ADAPTIVE COMPENSATION OF ANALOG-TO-DIGITAL CONVERTERS

By
James J. McDonald II

B.S. University of Maine, 1996

A THESIS
Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science
(in Electrical Engineering)

The Graduate School
University of Maine
August 1997

Advisory Committee:

Donald M. Hummels, Associate Professor of Electrical and Computer Engineering, Advisor

Fred H. Irons, Castle Professor of Electrical and Computer Engineering,

Allison I. Whitney, Lecturer in Electrical and Computer Engineering.

LIBRARY RIGHTS STATEMENT

Adaptive Compensation of Analog-to-Digital Converters

By James J. McDonald II

Thesis Advisor: Dr. Donald M. Hummels

An Abstract of the Thesis Presented

in Partial Fulfillment of the Requirements for the

Degree of Master of Science

(in Electrical Engineering)

August 1997

Previously, analog-to-digital converters have been compensated with state and slope dependent error models. These compensation methods are very effective; however, variations in temperature, power supply voltages, and other environmental factors affect the accuracy of the compensation over time.

This thesis will present the use of adaptive compensation algorithms, including Kalman filters and Least Mean Square (LMS) filters, to track and compensate for these time dependent changes. The effectiveness of the compensation will be demonstrated with extensive use of simulations, and real analog-to-digital converter output data.

This thesis will discuss, in depth, the trade-offs between tracking speed and tracking accuracy that must be made. It is shown that proper choices for these parameters, yield results closely matching the performance obtained by Larrabee [10].

ACKNOWLEDGMENTS

This work has been supported in part by the ARPA HBT/ADC program under a contract administered by the Office of Naval Research Grant N000149311007, and the Army Research Office Grant DAAH04-94-G-0387.

There are many who have helped this thesis come about. First I wish to thank Erica my wife. Without her support and endless sacrifices, I would never have written this. Thanks to Morgan for being my inspiration and best accomplishment. Thank you to my father, James, who always lets me make my mistakes and helps me get up when I fall. To my mother who always supports me when I need it most.

Thanks to Professor Don Hummels for being such a great teacher, in class and out. Thank you Professor Fred Irons, for having confidence in my abilities when I didn't have confidence in myself, and for being such a fantastic teacher. Thank you Al Whitney for always answering questions, even when it's not your class, and for teaching me to "Think about what's going on.".

I can't express the gratitude I have for each of you, for all you have done.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

Introduction

As high speed communications systems become more digital, accurate Analog-to-Digital Converters (ADCs) are becoming a critical component. These converters are necessary to convert the high speed analog signal to a digital signal.

Many advances have been made in designing accurate converters, however these converters still have harmonic distortions that affect their accuracy. By applying compensation techniques to these high speed converters, their accuracy can be significantly increased. This thesis presents algorithms for adaptively compensating these high speed converters.

1.1 Background

ADC compensation is not a new idea [6, 7, 9, 10]. Initial compensation models were static where ADC error was modeled as a function only of the output value, or state. These models were inadequate, and therefore state and input signal slope dependent error models were adopted [13].

Originally, the state and slope dependent error was modeled as a linear combination of a set of smooth predefined basis functions such as Gaussian functions or polynomials [9, 11]. These basis functions are continuous functions of state and slope with the coefficients estimated to model the error characteristics in a least square sense. Compensation with this dynamic model requires a table, or memory, where the state and slope are used to address an error value. This value is subtracted from the output value of the converter, creating the compensated sample.

Many methods for developing this table have been presented. Least square solutions have been used to find the basis function coefficients for an arbitrary set of basis functions [8, 5]. Neural networks have been used to find relevant basis functions and their coefficients [1]. For all of these methods, the error is estimated from an FFT. During calibration, the converter is excited by a spectrally pure sinusoid, and an FFT of the output is calculated. The error is estimated from the harmonics of the fundamental in the FFT spectrum.

Larrabee [10] has presented a new calibration technique based on histogram methods. This method estimates the error from a statistical distribution of the input to the converter. Larrabee used a different model for error, where the basis functions are not continuous in the state direction, but error is modeled as a continuous polynomial function of slope at each ADC output state.

1.2 Purpose of the Research

The compensation method developed by Larrabee [10] has the ability to compensate very complex error characteristics. However, the procedure requires that the converter be calibrated off-line, in a different environment from the environment where the converter is used. Variations in temperature, reference voltages, and other environmental factors affect the accuracy of the compensation model.

This thesis presents an adaptive compensation method, that will compensate a converter in the location it is used. This adaptive method also accounts for changing environments. If the ambient temperature, or reference voltages in the converter's location change significantly, the compensation will adapt to such changes.

1.3 Thesis Organization

This thesis presents adaptive algorithms for ADC calibration. These algorithms are discussed and compared. Chapter 2 discusses error modeling in Analog to Digital Converters. Previous models are discussed as well as the models used in this thesis. Chapter 3 gives a short introduction to the Discrete Kalman Filter, and describes the Kalman equations. Chapter 4 describes the conversion of the general Discrete Kalman Filter to a model describing ADC error. Chapter 5 presents the Least Mean Square (LMS) algorithm, another algorithm useful for adaptive ADC compensation. Finally, Chapter 6 gives a short comparison of the algorithms developed for this thesis.

CHAPTER 2

Modeling Error as a Function of State and Slope

Ideally an ADC converts a continuous input signal to an output binary code that is proportional to the input value at some sample instant. With the addition of a dither signal to the ADC input signal, an ideal ADC would introduce no harmonic distortion. Unfortunately real ADCs introduce error into the output code which in turn introduces harmonic distortion and limits the dynamic range performance of the ADC.

2.1 State and Slope Dependent Error

It has been shown that dominant ADC errors are dependent upon state x, the output code representing the sample input value, and slope y, the rate the input signal is changing at the sample instant. Most compensation methods therefore use models that define the error as a function of state and slope, $e(x, y)$. The state, slope, and error define a three dimensional space. The state and slope form the x and y plane respectively, and the error is on the z axis.

2.1.1 Previous Models

A common model for the error is to write $e(x, y)$ as a sum of basis functions and their respective coefficients.

$$e(x, y) = \sum_{l=0}^{m-1} \alpha_l b_l(x, y) \tag{2.1}$$

In (2.1), m is the number of basis functions, $b_l(x, y)$ are the state and slope dependent basis functions, and α_l are the respective basis function coefficients.

Initial state and slope compensation methods used smooth basis functions, resulting in error models which were slowly changing in the state and slope directions. A common example is the three dimensional Gaussian pulse. The pulse may be visualized as a hill in the state-slope plane, centered at x_l and y_l, and is described as .

$$b_l(x,y) = e^{-\frac{1}{2}(\frac{x-x_l}{\sigma_x})^2 - \frac{1}{2}(\frac{y-y_l}{\sigma_y})^2}. \tag{2.2}$$

A problem with this model is that it assumes the error from state i to $i+1$ is continuous and slowly changing. Because of numerical limitations in solving for the coefficients, these methods are typically limited in use to the order of 100 basis functions.

Recent work [10] has adopted a different error model. It has been found that the error from one state to the next is seldom smooth. To accommodate these state-to-state discontinuities, a discontinuous error model is used in the state direction. The error is written independently for each state and the basis functions are only dependent upon slope.

For state i the slope dependent error is represented by (2.3):

$$e^{(i)}(y) = \sum_{l=0}^{m-1} \alpha_l^{(i)} b_l(y) \tag{2.3}$$

In this equation $\alpha_l^{(i)}$ is the l^{th} basis function coefficient for state i. The m basis functions $b_l(y)$ are the same for each state and are only slope dependent.

If eight basis functions are used to represent the error as a function of the slope, and the converter is an eight bit converter, this method will use $256*8 = 2048$ basis functions.

2.1.2 Dynamic Error Model Used For This Thesis

In this thesis, as in the work performed by Larrabee [10], the error for each state $e^{(i)}(y)$, where i is the state, is modeled as a function of input slope. However, in this thesis the basis function coefficients are assumed to be changing with time. For the k^{th} time step the error is modeled by by (2.4). Note that the basis functions are not changing with time, but the coefficients $\alpha_{k,l}^{(i)}$ and the resulting error $e_k^{(i)}(y)$ are time dependent.

$$e_k^{(i)}(y) = \sum_{l=0}^{m} \alpha_{k,l}^{(i)} b_l(y) \tag{2.4}$$

The algorithms presented in this thesis will adaptively find the coefficients for the basis functions, $\alpha_{k,l}^{(i)}$, thus tracking any changes in the converter characteristic.

2.2 ADC Calibration

This thesis makes extensive use of mathematical simulation and actual ADC data to test the concepts presented. Effective compensation relies upon accurate error estimation. To assure accurate error measurements of the real ADC, a high frequency test facility was used to test the actual ADCs used in this research. The test bed configuration is described in the following section.

2.2.1 Obtaining Calibration Data

This thesis is based on calibration in which the converters tested are excited with sinusoidal signals. Sinusoidal signals are easy to generate with out extensive

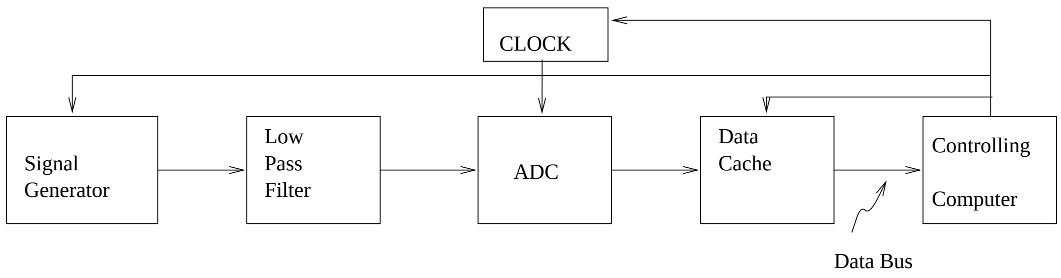

Figure 2.1: Calibration Configuration

test equipment, and error is easy to estimate.

A test configuration is shown in Figure (2.1). The input sinusoidal test signals are filtered to assure adequate spurious free dynamic range (SFDR) for the converter resolution. The SFDR is defined as the difference in dB from the amplitude of the fundamental test frequency, to the next largest component in the output spectrum. For this thesis all calibration signals had SFDRs above 90 dB. Sequential output sample sets of the ADC for each test signal, are acquired for analysis. The corresponding slope y is estimated from the fundamental harmonic. The slope is divided by f_s, the sample frequency, to normalize it's range.

If the state of the converter is plotted against the normalized slope estimates for each state, a trajectory plot is created. This trajectory, for a single sinusoid, is an ellipse in the state - slope plane. A single state on a trajectory is associated with two points on the ellipse, one positive slope and one negative slope. The amount of data to be manipulated is significantly reduced from the large array of calibration data with an array of state - slope pairs. The pairs are formed by the set of all possible states and the corresponding *average* positive (or negative) slope for that state.

