EFFICIENT DIGITAL BASEBAND PREDISTORTION FOR MODERN WIRELESS HANDSETS

A Thesis
Presented to
The Academic Faculty

by

Seydou Nourou Ba

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
December 2009

EFFICIENT DIGITAL BASEBAND PREDISTORTION FOR MODERN WIRELESS HANDSETS

For my family.

ACKNOWLEDGEMENTS

As my PhD endeavor comes to an end, I would like to express my gratitude to the people who greatly contributed to its success.

First, I would like to thank my advisors, Dr. Yucel Altunbasak and Dr. G. Tong Zhou for their continuous support and guidance throughout my PhD studies. I have greatly benefited from their valuable advices and encouragements. I will certainly continue to benefit from my experience in the Center for Signal and Image Processing in my future career.

I would like to thank my thesis committee members: Dr. Xiaoli Ma, Dr. Ronghua Pan, Dr. James Stevenson Kenney, and Dr. Ghassan Al-Regib for accepting to serve on my committee and helping to greatly improve the quality of my dissertation with their thoughtful suggestions.

I would also like to thank Dr. Khurram Waheed, Dr. Fernando Mujica, Dr. Imtinan Elahi and Dr. Darnell Moore from Texas Instruments, whose fruitful collaboration has been essential to the successful completion of my thesis.

I also tremendously benefited from my teaching assistant experience under the supervision of Dr. James H. McClellan.

I would like to thank my fellow PhD students: Ibrahima Ndiour, Mamadou Diao, Arnaud Amadjikpe, Illenin Kondo, Salman Aslam, Bob Baxley, Rajbabu Velmurugan, Tariq Arici, Brett Matthews, Milind Borkar, and Ali C. Begen for their friendship and insightful discussions.

I would also like to thank the ECE graduate academic advisors. My special thanks go to Marilou Mycko and Dr. David Hertling for offering me my first teaching assistant position.

And last but not least, I thank my family for their unconditional support, their encouragements, and their prayers during difficult times. A lot of credit goes to my wife Oumou, who inspired me and helped me complete this thesis with her kindness and patience.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

One of the major challenges in modern wireless transceiver design stems from the fundamental trade-off between the linearity and power efficiency of RF amplifiers. On one hand, the varying envelope of spectrally-efficient modulated signals such as wideband code division multiple access (WCDMA) and orthogonal frequency division multiplexing (OFDM) interacts with the amplifier nonlinearity, resulting in both in-band distortion and undesired out-of-band spectral regrowth. These distortions cause the violation of the strict standard requirements. On the other hand, improving the inherently low power efficiency of amplifiers generates significant savings in cooling and running costs at the base station infrastructure and enables an equally important increase in mobile handsets battery life.

Digital baseband predistortion is one of the most effective techniques used to reconcile the conflicting requirements of power efficiency and increased data throughput per unit bandwidth. The accuracy and flexibility of digital predistortion allows the use of a highly nonlinear amplifier to increase the overall power efficiency while meeting the strict performance requirements.

This dissertation studies the design of an efficient adaptive digital baseband predistorter for modern cellular handsets that combines low power consumption, low implementation complexity, and high performance. The proposed enhancements are optimized for hardware implementation.

We first present a thorough study of the optimal spacing of linearly-interpolated lookup tables supported by theoretical calculations as well as extensive simulation experiments. A constant-SNR compander that increases the LUT predistorter's supported input dynamic range is derived. A corresponding low-complexity approximation that lends itself to efficient hardware design is also implemented in VHDL and synthesized with the Synopsys Design Compiler. This dissertation also proposes an LMS-based predistorter adaptation that is optimized for hardware implementation and compares the effectiveness of the direct and indirect learning architectures.

Analog RF imperfections such as quadrature imbalances and varying antenna impedance during the device operation severely reduce the effectiveness of adaptive predistorters. A novel predistorter design with quadrature imbalance correction capability is developed and a corresponding adaptation scheme is proposed. This robust predistorter configuration is designed by combining linearization and I/Q imbalance correction into a single function with the same computational complexity as the widespread complex-gain predistorter. An adaptive gain and phase normalization technique that reuses the predistorter update hardware is also proposed to mitigate the effects of varying antenna matching condition during predistorter adaptation.

CHAPTER I

INTRODUCTION

1.1 Motivation

In modern wireless communication systems, the power added efficiency (PAE) of the underlying power amplifier (PA) is of paramount importance. On the base station side, a slight improvement in power amplifier efficiency translates into considerable savings in cooling bills and running costs for the network operator. On the other side of the transmission link, increased battery life is the main driver for improving the efficiency of the power amplifier in the mobile station.

In the current state of the art amplifier design methodology, there is a fundamental trade-off between power efficiency and device linearity. The power efficiency is improved at the cost of undesired envelope-dependent nonlinear distortions to the transmitted signal. For a long time, this has favored the design and extensive use of robust modulation techniques such as frequency shift keying (FSK) and Gaussian minimum shift keying (GMSK), which are immune to nonlinear amplifier distortions.

The strong demand in increased transmission rates has led to the development of modulation schemes with higher spectral efficiency such as orthogonal frequency division multiplexing (OFDM) and wideband code division multiple access (WCDMA). The resulting baseband signals have varying complex envelopes with high peak-to-average power ratios (PAPR), which further worsens the efficiency-linearity trade-off. The amplifier nonlinearity results in in-band distortion measured by the deterioration of the error vector magnitude (EVM) or the bit error rate (BER) and undesired out-of-band spectral regrowth [41, 52], which leads to the violation of the strict standard spectral requirements.

To reconcile the conflicting requirements of power efficiency and increased data throughput per unit bandwidth, it is necessary to linearize the radio frequency power amplifier.

Numerous analog linearization techniques have been proposed and most of them were extensively used over the years. These analog techniques can generally be classified into one of three different categories:

- Indirect feedback
- Feedforward linearization
- Predistortion linearization

Two widely used indirect feedback techniques are the polar and Cartesian corrections. The complex input and output envelopes are compared to generate a correcting function, which is subsequently applied to the input signal envelope. These techniques typically achieve moderate performance improvement and are subject to bandwidth and stability issues.

The feedforward method is a powerful linearization technique that is immune to the bandwidth and stability issues of the above mentioned indirect feedback techniques. The feedforward correction is directly applied to the output RF signal. The high power consumption and overall cost of the feedforward technique constitute major limitations.

Simple analog predistortion techniques have been successfully used to provide moderate correction with little additional power consumption and at a very low cost. But their inability to track the drifts of amplifier characteristics and provide correction over a large dynamic range considerably limits their effectiveness.

The development of faster low-cost digital signal processors (DSP) has favored the emergence of adaptive digital predistortion linearization techniques. These techniques combine the computational power of DSP processors and advanced signal processing algorithms to provide accurate correction with the capability to precisely track both the short-term variations of the amplifier characteristics, which are caused by temperature fluctuations and the long-term drift effects resulting from the aging of analog devices.

Furthermore, the recent appearance of digital cellular transmitters with embedded processors has increased the appeal of digital baseband predistortion, which benefits from the low cost of implementation and high flexibility offered by digital circuit design. Consequently, digital predistortion is receiving increased attention and many digital baseband linearizers of varying complexity and performance have been proposed in the recent literature.

Even though the general principles of digital predistortion are the same across all applications, each design must be optimized with respect to the system under consideration. For example, a simple memoryless predistorter would be ineffective in the case of high power amplifiers (HPA) used in cellular base-stations (BS), which produce strong memory effects. A more complex predistorter structure with memory effect mitigation capability would be required. The Hammerstein and memory polynomial predistorters are two examples of such systems. On the other hand, memory effects are often negligible in low-power cellular handset amplifiers. For these resource-constrained devices, the focus is on minimizing the computational complexity of the predistorter hardware to produce low-cost handsets with increased battery life.

1.2 Objectives

The objective of this dissertation is to develop an efficient digital baseband predistorter for modern cellular handsets that combines low power consumption, low implementation complexity, and high performance. The high peak-to-average power ratio and large signal bandwidth of modern modulation techniques (OFDM, WCDMA) represent significant challenges to the design of an effective digital predistorter for mobile devices with limited resources. Our research efforts focus on three main areas:

- Develop lookup table predistorter enhancement techniques
- Design an adaptive predistorter optimized for efficient hardware implementation
- Mitigate RF impairments affecting the effectiveness of an adaptive predistorter

Amplifiers used in cellular handsets are operated at a relatively low power (e.g. typical maximum of 1 W of power). At such a low output power and for the signal bandwidths considered, memory effects are negligible. Therefore, our research will focus on memoryless predistorter design, even tough most of the proposed design optimizations can be extended to predistorters with memory correction capability. Additionally, as signal bandwidths continue to increase with 4G standards (e.g. up to 20 MHz for LTE), memory predistorters might be required even for low-power handsets in the near future.

1.3 Outline

This dissertation is organized as follows.

Chapter 2 reviews the principles of digital predistortion and presents different predistorter configurations. The characterization of nonlinear amplifier characteristics is illustrated with the AM-AM and AM-PM responses of different types of amplifiers. The main predistorter configurations proposed in the literature such as the Cartesian mapping predistorter [39], the complex-gain predistorter [8], and the polar predistorter [21] are presented. Predistorter training and adaptation techniques are also reviewed.

Chapter 3 and Chapter 4 present a study of optimal spacing of linearly-interpolated lookup tables (LUT) for the polar and complex-gain predistorters, respectively. Previous studies [11] have shown that optimal LUT spacing has little effect on non-interpolated LUTs. This chapter theoretically and experimentally demonstrates that the combination of linear interpolation with optimal spacing can greatly enhance the performance of LUT predistorters and decrease their cost of implementation by reducing the required memory space.

Chapter 5 presents a constant-SNR compander that increases the LUT predistorter's input dynamic range. A corresponding low-complexity approximation that lends itself to efficient hardware implementation is also presented. This approximation is implemented in VHDL and synthesized with the SYNOPSYS DESIGN COMPILER.

Chapter 6 analyzes the performance of power and amplitude LUT indexing when input signal backoff is supported. Traditionally, the use of power indexing has been favored in digital predistorter design because of the convenient computation of the instantaneous power ($I^2 + Q^2$) as opposed to using a suitably-accurate amplitude approximation. It is shown that the amplitude indexing results in better performance for highly nonlinear amplifiers. An efficient amplitude approximation is proposed and its hardware implementation is illustrated.

Chapter 7 proposes an LMS-based predistorter update that is optimized for hardware implementation. A comparison of the direct and indirect learning architectures is also presented.

Chapter 8 introduces a robust predistorter configuration that also mitigates quadrature gain and phase imbalances. The proposed predistorter has the same computational complexity as the complex-gain predistorter, but is more effective in the presence of analog impairments such as I/Q imbalances. A simple and efficient LMS-based adaptation is also proposed for this predistorter.

Chapter 9 studies the effects of varying impedance matching conditions on the predistorter adaptation during real-life device operation. An adaptive gain and phase normalization that reuses the predistorter update hardware is proposed to mitigate these effects.

Finally, Chapter 10 summarizes the contributions in this dissertation and suggests future research directions.

CHAPTER II

BACKGROUND

In this chapter, an overview of the effects of amplifier nonlinearity is presented. The general concepts of predistortion linearization, the different prior-art predistorter configurations and their corresponding training algorithms are also reviewed.

2.1 Power Amplifier Nonlinearity

Class-A power amplifiers represent the family of most linear and well-behaved power amplifiers. But they can only achieve a maximum theoretical power efficiency of 50%. The transfer characteristic of a class-A amplifier is linear at low amplitude levels and is compressed as the saturation level is approached. An important reference is the 1 dB compression point(P_{1dB}), which represents the output power level at which the PA gain is compressed by 1 dB. An amplitude modulated signal can be successfully transmitted through a suitably backed-off class-A power amplifier without suffering major distortions. The amount of back-off needed is proportional to the peak-to-average power ratio (PAPR) of the input signal. But in this configuration, the inherently low power efficiency of class-A amplifiers is further degraded by the input signal back-off. In the case of high spectral efficiency modulation techniques such as OFDM and WCDMA, which often have PAPRs of more than 10 dB, the resulting power efficiency may be well below 10%.

A sharp increase in efficiency can be achieved by lowering the quiescent bias level of the amplifier. This consequently results in a reduction of the conduction angle and the appearance of high-order harmonics, which can be filtered by a carefully designed matching network. The maximum efficiency of an amplifier is given as function of the conduction angle by [30]

$$\eta = \frac{2\gamma - \sin 2\gamma}{4(\sin\gamma - \gamma\cos\gamma)}, \tag{2.1}$$

where γ is the conduction angle. This relationship between efficiency and conduction angle is illustrated in Figure 2.1. A class-A amplifier has a conduction angle of $\gamma = 2\pi$ and a maximum power efficiency of $\eta = 50\%$. Class-B operation corresponds to a conduction angle $\gamma = \pi$ with a maximum efficiency of $\eta = 78.5\%$. When $2\pi < \gamma < \pi$, the amplifier is in class-AB operation. A conduction angle of $\gamma < \pi$ corresponds to a class-C amplifier. The power efficiency increases dramatically when the conduction angle is biased towards class-C operation. But this huge gain in efficiency comes at the cost of severe degradation of the amplifier linearity, as illustrated by the decrease in signal to noise and distortion ratio (SNDR).

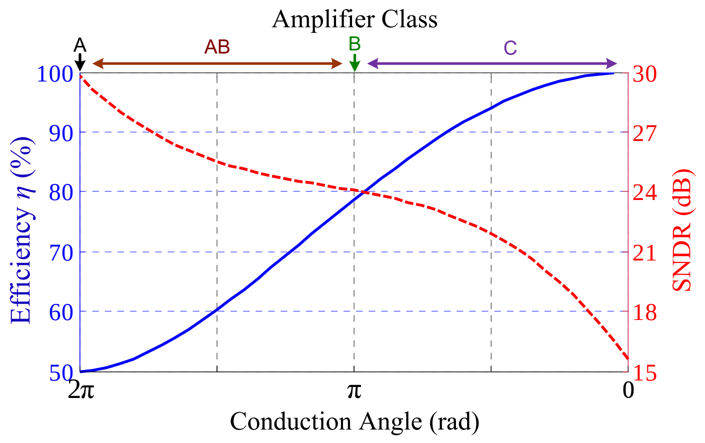


Figure 2.1: Power efficiency and SNDR vs. conduction angle.

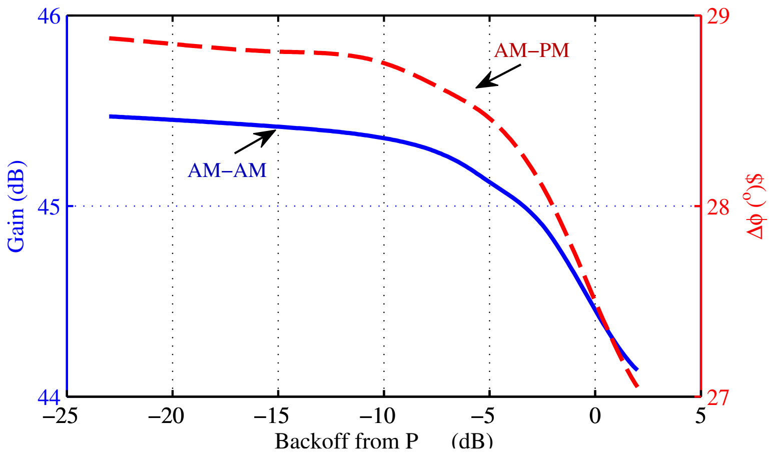
Low-power amplifiers exhibit a *memoryless* type of nonlinearity. This type of nonlinearity causes a static, envelope-dependent amplitude distortion known as AM-AM conversion. A less intuitive but equally destructive type of distortion is the envelope-dependent phase distortion or AM-PM, which is observed in amplifiers with very short memory. The latter type of devices are also known as *quasi-memoryless* amplifiers. Let us consider an amplitude and phase modulated carrier:

$$S(t) = A(t)\cos[w_c t + \theta(t)]\,, \tag{2.2}$$

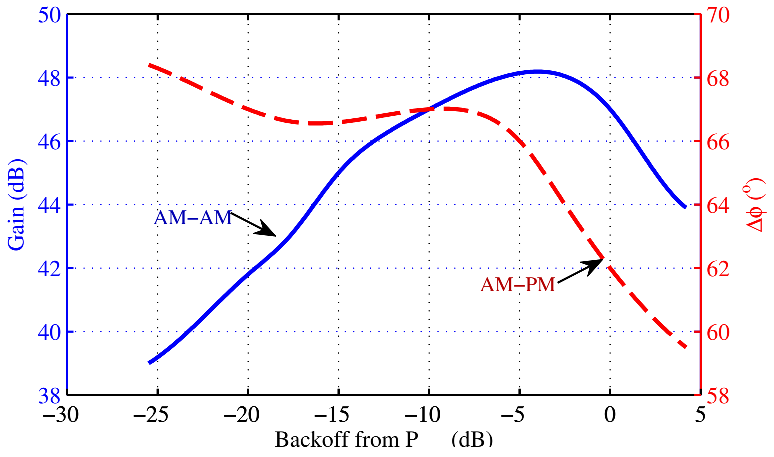
where $A(t)$ represents the amplitude modulation and $\theta(t)$ the phase modulation. The corresponding distorted output of a nonlinear amplifier is then given by

$$D(t) = f[A(t)]\cos\{w_c t + \theta(t) + g[A(t)]\}\,, \tag{2.3}$$

where $f(.)$ and $g(.)$ respectively represent the AM-AM and AM-PM conversion functions. Figure 2.2 compares the AM-AM and AM-PM of typical class-A and class-C amplifiers.



(a)



(b)

Figure 2.2: AM-AM and AM-PM conversion functions. (a) Class-A amplifier. (b) Class-C amplifier.

The class-A amplifier has a maximum gain variation of 1.4 dB and a maximum phase variation of less than 2° across a 25 dB power range. The class-C characteristic is subject to significantly stronger distortions with a maximum gain variation of 9 dB and a maximum phase variation of 8° across the same power ranger. Unlike the relatively smooth compressed gain characteristic of the class-A amplifier, its class-C counterpart displays severe amplitude and phase distortions across all amplitude levels. Therefore, backing off a class-C amplifier will only reduce its effective power efficiency while doing little to reduce the effects of nonlinear distortions.

High-power amplifiers (HPA) such as those used in cellular base stations exhibit a nonlinear behavior that is coupled with strong memory effects [32, 48], especially in the case of wideband signals. This type of nonlinearity causes complex distortions that depend on the current as well as past amplitude levels. The memory effects can be observed on the input-output characteristics, as illustrated in Figure 2.3.

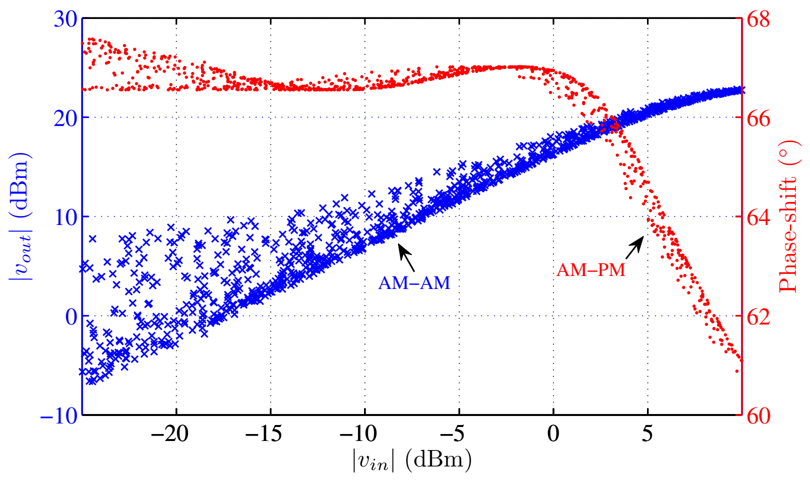


Figure 2.3: Amplifier nonlinearity with memory effects.

Two physical sources of PA memory effects are described in [48]. Electrical memory effects are due to the variations of complex envelope impedance across the signal's frequency band. The electro-thermal memory effects are caused by signal-dependent variations of the thermal impedance through the process of thermal power feedback (TPF). The transfer characteristics of amplifiers with memory effects can be accurately modeled with Volterra series [44]. The Wiener model [13], which is a special case of the Volterra system, consists of a linear time-invariant (LTI) system cascaded with a memoryless nonlinearity. Another special case of the Volterra series is the memory polynomial described in [31].

A widely adopted method of assessing an amplifier's linearity (or lack thereof) is the *two-tone* test. It consists of feeding the amplifier with a test signal that is constructed by summing two closely-spaced unmodulated RF carriers:

$$v_{in}(t) = A\cos(2\pi f_1 t) + A\cos(2\pi f_2 t). \tag{2.4}$$

This test signal is equivalent to an amplitude modulated carrier that is free of any distortion that would otherwise be caused by a non-ideal RF modulator. To illustrate this, equation (2.4) can be rewritten as follows:

$$v_{in}(t) = 2A\cos(2\pi f_m t) \times A\cos(2\pi f_c t), \tag{2.5}$$

with $f_m = \frac{f_2 - f_1}{2}$ and $f_c = \frac{f_1 + f_2}{2}$. Let us consider an amplifier with an amplitude conversion function (AM-AM) modeled by a polynomial function of arbitrary order N. The output is then given by

$$v_{out}(t) = a_1 v_{in}(t) + a_2 v_{in}(t)^2 + \cdots + a_k v_{in}(t)^k + \cdots + k_N v_{in}(t)^N. \tag{2.6}$$

When the two-tone signal is applied to the input of this amplifier, each order of nonlinearity k will generate additional frequency components or intermodulation products of the form

$$f_{im} = mf_1 + nf_2, \tag{2.7}$$

where m and n are positive integers and $m+n = k$. The even orders of nonlinearity generate intermodulation (IMD) products that are located far away from the input frequencies. These IMD product terms are less important since they can be easily filtered. On the other hand, the odd-order terms generate IMD products that lie in the frequency band of interest. Figure 2.4 illustrates the in-band IMD products caused by the third, fifth, and seventh orders of nonlinearity. It is analytically shown in [14, p. 203] that the AM-PM effect results in the appearance of similar intermodulation products in the frequency band of interest.

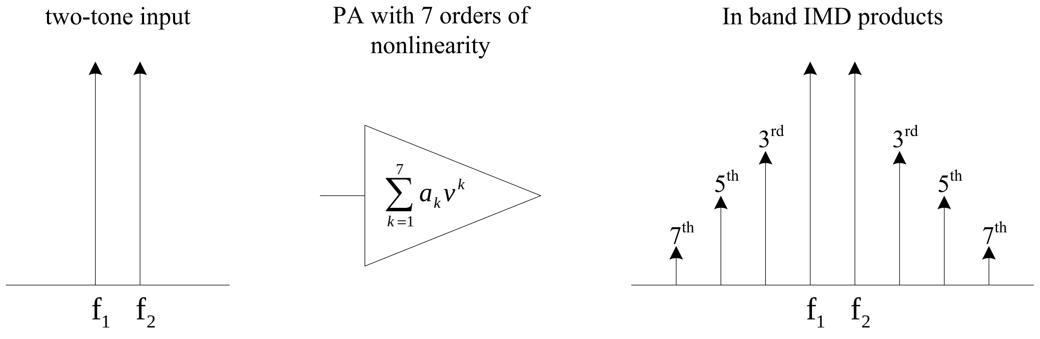


Figure 2.4: In-band IMD products for an amplifier with up to seven orders of nonlinearity.

Over the years, the two-tone test has proven to be a powerful PA linearity assessment procedure that can be performed with a relatively simple experimental setup. However, modern modulated signals are far more complex than the simple two-tone test signals. It is therefore often necessary to perform a test with the modulated signal of interest. As a general rule of thumb, the kth order of PA nonlinearity causes the appearance of a parasitic component occupying k times the bandwidth of the input signal [14, p. 185]. The actual power level of this parasitic component depends on the strength of the nonlinear term in question. This is illustrated in Figure 2.5.

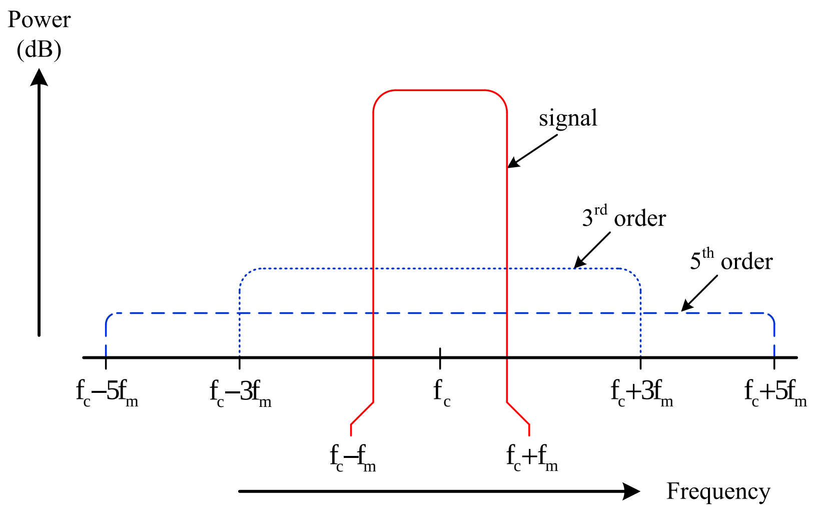


Figure 2.5: Intermodulation spectrum of a typical digitally modulated signal [14, p. 185].

It can be clearly seen in Figure 2.5 that the PA nonlinearity results in the appearance of parasitic components that span the adjacent channels, causing the effect known as spectral regrowth. It should also be noted that a portion of the unwanted distortion lies within the frequency band of the original signal, causing in-band distortions that degrade the BER performance. The amount of spectral spreading is measured by the adjacent channel power ratio (ACPR), which is defined as the ratio of the power contained in a given bandwidth at a defined frequency offset f_0, to the power in the channel bandwidth around the center frequency. The metric used for measuring the in-band distortion is the error vector magnitude (EVM). The EVM is defined as the ratio of RMS error vector power to the RMS power of the reference signal:

$$\mathrm{EVM}_{\mathrm{RMS}} = \sqrt{\frac{\sum_{k=1}^{N} |S_i[k] - S_m[k]|^2}{\sum_{k=1}^{N} |S_i[k]|^2}}, \tag{2.8}$$

where S_i and S_m are respectively the ideally amplified and distorted complex signal samples. The EVM can be expressed in percentage (%) or decibels (dB):

$$\begin{aligned} \mathrm{EVM}_{\%} &= 100 \times \mathrm{EVM}_{\mathrm{RMS}}. \\ \mathrm{EVM}_{\mathrm{dB}} &= 10 \log_{10} \left(\mathrm{EVM}_{\mathrm{RMS}}\right). \end{aligned} \tag{2.9}$$

2.2 Principles of Digital Predistortion

The basic principle of predistortion is illustrated in Figure 2.6. It consists of inserting in the transmit path, prior to amplification, a nonlinear block with a transfer characteristic that is the inverse of the PA nonlinearity. The cascade of the two nonlinear elements will ideally result in a distortion-free and perfectly linear transmitter.

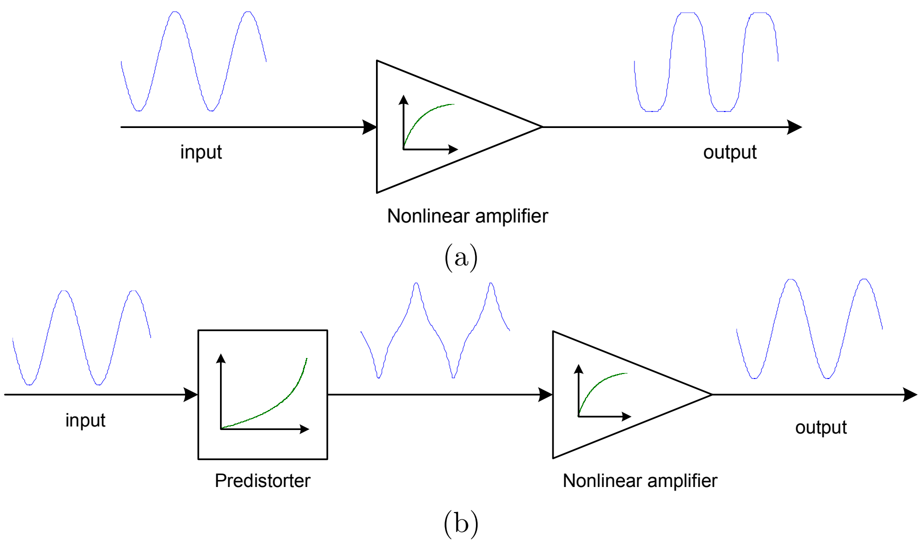


Figure 2.6: Principle of predistortion. (a) Amplifier distortion. (b) Cascade of predistorter and amplifier.

Predistortion can be applied at different points across the transmit chain. RF and IF signal predistorters are typically realized with relatively simple analog elements (such as diodes) that exhibit an expanding characteristic. The analog devices are calibrated to suppress the third order of amplitude nonlinearity, which is responsible for PA compression.

These types of predistorters are capable of only moderate correction. But their simplicity, very low cost, and low power consumption make them particularly appealing for wireless handsets. Other major disadvantages of analog RF/IF predistorters include their lack of flexibility and their inability to precisely track the variations of nonlinear characteristics.

Another type of predistortion linearizer known as the data predistorter applies the correction to the complex symbols [28,43]. The data predistorter has a very low implementation complexity since only a few complex corrections values are needed. On the downside, this predistorter is modulation dependent and requires placing the Nyquist pulse-shaping filter at the output of the amplifier. This unattractive arrangement is difficult to realize.

Predistortion can also be applied to the digital complex baseband signal after Nyquist filtering and prior to up-conversion. These digital baseband predistorters have been gaining increased popularity because of the flexibility and high accuracy provided by digital computations and their relatively low power consumption. Additionally, digital baseband predistorters further leverage the DSP processors that are embedded in modern wireless transmitters. They are also very well suited to adaptive algorithms, therefore allowing precise tracking of nonlinear characteristic variations resulting from temperature fluctuations and aging of analog devices. The basic structure of an adaptive baseband predistorter is illustrated in Figure 2.7.

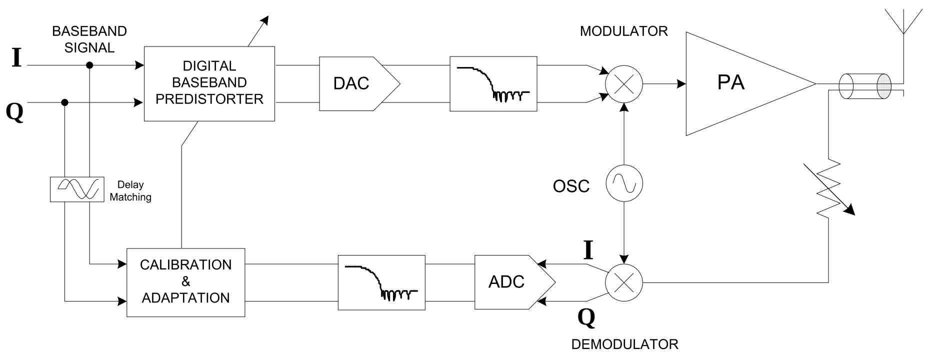


Figure 2.7: Structure of an adaptive baseband predistorter.

The adaptation process in general requires an auxiliary receiver for the feedback path. This additional hardware component accounts for most of the cost of the adaptive predistorter. Several digital predistorters have been reported in the literature. The Cartesian mapping predistorter, the polar predistorter, and the complex-gain predistorter are three of the most popular configurations.

2.2.1 Cartesian Mapping Predistorter

The Cartesian mapping predistorter was one of the early successful implementations of a digital baseband predistorter and was proposed by Nagata [39]. The structure of the mapping predistorter is illustrated in Figure 2.8.

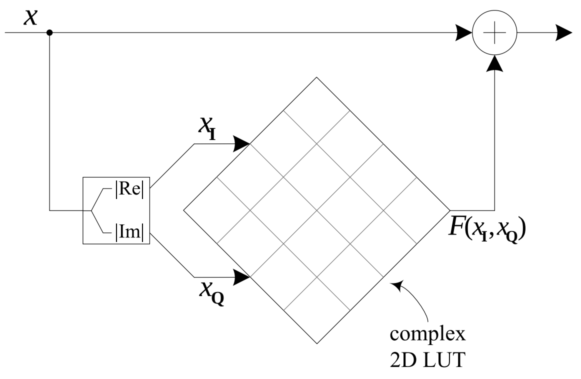


Figure 2.8: Mapping predistorter.

This predistorter is implemented as a bi-dimensional lookup table (LUT) that is addressed by the quadrature input components (I/Q). It provides an incremental complex correction to the I/Q input, thereby mapping the complex plan to itself. This linearization method has minimal computational complexity but is plagued by high memory requirements. It requires an LUT size of 2000 entries for an input signal resolution of 10 bits. This requirement increases to 8000 entries with 11 bits of input resolution [8]. A direct consequence of the large LUT size is the very slow convergence (10 s for a sampling rate of 16 kHz).

2.2.2 Complex-Gain Predistorter

The complex-gain predistorter was proposed by Cavers in [8]. The amplifier is modeled as a complex gain that is a function of the input power or amplitude. The complex envelopes of the amplifier's input v_p and its corresponding complex output v_o are then related by

$$v_o = v_p \, G\left(|v_p|^2\right). \tag{2.10}$$

The predistorter is similarly modeled and realized as a power-dependent complex gain. The predistorter's output is related to the input by

$$v_p = v_p \, F\left(|v_i|^2\right). \tag{2.11}$$

The transmitter is perfectly linear if the following condition is met:

$$F\left(|v_i|^2\right) G\left(\left|v_p F\left(|v_i|^2\right)\right|^2\right) = K, \tag{2.12}$$

where K is a constant representing the desired linear gain. An illustration of the complex-gain predistorter is shown in Figure 2.9.

