1997

Adaptive interference canceling using higher order statistics

Andrew B. Napell
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_theses

Recommended Citation

Napell, Andrew B., "Adaptive interference canceling using higher order statistics" (1997). *Master's Theses*. 1459.

DOI: <https://doi.org/10.31979/etd.9sds-fvj2>
https://scholarworks.sjsu.edu/etd_theses/1459

This Thesis is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Theses by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

**ADAPTIVE INTERFERENCE CANCELING
USING HIGHER ORDER STATISTICS**

A Thesis

Presented to

**The Faculty of the Department of Electrical Engineering
San Jose State University**

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Andrew B. Napell

May 1997

UMI Number: 1384708

UMI Microform 1384708
Copyright 1997, by UMI Company. All rights reserved.

**This microform edition is protected against unauthorized
copying under Title 17, United States Code.**

UMI
300 North Zeeb Road
Ann Arbor, MI 48103

© 1997

Andrew B. Napell

ALL RIGHTS RESERVED

APPROVED FOR THE DEPARTMENT OF ELECTRICAL ENGINEERING

A handwritten signature in black ink, appearing to read "Essam Marouf", written over a horizontal line.

Professor Essam Marouf

A handwritten signature in black ink, appearing to read "Richard Duda", written over a horizontal line.

Professor Richard Duda

A handwritten signature in black ink, appearing to read "Avtar Singh", written over a horizontal line.

Professor Avtar Singh

APPROVED FOR THE UNIVERSITY

A handwritten signature in black ink, appearing to read "M. Lou Swandowski", written over a horizontal line.

ABSTRACT

ADAPTIVE INTERFERENCE CANCELING USING HIGHER ORDER STATISTICS

by Andrew B. Napell

In this thesis we show how a higher order statistics (HOS) based algorithm can be used to update the filter coefficients in a transversal Adaptive Interference Canceller (AIC). The HOS algorithm based AIC is shown to outperform an NLMS algorithm based AIC in several important areas, including convergence speed, excess error, sensitivity to uncorrelated additive noise, and performance when the filter length is underdetermined.

Results of computer simulations using the order 2 and order 4 HOS algorithms are compared to results obtained using the NLMS algorithm for stationary and non-stationary, narrowband and wideband interference, both in the absence and presence of uncorrelated noise. The improved performance of the HOS algorithm over the NLMS is shown to be accompanied by a significant increase in computational complexity, and for the classes of signals considered we conclude that it is difficult to justify using the order 4 instead of the order 2 HOS algorithm.

TABLE OF CONTENTS

Glossary	vi
Introduction.....	1
Chapter 1 The Adaptive Interference Canceler	5
Chapter 2 The LMS and NLMS Algorithms.....	13
Chapter 3 Moments and Cumulants	18
Chapter 4 The HOS Algorithm	30
Chapter 5 The Order 2 HOS Algorithm	40
Chapter 6 The Order 4 HOS Algorithm	51
Chapter 7 Performance of the Order 4 HOS Algorithm	59
Chapter 8 The Order 4 HOS Algorithm and Non-Stationary Interference	68
Chapter 9 Comparing The Order 4 and Order 2 HOS Algorithms.....	73
Chapter 10 Summary and Conclusions	78
Bibliography	83
Appendix A Matlab 4.0 code	85

GLOSSARY

AIC	Adaptive Interference Canceler.
α	Positive constant to prevent dividing by zero.
$E\{\cdot\}$	Statistical expectation operator.
Cum $\{\cdot\}$	Cumulant operator.
CP3	Cumulant Property 3.
CP5	Cumulant Property 5.
Γ	The domain in N -dimensional space in which the cumulants are calculated.
HOS	Higher Order Statistics.
λ	Forgetting factor.
L	Number of lag values used in the computation of cumulants.
LMS	Least Mean Squared (algorithm).
M	Number of points in the cumulants.
μ_f	Constant adaptation parameter of the order 4 HOS algorithm.
μ_s	Constant adaptation parameter of the order 2 HOS algorithm.
N	Number of taps in the transversal adaptive filter.
NLMS	Normalized Least Mean Squared (algorithm).
PDF	Probability density function.
μ	Constant adaptation parameter of the NLMS algorithm.

INTRODUCTION

ADAPTIVE INTERFERENCE CANCELING

Adaptive Interference Canceling is a variation of optimal filtering which is highly effective in a broad variety of applications. In optimal filtering, a signal which is corrupted by additive interference is passed through a filter which is designed to suppress the interference while passing the signal relatively unchanged. If the filter is fixed, it must be designed based on a priori knowledge of the statistics of both the signal and the interference. If the filter is adaptive, it continuously updates its own parameters based on the incoming signals themselves.

The Adaptive Interference Canceler (also called the Adaptive Noise Canceler) uses a reference input derived from a sensor located at a point in the interference field where the signal is weak or, ideally, undetectable [16]. This reference input is passed through an adaptive filter, and the output of that filter is subtracted from the primary input, which contains both the signal and a correlated version of the interference. As a result, the interference in the primary signal is greatly attenuated, or canceled. Although what we are referring to as interference can, in general, be deterministic or stochastic in nature, wherever possible we will reserve the term “interference” for the correlated signal to be canceled, and the term “noise” for the uncorrelated additive random measurement noise which may appear at the primary and reference inputs.

Adaptive Interference Canceling finds applications in a wide variety of fields, including: Adaptive antennas and arrays, including adaptive beamforming and canceling antenna sidelobe interference; medical electronics and imaging, including canceling 60 cycle interference in electrocardiography, canceling the mother's ECG during the observation of the fetal ECG, and canceling the ECG of the donor-heart during heart transplant electrocardiography; speech communications, including echo canceling during long distance telephone conversations, and the cancellation of obscuring noises from speech transmissions made in the presence of loud machinery or other sources of background sound [16].

UPDATING THE TAPS OF THE ADAPTIVE FILTER: THE LMS AND NLMS ALGORITHMS

Conventionally, an Adaptive Interference Canceler uses either the LMS or NLMS algorithm for the updating of its tap weight coefficients [5]. The reasons for doing so are quite compelling, as the algorithms are extremely compact and easy to code, have an exceptionally low computational requirement, and are effective on a great variety of signals. However, the performance of the LMS and NLMS algorithms are highly problem dependent. The LMS algorithm is sensitive to both the statistics of the signal at the reference input and to the choice of the step size parameter. The NLMS variation of the LMS algorithm, while not being sensitive to the reference signal statistics, shares the sensitivity of its predecessor to the choice of step size. Furthermore, the performance of both algorithms is severely compromised by the presence of uncorrelated additive noise at the primary and reference inputs [5].

AN ALTERNATIVE: THE HOS ALGORITHM

In this thesis we present an alternative to the LMS and NLMS algorithms, which we call the HOS algorithm, in which the tap weight coefficients of the adaptive filter are updated based on the higher order statistics of the incoming signals, rather than on the signals themselves [5]. We show that the performance of the HOS algorithm is insensitive to both the reference signal statistics and to the choice of step size. Most importantly, it is much less affected by uncorrelated noise at the primary and reference inputs than the LMS and NLMS algorithms. However, we also show that the improved performance of the HOS algorithm comes at a price. The algorithm is highly complex and much more difficult to code, and requires a significant increase in computational complexity. It is important to keep in mind, as we shall see in Chapter 1, that an Adaptive Interference Canceler can not, in general, remove uncorrelated noise, and so the performance of the HOS algorithm may not be better than that of the NLMS algorithm when the uncorrelated noise power is large.

STRUCTURE OF THE THESIS

In Chapter 1, we develop the theory of operation of the Adaptive Interference Canceler, and analyze its steady-state behavior in terms of Wiener filter theory. In Chapter 2, we give a brief derivation of the LMS and NLMS algorithms, as the gradient descent methodology they employ will serve as a jumping off point in the forthcoming development of the HOS algorithm. In Chapter 3, we give definitions for, and discuss basic properties of the higher order statistical quantities which we will be using, i.e.,

moments and cumulants. Chapter 4 is devoted to the development of the HOS algorithm in general, for any order $n > 1$. In Chapter 5 we develop the HOS algorithm specifically for order $n = 2$, and compare the results of computer simulations to those achieved with the NLMS algorithm. In Chapter 6, we develop the HOS algorithm specifically for order $n = 4$. Chapter 7 is dedicated to comparing the results of computer simulations using the order 4 HOS algorithm to those achieved with the NLMS algorithm for the case of stationary interference. In Chapter 8, we compare the performances of these two algorithms, but for the case of non-stationary interference. In Chapter 9 we directly compare the performance of the order 4 HOS algorithm to that of the order 2 algorithm in the presence of uncorrelated noise. Finally, in Chapter 10 we summarize our work and state our conclusions. The final section contains the bibliographical references. All computer programs used, coded in Matlab 4.0, are included in Appendix A.

Chapter 1

THE ADAPTIVE INTERFERENCE CANCELER

Figure 1.1 shows a block diagram depicting the conceptual scheme employed by the Adaptive Interference Canceler. The following summary of the operation of an ideal Adaptive Interference Canceler has been adapted from [16].

Figure 1.1. Adaptive Interference Canceler conceptual scheme.

Let the input to the primary sensor be the sum of the signal s and an interference i_0 , and the input to the reference sensor be the interference i_1 . Let the output from the adaptive filter be y and let the system output be the error signal e , that is

$$e = s + i_0 - y. \quad (1-1)$$

The practical objective is to produce a system output e that is a best fit to the signal s in a least squares sense.

Assume that s , i_0 , i_1 , and y are stationary, zero-mean signals where s is uncorrelated with i_0 and i_1 , but i_0 and i_1 are mutually correlated. Squaring (1-1) and taking expected values yields

$$\begin{aligned} E\{e^2\} &= E\{s^2\} + E\{(i_0 - y)^2\} + 2E\{s(i_0 - y)\} \\ &= E\{s^2\} + E\{(i_0 - y)^2\}. \end{aligned}$$

Since the signal power $E\{s^2\}$ will be unaffected while the filter is adjusted to minimize $E\{e^2\}$, the minimum output power will be

$$E_{\min}\{e^2\} = E\{s^2\} + E_{\min}\{(i_0 - y)^2\}.$$

So by adjusting the filter to minimize $E\{e^2\}$, $E\{(i_0 - y)^2\}$ is also minimized. In other words, the filter output y is then a best least-squares estimate of the primary interference i_0 . Furthermore, when $E\{(i_0 - y)^2\}$ is minimized, $E\{(e - s)^2\}$ is also minimized, since from equation (1-1) we have $(e - s) = (i_0 - y)$. Therefore, adapting the filter to minimize the total output power causes the output e to be a best least-squares estimate of the signal s .

It is important to note that the output e will in general contain some noise in addition to the signal s . From equation (1-1) the output noise is given by $(i_0 - y)$. Since minimizing $E\{e^2\}$ minimizes $E\{(i_0 - y)^2\}$, minimizing the total output power minimizes the noise power at the output. Since the signal at the output remains constant, minimizing the total output power maximizes the output signal-to-noise ratio.

THE ADAPTIVE INTERFERENCE CANCELER AS A WIENER FILTER

If we assume stationary, stochastic inputs to the Adaptive Interference Canceler, its steady state performance closely approximates that of a fixed Wiener filter, and Wiener filter theory provides a convenient framework for mathematical analysis. For the analysis we will assume that the filter is a linear, discrete, infinitely long two sided adaptive transversal filter (or a noncausal IIR filter), which is designed to be optimal in the mean-square-error sense.

Referring to the block diagram of Figure 1.2, let the primary input $x(k)$ consist of a signal $s(k)$ plus the sum of two noises, $I(k)$ and $n_p(k)$. The reference input $z(k)$ consists of the sum of two other noises, $n_r(k)$ and $I(k)$ convolved with $g(k)$, where $g(k)$ is the impulse response of the LTI filter whose transfer function is $G(z)$. The noises $n_p(k)$ and $n_r(k)$ are uncorrelated with each other, with $s(k)$, with $I(k)$, and with $I(k)$ convolved with $g(k)$. The noises $I(k)$ and $I(k)$ convolved with $g(k)$ have a common origin, are correlated with each other, are uncorrelated with $s(k)$, and have finite power spectra at all frequencies.

Figure 1.2. Adaptive Interference Canceler with uncorrelated noise at the primary and reference inputs.

Figure 1.3. Wiener filter.

If we assume that the adaptive process has converged and the minimum mean square error solution has been found, then the adaptive filter $H(z)$ is equivalent to the Wiener filter of Figure 1.3, whose input is $z(k)$, whose desired response is $x(k)$, and whose error signal is $e(k)$. The error surface for such a filter is quadratic, and the minimum point on the surface corresponds to the optimum weight vector $\mathbf{H}^*$, the weights of the optimum Wiener filter with transfer function $H^*(z)$. From Wiener filter theory [16], we know that this optimal transfer function can be expressed as a ratio of power spectra, i.e.

$$H^*(z) = \frac{\Phi_{xz}(z)}{\Phi_{zz}(z)}$$

where $\Phi_{xz}(z)$ is the Z-Transform of the cross-correlation between $z(k)$ and $x(k)$, and $\Phi_{zz}(z)$ is the Z-Transform of the autocorrelation of $z(k)$. The cross power spectrum in the numerator depends only on the correlated components between the primary and reference inputs and is given by

$$\Phi_{xz}(z) = \Phi_{xx}(z)G(z^{-1}).$$

The spectrum of the filter's input, in the denominator, can be expressed in terms of the spectra of the two uncorrelated signals $I(k)$ convolved with $g(k)$, and $n_r(k)$. The spectrum of $I(k)$ convolved with $g(k)$ is $\Phi_{II}(z)|G(z)|^2$ and the spectrum of $n_r(k)$ is $\Phi_{n_r n_r}(z)$.

Therefore, the filter's input spectrum is

$$\Phi_{zz}(z) = \Phi_{n_r n_r}(z) + \Phi_{II}(z)|G(z)|^2$$

and the Wiener transfer function becomes

$$H^*(z) = \frac{\Phi_{II}(z)G(z^{-1})}{\Phi_{n_r n_r}(z) + \Phi_{II}(z)|G(z)|^2}.$$

It is important to note that $H^*(z)$ is independent of both the signal spectrum itself, $\Phi_{ss}(z)$, and the spectrum of the uncorrelated noise at the primary input, $\Phi_{n_p n_p}(z)$.

An informative special case to consider occurs when the additive noise $n_r(k)$ at the reference input is zero. In this case $\Phi_{n_r n_r}(z) = 0$ and the optimal transfer function simplifies to

$$H^*(z) = \frac{1}{G(z)}.$$

This is exactly what we would expect, as our ideal adaptive filter causes the noise $I(k)$ to be perfectly canceled at the system output $e(k)$. Notice, however, that the uncorrelated noise at the primary input $n_p(k)$ remains **uncanceled** and appears with the signal $s(k)$ at the system output.

If we define

$$\rho_{\text{output}}(z) = \frac{\text{signal power density}}{\text{output noise power density}} \quad \text{and}$$

$$\rho_{\text{primary}}(z) = \frac{\text{signal power density}}{\text{primary noise power density}},$$

then assuming that the signal spectrum is greater than zero at all frequencies we can write the ratio

$$\frac{\rho_{\text{output}}(z)}{\rho_{\text{primary}}(z)} = \frac{\Phi_{II}(z) + \Phi_{n_p n_p}(z)}{\Phi_{\text{output noise}}(z)}$$

where the output noise power spectrum (the denominator) is the sum of three components. One is due to the propagation of $n_p(k)$ directly to the output, another is due to the propagation of $n_r(k)$ to the output through $H^*(z)$, and the third is due to the propagation of $I(k)$ to the output through $G(z)$ and $H^*(z)$. Therefore, the output noise power spectrum is

$$\Phi_{\text{output noise}}(z) = \Phi_{n_p n_p}(z) + \Phi_{n_r n_r}(z) |H^*(z)|^2 + \Phi_{II}(z) [1 - G(z)H^*(z)]^2.$$

We now define the ratios of the spectra of the uncorrelated to the spectra of the correlated noises at the primary and reference inputs, respectively, as

$$A(z) \equiv \frac{\Phi_{n_p n_p}(z)}{\Phi_{II}(z)} \quad (\text{noise-to-noise density ratio at the primary input})$$

$$B(z) \equiv \frac{\Phi_{n_r n_r}(z)}{\Phi_{II}(z) |G(z)|^2} \quad (\text{noise-to-noise density ratio at the reference input}).$$

With these definitions the Wiener transfer function can be rewritten as

$$H^*(z) = \frac{1}{G(z)[B(z) + 1]}$$

and the output noise power spectrum can be rewritten as

$$\Phi_{\text{output noise}}(z) = \Phi_{\text{II}}(z)A(z) + \Phi_{\text{II}}(z) \frac{B(z)}{B(z) + 1}.$$

Finally, after substitution and simplification, the ratio of the primary to the output noise power density becomes

$$\frac{\rho_{\text{output}}(z)}{\rho_{\text{primary}}(z)} = \frac{\text{primary noise power density}}{\text{output noise power density}} = \frac{[A(z) + 1][B(z) + 1]}{A(z) + A(z)B(z) + B(z)}.$$

This is a general expression representing the performance of an ideal noise canceler with stationary, stochastic signals and noises as inputs. From it we can estimate the level of noise reduction to be expected from an ideal noise canceler containing an unconstrained two sided (noncausal IIR) Wiener filter. Notice that the signal $s(k)$ propagates to the output undistorted, with a transfer function of unity. Real adaptive filters, however, generally introduce some amount of signal distortion.

The performance of the noise canceling system is limited by the uncorrelated to correlated noise-to-noise density ratios at the primary and reference inputs, $A(z)$ and $B(z)$.

For relatively small $A(z)$ and $B(z)$, $\frac{\rho_{\text{output}}(z)}{\rho_{\text{primary}}(z)}$ is relatively large, and the canceler performs

more effectively. The following special cases demonstrate the desirability of low noise levels at both inputs.

$$1. \ A(z) \text{ small:} \quad \frac{\rho_{\text{output}}(z)}{\rho_{\text{primary}}(z)} \cong \frac{1 + B(z)}{B(z)}$$

$$2. B(z) \text{ small: } \frac{\rho_{\text{output}}(z)}{\rho_{\text{primary}}(z)} \cong \frac{1 + A(z)}{A(z)}$$

$$3. A(z) \text{ and } B(z) \text{ small: } \frac{\rho_{\text{output}}(z)}{\rho_{\text{primary}}(z)} \cong \frac{1}{A(z) + B(z)}$$

These cases imply infinite improvement in output to primary signal-to-noise density ratio when both $A(z)$ and $B(z)$ are zero. In this case there is total removal of noise at the system output, with the resulting perfect signal reconstruction. In practice, when $A(z)$ and $B(z)$ are small, other factors such as finite filter length and misadjustment due to gradient estimation noise become important in limiting the system's performance.

