Performance Analysis of Crosscorrelation Predistorters

André B.J. Kokkeler
University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands, Phone: +31 53 4894291

***Abstract*— Amplification of signals with fluctuating envelopes leads to distortion because of non-linear behavior of the Power Amplifier (PA). Digital Predistortion can counteract these non-linear effects. A crosscorrelation predistorter is a digital predistorter, based on the calculation of crosscorrelation functions. The crosscorrelation functions are transformed to the frequency domain and the spectra are used to control the predistorter. In this paper, the crosscorrelation predistorter is described and the performance is given together with the performance of an existing predistorter, the LS predistorter. The crosscorrelation predistorter outperforms the LS predistorter against lower complexity.**

I. Introduction

One of the characteristics of modern, spectrally efficient transmission formats (CDMA, OFDM) is their relatively high peak-to-average ratio. Non-linear Power Amplifiers (PAs), fed by signals with high peak-to-average ratio, introduce severe distortion. Distortion within the band of transmission degrades signal detection. Distortion outside the band of transmission (spectral regrowth) interferes with the signals transmitted by other users at adjacent channels and is therefore called Adjacent Channel Interference (ACI). Spectral masks have been defined by the regulatory bodies for different communication standards to specify allowed ACI levels.

Power Amplifiers are costly and their operation is power-inefficient because of the measures to limit distortion. To counteract the non-linearity of a PA, several techniques have been investigated: feedforward, feedback and predistortion. In this document we will concentrate on predistortion. The general principle is to apply the inverse input-to-output relation of the PA to the signal at the input of the PA. Predistortion followed by the PA (and its inherent distortion) should result in linear amplification. Predistortion is applied in the analog domain (see [1]) and in the digital domain (see [2]). In digital predistortion, the baseband signal is predistorted before it is converted to the analog domain, frequency translated to RF and amplified (see figure 1).

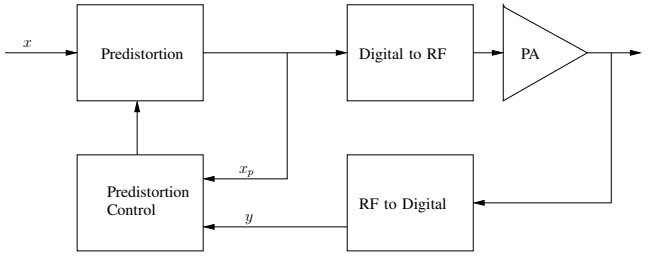

Fig. 1. Digital Predistortion

Because the input-output relation of the PA changes in time due to temperature changes and aging of components in the analog part, a control mechanism constantly adapts the predistortion. A small fraction of the PA output is fed back and converted from RF to baseband. An adaptation algorithm compares this signal with the output of the predistorter. Two different approaches exist: Direct- and Indirect Learning. When using Direct Learning, the input-to-output relation of the PA is determined. This relation is inverted and used for predistortion. For an example, see [3]. The Indirect Learning scheme was introduced in [4]. The input-to-output relation of the PA is determined indirectly, the inverse relation directly. The PA (in case of Direct Learning) or the predistorter (in case of Indirect Learning) can be modeled using memory polynomials (see [5]). In this document we present a framework which is used to describe predistorters in general. An existing predistorter is described and will be referred to as 'LS predistorter'. A new predistorter called the 'crosscorrelation predistorter' is presented and simulation results for both the Direct- and Indirect Learning schemes are given.

II. Digital Predistortion Schemes

In this section, a general framework for different configurations of a digital predistorter is presented. This framework is used to describe the LS- and crosscorrelation predistorters.

A. General Framework

Figure 1 shows that the predistorted signal x_p and the PA output (converted to a digital baseband signal) y are used to determine the non linear characteristics of the PA. The output can be written as a function of the input (Direct Learning) or vice versa (Indirect Learning). This is generally described by: $x_2 = f(x_1)$, where $x_1 = x_p$ and $x_2 = y$, for Direct Learning and $x_1 = y$ and $x_2 = x_p$, for Indirect Learning. The function f can be described by means of polynomial basis functions $\gamma_k(x)$:

$$x_2(t) = \sum_{k=1}^{K} \sum_{\tau=0}^{\tau_{max}-1} a_{k\tau} \gamma_k(x_1(t-\tau)) \tag{1}$$

where $\gamma_k(x) = \mid x \mid^{k-1} x$.
Using x_1, x_2 and γ, a single description for both the Direct- and Indirect Learning schemes can be used.