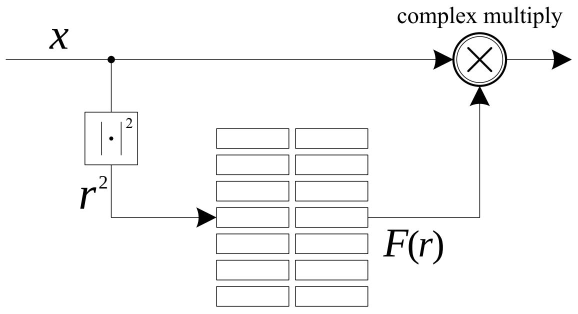


Figure 2.9: complex-gain predistorter.

In [8], the complex-gain predistorter is implemented as a lookup table that is indexed by the input power. This results in a non-uniformly spaced LUT in the amplitude domain, where entries are mostly concentrated near the saturation region. The predistorter F can also be implemented as a complex polynomial function [42]. The complex-gain predistorter's memory requirements and convergence time are four orders of magnitude lower than that of the Cartesian mapping predistorter. The cost of this improvement is the additional computations of the input power $|v_i|^2$ and the complex multiplication.

2.2.3 Polar Predistorter

The polar predistorter proposed in [21] consists of an amplitude correction function F_ρ and a phase correction F_θ that respectively compensate the power amplifier's AM-AM distortion G_ρ and AM-PM distortion G_θ. The polar predistorter exploits the fact that the PA's distortion is only a function of the input amplitude (or power). Consequently, F_ρ and F_θ

are one-dimensional functions of the input amplitude. Two equivalent versions of the polar predistorter are shown in Figure 2.10.

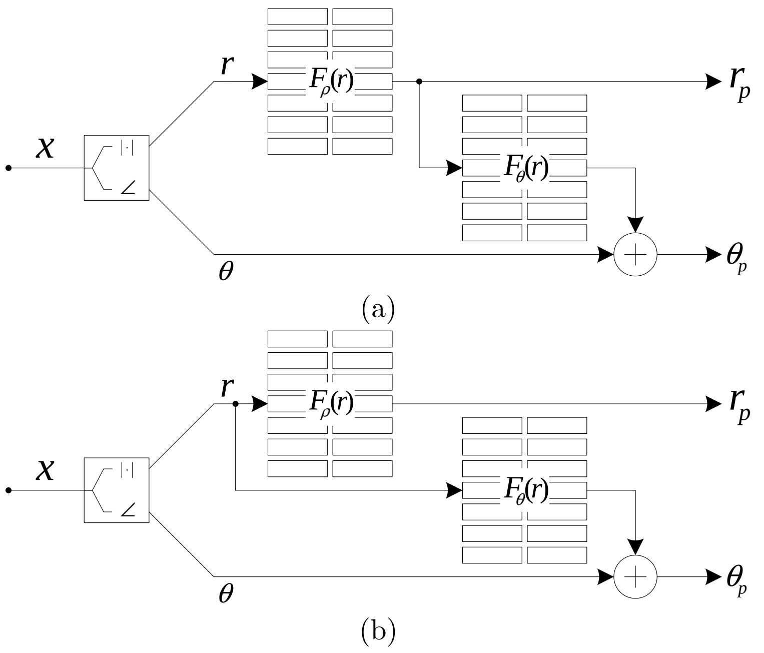


Figure 2.10: Polar predistorter configuration. (a) Cascaded AM-AM and AM-PM corrections. (b) Parallel AM-AM and AM-PM corrections.

In both cases, the AM-AM distortion is perfectly compensated if the following relation holds:

$$G_\rho\left[F_\rho(r)\right] = Kr, \tag{2.13}$$

where r is the input amplitude and K is the desired linear gain. For the cascaded polar predistorter in Figure 2.10(a), the phase correction is given by solving

$$F_\theta(r) + G_\theta(r) = 0. \tag{2.14}$$

In the parallel configuration case of Figure 2.10(b), the phase correction is given by:

$$F_\theta(r) + G_\theta\left(F_\rho(r)\right) = 0. \tag{2.15}$$

The cascaded version of the polar predistorter is simpler and has a faster phase convergence. The parallel version has lower latency since the amplitude and phase corrections are done in parallel.

The polar predistorter has low memory requirements and fast convergence speed, which are comparable to the complex-gain predistorter. The polar predistorter can be easily calculated from the amplifier's AM-AM and AM-PM characteristics. Unlike the complex-gain predistorter, the polar predistorter does not require a complex multiplier, but a rectangular to polar conversion is required if used in a Cartesian transmitter. This makes it more suitable for polar transmitter configurations since the baseband modulation signal is already available in polar form.

2.3 Calibration and Adaptation

Memoryless predistorters can be implemented as a set of lookup tables that model the inverse of the PA distortion. Alternatively, a functional approximation of the inverse distortion can be used as well. The most popular form of functional approximation is the polynomial predistorter, which has been extensively documented in the literature [4,5,25,26,45]. Its main advantage is the relatively low number of parameters that is needed to model the predistorter. A predistorter can be constructed by first identifying the amplifier's nonlinear transfer characteristic and then calculating its inverse. However, the inversion process of the nonlinear function is often a non-trivial task. The indirect learning architecture illustrated in Figure 2.11 avoids the inversion step by directly computing the inverse nonlinear function. The calibration process as well as the adaptation of the predistorter depend on the type of predistorter that is considered.

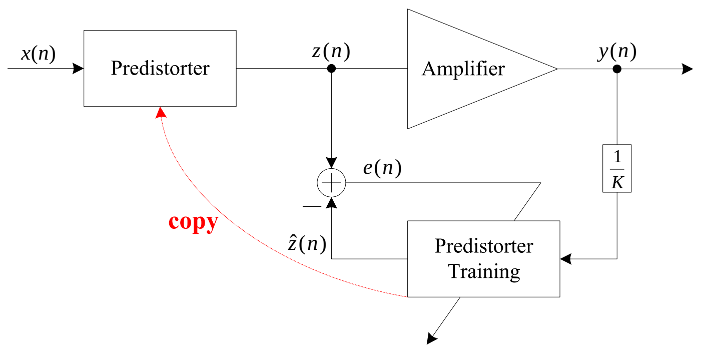


Figure 2.11: Indirect learning architecture.

2.3.1 Polynomial Predistorter

Polynomial functions or power series models are widely used in predistorter design. The polynomial approximation is attractive for its simplicity and the relatively reduced number of parameters. This model has been used to provide a good predistorter model for amplifiers presenting relatively weak nonlinearity such as class-A and class-AB amplifiers. The simplicity of such a model lies mainly in the fact that the output is a linear function of the parameters (or coefficients) to be estimated. Once the output signal corresponding to a chosen training signal is recovered, the predistorter design boils down to solving a linear least-squares system.

Let z_k denote the amplifier's input envelope samples and v_k the corresponding output values normalized by the gain K. The vector of polynomial predistorter coefficients $\mathbf{p}$ is obtained by solving the following equation:

$$\mathbf{z} = V\mathbf{p}, \tag{2.16}$$

where $\mathbf{z}$ and V are respectively given by

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_L \end{bmatrix} \quad \text{and} \quad V = \begin{bmatrix} v_1 & v_1^2 & \cdots & v_1^n \\ v_2 & v_2^2 & \cdots & v_2^n \\ \vdots & \vdots & \ddots & \vdots \\ v_L & v_L^2 & \cdots & v_L^n \end{bmatrix}.$$

To avoid numerical instabilities during the inversion of the matrix $\mathbf{V}$, the use of orthogonal polynomial basis is suggested in [42]. The high computational complexity associated with the least-squares identification method makes this solution quite unattractive for resource-constrained mobile devices. A significant amount of memory space is required to store the input and output samples z_k and v_k. Furthermore, this operation must be repeated periodically to track the nonlinear characteristic's drifts. A lower-complexity predistorter identification and adaptation method based on the adaptive least-mean-squares (LMS) algorithm is proposed in [4] and [45]. The algorithm in [4] was developed for the polar predistorter configuration discussed in Section 2.2.3. The amplitude and phase predistorters are modeled as polynomial functions of the input amplitude r.

$$\begin{aligned} F_\rho(r) &= \alpha_1 r + \alpha_2 r^2 + \cdots + \alpha_L r^L = \mathbf{a}^{\mathrm{T}} \mathbf{r}_\rho \\ F_\theta(r) &= \beta_0 + \beta_1 r + \beta_2 r^2 + \cdots + \beta_M r^M = \mathbf{b}^{\mathrm{T}} \mathbf{r}_\theta, \end{aligned} \tag{2.17}$$

with $\mathbf{r}_\rho = [r, r^2, \cdots, r^L]^{\mathrm{T}}$, $\mathbf{r}_\theta = [1, r, \cdots, r^M]^{\mathrm{T}}$, $\mathbf{a} = [\alpha_1, \cdots, \alpha_L]^{\mathrm{T}}$, and $\mathbf{b} = [\beta_1, \cdots, \beta_M]^{\mathrm{T}}$. The steepest descent algorithm is used to minimize the mean-squared amplitude and phase errors. The resulting LMS update algorithm is described by the following equations:

$$\mathbf{a}_{k+1} = \mathbf{a}_k + \mu_\rho \mathbf{r}_\rho^{[k]} e_\rho^{[k]} \tag{2.18}$$

$$\mathbf{b}_{k+1} = \mathbf{b}_k + \mu_\theta \mathbf{r}_\theta^{[k]} e_\theta^{[k]}, \tag{2.19}$$

where $e_\rho^{[k]}$ is the amplitude error between the kth input sample and the normalized output of the PA and $e_\theta^{[k]}$ is the corresponding phase error. μ_ρ and μ_θ are positive update coefficients that are carefully chosen to provide a suitable trade-off between the convergence speed and the steady-state error. This algorithm can be used for both the calibration and the adaptation of the predistorter. Its low complexity and minimal memory requirements make it particularly attractive. Even though the convergence speed is relatively slow, precise tracking of temperature and aging drifts is still achievable.

2.3.2 Lookup Table Predistorter

Building a LUT predistorter from a set of stored input and output complex envelope samples is a trivial process. But just as for the least-squares polynomial identification, high storage requirements and periodic updates contribute to the high cost and complexity of the system identification process. Alternative LUT adaptation techniques with low complexity and low memory requirements have been proposed in the literature. The linear update algorithm [11, 21, 39] consists of incremental small updates to LUT entries as they are accessed. When the nth entry of the LUT F is accessed and an error $e_n[k]$ is produced at the output of the PA, the updated LUT is computed as follows:

$$F_n[k+1] = F_n[k] + \mu \times e_n[k], \tag{2.20}$$

where μ is a positive step size that must be smaller than two to guarantee the stability of the algorithm. A value of μ less than one is required for better steady-state behavior, specially in the case of noisy output measurements. The linear update algorithm has a very low complexity but is plagued by slow convergence speed, which also depends on the signal envelope's statistics.

The secant update algorithm was proposed in [8] to provide faster convergence. With the secant method, the nth LUT entry is updated as follows:

$$F_n[k+1] = \frac{F_n[k-1]e_n[k] - F_n[k]e_n[k-1]}{e_n[k] - e_n[k-1]} \tag{2.21}$$

The secant update is reported as being twice as fast as the linear update algorithm. The cost of this improvement is the additional multiplications and the division required during the update step and to a lesser extent, the memory space needed to store the previous entries $F_n[k-1]$ and the error samples $e_n[k-1]$.

2.3.3 Predistorter with Memory

Memoryless predistortion of a high power amplifier (HPA) that exhibits strong memory yields very marginal performance improvements [33]. The predistorter itself must be a system with memory in order to successfully mitigate the memory effects. A memory predistorter can be generally represented by a discrete Volterra system [19, 34]:

$$\begin{aligned} y[n] = &\sum_{q=0}^{Q} h_q^{(1)} x[n-q] + \sum_{q1=0}^{Q}\sum_{q2=0}^{Q} h_{q1,q2}^{(2)} x[n-q_1]x[n-q_2] \\ &+ \sum_{q1=0}^{Q}\sum_{q2=0}^{Q}\sum_{q3=0}^{Q} h_{q1,q2,q3}^{(3)} x[n-q_1]x[n-q_2]x[n-q_3]\cdots . \end{aligned} \tag{2.22}$$

The Volterra system is not well suited to practical implementation because of the large number of parameters and the non-trivial estimation process involved. A special case of the Volterra system with a significantly lower number of parameters is the Hammerstein predistorter described in Figure 2.12.

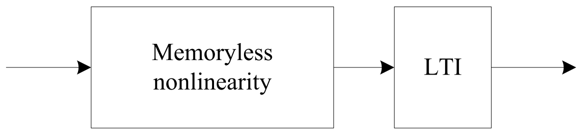


Figure 2.12: Hammerstein memory predistorter

The Hammerstein predistorter is merely the cascade of a memoryless nonlinearity and a linear time invariant (LTI) system that emulates the desired memory effect. The Hammerstein system can be identified by using an iterative process such as Newton's or Narenda-Gallman (NG) algorithms [18]. The potential convergence issues of the latter iterative algorithms can be avoided by using the two-stage LS/SVD algorithm presented in [16]. Another special case of the Volterra system is the memory polynomial [17] described by the following equation:

$$z[n] = \sum_{k=1}^{K} \sum_{q=0}^{Q} h_{k,q}\, x[n-q]\, |x[n-q]|^{k-1} \,. \tag{2.23}$$

The parameters of the memory polynomial system can be identified by using the least-squares approximation. Even though the memory polynomial predistorter requires a larger number of parameters than the Hammerstein system, it is more robust and the parameter identification process is much simpler.

CHAPTER III

SPACING OF A POLAR LUT PREDISTORTER

As previously discussed in Chapter 2, predistorters can implemented as lookup tables (LUT). Functional approximations such as fitted polynomials or power series approximations can be used as well. LUT predistorters have minimal computational complexity and can implement arbitrary nonlinear mappings. But they require significantly more memory space to store the model parameters than a polynomial approximation. A direct consequence of the larger number of parameters is a relatively slow convergence speed of iterative training algorithms. On the other hand, evaluating a polynomial function is more computationally complex than a simple memory lookup. Furthermore, compensating higher orders of nonlinearity requires a high-order polynomial predistorter evaluated at a sampling frequency a few orders of magnitude higher than the bandwidth of the input baseband signal. For modern high spectral efficiency modulation techniques such as WCDMA and OFDM, a predistorter bandwidth of several tens of MHz might be required. In this case, dedicated high-speed hardware is needed to implement a polynomial predistorter, therefore reducing its appeal in low-cost cellular handset applications. This chapter will focus on LUT enhancement techniques that combine the inherent low complexity of table lookup with reduced memory requirements achieved by using interpolation and efficient spacing of table entries.

3.1 Analysis of Linearly-Interpolated LUT

The use of linearly interpolated LUT predistorters has been reported in the literature [21] as an efficient way to reduce LUT approximation errors. A low-complexity and practical implementation of LUT interpolation is also discussed in [47]. In the present study, the problem is approached with a mathematical justification. The polar predistorter configuration [21] is considered here. The complex-gain configuration will be addressed in Chapter 4. A polar predistorter is illustrated in Figure 3.1.

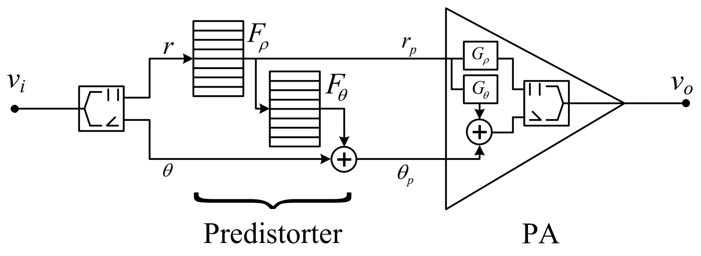


Figure 3.1: Polar predistorter LUT arrangement (complex baseband model)

This setup uses the cascaded LUT arrangement discussed in Section 2.2.3. The amplifier's nonlinear AM-AM and AM-PM distortions are respectively designated by the amplitude-dependent functions $g_\rho(r)$ and $g_\theta(r)$. To simplify the mathematical expressions, it is assumed that the gain of the amplifier is normalized to one (i.e., $K = 1$). The polar predistorter consists of two LUTs approximating the inverse function of the amplifier's amplitude distortion $f_\rho(r) = g_\rho^{-1}(r)$ and the phase compensation function $f_\theta(r) = -g_\theta(r)$. The LUT approximation error of $f_\rho(r)$ and $f_\theta(r)$ respectively result in independently computable amplitude and phase errors at the output of the PA. The derivation for the two types of errors is similar. Therefore, the focus here will be on the amplitude table. Let us consider the kth bin of the amplitude LUT, which is delimited by the amplitude entries r_k and r_{k+1} and let $d_k = r_{k+1} - r_k$ be the width of the kth interval. The amplitude predistorter $f_\rho(r)$ will be precisely determined at the amplitudes r_k and r_{k+1}, where the approximation error is equal to zero. For an input amplitude $r_i = r_k + \varepsilon_r$, with $0 < \varepsilon_r \leq d_k$, the output of the LUT predistorter is $f_\rho(r_i) + \varepsilon_{f_\rho}$. Assuming that the predistorter $f_\rho(r)$ is at least twice continuously differentiable and $f_\rho''(r)$ varies little within the bin, it is shown in Appendix A that the linear interpolation results in an approximation error given by

$$\varepsilon_{f_\rho} = f_\rho''(r)\,\frac{\varepsilon_r\,(\varepsilon_r - d_k)}{2}, \tag{3.1}$$

with f_ρ'' being the second-order derivative of f_ρ. The amplitude at the output of the amplifier, with normalized gain $K = 1$ is

$$\begin{aligned} r_o &= g_\rho\left[f_\rho(r_i) + \varepsilon_{f\rho}\right] \\ &\approx g_\rho\left[f_\rho(r_i)\right] + \varepsilon_{f\rho} g'_\rho\left[f_\rho(r_i)\right] \\ &\approx r_i + \frac{f''_\rho(r_i)}{f'_\rho(r_i)} \frac{\varepsilon_r\left(\varepsilon_r - d_k\right)}{2}, \end{aligned} \tag{3.2}$$

where (3.2) uses the relation between f_ρ and g_ρ, i.e., $f_\rho = g_\rho^{-1} \Rightarrow g'_\rho = 1/f'_\rho$. It is also assumed that f'_ρ does not change appreciably within the bin (i.e, the number of LUT entries is large enough). The amplitude error measured at the amplifier's output can therefore be expressed as follows:

$$\begin{aligned} e_\rho &= r_o - r_i \\ &= \frac{f''_\rho(r_i)}{f'_\rho(r_i)} \frac{\varepsilon_r\left(\varepsilon_r - d_k\right)}{2} \end{aligned} \tag{3.3}$$

Calculating the phase error at the amplifier's output is a more straightforward process since the phase LUT approximation error is simply propagated to the output phase. Therefore, the output phase error is simply expressed as follows:

$$e_\theta = g''_\rho(r_i)\, \frac{\varepsilon_r\left(\varepsilon_r - d_k\right)}{2} \tag{3.4}$$

Assuming that the amplitude and phase errors are small enough, the complex baseband equivalent output of the amplifier can be written as

$$\begin{aligned} v_o &= \left(r_i + e_r\right) e^{j(\theta_i + e_\theta)} \\ &\approx \left(r_i + e_r\right) \left[e^{j\theta_i} + e_\theta e^{j\left(\theta_i + \frac{\pi}{2}\right)}\right] \\ &\approx v_i + e_r\, e^{j\theta_i} + e_\theta\, r_i\, e^{j\left(\theta_i + \frac{\pi}{2}\right)}. \end{aligned} \tag{3.5}$$

The complex error at the output of the PA is thus given by

$$\begin{aligned} e_o &= v_o - v_i \\ &\approx e_r\, e^{j\theta_i} + e_\theta\, r_i\, e^{j\left(\theta_i + \frac{\pi}{2}\right)}. \end{aligned} \tag{3.6}$$

The residual error at the output of the PA consists of two terms resulting from the approximation errors in the amplitude and phase LUTs. These two error terms can be considered independent of each other to simplify the remaining calculations. The total noise contribution of the kth bin to the output SNR can be computed using the usual quantizer assumption [24] that ε_r is a random variable uniformly distributed in $[0, d_k]$. The mean-squared error (MSE) contribution of the kth bin of the amplitude table is

$$\begin{aligned} E\left[|e_\rho|^2\right] &= \frac{f_\rho''(r_k)^2}{4\,d_k\,f_\rho'(r_k)^2}\int_0^{d_k} \varepsilon^2(\varepsilon-d_k)^2\,d\varepsilon \\ &= \frac{f_\rho''(r_k)^2}{120\,f_\rho'(r_k)^2}\,d_k^4. \end{aligned} \tag{3.7}$$

Similarly, the contribution of the kth of the phase LUT to the total residual error is

$$E\left[|e_\theta|^2\right] \approx \frac{f_\theta''(r_k)^2}{120} r_k^2 d_k^4. \tag{3.8}$$

For an arbitrary LUT spacing achieved using a compander [24] $c(r)$, the bin width is related to the compander by

$$d_k \approx \frac{1}{Nc'(r_k)}, \tag{3.9}$$

where N is the LUT size and V is the maximum amplitude addressable by the LUT. For the special case of uniform spacing $c(r) = r$ and $d_k = 1/N$ is constant. Assuming that the number of LUT bins is large enough, the total residual distortion power at the PA output can be approximated as follows:

$$P_{\text{dis}} = P_\rho + P_\theta \tag{3.10}$$

with

$$P_\rho = \frac{1}{120\,N^4}\int_0^1 \frac{f_\rho''(r)^2}{c'(r)^4\,f_\rho'(r)^2}\,p(r)\,dr \tag{3.11}$$

$$P_\theta = \frac{1}{120\,N^4}\int_0^{\tilde{A}} \frac{f_\theta''(r)^2}{c'(r)^4}\,r^2\tilde{p}(r)\,dr \tag{3.12}$$

where $p(r)$ is the probability density function (pdf) of the input amplitude and $\tilde{p}(r)$ is the pdf of the predistorted amplitude of the input signal. Equations (3.11) and (3.12) show that the mean-squared amplitude and phase errors are inversely proportional to N^4, which is a much faster rate of decrease than for the ZOH LUT [11], which results in a MSE that is inversely proportional to N^2. The predistorted signal's SNR increases by 12 dB if the LUT size is doubled as opposed to just 6 dB for the ZOH LUT.

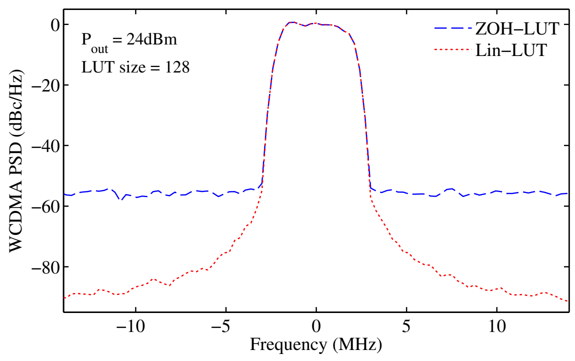


Figure 3.2: PSD of a WCDMA signal using a LIN-LUT and a ZOH-LUT, with $N = 128$

A comparison of the two approaches using a WCDMA input is summarized in Figure 3.2 and Figure 3.3. Figure 3.3(a) shows that linear interpolation improves the WCDMA error vector magnitude (EVM) by 25 dB. The adjacent channel leakage ratio at 5 MHz offset (ACLR1) and 10 MHz offset (ACLR2) are respectively improved by 0.5 dB and 17 dB.

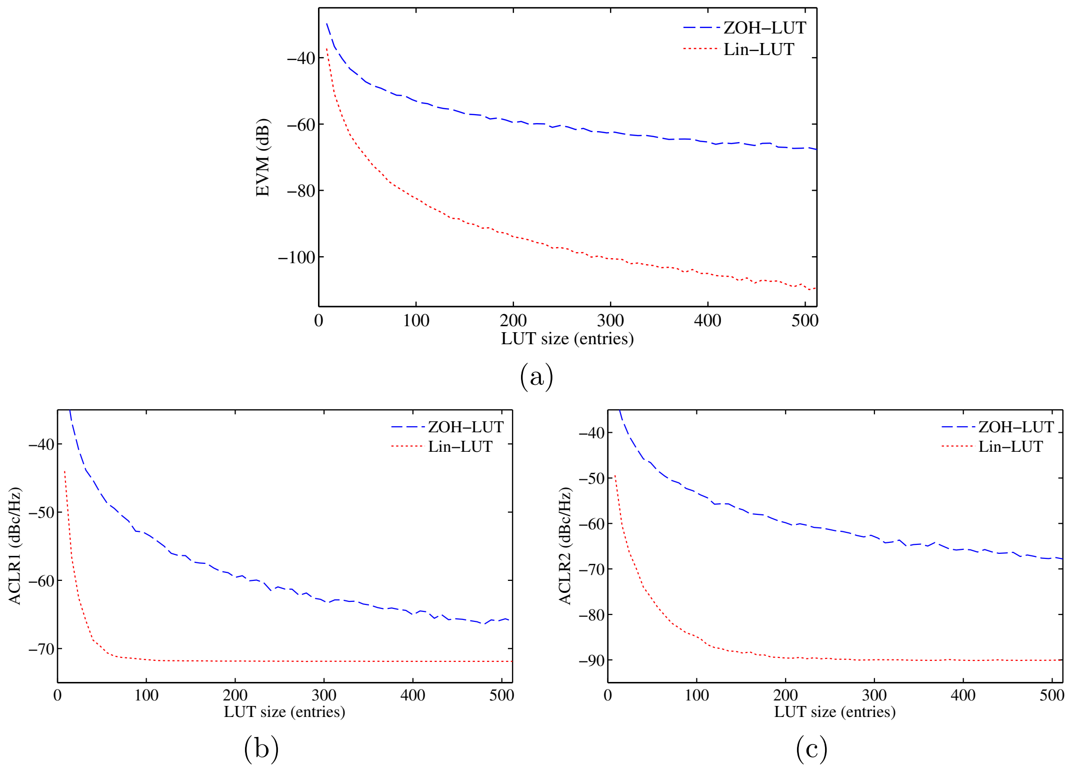


Figure 3.3: WCDMA measurements for LIN-LUT and ZOH-LUT. (a) Error vector magnitude. (b) Adjacent leakage ratio at 5 MHz offset (ACLR1). (c) Adjacent leakage ratio at 10 MHz offset (ACLR2).

3.2 Analysis of LUT Predistorter Spacing

The early implementations of LUT predistorters were mainly based on uniform spacing in power to minimize the complexity of the LUT address calculation. To reduce LUT approximation errors, non-uniform spacing has been studied for nearest-neighbor or zero-order hold LUT (ZOH-LUT) predistorters. Several spacing schemes for the complex-gain predistorter configuration have been proposed in the literature. In [11], a closed-form optimal spacing expression that depends on the signal's probability distribution as well as the amplifier's characteristics has been derived. This result is modified in [6] to remove the dependency with respect to the input signal's statistics while maintaining a considerable performance advantage over uniform spacing.

Other LUT techniques have been presented in [29, 35, 36, 38]. These results mainly address the complex-gain predistorter configuration. In this work, the optimal spacing expression will be derived for the polar LUT predistorter configuration (Figure 2.10), in which the amplitude and phase predistorters reside in separate LUTs. It was shown in Section 3.1 that linear interpolation tremendously improves the performance of LUT predistortion with relatively little added complexity. It is therefore important to consider optimal spacing in the context of linearly-interpolated LUTs (LIN-LUT) as well.

3.2.1 Optimal Spacing of Nearest Neighbor LUT Predistorter

The problem of optimal LUT spacing is quite similar to the design of an optimal quantizer. In the case of the polar LUT predistorter in Figure 3.1, both amplitude and phase predistorter LUTs must be optimally spaced to minimize the total transmitted residual distortion power resulting from LUT approximations. Optimal spacing can be achieved by applying a suitable compander $c(r)$ to the amplitude signal prior to addressing the LUT. This is illustrated in Figure 3.4. The objective of this study is to find a set of companders c_ρ and c_θ that minimize the total residual nonlinear distortion power at the output of the PA. To simplify the derived mathematical expressions, it is assumed that the input signal's amplitude and the amplifier's gain are normalized to unity. The generalization of the presented results is a matter of trivial extension.

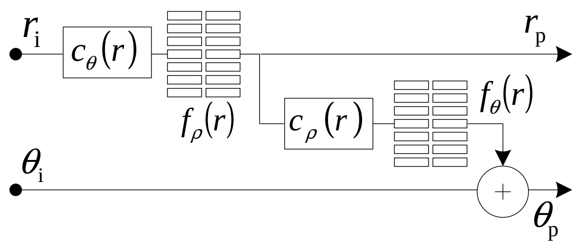


Figure 3.4: Non uniform LUT spacing by companding

The amplitude and phase LUTs are assumed to be of the same size N. Let us first consider the amplitude LUT. The amplitude signal r_i is quantized to the nearest LUT entry r_k, therefore causing a quantization error $\varepsilon_r = r_k - r_i$. Assuming that r_k is located in the middle of kth LUT interval with width d_k, then ε_k is bounded by $-d_k/2 < \varepsilon_r < d_k/2$. The quantized predistorted amplitude is

$$\begin{aligned} r_\mathrm{p} &= f_\rho(r_k) \\ &= f_\rho(r_i + \varepsilon_r). \end{aligned} \tag{3.13}$$

Assuming a gain of unity and applying (2.13) with $K = 1$, the output amplitude is given by

$$r_o = g_\rho[f_\rho(r_i + \varepsilon_r)] = r_i + \varepsilon_r. \tag{3.14}$$

Similarly, the output phase is given by

$$\theta_o = \theta_i + f_\theta(r_p + \tilde{\varepsilon}_r) + g_\theta(r_\mathrm{p}). \tag{3.15}$$

Using first-order approximation,

$$f_\theta(r_p + \tilde{\varepsilon}_r) \approx f_\theta(r_p) + \tilde{\varepsilon}_r f'_\theta(r_p).$$

The output phase can be simplified to

$$\begin{aligned} \theta_o &\approx \theta_i + f_\theta(r_p) + \tilde{\varepsilon}_r f'_\theta(r_p) + g_\theta(r_p) \\ &\approx \theta_i + \tilde{\varepsilon}_r\, f'_\theta(r_p). \end{aligned} \tag{3.16}$$

Assuming that the amplitude and phase errors are small enough, the output of the amplifier can be written as follows:

$$\begin{aligned} v_o &= (r_i + \varepsilon_r)\, e^{j\left(\theta_i + \tilde{\varepsilon}_r\, f_\theta'(r_p)\right)} \\ &\approx (r_i + \varepsilon_r)\, \left[e^{j\theta_i} + \tilde{\varepsilon}_r\, f_\theta'(r_p) e^{j\left(\theta_i + \frac{\pi}{2}\right)}\right] \\ &\approx v_i + \varepsilon_r\, e^{j\theta_i} + \tilde{\varepsilon}_r\, r_i\, f_\theta'(r_p)\, e^{j\left(\theta_i + \frac{\pi}{2}\right)}, \end{aligned} \tag{3.17}$$

with

$$v_i = r_i e^{j\theta_i} \qquad \text{and} \qquad v_o = r_o e^{j\theta_o}.$$

The distortion at the output of the PA is thus given by

$$e_o \approx \varepsilon_r e^{j\theta_i} + \tilde{\varepsilon}_r\, r_i\, f_\theta'(r_p)\, e^{j\left(\theta_i + \frac{\pi}{2}\right)}. \tag{3.18}$$

The result in equation (3.18) shows that the residual distortion error at the output of the PA consists of two terms resulting from the approximation errors in the amplitude and phase LUTs respectively. These two error terms can be reasonably considered independent of each other. For an LUT size N that is large enough, the errors ε_r and $\tilde{\varepsilon}_r$ can be approximated by uniformly distributed zero-mean random variables over their respective interval $[-\frac{d_k}{2}, \frac{d_k}{2}]$. Using the approach described in [11], the total residual distortion power and the optimal companders can be estimated. The residual distortion power is given by

$$\begin{aligned} P_{\text{dis}} &= E\left[|e_o|^2\right] \\ &= \frac{1}{12N^2}\left[\int_0^1 \frac{w_\rho(r)}{c_\rho(r)}\, dr + \int_0^1 \frac{w_\theta(r)}{c_\theta(r)}\, dr\right], \end{aligned} \tag{3.19}$$

where

$$w_\rho(r_i) = p(r) \qquad \text{and} \qquad w_\theta(r) = \left|g_\rho(r) f_\theta'(r)\right|^2 \tilde{p}(r).$$

$p(r)$ and $\tilde{p}(r)$ are the probability density functions of the input amplitude and predistorted amplitude, respectively. It is observed that similarly to the complex-gain predistorter configuration in [11], the total residual distortion power is inversely proportional to N^2. In other words, the residual distortion resulting from ZOH-LUT approximations decreases by 6 dB when the LUT size is doubled. The optimal companders for the non-interpolated polar predistorter LUTs are found by using an approach similar to the one described in [11]:

$$c_\rho(r) = \frac{w_\rho(r)^{1/3}}{\int_0^1 w_\rho(r)^{1/3} dr} \qquad \text{and} \qquad c_\theta(r) = \frac{w_\theta(r)^{1/3}}{\int_0^1 w_\theta(r)^{1/3}\, dr}. \tag{3.20}$$

It is interesting to note that the optimal amplitude LUT spacing only depends on the input probability density function. Experimental results have also shown that the dependence on the amplifier characteristics can be generally neglected for the phase LUT spacing. A residual distortion power difference of less than 0.1 dB is measured if $w_\theta \approx \tilde{p}(r)$ is used in computing the phase LUT compander. This result corresponds to the traditional optimal quantizer in [24]. Therefore, the same LUT spacing can be used for both amplitude and phase LUTs if $p(r) \approx \tilde{p}(r)$. In this case, the spacing only depends on the input amplitude's statistics. Alternatively, instead of using an explicit compander, the LUT can be defined as optimally spaced pairs of input and output values $[r_k, f(r_k)]$, which are obtained by using the Lloyd-Max algorithm [24]. The theoretical results derived above have been validated through extensive simulations. The PA model is based on the measured characteristics extracted from a near class-E RF power amplifier. The AM-AM and AM-PM profiles are shown in Figure 3.5.