In the next chapter, we will briefly investigate the standard algorithms used for updating the weights of the adaptive filter, the LMS and NLMS algorithms.

Chapter 2

THE LMS AND NLMS ALGORITHMS

THE LEAST MEAN SQUARE (LMS) ALGORITHM

Figure 2.1. FIR structure of order $(N-1)$.

The output sequence $\{y(k)\}$ of the order $(N-1)$ FIR structure pictured in Figure 2.1 is given by the convolution of the input sequence $\{z(k)\}$ with the impulse response of the tap weight coefficient sequence $\{h_k\}$, i.e.

$$y(k) = \sum_{i=0}^{N-1} h_i z(k-i).$$

This finite sum of products can be written more compactly, in matrix notation, as the vector inner product

$$y(k) = \mathbf{H}^T(k) \mathbf{Z}(k) \quad \text{where}$$

$$\mathbf{H}(k) = [h_0 \ h_1 \ \dots \ h_{N-1}]^T \quad \text{at time } k, \text{ and}$$

$$\mathbf{Z}(k) = [z(k) \ z(k-1) \ \dots \ z(k-N+1)]^T.$$

Then, the error sequence $\{e(k)\}$ is given by [12]

$$e(k) = d(k) - y(k) = d(k) - \mathbf{H}^T(k)\mathbf{Z}(k).$$

If we let the LMS cost function, J , be the mean squared error

$$J = E\{e^2(k)\}$$

then we find that the quadratic error surface defined by J can be expressed graphically in N dimensions. In order to minimize J , the LMS algorithm arrives at the optimal tap weight coefficients by iteratively traversing down the error surface, in the direction of the negative gradient (steepest descent), until the global minimum is found [13].

Suppose that the filter tap weight coefficient vector at iteration k is given by $\mathbf{H}(k)$. The gradient is then computed, and a small step proportional to that gradient is taken, leading to a reduction in the cost function J . Equivalently, we can write

$$\mathbf{H}(k+1) = \mathbf{H}(k) - \mu \nabla_H \quad \text{where}$$

$$\nabla_H = \frac{\partial}{\partial \mathbf{H}} J.$$

The gradient of J with respect to the filter tap weight coefficient vector $\mathbf{H}$ can be estimated at each iteration directly from the input data $\{x(k), d(k)\}$. Let us compute the gradient of the instantaneous squared error, i.e.,

$$\begin{aligned} \hat{\nabla}(k) &= \frac{\partial}{\partial \mathbf{H}} [e^2(k)] \\ &= 2e(k) \frac{\partial}{\partial \mathbf{H}} e(k) \end{aligned}$$

$$\begin{aligned}
&= 2e(k) \frac{\partial}{\partial \mathbf{H}} [d(k) - \mathbf{H}^T(k) \mathbf{Z}(k)] \\
&= -2e(k) \mathbf{Z}(k) .
\end{aligned}$$

The vector $\hat{\nabla}(k)$ is termed the instantaneous gradient, since it is based on only the values of e and $\mathbf{Z}$ at iteration k [14]. If we now use $\hat{\nabla}(k)$ in place of the true gradient $\nabla(k)$ in the iterative update equation for $\mathbf{H}$, the result is the LMS algorithm, summarized below.

LMS ALGORITHM

For $k = 1, 2, 3, \dots$

$$\begin{aligned}
\mathbf{Z}(k) &= [z(k) \ z(k-1) \ \dots \ z(k-N+1)]^T \\
y(k) &= \mathbf{H}^T(k) \mathbf{Z}(k) \\
e(k) &= d(k) - y(k) \\
\mathbf{H}(k+1) &= \mathbf{H}(k) + 2\mu e(k) \mathbf{Z}(k)
\end{aligned}$$

End

It has been widely shown [12], [14] et al., that convergence of the LMS algorithm is assured if $0 < \mu < \frac{1}{\lambda_{\max}}$ where $\lambda_{\max}$ is the maximum of the N eigenvalues of the autocorrelation matrix $\mathbf{R} = \mathbf{Z}^T \mathbf{Z}$.

THE NORMALIZED LMS (NLMS) ALGORITHM

In the standard LMS algorithm given above, the correction applied to the tap weight coefficient vector at iteration $k+1$ is directly proportional to the tap input vector $\mathbf{Z}(k)$. Therefore, when $\mathbf{Z}(k)$ is large, the algorithm suffers from the problem of *gradient noise amplification* [15]. In the NLMS algorithm, the correction applied to the tap weight

coefficient vector is normalized with respect to the squared Euclidean norm of the tap input vector.

The NLMS algorithm is shown in [15] to be the solution to the constrained optimization problem of minimizing the squared Euclidean norm of the change $\partial \mathbf{H}(k+1) = \mathbf{H}(k+1) - \mathbf{H}(k)$ in the tap weight coefficient vector subject to the constraint $\mathbf{H}^T(k+1)\mathbf{Z}(k) = d(k)$.

Alternatively, we note that since the largest possible value of μ is determined by $\lambda_{\max}$, the largest eigenvalue of $\mathbf{R}$, we would like to find some bound for $\lambda_{\max}$. According to [14] the average value of the inner product of the tap input vector $\mathbf{Z}(k)$ with itself equals the sum of the eigenvalues. Therefore, we can write

$$\text{Average}\{\mathbf{Z}^T(k)\mathbf{Z}(k)\} = \sum_{i=1}^N \lambda_i \geq \lambda_{\max}$$

since all $\lambda_i \geq 0$. In other words, the average value of the inner product is an upper bound

for $\lambda_{\max}$. Hence, by choosing the step size parameter to be $\frac{\mu}{\alpha + \mathbf{Z}^T(k)\mathbf{Z}(k)}$ the stability requirement, on average, will be satisfied. The term α is a small positive constant which is included to insure that the update term does not become excessively large for very small values of $\mathbf{Z}^T(k)\mathbf{Z}(k)$. We can now summarize the Normalized LMS algorithm [14].

NLMS ALGORITHM

For $k = 1, 2, 3, \dots$

$$\mathbf{Z}(k) = [z(k) \ z(k-1) \ \dots \ z(k-N+1)]^T$$

$$y(k) = \mathbf{H}^T(k) \mathbf{Z}(k)$$

$$e(k) = d(k) - y(k)$$

$$\mathbf{H}(k+1) = \mathbf{H}(k) + \frac{2\mu e(k) \mathbf{Z}(k)}{\alpha + \mathbf{Z}^T(k) \mathbf{Z}(k)}$$

End

Because of their simplicity and effectiveness, the LMS and NLMS algorithms are the workhorses for tap weight updating of adaptive FIR filters. Before developing a higher order statistics based algorithm as an alternative, we first need to investigate the definitions and properties of the fundamental entities of higher order statistics, moments and cumulants. This is the subject of Chapter 3.

Chapter 3

MOMENTS AND CUMULANTS

Given a set of n real random variables $\{x_1, x_2, \dots, x_n\}$, their joint characteristic function is defined as [1]

$$\Phi(\omega_1, \omega_2, \dots, \omega_n) \triangleq E\left\{\exp\left(j(\omega_1 x_1 + \omega_2 x_2 + \dots + \omega_n x_n)\right)\right\} \quad (3-1)$$

where the ω_i are real numbers. A second form of the joint characteristic function is defined [1] as the natural logarithm of $\Phi(\omega_1, \omega_2, \dots, \omega_n)$, i.e.

$$\tilde{\Psi}(\omega_1, \omega_2, \dots, \omega_n) \triangleq \ln[\Phi(\omega_1, \omega_2, \dots, \omega_n)]. \quad (3-2)$$

The joint moments of order $q = k_1 + k_2 + \dots + k_n$ of the set of random variables are given by

$$\text{Mom}[x_1^{k_1}, x_2^{k_2}, \dots, x_n^{k_n}] \triangleq E\{x_1^{k_1} x_2^{k_2} \dots x_n^{k_n}\} = (-j)^q \frac{\partial^q \Phi(\omega_1, \omega_2, \dots, \omega_n)}{\partial \omega_1^{k_1} \partial \omega_2^{k_2} \dots \partial \omega_n^{k_n}} \bigg|_{\omega_1 = \omega_2 = \dots = \omega_n = 0} \quad (3-3)$$

The joint cumulants of order q of the same set of random variables are given by

$$\text{Cum}[x_1^{k_1}, x_2^{k_2}, \dots, x_n^{k_n}] \triangleq (-j)^q \frac{\partial^q \tilde{\Psi}(\omega_1, \omega_2, \dots, \omega_n)}{\partial \omega_1^{k_1} \partial \omega_2^{k_2} \dots \partial \omega_n^{k_n}} \bigg|_{\omega_1 = \omega_2 = \dots = \omega_n = 0} \quad (3-4)$$

Inspection of equations (3-3) and (3-4) reveals that the joint cumulants of a set of random variables can be expressed in terms of their joint moments [1].

Example. Given the random variable $\{x\}$, its moments

$$\text{Mom}[x] = E\{x\} = r_1$$

$$\text{Mom}[x, x] = E\{x^2\} = r_2$$

$$\text{Mom}[x, x, x] = E\{x^3\} = r_3$$

$$\text{Mom}[x, x, x, x] = E\{x^4\} = r_4$$

are found by expanding its characteristic function in a Taylor series, i.e.

$$\begin{aligned}\Phi(\omega) &= E\{\exp(j\omega x)\} \\ &= E\left\{1 + j\omega x + \frac{(j\omega x)^2}{2!} + \frac{(j\omega x)^3}{3!} + \frac{(j\omega x)^4}{4!} + \dots\right\} \\ &= 1 + j\omega E\{x\} - \frac{\omega^2}{2!} E\{x^2\} - j\frac{\omega^3}{3!} E\{x^3\} + \frac{\omega^4}{4!} E\{x^4\} + \dots \\ &= 1 + j\omega r_1 - \frac{\omega^2}{2!} r_2 - j\frac{\omega^3}{3!} r_3 + \frac{\omega^4}{4!} r_4 + \dots\end{aligned}$$

and then applying equation (3-3) and working out the differentiations about zero. Next,

Rewriting equation (3-4) as

$$\text{Cum}[x^q] = (-j)^q \frac{d^q}{d\omega^q} \ln \Phi(\omega) \Big|_{\omega=0}$$

and working out the differentiations about zero, we find that the moments of $\{x\}$ are

related to its cumulants by

$$\text{Cum}[x] = c_1 = r_1$$

$$\text{Cum}[x, x] = c_2 = r_2 - r_1^2$$

$$\text{Cum}[x, x, x] = c_3 = r_3 - 3r_1 r_2 + 2r_1^3$$

$$\text{Cum}[x, x, x, x] = c_4 = r_4 - 4r_1 r_3 + 12r_1^2 r_2 - 3r_2^2 - 6r_1^4. \quad (3-5)$$

Since the moments about the origin of a probability distribution are the expected values of the random variable which has the given distribution, the moments (about the origin) of order q of $\{x\}$ can be generated from [2]

$$r_q = E\{x^q\} = \int_{-\infty}^{\infty} x^q f(x) dx. \quad (3-6)$$

Example. Let the real (continuous) random variable $\{x\}$ have zero mean and be normally distributed, i.e. according to the Gaussian probability density function [2]

$$f(x) = \frac{1}{\sqrt{2\pi} \sigma} \exp\left(-\frac{x^2}{2\sigma^2}\right).$$

Substitution of the Gaussian probability density function into equation (3-6) yields (after integration)

$$\begin{aligned} r_1 &= 0 \\ r_2 &= \sigma^2 \\ r_3 &= 0 \\ r_4 &= 3\sigma^4. \end{aligned}$$

Substituting these results into equations (3-5) yields

$$\begin{aligned} c_1 &= 0 \\ c_2 &= \sigma^2 \\ c_3 &= 0 \\ c_4 &= 0. \end{aligned}$$

Proceeding in this manner leads to two very important results: First, for the Gaussian probability density function all cumulants of order q greater than 2 are identically equal to zero. Second, for *all* symmetric probability density functions, all moments r_q and cumulants c_q for q odd are identically equal to zero [1].

The general relationship between the joint moments and cumulants, of order $q = n$, of $\{x_1, x_2, \dots, x_n\}$ is given by [1]

$$\text{Cum}[x_1, x_2, \dots, x_n] = \sum (-1)^{p-1} (p-1)! E \left\{ \prod_{i \in s_1} x_i \right\} E \left\{ \prod_{i \in s_2} x_i \right\} \dots E \left\{ \prod_{i \in s_p} x_i \right\} \quad (3-7)$$

where the summation extends over all partitions $(s_1, s_2, \dots, s_p)$ for $p = 1, 2, \dots, n$, of the set of integers $\{1, 2, \dots, n\}$. Notice that in order to compute the joint cumulant of order n , one requires knowledge of all moments up to and including order n . The following examples demonstrate the application of equation (3-7).

Example. Order $n = 2$. If we are given two random variables $\{x_1, x_2\}$ the set of integers $\{1, 2\}$ is partitioned into

$$\begin{aligned} p = 1 & \quad s_1 = \{1, 2\} \\ p = 2 & \quad s_1 = \{1\} \quad s_2 = \{2\} \end{aligned}$$

and equation (3-7) yields

$$\text{Cum}[x_1, x_2] = E\{x_1 x_2\} - E\{x_1\}E\{x_2\}.$$

Example. Order $n = 3$. For three random variables $\{x_1, x_2, x_3\}$ the set of integers $\{1, 2, 3\}$ is partitioned into

$p = 1$	$s_1 = \{1, 2, 3\}$		
$p = 2$	$s_1 = \{1\}$	$s_2 = \{2, 3\}$	
	$s_1 = \{2\}$	$s_2 = \{1, 3\}$	
	$s_1 = \{3\}$	$s_2 = \{1, 2\}$	
$p = 3$	$s_1 = \{1\}$	$s_2 = \{2\}$	$s_3 = \{3\}$

In this case we have

$$\begin{aligned} \text{Cum}[x_1, x_2, x_3] &= E\{x_1 x_2 x_3\} - E\{x_1\}E\{x_2 x_3\} - E\{x_2\}E\{x_1 x_3\} - E\{x_3\}E\{x_1 x_2\} \\ &\quad + 2E\{x_1\}E\{x_2\}E\{x_3\}. \end{aligned}$$

Example. Order $n = 4$. Finally, for the case of four random variables $\{x_1, x_2, x_3, x_4\}$ the set of integers $\{1, 2, 3, 4\}$ is partitioned into

p	s_1	s_2	s_3	s_4
1	1, 2, 3, 4			
2	1	2, 3, 4		
	2	1, 3, 4		
	3	1, 2, 4		
	4	1, 2, 3		
	1, 2	3, 4		
	1, 3	2, 4		
	1, 4	2, 3		
3	1, 2	3	4	
	1, 3	2	4	
	1, 4	2	3	
	2, 3	1	4	
	2, 4	1	3	
	3, 4	1	2	
4	1	2	3	4

which yields the general form for the joint cumulant of order $n = 4$, i.e.

$$\begin{aligned} \text{Cum}[x_1, x_2, x_3, x_4] &= E\{x_1 x_2 x_3 x_4\} - E\{x_1\}E\{x_2 x_3 x_4\} - E\{x_2\}E\{x_1 x_3 x_4\} \\ &\quad - E\{x_3\}E\{x_1 x_2 x_4\} - E\{x_4\}E\{x_1 x_2 x_3\} - E\{x_1 x_2\}E\{x_3 x_4\} \\ &\quad - E\{x_1 x_3\}E\{x_2 x_4\} - E\{x_1 x_4\}E\{x_2 x_3\} \\ &\quad + 2E\{x_1 x_2\}E\{x_3\}E\{x_4\} + 2E\{x_1 x_3\}E\{x_2\}E\{x_4\} \\ &\quad + 2E\{x_1 x_4\}E\{x_2\}E\{x_3\} + 2E\{x_2 x_3\}E\{x_1\}E\{x_4\} \\ &\quad + 2E\{x_2 x_4\}E\{x_1\}E\{x_3\} + 2E\{x_3 x_4\}E\{x_1\}E\{x_2\} \\ &\quad - 6E\{x_1\}E\{x_2\}E\{x_3\}E\{x_4\}. \end{aligned} \tag{3-8}$$

Notice that for zero-mean random variables equation (3-8) simplifies to

$$\begin{aligned} \text{Cum}[x_1, x_2, x_3, x_4] = & E\{x_1 x_2 x_3 x_4\} \\ & - E\{x_1 x_2\}E\{x_3 x_4\} - E\{x_1 x_3\}E\{x_2 x_4\} - E\{x_1 x_4\}E\{x_2 x_3\}. \end{aligned} \quad (3-8a)$$

The properties of cumulants are well documented in [1] and [3]. Two of these properties, which are very important for our purposes, are presented next. Adopting the notation of [3] we shall refer to these as Cumulant Property 3 (CP3) and Cumulant Property 5 (CP5), and they are defined as follows [3].

(CP3) Cumulants are additive in their arguments, i.e. cumulants of sums equal sums of cumulants. Even when the random variables x and y are *not* statistically independent,

$$\text{Cum}[x + y, z_1, z_2, \dots, z_n] = \text{Cum}[x, z_1, z_2, \dots, z_n] + \text{Cum}[y, z_1, z_2, \dots, z_n].$$

(CP5) Cumulants of a sum of statistically independent quantities equal the sum of the cumulants of the individual quantities. So, if the random variables $\{x_i\}$ are independent from the random variables $\{y_i\}$ then

$$\text{Cum}[x_1 + y_1, x_2 + y_2, \dots, x_n + y_n] = \text{Cum}[x_1, x_2, \dots, x_n] + \text{Cum}[y_1, y_2, \dots, y_n].$$

MOMENTS AND CUMULANTS OF STATIONARY PROCESSES

Let us consider the values of the random process $\{X(k)\}$ for any set of time instants $k_1, k_2, \dots, k_n$ where n is any positive integer. In general, the samples $X(k_i), i = 1 \dots n$ are n random variables statistically characterized by [10] their joint probability density function $f(x_1, x_2, \dots, x_n)$.

