DIGITAL ALGORITHMS FOR ANALOG ADAPTIVE FILTERS

by

Anthony Chan Carusone

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Electrical and Computer Engineering
University of Toronto

DIGITAL ALGORITHMS FOR ANALOG ADAPTIVE FILTERS

Anthony Chan Carusone

Degree of Doctor of Philosophy
Graduate Department of Electrical and Computer Engineering
University of Toronto
2002

Abstract

The use of analog adaptive filters in modern integrated systems is limited by the complexity of the analog adaptation hardware and by dc offset effects which limit the adaptation accuracy. Both problems can be addressed by using an analog filter with a digital adaptation algorithm.

The design of digitally programmable analog filters suitable for adaptive applications is examined. Novel Gm-C circuits are described and implemented in a CMOS prototype 5th order integrated filter with digitally programmable poles and zeros. However, the greatest challenge associated with performing digital adaptation of an analog filter is obtaining the gradient information without overcomplicating the analog design. Three main approaches to overcoming this challenge are described. First, the gradient information is obtained by correlating changes in the output squared error to independent dither on the adapted parameters. Second, the internal state signals (and, hence, the gradient signals) are calculated from a time delayed input vector using a co-ordinate transformation. Third, time delayed estimates of the filter input are obtained digitally from the filter output and used to calculate the required gradient signals. All three techniques use digital

signal processing to obtain the gradient information and require little, if any, additional analog hardware. The performance of the new adaptive algorithms are discussed and several variations are proposed to simplify integrated implementations. The prototype integrated analog filter was used as a testbed to verify two of the novel algorithms. Gradient descent optimization of analog filter parameters was successfully performed without access to any of the filter's internal state signals, which was not previously possible. As a result, designers of analog adaptive filters are now free to perform the filter design without being restricted by the requirements of the adaptation algorithm.

Acknowledgements

First and foremost, I would like to thank my supervisor and friend Prof. David A. Johns for providing an interesting project, constant direction, valuable advice, and most of all, for providing me with opportunity. He has my appreciation and tremendous respect.

Thanks, also, to my supervisory committee, Prof. Ken Martin and Prof. Bruce A. Francis, and to the external examiner Dr. Ayal Shoval for their time and effort. Their efforts have substantially improved the quality of this thesis.

Thanks to all of my peers in the electronics research group. It is not without considerable regret that I refrain from attempting to list them all; however, there have been so many people who have contributed in so many different ways that a list somehow seems inappropriate. I would also like to gratefully acknowledge the support of NSERC, CMC, and Micronet.

Great personal thanks goes to my family and friends who have provided me with much needed and appreciated support and guidance. Finally, I would like to thank my wife, Soo. She is my partner in everything, and this dissertation is no exception.

Contents

List of Tables

List of Figures

List of Abbreviations

A / D	Analog to Digital
ALC	Adaptive Linear Combiner
BIBO	Bounded-Input Bounded-Output
BiCMOS	Bipolar Complementary Metal Oxide Semiconductor
CMFB	Common Mode Feedback
CMOS	Complementary Metal Oxide Semiconductor
D / A	Digital to Analog
DAC	Digital to Analog Converter
DLS	Dithered Linear Search
DSD	Differential Steepest Descent
DSP	Digital Signal Processor
FIR	Finite Impulse Response
GPIB	General Purpose Interface Bus
IC	Integrated Circuit
IIR	Infinite Impulse Response
LMS	Least Mean Square
LMS-CT	Least Mean Square with a Co-ordinate Transform
LMS-ICT	Least Mean Square with an Inverse Co-ordinate Transform
LSB	Least Significant Bit
MDAC	Multiplying Digital to Analog Converter
MOS	Metal Oxide Semiconductor
MSE	Mean Squared Error
NMOS	N-type Metal Oxide Semiconductor
PAM	Pulse Amplitude Modulation
PC	Personal Computer
PCB	Printed Circuit Board
PMOS	P-type Metal Oxide Semiconductor

PRBS	Pseudorandom Binary Sequence
RAM	Random Access Memory
ROC	Region Of Convergence
SD-LMS	Sign-Data Least Mean Square
SE-LMS	Sign-Error Least Mean Square
SFDR	Spurious-Free Dynamic Range
SONET	Synchronous Optical Network
SS-LMS	Sign-Sign Least Mean Square
SS-LMS-CT	Sign-Sign Least Mean Square with a Co-ordinate Transform
SS-LMS-ICT	Sign-Sign Least Mean Square with an Inverse Co-ordinate Transform
THD	Total Harmonic Distortion
UIO	Unknown Input Observation
VLSI	Very Large Scale Integrated circuit

Chapter 1

Introduction

1.1 Motivation

Filters are general signal processing blocks used in virtually every modern electronic system. Whenever a filter's parameters must track poorly controlled or time varying conditions, adaptive filters are an attractive option. At low speeds, adaptive filtering is easily and efficiently performed using digital circuits. Presently, the vast majority of adaptive filters are implemented digitally and a wealth of literature has been published on the topic [1]. On the other hand, analog filters are preferable at high speeds when low power consumption, small integrated area, and moderate linearity are required. As digital logic continues to shrink and increase in speed, the minimum speed at which analog signal processing becomes beneficial increases. Therefore, this work focuses on high-speed applications where analog adaptive filters will continue to be an important part of systems for years to come.

The most popular adaptive algorithm for high-speed integrated filters today is the LMS algorithm, due primarily to its straightforward and robust digital hardware implementation. However, in analog adaptive filters, implementation of the LMS algorithm is neither straightforward nor robust. The hardware required to generate gradient information is cumbersome and power-hungry [2]. The LMS algorithm's accuracy is also hindered by the presence of dc offsets on the state and error signals [3]. These considerations are the primary factors which presently limit the use of analog adaptive fil-

ters. This thesis seeks to combine the advantages of digital adaptation algorithms with a high-speed analog signal path. The aim is to perform digital adaptation of analog filters. In the next section, several potential applications for digitally adaptive analog filters are presented. All are mixed-signal digital communications systems often including considerable dedicated digital signal processing hardware, usually integrated on the same die as the analog front end. In such systems analog circuit design is particularly challenging, so it is especially desirable to implement the adaptation algorithm digitally.

1.2 Applications and the State of the Art

An adaptive equalizer for digital communications was first proposed by Lucky in 1965 [4]. Since then, data rates over digital communication channels have increased by several orders of magnitude. Although the adaptive functions in some digital communications applications can be efficiently performed digitally, analog adaptive filters play a critical role when high speed and low power are required, as in the applications discussed below. In digital magnetic storage read channels and ethernet receivers, analog adaptive filters (usually digitally programmable) have already been used in practical systems. High speed serial links represent a burgeoning application for analog adaptive filters, and speculative research has begun on analog adaptive filters for optical and smart antenna applications.

1.2.1 Digital Magnetic Storage

Digital magnetic storage channels emerged as the primary application area for analog adaptive filters in the 1990's. The signals received from the read head in a magnetic storage channel are baseband pulses for which some forward equalization is required prior to detection. Adaptive equalization is desired because the characteristics of the read signal will depend upon the particular zone of the magnetic medium being accessed. High bandwidth in the analog front end is desirable to enable high storage densities and fast access times. The adaptive filter should also have a small integrated area and consume little power to facilitate the implementation of an entire read channel on a single chip.

Fortunately, only moderate linearity is required (approximately 40 dB) to obtain satisfactory bit-error-rates.

Fig. 1.1 shows three possible architectures for a digital magnetic storage read channel. The adaptive equalization can be implemented using either digital (Fig. 1.1A), analog discrete time (Fig. 1.1B), or analog continuous time (Fig. 1.1C) filters. In all-digital systems (Fig. 1.1A) some partial equalization is still performed in the analog domain in combination with the fixed lowpass anti-aliasing filter [5]. This is done to reduce the dynamic range and resolution required in the A/D converter and to shorten the length of the digital equalizer required. By making the analog filter adaptive, the digital circuitry and A/D converter complexity are reduced. In some systems, this approach is combined with an analog Viterbi detector to eliminate the A/D converter all together [6], [7].

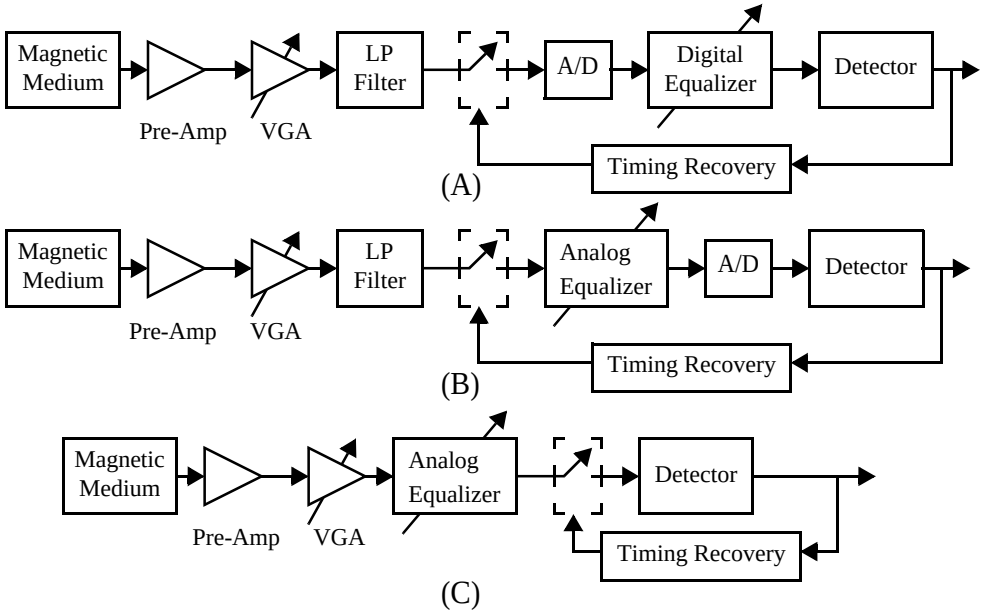


Figure 1.1 System architectures for digital magnetic recording read channels.
The adaptive feedforward equalizer can be realized using (A) *digital,* (B) *discrete time analog, or* (C) *continuous time analog circuitry.*

Analog discrete time transversal filters with 5 to 10 taps are common in current commercial systems. The delay lines are generally implemented using S/Hs. However, in [8] a cascade of continuous time Bessel allpass filters implemented with MOSFET-C circuits

was used. An indirect tuning scheme locked the delay time of each MOSFET-C circuit to the system's sampling frequency.

Tap weighting in the transversal filters can be implemented using either switched-capacitor MDACs [9], programmable transconductors [10], or, in a BiCMOS process, Gilbert multipliers [11]. If either of the latter two techniques is used, a current output is obtained and the output summer is easily realized by connecting together all output nodes. In general, digitally programmable tap weights are useful since the optimal filter coefficients for each zone of the magnetic medium can be stored in a RAM. The stored values are then used to initialize the adaptation algorithm at the start of each read operation, thereby ensuring rapid convergence. In this case, the adaptation algorithm must be implemented digitally. The SS-LMS [11], [12] or SD-LMS [13] algorithms are popular because of their simple hardware implementations.

More recently, continuous time adaptive analog equalizers have been examined for the magnetic storage channel. These offer several distinct advantages over both digital and discrete time analog adaptive equalizers. First, since a continuous time lowpass anti-aliasing filter is required prior to sampling when digital or discrete time filters are used, it would save power and area if the equalization and anti-aliasing functions could be combined and implemented in a single circuit. Second, a continuous time IIR filter with just a few adapted parameters can provide performance comparable to that of a higher order FIR filter. Third, when the equalizer is inside the system's timing recovery loop, as is the case with a discrete time or digital equalizer, the delay around the loop can cause slow convergence at start-up. A 7th order adaptive continuous time equalizer with 4 adapted zeros is described in [14]. Its performance is better than a fixed 7th order continuous time filter combined with a 9-tap adaptive FIR filter [15]. In [16], a continuous time 7th order orthonormal ladder filter implemented in a CMOS process using a G_m-C topology serves as both a lowpass anti-aliasing filter and an adaptive equalizer. Two parameters are adapted using digital SD-LMS circuitry. Although only one zero is adapted, the performance is comparable to systems with 5-tap adaptive FIR filters.

Based on the preceding discussion, it should be clear that continuous time filters with digitally programmable parameters and a digital adaptation algorithm are very useful for digital magnetic storage applications.

1.2.2 Ethernet Over Copper

Analog adaptive equalizers offer essentially the same advantages in ethernet receivers as they do in magnetic storage applications: smaller circuit area and power consumption at high speeds, largely due to the reduced A/D converter specifications. Again, analog continuous time equalization can also eliminate the start-up problems associated with having an adaptive equalizer inside of a timing recovery loop. However, a key difference between ethernet and magnetic storage applications is that the channel's impulse responses may be very long, so a transversal filter of great length would be required to perform equalization. Therefore, IIR adaptive filters may be preferred over FIR transversal structures. Several examples of continuous time analog adaptive equalizers for 100 Mb/s ethernet integrated transceivers have been reported [17], [18], [19]. Analog adaptive equalizers for next-generation gigabit ethernet over copper are also under research [20]. In all but [17], the equalizers are digitally adapted.

Analog adaptive filters can also be used for echo cancellation in full-duplex systems. Fig. 1.2 shows a full-duplex system with (A) digital, (B) mixed signal, and (C) analog adaptive echo cancellation. Note that the entire echo path in Fig. 1.2A including the transmit D/A, line driver, receive filter, and A/D must be highly linear to allow for linear echo cancellation in the digital domain. The analog adaptive echo canceller eases the D/A, line driver, and A/D specifications. The mixed signal equalizer offers a compromise where only the A/D converter specifications are relaxed. Historically, the advantages of analog echo cancellation have been particularly significant in applications using standard telephone lines where the echo signal can be as much as 30 dB louder than the far-end signal. (e.g. voiceband modems [21], ISDN [22], [23], and digital subscriber lines [24]) However, more recently a discrete time mixed signal echo canceller has been applied to gigabit ethernet over copper [25].

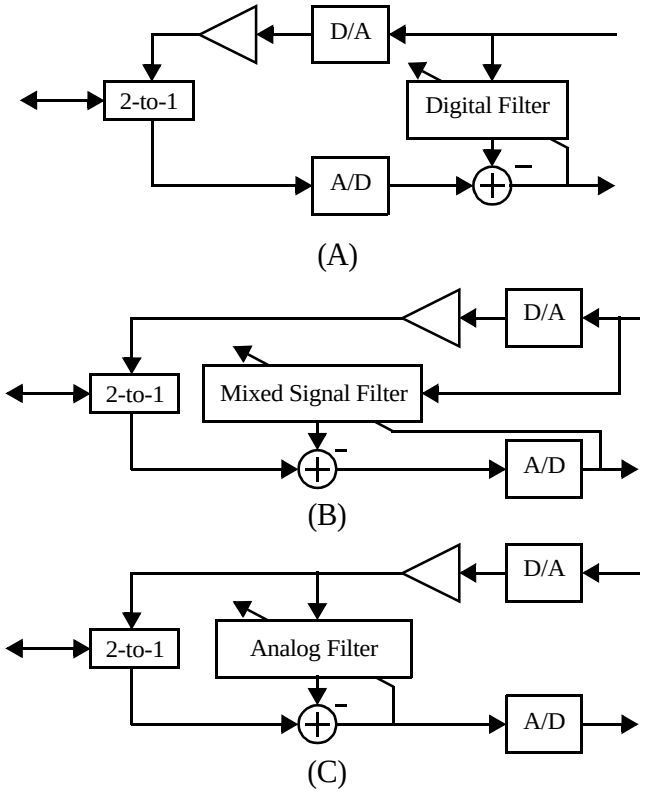


Figure 1.2 Adaptive echo cancellation in a full-duplex wired digital communication transceiver: (A) digital, (B) mixed signal, (C) analog.

1.2.3 High Speed Serial Links

For very high speed applications, an A/D and digital equalizer may be impractical so analog equalization is the only option. One such application is high speed serial links over coaxial cable. In [26] a continuous time digitally programmable CMOS analog equalizer was used for 155 Mb/s SONET over co-ax. In [27] and [28] bipolar continuous time analog circuits were used for adaptive equalization of a coaxial cable up to 400 Mb/s and 2.5 Gb/s respectively. A 4-PAM 8 Gb/s signal was equalized for co-ax using a discrete time analog filter in [29].

Considerable effort has also been aimed at equalizing chip-to-chip interfaces operating at several Gb/s. So far, most of the effort has been directed at pre-equalization using a discrete time mixed signal transmitter [30], [31]. However, in [31] a bondwire induc-

tance was used to provide high frequency peaking in a continuous time received signal at 8 Gb/s.

Although digital circuits in new process technologies will eventually reach these speeds, there will remain a frequency limit beyond which analog adaptive signal processing is more efficient. In order to keep the analog circuit design as simple as possible, the equalizers usually have only one or two adapted parameters, usually adjusting the amount of high-frequency peaking in the filter's transfer function. Since the connections are fixed, fast adaptation is generally not required so the adaptive algorithm need not be implemented with high speed analog circuits. The best compromise is often to use analog circuits for the high speed signal path only, and slower digital circuits for the adaptation.

1.2.4 Optical and Wireless

As digital CMOS circuits increase in speed and decrease in power consumption, research on analog adaptive filters will continue to move towards higher speed applications. Optical and RF signals operate at frequencies still far beyond the practical limits of integrated adaptive digital signal processing. Researchers are already beginning to consider the possibility of analog adaptive signal processing for these applications. Integrated analog equalizers have already been tested in experimental optical systems operating at 10 Gb/s [32]. However, the equalizer parameters were manually optimized. In [33], the control voltages on an array of varactors were adapted (in simulations) to direct the radiation pattern of an antenna array. In both cases, practical hardware-efficient adaptation algorithms are lacking.

1.3 Background

As mentioned earlier, the LMS algorithm is the most popular algorithm for the adaptation of integrated filters today. Although several other algorithms with superior convergence properties exist, the LMS algorithm remains popular because of its robust and

straightforward digital implementation. This section provides some background on the LMS algorithm and the dc offset effects which limit its use in analog adaptive filters [34].

1.3.1 The Least Mean-Square (LMS) Algorithm

A general LMS adaptive filter is shown in Fig. 1.3. It has two inputs (the filter input u, and a "desired" or reference output, d) and two outputs (the filter output y, and an error signal $e = d - y$). These may be either continuous or discrete time random processes whose statistics will depend upon the particular application. For all of the applications considered here, the signals u and d are either jointly stationary, or their joint probability distributions vary slowly compared to their bandwidth.

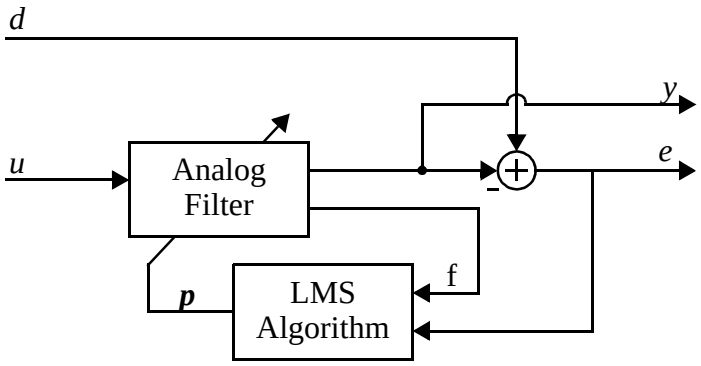


Figure 1.3 An LMS analog adaptive filter as a 2-input, 2-output system.

The performance criterion used for LMS adaptation is the mean-squared error (MSE),

$$\varepsilon(\boldsymbol{p}(k)) = E[(d(k) - y(k))^2] = E[e^2(k)] \tag{1.1}$$

where the operator $E[\ ]$ denotes expectation and discrete time signals have been assumed. In an adaptive system, the parameter vector $\boldsymbol{p}(k)$ and, hence, the MSE $\varepsilon(\boldsymbol{p}(k))$ are functions of time. The LMS algorithm is a gradient descent optimizer, which means that it seeks the parameter vector which minimizes $\varepsilon(\boldsymbol{p}(k))$ by updating $\boldsymbol{p}(k)$ iteratively in a direction opposite the gradient $\nabla_{\boldsymbol{p}(k)}\varepsilon(\boldsymbol{p}(k))$. In discrete time, the update rule is

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) - \mu \cdot \nabla_{\boldsymbol{p}(k)}\varepsilon(\boldsymbol{p}(k)) \tag{1.2}$$

where μ is a constant which determines the rate of adaptation.

The simple yet brilliant idea put forward by Widrow and Hoff in [35] was to drop the expectation operator when substituting Eqn. (1.1) into Eqn. (1.2). In doing so, they are taking the instantaneous value of the squared-error to be a noisy estimate of its expected value, $\varepsilon(\boldsymbol{p}(k)) = E[e^2(k)] \approx e^2(k)$. The resulting update rule is

$$\begin{aligned}
\boldsymbol{p}(k+1) &= \boldsymbol{p}(k) - \mu \cdot \nabla_{\boldsymbol{p}(k)} e^2(k) \\
&= \boldsymbol{p}(k) - \mu \cdot \frac{de^2(k)}{de(k)} \cdot \nabla_{\boldsymbol{p}(k)} e(k) \\
&= \boldsymbol{p}(k) - \mu \cdot (2e(k)) \cdot \nabla_{\boldsymbol{p}(k)} (d(k) - y(k)) \\
&= \boldsymbol{p}(k) - 2\mu e(k) \cdot (-\nabla_{\boldsymbol{p}(k)} y(k)) \\
&= \boldsymbol{p}(k) + 2\mu e(k) \cdot \mathrm{f}(k)
\end{aligned} \tag{1.3}$$

where $\mathrm{f}(k)$ is an instantaneous estimate of the gradient $\nabla_{\boldsymbol{p}(k)} y(k)$. Eqn. (1.3) is the parameter update equation for the LMS algorithm.[1] The parameter μ determines the rate of adaptation. In digital filters, the gradient signals $\mathrm{f}(k)$ are usually readily available in digital form [3]. In analog filters, additional analog circuitry if often required to generate the gradient signals from the filter's internal state signals. Furthermore, if the adaptation algorithm is implemented digitally the gradient signals must be digitized, which is an area- and power-hungry task.

Alternately, the LMS algorithm can operate directly on analog gradient signals using analog circuitry. In this case, a continuous time formulation of Eqn. (1.3) is used [36]:

$$\boldsymbol{p}(t) = 2\mu \int_{-\infty}^{t} e(\tau) \cdot \mathrm{f}(\tau) \cdot d\tau \tag{1.4}$$

Unfortunately, under these circumstances the LMS algorithm is sensitive to dc offsets on the state and error signals.

1.3.2 DC Offset Effects in Analog LMS

A block diagram of the LMS parameter update rule appears in Fig. 1.4. If the parameter updates are being performed using analog circuitry, dc offsets will appear at the inputs to the multiplier (m_e, m_ϕ) and integrator ($m_{e\phi}$). These offsets prevent the LMS

1. For a more rigorous treatment of the LMS algorithm, the reader is referred to [1].

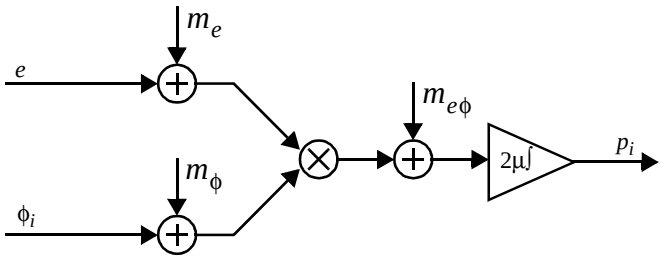


Figure 1.4 Analog implementation of the LMS parameter update equation.

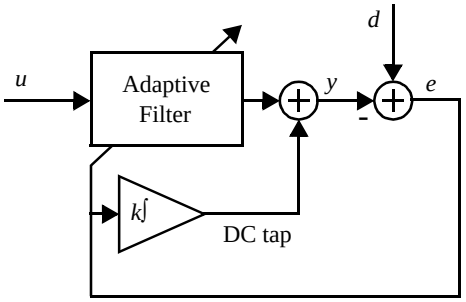


Figure 1.5 DC tap for adaptive offset cancellation.

algorithm from adapting to the optimal filter parameter values [3], [37], [38]. The excess steady state MSE is related to $m_{e\phi}$ and to the product $m_e \cdot m_\phi$.

Dc offsets represent a significant performance limitation in many analog adaptive filters [2], [39], [40]. Much research has been done to minimize the negative influence of dc offsets on analog adaptive filters. It was shown in [41] that the SE-LMS and SS-LMS algorithms are somewhat more robust than full-LMS with respect to dc offsets. An algorithmic approach to combatting dc offset effects in transversal filters was proposed in [37] requiring another set of N adapted coefficients. Circuit-level techniques for offset-compensation in analog adaptive filters have also been used with varying degrees of success in, for instance, [38], [42], and [43].

A relatively simple way to eliminate dc offsets on the error signal e is to add a dc offset cancellation tap to the filter output. Shown in Fig. 1.5, this tap can be included in any filter structure and essentially forces e to have zero dc content. As a result, $m_e \approx 0 \Rightarrow m_e \cdot m_\phi \approx 0$. In [44], the SS-LMS algorithm was used to adapt the dc offset cancellation tap resulting in a median-based offset compensation scheme. A hardware-

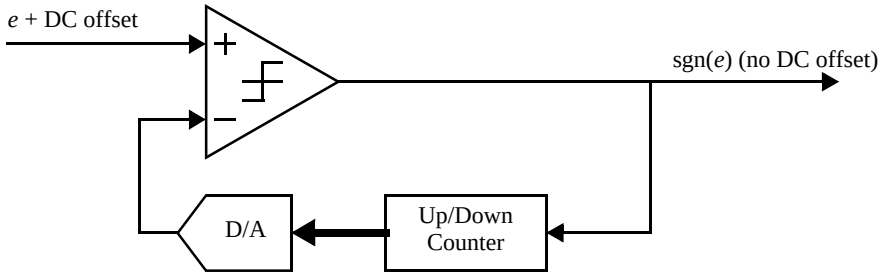


Figure 1.6 Median-based DC offset compensation scheme.

efficient realization is shown in Fig. 1.6. Unfortunately, it does not eliminate excess MSE entirely since dc offsets introduced by $m_{e\phi}$ persist.

Digital implementations of the LMS update equation are advantageous because performing the multiplication $e \cdot \phi$ digitally introduces no dc offset. Hence, $m_{e\phi} = 0$. By combining a digital implementation of the LMS algorithm with offset compensation at the filter output, one can perform LMS adaptation of an analog filter with no dc offset effects. Therefore, it is highly desirable to use digital adaptation for analog adaptive filters.

1.4 Outline

In order to perform digital adaptation of an analog filter, one requires both a digitally programmable analog filter, and an algorithm which can digitally adapt analog filter parameters. The next chapter of this dissertation addresses the first requirement, and the remainder tackles the second.

Chapter 2 describes circuit techniques for implementing a digitally programmable integrated analog filter. A submicron CMOS technology is targeted to ensure that the techniques are compatible with integrated mixed signal systems. The design of a 5th order continuous time filter is described and test results from a prototype are presented. The circuits are presented at the start of this dissertation because the prototype is used as a test vehicle for adaptive algorithms in later chapters.

In Chapter 3 a simple yet robust algorithm, called the “dithered linear search”, is described which adapts filter parameters by correlating changes in the output squared

error to independent dither simultaneously applied to each of the filter parameters. The algorithm has a straightforward hardware implementation requiring only a few gates of digital circuitry and no additional analog hardware. Theoretical analysis, simulations, and experimental results are used to verify the algorithm's robustness. Implementation issues such as quantization and dc offsets are also considered.

Chapter 4 describes a novel technique for performing LMS adaptation. The technique obviates the LMS algorithm's need to have access to the internal state signals of a filter. Instead, gradient information is obtained by performing simple digital signal processing on the digitized filter input. The resulting algorithm performs identically to the LMS algorithm, yet is much more practical for mixed signal systems. Again, theoretical, simulated, and experimental results are used to verify the algorithm.

In Chapter 5, another algorithm is described which generates gradient information without access to the filter's internal states. Although the implementation is somewhat more complicated than the approach in Chapter 4, this time only the filter's output must be digitized.

In the conclusion (Chapter 6) the work is summarized, the various algorithms are compared, and future directions are surmised.

1.5 References

[1] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*, Prentice Hall, New Jersey, 1985.

[2] K. A. Kozma, D. A. Johns, and A. S. Sedra, "Automatic Tuning of Continuous-Time Integrated Filters Using an Adaptive Filter Technique," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 1241-1248, Nov. 1991.

[3] D. A. Johns, W. M .Snelgrove, and A. S. Sedra, "Continuous-Time LMS Adaptive Recursive Filters," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 769-778, July 1991.

[4] R. Lucky, "Automatic equalization for digital communication," *Bell Syst. Tech. J.*, vol. 44, pp. 547-588, April 1965.

[5] R. D. Cideciyan, F. Dolivo, R. Hermann, W. Hirt, and W. Schott, "A PRML System for Digital Magnetic Recording," *IEEE J. Select. Areas Commun.*, vol. 10, pp. 38-56, Jan. 1992.

[6] M. H. Shakiba, D. A. Johns, K. W. Martin, "BiCMOS Circuits for Analog Viterbi Decoders," *IEEE Trans. Circuits Syst. II*, vol. 45, pp. 1527-1537, Dec. 1998.

[7] R. G. Yamasaki, *et al.*, "A 72 Mb/s PRML disk-drive channel chip with an analog sampled-data signal processor," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 278-279, Feb. 1994.

[8] N. P. Sands, M. W. Hause, G. Liang, G. Groenewold, S. Lam, C.-H. Lin, J. Kuklewicz, L. Lang, and R. Dakshinamurthy, "A 200Mb/s Analog DFE Read Channel," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 72-73, Feb. 1996.

[9] R. Gomez, M. Rofougaran, and A. A. Abidi, "A Discrete-Time Analog Signal Processor for Disk Read Channels," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 212-213, Feb. 1993.

[10] D. Xu, Y. Song, and G. T. Uehara, "A 200 MHz 9-Tap Analog Equalizer for Magnetic Disk Read Channels in 0.6μm CMOS," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 74-75, Feb. 1996.

[11] S. Kiriaki, T. L. Viswanathan, G. Feygin, B. Staszewski, R. Pierson, B. Krenik, M. de Wit, and K. Nagaraj, "A 160-MHz Analog Equalizer for Magnetic Disk Read Channels," *IEEE J. Solid-State Circuits*, vol. 32, pp. 1839-1850, Nov. 1997.

[12] N. Parsi, N. Rao, R. Burns, A. Chaiken, M. Chambers, R. Cheung, B. Forni, D. Harmishfeger, C. Jam, S. Kaylor, M. Pennell, J. Perez, M. Rohrbaugh, M. Ross, G. Stuhlmiller, and N. Weiner, "A 200Mb/s PRML Read/Write Channel IC," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 66-67, Feb. 1996.

[13] J. Sonntag, O. Agazzi, P. Aziz, H. Burger, V. Comino, M. Heimann, T. Karanink, J. Khoury, G. Madine, K. Nagaraj, G. Offord, R. Peruzzi, J. Plany, N. Rao, N. Sayiner, P. Setty, and K. Threadgill, "A High Speed, Low Power PRML Read Channel Device," *IEEE Trans. Magnetics*, vol. 31, pp. 1186-1195, March 1995.

[14] P. K. D. Pai, A. D. Brewster, and A. Abidi, "Analog Front-End Architectures for High-Speed PRML Magnetic Read Channels," *IEEE Trans. on Mag.*, vol. 31, pp. 1103-1108, March 1995.

[15] P. K. D. Pai, A. D. Brewster, and A. Abidi, "A 160-MHz Analog Front-End IC for EPR-IV PRML Magnetic Storage Read Channels," *IEEE J. Solid-State Circuits*, vol. 31, pp. 1803-1816, Nov. 1996.

[16] J. E. C. Brown, P. J. Hurst, B. C. Rothenberg, and S. H. Lewis, "A CMOS Adaptive Continuous-Time Forward Equalizer, LPF, and RAM-DFE for Magnetic Recording," *IEEE J. Solid-State Circuits*, vol. 34, pp. 162-169, Feb. 1999.

[17] J. N. Babanezhad, "A 3.3V Analog Adaptive Line-Equalizer for Fast Ethernet Data Communication," *IEEE Custom Integrated Circuits Conf.*, pp. 343-346, June 1998.

[18] A. Shoval, O. Shoaei, K. O. Lee, and R. H. Leonowich, "A CMOS Mixed-Signal 100 Mb/s Receive Architecture for Fast Ethernet," *IEEE Custom Integrated Circuits Conf.*, pp. 253-256, June 1999.

[19] O. Shoaei, A. Shoval, and R. H. Leonowich, "A 3V Low-Power 0.25 μm CMOS 100 Mb/s Receiver for Fast Ethernet," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 308-309, Feb. 2000.

[20] P. Amini and O. Shoaei, "A Low-Power Gigabit Ethernet Analog Equalizer," *IEEE Int. Symp. Circuits and Systems*, vol. 1, pp. 176-179, May 2001.

[21] J. P. Roesgen and G. H. Warren, "An Analog Front End Chip For V.32 Modems," *IEEE Custom Integrated Circuits Conf.*, pp. 16.1/1-16.1/5, May 1989.

[22] G. J. Smolka, "Analog CMOS Circuits for ISDN," *Proc. 1988 IEEE Int. Symp. Circuits and Systems*, vol. 2, pp. 1927-1930, June 1988.

[23] O. Agazzi, D. A. Hodges, D. G. Messerschmit, and W. Lattin, "Echo Canceller for a 80kbs Baseband Modem," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 144-145, Feb. 1982.

[24] F. Pecourt, J. Hauptmann, and A. Tenen, "An Integrated Adaptive Analog Balancing Hybrid for Use in (A)DSL Modems," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 252-253, Feb. 1999.

[25] T.-C. Lee and B. Razavi, "A 125 MHz Mixed-Signal Echo Canceller for Gigabit Ethernet on Copper Wire," *IEEE J. Solid-State Circuits*, vol. 36, pp. 366-373, March 2001.

[26] M. Altmann, J. M. Caia, R. Morle, M. Dunsmore, Y. Xie, and N. Kocaman, "A low-power CMOS 155 Mb/s Transceiver for SONET/SDH over Co-ax & Fibre," *IEEE Custom Integrated Circuits Conf.*, pp. 127-130, May 2001.

[27] A. J. Baker, "An Adaptive Cable Equalizer for Serial Digital Video Rates to 400 Mb/s," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 174-175, Feb. 1996.

[28] M. H. Shakiba, "A 2.5Gb/s Adaptive Cable Equalizer," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 396-397, Feb. 1999.

[29] R. Farjad-Rad, C.-K. K. Yang, and M. A. Horowitz, "A 0.3 μm CMOS 8 Gb/s 4-PAM Serial Link Transceiver," *IEEE J. Solid-State Circuits*, vol. 35, pp. 757-764, May 2000.

[30] M.-J. E. Lee, W. J. Dally, and P. Chiang, "Low-Power Area-Efficient High-Speed I/O Circuit Techniques," *IEEE J. Solid-State Circuits*, vol. 35, pp. 1591-1599, Nov. 2000.

[31] C.-K. K. Yang, V. Stojanovic, S. Modjtahedi, M. A. Horowitz, and W. F. Ellersick, "A Serial-Link Transceiver Based on 8-GSamples/s A/D and D/A Converters in 0.25 μm CMOS," *IEEE J. Solid-State Circuits*, vol. 36, pp. 1684-1692, Nov. 2001.

[32] F. Buchali, H. Bulow, W. Baumert, R. Ballentin, and T. Wehren, "Reduction of the Chromatic Dispersion Penalty at 10 Gbit/s by Integrated Electronic Equalisers," *Optical Fiber Comm. Conf.*, vol. 3, pp. 268-270, March 2000.

[33] T. Ohira, "Emerging Adaptive Antenna Techniques for Wireless Ad-hoc Networks," *Int. Symp. Circuits and Systems*, vol. 4, pp. 858-861,

[34] A. Carusone and D. A. Johns, "Analogue Adaptive Filters - Past and Present," *IEE Transactions on Circuits, Systems, and Devices*, February 2000, pp. 82-90.

[35] B. Widrow and M. E. Hoff, "Adaptive Switching Circuits," *IRE 1960 WESCON Conv. Rec.*, pp 96-104, 1960.

[36] B. Widrow, P. E. Mantey, L. J. Griffiths, and B. B. Goode, "Adaptive antenna systems," *Proc. IEEE*, vol. 55, pp. 2143-2159, Dec. 1967.

[37] C.-P. J. Tzeng, "An Adaptive Offset Cancellation Technique for Adaptive Filters," *IEEE Trans. on Acoust., Speech, Signal Processing*, vol. 38, pp. 799-803, May 1990.

[38] U. Menzi and G. S. Moschytz, "Adaptive Switched-Capacitor Filters Based on the LMS Algorithm," *IEEE Trans. Circuits Syst.*, vol. 40, pp. 929-942, Dec. 1993.

[39] T. Kwan and K. Martin, "An Adaptive Analog Continuous-Time CMOS Biquadratic Filter," *IEEE J. Solid-State Circuits*, vol. 26, pp. 859-867, June 1991.

[40] A. Shoval, W. M. Snelgrove, and D. A. Johns, "A 100Mb/s BiCMOS Adaptive Pulse-Shaping Filter," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1692-1702, Dec. 1995.

[41] A. Shoval, D. A. Johns, and W. M. Snelgrove, "Comparison of DC Offset Effects in Four LMS Adaptive Algorithms," *IEEE Trans. Circuits Syst. II*, vol. 42, pp. 176-185, March 1995.

[42] H. Qiuting, "Offset Compensation Scheme for Analogue LMS Adaptive FIR Filters," *Electron. Lett.*, vol. 38, pp. 1203-1205, June 1992.

[43] F. J. Kub and E. W. Justh, "Analog CMOS Implementation of High Frequency Least-Mean Square Error Learning Circuit," *IEEE J. Solid-State Circuits*, vol. 30, pp. 1391-1398, Dec. 1995.

[44] A. Shoval, D. A. Johns and W. M. Snelgrove, "Median-Based Offset Cancellation Circuit Technique," *IEEE Int. Symp. Circuits and Systems*, pp. 2033-2036, May 1992.

Chapter 2

Digitally Programmable Gm-C Filters

2.1 Introduction

As digital CMOS processes continue to advance, the size and power consumption of digital circuitry decreases dramatically. At the same time, lower supply voltages make the design of high resolution analog-to-digital converters increasingly difficult. Therefore, there is great motivation to implement some of the signal processing in the analog domain in order to reduce A/D converter resolution. Analog adaptive filters can be very useful in this regard. A major problem limiting the use of analog adaptive filters today is that analog adaptation circuitry is difficult to design and suffers from dc offset effects [1], [2]. In order to obviate the need for analog adaptation circuitry, it is desirable to have a digitally programmable analog filter, which is the focus of this chapter.

Most analog adaptive filters in use today tune only one or two parameters, such as the ω_0 and Q of a biquad [3] or one zero in a higher order filter [4]. However, it is well known that in many applications the ability to adapt several parameters, particularly filter poles, can significantly reduce the complexity of the remainder of the analog front end and/or DSP [5], [6]. This is usually not attempted in practice to ensure stability of both the filter and the adaptation algorithm. However, more sophisticated digital adaptation algorithms are capable of dealing with these problems (see, for instance, [7]). Therefore, analog filters with several digitally programmable parameters are examined here.

Discrete time integrated filters generally require opamps with a unity-gain bandwidth several times greater than the bandwidth of the signal being filtered in order to achieve reasonable settling times. Therefore, for high bandwidth applications continuous time filters are preferred. Log-domain continuous time filters make use of bipolar devices to produce a logarithmic conversion between current and voltage [8]. Unfortunately, bipolar devices are poorly characterized and relatively slow in digital CMOS processes. MOSFET-C integrators require high-speed output stages capable of driving resistive loads which also makes them difficult to implement in digital CMOS processes. Gm-C filters are currently the most popular technique for the realization of high-speed integrated filters in CMOS. Gm-Opamp-C circuits are discussed here. The addition of an opamp increases the integrators' output impedance by the gain of the opamp - an important consideration in deep submicron processes since short channel effects degrade MOSFET output impedances. Furthermore, linearity may be improved since the topology is less sensitive to nonlinear parasitic capacitances at the transconductors' outputs.

Since the time constant of a Gm-Opamp-C integrator is determined by the ratio Gm/C, a programmable filter must have programmable transconductances, Gm, and/or programmable capacitances, C. In this work capacitances are held constant and the time constants are controlled via digitally programmable transconductances. It has been shown that programmable filters based upon constant capacitances are optimal with respect to noise and dynamic range [9].

This chapter describes a 5th order digitally programmable orthonormal ladder filter in a 0.25 μm CMOS process intended for analog adaptive filtering applications. All poles and zeros are digitally programmable so that the adaptation algorithm can be implemented digitally. The orthonormal ladder structure has the advantage of maintaining near-optimum dynamic range scaling for any stable pole and zero locations [10], thus making it particularly well suited to highly programmable analog filters. A digital process technology was used (no double-poly or thick oxide transistors) to facilitate low cost integration on mixed-signal systems.

2.2 Circuit Description

2.2.1 A CMOS Transconductor

CMOS transconductors based upon the transconductance of a differential pair must have small input signal swings to ensure their voltage-current relationship remains linear. In order to allow for larger input signals (and hence improved dynamic range) a triode transistor can be used to source-degenerate the differential pair. Linearity can be further improved by using feedback to fix the drain-source voltage of the input transistors.

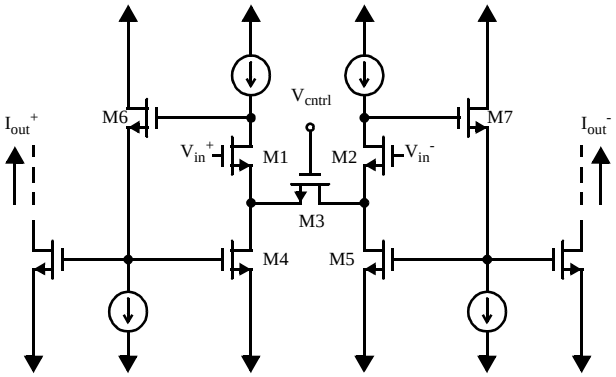
In the transconductor of Fig. 2.1A [11], M3 provides a triode-region transconductance and the source-followers at M6 and M7 provide a feedback path to increase linearity by maintaining constant gate-source voltages in transistors M1 and M2. However, this circuit is not well suited to low supply voltages because the gate-source voltages of two active devices are cascaded ($V_{gs,4}$ and $V_{gs,6}$). The circuit in Fig. 2.1B [3] is suitable for lower supply voltages, however its speed is hindered by the use of p-channel devices at M1, M2, and M3.

The transconductor shown in Fig. 2.1C was used in this design. Negative feedback is provided by directly connecting the gate of M4 (M5) to the drain of M1 (M2). By eliminating the source follower from Fig. 2.1A, lower supply voltages can be used and a parasitic pole is eliminated from the transconductor's frequency response. Unlike Fig. 2.1B, the signal path consists entirely of n-channel transistors; p-channel devices are used only for biasing.

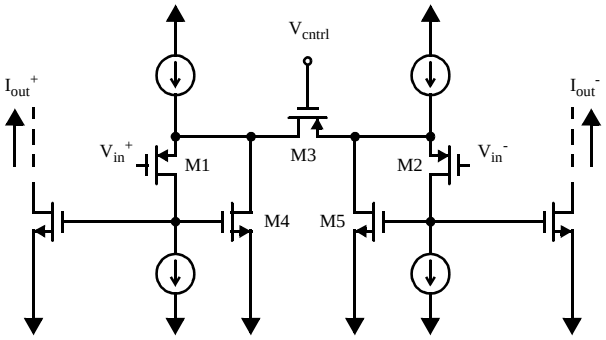
The dc small-signal transconductance of the circuit in Fig. 2.1C is approximately given by the following expression (derived in the Appendix, Section 2.7)

$$\frac{I_{out}^{+} - I_{out}^{-}}{V_{in}^{+} - V_{in}^{-}} = G_m = M(2g_{ds3} + g_{ds4} + g_{s1}) \tag{2.1}$$

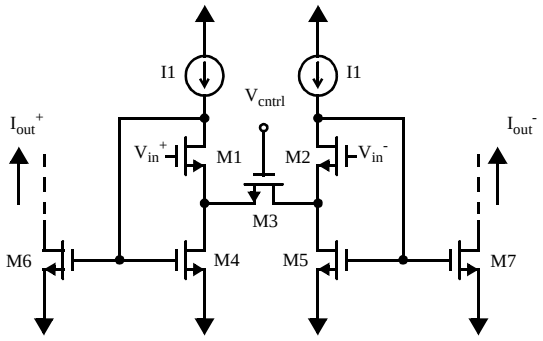
In Eqn. (2.1), M is the gain of the output current mirror formed by M4/M5 and M6/M7, g_{ds3} is the drain-source conductance of M3 in triode, g_{ds4} is the drain-source conductance of transistor M4 in saturation, and g_{s1} is due to the body effect of M1. The values of M and g_{ds3} are controlled digitally, but the values of g_{ds4} and g_{s1} are parasitic conductances which are not easily controlled.



(A)



(B)



(C)

Source	Current (μA)
I1	270

Transistor	W/L (μm)	M
M1, M2	8/.24	1
M3	*	*
M4 - M7	2.2/1	4

* see Fig. 2.2

Figure 2.1 Three transconductors based upon triode-region MOS devices (M3).
All use feedback to improve linearity: (A) [11] (B) [3] (C) *this work.*

2.2.2 Digitally Programmable Gain

Transistor M3 in Fig. 2.1C is realized as a binary-weighted array of unit transistors. The gate voltage of each can be set to either V_{SS} which turns it off, or V_{cntrl} which puts

it in triode [12]. The array in Fig. 2.2 provides 5 bits of control (g_2-g_{-2}) for the value of g_{ds3},

$$g_{ds3} = g_{ds(\text{unit-size})} \sum_{n=-2}^{2} 2^n g_n \tag{2.2}$$

An additional degree of programmability is provided at the output current mirrors formed by M5 & M7 and M4 & M6. Output stages are tuned on and off by two control bits, d_1 - d_0 (Fig. 2.3). As a result, the output current mirror gain can be set to M = 1, 0.25, or 0. When M < 1, parallel dc current sources are activated to sink the same bias current through the miller integrator input stage. In some transconductors, the positive and negative signal paths were cross-coupled using switches to provide digital control of the output polarity via a sign bit.

