrently sampled scattered DUT signals has harmonic components $\cos[2\pi f_k t + (2\pi f_k t_0 + \phi_{o,k})]$. Using the incident

excitation to create the phase reference, the time-invariant phase defined are

$$\phi'_{i,1} \quad = \phi_{i,1} - \phi_{i,1} \quad = 0 \tag{2.3}$$

for the incident excitation and

$$\phi'_{o,k} \quad = \phi_{o,k} - k\phi_{i,1} \tag{2.4}$$

for the scattered DUT signals.

Time-Zero Cancellation for 2-Tone Excitation

The definition presented here is equivalent but different from the notation used in [27]. Consider a 2-tone signal at frequencies f_1 and f_2, the phase of the tones sampled at time t_0 are

$$\Phi_{1,0}(t_0) = 2\pi f_1 t_0 + \phi_{1,0} \tag{2.5}$$

$$\Phi_{0,1}(t_0) = 2\pi f_2 t_0 + \phi_{0,1} \tag{2.6}$$

respectively. Consider the mixing product at frequency $f_{k,l} = kf_1 + lf_2$, the phase of the concurrently sampled tone is $\Phi_{k,l}(t_0) = 2\pi f_{k,l} t_0 + \phi_{k,l}$. The time-invariant phase $\phi'_{k,l}$ for the mixing product is defined as

$$\begin{aligned} \phi'_{k,l} \quad &= \Phi_{k,l}(t_0) - [k\Phi_{1,0}(t_0) + l\Phi_{0,1}(t_0)] \\ &= (2\pi f_{k,l} t_0 + \phi_{k,l}) - [k(2\pi f_1 t_0 + \phi_{1,0}) + l(2\pi f_2 t_0 + \phi_{0,1})] \\ &= \phi_{k,l} - [k\phi_{1,0} + l\phi_{0,1}] \end{aligned} \tag{2.7}$$

which is invariant to the sampling time t_0. For multi-tone signal, the transform can be generalized in a similar way.

When applied to absolute phase measurement data, the 2-tone signal is the incident excita-

tion indicated using subscript i, and the scattered DUT signals containing the mixing products are indicated using subscript o. Using the incident excitation to create the phase reference, the time-invariant phase defined are

$$\phi'_{i,(1,0)} = \phi_{i,(1,0)} - \phi_{i,(1,0)} = 0 \tag{2.8}$$

$$\phi'_{i,(0,1)} = \phi_{i,(0,1)} - \phi_{i,(0,1)} = 0 \tag{2.9}$$

for the incident excitation and

$$\phi'_{o,(k,l)} = \phi_{o,(k,l)} - \left[k\phi_{i,(1,0)} + l\phi_{i,(0,1)}\right] \tag{2.10}$$

for the scattered DUT signals.

Reference Nonlinearity Point of View

As pointed out in Section 2.2.3, the linear transform used to define the time-invariant phase can also be viewed as mathematically creating a phase reference from the tones in the incident excitation. An equivalent way to create this phase reference is by using a software reference nonlinearity if the excitation is captured with the DUT signals [28] [29] [30] [31], or a hardware reference nonlinearity if the excitation is not captured [33] [34] [35] [36] [37] [38]. Consider a software reference nonlinearity $x + x^3$ under 2-tone excitation, and using the same notation as above, the phase of a mixing product at $f_{k,l}$ is

$$k\phi_{1,0} + l\phi_{0,1} \tag{2.11}$$

which is same as in 2.7.

For uncorrelated multi-tone excitation generated by multiple frequency-locked CW generators, random phase variation of the independent fractional synthesizers only have a particular time-varying phase relationship on average, which significantly reduces the level of spurious

tones. Treating each measured phase in 2.7 as a random process, additional measurements and averaging operations are required when applying the time-invariant phase definition. For time-invariant nonlinear systems, this input output phase relation is expected to be a stationary process with constant ensemble average independent of time

$$E[\phi'_{k,l}] \quad = E[\phi_{k,l}] - (kE[\phi_{1,0}] + lE[\phi_{0,1}])\,. \tag{2.12}$$

In theory, for correlated multi-tone, the time-invariant phase can be calculated from a single acquisition of the input and output samples. In practice, averaging may also be needed to average out random measurement noise [31].

The incommensurate multi-tone are discrete tones with frequencies $\{w_1, ..., w_k\}$ such that

$$m_1 w_1 + m_2 w_2 + ... + m_k w_k = 0 \iff m_1 = m_2 = m_k = 0. \tag{2.13}$$

Being an aperiodic signal, the tones will be off the DFT frequency grid for any windows of the input and output samples captured, and the spectral leakage will create error in the phase. Consider that for the uncorrelated multi-tone, the random phase has the effect as randomizing frequency and moving the tones off the DFT frequency grid, the time average operation can be applied here to the incommensurate multi-tone. This method seemed to work well in [31], achieving good phase error bound.

Higher Order Statistics (HOS) Point of View

The averaged time-invariant phase can also be found via the frequency-domain version of the higher order cross-correlation between the input and output [31]. For 2-tone input, the ensemble averaged fourier transform of the higher order cross-correlation is

$$S_{yxxx}(kw_1 + lw_2) \quad = E[Y(kw_1 + lw_2)X^*(w_1)^k X^*(w_2)^l] \tag{2.14}$$

where X denotes the fourier transform of the signal used to create the phase reference, and Y denotes the fourier transform of the signal containing the mixing product. For $k < 0$ or $l < 0$, X is not conjugated. The phase of the higher order cross-correlation

$$\phi'_{kw_1+lw_2} = \phi_{kw_1+lw_2} - (k\phi_{w_1} + l\phi_{w_2}) \tag{2.15}$$

matches exactly (2.7).

The Q-tone case can be generalized as

$$\begin{aligned} &S_{y\ldots x}\left((m_1 - m_{-1})w_1 + \ldots + (m_Q - m_{-Q})w_Q\right) = \\ &E[Y\left((m_1 - m_{-1})w_1 + \ldots + (m_Q - m_{-Q})w_Q\right) \\ &X^*(w_1)^{m_1} X(w_1)^{m_{-1}} \ldots X^*(w_Q)^{m_Q} X(w_Q)^{m_{-Q}}] \end{aligned} \tag{2.16}$$

where the general mixing product frequency is denoted as

$$\vec{w} = [-w_Q, \ldots, -w_1, w_1, \ldots, w_Q] \quad : \text{input frequency vector} \tag{2.17}$$

$$\vec{m} = [m_{-Q}, \ldots, m_{-1}, m_1, \ldots, m_Q] \quad : \text{frequency mix vector} \tag{2.18}$$

$$\vec{m} \cdot \vec{w} = (m_1 - m_{-1})w_1 + \ldots + (m_Q - m_{-Q})w_Q. \quad : \text{mixing product frequency} \tag{2.19}$$

NLTF Point of View: Magnitude and Phase Reference

The reference nonlinearity idea for defining a phase reference, can be extend to define a magnitude reference, which is the magnitude of the mixing product produced by the reference nonlinearity [39]. The nonlinear transfer function (NLTF), which is the n-th order Volterra kernel in the frequency domain, is found as the response at frequency $(w_1 + \ldots + w_n)$ to n

incommensurate tones at the set of frequencies $\{w_1, \ldots, w_n\}$

$$x(t) = \frac{1}{2}\sum_{q=1}^{Q} x_q(t) \tag{2.20}$$

$$= \frac{1}{2}\sum_{q=-Q, q\neq 0}^{Q} X(w_q)e^{jw_q t} \tag{2.21}$$

The ensemble averaged NLTF can be expressed in the HOS notation

$$H_n(w_1, w_2, \ldots, w_n) = \frac{2^n}{n!} \frac{Y(w_1 + w_2 + \ldots + w_n)}{X(w_1)\ldots X(w_n)}$$

$$= \frac{2^n}{n!} \frac{Y(w_1 + w_2 + \ldots + w_n)}{X(w_1)\ldots X(w_n)} \frac{X(w_1)^*\ldots X(w_n)^*}{X(w_1)^*\ldots X(w_n)^*} \tag{2.22}$$

$$\bar{H}_n(w_1, w_2, \ldots, w_n) = \frac{2^n}{n!} \frac{S_{yx_1\ldots x_n}(w_1 + w_2 + \ldots + w_n)}{S_{x_1\ldots x_n x_1\ldots x_n}(w_1 + w_2 + \ldots + w_n)}. \tag{2.23}$$

In practice, an output mixing product at any frequencies other than $(w_1 + \ldots + w_n)$ will contain multiple frequency mix vectors and contributions from multiple orders. Thus while NLTF is useful when applied in characterizing the mixing product measurement, the mixing product measurement cannot uniquely determine the nonlinear system except in the incommensurate case.

2.2.5 Current Research

One issue with existing absolute phase measurement setups are their reliance on a multi-harmonic generator. Such a phase reference generator may not be readily available on the bench top in practice, or more importantly, impose tone-spacing limitation in multi-tone testing, which makes characterization at narrow tone-spacings as often required for memory behavior impossible. The multi-harmonic generator is driven at the fundamental and must generate harmonic tones with enough power up to the maximum frequency of the receiver to be measured with acceptable signal to noise ratio (SNR). The power of the fundamental driving signal cannot be simply increased, since these lower harmonic tones will saturate the receiver's front-end. A

microstrip filter can be designed to shape the generator output power and phase spectrum to lower its peak to average ratio (PAR). Still it is difficult to meet a typical receiver bandwidth of 20 GHz without placing a lower limit on the fundamental frequency. The commercially available type of harmonic generator is driven at 10 MHz, which limits the narrowest tone-spacing that can be used in multi-tone testing to 10 MHz. Other experimental types may exist, but are not readily available to the practicing engineer. The previous 4-port VNA-based NVNA [5] has a dynamic range around 80 dB, but is limited in the narrowest tone-spacing which is often required for characterizing the long-term memory behavior in power amplifiers. The sub-sampling mixer-based NVNA [17] is not limited in the narrowest tone-spacing, but has a lower dynamic range around 60 dB.

Another issue that hinders the wide adoption of the absolute phase concept in everyday design practice is that a commercially available NVNA constitutes a major capital investment. The more readily available bench top instruments are the arbitrary waveform generators (AWGs), spectrum analyzers, oscilloscopes, downconverters, and analog to digital converter (ADC) evaluations boards. It may be a lower cost and more practical solution to build an absolute phase measurement system using these hardware. Typically a single channel receiver such as a VSA can be built rather easily, compared to the four or five synchronous channels required in existing NVNAs. In the foreseeable future, self-tuning and self-healing concepts will likely to be applied to RF front-end architectures. This will require on-chip self-characterization employing accurate absolute phase measurement methods. A single receiver channel system capable of absolute phase measurement will certainly win over existing multi-channel setups, due to the reduced power consumption and circuitry real estate.

Section 3.2 describes a novel 1-channel VSA-based NVNA system that does not need a multi-harmonic generator. In place of the multi-harmonic generator used for absolute phase calibration, tones that cover the receiver band are generated in an AWG, and characterized in a nose-to-nose calibrated equivalent-time sampling oscilloscope. To mimic a synchronous multi-channel receiver, a 1-channel VSA sequentially samples each of the DUT signals. To

maintain timing relationship between the sequentially sampled signals, the 2-tone excitation waveforms are designed to be on a DFT grid that is commensurate with the VSA sampler time base. The proposed 1-channel receiver implementation can be easily adapted to hardware that are more readily available to the practicing engineer, and has potential for future self-tuning and self-healing RF front-end architectures that requires an accurate and low overhead on-chip self-characterization method.

Section 3.3 describes a novel 4-port VNA-based NVNA system that does not need a multi-harmonic generator. In place of the multi-harmonic generator used to maintain phase reference across a power sweep, a signal generator phase-locked to the multi-tone excitation is used, and can now tune to tone-spacing on the order of 1 Hz. The power sweep data is acquired by the high dynamic range super-heterodyne mixer-based VNA, and corrected in absolute phase by characterizing the DUT at a chosen power level using a low dynamic range equivalent sampling oscilloscope based NVNA. The narrowest tone-spacing usable is now determined by the memory depth of the oscilloscope. Using each instrument to its advantage, narrow tone-spacing, wide bandwidth, and high dynamic range is achieved.

2.3 Review Nonlinear RF Behavioral Modeling

Nonlinear models can be divided into two major groups: physical models and empirical models. A physical model has the constituent elements represented at a lower abstraction layer that is deemed the best implementable physical description, and has the rules governing the interaction between elements based on principles that are deemed more fundamental. Empirical models are anything other than the physical models. At each abstraction layer in the communication system hierarchy described in Section 1.1.1, both types of model are useful.

For transistor devices, the physical model consists of device layouts and material parameters for use in semiconductor physics based simulation, which offers the best implementable physical description. In the circuit implementation level of the communication system hierarchy, the physical device model is not suitable for circuit simulation due to the long simulation time

required. More suitable for circuit simulation are two levels of empirical models which can be extracted from the physical model's simulation or device measurement results. The equivalent circuit model of the device offers good predictability from a low dimension parameter set, but is not considered the most physical description possible. The behavioral black-box model of the device is based on observations that are valid for general active devices, but has lower predictability.

For a nonlinear circuit, a circuit-level description using equivalent circuit models becomes the best implementable physical description, since simulation based on device physics is too computationally intensive to be practical. In the link level of the communication system hierarchy, the circuit-level description is not suitable for link-level system simulation due to the long simulation time required. In this case, empirical models include behavioral black-box and grey-box models which can be extracted from circuit simulation or measurement results. Both being reduced-order system-level models, they enable rapid computation of the system behavior with digitally-modulated communication signals. The available approaches for reduced-order modeling of nonlinear circuits are reviewed in [40], [41], [42], [43].

Section 2.3.1 describes behavioral black-box approaches that have been used with success in modeling nonlinear RF circuits. Black-box models capture the behavior of a network at external ports but have little relationship to the underlying physics, which makes them suitable for intellectual property (IP) encapsulation.

Section 2.3.2 describes the grey-box approach which can provide similar, but usually much more precise RF system simulation than can be obtained using black-box models. The grey-box models, so called because the internal structure is based on physical insight, are often based on drastically simplified equivalent circuit-level description to capture the dominant contribution to the nonlinear behavior from a comparably low parameter set dimension. In fitting the terminal behavior, structural information is obtained as a byproduct of the model extraction process, and is the emphasis in system identification [32].

There are other important attributes to consider when choosing a modeling solution for a

particular application. One attribute is the model representation's compatibility with system-level simulators which use complex envelop signal representation, and circuit simulators which represent signals in real time. System-level simulation typically models the inband distortion behavior, namely the response generated within several modulation bandwidth of RF carriers, using complex lowpass equivalent models. Circuit simulation can predict broadband distortion behavior such as harmonics and IM products far away from the RF excitation, and mismatch effects in multi-port devices.

The ability to model a bandwidth-dependent memory effect is also of concern for applications modeling the IM products within several modulation bandwidths of the RF carriers. Linear memory effects refer to the amplitude and phase distortion that originate from the linear matching or filtering networks around RF carrier frequencies in typical bandpass communication systems. Nonlinear memory effects are anything other than linear memory. Electrical origins of nonlinear memory are distortion contributions containing nonlinear mixing products that have been shaped by frequency-dependent linear networks at frequencies other than the RF carriers. A prominent form of nonlinear memory is the long-term memory effect [44], which is manifested as IM responses that are dependent on the slowly-varying envelope of the RF carrier, and with time constants much longer than would be expected from the reactive elements in the RF path. The effect is attributed to RF signals being down converted to baseband through nonlinear electrical or thermal processes, and baseband impedances or device thermal characteristics affecting RF operation.

2.3.1 Black-Box Behavioral Models

Several black-box modeling approaches have been developed for both system-level and circuit-level simulation. One approach applies approximation to general nonlinear differential operators, and has formal mathematical support. The analytical form leads directly to a model topology with canonical realization suitable for signal processing structures. Model parameters are determined from fitting training data using linear least square or nonlinear optimization.

A related approach is the state space-based method typically applied in modeling RF transistors. The terminal voltages and currents and their time derivatives are states of the nonlinear differential operator representing the transistor device. The most viable approach adopted by the industry is based on the describing function.

Approximate Nonlinear Operator: Polynomial, Neural Network, Volterra Series

Under conditions of causality, stability, continuity and fading memory [45], a general nonlinear differential operator can be approximated by recursive and nonrecursive forms of multidimensional polynomial filter and artificial neural network (ANN), yielding multidimensional polynomial and ANN solutions. For nonrecursive forms, different function expansions such as Taylor series and orthogonal polynomials can also be used to approximate the multidimensional nonlinear function in the differential operator, yielding Volterra and Wiener series solutions to the operator.

The universal approximation property of these solutions is used to replicate arbitrary nonlinear behavior, with ANN being a global approximator due to its use of the strongly nonlinear sigmoidal function which produces bounded output in extrapolation, while all others are local approximators which may become unstable in extrapolation.

For computer implementation, the discrete-time domain formulation is most suitable. The nonrecursive polynomial and Volterra filters contain large numbers of model parameters, but the dependence is linear. They can be extracted using linear identification procedures such as linear least squares [46]. The recursive polynomial form can model the same nonlinear behavior with fewer parameters, but is nonlinear in the parameters, which is similar to an ANN-based model and requires nonlinear optimization procedures for parameter extraction.

In circuit simulation applications, these formulations can be generalized to multi-port devices to approximate each port wave. The so called Volterra input-output map (VIOMAP) [47] is a broadband 2-port model. Later work used general polynomials and ANNs [48], [49] to overcome the mild nonlinearity limitations. In [49] the number of embedded input variables which

determines the system dimension was found using nonlinear time series analysis techniques [50]. The training used excitation consisting of a 2-tone signal driving the input and another tone at the output.

In system simulation applications modeling inband distortion behavior, the complex lowpass equivalent form of the above universal approximation solutions are used. In [51] a hybrid time-and-frequency-domain Volterra formulation was developed. The kernel was extracted with random phase multi-tone excitation. In [52] a pruning approach eliminates the insignificant parameters in the lowpass discrete-time domain Volterra formulation, to produce a nearly diagonal Volterra series with minimal cross products between samples of different delay. The model parameters were extracted from time-domain envelope measurement with a modulated signal. A 7th order model was demonstrated and can model up to the 1 dB compression point.

To reduce the large number of parameters, the Volterra formulation can be simplified as a filter-memoryless-nonlinearity (Wiener), or as a memoryless-nonlinearity-filter (Hammerstein) model [53]. In the latter, there are no cross products between samples of different delay which is the same as with a diagonal Volterra series model. These were the earliest generation of models replicating the inband linear memory behavior in system simulation. They were extracted using swept 1-tone measurements, and can only model linear memory [9] [10] [11] [2].

To model nonlinear memory in system simulation, parallel branches of complex lowpass Wiener or Hammerstein models have been found to be a satisfactory solution. In [54] parallel branches of the Wiener model were used to model long-term memory in power amplifier circuits. The parallel branches were estimated one branch at a time, first using a correlation technique to estimate the linear block [55], then iteratively estimating the parameters of the nonlinear block. Since no physical correspondence can be made between the branches and origin of memory in circuit topology, these models are treated as black-box models. In [56] [57] [58] a solution based on a modified low pass equivalent Volterra formulation derived from nonlinear integral model, results in a parallel Hammerstein structure. In all cases 2-tone frequency-domain measurements were used to extract the model, with good fit up to 1 dB compression.

In another approach to modeling nonlinear memory in system simulation [59], the solution to the nonlinear differential operator is hypothesized to be dependent on hidden controlling variables in addition to the instantaneous envelop amplitude. Such hidden variables arise from physical processes such as self-heating or varying power supply. The solution is then linearized around a large-signal operating point chosen to be that modeled by the memoryless power series, with the deviation being the filtered excitation envelope. This filter function is extracted using an excitation with a step function envelope.

State Space

In modeling transistor devices to be used in circuit simulation, both RF measurements and DC I-V curves need to be predicted. The terminal voltages and currents and their time derivatives are states of the nonlinear differential operator representing the transistor device. Distinct from the previous approach where the nonlinear function of the differential operator is approximated using basis functions such as polynomials, here the nonlinear function is reconstructed from measurement data [60]. For the measurement data used in model extraction, it is important that the experiment be designed to excite all states of the system.

In [61] a transistor model based on the 1st order truncation of the modified Volterra series is implemented as a parallel connection of frequency-weighted static nonlinearities. The model is extracted using 2-tone frequency-domain measurements and can model up to 4 dB compression. The work of [49] can also be viewed as a state space method, where the number of embedded input variables which determines the system dimension is found using nonlinear time series analysis techniques [50].

Describing Function

The describing function approach is suitable for modeling a hard nonlinearity [62]. In [63] the X-parameters are described as the mathematically founded extension of the linear S-parameter matrix to the nonlinear case. They rely on the observation that in most RF circuits the super-

position principle is still valid at each large-signal dynamic (time-varying) quiescent operating point. The elements of the jacobian calculated from NVNA measurements at each operating point, are termed X-parameters and are used in harmonic balance circuit simulation. In this case no model parameters are extracted. The measurement and simulation technique is the complete modeling solution.

In [64] this principle is used in modeling PAs, with the operating point established by a high power 1-tone excitation to approximate the RF carrier. The X-parameters are determined experimentally using the response to a small-signal 1-tone perturbation injected into either the input or output, and at an offset frequency from the large 1-tone. As the model was conceived to describe the PA behavior around each harmonic of the large-signal 1-tone pumping signal, the authors called it the polyharmonic distortion model. Later in [65] a model extension based on the hidden variable method in [59] was added to model nonlinear memory.

Other novel 2-port circuit-level behavioral models have been developed based on the describing function idea. In [66] a model consisting of sets of 2-port y-parameter matrices was used in circuit design optimization. The measured IM2 and IM3 components as a function of the ports' envelope and 2nd harmonic impedance for a 2-tone input, are captured as lumped nonlinear sources in a general 2-port. These 2-port models were used in place of the active device in a 2-stage small signal amplifier circuit. Simulation and optimization was carried out in Matlab to find the optimal inter-stage matching network.

2.3.2 Grey-Box Behavioral Models

Grey-box model structures of unknown RF architecture are developed based on physical insight. Since there are a finite number of RF front-end architectures, a suite of grey-box models can be developed to capture the fundamental linear and nonlinear interactions of most systems. The model structures are often based on drastically simplified equivalent circuit-level description of the RF front-end.

Grey-box structures can be in an equivalent circuit form suitable for circuit simulation, or

a complex lowpass form which is suitable for system simulation, much like the case for the Volterra series based model. For parameter estimation, the equivalent circuit must be analyzed using general purpose circuit simulation to relate the model parameters to RF measurements. Typically frequency-domain Volterra circuit analysis is used to yield the envelope response expression at each IM frequency of interest for multi-tone excitation.

The Volterra NLTF can also be derived and simplified to yield a complex lowpass equivalent model which is usually applied to model the inband behavior for a modulated signal input. The simplification step lumps the instantaneous model parameters of the equivalent circuit into complex ones for the lowpass model. For parameter estimation from measurements, the lowpass model is used to calculate the time-domain envelope response.

The model identification problem is then one of extracting a large number of parameters, while possibly considering several candidate grey-box models in parallel. Historically, the parameter identification problem is studied on a case-by-case basis.

System Simulation Models

As was previously discussed in Section 2.3.1, the Wiener, Hammerstein and Wiener–Hammerstein structures were used in the earliest generation of system-level models [9] [10] [11] [2]. For parameter estimation from 1-tone swept frequency and power measurements, the frequency-domain form of the complex lowpass models were used. Here the instantaneous model parameters of the real equivalent circuit are related to the complex parameters of the lowpass model by a multinomial scaling coefficient. So if the RF front-end is a single-stage low- to medium-frequency microwave amplifier without long-term memory, with frequency shaping input and output networks such as matching networks or filters, and with frequency-independent nonlinear active device in the middle, then the three-box Weiner-Hammerstein model is usually an adequate model.

The complex lowpass parallel Wiener model, described in Section 2.3.1, was used to model the long-term memory effect in system simulation [54], but the structure does not correspond

to the physical origin of the nonlinear memory. It is a black-box model that has the particular approximation property needed to model the memory effect.

There is also a hybrid method combining prior physical knowledge in grey-box models and parameter estimations methods used in fitting Volterra series based models. By using physical knowledge the number of model parameters is reduced. In [67] [68] this approach was applied to model nonlinear memory, using a Wiener-Hammerstein model with linear feedback which has been found to model the physical mechanism responsible for the electronic contribution to the long-term memory effect from bias circuit impedance. In [67] the envelope response expressions at IM frequencies of interest are derived from Volterra circuit analysis, then simplified by lumping the instantaneous model parameters into complex ones. The feedback linear block is estimated in the frequency domain using the first adjacent channel IM response from 2-tone measurements. The complex nonlinear parameters are then estimated from the IM response of a chosen order. The model identified showed good prediction of the response to digitally-modulated communication signals, though fitting results to 2-tone measurements were not presented. In [68] the discrete-time Volterra series solution was first derived. However the nonlinear dependence on the instantaneous model parameters in the kernel expression would require nonlinear optimization for parameter estimation. To use linear parameter estimation techniques as in [52], coefficients of multi-nomials, which are products of input samples with different delay, are lumped into complex parameters. Then the Volterra series is linear as in the complex model parameters, and can be identified from time-domain envelope measurements with a modulated signal. In both works, the linear input and output networks were assumed to be frequency-independent.

In [69] the complex lowpass multi-slice model was used to model the long-term memory effect in system simulation. One slice is an odd-order complex power series which models the memoryless nonlinear effect. The second slice models the physical mechanism of the memory effect by multiplying the output of an even-order complex power series with the input. The parameters of each slice were extracted from 2-tone frequency-domain measurements. The

inband IM3 response was modeled into heavy compression, but only at one tone spacing.

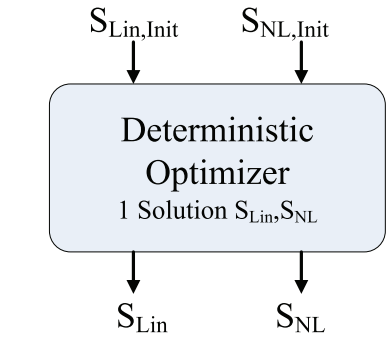
Circuit Simulation Models

The Wiener, Hammerstein, Wiener-Hammerstein, and Hammerstein-Wiener models are the most heavily studied RF nonlinear grey-box models for circuit simulation applications [55], [70], [71], [72], [73], [74], [75]. Numerous parameter identification methods for these models have been developed using various test excitations including 1-tone, multi-tone and noise signals with parameter fitting performed in the time or frequency domains. In all cases the model extraction procedures were specific to the model being developed and rely on deterministic optimization methods.

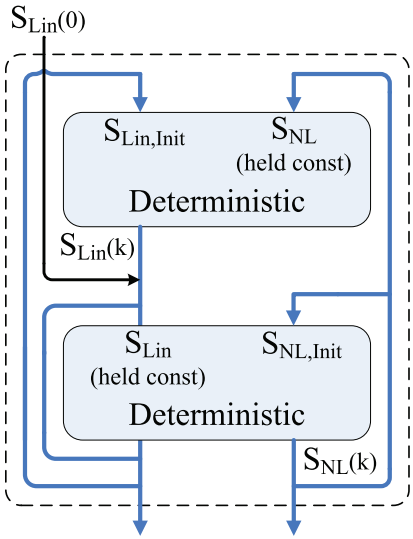
A previous limited stochastic method used correlation techniques but required noise test input signals only [55]. To a limited extent it is possible to use a non-iterative method known as over-parametrization [70] but this is effectively limited to low-dimensional systems. The most commonly used parameter estimation procedures rely on 'a human-in-the-loop' to guide the process of error minimization between the modeled and measured responses, and nearly always used nonlinear gradient-based deterministic optimization methods for which, given the same starting point, the optimizer always yields the same solution.

An error function is minimized either with respect to all linear and nonlinear model parameters simultaneously [71], [72] as in Fig. 2.9(a), or with respect to subsets of the parameters in an iterative loop [73], [74], [75] as in Fig. 2.9(b). The first, simultaneous, parameter extraction approach is illustrated in Fig. 2.9(a). In the second, partitioned approach, the procedure begins with an initial solution for the linear model parameters estimated in what is often an ad-hoc fashion. Following this the nonlinear parameters are estimated, and then the linear parameters are updated. In Fig. 2.9(b), '$\mathbf{S}$' represents the model parameter solution, with the subscript 'Lin' indicating the linear part and 'NL' indicating the nonlinear part. The subscript 'Init' identifies the initial guess and k indicates the kth iterative solution.

The form of the model can be exploited in coming up with an initial solution for a deter-



(a)



(b)

Figure 2.9: High level view of the identification algorithm using a deterministic optimizer for the linear and nonlinear model parameters: (a) brute force approach estimating all parameters simultaneously; and (b) partitioned parameter approach alternately optimizing linear and nonlinear parameters in separate iteration steps.

ministic optimizer. In [74] the parameters for the two linear blocks in the Wiener-Hammerstein model are initialized using the square root of the linear frequency response, and iteratively estimated using the partitioned approach. In [75] the Wiener-Hammerstein model with linear feedback is also identified using the partitioned approach. The parameters of the two linear blocks in the forward path can be estimated from the measured second-order NLTF,

$$\mathrm{G}_{[2]}(\omega_1, \omega_2) = b_2 \mathrm{G}_{12}(\omega_1 + \omega_2)\mathrm{G}_{21}(\omega_1)\mathrm{G}_{21}(\omega_2). \tag{2.24}$$

The model is simple enough that the expression is a product of the linear transfer functions scaled by one nonlinear parameter. Due to the parameter uniqueness modulo scaling which is described in detail in Section 4.1.3, the nonlinear parameter is ignored during the estimation step in Fig. 2.9(b) that estimates the linear transfer functions. This is because an arbitrarily scaled linear transfer function will result in arbitrarily scaled nonlinear parameters which still satisfy what can be uniquely identified. Thus the identification algorithm is effectively terminated after one loop, since the results of the nonlinear parameter iteration step will not affect the linear parameter iteration step.

2.3.3 Current Research

An important question raised by the preceding discussion is how other useful models can be discovered and identified, especially for multi-stage nonlinear circuits? It is essential that parameter extraction for these more sophisticated models be developed without a large amount of engineering effort expended in customizing parameter extraction as has been done for the Wiener-Hammerstein family of models.

Current research explores general identification methodology for more complex nonlinear models than have been used previously. The type of problem for which this work is targeted is extracting the grey-box models of the RF front-end of undocumented devices for co-site interference simulation, for example, where two or more radios are operated in close proximity

and performance is compromised by broadband noise and the unintended active scattering of intermodulation (IM) components, both effects interfering with radio reception.

The model identification procedures developed in past works are not suitable due to the ad-hoc nature of the identification algorithm and the use of deterministic optimization. Deterministic local search methods based on gradient or Nelder Mead simplex depends on the starting point of the search, which may suffice for simple models where initial solution close to the global optimum can be found. For more complex models a good initial solution is much harder to find. In addition, deterministic optimizers are prone to getting trapped in local optimum solutions due to their dependence on the starting point. That is, modifying the initial parameter estimates can often result in a different, and possibly preferable, solution being found. As a result the modeling exercise is not straight-forward and all too often must proceed either by choosing alternative starting points for the model fitting algorithm, or by modifying the error cost function.

Any new model identification procedure developed must have the ability to explore the search space. Stochastic optimizers have additional randomization mechanisms built in at each optimization iteration, so as to avoid local optimums. So unless the same random seed number is used, a stochastic optimizer will always yield a different solution each time it is run.

With a search space exploration ability, the global solution having the minimum error between the modeled and measured responses can be found. However the global solution may not be a physically or intuitively satisfying solution to a skilled and knowledgeable engineer with experience acquired with the type of system being modeled. One reason is that in most RF systems, the frequency shaping by bandpass filters and matching networks prevent direct observation of nonlinear effects. Also there are rarely enough external measurements available to unambiguously identify all of the model parameters. As a result of noise in the measured response, indirect observables, and model structure inadequacies, multiple optimum solutions with similar error metrics should be tracked for selection by the user.

Section 4.2 generalizes the identification methodology to more complex nonlinear models

than have been used previously. The ad-hoc higher-level identification algorithms presented in previous studies are generalized and adapted to stochastic optimization methods. A modified genetic algorithm (GA) procedure is developed to track sets (or families) of possible solutions. For each family, a derivative-free gradient descent algorithm, the Nelder-Mead simplex algorithm, is used to present a skilled operator with a number of distinct modeling solutions with comparable error metrics. The human in-the-loop is removed and user input is directed first at establishing several candidate grey-box models, establishing the measurement protocol, and then choosing the best solution at the end of the procedure.

Section 5.2 presents the direct calculation method of Volterra circuit analysis in an algorithmic form to handle an arbitrarily high degree of device nonlinearity. The expression for nonlinear current sources is determined by recursively finding the lower-order IM response due to the controlling variables of nonlinearities. For the same response frequency, contributions of different order are differentiated, which enables compression handling which is lacking in the original work [6]. A symbolic Volterra circuit simulator implementing this method in Matlab is used to automate the model analysis and response calculation. This can be compared to the typically hand-crafted Volterra NLTF analysis used in past work which are particular to model being developed. The new method enables more generalized model fitting and eliminates user tasks that can be automated.

2.4 Summary

In this chapter prior research on nonlinear RF circuit characterization and behavioral modeling was summarized in order to place current research in its proper historical context.

Current state of the art NVNA setups for nonlinear characterization rely on the multi-harmonic generator and require synchronous multi-channel receivers for implementation; Chapter 3 presents two novel NVNA setups that do not require a multi-harmonic generator and can be implemented using common benchtop instruments, using a 1-channel VSA receiver in one implementation and a 4-port VNA in another.

Prior work on RF model identification for circuit simulation applications have focused on the Wiener and Hammerstein family of models using ad-hoc higher-level identification algorithm and relying on deterministic optimization methods; Chapter 4 presents a methodology that reorganizes the way more sophisticated models are extracted for RF devices, using a symbolic circuit simulator compatible with parameter extraction procedure to automate model analysis and response calculation, and a generalized higher-level identification algorithm adapted for stochastic optimization methods to find multiple optimum solutions with comparable error metrics.

The original work on the direct calculation method of Volterra circuit analysis was not presented in an algorithmic form suitable for implementation in a circuit simulator and did not accounted for compression effects; Chapter 5 presents the direct calculation method in an algorithmic form to handle arbitrarily high degree of device nonlinearity and compression effects. A symbolic Volterra circuit simulator implementing this method in Matlab is used to automated the model analysis and response calculation.

Having reviewed a number of time-frequency concepts and past research, the remainder of this dissertation is devoted to original work contributed by the author.

Chapter 3

Nonlinear Metrology

3.1 Introduction

One issue with existing absolute phase measurement setups are their reliance on the multi-harmonic generator. Such a phase reference generator may not be readily available, or more importantly, impose tone-spacing limitations in multi-tone testing. Small tone spacing is required when characterizing memory behavior in nonlinear microwave circuits.

Another issue that hinders the wide adoption of the absolute phase concept in everyday design practice is that a commercially available NVNA constitutes a major capital investment.

This chapter presents two novel NVNA setups that do not need a multi-harmonic generator, and can be implemented using common bench top instruments.

3.1.1 Dependence on Multi-Harmonic Generator

Typically a multi-harmonic generator is driven at a low frequency fundamental, the grid frequency, f_{grid}, to produce harmonic tones on a grid, i.e nf_{grid}, reducing in amplitude as n increases. This grid defines the frequencies of the two-tone signal with the minimum tone-spacing being f_{grid}. The need for a harmonic reference tone to have an acceptable signal-to-noise ratio (SNR) at the two-tone signal frequencies establishes the minimum f_{grid}, i.e. the minimum tone

spacing.

The power of the fundamental driving signal cannot be simply increased, since these lower harmonic tones will saturate the receiver front-end. A microstrip filter can be designed to shape the generator output power and phase spectrum to lower its peak to average ratio (PAR). Still it is difficult to meet a typical receiver bandwidth of 20 GHz without placing a lower limit on the fundamental frequency. For example, in [5] the minimum f_{grid} is 1.242 MHz for a two-tone test signal centered at 3.01185 GHz; and a commercially available NVNA of this type has a minimum tone-spacing of 10 MHz.

The previous 4-port VNA-based NVNA [5] has a dynamic range around 85 dB, but is limited in the narrowest tone-spacing which is often required for characterizing the long-term memory behavior in power amplifiers. The previous sub-sampling mixer-based NVNA [17] is not limited in the narrowest tone-spacing, but has a lower dynamic range around 60 dB.

3.1.2 Implementation Using Common Benchtop Hardware

The more readily available bench top instruments are the AWGs, spectrum analyzers, oscilloscopes, downconverters, and analog to digital converter (ADC) evaluations boards. It may be a lower cost and more practical solution to build an absolute phase measurement system using these hardware.

Typically a single receiver channel can be built rather easily, compared to the four or five synchronous channels required in existing NVNAs. In the foreseeable future, self-tuning and self-healing concepts will likely be applied to RF front-end architectures. This will require on-chip self-characterization employing accurate absolute phase measurement methods. A single receiver channel system capable of absolute phase measurement will certainly win over existing multi-channel setups, due to the reduced power consumption and circuitry real estate.

3.1.3 Proposed NVNA Setups

Section 3.2 describes a 1-channel VSA-based NVNA used together with an AWG and an equivalent-time sampling oscilloscope. The proposed 1-channel receiver implementation can be easily adapted to other hardware that are more readily available to the practicing engineer, and has potential for future self-tuning and self-healing RF front-end architectures that require an accurate and low overhead on-chip self-characterization method.

Section 3.3 describes a 4-port VNA-based NVNA used together with common benchtop instruments; phase-locked signal generators and an equivalent-time sampling oscilloscope. Using each instrument to its advantage, narrow tone-spacing, wide bandwidth, and high dynamic range is achieved. The narrowest tone-spacing usable is now determined by the memory depth of the oscilloscope. Section 3.4 summarizes this chapter's results.

3.2 Novel 1-Channel VSA Based NVNA

The VSA is used in link-level performance testing using modulated signals, and is more widely adopted than the NVNA. Fig. 3.1 shows a block diagram of the VSA internals. Typically a heterodyne receiver-based spectrum analyzer downconverts the RF to 70 MHz IF, and a 14 bit 100 Msa/S ADC unit samples the IF and store the time-domain samples in memory, much like a bandpass real time oscilloscope. Thus the VSA can achieve 75 dB SFDR, greater than the 60 dB SFDR possible with the sub-sampling mixer-based receiver or the sampling oscilloscope. The real time bandwidth is limited to around 40 MHz, accounting for addition margin beyond that required for Nyquist sampling.

Like the sampling oscilloscope and the sub-sampling mixer used in previous NVNAs, the VSA can simultaneously capture all spectral components within its bandwidth. However to use it in an NVNA setup, several issues need to be addressed.

First a 1-channel receiver cannot synchronously sample the input excitation and the DUT signals as in a synchronous multi-channel receiver. Section 3.2.1 presents a solution to mimics

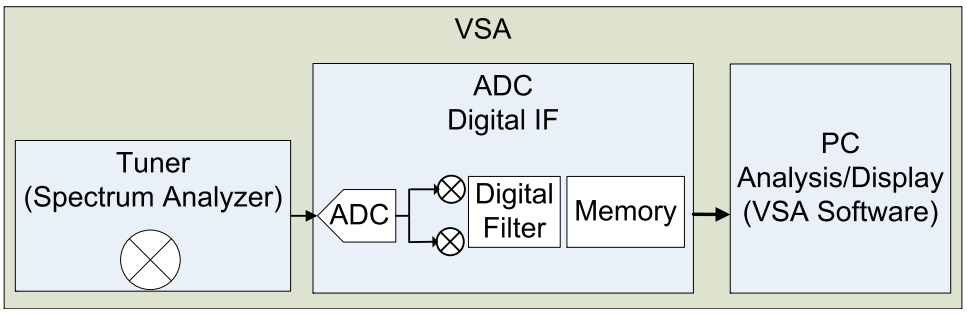


Figure 3.1: Block diagram of a 1-channel VSA.

synchronous multiple channels by sequentially sampling each port signal. To maintain timing relationship between the sequentially sampled signals, the excitation waveforms are designed to be on a DFT grid that is commensurate with the VSA sampler time base. This solution is experimentally validated.

Second the multi-harmonic generator is not used for absolute phase calibration. Instead an AWG can generate repeatable multi-tone signal which is characterized in a nose-to-nose calibrated equivalent-time sampling oscilloscope. Section 3.2.2 presents the NVNA setup, and compares it to previous broadband receiver based NVNAs [17]. Section 3.2.3 presents the theory of the absolute phase calibration for the proposed NVNA. Section 3.2.4 presents the measurement results.

The proposed NVNA can be viewed as a narrowband version of the previous broadband receiver based NVNAs [17], and suitable for characterizing DUT's in-band behavior for a narrowband multi-tone excitation. However the dynamic range is improved, due to the use of a narrowband tuned receiver instead of a sub-sampling mixer-based front-end where noise at all sidebands of the sampling frequency harmonics are downconverted to the baseband.

3.2.1 Mimic Synchronous Multi-Channel Receiver

To mimic a synchronous multi-channel receiver, the DUT signals are sequentially switched to the input of a 1-channel receiver for sampling. The incident excitation is first sampled. Then during the sampling of each DUT signal, the excitation waveform is repeated. This setup is

shown in Fig. 3.2, and the excitation sequence is shown in Fig. 3.3. Note that for active devices with gain, the DUT's output must be attenuated to maintain the same VSA setting, and thus the same dynamic range for both the input and output signals. Several issues must be addressed for this approach to work.

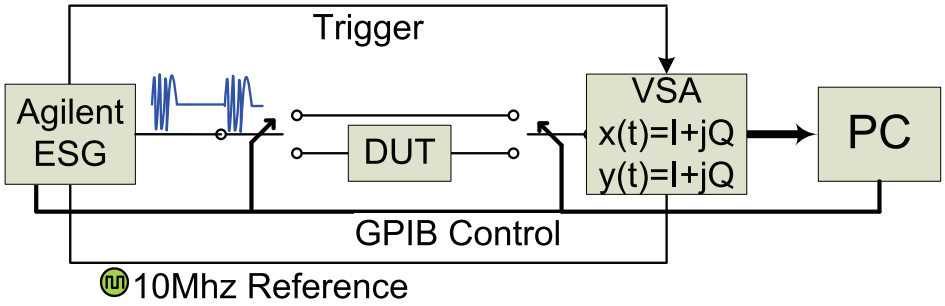


Figure 3.2: Sequentially sample DUT signals using VSA, to mimic a synchronous multi-channel receiver.

First the excitation waveform must be repeatable. The AWG can store in memory multiple waveform segments arranged in a sequence, and play back the same sequence multiple times. Multi-tone signals created in this way have been shown to be repeatable in [76], where the variance of the relative phase between the tones are less than 0.5° up to 2.5 MHz tone-spacing, and less than 1° up to 20 MHz tone-spacing over 100 measurements.

Second, the sequentially sampled waveforms must hold the same timing relation as if they were synchronously sampled. One possible solution is to align the DUT signal to the excitation as described in Section 2.2.2. However since the DUT introduces phase change and distortion, the assumption essential to alignment is violated, namely that signals are no longer time-delayed version of each other. Another possible solution is to trigger the VSA sampler each time the excitation waveform is repeated, using a marker signal from the AWG. However trigger signal is not reliable as a synchronization signal. As shown in the case of equivalent-time sampling

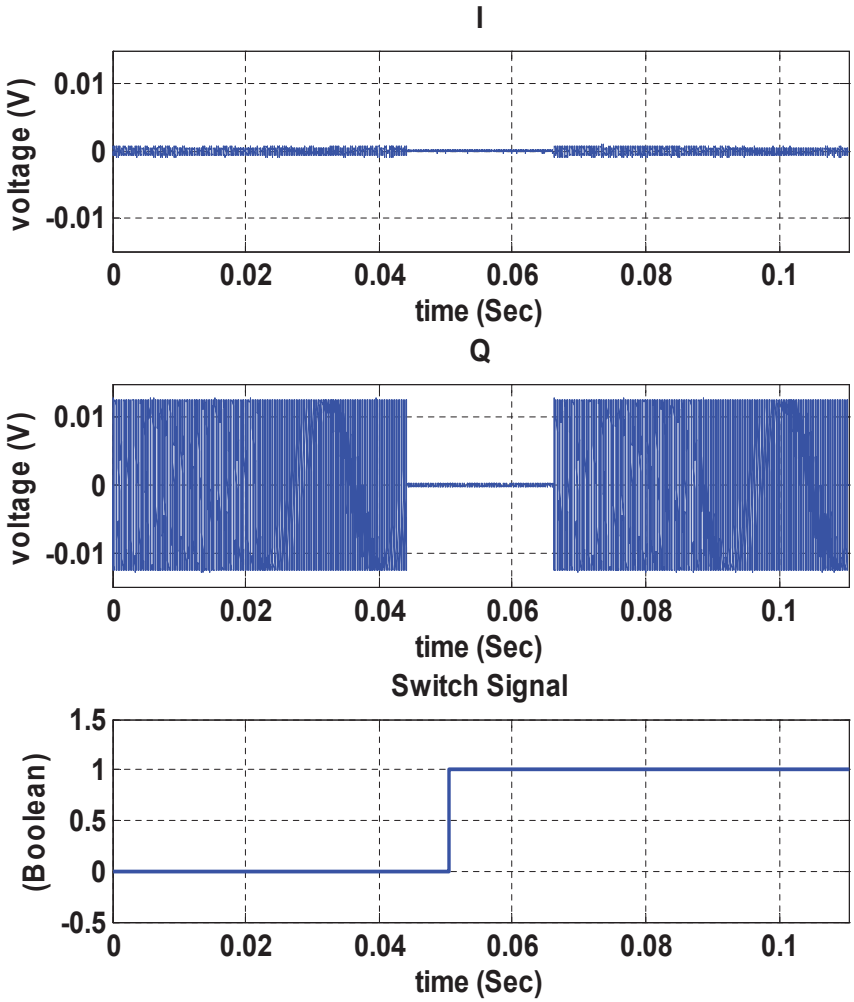


Figure 3.3: The excitation waveform sequence during sequential sampling.

oscilloscopes, the jitter created in the trigger circuitry is a dominant source of timebase error [23] [24]. In contrast, the jitter in the sampling clock is not a dominant source of timebase error.

Excitation Design to Align Signals Using VSA Timebase

The solution adopted in this work is to continuously sample the DUT signals as they are switched to the input of the VSA, then use the VSA sampler time base to maintain the timing relation between the excitation and the DUT signals.

To clearly explain this idea, the signal generation and sampling process must be considered in detail. To generate a periodic signal using an AWG without creating spurious response, the waveform segment must span an integer multiple of the signal period, with the first and last waveform points falling on a sampling point. If T_{Sig} and f_{Sig} are the signal period and frequency respectively, $T_{s,\text{AWG}}$ and $f_{s,\text{AWG}}$ are the sampling interval and frequency respectively, and n is an integer, then

$$T_{\text{Sig}} = n \times T_{s,\text{AWG}} \tag{3.1}$$

$$f_{\text{Sig}} = \frac{f_{s,\text{AWG}}}{n}. \tag{3.2}$$

Similarly, to sample a periodic signal without spectral leakage, the sampled waveform segment must span an integer multiple of the signal period, with the first and last waveform points falling on a sampling point. If $T_{s,\text{VSA}}$ and $f_{s,\text{VSA}}$ are VSA's sampling interval and frequency respectively, and m is an integer,, then

$$T_{\text{Sig}} = m \times T_{s,\text{VSA}} \tag{3.3}$$

$$f_{\text{Sig}} = \frac{f_{s,\text{VSA}}}{m}. \tag{3.4}$$

By searching through the available sampling frequencies of the AWG and the VSA, signal periods that satisfy the above requirements are found. For a 2-tone signal, the signal frequencies

found are the tone-spacings, which are also valid for a distorted 2-tone signal. The lowest tone-spacing achievable is limited by the memory depth of the AWG. For 8 Msa of memory and sampling rate of 1.25 Gsa/S, 1 kHz tone-spacing is possible.

Once the suitable signal period is selected, the switching interval is specified in terms of the number of signal periods. The excitation waveform segment and the switching interval waveform segment are stitched together in the AWG playback sequence, along with the appropriate number of repetitions for each segment. Now only one trigger signal plays back the entire sequence, during which each DUT signals is sequentially switched to the input of the VSA for sampling. Additional signal periods beyond the minimum required should be inserted in the playback sequence, to leave margin for the transient part of the DUT response.

Since the DUT is not turned off during the switching interval, this can be called 'hot-switching'. To prevent unpredictable DUT response during the switching interval, where two of the DUT ports are not terminated in a load, the switching interval waveform segment is nulled to all zeros.

In the post-processing step, the entire VSA sampled waveform is divided into segments corresponding to the sequentially sampled excitation and the DUT signals. The start and end of each segment can be determined, since the number of signal periods for the DUT signal duration and the switching interval duration is precisely known. The timing relation between the excitation and each DUT signal is maintained as if they were synchronously sampled.

Excitation Design to Reduce Measurement Time

Typical 2-tone testing over input power and tone-spacing, is conducted as a series of swept power measurements at a fixed tone-spacing. For the VSA, the sampling process has to be halted during the power sweep, as the attenuator in the spectrum analyzer has to be switched to maintain dynamic range.

It is more suitable if the testing is conducted as a series of swept tone-spacing measurements at a fixed power, which reduces the overall measurement time. There is no additional overhead,

since the AWG can easily concatenate waveform segments of different tone-spacings into the playback sequence. This dynamic frequency two-tone signal can be described mathematically as

$$x(t) \quad = \quad A\cos\left[2\pi(f_c - \frac{\delta(t)}{2})t\right] + A\cos\left[2\pi(f_c + \frac{\delta(t)}{2})t\right] \tag{3.5}$$

where the function $\delta(t)$ determines the tone-spacing over time. The discrete frequency separation over time is defined by

$$\delta(t) \quad = \quad \sum_{k=0}^{N} \rho_k \mathrm{rect}(t - t_k). \tag{3.6}$$

So for the duration of each t_k segment, the instantaneous spectrum can be considered a two-tone excitation. The baseband interpretation of this modulation is that the two single sideband tones are moving away from each other over time. The two dimensional time-frequency spectrogram calculated from the short time fourier transform is shown in Fig. 3.4.

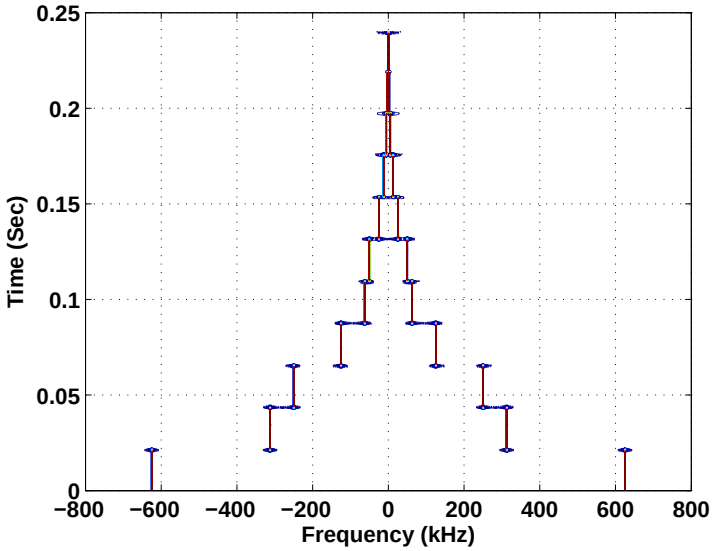


Figure 3.4: Spectrogram of a dynamic time-frequency two-tone signal to reduce measurement time, where the magnitude is plotted versus envelope frequency.

To select a set of tone-spacings from signal periods that satisfy (3.1) and (3.3), the corresponding set of signal periods selected must have its smallest $T_{s,\mathrm{AWG}}$ as a common divisor, and its smallest $T_{s,\mathrm{VSA}}$ as a common divisor. The switching interval can then be specified in terms of the number of the smallest signal period in the set.

In the post processing step, the entire VSA waveform is again divided into segments corresponding to each DUT signal and each tone-spacing. The start and end of each segment can be determined given the number of periods for each tone-spacing and the switching interval. Then the amplitude and phase of each IM component is extracted from the DFT result for each segment. Since each segment duration is an integer multiple of the 2-tone envelope period, there is no spectral leakage issue.

When determining the duration for each tone-spacing waveform, the signal to noise ratio (SNR) of the recorded signal needs to be taken into consideration. If the number of periods for each tone-spacing segment is equal, the time duration is smaller for larger tone-spacings. However bigger DFT frequency bins will collect more noise, and higher tone-spacing signals

will have degraded SNR. Thus uniform time duration is chosen for the tone-spacing segments, which shows similar SNR levels for all tone-spacings as shown in a composite envelope spectrum in Fig. 3.5(a). For comparison, the composite envelope spectrum using an equal number of periods for each tone-spacing segment is presented in Fig. 3.5(b) which shows SNR degradation. Both spectrum is based a dynamic 2-tone signal over a set of tone-spacings, and the waveform sequence lasts over one VSA record which is 102400 samples. Due to the long time-domain record captured by the VSA which keep the noise level low, and the spurious response of the VSA can be seen in the spectrum.

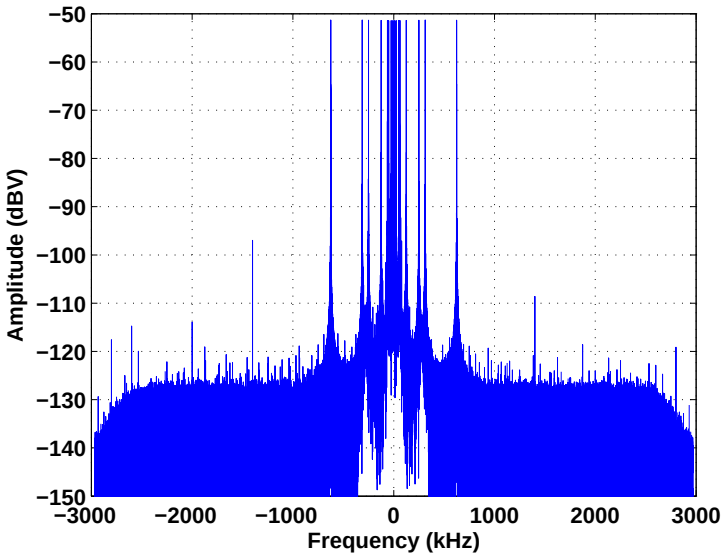
Additional signal periods beyond the minimum required should be inserted in the playback sequence for each tone-spacing, to leave margin for the transient part of the DUT response.

Experimental Validation: VNA

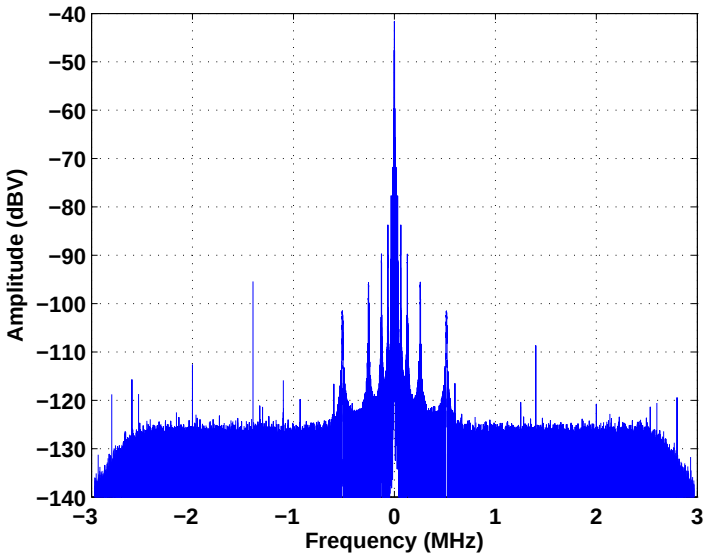
The 1-channel VSA method is compared with the 2-channel VNA using their measured linear operation region transmission response at low input power for a 2 GHz amplifier (Amplica SD513301). The VNA is calibrated using the full 2-port SOLT method, and the transmission response is the S_{21}.

The 1-channel VSA method does not incorporate any calibration at this point. Since the DUT is well matched, the only precaution taken during measurement is to ensure balance between the switched path to the signal generator and the switched path to the DUT output. Once the waveform segments for the excitation and the DUT output is separated from the VSA recording, DFT is performed to get the amplitude and phase. The ratio between the two signals at the excitation frequency is the linear transmission response.

The dynamic frequency 2-tone signal is used, with various tone spacings chosen to create the deviation from the 2 GHz center frequency. Fig. 3.6 compares the S_{21} measured using a VNA with the linear transmission response measured using the novel 1-channel VSA method, with one trace each for the upper and lower tone. The gain and phase are in good agreement. Note that the PA output is attenuated by an amount equaling its gain, in order to maintain



(a)



(b)

Figure 3.5: Composite envelope spectrum of a dynamic time-frequency two-tone signal using: (a) equal number of periods for each tone-spacing segment; and (b) uniform time duration for each tone-spacing segment. The magnitude is plotted versus envelope frequency.

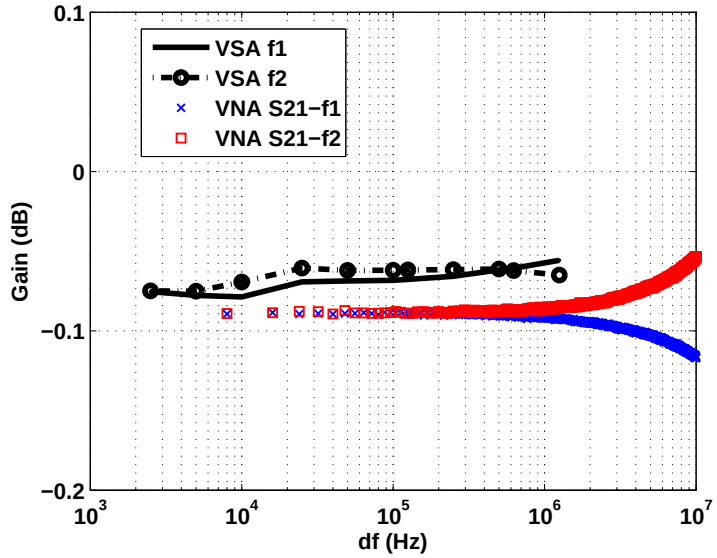
the same attenuator setting in the VSA.

Experimental Validation: Software Reference Nonlinearity

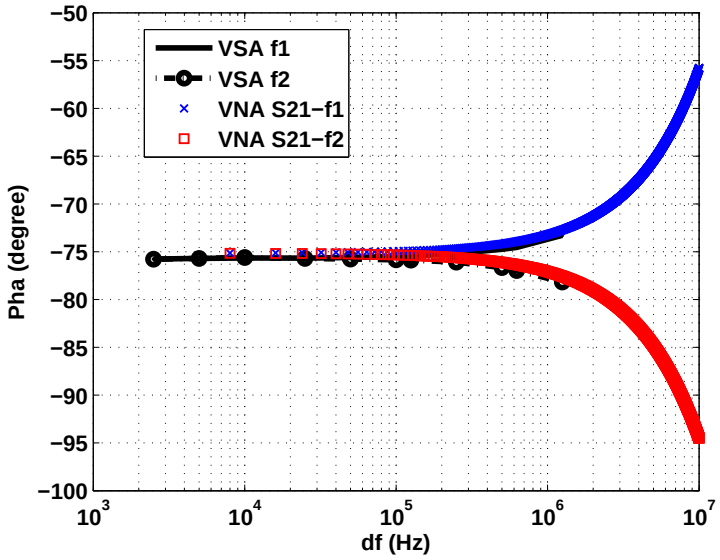
In this validation test, an artificial DUT is used. The DUT is synthesized using a known mathematical nonlinearity. First, the excitation $x(t)$ is played in the AWG sequence, followed by a switching interval. Then, the nonlinear response, which is the 3rd order adjacent channel response generated by the 3rd order polynomial nonlinearity $y(t) = a_1x(t) + a_3x(t)^3$, is played in the AWG sequence. The a_3 coefficient is set for certain IM3 ratio $\frac{3}{4}A^2\frac{a_3}{a_1}$ where A is the amplitude of the baseband excitation waveform. In effect, the DUT is replaced by a software nonlinearity, and the physical switching event is no longer required. Based on the measured excitation and nonlinear response, the a_3 coefficient is estimated. The amplitude and phase accuracy of the estimate will verify the proposed phase synchronization method in mimicking synchronous multi-channel receivers.

The dynamic frequency 2-tone signal is used with tone-spacing varied from 2.5 kHz to 1 MHz. For each tone-spacing, the waveform segment is played by AWG over one VSA record which is 102400 samples. The waveform sequence for the undistorted signal $x(t)$ and the distorted signal $y(t)$ is captured by the VSA. After post processing steps to align the time-domain samples for each tone-spacing, the spectrum for several tone-spacings is overlayed in Fig. 3.7(a) for $x(t)$ and Fig. 3.7(b) for $y(t)$. Due to the long time-domain record captured by the VSA which keep the noise level low, and the spurious response of the VSA can be seen in the spectrum. Note that there is almost no spectral leakage.

Figs. 3.8 and 3.9 compare the preset and the estimated value for the a_1 and a_3 coefficients. In each plot, separated traces are used for the upper and the lower tones. The NLTF which a variant of the time-invariant phase definition from Section 2.2.4, is used to estimate the values of a_1 and a_3. With 10 measurements averaged, the amplitude and phase error of the a_3 estimate

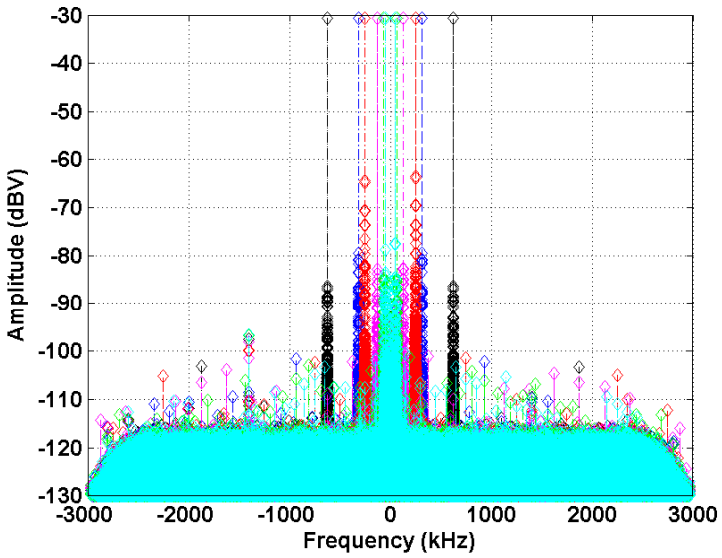


(a)

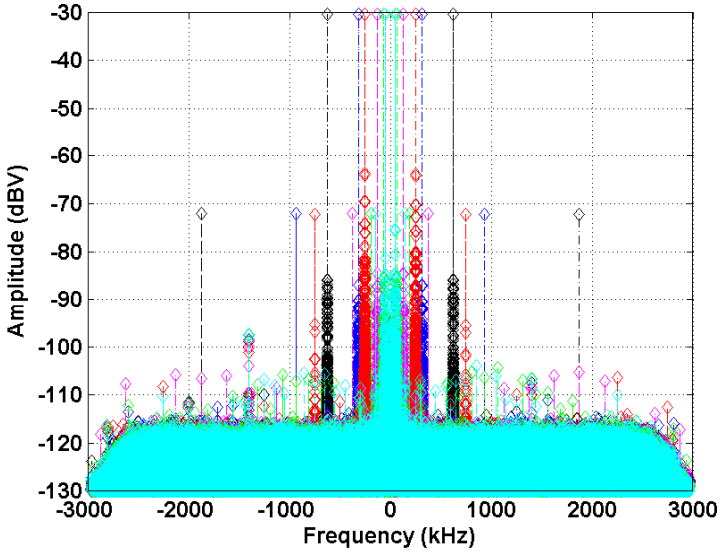


(b)

Figure 3.6: Compare S_{21} and linear transmission response measured using the novel 1-channel VSA method: (a) magnitude; and (b) phase. Both are plotted versus frequency deviation from the center frequency of the 2-tone.

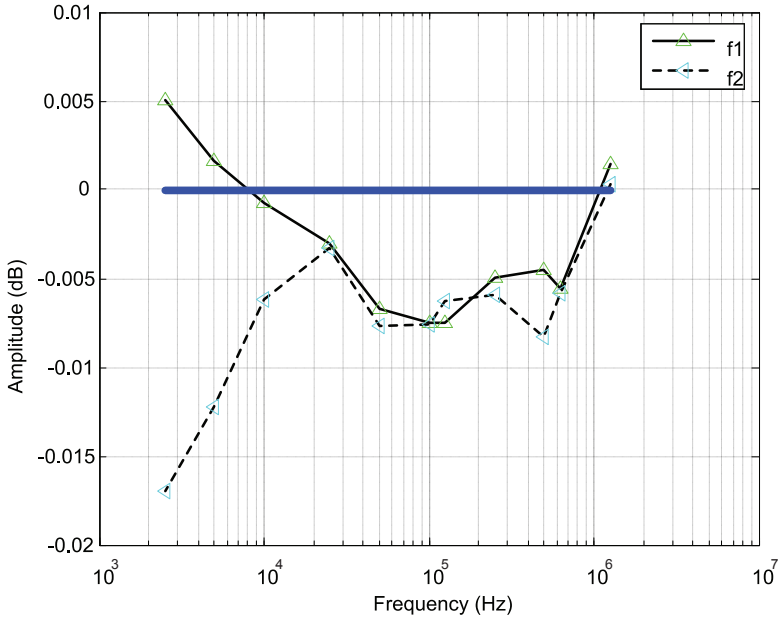


(a)

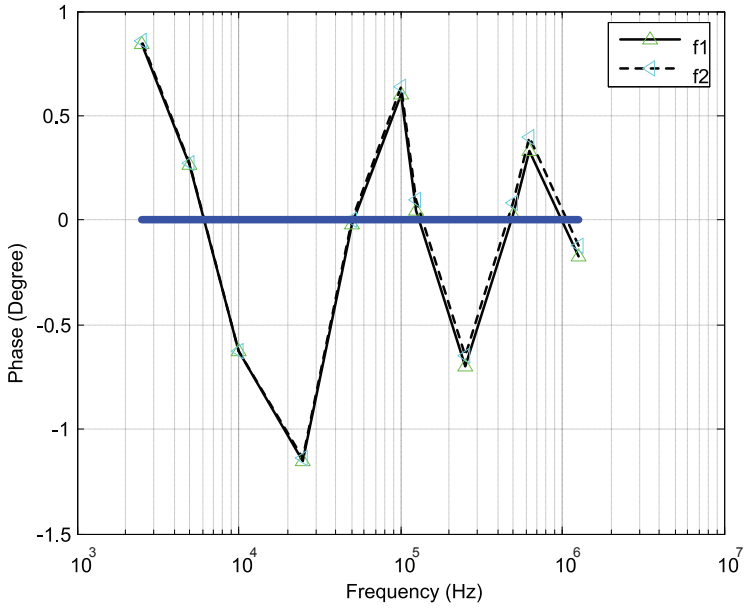


(b)

Figure 3.7: Envelope spectrum of time-domain record captured by the VSA at several tone-spacings for: (a) the undistorted signal $x(t)$; and (b) the distorted signal $y(t)$. The magnitude is plotted versus envelope frequency.

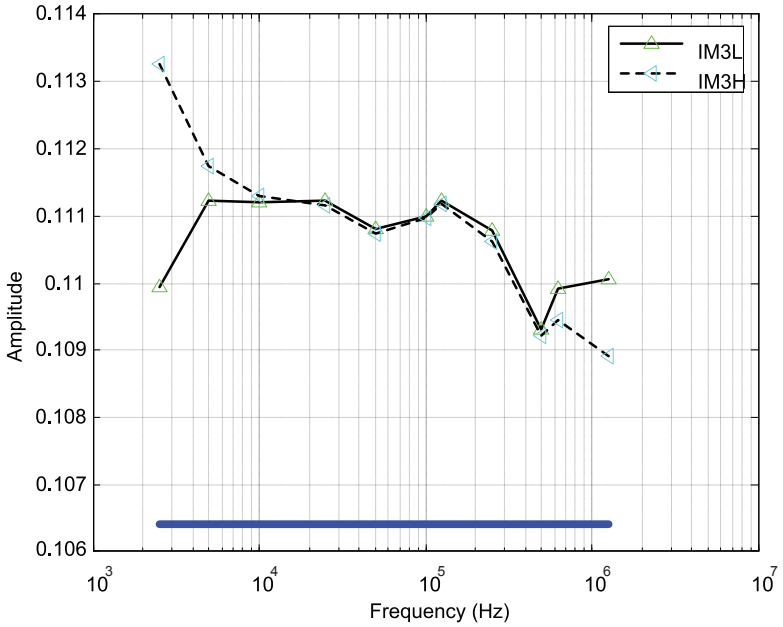


(a)

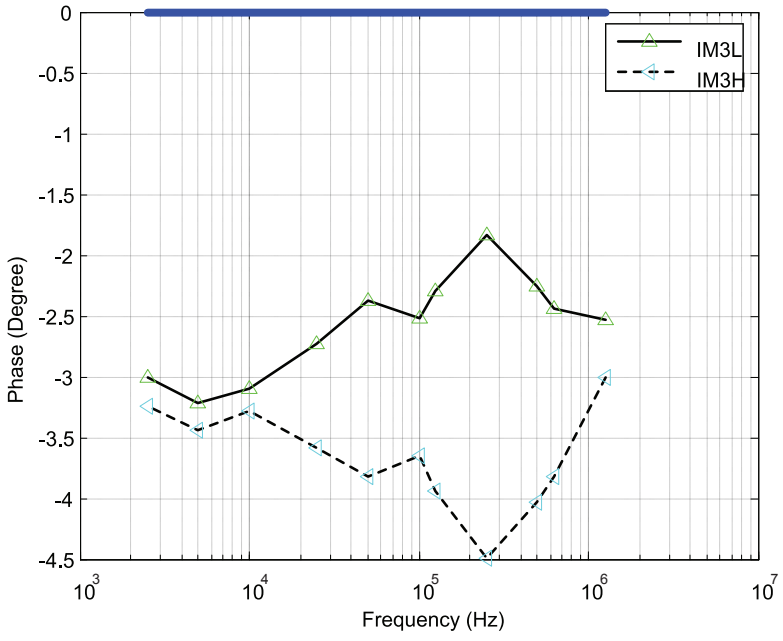


(b)

Figure 3.8: Compare predetermined (solid blue line) and estimated a_1: (a) magnitude; and (b) a_1 phase. Both are plotted versus frequency deviation from the center frequency of the 2-tone.



(a)



(b)

Figure 3.9: Compare predetermined (solid blue line) and estimated a_3: (a) magnitude; and (b) phase. Both are plotted versus frequency deviation from the center frequency of the 2-tone.

is 5% and 5° respectively.

3.2.2 Proposed NVNA Setup

Now that the 1-channel VSA can sample the input excitation and the DUT signals just like a synchronous multi-channel receiver, the make up of the proposed NVNA in Fig. 3.10 becomes obvious. Like the broadband receivers used in previous NVNAs [17], the VSA can simultaneously capture all spectral components within its bandwidth. So except for the presence of switches in the test set, the concepts in these NVNAs can be applied with little modification.

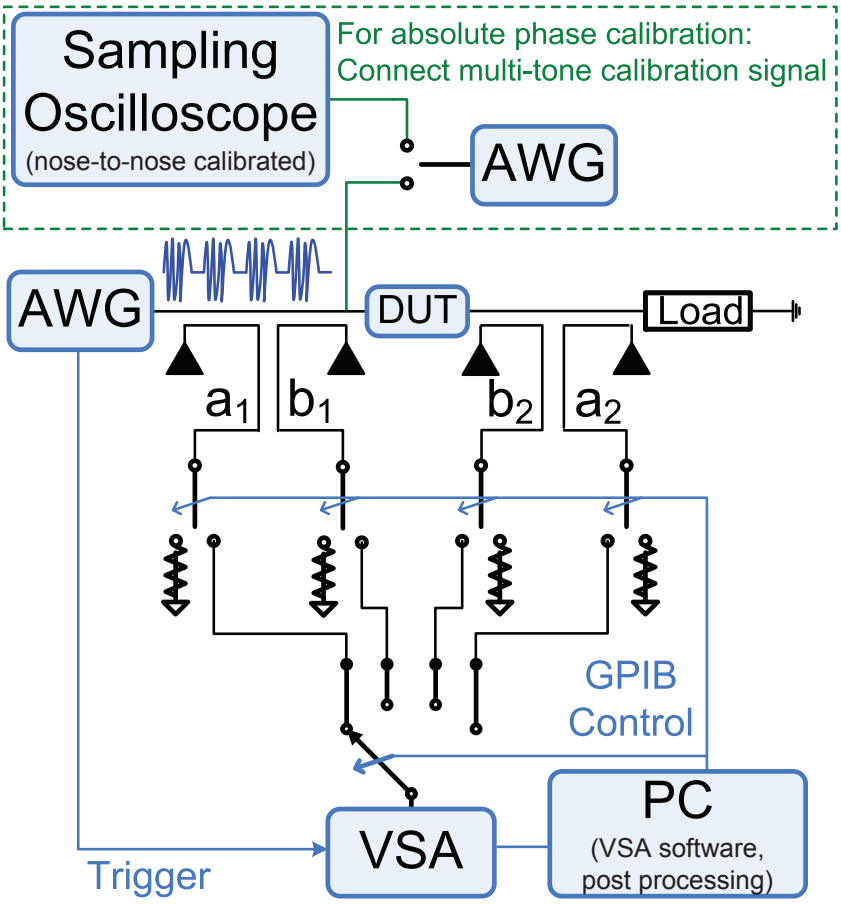


Figure 3.10: Proposed NVNA setup using a 1-channel VSA.

For absolute phase calibration, an AWG can generate a multi-tone calibration signal covering the VSA receiver band, which can be used in place of a multi-harmonic generator. As discussed in Section 3.2.1, the AWG can generate multi-tone signals with a consistent relative phase profile. The systematic receiver phase error is the relative phase difference between measurements from a nose-to-nose calibrated sampling oscilloscope and a VSA. Typically, the two measurements are performed at different time, so alignment is needed to shift the two results to a common time reference.

The calibration process follow essentially the same steps as in Section 2.2.1, which is repeated here for completeness.

1. Establish the linear error model (see (2.1)).
2. Establish the power error $|K(f)|$.
3. Establish the phase errors $\theta_K(f)$. $\theta_K(f)$ is the relative phase difference at each frequency f, between the original nose-to-nose calibrated oscilloscope measurement of the multi-tone calibration signal, and phase of the multi-tone calibration signal measured by the VSA at the DUT plane.

After determining the errors from calibration measurements, the DUT measurements are corrected as follows:

1. Linear correction to the DUT plane using the linear error model.
2. Absolute power correction at the DUT planes using $|K(f)|$.
3. Absolute phase correction at the DUT planes uses $\theta_K(f)$.
4. Align different data sets to a common time reference.

The proposed NVNA can be viewed as a narrowband version of the previous broadband receiver based NVNAs [17], and suitable for characterizing DUT's in-band behavior for a narrowband multi-tone excitation. However the dynamic range is improved, due to the use of a

narrowband tuned receiver instead of a sub-sampling mixer-based front-end where noise at all sidebands of the sampling frequency harmonics are downconverted to the baseband.

3.2.3 Theory of Absolute Phase Calibration

Phase Calibration Measurement

Consider the multi-tone calibration signal, for the mth tone at frequency f_m, the phase of the waveform sampled by the nose-to-nose calibrated oscilloscope at time t_0 is

$$\Psi\mathrm{R}(f_m, t_0) = 2\pi f_m t_0 + \psi(f_m), \tag{3.7}$$

and the phase sampled by the VSA (relative calibration corrected to the DUT plane) at dt time later is

$$\Psi\mathrm{VSA}(f_m, t_0 + dt) = 2\pi f_m(t_0 + dt) + \psi(f_m) + \epsilon(f_m) \tag{3.8}$$

where ϵ denotes the error of the receiver. There are not enough known constraints to solve for the unknown t_0, dt, ψ_{f_m}, and ϵ_{f_m}. Since only the phase dispersion over the receiver band is of interest, the receiver phase error at a chosen tone is ignored, and the error at the other tones are found relative to the chosen tone. Here the arbitrarily chosen tone is at the center of the receiver band f_0 (tone index runs from $-m$ to m). Using the measurement at f_0, the times delay $dt + \frac{\epsilon_{f_0}}{2\pi f_0}$ is found through alignment. Then the calculated receiver phase error ϵ'_{f_m} is

$$\begin{aligned}
\epsilon'(f_m) &= \Psi_{\mathrm{VSA}}\left(f_m, t_0 + dt - (dt + \frac{\epsilon(f_0)}{2\pi f_0})\right) - \Psi\mathrm{R}(f_m, t_0) \\
&= \left(2\pi f_m(t_0 - \frac{\epsilon(f_0)}{2\pi f_0}) + \psi(f_m) + \epsilon(f_m)\right) - (2\pi f_m t_0 + \psi(f_m)) \\
&= \epsilon(f_m) - \epsilon(f_0)\frac{f_m}{f_0}.
\end{aligned} \tag{3.9}$$

Phase Correction of DUT Measurements

To illustrate the phase correction of the DUT measurements, DUT characterization is performed using a two-tone signal having frequencies f_1 and f_2. Given a set of intermodulation frequencies of interest, the incident excitation a_1 at f_1 and f_2, and an arbitrary DUT port wave x at f_{IM} are considered. Using broadband receivers, the spectral components at f_1, f_2, and f_{IM} are concurrently sampled at time t_{D} which may be for a particular tone spacing or input power level. The phase of the sampled waveforms (relative calibration corrected to the DUT plane) are

$$\Phi_{a_1}(f_1, t_{\mathrm{D}}, P_{\mathrm{in}}) = 2\pi f_1 t_{\mathrm{D}} + \phi(f_1) + \varphi(f_1, P_{\mathrm{in}}) + \epsilon(f_1) \tag{3.10}$$

$$\Phi_{a_1}(f_2, t_{\mathrm{D}}, P_{\mathrm{in}}) = 2\pi f_2 t_{\mathrm{D}} + \phi(f_2) + \varphi(f_2, P_{\mathrm{in}}) + \epsilon(f_2) \tag{3.11}$$

$$\Phi_{x}(f_{\mathrm{IM}}, t_{\mathrm{D}}, P_{\mathrm{in}}) = 2\pi f_{\mathrm{IM}} t_{\mathrm{D}} + \phi(f_{\mathrm{IM}}, t_{\mathrm{D}}, P_{\mathrm{in}}) + \epsilon f_{\mathrm{IM}} \tag{3.12}$$

where φ denotes the time-independent phase shift due to the switching attenuator in the signal generator during a power sweep, $\phi(f_1)$ and $\phi(f_2)$ are the particular startup phases of the two-tone sources, and $\phi(f_{\mathrm{IM}}, t_{\mathrm{D}}, P_{\mathrm{in}})$ is phase of the DUT response. Since the DUT response depends on the incident excitation, the phase of the DUT response will depend on both t_{D} and P_{in}. The signal generators' startup phase ϕ is dependent on its power on time, but this time dependence is neglected because it has no significance in the phase calibration procedure.