If $\{X(k)\}$ is a (strict-sense) stationary random process, then its statistical properties are invariant to a shift of the origin, so the processes $\{X(k)\}$ and $\{X(k+c)\}$ have the same statistics for any c . Specifically, the joint density of the random variables $X(k_1+c)$ and $X(k_2+c)$ is independent of c and is dependent [11] only on the time difference $k_1 - k_2$.

Therefore, if $\{X(k)\}$, $k = 0, \pm 1, \pm 2, \pm 3, \pm 4, \dots$ is a real stationary discrete-time random process, and its moments up to order n exist, then we can write [1] for $m_i = 0, \pm 1, \pm 2, \dots$

$$\text{Mom}[X(k), X(k+m_1), \dots, X(k+m_{n-1})] = E\{X(k)X(k+m_1) \dots X(k+m_{n-1})\} \quad (3-9)$$

where the set of random variables $\{x_1, x_2, \dots, x_n\}$ from equation (3-3) have been generated by distinct observations of the random process $\{X(k)\}$ at times $\{k, k+m_1, \dots, k+m_{n-1}\}$. Note that the expected values in equation (3-9) depend only on the time differences $\{m_1, m_2, \dots, m_{n-1}\}$.

Now, we can define the moments of a real stationary discrete-time random process as [1],

$$r_n^x(m_1, m_2, \dots, m_{n-1}) \triangleq E\{X(k)X(k+m_1)X(k+m_2) \dots X(k+m_{n-1})\}. \quad (3-10)$$

Note that $r_n^x(m_1, m_2, \dots, m_{n-1})$, the n th order moment function of $\{X(k)\}$, is a function in $(n-1)$ dimensions. The second order moment function $r_2^x(m_1) = E\{X(k)X(k+m_1)\}$ is

simply the autocorrelation of $X(k)$, while $r_3^x(m_1, m_2)$ and $r_4^x(m_1, m_2, m_3)$ are the third and fourth order moments, respectively [4].

In the same manner, we define the cumulants of a real stationary discrete-time random process as [1],

$$c_n^x(m_1, m_2, \dots, m_{n-1}) \triangleq \text{Cum}[X(k), X(k+m_1), \dots, X(k+m_{n-1})]. \quad (3-11)$$

Likewise, the n th order cumulant function of $\{X(k)\}$ is a function in $(n-1)$ dimensions.

Combining equations (3-7), (3-10), and (3-11) we obtain the following relationships between the moment and cumulant sequences of the process $\{X(k)\}$:

Example. Order $n = 1$.

$$c_1^x = \text{Cum}[X(k)] = E\{X(k)\} = r_1^x \quad (3-12)$$

Example. Order $n = 2$.

$$\begin{aligned} c_2^x(m_1) &= \text{Cum}[X(k), X(k+m_1)] = E\{X(k)X(k+m_1)\} - E\{X(k)\}E\{X(k+m_1)\} \\ c_2^x(m_1) &= r_2^x(m_1) - [r_1^x]^2 \end{aligned} \quad (3-13)$$

Since $E\{X(k)\} = E\{X(k+m_i)\}$ for a stationary process.

Example. Order $n = 3$.

$$\begin{aligned} c_3^x(m_1, m_2) &= \text{Cum}[X(k), X(k+m_1), X(k+m_2)] \\ c_3^x(m_1, m_2) &= E\{X(k)X(k+m_1)X(k+m_2)\} - E\{X(k)\}E\{X(k+m_1)X(k+m_2)\} \\ &\quad - E\{X(k+m_1)\}E\{X(k)X(k+m_2)\} - E\{X(k+m_2)\}E\{X(k)X(k+m_1)\} \\ &\quad + 2E\{X(k)\}E\{X(k+m_1)\}E\{X(k+m_2)\} \\ c_3^x(m_1, m_2) &= r_3^x(m_1, m_2) - r_1^x r_2^x(m_2 - m_1) \\ &\quad - r_1^x r_2^x(m_2) - r_1^x r_2^x(m_1) \end{aligned}$$

$$+ 2[r_1^x]^3 \quad (3-14)$$

Example. Order $n = 4$.

$$\begin{aligned} c_4^x(m_1, m_2, m_3) &= \text{Cum}[X(k), X(k+m_1), X(k+m_2), X(k+m_3)] \\ c_4^x(m_1, m_2, m_3) &= r_4^x(m_1, m_2, m_3) - r_1^x r_3^x(m_2 - m_1, m_3 - m_1) - r_1^x r_3^x(m_2, m_3) \\ &\quad - r_1^x r_3^x(m_1, m_3) - r_1^x r_3^x(m_1, m_2) - r_2^x(m_1) r_2^x(m_3 - m_2) \\ &\quad - r_2^x(m_2) r_2^x(m_3 - m_1) - r_2^x(m_3) r_2^x(m_2 - m_1) \\ &\quad + 2[r_1^x]^2 [r_2^x(m_1) + r_2^x(m_2) + r_2^x(m_3) \\ &\quad + r_2^x(m_2 - m_1) + r_2^x(m_3 - m_1) + r_2^x(m_3 - m_2)] \\ &\quad - 6[r_1^x]^4 \end{aligned} \quad (3-15)$$

As we saw before, the first order cumulant c_1^x is identical to the first order moment m_1^x , and is simply the mean value of the process. Note that the second order cumulant $c_2^x(m_1) = r_2^x(m_1) - [r_1^x]^2$ is the covariance sequence, where $r_2^x(m_1)$ is the autocorrelation sequence [4].

If the process $\{X(k)\}$ has zero mean, then substitution of $r_1^x = 0$ into (3-13), (3-14), and (3-15) yields the following important relationships:

$$c_2^x(m_1) = r_2^x(m_1) \quad (3-16a)$$

$$c_3^x(m_1, m_2) = r_3^x(m_1, m_2) \quad (3-16b)$$

$$\begin{aligned} c_4^x(m_1, m_2, m_3) &= r_4^x(m_1, m_2, m_3) \\ &\quad - r_2^x(m_1) r_2^x(m_3 - m_2) - r_2^x(m_2) r_2^x(m_3 - m_1) - r_2^x(m_3) r_2^x(m_2 - m_1) \end{aligned} \quad (3-16c)$$

CROSS-MOMENTS AND CROSS-CUMULANTS

Given the real stationary discrete-time random processes $\{X_1(k)\}, \{X_2(k)\}, \dots$ where $k = 0, \pm 1, \pm 2, \pm 3, \dots$ we define the cross-moment and cross-cumulant functions of order n of the processes as [1]

$$r_n^{x_1 x_2 \dots x_n}(m_1, m_2, \dots, m_{n-1}) \stackrel{\Delta}{=} E\{X_1(k) X_2(k+m_1) \dots X_n(k+m_{n-1})\} \quad (3-17)$$

$$c_n^{x_1 x_2 \dots x_n}(m_1, m_2, \dots, m_{n-1}) \stackrel{\Delta}{=} \text{Cum}[X_1(k), X_2(k+m_1), \dots, X_n(k+m_{n-1})]. \quad (3-18)$$

Example. Order $n = 2$.

$$\begin{aligned} c_2^{x_1 x_2}(m_1) &= \text{Cum}[X_1(k), X_2(k+m_1)] \\ &= E\{X_1(k)X_2(k+m_1)\} - E\{X_1(k)\}E\{X_2(k+m_1)\} \\ &= E\{X_1(k)X_2(k+m_1)\} - r_1^{x_1}r_1^{x_2} \\ &= E\{X_1(k)X_2(k+m_1)\} \text{ for zero mean processes.} \end{aligned} \quad (3-19)$$

Note that for the *non* zero mean case $c_2^{x_1 x_2}(m_1)$ is the cross-covariance sequence, and for the zero mean case it is identical to the second order moment, i.e., the cross-correlation sequence.

In particular, suppose we are given the real, stationary, discrete-time, zero mean random processes $\{X(k)\}$, $\{Y(k)\}$, and $\{Z(k)\}$. Then

$$c_2^{xz}(m_1) = E\{X(k)Z(k+m_1)\} = r_2^{xz}(m_1)$$

$$c_2^{zy}(m_1) = E\{Z(k)Y(k+m_1)\} = r_2^{zy}(m_1).$$

Example. Order $n = 3$.

$$c_3^{x_1 x_2 x_3}(m_1, m_2) = \text{Cum}[X_1(k), X_2(k+m_1), X_3(k+m_2)]$$

$$\begin{aligned}
&= E\{X_1(k)X_2(k+m_1)X_3(k+m_2)\} - E\{X_1(k)\}E\{X_2(k+m_1)X_3(k+m_2)\} \\
&\quad - E\{X_2(k+m_1)\}E\{X_1(k)X_3(k+m_2)\} \\
&\quad - E\{X_3(k+m_2)\}E\{X_1(k)X_2(k+m_1)\} \\
&\quad + 2E\{X_1(k)\}E\{X_2(k+m_1)\}E\{X_3(k+m_2)\} \\
&= E\{[X_1(k) - E\{X_1(k)\}][X_2(k+m_1) - E\{X_2(k+m_1)\}] \\
&\quad [X_3(k+m_2) - E\{X_3(k+m_2)\}]\} \\
&= E\{[X_1(k) - r_1^{x_1}][X_2(k+m_1) - r_1^{x_2}][X_3(k+m_2) - r_1^{x_3}]\} \\
&= E\{X_1(k)X_2(k+m_1)X_3(k+m_2)\} \text{ for zero mean processes.} \quad (3-20)
\end{aligned}$$

In particular

$$c_3^{xyz}(m_1, m_2) = E\{X(k)Y(k+m_1)Z(k+m_2)\} = r_3^{xyz}(m_1, m_2).$$

Example. Order $n = 4$.

Combining equations (3-16), (3-17), and (3-18) we can now write by inspection the following two important fourth order cross-cumulants for zero mean stationary processes:

$$\begin{aligned}
c_4^{xxxx}(m_1, m_2, m_3) &= r_4^{xxxx}(m_1, m_2, m_3) - r_2^{xx}(m_1)r_2^{xx}(m_3 - m_2) - r_2^{xx}(m_2)r_2^{xx}(m_3 - m_1) \\
&\quad - r_2^{xx}(m_3)r_2^{xx}(m_2 - m_1)
\end{aligned}$$

$$\begin{aligned}
c_4^{xyxx}(m_1, m_2, m_3) &= r_4^{xyxx}(m_1, m_2, m_3) - r_2^{xy}(m_1)r_2^{xx}(m_3 - m_2) - r_2^{xy}(m_2)r_2^{xx}(m_3 - m_1) \\
&\quad - r_2^{xy}(m_3)r_2^{xx}(m_2 - m_1)
\end{aligned}$$

Likewise using (3-8a) we can write the more general expressions for these cross-cumulants:

$$\begin{aligned}
c_4^{xxxx}(0, m_1, m_2, m_3) &= r_4^{xxxx}(0, m_1, m_2, m_3) - r_2^{xx}(0, m_1)r_2^{xx}(m_2, m_3) \\
&\quad - r_2^{xx}(0, m_2)r_2^{xx}(m_1, m_3) - r_2^{xx}(0, m_3)r_2^{xx}(m_1, m_2)
\end{aligned}$$

$$\begin{aligned}
c_4^{xyxx}(0, m_1, m_2, m_3) &= r_4^{xyxx}(0, m_1, m_2, m_3) - r_2^{xy}(0, m_1)r_2^{xx}(m_2, m_3) \\
&\quad - r_2^{xy}(0, m_2)r_2^{xx}(m_1, m_3) - r_2^{xy}(0, m_3)r_2^{xx}(m_1, m_2).
\end{aligned}$$

The two m th order cross-cumulants $c_n^{\underline{z}\dots\bar{z}}(m_1, \dots, m_{n-1})$ and $c_n^{\bar{z}\dots\underline{z}}(m_1, \dots, m_{n-1})$ form the basis for the Adaptive Interference Canceler Higher Order Statistics (HOS) based tap weight update algorithm, the development of which is the subject of Chapter 4.

Chapter 4

THE HOS ALGORITHM

HOS ALGORITHM FOR ORDER N

Figure 4.1. Adaptive Interference Canceler for order n HOS algorithm.

Referring to the block diagram of Figure 4.1, let $\{x(k)\}$ denote measurements taken at the primary sensor at times $k = 1, 2, 3, \dots$ and let $\{z(k)\}$ denote measurements taken at the reference sensor. Furthermore, let $\{x(k)\}$ be a linear combination of the signal itself $\{s(k)\}$, the interference $\{I(k)\}$, and the measurement noise at the primary sensor $\{n_p(k)\}$ satisfying

$$x(k) = s(k) + I(k) + n_p(k) \quad (4-1)$$

and let $\{z(k)\}$ be a linear combination of $\{w(k)\}$ and the measurement noise at the reference sensor $\{n_r(k)\}$ satisfying

$$z(k) = w(k) + n_r(k) \quad (4-2)$$

where $\{w(k)\}$ is a reference signal highly correlated with the interference $\{I(k)\}$. We assume the relationship between $\{w(k)\}$ and $\{I(k)\}$ can be represented by an LTI (linear time invariant) transformation satisfying

$$I(k) = \sum_j g(j)w(k-j). \quad (4-3)$$

We assume that the measurement noises $\{n_s(k)\}$ and $\{n_p(k)\}$ are zero mean, stationary, white or colored Gaussian processes which are uncorrelated with each other, with the signal $\{s(k)\}$, and with the interference $\{I(k)\}$. The signal $\{s(k)\}$ is zero mean, and can be wideband or narrowband, deterministic or random, or some combination. The interference $\{I(k)\}$ is a zero mean, non Gaussian (narrowband or wideband) process which is uncorrelated with the signal $\{s(k)\}$.

Let $\{y(k)\}$ denote the output of the transversal adaptive filter of length N (filter order $N-1$) satisfying

$$y(k) = \sum_{i=0}^{N-1} h(i)z(k-i) \quad (4-4)$$

where $\{h(0), h(1), h(2), \dots, h(N-1)\}$ are the filter coefficients. Let the cross-cumulant of order n between the primary and reference inputs to the adaptive interference canceler be

$$c_n^{x \dots z}(m_1, m_2, \dots, m_{n-1}) = \text{Cum}[x(k), z(k+m_1), z(k+m_2), \dots, z(k+m_{n-1})]$$

and let the cross-cumulant of order n between the adaptive filter output and the reference input be

$$c_n^{y \dots z}(m_1, m_2, \dots, m_{n-1}) = \text{Cum}[y(k), z(k+m_1), z(k+m_2), \dots, z(k+m_{n-1})].$$

The general form of the cost function to be minimized is defined as the sum of the squared errors between these two order n cumulants as in [5] - [7], i.e.,

$$J_g = \sum_{m_1} \sum_{m_2} \cdots \sum_{m_{n-1}} \left[c_n^{z \cdots z}(m_1, m_2, \dots, m_{n-1}) - c_n^{y \cdots z}(m_1, m_2, \dots, m_{n-1}) \right]^2. \quad (4-5)$$

Since the measurement noises $\{n_p(k)\}$ and $\{n_r(k)\}$ are zero mean, stationary, Gaussian processes, their n th order cumulants are identical to zero, for $n > 2$. Therefore, by operating on equation (4-2) we find

$$\begin{aligned} c_n^{z \cdots z}(m_1, m_2, \dots, m_{n-1}) &= \text{Cum}[z(k), z(k+m_1), z(k+m_2), \dots, z(k+m_{n-1})] \\ &= \text{Cum}[w(k) + n_r(k), w(k+m_1) + n_r(k+m_1), \dots, w(k+m_{n-1}) + n_r(k+m_{n-1})] \\ &= \text{Cum}[w(k), w(k+m_1), \dots, w(k+m_{n-1})] + \text{Cum}[n_r(k), n_r(k+m_1), \dots, n_r(k+m_{n-1})] \\ &= \text{Cum}[w(k), w(k+m_1), \dots, w(k+m_{n-1})] \\ c_n^{z \cdots z}(m_1, m_2, \dots, m_{n-1}) &= c_n^{w \cdots w}(m_1, m_2, \dots, m_{n-1}) \end{aligned} \quad (4-6)$$

where we have applied Cumulant Property 5 (CP5). By operating on equation (4-4) we can write

$$\begin{aligned} c_n^{y \cdots z}(m_1, m_2, \dots, m_{n-1}) &= \text{Cum}[y(k), z(k+m_1), z(k+m_2), \dots, z(k+m_{n-1})] \\ &= \text{Cum}\left[\sum_{i=0}^{N-1} h(i)z(k-i), z(k+m_1), z(k+m_2), \dots, z(k+m_{n-1})\right] \\ &= \sum_{i=0}^{N-1} h(i) \text{Cum}[z(k-i), z(k+m_1), z(k+m_2), \dots, z(k+m_{n-1})] \\ &= \sum_{i=0}^{N-1} h(i) c_n^{z \cdots z}(k+m_1-(k-i), k+m_2-(k-i), \dots, k+m_{n-1}-(k-i)) \\ c_n^{y \cdots z}(m_1, m_2, \dots, m_{n-1}) &= \sum_{i=0}^{N-1} h(i) c_n^{z \cdots z}(m_1+i, m_2+i, \dots, m_{n-1}+i) \end{aligned} \quad (4-7)$$

where we have used CP3. Finally, from equation (4-1) we can write

$$\begin{aligned}
c_n^{x \dots z}(m_1, m_2, \dots, m_{n-1}) &= \text{Cum}[x(k), z(k+m_1), z(k+m_2), \dots, z(k+m_{n-1})] \\
&= \text{Cum}[s(k) + I(k) + n_p(k), w(k+m_1) + n_r(k+m_1), \dots, w(k+m_{n-1}) + n_r(k+m_{n-1})] \\
&= \text{Cum}[I(k), w(k+m_1), w(k+m_2), \dots, w(k+m_{n-1})] \\
c_n^{x \dots z}(m_1, m_2, \dots, m_{n-1}) &= c_n^{Iw \dots w}(m_1, m_2, \dots, m_{n-1}) \tag{4-8}
\end{aligned}$$

where in addition to the terms involving $\text{Cum}[n_p(k)]$ and $\text{Cum}[n_r(k+m_i)]$, the terms involving $\text{Cum}[s(k), w(k+m_i)]$ have vanished owing to the fact that $\{s(k)\}$ and $\{w(k)\}$ are uncorellated. In the same manner which led to equation (4-7), equation (4-3) becomes

$$c_n^{Iw \dots w}(m_1, m_2, \dots, m_{n-1}) = \sum_j g(j) c_n^{w \dots w}(m_1 + j, m_2 + j, \dots, m_{n-1} + j). \tag{4-9}$$

Finally, we can rewrite equation (4-5) as follows:

$$J_g = \sum_{m_1} \sum_{m_2} \dots \sum_{m_{n-1}} \left[\sum_j g(j) c_n^{w \dots w}(m_1 + j, \dots, m_{n-1} + j) - \sum_{i=0}^{N-1} h(i) c_n^{w \dots w}(m_1 + i, \dots, m_{n-1} + i) \right]^2 \tag{4-10}$$

The physical interpretation which can be made here is that by minimizing the cost function we are asking the adaptive filter to act in such a way so that the n th order cumulant of the reference signal $w(k)$, as computed at the output of the adaptive filter $h(k)$, matches as closely as possible on a point-by-point basis the n th order cumulant of the interference, which is simply $w(k)$ convolved with $g(k)$. In other words, we can think of

the filter $h(k)$ as trying to produce a signal $y(k)$ which is optimally “correlated” (in a sum-of-squared-errors sense) with the signal $x(k)$ in the n th order cumulant domain.