Accurate compensation requires that the state-slope plane be well covered. As more test frequencies at various amplitudes are used to excite the converter, the trajectories begin to cover the state-slope plane. The compensation is only

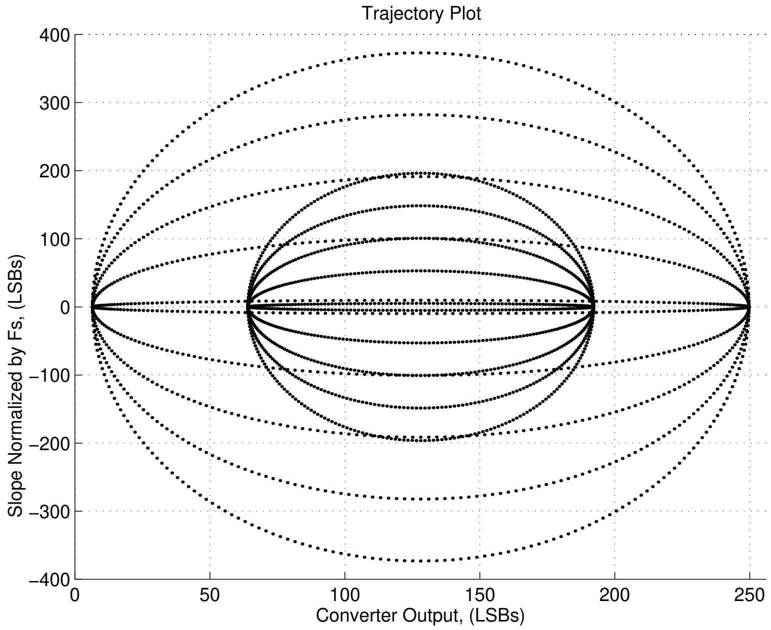

Figure 2.2: Trajectories in State-Slope Plane

effective where the converter has been calibrated. Figure 2.2 shows trajectories in the state - slope plane for five different frequencies at two different loadings (amplitudes).

2.2.2 Estimating Error

In order to calibrate the converter it is necessary to measure (estimate) the error. There are many ways to measure this error and two are discussed in the following sections. Once the converter error is estimated for each sample the average error for each state and average-slope pair is calculated and used to control the adaptive system.

2.2.2.1 Harmonic Distortion

For a single trajectory, the dominant error introduced by the ADC is observed as harmonic distortion and is evident from a plot of the DFT of the samples.

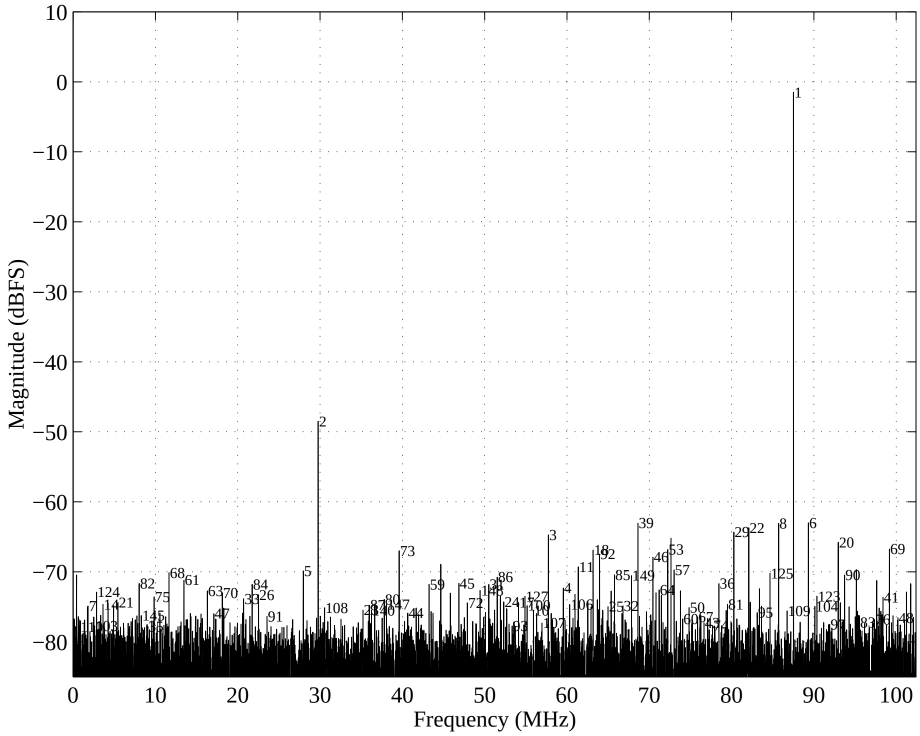

Figure 2.3: Spectrum of Uncompensated Samples

The spectral components of the samples which are not located at the test frequency f_T , are harmonic distortion introduced by the ADC. Figure (2.3) illustrates the loss in SFDR as a result of harmonic distortion. All components which are harmonically related to the fundamental are labeled, up to 150 harmonics. An ideal converter would have none of the harmonically related components in the output spectrum. The SFDR of this converter, at this frequency and loading, is measured between the fundamental harmonic which is at (-1) $dBFS$ and the second harmonic which is at (-48) $dBFS$. This gives an SFDR of $(-1-(-48)) = 47$ dB.

The error associated with these harmonic components can be estimated using DFTs and inverse DFTs (IDFTs). With M harmonics taken from a DFT, an inverse DFT with it's fundamental set to zero, as in (2.5), yields a time representation of the harmonic distortion.

$$e(n) = \sum_{l=2}^{M} (E(l)e^{j2\pi l n f_t/f_s} + E(l)^* e^{-j2\pi l n f_t/f_s}) \qquad n = 0, 1, \ldots, N_s - 1 \qquad (2.5)$$

In this equation, M is the number of harmonics, which are located at lf_t, $E(l)$ is the complex component for the lth harmonic frequency, and $E(l)^*$ is the conjugate term for the lth harmonic. The discrete time index is n. The error, $e(n)$, is averaged over all samples at each state, to obtain an expected error for each corresponding state. Note that the error averaging is done for both a positive and negative slope at each state.

2.2.2.2 Histogram Method

In the work presented by Larrabee [10], the expected error is calculated using a histogram. The histogram is used to determine the transfer function characteristic of the ADC. This transfer characteristic is compared to an ideal non-quantizing transfer function. The difference in transfer functions is the error, and this error can be converted into an expected error for each state. Previously ADC error was measured using the DFT method discussed. This limits the type signals that may be used to excite the converter. The histogram method, however, allows a much wider class of calibration signals to be used. All that is required is knowledge of the signal's statistical distribution function.

The average error for state i is the expected error, and is denoted by E_i. The expected error for state i may be written in terms of the quantization thresholds,

t_i.

$$E_i = i - G\left(\frac{t_i + t_{i+1}}{2} - x_o\right). \quad i = 1, 2, \ldots, 2^n - 2 \tag{2.6}$$

G and x_o are the gain and offset of the nominal (ideal) characteristic, n is the number of bits for the ADC, and t_i, t_{i+1} are quantization thresholds that yield the ith output state. Equation (2.6) gives the expected error as a function of output code in terms of the quantization thresholds t_i and t_{i+1}. These thresholds may be estimated for a sine wave by (2.7), where S_{i-1} is the cumulative number of samples less than output code i and N is the total number of samples. In this equation, A is the amplitude of the input sine wave, and C is the DC offset.

$$\hat{t}_i = -A\cos\left(\pi\frac{S_{i-1}}{N}\right) + C. \quad i = 1, 2, \ldots, 2^n - 1 \tag{2.7}$$

2.2.3 Compensation

Once the error has been measured and the basis function coefficients found, an error table is created. This table can be plotted in 3 dimensional space, where the state forms the x axis, the slope forms the y axis and the error the z axis.

Figure 2.4 shows an example of an error table. Given a state value and a slope, the compensation value is found from the table. By adding this value to the ADC output sample, the linearity of the ADC may be improved. Figure 2.5 shows a spectrum after compensation for the same test signal as Figure 2.3. Note that height of the second harmonic has been reduced to -60 $dBFS$, improving the SFDR to $(-1-(-60)) = 59$ dB for a 12 dB improvement in dynamic range over the uncompensated performance.

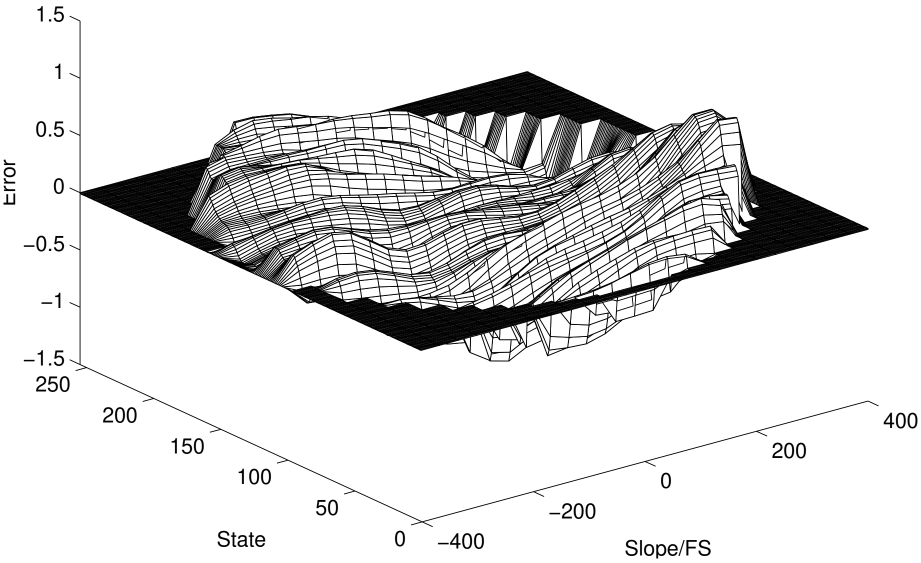

Figure 2.4: Error as a Function of State and Slope

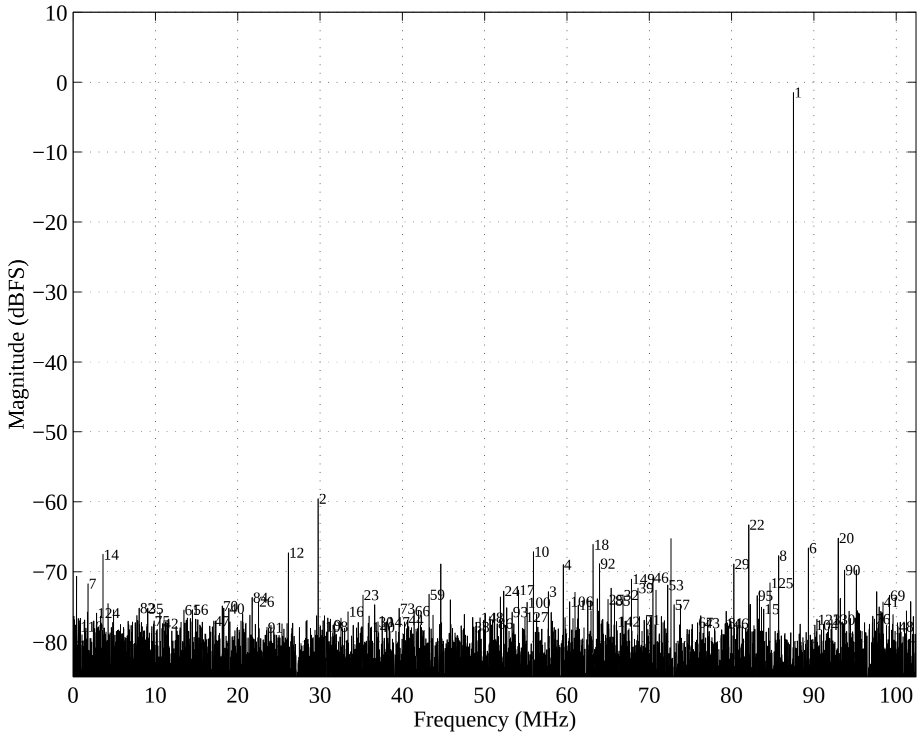

Figure 2.5: Spectrum of Compensated Samples

CHAPTER 3

Discrete Kalman Filter

Kalman Filters are commonly used in tracking problems that can be put in a linear form. The Kalman filter is used to track variations in a state variable vector $\vec{u}_k$, of a linear system with random inputs. The state estimates are formed from noisy observations of the system output. This chapter introduces the Kalman filter, and the next chapter discusses how the filter can be used to track errors of an ADC.

