Clayton Daigle

EE391

Autumn 2005

Supervised by Dr. Boris Murmann

(1)

Determination of nonlinear systems using statistical correlation across orthogonal error models.

A primary tenant of modeling is the idea that it is a correlation operation. Curve fitting schemes in fact compute coefficients that achieve the best possible correlation between the output of a function or model and the actual output of the system being modeled. Model effectiveness is judged using a variety of statistical benchmarks that measure this correlation. The most common statistical benchmark is mean squared error, though many others exist for various reasons. In any case, the degree of correlation describes how good the model is.

The second idea we wish to develop is that of orthogonal models. Two models are orthogonal if neither can be comprised of the other. For example, gain and offset of a linear system are orthogonal. The gain of a system can be computed independently of the offset and vice versa. However, care must be taken in order to define these models in a way that makes them orthogonal. For example, if you define offset to be the value of the system output when the input is zero, then gain error must be defined to strictly pivot around this point, even though there may be a more natural pivot point elsewhere (for example using a pivot point of 0 may not make sense in an industrial signaling system of 4 – 20mA). Further complications arise when nonlinearities are present. How do you define the gain and offset of a system with third order nonlinearity? In such cases, it is beneficial to retain a common ideal to which all errors can be referenced. This ensures orthogonality among all system equations and keeps various models from stepping on each others' toes by double-correcting for errors. One way to do this is to reference all errors to a best-fit line. If gain, offset and nonlinearity errors are all referenced to an agreed-upon best fit line, then these models are each orthogonal and therefore will not affect each other. This makes it possible to use combinations of models that otherwise would not be compatible. For example, one could use some type of binary weighted model to correct for ADC nonlinearities in addition to first order gain and offset correction and be confident that errors are not double-counted as long as both models work with respect to a common best-fit line.

Extending this idea to systems with memory is more difficult, but actually just adds the time dimension to the ideal behavior. In other words, the ideal behavior will be defined by a function of time as well as voltage, resulting in perhaps a Bessel filter or other such thing. The mapping becomes difficult to picture and perhaps impossible to plot, but fairly straightforward to define.

Once a model is chosen, the model coefficients can be calculated by comparing the system output with the model output. It is essential that the output be something that can be decomposed into orthogonal components and that these components sum to equal exactly the total signal. Also, coefficient convergence time can probably decreased by determining them in a certain order and removing each type of error in a cascaded fashion. For example, removing higher order distortion first might make it easier to determine the gain error. Then removing the gain error might make it easier to determine the offset error and so on.

(2)

Generalized Sampling Requirements for Characterization of a Nonlinear System

Nonlinear systems cause spectral spreading, generally resulting in an output bandwidth that is larger than the input signal bandwidth. However, when the input signal is known, it has been shown that system nonlinearities can be identified successfully without sampling at the Nyquist rate of the output signal (ie. Without sampling at a rate greater than twice the value of the highest frequency component in the output signal). Examples of nonlinear systems with known inputs are any digital to analog conversion process, where the digital signal is often ideal, only to be distorted by D/A conversion and analog processing such as amplification. Existing literature shows that system nonlinearities can be identified by sampling the output signal at the Nyquist rate of the *input* signal, a technique referred to as Input Nyquist Sampling. This was developed from Zhu's *Generalized Sampling Theorem*, stated here.

In 1992, Zhu published a "Generalized Sampling Theorem" [5]. Let $y(t)$ be a function whose spectrum may or may not be bandlimited. Suppose that as in Fig. 1, a one-to-one continuous mapping $g(.)$ exists such that $g(y(t))$ is bandlimited so that its Fourier transform $G_y(f) = 0$ for $|f| \geq F_0$, where $F_0 = 1/2T_s$. The fact that $g(y(t))$ has a lower bandwidth than $y(t)$ implies that $y(t)$ was originally obtained by nonlinear processing on a signal $x(t)$ by $f(.) = g^{-1}(.)$, where $g^{-1}(.)$ is the inverse of $g(.)$. Then, $y(t)$ can be sampled at the Nyquist rate of $g(y(t))$, that is, at the points $t_k = kT_s$, and $y(t)$ can be uniquely determined by lowpass filtering the samples $g(y(t_k))$ and transforming them by $g^{-1}(.)$ as in (2) and Fig. 2.

$$y(t) = g^{-1}\left\{ \sum_{k=-\infty}^{\infty} g(y(kT_s))sinc(t - kTs) \right\}. \qquad (2)$$

In this paper, we will introduce a random sampling technique that allows sample rates to be further relaxed. We will first cover the simple case of a memoryless nonlinear system, followed by the case of a linear system with memory. Then we extend these results to cover nonlinear systems with memory that can be decomposed into a memoryless nonlinear system plus a linear system with memory. Finally, we will examine the case of nonlinear systems with memory that cannot be decomposed.

