

AD-A092 734

TUFTS UNIV MEDFORD MASS DEPT OF ELECTRICAL ENGINEERING

F/6 9/5

ASPECTS OF SIGNAL PROCESSING FOR ARRAY ANTENNAS. (U)

SEP 80 D PREIS

AFOSR-80-0110

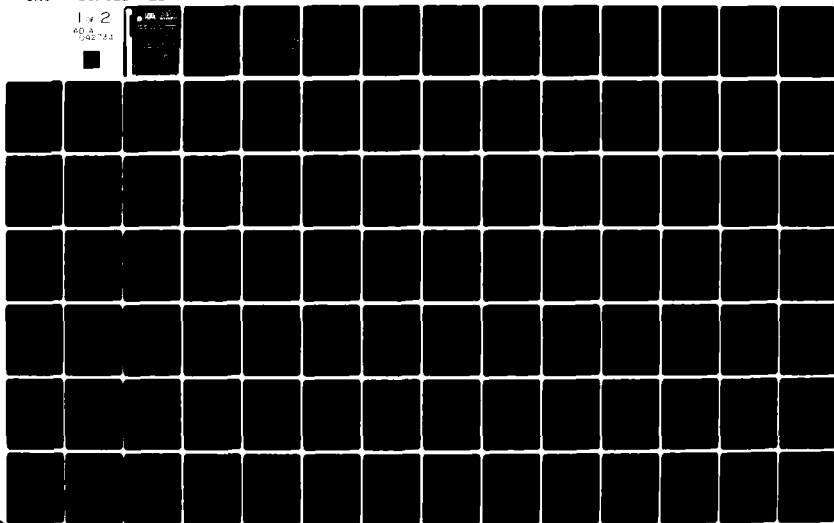
UNCLASSIFIED

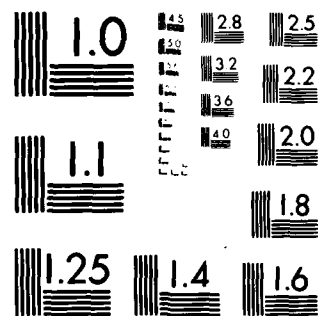
EE/TR-1980-1

AFOSR-TR-80-1242

NL

1 of 2
AD-A
042-734





MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU OF STANDARDS-1963-A

AD A092734

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM	
1. REPORT NUMBER AFOSR-TR-80-1242	2. GOVT ACCESSION NO. AD A692 734	3. RECIPIENT'S CATALOG NUMBER 9	
4. TITLE (and Subtitle) ASPECTS OF SIGNAL PROCESSING FOR ARRAY ANTENNAS.		5. TYPE OF REPORT & PERIOD COVERED Final Report 15 Jan 80	
6. AUTHOR(s) D. Preis		7. PERFORMING ORG. REPORT NUMBER AFOSR-80-0110	
8. CONTRACT OR GRANT NUMBER AFOSR-80-0110		9. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBER 61102F	
10. CONTROLLING OFFICE NAME AND ADDRESS Tufts University, Electrical Engineering Dept. Medford, MA 02155		11. REPORT DATE 30 Sep 1980	
12. CONTROLLING OFFICE NAME AND ADDRESS AFOSR Bolling AFB, D.C., 20332		13. NUMBER OF PAGES 155	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified	
15. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	
16. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)			
17. SUPPLEMENTARY NOTES Sections of this report were published in ICASSP-80 Proceedings, <u>Electronics Letters</u> (Dec. 1980), and presented at the 1980 L'Aquila DSP Workshop.			
18. KEY WORDS (Continue on reverse side if necessary and identify by block number) Equalization filters, equalizers, time domain, transversal filters, minimax, algorithms, Remez, computer-aided design, array element filters, signal processing, deconvolution.			
19. ABSTRACT (Continue on reverse side if necessary and identify by block number) Deconvolution algorithms are given for the computer-aided design of transversal-filter equalizers. These filters can be used to filter signals from individual elements of a sensing array or for array equalization. Minimax design procedures are developed for both continuous-time and discrete-time cases. Also, least-square error and envelope-constrained designs are considered. Essential design equations, numerous computed examples, mathematical details, and several computer codes are included.			

TABLE OF CONTENTS

<u>Section</u>	<u>Page</u>
I. Synopsis	I-1
A. Background	I-1
B. Objectives	I-2
C. Status of Research	I-2
D. Publications	I-3
E. Professional Personnel	I-4
II. Minimax Time-Domain Deconvolution for Transversal Filter Equalizers	II-1
III. Minimax Equalizers: Performance and Adjustment Algorithms	III-1
IV. Chebyshev Time-Domain Deconvolution for Transversal Filter Equalizers	IV-1
V. Three Algorithms for the Design of Transversal Filter Equalizers	V-1
VI. Minimax Equalization for Transversal Filters	VI-1

DTIC
ELECTE
DEC 10 1980

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFSC)
NOTICE OF TRANSMITTAL TO DDC
 This technical report has been reviewed and is
 approved for public release IAW AFR 190-12 (7b).
 Distribution is unlimited.
A. D. BLOOM
 Technical Information Officer

AIR I
 0011
 This
 apper
 Distr
 A. D.
 Techni

(C)
 is
 (7b).

Officer

I. Synopsis

As recommended in the brochure, "Administration of U.S. Air Force Grants and Cooperative Agreements for Basic Research," this report provides, in a single document, a permanent record of the progress and significant accomplishments achieved in the performance of the research effort. This first section contains summaries of research background, research objectives, status of research, publications and professional personnel associated with the research effort. Sections II, III, IV and V are technical papers resulting from the research effort and Section VI is a Master of Science thesis, supervised by the principal investigator, which contains mathematical details and several computer codes related to the four technical papers contained in this report.

A. Background

The research results reported here are direct extensions of earlier research conducted by the principal investigator under the auspices of the 1979 USAF-SCEEE Summer Faculty Research Program sponsored by AFOSR under contract No. F49620-79-0038. The final report summarizing the summer research effort, dated 17 August 1979 and authored by the principal investigator, is entitled, "Adaptive Signal Processing for Array Antennas." That research was performed at the Electronics Systems Division/Hanscom Air Force Base in collaboration with USAF Research Colleague, Dr. Donald B. Brick (Technical Director, Deputy for Development Plans, ESD-XR, (617)-271-3832).

B. Objectives

Virtually all research and development in the area of adaptive arrays has dealt with signal-to-noise ratio improvement by minimization of directional interference and ambient noise using adaptive signal processing. Interference of this nature is not correlated with the desired signal and, as such, can be called "signal-uncorrelated" interference. Another equally important mechanism of signal corruption is the linear distortion of desired signals caused by reflections, scattering, and multipath. In wideband digital communication systems, for example, this introduces intersymbol interference, detection errors and, consequently, significant degradation in performance. Each array element or each array beam is connected to an adjustable filter or equalizer (a transversal filter or tapped delay line) and each individual filter must be adjusted in a systematic way to minimize signal-correlated interference. The research effort reported here is concerned with the specific problem of equalizer adjustment algorithms and their performance. The equalization problem is formulated in the time domain as an integral equation of convolution which is solved in the minimax sense using the Remez algorithm. The discrete-time case is also considered and solved via the Ascent algorithm. Least-square-error and envelope-constrained algorithms also are included.

C. Status of Research

The fundamental research problem is equalizer design in the time domain. The minimax (Chebyshev) error criterion was chosen so that

Accession For	
NTIS GRA&I	
DTIC TAB	
Unannounced	
Justification	
By	
Distribution/	
Availability Co	
Dist	Avail and/o Special
A	

equalization errors would be both minimized and bounded in the time-domain. Since frequency-domain techniques are not used, hardware and software complexity and cost may be reduced and filter adjustment speeds increased. The studies reported here indicate that minimax time-domain design of an equalizer is both reasonable and practical. Experimental results from computer-aided designs of minimax equalizers illustrate several aspects of the performance capabilities and computational sensitivities of the Remez algorithm. Of particular interest are the trades between filter length, total computation time, and accuracy. Several representative examples are given. Equations describing all filter adjustment algorithms are included. Computer codes for the Remez algorithm and Ascent algorithm are listed.

D. Technical Publications Related to Research

1. D. Preis and C. Bunks, "Minimax Time-Domain Deconvolution for Transversal Filter Equalizers," ICASSP-80 Proceedings, pp. 943-946, IEEE International Conference on Acoustics, Speech and Signal Processing, Denver, Colorado, April 1980
2. D. Preis, and C. Bunks, "Chebyshev Time-Domain Deconvolution for Transversal Filter Equalizers," to appear in Electronics Letters, December 1980
3. C. Bunks, "Minimax Equalization for Transversal Filters," Master of Science Thesis, Tufts University, 101 pages, June 1980
4. D. Preis and C. Bunks, "Minimax Time-Domain Deconvolution for Transversal-Filter Equalizers, presented at the IEEE International Conference on Acoustics, Speech and Signal Processing, Paper DSP 11.1, Fairmont Hotel, Denver, Colorado, April 1980
5. D. Preis, "Minimax Equalizers: Performance and Adjustment Algorithms," presented at the 1980 L'Aquila Workshop on Digital Signal Processing, sponsored by IEEE Acoustics, Speech and Signal Processing Society, L'Aquila, Italy, September 1980

6. D. Preis and C. Bunks, "Computational and Performance Aspects of Minimax Equalizers," accepted for presentation at IEEE International Conference on Acoustics, Speech and Signal Processing, Sheraton-Atlanta Hotel, Atlanta, Georgia, March 1981
7. D. Preis and C. Bunks, "Three Algorithms for the Design of Transversal-Filter Equalizers," submitted for presentation at IEEE International Symposium on Circuits and Systems, Chicago, Illinois, April 1981

E. Professional Personnel

Biographical Sketch of Principal Investigator. Douglas Preis received the B.S.E.E. degree in 1964 and M.S.E.E. degree in 1966, both from the University of Santa Clara, CA. He was a NASA Fellow at Utah State University, Logan, where he received the Ph.D. degree in 1969. He joined the research faculty of Harvard University, Cambridge, MA, in 1970 where he was Post-Doctoral Research Fellow in the Division of Engineering and Applied Physics at the Gordon McKay laboratory until 1978. In 1978, he joined the faculty of Tufts University, Medford, MA, as Assistant Professor of Electrical Engineering. Dr. Preis has published or presented about thirty-five original technical papers and reports in the electrical engineering field. He received the Audio Engineering Society Publications Award for his paper, "Linear Distortion," which was published in the Journal of Audio Engineering Society in 1976. Dr. Preis is a member of Sigma Xi, Tau Beta Pi. He is a member of the Review Board of the Journal of the Audio Engineering Society and an Associate Editor of the IEEE Transactions on Acoustics, Speech, and Signal Processing.

Biographical Sketch of Research Assistant. Carey Bunks simultaneously received the B.S. degree summa cum laude and M.S. degree from Tufts University both in electrical engineering in June 1980.

He received the Amos Emerson Dolbear Scholarship and the Victor Prather Prize for demonstrated excellence in scientific research while at Tufts. He presented a technical paper at ICASSP-80 in Denver and is co-author of four other technical papers. He is currently pursuing the Ph.D. degree in electrical engineering at MIT. Mr. Bunks is a member of the Tau Beta Pi and Eta Kappa Nu.

MINIMAX TIME-DOMAIN DECONVOLUTION FOR TRANSVERSAL FILTER EQUALIZERS

C. Bunks and D. Preis

Tufts University
Department of Electrical Engineering
Medford, Massachusetts 02155

ABSTRACT

The subject of this paper is the design of an optimum transversal filter for time-domain equalization of a linear, time-invariant system. Given advance specification, in the time domain, of the unequalized system response and the desired equalized response, a numerical deconvolution is performed to determine the set of filter tap weights which optimally satisfies the continuous-time, convolutional integral equation in the minimax or Chebyshev sense. This solution is computed using the second algorithm of Remez. Numerical examples are provided which illustrate the efficacy of minimax deconvolution. A computer program flow chart is included.

INTRODUCTION

Deconvolution arises in practical engineering problems as diverse as time-domain equalization, instrumentation and measurement, system identification, ultrasonic diagnostics, geophysical exploration, and imaging. Convolution is itself an integral operation whereas deconvolution, or the inverse of convolution, requires that an integral equation be solved either directly or indirectly. In most linear signal processing applications, deconvolution is accomplished numerically and approximately using either iterative techniques [1] or discrete Fourier transform methods [2]. When exact solutions do not exist, the usual procedure is to seek an approximate solution having minimum square error [3]. Large errors as well as non-physical results sometimes associated with this kind of solution can be suppressed by imposing suitable constraints when deconvolving [1], [4]. An alternative approach is a minimax or Chebyshev approximate solution [5] wherein maximum individual errors are minimized rather than the cumulative square error. In this report, time-domain equalization of a continuous-time system with an N tap transversal filter is considered as a deconvolution problem. The convolutional integral equation is solved in the Chebyshev or minimax sense using the second algorithm of Remez [5]. Maximum deconvolution errors are controlled by the number of taps and minimized uniformly.

TIME-DOMAIN DECONVOLUTION

The overall impulse response of a linear system in cascade with an equalizing filter is determined by convolution in the time domain. Assuming that the system to be equalized is causal with impulse response $h(t)$, and that there is a particular desired response $g(t)$, then the filter's impulse response $f(t)$ must satisfy,

$$\int_0^t f(x)h(t-x)dx = g(t). \quad (1)$$

Since the unknown filter response appears under the integral sign, (1) is an integral equation for $f(t)$. The solution of (1) requires that an inverse convolution or *deconvolution* be performed to find $f(t)$ when $h(t)$ and $g(t)$ are specified.

Given a transversal filter with N tap weights, assume that the filter's finite-duration impulse response can be represented by,

$$f(t) = \sum_{n=1}^N f_n \delta(t-nT), \quad (2)$$

where f_n represents the value of the n-th tap weight, δ is the unit impulse function, and T is the time delay between the equally-spaced taps. (An alternative representation for $f(t)$ is a set of N contiguous rectangular pulses with amplitudes f_n and width T. The chosen representation simplifies the following development.) Substituting (2) into integral equation (1) yields the following continuous-time representation for $g(t)$,

$$\sum_{n=1}^N f_n h(t-nT) = g(t). \quad (3)$$

This result implies that $g(t)$ can be represented exactly by a superposition of N weighted and time-shifted versions of $h(t)$. Depending on both $h(t)$ and the choice of $g(t)$, it may not be possible to satisfy (3) exactly. In such cases, the solution to this problem requires finding a set of N tap weights f_n which satisfies (3) approximately. For any specific instant in time t_m within the domain of definition of $g(t)$, (3) will yield the following linear equation with the N tap weights as unknowns,

$$\sum_{n=1}^N f_n h(t_m - nT) = g(t_m). \quad (4)$$

If M distinct instants in time are chosen, then a system of M linear equations in N unknowns is generated which takes the form,

$$\sum_{n=1}^N f_n h(t_m - nT) = g(t_m), \quad m = 1, 2, \dots, M. \quad (5)$$

Since it may not be possible to solve (3) exactly, a systematic method will be developed to find an approximate solution. This is accomplished first by considering solutions to (5).

Equation (5) suggests three possibilities: $M < N$, $M = N$, and $M > N$. For $M < N$, there are more unknowns than equations so that the system of linear equations is underspecified and, consequently, an infinite number of solutions exists. For $M = N$, the number of equations equals the number of unknowns and a unique solution exists if these equations are linearly independent. This approximate solution, known as *collocation* [6] or *point matching*, only satisfies (3) at each of the chosen N points. A different selection of N points generally will yield a different solution. It is clear in this situation that the tap weights f_n depend upon the choice of the N values of t_m . Furthermore, even though the solution to (3) matches exactly at the selected N points t_m , the approximation error may be considerable between these points. For $M > N$, the system of linear equations is overspecified because there are more equations than unknowns. In such an overspecified system no exact solution exists which simultaneously satisfies each of the M equations. Here the only recourse is to find the "best" approximate solution to (5). A popular method for solving (5) approximately, which also ensures a minimum total approximation error, is the method of *least squares* [6] or *minimum-square-error* technique. In this method the sum of the square of each of the individual errors between the approximation and the desired solution of the M equations is minimized. However, in spite of the fact that the cumulative error for the total system of M equations is minimized, the error for any individual equation is unconstrained and may be very large indeed. Therefore, it is desirable to seek an approximate solution which simultaneously and uniformly minimizes the maximum error from each of the M individual equations. The Chebyshev solution to an overspecified system of linear equations satisfies these requirements [5].

In order to formulate the Chebyshev solution to (5), it is necessary to define an *error equation* associated with each of the M linear equations having the general form,

$$r(t_m) = g(t_m) - \sum_{n=1}^N f_n h(t_m - nT), \quad m = 1, 2, \dots, M. \quad (6)$$

Next, the objective is to minimize the maximum value of the magnitude of $r(t_m)$ for $n = 1, 2, \dots, M$, by appropriate adjustment of the N tap weights f_n . The domain of definition of a continuous function such as $g(t)$ is an interval containing an infinite number of points. This fact implies that (3) can be viewed as an overspecified system of M linear equations in N unknowns in the limit as $M \rightarrow \infty$. Thus, there is a corresponding infinite number of error equations, as specified in (6), which define an *error function* $r(t)$ of the form,

$$r(t) = g(t) - \sum_{n=1}^N f_n h(t - nT) \quad (7)$$

From a different viewpoint, the location of each t_m (for $m = 1, 2, \dots, M$ and $M > N$) remains unspecified yet it is required that the magnitude of each corresponding individual error $r(t_m)$ be minimized. So, it is necessary to examine $r(t_m)$ for all values of t_m within the domain of definition of $g(t)$. Specifically, the maximum magnitude of $r(t)$ must be minimized. This can be accomplished by invoking the *characterization theorem* which predicts that the maximum value of each of the $M > N$ error equations is minimized at the intersection of a particular subset of $N + 1$ of these M equations [5]. Therefore, finding the Chebyshev minimum of the set of error equations reduces to determining the appropriate subset of $N + 1$ error equations, or, equivalently, finding the best set of $N + 1$ points in time (viz., each t_m for $m = 1, 2, \dots, N + 1$). A major difficulty here is that for M infinite, an infinite number of subsets each consisting of $N + 1$ points must be examined. A systematic search for the subset of $N + 1$ points, from the infinite set of points on the domain of definition of $g(t)$, is known as the second algorithm of Remez [5]. It is a powerful iterative technique which converges to the set of $N + 1$ points for which error function $r(t)$ is minimized in the Chebyshev sense. The N tap weights thereby obtained yield the requisite minimax solution to (3). This entire procedure is defined as *minimax deconvolution*. It is the best way to minimize deconvolution errors uniformly.

COMPUTED EXAMPLES

Figure 1 illustrates equalization of a Gaussian to approximate a sinc function on the time interval $[-3\pi, 3\pi]$ first using a 6 tap ($T = \pi$) then an 8 tap ($T = 3\pi/4$) transversal filter. In this example, $h(t) = 0.337 \exp(-t^2/27.6)$ and $g(t) = (\sin t)/t$. Maximum value of the error magnitude $|r(t)|$ is 0.21 for 6 taps and is reduced (by a factor of ten) to 0.021 for 8 taps.

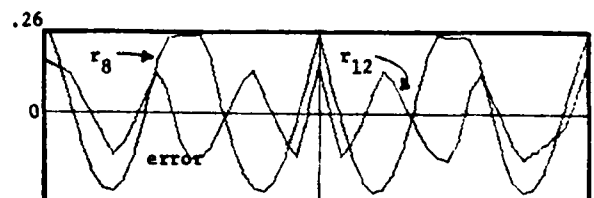
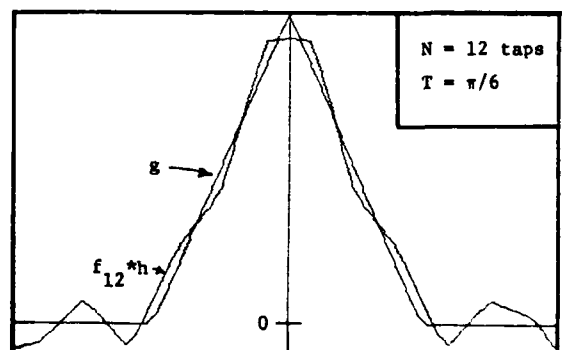
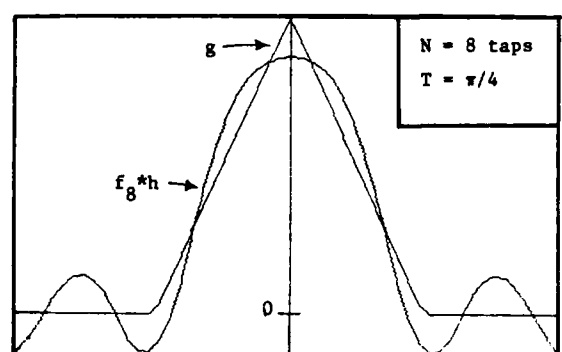
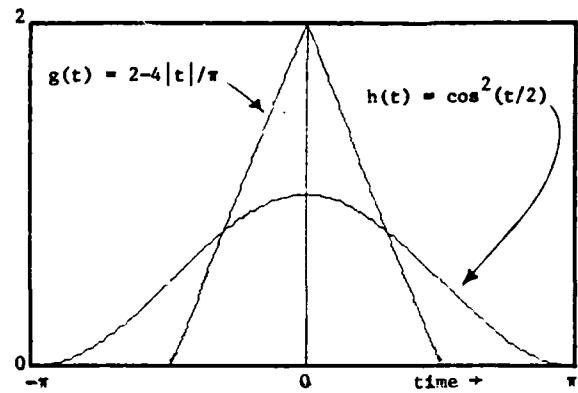
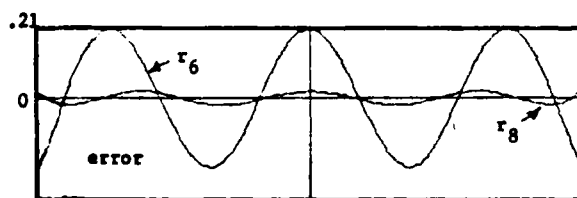
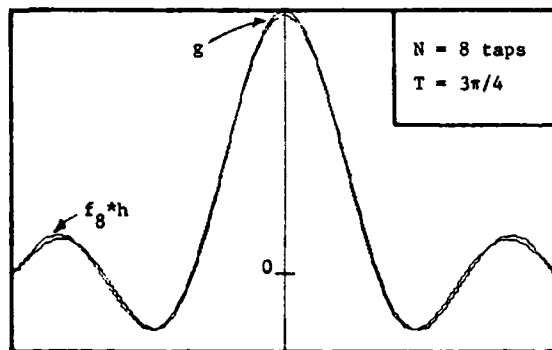
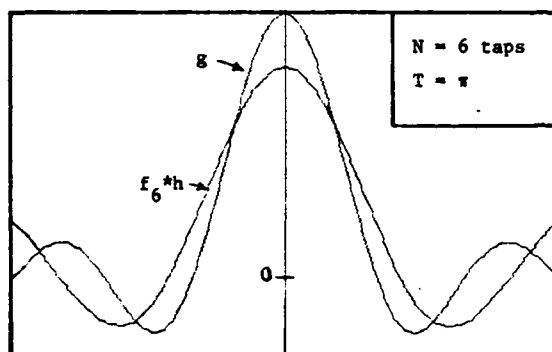
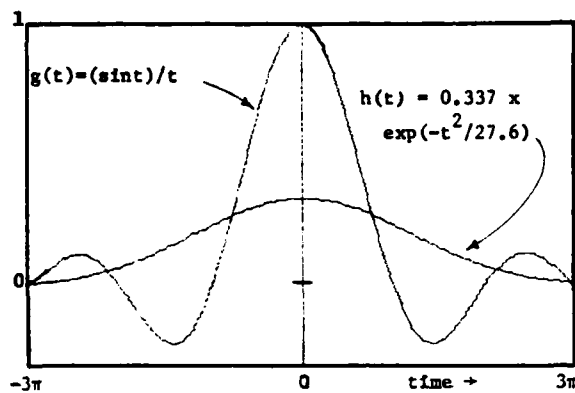


Figure 1. Minimax-deconvolution design of 6 tap (f_6) and 8 tap (f_8) transversal filters for equalization of Gaussian system function $h(t)$ to approximate sinc function $g(t)$. Lowest pair of curves shows respective approximation error functions r_6 and r_8 .