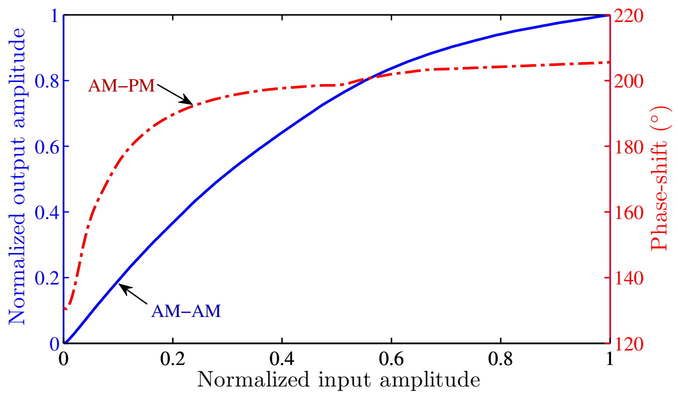


Figure 3.5: Class-E amplifier characteristics.

The signal to noise and distortion ratio (SNDR) has been measured from simulations for various LUT sizes, with both uniform and optimal spacing. The optimal spacing results in a 2.5 dB decrease of the residual distortion power when compared to uniform spacing. The experimental results in Figure 3.6 confirm the theoretical assertion that the residual distortion power decreases by 6 dB each time the LUT size is doubled. Moreover, the performance measured with the iterative Lloyd-Max algorithm spacing closely matches that of the optimal compander.

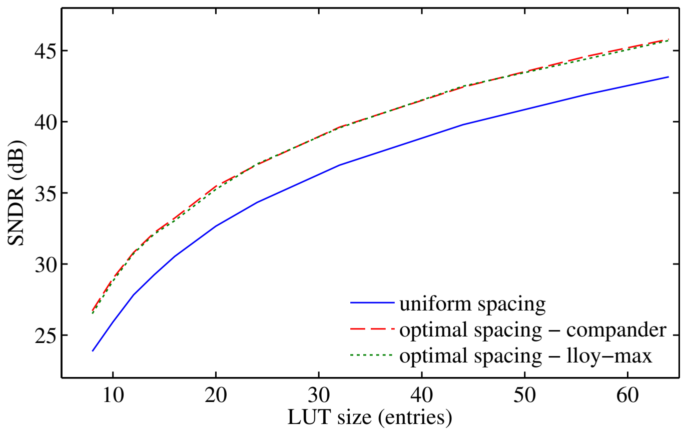


Figure 3.6: LUT distortion-to-signal power ratio vs. LUT size for ZOH-LUT.

3.2.2 Optimal Spacing of Linearly-Interpolated Polar LUT Predistorters

The use of linearly-interpolated LUT predistorters has been justified experimentally in the literature [21] as an efficient way to reduce the LUT approximation errors. A low-complexity and practical implementation of linear interpolation in LUT predistorters is also discussed in [47]. The problem has been approached with mathematical justification in Section 3.1 and the theoretical results validated with simulated experiments. As previously shown in Section 3.1, the residual distortion power resulting from linearly approximation errors can be written as follows:

$$P_{\text{dis}} = \frac{1}{120N^4}\left[\int_0^1 \frac{w_\rho(r_i)}{c'_\rho(r_i)^4}\,dr_i + \int_0^1 \frac{w_\theta(r_p)}{c'_\theta(r_p)^4}\,dr_p\right], \tag{3.21}$$

where

$$w_\rho(r_i) = \left|\frac{f''_\rho(r_i)}{f'_\rho(r_i)}\right|^2 p(r_i) \qquad \text{and} \qquad w_\theta(r_p) = |g_\rho(r_p) f''_\theta(r_p)|^2 p(r_p)$$

The set of companders that minimize the residual distortion power are found by using an approach similar to [11]:

$$c_\rho(r) = \frac{w_\rho(r)^{1/5}}{\int_0^1 w_\rho(r)^{1/5} dr} \qquad \text{and} \qquad c_\theta(r) = \frac{w_\theta(r)^{1/5}}{\int_0^1 w_\theta(r)^{1/5} dr}. \tag{3.22}$$

Equation (3.21) shows that the residual distortion power resulting from approximation errors in the linearly-interpolated predistorter is inversely proportional to N^4. The residual distortion power decreases by 12 dB when the LUT size is doubled as opposed to 6 dB for the nearest-neighbor LUT (ZOH-LUT). These results have been validated by simulations using the PA model previously shown in Figure 3.5. The simulated results are summarized in Figure 3.7.

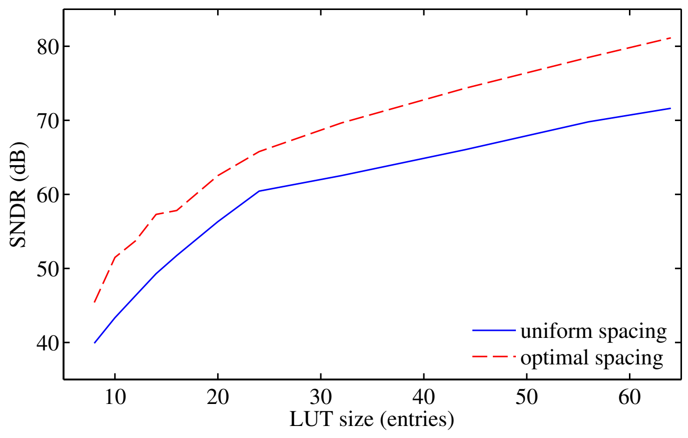


Figure 3.7: LUT distortion-to-signal-power ratio vs. LUT size for linearly-interpolated LUT.

The plot in Figure 3.7 shows that when optimal spacing is used in a linearly-interpolated LUT predistorter with 64 entries, the residual distortion power is reduced by about 10 dB, as opposed to just 3 dB for the ZOH-LUT predistorter. Therefore, the important conclusion reached here is that optimal spacing is more beneficial and justifiable in the context of linearly-interpolated LUT predistorters.

CHAPTER IV

OPTIMAL SPACING OF INTERPOLATED COMPLEX-GAIN LUT PREDISTORTERS

Increasing power-added efficiency, while maintaining requisite transmission characteristics, is a major concern for resource-constrained mobile devices. Unfortunately, power-efficient amplifiers have nonlinear amplitude-domain and phase-domain transfer characteristics that degrade the spectral efficiency of modern complex-envelope modulation signals, such as EDGE, WCDMA, and the OFDM family of modulations, to name a few. The nonlinearity of an amplifier manifests itself via in-band and out-of-band spectral regrowth [53], leading to violation of the strict modulation spectral mask and adjacent channel leakage specifications. Several baseband linearization techniques have been proposed to date, and the lookup table (LUT) based digital predistorter is one the most widely used distortion-mitigation techniques. This is because of its low implementation complexity, simplicity of operation, and capability to represent arbitrary nonlinear mappings.

In early implementations of digital baseband predistorters, the use of uniform spacing in the power domain was mainly motivated by the convenient computation of the instantaneous power ($I^2 + Q^2$), as opposed to using a suitably-accurate amplitude approximation. The uniform spacing in power has the effect of concentrating the entries near the higher amplitude region. This is suitable for a class-A amplifier since its characteristic curve is only compressed near maximum amplitudes. However, this is not well suited to amplifiers with higher power efficiency, such as class-AB, C, E, etc., which exhibit significant nonlinear amplitude and phase distortions across the entire amplitude range.

The goal of optimal LUT spacing is to reduce the level of residual distortion resulting from predistorter approximation errors. This would consequently allow the use of a smaller LUT size to achieve the targeted performance. In addition to memory savings, this would also result in faster convergence of iterative LUT training algorithms such as those described in [8, 21, 39]. In an attempt to achieve this goal, a theoretical closed-form optimal spacing expression, which depends on the signal probability distribution as well as the amplifier characteristics, was derived in [11]. As a result, an additional 4 to 5 dB intermodulation (IMD) power rejection was reported. This result is modified in [6] to remove the dependency with respect to input signal statistics while still maintaining a non-negligible performance advantage over uniform spacing. Other LUT spacing techniques have also been presented in [29, 35, 36, 38, 47]. Note that these prior-art LUT studies address non-interpolated lookup tables, henceforth referred to as zero-order-hold LUTs (ZOH-LUTs).

Linearly interpolated LUTs (LIN-LUTs) have been experimentally shown to improve predistorter performance in [21] and [47]. In this chapter, the performance improvement of a LIN-LUT predistorter is theoretically derived and validated through simulations. It is shown that the use of linear interpolation alone significantly reduces the minimum LUT size needed to meet the spectral performance required by modern cellular standards such as EDGE, WCDMA, WiMax, LTE, etc. In light of this result, the optimal spacing of a LIN-LUT predistorter is derived and is shown to provide a greater performance impact than in the case of the ZOH-LUT. An earlier study of LIN-LUT spacing [2] was dedicated to the polar predistorter configuration. The present chapter deals with the more widespread complex-gain predistorter configuration.

The residual distortion in a transmitter linearized using a memoryless LUT predistorter is derived in Section 4.1. The optimal spacing of a LIN-LUT is derived and validated in Section 4.2. An alternative method that consists of separately optimizing the spacing of the real-gain and imaginary-gain tables is studied in Section 4.3. In Section 4.4, the practical implementation of optimal spacing is addressed, and hardware synthesis results are presented.

4.1 Residual Distortion in LUT-Predistorted Transmitters

A power amplifier's nonlinear characteristics consist of amplitude-dependent gain and phase-shift curves, respectively called AM-AM and AM-PM conversion functions. In this work, the combination of AM-AM and AM-PM distortions is modeled as a single amplitude-dependent complex gain $g(r)$, as illustrated in Figure 4.1. The transmitter is successfully linearized if the complex-gain predistorter $f(r)$ meets the following condition:

$$g[r|f(r)|]f(r) = Ke^{j\phi_0}, \tag{4.1}$$

where K is the equivalent gain of the linearized amplifier, and ϕ_0 is an arbitrary constant phase-shift. For mathematical convenience, ϕ_0 will be assumed to be zero in this work. An arbitrary spacing of the complex-gain LUT is achieved by selecting an appropriate function $c(r)$, as shown in Figure 4.1. In related prior studies [11] and [6], the function $c(r)$ is called a companding function or simply a compander. For consistency, the same terminology is adopted in this study, although strictly speaking, *companding* means compression followed by expansion. As a first-order approximation, the density of LUT entries is inversely proportional to the first-order derivative of the compander $c(r)$. The LUT is uniformly spaced in amplitude if the compander is the identity function $[c(r) = r]$.

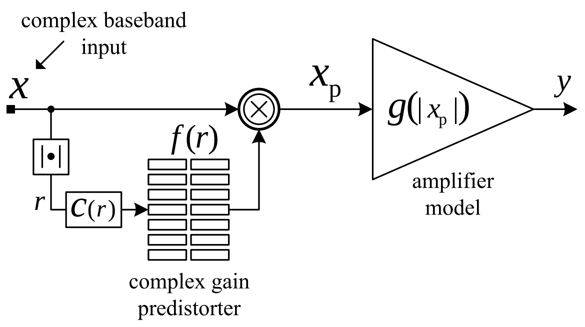


Figure 4.1: Complex-gain LUT predistorter arrangement with unified compander.

The approximation errors resulting from the LUT representation of the complex-gain predistorter translate into residual nonlinear distortion at the output of the transmitter. A simple, but not optimal, solution is to find the compander $c(r)$ that minimizes the error at the LUT output. But the overall optimal compander is the function $c_o(r)$ that minimizes the residual nonlinear distortion at the output of the amplifier. Because of the cascade of nonlinear functions involved, the derivation of the theoretically optimal compander is not a trivial operation. To obtain a closed-form solution to this multi-variable optimization problem, appropriate approximations are introduced as suited.

For mathematical convenience, we will consider, without loss of generality, that the gain of the amplifier K is normalized to unity and the input signal amplitude is also normalized to lie within the interval $[0, 1]$. The predistorter $f(r)$ in (4.1) is approximated by a linearly interpolated complex-gain LUT, as illustrated in Figure 4.1. The error resulting from this linear approximation will cause residual nonlinear distortion at the output of the transmit chain. Using an approach often employed in quantization theory [24,46], the total distortion at the output of the LUT can be approximated by integrating the error contributions across all the LUT bins, where a bin is defined as the region between two consecutive LUT entries.

Let us consider the kth bin of the complex-gain LUT, which is delimited by the entries indexed by the amplitudes r_k and r_{k+1}, and let $d = r_{k+1} - r_k$ be the width of this kth bin. The complex-gain predistorter $f(r)$ is precisely determined only at r_k and r_{k+1}, as shown in Figure 4.2(a).

For a complex input x, with amplitude $r = |x|$ falling anywhere else within the bin (i.e. $r = r_k + \varepsilon_r$, with $0 < \varepsilon_r < d$), there is a complex-gain approximation error ε_f:

$$\tilde{f}(r) = f(r) + \varepsilon_f.$$

Assuming that the bin is small enough so that the second-order derivative of the complex-gain varies little within the considered interval, the linear approximation error can be obtained after a few algebraic manipulations as

$$\varepsilon_f = \frac{\varepsilon_r(\varepsilon_r - d)}{2} f''(r_0), \tag{4.2}$$

where the amplitude r_0 is chosen as the center of the bin. Since the second-order derivative is assumed to be relatively constant within the bin, any point between r_k and r_{k+1} will yield a decent approximation. An example of this error estimation is illustrated in Figure 4.2(b), using the function $f(r) = r + 2r^2 + r^3/2$. The linear interpolation error estimated in (4.2) is shown to be very close to the actual computed error.

The linear approximation error calculated above will be propagated through the nonlinear amplifier. The complex baseband-equivalent signal at the output of the amplifier is given by

$$\begin{aligned}\tilde{y} &= g[r|\tilde{f}(r)|]\,\tilde{f}(r)\,x \\ &= g[r|f(r)+\varepsilon_f|]\Big[f(r)+\varepsilon_f\Big]x. \qquad (4.3)\end{aligned}$$

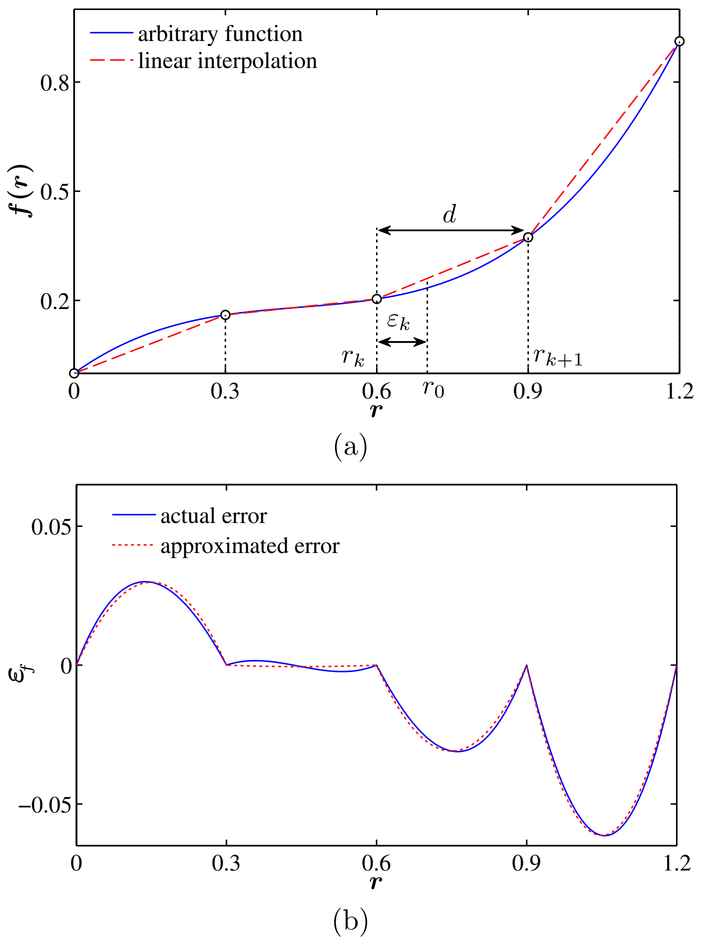


Figure 4.2: Linear interpolation of an arbitrary function. (a) Piece-wise linear interpolation. (b) Comparison of the actual and approximated residual error functions.

Using first-order approximation, the amplifier gain can be rewritten:

$$g[r|f(r)+\varepsilon_f|] = g[r|f(r)|] + \frac{\Re[f^*(r)\varepsilon_f]}{|f(r)|}g'[r|f(r)|]r,$$

where $\Re[\cdot]$ designates the real part of the argument. This result is substituted in (4.3) to get

$$\tilde{y} = \left\{g[r|f(r)|] + \frac{\Re[f^*(r)\varepsilon_f]}{|f(r)|}g'[r|f(r)|]r\right\}\Big[f(r)+\varepsilon_f\Big]x. \qquad (4.4)$$

The error at the output of the amplifier is obtained by subtracting (4.4) from the expression of the desired, distortion-free output $y = g[r|f(r)|]\,f(r)\,x$ and substituting (4.2):

$$\begin{aligned}
\varepsilon_y &= y - \tilde{y} \\
&\approx -\left\{ \varepsilon_f g[r|f(r)|] + r\,f(r)g'[r|f(r)|]\,\frac{\Re[f^*(r)\varepsilon_f]}{|f(r)|} \right\} x \\
&= \frac{\varepsilon_r(\varepsilon_r - d)}{2}\psi(r),
\end{aligned}$$

with

$$\psi(r) = \left\{ f''(r)g[r|f(r)|] + rf(r)g'[r|f(r)|]\,\frac{\Re[f^*(r)f''(r)]}{|f(r)|} \right\} x.$$

Taking the derivative of (4.1) and combining it with the above equation, we get an expression of $\psi(r)$ as a function of the complex-gain predistorter $f(r)$:

$$\psi(r) = \frac{f''(r)f^*(r) + j\,r\,\Im[f'^*(r)f''(r)]}{|f(r)|^2 + r\,\Re[f^*(r)f'(r)]}x, \tag{4.5}$$

where $\Im[\cdot]$ designates the imaginary part of the argument. A common approximation in quantization analysis is to assume that the amplitude displacement ε_r is a random variable, uniformly distributed across the width of the bin. Using this assumption, the contribution of the bin to the total residual distortion can be calculated as follows:

$$\begin{aligned}
E\Big[|\varepsilon_y|^2\Big] &= E\Big[|\varepsilon_r(\varepsilon_r - d)|^2\Big]\,\frac{|\psi(r)|^2}{4} \\
&= \frac{|\psi(r)|^2}{4d}\int_0^d \varepsilon^2(\varepsilon - d)^2\,d\varepsilon.
\end{aligned}$$

Evaluating the integral in the above equation and weighting the result with the amplitude probability density function leads to

$$E_\varepsilon = \alpha\,d^4\,p(r)|\psi(r)|^2, \tag{4.6}$$

where $\alpha = 1/120$ is a real constant. The bin width d can be expressed as a function of the compander $c(r)$ [11], yielding

$$d(r) \approx \frac{1}{N\,c'(r)},$$

where N is the LUT size. The total residual distortion can then be calculated from the above equation and the probability density of the input amplitude $p(r)$:

$$\begin{aligned} P_{\varepsilon_y} &= \int E_\varepsilon \, dr \\ &= \alpha \int |\psi(r)|^2 d^4(r) p(r) \, dr \\ &= \frac{\alpha}{N^4} \int \frac{|\psi(r)|^2}{c'^4(r)} p(r) \, dr. \end{aligned} \tag{4.7}$$

It is interesting to note that the residual distortion due to linear approximation errors is inversely proportional to N^4, as opposed to N^2 in the case of the non-interpolated complex-gain LUT [11]. The residual distortion is therefore decreased by 12 dB for every additional bit of precision (whenever the size of the LUT is doubled).

The theoretical expression for the total distortion in (4.7) and its distribution across the amplitude range given by (4.6) were validated with simulations based on a class-E amplifier model using a WCDMA signal. The real and imaginary parts of the class-E amplitude-dependent complex-gain $g(\cdot)$ are shown in Figure 4.3 for three different temperature settings. The nominal curve at 25°C is used for the purpose of the present experiment.

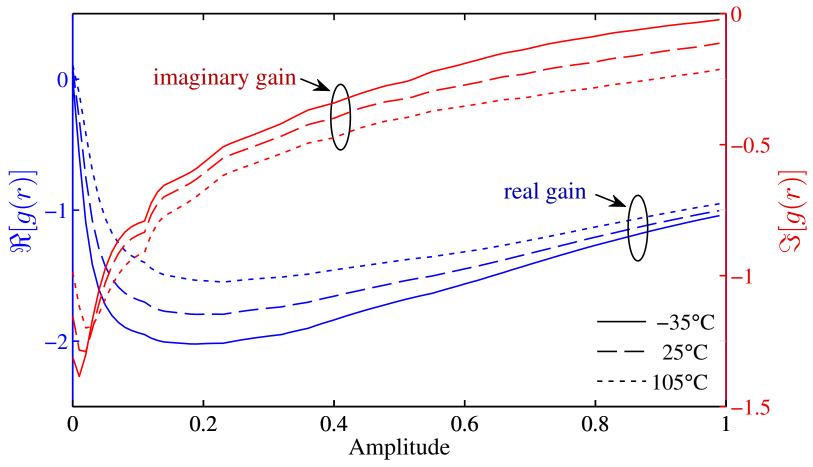
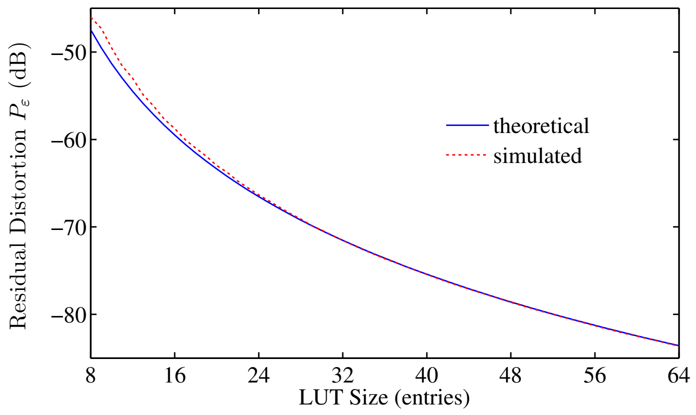
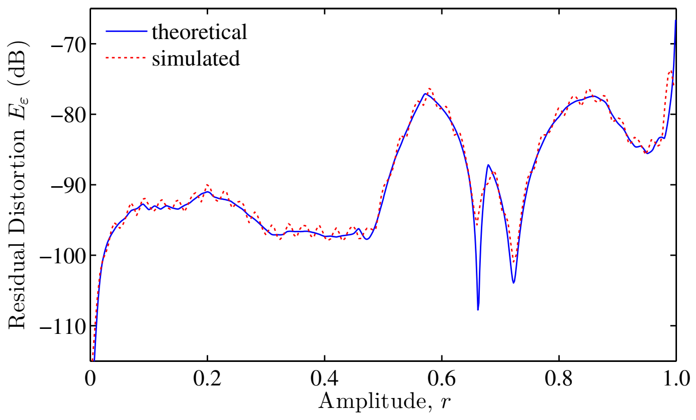


Figure 4.3: Real and imaginary parts of a class-E amplifier nonlinearity expressed as a complex-gain, over different temperature settings.

The LUT predistorter is uniformly spaced in this exercise, i.e., $c(r) = r$. As seen in Figure 4.4(a), the theoretical approximation of the total residual distortion closely matches the simulated results. The difference between the two curves is 1.4 dB for an LUT size of eight entries and decreases to 0.06 dB as the LUT size is increased to 64 entries. It can also be verified that the residual distortion decreases by about 12 dB when the LUT size is doubled (e.g., from 32 to 64).



(a)



(b)

Figure 4.4: Distortion characteristic of a class-E amplifier. (a) Residual distortion vs. LUT size. (b) Distribution of residual distortion across normalized amplitude for an LUT of size 64.

The distribution of the residual distortion across the amplitude range is critical to finding the optimal compander. To minimize the effect of approximation errors, the density of the LUT entries should be proportional to this error distribution. The accuracy of the theoretical expression in (4.6) is also illustrated in Figure 4.4(b) for an LUT size of 64 entries. A very close match is observed between the theoretically approximated and experimentally determined error distributions.

4.2 Optimal LUT Spacing

The optimal spacing of an LUT predistorter can be derived by finding the compander $c(r)$ that minimizes the total residual distortion (4.7), subject to the constraints $c(0) = 0$ and $c(1) = 1$. This is achieved by minimizing the following Lagrangian cost function:

$$J = \frac{\alpha}{N^4} \int_0^1 \frac{w(r)}{c'^4(r)}\, dr + \lambda \int_0^1 [c'(r) - 1]\, dr,$$

with

$$w(r) = |\psi(r)|^2\, p(r), \tag{4.8}$$

where $\psi(r)$ is defined in (4.5). Setting the derivative of the cost function J with respect to $c'(r)$ to zero, while exercising the above constraints, results in the derivative of the optimal compander given by

$$c'_{\text{opt}}(r) = \frac{w^{\frac{1}{5}}(r)}{\int_0^1 w^{\frac{1}{5}}(r)\, dr}.$$

The optimal compander is found by computing the integral of the previous equation. The resulting function depends on the amplifier's nonlinear characteristics, the predistorter, and the probability distribution of the input signal amplitude. It is independent of the LUT size, as expected.

The resulting residual distortion is obtained by substituting $c'(r)$ in (4.7):

$$P_{\text{opt}} = \left[\int_0^1 w^{\frac{1}{5}}(r)\, dr\right]^5. \tag{4.9}$$

When uniform spacing is used, $c'(r) = 1$. In this case, from (4.8), the residual distortion (4.7) is simply given by

$$P_{\text{uni}} = \int_0^1 w(r)\, dr. \tag{4.10}$$

The total reduction in residual distortion resulting from the use of the optimal spacing is computed by taking the ratio of (4.10) and (4.9):

$$\delta P = \frac{\int_0^1 w(r)\, dr}{\left[\int_0^1 w^{\frac{1}{5}}(r)\, dr\right]^5}. \tag{4.11}$$

4.2.1 Performance Evaluation with Optimal LUT Spacing

To assess the performance improvement of the optimal spacing scheme, three different types of signals with different distributions and varying bandwidths have been chosen: a two-tone signal, an EDGE-modulated signal, and a WCDMA-modulated signal. The block diagram in Figure 4.5 illustrates the simulation setup. The amplifier model is based on the extracted AM-AM and AM-PM characteristics of the class-E PA previously shown in Figure 4.3.

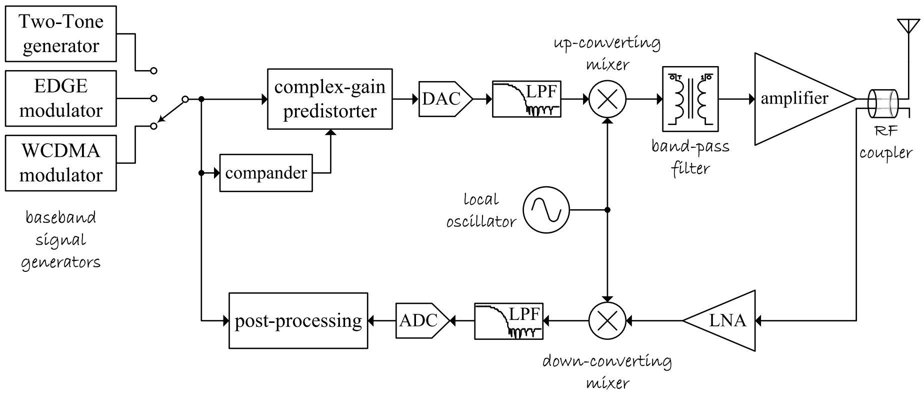


Figure 4.5: Simulation setup for measuring the predistorter performance, with optimal LUT spacing.

The envelope probability densities of the three test signals are shown in Figure 4.6.

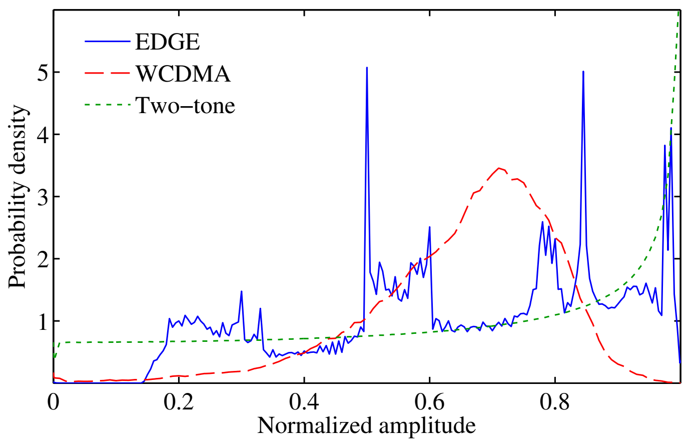


Figure 4.6: Estimated envelope probability densities of test signals.

It can be seen from Figure 4.6 that the three chosen signals have quite different envelope probability densities and therefore constitute an adequately diverse set to evaluate the potential benefits of optimal LUT spacing.

4.2.1.1 Two-tone Signal

The two-tone test has traditionally been heavily used by RF engineers to assess the linearity of power amplifiers. This is because of the relative ease of test setup and the simplicity of image and cross-product frequency measurements needed to estimate the underlying nonlinearity. Procedurally, when two closely-spaced RF tones with frequencies f_1 and f_2 are transmitted through a device with a kth-order nonlinearity, spurious inter-modulation (IMD) products are generated at frequencies $mf_1 \pm nf_2$, with $m + n = k$. The odd-ordered nonlinear terms generate IMD products that are difficult to filter because of their close spectral-proximity to the input-tone frequencies. An ideal predistorter would reduce the IMD tones to a level below a desired spectral noise floor. But when the same two-tone signal is used as an input to the predistorted transmitter of Figure 4.1, the LUT approximation errors will result in the amplification of some residual IMD tones. In this exercise, the IMD tone levels are measured in the case of uniform and optimal spacing of the linearly-interpolated LUT. The predistorter LUT size is set to 24 entries. The output spectrum of the two-tone simulation is shown in Figure 4.7.

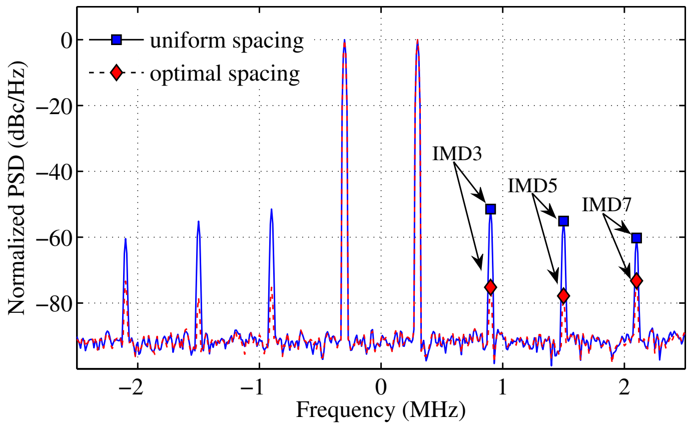


Figure 4.7: Two-tone IMD products for uniform and optimal LUT spacing.

This experiment shows that the use of optimal spacing results in improved rejection of the spurious IMD products. The third, fifth, and seventh-order IMD products are respectively improved by 22.3 dB, 22.7 dB, and 12.4 dB.

4.2.1.2 EDGE and WCDMA Modulated Signals

Practically, a cellular power amplifier is intended to transmit a modulated carrier. Therefore, the optimal LUT spacing has been evaluated with both WCDMA and EDGE modulated signals. The nonlinearity in the transmit path manifests itself through both in-band distortion, measured by error vector magnitude (EVM), and out-of-band spectral regrowth, which degrades the modulated spectrum and causes interference with adjacent channels. The signal EVMs resulting from the use of uniform and optimal spacing, at varying LUT sizes, are shown in Figure 4.8(a) and Figure 4.8(b) for the EDGE and WCDMA signals, respectively.

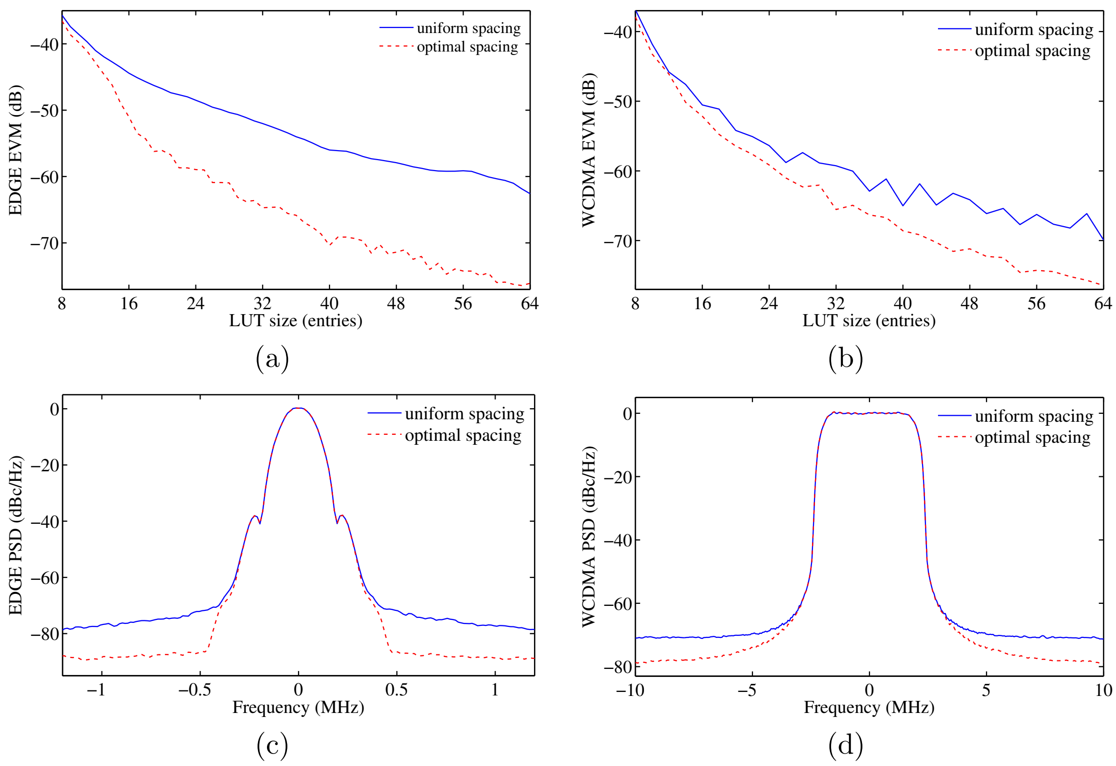


Figure 4.8: EVM as a function of LUT size N and PSD for $N = 40$, using EDGE and WCDMA modulated signals with uniform and optimal LUT spacing. (a) EDGE EVM. (b) WCDMA EVM. (c) EDGE PSD. (d) WCDMA PSD.