Memoryless Nonlinear Systems

Let y = x(t) be an ideal signal. Let g(.) be a nonlinear, memoryless system that operates on y. Furthermore, let g(.) be a polynomial, or a system that can be well approximated by a polynomial. Because g(.) is memoryless, it cannot be a filter. Therefore, g(y) has an output bandwidth equal to bandwidth(y) * order(g), provided that g(.) does not perform some inverse operation on y such that higher order components are canceled. In other words, if g(.) is a third order nonlinear system, then it will usually produce harmonics up to three times the highest frequency component of y. What are the sampling requirements imposed on the output signal such that g(.) can be completely identified? In the case of evenly spaced samples, the technique of Input Nyquist Sampling allows us to sample at the Nyquist rate of the input signal y. In the case of randomly spaced samples, however, the sample rate is much lower. The primary enabler for this is a change in perspective. Rather than considering the error function in the frequency domain with all of its various harmonics, let us revert to a timeless input-output mapping of g(y) versus y. In this case, we know that we only need N+1 distinct pairs of values (y, g(y)) in order to identify g(.) as an Nth order polynomial system. In the presence of noise, gathering more than N pairs decreases the error in the measurement of the coefficients of g(.). Therefore, the sampling requirements do not need to be such that they guarantee the ability to capture any particular frequency content, they only need to be such that they can guarantee some degree of uniqueness of (y, g(y)) pairs.

Let us again consider the case of evenly spaced samples, now applied to this input-output mapping perspective. In this perspective, the minimum sampling rate requirement guards against the unlucky case of repeatedly sampling the same values of (y, g(y)), or put another way, of not getting enough distinct pairs of (y, g(y)). That is, it guards against bad luck in the sampling process that would result in features of interest being skipped. Random sampling intervals, on the other hand, generate a high probability of capturing unique points. Furthermore, unlike Input Nyquist Sampling, random sampling for long periods of time eventually results in coverage of the entire range of g(y) with an effective frequency resolution limited only by the timing resolution of the sampling timing engine.

Linear Systems with Memory

Now let us apply a similar idea to the case of a linear system with memory. Obviously, a linear system does not produce nonlinearities, but it may nonetheless introduce unwanted errors in the form of offset or gain errors, or it may exhibit phase distortion or other unwanted filtering.

Again, let y = x(t) be an ideal signal, but this time let h(.) be a linear system with memory. Unfortunately, we cannot perform a timeless input-output mapping like we did for the memoryless case because of the delays associated with h(.). However, we can employ a similar input-output comparison, this time with the inclusion of linear operations of y (derivatives, integrals and scaling).

EXPLAIN HOW H(.) CAN BE DETERMINED IN THE UNDERSAMPLED SYSTEM USING THE IDEA OF MAPPING AND THE IDEAS FROM (1).

Memoryless Nonlinear Systems Followed by Linear Systems with Memory

Again, let y = x(t) be an ideal signal and let g(.) be a memoryless nonlinear system. Additionally, let h(.) be a linear system with memory. The output of the composite system is then h(g(y)). In this case, we wish to identify both the nonlinearity of g(.) and the linear errors of h(.). It is possible to separate some errors due to h(.) from some errors due to g(.). First order errors cannot be separated. For example, if there were some first order error resulting from the composite function h(g(y)), then we have no way of knowing whether to attribute it to h(.) or to g(.). We can overcome this difficulty by defining g(.) such that it contributes only static nonlinearities and does not contain zeroth or first order terms. h(.) is then defined to contain all the zeroth and first order scaling (commonly referred to as *gain* and *offset*). In other words, we define g(.) and h(.) such that gain and offset are entirely contained in h(.)[1].

The primary distinction between these two functions is that the components of h(.), as a consequence of being linear, will not correlate over time to the nonlinearities in g(.).

USE (1) TO EXPLAIN HOW LACK OF CORRELATION BETWEEN G AND H CAN BE USED TO DETERMINE BOTH FUNCTIONS. THIS IS BECAUSE WE HAVE A PREDETERMINED MODEL OF G AND OF H. BECAUSE THE TWO MODELS ARE ORTHOGONAL (DO NOT CONTAIN THE SAME TYPES OF ERROS BY NATURE OF THE WAY WE DEFINED THEM) THEN EACH TYPE OF ERROR WILL ONLY CORRELATE OVER TIME TO THE APPROPRIATE MODEL.

