Digital predistortion using stochastic conjugate gradient method

Hong Jiang, Paul Wilford and Xin Yu[1]
Bell Laboratories, Alcatel-Lucent
{hong.jiang, paw,xin.xy.yu}@alcatel-lucent.com

***Abstract*— This paper is concerned with digital predistortion for linearization of RF high power amplifiers (HPAs). An iterative stochastic conjugate gradient (SCG) method is used in computing the predistorter in the digital predistortion. The SCG has the advantage of reducing the complexity and, at the same time, increasing the convergence rate and stability of iteration. We present some results of using the SCG in a hardware platform with a solid state HPA.**

***Index Terms*— digital predistortion, power amplifier linearization, predistorter, postdistorter, stochastic conjugate gradient method**

I. INTRODUCTION

Linearization of RF high power amplifiers (HPAs) using digital predistortion has been widely studied in the literature and has been routinely implemented in telecommunication and broadcast systems. In a digital predistortion system, the transmitted signal is processed by a predistorter in the digital domain before it is converted to analog, upconverted to RF, and amplified. The purpose of the predistorter is to compensate nonlinear effects of the HPA, which include distorting signal constellation and spreading signal spectrum. Consequently, the use of digital predistortion can increase the efficiency of HPAs by transmitting at high output power without suffering from undesired nonlinear effects that impact the system performance.

In predistortions such as the polynomial and generalized polynomial predistortions [1,2], the predistorter is usually computed by using the least squares method. The predistorter takes the form of a linear combination of a set of predetermined basis functions, for example, orthogonal polynomials [3,4]. The coefficients of the predistorter are computed by solving the normal equations resulting from the least squares approximation of the input and output samples of the HPA. The normal equations for the coefficients of the predistorter can be solved by an iterative method such as the conjugate gradient (CG) method. An iterative method for solving the normal equations has many advantages over a direct method such as the Cholesky decomposition. When the CG method is used to solve the normal equations, the predistortion process becomes an inner-outer loop. In the outer loop, each iteration consists of taking a set of samples from the input and output of the HPA and forming the normal equations with the samples. The inner loop is the CG iteration for solving the normal equations. The CG iteration reduces the residual in the solution, and it is well known that the CG iteration in the inner loop will converge in a finite number of iterations (the number of iterations is no more than the rank of the matrix of the normal equations). In practice, the residual may be already small enough, i.e., solution may be already accurate enough, after only a few CG iterations in the inner loop.

In this paper, we propose to use a stochastic conjugate gradient (SCG) method to solve the normal equations for the coefficients of the predistorter. In the SCG method, only one iteration of CG is performed in the inner loop. That is, the predistortion process consist of capturing a set of input and output samples from HPA, performing one iteration of the conjugate gradient algorithm to obtain an update of the coefficients, and using the updated coefficients in the predistorter, and then repeating the process again. In the SCG method, the residuals and search directions are computed by evaluating the basis functions at the HPA input/output samples, and the matrix of the normal equations is not explicitly computed. Avoiding the computation of the normal matrix represents a significant reduction in the complexity, and helps speeding up the solution process. Since the coefficients of the predistorter are updated after only one iteration, this allows the predistorter to quickly adapt to the transient behavior of the HPA.

The proposed methods have been implemented on a hardware platform with predistortion for a multicarrier UMTS system, and the results will be presented.

II. STOCHASTIC CONJUGATE GRADIENT METHOD

A digital predistortion system using memory polynomials is shown in Figure 1. Here x_n are complex samples of the baseband signal in the time domain, n is the time sample index, z_n are complex samples after the predistorter, and y_n are the digitized complex samples of the signal after the HPA. The HPA is assumed to have memory effects. Since we are interested in digital processing, we combine the processing of DAC, upconverter, HPA, feedback, downconverters and ADC into one function F. That is, F is an operator mapping the sequence of complex numbers $\{z_n, n = 0,1,...\}$ to the sequence of complex numbers $\{y_n, n = 0,1,...\}$ shown in Figure 1. We informally call F the transfer function of the HPA, and are interested in predistortion of this function.