B. The LS predistorter

In this section, a description of the LS predistorter using the framework presented above, is given. The LS predistorter is based on the predistorter presented in [5]. We define:

$$\gamma_{k\tau} = \gamma_k(x_1(t-\tau)) \tag{2}$$

for $t \in \mathbb{Z}$. Via linear combination of the signals $\gamma_{k\tau}$, an estimate of x_2 can be constructed using the least squares

criterion. If we define

$$\begin{aligned} \mathbf{x_2} &= [x_2(1), ..., x_2(T)]^T && (3)\\ \mathbf{\Gamma} &= [\mathbf{y}_{10}, ..., \mathbf{y}_{K0}, ..., \mathbf{y}_{1\tau_{max}-1}, ..., \mathbf{y}_{K\tau_{max}-1}] && (4)\\ \mathbf{y}_{k\tau} &= [\gamma_{k\tau}(x_1(1)), ..., \gamma_{k\tau}(x_1(T))]^T && (5)\\ & && (6) \end{aligned}$$

we have to find the solution to the equation

$$\mathbf{x_2} = \mathbf{\Gamma} \cdot \mathbf{a} \tag{7}$$

The least squares solution of this equation is:

$$\hat{\mathbf{a}} = (\mathbf{\Gamma^H \Gamma})^{-1} \mathbf{\Gamma^H x_2} \tag{8}$$

where ^{H} indicates the complex conjugate transpose. The complexity of the LS predistorter is at least $O(T)$; the number of full-precision multiplications scales linearly with the number of samples used to update the predistorter polynomial coefficients $\hat{\mathbf{a}}$. If the feedback signal y is contaminated with noise, for example quantization noise, the number of samples needs to be increased. Thus the digital complexity depends on the quality of the feedback signal.

C. The crosscorrelation predistorter

To decouple the digital complexity from the quality of the feedback signal we developed the crosscorrelation predistorter. The algorithm consists of two parts. In the first part, crosscorrelation functions are generated without full-precision multiplications being involved. The crosscorrelation functions consist of a fixed number of elements (lags), independent of the number of samples. In the second part we estimate the polynomial coefficients using the least-squares solution based on the fourier transforms of the crosscorrelation functions. In this part, full precision multiplications are involved but the number of elements of the crosscorrelation functions is fixed.

So, for the crosscorrelation predistorter, the signals $\gamma_{k\tau}$ are not combined directly. First $\gamma_{k\tau}$ and x_2 are crosscorrelated with a reference signal. This means that $\mathbf{x_2}$ and the columns of $\mathbf{\Gamma}$ (see equation 7) are subjected to the same linear operation. Thus, after crosscorrelation, an equation similar to equation 7 is valid which can be solved using the least squares criterion.

In theory, the reference signal can be any signal with significant power within the primary and adjacent channels. We chose to crosscorrelate with a single-bit representation x_Q of x_p because this signal has significant power at the relevant frequencies and it is available for the predistorter. Single-bit quantization is defined as:

$$x_Q(x_p) = \text{sign}(\text{Re}(x_p)) + j\text{sign}(\text{Im}(x_p)) \tag{9}$$

where sign() is defined as:

$$\text{sign}(x) = \begin{cases} -1 & x < 0 \\ 1 & x \geq 0 \end{cases} \tag{10}$$

Re(x) indicates the real part of a complex value x and Im(x) the imaginary part. Using a single-bit quantized value as one of the operands of a crosscorrelation drastically reduces the complexity. The multiplications involved become very simple to implement: the second operand of the multiplication is either left unchanged or its sign is reversed if the first operand equals 1 or -1 respectively. The number of points (or lags) of the crosscorrelation is even and equals N. Using vector notations, the single-bit quantized signal equals:

$$\mathbf{x_Q} = \left[x_Q(x_p(\frac{1}{2}N + 1)), ..., x_Q(x_p(T - \frac{1}{2}N + 1)) \right]^T \tag{11}$$