From these measurements, the receiver phase error $\epsilon^{'}$ in (3.9) is subtracted from (3.10)–(3.12), resulting in

$$\Phi^{'}_{a_1}(f_1, t_{\mathrm{D}}) = 2\pi f_1 t_{\mathrm{D}} + \phi(f_1) + \varphi(f_1, (P_{\mathrm{in}}) + \epsilon(f_0)\frac{f_1}{f_0} \tag{3.13}$$

$$\Phi^{'}_{a_1}(f_2, t_{\mathrm{D}}) = 2\pi f_2 t_{\mathrm{D}} + \phi(f_2) + \varphi f_2(P_{\mathrm{in}}) + \epsilon(f_0)\frac{f_2}{f_0} \tag{3.14}$$

$$\Phi^{'}_{x}(f_{\mathrm{IM}} t_{\mathrm{D}}) = 2\pi f_{\mathrm{IM}} t_{\mathrm{D}} + \phi_{(} f_{\mathrm{IM}}, t_{\mathrm{D}}, P_{\mathrm{in}}) + \epsilon(f_0)\frac{f_{\mathrm{IM}}}{f_0}. \tag{3.15}$$

To compare data taken at different times t_{D}, time delay is found from aligning the excitation,

as long as the phase relation of the two-tone excitation is maintained. The time delay adjusted data is the absolute phase measurement.

As pointed out in Section 2.2.1, data from excitation with different tone-spacings cannot be aligned. While data over a power sweep can be aligned, there are phase contributions $\varphi(f_1, P_{\text{in}})$ and $\varphi(f_2, P_{\text{in}})$ due to the switching attenuator in the signal generator in the data, which requires that phase shifts be equal for all the excitation tones. Typically this condition is satisfied for narrowband excitations. More importantly, such nonconstant phase contribution makes it difficult for a designer to evaluate the phase change introduced by the DUT. To find the phase contribution due to the DUT, the time-invariant phase from Section 2.2.3 can be applied to the data.

Time-Invariant Phase

For simplicity the DUT response is represented as a Volterra series. For a 2-tone excitation, the response at any IM frequency f_{IM} is

$$\begin{aligned} Y(f_{\text{IM}}) &= \sum_{n} \sum_{\vec{m}_{n,\text{IM}}} \frac{(n:\vec{m})}{2^n} \, (A_2^*)^{m_{-2}} (A_1^*)^{m_{-1}} (A_1)^{m_1} (A_2)^{m_2} \\ &\times H_n\big(f_{-2_{(1)}}, \ldots, f_{-2_{(m_{-2})}}, \; f_{-1_{(1)}}, \ldots, f_{-1_{(m_{-1})}}, \\ &\qquad f_{1_{(1)}}, \ldots, f_{1_{(m_1)}}, \; f_{2_{(1)}}, \ldots, f_{2_{(m_2)}}\big) \end{aligned} \tag{3.16}$$

where H_n denotes the nth order NLTF, $\vec{m}_{n,\text{IM}}$ denotes the nth order contributing mixing vectors, and A_i denotes the complex amplitude of the ith excitation tone. Using a shortened notation $H_n(\vec{m})$ to denote H_n evaluated at the mixing product corresponding to $\vec{m}$, (3.16) can

be rearranged as

$$\begin{aligned} Y(f_{\mathrm{IM}}) &= \sum_{n}\sum_{\vec{m}_{n,\mathrm{IM}}} \frac{(n:\vec{m})}{2^n}\ |A_2|^{m_{-2}}|A_1|^{m_{-1}}|A_1|^{m_1}|A_2|^{m_2}|H_n(\vec{m})| \\ &\times\ e^{j\left(\theta_{A_1}(m_1-m_{-1})+\theta_{A_2}(m_2-m_{-2})+\theta_{H_n(\vec{m})}\right)} \\ &= e^{j\left(\theta_{A_1}(m_1-m_{-1})+\theta_{A_2}(m_2-m_{-2})\right)} \\ &\times \left[\sum_{n}\sum_{\vec{m}_{n,\mathrm{IM}}} \frac{(n:\vec{m})}{2^n}\ |A_2|^{m_{-2}}|A_1|^{m_{-1}}|A_1|^{m_1}|A_2|^{m_2}|H_n(\vec{m})|\ e^{j\theta_{H_n(\vec{m})}}\right] \\ &= e^{j\left(\theta_{A_1}(m_1-m_{-1})+\theta_{A_2}(m_2-m_{-2})\right)}\ \left[|Y(f_{\mathrm{IM}})|\ e^{j\theta_{\mathrm{DUT}}(|A_1|,|A_2|,H)}\right]. \end{aligned} \tag{3.17}$$

The second step is possible because each component of the double sum contributes to the same frequency $f_{\mathrm{IM}} = (m_1 - m_{-1})f_1 + (m_2 - m_{-2})f_2$. In the last step, the double sum is rewritten as a complex number with the phase θ_{DUT} depending only on the excitation magnitude and the DUT NLTF.

Applying the time-zero cancelation definition in Section 2.2.4, the time-invariant phase is

$$\begin{aligned} \phi'_{f_{\mathrm{IM}}} &= \phi_{f_{\mathrm{IM}}} - (\theta_{A_1}(m_1 - m_{-1}) + \theta_{A_2}(m_2 - m_{-2})) \\ &= \theta_{\mathrm{DUT}}(|A_1|, |A_2|, H). \end{aligned} \tag{3.18}$$

In application to the current measurement, the switching attenuator contributions $\varphi(f_1, P_{\mathrm{in}})$ and $\varphi(f_2, P_{\mathrm{in}})$ to the excitation tone phase θ_{A_1} and θ_{A_2} respectively will be canceled. Note also that the frequency-weighted residual phase error of the receiver, $\epsilon(f_0)\frac{f_{\mathrm{IM}}}{f_0}$, is also canceled in the time-invariant phase definition.

3.2.4 Experimental Results

For verification, measurements from the proposed 1-channel VSA-based NVNA and a commercial NVNA system similar to [17] are compared. Due to limited number of switches available, the setup in Fig. 3.10 is simplified as in Fig. 3.11 for transmission response measurement of

well-matched DUTs using two switches.

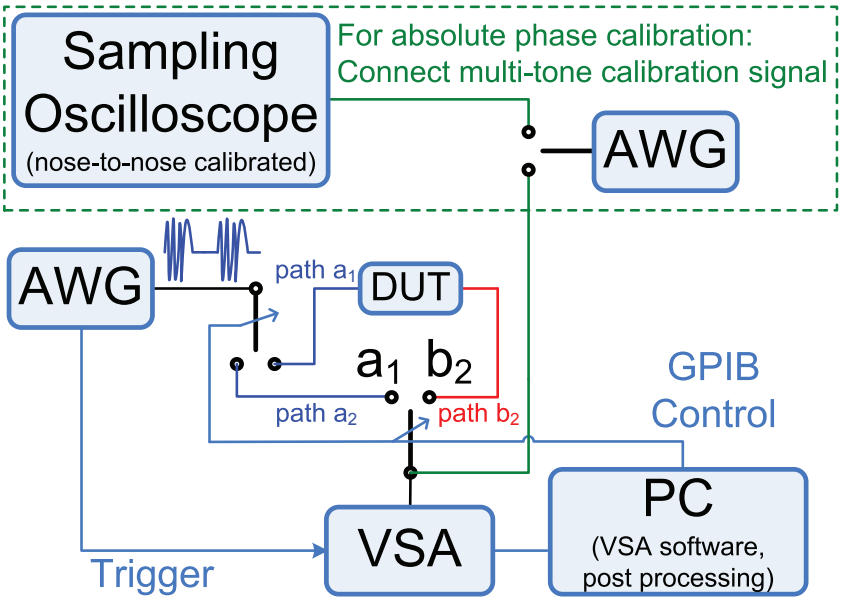


Figure 3.11: For transmission response measurement of well-matched DUTs, a simplified version of the proposed 1-channel VSA-based NVNA, which is the same as Fig. 3.2.

In place of full relative calibration, only a thru-response calibration is possible, which is suitable for well-matched DUTs. To ensure that the DUT and VSA measure the same signal, the path a_1 and path a_2 are matched by using interconnects of the same length. The path b_2 is characterized using a VNA. For absolute calibration, the VSA power and phase errors are determined at the VSA input rather than at the DUT plane. During the correction of the VSA measurements, first the absolute power and phase error is remove, then the output b_2 is corrected to the DUT plane.

The long-term memory effect is characterized for an amplifier (Amplica SD513301). The dynamic frequency 2-tone signal at 2 GHz center frequency is used, with tone-spacing varied from 2.5 kHz to 1 MHz. Fig. 3.12 shows the linear transmission response at the 2-tone input frequencies f_1 and f_2, which is plotted versus frequency deviation from the center frequency

of the 2-tone. Note that at low tone spacings, the commercial NVNA system interpolates over the missing data points. Also included is the low input power S_{21} measurement from a 2-port SOLT calibrated VNA. With 10 measurements averaged for the proposed setup, the gain and phase are within 0.2 dB and 2° respectively. Note that the PA output is attenuated by an amount equaling its gain, in order to maintain the same attenuator setting in the VSA.

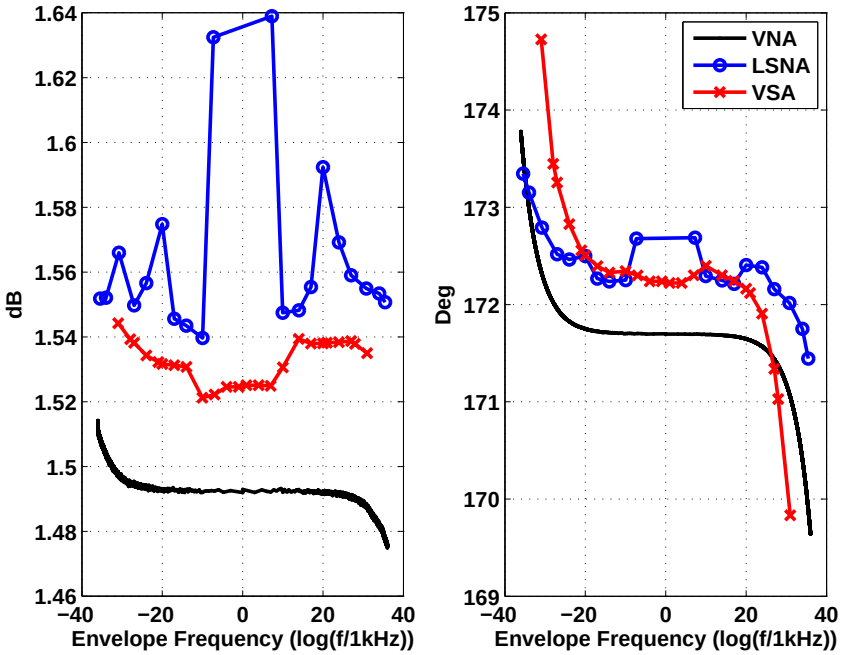


Figure 3.12: Compare the linear transmission response measured at 2-tone input frequencies f_1 and f_2, using the proposed 1-channel VSA-based NVNA, a commercial NVNA system, and a VNA. The frequency deviation are in dB scale referenced to 1 kHz.

Fig. 3.13 shows the nonlinear transmission response at frequencies $2f_1 - f_2$ and $2f_2 - f_1$ for several input power levels. The NLTF (a variant of the time-invariant phase definition described in Section 2.2.4) is used to represent the change in magnitude and phase caused by the DUT. In Fig. 3.13, note the increasing contribution of the third-order response as the

input power increases. At tone spacings below 10 kHz, the phase asymmetry in the third-order characteristic converges to what a typical AM-AM and AM-PM measurement of a quasi-static response would yield. With 10 measurements averaged for the proposed setup, the amplitude and phase are within 10% and 5° respectively.

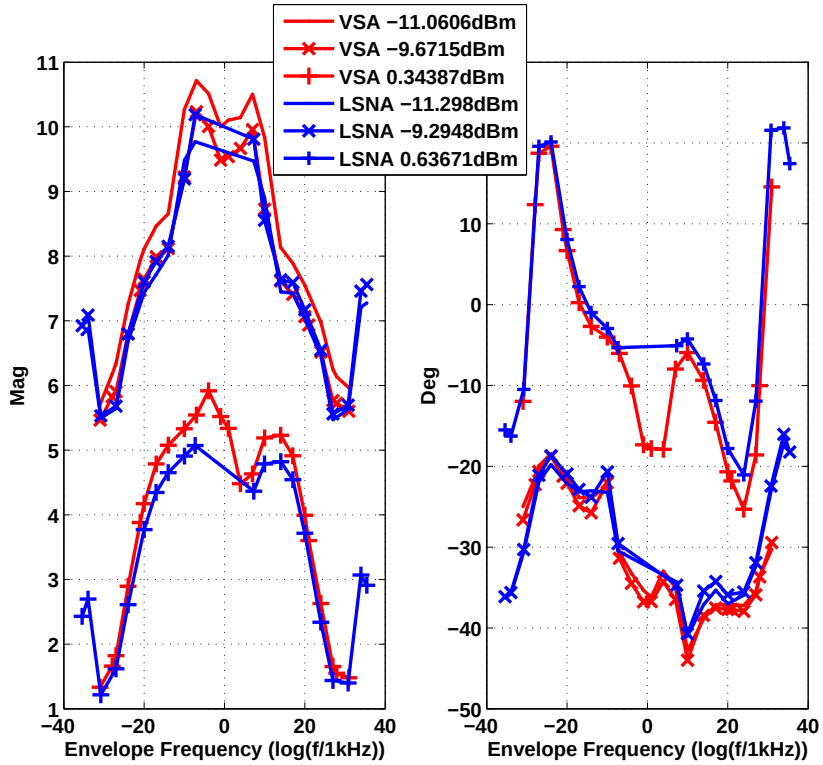


Figure 3.13: Comparison the nonlinear transmission response at frequencies $2f_1 - f_2$ and $2f_2 - f_1$ for several input power levels, using the proposed 1-channel VSA-based NVNA and a commercial NVNA system. The frequency deviation are in dB scale referenced to 1 kHz.

3.2.5 Summary

To use a 1-channel receiver to characterize a multi-port DUT, the DUT signals are sequentially sampled to mimic a synchronous multi-channel receiver. To maintain the same timing relation

between sequentially sampled waveforms as if they were synchronously sampled, excitation waveforms are designed to be on a DFT grid that is commensurate with the AWG and the VSA sampler time base. In the post processing step, the entire VSA sampled waveform is divided into segments corresponding to each sequentially sampled DUT signals. The start and end of each segment can be determined, since the excitation duration and the switching interval are specified in terms of the smallest period in the excitation waveform.

Once the 1-channel VSA can be made to behave like a multi-channel receiver, concepts from absolute phase measurement using broadband receivers can be applied with minor modifications. Namely, a multi-tone signal generated by an AWG replaces the multi-harmonic generator, and a nose-to-nose calibrated oscilloscope provides the reference measurements for VSA phase error correction. The quantitative theory of absolute phase calibration for this type of NVNA is also presented.

The proposed NVNA can be viewed as a narrowband version of the previous broadband receiver based NVNAs [17], and suitable for characterizing DUT's in-band behavior for a narrowband multi-tone excitation. However the dynamic range is improved, due to the use of a narrowband tuned receiver instead of a sub-sampling mixer-based front-end where noise at all sidebands of the sampling frequency harmonics are downconverted to the baseband.

3.3 Novel 4-Port VNA Based NVNA

Typical VNAs have instantaneous bandwidth of 250 kHz and measure only one discrete tone at a time. As discussed in Section 2.2.1, a solution to absolute phase measurement using narrowband receivers requires that the multi-harmonic generator be concurrently measured with the DUT signals. An effective bandwidth of 20 GHz is achieved by sequentially measuring the spectral components of interest.

In the previous 4-port VNA-based NVNA [5], the multi-harmonic generator output is measured in the reference receiver which is the 5th channel normally used to sample the 1-tone excitation in S-parameter measurements. As discussed in Section 3.1.1, the commercially available

multi-harmonic generator limits the narrowest tone-spacing in multi-tone testing to 10 MHz, while other possible existing experimental types are not readily available to the practicing engineer.

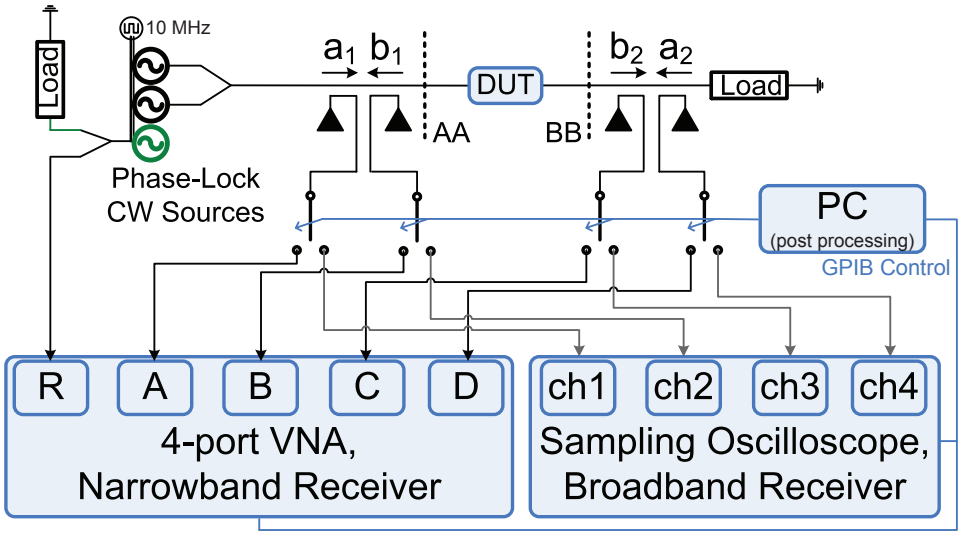
Section 3.3.1 presents a 4-port VNA-based NVNA used with common benchtop instruments: phase-locked signal generators and an equivalent-time sampling oscilloscope. In place of a multi-harmonic generator, a signal generator phase-locked to the multi-tone excitation is used to maintain phase reference across a power sweep, and can now tune to a tone-spacing on the order of 1 Hz. The power sweep data is acquired by the high dynamic range heterodyne receiver-based VNA, and corrected in absolute phase by characterizing the DUT at a chosen power level using a low dynamic range equivalent sampling oscilloscope based NVNA.

Section 3.3.2 presents the theory of absolute phase calibration procedure for the proposed NVNA, which will clarify the measurement procedure. The procedure is compared to that used in the previous 4-port VNA-based NVNA [5].

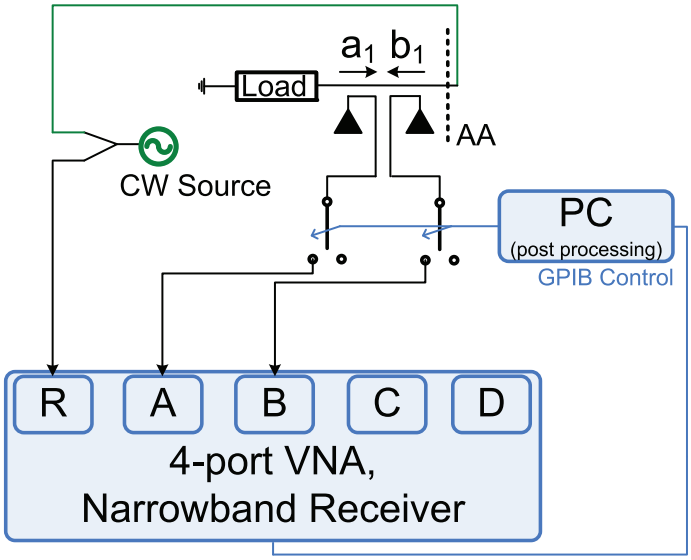
3.3.1 Proposed NVNA Setup

The architecture of the hybrid NVNA is shown in Fig. 3.14(a). Noticeably absent is a multi-harmonic generator to establish an absolute phase reference. A typical DUT measurement sweeps over the tone-spacing and power. The four-port VNA portion is used to make sequential measurements at each spectral component during a two-tone power sweep. The output of the top two of the RF generators are combined as the two-tone test signal. The third generator provides the relative phase reference for the spectral component being measured. The output of this reference RF generator is measured by the VNA reference receiver. The load connected to the reference RF generator is required, because the signal split from that branch will be used later in the phase calibration measurement which is shown in Fig. 3.14(b).

For a fixed tone-spacing, the DUT response at each IM frequency is sequentially measured, with the VNA and the reference signal generator tuned to each IM frequency. For a particular IM frequency, the excitation power is then swept. Power sweeps usually require switching



(a)



(b)

Figure 3.14: Connection diagrams of the proposed hybrid NVNA: (a) for DUT measurement; and (b) for phase calibration measurement. The proposed hybrid NVNA system combines a sampling oscilloscope-based NVNA, and a four-port VNA-based NVNA, with phase-locked signal generators for multi-tone testing.

attenuators in the signal generator, but since the reference signal generator is kept at constant power level, its constant output phase can be used as the reference during the power sweep. However this phase is not maintained as the reference signal generator is tuned to each IM frequency.

In the measurement performed by the VNA portion of the hybrid NVNA, the phase-locked sources, together with the phase synchronous VNA receivers, serve as the phase transfer mechanism which relates the sequential measurements during a power sweep at a particular spectral component. As such, the RF generators and the receiver channels must be phase synchronous. A common reference is used to phase-lock (not frequency-lock) three RF generators, although the relative phases of the generators are random. If the frequency of one of the RF generators is changed, these relative phases will be randomized but the phase relationship is fixed if only the signal level changes. The phase-locked sources and frequency-locked sources differ in the time duration of their true phase-lock. In typical frequency-locked sources, the synthesizer architecture uses dithering to reduce the level of spurious tones. As a result, the random phase variations of the independent fractional synthesizers only have a particular time-varying phase relationship on average. This has been empirically verified by comparing the phases measured by two receiver channels of the VNA. The outputs of two phase-locked sources such as the Marconi 2024 are phase-synchronous for a long time duration (typically for several hours), while the outputs of two frequency-locked sources such as the Agilent E4438 have random phase variation.

At a threshold power level, P_t, during the power sweep, the port wave signals are switched from the receivers of the VNA to the inputs of the sampling oscilloscope. Then the nose-to-nose calibrated sampling oscilloscope is used to synchronously measure all spectral components on a frequency grid determined by the sampling rate and sampling duration. At P_t the absolute phases of the spectral components are available from the (phase-calibrated) oscilloscope-based measurements. Since the spectral components are also measured by the VNA at P_t, an absolute phase calibration becomes available for the signals measured by the VNA.

Calibration

Calibration of the NVNA requires the establishment of calibrated reference planes AA and BB in Fig. 3.14(a). Full calibration requires linear (or relative) calibration, power calibration, and absolute phase calibration. Linear calibration determines the seven e's in the 4×4 matrix in (2.1) using a modified version of one of many procedures developed for linear VNAs [1], [20]. In linear calibration measurement, the hybrid NVNA connection in Fig. 3.14(a) is altered by replacing the DUT with calibration standards, replacing the two-tone source with a one-tone source, and disregarding the reference receiver (R) measurements. The seven e's are then computed from the calibration measurements using known procedures. After linear correction to the DUT plane using the linear error model, only the relative power and phase between the port waves at each frequency have the correct values. For absolute calibration, one of the port waves must be corrected in absolute power and phase, so that the other port waves can be corrected accordingly.

For broadband receiver-based NVNAs such as the sub-sampling mixer-based NVNA in [1], determining the terms in (2.1) provides complete calibration. In determining $\theta_K(f)$, which accounts for errors due to receiver phase dispersion, a multi-harmonic generator output is characterized using two of the receivers at port 1.

However for narrowband receiver-based NVNAs such as the four-port VNA-based NVNA in [5], absolute phase calibration requires additional correction steps besides determining the terms in (2.1). Here the phase difference between the reference receiver R measurements and the DUT response (measured by receivers A-D), is added to the pre-determined phase value of the multi-harmonic generator to obtain the absolute phase of the port waves.

For the hybrid NVNA, the linear and power calibration procedures are the same for the oscilloscope- and heterodyne-based measurements. However the two types of measurements require different absolute phase calibration procedures. Oscilloscope-based measurements are calibrated in absolute phase using the procedure described in [1] and used here. The absolute phase calibration procedure used here for heterodyne-based measurements differs from that in

[5] and is described below.

The calibration procedure is made up of the calibration measurement part and the error correction part. The outline of the calibration measurement is as follows:

1. Establish the linear error model (see (2.1)).

2. Establish the power error $|K(f)|$.

3. Establish the phase error $\theta_K(f)$. $\theta_K(f)$ is the relative phase difference at each frequency f, between the reference receiver (R) used to measure the reference signal generator, and each of the other receivers (A-D) used to measure the DUT signals at the DUT plane.

Just as for the four-port VNA-based NVNA, the heterodyne-based measurements use the reference receiver (R) as a phase transfer mechanism between the sequentially sampled spectral components. As such, the relative phase error of the phase transfer mechanism as represented by $\theta_K(f)$ must be determined. This calibration step was not described in [5], but is obviously necessary since the reference receiver and the other receivers in a VNA give different measurements for the same signal. For this phase calibration measurement, the VNA portion of the hybrid NVNA is connected as in Fig. 3.14(b). The output of the one-tone source is concurrently measured in receiver R and receivers A and B connected at the DUT plane AA at each intermodulation frequency of interest. The load in the figure represents the off-state output impedance presented by the two-tone source with its RF output disabled. The analytical determination of $\theta_K(f)$ from the phase calibration measurements will be presented later.

After determining the errors from calibration measurements, the DUT measurements are corrected as follows:

1. Linear correction to the DUT plane using the linear error model.

2. Absolute power correction at the DUT planes using $|K(f)|$.

3. Absolute phase correction at the DUT planes uses $\theta_K(f)$, with the phase difference established over a power sweep between the concurrently measured reference signal generator

and the DUT signals at the DUT plane, and with the phase of the DUT signals characterized at a specific power level by the sampling oscilloscope portion.

For the four-port VNA-based NVNA and the hybrid NVNA, $\theta_K(f)$ only corrects the relative phase error of the phase transfer mechanism, so additional phase correction steps are required. For the hybrid NVNA, this is the third correction step above. In the following, descriptions are given first for the calibration step to determine $\theta_K(f)$, then for the absolute phase correction applied to the DUT measurements.

Assumptions

The phase-locked signal generator must satisfy several requirements for the proposed approach to work. Firstly, stable phase-lock is needed for the power sweep over each spectral component. This is easily met as the Marconi 2024 can provide hours of stable phase-lock, and has been successfully used in a high dynamic range feed-forward cancellation system [77], [78].

Secondly, the power output of the multi-tone source must be repeatable during the power sweep to provide repeatable input. This has been empirically verified. Since the phase reference signal generator is set to constant output power this is not an issue.

Most importantly, stable phase-lock is also needed during the time all of the spectral components are measured by the VNA and oscilloscope-portions of the hybrid NVNA. This is so that the multi-tone sources are locked in phase, and provide repeatable input for the power sweep at all spectral component. For example, if power is swept over 30 dB with 1 dB resolution and each point takes 1 s to measure by the VNA, then each spectral component takes 30 s. Thus for two-tone testing, measuring all fifth-order responses, which include 30 spectral components, will take 15 min. So the hours of stable phase-lock time available is sufficient, and plenty of the time is available for the additional oscilloscope acquisition time. For sweeps over tone-spacing, the input frequency changes and the phase-lock is lost in any case, and complete phase recalibration is (automatically) required.

Previous NVNA Setup

The diagram of the previous NVNA in [5] is shown in Fig. 2.5. For the same DUT measurement sweeps over the tone-spacing and power, the order between power sweep or sequential IM measurement is not important. This is because the multi-harmonic generator maintains the phase reference. So unlike for the proposed setup, the power sweep does not have to be in the inner most measurement loop. The phase calibration is simplified using the characterized phase of the multi-harmonic generator.

3.3.2 Theory of Absolute Phase Calibration

Phase Calibration Measurement

Connecting the one-tone source as in Fig. 3.14(b), the port waves at the DUT plane AA must satisfy

$$M_{\text{SigGen}}(f) = b_1(f) - \Gamma_{\text{SigGen}}(f)a_1(f) \tag{3.19}$$

where M_{SigGen} is the output of the one-tone source, b_1 and a_1 are related to the VNA A and B receiver measurements from (2.1), and Γ_{SigGen} is its reflection coefficient characterized at the DUT plane AA using a VNA. Making the appropriate substitutions, (3.19) becomes

$$\begin{aligned} M_{\text{SigGen}}(f) &= K(f)\left[e_{21}(f)A(f) + e_{22}(f)B(f)\right] \\ &- \Gamma_{\text{SigGen}}(f)K(f)\left[A(f) + e_{12}(f)B(f)\right]. \end{aligned} \tag{3.20}$$

Then solving for $K(f)$,

$$K(f) = \frac{M_{\text{SigGen}}(f)}{\left[e_{21}(f)A(f) + e_{22}(f)B(f)\right] - \Gamma_{\text{SigGen}}(f)\left[A(f) + e_{12}(f)B(f)\right]} \tag{3.21}$$

which has an angle $\theta_K(f)$. Since the relative phase error of the phase transfer mechanism is to be determined, the measured phase of receiver R is taken as the phase of M_{SigGen}. Noting that A and B are linearly related to M_{SigGen}, (3.21) implies that $\theta_K(f)$ is the relative phase difference between the measurement at receiver R and the mismatch corrected b_1. The power of M_{SigGen} has not been specified because it is not necessary in determining $\theta_K(f)$.

Phase Correction of DUT Measurements

In the following discussion, DUT characterization is performed using a two-tone signal having frequencies f_1 and f_2, so the incident excitation a_1 at f_1 and f_2 are considered. Also given a set of intermodulation frequencies of interest, an arbitrary DUT port wave x at f_{IM} is considered.

For the VNA measurement, the responses at f_1, f_2, and f_{IM} are sequentially measured at times t_{f_1}, t_{f_2}, and $t_{f_{\text{IM}}}$ respectively. After correction, see (2.1), the phase of the sampled waveforms at the DUT plane are

$$\Phi_{a_1}(f_1, t_{f_1}, P_{\text{in}}) = 2\pi f_1 t_{f_1} + \phi(f_1, t_{\text{D}}) + \varphi(f_1, P_{\text{in}}) \tag{3.22}$$

$$\Phi_{a_1}(f_2, t_{f_2}, P_{\text{in}}) = 2\pi f_2 t_{f_2} + \phi(f_2, t_{\text{D}}) + \varphi(f_2, P_{\text{in}}) \tag{3.23}$$

$$\Phi_{x}(f_{\text{IM}}, t_{f_{\text{IM}}}, P_{\text{in}}) = 2\pi f_{\text{IM}} t_{f_{\text{IM}}} + \phi(f_{\text{IM}}, t_{\text{D}}, P_{\text{in}}) \tag{3.24}$$

where $\phi(f_1, t_{\text{D}})$ and $\phi(f_2, t_{\text{D}})$ are the particular startup phases of the two-tone sources, and $\phi(f_{\text{IM}}, t_{\text{D}}, P_{\text{in}})$ is phase of the DUT response. When setting the frequencies of the two-tone sources for a particular tone-spacing, the two sources phase-lock at a particular relative phase which is held during the sequential power-swept measurements of all IM products. This phase profile changes if the frequencies of the two-tone sources are reset, and this effect is captured by the experiment time dependence t_{D}. During a power sweep, the switching attenuators in the two-tone sources also contribute phase shifts $\varphi(f_1, P_{\text{in}})$ and $\varphi(f_2, P_{\text{in}})$ which are time independent but vary with the power P_{in} of the two-tone signal. Since the DUT response depends on the incident excitation, the phase of the DUT response will depend on both t_{D} and

P_{in}.

The reference signal generator output is concurrently sampled by the reference receiver R at measurement times t_{f_1}, t_{f_2}, and $t_{f_{\text{IM}}}$, which correspond to measurement frequencies f_1, f_2, and f_{IM}. The phase values are

$$\Psi_{\text{R}}(f_1, t_{f_1}) = 2\pi f_1 t_{f_1} + \psi(f_1, t_{f_1}) \tag{3.25}$$

$$\Psi_{\text{R}}(f_2, t_{f_2}) = 2\pi f_2 t_{f_2} + \psi(f_2, t_{f_2}) \tag{3.26}$$

$$\Psi_{\text{R}}(f_{\text{IM}}, t_{f_{\text{IM}}}) = 2\pi f_{\text{IM}} t_{f_{\text{IM}}} + \psi(f_{\text{IM}}, t_{f_{\text{IM}}}) \tag{3.27}$$

where ψ indicates the startup phase of the reference signal generator each time it is tuned to a different measurement frequency. Note that in (3.25)–(3.27) there is no dependence on P_{in} because the reference signal generator maintains a constant output power, and thus the same relative phase during a power sweep.

The phase correction procedure can now be described. First the phase is corrected relative to the reference receiver as follows:

$$\begin{aligned} \Phi_{a_1}^{'}(f_1, t_{f_1}, P_{\text{in}}) &= \Phi_{a_1}(f_1, t_{f_1}, P_{\text{in}}) - \Psi_{\text{R}}(f_1, t_{f_1}) \\ &= \phi(f_1, t_{\text{D}}) + \varphi(f_1, P_{\text{in}}) - \psi(f_1, t_{f_1}) \end{aligned} \tag{3.28}$$

$$\begin{aligned} \Phi_{a_1}^{'}(f_2, t_{f_2}, P_{\text{in}}) &= \Phi_{a_1}(f_2, t_{f_2}, P_{\text{in}}) - \Psi_{\text{R}}(f_2, t_{f_2}) \\ &= \phi(f_2, t_{\text{D}}) + \varphi(f_2, P_{\text{in}}) - \psi(f_2, t_{f_2}) \end{aligned} \tag{3.29}$$

$$\begin{aligned} \Phi_{x}^{'}(f_{\text{IM}}, t_{f_{\text{IM}}}, P_{\text{in}}) &= \Phi_{x}(f_{\text{IM}}, t_{f_{\text{IM}}}, P_{\text{in}}) - \Psi_{\text{R}}(f_{\text{IM}}, t_{f_{\text{IM}}}) \\ &= \phi(f_{\text{IM}}, t_{\text{D}}, P_{\text{in}}) - \psi(f_{\text{IM}}, t_{f_{\text{IM}}}). \end{aligned} \tag{3.30}$$

At this stage the phase of each spectral component has been self-consistently established. However the phase relationship of different spectral components is arbitrary. The phase of the reference signal generator varies as it is tuned to different measurement frequencies. The next step in the phase correction procedure is to determine an absolute phase reference.

The key is to cross reference the corrected phase from the VNA-based data, e.g. (3.28)–(3.30), with the absolute phase derived from measurements made with the sampling oscilloscope. The oscilloscope-based measurements at a chosen excitation power $P_{\rm t}$ provides the needed absolute phase reference. All spectral components are captured by the oscilloscope at the same time, $t_{\rm DSO}$. The relative phase profile of the two-tone sources is set at experiment time $t_{\rm D}$. For the same incident excitation frequency and power level, the DUT response will be the same, and the absolute phase values corrected to the DUT plane are

$$\Phi_{a_1,\rm DSO}(f_1, t_{\rm DSO}, P_{\rm t}) = 2\pi f_1 t_{\rm DSO} + \phi(f_1, t_{\rm D}) + \varphi(f_1, P_{\rm t}) \tag{3.31}$$

$$\Phi_{a_1,\rm DSO}(f_2, t_{\rm DSO}, P_{\rm t}) = 2\pi f_2 t_{\rm DSO} + \phi(f_2, t_{\rm D}) + \varphi(f_2, P_{\rm t}) \tag{3.32}$$

$$\Phi_{x,\rm DSO}(f_{\rm IM}, t_{\rm DSO}, P_{\rm t}) = 2\pi f_{\rm IM} t_{\rm DSO} + \phi(f_{\rm IM}, t_{\rm D}, P_{\rm t}) \tag{3.33}$$

which are repeatable to within an unknown phase $2\pi f_{\rm IM} t_{\rm DSO}$ at each IM frequency.

The final phase correction step makes use of the absolute phase reference established by the oscilloscope measurements to relate the phase of different spectral components measured by the VNA. The phase values (3.28)–(3.30) measured by the VNA over a power sweep, are shifted to the absolute phase (3.31)–(3.33) measured by the oscilloscope at power level $P_{\rm t}$, as follows:

$$\begin{aligned}\Phi''_{a_1}(f_1, t_{\rm DSO}, P_{\rm in}) &= \Phi_{a_1,\rm DSO}(f_1, t_{\rm DSO}, P_{\rm t}) + \left(\Phi'_{a_1}(f_1, t_{f_1}, P_{\rm in}) - \Phi'_{a_1}(f_1, t_{f_1}, P_{\rm t})\right) \\ &= 2\pi f_1 t_{\rm DSO} + \phi(f_1, t_{\rm D}) + \varphi(f_1, P_{\rm in}) \end{aligned} \tag{3.34}$$

$$\begin{aligned}\Phi''_{a_1}(f_2, t_{\rm DSO}, P_{\rm in}) &= \Phi_{a_1,\rm DSO}(f_2, t_{\rm DSO}, P_{\rm t}) + \left(\Phi'_{a_1}(f_2, t_{f_2}, P_{\rm in}) - \Phi'_{a_1}(f_2, t_{f_2}, P_{\rm t})\right) \\ &= 2\pi f_2 t_{\rm DSO} + \phi(f_2, t_{\rm D}) + \varphi(f_2, P_{\rm in}) \end{aligned} \tag{3.35}$$

$$\begin{aligned}\Phi''_{x}(f_{\rm IM}, t_{\rm DSO}, P_{\rm in}) &= \Phi_{x,\rm DSO}(f_{\rm IM}, t_{\rm DSO}, P_{\rm t}) + \left(\Phi'_{x}(f_{\rm IM}, t_{f_{\rm IM}}, P_{\rm in}) - \Phi'_{x}(f_{\rm IM}, t_{f_{\rm IM}}, P_{\rm t}\right) \\ &= 2\pi f_{\rm IM} t_{\rm DSO} + \phi(f_{\rm IM}, t_{\rm D}, P_{\rm in}) \end{aligned} \tag{3.36}$$

which is in the same form as (3.13)–(3.15). These are the so-called absolute phase measurement because receiver and interconnect phase dispersion across a broad range of spectral components

being measured have been corrected.

Note that (3.34)–(3.36) are repeatable to within an unknown phase $2\pi f_{\mathrm{IM}} t_{\mathrm{DSO}}$ at each IM frequency. This is because phase is intrinsically a relative quantity which depends on the sampling time. Even during the characterization of a phase transfer standard, different phase values are measured at different times. For example, even though a multi-harmonic generator is designed to maintain a constant phase relationship among the spectral components, it has a time-varying phase at its fundamental component. Phase-locking the signal source and the receivers cannot solve the problem because of the imperfect phase-locking mechanism which exhibits long-term drift, and due to the unknown initial relative phase established by the phase-lock.

Absolute phase measurement also requires the ability to compare measurements made at different times. The solution is to measure the incident excitation simultaneously with the DUT response. Then to compare measurements made at different times, the different sets of data are shifted to a common time origin using an alignment procedure [27] which removes the arbitrary time delays between the data sets. Assuming one set of measurements is a time delayed version of the other, the time delay between the incident excitations can be found, and all scattered port waves are adjusted by this time delay. This requires that the incident excitations be of the same type, such as two-tone signals with the same tone-spacing, amplitude and phase profile.

The other solution is to apply the time-invariant phase definition [27] to find the phase relation between the incident excitation and the scattered port waves. This yields more intuitive results in typical DUT nonlinear characterization since the engineer is interested in the phase change caused by the DUT as a function of input power or tone-spacing. However as discussed in [27], the alignment procedure is more robust (than the time-invariant procedure) from noise considerations. This restricts the application of the time-invariant phase definition to narrow-band signals such as two-tones, or weakly nonlinear characterization for wideband signals such as multi-tones. Nevertheless the intuitive results of time-invariant phase definition is invaluable in design, since measurements of different tone-spacings can be compared, and the phase con-

tributions from the switching attenuator in the signal generator can also be removed. Strictly speaking, these phase contributions are part of the signal being characterized. So absolute phase measurement using the alignment procedure is best suited to signal characterization.

Comparison to Previous NVNA

For the previous 4-port VNA-based NVNA [5], the step for finding the phase error between the reference and the other receiving channels, is the same as for the proposed NVNA, except for the use of a multi-harmonic generator. So the analysis is skipped here. In the following, ψ denotes the phase of the multi-harmonic generator characterized by the nose-to-nose calibrated sampling oscilloscope

The difference is in the correction of the measurement. The phase of the sampled DUT signals are the same as in (3.22)–(3.24). The phase of the concurrently sampled multi-harmonic generator output are

$$\Psi_{\mathrm{R}}(f_1, t_{f_1}) = 2\pi f_1 t_{f_1} + \psi(f_1, t_{\mathrm{D}}) \tag{3.37}$$

$$\Psi_{\mathrm{R}}(f_2, t_{f_2}) = 2\pi f_2 t_{f_2} + \psi(f_2, t_{\mathrm{D}}) \tag{3.38}$$

$$\Psi_{\mathrm{R}}(f_{\mathrm{IM}}, t_{f_{\mathrm{IM}}}) = 2\pi f_{\mathrm{IM}} t_{f_{\mathrm{IM}}} + \psi(f_{\mathrm{IM}}, t_{\mathrm{D}}) \tag{3.39}$$

where the t_{D} time dependence in ψ indicates the unknown sampling time. However the difference from (3.25)–(3.27), is that the multi-harmonic generator holds the same phase during the sequential measurement of all the spectral components.

The corrected phase is

$$\begin{aligned}
\Phi'_{a_1}(f_1, t_{\mathrm{D}}) &= \psi(f_1) + (\Phi(f_1, t_{f_1}) - \Psi_{\mathrm{R}}(f_1, t_{f_1})) \\
&= \psi(f_1) + \phi(f_1, t_{\mathrm{D}}) + \varphi(f_1, P_{\mathrm{in}}) - \psi(f_1, t_{\mathrm{D}}) \\
&= 2\pi f_1 t_{\mathrm{D}} + \phi(f_1, t_{\mathrm{D}}) + \varphi(f_1, P_{\mathrm{in}}) \qquad (3.40) \\
\Phi'_{a_1}(f_2, t_{\mathrm{D}}) &= \psi(f_2) + (\Phi(f_2, t_{f_2}) - \Psi_{\mathrm{R}}(f_2, t_{f_2})) \\
&= \psi(f_2) + \phi(f_2, t_{\mathrm{D}}) + \varphi(f_2, P_{\mathrm{in}}) - \psi(f_2, t_{\mathrm{D}}) \\
&= 2\pi f_2 t_{\mathrm{D}} + \phi(f_2, t_{\mathrm{D}}) + \varphi(f_2, P_{\mathrm{in}}) \qquad (3.41) \\
\Phi'_{x}(f_{\mathrm{IM}}, t_{\mathrm{D}}) &= \psi(f_{\mathrm{IM}}) + (\Phi(f_{\mathrm{IM}}, t_{f_{\mathrm{IM}}}) - \Psi_{\mathrm{R}}(f_{\mathrm{IM}}, t_{f_{\mathrm{IM}}})) \\
&= \psi(f_{\mathrm{IM}}) + \phi(f_{\mathrm{IM}}, t_{\mathrm{D}}, P_{\mathrm{in}}) - \psi(f_{\mathrm{IM}}, t_{\mathrm{D}}) \\
&= 2\pi f_{\mathrm{IM}} t_{\mathrm{D}} + \phi(f_{\mathrm{IM}}, t_{\mathrm{D}}, P_{\mathrm{in}}) \qquad (3.42)
\end{aligned}$$

which is in the same form as the (3.13)–(3.15). The second step follows from the equivalence between time delay of t_{D} and phase change of $2\pi f t_{\mathrm{D}}$.

3.3.3 Experimental Results

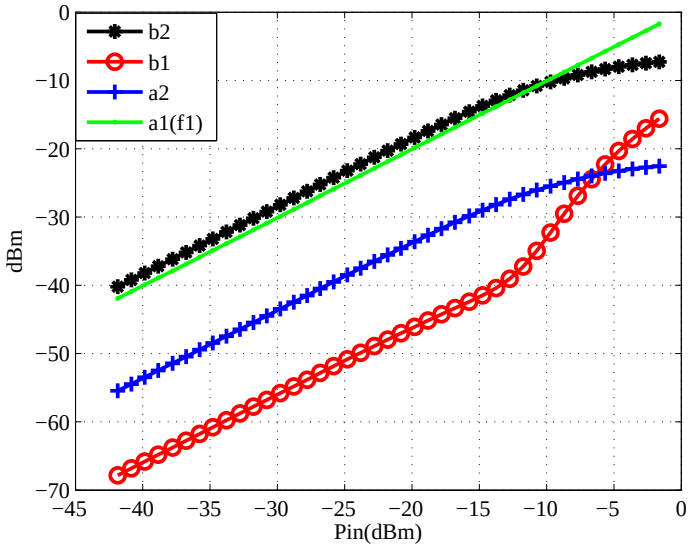
The hybrid NVNA is applied to the nonlinear characterization of a power amplifier using a two-tone signal with frequencies f_1 and f_2 centered at $(f_1 + f_2)/2$= 2 GHz with a $(f_2 - f_1)$= 200 kHz tone-spacing. The power amplifier is an Amplica SD513301 with approximately 20 dB gain between 900 MHz and 2 GHz, and 17 dBm output at the 1 dB compression point. The procedure for correcting the raw DUT measurements is illustrated here by plotting the port waves during a power sweep after each correction step. For simplicity only results at the lower fundamental frequency f_1 and lower IM3 frequency $2f_1 - f_2$ are presented.

Figs. 3.15 and 3.16 presents the amplitudes and phases of power waves following the correction described in (2.1) which includes linear correction, power correction and relative phase error correction. Fig. 3.15(a) shows that at the lower fundamental frequency f_1, the incident

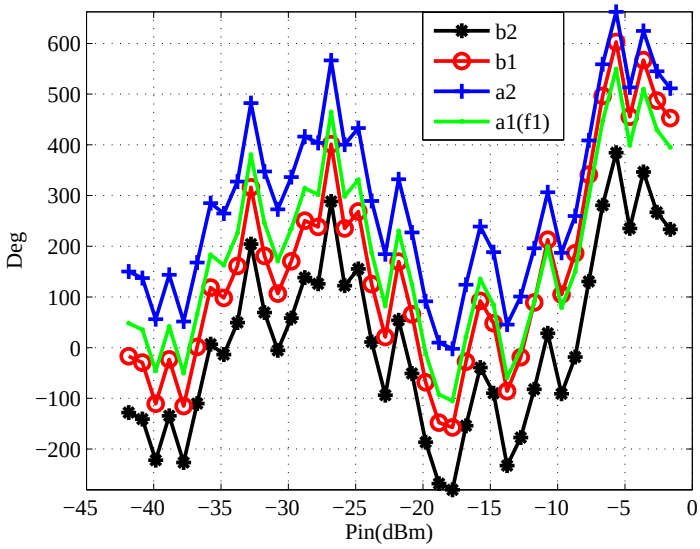
wave at port 1, $a_1(f_1)$, is linearly increased during the power sweep. Also at the same frequency, the power of the transmission wave at port 2, b_2, undergoes compression, while the reflected wave at port 1, b_1, undergoes expansion. Since b_2 is reflected by an imperfect load, there is an incident wave at port 2, a_2, which also shows compression. At the lower IM3 frequency $2f_1 - f_2$, there is no incident wave at port 1. At the same frequency, Fig. 3.16(a) shows that the power of the transmission wave b_2 undergoes compression, while the reflected wave at port 1, b_1, undergoes a null presumably due to cancellation by lower-order and higher-order contributions. Due to reflection by an imperfect load, there is also an incident wave a_2 which also shows compression. Both Figs. 3.15(b) and 3.16(b) show that the phases of the port waves track each other during the power sweep, indicating that the VNA receivers are synchronous.

Only the relative phase error of the phase transfer mechanism has been corrected so far, so additional phase correction steps are required. The remaining steps correct for the phase measurement and have no effect on the amplitude characterization. In the second correction step described by (3.28)–(3.30), the phase of the reference signal generator (which is constant during a power sweep), is subtracted from the VNA measurements corrected in the first correction step. In the final correction step described in (3.34)–(3.36), the power sweep data is shifted to the absolute phase reference provided by the oscilloscope-based measurements at a chosen excitation power. After these correction steps, Fig. 3.17 shows the phases of power waves. This result is an absolute phase measurement since the receiver phase dispersion has been removed.

At this point the alignment procedure can be applied to compare different sets of data. Even with alignment, which simply shifts each curve in Fig. 3.17(a) and Fig. 3.17(b) up or down by a constant phase value, there will still be abrupt phase jumps. The phase jumps are due to the switching attenuator in the two-tone sources, and this dependency shows up in (3.34)–(3.36) as the dependence on P_{in}. Strictly speaking, these phase jumps are part of the signal being characterized. So absolute phase measurement using the alignment procedure is best suited to signal characterization.

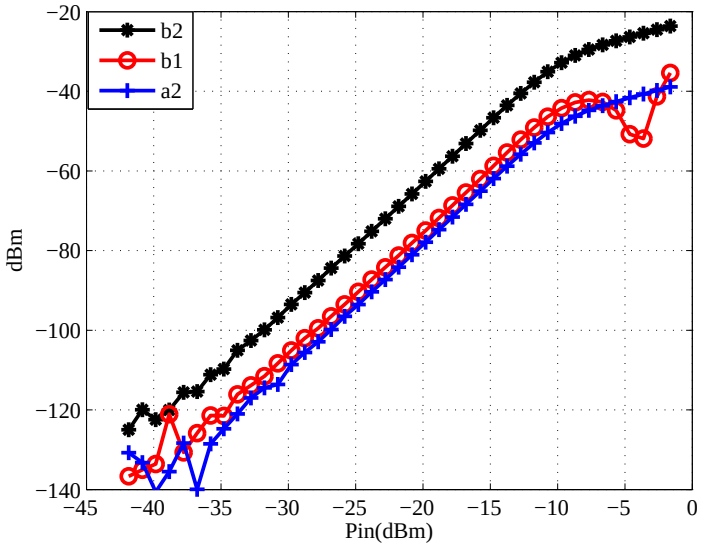


(a)

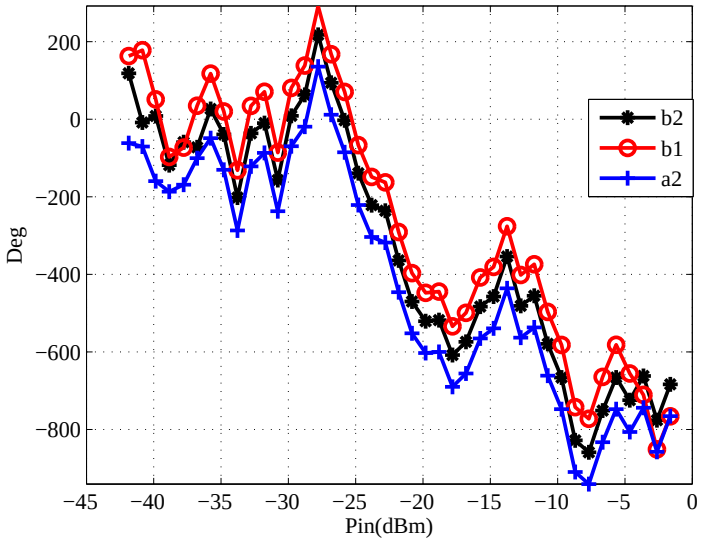


(b)

Figure 3.15: Power waves at the fundamental frequency f_1 for a two-tone test at 2 GHz with 200 kHz tone-spacing following the first phase correction described in (2.1): (a) amplitude and (b) phase.

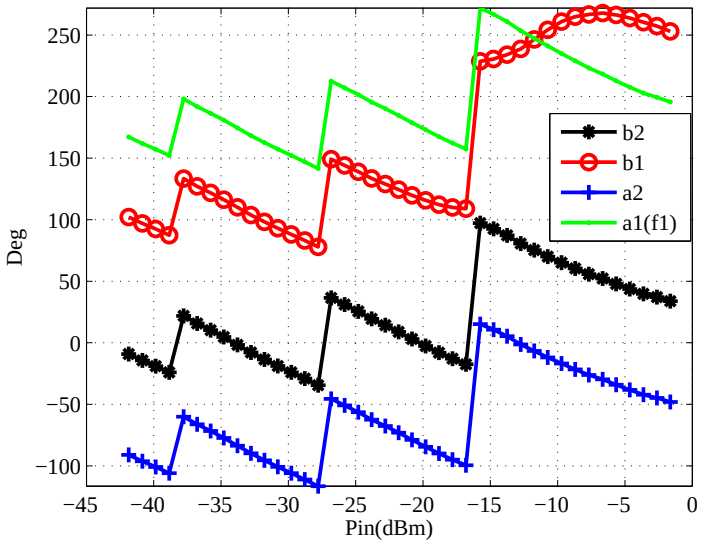


(a)

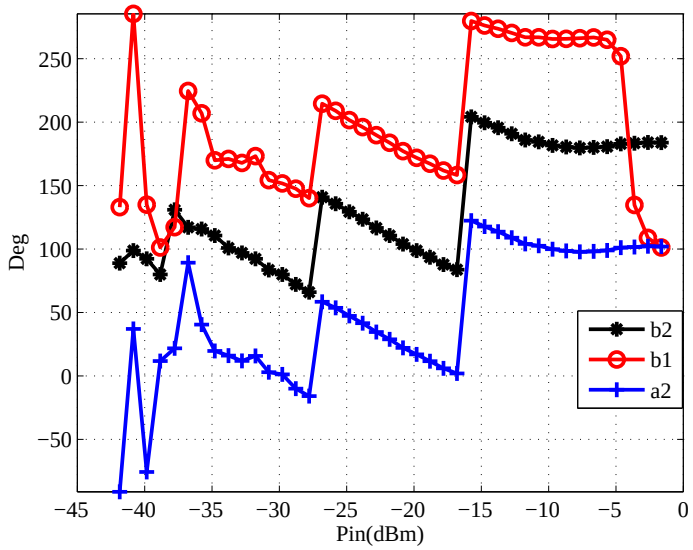


(b)

Figure 3.16: Power waves at the IM3 frequency $2f_1 - f_2$ for a two-tone test at 2 GHz with 200 kHz tone-spacing following the first phase correction described in (2.1): (a) amplitude and (b) phase.



(a)



(b)

Figure 3.17: Phases of power waves at port 1 and port 2 for the two-tone test following the final phase correction step as described in (3.34)–(3.36): (a) at the fundamental, f_1; and (b) at the IM3, $2f_1 - f_2$.

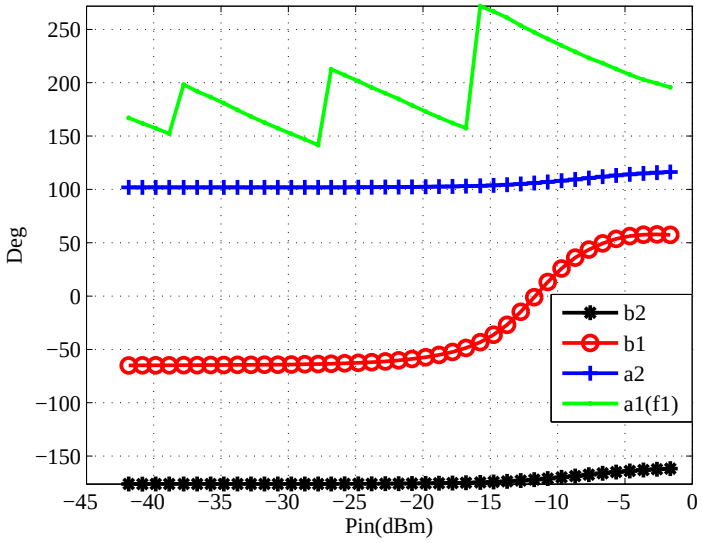
More intuitively satisfying DUT characterization results are obtained by applying the time-invariant phase definition described by (3.18). Fig. 3.18 shows that the phase contribution of the multi-tone signal generators is removed, and the scattered port waves have a slowly-varying phase change that is only contributed to by the DUT. For the lower fundamental frequency f_1 in Fig. 3.18(a), the phase of the port 1 reflected wave, b_1, undergoes expansion, which correlates with the magnitude expansion observed in Fig. 3.15(a). For the lower IM3 frequency $2f_1 - f_2$ in Fig. 3.18(b), the port 1 reflected wave b_1 undergoes abrupt phase change at high input power levels, which correlates with the IM3 sweet-spot observed in the magnitude response in Fig. 3.16(a).

Tone-Spacing Limitations

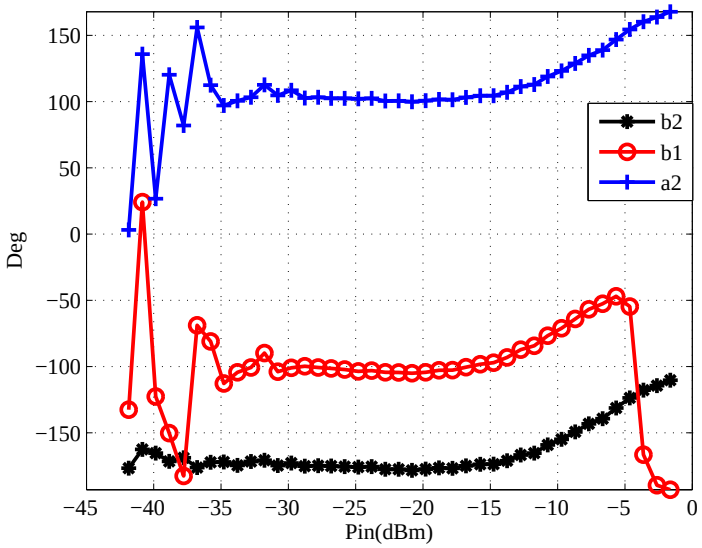
Now that the phase reference signal generator can be fine tuned in 1 Hz steps (typically), the minimum tone-spacing is established by other issues related to measuring a weak tone next to a strong tone. Firstly, the heterodyne receivers of the VNA is FFT-based. So when two tones are present in the signal, the rectangular window used in the FFTs creates spectral leakage that only decays at 6 dB per octave. Though the VNA has a minimum resolution bandwidth of 1 Hz, the weak tone needs to be 100s of hertz away from the strong tone to be measured with about 50 dB dynamic range.

Secondly, the phase noise of the multi-tone sources can degrade the accuracy the measured IM product. Using the Marconi 2024 to generate a two-tone signal with 200 Hz tone-spacing, this effect is noticeably affecting the accuracy for IM3 product 45 dBc down as shown in Fig. 3.19.

The last issue is that the memory depth of the oscilloscope may be too small for the desired frequency resolution. For an RF frequency of 1 GHz and tone-spacing of 1 kHz and Nyquist sampling up to the first harmonic band, two mega-samples are needed. This is readily available.

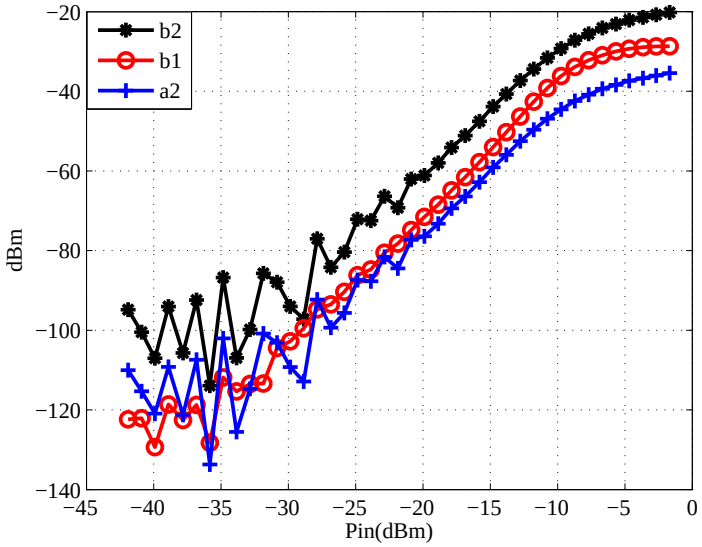


(a)

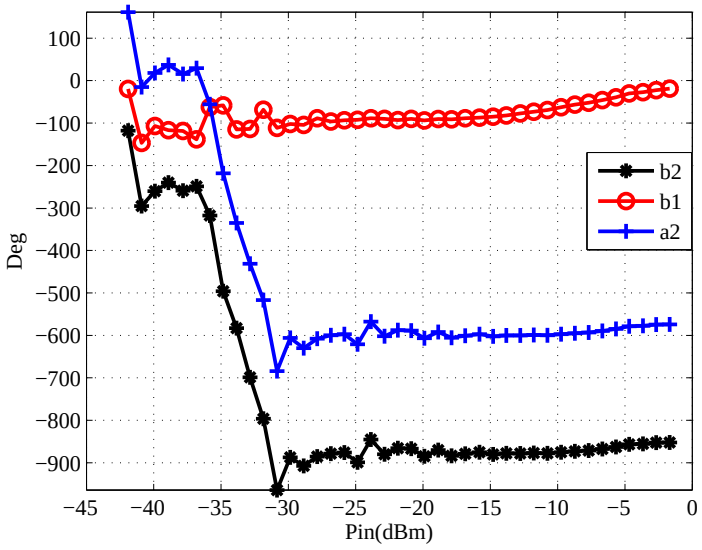


(b)

Figure 3.18: Phases of power waves for the two-tone test after applying the time-invariant phase definition (3.18): (a) at the fundamental, f_1; and (b) at the IM3, $2f_1 - f_2$.



(a)



(b)

Figure 3.19: Power waves at the IM3 frequency $2f_1 - f_2$ for a two-tone test at 2 GHz with 200 Hz tone-spacing after applying the time-invariant phase definition (3.18): (a) amplitude and (b) phase.

3.3.4 Dynamic Range

The dynamic range of the hybrid NVNA can determined either from the amplitude or the phase response. In the following discussion, the transmission wave at port 2, b_2, is considered because it has the highest IM ratio (in dBc) and is the most difficult to measure. For measurements with tone-spacing set at 200 kHz, Fig. 3.16(a) is the amplitude response at the lower IM3 frequency as the two-tone signal ramps from low power to high power with the power of each tone being equal. At low two-tone powers it is expected that the response level (in dBm) will vary linearly with changes in the two-tone power. Thus a departure from linearity indicates the minimum detectable signal, which here is -122 dBm with corresponding P_{in} of -40 dBm. At the same P_{in} in Fig. 3.15(a), the lower fundamental is -39 dBm. This indicates that the dynamic range is 83 dB. Similarly, for a two-tone signal with tone-spacing set at 200 Hz, Fig. 3.19(a) shows that the minimum detectable signal is -62 dBm with corresponding P_{in} of -21 dBm. At the same P_{in}, the lower fundamental is -20 dBm, which indicates that the dynamic range is 42 dB.

The phase response is more sensitive to noise, and more accurate in determining the minimum detectable signal. At low two-tone powers, it is expected that the phase of the response will stay constant with changes in the two-tone power, thus a departure from a constant phase indicates the minimum detectable signal. For a two-tone signal with tone-spacing set at 200 kHz, Fig. 3.18(b) shows that the minimum detectable signal corresponds to a P_{in} of -37 dBm. From the corresponding power level in Fig. 3.15(a) for the lower fundamental and Fig. 3.16(a) for the lower IM3 which are -116 dBm and -35 dBm respectively, the dynamic range is computed to be 79 dB. For a two-tone signal with tone-spacing set at 200 Hz, Fig. 3.19(b) shows that the minimum detectable signal corresponds to a P_{in} of -20 dBm. Similarly, from the corresponding power level for the lower fundamental and the lower IM3, the dynamic range is computed to be 40 dB.

The usable dynamic range of the hybrid NVNA is about 40 dB at 200 Hz tone-spacing, and this monotonically increases to about 80 dB at 200 kHz tone-spacing. While this is an improvement over the previously reported 4-port VNA-based NVNA, the dynamic range is still

well below what is required to measure heating effects in planar transmission lines [78]. What can be characterized are the long-term memory effects in RF amplifier circuits, where third-order intermodulation products of concern are typically at the −40 dBc level. The fifth- and seventh- order contributions to the intermodulation products are typically present at about the −20 dBc level and can also be measured. At narrow tone spacings, the limitation in dynamic range is due in large part to the phase noise in the phase-locked sources that are combined as the multi-tone excitation. In the feedforward cancellation system presented in [78], along with the discrete tone, the phase noise can also be cancelled since the phase noise of one source is phase synchronous with the phase noise of the other. In the hybrid NVNA the discrete tones are measured rather being cancelled, and the SNR is degraded by the phase noise.

3.3.5 Measurement Time

In measurements at a particular tone spacing using the hybrid NVNA, the VNA portion acquires data at all spectral components with swept excitation power levels, and the oscilloscope portion acquires data at all spectral components at one power level. Compared to the previously reported 4-port VNA-based NVNA, the hybrid NVNA requires additional data acquisition from its oscilloscope portion. If the hybrid NVNA uses an outdated sampling oscilloscope, the duration of each oscilloscope acquisition is dominated by the data transfer time over the GPIB interface. At each tone spacing, the acquisition duration of the oscilloscope portion is comparable to the acquisition duration of the VNA portion. In this case, the hybrid NVNA doubles the measurement time when compared to that of the 4-port VNA-based NVNA. If the hybrid NVNA uses a sampling oscilloscope with a faster data transfer interface, then the duration of each oscilloscope acquisition is negligible when compared to the duration of the VNA acquisition. In this case, the hybrid NVNA takes approximately the same amount of measurement time as the 4-port VNA-based NVNA.

In the sub-sampling mixer-based NVNA measurement at a particular tone spacing, all spectral components at each power level are captured at once. At each power level, the total

data acquisition time is approximately the same as the amount of time it takes the VNA to capture one spectral component. Thus the hybrid NVNA measurement time exceeds the sub-sampling mixer-based NVNA measurement time by a factor equaling the number of spectral components being measured.

3.3.6 Summary

This Section presented a hybrid NVNA system used with common bench-top instruments: phase-locked signal generators, 4-port VNA, and an equivalent-time sampling oscilloscope. The multi-harmonic generator is not required, and the associated tone-spacing limitation in multi-tone testing is removed, by using a reference signal generator phase-locked to the multi-tone excitation which can tune to tone-spacings on the order of 1 Hz. The high dynamic range heterodyne receiver-based VNA is used for data acquisition during the power sweep, while the low dynamic range sampling oscilloscope is used for its absolute phase calibration ability.

Using each instrument to its advantage, narrow tone-spacing, wide bandwidth, and high dynamic range is achieved. The narrowest tone-spacing usable is now determined by the memory depth of the oscilloscope.

3.4 Conclusion

This chapter present two novel NVNA setups that do not need a multi-harmonic generator to establish a phase reference, and that can be implemented using common bench top instruments.

To use a 1-channel receiver to characterize a multi-port DUT, the DUT signals is sequentially sampled to mimic a synchronous multi-channel receiver. To maintain the same timing relation between sequentially sampled waveforms as if they were synchronously sampled, excitation waveforms are designed to be commensurate with the AWG and the VSA sampler time base. Once the 1-channel VSA can be made to behave like a multi-channel receiver, concepts from absolute phase measurement using broadband receivers can be applied with minor modifications. Namely, a multi-tone signal generated by an AWG replaces the multi-harmonic generator, and

a nose-to-nose calibrated oscilloscope provides the reference measurements for VSA phase error correction.

The 1-channel VSA-based NVNA can be viewed as a narrowband version of the previous broadband receiver based NVNAs [17], and suitable for characterizing DUT's in-band behavior for a narrowband multi-tone excitation. However the dynamic range is improved, due to the use of a narrowband tuned receiver instead of a sub-sampling mixer-based front-end where noise at all sidebands of the sampling frequency harmonics are downconverted to the baseband. The proposed 1-channel receiver implementation only requires an AWG and a sampling oscilloscpe, can be easily adapted to other hardware that are more readily available to the practicing engineer It has potential for future self-tuning and self-healing RF front-end architectures that requires an accurate and low overhead on-chip self-characterization method.

The proposed 4-port VNA-based NVNA is used together with common benchtop instruments: phase-locked signal generators, and an equivalent-time sampling oscilloscope. The multi-harmonic generator is not required, and the associated tone-spacing limitation in multi-tone testing is removed, by using a reference signal generator phase-locked to the multi-tone excitation which can tune to tone-spacings on the order of 1 Hz. Also the concept of using a relative phase reference enables optimum use of different instruments for different parts of the measurement. The high dynamic range heterodyne receiver-based VNA is used for data acquisition during a power sweep, while the low dynamic range sampling oscilloscope is used for its absolute phase calibration ability. Using each instrument to its advantage, narrow tone-spacing, wide bandwidth, and high dynamic range is achieved. The narrowest tone-spacing usable is now determined by the memory depth of the oscilloscope, and the phase-noise characteristic of the phase-locked signal generators.

Chapter 4

Grey-Box Model Identification

4.1 Introduction

For circuit simulation applications, past works on RF model identification have focused on the Wiener and Hammerstein model family [55], [70], [71], [72], [73], [74], [75]. In all cases the model extraction procedures were developed specific to the model being developed and rely on deterministic optimization methods.

An important question is how other useful models can be discovered and identified, especially for multi-stage nonlinear circuits? It is essential that a parameter extraction technique for these more sophisticated models be developed without a large amount of engineering effort being expended in customizing parameter extraction as has been done for the Wiener-Hammerstein family of models.

Current research explores a general identification methodology for more complex nonlinear models than have been used previously. The type of problem for which this work is targeted is extracting the grey-box models of the RF front-end of undocumented devices for co-site interference simulation, for example, where two or more radios are operated in close proximity and performance is compromised by broadband noise and the unintended active scattering of intermodulation (IM) components, both effects interfering with radio reception.

4.1.1 Stochastic Optimization and Multiple Optimum Solutions

The model identification procedures developed in past works are not suitable due to the ad-hoc nature of the identification algorithm and the use of deterministic optimization. Deterministic local search methods based on gradient or Nelder Mead simplex methods depend on the starting point of the search, which may suffice for simple models where initial solution close to the global optimum can be found. For more complex models, a good initial solution is much harder to find. In addition, deterministic optimizers are prone to getting trapped in local optimum solutions due to their dependence on the starting point. That is, modifying the initial parameter estimates can often result in a different, and possibly preferable, solution being found. As a result, the modeling exercise is not straight-forward and all too often must proceed either by choosing alternative starting points for the model fitting algorithm, or by modifying the error cost function.

Any new model identification procedures developed must have the ability to explore the search space. Stochastic optimizers have additional randomization mechanisms built in at each optimization iteration, so as to avoid local optimums. So unless the same random seed number is used, a stochastic optimizer will always yield a different solution each time it is run.

With search space exploration ability, the global solution having the minimum error between the modeled and measured responses can be found. However the global solution may not be a physically or intuitively satisfying solution to a skilled and knowledgeable engineer with experience acquired with the type of system being modeled. One reason is that in most RF systems, the frequency shaping by bandpass filters and matching networks prevent direct observation of nonlinear effects. Also there are rarely enough external measurements available to unambiguously identify all of the model parameters. As a result of noise in the measured response, indirect observables, and model structure inadequacies, multiple optimum solutions with similar error metrics should be tracked for selection by the user.

4.1.2 Proposed Identification Methodology

Section 4.2 generalizes the identification methodology to more complex nonlinear models than have been used previously. Section 4.2.1 presents a modified genetic algorithm (GA) procedure that tracks sets (or families) of possible solutions. For each family, a derivative-free gradient descent algorithm, the Nelder-Mead simplex algorithm, is used to present a skilled operator with a number of distinct modeling solutions with comparable error metrics. In Section 4.2.2 the ad-hoc higher-level identification algorithms presented in previous studies are generalized and adapted to stochastic optimization methods. In the result hierarchical identification algorithm, the focus is on the treatment of families of distinct solutions between iteration steps.

For more generalized model fitting and elimination of user tasks that can be automated, general purpose circuit simulation is used to relate the model parameters to the measured data which typically consists of the amplitude and phase of discrete tone responses to swept-power and swept-frequency one-tone and multi-tone excitations. Here the previously typical hand-crafted Volterra NLTF analysis specific to the particular model being developed, is replaced by a symbolic Volterra circuit simulator to automate the model analysis and response calculation. Chapter 5 presents the implementation of a direct calculation method for Volterra circuit analysis which enables compression handling lacking in an earlier implementation [6].

The main result is a reorganization of the way models are extracted for RF devices. Instead of a skilled user being involved throughout the model fitting process, user input is directed first at establishing several candidate grey-box models, establishing the measurement protocol, and then choosing the best solution at the end of the procedure.

Section 4.2.3 discusses the IM response data used in the identification algorithm. Section 4.3 applies the model extraction procedure to a multi-stage solid state power amplifier exhibiting long-term memory effects.

4.1.3 Uniqueness of Model Parameters and Structures

Before proceeding to the proposed solution for handling more sophisticated models, it is necessary to consider what can be identified. For linear systems, structural identifiability assesses the uniqueness of the parameters for a given model from input-output experiments, and structural indistinguishability assesses the uniqueness of the model structures with the same input-output behavior but possibly different parameter values. Such questions are answered through the analysis of the transfer function using algebraic techniques, and analysis of the block diagram using geometric techniques [79].

For nonlinear single-input single-output (SISO) systems containing one memoryless nonlinearity, such uniqueness questions were studied in [80]. Systems belonging to this family include the Wiener, Hammerstein and Wiener-Hammerstein models, with possibly additional linear or nonlinear blocks placed in a feedback configuration as illustrated in Fig. 4.1. Using the NLTF representation of the nonlinear system, it was proved that the parameters for such a system can only be uniquely identified modulo a loop transformation, scaling, or delay constant [80]. That is, different sets of parameters related by scaling, delay, or loop transformation result in the same NLTF.

Similar conclusions were reached in [73] for nonlinear single-input multiple-output (SIMO), multi-input single-output (MISO), and multi-input multi-output (MIMO) systems constructed from branches of Wiener or Hammerstein models. By solving for the first few slices of the NLTF for the constraining kernel relation, it was shown that for models that are parameter unique, the model parameters can only be determined within arbitrary constants.

Intuitively, the uniqueness of a set of model parameters can be assumed from the block diagram of the model. A test of uniqueness is to choose blocks with the same parametrization and insert two scaling multipliers at the input and output. Then proceed to the other connected blocks and insert matching inverse scaling multipliers. This process is continued until all blocks in the model have scaling multipliers at the input and output. If all the matched scaling

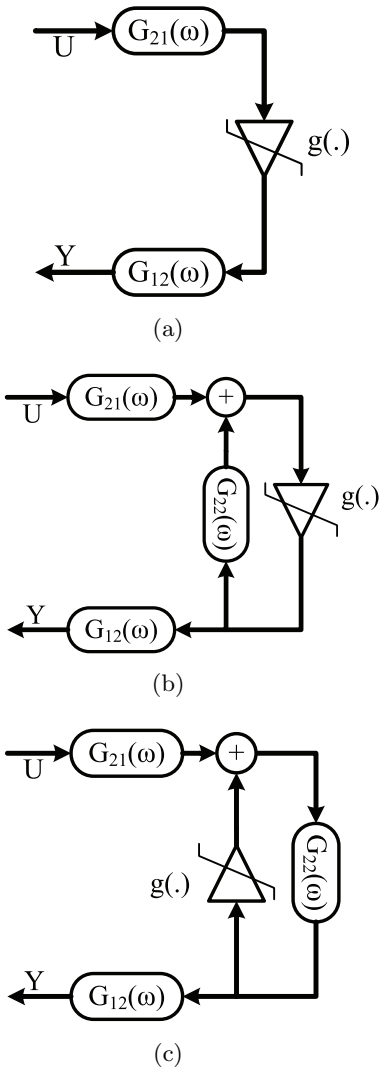


Figure 4.1: Models containing one memoryless nonlinearity can be separated into disjoint parts: (a) basic Wiener-Hammerstein model; (b) Wiener-Hammerstein model with a linear feedback network; and (c) Wiener-Hammerstein model with a nonlinear feedback network.

multipliers are consistent, that is the scaling multipliers at a block's input and the block's output are inverses of each other, then the model parameters are unique modulo scaling. This is illustrated in Fig. 4.2. If an inconsistency arises, then constraining conditions exist on the scaling between blocks with different parametrization although there are not enough conditions to uniquely determine the parameters. If there is inconsistency at the input and output of every block, then enough conditions exist to uniquely determine the parameters.

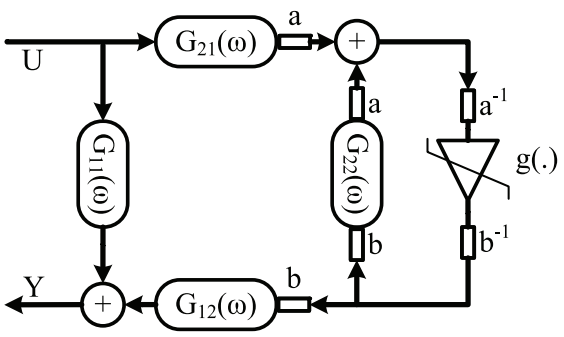


Figure 4.2: SISO model containing 1 memoryless nonlinearity.

4.2 Grey-Box Model Identification Methodology with Minimal User Intervention

At the highest level, the model identification procedure developed in this paper is shown in Fig. 4.3. The measured data consists of the amplitude and phase of discrete tone responses to one-tone and multi-tone excitations swept in power and frequency. In grey-box modeling, parameters are not directly related to the quantities measured and so general purpose nonlinear simulation is required to relate the parameters to the measured discrete tone responses.

We contend that there are many situations where it is preferable to find multiple local optimum solutions. One situation is when the model does not allow for the unique identification

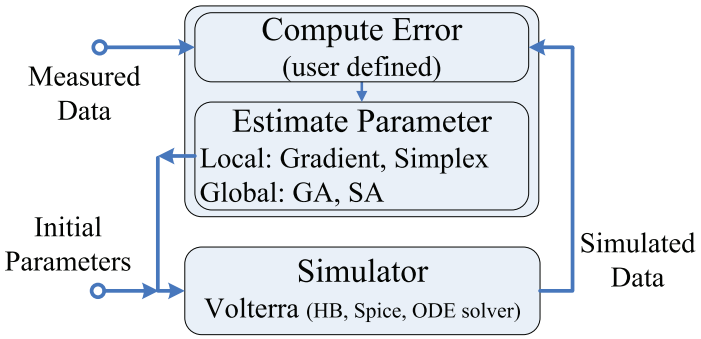


Figure 4.3: Proposed general approach for the extraction of nonlinear equivalent circuit models in the frequency domain.

of its parameters. Other situations include limited fitting capability of the model chosen, limited and possibly noisy measurement data available, and parameters that can only be indirectly observed.