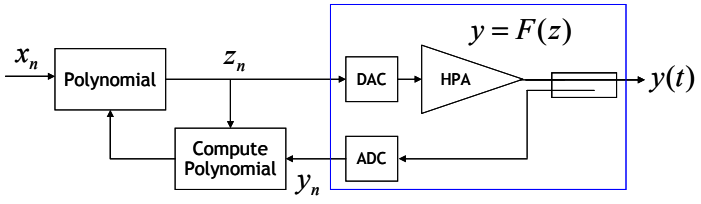

Figure 1. Polynomial predistorter

For HPA with memory effects, it has been demonstrated [1,2] that the inverse of the HPA can be approximated by a memory polynomial of the form

[1] Hong Jiang and Paul Wilford are with Bell Labs, Alcatel-Lucent, 700 Mountain Ave, Murray Hill, NJ 07974, USA. Xin Yu is with Alcatel-Lucent AG, Bell Labs Germany, Radio Communication - ZFZ/WA4, Lorenzstraße 10, 70435 Stuttgart, Germany.

CHINACOM 2010, August 25-27, Beijing, China

DOI 10.4108/chinacom.2010.71

$$z_n = P(x_n,...,x_{n-Q+1}) = \sum_{q=1}^{Q} x_{n-q+1} P_q(|x_{n-q+1}|). \quad (2.1)$$

In (2.1), $P_q(\cdot)$ is a polynomial of a degree less than M. The objective of the polynomial computation is to find polynomials $P_q(\cdot)$, $q = 1,...,Q$, so that the pair of sample sets $\{x_n\}$, $\{z_n\}$ from (2.1) best match the sample sets from observing the output and input signals of the HPA. In this context, the inverse of HPA is regarded to be a multivariate function in which an independent variable presents the current memory state or a memory state in the past.

In the following, we will establish a general framework for computing the predistorter. We generalize the predistorter of (2.1) to a linear combination of arbitrary linearly independent basis functions, and derive algorithms for computing the best predistorter.

We first introduce some formal definitions. Let $g(x^1,...,x^Q)$ be the complex valued function of Q complex variables representing the inverse of the HPA transfer function F, where each independent variable of $g(x^1,...,x^Q)$ represents a memory delay in samples. Let $Y^1,...,Y^Q$ be complex valued random processes. Each of $|Y^q|$ has the same probability density function $\rho(x)$. Let $\mathbf{Y} = [Y^1,...,Y^Q]$. Define the random variable $Z = g(\mathbf{Y}) = g(Y^1,...,Y^Q)$. Define the inner product of two multivariate functions by

$$\langle u, v \rangle = \mathbf{E}(u^* v), \quad (2.2)$$

where $\mathbf{E}(\cdot)$ is the expected value, and u^* is the complex conjugate of u.

Let $\{\psi_0(|x|),...,\psi_{M-1}(|x|)\}$ be a set of linearly independent complex valued functions of one real variable defined on the interval $[0,1]$. Let $\{\tau_1(x),...,\tau_Q(x)\}$ be a given set of complex valued functions of one complex variable. A generalization of the memory polynomial predistorter (2.1) is given by

$$z_n = P(x_n,..,x_{n-Q+1}) = \sum_{q=1}^{Q} \tau_q(x_{n-q+1}) \sum_{i=0}^{M-1} u_i^q \psi_i(|x_{n-q+1}|).$$

The memory polynomial predistorter in (2.1) becomes a special case of above with $\tau_q(x) = x$, $q = 1,...,Q$ and $\psi_i(x) = x^i$, $i = 0,...,M-1$.