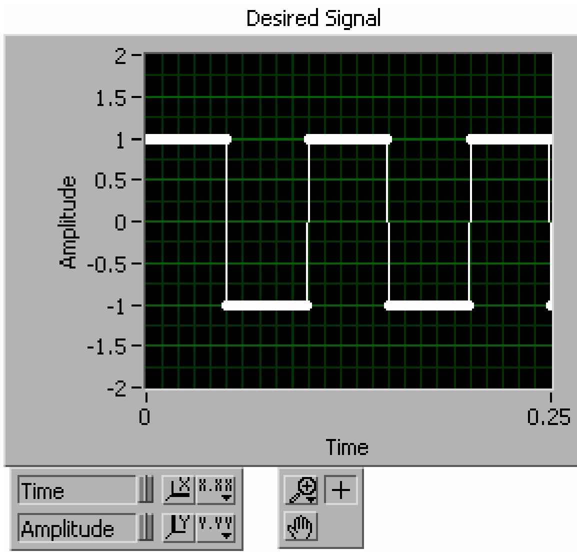
General Nonlinear Systems with Memory

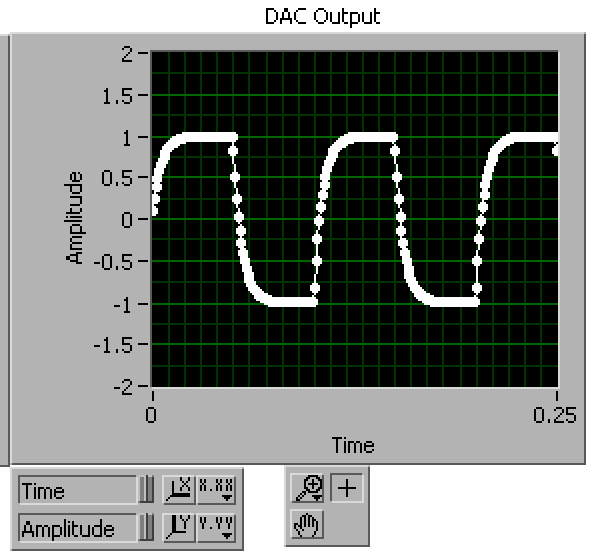
OBVIOUSLY THE MOST DIFFICULT CASE, BUT FOLLOWS FROM THE IDEA THAT, OVER TIME, ERRORS WILL ONLY CORRELATE TO THEIR APPROPRIATE MODELS. YOU JUST NEED A SET OF ORTHOGONAL MODELS THAT ADEQUATELY CHARACTERIZE THE SYSTEM. AGAIN THIS FOLLOWS FROM (1).

FINALLY, WE NEED TO FORMALIZE THE PERFORMANCE OF THE TECHNIQUE. HOW FAST DO WE CONVERGE? HOW DOES IT PERFORM IN THE PRESENCE OF NOISE? CAN WE QUANTIFY THE SAMPLING REQUIREMENTS AND DETERMINE THE PROBABILITY OF DETERMINING THE ERROR FUNCTIONS GIVEN A CERTAIN NUMBER OF SAMPLES?

Examples: Let us demonstrate these concepts with an example. Let's say we have a 10 Hz square wave signal, generated with a DAC updating at 1ksps. The ideal signal is shown below on the left, and the actual signal is shown on the right. In this case, the DAC has an output bandwidth limit of 20Hz, so the signal is heavily attenuated.

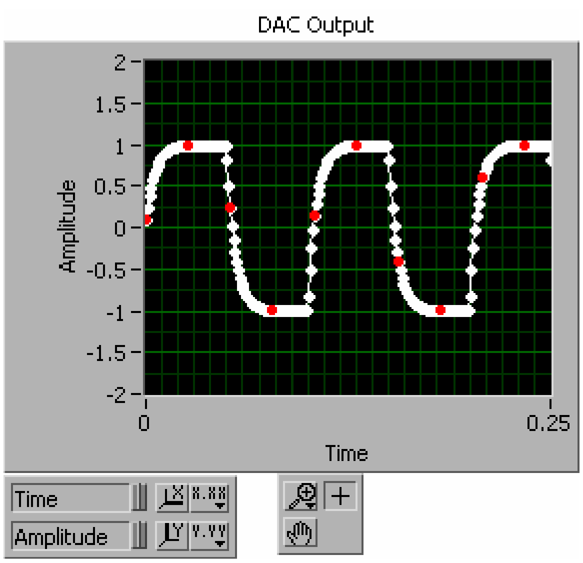
[1] There is a subtlety here that the two functions g(.) and h(.) must share a common reference of "errorless." When we say that g(.) does not contain any linear errors (gain or offset) we are faced with the difficulty of deciding how to define gain and offset in the presence of nonlinearities. To overcome this, we define g(.) such that a best-fit line of (y, g(y)) over the range of y has a slope of zero and an offset of zero.