Figure 2. Minimax-deconvolution design of 8 tap (f_8) and 12 tap (f_{12}) transversal filters for equalization of raised-cosine system function $h(t)$ to approximate triangular function $g(t)$. Lowest pair of curves shows respective approximation error functions r_8 and r_{12} .

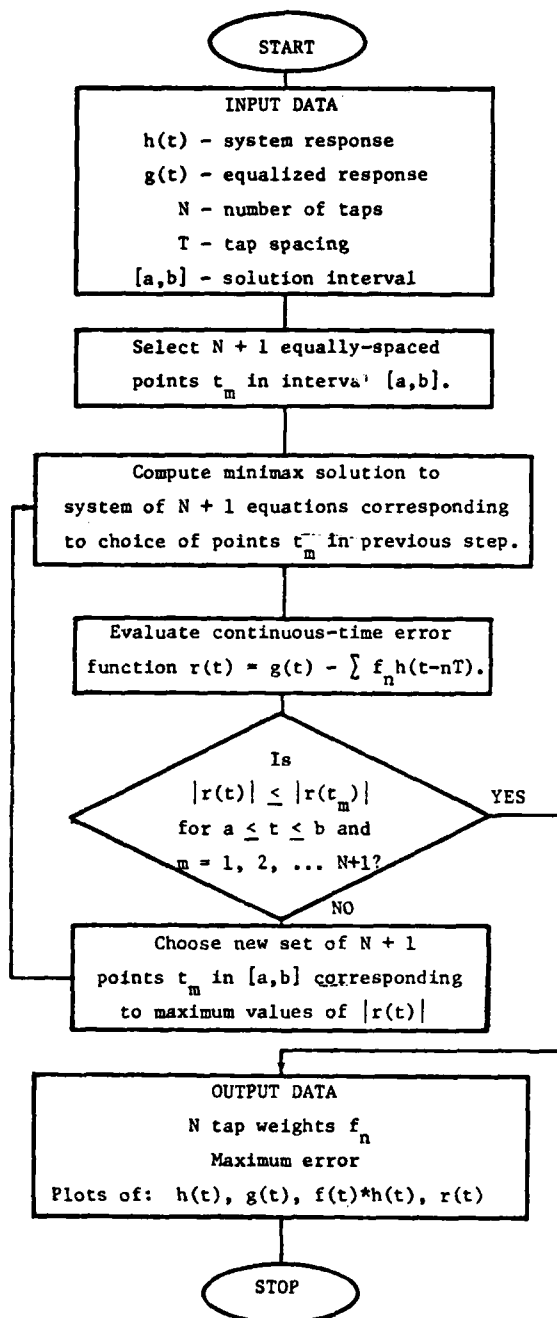


Figure 3. Flow Chart for Minimax Deconvolution

Figure 2 shows a raised-cosine pulse equalized to approximate a narrow triangular pulse on the time interval $[-\pi, \pi]$ using an 8 tap ($T = \pi/4$) then a 12 tap ($T = \pi/6$) transversal

filter. For this second example, $h(t) = 0.5 \times (1 + \cos t) = \cos^2(t/2)$; $g(t) = 2 - 4|t|/\pi$ for $|t| < \pi/2$ and $g(t) = 0$ for $|t| > \pi/2$. Maximum value of the error magnitude $|r(t)|$ is 0.263 for 8 taps and is reduced to 0.159 for 12 taps. On a percentage basis, these errors are 13% and 8%, respectively, because the peak value of $g(t)$ is 2.

COMPUTER PROGRAM

The computer program (see Figure 3) first selects $N + 1$ evenly-spaced points t_m in the solution interval $[a, b]$. Each of these points determines one linear equation with the N tap weights as unknowns. The minimax solution to this overspecified system of $N + 1$ equations in N unknowns is computed. Next, the resulting continuous-time error function $r(t)$ is evaluated. If the maxima of $|r(t)|$ do not occur at the chosen values of t_m then the continuous-time minimax solution has not been found. The next iteration begins by selecting new points t_m in the interval $[a, b]$ which correspond to the $N + 1$ largest values of $|r(t)|$. The minimax solution to this new overspecified system is computed and its error function $r(t)$ evaluated. The process is repeated until $|r(t)|$ is bounded by $|r(t_m)|$. When this occurs, the appropriate subset of $N + 1$ points t_m is known and the minimax solution to its corresponding overspecified system of $N + 1$ linear equations in N unknowns yields that set of N tap weights which satisfies the convolutional integral equation in the minimax sense. In summary, the Remez algorithm is a systematic, iterative procedure for finding a unique set of $N + 1$ points t_m in the interval exchanging one set of $N + 1$ points t_m for another until each point corresponds to an extreme value of the error function $r(t)$.

REFERENCES

- [1] R.M. Mersereau and R.W. Schafer, "Comparative Study of Iterative Deconvolution Algorithms," *IEEE Conference on Acoustics, Speech, and Signal Processing*, pp. 192-194, April, 1978.
- [2] B.R. Hunt, "Biased Estimation of Nonparametric Identification of Linear Systems," *Math. Biosci.*, vol. 10, pp. 215-237, 1971.
- [3] D. Preis, "Least-Squares Time-Domain Deconvolution for Transversal-Filter Equalizers," *Electron Lett.*, vol. 13, pp. 356-357, June, 1977.
- [4] D. Preis, "Envelope-Constrained Time-Domain Deconvolution for Transversal-Filter Equalizers," *Electron. Lett.*, vol. 14, pp. 37-38, Jan. 1978.
- [5] E.W. Cheney, *Introduction to Approximation Theory*. New York: McGraw-Hill, 1966, Chapter 3.
- [6] F.B. Hildebrand, *Methods of Applied Mathematics*. New Jersey: Prentice Hall, 1965, Chapter 3.

MINIMAX EQUALIZERS: PERFORMANCE AND
ADJUSTMENT ALGORITHMS

D. Preis

Tufts University
Department of Electrical Engineering
Medford, Massachusetts 02138 USA

- Abstract -

Minimax time-domain deconvolution to determine the N tap weights of a transversal filter equalizer (FIR) is accomplished iteratively using the Remez algorithm. This minimax design can be converted to a recursive (IIR) filter possibly having greater error but only half the number of delay elements T . Results of computations for three different examples illustrate the dependence of equalization error ϵ on the number of taps N and width of solution interval W . Computation time and convergence are related to N . Representative FIR and IIR designs are compared.

Presented at 1980 L'Aquila Workshop on Digital Signal Processing,
sponsored by IEEE ASSP Society, L'Aquila, Italy, September 1980

INTRODUCTION

The subject of this contribution is the automatic design of an optimum transversal filter for time-domain equalization of a linear system. The equalization problem is formulated in the time domain as an integral equation of convolution. Frequency-domain techniques are not used. Given advance specification (or an estimate), in the time domain, of the unequalized system response and the desired equalized response, a numerical deconvolution is performed to determine the set of filter tap weights which optimally satisfies the convolutional integral equation in the minimax or Chebyshev sense. The transversal-filter equalizer is assumed to be a tapped delay line having either a finite-duration impulse response or an infinite-duration impulse response while the linear system to be equalized is continuous-time. The filter tap weights are computed iteratively using the second algorithm of Remez. A major advantage of minimax time-domain deconvolution for equalizer design is that the error magnitude is both minimized and bounded. The question of solution feasibility, which arises in envelope-constrained deconvolution, is avoided. Also, the problem of large, unconstrained individual errors, which often occurs with least-square-error approximate solutions or discrete-Fourier-transform solutions, is eliminated.

Specific aspects of the equalization problem also considered here include an intercomparison of the performance of both recursive and non-recursive minimax transversal-filter equalizers in terms of equalization accuracy and filter length.

Prospective applications of this research include: high-data-rate communication through time-varying dispersive channels, reduction of intersymbol interference, system identification and modeling, minimax inverse filtering, multichannel filtering, and sensor array equalization. Potential advantages of minimax time-domain equalization are: a reduction of hardware and software complexity and costs normally associated with time-domain to frequency-domain and frequency-domain to time-domain transformations, increased processing speed, uniform control and minimization of deconvolution errors in the time domain, and the possibility of reduced filter complexity.

Computed results for about 50 FIR equalizer designs are summarized in graphical forms which relate accuracy, convergence, and computation time to filter length. Also illustrated are sensitivities of equalization error to solution-interval width. These minimax designs were computed using the Remez algorithm [1].

EXAMPLES AND RESULTS

Figure 1 summarizes three different time-domain equalizer design problems. In the left column of this figure are the key symbols which are used in subsequent figures to identify each of these three examples. The first case is equalization of a Gaussian to a Sinc function, the second, a raised cosine to a triangular function and the third, a raised cosine to a rectangular function. The filter or equalizer impulse response is $f(t)$, the system response $h(t)$, and the overall response $g(t)$. The minimax equalizer is a tapped delay line or transversal filter having finite-duration impulse response (FIR).

In some examples to follow infinite duration impulse response (IRR) filters also will be considered. Note that $h(t)$ and $g(t)$ are continuous-time functions while the filter $f(t)$ is discrete-time device.

Two equalizer canonical forms are shown in Figure 2. The letter T indicates a time delay of T seconds. Lower case letters f , a , and b designate filter coefficients or tap weights.

Figure 3 illustrates the dependence of equalization accuracy on the number of taps N for the FIR designs. The number of taps equals the number of delay elements minus one. Since the filter output is a weighted sum of time delayed versions of the input, the symmetry of the taps about $t = 0$ is important. For N odd equalization accuracy is better than N even because the middle tap is centrally located at $t = 0$. Considering N even or N odd separately it is seen that, for fixed solution interval width W , the error decreases the equalization accuracy improves as the number of taps N is increased. The error is computed as the ratio of maximum deviation of equalized response from $g(t)$ to maximum value of $g(t)$ and this is expressed in percent. Note that for the third example (square symbol) the error can not be less than 50% because (at best) the equalized response will pass through the vertical sides of the rectangle at their midpoints.

In Figure 4 minimax equalization error is plotted for the three test cases where the number of taps is fixed at $N = 8$ and the width of the solution interval W is varied. It seems intuitive that the error should increase as the solution interval is made wider, however, this is not necessarily true as can be observed for the first example (round dots). Also, it is interesting to note that if

the solution interval is chosen to be wider than the sum of the durations of $h(t)$ and $f(t)$ then no change in error can be expected since the convolution is zero for $|t|$ large.

Convergence of the Remez algorithm depends on several factors. Figure 5 illustrates convergence as a function of the number of tap weights N . It is seen that filter symmetry (N odd or even) is influential. On the average, the number of iterations required for convergence is approximately equal to the number of taps N .

Figure 6 shows total computation time in seconds as a function of the number of tap weights N . With one exception, computation time increases with N . Those filters offering higher accuracy (N odd) require proportionally more computation time.

Figure 7 combines the data of Figures 5 and 6 to illustrate the average time per iteration as a function of N . Note that the increased time for the first example (round dots) is due to the comparatively more complicated form of the mathematical expression for $g(t)$ and $h(t)$.

Figure 8 shows the trade-off between filter accuracy and invested computation time. Equalization error can be reduced significantly by lengthening the filter. For short filters, computation time and accuracy are related linearly.

Figure 9 is a table of two sets of computed tap weights for nonrecursive (FIR) and recursive (IIR) designs when $N = 7$, $T = \pi/7$, for $W = 6\pi$, and for $W = 12\pi$. Tap weights for the IIR case were computed from the FIR values using an algorithm [2] which forces the initial N values of the impulse response of the IIR filter to equal the corresponding values of the FIR filter.

Figure 10 compares FIR and IIR equalized responses with $g(t)$. The maximum FIR error in the solution interval $W = 6\pi$ is 1.52% for $N = 7$.

Figure 11 illustrates the large errors (side lobes) outside the solution interval $W = 6\pi$ for the designs of Figure 10. These sidelobes can be reduced by increasing W as shown in Figure 12 where the maximum error is 13.3%.

REFERENCES

- [1] D. Preis and C. Bunks, "Minimax Time-Domain Deconvolution for Transversal Filter Equalizers," IEEE Conference Record, 1980 ICASSP, pp. 943-946, April 1980.
- [2] C. S. Burrus and T. W. Parks, "Time Domain Design of Recursive Digital Filters," IEEE Trans. AU-18 pp. 137-141, June 1970.

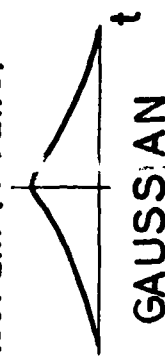
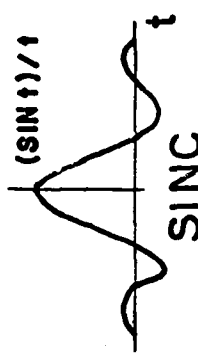
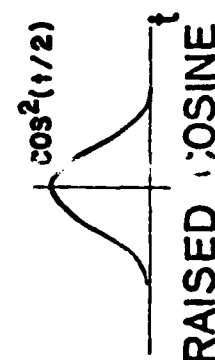
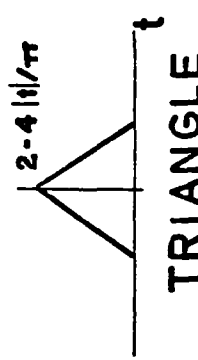
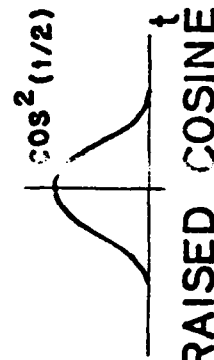
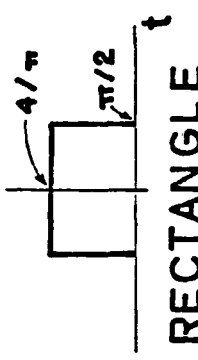
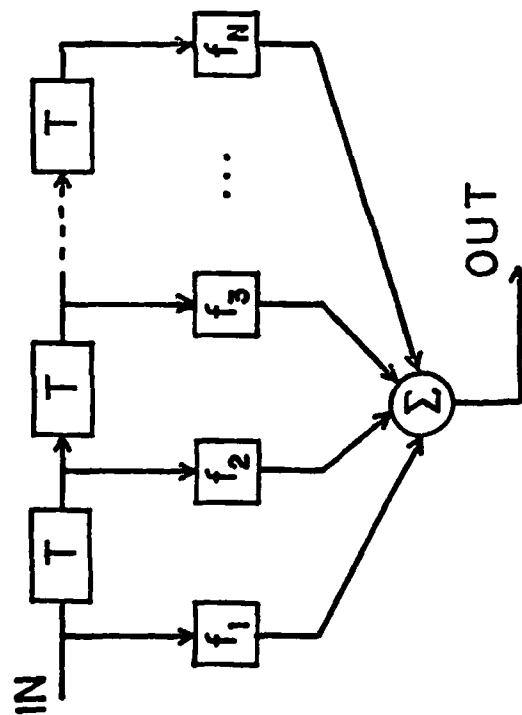
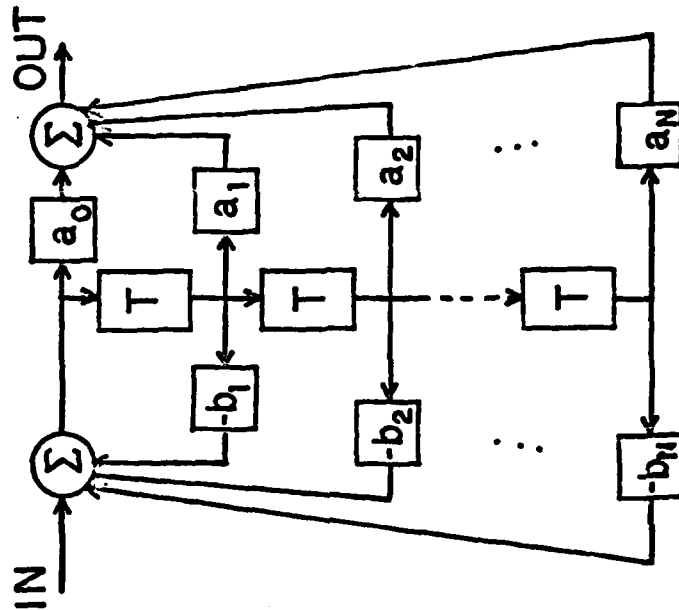
EXAMPLES		RESPONSE	
KEY	SYSTEM $h(t)$	FILTER $f(t)$	$=$ $g(t)$
1	$.337 \exp(-t^2/27.6)$  GAUSSIAN	MINIMAX EQUALIZER	 SINC
2	$\cos^2(t/2)$  RAISED COSINE	MINIMAX EQUALIZER	 TRIANGLE
3	$\cos^2(t/2)$  RAISED COSINE	MINIMAX EQUALIZER	 RECTANGLE

Figure 1. Three equalizer design examples

EQUALIZER CANONICAL FORMS



(a) NONRECURSIVE
(FIR)



(b) RECURSIVE
(IIR)