With these definitions, finding the best predistorter is tantamount to finding $\overline{u}^q(|x|) = \sum_{i=0}^{M-1} \overline{u}_i^q \psi_i(|x|)$, $q = 1,...,Q$ such that the following expression is minimized

$$\mathbf{E}\left(\left(g(\mathbf{Y}) - \sum_{q=1}^{Q} \tau_q(Y^q) \overline{u}^q(|Y^q|) \right)^* \cdot \left(g(\mathbf{Y}) - \sum_{q=1}^{Q} \tau_q(Y^q) \overline{u}^q(|Y^q|) \right) \right). \quad (2.3)$$

The coefficients $\overline{u}_i^q$ of the solution $\overline{u}^q(|x|), q = 1,...,Q$ minimizing (2.3) satisfy the normal equations

$$Au = b, \quad (2.4)$$

where $A = A(\psi)$ is the $QM \times QM$ covariance matrix whose components $a_{n,m}$ are given by

$$a_{p(Q-1)+i,\,q(Q-1)+j} = \mathbf{E}\left((\tau_p \psi_i)^* \tau_q \psi_j \right), \quad \begin{matrix} p,q = 1,...,Q \\ i,j = 0,...,M-1 \end{matrix}.$$

The vector $u = [u_0^1,...,u_{M-1}^1,...,u_0^Q,...,u_{M-1}^Q]^T$ is a complex QM-tuplets representing the coefficients of the basis functions. The right hand side (RHS)

$$b = [b_0^1,...,b_{M-1}^1,...,b_0^Q,...,b_{M-1}^Q]^T$$

is also a complex QM-tuplets whose components are given by

$$b_i^q = \mathbf{E}\left((\tau_q \psi_i)^* g \right), q = 1,...,Q, i = 0,...,M-1.$$

The normal equations (2.4) can be solved by an iterative method such as conjugate gradient (CG) method. One difficulty in solving equation (2.4) is that the RHS, b, is unknown because it is the expected value of a random variable involving $g(x^1,...,x^Q)$ which is the inverse function of the HPA, and hence is unknown. However, the function $g(x^1,...,x^Q)$ can be observed through the input and output samples of the HPA, and therefore, the expected value can be approximated by using sample averages. This approximation process will be described in details as follows.

At iteration k of the outer loop, samples are taken from the input and output of the HPA. These samples are used to form the approximations of the matrix A and the RHS b of (2.4) by replacing the expected value $\mathbf{E}(\cdot)$ with the sample averages. Specifically, let the output and input samples of the HPA be $\{y_0,...,y_{N-1}\}$ and $\{z_0,...,z_{N-1}\}$, respectively. Referring to Figure 1, z_n are input samples to the HPA, and therefore, the output of the inverse g, and y_n are the output samples of the HPA, and therefore, the input to the inverse. Therefore, $z_n = g(y_n, y_{n-1}..., y_{n-Q+1}), \quad n = 0,...,N-1$. When the expected value $\mathbf{E}(\cdot)$ is approximated by the sample average, equation (2.4) becomes

$$A^k u = b^k. \quad (2.5)$$

The superscript k in (2.5) indicates that the matrix and RHS are computed by using the sample sets taken at the iteration k of the outer loop. The components of A^k, b^k are given by

$$a^k_{p(Q-1)+i,\,q(Q-1)+j} = \frac{1}{N} \sum_{n=0}^{N-1} \omega^*_{npi} \omega_{nqj}, \quad (2.6)$$

$$\omega_{npi} = \tau_p(y_{n-p+1}) \psi_i(|y_{n-p+1}|)$$

$$(b_i^q)^k = \frac{1}{N} \sum_{n=0}^{N-1} \left(\tau_q(y_{n-q+1}) \psi_i(|y_{n-q+1}|) \right)^* z_n. \quad (2.7)$$

In (2.6) and (2.7), $\{y_n\}$ and $\{z_n\}$ are sample sets taken at the iteration k of the outer loop.

After the approximation of the expected values by sample averages, the inner loop is to solve equation (2.5) by using the CG method. Since now (2.5) is only an approximation of (2.4), there is no need to solve (2.5) to full accuracy, because the exact solution to (2.5) may not be the solution to (2.4), and hence does not necessarily give rise to the best predistorter in the sense of minimizing (2.3). Consequently, instead of performing QM iterations in the inner loop, only one CG iteration is performed in the inner loop. This method, in which equation (2.5) is solved by one iteration of CG algorithm, is called a stochastic conjugate gradient (SCG). The outer loop does not stop because the predistortion process always runs continuously to track changes in the HPA characteristics over time. Therefore, there are no stop criteria for the SCG iterations.