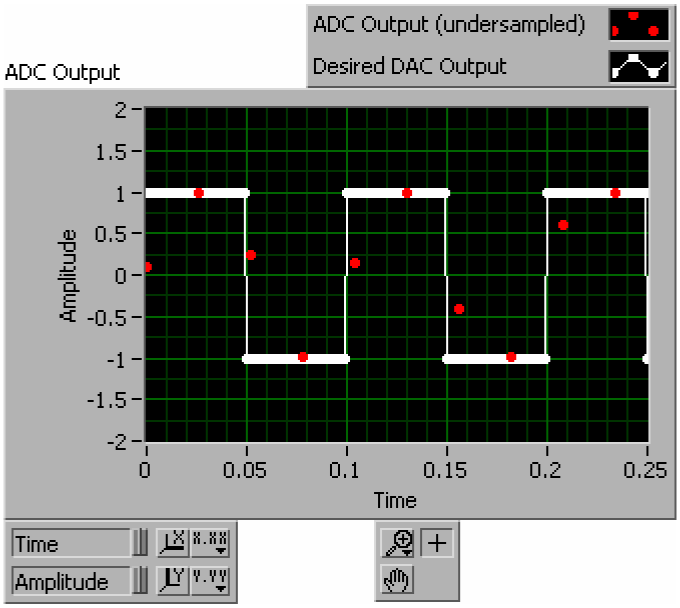


Desired Signal **Actual DAC Output**

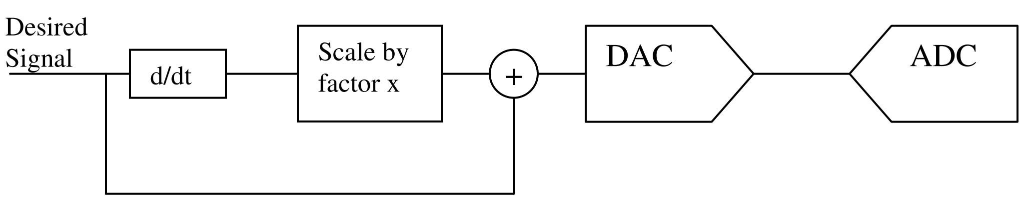
This situation is further complicated by the fact that we are undersampling the DAC Output. Here is what the ADC Output looks like if we use an undersampling factor of 26 (so the ADC samples 26 times slower than the DAC updates). This is a graph of ADC samples (red dots) superimposed on the Actual DAC Output (for reference).



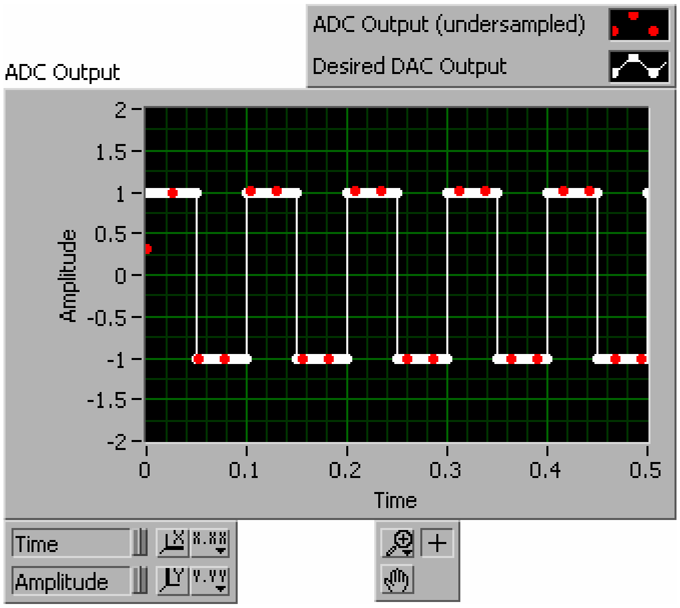
Finally, of course, we remember that we won't have access to the actual DAC output. The only two signals we will actually know are the ideal (desired) output and the ADC output (undersampled, actual DAC output). Plotting them on the same graph, we get the following:



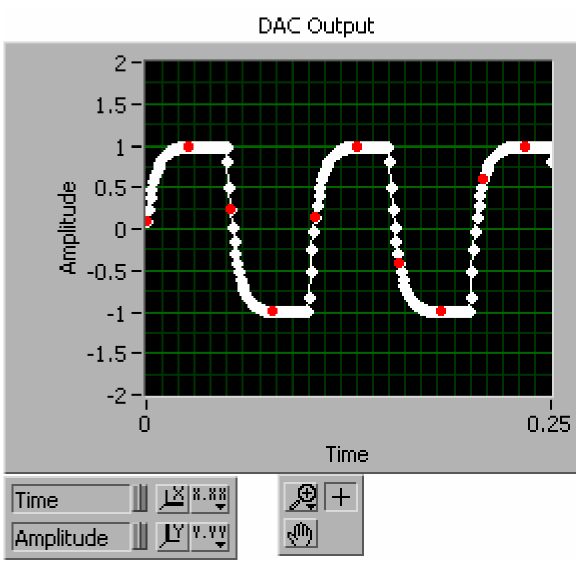
This situation at first looks completely hopeless, but the input and output signals can, in this case, be used to completely determine the DAC's behavior, despite the severe undersampling. Consider this diagram, which uses the DAC correction model developed in (4):