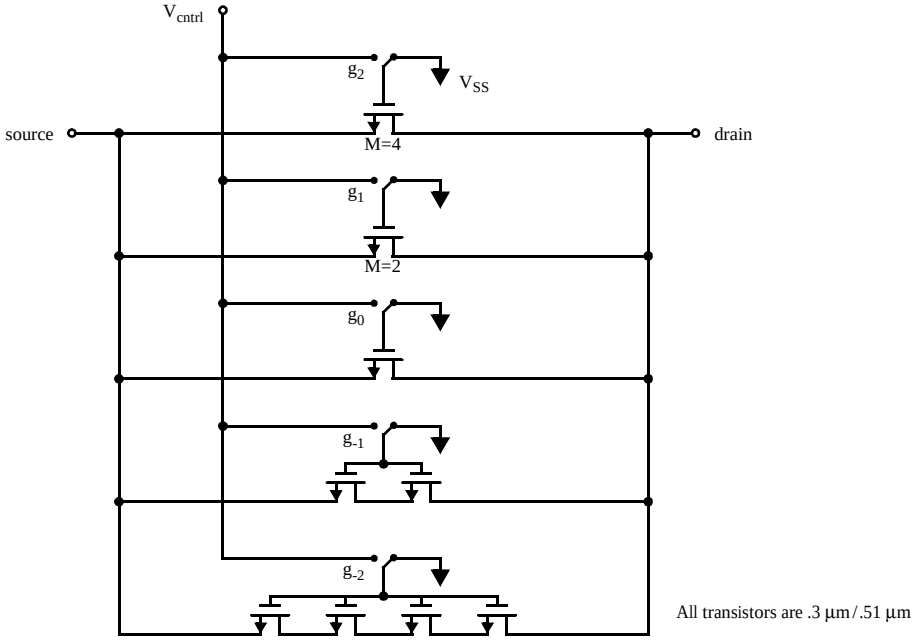
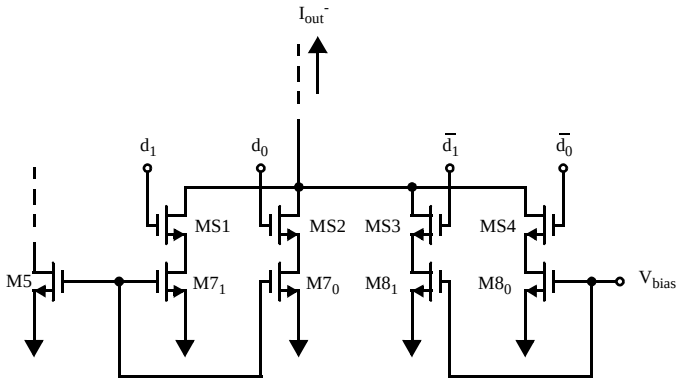


Figure 2.2 A five-bit programmable triode conductance.

2.2.3 Miller Integrator

It is possible to realize an integrator simply by placing capacitors at the outputs of a transconductor (Fig. 2.4A). However, in order to reduce the effects of non-linear junc-



Transistor	W/L (μm)	M
MS1, MS3	1.5/.24	3
MS2, MS4	1.5/.24	1
M5	2.2/1	4
$M7_1$, $M8_1$	2.2/1	3
$M7_0$, $M8_0$	2.2/1	1

Figure 2.3 Digital control of the output current mirror gain.

tion capacitances and the finite output impedance of the transconductor, a Miller integrator was used as shown in Fig. 2.4B.

In order to maintain high-speed operation it is desirable to use a simple opamp design with as few internal nodes as possible. In [13] a PMOS common-source stage provides high gain with common-gate current-mode inputs (Fig. 2.5A). The bandwidth of this circuit is limited by the high-impedance nodes at the gates of the common-source transistors M3 and M4. In order to provide sufficient gain, these devices must be relatively large. An NMOS cascode gain stage can be used instead as shown in Fig. 2.5B to provide the same gain with smaller transistor sizes and, hence, less capacitance at the speed-critical nodes.

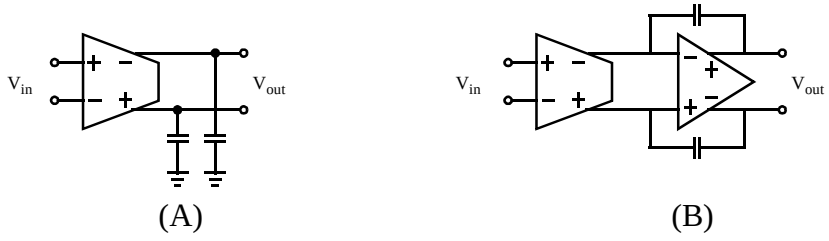
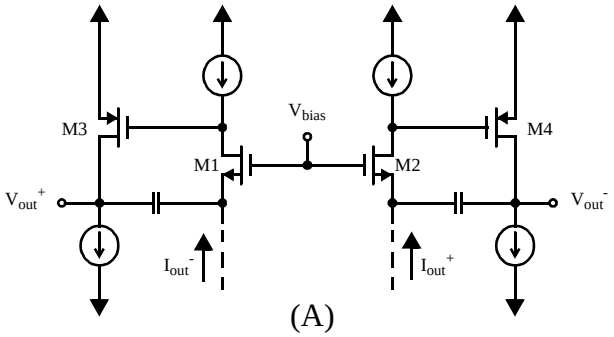


Figure 2.4 Differential integrators: (A) G_m-C (B) G_m-Opamp-C.

Again, the entire signal path consists of NMOS transistors; PMOS transistors are used only for biasing. Two bits, c_1-c_0, provide ±20% control of the value of the integrating capacitors. These controls are only intended to compensate for variations in oxide thickness and will remain fixed during the operation of the filter. The bottom plates of all

capacitors are connected to the integrator inputs which remain at a fixed potential making them insensitive to parasitics. Transistor M_C is biased in the triode region with a drain-source resistance of approximately 715 Ω to provide lead compensation.



Source	Current (µA)
I2	1350

Transistor	W/L (µm)	M
M1, M2	8/.24	3
M3 - M6	1/.27	30
M7 - M10	3.6/.24	18
M_C	5.4/.24	1

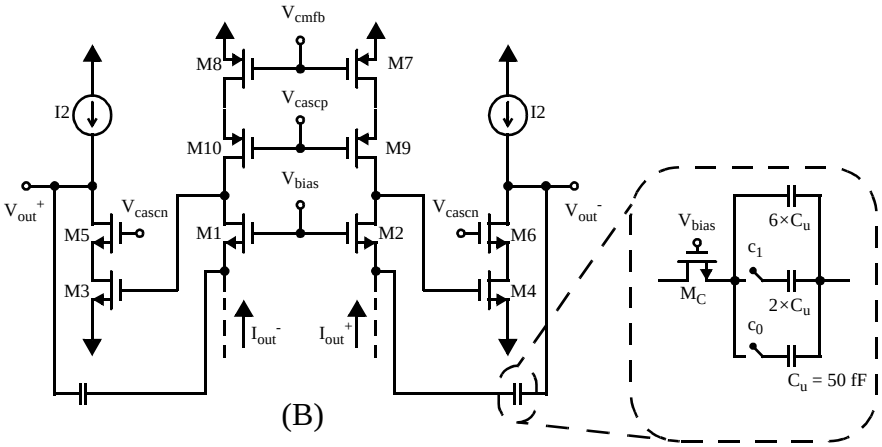


Figure 2.5 Current input CMOS active integrators.
(A) [13] (B) *this work*

2.2.4 Common Mode Feedback

Although a continuous time CMFB obviates the need for clock generation circuitry and avoids clock feed through glitches at the filter output, a continuous time design is complicated by the low supply voltage (2.5 V). The CMFB circuit must be stable and relatively fast to eliminate as much high-frequency common mode noise as possible. Furthermore, in this application it is also important that the common mode potential of the output accurately matches the common mode reference, V_{cmref}. Since the control voltage V_{cntrl} is referenced to V_{cmref}, a change in the output common mode level will affect the gain of a subsequent transconductor stage.

A block diagram of the design appears in Fig. 2.6. The transconductors are scaled-down copies of the signal-path transconductors with the outputs taken single-ended. By matching the common mode path to the signal path as closely as possible, systematic offsets between V_{cmref} and V_{out} are minimized. The diode-connected load serves as an I-V converter to drive current sources in the integrator.

A more detailed schematic of the original CMFB design appears in Fig. 2.7. The transconductors are scaled by one-half compared to the signal path transconductors except for triode transistors M3 which were sized to provide a drain-source conductance of 35 $\mu A/V$. The output currents of both transconductors are summed to reject differential signals. Transistor M10 is used to match the level shift introduced by M1 & M2 in Fig. 2.5B while M11 is matched to switches MS1 - MS4 in Fig. 2.3. An opamp fixes the drain of M7 to the same potential as the drain of M9 in Fig. 2.5B.

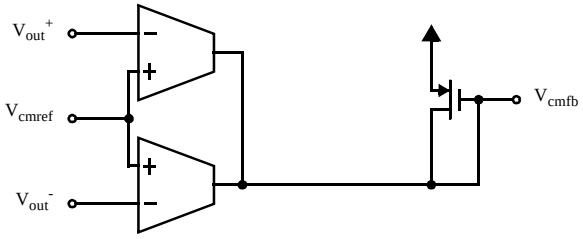
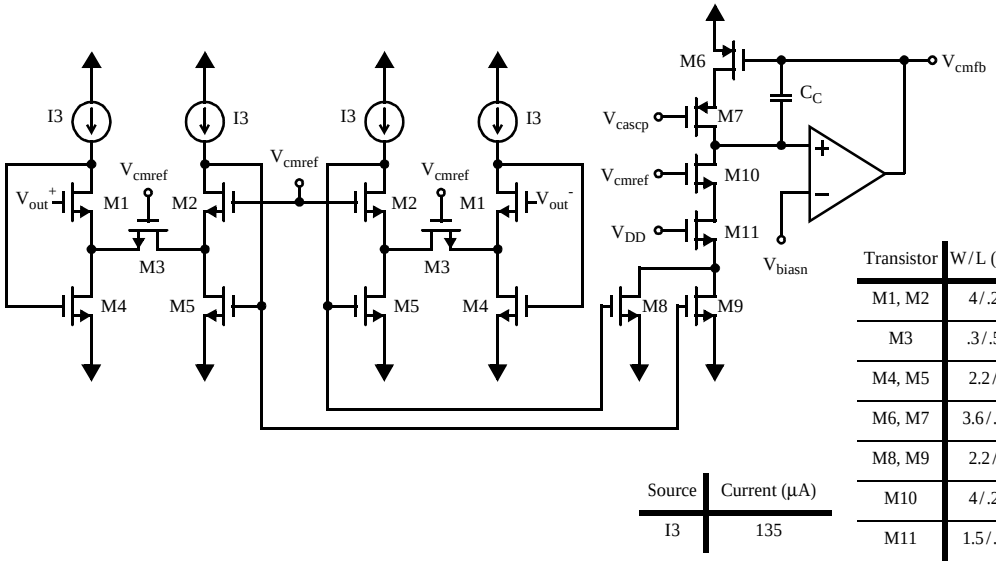


Figure 2.6 Block diagram of the continuous time CMFB circuit.



Transistor	W/L (μm)	M
M1, M2	4/.24	1
M3	.3/.51	1
M4, M5	2.2/1	2
M6, M7	3.6/.24	3
M8, M9	2.2/1	1
M10	4/.24	1
M11	1.5/.24	2

Figure 2.7 Schematic diagram of the continuous time CMFB.

Compensation of the loop is complicated by the presence of high-impedance nodes at the integrator outputs, V_{out}^{+} and V_{out}^{-}. In a simple Gm-C topology, it would be possible to compensate the loop by adding capacitance to the output nodes. However, because the integrating capacitors in this design are not grounded, they cannot be used to stabilize the CMFB loop. Instead, miller compensation was applied at the output of the CMFB circuit via C_C.

Another difficulty in designing the common mode feedback circuit is that it must fix the common mode level of both the first and second stages of the Miller integrator. Therefore, the feedback control voltage V_{cmfb} is applied to the first stage current source. As a result, there will be a large dc gain from V_{cmfb} to the filter outputs, V_{out}^{+} and V_{out}^{-}. To maintain stability of the feedback loop, a very small dc gain is realized in the CMFB circuit. Fig. 2.8 shows a block diagram of the entire CMFB loop with typical simulated dc gains.

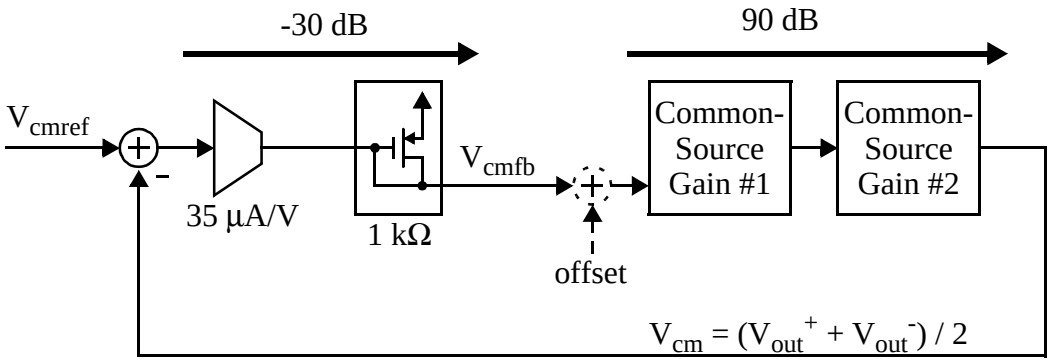


Figure 2.8 CMFB loop block diagram.

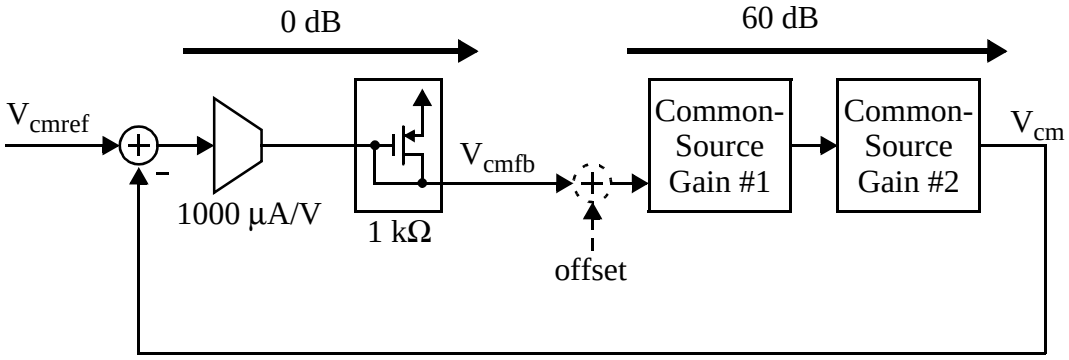


Figure 2.9 Redesigned CMFB loop block diagram.

Source	Current (μA)
I2	1350

Transistor	W/L (μm)	M
M1, M2	8/.24	3
M3 - M6	1/.27	30
M7 - M10	3.6/.24	2
M11 - M14	3.6/.24	16

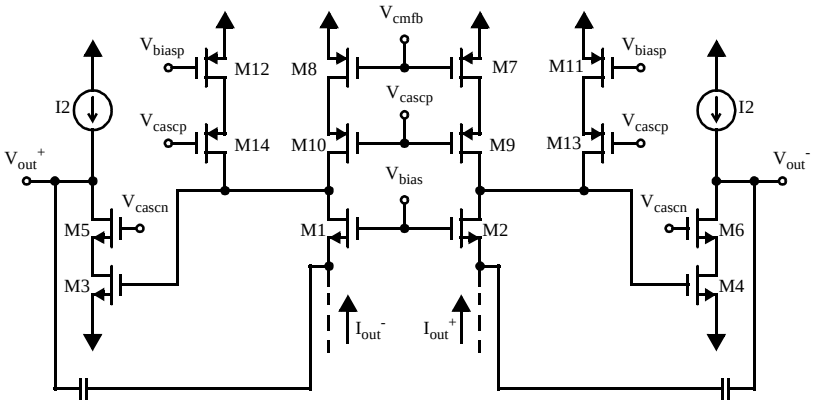


Figure 2.10 Redesigned Miller integrator to reduce common mode loop gain.

As you can see, the dc gain around the loop is 60 dB which is reasonable and easily stabilized. However, if an offset is introduced at V_{cmfb}, it will be amplified by 30 dB at the output. Offsets can be introduced by variations in supply voltage across the chip and threshold voltage mismatches. In this case, 50 mV of mismatch between the CMFB circuit and the Miller integrator is sufficient to cause the integrator outputs to saturate. Saturation of the common mode output levels was observed in the first integrated prototypes. Therefore, the CMFB was modified to redistribute dc gain more evenly around the loop (Fig. 2.9). The transconductor gain was increased by replacing M3 in Fig. 2.7 with 1.5 kΩ polysilicon resistors. At the same time, the gain of the first common-source stage was reduced by connecting V_{cmfb} to only two of 18 fingers in the first stage current source as shown in Fig. 2.10. The result is a much more robust CMFB.

2.2.5 Prototype

A prototype was fabricated in a standard digital 0.25 μm CMOS process. The 5th order orthonormal ladder filter structure is shown in Fig. 2.11. All of the summations are performed by tying together the output currents of transconductors. Where a summation node has fewer than three inputs, fixed dc current sources were added to properly bias the input stage of the subsequent miller integrator.

A die photo of the prototype is shown in Fig. 2.12. The total area of test chip is 2.5 mm × 1.7 mm. Of that, only 1.25 mm × 0.38 mm ≈ 0.5 mm^2 is occupied by the actual filter. The rest includes test circuitry, probe pads, and digital logic to store the filter coef-

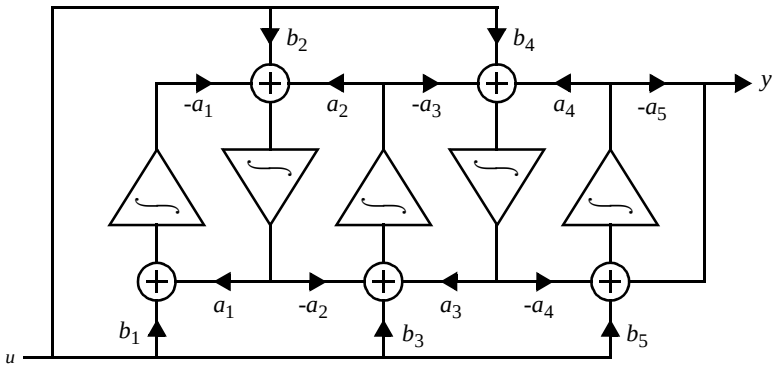


Figure 2.11 5th order orthonormal ladder filter structure with multiple feed-ins.

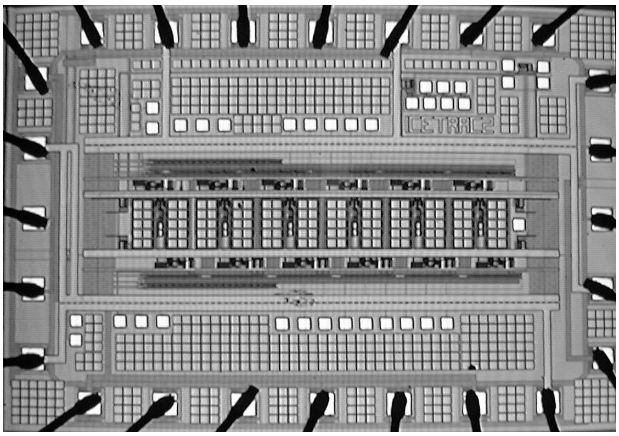


Figure 2.12 Die photo of the prototype.

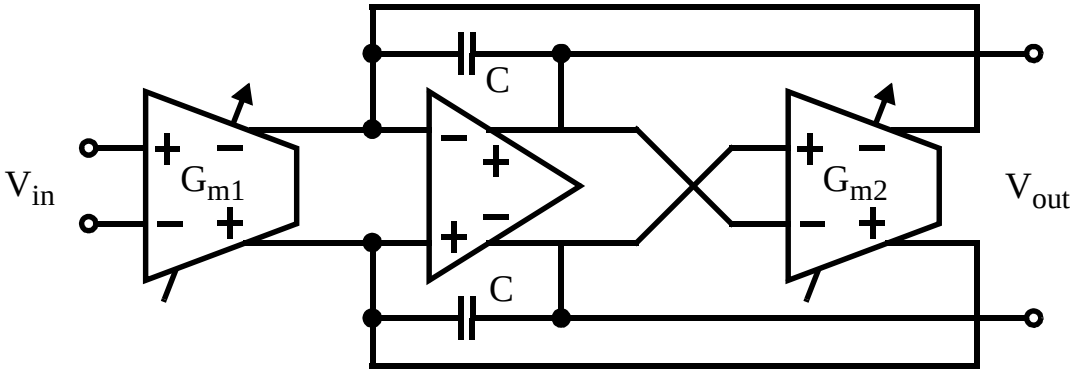


Figure 2.13 First order G_m-C filter with programmable pole and dc gain.

ficients. (A description of the digital interface is provided in an Appendix, Section 2.8) The chip also included a first order lowpass filter section as a test circuit to help charac-

terize the transconductor and integrator blocks (Fig. 2.13). Most of the filter area is occupied by metal-metal capacitors on layers 4 and 5. The capacitance per unit area could have been increased by using all of the metal layers in a stacked capacitor structure resulting in a 40% decrease in the analog filter layout area to 0.3 mm^2. This was not done to allow for proper extraction of the capacitors using an existing design flow.

2.3 Experimental Measurements

2.3.1 Programmable Transconductors

The programmable transconductors were tested using the first order filter in Figure 2.13. A sinusoid was input at 8 MHz with a constant small-signal amplitude of -30 dBm. The feedback gain G_{m2} was held constant while the control bits g_2-g_{-2} for G_{m1} were stepped through all 32 possible combinations. Fig. 2.14 shows the gain as a function of the 5-bit control word. Note that the top curve in Fig. 2.14 does not pass through the origin because even when all of the triode transistors are turned off, the transconductor in Fig. 2.1C has a nonzero gain due to parasitic conductances between the drain/source of M3 and ground. Reprogramming the transconductor's output current mirror gain to $M = 0.25$ results in the lower curve in Figure 2.14 which can be used to realize smaller gains. However, many transfer functions require feed-in gains of exactly zero. In these cases, the output current mirror gain can be set to zero.

2.3.2 Frequency Response

In order to measure the frequency response of the filter, a reference path was provided through the prototype including packaging, bondwires of the same length as the signal path, the same pad drivers, and the same PCB traces. The frequency response of both the reference path, $H_{ref}(s)$, and the signal path, $H_{total}(s)$, are measured using the test setup shown in Fig. 2.15. The frequency response of the filter alone is then given by,

$$H(s) = \frac{H_{total}(s)}{H_{ref}(s)} \tag{2.3}$$

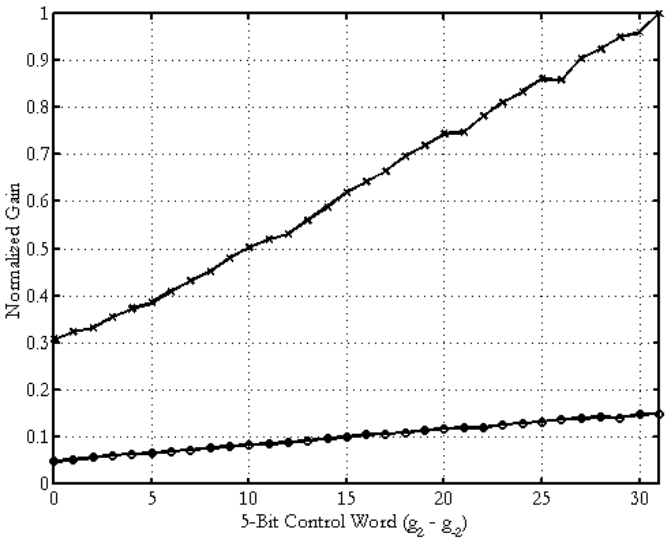


Figure 2.14 Gain of the first order filter as a function of the G_{m1} control word.
The output current mirror gain is $M = 1$ for the top curve (×) *and $M = 1/4$ for the lower curve* (O).

The magnitude response of the reference filter path is plotted in Fig. 2.16. The insertion loss at dc is 10 dB and remains between 10 dB - 11 dB up to 250 MHz. The ringing above 300 MHz appears because the output buffer termination is off-chip and is not matched to the 50 Ω test equipment. Most of the low frequency loss is due to the output buffers which are simply differential pairs with external resistive loads. Larger load resistors could have been used to increase the gain, but matching and linearity would have suffered.

To demonstrate the programmability of the filter, the orthonormal ladder was configured as a 5th order lowpass filter and reprogrammed with different cutoff frequencies and dc gains. The resulting magnitude responses are plotted in Fig. 2.17. Unfortunately, the cutoff is not as sharp as one would expect from a 5th order filter. This is partly due to the unpredictability of the exact feed-in and feedback gains. Unknown dc offsets and mismatches between stages effect the gains making it difficult to predict the transfer function which results from a given set of digital parameter values.

Another serious problem is that peaking in the magnitude response is observed around 500 MHz (Fig. 2.18). The peaking also occurs in simulations with the same pro-

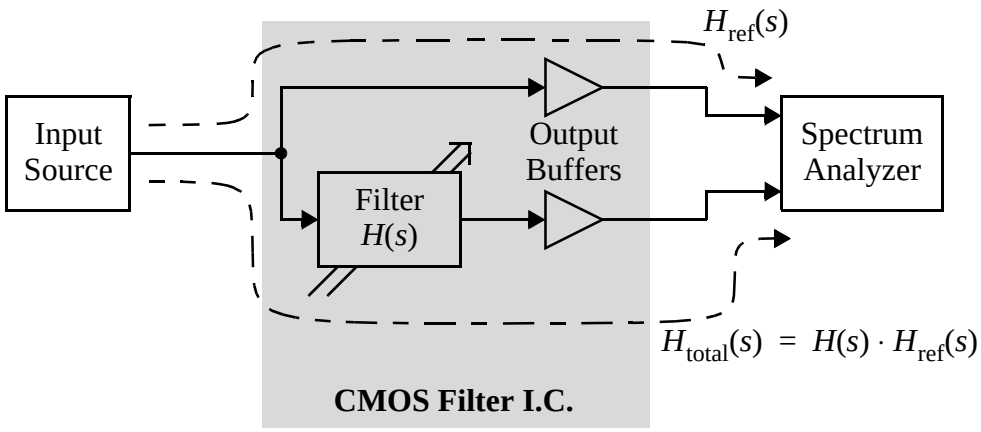


Figure 2.15 Test setup used to isolate the frequency response of the filter.

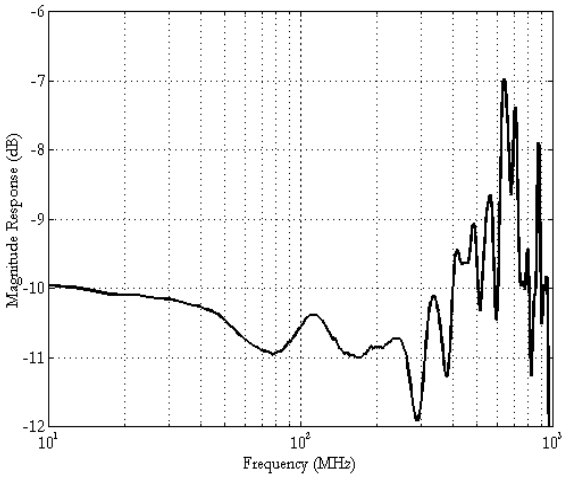
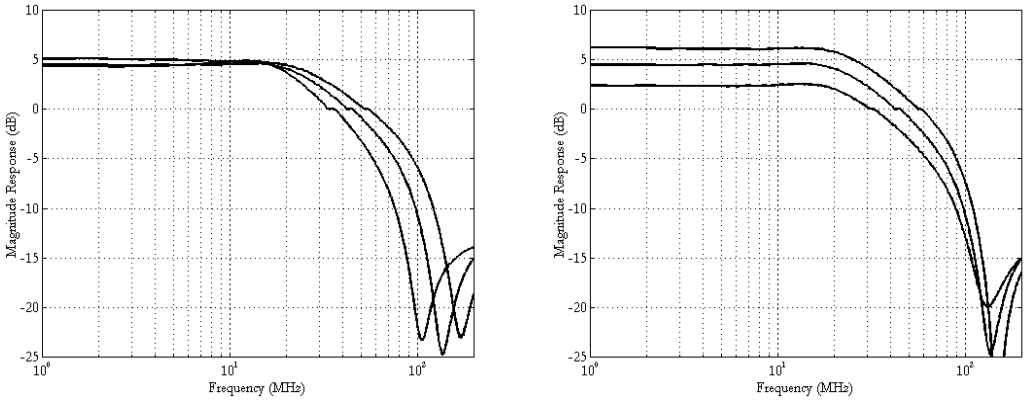


Figure 2.16 Magnitude response of the reference filter path, $H_{ref}(s)$.

grammed coefficient values (Fig. 2.19A). Unfortunately, the prototypes have this peaking because the compensation network was changed after complete chip simulations were complete. (The changes were verified by simulating only one integrator.) If the unit capacitors, C_u in Fig. 2.5, were increased from 50 fF to 70 fF and the triode resistance of Mc was decreased from 715 Ω to 400 Ω, the filter's maximum frequency of operation is decreased but stability of the integrators is improved. The resulting overall simulated magnitude response is shown in Fig. 2.19B. The peaking at 500 MHz is eliminated, the cutoff is much sharper, and a 2nd notch appears in the magnitude response



(A) (B)

Figure 2.17 Measured frequency responses of the orthonormal ladder.
The filter was configured as a lowpass filter with (A) *varying cutoff frequency and* (B) *varying dc gain.*

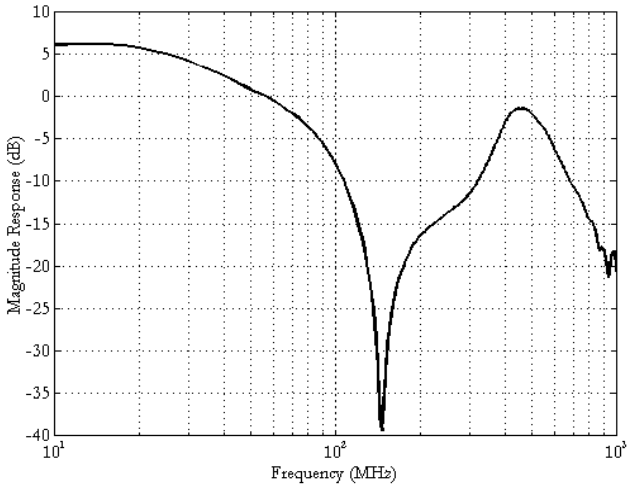


Figure 2.18 Measured frequency response of the lowpass orthonormal ladder showing ringing around 500 MHz.

which was not present in Fig. 2.19A due to phase errors caused by the poor compensation.

2.3.3 Linearity

First, the linearity of the first order filter (Fig. 2.13) was tested using an 8 MHz input tone at 200 mVpp. The output spectrum is shown in Fig. 2.20. The total harmonic dis-

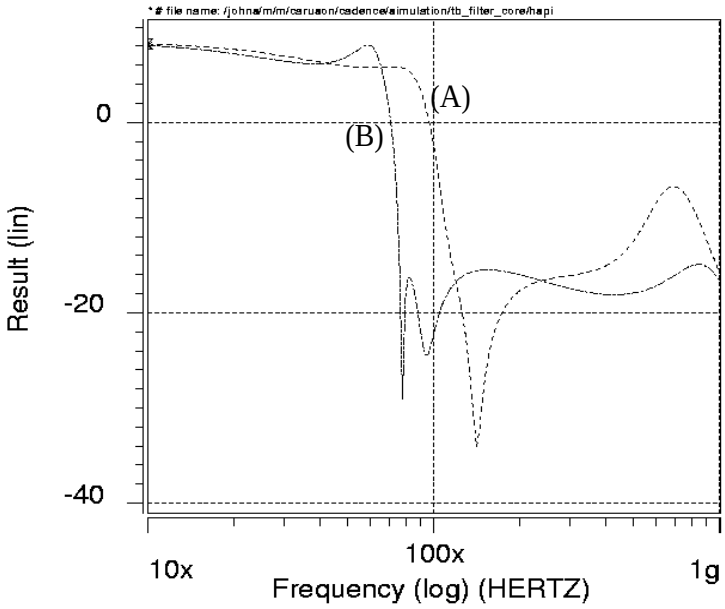


Figure 2.19 Simulated magnitude response of the 5th order lowpass filter.
(A) *peaking in the stopband with C_u = 50 fF and R_C = 715 W*
(B) *with the compensation network redesigned, C_u = 70 fF and R_C = 400 W.*

tortion is 41 dB limited by the 3rd harmonic, as one would expect in a fully-differential circuit.

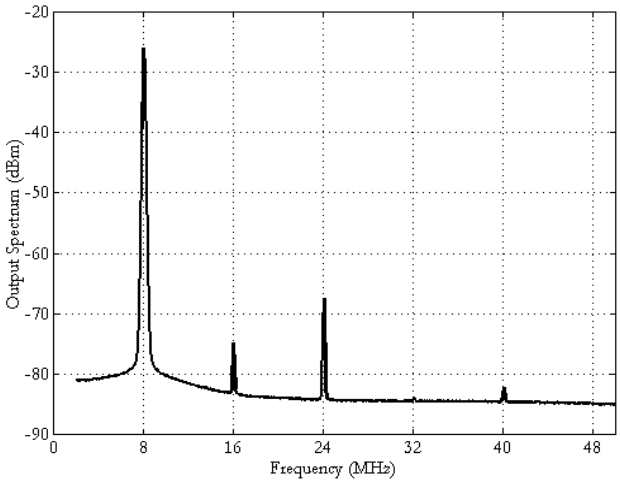


Figure 2.20 Output spectrum of the 1st order filter with an input tone at 8 MHz.

The linearity measurements for the 5th order structure, however, are limited by the 2nd order harmonics (Fig. 2.21). Total harmonic distortion (THD) was measured at dif-

ferent signal amplitudes (Fig. 2.22) and frequencies (Fig. 2.23) using the medium bandwidth setting from Fig. 2.17A. In Fig. 2.23, the linearity is worst at mid-range frequencies as one would expect since, at higher frequencies, the harmonics are attenuated by the filter itself.

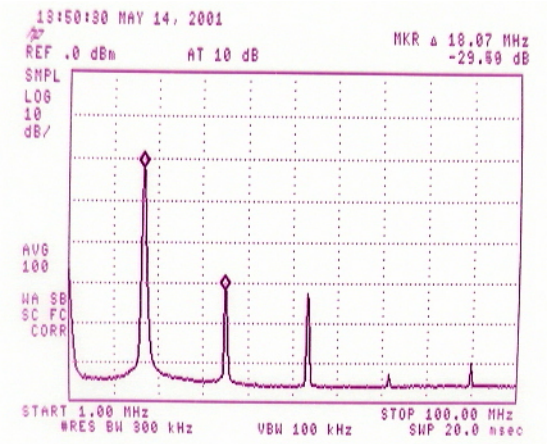


Figure 2.21 Spectrum analyzer screen shot of the 5th order filter output THD.
Testing done with a 125 mVpp input sinusoid at 18 MHz.

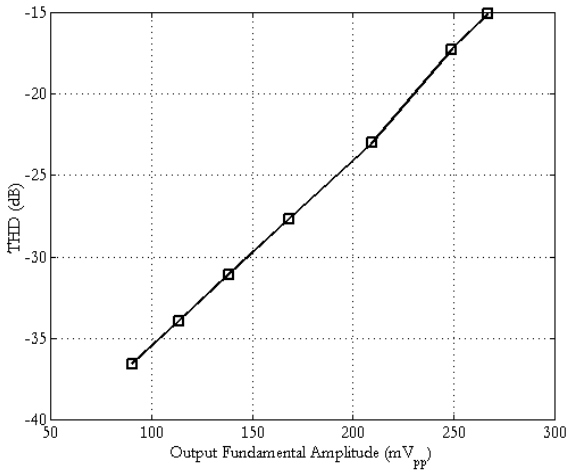


Figure 2.22 THD of the 5th order lowpass filter vs. output amplitude.
Testing done with a 18 MHz input tone, chosen so that the 2nd harmonic coincides with the filter's -3dB frequency.

The linearity of the analog building blocks appears to be moderate, as indicated by the 41 dB THD measured for the first order filer, which would be sufficient for applications with 6-8 bit front ends such as magnetic storage, fast ethernet over copper, and high-

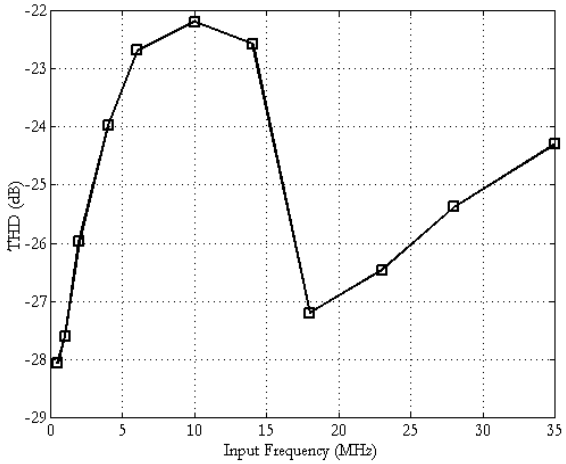


Figure 2.23 THD of 5th order lowpass filter vs. input frequency.
Testing done with a -14 dBm (125 mVpp) sinusoidal input.

speed wired serial links. However, the linearity of the 5th order filter was poor (25-30 dB). There are numerous sources of nonlinearity in the signal path including the nonlinear V-I characteristic of triode transistor M3 in Fig. 2.1C, the body effect of transistors M1 and M2 in Fig. 2.1C, and active devices in the transconductor or miller integrator that may enter triode for large signal swings. However, the presence of large 2nd order harmonics, which should be cancelled by the fully differential topology, indicates that imbalances between the positive and negative signal paths limit linearity. This was further evidenced by the different dc voltages observed on each side of the differential output.

Simulations revealed that the filter's linearity is particularly sensitive to mismatches in the transconductor's current mirror transistors M4 - M7 (Fig. 2.1C). Fig. 2.24 shows simulation results for the same lowpass filter as in Fig. 2.21, Fig. 2.22, and Fig. 2.23, again with a 125 mVpp differential input at 18 MHz. The threshold voltages of transistors M4 - M7 were made independent gaussian random variables in each transconductor. As the 3σ deviation of the threshold voltages is increased from 0 mV to 20 mV, the 2nd harmonic goes from being non-existent to being dominant. With a 3σ deviation of 20 mV (equivalent to a standard deviation of 7 mV) the THD was simulated at -26 dB, which is comparable to the measured results. Unfortunately, to simplify the routing of the digital control lines in the transconductors, transistors M4 - M7 were not interdigitated in the

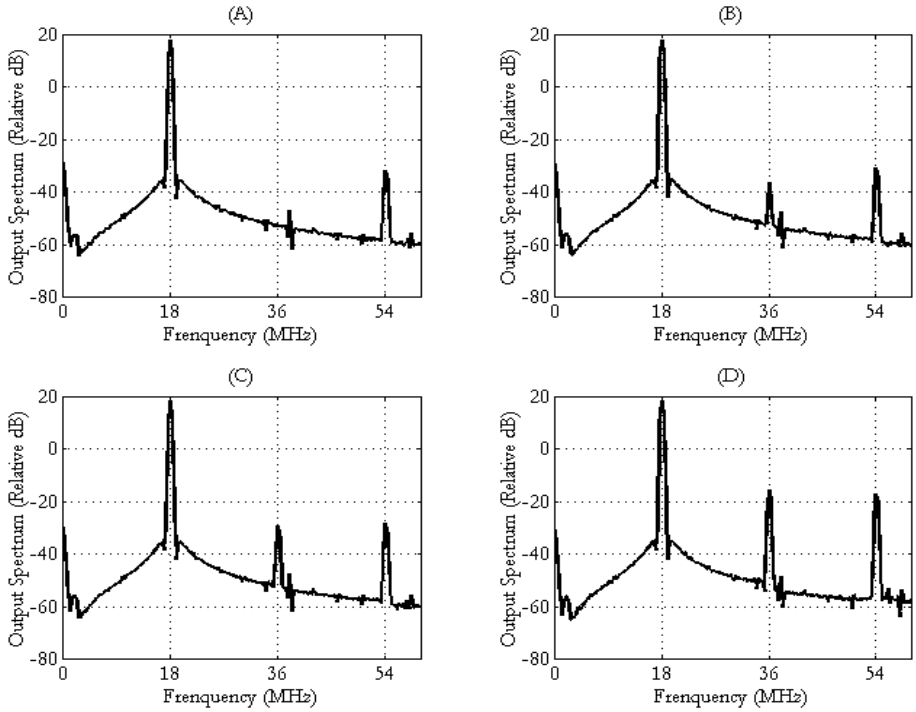


Figure 2.24 Linearity simulations of the 5th order lowpass filter.
With a 125 mVpp differential output at 18 MHz, the threshold voltages of transconductor transistors M4 - M7 were made gaussian random variables with a 3s spread of (A) *0 mV (THD = -48 dB),* (B) *5 mV (THD = -44 dB),* (C) *10 mV (THD = -40 dB),* (D) *20 mV (THD = -26 dB).*

layout. This likely limited the matching between the positive and negative signal paths and, hence, the filter's linearity. If the 3σ spread is decreased to 5 mV (an achievable value with more careful layout), the THD improves to -44 dB.

2.3.4 Noise

The noise spectrum of the reference and filter paths were both measured using a spectrum analyzer (Fig. 2.25) with the filter again programmed to the medium bandwidth setting in Fig. 2.17A. Integrating the noise spectrum over the frequency range of interest (1 - 250 MHz), the noise power can be computed for each path.

$$N_{\text{ref}} = 3.55 \cdot 10^{-6} \text{ mW} \tag{2.4}$$

$$N_{\text{total}} = 9.03 \cdot 10^{-6} \text{ mW} \tag{2.5}$$

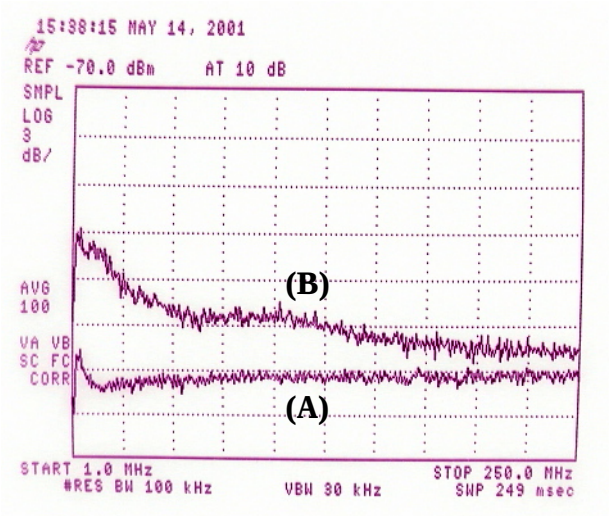


Figure 2.25 Noise spectrum of (A) the reference and (B) the signal paths.

The noise in the filter path, N_{total}, includes the noise introduced by the filter, the output buffers, and the test equipment. Assuming the noise generated by the filter is independent of the noise introduced elsewhere in the system, the filter noise can be computed as follows,

$$\begin{aligned} N_{\text{filter}} &= N_{\text{total}} - N_{\text{ref}} \\ &= 5.48 \cdot 10^{-6}\ \text{mW} \end{aligned} \tag{2.6}$$

This is the noise power at the spectrum analyzer, but assuming all of the attenuation in the reference signal path occurs at the output buffer, it can be referred to the filter output as $5.48 \cdot 10^{-5}$ mW or 1.656 mVrms. This is the same as the power in the harmonic distortion in Fig. 2.21. For this measurement, the filter was programmed for a dc gain of 5 dB. The noise power would likely be less for transfer functions with 0 dB dc gain. These measurements correspond to an input referred noise density of approximately 120 $\text{nV}/(\sqrt{\text{Hz}})$ in the passband.

2.3.5 Spurious-Free Dynamic Range

Fig. 2.26 plots the total harmonic distortion at the filter output as the signal amplitude increases. At an output power of -12.6 dBm, the combined power of the output harmonics is the same as the total noise power, -42.6 dBm. Therefore, the spurious-free dynamic range (SFDR) of the filter is -12.6 dBm - 42.6 dBm = 30.0 dB.

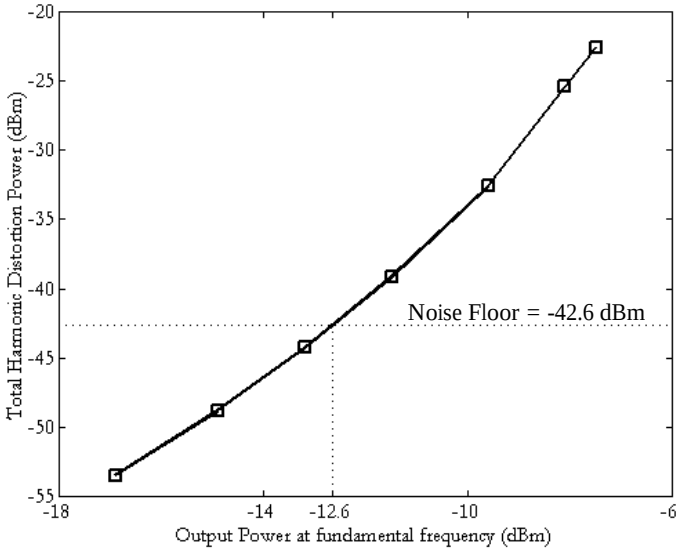


Figure 2.26 Total harmonic distortion power of the 5th order filter referenced to the filter output.
Testing done with a 18 MHz input

2.4 Fine Tuning the Transconductances

In many applications, 5 bits of control for the transconductors is sufficient. However, if more resolution is required in setting the filter coefficients, it is possible to exercise fine control over the transconductances via the gate voltage, V_{cntrl}. V_{cntrl} is proportional to the conductance of each triode transistor in the binary-weighted array of Figure 2.2 and, hence, the overall transconductance. Therefore, a 4-bit DAC with inputs b_0-b_3 is used to set this voltage in each transconductor separately.

Although the voltage V_{cntrl} is useful for fine tuning, it can not be used to vary G_m over more than about one octave. The voltage V_{cntrl} must be kept high enough to prevent M3 from entering the active region and introducing nonlinearities. On the other hand, switching binary-weighted transistors can not be used for fine control since that would require a very large transistor array and the associated parasitic drain and source junction capacitances would limit speed.

2.4.1 4-Bit V_{cntrl} DAC

A 4-bit DAC is used to generate V_{cntrl} as shown in Fig. 2.27. Transistor Mt is operated in triode and is a scaled version of triode transistor M3 in Fig. 2.1C. Opamps O1 and O2 maintain a constant drain-source voltage $\Delta V \approx 100$ mV across Mt. Meanwhile a dc current, I_a, and the 4-bit DAC output current, I_{DAC}, are summed and forced through Mt. As a result, triode transistor Mt has a drain-source conductance of,

$$G_{ds,t} = \frac{I_{DAC} + I_a}{\Delta V} = \mu_n C_{ox}(W/L)_t V_{gs,t} \tag{2.7}$$

A more detailed schematic is shown in Figure 2.28. The test voltage ΔV is the product of a reference current I_{ref} and a resistor R1. (On this chip, R1 is a triode transistor with the gate voltage set externally for tuning.) The DAC is an R-2R ladder with unit current sources, I_b. As the DAC input bits b_0-b_3 are swept over the range 0000 to 1111, I_{DAC} varies over the range 0 to $1.875 \cdot I_b$. When $I_{DAC} = 0$, $G_{ds,t} = I_a / \Delta V$ resulting in the minimum possible value of V_{cntrl}. Since the transconductor linearity is worst for small values of V_{cntrl}, the current I_a must be chosen large enough to provide acceptable linearity performance.

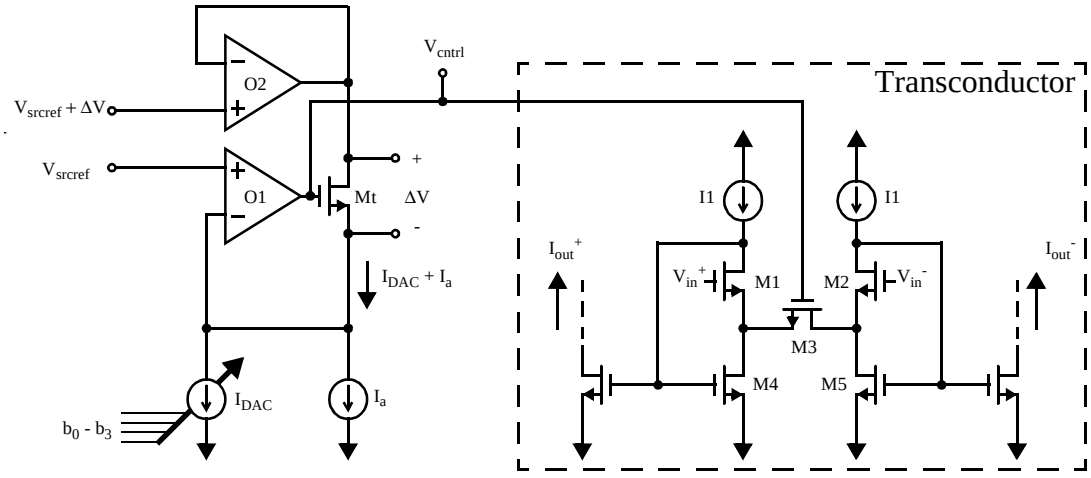


Figure 2.27 Using a four-bit DAC to program the gate control voltage, V_{cntrl}.