We define the matrices $\mathbf{\Gamma_{k\tau}}$ for $k = 1, ..., K$ and $\tau = 0, ..., \tau_{max} - 1$, and $\mathbf{X_2}$ as:

$$\mathbf{\Gamma_{k\tau}} = \left[[\gamma_{k\tau}(1), ..., \gamma_{k\tau}(T - N + 1)]^T, ..., [\gamma_{k\tau}(N), ..., \gamma_{k\tau}(T)]^T \right]^T \tag{12}$$

$$\mathbf{X_2} = \left[[x_2(1), ..., x_2(T - N + 1)]^T, ..., [x_2(N), ..., x_2(T)]^T \right]^T \tag{13}$$

T is the number of consecutive samples, available to the predistorter. The crosscorrelations are defined as:

$$\begin{aligned} \mathbf{r_{k\tau}} &= \mathbf{\Gamma_{k\tau}} \mathbf{x_Q^*} && (14)\\ \mathbf{r} &= \mathbf{X_2} \mathbf{x_Q^*} && (15) \end{aligned}$$

The crosscorrelation vectors $\mathbf{r_{k\tau}}$ and $\mathbf{r}$ are tapered with a Hanning taper:

$$h(j) = \frac{1}{2}(1 - \cos(2\pi(j-1)/(N-1))) \tag{16}$$

$$\mathbf{H} = \text{d}iag(h(1), ..., h(N)) \tag{17}$$

where 'diag()' indicates a diagonal $N \times N$ matrix. A Discrete Fourier Transform is applied to the tapered crosscorrelation vectors:

$$\begin{aligned} \mathbf{f_{k\tau}} &= \mathbf{W H r_{k\tau}} && (18)\\ \mathbf{f} &= \mathbf{W H r} && (19) \end{aligned}$$

where $\mathbf{W}$ equals the DFT kernel. The elements w_{pq} of the kernel are defined as:

$$w_{pq} = \exp^{-i2\pi \frac{p-1}{N}(q-1)} \tag{20}$$

If the vectors (spectra) $\mathbf{f_{k\tau}}$ are concatenated into a matrix $\mathbf{F} = [\mathbf{f_{10}}, ..., \mathbf{f_{K0}}, ..., \mathbf{f_{1\tau_{max}-1}}, ..., \mathbf{f_{K\tau_{max}-1}}]$ and the vector $\mathbf{a}$ is defined as $\mathbf{a} = [a_{10}, ..., a_{K0}, ..., a_{1\tau_{max}-1}, ..., a_{K\tau_{max}-1}]^T$, the memory polynomial predistorter is described in the frequency domain as:

$$\mathbf{f} = \mathbf{F a} \tag{21}$$

The least squares solution minimizes the *absolute* error over the frequency domain. Because the signal has a relatively low power spectral density in the adjacent channels, the *relative* errors in the adjacent channels can be (too) large. For that reason, we minimize the relative error. This is realized by normalizing the spectra with $\mathbf{f}$:

$$g_{k\tau}(n) = \frac{f_{k\tau}(n)}{f(n)} \tag{22}$$

$$\mathbf{g_{k\tau}} = [g_{k\tau}(1), ..., g_{k\tau}(N)]^T \tag{23}$$

$$\mathbf{G} = [\mathbf{g_{10}}, ..., \mathbf{g_{K0}}, ..., \mathbf{g_{1\tau_{max}-1}}, ..., \mathbf{g_{K\tau_{max}-1}}] \tag{24}$$

The N element vector $\mathbf{g}$ is defined as:

$$\mathbf{g}(n) = 1, \quad n = 1, ..., N \tag{25}$$

The normalized version of equation (21) becomes:

$$\mathbf{g} = \mathbf{Ga} \tag{26}$$

The least-squares solution $\hat{\mathbf{a}}$ equals:

$$\hat{\mathbf{a}} = (\mathbf{G^H G})^{-1}\mathbf{G^H g} \tag{27}$$

To reduce the effects of noise on the correlation functions $\mathbf{g_{k\tau}}$, we select only those vector elements which represent the power in the main channel and the first adjacent channel leading to vectors $\mathbf{g}$ and $\mathbf{g_{k\tau}}$ with limited length.