These results show that the benefits of optimal spacing are limited when the LUT size is very small (say below 10 entries). But the gap quickly widens as the LUT size is increased above 16 entries. This is expected, as the derivation of optimal spacing used the assumption of closely spaced entries.

For an LUT size of 64 entries, the optimal spacing improves the EVM by 13.5 dB for the EDGE signal, and by 6.5 dB in the case of the WCDMA signal. The power spectral densities (PSDs) for an LUT size of 40 entries are also shown in Figure 4.8(c) and Figure 4.8(d), for the EDGE and WCDMA signals. The use of optimal LUT spacing lowers their respective spectral floors by more than 10 dB and 8 dB.

Apparently, the use of optimal spacing has slightly higher impact in the case of the EDGE-modulated signal. It is clear from (4.11) that the attenuation of residual distortion resulting from optimal spacing depends on the combination of the amplifier nonlinearity, the predistorter, and the probability density of the input signal. As a result, for a given set of nonlinear characteristics, the measured impact of the optimal LUT spacing will depend on the statistics of the signal under consideration. Therefore, in a multi-standard transceiver system (supporting different modulation schemes), the LUT spacing may need to be programmable to achieve optimal performance for each supported class of signals. Furthermore, the envelope probability density of a WCDMA signal, for example, strongly depends on the number of simultaneous channels being transmitted. In this case, it might be impractical to derive and store an optimal compander for each possible probability density profile. An alternative solution would be to either use a dynamically adaptive LUT spacing such as in [35], or design a uniformly-spaced LUT with a sufficient number of entries to guaranty acceptable performance across all supported signal types.

4.2.2 Interpolated vs. Non-Interpolated LUT Predistorter

The optimal compander previously derived for LIN-LUTs is not optimal for non-interpolated LUTs (ZOH-LUT). Using the optimal compander derived for a ZOH-LUT in a LIN-LUT, or vice versa, would actually worsen the performance with respect to uniform spacing.

The optimal spacing for a ZOH-LUT predistorter is derived in [11]. In this case, the optimal compander is given by

$$c_{\text{opt}}(r) = \frac{1}{\int_0^1 w^{\frac{1}{3}}(r)\,dr} \int w^{\frac{1}{3}}(r)\,dr, \tag{4.12}$$

where $w(r)$ is a function of the predistorter $f(\cdot)$, the amplifier's nonlinear characteristics

$g(\cdot)$ and the probability density of the input signal amplitude $p(r)$:

$$w(r) = \frac{\left|g'[r|f(r)|]\right|^2}{\left|g[r|f(r)|]\right|^4} r^2 p(r).$$

The use of a ZOH-LUT results in an approximation error dominated by the magnitude of the first-order derivative of the complex-gain $f(\cdot)$ [11]. If a uniformly-distributed signal is considered, the optimal compander roughly makes the density of entries proportional to the magnitude of the first-order derivative. On the other hand, the LIN-LUT generates an error that is proportional to the second-order derivative. Consequently, its optimal compander will tend to increase the density of entries in the regions of higher magnitude of the second-order derivative of the complex-gain $f(\cdot)$.

To achieve a fair comparison, the impact of optimal spacing is measured for both interpolated and non-interpolated LUT predistorters, using the same class-E nonlinear characteristics and the EDGE-modulated test signal. The results in Figure 4.9 show that the optimal spacing of ZOH-LUT improves the EVM by 4 dB with respect to uniform spacing, supporting the results obtained in [11].

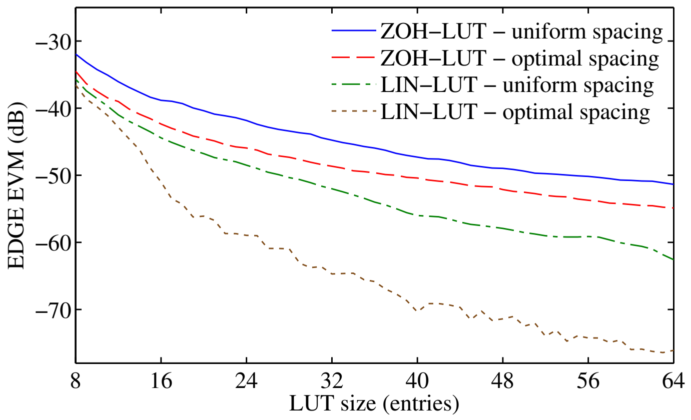


Figure 4.9: Impact of optimal spacing on EDGE EVM using linearly-interpolated (Lin) and non-interpolated (ZOH) LUT predistorters.

Under the same operating conditions, the optimal spacing of the LIN-LUT predistorter improves the EVM by at least 15 dB for LUT sizes larger than 50 entries. This result shows that the impact of optimal spacing is significantly higher when linear interpolation is used.

The combination of linear interpolation and optimal spacing reduces the residual EVM to below $-60\,\text{dB}$ for an LUT size of only 20 entries. The size of a non-interpolated LUT must be increased to 200 entries to achieve the same level of residual distortion. In addition to the savings in required memory space, the smaller LUT size allows faster convergence during adaptive training of the predistorter.

4.2.3 Effect of Operating Temperature

The nonlinear characteristics of power amplifiers vary with the operating temperature, especially in the case of modern 3G and 4G duplex modulation schemes, where the temperature can change drastically during active operation of the device. It is therefore important to determine the sensitivity of the optimal compander to temperature changes. The extracted characteristics of the class-E amplifier were used across a wide range of temperature settings. A sample of nonlinearity curves and their variation at extreme temperatures has been previously shown in Figure 4.3, where the real and imaginary parts of the amplifier's complex gain are shown for three different temperatures: $-35°$C, $25°$C, and $105°$C. To assess the sensitivity of optimal spacing to temperature variations, the optimal compander is computed using the characteristics measured at a nominal temperature of $25°$C, and then the temperature of operation is varied across the entire operational range. Figure 4.10 shows the incremental EVM improvement resulting from the use of the nominal compander across temperature for LUT sizes of 32 and 128 entries.

For an LUT size of 128 entries, the EVM improvement quickly drops from 18 dB to 9 dB as the temperature changes by $\pm 10°$C. The EVM improvement is less sensitive to temperature for an LUT size of 32 entries. It is important to note that the LUT with 32 entries is sufficient to reduce the EVM to less than $-60\,\text{dB}$. In this case the single, nominal optimal compander can maintain the EVM improvement above 10 dB over a wide range of operating temperatures.

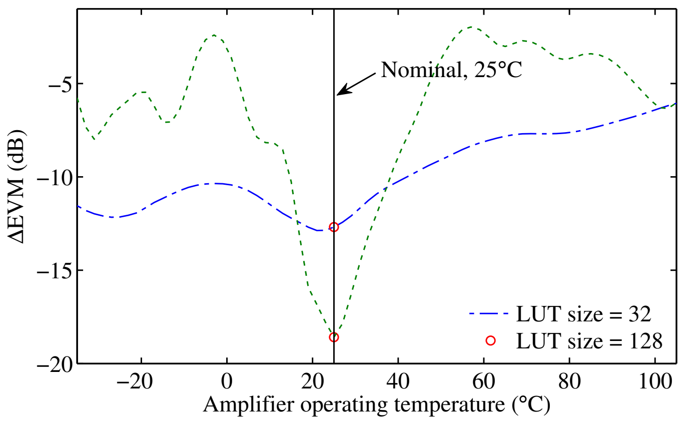


Figure 4.10: Impact of temperature on EVM of fixed, nominally-derived optimal compander.

4.3 Complex Companding

In Section 4.2, we derived an optimal compander that maps the input amplitude domain to the address of the complex-gain LUT predistorter. In practice, the complex-gain predistorter is implemented by a parallel combination of two sub-tables representing the real and imaginary parts of the complex gain. However, both the real and imaginary tables share the same lookup address and consequently have the same spacing scheme.

If a uniformly-distributed input signal is considered, a first-order approximation of the optimal spacing of a real-valued LUT consists of having a density of entries that is proportional to the second-order derivative of the curve. It can therefore be intuitively assumed that, if the shapes of the real and imaginary parts of the complex-gain curve possess significantly different characteristics, the residual distortion could be better reduced by individually optimizing the spacing of the two tables. This could be achieved by using two separate companders that jointly optimize the spacing of the two tables, as illustrated in Figure 4.11. This approach is termed *complex companding* in this work. Finding two jointly-optimal companders is mathematically cumbersome. A simplified approach is to find two separate companders that minimize the individual error contribution of the two tables.

If the accuracy of the imaginary-gain table is assumed to be perfect, then the linear approximation error in (4.2) is purely real and is given by

$$\varepsilon_{\mathrm{re}} = \frac{\varepsilon_r(\varepsilon_r - d_{\mathrm{re}})}{2} \Re[f''(r_{\mathrm{o}})] \,.$$

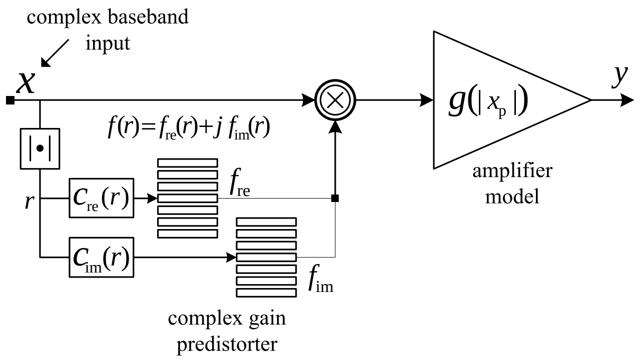


Figure 4.11: Predistorter LUT implementation using a complex compander.

In the previous equation, d_{re} is the width of the current bin of the real-gain table. Following the same procedure developed in Section 4.2, the error at the output of the transmitter is given by

$$\varepsilon_y = \frac{\varepsilon_r(d_{\mathrm{re}} - \varepsilon_r)}{2} \psi_{\mathrm{re}}(r),$$

where

$$\psi_{\mathrm{re}}(r) = \frac{f^*(r) - j\, r\, \Im[f'(r)]}{|f(r)|^2 + r\, \Re[f^*(r) f'(r)]} \Re[f''(r)] x. \tag{4.13}$$

From the above equation and the derivation in Section 4.2, the optimal compander for the real-gain table is given by

$$c'_{\mathrm{re}}(r) = \frac{w_{\mathrm{re}}^{\frac{1}{5}}(r)}{\int_0^1 w_{\mathrm{re}}^{\frac{1}{5}}(r)\, dr}, \tag{4.14}$$

with

$$w_{\mathrm{re}} = \left| \frac{f^*(r) - j\, r\, \Im[f'(r)]}{|f(r)|^2 + r\, \Re[f^*(r) f'(r)]} \Re[f''(r)] x \right|^2 p(r).$$

Similarly, if the accuracy of the real-gain table is assumed to be perfect, then the optimal compander for the imaginary-gain table is given by

$$c'_{\mathrm{im}}(r) = \frac{w_{\mathrm{im}}^{\frac{1}{5}}(r)}{\int_0^1 w_{\mathrm{im}}^{\frac{1}{5}}(r)\, dr}, \tag{4.15}$$

with

$$w_{\mathrm{im}} = \left| \frac{f^*(r) + r\,\Re[f'(r)]}{|f(r)|^2 + r\,\Re[f^*(r)f'(r)]} \Im[f''(r)]\,x \right|^2 p(r).$$

To assess the effectiveness of the complex companding method, the resulting residual distortion is compared to the previously obtained results using the unified optimal compander. Figure 4.12 shows that the EVM improvement resulting from the use of complex companding is relatively small even with a highly-compressed memoryless class-E amplifier. However, the difference between the two approaches may turn out to be more significant for other classes of high-power amplifiers.

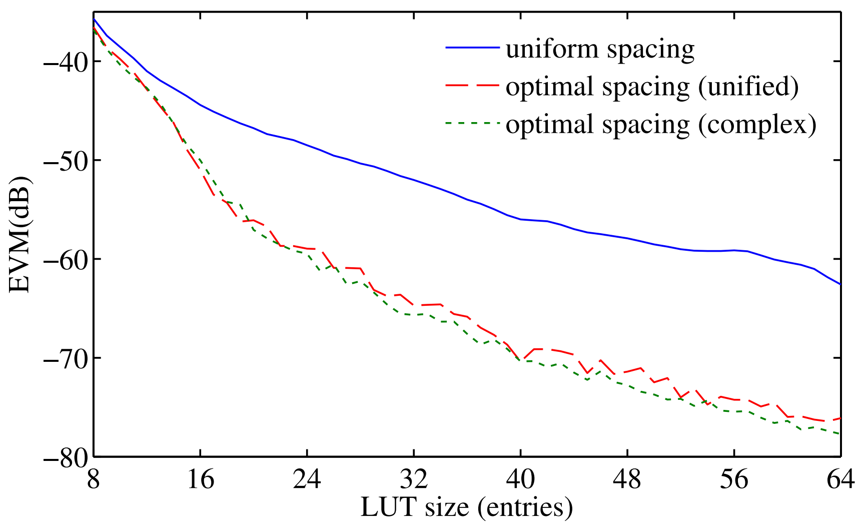


Figure 4.12: Performance comparison of complex and unified companding schemes.

4.4 Physical Implementation of LUT Spacing

Once the benefits of optimal spacing have been studied, the next logical step is to factor in the additional complexity associated with the compander implementation. In practice, the compander is itself implemented as an LUT. It is therefore critical that the additional memory requirements do not offset the gains obtained from using optimal spacing. In the following study, the compander is implemented as a uniformly-spaced, linearly-interpolated LUT of size L, forming a piece-wise linear function. Table 4.1 shows the resulting EDGE EVM for different values of L, with the predistorter LUT size set to 24 entries.

Table 4.1: EDGE EVM for different compander LUT size L.

Spacing		**EDGE EVM (dB)**
Uniform		-48.50
Optimal	$L = 8$	-56.31
	$L = 16$	-58.24
	$L = 32$	-58.73
	$L = 64$	-58.87
	$L = \infty$	-58.96

These results show that a compander LUT with as little as eight entries improves the EVM by 8 dB as compared to uniform spacing. Increasing the size (L) to 16 entries further reduces the EVM by an additional 2 dB. To design a uniformly-spaced LUT with similar performance, the number of entries must be increased to at least 128. In this comparison, it is also important to note that the entries of the predistorter LUT are complex, whereas the compander LUT is real. These two configurations (optimally spaced LUT with 32 entries and uniformly spaced LUT with 128 entries) were implemented in VHDL and synthesized with the Synopsys Design Compiler. The resulting gate counts are summarized in Table 4.2.

Table 4.2: Gate count resulting from hardware synthesis.

		Uniform spacing[a]	**Optimal spacing**[b]
LUT memory		$11,883$	$2,997$
Predistorter logic		$6,663$	$5,241$
Compander	**memory**	—	775
	logic	—	456
Total gate count		$18,546$	$9,469$

[a] Using a single linearly-interpolated complex-gain LUT of size 128.
[b] Using a linearly-interpolated complex-gain LUT of size 32 with a 16-entry compander LUT.

This design uses an amplitude approximation method similar to the one presented in [23]. The results in Table 4.2 show that, in this example, the use of optimal spacing reduces the total predistorter gate count by half, despite the additional memory and logic used to implement the optimal compander. This reduction in the total gate count directly translates into both area and power reduction.

A predistorter adaptation block based on the indirect-learning architecture [53] was also synthesized. The adaptation block trains a replica of the feedforward predistorter using an LMS-like algorithm. The gate count of the adaptation block is about 21 kgates for the uniformly-spaced predistorter and 12 kgates for the optimally-spaced LUT predistorter (this gate count includes replication of the feedforward predistorter). In addition to the area and power savings, the smaller complex-gain LUT size allows faster convergence of this LUT training algorithm. Figure 4.13 illustrates the error convergence of the adaptation algorithm for the two cases. The results show that the smaller, optimally-spaced LUT predistorter converges four times faster than the larger, uniformly-spaced one.

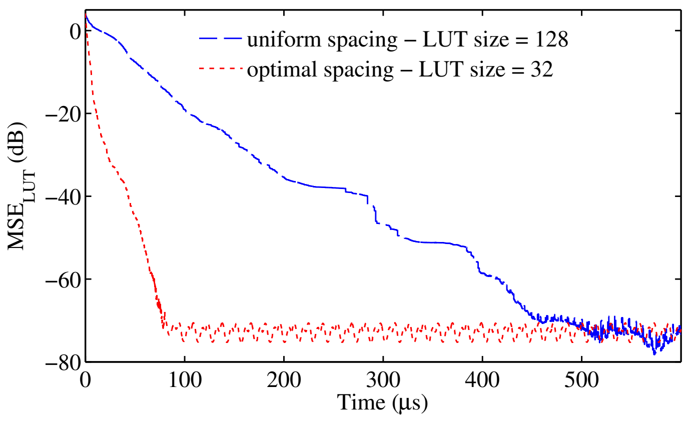


Figure 4.13: Training convergence of uniformly and optimally spaced LUT predistorters.

It is important to note that the compander must be reloaded with adequate values for every signal envelope probability density. The envelope probability density of a WCDMA signal, for example, strongly depends on the number of simultaneous channels being transmitted. In this case, it might be impractical to derive and externally store an optimal LUT spacing for each possible probability density profile. To circumvent this issue, a signal-independent compander can be derived by adapting the approach outlined in [6] to the LIN-LUT. If memory size and convergence speed are not critical for the design, a simpler approach is to use uniform spacing and increase the LUT size to meet the same performance.

4.5 Conclusion

In this chapter, the optimal spacing of a linearly-interpolated LUT predistorter has been studied. The theoretically-derived optimal compander is a function of the probability density of the input signal and the nonlinear characteristics of the power amplifier. The theoretical results have been validated through simulations, using the extracted nonlinear characteristics of a class-E amplifier at different temperature settings. A variety of test signals have been used to measure the effectiveness of the proposed optimal spacing method. Error vector magnitude (EVM) improvements of 10 dB to 15 dB were observed, over the use of the simpler, but generally less effective, uniform spacing. The effects of characteristic variations caused by changes in temperature have also been studied. A practical implementation that leads to significant performance improvement has been illustrated. It is demonstrated that optimal spacing is far more beneficial in the case of a linearly-interpolated LUT predistorter than in the case of its non-interpolated counterpart. A practical implementation with significantly reduced gate count and faster convergence of the LUT training algorithms can be achieved, if only a limited number of signal probability densities must be supported.

CHAPTER V

POWER CONTROL AND LUT SPACING

It has been shown in the previous chapters that significant performance improvement can be achieved by using a compander to optimally space the LUT entries. Such a compander depends on the nonlinear characteristics of the amplifier as well as the input signal's statistics. To maximize the battery life of mobile devices and efficiently manage the scarce spectral resources, modern wireless standards implement advanced power control schemes. Power control is often achieved by simply scaling the input signal, which in turn changes its statistics. Therefore, a given compander would only be optimal for a given input backoff setting. It is shown in [11] that a few decibels of power back-off will result in a significant decrease in the performance of the initially optimal LUT spacing. Optimal performance across all power levels would require a redefinition of the LUT spacing and an update of the entries for each power level, which is obviously a very unpractical solution. As the input power is decreased, only a small portion of the LUT is effectively active. This effect is illustrated in Figure 5.1.

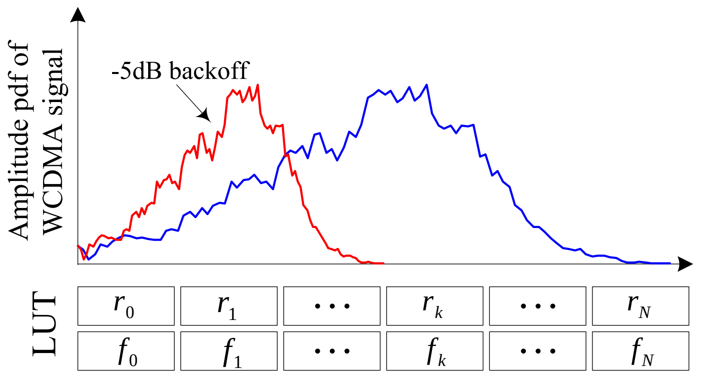


Figure 5.1: Probability density function of the amplitude of a WCDMA backoff by 5dB.

The amplitude of the input WCDMA signal backed off by 5 dB covers less than half of the LUT entries. This amounts to using a smaller LUT size as the input signal is backed off.

Therefore, the distortions resulting from larger LUT approximation errors will decrease the SNR and reduce the predistorter's performance unless the LUT size is greatly increased.

5.1 Constant-SNR Spacing

To prevent a huge increase in memory space requirements, a non-uniform spacing scheme for linearly-interpolated LUTs can be designed to yield constant SNR across the entire supported power dynamic range. This will allow the use of a single low-resolution, fixed-range LUT predistorter to provide adequate correction across a relatively large range of input power backoff. The *μ-law* compander [24] was developed to achieve such a goal for the quantization of speech signals. The *μ-law* compander is mathematically given by

$$c_\mu(r) = \frac{\ln(1+\mu\, r)}{\ln(1+\mu)}. \tag{5.1}$$

For speech signal quantization, μ is often chosen to be equal to 256 (eight bits). But for the polar predistorter, our experiments show that a value between 32 and 64 provides a better performance balance across an input backoff dynamic range of up to 40 dB. For the complex-gain predistorter, μ should be set to a value between 8 and 16. The resulting spacing $d(r)$ can be derived from equation (3.9):

$$\begin{aligned} d(r) &\approx \frac{1}{N\, c'_\mu(r)} \\ &\approx \frac{1\,\mu}{N\,\ln(1+\mu)}\,(1+\mu\, r). \end{aligned} \tag{5.2}$$

Note that the spacing between LUT entries increases linearly with the input amplitude. This effect improves the performance when the input signal is backed off by providing an even better predistorter resolution at low amplitudes.

5.1.1 Constant-SNR Spacing in Polar Predistorters

In Chapter 3, an optimal spacing scheme for polar LUT predistorters was derived for a given input backoff setting. Using those previously obtained results, a compander that yields a constant SNR across all amplitude regions can be derived. From (3.7), the SNR of the kth interval of the amplitude table can be approximated by

$$\begin{aligned}\mathrm{SNR}_\rho(r) &= \frac{r^2}{E({e_r}^2)} \\ &= \frac{120\,N^4\,r^2\,c_\rho'(r)^4\,f'(r_k)^2}{V^4\,f''(r_k)^2}.\end{aligned} \tag{5.3}$$

Using equation (5.3), a constant SNR across all amplitudes can only be obtained if

$$c_\rho'(r) = \lambda_\rho \left|\frac{f_\rho''(r)}{r f_\rho'(r)}\right|^{1/2}, \tag{5.4}$$

where λ_ρ is a constant. Similarly for the phase LUT, the SNR is constant across amplitudes if

$$c_\theta'(r) = \lambda_\theta \left|\frac{f_\theta''(r)}{r}\right|^{1/2}. \tag{5.5}$$

The corresponding amplitude and phase LUT companders are given below:

$$c_\rho(r) = \lambda_\rho \int \left|\frac{f_\rho''(r)}{r f_\rho'(r)}\right|^{1/2} dr \qquad \text{and} \qquad c_\theta(r) = \lambda_\theta \int \left|\frac{f_\theta''(r)}{r}\right|^{1/2} dr. \tag{5.6}$$

λ_ρ and λ_θ are chosen such that $c_\rho(1) = 1$ and $c_\theta(1) = 1$. Practically, c_ρ and c_θ can be numerically estimated.

5.1.2 Constant-SNR Spacing in Complex-Gain Predistorters

The optimal compander for a linearly-interpolated complex-gain predistorter was derived in Chapter 4. Those previously obtained results can be used to derive a constant-SNR spacing for the complex-gain predistorter. The individual bin contribution to the overall residual distortion was given by (4.6) as

$$E_\varepsilon = \alpha\, d^4\, |\psi(r)|^2,$$

where

$$\alpha = \frac{1}{120} \qquad \text{and} \qquad \psi(r) = \frac{f''(r)f^*(r) + j\,r\,\Im[f'^*(r)f''(r)]}{|f(r)|^2 + r\,\Re[f^*(r)f'(r)]}x.$$

Using the relation between the bin spacing and the compander $d(r) = \dfrac{1}{Nc'(r)}$, the signal to noise ration across the bin can be approximated by

$$\mathrm{SNR(r)} \approx \frac{N^4 r^2 c'^4(r)}{\alpha|\psi(r)|^2}.$$

The constant SNR condition is satisfied if

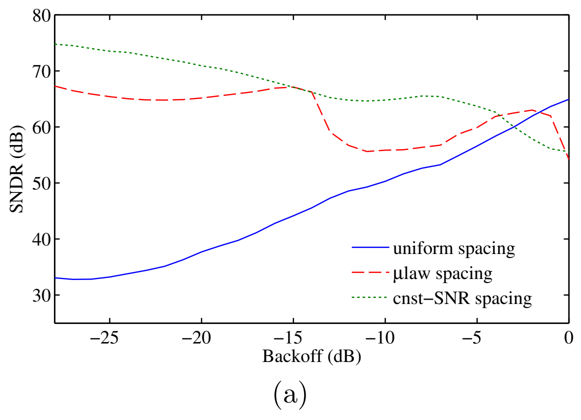
$$c'(r) = \lambda \left| \frac{\psi(r)}{r} \right|^{1/2}.$$

The constant-SNR compander is given by

$$c(r) = \lambda \int \left| \frac{\psi(r)}{r} \right|^{1/2} dr,$$

where λ is chosen so that $c(1) = 1$.

Simulation experiments were carried out using the amplifier characteristics previously shown in Figure 3.5. The input signal backoff is varied from zero to -30 dB. The signal to noise and distortion ratio (SNDR) is measured for uniform, μ-law, and constant-SNR spacings. The experiment was carried out for both the polar and complex-gain predistorter configurations and the results are shown in Figure 5.2(a) and Figure 5.2(b), respectively.



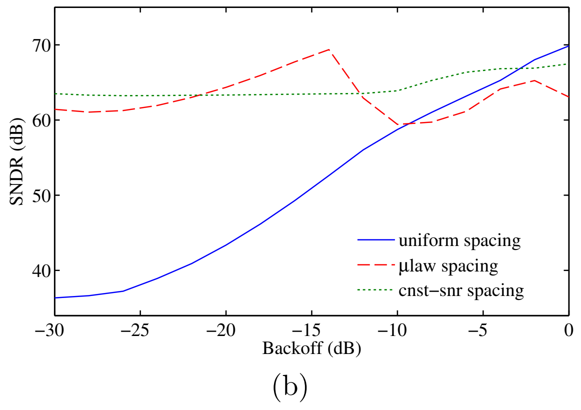


Figure 5.2: Signal to noise and distortion ration (SNDR) vs. input power backoff for uniform, μ-law and constant-SNR spacings. (a) Polar predistorter. (b) Complex-gain predistorter.

The μ-law spacing parameter is set to 64 in the polar configuration and 16 in the complex-gain predistorter. These values are set to balance the performance between the low and high amplitude regions. In both experiments the LUT size is set to 64 entries. It can be observed that the proposed constant-SNR spacing results in a balanced performance over all power levels as opposed to the uniform spacing, which suffers severe performance degradation at large input power backoffs. The simpler μ-law compander also improves the supported dynamic range but is less effective than the constant-SNR spacing in the case of the polar predistorter. The performance of the μ-law spacing approaches the constant-SNR spacing when the LUT size is further increased. The main advantage of the μ-law spacing is its independence with respect to signal statistics and amplifier nonlinear characteristics, whereas the constant-SNR spacing must be updated for signals with different probability densities.

In practical implementations, the compander itself must be approximated with a lookup table, adding additional memory requirements to the design. To linearly interpolate the LUT predistorter, the inverse of the bin width must be calculated. For both the μ-law and constant-SNR spacings, computing the inverse of the bin width on the fly is not practical for hardware implementation. Therefore, the inverse bin width must be pre-calculated and stored. This further increases the memory requirements. To circumvent these issues and minimize the memory requirements, an alternative approach that is well suited to practical hardware implementation is presented in Section 5.2.

5.2 Low-Complexity LUT Spacing

The μ-law spacing results in good predistorter performance across a large range of input power backoffs. It is independent from signal statistics unlike the theoretically derived constant-SNR spacing, which has better performance. Meanwhile, the computational complexity of the ideal μ-law address calculation makes it difficult to meet the high speed requirements of 3G and 4G transceivers. An alternative low complexity spacing scheme is therefore proposed in this section. This spacing scheme is designed to meet the following requirements:

- Closely approach the dynamic range and performance of the μ-law spacing.
- Provide low-complexity, fast address calculation that is easily amenable to hardware implementation.
- Simplify the computation of the linear interpolation.

This is achieved by approximating the μ-law spacing with a base-2 logarithm instead of the natural logarithm. For this reason the proposed method will termed *B2 spacing*. For an LUT size L, the proposed spacing scheme divides the signal range into N intervals with exponentially increasing width. For simplicity, each interval contains exactly M uniformly spaced entries. It is also assumed that the intervals are numbered from low to high amplitudes. Let $W_k = \alpha 2^k$ be the width of the kth interval with α a real constant and $k \in \{0, 1, \cdots, N-1\}$. The width of the intervals increases exponentially with k. The number of entries per interval is $M = L/N$.

L, M and N should be powers of two to simplify the implementation, but this is not a necessary condition. Let us consider an amplitude resolution of 12 bits. Let us also assume an LUT of size $L = 128$ is used and N and M are set to 8 and 16, respectively. The binary representation of the amplitude signal is $\mathrm{b_{11}b_{10}b_9b_8b_7b_6b_5b_4b_3b_2b_1b_0}$. The LUT is addressed by $\log_2(L) = 7$ address bits: $\mathrm{a_6a_5a_4a_3a_2a_1a_0}$. The B2 spacing nonlinearly maps the amplitude bits $\mathrm{b_n}$ to the address bits $\mathrm{a_m}$. The proposed mapping is illustrated in Figure 5.3.

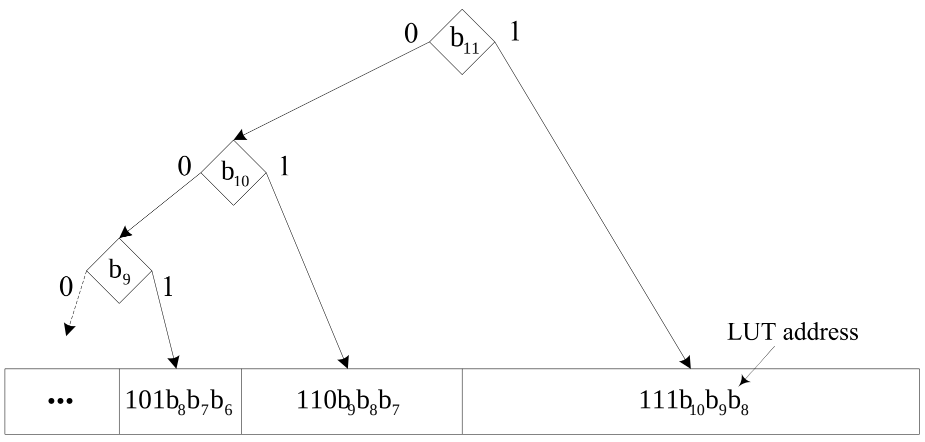


Figure 5.3: Mapping of address bits in B2 spacing.

The position of the most significant non-zero bit (if any) among the $N - 1 = 7$ most significant bits (MSB) is used to select the corresponding interval, and consequently determine the first $\log_2(n) = 3$ bits of the LUT address $\mathrm{a}_6\mathrm{a}_5\mathrm{a}_4$. The remaining address bits $\mathrm{a}_3\mathrm{a}_2\mathrm{a}_1\mathrm{a}_0$ are equal to the amplitude bits immediately following the most significant non-zero bit among the seven MSBs. If the first seven MSBs are all equal to zero, then the remaining address bits are equal to the amplitude bits immediately following the first seven MSBs: $\mathrm{a}_3\mathrm{a}_2\mathrm{a}_1\mathrm{a}_0 = \mathrm{b}_4\mathrm{b}_3\mathrm{b}_2\mathrm{b}_1$. This will result in the first two intervals having equal width. Thereafter, the interval width will increase as a power of two. A logical truth table for the address calculation is shown in Table 5.1.

Table 5.1: Practical implementation of optimal spacing.

Amplitude bits												Address bits						
b_{11}	b_{10}	b_9	b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	a_6	a_5	a_4	a_3	a_2	a_1	a_0
1	–	–	–	–	–	–	–	–	–	–	–	1	1	1	b_{10}	b_9	b_8	b_7
0	1	–	–	–	–	–	–	–	–	–	–	1	1	0	b_9	b_8	b_7	b_6
0	0	1	–	–	–	–	–	–	–	–	–	1	0	1	b_8	b_7	b_6	b_5
0	0	0	1	–	–	–	–	–	–	–	–	1	0	0	b_7	b_6	b_5	b_4
0	0	0	0	1	–	–	–	–	–	–	–	0	1	1	b_6	b_5	b_4	b_3
0	0	0	0	0	1	–	–	–	–	–	–	0	1	0	b_5	b_4	b_3	b_2
0	0	0	0	0	0	1	–	–	–	–	–	0	0	1	b_4	b_3	b_2	b_1

The calculated address corresponds to the LUT entry immediately below (or equal to) the amplitude value. An example implementation is presented in Figure 5.4.