Figure 2. Equalizer canonical forms

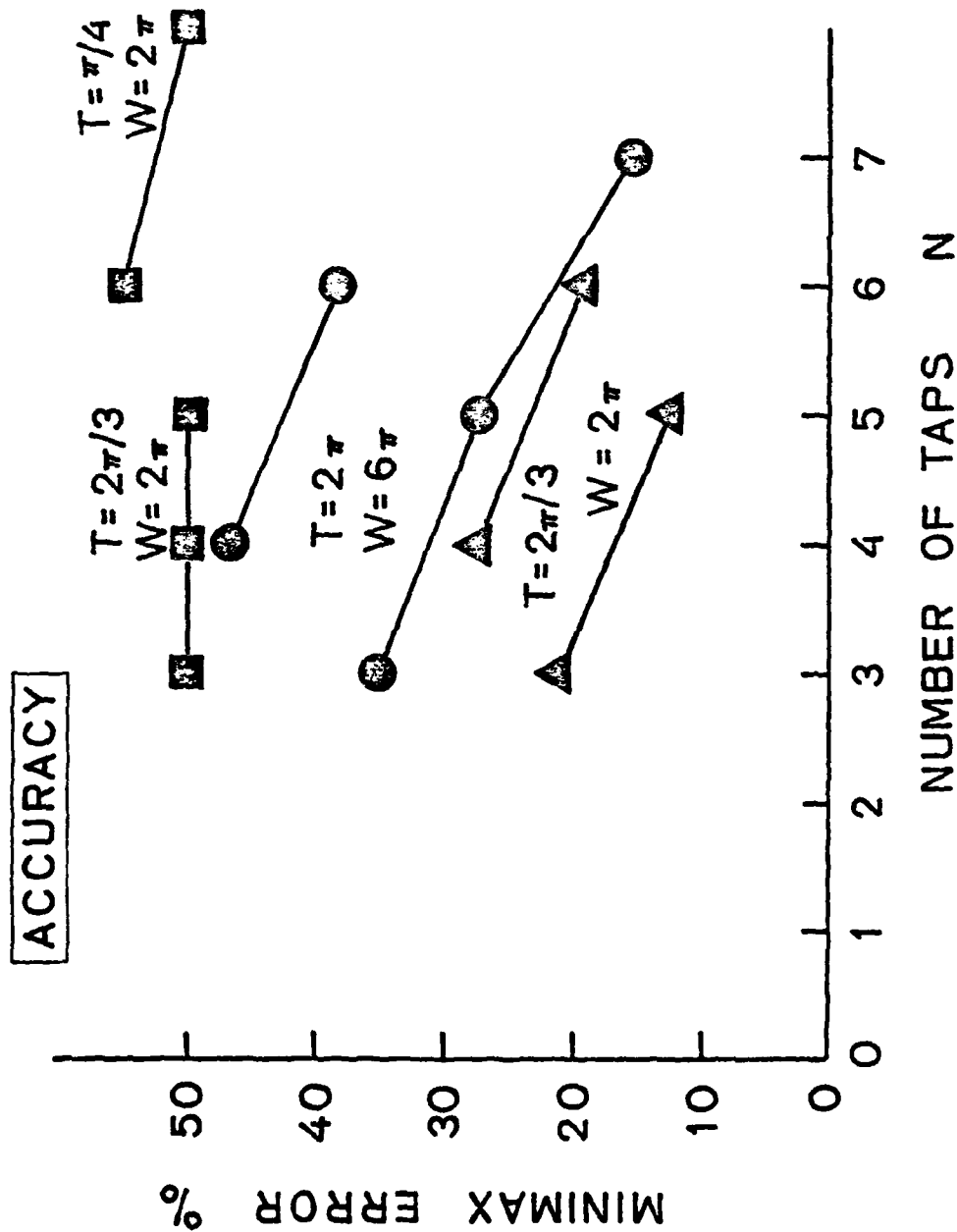


Figure 3. Dependence of minimax error on number of taps N

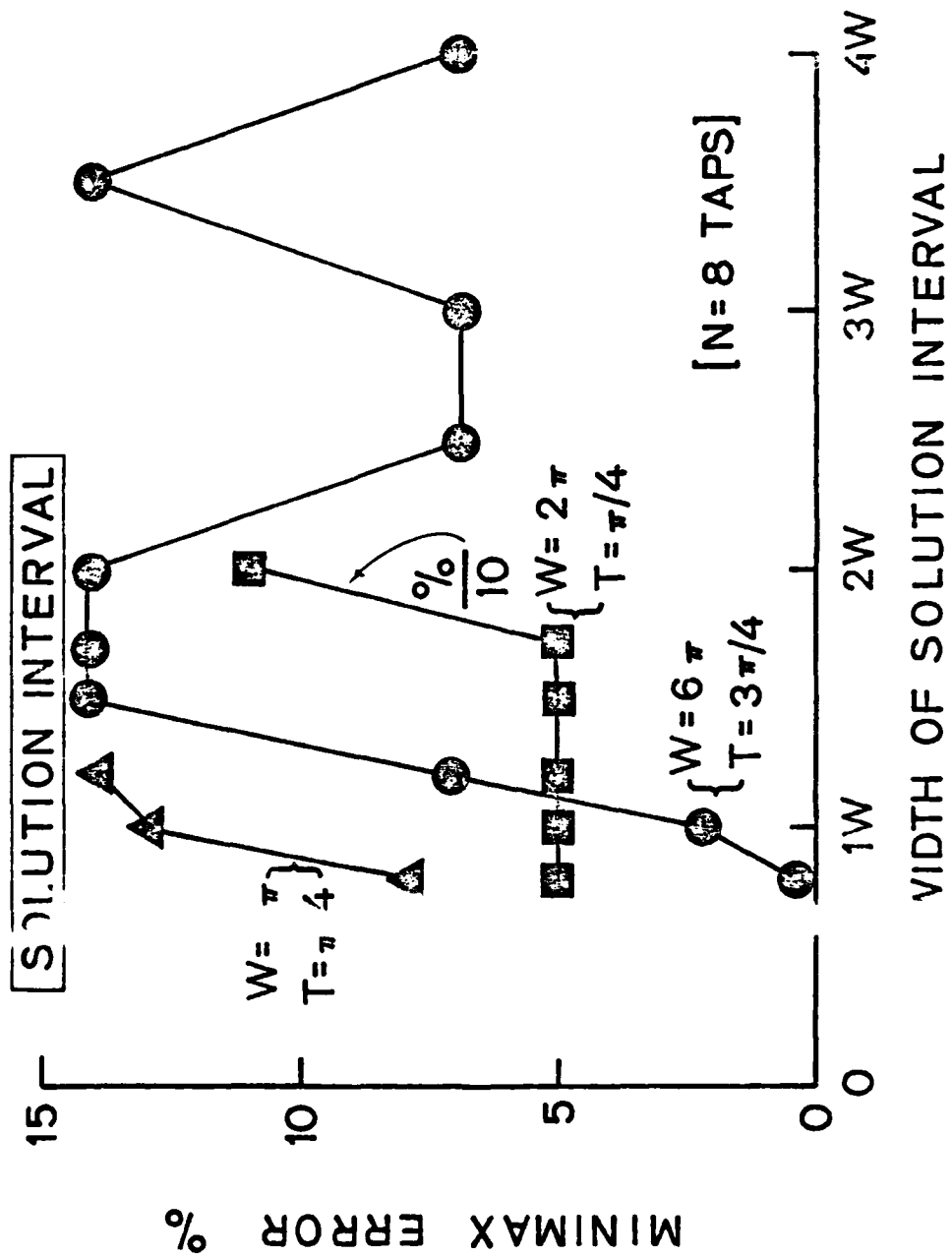


Figure 4. Dependence of minimax error on width of solution interval

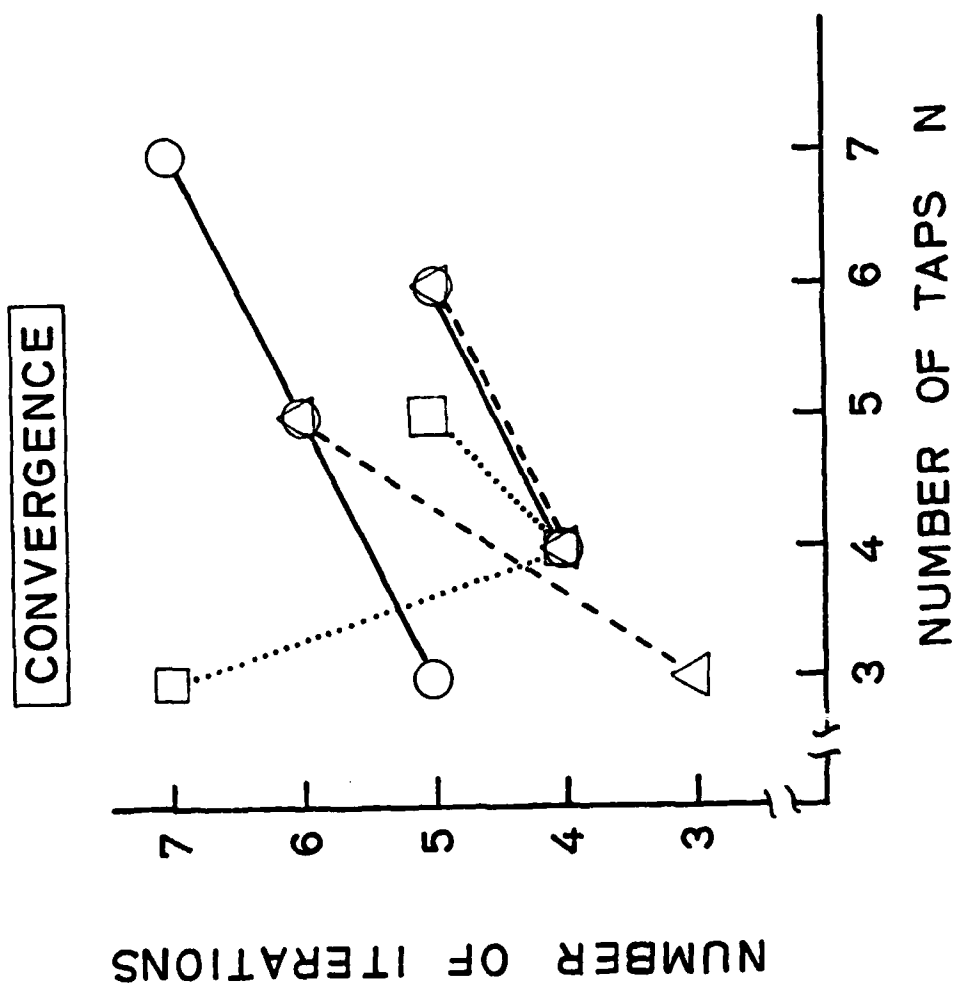


Figure 5. Number of iterations for convergence vs. number of taps N

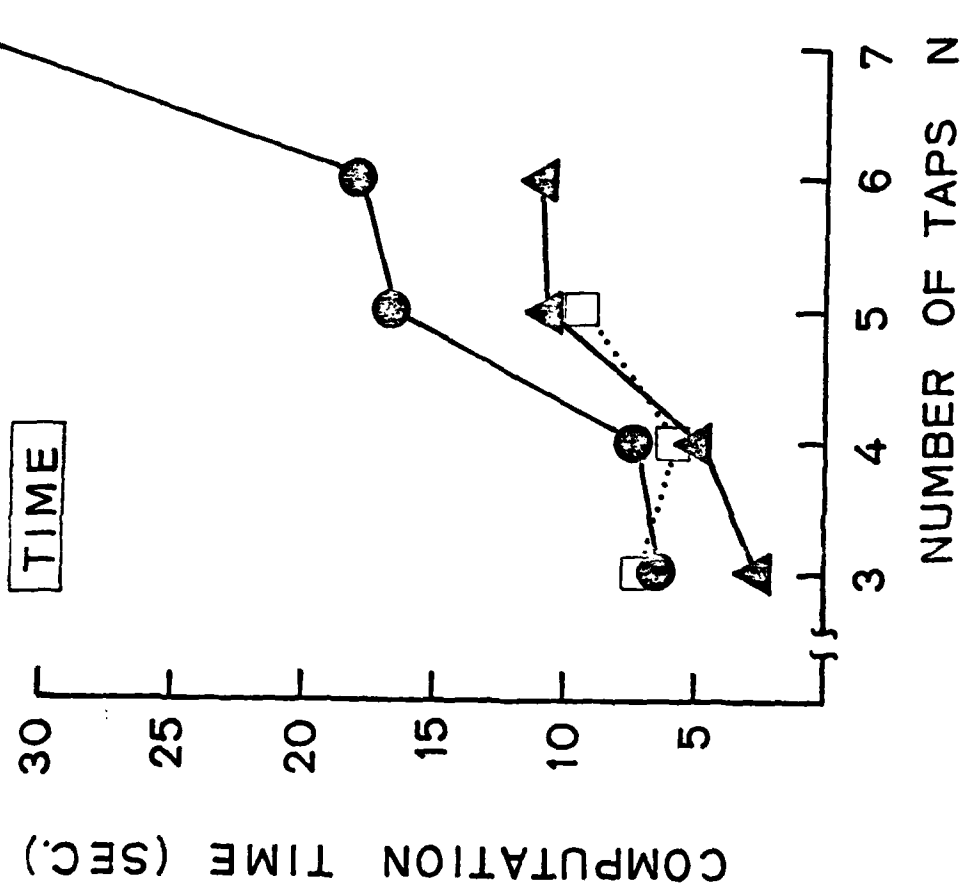


Figure 6. Computation time vs. number of taps N

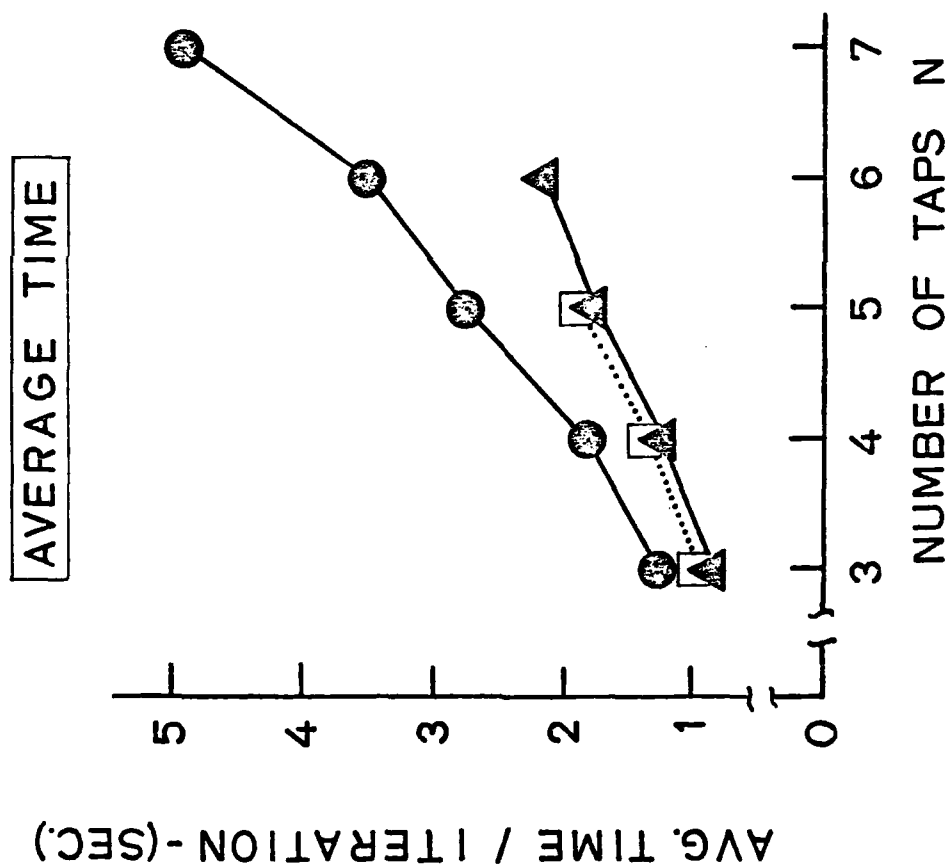


Figure 7. Average time per iteration vs. number of taps N

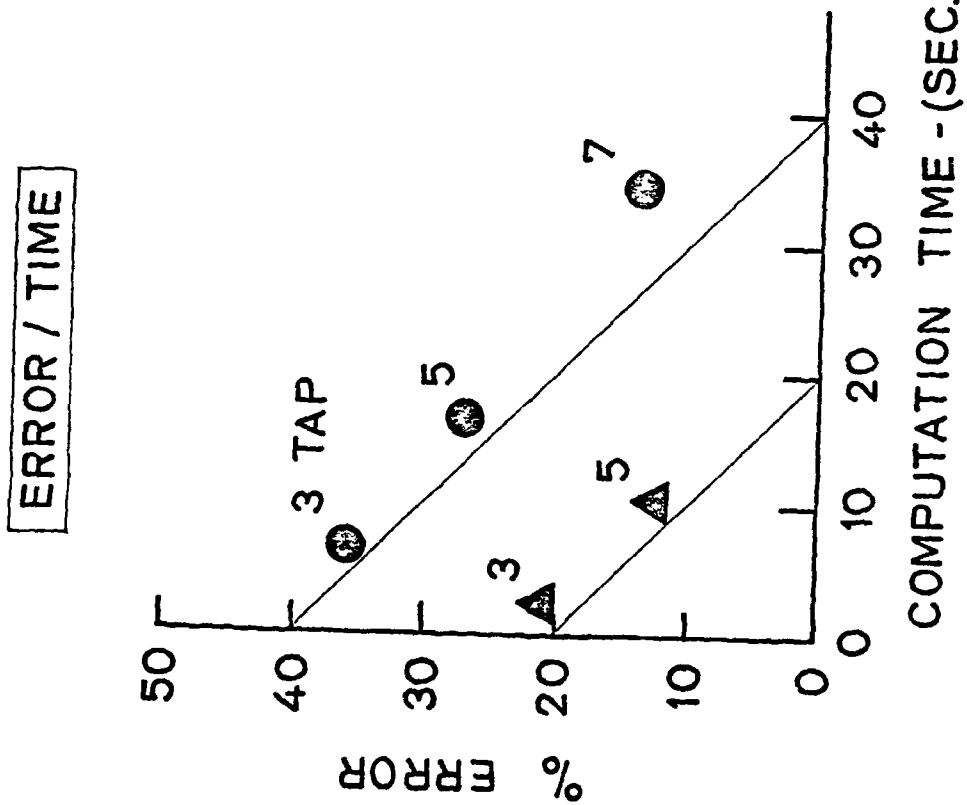


Figure 8. Reduction of equalization error as a function of total computation time for 3, 5, and 7 tap filters

$N = 7$ $N = 7$
 $T = 6\pi/7$ $T = 6\pi/7$
 $W = 6\pi$ $W = 12\pi$

TAP WEIGHTS
 (GAUSSIAN $\rightarrow$ SINC)

f	a	b	f	a	b
-10.26	-10.26	1.51	-2.84	-2.84	1.23
34.53	19.02	1.04	12.60	9.03	.78
-64.17	-22.64	.31	-28.03	-14.44	.25
79.41	15.06		37.41	11.42	
-64.17			-28.03		
34.53			12.60		
-10.26			-2.84		
1.52 %	7.37 %	← ERROR	13.3 %	13.3 %	

Figure 9. Tap weights for FIR and IIR equalizer using in figures to follow

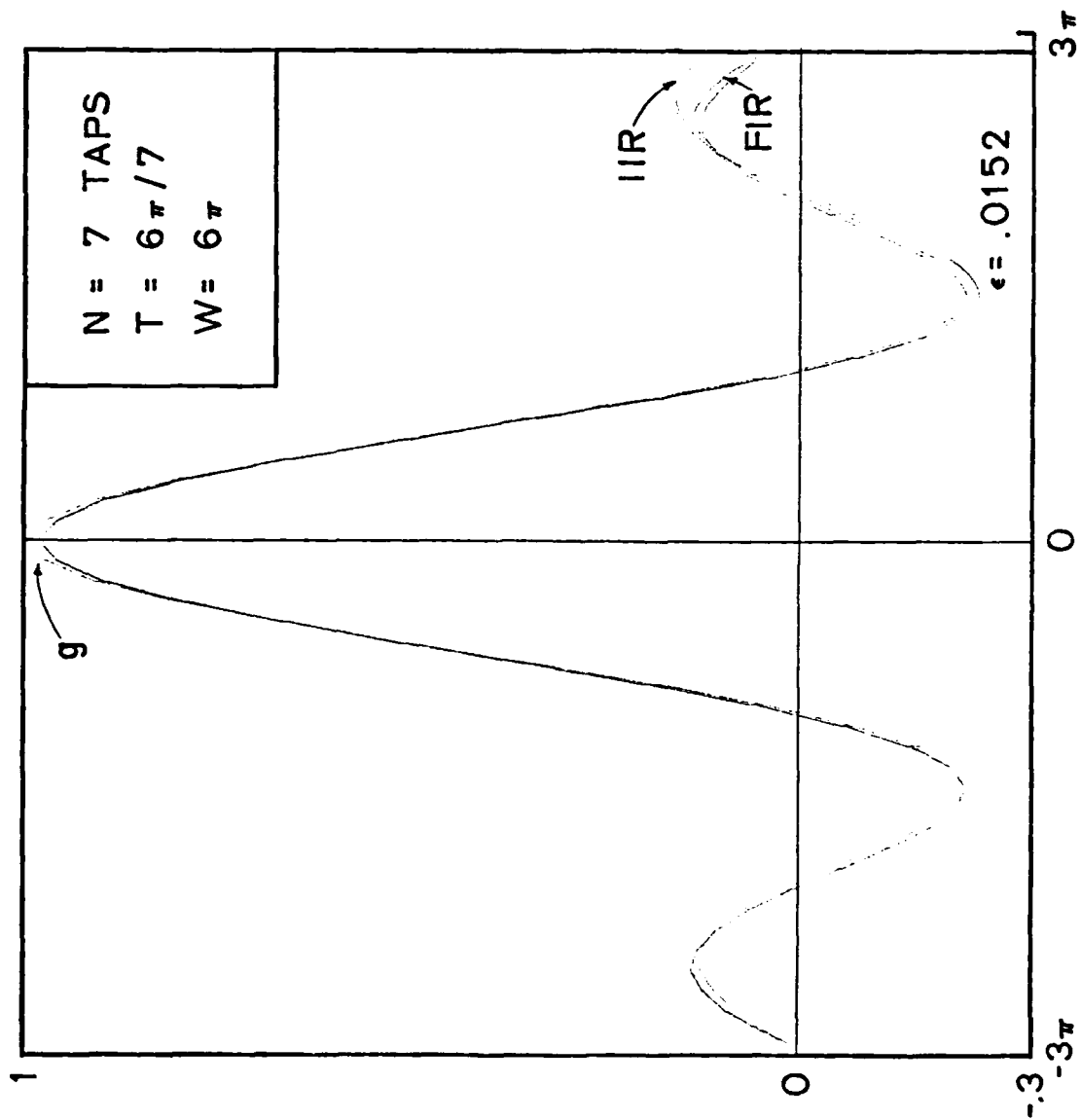


Figure 10. Comparison of $g(t)$ with FIR and IIR approximations over 6π solution interval W .

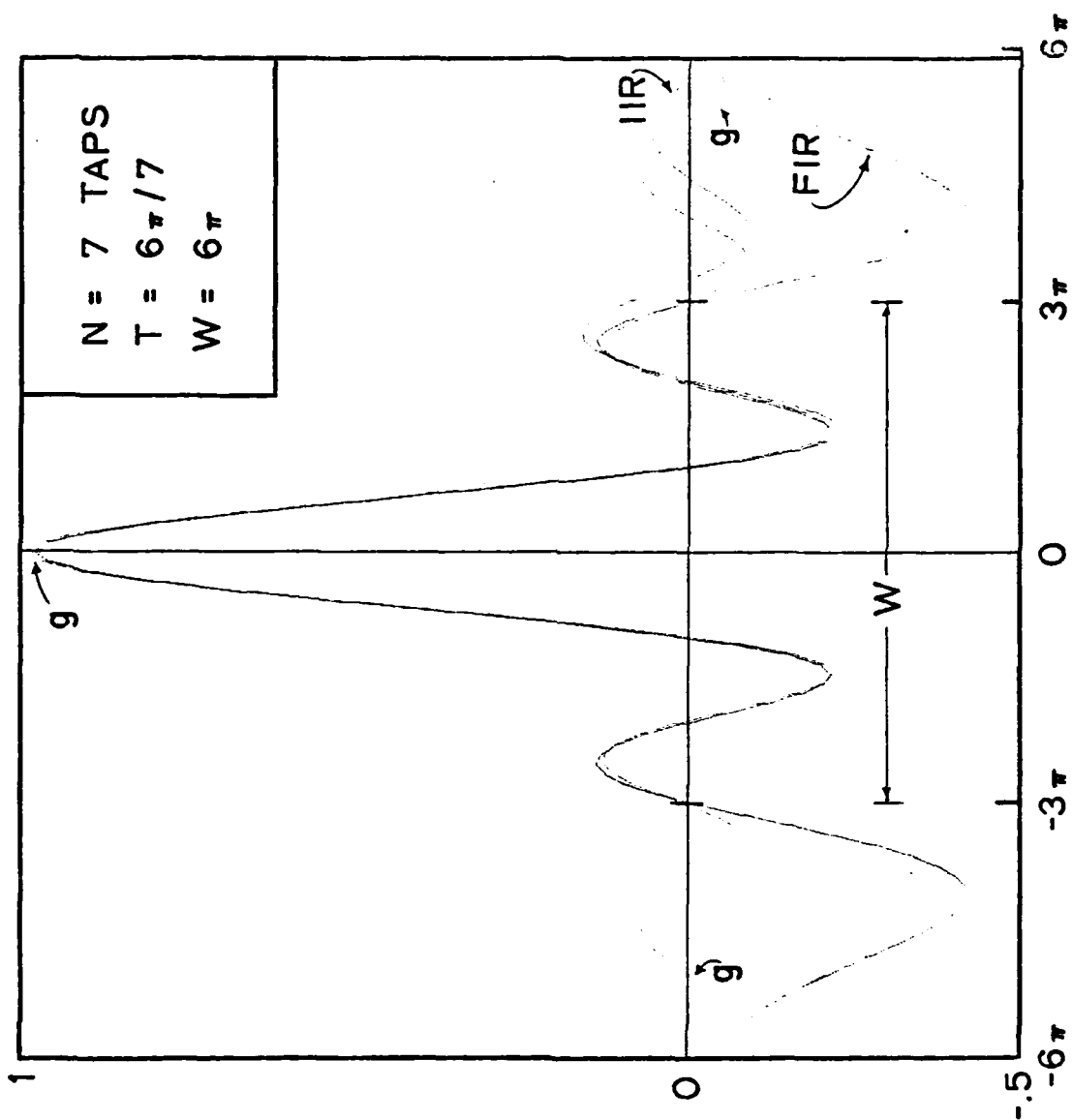


Figure 11. Same as previous figure but including response outside solution interval W .