3.1 Kalman Equations

The Kalman filter is an algorithm for optimally estimating the unknown state vector, $\vec{u}_k$ from a sequence of noisy observations ($\vec{z}_k$) of linear functions of the state variables [2]. The model for the system is shown in Figure 3.1. B_k is a matrix which defines a linear combination of state variables, to be measured. $\vec{z}_k$ is generally a vector of length different from the length of $\vec{u}_k$. The noise in this measurement is $\vec{v}_k$ which is assumed white, with covariance R_k. This model is described by two separate equations, given in (3.1) and (3.2). The first equation (3.1)

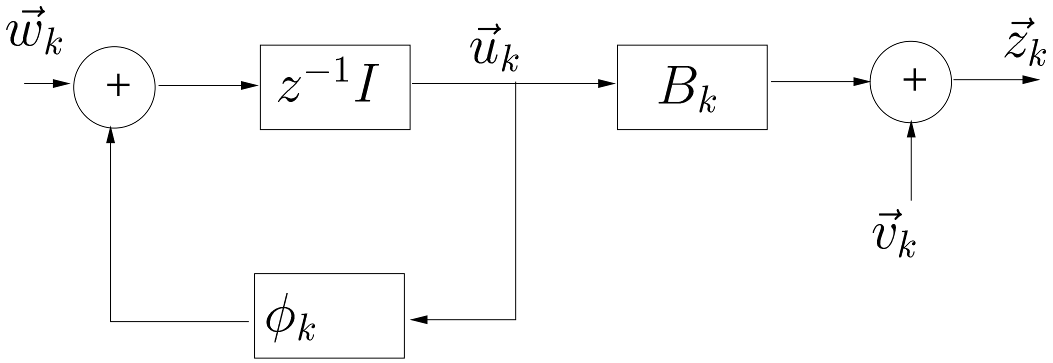

Figure 3.1: Kalman Model Block Diagram

is called the "system equation" and describes how the state variables are changing from each discrete time interval to the next. The second equation (3.2) is called the "measurement equation" and relates the state variables to the measurement vector.

$$\vec{u}_{k+1} = \phi_k \vec{u}_k + \vec{w}_k \tag{3.1}$$

$$\vec{z}_k = B_k \vec{u}_k + \vec{v}_k \tag{3.2}$$

The state variables change according to (3.1),where ϕ_k is the state transition matrix, and w_k is a random (white) input with covariance matrix Q_k. The state transition matrix, ϕ_k, maps changes in the state variables, and determines how $\vec{u}_k$ is changing relative to $\vec{u}_{k+1}$. The state transition matrix is a function of the system to be tracked and may vary with time.

The Kalman filter is an iterative procedure for estimating the state variables from a sequence of measurements ($\vec{z}_0$, $\vec{z}_1$, ... , $\vec{z}_k$). The filter assumes that the measurments $\vec{z}_k$ are known, as well as the transition matrix, ϕ_k , and measurement matrix, B_k. It is also necessary to assume covariances R_k and Q_k are known. The random inputs $\vec{w}_k$ and $\vec{v}_k$ are unknown. The state variables $\vec{u}_k$ are random and are only measurable indirectly, through $\vec{z}_k$. Each step of the algorithm involves updating the current estimate of $\vec{u}_k$ (from $\vec{z}_0$, $\vec{z}_1$, ... ,$\vec{z}_{k-1}$) to an improved estimate based on ($\vec{z}_0$, $\vec{z}_1$, ... ,$\vec{z}_k$).

3.2 Updating the Kalman Equations

This section provides an overview of the steps required to implement the Kalman filter. A more complete derivation may be found in [2]. Before proceeding it is necessary to define more notation.

The actual state variables, which are unknown, are $\vec{u}_k$. These state variables will be estimated from all of the previous measurements $\vec{z}_0$, $\vec{z}_1$, ... , $\vec{z}_{k-1}$, and this estimate will be written as $\vec{\hat{u}}_k^-$. This estimate of the state variables, $\vec{\hat{u}}_k^-$, has covariance denoted as P_k^-. After the kth measurement, the estimate from measurements $\vec{z}_0$, $\vec{z}_1$, ... , $\vec{z}_k$ will be $\vec{\hat{u}}_k$. The covariance after the new measurement is P_k.

To begin the Kalman filtering, it is necessary to use an initial estimate of the state variables. This estimate is commonly the mean value of the state variables, $\hat{u}_0^- = \bar{u}_0$. This initial estimate has covariance matrix, P_0^-, which is assumed known.

Measuring $\vec{z}_k$ is the initial step in each iteration of the Kalman filter. The measurement vector, $\vec{z}_k$, is used to update the estimate of the state variables.

$$\vec{\hat{u}}_k = \vec{\hat{u}}_k^- + K_k(\vec{z}_k - B_k \vec{\hat{u}}_k^-) \tag{3.3}$$

where K_k, the "Kalman Gain Matrix" minimizes the mean square estimation error. K_k may be shown to be

$$K_k = P_k^- B_k^T (B_k P_k^- B_k^T + R_k)^{-1} \tag{3.4}$$

where R_k is the covariance matrix of v_k, the noise in the measurement vector[2]. Note that in equation (3.3) the new estimate of the state variables is the old estimate plus the Kalman gain matrix times $(\vec{z}_k - B_k \vec{\hat{u}}_k^-)$. The term in parentheses is the measurement vector minus the filter's best estimate of the measurement vector. As the filter makes better estimates, this quantity will decrease. The values in the Kalman gain matrix will generally get smaller as the eigenvalues of the covariance matrix P_k^- decreases, and the $(\vec{z}_k - B_k\vec{\hat{u}}_k^-)$ term will have less effect upon the state variable updates.

The covariance of the estimate is now updated as well, using (3.5).

$$P_k = (I - K_k B_k) P_k^- \tag{3.5}$$

Tracking the covariance matrix determines the influence that each new measurement of the process will have. If the covariance is large, measured values will be weighted more heavily. If the covariance is small, the measured values will affect the state variables less.

The Kalman filter now uses the ϕ_k matrix to project the present state variables, $\vec{\hat{u}}_k$, to the next estimated state variables, $\vec{\hat{u}}_{k+1}^-$. The best estimate of the next state variables will be the current best guess, applied to the linear system describing their behavior.

$$\vec{\hat{u}}_{k+1}^- = \phi_k \vec{\hat{u}}_k \tag{3.6}$$

The ϕ_k matrix is also used to project the covariance matrix to the new covariance. The estimate covariance is updated by (3.7).

$$P_{k+1}^- = \phi_k P_k \phi_k^T + Q_k \tag{3.7}$$

This completes one iteration of the filter. A new measurement is now made and a new iteration is begun. As the Kalman filter makes measurements, the filter continues to update the estimates of $\hat{u}_k$, and P_k. As the filter gains confidence in the estimates of the state variables, the covariance will reflect that confidence, by

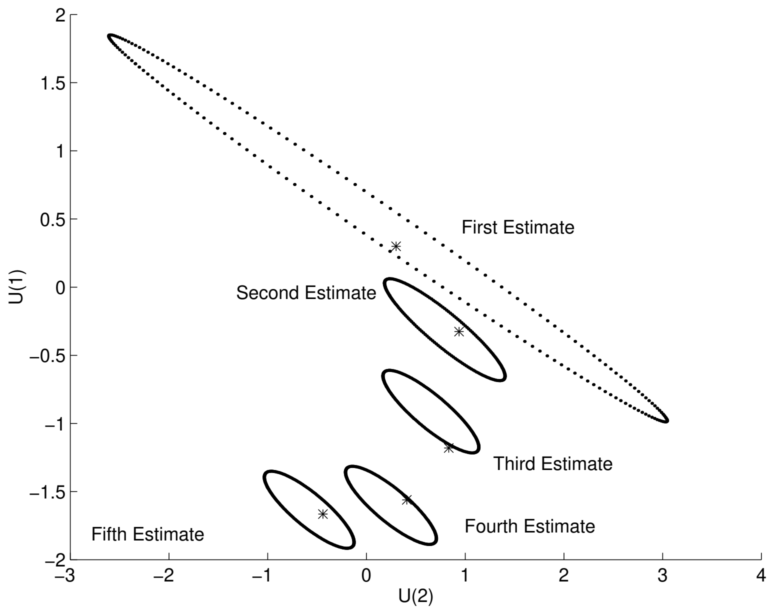

Figure 3.2: Kalman Example with One Standard Deviation Ellipses

decreasing and cause K_k to decrease also.

The covariance matrix provides memory for the process, therefore the Kalman filter converges very quickly to the state variable vector to be tracked. Figure 3.2 illustrates the effect measurement has on the covariance for the first five iterations of the filtering process. The * represent the actual state variables. The ellipses are centered around the Kalman estimate of the state variables. The ellipse forms the one standard deviation bound for the estimate of the state variables. It should be noted that the size of the ellipses are quickly limited by the noise in the process (R_k, Q_k).

CHAPTER 4

Using the Kalman Filter To Compensate ADCs

Chapter 3 discussed the Kalman filter and how it is used to track state variables. This chapter will discuss how to apply the Kalman filter to an ADC and the models used to do this.

4.1 Kalman Equations to Compensate ADCs

To apply the Kalman filter, it is necessary to adapt the Kalman state model to a form that is useful for ADC error modeling. The linear state model is given in (3.1) and (3.2) and is restated here.

$$\vec{u}_{k+1} = \phi_k \vec{u}_k + \vec{w}_k \tag{4.1}$$

$$\vec{z}_k = B_k \vec{u}_k + \vec{v}_k \tag{4.2}$$

As discussed in Chapter 2, the ADC error will be modeled independently for each state as a function of slope. The error for each state will be modeled as slope dependent basis functions with corresponding coefficients which will be found by the calibration method.

$$e_k^{(i)}(y) = \sum_{l=0}^{m} \alpha_{k,l}^{(i)} b_l(y) \tag{4.3}$$

To compensate the ADC, a Kalman filter is used to track the basis function coefficients $\alpha_{k,l}^{(i)}$ for each state. For example an 8 bit converter has 256 independent Kalman filters, one for each ADC output code. To track drifts in the ADC characteristics, a linear model will be used.