It is important to note that the spacing between two consecutive entries is always a power of two. This facilitates linear interpolation. In fact the interpolation factor can be readily obtained from the amplitude bits. Table 5.2 shows a truth table that generates a five-bit interpolation factor $\mathrm{c}_4\mathrm{c}_3\mathrm{c}_2\mathrm{c}_1\mathrm{c}_0$. The address and interpolation factor calculation circuits were implemented in VHDL and synthesized with the SYNOPSYS DESIGN COMPILER. The synthesis resulted in a total of just 141 nand2-equivalent gates for an amplitude resolution of 16 bits and an interpolation factor of 6 bits. The number of intervals was set to $N = 8$ and the number of entries per interval was set to $M = 16$. The VHDL code for the B2 address calculation is provided in Appendix B.

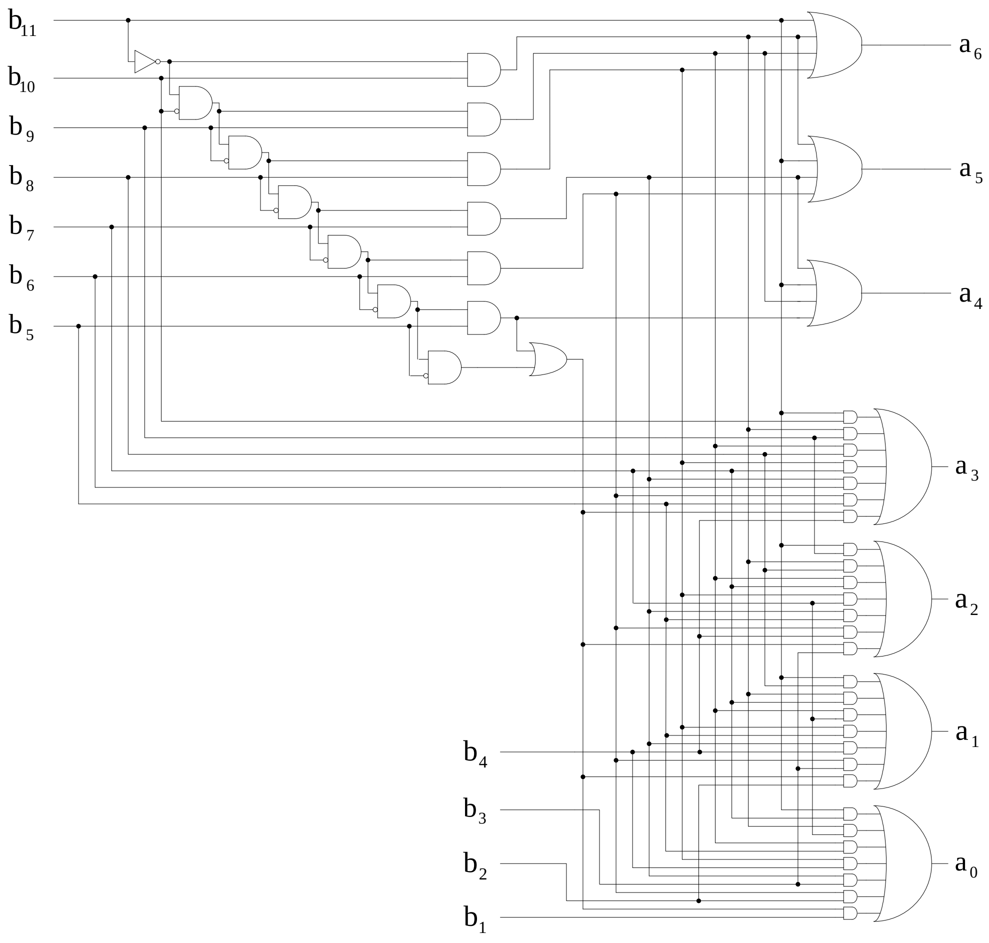


Figure 5.4: B2 spacing address calculation circuit.

The performance of the B2 spacing was also simulated and compared to the uniform and μ-law spacings. A complex-gain predistorter is used with a WCDMA signal as input. The results are shown in Figure 5.5.

These results show that the B2 spacing closely approaches the performance of the μ-law spacing while allowing simple low-complexity hardware implementation. The performance of uniform spacing decreases almost linearly with the input power backoff. The B2 spacing maintains good predistorter performance even with a $-30\,\text{dB}$ backoff. The B2 spacing example presented here uses equal number of entries M for each interval. In general the number of intervals N and the number of entries in each interval M_k should be optimized with respect to the considered nonlinear characteristics, the target dynamic range, and the LUT size to provide the best performance across the supported range of power backoff.

Table 5.2: Practical implementation of optimal spacing.

Amplitude bits												Interpolation factor				
b_{11}	b_{10}	b_9	b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	c_4	c_3	c_2	c_1	c_0
1	–	–	–	–	–	–	–	–	–	–	–	b_6	b_5	b_4	b_3	b_2
0	1	–	–	–	–	–	–	–	–	–	–	b_5	b_4	b_3	b_2	b_1
0	0	1	–	–	–	–	–	–	–	–	–	b_4	b_3	b_2	b_1	b_0
0	0	0	1	–	–	–	–	–	–	–	–	b_3	b_2	b_1	b_0	0
0	0	0	0	1	–	–	–	–	–	–	–	b_2	b_1	b_0	0	0
0	0	0	0	0	1	–	–	–	–	–	–	b_1	b_0	0	0	0
0	0	0	0	0	0	1	–	–	–	–	–	b_0	0	0	0	0

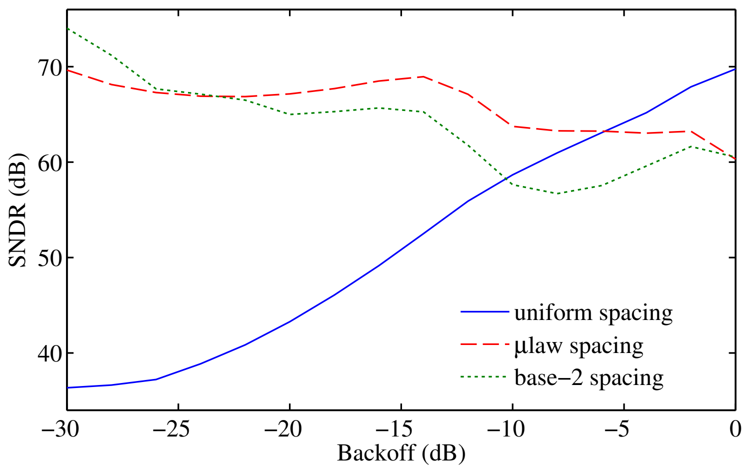


Figure 5.5: Signal to noise and distortion ratio (SNDR) across different backoffs for uniform, μ-law, and B2 spacing.

CHAPTER VI

AMPLITUDE APPROXIMATION FOR DIGITAL PREDISTORTERS

The nonlinear gain and phase distortions of RF amplifiers is a function of the input signal's envelop. Consequently, the digital baseband predistorter must also be a function of the amplitude. In the case of the complex-gain LUT predistorter, the amplitude signal's most significant bits (MSBs) are ideally used to address an LUT predistorter. For example, the first seven MSBs are used to address an LUT with 128 entries. The accurate computation of amplitude requires a square-root function, which is not amenable to efficient hardware implementation, especially at very high processing rates. To circumvent this issue, practical digital-baseband predistorters have traditionally been implemented as a function of the instantaneous envelope power I^2+Q^2. The resulting, often unintended effect is a concentration of the LUT entries around the higher amplitude region [11]. This power indexing scheme is suitable for class-A and mild class-AB amplifiers since their characteristics are mostly linear until close to saturation. However, this is not well suited to amplifiers with higher power efficiency, such as deep class-AB, class-B, C, E, etc., which exhibit significant nonlinear amplitude and phase distortions across the entire amplitude range.

Furthermore, if a certain level of digital power backoff must be supported (e.g. for the purpose of adaptive power control), the power indexing scheme will require a significantly larger LUT size to meet the same performance as the amplitude-indexed LUT. For example, when the input signal is backed off by 6 dB, the signal range covers half of the entries in the amplitude-indexed LUT, but only one quarter of the entries in the power-indexed LUT. The latter will therefore result in a coarser quantization of the predistorter and higher levels of residual distortion. This effect is illustrated in Table 6.1. These results are obtained using a complex-gain LUT with 64 entries to predistort a class-E amplifier. The amplitude indexing is shown to consistently outperform the power indexing scheme for all backoff settings and the performance gap increases as the input signal is further backed off.

Table 6.1: WCDMA EVM and ACLR for amplitude and power indexing.

Backoff (dB)	**Indexing**	**EVM** (dB)	**ACLR1** (dBc/Hz)	**ACLR2** (dBc/Hz)
0.0	Amplitude	-66.94	-70.37	-74.93
	Power	-50.90	-59.83	-60.46
3.0	Amplitude	-63.60	-68.99	-71.68
	Power	-44.71	-53.60	-53.86
6.0	Amplitude	-60.42	-67.09	-68.55
	Power	-38.28	-46.88	-48.09

6.1 A Low-Complexity Amplitude Approximation

Simple linear amplitude approximation techniques have been extensively studied in the context of radar detection applications [7, 22, 23, 40]. Most of the methods presented result in relatively coarse approximations, even though their precision is within the tolerance of the targeted applications. But since the digital baseband predistorter is located in the direct transmit path, such large amplitude approximation errors would severely limit the performance of the predistorter, resulting in both residual EVM degradation and spectral distortions.

A common general approach to amplitude approximation consists in rotating the complex input $X = I + j\,Q$ so that its phase lies in $[0,\ \frac{\pi}{4}]$, then compute a linear combination of the real and imaginary parts of the rotated signal $Y = I_\mathrm{r} + j\,Q_\mathrm{r}$. The rotated vector Y is given by

$$I_\mathrm{r} = \max\left(|I|, |Q|\right) \qquad Q_\mathrm{r} = \min\left(|I|, |Q|\right). \tag{6.1}$$

It can be easily observed that the magnitude of the rotated vector Y is equal to the magnitude of the initial vector X:

$$\begin{aligned}|Y| &= \sqrt{\left[\max\left(|I|, |Q|\right)\right]^2 + \left[\min\left(|I|, |Q|\right)\right]^2}\\ &= \sqrt{I^2 + Q^2} = |X|.\end{aligned}$$

The approximated amplitude is obtained by a linear combination of the real and imaginary parts of Y:

$$\hat{R} = a\, I_\mathrm{r} + b\, Q_\mathrm{r}.$$

In [23], the approximation accuracy is improved by further dividing the angular interval $[0,\, \frac{\pi}{4}]$ into two intervals, and using two different sets of coefficients $(a_k,\, b_k)$, $k \in \{1,\, 2\}$ that are optimized for their corresponding intervals. The precision of the approximation can be arbitrarily improved by increasing the number of angular intervals N. In the kth angular interval, the amplitude approximation is given by

$$\hat{R} = a_k\, I_\mathrm{r} + b_k\, Q_\mathrm{r}, \qquad \text{if } \theta_{k-1} \le \theta < \theta_k,$$

where $\theta = \arctan\!\left(\frac{Q_\mathrm{r}}{I_\mathrm{r}}\right)$ and θ_k are the threshold angles delimiting the angular intervals ($k \in \{1,\, N\}$, $\theta_0 = 0$ and $\theta_N = \frac{\pi}{4}$). Figure 6.1 illustrates the use of two and three equal angular intervals.

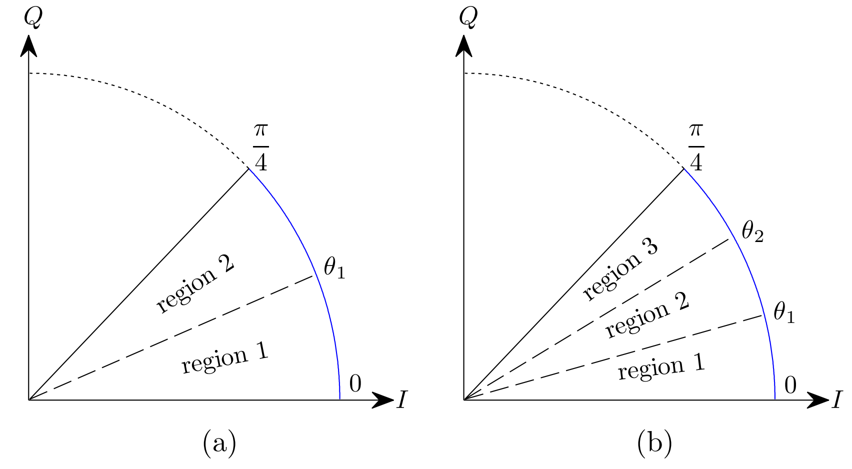


Figure 6.1: Linear amplitude approximations. (a) Two angular intervals. (b) Three angular intervals.

The amplitude error in the kth angular interval is given by

$$\begin{aligned} R - \hat{R} &= R - \big(a_k I_\mathrm{r} + b_k Q_\mathrm{r}\big) \\ &= R - R\big(a_k \cos\theta + b_k \sin\theta\big) \\ &= R\big(1 - a_k \cos\theta - b_k \sin\theta\big). \end{aligned} \tag{6.2}$$

The relative amplitude error ε is given by

$$\varepsilon = 1 - a_k \cos\theta - b_k \sin\theta$$

The coefficients $(a_k,\, b_k)$ must be chosen to minimize a given error metric for each angular interval delimited by the angles θ_{k-1} and θ_k. Assuming that the input angle θ is uniformly distributed, we can obtain a closed-form solution for the coefficients $(a_k,\, b_k)$ that minimizes the mean square of the relative amplitude error ε. The mean squared error $\mathcal{J}$ can be evaluated as follows:

$$\begin{aligned}\mathcal{J} &= E\Big[\varepsilon^2\Big] \\ &= p_0 \int_{\theta_{k-1}}^{\theta_k} \varepsilon^2 \, d\theta. \end{aligned} \tag{6.3}$$

The optimal coefficients are obtained by setting the partial derivatives of $\mathcal{J}$ with respect to the coefficients a_k and b_k to zero. Taking the partial derivative of the mean squared error $\mathcal{J}$ with respect to the coefficient a_k gives

$$\begin{aligned}\frac{\partial \mathcal{J}}{\partial a_k} &= p_0 \int_{\theta_{k-1}}^{\theta_k} \frac{\partial \varepsilon^2}{\partial a_k} \, d\theta \\ &= p_0 \int_{\theta_{k-1}}^{\theta_k} 2\varepsilon \frac{\partial \varepsilon}{\partial a_k} \, d\theta \\ &= 2p_0 \int_{\theta_{k-1}}^{\theta_k} a_k \cos^2\theta + b_k \cos\theta \sin\theta - \cos\theta \, d\theta \\ &= p_0 \int_{\theta_{k-1}}^{\theta_k} a_k(1+\cos 2\theta) + b_k \sin 2\theta - 2\cos\theta \, d\theta \\ &= \frac{p_0}{2}\Big[a_k(2\Delta\theta + \sin 2\theta_k - \sin 2\theta_{k-1}) + b_k(\cos\theta_{k-1} - \cos\theta_k) + 4(\sin\theta_{k-1} - \sin\theta_k)\Big]. \end{aligned} \tag{6.4}$$

Similarly, taking the partial derivative with respect to b_k gives

$$\frac{\partial \mathcal{J}}{\partial b_k} = \frac{p_0}{2}\Big[b_k(2\Delta\theta - \sin 2\theta_k + \sin 2\theta_{k-1}) + a_k(\cos\theta_{k-1} - \cos\theta_k) + 4(\cos\theta_k - \cos\theta_{k-1})\Big],$$

where $\Delta\theta = \theta_k - \theta_{k-1}$. Setting the partial derivatives to zero yields

$$\begin{bmatrix} 2\Delta\theta + \alpha & \beta \\ \beta & 2\Delta\theta - \alpha \end{bmatrix} \begin{bmatrix} a_k \\ b_k \end{bmatrix} = 4 \begin{bmatrix} c_1 \\ c_2 \end{bmatrix},$$

with

$$\begin{aligned} \alpha &= \sin 2\theta_k - \sin 2\theta_{k-1} & c_1 &= \sin\theta_k - \sin\theta_{k-1} \\ \beta &= \cos 2\theta_{k-1} - \cos 2\theta_k & c_2 &= \cos\theta_{k-1} - \cos\theta_k \end{aligned}.$$

The optimal coefficients for the kth angular interval are obtained by solving the above system of linear equations:

$$\begin{bmatrix} a_k \\ b_k \end{bmatrix} = \frac{2}{2\Delta\theta^2 + \cos(2\Delta\theta) - 1} \begin{bmatrix} (2\Delta\theta + \alpha)c_1 - \beta\, c_2 \\ (2\Delta\theta - \alpha)c_2 - \beta\, c_1 \end{bmatrix}. \tag{6.5}$$

For any angular interval delimited by the angles θ_{k-1} and θ_k, the relatively simple closed-form solution (6.5) can be evaluated to find the optimal coefficients (a_k, b_k) in the mean squared error sense. Figure 6.2 shows the mean squared and peak errors as the number of angular intervals is increased from one to eight.

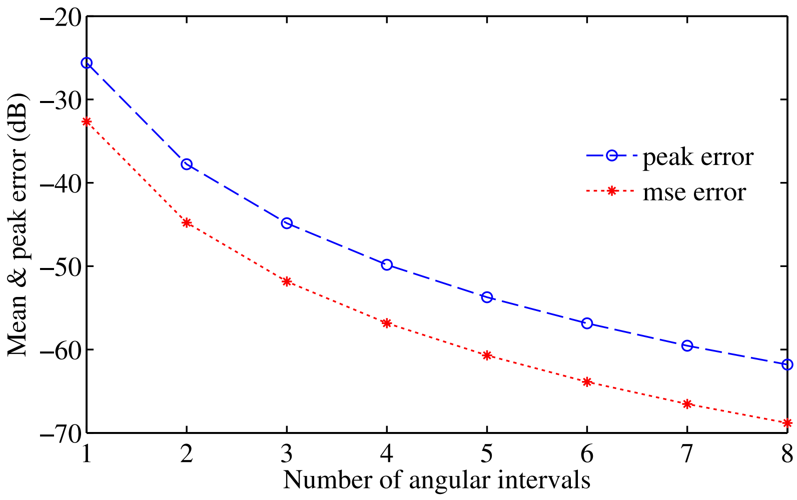


Figure 6.2: Mean squared and peak errors as a function of the number of angular intervals.

These results show that the use of three angular intervals is sufficient to decrease the mean square of the relative amplitude error to below −50 dB. As shown by these results, an arbitrary amplitude approximation accuracy can be achieved by selecting a large enough number of angular intervals. But a larger number of angular intervals will result in a more complex decision process and the approximation is useful only if it is amenable to efficient implementation. The latter aspect is further addressed in Section 6.2. It should be noted that the optimal coefficients obtained here are based on the assumption that the phase of the input signal is uniformly distributed. This assumption applies very well to most signal modulations. In the special case of a skewed phase probability density, the true optimal coefficients can be better approached by using unequal angular intervals.

6.2 Practical Implementation and Performance

For the purpose of practical implementation, the approximation based on three angular intervals is chosen. The intervals are equally spaced to minimize the peak error. The threshold angles are $\theta_1 = \frac{\pi}{12}$, and $\theta_2 = \frac{\pi}{6}$. For each input sample $(I_\mathrm{r} + jQ_\mathrm{r})$, the corresponding angular interval is determined by comparing Q_r to $I_\mathrm{r} \times \tan(\theta_k)$ since $\tan(\cdot)$ is a monotonic function in the interval $[0, \frac{\pi}{4}]$. For efficient hardware implementation, we select $\tan(\theta_1) = \frac{1}{4}$ and $\tan(\theta_2) = \frac{9}{16}$. The coefficients obtained from (6.5) are quantized to six bits of resolution. For best results, the quantized coefficients a_k are used to generate new sub-optimal coefficients b_k, which are in turn quantized. This two-step process results in a slightly better performance than the direct quantization of the coefficients a_k and b_k. The coefficients and error characteristics of the floating-point and quantized amplitude approximations are summarized in Table 6.2. Even though the fixed-point approximation is more practical, its performance is very close to that of the floating-point approximation.

Table 6.2: Coefficients and error for an amplitude approximation with three intervals.

Parameters & errors	**Floating-point**	**Fixed-point**
$[\mathbf{a_1\ a_2\ a_3}]$	[0.994 0.927 0.796]	[1 60/64 51/64]
$[\mathbf{b_1\ b_2\ b_3}]$	[0.131 0.384 0.610]	[6/64 23/64 39/64]
$\tan(\boldsymbol{\theta_1})$	0.268	1/4
$\tan(\boldsymbol{\theta_2})$	0.577	9/16
$\varepsilon_{\mathbf{peak}}(\%)$	0.572	0.712
$\varepsilon_{\mathbf{mean}}(\%)$	0.001	0.082
$\varepsilon_{\mathbf{rms}}(\%)$	0.256	0.306

The performance of the fixed-point amplitude approximation was simulated with a predistorted transmitter using a WCDMA signal as input. A linearly-interpolated complex-gain predistorter with an LUT size of 64 entries was used. The transmitter features a highly nonlinear class-E amplifier. The input signal (I/Q) resolution was set to 13 bits and a 3 dB backoff was selected. Table 6.3 shows the resulting error vector magnitude (EVM) and the adjacent channel leakage ratios at 5 MHz offset (ACLR1) and 10 MHz offset (ACLR2). The EVM resulting from the use of the amplitude indexing is 8 dB lower than that of the power indexing, and only 2 dB higher than that of the ideal amplitude indexing. The ACLR1 measurements are very close with a maximum difference equal to 0.5 dB. The ACLR2 measurements show a 6 dB improvement when using the amplitude approximation instead of the power indexing.

Table 6.3: Performance of amplitude approximation: WCDMA EVM and ACLR.

Indexing		**EVM** dB	**ACLR1** dBc/Hz	**ACLR2** dBc/Hz
Power		−44.71	−53.60	−53.86
Amplitude	approx	−61.57	−67.86	−69.95
	ideal	−63.60	−68.99	−71.68

Figure 6.3 shows the WCDMA PSD resulting from the above experiment. The higher spectral floor resulting from the power indexing shows that it suffers from stronger residual nonlinear distortions.

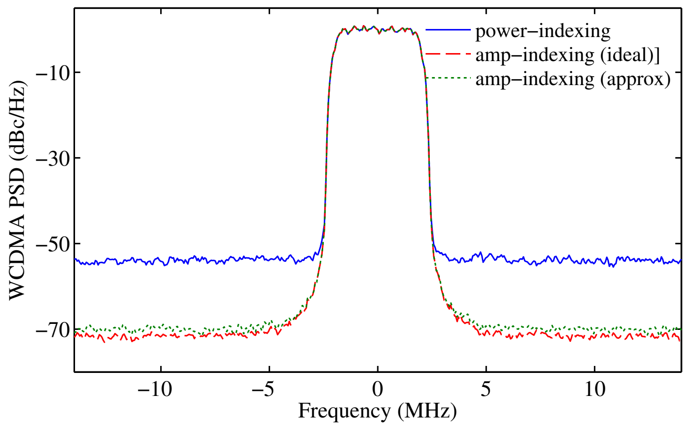


Figure 6.3: Performance of amplitude approximation: WCDMA PSD.

The fixed-point coefficients and angular thresholds are chosen to minimize the hardware implementation complexity while maintaining an approximation error close to the optimum value. The diagram of Figure 6.4 illustrates a possible implementation.

This design requires two conditional *2's complement* operations to implement the $abs(\cdot)$ function, three comparators, and four two-to-one multiplexers. The coefficients were chosen to minimize the complexity of the scaling operations. To achieve a fair comparison, the

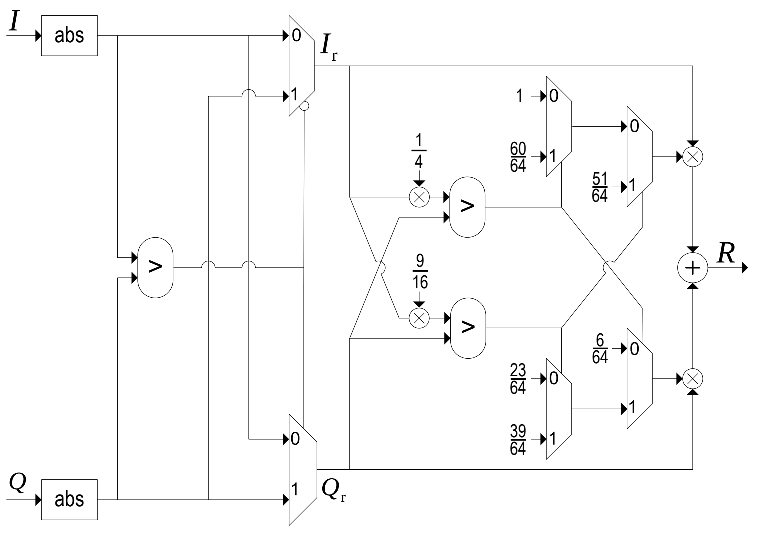


Figure 6.4: Implementation of the amplitude approximation with three angular intervals.

implementation complexity of the amplitude approximation must be compared to that of the instantaneous power computation (I^2+Q^2). Both options were implemented in VHDL and synthesized with the Synopsys Design Compiler. The resulting nand2-equivalent gate count is obtained for different resolutions of the input quadrature components (I/Q). The synthesis results are summarized in Figure 6.5.

It is clear from these results that the amplitude approximation design results in lower gate count for the considered range of input resolutions. The gap rapidly increases as the resolution is increased from 8 to 20 bits. For input resolutions lower than 8 bits, the power computation results in a slightly lower gate count. But at such low resolutions, the performance is limited by the I/Q quantization error. In this case, the resolution of the $(a_k,\, b_k)$ coefficients can be reduced to 5 or 4 bits to further reduce the gate count of the amplitude approximation block. For most wireless transmitters, an I/Q resolution of more than 10 bits is required to meet the standard specifications. Therefore, the proposed amplitude approximation design has a clear advantage both in terms of total design area and performance.

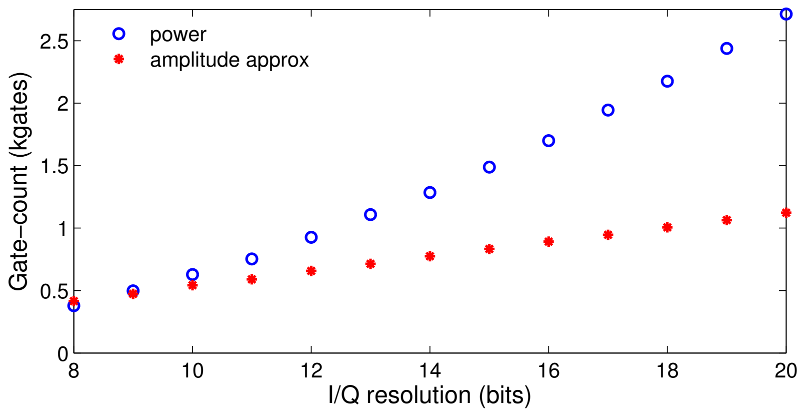


Figure 6.5: Nand2-equivalent gate count for power index computation and amplitude approximation with three angular intervals.

6.3 Conclusion

In this chapter, an amplitude approximation suitable for digital baseband predistorters is proposed. A closed-form solution is derived to determine the optimal parameters for any arbitrary angular interval. A quantized amplitude approximation with three angular intervals is implemented in VHDL and synthesized with the SYNOPSYS DESIGN COMPILER. It is shown to be very close to the ideal amplitude computation and outperforms the power-indexing in both design area and rejection of residual distortion.

CHAPTER VII

EFFICIENT LUT PREDISTORTER ADAPTATION

The nonlinear characteristics of power amplifiers display significant variations when the operating temperature fluctuates and as the device ages. To maintain the effectiveness of the predistorter and minimize the residual distortions, an adaptive predistorter should be used. For resource-constrained mobile devices, the computational complexity of the chosen adaptation algorithms and the number additional components must be minimized.

In this chapter, an efficient LMS-based [51] adaptation technique for LUT predistorters is presented as well as its optimization for low complexity hardware implementation. The identification of the inverse nonlinearity is based on the indirect learning architecture [20].

7.1 Adaptation of Complex-Gain LUT Predistorters

The indirect learning architecture is illustrated in Figure 7.1. A replica of the feedforward predistorter is trained in the feedback as the post-inverse of the amplifier nonlinearity. The updated LUT is periodically copied to the feedforward predistorter. This configuration has the advantage of decoupling the transmit path from the update branch. The transmitted signal is therefore isolated from any impulse noise in the feedback path at the cost of replicating the predistorter.

The LUT is an array of L complex-gain entries $F^{[n]}$ corresponding to input amplitude indexes $r_n = |y_n|$. If the LUT is not interpolated, the nth LUT entry is selected for all feedback signals y_k in the interval defined by

$$\frac{y_n + y_{n-1}}{2} \leq y_k < \frac{y_n + y_{n+1}}{2}.$$

For every signal sample y_k in this interval, an error signal e_k is generated:

$$e_k = z_k - F^{[n]} y_k.$$

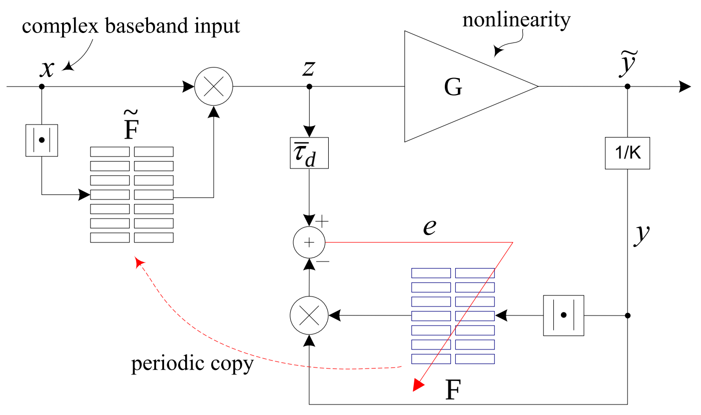


Figure 7.1: Adaptation of complex-gain LUT predistorters using the indirect learning architecture.

The nth entry $F^{[n]}$ can be updated using the LMS algorithm as follows:

$$F_{k+1}^{[n]} = F_k^{[n]} - \mu \frac{\partial e_k^* e_k}{\partial F^{[n]}}. \tag{7.1}$$

The gradient with respect to the complex gain $F^{[n]}$ is obtained by taking the partial derivatives with respect to real and imaginary parts:

$$\frac{\partial e_k^* e_k}{\partial F^{[n]}} = \frac{\partial e_k^* e_k}{\partial \Re\{F^{[n]}\}} + j \frac{\partial e_k^* e_k}{\partial \Im\{F^{[n]}\}}, \tag{7.2}$$

where $\Re\{\cdot\}$ and $\Im\{\cdot\}$ respectively designate the real and imaginary parts of the argument. Substituting (7.2) in (7.1) and carrying out the partial derivatives gives

$$F_{k+1}^{[n]} = F_k^{[n]} + 2\mu y_k^* e_k. \tag{7.3}$$

Considering a single interval at a time allows to simplify the problem by reducing it to finding an approximate inverse of the average amplifier complex gain within the considered interval. For each incoming feedback sample, only the corresponding entry, which is addressed by its magnitude, is updated. The update operation requires two complex multiplies (one to compute the error e and one to evaluate the gradient), two additions and the scaling by μ, which can be simplified if it is restricted to powers of two. The update system is stable provided that $0 < \mu < \frac{1}{\sigma_n^2}$, with σ^2 being equal to $E[|y_k|^2]$ for all y_k falling in the nth interval. If the LUT size is large, the samples y_k can be assumed to have a uniform distribution across the interval. In this case, the expectation can be approximated by the square of the average magnitude, which is the point located at the center of the interval: $\sigma_n^2 \approx |y_n|^2$.

If the regular LMS update equation (7.3) is used, the convergence speed will vary across the table entries. The upper entries will converge significantly faster than lower entries. To avoid this issue, the normalized LMS algorithm [27] can be used:

$$F_{k+1}^{[n]} = F_k^{[n]} + 2\frac{\mu}{|y_k|^2}\, y_k^*\, e_k. \tag{7.4}$$

The normalized LMS (NLMS) update of (7.4) results in faster and uniform convergence of the entries across the LUT. But its direct implementation has two limitations:

- For very low values of $|y_k|$ the system becomes very susceptible to the noise in the feedback path, potentially driving the update system into instability.
- The scaling by the magnitude is an expensive operation that is not directly amenable to efficient hardware implementation.

An approximation of the normalized LMS similar to the clipped LMS algorithm [15, 37, 50] is proposed. This approach termed low-complexity normalized LMS (LCNLMS) is suitable for efficient hardware implementation and maintains the fast convergence of the normalized LMS. First, the update equation of (7.4) can be reformulated as follows:

$$\begin{aligned} F_{k+1}^{[n]} &= F_k^{[n]} + 2\frac{\mu}{|y_k|}\frac{y_k^*}{|y_k|} e_k \\ &= F_k^{[n]} + 2\mu_k\, e^{j\phi_k}\, e_k, \end{aligned}$$

where $\phi_k = \angle y_k^*$ is the complex argument of y_k^* and $\mu_k = \dfrac{\mu}{|y_k|}$. It is clear from this incremental update that the normalized LMS is equivalent to using a variable update coefficient that is inversely proportional to the input amplitude $|y_k|$ and replacing the complex multiply with a rotation of the error by ϕ_k. The computational complexity of the rotation operation can be greatly simplified by quantizing the angle ϕ_k. To do so, let us define the sign function $\operatorname{sgn}(\cdot)$, corresponding to the sign bit in the *two's complement* representation as:

$$\operatorname{sgn}(x) = \begin{cases} +1, & \text{if } x \geq 0 \\ -1, & \text{if } x < 0 \end{cases}.$$

Let S_I and S_Q respectively be the signs of the real and imaginary parts of the feedback signal y_k:

$$S_I = \text{sgn}(\Re\{y_k\}) \qquad\qquad S_Q = \text{sgn}(\Im\{y_k\}).$$

The quantization of the angle ϕ_k can be achieved by using the following update equation:

$$\begin{aligned} F_{k+1}^{[n]} &= F_k^{[n]} + 2\mu_k(S_I - jS_Q)e_k \\ &= F_k^{[n]} + 2\mu_k(\pm 1 \pm j)e_k \\ &= F_k^{[n]} + 2\sqrt{2}\mu_k e^{j\frac{m\pi}{4}} e_k, \end{aligned} \tag{7.5}$$

with

$$m = S_Q(S_I - 2).$$

The phase ϕ_k is therefore quantized to four possible values, i.e. $\phi_k \in \{\pm\frac{\pi}{4}, \pm\frac{3\pi}{4}\}$, thus effectively eliminating one complex multiplier (four real multipliers). The phase quantization affects the convergence trajectory but has little effect on the convergence speed. This effect is illustrated in Figure 7.2, where the Normalized LMS is shown to take a direct path to the optimal solution, while the phase quantization causes the value of $F^{[n]}$ to oscillate around the shortest path.