The SCG algorithm works as follows. Let u^{k-1} be an approximation of the solution to (2.4) after $k-1$ iterations in the outer loop (from now on, we will omit "out loop" with the understanding that all iterations refer to the outer loop because there is no "inner loop" in SCG). At iteration k, new samples are taken from HPA, and we want to compute the approximation u^k by applying one step of CG to equation (2.5). First, the residual of u^{k-1} is computed by

$$r^{k-1} = b^k - A^k u^{k-1}. \tag{2.8}$$

Note that neither the explicit formation of matrix A^k nor the explicit matrix-vector multiplication $A^k u^{k-1}$ is needed in the residual computation, as we will see later. Then, a new search direction is computed by

$$v^k = r^{k-1} + \beta_k v^{k-1} \tag{2.9}$$

where v^{k-1} is the search direction from iteration $k-1$. Initially, $v^1 = r^0$. The value of β_k is computed so that v^k is conjugate-orthogonal to v^{k-1}. Since there are no stopping criteria for the iterations, the search direction v^k needs to be reset to r^{k-1} periodically, in order to avoid the loss of orthogonality. Finally, the new approximation is computed as

$$u^k = u^{k-1} + \alpha_k v^k \tag{2.10}$$

where α_k is computed to minimize the residual in the search direction v^k. This algorithm is given below.

Algorithm SCG

At start:

Given $\varepsilon > 0$, and basis functions $\{\psi_0(|x|),...,\psi_{M-1}(|x|)\}$

Determine a strategy to reset the search direction at least once every QM iterations.

Let $u^q(|x|) = 0$, for $q = 1,...,Q$

loop:

Take input samples $\{z_0,...,z_{N-1}\}$, and output samples $\{y_0,...,y_{N-1}\}$, from HPA,

$$\gamma_i^q = \frac{1}{N}\sum_{n=0}^{N-1}\left((\tau_q(y_{n-q+1})\psi_i(|y_{n-q+1}|))^* \cdot (z_n - \sum_{p=0}^{Q-1}\tau_p(y_{n-p})u^p(|y_{n-p}|))\right),$$
$$q = 1,...,Q, i = 0,..,M-1 \tag{2.11}$$
$$r^q = [\gamma_0^q,...,\gamma_{M-1}^q]^T, q = 1,...,Q$$

if at start or reset

$$v^q(|x|) \leftarrow \sum_{i=0}^{M-1}\gamma_i^q\psi_i(|x|), q = 1,...,Q \tag{2.12}$$

else

$$\beta = \left(\sum_{q=1}^{Q}(r^q)^H r^q\right)/\omega,\ \omega \leftarrow \sum_{q=1}^{Q}(r^q)^H r^q$$
$$v^q(|x|) \leftarrow \sum_{i=0}^{M-1}\gamma_i^q\psi_i(|x|) + \beta v^q(|x|), \quad q = 1,..,Q \tag{2.13}$$

end if

if $\omega < \varepsilon$

reset at next iteration and go to next iteration

else

$$\lambda = \sum_{n=0}^{N-1}\left|\sum_{q=0}^{Q-1}\tau_q(y_{n-q})v^q(|y_{n-q}|)\right|^2$$
$$\alpha = \frac{\omega}{\lambda} \tag{2.14}$$

$$u^q(|x|) \leftarrow u^q(|x|) + \alpha v^q(|x|), q = 1,...,Q \tag{2.15}$$

end if

end loop

At each iteration of Algorithm SCG, the computed functions u^q can be used to form the predistorter as

$$z_n = P(x_n,...,x_{n-Q+1}) = \sum_{q=1}^{Q}\tau_q(x_{n-q+1})u^q(|x_{n-q+1}|). \tag{2.16}$$

One advantage of Algorithm SCG is that there is no need to explicitly compute the covariance matrix A^k of (2.6). There is also no explicit computation of the matrix-by-vector multiplication because the residual is computed by the RHS of (2.12). Another advantage is that the solutions computed from Algorithm SCG is less sensitive to the round-off error in the computation and the noise in the samples.