The opamps used are simple low-speed 2-stage p-channel input opamps with a nominal current consumption of 300 μA. Opamp O2 has a source-follower output stage which consumes an additional 250 μA.

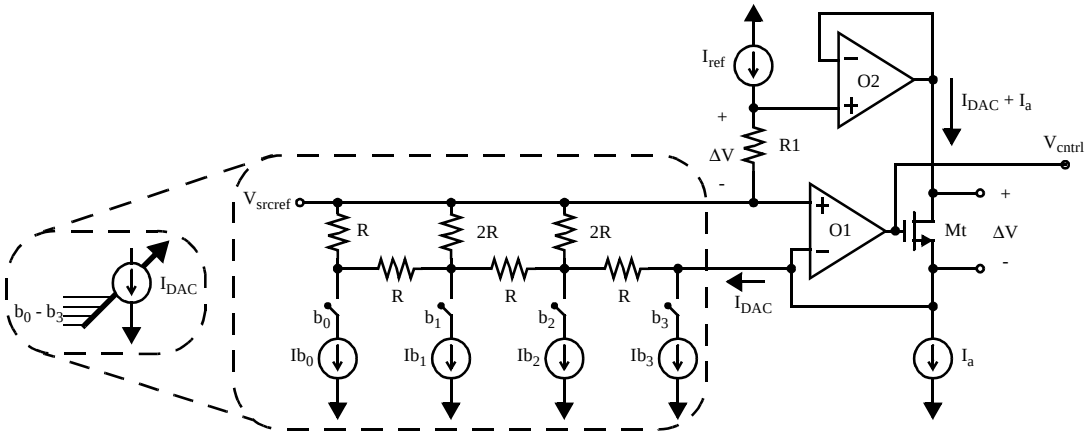


Figure 2.28 Details of the 4-bit DAC.

Opamp O1 keeps the source of Mt at V_{srcref}. If the source voltage of M3 is also V_{srcref}, then its drain-source conductance will be a scaled duplicate of $G_{ds,t}$. If, on the other hand, the dc voltage at the source of M3 differs from V_{srcref}, there will be an offset between $G_{ds,t}$ and $G_{ds,3}$. V_{srcref} is produced by a bias circuit which is a scaled copy of a transconductor half-circuit (Figure 2.29). The bias circuit uses the common mode reference voltage, V_{cmref}, to produce V_{srcref}. Therefore, all common mode levels must be accurately set to V_{cmref}. Any offset in the common mode feedback loop would affect the value of $V_{gs,3}$ and, hence, the filter time constants.

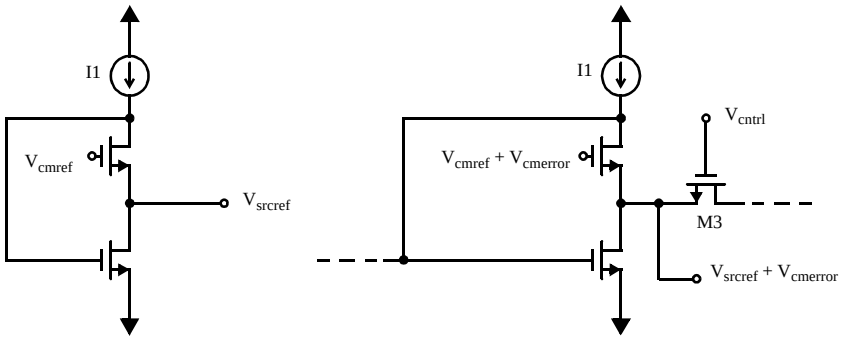


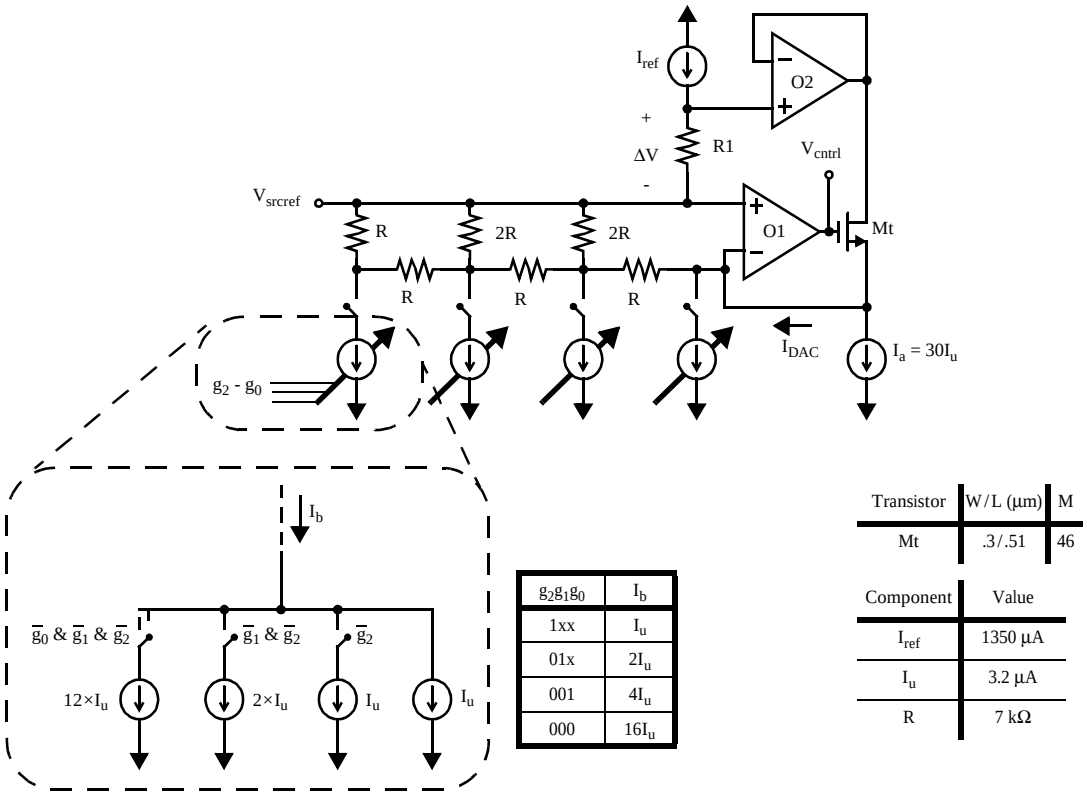
Figure 2.29 A systematic offset error in the CMFB.

$V_{cmerror}$ effects the gate-source voltage of M3, $V_{gs,3} = V_{cntrl} - V_{srcref} + V_{cmerror}$, which changes the degeneration conductance, $G_{ds,3}$ and, hence, the cell's transconductance.

2.4.2 Hysteresis

Whenever separate mechanisms are used for coarse discrete control and fine tuning of the same adaptive parameter, it is desirable to introduce hysteresis at transitions of the coarse control word in order to ensure that such transitions occur infrequently in steady state [14]. In this design, b_3 - b_0 are used for fine tuning via V_{cntrl} while g_2 - g_{-2} provide coarse discrete control via the binary weighted transistor array. Hysteresis is introduced by insuring that the fine tuning range is always greater than one LSB of the coarse control.

Unfortunately, hysteresis also creates some redundancy over the tuning range. When there is too much hysteresis, fine control is lost. Therefore, I_b was made a function of the coarse control signals g_2 - g_0 as shown in Fig. 2.30.



$g_2g_1g_0$	I_b
1xx	I_u
01x	$2I_u$
001	$4I_u$
000	$16I_u$

Transistor	W/L (μm)	M
Mt	.3/.51	46

Component	Value
I_{ref}	1350 μA
I_u	3.2 μA
R	7 kΩ

Figure 2.30 Implementation of DAC current sources in terms of unit current sources.

Each unit current source I_u is a 0.3/1.6 mm NMOS device.

2.4.3 Results

The fine control DAC was tested in two ways:

- by probing the dc voltage V_{cntrl} directly (Fig. 2.31).
- by measuring the low-frequency gain of the first order filter test structure (Fig. 2.32).

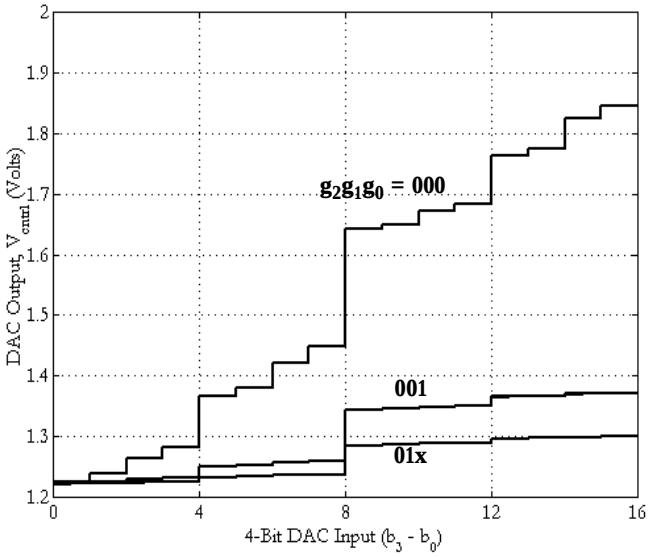


Figure 2.31 Probed DAC output voltage vs. 4-bit input word for different coarse control words.

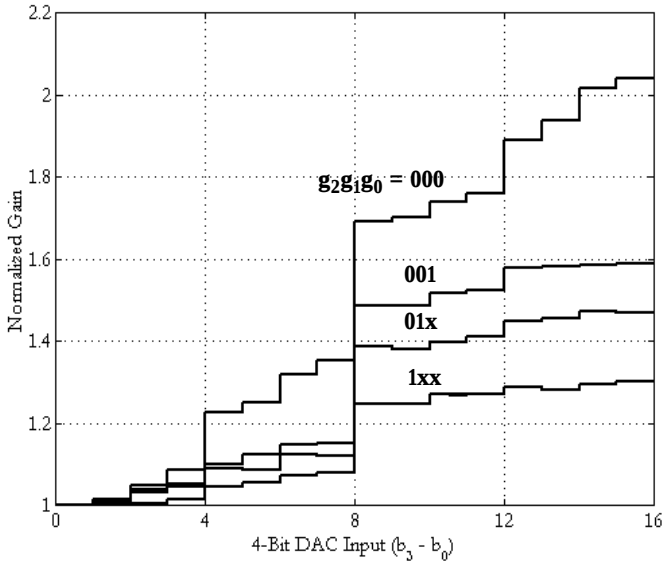


Figure 2.32 Normalized gain of the 1st order test structure vs. 4-bit input word for different coarse control words.

Both tests indicated a strong discontinuity in the D/A characteristic when the MSB is activated. Another significant discontinuity appears whenever the 2nd MSB is toggled.

In order for the D/A characteristic of the DAC in Fig. 2.28 to be linear, all of the current sources must have the same output current: $I_0 = I_1 = I_2 = I_3$. However, this was not the case because the NMOS transistors in current sources I_0 - I_3 were not operating in the active mode. Therefore, the current sources had a low output impedance and voltage drops along the R-2R ladder caused the currents to vary: $Ib_0 < Ib_1 < Ib_2 < Ib_3$.

Three possible solutions to this problem are:

1. Use PMOS current sources instead of NMOS current sources. The PMOS transistors would remain active ensuring high output impedances for all of the current sources. This would require a significant redesign of the entire DAC.
2. Increase the potential at V_{srcref}. This would also require the common mode reference potential, V_{cmref}, to be increased everywhere in the chip to avoid the offset problem described in Fig. 2.29.
3. Decrease the value of all resistors in the R-2R ladder. This would decrease the IR drops in the resistor ladder, thereby putting all internal nodes at roughly the same potential and mitigating the effect of the current sources' finite output impedance.

Solutions #2 and #3 were verified using simulations in Fig. 2.33. The problem was not noticed during design of the DAC because the value of V_{srcref} was decreased after the DAC design was complete.

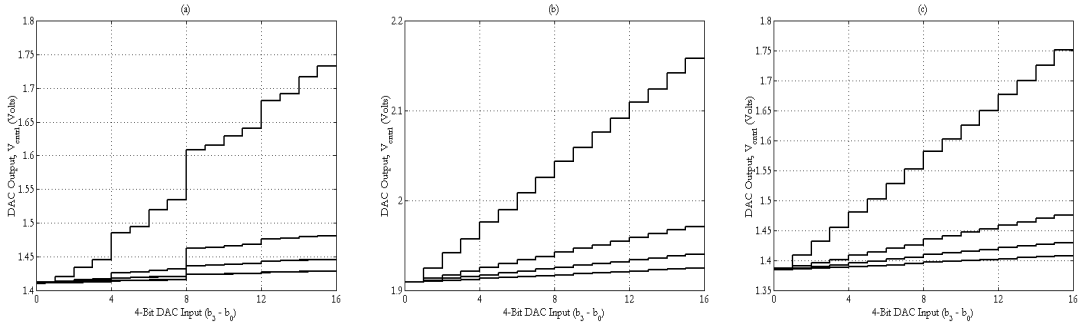


Figure 2.33 Simulation results of the DAC subcircuit.

(A) *design as in prototype* (B) *with* V_{srcref} *increased by 350 mV* (C) *with R decreased from 7 kW to 0.5 kW.*

2.5 Power Consumption

Programmability is generally accompanied by increased power consumption since some overdesign is necessary to insure proper operation over a wide range of conditions. This design was no exception. The total power consumption for the entire prototype IC was simulated at 60 deg. C to be 93.1 mA from a 2.5 V supply or 233 mW. A current consumption of 100 mA for the entire prototype IC was measured in the lab, providing reasonable agreement with the simulations. Much of the current was consumed by test structures and output buffers (Fig. 2.34). The current consumed by the 5th order filter alone, including ten fine control DACs, bias circuitry, etc., is 52.3 mA or 130 mW. However, some obvious avenues for improvement exist.

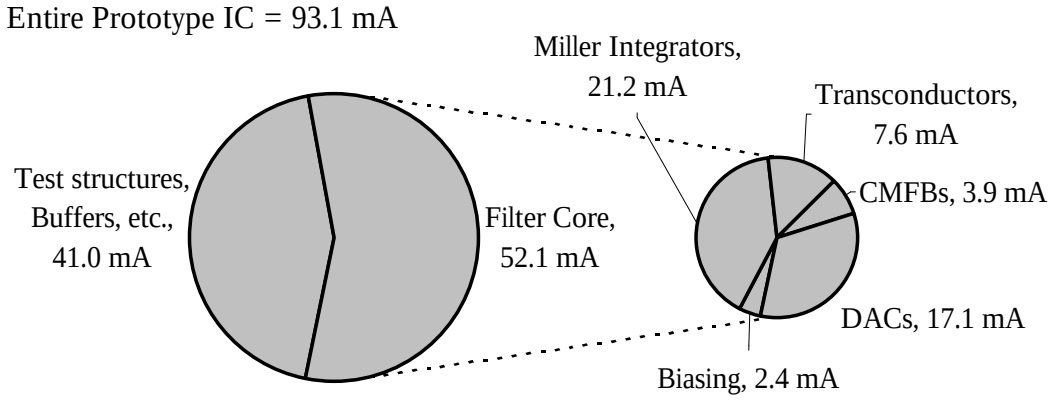


Figure 2.34 Current consumption of the prototype IC by functional block.
These results were simulated at 60 deg. C. A current consumption of 100 mA was measured in the lab.

The fine control DACs could be eliminated. This would provide a power savings of 17.1 mA resulting in a 5th order filter consuming 35.2 mA or 88 mW with 5-bit programmable coefficients, still sufficient programmability for most applications. This requires no redesign other than connecting the control voltage nodes, V_{cntrl}, to a fixed potential. In fact, for most of the testing, only 5 bits of programmability were used and the DAC input was held constant at $b_3b_2b_1b_0 = 0000$. Other improvements in power consumption would require more significant redesign. The biggest improvement would be obtained by replacing the miller integrators with simple capacitor integrators, which

might also simplify the CMFB loop and enable higher speed operation. In this case, the power savings would be approximately an additional 20 mA or 50 mW.

2.6 Conclusions

Techniques for implementing a 5th order digitally programmable filter in a 0.25 μm CMOS process were described for analog adaptive filtering applications. The measured results from a prototype implementation of the filter are summarized in Table 2.1.

An orthonormal structure was chosen to provide a high degree of programmability while maintaining good dynamic range scaling. The mechanism provided for coarse digital gain control suffers from a finite transconductance even when the control word is all zeros. However, this can be overcome by turning off the output current mirror. The fine tuning mechanism described in Section 2.4 could be useful when accurate gain control is required, but its repeated use caused a dramatic increase in circuit size, complexity, and power consumption.

Traditional transconductor and miller integrator circuits were redesigned with only NMOS devices in the signal path to take advantage of their superior speed. A major limitation of transconductor circuits which rely on differential pairs that are source-degenerated by a triode MOS device is that their transconductance and linearity depend upon the input common mode level. Therefore, strongly inverted triode devices should be used to reduce this dependence, or the MOS devices should be replaced by passive resistors. Also, although feedback can reduce nonlinearity in a CMOS transconductor by maintaining a constant current through the input differential pair, the feedback can also cause the input devices to go into triode for large input swings. Whether the net effect of the feedback is to improve or worsen linearity will depend upon the particular application and process technology used.

Although the use of miller integrators offers the potential for better linearity, at low supply voltages it is difficult to get better than 40-50 dB of linearity anyway. Furthermore, the use of a Miller stage can limit speed compared with a passive capacitor integrator. Generally, in mixed-signal systems, low-speed high-accuracy filtering can be easily

Entire Prototype Gm-C Filter IC	
Technology	0.25 μm CMOS
Supply Voltage	2.5 V
Integrated Area	2.5 mm × 1.7 mm = 4.25 mm^2
Power	233 mW (Simulated) 250 mW (Measured)
First Order Filter Test Structure	
Integrated Area	0.25 mm × 0.38 mm = 0.094 mm^2
Power	29.7 mW[a]
THD with 200 mVpp input, 120 mVpp output tone at 8 MHz	-41 dB
5th Order Lowpass Orthonormal Ladder Filter	
Integrated Area	1.25 mm × 0.38 mm = 0.469 mm^2
Power with 9-Bit Programmable Coefficients	130 mW (Simulated)[a]
Power with 5-Bit Programmable Coefficients	87.5 mW (Simulated)[a]
Input-referred Noise Power over 250 MHz BW	1.656 mVrms
Input-referred Noise Spectral Density in the passband	120 $nV/(\sqrt{Hz})$
THD with 125 mVpp input, 170 mVpp output tone at $f_{3\,dB}/2$	-27.7 dB[b]
SFDR with input tone at $f_{3\,dB}/2$	30 dB[c]

Table 2.1 Summary of prototype filter IC measurements.

a. Includes shared bias circuitry.

b. Corresponds to measurement made in Fig. 2.21. Measurements for different transfer functions varied between -20 dB to -30 dB.

c. Corresponds to input amplitude of 105 mVpp and output amplitude of 145 mVpp.

performed digitally and the extra analog complexity and current consumption which Miller integrators imply are undesirable.

Although many useful circuit techniques have been demonstrated, such as the all-NMOS transconductor and Miller integrator and the use of multiple transconductance control mechanisms with hysteresis, the poor linearity measurements limited the design's

relevance to real applications. It was useful primarily as a test vehicle for exploring digital adaptation algorithms for analog filters. For comparison, the design of a 4th order digitally programmable Gm-C filter in a similar process technology (0.25 μm CMOS) was described in [15]. Four bits of programmability were provided for each transconductor by switching binary-scaled differential pair transconductors on and off. Passive capacitors were used as integrators. This approach provided a much simpler and higher speed design, although the circuit used a higher supply voltage (3.3 V) and was limited to butterworth lowpass transfer functions. In [16], an 8th order lowpass Gm-C filter was described in 0.25 μm CMOS with a 2.5 V supply. The transconductors were weakly degenerated differential pairs. Digital programmability was provided using a combination of tail current DACs, binary-ratioed degeneration resistors, and binary-scaled parallel-connected transconductors. In both [15] and [16], no Miller integrators were used yet the linearity was far superior to that reported here (40 - 50 dB THD). However, the filters were restricted to fixed lowpass transfer functions with programmable cutoff frequencies and (in [16]) programmable high-frequency boost. An analog filter with all poles and zeros digitally programmable has only been reported at much lower speeds (up to 8 MHz) in a BiCMOS process [8].

2.7 Appendix - Derivation of Eqn. (2.1)

The low frequency small signal equivalent half-circuit of the transconductor in Fig. 2.1C is shown in Fig. 2.35. The PMOS current sources were cascoded so their output impedance is far greater than all others in the circuit. Therefore, infinite output impedance is assumed to simplify the small signal analysis.

A node equation at v_{s1} gives,

$$g_{m4}v_{d1} + (g_{ds4} + 2g_{ds3} + g_{s1})v_{s1} = 0 \tag{2.8}$$

$$\Rightarrow v_{s1} = -\frac{g_{m4}}{g_{ds4} + 2g_{ds3} + g_{s1}}v_{d1} \tag{2.9}$$

A node equation at v_{d1} gives,

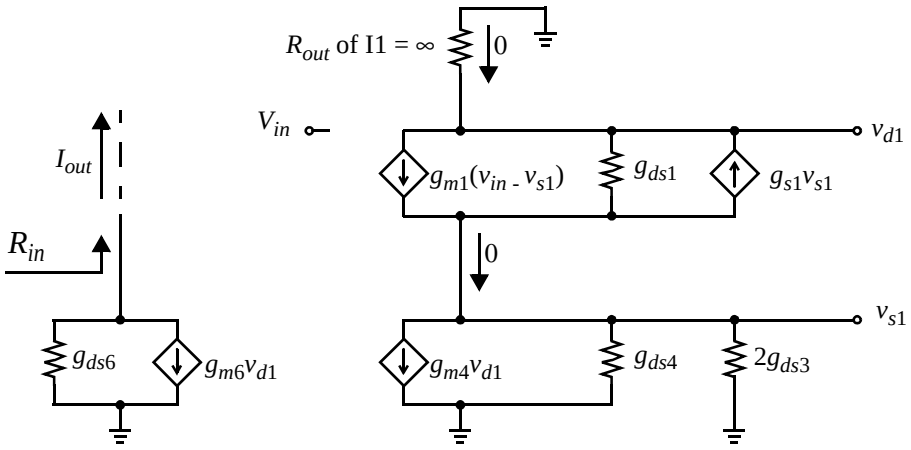


Figure 2.35 Small-signal equivalent half-circuit of the transconductor.

$$g_{m1}(V_{in} - v_{s1}) + g_{ds1}(v_{d1} - v_{s1}) - g_{s1}v_{s1} = 0 \tag{2.10}$$

Substituting Eqn. (2.9) into Eqn. (2.10) gives,

$$g_{m1}V_{in} + \frac{(g_{m1} + g_{ds1} + g_{s1})g_{m4}}{g_{ds4} + 2g_{ds3} + g_{s1}}v_{d1} + g_{ds1}v_{d1} = 0 \tag{2.11}$$

Making the following approximation,

$$\frac{(g_{m1} + g_{ds1} + g_{s1})g_{m4}}{g_{ds4} + 2g_{ds3} + g_{s1}} \cong \frac{g_{m1}g_{m4}}{g_{ds4} + 2g_{ds3} + g_{s1}} \gg g_{ds1} \tag{2.12}$$

one can now write, from Eqn. (2.11),

$$\begin{aligned} & g_{m1}V_{in} + \frac{g_{m1}g_{m4}}{g_{ds4} + 2g_{ds3} + g_{s1}}v_{d1} \cong 0 \\ \Rightarrow\ & g_{m4}v_{d1} = -(g_{ds4} + 2g_{ds3} + g_{s1})V_{in} \end{aligned} \tag{2.13}$$

Assuming the subsequent stage has a relatively low input resistance (i.e. $R_{in} \ll 1/g_{ds6}$) the output current is simply given by,

$$\begin{aligned} I_{out} &= -g_{m6}v_{d1} \\ &= \frac{g_{m6}}{g_{m4}}(g_{ds4} + 2g_{ds3} + g_{s1})V_{in} \\ &= M(2g_{ds3} + g_{ds4} + g_{s1})V_{in} \end{aligned} \tag{2.14}$$

This is the result presented in Eqn. (2.1).

2.8 Appendix - Digital Circuitry

Simple digital circuitry was included on the prototype to program the filter coefficients. The circuitry includes a register file for storing the following data:

- 11 bits (b_3-b_0, g_2-g_{-2}, and d_1-d_0) for each of the 5 filter feedback parameters: 55 bits total
- 12 bits (b_3-b_0, g_2-g_{-2}, d_1-d_0, plus one sign bit) for each of the 5 filter feed-in parameters: 60 bits total
- 2 bits (c_1-c_0) for tuning the integrating capacitors

The result is a total of 117 bits. The register file was organized as 16 words x 12 bits per word, so each write operation requires 4 address bits plus 12 data bits. Because a 24-pin package was used, only 6 pins were allocated to the digital circuitry, two of which were used for digital power supplies. The remaining 4 signals provide a serial interface through which an address and 12 bits of data can be written. No facility for reading from the register file (other than probe pads) was provided.

The 4 serial interface signals are:

- **Clock** - (rising edge triggered) a clock is required to serially shift data into the chip
- **Reset** - (active low) this signal clears the contents of all flip-flops in the register file
- **Start** - (active high) a write operation is initiated by activating the Start signal for one clock cycle
- **Data** - provides the address and data to be written into the register file

After a write operation is initiated by the “Start” signal, the next bit clocked into the “Data” pin must be a 1. The proceeding 4 bits are used as the address in the register file (a_3 - a_0) and the remaining 12 bits are the data (r_{11} - r_0). A sample timing diagram is provided in Fig. 2.36. Table 2.2 is an address map for the register file.

The serial interface was implemented using standard cells which were manually placed and routed. Since the interface was not speed-critical, no effort was made to optimize the netlist or layout. A schematic of the digital circuitry is provided in Fig. 2.37.

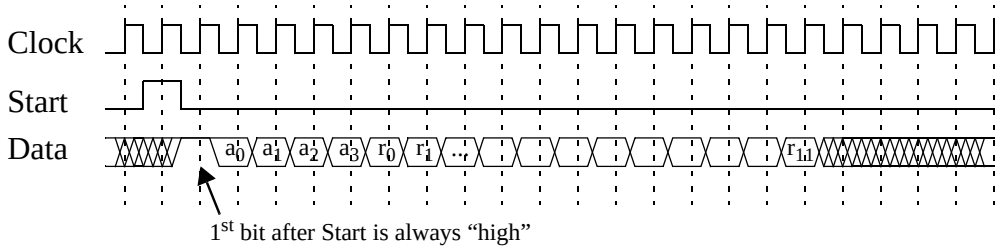


Figure 2.36 Sample timing diagram for serial digital interface.

Binary Address(es)	Register Contents (r_{11} - r_0)
0000 - 0100	Filter feed-in coefficients. r_0: sign bit r_2 - r_1: d_1-d_0 - output current mirror control r_7 - r_3: g_2-g_{-2} - transistor array control bits r_{11} - r_8: b_3-b_0 - fine control DAC
0101 - 0111	Not used
1000 - 1100	Filter feedback coefficients. r_0: Not used r_2 - r_1: d_1-d_0 - output current mirror control r_7 - r_3: g_2-g_{-2} - transistor array control bits r_{11} - r_8: b_3-b_0 - fine control DAC
1101 - 1110	Not used
1111	Capacitance tuning: r_1 - r_0: c_1-c_0 r_{11} - r2: Not used

Table 2.2 Digital register file address map.

2.9 References

[1] D. A. Johns, W. M. Snelgrove, and A. S. Sedra, "Continuous-Time LMS Adaptive Recursive Filters," *IEEE Trans. Circuits Syst.*, Vol. 38, No. 7, pp. 769-777, July 1991.

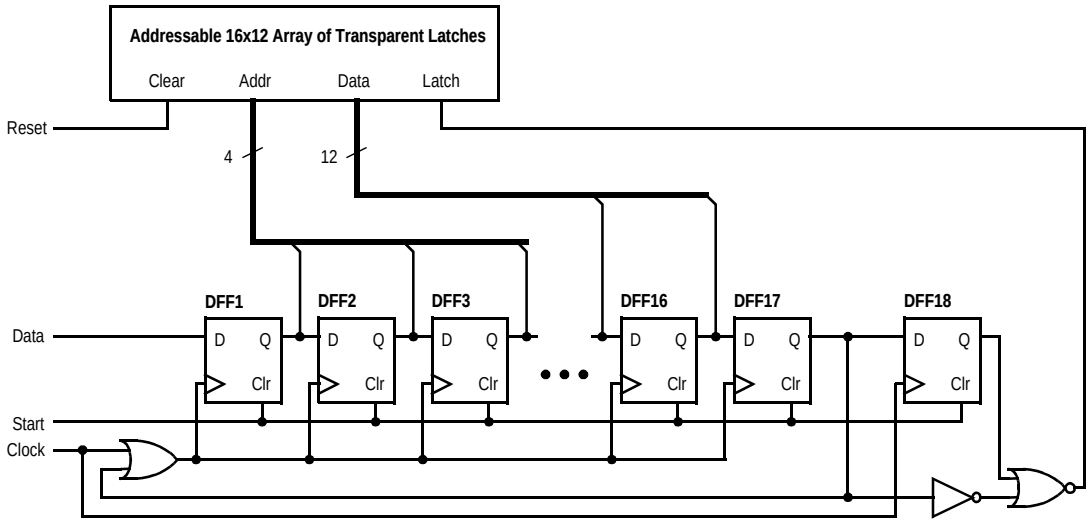


Figure 2.37 Schematic diagram for a 4-pin digital serial interface.

[2] A. Shoval, D. A. Johns, and W. M. Snelgrove, "Comparison of DC Offset Effects in Four LMS Adaptive Algorithms," *IEEE Trans. Circuits Syst. II*, Vol. 43, No. 3, pp. 176-185, March 1995.

[3] T. Kwan and K. Martin, "An Adaptive Analog Continuous-Time CMOS Biquadratic Filter," *IEEE Journal of Solid-State Circuits*, Vol. 26, No. 6, pp. 859-867, June 1991.

[4] J. E. C. Brown, P. J. Hurst, B. C. Rothenberg, and S. H. Lewis, "A CMOS Adaptive Continuous-Time Forward Equalizer, LPF, and RAM-DFE for Magnetic Recording," *IEEE Journal of Solid-State Circuits*, Vol. 34, No. 2, pp. 162-169, Feb. 1999.

[5] P. K. D. Pai, A. D. Brewster, and A. Abidi, “Analog Front-End Architectures for High-Speed PRML Magnetic Read Channels”, *IEEE Trans. Magnetics*, pp. 1103-1108, March 1995.

[6] F. Pecourt, J. Hauptmann, and A. Tenen, “An Integrated Adaptive Analog Balancing Hybrid for Use in (A)DSL Modems,” *Int. Solid-State Circuits Conf.*, pp. 252-253, San Francisco, February 1999.

[7] M. G. Larimore, J. R. Treichler, and C. R. Johnson, “SHARF: An Algorithm for Adapting IIR Digital Filters,” *IEEE Trans. Acoust. Speech Signal Process.*, vol. ASSP-28, p. 428-440, Aug. 1980.

[8] A. Hematy and G. W. Roberts, “A fully-programmable analog log-domain filter circuit”, *IEEE Int. Symp. Circuits and Syst.*, vol. 1, p. 309-312, June 1998.

[9] S. Pavan and Y. Tsividis, “Time-Scaled Electrical Networks — Properties and Applications in the Design of Programmable Analog Filters,” *IEEE Trans. Circuits Syst. II*, vol. 47, pp. 161-165, Feb. 2000.

[10] D. A. Johns, W. M. Snelgrove and A. S. Sedra, "Orthonormal Ladder Filters," *IEEE Trans. Circuits Syst.*, Vol. 36, No. 3, pp. 337-343, March 1989.

[11] D. R. Welland, et. al. “A Digital Read/Write Channel with EEPR4 Detection,” *IEEE Int. Solid-State Circuits Conf.*, pp. 276-277, San Francisco, February 1994.

[12] J. Cheng and D. A. Johns, “A 100 MHz Partial Analog Adaptive Equalizer for use in Wired Data Transmission,” *European Solid State Circuits Conference*, Sept. 1999.

[13] M. Padmanabhan and K. Martin, “A CMOS Analog Multi-Sinusoidal Phase Locked-Loop,” *IEEE Journal of Solid-State Circuits*, Vol. 29, No. 9, pp. 1046-1057, September 1994.

[14] K. Kamra, *Cable equalization using adaptive analog filters*, M. A. Sc. Thesis, University of Toronto, 1996.

[15] S. Pavan, Y. Tsividis, and K. Nagaraj, “Widely Programmable High-Frequency Continuous-Time Filters in Digital CMOS Technology,” *IEEE Journal of Solid-State Circuits*, Vol. 35, No. 4, pp. 503-511, April 2000.

[16] G. Bollati, S. Marchese, M. Demicheli, and R. Castello, “An Eigth-Order CMOS Low-Pass Filter with 30-120 MHz Tuning Range and Programmable Boost,” *IEEE Journal of Solid-State Circuits*, Vol. 36, No. 7, pp. 1056-1066, July 2001.

Chapter 3

The Dithered Linear Search Algorithm

3.1 Introduction

At low speeds, adaptive filtering is easily and efficiently performed using digital circuits. Analog filters are preferable at high speeds when low power consumption, small integrated area, and moderate linearity are required. The LMS algorithm is currently the most popular technique for digital filter adaptation. Unfortunately, implementation of the LMS algorithm in analog adaptive filters is challenging. Dc offsets on the analog signals can prevent accurate convergence of the LMS algorithm [1], [2]. Also, significant additional analog hardware may be required to obtain the gradient signals required by the LMS algorithm [3].

As a result, numerous heuristic algorithms have been developed for specific applications based upon adjusting analog filter parameters to satisfy some desirable and easily observed condition [4], [5], [6]. Although relatively simple, both conceptually and in terms of their hardware implementation, these approaches are tailored to specific applications and filter structures. Therefore, they are not easily generalized to new applications, particularly those requiring high order filters with several adapted parameters.

In this chapter, a technique for analog filter adaptation is discussed which is general, has a straightforward hardware implementation, and is robust with respect to dc offsets. The dithered linear search (DLS) technique does not require access to the filter's internal

states and little additional analog hardware is required. All filter parameters may be adapted simultaneously and independently of one another.

In Section 3.2 of this chapter, a theoretical framework for the problem of filter adaptation is established and two gradient descent adaptation algorithms are presented. In Section 3.3, the DLS algorithm is introduced. Theoretical results for the misadjustment and convergence rates are derived in Section 3.4. Behavioral simulation results are then presented in Section 3.5 for both a tapped delay line and a continuous time filter. Different possible dither signals are considered in Section 3.6, and dc offset effects are discussed in Section 3.7. In Section 3.8, subsampling of the output error signal is discussed, and in Section 3.9 the effect of quantization of the filter parameters is analyzed. Finally, in Section 3.10, experimental results of the algorithm applied to a continuous time integrated analog filter are presented.

3.2 Background

Using the general framework for filter adaptation established in Section 1.3.1, recall that gradient descent optimizers proceed by updating the filter's parameters iteratively in a direction opposite the gradient $\nabla_{\boldsymbol{p}(k)}\varepsilon(\boldsymbol{p}(k))$. The iterative update rule is repeated here,

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) - \mu \cdot \nabla_{\boldsymbol{p}(k)}\varepsilon(\boldsymbol{p}(k)) \tag{3.1}$$

where μ is a parameter controlling the rate of adaptation. The only problem is how to obtain the gradient $\nabla_{\boldsymbol{p}(k)}\varepsilon(\boldsymbol{p}(k))$, or $\nabla_{\boldsymbol{p}}\varepsilon$ for short. Generally the exact value of $\nabla_{\boldsymbol{p}}\varepsilon$ can not be determined, so an unbiased estimate of the gradient is used instead,

$$\hat{\nabla}_{\boldsymbol{p}}\varepsilon = \left[\frac{\hat{\partial}\varepsilon}{\partial p_1}\ \frac{\hat{\partial}\varepsilon}{\partial p_2}\ \cdots\ \frac{\hat{\partial}\varepsilon}{\partial p_N}\right]^T \tag{3.2}$$

This estimate is substituted into Eqn. (3.1) resulting in,

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) - \mu \cdot \hat{\nabla}_{\boldsymbol{p}}\varepsilon \tag{3.3}$$

Different approaches to obtaining the gradient estimates define different gradient descent algorithms. The remainder of this section will describe how gradient estimates are obtained for the LMS and differential steepest descent (DSD) algorithms.

3.2.1 The Least Mean Square (LMS) Algorithm

The LMS algorithm uses the gradient of the instantaneous squared error, $e(k)^2$, as an unbiased noisy estimate of the actual gradient,

$$\begin{aligned}\hat{\nabla}_{\boldsymbol{p}}\varepsilon &= \nabla_{\boldsymbol{p}}(e(k)^2) \\ &= 2e(k)\nabla_{\boldsymbol{p}}(d(k) - y(k)) \\ &= -2e(k)\nabla_{\boldsymbol{p}}y(k) \\ &= -2e(k)\,\mathrm{f}(k)\end{aligned} \tag{3.4}$$

where $\mathrm{f}(k)$ is defined as the gradient $\nabla_{\boldsymbol{p}}y(k)$. The LMS update rule is then derived by substituting Eqn. (3.4) into Eqn. (3.3),

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) - 2\mu e(k)\,\mathrm{f}(k) \tag{3.5}$$

Unfortunately, obtaining the gradient signals $\mathrm{f}(k)$ for analog filters can require considerable additional hardware.

3.2.2 The Differential Steepest Descent Algorithm

The differential steepest descent (DSD) algorithm makes a direct measurement of each gradient component by perturbing the parameters one at a time symmetrically around their current values and measuring the resulting change in the error function. The derivative is approximated by a finite difference expression,[1]

$$\frac{\partial\varepsilon}{\partial p_i} \approx \frac{\varepsilon(\begin{bmatrix}p_1 & p_2 & \ldots & p_i + \Delta & \ldots\end{bmatrix}^T) - \varepsilon(\begin{bmatrix}p_1 & p_2 & \ldots & p_i - \Delta & \ldots\end{bmatrix}^T)}{2\Delta} \tag{3.6}$$

In order to use Eqn. (3.6), ε is estimated first by operating the filter with $\boldsymbol{p} = \begin{bmatrix}p_1 & p_2 & \ldots & p_i + \Delta & \ldots\end{bmatrix}^T$ and averaging $e^2(k)$ over $L/2$ data samples, then setting $\boldsymbol{p} = \begin{bmatrix}p_1 & p_2 & \ldots & p_i - \Delta & \ldots\end{bmatrix}^T$ and averaging $e^2(k)$ over the following $L/2$ data samples,

$$\frac{\hat{\partial}\varepsilon}{\partial p_i} = \frac{1}{2\Delta}\left[\frac{2}{L}\left(\sum_{k=l}^{l+L/2} e^2(k)\right) - \frac{2}{L}\left(\sum_{k=l+L/2}^{l+L} e^2(k)\right)\right] \tag{3.7}$$

---

1. The approximation in Eqn. (3.6) becomes exact in the limit $\Delta \to 0$ or in the case of a parabolic performance surface such as adaptive linear combiners with an MSE error function [7].

An advantage of the DSD algorithm over LMS adaptation is that it does not require access to the filter's internal states. It is simple and intuitive. However, its hardware implementation is somewhat complicated by the fact that the gradient components must be estimated one at a time. During their discussion of the DSD algorithm, the authors of [8] point out that, "All weights can be simultaneously dithered at individual frequencies and the gradient components obtained by cross correlation." That idea is the basis of the "dithered linear search" algorithm described below.

3.3 The Dithered Linear Search

The dithered linear search (DLS) algorithm[1] is also a gradient descent optimizer. The term "dither" refers to a signal with small amplitude and zero mean that is intentionally injected into a system,

$$p_i'(k) = p_i(k) + \delta_i(k) \tag{3.8}$$

$$E[\delta_i(k)] = 0 \tag{3.9}$$

In Eqn. (3.8), p_i is the algorithm's current estimate of an optimal parameter value whereas p_i' is the parameter value actually applied to the filter including the dither δ_i. Each gradient component is then inferred by correlating changes in the corresponding dither signal to changes in $\varepsilon(\boldsymbol{p}'(k))$,

$$p_i(k+1) = p_i(k) - \frac{\mu}{\sigma^2} \cdot \delta_i(k) \cdot \varepsilon(\boldsymbol{p}'(k)) \tag{3.10}$$

In Eqn. (3.10), the gradient estimate is given by $\delta_i(k) \cdot \varepsilon(\boldsymbol{p}'(k)) / \sigma^2$. The product $\delta_i(k) \cdot \varepsilon(\boldsymbol{p}'(k))$ correlates changes in the output error with the parameter dither to give the sign of the gradient. The product is divided by σ^2 (the variance of $\delta_i(k)$) to normalize the gradient estimate with respect to the dither signal's magnitude. Again, instantaneous estimates can be substituted for $\varepsilon(\boldsymbol{p}'(k))$ in which case Eqn. (3.10) becomes,

1. The name "dithered linear search" was chosen in reference to the other linear search adaptation algorithms (i.e. sequential linear search, random linear search) to which the DLS bears a strong resemblance.

$$p_i(k+1) = p_i(k) - \frac{\mu}{\sigma^2} \cdot \delta_i(k) \cdot e^2(k) \tag{3.11}$$

The gradient estimate is given by,

$$\frac{\hat{\partial \varepsilon}}{\partial p_i} = \frac{1}{\sigma^2} \cdot \delta_i(k) \cdot e^2(k) \tag{3.12}$$

Multiple filter parameters can be adapted simultaneously by applying independent (i.e. uncorrelated) dither signals to all of the parameters.

$$\boldsymbol{p}'(k) = \boldsymbol{p}(k) + \begin{bmatrix} \delta_1(k) \\ . \\ \delta_N(k) \end{bmatrix} = \boldsymbol{p}(k) + \mathrm{d}(k) \tag{3.13}$$

$$E[\delta_i(k)\delta_j(k)] = 0, \quad i \neq j \tag{3.14}$$

The adaptation of any one parameter, p_i, will not be affected by the dither on the others since, over time, only those changes in ε which are correlated with δ_i will influence the parameter p_i.

The DLS algorithm has a straightforward hardware implementation. A block diagram is shown in Fig. 3.1 for one parameter. The filter input u, output y, reference signal d, and error signal e all have their usual characteristics, as described in Section 1.3.1. The algorithm is easily scaled to adapt more parameters using more copies of the same hardware. If the dither signals are binary, the hardware is even simpler. The correlator becomes trivial and the dither signal can be directly connected to the LSB of the parameter control word. Uncorrelated binary dither signals are also easy to generate. Several different possibilities are discussed in Section 3.6.

3.3.1 The Block DLS Algorithm

Rather than continuously updating the parameters via Eqn. (3.11), one may keep them fixed (except for dither) for a block of L consecutive data samples. During this period, derivative estimates are constructed by cross-correlating the dither and squared error signals,

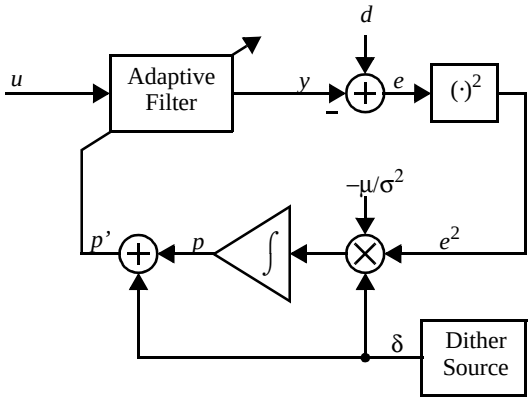


Figure 3.1 Block diagram of the dithered linear search algorithm.
No knowledge of the filter's internal states is required.

$$\frac{\partial \hat{\varepsilon}}{\partial p_i} = \frac{1}{L}\sum_{k=l}^{l+L}\left(\frac{\delta_i(k)\cdot e^2(k)}{\sigma^2}\right) \tag{3.15}$$

After every L samples, the gradient estimates are used to update the parameters via Eqn. (3.3). The standard DLS algorithm corresponds to the limiting case of the block DLS algorithm where $L = 1$. Although the complexity and theoretical performance of the block DLS algorithm is the same as the DLS algorithm, the block adaptation can offer better stability and performance in practice, as we shall see in Section 3.6.

3.4 Theoretical Analysis

3.4.1 Preliminaries

In order to simplify the theoretical analysis of the DLS algorithm and to cast it in a form similar to that used in [8], only adaptive linear combiners are considered. This is common practice in the analysis of adaptive algorithms and the results are also applicable whenever the performance surface is approximately quadratic near its minimum. Also, only binary dither signals are analyzed in detail.

As with all gradient descent optimizers, a necessary condition for stability of the DLS algorithm is,

$$0 < \mu < 1 / \lambda_{max} \tag{3.16}$$

where λ_{max} is the largest eigenvalue of the state correlation matrix, $\boldsymbol{R}$, defined below in terms of the filter's state variables x_i,

$$\boldsymbol{R} = E\begin{bmatrix} x_1x_1 & x_1x_2 & \dots & x_1x_N \\ x_2x_1 & x_2x_2 & \dots & x_2x_N \\ . & . & & . \\ & & \dots & x_Nx_N \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1N} \\ r_{21} & r_{22} & \dots & r_{2N} \\ . & . & & . \\ & & \dots & r_{NN} \end{bmatrix} \tag{3.17}$$

The MSE performance surface of an adaptive linear combiner can be shown to be quadratic,

$$\begin{aligned} \varepsilon &= \varepsilon_{min} + (\boldsymbol{p}^T - \boldsymbol{p}^{*T})\boldsymbol{R}(\boldsymbol{p} - \boldsymbol{p}^*) \\ &= \varepsilon_{min} + \boldsymbol{v}^T\boldsymbol{R}\boldsymbol{v} \end{aligned} \tag{3.18}$$

where $\boldsymbol{v} = \boldsymbol{p} - \boldsymbol{p}^*$. A co-ordinate transformation can be used to separate the independent modes of the adaptive process, $\boldsymbol{w} = \boldsymbol{Q}^{-1}\boldsymbol{v}$. A gradient descent algorithm causes the kth component of $\boldsymbol{w}$ to converge towards its optimal value exponentially with a time constant of,

$$\tau_k = \frac{1}{2\mu\lambda_k} \tag{3.19}$$

The MSE will decay with a kth mode time constant of,

$$\tau_{k_{MSE}} = \frac{1}{4\mu\lambda_k} \tag{3.20}$$

Eqn. (3.20) is written in terms of iterations of the DLS algorithm. For the block DLS algorithm, each iteration corresponds to L data samples. Therefore, in terms of data samples the kth mode time constant is,

$$T_{k_{MSE}} = L\tau_{k_{MSE}} = \frac{L}{4\mu\lambda_k} \tag{3.21}$$

3.4.2 Convergence

In order to ensure that the DLS algorithms converge, it is desirable to show that the gradient estimates in Eqn. (3.12) and Eqn. (3.15) provide unbiased estimates of the true gradient.

$$E[\hat{\nabla}_{\boldsymbol{p}}\varepsilon] = \nabla_{\boldsymbol{p}}\varepsilon, \text{ or equivalently} \tag{3.22}$$

$$E\left[\frac{\hat{\partial}\varepsilon}{\partial p_i}\right] = \frac{\partial\varepsilon}{\partial p_i}, \text{ for all } i \tag{3.23}$$

A proof of Eqn. (3.23) for the DLS with binary dither follows.