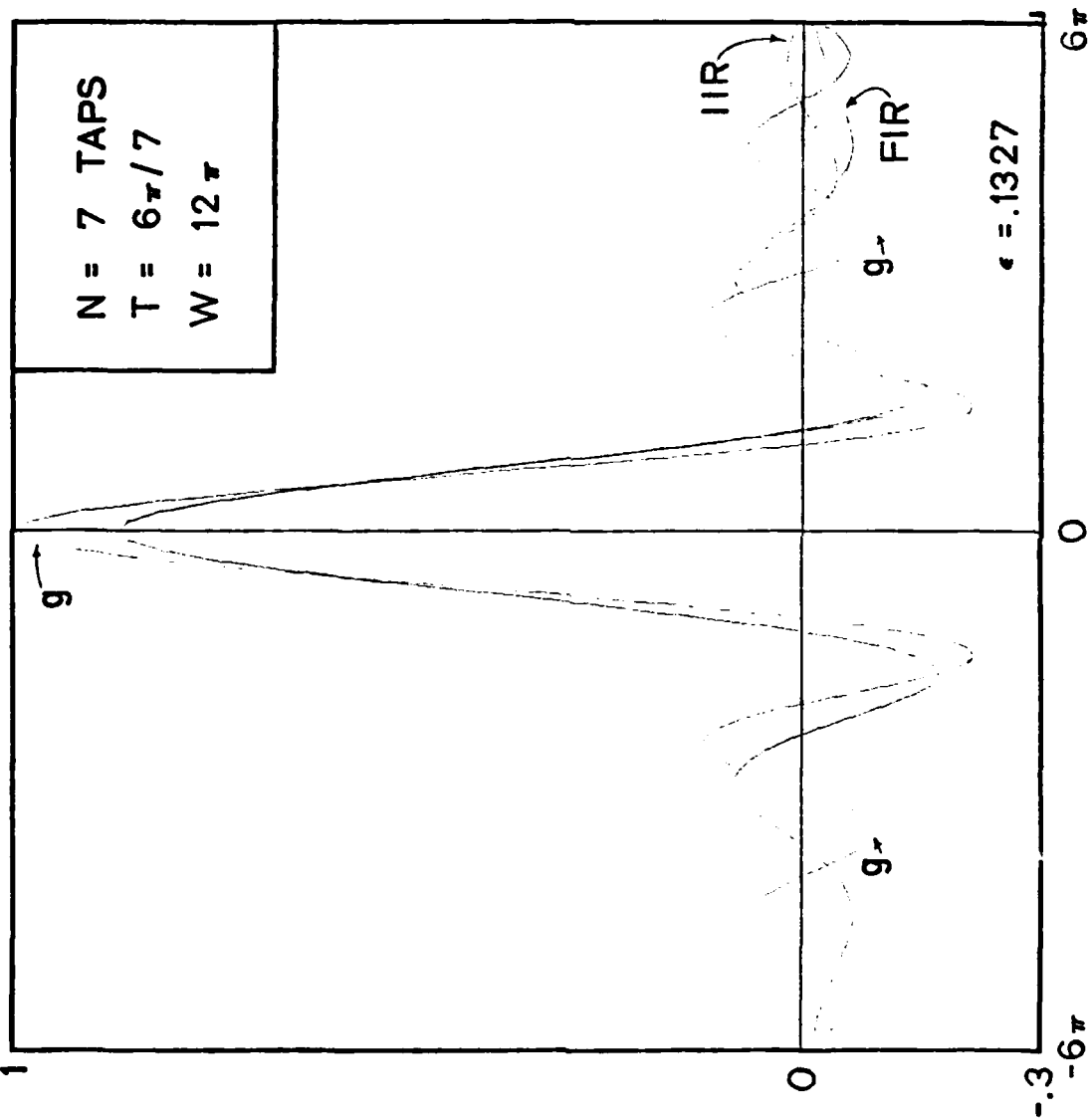


Figure 12. Comparison of $g(t)$ with FIR and IIR approximations over 12π solution interval

CHEBYSHEV TIME-DOMAIN
DECONVOLUTION FOR TRANSVERSAL-
FILTER EQUALIZERS

D. Preis and C. Bunks*

- Abstract -

Design equations are presented for computing the N tap weights of a transversal filter equalizer. The associated time-domain deconvolution is performed subject to the Chebyshev (minimax) error criterion. Individual or pointwise deconvolution errors are controlled and minimized uniformly. This method offers an alternative to least-square-error or discrete-Fourier-transform designs wherein only the cumulative deconvolution error is minimized. Computationally, the Ascent algorithm is used which requires inversion of two $N + 1$ by $N + 1$ matrices followed by a finite number of elementary algebraic exchange operations. The equations needed to implement this algorithm are presented in a form suitable for digital computation.

To appear in Electronics Letters

*The authors are with the Department of Electrical Engineering,
Tufts University, Medford, Massachusetts 02155 USA.

When a transversal filter (tapped delay line) is used as an equalizer for a discrete-time system, it is necessary to compute the N tap weights of the filter by deconvolution. Because the operation of discrete-time convolution generates more equations than unknowns, deconvolution generally will be inexact and, consequently, equalization will be approximate. Any approximate solution to this equalizer design problem will depend on the error criterion and constraints imposed when deconvolving. In two previous communications^{1,2} design equations were given for least-square-error and envelope-constrained transversal-filter equalizers. The present design procedure offers an alternative wherein the maximum deconvolution errors are minimized uniformly. This is known as the Chebyshev (or minimax) approximate solution.

Consider the discrete-time description of equalization given by the convolution summation,

$$g(m) = \sum_{n=1}^m h(m-n+1) f(n), \quad (1)$$

where g is a specified M element sequence representing the desired equalized impulse response, f is an N element sequence composed of the unknown transversal-filter tap weights, and h is a known $M-N+1$ element sequence which represents the impulse response of the linear, time-invariant system requiring equalization. It is assumed, for convenience, that the time interval between adjacent sequence elements is unity. In all practical cases, the sequence h has more than one element so that $M > N$ and eqn. 1 generates an overspecified system of M linear equations in N unknowns. Since an overspecified system such as this may not have an exact solution, it is appropriate to consider approximate solutions. A useful measure of the approximation error (or equalization error) corresponding to a specific set of N tap weights f is the M element error sequence r defined by

$$r(m) = g(m) - \sum_{n=1}^m h(m-n+1) f(n) \quad (2)$$

Comparison of eqn. 2 with eqn. 1 will indicate that the closer each numerical value $r(m)$ is to zero, the better the approximate solution becomes. If, for example, $r(m) = 0$ for $m = 1, 2, 3, \dots M$ then the N tap weights satisfy eqn. 1 exactly.

The intent here is to outline a procedure for computing those N tap weights f which both minimize and bound all the elements of the error sequence r , that is,

$$|r(m)| \leq \epsilon, \quad (3)$$

where ϵ is the smallest possible non-negative number. This solution is known as the Chebyshev (or minimum-maximum-error) approximate solution. There are $N+1$ unknowns for a given equalizer design, namely, the N tap weights and ϵ . Intuitively, these unknown quantities can be found by solving an appropriate set of $N+1$ linear equations in $N+1$ unknowns. It turns out that this set is, in fact, a subset of the M equations specified by eqn. 1. Assuming $\epsilon > 0$ the Chebyshev solution to these $N+1$ equations is characterized by the following two properties. First,

$$r(m_i) = \sigma(m_i)\epsilon \quad (4)$$

for $N+1$ of the M elements of the error sequence given by eqn. 2 where $\sigma(m_i)$ equals $+1$ or -1 and the subscript $i = 1, 2, 3, \dots N+1$ denotes $N+1$ specific integer values of m from the total set of M values. Second, the remaining $M - (N+1)$ elements of r will satisfy eqn. 3. One systematic search for the appropriate set of $N+1$ linear equations and the subsequent solution for the N tap weights f and maximum error ϵ is known as the Ascent algorithm.³ The equations required to implement this algorithm follow.

Initiate the search by selecting any subset of $N+1$ of the M equations given by eqn. 2. Using eqn. 4 write the result in the following matrix form,

$$\begin{bmatrix} g(m_1) \\ g(m_2) \\ g(m_3) \\ \vdots \\ g(m_{N+1}) \end{bmatrix} = \begin{bmatrix} \sigma(m_1) & h(m_1) & h(m_1-1) & \dots & h(m_1-N+1) \\ \sigma(m_2) & h(m_2) & h(m_2-1) & \dots & h(m_2-N+1) \\ \sigma(m_3) & h(m_3) & h(m_3-1) & \dots & h(m_3-N+1) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma(m_{N+1}) & h(m_{N+1}) & h(m_{N+1}-1) & \dots & h(m_{N+1}-N+1) \end{bmatrix} \begin{bmatrix} \epsilon \\ f(1) \\ f(2) \\ \vdots \\ f(N) \end{bmatrix} \quad (5)$$

where the subscripts associated with the index m denote those specific $N+1$ integer values of m selected from the M available values. Each of the $N+1$ values of σ appearing in eqn. 5 is either $+1$ or -1 and is specified by the relation

$$\sigma(m_1) = \text{sgn} [\theta(m_1)], \quad (6)$$

where the numerical value of each of the $N+1$ elements of θ is found by inverting the following matrix equation,

$$\begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ h(m_1) & h(m_2) & h(m_3) & \dots & h(m_{N+1}) \\ h(m_1-1) & h(m_2-1) & h(m_3-1) & \dots & h(m_{N+1}-1) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h(m_1-N+1) & h(m_2-N+1) & h(m_3-N+1) & \dots & h(m_{N+1}-N+1) \end{bmatrix} \begin{bmatrix} \theta(m_1) \\ \theta(m_2) \\ \theta(m_3) \\ \vdots \\ \theta(m_{N+1}) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (7)$$

The existence of the Chebyshev solution requires that zero be contained in the convex hull of the $N+1$ vectors whose N components are the elements of the $N+1$ columns appearing below the top row of the square matrix in eqn. 7. This condition is satisfied if a linear combination of these $N+1$ vectors can be found which yields the null (zero) vector and if those $N+1$ scalars θ , used to form the linear combination, sum

to unity. Equation 7 is a mathematical statement of these two requirements. The algebraic sign of each of the $N+1$ scalars θ determines whether the error ϵ , associated with each linear equation in eqn. 5, is positive or negative.

Write the solution to eqn. 5 in matrix form as,

$$\begin{bmatrix} c(0,0) & c(0,1) & \dots & c(0,j) & \dots & c(0,N) \\ c(1,0) & c(1,1) & \dots & c(1,j) & \dots & c(1,N) \\ \vdots & \vdots & & \vdots & & \vdots \\ c(k,0) & c(k,1) & \dots & c(k,j) & \dots & c(k,N) \\ \vdots & \vdots & & \vdots & & \vdots \\ c(N,0) & c(N,1) & \dots & c(N,j) & \dots & c(N,N) \end{bmatrix} \begin{bmatrix} g(m_1) \\ g(m_2) \\ \vdots \\ g(m_{N+1}) \end{bmatrix} = \begin{bmatrix} \epsilon \\ f(1) \\ \vdots \\ f(N) \end{bmatrix} \quad (8)$$

The matrix multiplication prescribed by eqn. 8 yields the initial set of numerical values of the N tap weights f and error ϵ . Insert these computed tap weights in eqn. 2 and examine each element of the error sequence r to verify that $|r(m)| \leq \epsilon$ for $m = 1, 2, 3, \dots M$. If this is true then eqn. 3 is satisfied and the Chebyshev solution has been found. No further computations are necessary.

If eqn. 3 is not satisfied then the following iterative exchange procedure is required. Select that specific integer $m=p$ for which $|r(m)|$ is maximum and let

$$\mu = \text{sgn } [r(p)] \quad (9)$$

Next, calculate the $N+1$ elements of sequence $a(i)$ defined by

$$a(j) = c(0,j) + \sum_{k=1}^N h(p-k+1) c(k,j) \quad (10)$$

Now select the integer q so that

$$b(q) = \mu a(q)/c(0,q) \quad (11)$$

is a maximum. Next replace the $N+1$ element column $c(k, q)$ in eqn. 8

with $c'(k,q)$ where

$$c'(k,q) = c(k,q)/a(q), \quad (12)$$

and replace all other elements $c(k,j)$ in eqn. 8 for which $j \neq q$ with $c'(k,j)$ where

$$c'(k,j) = c(k,j) - a(j)c'(k,q) \quad (13)$$

Finally, replace $g(q)$ with $g(p)$ in eqn. 8. Recompute the error ϵ , and the sequences f and r using eqn. 8 and eqn. 2. As before, verify that $|r(m)| \leq \epsilon$ for $m = 1, 2, 3, \dots M$. If eqn. 3 is not satisfied then repeat the foregoing exchange procedure by selecting a new value of p and returning to eqn. 9.

Note that although $M - (N+1)$ exchange operations are possible, the Chebyshev solution usually is found with fewer iterations than this. The maximum error ϵ will increase (ascend) with each successive iteration.

In certain applications, h and g are continuous-time functions rather than finite sequences of sampled values as in eqn. 1. If this is the case, then the Chebyshev solution can be found using the Remez algorithm.³ Mathematical details and representative examples are available elsewhere.⁴

References

1. PREIS, D.: 'Least-squares time-domain deconvolution for transversal-filter equalizers'. Electron. Lett., 1977, 13, pp. 356-357
2. PREIS, D.: 'Envelope-constrained time-domain deconvolution for transversal-filter equalizers', Electron. Lett., 1978, 14, pp. 37-38.
3. CHENEY, E. W.: 'Introduction to approximation theory' (McGraw-Hill, New York, 1966)
4. BUNKS, C., PREIS, D.: 'Minimax time-domain deconvolution for transversal-filter equalizers', IEEE International Conference on Acoustics, Speech and Signal Processing, April 1980, pp. 943-946

THREE ALGORITHMS FOR THE DESIGN
OF TRANSVERSAL-FILTER EQUALIZERS

D. Preis
Tufts University
Department of Electrical Engineering
Medford, Massachusetts 02155
(617) 628-5000 x287

C. Bunks
Massachusetts Institute of Technology
Department of Electrical Engineering
and Computer Science
Cambridge, Massachusetts 02139

- Abstract -

This report contains the equations needed to implement three different equalizer design algorithms. The equations are presented in matrix form and are suitable for digital computation. The first design, which is based on the Chebyshev or minimax error criterion, is accomplished using an iterative procedure known as the Ascent algorithm. The second design is a direct, least-square minimization of the cumulative equalization error by fast Toeplitz matrix inversion. The third design technique attempts to enforce specific constraints on the envelope of the equalized response. If an envelope-constrained solution is feasible, then it can be found iteratively beginning with the least-square-error solution. Computationally, the least-square minimization is the fastest while the envelope-constrained design is the slowest. The minimax design offers the best error control.

Submitted for presentation at The 1981 International Symposium on
Circuits and Systems, Chicago, Illinois, April 27-29.

Suggested Session: Computer-Aided Design, Active Filters, or
Digital Filters

1. Introduction. When a transversal filter (tapped delay line) is used as an equalizer for a discrete-time system, it is necessary to compute the N tap weights of the filter by deconvolution. Because the operation of discrete-time convolution generates more equations than unknowns, deconvolution generally will be inexact and, consequently, equalization will be approximate. Approximate solutions to this equalizer design problem depend on both the error criterion and constraints imposed when deconvolving. This report presents three different deconvolution algorithms to compute the N tap weights of a transversal-filter equalizer: (1) minimax design, (2) least-square-error design, and (3) envelope-constrained design.

2. Minimax Design. The basis for the minimax design is a uniform minimization of maximum deconvolution errors. Computationally, the Ascent algorithm is used to determine the N tap weights of the filter and this requires inversion of two N+1 by N+1 matrices followed by a finite number of elementary exchange operations.

Consider the discrete-time description of equalization given by the convolutional summation.

$$g(m) = \sum_{n=1}^m h(m-n+1) f(n), \quad (1)$$

$$m = 1, 2, 3, \dots, M$$

$$n = 1, 2, 3, \dots, N$$

where g is a specified M element sequence representing the desired equalized impulse response, f is an N element sequence composed of the unknown transversal-filter tap weights, and h is a known M - N + 1 element sequence which represents the impulse response of the linear,

time-invariant system requiring equalization. It is assumed, for convenience, that the time interval between adjacent sequence elements is unity. In all practical cases, the sequence h has more than one element so that $M > N$ and eqn. 1 generates an overspecified system of M linear equations in N unknowns. Since an overspecified system such as this may not have an exact solution, it is appropriate to consider approximate solutions. A useful measure of the approximation error (or equalization error) corresponding to a specific set of N tap weights f is the M element error sequence r defined by

$$r(m) = g(m) - \sum_{n=1}^m h(m-n+1) f(n), \quad (2)$$

$m = 1, 2, 3, \dots M$
 $n = 1, 2, 3, \dots N$

Comparison of eqn. 2 with eqn. 1 will indicate that the closer each numerical value $r(m)$ is to zero, the better the approximate solution becomes. If, for example, $r(m) = 0$ for $m = 1, 2, 3, \dots M$ then the N tap weights satisfy eqn. 1 exactly.

The intent here is to outline a procedure for computing those N tap weights f which both minimize and bound all the elements of the error sequence r , that is,

$$|r(m)| \leq \epsilon, \quad (3)$$

$m = 1, 2, 3, \dots M$

where ϵ is the smallest possible non-negative number. This solution is known as the Chebyshev (or minimum-maximum-error) approximate solution. There are $N + 1$ unknowns for a given equalizer design, namely, the N tap weights and ϵ . Intuitively, these unknown quantities can be found by solving an appropriate set of $N + 1$ linear equations in

$N + 1$ unknowns. It turns out that this set is, in fact, a subset of the M equations specified by eqn. 1. Assuming $\epsilon > 0$, the Chebyshev solution to these $N + 1$ equations is characterized by the following two properties. First,

$$r(m_i) = \sigma(m_i)\epsilon \quad (4)$$

for $N + 1$ of the M elements of the error sequence given by eqn. 2, where $\sigma(m_i)$ equals $+1$ or -1 and the subscript $i = 1, 2, 3, \dots, N + 1$ denotes $N + 1$ specific integer values of m from the total set of M values. Second, the remaining $M - (N + 1)$ elements of r will satisfy eqn. 3. One systematic search for the appropriate set of $N + 1$ linear equations and the subsequent solution for the N tap weights f and maximum error ϵ is known as the Ascent algorithm.¹ The equations required to implement this algorithm follow.

Initiate the search by selecting any subset of $N + 1$ of the M equations given by eqn. 2. Using eqn. 4 write the result in the following matrix form,

$$\begin{bmatrix} g(m_1) \\ g(m_2) \\ g(m_3) \\ . \\ . \\ . \\ g(m_{N+1}) \end{bmatrix} = \begin{bmatrix} \sigma(m_1) & h(m_1) & h(m_1-1) & \dots & h(m_1-N+1) \\ \sigma(m_2) & h(m_2) & h(m_2-1) & \dots & h(m_2-N+1) \\ \sigma(m_3) & h(m_3) & h(m_3-1) & \dots & h(m_3-N+1) \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ \sigma(m_{N+1}) & h(m_{N+1}) & h(m_{N+1}-1) & \dots & h(m_{N+1}-N+1) \end{bmatrix} \begin{bmatrix} \epsilon \\ f(1) \\ f(2) \\ . \\ . \\ . \\ f(N) \end{bmatrix} \quad (5)$$

where the subscripts associated with the index m denote those specific $N + 1$ integer values of m selected from the M available values.

Each of the $N + 1$ values of σ appearing in eqn. 5 is either +1 or -1 and is specified by the relation

$$\sigma(m_i) = \text{sgn} [\theta(m_i)], \quad (6)$$

$$i = 1, 2, 3, \dots, N + 1$$

where the numerical value of each of the $N + 1$ elements of θ is found by inverting the following matrix equation,

$$\begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ h(m_1) & h(m_2) & h(m_3) & \dots & h(m_{N+1}) \\ h(m_1-1) & h(m_2-1) & h(m_3-1) & \dots & h(m_{N+1}-1) \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ h(m_1-N+1) & h(m_2-N+1) & h(m_3-N+1) & \dots & h(m_{N+1}-N+1) \end{bmatrix} \begin{bmatrix} \theta(m_1) \\ \theta(m_2) \\ \theta(m_3) \\ \cdot \\ \cdot \\ \cdot \\ \theta(m_{N+1}) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix} \quad (7)$$

The existence of the Chebyshev solution requires that zero be contained in the convex hull of the $N + 1$ vectors whose N components are the elements of the $N + 1$ columns appearing below the top row of the square matrix in eqn. 7. This condition is satisfied if a linear combination of these $N + 1$ vectors can be found which yields the null (zero) vector and if those $N + 1$ scalars θ , used to form the linear combination, sum to unity. Equation 7 is a mathematical statement of these two requirements. The algebraic sign of each of the $N + 1$ scalars θ determines whether the error ϵ , associated with each linear equation in eqn. 5, is positive or negative.