In implementations, the functions involved in Algorithm SCG, the basis functions ψ_i, search directions v^q, and the approximation of the solution u^q, can be represented by lookup tables (LUT). At each iteration, the computed LUT u^q can be used directly to form the predistorter (2.16). LUTs are easy to implement in hardware platform such as an FPGA.

We note that Algorithm SCG becomes the conventional CG algorithm if the steps (2.11) through (2.15) are carried out with the same sample sets $\{z_0,...,z_{N-1}\}$ and $\{y_0,...,y_{N-1}\}$ in each iteration, i.e., if the samples are taken outside of the "loop" in Algorithm SCG.

III. SIMULATIONS AND LAB EXPERIMENTS

3.1 Simulations of SCG

In this subsection, we present simulation results in applying Algorithm SCG to memory polynomial predistortion of HPA as shown in Figure 1

In the simulations, we use the memory polynomial PA model as given in Example 2 of [2]. The memory polynomial of degree 5 with 3 memory delay taps is used as the predistorter. That is, the parameters of Algorithm SCG are given by $M = 5, Q = 3$. As is suggested by [2], this choice of parameters results in good performance for the predistortion with the given PA model.

An OFDM signal with the 16QAM modulation is used in our simulations. At the beginning of each simulation, 25,600 samples are captured for each of y_n, z_n. These samples are used to estimate the probability density function ρ using the histogram method. The orthogonal polynomial basis $\{\psi_0(|x|),...,\psi_{M-1}(|x|)\}$ is then formed using the three term recursion. These functions are represented by LUTs with 4096 entries each. At each iteration of the SCG algorithm, a total of $N = 1280$ samples are taken for each of y_n, z_n.

Three simulations using Algorithm SCG are performed, which are named Sim1, Sim2 and Sim3. In Sim1, the traditional CG algorithm is performed. That is, the inner loop is performed with the maximum number of iterations per set of samples captured, i.e., $MQ = 15$ iterations are performed for each data captured in the outer loop. The solution at the last iteration of the inner loop corresponds to the solution by a direct method applied to the normal equations (2.5), in the absence of round-off error. In Sim2, only one iteration is performed for each set of samples captured. Sim2 represents the true SCG. In both Sim1 and Sim2, the weight function used for the orthogonalization of the basis functions is the estimated probability density function. Sim3 is similar to Sim2, in which Algorithm SCG is performed but the weight function is the uniform distribution (that is, no weight function is used in the orthogonalization process). In all simulations, the search direction v is reset in every $MQ = 15$ iterations. In Sim1, the reset is performed every time a new set of samples is captured. In all simulations, at each iteration of algorithm SCG, the newly updated u is immediately used in the outer loop as the predistorter.

In each simulation, a total of 210 iterations are performed. In Sim1, a total of 14 data captures are performed (there are 15 iterations for each data capture), and 210 data captures are performed in other simulations. At the end of each simulation, the linearization of the HPA is achieved, because the HPA output signal y_n is almost identical to the original signal x_n. A plot of the spectra of different signals for Sim2 is shown in Figure 2. Spectra from Sim1 and Sim3 are indistinguishable from Figure 2.

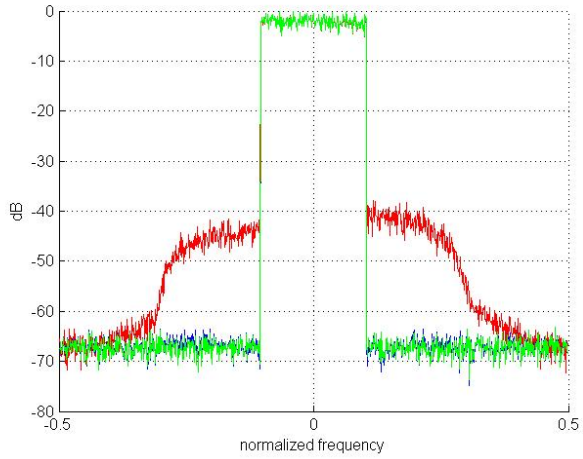

Figure 2. Power spectra

In Figure 2, the red curve, the curve with the highest spectrum shoulders, shows the spectrum of the signal after HPA when no predistortion is used. The green and blue curves, which are almost identical and indistinguishable, show the spectra of the original signal x_n and the signal y_n after HPA when the predistortion is used, respectively.