Consider again the generalized cost function J_g in equation (4-5), where $\{m_1, \dots, m_{n-1}\}$ may be defined to include the entire $n - 1$ dimensional space. If we choose a proper domain Γ which is a subset of that space, the new cost function becomes [5] one special case of equation (4-5), i.e.,

$$J = \sum_{(m_1, m_2, \dots, m_{n-1})} \dots \sum_{\in \Gamma} \left[c_n^{\mathbf{x} \dots \mathbf{z}}(m_1, \dots, m_{n-1}) - \sum_{i=0}^{N-1} h(i) c_n^{\mathbf{z} \dots \mathbf{z}}(m_1 + i, \dots, m_{n-1} + i) \right]^2 \quad (4-11)$$

Now, we define the $M \times 1$ column vector $\mathbf{C}_{\mathbf{x} \dots \mathbf{z}}$ and the $M \times N$ matrix $\mathbf{C}_{\mathbf{z} \dots \mathbf{z}}$ as being constructed from the individual elements of $c_n^{\mathbf{x} \dots \mathbf{z}}(m_1, \dots, m_{n-1})$ and $c_n^{\mathbf{z} \dots \mathbf{z}}(m_1, \dots, m_{n-1})$ respectively, in such a way that when the element in the i th row of column vector $\mathbf{C}_{\mathbf{x} \dots \mathbf{z}}$ is

$$\mathbf{C}_{\mathbf{x} \dots \mathbf{z}}(i) = [c_n^{\mathbf{x} \dots \mathbf{z}}(m_1, \dots, m_{n-1})]$$

the i th row of matrix $\mathbf{C}_{\mathbf{z} \dots \mathbf{z}}$ is

$$\mathbf{C}_{\mathbf{z} \dots \mathbf{z}}(i, :) = [c_n^{\mathbf{z} \dots \mathbf{z}}(m_1, \dots, m_{n-1}) \quad c_n^{\mathbf{z} \dots \mathbf{z}}(m_1 + 1, \dots, m_{n-1} + 1) \quad \dots \\ \dots \quad c_n^{\mathbf{z} \dots \mathbf{z}}(m_1 + N - 1, \dots, m_{n-1} + N - 1)]$$

where the notation $\mathbf{C}_{\mathbf{z} \dots \mathbf{z}}(i, :)$ means “row i , all columns of matrix $\mathbf{C}_{\mathbf{z} \dots \mathbf{z}}$.” The parameter M denotes the total number of points in the $n - 1$ dimensional domain Γ . For example, in the order $n = 3$ case we might choose $\Gamma = \{(m_1, m_2)\}$ in the 2 dimensional plane which satisfies $0 \leq m_1, m_2 \leq L - 1$, in which case M would be equal to L^2 . Similarly, in the order $n = 4$ case we might choose $\Gamma = \{(m_1, m_2, m_3)\}$ in 3 dimensional space which satisfies $0 \leq m_1, m_2, m_3 \leq L - 1$, in which case M would be equal to L^3 .

We also define the $N \times 1$ adaptive filter coefficient column vector

$$\mathbf{H} = [h_0 \ h_1 \ \dots \ h_{N-1}]^T.$$

We can now rewrite equation (4-11) in matrix notation as

$$J = (\mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}} - \mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}} \mathbf{H})^T (\mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}} - \mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}} \mathbf{H}). \quad (4-12)$$

Taking the partial derivative with respect to $\mathbf{H}$, we arrive at an expression for the gradient vector of the cost function J in terms of the filter coefficients, i.e.

$$\begin{aligned} \nabla_H J &= \frac{\partial J}{\partial \mathbf{H}} \\ &= (-\mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}}^T)(\mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}} - \mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}} \mathbf{H}) + (-\mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}}^T)(\mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}} - \mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}} \mathbf{H}) \\ &= 2(\mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}}^T \mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}} \mathbf{H} - \mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}}^T \mathbf{C}_{\mathbf{z}, \dots, \mathbf{z}}). \end{aligned} \quad (4-13)$$

To employ a gradient type algorithm in practice we need to substitute for the cumulants $c_n^{\mathbf{z}, \dots, \mathbf{z}}(m_1, \dots, m_{n-1})$ and $c_n^{\mathbf{z}, \dots, \mathbf{z}}(m_1, \dots, m_{n-1})$ in equation (4-11) their estimates at iteration k , $\hat{c}_n^{\mathbf{z}, \dots, \mathbf{z}}(k; m_1, \dots, m_{n-1})$ and $\hat{c}_n^{\mathbf{z}, \dots, \mathbf{z}}(k; m_1, \dots, m_{n-1})$, and from these we form the estimate matrices $\hat{\mathbf{C}}_{\mathbf{z}, \dots, \mathbf{z}}(k)$ and $\hat{\mathbf{C}}_{\mathbf{z}, \dots, \mathbf{z}}(k)$. We then have

$$\begin{aligned} \hat{J}(k) &= \sum_{(m_1, m_2, \dots, m_{n-1})} \dots \sum_{\in \Gamma} \left[\hat{c}_n^{\mathbf{z}, \dots, \mathbf{z}}(k; m_1, \dots, m_{n-1}) - \sum_{i=0}^{N-1} h(i) \hat{c}_n^{\mathbf{z}, \dots, \mathbf{z}}(k; m_1 + i, \dots, m_{n-1} + i) \right]^2 \\ &= \left(\hat{\mathbf{C}}_{\mathbf{z}, \dots, \mathbf{z}}(k) - \hat{\mathbf{C}}_{\mathbf{z}, \dots, \mathbf{z}}(k) \mathbf{H}(k) \right)^T \left(\hat{\mathbf{C}}_{\mathbf{z}, \dots, \mathbf{z}}(k) - \hat{\mathbf{C}}_{\mathbf{z}, \dots, \mathbf{z}}(k) \mathbf{H}(k) \right) \end{aligned}$$

as an estimate of J at iteration k , and the gradient estimate becomes

$$\hat{\nabla}_H(k) = 2 \left(\hat{\mathbf{C}}_{\mathbf{z}, \dots, \mathbf{z}}^T(k) \hat{\mathbf{C}}_{\mathbf{z}, \dots, \mathbf{z}}(k) \mathbf{H}(k) - \hat{\mathbf{C}}_{\mathbf{z}, \dots, \mathbf{z}}^T(k) \hat{\mathbf{C}}_{\mathbf{z}, \dots, \mathbf{z}}(k) \right). \quad (4-14)$$

The adaptive filter coefficient update equation can now take the form

$$\mathbf{H}(k+1) = \mathbf{H}(k) - \mu(k) \hat{\nabla}(k). \quad (4-15)$$

The step size parameter $\mu(k)$ is chosen to satisfy

$$0 < \mu(k) < \frac{1}{\text{tr}\{\hat{\mathbf{C}}_{z,\dots,z}^T(k) \hat{\mathbf{C}}_{z,\dots,z}(k)\}}$$

in order to satisfy the stability requirement [5] - [7] without knowledge of the spread of the eigenvalues of $\hat{\mathbf{C}}_{z,\dots,z}^T(k) \hat{\mathbf{C}}_{z,\dots,z}(k)$. Notice that both the gradient estimate and the step size parameter are computed using order n cumulants of the primary and reference inputs, which are not affected by uncorrelated Gaussian noises for $n > 2$. Hence, the adaptive filter coefficient update equation is also independent of these noises for $n > 2$.

As we saw before, the computation of order n cumulants requires knowledge of all moments up to and including order n . Hence, the cumulant estimates used in (4-14) above are calculated from time-average estimates of the corresponding moment functions. Let us show how these moment estimates can be computed.

Suppose we need to replace $r_2^{\mathbf{x}}(m_1) = r_{\mathbf{x}}(m_1) = E\{x(k)z(k+m_1)\}$ by its time estimate $\hat{r}_{\mathbf{x}}(k; m_1) = \frac{1}{k} \sum_{i=1}^k x(i)z(i+m_1)$. At every iteration (new sample) we don't have to recompute the entire summation, because it can be done recursively. We can write

$$\hat{r}_{\mathbf{x}}(k-1; m_1) = \frac{1}{k-1} \sum_{i=1}^{k-1} x(i)z(i+m_1)$$

$$\hat{r}_x(k; m_1) = \frac{1}{k} \left[\sum_{i=1}^{k-1} x(i)z(i + m_1) + x(k)z(k + m_1) \right]$$

$$\hat{r}_x(k; m_1) = \frac{k-1}{k} \frac{1}{k-1} \left[\sum_{i=1}^{k-1} x(i)z(i + m_1) \right] + \frac{1}{k} [x(k)z(k + m_1)]$$

$$\hat{r}_x(k; m_1) = \frac{k-1}{k} \hat{r}_x(k-1; m_1) + \frac{1}{k} x(k)z(k + m_1)$$

If the statistics of the interference are slowly changing with time (i.e. non-stationary), we would like to give the greatest weight to the newest samples, and gradually forget about older samples in the time-average. But the above summation gives all samples equal weight, from the very first sample at $i=1$ to the most recent sample at $i=k$. Therefore, we introduce a “forgetting factor” λ , satisfying $0 < \lambda \leq 1$, the choice of which controls the number of recent samples that are significant in the computation of the time-average.

First, we note that

$$\begin{aligned} \sum_{i=0}^k \lambda^{k-i} &= \lambda^k \sum_{i=0}^k (\lambda^{-1})^i \\ &= \lambda^k \frac{1 - \lambda^{-(k+1)}}{1 - \lambda^{-1}} \\ &= \frac{\lambda^k - \lambda^{-1}}{1 - \lambda^{-1}} \\ &= \frac{-\lambda^{-1}}{1 - \lambda^{-1}} \quad \text{since for large } k \quad \lambda^k \longrightarrow 0 \\ &= \frac{1}{1 - \lambda} \end{aligned}$$

If the argument of the summation is a constant, we would like to normalize in such a way as to recover that constant, i.e. by multiplying by the factor $(1-\lambda)$. Furthermore, for large values of the iteration index k we would like the multiplicative factor of the recursion to be independent of k . Therefore, we calculate the time-average estimate of the moment function for the non-stationary case in the following manner:

$$\hat{r}_{\Sigma}(k; m_1) = \frac{1 - \lambda^{-1}}{\lambda^k - \lambda^{-1}} \sum_{i=0}^k \lambda^{k-i} x(i) z(i + m_1)$$

$$\hat{r}_{\Sigma}(k-1; m_1) = \frac{1 - \lambda^{-1}}{\lambda^{k-1} - \lambda^{-1}} \sum_{i=0}^{k-1} \lambda^{k-1-i} x(i) z(i + m_1)$$

$$\hat{r}_{\Sigma}(k; m_1) = \frac{1 - \lambda^{-1}}{\lambda^k - \lambda^{-1}} \left[\sum_{i=0}^{k-1} \lambda^{k-i} x(i) z(i + m_1) + \lambda^0 x(k) z(k + m_1) \right]$$

$$\hat{r}_{\Sigma}(k; m_1) = \frac{1 - \lambda^{-1}}{\lambda^k - \lambda^{-1}} \left[\sum_{i=0}^{k-1} \lambda^{k-1-i} x(i) z(i + m_1) + x(k) z(k + m_1) \right]$$

$$\hat{r}_{\Sigma}(k; m_1) = \lambda \frac{\lambda^{k-1} - \lambda^{-1}}{\lambda^k - \lambda^{-1}} \frac{1 - \lambda^{-1}}{\lambda^{k-1} - \lambda^{-1}} \sum_{i=0}^{k-1} \lambda^{k-1-i} x(i) z(i + m_1) + \frac{1 - \lambda^{-1}}{\lambda^k - \lambda^{-1}} x(k) z(k + m_1)$$

$$\hat{r}_{\Sigma}(k; m_1) = \lambda \frac{\lambda^{k-1} - \lambda^{-1}}{\lambda^k - \lambda^{-1}} \hat{r}_{\Sigma}(k-1; m_1) + \frac{1 - \lambda^{-1}}{\lambda^k - \lambda^{-1}} x(k) z(k + m_1)$$

For large values of k the recursion becomes

$$\hat{r}_{\Sigma}(k; m_1) = \lambda \hat{r}_{\Sigma}(k-1; m_1) + (1 - \lambda) x(k) z(k + m_1)$$

which satisfies both of the objectives set forth above.

The HOS algorithm, as developed here, is valid for any order n greater than one. The order $n = 2$ version, which calculates its gradient estimate based on auto and cross-

correlation functions, is highly effective when used for updating tap weights in an Adaptive Interference Canceler, and it is the subject of Chapter 5.

Chapter 5

THE ORDER 2 HOS ALGORITHM

The HOS algorithm for order $n = 2$ uses the same form of cost function and gradient descent method of the higher order algorithms, but based on second order statistics.

Therefore, we have $c_n^{\underline{x}, \dots, \underline{x}}(m_1, \dots, m_{n-1}) = c_2^{\underline{x}}(m_1) = r_2^{\underline{x}}(m_1) = r_{\underline{x}}(m_1)$ is simply the cross correlation between the primary and reference inputs, and

$c_n^{\underline{x}, \dots, \underline{x}}(m_1, \dots, m_{n-1}) = c_2^{\underline{x}}(m_1) = r_2^{\underline{x}}(m_1) = r_{\underline{x}}(m_1)$ is the autocorrelation of the reference input. If we choose the domain Γ so that $0 \leq m_1 \leq L - 1$, then the number of points in the cumulants (i.e. the number of rows in $\hat{\mathbf{C}}_{\underline{x}}(k)$ and $\hat{\mathbf{C}}_{\underline{z}}(k)$) will be $M = L$. Now we can write the expression for the cost function estimate as

$$\hat{J}(k) = \sum_{m_1=0}^{L-1} \left[\hat{\mathbf{c}}_{\underline{x}}(k; m_1) - \sum_{i=0}^{N-1} h(i) \hat{\mathbf{c}}_{\underline{z}}(k; m_1 + i) \right]^2 \quad (5-1)$$

$$= \left(\hat{\mathbf{C}}_{\underline{x}}(k) - \hat{\mathbf{C}}_{\underline{z}}(k) \mathbf{H}(k) \right)^T \left(\hat{\mathbf{C}}_{\underline{x}}(k) - \hat{\mathbf{C}}_{\underline{z}}(k) \mathbf{H}(k) \right). \quad (5-1a)$$

Similarly, the gradient estimate becomes

$$\hat{\mathbf{V}}_H(k) = 2 \left(\hat{\mathbf{C}}_{\underline{x}}^T(k) \hat{\mathbf{C}}_{\underline{z}}(k) \mathbf{H}(k) - \hat{\mathbf{C}}_{\underline{z}}^T(k) \hat{\mathbf{C}}_{\underline{x}}(k) \right). \quad (5-2)$$

Now we can summarize the order $n = 2$ version of the Adaptive Interference Canceler HOS algorithm.

HOS algorithm for order 2

Given:

N	number of taps or filter length (filter order = $N - 1$)
L	maximum lag
M	number of points in cumulants ($= L$)
λ	forgetting factor, where $0 < \lambda \leq 1$
α	positive constant
mus	constant adaptation parameter, where $0 < mus < 1$
$x(k)$	primary input at iteration k , where $x(k) = 0$ for $k < 1$
$z(k)$	reference input at iteration k , where $z(k) = 0$ for $k < 1$

Initial Conditions:

$$\begin{aligned} \mathbf{H}(1) &= [h_0(1) \ h_1(1) \ \dots \ h_{N-1}(1)]^T = [0 \ 0 \ \dots \ 0]^T & (N \times 1) \\ \mathbf{R}_{\varepsilon}(0) &= [\hat{r}_{\varepsilon}(0;0) \ \hat{r}_{\varepsilon}(0;1) \ \dots \ \hat{r}_{\varepsilon}(0;M-1)]^T = [0 \ 0 \ \dots \ 0]^T & (M \times 1) \\ \mathbf{R}_{\varepsilon}(0) &= [\hat{r}_{\varepsilon}(0;0) \ \hat{r}_{\varepsilon}(0;1) \ \dots \ \hat{r}_{\varepsilon}(0;M+N-2)]^T = [0 \ 0 \ \dots \ 0]^T & (M+N-1 \times 1) \end{aligned}$$

For $k = 1, 2, 3, \dots$

$$\begin{aligned} \mathbf{Z}_{\varepsilon} &= [z(k) \ z(k+1) \ \dots \ z(k+M-1)]^T \\ \mathbf{Z}_{\varepsilon} &= [z(k) \ z(k+1) \ \dots \ z(k+M+N-2)]^T \end{aligned}$$

for stationary case

$$\begin{aligned} \mathbf{R}_{\varepsilon}(k) &= \frac{k-1}{k} \mathbf{R}_{\varepsilon}(k-1) + \frac{1}{k} x(k) \mathbf{Z}_{\varepsilon} \\ \mathbf{R}_{\varepsilon}(k) &= \frac{k-1}{k} \mathbf{R}_{\varepsilon}(k-1) + \frac{1}{k} z(k) \mathbf{Z}_{\varepsilon} \end{aligned}$$

end

for non-stationary case

$$B = \frac{\lambda^{k-1} - \lambda^{-1}}{\lambda^k - \lambda^{-1}}$$

$$A = \frac{1 - \lambda^{-1}}{\lambda^k - \lambda^{-1}}$$

$$\begin{aligned} \mathbf{R}_{\varepsilon}(k) &= \lambda B \mathbf{R}_{\varepsilon}(k-1) + A x(k) \mathbf{Z}_{\varepsilon} \\ \mathbf{R}_{\varepsilon}(k) &= \lambda B \mathbf{R}_{\varepsilon}(k-1) + A z(k) \mathbf{Z}_{\varepsilon} \end{aligned}$$

end

$$\hat{\mathbf{C}}_{\varepsilon} = \mathbf{R}_{\varepsilon}(k)$$

for $m = 1, 2, \dots, N$

$$\hat{\mathbf{C}}_{\varepsilon}(:, m) = \mathbf{R}_{\varepsilon}(k; m; m+M)$$

end

$$\mathbf{Z} = [z(k) \ z(k-1) \ \dots \ z(k-(N-1))]^T$$

$$\begin{aligned}
y(k) &= \mathbf{H}(k)^T \mathbf{Z} \\
e(k) &= x(k) - y(k) \\
\hat{\mathbf{V}} &= 2(\hat{\mathbf{C}}_{\mathbf{z}}^T \hat{\mathbf{C}}_{\mathbf{z}} \mathbf{H}(k) - \hat{\mathbf{C}}_{\mathbf{z}}^T \hat{\mathbf{C}}_{\mathbf{z}}) \\
\mu &= \frac{mus}{\alpha + \text{tr}(\mathbf{C}_{\mathbf{z}}^T \mathbf{C}_{\mathbf{z}})} \\
\mathbf{H}(k+1) &= \mathbf{H}(k) - \mu \hat{\mathbf{V}} \\
\text{End}
\end{aligned}$$

Let us explain the matrix operations which take place in the for $m = 1, 2, \dots, N$ loop. For each value of m , fill (all rows) column m of matrix $\hat{\mathbf{C}}_{\mathbf{z}}$ with the elements from rows (m) through $(m + M)$ of column vector $\mathbf{R}_{\mathbf{z}}(k)$, where $\mathbf{R}_{\mathbf{z}}(k)$ has been computed recursively as detailed in Chapter 4. With $\hat{\mathbf{C}}_{\mathbf{z}}$ thus constructed (and likewise $\hat{\mathbf{C}}_{\mathbf{x}}$), we are able to use the matrix formulation of the cost function, equation (5-1a).