For a binary dither signal,

$$\delta_i(k) = \pm\Delta \tag{3.24}$$

Hence,

$$\sigma^2 \equiv E[\delta_i^2(k)] = \Delta^2 \tag{3.25}$$

Taking the expectation of both sides of Eqn. (3.12),

$$\begin{aligned} E\left[\frac{\hat{\partial}\varepsilon}{\partial p_i}\right] &= E\left[\frac{1}{\sigma^2}\cdot\delta_i(k)\cdot e^2(k)\right] \\ &= \frac{1}{\Delta^2}\cdot E[\delta_i(k)\cdot e^2(k)] \end{aligned} \tag{3.26}$$

Also, a dither signal has zero mean.

$$E[\delta_i(k)] = 0 \tag{3.27}$$

$$\Rightarrow \begin{cases} \delta_i(k) = +\Delta, \text{ one-half of the time} \\ \delta_i(k) = -\Delta, \text{ one-half of the time} \end{cases} \tag{3.28}$$

Combining Eqn. (3.28) with Eqn. (3.26) yields,

$$\begin{aligned} E\left[\frac{\hat{\partial}\varepsilon}{\partial p_i}\right] &= \frac{1}{\Delta^2}\left(\left(\frac{1}{2}\cdot E[\delta_i(k)\cdot e^2(k)]\Big|_{\delta_i=+\Delta}\right) + \left(\frac{1}{2}\cdot E[\delta_i(k)\cdot e^2(k)]\Big|_{\delta_i=-\Delta}\right)\right) \\ &= \frac{1}{\Delta^2}\left(\left(\frac{\Delta}{2}E[e^2(k)]\Big|_{\delta_i=+\Delta}\right) + \left(\frac{-\Delta}{2}E[e^2(k)]\Big|_{\delta_i=-\Delta}\right)\right) \\ &= \frac{1}{2\Delta}\left(E[e^2(k)]\Big|_{\delta_i=+\Delta} - E[e^2(k)]\Big|_{\delta_i=-\Delta}\right) \end{aligned} \tag{3.29}$$

It can be shown (see Appendix) that the MSE estimates have the following bias,

$$E[e^2(k)]\Big|_{\delta_i=\pm\Delta} = \varepsilon\left(\left[p_1\ p_2\ \ldots\ p_i\pm\Delta\ \ldots\right]^T\right) + \Delta^2(\mathrm{tr}(\boldsymbol{R}) - r_{ii}) \tag{3.30}$$

Substituting Eqn. (3.30) into Eqn. (3.29) gives,

$$E\left[\frac{\hat{\partial}\varepsilon}{\partial p_i}\right] = \frac{1}{2\Delta}\left(\varepsilon\left(\begin{bmatrix} p_1 \\ p_2 \\ \dots \\ p_i+\Delta \\ \dots \end{bmatrix}\right) + \Delta^2(\mathrm{tr}(\boldsymbol{R}) - r_{ii})\right) - \left(\varepsilon\left(\begin{bmatrix} p_1 \\ p_2 \\ \dots \\ p_i-\Delta \\ \dots \end{bmatrix}\right) + \Delta^2(\mathrm{tr}(\boldsymbol{R}) - r_{ii})\right)$$

$$= \frac{\varepsilon(\begin{bmatrix} p_1 & p_2 & \dots & p_i+\Delta & \dots \end{bmatrix}^T) - \varepsilon(\begin{bmatrix} p_1 & p_2 & \dots & p_i-\Delta & \dots \end{bmatrix}^T)}{2\Delta} \tag{3.31}$$

In order to complete the proof of Eqn. (3.23), it is required that,

$$\frac{\varepsilon(\begin{bmatrix} p_1 & p_2 & \dots & p_i+\Delta & \dots \end{bmatrix}^T) - \varepsilon(\begin{bmatrix} p_1 & p_2 & \dots & p_i-\Delta & \dots \end{bmatrix}^T)}{2\Delta} = \frac{\partial\varepsilon}{\partial p_i} \tag{3.32}$$

Eqn. (3.32) is true by definition of the gradient in the limit $\Delta \to 0$. Eqn. (3.32) is also true for any value of Δ when the error function ε is a quadratic function of $\boldsymbol{p}$, as is the case here since an adaptive linear combiner is assumed. Therefore, the DLS algorithm provides unbiased gradient estimates.

The proof for the block DLS algorithm is similar. If one assumes that $\delta_i(k) = +\Delta$ for $L/2$ samples and $\delta_i(k) = -\Delta$ for $L/2$ samples within each block, then take the expectation of Eqn. (3.15),

$$E\left[\frac{\hat{\partial}\varepsilon}{\partial p_i}\right] = E\left[\frac{1}{L}\sum_{k=l}^{l+L}\left(\frac{\delta_i(k)\cdot e^2(k)}{\sigma^2}\right)\right]$$

$$= \frac{1}{L\Delta^2}\left(E\left[\left(\sum_{\substack{k=l \\ \delta_i=+\Delta}}^{l+L} \delta_i(k)\cdot e^2(k)\right) + \left(\sum_{\substack{k=l \\ \delta_i=-\Delta}}^{l+L} \delta_i(k)\cdot e^2(k)\right)\right]\right)$$

$$= \frac{1}{L\Delta^2}\left((+\Delta)\cdot\frac{L}{2}E[e^2(k)]\Big|_{\delta_i=+\Delta} + (-\Delta)\cdot\frac{L}{2}E[e^2(k)]\Big|_{\delta_i=-\Delta}\right)$$

$$= \frac{1}{2\Delta}(E[e^2(k)]\Big|_{\delta_i=+\Delta} - E[e^2(k)]\Big|_{\delta_i=-\Delta}) \tag{3.33}$$

This result is the same as Eqn. (3.29). From here, the proof is identical to the proof for the standard DLS algorithm.

Eqn. (3.31) is a finite difference gradient estimate, just like the one used for the DSD algorithm in Eqn. (3.6). The similarity is not surprising since the DSD algorithm can be considered a special case of the DLS algorithm where the parameters are dithered one at a time.

Since the gradient estimates are unbiased, as they are averaged over a longer time they approach the true value of the gradient resulting in a perfect gradient descent optimizer. This limit is approached by decreasing μ. So, the DLS is guaranteed stable as long as μ is taken "small enough". The bounds imposed by Eqn. (3.16) apply to any gradient descent optimizer, so they are necessary but not sufficient conditions on μ to ensure that the DLS algorithm converges to a minimum in $\varepsilon(\boldsymbol{p}(t))$. In general, the range of stable values for μ will depend upon the filter structure, the type of dither signal, and the statistics of the input data (which may not be precisely known *a priori*). Therefore, stability of the adaptive process must be verified via simulation.

3.4.3 Perturbation

Even when the DLS algorithm is in steady state and all parameters have converged to their optimal values, the MSE will be somewhat greater than the minimum MSE because the additive dither causes the filter to operate with sub-optimal parameter values. The excess MSE caused by the dither in steady state is referred to as "perturbation". Perturbation is a phenomenon which occurs in the DLS and DSD algorithms, but not the LMS algorithm where no dither is used. In this section, theoretical expressions for the perturbation will be derived.

The mean squared error, including the effect of dither, is obtained by using $\boldsymbol{p}'(k)$ in place of $\boldsymbol{p}(k)$ in Eqn. (3.18),

$$\varepsilon = \varepsilon_{\min} + (\boldsymbol{p}'(k) - \boldsymbol{p}^*)^T \boldsymbol{R} (\boldsymbol{p}'(k) - \boldsymbol{p}^*) \tag{3.34}$$

After convergence, neglecting misadjustment or any other sources of error such as dc offset effects, $(\boldsymbol{p}(k) = \boldsymbol{p}^*) \Rightarrow (\boldsymbol{p}'(k) - \boldsymbol{p}^* = \mathrm{d}(k))$. Substituting this into Eqn. (3.34),

$$\varepsilon = \varepsilon_{\min} + \mathrm{d}^T(k)\boldsymbol{R}\mathrm{d}(k) \tag{3.35}$$

Taking the expected value of both sides over time gives the steady-state MSE,

$$\begin{aligned}\varepsilon_{SS} &= \varepsilon_{\min} + E[\mathrm{d}^T(k)\boldsymbol{R}\mathrm{d}(k)] \\ &= \varepsilon_{\min} + \sum_i r_{ii}E[\delta_i^2(k)]\end{aligned} \tag{3.36}$$

The second line of Eqn. (3.36) was obtained by cancelling all of the other terms in the product $\mathrm{d}^T(k)\boldsymbol{R}\mathrm{d}(k)$ due to the independence of the dither signals: $E[\delta_i(k)\delta_j(k)] = 0$, $i \neq j$. For binary dither where $\delta_i = \pm\Delta$,

$$\varepsilon_{SS} = \varepsilon_{\min} + \Delta^2\sum_i r_{ii} = \varepsilon_{\min} + \Delta^2 \cdot \mathrm{tr}(\boldsymbol{R}) \tag{3.37}$$

The perturbation, P, is defined as the ratio of the excess MSE over the minimum MSE.

$$P = \frac{\varepsilon_{SS} - \varepsilon_{\min}}{\varepsilon_{\min}} = \frac{\Delta^2}{\varepsilon_{\min}} \cdot \mathrm{tr}(\boldsymbol{R}) \tag{3.38}$$

Using the fact that the trace of a matrix is equal to the sum of its eigenvalues, Eqn. (3.38) can be rewritten in terms of the average eigenvalue of the state correlation matrix, λ_{avg}.

$$P = \frac{N\Delta^2\lambda_{avg}}{\varepsilon_{\min}} \tag{3.39}$$

Eqn. (3.39) indicates that the perturbation is larger using the DLS algorithm than using the DSD algorithm by a factor of N. As shown in Fig. 3.2, this is because all N parameters are dithered simultaneously in the DLS algorithm, instead of one at a time as in the DSD algorithm.

3.4.4 Noise in the Gradient Estimates

As with all gradient descent optimizers, the DLS algorithm relies upon noisy estimates of the gradient to perform adaptation. The difference between the estimated gradient and actual gradient is a vector random variable, $\boldsymbol{n}$.

$$\nabla_{\boldsymbol{p}}\varepsilon - \hat{\nabla}_{\boldsymbol{p}}\varepsilon = \boldsymbol{n} \tag{3.40}$$

The variance of $\boldsymbol{n}$ measures the accuracy of the gradient estimates for different gradient descent algorithms.

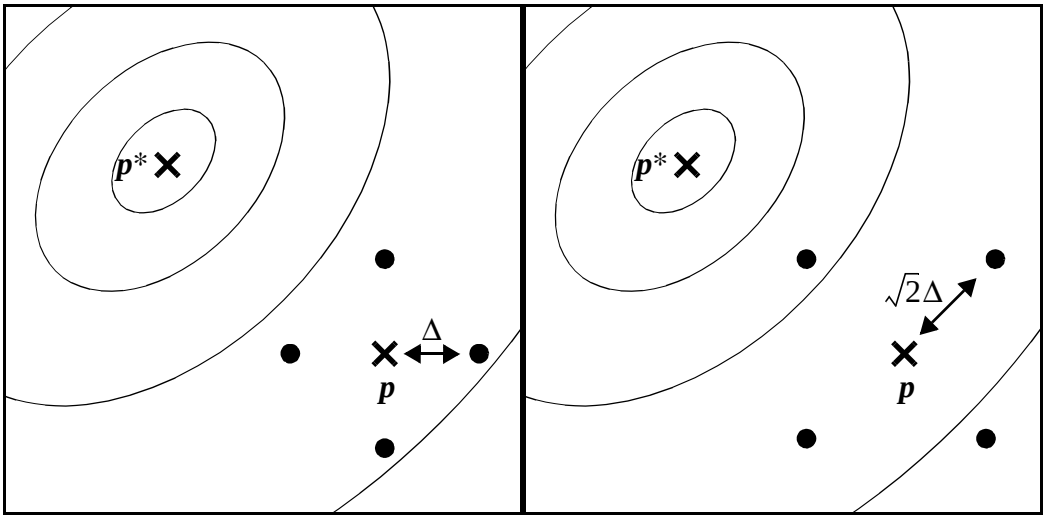


Figure 3.2 Perturbation in a 2-dimensional parameter space using (A) the DSD algorithm (B) the DLS algorithm.

The figure illustrates the points $\boldsymbol{p}'$ *where the filter is operated when the nominal parameter values are* $\boldsymbol{p} = [p_1\ p_2]$. (A) *Using the DSD, the filter operates one-quarter of the time at each of* $\boldsymbol{p}' = [p_1+D\ \ p_2]$, $[p_1-D\ \ p_2]$, $[p_1\ \ p_2+D]$, *and* $[p_1\ \ p_2-D]$. *(B) The DLS operates one-quarter of the time at each of* $\boldsymbol{p}' = [p_1+D\ \ p_2+D]$, $[p_1+D\ \ p_2-D]$, $[p_1-D\ \ p_2+D]$, *and* $[p_1-D\ \ p_2-D]$. *Note that the total perturbation is larger in (B) because both parameters are simultaneously dithered.*

Since $E[\hat{\nabla}_{\boldsymbol{p}}\varepsilon] = \nabla_{\boldsymbol{p}}\varepsilon$, it is known that $E[\boldsymbol{n}] = \mathbf{0}$ and,

$$\text{var}[\boldsymbol{n}] = \text{var}[\nabla_{\boldsymbol{p}}\varepsilon - \hat{\nabla}_{\boldsymbol{p}}\varepsilon] = \text{var}[\hat{\nabla}_{\boldsymbol{p}}\varepsilon] = \text{var}\left[\frac{\hat{\partial}\varepsilon}{\partial p_1} \ \cdots\ \frac{\hat{\partial}\varepsilon}{\partial p_N}\right]^T \tag{3.41}$$

Using Eqn. (3.25),

$$\begin{aligned}
\text{var}\left[\frac{\hat{\partial}\varepsilon}{\partial p_i}\right] &= \text{var}\left[\frac{1}{\sigma^2}\cdot\delta_i\cdot e^2\right] \\
&= \frac{1}{\Delta^4}\cdot\text{var}[\delta_i\cdot e^2] \\
&= \frac{1}{\Delta^4}\left(\frac{1}{2}\text{var}[\delta_i\cdot e^2]\Big|_{\delta_i=-\Delta} + \frac{1}{2}\text{var}[\delta_i\cdot e^2]\Big|_{\delta_i=\Delta}\right) \\
&= \frac{1}{\Delta^4}\left(\frac{1}{2}\Delta^2\text{var}[e^2]\Big|_{\delta_i=-\Delta} + \frac{1}{2}\Delta^2\text{var}[e^2]\Big|_{\delta_i=\Delta}\right)
\end{aligned} \tag{3.42}$$

As in [8], assume that $e = (d - y)$ is normally distributed with zero mean.[1] Therefore,

$$\text{var}[e^2] = 2\varepsilon^2 \tag{3.43}$$

In steady state with the filter parameters near their optimal values and assuming the perturbation is small,

$$\text{var}[e^2]\Big|_{\delta_i = -\Delta} \cong \text{var}[e^2]\Big|_{\delta_i = \Delta} \cong 2\varepsilon_{\min}^2 \tag{3.44}$$

Substituting Eqn. (3.44) into Eqn. (3.42),

$$\begin{aligned} \text{var}\left[\frac{\hat{\partial}\varepsilon}{\partial p_i}\right] &= \frac{1}{\Delta^4}(\Delta^2\varepsilon_{\min} + \Delta^2\varepsilon_{\min}) \\ &= \frac{2\varepsilon_{\min}^2}{\Delta^2} \end{aligned} \tag{3.45}$$

For the block DLS, take the variance of both sides of Eqn. (3.15),

$$\begin{aligned} \text{var}\left[\frac{\hat{\partial}\varepsilon}{\partial p_i}\right] &= \text{var}\left[\frac{1}{L}\sum_{k=l}^{l+L}\left(\frac{\delta_i(k)\cdot e^2(k)}{\sigma^2}\right)\right] \\ &= \frac{1}{\Delta^2}\left(\frac{1}{2}\text{var}\left[\frac{1}{L}\sum_{\substack{k=l \\ \delta_i=-\Delta}}^{l+L} e^2(k)\right] + \frac{1}{2}\text{var}\left[\frac{1}{L}\sum_{\substack{k=l \\ \delta_i=\Delta}}^{l+L} e^2(k)\right]\right) \end{aligned} \tag{3.46}$$

In Eqn. (3.46), each MSE estimate is made by averaging e^2 over $L/2$ data samples. Again assuming that in steady state, $\varepsilon \cong \varepsilon_{\min}$, the variance of the estimates are reduced by a factor of $L/2$ compared with the regular DLS algorithm.

$$\text{var}\left[\frac{2}{L}\sum_{\substack{k=l \\ \delta_i=\pm\Delta}}^{l+L} e^2(k)\right] = \frac{4\varepsilon_{\min}^2}{L} \tag{3.47}$$

Substituting Eqn. (3.47) into Eqn. (3.46) gives the variance of the block DLS gradient estimates,

1. The assumption of zero mean can be guaranteed by using an adaptive dc tap weight. In [7] it was shown that probability distributions for e other than gaussian result in a smaller misadjustment.

$$\text{var}\left[\frac{\hat{\partial\varepsilon}}{\partial p_i}\right] = \frac{1}{4\Delta^2}\left(\frac{4\varepsilon_{\min}^2}{L} + \frac{4\varepsilon_{\min}^2}{L}\right)$$
$$= \frac{2\varepsilon_{\min}^2}{L\Delta^2} \tag{3.48}$$

By taking a block length of $L = 1$, Eqn. (3.48) becomes the same as Eqn. (3.45) for the regular DLS algorithm.

3.4.5 Misadjustment

Perturbation is not the only source of excess MSE in steady state. The filter parameter values will persistently bounce around and away from their optimal values because the gradient estimates are noisy. This effect is called "misadjustment" and the excess MSE it causes will be quantified presently.

The noise in the gradient estimates given by Eqn. (3.48) for the DLS algorithm is the same as for the DSD algorithm [8]. The rest of the analysis follows exactly as in [8]. The misadjustment (i.e. the excess MSE excluding perturbation in steady state divided by the minimum MSE) using the DLS algorithm is,

$$M = \frac{N\mu\varepsilon_{\min}}{2L\Delta^2} \tag{3.49}$$

In order to write the misadjustment in terms of the algorithm's settling time and perturbation, an expression for λ_{avg} is required. Eqn. (3.21) can be rewritten as follows,

$$\lambda_k = \frac{L}{4\mu T_{k_{\text{MSE}}}} \tag{3.50}$$

$$\Rightarrow \lambda_{\text{avg}} = \frac{L}{4\mu}\left(\frac{1}{T_{\text{MSE}}}\right)_{\text{avg}} \tag{3.51}$$

Finally, Eqn. (3.39), Eqn. (3.49), and Eqn. (3.51) can be combined to yield,

$$M = \frac{N^2}{8P}\left(\frac{1}{T_{\text{MSE}}}\right)_{\text{avg}} \tag{3.52}$$

3.4.6 Total Excess MSE

In steady state the total excess MSE is the actual MSE minus the minimum MSE,

$$\varepsilon_{excess} = \varepsilon - \varepsilon_{min} \tag{3.53}$$

The ratio of this quantity to the minimum MSE is given by the sum of the perturbation and the misadjustment,

$$\zeta = \frac{\varepsilon_{excess}}{\varepsilon_{min}} = M + P \tag{3.54}$$

In order to minimize the relative excess MSE, ζ, it was shown in [8] that for the DSD algorithm one must make the perturbation term, P, one-half of the total ζ. Since the DLS misadjustment in Eqn. (3.52) is exactly the same as for the DSD algorithm, the same analysis applies here,

$$P_{optimum} = \frac{1}{2} \cdot \zeta \tag{3.55}$$

The resulting minimum relative excess MSE is,

$$\zeta_{min} = N \sqrt{\frac{1}{2} \cdot \left(\frac{1}{T_{MSE}} \right)_{avg}} \tag{3.56}$$

3.4.7 Comparison of the LMS, DSD, and DLS Algorithms

Table 3.1 shows a comparison of key performance measures for the LMS, DSD, and DLS adaptive algorithms. Block lengths of L data samples are used for the DSD and DLS algorithms. Taking $L = 1$ results in the regular DLS algorithm. The DSD algorithm requires NL data samples to generate one estimate of all N gradient components. The DLS algorithm can measure all N gradient components simultaneously in just L data samples thereby providing a factor of N improvement in settling time. On the other hand, since all N parameters are being dithered simultaneously its perturbation is N-times larger than for the DSD algorithm. So, in order to achieve the same performance with the DLS algorithm one must reduce both Δ^2 and μ by a factor of N compared with the DSD algorithm.

The performance of the DSD and DLS algorithms does not compare favorably with the LMS algorithm. Generally, the LMS algorithm will converge much faster when used in the same application. For example, a 10-tap adaptive FIR filter with white noise inputs which must converge with 1% excess MSE in steady state ($\zeta = 0.01$) will con-

	LMS	**DSD**	**DLS**
Gradient info req'd?	Yes	No	No
MSE decay time constant, $T_{\mathrm{MSE}_{\mathrm{avg}}}$	$\frac{1}{4\mu} \cdot \left(\frac{1}{\lambda}\right)_{\mathrm{avg}}$	$\frac{NL}{4\mu} \cdot \left(\frac{1}{\lambda}\right)_{\mathrm{avg}}$	$\frac{L}{4\mu} \cdot \left(\frac{1}{\lambda}\right)_{\mathrm{avg}}$
Perturbation, P	None	$\frac{\Delta^2 \lambda_{\mathrm{avg}}}{\varepsilon_{\min}}$	$\frac{N\Delta^2 \lambda_{\mathrm{avg}}}{\varepsilon_{\min}}$
Misadjustment, M	$\mu \cdot \mathrm{tr}(\boldsymbol{R}) = \frac{N}{4}\left(\frac{1}{T_{\mathrm{MSE}}}\right)_{\mathrm{avg}}$	$\frac{N\mu\varepsilon_{\min}}{2L\Delta^2} = \frac{N^2}{8P}\left(\frac{1}{T_{\mathrm{MSE}}}\right)_{\mathrm{avg}}$	$\frac{N\mu\varepsilon_{\min}}{2L\Delta^2} = \frac{N^2}{8P}\left(\frac{1}{T_{\mathrm{MSE}}}\right)_{\mathrm{avg}}$
Minimum relative excess MSE in steady state, $\zeta_{\min}$	$\frac{N}{4}\left(\frac{1}{T_{\mathrm{MSE}}}\right)_{\mathrm{avg}}$	$\frac{N}{\sqrt{2}}\left(\frac{1}{T_{\mathrm{MSE}}}\right)^{1/2}_{\mathrm{avg}}$	$\frac{N}{\sqrt{2}}\left(\frac{1}{T_{\mathrm{MSE}}}\right)^{1/2}_{\mathrm{avg}}$

Table 3.1 Performance measures for gradient descent adaptive algorithms.

verge 2,000× faster using the LMS algorithm than with the DLS algorithm. Furthermore, it is generally possible to be more aggressive in choosing μ with the LMS algorithm resulting in even faster convergence. Clearly the DLS algorithm would only be used when it is difficult or impossible to access the internal states of the adapted filter. As noted in the introduction, this is frequently the case when attempting to digitally adapt an analog filter. Typically, an analog filter would only be used if the signal bandwidth were high. So, even if the adaptation algorithm requires a large number of iterations to converge, the sampling rate is so high that the process converges quickly in absolute terms. In these situations, the DLS algorithm's straightforward hardware implementation is an attractive alternative to the LMS algorithm. The main advantage of the DLS algorithm over the DSD algorithm is its simpler hardware implementation. There are also performance trade-offs when one takes into account quantization of the filter parameters, which will be discussed in Section 3.9.

3.5 Behavioral Simulations

Behavioral simulations were performed using two filter structures: a 5-tap transversal filter and a 3rd order continuous time orthonormal ladder filter. A block diagram of the model matching system used is shown in Fig. 3.3.

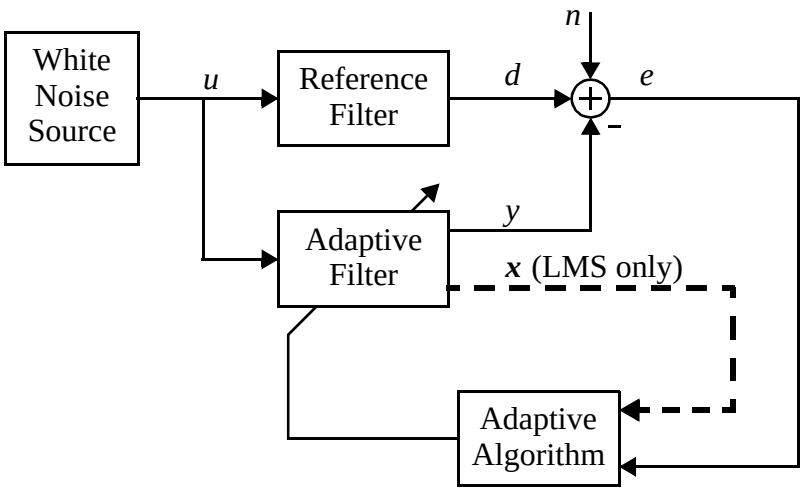


Figure 3.3 Model matching simulations block diagram.

3.5.1 5-Tap FIR Filter

An adaptive transversal filter is an adaptive linear combiner whose state signals are time delayed samples of the input.

$$\boldsymbol{x}(k) = \left[u(k)\ u(k-1)\ \ldots\ u(k-N+1)\right]^T \tag{3.57}$$

The filter order is $N = 5$ and a finite minimum MSE is introduced via the zero mean gaussian additive noise, n. For all of the simulations in this section,

$$\varepsilon_{\min} = \mathrm{var}[n] = 0.01 \tag{3.58}$$

The input sequence $u(k)$ is white (uncorrelated) with zero-mean and unit power. Therefore, the state correlation matrix is equal to the identity matrix, $\boldsymbol{R} = \boldsymbol{I}$, and all eigenvalues are unity: $\lambda_k = \lambda_{\text{avg}} = 1$. The equations in Table 3.1 were used to design LMS, DSD, DLS and block DLS adaptive processes with the same total excess MSE, $\zeta = 0.01$. The resulting values for Δ, μ and L are tabulated in Table 3.2. Pseudoran-

	LMS	DSD	DLS	Block DLS
Δ^2	-	5×10^{-5}	10^{-5}	10^{-5}
μ	2×10^{-3}	10^{-3}	2×10^{-6}	2×10^{-4}
L	-	100	-	100
T_{MSE} (data samples)	125	125,000	125,000	125,000
ζ	0.01	0.01	0.01	0.01

Table 3.2 Summary of adaptive algorithms for the 5-tap transversal filter simulations.

dom binary sequences of length (2^{24} - 1) were used as dither for the DLS and block DLS algorithms.

Simulation results are plotted in Fig. 3.4. The DSD, DLS and block DLS algorithms all converge at the exact same rate, as predicted by the analytical results. The convergence rate of the LMS algorithm is far superior, but it is also the only algorithm which requires a knowledge of the state signals, $\boldsymbol{x}$.

3.5.2 3rd Order Continuous Time Orthonormal Ladder Filter

The structure for a 3rd order orthonormal ladder with programmable feed-ins is presented in Fig. 3.5 [10].

With the feedback parameters α_i fixed and the feed-in terms β_i adapted, the filter is an adaptive linear combiner. However, the state signals are not available at any internal nodes in the structure of Fig. 3.5, so the gradient signals required for LMS adaptation would be very difficult to obtain. Specifically, it would be necessary to operate a second 3rd order continuous time filter in parallel just to generate the gradient signals, $\mathbf{f}$ [11]. Therefore, the DSD, DLS and block DLS algorithms are particularly desirable for this structure and simulation data for the LMS algorithm is not even presented.

Again, a model matching experiment was performed with $\varepsilon_{\text{min}} = 0.01$. The input signal is white with a power of 10. The reference filter is a 3rd order orthonormal ladder filter. The gain parameters were designed for an elliptic lowpass transfer function with 0.5 dB ripple in the passband extending to a frequency of 10 (normalized with respect to

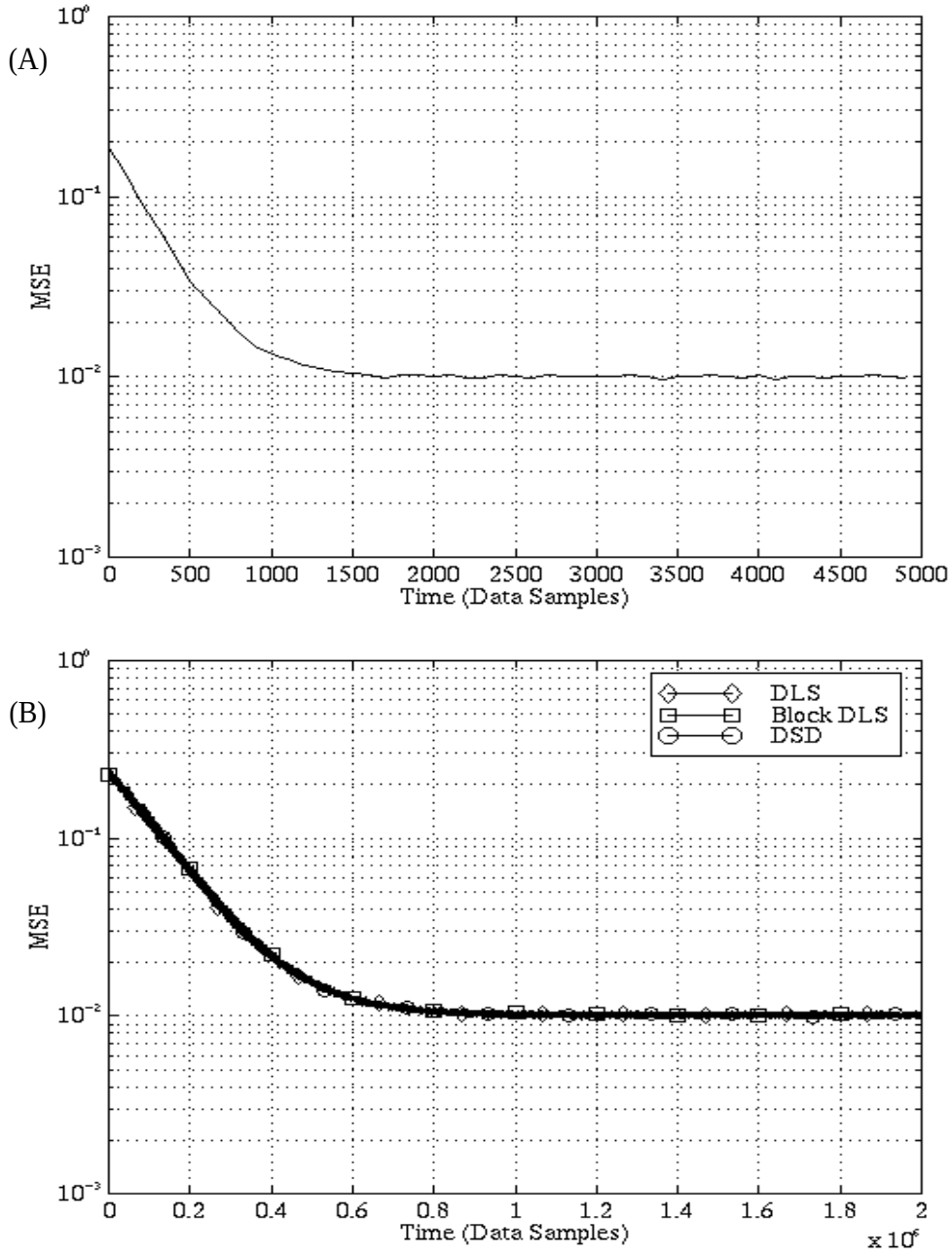


Figure 3.4 Simulation results for a 5-tap adaptive transversal filter.
(A) LMS (B) DLS, Block DLS, and DSD algorithms. Each plotted MSE point is computed by taking the mean of $e^2(k)$ *over an ensemble of 100 simulation runs and 100 consecutive data samples.*

the sampling frequency) and 40 dB of stopband attenuation. The same feedback parameters, α_i, were used in the adaptive filter, but the feed-in parameters, β_i, were adapted. The resulting inputs to the adaptive linear combiner are orthogonal with power

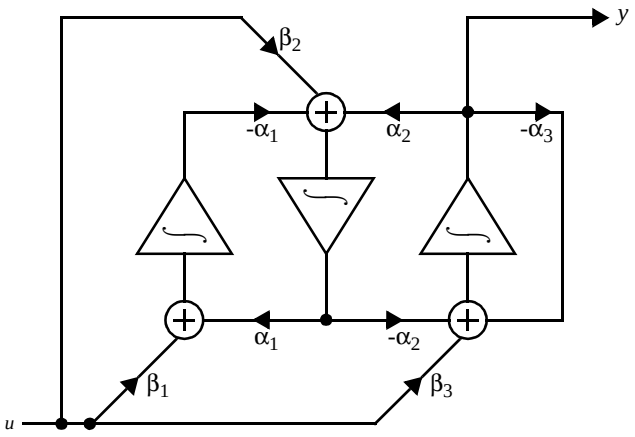


Figure 3.5 A 3rd order orthonormal ladder filter using multiple feed-ins of the input signal.

$\lambda \cong 6 \times 10^{-4}$. The values of Δ, μ, and L were again computed using Table 3.1 to yield a total relative excess error of $\zeta = 0.01$ and are tabulated in Table 3.3.

Results from an ensemble of 25 simulation runs are plotted in Fig. 3.6. Again, pseudorandom binary sequences of length (2^{24} - 1) were used as dither for the DLS and block DLS algorithms. All three algorithms converge at the same rate.

	DSD	**DLS**	**Block DLS**
Δ^2	0.083333	0.027778	0.027778
μ	2.7500	0.0092593	0.92593
L	100	-	100
T_{MSE} (data samples)	45,000	45,000	45,000
ζ, theoretical	0.01	0.01	0.01

Table 3.3 Summary of adaptive algorithm simulations for the 3rd order orthonormal ladder filter.

3.6 Different Dither Signals

Until now, all simulations have used independent pseudorandom binary sequences (PRBSs) for the dither. These signals are practical because,

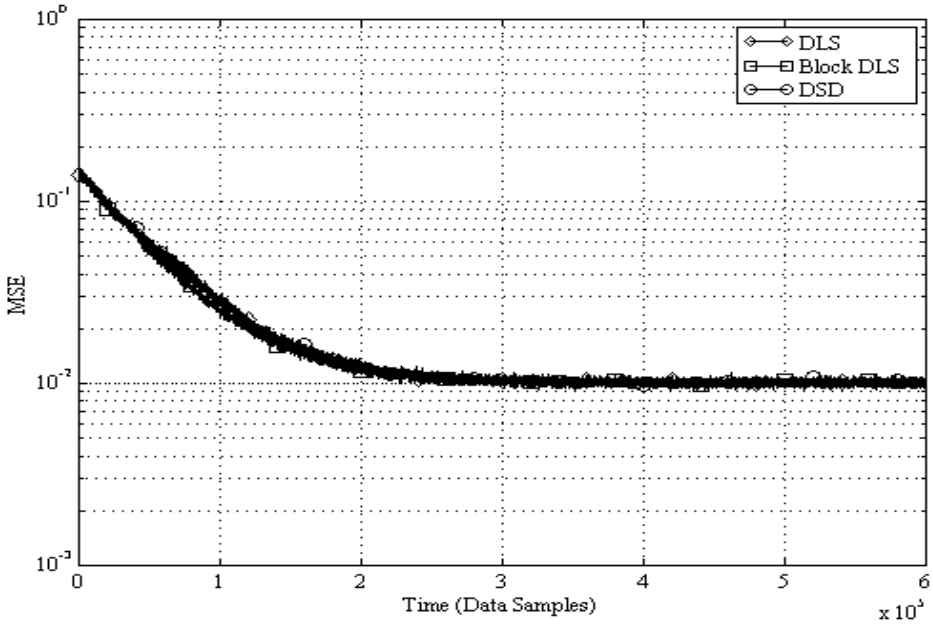


Figure 3.6 Simulation results for a 3rd order continuous time adaptive orthonormal ladder with variable feed-ins using the DLS, Block DLS, and DSD algorithms.

Each plotted MSE point is computed by taking the mean of $e^2(k)$ *over an ensemble of 25 simulation runs and 100 consecutive data samples.*

- Binary signals are amenable to exact convergence and misadjustment analyses.
- PRBSs are easily generated with simple digital hardware. Delayed versions of the same sequence are uncorrelated so only one pseudorandom bit generator is required, as shown in Fig. 3.7.
- PRBSs have a flat spectrum so even if the dither is fast, it will not introduce spurious tones in the filter output spectrum.

One disadvantage of PRBSs is that they occasionally contain long strings of consecutive ones or zeros. When the expectation is taken over a long time interval, $E[\delta_i] = 0$. However, over a shorter time interval PRBSs can have significant dc content. On each iteration, the DLS update equation Eqn. (3.11) moves the parameters in the same direction as the dither signal's sign. Therefore, a long string of ones will cause the filter

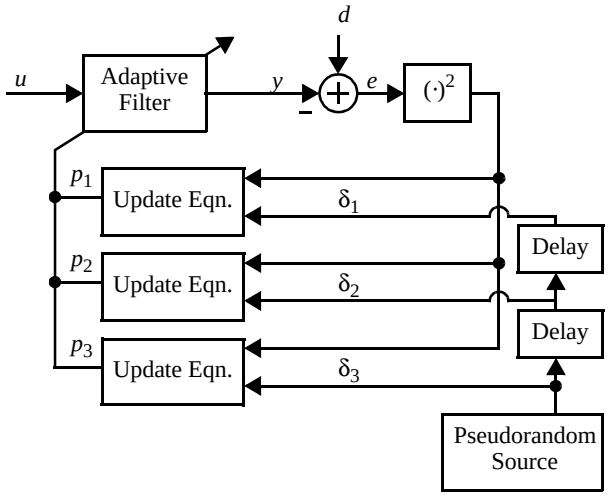


Figure 3.7 Dithered linear search using pseudorandom binary dither.
One pseudorandom bit generator is used to generate all of the dither signals.

parameter to drift upward, even if this means climbing the performance surface. During convergence, this can cause the learning curves to bounce unpredictably or even go unstable. For instance, Fig. 3.8 shows a parameter learning curve for the same DLS adaptive filter simulated in Section 3.5.2 except that one of the pseudorandom binary dither sources is initialized to all zeros. The resulting string of consecutive zeros causes the algorithm to quickly diverge.

Even when the algorithm does converge, the theoretical analysis performed in Section 3.4 is not valid because it assumes that the parameters remain near their optimal values in steady state. Since a long string of ones or zeros can cause the parameters to drift away from their optimal values, the steady state misadjustment will be somewhat greater than predicted by Eqn. (3.52). For instance, the steady state excess MSE observed in Fig. 3.6 for the DLS and block DLS algorithms are 31% and 24% higher than predicted, respectively. Parameter drift is less likely using a long block length, L, since the parameters are only updated after the gradient estimates are averaged over a long period of time thereby filtering out the effect of any long strings of ones or zeros.

To avoid these problems, it is desirable to combine the block DLS algorithm with a dither signal that is dc-free within each block. The DSD algorithm is a special case of the block DLS algorithm which does exactly that. The dither signals are independent

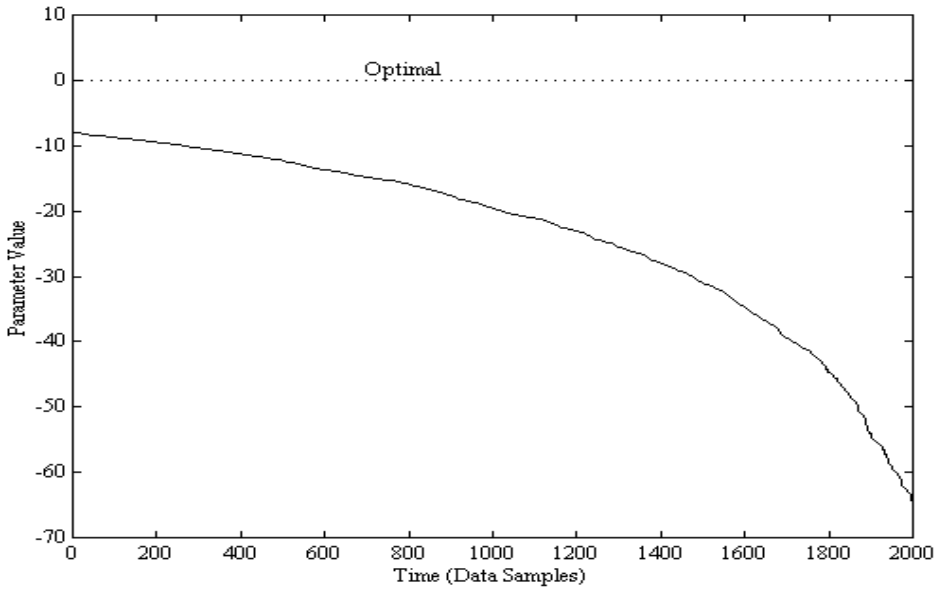


Figure 3.8 Divergent learning curve of the DLS with a long string of consecutive zeros in the binary dither.

because they are separated in time, and they are dc-free over each block of length NL, as shown in Fig. 3.9. As a result, there is no parameter drift. The simulation results in Fig. 3.6 indicate a lower steady state misadjustment for the DSD algorithm than the DLS algorithm with pseudorandom dither.

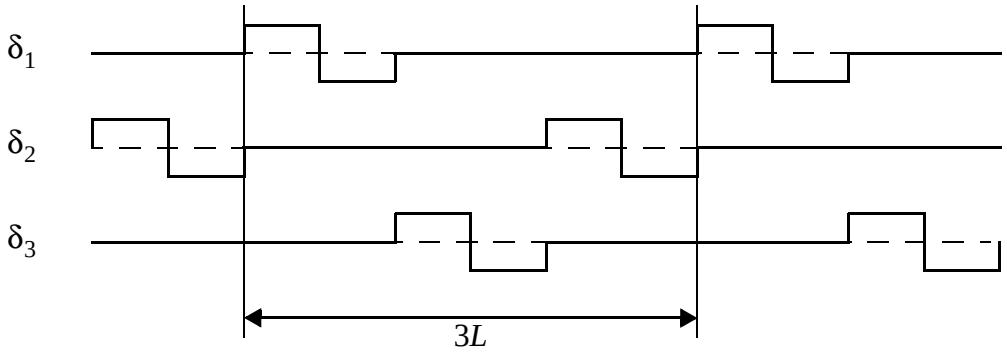


Figure 3.9 The dither signals used for the DSD algorithm with 3 parameters.
Note that within the time interval for each iteration, all of the dither signals have zero dc content.

The dither signals for the DSD algorithm are 3-level signals, which complicates the hardware implementation somewhat. Instead, balanced binary sequences can be

repeated over and over to produce a periodic dither which is dc-free over a fixed time interval. For instance, two clock signals with periods of $L/2$ and L are uncorrelated and dc-free over a block of length L.

Hadamard sequences are finite length, independent, binary-valued vectors derived from the rows of Hadamard matrices [12]. For example, the Hadamard sequences of length 4 are $\begin{bmatrix}1 & 1 & 0 & 0\end{bmatrix}$, $\begin{bmatrix}0 & 1 & 1 & 0\end{bmatrix}$, $\begin{bmatrix}1 & 0 & 1 & 0\end{bmatrix}$ and $\begin{bmatrix}1 & 1 & 1 & 1\end{bmatrix}$. The fourth sequence is not balanced, but the other three can be repeated as independent periodic 2-level dither signals (Fig. 3.10) which are dc-free over a block of length 4.

The 3rd order orthonormal ladder model matching simulation from Section 3.5.2 was repeated using Hadamard sequences for the dither. Again, μ and Δ were chosen for a relative excess MSE of 0.01. The results are plotted in Fig. 3.11 and indicate the exact same rate of convergence as was obtained with the pseudorandom dither. The steady state excess MSE observed during simulations with different algorithms and dither signals are tabulated in Table 3.4. As expected, there is less excess MSE with Hadamard sequence dither than with pseudorandom dither. The block DLS and DSD algorithms provide a further improvement in misadjustment because the gradient estimates for each parameter update are averaged over many data samples and, hence, more accurate so the parameters are less likely to drift away from their optimal values.

Hadamard sequences are easily generated by a few gates of digital hardware. As long as the dither source is clocked slowly, Hadamard sequences are a desirable alternative to pseudorandom sequences. However, the problem with any periodic dither is that it introduces spurious tones at the filter output when applied quickly.

3.7 Dc Offset Effects

It is well known that dc offsets can limit the performance of the LMS algorithm for analog filters [1], [2], [3]. Offsets on the state and error signals cause excess MSE at steady state. Therefore, dc offsets represent a significant performance limitation for analog adaptive filters and much research has been done to reduce the effect of dc offsets on LMS adaptation. Offsets on the error signal, $e(k)$ are usually eliminated using an adap-

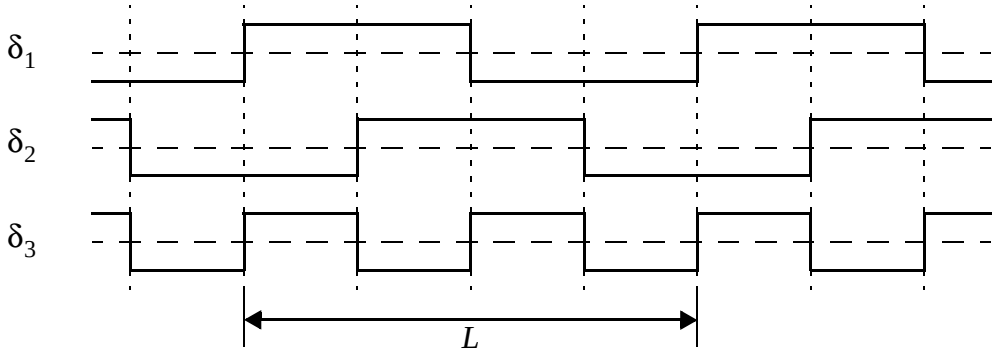


Figure 3.10 Dither signals generated from Hadamard sequences suitable for 3 parameters.

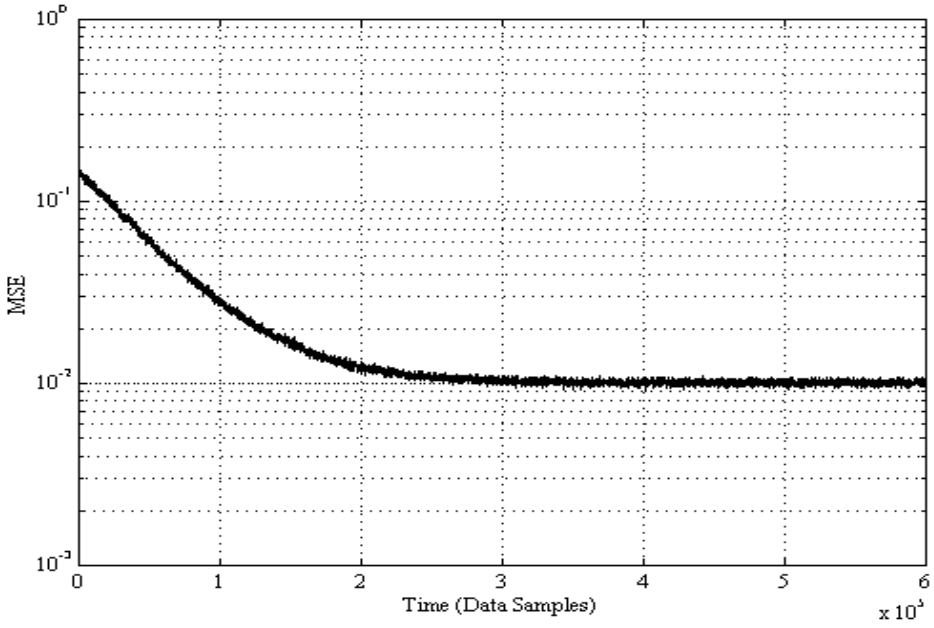


Figure 3.11 Simulation results for a 3rd order continuous time adaptive orthonormal ladder with variable feed-ins using the DLS algorithm with Hadamard dither.