A key component of the proposed procedure is the use of stochastic optimization to track multiple distinct solutions. In part this addresses the difficulty of finding a good initial parameter estimate as required in deterministic optimization. In addition, the randomization mechanisms built in at each iteration can help the optimizer avoid being trapped in local optima. So, unless the same random seed number is used, a stochastic optimizer will always yield a different solution each time it is run.

The various dialects of the evolutionary algorithm (EA) are based on different heuristics. Some EA methods such as the particle swarm (POS) algorithm, ant-colony algorithm, etc., embody more specialized heuristics and work particularly well for particular problems. The genetic algorithm (GA) is the most versatile EA and uses the least specialized heuristics.

As discussed in Section 4.2.1, the GA algorithm was augmented by us to track multiple families of solutions and Nelder-Mead simplex optimization is regularly performed for each family to find local optimums. In effect, the randomization of initial parameter estimates required by deterministic optimization is replaced by a hybrid technique that combines the stochastic (intelligent randomized parameter space exploration of) GA and Nelder-Mead methods. Com-

pared to gradient-based methods, the simplex method can handle complex solution contours, and has lower computation cost per iteration. During hybrid GA optimization, function calls were made to the standard GA and simplex search implementation using Matlab's nonlinear optimization toolbox. Section 4.2.2 discusses the hybrid model identification algorithm including the tracking of populations of possible solutions. Section 4.2.3 discusses the design of the optimum measurement regime that produce the IM response data used in the identification algorithm.

The most important factor affecting the complexity of the parameter estimation problem is the structure of the grey-box model. A suitable strategy is parameterizing certain blocks first as linear then as nonlinear, and then gradually introducing structural detail such as decomposing one nonlinear block into a interconnection of linear and nonlinear blocks. Such a gradual increase of complexity at the model structure level and model parametrization level — a task for the experienced engineer — is what is often needed to bootstrap the identification process.

4.2.1 Hybrid Genetic Algorithm Optimizer

The standard GA implementation maintains a population representing that part of the search space being explored, and eliminates individuals around local optimums, thus promoting only the sub population around the global optimum. The parallel GA approach presented in [81], maintains multiple separately evolving and occasionally interacting sub populations, which is well suited to parallel computing. Another approach that aims to find local optimums is the niching GA approach [82] which keeps individuals within a niche-radius of local optimums, thus promoting multiple sub populations or niches around each local optimum. Augmenting the GA search, a hybrid approach has been devised which applies local search to the candidates in the terminal generation at the conclusion of the GA search. In this way GA, which dominates the number of goal function evaluations, can be ended prematurely. The major problem is determining when to terminate the GA search and launch deterministic optimization.

The new formulation of hybrid optimization for identifying multiple local optimum solutions

is outlined in Fig. 4.4. The concept here is modifying standard GA so that instead of eliminating possible individuals around local minimums, they are retained and used as the starting points for local searches. During the earlier generations of a standard GA search, it is hypothesized that individuals around the multiple local optimums exist. Thus local search is applied to a portion of the population to find each local optimum. These local optimums are recorded and their ranking continuously updated as the process is repeated in successive GA generations. (The architecture of the optimizer is targeted for implementation with hundreds or more of processor cores so that GA can continue independently of parallel deterministic searches.)

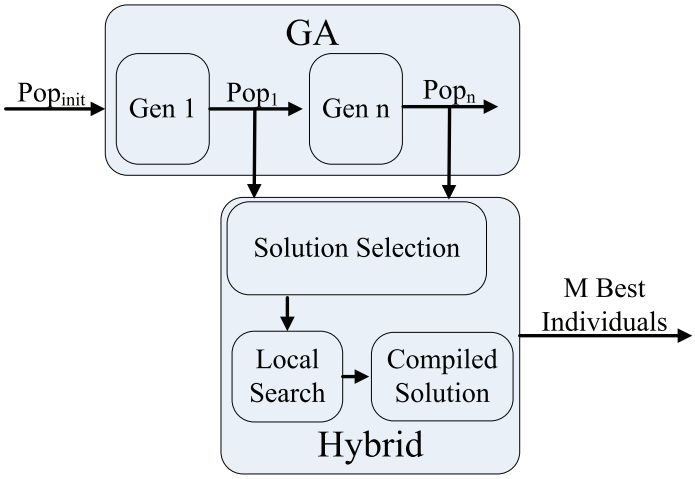


Figure 4.4: Proposed hybrid optimization method. The settings for GA are the number of generations, the population size, and the selection of individuals used in local search. The settings for local search are the relative error tolerance defined as the error difference between the iterations relative to the lowest error achieved (4.3), and the maximum number of consecutive iterations such that the result is below the relative error tolerance. In this way, the absolute error tolerance need not be set, but will be changing adaptively.

An alternative is to retain a record of all past searches to guide and eliminate redundant searches, as was done in Bilbro et al. [83], [84] using tree annealing, and Yuen and Chow [85] using GA search. In [84] a hybrid tree annealing search for multiple optimums, local search is applied after a fixed number of stochastic iterations. Which hybrid optimization method, tree

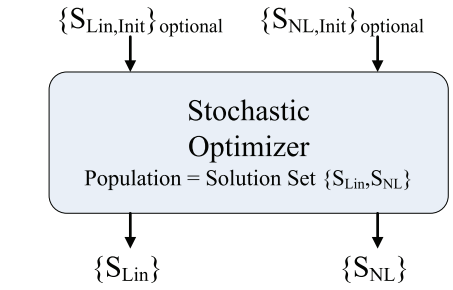
annealing or the hybrid GA approach presented here, is more efficient depends on the number of generations and the population size required for convergence. Empirically it has been found that the required population size is proportional to the number of model parameters for adequate sampling of the solution space. For a large number of parameters (48 parameters are considered later), the hybrid GA approach is found to be greatly superior.

4.2.2 Identification Algorithm

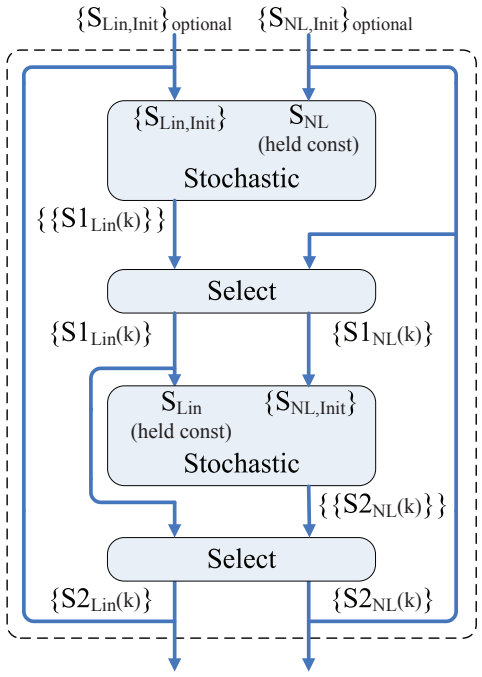
In this Section the identification algorithm using the new hybrid optimizer is presented. While the strategy of GA search with periodic deterministic search is simple, this is a more computationally intensive process than necessary. As shown in 4.5(a), while GA can start from a population of random initial solutions, the population size needs to be large for good sampling of the solution space. Microwave and RF circuits generally have a linear small signal response that departs from linearity as applied signals become larger. Also they tend to have rich frequency-dependent responses, but an intrinsic linearity that is largely frequency independent. Since the linear parameters constitute the majority of the model parameters, this behavior is exploited to reduce problem complexity in the partitioned approach shown in Fig. 4.5(b). Alternatively, the linear blocks could be parameterized as rational transfer functions, but extracting a passive and causal rational transfer function is a complex problem in itself [86].

Starting from the $(k+1)$th iteration step that estimates the linear part, the nonlinear parameters take on values $S2_{\mathrm{NL}}(k)$ from its population at the kth iteration step. For each individual in $\{S2_{\mathrm{NL}}(k)\}$, the optimizer outputs a population of linear parameters $\{S1_{\mathrm{Lin}}(k+1)\}$, which can be collated into a set of populations $\{\{S1_{\mathrm{Lin}}(k+1)\}\}$. Selection is then made from this superset with respect to the fitness value, with the corresponding nonlinear parameter taken from $\{S2_{\mathrm{NL}}(k)\}$, to arrive at condensed $\{S1_{\mathrm{Lin}}(k+1)\}$ and $\{S1_{\mathrm{NL}}(k+1)\}$ estimates. The identification algorithm then continues with the role of the linear and nonlinear parameters reversed.

For general RF model structures a typical expression used to estimate the linear response



(a)



(b)

Figure 4.5: Higher level identification algorithm: (a) estimating all parameters simultaneously; and (b) partitioned approach alternately optimizing linear and nonlinear parameters in separate iteration steps. The settings of the identification algorithm are the relative error tolerance defined as the error difference between the iterations relative to the lowest error achieved (4.3), the maximum number of consecutive iterations such that the result is below the relative error tolerance, and the selecting of individuals from each population between the iteration steps. In this way, the absolute error tolerance need not be set, but will be changing adaptively.

is a sum of products of linear transfer functions, that is,

$$\mathrm{IM3}_L(f_{\mathrm{im3}_L}) = \tfrac{1}{4}Z(f_{\mathrm{im3}_L})Z^2(f_1)Z^*(f_2)(g_2^2\,[4Z^*(\mathrm{f_B}) + 2Z(f_{\mathrm{hd2}_L})] - 3g_3) \tag{4.1}$$

with each product scaled by a product of the nonlinear parameters. Such nonlinear dependence on the nonlinear parameters means that each iteration step depends on the results of previous iterations. The first consequence is that a good initial solution to the linear parameters cannot be found without considering the result from the second iteration step for the nonlinear parameters. The second consequence is that the identification algorithm must go through more than one loop.

The computation cost may still be high considering that the stochastic search may be run as many times as the population size of the nonlinear parameters. For a narrow-band communication circuit, a less computationally intensive variation of the approach above is shown in Fig. 4.6. Here, the input signal bandwidth is a fraction of the carrier frequency, so the output IM products are in groups narrowly spread in frequency centered around carrier harmonics. For a first-pass estimate the linear parameters around the carrier harmonics can be considered to be frequency independent. This dramatically reduces the number of linear parameters. This can be followed by a more detailed estimate of the linear parameters in certain harmonic bands. To reduce the amount of computation, only a select number of individuals from the first-pass estimate would normally be considered.

4.2.3 Experiment Design

For the iteration step that estimates the linear parameters, it is important to consider the response data used. Generalizing from [73], [74], [75], [67], the expressions chosen are the lowest-order NLTF or the lowest-order contributions to the measured IM response in which the linear parameters appear. Such low-order expressions are the least complex, with minimal dependence on higher-order nonlinear parameters, yet still dominant enough to be measured

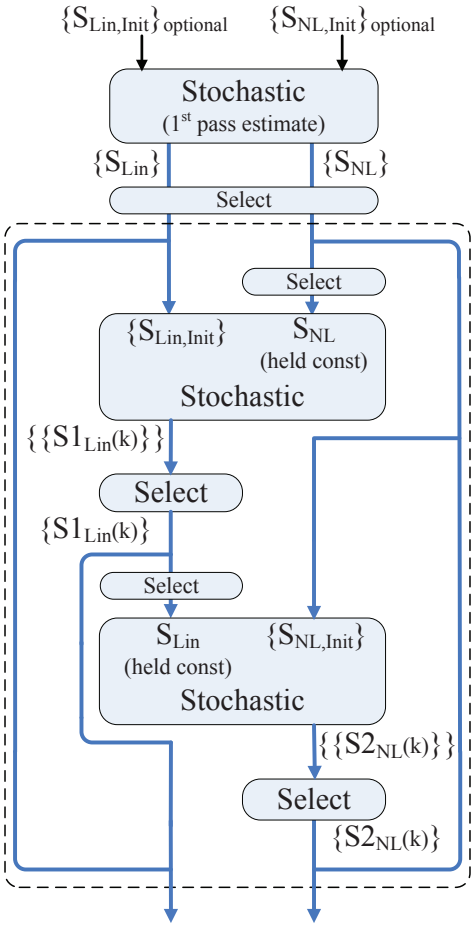


Figure 4.6: For narrow-band circuits, a less computation intensive variation of the identification algorithm. Initially, assuming constant linear parameters in the carrier harmonic bands, a 1st-pass estimate results from stochastic optimizer. Then, the linear and the nonlinear parameters are alternately estimated in separate iteration steps.

with good signal-to-noise ratio. As an example consider two-tone excitation, the linear parameters at a fundamental band first appear in the first-order response at frequencies f_1 and f_2. In [74] this consideration lead to initializing parameters for the two linear blocks in the Weiner-Hammerstein model using the square root of the linear frequency response. The linear parameters at the envelope and the second harmonic band first appear in the second-order response at frequency $(f_2 - f_1)$ and at frequencies $2f_1$ and $2f_2$ respectively. Depending on the measurement equipment and the structure of the device under test, it may be difficult to measure the second-order response directly. For example, in narrow band systems neither the baseband nor the second harmonic are available at the output ports. Then as far as the model fitting problem is concerned, for certain models the linear parameters at the envelope and the second harmonic frequencies are observed indirectly through their impact on the third-order adjacent channel IM responses at $(2f_1 - f_2)$ and $(2f_2 - f_1)$. In [67] this consideration lead to using the third-order adjacent channel IM response for the extraction of the transfer functions of the feedback path at the envelope and second harmonic bands.

Similarly, it is important to consider the response data used in the iteration step that identifies the nonlinear parameters. The frequency separation technique [87] can improve the accuracy of the estimate in this iteration loop. For a power series model this means that the degree of the coefficient to be extracted should equal the order of the IM response. For example, if the coefficient of the fifth power is to be determined then using measurements of the fifth harmonic at the lowest input powers that produce a measurable response should be used. Thus for a narrow band communication system a measurement of the fifth-order IM frequency should be used. Then the estimate of the lowest degree coefficient will be the most accurate. This is the reason why the AM-PM model cannot accurately predict third-order responses, even for a memoryless circuit. Similarly, for a two-tone excitation, a model fitted using only the response at tone frequencies f_1 and f_2 cannot predict the first adjacent channel IM response. For broadband multi-tone excitation the same consideration led to the use of sparse odd-multisine excitation [74], which results in an uncluttered response spectrum for second- and third-order

IM products.

In theory the input power level of the IM response data used to estimate the nonlinear parameters can be determined from the slope of the chosen IM response versus input power, as done in [67]. In practice, this becomes increasingly difficult at high input power levels due to higher order contributions from higher degree coefficients, and possible cancelation effects. Thus identification of the nonlinear parameters requires measured IM responses over swept input power. Choosing the IM response with the same order as the coefficient degree provides constraints that ensure the accuracy of the nonlinear parameter estimate.

An important consequence of the input power range chosen is that it determines the maximum contribution order calculated by the Volterra circuit simulator. To estimate this maximum order for a multi-tone signal, we perform a power series fit to the one-tone AM-AM and AM-PM responses. The maximum one-tone output power fitted equals the multi-tone output power summed, assuming incoherence of the tones. Several orders of the power series are tried. The lowest order that shows unbiased residue is the maximum contribution order to be calculated.

4.3 Case-Study: PA with Long-Term Memory

In this Section the new parameter identification algorithm is applied to the identification of a narrow-band multi-stage power amplifier (PA) with appreciable long-term memory effects. The identification process is: (1) identifying the model architecture that captures important physical effects; (2) approximate partitioning of the parameters into linear and nonlinear parameters; (3) designing an experimental protocol to facilitate extraction.

4.3.1 Model Structure Simplification and Parametrization

The structure of the multi-stage PA in Fig. 4.7 is transformed into a transfer-function oriented model shown in Fig. 4.8. The final model can be put into an equivalent form similar to Fig. 4.2, but now $G_{12}(\omega)$ is nonlinear.

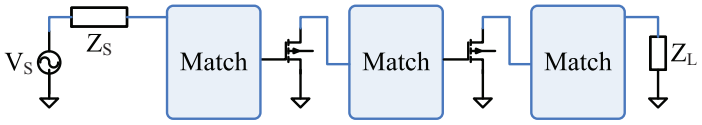


Figure 4.7: Multi-stage PA with 2-stages shown.

As a simplification of the structure, only the last stage is treated as being nonlinear, which is reasonable considering that the signal level is the largest at the last stage. The optimum design of a multi-stage amplifier assigns equal dynamic range to each stage [88], with the early stages dominated by noise performance, and the last stage dominated by distortion. Thus the gain decreases going from the input stage to output stage. Quantitatively, if each stage in a two-stage amplifier is modeled as a transconductance power series with coefficients a_n and b_n respectively, then each stage's third-order adjacent channel output IM products under two-tone excitation are proportional to $b_1 a_3$ and $b_3 a_1^3$ respectively. In a good design, the bias and device sizing are such that $b_3 \approx a_3$, so Stage 2 dominates the IM response as $b_1 \ll a_1^3$. It is also assumed in model topology that the last stage does not have a significant feedback path or if it is significant, that the miller approximation enables the unilateral device model to be used.

If possible the parameter set should also be simplified. The PA being modeled uses BJT transistors, so that the output impedance is much more linear than for a FET amplifier stage. Thus the output impedance of each stage is parameterized as being linear. Note that the input impedance is still nonlinear. A further simplification is that the linear part of the input and output networks are frequency independent; and that the model is required to capture transmission characteristics only. Then the frequency-dependent linear behavior is captured by $Z(\omega)$ and so the frequency independent elements can be lumped together as a constant gain a for the input network, and absorbed by the nonlinear gain b for the output network. For narrow band circuits the measured linear response $aZ(\omega)b_1$ is relatively flat in-band. So when the IM products fed back are shaped by $Z(\omega)$, the assumption does not severely affect model accuracy.

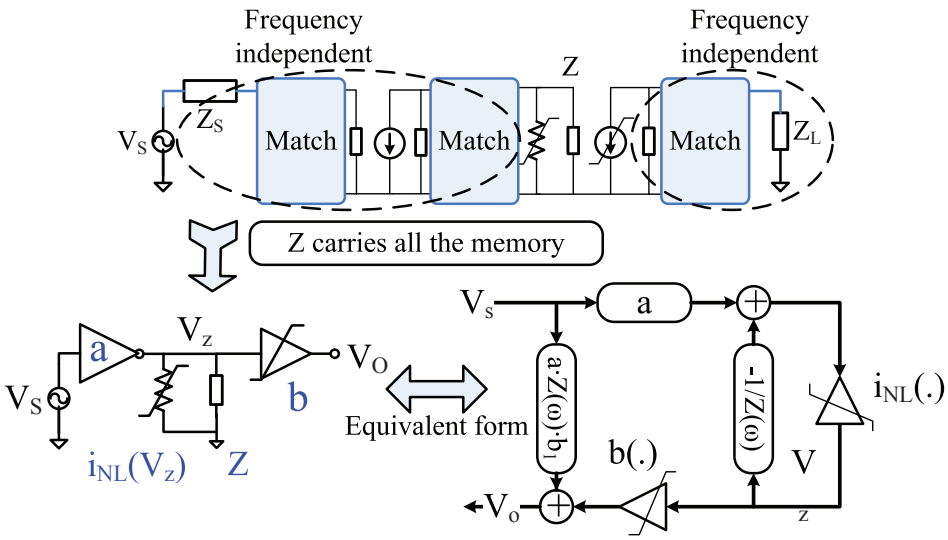


Figure 4.8: Model after simplification of the structure and parameter set. Structural simplification assumptions: last stage nonlinear, no dominant feedback or any linear feedback can be approximately transformed so unilateral device model can be used. Parameter simplification assumptions: active device is a BJT, only transmission measurements will be modeled, all linear memory is now carried by $Z(w)$.

4.3.2 Identification Algorithm and Experiment Design

Input test waveforms must be designed to extract responses to important physical effects and to enhance model identification. Here to excite the long-term memory effect, the test signal must have finite bandwidth and be compatible with Volterra circuit analysis. While a continuously modulated signal is compatible with a Volterra NLTF model, model output evaluation with discrete tones is rapid, which is particularly important as the number of evaluations required in the identification can be very large. Thus while actual signals can only be approximately modeled as discrete tones, it is preferable, almost necessary, to use discrete tone testing for identification. The two-tone excitation is the simplest multi-tone signal that can be used. Since it is narrowband, the identification algorithm in Fig. 4.6 is applied.

The analysis order of the Volterra circuit simulator is determined by the power range of the IM response to be modeled. As discussed in Section 4.2.3, to model up to 1-dB compression of the two-tone fundamental output which is about 8 dBm, the power series is fitted up to a

one-tone output power of 11 dBm for a variety of orders. The lowest order that showed unbiased residue was the fifth.

With the carrier at 2 GHz, the tone spacing was swept from 1 kHz to 7 MHz in 12 logarithmically spaced steps to capture the essential features of the tone spacing dependent memory effect. The output power was swept from -2 dBm to 8 dBm in 1 dB steps, with the maximum output power corresponding to a first adjacent channel IM relative distortion of -25 dBc. The PA was characterized using the one-channel VSA technique [89], which measures the amplitude and phase of the IM response at the carrier band.

The response data used in the iteration step that estimates the linear and nonlinear parameters follows from the discussion in Section 4.2.3. The second-order response at $(f_2 - f_1)$ cannot be used due to the narrow band matching networks. Instead, the parameters that describe the linear in-band memory and envelope responses are estimated from the first-order responses at f_1 and f_2 and third-order responses at $(2f_1 - f_2)$ and $(2f_2 - f_1)$ measured at low input power. Additionally, the physical constraint requires that the baseband impedance have an even real part, and an odd imaginary part. So the impedance at baseband frequency is parameterized as $Z_B = Z_{B_{\text{even}}} + \jmath Z_{B_{\text{odd}}}$. The linear parameters at second and higher harmonic bands of the carrier frequency are parameterized as constants. Fitting to the first adjacent channel IM response across power, the third-order nonlinear parameters will be well resolved. Higher-order nonlinear parameters can be accurately resolved if higher-order adjacent channel responses are fitted.

Since discrete tone testing is used, the model fit error to be minimized is the relative difference of the measured and simulated IM responses. These relative differences are grouped over the range of input powers, tone-spacings, etc., to create the error vector

$$\text{ErrVector} = \left[\left[\left[\frac{\Re(\text{IM}_{\text{meas}}) - \Re(\text{IM}_{\text{sim}})}{\Re(\text{IM}_{\text{meas}})}; \frac{\Im(\text{IM}_{\text{meas}}) - \Im(\text{IM}_{\text{sim}})}{\Im(\text{IM}_{\text{meas}})} \right]_{\text{P}_{\text{in}}} \right]_{\text{ToneSpace}} \times Wt_{\text{IM}} \right]_{\text{IM}}. \quad (4.2)$$

Generally, IM products resulting from lower-order NLTFs should be modeled more accurately

than higher-order NLTFs, thus a weighting factor $Wt_{\text{IMproduct}}$ is applied to each element of the error vector. Empirically it was found that weighting the relative error at f_1 and f_2 by a factor of ten is sufficient.

4.3.3 Optimization Settings and Hybrid GA Performance

There is some flexibility in choosing the settings of the identification algorithm and the hybrid GA optimizer. These settings partition the compute time between GA search and Nelder-Mead search. The more resources devoted to GA search the more local minimums will be found. The maximum number of consecutive iterations with improvement below the relative error tolerance was set to 3 for the higher-level identification algorithm, and to 10 in the local (Nelder-Mead) search portion of the hybrid GA optimizer. For both the identification algorithm and hybrid GA search, the relative error tolerance is defined as

$$\text{ErrTolerance}_{\text{relative}} = \frac{|\|\text{ErrVector}_i\| - \|\text{ErrVector}_{(i-1)}\||}{\min_i \|\text{ErrVector}\|} \tag{4.3}$$

and was set to 10%. Here i denotes iteration index, and $\min_i$ denotes the operation to select the minimum achieved at the ith iteration.

The GA population size was set to 250, meaning that 250 solution paths were followed simultaneously. One generation of the GA search yields a population of likely candidate solutions. From these the best five individuals were selected as trial solutions for each of five Nelder-Mead searches. The entire model identification process took 5 hours (GA search took 3.75 hours and the Nelder-Mead deterministic search took 1.25 hours) on an eight-core node clocked at 1.6 GHz. One particular feature is that function evaluations are efficiently distributed across the cluster, whereas a gradient-search algorithm confines the search to a single compute node.

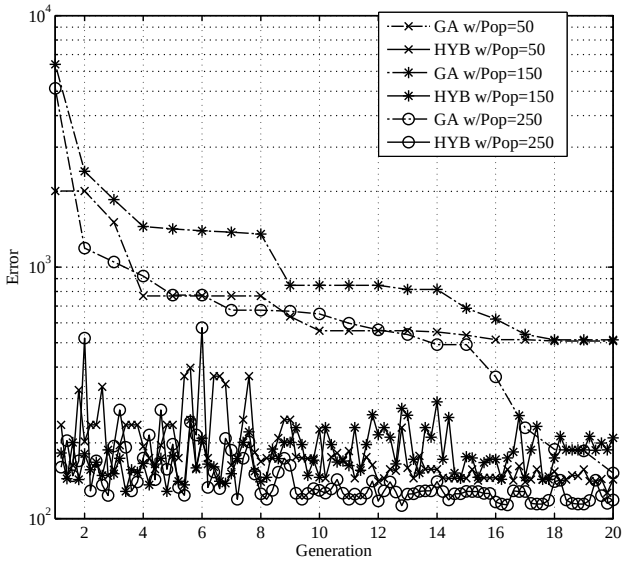
In GA search, the number of generations required can be deduced approximately from the plot of the lowest error of each GA generation before a local search is initiated. The size of the GA population was varied from 50 to 250. Fig. 4.9(a) plots the error across the generations for

the best individual in each GA generation and for the 5 individuals selected for local searches. Note that the error reaches numerous plateaus, before the mutate and crossover operators take the population to regions of the search space with even lower errors. Fig. 4.9(b) plots the value of an example model parameter after local search, which shows reduced variation after a certain number of plateaus, supporting the hypothesis that individuals around the multiple local optimum exist during the earlier GA generations, before being eliminated as GA converges to the global optimum. The parameter variation shows stronger correlation with the plateauing effect than the error level, which may show rippling though generally tending towards lower values. Thus, the number of plateaus can be specified in place of the number of generations. Here, 3 plateaus is sufficient for a variety of population sizes, and the corresponding number of GA generations is kept under 20. In addition, the first few generations maybe be skipped since GA has not undergone enough evolution to properly direct the search space.

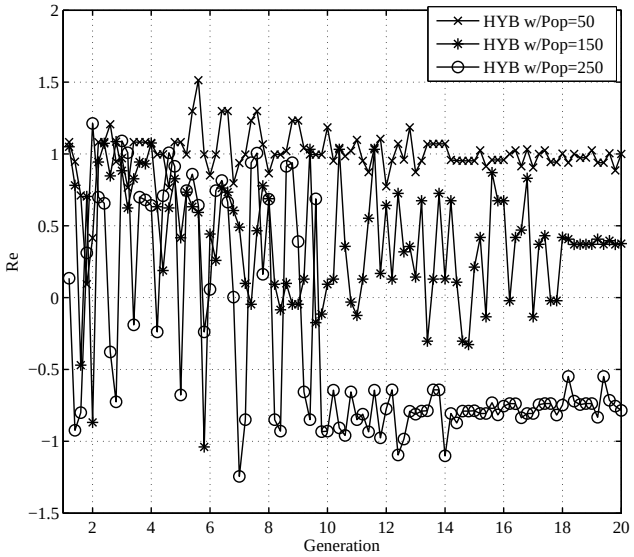
Fig. 4.10 plots the corresponding number of function evaluations. Since local search is applied to a larger number of individuals at each GA generation, the number of function evaluations required by GA and local search becomes comparable.

To choose the population size we note that the correlation between parameter variation and plateauing effect in GA error is also stronger for larger populations, without significantly increasing the number of function evaluations. Thus it is beneficial to set the population size reasonably large, without regard to the size of the portion used in local search. While a larger population size requires more function evaluations in the GA search, fewer generations are needed to achieve convergence due to the increased sampling of the search space.

The trade-off between solution space exploration and the amount of function evaluation depends on the number of generations, the population size and the portion selected for local search. Fig. 4.11 plots the error and the number of function evaluations, in order of increasing error for several hybrid GA solutions. In comparison, it is not possible to achieve the same levels of error using standard GA alone, which requires anywhere between the same to seven



(a)



(b)

Figure 4.9: Across the generations: (a) error for the best individual in each GA generation and for the 5 individuals selected for local searches; and (b) corresponding values of an estimated model parameter for the 5 individuals. There is correlation between the error plateaus in GA, and the error level after the proposed Hybrid-GA method.

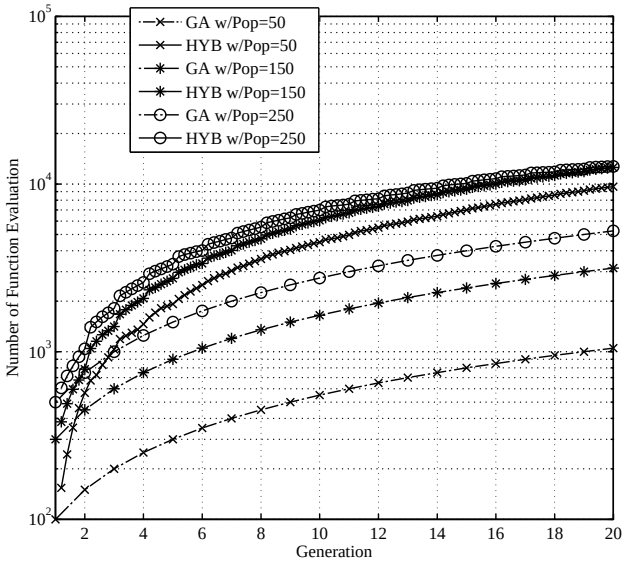


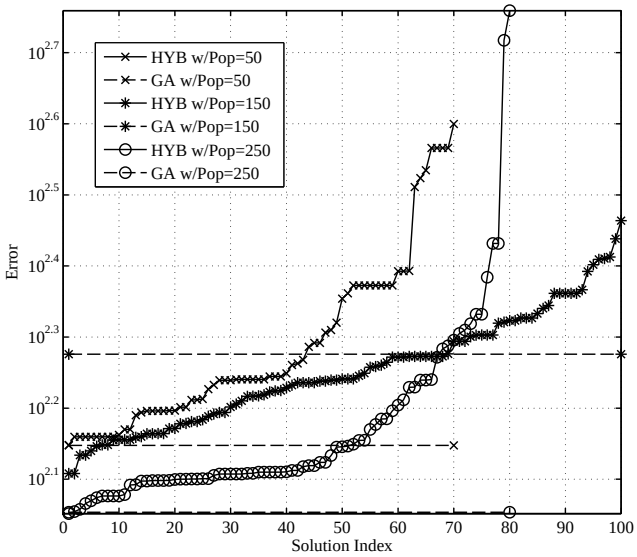
Figure 4.10: Across the generations, the number of function evaluations for the 5 individuals selected for local searches.

times the number of function evaluations.

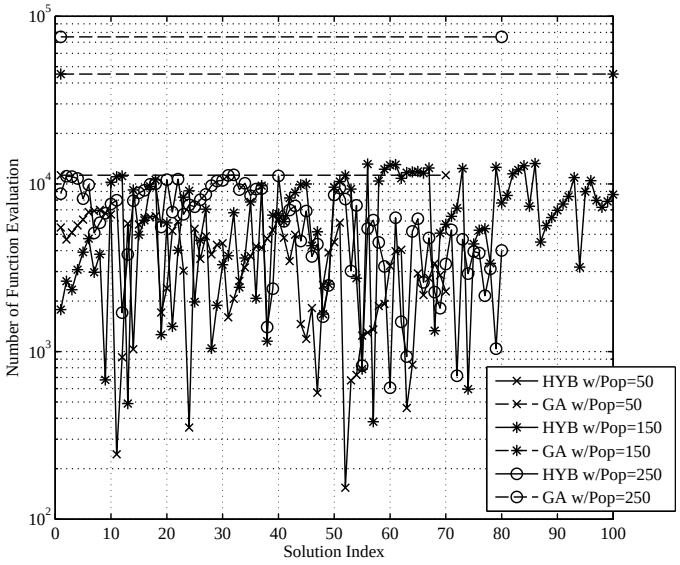
4.3.4 Model Fitting Result

At the termination of the identification algorithm, models can be selected by plotting the model parameter values in order of increasing error. Fig. 4.12 plots one parameter for the first forty models with the lowest error. The variation in the parameter values indicates the presence of multiple optimum solutions. With a given error threshold, multiple optimum solutions can then be selected.

The modeled and measured responses are compared for the in-band fundamental responses at f_1 and f_2 in Fig. 4.13, and the adjacent channel IM products at $(2f_1 - f_2)$ and $(2f_2 - f_1)$ in Fig. 4.14. The results show good fit up to the 1 dB compression level for the two-tone fundamental.



(a)



(b)

Figure 4.11: Error and the number of function evaluations versus index to the final solutions of hybrid GA method, with the index sorted in order of increasing error. Compared to hybrid GA, the standard GA requires anywhere between the same to seven times the number of function evaluations.

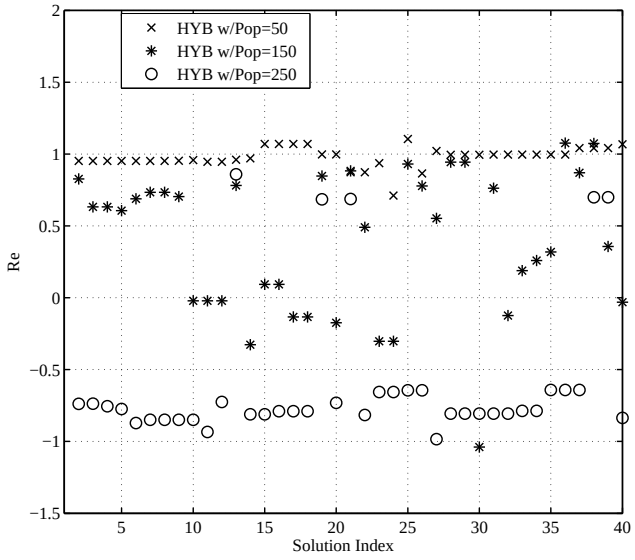
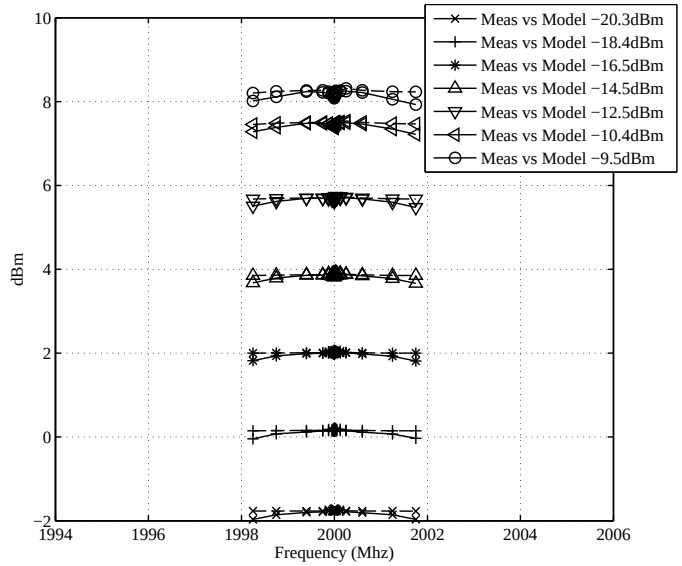


Figure 4.12: Values of an estimated model parameter with the first forty lowest errors.

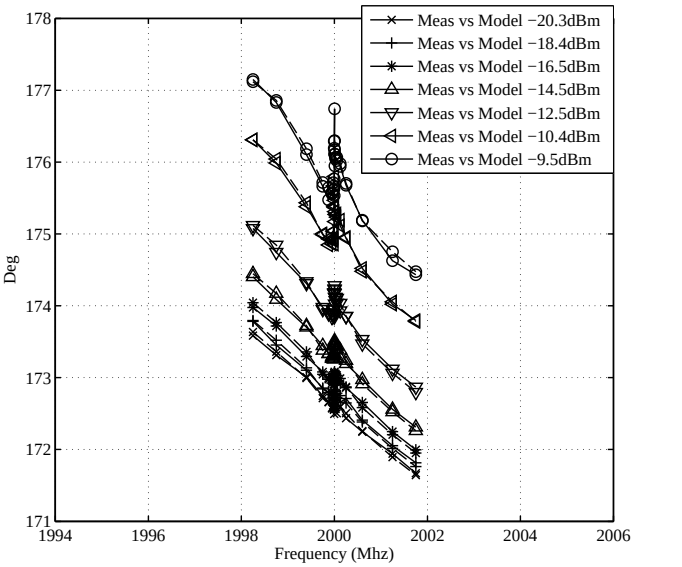
The estimated linear parameter at the baseband frequency is shown in Fig. 4.15, which has even magnitude and odd phase as expected. The high-pass nature corresponds to the inductive bias network.

4.4 Discussion

Convergence analysis of identification algorithms is difficult in general, especially when stochastic optimizers are involved. For identification algorithms developed for simple Wiener and Hammerstein types of models using deterministic optimizers, local convergence proof was previously shown, with the linear part rescaled at each frequency between iterations because of the parameter uniqueness modulo scaling.

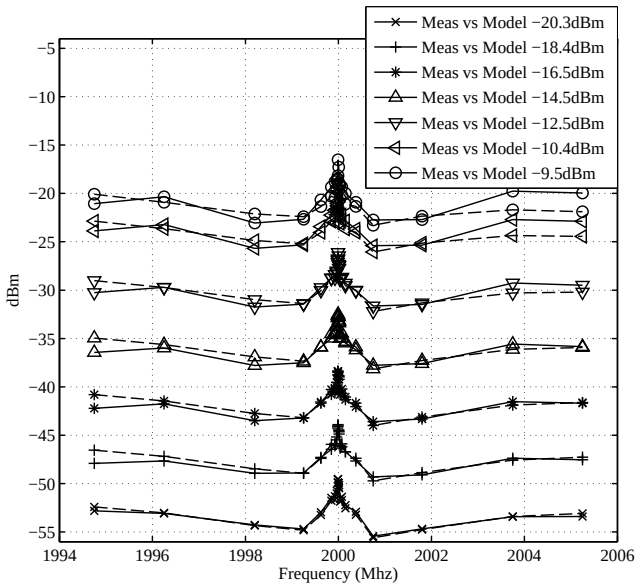


(a)

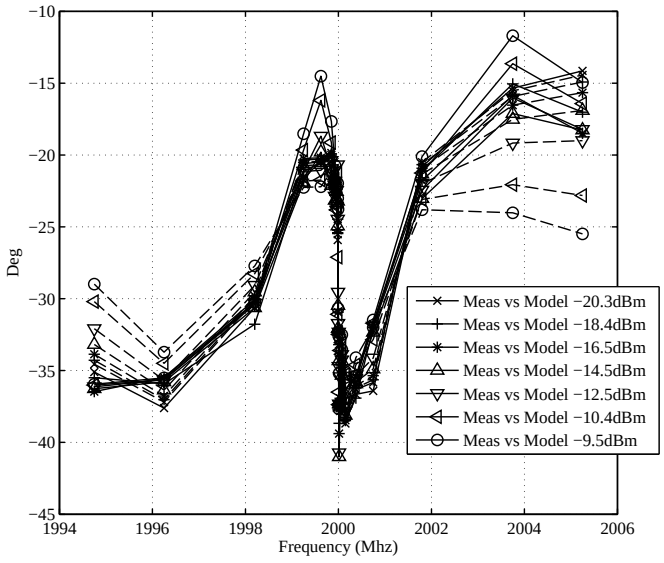


(b)

Figure 4.13: Modeled versus measured response: (a) fundamental power; (b) fundamental phase.

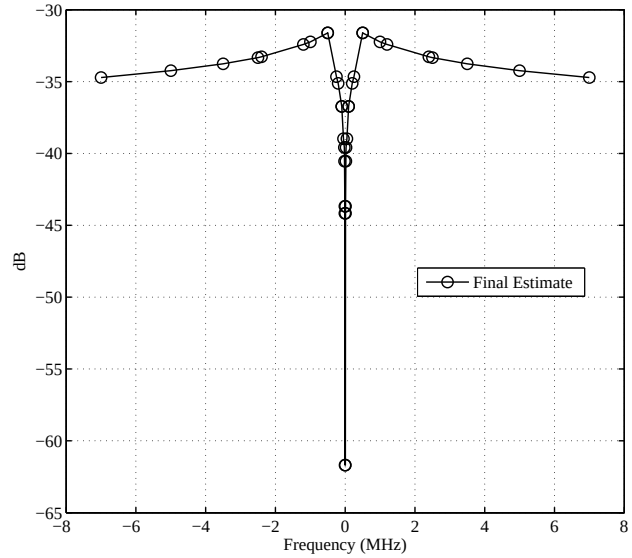


(a)

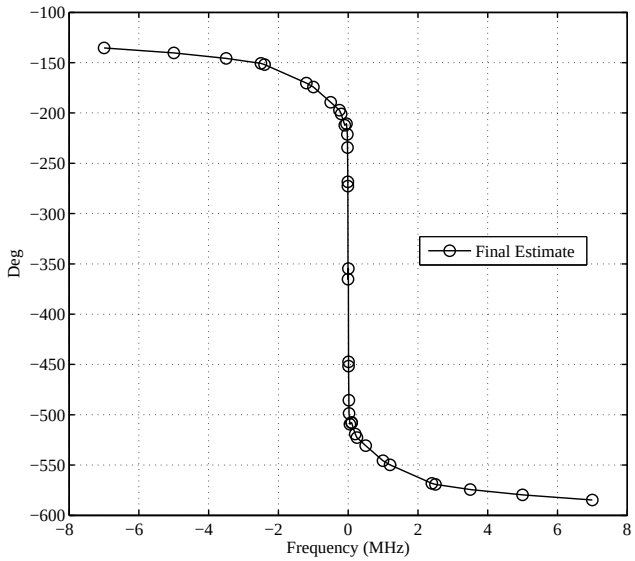


(b)

Figure 4.14: Modeled versus measured response: (a) IM3 power; and (b) IM3 phase.



(a)



(b)

Figure 4.15: Estimated linear parameter at baseband frequency: (a) magnitude; and (b) phase. The high-pass nature corresponds physically to the inductive nature nature of the bias network.

4.4.1 Model Parameter and Structure Uniqueness

Since the internal signals cannot be measured, model parameters can only be determined within arbitrary scaling constants. Thus the relative distortion levels of the internal signals cannot be determined. For example, by inserting scaling multipliers α at the input and β at the output of an amplifier, the input referred third-order harmonic distortion is modified by a factor $\alpha\beta$. What can be determined with certainty is the relative frequency response of the linear dynamic blocks.

As shown in [73] and [80], establishing the uniqueness of model parameters and model structures for even simple models requires careful examination of the NLTFs. For general nonlinear models of complex RF hardware, especially multi-stage systems, it may not be feasible to establish uniqueness without automated algebraic manipulation. In [73] it was noted that the more complicated a system is, the more likely that the system's parameters can be uniquely estimated modulo a scaling, which is reasonable since solving complicated NLTFs for constraining kernel relation tend to yield more constraint equations than unknown parameters. However for model uniqueness, NLTFs of the models being compared must be different for any set of model parameters. This is difficult to assess when the structures differ only slightly from each other.

The uniqueness question may be addressed by examining the multiple optimum solutions found, assuming that the optimizer has found all of the good solutions. For parameter uniqueness the necessary condition is that the solutions around the lowest error level be related modulo a scaling. For model uniqueness the necessary condition is for the lowest error level achieved by the two models to be significantly different, as quantified by the quality of the fit to the measured input-output behavior.

4.4.2 Comparison to Circuit Design

The proposed method can be compared to circuit design process which evolves in complexity as well. Indeed, the optimizer is trying to come up with circuit parameters in both cases, although

there are fundamental differences.

For example, in circuit design the strongly nonlinear, often exponential, characteristics of semiconductor components are modified by feedback and passive elements to produce a weakly nonlinear response. In extracting the parameters of a grey-box model it is not necessary to capture the intrinsic strong nonlinearities. So the element models used are at different layer of abstraction, as shown in Fig. 4.16. In circuit design, the active devices are Gummel-Poon based BJT models or Angelov based FET models, and the parameters estimated are active device sizing, passive component values. Due to circuit complexity and the large number of parameters, optimization often requires a working design as the initial solution. In nonlinear model identification, the element models are chosen to reduce the complexity. Each active stage is modeled as unilateral controlled sources, each passive network is modeled in the frequency domain, and any significant physical feedback is modeled explicitly.

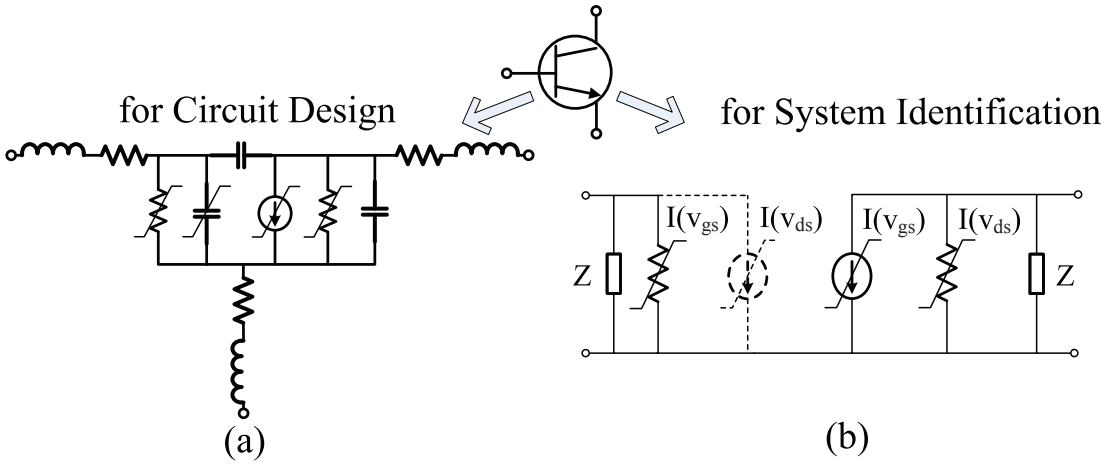


Figure 4.16: The different levels of element model abstraction for: (a) circuit design; and (b) RF model identification.

Also higher-level identification algorithm is devised here to bootstrap the identification process, where as all parameters are optimized simultaneously in circuit design.

Lastly the non-iterative Volterra circuit simulator is used in place of iterative ones such as SPICE or harmonic balance, which saves computation time. While Volterra series is only suitable for weakly nonlinear systems, it can model up to 1 dB gain compression which is suitable for the current application.

4.5 Conclusions

The main result of the work presented here is a reorganization of the way models are extracted for RF devices. Instead of a skilled user being involved throughout the model fitting process, user input is directed first at establishing several candidate grey-box models, and then establishing a measurement regime.

A generalized model identification algorithm using a hybrid GA optimizer with Nelder-Mead simplex search around local optimums, presents a large number of good candidate models. A skilled user then chooses the best solution at the end of the procedure based on intuition and experience.

Each step of the identification process was devised to minimize the amount of user intervention. The partitioning of labor is similar to that in electronic design automation wherein engineers with good device physics backgrounds are often involved in model development. Now the suite of standard grey-box models and measurement protocols can be developed by one set of engineers and used by others, perhaps with a different skill set, who are directly involved in applied model identification.

The hybrid GA optimizer was developed to specifically track local optimum solutions, rather than discard them as typically done. This is particularly suited to incompletely-observed RF systems, since initial solution close to the optimums can no longer be easily found for complex models. The approach can also be adapted to fit several grey-box structures simultaneously, and can efficiently use distributed computing resources.

The proposed grey-box model identification method developed here was successfully demonstrated in modeling a multi-stage power amplifier with significant long-term memory effects.

The multi-stage PA was modeled up to 1 dB compression of the two-tone fundamental.

Chapter 5

Direct Calculation Method for Volterra Circuit Analysis

5.1 Introduction

In chapter 4, a modeling strategy was developed to minimize the user intervention required. For more generalized model fitting and elimination of user tasks that can be automated, general purpose circuit simulation is used to relate the model parameters to the measured data which typically consists of the amplitude and phase of discrete tone responses to swept-power and swept-frequency one-tone and multi-tone excitations. Here the previously typical hand-crafted Volterra NLTF analysis specific to the particular model being developed, is replaced by a symbolic Volterra circuit simulator to automate the model analysis and response calculation.

5.1.1 NLTF Method

The original application of Volterra series to nonlinear circuit analysis is due to Wiener [90]. Further studies [91] [92] described the harmonic input method and the nonlinear current method for finding NLTFs, from which the output IM response can be calculated. The basis of these

methods is assuming a solution of the form

$$y(t) = \sum_{n=1}^{N} \left[\frac{1}{2^n} \sum_{q_1=-Q}^{Q} \cdots \sum_{q_n=-Q}^{Q} A_{q_1} ... A_{q_n} H_n(f_{q_1}, \ldots, f_{q_n}) e^{j2\pi(f_{q_1}+...+f_{q_n})t} \right] \tag{5.1}$$

for the response to a real Q-tone excitation

$$x(t) = \sum_{q=-Q}^{Q} \frac{1}{2} A_q e^{j2\pi f_q t} \tag{5.2}$$

with A_q being the amplitude of the qth tone. Spectral balance in the circuit equations then yields the NLTF H_n.

Volterra circuit simulation is usually done numerically, implemented first in SIGNCAP [91], then in C/NL2, and most recently in the AWR design environment. Active devices are modeled up to the equivalent of a third-degree polynomial. (Note that there is an equivalence between the order of a polynomial description and the order of a Volterra NLTF [93].)

The first symbolic circuit simulator incorporating the nonlinear current method (coded in LISP [94]) was geared towards analog circuits, and provided numerous approximation methods to simplify the output expressions. A later work [95] implementing the harmonic input method in Mathematica was aimed at circuit optimization. Closed form expressions identify the dominant circuit parameters and provide insight to the circuit designer as to the sources of distortion. In addition these can be pre-computed and reused to speed up numerical calculation.

5.1.2 Direct Calculation Method

Alternatively the direct calculation method in [6], which calculates the IM response directly, can easily handle nonlinear circuits with multiple inputs (such as mixers) which would otherwise require tensor manipulation to find the NLTFs. The method assumes a solution of the form

$$y(t) = \frac{1}{2} \sum_{\{f_{\mathrm{IM}}=k_1 f_1+...+k_Q f_Q\}} Y(f_{\mathrm{IM}}) e^{j2\pi f_{\mathrm{IM}} t} \tag{5.3}$$

where the summation is over both positive and negative IM frequencies. For a real circuit, the responses are conjugate symmetric in frequency which means $Y(f_{\mathrm{IM}}) = Y^*(-f_{\mathrm{IM}})$. Spectral balance in the circuit equations at the IM response frequencies f_{IM} then yields $Y(f_{\mathrm{IM}})$.

However the expression for the nonlinear current source was only presented up to third order, and the method was not in an algorithmic form suitable for implementation in a circuit simulator. Also there is a lack of differentiation between contributions of different order to the same f_{IM}, which together with the limited nonlinear order that can be considered, leads to compression effects not being accounted for.

5.1.3 Proposed Improvement to Direct Calculation Method

This Section presents the direct calculation method in an algorithmic form suited to handle arbitrarily high degree of device nonlinearity. The expression for nonlinear current sources is determined by recursively finding the lower-order IM response due to the controlling variables of nonlinearities. The assumed solution

$$y(t) \;= \sum_{n=1}^{N} \left[\frac{1}{2} \sum_{\{f_{\mathrm{IM}_n} = k_1 f_1 + \ldots + k_Q f_Q\}} Y(f_{\mathrm{IM}_n}) e^{j2\pi f_{\mathrm{IM}_n} t} \right] \tag{5.4}$$

follows directly from (5.1), by collating frequency mixing that contributes to the same nth order response at f_{IM}. Since there are many contributions of different (and arbitrarily high) order that contribute to the same IM response, the compression effect can be handled.

This work implements the improved direct calculation method in a symbolic Volterra circuit simulator in Matlab. The choice of a symbolic implementation is important for the application in this paper. As discussed in Section 4.2.3, the IM response used in the identification algorithm can only be chosen when given the closed form expressions.

5.2 Improved Direct Calculation Method

5.2.1 General Formulation of Circuit Equations

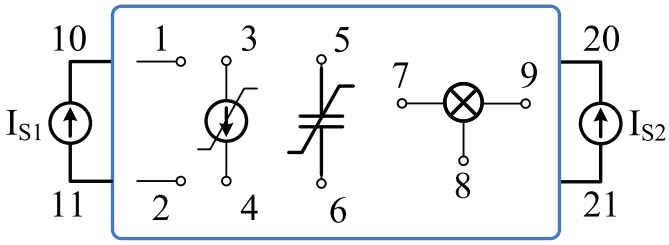


Figure 5.1: General nonlinear network.

For nodal analysis of a general nonlinear network shown in Fig. 5.1, Kirchoff's current law is applied. This yields the following set of circuit equations at node 10 and 20 for the input excitations, node 3 for the nonlinear trans-conductance, node 5 for the nonlinear capacitance, and node 9 for the behavioral multiplier

$$\ldots = \ldots + i_{s1}(t) \tag{5.5}$$

$$\ldots = \ldots + i_{s2}(t) \tag{5.6}$$

$$\ldots + g_1(v_1(t) - v_2(t)) = \ldots - \sum_{r=2}^{N} g_r(v_1(t) - v_2(t))^r \tag{5.7}$$

$$\ldots + c_1 \frac{d}{dt}(v_5(t) - v_6(t)) = \ldots - \sum_{r=2}^{N} c_r(v_5(t) - v_6(t))^{r-1} \frac{d}{dt}(v_5(t) - v_6(t)) \tag{5.8}$$

$$\ldots = \ldots + g_{\text{mix}} v_7(t) v_8(t). \tag{5.9}$$

For multi-tone input from multiple inputs, the node voltage solution assumes the form (5.4).

Using this representation of the circuit solution, the previous set of circuit equations become

$$\ldots = \ldots + \tfrac{1}{2}\left(I_{\mathrm{s1}}e^{j2\pi f_1 t} + I_{\mathrm{s1}}^{*}e^{-j2\pi f_1 t}\right) \tag{5.10}$$

$$\ldots = \ldots + \tfrac{1}{2}\left(I_{\mathrm{s2}}e^{j2\pi f_2 t} + I_{\mathrm{s2}}^{*}e^{-j2\pi f_2 t}\right) \tag{5.11}$$

$$\begin{aligned}
&\ldots + g_1\left[\tfrac{1}{2}\sum_{n=1}^{N}\sum_{\{f_{\mathrm{IM}_n}\}}(V_1(f_{\mathrm{IM}_n}) - V_2(f_{\mathrm{IM}_n}))e^{j2\pi f_{\mathrm{IM}_n}t}\right] = \\
&\ldots - \sum_{r=2}^{N} g_r\left[\tfrac{1}{2}\sum_{n=1}^{N-1}\sum_{\{f_{\mathrm{IM}_n}\}}(V_1(f_{\mathrm{IM}_n}) - V_2(f_{\mathrm{IM}_n}))e^{j2\pi f_{\mathrm{IM}_n}t}\right]^{r}
\end{aligned} \tag{5.12}$$

$$\begin{aligned}
&\ldots + c_1\tfrac{d}{dt}\left[\tfrac{1}{2}\sum_{n=1}^{N}\sum_{\{f_{\mathrm{IM}_n}\}}(V_5(f_{\mathrm{IM}_n}) - V_6(f_{\mathrm{IM}_n}))e^{j2\pi f_{\mathrm{IM}_n}t}\right] = \\
&\ldots - \sum_{r=2}^{N} c_r\left[\tfrac{1}{2}\sum_{n=1}^{N-1}\sum_{\{f_{\mathrm{IM}_n}\}}(V_5(f_{\mathrm{IM}_n}) - V_6(f_{\mathrm{IM}_n}))e^{j2\pi f_{\mathrm{IM}_n}t}\right]^{r-1} \\
&\qquad \tfrac{d}{dt}\left[\tfrac{1}{2}\sum_{n=1}^{N-1}\sum_{\{f_{\mathrm{IM}_n}\}}(V_5(f_{\mathrm{IM}_n}) - V_6(f_{\mathrm{IM}_n}))e^{j2\pi f_{\mathrm{IM}_n}t}\right]
\end{aligned} \tag{5.13}$$

$$\begin{aligned}
&\ldots = \ldots + g_{\mathrm{mix}}\left[\tfrac{1}{2}\sum_{n=1}^{N-1}\sum_{\{f_{\mathrm{IM}_n}\}}(V_7(f_{\mathrm{IM}_n}))e^{j2\pi f_{\mathrm{IM}_n}t}\right] \\
&\qquad \left[\tfrac{1}{2}\sum_{n=1}^{N-1}\sum_{\{f_{\mathrm{IM}_n}\}}(V_8(f_{\mathrm{IM}_n}))e^{j2\pi f_{\mathrm{IM}_n}t}\right].
\end{aligned} \tag{5.14}$$

The circuit's nodal response is found by solving the spectral balance equations above at each f_{IM}. First the linear response is determined by solving the circuit equations at frequencies of the multi-tone input. Then the nonlinear response is determined by solving equations of the same form but with different current source expression on the right-hand side:

$$\mathbf{Y}(f_{\mathrm{IM}})\frac{\overrightarrow{V}(f_{\mathrm{IM}})}{2} = \overrightarrow{I_{\mathrm{NL}}}(f_{\mathrm{IM}}), \tag{5.15}$$

where $\mathbf{Y}$ is the node-admittance matrix of the linearized circuit, $\overrightarrow{V}$ is the vector of node voltages to be solved, and $\overrightarrow{I_{\mathrm{NL}}}$ is the nonlinear current source vector created by grouping terms of the same frequency f_{IM} on the right-hand side. The $\frac{1}{2}$ factor in the equation is due the assumption of a real circuit which only has real responses that are conjugate symmetric in

frequency $V(f_{\mathrm{IM}}) = V^*(-f_{\mathrm{IM}})$.

5.2.2 Determining the Nonlinear Current Source

Determining $\overrightarrow{I_{\mathrm{NL}}}(f_{\mathrm{IM}})$ is the main problem in Volterra circuit analysis. In the following, the algorithm used in evaluating the nonlinear current source in the NLTF method is first described, to introduce the notation and serve as a reference for comparison. Then the proposed algorithm used in the direct calculation method is presented, for several types of circuit and behavioral nonlinear sources previously encountered.

Zero-Memory Polynomial Nonlinearity in NLTF method

For a power series expansion of the nonlinear characteristic of a device, the nonlinear current accounting for contributions up to order N is

$$i_{\mathrm{NL}}(t) = \sum_{r=2}^{N} g_r \left[\sum_{n=1}^{N-1} v_n(t) \right]^r \tag{5.16}$$

where the Nth order contribution can be generated from degree $r = N$ as ${v_1}^N$, or from $r = 2$ as $v_{N-1}v_1$.

To facilitate evaluation, $i_{\mathrm{NL}}(t)$ can be broken down into its nth order and rth degree contributions [92]:

$$\begin{aligned} i_{\mathrm{NL}}(t) &= \sum_{n=2}^{N} i_n(t) \\ &= \sum_{n=2}^{N} \left(\sum_{r=2}^{n} g_r i_{n,r}(t) \right) \end{aligned} \tag{5.17}$$

where $i_{n,r}(t)$ is the nth order component generated by the rth degree term and can be recursively

evaluated [92] as

$$\begin{aligned} i_{n,r}(t) &= \sum_{i=1}^{n-r+1} v_{\rm i}(t) i_{n-i,r-1}(t) \\ &= \sum_{\{[n_1 \ldots n_r]\}} \big(C v_{n_1}(t) \ldots v_{n_r}(t) \text{ s.t } (n_1 + \ldots + n_r = n) \big). \end{aligned} \tag{5.18}$$

Here the second equation shows its general form as a summation over all contributing vectors $[n_1 \ldots n_r]$, and C is a constant coefficient to be determined from the recursive evaluation.

In the NLTF method, an incommensurate positive-frequency exponential n-tone probing signal is applied at the input

$$\sum_{q=1}^{n} 2e^{j2\pi f_q t}, \tag{5.19}$$

and the nth order Volterra series output (5.1) is

$$v_n(t) = \sum_{q_1=1}^{n} \ldots \sum_{q_n=1}^{n} V_n(f_{q_1}, \ldots, f_{q_n}) e^{j2\pi(f_{q_1}+\ldots+f_{q_n})t}. \tag{5.20}$$

From combinatorics, the response at $f_1 + \ldots + f_n$ is only contributed to by the nth order Volterra series output (5.20):

$$V_n(f_1 + \ldots + f_n) = n! \overline{V_n(f_1, \ldots, f_n)} \tag{5.21}$$

where the over-line denotes the arithmetic average over $n!$ permutations of the arguments in the nth order NLTF $V_n(f_1, \ldots, f_n)$. Note that only positive frequency responses are possible for the positive-frequency exponential n-tone probing signal, so the $\frac{1}{2}$ factor in (5.15) can be removed. By solving for the circuit response at $f_1 + \ldots + f_n$ using (5.21), the spectral balance equation (5.15) becomes

$$\mathbf{Y}(f_1 + \ldots + f_n) \overrightarrow{V_n}(\overline{f_1, \ldots, f_n}) = \frac{\overrightarrow{I_{\rm NL}}(f_1 + \ldots + f_n)}{n!}, \tag{5.22}$$

and the nth order NLTF for each circuit node voltage is then

$$\overrightarrow{V_n}(\overline{f_1,\ldots,f_n}) = \mathbf{Y}^{-1}(f_1+\ldots+f_n)\frac{\overrightarrow{I_{\rm NL}}(f_1+\ldots+f_n)}{n!}. \tag{5.23}$$

To evaluate the nonlinear current (5.17) which depends on the lower-order NLTF, the general expression for $v_n(t)$ (5.20) is substituted into (5.18) yielding

$$\begin{aligned} i_{n,r}(t) \quad = \quad & \sum_{\{[n_1\ldots n_r]\}} C\left[\sum_{q_1=1}^{n}\cdots\sum_{q_{n_1}=1}^{n} V_{n_1}(f_{q_1},\ldots,f_{q_{n_1}})e^{j2\pi(f_{q_1}+\ldots+f_{q_{n_1}})t}\right] \\ \cdots \quad & \left[\sum_{q_1=1}^{n}\cdots\sum_{q_{nr}=1}^{n} V_{n_r}(f_{q_1},\ldots,f_{q_{nr}})e^{j2\pi(f_{q_1}+\ldots+f_{q_{nr}})t}\right]. \end{aligned} \tag{5.24}$$

Combinatorics is then applied to find $i_{n,r}(t)$ contributions to the nonlinear current at $f_1+\ldots+f_n$:

$$I_{n,r}(f_1+\ldots+f_n) = n!\sum_{\{[n_1\ldots n_r]\}} C\overline{\left[V_{n_1}(f_1,\ldots,f_{n_1})\right]\ldots\left[V_{n_r}(f_1,\ldots,f_{n_r})\right]} \tag{5.25}$$

which are listed in Table 5.1 up to the fifth order. To evaluate the total nonlinear current at $f_1+\ldots+f_n$, (5.25) is substituted into (5.17).

Starting from the lowest-order transfer functions, all NLTFs can be found by repeating the above steps. The actual response at arbitrary $f_{\rm IM}$ for a Q-tone real excitation is then evaluated from the Volterra series (5.1).

Due to the fact that the response at all possible $f_{\rm IM}$ can be related back to the same set of NLTFs, and that the incommensurate positive-frequency exponential nature of the n-tone probing signal enables the application of combinatorics, the nonlinear current (5.17) can be non-recursively evaluated.

Table 5.1: For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the NLTF up to the fifth order.

n	r	$\{[n_1 \ldots n_r]\}$	$g_r I_{n,r}(f_1 + \ldots + f_n)$
2	2	[1 1]	$2!g_2\big(\overline{V_1(f_1)V_1(f_2)}\big)$
3	2	[1 2]	$3!g_2\big(2\overline{V_1(f_1)V_2(f_2,f_3)}\big)$
	3	[1 1 1]	$3!g_3\big(\overline{V_1(f_1)V_1(f_2)V_1(f_3)}\big)$
4	2	[1 3] [2 2]	$4!g_2\big(2\overline{V_1(f_1)V_3(f_2,f_3,f_4)} + \overline{V_2(f_1,f_2)V_2(f_3,f_4)}\big)$
	3	[1 1 2]	$4!g_3\big(3\overline{V_1(f_1)V_1(f_2)V_2(f_3,f_4)}\big)$
	4	[1 1 1 1]	$4!g_4\big(\overline{V_1(f_1)V_1(f_2)V_1(f_3)V_1(f_4)}\big)$
5	2	[1 4] [2 3]	$5!g_2\big(2\overline{V_1(f_1)V_4(f_2,f_3,f_4,f_5)} + 2\overline{V_2(f_1,f_2)V_3(f_3,f_4,f_5)}\big)$
	3	[1 1 3] [1 2 2]	$5!g_3\big(3\overline{V_1(f_1)V_1(f_2)V_3(f_3,f_4,f_5)} + 3\overline{V_1(f_1)V_2(f_2,f_3)V_2(f_4,f_5)}\big)$
	4	[1 1 1 2]	$5!g_4\big(4\overline{V_1(f_1)V_1(f_2)V_1(f_3)V_2(f_4,f_5)}\big)$
	5	[1 1 1 1 1]	$5!g_5\big(\overline{V_1(f_1)V_1(f_2)V_1(f_3)V_1(f_4)V_1(f_5)}\big)$

Zero-Memory Polynomial Nonlinearity in Direct Calculation Method

In the direct calculation method, finding the nth order response at an arbitrary f_{IM} from the circuit equations requires evaluating the nonlinear current (5.17) which depends on the lower-order responses at other f_{IM}. These lower-order responses at other f_{IM} may not have been computed yet, and requires separate evaluation which may entail finding responses of even lower-order and at another f_{IM} not yet considered. Thus this is a recursive evaluation process. This is unlike the situation with the NLTF method where all lower-order NLTFs have been computed and are available for computing the current order NLTF. In the original direct calculation method, the evaluation process is also non-recursive because contributions of different order at the same f_{IM} are not distinguished. The nonlinear current expressions were only presented for 2-tone responses at $f_1 \pm f_2$, $2f_1$, $2f_2$, $2f_1 \pm f_2$, $2f_2 \pm f_1$, $3f_1$ and $3f_2$. For estimating the first order nonlinear behavior using hand calculation, this may be suitable.

Here finding the nonlinear current expression which is independent of the particular circuit network, entails finding the dependence on lower-order responses for responses at all possible

f_{IM} of all orders. For a Q-tone input, all possible response frequencies f_{IM} of all orders,

$$\begin{aligned} f_{\mathrm{IM}_n} &= \overrightarrow{m_n} \cdot \begin{bmatrix} f_{-Q} & \dots & f_{-1} & f_1 & \dots & f_Q \end{bmatrix} \\ &= k_1 f_1 + \ldots + k_Q f_Q, \end{aligned} \tag{5.26}$$

can be found by exhaustively searching through all nth order mixing vectors,

$$\overrightarrow{m_n} = \begin{bmatrix} m_{-Q} & \dots & m_{-1} & m_1 & \dots & m_Q \end{bmatrix} \text{ s.t } (m_{-Q} + \ldots + m_Q = n), \tag{5.27}$$

and collating the distinct frequencies.

Substituting the assumed solution (5.4) into (5.18), each $i_{n,r}$ term becomes

$$\begin{aligned} i_{n,r}(t) = \sum_{\{[n_1 \ldots n_r]\}} & C \left[\frac{1}{2} \sum_{\{f_{\mathrm{IM}_{n_1}}\}} V_{n_1}(f_{\mathrm{IM}_{n_1}}) e^{j2\pi f_{\mathrm{IM}_{n_1}} t} \right] \\ & \ldots \left[\frac{1}{2} \sum_{\{f_{\mathrm{IM}_{n_r}}\}} V_{n_r}(f_{\mathrm{IM}_{n_r}}) e^{j2\pi f_{\mathrm{IM}_{n_r}} t} \right] \end{aligned} \tag{5.28}$$

which contains many spectral components of the form

$$I_{n,r}(f_{\mathrm{IM}_n}) = \sum_{\{[n_1 \ldots n_r]\}} C \frac{1}{2}^r \left[\sum_{\{[f_{\mathrm{IM}_{n_1}} \ldots f_{\mathrm{IM}_{n_r}}]\}} V_{n_1}(f_{\mathrm{IM}_{n_1}}) \ldots V_{n_r}(f_{\mathrm{IM}_{n_r}}) \right]. \tag{5.29}$$

To reconstruct the nonlinear current for use during the recursive circuit analysis, the necessary dependence information are all f_{IM} generated in $i_{n,r}$ and the contributing $[n_1 \ldots n_r]$ and $[f_{\mathrm{IM}_{n_1}} \ldots f_{\mathrm{IM}_{n_r}}]$. For hand calculation, this dependence information maybe presented in a table format as in the original direct calculation method, though the look-up process is more cumbersome due to the complex dependence associated with higher order responses. For computer implementation, a data structure is more suitable for storing this dependence information, which is recalled to evaluate the nonlinear current during the recursive circuit analysis for a particular circuit.

As an example for a 2-tone input at frequencies f_1 and f_2, the nonlinear current expressions needed to evaluate the responses up to the fifth order are given in Table 5.2 for $f_2 - f_1$, Table 5.3 for $f_1 + f_2$, Table 5.4 for $2f_1$, Table 5.5 for f_1, Table 5.6 for $2f_1 - f_2$, Table 5.7 for $2f_1 + f_2$, and Table 5.8 for $3f_2$.

For comparison, similar tables were presented in [6] for the original direct calculation method. In conformance with [6], the nonlinear current expression given in the tables is the right-hand side of

$$\mathbf{Y}(f_{\mathrm{IM}})\overrightarrow{V}(f_{\mathrm{IM}}) = 2\overrightarrow{I_{\mathrm{NL}}}(f_{\mathrm{IM}}) \tag{5.30}$$

which follows from (5.15). Here the nth order response at each $f_{\mathrm{IM}_n} = k_1 f_1 + k_2 f_2$ is denoted using a short hand notation as V_{n,k_1,k_2}.

In the analytical derivation, the circuit's response is assumed to be real and conjugate symmetric in frequency, i.e. $V(f_{\mathrm{IM}}) = V^*(-f_{\mathrm{IM}})$, so factors of $\frac{1}{2}^r$ must be tracked in the equations. It is also valid to implicitly include the $\frac{1}{2}$ factor in the excitation multi-tone amplitude and the computed $V(f_{\mathrm{IM}})$ responses, which simplifies the analysis and computer implementation. In either case, it is up to the user to use consistent notation throughout the analysis.

It is assumed that the controlling variable of the nonlinearity will contain responses at all f_{IM}. For certain nodes in the circuit such as those that are connected to the input excitation by a linear network, some f_{IM} other than the input Q-tone frequencies will be missing. These degenerate cases are skipped during the nonlinear current evaluation in the recursive circuit analysis.

Table 5.2: For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $f_2 - f_1$ up to the fifth order.

n	r	$\{[n_1 \dots n_r]\}$	$2g_r I_{n,r}(f_2 - f_1)$
2	2	[1 1]	$g_2\big(V_{1,0,1}V^*_{1,1,0}\big)$
3	2		
	3		
4	2	[1 3] [2 2]	$g_2\big([V_{1,1,0}V^*_{3,2,-1} + V_{1,0,1}V^*_{3,1,0} + V^*_{1,1,0}V_{3,0,1} + V^*_{1,0,1}V_{3,-1,2}]$ $+[V_{2,0,0}V_{2,-1,1} + V_{2,1,1}V^*_{2,2,0} + V_{2,0,2}V^*_{2,1,1}]\big)$
	3	[1 1 2]	$g_3\frac{3}{4}\big(2V_{1,1,0}V_{1,0,1}V^*_{2,2,0} + 2V_{1,1,0}V^*_{1,1,0}V_{2,-1,1}$ $+V_{1,0,1}V_{1,0,1}V^*_{2,1,1} + 2V_{1,0,1}V^*_{1,1,0}V_{2,0,0} + 2V_{1,0,1}V^*_{1,0,1}V_{2,-1,1}$ $+V^*_{1,1,0}V^*_{1,1,0}V_{2,1,1} + 2V^*_{1,1,0}V^*_{1,0,1}V_{2,0,2}\big)$
	4	[1 1 1 1]	$g_4\frac{3}{2}\big(V_{1,1,0}V_{1,0,1}V^*_{1,1,0}V^*_{1,1,0} + V_{1,0,1}V_{1,0,1}V^*_{1,1,0}V^*_{1,0,1}\big)$
5	2		
	3		
	4		
	5		

Table 5.3: For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $f_1 + f_2$ up to the fifth order.

n	r	$\{[n_1 \dots n_r]\}$	$2g_r I_{n,r}(f_1 + f_2)$
2	2	[1 1]	$g_2\big(V_{1,1,0}V_{1,0,1}\big)$
3	2		
	3		
4	2	[1 3] [2 2]	$g_2\big([V_{1,1,0}V_{3,0,1} + V_{1,0,1}V_{3,1,0} + V^*_{1,1,0}V_{3,2,1} + V^*_{1,0,1}V_{3,1,2}]$ $+[V_{2,0,0}V_{2,1,1} + V_{2,-1,1}V_{2,2,0} + V_{2,0,2}V^*_{2,-1,1}]\big)$
	3	[1 1 2]	$g_3\frac{3}{4}\big(V_{1,1,0}V_{1,1,0}V_{2,-1,1} + 2V_{1,1,0}V_{1,0,1}V_{2,0,0}$ $+2V_{1,1,0}V^*_{1,1,0}V_{2,1,1} + 2V_{1,1,0}V^*_{1,0,1}V_{2,0,2} + V_{1,0,1}V_{1,0,1}V^*_{2,-1,1}$ $+2V_{1,0,1}V^*_{1,1,0}V_{2,2,0} + 2V_{1,0,1}V^*_{1,0,1}V_{2,1,1}\big)$
	4	[1 1 1 1]	$g_4\frac{3}{2}\big(V_{1,1,0}V_{1,1,0}V_{1,0,1}V^*_{1,1,0} + V_{1,1,0}V_{1,0,1}V_{1,0,1}V^*_{1,0,1}\big)$
5	2		
	3		
	4		
	5		

Table 5.4: For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1$ up to the fifth order.

n	r	$\{[n_1 \dots n_r]\}$	$2g_r I_{n,r}(2f_1)$
2	2	[1 1]	$g_2 \frac{1}{2}\big(V_{1,1,0}V_{1,1,0}\big)$
3	2		
	3		
4	2	[1 3] [2 2]	$g_2\big([V_{1,1,0}V_{3,1,0} + V_{1,0,1}V_{3,2,-1} + V_{1,1,0}^*V_{3,3,0} + V_{1,0,1}^*V_{3,2,1}]$ $+[V_{2,0,0}V_{2,2,0} + V_{2,1,1}V_{2,-1,1}^*]\big)$
	3	[1 1 2]	$g_3 \frac{3}{4}\big(V_{1,1,0}V_{1,1,0}V_{2,0,0} + 2V_{1,1,0}V_{1,0,1}V_{2,-1,1}^*$ $+2V_{1,1,0}V_{1,1,0}^*V_{2,2,0} + 2V_{1,1,0}V_{1,0,1}^*V_{2,1,1} + 2V_{1,0,1}V_{1,0,1}^*V_{2,2,0}\big)$
	4	[1 1 1 1]	$g_4 \frac{1}{2}\big(V_{1,1,0}V_{1,1,0}V_{1,1,0}V_{1,1,0}^* + 3V_{1,1,0}V_{1,1,0}V_{1,0,1}V_{1,0,1}^*\big)$
5	2		
	3		
	4		
	5		

Capacitive Nonlinearity

For a capacitive nonlinearity, the nonlinear current accounting for contributions up to order N is

$$i_{\mathrm{NL}}(t) = \sum_{r=2}^{N} c_r \left[\sum_{n=1}^{N-1} v_n(t)\right]^{r-1} \frac{d}{dt}\left[\sum_{n=1}^{N-1} v_n(t)\right] \tag{5.31}$$

where the Nth order contribution can be generated from degree $r = N$ as ${v_1}^{N-1}\frac{dv_1}{dt}$, or from $r = 2$ as $v_{N-1}\frac{dv_1}{dt}$ and $v_1\frac{dv_{N-1}}{dt}$.