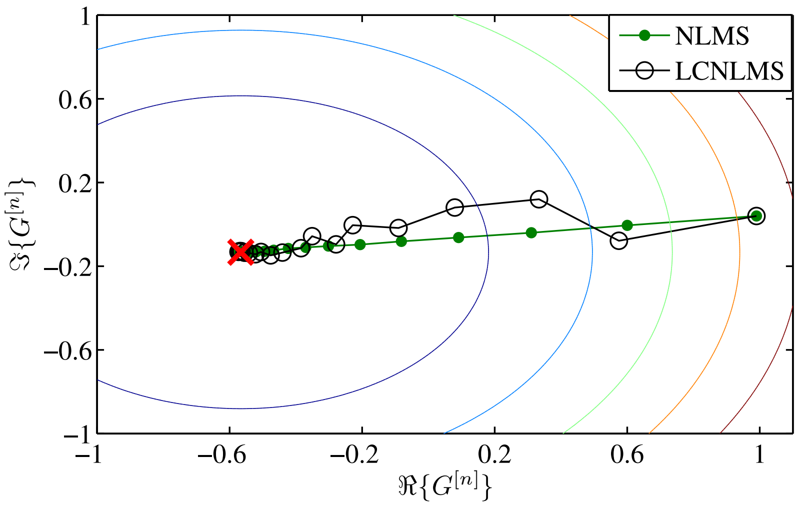


Figure 7.2: Convergence paths of NLMS and LCNLMS.

The amplitude-dependent coefficient μ_k could be implemented as a lookup table with one coefficient per table entry. To minimize the required memory space and further reduce the implementation costs, μ_k can be constrained to powers of two and generated from the magnitude $|y_k|$ as follows :

$$u_k = 2^{\eta_k} \qquad \text{with} \qquad \eta_k = \min\big\{ -\lceil \log_2(|y_k|) \rceil,\, \eta_0 \big\},$$

where $\lceil \cdot \rceil$ stands for the ceil$(\cdot)$ rounding function and η_0 is an arbitrary integer. In the above expression, it is assumed without loss of generality that the signal is normalized such that $|y_k| < 1$. The base-two exponent η_k can be very efficiently generated with the simple combinatorial circuit illustrated in Figure 7.3. Forcing the maximum exponent to η_0 sets a maximum value for μ_k to prevent any instability caused by the sensitivity to noise at low amplitudes.

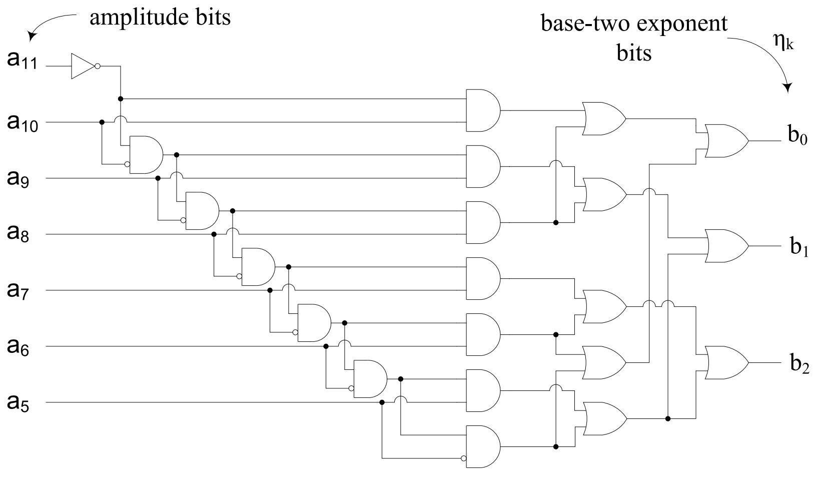


Figure 7.3: Circuit that generates the base-two exponent η_k.

In the circuit implementation of Figure 7.3, the amplitude is represented with 12 bits of resolution and the exponent η_k is represented with a three-bit binary word. This is equivalent to setting $\eta_0 = 7$. The scaling by μ_k can be implemented by a simple binary shifter.

The combinatorial implementation of the amplitude-dependent update coefficient lacks flexibility since the update speed cannot be changed. This issue can be tackled by introducing an additional coefficient μ_a that is programmable:

$$F_{k+1}^{[n]} = F_k^{[n]} + 2\mu_a \mu_k (S_I - jS_Q) e_k.$$

It should be noted that this low complexity update is even simpler to realize in hardware than the regular LMS, which requires two complex multipliers and has a much slower convergence speed. Figure 7.4 compares the convergence of the regular LMS, the normalized LMS (NLMS), and the proposed low complexity update method (LCNLMS).

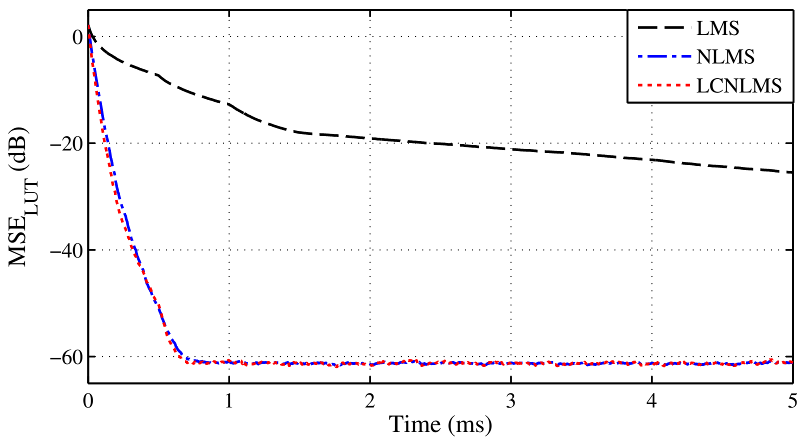


Figure 7.4: Convergence speed of LMS, NLMS, and LCNLMS.

This experiment is based on a linearly interpolated complex-gain LUT predistorter of size $L = 64$ entries that linearizes a class-E power amplifier. The LUT is updated at a sample rate of 30.76 MHz. Figure 7.4 shows the instantaneous mean squared error MSE_{LUT} between the updated LUT F and an optimal reference LUT H obtained by using least-square approximations in each interval.

$$\text{MSE}_{\text{LUT}} = \frac{1}{L}\sum_{n=1}^{L}\left|F^{[n]} - H^{[n]}\right|^2$$

These results show that the proposed LCNLMS achieves the same convergence speed as the normalized LMS and has an implementation complexity even lower than the regular LMS, which suffers from a very slow convergence of lower table entries. To achieve a fair comparison, the update coefficient of the low complexity normalized LMS was normalized so that the additional factor of $\sqrt{2}$ in (7.5) is canceled.

7.2 Updating a Linearly-Interpolated LUT

Linear interpolation greatly reduces the LUT approximation errors and enables significant reduction of the required LUT size. If linear interpolation is used, for each feedback sample magnitude $|y_k|$ falling between addresses n and $n+1$, the interpolated complex-gain is

$$F_k = F^{[n]} + \lambda_k \left(F^{[n+1]} - F^{[n]} \right),$$

where λ_k is the interpolation factor. For the purpose of practical implementation, the address n and the interpolation factor λ_k are readily obtained from the amplitude bits:

$$\text{amplitude bits } (|y_k|) \quad \Longrightarrow \quad \underbrace{\mathrm{a}_{11}\mathrm{a}_{10}\mathrm{a}_{09}\mathrm{a}_{08}\mathrm{a}_{07}\mathrm{a}_{06}}_{\text{address bits } (n)}\underbrace{\mathrm{a}_{05}\mathrm{a}_{04}\mathrm{a}_{03}\mathrm{a}_{02}\mathrm{a}_{01}\mathrm{a}_{00}}_{\text{interpolation factor}(\lambda)}.$$

It should be noted that for each input sample, two consecutive LUT entries must be fetched from memory and interpolated to compute the complex-gain. The hardware implementation and the sequencing of operations can be greatly simplified by using a dual-port memory. In general, dual-port memories are more expensive and larger in size than single-port memories of same capacity. But in the case of the LUT interpolation, the two entries to be fetched are always located at consecutive addresses. Consequently, a dual-port memory of size N can be emulated from two single-port memory blocks of size $\frac{N}{2}$ and simple additional logic. One of the blocks stores the entries located at even addresses and the other one store the entries at odd addresses. See Figure 7.5 for an illustration.

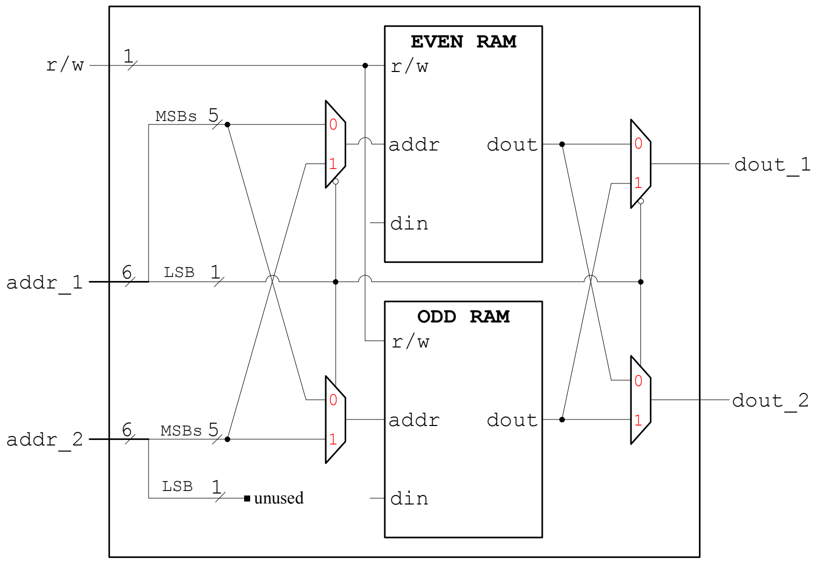


Figure 7.5: Pseudo dual-port memory to implement a linearly-interpolated LUT.

This process allows the implementation of a pseudo dual-port memory at the same cost as a single-port memory. The only limitation is that simultaneous read/write operations require one address to be odd and the other to be even. In the case of a linearly-interpolated LUT, this requirement is always satisfied because the addresses n and $n+1$ are consecutive. If linear interpolation is used in the feedback path (adapted LUT), the error signal e_k is given by

$$\begin{aligned} e_k &= z_k - F_k\, y_k \\ &= z_k - \left[(1-\lambda)F^{[n]} - \lambda F^{[n+1]}\right] y_k. \end{aligned} \tag{7.6}$$

Since two entries are used to generate the interpolated complex-gain, both entries should be updated with each new data sample. The application of the same LMS algorithm by alternatively computing the gradients with respect to $F^{[n]}$ and $F^{[n+1]}$ results in the following update equations:

$$\begin{aligned} F_{k+1}^{[n]} &= F_k^{[n]} + \mu(1-\lambda)\, y_k^*\, e_k \\ F_{k+1}^{[n+1]} &= F_k^{[n+1]} + \lambda\mu\, y_k^*\, e_k. \end{aligned}$$

It should be noted that both the nearest neighbor and linear interpolation adaptations converge to the same solution. The linearly interpolated case has lower LUT approximation errors and therefore results in a slightly better steady state performance. The nearest neighbor method can generally match that performance if the update coefficient is reduced, at the cost of slower convergence. For practical implementations, the nearest neighbor approach is more attractive in the feedback path since it requires only one memory read and write for each data sample. On the other hand, the linearly interpolated adaptation requires two memory reads and writes per data sample, putting more stringent timing requirements on the adaptation hardware. It is therefore judicious to restrict the use of linear interpolation to the feedforward predistorter, where the approximation errors are directly reflected in the transmitted signal.

7.3 Limitations of Direct Learning

In the direct learning architecture illustrated in Figure 7.6, the feedforward predistorter is directly updated. Since a secondary feedback predistorter is not needed, the direct learning architecture requires fewer additional components than the indirect learning architecture.

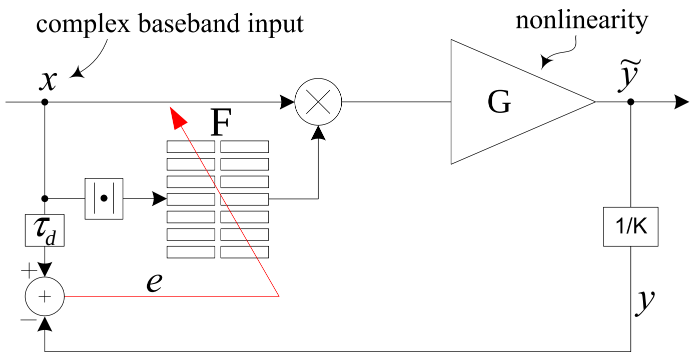


Figure 7.6: Direct learning architecture.

Using the same approach as above, the update equations can be computed by considering a single LUT bin at a time, so that the predistorter reduces to a simple average complex factor across that bin. The error signal is

$$\begin{aligned} e_k &= x_k - y_k \\ &= x_k - F^{[n]} g_k x_k, \end{aligned} \tag{7.7}$$

where g_k is the average of the amplifier's complex gain across the considered signal values, normalized by the real linear gain K. The complex-gain g_k includes any phase-shift in the feedback path. Applying the LMS algorithm we get

$$F^{[n]}_{k+1} = F^{[n]}_k + 2\mu\, g^*_k\, x^*_k\, e_k.$$

The complex factor g_k is unknown a priory and cannot be used for the update. Since its magnitude is positive, it can be ignored as long as an appropriate update coefficient μ is selected to guaranty stability. Its phase can also be ignored provided that $-\frac{\pi}{2} < \angle g_k < \frac{\pi}{2}$. This will result in the following simplified update equation:

$$F^{[n]}_{k+1} = F^{[n]}_k + 2\mu\, x^*_k\, e_k.$$

A necessary condition for the convergence of the above update equation is $-\frac{\pi}{2} < \angle g_k < \frac{\pi}{2}$. The total phase-shift across the feedback loop depends on many varying factors such as the matching conditions, the amount of delay in the RF path or the feedback oscillator synchronization etc.. Therefore, an adaptive phase-shifter is required in the feedback to guaranty that the convergence condition is always met.

In addition to this limitation, the direct learning architecture exposes the transmit path to potential impulse noise in the feedback. Any temporary disruption in the loop is reflected in the transmitted signal, potentially causing a violation of the spectral requirements. Additionally, wideband modulated signals require a feedforward predistorter operated at a very high sampling rate. In this case, reading the LUT at very high speed for the purpose of predistortion and updating it simultaneously might result in difficult timing challenges in a practical implementation. For all the reasons enumerated above, the indirect learning method appears to be more suitable to the design of an adaptive predistorter.

7.4 Conclusion

An adaptive predistortion algorithm with efficient hardware implementation is proposed in this chapter. An optimized LMS adaptation based on the indirect learning architecture is presented. The proposed adaptation has a convergence speed that is comparable to the normalized LMS and lends itself to very efficient hardware implementation. Finally the indirect learning and direct learning architectures are compared and the shortcomings of the latter are exposed.

CHAPTER VIII

PREDISTORTION AND QUADRATURE IMBALANCES

It has been shown in [9] that quadrature (I/Q) gain and phase imbalances severely limit the effectiveness of adaptive complex-gain predistorters in mitigating the effects of transmitter nonlinearity. In this chapter, it is shown that the transmitter nonlinearity can also affect the identification and compensation of the quadrature imbalances. This relatively strong interaction between predistortion and I/Q mismatch (IQM) compensation makes it difficult to individually address these two issues, one at a time. The 2D mapping predistorter proposed by Nagata [39] is a comprehensive solution that simultaneously linearizes the transmitter and compensates I/Q imbalances and DC offsets. But the high memory requirements and slow training convergence makes it an unattractive solution that is unsuitable to low-cost resource-constrained wireless handsets.

In this chapter, two alternative solutions are studied. First, a method alternating I/Q mismatch correction (IQMC) and predistorter identification is presented. The second solution is a novel predistorter structure that has lower computational complexity than the combination of a complex-gain predistorter plus a separate IQMC. This new solution requires twice the memory space of the complex-gain predistorter, but is still far more memory-efficient than the mapping predistorter. Furthermore, its convergence speed is similar to that of the complex-gain predistorter.

8.1 Interactions Between IQ Imbalance and Nonlinearity

The effects of I/Q imbalances (IQM) on the effectiveness of the complex-gain predistorter were studied in great details in [9, 12]. In these previous studies, it is shown that even carefully-designed modulators with relatively low levels of IQM could erase the benefits of the predistorter and make its adaptation very challenging.

Let us consider the simplified transmitter modeled in Figure 8.1. The amplifier's characteristic is modeled as the product of a real, constant gain K and an amplitude-dependent complex gain component $g(r)$, which emulates the amplifier's nonlinear behavior.

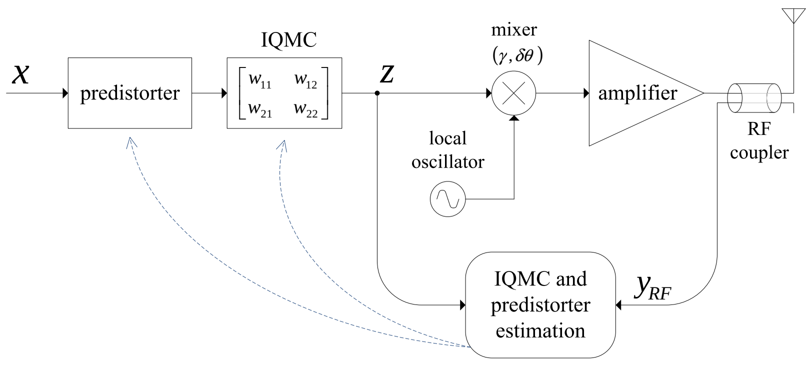


Figure 8.1: Transmitter configuration with predistorter and IQM corrections.

The quadrature gain and phase imbalances can be modeled in many different but equivalent ways [9, 10]. Without loss of generality, we model the imbalance as a set of complex coefficients $\left(\alpha e^{j\theta_1},\ \beta e^{j\theta_2}\right)$ that respectively scale the in-phase (I) and quadrature (Q) components of the complex baseband input signal. Using vector notation, the IQMC can be modeled as a 2×2 matrix that transforms the 2D input signal vector $\mathbf{x} = [x_i \quad x_q]^{\mathrm{T}}$:

$$\mathbf{z} = M\mathbf{x},$$

with

$$M = \begin{bmatrix} \alpha\cos\theta_1 & -\beta\sin\theta_2 \\ \alpha\sin\theta_1 & \beta\cos\theta_2 \end{bmatrix}.$$

Adopting the terminology in [12], we define the dimensionless gain ratio $\gamma = \alpha/\beta$ and the gain imbalance $\varepsilon = \gamma - 1$, expressed in percentage. The phase mismatch is represented by $\delta\theta = \theta_1 - \theta_2$. To distinguish between a power change and pure I/Q imbalance, the condition $\alpha^2 + \beta^2 = 2$ must be satisfied [12]. The magnitudes of x and z are related as follows:

$$\begin{aligned} r_{\mathbf{z}}^2 &= \alpha^2 x_i^2 + \beta^2 x_q^2 + 2\alpha\beta x_i x_q \sin(\theta_1 - \theta_2) \\ &= \alpha^2 \left[r_{\mathbf{x}}^2 + (1/\gamma^2 - 1)x_q + 2\alpha\beta x_i x_q \sin\delta\theta \right], \end{aligned} \tag{8.1}$$

with $r_{\mathbf{x}}^2 = \mathbf{x}^{\mathrm{T}}\mathbf{x}$ and $r_{\mathbf{z}}^2 = \mathbf{z}^{\mathrm{T}}\mathbf{z}$. The amplifier will scale the signal $\mathbf{z}$ with the complex gain $Kg(r_{\mathbf{z}})$. Combining this result with (8.1) shows that the nonlinearity is now a function of both the magnitude and phase of the input signal $\mathbf{x}$. Therefore, an amplitude-dependent complex-gain predistorter cannot fully mitigate the amplifier's nonlinearity, unless and an IQM correction is inserted in the transmitter. As noted in [12], even the envelopes of constant-magnitude signals (CW tone, GMSK signal, etc..), known for their immunity to amplifier nonlinearity, will be modulated by I/Q imbalances causing the appearance of unexpected spectral regrowth. If a CW input of frequency f_o is considered, the RF output of an ideal mixer would be:

$$v_z = \cos[2\pi(f_c + f_o)t + \phi_o],$$

where f_c is the carrier frequency and ϕ_o an arbitrary phase shift. The input of the amplifier is then a single tone at frequency $f_c + f_o$, which is immune to nonlinear distortions. In the presence of I/Q imbalance, the modulator output would be:

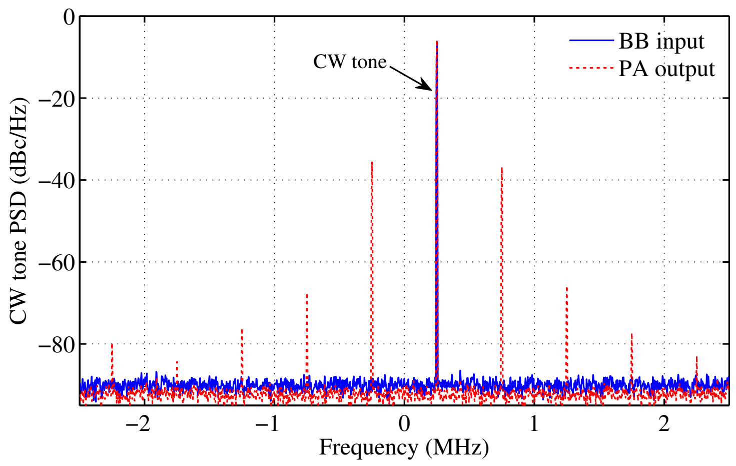
$$\tilde{v}_z = A_1 \cos[2\pi(f_c + f_o)t + \phi_1] + A_2 \cos[2\pi(f_c - f_o)t + \phi_2],$$

with

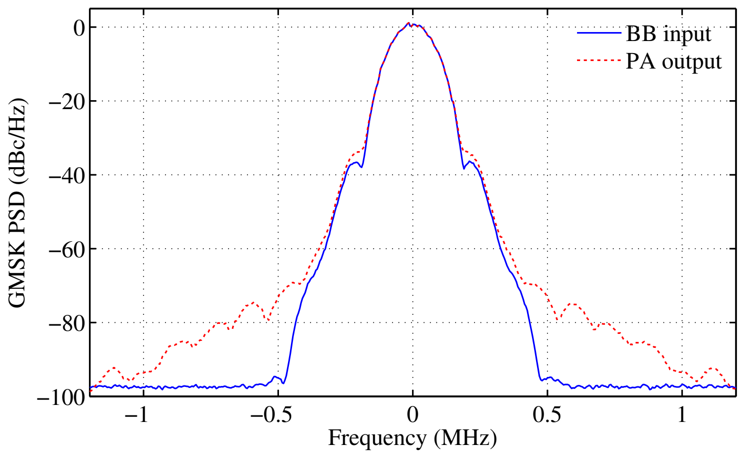
$$\begin{aligned} A_1 &= \sqrt{\frac{1 + 2\gamma\cos\delta\theta + \gamma^2}{2(1+\gamma^2)}} & A_2 &= \sqrt{\frac{1 - 2\gamma\cos\delta\theta + \gamma^2}{2(1+\gamma^2)}} \\ \phi_1 &= \arctan\left(\frac{\alpha\sin\theta_1 + \beta\sin\theta_2}{\alpha\cos\theta_1 + \beta\cos\theta_2}\right) & \phi_2 &= \arctan\left(\frac{\alpha\sin\theta_1 - \beta\sin\theta_2}{\alpha\cos\theta_1 - \beta\cos\theta_2}\right). \end{aligned}$$

Because of the I/Q imbalance, the output of the modulator now contains a residual image tone at frequency $f_c - f_o$. When transmitted through the nonlinear amplifier, spurious intermodulation products will be generated. This is illustrated by the simulations results in Figure 8.2, showing the PSD of the amplifier's output for an input CW tone at $f_0 = 250\,\text{kHz}$ offset. The gain and phase mismatches are respectively $\varepsilon = 5\%$ and $\delta\theta = 6°$. Figure 8.2(a) shows that the modulation of a CW tone by the quadrature imbalances interacts with the amplifier nonlinearity causing undesired inter-modulation products at frequencies $f_c + (2k + 1)f_0$, with $k \in \{\pm1, \pm2, \pm3, \cdots\}$.

The transmission of a constant-envelope GMSK input signal under the same imbalance conditions is illustrated in Figure 8.2(b). It is apparent from these results that a relatively low level of I/Q imbalance makes the constant-envelope GMSK signal vulnerable to amplifier nonlinearity causing non-negligible levels of spectral regrowth.



(a)



(b)

Figure 8.2: PSD of PA output, when $\varepsilon = 5\%$ and $\delta\theta = 6°$. (a) CW tone input at 250 kHz offset. (b) GMSK input.

8.1.1 Effect of Nonlinearity on IQM Correction

It was shown above that the I/Q imbalance significantly reduce the effectiveness of a previously trained predistorter. Similarly, the nonlinearity of the amplifier will negatively impact the identification of the IQM correction (IQMC).

For the purpose of the IQMC identification, we assume that the predistorter is set to the identity mapping. We also assume an input signal $\mathbf{x}_k = [x_{ik}\ x_{qk}]^{\mathrm{T}}$ with a corresponding baseband output signal $\mathbf{y}_k = [y_{ik}\ y_{qk}]^{\mathrm{T}}$. It is assumed that $\mathbf{y}_k$ is ideally down-converted, delay-matched, and normalized by the linear gain of the amplifier K. If the amplifier is memoryless, the input/output relationship can be summarized by

$$\mathbf{y}_k = h_k P_k M \mathbf{x}_k + \mathbf{n}_k,$$

where h_k is the amplifier's gain variation obtained from (8.1):

$$h_k = \left| g[\alpha^2 x_{ik}^2 + \beta^2 x_{qk}^2 + 2\alpha\beta x_{ik} x_{qk} \sin(\delta\theta)] \right|.$$

P_k is a rotation matrix combining the amplitude-dependent phase-shift of the amplifier and any arbitrary phase-shift in the feedback path. n_k is assumed to be a zero-mean additive noise signal. The IQM correction matrix can be estimated by finding the least-squares solution of the post-inverse equation:

$$WY = X,$$

with

$$Y = [\mathbf{y}_1 \; \mathbf{y}_2 \; \cdots \; \mathbf{y}_N] \quad \text{and} \quad X = [\mathbf{x}_1 \; \mathbf{x}_2 \; \cdots \; \mathbf{x}_N].$$

The least-squares solution to the above equation is given by

$$W = XY^{\mathrm{T}}(YY^{\mathrm{T}})^{-1}.$$

With a large number of samples N, the additive noise n_k will have little to no effect on the estimation of the IQM correction matrix W. But the multiplicative correlated distortion term h_k and the signal-dependent rotation matrix P_k will negatively affect the accuracy of the estimated IQM correction matrix W. If the correction is accurate, then the residual matrix $R = MW$ will, in the general case, correspond to a constant rotation matrix. If all phase-shifts in feedback loop have been compensated before the estimation of W, then R would reduce to an identity matrix. If the IQM estimation is not accurate, R will correspond to a residual I/Q imbalance matrix.

This IQM correction estimation has been simulated with a class-E amplifier for various gain and phase mismatch settings. To reduce the effect of nonlinearity, a random input signal with constant amplitude and uniformly distributed phase is used. Table 8.1 shows the residual gain and phase mismatch $\tilde{\gamma}$ and $\delta\tilde{\theta}$ after pre-compensation with a correction matrix W estimated from $N = 1000$ samples.

Table 8.1: Effect of nonlinearity on IQM correction estimation.

Initial I/Q imbalance		**Residual I/Q imbalance**	
$\gamma\,(\%)$	$\delta\theta\,(^o)$	$\tilde{\gamma}\,(\%)$	$\delta\tilde{\theta}\,(^o)$
3.0	3.0	1.6	1.1
5.0	6.0	2.7	2.2
8.0	9.0	4.3	3.7

These results show that even with a constant envelope input, the amplifier nonlinearity significantly degrades the estimation accuracy of the IQ imbalance correction matrix. An accurate IQMC estimation requires the compensation of the nonlinear distortion in the output data samples $\mathbf{y}_k$. In other words, the samples $\mathbf{y}_k$ should be post-distorted before estimating the matrix W. On the other hand the accuracy of the post-distorter estimated from the data samples $\{\mathbf{x}_k, \mathbf{y}_k\}$ is deteriorated by the quadrature gain and phase imbalances.

It appears from the previous results that the nonlinearity and I/Q imbalances cannot be easily corrected individually, unless the amplifier can be bypassed or operated in a relatively linear region. But high-efficiency amplifiers exhibit severe nonlinear distortion across their entire amplitude range and a amplifier-bypass capability is not always a feasible solution.

8.1.2 The Mapping Predistorter

The two-dimensional mapping predistorter was proposed by Nagata in [39]. It is a two-dimensional complex LUT that is indexed by the complex baseband input's in-phase (I) and quadrature (Q) components. The two-dimensional nature of the mapping predistorter enables the simultaneous correction of the amplifier nonlinearity and I/Q imbalances. Unlike the complex-gain predistorter, the mapping predistorter is an additive correction. As a result, it is also capable of correcting the undesired DC offsets. The mapping predistorter is illustrated in Figure 8.3. Nagata also proposed a low-complexity training algorithm to build the 2D LUT and track the characteristic variations. The main downside of the mapping predistorter is its very high memory requirements. To achieve the same approximation accuracy as a 1D complex-gain LUT with N entries, a mapping predistorter of more than $4N^2$ entries must be used. This also results in a very slow convergence of training algorithms [8].

Additionally, the adaptation of the mapping predistorter is very sensitive to phase-shifts in the feed-back path and therefore, requires a fast adaptive phase correction function.

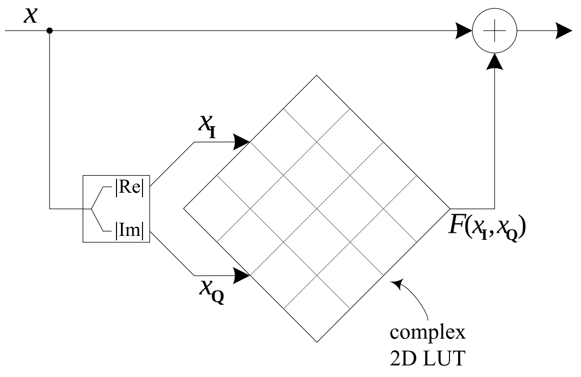


Figure 8.3: Mapping predistorter [39].

The performance of the mapping predistorter proposed by Nagata can be improved with a multiplicative correction, effectively making it a 2D complex-gain predistorter. Even though the DC offset correction capability will be lost, a significant reduction in the required memory size and can be achieved.

8.1.3 Iterative IQMC Estimation

The above results showed that separate identification of the IQM correction and the inverse nonlinearity is difficult from the set of data samples $\{\mathbf{x}_k,\ \mathbf{y}_k\}$. A possible solution is to iteratively estimate and apply IQM corrections of increasing accuracy. For each iteration step the data samples $\{\mathbf{x}_k,\ \mathbf{y}_k\}$ are measured and used to estimate an incremental IQM correction, which will be combined with the previous feedforward correction. The process is then repeated until a satisfactory accuracy of the feedforward IQMC is achieved. The iterative IQMC estimation is summarized in Figure 8.4. Table 8.2 shows the residual gain and phase imbalances for the first five iteration steps, when the initial gain and phase imbalances are respectively $\gamma = 8\%$ and $\delta\theta = 9°$.

The results in Table 8.2 show that the residual gain and phase imbalances are respectively reduced to 0.46% and 0.05° after just three iterations. The accuracy is further improved with five iterations. Icreasing the number of iterations above five has no noticeable effect. The iterative IQM correction method is experimentally shown to converge. It also has lower memory requirements than the mapping predistorter. But the high computational complexity of the multiple iterative least-squares estimations is a major limitation.

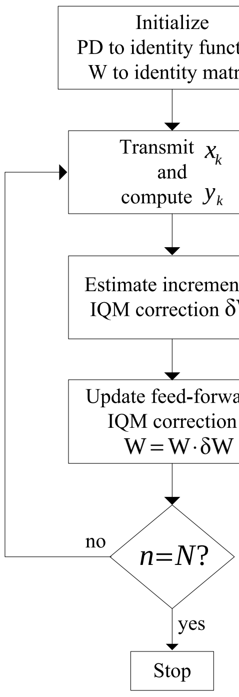

Initialize
PD to identity function
W to identity matrix
Transmit x_k and compute y_k
Estimate incremental IQM correction δW
Update feed-forward IQM correction W = W·δW
$n=N$?
no
yes
Stop

Figure 8.4: Iterative IQM correction.

Table 8.2: Iterative IQMC Estimation.

Iteration number	**Residual I/Q imbalance**	
k	$\tilde{\gamma}$ (%)	$\delta\tilde{\theta}$ (°)
0	8.00	9.00
1	5.72	2.51
2	1.95	0.49
3	0.46	0.05
4	0.08	0.02
5	0.01	0.01

8.2 2×2 Transform Predistorter

In the previous sections the mapping predistorter and the iterative IQMC estimations were shown to successfully correct I/Q mismatches in the presence of amplifier nonlinearity. But the high memory requirements of the mapping predistorter and the high computational complexity of the iterative IQMC estimation make these methods unpractical for resource-limited mobile devices. An alternative method is derived in this section.