In the crosscorrelation predistorter, the size of the vectors $\mathbf{f_p}$ and $\mathbf{f_{k\tau}}$ equals N. If an FFT is used to transform the vectors from the time domain to the frequency domain, the complexity is $O(N \log_2 N)$. The complexity of the LS predistorter is $O(T)$. In general N is much smaller than T so the crosscorrelation predistorter has lower complexity than the LS predistorter. The reduction of the complexity is due to the crosscorrelation which is easy to implement. Due to the single-bit quantization of x_p, no full-precision complex multiplications are required.

D. The inverse of the PA memory polynomial

In the Indirect Learning scheme, the estimated polynomial coefficients ($\hat{\mathbf{a}}$) can be directly used in the predistorter; the incoming data x is predistorted according to the estimated memory polynomial. In the Direct Learning scheme however, the inverse of the estimated memory polynomial has to be determined. A procedure to approximate the inverse of a memory polynomial in case of Direct Learning was presented in [3]. The predistorted signal x_p is generated according to the following algorithm:

$$x_p(t) = \frac{1}{\beta_0(\mid x_p(t) \mid)}\Big(x(t) - \beta_1(\mid x_p(t-1) \mid)x_p(t-1)\Big) \tag{28}$$

where

$$\beta_\tau(x) = \sum_{k=1}^{K} \tilde{a}_{k\tau} x^{k-1} \tag{29}$$

$$\tilde{a}_{k\tau} = \hat{a}_{k\tau} \tag{30}$$

The problem with this algorithm is that $\mid x_p(t) \mid$ needs to be known in order to calculate $x_p(t)$. In [3], it is suggested to use $x_p^{initial} = x(t)$ as an initial value. The algorithm then calculates a new estimate of $x_p(t)$, uses this as the next initial value, calculates a new estimate etc. This process is repeated until a stable estimate of $x_p(t)$ is obtained. From simulations we found that the algorithm gives satisfactory results if the PA satisfies 2 conditions. First, the memory effects should not be severe which means that the β_1 values should be significantly smaller than the β_0 values. If for example, the PA only delays the signal with one sample, $a_{k0} = 0, \ \forall k$ which implies that $\beta_0 = 0$, leading to a divide by zero error. Second, the distortion should be weak; the first order parameter $\hat{a}_{10}$ should be close to 1. To cope with amplifiers with severe distortion and memory effects, we modified the algorithm in two ways. First, in case of severe memory effects where $\hat{a}_{11} > \hat{a}_{10}$, we assume that $\hat{a}_{k2} = \hat{a}_{k0}$ and use an additional delay in the predistorter:

$$\begin{aligned} \tilde{a}_{k\tau} = &\quad \hat{a}_{k\tau} \ \text{ if } \ \hat{a}_{11} \leq \ \hat{a}_{10} \\ = &\quad \hat{a}_{k\tau+1} \ \text{ if } \ \hat{a}_{11} > \ \hat{a}_{10} \end{aligned} \tag{31}$$

Second, instead of using $x_p^{initial}(t) = x(t)$ we used as initial value:

$$x_p^{initial}(t) = \frac{1}{\tilde{a}_{10}}\Big(x(t) - \beta_1(\mid x_p(t-1) \mid)x_p(t-1)\Big) \tag{32}$$

The part between the large brackets takes into consideration the memory effects from the beginning. The effects of the first order component differing from 1 are canceled by division with $\tilde{a}_{10}$.

III. Simulation results

In this section we briefly describe the simulator that was developed to analyze both the LS- and crosscorrelation predistorters. After that, the results of simulations are presented.

A simulator, based on the structure presented in the previous sections has been implemented. As a signal source we used two data generators: one for the real part of the signal and one for the imaginary part. A data generator is based on the signal source used in [6] and generates sine waves with different consecutive frequencies, equal amplitudes and randomly selected phase. The oversampling factor is 10.