To reuse the results from the zero-memory polynomial nonlinearity, $i_{\mathrm{NL}}(t)$ can be rewritten as [92]

$$\begin{aligned} i_{\mathrm{NL}}(t) &= \frac{d}{dt}\sum_{r=2}^{N}\frac{c_r}{r}\left[\sum_{n=1}^{N-1} v_n(t)\right]^r \\ &= \frac{d}{dt}\sum_{n=2}^{N}\left(\sum_{r=2}^{n}\frac{c_r}{r} i_{n,r}(t)\right) \end{aligned} \tag{5.32}$$

Table 5.5: For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at f_1 up to the fifth order.

n	r	$\{[n_1 \dots n_r]\}$	$2g_r I_{n,r}(f_1)$
2	2		
3	2	[1 2]	$g_2\big(V_{1,1,0}V_{2,0,0} + V_{1,0,1}V^*_{2,-1,1} + V^*_{1,1,0}V_{2,2,0} + V^*_{1,0,1}V_{2,1,1}\big)$
	3	[1 1 1]	$g_3\frac{3}{4}\big(V_{1,1,0}V_{1,1,0}V^*_{1,1,0} + 2V_{1,1,0}V_{1,0,1}V^*_{1,0,1}\big)$
4	2		
	3		
	4		
5	2	[1 4] [2 3]	$g_2\big([V_{1,1,0}V_{4,0,0} + V_{1,0,1}V^*_{4,-1,1} + V^*_{1,1,0}V_{4,2,0} + V^*_{1,0,1}V_{4,1,1}]+$ $[V_{2,0,0}V_{3,1,0} + V_{2,-1,1}V_{3,2,-1} + V_{2,2,0}V^*_{3,1,0} + V_{2,1,1}V^*_{3,0,1} + V_{2,0,2}V^*_{3,-1,2}$ $+V^*_{2,-1,1}V_{3,0,1} + V^*_{2,2,0}V_{3,3,0} + V^*_{2,1,1}V_{3,2,1} + V^*_{2,0,2}V_{3,1,2}]\big)$
	3	[1 1 3] [1 2 2]	$g_3\frac{3}{4}\big([V_{1,1,0}V_{1,1,0}V^*_{3,1,0} + 2V_{1,1,0}V_{1,0,1}V^*_{3,0,1} + 2V_{1,1,0}V^*_{1,1,0}V_{3,1,0}$ $+2V_{1,1,0}V^*_{1,0,1}V_{3,0,1} + V_{1,0,1}V_{1,0,1}V^*_{3,-1,2} + 2V_{1,0,1}V^*_{1,1,0}V_{3,2,-1}$ $+2V_{1,0,1}V^*_{1,0,1}V_{3,1,0} + V^*_{1,1,0}V^*_{1,1,0}V_{3,3,0} + 2V^*_{1,1,0}V^*_{1,0,1}V_{3,2,1}$ $+V^*_{1,0,1}V^*_{1,0,1}V_{3,1,2}] + [V_{1,1,0}V_{2,0,0}V_{2,0,0} + 2V_{1,1,0}V_{2,-1,1}V^*_{2,-1,1}$ $+2V_{1,1,0}V_{2,2,0}V^*_{2,2,0} + 2V_{1,1,0}V_{2,1,1}V^*_{2,1,1} + 2V_{1,1,0}V_{2,0,2}V^*_{2,0,2}$ $+2V_{1,0,1}V_{2,0,0}V^*_{2,-1,1} + 2V_{1,0,1}V_{2,2,0}V^*_{2,1,1} + 2V_{1,0,1}V_{2,1,1}V^*_{2,0,2}$ $+2V^*_{1,1,0}V_{2,0,0}V_{2,2,0} + 2V^*_{1,1,0}V_{2,1,1}V^*_{2,-1,1} + 2V^*_{1,0,1}V_{2,0,0}V_{2,1,1}$ $+2V^*_{1,0,1}V_{2,-1,1}V_{2,2,0} + 2V^*_{1,0,1}V_{2,0,2}V^*_{2,-1,1}]\big)$
	4	[1 1 1 2]	$g_4\frac{1}{2}\big(V_{1,1,0}V_{1,1,0}V_{1,1,0}V^*_{2,2,0} + 3V_{1,1,0}V_{1,1,0}V_{1,0,1}V^*_{2,1,1}$ $+3V_{1,1,0}V_{1,1,0}V^*_{1,1,0}V_{2,0,0} + 3V_{1,1,0}V_{1,1,0}V^*_{1,0,1}V_{2,-1,1}$ $+3V_{1,1,0}V_{1,0,1}V_{1,0,1}V^*_{2,0,2} + 6V_{1,1,0}V_{1,0,1}V^*_{1,1,0}V^*_{2,-1,1}$ $+6V_{1,1,0}V_{1,0,1}V^*_{1,0,1}V_{2,0,0} + 3V_{1,1,0}V^*_{1,1,0}V^*_{1,1,0}V_{2,2,0}$ $+6V_{1,1,0}V^*_{1,1,0}V^*_{1,0,1}V_{2,1,1} + 3V_{1,1,0}V^*_{1,0,1}V^*_{1,0,1}V_{2,0,2}$ $+3V_{1,0,1}V_{1,0,1}V^*_{1,0,1}V^*_{2,-1,1} + 6V_{1,0,1}V^*_{1,1,0}V^*_{1,0,1}V_{2,2,0}$ $+3V_{1,0,1}V^*_{1,0,1}V^*_{1,0,1}V_{2,1,1}\big)$
	5	[1 1 1 1 1]	$g_5\frac{5}{8}\big(V_{1,1,0}V_{1,1,0}V_{1,1,0}V^*_{1,1,0}V^*_{1,1,0}$ $+6V_{1,1,0}V_{1,1,0}V_{1,0,1}V^*_{1,1,0}V^*_{1,0,1} + 3V_{1,1,0}V_{1,0,1}V_{1,0,1}V^*_{1,0,1}V^*_{1,0,1}\big)$

Table 5.6: For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1 - f_2$ up to the fifth order.

n	r	$\{[n_1 \dots n_r]\}$	$2g_r I_{n,r}(2f_1 - f2)$
2	2		
3	2	[1 2]	$g_2\big(V_{1,1,0}V^*_{2,-1,1} + V^*_{1,0,1}V_{2,2,0}\big)$
	3	[1 1 1]	$g_3\frac{3}{4}\big(V_{1,1,0}V_{1,1,0}V^*_{1,0,1}\big)$
4	2		
	3		
	4		
5	2	[1 4] [2 3]	$g_2\big([V_{1,1,0}V^*_{4,-1,1} + V_{1,0,1}V^*_{4,-2,2} + V^*_{1,1,0}V_{4,3,-1} + V^*_{1,0,1}V_{4,2,0}]$ $+[V_{2,0,0}V_{3,2,-1} + V_{2,2,0}V^*_{3,0,1} + V_{2,1,1}V^*_{3,-1,2}$ $+V^*_{2,-1,1}V_{3,1,0} + V^*_{2,1,1}V_{3,3,0} + V^*_{2,0,2}V_{3,2,1}]\big)$
	3	[1 1 3] [1 2 2]	$g_3\frac{3}{4}\big([V_{1,1,0}V_{1,1,0}V^*_{3,0,1} + 2V_{1,1,0}V_{1,0,1}V^*_{3,-1,2} + 2V_{1,1,0}V^*_{1,1,0}V_{3,2,-1}$ $+2V_{1,1,0}V^*_{1,0,1}V_{3,1,0} + 2V_{1,0,1}V^*_{1,0,1}V_{3,2,-1} + 2V^*_{1,1,0}V^*_{1,0,1}V_{3,3,0}$ $+V^*_{1,0,1}V^*_{1,0,1}V_{3,2,1}] + [2V_{1,1,0}V_{2,0,0}V^*_{2,-1,1} + 2V_{1,1,0}V_{2,2,0}V^*_{2,1,1}$ $+2V_{1,1,0}V_{2,1,1}V^*_{2,0,2} + 2V_{1,0,1}V_{2,2,0}V^*_{2,0,2} + V_{1,0,1}V^*_{2,-1,1}V^*_{2,-1,1}$ $+2V^*_{1,1,0}V_{2,2,0}V^*_{2,-1,1} + 2V^*_{1,0,1}V_{2,0,0}V_{2,2,0} + 2V^*_{1,0,1}V_{2,1,1}V^*_{2,-1,1}]\big)$
	4	[1 1 1 2]	$g_4\frac{1}{2}\big(V_{1,1,0}V_{1,1,0}V_{1,1,0}V^*_{2,1,1} + 3V_{1,1,0}V_{1,1,0}V_{1,0,1}V^*_{2,0,2}$ $+3V_{1,1,0}V_{1,1,0}V^*_{1,1,0}V^*_{2,-1,1} + 3V_{1,1,0}V_{1,1,0}V^*_{1,0,1}V_{2,0,0}$ $+6V_{1,1,0}V_{1,0,1}V^*_{1,0,1}V^*_{2,-1,1} + 6V_{1,1,0}V^*_{1,1,0}V^*_{1,0,1}V_{2,2,0}$ $+3V_{1,1,0}V^*_{1,0,1}V^*_{1,0,1}V_{2,1,1} + 3V_{1,0,1}V^*_{1,0,1}V^*_{1,0,1}V_{2,2,0}\big)$
	5	[1 1 1 1 1]	$g_5\frac{5}{8}\big(2V_{1,1,0}V_{1,1,0}V_{1,1,0}V^*_{1,1,0}V^*_{1,0,1} + 3V_{1,1,0}V_{1,1,0}V_{1,0,1}V^*_{1,0,1}V^*_{1,0,1}\big)$

Table 5.7: For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1 + f_2$ up to the fifth order.

n	r	$\{[n_1 \dots n_r]\}$	$2g_r I_{n,r}(2f_1 + f_2)$
2	2		
3	2	[1 2]	$g_2\big(V_{1,1,0}V_{2,1,1} + V_{1,0,1}V_{2,2,0}\big)$
	3	[1 1 1]	$g_3\frac{3}{4}\big(V_{1,1,0}V_{1,1,0}V_{1,0,1}\big)$
4	2		
	3		
	4		
5	2	[1 4] [2 3]	$g_2\big([V_{1,1,0}V_{4,1,1} + V_{1,0,1}V_{4,2,0} + V^*_{1,1,0}V_{4,3,1} + V^*_{1,0,1}V_{4,2,2}]$ $+[V_{2,0,0}V_{3,2,1} + V_{2,-1,1}V_{3,3,0} + V_{2,2,0}V_{3,0,1}$ $+V_{2,1,1}V_{3,1,0} + V_{2,0,2}V_{3,2,-1} + V^*_{2,-1,1}V_{3,1,2}]\big)$
	3	[1 1 3] [1 2 2]	$g_3\frac{3}{4}\big([V_{1,1,0}V_{1,1,0}V_{3,0,1} + 2V_{1,1,0}V_{1,0,1}V_{3,1,0} + 2V_{1,1,0}V^*_{1,1,0}V_{3,2,1}$ $+2V_{1,1,0}V^*_{1,0,1}V_{3,1,2} + V_{1,0,1}V_{1,0,1}V_{3,2,-1} + 2V_{1,0,1}V^*_{1,1,0}V_{3,3,0}$ $+2V_{1,0,1}V^*_{1,0,1}V_{3,2,1}] + [2V_{1,1,0}V_{2,0,0}V_{2,1,1} + 2V_{1,1,0}V_{2,-1,1}V_{2,2,0}$ $+2V_{1,1,0}V_{2,0,2}V^*_{2,-1,1} + 2V_{1,0,1}V_{2,0,0}V_{2,2,0} + 2V_{1,0,1}V_{2,1,1}V^*_{2,-1,1}$ $+2V^*_{1,1,0}V_{2,2,0}V_{2,1,1} + 2V^*_{1,0,1}V_{2,2,0}V_{2,0,2} + V^*_{1,0,1}V_{2,1,1}V_{2,1,1}]\big)$
	4	[1 1 1 2]	$g_4\frac{1}{2}\big(V_{1,1,0}V_{1,1,0}V_{1,1,0}V_{2,-1,1} + 3V_{1,1,0}V_{1,1,0}V_{1,0,1}V_{2,0,0}$ $+3V_{1,1,0}V_{1,1,0}V^*_{1,1,0}V_{2,1,1} + 3V_{1,1,0}V_{1,1,0}V^*_{1,0,1}V_{2,0,2}$ $+3V_{1,1,0}V_{1,0,1}V_{1,0,1}V^*_{2,-1,1} + 6V_{1,1,0}V_{1,0,1}V^*_{1,1,0}V_{2,2,0}$ $+6V_{1,1,0}V_{1,0,1}V^*_{1,0,1}V_{2,1,1} + 3V_{1,0,1}V_{1,0,1}V^*_{1,0,1}V_{2,2,0}\big)$
	5	[1 1 1 1 1]	$g_5\frac{5}{8}\big(2V_{1,1,0}V_{1,1,0}V_{1,1,0}V_{1,0,1}V^*_{1,1,0} + 3V_{1,1,0}V_{1,1,0}V_{1,0,1}V_{1,0,1}V^*_{1,0,1}\big)$

Table 5.8: For the zero-memory polynomial nonlinearity, nonlinear current sources for calculating the 2-tone response at $3f_2$ up to the fifth order.

n	r	$\{[n_1 \dots n_r]\}$	$2g_r I_{n,r}(3f_2)$
2	2		
3	2	[1 2]	$g_2\big(V_{1,0,1}V_{2,0,2}\big)$
	3	[1 1 1]	$g_3\frac{1}{4}\big(V_{1,0,1}V_{1,0,1}V_{1,0,1}\big)$
4	2		
	3		
	4		
5	2	[1 4] [2 3]	$g_2\big([V_{1,1,0}V_{4,-1,3} + V_{1,0,1}V_{4,0,2} + V_{1,1,0}^*V_{4,1,3} + V_{1,0,1}^*V_{4,0,4}]$ $+[V_{2,0,0}V_{3,0,3} + V_{2,-1,1}V_{3,1,2} + V_{2,1,1}V_{3,-1,2} + V_{2,0,2}V_{3,0,1}]\big)$
	3	[1 1 3] [1 2 2]	$g_3\frac{3}{4}\big([2V_{1,1,0}V_{1,0,1}V_{3,-1,2} + 2V_{1,1,0}V_{1,1,0}^*V_{3,0,3}$ $+V_{1,0,1}V_{1,0,1}V_{3,0,1} + 2V_{1,0,1}V_{1,1,0}^*V_{3,1,2} + 2V_{1,0,1}V_{1,0,1}^*V_{3,0,3}]$ $+[2V_{1,1,0}V_{2,-1,1}V_{2,0,2} + 2V_{1,0,1}V_{2,0,0}V_{2,0,2} + 2V_{1,0,1}V_{2,-1,1}V_{2,1,1}$ $+2V_{1,1,0}^*V_{2,1,1}V_{2,0,2} + V_{1,0,1}^*V_{2,0,2}V_{2,0,2}]\big)$
	4	[1 1 1 2]	$g_4\frac{1}{2}\big(3V_{1,1,0}V_{1,0,1}V_{1,0,1}V_{2,-1,1} + 6V_{1,1,0}V_{1,0,1}V_{1,1,0}^*V_{2,0,2}$ $+V_{1,0,1}V_{1,0,1}V_{1,0,1}V_{2,0,0} + 3V_{1,0,1}V_{1,0,1}V_{1,1,0}^*V_{2,1,1}$ $+3V_{1,0,1}V_{1,0,1}V_{1,0,1}^*V_{2,0,2}\big)$
	5	[1 1 1 1 1]	$g_5\frac{5}{16}\big(4V_{1,1,0}V_{1,0,1}V_{1,0,1}V_{1,0,1}V_{1,1,0}^* + V_{1,0,1}V_{1,0,1}V_{1,0,1}V_{1,0,1}V_{1,0,1}^*\big)$

where the second equation comes from breaking down $i_{\mathrm{NL}}(t)$ into its nth order and rth degree contributions and has the same form as (5.17). Now $\frac{d}{dt}$ can be brought back into the double sum and applied on $i_{n,r}(t)$, so (5.29) becomes

$$I_{n,r}(f_{\mathrm{IM}_n}) = j2\pi f_{\mathrm{IM}_n} \sum_{\{[n_1 \dots n_r]\}} C\frac{1}{2}^r \left[\sum_{\{[f_{\mathrm{IM}_{n_1}} \cdots f_{\mathrm{IM}_{n_r}}]\}} V_{n_1}(f_{\mathrm{IM}_{n_1}}) \dots V_{n_r}(f_{\mathrm{IM}_{n_r}}) \right]. \qquad (5.33)$$

Thus the only difference from the zero-memory polynomial nonlinearity case is the $j2\pi f_{\mathrm{IM}_n}$ factor, so both nonlinearities can share the same data structure for use during the recursive circuit analysis.

Inductive Nonlinearity

For an inductive nonlinearity, the nonlinear current accounting for contributions up to order N is

$$i_{\mathrm{NL}}(t) \;=\; \sum_{r=2}^{N} \gamma_r \left[\int_{-\infty}^{t} \sum_{n=1}^{N-1} v_n(t) dt \right]^r \tag{5.34}$$

where the Nth order contribution can be generated from degree $r = N$ or $r = 2$.

As the zero-memory polynomial case in (5.17), $i_{\mathrm{NL}}(t)$ can be broken down into its nth order and rth degree contributions

$$i_{\mathrm{NL}}(t) \;=\; \sum_{n=2}^{N} \left(\sum_{r=2}^{n} \gamma_r i_{n,r}(t) \right). \tag{5.35}$$

Each $i_{n,r}(t)$ term has the form

$$i_{n,r}(t) \;=\; \sum_{\{[n_1 \ldots n_r]\}} C \left[\int_{-\infty}^{t} v_{n_1}(t) dt \right] \ldots \left[\int_{-\infty}^{t} v_{n_r}(t) dt \right] \tag{5.36}$$

which is unlike the capacitive nonlinearity case where the $i_{n,r}(t)$ term only differs from the zero-memory case (5.18) by a $\frac{d}{dt}$ factor in front.

Substituting the assumed solution (5.4) into (5.36), a particular spectral component in $i_{n,r}(t)$ has the form

$$I_{n,r}(f_{\mathrm{IM}_n}) = \sum_{\{[n_1 \ldots n_r]\}} C \frac{1}{2}^r \left[\sum_{\{[f_{\mathrm{IM}_{n_1}} \ldots f_{\mathrm{IM}_{n_r}}]\}} \frac{V_{n_1}(f_{\mathrm{IM}_{n_1}})}{j2\pi f_{\mathrm{IM}_{n_1}}} \ldots \frac{V_{n_r}(f_{\mathrm{IM}_{n_r}})}{j2\pi f_{\mathrm{IM}_{n_r}}} \right]. \tag{5.37}$$

Despite the different analytical forms of (5.29), (5.33) and (5.37), the dependence information required to reconstruct the nonlinear current during the recursive circuit analysis is the same, namely all f_{IM} generated in $i_{n,r}$ and the contributing $[n_1 \ldots n_r]$ and $[f_{\mathrm{IM}_{n_1}} \ldots f_{\mathrm{IM}_{n_r}}]$. So the zero-memory, capacitive and inductive polynomial nonlinearities can share the same data structure.

Multiplier Nonlinearity

For the behavioral multiplier with two inputs v and u, the nonlinear current is

$$i_{\mathrm{NL}}(t) \;=\; g_2\left[\sum_{n_v=1}^{N-1} v_n(t)\right]\left[\sum_{n_u=1}^{N-1} u_n(t)\right] \tag{5.38}$$

where the Nth order contribution can be generated from this second degree nonlinearity as $v_{N-1}u_1$ or v_1u_{N-1}.

Analogous to (5.17) and (5.18), $i_{\mathrm{NL}}(t)$ can also be broken down into its nth order contributions

$$i_{\mathrm{NL}}(t) \;=\; \sum_{n=2}^{N}\big(g_2 i_{n,2}(t)\big) \tag{5.39}$$

where $i_{n,2}(t)$ is the nth order component generated by this second degree nonlinearity

$$i_{n,2}(t) \;=\; \sum_{\{[n_v\ n_u]\}}\big(v_{n_v}(t)u_{n_u}(t)\ \text{s.t}\ (n_v+n_u=n)\big). \tag{5.40}$$

For a Q-tone input, all possible response frequencies f_{IM} of all orders are found as in (5.26). Using the assumed solution (5.4), each $i_{n,2}$ term (5.40) in (5.39) is

$$i_{n,2}(t) = \sum_{\{[n_v\ n_u]\}}\left[\frac{1}{2}\sum_{\{f_{\mathrm{IM}_{n_v}}\}} V_{n_v}(f_{\mathrm{IM}_{n_v}})e^{j2\pi f_{\mathrm{IM}_{n_v}}t}\right]\left[\frac{1}{2}\sum_{\{f_{\mathrm{IM}_{n_u}}\}} U_{n_u}(f_{\mathrm{IM}_{n_u}})e^{j2\pi f_{\mathrm{IM}_{n_u}}t}\right] \tag{5.41}$$

which contains many spectral components of the form

$$I_{n,2}(f_{\mathrm{IM}_n}) = \sum_{\{[n_v\ n_u]\}}\frac{1}{2}^2\left[\sum_{\{[f_{\mathrm{IM}_{n_v}}\ f_{\mathrm{IM}_{n_u}}]\}} V_{n_v}(f_{\mathrm{IM}_{n_v}})U_{n_u}(f_{\mathrm{IM}_{n_u}})\right]. \tag{5.42}$$

To reconstruct the nonlinear current during the recursive circuit analysis, all f_{IM} generated in i_n and the contributing $[n_v\ n_u]$ and $[f_{\mathrm{IM}_{n_v}}\ f_{\mathrm{IM}_{n_u}}]$ are stored in another data structure separate

from the polynomial nonlinearity case.

As an example for a 2-tone input at frequencies f_1 and f_2, the nonlinear current expressions needed to evaluate the responses up to the fifth order are given in Table 5.9 for $f_2 - f_1$, Table 5.10 for $f_1 + f_2$, Table 5.11 for $2f_1$, Table 5.12 for f_1, Table 5.13 for $2f_1 - f_2$, Table 5.14 for $2f_1 + f_2$, and Table 5.15 for $3f_2$.

In [6] for the original direct calculation method, the multiplier nonlinearity was not discussed as a part of transistor device modeling. The behavioral multiplier nonlinearity is included here because it is an essential part of any RF front-end behavioral model. In conformance with [6], the nonlinear current expression given in the tables is the right-hand side (5.30). Here the nth order response at each $f_{\mathrm{IM}_n} = k_1 f_1 + k_2 f_2$ is denoted using a short hand notation as V_{n,k_1,k_2}.

Table 5.9: For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $f_2 - f_1$ up to the fifth order.

n	$\{[n_v\ n_u]\}$	$2g_2 I_{n,2}(f_2 - f_1)$
2	[1 1]	$g_2 \frac{1}{2}\big(V_{1,0,1}U^*_{1,1,0} + V^*_{1,1,0}U_{1,0,1}\big)$
3		
4	[3 1] [2 2] [1 3]	$g_2 \frac{1}{2}\big([V_{3,0,1}U^*_{1,1,0} + V_{3,-1,2}U^*_{1,0,1} + V^*_{3,2,-1}U_{1,1,0} + V^*_{3,1,0}U_{1,0,1}]$ $+[V_{2,0,0}U_{2,-1,1} + V_{2,-1,1}U_{2,0,0} + V_{2,1,1}U^*_{2,2,0}$ $+V_{2,0,2}U^*_{2,1,1} + V^*_{2,2,0}U_{2,1,1} + V^*_{2,1,1}U_{2,0,2}]$ $+[V_{1,1,0}U^*_{3,2,-1} + V_{1,0,1}U^*_{3,1,0} + V^*_{1,1,0}U_{3,0,1} + V^*_{1,0,1}U_{3,-1,2}]\big)$
5		

Table 5.10: For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $f_1 + f_2$ up to the fifth order.

n	$\{[n_v\ n_u]\}$	$2g_2 I_{n,2}(f_1+f_2)$
2	[1 1]	$g_2\frac{1}{2}\big(V_{1,1,0}U_{1,0,1}+V_{1,0,1}U_{1,1,0}\big)$
3		
4	[3 1] [2 2] [1 3]	$g_2\frac{1}{2}\big([V_{3,1,0}U_{1,0,1}+V_{3,0,1}U_{1,1,0}+V_{3,2,1}U^*_{1,1,0}+V_{3,1,2}U^*_{1,0,1}]$ $+[V_{2,0,0}U_{2,1,1}+V_{2,-1,1}U_{2,2,0}+V_{2,2,0}U_{2,-1,1}$ $+V_{2,1,1}U_{2,0,0}+V_{2,0,2}U^*_{2,-1,1}+V^*_{2,-1,1}U_{2,0,2}]$ $+[V_{1,1,0}U_{3,0,1}+V_{1,0,1}U_{3,1,0}+V^*_{1,1,0}U_{3,2,1}+V^*_{1,0,1}U_{3,1,2}]\big)$
5		

Table 5.11: For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1$ up to the fifth order.

n	$\{[n_v\ n_u]\}$	$2g_2 I_{n,r}(2f_1)$
2	[1 1]	$g_2\frac{1}{2}\big(V_{1,1,0}U_{1,1,0}\big)$
3		
3	[3 1] [2 2] [1 3]	$g_2\frac{1}{2}\big([V_{3,2,-1}U_{1,0,1}+V_{3,1,0}U_{1,1,0}+V_{3,3,0}U^*_{1,1,0}+V_{3,2,1}U^*_{1,0,1}]$ $+[V_{2,0,0}U_{2,2,0}+V_{2,2,0}U_{2,0,0}+V_{2,1,1}U^*_{2,-1,1}+V^*_{2,-1,1}U_{2,1,1}]$ $+[V_{1,1,0}U_{3,1,0}+V_{1,0,1}U_{3,2,-1}+V^*_{1,1,0}U_{3,3,0}+V^*_{1,0,1}U_{3,2,1}]\big)$
5		

Table 5.12: For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at f_1 up to the fifth order.

n	$\{[n_v\ n_u]\}$	$2g_2 I_{n,r}(f_1)$
2		
3	[2 1] [1 2]	$g_2\frac{1}{2}\big([V_{2,0,0}U_{1,1,0}+V_{2,2,0}U^*_{1,1,0}+V_{2,1,1}U^*_{1,0,1}+V^*_{2,-1,1}U_{1,0,1}]$ $+[V_{1,1,0}U_{2,0,0}+V_{1,0,1}U^*_{2,-1,1}+V^*_{1,1,0}U_{2,2,0}+V^*_{1,0,1}U_{2,1,1}]\big)$
4		
5	[4 1] [3 2] [2 3] [1 4]	$g_2\frac{1}{2}\big([V_{4,0,0}U_{1,1,0}+V_{4,2,0}U^*_{1,1,0}+V_{4,1,1}U^*_{1,0,1}+V^*_{4,-1,1}U_{1,0,1}]$ $+[V_{3,2,-1}U_{2,-1,1}+V_{3,1,0}U_{2,0,0}+V_{3,0,1}U^*_{2,-1,1}+V_{3,3,0}U^*_{2,2,0}+V_{3,2,1}U^*_{2,1,1}$ $+V_{3,1,2}U^*_{2,0,2}+V^*_{3,1,0}U_{2,2,0}+V^*_{3,0,1}U_{2,1,1}+V^*_{3,-1,2}U_{2,0,2}]$ $+[V_{2,0,0}U_{3,1,0}+V_{2,-1,1}U_{3,2,-1}+V_{2,2,0}U^*_{3,1,0}+V_{2,1,1}U^*_{3,0,1}+V_{2,0,2}U^*_{3,-1,2}$ $+V^*_{2,-1,1}U_{3,0,1}+V^*_{2,2,0}U_{3,3,0}+V^*_{2,1,1}U_{3,2,1}+V^*_{2,0,2}U_{3,1,2}]$ $+[V_{1,1,0}U_{4,0,0}+V_{1,0,1}U^*_{4,-1,1}+V^*_{1,1,0}U_{4,2,0}+V^*_{1,0,1}U_{4,1,1}]\big)$

Table 5.13: For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1 - f_2$ up to the fifth order.

n	$\{[n_v\ n_u]\}$	$2g_2 I_{n,r}(2f_1 - f_2)$
2		
3	[2 1] [1 2]	$g_2\frac{1}{2}\big([V_{2,2,0}U^*_{1,0,1} + V^*_{2,-1,1}U_{1,1,0}]$ $+[V_{1,1,0}U^*_{2,-1,1} + V^*_{1,0,1}U_{2,2,0}]\big)$
4		
5	[4 1] [3 2] [2 3] [1 4]	$g_2\frac{1}{2}\big([V_{4,3,-1}U^*_{1,1,0} + V_{4,2,0}U^*_{1,0,1} + V^*_{4,-1,1}U_{1,1,0} + V^*_{4,-2,2}U_{1,0,1}]$ $+[V_{3,2,-1}U_{2,0,0} + V_{3,1,0}U^*_{2,-1,1} + V_{3,3,0}U^*_{2,1,1} + V_{3,2,1}U^*_{2,0,2}$ $+V^*_{3,0,1}U_{2,2,0} + V^*_{3,-1,2}U_{2,1,1}] + [V_{2,0,0}U_{3,2,-1} + V_{2,2,0}U^*_{3,0,1}$ $+V_{2,1,1}U^*_{3,-1,2} + V^*_{2,-1,1}U_{3,1,0} + V^*_{2,1,1}U_{3,3,0} + V^*_{2,0,2}U_{3,2,1}]$ $+[V_{1,1,0}U^*_{4,-1,1} + V_{1,0,1}U^*_{4,-2,2} + V^*_{1,1,0}U_{4,3,-1} + V^*_{1,0,1}U_{4,2,0}]\big)$

Table 5.14: For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $2f_1 + f_2$ up to the fifth order.

n	$\{[n_v\ n_u]\}$	$2g_2 I_{n,r}(2f_1 + f_2)$
2		
3	[2 1] [1 2]	$g_2\frac{1}{2}\big([V_{2,2,0}U_{1,0,1} + V_{2,1,1}U_{1,1,0}]$ $+[V_{1,1,0}U_{2,1,1} + V_{1,0,1}U_{2,2,0}]\big)$
4		
5	[4 1] [3 2] [2 3] [1 4]	$g_2\frac{1}{2}\big([V_{4,2,0}U_{1,0,1} + V_{4,1,1}U_{1,1,0} + V_{4,3,1}U^*_{1,1,0} + V_{4,2,2}U^*_{1,0,1}]$ $+[V_{3,2,-1}U_{2,0,2} + V_{3,1,0}U_{2,1,1} + V_{3,0,1}U_{2,2,0} + V_{3,3,0}U_{2,-1,1}+$ $V_{3,2,1}U_{2,0,0} + V_{3,1,2}U^*_{2,-1,1}] + [V_{2,0,0}U_{3,2,1} + V_{2,-1,1}U_{3,3,0}$ $+V_{2,2,0}U_{3,0,1} + V_{2,1,1}U_{3,1,0} + V_{2,0,2}U_{3,2,-1} + V^*_{2,-1,1}U_{3,1,2}]$ $+[V_{1,1,0}U_{4,1,1} + V_{1,0,1}U_{4,2,0} + V^*_{1,1,0}U_{4,3,1} + V^*_{1,0,1}U_{4,2,2}]\big)$

Table 5.15: For the zero-memory behavioral multiplier nonlinearity, nonlinear current sources for calculating the 2-tone response at $3f_2$ up to the fifth order.

n	$\{[n_v\ n_u]\}$	$2g_2 I_{n,r}(3f_2)$
2		
3	[2 1] [1 2]	$g_2\frac{1}{2}\big([V_{2,0,2}U_{1,0,1}]+[V_{1,0,1}U_{2,0,2}]\big)$
4		
5	[4 1] [3 2] [2 3] [1 4]	$g_2\frac{1}{2}\big([V_{4,0,2}U_{1,0,1}+V_{4,-1,3}U_{1,1,0}+V_{4,1,3}U^*_{1,1,0}+V_{4,0,4}U^*_{1,0,1}]$ $+[V_{3,0,1}U_{2,0,2}+V_{3,-1,2}U_{2,1,1}+V_{3,1,2}U_{2,-1,1}+V_{3,0,3}U_{2,0,0}]$ $+[V_{2,0,0}U_{3,0,3}+V_{2,-1,1}U_{3,1,2}+V_{2,1,1}U_{3,-1,2}+V_{2,0,2}U_{3,0,1}]$ $+[V_{1,1,0}U_{4,-1,3}+V_{1,0,1}U_{4,0,2}+V^*_{1,1,0}U_{4,1,3}+V^*_{1,0,1}U_{4,0,4}]\big)$

5.2.3 Circuit Analysis

In the NLTF method of Volterra circuit analysis, the NLTFs are determined non-recursively starting with the lowest order. In the direct calculation method, the recursive nature of the circuit analysis was pointed out in the Section 5.2.2, which outlined a procedure for finding the nonlinear current expressions for several circuit and behavioral nonlinearities.

Here the recursive circuit analysis algorithm presented in Fig. 5.2 makes use of these nonlinear current expressions which are stored in a look-up data structure. The required user inputs are the number of tones in the multi-tone excitation, the set of IM response frequencies $\{f_{\rm IM}\}$ to be computed for all the node voltages, and the order N which sets the highest order contribution to be taken into account at each $f_{\rm IM}$. The nonlinear current data structure is then generated, which contains the dependence information required to reconstruct nonlinear current expressions. At each $f_{\rm IM}$, the algorithm then proceeds to compute all the nth order contribution to the nodal responses which is denoted as $f_{{\rm IM}_n}$. The circuit response data structure stores the results computed by the algorithm.

In the recursive function which computes the nodal responses at each $f_{{\rm IM}_n}$, a test checks whether the terminal conditions is reached where the linear response is recalled from a circuit response data structure. If the response at $f_{{\rm IM}_n}$ needs to be computed, the associated nonlinear

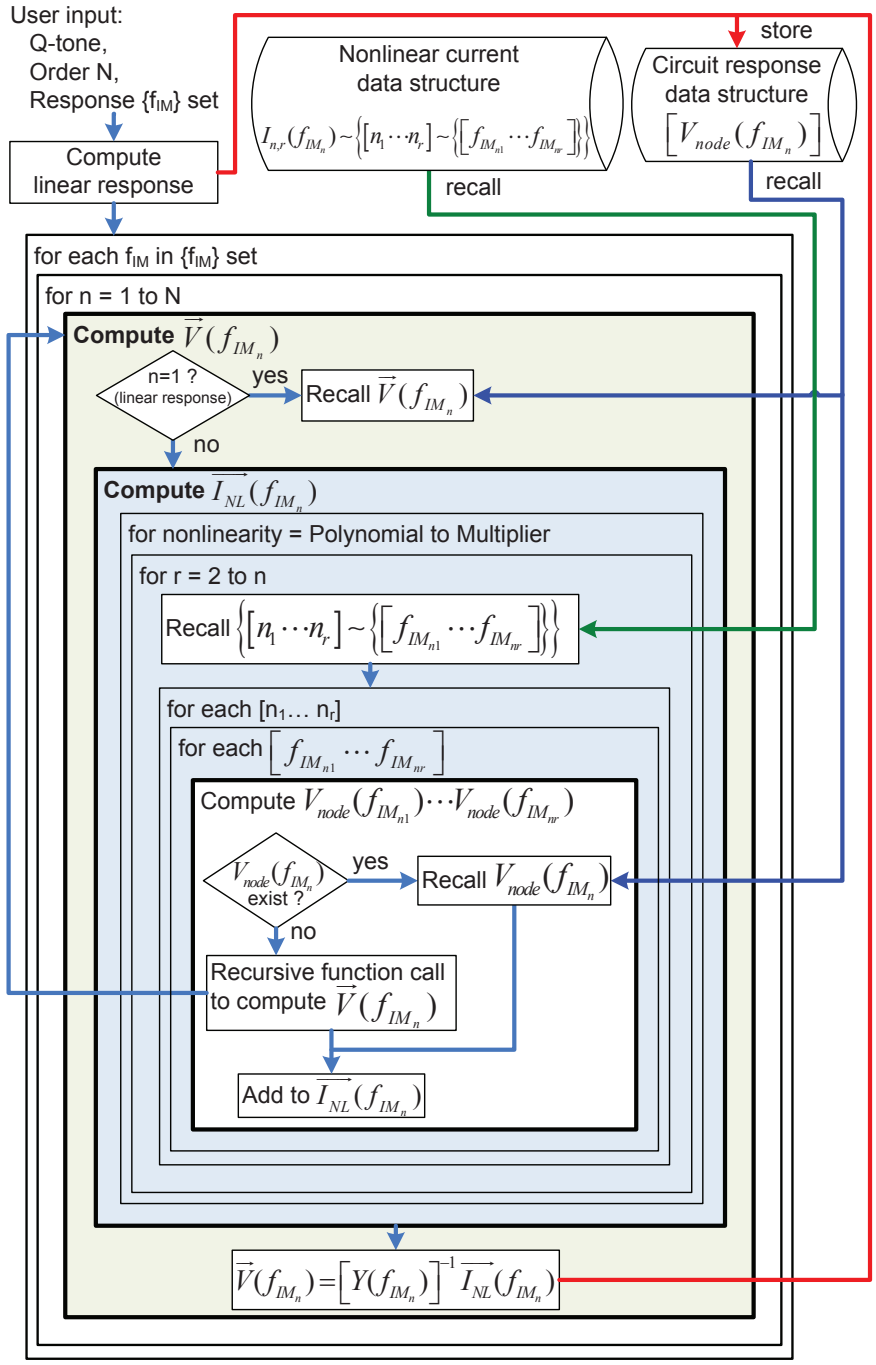


Figure 5.2: Algorithm flow chart for the proposed direct calculation method of Volterra circuit analysis.

current vector will be computed. For each type of nonlinearity sources, the contribution from each degree of nonlinearity is computed following (5.29) or (5.33) or (5.42), using dependence information recalled from the nonlinear current data structure. For each term in the product, a test checks whether the required response at $f_{\mathrm{IM}_{nr}}$ has already been computed, and whether to recall it from a circuit response data structure or to issue a call to the recursive function which will computes the required response at $f_{\mathrm{IM}_{nr}}$. Once the nonlinear current vector is computed, the nodal responses at each f_{IM_n} is found from a matrix solution using the inverse nodal admittance matrix of the linearized circuit, and stored in the circuit response data structure.

5.3 Circuit Analysis Examples

In this chapter, the symbolic Volterra circuit simulator developed that implements the improved direct calculation method is applied to analyzing some example nonlinear circuits with multiple inputs. In Section 5.3.1, a transistor level single-balanced mixer circuit under a 2-tone input is analyzed. This example is chosen to verify and differentiate from analysis results presented in [6] using the original direct calculation method. In Section 5.3.2, a system-level behavioral model of a RF receiver front-end under a 2-tone input is analyzed.

5.3.1 Differential Pair as Single-Balanced Mixer

The differential pair circuit is shown in Fig. 5.3. To use the differential pair as a single balanced mixer, one signal is applied at the differential input and another signal is applied at the current source. For compatibility with Volterra circuit analysis, the signal applied at each input is represented as a multi-tone signal. To simplify the analysis, each signal is represented as a sinusoid.

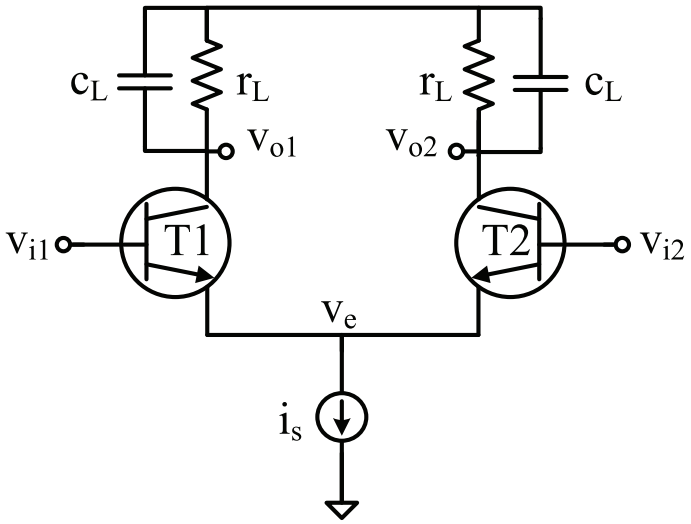


Figure 5.3: Circuit schematic of a bipolar differential pair.

Equivalent Circuit Model

Here the analysis focus on the weakly nonlinear operation, so the tone at the differential inputs is a small signal perturbation about the bias point, rather than steering the current completely into one branch or the other as in strongly nonlinear operation.

In conformance with [6], only the dominant drain or collector current nonlinearity in the transistor devices are modeled, using power series representations of the nonlinearity characteristics. The resulting equivalent circuit is shown in Fig. 5.4.

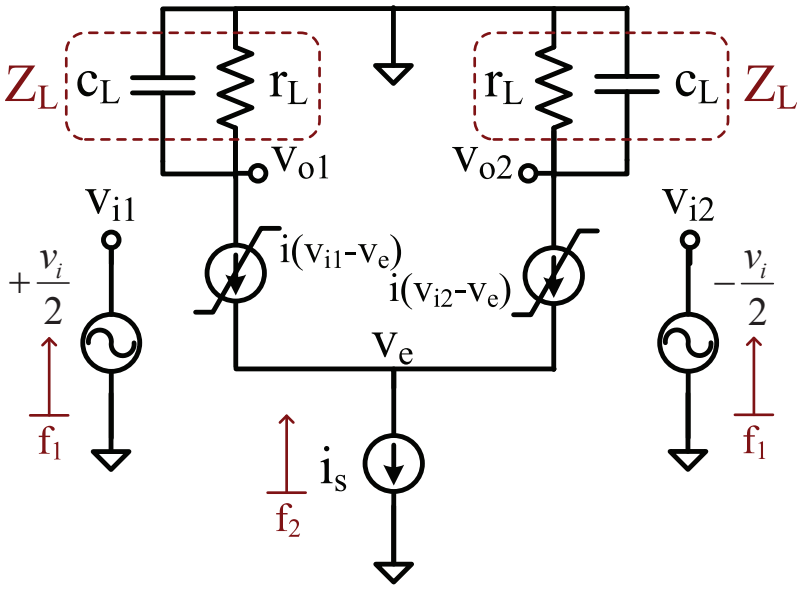


Figure 5.4: Equivalent circuit of a bipolar differential pair used as single-balanced mixer.

Circuit Analysis Using Direct Calculation Method

For nodal analysis, Kirchoff's current law yields the following set of circuit equations:

$$\left(g_{\mathrm{L}} + c_{\mathrm{L}}\frac{d}{dt}\right) v_{\mathrm{o1}}(t) + g_{\mathrm{T_1}}\left(-v_{\mathrm{e}}(t)\right) = -g_{\mathrm{T_1}}\left(\frac{v_{\mathrm{i}}(t)}{2}\right) - \sum_{r=2}^{N} g_{\mathrm{T}_r}\left(\frac{v_{\mathrm{i}}(t)}{2} - v_{\mathrm{e}}(t)\right)^r \tag{5.43}$$

$$\left(g_{\mathrm{L}} + c_{\mathrm{L}}\frac{d}{dt}\right) v_{\mathrm{o2}}(t) + g_{\mathrm{T_1}}\left(-v_{\mathrm{e}}(t)\right) = -g_{\mathrm{T_1}}\left(-\frac{v_{\mathrm{i}}(t)}{2}\right) - \sum_{r=2}^{N} g_{\mathrm{T}_r}\left(\frac{-v_{\mathrm{i}}(t)}{2} - v_{\mathrm{e}}(t)\right)^r \tag{5.44}$$

$$g_{\mathrm{T_1}}\left(-v_{\mathrm{e}}(t)\right) + g_{\mathrm{T_1}}\left(-v_{\mathrm{e}}(t)\right) = i_{\mathrm{s}}(t) - g_{\mathrm{T_1}}\left(\underbrace{\frac{v_{\mathrm{i}}(t)}{2}}\right) - g_{\mathrm{T_1}}\left(\underbrace{-\frac{v_{\mathrm{i}}(t)}{2}}\right) - \sum_{r=2}^{N} g_{\mathrm{T}_r}\left(\frac{v_{\mathrm{i}}(t)}{2} - v_{\mathrm{e}}(t)\right)^r - \sum_{r=2}^{N} g_{\mathrm{T}_r}\left(-\frac{v_{\mathrm{i}}(t)}{2} - v_{\mathrm{e}}(t)\right)^r \tag{5.45}$$

where the transistors T_1 and T_2 are assumed to be matched and can be modeled using the same power series coefficients g_{T_r}. The terms underlined with brace in the last equation are then

canceled.

In the resulting spectral balance equation (5.30) to be solved in the direct calculation method, the left-hand side representing the linearized circuit becomes

$$\mathbf{Y}(s)\overrightarrow{V}(s) = \begin{bmatrix} G_{\mathrm{L}}(s) & 0 & -g_{\mathrm{T}_1} \\ 0 & G_{\mathrm{L}}(s) & -g_{\mathrm{T}_1} \\ 0 & 0 & -2g_{\mathrm{T}_1} \end{bmatrix} \begin{bmatrix} V_{\mathrm{o}1}(s) \\ V_{\mathrm{o}2}(s) \\ V_{\mathrm{e}}(s) \end{bmatrix} \tag{5.46}$$

where the laplace variable s takes on $j2\pi f_{\mathrm{IM}}$ at each IM frequency, and $G_{\mathrm{L}}(s) = g_{\mathrm{L}} + sc_{\mathrm{L}}$. The right-hand side linear current source vector at the 2-tone input frequencies f_1 and f_2 are

$$2\overrightarrow{I_{\mathrm{LIN}}}(f_1) = 2\begin{bmatrix} -g_{\mathrm{T}_1}(\frac{A_1/2}{2}) \\ g_{\mathrm{T}_1}(\frac{A_1/2}{2}) \\ 0 \end{bmatrix},\ 2\overrightarrow{I_{\mathrm{LIN}}}(f_2) = 2\begin{bmatrix} 0 \\ 0 \\ A_2/2 \end{bmatrix} \tag{5.47}$$

where A_1 and A_2 are the complex amplitudes of the input tones. The nonlinear current source vector on the right-hand side is evaluated using the recursive direct calculation algorithm, as the response at the contributing IM frequencies are found for the controlling variables of the device nonlinearities. The required dependence information for this evaluation is store in the nonlinear current data structure as described in Section 5.2.2.

Analysis Results

The $V_{\mathrm{o}2} - V_{\mathrm{o}1}$ output responses up to the fifth order are given in Table 5.16 for $f_2 - f_1$, Table 5.17 for $f_1 + f_2$, Table 5.18 for f_1, Table 5.19 for $2f_2 - f_1$, Table 5.20 for $2f_2 + f_1$, and Table 5.21 for $3f_1$.

The $V_{\mathrm{o}2} - V_{\mathrm{o}1}$ output responses at $2f_1$, $2f_2$, f_2, $2f_1 - f_2$, $2f_1 + f_2$ and $3f_2$ are zero, at least for the perfectly matched circuit elements case here. The admittance $G_{\mathrm{L}}(s)$ at each IM frequency $f_{\mathrm{IM}} = kf_1 + lf_2$ is denote as $G_{k,l}$. Note that there are contributions of different order at each

f_{IM}, thus the compression effect is accounted for.

Table 5.16: The 2-tone response of a differential pair single-balanced mixer at $f_2 - f_1$ up to the fifth order.

n	$V_{\mathrm{o}2}(f_2 - f_1) - V_{\mathrm{o}1}(f_2 - f_1)$
1	0
2	$\frac{1}{2}\frac{1}{G_{-1,1}{g_{\mathrm{T}_1}}}\left(g_{\mathrm{T}_2}A_1^*A_2\right)$
3	0
4	$\frac{3}{16}\frac{1}{G_{-1,1}{g_{\mathrm{T}_1}}^5}A_1^*A_2\big({g_{\mathrm{T}_2}}^3{g_{\mathrm{T}_1}}^2A_1A_1^* - 3g_{\mathrm{T}_2}g_{\mathrm{T}_3}{g_{\mathrm{T}_1}}^3A_1A_1^*$ $+{g_{\mathrm{T}_2}}^3A_2A_2^* - 2g_{\mathrm{T}_2}g_{\mathrm{T}_3}g_{\mathrm{T}_1}A_2A_2^* + g_{\mathrm{T}_4}{g_{\mathrm{T}_1}}^4A_1A_1^* + g_{\mathrm{T}_4}{g_{\mathrm{T}_1}}^2A_2A_2^*\big)$
5	0

Table 5.17: The 2-tone response of a differential pair single-balanced mixer at $f_1 + f_2$ up to the fifth order.

n	$V_{\mathrm{o}2}(f_1 + f_2) - V_{\mathrm{o}1}(f_1 + f_2)$
1	0
2	$\frac{1}{2}\frac{1}{G_{1,1}{g_{\mathrm{T}_1}}}\left(g_{\mathrm{T}_2}A_1A_2\right)$
3	0
4	$\frac{3}{16}\frac{1}{G_{1,1}{g_{\mathrm{T}_1}}^5}A_1A_2\big({g_{\mathrm{T}_2}}^3{g_{\mathrm{T}_1}}^2A_1A_1^* - 3g_{\mathrm{T}_2}g_{\mathrm{T}_3}{g_{\mathrm{T}_1}}^3A_1A_1^*$ $+{g_{\mathrm{T}_2}}^3A_2A_2^* - 2g_{\mathrm{T}_2}g_{\mathrm{T}_3}g_{\mathrm{T}_1}A_2A_2^* + g_{\mathrm{T}_4}{g_{\mathrm{T}_1}}^4A_1A_1^* + g_{\mathrm{T}_4}{g_{\mathrm{T}_1}}^2A_2A_2^*\big)$
5	0

Table 5.18: The 2-tone response of a differential pair single-balanced mixer at f_1 up to the fifth order.

n	$V_{o2}(f_1) - V_{o1}(f_1)$
1	$\left(\frac{g_{T_1}}{G_{1,0}}\right) A_1$
2	0
3	$-\frac{1}{16}\frac{1}{G_{1,0}{g_{T_1}}^3} A_1 \left(6{g_{T_2}}^2{g_{T_1}}^2 A_1 A_1^* + 4{g_{T_2}}^2 A_2 A_2^* - 3g_{T_3}{g_{T_1}}^3 A_1 A_1^* - 6g_{T_3} g_{T_1} A_2 A_2^*\right)$
4	0
5	$\frac{1}{128}\frac{1}{G_{1,0}{g_{T_1}}^7} A_1 \big(-36{g_{T_2}}^4{g_{T_1}}^2 A_1 A_1^* A_2 A_2^* - 30{g_{T_2}}^4 {A_2}^2 {A_2^*}^2$ $+126{g_{T_2}}^2 g_{T_3}{g_{T_1}}^3 A_1 A_1^* A_2 A_2^* + 75{g_{T_2}}^2 g_{T_3} g_{T_1} {A_2}^2 {A_2^*}^2 - 10{g_{T_2}}^4{g_{T_1}}^4 {A_1}^2 {A_1^*}^2$ $+45 g_{T_3}{g_{T_2}}^2{g_{T_1}}^5 {A_1}^2 {A_1^*}^2 - 30 g_{T_2} g_{T_4}{g_{T_1}}^6 {A_1}^2 {A_1^*}^2 - 84 g_{T_2} g_{T_4}{g_{T_1}}^4 A_1 A_1^* A_2 A_2^*$ $-42 g_{T_2} g_{T_4}{g_{T_1}}^2 {A_2}^2 {A_2^*}^2 - 54{g_{T_3}}^2{g_{T_1}}^4 A_1 A_1^* A_2 A_2^* - 18{g_{T_3}}^2{g_{T_1}}^2 {A_2}^2 {A_2^*}^2$ $+5 g_{T_5}{g_{T_1}}^7 {A_1}^2 {A_1^*}^2 + 30 g_{T_5}{g_{T_1}}^5 A_1 A_1^* A_2 A_2^* + 15 g_{T_5}{g_{T_1}}^3 {A_2}^2 {A_2^*}^2 \big)$

Table 5.19: The 2-tone response of a differential pair single-balanced mixer at $2f_2 - f_1$ up to the fifth order.

n	$V_{o2}(2f_2 - f_1) - V_{o1}(2f_2 - f_1)$
1	0
2	0
3	$-\frac{1}{16}\frac{1}{G_{-1,2}{g_{T_1}}^3} A_1^* {A_2}^2 \left(2{g_{T_2}}^2 - 3 g_{T_3} g_{T_1}\right)$
4	0
5	$\frac{1}{128}\frac{1}{G_{-1,2}{g_{T_1}}^7} A_1^* {A_2}^2 \big(-18{g_{T_2}}^4 A_1 A_1^* {g_{T_1}}^2 + 63{g_{T_2}}^2 g_{T_3}{g_{T_1}}^3 A_1 A_1^*$ $-42 g_{T_2} g_{T_4}{g_{T_1}}^4 A_1 A_1^* - 20{g_{T_2}}^4 A_2 A_2^* + 50{g_{T_2}}^2 g_{T_3} g_{T_1} A_2 A_2^*$ $-28 g_{T_2} g_{T_4}{g_{T_1}}^2 A_2 A_2^* - 27{g_{T_3}}^2{g_{T_1}}^4 A_1 A_1^* - 12{g_{T_3}}^2{g_{T_1}}^2 A_2 A_2^*$ $+15 g_{T_5}{g_{T_1}}^5 A_1 A_1^* + 10 g_{T_5}{g_{T_1}}^3 A_2 A_2^* \big)$

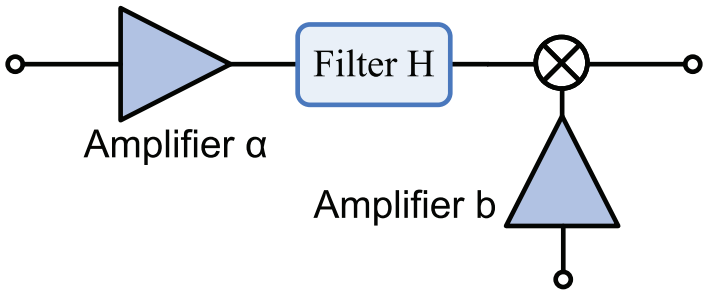


Figure 5.5: Block diagram of a RF receiver front-end.

Table 5.20: The 2-tone response of a differential pair single-balanced mixer at $2f_2 + f_1$ up to the fifth order.

n	$V_{o2}(2f_2 + f_1) - V_{o1}(2f_2 + f_1)$
1	0
2	0
3	$-\frac{1}{16}\frac{1}{G_{1,2}{g_{T_1}}^3}A_1{A_2}^2\left(2{g_{T_2}}^2 - 3g_{T_3}g_{T_1}\right)$
4	0
5	$\frac{1}{128}\frac{1}{G_{1,2}{g_{T_1}}^7}A_1{A_2}^2\big(-18{g_{T_2}}^4A_1A_1^*{g_{T_1}}^2 + 63{g_{T_2}}^2g_{T_3}{g_{T_1}}^3A_1A_1^*$ $-42g_{T_2}g_{T_4}{g_{T_1}}^4A_1A_1^* - 20{g_{T_2}}^4A_2A_2^* + 50{g_{T_2}}^2g_{T_3}g_{T_1}A_2A_2^*$ $-28g_{T_2}g_{T_4}{g_{T_1}}^2A_2A_2^* - 27{g_{T_3}}^2{g_{T_1}}^4A_1A_1^* - 12{g_{T_3}}^2{g_{T_1}}^2A_2A_2^*$ $+15g_{T_5}{g_{T_1}}^5A_1A_1^* + 10g_{T_5}{g_{T_1}}^3A_2A_2^*\big)$

Table 5.21: The 2-tone response of a differential pair single-balanced mixer at $3f_1$ up to the fifth order.

n	$V_{o2}(3f_1) - V_{o1}(3f_1)$
1	0
2	0
3	$-\frac{1}{16}\frac{1}{G_{3,0}{g_{T_1}}}{A_1}^3\left(2{g_{T_2}}^2 - g_{T_3}g_{T_1}\right)$
4	0
5	$\frac{1}{256}\frac{1}{G_{3,0}{g_{T_1}}^5}{A_1}^3\big(-24{g_{T_2}}^4A_2A_2^* + 84{g_{T_2}}^2g_{T_3}g_{T_1}A_2A_2^*$ $-10{g_{T_2}}^4{g_{T_1}}^2A_1A_1^* + 45{g_{T_2}}^2g_{T_3}{g_{T_1}}^3A_1A_1^* - 30g_{T_2}g_{T_4}{g_{T_1}}^4A_1A_1^*$ $-56g_{T_2}g_{T_4}{g_{T_1}}^2A_2A_2^* - 36{g_{T_3}}^2{g_{T_1}}^2A_2A_2^* + 5g_{T_5}{g_{T_1}}^5A_1A_1^*$ $+20g_{T_5}{g_{T_1}}^3A_2A_2^*\big)$

5.3.2 RF Receiver Front-End

The system-level diagram of a receiver front-end is shown in Fig. 5.5. For compatibility with Volterra circuit analysis, the signal applied at each input is represented as a multi-tone. To simplify the analysis, each signal is again represented as a sinusoid.

Equivalent Circuit Model

Compare to transistor level circuits where standard equivalent circuit models of varying complexity are used to model the active devices, behavioral models require further user decisions to balance modeling complexity and fidelity. To support RF grey-box model identification as discussed in Section 4.2.2 and 4.4.2, the equivalent circuit model of each system-level functional block should reflect the topology of the actual circuit-level implementation of each block, but with a minimum required amount of detail to limit the number of model parameters that has to be estimated.

In the following analysis focusing on the weakly nonlinear operation, the dominant nonlinear behavior of the amplifier and behavioral mixer nonlinearities are modeled using a combination of power series representations and linear frequency-dependent circuit elements. Microwave and RF circuits generally have a linear small signal response that departs from linearity as applied signals become larger. Also they tend to have rich frequency-dependent responses, but an intrinsic linearity that is largely frequency independent. Since the linear parameters constitute the majority of the model parameters, this behavior is exploited to reduce problem complexity.

For a filter, Fig. 5.6 shows a frequency-domain model which consists of a combination of linear frequency-dependent impedances and transfer functions. It is compatible with the grey-box model identification methodology outlined in chapter 4, and with frequency-domain harmonic balance and Volterra circuit simulators. For transient domain simulation, the linear frequency response can be fitted as a rational function.

For an amplifier, Fig. 5.7 shows a simple frequency-domain y-parameter model. This model

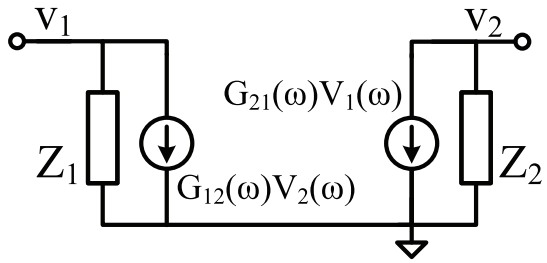


Figure 5.6: A frequency-domain filter model equivalent circuit, with element definitions similar to a y-parameter model.

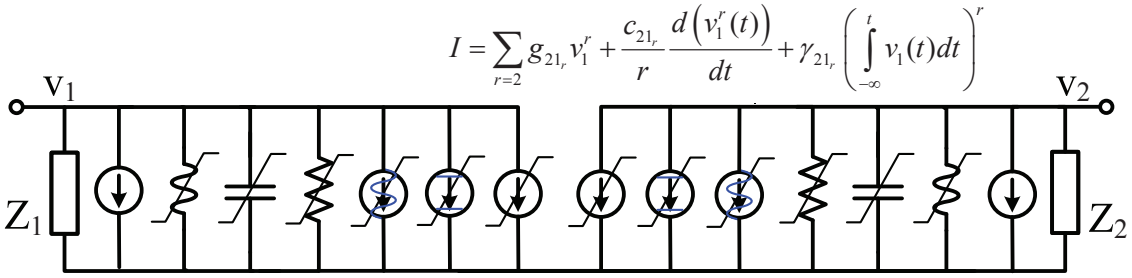


Figure 5.7: A simple frequency-domain amplifier model equivalent circuit, with element definitions similar to a y-parameter model. The port impedance behavior is modeled by a parallel combination of linear frequency-dependent impedances, nonlinear resistances, nonlinear capacitances and nonlinear inductances. Analogously, the transfer characteristic is modeled as a parallel combination of linear frequency-dependent transconductances, nonlinear transconductances, nonlinear trans-capacitances and a nonlinear trans-inductances. (The latter two are similar to their 2-terminal counterpart, except that the generated nonlinear current is summed at terminals other than terminals of the controlling variable.)

is very similar to a 2-port RF transistor model in [61] where each port has a parallel connection of dependent current sources controlled by port voltages and weighted in frequency. To see the similarity, a particular spectral component in the current due to port impedance behavior alone is

$$I_{\mathrm{NL}}(f_{\mathrm{IM}_n}) = \frac{V_1(f_{\mathrm{IM}_n})}{Z_1(f_{\mathrm{IM}_n})} + \sum_{n=2}^{N}\sum_{r=2}^{n}\frac{1}{2}^{r}\Big(\underbrace{\sum_{\{[n_1\ldots n_r]\}} C \sum_{\{[f_{\mathrm{IM}_{n_1}}\cdots f_{\mathrm{IM}_{nr}}]\}}}\Big[g_r + \frac{c_r}{r} j2\pi f_{\mathrm{IM}_n} + \frac{\gamma_r}{j2\pi f_{\mathrm{IM}_{n_1}}\ldots j2\pi f_{\mathrm{IM}_{n_r}}}\Big] V_{n_1}(f_{\mathrm{IM}_{n_1}})\ldots V_{n_r}(f_{\mathrm{IM}_{n_r}})\Big) \tag{5.48}$$

based on (5.29), (5.33) and (5.37). The expression will have similar form when the port transfer behavior is taken into account. What is notable is the bracketed complex coefficient with degree r dependence and fixed frequency weighting, which models the input amplitude dependent nonlinear behavior of the resistive and reactive nonlinearities.

Based on the above observation, Fig. 5.8 shows a more general frequency-domain amplifier model with arbitrary frequency weighting for the nonlinear coefficients, which is more suitable for modeling the complex nonlinear interactions in multi-stage amplifiers. For comparison with (5.48), the bracketed complex coefficient becomes

$$\begin{aligned} &\Re(g_r) \quad K_R(f_{\mathrm{IM}_n}) \quad H_R(f_{\mathrm{IM}_{n_1}})\ldots H_R(f_{\mathrm{IM}_{n_r}}) \\ +j&\Im(g_r) \quad K_I(f_{\mathrm{IM}_n}) \quad H_I(f_{\mathrm{IM}_{n_1}})\ldots H_I(f_{\mathrm{IM}_{n_r}}) \end{aligned} \tag{5.49}$$

where the nonlinear coefficient g_r is now complex and the frequency-dependent K_R, K_I, H_R and H_I are real. Each of the nonlinear circuit elements underlined with a brace is implemented as the sub-circuit shown. Note that the bottom branch of the sub-circuit has a quadrature phase shift associated with K_I. In transient simulation, this 90° phase shift can be implemented as a filter [96].

Using the above models simplified for unilateral operation, the resulting RF receiver front-end equivalent circuit is shown in Fig. 5.9. Here the unilateral amplifier model is further

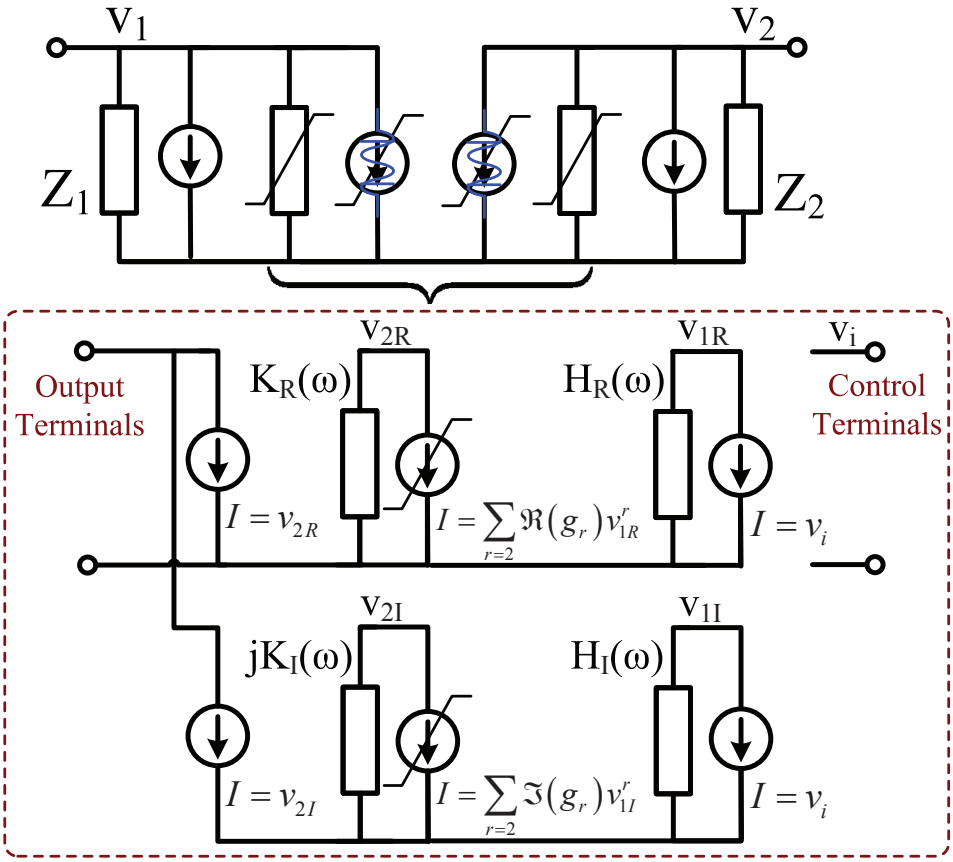


Figure 5.8: A more general frequency-domain amplifier model based on Fig. 5.7. The arbitrary frequency weighting for the nonlinear coefficients is more suitable for modeling the complex nonlinear interactions in multi-stage amplifiers.

simplified by equaling K_R, K_I, H_R and H_I to one, so coefficients a_{zr}, a_{g_r}, b_{zr} and b_{g_r} are complex. The amplifier output impedance is also assumed to be linear, and can be lumped with the input impedance of the filter model.

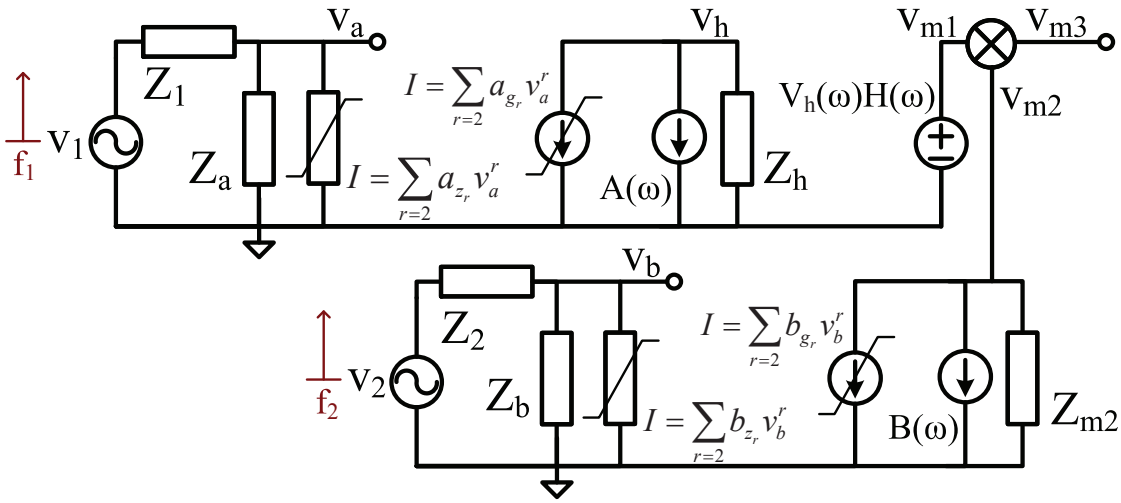


Figure 5.9: A RF receiver front-end equivalent circuit model.

Circuit Analysis Using Direct Calculation Method

Due to the use of frequency-domain circuit elements and the presence of complex coefficients in the amplifier model, the frequency-domain circuit equations are much simpler than the time

domain ones. For nodal analysis, Kirchoff's current law yields

$$\begin{aligned}
\frac{V_{\rm a}}{Z_1}+\frac{V_{\rm a}}{Z_{\rm a}} &= \frac{V_1}{Z_1}-I_{\rm NL}(a_{\rm z},v_{\rm a}) && (5.50)\\
AV_{\rm a}+\frac{V_{\rm h}}{Z_{\rm h}} &= -I_{\rm NL}(a_{\rm g},v_{\rm a}) && (5.51)\\
HV_{\rm h}-V_{\rm m1} &= 0 && (5.52)\\
\frac{V_{\rm b}}{Z_2}+\frac{V_{\rm b}}{Z_{\rm b}} &= \frac{V_2}{Z_2}-I_{\rm NL}(b_{\rm z},v_{\rm b}) && (5.53)\\
BV_{\rm b}+\frac{V_{\rm m2}}{Z_{\rm m2}} &= -I_{\rm NL}(b_{\rm g},v_{\rm b}) && (5.54)\\
V_{\rm m3} &= I_{\rm NL}(v_{\rm m1},v_{\rm m2}) && (5.55)
\end{aligned}$$

where $I_{\rm NL}$ denotes the nonlinear current with dependent parameters inside the parenthesis.

In the resulting spectral balance equation (5.30) to be solved in the direct calculation method, the left-hand side representing the linearized circuit becomes

$$\mathbf{Y}(s)\overrightarrow{V}(s)=\begin{bmatrix}
\frac{1}{Z_1}+\frac{1}{Z_{\rm a}} & 0 & 0 & 0 & 0 & 0\\
A & \frac{1}{Z_{\rm h}} & 0 & 0 & 0 & 0\\
0 & H & -1 & 0 & 0 & 0\\
0 & 0 & 0 & \frac{1}{Z_2}+\frac{1}{Z_{\rm b}} & 0 & 0\\
0 & 0 & 0 & B & \frac{1}{Z_{\rm m2}} & 0\\
0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}\begin{bmatrix}
V_{\rm a}(s)\\ V_{\rm h}(s)\\ V_{\rm m1}(s)\\ V_{\rm b}(s)\\ V_{\rm m2}(s)\\ V_{\rm m3}(s)
\end{bmatrix} \qquad (5.56)$$

where the laplace variable s takes on $j2\pi f_{\rm IM}$ at each IM frequency. The right-hand side linear

current source vector at the 2-tone input frequencies f_1 and f_2 are

$$2\overrightarrow{I_{\mathrm{LIN}}}(f_1) = 2\begin{bmatrix} \frac{A_1/2}{Z_1} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \; 2\overrightarrow{I_{\mathrm{LIN}}}(f_2) = 2\begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{A_2/2}{Z_2} \\ 0 \\ 0 \end{bmatrix} \tag{5.57}$$

where A_1 and A_2 are the complex amplitudes of the input tones. The right-hand side nonlinear current source vector is evaluated in the recursive direct calculation algorithm, as the response at the contributing IM frequencies are found for the controlling variables of the device nonlinearities. The required dependence information for this evaluation is store in the nonlinear current data structure as described in Section 5.2.2.

Analysis Results

To limit the length of the expressions, the V_{m3} output responses are presented only up to the third order in Table 5.22 for $f_2 - f_1$, Table 5.23 for $f_1 + f_2$, Table 5.24 for f_1, Table 5.25 for f_2, Table 5.26 for $2f_1 - f_2$, Table 5.27 for $2f_1 + f_2$, Table 5.28 for $2f_2 - f_1$, and Table 5.29 for $2f_2 + f_1$.

The V_{m3} output responses at $2f_1$, $2f_2$, $3f_1$ and $3f_2$ are zero, at least for responses up to the third order. The frequency-dependent linear elements $Z_(s)$ at each IM frequency $f_{\mathrm{IM}} = kf_1 + lf_2$ is denote as $Z_{(k,l)}$. Note that there are contributions of different order at each f_{IM}, thus the

Table 5.22: The 2-tone response of a RF receiver front-end model at $f_2 - f_1$ up to the third order.

n	$V_{\mathrm{m3}}(f_2 - f_1)$
1	0
2	$\frac{1}{2}A_1^* A_2\ mH_{(1,0)}^* A_{(1,0)}^* Z_{\mathrm{a}(1,0)}^* Z_{\mathrm{h}(1,0)}^* B_{(0,1)} Z_{\mathrm{b}(0,1)} Z_{\mathrm{m2}(0,1)}$ $/\big(Z_{\mathrm{a}(1,0)}^* Z_{\mathrm{b}(0,1)} + Z_{\mathrm{a}(1,0)}^* Z_{2(0,1)} + Z_{1(1,0)}^* Z_{\mathrm{b}(0,1)} + Z_{1(1,0)}^* Z_{2(0,1)}\big)$
3	0

Table 5.23: The 2-tone response of a RF receiver front-end model at $f_1 + f_2$ up to the third order.

n	$V_{\mathrm{m3}}(f_1 + f_2)$
1	0
2	$\frac{1}{2}A_1 A_2\ mH_{(1,0)} A_{(1,0)} Z_{\mathrm{a}(1,0)} Z_{\mathrm{h}(1,0)} B_{(0,1)} Z_{\mathrm{b}(0,1)} Z_{\mathrm{m2}(0,1)}$ $/\big(Z_{\mathrm{a}(1,0)} Z_{\mathrm{b}(0,1)} + Z_{\mathrm{a}(1,0)} Z_{2(0,1)} + Z_{1(1,0)} Z_{\mathrm{b}(0,1)} + Z_{1(1,0)} Z_{2(0,1)}\big)$
3	0

Table 5.24: The 2-tone response of a RF receiver front-end model at f_1 up to the third order.

n	$V_{\mathrm{m3}}(f_1)$
1	0
2	0
3	$\frac{1}{2}A_1 A_2 A_2^*\ mH_{(1,0)} A_{(1,0)} Z_{\mathrm{b}(0,1)}^* Z_{\mathrm{b}(0,1)} Z_{\mathrm{h}(1,0)} Z_{\mathrm{a}(1,0)} Z_{\mathrm{m2}(0,0)}$ $\big(-b_{z_2} B_{(0,0)} Z_{2(0,0)} Z_{\mathrm{b}(0,0)} + b_{g_2} Z_{\mathrm{b}(0,0)} + b_{g_2} Z_{2(0,0)}\big)$ $/\big(Z_{\mathrm{a}(1,0)} Z_{\mathrm{b}(0,0)} Z_{\mathrm{b}(0,1)} Z_{\mathrm{b}(0,1)}^* + Z_{\mathrm{a}(1,0)} Z_{\mathrm{b}(0,0)} Z_{\mathrm{b}(0,1)} Z_{2(0,1)}^* + Z_{\mathrm{a}(1,0)} Z_{\mathrm{b}(0,0)} Z_{2(0,1)} Z_{\mathrm{b}(0,1)}^*$ $+Z_{\mathrm{a}(1,0)} Z_{\mathrm{b}(0,0)} Z_{2(0,1)} Z_{2(0,1)}^* + Z_{\mathrm{a}(1,0)} Z_{2(0,0)} Z_{\mathrm{b}(0,1)} Z_{\mathrm{b}(0,1)}^* + Z_{\mathrm{a}(1,0)} Z_{2(0,0)} Z_{\mathrm{b}(0,1)} Z_{2(0,1)}^*$ $+Z_{\mathrm{a}(1,0)} Z_{2(0,0)} Z_{2(0,1)} Z_{\mathrm{b}(0,1)}^* + Z_{\mathrm{a}(1,0)} Z_{2(0,0)} Z_{2(0,1)} Z_{2(0,1)}^* + Z_{1(1,0)} Z_{\mathrm{b}(0,0)} Z_{\mathrm{b}(0,1)} Z_{\mathrm{b}(0,1)}^*$ $+Z_{1(1,0)} Z_{\mathrm{b}(0,0)} Z_{\mathrm{b}(0,1)} Z_{2(0,1)}^* + Z_{1(1,0)} Z_{\mathrm{b}(0,0)} Z_{2(0,1)} Z_{\mathrm{b}(0,1)}^* + Z_{1(1,0)} Z_{\mathrm{b}(0,0)} Z_{2(0,1)} Z_{2(0,1)}^*$ $+Z_{1(1,0)} Z_{2(0,0)} Z_{\mathrm{b}(0,1)} Z_{\mathrm{b}(0,1)}^* + Z_{1(1,0)} Z_{2(0,0)} Z_{\mathrm{b}(0,1)} Z_{2(0,1)}^* + Z_{1(1,0)} Z_{2(0,0)} Z_{2(0,1)} Z_{\mathrm{b}(0,1)}^*$ $+Z_{1(1,0)} Z_{2(0,0)} Z_{2(0,1)} Z_{2(0,1)}^*\big)$

Table 5.25: The 2-tone response of a RF receiver front-end model at f_2 up to the third order.

n	$V_{\mathrm{m3}}(f_2)$
1	0
2	0
3	$\frac{1}{2}A_1A_1^*A_2\ mH_{(0,0)}B_{(0,1)}Z^*_{\mathrm{a}(1,0)}Z_{\mathrm{a}(1,0)}Z_{\mathrm{h}(0,0)}Z_{\mathrm{b}(0,1)}Z_{\mathrm{m2}(0,1)}$ $\big(-a_{z_2}A_{(0,0)}Z_{1(0,0)}Z_{\mathrm{a}(0,0)}+a_{g_2}Z_{\mathrm{a}(0,0)}+a_{g_2}Z_{1(0,0)}\big)$ $/\big(Z_{\mathrm{a}(0,0)}Z_{\mathrm{a}(1,0)}Z^*_{\mathrm{a}(1,0)}Z_{\mathrm{b}(0,1)}+Z_{\mathrm{a}(0,0)}Z_{\mathrm{a}(1,0)}Z^*_{\mathrm{a}(1,0)}Z_{2(0,1)}+Z_{\mathrm{a}(0,0)}Z_{\mathrm{a}(1,0)}Z^*_{1(1,0)}Z_{\mathrm{b}(0,1)}$ $+Z_{\mathrm{a}(0,0)}Z_{\mathrm{a}(1,0)}Z^*_{1(1,0)}Z_{2(0,1)}+Z_{\mathrm{a}(0,0)}Z_{1(1,0)}Z^*_{\mathrm{a}(1,0)}Z_{\mathrm{b}(0,1)}+Z_{\mathrm{a}(0,0)}Z_{1(1,0)}Z^*_{\mathrm{a}(1,0)}Z_{2(0,1)}$ $+Z_{\mathrm{a}(0,0)}Z_{1(1,0)}Z^*_{1(1,0)}Z_{\mathrm{b}(0,1)}+Z_{\mathrm{a}(0,0)}Z_{1(1,0)}Z^*_{1(1,0)}Z_{2(0,1)}+Z_{1(0,0)}Z_{\mathrm{a}(1,0)}Z^*_{\mathrm{a}(1,0)}Z_{\mathrm{b}(0,1)}$ $+Z_{1(0,0)}Z_{\mathrm{a}(1,0)}Z^*_{\mathrm{a}(1,0)}Z_{2(0,1)}+Z_{1(0,0)}Z_{\mathrm{a}(1,0)}Z^*_{1(1,0)}Z_{\mathrm{b}(0,1)}+Z_{1(0,0)}Z_{\mathrm{a}(1,0)}Z^*_{1(1,0)}Z_{2(0,1)}$ $+Z_{1(0,0)}Z_{1(1,0)}Z^*_{\mathrm{a}(1,0)}Z_{\mathrm{b}(0,1)}+Z_{1(0,0)}Z_{1(1,0)}Z^*_{\mathrm{a}(1,0)}Z_{2(0,1)}+Z_{1(0,0)}Z_{1(1,0)}Z^*_{1(1,0)}Z_{\mathrm{b}(0,1)}$ $+Z_{1(0,0)}Z_{1(1,0)}Z^*_{1(1,0)}Z_{2(0,1)}\big)$

Table 5.26: The 2-tone response of a RF receiver front-end model at $2f_1-f_2$ up to the third order.

n	$V_{\mathrm{m3}}(2f_1-f_2)$
1	0
2	0
3	$\frac{1}{4}A_1^2A_2^*\ mH_{(2,0)}Z_{\mathrm{h}(2,0)}{Z_{\mathrm{a}(1,0)}}^2B^*_{(0,1)}Z^*_{\mathrm{b}(0,1)}Z^*_{\mathrm{m2}(0,1)}$ $\big(-a_{z_2}A_{(2,0)}Z_{1(2,0)}Z_{\mathrm{a}(2,0)}+a_{g_2}Z_{\mathrm{a}(2,0)}+a_{g_2}Z_{1(2,0)}\big)$ $/\big(Z_{\mathrm{a}(2,0)}{Z_{\mathrm{a}(1,0)}}^2Z^*_{\mathrm{b}(0,1)}+Z_{\mathrm{a}(2,0)}{Z_{\mathrm{a}(1,0)}}^2Z^*_{2(0,1)}+2Z_{\mathrm{a}(2,0)}Z_{\mathrm{a}(1,0)}Z_{1(1,0)}Z^*_{\mathrm{b}(0,1)}$ $+2Z_{\mathrm{a}(2,0)}Z_{\mathrm{a}(1,0)}Z_{1(1,0)}Z^*_{2(0,1)}+Z_{\mathrm{a}(2,0)}{Z_{1(1,0)}}^2Z^*_{\mathrm{b}(0,1)}+Z_{\mathrm{a}(2,0)}{Z_{1(1,0)}}^2Z^*_{2(0,1)}$ $+Z_{1(2,0)}{Z_{\mathrm{a}(1,0)}}^2Z^*_{\mathrm{b}(0,1)}+Z_{1(2,0)}{Z_{\mathrm{a}(1,0)}}^2Z^*_{2(0,1)}+2Z_{1(2,0)}Z_{\mathrm{a}(1,0)}Z_{1(1,0)}Z^*_{\mathrm{b}(0,1)}$ $+2Z_{1(2,0)}Z_{\mathrm{a}(1,0)}Z_{1(1,0)}Z^*_{2(0,1)}+Z_{1(2,0)}{Z_{1(1,0)}}^2Z^*_{\mathrm{b}(0,1)}+Z_{1(2,0)}{Z_{1(1,0)}}^2Z^*_{2(0,1)}\big)$

Table 5.27: The 2-tone response of a RF receiver front-end model at $2f_1 + f_2$ up to the third order.

n	$V_{\mathrm{m3}}(2f_1 + f_2)$
1	0
2	0
3	$\frac{1}{4}A_1^2A_2\ mH_{(2,0)}Z_{\mathrm{h}(2,0)}Z_{\mathrm{a}(1,0)}{}^2B_{(0,1)}Z_{\mathrm{b}(0,1)}Z_{\mathrm{m2}(0,1)}$ $\big(-a_{z_2}A_{(2,0)}Z_{1(2,0)}Z_{\mathrm{a}(2,0)} + a_{g_2}Z_{\mathrm{a}(2,0)} + a_{g_2}Z_{1(2,0)}\big)$ $/\big(Z_{\mathrm{a}(2,0)}Z_{\mathrm{a}(1,0)}{}^2Z_{\mathrm{b}(0,1)} + Z_{\mathrm{a}(2,0)}Z_{\mathrm{a}(1,0)}{}^2Z_{2(0,1)} + 2Z_{\mathrm{a}(2,0)}Z_{\mathrm{a}(1,0)}Z_{1(1,0)}Z_{\mathrm{b}(0,1)}$ $+2Z_{\mathrm{a}(2,0)}Z_{\mathrm{a}(1,0)}Z_{1(1,0)}Z_{2(0,1)} + Z_{\mathrm{a}(2,0)}Z_{1(1,0)}{}^2Z_{\mathrm{b}(0,1)} + Z_{\mathrm{a}(2,0)}Z_{1(1,0)}{}^2Z_{2(0,1)}$ $+Z_{1(2,0)}Z_{\mathrm{a}(1,0)}{}^2Z_{\mathrm{b}(0,1)} + Z_{1(2,0)}Z_{\mathrm{a}(1,0)}{}^2Z_{2(0,1)} + 2Z_{1(2,0)}Z_{\mathrm{a}(1,0)}Z_{1(1,0)}Z_{\mathrm{b}(0,1)}$ $+2Z_{1(2,0)}Z_{\mathrm{a}(1,0)}Z_{1(1,0)}Z_{2(0,1)} + Z_{1(2,0)}Z_{1(1,0)}{}^2Z_{\mathrm{b}(0,1)} + Z_{1(2,0)}Z_{1(1,0)}{}^2Z_{2(0,1)}\big)$

Table 5.28: The 2-tone response of a RF receiver front-end model at $2f_2 - f_1$ up to the third order.

n	$V_{\mathrm{m3}}(2f_2 - f_1)$
1	0
2	0
3	$\frac{1}{4}A_1^*A_2^2\ mH^*_{(1,0)}A^*_{(1,0)}Z^*_{\mathrm{a}(1,0)}Z^*_{\mathrm{h}(1,0)}Z_{\mathrm{b}(0,1)}{}^2Z_{\mathrm{m2}(0,2)}$ $\big(b_{g_2}Z_{\mathrm{b}(0,2)} + b_{g_2}Z_{2(0,2)} - B_{(0,2)}b_{z_2}Z_{2(0,2)}Z_{\mathrm{b}(0,2)}\big)$ $/\big(Z^*_{\mathrm{a}(1,0)}Z_{\mathrm{b}(0,2)}Z_{\mathrm{b}(0,1)}{}^2 + 2Z^*_{\mathrm{a}(1,0)}Z_{\mathrm{b}(0,2)}Z_{\mathrm{b}(0,1)}Z_{2(0,1)} + Z^*_{\mathrm{a}(1,0)}Z_{\mathrm{b}(0,2)}Z^2_{2(0,1)}$ $+Z^*_{\mathrm{a}(1,0)}Z_{2(0,2)}Z_{\mathrm{b}(0,1)}{}^2 + 2Z^*_{\mathrm{a}(1,0)}Z_{2(0,2)}Z_{\mathrm{b}(0,1)}Z_{2(0,1)} + Z^*_{\mathrm{a}(1,0)}Z_{2(0,2)}Z^2_{2(0,1)}$ $+Z^*_{1(1,0)}Z_{\mathrm{b}(0,2)}Z_{\mathrm{b}(0,1)}{}^2 + 2Z^*_{1(1,0)}Z_{\mathrm{b}(0,2)}Z_{\mathrm{b}(0,1)}Z_{2(0,1)} + Z^*_{1(1,0)}Z_{\mathrm{b}(0,2)}Z^2_{2(0,1)}$ $+Z^*_{1(1,0)}Z_{2(0,2)}Z_{\mathrm{b}(0,1)}{}^2 + 2Z^*_{1(1,0)}Z_{2(0,2)}Z_{\mathrm{b}(0,1)}Z_{2(0,1)} + Z^*_{1(1,0)}Z_{2(0,2)}Z^2_{2(0,1)}\big)$

Table 5.29: The 2-tone response of a RF receiver front-end model at $2f_2 + f_1$ up to the third order.

n	$V_{\mathrm{m3}}(2f_2 + f_1)$
1	0
2	0
3	$\frac{1}{4}A_1A_2^2\ mH_{(1,0)}A_{(1,0)}Z_{\mathrm{a}(1,0)}Z_{\mathrm{h}(1,0)}Z_{\mathrm{b}(0,1)}{}^2Z_{\mathrm{m2}(0,2)}$ $\big(b_{g_2}Z_{\mathrm{b}(0,2)} + b_{g_2}Z_{2(0,2)} - B_{(0,2)}b_{z_2}Z_{2(0,2)}Z_{\mathrm{b}(0,2)}\big)$ $/\big(Z_{\mathrm{a}(1,0)}Z_{\mathrm{b}(0,2)}Z_{\mathrm{b}(0,1)}{}^2 + 2Z_{\mathrm{a}(1,0)}Z_{\mathrm{b}(0,2)}Z_{\mathrm{b}(0,1)}Z_{2(0,1)} + Z_{\mathrm{a}(1,0)}Z_{\mathrm{b}(0,2)}Z^2_{2(0,1)}$ $+Z_{\mathrm{a}(1,0)}Z_{2(0,2)}Z_{\mathrm{b}(0,1)}{}^2 + 2Z_{\mathrm{a}(1,0)}Z_{2(0,2)}Z_{\mathrm{b}(0,1)}Z_{2(0,1)} + Z_{\mathrm{a}(1,0)}Z_{2(0,2)}Z^2_{2(0,1)}$ $+Z_{1(1,0)}Z_{\mathrm{b}(0,2)}Z_{\mathrm{b}(0,1)}{}^2 + 2Z_{1(1,0)}Z_{\mathrm{b}(0,2)}Z_{\mathrm{b}(0,1)}Z_{2(0,1)} + Z_{1(1,0)}Z_{\mathrm{b}(0,2)}Z^2_{2(0,1)}$ $+Z_{1(1,0)}Z_{2(0,2)}Z_{\mathrm{b}(0,1)}{}^2 + 2Z_{1(1,0)}Z_{2(0,2)}Z_{\mathrm{b}(0,1)}Z_{2(0,1)} + Z_{1(1,0)}Z_{2(0,2)}Z^2_{2(0,1)}\big)$

compression effect is accounted for.