Each plotted MSE point is the mean of $e^2(k)$ taken over an ensemble of 25 simulation runs and 100 consecutive data samples.

tive dc tap at the filter output as described in Section 1.3.2. Furthermore, since the DLS and DSD algorithms estimate the gradient from observations of the squared error, $e^2(k)$, they are not susceptible to dc offsets on the state signals.

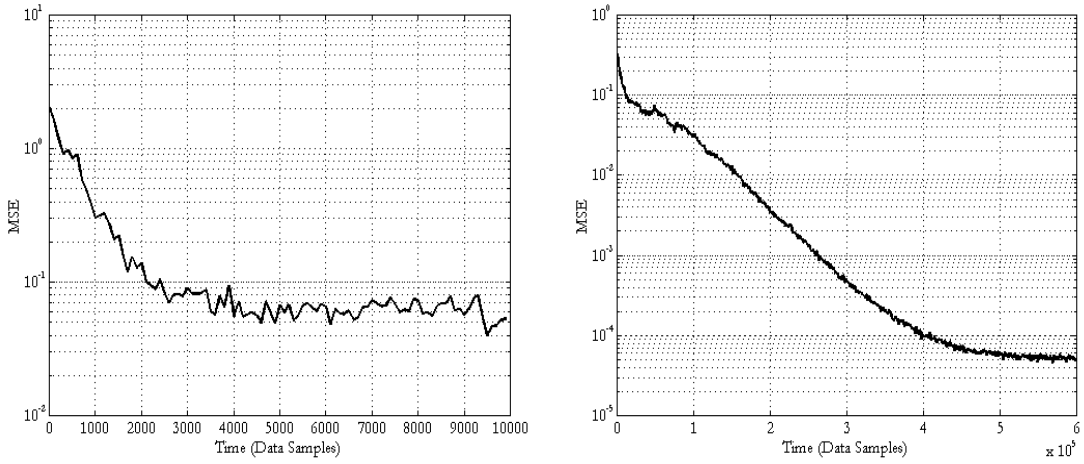
DLS		Block DLS		DSD
PRBS	**Hadamard**	**PRBS**	**Hadamard**	
0.0131	0.0119	0.0124	0.0090	0.0094

Table 3.4 Comparison of relative excess MSE in steady state observed in simulations using different algorithms and dither signals.

In all cases, the values of Δ and μ were chosen optimally to yield a theoretical excess MSE of 0.01 (1%). For the BDLS and DSD algorithms, a block size of 100 was used. Each value is the mean of 200,000 consecutive data samples and 25 independent simulation runs. The simulation conditions are identical to those described in Section 3.5.2.

In order to highlight the effect of dc offsets, behavioral simulations were performed for the same model matching 3rd order orthonormal ladder filter described in Section 3.5.2, this time with dc offsets introduced on each of the filter's internal state nodes and on the error signal. The dc offsets have a mean squared value $1/10$th that of the state and error signals respectively. No steady state error is introduced at n. Using LMS adaptation (Fig. 3.12A) a residual error of approximately -15 dB relative to the reference signal persists due to the dc offsets. Using the same dc offsets and the DLS algorithm with an adaptive dc tap at the filter output (Fig. 3.12B) the only residual steady state error is due to the perturbation caused by the dither itself, in this case approximately -45 dB relative to the reference signal.

Improved performance in the presence of dc offsets is a feature of digital implementations of gradient descent algorithms generally, not the DLS algorithm specifically. As mentioned in Section 1.3.2, it is possible to combine an adaptive dc tap with a digital LMS algorithms to eliminate the excess MSE due to dc offsets. It is also possible to use an adaptive dc tap in a filter with analog LMS adaptation; however, implementation of the dc tap would be much more complicated in this case. Furthermore, dc offsets in the adaptation hardware would persist, and since these offsets usually dominate, any performance improvement obtained would be marginal.



(A) (B)

Figure 3.12 Mean squared error simulated with dc offsets on the state and error signals.
(A) *the LMS algorithm* (B) *the DLS algorithm with an adaptive dc tap.*

3.8 Subsampling

In digital communications receivers, analog adaptive filters are generally followed by an A/D converter operating at or above the input's Nyquist rate to allow for digital demodulation. In such cases, a digital error signal can be generated from the A/D output and used for DLS or DSD adaptation (Fig. 3.13). However, if the adaptive filter is followed by more analog signal processing, an extra A/D converter must be built just to obtain the error signal for adaptation (Fig. 3.14). Fortunately, there is no reason why the update equation must iterate at the Nyquist rate. In fact, it is possible to subsample the error signal and iterate the update equations as slowly as desired. This would allow the DLS algorithm to proceed using just one subsampled A/D converter, as shown in Fig. 3.14, compared with up to $(N + 1)$ A/D converters required to perform the LMS algorithm digitally.

Interestingly, when programmable feed-ins are being adapted, it is actually *required* that the error signal be subsampled. The feed-ins must be held constant long enough for the filter's output to settle prior to sampling the output error signal. If the feed-in gains are

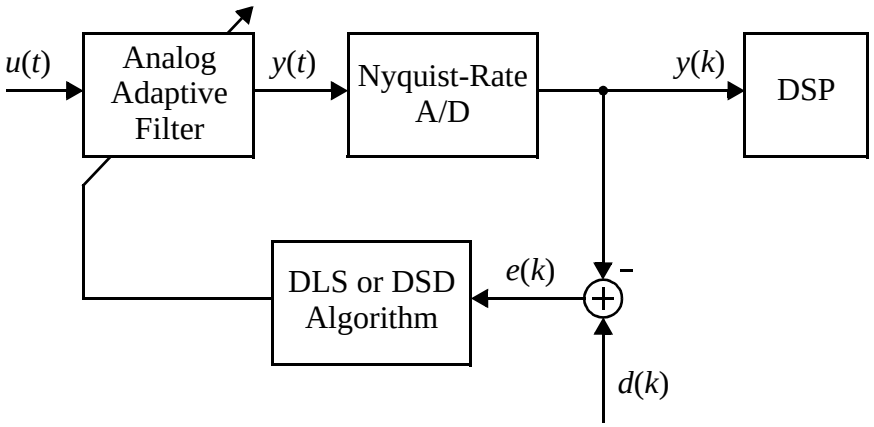


Figure 3.13 Implementing the DLS algorithm digitally when the adaptive filter is followed by a Nyquist-rate A/D converter.

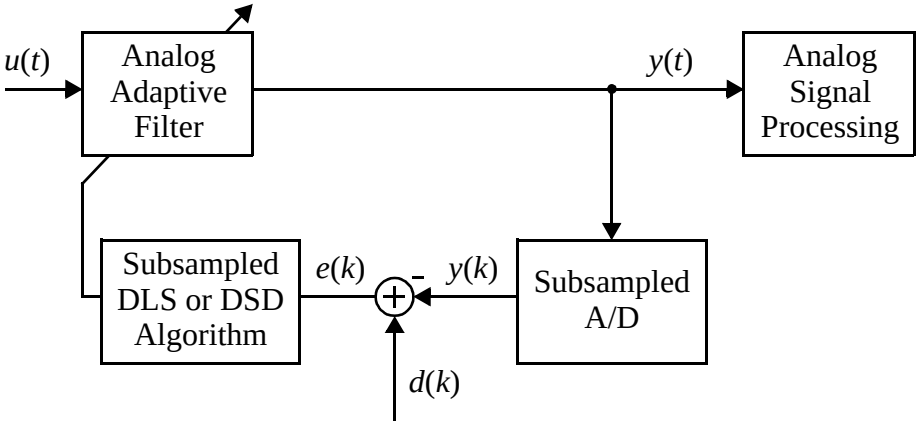


Figure 3.14 Implementing the DLS algorithm digitally with a subsampled A/D converter.

dithered faster than the filter bandwidth, the filter output (and, hence, the error signal) will not be able to track the dither and the DLS algorithm will not work.

An example of subsampled DLS adaptation was already provided in Section 3.5.2 since the 3rd order orthonormal ladder had programmable feed-ins. In that example, the filter and input signal had a bandwidth 10× greater than the sampling rate of the adaptation algorithm. This allowed the filter's output to settle before each new error signal sample was taken. In Fig. 3.15 the simulation was repeated with the sampling rate increased beyond the filter's Nyquist rate. As expected, the algorithm diverges.

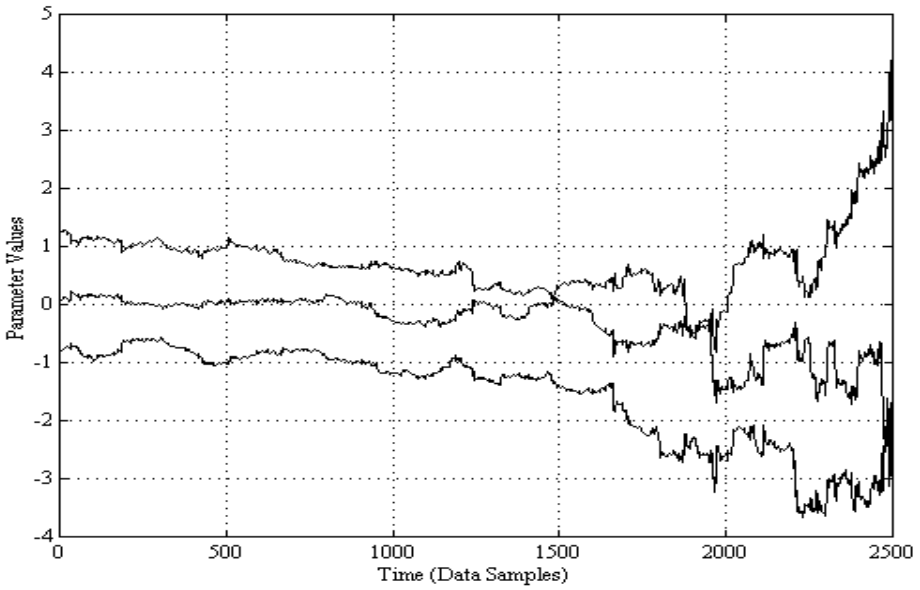


Figure 3.15 Oversampled DLS adaptation of programmable feed-ins.
The parameters were initialized to their optimal values, yet the algorithm diverges because the filter output cannot track the dither signal fast enough.

3.9 Quantization

So far, it has been assumed that the amplitude of the perturbation, Δ, can be chosen with arbitrary accuracy. In practice, this is not the case since the dither is introduced on digitally programmable filter parameters with finite resolution. This section will consider how quantization of the filter parameters affects the choice of a dither signal.

As described in Section 3.4, the excess steady state MSE has two components: misadjustment and perturbation. As long as a slow rate of convergence can be tolerated, misadjustment can be made arbitrarily small by decreasing μ. (If fast convergence is required, the DLS algorithm is probably not a good choice anyway.) Unfortunately, the only way to decrease the perturbation is to decrease the dither amplitude. The minimum dither amplitude will be limited by the resolution of the parameters' digital control since the dither must be sufficient to toggle at least 1 LSB of the parameters. In many analog adaptive filters, the parameters will have a wide tuning range but only modest resolution

(5 to 8 bits is typical). Therefore, the excess steady state MSE will be dominated by the perturbation.

In some applications, after initial convergence, the adaptation can be frozen and the dithers turned off to eliminate perturbation errors. If this is not possible, the dither amplitude has to be minimized. In general, this means using a binary dither signal connected to the parameters' LSB leading to a straightforward hardware implementation. However, in some circumstances, multi-level dither can actually reduce the perturbation. For instance, the DSD algorithm uses a 3-level dither signal: $+\Delta, 0, -\Delta$. Assuming $\Delta = 1\,\text{LSB}$, the perturbation is

$$P_{\text{DSD}} = \frac{(1\,\text{LSB})^2 \lambda_{\text{avg}}}{\varepsilon_{\min}} \tag{3.59}$$

Using the DLS algorithm with binary dither, the dither will be two-level: +1 LSB or 0. Therefore, $\Delta = 0.5\,\text{LSB}$ and the resulting perturbation is,

$$P_{\text{DLS}} = \frac{N}{4}\frac{(1\,\text{LSB})^2 \lambda_{\text{avg}}}{\varepsilon_{\min}} = \frac{N}{4} \cdot P_{\text{DSD}} \tag{3.60}$$

If 3 or fewer parameters are being adapted ($N < 4$) the DLS algorithm introduces less perturbation ($P_{\text{DLS}} < P_{\text{DSD}}$) and, hence, a smaller excess MSE in steady state. However, when 5 or more parameters are adapted, the DSD provides less perturbation.

Other multi-level dither signals with low duty cycles, like the DSD dither signals, could provide similar advantages in certain situations. Of course, the slight improvement in performance would have to be weighed against the accompanying increase in complexity.

3.10 Experimental Results

Model matching experiments were performed using the DLS algorithm and continuous time analog integrated filters on the test chip described in Chapter 2. Only the digitally programmable analog signal path was integrated. The adaptation algorithm was implemented in software to provide greater flexibility. Model matching experiments

were performed using both the 1st order lowpass filter and the 5th orthonormal ladder filter.

3.10.1 1st Order Lowpass Filter

The first order CMOS G_m-C filter has a digitally programmable pole and dc gain (Fig. 2.13). The transfer function is,

$$\frac{g_{m1}/g_{m2}}{1 - sC/g_{m2}} \tag{3.61}$$

Both transconductances g_{m1} and g_{m2} are 5-bits programmable, but g_{m1} has an additional sign bit. The sign of g_{m2} is fixed positive to ensure stability of the filter. A dither of 1 LSB was applied to both transconductors simultaneously.

The use of a gradient descent algorithm to adapt filter poles can be somewhat problematic since the MSE performance surface is not necessarily quadratic. However, as

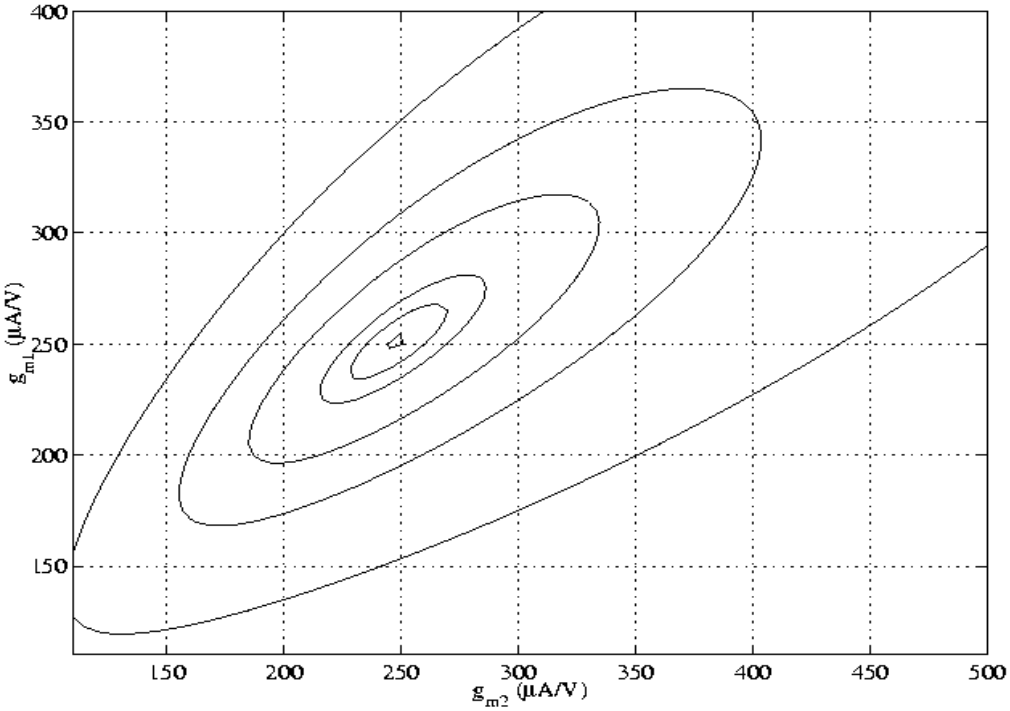


Figure 3.16 Contour plot of MSE for the 2-parameter adaptive filter in Fig. 2.13 in a model matching experiment.

The reference filter is a first order lowpass filter with 0 dB dc gain, a -3dB frequency of 80 MHz and C = 500 fF.

long as no local minima exist in the vicinity of the optimal parameter values, gradient descent algorithms do converge to the optimal parameter values. This is verified in Fig. 3.16 which shows a contour plot of the performance surface.

A block diagram of the experimental setup is shown in Fig. 3.17. The input signal is bandlimited noise. When triggered by the PC, the oscilloscope stores 8000 samples of the filter's input and output at a rate of 1 GSample/second. The waveform data is then transferred to the PC via a GPIB interface. The digitized input signal is passed through a first order digital reference filter in software. The first 1000 data samples are discarded to eliminate any transient effects, and an estimate of the squared error is obtained by averaging the remaining 7000 data samples. This estimate is then used to update the filter parameters via the DLS algorithm. Both oscilloscope inputs are ac coupled, so there is no dc offset in the error signal.

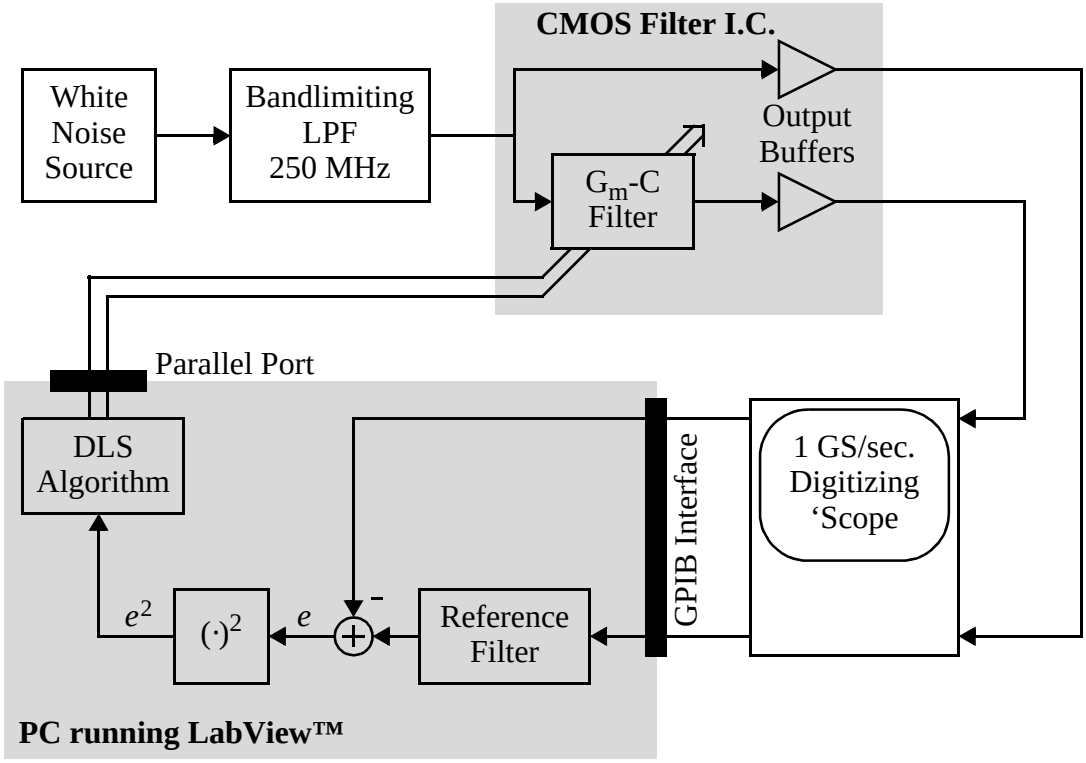


Figure 3.17 A Block diagram of the 1st order 2 parameter model matching experiment.

The digital filter used for the reference signal path was designed by taking the bilinear transform of a first order continuous time filter with a 3dB frequency of 80 MHz, a dc gain of 3 dB and a sampling rate of 1 GS/sec. Pseudorandom binary dither signals were

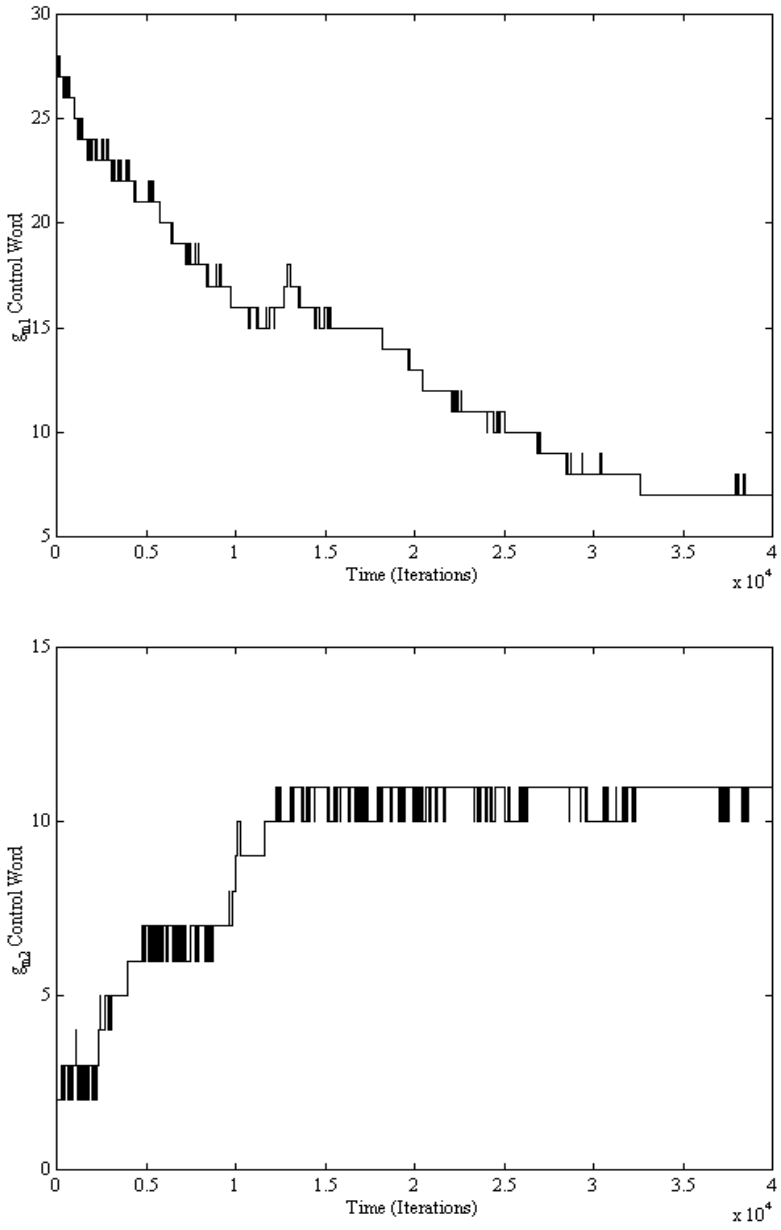


Figure 3.18 Parameter evolution in the first order, two parameter hardware model matching experiment.

The additive dither is not shown on this plot.

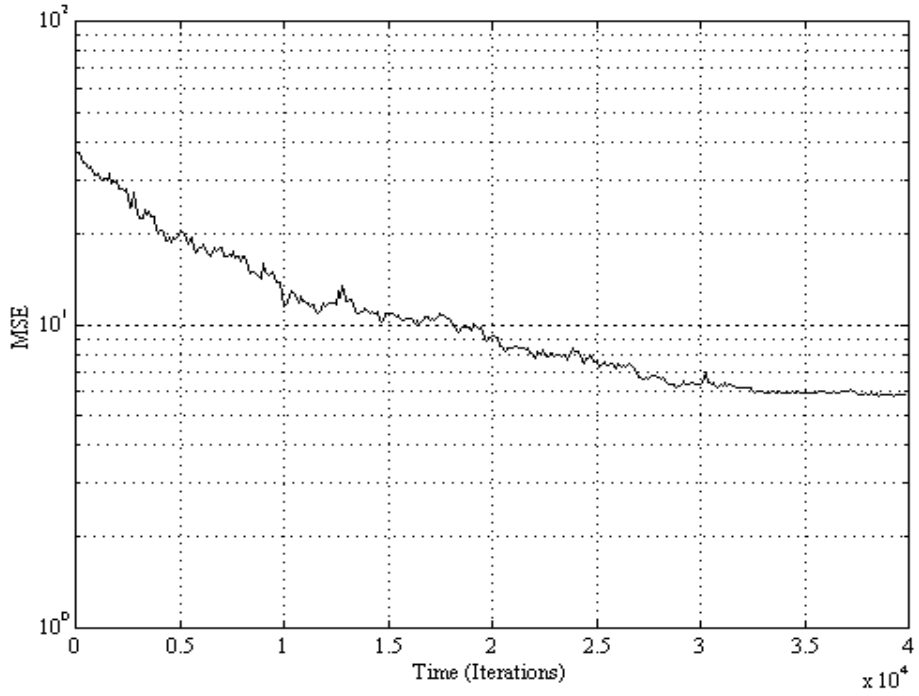


Figure 3.19 Mean squared error estimates over time in the first order, two parameter hardware model matching experiment.

used. The filter parameters are plotted over time in Fig. 3.18 and the MSE estimates in Fig. 3.19.

3.10.2 5th Order Orthonormal Ladder Filter

Experiments were also performed on the 5th order orthonormal ladder CMOS prototype filter. The filter was configured as shown in Fig. 3.20. Each of the three feed-in

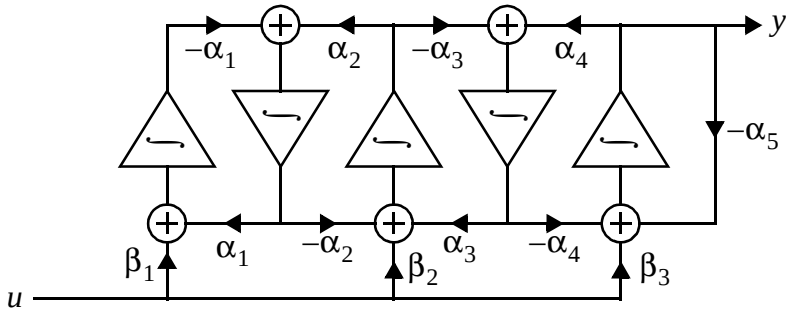


Figure 3.20 5th order orthonormal ladder filter structure with programmable feed-ins.

taps is digitally programmable with 5-bits of resolution. The shape of the frequency response was fixed, but the cutoff frequency was made programmable up to around 70 MHz by scaling the feedback gains α_i (Fig. 3.21A) and the dc gain was programmable by scaling the feed-in parameters, β_i (Fig. 3.21B). These two scaling factors were used as the adapted parameters for another model matching experiment. The block diagram for the experiment is somewhat different than the simulations; for convenience in testing, the integrated filter was used as both the reference and adapted signal path (Fig. 3.22). This time Hadamard sequences were used for the dither, specifically signals δ_1 and δ_2 from Fig. 3.10.

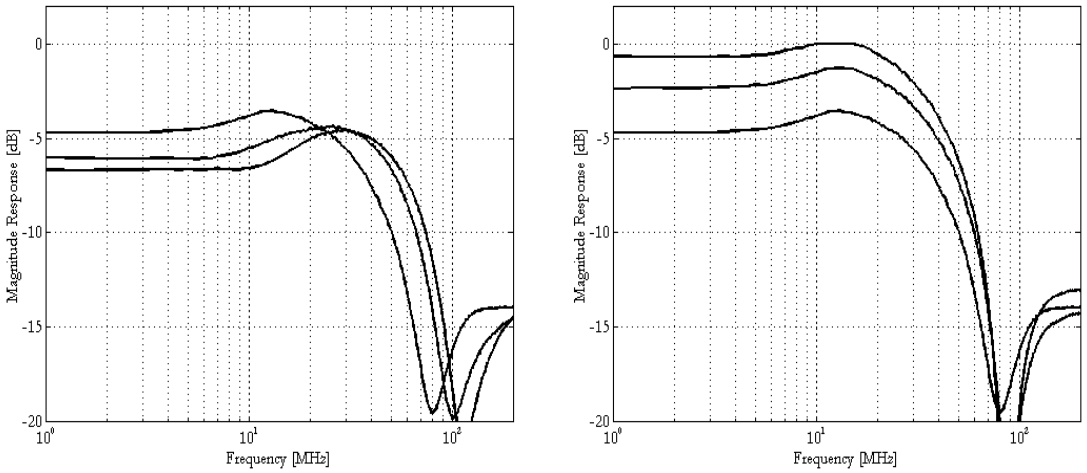


Figure 3.21 Magnitude responses of the adapted 5th order filter.
The measurements were made while varying (A) *the cutoff frequency and* (B) *the gain.*

The filter parameters are plotted over time in Fig. 3.23, and the relative mean squared error in Fig. 3.24. The non-zero MSE observed in steady state is due to measurement noise. The adapted parameters are unquantized scaling factors, unlike the quantized control words which were the parameters in Section 3.10.1. As a result, the real parameter values bounce slightly around the optimal values in steady state.

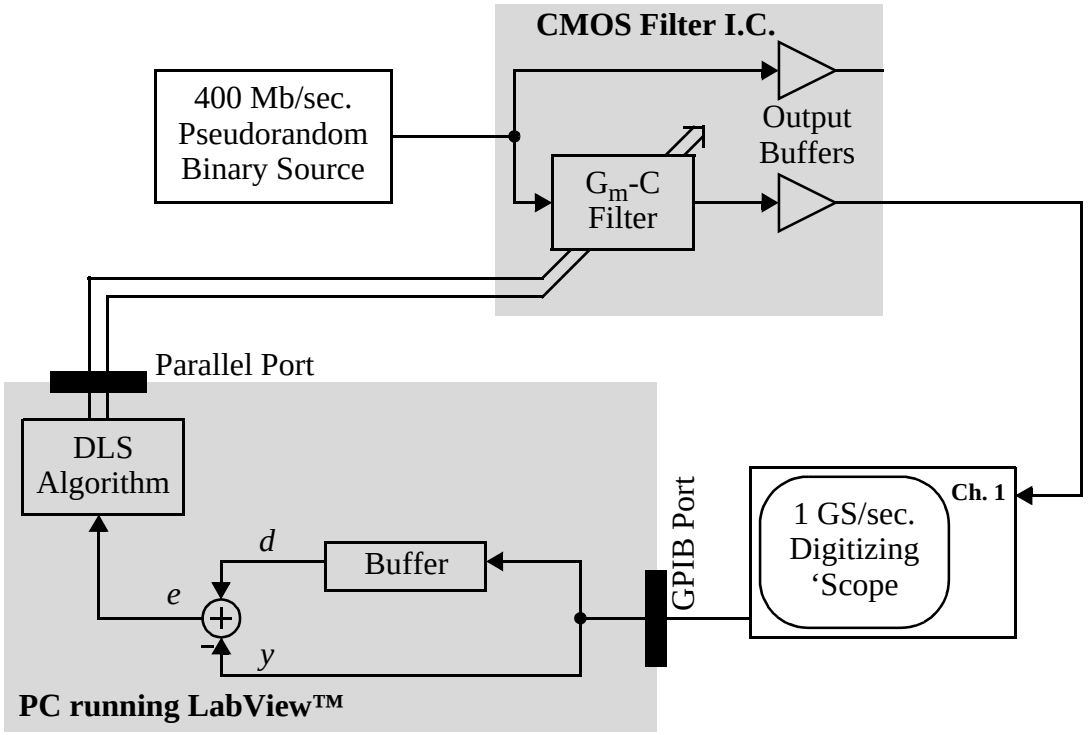


Figure 3.22 A Block diagram of the 5th order 2 parameter model matching experiment.

3.11 Conclusions

A generalization of the differential steepest descent (DSD) algorithm, here called the dithered linear search (DLS), was described and analyzed. Its performance was shown to be the same as the DSD algorithm, however its hardware implementation can be somewhat simpler since only the parameters' LSB must be dithered during adaptation. When compared with the popular LMS algorithm, the DLS algorithm is much slower because it does not make use of the filter's internal state signals. However, the LMS algorithm has a number of serious drawbacks, particularly for analog continuous time integrated filters. If implemented digitally, the state signals required for LMS adaptation are difficult to obtain or completely unavailable. If implemented with analog circuits, the LMS algorithm is susceptible to dc offsets. The DLS algorithm does not require access to any internal state signals and, by introducing an adaptive dc tap, the DLS algorithm is robust

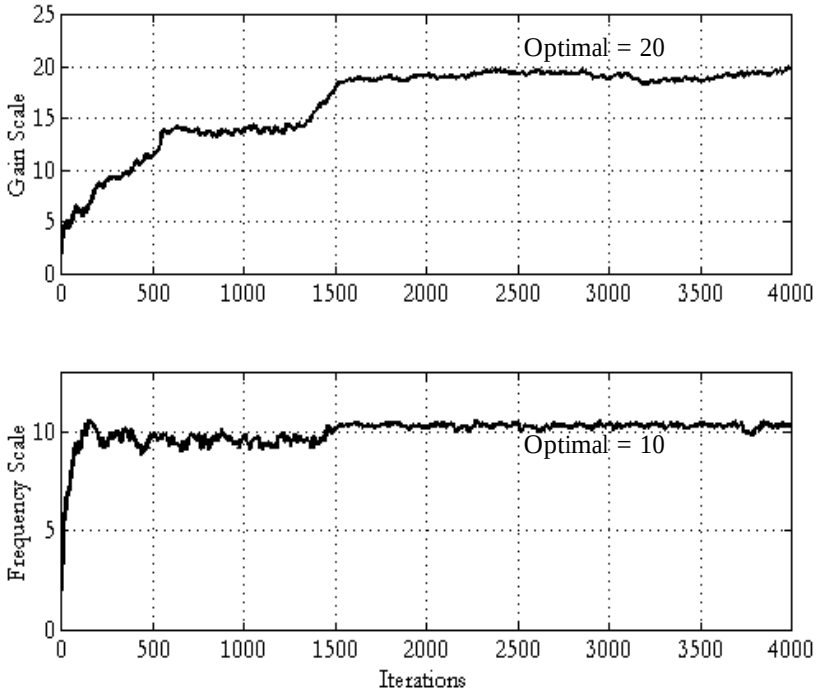


Figure 3.23 Adapted parameters of a 5th order analog integrated filter using the DLS algorithm.

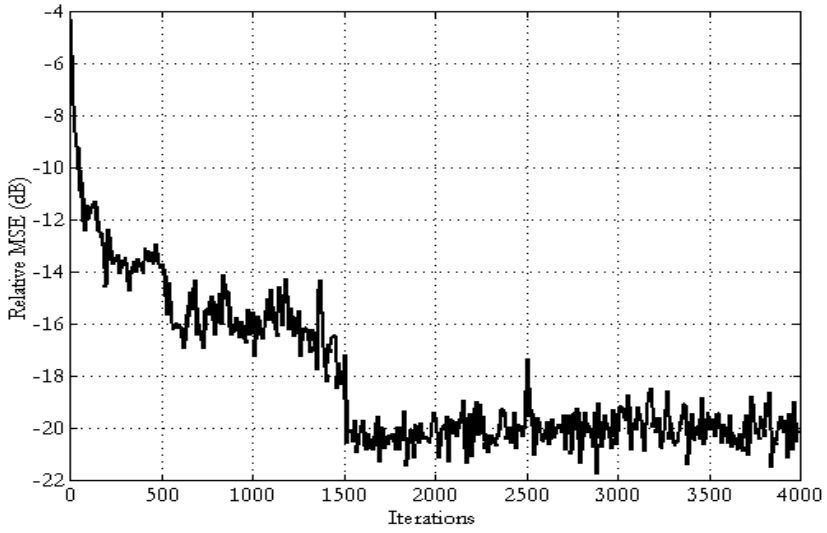


Figure 3.24 MSE relative to filter output for the DLS algorithm applied to a 5th order analog filter.

with respect to dc offsets. Furthermore, the rate of convergence is not a limiting factor in many digital communications applications where analog adaptive filters are often used.

The error signal required can be subsampled below the Nyquist rate to simplify the sampling circuitry. The effect of quantization of the filter parameters was also discussed. The adaptive algorithm was tested using both simulations and in hardware. In the hardware tests, the DLS algorithm successfully adapted the poles and zeros of a continuous time 5th order analog filter.

3.12 Appendix - Proof of Eqn. (3.30)

Eqn. (3.30) is restated here for the case $\delta_i = -\Delta$. (The proof for $\delta_i = \Delta$ is identical.)

$$E[e^2(k)]\Big|_{\delta_i = -\Delta} = \varepsilon(\left[p_1\ p_2\ \ldots\ p_i - \Delta\ \ldots\right]^T) + \Delta^2(\mathrm{tr}(\boldsymbol{R}) - r_{ii}) \tag{3.62}$$

The left-hand side of Eqn. (3.62) is the expected value of the squared error including only those times when $\delta_i = -\Delta$. So, $\boldsymbol{p}'$ takes on all of these values:

$$\boldsymbol{p}' = \left[p_1 \pm \Delta\ p_2 \pm \Delta\ \ldots\ p_i - \Delta\ \ldots\ p_N \pm \Delta\right]^T \tag{3.63}$$

Since the perturbations have no dc content and are independent, all of the $2(N-1)$ values in Eqn. (3.63) occur with equal frequency. Therefore,

$$E[e^2(k)]\Big|_{\delta_i = -\Delta} = \frac{1}{2(N-1)}\sum\varepsilon(\left[p_1 \pm \Delta\ p_2 \pm \Delta\ \ldots\ p_i - \Delta\ \ldots\ p_N \pm \Delta\right]^T) \tag{3.64}$$

The summation in Eqn. (3.64) is taken over all $2(N-1)$ possible choices of the ± signs. The right-hand side of Eqn. (3.64) can be rewritten in terms of the translated parameter vector, $\boldsymbol{v} = \boldsymbol{p} - \boldsymbol{p}^*$,

$$E[e^2(k)]\Big|_{\delta_i = -\Delta} = \frac{1}{2(N-1)}\sum\varepsilon(\left[v_1 \pm \Delta\ v_2 \pm \Delta\ \ldots\ v_i - \Delta\ \ldots\ v_N \pm \Delta\right]^T) \tag{3.65}$$

Let,

$$\boldsymbol{v}_a \equiv \left[v_1\ v_2\ \ldots\ v_i - \Delta\ \ldots\right]^T \tag{3.66}$$

$$\boldsymbol{v}_b \equiv \left[\pm\Delta\ \ldots\ \pm\Delta\ 0\ \pm\Delta\ \ldots\ \pm\Delta\right]^T, (N \times 1) \text{ vector with 0 in the } i\text{th position} \tag{3.67}$$

$$\therefore \left[v_1 \pm \Delta\ v_2 \pm \Delta\ \ldots\ v_i - \Delta\ \ldots\ v_N \pm \Delta\right]^T = \boldsymbol{v}_a + \boldsymbol{v}_b \tag{3.68}$$

There are $2(N-1)$ possible values of $\boldsymbol{v}_b$ corresponding to the $2(N-1)$ terms in the summation of Eqn. (3.65). Substituting Eqn. (3.68) into Eqn. (3.65),

$$E[e^2(k)]\Big|_{\delta_i = -\Delta} = \frac{1}{2(N-1)} \sum_{\text{all } \boldsymbol{v}_b} \varepsilon(\boldsymbol{v}_a + \boldsymbol{v}_b) \tag{3.69}$$

Using the quadratic MSE relationship from Eqn. (3.18),

$$\begin{aligned} E[e^2(k)]\Big|_{\delta_i = -\Delta} &= \frac{1}{2(N-1)} \sum_{\text{all } \boldsymbol{v}_b} (\varepsilon_{\min} + (\boldsymbol{v}_a + \boldsymbol{v}_b)^T \boldsymbol{R} (\boldsymbol{v}_a + \boldsymbol{v}_b)) \\ &= \varepsilon_{\min} + \boldsymbol{v}_a{}^T \boldsymbol{R} \boldsymbol{v}_a + \frac{1}{2(N-1)} \sum_{\text{all } \boldsymbol{v}_b} (2\boldsymbol{v}_a{}^T \boldsymbol{R} \boldsymbol{v}_b + \boldsymbol{v}_b{}^T \boldsymbol{R} \boldsymbol{v}_b) \\ &= \varepsilon(\boldsymbol{v}_a) + \frac{1}{(N-1)} \sum_{\text{all } \boldsymbol{v}_b} (\boldsymbol{v}_a{}^T \boldsymbol{R} \boldsymbol{v}_b) + \frac{1}{2(N-1)} \sum_{\text{all } \boldsymbol{v}_b} (\boldsymbol{v}_b{}^T \boldsymbol{R} \boldsymbol{v}_b) \end{aligned} \tag{3.70}$$

The summation in the second term of Eqn. (3.70) cancels out.

$$\frac{1}{(N-1)} \sum_{\text{all } \boldsymbol{v}_b} (\boldsymbol{v}_a{}^T \boldsymbol{R} \boldsymbol{v}_b) = \frac{\boldsymbol{v}_a{}^T \boldsymbol{R}}{(N-1)} \cdot \sum_{\text{all } \boldsymbol{v}_b} \boldsymbol{v}_b = \frac{\boldsymbol{v}_a{}^T \boldsymbol{R}}{(N-1)} \cdot (0) = 0 \tag{3.71}$$

Expanding the expression $(\boldsymbol{v}_b{}^T \boldsymbol{R} \boldsymbol{v}_b)$ in the third term of Eqn. (3.70),

$$\begin{aligned} \boldsymbol{v}_b{}^T \boldsymbol{R} \boldsymbol{v}_b &= \left[\pm\Delta \ \ldots \ \pm\Delta \ 0 \ \pm\Delta \ \ldots \ \pm\Delta\right] \boldsymbol{R} \left[\pm\Delta \ \ldots \ \pm\Delta \ 0 \ \pm\Delta \ \ldots \ \pm\Delta\right]^T \\ &= (\Delta^2 r_{11} + \ldots + \Delta^2 r_{(i-1)(i-1)} + \Delta^2 r_{(i+1)(i+1)} + \ldots + \Delta^2 r_{NN}) + \\ &\quad + (\pm 2\Delta^2 r_{12} \pm 2\Delta^2 r_{13} \pm 2\Delta^2 r_{23} \pm \ldots) \\ &= \Delta^2(\text{tr}(\boldsymbol{R}) - r_{ii}) + \{\ldots \text{cross terms} \ldots\} \end{aligned} \tag{3.72}$$

The "cross terms" in Eqn. (3.72) cancel each other over the summation in the third term of Eqn. (3.70). Using this fact, and substituting Eqn. (3.71) into Eqn. (3.72) gives the desired result,

$$\begin{aligned} E[e^2(k)]\Big|_{\delta_i = -\Delta} &= \varepsilon(\boldsymbol{v}_a) + \Delta^2(\text{tr}(\boldsymbol{R}) - r_{ii}) \\ &= \varepsilon(\left[p_1 \ p_2 \ \ldots \ p_i - \Delta \ \ldots\right]^T) + \Delta^2(\text{tr}(\boldsymbol{R}) - r_{ii}) \end{aligned} \tag{3.73}$$

3.13 References

[1] C.-P. J. Tzeng, "An Adaptive Offset Cancellation Technique for Adaptive Filters," *IEEE Trans. on Acoust., Speech, Signal Processing*, vol. 38, pp. 799-803, May 1990.

[2] D. A. Johns, W. M .Snelgrove, and A. S. Sedra, "Continuous-Time LMS Adaptive Recursive Filters," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 769-778, July 1991.

[3] K. A. Kozma, D. A. Johns, and A. S. Sedra, "Automatic Tuning of Continuous-Time Integrated Filters Using an Adaptive Filter Technique," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 1241-1248, Nov. 1991.

[4] T. Kwan and K. Martin, "An Adaptive Analog Continuous-Time CMOS Biquadratic Filter," *IEEE J. Solid-State Circuits*, vol. 26, pp. 859-867, June 1991.

[5] A. Shoval, W. M. Snelgrove, and D. A. Johns, "A 100Mb/s BiCMOS Adaptive Pulse-Shaping Filter," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1692-1702, Dec. 1995.

[6] J. Everitt, J. F. Parker, P. Hurst, D. Nack, and K. R. Konda, "A CMOS Transceiver for 10-Mb/s and 100-Mb/s Ethernet," *IEEE J. Solid-State Circuits*, vol. 33, pp. 2169-2177, Dec. 1998.

[7] B. Widrow and S.D. Stearns, *Adaptive Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall, 1985.

[8] B. Widrow and J. M. McCool, "A Comparison of Adaptive Algorithms Based on the Methods of Steepest Descent and Random Search," *IEEE Trans. on Antennas and Propagation*, vol. AP-24, pp. 615-637, Sept. 1976.

[9] S. Kiriaki, T. L. Viswanathan, G. Feygin, B. Staszewski, R. Pierson, B. Krenik, M. de Wit, and K. Nagaraj, "A 160-MHz Analog Equalizer for Magnetic Disk Read Channels," *IEEE J. Solid-State Circuits*, vol. 32, pp. 1839-1850, Nov. 1997.

[10] D. A. Johns, W. M. Snelgrove and A. S. Sedra, "Orthonormal Ladder Filters," *IEEE Trans. Circuits Syst.*, vol. 36, pp. 337-343, March 1989.

[11] K. Kozma, D. A. Johns, A. S. Sedra, "Automatic Tuning of Continuous-Time Integrated filters Using an Adaptive Filter Technique," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 1241-1248, Nov. 1991.

[12] J.H. van Lint, R.M. Wilson, *A Course in Combinatorics*, New York, NY: Cambridge University Press, 1992.

Chapter 4

Filter Adaptation Using Co-Ordinate Transformations

4.1 Introduction

The LMS algorithm is a popular technique for adapting digital filters in an unknown or time varying environment. Although algorithms with faster convergence properties exist, the LMS algorithm has remained popular because it offers reasonable performance with a straightforward hardware implementation. Unfortunately, the LMS algorithm has practical problems in the analog domain due to dc offset effects [1], [2]. Digital implementations of the algorithm are possible, but they require access to digital gradient signals which in turn must be produced by additional high-speed A/D converters. This chapter describes techniques for obtaining the digital gradient signals required for LMS adaptation without access to the filter's internal state signals. Emphasis is placed upon reducing the adaptation hardware requirements. Also, since adapting the poles of a continuous time filter can cause instability in the signal path and in the adaptation algorithm, this chapter considers the more practical case of an adaptive linear combiner where only the filter zeros are adapted.

In Section 4.2, two main approaches are described for FIR filters, both based on the LMS algorithm with a co-ordinate transformation applied to the adapted parameters' vector space. Both require A/D converters operating only on the input signal and error signal. The digitized input samples are stored in a shift register and multiplied by a constant matrix to obtain the gradient vector. Once the gradient signals are obtained, adap-

tation proceeds exactly as in the LMS algorithm. Analysis of the algorithms in Section 4.3 reveals that their performance is identical to the traditional LMS algorithm in terms of misadjustment and convergence rate. Section 4.4 extends the results to IIR and continuous time filters and Section 4.5 verifies the theoretical results with behavioral simulations. In Section 4.6, techniques for reducing the complexity of a practical implementation are presented. Although in many applications an A/D converter will be operating on the filter output or error signal anyway, an additional high-speed A/D converter at the filter input is very undesirable. Signed algorithms are presented which allow the A/D converters to be replaced by comparators. Then, it is shown that the digitizers and digital circuitry can be subsampled at a slower rate. Both changes result in slower convergence of the adapted parameters but greatly simplified circuit design. Finally, experimental results from an integrated analog filter are presented in Section 4.7.