Write the solution to eqn. 5 in matrix form as,

$$\begin{bmatrix} c(0,0) & c(0,1) & \dots & c(0,j) & \dots & c(0,N) \\ c(1,0) & c(1,1) & \dots & c(1,j) & \dots & c(1,N) \\ \cdot & \cdot & & \cdot & & \cdot \\ \cdot & \cdot & & \cdot & & \cdot \\ \cdot & \cdot & & \cdot & & \cdot \\ c(k,0) & c(k,1) & \dots & c(k,j) & \dots & c(k,N) \\ \cdot & \cdot & & \cdot & & \cdot \\ \cdot & \cdot & & \cdot & & \cdot \\ \cdot & \cdot & & \cdot & & \cdot \\ c(N,0) & c(N,1) & \dots & c(N,j) & \dots & c(N,N) \end{bmatrix} \begin{bmatrix} g(m_1) \\ g(m_2) \\ \cdot \\ \cdot \\ \cdot \\ g(m_{N+1}) \end{bmatrix} = \begin{bmatrix} \epsilon \\ f(1) \\ \cdot \\ \cdot \\ \cdot \\ f(N) \end{bmatrix} \quad (8)$$

The matrix multiplication prescribed by eqn. 8 yields the initial set of numerical values of the N tap weights f and error ϵ . Insert these computed tap weights in eqn. 2 and examine each element of the error sequence r to verify that $|r(m)| \leq \epsilon$ for $m = 1, 2, 3, \dots, M$. If this is true then eqn. 3 is satisfied and the Chebyshev solution has been found. No further computations are necessary.

If eqn. 3 is not satisfied then the following iterative exchange procedure is required. Select that specific integer $m = p$ for which $|r(m)|$ is maximum and let

$$\mu = \text{sgn} [r(p)] \quad (9)$$

Next, calculate the $N + 1$ elements of sequence $a(j)$ defined by

$$a(j) = \mu c(0,j) + \sum_{k=1}^N h(p - k + 1) c(k, j), \quad (10)$$

$$j = 0, 1, 2, 3, \dots, N.$$

Now select the integer q so that

$$b(q) = \frac{ua(q)}{c(0, q)}, \quad (11)$$

$$q = 0, 1, 2, 3, \dots N$$

is a maximum. Next replace the $N + 1$ element column $c(k, q)$ in eqn. 8 with $c'(k, q)$ where

$$c'(k, q) = \frac{c(k, q)}{a(q)}, \quad (12)$$

$$k = 0, 1, 2, 3, \dots N$$

and replace all other elements $c(k, j)$ in eqn. 8 for which $j \neq q$ with $c'(k, j)$ where

$$c'(k, j) = c(k, j) - a(j)c'(k, q), \quad (13)$$

$$k = 0, 1, 2, 3, \dots N$$

$$j = 0, 1, 2, 3, \dots N$$

$$(j \neq q)$$

Finally, replace $g(q)$ with $g(p)$ in eqn. 8. Recompute the error ϵ , and the sequences f and r using eqn. 8 and eqn. 2. As before, verify that $|r(m)| \leq \epsilon$ for $m = 1, 2, 3, \dots M$. If eqn. 3 is not satisfied then repeat the foregoing exchange procedure by selecting a new value of p and returning to eqn. 9.

Note that although $M - (N + 1)$ exchange operations are possible, the Chebyshev solution usually is found with fewer iterations than this. The maximum error ϵ will increase (ascend) with each successive iteration.

In certain applications, h and g are continuous-time functions rather than finite sequences of sampled values as in eqn. 1. If this is the case, then the Chebyshev solution can be found using the Remez algorithm.¹ Mathematical details and representative examples are available elsewhere.²

3. Least-Square-Error Design. The least-square-error design is based on minimization of sum of the squares of the M individual deconvolution errors $r(m)$ as defined in eqn. 2. Minimization of this cumulative error is different from the minimax design which uniformly and simultaneously minimizes each individual deconvolution error $r(m)$. Very efficient computation of the least-square-error solution is possible using fast Toeplitz matrix inversion.

A matrix representation of the convolutional summation given in eqn. 1 is

$$\begin{bmatrix}
 h(1) & 0 & \dots & 0 & 0 \\
 h(2) & & & & \vdots \\
 \vdots & & & & \vdots \\
 h(N) & & & h(1) & 0 \\
 \vdots & & & h(2) & \vdots \\
 \vdots & & & \vdots & \vdots \\
 h(M-N) & & & \vdots & \vdots \\
 h(M-N+1) & & & h(N) & \vdots \\
 0 & & & \vdots & \vdots \\
 \vdots & & & \vdots & \vdots \\
 0 & 0 & \dots & 0 & h(M-N) \\
 & & & & h(M-N+1)
 \end{bmatrix}
 \begin{bmatrix}
 f(1) \\
 f(2) \\
 \vdots \\
 f(N)
 \end{bmatrix}
 =
 \begin{bmatrix}
 g(1) \\
 g(2) \\
 \vdots \\
 g(M)
 \end{bmatrix}
 \quad (14)$$

and this equation can be written as

$$\mathbf{H} \mathbf{f} = \mathbf{g} \quad (15)$$

where $\mathbf{f}$ and $\mathbf{g}$ now are understood to be column vectors. The elements of $\mathbf{H}$ are known and $\mathbf{g}$ is specified. The unknown tap-weight vector

is f . Thus, eqn. 15 represents an overspecified system of M linear equations in N unknowns, that is $M > N$. The least-square-error solution $\hat{f}$ to eqn. 15 is³

$$\hat{f} = Sg \quad (16)$$

where $S = (H' H)^{-1} H'$. The prime symbol denotes transpose. Note that $H' H$ is an $N \times N$ symmetric Toeplitz matrix whose elements are $\tau_{ij} = \tau_{|i - j|} = \tau_k$ where

$$\tau_k = \sum_{n=1}^{M-N+1-k} h(n+k) h(n). \quad (17)$$

A rapid inversion algorithm is available⁴ to compute $(H' H)^{-1}$. After this inverse is evaluated, the least-square-error tap weights $\hat{f}$ are found using eqn. 16. The resultant mean-square error is $[H\hat{f} - g][H\hat{f} - g]/M$, and it is this quantity that has been minimized. However, the individual deconvolution errors $r(m)$ are unconstrained. These errors can be evaluated quantitatively by replacing $f(n)$ with $\hat{f}(n)$ in eqn. 2. If the least-square-error solution $\hat{f}$ yields unacceptably large individual errors $r(m)$, it may be possible to reduce them using the envelope-constrained design procedure discussed in the following section. With this algorithm somewhat larger mean-square error is accepted in return for control and reduction of larger individual deconvolution errors $r(m)$.

The least-square-error solution $\hat{f}$ also can be found using discrete Fourier transform methods. Computationally, these two different methods are comparable. One requires matrix algebra and Toeplitz

inversion while the other relies on the availability of FFT routines and requires algebraic operations with complex numbers.

4. Envelope-Constrained Design. The envelope-constrained design algorithm is an iterative procedure which begins with the least-square-error design. The least-square-error tap weights $\hat{f}$ are then systematically re-adjusted to the new values $\tilde{f}$ in an attempt to satisfy specific constraints imposed on the envelope of the equalized response $g(m)$.

Assume that each $g(m)$ is bounded above by $a(m)$ and below by $b(m)$. Next require that the envelope-constrained tap weights $\tilde{f}$ satisfy

$$\mathbf{H}\tilde{f} = \tilde{g} \quad (18)$$

where

$$b(m) \leq \tilde{g}(m) \leq a(m), \quad (19)$$

$$m = 1, 2, 3, \dots, M.$$

In this way, the equalized response $\tilde{g}$ is constrained to lie within the tolerance envelope prescribed by sequences a and b ; and $\tilde{g}$ is the envelope-constrained approximation to g . If such a solution is feasible then $\tilde{f}$ can be found using the following iterative adjustment procedure.⁵

For the k -th iteration eqn. 15 is written as

$$\mathbf{H}f_k = g_k, \quad (20)$$

where g_k is the k -th constrained response vector corresponding to the k -th iteration of the tap weight vector f_k . For $k = 0$, choose the least-square-error solution $f_0 = \hat{f} = \mathbf{S}g$ then compute g_0 from eqn. 20 as $g_0 = \mathbf{H}f_0$. The tap weight vector is adjusted or "stepped"

using the relation

$$f_{k+1} = f_k + \Delta f_k, \quad (21)$$

$$k = 0, 1, 2, \dots$$

where Δf_k is given by

$$\Delta f_k = S \Delta g_k \quad (22)$$

and where Δg_k itself depends on the following combination of the errors

$$\Delta g_k(m) = \begin{cases} 0 & \text{if } b(m) \leq g_k(m) \leq a(m) \\ -\alpha[g_k(m) - a(m)] & \text{if } g_k(m) > a(m) \\ -\alpha[g_k(m) - b(m)] & \text{if } g_k(m) < b(m) \end{cases} \quad (23)$$

The unspecified positive constant α in eqn. 23 must be sufficiently small to ensure a stable iterative process. If a solution is feasible then to guarantee convergence it has been proved,⁶ for a similar iterative scheme, that α must be equal to or less than half the reciprocal of the norm of H .

In the special case where the envelope is collapsed, that is $a(m) = g(m) = b(m)$ for $m = 1, 2, 3, \dots, M$, iteration is not possible and the appropriate solution is the least-square-error solution $\hat{f}$. If, on the other hand, $[a(m) + b(m)]/2 = g(m)$ and $a(m) - b(m) = 2\epsilon$ for $m = 1, 2, 3, \dots, M$ where ϵ is the minimax error, then the envelope-constrained design should iterate the tap weights from their least-square-error values to the minimax-error values. In this sense, the envelope-constrained algorithm relates the least-square design to the minimax design.

5. Summary and Conclusions. When a finite-length transversal filter is used as an equalizer for a discrete-time system generally it is only possible to equalize the system approximately.⁷ Three different algorithms have been presented to compute the tap weights of the filter subject to different error criteria and constraints. Each of these design algorithms was presented in matrix form suitable for digital computation. While the least-square-error design is fast, it may result in large, unconstrained equalization (deconvolution) errors. The minimax design provides optimum control of individual deconvolution errors. The envelope-constrained algorithm is an intermediate design which offers greater individual error control at the expense of larger mean-square error. For practical equalizer design, it is probably worthwhile to evaluate all three designs and then select the one that provides the most suitable approximate equalization.

6. References.

1. E. W. Cheney, Introduction to Approximation Theory. New York: McGraw-Hill, 1966, Chapter 3.
2. C. Bunks and D. Preis, "Minimax Time-Domain Deconvolution for Transversal Filter Equalizers," IEEE Conference Record, ICASSP-80, pp. 943-946, April 1980.
3. D. Preis, "A Minimax Design for Transversal Filter Equalizers," Electron. Lett., vol. 13, pp. 356-357, June 1977.
4. D. Preis, "The Toeplitz Matrix: Its Occurrence in Antenna Problems and a Rapid Inversion Algorithm," IEEE Trans. Antennas and Propag., vol. AP-20, pp. 204-206, March 1972.

5. D. Preis, "Envelope-Constrained Time-Domain Deconvolution for Transversal-Filter Equalizers," Electron. Lett., vol. 14, pp. 37-38, Jan. 1978.
6. R. J. Evans, T. E. Fortmann, and A. Cantoni, "Envelope-Constrained Filters," IEEE Trans., IT-23, pp. 421-444, 1977.
7. D. Preis, "Audio Signal Processing with Transversal Filters," IEEE Conference Record, ICASSP-79, pp. 310-313, April 1979.

MINIMAX EQUALIZATION FOR TRANSVERSAL FILTERS

A THESIS

Submitted by

CAREY DAVID BUNKS

In partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

TUFTS UNIVERSITY

June 1980

ACKNOWLEDGMENTS

I would like to recognize the fact that I have been extremely fortunate during my stay at Tufts. I am particularly grateful to my advisor, Dr. Douglas Preis, for his outstanding guidance throughout the course of my work. The professional and academic maturity which I have gained through close association with Dr. Preis is without measurable value.

I would also like to acknowledge Dr. Michael Brown and Dr. Howard Hunter. The avuncular relationship I have had with both has been a rare and delightful experience, and I will remember them with great affection.

ABSTRACT

ABSTRACT

The time-domain response of a linear system in cascade with a filter is determined by the convolution of their respective impulse responses. Thus, if the system to be equalized is causal with impulse response $h(t)$, and there is a particular desired response $g(t)$, then the filter's impulse response $f(t)$ must satisfy,

$$\int_0^t f(x)h(t-x)dx = g(t). \quad (1)$$

Since the unknown filter response appears under the integral sign, (1) is an integral equation for $f(t)$. The solution of (1) requires that an inverse convolution or deconvolution be performed to find $f(t)$ when $h(t)$ and $g(t)$ are specified.

Deconvolution is accomplished by direct solution of the time-domain convolutional integral equation (1). Previous solutions to this integral equation have been found by the method of collocation, or by minimizing the square of the solution error, or by using envelope-constrained procedures. The proposed minimax or Chebyshev solution minimizes the maximum solution error and, thereby, provides a means to uniformly control deconvolution errors.

The minimax solution is found using an iterative procedure known as the second algorithm of Remez.

TABLE OF CONTENTS

TABLE OF CONTENTS

	Page
ACKNOWLEDGMENTS	2
ABSTRACT	4
TABLE OF CONTENTS	6
LIST OF FIGURES	9
LIST OF TABLES	11
1. INTRODUCTION	13
1.1 Equalization	13
1.2 Transversal Filters	15
1.3 Approximate Solutions to Deconvolution Problems	16
1.4 Approximation Theory	18
1.5 Minimax Approximation	23
2. MINIMAX APPROXIMATION OF OVERSPECIFIED SYSTEMS OF LINEAR EQUATIONS	24
2.1 Systems of Linear Equations	24
2.2 Solving Two Equations in One Unknown	26
2.3 Solving M Equations in One Unknown	30
2.4 Characterization of the Minimax Approximation	33
3. USING A TRANSVERSAL FILTER AS A DISCRETE TIME-DOMAIN EQUALIZER	35
3.1 Convolution and the Transversal Filter	35
3.2 Solving the Equalization Problem	37
4. DISCRETE-TIME ALGORITHM	40
4.1 Ascent Algorithm for One Unknown	40
4.2 Generalization of the Ascent Algorithm to N Unknowns	42
4.2.1 Forcing Zero to Belong to the Convex Hull	42
4.2.2 Finding the Intersection of the N+1 Error Equations	45
4.2.3 Exchange Theorem	48
4.3 Manipulation of the Inverse Matrix	52
4.4 Convergence	55
4.5 Restrictions on the Ascent Algorithm	57
5. USING THE TRANSVERSAL FILTER AS A CONTINUOUS TIME EQUALIZER	58
5.1 Statement of the Equalization Problem	58
5.2 Linearization of the Continuous Error Function	60

	Page
6. REMEZ ALGORITHM	61
6.1 Process of the Remez Algorithm	61
6.2 Operational Details of the Remez Algorithm	64
6.3 Convergence of the Remez Algorithm	68
7. NUMERICAL EXAMPLES AND SUGGESTED FUTURE RESEARCH	73
7.1 Examples	73
7.2 Future Research	78
BIBLIOGRAPHY	80
APPENDIX	83
A.1 Proof That the Convex Hull Contains Zero	83
A.2 Property of the Convex Hull of A Set	86
A.3 Proof That Minimax Solutions are Determined by Subsets of N+1 Elements	87
A.4 Alternation Theorem	89
A.5 Program Listings	93
A.5.1 Ascent Algorithm	93
A.5.2 Remez Algorithm	96

LIST OF FIGURES

LIST OF FIGURES

<u>Figure</u>		<u>Page</u>
1.1	Equalization Scheme	13
1.2	Convolutional Transformation	14
1.3	Block Diagram of Transversal Filter	15
1.4	Approximation Problem in Deconvolution	17
1.5	Envelope Constrained Deconvolution	22
2.1	Solution of a Linear Equation in One Unknown	27
2.2	Approximation to Two Linear Equations in One Unknown	28
2.3	Approximation to Four Linear Equations in One Unknown	31
2.4	Properties of a Convex Function	32
2.5	Convex Sets	33
3.1	Transversal Filter	36
4.1	Ascent Algorithm for One Unknown	41
6.1	Example of Multiple Exchanges in Ascent Algorithm	62
6.2	Details of Exchanging Subsets of $N+1$ Points in the Remez Algorithm (for $N=4$)	66
6.3	Flow Chart for the Remez Algorithm	69
7.1	Equalization of Gaussian Function to $\sin(t)/t$	74
7.2	Equalization of Raised Cosine Function to Triangular Pulse	76
A.1	Proof of the Convex Hull Containing Zero	84

LIST OF TABLES

LIST OF TABLES

<u>Table</u>		<u>Page</u>
7.1	Tap-Weights and Error for f_6 and f_8	73
7.2	Tap-Weights and Error for f_8 and f_{12}	77

MINIMAX EQUALIZATION FOR TRANSVERSAL FILTERS

1. INTRODUCTION

1.1 Equalization

Equalization is a practical and important problem in communication systems engineering. The purpose of equalization is to compensate for the linear distortions an electrical signal undergoes when passed through a linear system or communication channel. This report will briefly describe current time-domain equalization techniques, and then present a new approach to the problem called minimax time-domain deconvolution.

Assume that the system or channel to be equalized has impulse response $h(t)$ and that after equalization the desired impulse response is $g(t)$. Equalization can be accomplished using a filter with impulse response $f(t)$ in cascade with $h(t)$ as shown in Fig. 1.1.

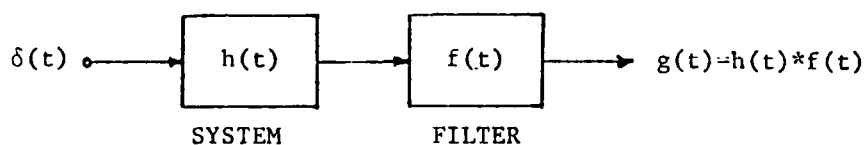


Figure 1.1 Equalization Scheme

From linear system theory the combined response $g(t)$ is given by the following convolutional integral.

$$g(t) = \int_0^t h(x)f(t-x)dx = h(t)*f(t) \quad (1.1)$$

The integral in (1.1) is evaluated between zero and t because $h(t)$ and $f(t)$ are assumed to be causal.

Mathematically, the continuous functions on an interval $[a,b]$ represent an abstract vector space and the operator $h*(.)$, representing convolution, is a linear transformation on the space of continuous functions. Equation (1.1) can be viewed as a linear transformation from the space of filter responses $f(t)$ to the space of desired responses $g(t)$ as shown in Fig. 1.2.

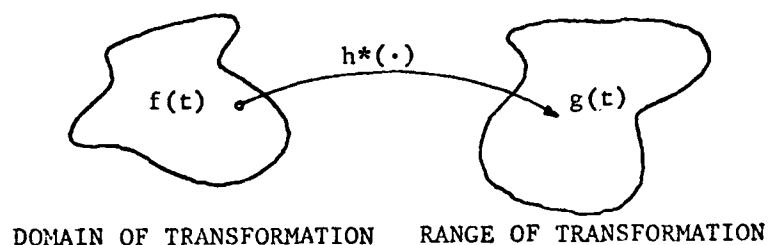


Figure 1.2 Convolutional Transformation

Although $[a,b]$ could be $(-\infty, \infty)$, in practice this interval is finite.

In Fig. 1.2 there is a particular vector $g(t)$ in the range which is the most desirable response. The process of searching for the appropriate vector $f(t)$ in the domain to satisfy (1.1) is known as

deconvolution or inverse-convolution and is an important, contemporary engineering problem. Thus, equalization becomes a deconvolution problem.

1.2 Transversal Filters

This report will not concern itself with the space of all possible filter functions $f(t)$, only with a particular class of filters known as transversal filters or tapped delay lines. A transversal filter is composed of N serial delay sections whose outputs are weighted by scalar multipliers (tap-weights). All these delayed and weighted signals are then added together to produce the filter output. Figure 1.3 shows a block diagram of a transversal filter.

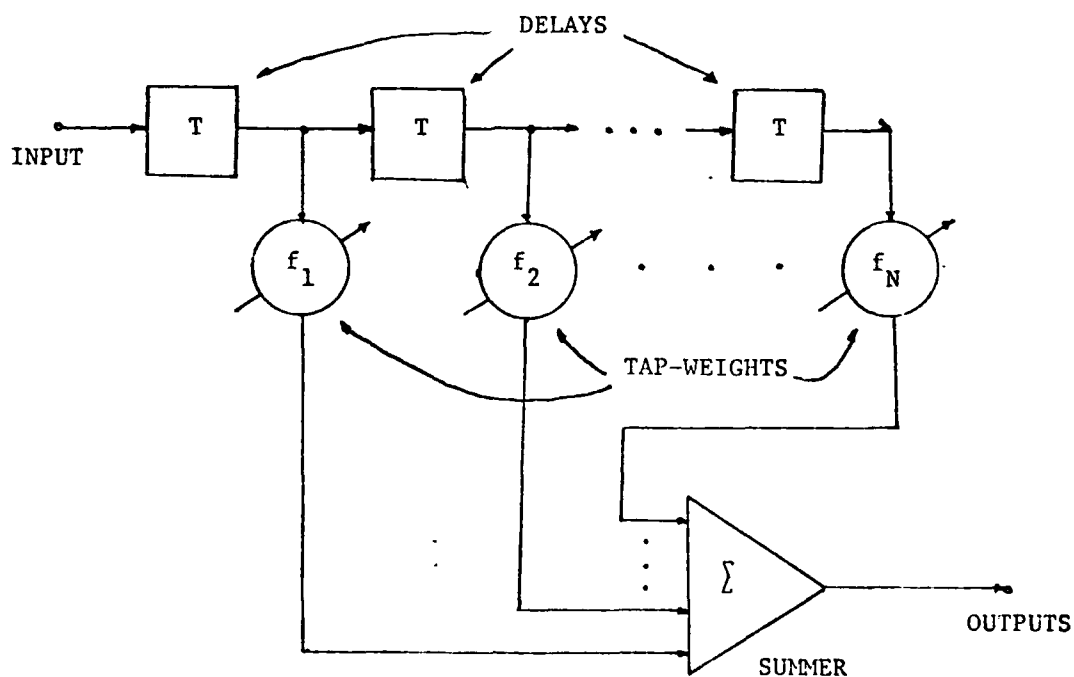


Figure 1.3 Block Diagram of Transversal Filter

One mathematical representation of the impulse response for a transversal filter is a series of N contiguous rectangular pulses of heights f_n and of width T . However, an alternate form which greatly simplifies the following discussion is the Dirac Comb,

$$f(t) = \sum_{n=1}^N f_n \delta(t-nT) \quad (1.2)$$

where the f_n are the filter tap weights and $\delta(t)$ is the unit impulse function.