5.4 Conclusions

This chapter presented a improved direct calculation method of Volterra circuit analysis for handling compression effects lacking in the original work. In direct calculation method, the nonlinear current expressions are particular to the number of tones in the multi-tone excitation. One recursive algorithm was developed to compute nonlinear current expressions for various types of nonlinearity sources, and can handle arbitrary degrees of nonlinearity. Another recursive circuit analysis algorithm was developed to compute the nodal voltage IM responses, which makes use of a data structure storing the dependence information required to generate the nonlinear current.

The proposed direct calculation method is then implemented as a symbolic Volterra circuit simulator in Matlab. Typically in RF model identification schemes, the Volterra analysis portion is hand-crafted for the particular model being developed. The new symbolic evaluator is suitable for more generalized model fitting, again eliminating user tasks that can be automated.

Chapter 6

Modeling Cosite Inteference

6.1 Introduction

Cosite interference is electromagnetic interference (EMI) that occurs when multiple radios operate in close proximity [97] [98] [99] [100] [101] [102]. Current technology typically implements dedicated antenna for each radio operating in a specified frequency band. Due to space constraints on the communication platform, the antennas are often placed in close proximity which results in poor isolation. Since the co-located radios are simultaneously transmitting and receiving, the poor antenna-to-antenna isolation couples interference between the radios. For radios operating in different frequency bands, the filtering in the RF front-end will greatly reduce the interference. However for radios operating in the same frequency band, the filtering in the RF front-end has no effect.

Current generation of multi-band and multi-standard consumer wireless devices hosts multiple radios, but exhibits negligible cosite interference because two radios seldom operate at the same time. A notable exception is when a cell phone communicates to a cell tower and a blue-tooth headset at the same time. However both cell phone and blue-tooth transmissions are small (on the order of 1 W), and more importantly, each radio is operated in a different frequency band, so interfering signals are reduced by the band filter in each radio's RF front-end.

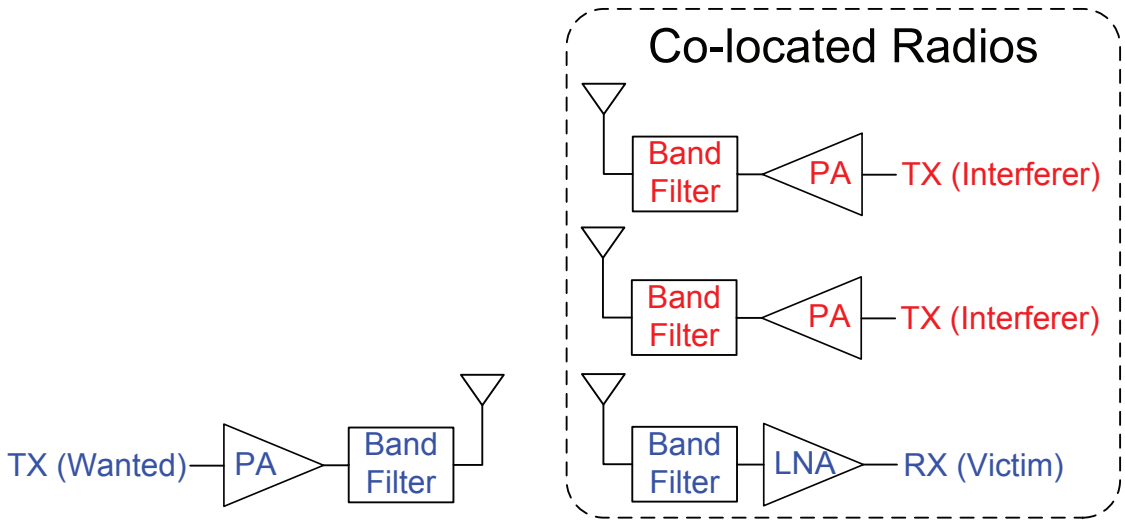


Figure 6.1: In a cosite interference scenario, multiple co-located radios simultaneously operate in the same frequency band. The poor antenna-to-antenna isolation couples interference between the radios.

Cosite interference will become a more significant issue in the future as more services make use of the same unlicensed spectrum.

In comparison, military communication systems transmit signals on the order of 100 W, and more importantly, dozens of co-located radios operate simultaneously in the same frequency band. So a band filter cannot eliminate the interfering signals. Using additional narrowband filter in each radio's front-end is not applicable, because to prevent jamming, the transmission of each radio is frequency-hopped.

Modeling interference is essential in uncovering the underlying mechanisms and ensuring sound design practice. As pointed out in Section 1.1.1, a hierarchical view is needed to deal with the complexity of modern communication systems. Much of the existing communication system analysis techniques for single-input single-output (SISO) point-to-point radio links [103] and for multi-input multi-output (MIMO) radio networks [104], assume that communication signals are transmitted by linear systems in additive noisy channels. Thus only when the RF front-end is suitably designed to behave effectively as a linear system with an additive noise source, can communication system analysis techniques be applied.

6.1.1 Cosite Inteference and the Near-Far Problem

Cosite interference is very similar to the well known "near-far" problem in RF engineering. In both cases, the receiver is trying to discern a weak transmission from a far away radio in the presence of strong interfering transmissions from nearby radios. Thus the sources of cosite interference is very much the same as in any wireless communication system [99] [105], such as IM distortion generated in the transceiver, broadband and sideband noise generated by transmitters, reciprocal mixing of the receiver local oscillator (LO) noise, and passive IM distortion generated from the so called "rusty bolt" effect.

The significant difference from the "near-far" problem is that in cosite interference, the interfering radios just happen to be co-located with the receiver, which is the worst case possible. As a result, the receivers must simultaneously achieve high sensitivity to detect a low level transmission, and high dynamic range to tolerate large interferers, which leads to high power consumption. Often drastically different receiver architectures are implemented to overcome the penalty in power consumption.

Cosite interference can be mitigated by reducing the number antenna through shared usage, and using a combination of media access techniques. In the time division method, each of the co-located radios can be assigned different time slots so that only one radio will transmit or receive at a given time, eliminating the interference. However time slot scheduling is a problem for radios using different standards.

In the frequency division method, each of the co-located radios can be assigned different frequency slots in the band, and a comb filter-based RF signal distribution system [101] [102] routes all transmit and all receive signals to a common transmit antenna and a common receive antenna. Instead of assigning a fixed center frequency narrowband filter for each radio as discussed previously, transmit signals from all co-located radios are combined and input to a bank of filter-amplifier-filter cascade. The output of the cascade is connected to a common transmit antenna. On the receiver side, the signal from the common receive antenna is sent to a filter-amplifier-filter bank, and distributed to the receiver in each radio. The narrowband filter

before and after each amplifier eliminates the received interferers and the nonlinear distortion created during amplification.

In the code division method, direct-sequence spread spectrum technique can be used. This method makes interferers orthogonal to the wanted signal appear as noise to the receiver. However code scheduling is a problem for radios using different standards. In addition transmitters using quadrature modulation cannot achieve the same level of power output and efficiency using existing solid-state technology as FM transmitters.

6.1.2 Cosite Simulation in RF Design

Cosite interference modeling utilizes many of the same techniques used in RF front-end design. In the link-level design discussed in Section 1.1.1, the RF front-end is designed to meet a certain data rate and BER specifications, which can be translated to standard specifications (IM characteristics, adjacent channel selectivity, blocker profile, etc.) for a particular wireless standard. There are two types of design tasks involved in meeting the standard specifications.

In top-down design, the transceiver architecture and the characteristics of each block (gain, noise figure and intercept point) must be derived from the standard specifications. This is an iterative design process which searches for combinations of implementable architectures and block specifications that meet the standard specifications. The estimated transceiver performance is used with optimization schemes in the design-space exploration process, which demands simulation efficiency that can only be achieved using simple behavioral models for the transceiver blocks. The models have known parameters which are the block specifications, and are thus called parametric models.

In bottom-up verification, the transceiver is verified in simulation against the standard specifications after circuit implementation of each block is available. Again for simulation efficiency, behavioral models of varying degrees of fidelity are used to model the measured or simulated response of each block. In addition to simple parametric behavioral models, more widely used are data-based behavioral models which are more accurate. The parameters of

data-based models are identified from the measured or simulated response.

Cosite interference modeling verifies transceiver designs in realistic scenarios where several transceivers are laid out in a radio test-bed, in contrast with stand alone testing of a transmitter or a receiver. The wireless channel must be modeled in cosite interference simulations. Also in stand alone testing of a receiver, the interfering transmissions are represented using one-tone or two-tone signals such as those specified in a blocker profile. This assumes an ideal transmitter with no spurious transmission outside the transmission band, and may result in an overly stringent spectral mask specification. The actual transmitter output contains residual harmonic and IM spectral components that have not been completely eliminated by the filtering elements in the transmitter.

6.1.3 Simulation Approaches and Behavioral Models

For accurate prediction of interference and their effects on radio reception, modern CAD tools have become indispensable. In link-level design, typically, waveform level simulations are performed to check adherence to standard specifications. With further processing of the simulation results, BER performance can be evaluated. As discussed in Section 1.1.2 two types of approaches can be distinguished based on the fidelity of the models and the type of simulator.

In the formula-based approach, systems are described using functional languages such as Matalb or C, and cannot easily implement electrical characteristics. Here signals are represented in a low-pass equivalent (LPE) form in the time domain, and the model output is calculated from close-form expressions. Simple parametric behavioral models are used, and the model parameters are converted from block specifications (gain, noise figure and intercept point). For example IP2 and IP3 specifications can be analytically related to a third-order power series, and gain compression specification can be analytically related to a higher order power series. Then the power series is used to compute the model output in the time domain.

The formula-based approach is computational efficient, and compatible with development of signal processing algorithms. It is widely adopted in the system-level simulation and design

of RF frontends as in [106] [107] [108] [109] using the Matlab environment. This approach was also used for cosite simulation in [99] and [100], and showed acceptable accuracy in predicting BER.

In one type of simulation-based approach, systems are described by a structural language such as a SPICE netlist. Here real signals are represented in the time- or frequency-domain, and the model is simulated using SPICE like time-domain circuit simulators, or harmonic balance- or Volterra-based frequency-domain circuit simulators. This approach is capable of modeling more realistic behavior of RF circuits such as mismatch effects and broadband IM response, which are neglected in the formula-based approach but are essential in modeling EMI. However simulation of complex systems is only possible if behavioral models are developed from measured or simulated response of circuit components. This is the main difficulty of this approach.

In another type of simulation-based approach, systems are described by hardware description languages such as VHDL or Verilog, which are different from functional language and structural language in that behavioral descriptions are directly supported. RF and Mixed-signal simulation is supported through language extensions such as VHDL-AMS and Verilog-AMS. The VHDL-AMS simulation environment was used to model a WCDMA transceiver in [110] and an UWB impulse radio transceiver in [111].

6.2 Behavioral Modeling for Cosite Interference Simulation

Cosite interference modeling can be approximately partitioned into two parts. The electromagnetic field aspect of modeling concerns with capturing the antenna-to-antenna coupling between co-located radios. The circuit aspect of modeling is concerned with creating behavioral models of the RF front-end in each radio transceiver. In both cases, the modeling issues are the fidelity of the modeled response, and the compatibility with the time- or frequency-domain simulation.

As pointed out in Section 6.1.3, past work on cosite interference modeling in [99] and [100] applied the formula-based approach and used simple parametric models. However cosite interference appears over a wide frequency range, so simple parametric models cannot accurately

model the antenna-to-antenna coupling and the nonlinear circuit behavior. For example, IP3 may be adequate for modeling adjacent channel third-order distortion for an RF carrier with a narrowband modulation, but cannot accurately model the harmonic components or the IM components for excitations with several widely spaced RF carriers.

The current work focuses on the simulation-based approach which is more suited to verification. The modeling of antenna-to-antenna coupling is discussed in Section 6.2.1. The modeling of nonlinear RF front-end circuits is discussed in Section 6.2.2. Each part of the co-located radio setup is modeled using data-based behavioral models. The model of the radio test-bed is verified in circuit simulation against cosite interference measurements.

6.2.1 Modeling Antenna-to-Antenna Coupling

The antenna-to-antenna coupling must be modeled across a wide frequency range covering all spectral components generated by the nonlinear transceiver circuits up to a certain order. Here it is characterized in the steady-state frequency-domain using a VNA. The measured S-parameters can be directly used in frequency-domain circuit simulators based on the harmonic balance or the Volterra method. The step size of the frequency sweep determines the fidelity of the modeled response.

For transient time-domain simulation, S-parameter measurements are first transformed into equivalent y-parameters, then fitted to rational transfer function representations [86]. The fitting process needs to ensure causality and passivity [112]. The rational transfer function representation corresponds directly to the differential equation representation which is supported by transient time-domain simulators such as Spice. A roundabout alternative is to realize the rational transfer functions as a RLC filter network using a filter synthesis program.

6.2.2 Modeling Nonlinear RF Front-End Circuits

In cosite interference modeling, all spectral components generated by nonlinear RF front-end circuits up to a certain order must be replicated. To utilize a priori insight to transceiver archi-

tectures, grey-box models are used here. Compared to other black-box modeling approaches, the number of model parameters is reduced in comparison to Volterra filter based model, and greater fidelity is achieved in comparison to describing function based models.

The RF front-end circuits are characterized in the steady-state frequency-domain using a NVNA system from Section 3.3. The grey-box model is identified in the frequency domain using the identification methodology from Section 4.2. For transient time-domain simulation, linear frequency-dependent parameters of the grey-box model are fitted to a rational transfer function representation, which is similar to modeling antenna-to-antenna coupling.

Excitation Design and Model Parametrization

With the measurement and model identification issues addressed, the rest of the discussion focuses on the design of the model identification signal. The grey-box identification algorithm estimates the model parameters by using various nonlinear optimization schemes to minimize the error between modeled response and measured data. In effect the optimizer is solving a system of coupled nonlinear equations, with the unknown variables being the model parameters, and the known constraints being the measurements.

It is up to the user to setup the problem in such a way so as to avoid an under-determined case with more parameters than constraints which theoretically can have an infinite number of solutions, or an over-determined case with more constraints than parameters which may have no solution. The number of unknown parameters is controlled by user choice in parametrization, particularly in the linear model parameters since they constitute the majority of the model parameters. The number of known constraining measurements is controlled by user design of the model identification signal, which depends on the whether the excitation of the intended application is known.

Typically the excitation in the intended application is known, and often consists of several modulated RF carriers widely spaced at known frequencies. If the nonlinear circuit does not exhibit bandwidth-dependent memory effect when excited by the modulated excitation, the

model identification signal can simply be a multi-tone signal with tones at the center frequency of the RF carriers. To model the bandwidth-dependent memory effect, the multi-tone signal is modified by replacing the single tone at each RF carrier frequency with a two-tone signal.

For a given model identification signal and a user selected nonlinear order, a chosen model will generate nonlinear response at a set of IM frequencies. Since these IM responses will excite the filtering elements in the model, the IM frequencies are sorted into frequency bands for parametrization purposes. Sorting is based on linear S-parameter measurements, where IM frequencies with frequency response variation below a user-set threshold are grouped into the same band. This is an approximate sorting criteria based on the observation that any significant filtering in the transceiver chain will show up in the S-parameter measurement. After creating the parametrization frequency bands, the user can select which band response to model,

Finally, based on characterization results using the excitation of the intended application, a user can select which bands to parameterize as flat (i.e. frequency independent), and which bands to parameterize as frequency varying. Then the number of unknown parameters and known measurements are tallied. If the problem is under-determined, either more measurements need to be generated or the parametrization needs to be altered.

The outline of the grey-box identification process is as follows:

1. Perform characterization using the excitation corresponding to the intended application.
2. Choose the model identification signal and the nonlinear order.
3. Choose the frequency variation threshold and sort IM frequencies into frequency bands based on linear S-parameter measurements. Then choose frequency bands to model.
4. Choose parametrization for each frequency band.
5. Avoid an under-determined optimization problem by tallying the number of unknown parameters and known measurements. If necessary, modify the identification signal (step 2) or the parametrization (step 4).

6. Perform characterization using the model identification signal.

7. Apply grey-box identification algorithm.

The above ideas will be described in detail first for the single RF carrier case, then for the two RF carrier case. Generalization can be made for the case with multiple RF carriers. This will show the interdependence between model identification signal design, parametrization of the model parameters, and the fidelity of the identified model.

One RF Carrier

When the excitation of the intended application is an unmodulated RF carrier, the bandwidth-dependent memory effect will not be excited, so the same single-tone signal will be used in model identification. If the excitation is a narrowband modulated RF carrier and the characterization results shows bandwidth-dependent memory effect, then a narrowband two-tone signal will be used in model identification. (This corresponds to the case-study in Section 4.3). The maximum tone-spacing is set to match the bandwidth of the modulation. Suppose the two-tone signal is centered at 2 GHz with a tone-spacing of 60 MHz, then the IM response up to the third-order are centered about 2 GHz harmonics.

Suppose the transmission response of an amplifier is modeled, its linear S_{21} measurement can be used to sort the IM frequencies into bands for parametrization. The resulting parametrization frequency bands are shown in Fig. 6.2. First the magnitude of the S_{21} measurement is plotted as a solid black line in the figure. Then at each IM response frequency, the nonlinear order in which the IM response first appear is plotted as a solid red stem line with circular marker. Based on the variation of the S_{21} magnitude, the IM response frequencies are sorted into frequency bands. For each band, the starting frequency is marked by a dotted blue stem line with right-pointing triangular marker, and the stoping frequency is marked by a dotted blue stem line with left-pointing triangular marker. In particular, the baseband is between 0 to 60 MHz. As reflected in the S_{21} measurement, the baseband typically encounters 40 dB or more variation in frequency response, while the harmonic bands typically encounters variation of a few dB

in frequency response. The different variation in frequency response encountered in different frequency bands translates into different model parametrization in each band. Typically the linear model parameters are parameterized by the user as being frequency varying in baseband and constants in harmonic bands. For the case here where the excitation consists of one RF carrier, the parametrization frequency bands coincide with the harmonic frequency bands. This step may seem obvious and redundant, but its necessity will become apparent as two or more RF carriers are considered.

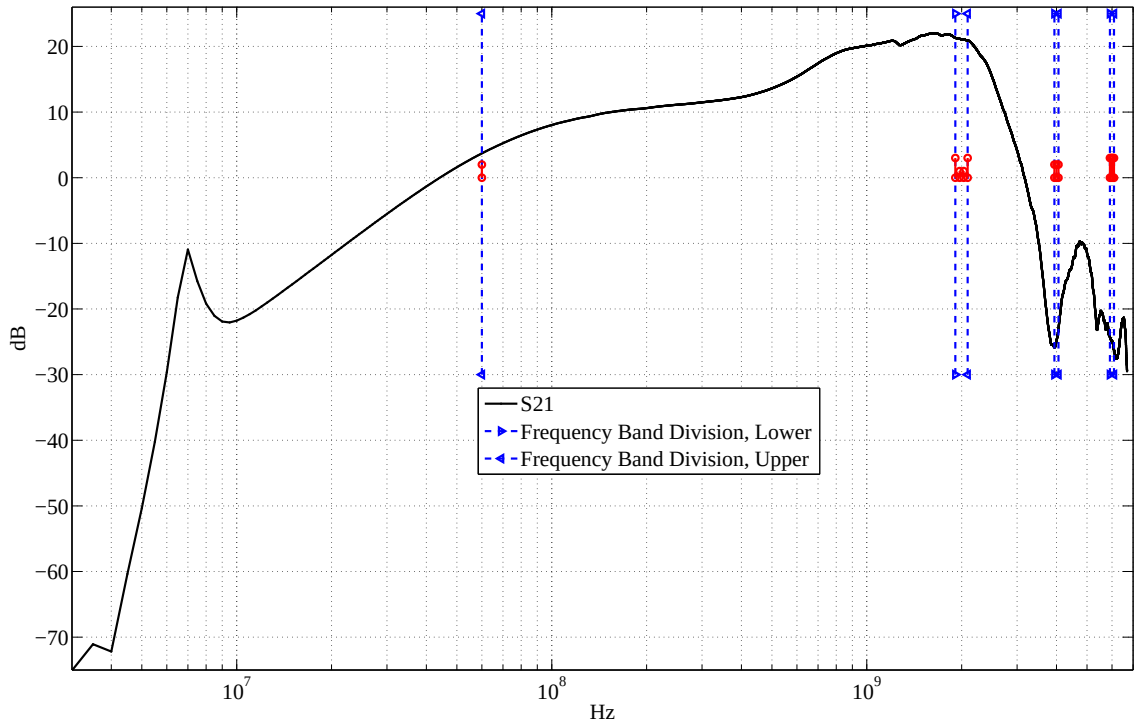


Figure 6.2: In model identification using a narrowband two-tone signal (centered at 2 GHz with 60 MHz tone-spacing), nonlinear response up to the third-order are sorted into bands for parametrization. Sorting is based on the magnitude of the S_{21} measurement of an amplifier. IM frequencies with frequency response variation below a user set threshold (2 dB here) are grouped into the same band.

From the S_{21} measurement in Fig. 6.2, it is observed that responses at harmonic bands

higher than the first are greatly attenuated. In the grey-box model identification algorithm, IM response dominated by the third-order contribution are used to estimate the linear parameters. Typically IM response are purely third-order only at power levels 40 dBc below the linear frequency output. If the IM response are attenuated by additional 40 dB as is the case in Fig. 6.2, a NVNA's approximately 80 dB dynamic range will be reached.

Suppose that only the response in the first harmonic band will be modeled. There are many nonlinear contributions to each response, all of which excite the filtering elements in the model. To proceed with parametrization of the linear model parameters, the contributing nonlinear response are determined using the nonlinear current source information derived in Section 5.2.2. For IM response at frequencies f_1, f_2, $2f_1 - f_2$, and $2f_2 - f_1$, Table 6.1 shows the contributions up to the third-order. The numeric value of the contributing f_{IM} is listed in the left most column. For each f_{IM} being modeled, the lowest order of each contributing f_{IM} is listed. The dependence on the contributing f_{IM} is marked in the right most column.

Table 6.1: In model identification using a narrowband two-tone signal (centered at 2 GHz with 60 MHz tone-spacing), the contributing f_{IM} are listed for IM response being modeled at frequencies f_1, f_2, $2f_1 - f_2$, and $2f_2 - f_1$.

Band	Frequencies in band	Contributing f_{IM}	f_{IM} being modeled				Dependence
			f_1	f_2	$2f_1 - f_2$	$2f_2 - f_1$	
1	0	0	2	2			$\surd$
	6E7	$f_2 - f_1$	2	2	2	2	$\surd$
2	1.97E9	f_1	1	1	1	1	$\surd$
	2.03E9	f_2	1	1	1	1	$\surd$
	1.91E9	$2f_1 - f_2$			3		$\surd$
	2.09E9	$2f_2 - f_1$				3	$\surd$
3	3.94E9	$2f_1$	2		2		$\surd$
	4E9	$f_1 + f_2$	2	2			$\surd$
	4.06E9	$2f_2$		2		2	$\surd$
4	5.91E9	$3f_1$					
	5.97E9	$2f_1 + f_2$					
	6.03E9	$f_1 + 2f_2$					
	6.09E9	$3f_2$					

After determining dependence on the contributing f_{IM}, the number of unknown model parameters and known constraining measurements can be tallied to avoid an under-determined optimization problem. The unknown model parameters are the nonlinearity coefficients denoted as g_{NL}'s, the frequency independent linear parameters denoted as R's, and the frequency-dependent linear parameters denoted as Z's. User decision is made in parameterizing Z's in frequency bands which contribute to response at f_{IM} being modeled. For frequency bands which user deem as inconsequential in causing bandwidth-dependent behavior, the Z's are parameterized as constants. Otherwise, the Z's are parameterized as a frequency varying, and certain number of frequency points are assigned to those frequency bands.

For example, baseband effects are responsible for nonlinear memory behavior, so Z's in band 1 are parameterized as frequency varying, while Z's in all other bands are parameterized as constant. The frequency points assigned to band 1 are chosen by the user to capture features observed in DUT characterization, such as sudden changes in amplitude or phase. To calculate

the response being modeled, Z's at the contributing f_{IM} in each band are needed, and can be interpolated from the optimizer estimated Z's at the chosen frequency points. Thus the number of unknown model parameters is

$$\begin{aligned}\text{Number Of Unknowns} = \quad & \text{SizeOf}\big(\ \{g_{\mathrm{NL}}\} \bigcup \{R\} \\ & \bigcup \{Z(\text{user select frequency points in contributing bands})\}\ \big)\end{aligned} \tag{6.1}$$

The known constraints are measurements at f_{IM} being modeled, which need to be slightly larger in number than the unknown parameters. For the identification algorithm step that estimates the linear model parameters in Fig. 4.5(b), the measurement data used are response at one power level. Since there may be a large number of Z's, the tone-spacing is swept to generate additional constraints. The number of constraining measurements is

$$\text{Number Of Constraints} = \text{SizeOf}\big(f_{\mathrm{IM,Modeled}}\big) \times \text{SizeOf}\big(\text{Tone-Spacing Swept}\big) \tag{6.2}$$

The tone-spacings are also chosen by the user to capture features observed in DUT characterization, and are often similar in number to the frequency points chosen in parametrization. This limits the number of Z's that can be uniquely identified.

For the identification algorithm step that estimates the nonlinear model parameters in Fig. 4.5(b), the measurement data used are response over swept power levels, which generates more constraints. In addition the number of nonlinear model parameters is smaller than the linear parameters. Thus the optimization problem at this estimation step is almost always an over-determined one.

The above tallying calculation assumes that responses at all contributing f_{IM} are generated in the model, and these responses with excite the filtering networks in the model. This assumption does not hold for all models. For example, the model may only have odd-degree nonlinear coefficients, or the filtering network is not excited by nonlinear responses which is the case for the input filter of the Weiner-Hammerstein model. Nevertheless the above tallying

calculation provides an upper-bound on the number of measurements required, in the sense that it always avoids an under-determined optimization problem, although more measurements are required than are necessary for the particular model. More detailed tallying calculation can be implemented for each model to account for situations that fail the assumption, and provide a tighter bound on the number of measurements required.

Two RF Carriers

When the excitation of the intended application is a wideband signal consisting of two unmodulated RF carriers, bandwidth-dependent memory effect will not be excited, so the same excitation will be used in model identification. Compared to the single modulated RF carrier case, the spacing between the two carriers are on the order of the carrier frequencies. Suppose the two carriers are at 500 MHz and 900 MHz, IM responses are no longer clustered in certain frequency bands as in the single carrier case.

Suppose the transmission response of an amplifier is modeled, its linear S_{21} measurement can be used to sort the IM frequencies into bands for parametrization. The resulting parametrization frequency bands are shown in Fig. 6.3. First the magnitude of the S_{21} measurement is plotted as a solid black line in the figure. Then at each IM response frequency, the nonlinear order in which the IM response first appear is plotted as a solid red stem line with circular marker. Based on the variation of the S_{21} magnitude, the IM response frequencies are sorted into frequency bands. For each band, the starting frequency is marked by a dotted blue stem line with right-pointing triangular marker, and the stoping frequency is marked by a dotted blue stem line with left-pointing triangular marker.

Suppose that all responses except the last two highest spectral components will be modeled. To proceed with parametrization of the linear model parameters, the contributing nonlinear response are determined using the nonlinear current source information derived in Section 5.2.2. Table 6.2 displays the contributions up to the third-order. The numeric value of the contributing f_{IM} is listed in the left most column. For each f_{IM} being modeled, the lowest order

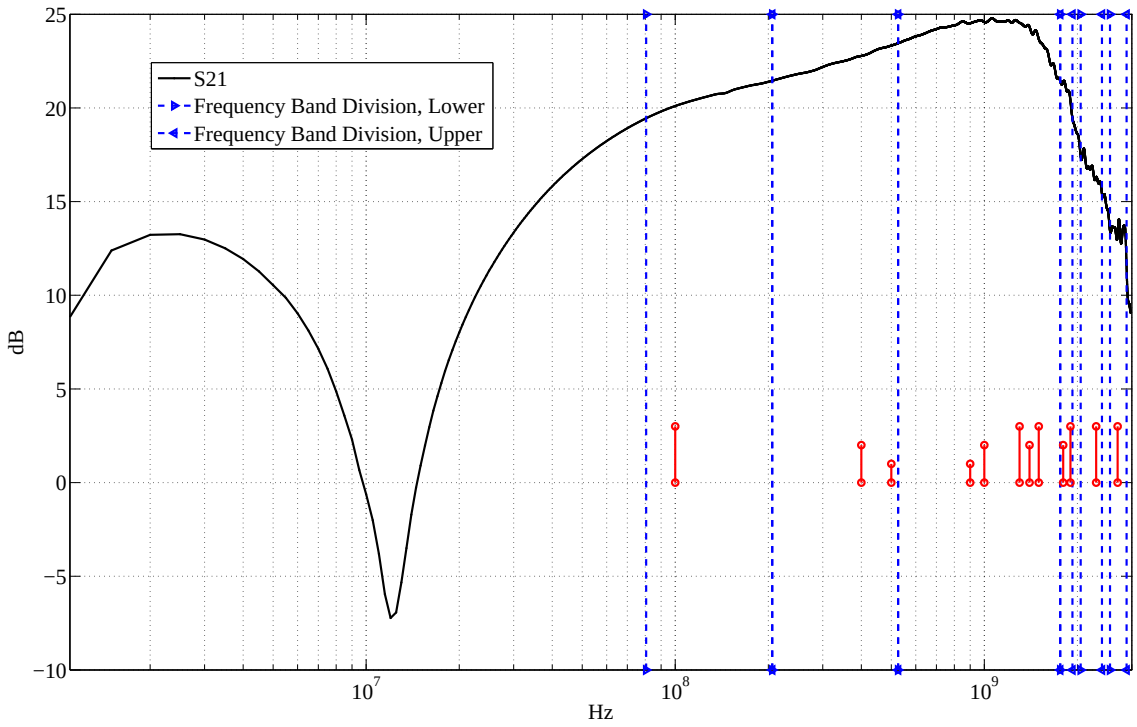


Figure 6.3: In model identification using a broadband two-tone signal (f_1= 500 MHz and f_2= 900 MHz), nonlinear response up to the third-order are sorted into bands for parametrization. Sorting is based on the magnitude of the S_{21} measurement of an amplifier. IM frequencies with frequency response variation below an user set threshold (2 dB here) are grouped into the same band.

of each contributing f_{IM} is listed. The dependence on the contributing f_{IM} is marked in the right most column.

After determining dependence on the contributing f_{IM}, the number of unknown model parameters and known constraining measurements can be tallied to avoid an under-determined optimization problem. User decision is made in parameterizing Z's in frequency bands which contribute to response at f_{IM} being modeled. Since the unmodulated RF carriers cannot excite bandwidth-dependent memory behavior, the Z's are parameterized as constants.

The known constraints are the measurements which are at f_{IM} frequencies being modeled, and needs to be slightly larger in number than the unknown parameters. For the identification algorithm step that estimates the linear model parameters in Fig. 4.5(b), the measurement data used are response at one power level. Since there may be a large number of Z's, the carrier frequencies are swept, with the parametrization frequency bands, to generate additional constraints.

For the case considered in Fig. 6.3, there is no baseband frequencies. This is because the excitation consists of two unmodulated RF carriers. If any of the RF carriers is modulated and if the bandwidth-dependent memory effect is to be modeled, the model identification signal is modified by placing a narrowband two-tone signal at that carrier frequency. Using a three-tone or a four-stone signal, the generated IM frequencies are sorted into bands as before. Compared to the unmodulated RF carrier case, baseband frequencies will be present and additional IM frequencies will cluster around each IM frequencies in Fig. 6.3. As reflected in the S_{21} measurement, baseband wider than 20 MHz will encounter 20 dB or more variation in frequency response, while the higher frequency bands typically encounters variation of a few dB in frequency response. The different variation in frequency response encountered in different frequency bands translates into different model parametrization in each band. Typically the linear model parameters are parameterized by the user as being frequency varying in baseband and constants in higher frequency bands. For parametrization, the contributing nonlinear response are determined in a way similar to that used to produce Table 6.2. Then the number

Table 6.2: In model identification using a broadband two-tone signal (f_1= 500 MHz and f_2= 900 MHz), the contributing f_{IM} frequencies are listed for IM response being modeled at frequencies f_1, f_2, $2f_1 - f_2$, and $2f_2 - f_1$.

Band	Freq. in band	Contrib. f_{IM}	f_{IM} being modeled										Dep.
			$2f_1 - f_2$	f_1	$f_2 - f_1$	f_2	$2f_1$	$f_1 + f_2$	$2f_2 - f_1$	$3f_1$	$2f_2$	$2f_1 + f_2$	
1	0	0		2		2							√
2	1E8	$2f_1 - f_2$	3										√
3	5E8	f_1	1	1	1	1	1	1	1	1		1	√
	4E8	$f_2 - f_1$	2	2	2	2			2				√
4	9E8	f_2	1	1	1	1		1	1		1	1	√
	1E9	$2f_1$	2	2			2			2		2	√
	1.4E9	$f_1 + f_2$		2		2		2				2	√
	1.3E9	$2f_2 - f_1$							3				√
	1.5E9	$3f_1$								3			√
5	1.8E9	$2f_2$				2			2		2		√
	1.9E9	$2f_1 + f_2$										3	√
6	2.3E9	$f_1 + 2f_2$											
7	2.7E9	$3f_2$											

of unknowns and constraints in the optimization problem are tallied. Finally the tone-spacings can be varied to generate additional constraints.

Multiple RF Carriers

When the excitation of the intended application consists of more than two modulated or unmodulated RF carriers, the model identification signal design and parametrization can be generalized from the two RF carrier case. The model identification signal is a multi-tone signal which can also be applied in model identification in cases where the excitation of the intended application is not known.

If the excitation of the intended application is not known, the model must be able to replicate response under any type of excitation applied within a specified frequency range, which in the worst case may be an ultra wide band (UWB) signal. The tones in the model identification signal should be placed to capture certain frequency varying behavior observed in DUT characterization. Also the tone frequencies must be chosen so as not to generate coincident IM response frequencies, because cancelation effects entangle the model parameters and pose difficulties in parameter estimation. This can only be achieved for IM responses up to a certain nonlinear order.

In [74] the Wiener-Hammerstein model was identified using a broadband sparse odd-multisine signal. The achievable model fidelity depends on the minimum tone-spacing which is limited by the need to leave room in the spectrum for certain IM products, and by frequency resolution of the measurement system.

6.3 Case-Study

Section 6.2 presented an approach to model the antenna-to-antenna coupling and the nonlinear RF front-end circuits for cosite simulation. This approach is applied and verified here for a cosite interference scenario shown in Fig. 6.4, which consists of two transmitters and one receiver. The aim is to show that detailed spectral features, which was not considered in past

cosite interference investigation and cannot be replicated by other modeling approaches, can be modeled using the chosen approach.

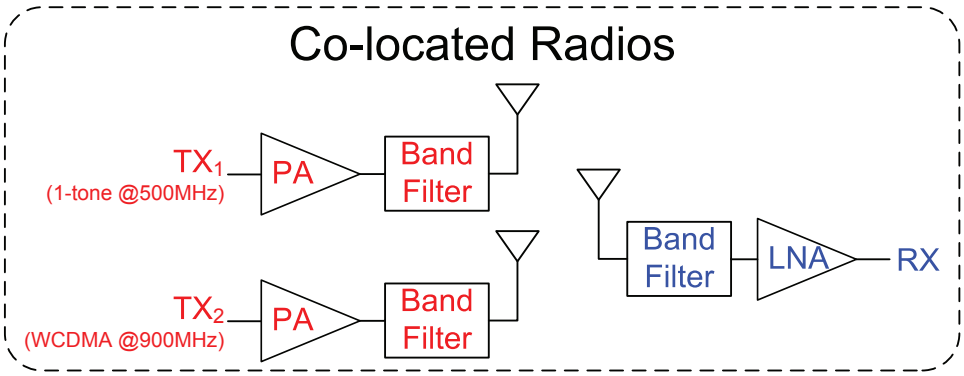


Figure 6.4: Block diagram of the cosite interference scenario under study.

Three types of characterization were performed, first using VNA to measure the linear frequency response, then using single-tone signals at both carriers, and finally using a WCDMA signal at the 900 MHz carrier. Based on the characterization results, models and identification signals are selected. The modeled behavior is then compared with measured results.

6.3.1 Measurement Setup

The carriers of the two transmitters are at 500 MHz and 900 MHz. Realistic antennas links have bandpass characteristics, so explicit band filters will not be included in the test setup. However additional narrowband filters are placed at each transmitter output to limit the spectral response to within several modulation bandwidth around each carrier. This is because in modeling the receiver, the circuit simulator used in the grey-box model identification procedure is limited in the number of tones supported in the excitation. The current implementation of Volterra circuit simulator based on the direct calculation method, only supports two or three tones. So if harmonic components present in the transmitter output are not eliminated, model prediction accuracy is degraded.

Linear S-Parameter Measurement

First the antenna-to-antenna coupling is characterized. The antennas are placed in an equilateral triangle pattern (60 cm on each side) in an anechoic chamber. The measured transmission response between each of the transmitter antennas and the receiver antenna are shown in Fig. 6.5. There is approximately −50 dB coupling at 500 MHz and 900 MHz.

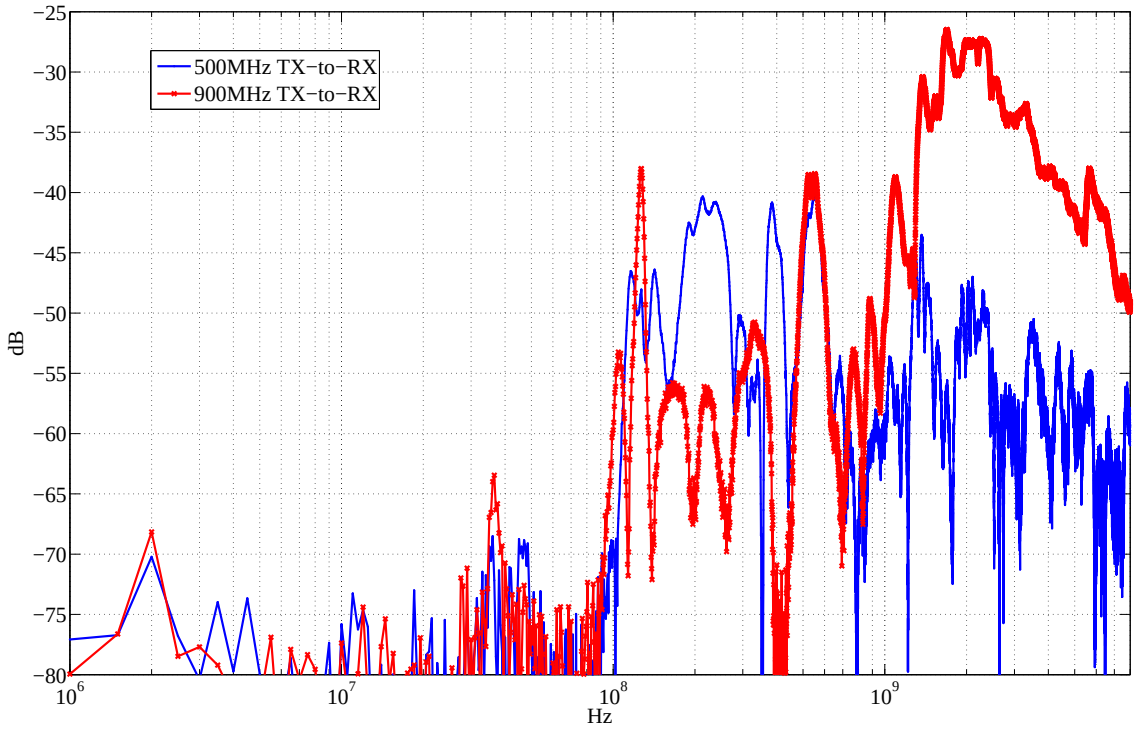


Figure 6.5: Antenna-to-antenna coupling between each of the transmitter antennas and the receiver antenna from linear S-parameter measurements.

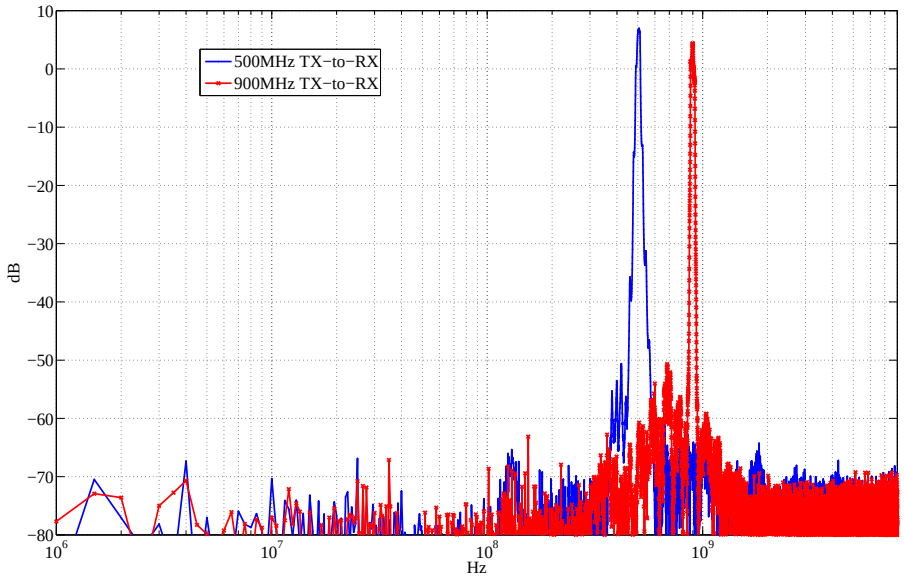
Based on the antenna measurement results, circuit components of the test setup are chosen to transmit enough power through the antennas, so that the receiver generated IM response has enough power to be measured by the VNA. The two-tone characterization will verify the allocated link budget. Connecting the antennas to the selected circuit components outside

the chamber, the transmission response between each of the transmitters and the receiver are measured, as shown in Fig. 6.6(a). The transmission is mainly determined by the narrowband filter at each transmitter's output. The reflection response at the input of each transmitter and the output of the receiver are shown in Fig. 6.6(b). The transmitter and receiver amplifiers are wideband and have small reflections.

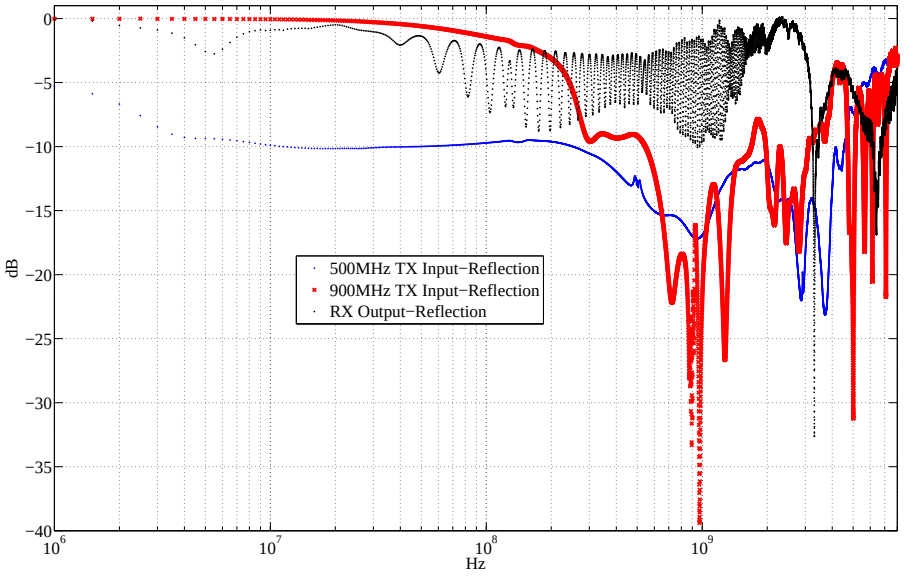
Measurement with Unmodulated RF Carriers

For characterization using CW signals at both RF carriers, the measurement setup is shown in Fig. 6.7. For absolute phase measurement, an instrument with six receiver is required to capture the incident and reflected power waves at all three ports. With a 4-port VNA, mismatch calibration cannot be performed at each port. The incident power to the two transmitters and the output power from the receiver are measured and calibrated to the measurement planes AA, BB and CC. Using the phase-locked signal generator idea from Section 3.3, the relative phase can be measured by connecting the reference signal generator output to the reference receiver in the VNA.

The power of the incident 500 MHz CW signal ($P1$) is calibrated to the measurement plane AA and is swept from -12.25 dBm to -2.25 dBm in 2.5 dB steps. The power of the incident 900 MHz CW signal ($P2$) is calibrated to the measurement plane BB and is swept from -12.5 dBm to -2.5 dBm in 2.5 dB steps. The power of the transmitted IM responses is calibrated to the measurement plane CC and depends on three independent variables, namely the IM frequency, $P1$ and $P2$. When plotting the power of the transmitted IM responses in a 3D plot, two of the independent variables, IM frequency and $P2$, are used as the coordinate axis. The third independent variable, $P1$, is used as a parameter. Fig. 6.8 is the parametric plot of all IM responses of order less than or equal to three. The power of the transmitted IM responses are plotted using solid lines. The power of the incident CW signal at 500 MHz and 900 MHz are plotted using dashed lines. Lines with different parametric $P1$ values are



(a)



(b)

Figure 6.6: Linear S-parameter measurement: (a) transmission response between each of the transmitters and the receiver; and (b) reflection at the input of each transmitter and the output of the receiver.

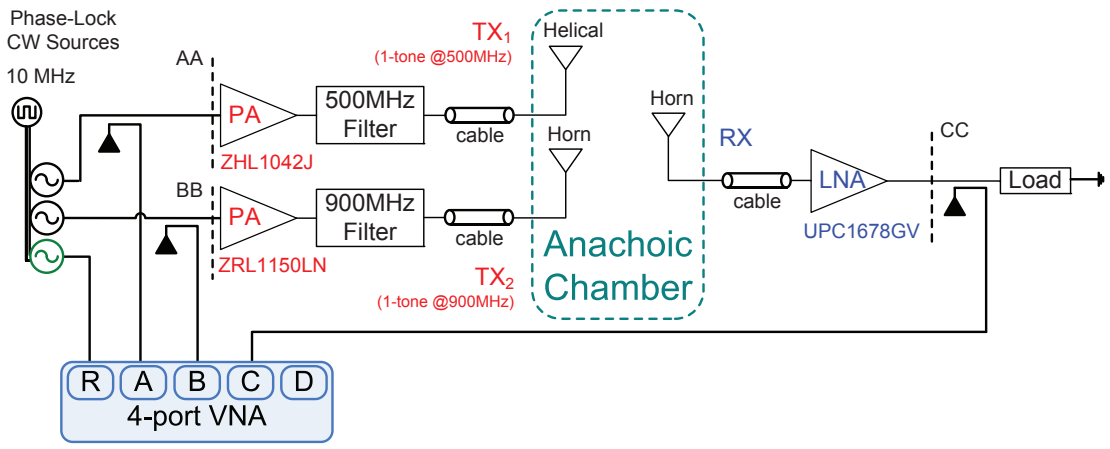


Figure 6.7: Block diagram of the cosite interference measurement step for the case where a single tone is transmitted at 500 MHz, and another single tone is transmitted at 900 MHz.

differentiated by color. In a similar way, Fig. 6.9 is the parametric plot of all fifth-order IM responses. Almost all fifth-order IM responses have low power levels, and are unreliable for model extraction purposes.

Measurement with WCDMA Signal

For characterization using a single tone at 500 MHz and a WCDMA signal at 900 MHz, the measurement setup is shown in Fig. 6.10. The incident power to the 500 MHz transmitter is measured by a VNA. The incident power to the 900 MHz transmitter is measured by a power meter. The output from the receiver is measured in a VSA. All power measurements are calibrated to the measurement planes AA, BB and CC.

Phase measurement of a modulated signal is difficult. When the frequency content of a modulated signal is not known, the FFT operation will generally result in spectral leakage. Thus modulated signals are best characterized in the time domain. A VSA receiver's mismatch errors can be corrected using a VNA, and the receiver's phase dispersion can be corrected using the techniques described in Section 3.2. However the remaining difficulty is the lack of time-base synchronization of the vector signal generator with other CW sources and with the

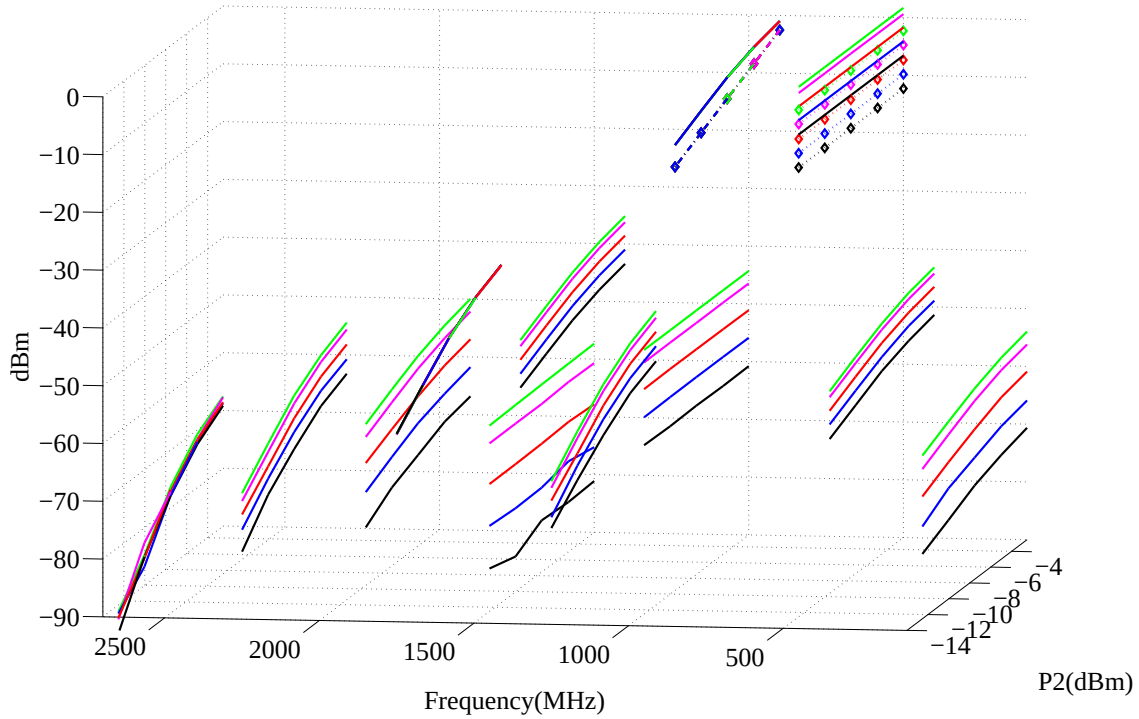


Figure 6.8: Measurement results from the test setup shown in Fig. 6.7. The power of the incident 500 MHz CW signal ($P1$) at the measurement plane AA is swept from -12.25 dBm to -2.25 dBm in 2.5 dB steps. The power of the incident 900 MHz CW signal ($P2$) at the measurement plane BB is swept from -12.5 dBm to -2.5 dBm in 2.5 dB steps. The power of the transmitted IM responses at the measurement plane CC depends on three independent variables, namely the IM frequency, $P1$ and $P2$. Two of the independent variables, IM frequency and $P2$, are used as the coordinate axis. The third independent variable, $P1$, is used as a parameter. The power of the transmitted IM responses with order less than or equal to three are plotted using solid lines. The power of the incident CW signal at 500 MHz and 900 MHz are plotted using dashed lines. Lines with different parametric $P1$ values are differentiated by color.

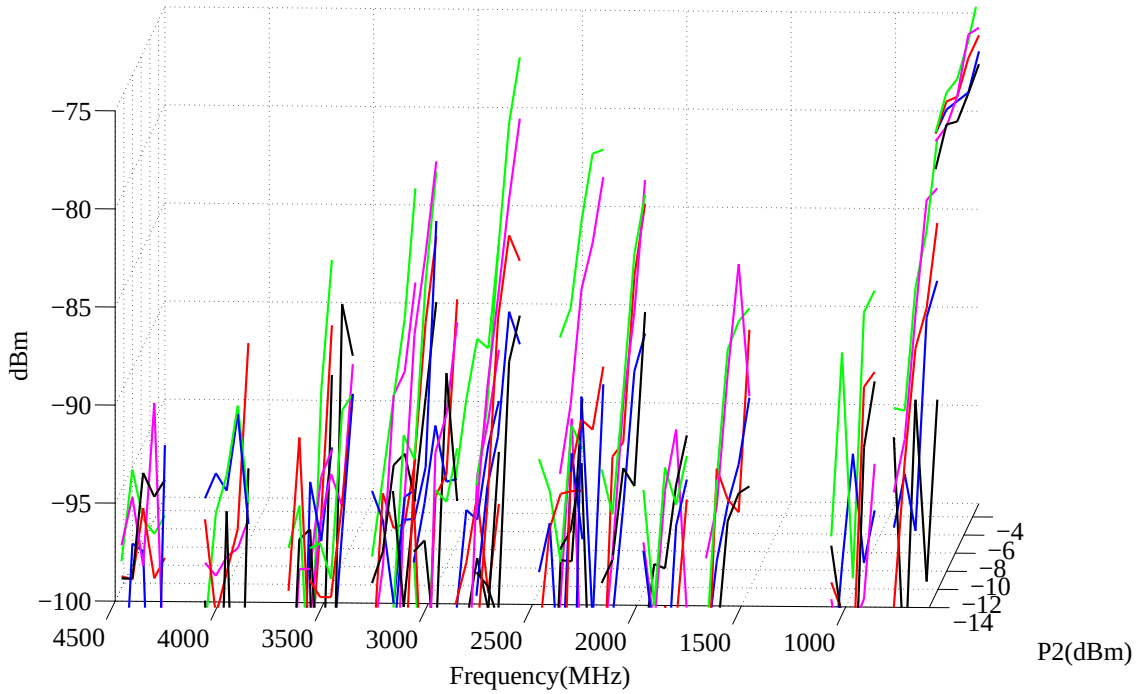


Figure 6.9: Measurement results from the test setup shown in Fig. 6.7. The power of the incident 500 MHz CW signal ($P1$) at the measurement plane AA is swept from -12.25 dBm to -2.25 dBm in 2.5 dB steps. The power of the incident 900 MHz CW signal ($P2$) at the measurement plane BB is swept from -12.5 dBm to -2.5 dBm in 2.5 dB steps. The power of the transmitted IM responses at the measurement plane CC depends on three independent variables, namely the IM frequency, $P1$ and $P2$. Two of the independent variables, IM frequency and $P2$, are used as the coordinate axis. The third independent variable, $P1$, is used as a parameter. The power of the transmitted fifth-order IM responses are plotted using solid lines. Lines with different parametric $P1$ values are differentiated by color.

VSA receiver, which are often frequency-locked instead of phase-locked. In the end, except in simulation environments, measured phase or timing information for modulated signal often can not be used in model verification. Only the power spectrum will be verified in Section 6.3.2.

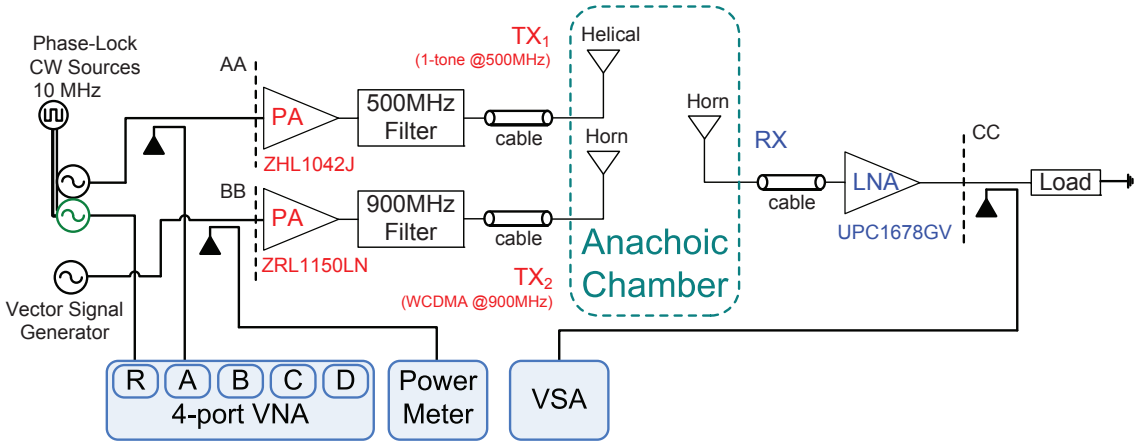
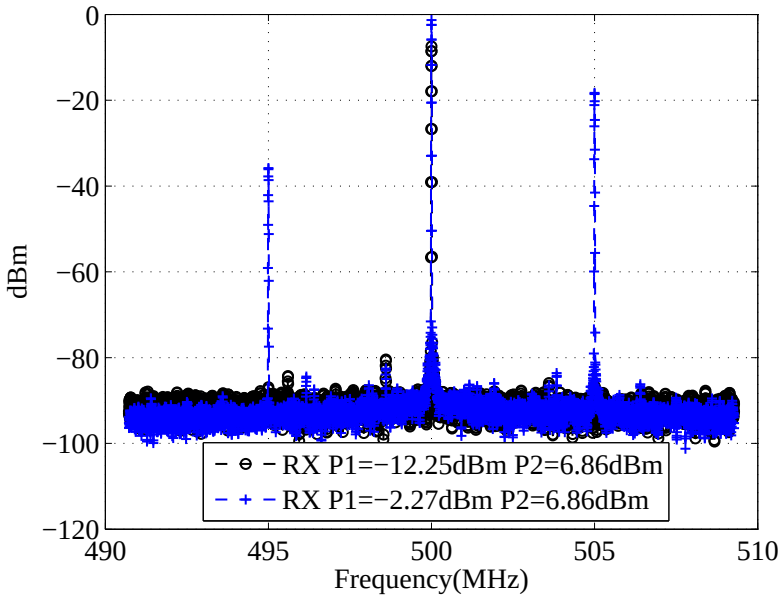
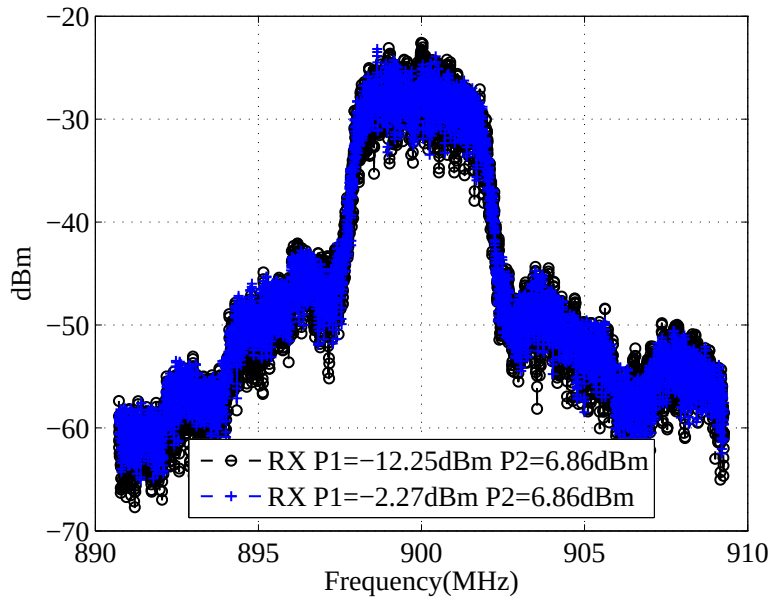


Figure 6.10: Block diagram of the cosite interference measurement step for the case where a single tone is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz.

The power of the single-tone signal at 500 MHz ($P1$) is swept from -12.25 dBm to -2.25 dBm in 2.5 dB steps. The power of the WCDMA signal at 900 MHz ($P2$) is swept from -3 dBm to 7 dBm in 2.5 dB steps. The power spectrum of linear responses are shown in Fig. 6.11. The power spectrum of second-order IM responses are shown in Fig. 6.12 and Fig. 6.13. The power spectrum of third-order IM responses are shown in Fig. 6.14, 6.15 and 6.16. Based on the previous two-tone measurement results, all fifth-order response have low power levels and are not measured.

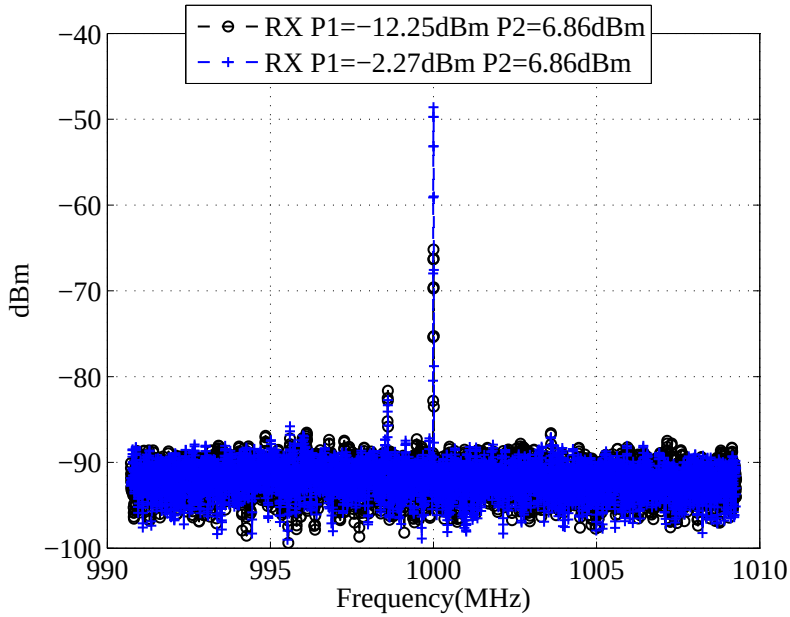


(a)

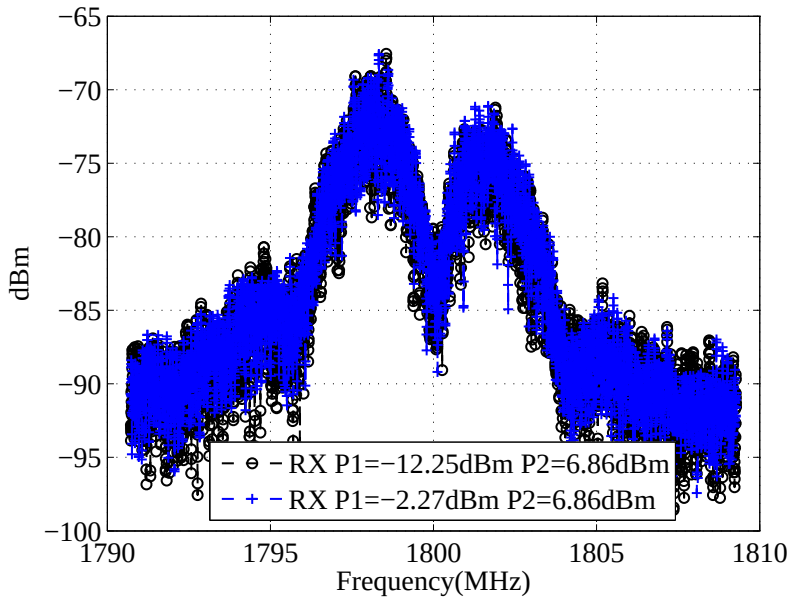


(b)

Figure 6.11: Power spectrum of the linear responses at the receiver output centered at: (a) f_1; and (b) f_2. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz.

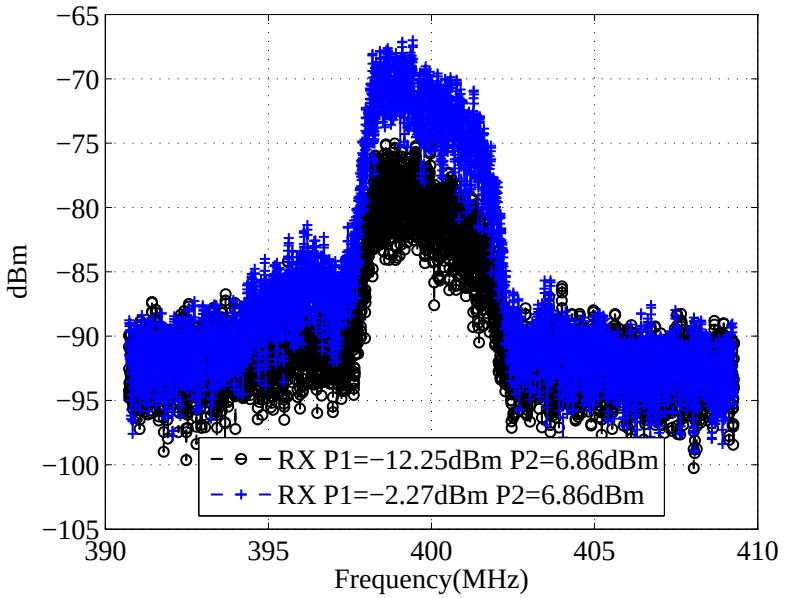


(a)

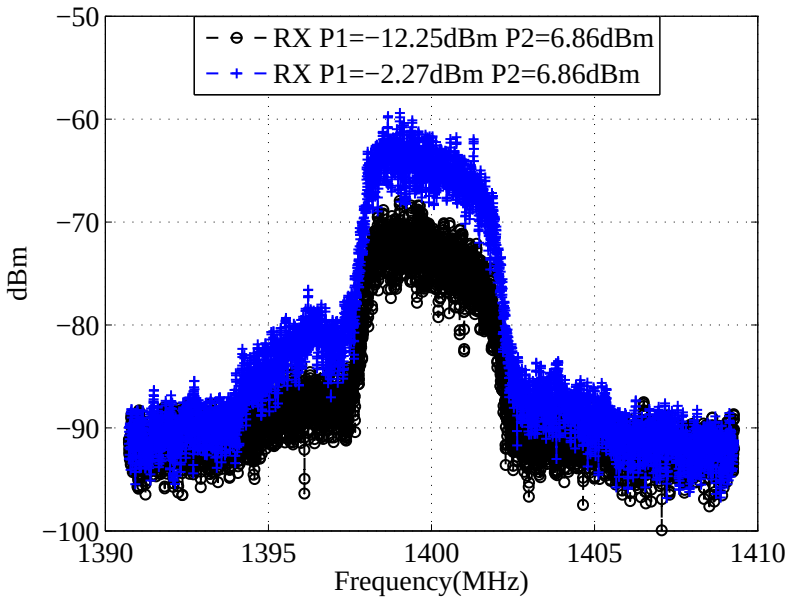


(b)

Figure 6.12: Power spectrum of the second-order IM responses at the receiver output centered at: (a) $2f_1$; and (b) $2f_2$. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz.

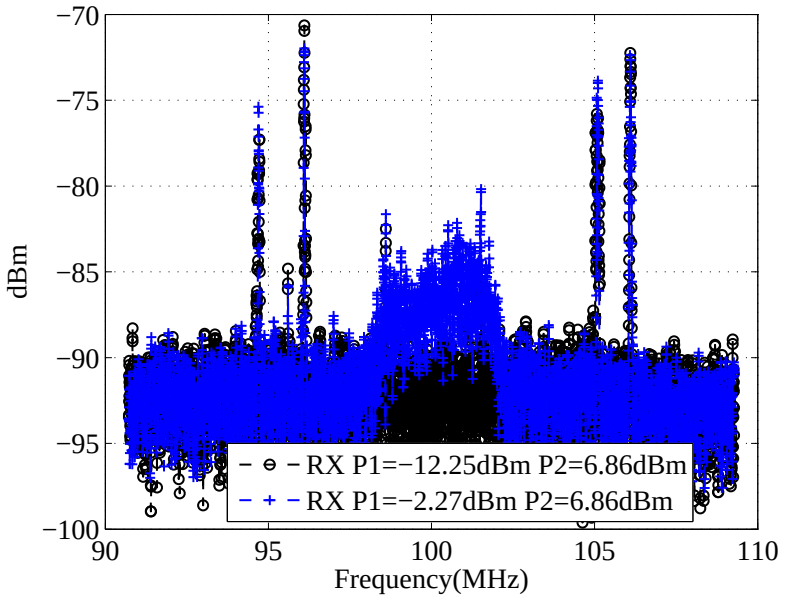


(a)

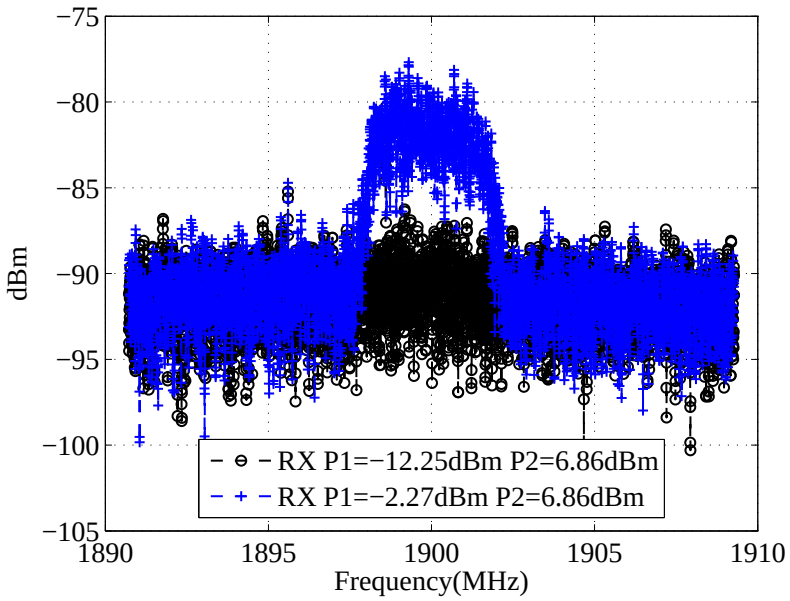


(b)

Figure 6.13: Power spectrum of the second-order IM responses at the receiver output centered at: (a) $f_2 - f_1$; and (b) $f_2 + f_1$. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz.