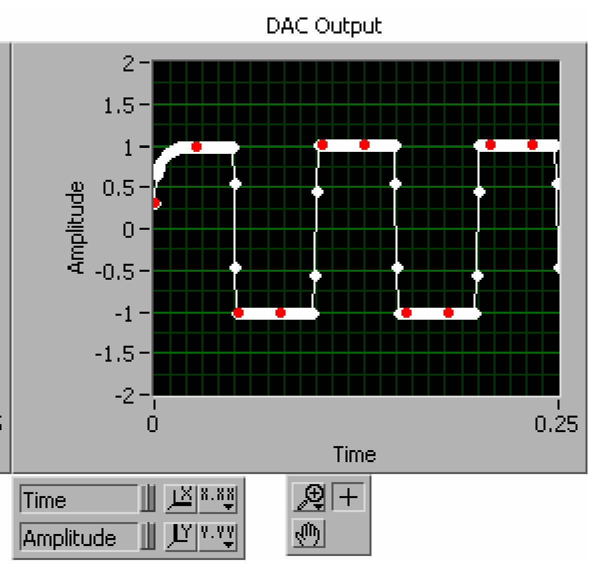
Pretend we have a manual knob that we can use to adjust the 'x' factor, meaning we can adjust the amount of input derivative that gets added in to the input signal. Simulating this in LabVIEW, I can create such a scenario. My goal then is to adjust the knob until the undersampled output (the red dots) sit right on top of the desired signal (the white dots). After doing my best manual adjustment, I get a graph like this one:



But of course the real test is to go back and see how well the actual DAC output matches the desired signal. In this case, we have done very well:



Uncorrected DAC Output



DAC Output with Predistortion

In these graphs, the red dots are the ADC output and the White dots are the actual DAC output. This shows that the need to determine derivative coefficients is not at all hampered by undersampling, and that we can achieve the desired square wave signal at the DAC output using coefficients derived in an undersampled system.

The next step is to remove the manual work from this process and let the correction happen in a feedback loop.

(3)

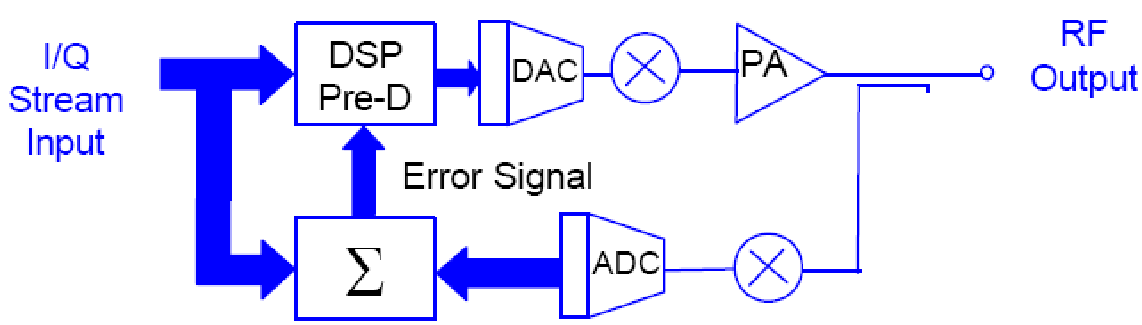
Correction of various errors in a Digital to Analog Converter using predistortion while operating within a mixed signal feedback loop

The problem of trying to correct for Digital to Analog Converter (DAC) errors in a mixed signal feedback loop is not new. In such a loop, DAC errors are corrected via a digital predistortion algorithm whose coefficients are generated by measuring the DAC output with an Analog to Digital Converter (ADC). However, most methods employed are constrained by Nyquist sampling criteria in reference to the ADC, meaning that they only work within one ADC Nyquist band. To perform this type of correction, anti-alias filters are used within the feedback loop, sometimes in conjunction with downconversion (in the case that the band of interest is not centered at DC) to present the ADC with a single Nyquist band of frequencies. The primary limitation in such a system is that the ADC must be able to run at the same speed as the DAC in order to determine everything about the signal. In actuality, the ADC sample rate requirement could be quite higher if there is a need to capture information about distortion products that are higher than the DAC Nyquist rate (which is often the case).

Additionally, techniques that do allow for the identification of system nonlinearities using undersampling, are generally still constrained by some type of Nyquist consideration. In this paper, we eliminate Nyquist sampling restrictions from such a system and present several ideas for correcting DAC errors. It is imperative that we let go of the idea of trying to determine everything about the *signal* and instead focus on the root problem of determining everything about the *system*. So although it is impossible to reconstruct a signal without obeying Nyquist rules, it is completely possible to reconstruct a system despite being in clear violation of them. This is analogous to using a slow RMS meter to characterize an analog filter. Even if the RMS meter only updates at a few Hz, you can still determine the filter shape by stimulating it at different frequencies and measuring input and output power at each frequency with your slow RMS meter.