Substituting the Dirac Comb representation (1.2) into (1.1) and integrating yields the summation,

$$g(t) = \sum_{n=1}^N f_n h(t-nT) \quad (1.3)$$

for which $h(t)$ is a given system or channel impulse response and the f_n are variable scalar coefficients. The deconvolution problem is concerned with adjusting the N tap weights f_n to force equality in (1.3).

1.3 Approximate Solutions to Deconvolution Problems

The linear transformation described in (1.1) is well known in mathematical and physical sciences as an integral transformation. Finding the vector $f(t)$ which satisfies the transformation for a particular $g(t)$ is often difficult or impossible. The difficulties are present because often the image space of the transformation is smaller than the range space. This means that as the transformation operates on each vector $f(t)$ in the domain space, the set of vectors

$g(t)$ produced by the transformation is only a subset of the set of all possible vectors (see Fig. 1.4).

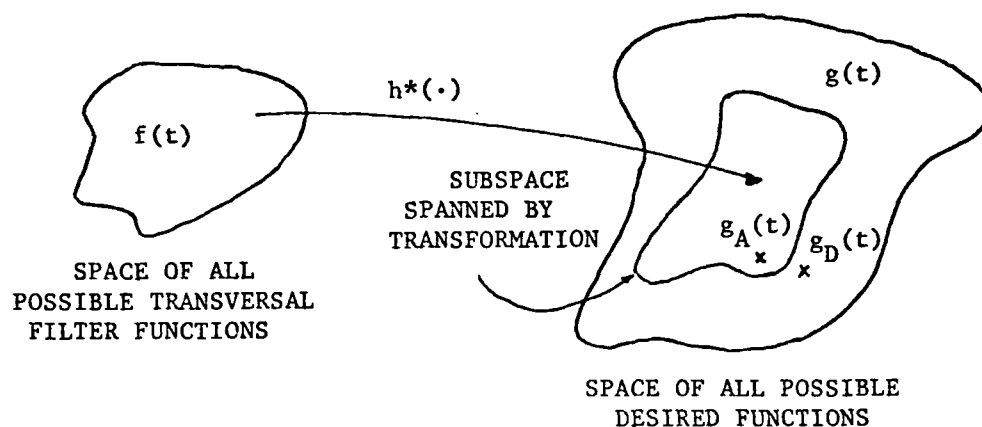


Figure 1.4 Approximation Problem in Deconvolution

If a particular desired response $g_D(t)$ is not mapped by the transformation as it ranges over all possible filter functions (see Fig. 1.4) then it is desirable to find a vector $g_A(t)$ in the image space which is in some way close to $g_D(t)$. Thus, $g_A(t)$ is an approximation to $g_D(t)$ and there exists an $f(t)$ in the domain which is mapped to $g_A(t)$ by the transformation $h^*(\cdot)$.

The equality stated by (1.3) is satisfied only if the choice of $g(t)$ can be written as a linear combination of the $h(t-nT)$. Deconvolving (1.3) to determine the values of the N tap weights f_n is straightforward only when $g(t)$ lies in the span of the N time shifted functions $h(t-nT)$. This report will investigate possible approximate solutions for the more general case when $g(t)$ is not spanned by the right-hand side of (1.3).

1.4 Approximation Theory

Referring to Fig. 1.4 it is desired to pick an approximate vector $g_A(t)$ so that the distance between it and the desired vector $g_D(t)$ is small. To determine a best approximation it is first necessary to define a measure of closeness between vectors of an abstract vector space. For example, it is not clear what the distance between the continuous functions $\cos(t)$ and $\sin(t)$ is on the interval $[0, 2\pi]$.

Intuitively, a distance operator $d(\cdot, \cdot)$ on pairs of vectors must have certain properties. The operator assigns a real number to the vector pair such that the distance between the two vectors is greater than zero unless the two vectors are equal. If the vectors are equal, then the operator assigns zero to be the distance. Furthermore, the distance operator must be independent of direction and obey the triangle inequality.

Summarizing for the vectors x, y, z

$$\begin{aligned}
 d(x, y) &\geq 0 \quad , \quad x \neq y \\
 d(x, x) &= 0 \\
 d(x, y) &= d(y, x) \\
 d(x, z) &\leq d(x, y) + d(y, z)
 \end{aligned}
 \tag{1.4}$$

A well-known distance operator on the real numbers is the absolute value function which obeys the properties of (1.4) for any real numbers x, y, z . For abstract vector spaces a general class of operators called norms have the properties of a distance operator. A norm assigns a real number to a vector and is denoted by $||\cdot||$. The norm has the

following properties for any vectors x, y

$$\begin{aligned}
 ||x|| &= 0, \quad x = 0 \\
 ||x|| &> 0, \quad x \neq 0 \\
 ||ax|| &= |a| \cdot ||x||, \quad a \in \mathbb{R} \\
 ||x+y|| &\leq ||x|| + ||y||
 \end{aligned} \tag{1.5}$$

It can easily be shown that $||x-y||$ is a distance operator on the pair of vectors x, y .

Referring to Fig. 1.4 and using the norm as the measure of distance, it is desired to minimize the distance between the desired response $g_D(t)$ and the approximate response $g_A(t)$. For an interval $[a, b]$ this procedure is represented

$$\min ||g_D(t) - g_A(t)|| \tag{1.6}$$

and is determined by examining (1.6) for all possible $g_A(t)$.

Since $g_A(t)$ can be represented as a linear combination of the N functions $h(t-nT)$ (1.6) becomes

$$\min ||g_D(t) - \sum_{n=1}^N f_n h(t-nT)||. \tag{1.7}$$

It is convenient to represent the arguments of (1.6) and (1.7) as an error function

$$r(t) = g(t) - \sum_{n=1}^N f_n h(t-nT) \quad (1.8)$$

where a best approximation is recognized when the norm of $r(t)$ is minimized. An important point is that $r(t)$ is a function of the N tap weights f_n and that $||r(t)||$ will vary as f_n vary.

A norm that is widely used as a distance measure in the engineering sciences is the Euclidean norm. The Euclidean norm of a vector (x,y) in the Cartesian plane is $\sqrt{x^2 + y^2}$ and for two vectors, $v_1 = (x_1, y_1)$ and $v_2 = (x_2, y_2)$ the norm of the difference is

$$||v_1 - v_2|| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (1.9)$$

Equation (1.9) can be generalized to an n -dimensional space R^n .

For a vector $x = (x_1, \dots, x_n)$, $x \in R^n$

$$||x|| = \sqrt{x_1^2 + \dots + x_n^2} \quad (1.10)$$

There is an analogous representation for the Euclidean norm operating on the abstract vector space of continuous functions on an interval $[a,b]$. For $\phi(t)$ continuous on $[a,b]$

$$||\phi(t)|| = \sqrt{\frac{1}{b-a} \int_a^b \phi^2(t) dt} \quad (1.11)$$

and for $\phi_1(t)$ and $\phi_2(t)$ continuous on $[a,b]$

$$||\phi_1(t) - \phi_2(t)|| = \sqrt{\frac{1}{b-a} \int_a^b [\phi_1(t) - \phi_2(t)]^2 dt} . \quad (1.12)$$

In electrical engineering (1.11) is the RMS (root mean square) value of $\phi(t)$.

Using the Euclidean norm as defined in (1.12) it is now possible to calculate a distance between $\sin(t)$ and $\cos(t)$, the functions of the previous example,

$$||\sin t - \cos t|| = \sqrt{\frac{1}{2\pi} \int_0^{2\pi} (\sin t - \cos t)^2 dt} = 1. \quad (1.13)$$

Note that the value of the norm in (1.13) is related to the area between the curves $\sin(t)$ and $\cos(t)$ in the interval $[0, 2\pi]$. This is an important characteristic of the Euclidean norm. Minimizing the Euclidean norm is in essence minimizing the area between an approximation curve and the desired curve.

Minimization of the Euclidean norm is the approximation technique known as least squares. The characteristic feature of least squares solutions is that the area between the desired curve and the approximate curve is minimized. A significant drawback of the method is that individual errors are not bounded. Often a least squares solution has large deviations from the desired curve at certain points. For example, the Gibb's phenomenon observed with finite Fourier series solutions produces sharp spikes in the error curve at discontinuities of the desired curve.

One case where large deviations in the error curve are particularly undesirable is in digital signal detection. Since the detection scheme is often a simple level detector the presence of large deviations in the approximation curve could result in erroneous detection of information during transmission.

An approach to limiting unbounded errors in the least squares approximation scheme has been explored in the literature. The method is called envelope-constrained deconvolution and it is an iterative technique which is initiated by computing the least squares approximation. In successive iterations, the largest errors are reduced, if possible, until they are bounded by an envelope defined around the desired curve as in Fig. 1.5.

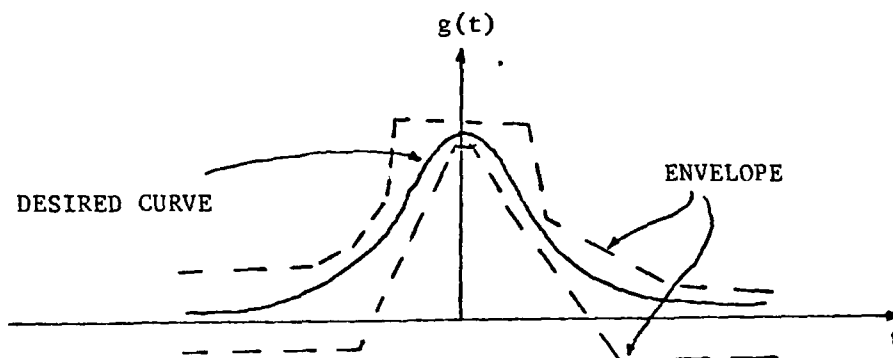


Figure 1.5 Envelope Constrained Deconvolution

Pushing errors down in one place may push errors up in other places. Generally, mean square error is greater when envelope constraints are imposed to suppress large, individual errors. The envelope-constrained approximation is a practical approach to restricting the

magnitude of the error. The difficulty is that it is not generally known how narrow the envelope can be made, and there is an important question of solution feasibility.

1.5 Minimax Approximation

Here, a different approach to the deconvolution problem is proposed, namely, the minimax solution. The minimax solution is a best bounded approximation and is derived as a characteristic of using the Chebyshev norm as a distance measure instead of the Euclidean norm.

The Chebyshev norm of a vector $x = (x_1, \dots, x_n)$ which is a member of R^n is

$$||x|| = \max_{1 \leq i \leq n} |x_i|. \quad (1.14)$$

The analogous representation of (1.14) for continuous functions on an interval $[a, b]$ is

$$||\phi(t)|| = \max_{a \leq t \leq b} |\phi(t)| \quad (1.15)$$

where $\phi(t)$ is a continuous function of $[a, b]$. The minimax approximation comes from minimizing (1.14) or (1.15).

The remainder of this report deals with further discussion of the minimax approximation and its applications to the design of transversal filter equalizers. Future reference to the norm or the use of the norm will imply the Chebyshev norm unless otherwise noted.

2. MINIMAX APPROXIMATION OF OVERSPECIFIED SYSTEMS OF LINEAR EQUATIONS

2.1 Systems of Linear Equations

Equations (2.1) and (2.2) are equivalent representations of a system of M linear equations in N unknowns. The a_{mn} and b_m are constants and the x_n are the N unknowns.

$$\sum_{n=1}^N a_{mn} x_n = b_m \quad m = 1, \dots, M \quad (2.1)$$

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ a_{M1} & a_{M2} & \dots & a_{MN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ b_M \end{bmatrix} \quad (2.2)$$

The matrix of coefficients $A = [a_{mn}]$ on the left-hand side of (2.2) is a linear transformation on the N -dimensional vectors $x = (x_1, \dots, x_N)$, $x \in R^N$ to the M -dimensional vectors $y = (y_1, \dots, y_M)$, $y \in R^M$. The vector $b = (b_1, \dots, b_M)$ on the right-hand side of (2.2) is a constant vector for which it is desired to find the vector x which under the transformation A maps to b :

$$Ax = b. \quad (2.3)$$

Equation (2.3) is equivalent to (2.2).

The existence and the number of solutions to (2.3) depends upon the relationship between the number of equations and the number of unknowns. Three possibilities exist, they are $M < N$, $M = N$, and $M > N$ where M is the number of equations and N is the number of unknowns.

For each case, the transformation A is mapping vectors from N -dimensional space R^N to M -dimensional space R^M . In the case $M < N$ the transformation maps from a larger space to a smaller space and if the rows of the matrix are linearly independent, then for any particular vector b which belongs to R^M there are always infinitely many vectors $x \in R^N$ such that b is mapped by these vectors under the transformation. Referring to Fig. 1.2 in Chapter 1, it is intuitively easy to see how a domain space of larger dimension would fill up many times over a range space of smaller dimension.

In the case $M = N$ the transformation maps from a domain space of equal dimension to the range space. From linear algebra it is known that a system of linear equations with the same number of equations as unknowns has a unique solution if the equations are linearly independent.*

Finally, for the case $M > N$ the transformation operates on a domain space which is smaller than the range space. As in Fig. 1.4 of Chapter 1 it can be seen that the transformation of the domain space does not fill up the range space. That is, the transformation of the domain is a subspace of the range. If the vector b in (2.1) - (2.3) lies in the image space of the transformation A , then a solution exists. Alternatively, if b does not lie in the image space of A then no exact solution exists. When an exact solution does not exist the best

* O'nan, H., Linear Algebra, Harcourt Brace Jovanovich, 1976

solution to (2.1) - (2.3) is found by selecting a vector which lies in the image space of A and is a closest approximation to b .

When the number of equations in a linear system exceeds the number of unknowns the system of equations is called overspecified. To find the best approximation $\hat{b}$ to the vector b the Chebyshev norm will be used as a distance measure. If ϵ is the smallest distance that can be achieved between the two vectors, then

$$\epsilon = ||b - \hat{b}|| \quad (2.4)$$

Since $\hat{b}$ is mapped by the transformation A for some choice of x (2.4) becomes

$$\epsilon = \min ||b - Ax|| \quad (2.5)$$

2.2 Solving Two Equations in One Unknown

Often only approximate solutions exist for overspecified systems of linear equations. This section and the rest of this chapter will be concerned with describing the characteristics of properties and the minimax approximation.

The linear equation $2x = 1$ has an exact algebraic solution, namely, $x = 1/2$. The equation $2x = 1$ is, of course, the most elementary case of a linear system of equations. The solution to the equation $2x = 1$ can be found graphically by defining an error equation $r_1(x)$, such that $r_1(x) = |2x - 1|$. Plotting $r_1(x)$ in Fig. 2.1 shows that the solution to the equation $2x = 1$ is found where $r_1(x)$ intersects

the x -axis. That is, when the solution has been found the error is zero. In the definition of $r_1(x)$ magnitude brackets are used since it is only the magnitude of the error which is of concern.

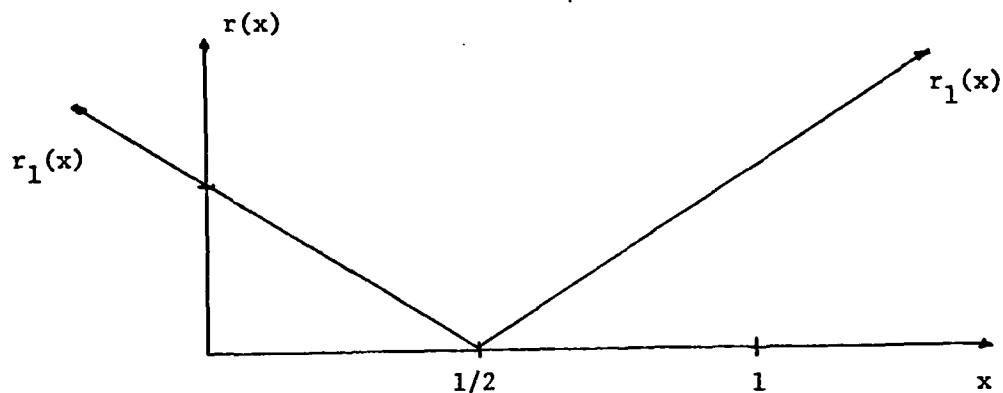


Figure 2.1 Solution of a Linear Equation in One Unknown

If a system of linear equations has more linearly independent equations than unknowns, then an exact solution does not exist. This is so, for example, with the system:

$$\begin{aligned} 2x &= 1 \\ (1/3)x &= 1 \end{aligned} \tag{2.6}$$

Each separate equation has a unique, exact solution ($1/2$ and 3 respectively) which can be found either algebraically or by the graphical method of Fig. 2.1. However, there is no solution which simultaneously satisfies both equations in (2.6) exactly.

Figure 2.2 which is a plot of the error equations $r_1(x)$ and $r_2(x)$ associated with (2.6). An intuitive approximate solution to (2.6) would be $x = 6/7$.

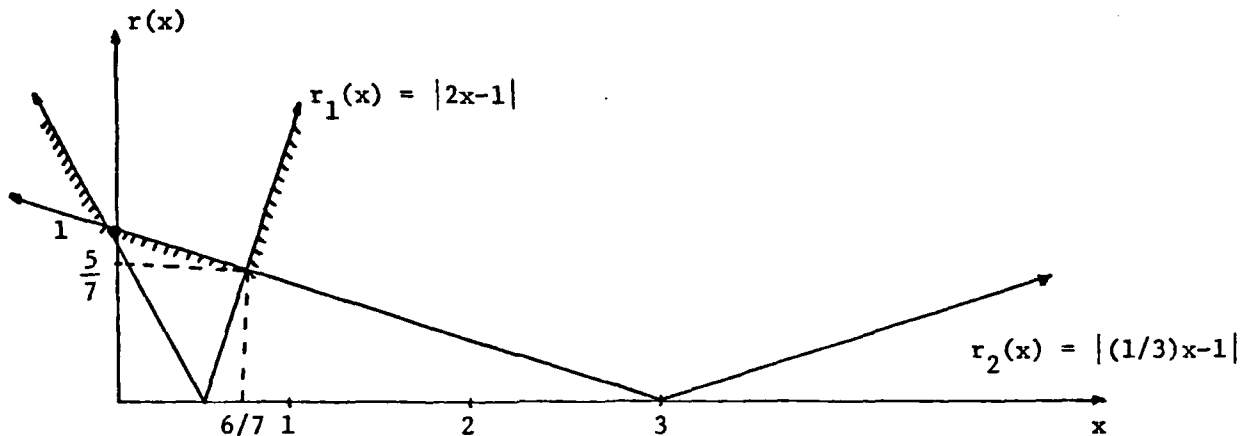


Figure 2.2 Approximation to Two Linear Equations in One Unknown

Plugging the value $x = 6/7$ into the left-hand side of (2.6) yields

$$2x = 2(6/7) = 1 \frac{5}{7}$$

$$\frac{1}{3}x = \frac{1}{3}(6/7) = 2/7 \quad (2.7)$$

Note that the proposed solution $x = 6/7$ produces values in (2.7) which are in error with respect to the right-hand side of (2.6) by $5/7$ for each equation. Furthermore, the value $x = 6/7$ is where the functions $r_1(x)$ and $r_2(x)$ intersect in Fig. 2.2 ($r_1(x)$ and $r_2(x)$ also intersect at $x=0$, however, the error produced for each equation when zero is used in the left-hand side of (2.7) is $r_1 = r_2 = 1$ which is greater than the error produced by $x = 6/7$).

Now the minimax approximation to (2.6) will be computed. The system of equations in (2.6) are of the form of (2.3) where the matrix of coefficients is $A = \begin{bmatrix} 2 \\ 1/3 \end{bmatrix}$ and the vector $b = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$. Using the norm as the distance measure between the approximation Ax and the exact solution b yields

$$||b - Ax|| = \left\| \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 2x \\ \frac{1}{3}x \end{bmatrix} \right\| = \left\| \begin{bmatrix} 1 - 2x \\ 1 - \frac{1}{3}x \end{bmatrix} \right\| \quad (2.8)$$

The minimax solution is found by minimizing the norm of (2.8). Using the Chebyshev norm and denoting ϵ as the minimax error,

$$\begin{aligned} \epsilon &= \min \left\| \begin{bmatrix} 1 - 2x \\ 1 - \frac{1}{3}x \end{bmatrix} \right\| = \min \left[\max \left\{ |1 - 2x|, \left| 1 - \frac{1}{3}x \right| \right\} \right] \\ &= \min \left[\max \{ r_1(x), r_2(x) \} \right]. \end{aligned} \quad (2.9)$$

The function $\max \{ r_1(x), r_2(x) \}$ in (2.9) is the crosshatched portion of the graph in Fig. 2.2, and the minimum value of this function is found when $x = 6/7$ as shown. The value of the error when $x = 6/7$ in (2.9) is $\epsilon = 5/7$.

The intuitive approximation $x = 6/7$ was chosen because it was the value which minimized the error for both equations in (2.6). Note that moving either to the left or right of $x = 6/7$ in Fig. 2.2 increases the error for one of the two error equations. The minimax approximation is equivalent to the intuitive approximation, however, the procedure for finding the minimax approximation can be formulated graphically.