The Kalman equations for state i, and measurement k are

$$\vec{\alpha}_{k+1}^{(i)} = \phi\vec{\alpha}_k^{(i)} + \vec{w}_k \tag{4.4}$$

$$e_k^{(i)}(y) = B_k(y)\vec{\alpha}_k^{(i)} + v_k \tag{4.5}$$

where the variables are defined as follows:

$\vec{\alpha}_k^{(i)}$	The basis function coefficients to be tracked for state i at time k: $\vec{\alpha}_k^{(i)} = [\alpha_{k,0}^{(i)} \ \ \alpha_{k,1}^{(i)} \ \ \ldots \ \ \alpha_{k,m-1}^{(i)}]^T$
ϕ	The constant state transition matrix which defines how the state variables are changing with time. ϕ is assumed constant.
$\vec{w}_k$	The random component of the changes in $\vec{\alpha}_{k+1}^{(i)}$ with covariance Q. Q is assumed constant.
$B_k(y)$	The vector of basis functions evaluated at slope y, $B_k(y) = [b_0(y)b_1(y)\ldots b_{m-1}(y)]$.
$e_k^{(i)}(y)$	The average error measured at time k for state i at slope y (a single value).
v_k	The noise in the measurement of the error $e_k^{(i)}(y)$, which is assumed to have time invariant covariance R.

These Kalman equations describe how the basis function coefficients are evolving with time, and how they are related to estimated ADC error. For example, the selection of various parameters in (4.4) and (4.5) depends upon the environment the converter is in. If environmental factors such as temperature and power supply drift are changing quickly, the parameters of (4.4) and (4.5) will be different than if the ADC environment is very stable.

Note that the Kalman filter for each state has the same dynamic description, (same $\phi, B_k(y), Q, R$) and only the function coefficients $\vec{\alpha}_k^{(i)}$ are different. That is, each state is assumed to be described by a different time-varying equation, but the statistical drift properties for all states are assumed identical.

4.1.1 Selection of Parameters

The selection of parameters depends upon the system containing the ADC. This section gives a brief discussion on choosing good parameters, based on the rate the environment is changing.

4.1.1.1 System Equation

The ϕ matrix, and the covariance matrix, Q, of the process disturbance, $\vec{w}_k$ in (4.4), determines how the basis function coefficients are evolving from the k^{th} measurement to the $(k+1)^{th}$ measurement. The basis function coefficients will be assumed to be changing at the same rate, and independently. This assumption yields a phi matrix given by (4.6), where c is a constant.

$$\phi = cI \tag{4.6}$$

With this equation and assuming c to be less than one (giving a stable system), the adaptation rate can be controlled by choosing a "time constant", τ. The time constant is the number of measurements made before the current vector $\vec{\alpha}_k^{(i)}$ would decay to 1/2 its initial value, with $\vec{w}_k = \vec{0}$. Using τ, the value of c is found using (4.7).

$$c^{\tau} = 0.5 \tag{4.7}$$

The ϕ matrix can be an identity matrix, by choosing a time constant of infinity. This implies the $\vec{\alpha}_k^{(i)}$ are slowly changing as a result only of the random process disturbance $\vec{w}_k$. The filter will tend to not forget previous values of $\vec{\alpha}_k^{(i)}$.

With the system equation given in (4.1), and ϕ given by (4.6), the covariance of the white noise $\vec{w}_k$ is related to the covariance of the state variables by (4.8) [12].

$$\Sigma_\alpha = (I - \phi * \phi^T)^{-1} Q \tag{4.8}$$

If the basis functions are normalized to one, then setting the variance of the state variables to one will result in ADC error characteristics with a few LSBs of error (assuming a normal distribution of error). Assuming the $\vec{\alpha}_k^{(i)}$ are independent with variance of one, $\Sigma_\alpha = I$, and using (4.6) yields a covariance for w_k of

$$Q = (1 - c^2) I \tag{4.9}$$

Other choices for the Q matrix may be used to alter the behavior of the Kalman filter. Increasing the coefficient on the Identity matrix of (4.9) will make the algorithm more willing to track large changes in ADC error (at the expense of larger steady state error).

If Q is set to zero, it implies that the coefficients, in (4.4), have no changes being made to them, other than the changes made by the ϕ matrix. For ADC compensation this would reduce the ability of the algorithm to track deviations in error. This also introduces instability into the system. When the value of c is less than one, and Q is set to zero, the basis function coefficients iterate to zero and measurements,$e_k^{(i)}(y)$ will be ignored as the covariance, P_k, becomes small (because $Q = 0$). Consequently, no new measurements will be considered in estimating the $\vec{\alpha}_k^{(i)}$. This causes the estimate of the basis coefficients to drift away from the coefficients which minimize the mean square error.

4.1.1.2 Measurement Equation

For state i, each iteration of the filter requires a measurement of the slope dependent error $e_k^{(i)}(y)$. This error may be measured by histogram methods presented by Larrabee or harmonic measurements as discussed in Chapter 2. If an N point DFT of the output samples of the ADC, with a dithered input signal, is taken, the error can be measured by taking Nh harmonics from the DFT and taking the IDFT of those harmonics (and their conjugates).

The constant R, the variance of v_k, depends directly upon the measurement accuracy. For measurements obtained from DFTs, with the input dithered, the value is determined by (4.10).

$$R = \frac{1}{3}\frac{Nh}{N} \quad LSBs \tag{4.10}$$

where Nh is the number of harmonics to be compensated or measured, and N is the DFT length.

As defined earlier in this chapter, $B_k(y)$ is a vector of the basis functions evaluated at the slope y.

$$B_k(y) = [b_0(y)\ b_1(y)\ \ldots\ b_{m-1}(y)] \tag{4.11}$$

4.1.1.3 Initial Covariance and Alpha

To begin the Kalman filter calculations it is necessary to initialize the $\vec{\alpha}_0^{(i)}$ and the P_0^-. The initial $\vec{\alpha}_k^{(i)}$ may be obtained from a table calibrated off line, or as an initial condition, $\vec{\alpha}_k^{(i)}$ may be set to $\vec{0}$. This will reflect no error for the ADC. For most of the tests in this thesis, $\vec{0}$ is used as the initial condition. With a weight

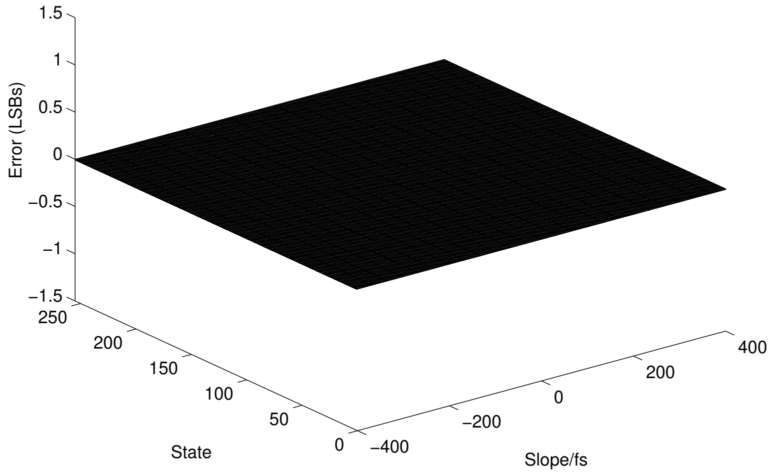

Figure 4.1: Error Table with $\vec{\hat{\alpha}}_0^- = 0$

of 0 on all basis functions, the error table has a value of 0 for all states and slopes, as is shown in Figure (4.1).

$$\vec{\hat{\alpha}}_0^{(i)-} = \vec{0} \tag{4.12}$$

To initialize the equations we define the initial covariance estimate to be large. The error for an ADC is expected to be around ± 1 LSB (Least Significant Bit). Assuming the error to be Gaussian with a standard deviation of one LSB, the covariance will be set to 3 LSBs.

$$P_0^- = 3I \tag{4.13}$$

4.2 Updating the Error Estimate

With the initial conditions set, a sample set of test data is acquired. The converter is driven with a sinusoid, which is conditioned to be spectrally pure. N sequential samples at the converter output are stored, and the error due to harmonic distortion is found for each state. It has been shown that there are different errors for the same state at positive and negative slope values. For this reason, an expected error is found for each state at positive and negative slopes. Two iterations of the Kalman filter is performed for each state. The positive slope expected error for state i is the system output used to update the basis function coefficients, $\vec{\alpha}_k^{(i)}$, in (4.14).

$$e_k^{(i)}(y) = B_k(y)\vec{\alpha}_k^{(i)} + v_k \tag{4.14}$$

The basis function coefficients are updated with (4.15).

$$\hat{\vec{\alpha}}_k^{(i)} = \hat{\vec{\alpha}}_k^{(i)-} + K_k(e_k^{(i)}(y) - B_k(y)\hat{\vec{\alpha}}_k^{(i)-}) \tag{4.15}$$

$B_k(y)$ is the vector of basis functions evaluated at slope y. The Kalman gain matrix K_k is calculated using (4.16).

$$K_k = P_k^- B_k(y)^T (B_k(y) P_k^- B_k(y)^T + R)^{-1} \tag{4.16}$$

The new covariance of the estimate is calculated from (4.17).

$$P_k = (I - K_k B_k(y)) P_k^- \tag{4.17}$$

This updated $\hat{\vec{\alpha}}_k^{(i)}$ is projected to the next $\hat{\vec{\alpha}}_{k+1}^{(i)-}$ estimate (4.18) using the ϕ

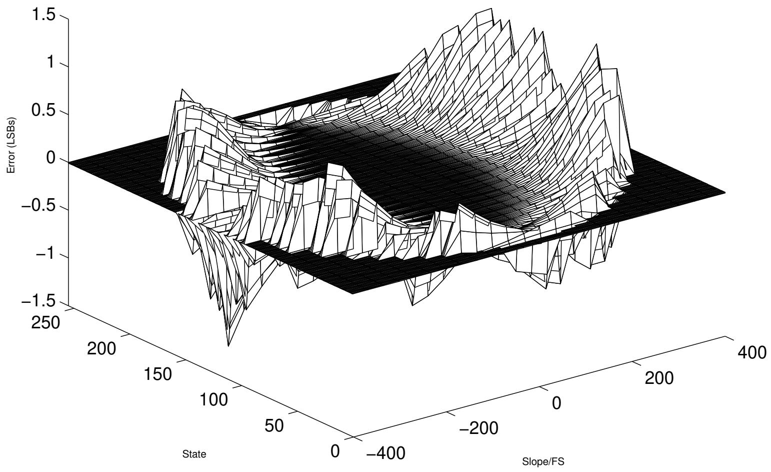

Figure 4.2: Error Table with One Iteration of the Kalman Filter

matrix, and the covariance is also updated (4.19).

$$\vec{\hat{\alpha}}_{k+1}^{(i)-} = \phi \vec{\hat{\alpha}}_{k}^{(i)} \tag{4.18}$$

$$P_{k+1} = \phi P_k \phi^T + Q \tag{4.19}$$

This same procedure is followed for the negative slope error estimate, at each of all possible states. Once the error for all states that exist in the output buffer have been passed to the filter, the table is updated. Figure 4.2 shows a table as a function of state and slope after one complete iteration.

After updating the table, a new set of data is taken and processed similarly to the first iteration. This procedure continues, tracking the state and slope dependent error of the converter.