The PA model we used is a Wiener-Hammerstein model presented in [5]. This model consists of an IIR filter $H(z)$ followed by a memoryless polynomial which on its turn is followed by a second IIR filter $G(z)$. The filter specifications are:

$$H(z) = \frac{1 + 0.5z^{-2}}{1 - 0.2z^{-1}}, \quad G(z) = \frac{1 - 0.1z^{-2}}{1 - 0.4z^{-1}} \tag{33}$$

The memoryless polynomial equals:

$$w(n) = \sum_{k=1}^{K} b_k \ v(n) \mid v(n) \mid^{k-1} \tag{34}$$

where v indicates the output of filter H and w the input of filter G. The coefficients are: $b_1 = 1.0108 + 0.0858j, b_3 = 0.0879 - 0.1583j, b_5 = -1.0992 - 0.8891j$. In our simulations, we used blocks of 8K samples ($T = 8192$) and 64-point crosscorrelation functions ($N = 64$). For the predistorter, we used only odd polynomials up to the fifth order ($k = 1, 3, 5$) and the memory depth is 2 ($\tau = 0, 1$).

The performance of the different schemes is characterized by the ACI level: the power in the channel adjacent to the primary channel. The bandwidth of the adjacent channel equals the bandwidth of the primary channel. The power in the adjacent channel is related to the power in the primary channel and for that reason measured in dB. The performance depends on the input signal. New settings of the predistorter are based on data that already has been transmitted. New data might contain signal-excursions which have not been accounted for. This can lead to limited suppression of ACI levels and in severe cases to settings from which the predistorter cannot recover. For that reason, we simulated 40 cycles for all 4 predistortion schemes where every cycle consisted of the following stages: generation of an 8K-samples-dataset for cycle i, predistortion of this dataset using the predistorter settings found in cycle $i-1$, distortion, estimation of the memory polynomial and

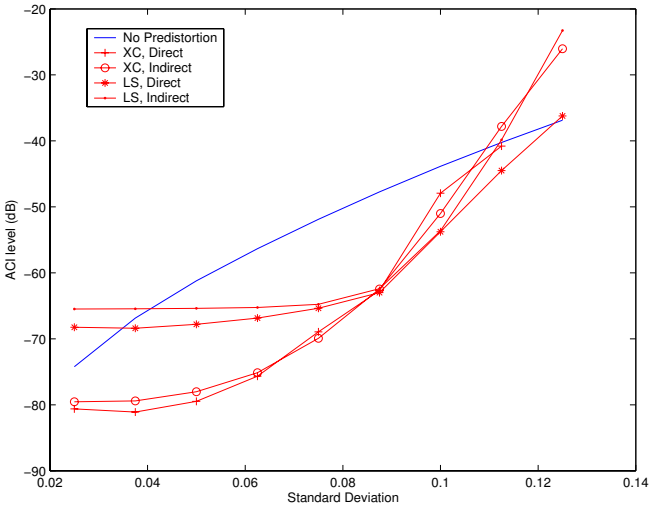

Fig. 2. Performance of the predistortion schemes

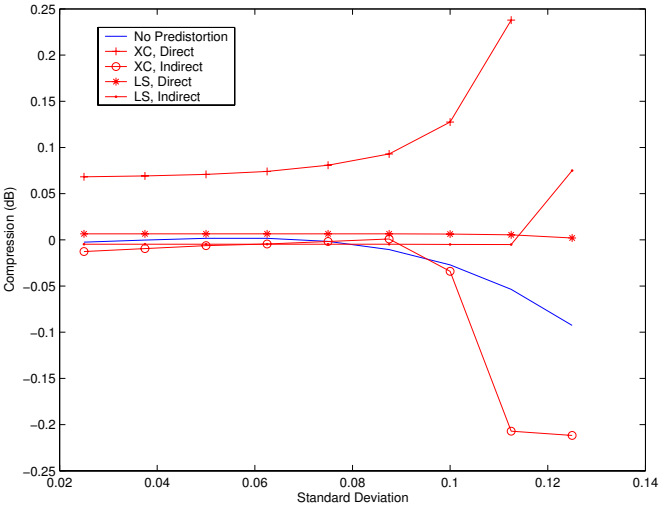

Fig. 3. Compression of the predistortion schemes

determination of the new predistorter settings by averaging the new estimate of the memory polynomial (weight = $\frac{1}{3}$) with the averaged previous polynomials (weight = $\frac{2}{3}$). For all predistortion schemes the same data is used. In figure 2, the average ACI levels in the adjacent channel are presented. The average ACI levels represent the ACI levels, averaged over the last 30 cycles in a simulation of 40 cycles. The ACI levels with- and without predistortion are given as a function of the standard deviation of the predistorter input signal.