PERFORMANCE OF HOS ORDER 2 ALGORITHM VS. NLMS ALGORITHM

To compare the performance of the order 2 HOS algorithm with that of the NLMS algorithm, we will generate test signals on the computer, and run both algorithms using Matlab version 4.0.

The signal $s(k)$ will be the quasi-random signal defined by

$$s(k) = a \cos(2\pi FTk)$$

where a is a discrete random variable which takes on only the values 1 and -1, with equal probability, and we have arbitrarily chosen $FT = 0.43$. The reference signal $w(k)$ will be a sum of real sinusoids, $w(k) = \sum_j w_j(k)$, hence the interference will also be a sum of real sinusoids, $I(k) = \sum_j I_j(k)$. Furthermore, each individual $I_j(k)$ is generated by passing

the corresponding $w_j(k)$ through its own moving average (MA) system, $g_j(k)$. The corresponding MA coefficients are

$$g_1 = [1 \ 0.1 \ -0.3 \ 0.5 \ 0.25]$$

$$g_2 = [1 \ 0.5 \ -0.1 \ -0.75 \ 0.41 \ 0.21 \ 0.12 \ -0.05 \ 0.01 \ -0.2 \ -0.15]$$

$$g_3 = [1 \ -0.2 \ 0.2 \ 0.7 \ -0.65 \ 0.3 \ -0.25].$$

Each $w_j(k)$ has the form $w_j(k) = A_j \sin(2\pi F_j T k + \phi_j)$ where the ϕ_j are independent random variables uniformly distributed over the interval $[-\pi, \pi]$. Also, $F_1 T = 0.1$, $F_2 T = 0.25$, and $F_3 T = 0.3$. The sequences $n_r(k)$ and $n_p(k)$ are independent realizations of zero-mean, white Gaussian noise.

Since, with simulated signals, we have at our disposal the signal $s(k)$ separate from the interference $I(k)$ we can judge the performance of the algorithms by plotting a “system error” signal, defined as the difference between the original signal $s(k)$ and the system output $e(k)$, versus iteration k . That is, system error = $s(k) - e(k)$. Clearly, a perfect adaptive interference canceler would have system error = 0.

First, let us consider the case where the interference is a stationary, narrowband signal (single sinusoid) and there are no uncorrelated measurement noises, i.e. $n_r(k) = n_p(k) = 0$. Let the reference signal be $w(k) = w_1(k)$, with $A_1 = 1$. We used a filter length $N = 16$ and positive constant $\alpha = 2$ for both algorithms, and a maximum lag $L = 16$ for the HOS order 2 algorithm (using our nomenclature, the NLMS algorithm has $L = N$ by design). The system errors for the NLMS and HOS order 2 algorithms are shown in Figure 5.1 for three different values of the adaptation parameters μ (NLMS) and mus (HOS order 2).

Figure 5.1. The system errors for the NLMS and HOS order 2 algorithms with $N = 16$, $L = 16$, $\alpha = 2$, $n_r(k) = n_p(k) = 0$, and $I(k) = I_1(k)$ with $A_1 = 1$.

As is evident from Figure 5.1, the performance of the NLMS algorithm is highly sensitive to the choice of constant adaptation parameter μ . The HOS order 2 algorithm, on the other hand, performs essentially equally as well, in terms of convergence speed and excess error, at mitigating the stationary narrowband interference for any value of mus in the same range, when the amplitude of the reference signal is constant. It is important to notice, also, that for $\mu = mus = 0.1$ (that being the approximate value for μ found empirically to optimize the performance of the NLMS algorithm with the given input signals), the HOS order 2 algorithm clearly outperforms the NLMS algorithm at canceling the interference.

Next, we consider the case of multiple narrowband interference with various filter lengths. The interference in this case is $I(k) = I_1(k) + I_3(k)$ with $A_1 = .75$ and $A_2 = 2$. We also used $n_r(k) = n_p(k) = 0$, $\alpha = 2$, $\mu = .075$, $mus = 0.99$, and $\alpha = 2$. The results are shown in Figure 5.2. Of particular interest is the case where we have underestimated the relationship between the reference signal and the interference, and have used too few taps in the adaptive filter, a topic which we will return to later. As shown in panels (a.1) and (b.1), when the number of filter taps is underdetermined, the HOS order 2 algorithm outperforms the NLMS algorithm.

Figure 5.2. System errors for the NLMS and HOS order 2 algorithms with $\mu = .075$, $mus = 0.99$, $\alpha = 2$, $n_r(k) = n_p(k) = 0$, and $I(k) = I_1(k) + I_3(k)$ with $A_1 = .75$ and $A_2 = 2$.

Let us now consider the case of non-stationary interference, and investigate the tracking ability of the HOS order 2 algorithm. As we saw before, for non-stationary

interference we choose a value for λ which is less than one. The closer λ is to one, the greater the number of significant samples in the “data window” and hence the greater the accuracy of the recursive time average estimates of the cumulants. However, values for λ closer to one compromise the ability of the time average recursions to track the statistics of rapidly changing signals. Hence, the optimal choice for λ depends on a priori knowledge of the rate of change of the interference.

For the non-stationary test cases shown in Figure 5.3, we first used an amplitude modulated version of $w_1(k)$ as the reference signal (panel (a.1)) which was convolved with $g_1(k)$ to form the interference $I(k)$. Second, we used a linear FM chirp signal as the reference signal (panel (b.1)) which was convolved with $g_3(k)$ to form the interference $I(k)$. Again, there were no uncorrelated measurement noises. The system errors for the AM case are shown in panels (a.2) and (a.3), and for the FM case in panels (b.2) and (b.3). For the NLMS algorithm we used $N = 10$, $\mu = .075$, and $\alpha = 2$. For the HOS order 2 algorithm we used $N = 10$, $L = 4$, $\mu = 0.99$, $\alpha = 2$, and $\lambda = 0.99$ or $\lambda = 0.995$.

For the AM interference, the performance of the HOS order 2 algorithm is characterized by extremely rapid convergence and smaller excess error by comparison to the NLMS algorithm. For the FM interference, both algorithms converge more slowly, as the adaptive filters must continuously adjust their frequency response, instead of only their amplitude response. However, we can see that the HOS order 2 algorithm surpasses the NLMS algorithm in terms of excess error before the 400th iteration. Both algorithms, we conclude, are well adapted to tracking non-stationary interference. It is interesting to note

that a distinct advantage of the NLMS algorithm over the HOS algorithm is the fact that the algorithm is the same for both stationary and non-stationary interference, relieving the user of the burden of choosing the proper value for λ .

Figure 5.3. System errors for the NLMS and HOS order 2 algorithms with non-stationary interference and $N = 10$, $L = 4$, $\mu = .075$, $\mu_{us} = 0.99$, $\alpha = 2$, $n_r(k) = n_p(k) = 0$. For (a.2) and (a.3) the interference is $I(k) = w(k)*g_1(k)$. For (b.2) and (b.3) the interference is $I(k) = w(k)*g_3(k)$.

Thus far we have seen that the stochastic nature of the signal $s(k)$ compromises the performance of the NLMS algorithm, which estimates its gradient based only on the instantaneous error between the desired signal and the filter output. The HOS algorithm is much less sensitive to the random nature of the signal $s(k)$ because it calculates a “smoothing gradient” at each iteration based on the convolution of the cumulant $c_n^{i_1, \dots, i_n}(m_1, \dots, m_{n-1})$ with the filter tap weight vector. For a deterministic signal $s(k)$,

therefore, we expect the effectiveness of the NLMS algorithm to improve, but as shown in Figure 5.4, the HOS algorithm outperforms it again. For the simulation of Figure 5.4, the signal $s(k)$ is a binary PSK signal defined by

$$s(k) = \text{state} \cdot \cos(2\pi (0.43)k)$$

where state is a discrete random variable taking on the values +1 or -1 with equal probability, and holding each value for a duration of eight samples. The interference is given by $I(k) = I_2(k)$ with $A_2 = 0.5$. In panels (a.1) and (b.1) there is no uncorrelated noise at either the primary or reference input, i.e. $n_r(k) = n_p(k) = 0$. In panels (a.2) and (b.2) the uncorrelated noises have variance $\text{var}\{n_r(k)\} = \text{var}\{n_p(k)\} = 0.005$. We also used $N = 12$, $L = 4$, $\alpha = 2$, $\mu = 0.075$, and $\text{mus} = 0.99$.

Figure 5.4. The system errors for the NLMS and HOS order 2 algorithms applied to a binary PSK signal with $N = 12$, $L = 4$, $\alpha = 2$, $\mu = 0.075$, $\text{mus} = 0.99$, $I(k) = I_2(k)$ with $A_2 = 0.5$. For (a.1) and (b.1), $n_r(k) = n_p(k) = 0$. For (a.2) and (b.2), $\text{var}\{n_r(k)\} = \text{var}\{n_p(k)\} = 0.005$.

While the NLMS algorithm is somewhat effective at canceling the narrowband interference from the binary PSK signal in the absence of noise (panel (a.1)), the HOS

order 2 algorithm is more effective (panel (b.1)). However, in the presence of uncorrelated noise at the primary and reference inputs, the performance of both second order statistics based algorithms degrades. We can see why the performance of the HOS order 2 algorithm in the presence of noise at the reference input is compromised by considering the expression for the gradient more closely.

From equation (5-2) we can write the expression for the order 2 gradient as

$$\nabla_2 = 2(C_{\underline{z}}^T C_{\underline{z}} \mathbf{H} - C_{\underline{z}}^T C_{\underline{x}}).$$

Expanding the expression for the cross-correlation we find

$$\begin{aligned} c_{\underline{x}}(m_1) &= E\{x(k)z(k+m_1)\} \\ &= E\{[s(k) + I(k) + n_p(k)][w(k+m_1) + n_r(k+m_1)]\} \\ &= E\{I(k)w(k+m_1)\} \end{aligned}$$

since the primary and reference noises are uncorrelated with each other, with the signal, and with the interference. On the other hand the expression for the autocorrelation yields upon expansion

$$\begin{aligned} c_{\underline{z}}(t) &= E\{z(k)z(k+m_1)\} \\ &= E\{[w(k) + n_r(k)][w(k+m_1) + n_r(k+m_1)]\} \\ &= E\{w(k)w(k+m_1)\} + E\{n_r(k)n_r(k+m_1)\}. \end{aligned}$$

Therefore, the adaptive filter coefficient update equation will be affected by the autocorrelation of the noise at the reference input, and the performance of the HOS order 2 algorithm will not be independent of uncorrelated Gaussian measurement noises in general [5].

As we saw earlier, for orders greater than two the filter update equation is, in theory, independent of Gaussian noise, so we need to investigate the algorithm in some order higher than two. In the domain of third-order cumulants, symmetrically distributed noise superimposed on a non-symmetrically distributed (i.e. skewed) signal will be rejected asymptotically as the number of data points increases. Interestingly, real (as opposed to synthetic) speech is skewed enough to yield significantly non-zero third-order cumulants, due mainly to non-linearities in the vocal tract [9]. Consequently, a third-order cumulant based speech detector for use on noisy speech signals is presented in [9]. However, results from our simulations using the HOS order 3 algorithm for the extraction of speech from Gaussian noise by linear prediction were inferior to results obtained with the NLMS and other second-order statistics based algorithms. Whereas the third-order cumulants of speech are significant enough for the speech *detection* problem, they are not non-zero enough for high quality linear prediction.

For the problem of the cancellation (e.g. from speech) of quasi-periodic interference, such as loud noises emitted by complex rotating mechanical machinery, we need to develop the HOS algorithm for some order n in which the n th order cumulants of the interference are not identically zero. Noting that these signals can be regarded as non-Gaussian [4], but having symmetric probability density functions, we know that although their third-order cumulants will be zero, their fourth-order cumulants will not. Therefore, in Chapter 6 we will develop the order 4 HOS algorithm.

Chapter 6

THE ORDER 4 HOS ALGORITHM

For order $n = 4$, the cumulants $\hat{c}_n^{x \dots x}(m_1, \dots, m_{n-1}) = \hat{c}_4^{xxx}(m_1, m_2, m_3)$ and $\hat{c}_n^{z \dots z}(m_1, \dots, m_{n-1}) = \hat{c}_4^{zzz}(m_1, m_2, m_3)$ are functions of three independent variables, specifically the lags m_1 , m_2 , and m_3 . As in chapter 5, we will drop the superscript and replace the order $n = 4$ subscript with the signal vector labels for notational convenience. Recalling equation (3-8a), the fourth order cumulants of zero-mean signals are computed from the second and fourth order moment functions according to the relations

$$\begin{aligned} \hat{c}_{\equiv\equiv\equiv}(m_1, m_2, m_3) &= \hat{r}_{\equiv\equiv\equiv}(m_1, m_2, m_3) \\ &\quad - \hat{r}_{\equiv\equiv}(m_1)\hat{r}_{\equiv\equiv}(m_2, m_3) \\ &\quad - \hat{r}_{\equiv\equiv}(m_2)\hat{r}_{\equiv\equiv}(m_1, m_3) \\ &\quad - \hat{r}_{\equiv\equiv}(m_3)\hat{r}_{\equiv\equiv}(m_1, m_2) \end{aligned} \quad (6-1a)$$

$$\begin{aligned} \hat{c}_{\equiv\equiv\equiv}(m_1, m_2, m_3) &= \hat{r}_{\equiv\equiv\equiv}(m_1, m_2, m_3) \\ &\quad - \hat{r}_{\equiv\equiv}(m_1)\hat{r}_{\equiv\equiv}(m_2, m_3) \\ &\quad - \hat{r}_{\equiv\equiv}(m_2)\hat{r}_{\equiv\equiv}(m_1, m_3) \\ &\quad - \hat{r}_{\equiv\equiv}(m_3)\hat{r}_{\equiv\equiv}(m_1, m_2) \end{aligned} \quad (6-1b)$$

Again, the estimates of the moment functions at iteration k are computed recursively using (for the stationary case)

$$\hat{r}_{\equiv\equiv\equiv}(k; m_1, m_2, m_3) = \frac{k-1}{k} \hat{r}_{\equiv\equiv\equiv}(k-1; m_1, m_2, m_3) + \frac{1}{k} x(k)z(k+m_1)z(k+m_2)z(k+m_3)$$

$$\hat{r}_{\equiv}(k; m_1, m_2, m_3) = \frac{k-1}{k} \hat{r}_{\equiv}(k-1; m_1, m_2, m_3) + \frac{1}{k} z(k)z(k+m_1)z(k+m_2)z(k+m_3)$$

$$\hat{r}_{\equiv}(k; p) = \frac{k-1}{k} \hat{r}_{\equiv}(k-1; p) + \frac{1}{k} x(k)z(k+p) \quad p = m_1, m_2, m_3$$

$$\hat{r}_{\equiv}(k; p, q) = \frac{k-1}{k} \hat{r}_{\equiv}(k-1; p, q) + \frac{1}{k} z(k+p)z(k+q) \quad p, q = 0, m_1, m_2, m_3 \quad (5-2)$$

For the non-stationary case, it is understood that the alternate recursive relationships involving λ are used instead.