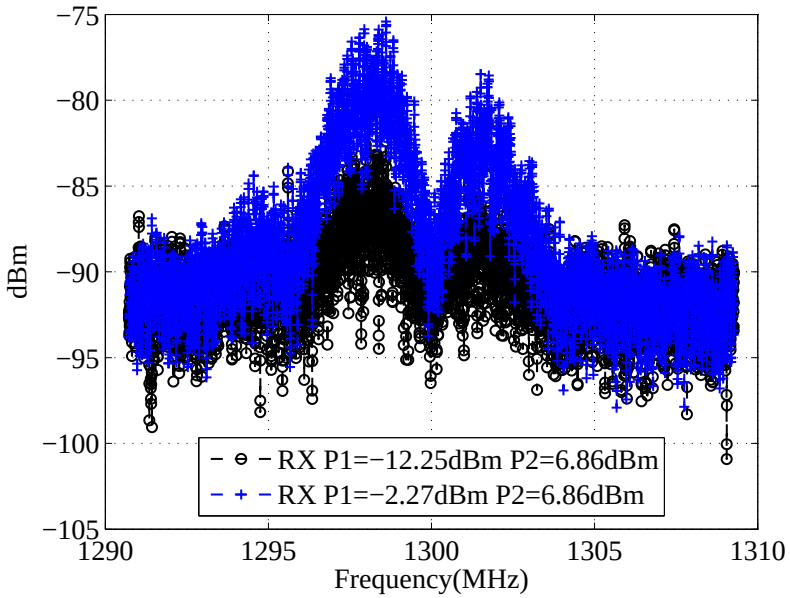


(a)

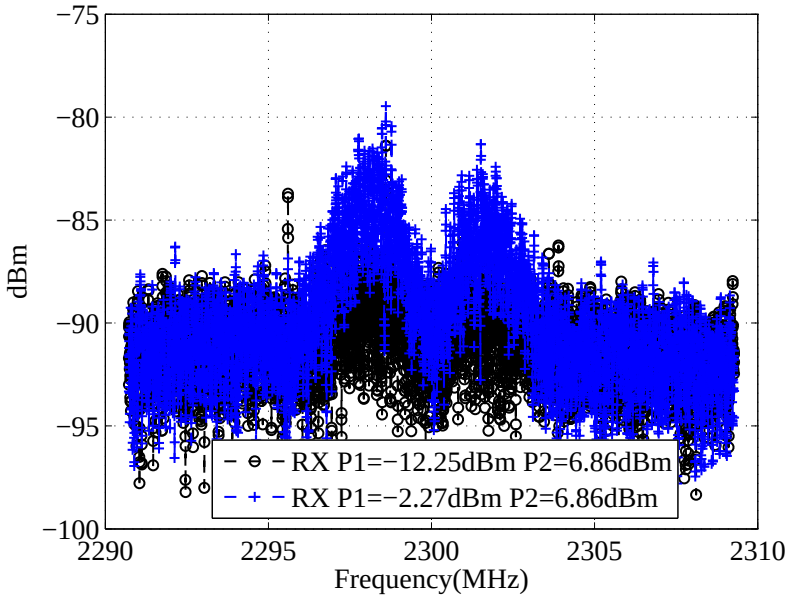


(b)

Figure 6.14: Power spectrum of the third-order IM responses at the receiver output centered at: (a) $2f_1 - f_2$; and (b) $2f_1 + f_2$. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz.

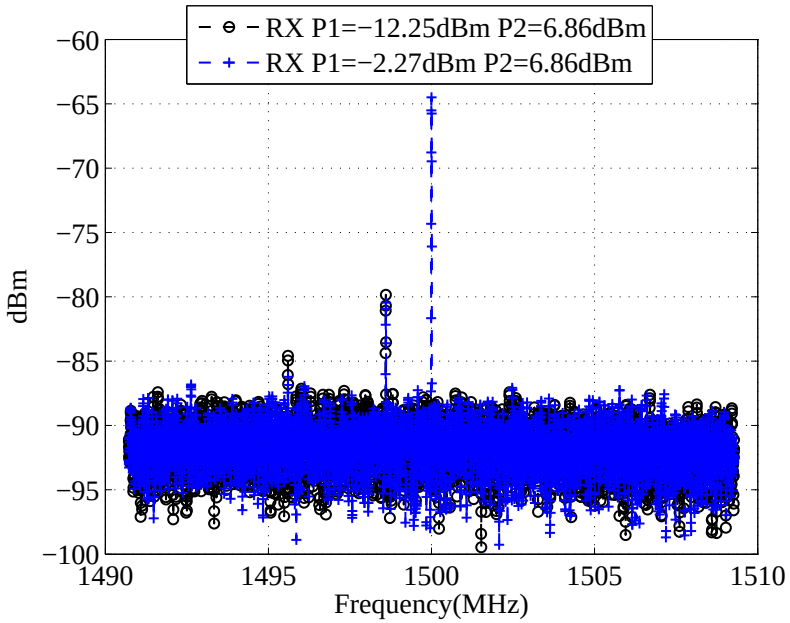


(a)

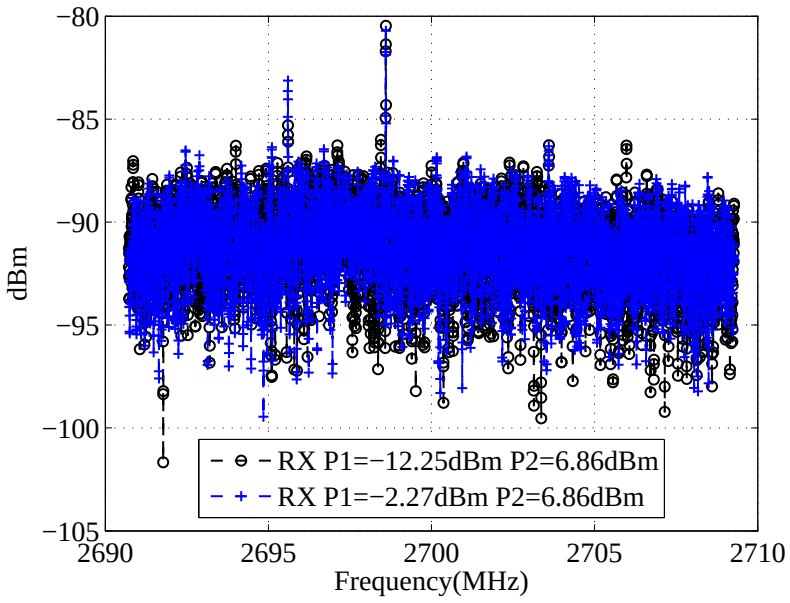


(b)

Figure 6.15: Power spectrum of the third-order IM responses at the receiver output centered at: (a) $2f_2 - f_1$; and (b) $2f_2 + f_1$. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz.



(a)



(b)

Figure 6.16: Power spectrum of the third-order IM responses at the receiver output centered at: (a) $3f_1$; and (b) $3f_2$. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz.

6.3.2 Modeling Results

The methods described in Section 6.2 are applied to model each component of the experimental setup. In particular, methods of Section 6.2.2 are used in modeling the receiver amplifier. To model the bandwidth-dependent memory effect, a three-tone model identification signal was used, with a single tone at 500 MHz and a narrowband two-tone at 900 MHz.

A particular feature observed during characterization using a WCDMA signal, is the notch in the power spectrum for IM responses centered at $2f_2$, $2f_1 - f_2$, and $2f_1 + f_2$. The notch is exactly at the center of the spectrum in every case. Such a notch will be generated if there are filtering networks in the circuit with a high-pass baseband response. This high-pass baseband response was observed in the linear S-parameter measurement of the receiver amplifier in Fig. 6.3.

A behavioral model with build-in feedback was selected from Section 4.3.1. The linear parameters at baseband are parameterized as frequency varying, and constants in all other bands. The simulated result is compared with the measurement in Fig. 6.17 for IM response centered about $2f_2$.

6.4 Conclusions

Past work on cosite interference simulation of co-located radio focused on a formula-based approach, and used simple parametric models. Such an approach becomes in accurate for design verification. The work here focused on a simulation-based approach using data-based behavioral models. The increase in model fidelity makes this approach particularly suited for verification. The grey-box model identification methodology developed in Chapter 4 was applied in modeling a cosite interference scenario. The modeled results replicated the essential features in the measurements.

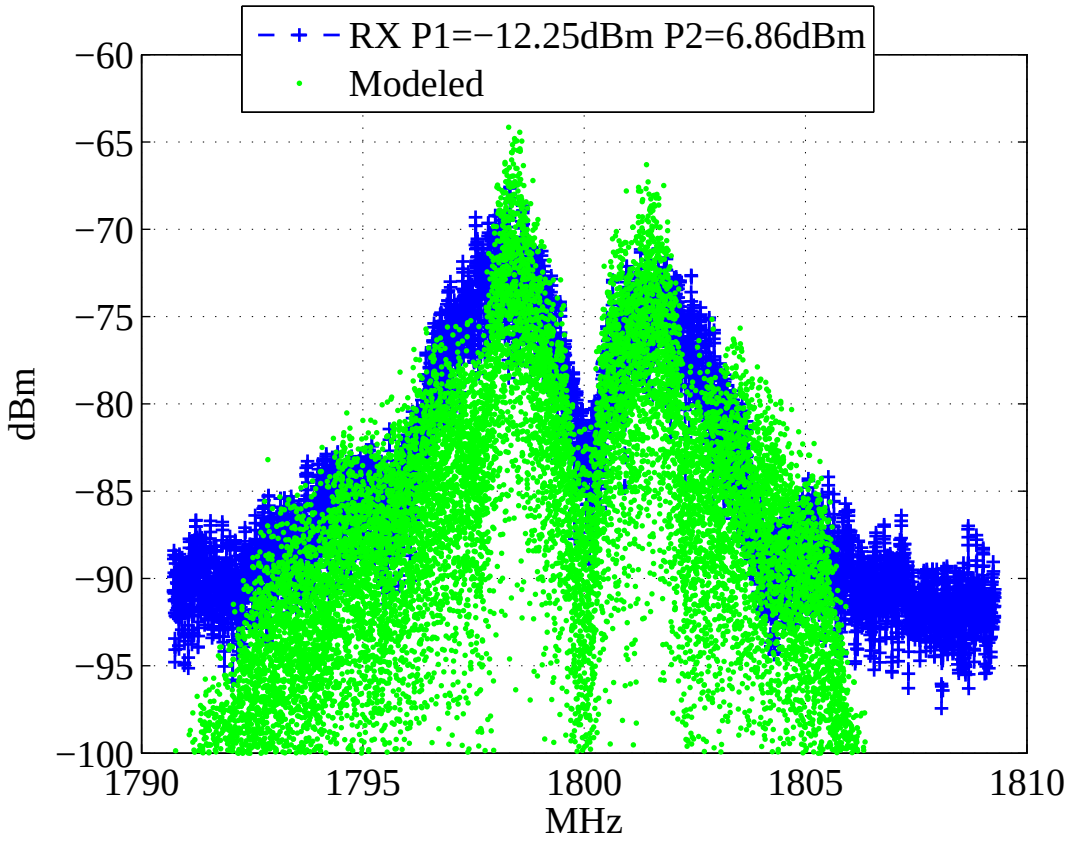


Figure 6.17: Compare simulated versus the measured IM response centered about $2f_2$. A single-tone signal is transmitted at 500 MHz, and a WCDMA signal is transmitted at 900 MHz.

Chapter 7

Conclusions and Future Work

7.1 Summary of Research and Original Contributions

This dissertation presented several results for characterization and modeling of RF nonlinear circuits. Two novel NVNA setups were developed for implementation using common bench top instruments, and do not require a multi-harmonic generator. A general identification methodology was developed for more complex grey-box models of RF nonlinear circuits, that is capable of tracking multiple optimum solutions with minimum user intervention. A recursive algorithm for implementing an improved direct calculation method was developed for Volterra circuit analysis which is capable of handling compression effects lacking in the original work.

A 1-channel receiver has been empirically found to be capable of characterizing a multi-port DUT, by sequentially sampling the DUT signals to mimic a synchronous multi-channel receiver. To maintain the same timing relationship of sequentially sampled waveforms as if they were synchronously sampled, excitation waveforms are designed to be commensurate with the signal generation DAC and the receiver ADC sampler time base. Once the 1-channel VSA can be made to behave like a multi-channel receiver, concepts from absolute phase measurement using broadband receivers can be applied with minor modifications. Namely, a multi-tone signal generated by an AWG replaces the multi-harmonic generator, and a nose-to-nose calibrated

sampling oscilloscope provides the reference measurements for VSA phase error correction. The 1-channel VSA-based NVNA is suitable for characterizing DUT's in-band behavior for narrowband multi-tone excitation. Compared to existing NVNA based on sub-sampling mixer where noise at all sidebands of the sampling frequency harmonics are downconverted to the baseband, the use of a narrowband tuned receiver achieves larger dynamic range.

A novel 4-port VNA-based NVNA was also developed using only common benchtop instruments including phase-locked signal generators and an equivalent-time sampling oscilloscope. The multi-harmonic generator (commonly used with NVNAs) is not required, and the associated tone-spacing limitation in multi-tone testing is removed. Instead a reference signal generator, phase-locked to the multi-tone excitation, is used to provide a phase reference and can tune to tone-spacings on the order of 1 Hz. Also the concept of using a relative phase reference phase-locked to the excitation enables optimum use of different instruments at different power levels during measurement. The high dynamic range heterodyne receiver-based VNA is used for data acquisition during a power sweep, while the low dynamic range sampling oscilloscope is used for its absolute phase calibration ability. Using each instrument to its advantage, narrow tone-spacing, wide bandwidth, and high dynamic range are achieved.

A general identification methodology was developed for more complex grey-box models of RF nonlinear circuits than have been used previously. A general-purpose symbolic Volterra circuit simulator was used to automate the model analysis and response calculation. A generalized higher-level identification algorithm was developed to be compatible with stochastic optimization methods. A hybrid GA optimizer was developed to specifically track local optimum solutions, and present the user with a number of distinct modeling solutions with comparable error metrics. This is particularly suited to incompletely-observed RF systems since initial solution close to the optimums can no longer be easily found for complex models. Instead of a skilled user being involved throughout the model fitting process, user input is directed first at establishing several candidate grey-box models, then establishing a measurement regime, and choosing the best solution at the end of the procedure based on intuition and experience. Now

the suite of standard grey-box models and measurement protocols can be developed by one set of engineers and used by others, perhaps with a different skill set, who are directly involved in applied model identification. The proposed grey-box model identification method developed here was successfully demonstrated in modeling a multi-stage power amplifier with significant long-term memory effects.

An improved direct calculation method was developed for Volterra circuit analysis and is capable of handling compression effects lacking in the original work. One recursive algorithm was developed to generate nonlinear current expressions under multi-tone excitations for various types of nonlinearity sources, and can handle arbitrary degrees of nonlinearity. Another recursive circuit analysis algorithm was developed to generate the nodal voltage IM responses under multi-tone excitations. This makes use of the nonlinear current expressions stored in a look-up data structure. The proposed direct calculation method is then implemented as a symbolic Volterra circuit simulator in Matlab. Typically in RF model identification schemes, the Volterra analysis portion is hand-crafted for the particular model being developed. The new symbolic evaluator is suitable for more generalized model fitting, again eliminating user tasks that can be automated.

7.2 Future Research

There are several ways to build upon the results presented in this work. Chapters 3 through 5 each provide a framework from which new research initiatives may be drawn.

In mimicking a synchronous multi-channel receiver using a 1-channel receiver in Chapter 3, the receiver ADC sampler time base was used to maintain the timing relation between sequentially sampled waveforms. It has been found empirically that multiple measurements needs to be averaged to obtain sufficiently accurate results. Due to the use of a much lower frequency ADC sampler time base to measure phase of signals at RF frequencies, the accumulated jitter between each sequentially sampled waveforms leads to a noise like variance in the measured phase. This measurement noise is zero in the mean, so averaging can be applied with success

to determine the nominal phase value. In future works, this qualitative explanation should be quantified to determined an upper bound on the phase noise allowed in the receiver ADC sampler time base and the require number of averages.

Another direction in extending the 1-channel receiver implementation in Chapter 3, is the characterization of frequency conversion circuits. The idea is to use a receiver ADC sampler time base that is phase-locked to the common local oscillation (LO) signal in the 1-channel receiver and the frequency conversion circuit. So when the receiver LO is retuned to measure responses in another frequency band, the ADC sampler time base can again be used to relate the phase of sequentially sampled waveforms.

The modeling methodology in Chapter 4 has so far been applied to modeling nonlinear transmission responses of RF circuits. It was noted that when modeling the transmission response up to the 1 dB compression point, the excitation power caused significant distortion in the port reflection responses, so that the matched assumption is no longer valid. To model more nonlinear behavior beyond the 1 dB compression point, future work should extend the application to modeling nonlinear reflection responses, so that interface nonlinear mismatch effects are accounted for. The general identification scheme does not change. The formally unidirectional transfer function type of grey-box model, is modified by adding nonlinear circuit elements which are more accurate representation of the RF input and output circuitry.

Supporting the above expanded application of the proposed modeling methodology requires improvements in the current form of the direct calculation method. Since both terminal current and voltages are required, and parameters of nonlinear sources in their natural form are desired, current nodal analysis can be changed into mixed-variable analysis. In addition, the ubiquitous RF front-end filter currently modeled as an unidirectional transfer function should be changed into a bilateral S-parameter block. This can be implemented by borrowing existing techniques used in the NLTF method of Volterra circuit analysis.

REFERENCES

[1] J. Verspecht, "Calibration of a measurement system for high-frequency nonlinear devices," Ph.D. dissertation, Vrije Universiteit, Brussels, Belgium, 1995.

[2] M. Jeruchim, P. Balaban, and K. Shanmugan, *Simulation of Communication Systems*. New York: Kluwer, 2002.

[3] W. H. Tranter, K. S. Shanmugan, T. S. Rappaport, and K. L. Kosbar, *Principles of Communication Systems Simulation with Wireless Applications*. Upper Saddle River, New Jersey: Prentice Hall, 2004.

[4] *Behavioral Modeling of Amplifiers in ADS*, Agilent Technologies, 2006.

[5] P. S. Blockley, D. Gunyan, and J. B. Scott, "Mixer-based, vector-corrected, vector signal/network analyzer offering 300kHz-20GHz bandwidth and traceable phase response," *IEEE MTT-S Int. Microwave Symposium Digest*, pp. 1497–1500, Jun. 2005.

[6] P. Wambacq and W. Sansen, *Distortion Analysis of Analog Integrated Circuits*. Kluwer Academic Publisher, 1998.

[7] W. J. Rugh, *Nonlinear System Theory, The Volterra/Wiener Approach*. The Johns Hopkins University Press, 1981.

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

[9] H. B. Poza, Z. A. Sarkozy, and H. L. Berger, "A wideband data link computer simulation model," *Proc. NAECON Conference*, 1975.

[10] A. M. Saleh, "Frequency-independent and frequency-dependent nonlinear models of TWT amplifiers," *IEEE Trans. Commun.*, vol. 29, no. 11, pp. 1715–1720, 1981.

[11] M. T. Abuelmaatti, "Frequency-dependent nonlinear quadrature model for TWT amplifiers," *IEEE Trans. Commun.*, vol. 32, no. 8, pp. 982–986, 1984.

[12] K. Gard, H. Gutierrez, and M. B. Steer, "Characterization of spectral regrowth in microwave amplifiers based on the nonlinear transformation of a complex Gaussian process," *IEEE Trans. Microwave Theory Tech.*, vol. 47, no. 7, pp. 1059–1069, Jul. 1999.

[13] H. Gutierrez, K. Gard, and M. B. Steer, "Nonlinear gain compression in microwave amplifiers using generalized power-series analysis and transformation of input statistics," *IEEE Trans. Microwave Theory Tech.*, vol. 48, no. 10, pp. 1774–1777, Oct. 2000.

[14] K. Gard, L. Larson, and M. B. Steer, "The impact of RF front-end characteristics on the spectral regrowth of communications signals," *IEEE Trans. Microwave Theory Tech.*, vol. 53, no. 6, pp. 2179–2186, Jun. 2005.

[15] J. Pedro and N. Carvalho, "On the use of multitone techniques for assessing RF components intermodulation distortion," *IEEE Trans. Microwave Theory Tech.*, vol. 47, no. 12, pp. 2393–2402, Dec. 1999.

[16] D. Schreurs and K. Remley, "Use of multisine signals for efficient behavioural modelling of RF circuits with short memory effects," *61st ARFTG Conf. Dig.*, Jun. 2003.

[17] T. V. Broeck and J. Verspecht, "Calibrated vectorial nonlinear network analyzers," *IEEE Microwave Theory Tech. Symposium Digest*, pp. 1069–1072, May. 1994.

[18] J. Jargon, D. DeGroot, and D. Vecchia, "Repeatability study of commercial harmonic phase standards measured by a nonlinear vector network analyzer," *62nd ARFTG Conf. Dig.*, vol. 3, pp. 243–258, Dec. 2003.

[19] J. Verspecht and K. Rush, "Individual characterization of broadband sampling oscilloscopes with a nose-to-nose calibration procedure," *IEEE Trans. Instrum. Meas.*, vol. 43, no. 2, pp. 347–354, Apr. 1994.

[20] M. B. Steer, *Microwave and RF Design: A Systems Approach.* Scitech, 2010.

[21] C. D. Woody, "Characterization of an adaptive filter for the analysis of variable latency neuroelectric signals," *IEE Med. Biol. Eng. Comput.*, vol. 5, pp. 539–553, 1967.

[22] K. J. Coakley and P. Hale, "Alignment of noisy signals," *IEEE Trans. Instrum. Meas.*, vol. 50, no. 1, pp. 141–149, Feb. 2001.

[23] D. Williams, P. Hale, and K. A. Remley, "The sampling oscilloscope as a microwave instrument," *IEEE Microwave Magazine*, pp. 59–68, Aug. 2007.

[24] P. Hale, C. M. Wang, D. F. Williams, K. A. Remley, and J. D. Wepman, "Compensation of random and systematic timing errors in sampling oscilloscopes," *IEEE Trans. Instrumentation and Measurement*, vol. 55, no. 6, pp. 2146–2154, Dec. 2006.

[25] *HP 71500A and HP 70820A Microwave Transition Analyzer User's Guide*, Agilent Technologies, 1992.

[26] K. A. Remley, D. F.Williams, D. M. Schreurs, G. Loglio, and A. Cidronali, "Phase detrending for measured multisine signals," *61th ARFTG Conf. Dig.*, pp. 73–83, Jun. 2003.

[27] P. S. Blockley, J. B. Scott, D. Gunyan, and A. E. Parker, "Noise considerations when determining phase of large-signal microwave measurements," *IEEE Trans. Microwave Theory Tech.*, vol. 54, no. 8, pp. 3182–3190, Aug. 2006.

[28] J. Jargon, D. DeGroot, K. Gupta, and A. Cidronali, "Calculating ratios of harmonically related, complex signals with application to nonlinear large-signal scattering parameters," *60th ARFTG Conf. Dig.*, pp. 113–122, Dec. 2002.

[29] T. Nakatani, T. Matsuura, and K. Ogawa, "A simple method for measuring the IM3 components of multi-stage cascaded power amplifiers considering the phase characteristics," *IEEE MTT-S Int. Microwave Symposium Digest*, pp. 1731–1734, Jun. 2004.

[30] G. Loglio, J. Jargon, and D. C. DeGroot, "Phasor angle definition suitable for intermodulation measurements," *65th ARFTG Conf. Dig.*, pp. 83–89, Jun. 2005.

[31] J. P. Martins, J. C. Pedro, and P. M. Cabral, "New method for phase characterization of nonlinear distortion products," *IEEE MTT-S Int. Microwave Symposium Digest*, pp. 971–974, Jun. 2005.

[32] R. Pintelon and J. Schoukens, *System Identification - A Frequency Domain Approach*. Piscataway, NJ: IEEE Press, 2001.

[33] N. Suematsu, Y. Lyama, and O. Ishida, "Transfer characteristic of IM3 relative phase for a GaAs FET amplifier," *IEEE Trans. Microwave Theory Tech.*, vol. 45, no. 12, pp. 2509–2514, Dec. 1997.

[34] Y. Yang, J. Yi, J. Nam, B. Kim, and M. Park, "Measurement of two-tone transfer characteristics of high-power amplrfiers," *IEEE Trans. Microwave Theory Tech.*, vol. 49, no. 3, pp. 2509–2514, Mar. 2001.

[35] J. Martins and N. Carvalho, "Multitone phase and amplitude measurement for nonlinear device characterization," *IEEE Trans. Microwave Theory Tech.*, vol. 53, no. 6, pp. 1982–1989, Jun. 2005.

[36] J. H. K. Vuolevi, T. Rahkonen, and J. P. A. Manninen, "Measurement technique for characterizing memory effects in RF power amplifiers," *IEEE Trans. Microwave Theory Tech.*, vol. 49, no. 8, pp. 1383–1389, Aug. 2001.

[37] A. Walker, M. Steer, and K. Gard, "Simple, broadband relative phase measurement of intermodulation products," *65th ARFTG Conf. Dig.*, pp. 123–127, Jun. 2005.

[38] J. Martins and N. Carvalho, "Practical multitone amplitude and phase characterization," *Asia-Pacific Microwave Conference Proceedings*, Dec. 2005.

[39] J. C. Pedro and J. P. Martins, "Amplitude and phase characterization of nonlinear mixing products," *IEEE Trans. Microwave Theory Tech.*, vol. 54, no. 8, pp. 3237–3245, Aug. 2006.

[40] J. C. Pedro and S. A. Maas, "A comparative overview of microwave and wireless power-amplifier behavioral modeling approaches," *IEEE Trans. Microwave Theory Tech.*, vol. 53, no. 4, pp. 1150–1163, Apr. 2005.

[41] M. Isaksson, D. Wisell, and D. Ronnow, "A comparative analysis of behavioral models for RF power amplifiers," *IEEE Trans. Microwave Theory Tech.*, vol. 54, no. 1, Jan. 2006.

[42] J. Wood and D. E. Root, Eds., *Fundamentals of Nonlinear Behavioral Modeling for RF and Microwave Design*. Norwood, MA: Artech House, 2005.

[43] D. Schreurs, M. ODroma, A. Goacher, and M. Gadringer, *RF Power Amplifier Behavioral Modeling*. Cambridge, UK: Cambrige University Press, 2008.

[44] W. Bosch and G. Gatti, "Measurement and simulation of memory effects in predistortion linearizers," *IEEE Trans. Microwave Theory Tech.*, vol. 37, no. 12, pp. 1885–1890, Dec. 1989.

[45] S. Boyd and L. Chua, "Fading memory and the problem of approximating nonlinear operators with Volterra series," *IEEE Trans. on Circuits & Syst.*, vol. CAS-32, pp. 1150–1161, Nov. 1985.

[46] V. Mathews and G. Sicuranza, *Polynomial Signal Processing*. New York: John Wiley & Sons, 2000.

[47] F. Verbeyst and M. V. Bossche, "VIOMAP, the S-parameter equivalent for weakly nonlinear RF and microwave devices," *IEEE Trans. Microwave Theory Tech.*, vol. 42, no. 12, p. 25312533, Dec. 1994.

[48] J. Xu, M. C. E. Yagoub, R. Ding, and Q. J. Zhang, "Neural based dynamic modeling of nonlinear microwave circuits," *IEEE MTT-S Int. Microwave Symposium Digest*, pp. 1101–1104, Jun. 2002.

[49] J. Wood, D. E. Root, and N. B. Tufillaro, "A behavioral modeling approach to nonlinear model-order reduction for RF/microwave ICs and systems," *IEEE Trans. Microwave Theory Tech.*, vol. 52, no. 9, pp. 2274–2284, Sep. 2004.

[50] H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*. Cambridge University Press, 1997.

[51] T. Wang and T. J. Brazil, "Volterra-mapping-based behavioral modeling of nonlinear circuits and systems for high frequencies," *IEEE Trans. Microwave Theory Tech.*, vol. 51, no. 5, pp. 1433–1440, May. 2003.

[52] A. Zhu and T. J. Brazil, "Behavioral modeling of RF power amplifiers based on pruned Volterra series," *IEEE Microwave and Wireless Components Letters*, vol. 14, no. 12, pp. 563–565, Dec. 2004.

[53] J. Bendat and A. Piersol, *Engineering Applications of Correlation and Spectral Analysis*. New York: John Wiley & Sons, 1993.

[54] H. Ku and J. S. Kenney, "Behavioral modeling of nonlinear RF power amplifiers considering memory effects," *IEEE Trans. Microwave Theory Tech.*, vol. 51, no. 12, pp. 2495–2504, Dec. 2003.

[55] S. A. Billings and S. Y. Fakhouri, "Identification of a class of nonlinear systems using correlation analysis," *Proc. Inst. Electrical Engineers*, vol. 125, pp. 691–697, Jul. 1978.

[56] E. Ngoya, N. L. Gallou, J. M. Nebus, H. Buret, and P. Reig, "Accurate RF and microwave system level modeling of wide band nonlinear circuits," *IEEE MTT-S Int. Microwave Symp. Dig.*, pp. 79–82, Jun. 2000.

[57] E. Ngoya and A. Soury, "Modeling memory effects in nonlinear subsystems by dynamic Volterra series," *Proc. Int. Behavioral Modeling and Simulation Workshop*, pp. 28–33, Oct. 2003.

[58] ——, "A two-kernel nonlinear impulse response model for handling long term memory effects in RF and microwave solid state circuits," *IEEE MTT-S Int. Microwave Symp. Dig.*, pp. 1105–1108, Jun. 2006.

[59] P. Asbeck, H. Kobayashi, M. Iwamoto, G. Nanington, S. Nam, and L. E. Larson, "Augmented behavioral characterization for modeling the nonlinear response of power amplifiers," *IEEE MTT-S Int. Microwave Symp. Dig.*, p. 135138, Jun. 2002.

[60] T. Narhi, "Frequency-domain analysis of strongly nonlinear circuits using a consistent large-signal model," *IEEE Trans. Microwave Theory Tech.*, vol. 44, no. 2, pp. 182–192, Feb. 1996.

[61] A. Cidronali, C. Accillaro, and G. Manes, "Mildly nonquasi-static two-port device model extraction by integrating linearized large-signal vector measurements," *IEEE Trans. Microwave Theory and Tech.*, vol. 55, no. 11, pp. 2277–2289, Nov. 2007.

[62] J. C. P. Jones and S. Billings, "Describing functions, Volterra series, and the analysis of non-linear systems in the frequency domain," *Int. J. Contr.*, vol. 53, no. 4, pp. 871–887, Apr. 1991.

[63] J. Verspecht, D. Williams, D. Schreurs, K. Remley, and M. McKinley, "Linearization of large-signal scattering functions," *IEEE Trans. Microwave Theory Tech.*, vol. 53, no. 4, pp. 1369–1376, Apr. 2005.

[64] D. E. Root, J. Verspecht, D. Sharrit, J. Wood, and A. Cognata, "Broad-band poly-harmonic distortion (PHD) behavioral models from fast automated simulations and large-signal vectorial network measurements," *IEEE Trans. Microwave Theory Tech.*, vol. 53, no. 11, pp. 3656–3664, Nov. 2005.

[65] J. Verspecht, D. Gunyan, J. Horn, J. Xu, A. Cognata, and D. E. Root, "Multi-tone, multi-port, and dynamic memory enhancements to PHD nonlinear behavioral models from large-signal measurements and simulations," *IEEE MTT-S Int. Microwave Symposium Digest*, pp. 969–972, Jun. 2007.

[66] Q. Liang, J. Andrews, J. Cressler, and G. Niu, "Systematic linearity analysis of RFICs using a two-port lumped-nonlinear-source model," *IEEE Trans. Microwave Theory Tech.*, vol. 53, no. 5, pp. 1745–1755, Feb. 2005.

[67] T. R. Cunha, J. C. Pedro, and P. M. Cabral, "Design of a power amplifier feed-forward RF model with physical knowledge considerations," *IEEE Trans. Microwave Theory Tech.*, vol. 55, no. 12, pp. 2747–2756, Dec. 2007.

[68] A. Zhu, J. Pedro, and T. Cunha, "Pruning the Volterra series for behavioral modeling of power amplifiers using physical knowledge," *IEEE Trans. Microwave Theory Tech.*, vol. 55, no. 5, pp. 813–821, May. 2007.

[69] A. Walker, M. B. Steer, and K. G. Gard, "Capturing asymmetry in distortion of an RF system using a multislice behavioral model," *IEEE Radio and Wireless Conference*, vol. 16, no. 4, pp. 212–214, Apr. 2006.

[70] E. W. Bai, "An optimal two-stage identification algorithm for Hammerstein-Wiener nonlinear systems," *Automatica*, vol. 34, no. 3, pp. 333–338, Mar. 1998.

[71] M. Boutayeb and M. Darouach, "Recursive identification method for MISO Wiener-Hammerstein model," *IEEE Trans. Automatic Control*, vol. 40, pp. 287–291, Feb. 1995.

[72] Y. Zhu, "Estimation of an N-L-N Hammerstein-Wiener model," *Automatica*, vol. 38, no. 9, pp. 1607–1614, 2002.

[73] H. W. Chen, "Modeling and identification of parallel nonlinear systems: structural classification and parameter estimation methods," *Proc. IEEE*, vol. 83, no. 1, pp. 39–66, Jan. 1995.

[74] P. Crama and J. Schoukens, "Computing an initial estimate of a Wiener-Hammerstein system with a random phase multisine," *IEEE Trans. Intrumentation and Measurement*, vol. 54, no. 1, pp. 117–122, Feb. 2005.

[75] G. Vandersteen and J. Schoukens, "Measurement and identification of nonlinear systems consisting of linear dynamic blocks and one static nonlinearity," *IEEE Trans. Automatic Control*, vol. 44, no. 6, pp. 1266–1271, Jun. 1999.

[76] K. A. Remley, P. Hale, D. I. Bergman, and D. Keenan, "Comparison of multisine measurements from instrumentation capable of nonlinear system characterization," *66th ARFTG Conf. Dig.*, pp. 34–43, Dec. 2006.

[77] A. Walker, M. B. Steer, and K. G. Gard, "A vector intermodulation analyzer applied to behavioral modeling of nonlinear amplifiers with memory," *IEEE Trans. Microwave Theory Tech.*, vol. 54, no. 5, pp. 1991–1999, May 2006.

[78] J. R. Wilkerson, K. G. Gard, and M. B. Steer, "Automated broadband high-dynamic-range nonlinear distortion measurement system," *IEEE Trans. Microwave Theory Tech.*, vol. 58, no. 5, pp. 1273–1282, May 2010.

[79] K. R. Godfrey and M. J. Chapman, "Identifiability and indistinguishability of linear compartmental models," *Math. Comput. Simul.*, vol. 32, no. 2, pp. 273–295, Jun. 1990.

[80] S. Boyd and L. O. Chua, "Uniqueness of circuits and systems containing one nonlinearity," *IEEE Trans. Automatic Control*, vol. 30, no. 7, pp. 674–681, July. 1985.

[81] A. Quintero and S. Pierre, "On the design of large-scale UMTS mobile networks using hybrid genetic algorithms," *IEEE Trans. Vehicular Technology*, vol. 57, no. 4, pp. 2498–2508, 2008.

[82] E. Dilettoso and N. Salerno, "A self-adaptive niching genetic algorithm for multimodal optimization of electromagnetic devices," *IEEE Trans. Magnetics*, vol. 42, no. 4, pp. 1203–1206, 2006.

[83] 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 Trans. Microwave Theory Tech.*, vol. 38, no. 11, pp. 1711–1718, 1990.

[84] S. G. Skaggs, J. Gerber, G. L. Bilbro, and M. B. Steer, "Parameter extraction of microwave transistors using a hybrid gradient descent and tree annealing approach," *IEEE Trans. Microwave Theory Tech.*, vol. 41, no. 4, pp. 726–729, 1993.

[85] S. Y. Yuen and C. K. Chow, "A genetic algorithm that adaptively mutates and never revisits," *IEEE Trans. Evolutionary Computation*, vol. 13, no. 2, pp. 454–472, 2008.

[86] D. Deschrijver, B. Haegeman, and T. Dhaene, "Orthonormal vector fitting: a robust macromodeling tool for rational approximation of frequency domain responses," *IEEE Trans. Advanced Packaging*, vol. 30, no. 2, pp. 216–225, May. 2007.

[87] S. Boyd, Y. S. Tang, and L. O. Chua, "Measuring Volterra kernels," *IEEE Trans. Circuits and Systems*, vol. CAS-30, no. 8, pp. 571–577, Aug. 1983.

[88] A. M. Victor and M. B. Steer, "Transceiver cascade system analysis and design via a contribution method," *Int. J. on RF and Microwave Computer Aided Engineering*, vol. 16, no. 4, pp. 338–345, Jul. 2006.

[89] J. Hu, K. G. Gard, N. Carvalho, and M. B. Steer, "Time-frequency characterization of long-term memory in nonlinear power amplifiers," *IEEE MTT-S Int. Microwave Symposium Digest*, pp. 269–272, 2008.

[90] N. Wiener, "Response of a nonlinear device to noise," M.I.T. Radiation Lab., Cambridge, MA, Tech. Rep. V-16S, Apr. 1942.

[91] J. J. Bussang, L. Ehrman, and J. W. Graham, "Analysis of nonlinear systems with multiple inputs," *Proc. IEEE*, vol. 62, no. 8, pp. 1088–1119, 1974.

[92] D. D. Wiener and J. F. Spina, *Sinusoidal Analysis and Modeling of Weakly Nonlinear Circuits*. New York: Van Nostrand, 1980.

[93] M. B. Steer, P. J. Khan, and R. S. Tucker, "Relationship of volterra series and generalised power series," *Proc. IEEE*, vol. 71, pp. 1453–1454, Dec. 1983.

[94] G. Gielen, H. Walscharts, and W. Sansen, "A symbolic simulator for analog integrated circuits," *IEEE Journal of Solid-State Circuits*, vol. 24, pp. 1587–1597, Dec. 1989.

[95] A. Bauer and W. Schwarz, "Circuit analysis and optimization with automatically derived Volterra kernels," *Proc. IEEE International Symposium on Circuits and Systems*, vol. 1, pp. 491–494, 2000.

[96] S. D. Bedrosian, "Normalized design of 90 phase-difference networks," *IRE Transactions on Circuit Theory*, vol. CT-7, Jun. 1960.

[97] J. Low and A. S. Wong, "Systematic approach to cosite analysis and mitigation techniques," *Proceedings of the Tactical Communication Conference*, pp. 555–567, Apr. 1990.

[98] M. Lustgarten, T. Hensler, and M. Maiuzzo, "A cosite analysis capability for evaluating the EMC of frequency hopping and single channel radios," *Proceedings of the Tactical Communication Conference*, pp. 543–553, Apr. 1990.

[99] J. Isaacs, W. Robertson, and R. Morrison, "A cosite analysis method for frequency hopping radio systems," *1991 MILCOM Conference Record*, pp. 522–526, Nov. 1991.

[100] D. E. Pauley, "Cosite interference simulation," *1999 MILCOM Conference Record*, pp. 1267–1271, Nov. 1999.

[101] M. Maiuzzo, T. Harwood, and W. Duff, "Radio frequency distribution system (RFDS) for cosite electromagnetic compatibility," *International Symposium on Electromagnetic Compatibility*, pp. 250–255, Aug. 2005.

[102] R. Adams, T. Harwood, and M. Maiuzzo, "An innovative signal distribution system that allows EMI free communications for navy ships," *2008 MILCOM Conference Record*, pp. 1–4, Nov. 2008.

[103] J. Proakis, *Digital Communications*. New York: McGraw-Hill, 2000.

[104] S. Verdu, *Multiuser Detection*. Cambrige, UK: Cambrige University Press, 1998.

[105] J. C. Rudell, "Frequency translation techniques for high-integration high-selectivity multistandard wireless communication systems," Ph.D. dissertation, University of California, Berkeley, Berkeley, California, 2000.

[106] S. Vitali, N. D. Laurentiis, G. Albertazzi, F. Agnelli, and R. Rovatti, "Multi-standard simulation of WLAN/UMTS/GSM transceivers for analog front-end validation and design," *2004 International Symposium on Wireless Communication Systems*, pp. 16–20, Sep. 2004.

[107] D. R. de Llera Gonzalez, A. Rusu, M. Ismail, and H. Tenhunen, "TACT: A multi-standard RF transceiver architecture comparison tool," *2005 Midwest Symposium on Circuits and Systems*, pp. 659–662, Aug. 2005.

[108] D. R. de Llera Gonzalez, A. Rusu, and M. Ismail, "EDA for RF and analog front-ends in the 4G era: challenges and solutions," *2007 European Conference on Circuit Theory and Design*, pp. 24–27, Aug. 2007.

[109] A. Morgado, R. del Rio, and J. M. de la Rosa, "A SIMULINK block set for the high-level simulation of multistandard radio receivers," *2007 IEEE International Symposium on Circuits and Systems*, pp. 2950–2953, May. 2007.

[110] Y. Joannon, V. Beroulle, R. Khouri, C. Robach, S. Tedjini, and J. L. Carbonero, "Behavioral modeling of WCDMA transceiver with VHDL-AMS language," *2006 IEEE Design and Diagnostics of Electronic Circuits and systems*, pp. 111–116, Jul. 2006.

[111] M. R. Casu, M. Crepaldi, and M. Graziano, "A VHDL-AMS simulation environment for an UWB impulse radio transceiver," *IEEE Transactions on Circuits and Systems I*, pp. 1368–1381, Jun. 2008.

[112] C. Saunders, G. J. Mazzaro, and M. B. Steer, "Robust reduced-order modelling of distributed linear networks," *IET Microwaves Antennas and Propagation*, vol. 4, no. 7, pp. 962–973, Jul. 2010.

APPENDICES

Appendix A

Symbolic Volterra Circuit Simulator Implementing Direct Calculation Method

A.1 User Guide

The Matlab implementation of the direct calculation method for symbolic Volterra circuit analysis is given here. The Matlab code is presented in a sequence as seen from the user's perspective. Fig. 5.2 can be used to graphically illustrate this sequence of code development. The complete collection of Matlab code is also available at Mathworks file exchange website http://www.mathworks.com/matlabcentral/fileexchange/31124, Direct Calculation Method of Symbolic Volterra Circuit Analysis, so that user feedback can be easily incorporated in future code improvements.

First, the nonlinear current source information is generated using the main script in Section A.2. The required user inputs are the maximum nonlinear order to be considered and the number of tones in the excitation. The nonlinear current source information generated are stored in the nonlinear current data structures shown in Fig. 5.2, and recalled during circuit

analysis. For the same type of excitation, these data structures can be saved to avoid redundant script execution. For completeness, the code for the functions called in the main script are also presented in Section A.2. The corresponding underlying theoretical development was presented in Section 5.2.2.

Next, the circuit to be analyzed needs to defined by the user. Two example scripts are given in Section A.3 to create the data structure describing the circuit. This data structure is accessed later by the circuit analysis program, and is implicitly part of the user input in Fig. 5.2. The two examples given correspond to the ones presented in Section 5.3 where circuit equations are derived and results from the circuit analysis program are presented. For other circuits, the user must first derive the circuit equations, then encode circuit information (such as the admittance matrix and controlling variables for the dependent nonlinear current sources) into appropriate data structures by following the examples scripts.

Finally, the user defined circuit is analyzed using the main script in Section A.4. The required user inputs are the IM response frequencies which are specified as frequency mix vectors. The circuit analysis results are stored in the circuit response data structure in Fig. 5.2. These results were presented in a tabular format for the two example circuits in Section 5.3. For completeness, the code for the functions called in the main script are also presented in Section A.4. The corresponding underlying theoretical development was presented in Section 5.2.3.

A.2 Generate Nonlinear Current Source

A.2.1 Main Script

```
%% ################################################## MIX, Wnr, Fp, Mn, Fm
%measure 1-tone + 2-tone response: find order N
N = 3;% max nonlinear order
Q = 2;% number of tones
```

```
[ MIX ] = Mix_Qtone_Sym( N, Q, struct('HD',{'wHD'}, ...
        'fListOrd',{'ascend'}), Opt_Disp);
%------------------------------ Poly
[ Wnr ]         = INL_Wnr( N, Opt_Disp);
[ Fp  Fp_ ]     = INL_Poly( N, MIX, Wnr, Opt_Disp);% Fp_{:}
[ Fp2 ]         = INL_Poly_Condense( Fp, Wnr, MIX, 'ALL', 0);
%------------------------------ Mult
[ Mn  Fm  Fm_ ] = INL_Mult( N, MIX, Opt_Disp);% Fm_{:}
[ Fm2 ]         = INL_Mult_Condense( Fm, Mn, MIX, 'ALL', 0);
%------------------------------
% save MIX Wnr Fp  Fp_ Fp2 Mn  Fm  Fm_ Fm2

[ MIX ] = Mix_BandDiv( MIX );
[ MIX ] = Mix_BandSYM2( MIX, 1);
```

A.2.2 Functions

Mix_Qtone_Sym Function

```
function [ MIX ] = Mix_Qtone_Sym( N, Q, Opt, Opt_Disp)
% for Q < 2-Tone only
%----------------------------
% mix_fSymVEC = [-f2 -f1 f1 f2]  Symbolic Freq
% fSymVEC       = [        f1 f2]   Symbols in Symbolic Freq
% fSymCEL       = {        f1 f2}
% fNumVEC       = [        #  # ]   Numeric-Val in Symbolic Freq
%----------------------------
if ~exist('Opt_Disp','var')
    Opt_Disp = 0;
end


    Nord    = N;
    Nfreq   = CumSum( Q );
    Ntone   = Q;
Buss_Sym
switch Q
    %====================================================================
    case{1} %Q=1 for 1-tone, thus mix_MAT = [m1 m2]
    MIX.mix_fSymVEC = [-fc fc];
    MIX.fNumVEC     = [ 200 ];%{-200,200}
    %====================================================================
    case{2} %Q=2 for 2-tone, thus mix_MAT = [m1 m2 m3 m4]
    MIX.mix_fSymVEC = [-f2 -f1  f1  f2];
```

```
    MIX.fNumVEC     = [ 200, 201 ];% {-201,-200,200,201}
    %==================================================================
    case{3} %Q = 3 for 3-tone, thus mix_MAT = [m1 m2 m3 m4 m5 m6]
    MIX.mix_fSymVEC = [-f3 -f2 -f1  f1  f2  f3];
    MIX.fNumVEC     = [ 200, 201, 202 ];% {-202,-201,-200,200,201,202}
    %==================================================================
    otherwise
        disp('Q > 3'); return;
    %==================================================================
end
    MIX.fSymVEC     = MIX.mix_fSymVEC( length(MIX.mix_fSymVEC)/2+1 : end);
    MIX.fSymCEL     = SYMvec2SYMcel( MIX.fSymVEC );

%% ########################################################
MIX.N   = N;
MIX.Q   = Q;
%------------------------------------------------------------------------
MIX.mix_MAT     = {};% {[4x4 double]  [10x4 double]  [20x4 double]}
MIX.fmix_MAT    = {};% {[4x2 double]  [10x2 double]  [20x2 double]}
MIX.fmixHD_VEC = {};% {[4x1 double]  [10x1 double]  [20x1 double]}
MIX.fmixSYM_VEC = {};% {[4x1 sym]     [10x1 sym]     [20x1 sym]}
MIX.mixCoef_VEC = {};% {[4x1 double]  [10x1 double]  [20x1 double]}
%------------------------------------------------------------------------
MIX.fmix_RCD        = {};% {[2x2 double]  [5x2 double]  [8x2 double]}
MIX.fmixHD_RCD     = {};% {[2x1 double]  [5x1 double]  [8x1 double]}
MIX.fmixSYM_RCD   = {};% {[2x1 sym]  [5x1 sym]  [8x1 sym]}
MIX.fmix_RCDIDX     = {};% {}
```

```
MIX.fmix_RCDIDX2    = {};% ptr to xxx_ALL  {[1 2]  [3 4 5 6 7]}
MIX.fmix_RCDIDXcel = {};% {}
%-------------------------------------------------------------------------
MIX.fmix_ALL        = [];% [13x2 double]
MIX.fmixHD_ALL    = [];% [13x1 double]
MIX.fmixSYM_ALL     = [];% [13x1 sym]
MIX.fmix_ALLIDXcel = {};% ptr to xxx_RCD  {1x13 cell}
%-------------------------------------------------------------------------

for n = 1 : N
    if(Opt_Disp)
        disp(['%############################ n=' int2str(n)])% DEBUG
    end
    [MIX.mix_MAT{n}     MIX.fmix_MAT{n}     MIX.fmixHD_VEC{n}       ...
     MIX.fmixSYM_VEC{n}      MIX.mixCoef_VEC{n} ...
     MIX.fmix_RCD{n}     MIX.fmixHD_RCD{n}       MIX.fmixSYM_RCD{n} ] = ...
        Mix_Qtone(Q, MIX.mix_fSymVEC, MIX.fSymVEC, MIX.fNumVEC, ...
            n, Opt, Opt_Disp);

    [ALL_IDX ALLXOR_IDX RCD_IDX     RCDXOR_IDX] = Find_VecIDX( ...
        MIX.fmix_ALL, MIX.fmix_RCD{n}, 'MATRIX', 'Not symSUBnum', [],0);
    MIX.fmix_RCDIDX2{n}(RCD_IDX)      = ALL_IDX;
    MIX.fmix_RCDIDX2{n}(RCDXOR_IDX)   = length( MIX.fmix_ALL ) ...
        + [1 : length(RCDXOR_IDX)];

    for nRCD = 1 : length( RCD_IDX )
        MIX.fmix_ALLIDXcel{ALL_IDX(nRCD)} = ...
```

```
            [MIX.fmix_ALLIDXcel{ALL_IDX(nRCD)}  [n; RCD_IDX(nRCD)] ];
    end
    for nRCD = 1 : length( RCDXOR_IDX )
        MIX.fmix_ALLIDXcel{length(MIX.fmixSYM_ALL) + nRCD} = ...
            [n; RCDXOR_IDX(nRCD)];
    end
    MIX.fmix_ALL = [MIX.fmix_ALL;   MIX.fmix_RCD{n}(RCDXOR_IDX,:) ];
    MIX.fmixHD_ALL = [MIX.fmixHD_ALL; MIX.fmixHD_RCD{n}(RCDXOR_IDX) ];
    MIX.fmixSYM_ALL = [MIX.fmixSYM_ALL; MIX.fmixSYM_RCD{n}(RCDXOR_IDX) ];

end
```

INL_Wnr Function

```
function [ Wnr ] = INL_Wnr( N, Opt_Disp)
if ~exist('Opt_Disp','var')
    Opt_Disp = 0;
end

ord_FACTORIAL = FactorialVec([1:N]);
%-----------------------------------------------------------
%Wnr.OrdPwr_MAT{n,r}               [ Vn^Pwr ] = [n1 ... nr]
%Wnr.c_VEC{n,r}                   ([Multiplicant])
%   .Ord_MAT(remove bc implicit)   position cC of non-zero in OrdPwr_MAT
%-----------------------------------------------------------
%Wnr.PolyOrd_MAT{n,1}   k=[Multiplicant Vr1-Ord ... Vrn-Ord]=[n1 n1 n2 n2]
%    PolyDeg_MAT        implicit: [1...1]
%Wnr.cFCT_VEC{n,1}  calc iNL for NLTF: n!/([1!^k1...n!^kn]).*[k1! ... kn!])
```

```
%                                       * limited-Arithm-Avg( H(f1)H(f1,f2) )
%                          normally: n! * Arithm-Avg( H(f1)H(f1,f2) )
if(Opt_Disp)
sprintf('c \t cFCT \t OrdPwr')
end

% Wnr.OrdPwr_MAT: {3x3 cell}
%      Wnr.c_VEC: {3x3 cell}
%   Wnr.cFCT_VEC: {3x3 cell}
%Wnr.PolyOrd_MAT: {3x3 cell}
%% ############################ Wnr{n,r}.OrdPwr_MAT
for n = 1 : N
    if(Opt_Disp)
        disp(['%================================ n=' num2str(n)]);
    end
    Wnr.OrdPwr_MAT{n,1}   = zeros(1,N);   Wnr.OrdPwr_MAT{n,1}(n) = 1;
    Wnr.c_VEC{n,1}          = [1];
    Wnr.cFCT_VEC{n,1}    = factorial(n);
    Wnr.PolyOrd_MAT{n,1} = [n];
for r = 2 : n
    if(Opt_Disp) disp(['%--------------------(n,r)=' num2str([n r])]); end
    OrdPwr_MAT = [];
    c_VEC       = [];
cFCT_VEC    = [];
    PolyOrd_MAT = [];
%================================================== OrdPwr_MAT, c_VEC
for nI = 1 : min([n-r+1 N])
```

```
    if(Opt_Disp)
        [[nI 1]   [n-nI r-1]]% DEBUG
    end
MAT1 = Wnr.OrdPwr_MAT{nI,1};
    MAT2    = Wnr.OrdPwr_MAT{n-nI,r-1};
    VEC1 = Wnr.c_VEC{nI,1};
    VEC2 = Wnr.c_VEC{n-nI,r-1};
    wH  = [];
    cH  = [];
for n1 = 1 : size(MAT1,1)
    for n2 = 1 : size(MAT2,1)
        % ADD vector of power of each Vn
        wH(size(wH,1)+1,:)  = MAT1(n1,:) + MAT2(n2,:);
        % MULTIPLY coef of prod
        cH(size(cH,1)+1,1) = VEC1(n1,1) * VEC2(n2,1);
    end
    end
    [wnr_IDX wnr_IDX_ wH_IDX wH_IDX_]    = Find_VecIDX( ...
        OrdPwr_MAT, wH,'MATRIX','Not symSUBnum',[],0);
    c_VEC(wnr_IDX) = c_VEC(wnr_IDX) + cH(wH_IDX);% existing
    OrdPwr_MAT     = [OrdPwr_MAT;  wH(wH_IDX_,:)];% new
    c_VEC          = [c_VEC;       cH(wH_IDX_,1)];% new
end
    Wnr.OrdPwr_MAT{n,r} = OrdPwr_MAT;
    Wnr.c_VEC{n,r}      = c_VEC;
%================================================== PolyOrd_MAT, cFCT_VEC
for rC = 1 : size(OrdPwr_MAT,1)
```

```
        PolyOrd_VEC = [];
        %         ([n1! ... nr!] = [1!^k1 ... n!^kn]) .* [k1! ... kn!]
        cFCT = (ord_FACTORIAL .^ OrdPwr_MAT(rC,:)) ...
                    .* FactorialVec(OrdPwr_MAT(rC,:));
        cFCT = cFCT( find(cFCT ~= 0) );
        cFCT = prod( cFCT );
    for cC = 1 : size(OrdPwr_MAT,2)
        if(OrdPwr_MAT(rC,cC)~=0)
           PolyOrd_VEC = [PolyOrd_VEC repmat(cC,[1 OrdPwr_MAT(rC,cC)])];
        end
   end
        PolyOrd_MAT(rC,:)   = PolyOrd_VEC;
        cFCT_VEC            = [cFCT_VEC;    cFCT];
end
    Wnr.PolyOrd_MAT{n,r} = PolyOrd_MAT;
    Wnr.cFCT_VEC{n,r}       = cFCT_VEC;
%=================================================
    if(Opt_Disp)
        [c_VEC      cFCT_VEC    OrdPwr_MAT]% DEBUG
        PolyOrd_MAT% DEBUG
    end
end
end
```

INL_Poly Function

```
function [ Fp  Fp_ ] = INL_Poly( N, MIX, Wnr, ...
    Opt_Disp)
```

```
if ~exist('Opt_Disp','var')
    Opt_Disp = 0;
end

%% ########################################################## initialize
for n = 1 : N
    Fp_{n} = zeros( size(MIX.fmix_RCD{n},1), 2);
for nRESP = 1 : size( MIX.fmix_RCD{n},1 )
%------------------------------------------------------------------ init .wnr
for r = 2 : n
for rC = 1 : size( Wnr.PolyOrd_MAT{n,r},1 )
    for nri = 1 : 2
        Fp{n}{nRESP,nri}.wnr{r}{rC,1} = [];% PolyOrd_nMIXMAT
        Fp{n}{nRESP,nri}.wnr{r}{rC,2} = [];% PolyOrd_riMAT
    end
end
end
%------------------------------------------------------------------ init .fmix
for k = 1 : N
    for nri = 1 : 2
        Fp{n}{nRESP,nri}.fmix{k,1} = zeros( size( MIX.fmix_RCD{k},1), 2);
    end
end
    %------------------------------ Lin-fResp for 2Tone
    if(n == 1)
        Fp{1}{nRESP,1}.fmix{1}(nRESP,:) = [1 0];
        Fp{1}{nRESP,2}.fmix{1}(nRESP,:) = [0 1];
```

```
    end
%-------------------------------------------------------------------
end
end

%% ######################################################
tic
for n = 2 : N
for r = 2 : n
%-------------------------------------------------------------------
for rC = 1 : size( Wnr.PolyOrd_MAT{n,r}, 1 )
    PolyOrd_IDX    = 1;% (current position in VEC)
PolyOrd_VEC     = Wnr.PolyOrd_MAT{n,r}(rC,:);% [ord ord ...]
    PolyOrd_nMIXVEC = [];% [nMIX nMIX ...]
    PolyOrd_riVEC = [];% [ri   ri   ...]
if(Opt_Disp)
      disp('%=======================================')
        PolyOrd_VEC
end
    %iterate all possible nMIX/nri @(current position in VEC)
    %Recurrsive func call bc: VEC is variable length, can't use for-loop
    [ Fp{n} Fp_{n}] = INL_Fnr_ForRecur( MIX.fmix_RCD, PolyOrd_VEC, ...
        PolyOrd_IDX, PolyOrd_nMIXVEC, PolyOrd_riVEC, ...
        n, r, rC, Fp{n}, Fp_{n}, Opt_Disp);
end
%-------------------------------------------------------------------
end
```

```
        if(Opt_Disp)
            Fn_ = Fp_{n}
        end
end
toc

%% ######################################################## INL_Fnr_ForRecur
function [ Fn Fn_ ] = INL_Fnr_ForRecur( fmix_RCD, PolyOrd_VEC, ...
    PolyOrd_IDX, PolyOrd_nMIXVEC, PolyOrd_riVEC, ...
    n, r, rC, Fn, Fn_, Opt_Disp)
if ~exist('Opt_Disp','var')
    Opt_Disp = 0;
end

%ord @(current position in VEC)
nORD = PolyOrd_VEC( PolyOrd_IDX );
    if(Opt_Disp)
        disp('%---------------------------------------')
        sprintf('PolyOrd_IDX=%d     nORD=%d', PolyOrd_IDX, nORD)
    end
%iterate all possible nMIX/nri @(current position in VEC)
for ri = 1 : -2 : -1
    PolyOrd_riVEC( PolyOrd_IDX )    = ri;
    nMIXstart = 1;
    if (ri == -1) && ...
        isequal( fmix_RCD{nORD}(1,:), zeros( size(fmix_RCD{nORD}(1,:)) ) )
        nMIXstart = 2;% skip double-counting if DC (sorted to be 1st-pt)
```

```
    end
for nMIX = nMIXstart : size( fmix_RCD{nORD}, 1 )
    PolyOrd_nMIXVEC( PolyOrd_IDX )  = nMIX;
%-----------------------------------------------------------------
if( PolyOrd_IDX < length( PolyOrd_VEC ) )% not at final position in VEC
    %############### Don't forget to modify
    %iterate all possible nMIX/nri @(current position in VEC)
    %Recurrsive func call bc: VEC is variable length, can't use for-loop
[ Fn Fn_ ] = INL_Fnr_ForRecur( fmix_RCD, PolyOrd_VEC, ...
     PolyOrd_IDX+1, PolyOrd_nMIXVEC, PolyOrd_riVEC, ...
        n, r, rC, Fn, Fn_, Opt_Disp);
else% reaches final position in VEC: can calc Result fmix_VEC ~OUTPUT
    [ Fn    Fn_ ] = INL_Fnr_fmixVEC( fmix_RCD, PolyOrd_VEC, ...
        PolyOrd_nMIXVEC, PolyOrd_riVEC, ...
        n, r, rC, Fn, Fn_, Opt_Disp);
end
%-----------------------------------------------------------------
end
end

%% ########################################################
function [ Fn Fn_ ] = INL_Fnr_fmixVEC( fmix_RCD, PolyOrd_VEC, ...
    PolyOrd_nMIXVEC, PolyOrd_riVEC, ...
    n, r, rC, Fn, Fn_, Opt_Disp)

fmix_VEC = [];
for k = 1 : length( PolyOrd_VEC )
```

```
    Ord_k   = PolyOrd_VEC(k);
    Mix_k   = PolyOrd_nMIXVEC(k);
    ri_k    = PolyOrd_riVEC(k);
fmix_VEC = [fmix_VEC;       fmix_RCD{ Ord_k }(Mix_k,:) * ri_k];
end
    if(Opt_Disp)
        disp('%--------------------')
        fmix_VEC
    end
fmix_VEC = sum( fmix_VEC, 1 );
%----------------------------- find fmix_VEC in fmix_RCD{n} = nmix
%----------------------------- Fn{nmix,nri}.wnr{r}{rC,:}={nMIXVEC  riVEC}
[ nmix  nri ] = Mix_RCDsearch( fmix_VEC, fmix_RCD{n});
    Fn{nmix,nri}.wnr{r}{rC,1}=[Fn{nmix,nri}.wnr{r}{rC,1}; PolyOrd_nMIXVEC];
    Fn{nmix,nri}.wnr{r}{rC,2}=[Fn{nmix,nri}.wnr{r}{rC,2}; PolyOrd_riVEC];
    Fn{nmix,nri}.fmix{n,1}(nmix, nri) = 1;
    Fn_(nmix,nri) = 1;
%----------------------------- Fn{nmix,nri}.fmix(PolyOrd, nMIXVEC, riVEC)=1
for k = 1 : length( PolyOrd_VEC )
    Ord_k = PolyOrd_VEC(k);
    Mix_k = PolyOrd_nMIXVEC(k);
    ri_k    = PolyOrd_riVEC(k);
        if(ri_k == -1)
            nri_k = 2;
        else
            nri_k = 1;
        end
```

```
Fn{ nmix, nri}.fmix{Ord_k,1}(Mix_k, nri_k) = 1;
    %for NL-Cap, keep track 'jw' contribs (to the same fresp) in list?
    %can use same Wnr expansion: need further modification
end
```

INL_Poly_Condense Function

```
function [ Fp2 ] = INL_Poly_Condense( Fp, Wnr, MIX, Opt, Opt_Disp,...
    RCD_IDX, N)
if ~exist('Opt_Disp','var')
    Opt_Disp = 0;
end

switch Opt
%% ============================================
    case{'Select'}
for n = 1 : N
for nRESP = 1 : size( RCD_IDX.nMIX, 1 )
nmix = RCD_IDX.nMIX(nRESP,n);
nri  = RCD_IDX.ri(nRESP,n);
if(nmix ~= 0)
if(Opt_Disp)
        disp(sprintf('==========n=%-5d  nMIX=%-5d  ri=%-5d fmixSYM=%-s',...
        n, nmix, nri, char(MIX.fmixSYM_RCD{n}(nmix))) );
    end
%############################################################
for r = 2 : n
    if(Opt_Disp)
```

```
        disp(['%====================================[r]=' num2str([r])]);
        Wnr.OrdPwr_MAT{n,r}
        Wnr.PolyOrd_MAT{n,r}
        Wnr.c_VEC{n,r}
end
for rC = 1 : size(Wnr.PolyOrd_MAT{n,r}, 1)%size(Fp{n}{ nmix,nri }.wnr{r},1)
    PolyOrd_VEC       = Wnr.PolyOrd_MAT{n,r}(rC,:);
    PolyOrd_nMIXMAT     = Fp{n}{ nmix,nri }.wnr{r}{rC,1};
    PolyOrd_riMAT       = Fp{n}{ nmix,nri }.wnr{r}{rC,2};
    if(Opt_Disp)
        disp(['%---------------------------------[rC]=' num2str([rC])]);
    end
if(size( PolyOrd_nMIXMAT, 1 ) > 0)
%============================================================rMIX
[ PolyOrd_nMIXMAT_2  PolyOrd_riMAT_2  PolyOrd_coefVEC_2 ] = ...
    Poly_Condense( PolyOrd_VEC, PolyOrd_nMIXMAT, PolyOrd_riMAT, Opt_Disp);
Fp2{n}{ nmix,nri }.wnr{r}{rC,1} = PolyOrd_nMIXMAT_2;
Fp2{n}{ nmix,nri }.wnr{r}{rC,2} = PolyOrd_riMAT_2;
Fp2{n}{ nmix,nri }.wnr{r}{rC,3} = PolyOrd_coefVEC_2;
%============================================================
end
    if(Opt_Disp)
        PolyOrd_nMIXMAT
        PolyOrd_riMAT
        disp( sprintf('%#################mix = %d reduced to %d', ...
            size(PolyOrd_nMIXMAT,1), size(PolyOrd_nMIXMAT_2,1)) );
        PolyOrd_nMIXMAT_2
```

```
        PolyOrd_riMAT_2
        PolyOrd_coefVEC_2
    end
end
end
%#############################################################
end
end
end
%% ==========================================
    case{'ALL'}
for n           = 1 : length( Fp )
for nmix = 1 : size( Fp{n},1 )
for nri     = 1 : size( Fp{n},2 )
%#############################################################
for r = 2 : n
    if(Opt_Disp)
        disp(['%====================================[r]=' num2str([r])]);
        Wnr.OrdPwr_MAT{n,r}
        Wnr.PolyOrd_MAT{n,r}
        Wnr.c_VEC{n,r}
end
for rC = 1 : size(Wnr.PolyOrd_MAT{n,r},1)%size(Fp{n}{ nmix,nri }.wnr{r}, 1)
    PolyOrd_VEC       = Wnr.PolyOrd_MAT{n,r}(rC,:);
    PolyOrd_nMIXMAT     = Fp{n}{ nmix,nri }.wnr{r}{rC,1};
    PolyOrd_riMAT       = Fp{n}{ nmix,nri }.wnr{r}{rC,2};
    if(Opt_Disp)
```

```
        disp(['%----------------------------------[rC]=' num2str([rC])]);
        PolyOrd_nMIXMAT
        PolyOrd_riMAT
    end
if(size( PolyOrd_nMIXMAT, 1 ) > 0)
%============================================================rMIX
[ PolyOrd_nMIXMAT_2  PolyOrd_riMAT_2  PolyOrd_coefVEC_2 ] = ...
    Poly_Condense( PolyOrd_VEC, PolyOrd_nMIXMAT, PolyOrd_riMAT, Opt_Disp);
Fp2{n}{ nmix,nri }.wnr{r}{rC,1} = PolyOrd_nMIXMAT_2;
Fp2{n}{ nmix,nri }.wnr{r}{rC,2} = PolyOrd_riMAT_2;
Fp2{n}{ nmix,nri }.wnr{r}{rC,3} = PolyOrd_coefVEC_2;
%============================================================
end
    if(Opt_Disp)
        disp(['%###############']);
        PolyOrd_nMIXMAT_2
        PolyOrd_riMAT_2
        PolyOrd_coefVEC_2
        disp( sprintf('#mix = %d reduced to %d', ...
            size(PolyOrd_nMIXMAT,1), size(PolyOrd_nMIXMAT_2,1)) );
    end
end
end
%#############################################################################
end
end
end
```

```
%% ============================================
end

%% #####################################################
function [ PolyOrd_nMIXMAT_2  PolyOrd_riMAT_2  PolyOrd_coefVEC_2 ] = ...
    Poly_Condense( PolyOrd_VEC, PolyOrd_nMIXMAT, PolyOrd_riMAT, Opt_Disp)

PolyOrd_nMIXMAT_2 = PolyOrd_nMIXMAT(1,:);% Init
PolyOrd_riMAT_2     = PolyOrd_riMAT(1,:);% Init
PolyOrd_coefVEC_2  = [1];% Init

for rMIX = 2 : size( PolyOrd_nMIXMAT, 1 )
    isFOUND = 0;
    %-----------------------------------------------------------------
    for rMIX_2 = 1 : size( PolyOrd_nMIXMAT_2, 1 )
        [ isFOUND ] = isSAME_Permute( PolyOrd_VEC, ...
            PolyOrd_nMIXMAT(rMIX,:),        PolyOrd_riMAT(rMIX,:), ...
            PolyOrd_nMIXMAT_2(rMIX_2,:),   PolyOrd_riMAT_2(rMIX_2,:) );
        if( isFOUND )% already exist in *_2
            break
        end
    end
    %-----------------------------------------------------------------
    if( isFOUND )% already exist in *_2
        PolyOrd_coefVEC_2( rMIX_2 ) = PolyOrd_coefVEC_2( rMIX_2 ) + 1;
        if( Opt_Disp )
            disp(['%###############[rMIX rMIX_2]=' num2str([rMIX rMIX_2])]);
```

```
        end
    else% NOT in *_2
        PolyOrd_nMIXMAT_2   = [PolyOrd_nMIXMAT_2; PolyOrd_nMIXMAT(rMIX,:)];
        PolyOrd_riMAT_2     = [PolyOrd_riMAT_2;   PolyOrd_riMAT(rMIX,:)];
        PolyOrd_coefVEC_2  = [PolyOrd_coefVEC_2; 1];
    end
    %-----------------------------------------------------------------
end

%% ######################################################
function [ isFOUND ] = isSAME_Permute( Ord_VEC, ...
    mixVEC, riVEC, mixVEC2, riVEC2 )

ordPERM = perms( Ord_VEC );
mixPERM = perms( mixVEC );
riPERM  = perms( riVEC );

isFOUND = 0;
for nPERM = 1 : size( mixPERM, 1)
ordBOOL = (ordPERM(nPERM,:) == Ord_VEC);
    if isempty(find(ordBOOL == 0))% all 1's
        %------------------------------
      mixBOOL = (mixPERM(nPERM,:) == mixVEC2);
        riBOOL  = (riPERM(nPERM,:)  == riVEC2);
        if isempty(find(mixBOOL == 0)) ...
            && isempty(find(riBOOL == 0))%all 1s
            isFOUND = 1;
```

```
            break;
        end
        %------------------------------
    end
end
```

INL_Mult Function

```
function [ Mn  Fm  Fm_ ] = INL_Mult( N, MIX, ...
    Opt_Disp)
if ~exist('Opt_Disp','var')
    Opt_Disp = 0;
end

%% ############################################## Mn (like Wnr for INL_Poly)
for n = 1 : N
    Mn.PolyOrd_MAT{n} = [];% initialize
end

for nU = 1 : N
for nV = 1 : N
    n = nV + nU;
    if( n <= N )
        Mn.PolyOrd_MAT{n} = [Mn.PolyOrd_MAT{n};  [nV  nU]];
    end
end
end
```

```
%% ############################################## init
for n = 1 : N
    Fm_{n} = zeros( size(MIX.fmix_RCD{n},1), 2);
for nRESP = 1 : size( MIX.fmix_RCD{n},1 )
%------------------------------------------------------------- init .wnr
for rC = 1 : size( Mn.PolyOrd_MAT{n},1 )
    for nRI = 1 : 2
        Fm{n}{nRESP,nRI}.Mn{rC,1} = [];% PolyOrd_nMIXMAT
        Fm{n}{nRESP,nRI}.Mn{rC,2} = [];% PolyOrd_riMAT
        Fm{n}{nRESP,nRI}.Mn{rC,3} = [];% PolyOrd_coefVEC
    end
end
%------------------------------------------------------------- init .fmix
for k = 1 : N
    for nRI = 1 : 2
        Fm{n}{nRESP,nRI}.fmix{k,1} = zeros( size( MIX.fmix_RCD{k},1), 2);
    end
end
%-------------------------------------------------------------------
end
end

%% ##################################################### Fm  Fm_
for n = 2 : N
for rC = 1 : size( Mn.PolyOrd_MAT{n},1 )
    nV          = Mn.PolyOrd_MAT{n}(rC,1);
    nU          = Mn.PolyOrd_MAT{n}(rC,2);
```

```
%================================================================
    %iterate all possible nmixRCD/nRI @(current position in VEC)
    %Recurrsive func call bc: VEC is variable length, can't use for-loop
for riV = 1 : -2 : -1
    mixStartV = 1;
    if (riV == -1) && isequal( MIX.fmix_RCD{nV}(1,:), ...
            zeros( size(MIX.fmix_RCD{nV}(1,:)) ) )
        mixStartV = 2;% skip double-counting if DC (sorted to be 1st-pt)
    end
for riU = 1 : -2 : -1
    mixStartU = 1;
    if (riU == -1) && isequal( MIX.fmix_RCD{nU}(1,:), ...
            zeros( size(MIX.fmix_RCD{nU}(1,:)) ) )
        mixStartU = 2;% skip double-counting if DC (sorted to be 1st-pt)
    end
%----------------------------------------------------------------
for nmixRCDV = mixStartV : size( MIX.fmix_RCD{nV},1 )
for nmixRCDU = mixStartU : size( MIX.fmix_RCD{nU},1 )
    fmixVEC_V   = MIX.fmix_RCD{nV}(nmixRCDV,:);
    fmixVEC_U  = MIX.fmix_RCD{nU}(nmixRCDU,:);
    %-----------------------------------------------------
    % calc Result fmix_VEC ~OUTPUT
    fmixVEC         = fmixVEC_V * riV + fmixVEC_U * riU;
    [ nmixRCD  nRI ] = Mix_RCDsearch( fmixVEC, MIX.fmix_RCD{n});
        Fm{n}{ nmixRCD, nRI}.Mn{rC,1} = [Fm{n}{ nmixRCD, nRI}.Mn{rC,1};...
                [nmixRCDV nmixRCDU]];
        Fm{n}{ nmixRCD, nRI}.Mn{rC,2} = [Fm{n}{ nmixRCD, nRI}.Mn{rC,2};...
```

```
            [riV        riU]];
        Fm{n}{ nmixRCD, nRI}.fmix{n,1}(nmixRCD, nRI) = 1;
        Fm_{n}( nmixRCD, nRI) = 1;% indicate fIM generated
    %-----------------------------------------------------
        if(riV == -1)
            Fm{n}{nmixRCD, nRI}.fmix{nV,1}(nmixRCDV, 2) = 1;
        else
            Fm{n}{nmixRCD, nRI}.fmix{nV,1}(nmixRCDV, 1) = 1;
        end
        if(riU == -1)
            Fm{n}{nmixRCD, nRI}.fmix{nU,1}(nmixRCDU, 2) = 1;
        else
            Fm{n}{nmixRCD, nRI}.fmix{nU,1}(nmixRCDU, 1) = 1;
        end
    %-----------------------------------------------------
end
end
%-------------------------------------------------------------------
end
end
%===================================================================
end
end
```

INL_Mult_Condense Function

```
function [ Fm2 ] = INL_Mult_Condense( Fm, Mn, MIX, Opt, ...
    Opt_Disp)
```

```
if ~exist('Opt_Disp','var')
    Opt_Disp = 0;
end

switch Opt
%% ============================================
    case{'ALL'}
for n    = 1 : length( Fm )
for nmixRCD = 1 : size( Fm{n},1 )
for nRI     = 1 : size( Fm{n},2 )
  if(Opt_Disp)
    disp(sprintf('=========n=%-5d  nmixRCD=%-5d  nRI=%-5d  fmixSYM=%-s',...
    n, nmixRCD, nRI, char(MIX.fmixSYM_RCD{n}(nmixRCD))) );
    Mn.PolyOrd_MAT{n}
  end
%############################################################
for rC = 1 : size( Mn.PolyOrd_MAT{n}, 1 )%size( Fm{n}{nmixRCD, nRI}.Mn, 1 )
    PolyOrd_nMIXMAT     = Fm{n}{nmixRCD, nRI}.Mn{rC,1};
    PolyOrd_riMAT       = Fm{n}{nmixRCD, nRI}.Mn{rC,2};
    if(Opt_Disp)
        disp(['%--------------------------[rC]=' num2str([rC])]);
    end
if(size( PolyOrd_nMIXMAT, 1 ) > 0)
%============================================================rMIX
[PolyOrd_nMIXMAT_2  PolyOrd_riMAT_2  PolyOrd_coefVEC_2 ] =Mult_Condense(...
    PolyOrd_nMIXMAT, PolyOrd_riMAT, Opt_Disp);
Fm2{n}{ nmixRCD,nRI }.Mn{rC,1} = PolyOrd_nMIXMAT_2;
```

```
Fm2{n}{ nmixRCD,nRI }.Mn{rC,2} = PolyOrd_riMAT_2;
Fm2{n}{ nmixRCD,nRI }.Mn{rC,3} = PolyOrd_coefVEC_2;
%============================================================
end
    if(Opt_Disp)
        if( size(PolyOrd_nMIXMAT,1) ~= size(PolyOrd_nMIXMAT_2,1) )
            PolyOrd_nMIXMAT
            PolyOrd_riMAT
            disp( sprintf('%################mix = %d reduced to %d', ...
                size(PolyOrd_nMIXMAT,1), size(PolyOrd_nMIXMAT_2,1)) );
            PolyOrd_coefVEC_2
            PolyOrd_nMIXMAT_2
            PolyOrd_riMAT_2
        end
    end
end
%############################################################
end
end
end
%% ===========================================
end

%% ########################################################
function [ PolyOrd_nMIXMAT_2  PolyOrd_riMAT_2  PolyOrd_coefVEC_2 ] = ...
    Mult_Condense( PolyOrd_nMIXMAT, PolyOrd_riMAT, Opt_Disp)
```

```
PolyOrd_nMIXMAT_2 = PolyOrd_nMIXMAT(1,:);% Init
PolyOrd_riMAT_2     = PolyOrd_riMAT(1,:);% Init
PolyOrd_coefVEC_2  = [1];% Init

for rMIX = 2 : size( PolyOrd_nMIXMAT, 1 )
    isFOUND = 0;
    %------------------------------------------------------ find in *_2
    for rMIX_2 = 1 : size( PolyOrd_nMIXMAT_2, 1 )
        mixBOOL = (PolyOrd_nMIXMAT(rMIX,:) == PolyOrd_nMIXMAT_2(rMIX_2,:));
        riBOOL  = (PolyOrd_riMAT(rMIX,:)   == PolyOrd_riMAT_2(rMIX_2,:));
        if isempty(find(mixBOOL == 0)) && isempty(find(riBOOL == 0))%all 1s
            isFOUND = 1;
            break;
        end
    end
    %------------------------------------------------------
    if( isFOUND )% IN *_2
        PolyOrd_coefVEC_2( rMIX_2 ) = PolyOrd_coefVEC_2( rMIX_2 ) + 1;
        if( Opt_Disp )
            disp(['%################ [rMIX  rMIX_2]=' ...
                num2str([rMIX  rMIX_2])]);
        end
    else% NOT IN *_2
        PolyOrd_nMIXMAT_2   = [PolyOrd_nMIXMAT_2; PolyOrd_nMIXMAT(rMIX,:)];
        PolyOrd_riMAT_2     = [PolyOrd_riMAT_2;   PolyOrd_riMAT(rMIX,:)];
        PolyOrd_coefVEC_2  = [PolyOrd_coefVEC_2;   1];
    end
```

```
    %-----------------------------------------------------------------

end
```

A.3 Define Circuit

A.3.1 Differential Pair as Single-Balanced Mixer

```
%========================================================= Lin
%Ckt.CharCEL_Ymat    = { 'gL+s*cL'       '0'           '-gT'; ...
%                        '0'             'gL+s*cL'     '-gT'; ...
%                        '0'             '0'           '-2*gT' };
Ckt.CharCEL_Ymat    = { 'G'           '0'           '-gT'; ...
                        '0'             'G'             '-gT'; ...
                        '0'             '0'           '-2*gT' };
Ckt.CharCEL_ILIN = { '-gT*Vi/2'      'gT*Vi/2'       'Is' };
%-----------------------------------------------------------
Ckt.CharCEL_Vdep  = { 'Vo1'          'Vo2'           'Ve' };
%--------------------------
Ckt.CharCEL_Vsrc = {'Vi' 'Is'};
Ckt.CharCEL_VsrcTone = {'Vi' 'Is'};
% {A1 A2}->creat voltage spectral (#source <= #tone),
%    {'Vin' 'Vin'} if 2tone in same source
%--------------------------
Ckt.CharCEL_R       = {'gT'};% freqIndep
%--------------------------
Ckt.CharCEL_Z       = {'G'};
% freqDep ('s' 'S' SubSYM/SubNUM as 'k*f1+l*f2', offlimits)
%============================================================ NL
Ckt.INL_types      = {'poly'};
Ckt.INL_Vdep_type = {{'poly'} {'poly'} {'poly' 'poly'}};% {} if no INL
Ckt.INL_Vdep_eo   = {{'eo'}   {'eo'}   {'eo'   'eo'}};
```

```
Ckt.INL_Vdep_Sign  = {[-1]     [-1]     [-1     -1]};
Ckt.INL_Vdep_gNL   = {{'gT'}   {'gT'}   {'gT'   'gT'}};% No Sign, only SYM
%----------
Ckt.INL_Vdep        = {{'Vi/2-Ve'} {'-Vi/2-Ve'} {'Vi/2-Ve'  '-Vi/2-Ve'}};
Ckt.INL_Vdep_SymCEL = {{{'Ve'}}    {{'Ve'}}     {{'Ve'}     {'Ve'}}};
Ckt.INL_Vdep_IDX = {{[3]}       {[3]}        {[3]        [3]}};
Ckt.INL_Vsrc_SymCEL = {{{'Vi'}}    {{'Vi'}}     {{'Vi'}     {'Vi'}}};
Ckt.INL_Vsrc_IDX = {{[1]}       {[1]}        {[1]        [1]}};
```