4.2 FIR Filter Adaptation Using Co-Ordinate Transformations

4.2.1 The LMS Algorithm with a Co-ordinate Transform

The adaptive linear combiner (ALC) is a popular structure because it has a unimodal mean squared error performance surface (where the error is with respect to a reference signal, d, as before) which guarantees convergence to a global minimum using the LMS algorithm. An ALC with N parameters, p_i, is shown in Fig. 4.1.

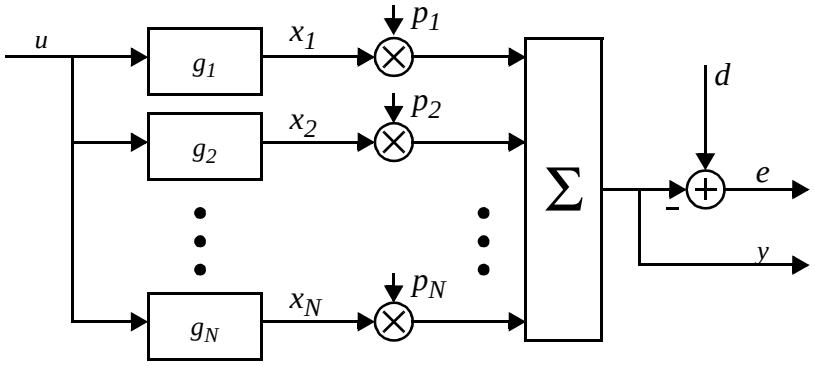


Figure 4.1 An FIR N-Parameter Adaptive Linear Combiner

For now, assume the impulse responses g_i are discrete time and finite in duration (i.e. $g_i(k) = 0$ for $k < 0$ and $k \geq M$). Therefore, the ALC in Fig. 4.1 is equivalent to the M-parameter transversal filter shown in Fig. 4.2. The vector of transversal filter parameters, $\boldsymbol{q}$, is related to the ALC parameters, $\boldsymbol{p}$, via equation Eqn. (4.1).

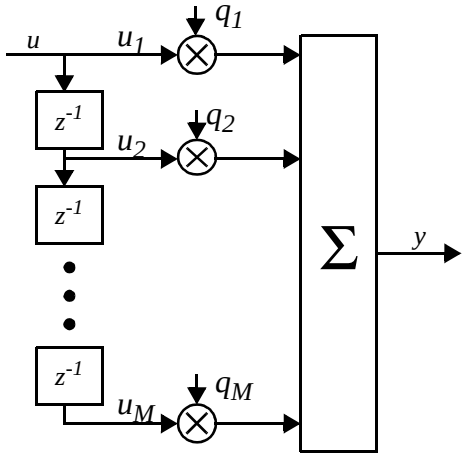


Figure 4.2 An M-Parameter Transversal Filter

$$\boldsymbol{q} = \boldsymbol{G}\boldsymbol{p} \tag{4.1}$$

where $\boldsymbol{p} = \begin{bmatrix} p_1 \\ \vdots \\ p_N \end{bmatrix}$, $\boldsymbol{q} = \begin{bmatrix} q_1 \\ \vdots \\ q_M \end{bmatrix}$, and $\boldsymbol{G} = \begin{bmatrix} g_1(0) & \cdots & g_N(0) \\ \vdots & & \vdots \\ g_1(M-1) & \cdots & g_N(M-1) \end{bmatrix}$

The LMS algorithm for an ALC is described by the following iterative equation,

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) + 2\mu e(k)\,\mathrm{f}(k) \tag{4.2}$$

The gradient signals $\phi_i(k)$ for the ALC in Fig. 4.1 are equal to the filter's state signals,

$$\phi_i(k) = x_i(k) \tag{4.3}$$

Substituting Eqn. (4.3) into Eqn. (4.2), the LMS update equation for an ALC is,

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) + 2\mu e(k)\boldsymbol{x}(k) \tag{4.4}$$

As shown in Fig. 4.1, the state signals are the outputs of FIR filters whose impulse responses are the columns of $\boldsymbol{G}$. So,

$$\boldsymbol{x}(k) = \boldsymbol{G}^T\boldsymbol{u}(k) \tag{4.5}$$

where $\boldsymbol{u}(k)$ is the vector of time delayed input samples,

$$\begin{aligned} \boldsymbol{u}(k) &= \left[u_1 \ \ldots \ u_M\right]^T \\ &= \left[u(k) \ u(k-1) \ \ldots \ u(k-M+1)\right]^T \end{aligned} \tag{4.6}$$

and $\boldsymbol{x}(k)$ is the vector of ALC state signals,

$$\boldsymbol{x}(k) = \left[x_1(k) \ x_2(k) \ \ldots \ x_N(k)\right]^T \tag{4.7}$$

Substituting Eqn. (4.5) into Eqn. (4.4) yields the following,

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) + 2\mu e(k)\boldsymbol{G}^T\boldsymbol{u}(k) \tag{4.8}$$

Eqn. (4.8) can be used to iteratively adapt the parameters of an ALC. The resulting algorithm will be called the LMS algorithm with a co-ordinate transform (LMS-CT) since the input vector $\boldsymbol{u}(k)$ is transformed to the state vector $\boldsymbol{x}(k)$ by the matrix $\boldsymbol{G}^T$.

The LMS-CT algorithm allows one to adapt an ALC without access to the filter's internal state signals. Only the filter input and the error signal must be sampled. This can be accomplished with two A/D converters operating at or above the input's Nyquist rate. The matrix multiplication expressed in Eqn. (4.5) is equivalent to estimating the state signals by operating N digital FIR filters in parallel with the signal path filter, as shown in Fig. 4.3. The adaptation hardware required is similar to that required by the basic LMS algorithm, except that a $N \times M$ matrix multiplication must be performed on the input sample vector. The matrix entries are the impulse responses $g_i(k)$ which remain constant throughout adaptation.

4.2.2 The LMS Algorithm with an Inverse Co-ordinate Transform

In Eqn. (4.1) the matrix $\boldsymbol{G}$ maps the space of all possible ALC parameter vectors, $\boldsymbol{p} \in \Re^N$, to the space of equivalent FIR filters, $\boldsymbol{q} \in \Re^M$. A direct inverse mapping is impossible when $M > N$, which is likely for any practical filter. However, an approximate inverse mapping which minimizes the squared error is given by Eqn. (4.9) and Eqn. (4.10).

$$\hat{\boldsymbol{p}} = \boldsymbol{K}\boldsymbol{q} \tag{4.9}$$

$$\text{where,} \quad \boldsymbol{K} = (\boldsymbol{G}^T\boldsymbol{G})^{-1}\boldsymbol{G}^T \tag{4.10}$$

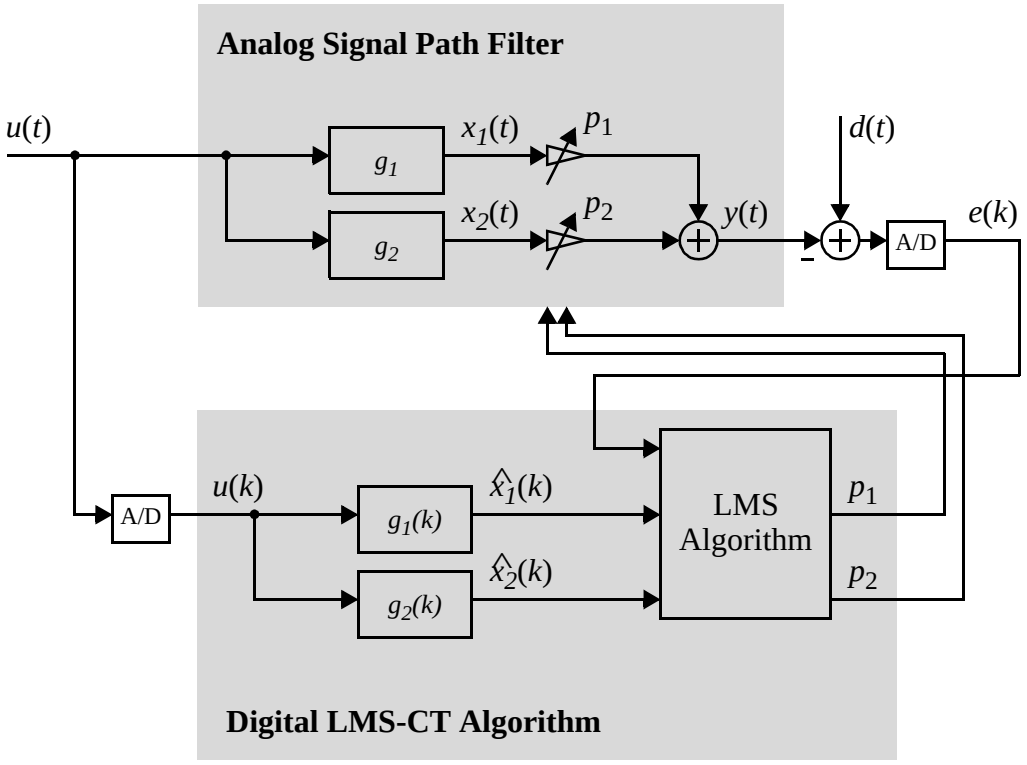


Figure 4.3 LMS-CT algorithm for a 2-parameter ALC.

The matrix $\boldsymbol{K}$ is the "pseudoinverse" of $\boldsymbol{G}$ [3]. It will be used to map gradient estimates for a FIR filter (which are easily obtained from samples of the input) to gradient estimates for the ALC.

The LMS algorithm for an adaptive transversal filter is described by the following update equation:

$$\boldsymbol{q}(k+1) = \boldsymbol{q}(k) + 2\mu e(k)\boldsymbol{u}(k) \tag{4.11}$$

Left-multiplying both sides of Eqn. (4.11) by $\boldsymbol{K}$ and dropping the hats for convenience results in the following update rule:

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) + 2\mu e(k)\boldsymbol{K}\boldsymbol{u}(k) \tag{4.12}$$

The algorithm defined by Eqn. (4.12) will be referred to as the LMS algorithm with an inverse co-ordinate transform (LMS-ICT). Like the LMS-CT algorithm, only the filter input and error signal must be sampled. The computational complexity is identical for

the LMS-CT and LMS-ICT algorithms. However, the LMS-ICT algorithm has a significant advantages when it comes to implementation as we shall see in Section 4.6.2.

4.3 Convergence and Misadjustment Analysis

Since the state estimates obtained by the LMS-CT algorithm are exact, it will perform exactly the same as the full LMS algorithm. Therefore, it can be guaranteed stable under all of the same conditions as the full LMS algorithm.

An analysis of the LMS-ICT algorithm, however, is not so simple. Basically, the algorithm assumes that the adapted filter has a transversal structure and uses $2e\boldsymbol{u}$ as an unbiased estimate of the gradient $\nabla_\varepsilon \boldsymbol{q} \in \Re^M$, $\varepsilon = E[e^2]$. However, the ALC impulse response is restricted to a N-dimensional subspace of $\Re^M$, namely range($\boldsymbol{G}$). In order to update the parameter vector $\boldsymbol{p}$, $2e\boldsymbol{u}$ must be mapped from $\Re^M$ back into range($\boldsymbol{G}$). The mapping which performs this with the smallest squared error is $\boldsymbol{K} \cdot 2e\boldsymbol{u}$. The expected value of the resulting gradient estimate is

$$\begin{aligned} E[\boldsymbol{K} \cdot 2e\boldsymbol{u}] &= \boldsymbol{K} \cdot E[2e\boldsymbol{u}] \\ &= \boldsymbol{K} \cdot \nabla_\varepsilon \boldsymbol{q} \end{aligned} \tag{4.13}$$

In Eqn. (4.13) we have used the fact that $E[2e\boldsymbol{u}] = \nabla_\varepsilon \boldsymbol{q}$ since $2e\boldsymbol{u}$ is the unbiased gradient estimate in an LMS adaptive transversal filter. Unfortunately, $\boldsymbol{K} \cdot \nabla_\varepsilon \boldsymbol{q}$ is not necessarily parallel to $\nabla_\varepsilon \boldsymbol{p}$ causing gradient misalignment.

Nevertheless, it can be shown that the expected value of the ALC parameter vector converges to the optimal value, $\boldsymbol{p}^*$, as $k \to \infty$. Taking the expectation of both sides of Eqn. (4.12) yields,

$$E[\boldsymbol{p}(k+1)] = E[\boldsymbol{p}(k)] + 2\mu \boldsymbol{K} \cdot E[e(k)\boldsymbol{u}(k)] \tag{4.14}$$

The term $E[e(k)\boldsymbol{u}(k)]$ can be rewritten in terms of the optimal transversal filter parameters $\boldsymbol{q}^*$, the input autocorrelation matrix $\boldsymbol{R} \equiv E[\boldsymbol{u}\boldsymbol{u}^T]$, and $E[\boldsymbol{q}(k)]$ using the Wiener-Hopf equation and assuming $\boldsymbol{q}(k)$ is independent of $\boldsymbol{u}(k)$ [1],

1. The independence assumption is often invoked in statistical analyses of the LMS algorithm [4], [5], [6] and leads to reliable theoretical predictions of performance, even when there is some dependence between $\boldsymbol{q}(k)$ and $\boldsymbol{u}(k)$.

$$\begin{aligned} E[e(k)\boldsymbol{u}(k)] &= E[d(k)\boldsymbol{u}(k)] - E[y(k)\boldsymbol{u}(k)] \\ &= \boldsymbol{R}\boldsymbol{q}^* - E[\boldsymbol{u}(k)\boldsymbol{u}^T(k)\boldsymbol{q}(k)] \\ &= \boldsymbol{R}\boldsymbol{q}^* - \boldsymbol{R}E[\boldsymbol{q}(k)] \end{aligned} \tag{4.15}$$

Substituting Eqn. (4.15) into Eqn. (4.14) yields,

$$\begin{aligned} E[\boldsymbol{p}(k+1)] &= E[\boldsymbol{p}(k)] + 2\mu\boldsymbol{K}\cdot(\boldsymbol{R}\boldsymbol{q}^* - \boldsymbol{R}E[\boldsymbol{q}(k)]) \\ &= E[\boldsymbol{p}(k)] + 2\mu\boldsymbol{K}\cdot(\boldsymbol{R}\boldsymbol{q}^* - \boldsymbol{R}\boldsymbol{G}E[\boldsymbol{p}(k)]) \\ &= (\boldsymbol{I} - 2\mu\boldsymbol{K}\boldsymbol{R}\boldsymbol{G})E[\boldsymbol{p}(k)] + 2\mu\boldsymbol{K}\boldsymbol{R}\boldsymbol{q}^* \end{aligned} \tag{4.16}$$

Using the Wiener-Hopf equation again allows us to relate $\boldsymbol{p}^*$ and $\boldsymbol{q}^*$,

$$\begin{aligned} E[\boldsymbol{x}\boldsymbol{x}^T]\boldsymbol{p}^* &= E[d\boldsymbol{x}] \\ \boldsymbol{G}^T E[\boldsymbol{u}\boldsymbol{u}^T]\boldsymbol{G}\boldsymbol{p}^* &= \boldsymbol{G}^T E[d\boldsymbol{u}] \\ \boldsymbol{G}^T\boldsymbol{R}\boldsymbol{G}\boldsymbol{p}^* &= \boldsymbol{G}^T\boldsymbol{R}\boldsymbol{q}^* \\ (\boldsymbol{G}^T\boldsymbol{G})^{-1}\boldsymbol{G}^T\boldsymbol{R}\boldsymbol{G}\boldsymbol{p}^* &= (\boldsymbol{G}^T\boldsymbol{G})^{-1}\boldsymbol{G}^T\boldsymbol{R}\boldsymbol{q}^* \\ \boldsymbol{K}\boldsymbol{R}\boldsymbol{G}\boldsymbol{p}^* &= \boldsymbol{K}\boldsymbol{R}\boldsymbol{q}^* \end{aligned} \tag{4.17}$$

Substituting Eqn. (4.17) into Eqn. (4.16) gives

$$E[\boldsymbol{p}(k+1)] = (\boldsymbol{I} - 2\mu\boldsymbol{K}\boldsymbol{R}\boldsymbol{G})E[\boldsymbol{p}(k)] + 2\mu\boldsymbol{K}\boldsymbol{R}\boldsymbol{G}\boldsymbol{p}^* \tag{4.18}$$

After performing a co-ordinate transformation to a principal axis system [5], Eqn. (4.18) can be rewritten in terms of a transformed weight-error vector, $\boldsymbol{w} = \boldsymbol{Q}^{-1}\cdot(\boldsymbol{p} - \boldsymbol{p}^*)$ where $\boldsymbol{Q}$ is the eigenvector matrix of $\boldsymbol{K}\boldsymbol{R}\boldsymbol{G}$.

$$E[\boldsymbol{w}(k)] = (\boldsymbol{I} - 2\mu\mathrm{L})^k\boldsymbol{w}(0) \tag{4.19}$$

In Eqn. (4.19), L is the diagonal eigenvalue matrix of $\boldsymbol{K}\boldsymbol{R}\boldsymbol{G}$. Eqn. (4.19) also describes the convergence of the LMS algorithm, except that L becomes the diagonal eigenvalue matrix of $\boldsymbol{R}$. Therefore, much of the analysis on LMS adaptive transversal filters can also be applied here by replacing the autocorrelation matrix $\boldsymbol{R}$ with $\boldsymbol{K}\boldsymbol{R}\boldsymbol{G}$. Specifically, if

$$\mu < \frac{1}{\lambda_{\max}}, \tag{4.20}$$

where $\lambda_{\max}$ is the largest eigenvalue of the matrix $\boldsymbol{K}\boldsymbol{R}\boldsymbol{G}$, then $(\boldsymbol{I} - 2\mu\mathrm{L})^k \to 0$ as $k \to \infty$. Hence, $E[\boldsymbol{w}(k)] \to 0$ and $E[\boldsymbol{p}(k)] \to \boldsymbol{p}^*$. Furthermore, the time constant of

decay of the MSE (in terms of the sampling time) and the steady state misadjustment can be shown to be [5],

$$\tau_{\text{MSE}} = \frac{1}{4\mu} \cdot \left(\frac{1}{\lambda}\right)_{\text{avg}} \tag{4.21}$$

$$\text{Misadjustment} = \mu \sum_i \lambda_i = \mu \cdot \text{tr}(\boldsymbol{KRG}) \tag{4.22}$$

4.4 Extension to IIR and Continuous Time Filters

In this section, we will consider the ALC shown in Fig. 4.4 where the h_i are IIR or continuous time filters.

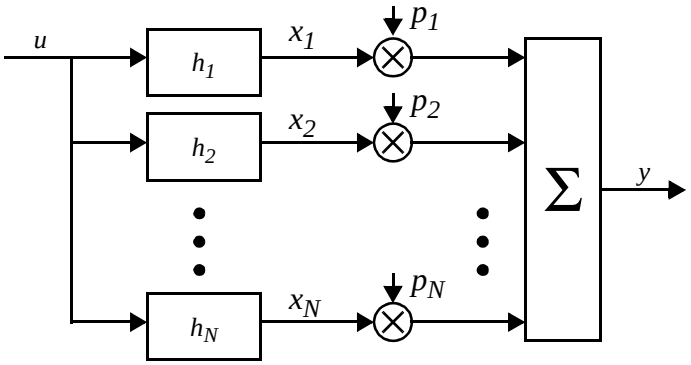


Figure 4.4 An N-parameter IIR adaptive linear combiner with independent impulse responses, h_i.

If the h_i have infinite duration discrete time impulse responses, they can be approximated by truncated impulse responses $g_i(k)$,

$$g_i(k) = \begin{cases} h_i(k), & 0 \le k \le M-1 \\ 0, & \text{otherwise} \end{cases} \tag{4.23}$$

For continuous time filters the approximation is similar except that the continuous time impulse responses $h_i(t)$ are sampled and scaled to obtain approximately equivalent FIR filters.

$$g_i(k) = \begin{cases} T_S \cdot h_i(kT_S) & 0 \le k \le M-1 \\ 0 & \text{otherwise} \end{cases} \tag{4.24}$$

The matrix G is created from the g_i as in Eqn. (4.1) and adaptation proceeds exactly as before for either the LMS-CT or LMS-ICT algorithm.

In order to avoid aliasing in the continuous time case, the following conditions must be met:

- The sampling rate, $1/T_S$, must be greater than twice the bandwidth of the filter's input signal in order to avoid aliasing.
- The frequency responses of the filters h_i must be bandlimited below $1/T_S$.
- The impulse responses must be zero outside the range $0 \le t < MT_S$.

Of course, in practice it is impossible to meet all of these conditions simultaneously. Instead, the sampling rate is chosen so that aliasing is minimal and M is taken large enough so that most of the energy in the impulse responses h_i lie within the time interval $0 \le t < MT_S$. Therefore, for IIR or continuous time filters, Eqn. (4.5) is only approximate, as are the theoretical results for convergence and misadjustment derived in Section 4.3. Fortunately, behavioral simulations indicate that this is not a major concern since the LMS-CT and LMS-ICT algorithms are robust with respect to these approximation errors.

In practice, a continuous time filter may have gains and time constants which are not accurately known *a priori*. In this case, it may be necessary to measure the impulse responses $g_i(k)$ before determining the matrices $\boldsymbol{G}$ and $\boldsymbol{K}$. Each impulse response $g_i(k)$ can be measured by setting all of the ALC parameters to zero except for $p_i = 1$ and observing the ALC output. The matrix $\boldsymbol{G}$ or $\boldsymbol{K}$ would then be calculated and stored in a memory which is accessed during adaptation. If process variations are significant, this procedure would have to be automated and repeated on each chip.

4.5 Simulation Results

Model matching experiments were used to verify the LMS-CT and LMS-ICT algorithms on continuous time filters. All of the simulations described in this section use the block diagram shown in Fig. 4.5. An independent additive noise source n is introduced to control the steady state error after convergence. The time scale is normalized to a sampling rate of $f_S = 1$. The reference filter is a third order elliptic lowpass transfer function with 0 dB dc gain, 0.5 dB of ripple in the passband extending to $0.1f_S$, and 40 dB of stopband attenuation. The filter input is white noise bandlimited by an eighth order elliptic filter with 0.1 dB of passband ripple to $0.4f_S$ and 60 dB of stopband attenuation beyond $0.5f_S$.

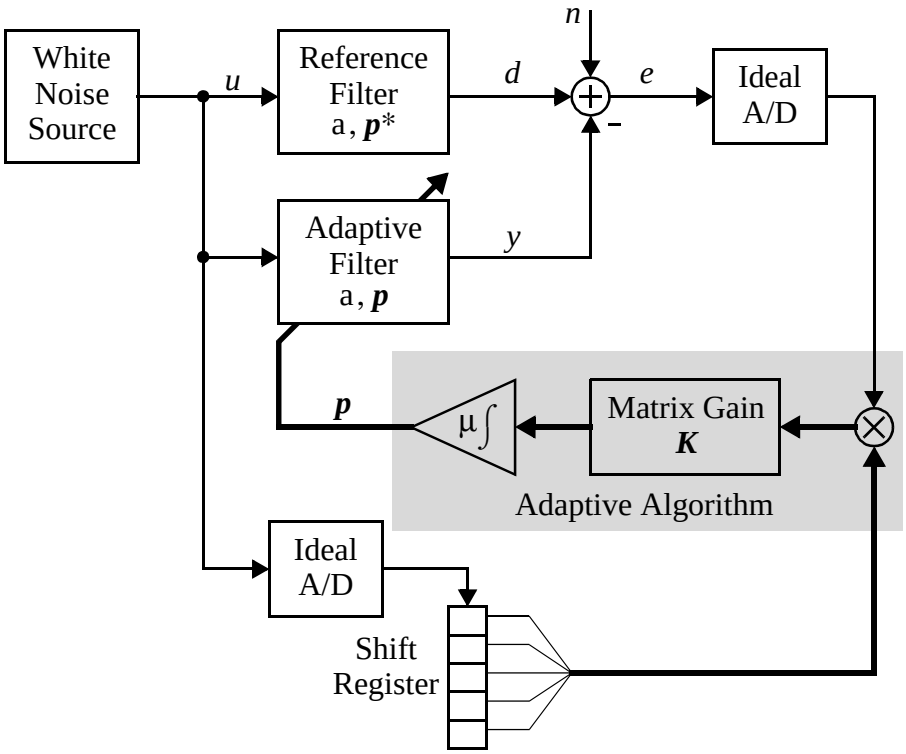


Figure 4.5 Model matching simulation for 3rd order orthonormal ladder filter.

4.5.1 Orthonormal Ladder Filter

Interestingly, when the impulse responses $g_i(k)$ are orthonormal, the LMS-CT and LMS-ICT algorithms become identical. This can be seen by arranging the impulse responses into column vectors, $\boldsymbol{g_i} = \left[g_i(1)\ g_i(2)\ \ldots\ g_i(N)\right]^T$. Since the vectors are

orthonormal, $\boldsymbol{g_i^T} \cdot \boldsymbol{g_k} = 0$ for $i \neq k$ and $\boldsymbol{g_i^T} \cdot \boldsymbol{g_i} = 1$. By substituting $\boldsymbol{G} = \left[\boldsymbol{g_1}\ \boldsymbol{g_2}\ \cdots\ \boldsymbol{g_N}\right]$ into Eqn. (4.10) it is easily verified that $\boldsymbol{K} = \boldsymbol{G}^T$,

$$\begin{aligned} \boldsymbol{K} &= \left(\left[\boldsymbol{g_1}\ \boldsymbol{g_2}\ \cdots\ \boldsymbol{g_N}\right] T \cdot \left[\boldsymbol{g_1}\ \boldsymbol{g_2}\ \cdots\ \boldsymbol{g_N}\right] \right)^{-1} \cdot \boldsymbol{G}^T \\ &= (\boldsymbol{I}_N)^{-1} \cdot \boldsymbol{G}^T \\ &= \boldsymbol{G}^T \end{aligned} \tag{4.25}$$

Orthonormal ladder filters have this property [7]. A third order continuous time orthonormal ladder structure is shown in Fig. 4.6. By making the feed-in parameters $\boldsymbol{p}$

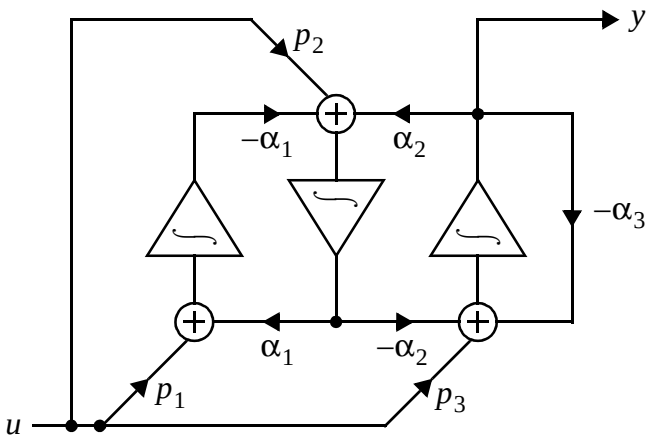


Figure 4.6 A 3rd order orthonormal ladder filter using multiple feed-ins of the input signal.

adaptive, the structure becomes an adaptive linear combiner. Filters with adaptive feed-ins are particularly interesting examples because the state signals required for traditional LMS adaptation are not available anywhere in the system. In order to perform LMS adaptation, it would be necessary to operate a second filter in parallel with the first just to obtain the gradient signals [1]. In addition to the extra complexity and power consumption which this implies, mismatches between the two filters result in dc offsets which limit the accuracy of the adaptation. Fortunately the LMS-CT and LMS-ICT algorithms can be used without access to the filter's internal states.

In order to achieve the desired pole locations, the feedback parameters for both the adaptive filter and reference filter were fixed at $\mathrm{a} = \left[0.4905\ 0.6025\ 0.7789\right]$. The

feed-in parameters for the reference filter were fixed at $\boldsymbol{p}^* = \begin{bmatrix} 0.5940 & 0 & 0.0493 \end{bmatrix}^T$.

The impulse responses $h_1(t)$, $h_2(t)$ and $h_3(t)$ of the ALC were measured by setting $\boldsymbol{p} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T$, $\begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$ and $\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$. The sampled impulse responses are plotted in Fig. 4.7 which shows that $M = 20$ samples are sufficient to capture at least 99.8% of the impulse response power.

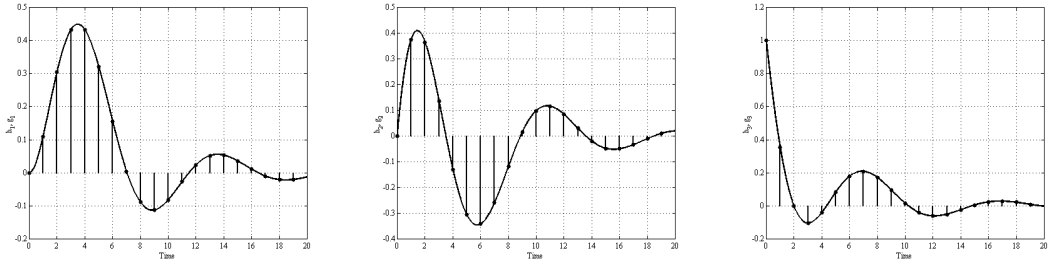


Figure 4.7 Truncated and sampled impulse responses for the 3rd order orthonormal ladder.

The $\boldsymbol{G}$ matrix was then constructed as described in Section 4.4,

$$\boldsymbol{G}^T = \begin{bmatrix} 0 & 0.110 & 0.304 & 0.432 & \dots \\ 0 & 0.374 & 0.365 & 0.137 & \dots \\ 1.000 & 0.355 & -0.002 & -0.104 & \dots \end{bmatrix}, \tag{4.26}$$

and the pseudoinverse $\boldsymbol{K}$ is calculated from Eqn. (4.10),

$$\boldsymbol{K} = \begin{bmatrix} 0.001 & 0.171 & 0.473 & 0.673 & \dots \\ 0.059 & 0.609 & 0.572 & 0.208 & \dots \\ 0.790 & 0.303 & 0.020 & -0.074 & \dots \end{bmatrix}. \tag{4.27}$$

The rows of $\boldsymbol{K}$ are plotted in Fig. 4.8. Except for a scaling factor for normalization, the waveforms are similar to those plotted in Eqn. (4.7) for the columns of $\boldsymbol{G}$. The waveforms would be identical if the columns of $\boldsymbol{G}$ were perfectly orthogonal. However, the frequency response of h_3 extends beyond the Nyquist rate (only 12 dB of attenuation at $f_S/2$) which causes aliasing in $g_3(k)$.

First, simulations were performed with the noise source turned off ($n = 0$). As can be seen from Fig. 4.9, both the LMS-CT and the LMS-ICT converged to their optimal

parameter values with zero steady state error. The errors incurred by aliasing and truncating the impulse responses had no effect on the result.

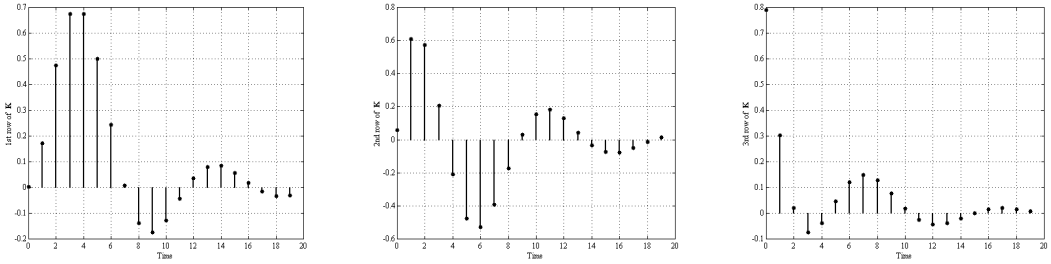
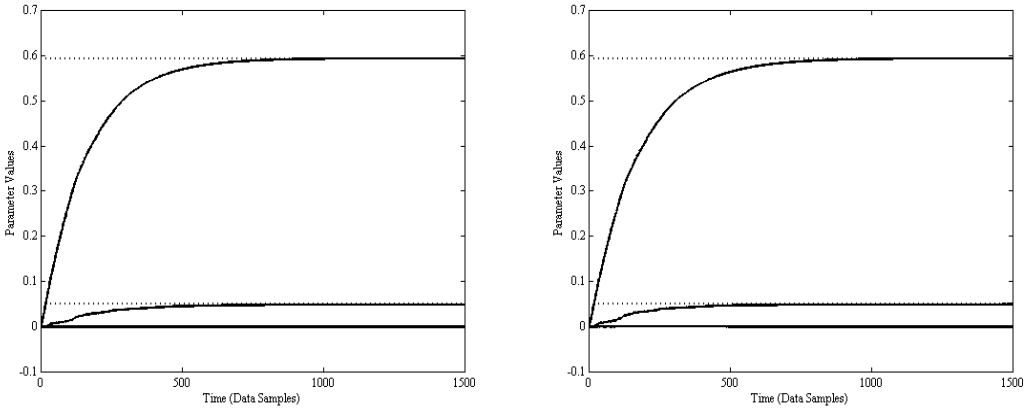


Figure 4.8 Rows of the matrix *K* plotted versus time.



(A) (B)

Figure 4.9 Sample learning curves for the parameters in a 3rd order model matching experiment.

(A) *the LMS-CT algorithm with* $\mu = 0.001$ *and* (B) *the LMS-ICT algorithm with* $\mu = 0.0006$ *and no steady state error.*

Next, some finite steady state error was introduced via n to examine the algorithms' misadjustment. A noise power of $\text{var}(n) = 0.001$ was used, which is about 13.5 dB less than the output power of the reference filter, $\text{var}(d)$. The input autocorrelation matrix is a 20×20 matrix, $\boldsymbol{R} = E[\boldsymbol{u}\boldsymbol{u}^T]$, which can be calculated from a knowledge of the input statistics,

$$\boldsymbol{R} = \begin{bmatrix} 0.0813 & 0.0166 & -0.0142 & \dots & -0.0012 \\ 0.0166 & 0.0813 & 0.0166 & \dots & 0.0008 \\ -0.0142 & 0.0166 & 0.0813 & \dots & -0.0001 \\ . & . & . & & . \\ -0.0012 & 0.0008 & -0.0001 & \dots & 0.0813 \end{bmatrix} \tag{4.28}$$

The convergence properties of the LMS-ICT algorithm are determined by the eigenvalues of $\boldsymbol{KRG}$,

$$\lambda_{\text{LMS-ICT}} = \text{eig}(\boldsymbol{KRG}) = 0.0993, 0.0988, 0.0967 . \tag{4.29}$$

Designing for a misadjustment of 10% using Eqn. (4.22), the required value of μ is

$$\mu_{\text{LMS-ICT}} = \frac{0.1}{\sum \lambda_{\text{LMS-ICT}}} = 0.3453 \tag{4.30}$$

For the LMS and LMS-CT algorithms, the eigenvalues are all identically $\lambda = 0.0607$ owing to the orthonormal filter structure. For 10% misadjustment, the value of μ is,

$$\mu_{\text{LMS and LMS-CT}} = \frac{0.1}{\sum \lambda} = 0.5490 \tag{4.31}$$

These values together with Eqn. (4.21) predict that the decay time constant of the MSE should be approximately 8 iterations for all three algorithms. The simulation results plotted in Fig. 4.10 verify this result. All three converge at the same rate with the same final misadjustment. Fig. 4.11 shows simulation results for a misadjustment of 1%. Again, the rate of decay is identical.

Of course, in general the autocorrelation matrix $\boldsymbol{R}$ will not be known *a priori*. However, its knowledge was assumed here to demonstrate that the LMS, LMS-CT and LMS-ICT algorithms all have the same performance, although different values of μ may be required for each.

4.5.2 Feed Forward Companion Form Filter

In this section, the same model matching experiments are performed using a different filter structure. The filter structure is shown in Fig. 4.12. It is a third order continuous time companion form filter with variable feed-in coefficients, p_i. Again, since the feed-in parameters are adapted, the state signals required for traditional LMS adaptation are

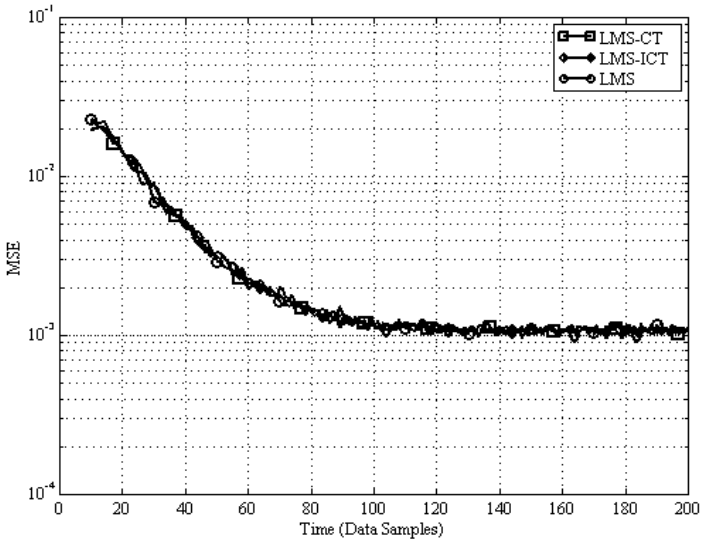


Figure 4.10 Simulation results for the 3rd order orthonormal ladder model matching experiment with an excess MSE of 10%.

The LMS-CT, LMS-ICT and full LMS algorithms were simulated. The results are averaged over an ensemble of 1000 simulation runs.

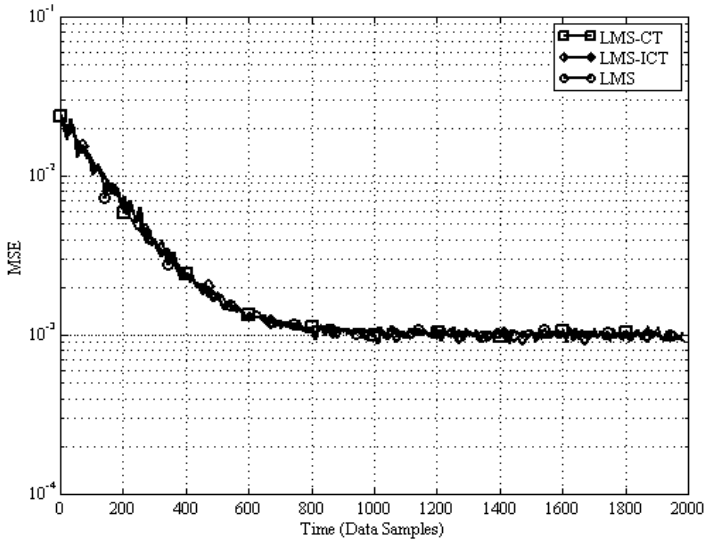


Figure 4.11 Simulation results for the 3rd order orthonormal ladder model matching experiment with an excess MSE of 1%.

The LMS-CT, LMS-ICT and full LMS algorithms were simulated. Each data point is averaged over 10 consecutive samples and 100 independent simulation runs.

not available. Unlike the orthonormal ladder filter, the impulse responses are not orthogonal (Fig. 4.13). As a result, the matrices $\boldsymbol{G}^T$ and $\boldsymbol{K}$ are quite different and there is gradient misalignment in the LMS-ICT algorithm.

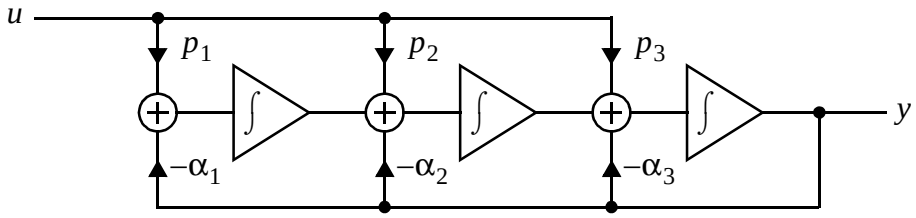


Figure 4.12 Third order feed forward companion form filter.

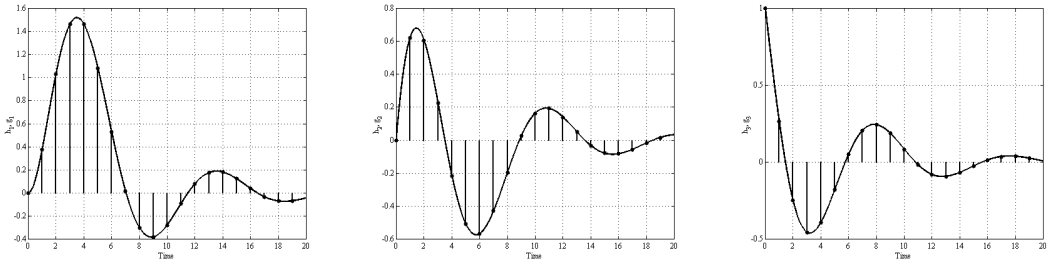
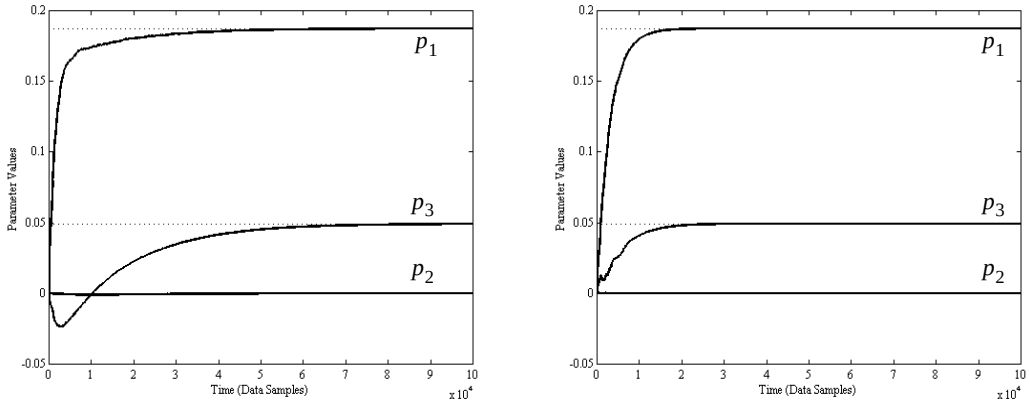


Figure 4.13 Sampled, truncated impulse responses of the 3rd order feed forward companion form filter.



(A) (B)

Figure 4.14 Model matching learning curves for a feed forward companion form filter.

(A) *The LMS-CT algorithm and* (B) *the LMS-ICT algorithm.*

Fig. 4.14 shows behavioral simulations with length $M = 20$ impulse responses and zero excess error added (i.e. $n = 0$) for both the LMS-CT and LMS-ICT algorithms. Although both algorithms converge with zero steady state error, notice that they take different trajectories. The trajectories are projected onto the $p_2 = 0$ plane and plotted along with error surface contours in Fig. 4.15. In this plot, the gradient misalignment of the LMS-ICT algorithm is evident.

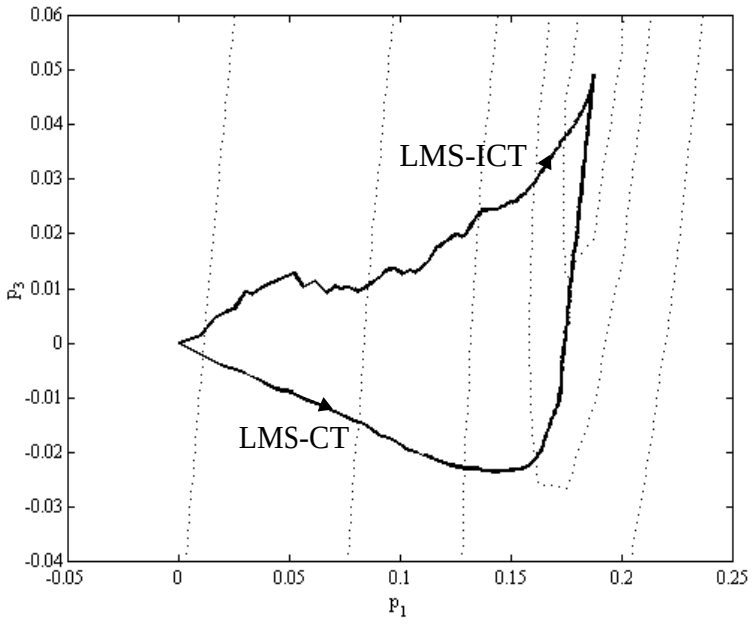


Figure 4.15 Learning trajectories of two model matching experiments on MSE contours.

4.6 Hardware Implementation Issues

Although the LMS-CT and LMS-ICT algorithms offer the same performance as the LMS algorithm without access to the adapted filter's internal states, both algorithms still require two Nyquist-rate A/D converters: one on the filter input and one on the error signal (or the filter output if the error signal is being generated digitally). Many applications will have an A/D converter at the filter output anyway (for example, for demodulation of a communication signal). However, the extra power consumption of an

additional high-speed A/D converter at the filter input may be prohibitive. This section examines techniques for further reducing the hardware complexity of these algorithms. First, the A/D converters are replaced by single-bit quantizers (comparators). Then, the possibility of subsampling the filter input and output at a slower rate is considered for the LMS-ICT algorithm.

4.6.1 Signed Algorithms

It is possible to take the sign of the error signal or the gradient signal or both in Eqn. (4.4) in order to simplify the implementation of the LMS algorithm [8]. Taking the sign of both results in the "sign-sign LMS" (SS-LMS) algorithm,

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) + 2\mu \cdot \mathrm{sgn}(e(k)) \cdot \mathrm{sgn}(\boldsymbol{x}(k)) \tag{4.32}$$

The product $-\mathrm{sgn}(e(k)) \cdot \mathrm{sgn}(\boldsymbol{x}(k))$ provides, on average, the correct sign of each gradient component. The SS-LMS algorithm proceeds by changing each parameter in fixed steps of size 2μ in a direction opposite the gradient. The digital multiplication of the error and state signals is performed by a single exclusive-OR gate, $\mathrm{sgn}(e) \cdot \mathrm{sgn}(\boldsymbol{x}) = \mathrm{sgn}(e) \oplus \mathrm{sgn}(\boldsymbol{x})$, resulting in considerable hardware savings.