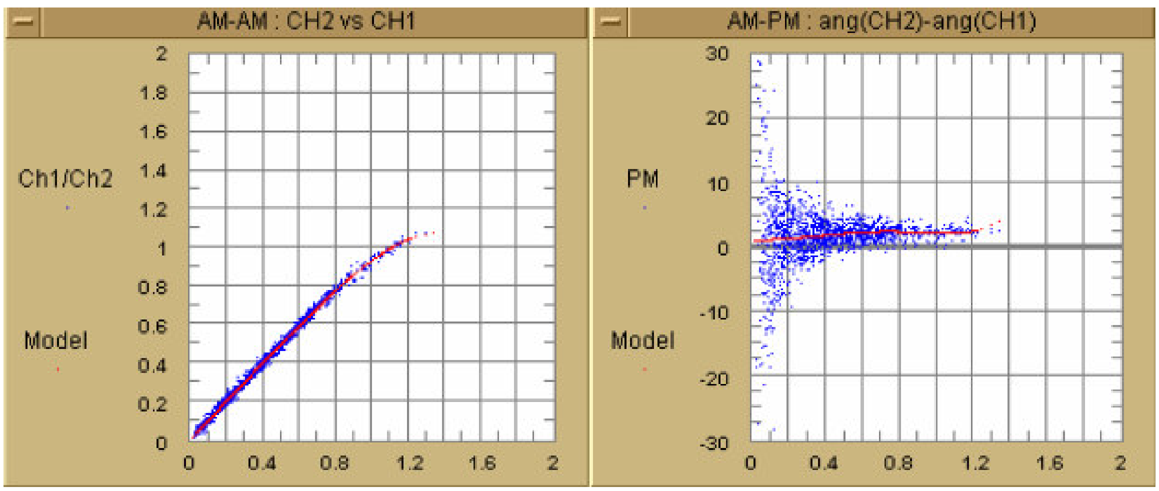
Prior Art

J Stevenson Kenney (GA Tech Presentation), 2002. This presentation highlights Predistortion Techniques for Cellular Base Stations. The high-level block diagram of this system is as follows:



In this case, the nonlinearity corrections are done in the baseband signal to correct for nonlinearities in the PA (which obviously does not operate in the baseband). In the case of correcting a general purpose DAC, we would not have the up-converter and down-converter in the feedback loop and the DAC output will be sampled by the ADC directly. But in searching for relevant predistortion correction concepts, the above architecture is actually much more prevalent, owing largely to the dominating influence of the communications industry where this architecture directly applies. The first change from this architecture is to slow the ADC rate considerably so that a higher resolution ADC can be used. As shown in (2), we do not need to adhere to Nyquist rules in order to characterize the system.

One other shortcoming of this design is the use of a memoryless Look Up Table (LUT) for linearity correction. This causes considerable trouble for systems with memory effects (the authors of this presentation themselves point out the resulting difficulties that arise with this technique when memory errors are present).



As shown in the AM-PM graph, the presence of memory effects causes the straightforward 1:1 mapping of input to output to fall apart. With enough samples, it may be possible to still extract an accurate model of the DC distortion, but it would still not solve the problem of correcting over frequency.

In summary, most prior art techniques are similar to this and are good for the intended base station application, but fall short in several areas that are important for a general purpose DAC. No attention is paid whatsoever to absolute accuracy. The majority of the focus is on eliminating the unwanted sideband errors caused by intermodulation products, which ultimately come from distortion. Also, because these techniques are intended for only moderate resolution (the ADC is only 10-bit), they do not address the correction of more subtle errors in the DAC. In fairness, they are exclusively interested in correcting the PA and not in dealing with any DAC errors – assuming them to not be a dominant source of system error.

New Correction Scheme: Using the ideas of (1) and (2) and the model presented in (4), we present a scheme for correction of errors in a general purpose DAC. Because this operates in a feedback loop, many of those ideas must be revisited. One glaring thing to determine is the stability of such a system. In other words, do the correction coefficients "settle?"

Before going very far, it is useful to point out a nice simplification resulting from a special case of these concepts. Supposed you could create a DAC where the distortion mechanisms are not frequency dependent. In that case, DC correction is the only thing you have to do. Although this may sound unrealistic at first, it is helpful to consider the reasons why distortion changes over frequency. In many cases, distortion at DC is so good because of low-frequency, analog feedback loops. In order to keep these loops stable, loop gain, the primary mechanism for limiting distortion, must roll off quickly. On the other hand, if the DAC and its following circuitry were open loop (in the analog sense), then distortion products might not become significant until very high frequencies. Certainly, excellent DACs could be built using this concept, and they would be of fairly low complexity. Such a mixed-signal loop could even run fast enough to chop 1/f noise at frequencies as high as 1kHz or so.

DAC Requirements: The purpose of these types of technique is to relax the quality of the analog components in the signal path. However, we recognize that the system must maintain certain characteristics in order to be correctable. Some requirements are obvious, such as monotonicity and stability, while others are not. Here is a short and probably incomplete list of analog requirements of the DAC.