A.3.2 RF Front-End Behavioral Model

```
%========================================================== Lin
Ckt.CharCEL_Ymat    = ...
{ '1/Z1+1/Za''0'       '0'         '0'         '0'         '0'; ...
  'A'           '1/Zh'      '0'         '0'         '0'         '0'; ...
  '0'        'H'        '-1'        '0'         '0'         '0'; ...
  '0'        '0'        '0'         '1/Z2+1/Zb''0'         '0'; ...
  '0'        '0'        '0'         'B'         '1/Zm2'     '0'; ...
  '0'        '0'        '0'         '0'         '0'         '1'};
Ckt.CharCEL_ILIN = { 'V1/Z1' '0' '0' 'V2/Z2' '0' '0'};
%Ckt.CharCEL_Ymat   = ...
%{ 'G1+Ga'     '0'       '0'         '0'         '0'         '0'; ...
%  'A'       'Gh'      '0'         '0'         '0'         '0'; ...
%  '0'        'H'        '-1'        '0'         '0'         '0'; ...
%  '0'        '0'        '0'         'G2+Gb'     '0'         '0'; ...
%  '0'        '0'        '0'         'B'         'Gm2'    '0'; ...
%  '0'        '0'        '0'         '0'         '0'         '1'};
%Ckt.CharCEL_ILIN = { 'V1*G1' '0' '0' 'V2*G2' '0' '0'};
```

```
%-----------------------------------------------------------
Ckt.CharCEL_Vdep  = { 'Va' 'Vh' 'Vm1' 'Vb' 'Vm2' 'Vm3'};
%--------------------------
Ckt.CharCEL_Vsrc = {'V1' 'V2'};
Ckt.CharCEL_VsrcTone = {'V1' 'V2'};
%--------------------------
Ckt.CharCEL_R       = {};% freqIndep
%--------------------------
Ckt.CharCEL_Z      = {'Z1' 'Z2' 'Za' 'Zb' 'Zh' 'A' 'B' 'H' 'Zm2'};
%Ckt.CharCEL_Z       = {'G1' 'G2' 'Ga' 'Gb' 'Gh' 'A' 'B' 'H' 'Gm2'};
%=========================================================== NL
Ckt.INL_types      = {'poly' 'multiply'};
Ckt.INL_Vdep_type = {{'poly'} {'poly'} {} {'poly'} {'poly'} {'multiply'}};
Ckt.INL_Vdep_eo   = {{'eo'}   {'eo'}   {}   {'eo'}   {'eo'}   {'eo'}};
Ckt.INL_Vdep_Sign  = {[-1]     [-1]     []   [-1]     [-1]     [1]};
Ckt.INL_Vdep_gNL   = {{'az'}   {'ag'}   {}   {'bz'}   {'bg'}   {'m'}};
%----------
Ckt.INL_Vdep        = {{'Va'} {'Va'} {} {'Vb'} {'Vb'} {'Vm1'; 'Vm2'}};
Ckt.INL_Vdep_SymCEL = {{{'Va'}} {{'Va'}} {{}} {{'Vb'}} {{'Vb'}} ...
    {{'Vm1'}; {'Vm2'}}};
Ckt.INL_Vdep_IDX = {{[1]} {[1]} {[]} {[4]} {[4]} {[3]; [5]}};
Ckt.INL_Vsrc_SymCEL = {{{}} {{}} {{}} {{}} {{}} {{}; {}}};
Ckt.INL_Vsrc_IDX = {{[]} {[]} {[]} {[]} {[]} {[]; []}};
```

A.4 Circuit Analysis

A.4.1 Main Script

```
%% ############################################### IM response of Interest
%Fresp.fRESP_MAT = [ 0    0; -1    1;  1    1;  2    0;  0    2];% 2nd
Fresp.fRESP_MAT = [ 1    0;  0    1;  2   -1;  2    1; ...
                    -1    2;  1    2;  3    0;  0    3];% 3rd
[ Fresp ]   = Fresp_SelectFMIX( Fresp, 'fRESP_MAT', ...
        Fp2, MIX, Wnr, 1,1);%Opt_Print, Opt_Disp


%% ############################################### Ckt
Ckt.N = N;
Ckt.Q = Q;
[ Ckt ]     = Ckt_Gen2( Ckt, Fresp, MIX, 1);
%------------------------------
[ Ckt ]     = Ckt_SolnLin3( Ckt, Fresp, MIX, 1);
%------------------------------
[ Ckt ]     = Ckt_SolnNL4( Ckt, Fresp,    MIX, Wnr, Mn, Fp2, Fm2, 1 );
%------------------------------
[ Ckt ]     = Ckt_Soln2( Ckt, Fresp, 1 );
```

A.4.2 Functions

Ckt_Gen2 Function

```
function [ Ckt ] = Ckt_Gen2( Ckt, Fresp, MIX, ...
    Opt_Disp)
if ~exist('Opt_Disp','var')
    Opt_Disp = 0;
```

```
end


Ckt.NumV = length( Ckt.CharCEL_Vdep );
%% ##########################
%CharCEL => SYM_[Ymat/ILIN/  INLvdep],  SYM_[R/gNL/  zBD.(Z)],  SYM_[A]
%============================================================ Lin
%[ Ckt.SYM_Vdep ] = CHARcel2SYMmat( Ckt.CharCEL_Vdep ).';
%[ Ckt.SYM_Vsrc ] = CHARcel2SYMmat( Ckt.CharCEL_Vsrc ).';
%------------------------------ usable as SymCEL
[ Ckt.SYM_Ymat ]    = CHARcel2SYMmat( Ckt.CharCEL_Ymat );
[ Ckt.SYM_ILIN ]    = CHARcel2SYMmat( Ckt.CharCEL_ILIN ).';
%-------------------------- SYM_R
[ Ckt.SYM_R ]       = mat2cell( CHARcel2SYMmat( Ckt.CharCEL_R ), ...
                                ones(1,size(Ckt.CharCEL_R,1)), ...
                                ones(1,size(Ckt.CharCEL_R,2)) );
%-------------------------- SYM_ZBD
for nz = 1 : length( Ckt.CharCEL_Z )
    Z = Ckt.CharCEL_Z{ nz };
for nBD = 1 : length( MIX.fmix_BD )
for nmixBD = 1 : size( MIX.fmix_BD{nBD},1 )
    Ckt.SYM_ZBD.( Z ){nBD}(nmixBD,:) = ...
        { sym([Z  MIX.fmixCHAR_BD{nBD}{nmixBD,1}]) ...
       sym([Z  MIX.fmixCHAR_BD{nBD}{nmixBD,2}]) };
end
end
end
%============================================================ NL
```

```
%-------------------------- SYM_INLvdep
for nV = 1 : Ckt.NumV
for nL = 1 : length( Ckt.INL_Vdep_type{nV} )
for k = 1 : size( Ckt.INL_Vdep{nV}, 1 )
if ~isempty( Ckt.INL_Vdep{nV}{k,nL} )
    Ckt.SYM_INLvdep{nV}{k,nL} = sym( Ckt.INL_Vdep{nV}{k,nL} );
else
    Ckt.SYM_INLvdep{nV}{k,nL} = [];% sym( [] )
end
end
end
end
%-------------------------- SYM_gNL
    Ckt.SYM_gNL = struct;
for nV = 1 : Ckt.NumV
for nL = 1 : length( Ckt.INL_Vdep_type{nV} )
    gNL = Ckt.INL_Vdep_gNL{nV}{nL};
if ~isfield( Ckt.SYM_gNL, gNL )
switch Ckt.INL_Vdep_type{nV}{nL}
%-----------------------------------------
case{'multiply'}% mixer
        Ckt.SYM_gNL.( gNL )(2) = sym( [gNL] );
%-----------------------------------------
case{'poly','capNL','indNL'}% gm, capNL, indNL
        for r = 2 : MIX.N
            switch Ckt.INL_Vdep_eo{nV}{nL}
                case{'eo'}
```

```
                    Ckt.SYM_gNL.( gNL )(r) =sym([gNL '_' num2str(r)]);
                case{'e'}
                    if IsEven_Func( r )
                        Ckt.SYM_gNL.( gNL )(r) =sym([gNL '_' num2str(r)]);
                    end
                case{'o'}
                    if ~IsEven_Func( r )
                        Ckt.SYM_gNL.( gNL )(r) =sym([gNL '_' num2str(r)]);
                    end
            end
        end
%-----------------------------------------
end
end
end
end
% SYM_gNL_FLAG
%========================================================== Mtone Source
for q = 1 : Ckt.Q
    Ckt.SYM_A(q,:) = { sym(['A' num2str(q)]) sym(['A' num2str(q) '_']) };
end

%% ############################ Soln.( Vdep, Vsrc )
if(Opt_Disp)
    disp([ '%############################ Ckt.Soln( Vdep )' ])
end
    Vdep1               = Ckt.CharCEL_Vdep{1};
```

```
    Ckt.Soln.( Vdep1 ) = Soln_Init( Fresp );
for nV = 2 : Ckt.NumV
    Vdep                = Ckt.CharCEL_Vdep{ nV };
    Ckt.Soln.( Vdep ) = Ckt.Soln.( Vdep1 );% Soln_Init( Fresp )
end
for nSRC = 1 : length( Ckt.CharCEL_Vsrc )
    Vsrc                = Ckt.CharCEL_Vsrc{ nSRC };
    Ckt.Soln.( Vsrc ) = Ckt.Soln.( Vdep1 );% Soln_Init( Fresp )
end

if(Opt_Disp)
    disp([ '%############################ Ckt.Soln.( Vsrc )' ])
end
for n = 1 : 1
for nmixRCD = 1 : length( Ckt.CharCEL_VsrcTone )
    %-------------------------
    [ nSRC xXORy_IDX yVEC_IDX yXORy_IDX] = Find_VecIDX_Func(...
            Ckt.CharCEL_Vsrc, ...
            Ckt.CharCEL_VsrcTone( nmixRCD ), 'VECTOR', [], 0);
    Vsrc = Ckt.CharCEL_Vsrc{ nSRC };
    Ckt.Soln.( Vsrc ){n}{nmixRCD,1} = Ckt.SYM_A{nmixRCD,1};% / 2
    Ckt.Soln.( Vsrc ){n}{nmixRCD,2} = Ckt.SYM_A{nmixRCD,2};% / 2
    %-------------------------
    if(Opt_Disp)
        disp([Vsrc '{' num2str(n) '}{' num2str(nmixRCD) ':}' ])
        Ckt.Soln.( Vsrc ){n}{nmixRCD,:}% DEBUG
    end
```

```
    %-------------------------
end
end

if(Opt_Disp)
    Ckt.Soln% DEBUG
end

%% ***************************************************
function [ V ] = Soln_Init( Fresp )

for n = 1 : length( Fresp.fmixFLAG_RCD )
for nmixRCD = 1 : size( Fresp.fmixFLAG_RCD{n},1 )
for nRI = 1 : 2% length( riVEC )
        V{n}{nmixRCD,nRI} = sym('0');
end
end
end
```

Ckt_SolnLin3 Function

```
function [ Ckt ] = Ckt_SolnLin3( Ckt, Fresp, MIX, ...
    Opt_Disp )
riFCT  = [1 -1];

%% ############################ Lin-Soln: Ckt.Soln.( Vdep ){n}{nmixRCD,nRI}
n = 1;
for nmixRCD = 1 : size( Fresp.fmixFLAG_RCD{n},1 )
```

```
    if(Opt_Disp)
        disp(['================= [nmixRCD n]=' num2str([nmixRCD  n])]);
    end
for nRI = 1 : 2
    if(Opt_Disp) disp(['-------- nRI=' num2str(nRI)]);end
%================================================ Lin Real/Imag resp
    nBD     = MIX.IdxBD_RCD{n}(nmixRCD,1);
    nmixBD = MIX.IdxBD_RCD{n}(nmixRCD,2);
Z_CEL   = {};
    for nz = 1 : length( Ckt.CharCEL_Z )
        Z               = Ckt.CharCEL_Z{ nz };
        Z_CEL{ nz }     = Ckt.SYM_ZBD.( Z ){nBD}{nmixBD,nRI};
    end
Vsrc_CEL = {};
    for nSRC = 1 : length( Ckt.CharCEL_Vsrc )
        Vsrc            = Ckt.CharCEL_Vsrc{ nSRC };
        Vsrc_CEL{ nSRC } = Ckt.Soln.( Vsrc ){n}{nmixRCD,nRI};
    end
%----------------------
    %{Ckt.CharCEL_Z{:}}
Ymat = subs( Ckt.SYM_Ymat, ...
        {Ckt.CharCEL_Z{:}  's'}, ...
        {Z_CEL{:}   j*2*pi * riFCT(nRI) * MIX.fmixSYM_RCD{n}(nmixRCD))} );
%----------------------
    %Ckt.SYM_ILIN
    %{ Ckt.CharCEL_Vsrc{:} }
    %[ Vsrc_CEL{:}         ]
```

```
Ilin_VEC = subs( Ckt.SYM_ILIN, ...
        {Ckt.CharCEL_Vsrc{:} Ckt.CharCEL_Z{:} 's'}, ...
        {Vsrc_CEL{:}            Z_CEL{:}          ...
            (j*2*pi * riFCT(nRI) * MIX.fmixSYM_RCD{n}(nmixRCD))} )
V_VEC = Ymat \ Ilin_VEC% V_VEC * 2, Ilin_VEC / 2
    for nV = 1 : Ckt.NumV
        Vdep = Ckt.CharCEL_Vdep{ nV };
        Ckt.Soln.( Vdep ){n}{nmixRCD,nRI} = V_VEC(nV);
    end
%================================================
end
end
```

Ckt_SolnNL4 Function

```
function [ Ckt ] = Ckt_SolnNL4( Ckt, Fresp, ...
    MIX, Wnr, Mn, Fp2, Fm2, ...
    Opt_Disp )
if ~exist('Opt_Disp','var')
    Opt_Disp = 0;
end

%========================================== Output
Fresp.fmixFLAG_RCD3 = {};% indicat path traveled {n,1}{nmixRCD,:}=[R I]
    Fresp.fmixFLAG_RCD3{1} = Fresp.fmixFLAG_RCD{1};%LinSoln computed
    for n = 2 : MIX.N
        Fresp.fmixFLAG_RCD3{n} = zeros(size( MIX.fmix_RCD{n} ));%Init
    end
```

```
%% ########################## NL-Soln:    Ckt.Soln.( Vdep ){n}{nRESP,nRI}
for nRESP = 1 : Fresp.NumResp
if(Opt_Disp) ...
        disp(['====================== [nRESP]=' num2str([nRESP])]);
    end
%=========================================
for nn = 1 : size( Fresp.IdxRCD_fRESP{ nRESP },1 )
    n       = Fresp.IdxRCD_fRESP{ nRESP }(nn,1);
    nmixRCD = Fresp.IdxRCD_fRESP{ nRESP }(nn,2);
    nRI     = 1;% for measured
if(Opt_Disp) disp(['-----------------[nRESP n]=' num2str([nRESP  n])' ...
          ' char( Fresp.fmixSYM_fRESP{ nRESP }(nn) )]);end
    %---------------------- Recur given (n, nmixRCD, nRI)
    if(Fresp.fmixFLAG_RCD3{n}(nmixRCD, nRI) == 0)% if path not traveled
        [ Ckt Fresp ] = Recur( n, nmixRCD, nRI, ...
            Ckt, Fresp, MIX, Wnr, Mn, Fp2, Fm2, Opt_Disp);
    end
%=========================================
end
end

%% ################################################ Fp2{n}{nmixRCD, nRI}
function [ Ckt Fresp ] = Recur( n, nmixRCD, nRI, ...
    Ckt, Fresp, MIX, Wnr, Mn, Fp2, Fm2, ...
    Opt_Disp)
if(Opt_Disp)  disp(['%###################[n, nmixRCD, nRI] ' ...
```

```
    num2str([n, nmixRCD, nRI])]); end
riFCT  = [1 -1];


if(n == 1)
%================================================================
    % LinSoln computed already
%================================================================
else%(n > 1)
%##################################################### calc INL_VEC
INL_VEC  = sym(zeros( Ckt.NumV,1 ));% (nV)
FlagPOLY = 0;%set once bc FuncCall find INL(gm, capNL, indNL) simultaneous
for nTYPE = 1 : length( Ckt.INL_types )
switch Ckt.INL_types{nTYPE}
        %-----------------------------------------
        case{'poly','capNL','indNL'}% gm, capNL, indNL
        if( FlagPOLY == 0 )
            [ INLpoly_VEC  INLcap_VEC  INLind_VEC ] = Poly( ...
                n, nmixRCD, nRI, ...
                Ckt, Fresp, MIX, Wnr, Mn, Fp2, Fm2, 1);
            INL_VEC     = INL_VEC + INLpoly_VEC + INLcap_VEC + INLind_VEC ;
            FlagPOLY = 1;
            %set once bc FuncCall find INL(gm, capNL, indNL) simultaneously
        end
        %-----------------------------------------
        case{'multiply'}% mixer
            [ INLmult_VEC ] = Multiply( n, nmixRCD, nRI, ...
                Ckt, Fresp, MIX, Wnr, Mn, Fp2, Fm2, 1);
```

```
            INL_VEC     = INL_VEC + INLmult_VEC;
        %-----------------------------------------
        otherwise
            disp('unknown Ckt.INL_types');
    end
end
%##################################################### calc V_VEC
    nBD     = MIX.IdxBD_RCD{n}(nmixRCD,1);
    nmixBD = MIX.IdxBD_RCD{n}(nmixRCD,2);
Z_CEL = {};
    for nz = 1 : length( Ckt.CharCEL_Z )
        Z         = Ckt.CharCEL_Z{ nz };
        Z_CEL{ nz } = Ckt.SYM_ZBD.( Z ){nBD}{nmixBD,nRI};
    end
Ymat = subs( Ckt.SYM_Ymat, ...
        {Ckt.CharCEL_Z{:}   's'}, ...
        {Z_CEL{:}  j*2*pi * riFCT(nRI) * MIX.fmixSYM_RCD{n}(nmixRCD)} );
V_VEC = Ymat \ INL_VEC;
    for nV = 1 : Ckt.NumV
        Vdep        = Ckt.CharCEL_Vdep{ nV };
        Ckt.Soln.( Vdep ){n}{nmixRCD,nRI} = V_VEC(nV);
    end
%#####################################################
%=============================================================
end
    Fresp.fmixFLAG_RCD3{n}(nmixRCD, nRI) = 1;% indicate path was traveled
return
```

```
%% ########################################################
function [ INLmult_VEC ] = Multiply( n, nmixRCD, nRI, ...
    Ckt, Fresp, MIX, Wnr, Mn, Fp2, Fm2, ...
    Opt_Disp)
INLmult_VEC = sym(zeros( Ckt.NumV,1 ));% (nV)

%############################################################
r = 2;
  Vr_VecCEL    = Vrc_Init( 0, 1, Ckt.INL_Vdep );% {1,nV}{nL}
    C_r       = 1 / 2^(r-1);
%======================================================== OrdPwr_MAT, c_VEC
  Vc_MatCEL     = Vrc_Init( 0, size( Mn.PolyOrd_MAT{n}, 1 ), ...
        Ckt.INL_Vdep );% {rC,nV}{nL}
for rC = 1 : size( Mn.PolyOrd_MAT{n}, 1 )
    if(Opt_Disp)  disp(['%----------------------[rC] ' num2str([rC])]); end
PolyOrd_VEC         = Mn.PolyOrd_MAT{n}(rC,:);
    PolyOrd_nMIXMAT     = Fm2{n}{ nmixRCD,nRI }.Mn{rC,1};
    PolyOrd_riMAT       = Fm2{n}{ nmixRCD,nRI }.Mn{rC,2};
    PolyOrd_coefVEC     = Fm2{n}{ nmixRCD,nRI }.Mn{rC,3};
    if(Opt_Disp)
        PolyOrd_VEC% DEBUG
        PolyOrd_nMIXMAT% DEBUG
        PolyOrd_riMAT% DEBUG
        PolyOrd_coefVEC% DEBUG
    end
```

```
%------------------------------------------------------------ rMIX
  Vrc_MatCEL    = Vrc_Init( 1, size( PolyOrd_nMIXMAT,1), Ckt.INL_Vdep );
for rMIX = 1 : size( PolyOrd_nMIXMAT,1)
    if(Opt_Disp)  disp(['%----------------[rMIX] ' num2str([rMIX])]); end
C_rMIX = PolyOrd_coefVEC(   rMIX);
for k = 1 : length( PolyOrd_VEC )
    if(Opt_Disp)  disp(['%----------[k] ' num2str([k])]); end
Ord_k = PolyOrd_VEC(           k);
    Mix_k = PolyOrd_nMIXMAT( rMIX,k);
    ri_k    = PolyOrd_riMAT(    rMIX,k);
        if(ri_k == -1)
            nRI_k = 2;
        else
            nRI_k = 1;
        end
%@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ calc Vdep (must call bc NonEmpty iNL)
if( Fresp.fmixFLAG_RCD3{Ord_k}(Mix_k, nRI_k) == 0)% if path not traveled
    [ Ckt Fresp ] = Recur( Ord_k, Mix_k, nRI_k, ...
        Ckt, Fresp, MIX, Wnr, Mn, Fp2, Fm2, Opt_Disp);
end
%@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ calc INL
for nV = 1 : Ckt.NumV
for nL = 1 : length( Ckt.INL_Vdep_type{nV} )% effectiv take care of empty
switch Ckt.INL_Vdep_type{nV}{nL}
%-----------------------------------------
case{'multiply'}% mixer
    Vdep_CEL = {};
```

```
    Vsrc_CEL = {};
    for nIDX = 1 : length( Ckt.INL_Vdep_SymCEL{nV}{k,nL} )
        Vdep_IDX        = Ckt.INL_Vdep_IDX{nV}{k,nL}(nIDX);
        Vdep            = Ckt.CharCEL_Vdep{ Vdep_IDX };
        Vdep_CEL{nIDX} = Ckt.Soln.( Vdep ){Ord_k}{Mix_k,nRI_k};
    end
    for nIDX = 1 : length( Ckt.INL_Vsrc_SymCEL{nV}{k,nL} )
        Vsrc_IDX        = Ckt.INL_Vsrc_IDX{nV}{k,nL}(nIDX);
        Vsrc            = Ckt.CharCEL_Vsrc{ Vsrc_IDX };
        Vsrc_CEL(nIDX) = Ckt.Soln.( Vsrc ){Ord_k}{Mix_k,nRI_k};
    end
    if(Opt_Disp)
        %[ Vdep_CEL{:} ]% DEBUG
        %[ Vsrc_CEL{:} ]% DEBUG
    end
    Vtemp = subs( Ckt.SYM_INLvdep{nV}{k,nL}, ...
        {Ckt.INL_Vdep_SymCEL{nV}{k,nL}{:} ...
            Ckt.INL_Vsrc_SymCEL{nV}{k,nL}{:}}, ...
        {Vdep_CEL{:}                        Vsrc_CEL{:}} );
%Ckt.SYM_INLvdep{nV}{k,nL}% DEBUG subs
%Vtemp% DEBUG subs
        Vrc_MatCEL{rMIX,nV}{nL} = Vrc_MatCEL{rMIX,nV}{nL}     * Vtemp;
%-----------------------------------------
end
end
end
%@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
```

```
end%----------------------[k]  Vrc_MatCEL
    for nV = 1 : Ckt.NumV
    for nL = 1 : length( Ckt.INL_Vdep_type{nV} )
        switch Ckt.INL_Vdep_type{nV}{nL}
            case{'multiply'}% mixer
                Vc_MatCEL{rC,nV}{nL} = Vc_MatCEL{rC,nV}{nL}...
                     + C_rMIX * Vrc_MatCEL{rMIX,nV}{nL};
        end
    end
    end
end%------------------------------------------ [rMIX]     Vc_MatCEL
    for nV = 1 : Ckt.NumV
    for nL = 1 : length( Ckt.INL_Vdep_type{nV} )
        switch Ckt.INL_Vdep_type{nV}{nL}
            case{'multiply'}% mixer
                Vr_VecCEL{1,nV}{nL}     = Vr_VecCEL{1,nV}{nL}...
                     + Vc_MatCEL{rC,nV}{nL};
        end
    end
    end
end%========================================== [rC]  Vr_VecCEL
    for nV = 1 : Ckt.NumV
    for nL = 1 : length( Ckt.INL_Vdep_type{nV} )
    switch Ckt.INL_Vdep_type{nV}{nL}
        %-----------------------------------------
        case{'multiply'}% mixer
            Sign    = Ckt.INL_Vdep_Sign{nV}(nL);
```

```
            gNL  = Ckt.INL_Vdep_gNL{nV}{nL};
            gNL_r   = Ckt.SYM_gNL.( gNL )(r);
            if(0)%disp('OK')
                %Ir_VEC(nV)
                %Vr_VecCEL{1,nV}{nL}
            end
            INLmult_VEC(nV)     = INLmult_VEC(nV)...
                + Sign * C_r * gNL_r  * Vr_VecCEL{1,nV}{nL};
        %-----------------------------------------
    end
    end
    end
%######################################### r = 2  Ir_VEC --> INLpoly_VEC
return


%% ########################################################
function [ INLpoly_VEC  INLcap_VEC  INLind_VEC ] = Poly( ...
    n, nmixRCD, nRI, ...
    Ckt, Fresp, MIX, Wnr, Mn, Fp2, Fm2, ...
    Opt_Disp)
INLpoly_VEC = sym(zeros( Ckt.NumV,1 ));% (nV)
INLcap_VEC = sym(zeros( Ckt.NumV,1 ));% (nV)
INLind_VEC = sym(zeros( Ckt.NumV,1 ));% (nV)
riFCT       = [1 -1];

%############################################################
```

```
for r = 2 : length( Fp2{n}{nmixRCD, nRI}.wnr )
    if(Opt_Disp)  disp(['%==================[r] ' num2str([r])]); end
  Ir_VEC       = sym(zeros( Ckt.NumV,1 ));% (nV)
  Ir_VEC_CAP    = Ir_VEC;
  Ir_VEC_IND    = Ir_VEC;
  Vr_VecCEL    = Vrc_Init( 0, 1, Ckt.INL_Vdep );% {1,nV}{nL}
  Vr_VecCEL_CAP = Vr_VecCEL;
  Vr_VecCEL_IND = Vr_VecCEL;
%============================================== OrdPwr_MAT, c_VEC
  Vc_MatCEL     = Vrc_Init( 0, size( Wnr.PolyOrd_MAT{n,r}, 1 ), ...
        Ckt.INL_Vdep );% {rC,nV}{nL}
  Vc_MatCEL_CAP = Vc_MatCEL;
  Vc_MatCEL_IND = Vc_MatCEL;
for rC = 1 : size( Wnr.PolyOrd_MAT{n,r}, 1 )
    if(Opt_Disp)  disp(['%------------------[rC] ' num2str([rC])]); end
    C_rC              = Wnr.c_VEC{n,r}(rC) / 2^(r-1);
PolyOrd_VEC         = Wnr.PolyOrd_MAT{n,r}(rC,:);
    PolyOrd_nMIXMAT     = Fp2{n}{ nmixRCD,nRI }.wnr{r}{rC,1};
    PolyOrd_riMAT       = Fp2{n}{ nmixRCD,nRI }.wnr{r}{rC,2};
    PolyOrd_coefVEC     = Fp2{n}{ nmixRCD,nRI }.wnr{r}{rC,3};
    if(Opt_Disp)
        PolyOrd_VEC% DEBUG
        PolyOrd_nMIXMAT% DEBUG
        PolyOrd_riMAT% DEBUG
        PolyOrd_coefVEC% DEBUG
    end
%------------------------------------------------------------ rMIX
```

```
  Vrc_MatCEL    = Vrc_Init( 1, size( PolyOrd_nMIXMAT,1),...
     Ckt.INL_Vdep );% {rMIX,nV}{nL}
  Vrc_MatCEL_CAP= Vrc_MatCEL;
  Vrc_MatCEL_IND= Vrc_MatCEL;
for rMIX = 1 : size( PolyOrd_nMIXMAT,1)
    if(Opt_Disp)
        disp(['%----------------------[rMIX] ' num2str([rMIX])]);
    end
C_rMIX = PolyOrd_coefVEC(   rMIX);
for k = 1 : length( PolyOrd_VEC )
    if(Opt_Disp)  disp(['%--------------[k] ' num2str([k])]); end
Ord_k = PolyOrd_VEC(           k);
    Mix_k = PolyOrd_nMIXMAT( rMIX,k);
    ri_k    = PolyOrd_riMAT(    rMIX,k);
        if(ri_k == -1)
            nRI_k = 2;
        else
            nRI_k = 1;
        end
%@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ calc Vdep (must call bc NonEmpty iNL)
if( Fresp.fmixFLAG_RCD3{Ord_k}(Mix_k, nRI_k) == 0)% if path not traveled
    [ Ckt Fresp ] = Recur( Ord_k, Mix_k, nRI_k, ...
        Ckt, Fresp, MIX, Wnr, Mn, Fp2, Fm2, Opt_Disp);
end
%@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ calc INL
for nV = 1 : Ckt.NumV
for nL = 1 : length( Ckt.INL_Vdep_type{nV} )% effectiv take care of empty
```

```
switch Ckt.INL_Vdep_type{nV}{nL}
%-----------------------------------------
case{'poly','capNL','indNL'}% gm, capNL, indNL
    Vdep_CEL = {};
    Vsrc_CEL = {};
    for nIDX = 1 : length( Ckt.INL_Vdep_SymCEL{nV}{1,nL} )
        Vdep_IDX        = Ckt.INL_Vdep_IDX{nV}{1,nL}(nIDX);
        Vdep            = Ckt.CharCEL_Vdep{ Vdep_IDX };
        Vdep_CEL{nIDX} = Ckt.Soln.( Vdep ){Ord_k}{Mix_k,nRI_k};
    end
    for nIDX = 1 : length( Ckt.INL_Vsrc_SymCEL{nV}{1,nL} )
        Vsrc_IDX        = Ckt.INL_Vsrc_IDX{nV}{1,nL}(nIDX);
        Vsrc            = Ckt.CharCEL_Vsrc{ Vsrc_IDX };
        Vsrc_CEL(nIDX) = Ckt.Soln.( Vsrc ){Ord_k}{Mix_k,nRI_k};
    end
    if(Opt_Disp)
        %[ Vdep_CEL{:} ]% DEBUG
        %[ Vsrc_CEL{:} ]% DEBUG
    end
    Vtemp = subs( Ckt.SYM_INLvdep{nV}{1,nL}, ...
        {Ckt.INL_Vdep_SymCEL{nV}{1,nL}{:} ...
            Ckt.INL_Vsrc_SymCEL{nV}{1,nL}{:}},...
        {Vdep_CEL{:}                        Vsrc_CEL{:}} );
        switch Ckt.INL_Vdep_type{nV}{nL}
            case{'poly'}
                Vrc_MatCEL{rMIX,nV}{nL}    =Vrc_MatCEL{rMIX,nV}{nL}    ...
                    * Vtemp;
```

```
            case{'capNL'}
                Vrc_MatCEL_CAP{rMIX,nV}{nL}=Vrc_MatCEL_CAP{rMIX,nV}{nL}...
                    * Vtemp;
            case{'indNL'}
                Vrc_MatCEL_IND{rMIX,nV}{nL}=Vrc_MatCEL_IND{rMIX,nV}{nL}...
                    * (Vtemp(j*2*pi*ri_k*MIX.fmixSYM_RCD{Ord_k}(Mix_k)));
        end
%-----------------------------------------
end
end
end
%@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
end%----------------------[k]                              Vrc_MatCEL
    for nV = 1 : Ckt.NumV
    for nL = 1 : length( Ckt.INL_Vdep_type{nV} )
        switch Ckt.INL_Vdep_type{nV}{nL}
            case{'poly'}
                Vc_MatCEL{rC,nV}{nL}     = Vc_MatCEL{rC,nV}{nL}     ...
                    + C_rMIX * Vrc_MatCEL{rMIX,nV}{nL};
            case{'capNL'}
                Vc_MatCEL_CAP{rC,nV}{nL} = Vc_MatCEL_CAP{rC,nV}{nL} ...
                    + C_rMIX * Vrc_MatCEL_CAP{rMIX,nV}{nL};
            case{'indNL'}
                Vc_MatCEL_IND{rC,nV}{nL} = Vc_MatCEL_IND{rC,nV}{nL} ...
                    + C_rMIX * Vrc_MatCEL_IND{rMIX,nV}{nL};
        end
    end
```

```
    end
end%---------------------------------------------- [rMIX]     Vc_MatCEL
    for nV = 1 : Ckt.NumV
    for nL = 1 : length( Ckt.INL_Vdep_type{nV} )
        switch Ckt.INL_Vdep_type{nV}{nL}
            case{'poly'}
                Vc_MatCEL{rC,nV}{nL}     = Vc_MatCEL{rC,nV}{nL}     * C_rC;
            case{'capNL'}
                Vc_MatCEL_CAP{rC,nV}{nL} = Vc_MatCEL_CAP{rC,nV}{nL} * C_rC;
            case{'indNL'}
                Vc_MatCEL_IND{rC,nV}{nL} = Vc_MatCEL_IND{rC,nV}{nL} * C_rC;
        end
    end
    end
    for nV = 1 : Ckt.NumV
    for nL = 1 : length( Ckt.INL_Vdep_type{nV} )
        switch Ckt.INL_Vdep_type{nV}{nL}
            case{'poly'}
                Vr_VecCEL{1,nV}{nL}     = Vr_VecCEL{1,nV}{nL}     ...
                    + Vc_MatCEL{rC,nV}{nL};
            case{'capNL'}
                Vr_VecCEL_CAP{1,nV}{nL} = Vr_VecCEL_CAP{1,nV}{nL} ...
                    + Vc_MatCEL_CAP{rC,nV}{nL};
            case{'indNL'}
                Vr_VecCEL_IND{1,nV}{nL} = Vr_VecCEL_IND{1,nV}{nL} ...
                    + Vc_MatCEL_IND{rC,nV}{nL};
        end
```

```
    end
    end
end%========================================== [rC]       Vr_VecCEL
    for nV = 1 : Ckt.NumV
    for nL = 1 : length( Ckt.INL_Vdep_type{nV} )
    switch Ckt.INL_Vdep_type{nV}{nL}
        %-----------------------------------------
        case{'poly','capNL','indNL'}% gm, capNL, indNL
            Sign    = Ckt.INL_Vdep_Sign{nV}(nL);
            gNL  = Ckt.INL_Vdep_gNL{nV}{nL};
            gNL_r   = Ckt.SYM_gNL.( gNL )(r);
            if(0)%disp('OK')
                %Ir_VEC(nV)
                %Vr_VecCEL{1,nV}{nL}
            end
            switch Ckt.INL_Vdep_type{nV}{nL}
                case{'poly'}
                    Ir_VEC(nV)      = Ir_VEC(nV)    ...
                        + Sign * gNL_r  * Vr_VecCEL{1,nV}{nL};
                case{'capNL'}
                    Ir_VEC_CAP(nV) = Ir_VEC_CAP(nV) ...
                        + Sign * gNL_r  /r * Vr_VecCEL_CAP{1,nV}{nL};
                case{'indNL'}
                    Ir_VEC_IND(nV) = Ir_VEC_IND(nV) ...
                        + Sign * gNL_r     * Vr_VecCEL_IND{1,nV}{nL};
                otherwise
                    disp('unknown Ckt.INL_Vdep_type');
```

```
            end
        %-----------------------------------------
    end
    end
    end
    INLpoly_VEC = INLpoly_VEC   + Ir_VEC;
    INLcap_VEC  = INLcap_VEC    + Ir_VEC_CAP;
    INLind_VEC  = INLind_VEC    + Ir_VEC_IND;
end%#################################### [r]     Ir_VEC --> INLpoly_VEC
    INLcap_VEC  = INLcap_VEC * (j*2*pi * riFCT(nRI) ...
        * MIX.fmixSYM_RCD{n}(nmixRCD));
return



%% ############################
function [ Vrc_MatCEL ] = Vrc_Init( const, NumMIX, INL_Vdep )

for rMIX = 1 : NumMIX
for nV = 1 : length( INL_Vdep )
for nL = 1 : size( INL_Vdep{nV}, 2 )
    Vrc_MatCEL{rMIX,nV}{nL} = sym( num2str(const) );
end
end
end
return
```

Ckt_Soln2 Function

```
function [ Ckt ] = Ckt_Soln2( Ckt, Fresp, ...
    Opt_Disp )
if ~exist('Opt_Disp','var')
    Opt_Disp = 0;
end

%% ######## Soln.( Vdep ){n}{nRESP,ri} => Soln2_[fSEPAR].( Vdep ){nRESP,1}
for nRESP = 1 : Fresp.NumResp
    Vresp_VEC  = sym( zeros([Ckt.NumV,1]) );
if(Opt_Disp) disp(['==================== [nRESP]=' num2str([nRESP])]);end
%=========================================
for nn = 1 : size( Fresp.IdxRCD_fRESP{ nRESP },1 )
    n       = Fresp.IdxRCD_fRESP{ nRESP }(nn,1);
    nmixRCD = Fresp.IdxRCD_fRESP{ nRESP }(nn,2);
    nRI     = 1;% for measured
if(Opt_Disp) disp(['----------------- [n  fmixSYM]=' num2str([n]) '...
      ' char( Fresp.fmixSYM_fRESP{ nRESP }(nn) )]);end
    %------------------------------------------------- Soln2
    for nV = 1 : Ckt.NumV
        Vdep            = Ckt.CharCEL_Vdep{ nV };
        Vresp_VEC(nV) = Vresp_VEC(nV) + Ckt.Soln.( Vdep ){n}{nmixRCD,nRI};
        Vresp_nth(nV)  =                Ckt.Soln.( Vdep ){n}{nmixRCD,nRI};
    end
end
%=========================================
    for nV = 1 : Ckt.NumV
```

```
        Vdep        = Ckt.CharCEL_Vdep{ nV };
        Ckt.Soln2.( Vdep ){nRESP,1} = Vresp_VEC(nV);
    end
end
```

Appendix B

4-Port VNA-Based NVNA: Measurement and Calibration

B.1 User Guide

The Matlab code for controlling instruments in the hybrid NVNA setup presented in Section 3.3.1 is given here. The Matlab code is sequentially presented from the user's perspective in two major parts. In the first part, calibration measurements and DUT measurements are performed. Diagrams are given to illustrate the necessary instrument connections for the measurement steps. In the second part, calibration errors are determined, and the DUT measurements are corrected. This sequence follows approximately the calibration outline given in Section 3.3.1. The complete collection of Matlab code is also available at Mathworks file exchange website http://www.mathworks.com/matlabcentral/fileexchange/31121, 4-Port VNA-based NVNA, so that user feedback can be easily incorporated in future code improvements.

Section B.2 contains the instrument control Matlab scripts for conducting calibration measurements and DUT measurements. The outline of the calibration measurement is as follows:

1. Execute the instrument connection script. Before establishing the connections, the relevant instruments in the script must be physically connected to the PC vias GPIB or

ethernet.

2. Execute the SOLT calibration measurement script to collect data for use later in determining the linear calibration error. Fig. B.1, Fig. B.2 and Fig. B.3 shows the required connections between the coaxial calibration standards and the Agilent N5230 VNA. The loads represent the off-state output impedances of the signal generators making up the two-tone source. Compared to Fig. 3.14(a), the DUT is replaced by a calibration standard, and the two-tone source is replaced by the CW source internal to the N5230 VNA.

3. Execute the power calibration measurement script to collect data for use later in determining the power calibration error $|K(f)|$. Fig. B.4 shows the required connections between the power meter and the Agilent N5230 VNA. The load represents the off-state output impedance of one of the signal generators making up the two-tone source. Compared to Fig. 3.14(a), the DUT is replaced by a power meter, and two-tone source is replaced by the CW source internal to the N5230 VNA.

4. Execute the phase calibration measurement script to collect data for use later in determining the phase calibration error $\theta_K(f)$. Fig. B.5 shows the required connections between the CW source and the Agilent N5230 VNA. The loads represent the off-state output impedances of the signal generators making up the two-tone source. Compared to Fig. 3.14(b), the CW source used is the one internal to the N5230 VNA.

5. Execute the DUT measurement script to collect data for the VNA portion of the hybrid NVNA measurement. Fig. B.6 shows the required connections to the test-set using the internal diagram of N5230 VNA, and has the same configuration as Fig. 3.14(a).

6. Execute the DUT measurement script to collect data for the oscilloscope portion of the hybrid NVNA measurement. (Due to the limited number of equivalent-time sampling oscilloscopes available in the research facility, the oscilloscope is not nose-to-nose calibrated. Thus to ensure accuracy, no measurement script was provided for this step. The user should implement his or her own measurement script when the facilities allow.)

At the beginning each script, the user needs to check certain instrument settings such as the frequency range or the power range. At the end of the each script, the collected data is saved as a data structure in a Matlab binary file.

Section B.3 contains the Matlab scripts for determining the calibration errors and correcting the DUT measurements. To process the collected data, the correction script is executed. The corrected port waves are stored in a data structure. (Due to the limited number of equivalent-time sampling oscilloscopes available in the research facility, the oscilloscope is not nose-to-nose calibrated. Thus to ensure accuracy, the correction script does not include correction using the measurements from the oscilloscope portion. The user should implement his or her own extension to the correction script when the facilities allow.) The required user inputs are the calibration kit's name corresponding to the SOLT standards used and the Matlab filenames storing the collected data. For completeness, the code for the functions called in the correction script are also presented in Section B.3. The corresponding underlying theoretical development was presented in Section 3.3.2.

B.2 Calibration Measurement and DUT Measurement

B.2.1 Open Instrument Connection

```
%% ################################################ PNA: Start Appl
%more flexible than specify IP-address 152.14.55.100
PNA_App = actxserver('AgilentPNA835X.Application', 'AGILENT-4E0C3E1');
    PNA_App.Visible = 1;% set(VSA_App,'Visible',-1);
PNA_App.Preset
    %PNA_App.RecallKits
```

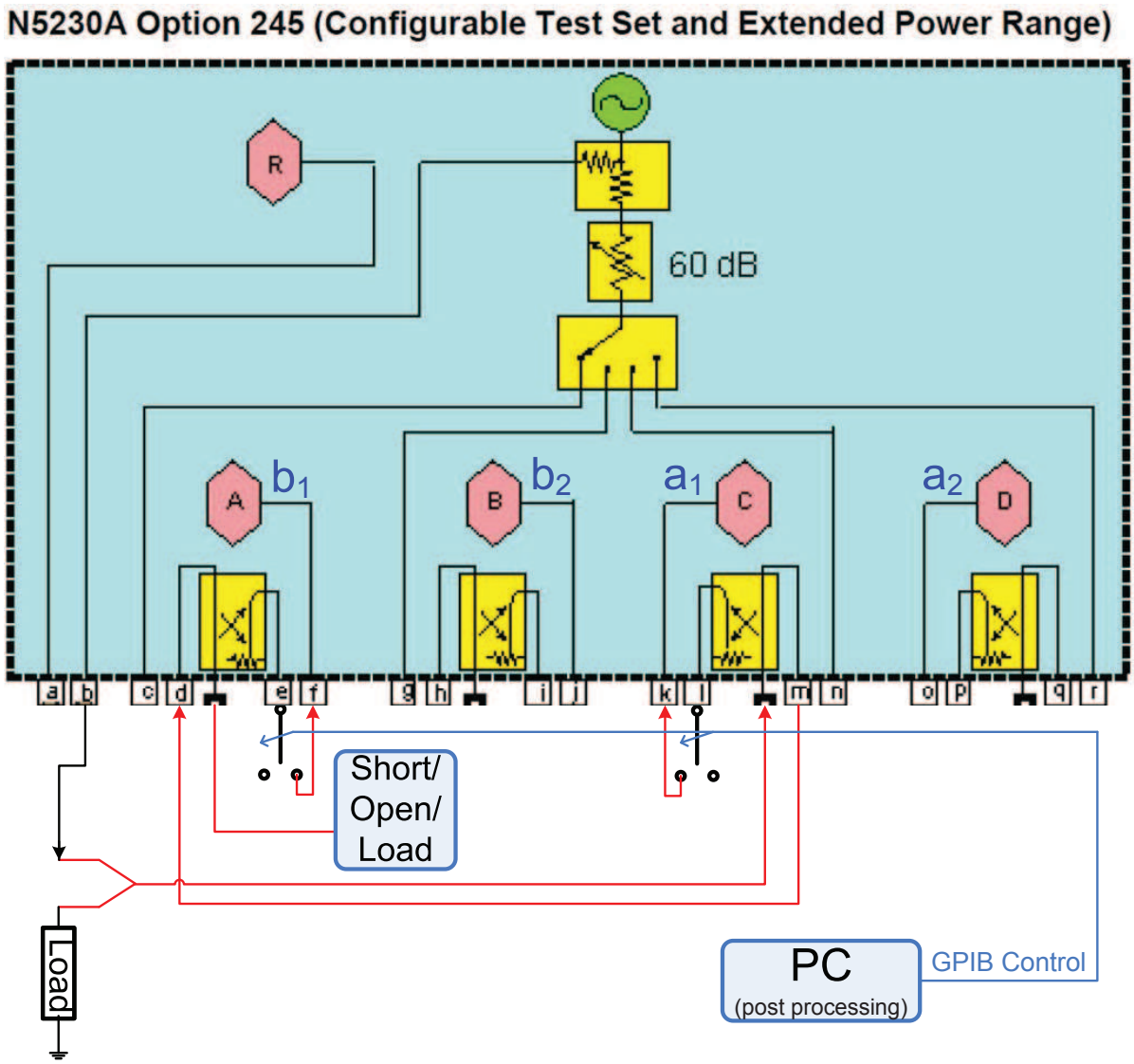


Figure B.1: Connection diagrams of linear SOLT calibration: connecting short-open-load at port-1. The loads represent the off-state output impedances of the signal generators making up the two-tone source.

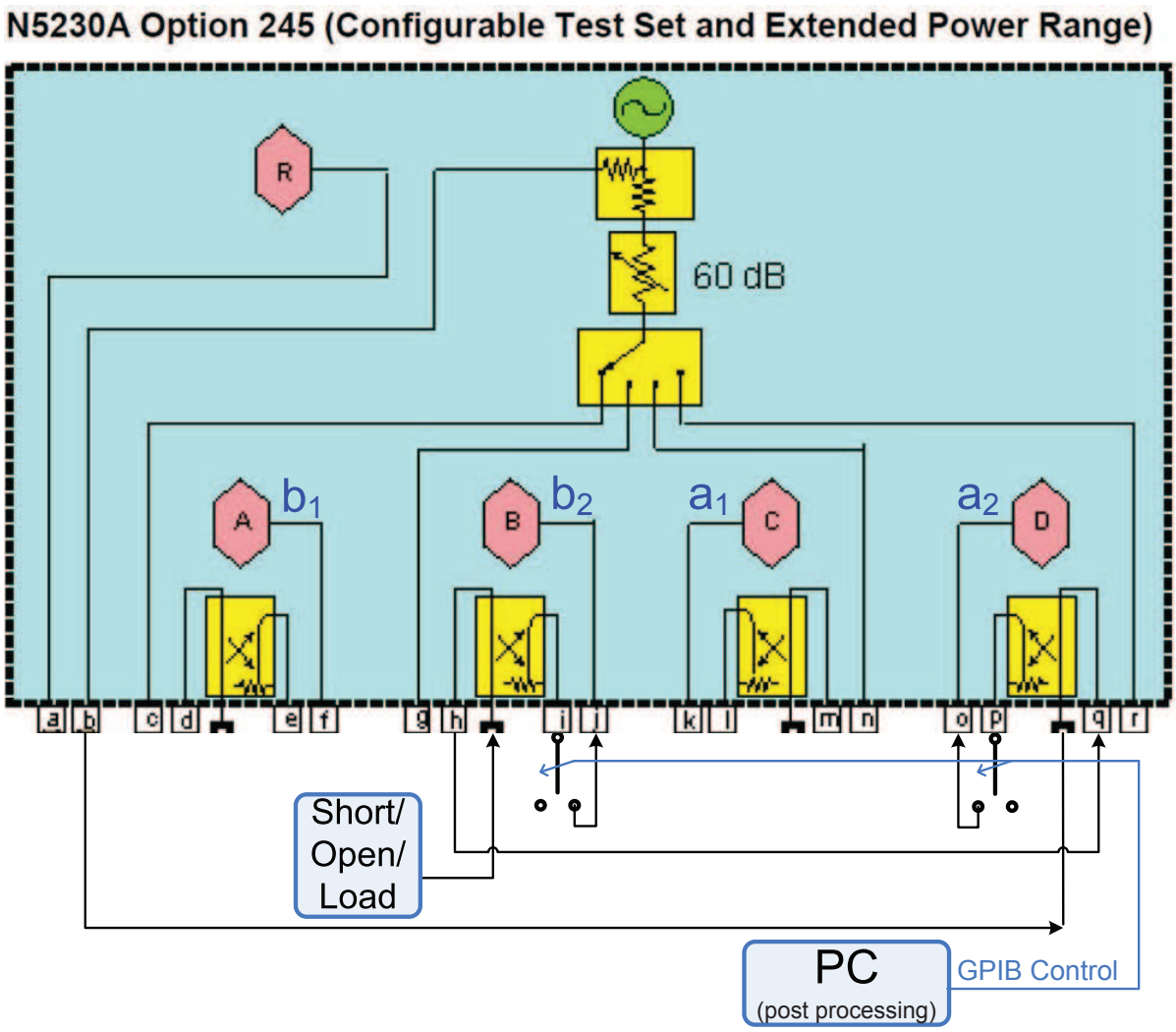


Figure B.2: Connection diagrams of linear SOLT calibration: connecting short-open-load at port-2.

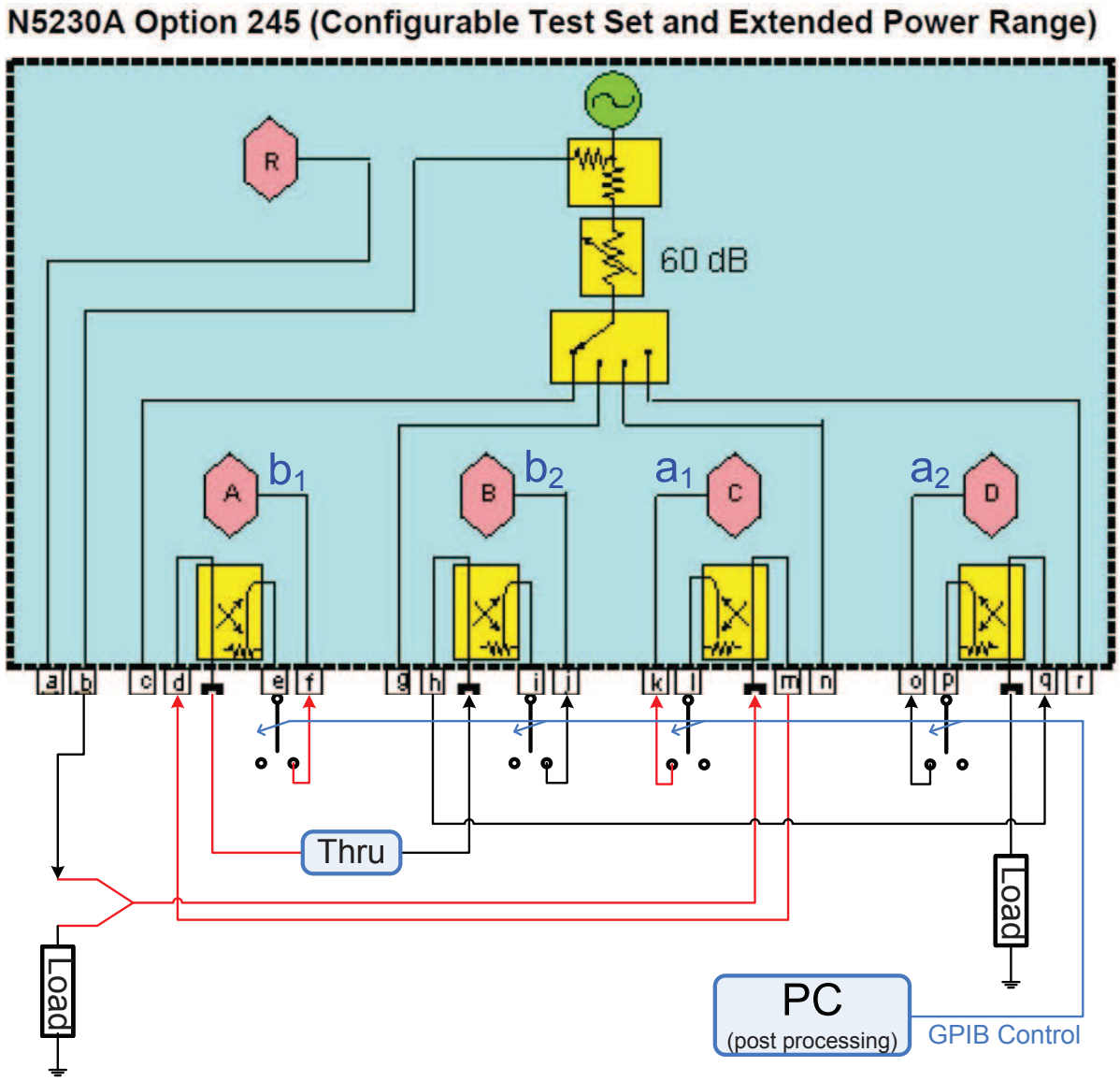


Figure B.3: Connection diagrams of linear SOLT calibration: connecting thru between port-1 and port-2. The loads represent the off-state output impedances of the signal generators making up the two-tone source.

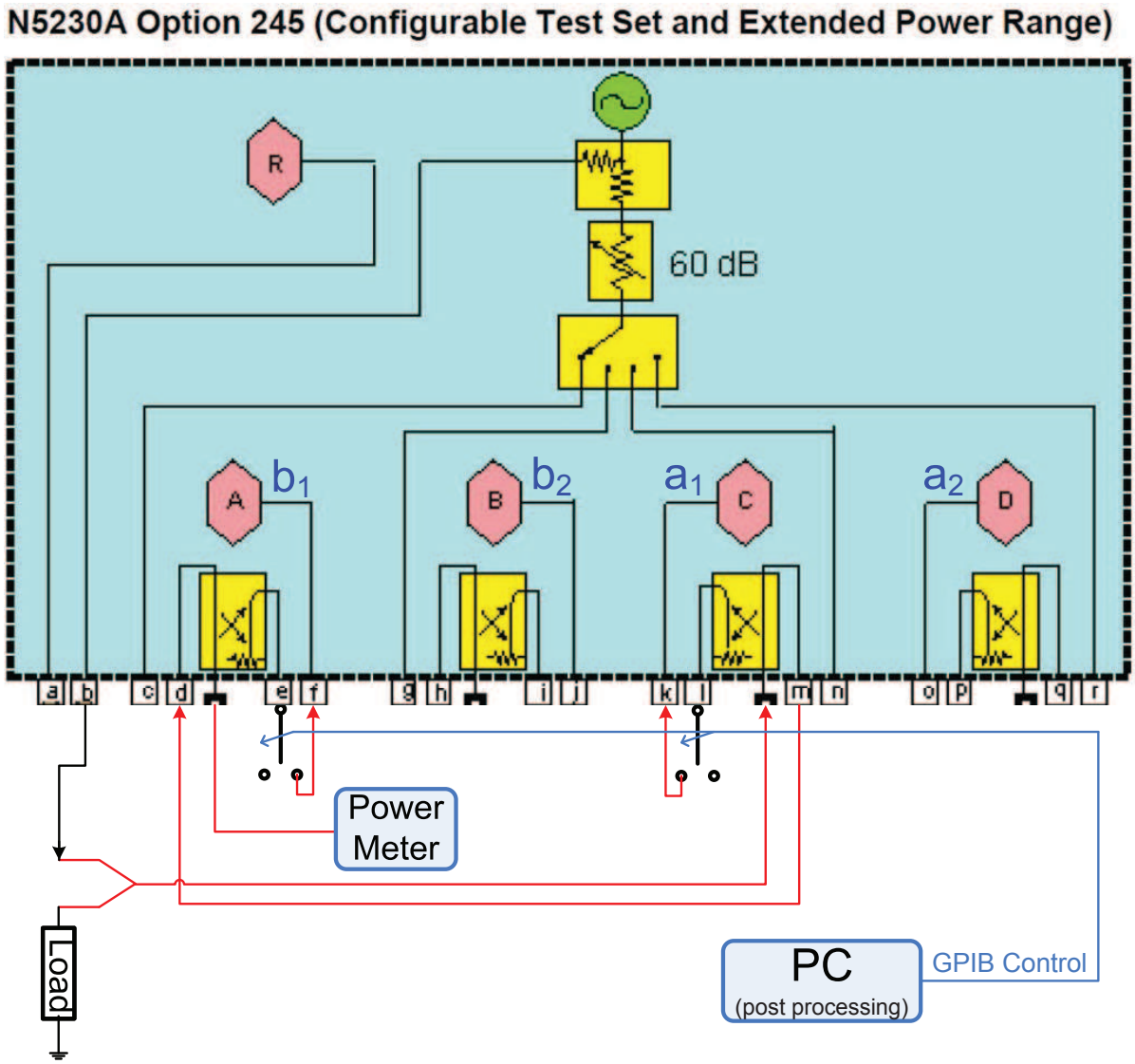


Figure B.4: Connection diagram of power calibration. The load represents the off-state output impedance of one of the signal generators making up the two-tone source.

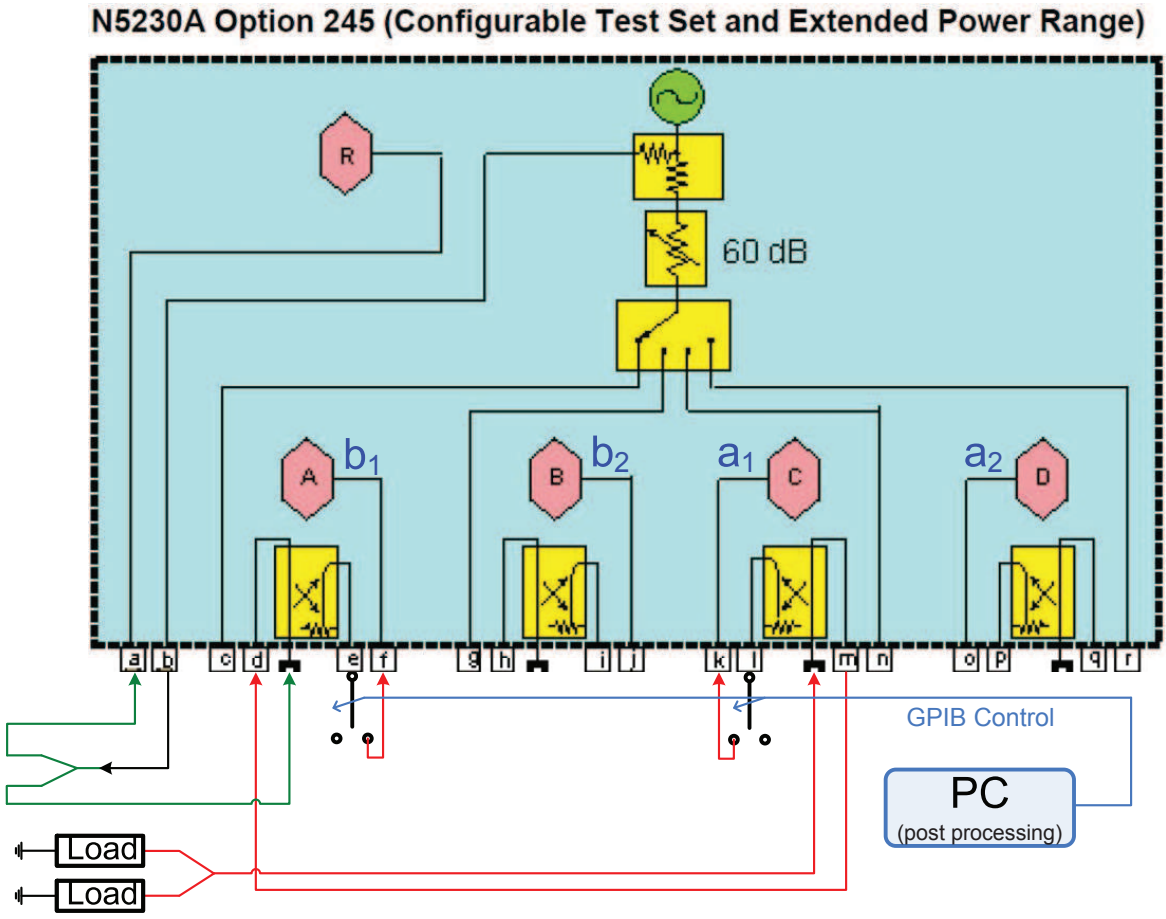


Figure B.5: Connection diagram of phase calibration. The loads represent the off-state output impedances of the signal generators making up the two-tone source.

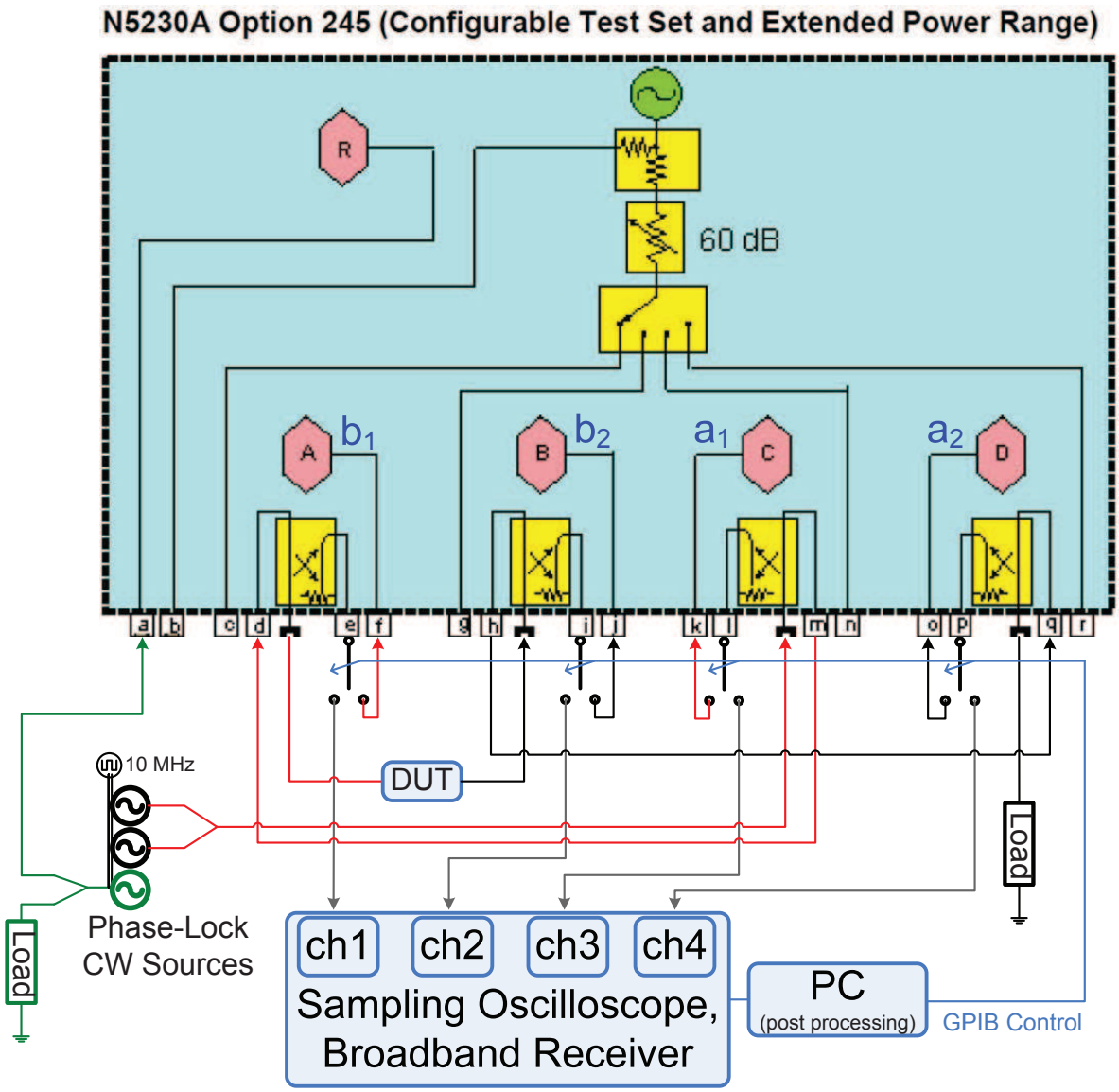


Figure B.6: Connection diagram of DUT measurement for the hybrid NVNA.

```
    %--------------------------------
    MeasCollection  = PNA_App.Measurements;
    CHcollection    = PNA_App.Channels;

%% ################################################## PwrMeter (REF)
% Meas Noise vs. Avg
% Watt2dBm( 110e-9 * [1.94 1 0.85 0.61 0.49 0.34] )
%  -36.7081  -39.5861  -40.2919  -41.7328  -42.6841  -44.2713
%  8         16        32        64        128       256
% sensor noise not much lower w/avg
% keep Signal-level > -10dBm
% use auto-mode; or filter-Len = 16,32
EPM.GPIB            = 13;   EPM.VisaTimeout = 5 * 60e3;% msec
EPM.Reset          = 0;
%EPM.RegRecall    = 9;
%-------------
EPM.Ch              = 1;
EPM.Window          = 1;
%-------------
EPM.Format          = 'ASC';% 'REAL'
EPM.Resoln          = 3;
EPM.Unit            = 'DBM';
%-------------
EPM.SensorCalTable  = '"TE_63718"';% '"TE_63718"'
EPM.AvgAuto         = 'ON';
EPM.Speed           = 20;
EPM.FreqMhz       = 1000;% check if correct CalFactor is used
```

```
%-------------
[EPM] = E4419A_Open2( EPM, EPM.Reset )%, EPM.RegRecall
E4419A_ZeroCal( EPM );% connect sensor to 1 mW reference
E4419A_Set( EPM )

%% ################################################## 2024 SigGen: f0
AF2024_f0.GPIB  = 22;
AF2024_f0.Reset  = 1;       AF2024_f0.VisaTimeout = 5 * 60e3;% msec
AF2024_f0.FSTD  = 'INT10OUT';% 1-EXT10DIR 3-EXT10IND 4-INT10OUT
AF2024_f0.RFLV.unit = 'DBM';  %AF2024_f0.RFLV.inc = 3;
AF2024_f0 = AF2024_Open( AF2024_f0, AF2024_f0.Reset);
%-------------------------------- RFLV
agt_sendcommand( AF2024_f0.IO, ['RFLV:OFF']);
    agt_query( AF2024_f0.IO, '*OPC?');
AF2024_f0.RFLV.val = -35;
AF2024_Rflv( AF2024_f0 );
agt_sendcommand( AF2024_f0.IO, ['RFLV:ON']);
    agt_query( AF2024_f0.IO, '*OPC?');
%-------------------------------- CFRQ
AF2024_f0.CFRQ.val = 1e6 * 400;
AF2024_Cfrq( AF2024_f0 );

%% ################################################## 2024 SigGen: f1
AF2024_f1.GPIB  = 21;
AF2024_f1.Reset  = 1;     AF2024_f1.VisaTimeout = 5 * 60e3;% msec
AF2024_f1.FSTD  = 'EXT10DIR';% 1-EXT10DIR 3-EXT10IND 4-INT10OUT
AF2024_f1.RFLV.unit = 'DBM';  %AF2024_f1.RFLV.inc = 3;
```

```
AF2024_f1 = AF2024_Open( AF2024_f1, AF2024_f1.Reset);
%-------------------------------- RFLV
agt_sendcommand( AF2024_f1.IO, ['RFLV:OFF']);
    agt_query( AF2024_f1.IO, '*OPC?');
AF2024_f1.RFLV.val = -40;
AF2024_Rflv( AF2024_f1 );
%-------------------------------- CFRQ
AF2024_f1.CFRQ.val = 1e6 * 500;
AF2024_Cfrq( AF2024_f1 );

%% ################################################## 2024 SigGen: f2
AF2024_f2.GPIB  = 23;
AF2024_f2.Reset  = 1;      AF2024_f2.VisaTimeout = 5 * 60e3;% msec
AF2024_f2.FSTD  = 'EXT10DIR';% 1-EXT10DIR 3-EXT10IND 4-INT10OUT
AF2024_f2.RFLV.unit = 'DBM';    %AF2024_f2.RFLV.inc = 3;
AF2024_f2 = AF2024_Open( AF2024_f2, AF2024_f2.Reset);
%-------------------------------- RFLV
agt_sendcommand( AF2024_f2.IO, ['RFLV:OFF']);
    agt_query( AF2024_f2.IO, '*OPC?');
AF2024_f2.RFLV.val = -40;
AF2024_Rflv( AF2024_f2 );
%-------------------------------- CFRQ
AF2024_f2.CFRQ.val = 1e6 * 900;
AF2024_Cfrq( AF2024_f2 );

%% ################################################## Tek_11801a
%Tek11801a.GPIB = 16;
```

```
%Tek11801a.Reset = 1; Tek11801a.VisaTimeout = 5 * 60e3;% msec
%Tek11801a.FP_access = 'ON';     Tek11801a.FP_update = 'ON';
%Tek11801a = Tek11801a_Open( Tek11801a, Tek11801a.Reset);
```

B.2.2 Close Instrument Connection

```
%% ################################################## PNA: Quit
    CHcollection.release;
    MeasCollection.release;
    %---------------
    PNA_App.Visible = 0;
    PNA_App.release;
    % server shutdown (required before restart VNA application)
    %PNA_App.Quit;% brute force way of terminating the application

%% ##################################################
%close all sessions. (Not useful for GPIB, which has no limit)
agt_closeAllSessions;
```

B.2.3 SOLT Calibration Measurement

```
%% S O L T @(Pin, Spectral-FreqSwp)###################### PNA=SigGen PNA=Rx
Meas.SigGen ='PNA';
Meas.DataRun ={'Sp1' 'Op1' 'Lp1' 'Tp12' 'Sp2' 'Op2' 'Lp2'};% Fwd+Rev 'Tp21'
%-------------------------------- 1Tone
Meas.SigGen0_dBm  = [ -20 ];
Meas.SigGen0_Mhz  = [ 1 : 2 : 8001 ];
%-------------------------------- Ref = C (User Testset)
Meas.CH     = [ 1 ];% same CH measured for each of Meas.DataRun
Meas.Param = {{'A' 'B' 'C' 'D'}};
Meas.Iport = {[ 1   1   1   1 ]};
Meas.Window = {[ 1   2   3   4 ]};
Meas.Wav_Name = struct( ...
                        'S11_S_m1',{'C_Sp1'}, ...
                        'S11_S_m2',{'A_Sp1'}, ...%----
                        'S11_O_m1',{'C_Op1'}, ...
                        'S11_O_m2',{'A_Op1'}, ...%----
                        'S11_L_m1',{'C_Lp1'}, ...
                        'S11_L_m2',{'A_Lp1'}, ...%----
                        'S12_T_m1',{'C_Tp12'}, ...
                        'S12_T_m2',{'A_Tp12'}, ...
                        'S12_T_m3',{'D_Tp12'}, ...
                        'S12_T_m4',{'B_Tp12'}, ...%----
                        'S22_S_m1',{'D_Sp2'}, ...
                        'S22_S_m2',{'B_Sp2'}, ...%----
                        'S22_O_m1',{'D_Op2'}, ...
                        'S22_O_m2',{'B_Op2'}, ...%----
```

```
                        'S22_L_m1',{'D_Lp2'}, ...
                        'S22_L_m2',{'B_Lp2'} );  %----

%% ################################################
%--------------------------------- PNA settings
Meas.TrigSrc        = 1;
Meas.TrigScope      = 1;
%--------------------------------- CH settings
Meas.TrigMod = repmat( 1,         [1 length(Meas.CH)] );
%---------------------------------
%Meas.AttP1  = repmat( 60,          [1 length(Meas.CH)] );
%Meas.AttP2  = repmat( 60,          [1 length(Meas.CH)] );
%Meas.AttP3  = repmat( 60,         [1 length(Meas.CH)] );
%Meas.AttP4  = repmat( 60,         [1 length(Meas.CH)] );
Meas.AttA   = repmat( 0,          [1 length(Meas.CH)] );
Meas.AttB   = repmat( 0,           [1 length(Meas.CH)] );
Meas.AttC   = repmat( 0,           [1 length(Meas.CH)] );
Meas.AttD   = repmat( 0,            [1 length(Meas.CH)] );
%---------------------------------
Meas.SwpTyp = repmat( 0,                            [1 length(Meas.CH)] );
Meas.fStart = repmat( 1e6 * Meas.SigGen0_Mhz( 1 ),[1 length(Meas.CH)] );
Meas.fStop = repmat( 1e6 * Meas.SigGen0_Mhz( end ),[1 length(Meas.CH)] );
Meas.NumPts = repmat( length( Meas.SigGen0_Mhz ),  [1 length(Meas.CH)] );
%---------------------------------
Meas.AltSwp = repmat( 0,           [1 length(Meas.CH)] );
Meas.SwpGen = repmat( 0,            [1 length(Meas.CH)] );
Meas.DwellT = repmat( 0.05,         [1 length(Meas.CH)] );
```

```
Meas.IfbwHz = repmat( 100,        [1 length(Meas.CH)] );

PNA_CHSet

%% ##################################################
%================================================================
% To Restart CH-Trig
% In code: toggle CH.TrigMode
        for nCH = 1 : length( Meas.CH )
            CHcollection.Item( nCH ).TriggerMode = 0;
            CHcollection.Item( nCH ).TriggerMode = Meas.TrigMod( nCH );
        end
%================================================================
% Pt-Trig dont like setting TestPortPower
% Work around 1: set TestPortPower once
% Work around 2: meas nCH=1 twice
nP = 1;
for nCH = 1 : length( Meas.CH )
  % CHcollection.Item( nCH ).TestPortPower(1)% =get
  CHcollection.Item( nCH ).set('TestPortPower',1,Meas.SigGen0_dBm( nP ));
  CHcollection.Item( nCH ).set('TestPortPower',2,Meas.SigGen0_dBm( nP ));
  CHcollection.Item( nCH ).set('TestPortPower',3,Meas.SigGen0_dBm( nP ));
  CHcollection.Item( nCH ).set('TestPortPower',4,Meas.SigGen0_dBm( nP ));
end

%% ##################################################
Meas.PNA_mWatt_RI = [];
```

```
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ nRun
for nRun = 1 : length( Meas.DataRun )
    disp(['Change connection for ' Meas.DataRun{nRun} ' SOLT']);
pause
%========================================================== nCH
for nCH = 1 : length( Meas.CH )
%CHnum = Meas.CH( nCH );
%CHnum = CHcollection.Item( nCH ).ChannelNumber
    MeasCollection.Item( Meas.Param{nCH}{1} ).Activate;
%========================================================== @(Pin)
    disp(['%======================================== @(Pin)']);
for nP = 1 : length( Meas.SigGen0_dBm )
    sprintf('@(Pin): %f dBm', Meas.SigGen0_dBm( nP ))% DEBUG
switch Meas.SigGen
    %-------------------------------- PNA (RX)
    case{'PNA'}
       PNA_App.ManualTrigger( 1,-1 );
disp('see if PNA finish sweep: if sweep, then need pause sweep-time')
pause(0.05+Meas.DwellT(1))% max([0.5 3*CHcollection.Item( nCH ).SweepTime])
    %--------------------------------
end
%========================================================== @(Spectral-FreqSwp)
%========================================================== @(Spectral-FreqSwp)
end
%========================================================== @(Pin)
end
%========================================================== nCH
```

```
    Opt = struct('Format',{3}, ...
        'VariantTyped',{'variant'}, 'Disp',{1}, 'XAxis',{0});
    for nMeas = 1 : length( Meas.Param{ nCH } )
        MEAS            = MeasCollection.Item( Meas.Param{nCH}{nMeas} );
        MEAS_TempNam    = [ Meas.Param{nCH}{nMeas} '_' Meas.DataRun{nRun} ]
      Meas.PNA_mWatt_RI.(MEAS_TempNam)(nP,:) = PNA_DataMeas( MEAS, Opt );
        MEAS.release;
    end
end
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ nRun

%% ################################################## SAVE
%---------------- SOLT Freq
FileNam = ['PNA_SOLT-freq_' cell2mat( Meas.Param{1} ) '_' ...
    num2str(Meas.SigGen0_Mhz(1)) '-' ...
    num2str(Meas.SigGen0_Mhz(2)-Meas.SigGen0_Mhz(1)) '-' ...
    num2str(Meas.SigGen0_Mhz(end)) 'Mhz_' ...
    num2str(Meas.SigGen0_dBm) 'dBm_IFBW' num2str(Meas.IfbwHz(1)) 'Hz.mat']
%----------------
if exist( FileNam,'file')
    disp([ FileNam ' exists']);
    pause
else
    save( FileNam, 'Meas', 'Opt' )
end
```