4.3 Convergence of the Kalman Filter

4.3.1 Choosing Trajectories

For the table illustrated in Figure 4.2, the first trajectory was chosen to be a 95 percent loading sine wave with frequency given by an odd numbered DFT bin just below the sampling frequency (f_s). (For this simulated ADC the sampling frequency was $3GHz$). This is the largest trajectory that the table will compensate. the first trajectory was the same for all tests. The second trajectory was chosen to be a $2.5MHz$ sine wave, the lowest frequency the test bed will allow for real data, with 95 percent loading. This choice causes the first two trajectories to be tangent, and all future trajectories will intersect, or be tangent to one of these two trajectories. Other calibration techniques typically iterate through a number of frequencies, choosing a number of loadings at each frequency. The calibration is done after all data is acquired. For the adaptive method, iterating through the frequencies and loadings, in this manner, tends to weight the table toward the most recent frequency. To combat this problem, after the first two trajectories,which set the calibration bounds , all subsequent frequencies and amplitudes are chosen randomly.

Figure 2.2 shows a state-slope plane which is uniformly filled when trajectories are well chosen. The normalized slope (slope divided by the sampling frequency) varies from -400 to +400 for an eight bit converter. The trajectories were chosen, for updating methods, by randomly (uniformly) selecting a slope value between the maximum normalized slope and the minimum positive slope. The maximum is determined by the maximum frequency at 95 percent loading $2*\pi*128*f_s/2*.95/f_s$. The minimum is determined by the lowest frequency the test bed can support ($2.5MHz$) at a loading of 50 percent, $2*\pi*128*2.5MHz*.50/f_s$. A randomly selected slope, between these limits, is the maximum slope for the next test trajectory. This slope sets limits for the frequencies that can be used when the loading is constrained to be between 50 and 90 percent loading. One of the possible frequencies is then selected using a uniform random variable over this chosen frequency range. Given the slope value, which is to be the maximum, and the test frequency (f_t), the loading can be calculated.

$$loading = \frac{maxslope * f_s}{2 * \pi * 128 * f_t} \tag{4.20}$$

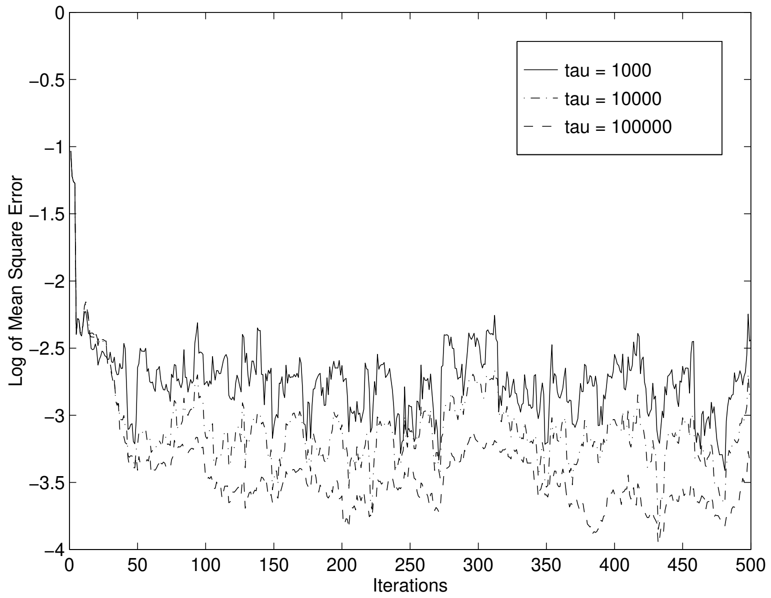

Figure 4.3: Convergence Comparison

4.3.2 Convergence Rates

The Kalman filter converges very quickly, the accuracy of the $\vec{\alpha}_k^{(i)}$ is controlled by the Q matrix. If the diagonal terms of the Q matrix are large, the Kalman filter will have less confidence in the stability of the $\vec{\alpha}_k^{(i)}$, and may make large changes in $\vec{\alpha}_k^{(i)}$. The $\vec{\alpha}_k^{(i)}$ will be less accurate than if the diagonal terms are small. If the diagonal terms are small, the Kalman filter may take more iterations to converge, but the accuracy of $\vec{\alpha}_k^{(i)}$ will be better. Figure 4.3 demonstrates the difference the value of τ makes in convergence and accuracy of the filter.

To generate Figure 4.3, random values between -400 and 400 were generated and used as the argument of a test polynomial, $(0.5+1y^2+0.5y^3)/10^5$, to represent error at some state. The Kalman filter was passed the value of the polynomial plus some random noise. The Kalman matrices were determined using various values of τ in (4.7-4.9). The Kalman filter was used to track eight slope dependent Gaussian basis function coefficients to estimate the polynomial. These results were compared to the known results . The $m =$ eight Gaussian basis functions had means which were evenly spaced from 400 and -400, with standard deviation $\frac{400}{m-1}$. This example is analogous to tracking the slope dependent error for each state of the converter. Figure 4.3 is a plot of the log of the mean square error of the difference between the Gaussian function error estimates and the true error given by the test polynomial. The convergence to a given accuracy, due to different τ values is shown in this figure. This illustrates the importance of choosing a value of τ which is large enough to get the necessary accuracy. In this figure, the filters all converge at roughly the same rate, which is due to the size of the initial covariance matrix. The Kalman filter learns quickly in the initial training steps.

If however the polynomial is changed during the tracking process, the differences in tracking rate can be seen. This is demonstrated in Figure 4.4. When the polynomial is changed after 500 iterations to $(2 + 1y^2 + 2y^3)/10^5$, the Kalman filter is tracking the basis function coefficients, and has a good estimate of them and the covariance matrices are small. This causes the filter to adapt more slowly than if the covariance matrix is large. During the reacquisition time, the filter with the lowest time constant will be the fastest to reacquire.

One can easily see that if τ is chosen to be too small the filter doesn't track the coefficients with as much accuracy as a larger value of τ. If, however the value of τ is chosen to be too large, the filter will not respond to system (or environmental) changes in the error fast enough.

When the Kalman algorithm is implemented on the simulated ADC, with a τ value of 2000, the filter yields results which are comparable to those obtained by Larrabee. Figure (4.5) compares uncompensated results to compensated results obtained using the Kalman filter. The SFDR in the compensated spectrum has been improved by about 15 dB.

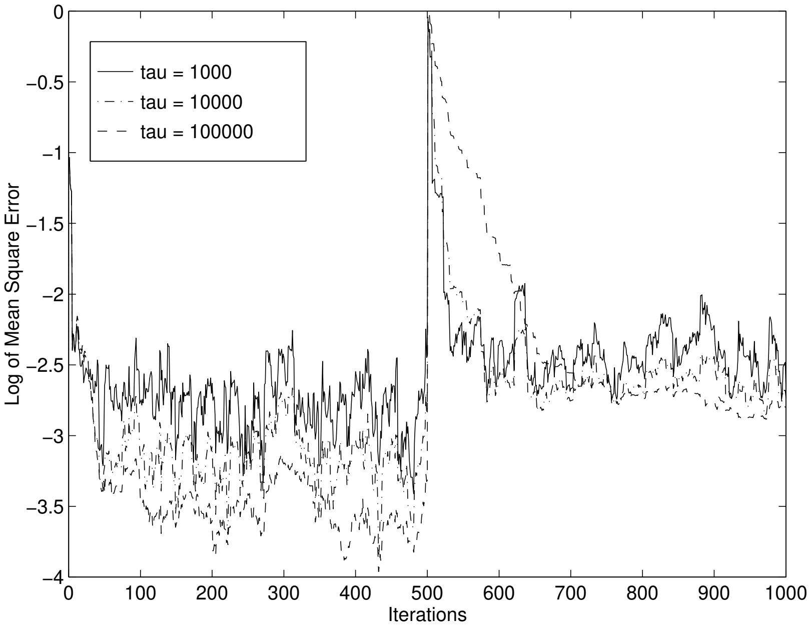

Figure 4.4: Convergence Comparison with Changing Polynomial Coefficients

The Kalman filter converges and tracks deviations quite quickly, however, it is both memory and computationally intensive. Storing and using it's covariance and gain matrices makes the Kalman filter difficult to implement.

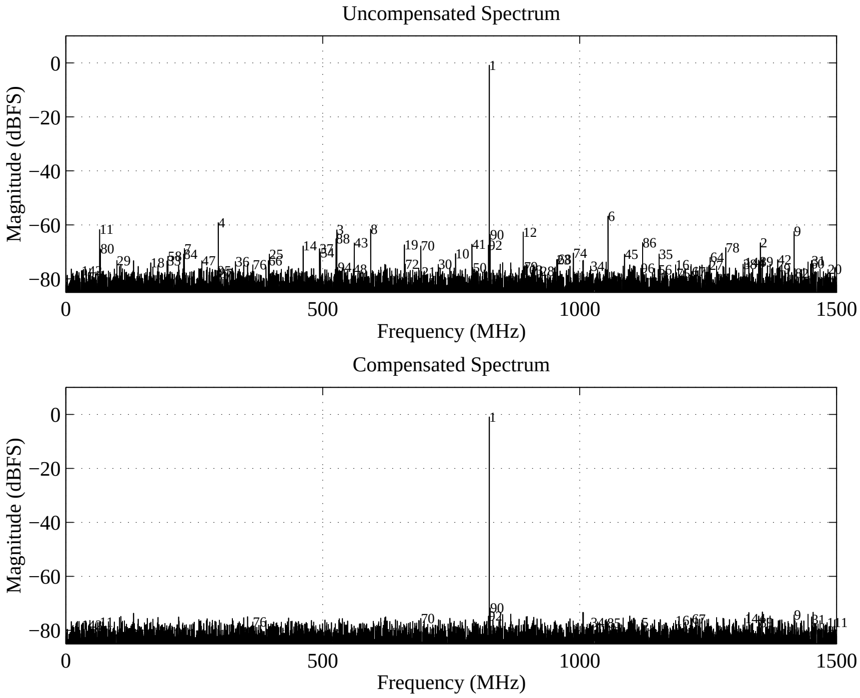

Figure 4.5: Spectral Results from Kalman Algorithm

CHAPTER 5

Least Mean Square Algorithm

Experimental results demonstrate that this Kalman filter formulation is very effective at adaptive ADC compensation, however the Kalman filter is very memory intensive. The Least Mean Square (LMS) algorithm is another adaptive algorithm useful for ADC compensation, which is much less memory intensive, but is much slower to converge.

5.1 LMS Algorithm

The LMS algorithm is often used to implement adaptive tapped delay line filters. The tapped delay line filter block diagram is shown in Figure 5.1. The LMS algorithm is an iterative procedure for minimizing the mean square error of the filter output. More information on the LMS algorithm may be found in [4].