2.3 Solving M Equations in One Unknown

The graphical technique for finding the minimax approximation to a set of two equations in one unknown can be expanded to a larger set of equations. Consider, for example, the set of equations

$$\begin{aligned} 2x &= 1 \\ 1/3x &= 1 \\ 5/2x &= 5 \\ 5/8x &= 5/2 \end{aligned} \tag{2.10}$$

The associated error equations are

$$\begin{aligned} r_1(x) &= |2x - 1| \\ r_2(x) &= |\frac{1}{3}x - 1| \\ r_3(x) &= |\frac{5}{2}x - 5| \\ r_4(x) &= |\frac{5}{8}x - \frac{5}{2}| \end{aligned} \tag{2.11}$$

Defining an error vector $r(x)$ which has the four components $r_1(x)$, $r_2(x)$, $r_3(x)$, $r_4(x)$ it is possible to find the norm of $r(x)$ from the graph of the error functions r_1 , r_2 , r_3 , r_4 . The norm of $r(x)$ is

$$||r(x)|| = \max \{r_1(x), r_2(x), r_3(x), r_4(x)\}$$

and it is the crosshatched region of Fig. 2.3. From Fig. 2.3 it is easy to pick out the minimum value of (2.12) and the value of x associated to it. These values are $\min ||r(x)|| = 5/3$ and $x = 4/3$. Thus $x = 4/3$ is the minimax approximation to (2.10).

Several characteristics of the graphical solution to (2.10) are important and deserve further discussion. First, the minimax solution $x = 4/3$ is found at the intersection of two of the error equations in (2.11). Referring to Fig. 2.3 the particular equations are $r_1(x) = |2x-1|$ and $r_4(x) = |\frac{5}{8}x - \frac{5}{2}|$.

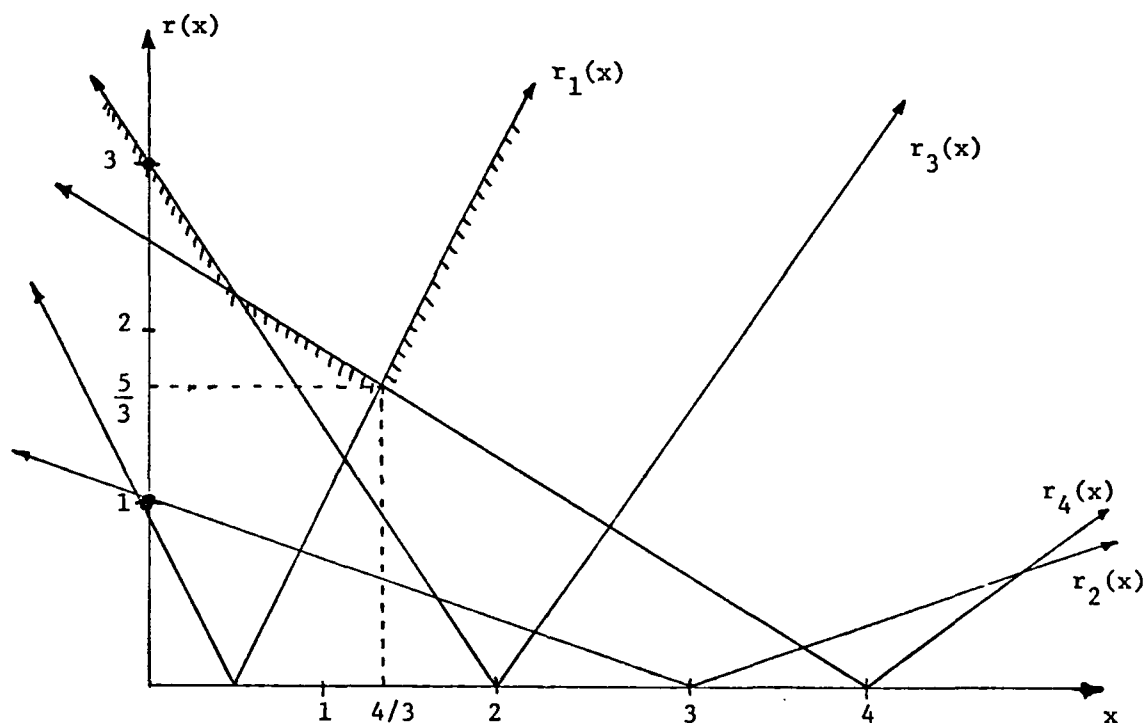


Figure 2.3 Approximation to Four Linear Equations in One Unknown

Note that all other error equations lie below the intersection of $r_1(x)$ and $r_4(x)$ at $x = 4/3$. Thus, the maximum error has been minimized for all four equations. If an approximation is chosen slightly to the left or right of $x = 4/3$ then either the value of r_1 increases or the value of r_4 increases. For $x = 4/3$ the error for each equation has been bounded and minimized.

Second, the solution occurs at the intersection of two lines which are of opposite slope (see Fig. 2.3). This is a very important property and is related to the fact that the Chebyshev norm of $r(x)$ is a convex function.

A convex function is defined as being a function $\phi(x)$ such that for any three values of x , $a < b < c$, $\phi(b) \leq \theta\phi(a) + (1-\theta)\phi(c)$ where θ varies from zero to one. Figure 2.4 shows a convex function.

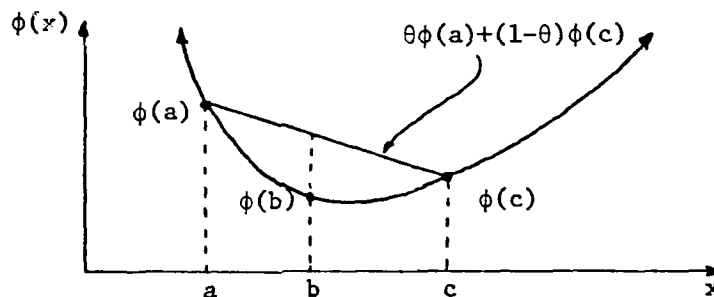


Figure 2.4 Properties of a Convex Function

A convex function always has a global minimum. Since the norm of $r(x)$ is a convex function (see Fig. 2.3) a global minimum can be found for it, and the minimax approximation is guaranteed to exist.

A change in the sign of the slope of $||r(x)||$ at the minimax solution is an indication that a global minimum of the convex function $||r(x)||$ has been found. This can be seen by moving along $||r(x)||$ in Fig. 2.3. If a point of intersection is made of two lines with the same sign in slope, it indicates that $||r(x)||$ can be further decreased.

2.4 Characterization of the Minimax Approximation

The two properties described at the end of section 2.3 apply to overspecified systems of linear equations with more than one unknown. The generalization of these two properties specifies the minimax solution and is known as the Characterization Theorem.* In section 2.3 which considered systems of linear equations in one unknown the minimax approximations were found at the intersection of two of the equations. Furthermore, the slopes of the two equations were of opposite sign.

In general, a system of M linear equations in N unknowns with $M > N$, the minimax approximation is found at the intersection of some subset of $N+1$ of the M equations. Generalizing to N unknowns the property that the minimax approximation is found at the intersection of two equations which have slopes of opposite sign requires the introduction of the concepts of a *convex set* and a *convex hull*.

A convex set is a set of vectors such that for any two vectors v and w contained in the set, the set also contains all vectors of the form $\theta v + (1-\theta)w$ where θ is a scalar which varies from zero to one.

Figure 2.5 shows some examples of convex and non-convex sets in R^2 .

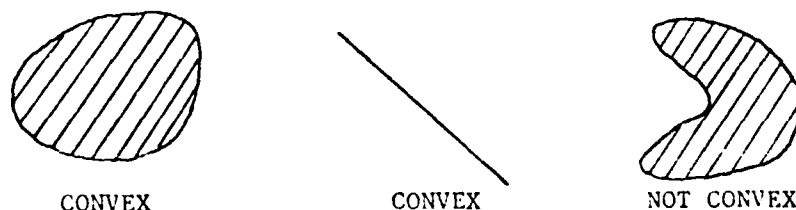


Figure 2.5 Convex Sets

* Cheney, E.W., Introduction to Approximation Theory, McGraw Hill,

The convex hull of a set S which itself is not convex is the smallest set which contains S and is convex.

Now it is possible to generalize the fact that for systems with one unknown the slopes are of opposite sign. If the m^{th} row of the matrix on the left-hand side of (2.2) is denoted A_m then to determine whether a particular intersection of $N+1$ error equations determines the minimax approximation depends upon a property of the $N+1$ associated rows of the matrix A (i.e., associated by the expression $r_m(x) = |b_m - \sum_{n=1}^N a_{mn} x_n|$ where $A_m = (a_{m1}, \dots, a_{mN})$). Denoting these $N+1$ rows as $A_{m(1)}, \dots, A_{m(N+1)}$ the condition states that the minimax approximation exists when zero belongs to the convex hull of the set

$$\{\sigma_{m(1)} A_{m(1)}, \dots, \sigma_{m(N+1)} A_{m(N+1)}\}. \quad (2.12)$$

This is where $\sigma_{m(1)}, \dots, \sigma_{m(N+1)}$ are the signs ± 1 determined by the equation

$$\sigma_m = \text{sgn} \{b_m - \sum_{n=1}^N a_{mn} x_n\} \quad (2.13)$$

for $A_m = (a_{m1}, \dots, a_{mN})$.

An equivalent statement of the condition that zero lie in the convex hull of (2.12) is that

$$0 = \sum_{n=1}^{N+1} \theta_{m(n)} \sigma_{m(n)} A_{m(n)}^* \quad (2.14)$$

* See section A.2.

where the $\theta_{m(n)}$ are scalar constants and

$$\theta_{m(n)} \geq 0 \quad \text{and} \quad \sum_{n=1}^{N+1} \theta_{m(n)} = 1 \quad (2.15)$$

In the case of one unknown the vectors A_m are one-dimensional and thus belong to the real line R . Two of the vectors determine a minimax approximation only if their slopes are of opposite sign. For example, the two vectors could be -5 and 3 which belong to R . From the definition of the convex hull, it is true that zero is contained in the convex hull of the set $\{-5, 3\}$.

A proof of the facts that the minimax approximation is found in some subset of $N+1$ of the M error equations and that zero is contained in the convex hull of the vectors $A_{m(n)}$ associated with the subset of $N+1$ error equations can be found in the appendix in sections A.1 and A.3.

3. USING A TRANSVERSAL FILTER AS A DISCRETE TIME-DOMAIN EQUALIZER

3.1 Convolution and the Transversal Filter

As discussed in section 1.2 a transversal filter has an impulse response which can be represented as in (1.2)

$$f(t) = \sum_{n=1}^N f_n \delta(t-nT) \quad (3.1)$$

Since the impulse response of the filter is finite in length, it is known as a Finite Duration Impulse Response (FIR) filter. Referring to Fig. 3.1 a transversal filter is composed of N sections each of which consists of a delay T and a scalar multiplier f_n . An incoming signal

is first delayed and then multiplied by the scalar weight of the first section. The next section further delays the unweighted signal and then multiplies it by its own scalar weight. The signal traverses all N sections of the filter and the output of the filter is a summation of the N delayed and weighted versions of the input signal. The scalar weights are variable and can be manipulated either electrically or mechanically.

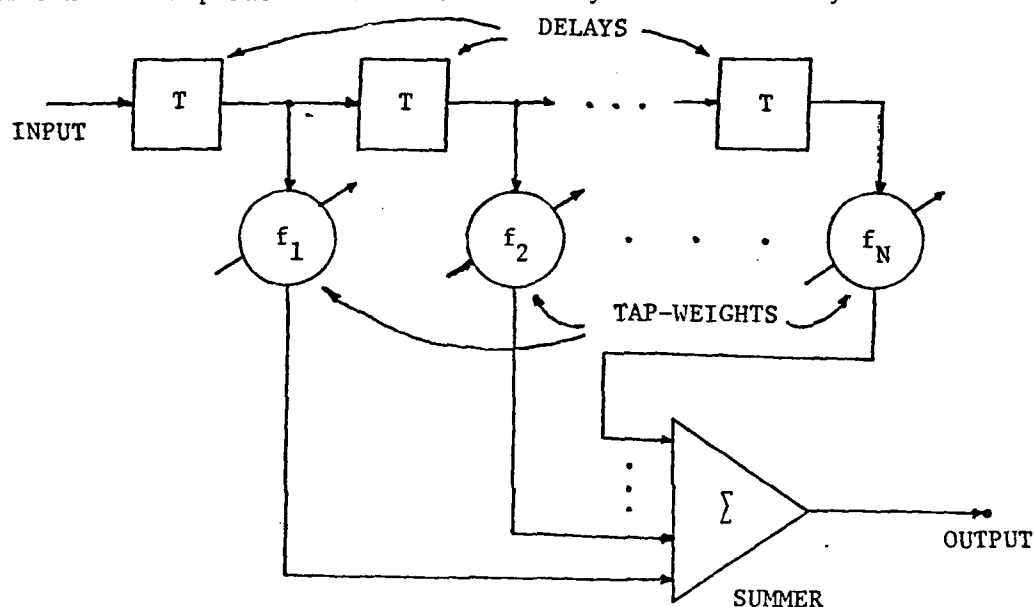


Figure 3.1 Transversal Filter

If the input to a transversal filter is a discrete-time signal $x(t)$ which only takes non-zero values at equally spaced discrete points of time, then $x(t)$ can be represented by $x(nT)$ where T is the time between non-zero values of $x(t)$ and n is an integer. The transversal filter can be represented as a discrete-time system with non-zero values $f(nT)$.

The output of a transversal filter with input $x(nT)$ is the convolution sum of $x(nT)$ and $f(nT)$. Calling $y(mT)$ the output of the

filter for m an integer and assuming that the value of T is one yields

$$y(m) = \sum_{n=1}^N x(m-n)f(n). \quad (3.2)$$

The convolution sum of (3.2) is from $n=1$ to $n=N$ since $f(n)$ is of finite length N .

For a discrete-time system or communication channel which has impulse response $h(nT) = h(n)$ for $T = 1$ the combined impulse response of $h(n)$ in cascade with $f(n)$ is the convolution sum of the two. If $h(n)$ is causal starting at $n=1$ and of finite length L , then the combined response $g(m)$ is

$$g(m) = \sum_{n=1}^N h(m-n+1)f(n) \quad m=1, \dots, M \quad (3.3)$$

where the length of $g(m)$ is $M = N+L+1$. The operation of convolution sum as in (3.2) and (3.3) is often represented by an asterisk so that (3.3) becomes

$$g(n) = h(n)*f(n). \quad (3.4)$$

3.2 Solving the Equalization Problem

The purpose of the discussion to this point has been to explain how the transversal filter can act as an equalizer for a system or a communication channel. A system or channel may have an undesirable impulse response because in some way it introduces linear distortion (i.e., alters the waveshape) to signals which traverse it. Equalization

is the process of changing the impulse response of the system or channel through the addition of a transversal filter as an equalizer. Thus, the combination of the system or channel with the filter has a more desirable response.

If the system or channel response is measured and called $h(n)$ and the desired response is specified as $g(m)$ then the problem is to determine the $f(n)$'s which satisfy (3.3). Equation (3.3) is of the form (2.1) in section 2.1 and can be written in the form of (2.2)

$$\begin{bmatrix}
 h(1) & 0 & 0 & \dots & 0 \\
 h(2) & h(1) & 0 & \dots & 0 \\
 \cdot & \cdot & \cdot & & \cdot \\
 \cdot & \cdot & \cdot & & \cdot \\
 \cdot & \cdot & \cdot & & \cdot \\
 h(N) & h(N-1) & h(N-2) & \dots & h(1) \\
 \cdot & \cdot & \cdot & & \cdot \\
 \cdot & \cdot & \cdot & & \cdot \\
 \cdot & \cdot & \cdot & & \cdot \\
 h(M-N) & h(M-N-1) & h(M-N-2) & \dots & h(M-2N) \\
 h(M-N+1) & h(M-N) & h(M-N-1) & \dots & h(M-2N+1) \\
 0 & h(M-N+1) & h(M-N) & \dots & h(M-2N+2) \\
 0 & 0 & h(M-N+1) & \dots & h(M-2N+3) \\
 \cdot & \cdot & \cdot & & \cdot \\
 \cdot & \cdot & \cdot & & \cdot \\
 \cdot & \cdot & \cdot & & \cdot \\
 0 & 0 & 0 & \dots & h(M-N+1)
 \end{bmatrix}
 \begin{bmatrix}
 f(1) \\
 f(2) \\
 \cdot \\
 \cdot \\
 \cdot \\
 f(N)
 \end{bmatrix}
 =
 \begin{bmatrix}
 g(1) \\
 g(2) \\
 \cdot \\
 \cdot \\
 \cdot \\
 g(M)
 \end{bmatrix}
 \quad (3.5)$$

If M is greater than N , which must be the case for $L > 1$, then (3.5) is an overspecified system of linear equations where the unknowns are the $f(n)$.

From the discussion in section 2.1 it is known that an over-specified system of linear equations generally has no exact solution. Thus, to solve the equalization problem it is desired to find a best approximate solution to (3.5) which may be accomplished as described in Chapter 2.

4. DISCRETE TIME ALGORITHM

4.1 Ascent Algorithm for One Unknown

In the equalization problem a system impulse response $h(n)$ is given and a desired response $g(n)$ is specified. The tap weights $f(n)$ are unknown. As pointed out in Chapter 3 the equalization problem generates an overspecified system of linear equations in N unknowns, the $f(n)$ being those unknowns. The algorithm desired must seek the minimax approximation to the $f(n)$.

The graphical technique of Chapter 2 for finding the minimax approximation to an overspecified system of linear equations in one unknown is not useful as an algorithm when the system has more than one unknown. A generalized algorithm is needed which has the capacity to find the minimax approximation to an overspecified system of linear equations in N unknowns.

This chapter gives an intuitively plausible algorithm for the case of an overspecified system in one unknown. Then the algorithm is generalized to the case of N unknowns. The algorithm to be described is known as the ascent algorithm.* For the case of one unknown a graphical description of the algorithm is based on properties of the minimax approximation as described in the characterization theorem in section 2.4.

Referring to Fig. 4.1, a particular iteration of the algorithm has located a point x_0 corresponding to the intersection of two lines. Each line represents an error equation (see section 2.3), and these lines

*Cheney, E.W., Introduction to Approximation Theory, McGraw Hill, 1966.

have slopes of opposite sign satisfying the requirements of the characterization theorem. The characterization theorem states that zero must belong to the convex hull of the vectors which represent the slopes of the lines. This condition is satisfied in the case of one unknown only when the slopes of the two lines are of opposite sign.

If x_0 corresponded to the global minimax solution then at the point x_0 there would be no lines above the two lines whose intersection defines x_0 and the algorithm would terminate. This is true because the minimax approximation is found at the intersection of two lines which belong to the function $||r(x)|| = \max_{1 \leq m \leq M} \{r_m(x)\}$ (see (2.12)). As in Fig. 4.1, if it is not the case that x_0 is the global minimax approximation

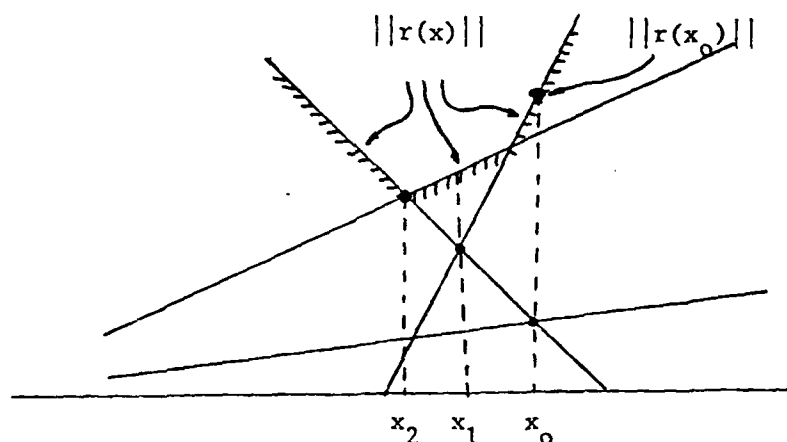


Figure 4.1 Ascent Algorithm for One Unknown

then a new line is chosen such that the magnitude of the chosen line evaluated at x_0 is equal to $||r(x_0)||$. Now the algorithm has three lines. Out of the three lines the algorithm will choose two and delete the third. The algorithm retains the new line and one of the old lines which has slope of opposite sign to the new line.

Where the two new lines intersect is the new point of the algorithm. In Fig. 4.1 this point is x_1 . The discovery of x_1 defines the next iteration of the algorithm and the algorithm proceeds with x_1 as it did with x_0 .

4.2 Generalization of the Ascent Algorithm to N Unknowns

There are three aspects of the ascent algorithm which require generalization to N unknowns. They are first, how to ensure that the convex hull of the N+1 error equations contains zero; second, how to arrive at the intersection of N+1 error equations; third, how to exchange one of the N+1 error equations for the particular equation that belongs to the function $\|r(x)\|$. The algorithm should solve all these problems efficiently and should be able to determine when the solution has been found.

4.2.1 Forcing zero to belong to the convex hull

The overspecified system of linear equations representing the equalization problem is of the form (2.2)

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ a_{M1} & a_{M2} & & a_{MN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ \cdot \\ b_M \end{bmatrix} \quad (4.1)$$

for $M > N$. For each row of the matrix in (4.1) there is an associated error equation

$$r_m(x) = \left| b_m - \sum_{n=1}^N a_{mn} x_n \right|. \quad (4.2)$$

The algorithm begins by randomly choosing a subset of $N+1$ of the M error equations. To denote which subset of $N+1$ equations are being considered, the error equations will be subscripted $r_{m(1)}(x)$, $r_{m(2)}(x)$, . . . , $r_{m(N+1)}(x)$.

It is desired that zero belong to the convex hull of the set $\{\sigma_{m(1)}A_{m(1)}, \sigma_{m(2)}A_{m(2)}, \dots, \sigma_{m(N+1)}A_{m(N+1)}\}$ where A_m is the m^{th} row vector of the matrix in (4.1) and σ_m are the signs ± 1 defined by the equation

$$\sigma_m = \text{sgn} \left\{ b_m - \sum_{n=1}^N a_{mn} x_n \right\}. \quad (4.3)$$

An intuitive understanding of the purpose of the σ_m can be obtained by referring to the case of an overspecified system in one unknown in Fig. 2.3. Each error equation in Fig. 2.3 has two parts, one part is of negative slope and the other part is of positive slope. It is important to distinguish between these two portions of $r_m(x)$. For example, the intersection of the lines $r_1(x)$ and $r_2(x)$ in Fig. 2.3 occurs in two places $x = 0$ and $x = 6/7$. At $x = 0$ the two lines have slopes of similar sign and at $x = 6/7$ the lines have slopes of opposite sign. Thus, one of the intersections constitutes a minimax approximation and the other does not. The σ_m are the devices which keep track of which of the two portions of an error equation the algorithm is using.