The spectrum plot in Figure 2 shows that at the end of the iterations after the convergence, the solutions from all three simulations have the desired accuracy because the nonlinear efforts of the HPA have been completely removed with the computed predistorter. A comparison of Figure 2 with the plot in [2, Figure 4] also demonstrates that the solutions from iterative methods of Sim1, Sim2 and Sim3 all have the same accuracy as a solution obtained by a direct method from [2]. However, the SGC algorithm is much more efficient in terms of computational complexity than a direct method.

To show the performance of Algorithm SCG algorithm, we examine the residual computed at the beginning of each SCG iteration. The normalized residuals as functions of SCG iteration number are shown in Figure 3 for the three simulations.

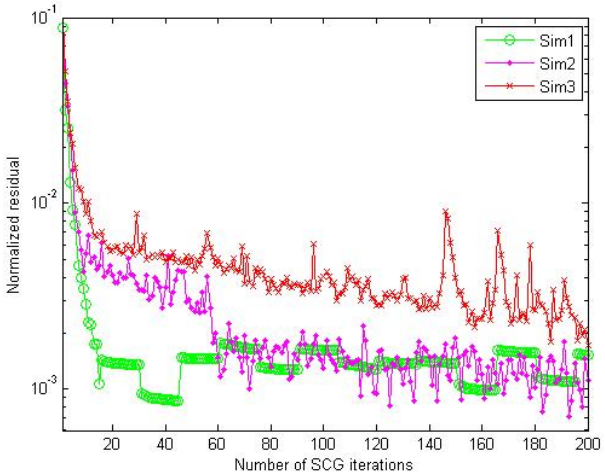

Figure 3. The convergence of normalized residuals

We can make the following observations. First, as expected, in Sim1, a fairly accurate solution is obtained in 15 iterations. After that, the residual does not change

significantly. The variation in the residuals after 15 iterations is mainly due to the fact that they are computed with different sets of samples. After convergence, the residual remains almost constant during the 15 iterations in which the same set of samples is used. This demonstrates that after some initial time, there is no need to perform more iterations in the CG algorithm using the same set of captured samples, which justifies the SCG algorithm.

In Sim2, the first 5 or 6 iterations are almost identical to those in Sim1. After that, the convergence slows down. Again this is expected because the SCG algorithm loses orthogonality when different samples are taken at different iterations. However, the normalized residuals are reduced to the similar level as in Sim1 after about 60 SCG iterations (which is equivalent to 4 data captures in Sim1).

In Sim3, we see that the convergence of the SCG is significantly slower when the uniform distribution is used as the weight function in forming the orthogonal basis. This is because the condition number of the covariance matrix is larger when the weight function is not equal to the probability density function of the samples.

3.2 Lab experiments

To verify the effectiveness of the SCG method in predistortion in the real world, an offline digital predistortion system with memory compensation is implemented in a high speed Field Programmable Gate Array (FPGA). The computation of polynomial coefficients, i.e., the SCG algorithm, is performed offline using Matlab on a PC. The predistorter with separate LUTs structure is implemented in signal path in FPGA to predistort the original input signal, shown in Figure 4. The HPA used in the experiments is a B-Class power amplifier with average output power of 50W.