The crosscorrelation predistorter (results indicated with XC) performs better than the LS predistorter especially for small standard deviations at the input. For standard deviations larger than 0.09, the performance is comparable. For large standard deviations, the Indirect Learning scheme leads to higher in stead of lower ACI levels. For a standard deviation of 0.0875, the performance of all schemes is approximately equal. At this standard deviation, the ACI levels are reduced significantly from -48 dB to -62 dB. Because of digital predistortion more power can be transmitted while still satisfying maximum requirements for the ACI levels. If for example the maximum allowed ACI level equals -62 dB, the maximum standard deviation at the input in case no predistortion is used equals 0.0482. When using predistortion, the maximum standard deviation at the input equals 0.0875 which is an increase of 2.6 dB. So, the predistorted PA can have 2.6 dB more output power than the PA without predistortion. This leads to a higher efficiency of the PA.

Besides the ACI levels, the compression of the Predistortion-PA combination is an important measure of the performance. In the ideal case, the compression after correction for linear gain equals 0 dB. The compression with and without predistortion is presented in figure 3.

We see that the LS predistorter reduces the compression and leads to a gain (after correction for linear gain) close to 0 dB over the complete input range. The compression for the crosscorrelation predistorter differs for the Direct- and Indirect Learning schemes. The Indirect Learning scheme leads to more compression compared to the case where no predistortion is applied. The Direct Learning scheme leads to an additional gain.

IV. Conclusions

In this paper, the crosscorrelation predistorter is presented as an alternative to an existing digital predistorter, the LS predistorter. The advantage of the crosscorrelation predistorter is that the digital complexity is independent of the number of samples used to estimate the non-linearity of the predistorter. The complexity of the LS predistorter scales linearly with the number of samples used to estimate the non linear behavior of the PA. The basic idea behind the crosscorrelation predistorter is to split the algorithm into two parts. First, a crosscorrelation part which has low digital complexity and which transforms the input signals to crosscorrelation functions of fixed size. Second, a part which estimates the non-linearity of the PA, based on the the crosscorrelation functions using the least squares criterion. Because correlation functions of fixed size are used, the complexity is fixed. For both predistorters, a description and simulation results are given. Both predistorters can be based on either the Direct Learning or Indirect Learning scheme. The crosscorrelation predistorter outperforms the LS predistorter for small standard deviations at the input. For larger standard deviations the performances are equal. The compression of the Predistorter-PA combination is reduced for both predistorters except for the crosscorrelation predistorter using the Indirect Learning scheme and large standard deviation.

References

[1] C.G. Rey, "Predistorter linearizes CDMA power amplifiers," *Microwaves RF*, Oct. 1998, pp. 114-123.

[2] F. Zavosh, M. Thomas, C. Thron, T. Hall, D. Artusi, D. Anderson, D. Ngo, and D. Runton, "Digital predistortion techniques for RF power amplifiers with CDMA applications," *Microwave Journal*, Oct. 1999.

[3] J. Kim and K. Konstantinou, "Digital predistortion of wideband signals based on power amplifier model with memory," *Electronics Letters*, vol. 37, no. 23, Nov. 2001, pp. 1417-1418.

[4] C. Eun and E.J. Powers, "A predistorter design for a memory-less nonlinearity preceded by a dynamic linear system", *Proc. GLOBECOM 1995*, pp. 152-156.

[5] L. Ding, G.T.Zhou, Z. Ma, D.R. Morgan, J.S. Kenney, J. Kim, and C.R. Giardina, "A robust digital baseband predistorter constructed using memory polynomials," *IEEE Trans. on Communications*, March 2002 (submitted).

[6] J. K. Cavers, "Amplifier linearization using a digital predistorter with fast adaptation and low memory requirements", *IEEE Trans. Veh. Technol.*, vol. 39, no. 4, Nov. 1990, pp. 374-383.