The complex-gain predistorter $f(r) = f_I(r) + jf_Q(r)$ can be reformulated as a real 2×2 transform F that is applied to the input signal vector $[x_i \; x_q]^{\mathrm{T}}$:

$$F = \begin{bmatrix} f_I(r) & -f_Q(r) \\ f_Q(r) & f_I(r) \end{bmatrix}.$$

The equation above shows that the complex multiplication can be written as a special case 2×2 transform. The combination of the predistorter F and IQM correction W is computed as follows:

$$WF = \begin{bmatrix} w_{11}f_I(r) + w_{12}f_Q(r) & w_{12}f_I(r) - w_{11}f_Q(r) \\ w_{21}f_I(r) + w_{22}f_Q(r) & w_{21}f_I(r) - w_{22}f_Q(r) \end{bmatrix}.$$

We see from this result that by generalizing the complex-gain predistorter to an arbitrary amplitude-dependent 2×2 transform, the predistorter will be able to correct I/Q imbalances. The generalized predistorter is

$$F = \begin{bmatrix} f_{11}(r) & f_{12}(r) \\ f_{21}(r) & f_{22}(r) \end{bmatrix},$$

where the functions $f_{kj}(r)$ can be represented with polynomial functions or with lookup tables. It must be noted that even though the generalized 2×2 transform predistorter has twice the memory requirements of the complex-gain predistorter, it remains far more memory-efficient than the mapping predistorter, while providing better IQM correction accuracy. The mapping predistorter has the advantage of correcting DC offsets, but this capability cannot justify the difference in memory requirements since the DC offset correction can be separately implemented at a much lower cost.

It is also worth noting that the 2×2 transform requires four real multiplies and two additions. Therefore, it has the same computational complexity as a complex multiplier. Additionally, by combining predistortion and IQM correction into a single operation, the 2×2 transform predistorter has lower overall complexity than the combination of a complex-gain predistorter and a separate IQM correction.

Figure 8.5 illustrates the training of a 2×2 transform predistorter implemented as a set of LUTs indexed by the most significant bits of the signal's amplitude. The training of Figure 8.5 uses the indirect learning architecture [20].

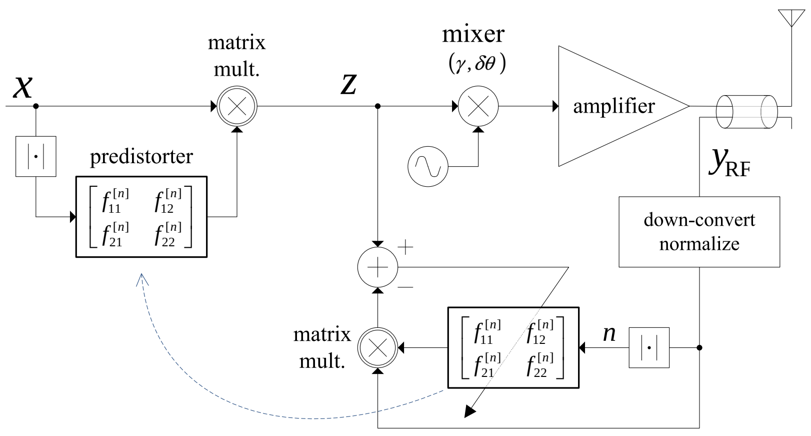


Figure 8.5: Training of 2×2 transform predistorter.

To minimize the implementation complexity, the least mean-squares (LMS) algorithm can be used to minimize the cost function

$$\begin{aligned} J &= E\left[|\mathbf{e}|^2\right] \\ &= E\left[\mathbf{e}^{\mathrm{T}}\mathbf{e}\right], \end{aligned} \tag{8.2}$$

with the error signal signal $\mathbf{e}$ given by

$$\mathbf{e}_k = \mathbf{z}_k - F^{[n]}\mathbf{y}_k,$$

where $F^{[n]}$ is the nth entry indexed by $|\mathbf{y}_k|$. For every new set of samples $\{\mathbf{z}_k, \mathbf{y}_k\}$, the LUT is updated as follows:

$$\begin{aligned} F_{k+1}^{[n]} &= F_k^{[n]} + \mu \frac{\partial\,|\mathbf{e}_k|^2}{\partial F^{[n]}} \\ &= F_k^{[n]} + 2\mu \mathbf{e}_k \mathbf{y}_k^{\mathrm{T}}. \end{aligned} \tag{8.3}$$

Figure 8.6 shows the EDGE PSD for a complex-gain predistorter(CGP), a mapping predistorter (MAP), a 2×2 transform predistorter (PD2×2), and the combination of a complex-gain predistorter and an IQM correction (CGP+IQMC) obtained after four iterations.

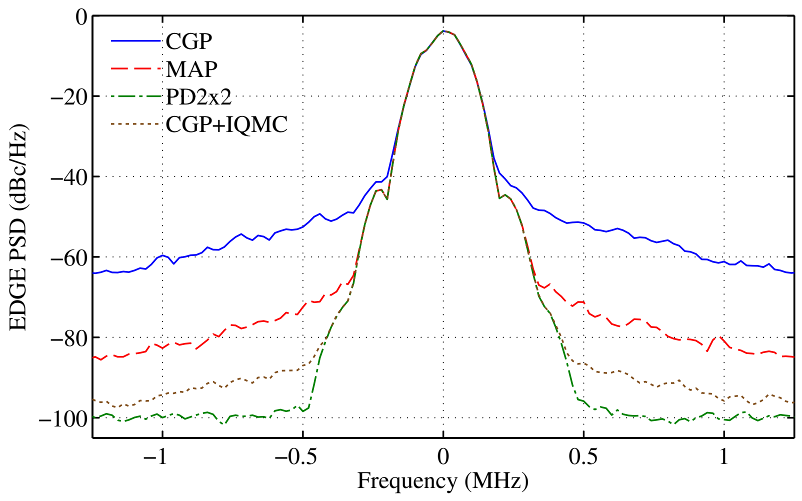


Figure 8.6: EDGE PSD for a complex-gain (CGP), a mapping (MAP), 2×2 transform predistorters (PD2×2) and a combination complex-gain predistorter and the iteratively trained IQM correction (CGP+IQMC).

The mapping predistorter uses a 32×32 2D LUT, with additive correction and the other configurations use an LUT size of 32 entries. The gain and phase imbalances are respectively set to $\gamma = 5\%$ and $\delta\theta = 6°$. The 2×2 transform predistorter achieves the best performance followed by the combination of a complex-gain predistorter and a IQM correction. Despite a very large LUT size, the mapping predistorter is less effective than the CGP+IQMC and PD2x2 configurations.

It must be noted that the CGP+IQMC has a good performance as long as the calibrated IQM correction remains accurate. In the situation where the I/Q imbalances vary during the operation of the device (e.g. temperature related variations), the IQM correction must be updated. The iterative IQM correction presented requires a special constant-envelope training signal and therefore cannot necessarily be used during the device operation. In this case the PD2×2 represents an attractive solution that can be trained with the transmitted modulated signal to adaptively linearize the transmitter and correct the temperature-dependent quadrature imbalances.

8.3 Conclusion

In this chapter, the interactions between amplifier nonlinearity and quadrature imbalances (IQM) were studied. A novel predistorter configuration that simultaneously linearizes the transmitter and compensates the quadrature gain and phase imbalances is proposed. The complex-gain predistorter is highly vulnerable to I/Q mismatches. Even relatively low levels of quadrature gain and phase imbalances can significantly reduce the effectiveness of the adaptive complex-gain predistorter.

It is shown that the proposed method is more robust than the complex-gain predistorter and can compensate for high levels of quadrature gain and phase imbalances. Even though it requires twice as much memory as the complex-gain predistorter, the proposed method combines linearization and I/Q imbalance correction in a single operation, resulting in lower computational complexity than a complex-gain predistorter followed by separate IQM correction. The proposed configuration with a simple adaptation algorithm, which tracks both types of impairments, is also a more attractive arrangement than two separate adaptation loops for the predistorter and IQM corrections. It should also be noted that the proposed method requires only a fraction of the memory of the 2D mapping predistorter by Nagata [39], which also provides IQM and DC offset correction.

CHAPTER IX

EFFECT OF VSWR ON PREDISTORTER ADAPTATION

The predistorter adaptation relies on the feedback signal to identify the inverse nonlinearity and track characteristic changes caused by temperature variations. Impedance mismatches in the RF transmit chain cause distortions to both the transmitted wave and the feedback signal sensed at the coupler. Furthermore, the matching conditions (mainly at the antenna) are subject to frequent variations depending on the handset position and its proximity to certain real life surfaces and objects. It is therefore necessary to measure the impact of such distortions on the predistortion adaptation and determine the VSWR conditions under which adaptative predistortion compensation is feasible.

9.1 Effect of Varying Antenna Impedance

The impedance matching between two elements is generally characterized by the reflection coefficient Γ. Figure 9.1 illustrates the reflections in the connection between the amplifier's output stage the and the antenna.

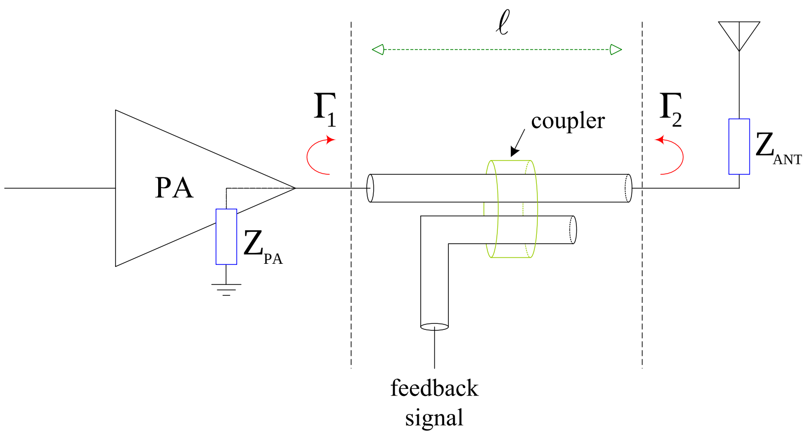


Figure 9.1: Impedance mismatch and reflections at the antenna connection.

The mismatch between the impedance of the amplifier's output stage Z_{PA} and the characteristic impedance of the transmission line Z_0 is measured by the reflection coefficient

Γ_1 given by

$$\Gamma_1 = \frac{Z_{\text{PA}} - Z_0}{Z_{\text{PA}} + Z_0}.$$

Similarly, the mismatch at the antenna connection is measured by the reflection coefficient Γ_2 given by

$$\Gamma_2 = \frac{Z_{\text{ANT}} - Z_0}{Z_{\text{ANT}} + Z_0}.$$

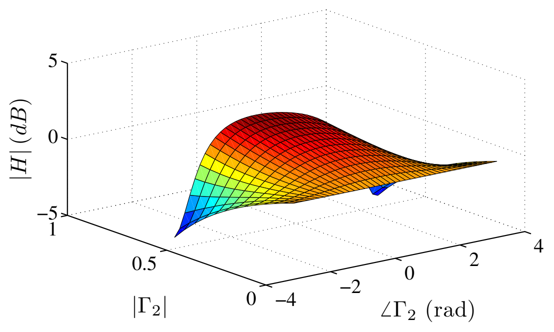
The impedance mismatches cause the appearance of a standing wave across the transmission line. The voltage standing wave ratio (VSWR) is defined as the ratio between the maximum and minimum amplitude of this standing wave. It is related to the reflection coefficient by

$$\text{VSWR} = \frac{1 - |\Gamma|}{1 + |\Gamma|}.$$

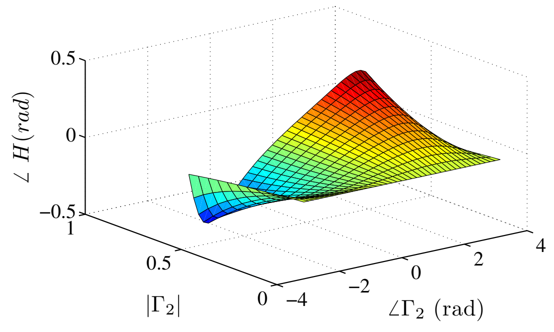
Part of the RF signal transmitted by the amplifier is reflected at the antenna connection and travels back to the amplifier output where it is again reflected. This process is repeated indefinitely. The signal that is measured at the coupler is the sum of all those incident and reflected signals. The baseband equivalent signal measured at the coupler is calculated as follows:

$$\begin{aligned} v_c &= v_{\text{PA}} \left(1 + \Gamma_2 + \Gamma_1\Gamma_2 + \Gamma_1\Gamma_2^2 + \Gamma_1^2\Gamma_2^2 + \Gamma_1^2\Gamma_2^3 + \cdots\right) \\ &= v_{\text{PA}} \left(1 + \Gamma_2\right) \left(1 + \Gamma_1\Gamma_2 + \Gamma_1^2\Gamma_2^2 + \Gamma_1^3\Gamma_3^2 + \cdots\right) \\ &= v_{\text{PA}} \underbrace{\frac{1 + \Gamma_2}{1 - \Gamma_1\Gamma_2}}_{H} \end{aligned} \tag{9.1}$$

For integrated circuits, the length of the transmission line is insignificant when compared to the signal's wavelength. Therefore the phase-shift terms $e^{jk\beta\ell}$ are neglected. The effect of impedance mismatches is a complex factor H multiplying the baseband feedback signal as shown in (9.1). This complex factor does not affect the predistorter adaptation as long as it is constant. As previously stated, the antenna impedance changes during the operation of the device causing the variation of H. Even a change of the phase of Γ_2 can result in several dBs of gain changes. Figure 9.2(a) and Figure 9.2(b) respectively show the variations of the gain and phase of H, when the reflection coefficient at the antenna Γ_2 changes.



(a)



(b)

Figure 9.2: Variations of the complex factor H when the antenna matching conditions change. (a) Magnitude of $|H|$. (b) Angle of $|H|$.

If the changes of complex gain H are not compensated, the predistorter will try to compensate for it, causing unnecessary repeated re-convergences, with a performance degradation during those transient convergence periods.

9.2 Adaptive Loop Gain Normalization

The complex-gain predistorter update mechanism presented in Chapter 4 is very sensitive to the gain and phase variations in the feedback path. Any change in the gain of the loop will be reflected in the updated predistorter. Therefore, it is necessary to have an accurate normalization of the complex loop gain during adaptation. An initial normalization factor G_{loop} is computed from the different gain settings in the feedback loop (amplifier, LNA etc..). Since the gains of the different components in the feedback loop vary with temperature, the calculated normalization factor is at best a rough estimate and is not expected to be very accurate (intractable temperature variations etc..).

More importantly, the antenna impedance variations discussed in Section 9.1 can cause significant and relatively fast gain and phase variations. In the absence of an equally fast and accurate adaptive correction, the predistorter update algorithm will attempt to compensate for these variations, potentially creating LUT curve discontinuities since the updates are local (only the active entries are updated). In addition to creating discontinuities, the accuracy of open-loop power control could be severely degraded. To avoid these effects, an adaptive gain and phase normalization block is proposed to improve the normalization accuracy and closely track the potential variations of the feedback loop gain.

The process of adaptive normalization is illustrated in Figure 9.3. A complex factor $\alpha = \alpha_{\mathrm{I}} + j\alpha_{\mathrm{Q}}$ is adapted such that $|E(x_{\mathrm{p}} - y_{\mathrm{o}})|^2$ is minimized. x_{p} is the output of the feedforward predistorter, and y_{o} is the baseband feedback signal. When the initial loop correction factor G_{loop} is inaccurate, α will converge to an incremental complex correction to compensate for the inaccuracies.

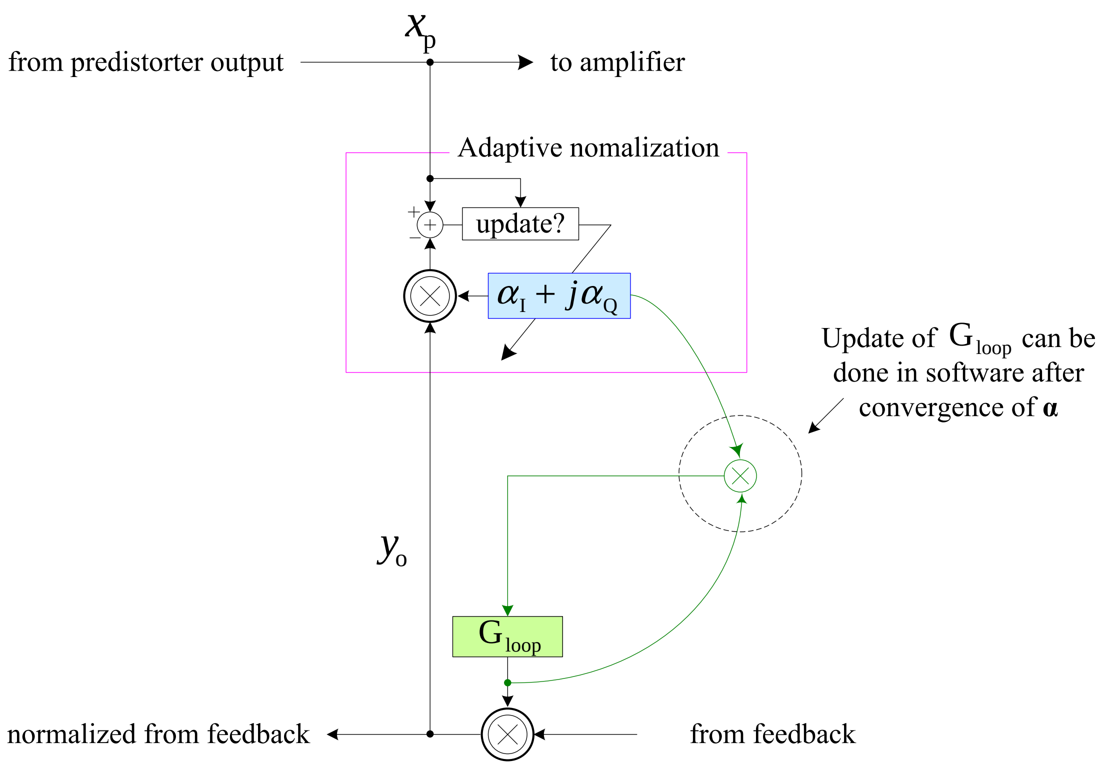


Figure 9.3: Adaptive loop gain normalization.

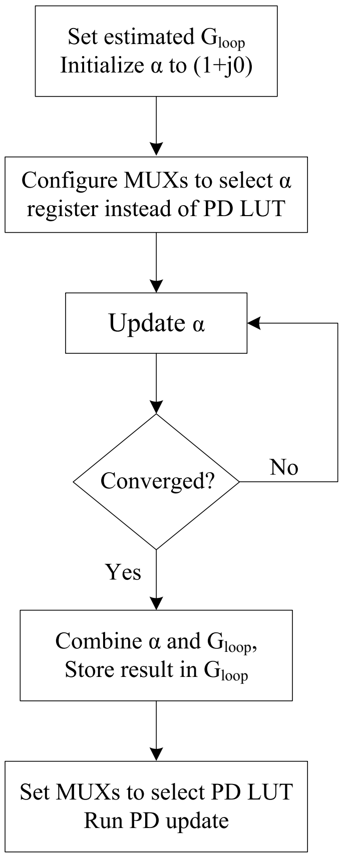
The proposed update equation for α is similar to the predistorter update. The LUT is simply replaced by a register containing α. The feedback error is first computed as follows:

$$e_k = x_{\mathrm{p}} - \alpha_k \, y_{\mathrm{o}}$$

Then the LMS update of α is given by

$$\alpha_k = \alpha_{k-1} + \mu \, y_{\mathrm{o}}^*$$

It is important to note that the predistorter update calculation hardware can be reused for the adaptive gain normalization, since the predistorter adaptation should be activated only after correct loop normalization is achieved. It should be noted that the convergence of α is several orders of magnitude faster than the predistorter LUT convergence, since all signal samples are used to update a register, as opposed to a single LUT entry being updated per sample. Figure 9.4 shows a flow diagram of the adaptive gain normalization and illustrates the hardware sharing between predistorter adaptation and adaptive loop gain normalization.



Set estimated G_{loop}
Initialize α to (1+j0)
Configure MUXs to select α register instead of PD LUT
Update α
Converged?
No
Yes
Combine α and G_{loop}, Store result in G_{loop}
Set MUXs to select PD LUT
Run PD update

Figure 9.4: Adaptive loop gain normalization and hardware sharing flow chart.

CHAPTER X

CONCLUSIONS

In this dissertation, an adaptive digital baseband predistortion system targeted to resource-constrained cellular handsets was studied. The results of the study contribute to the design of highly power-efficient transceivers for mobile devices by proposing a low-complexity adaptation scheme that lends itself to efficient hardware implementation. By combining the proposed adaptative predistorter with a highly nonlinear but power-efficient amplifier (e.g. class-E and above), significant power savings can be achieved while meeting the strict standard performance requirements.

10.1 Contributions

This thesis considered many different aspects of adaptive predistorter design. The primary contributions are summarized below:

- The optimal spacing of linearly-interpolated LUT predistorters was thoroughly studied. The LUT entries can be arbitrarily spaced by preceding the predistorter with a compander [11]. Optimal companders were derived for both the polar and complex-gain predistorter configurations. It was shown that the combination of linear interpolation and optimal spacing results in significant reduction in required memory space.

- A theoretically-derived constant-SNR compander that increases the supported dynamic range of LUT predistorters was also proposed. The use of this compander results in constant performance across a large input backoff range. A low-complexity approximation of the constant-SNR spacing is also proposed and its implementation in hardware is presented.

- An LMS-based complex-gain predistorter adaptation using the indirect learning architecture is presented. The update equations are optimized for efficient hardware implementation. An amplitude approximation with efficient hardware implementation is also presented.

- A novel predistorter configuration based on a generalization of the complex-gain predistorter is proposed. This new predistorter adaptively compensates the quadrature imbalances and linearizes the amplifier, while requiring the same computational complexity as the complex-gain predistorter.

- The effects of varying matching conditions on the predistorter adaptation are studied. An adaptive gain and phase normalization algorithm that mitigates these effects is presented. The proposed design reuses the predistorter update hardware to minimize the implementation costs.

Additionally, the proposed adaptive predistorter is implemented in VHDL and synthesized with the SYNOPSYS DESIGN COMPILER. The VHDL source code is provided in Appendix B.

10.2 Future Research

This dissertation can be extended in a number of different directions, including:

- The predistorter with quadrature imbalance correction presented in Chapter 8 can be extended to mitigate frequency-selective I/Q mismatches. This capability could be required for very wideband modulations (e.g. 20 MHz LTE signal).

- Even though the proposed adaptive predistorter is targeted to resource-constrained handsets, it could be adapted to high-power base station transceivers by adding memory effects mitigation capabilities.

- The predistorter adaptation schemes proposed in this thesis and most studies in the literature require an auxiliary receiver to provide a feedback signal. A potential area of future research is the design of a predistorter adjustment technique based on temperature feedback. Since temperature measurement capabilities are already available in most transceiver systems, this approach could potentially result in significant savings to the cost of implementation of adaptive digital predistorters.

APPENDIX A

LINEAR INTERPOLATION ERROR

The purpose of the following derivation is to come up with a closed-form expression for the linear interpolation error between two reference points.

Let $f(x)$ be a doubly differentiable function that is evaluated at points (x_i, y_i) with $y_i = f(x_i)$, $i \in 1, 2, \cdots, N$. The closed-form expression of the interpolation error within the interval $[x_k, x_{k+1}]$ will be derived, for $1 < k < N$. Let us assume that the interval considered is small enough that $f(x)$ can be approximated by a second order polynomial within the kth interval. This is equivalent to neglecting the error terms above the second order term in a Taylor series expansion. Consequently, the second order derivative of $f(x)$ can be considered as a constant within the interval. The previous assumption is reasonable here since otherwise a linear approximation of f in that interval would yield a very inaccurate approximation. $f(x)$ is therefore given by

$$f(x) = ax^2 + bx + c \qquad \text{for} \qquad x_k \leq x \leq x_{k+1}. \tag{A.1}$$

After a few trivial algebraic manipulations, the coefficients a, b, and c can be calculated from the two reference points as functions of $f''(x)$, x_k, x_{k+1}, y_i and y_{k+1}:

$$\begin{aligned}
a &= \frac{f''(x)}{2} \\
b &= \frac{y_k - y_{k+1}}{x_k - x_{k+1}} - \frac{f''(x)}{2}(x_k + x_{k+1}) \\
c &= \frac{x_i \times y_{k+1} - x_{k+1} \times y_k}{x_i - x_{k+1}} + \frac{f''(x)}{2}(x_k \times x_{k+1}).
\end{aligned}$$

The linear approximation of $f(x)$ in the kth interval can be formulated as follows:

$$\tilde{f}(x) = \beta x + \gamma, \tag{A.2}$$

where β and γ can also be expressed as a function of x_k, x_{i+1}, y_k and y_{k+1}:

$$\begin{aligned}\beta &= \frac{y_k - y_{k+1}}{x_k - x_{k+1}} \\ \gamma &= \frac{x_k \times y_{k+1} - x_{k+1} \times y_k}{x_k - x_{k+1}}.\end{aligned} \tag{A.3}$$

The interpolation error between $f(x)$ and $\tilde{f}(x)$ for $x_k < x < x_{k+1}$ can be calculated as follows:

$$\begin{aligned}e_i(x) &= f(x) - \tilde{f}(x) \\ &= \frac{f''(x)}{2} + x(b - \beta) + c - \gamma \\ &\vdots \\ &= \frac{f''(x)}{2}(x - x_k)(x - x_{k+1}).\end{aligned} \tag{A.4}$$

Let ε_x be the deviation between x and x_k, and d_k the width of the kth interval:

$$\varepsilon_x = x - x_k \qquad \text{and} \qquad d_k = x_{k+1} - x_k$$

The expression of the linear interpolation error can then be rewritten as follows:

$$e_i(x) = f''(x)\frac{\varepsilon_x\left(\varepsilon_x - d_k\right)}{2} \tag{A.5}$$

APPENDIX B

VHDL CODE FOR HARDWARE SYNTHESIS

This appendix lists the VHDL code used to simulate and synthesize different hardware blocks proposed in this thesis.

B.1 Base-2 spacing VHDL Code

The code in Listing B.1 implements the base-2 addressing, for an amplitude resolution of 16 bits and an interpolation factor of 6 bits. The number of intervals and the number of entries per interval are $N = 8$ and $M = 16$, respectively.

```
-- b2addr.vhd
library ieee;
  use ieee.std_logic_1164.all;

entity b2addr is
  generic(
    DW   : integer := 7;
    AW   : integer := 16;
    NI   : integer := 8;
    IW   : integer := 6);
  port(
    en   : in std_logic;
    amp  : in std_logic_vector(AW-1 downto 0); -- amplitude in
    addr : out std_logic_vector(DW-1 downto 0); -- address out
    int  : out std_logic_vector(IW-1 downto 0));
end;

architecture addr_arch of b2addr is
-- Logic OR of the bits in a std_logic_vector
function orbits(val : std_logic_vector) return std_logic is
variable result : std_logic := '0';
begin
  for i in val'low to val'high loop
    result := result or val(i);
  end loop;
  return result;
end orbits;

begin   -- architecture
  add_calc: process(amp, en)
  variable CI    : std_logic_vector(NI-1 downto 0);
  variable cum   : std_logic := '1';
  variable n     : integer := 0;
  begin
    if en = '1' then
      ------------- Intermediate Signal C -------------
      cum := '1';
      for i in 1 to NI-1 loop
        CI(NI - i) := cum and amp(AW-i);
        cum := cum and (not amp(AW - i));
      end loop;
      CI(0) := cum;
      ------------------- Compute address -------------------
      addr(6) <= orbits(CI(7 downto 4));
      addr(5) <= orbits(CI(7) & CI(6) & CI(3) & CI(2));
      addr(4) <= orbits(CI(7) & CI(5) & CI(3) & CI(1));
      addr(3) <= orbits(CI and (amp(14 downto 8) & amp(8)));
      addr(2) <= orbits(CI and (amp(13 downto 7) & amp(7)));
      addr(1) <= orbits(CI and (amp(12 downto 6) & amp(6)));
      addr(0) <= orbits(CI and (amp(11 downto 5) & amp(5)));
      ------------- Compute intolation factor  -------------
      int(5) <= orbits(CI and (amp(10 downto 4) & amp(4)));
      int(4) <= orbits(CI and (amp(09 downto 3) & amp(3)));
      int(3) <= orbits(CI and (amp(08 downto 2) & amp(2)));
      int(2) <= orbits(CI and (amp(07 downto 1) & amp(1)));
      int(1) <= orbits(CI and (amp(06 downto 0) & amp(0)));
      int(0) <= orbits(CI and (amp(05 downto 0) & "00"));
    else
      addr <= (others => '0');
      int  <= (others => '0');
    end if;
  end process;
end addr_arch;
```

Listing B.1: Synthesizable VHDL code for the base-2 address calculation.

B.2 Amplitude Approximation Code

The code in Listing B.2 illustrates the VHDL implementation of the amplitude approximation method developed in Chapter 6.

```
-- amp_approx.vhd
library ieee;
  use ieee.std_logic_1164.all;
  use ieee.numeric_std.all;

entity amp_approx is
  generic(IQ_W      : integer:=13);
  port(
    data_in_i  : in signed(IQ_W-1 downto 0);
    data_in_q  : in signed(IQ_W-1 downto 0);
    amp        : out unsigned(IQ_W-1 downto 0));
end;

architecture rtl of amp_approx is
constant cnst_11    : unsigned(3 downto 0) := "1011";
constant cnst_29    : unsigned(4 downto 0) := "11101";
constant cnst_53    : unsigned(5 downto 0) := "110101";
constant cnst_37    : unsigned(5 downto 0) := "100101";
signal absI, absQ   : unsigned(IQ_W-1 downto 0);
signal x, y         : unsigned(IQ_W-1 downto 0);
signal amp1         : unsigned(IQ_W-1 downto 0);
signal amp2         : unsigned(IQ_W+5 downto 0);
begin
  absI <= unsigned(abs(data_in_i));
  absQ <= unsigned(abs(data_in_q));
  x <= absI   when absI > absQ  else absQ;
  y <= absQ   when absI > absQ  else absI;
  amp1 <= x + shift_right(y+4, 3);
  amp2 <= shift_right(cnst_53 * x + cnst_37 * y + 32, 6);
  amp <= amp1(IQ_W-1 downto 0) when cnst_11*x > cnst_29*y else
         amp2(IQ_W-1 downto 0);
end;
```

Listing B.2: Synthesizable VHDL code for the amplitude approximation with three angular intervals.