The system inputs, $u(k)$, $u(k-1)$, $\ldots$ $u(k-m)$ are each weighted by time varying coefficients $w_0(k), w_1(k), w_2(k) \ldots w_m(k)$. The sum of the weighted inputs form the estimate, $\hat{d}(k)$, of the desired response, $d(k)$. The error $e(k)$ is used to make changes to the weighting factors according to (5.1), where $\vec{u}_k$ is $[u(k)\ u(k-1)\ u(k-2)\ \ldots\ u(k-m)]^T$.

$$\vec{w}_{k+1} = \vec{w}_k + \mu e(k)\vec{u}_k \tag{5.1}$$

The convergence parameter μ is a constant which determines the amount of change in $\vec{w}_{k+1}$ that each iteration can possibly make. This convergence parameter also determines the accuracy of the algorithm. If some assumptions are made, the accuracy and convergence can be bounded [3]. The necessary assumptions are:

- the changes in $\vec{w}_{k+1}$ are small

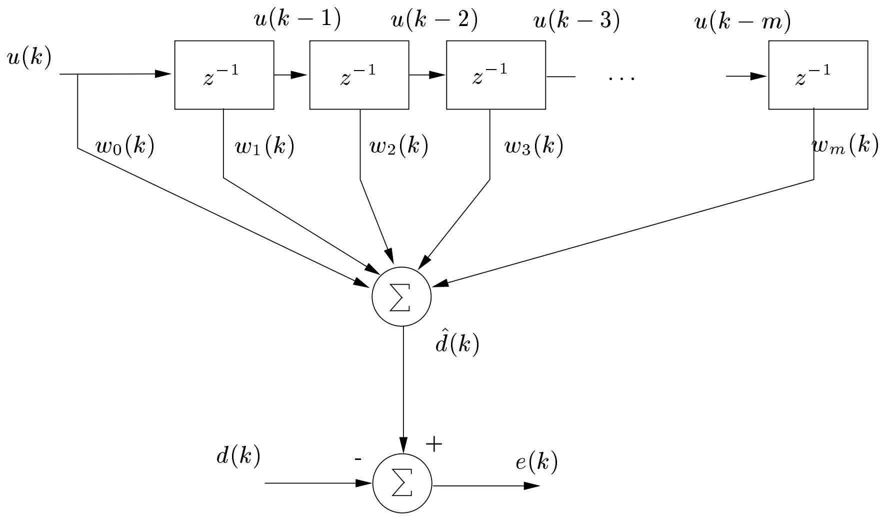

Figure 5.1: Tapped Delay Filter Block Diagram

- $[u(k)\ u(k-1)\ u(k-2)\ \ldots u(k-m)]$ are statistically independent
- $u(k)$ are independent of $d(0), d(1) \ldots d(k-1)$
- $\vec{u}_k$ and $\vec{d}_k$ are jointly Gaussian with variances σ_u^2 and σ_d^2 respectively

With these assumptions, a misadjustment parameter $\mathcal{M}$ may be defined as the ratio of the excess mean square error over the minimum mean square error [3]. The excess mean square error is the actual mean square error minus the minimum mean square error. This misadjustment is dependent upon μ, and can be approximated by

$$\mathcal{M} \approx \frac{\mu m {\sigma_u}^2}{2} \tag{5.2}$$

where m is the number of filter coefficients. In equation (5.2), the misadjustment is proportional to the value of μ. Specifically , there is a linear relationship between

μ and the excess error. The excess mean square error may diverge if μ is chosen to be too large. To assure convergence μ must be between 0 and $\frac{2}{m\sigma_u^2}$, where the variance of $\vec{u}_k$ is σ_u^2.

The convergence parameter also determines τ, the average time constant [3]. This time constant gives an estimate of the convergence rate. Note that the estimate of τ is inversely proportional to μ, therefore a doubling of μ will halve the time constant.

$$\tau \approx \frac{1}{\mu 2 {\sigma_u}^2} \tag{5.3}$$

5.2 Application of the LMS Algorithm to ADCs

Converting the LMS algorithm to a form that is useful for ADC compensation is straightforward. Like the Kalman filter, an LMS filter is applied to each state, i, of the converter. Instead of having a vector $\vec{u}_k$, the input vector is the basis functions, $b(y)$, evaluated at slope y. The $\vec{w}_k$ vector is replaced with the basis function coefficients $\vec{\alpha}_k^{(i)}$. The desired response is the measured expected error $e_k^{(i)}(y)$, and $\epsilon_k^{(i)}$ is the error in the current estimate of the error. These changes lead to the block diagram shown in Figure 5.2. The "training rule" to determine the unknown $\vec{\alpha}_{k+1}^{(i)}$ is described by

$$\vec{\alpha}_{k+1}^{(i)} = \vec{\alpha}_k^{(i)} + \mu \epsilon_k^{(i)} b(\vec{y}) \tag{5.4}$$

5.3 Convergence of the LMS Algorithm

The chosen μ determines the convergence of the algorithm. If μ is chosen to be too large the algorithm will converge quite quickly but will not necessarily

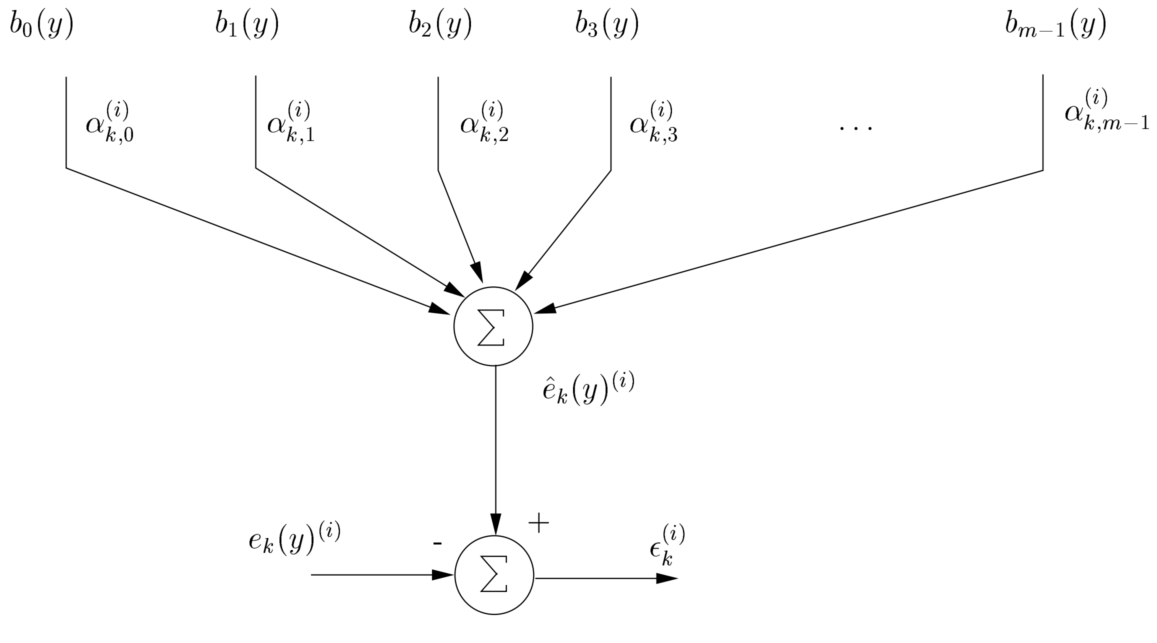

Figure 5.2: Block Diagram of a LMS Algorithm System

minimize the mean square error.

A time constant for the ADC LMS algorithm can be found using (5.3). However in experiments, the observed time constant for training an ADC is different than predicted. This is due to the manner that basis functions are updated. For any single iteration, all but two or three slope dependent basis functions are zero, therefore not all coefficients are trained on each iteration. The time constant is still useful in comparing relative convergence rates. The time constant is still inversely related to the μ value.

$$\tau \approx \frac{1}{\mu 2 {\sigma_\alpha}^2} \tag{5.5}$$

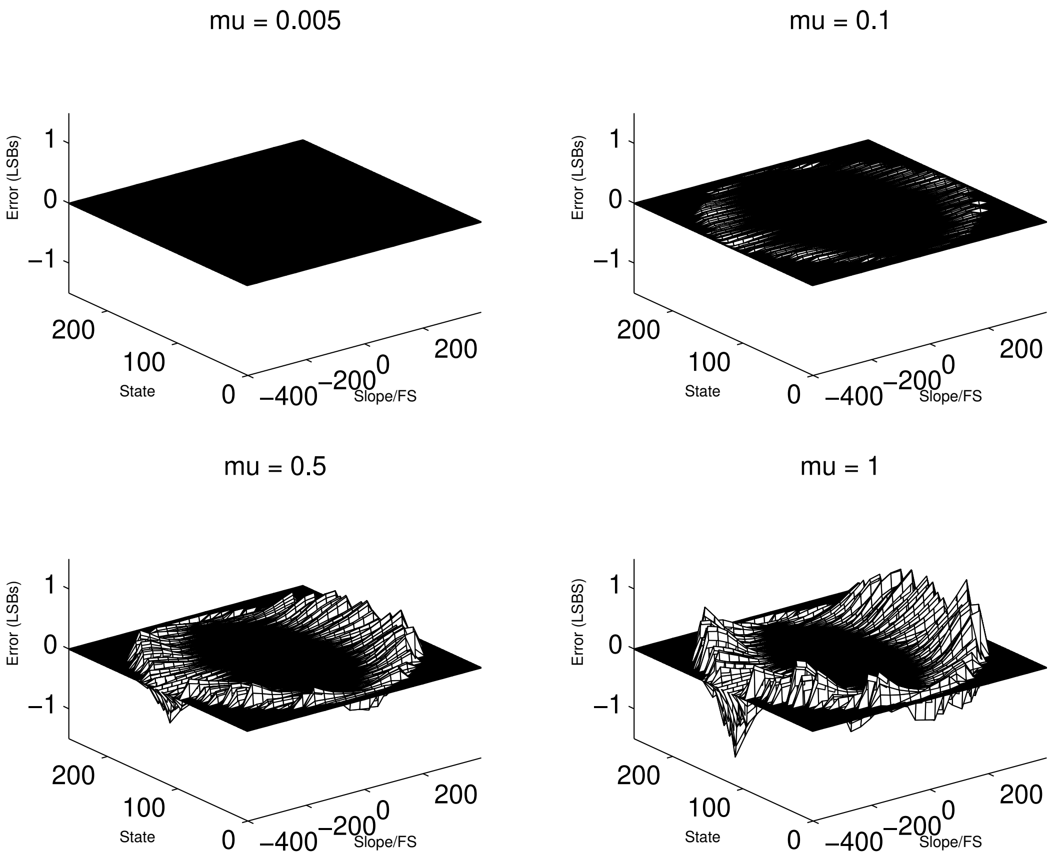

Figure 5.3: Comparison of μ Values After one Iteration

Figure 5.3 shows three tables, with four different μ values, after one iteration of the LMS algorithm. The μ values are clockwise from top left 0.005 ,0.1 ,1 ,0.5. It is obvious that the μ of 1 causes the table to converge faster, however it will be a less accurate table than any of the others. The μ value of 0.005 has made almost no change in the table after one iteration, and will (after many training iterations) be more accurate than the larger μ values. The convergence parameter μ can be set very small, to give an error table which is close to the minimum mean square error, but this table will take a lot of iterations to converge. Figure 5.4 illustrates the convergence rate and accuracy for different μ values. This plot is generated in the same manner as the example at the end of Chapter 4. From the figure, it is easy to see that the smaller μ gives a more accurate estimate, but takes a much