In order to exploit the linear algebra capabilities of Matlab, we will perform the recursive moment estimate computations by first forming the following vectors:

$$\mathbf{Z0} = [z(k) \ z(k+1) \ \dots \ z(k+L-1)]^T$$

$$\mathbf{Z0big} = [z(k) \ z(k+1) \ \dots \ z(k+L+N-2)]^T$$

We can then recursively compute the matrix quantities

$$\mathbf{r}_{\equiv}(k) = \frac{k-1}{k} \mathbf{r}_{\equiv}(k-1) + \frac{1}{k} x(k)\mathbf{Z0}^T$$

$$\mathbf{R}_{\equiv}(k) = \frac{k-1}{k} \mathbf{R}_{\equiv}(k-1) + \frac{1}{k} z(k)[\mathbf{Z0} \cdot \mathbf{Z0}^T]$$

$$\mathbf{R}_{\equiv big}(k) = \frac{k-1}{k} \mathbf{R}_{\equiv big}(k-1) + \frac{1}{k} z(k)[\mathbf{Z0big} \cdot \mathbf{Z0big}^T]$$

The three dimensional moment functions $\hat{r}_{\equiv}(m_1, m_2, m_3)$ and $\hat{r}_{\equiv}(m_1, m_2, m_3)$ need to be represented by a “stack” of distinct matrices, one matrix for each separate value of the lag m_1 as follows:

$$\mathbf{R}_{\equiv 0}(k) = \frac{k-1}{k} \mathbf{R}_{\equiv 0}(k-1) + \frac{1}{k} x(k)z(k)[\mathbf{Z0} \cdot \mathbf{Z0}^T]$$

$$\begin{aligned}
\mathbf{R}_{\underline{\underline{m}}_1}(k) &= \frac{k-1}{k} \mathbf{R}_{\underline{\underline{m}}_1}(k-1) + \frac{1}{k} x(k)z(k+1)[\mathbf{Z0} \cdot \mathbf{Z0}^T] \\
&\vdots \\
\mathbf{R}_{\underline{\underline{m}}_{L-1}}(k) &= \frac{k-1}{k} \mathbf{R}_{\underline{\underline{m}}_{L-1}}(k-1) + \frac{1}{k} x(k)z(k+L-1)[\mathbf{Z0} \cdot \mathbf{Z0}^T]
\end{aligned}$$

and

$$\begin{aligned}
\mathbf{R}_{\underline{\underline{m}}_0}(k) &= \frac{k-1}{k} \mathbf{R}_{\underline{\underline{m}}_0}(k-1) + \frac{1}{k} z(k)z(k)[\mathbf{Z0big} \cdot \mathbf{Z0big}^T] \\
\mathbf{R}_{\underline{\underline{m}}_1}(k) &= \frac{k-1}{k} \mathbf{R}_{\underline{\underline{m}}_1}(k-1) + \frac{1}{k} z(k)z(k+1)[\mathbf{Z0big} \cdot \mathbf{Z0big}^T] \\
&\vdots \\
\mathbf{R}_{\underline{\underline{m}}_{L+N-2}}(k) &= \frac{k-1}{k} \mathbf{R}_{\underline{\underline{m}}_{L+N-2}}(k-1) + \frac{1}{k} z(k)z(k+L+N-2)[\mathbf{Z0big} \cdot \mathbf{Z0big}^T]
\end{aligned}$$

Let us recap what has been calculated up to this point. The values of $\hat{r}_{\underline{\underline{m}}}(m_1, m_2, m_3)$ for all points in the three-dimensional cube $0 \leq m_1, m_2, m_3 \leq L-1$ are stored in the matrices $\mathbf{R}_{\underline{\underline{m}}_0}, \mathbf{R}_{\underline{\underline{m}}_1}, \dots, \mathbf{R}_{\underline{\underline{m}}_{L-1}}$ and are indexed, within each matrix, by the values of the lags m_2 and m_3 . The values of $\hat{r}_{\underline{\underline{z}}}(p)$ $0 \leq p \leq L-1$ are stored in row vector $\mathbf{r}_{\underline{\underline{z}}}$. The values of $\hat{r}_{\underline{\underline{m}}}(m_1, m_2, m_3)$ for all points in the three-dimensional cube $0 \leq m_1, m_2, m_3 \leq L+N-2$ are stored in the matrices $\mathbf{R}_{\underline{\underline{m}}_0}, \mathbf{R}_{\underline{\underline{m}}_1}, \dots, \mathbf{R}_{\underline{\underline{m}}_{L+N-2}}$ and are again indexed, within each matrix, by the lags m_2 and m_3 . The values of $\hat{r}_{\underline{\underline{z}}}(p, q)$ for $0 \leq p, q \leq L-1$ are stored in matrix $\mathbf{R}_{\underline{\underline{z}}}$ and the values of $\hat{r}_{\underline{\underline{z}}}(p, q)$ for $0 \leq p, q \leq L+N-2$ are stored in matrix $\mathbf{R}_{\underline{\underline{z}}big}$. Again,

the stored values are indexed by the values of the lags p and q , where

$$p, q = 0, m_1, m_2, m_3.$$

Next, we compute the values of $\hat{c}_{\equiv}(m_1, m_2, m_3)$ in the cube

$$0 \leq m_1, m_2, m_3 \leq L - 1 \text{ and the values of } \hat{c}_{\equiv}(m_1, m_2, m_3) \text{ in the cube}$$

$$0 \leq m_1, m_2, m_3 \leq L + N - 2 \text{ using the linear algebra equivalents of equations (6-1a)}$$

and (6-1b). Finally, the cumulant estimate matrices $\hat{\mathbf{C}}_{\equiv}$ and $\hat{\mathbf{C}}_{\equiv}$ are

constructed from values of $\hat{c}_{\equiv}(m_1, m_2, m_3)$ and $\hat{c}_{\equiv}(m_1, m_2, m_3)$ taken from a

domain Γ , which will be a suitably chosen subset of the cube

$$0 \leq m_1, m_2, m_3 \leq L + N - 2.$$

Before choosing a domain Γ , we note that due to the symmetry properties of fourth order moments and cumulants, knowledge of the cumulant sequence in any one of several wedge shaped sectors in three-dimensional space is sufficient for a complete description of the cumulant [8], et al. For simplicity, therefore, we will choose the domain $\Gamma = \{(m_1, m_2, m_3)\}$ such that

$$0 \leq m_1, m_2, m_3 \leq L - 1$$

$$m_2 \leq m_1 \text{ and } m_3 \leq m_1$$

This domain is the rectangular pyramid with vertex at the origin, while having one edge along the m_1 axis (the line $m_2 = m_3 = 0$), and one edge each along the lines $m_1 = m_2$, $m_1 = m_3$, and $m_1 = m_2 = m_3$. Finally, the base of the pyramid lies in the plane

$m_1 = L - 1$. With this choice of Γ , the number of points in the cumulants (i.e. the

number of rows in $\hat{\mathbf{C}}_{\text{cum}}$ and $\hat{\mathbf{C}}_{\text{cum}}$) will be $M = \sum_{n=1}^L n^2 = \frac{L(L+1)(2L+1)}{6}$.

We can now write the expression for the cost function estimate at iteration k as

$$\begin{aligned} \hat{J}(k) &= \sum_{m_1=0}^{L-1} \sum_{m_2=0}^{m_1} \sum_{m_3=0}^{m_2} \left[\hat{c}_{\text{cum}}(k; m_1, m_2, m_3) - \sum_{i=0}^{N-1} h(i) \hat{c}_{\text{cum}}(k; m_1 + i, m_2 + i, m_3 + i) \right]^2 \\ &= \left(\hat{\mathbf{C}}_{\text{cum}}(k) - \hat{\mathbf{C}}_{\text{cum}}(k) \mathbf{H}(k) \right)^T \left(\hat{\mathbf{C}}_{\text{cum}}(k) - \hat{\mathbf{C}}_{\text{cum}}(k) \mathbf{H}(k) \right) \end{aligned}$$

Similarly, the expression for the gradient estimate at iteration k becomes

$$\hat{\nabla}_H(k) = 2 \left(\hat{\mathbf{C}}_{\text{cum}}^T(k) \hat{\mathbf{C}}_{\text{cum}}(k) \mathbf{H}(k) - \hat{\mathbf{C}}_{\text{cum}}^T(k) \hat{\mathbf{C}}_{\text{cum}}(k) \right)$$

Let us reemphasize that the $(M \times 1)$ and $(M \times N)$ matrices $\hat{\mathbf{C}}_{\text{cum}}(k)$ and $\hat{\mathbf{C}}_{\text{cum}}(k)$

are formed from the elements of $\hat{c}_{\text{cum}}(k; m_1, m_2, m_3)$ and $\hat{c}_{\text{cum}}(k; m_1, m_2, m_3)$ so that

when row i of $\hat{\mathbf{C}}_{\text{cum}}(k)$ is given by $\hat{c}_{\text{cum}}(k; m_1, m_2, m_3)$, row i of $\hat{\mathbf{C}}_{\text{cum}}(k)$ is given

by

$$\begin{bmatrix} \hat{c}_{\text{cum}}(k; m_1, m_2, m_3) & \hat{c}_{\text{cum}}(k; m_1 + 1, m_2 + 1, m_3 + 1) & \cdots \\ \cdots & \hat{c}_{\text{cum}}(k; m_1 + N - 1, m_2 + N - 1, m_3 + N - 1) \end{bmatrix}.$$

We can now summarize the order 4 HOS algorithm.

HOS algorithm for order 4

Given:

N	number of taps or filter length (filter order = $N - 1$)
$L-1$	maximum lag
M	number of points in cumulants = $\frac{L(L+1)(2L+1)}{6}$
λ	forgetting factor, where $0 < \lambda \leq 1$
α	positive constant

muf constant adaptation parameter, where $0 < muf < 1$
 $x(k)$ primary input at iteration k , where $x(k) = 0$ for $k < 1$
 $z(k)$ reference input at iteration k , where $z(k) = 0$ for $k < 1$

Initial Conditions:

$$\mathbf{H}(1) = [h_0(1) \ h_1(1) \ \dots \ h_{N-1}(1)]^T = [0 \ 0 \ \dots \ 0]^T \quad (N \times 1)$$

$$\mathbf{r}_z(0) = [0 \ 0 \ \dots \ 0] \quad (1 \times L)$$

$$\mathbf{R}_z(0) = \begin{bmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{bmatrix} \quad (L \times L)$$

$$\mathbf{R}_{zbig}(0) = \begin{bmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{bmatrix} \quad (L+N-1 \times L+N-1)$$

$$\mathbf{R}_{z=0}(0) = \mathbf{R}_{z=1}(0) = \dots = \mathbf{R}_{z=L-1}(0) = \begin{bmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{bmatrix} \quad (L \times L)$$

$$\mathbf{R}_{z=0}(0) = \mathbf{R}_{z=1}(0) = \dots = \mathbf{R}_{z=L+N-1}(0) = \begin{bmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{bmatrix} \quad (L+N-1 \times L+N-1)$$

For $k = 1, 2, 3, \dots$

$$\mathbf{Z0} = [z(k) \ z(k+1) \ \dots \ z(k+L-1)]^T$$

$$\mathbf{Z0big} = [z(k) \ z(k+1) \ \dots \ z(k+L+N-2)]^T$$

for stationary case

$$\mathbf{r}_z(k) = \frac{k-1}{k} \mathbf{r}_z(k-1) + \frac{1}{k} x(k) \mathbf{Z0}^T$$

$$\mathbf{R}_z(k) = \frac{k-1}{k} \mathbf{R}_z(k-1) + \frac{1}{k} z(k) [\mathbf{Z0} \cdot \mathbf{Z0}^T]$$

$$\mathbf{R}_{zbig}(k) = \frac{k-1}{k} \mathbf{R}_{zbig}(k-1) + \frac{1}{k} z(k) [\mathbf{Z0big} \cdot \mathbf{Z0big}^T]$$

end

for non-stationary case

$$B = \frac{\lambda^{k-1} - \lambda^{-1}}{\lambda^k - \lambda^{-1}}$$

$$A = \frac{1 - \lambda^{-1}}{\lambda^k - \lambda^{-1}}$$

```


$$\mathbf{r}_z(k) = \lambda B \mathbf{r}_z(k-1) + A x(k) \mathbf{Z0}^T$$


$$\mathbf{R}_z(k) = \lambda B \mathbf{R}_z(k-1) + A z(k) [\mathbf{Z0} \cdot \mathbf{Z0}^T]$$


$$\mathbf{R}_{zbig}(k) = \lambda B \mathbf{R}_{zbig}(k-1) + A z(k) [\mathbf{Z0big} \cdot \mathbf{Z0big}^T]$$

end
for  $m = 0, 1, 2, \dots, L-1$ 
  for stationary case
    
$$\mathbf{R}_{zm}(k) = \frac{k-1}{k} \mathbf{R}_{zm}(k-1) + \frac{1}{k} x(k) z(k+m) [\mathbf{Z0} \cdot \mathbf{Z0}^T]$$

  end
  for non-stationary case
    
$$\mathbf{R}_{zm}(k) = \lambda B \mathbf{R}_{zm}(k-1) + A x(k) z(k+m) [\mathbf{Z0} \cdot \mathbf{Z0}^T]$$

  end
  end
  
$$\mathbf{C}_{zm} = \mathbf{R}_{zm} - \mathbf{r}_z[m] \cdot \mathbf{R}_z - \mathbf{r}_z^T \cdot \mathbf{R}_z[:,m]^T - \mathbf{R}_z[:,m] \cdot \mathbf{r}_z$$

end
for  $m = 0, 1, 2, \dots, L+N-2$ 
  for stationary case
    
$$\mathbf{R}_{zm}(k) = \frac{k-1}{k} \mathbf{R}_{zm}(k-1) + \frac{1}{k} z(k) z(k+m) [\mathbf{Z0big} \cdot \mathbf{Z0big}^T]$$

  end
  for non-stationary case
    
$$\mathbf{R}_{zm}(k) = \lambda B \mathbf{R}_{zm}(k-1) + A z(k) z(k+m) [\mathbf{Z0big} \cdot \mathbf{Z0big}^T]$$

  end
  end
  
$$\mathbf{C}_{zm} = \mathbf{R}_{zm} - \mathbf{R}_{zbig}[1,m] \cdot \mathbf{R}_{zbig} - \mathbf{R}_{zbig}[1,:]^T \cdot \mathbf{R}_{zbig}[:,m]^T - \mathbf{R}_{zbig}[:,m] \cdot \mathbf{R}_{zbig}[1,:]$$

end

vector = []
for  $m_1 = 0, 1, 2, \dots, L-1$ 
  for  $m_2 = 0 \dots m_1$ 
    for  $m_3 = 0 \dots m_1$ 
      vector = [vector Czm1[ $m_2, m_3$ ]]
    end
  end
end
end
 $\hat{\mathbf{C}}_{zm} = \mathbf{vector}^T$ 

for  $n = 0 \dots N-1$ 
  vector = []
  for  $m_1 = n \dots n+L-1$ 

```

```

    for  $m_2 = n \dots n + m_1$ 
      for  $m_3 = n \dots n + m_1$ 
         $\mathbf{vector} = [\mathbf{vector} \quad \mathbf{C}_{\underline{\underline{m}}}[m_2, m_3]]$ 
      end
    end
  end
   $\hat{\mathbf{C}}_{\underline{\underline{m}}}[:, n] = \mathbf{vector}^T$ 
end

 $\mathbf{Z} = [z(k) \quad z(k-1) \quad \dots \quad z(k-(N-1))]^T$ 
 $y(k) = \mathbf{H}(k)^T \mathbf{Z}$ 
 $e(k) = x(k) - y(k)$ 
 $\hat{\mathbf{V}} = 2(\hat{\mathbf{C}}_{\underline{\underline{m}}}^T \hat{\mathbf{C}}_{\underline{\underline{m}}} \mathbf{H}(k) - \hat{\mathbf{C}}_{\underline{\underline{m}}}^T \hat{\mathbf{C}}_{\underline{\underline{m}}})$ 
 $\mu = \frac{\mu f}{\alpha + \text{tr}(\mathbf{C}_{\underline{\underline{m}}}^T \mathbf{C}_{\underline{\underline{m}}})}$ 
 $\mathbf{H}(k+1) = \mathbf{H}(k) - \mu \hat{\mathbf{V}}$ 
End

```

In Chapter 7 we will look at the computational complexity of the order 4 HOS algorithm, and compare it to that for the order 2 algorithm and the NLMS algorithm. Then, we will present results of the Adaptive Interference Canceler using the order 4 HOS algorithm on computer generated test signals

Chapter 7

PERFORMANCE OF THE ORDER 4 HOS ALGORITHM

COMPUTATIONAL COMPLEXITY

Before investigating the performance of the order 4 HOS algorithm, we need to address the subject of computational complexity. A good yardstick by which to measure the computational complexity of adaptive algorithms is the number of multiplications per iteration.

One of the greatest assets of the NLMS algorithm is its computational simplicity, as its “instantaneous” gradient estimate is based only on the squared difference between the two output data values, $d(k)$ and $y(k)$. Higher up on the complexity scale, the HOS order 2 algorithm not only computes cross-correlation and autocorrelation function estimates at every iteration, but it also computes one vector inner product, three matrix/vector multiplications, and a matrix/matrix multiplication in the calculation of its “smoothing” gradient. Higher still on the complexity scale is the HOS order 4 algorithm, which, in addition to the aforementioned matrix multiplications, computes fourth order self and cross-cumulant estimates employing vector outer product operations.

As Table 7.1 shows, the improved performance of each algorithm over its predecessor carries a significant computational price tag. The third row in the table shows the number of multiplications per iteration for the HOS order 4 algorithm when coded in the

MATLAB programming language, exactly as it appears in Chapter 6. The fourth row in the table shows the number of multiplications which can be achieved by coding the algorithm in a typical high level language, such as C++ or FORTRAN.

The astute reader will have noticed that in the algorithm as given, the cumulants are computed for all points in the $L \times L \times L$ cubic space, while only the subset consisting of the $(1/6)(L)(L+1)(2L+1)$ points in the domain Γ are actually used in the computation of the gradient. In order to achieve the lower number of multiplications shown in row four of the table, it is necessary to compute the cumulants for only those points in the three-dimensional wedge shaped domain Γ which are actually to be used for the gradient computation. Coding the algorithm in this manner requires the use of nested “for” loops, of up to four levels in depth. In the MATLAB programming language, nested loops of this type execute exceedingly slowly, whereas the matrix operations used to compute the cumulants in the cubic space execute significantly faster, even with the greater number of multiplications. For example, with $N = 12$ and $L = 4$, the matrix based algorithm has approximately twice the number of multiplications per iteration as does the nested for loop based algorithm, but it executes over ten times faster in the MATLAB programming language.

Table 7.1. Computational complexity of the NLMS, HOS order 2, and HOS order 4 algorithms.

Algorithm	Number of multiplications per iteration (stationary case)			
NLMS	$3N+2$	$N=8$ 26	$N=12$ 38	$N=16$ 50
HOS 2	$5+5N+(N^2+3N+4)L$	$N=8$ $L=3$ 321	$N=12$ $L=4$ 801	$N=16$ $L=5$ 1,625
HOS 4 (Matlab)	$5L^3+3(L^2+3L+1)+3(L+N-1)^2+5(L+N-1)^3$ $+(3N+N^2)(1/6)(L)(L+1)(2L+1)+6N$	$N=8$ $L=3$ 6,772	$N=12$ $L=4$ 23,429	$N=16$ $L=5$ 58,764
HOS 4 (C++)	$(1/6)(L)(L+1)(2L+1)(N^2+20N+19)$ $+N^2+2N+2$	$N=8$ $L=3$ 3,484	$N=12$ $L=4$ 12,260	$N=16$ $L=5$ 33,015

TEST RESULTS

We now proceed to demonstrate the effectiveness of the Adaptive Interference Canceler using the order 4 HOS algorithm at mitigating single and multiple stationary narrowband interference both in the absence and presence of uncorrelated measurement noise, and comparing the performance to that of the NLMS algorithm. Then, we will consider the case of non-stationary interference.