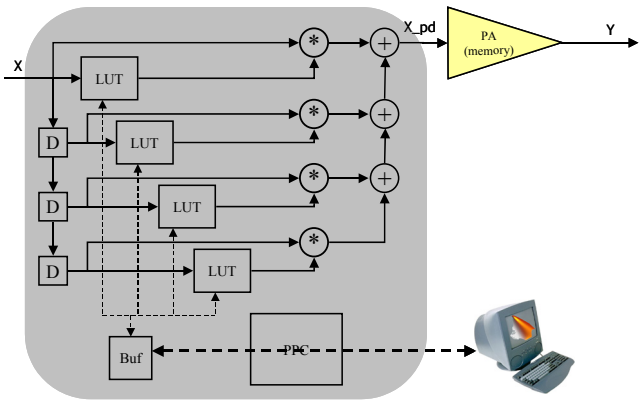

Figure 4. Structure of offline predistortion System

The SCG method performs very well in conjunction with the polynomial predistorter in the suppression of adjacent channel power (ACP) and reduction of error vector magnitude (EVM). In Figure 5 and Figure 6, the spectra of PA output for one carrier and four carrier UMTS signals are illustrated, respectively. In both Figure 5 and Figure 6, the top curve, the black color curve with higher shoulder values, is the spectrum of the HPA output signal without using a predistortion. The bottom curve, the blue color curve with lower shoulder values, is the spectrum of the HPA output signal with the predistortion. As shown, in both cases, the ACPs are suppressed over 20 dB through the polynomial predistorter in conjunction with the SCG method.

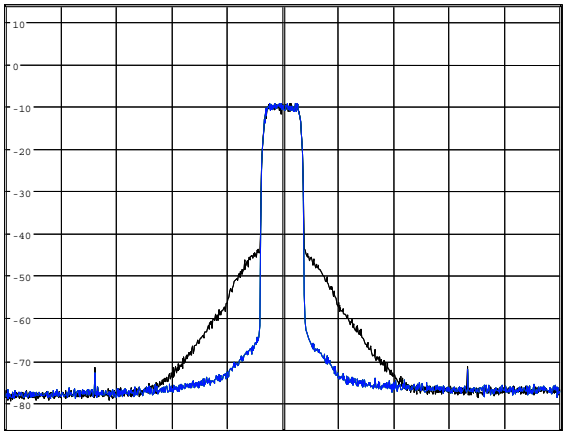

Figure 5. Spectrum of PA output for one carrier UMTS signal with and without DPD

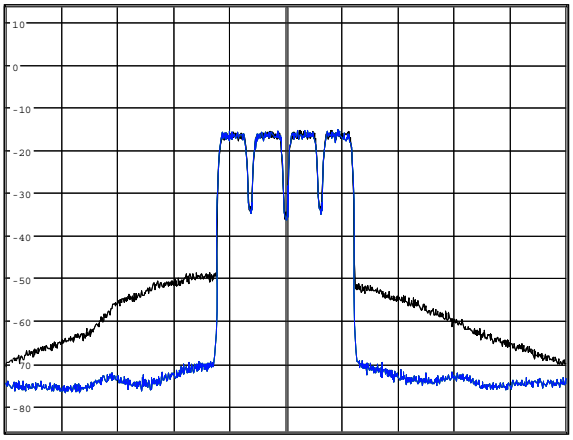

Figure 6. Spectrum of PA output for four carriers UMTS signal with and without DPD

REFERENCES

[1] D. R. Morgan, Z. Ma, J. Kim, M. G. Zierdt, and J. Pastalan, "A generalized memory polynomial model for digital predistortion of RF power amplifiers", *IEEE transaction on signal processing*, vol 54, no 10, 2006, pp. 3852-3860.

[2] L. Ding, G. T. Zhou, D. R. Morgan, Z. Ma, J. S. Kenney, J. Kim, and C. R. Giardina, "A robust digital baseband predistorter constructed using memory polynomials", *IEEE transaction on communications*, vol 52, no 1, 2004, pp. 159-165.

[3] R. Raich, H. Qian, and G. T. Zhou, "Orthogonal Polynomials for Power Amplifier Modeling and Predistorter Design", *IEEE Trans. Veh. Technol.*, 2004, vol. 53, no 5, pp. 1468-1479.

[4] R. Raich, H. Qian, and G. T. Zhou, "Orthogonal polynomials for complex Gaussian processes", *IEEE transactions on signal processing*, vol 52, no 10, 2004, pp. 2788-2797.