B.2.4 Power Calibration Measurement

```
%% ############################## PNA=SigGen PNA=Rx @(Pin, Spectral-FreqSwp)
% SigGen    9khz    - 2400Mhz
% Splitter  500Mhz  - 5000Mhz
% PNA       0.3Mhz  - 20,000Mhz
Meas.SigGen         = 'PNA';
Meas.SigGen0_dBm  = [ -20 ];

%--------------------------------- Ref = C (User Testset)
Meas.SigGen0_Mhz   = [ 1 : 2 : 8001 ];
Meas.CH     = [ 1 ];
Meas.Param = {{'C' 'A'}};
Meas.Iport = {[ 1   1 ]};
Meas.Window = {[ 1   2 ]};
Meas.Wav_Name = struct( 'EPM_m1',{'CA1'}, ...
                        'P1_m1',{'C1'}, ...
                        'P1_m2',{'A1'} );
% R A B C D: separate setup, meas in separate CH
% CH-Trig
% Freq sweep: trace Pt-Trig

%--------------------------------- PNA settings
Meas.TrigSrc        = 1;% 0-Internal   1-Manual   2-External
Meas.TrigScope      = 1;% 0-Global-all-CH  1-current CH
%--------------------------------- CH settings
% 0-1pt in ch  1-all pt/meas in ch
Meas.TrigMod = repmat( 0,         [1 length(Meas.CH)] );
```

```
%--------------------------------
% 0 - 60
%Meas.AttP1  = repmat( 60,          [1 length(Meas.CH)] );
%Meas.AttP2  = repmat( 60,          [1 length(Meas.CH)] );
%Meas.AttP3  = repmat( 60,         [1 length(Meas.CH)] );
%Meas.AttP4  = repmat( 60,         [1 length(Meas.CH)] );
% 0 - 35
Meas.AttA   = repmat( 0,          [1 length(Meas.CH)] );
Meas.AttB   = repmat( 0,           [1 length(Meas.CH)] );
Meas.AttC   = repmat( 0,           [1 length(Meas.CH)] );
Meas.AttD   = repmat( 0,            [1 length(Meas.CH)] );
%--------------------------------

Meas.SwpTyp = repmat( 0,                              [1 length(Meas.CH)] );
Meas.fStart = repmat( 1e6 * Meas.SigGen0_Mhz( 1 ),[1 length(Meas.CH)] );
Meas.fStop = repmat( 1e6 * Meas.SigGen0_Mhz( end ),[1 length(Meas.CH)] );
Meas.NumPts = repmat( length( Meas.SigGen0_Mhz ),  [1 length(Meas.CH)] );
%--------------------------------
% 0 - Chopped 1 - Alternate   reflect/transmission
Meas.AltSwp = repmat( 0,           [1 length(Meas.CH)] );
% 0 - naSteppedSweep 1 - naAnalogSweep
Meas.SwpGen = repmat( 0,            [1 length(Meas.CH)] );
Meas.DwellT = repmat( 0.05,         [1 length(Meas.CH)] );
Meas.IfbwHz = repmat( 100,        [1 length(Meas.CH)] );

PNA_CHSet
```

```
%% ################################################
%================================================================
% To Restart Pt-Trig from GUI
% GUI:     toggle CH1 PointSweep box
% In code: toggle Trig.Scope to disable Pt-Trig,
%          then enable CH.TrigMode=Pt
% In code: toggle CH.TrigMode
% Note: code dont update Trig GUI
switch 'Meth_2'
    %---------------------------------
    case{'Meth_1'}
        % 0-Global-all-CH     1-current CH
        PNA_App.TriggerSetup.Scope      = 0;
        PNA_App.TriggerSetup.Scope      = Meas.TrigScope;
        for nCH = 1 : length( Meas.CH )
            % 0-1pt in ch  1-all meas in ch
            CHcollection.Item( nCH ).TriggerMode    = Meas.TrigMod( nCH );
        end
    %---------------------------------
    case{'Meth_2'}
        for nCH = 1 : length( Meas.CH )
            CHcollection.Item( nCH ).TriggerMode = 1;
            CHcollection.Item( nCH ).TriggerMode = Meas.TrigMod( nCH );
        end
    %---------------------------------
end
    %CHcollection.Item( nCH ).Abort( 1 );
```

```
    %CHcollection.Item( nCH ).Single(1);% for Intern + Extern trig-source
%=================================================================
% Pt-Trig dont like setting TestPortPower
% Work around 1: set TestPortPower once
% Work around 2: meas nCH=1 twice
nP = 1;
for nCH = 1 : length( Meas.CH )
    % CHcollection.Item( nCH ).TestPortPower(1)% =get
    CHcollection.Item( nCH ).set('TestPortPower',1,Meas.SigGen0_dBm( nP ));
    CHcollection.Item( nCH ).set('TestPortPower',2,Meas.SigGen0_dBm( nP ));
    CHcollection.Item( nCH ).set('TestPortPower',3,Meas.SigGen0_dBm( nP ));
    CHcollection.Item( nCH ).set('TestPortPower',4,Meas.SigGen0_dBm( nP ));
end

%% ###################################### RxPwrCal @(Pin, Spectral-FreqSwp)
Meas.EPM_dBm        = [];
Meas.PNA_mWatt_RI   = [];
%=================================================== nCH
for nCH = 1 : length( Meas.CH )
%CHnum = Meas.CH( nCH );
%CHnum = CHcollection.Item( nCH ).ChannelNumber
    % [1.meas.name -> meas.ch# -> activate ch]
    % [2.record item#in MeasCollection]
    MeasCollection.Item( Meas.Param{nCH}{1} ).Activate;
    [ num2str( PNA_App.ActiveChannel.ChannelNumber ) ...
        PNA_App.ActiveMeasurement.Name ...
        num2str( PNA_App.ActiveNAWindow.WindowNumber ) ]
```

```
    disp(['Change connection for ' Meas.Param{nCH}{1} ' RxPwrCal']);
pause
%====================================================== @(Pin)
    disp(['%==================================== @(Pin)']);
for nP = 1 : length( Meas.SigGen0_dBm )
    sprintf('@(Pin): %f dBm', Meas.SigGen0_dBm( nP ))% DEBUG
%====================================================== @(Spectral-FreqSwp)
    disp(['%================== @(Spectral-FreqSwp)']);
for nF = 1 : length( Meas.SigGen0_Mhz )
    switch Meas.SigGen
        %-------------------------------- PNA (RX)
        case{'PNA'}
            % (1-Waits till trigger complete, [-1]-Wait indefinitely)
            PNA_App.ManualTrigger( 1,-1 );
% DwellT < this pause, so following Non-PNA-Rx don't read the wrong val
pause( 2 * Meas.DwellT(1))
    % After trig, PNA source to next pt
    % imprecise to set DwellT long enough to cover the following Non-PNA-Rx
    % so Non-PNA-Rx always read the next-pt, 1st-pt is read last
if(nF == length(Meas.SigGen0_Mhz))
    idx = 1;% wrap-around after Trigger @ last freq-pt
else
    idx = nF+1;% EPM actually sensing the next freq-pt
end
f_MHz = Meas.SigGen0_Mhz( idx );
        %--------------------------------
    end
```

```
    [Meas.SigGen0_Mhz( nF )     f_MHz]
    %-------------------------------- PwrMeter (REF)
EPM.FreqMhz     = f_MHz;
EPM_TempNam     = [ [Meas.Param{nCH}{:}] int2str(Meas.CH( nCH )) ];
  Meas.EPM_dBm.(EPM_TempNam)(nP,idx) = E4419A_SensFreq_Meas( EPM );
    %-------------------------------- PSA (RX)
sprintf('%s   %d   %f', EPM_TempNam, ...
        EPM.FreqMhz, Meas.EPM_dBm.(EPM_TempNam)(nP,idx))
end
%====================================================== @(Spectral-FreqSwp)
    Opt = struct('Format',{3}, ...
        'VariantTyped',{'variant'}, 'Disp',{1}, 'XAxis',{0});
    for nMeas = 1 : length( Meas.Param{ nCH } )
        MEAS        = MeasCollection.Item( Meas.Param{nCH}{nMeas} );
        MEAS_TempNam= [ Meas.Param{nCH}{nMeas}  int2str(Meas.CH( nCH )) ]
      Meas.PNA_mWatt_RI.(MEAS_TempNam)(nP,:) = PNA_DataMeas( MEAS, Opt );
        MEAS.release;
        %real( Lin2dB_dB(Meas.PNA_mWatt_RI.(MEAS_TempNam)(nP,:)) )%DEBUG
    end
end
%====================================================== @(Pin)
end
%====================================================== nCH

%% ################################################## SAVE
FileNam = ['PNA_RxPwrCal-freq_' Cel2Mat( Meas.Param ) '_' ...
    num2str(Meas.SigGen0_Mhz(1)) '-' ...
```

```
    num2str(Meas.SigGen0_Mhz(2)-Meas.SigGen0_Mhz(1)) '-' ...
    num2str(Meas.SigGen0_Mhz(end)) 'Mhz_' ...
    num2str(Meas.SigGen0_dBm) 'dBm_IFBW' num2str(Meas.IfbwHz(1)) 'Hz.mat']
%----------------
if exist( FileNam,'file')
    disp([ FileNam ' exists']);
    pause
else
    save( FileNam, 'Meas','Opt' )
end
```

B.2.5 DUT Measurement

```
%% ################################################ Meas = PNA settings
Meas.SigGen         = 'AF2024';% PNA';%
Meas.SigGen0_dBm   = [ -35 ];% Rsrc(rx=-10dBm, meter=-35dBm)
Meas.AmpUnit        = 'dBm';
Meas.FreqUnit       = 'MHz';

Meas.fLimLO_Hz      = 1e6;
Meas.fLimHI_Hz      = 20e9;
%% -------------------------------- DUT (User Testset)
Meas.CH     = [ 1 ];
Meas.Param = {{'R' 'A' 'B' 'C' 'D'}};
Meas.Iport = {[ 1   1   1   1   1 ]};
Meas.Window = {[ 1   2   3   4   5 ]};
% R A B C D: same setup, meas in same CH
% Pwr sweep: trace Pt-Trig
Meas.Wav_Name = struct( 'R',{'R'}, ...
                          'P1_m1',{'C'}, ...
                         'P1_m2',{'A'}, ...
                         'P2_m3',{'D'}, ...
                          'P2_m4',{'B'} );

%%################################################
Meas.Qtone          = 2;
%-------------------------------- 2Tone
Meas.DUT_name       = 'SD513301';
Meas.Att_dB         = 20;
```

```
Meas.SigGen_dBm     = [ -30 : 1 : 10 ];
    Tek11801a.SigGen_dBm = [ -20 ];
Meas.Qtone_swp      = '1Carr_DfNarrow';
Meas.Qtone_Mhz      = [ 2000 ];
Meas.Qtone_swpMAX   = [ 60 ];
Meas.nHD            = 5;
%-----------------------------------DF_VEC
Meas.DFmax          = [1e6  1.25e6  2e6  2.5e6  5e6];
[ Tek11801a     Meas ] = Oscope_TimeDiv( Tek11801a, Meas, 'DF_sweep');
    Meas.DF_VEC
    Meas.DF_VEC .* Tek11801a.m_VEC
    % [0 1250000 2000000 2500000 5000000]
    % 1.0e+010 * [0    0.3125    0.5000    0.6250    1.2500]
%-----------------------------------DFMhz: pick from DF_VEC
Meas.Qtone_DFMhz = 2;
%-----------------------------------SigGen1_Mhz,SigGen2_Mhz:
[ Tek11801a     Meas ] = Oscope_QtoneGen( Tek11801a, Meas );
%-----------------------------------SigGen0_Mhz
[ MIX ] = Mix_Qtone_Sym( Meas.nHD, 2, ...
    struct('HD',{'wHD'}, 'fListOrd',{'ascend'}) );
[Meas.SigGen0_Mhz Meas.SigGen0_Mhz_IDXcel ...
 Meas.fnumMAT_ALL Meas.fnumMAT_ALLIDX     Meas.fnumMAT_f12IDX] =
    Mix_Qtone_Num2( MIX, ...
        {(Meas.SigGen1_Mhz).'*1e6 (Meas.SigGen2_Mhz).'*1e6}, ...
        struct('fLimLO_Hz',{Meas.fLimLO_Hz},'fLimHI_Hz',{Meas.fLimHI_Hz}));

%% ##################################################
```

```
%--------------------------------- PNA settings
Meas.TrigSrc        = 1;% 0-Internal   1-Manual   2-External
Meas.TrigScope      = 1;% 0-Global-all-CH   1-current CH
%---------------------------------
Meas.TrigMod = repmat( 0,         [1 length(Meas.CH)] );
%---------------------------------
%Meas.AttP1  = repmat( 60,          [1 length(Meas.CH)] );
%Meas.AttP2  = repmat( 60,          [1 length(Meas.CH)] );
%Meas.AttP3  = repmat( 60,         [1 length(Meas.CH)] );
%Meas.AttP4  = repmat( 60,         [1 length(Meas.CH)] );
Meas.AttA   = repmat( 0,          [1 length(Meas.CH)] );
Meas.AttB   = repmat( 0,           [1 length(Meas.CH)] );
Meas.AttC   = repmat( 0,           [1 length(Meas.CH)] );
Meas.AttD   = repmat( 0,            [1 length(Meas.CH)] );
%---------------------------------
Meas.SwpTyp = repmat( 2,                        [1 length(Meas.CH)] );
Meas.pStart = repmat( Meas.SigGen_dBm( 1 ),    [1 length(Meas.CH)] );
Meas.pStop = repmat( Meas.SigGen_dBm( end ),  [1 length(Meas.CH)] );
Meas.NumPts = repmat( length( Meas.SigGen_dBm ),[1 length(Meas.CH)] );
%---------------------------------
Meas.AltSwp = repmat( 0,           [1 length(Meas.CH)] );
Meas.SwpGen = repmat( 0,            [1 length(Meas.CH)] );
Meas.DwellT = repmat( 0.05,         [1 length(Meas.CH)] );
Meas.IfbwHz = repmat( 1,            [1 length(Meas.CH)] );


PNA_CHSet
```

```
%% ################################
% To Restart CH-Trig
% In code: toggle CH.TrigMode
        for nCH = 1 : length( Meas.CH )
            CHcollection.Item( nCH ).TriggerMode = 1;
            CHcollection.Item( nCH ).TriggerMode = Meas.TrigMod( nCH );
        end

%% ################################ DUT 1-Tone@(Spectral-FreqSwp, PinSwp)
Meas.PNA_mWatt_RI = [];
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Input MTone
for nDF = 1 : length( Meas.SigGen1_Mhz )
        AF2024_f1.CFRQ.val = 1e6 * Meas.SigGen1_Mhz( nDF );
        AF2024_Cfrq( AF2024_f1 );
    if ~isempty( Meas.SigGen2_Mhz )
        AF2024_f2.CFRQ.val = 1e6 * Meas.SigGen2_Mhz( nDF );
        AF2024_Cfrq( AF2024_f2 );
    end
%============================================== Ouput @(Spectral-FreqSwp)
for nF = 1 : length( Meas.SigGen0_Mhz{nDF} )
    AF2024_f0.CFRQ.val = 1e6 * Meas.SigGen0_Mhz{nDF}(nF);
    AF2024_Cfrq( AF2024_f0 );
    sprintf('@(SpectralTone): %f Mhz', Meas.SigGen0_Mhz{nDF}(nF))% DEBUG
%============================================== nCH
for nCH = 1 : length( Meas.CH )
    CHcollection.Item( nCH ).CWFrequency = 1e6 * Meas.SigGen0_Mhz{nDF}(nF);
    MeasCollection.Item( Meas.Param{nCH}{1} ).Activate;
```

```
    [ num2str( PNA_App.ActiveChannel.ChannelNumber )  ...
        PNA_App.ActiveMeasurement.Name num2str( ...
            PNA_App.ActiveNAWindow.WindowNumber ) ]
%============================================== @(Pin)
    disp(['%=================================== @(Pin)']);
for nP = 1 : length( Meas.SigGen_dBm )
        AF2024_f1.RFLV.val  = Meas.SigGen_dBm( nP );
        AF2024_Rflv( AF2024_f1 );
        agt_sendcommand( AF2024_f1.IO, ['RFLV:ON']);
            agt_query( AF2024_f1.IO, '*OPC?');
    if ~isempty( Meas.SigGen2_Mhz )
        AF2024_f2.RFLV.val  = Meas.SigGen_dBm( nP );
        AF2024_Rflv( AF2024_f2 );
        agt_sendcommand( AF2024_f2.IO, ['RFLV:ON']);
            agt_query( AF2024_f2.IO, '*OPC?');
    end
sprintf('@(Pin): %f dBm', Meas.SigGen_dBm( nP ))% DEBUG
pause( 0.5 );% check SigGen
    PNA_App.ManualTrigger( 1,-1);
    %seem to finish data-acquisition before going to next command
disp('see if PNA finish sweep: if sweep, then need pause sweep-time')
pause( 2*Meas.DwellT(1))%max([0.5 3*CHcollection.Item( nCH ).SweepTime])
        agt_sendcommand( AF2024_f1.IO, ['RFLV:OFF']);
            agt_query( AF2024_f1.IO, '*OPC?');
if ~isempty( Meas.SigGen2_Mhz )
        agt_sendcommand( AF2024_f2.IO, ['RFLV:OFF']);
            agt_query( AF2024_f2.IO, '*OPC?');
```

```
    end
pause( 0.1 );% DUT cool off
end
%============================================== @(Pin)
end
%============================================== nCH
    Meas.Opt = struct('Format',{3}, 'VariantTyped',{'variant'}, ...
        'Disp',{1}, 'XAxis',{0});
    for nMeas = 1 : length( Meas.Param{ nCH } )
        MEAS            = MeasCollection.Item( Meas.Param{nCH}{nMeas} );
        MEAS_TempNam   = [ Meas.Param{nCH}{nMeas} ]
      Meas.PNA_mWatt_RI.(MEAS_TempNam){nDF}(nF,:) = ...
            PNA_DataMeas( MEAS, Meas.Opt);
        MEAS.release;
    end
    save(FileNam, 'Meas')% prevent data lose if matlab crash
end
%============================================== Ouput @(Spectral-FreqSwp)
end
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ nTone

%% ################################################## SAVE
if isempty( Meas.SigGen2_Mhz )% 1-Tone
    if (length(Meas.SigGen1_Mhz) == 1)
    FileNam = ['PNA_' Meas.DUT_name '_1Tone-Ord' num2str(Meas.nHD) '_' ...
    num2str(Meas.SigGen1_Mhz(1)) 'Mhz_' ...
    num2str(Meas.SigGen_dBm(1))  '-' ...
```

```
    num2str(Meas.SigGen_dBm(end)) 'dBm' num2str(Meas.SigGen_dBm(2) - ...
    Meas.SigGen_dBm(1)) 'dB_' ...
    'Lim' num2str(Meas.fLimLO_Hz/1e6)  '-' ....
num2str(Meas.fLimHI_Hz/1e6) 'Mhz_' 'IFBW' num2str(Meas.IfbwHz(1)) 'Hz.mat']
    else%----------------
    FileNam = ['PNA_' Meas.DUT_name '_1Tone-Ord' num2str(Meas.nHD) '_' ...
    num2str(Meas.SigGen1_Mhz(1)) '-' num2str(Meas.SigGen1_Mhz(end)) '-' ...
    num2str(Meas.SigGen1_Mhz(2) - Meas.SigGen1_Mhz(1)) 'Mhz_' ...
    num2str(Meas.SigGen_dBm(1)) '-' num2str(Meas.SigGen_dBm(end)) ...
    'dBm' num2str(Meas.SigGen_dBm(2) - Meas.SigGen_dBm(1)) 'dB_' ...
    'Lim' num2str(Meas.fLimLO_Hz/1e6)  '-' ....
num2str(Meas.fLimHI_Hz/1e6) 'Mhz_' 'IFBW' num2str(Meas.IfbwHz(1)) 'Hz.mat']
    end
else%----------------
    FileNam = ['PNA_' Meas.DUT_name '_2Tone-Ord' num2str(Meas.nHD) '_' ...
    num2str(Meas.SigGen2_Mhz(1)-Meas.SigGen1_Mhz(1)) '-' ...
    num2str(Meas.SigGen2_Mhz(end)-Meas.SigGen1_Mhz(end)) 'Mhz' ...
    num2str(length(Meas.SigGen1_Mhz)) 'pt_' ...
    num2str(Meas.SigGen_dBm(1)) '-' num2str(Meas.SigGen_dBm(end)) 'dBm' ...
    num2str(Meas.SigGen_dBm(2) - Meas.SigGen_dBm(1)) 'dB_' ...
    'Lim' num2str(Meas.fLimLO_Hz/1e6)  '-' ....
num2str(Meas.fLimHI_Hz/1e6) 'Mhz_' 'IFBW' num2str(Meas.IfbwHz(1)) 'Hz.mat']
end
%----------------
if exist( FileNam,'file')
    disp([ FileNam ' exists']);
    pause
```

```
else
    save( FileNam, 'Meas')
end
```

B.3 DUT Measurement Correction

B.3.1 Correction Script

```
%clear all
PC  = 'DODO';% 'DODO' 'Grendel' 'Gromit' 'Coolum'
[PATH, PlotOpt] = Matlab_Settings(PC)


%% ################################################## LSNAncsu NL-meas
DUT     = '_CAL';
Char    = 'LSNA_ncsu';
CalKit.Name = '85052A_APC35';
[ CalKit ] = PNA_85052A_StdDef( CalKit )
%==================== SOLT_ErrTerm, SOLT_StdData w/(ABC StdData)
FileNam     = 'PNA_SOLT-freq_ABCD_1-2-8001Mhz_-20dBm_IFBW100Hz.mat';
    File_Load   = fullfile(PATH.Data, DUT, Char, FileNam)
    Meas_SOLT   = load( File_Load,'Meas','Opt' );
    Meas_SOLT   = Meas_SOLT.Meas
%==================== RxPwrCal_ErrTerm w/(EPM + RABC Data)
FileNam     = 'PNA_RxPwrCal-freq_CA_1-2-8001Mhz_-20dBm_IFBW100Hz.mat';
    File_Load = fullfile(PATH.Data, DUT, Char, FileNam)
Meas_Pwr    = load( File_Load,'Meas','Opt' );
    Meas_Pwr = Meas_Pwr.Meas
%==================== SigGen0_Mhz: common to Meas_SOLT + Meas_Pwr
[ CAL.StdVal  CAL.ErrTerm_Hz   ] = PNA_85052A_StdSparam( ...
    CalKit, Meas_SOLT.SigGen0_Mhz * 1e6 );
[ CAL.ErrTerm ] = LSNAncsu_SOLTpwr_ErrTerm( CAL.StdVal, ...
    Meas_SOLT.PNA_mWatt_RI, Meas_SOLT.Wav_Name, ...
```

```
    Meas_Pwr.PNA_mWatt_RI, Meas_Pwr.EPM_dBm, ...
    Meas_Pwr.Wav_Name, Meas_Pwr.SigGen0_Mhz * 1e6);


%% ################################################## LSNAncsu NL-meas
%==============================================
DUT     = 'SD513301';
Char    = 'LSNA';
FileNam = ['PNA_SD513301_2Tone-Ord5_0.0002-60Mhz20pt_' ...
        '-30-10dBm1dB_Lim1-20000Mhz_IFBW1Hz.mat'];
%==============================================
    File_Load = fullfile(PATH.Data, DUT, Char, FileNam)
    Meas_DUT    = load( File_Load,'Meas','Opt' );
    Meas_DUT    = Meas_DUT.Meas
%============================================== DUT: ErrCorr_SelFreq
[ temp MCorr.ErrTerm ] = PNA_ErrCorr_SelFreq( Meas_DUT.SigGen0_Mhz, ...
    [], [], CAL.ErrTerm, CAL.ErrTerm_Hz );
%============================================== ErrCorr_ABS (SOLT + RxPwr)
[ MCorr.Wav1 ] = LSNAncsu_ErrCorr_SOLTpwr( MCorr.ErrTerm, ...
    Meas_DUT.PNA_mWatt_RI, Meas_DUT.Wav_Name );
%============================================== ErrCorr_RelPha
[ MCorr.Wav2 ] = LSNAncsu_ErrCorr_RelPha( Meas_DUT.PNA_mWatt_RI.R, ...
    MCorr.Wav1,{'a1' 'b1' 'a2' 'b2'},{});
%============================================== ErrCorr_IOrelation
[ MCorr.Wav3  MCorr.WavIN ] = LSNAncsu_ErrCorr_IOrelation( ...
    MCorr.Wav2,{'a1' 'b1' 'a2' 'b2'},{}, Meas_DUT, MIX);
```

B.3.2 Correction Functions

PNA_85052A_StdDef Function

```
function [ CalKit ] = PNA_85052A_StdDef( CalKit )


switch CalKit.Name
%% ################### 85052A Rev March 1986 (family: APC3.5)
    case{ '85052A_APC35' }
Of.C0       = 58e-15;% F
Of.C1       = 200e-27;% F/Hz
Of.C2       = 0;% F/(Hz^2)
Of.C3       = 0;% F/(Hz^3)
Of.Delay    = 14.448e-12;% S
Of.Loss     = 0.13;% Gohm/s

Sf.L0       = 0;% H
Sf.L1       = 0;% H/Hz
Sf.L2       = 0;% H/(Hz^2)
Sf.L3       = 0;% H/(Hz^3)
Sf.Delay    = 16.684e-12;% S
Sf.Loss     = 0.13;% Gohm/s

LBROADf.Type    = 'fixed';
LBROADf.Delay = 0;% S
LBROADf.Loss   = 0;% Gohm/s

LLOWBANDf.Type  = 'fixed';
```

```
LLOWBANDf.Delay = 0;% S
LLOWBANDf.Loss  = 0;% Gohm/s

T.Delay     = 0;% S
T.Loss      = 0;% Gohm/s

%% ################### 85052A 3.5mm A.1 (family: 3.5mm)
    case{ '85052A_35mm' }
Of.C0       = -9.7e-15;% F
Of.C1       = 2150e-27;% F/Hz
Of.C2       = -39e-36;% F/(Hz^2)
Of.C3       = 0;% F/(Hz^3)
Of.Delay    = 0;% S
Of.Loss     = 1.3;% Gohm/s

Sf.L0       = 0;% H
Sf.L1       = 0;% H/Hz
Sf.L2       = 0;% H/(Hz^2)
Sf.L3       = 0;% H/(Hz^3)
Sf.Delay    = 16.684e-12;% S
Sf.Loss     = 0.13;% Gohm/s

LBROADf.Type    = 'fixed';
LBROADf.Delay = 0;% S
LBROADf.Loss   = 0;% Gohm/s

LLOWBANDf.Type  = 'fixed';
```

```
LLOWBANDf.Delay = 0;% S
LLOWBANDf.Loss  = 0;% Gohm/s

T.Delay     = 0;% S
T.Loss      = 0;% Gohm/s

%% ################### 85052B (family: APC3.5)
    case{ '85052B_APC35' }
Of.C0       = 49.433e-15;% F
Of.C1       = -310.13e-27;% F/Hz
Of.C2       = 23.168e-36;% F/(Hz^2)
Of.C3       = -0.15966e-45;% F/(Hz^3)
Of.Delay    = 29.243e-12;% S
Of.Loss     = 2.2;% Gohm/s

Sf.L0       = 2.0765e-12;% H
Sf.L1       = -108540e-27;% H/Hz
Sf.L2       = 2170.5e-36;% H/(Hz^2)
Sf.L3       = -10e-45;% H/(Hz^3)
Sf.Delay    = 31.785e-12;% S
Sf.Loss     = 2.36;% Gohm/s

LBROADf.Type    = 'fixed';
LBROADf.Delay = 0;% S
LBROADf.Loss   = 0;% Gohm/s

LLOWBANDf.Type  = 'fixed';
```

```
LLOWBANDf.Delay = 0;% S
LLOWBANDf.Loss  = 0;% Gohm/s

T.Delay     = 0;% S
T.Loss      = 0;% Gohm/s
%% ###################
end

CalKit.Sf       = Sf;
CalKit.Of       = Of;
CalKit.LBROADf      = LBROADf;
CalKit.LLOWBANDf    = LLOWBANDf;
CalKit.T        = T;
```

PNA_85052A_StdSparam Function

```
function [ StdVal fHz ] = PNA_85052A_StdSparam( CalKit, fHz )
% fHz: Hz
% StdVal.(TS/TO/TL)
Zr = 50;

Sf          = CalKit.Sf;
Of          = CalKit.Of;
LBROADf    = CalKit.LBROADf;
T           = CalKit.T;

%% ################################################
LS          = Sf.L0 + Sf.L1*fHz + Sf.L2*(fHz.^2) + Sf.L3*(fHz.^3);
```

```
ZS          = j*2*pi.* fHz .* LS;
StdVal.TS   = (ZS - Zr) ./ (ZS + Zr);

CO          = Of.C0 + Of.C1*fHz + Of.C2*(fHz.^2) + Of.C3*(fHz.^3);
ZO          = 1 ./ (j*2*pi.* fHz .* CO);
StdVal.TO   = (ZO - Zr) ./ (ZO + Zr);

StdVal.TL   = 0;
```

LSNAncsu_SOLTpwr_ErrTerm Function

```
function [ ErrTerm ] = LSNAncsu_SOLTpwr_ErrTerm( ...
    StdVal, PNA_mWatt_RI_SOLT, WaveName_SOLT, ...
    PNA_mWatt_RI_RxPwr, EPM_dBm, WavName_RxPwr, fHz )
% fHz: common to SOLT + RxPwr, else need to interpolate
if( length( fHz ) ~= length( StdVal.TS ))
    disp('unequal #freq-pts during SOLT-cal and RxPwr-Cal');
    return;
end

%================================== fwd
[ B1 G1 D1 ] = Solve_Bgd(  StdVal.TS, StdVal.TO, StdVal.TL, ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S11_S_m1), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S11_S_m2), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S11_O_m1), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S11_O_m2), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S11_L_m1), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S11_L_m2) );
```

```
ErrTerm.B1 = B1;
    ErrTerm.G1 = G1;
    ErrTerm.D1 = D1;
%================================== rev
[ B2_ G2_ D2_ ] = Solve_Bgd(  StdVal.TS, StdVal.TO, StdVal.TL, ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S22_S_m1), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S22_S_m2), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S22_O_m1), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S22_O_m2), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S22_L_m1), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S22_L_m2) );
[ A2 ] = Solve_A( B1, G2_, D2_, ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S12_T_m1), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S12_T_m2), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S12_T_m3), ...
                        PNA_mWatt_RI_SOLT.(WaveName_SOLT.S12_T_m4) );
    ErrTerm.A2 =        A2;
ErrTerm.B2 = B2_ .* A2;
    ErrTerm.G2 = G2_ .* A2;
    ErrTerm.D2 = D2_ .* A2;
%================================== RxPwr
K   = dB2Lin( EPM_dBm.(WavName_RxPwr.EPM_m1) ...
            - Lin2dB( PNA_mWatt_RI_RxPwr.(WavName_RxPwr.P1_m1) ...
            + B1.*PNA_mWatt_RI_RxPwr.(WavName_RxPwr.P1_m2) ) ...
              );
    ErrTerm.K = K;
```

```
%% ##################################################
function [ B G D ] = Solve_Bgd( T1, T2, T3, a1, b1, a2, b2, a3, b3 )

B = -1./(-b3.*a2.*T1.*b1+b3.*T2.*b2.*a1-b3.*b2.*T3.*a1-T2.*b2.*b1.*a3...
+b3.*a2.*b1.*T3+b2.*a3.*T1.*b1).*(b3.*T2.*a2.*a1+b2.*a3.*T1.*a1...
-b3.*a2.*T1.*a1+a2.*b1.*T3.*a3-T2.*a2.*b1.*a3-b2.*T3.*a3.*a1);
G = (a1.*b3.*T1.*T2.*b2-a1.*b3.*T1.*b2.*T3+b3.*b1.*T3.*T2.*a2...
-b3.*b1.*T1.*T2.*a2...+b1.*T1.*b2.*T3.*a3-b1.*T3.*a3.*T2.*b2)...
./(-b3.*a2.*T1.*b1+b3.*T2.*b2.*a1-b3.*b2.*T3.*a1-T2.*b2.*b1.*a3...
+b3.*a2.*b1.*T3+b2.*a3.*T1.*b1);
D = -(T1.*a2.*b1.*T3.*a3-T3.*a3.*T2.*b2.*a1+T3.*b3.*T2.*a2.*a1...
-T3.*b3.*a2.*T1.*a1+a3.*a1.*T1.*T2.*b2-T1.*T2.*a2.*b1.*a3)...
./(-b3.*a2.*T1.*b1+b3.*T2.*b2.*a1-b3.*b2.*T3.*a1-T2.*b2.*b1.*a3...
+b3.*a2.*b1.*T3+b2.*a3.*T1.*b1);

return

%% ##################################################
function [ A ] = Solve_A( B1, G2_, D2_, a1, b1, a2, b2 )
A = (a1 + B1.*b1) ./ (G2_.*a2 + D2_.*b2);
return
```

PNA_ErrCorr_SelFreq Function

```
function [ RXpwrcorr_ ErrTerm_ ] = PNA_ErrCorr_SelFreq( fMHz, ...
    RXpwrcorr, RXpwrcorr_Hz, ...
    ErrTerm,  ErrTerm_Hz )
```

```
% Wav_Name{n}: RXpwrcorr
% Wav_Name{n}: ErrTerm
%% ################################################
for nDF = 1 : length( fMHz )
    fHz = [fMHz{ nDF } * 1e6].';
%=================================== RXpwrcorr (NOT used for ABSproc)
if ~(isempty(RXpwrcorr) || isempty(RXpwrcorr_Hz))
    Wav_Name = fieldnames( RXpwrcorr );
    for nWAV = 1 : length( Wav_Name )
        Wav_Name{nWAV}% DEBUG
        %-------------------------------------------------------
        if ~isa(RXpwrcorr_Hz,'struct')% use same [RXpwrcorr_Hz]
            temp = Interp1_Func( RXpwrcorr_Hz, ...
                RXpwrcorr.( Wav_Name{nWAV} ), ...
                fHz, struct('xScale',{'xLIN'}, 'yFormat',{'RI'}, ...
                'Meth',{'linear'}) );
        else
            temp = Interp1_Func( RXpwrcorr_Hz.( Wav_Name{nWAV} ),...
                RXpwrcorr.( Wav_Name{nWAV} ), ...
                fHz, struct('xScale',{'xLIN'}, 'yFormat',{'RI'}, ...
                'Meth',{'linear'}) );
        end
        RXpwrcorr_.( Wav_Name{nWAV} ){nDF} = temp;
        %-------------------------------------------------------
    end
else
    RXpwrcorr_ = [];
```

```
end
%====================================== ErrTerm
if ~(isempty(ErrTerm) || isempty(ErrTerm_Hz))
    Wav_Name = fieldnames( ErrTerm );
    for nWAV = 1 : length( Wav_Name )
        %Wav_Name{nWAV}% DEBUG
        %-----------------------------------------------------------
        if ~isa(ErrTerm_Hz,'struct')% use same [ErrTerm_Hz]
            temp = Interp1_Func( ErrTerm_Hz, ...
                ErrTerm.( Wav_Name{nWAV} ), ...
                fHz, struct('xScale',{'xLIN'}, 'yFormat',{'RI'}, ...
                'Meth',{'linear'}) );
        else
            temp = Interp1_Func( ErrTerm_Hz.( Wav_Name{nWAV} ), ...
                ErrTerm.( Wav_Name{nWAV} ), ...
                fHz, struct('xScale',{'xLIN'}, 'yFormat',{'RI'}, ...
                'Meth',{'linear'}) );
        end
        ErrTerm_.( Wav_Name{nWAV} ){nDF} = temp;
        %-----------------------------------------------------------
    end
else
    ErrTerm_ = [];
end
%======================================
end
```

LSNAncsu_ErrCorr_SOLTpwr Function

```
function [ Wav1 ]   = LSNAncsu_ErrCorr_SOLTpwr( ErrTerm, ...
    PNA_mWatt_RI, Wav_Name )


%% ############################################## PNA_mWatt_RI * RxPwrErr
for nDF = 1 : length( PNA_mWatt_RI.( Wav_Name.R ) )
%=====================================================================
nP = size( PNA_mWatt_RI.( Wav_Name.R ){nDF}, 2);% DEBUG
%###################### special case: data W/b2, but WO/a2
if(isfield( Wav_Name,'P1_m4') && isfield( Wav_Name,'P2_m4'))
    PNA_mWatt_RI.( Wav_Name.P2_m3 ){nDF} = ...
        zeros(size( PNA_mWatt_RI.( Wav_Name.P1_m4 ){nDF} ));
end
%######################
[ a1 b1 a2 b2 ] = m2ab_Wave( ...
                PNA_mWatt_RI.( Wav_Name.P1_m1 ){nDF}, ...
                PNA_mWatt_RI.( Wav_Name.P1_m2 ){nDF}, ...
                PNA_mWatt_RI.( Wav_Name.P2_m3 ){nDF}, ...
                PNA_mWatt_RI.( Wav_Name.P2_m4 ){nDF}, ...
                repmat( ErrTerm.B1{nDF}, [1 nP]), ...
                repmat( ErrTerm.G1{nDF}, [1 nP]), ...
                repmat( ErrTerm.D1{nDF}, [1 nP]), ...
                repmat( ErrTerm.B2{nDF}, [1 nP]), ...
                repmat( ErrTerm.G2{nDF}, [1 nP]), ...
                repmat( ErrTerm.D2{nDF}, [1 nP]), ...
                repmat( ErrTerm.A2{nDF}, [1 nP]), ...
                repmat( ErrTerm.K{nDF},  [1 nP]) );
```

```
Wav1.a1{nDF} = a1;

Wav1.b1{nDF} = b1;

Wav1.a2{nDF} = a2;

Wav1.b2{nDF} = b2;

%================================================================

end


%% ################################################

function [ a1 b1 a2 b2 ] = m2ab_Wave( m1, m2, m3, m4, ...

    B1, G1, D1, ...

    B2, G2, D2, ...

    A2, K)


a1 = K .* (    m1   + B1.*m2);

b1 = K .* (G1.*m1   + D1.*m2);

a2 = K .* (A2.*m3   + B2.*m4);

b2 = K .* (G2.*m3   + D2.*m4);


return
```

LSNAncsu_ErrCorr_RelPha Function

```
function [ Wav2 ] = LSNAncsu_ErrCorr_RelPha( R, ...

    Wav1, Wav_Name, Wav_Name_COPY, Opt_MeasType)

if ~exist('Opt_MeasType','var')

    Opt_MeasType = 'otherwise';

end
```

```
%% ################################################## MeasCorr Wave
switch Opt_MeasType
%================================================== Cosite 2Tone
    case{'{nDF,nP1}(nF,nP2)'}
NDF     = size( Wav1.(Wav_Name{1}),1 );
NP1     = size( Wav1.(Wav_Name{1}),2 );
for nDF = 1 : NDF
for nP1 = 1 : NP1
    %-----------------------------------------
    PhaRef_RI = Norm_RI( R{nDF,nP1} );
    for nWAV = 1 : length( Wav_Name )
        Wav2.(Wav_Name{nWAV}){nDF,nP1} = Wav1.(Wav_Name{nWAV}){nDF,nP1} ...
                                    ./ PhaRef_RI;
    end
    %-----------------------------------------
end
end
%================================================== 2Tone MemoryTest
    otherwise
NDF     = length( Wav1.(Wav_Name{1}) );
for nDF = 1 : NDF
    %-----------------------------------------
    PhaRef_RI = Norm_RI( R{nDF} );
    for nWAV = 1 : length( Wav_Name )
        Wav2.(Wav_Name{nWAV}){nDF} = Wav1.(Wav_Name{nWAV}){nDF} ...
                                    ./ PhaRef_RI;
    end
```

```
    %-------------------------------------------
end
%==================================================
end

%% ################################################## COPY
for nWAV = 1 : length( Wav_Name_COPY )
    Wav2.(Wav_Name_COPY{nWAV}) = Wav1.(Wav_Name_COPY{nWAV});
end
```

LSNAncsu_ErrCorr_IOrelation Function

```
function [ Wav3  WavIN ] = LSNAncsu_ErrCorr_IOrelation( ...
    Wav2, Wav_Name, Wav_Name_COPY, ...
    Meas, MIX, ...
    Opt_MeasType)
if ~exist('Opt_MeasType','var')
    if isempty( Meas.SigGen2_Mhz )
        Opt_MeasType = '1Tone';
    else
        Opt_MeasType = '2Tone';
    end
end

switch Opt_MeasType
%% ##################################################
    case{'1Tone'}
Wav3.a1 = Wav2.a1;
```

```
%================================================
NDF     = length( Wav2.(Wav_Name{1}) );
for nDF = 1 : NDF
%------------------------------------------------
[NF NP]         = size( Wav2.a1{nDF} );
    ExpMAT          = repmat( [1:NF].', [1 NP] );
    a1_f1_MAT       = repmat( Wav2.a1{nDF}(1,:), [NF 1] );
    PhaRef_RI_MAT   = Norm_RI( a1_f1_MAT .^ ExpMAT );
for nWAV = 1 : length( Wav_Name )
    switch Wav_Name{nWAV}
        case{'b1','a2','b2'}
            Wav3.( Wav_Name{nWAV} ){nDF} = Wav2.( Wav_Name{nWAV} ){nDF} ...
                                    ./ PhaRef_RI_MAT;
    end
end
%------------------------------------------------
end
%% ################################################
    case{'2Tone'}
Wav3.a1 = Wav2.a1;
%================================================
    nMIX_f1 = Meas.fnumMAT_f12IDX(:,1);
    nMIX_f2 = Meas.fnumMAT_f12IDX(:,2);
NDF     = length( Meas.SigGen0_Mhz );
for nDF = 1 : NDF
    WavIN.a1_f1{nDF} = Wav2.a1{nDF}( nMIX_f1(nDF),:);% 'a1'
    WavIN.a1_f2{nDF} = Wav2.a1{nDF}( nMIX_f2(nDF),:);% 'a1'
```

```
for nMIX = 1 : length( Meas.SigGen0_Mhz{ nDF } )
    idx2_fnumMAT_ALL = Meas.SigGen0_Mhz_IDXcel{nDF}{nMIX};
    fmixVEC        = MIX.fmix_ALL( idx2_fnumMAT_ALL, : );%use MIX
    PhaRef_RI_VEC   = PhaseRef_2Tone( {WavIN.a1_f1{nDF}, ...
        WavIN.a1_f2{nDF}}, fmixVEC );
%-------------------------------------------
for nWAV = 1 : length( Wav_Name )
    switch Wav_Name{nWAV}
        case{'b1','a2','b2'}
            Wav3.( Wav_Name{nWAV} ){nDF}(nMIX,:) = ...
                Wav2.( Wav_Name{nWAV} ){nDF}(nMIX,:) ...
                                    ./ PhaRef_RI_VEC;
    end
end
%-------------------------------------------
end
end
%% ################################################
    case{'2Tone_Cosite'}%,'2Tone_WCDMA'
Wav3.A = Wav2.A;
Wav3.B = Wav2.B;
%================================================
    nMIX_f1 = Meas.fnumMAT_f12IDX(:,1);
    nMIX_f2 = Meas.fnumMAT_f12IDX(:,2);

NDF     = length( Meas.SigGen0_Mhz );
for nDF = 1 : NDF
```

```
for nP1 = 1 : length( Meas.SigGen1_dBm )
    WavIN.A_f1{nDF,nP1} = Wav2.A{nDF,nP1}( nMIX_f1(nDF),:);% 'A'
    WavIN.B_f2{nDF,nP1} = Wav2.B{nDF,nP1}( nMIX_f2(nDF),:);% 'B'
for nMIX = 1 : length( Meas.SigGen0_Mhz{ nDF } )
    idx2_fnumMAT_ALL = Meas.SigGen0_Mhz_IDXcel{nDF}{nMIX};
    fmixVEC        = MIX.fmix_ALL( idx2_fnumMAT_ALL, : );%use MIX
    PhaRef_RI_VEC   = PhaseRef_2Tone( {WavIN.A_f1{nDF,nP1}, ...
        WavIN.B_f2{nDF,nP1}}, fmixVEC );
%-----------------------------------------------
for nWAV = 1 : length( Wav_Name )
            Wav3.( Wav_Name{nWAV} ){nDF,nP1}(nMIX,:) = ...
                Wav2.( Wav_Name{nWAV} ){nDF,nP1}(nMIX,:) ...
                                    ./ PhaRef_RI_VEC;
end
%-----------------------------------------------
end
end
end
%% ##################################################
end

%% ################################################## COPY
for nWAV = 1 : length( Wav_Name_COPY )
    Wav3.(Wav_Name_COPY{nWAV}) = Wav2.(Wav_Name_COPY{nWAV});
end

%% ##################################################
```

```
function [ PhaRef_RI_VEC ] = PhaseRef_2Tone( a1_f1f2_CEL, fmixVEC )


PhaRef_RI_VEC = ones(size( a1_f1f2_CEL{1} ));
for q = 1 : length( fmixVEC )
    if(fmixVEC(q) < 0)
        PhaRef_RI_VEC = PhaRef_RI_VEC .* conj(a1_f1f2_CEL{q} ...
            .^ abs( fmixVEC(q) ));
    elseif(fmixVEC(q) > 0)
        PhaRef_RI_VEC = PhaRef_RI_VEC .*     (a1_f1f2_CEL{q} ...
            .^ abs( fmixVEC(q) ));
    end
end
PhaRef_RI_VEC = Norm_RI( PhaRef_RI_VEC );


return
```

Appendix C

Grey-Box Model ID: Hybrid Genetic Optimizer

C.1 User Guide

The Matlab code of the hybrid genetic algorithm optimizer used in Grey-Box model identification is given here. The complete collection of Matlab code is also available at Mathworks file exchange website http://www.mathworks.com/matlabcentral/fileexchange/31179, Hybrid Genetic Optimizer for Grey-box Model Identification, so that user feedback can be easily incorporated in future code improvements.

Section C.2 contains the Matlab code used in Chapter 4 to compare the performance of hybrid genetic optimizer versus the performance of the conventional genetic algorithm optimizer. The main script shows the typical way the optimizer functions are called. Then the optimizer functions are presented. For both optimizers, the optimization options, the parameters being optimized and the measurement data being fitted are passed to the optimizer functions as input variables. On the other hand, the error function is "hard coded" in the optimizer functions. The error function included here computes the model fit error for the power amplifier model discussed in Section 4.3.

Section C.3 contains the Matlab code used in model parameter identification of the power amplifier model discussed in Section 4.3. The main script are broken into three sections. The first section sets up the modeling problem by setting the model order, the input excitation, the measured data to be fitted, and the model response to be computed. After setting up the modeling problem, the second section presents a proprietary model identification solution (not discussed in the dissertation) for completeness and comparison purposes. This proprietary solution also identifies the model parameters in several stages, with the initial estimate based on heuristics. What's different from the solution presented in Chapter 4 is that the nonlinear least square optimizer is used at each stage. This proprietary solution is the precursor to the iterative identification algorithm discussed in Chapter 4. The last section contains the same main script presented in Section C.2 to invoke either the conventional genetic algorithm optimizer or the hybrid genetic optimizer. All other functions called in the script can be found in the file archive at Mathworks file exchange.

C.2 Hybrid Genetic Optimizer

C.2.1 Main Script

```
%% ######################################################## GA
Opt.Optim_Meth = 'ga';% 'lsqnonlin';%
GA.TimeLimit        = 10*3600;% 10hr
GA.FitnessLimit     = 1;
GA.StallGenLimit = 500;
    %-----------------------------------------------
GA.Run = 1;
GA.gen = 300;       GA.TolFun = 1e-5;
    GA.pop = 250; Mark = 'ko';
    %-----------------------------------------------
```

```
    GA.FileNam = [Opt.DataType 'GA' '_N' num2str(Opt.N) '-PinIdx' ...
        num2str(Opt.MaxPin_IDX) '-' Opt.Optim_MtoneFit ...
        '_' Opt.Optim_Wt num2str(Opt.Optim_WtConst) ...
        '_' Opt.Optim_Err '-' Opt.Optim_Err2 ...
       '_' Opt.Nth_Atmpt{1} '-' Opt.Nth_Atmpt{2} '-' Opt.Nth_HD2band ...
       '_' num2str(GA.Run) '-Pop' num2str(GA.pop) '-Gen' num2str(GA.gen)];
    GA.FileNam
    GA.FileSav = fullfile(PATH.MatlabSave, [GA.FileNam '.mat']);
     whos('-file', GA.FileSav);
%%=====================================================
[GA.ITER1]      = Buss_3_GA( Opt, ...
    [], GA, ...%PATH,
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell);
[b1  InputNumCEL  aNumCEL  zxNumQMIX  Opt.y]  = ...
    Buss_XR4_XReal2zxNumQMIX_2(GA.ITER1.xReal, Opt, InputSymCEL, 1);
    Opt.y
    GA.ITER1.Opt = Opt;
    save(GA.FileSav, 'GA', '-append');
    whos('-file', GA.FileSav);% load(GA.FileSav);

%% ######################################################  Hybrid-GA
    HYB.Opt              = Opt;
      HYB.Opt.Optim_Meth = 'lsqnonlin';
    HYB.Optim_OutputFcn = '';
    HYB.MaxIter         = 100;
    HYB.TolFun          = 0.001;
```

```
HYB.PrecondBandWidth = inf;% Inf-ChoskyFactor Better than 0=ConjGrad;
    GA.HYB = HYB;
%FileNam_HYB = [ GA.FileNam ...
%    '_LSQ_Iter' num2str(HYB_Opt.MaxIter)]% '_Soln' num2str(NumBestSoln)
    NumGenVEC       = [1 10];
    NumBestSolnVEC = [1 2];% [1 5] [6 10] [1 10]
%%=====================================================
[GA.ITER1.State_HYBRID] = Buss_3_HYBRID( GA, ...
    NumBestSolnVEC, NumGenVEC, ...
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell)
        %iter.Opt     = Opt;
        %iter.GA       = GA;
```

C.2.2 Functions

Conventional Genetic Optimizer Function

```
function [iter] = Buss_3_GA( FileNam, FileSav, iter, ...
    LSQ, FMINsear, GA, ...
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell)
Opt = iter.Opt;


NumParam = length(Opt.xReal_0);
switch Opt.Optim_Meth
    %================================
    case{'lsqnonlin','fminsearch'}
        [LSQ.MaxIter            NumParam]
        [FMINsear.MaxIter      NumParam]
        FileSav_LSQ = FileSav;
        save('FileNam_LSQ.mat', 'FileSav_LSQ')% filename var
    %================================
    case{'ga'}
        [GA.pop     GA.gen      NumParam]
        FileSav_GA = FileSav;
        save('FileNam_GA.mat', 'FileSav_GA')% filename var
    %================================
end


if( exist(FileSav, 'file') )
%--------------------------------
```

```
whos('-file', FileSav);
FileOpt = input([FileNam ' exist [0:del |1:keep ]: '], 's');
switch FileOpt
      %----------------
        case{'0'}
            delete(FileSav);
            FileCreat   = 1;
      %----------------
     %case{'2'}
        %  FileCreat = 1;
        %   FileNam     = input(['FileNam ' Opt.Optim_Meth ': '], 's');
        %   FileSav     = fullfile(PATH.MatlabSave, [FileNam '.mat'])
      %----------------
        otherwise% case{'1'}
            FileCreat   = 0;
      %----------------
end
else%---------------------------------
FileCreat = 1;
%---------------------------------
end
if FileCreat
    switch Opt.Optim_Meth
        case{'lsqnonlin'}
            save(FileSav_LSQ, 'LSQ');
        case{'fminsearch'}
            save(FileSav_LSQ, 'FMINsear');
```

```
        case{'ga'}
            save(FileSav_GA, 'GA');
    end
end

pause
%% ================================================== ITER1
switch Opt.Optim_Meth
    %====================================================
    case{'lsqnonlin'}
        LSQ.Optim_Options = optimset(Opt.Optim_Meth);
        LSQ.MaxFunEvals     = 400 * NumParam;
        LSQ.Optim_Options  = optimset(LSQ.Optim_Options,...
            'PrecondBandWidth',LSQ.PrecondBandWidth, ...
          'MaxIter',LSQ.MaxIter, ...% 'MaxIter',400,
            'OutputFcn',{@Plot_LSQ_Func2}, ...
            'TolFun',LSQ.TolFun,'MaxFunEvals',LSQ.MaxFunEvals);
        % 'TolFun',0.1, 'MaxFunEvals',100*
        % {'Display','final', 'notify' 'iter' 'final'}
        % 'TolX',1e-4,
RunTime_LSQ = tic;
[x, resnorm, residual, exitflag, output] = lsqnonlin( @(x) ...
   Buss_XR4_Iter( x, Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
  Opt.xReal_0, [],[],LSQ.Optim_Options);
RunTime_LSQ = toc(RunTime_LSQ)
```

```
        %----------------------
        iter.xReal_0 = Opt.xReal_0;
        iter.xReal     = x;
       iter.residual = residual;
       iter.output    = output;
        iter.RunTime    = RunTime_LSQ;
            iter.Pop_LSQ = load(FileSav_LSQ, 'Pop_LSQ');
            iter.Pop_LSQ = iter.Pop_LSQ.Pop_LSQ;
    %=====================================================
    case{'fminsearch'}%'OutputFcn' problem: dont like 'FunValCheck','on'
        %NOT PREFERRED to solve sum-of-square problem => lsqnonlin
        %only minimizes over the real, f(x) must return real
        %COMPLEX VARIABLES must be split into R+jI
     %robust for discontin problem, less efficient than fminunc ord >2
        FMINsear.Optim_Options = optimset(Opt.Optim_Meth);
        FMINsear.MaxFunEvals    = 400 * NumParam;
        FMINsear.MaxIter        = FMINsear.MaxIter * NumParam / 4;
        FMINsear.Optim_Options  = optimset(FMINsear.Optim_Options, ...
          'MaxIter',FMINsear.MaxIter, ...
            'OutputFcn',@Plot_LSQ_Func2, ...%
            'TolFun',FMINsear.TolFun,'MaxFunEvals',FMINsear.MaxFunEvals);
        % fminsearch=[1e-4 1e-4]
        % {'Display','final', 'notify' 'iter' 'final'}
        % 'FunValCheck','on',
        % 'TolX',1e-4,
RunTime_LSQ = tic;
[x, fval, exitflag, output] = fminsearch( @(x) ...
```

```
    Buss_XR4_Iter( x, Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
  Opt.xReal_0, FMINsear.Optim_Options);
RunTime_LSQ = toc(RunTime_LSQ)
        %----------------------
        iter.xReal_0 = Opt.xReal_0;
        iter.xReal     = x;% x of next to Last iter
       iter.output    = output;
        iter.RunTime    = RunTime_LSQ;
            iter.Pop_LSQ = load(FileSav_LSQ, 'Pop_LSQ');
            iter.Pop_LSQ = iter.Pop_LSQ.Pop_LSQ;
    %=====================================================
    case{'fminunc'}
        %NOT PREFERRED to solve sum-of-square problem => lsqnonlin
        %LargeScale:  must provide the gradient
        %Medium-scale: Quasi-Newton line search (Algorithm selected)
      %Warning: Gradient must be provided for trust-region method
        %       =>  using line-search method instead.
        %Optimization terminated: relative infinity-norm of gradient
        %       less than options.TolFun.
RunTime_LSQ = tic;
[x, fval, exitflag, output] = fminunc( @(x) ...
   Buss_XR4_Iter( x, Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
  Opt.xReal_0, FMINunc.Optim_Options);
```

```
        % fminunc=[1e-6 1e-6]
        % 'PrecondBandWidth',LSQ.PrecondBandWidth, ...
        % {'Display','final', 'notify' 'iter' 'final'}
        % 'FunValCheck','on', 'Diagnostics','on'
        % 'TolX',1e-4,
RunTime_LSQ = toc(RunTime_LSQ)
        %----------------------
    %=====================================================
    case{'ga'}
        GA.Optim_Options = gaoptimset;
        GA.Optim_Options  = gaoptimset(GA.Optim_Options, ...
           'PopulationSize',GA.pop, 'Generations',GA.gen,...
            ...     % 'InitialPopulation',final_pop,
            'PlotFcns',{@Plot_GA_Func}, 'PlotInterval',1,...
            ...     % @gaplotbestf @gaplotbestindiv @gaplotdistance
            'FitnessLimit',GA.FitnessLimit, 'TimeLimit',GA.TimeLimit,...
            ...     %500 default=-inf
            'TolFun',GA.TolFun, 'StallTimeLimit',GA.TimeLimit,...
            'StallGenLimit',GA.StallGenLimit)
            % 'FitnessLimit' = (RelErr 0.1 ~ -10dB) * Wt * #pts
            % 'Display','iter' , ...
            % 'PopInitRange', [x2INTERMED-(1+j); x2INTERMED+(1+j)],
            %       [Opt.xReal_0-(1+j); Opt.xReal_0+(1+j)]
            % 'HybridFcn',{@fminunc}, @fminsearch
            % MigrationDirection
%output.XXX
%randstate       rand  state before the algorithm started.
```

```
%randnstate      randn state (randstate + randnstate to reproduce)
%generations     # generations computed.
%funccount       # evaluations of fitness function
%message     same as output argument reason (why algorithm terminated)
%maxconstraint   Maximum constraint violation, if any
    %rand('state',  output.randstate);
    %randn('state', output.randnstate);
RunTime_GA = tic;
[x, F2, reason, output, final_pop, final_score] = ga(@(x) ...
   Buss_XR4_Iter( x, Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
NumParam, [],[],[],[],[],[],[],GA.Optim_Options);
RunTime_GA = toc(RunTime_GA)
        %----------------------
        iter.xReal      = x.';% GA pass ROWvec
        iter.output    = output;
        iter.final_score = final_score;
        iter.RunTime    = RunTime_GA;
            iter.Pop_GA     = load(FileSav_GA, 'Pop_GA');
            iter.Pop_GA     = iter.Pop_GA.Pop_GA;
    %=================================================
end
```

Hybrid Genetic Optimizer Function

```
function [State_HYBRID] = Buss_3_HYBRID( GA, ...
    NumBestSolnVEC, NumGenVEC, ...
```

```
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell)


HYB = GA.HYB;
load(GA.FileSav, 'Pop_GA');% 'StateSave','ScoreSortIDX')
    whos('-file', GA.FileSav);
%================================================
    temp = load(GA.FileSav);
if( isfield(temp, 'State_HYBRID') )
        %---------------------------------
        State_HYBRID          = temp.State_HYBRID;
        [NumGen  NumBestSoln] = size(State_HYBRID.Score)
      NumParam              = size(Pop_GA.StateSave(1).Population, 2)
    else
        %---------------------------------
        NumGen          = size(Pop_GA.StateSave, 2)
        NumParam        = size(Pop_GA.StateSave(1).Population, 2)
        State_HYBRID.Runtime    = zeros(NumGen,NumBestSolnVEC(2));
        State_HYBRID.Score      = zeros(NumGen,NumBestSolnVEC(2));
        State_HYBRID.Population=zeros(NumGen,NumBestSolnVEC(2),NumParam);
        %State_HYBRID(ngen, nsoln).Output = output;
        %---------------------------------
        switch HYB.Optim_OutputFcn
            case{'Plot_LSQ_Func'}
        State_HYBRID.Pop0  = zeros(NumGen, NumBestSolnVEC(2), NumParam);
        State_HYBRID.Optimval0  = zeros(NumGen, NumBestSolnVEC(2));
        end
```

```
        %--------------------------------
        save(GA.FileSav, 'State_HYBRID', '-append');
    end
  clear temp
%================================================
FileSav_LSQ = GA.FileSav;
save('FileNam_LSQ.mat', 'FileSav_LSQ')% filename var
%================================================

[NumGenVEC]
[NumBestSolnVEC]
NumGenOpt = input(['use [0:NumGen |1:NumGenVEC]: '], 's');
    switch NumGenOpt
        case{'0'}
            NumGenVEC = [1 NumGen];
    end
    [NumBestSolnVEC     NumGenVEC       HYB.MaxIter     NumParam]
pause
%% ######################################## HYBRID
HYB.Optim_Options = optimset(HYB.Opt.Optim_Meth);
HYB.MaxFunEvals     = 400 * NumParam;
switch HYB.Optim_OutputFcn
    %=================================================
    case{'Plot_LSQ_Func'}
        HYB.Optim_Options = optimset(HYB.Optim_Options, ...
        'PrecondBandWidth', HYB.PrecondBandWidth, ...
        'MaxIter',HYB.MaxIter, ...
```

```
        'OutputFcn',{@Plot_LSQ_Func},...
        ... %{'Display','final', 'notify' 'iter' 'final'}
        'TolFun',HYB.TolFun, 'MaxFunEvals',HYB.MaxIter);
        % Default=[100*,400], [400*length(x2_0), 600]
    %=====================================================
    otherwise
        HYB.Optim_Options = optimset(HYB.Optim_Options, ...
        'PrecondBandWidth',HYB.PrecondBandWidth, ...
        'MaxIter',HYB.MaxIter, ...
        ...%'OutputFcn',{@Plot_LSQ_Func},...
        ...%{'Display','final', 'notify' 'iter' 'final'}
        'TolFun',HYB.TolFun, 'MaxFunEvals',HYB.MaxIter);
        % Default=[100*,400], [400*length(x2_0), 600]
    %=====================================================
end

%% ######################################## HYBRID
for ngen = NumGenVEC(1) : NumGenVEC(2)
    %[ScoreSort, ScoreSortIDX] = sort( ...
    %       Pop_GA.StateSave(ngen).Score, 'ascend');
for nsoln = NumBestSolnVEC(1) : NumBestSolnVEC(2)
x0_HYBRID   = Pop_GA.StateSave(ngen).Population( ...
            Pop_GA.ScoreSortIDX(ngen,nsoln), :);
x0_HYBRID   = x0_HYBRID.';% % GA pass ROWvec
switch HYB.Opt.Optim_Meth
%=====================================================
   case{'lsqnonlin'}
```

```
%output.XXX
%iterations     Number of iterations taken
%funcCount      # function evaluations
%algorithm      Algorithm used
%cgiterations   # PCG iterations (large-scale algorithm only)
%stepsize       final step size taken (medium-scale algorithm only)
%firstorderopt Meas 1st-ord optimality (lg-scale algorithm only)
%       lg-scale bound constrain problem: inf-norm of v.*g,
%       v:as in Box Constraint, g:gradient g=JTF (see NLlsq).
RunTime_HYBRID = tic;
[x_HYBRID, resnorm,residual,exitflag,output] = lsqnonlin( @(x) ...
      Buss_XR4_Iter( x, HYB.Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
            x0_HYBRID, [],[],HYB.Optim_Options);
RunTime_HYBRID = toc(RunTime_HYBRID)
    %----------------------
    %save(GA.FileSav, 'RunTime_HYBRID','residual','x_HYBRID', '-append');
    State_HYBRID(ngen, nsoln).Runtime = RunTime_HYBRID;
    State_HYBRID(ngen, nsoln).Score = sum(abs(residual));%=sum(abs(F2))
    State_HYBRID(ngen, nsoln, :).Population = x_HYBRID;%
    State_HYBRID(ngen, nsoln).Output  = output;
    %----------------------
    switch HYB.Optim_OutputFcn
        case{'Plot_LSQ_Func'}
        load(FileSav_LSQ, 'Pop_LSQ');
        State_HYBRID(ngen, nsoln, :).Pop0   = Pop_LSQ.Pop_0;
```

```
        State_HYBRID(ngen, nsoln).Optimval0 = Pop_LSQ.Optimval_0;
        State_HYBRID(ngen, nsoln, :,:).PopIter  = Pop_LSQ.Pop_Iter;
        State_HYBRID(ngen, nsoln, :).OptimvalIter = Pop_LSQ.Optimval;
    end
    %----------------------
    save(GA.FileSav, 'State_HYBRID', 'ngen','nsoln','-append');
%=====================================================
case{'fminunc'}
        %NOT PREFER to solve sum-of-square problem => lsqnonlin
        %LargeScale:  must provide the gradient
        %Medium-scale: Quasi-Newton line search (Algorithm selected)
      %Warning: Gradient must be provided for trust-region method
        %       =>  using line-search method instead.
        %Optimization terminated:
        %       relative inf-norm of gradient less than options.TolFun.
    [x_HYBRID, HYBRID_F2, exitflag, HYBRID_output] = fminunc(@(x)...
        Buss_XR4_Iter( x, HYB.Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
        x_GA, HYB.Optim_Options)
%=====================================================
   case{'fminsearch'}
        %NOT PREFER to solve sum-of-square problem => lsqnonlin
        %only minimiz over real, f(x) must return real
        %COMPLEX VARIABLES must be split into R+jI
     %robust for discontin problem, less efficient than fminunc ord >2
    [x_HYBRID, HYBRID_F2, exitflag, HYBRID_output] = fminsearch(@(x)...
```

```
        Buss_XR4_Iter( x, HYB.Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
        x_GA, HYB.Optim_Options)
%=====================================================
end
end
end
```

C.3 Example Usage: PA Model ID

C.3.1 Main Script: Part 1

```
[PATH, PlotOpt] = Matlab_Settings('Athlon64X2')
%-----------------------------------------------
Opt.N = 5;
    Opt.Nth_HD2band     = 'ZHD2_Flat';% 'No' 'ZHD2_NO_REIO' 'ZHD2_REIO'
Opt.ZBre_ZHD2ie     = 'ZBreioZHD2roie';% 'ZBreZHD2ie'
    Opt.DataType        = 'LSNA_RFMD';% 'VSA'
    Opt.Optim_WtConst = 10;% 100 10 2
%-----------------------------------------------
FileNam_iNL = ['Buss_iNL_' num2str(Opt.N) '_Local.mat'];%_Globe
    temp = userpath;        temp = temp(1:end-1);
FileSav_iNL = fullfile(temp, FileNam_iNL)
%-----------------------------------------------
    Nord    = 9;
    Nfreq   = 3;
    Ntone   = 2;
    Buss_Sym% (Nord, Nfreq, Ntone, FileSav_iNL)
    Q_cell = {  1 [-f1 f1]        [f1]; ...
                2   [-f3 -f2 f2 f3] [f2 f3 2*f2-f3 2*f3-f2]};
    [cSIZE c cFCT Q_cell Qmix] = Buss_2_iNL( Opt.N, Q_cell, ...
                    s_SymVEC, FileSav_iNL);
%-----------------------------------------------
Opt.NumHD = size(Qmix,2)-1
HdBand      = ones(1, Opt.NumHD+1);
switch Opt.Nth_HD2band
```

```
    case{'No'}
        HdBand(1:2) = 0 %[ENV-band HD1-band]
    otherwise
        HdBand(1:3) = 0 %[ENV-band HD1-band HD2-band]
end

%% ######################################################## H=Zmat*(I=iNL)
syms Z% Zeq = sym('Z(s)');      % dummy abstr func
z_SymVEC = [Z];
syms V
Vvec = [V];
syms I
Ivec = [I];
%---------------------- Lin-Ckt-Source
Ymat = [ 1/Z ]
%---------------------- Lin-Txfr(s) for I{1}~=[1], control HiOrd iNL
ILin = [1];
Zmat        = factor( Ymat \ ILin );
%---------------------- Lin-Resp
VLin = subs(Zmat, s, s_SymVEC(1));% Zmat .* ILin



%% ######################################################## Meas Data
Opt.Atten_dB = 20;
switch Opt.DataType
    %========================== dataset VSA
    case{'VSA'}
```

```
        Opt.MaxPin_IDX  = 5;% 5~1.5dB 6~3.5dB
        [Q_RespMTONE VSA] = VSA_QRespMtoneLoad_Func( Opt, ...
    PATH, 'SD513301', ...
    'SD513301_2000Mhz_-28dBm12dBmAtt0_100hz_401pts10ms_LSNA.s2p', ...
    'SD513301_VSA_Arb2_Test.mat', ...
    PlotOpt.df_dBRef, Opt.Atten_dB);
    %========================== dataset LSNA
    case{'LSNA_RFMD'}
        Opt.MaxPin_IDX = 12;% 12~1dB 15~2.1dB
        [Q_RespMTONE LSNA] = VSA_QRespMtoneLoad_Func( Opt, ...
    PATH, 'SD513301', ...
    'SD513301_2000Mhz_-28dBm12dBmAtt0_100hz_401pts10ms_LSNA.s2p', ...
    'SD513301_LSNA_SWP_Corr.mat', ...
    PlotOpt.df_dBRef, Opt.Atten_dB);
    %==========================
end

%% ######################################################## MTONE
[   Qmix zSymQMIX VvecQMIX    zSymQMIX_ENV zSymCEL ...
fQMIX fxQMIX   fxVEC       fCEL] = Buss_zvQMIX_Func( ...
    z_SymVEC, Vvec, ...
    HdBand, Q_RespMTONE, ...
   Qmix, Q_cell, ...
    0);% (, Opt_Disp)
%=============================================
aSymCEL         = SYMvec2SYMcel(a_SymVEC(1:Opt.N));% [ aSymCEL{:} ]
%---------------------
```

```
InputCharVEC    = 'xy';
InputSymVEC     = CHARvec2SYMvec(InputCharVEC);
InputSymCEL     = {InputSymVEC};% SYMvec2SYMcel(InputSymVEC);


Opt_Disp = 0;
for nQ = 1 : size(Q_cell,1)%nQ = 2;
Q    = Q_cell{nQ,1};
  fVEC = Q_cell{nQ,2};
        f_SymVEC = fVEC(length(fVEC)/2+1 : end);
    if(Opt_Disp)
        disp(['%##################################### nQ=' int2str(nQ)])
    end
for nRESP = 1 : length(Q_cell{nQ,3})%nRESP = 3
   fRESP   = Q_cell{nQ,3}(nRESP);
    Fmix    = Q_cell{nQ,4}{1,nRESP};
    Fidx    = Q_cell{nQ,4}{2,nRESP};
    cFidx   = Q_cell{nQ,4}{3,nRESP};
    if(Opt_Disp)
        disp([ '%########################### nRESP=' num2str([nRESP]) ])
        %fRESP   = Q_cell{nQ,3}(nRESP)
        %Fmix
    end
%===========================================
tic
switch Opt.DataType
    %-------------------------
    case{'LSNA_RFMD', 'VSA'}% Core1: 67sec MOST TIME CONSUME
```

```
    [Hv     Hv_SymCHAR Hv_SymCEL Hv_SymCEL_IDX] = Buss_HvCell4_Func( ...
        Qmix, zSymQMIX, Zmat, ...
        z_SymVEC, a_SymVEC, InputSymVEC, f_SymVEC, ...% VvecQMIX,
     zSymCEL, aSymCEL, InputSymCEL, ...
        Opt.N, cSIZE, c, cFCT, ...
        Fmix, Fidx, cFidx, ...
        0);
    Q_cell{nQ,4}{4,nRESP}=Hv;           %{n}[nV,1]
    Q_cell{nQ,4}{5,nRESP}=Hv_SymCHAR;   %{nV,n}[Char]
    Q_cell{nQ,4}{6,nRESP}=Hv_SymCEL;    %{nV,n}{nSYM}
    Q_cell{nQ,4}{7,nRESP}=Hv_SymCEL_IDX;%{nV,n}{nCEL [nSYM;IDX nSYM;IDX]}
    Q_cell{nQ,4}{8,nRESP}={};          % {nV,n} NumFreqPts
    %-------------------------
    case{'CDMA'}
    %-------------------------
end
toc
%=============================================
end
end
```

C.3.2 Main Script: Part 2

```
%% ######################################################## InitGuess
Opt.Third_Atmpt = 21;
Opt.Third_nPwr = 1;
    figVEC = [  600 601     0   603 ... %yModel (EVEN,ODD,RI,dB)
                610 611          ...    %ZB    (RI,MA)
                0   621         ...     %Z_Im3 (RI,dB)
                630 631     632 633];   %Z_HD2 (EVEN,ODD,RI,dB)
switch Opt.DataType
    %------------------------- dataset VSA
    case{'VSA'}
        [Opt.x0     Opt.x] = Buss_XR4_N3Estim_NLSQ_3b( Opt, ...
            VSA, fQMIX, fxQMIX, ...% Q_RespMTONE,
            Qmix, Q_cell, ...
            figVEC, PlotOpt, 1);
    %------------------------- dataset LSNA
    case{'LSNA_RFMD'}
        [Opt.x0     Opt.x] = Buss_XR4_N3Estim_NLSQ_3b( Opt, ...
            LSNA, fQMIX, fxQMIX, ...% Q_RespMTONE,
            Qmix, Q_cell, ...
            figVEC, PlotOpt, 1);
%-------------------------
end
    % Opt = rmfield(Opt, 'x_');
    Opt.x0
    Opt.x% close all
%==============================================
```

```
n_start = 5;      n_step = 2;     numpts = 4;
    Opt.PS.ordVEC = [n_start : n_step : (n_start+(numpts-1)*n_step)];
    Opt.PS.ordVEC
[Opt.PS.a3 Opt.PS.b3] = PolyFit_AMPM2( Q_RespMTONE{1,1}{1}, ...
    Vp02dBm_dB( Q_RespMTONE{1,2}{2} ), ...
    Opt.PS.ordVEC, [1:7], 'AMPM_dBm', 'Odd', 1, ...
    '.', PlotOpt);
Opt.PS.a3( 1:2, 2:8 )%DEBUG

%% ######################################################
Core  = '1';% 1 2 8 Grendel
switch Opt.Nth_HD2band
    case{'No'}
        FileNam_Optim = ['matlab_core' Core '_N' num2str(Opt.N) ...
            '_HD2' Opt.Nth_HD2band]
    otherwise
        FileNam_Optim = ['matlab_core' Core '_N' num2str(Opt.N) ...
            '_HD2band']
end
    FileSav_Optim = fullfile(PATH.MatlabSave, [FileNam_Optim '.mat'])
    whos('-file', FileSav_Optim);
    % save( FileSav_Optim )
    % load( FileSav_Optim )

%------------------------------------------------- clear LSQ
    LSQ.MaxIter         = 100;
LSQ.PrecondBandWidth = inf;% Inf-ChoskyFactor Better than 0=ConjGrad;
```

```
    FMINsear.MaxIter   = 100;
%-------------------------------------------------
    Opt.Qstart     = 2;
    Opt.Optim_Wt    = 'WtConst';% 'WtConst' 'WtPin' 'WtPinDf'
        % 'WtConst':    [10 10 1 1] Better Match S21
        % 'WtPin':      Reduce Accur at HI-PWR
    Opt.Optim_Err = 'DiffRel_RI';% 'DiffRel' 'Diff' 'RatLog_RI'
    Opt.Optim_Err2 = 'RI';% 'RI'='RIAbs' 'RIAbsSqr'=worst 'Abs'='AbsSqr'
        %Opt.Optim_Err2 = 'Abs';        46.7606 +89.8401i
        %Opt.Optim_Err2 = 'AbsSqr';     46.7606 +89.8401i
        %Opt.Optim_Err2 = 'RIAbsSqr';   117.13  +222.48i
    Opt.Optim_MtoneFit  = 'Pout';% 'S21H3';%
    %-------------------------
    Opt.Nth_Atmpt = {'a1Magb1'    'g2Mag'   'Zf'    'Real'};

%%######################################################## ITER1
    LSQ.TolFun        = 0.001;% 1e-6;%
    FMINsear.TolFun    = 1;
    %-------------------------
Opt.Nth_HD2band     = 'ZHD2_Flat';% whats updated
Opt.ZBre_ZHD2ie     = 'ZBreZHD2ie';%
    Opt.Optim_Meth      = 'lsqnonlin';% 'ga';% 'fminsearch';%
    %-------------------------
    FileNam_LocSear     = [Opt.DataType 'LSQ' '_N' num2str(Opt.N) ...
        '-PinIdx' num2str(Opt.MaxPin_IDX) '-' Opt.Optim_MtoneFit ...
        '_' Opt.Optim_Wt num2str(Opt.Optim_WtConst)  ...
        '_' Opt.Optim_Err '-' Opt.Optim_Err2 ...
```

```
        '_' cell2mat(Opt.Nth_Atmpt) '-' Opt.Nth_HD2band]
    FileSav_LocSear         = fullfile(PATH.MatlabSave, ...
        [FileNam_LocSear '.mat']);
        whos('-file', FileSav_LocSear);
    %-------------------------
Opt.x   = rmfield( Opt.x, 'gN');
Opt.x   = rmfield( Opt.x, 'ZHDN');
%-------------------------
    ITER1.Opt       = Opt;
[ITER1.Opt]     = Buss_XR4_Param2XReal_3b( ITER1.Opt, ...
        ITER1.Opt.x, ITER1.Opt);
[b1     InputNumCEL     aNumCEL     zxNumQMIX   ITER1.Opt.y0] = ...
    Buss_XR4_XReal2zxNumQMIX_3b( ITER1.Opt.xReal_0, ...
            ITER1.Opt, InputSymCEL, 0);
        %ITER1.Opt.x
        %ITER1.Opt.y0
    [Q_RespMTONE Q_cell]    = Buss_XR4_Init( ITER1.Opt.xReal_0, ...
        ITER1.Opt, Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
        Qmix, Q_cell, 1);
    [F2_0           Q_RespMTONE] = Buss_XR4_Iter( ITER1.Opt.xReal_0, ...
        ITER1.Opt, Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
        Qmix, Q_cell, 0);
%===========================================
[ITER1] = Buss_3_GA( FileNam_LocSear, FileSav_LocSear, ITER1, ...
    LSQ, FMINsear, [], ...
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell);
```

```
[b1     InputNumCEL     aNumCEL     zxNumQMIX   ITER1.Opt.y]  = ...
    Buss_XR4_XReal2zxNumQMIX_3b( ITER1.xReal, ITER1.Opt, InputSymCEL,0);
        ITER1.Opt.y
save(FileSav_LocSear, 'ITER1', 'FileNam_LocSear', ...
    'FileSav_LocSear', '-append');
    %f1f2 dBErr:    40  / 10 /(10*12)   = 0.0333
    %im3 dBErr:     100 / 1  /(10*12)   = 0.8333
    %VSA    PinIdx=5    Wt=100 Err = [130 126] RunTime_LSQ = [305 518]
    %LSNA   PinIdx=15   Wt=100 Err = [366 ]    RunTime_LSQ = [232 ]
    %LSNA   PinIdx=12   Wt=10  Err = [23]      RunTime_LSQ = [550 ]

%%######################################################## ITER2
    LSQ.TolFun        = 0.001;
    FMINsear.TolFun    = 0.001;
    %-------------------------
Opt2 = ITER1.Opt;
    Opt2.Nth_HD2band = 'ZHD2_REIO';% whats updated
Opt2.ZBre_ZHD2ie    = 'ZBreioZHD2roie';
    Opt2.Optim_Meth   = 'lsqnonlin';% 'ga';% 'fminsearch';%
%-------------------------
    ITER2.Opt           = Opt2;
[ITER2.Opt]         = Buss_XR4_Param2XReal_3b( ITER2.Opt, ...
                            ITER1.Opt.y, ITER1.Opt);
[b1     InputNumCEL     aNumCEL     zxNumQMIX   ITER2.Opt.y0] = ...
    Buss_XR4_XReal2zxNumQMIX_3b( ITER2.Opt.xReal_0,...
        ITER2.Opt, InputSymCEL, 0);
        %ITER2.Opt.x
```

```
        %ITER2.Opt.y0
    [Q_RespMTONE Q_cell]         = Buss_XR4_Init( ITER2.Opt.xReal_0,...
        ITER2.Opt, ...
        Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
        Qmix, Q_cell, 1);
    [F2_0           Q_RespMTONE] = Buss_XR4_Iter( ITER2.Opt.xReal_0,...
        ITER2.Opt, ...
        Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
        Qmix, Q_cell, 0);
%=============================================
[ITER2] = Buss_3_GA( FileNam_LocSear, FileSav_LocSear, ITER2, ...
    LSQ, FMINsear, [], ...
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell);
[b1     InputNumCEL     aNumCEL     zxNumQMIX   ITER2.Opt.y]  = ...
    Buss_XR4_XReal2zxNumQMIX_3b( ITER2.xReal, ITER2.Opt, InputSymCEL, 0);
        ITER2.Opt.y
save(FileSav_LocSear, 'ITER2', '-append');
```