SINGLE STATIONARY NARROWBAND INTERFERENCE WITHOUT UNCORRELATED NOISE

Let the single stationary narrowband interference be $I(k) = I_2(k)$, where A_2 takes on different values. The number of taps for both algorithms is $N = 16$, and for the HOS order 4 algorithm $L = 4$. The system error signals for the NLMS and HOS order 4 algorithms are shown in Figure 7.1.

Figure 7.1. System errors for the NLMS and HOS order 4 algorithms with $N = 16$, $L = 4$, $\mu = .075$, $muf = .99$, $\alpha = 2$, $n_p(k) = n_r(k) = 0$, and $I(k) = I_2(k)$. For (a.1) and (b.1), $A_2 = 0.5$. For (a.2) and (b.2), $A_2 = 1.5$. For (a.3) and (b.3), $A_2 = 3$.

With the chosen values of $\mu = .075$ for the NLMS and $muf = .99$ for the HOS order 4, let us compare the rate of convergence and excess error for the two algorithms. In all three cases, the HOS order 4 algorithm converges more rapidly than the NLMS algorithm. Even for the low level interference case, $A_2 = 0.5$, in which the HOS order 4 algorithm converges most slowly, its convergence is still more rapid than that of the NLMS. In the high level interference cases, $A_2 = 1.5$ and $A_2 = 3$, the convergence of the HOS order 4 is extremely rapid (note the duration of the interval before the first noisy burst in panels (b.2) and (b.3)), followed by one noisy burst of excess error roughly equivalent to those characterizing the performance of the NLMS after convergence. Importantly, even

though the HOS order 4 algorithm converges more rapidly than the NLMS in all three cases, its excess error is considerably less than that of the NLMS algorithm.

MULTIPLE STATIONARY NARROWBAND INTERFERENCE WITHOUT UNCORRELATED NOISE

Let us return to the topic of the underdetermined filter length, where we have underestimated the relationship between the reference signal and the interference, and have chosen too small a value for N . In a case such as this where we have the sum of multiple interference signals, each of which is processed through a separate LTI filter, the number of AIC filter taps is underdetermined when $\text{length}\{H(z)\}$ is less than the minimum of $\text{length}\{g_1(z), g_2(z), g_3(z)\}$. For filter lengths greater than this minimum, it is not clear what will be the critical value of N with respect to the lengths of g_1 , g_2 , and g_3 , and/or some possible relationship between them.

For this simulation we used $I(k) = I_1(k) + I_3(k)$ with $A_1 = .75$, $A_3 = 2$, or $I(k) = I_1(k) + I_2(k) + I_3(k)$ with $A_1 = 1$, $A_2 = .5$, $A_3 = .75$. In both cases we used $\mu = .075$, $muf = .99$, $\alpha = 2$, and $L = 4$ for various values of N . The results for $N = 4, 8$, and 12 with $I(k) = I_1(k) + I_3(k)$ appear in Figure 7.2, and the result for $N = 16$ with $I(k) = I_1(k) + I_2(k) + I_3(k)$ appears in Figure 7.3.

The case $N = 4$, in Figure 7.2, is clearly the case where the number of AIC filter taps is underdetermined. In comparing panels (a.1) and (b.1) we can see that the performance of the order 4 HOS algorithm is superior to that of the NLMS algorithm for the case of underestimated filter length. When the composite interference signal is the sum of all

three distinct interference signals, the case $N = 16$ proves to be underdetermined, and the above result is corroborated by Figure 7.3.

Figure 7.2. System error signals of the NLMS and HOS order 4 algorithms for $I(k) = I_1(k) + I_3(k)$ with $A_1 = .75$, $A_3 = 2$, $\mu = .075$, $muf = .99$, $\alpha = 2$, $n_p(k) = n_r(k) = 0$, and $L = 4$.

Figure 7.3. System error signals of the NLMS and HOS order 4 algorithms for interference $I(k) = I_1(k) + I_2(k) + I_3(k)$ with $A_1 = 1$, $A_2 = .5$, $A_3 = .75$, $\mu = .075$, $muf = .99$, $\alpha = 2$, $n_p(k) = n_r(k) = 0$, $N = 16$, and $L = 4$.

We will now add various levels of uncorrelated additive Gaussian noise to the primary and reference inputs, and compare the performance of the Adaptive Interference Canceler using the NLMS and HOS algorithms on noisy signals.

SINGLE STATIONARY NARROWBAND INTERFERENCE WITH UNCORRELATED NOISE

For the single interference case we used $I(k) = I_2(k)$ with $A_2 = 2$, $\mu = .075$, $muf = .99$, $\alpha = 2$, $N = 16$, and $L = 4$ for three different noise levels at the two inputs. The results are shown in Figure 7.4. In Chapter 1 we referred to the uncorrelated to correlated noise-to-noise density ratio, when we were assuming stochastic inputs, and we deduced that the performance of the Adaptive Interference Canceler in the presence of uncorrelated measurement noise would be best when that ratio small. For this case, where the reference signal is a stationary sinusoid of constant amplitude, we can define the uncorrelated to correlated noise-to-interference ratio at the reference input as

$$(N / I)_r = 10 \cdot \log_{10} \left\{ \frac{\text{avg. power in noise}}{\text{avg. power in reference signal}} \right\} = 10 \cdot \log_{10} \left\{ \frac{\sigma^2}{\frac{1}{2} A^2} \right\}$$

where σ^2 is the variance of the noise at the reference input. From Figure 7.4 we see that the order 4 HOS algorithm outperforms the NLMS algorithm in terms of both convergence speed and excess error for all three $(N / I)_r$ cases: -29 dB for (a.1) and (b.1), -26 dB for (a.2) and (b.2), and -23 dB for (a.3) and (b.3). As we expected, the improvement in performance of the HOS algorithm is greatest for the small $(N / I)_r$ cases.

Figure 7.4. System errors for the NLMS and HOS order 4 algorithms for interference $I(k) = I_2(k)$ with $A_2 = 2$, $\mu = .075$, $muf = .99$, $\alpha = 2$, $N = 16$, and $L = 4$. The variance of the uncorrelated additive Gaussian noise at the primary and reference inputs is .0025 for (a.1) and (b.1), .005 for (a.2) and (b.2), and .01 for (a.3) and (b.3).

MULTIPLE STATIONARY NARROWBAND INTERFERENCE WITH UNCORRELATED NOISE

For the simulation of Figure 7.5 we used a multiple interference consisting of $I(k) = I_1(k) + I_3(k)$ with $A_1 = .75$ and $A_2 = 2$ for three different levels of uncorrelated noise. We also used $\mu = .075$, $muf = .99$, $\alpha = 2$, $N = 16$, and $L = 5$. As in Figure 7.4, the HOS algorithm clearly outperforms the NLMS algorithm, particularly in the small $(N / L)_r$ cases.

Figure 7.5. System errors for the NLMS and HOS order 4 algorithms. The interference is $I(k) = I_1(k) + I_3(k)$ with $A_1 = .75$ and $A_2 = 2$. Also, $\mu = .075$, $muf = .99$, $\alpha = 2$, $N = 16$, and $L = 5$. The variance of the uncorrelated additive Gaussian noise at the primary and reference inputs is .0025 for (a.1) and (b.1), .005 for (a.2) and (b.2), and .01 for (a.3) and (b.3).

In Chapter 8, we will present the results from simulations using non-stationary interference signals in the presence of uncorrelated noise.

Chapter 8

THE ORDER 4 HOS ALGORITHM AND NON-STATIONARY INTERFERENCE

AM INTERFERENCE WITH NOISE

For the AM interference in this case we used $I_3(k)$, where $w_3(k)$ had a time varying amplitude A_3 . The filter length was $N = 8$, and $L = 3$ for the HOS order 4 algorithm. The forgetting factor was $\lambda = 0.99$. The system errors for both algorithms, with the uncorrelated noise variance equal to 0.001 at both inputs, are shown in Figure 8.1. The performance of the HOS order 4 algorithm is characterized by significantly more rapid convergence, with smaller excess error, than the NLMS algorithm. The effectiveness of both algorithms at tracking the time varying amplitude of the interference is demonstrated by the figure, but even in the presence of noise, the HOS order 4 algorithm is able to track while producing a smaller excess error.

Figure 8.1. System errors for the NLMS and HOS order 4 algorithms for AM interference $I_3(k)$ with time varying amplitude A_3 , and $N = 8$, $L = 3$, $\alpha = 2$, $\mu = .075$, $muf = 0.99$, $\lambda = 0.99$. The variance of the uncorrelated noises is 0.001.

FM INTERFERENCE WITH NOISE

For the FM interference we used a single unit amplitude sinusoid with time varying frequency for the reference signal, and generated the interference through $g_3(k)$. We used a value of $\lambda = 0.99$ for the forgetting factor and a filter length of $N = 10$ for both algorithms. For the HOS order 4 algorithm, we used $L = 4$. The system errors for increasing levels of noise are shown in Figure 8.2. As in the AM case, the HOS order 4 algorithm converges more rapidly than the NLMS with smaller excess error.

Figure 8.2. System errors for the NLMS and HOS order 4 algorithms for FM interference $I_3(k)$ with time varying frequency F_3T and $N = 10$, $L = 4$, $\alpha = 2$, $\mu = .075$, $muf = 0.99$, $\lambda = 0.99$. The variance of the uncorrelated noises are 0.001 (a.1) and (b.1), and 0.005 (a.2) and (b.2).

STATIONARY PLUS NONSTATIONARY INTERFERENCE WITH NOISE

The system outputs for one stationary sinusoid plus one FM sinusoid appear in Figure 8.3. For both algorithms $N = 12$ and for the HOS order 4 algorithm $L = 5$ and $\lambda = 0.99$. Again, as we saw for the AM and FM interferences, the HOS order 4 algorithm converges more rapidly with smaller excess error than the NLMS algorithm.

Figure 8.3. System errors for the NLMS and HOS order 4 algorithms for interference $I(k) = I_1(k) + I_3(k)$ with $A_1 = 0.5$, $A_3 = 0.75$ and time varying frequency F_3T . Also, $N = 12$, $L = 5$, $\alpha = 2$, $\mu = .075$, $muf = 0.99$, $\lambda = 0.99$. The variance of the uncorrelated noises are 0.001 (a.1) and (b.1), 0.0025 (a.2) and (b.2), 0.005 (a.3) and (b.3).

LINEAR FM CHIRP WITH NOISE

The reference signal for Figure 8.4 was a unit amplitude linear FM chirp signal, and the interference was generated through $g_3(k)$. Again we used a filter length $N = 12$ for both algorithms, and for the HOS order 4, we used $L = 5$ and $\lambda = 0.975$. The HOS order 4 algorithm shows a larger excess error during convergence, but the two algorithms

converge at the same rate, and demonstrate equivalent levels of excess error. The frequency of the linear FM chirp interference signal is changing much more rapidly than the frequency of the FM interference of Figure 8.2, and we see that both algorithms are well suited to track this rapidly changing frequency. To reiterate an important point we made earlier, the NLMS algorithm proves to be especially well suited to tracking time varying interference, and its performance, even in the presence of noise, approaches that of the HOS algorithm in some instances.

In Chapter 9 we will make some direct comparisons between the performance of the order 2 and order 4 HOS algorithms.

Figure 8.4. System errors for the NLMS and HOS order 4 algorithms for interference consisting of a unit amplitude linear FM chirp reference signal convolved with $g_3(k)$, and $N = 12$, $L = 5$, $\alpha = 2$, $\mu = .075$, $muf = 0.99$, $\lambda = 0.975$. The variance of the uncorrelated noises are 0.001 (a.1) and (b.1), 0.0025 (a.2) and (b.2), 0.005 (a.3) and (b.3).

Chapter 9

COMPARING THE ORDER 4 AND ORDER 2 HOS ALGORITHMS

STATIONARY INTERFERENCE

Let us compare the performance of the HOS order 4 algorithm with that of the HOS order 2 algorithm on identical signals, first with low level noise at both the primary and reference inputs, then with the noise level at the reference input increased. Since the HOS order 2 algorithm is a member of the HOS algorithm family, the two algorithms should perform in a similar manner for the low noise case. Where the order 2 algorithm's tap weight update equation is directly affected by the autocorrelation function of the noise at the reference input, the order 4 algorithm's update equation should reject it asymptotically as the number of data points increases. The respective system errors for the two cases are shown in Figure 9.1. As expected, from panels (a.1) and (b.1) we can see that the two algorithms perform in a similar manner for the low noise case. In the case of increased noise power at the reference input, panels (a.2) and (b.2) demonstrate that the HOS order 4 algorithm outperforms the HOS order 2 algorithm at canceling the interference. Notice, however, that both algorithms are effected by the increased noise level, as we expect. In both cases the interference was $I(k) = I_1(k)$ with $A_1 = 0.4$, $N = 8$, and $L = 5$.

Figure 9.1. System errors for the HOS order 2 and HOS order 4 algorithms, with $N = 8$, $L = 5$, $\mu_{us} = .99$, $\mu_{uf} = .99$, $\alpha = 2$, $I(k) = I_1(k)$, and $A_1 = 0.4$. For (a.1) and (b.1) the noise variances at the primary and reference inputs are .001. For (a.2) and (b.2) the noise variances at the primary and reference inputs are .001 and .01 respectively.

NON-STATIONARY AM INTERFERENCE

For the AM interference we used $I(k) = I_2(k)$ with time varying A_2 . We also used $N = 8$, $L = 3$, $\mu_{us} = .99$, $\mu_{uf} = .99$, $\alpha = 2$, and $\lambda = 0.99$. The system errors for two cases, with equal noise levels at the primary and reference inputs, are shown in Figure 9.2. As we see in the figure, the order 4 HOS algorithm produces a slightly smaller excess error than the order 2 algorithm for both noise levels, particularly for the higher level noise case (panels (a.2) and (b.2)).

Figure 9.2. System errors for the HOS order 2 and HOS order 4 algorithms for interference $I(k) = I_2(k)$, and time varying A_2 . Also, $N = 8$, $L = 3$, $\mu_{us} = .99$, $\mu_{uf} = .99$, $\alpha = 2$, and $\lambda = 0.99$. For (a.1) and (b.1) the noise variance at the primary and reference inputs is 0.001. For (a.2) and (b.2) the noise variance at the primary and reference inputs is 0.005.

NON-STATIONARY FM INTERFERENCE

For the FM interference we used $I(k) = I_3(k)$ with time varying $F_3 T$. In this simulation, we used $N = 10$, $L = 6$, $\mu_{us} = .99$, $\mu_{uf} = .99$, $\alpha = 2$, and $\lambda = 0.99$. The system errors for uncorrelated noise with variance 0.001 at both inputs are shown in panels (a.1) and (b.1) of Figure 9.3. For the case shown in panels (a.2) and (b.2) the noise variance at the reference input was increased to 0.01. As the figure demonstrates, the performance of the two algorithms is very similar for the FM interference. We can see a slight improvement

in excess error for the order 4 algorithm over the order 2 for both cases, most evident after the conclusion of the noisy burst around iteration 250.

As we saw for both non-stationary cases, AM and FM interference, the improvement in excess error of the order 4 algorithm over the order 2 algorithm is relatively small, while both algorithms converge at essentially equal rates. For stationary interference, however, the difference in performance between the algorithms is more pronounced. Possibly, the introduction of the forgetting factor λ compromises the noise suppressing ability of the order 4 algorithm. We must note that a large number of data are necessary before fourth order cumulants begin to effectively suppress a Gaussian process [5], and perhaps by introducing the forgetting factor λ we are not allowing a sufficient number of significant data to accumulate for this suppression to occur.

In Chapter 10, we will summarize our work and our results, and draw some final conclusions.

Figure 9.3. System errors for the HOS order 2 and HOS order 4 algorithms for interference $I(k) = I_3(k)$ with time varying F_3T , and $N = 10$, $L = 6$, $\mu_{us} = 0.99$, $\mu_{uf} = 0.99$, $\alpha = 2$, and $\lambda = 0.99$. For (a.1) and (b.1) the noise variance at the primary and reference inputs was 0.001. For (a.2) and (b.2) the noise variance at the reference input was increased to 0.01.

Chapter 10

SUMMARY AND CONCLUSIONS

SUMMARY

In this thesis, we first analyzed the steady-state behavior of the ideal Adaptive Interference Canceler in terms of Wiener Filter theory. We showed that for an infinitely long two sided adaptive transversal filter where the uncorrelated noise powers at both the primary and reference inputs were zero the Adaptive Interference Canceler would provide perfect signal reconstruction. We also showed that for any Adaptive Interference Canceler, the uncorrelated noise at the primary input remains uncanceled and appears with the signal at the system output.

Next, we derived the standard LMS algorithm as a means of updating the adaptive filter's N tap weight coefficients in order to reach the minimum point on the N dimensional quadratic error surface. This was accomplished by first taking the gradient of the instantaneous squared error, $e^2(k)$, with respect to the filter tap weight vector $\mathbf{H}$, and then taking a small step proportional to the negative of that gradient. We then modified the LMS algorithm which suffers from the problem of *gradient noise amplification*, because the step taken is directly proportional to the tap input vector $\mathbf{Z}$. In the resulting NLMS algorithm, instead of being proportional to the tap input vector $\mathbf{Z}$, the correction

applied to the tap weight coefficient vector is normalized by the squared Euclidean norm of $\mathbf{Z}$.

Next we proposed an alternative to the NLMS algorithm based on higher-order statistics, and proceeded to familiarize ourselves with the definitions and important properties of moments and cumulants. Two of these properties, in particular, would later motivate our development of the fourth order algorithm. Specifically, for all orders greater than two, the cumulants of a Gaussian process are identically zero. Also, for processes whose probability density functions are symmetric, all odd order cumulants for n greater than two are identically zero.