The same approach can be taken to try and simplify the hardware required for the LMS-CT and LMS-ICT algorithms. Taking the sign of the error and input data signals results in the following update equations,

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) + 2\mu\boldsymbol{G}^T \cdot \mathrm{sgn}(e(k)) \cdot \mathrm{sgn}(\boldsymbol{u}(k)) \tag{4.33}$$

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) + 2\mu\boldsymbol{K} \cdot \mathrm{sgn}(e(k)) \cdot \mathrm{sgn}(\boldsymbol{u}(k)) \tag{4.34}$$

This allows the two A/D converters required at the filter input and error signal to be replaced by comparators. The result is a significant decrease in circuit complexity and power consumption. The digital signal processing required for adaptation is also simplified considerably, but the matrix entries are still multi-bit values which means that fast arithmetic logic is still required. The update equation can be further simplified by taking the sign of each entry in the matrices $\boldsymbol{G}^T$ and $\boldsymbol{K}$,

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) + 2\mu\,\mathrm{sgn}(\boldsymbol{G}^T) \cdot \mathrm{sgn}(e(k)) \cdot \mathrm{sgn}(\boldsymbol{u}(k)) \tag{4.35}$$

$$\boldsymbol{p}(k+1) = \boldsymbol{p}(k) + 2\mu\,\mathrm{sgn}(\boldsymbol{K}) \cdot \mathrm{sgn}(e(k)) \cdot \mathrm{sgn}(\boldsymbol{u}(k)) \tag{4.36}$$

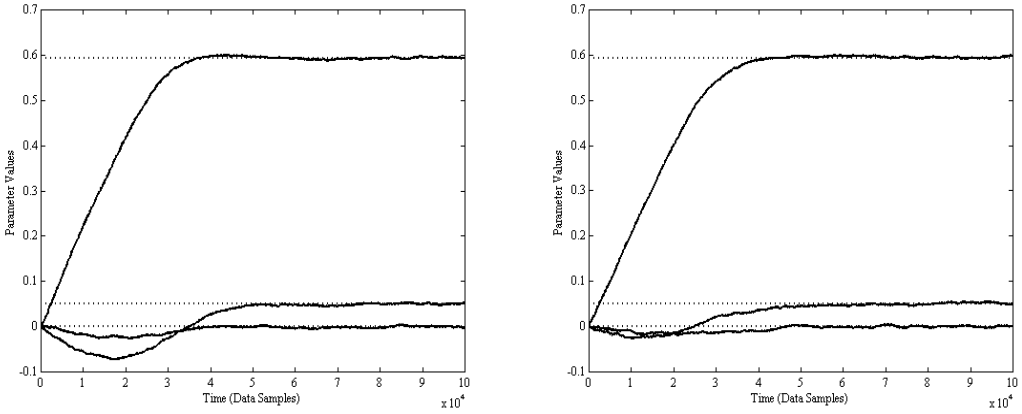
Eqn. (4.35) will be used as the update rule for the "sign-sign LMS-CT" algorithm (SS-LMS-CT) and Eqn. (4.36) for the "sign-sign LMS-ICT" algorithm (SS-LMS-ICT). The multiplication of the three signed quantities in both Eqn. (4.35) and Eqn. (4.36) can be performed by 3-input XOR gates.

To verify these algorithms, behavioral simulations were performed using the same model matching experiment as in Section 4.5.1. Simulation results are plotted in Fig. 4.16, Fig. 4.17, and Fig. 4.18. There is no noticeable difference between the performance of the SS-LMS-CT and SS-LMS-ICT algorithms. For the same misadjustment, the signed algorithms converge slower than the full LMS-CT and LMS-ICT algorithms, but this is not surprising since it is well known that the SS-LMS algorithm is slower than the full LMS algorithm. Of course, by taking a larger value of μ, the slower convergence can be traded of for increased misadjustment.

Although it is true that the SS-LMS algorithm has demonstrated instability in certain circumstances [9], its simplified hardware has proved useful in numerous applications. Stability of the SS-LMS algorithm is usually verified for a particular application via extensive simulations. Similarly, it is not immediately obvious whether the SS-LMS-CT and SS-LMS-ICT algorithms will always provide robust convergence in spite of the large quantization errors introduced on the matrix entries, the input signal, and the error signal. Therefore, extensive simulations should also be used to verify the SS-LMS-CT and SS-LMS-ICT algorithms for a particular application.

4.6.2 Subsampling

Until now, there has been little practical difference between the LMS-CT and LMS-ICT algorithms. However, in this section we shall see that a significant advantage of the LMS-ICT algorithm is that both digitizers (A/D converters for the LMS-ICT or comparators for the SS-LMS-ICT) can be subsampled. Both the input and error signal digitizer can be clocked below the Nyquist rate, thereby reducing power consumption and simplifying the analog circuit design. Although subsampling increases the time required for convergence, this is rarely a problem in high-speed applications. Subsampling also gives



(A) (B)

Figure 4.16 Sample learning curves for the parameters in a 3rd order model matching experiment using signed algorithms.

(A) *The SS-LMS-CT algorithm and* (B) *the SS-LMS-ICT algorithm, both simulated with a steady state error variance of* $\mathrm{var}(n) = 0.001$ *and* $\mu = 10^{-5}$.

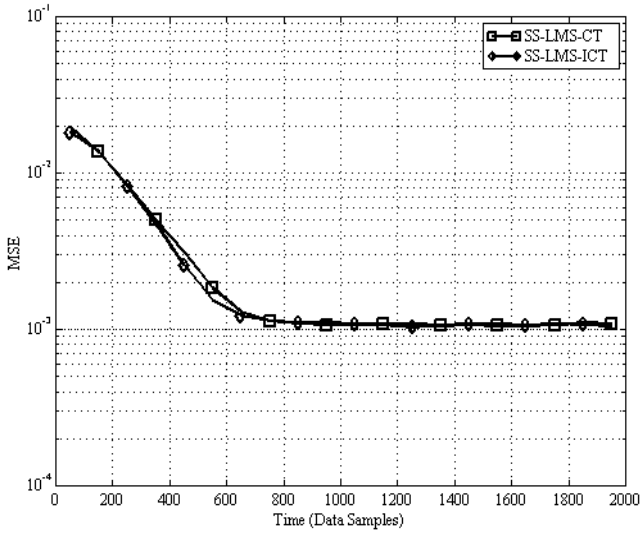


Figure 4.17 Simulation results for the 3rd order orthonormal ladder model matching experiment using signed algorithms with an excess MSE of 10%.

Both simulations have a steady state error variance of $\mathrm{var}(n) = 0.001$ *and* $\mu = 5.454 \cdot 10^{-4}$. *Each data point is averaged over 100 consecutive data samples and 100 separate simulation runs*

the adaptation circuitry more time to perform the $N \times M$ matrix multiplications required in Eqn. (4.12) and Eqn. (4.36).

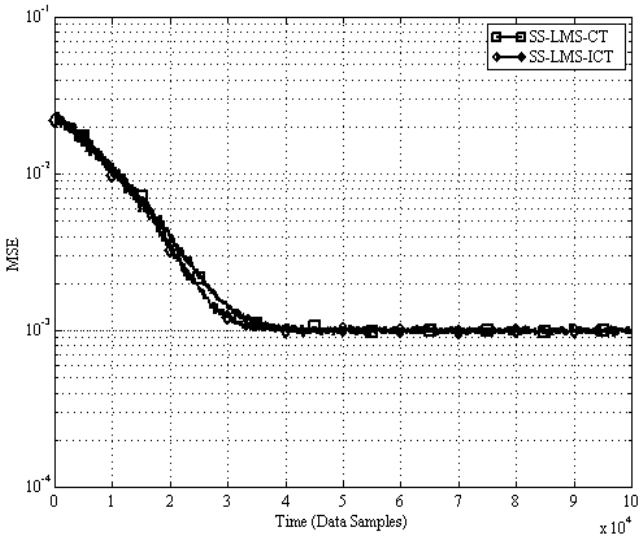


Figure 4.18 Simulation results for the 3rd order orthonormal ladder model matching experiment using signed algorithms with $\mu = 10^{-5}$.
Both simulations have a steady state error variance of $\mathrm{var}(n) = 0.001$. *Each data point is averaged over 100 consecutive data samples and 30 separate simulation runs.*

The basic idea is first described for a transversal filter structure in Section 4.6.2.1. Then, the technique is modified for use in analog adaptive linear combiners in Section 4.6.2.2.

4.6.2.1 Subsampling an LMS transversal filter

The update equation for a LMS adaptive transversal filter is given in Eqn. (4.11). Generally, when the filter is implemented in the analog domain, just one digitizer operating at the Nyquist rate is used on the filter input, u. A digital shift register is used to emulate the analog delay line as shown in Fig. 4.19 [10], [11]. At very high speeds it is desirable to operate the digitizers and all digital circuitry slower than the Nyquist rate.

Since the gradient signals in Eqn. (4.11) are unbiased estimates of the true gradient at all times, there is no reason why the LMS algorithm must iterate at each sampling instant. For instance, it is possible to iterate only at $k = 0, 2, 4, 6\ldots$ resulting in a system where the error signal is subsampled by a factor of 2. The adaptive process converges at one-half the rate, but with the same accuracy and stability properties as the full rate system. Unfortunately, this still requires a digitizer at the filter input capable of operating at the

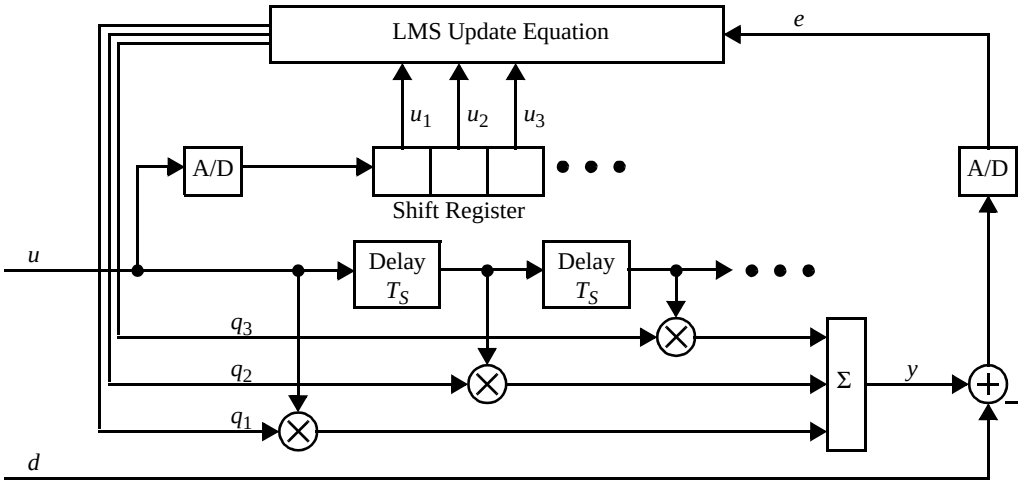


Figure 4.19 Block diagram of an analog adaptive transversal filter with a digital LMS algorithm using just one digitizer to obtain the state vector.

Nyquist rate because T_S-spaced samples are required at $u(k), u(k-1), \ldots u(k-M+1)$. However, each gradient component depends upon only one input sample so all M parameters do not have to be updated simultaneously. If each gradient signal ϕ_i is obtained at a different time, k_i, then the LMS update equation looks like this:

$$\begin{bmatrix} q_1(k+1) \\ q_2(k+1) \\ . \\ q_M(k+1) \end{bmatrix} = \begin{bmatrix} q_1(k) \\ q_2(k) \\ . \\ q_M(k) \end{bmatrix} - \mu \begin{bmatrix} (\partial \hat{q_1} / \partial \varepsilon)(k_1) \\ (\partial \hat{q_2} / \partial \varepsilon)(k_2) \\ . \\ (\partial \hat{q_M} / \partial \varepsilon)(k_M) \end{bmatrix}$$

$$= \begin{bmatrix} q_1(k) \\ q_2(k) \\ . \\ q_M(k) \end{bmatrix} + 2\mu \begin{bmatrix} e(k_1)u(k_1) \\ e(k_2)u(k_2-1) \\ . \\ e(k_M)u(k_M-M+1) \end{bmatrix} \tag{4.37}$$

If the integers k_i are properly chosen both the filter input and error signal can be subsampled while providing the same accuracy and stability as the full-rate LMS algorithm.

For a M-tap filter, a convenient choice of the sampling instants is to subsample the input by $M\times$ and the output error and shift register by $(M+1)\times$. In this case, each parameter is updated every $M(M+1)$ samples. A block diagram of this approach is

shown in Fig. 4.20 for $M = 5$. At first, the signals e and u are sampled at the same time and the product $-2e(k)u(k)$ provides the first transversal filter gradient estimate, $(\partial \hat{q}_1 / \partial \varepsilon)$. The next samples, taken M and $(M+1)$ sampling intervals later, can be used for the second gradient estimate, $(\partial \hat{q}_2 / \partial \varepsilon) = -2e(k+M+1)u(k+M)$. The next samples are separated by two sampling intervals and are used to calculate $(\partial \hat{q}_3 / \partial \varepsilon)$. The algorithm cycles through all of the required gradient estimates until, after $M(M+1)$ sampling intervals, all of the parameters have been updated once and the input and error signal samples are again coincident.

Model matching simulations were performed for the 5-tap example. An additive Gaussian disturbance of variance 10^{-3} was introduced to ensure a finite MSE in steady state. The average MSE over an ensemble of 100 simulation runs is plotted versus time in Fig. 4.21 for both the full-rate and subsampled LMS algorithms. Since iterations are performed only every 30 sampling intervals, the subsampled algorithm converges at 1/30th the original rate. However, in high speed systems (where subsampling is most beneficial) this is often not a problem. Indeed, it is possible to sample the error and input signals even more slowly if desired resulting in even slower convergence.

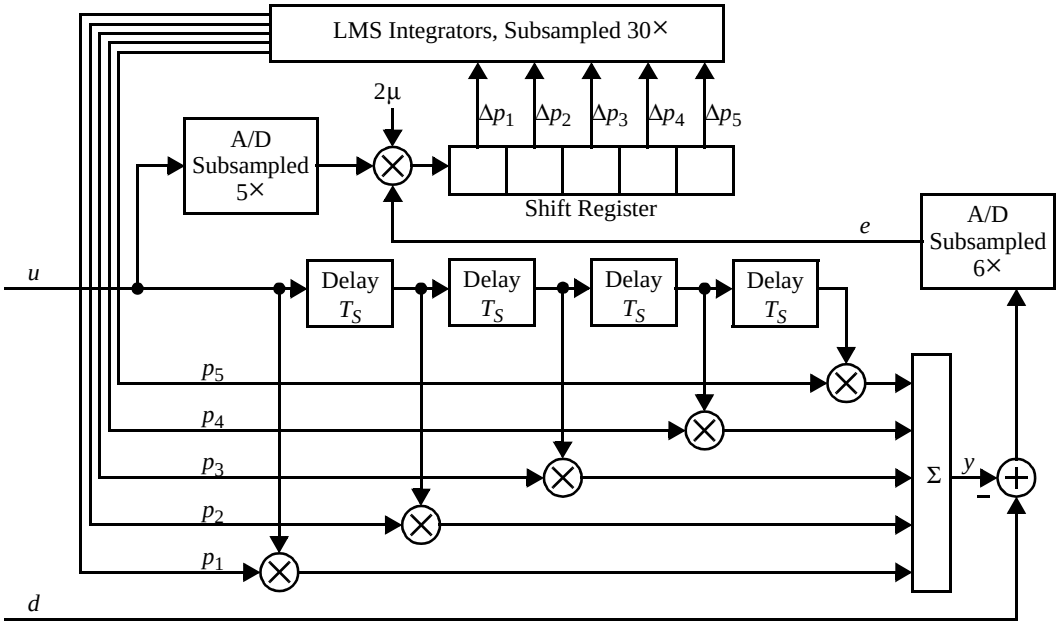


Figure 4.20 Block diagram of a 5-tap analog adaptive transversal filter with the subsampled digital LMS algorithm.

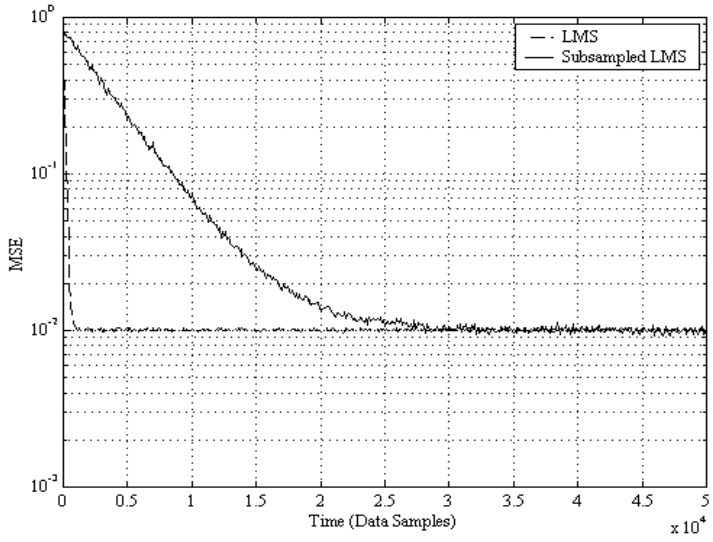


Figure 4.21 MSE convergence of the traditional LMS algorithm and the LMS algorithm with 5x subsampling of the filter input.
The data is averaged over an ensemble of 100 simulation runs.

4.6.2.2 Extension to adaptive linear combiners

The LMS-CT algorithm operates by generating estimates of the ALC's internal state signals using discrete time FIR filters as shown in Fig. 4.3. Unfortunately, in order for the state estimates to be accurate, the input signal must be sampled at or above its Nyquist rate. However, it is not necessary to perform iterations of the LMS-CT update at every sampling instant. Parameter updates can be performed less often resulting in slower convergence but allowing more time to perform the matrix multiplication. In this case, the error signal could be subsampled but the input signal still must be sampled at the Nyquist rate.

The LMS-ICT algorithm, on the other hand, operates by calculating the desired change in the filter parameters as if it were a transversal filter, $\Delta \boldsymbol{q}$. This change is then mapped to a change in the ALC parameter vector, $\Delta \boldsymbol{p} = \boldsymbol{K} \cdot \Delta \boldsymbol{q}$. So, one may use the techniques in Section 4.6.2.1 to generate the transversal filter change $\Delta \boldsymbol{q}$ from subsampled error *and* input signals. For example, in the 5-tap filter discussed above, the change in the transversal parameter vector for each iteration is,

$$\Delta \boldsymbol{q} = 2\mu \begin{bmatrix} e(30l)u(30l) \\ e(30l+6)u(30l+5) \\ e(30l+12)u(30l+10) \\ e(30l+18)u(30l+15) \\ e(30l+24)u(30l+20) \end{bmatrix} \tag{4.38}$$

The LMS-ICT algorithm can be performed with $M = 5$ by simply mapping $\Delta \boldsymbol{q}$ through $\boldsymbol{K}$ and updating the ALC parameters $\boldsymbol{p}$ every 30 sampling intervals,

$$\begin{aligned} \boldsymbol{p}(30(l+1)) &= \boldsymbol{p}(30l) + \Delta \boldsymbol{p} \\ &= \boldsymbol{p}(30l) + \boldsymbol{K} \cdot \Delta \boldsymbol{q} \\ &= \boldsymbol{p}(30l) + 2\mu \boldsymbol{K} \begin{bmatrix} e(30l+1)u(30l) \\ e(30l+7)u(30l+5) \\ e(30l+13)u(30l+10) \\ e(30l+19)u(30l+15) \\ e(30l+25)u(30l+20) \end{bmatrix} \end{aligned} \tag{4.39}$$

The resulting algorithm would converge 30 times slower than the standard LMS-ICT algorithm, but with the same final misadjustment.

As a more realistic example, the same system from Section 4.5.1 was simulated. Since $M = 20$, the input is subsampled by 20× and the error signal is subsampled by 21×. The results for misadjustments of 0.1 and 0.01 are plotted in Fig. 4.22 and Fig. 4.23. Comparing them with the simulation results in Fig. 4.10 and Fig. 4.11, the convergence is 420× slower, as expected, but the misadjustment is the same.

4.7 Experimental Results

To verify the practicality of the LMS-CT and LMS-ICT algorithms in a real integrated system, model matching experiments were performed using the 5th order orthonormal ladder prototype CMOS integrated filter. The filter was configured as shown in Fig. 4.24.

First, the required impulse responses were obtained by differentiating the step responses measured on an oscilloscope. The results are plotted in Fig. 4.25. The waveforms include errors due to noise and nonlinearities introduced by the filter and measure-

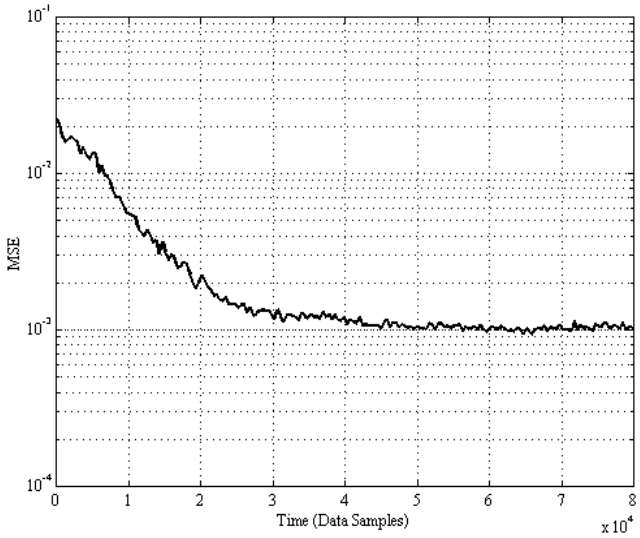


Figure 4.22 Simulation results for the 3rd order orthonormal ladder model matching experiment using the subsampled LMS-ICT algorithm with an excess MSE of 10%.

Each data point is averaged over 420 consecutive data samples and 3 separate simulation runs.

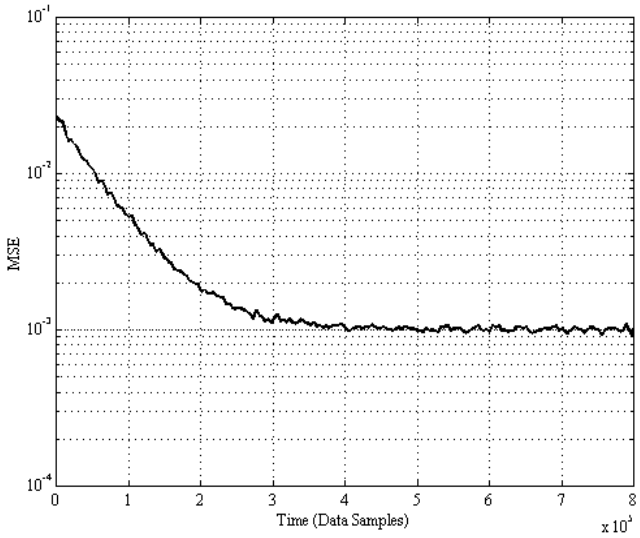


Figure 4.23 Simulation results for the 3rd order orthonormal ladder model matching experiment using the subsampled LMS-ICT algorithm with an excess MSE of 1%.

Each data point is averaged over 420 consecutive data samples and 5 separate simulation runs.

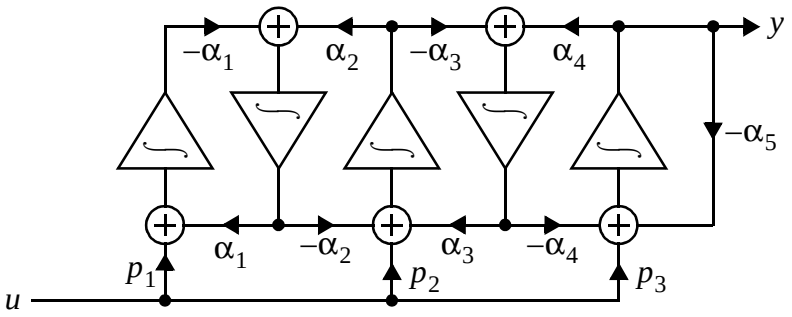


Figure 4.24 5th order orthonormal ladder filter structure with programmable feed-ins.

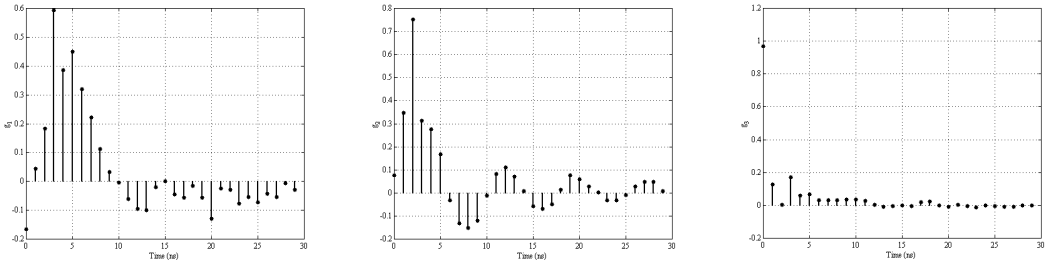


Figure 4.25 Sampled, truncated impulse responses of the fifth order integrated analog filter.

ment equipment. Fortunately, the LMS-CT and LMS-ICT algorithms are robust with respect to these errors.

The experimental setup is diagrammed in Fig. 4.26. The same filter is used for the adapted and reference signal paths to avoid any mismatch. First, the oscilloscope digitizes the filter output with the filter's feed-in values programmed to their optimal values, $\boldsymbol{p}^*$. The digitized waveform is then stored by the PC for use as the desired signal, d. Then, the same input sequence is repeated with the feed-in parameters programmed to the current adapted values, $\boldsymbol{p}(k)$. This time, the digitized waveform is used as the output signal, y. The oscilloscope also digitizes the filter input, u, on a second channel. The error signal $e = d - y$ and the input u are then used to perform one iteration of the adaptive algorithm's parameter update equation in software.

Under these conditions, it would be impossible to use a traditional LMS algorithm since the filter's state signals are completely unavailable. However, using the LMS-CT

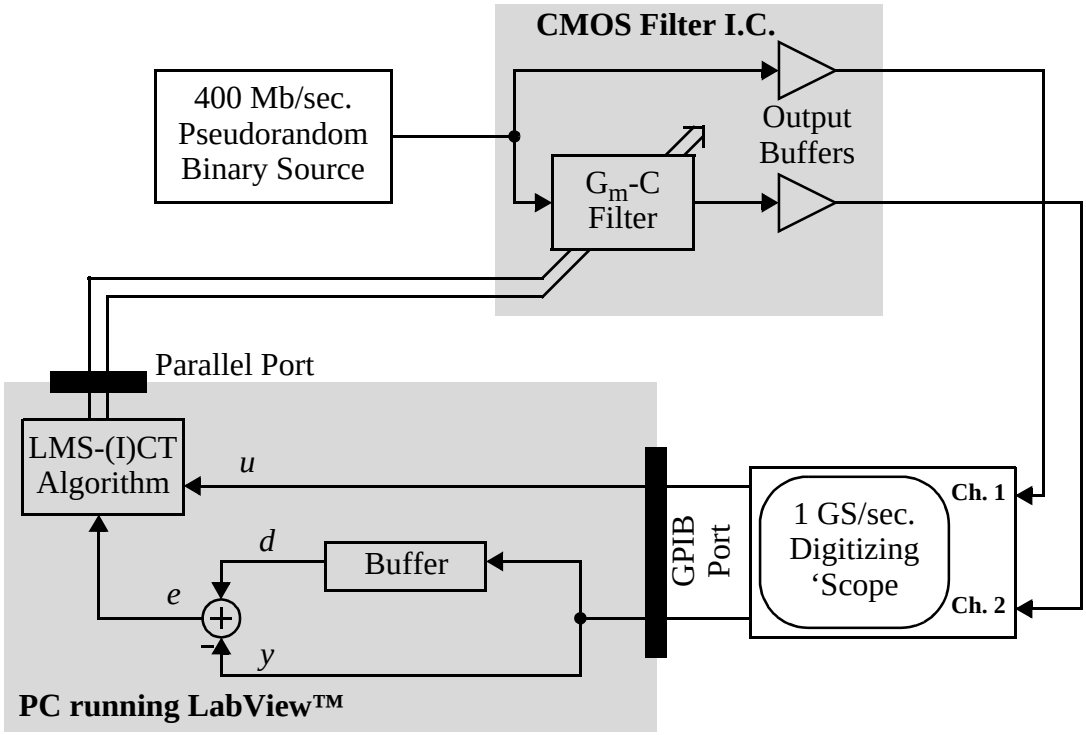


Figure 4.26 Experimental setup for testing the adaptive algorithms on an integrated analog filer.

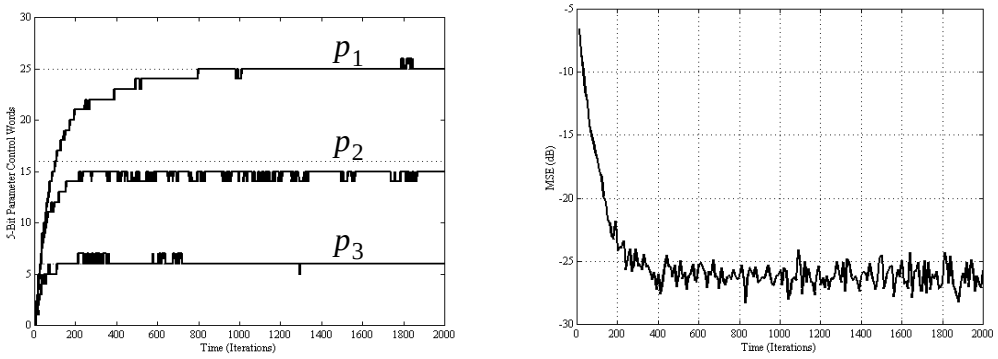


Figure 4.27 Model matching learning curves and MSE relative to the desired output using the LMS-CT algorithm on integrated hardware.

and LMS-ICT algorithms, the model matching experiment succeeds. The 5-bit parameter values and MSE are plotted over time in Fig. 4.27 and Fig. 4.28. Approximately 2000 iterations are required to obtain convergence. A steady state error of 1 LSB persists on

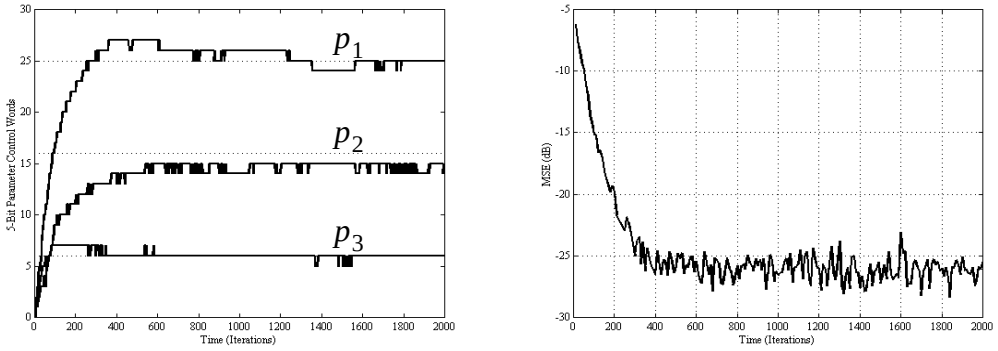


Figure 4.28 Model matching learning curves and MSE relative to the desired output using the LMS-ICT algorithm on integrated hardware.

p_2 and the resulting steady state MSE is 26 dB below the signal level for both algorithms. These steady state errors are comparable in magnitude to the filter's nonlinearities.

4.8 Conclusions

This chapter has described techniques for obtaining digital estimates of the gradient signals required for LMS adaptation without access to a filter's internal state signals. The techniques are particularly useful for digitally adapting high-speed analog filters with several adapted parameters. Using traditional LMS adaptation, a digitizer (A/D converter or comparator) is required for each gradient signal as well as the filter output. Furthermore, in some filter structures, such as those with programmable feed-ins, the state signals are not available anywhere in the analog signal path so additional analog filters must be built to accommodate the LMS algorithm.

Using the LMS-CT or LMS-ICT algorithms, A/D converters are required on the input and error signals only. Digital signal processing is used to obtain estimates of the gradient signals from the digitized input and error samples. Compared to the traditional LMS algorithm, the convergence rate and misadjustment are identical. An additional matrix multiplication is required for each iteration of the algorithm. So, analog circuit complexity is reduced but digital circuit complexity is increased with little or no change in overall performance making it an attractive option for mixed-signal integrated systems in

digital CMOS processes. The signed and subsampled variations of these algorithms allow a system designer to reduce the analog and digital circuit complexity even further, but with a slower convergence rate. This is likely to be a desirable trade-off in applications such as wired communications where the adaptation rate is not a limiting factor. Combining these techniques, adaptation can be performed using just two comparators and relatively simple digital logic, all of which can be subsampled below the Nyquist rate. By comparison, an adaptive linear combiner with variable gains at the output summing node can be adapted using the SS-LMS algorithm requiring one comparator on each state signal and one comparator on the error signal alone, all of which may be subsampled. In this case, the advantages offered by the SS-LMS-ICT algorithm are modest, particularly when only 2 or 3 parameters are being adapted. However, in some filter structures (such as the orthonormal ladder filter) the state signals do not exist anywhere in the filter. Hence, direct sampling of the states for SS-LMS adaptation is impossible, and a co-ordinate transform greatly simplifies gradient signal generation.

In practice, there is likely to be some mismatch between the analog filter and the digital representation of its impulse response in the matrices $\boldsymbol{G}$ and $\boldsymbol{K}$. Fortunately, the LMS-CT and LMS-ICT algorithms are robust in the presence of these mismatches as evidenced by the robustness of the signed algorithms, which use only one bit to represent each entry in the $\boldsymbol{G}$ and $\boldsymbol{K}$ matrices, and the successful model-matching experiments performed on an integrated filter with noisy measured impulse responses (Fig. 4.25).

In order to use the co-ordinate transform techniques, it must be possible to model the adaptive filter as an adaptive linear combiner. Therefore, it is not possible to use the LMS-CT and LMS-ICT algorithms to adapt the poles of a filter. In general, locating the fixed poles in an analog continuous time filter is performed rather heuristically from a knowledge of the input and desired signal statistics. For instance, over wired channels the equalizer poles are usually positioned to provide an approximation to the channel's inverse [12], [13], [14]. In magnetic storage read channels, the equalizer poles are placed to provide a lowpass anti-aliasing magnitude response with linear phase in the passband in order to preserve the signal's timing information [15]. However, as demonstrated in

[16], better pole locations can sometimes be found by using a more general optimization approach. Adaptive techniques have been used to optimize filter poles in the past [17]. One might consider combining the co-ordinate transform techniques of this chapter with other adaptive algorithms capable of adapting the filter poles, such as the DLS algorithm in Chapter 3.

4.9 References

[1] D. A. Johns, W. M .Snelgrove, A. S. Sedra, "Continuous-Time LMS Adaptive Recursive Filters," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 769-777, July 1991.

[2] A. Shoval, D. A. Johns, and W. M. Snelgrove, "Comparison of Dc Offset Effects in Four LMS Adaptive Algorithms," *IEEE Trans. Circuits Syst. II*, vol. 42, pp. 176-185, March 1995.

[3] G. Strang, *Linear Algebra and Its Applications*, Sec. Ed., Academic Press, New York, 1980.

[4] B. Widrow, J. M. McCool, M. G. Larimore, and C. R. Johnson, "Stationary and Nonstationary Learning Characteristics of the LMS Adaptive Filter," *Proc. IEEE*, vol. 64, pp. 1151-1162, Aug. 1976.

[5] B. Widrow and S.D. Stearns, *Adaptive Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall, 1985.

[6] S. Haykin, *Adaptive Filter Theory*, 3rd Ed., Upper Saddle River, NJ: Prentice-Hall, 1996.

[7] D.A. Johns, W.M. Snelgrove and A.S. Sedra, "Orthonormal ladder filters", *IEEE Trans. on Circuits and Systems*, vol. CAS-36, pp. 337-343, March 1989.

[8] D. Hirsch and W. Wolf, "A simple adaptive equalizer for efficient data transmission," *IEEE Trans. Commun.*, vol. COM-18, p. 5, 1970.

[9] C. R. Rohrs, C. R. Johnson and J. D. Mills, "A Stability Problem in Sign-Sign Adaptive Algorithms," *IEEE Int. Conf. Acoust., Speech, Signal Processing*, vol. 4, pp. 2999-3001, April 1986.

[10] S. Kiriaki, T. L. Viswanathan, G. Feygin, B. Staszewski, R. Pierson, B. Krenik, M. de Wit, and K. Nagaraj, "A 160-MHz Analog Equalizer for Magnetic Disk Read Channels," *IEEE J. Solid-State Circuits*, vol. 32, pp. 1839-1850, Nov. 1997.

[11] J. Sonntag, O. Agazzi, P. Aziz, H. Burger, V. Comino, M. Heimann, T. Karanink, J. Khoury, G. Madine, K. Nagaraj, G. Offord, R. Peruzzi, J. Plany, N. Rao, N. Sayiner, P. Setty, and K. Thread-gill, "A High Speed, Low Power PRML Read Channel Device," *IEEE Trans. Magnetic*s, vol. 31, pp. 1186-1195, March 1995.

[12] K. T. Oshiro and G. T. Uehara, "A 10 Gb/s 83 mW GaAs HBT Equalizer/ Detector for Coaxial Cable Channels," *IEEE Custom Integrated Circuits Conf.*, pp.347-350, 1998.

[13] M. H. Shakiba, "A 2.5Gb/s Adaptive Cable Equalizer," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 396-397, Feb. 1999.

[14] P. Amini and O. Shoaei, "A Low-Power Gigabit Ethernet Analog Equalizer," *IEEE Int. Symp. Circuits and Systems*, vol. 1, pp. 176-179, May 2001.

[15] W. Dehaene, M. S. J. Steyaert, and W. Sansen, "A 50 MHz Standard CMOS Pulse Equalizer for Hard Disk Read Channels," *IEEE J. Solid-State Circuits*, vol. 32, pp. 977-988, July 1997.

[16] P. K. D. Pai, A. D. Brewster, and A. Abidi, "A 160-MHz Analog Front-End IC for EPR-IV PRML Magnetic Storage Read Channels," *IEEE J. Solid-State Circuits*, vol. 31, pp. 1803-1816, Nov. 1996.

[17] K. A. Kozma, D. A. Johns, and A. S. Sedra, "Automatic Tuning of Continuous-Time Integrated Filters Using an Adaptive Filter Technique," *IEEE Trans. on Circuits and Systems*, vol. 38, pp. 1241-1248, Nov. 1991.

Chapter 5

Obtaining Gradient Signals by Unknown Input State Observation

5.1 Introduction

The LMS algorithm requires a gradient signal, ϕ_i, for each adapted coefficient. If a state-space model is used for the adapted filter, the gradient signals can be generated from a knowledge of the filter's internal states [1]. This suggests the architecture depicted in Fig. 5.1A. However, in order to digitally adapt an analog filter, digitized state signals are required. If A/D converters are used to sample each of the filter's internal states, the size and complexity of the analog circuitry may become prohibitive. This chapter examines a method for estimating the state signals of an arbitrary state-space adaptive filter using only the digitized filter output, allowing for the modified system architecture depicted in Fig. 5.1B [2]. The technique is based upon estimating the filter's internal state signals from the filter output.

Section 5.2 looks at established techniques for estimating the states of a system. Section 5.3 describes a method for generating approximate time delayed state estimates of non-minimum phase systems with unknown inputs. Section 5.4 provides behavioral simulation results of the LMS algorithm using unknown input state observation. Practical considerations such as dc offsets and mismatch between the analog and digital signal processing are considered. The hardware requirements of this approach are discussed in Section 5.5 and compared with traditional LMS adaptation.

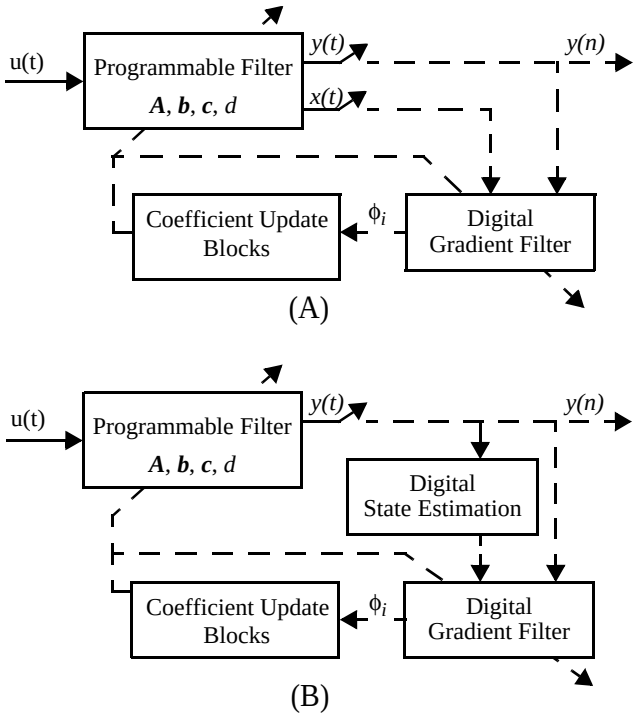


Figure 5.1 Digital adaptation of an analog filter (A) with and (B) without sampling the internal state signals.
The dashed lines correspond to digitized signals.

5.2 Unknown Input Observation

The estimation of a system's internal states, called state observation, is well documented in control theory [3], [4], [5]. A linear state observer can be designed of order equal to that of the system under observation and the dynamics of the state observer can be designed to meet a desired tracking performance specification. Unfortunately, in general, access to both the system inputs and outputs is required to generate these state estimates (Fig. 5.2A).

In Fig. 5.2B, the "Observed System" is the programmable filter (***A***, ***b***, ***c***, *d*) and only its output $y(k)$ is available. Furthermore, the programmable filter coefficients can be adapted based upon gradient information from some time in the past (as long as the overall system's adaptation rate is satisfactory), so time delayed state estimates $\hat{\boldsymbol{x}}(k-D)$ are sufficient. A directly analogous problem appears in the control literature and is called

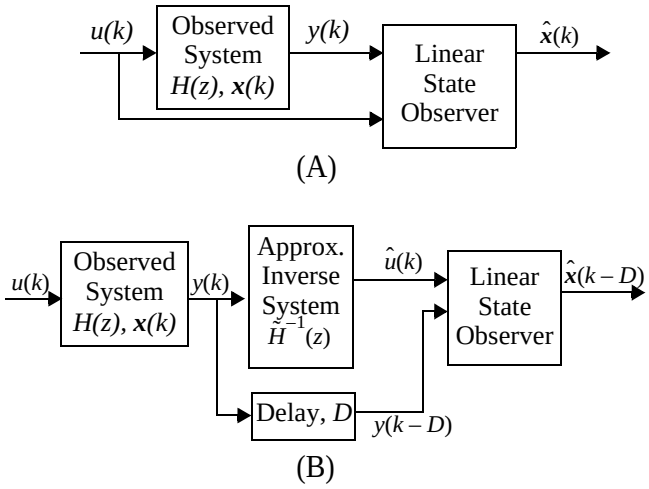


Figure 5.2 State estimation using (A) established techniques and (B) time delayed unknown input observation.

"time delayed and unknown input observation". It was shown in [6] that this problem is equivalent to inversion of the observed system. This is intuitively satisfying because if the observed system can be inverted then its input can be reproduced from its output and the problem of state observation becomes straight forward.

Unfortunately, it is difficult to ensure that an adaptive filter will always be invertible, particularly if the zeros are being adapted. Since the filter may become non-minimum phase, the resulting inverted system might have unstable poles.

5.3 Approximate Time Delayed State Observation

This section describes a method for generating time delayed state estimates of an arbitrary state-space system. This approach approximates the inverse transfer function of the system under observation by introducing delay. Once a delayed estimate of the system input is obtained, $\hat{u}(k-D)$, a standard linear state observer may be used to produce time delayed state estimates, as in Fig. 5.2B.

5.3.1 Background

Consider a discrete time linear time-invariant filter defined by its impulse response,

$$f(k) = \begin{cases} 1, k = 0 \\ -z_i, k = 1 \\ 0, \text{otherwise} \end{cases} \tag{5.1}$$

The z-transform of $f(k)$ is,

$$F(z) = 1 - z_i z^{-1} \tag{5.2}$$

$F(z)$ has one zero at z_i, and a region of convergence (ROC) that includes the entire z-plane.

In order to recreate the input of filter f from its output, a BIBO stable filter[1] f_i is required which satisfies the following condition,

$$f * f_i(k) = \begin{cases} 1, k = 0 \\ 0, k \neq 0 \end{cases} \tag{5.3}$$

This can be accomplished using the inverse system,

$$F_i(z) = \frac{1}{1 - z_i z^{-1}} \tag{5.4}$$

$F_i(z)$ has one pole at z_i and two possible regions of convergence:

- $|z| > |z_i|$ resulting in a causal impulse response, $f_i^{C}(k) = 0$ for $k < 0$
- $|z| < |z_i|$ resulting in a non-causal impulse response, $f_i^{NC}(k) = 0$ for $k > 0$

The convolution theorem requires that the regions of convergence of $F(z)$ and $F_i(z)$ overlap in order to satisfy Eqn. (5.3) [7]. Fortunately, the ROC of $F(z)$ is the entire z-plane, so they will always overlap. Therefore, Eqn. (5.3) is satisfied by both the causal and non-causal inverses, although only one will be stable.

For stability, it is necessary that the ROC includes the unit circle. There are three cases of interest:

1. $|z_i| < 1$, in which case the causal inverse is stable and the non-causal inverse is unstable. Therefore, the required stable filter is $f_i = f_i^{C}$.

1. BIBO stability implies that for every bounded input, the system produces a bounded output

2. $|z_i| > 1$, in which case the non-causal inverse is stable and the causal inverse is unstable. Therefore, the required stable filter is $f_i = f_i^{\mathrm{NC}}$.
3. $|z_i| = 1$, in which case there is no stable inverse of f since $F(z)$ has a transmission zero. Hence, its frequency response has a spectral null making it impossible to recreate input signals at that frequency.

5.3.2 Derivation of the Approximate Inverse Filter

Assume the adapted filter has a stable, real, rational, discrete time transfer function $H(z)$ with zeros z_i and poles p_l.

$$H(z) = A \cdot \frac{\prod_i (1 - z_i z^{-1})}{\prod_l (1 - p_l z^{-1})} \tag{5.5}$$

This could be the transfer function of any real stable discrete state space system, or the equivalent transfer function of a continuous time state space system sampled at or above the Nyquist rate. Again, a stable filter is sought to recreate the filter input $u(k)$ from its output $y(k)$. The inverse transfer function is,

$$H^{-1}(z) = \frac{1}{A} \cdot \frac{\prod_l (1 - p_l z^{-1})}{\prod_i (1 - z_i z^{-1})} = \frac{1}{A}\left(\prod_l (1 - p_l z^{-1})\right)\left(\prod_i H_i(z)\right) \tag{5.6}$$

As long as none of the zeros of $H(z)$, z_i, lie on the unit circle, it is possible to make $H^{-1}(z)$ stable by choosing the ROC of each term $H_i(z) \equiv 1/(1 - z_i z^{-1})$ to include the unit circle, as described for $F_i(z)$ in Section 5.3.1. (Equivalently, one may think of choosing the ROC of $H^{-1}(z)$ to be a ring around the origin which includes the unit circle.) Since both $H(z)$ and $H^{-1}(z)$ are stable, both ROCs overlap on the unit circle guaranteeing that $H^{-1}(z)$ perfectly reconstructs $u(k)$ from $y(k)$.

In general, $H^{-1}(z)$ will be non-causal making it difficult to implement in real-time. Instead, a time-delayed approximate inverse $\tilde{H}^{-1}(z)$ is used,

$$\tilde{H}^{-1}(z) = \frac{1}{A}\left(\prod_l (1 - p_l z^{-1})\right)\left(\prod_i \tilde{H}_i(z)\right) \tag{5.7}$$

Eqn. (5.7) differs from the definition of $H^{-1}(z)$ in Eqn. (5.6) only in that the first-order terms $H_i(z)$ have been replaced by $\tilde{H}_i(z)$. There are 3 cases to consider in defining $\tilde{H}_i(z)$,

1. $|z_i| < 1$: The ROC is taken to be $|z| > |z_i|$ resulting in a causal stable impulse response. In this case, $\tilde{H}_i(z) = H_i(z)$ can be implemented in direct form.
2. $|z_i| > 1$: The ROC is taken as $|z| < |z_i|$. The result is a non-causal system which can not be implemented in real time.

$$\begin{aligned} H_i(z) &= 1/(1 - z_i z^{-1}) \\ &= (-z/z_i)/(1 - z/z_i) \\ &= (-z/z_i)(1 + z_i^{-1}z + z_i^{-2}z^2 + \ldots) \end{aligned} \tag{5.8}$$

The Taylor series in Eqn. (5.8) is guaranteed convergent on the unit circle since $|z/z_i| < 1$. In order to implement $H_i(z)$ in real time, a delay of d time steps is introduced and the remaining (vanishingly small) non-causal terms are dropped,

$$\begin{aligned} z^{-d}H_i(z) &= (-1/z_i)(z^{-d+1} + z_i^{-1}z^{-d+2} + \ldots + z_i^{-d+2}z^{-1} + z_i^{-d+1} + z_i^{-d}z + \ldots) \\ &\cong (-1/z_i)(z^{-d+1} + z_i^{-1}z^{-d+2} + \ldots + z_i^{-d+2}z^{-1} + z_i^{-d+1}) \\ &= \tilde{H}_i(z) \end{aligned} \tag{5.9}$$

3. $|z_i| = 1$: This case is discussed in detail in Section 5.3.4.