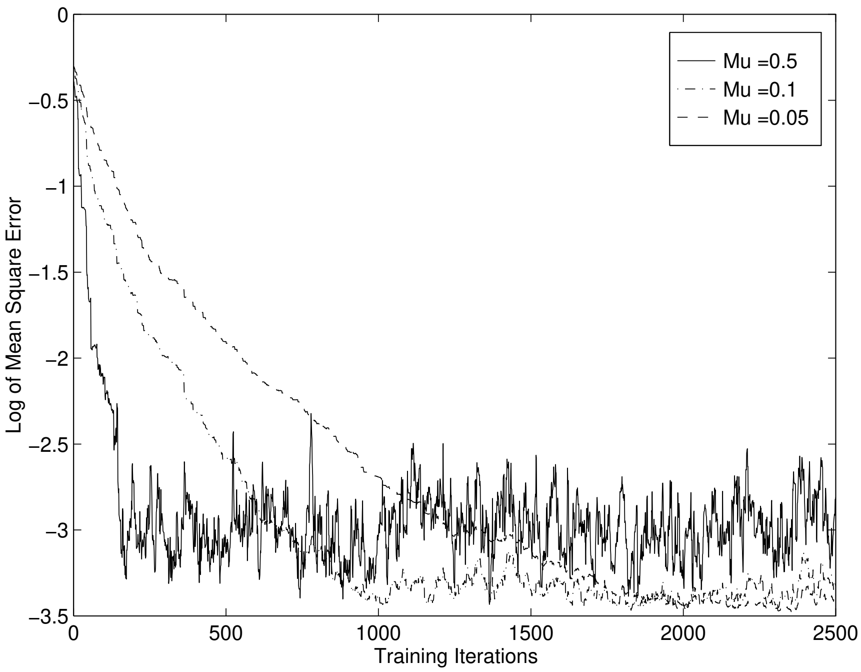

Figure 5.4: Comparison of μ Values

longer time to converge. The opposite is true of the larger μ. Note that the number of iterations is 25 times longer, than the Kalman algorithm, for comparable error.

There is a tradeoff between convergence speed and accuracy, therefore μ is a design parameter and depends upon the adaptive system. The rate at which training data is available will help determine the smallest value of μ that is feasible to use.

CHAPTER 6

Results

6.1 Kalman Filter

To test the Kalman and LMS algorithms, an extensive converter simulation was used. The simulation models a folding and interpolating ADC, with architecture dependent distortions. This architecture is common in high speed ADCs. The output of this simulated converter has state and slope dependent error.

The Kalman filter was used to track the basis coefficients for eight Gaussian basis functions, at each of 256 states, with means evenly spaced from 400 and -400, and standard deviation $\frac{400}{m-1}$, as discussed in Chapter 4. Figure 6.1 shows a number of plots of the spectrum, with 16384 samples, for low, mid and high frequencies, before and after compensation. This Figure also gives a plot of the SFDR vs frequency of the converter before and after compensation. A 2000 iteration time constant was selected and 200 trajectories were used to update the table. The SFDR has been improved by approximately 15 dB for all frequencies and works across the full first Nyquist band. This figure demonstrates that the Kalman filter method is capable of tracking and compensating for ADC error which is a function of only state and slope. The Figure also matches the result found by Larrabee and shows that the Kalman adaptive method can duplicate analytical non-iterative procedures.

To test the compensation on a real converter, the Kalman algorithm was implemented on an eight bit 250 MHz ADC. The actual clock frequency of the converter was chosen to be 204.8 MHz. This ADC was excited by sinusoids using test frequencies between 2.5 to 102.4 MHz. The Kalman method was used to build a compensation table. The results for three test frequencies are shown in Figure

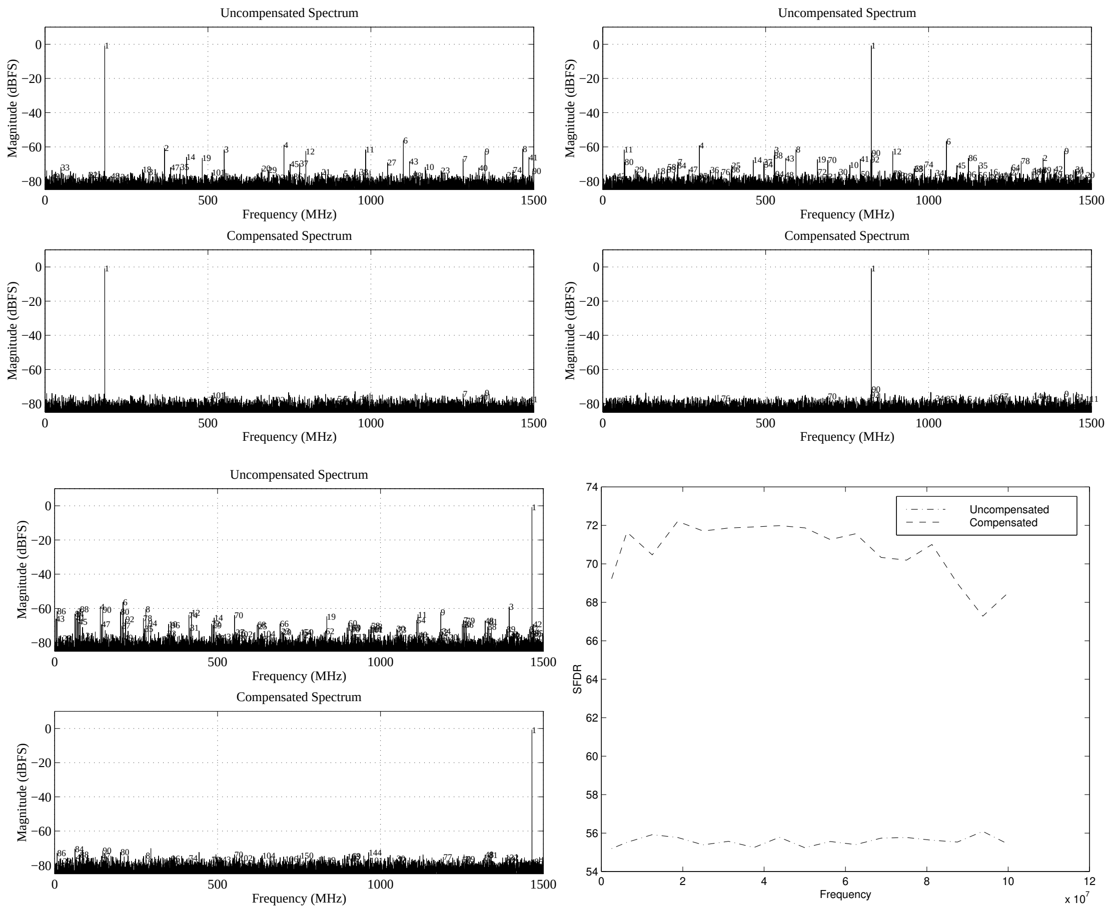

Figure 6.1: Kalman Algorithm Results for a Simulated Converter

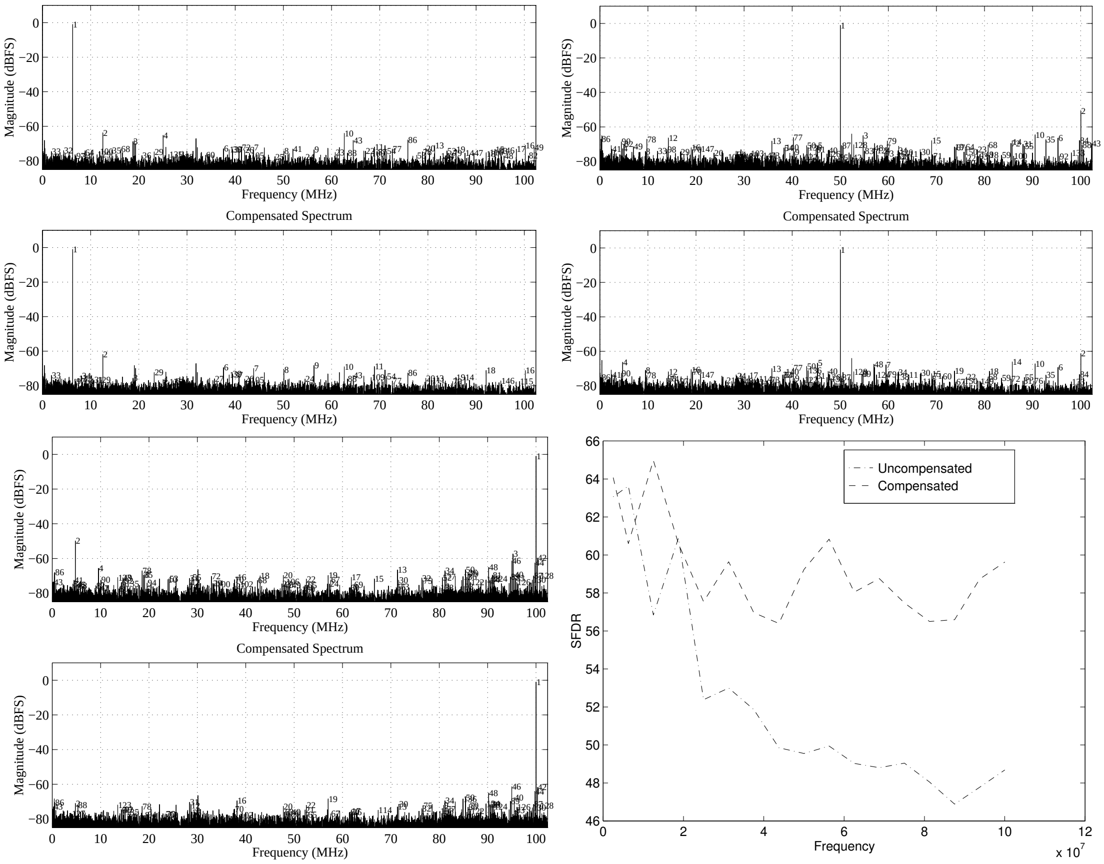

Figure 6.2: Kalman Algorithm Results for a Real Converter

6.2, as well as the SFDR vs frequency, before and after compensation.

The results for this converter match the results obtained by Larrabee [10] at higher frequencies but the adaptive result is less accurate at lower frequencies. This was found to be due to the method used to select the test frequencies. The method used tends to select higher test frequencies, therefore there are less low frequency calibration signals than high frequency ones. This causes the table to be weighted toward high frequencies. When the frequencies are uniformly chosen, the table tends to be weighted toward the center of the state-slope plane.