We then developed the Adaptive Interference Canceller HOS algorithm, which also employs a gradient descent methodology. In this case, however, the error is defined as the sum of the squared errors between the two order n cumulants $c_n^{x \dots x}(m_1, m_2, \dots, m_{n-1})$ and $c_n^{y \dots y}(m_1, m_2, \dots, m_{n-1})$. We then showed how the moment functions of all orders up to n , knowledge of which is required for the calculation of cumulants of order n , can be calculated recursively. For the non-stationary recursive relationships, we introduced a forgetting factor λ , the choice of which determines the number of recent data points which are significant in the recursive moment function computation.

Next, we implemented the HOS algorithm for order $n = 2$, and compared its performance to that of the NLMS algorithm for single and multiple stationary narrowband interference, and for non-stationary interference, in the absence of uncorrelated measurement noise. We showed that while the performance of the NLMS algorithm is

highly sensitive to the choice of constant adaptation parameter μ , the performance of the HOS algorithm is essentially unaffected for any choice of its constant adaptation parameter μ in the same range. We showed that the HOS algorithm dramatically outperforms the NLMS algorithm when we have underestimated the relationship between the reference signal and the interference, and have used too few taps in the adaptive filter. We also showed that the HOS algorithm converges more quickly and to a smaller excess error than the NLMS algorithm, both for the case of stochastic and deterministic input signals $s(k)$. We then showed why the tap weight update equation of the order 2 HOS algorithm is not independent of uncorrelated Gaussian measurement noise, but is affected by the auto-correlation of the noise at the reference input. Finally, by noting that the fourth order cumulants of Gaussian processes are zero, while the fourth order cumulants of periodic interference are non-zero, we justified developing the order 4 HOS algorithm so that we might effectively cancel this interference in the presence of noise.

We proceeded to develop the HOS algorithm for order $n = 4$, and compared its performance to that of the NLMS algorithm, once again, for single and multiple stationary narrowband, and non-stationary interference, but this time both in the absence and in the presence of uncorrelated measurement noise. We saw that, as expected, the performance of both algorithms degrades with the introduction of noise, as the output of an adaptive interference canceler in the presence of noise will contain uncanceled noise in general. However, when the ratio of the uncorrelated noise power to the interference power was small, we saw that the adaptive interference canceler using the HOS algorithm dramatically improves upon the performance of the NLMS based canceler. We noted that

a distinct advantage of the NLMS algorithm is the fact that the algorithm is the same for both stationary and non-stationary interference, thus relieving the user of the burden of choosing the proper value for the forgetting factor λ . We also saw that the NLMS algorithm is particularly well suited for tracking certain non-stationary interferences in the presence of noise, and that its performance in some cases approached that of the HOS algorithm. We also considered the computational complexity of the NLMS and HOS algorithms, and saw that in order to benefit from the improvements in performance of the HOS algorithm over the NLMS, we must pay a significant computational price.

Finally, we made some direct comparisons between the performance of the order 2 and order 4 HOS algorithms in the presence of noise. We showed that the order 4 algorithm is indeed less affected by an increase in noise at the reference input than is the order 2 algorithm, as our earlier theoretical analysis had indicated. We saw the greatest benefit to be in the case of stationary interference, where all past data values are equally weighted. In the case of non-stationary interference, where only a number of the most recent data values are significant, the improvement of the order 4 algorithm over the order 2 was seen to be marginal. We speculated that perhaps by introducing the forgetting factor λ , the noise suppression capability of the order 4 algorithm was compromised.

CONCLUSIONS

The HOS algorithm based Adaptive Interference Canceler offers considerable performance improvements over the NLMS algorithm based canceler at the cost of an increase in computational burden. However, the algorithm as presented in this thesis

assumes strictly sequential processing, and does not take into consideration any of the structural properties of the algorithm that would lend themselves to the integration of “fast” algorithm techniques which might exploit parallelism, shift properties, efficient VLSI structures, etc.

In some applications, the increase in computational complexity from order $3N+2$ for the NLMS algorithm to (approximately) order N^2L for the HOS order 2 algorithm is probably a price worth paying in order to enjoy all the benefits offered by the HOS algorithm over the NLMS. However, based on the relatively minor improvement in performance demonstrated by the HOS order 4 algorithm over the HOS order 2, even in the presence of noise, it is difficult to conceive of applications where the accompanying increase in computational complexity from order (approximately) N^2L to order (approximately) $(1/3)(L^3)(N^2 + 20N) + N^2$ would be an acceptable tradeoff.

There might be other classes of signals, different from those that were used in our simulations, for which the improvement in performance of the order 4 HOS algorithm over its order 2 counterpart in the presence of noise is much more acute. If so, then such a jump in computational complexity might be worthwhile, especially if a “fast” algorithmic structure is developed. This is an area which is open to further research. Also, the performance of the algorithm is expected to be different for different choices of the domain Γ . As such, there could be an optimal domain Γ in which to calculate the cumulants, for a particular class of signals. If this is the case, then the noise suppression capability of the order 4 algorithm over its order 2 counterpart might be greatly enhanced if the algorithm operated in that optimal domain Γ .

BIBLIOGRAPHY

- [1] Nikias, C. L., & Petropolu, A. P. (1993). Higher-order spectra analysis: A nonlinear signal processing framework. Englewood Cliffs, NJ: Prentice-Hall.
- [2] Beyer, W. H. (1981). CRC Standard Mathematical Tables (26th ed.). Boca Raton, FL: CRC Press.
- [3] Mendel, J. M. (1991). Tutorial on higher-order statistics (spectra) in signal processing and system theory: Theoretical results and some applications. Proceedings of the IEEE, 79, 278-305.
- [4] Nikias, C. L., & Mendel, J. M. (1993, July). Signal processing with higher-order spectra. IEEE Signal Processing Magazine, 10-37.
- [5] Shin, D. C., & Nikias, C. L. (1994). Adaptive interference canceler for narrowband and wideband interferences using higher order statistics. IEEE Transactions on Signal Processing, 42, 2715-2728.
- [6] Chiang, H.-H., & Nikias, C. L. (1990). Adaptive deconvolution and identification of nonminimum phase FIR systems based on cumulants. IEEE Transactions on Automatic Control, 35, 36-47.
- [7] Chiang, H.-H., & Nikias, C. L. (1990). A new method for adaptive time delay estimation for non-Gaussian signals. IEEE Transactions on Acoustics, Speech, and Signal Processing, 38, 209-219.
- [8] Nikias, C. L., & Raghuveer, M. R. (1987). Bispectrum estimation: A digital signal processing framework. Proceedings of the IEEE, 75, 869-891.
- [9] Rangoussi, M., & Carayannis, G. (1994). Higher order statistics based Gaussianity test applied to on-line speech processing. Proceedings of the 28th Asilomar Conference on Signals, Systems, and Computers, Vol. 1, 303-307.
- [10] Proakis, J. G., & Manolakis, D. G. (1996). Digital signal processing: Principles, algorithms, and applications. Englewood Cliffs, NJ: Prentice-Hall.
- [11] Papoulis, A. (1991). Probability, random variables, and stochastic processes (3rd ed.). New York: McGraw-Hill.

- [12] Mulgrew, B., & Cowan, C. F. N. (1988). Adaptive filters and equalizers. Boston: Kluwer Academic Publishers.
- [13] Milner, S.H., Cryan, R.A., & Senior, J.M. (1994). In-service non-intrusive characterization of speech channels. Proceedings of the International Symposium on Information Theory and its Applications, Sydney, Australia, 721-725.
- [14] Treichler, J. R., Johnson, C. R., Jr. & Larimore, M. G. (1987). Theory and design of adaptive filters. New York: John Wiley & Sons.
- [15] Haykin, S. (1996). Adaptive filter theory. Englewood Cliffs, NJ: Prentice-Hall.
- [16] Widrow, B., & Stearns, S. D. (1985). Adaptive signal processing. Englewood Cliffs, NJ: Prentice-Hall.

APPENDIX A

MATLAB 4.0 CODE

AIC ORDER 2 HOS ALGORITHM

```
%hos2.m
KK=length(s);
e_hos2=x;%start e(k) as (row vector) v'(k) [uncorrupted]
y_hos2=zeros(1,KK);%initial value for row vector y(k)
mus=.99;
alpha=2;%small positive constant
lambda=.995;%forgetting factor
N=12;%filter order=N-1, filter length=N
L=4;
M=L;%M=# of points in cumulants
H=zeros(N,1);%initialize (column) weight vector
Rxx=zeros(M,1);
Cxx=zeros(M,1);
Rzz=zeros(M+N-1,1);
Czz=zeros(M,N);
for k=N:KK-(M+N-2)
    kk=k-(N-1);
    Z0=z(k+1:k+(M-1))';%Z0 is (M,1) vector
    Z0big=z(k+1:k+(M+N-2))';%Z0big is (M+N-1,1) vector
    Bscale=((lambda^(kk-1))-(lambda^(-1)))/((lambda^kk)-(lambda^(-1)));
    Ascale=(1-(lambda^(-1)))/((lambda^kk)-(lambda^(-1)));
    %Rxx=(((kk-1)/kk)*Rxx)+((1/kk)*(x(k)*Z0));
    Rxx=lambda*Bscale*Rxx+Ascale*x(k)*Z0;
    %Rzz=(((kk-1)/kk)*Rzz)+((1/kk)*(z(k)*Z0big));
    Rzz=lambda*Bscale*Rzz+Ascale*z(k)*Z0big;
    Cxx=Rxx;
    for m=0:N-1
        Czz(:,m+1)=Rzz(m+1:m+M);
    end
    Z=z(k:-1:k-(N-1))';%FIR signal vector
    y_hos2(k)=H'*Z;%vector inner product is the filter
    e_hos2(k)=x(k)-y_hos2(k);%error=primary-reference
```

```

delta=2*(Czz'*Czz*H-Czz'*Cxx);%update delta gradient column vector
muofk=mus/(alpha+trace(Czz'*Czz));%update adaptation time factor
H=H-muofk*delta;%update tap weight column vector
end

```

```

errhos2=s-e_hos2;
%i=1:3000;
subplot(324)
plot(errhos2(N:N-1+600),'k')
axis([1 600 -1 1])
title(['(b.2) HOS 2 var{nr,np}=' num2str(sigmanr^2)])

```

AIC ORDER 4 HOS ALGORITHM

```

%hos4.m
KK=length(s);
N=8;%filter length=N, filter order=N-1
L=3;%choose L so that M>N
M=L*(L+1)*(2*L+1)/6;%where M=L(L+1)(2L+1)/6 = # points in cumulants
e4=x;
y=zeros(1,KK);%initial value for row vector y(k)
muf=.99;%constant adaptation parameter
alpha=2;%small positive constant
lambda=.999;%forgetting factor
H=zeros(N,1);%initialize (column) tap weight vector

%initialize rxz and Rzz
rxz=zeros(1,L);%rxz is a row vector
Rzz=zeros(L,L);
Rzzbig=zeros(L+N-1,L+N-1);
Cxzzz=zeros(M,1);
Czzzz=zeros(M,N);

%initialize Rxzz0...RxzzL-1 and Cxzz0...CxzzL-1
for m1=0:L-1
    lag1=int2str(m1);
    com=['Rxzz' lag1 '=zeros(L,L);'];
    eval(com);
    com=['Cxzz' lag1 '=zeros(L,L);'];
    eval(com);
end

```



```

%initialize Rzzz0...RzzzL+N-2, dRzzz0...dRzzzL+N-2, and Czzz0...CzzzL+N-2
for m1=0:L+N-2
    lag1=int2str(m1);
    com=['Rzzz' lag1 '=zeros(L+N-1,L+N-1);'];
    eval(com)
    com=['dRzzz' lag1 '=zeros(L+N-1,L+N-1);'];
    eval(com)
    com=['Czzz' lag1 '=zeros(L+N-1,L+N-1);'];
    eval(com)
end

for k=N:KK-(L+N-2)
    kk=k-(N-1);
    Z0=z(k+1:k+(L-1));%column vectors
    Z0mat=Z0*Z0';%build the Z0 outer product autocorr. matrix
    Bscale=((lambda^(kk-1))-(lambda^(-1)))/((lambda^kk)-(lambda^(-1)));
    Ascale=(1-(lambda^(-1)))/((lambda^kk)-(lambda^(-1)));
    rxz=lambda*Bscale*rxz+Ascale*(x(k)*Z0');
    %rxz=((kk-1)/kk)*rxz+(1/kk)*x(k)*Z0';
    Rzz=lambda*Bscale*Rzz+Ascale*Z0mat;
    %Rzz=((kk-1)/kk)*Rzz+(1/kk)*Z0mat;
    ZLN=z(k+1:k+(L+N-2));
    ZLNmat=ZLN*ZLN';
    Rzzbig=lambda*Bscale*Rzzbig+Ascale*ZLNmat;
    %Rzzbig=((kk-1)/kk)*Rzzbig+(1/kk)*ZLNmat;

    %Compute Rxzz0...RxzzL-1 and Cxzz0...CxzzL-1
    for m1=0:L-1
        em1=int2str(m1);
        com=['Rxzz' em1 '=lambda*Bscale*Rxzz' em1 '+Ascale*x(k)*z(k+m1)*Z0mat;'];
        %com=['Rxzz' em1 '=((kk-1)/kk)*Rxzz' em1 '+ (1/kk)*x(k)*z(k+m1)*Z0mat;'];
        eval(com)
        com1=['Cxzz' em1 '=Rxzz' em1 '-(rxz"*Rzz(:,m1+1))-(Rzz(:,m1+1)*rxz)-
        (rxz(m1+1)*Rzz);'];
        eval(com1)
    end
    %Compute Rzzz0...RzzzL+N-2 and Czzz0...CzzzL+N-2 using matrix algebra
    for m1=0:L+N-2
        lag1=int2str(m1);
        com=['dRzzz' lag1 '=z(k)*z(k+m1)*ZLNmat;'];
        eval(com)
        com=['Rzzz' lag1 '=lambda*Bscale*Rzzz' lag1 '+Ascale*dRzzz' lag1 ';'];
        %com=['Rzzz' lag1 '=((kk-1)/kk)*Rzzz' lag1 '+ (1/kk)*dRzzz' lag1 ';'];

```

```

eval(com)
com1=['Czzz'      lag1      '=Rzzz'      lag1      '-(Rzzbig(1,:)"*Rzzbig(:,m1+1))"-
(Rzzbig(:,m1+1)*Rzzbig(1,:))-(Rzzbig(1,m1+1)*Rzzbig);'];
eval(com1)
end

%Construct column vector Cxzzz from the elements of Cxzz0...CxzzL-1
%using matlab array ops {builds the column vector columnwise from arrays}
Cxzzz=[];
for m1=0:L-1
    em1=int2str(m1);
    com=['Cxzzz=[Cxzzz reshape(Cxzz' em1 '(1:m1+1,1:m1+1),1,(m1+1)^2)];'];
    eval(com)
end
Cxzzz=Cxzzz';

%Construct M x N matrix Czzzz from the elements of Czzz0...CzzzL+N-2
%using matlab array ops
for l=0:N-1
    tempvec=[];
    for m1=0:L-1
        lag1=int2str(m1+1);
        com=['tempvec=[tempvec      reshape(Czzz'      lag1
'(l+1:l+m1+1,l+1:l+m1+1),1,(m1+1)^2)];'];
        eval(com)
    end
    Czzzz(:,l+1)=tempvec';%construct Czzzz matrix column by column
end

Z=z(k:-1:k-(N-1))';%build the signal (column) vector for FIR
y(k)=H'*Z;%y=output of linear predictor (tuned filter)
e4(k)=x(k)-y(k);%e=input-predictable part (notch)
delta=2*(Czzzz'*Czzzz'*H-Czzzz'*Cxzzz);%update delta gradient column vector
mu=muf/(alpha+trace(Czzzz'*Czzzz));%update adaptation time factor
H=H-mu*delta;%update tap weight column vector
end%end of main for k loop

err4=s-e4;
subplot(326)
plot(err4(N:N-1+600),'k'),axis([1 600 -1 1])
title(['(b.2) HOS 4 var{nr}=' num2str(sigmanr^2)])

```

AIC NLMS ALGORITHM

```
%aiclms.m
KK=length(s);
e_nlms=x;
mu_nlms=.075;
alpha=2;
N=12;%filter order=N-1, filter length=N
W=zeros(N,1);%initialize (column) weight vector
for k=N:KK
    dk=x(k);%the new sample at primary input
    Z=z(k:-1:k-(N-1));%Z is column vector of recent samples
    y_nlms(k)=W'*Z;%vector inner product is the filter
    e_nlms(k)=dk-y_nlms(k);%error=primary-reference
    W=W+(mu_nlms/(alpha+(Z'*Z)))*e_nlms(k)*Z;%NLMS update
end

errnlms=s-e_nlms;
subplot(323)
plot(errnlms(N:N-1+600),'k')
axis([1 600 -1 1])
title(['(a.2) NLMS var{nr,np}=' num2str(sigmamn^2)])
```

TEST SIGNAL GENERATOR

```
%makesigs.m
s=[];
k=0;
for pieces=1:79
    %state=sign(rand-1/2);
    for sample=1:8
        s(k+1)=sign(rand-1/2)*cos(2*pi*.43*k);
        %s(k+1)=state*cos(2*pi*.43*k);
        k=k+1;
    end
end

%generate the reference signals w1(k), w2(k), w3(k)
k=0:length(s)-1;
A1=.65;
A2=.5;
A3=.75;
```

```

f1T=.1;
f2T=.25;
f3T=.3;
phi1=(2*pi*rand)-pi;
phi2=(2*pi*rand)-pi;
phi3=(2*pi*rand)-pi;
g1=[1 .1 -.3 .5 .25];
g2=[1 .5 -.1 -.75 .41 .21 .12 -.05 .01 -.2 -.15];
g3=[1 -.2 .2 .7 -.65 .3 -.25];

sigmanp=sqrt(.005);%variance of np(k)
sigmanr=sqrt(.005);%variance of nr(k)
np=sigmanp*randn(size(s));
nr=sigmanr*randn(size(s));

w1=A1*sin(2*pi*f1T*k+phi1);
w2=A2*sin(2*pi*f2T*k+phi2);
w3=A3*sin(2*pi*f3T*k+phi3);
I1=filter(g1,1,w1);%generate interference I1(k)
I2=filter(g2,1,w2);%generate interference I2(k)
I3=filter(g3,1,w3);%generate interference I3(k)
w=w1+w3;%generate the composite reference signal
I=I1+I2;%generate the composite interference signal

%generate the primary and reference sensor inputs x(k) and z(k) (row vectors)
x=s+I+np;
z=w+nr;

```