The total delay, D, introduced by the approximation $\tilde{H}^{-1}(z) \approx z^{-D}H^{-1}(z)$ is d times the number of factors $H_i(z)$ for which $|z_i| > 1$.

5.3.3 Approximation Error

An approximation error is incurred by truncating the Taylor series in Eqn. (5.9) to exclude the non-causal terms. Specifically, in Fig. 5.2B, the truncation causes $\hat{u}(k) \neq u(k - D)$. Let,

$$e_u(k) = \hat{u}(k) - u(k - D) \tag{5.10}$$

The error $e_u(k)$ can be considered an additive noise on the input estimate $\hat{u}(k)$, which in turn becomes noise on the gradient estimate used for LMS adaptation. Fortunately, the

LMS algorithm is robust with respect to noise on the gradient estimates. However, at some point gradient noise will limit the accuracy of the adaptation.

As shown in Fig. 5.3, $e_u(k)$ can be considered the difference between the output of $\tilde{H}^{-1}(z)$ and the output of $z^{-D}H^{-1}(z)$ when subjected to the same input, $y(k)$. Assuming the filter output samples $y(k)$ are independent with zero mean and variance σ_y^2, the variance of the estimation error is given by,

$$E[e_u^2(k)] = \sigma_y^2 \cdot \sum_{i,\,|z_i|>1} \left(\sum_{m=d}^{\infty} |z_i|^{-2m} \right) \tag{5.11}$$

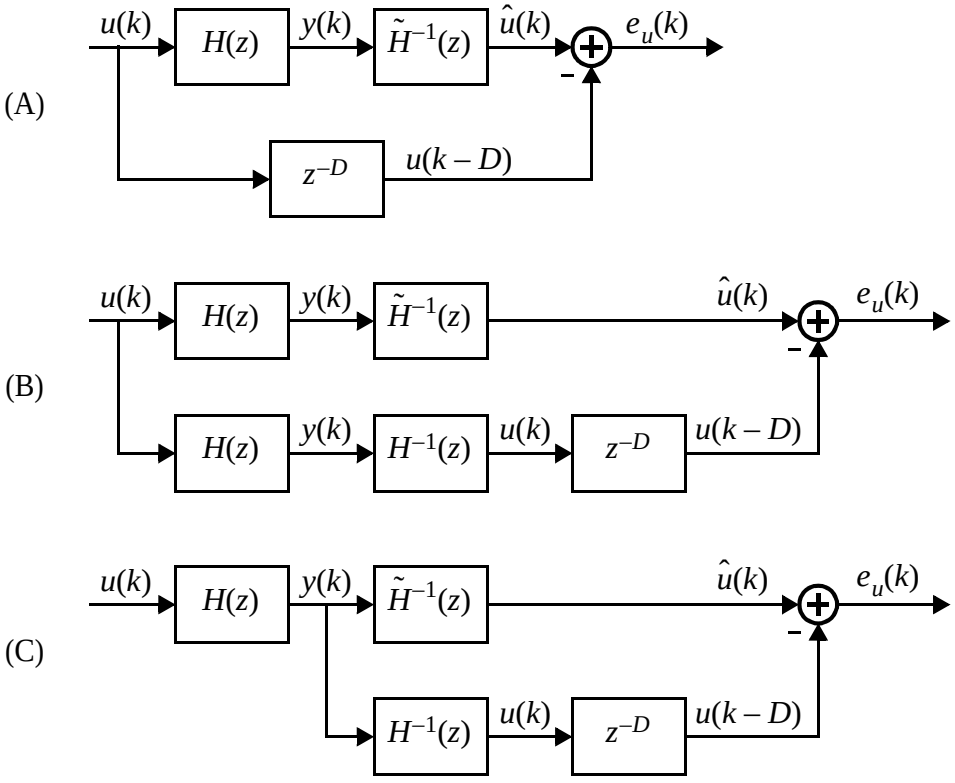


Figure 5.3 System models for determining the input estimation error.

(A) *basic model*

(B) *introducing* $H(z) \cdot H^{-1}(z)$ *has no effect on the bottom signal path since* $H^{-1}(z)$ *provides perfect reconstruction of* $u(k)$

(C) *the two* $H(z)$ *systems can be combined*

Recall that since $|z_i| > 1$, the bracketed power series in Eqn. (5.11) is convergent. By increasing d, $E[e_u^2(k)]$ can be made arbitrarily small. However, the complexity of calculating the coefficients and operating the filter $\tilde{H}^{-1}(z)$ increases.

5.3.4 Transmission Zeros in the Adapted Filter

A problem arises when the zeros of $H(z)$ lie on (or close to) the unit circle. As mentioned above, the resulting transmission zeros will cause part of the input spectrum to be completely attenuated at the output $y(k)$. However, if the input is broadband (as in most digital communications systems) enough of the input spectrum can be recovered from $y(k)$ to perform adaptation. This scenario must be carefully simulated since more taps may be required for $\tilde{H}^{-1}(z)$ to ensure robust convergence.

When a zero of $H(z)$ is near the unit circle (within 0.1), $\tilde{H}^{-1}(z)$ develops a large gain (>20 dB) at frequencies near the zero. To avoid instability, the inverse is calculated by moving the offending zeros of $H(z)$ away from the unit circle (Fig. 5.4). The zeros of the actual adapted filter are not changed; however, the calculation of $\tilde{H}^{-1}(z)$ proceeds as if they were moved. This purely heuristic trick has the effect of reducing the gain of $\tilde{H}^{-1}(z)$ near those frequencies and stabilizing the transfer function. The extra approximation error which results did not cause problems during any of the behavioral simulations performed.

5.4 Simulation Results

Behavioral model matching simulations were used to study the performance of the adaptation algorithm described above. A block diagram of the simulated system is presented in Fig. 5.5. The "Controller" in Fig. 5.5 calculates the approximate inverse transfer function, $\tilde{H}^{-1}(z)$ from a knowledge of $H(z)$ using (if necessary) a truncated Taylor series expansion as described above. Rather than using a linear state observer to estimate the state vector $\boldsymbol{x}$, it is possible to run a digital model of $H(z)$ in parallel driven by the input estimates, $\hat{u}$. Since the adapted filter $H(z)$ is guaranteed stable, the state estimates so obtained will asymptotically track $\boldsymbol{x}$ regardless of the initial conditions on $H(z)$ and

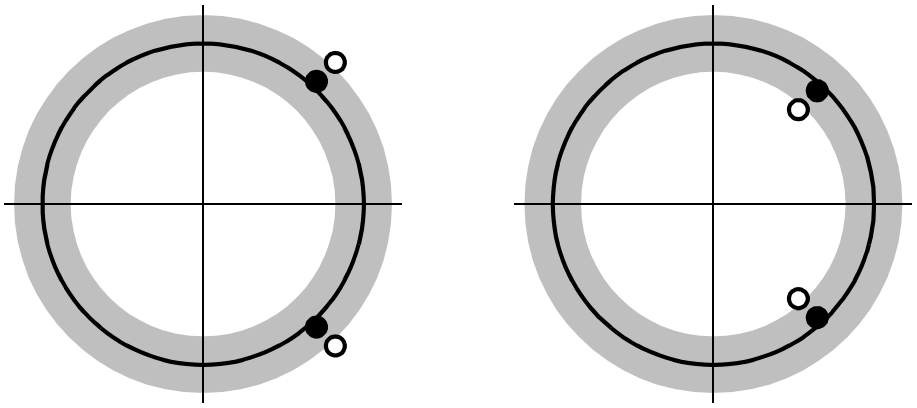


Figure 5.4 Approximation of zeros on the unit circle.

(A) *When zeros of* $H(z)$ *(filled circles, ●) are just outside the unit circle but within the shaded region,* $\tilde{H}^{-1}(z)$ *is calculated by moving the zeros just outside the shaded region (hollow circles, ○).* (B) *If the zeros are just inside the unit circle, they are moved to the interior of the shaded region. In all simulations, the shaded region extends 0.1 to either side of the unit circle.*

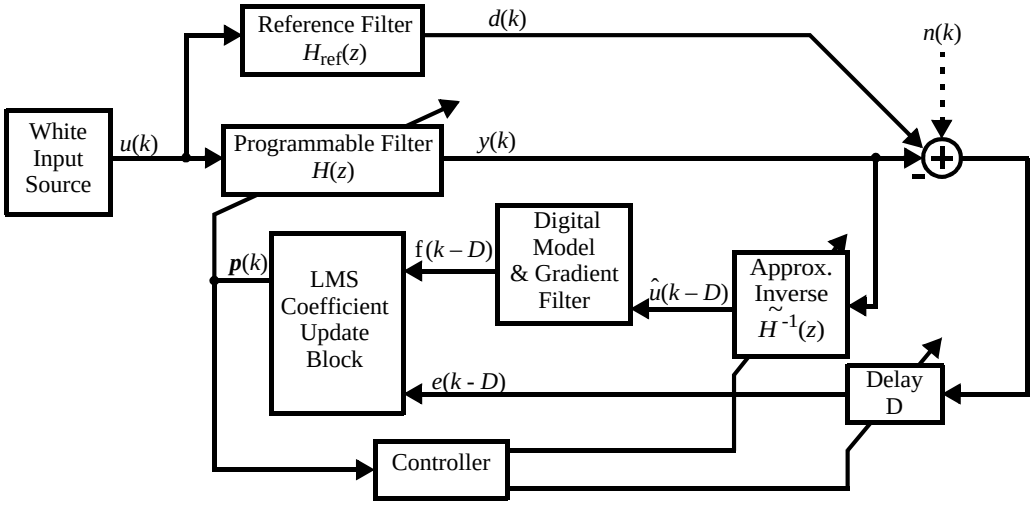


Figure 5.5 Model matching behavioral simulation of LMS adaptation using unknown input state estimation.

the model. A finite steady state error may be introduced to the model matching simulations via $n(k)$.

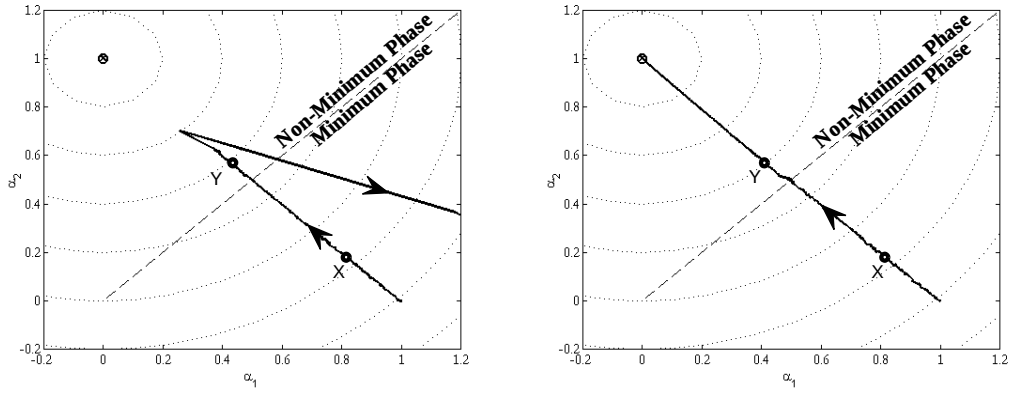
The delay of D time steps does not destabilize the feedback loop formed by the LMS algorithm so long as the delay is small relative to the algorithm's adaptation rate. This is easily ensured in practical applications by appropriately selecting μ. For instance, if $D \leq 10$, μ should be chosen so that the adaptation takes several hundred baud intervals to converge.[1] If fast adaptation is a requirement, this may place a limit on the maximum delay D which can be introduced by $\tilde{H}^{-1}(z)$

5.4.1 2-Tap Transversal Filter

First, a simple 2-tap adaptive transversal filter was simulated. The FIR filter transfer function was,

$$H(z) = \alpha_1 + \alpha_2 z^{-1} \tag{5.12}$$

The two adapted parameters were α_1 and α_2. No steady state error was introduced via n. Fig. 5.6A shows the trajectory of the filter using $\tilde{H}^{-1}(z) = 1/H(z)$. As you can see, this approach will work as long as the signal-path filter $H(z)$ is minimum phase. However, as soon as the adapted zero moves outside of the unit circle, $1/H(z)$ becomes unstable and the algorithm diverges. Using a FIR approximation for $\tilde{H}^{-1}(z)$ with



(A) (B)

Figure 5.6 Trajectory of filter coefficients superimposed on MSE contours. (A) *With* $\tilde{H}^{-1}(z) = 1/H(z)$, *the algorithm diverges when* $H(z)$ *becomes non-minimum phase.* (B) *Using an approximate FIR inverse of length* $M = 10$ *for* $\tilde{H}^{-1}(z)$, *the adaptive process converges.*

1. Expressions relating μ to the adaptation rate of the LMS algorithm are provided in Table 3.1.

$M = 10$ taps as described in Section 5.3, the state trajectory is plotted in Fig. 5.6B with no steady state error.

In Fig. 5.6 at point X, $H(z)$ is a minimum phase transfer function:

$$H_X(z) = 0.8 + 0.2z^{-1} \tag{5.13}$$

Since $H_X(z)$ has a stable inverse, there is no need to make any approximations.

$$\tilde{H}_X^{-1}(z) = \frac{1}{H_X(z)} = \frac{1}{0.8 + 0.2z^{-1}} \tag{5.14}$$

The series connection of $H_X(z)$ and $\tilde{H}_X^{-1}(z)$ will perform perfect reconstruction of the filter input at the output with no delay, $\hat{u} \equiv u$. It may be desirable to truncate the impulse response of $\tilde{H}_X^{-1}(z)$ in order to implement it in hardware with an FIR filter. In simulations, the impulse response was truncated after $M = 10$ samples (Fig. 5.7).

$$\tilde{H}_X^{-1}(z) \cong 1.2500 - 0.3125z^{-1} + 0.0781z^{-2} - \ldots - (4.77 \cdot 10^{-6})z^{-9} \tag{5.15}$$

The truncated impulse response no longer provides perfect reconstruction of the filter input at its output. Since $\hat{u} \neq u$, the gradient signals are noisy. Of course, the LMS algorithm always operates on noisy gradient estimates, so it is no surprise that the adaptation is still stable.

At the point Y on the trajectories in Fig. 5.6, $H(z)$ is non-minimum phase.

$$H_Y(z) = 0.4 + 0.6z^{-1} \tag{5.16}$$

The causal inverse is unstable, as evidenced by its ever-increasing impulse response (Fig. 5.8); this inverse filter was used in the adaptation experiment of Fig. 5.6A causing it to diverge. Instead, in Fig. 5.6B a time delayed and truncated non-causal inverse is used. The approximate inverse (truncated after 10 taps) is,

$$\tilde{H}_Y^{-1}(z) = -0.0434 + 0.0650z^{-1} - \ldots - 1.1111z^{-8} + 1.6667z^{-9} \tag{5.17}$$

Note that in the frequency domain, $\tilde{H}_Y^{-1}$ has a magnitude response approximately equal to $1 / |H_Y|$, but because the non-causal inverse is delayed, the phase response has a steep negative slope. As a result, the series connection of $H_Y(z)$ and $\tilde{H}_Y^{-1}$ has a linear phase response with a group delay of 10 sampling intervals, which also appears in the impulse

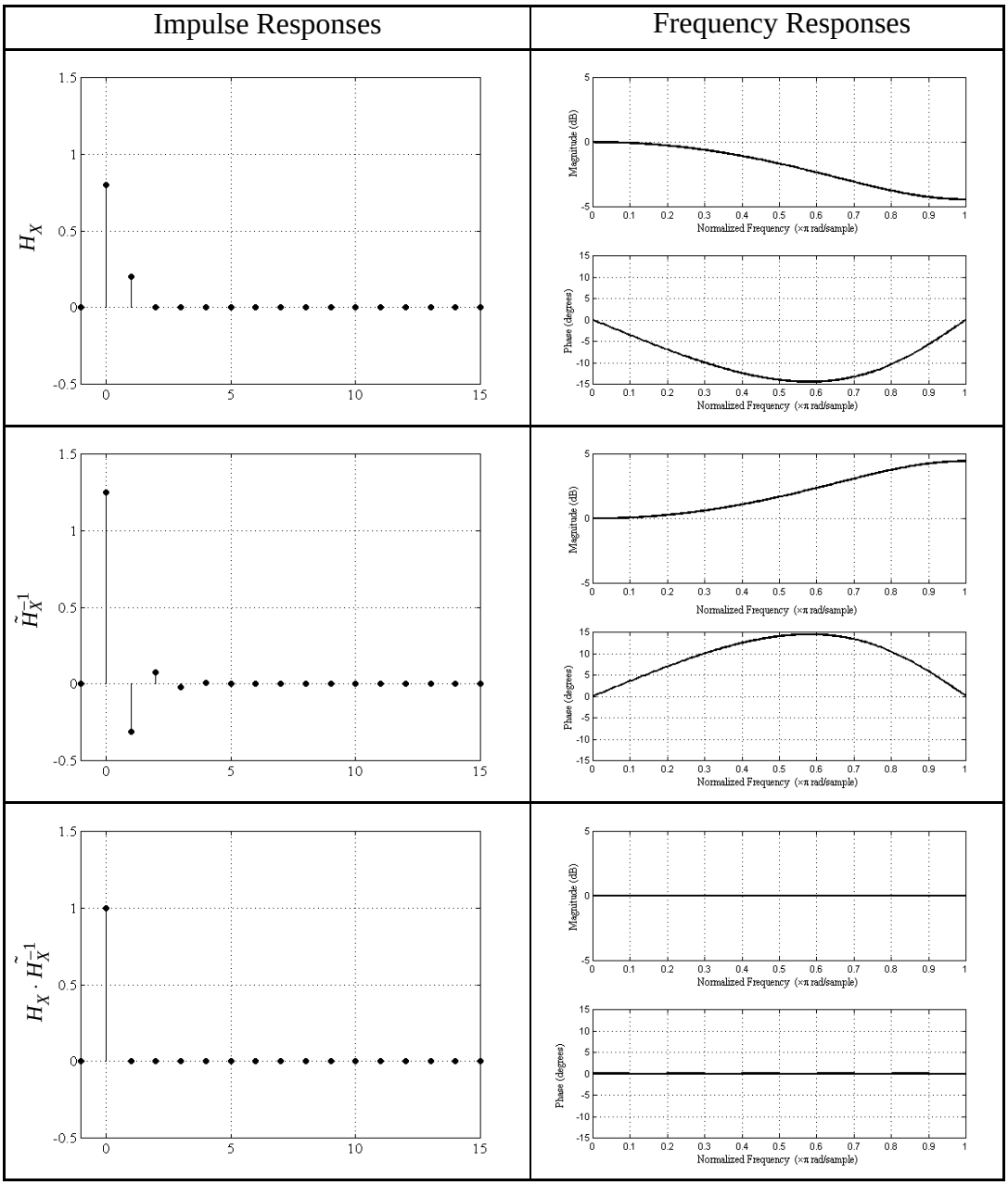


Figure 5.7 Impulse and frequency responses of $H_X(z)$ and its approximate inverse.

response. Again, the truncation introduces noise onto the gradient estimates, but the adaptation is robust.

Next, the simulation was repeated with some finite steady state error introduced via *n* to emulate a real system. A reference filter with a zero on the unit circle is considered,

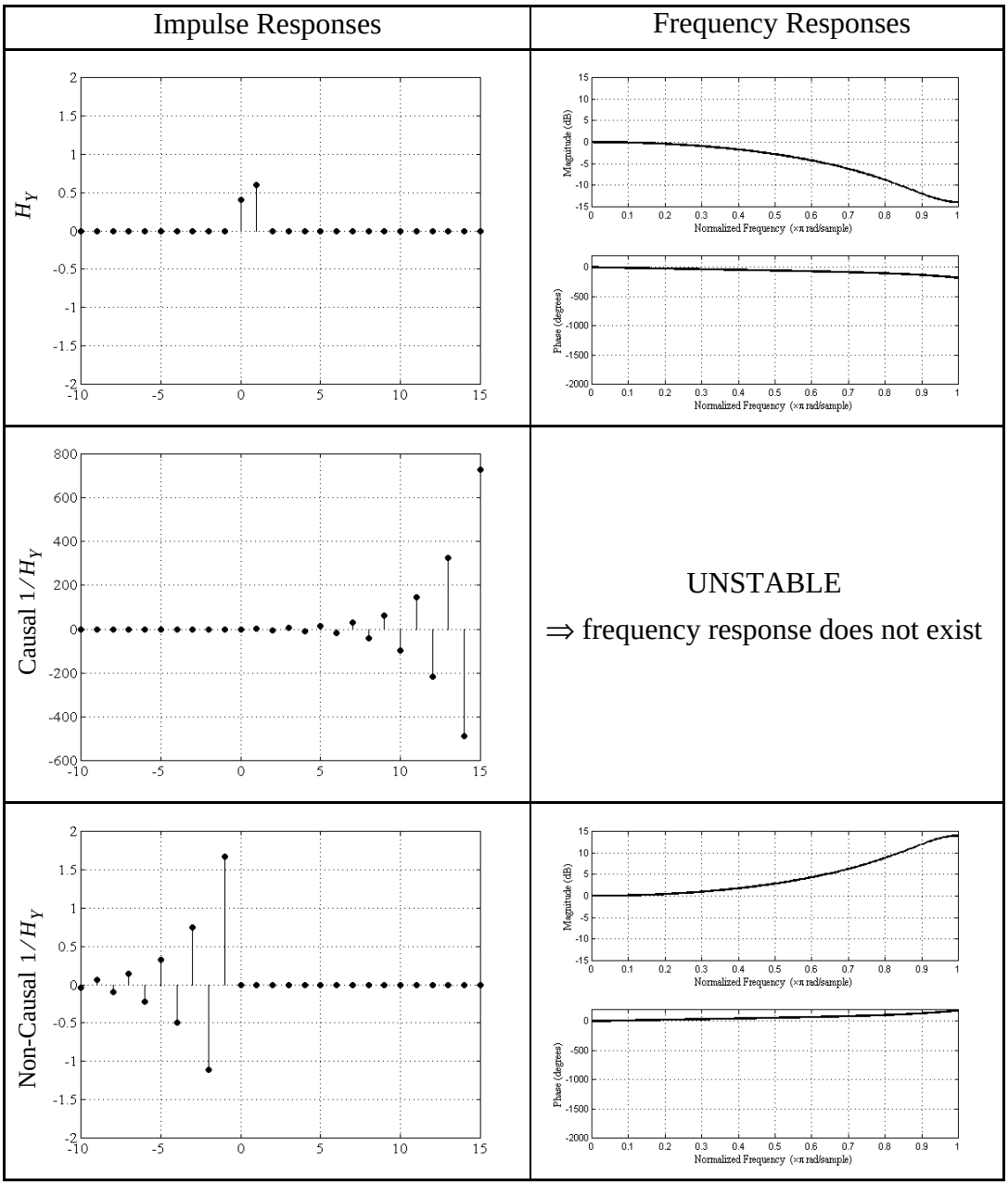


Figure 5.8 Impulse and frequency responses of $H_Y(z)$ and its approximate inverses.

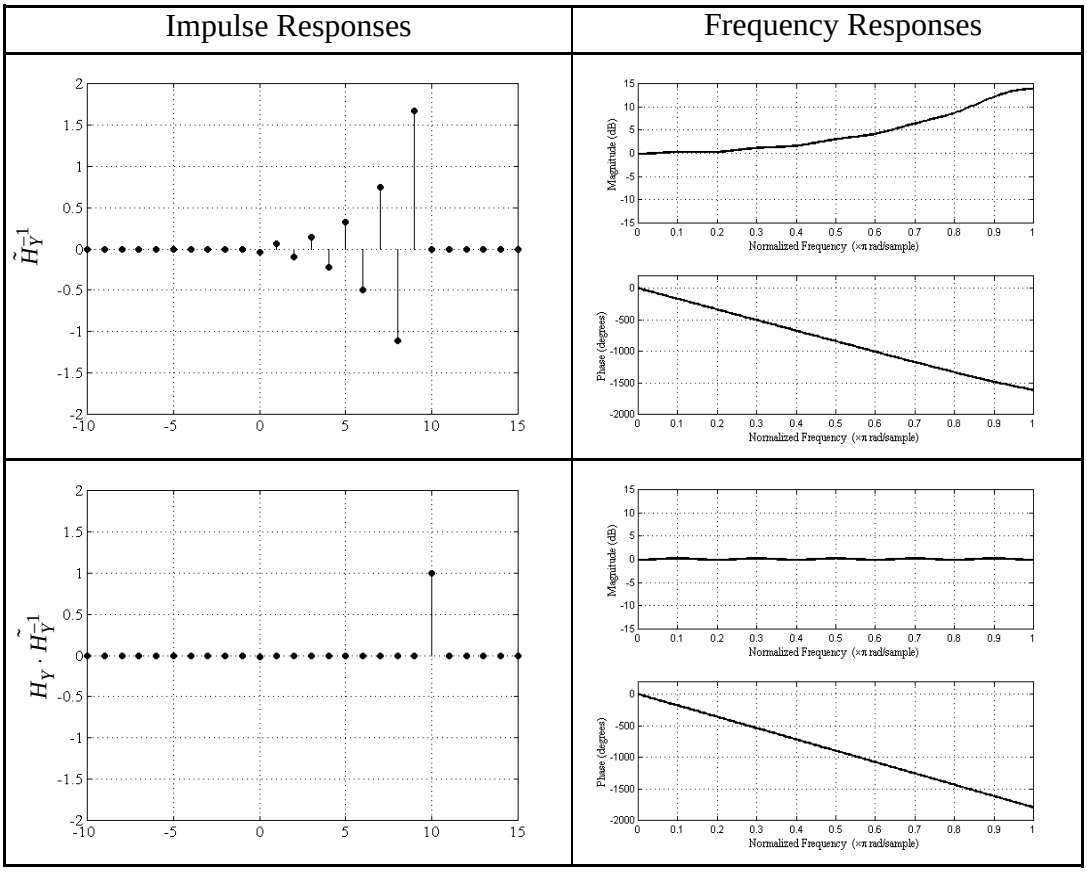


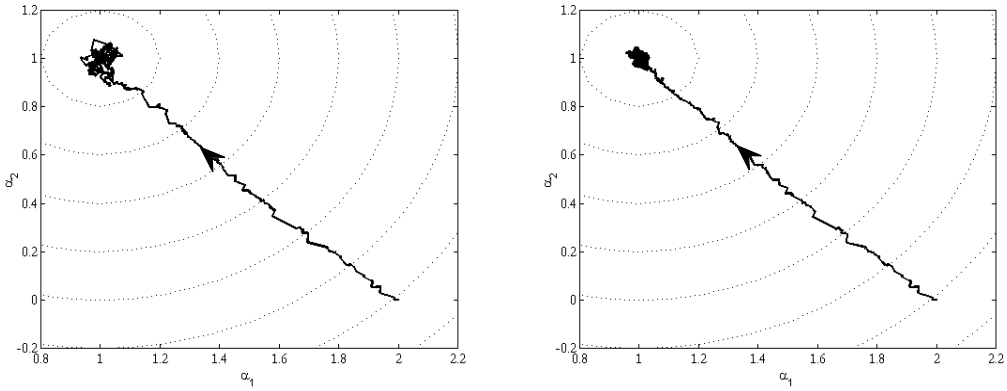
Figure 5.8 Impulse and frequency responses of $\mathbf{H_Y(z)}$ and its approximate inverses.

$$H_{\text{ref}}(z) = 1 + z^{-1} \tag{5.18}$$

As described above, state estimation is difficult under these conditions since part of the input spectrum is completely attenuated at the output. Therefore, the technique described in Section 5.3.4 must be used. The results using only $M = 2$ taps for $\tilde{H}^{-1}(z)$ are plotted in Fig. 5.9A. In steady state, the filter parameters "bounce" around their optimal values because 2 taps are insufficient to produce an accurate stable approximation of $1/(1 + z^{-1})$. However, using $M = 10$ taps (Fig. 5.9B), the performance is the same as a traditional LMS algorithm (not shown here).

5.4.2 Effect of Mismatches

In practice, a digital filter $\tilde{H}^{-1}(z)$ is used to approximate the inverse of an analog filter, $H(z)$. Therefore, this technique presupposes an accurate knowledge of the gains and



(A) (B)

Figure 5.9 Trajectory of filter coefficients superimposed on MSE contours.
(A) *Uses state estimation with* $M = 2$. (B) *Uses state estimation with* $M = 10$.

time constants in the analog signal path. These may be measured during testing and stored in a digital memory, or measured on-the-fly using some startup or calibration procedure. Even so, some mismatch is likely to persist between the signal path transfer function $H(z)$ and the transfer function used to calculate $\tilde{H}^{-1}(z)$, which will be referred to as $H'(z)$. The effect of such mismatches are considered in this section.

If there is only a gain mismatch between $H(z)$ and $H'(z)$, the effect is only a gain offset in the gradient estimates used for LMS adaptation. This will change the rate of adaptation and steady state misadjustment slightly, but will not cause instability as long as μ is chosen sufficiently small. Fig. 5.10B shows simulation results for the 2-tap transversal filter in the presence of a 30% gain mismatch between the analog and digital filters. Convergence is obtained with a 28% reduction in steady state misadjustment due to the slower rate of convergence.

Mismatches between the time constants of $H(z)$ and $H'(z)$ introduce noise onto the gradient estimates. Like the noise caused by truncating the approximate inverse filter, this noise is due to the fact that $\tilde{H}^{-1}(z)$ is not a perfect inverse of $H(z)$ and, hence, $\hat{u} \neq u$. Fortunately (again), since the LMS algorithm is robust with respect to noisy gradient estimates, the effect is usually not catastrophic. In Fig. 5.10C, the 2-tap adaptive transversal filter was again simulated. This time, a mismatch in the time constant was

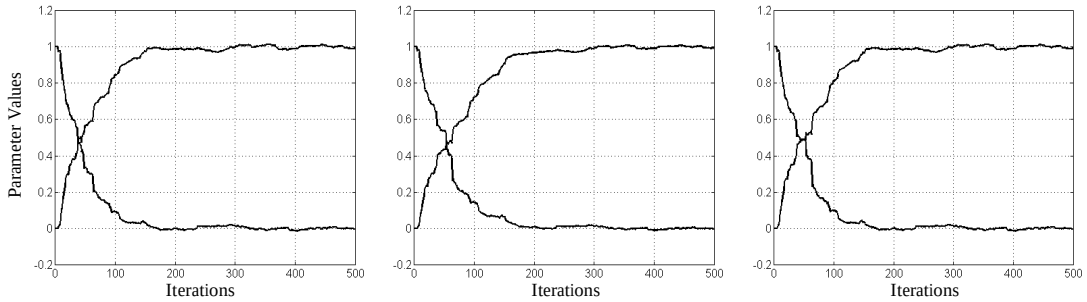


Figure 5.10 Two-tap adaptive transversal filter parameter evolution in the presence of mismatches.

(A) *No mismatch* (B) *30% gain mismatch* (C) *10% mismatch in the time constant*

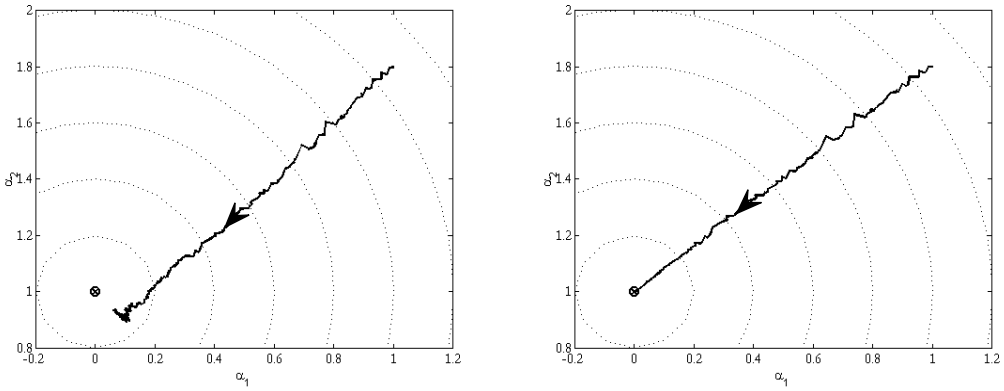
emulated by assuming a 10% mismatch in parameter α_1 only (no mismatch in α_2). This translates into a 10% discrepancy between the zero frequency of $H(z)$ and $H'(z)$. The discrepancy caused no observable difference in convergence rate or misadjustment.

5.4.3 Dc Offset Effects

In Fig. 5.11A the trajectory of a traditional LMS algorithm is plotted for a 2-tap adaptive transversal filter with dc offsets introduced on the state signals, the error signal, and at the output of the LMS multipliers, as would be expected in an analog circuit implementation. The dc offsets have magnitudes equal to 0.1% of the RMS input signal power. The algorithm does not converge to the optimal coefficients in steady state. However, using the state estimation techniques described here in combination with the adaptive dc cancellation tap described in the Section 1.3.2, dc offsets have no effect. Fig. 5.11B shows the simulation results using the same dc offsets as before, but this time using state estimation and an adaptive dc tap. There is no steady state error. As mentioned in Chapter 4, the same advantages are obtainable using any digital gradient descent algorithm in combination with an adaptive dc tap.

5.4.4 5th Order Continuous Time Filter

As a more practical example, the adaptation of a 5th order orthonormal ladder continuous time filter, $H(s)$, was considered. If the continuous time output is sampled at the Nyquist rate, the combination of the filter and subsequent sampling block can be mod-



(A) (B)

Figure 5.11 Trajectory of filter coefficients superimposed on MSE contours. Dc offsets are introduced on all state and error signals.

(A) *Uses the standard LMS algorithm.* (B) *Uses state estimation with* $M = 10$.

eled by a discrete time transfer function $H(z)$ obtained from $H(s)$ via an impulse invariance transformation [7]. State estimation and filter adaptation can then proceed just as before. Fortunately, in most digital communications applications the output of a continuous time filter is sampled at the Nyquist rate anyway in order to perform clock recovery and demodulation. Therefore, no additional analog hardware is required for adaptation.

The reference filter for the continuous time simulations was a 5th order elliptic filter, so $H_{\text{ref}}(s)$ had zeros on the $j\omega$-axis. As a result, in steady state, portions of the input spectrum were completely attenuated at the filter output. However, as long as enough taps are used in the approximate inverse, the state estimates are still accurate enough to allow for convergence using the LMS algorithm (Fig. 5.12). Notice that as M increases, the performance approaches that of the standard LMS algorithm. Using simulations like this one, a value for M can be selected based on the amount of excess MSE which can be tolerated.

5.5 Hardware Requirements

Implementation of an adaptation algorithm using unknown input state observation requires more complex signal processing than straight LMS adaptation. However, in

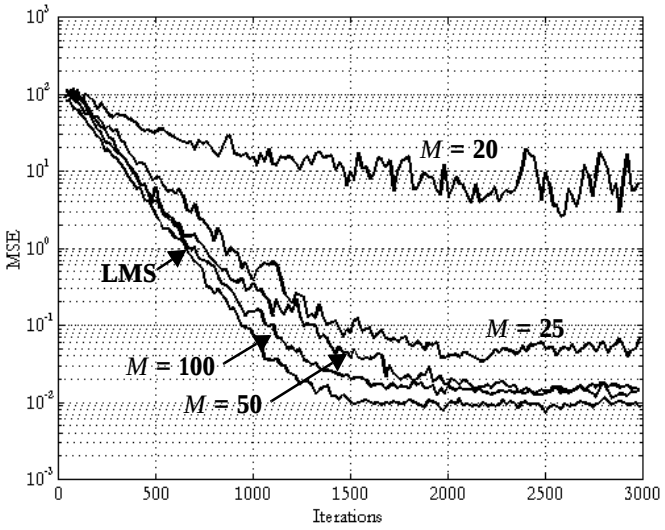


Figure 5.12 LMS adaptation of a 5th order orthonormal ladder filter using unknown input state observation.

The length of the approximate inverse filter, $\tilde{H}^{-1}(z)$, is varied from $M = 20$ to 100. Simulation results for the standard LMS algorithm are also plotted for comparison.

many digital communications applications a DSP core is integrated along with the analog front-end and is mostly idle during startup while adaptation is being performed. The DSP core could be used to perform the state estimation. Specifically, the following digital signal processing functions are required:

- A digital filter operating at the Nyquist rate on the adapted filter's output, $y(k)$, is required to implement the approximate inverse $\tilde{H}^{-1}(z)$. An FIR filter was used in all of the behavioral simulations above. The number of taps depends on the order of the adapted filter, and the amount of noise in the gradient estimate which can be tolerated. For the 2nd order transversal filter, 10 taps were used. For the 5th order continuous time filter, up to 100 taps were used. It might be possible to reduce the number of computations required by using a recursive filter structure for $\tilde{H}^{-1}(z)$. Furthermore, assuming slower convergence can be tolerated, the filter's speed requirement can also be relaxed by using a polyphase implementation. This would entail storing a burst of samples of the filter output in a memory at the Nyquist rate. The samples would then be processed at a slower clock rate to obtain the input & gra-

dient estimates, after which one iteration of the LMS update equation could be performed and the entire process repeated. This is very likely to be the strategy used for any practical integrated implementation.

- A variable delay line is required for the error signal, $e(k)$. The maximum delay is equal to the length of the FIR implementation of $\tilde{H}^{-1}(z)$.
- The "controller" in Fig. 5.5 must calculate the approximate inverse transfer function $\tilde{H}^{-1}(z)$ from a knowledge of the adapted filter parameters, $\boldsymbol{p}(k)$. If this calculation is repeated at the Nyquist rate using the step-by-step procedure described in Section 5.3, it would involve considerable overhead: one Taylor series expansion for each adapted pole or zero involving several digital multiplications and divisions each. However, $\tilde{H}^{-1}(z)$ only has to be recalculated when $H(z)$ changes. It can be recalculated less frequently if a block LMS update rule is used whereby the gradient estimates are averaged for a while before the filter parameters $\boldsymbol{p}(k)$ are updated. Further simplifications might be possible depending upon the exact filter structure. For instance, a simple and fast realization of the controller would be to use the parameters $\boldsymbol{p}(k)$ as the address to a lookup table where the approximate inverse transfer functions are stored for all possible combinations of parameter values. As the number of parameter bits increases, the lookup table will grow large. Depending upon the particular application, the best implementation might be a combination of the block LMS algorithm with lookup tables followed by more accurate calculations.

Of course, unknown input observation implies a savings in analog hardware complexity since sampling of the filter input and state signals is not required. The trade-off of analog hardware for digital hardware is desirable in many mixed-signal ICs, particularly in deep submicron CMOS.

5.6 Conclusions

Techniques for state observation which are studied in the control literature either require access to both the system inputs and outputs, or require the system under obser-

vation to be minimum phase. An approximate time delayed state estimator was proposed which can be applied to any state space system with unknown inputs. This technique is particularly relevant in analog filters with digitally programmable parameters since the LMS algorithm requires estimates of the filter's internal state signals, but it is impractical to build the additional analog circuitry to generate and digitize them. Instead, time delayed state estimates are calculated digitally from the filter output and used for LMS adaptation with minimal additional analog hardware.

The technique is also potentially immune to dc offset effects which usually hinder analog implementations of the LMS algorithm. Since the state estimates are generated digitally from the filter output, an adaptive dc tap at the filter output results in state estimates which are offset-free. However, the adaptation algorithm is computationally complex compared to straight LMS adaptation and efficient hardware implementations are still an open issue.

The general state observation technique is potentially applicable to any adaptation algorithm where the filter's state information is required but inaccessible. Although only the LMS algorithm was discussed here due to its popularity and straightforward implementation, the state estimates could also be used in (for example) the Newton-Raphson algorithm.

Filter adaptation using unknown input state observation was verified in behavioral simulations on a 2-tap adaptive transversal filter and a 5th order continuous time filter. The approximate inverse filter must have a long enough impulse response to provide good state estimates, even when the signal path filter has spectral nulls. If the filter $\tilde{H}^{-1}(z)$ is made long enough, the performance of the adaptation algorithm approaches that of the standard LMS algorithm.

5.7 References

[1] D.A. Johns, W.M. Snelgrove and A.S. Sedra, "Continuous-Time LMS Adaptive Recursive Filters," *IEEE Trans. Circuits Syst.*, vol. CAS-38, pp. 769-778, July 1991.

[2] A. Carusone and D. A. Johns, "Obtaining Digital Gradient Signals for Analog Adaptive Filters," *IEEE Int. Symp. Circuits and Syst.*, May 1999.

[3] R.E. Kalman, “A New Approach to Linear Filtering and Prediction Problems,” *ASME J. of Basic Eng.*, vol. 820, pp. 35-45, 1960.

[4] R.E. Kalman and R.S. Bucy, “New Results in Linear Filtering and Prediction Problems,” *ASME J. of Basic Eng.*, vol. 830, pp. 95-108, 1961.

[5] D.G. Luenberger, “Observing the State of a Linear System,” *IEEE Trans. on Military Electronics*, vol. MIL-8, pp. 74-80, 1964.

[6] J. Jin, M. Tahk and C. Park, “Time-delayed state and unknown input observation,” *Int. J. Control*, vol. 66, pp. 733-745, 1997.

[7] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, N.J., 1975.

Chapter 6

Conclusion

6.1 Summary

The problems that limit the use of analog adaptive filters in modern integrated systems were studied. Chief among those problems are analog circuit complexity (particularly in deep submicron CMOS process technologies) and dc offset effects. The problems are overcome by combining a digitally programmable analog filter in the signal path with a digital adaptation algorithm. Both the analog circuits and the digital adaptation algorithms were discussed.

Chapter 2 described several circuit techniques for implementing a digitally programmable analog filter in a CMOS process technology. The techniques were verified in hardware on a prototype 5th order integrated filter with all poles and zeros digitally programmable. Although the results were modest (particularly the linearity), the degree of programmability realized had previously only been reported in BiCMOS processes with cutoff frequencies below 10 MHz [1] (compared with up to 70 MHz in this work).

The remainder of the thesis sought to alleviate the LMS algorithm's need for direct access to the filter's internal state signals by performing some additional signal processing digitally. Three techniques were described: Chapter 3, the dithered linear search (DLS); Chapter 4, the LMS algorithm with a co-ordinate transform (LMS-CT and LMS-ICT); and Chapter 5, the LMS algorithm using an unknown input state observer (LMS with UIO). All three are compared qualitatively in terms of their performance and complexity

in Table 6.1. All compare favorably with the LMS algorithm in terms of analog circuit complexity. This is the most important criteria in many data communication systems since the performance requirements are not stringent and digital circuits are relatively easy and cheap to integrate in deep submicron CMOS. The DLS algorithm is simple and general, but its rate of convergence is orders of magnitude slower than the others restricting it to applications where slow adaptation can be tolerated. The LMS algorithm with a UIO converges quickly (at the same rate as the LMS algorithm) but has the greatest computational complexity (and, as a result, was not verified on the prototype filter). The LMS-CT/ICT algorithms combine a simple hardware implementation with performance identical to the LMS algorithm making them preferable to the LMS algorithm for almost any analog adaptive linear combiner in a mixed signal application. Unfortunately, the LMS-CT/ICT algorithms can not be used to adapt filter poles.

Algorithm	**LMS**	**DLS**	**LMS-CT/ICT**	**LMS with UIO**
Adaptation rate	Fast	Slow	Fast	Fast
Analog circuit complexity	High	Low	Low	Low
Digital circuit complexity	Low[a]	Low	Medium[a]	High
Applicable filter structures	Any with a unimodal performance surface	Any with a unimodal performance surface	Adaptive Linear Combiner	Any with a unimodal performance surface

Table 6.1 Comparison of digital algorithms for analog adaptive filters.

a. Digital circuit complexity is significantly reduced by using signed variations.

6.2 Future Work

The digital adaptation algorithms in this thesis have been presented without reference to any particular application in order to highlight their generality. The logical next step is to target a specific application in order to demonstrate their practicality. Applications where analog adaptive filters offer practical advantages over digital adaptive filters are generally restricted to those where all of the following criteria apply:

1. The input signal already exists in analog form, so that additional data converters are not required to perform the filtering in the analog domain.
2. Only moderate linearity is required, because high linearity (> 50 dB) is difficult to achieve in analog circuitry with low supply voltages.
3. High speed and/or low power are important design objectives, otherwise digital circuits are efficient and easier to implement.

All of the applications discussed in Section 1.2 meet these criteria, with the possible exception of magnetic storage read channels where CMOS process scaling appears to have allowed digital adaptive filters to catch up to the system requirements [2] and analog adaptive filters are losing favor. The most promising candidates in the near-term are long haul fibre optic and short-distance very-high-speed wired (including chip-to-chip) channels. In these applications, filter zeros could be adapted quickly and with low complexity using a subsampled SS-LMS-ICT algorithm providing programmable high-frequency boost to equalize the lowpass channels. Furthermore, the linearity requirements are modest since very few data levels are transmitted, and a digital approach is unlikely to be practical at the targeted speeds (10+ Gb/s) in the near future.

Several other potential topics for future study suggest themselves:

- **Integrated implementation**: An obvious extension would be to prototype an analog adaptive filter integrated alongside one of the digital adaptation algorithms described above. This would be particularly interesting for the LMS algorithm with a UIO since it is not clear how to efficiently implement the required digital signal processing in hardware.
- **Programmable analog signal conditioner**: The circuit techniques that were proven successful in Chapter 2 could be used to implement a highly programmable analog Gm-C filter in 0.18 μm CMOS. Based upon the experience gained from the prototype filter design, realistic design specifications would be a filter with 40 dB linearity and poles & zeros that are digitally programmable beyond 100 MHz. Such a block could be used as a general analog signal conditioner in a wide range of applications, and would be far beyond any previously reported digitally configurable analog filter.

- **Further stability analyses**: General stability proofs of the adaptation algorithms described in this thesis are still lacking. Stability of the DLS algorithm using multilevel dither was not verified analytically. Neither was stability of the signed variations of the LMS-CT and LMS-ICT algorithms. The LMS algorithm with a UIO uses heuristic techniques to perform gradient estimation when there is a transmission zero in the adapted filter; further stability analysis is also warranted here.
- **Exploration of other dither signals**: Other, possibly multilevel, dither signals may offer advantages over either pseudorandom or Hadamard sequence dither. One possibility is to use delta-sigma modulation to shift the dither energy outside of the adapted filter's bandwidth of interest.
- **Use of UIO for other applications**: Although the technique developed for unknown input state observation in Chapter 5 was intended for gradient estimation in analog filter adaptation, the same technique could be useful anywhere an approximate, time delayed, unknown input state observer is needed. It would be interesting to explore other applications for the technique.
- **Combine algorithms**: It might be possible to combine the different algorithms in the adaptation of a single filter to take advantage of the merits of each. For instance, the DLS could be used during startup to adapt the filter's poles. Then, the LMS-ICT could be used to adapt the zeros. The input sampling circuit required by the LMS-ICT could be eliminated by using the approximate inverse technique from Chapter 5 to obtain estimates of the filter input from its output.

6.3 References

[1] A. Hematy and G. W. Roberts, "A Fully-Programmable Analog Log-Domain Filter Circuit," *1998 IEEE Int. Conf. Circuits and Syst.*, pp. 309-312, May 1998.

[2] S. Rylov, A. Rylyakov, J. Tierno, M. Immediato, M. Beakes, M. Kapur, P. Ampadu, and D. Pearson, "A 2.3 GSample/s 10-tap Digital FIR Filter for Magnetic Recording Read Channels," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, pp. 190-191, Feb. 2001.