B.3 Predistorter Code

The code in Listing B.3 shows the VHDL implementation of a predistorter using a uniformly spaced LUT.

```
-- plut.vhd
library ieee;
  use ieee.std_logic_1164.all;
  use ieee.numeric_std.all;

entity plut is
  generic( ADDR_W   : integer := 7;
           IQ_W     : integer := 13;
           GAIN_W   : integer := 8);
  port( clk       : in std_logic;
        en        : in std_logic;
        din_i     : in signed(IQ_W-1 downto 0);
        din_q     : in signed(IQ_W-1 downto 0);
        wr        : in std_logic;   -- to initialize MEM from outside
        addr0_in  : in std_logic_vector(ADDR_W-1 downto 0);
        addr1_in  : in std_logic_vector(ADDR_W-1 downto 0);
        lut0_in   : in std_logic_vector(2*GAIN_W-1 downto 0);
```

```
         lut1_in  : in std_logic_vector(2*GAIN_W-1 downto 0);
         dout_i   : out signed(IQ_W-1   downto 0);
         dout_q   : out signed(IQ_W-1   downto 0));
end;

architecture arch_plut of plut is
constant AMP_W   : integer := IQ_W-1;
constant NTRP_W  : integer := AMP_W-ADDR_W;
constant MADR    : std_logic_vector(ADDR_W-1 downto 0) :=(others
    =>'1');
constant DATA_W  : integer := 2*GAIN_W;
signal amp       : unsigned(IQ_W-1 downto 0);
signal addrn     : std_logic_vector(ADDR_W-1 downto 0);
signal addr0_amp : std_logic_vector(ADDR_W-1 downto 0);
signal addr1_amp : std_logic_vector(ADDR_W-1 downto 0);
signal addr0     : std_logic_vector(ADDR_W-1 downto 0);
signal addr1     : std_logic_vector(ADDR_W-1 downto 0);
signal ntrp      : signed(NTRP_W downto 0);
signal ntrp_d    : signed(NTRP_W downto 0);
signal ddelay_i  : signed(IQ_W-1 downto 0);
signal ddelay_q  : signed(IQ_W-1 downto 0);
signal lut0      : std_logic_vector(DATA_W-1 downto 0);
signal lut1      : std_logic_vector(DATA_W-1 downto 0);
signal xtrp      : std_logic;
signal xtrp_d    : std_logic;
signal mem_en    : std_logic;
--
component amp_approx is
  generic(IQ_W : integer);
  port( data_in_i  : in signed(IQ_W-1 downto 0);
        data_in_q  : in signed(IQ_W-1 downto 0);
        amp        : out unsigned(IQ_W-1 downto 0));
end component;
--
component ram_2p is
  generic(ADDR_W : integer; DATA_W : integer);
  port( clk        : in std_logic;
        en         : in std_logic;
        addr0      : in std_logic_vector(ADDR_W-1 downto 0);
        addr1      : in std_logic_vector(ADDR_W-1 downto 0);
        rw         : in std_logic;
        data0_in   : in std_logic_vector(DATA_W-1 downto 0);
        data1_in   : in std_logic_vector(DATA_W-1 downto 0);
        data0_out  : out std_logic_vector(DATA_W-1 downto 0);
        data1_out  : out std_logic_vector(DATA_W-1 downto 0));
end component;

begin
 -- Amplitude approximation
amp_approximation : amp_approx
  generic map(IQ_W => IQ_W)
  port map( data_in_i => din_i,
            data_in_q => din_q,
            amp    => amp);
```

```
-- ADDR CALC
  addrn <= std_logic_vector(amp(AMP_W-1 downto AMP_W-ADDR_W));
  ntrp <= signed( '0' & amp(AMP_W-ADDR_W-1 downto 0));
  xtrp <= '1' when addrn=MADR else '0';
  --- Set Addresses for interpolation or xtrpolation
  addr0_amp <= addrn  when xtrp='0' else std_logic_vector(unsigned(
      addrn)-1);
  addr1_amp <= addrn  when xtrp='1' else std_logic_vector(unsigned(
      addrn)+1);
  --
  addr0 <= addr0_amp  when wr='0' else addr0_in;
  addr1 <= addr1_amp  when wr='0' else addr1_in;
  --  Memory
  mem_en <= en or wr;
  MemDual : ram_2p
  generic map(ADDR_W => ADDR_W, DATA_W => DATA_W)
  port map( clk        => clk,
            en         => mem_en,
            addr0      => addr0,
            addr1      => addr1,
            rw         => wr,
            data0_in   => lut0_in ,
            data1_in   => lut1_in ,
            data0_out  => lut0 ,
            data1_out  => lut1);
  -- predistort
  predistort : process(en, clk)
  variable g_re    : signed(GAIN_W-1 downto 0);
  variable g_im    : signed(GAIN_W-1 downto 0);
  variable g0_re   : signed(GAIN_W-1 downto 0);
  variable g0_im   : signed(GAIN_W-1 downto 0);
  variable g1_re   : signed(GAIN_W-1 downto 0);
  variable g1_im   : signed(GAIN_W-1 downto 0);
  variable dg_re   : signed(GAIN_W+NTRP_W downto 0);
  variable dg_im   : signed(GAIN_W+NTRP_W downto 0);
  variable dtmp_i  : signed(GAIN_W+IQ_W-1 downto 0);
  variable dtmp_q  : signed(GAIN_W+IQ_W-1 downto 0);
  begin
    if en = '1' then
      if rising_edge(clk) and wr='0' then
        ntrp_d <= ntrp; -- latch interpolation factor
        ddelay_i <= din_i;
        ddelay_q <= din_q;
        xtrp_d <= xtrp;
        -- retrieve real and img parts gains from previous signal
        g0_re := signed(lut0(DATA_W-1 downto GAIN_W));
        g0_im := signed(lut0(GAIN_W-1 downto 0));
        g1_re := signed(lut1(DATA_W-1 downto GAIN_W));
        g1_im := signed(lut1(GAIN_W-1 downto 0));
        dg_re := ntrp_d * (g1_re - g0_re) + (2**NTRP_W-1);
        dg_im := ntrp_d * (g1_im - g0_im) + (2**NTRP_W-1);
        if xtrp_d ='0' then     -- linear interpolation
          g_re := g0_re + dg_re(GAIN_W+NTRP_W-1 downto NTRP_W);
          g_im := g0_im + dg_im(GAIN_W+NTRP_W-1 downto NTRP_W);
        else                        -- linear xtrpolation
          g_re := g1_re + dg_re(GAIN_W+NTRP_W-1 downto NTRP_W);
          g_im := g1_im + dg_im(GAIN_W+NTRP_W-1 downto NTRP_W);
        end if;
        -- Complex multiply + Rounding
        dtmp_i := ddelay_i * g_re - ddelay_q * g_im + (2**(GAIN_W-1)
            -1);
        dtmp_q := ddelay_i * g_im + ddelay_q * g_re + (2**(GAIN_W-1)
            -1);
        dout_i <= dtmp_i(IQ_W+GAIN_W-2 downto GAIN_W-1);
        dout_q <= dtmp_q(IQ_W+GAIN_W-2 downto GAIN_W-1);
      end if;
    else --- en =0
      dout_i <= (others => '0');
      dout_q <= (others => '0');
  end if;
end process;

end arch_plut;
```

Listing B.3: Synthesizable VHDL code for the predistorter using uniform LUT spacing.

The code in Listing B.4 shows the VHDL implementation for a predistorter using an optimal compander.

```
-- plut_comp.vhd
library ieee;
  use ieee.std_logic_1164.all;
  use ieee.numeric_std.all;

entity plut_comp is
  generic( ADDR_W       : integer := 5;
           IQ_W         : integer := 12;
           GAIN_W       : integer := 12;
           CMP_ADDR_W   : integer := 4);
  port( clk        : in std_logic;
        en         : in std_logic;
        din_i      : in signed(IQ_W-1 downto 0);
        din_q      : in signed(IQ_W-1 downto 0);
        wr_ram     : in std_logic;   -- to initialize MEM from outside
        wr_cmp     : in std_logic;   -- to initialize CMP from outside
        addr0_in   : in std_logic_vector(ADDR_W-1 downto 0);
        addr1_in   : in std_logic_vector(ADDR_W-1 downto 0);
        lut0_in    : in std_logic_vector(2*GAIN_W-1 downto 0);
        lut1_in    : in std_logic_vector(2*GAIN_W-1 downto 0);
        dout_i     : out signed(IQ_W-1 downto 0);
        dout_q     : out signed(IQ_W-1 downto 0));
end;

architecture arch_plut of plut_comp is
constant AMP_W  : integer := IQ_W-1;
constant NTRP_W : integer := AMP_W-ADDR_W;
constant MADR    : std_logic_vector(ADDR_W-1 downto 0):=(others
    =>'1');
constant DATA_W : integer := 2*GAIN_W;
signal amp       : unsigned(IQ_W-1 downto 0);
signal amp_cmp   : unsigned(AMP_W-1 downto 0);
signal addrn     : std_logic_vector(ADDR_W-1 downto 0);
signal addr0_amp: std_logic_vector(ADDR_W-1 downto 0);
signal addr1_amp: std_logic_vector(ADDR_W-1 downto 0);
signal addr0     : std_logic_vector(ADDR_W-1 downto 0);
signal addr1     : std_logic_vector(ADDR_W-1 downto 0);
signal ntrp      : signed(NTRP_W downto 0);
signal ntrp_d    : signed(NTRP_W downto 0);
signal ddelay_i  : signed(IQ_W-1 downto 0);
signal ddelay_q  : signed(IQ_W-1 downto 0);
signal lut0      : std_logic_vector(DATA_W-1 downto 0);
signal lut1      : std_logic_vector(DATA_W-1 downto 0);
signal xtrp      : std_logic;
signal xtrp_d    : std_logic;
signal mem_en    : std_logic;
-- Amplitude Approximation
component amp_approx is
  generic(IQ_W : integer);
  port( data_in_i  : in signed(IQ_W-1 downto 0);
        data_in_q  : in signed(IQ_W-1 downto 0);
        amp     : out unsigned(IQ_W-1 downto 0));
end component;
-- Companding
component companding is
  generic(AMP_W : integer; ADDR_W : integer);
  port( clk      : in std_logic;
        en       : in std_logic;
        amp_in   : in unsigned(AMP_W-1 downto 0);
        wr       : in std_logic;    -- to initialize MEM
        lut0_in  : in std_logic_vector(AMP_W-1 downto 0);
        lut1_in  : in std_logic_vector(AMP_W-1 downto 0);
        amp_out  : out unsigned(AMP_W-1   downto 0));
end component;
-- Two-Port RAM
component ram_2p is
  generic(ADDR_W  : integer; DATA_W   : integer);
  port( clk      : in std_logic;
        en       : in std_logic;
        rw       : in std_logic;
        addr0    : in std_logic_vector(ADDR_W-1 downto 0);
        addr1    : in std_logic_vector(ADDR_W-1 downto 0);
        d0_in    : in std_logic_vector(DATA_W-1 downto 0);
        d1_in    : in std_logic_vector(DATA_W-1 downto 0);
        d0_out   : out std_logic_vector(DATA_W-1 downto 0);
        d1_out   : out std_logic_vector(DATA_W-1 downto 0));
end component;

begin
  -- Amplitude approximation
  amp_approximation : amp_approx
```

```vhdl
  generic map(IQ_W => IQ_W)
  port map( data_in_i => din_i,
            data_in_q => din_q,
            amp       => amp);
  -- Companding
  comp : companding
  generic map(AMP_W => AMP_W, ADDR_W  => CMP_ADDR_W)
  port map( clk      => clk,
            en       => en,
            wr       => wr_cmp,
            amp_in   => amp(AMP_W-1 downto 0),
            lut0_in  => lut0_in(AMP_W-1 downto 0),
            lut1_in  => lut1_in(AMP_W-1 downto 0),
            amp_out  => amp_cmp);
  --------------------- ADDR CALC ---------------------
  addrn <= std_logic_vector(amp_cmp(AMP_W-1 downto AMP_W-ADDR_W));
  ntrp  <= signed('0' & amp_cmp(AMP_W-ADDR_W-1 downto 0));
  xtrp  <= '1'  when addrn=MADR else  '0';
  -- Set Addresses for interpolation or xtrpolation
  addr0_amp <= addrn  when xtrp='0' else std_logic_vector(unsigned(
      addrn)-1);
  addr1_amp <= addrn  when xtrp='1' else std_logic_vector(unsigned(
      addrn)+1);
  addr0 <= addr0_amp  when wr_ram='0' else addr0_in;
  addr1 <= addr1_amp  when wr_ram='0' else addr1_in;
-- Memory
  mem_en <= en or wr_ram;
  MemDual : ram_2p
  generic map(ADDR_W => ADDR_W, DATA_W => DATA_W)
  port map( clk      => clk,
            en       => mem_en,
            addr0    => addr0,
            addr1    => addr1,
            rw       => wr_ram,
            d0_in    => lut0_in,
            d1_in    => lut1_in,
            d0_out   => lut0,
            d1_out   => lut1);
-- predistort
predistort : process(en, clk)
variable g_re    : signed(GAIN_W-1 downto 0);
variable g_im    : signed(GAIN_W-1 downto 0);
variable g0_re   : signed(GAIN_W-1 downto 0);
variable g0_im   : signed(GAIN_W-1 downto 0);
variable g1_re   : signed(GAIN_W-1 downto 0);
variable g1_im   : signed(GAIN_W-1 downto 0);
variable dg_re   : signed(GAIN_W+NTRP_W downto 0);
variable dg_im   : signed(GAIN_W+NTRP_W downto 0);
variable dtmp_i  : signed(GAIN_W+IQ_W-1 downto 0);
variable dtmp_q  : signed(GAIN_W+IQ_W-1 downto 0);
--
begin
  if en = '1' then
    if rising_edge(clk) and wr_ram='0' then
```

```vhdl
        ntrp_d <= ntrp; -- latch interpolation factor
        ddelay_i <= din_i;
        ddelay_q <= din_q;
        xtrp_d <= xtrp;
        -- retrieve real and img parts gains from previous signal
        g0_re :=  signed(lut0(DATA_W-1 downto GAIN_W));
        g0_im :=  signed(lut0(GAIN_W-1 downto 0));
        g1_re :=  signed(lut1(DATA_W-1 downto GAIN_W));
        g1_im :=  signed(lut1(GAIN_W-1 downto 0));
        dg_re := ntrp_d * (g1_re - g0_re) + (2**NTRP_W-1);
        dg_im := ntrp_d * (g1_im - g0_im) + (2**NTRP_W-1);
        if xtrp_d ='0' then    -- linear interpolation
          g_re := g0_re + dg_re(GAIN_W+NTRP_W-1 downto NTRP_W);
          g_im := g0_im + dg_im(GAIN_W+NTRP_W-1 downto NTRP_W);
        else                          -- linear xtrpolation
          g_re := g1_re + dg_re(GAIN_W+NTRP_W-1 downto NTRP_W);
          g_im := g1_im + dg_im(GAIN_W+NTRP_W-1 downto NTRP_W);
        end if;
        -- Complex multiply + Rounding
        dtmp_i := ddelay_i * g_re - ddelay_q * g_im + (2**(GAIN_W-1)
            -1);
        dtmp_q := ddelay_i * g_im + ddelay_q * g_re + (2**(GAIN_W-1)
            -1);
        dout_i <= dtmp_i(IQ_W+GAIN_W-2 downto GAIN_W-1);
        dout_q <= dtmp_q(IQ_W+GAIN_W-2 downto GAIN_W-1);
      end if;
    else --- en =0
      dout_i <= (others => '0');
      dout_q <= (others => '0');
      end if;
end process;
end arch_plut;
```

Listing B.4: Synthesizable VHDL code for the predistorter using an optimal compander.

The code in Listing B.5 implements the optimal compander used in the predistorter VHDL code (Listing B.4).

```vhdl
-- companding.vhd
library ieee;
  use ieee.std_logic_1164.all;
  use ieee.numeric_std.all;

entity companding is
  generic(AMP_W : integer := 12; ADDR_W : integer := 8);
  port( clk       : in std_logic;
        en        : in std_logic;
        amp_in    : in unsigned(AMP_W-1 downto 0);
        wr        : in std_logic;   -- to initialize MEM from outside
        lut0_in   : in std_logic_vector(AMP_W-1 downto 0);
        lut1_in   : in std_logic_vector(AMP_W-1 downto 0);
        amp_out   : out unsigned(AMP_W-1    downto 0));
```

```vhdl
end;

architecture arch_comp of companding is
constant NTRP_W : integer := AMP_W-ADDR_W;
constant MADR : std_logic_vector(ADDR_W-1 downto 0) := (others =>
     '1');
signal addr0  : std_logic_vector(ADDR_W-1 downto 0);
signal addr1  : std_logic_vector(ADDR_W-1 downto 0);
signal ntrp   : unsigned(NTRP_W-1 downto 0);
signal ntrp_d : unsigned(NTRP_W-1 downto 0);
signal lut0   : std_logic_vector(AMP_W-1 downto 0);
signal lut1   : std_logic_vector(AMP_W-1 downto 0);
signal xtrp   : std_logic;
signal xtrp_d : std_logic;
signal mem_en : std_logic;

-- Memory component
component ram_2p is
  generic(ADDR_W  : integer; DATA_W : integer);
  port( clk       : in std_logic;
        en        : in std_logic;
        addr0     : in std_logic_vector(ADDR_W-1 downto 0);
        addr1     : in std_logic_vector(ADDR_W-1 downto 0);
        rw        : in std_logic;
        data0_in  : in std_logic_vector(AMP_W-1 downto 0);
        data1_in  : in std_logic_vector(AMP_W-1 downto 0);
        data0_out : out std_logic_vector(AMP_W-1 downto 0);
        data1_out : out std_logic_vector(AMP_W-1 downto 0));
end component;

begin
  -- ADDR CALC
    addr0 <= std_logic_vector(amp_in(AMP_W-1 downto AMP_W-ADDR_W));
    addr1 <= std_logic_vector(unsigned(addr0)+1);
    ntrp <= amp_in(AMP_W-ADDR_W-1 downto 0);
    xtrp <= '1' when addr0=MADR else '0';
  -- Memory
  mem_en <= en or wr;
  MemDual : ram_2p
    generic map(ADDR_W => ADDR_W, DATA_W => AMP_W)
    port map( clk => clk,
              en        => mem_en,
              addr0     => addr0,
              addr1     => addr1,
              rw        => wr,
              data0_in  => lut0_in,
              data1_in  => lut1_in,
              data0_out => lut0,
              data1_out => lut1);
-- predistort
  predistort : process(en, clk)
  constant MS : unsigned(AMP_W-1 downto 0):= (others => '1');
  variable s  : unsigned(AMP_W-1 downto 0);
  variable s0 : unsigned(AMP_W-1 downto 0);
```

```
    variable s1 : unsigned(AMP_W-1 downto 0);
    variable ds : unsigned(AMP_W+NTRP_W-1 downto 0);
    begin
      if en = '1' then
        if rising_edge(clk) and wr='0' then
          xtrp_d     <= xtrp;
          ntrp_d <= ntrp; -- latch interpolation factor
          -- retrieve prev real & img parts gains
          s0 := unsigned(lut0);
          if xtrp_d = '0' then  -- interpolating
            s1 := unsigned(lut1);
          else
            s1 := MS;       -- xtrpolating
          end if;
          ds := ntrp_d * (s1-s0) + 2**NTRP_W-1;
          s := s0 + ds(AMP_W+NTRP_W-1 downto NTRP_W);
          amp_out <= s;
        end if;
      else -- en =0
        amp_out <= (others=>'0');
      end if;
    end process;
  end arch_comp;
```

Listing B.5: Synthesizable VHDL code for the LUT compander.

B.4 Predistorter Adaptation Code

The code in Listing B.6 shows the VHDL implementation of the low complexity adaptation algorithm proposed in Chapter 7.

```
-- adapt.vhd
library ieee;
  use ieee.std_logic_1164.all;
  use ieee.numeric_std.all;

entity pd_adapt is
  generic(ADDR_W: integer:=7; IQ_W: integer:=13; AMP_W: integer
      :=12;
          SC_AMP_W: integer:=21; NTRP_W: integer:=6; GAIN_W: integer
              :=12);
  port( en         : in std_logic;
        en_update  : in std_logic;
        clk_1      : in std_logic;                              --
            typical 30Mhz
        clk_2      : in std_logic;                              --
            half of clk_1 freq
        dat_ff_i   : in signed(IQ_W-1 downto 0);                 --
            From FF Path
        dat_ff_q   : in  signed(IQ_W-1 downto 0);              --
            From FF Path
        dat_fb_i   : in  signed(IQ_W-1 downto 0);              --
            From FB Path
        dat_fb_q   : in  signed(IQ_W-1 downto 0);              --
            From FB Path
        lut0_in    : in  std_logic_vector(2*GAIN_W-1 downto 0);  --
            From LUT_MEM UNIT
        lut1_in    : in  std_logic_vector(2*GAIN_W-1 downto 0);  --
            From LUT_MEM UNIT
        mu_adapt   : in  unsigned(3 downto 0);                 --
            update factor
        rw         : out std_logic:='0';                        --
            To LUT_MEM UNIT
        addr0_out  : out unsigned(ADDR_W-1 downto 0);           --
            To LUT_MEM UNIT
        addr1_out  : out unsigned(ADDR_W-1 downto 0);           --
            To LUT_MEM UNIT
        lut0_out   : out std_logic_vector(2*GAIN_W-1 downto 0);  --
            To LUT_MEM UNIT
        lut1_out   : out std_logic_vector(2*GAIN_W-1 downto 0)); --
            To LUT_MEM UNIT
end;

architecture arch_adapt of pd_adapt is
signal plut_out_i : signed(IQ_W-1 downto 0);
signal plut_out_q : signed(IQ_W-1 downto 0);
signal ntrp       : signed(NTRP_W downto 0);
signal clk_3      : std_logic;
begin
  clk_3 <= not clk_2;
  rw <= clk_3;
  plut_fb : entity work.pd_plut
  generic map(ADDR_W=>ADDR_W, IQ_W=>IQ_W, AMP_W=>AMP_W, SC_AMP_W=>
      SC_AMP_W,
              NTRP_W=>NTRP_W, GAIN_W=>GAIN_W, KP_W=>KP_W)
  port map( en         => en,
            clk        => clk_1,
            dat_in_i   => dat_fb_i,
            dat_in_q   => dat_fb_q,
            -- in from mem
            lut0_in    => lut0_in,
            lut1_in    => lut1_in,
            -- out to mem
            addr0_out  => addr0_out,
            addr1_out  => addr1_out,
            -- out to ppa
            dat_out_i  => plut_out_i,
            dat_out_q  => plut_out_q,
            ntrp_out   => ntrp);
        -- compute error and apply update factor mu
  latch_err : process(clk_3)
  variable er_i, er_q      : signed(IQ_W-1 downto 0) :=(others
      =>'0');
```

```
    variable d_i, d_q          : signed(IQ_W-1 downto 0) :=(others
        =>'0');
    variable d0_i, d0_q        : signed(IQ_W-1 downto 0) :=(others
        =>'0');
    variable d1_i, d1_q        : signed(IQ_W-1 downto 0) :=(others
        =>'0');
    variable sc_d_i,sc_d_q     : signed(IQ_W+NTRP_W  downto 0):=(
        others =>'0');
    variable ss_i, ss_q        : signed (1 downto 0) := "00";
    variable sign_i, sign_q    : signed(IQ_W-1 downto 0) :=(others
        =>'0');
    variable newg0_i, newg0_q  : signed(GAIN_W-1 downto 0) :=(others
        =>'0');
    variable newg1_i, newg1_q  : signed(GAIN_W-1 downto 0) :=(others
        =>'0');
    variable rr_i, rr_q        : signed(1 downto 0) := "01"; --{1 or
        -1}
    begin
      if rising_edge(clk_3) then
        er_i := shift_right(dat_ff_i - plut_out_i, to_integer(mu_adapt
            ));
        er_q := shift_right(dat_ff_q - plut_out_q, to_integer(mu_adapt
            ));
        sign_i    := (others => dat_fb_i(IQ_W-1));
        sign_q    := (others => dat_fb_q(IQ_W-1));
        ss_i(0)   := dat_fb_i(IQ_W-1);
        ss_q(0)   := dat_fb_q(IQ_W-1);
        -- rotate by k*pi/4 replace mult with 2's complements
        d_i    := ((er_i xor sign_i) + ss_i)  +  ((er_q xor sign_q) +
              ss_q);
        d_q    := ((er_q xor sign_i) +  ss_i) - ((er_i xor sign_q) +
            ss_q);
        -- scale d_i and d_q by update factor for LUT(n+1) update
        sc_d_i := d_i * ntrp;
        sc_d_q := d_q * ntrp;
        d1_i   := sc_d_i(IQ_W+NTRP_W downto NTRP_W+1);
        d1_q   := sc_d_q(IQ_W+NTRP_W downto NTRP_W+1);
        -- avoid multiplies by using subtraction instead of multiply by (1-ntrp
            )
        rr_i(1)  := d_i(IQ_W-1);              -- sign bit
        rr_q(1)  := d_q(IQ_W-1);              -- sign bit
        d0_i      := shift_right(d_i+rr_i,1) - d1_i;
        d0_q      := shift_right(d_q+rr_q,1) - d1_q;
        -- Computes updates at address n
        newg0_i := signed(lut0_in(2*GAIN_W-1 downto GAIN_W)) + d0_i(
            IQ_W-1 downto IQ_W-GAIN_W);
        newg0_q := signed(lut0_in(  GAIN_W-1 downto 0))       + d0_q(
            IQ_W-1 downto IQ_W-GAIN_W);
        -- Computes updates at address n+1
        newg1_i := signed(lut1_in(2*GAIN_W-1 downto GAIN_W)) + d1_i(
            IQ_W-1 downto IQ_W-GAIN_W);
        newg1_q := signed(lut1_in(  GAIN_W-1 downto 0))       + d1_q(
            IQ_W-1 downto IQ_W-GAIN_W);
        -- ouput new updated entries
        if en_update = '1' then
          lut0_out <= std_logic_vector(newg0_i) & std_logic_vector(
              newg0_q);
          lut1_out <= std_logic_vector(newg1_i) & std_logic_vector(
              newg1_q);
        else
          lut0_out  <= lut0_in;
          lut1_out  <= lut1_in;
        end if;
      end if;
    end process;
  end arch_adapt;
```

Listing B.6: Synthesizable VHDL code for the predistorter update logic.

REFERENCES

[1] Ba, S. N., Waheed, K., and T., Z. G., "Optimal spacing of a linearly-interpolated complex-gain LUT predistorter," *IEEE Trans. Veh. Technol., accepted for future publication*, 2009.

[2] Ba, S. N., Waheed, K., and Zhou, G. T., "Optimal spacing for a polar lookup table predistorter," in *Proc. IEEE Northeast Workshop on Circuits and Systems*, pp. 189–192, Aug. 2007.

[3] Ba, S. N., Waheed, K., and Zhou, G. T., "Efficient spacing scheme for a linearly interpolated lookup table predistorter," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1512–1515, May 2008.

[4] Besbes, H., Le-Ngoc, T., and Lin, H., "A fast adaptive predistorter for nonlinearly amplified M-QAM signals," in *IEEE Global Telecommunications Conference. GLOBCOM '00*, vol. 1, pp. 108–112, 2000.

[5] Besbes, H., Le-Ngoc, T., and Lin, H., "A fast adaptive polynomial predistorter for power amplifiers," in *IEEE Global Telecommunications Conference. GLOBCOM '01*, vol. 1, pp. 659–663, 2001.

[6] Boumaiza, S., Li, J., Jaidane-Saidane, M., and Ghannouchi, F., "Adaptive digital/RF predistortion using a nonuniform LUT indexing function with built-in dependence on the amplifier nonlinearity," *IEEE Trans. Microw. Theory Tech.*, vol. 52, pp. 2670–2677, Dec. 2004.

[7] Braun, F. and Blaser, H., "Digital hardware for approximating to the amplitude of quadrature pairs," *Electronics Letters*, vol. 10, pp. 255–256, 27 1974.

[8] Cavers, J. K., "Amplifier linearization using a digital predistorter with fast adaptation and low memory requirements," *IEEE Trans. Veh. Technol.*, vol. 39, pp. 374–382, Nov. 1990.

[9] Cavers, J. K., "The effect of quadrature modulator and demodulator errors on adaptive digital predistorters for amplifier linearization," *IEEE Transactions on Vehicular Technology*, vol. 46, pp. 456–466, May 1997.

[10] Cavers, J. K., "New methods for adaptation of quadrature modulators and demodulators in amplifier linearization circuits," *IEEE Transactions on Vehicular Technology*, vol. 46, pp. 707–716, Aug. 1997.

[11] Cavers, J. K., "Optimum table spacing in predistorting amplifier linearizers," *IEEE Trans. Veh. Technol.*, vol. 48, pp. 1699–1705, Sep. 1999.

[12] Cavers, J. K. and Liao, M. W., "Adaptive compensation for imbalance and offset losses in direct conversion transceivers," *IEEE Transactions on Vehicular Technology*, vol. 42, pp. 581–588, Nov. 1993.

[13] Clark, C. J., Chrisikos, G., Muha, M. S., Moulthrop, A. A., and Silva, C. P., "Time-domain envelope measurement technique with application to wideband power amplifier modeling," *IEEE Transactions on Microwave Theory and Techniques*, vol. 46, pp. 2531–2540, Dec. 1998.

[14] Cripps, S. C., *RF Power Amplifiers for Wireless Communications.* Artech House Publishers, 1999.

[15] Crum, L. and Wu, S., "Convergence of the quantizing learning method for system identification," *IEEE Transactions on Automatic Control*, vol. 13, pp. 297–298, June 1968.

[16] Ding, L., Raich, R., and Zhou, G. T., "A hammerstein predistortion linearization design based on the indirect learning architecture," in *Acoustics, Speech, and Signal Processing, 2002. Proceedings. (ICASSP '02). IEEE International Conference on*, vol. 3, 2002.

[17] Ding, L., Zhou, G. T., Morgan, D. R., Ma, Z., Kenney, J. S., Kim, J., and Giardina, C. R., "Memory polynomial predistorter based on the indirect learning architecture," in *Global Telecommunications Conference, 2002. GLOBECOM '02. IEEE*, vol. 1, pp. 967–971, Nov. 2002.

[18] Eskinat, E., Johnson, S. H., and Luyben, W. L., "Use of hammerstein models in identification of nonlinear systems," *AIChE Journal*, vol. 37, pp. 255–268, Feb. 2000.

[19] Eun, C. and Powers, E. J., "A new volterra predistorter based on the indirect learning architecture," *Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on]*, vol. 45, pp. 223–227, Jan. 1997.

[20] Eun, C. and Powers, E. J., "A new volterra predistorter based on the indirect learning architecture," *IEEE Transactions on Signal Processing*, vol. 45, pp. 223–227, Jan. 1997.

[21] Faulkner, M. and Johansson, M., "Adaptive linearization using predistortion-experimental results," *IEEE Transactions on Vehicular Technology*, vol. 43, pp. 323–332, May 1994.

[22] Filip, A. E., "Linear approximations to $\sqrt{x^2 + y^2}$ having equiripple error characteristics," *IEEE Transactions on Audio and Electro-acoustics*, vol. 21, pp. 554–556, Dec. 1973.

[23] Filip, A. E., "A baker's dozen magnitude approximations and their detection statistics," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-12, pp. 86–89, Jan. 1976.

[24] Gersho, A., "Principles of quantization," *IEEE Transactions on Circuits Systems*, vol. 25, pp. 427–436, July 1978.

[25] Ghaderi, M., Kumar, S., and Dodds, D. E., "Fast adaptive predistortion lineariser using polynomial functions," in *Electronics Letters*, vol. 29, pp. 1526–1528, 1993.

[26] Ghaderi, M., Kumar, S., and Dodds, D. E., "Adaptive predistortion lineariser using polynomial functions," in *IEEE Proceedings- Communications*, vol. 141, pp. 49–55, 1994.

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

[28] Graboski, J. and Davis, R. C., "An experimental MQAM modem using amplifier linearization and baseband equalization techniques," in *in Proceedings National Communications Conference*, pp. E3.2.1–E3.2.6, 1982.

[29] Hassani, J. Y. and Kamarei, M., "A flexible method of LUT indexing in digital predistortion linearization of RF power amplifiers," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 53–56, May 2001, vol. 1.

[30] Kenington, P. B., *High Linearity RF Amplifier Design*. Artech House Publishers, 2000.

[31] Kim, J. and Konstantinou, K., "Digital predistortion of wideband signals based on power amplifier model with memory," *Electronics Letters*, vol. 37, pp. 1417–1418, Nov. 2001.

[32] Ku, H., McKinley, M. D., and Kenney, J. S., "Quantifying memory effects in RF power amplifiers," *IEEE Transactions on Microwave Theory and Techniques*, vol. 50, pp. 2843–2849, Dec. 2002.

[33] Ku, H. and Kenney, J. S., "Analysis of ACPR performance for memoryless predistorter considering power amplifier memory effects," in *Microwave Symposium Digest, 2004 IEEE MTT-S International*, vol. 3, pp. 1863–1866, June 2004.

[34] Lazzarin, G., Pupolin, S., and Sarti, A., "Nonlinearity compensation in digital radio systems," *IEEE Transactions on Communications*, vol. 42, pp. 988–999, Feb./Apr. 1994.

[35] Lin, C.-H., Chen, H.-H., Wang, Y.-Y., and Chen, J.-T., "Dynamically optimum lookup-table spacing for power amplifier predistortion linearization," *IEEE Transactions on Microwave Theory and Techniques*, vol. 54, pp. 2118–2127, May 2006.

[36] Mao, W.-J., Ran, L.-X., and Chen, K.-S., "Adaptive predistortion for RF power amplifier based on new look-up table indexing method," in *Proc. 3rd International Conf. on Microwave and Millimeter Wave Technology*, pp. 932–935, Aug. 2002.

[37] Moschner, J. L., *Adaptive filtering with clipped input data*. PhD thesis, Stanford University, Stanford, CA, USA, June 1970.

[38] Muhonen, K. J., Kavehrad, M., and Krishnamoorthy, R., "Look-up table techniques for adaptive digital predistortion: a development and comparison," *IEEE Transactions on Vehicular Technology*, vol. 49, pp. 1995–2002, Sep. 2000.

[39] Nagata, Y., "Linear amplification technique for digital mobile communications," in *Proc. IEEE Vehicular Technology Conference*, vol. 1, pp. 159–164, May 1989.

[40] Onoe, M., "Fast amplitude approximation yielding either exact mean or minimum deviation for quadrature pairs," *Proceedings of the IEEE*, vol. 60, pp. 921–922, July 1972.

[41] Raich, R. and Zhou, G. T., "Analyzing spectral regrowth of QPSK and OQPSK signals," in *Proc. IEEE International Conf. on Acoustics, Speech, and Signal Processing (ICASSP '01)*, vol. 4, pp. 2673–2676, 2001.

[42] Raich, R., Qian, H., and Zhou, G. T., "Orthogonal polynomials for power amplifier modeling and predistorter design," *IEEE Transactions on Vehicular Technology*, vol. 53, pp. 1468–1479, Sept. 2004.

[43] Saleh, A. and Salz, J., "Adaptive linearization of power amplifiers in digital radio systems," *Bell Systems Technical Journal*, vol. 62, pp. 1019–1033, Apr. 1983.

[44] Schetzen, M., ed., *The Volterra and Wiener Theories of Nonlinear Systems.* New York: Wiley, 1980.

[45] Stonick, J. T., Stonick, V. L., Moura, J. M. F., and Zborowski, R. S., "Memoryless polynomial adaptive predistortion," in *International Conference on Acoustics, Speech, and Signal Processing. ICASSP-95*, vol. 2, pp. 981–984, May 1995.

[46] Sundstrom, L., Faulkner, M., and Johansson, M., "Quantization analysis and design of a digital predistortion linearizer for RF power amplifiers," *IEEE Transactions Vehicular Technology*, vol. 45, pp. 707–719, Nov. 1996.

[47] Teikari, I., Vankka, J., and Halonen, K., "Baseband digital predistorter with quadrature error correction," in *Proc. Norchip Conference*, pp. 159–162, Nov. 2004.

[48] Vuolevi, J. H. K., Rahkonen, T., and Manninen, J. P. A., "Measurement technique for characterizing memory effects in RF power amplifiers," *IEEE Transactions on Microwave Theory and Techniques*, vol. 49, pp. 1383–1389, Aug. 2001.

[49] Waheed, K. and Ba, S. N., "Adaptive digital linearization of a DRP-based EDGE transmitter for cellular handsets," in *Proc. IEEE Midwest Symposium on Circuits and Systems*, pp. 706–709, Aug. 2007.

[50] White, M., Mack, I., Borsuk, G., Lampe, D., and Kub, F., "Charge-coupled device (CCD) adaptive discrete analog signal processing," *IEEE Transactions on Communications*, vol. 27, pp. 390–405, Feb. 1979.

[51] Widrow, B. and Stearns, S., *Introduction to Magnetic Materials.* Englewood Cliffs, NJ: Prentice Hall, 1985.

[52] Zhou, G. T., "Analysis of spectral regrowth of weakly nonlinear power amplifiers," in *Acoustics, Speech, and Signal Processing, 2000. ICASSP '00. Proceedings. 2000 IEEE International Conference on*, vol. 5, pp. 2737–2740, 2000.

[53] Zhou, G. T., "Analysis of spectral regrowth of weakly nonlinear power amplifiers," *IEEE Communication Letters*, vol. 4, pp. 357–359, Nov. 2000.