First of all, the distortion mechanisms and other errors must change slowly. The more slowly they chance, the longer the allowable memory of the correction coefficients can be.

ADC Requirements: The ADC does not need to be fast, but it needs to not have frequency dependent distortion in the sampling circuitry.
Possible ways to correct for ADC errors: Utilize an ideal signal produced using a PWM DAC. The ADC reads the sum of the assumed-errorless PWM DAC signal plus the DAC output for which we wish to correct. Because the PWM signal is random, it is uncorrelated with the ideal DAC output. In this way, ADC errors can be extracted. The technique of using a random ideal signal to correct for ADC errors on the fly is also listed as a "future idea."

Practical Limits: DAC cannot do all that much overshoot in order to correct for rolloff

(4)

Development of a simple function to correct for distortion elements with memory to be used within a feedback loop

In this paper we develop a good model of DAC errors. The motivation is that we will then be able to create a function which can be used to predistort the DAC signal in the digital domain. By providing the correct inverse function, we can achieve accurate signal generation using cheaper analog components than would otherwise be possible. Let us first examine common types of DAC error.

DC Errors: This includes things like gain, offset, INL, DNL, and lower order nonlinearities resulting from analog signal processing (note that these can be included in INL correction rather than extracted separately). If we can determine what the DC errors are, we will not have trouble correcting for them. Any number of existing techniques can be employed for the actual correction, from polynomial functions, which are effective when lower order, analog distortion is dominant, to binary-weighted correction, which is most effective when binary-mismatch is dominant. In the case that both types of errors contribute, we could do some combination of INL/DNL correction as well as polynomial correction, or we could use segmented correction (where upper bit correction is done based on patterns rather than strictly bit-wise) giving the ability to approximately fit lower order distortion on top of the high-order binary weighted error.

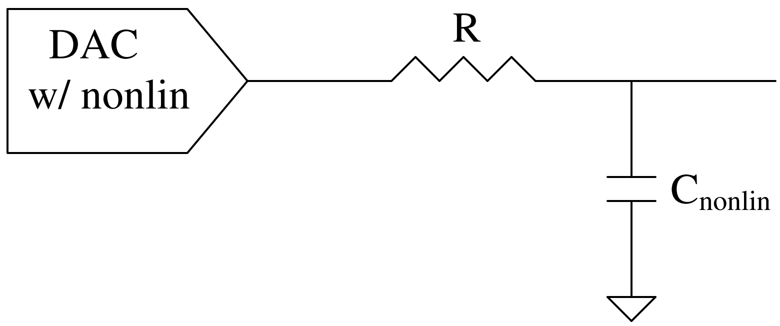
In any case, the more difficult question is how to determine what the nonlinearities actually are. The most basic method is to map signal out vs. signal in as suggested in (2). Using the orthogonality argument of (1), we know that over time, frequency dependent errors and other uncertainties will average out and DC errors can be extracted by curve fitting the residual error. An analogous method, focused on binary weighted errors, would be to keep track of the amount of error that occurs anytime certain bits are asserted. Again, over time other errors will average out and the DC characteristics can be extracted.

Linear Frequency-Dependent Errors (flatness, phase distortion)
Linear frequency-dependent errors are those that result from linear circuit models. Correction is not necessarily easy, but there exists a rich foundation of mathematics that allows for the computation of an inverse function of the linear transfer function in order to predistort the signal. Therefore, I will not address this directly. Additionally, I believe it fits into the more general, nonlinear, frequency-dependent errors described next.

Nonlinear Frequency-Dependent Errors (distortion)
In addition to linear frequency response, the DAC may have any amount of frequency dependent distortion. Such distortion may be things like nonlinear capacitances, loss of loop gain at high frequencies, slew rate limitations, asymmetrical slew rate limitations, asymmetrical drive strength, etc. It is impossible to create a general model that will work for every type of signal distortion that you can imagine (with the possible exception of neural networks, which are extraordinarily adaptive, but may be in fact too flexible to be efficient). In fact, there generally exists no closed form solution to the nonlinear

differential equations that result from such systems. Therefore we will have to make some assumptions about the nature of the expected types of distortion and then derive a model that will handle the largest number of cases. This will also place constraints on the actual DAC architectures that can be used.

Simple DAC Model with Both DC Errors and High Frequency Errors
In order to facilitate the development of a simple correction model, we first need a model of the DAC. The model I have created is straightforward: a DAC with DC errors that has some type of DC nonlinearity, followed by a nonlinear RC circuit.



Although conceptually straightforward, this circuit allows the development of a model that can be expanded to cover more complex cases. This model incorporates a distortion element with memory (the nonlinear capacitor) as well as a distortion mechanism without memory (the DAC). The memoryless errors are the result of mismatches within the DAC. They are memoryless in the sense that the mismatches are only a weak function of the derivatives of the signal. Although not all DACs actually behave this way, we can purposely develop one that does. The nonlinear RC circuit that follows represents nonlinear RC errors that could actually reside within the DAC itself, such as those that come from analog switches or from any internal analog processing. Or these could also be external errors in the amplifier(s) following the DAC.