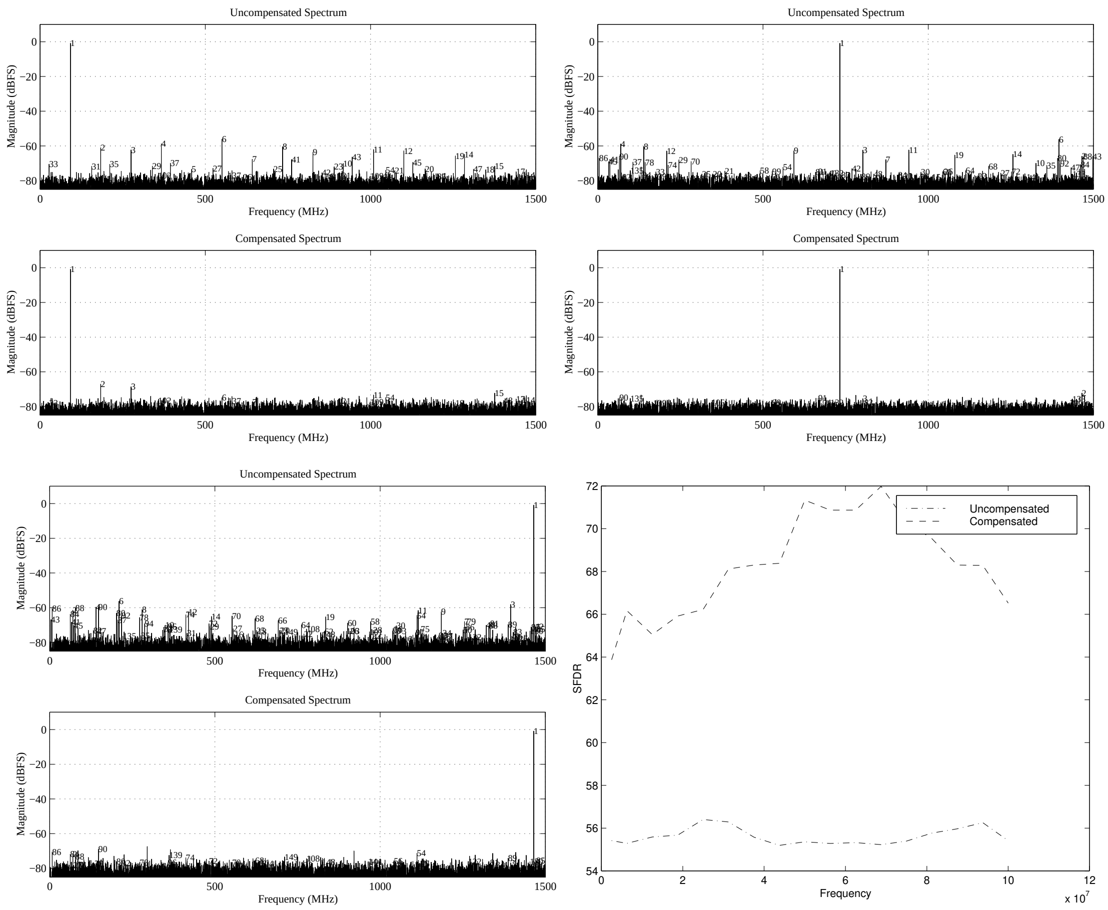

Figure 6.3: LMS Algorithm Results for a Simulated Converter

6.2 LMS Algorithm

The LMS algorithm was also used on the simulated converter. In order to obtain comparable SFDR results to the Kalman algorithm, a μ value of 0.05 was used. Figure 5.4 shows that a mu of 0.05 takes approximately 2000 iterations to converge to the minimum mean square error from an initial estimate of $\vec{0}$. To speed the initial convergence, a table was created with a larger μ, then 300 trajectories were used to update the error table at a μ of 0.05. As mentioned, the results shown in Figure 6.3, are comparable to the Kalman results, but even with a μ of 0.05 the results aren't as good for low frequencies as the simulated Kalman results.

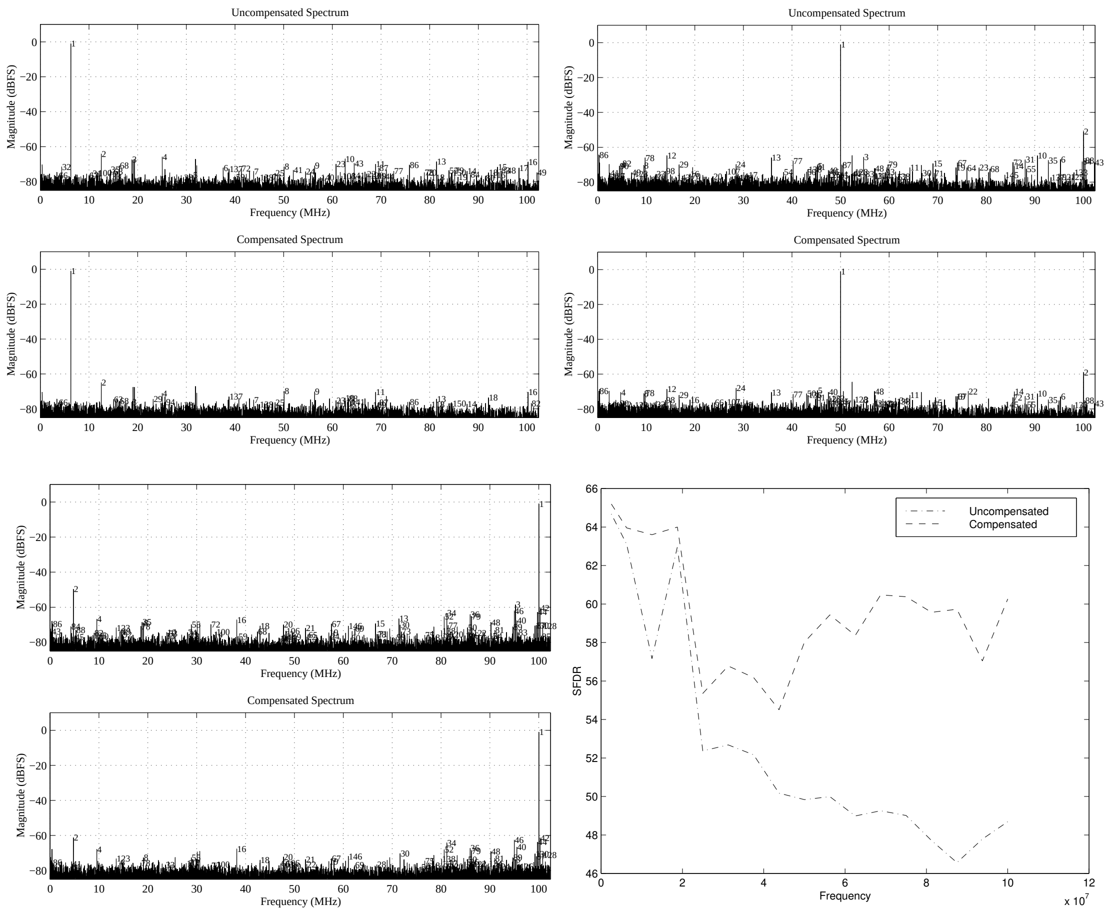

Figure 6.4: LMS Algorithm Results for a Real Converter

These results improve with a decreased μ, and many more trajectories (iterations) at lower frequencies.

The testbed was used to test the LMS algorithm on the actual converter, using a μ of 0.05. Two hundred trajectories were used to update the error estimate. The results are shown in Figure 6.4. Again the results aren't as good at lower frequencies as those presented by Larrabee, but high frequency results are comparable to his and confirm the method is a broadband algorithm.

6.3 Conclusions

The Kalman filter converges very quickly and by controlling the value of the diagonal terms of the Q matrix, the accuracy and adaptive rate can be controlled. The Kalman filter converges very quickly from an error table where all of the starting (or initial) basis function coefficients are zero, due to the P matrices being large.

The Kalman filter is however very memory intensive. For each of 256 states for an eight bit converter, the P ,P_m, and K matrices must be stored. In a system where memory is a concern, the Kalman filter will not be a first choice. The LMS algorithm, however is not very memory intensive. The LMS algorithm doesn't store any of the covariances that the Kalman filter stores. This causes the algorithm to converge more slowly, but it is easier to implement.

The value of μ is critical to the LMS algorithm. If μ is chosen to be too large the SFDR will be limited. The size of μ directly determines the error, and therefore the SFDR. Conversely, if μ is too small, the algorithm will not adapt as fast as necessary. The system to be compensated will determine the rate of error drift. If the error is slowly changing, a small μ value will work well. If however, the error is a quickly changing phenomenon a larger μ will be required to track such changes.

The adaptive methods presented enable a system to be continually updated, maintaining the compensation's effectiveness over time. Analytic methods, however, need to be redone completely to change compensation values. By choosing large time constants, the adaptive algorithm could be used to develop a steady state table with accuracy comparable to Larrabee's. In order to enable the system to track deviations, the performance, compared to Larrabee's results, is degraded.

References

[1] W. Ahmed. Fast orthogonal search for training radial basis function neural networks. Master's thesis, University of Maine, Dept. of Electrical and Computer Eng., Orono Maine, August 1994.

[2] R. Brown and P. Hwang. *Introduction to Random Signals and Applied Kalman Filtering.* Wiley, New York, New York, 1997.

[3] S. Haykin. *Modern Filters.* Macmillan, New York, New York, 1989.

[4] S. Haykin. *Adaptive Filter Theory.* Macmillan, New York, New York, 1996.

[5] D.M. Hummels, F.H. Irons, R. Cook, and I. Papantonopoulos. Characterizaion of ADCs using a non-iterative procedure. In *Proceedings of IEEE International Symp. on Circuits and Systems*, volume 1, pages 5–8, London, May 1994.

[6] D.M Hummels, J.J. McDonald, and F.H. Irons. Distortion compensation for time-interleaved ADCs. In *Proceedings of IEEE International Instumentation and Measurement Technology Conference*, volume 1, pages 728–731, Brussels, June 1996.

[7] F.H. Irons, D.M. Hummels, and S.P. Kennedy. Improved error compensation for analog-to-digital converters. *IEEE Trans. Circuits and Systems*, 38:958–961, June 1991.

[8] F.H. Irons, D.M. Hummels, I.N. Papantonopoulos, and C.A. Zoldi. ADC error diagnosis. In *Proceedings of IEEE International Instumentation and Measurement Technology Conference*, volume 1, pages 732–737, Brussels, June 1996.

[9] S.P. Kennedy. Improved ADC phase plane error compensation. Master's thesis, University of Maine, Dept. of Electrical and Computer Eng., Orono Maine, May 1992.

[10] J. Larrabee. ADC compensation using a sinewave histogram method. Master's thesis, University of Maine, Dept. of Electrical and Computer Eng., Orono Maine, August 1997.

[11] I.N. Papantonopoulos. Error modeling for folding and interpolating analog to digital converters. Master's thesis, University of Maine, Dept. of Electrical and Computer Eng., Orono Maine, August 1995.

[12] A. Papoulis. *Probability, Random Variables, and Stochastic Processes.* MacGraw Hill, New York, New York, 1991.

[13] T.A. Rebold and F.H. Irons. A phase plane approach to the compensation of high speed analog to digital converters. In *Proceedings of IEEE International Symp. on Circuits and Systems*, Chicago, May 1987.

Biography of the Author

James J. McDonald II was born in Farmington, Maine on September 28, 1970. He received his high school education from Dirigo High School in Dixfield, Maine in 1988.

He entered the University of Maine in 1992 and obtained his Bachelor of Science degree in Electrical Engineering in 1996.

In May 1996, he was enrolled for graduate study in Electrical Engineering at the University of Maine and served as Research Assistant . His current research interests include communications and circuit design. He is a member of IEEE, Tau Beta Pi, Eta Kappa Nu, and Phi Kappa Phi, and his interests include white water canoing, woodworking, and his family.

He is a candidate for the Master of Science degree in Electrical Engineering from the University of Maine in August 1997.