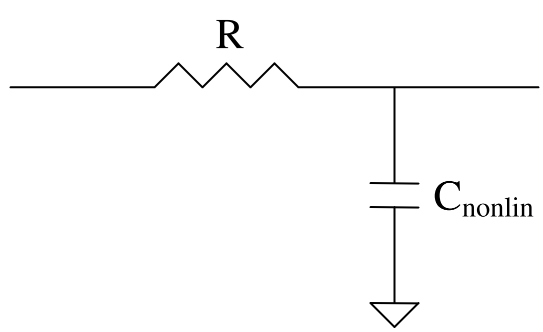
I'm also taking the nonlinearity of the C to be only voltage dependent (not frequency dependent or power dependent (many PA's have rapidly dynamic power dissipation characteristics that must be accounted for)). This means that the capacitor can be modeled as follows:

$$C_{nonlin} := C_0\left(1 + \alpha_1 \cdot V_o + \alpha_2 \cdot V_o^{\,2} + \alpha_3 \cdot V_o^{\,3} + ..\right)$$

The resistor in the model is ideal.

First of all, development of the correction model is non-trivial. It cannot be modeled by separating the elements with memory from the elements with distortion and treating the system as a cascaded memoryless circuit with distortion, followed by a distortionless circuit with memory (Such as talked about in Pedram's EE391 paper). It is also not a simple input to output mapping that perhaps changes over frequency and that could be kept in a 2 dimensional (voltage and frequency) LUT. In this case, the current output voltage depends on the recent history of the time-value of the output voltage (not just on the present frequency of the signal – whatever 'present frequency' means for arbitrary signals). This is, in fact, the whole difficulty of dealing with systems that have memory. It becomes clear, therefore, that a good predistorter must utilize derivatives in order to make corrections that will hold over frequency.

Analysis of the nonlinear RC errors



The differential input-output equation is as follows:

$$\frac{V_{out} - V_{in}}{R} := C_{nonlin}\left(\frac{dV_{out}}{dt}\right)$$

$$V_{out} - V_{in} := R \cdot C_{nonlin}\left(\frac{dV_{out}}{dt}\right)$$

$$V_{out} - V_{in} := R \cdot C_0\left(\frac{dV_{out}}{dt}\right)\left(1 + \alpha_1 \cdot V_o + \alpha_2 \cdot V_o^{\,2} + \alpha_3 \cdot V_o^{\,3} + ..\right)$$

This equation states that, if we assume C to be a function of Vout, then correcting for its errors comes down to determining the coefficients of this polynomial. If we distribute the RCo term into the polynomial, then we can make the analysis more transparent:

$$V_{out} - V_{in} := \left(\frac{dV_{out}}{dt}\right)\left(R \cdot C_0 + R \cdot C_0 \cdot \alpha_1 \cdot V_o + R \cdot C_0 \cdot \alpha_2 \cdot V_o^{\,2} + R \cdot C_0 \cdot \alpha_3 \cdot V_o^{\,3} + ..\right)$$

Then cleaning the equation up with a small substitution, we get:

$$R \cdot C_0 \cdot \alpha_i := \beta_i \qquad\qquad V_{out} - V_{in} := \left(\frac{dV_{out}}{dt}\right)\left(\beta_0 + \beta_1 \cdot V_o + \beta_2 \cdot V_o^{\ 2} + \beta_3 \cdot V_o^{\ 3} + ..\right)$$

Therefore, the error is a function of the derivative of the output itself. No closed for solution exists to this problem, but we do not need a closed form solution when the model operates within a feedback loop. We simply need a model that provides the adequate knobs and then we let averaging and curve fitting tools correlate the system output to the ideal output. All we need is a function that has adequate flexibility.

There is still a lot of work to be done here. We have access to the input signal and we must create a model that uses this input signal plus its history plus its derivatives, etc.

FUTURE IDEAS

Flex Resolution DAC: Using predistortion and digital smart dithering, we could create a high resolution DAC using a low resolution DAC. By knowing the INL and DNL characteristics and oversampling the DAC and filtering, very accurate voltages can be produced. Glitch energy limitation?

ADC that adds a random signal to the input signal to dither and correct for linearity:

Neurocontrollers: Because neural networks are so flexible, might they be used to adapt to distortion and other error components and learn the best predistortion function? Because of the speed with which you can acquire data points, it is possible for these networks to evolve fairly quickly.

RANDOM NOTES

Doug mercer
Training mode

Lookup ETS sampling requirements

Must determine the d/dt coefficient at every voltage level (for nonlinear distortion). The number of derivatives you need corresponds to the order of the *linear* system. The number of polynomial coefficients corresponds to the order of the distortion.