It is now possible to determine what values of the σ_m will make zero belong to the convex hull of the set $\{\sigma_{m(1)}A_{m(1)}, \dots, \sigma_{m(N+1)}A_{m(N+1)}\}$. By (2.14) and (2.15) zero will belong to the convex hull of this set when

$$0 = \sum_{n=1}^{N+1} \theta_{m(n)} \sigma_{m(n)} A_{m(n)} \quad (4.4)$$

for

$$\theta_{m(n)} \geq 0 \quad \text{and} \quad \sum_{n=1}^{N+1} \theta_{m(n)} = 1 \quad (4.5)$$

By choosing

$$\sigma_{m(n)} = \text{sgn} \{ \theta_{m(n)} \} \quad (4.6)$$

and recognizing that $\theta_{m(n)} \sigma_{m(n)}$ for $\theta_{m(n)} \geq 0$ is identical to $\theta_{m(n)}$ (for any positive or negative value of $\theta_{m(n)}$), (4.4) can be rewritten as

$$0 = \sum_{n=1}^{N+1} \sigma_{m(n)} A_{m(n)} = \sum_{n=1}^{N+1} [\theta_{m(n)} \sigma_{m(n)}] [\sigma_{m(n)} A_{m(n)}]. \quad (4.7)$$

Recalling that $A_m = (a_{m1}, a_{m2}, \dots, a_{mN})$ and that the sum of the $\theta_{m(n)}$ must equal 1 allows (4.7) to be rewritten in the form of a matrix transformation.

$$\begin{bmatrix} 1 & 1 & \dots & 1 \\ a_{m(1)1} & a_{m(2)1} & \dots & a_{m(N+1)1} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ a_{m(1)N} & a_{m(2)N} & \dots & a_{m(N+1)N} \end{bmatrix} \begin{bmatrix} \theta_{m(1)} \\ \theta_{m(2)} \\ \cdot \\ \cdot \\ \cdot \\ \theta_{m(N+1)} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix} \quad (4.8)$$

A solution for the $\theta_{m(n)}$ exists if the square matrix in (4.8) has an inverse. The inverse of the matrix in (4.8) is defined as the matrix

such that

$$\begin{bmatrix} 1 & 1 & \dots & 1 \\ a_{m(1)1} & a_{m(2)1} & \dots & a_{m(N+1)1} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ a_{m(1)N} & a_{m(2)N} & \dots & a_{m(N+1)N} \end{bmatrix} \begin{bmatrix} d_{11} & \dots & d_{1N+1} \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ d_{N+11} & \dots & d_{N+1N+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ 0 & 0 & \dots & 1 \end{bmatrix} \quad (4.9)$$

If the inverse exists then (4.8) can be solved for the $\theta_{m(n)}$

$$\begin{bmatrix} \theta_{m(1)} \\ \theta_{m(2)} \\ \cdot \\ \cdot \\ \cdot \\ \theta_{m(N+1)} \end{bmatrix} = \begin{bmatrix} d_{11} & \dots & d_{1N+1} \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ d_{N+11} & \dots & d_{N+1N+1} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix} = \begin{bmatrix} d_{11} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ d_{N+11} \end{bmatrix} \quad (4.10)$$

As required in (4.6) the $\sigma_{m(n)} = \text{sgn}\{\theta_{m(n)}\}$ and this choice of $\sigma_{m(n)}$ guarantees that zero belongs to the convex hull of the set

$$\{\sigma_{m(1)}A_{m(1)}, \dots, \sigma_{m(N+1)}A_{m(N+1)}\}.$$

4.2.2 Finding the intersection of the $N+1$ error equations

Knowing the signs $\sigma_{m(n)}$ it now is possible to find the minimax approximation for the subsystem of $N+1$ error equations. The only condition that needs to be satisfied is to determine where the set of $N+1$ error equations intersect. At that point is where the minimax solution

is found. Since the point being searched for occurs at the intersection of the $N+1$ error equations, the magnitudes of all the error equations are equal at this point. If the vector $y^1 = (y_1, \dots, y_N)$ is the point desired and if ϵ^1 is the magnitude of each error equation evaluated at y^1 , then $r_{m(n)}(y^1)$ can be written

$$\sigma_{m(n)} r_{m(n)}(y^1) = b_{m(n)} - \sum_{k=1}^N a_{m(n)k} y_k = \sigma_{m(n)} \epsilon^1, \quad (4.11)$$

$$m(n) = m(1), \dots, m(n+1)$$

where $A_{m(n)} = (a_{m(n)1}, \dots, a_{m(n)N})$. Rearranging (4.11) yields

$$\sigma_{m(n)} \epsilon^1 + \sum_{k=1}^N a_{m(n)k} y_k = b_{m(n)} \quad m(n)=m(1), \dots, m(N+1) \quad (4.12)$$

and this system of $N+1$ equations in $N+1$ unknowns can be represented in matrix form:

$$\begin{bmatrix} \sigma_{m(1)} & a_{m(1)N} & \dots & a_{m(1)N} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \sigma_{m(N+1)} & a_{m(N+1)N} & \dots & a_{m(N+1)N} \end{bmatrix} \begin{bmatrix} \epsilon^1 \\ y_1 \\ \cdot \\ \cdot \\ y_N \end{bmatrix} = \begin{bmatrix} b_{m(1)} \\ \cdot \\ \cdot \\ \cdot \\ b_{m(N+1)} \end{bmatrix} \quad (4.13)$$

AD-A092 734

TUFTS UNIV MEDFORD MASS DEPT OF ELECTRICAL ENGINEERING
ASPECTS OF SIGNAL PROCESSING FOR ARRAY ANTENNAS.(U)

F/6 9/5

UNCLASSIFIED
SEP 80 D PREIS
EE/TR-1980-1

AFOSR-80-0110

AFOSR-TR-80-1242 NL

2 of 2

FILED



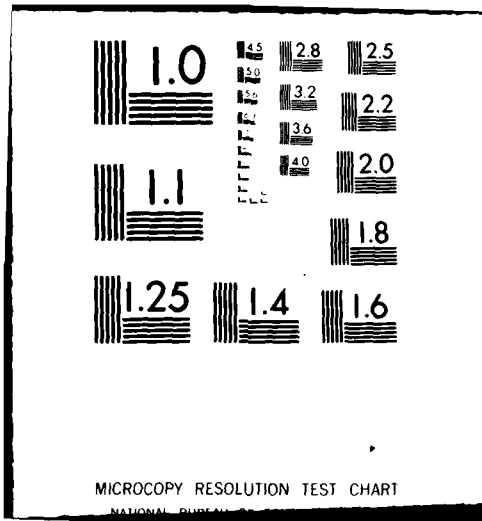
END

DATE

FILED

81

DTIC



The solution to y^1 is found by inverting the matrix on the left-hand side of (4.13).

$$\begin{bmatrix} \epsilon^1 \\ y_1 \\ \cdot \\ \cdot \\ \cdot \\ y_N \end{bmatrix} = \begin{bmatrix} \sigma_{m(1)} & a_{m(1)1} & \cdots & a_{m(1)N} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \sigma_{m(N+1)} & a_{m(N+1)1} & \cdots & a_{m(N+1)N} \end{bmatrix}^{-1} \begin{bmatrix} b_{m(1)} \\ \cdot \\ \cdot \\ \cdot \\ b_{m(N+1)} \end{bmatrix} = \begin{bmatrix} c_{11} & \cdots & c_{1\ N+1} \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ c_{N+1\ 1} & \cdots & c_{N+1\ N+1} \end{bmatrix} \begin{bmatrix} b_{m(1)} \\ \cdot \\ \cdot \\ \cdot \\ b_{m(N+1)} \end{bmatrix} \quad (4.14)$$

where the matrix on the right-hand side of (4.14) is the inverse matrix.

Each component y_n of the vector y^1 is given by the sum,

$$y_n = \sum_{k=1}^{N+1} c_{N+1\ k} b_{m(k)} \quad (4.15)$$

and the magnitude of the error is

$$|\epsilon| = \left| \sum_{k=1}^{N+1} c_{1k} b_{m(k)} \right| \quad (4.16)$$

It is important to note that y^1 is not necessarily the global minimax approximation to the system of M equations in N unknowns. However, y^1 is the minimax solution to the subset of the $N+1$ chosen equations. At this point the algorithm tests whether y^1 is the global minimax solution.

Since the system of linear equations has a total of M equations in N unknowns, it may be determined whether y^1 is a global minimax approximation by evaluating each of the remaining $M-(N+1)$ error equations at the proposed solution y^1 . If any member of the $M-(N+1)$ equations has a magnitude greater than ϵ when evaluated at y^1 then y^1 is not the global minimax approximation. Otherwise, y^1 is the solution and the algorithm terminates.

4.2.3 Exchange Theorem

In the case where y^1 is not the solution a new solution vector must be computed for a different set of $N+1$ equations. As in the case of the ascent algorithm for one unknown the algorithm will exchange one of the vectors in the original set of $N+1$ for the vector associated to the error equation which evaluated at y^1 has the greatest magnitude. Calling the new vector $A_{m(N+2)}$ the problem is to decide which of the $N+1$ vectors should be replaced, and how this can be accomplished while simultaneously satisfying the condition that zero belong to the convex hull of the new set of $N+1$ vectors.

It is known that zero lies in the convex hull of the original set of $N+1$ vectors $\{\sigma_{m(1)} A_{m(1)}, \dots, \sigma_{m(N+1)} A_{m(N+1)}\}$. Therefore, by (4.7)

$$0 = \sum_{n=1}^{N+1} \theta_{m(n)} A_{m(n)} \quad \text{and} \quad \sum_{n=1}^{N+1} \theta_{m(n)} = 1 \quad (4.17)$$

The condition that zero lie in the convex hull of the $A_{m(n)}$ implies that one of the $N+1$ vectors, say $A_{m(j)}$, can be written as a linear

combination of the N other vectors:

$$A_{m(j)} = \sum_{\substack{n=1 \\ n \neq j}}^{N+1} - \frac{\theta_{m(n)}}{\theta_{m(j)}} A_{m(n)} . \quad (4.18)$$

If it is assumed that the $N+1$ vectors $A_{m(n)}$ span N space then it is possible to express the new vector $A_{m(N+2)}$ as a linear combination of the $A_{m(n)}$ for $n = 1, \dots, N+1$ and some scalar coefficients $\lambda_{m(n)}$,

$$A_{m(N+2)} = \sum_{n=1}^{N+1} \lambda_{m(n)} A_{m(n)} . \quad (4.19)$$

Equation (4.19) can be rewritten

$$\begin{aligned} A_{m(N+2)} &= \lambda_{m(j)} A_{m(j)} + \sum_{\substack{n=1 \\ n \neq j}}^{N+1} \lambda_{m(n)} A_{m(n)} \\ &= \lambda_{m(j)} \left[\sum_{n=1}^{N+1} - \frac{\theta_{m(n)}}{\theta_{m(j)}} A_{m(n)} \right] + \sum_{\substack{n=1 \\ n \neq j}}^{N+1} \lambda_{m(n)} A_{m(n)} \\ &= \sum_{\substack{n=1 \\ n \neq j}}^{N+1} \left(\lambda_{m(n)} - \lambda_{m(j)} \frac{\theta_{m(n)}}{\theta_{m(j)}} \right) A_{m(n)} \end{aligned} \quad (4.20)$$

so that $A_{m(N+2)}$ is now expressed as a linear combination of the $A_{m(n)}$ for $n = 1, \dots, N+1$ excluding $n=j$. Rewriting (4.20)

$$0 = A_{m(N+2)} + \left[\sum_{\substack{n=1 \\ n \neq j}}^{N+1} \lambda_{m(j)} \frac{\theta_{m(n)}}{\theta_{m(j)}} - \lambda_{m(n)} \right] A_{m(n)} \quad (4.21)$$

it is seen that zero is expressed as a linear combination of $N+1$ vectors and that the coefficient of each vector is positive if j is chosen such that $\lambda_{m(j)} \frac{\theta_{m(n)}}{\theta_{m(j)}} - \lambda_{m(n)} > 0$, or equivalently $\lambda_{m(j)}/\theta_{m(j)} > \lambda_{m(n)}/\theta_{m(n)}$ for all n . This condition ensures that the vector replaced by $A_{m(N+2)}$ is such that zero remains in the convex hull of the new $N+1$ vectors.

The sign $\sigma_{m(N+2)}$ associated with vector $A_{m(N+2)}$ is simply the sign of $r_{m(N+2)}(x)$ without magnitude brackets evaluated at y^1 ,

$$\sigma_{m(N+2)} = \text{sgn} \left\{ b_{m(N+2)} - \sum_{k=1}^N a_{m(N+2)k} y_k \right\}. \quad (4.22)$$

Since it is desired that zero belong to the convex hull of the vectors modified by their signs, the modified coefficients $\lambda_{m(n)}$ and $\theta_{m(n)}$ will now be determined.

To find $\theta_{m(n)}$, first recall that the matrix of the $\sigma_{m(n)} A_{m(n)}$ times its inverse equals the identity matrix

$$\begin{bmatrix} C_{11} & \dots & C_{1N+1} \\ \vdots & & \vdots \\ C_{N+1,1} & \dots & C_{N+1,N+1} \end{bmatrix} \begin{bmatrix} \sigma_{m(1)} a_{m(1)1} & \dots & a_{m(1)N} \\ \vdots & & \vdots \\ \sigma_{m(N+1)} a_{m(N+1)1} & \dots & a_{m(N+1)N} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} \quad (4.23)$$

Thus,

$$\sum_{n=1}^{N+1} \sigma_{m(n)} C_{1n} = 1 \quad \text{and} \quad \sum_{n=1}^{N+1} C_{1n} a_{m(n)k} = 0 \quad k=1, \dots, N \quad (4.24)$$

which is exactly the condition that zero belong to the convex hull of

the $\sigma_{m(n)} A_{m(n)}$ where the $\sigma_{m(n)} C_{1n}$ are the modified $\theta_{m(n)}$.

Now the $\lambda_{m(n)}$ are the coefficients of the vector $A_{m(N+2)}$ in the linear combination

$$A_{m(N+2)} = \sum_{n=1}^{N+1} \lambda_{m(n)} A_{m(n)} \quad (4.25)$$

which is modified in matrix form as

$$(\sigma_{m(N+2)}, a_{m(N+2)1}, \dots, a_{m(N+2)N}) = (\lambda_{m(1)}, \dots, \lambda_{m(N+1)}) \begin{bmatrix} \sigma_{m(1)} & a_{m(1)1} & \dots & a_{m(1)N} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \sigma_{m(N+1)} & a_{m(N+1)1} & \dots & a_{m(N+1)N} \end{bmatrix} \quad (4.26)$$

and therefore,

$$(\sigma_{m(N+2)}, a_{m(N+2)1}, \dots, a_{m(N+2)N}) = (\lambda_{m(1)}, \dots, \lambda_{m(N+1)}) \begin{bmatrix} c_{11} & \dots & c_{1 \ N+1} \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ c_{N+1 \ 1} & \dots & c_{N+1 \ N+1} \end{bmatrix}. \quad (4.27)$$

Since

$$\begin{aligned}\sigma_{m(N+2)} A_{m(N+2)} &= \sum_{n=1}^{N+1} \sigma_{m(N+2)} (\lambda_{m(n)} A_{m(n)}) \\ &= \sum_{n=1}^{N+1} (\sigma_{m(N+2)} \sigma_{m(n)} \lambda_{m(n)}) (\sigma_{m(n)} A_{m(n)})\end{aligned}\quad (4.28)$$

the modified $\lambda_{m(n)}$ are the $\sigma_{m(N+2)} \sigma_{m(n)} \lambda_{m(n)}$ and thus the modified ratios to be computed are $\sigma_{m(N+2)} \sigma_{m(n)} \lambda_{m(n)} / \sigma_{m(n)} c_{ln} = \sigma_{m(N+2)} \lambda_{m(n)} / c_{ln}$. The index which has the greatest such ratio is the index of the vector which will be replaced.

Now a minimax approximation can be computed from the new set of $N+1$ equations and tested to determine whether it is the global minimax approximation.

4.3 Manipulation of the Inverse Matrix

Two important topics remain. The first topic is concerned with the nature of the inverse matrix in (4.14). The second topic deals with convergence of the algorithm. Convergence of the algorithm will be discussed in the next section.

The algorithm described to this point requires two matrix inversions for the first iteration and one inversion for each successive iteration. The first two inversions are to determine the signs σ_m in (4.10) and to find the intersection of the initial set of $N+1$ error equations (see (4.14)). For each successive iteration of the algorithm a vector from the old set of $N+1$ vectors is exchanged for a new vector $A_{m(N+2)}$ (see section 4.2.3). The sign of the new vector is established, and the only operation to be performed is the location of the point of intersection of the set of $N+1$ new error equations. As in (4.14) this

can be accomplished through a matrix inversion of the new set of $N+1$ vectors and their associated signs.

An algorithm with many matrix inversions is undesirable and should be avoided, if possible. The reasons for this are because matrices can be difficult to invert numerically (due to round-off error, and data inaccuracies), matrix inversion algorithms are time-consuming.

Since only one row of the matrix is exchanged for each iteration of the ascent algorithm it is possible to eliminate all of the matrix inversions beyond the initial two. In general, if only the β^{th} row of a square matrix (whose inverse is known) is altered, the changes in the inverted matrix can be predicted without inverting the altered matrix:

Assume that an $N \times N$ matrix has rows $A_n = (a_{n1}, \dots, a_{nN})$ and its inverse has columns $C_n = (C_{1n}, \dots, C_{Nn})$. Furthermore, assume that the matrix is altered by exchanging A_β , the matrix's β^{th} row, for the N -vector v . The altered matrix will now have rows A_n^* and the inverse will have columns C_n^* . Note that the only difference between the matrix A and A^* is that the β^{th} row of A^* is $A_\beta^* = (v_1, \dots, v_N)$.

It now will be shown that the β^{th} column of the altered inverse C_β^* is

$$C_\beta^* = C_\beta / \langle v, C_\beta \rangle \quad (4.29)$$

and that every other column of the altered inverse C_n^* is

$$C_n^* = C_n - \langle v, C_n \rangle C_\beta^* \quad \begin{matrix} n = 1, \dots, N+1 \\ n \neq \beta \end{matrix} \quad (4.30)$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. It turns out that the transformation in (4.29) and (4.30) is such that the matrix C^* is the inverse of the matrix A^* (i.e., $A^* C^* = C^* A^* = I$, for I the identity matrix).

There are four cases to take into consideration to prove (4.29) and (4.30),

Proof:

Case (i): for $m = n = \beta$,

$$\langle A_\beta^*, C_\beta^* \rangle = \langle v, C_\beta / \langle v, C_\beta \rangle \rangle = \langle v, C_\beta \rangle / \langle v, C_\beta \rangle = 1$$

Case (ii): for $m = \beta$ & $n \neq \beta$

$$\begin{aligned} \langle A_\beta^*, C_n^* \rangle &= \langle v, C_n - \langle v, C_n \rangle C_\beta^* \rangle = \langle v, C_n \rangle - \langle v, C_n \rangle \langle v, C_\beta^* \rangle \\ &= \langle v, C_n \rangle - \langle v, C_n \rangle [\langle v, C_\beta \rangle / \langle v, C_\beta \rangle] = 0 \end{aligned}$$

Case (iii): for $m \neq \beta$ & $n = \beta$

$$\begin{aligned} \langle A_m^*, C_\beta^* \rangle &= \langle A_m, C_\beta / \langle v, C_\beta \rangle \rangle = \langle A_m, C_\beta \rangle / \langle v, C_\beta \rangle \\ &= 0 / \langle v, C_\beta \rangle = 0 \end{aligned}$$

Case (iv): for $m \neq \beta$ & $n \neq \beta$

$$\begin{aligned} \langle A_m^*, C_n^* \rangle &= \langle A_m, C_n - \langle v, C_n \rangle C_\beta^* \rangle = \langle A_m, C_n \rangle - \langle v, C_n \rangle \langle A_m, C_\beta^* \rangle \\ &= \langle A_m, C_n \rangle - \langle v, C_n \rangle [\langle A_m, C_\beta \rangle / \langle v, C_\beta \rangle] \\ &= \langle A_m, C_n \rangle - \langle v, C_n \rangle [0 / \langle v, C_\beta \rangle] = \langle A_m, C_n \rangle = \delta_{nm} \end{aligned}$$

Cases (i)-(iv) have shown that C^* is the inverse of the matrix A^* .

In the algorithm, one of the rows of the matrix is replaced with the vector $v = (\sigma_{m(N+2)}, a_{m(N+2)1}, \dots, a_{m(N+2)N})$ and the inner products between v and the columns of the original inverse the $\langle v, C_n \rangle$ need to be computed to take advantage of the above proof.

However, these inner products have previously been computed in the exchange portion of the algorithm and are the $\lambda_{m(n)}$. Specifically,

$$\lambda_{m(n)} = \langle \sigma_{m(N+2)} A_{m(N+2)}, C_{m(n)} \rangle \quad (4.31)$$

(see (4.27)).

There is one condition which must be satisfied when manipulating the inverse of the matrix in (4.14). If A^* is a singular matrix then the inner product $\langle v, C_\beta \rangle = 0$ and the foregoing calculations are not valid.

4.4 Convergence

It remains to be shown that the ascent algorithm converges to a solution. Since there are only a finite number of subsets of $N+1$ vectors out of the M $\left(\begin{matrix} M \\ N+1 \end{matrix} \right) = M! / (N+1)!(M-N-1)!$ to be exact), the solution would be guaranteed simply by testing all possible combinations. Since the ascent algorithm does not check all possible combinations, it must be shown that the algorithm does not cycle - i.e., alternate from the current solution to a previous solution. This is shown by proving that the magnitude of the minimax error grows larger for each iteration of the algorithm.

Suppose that a certain step of the algorithm yields the set of vectors $\{A_1, \dots, A_{N+1}\}$ and associated with this set is a computed solution vector y and error ϵ . Assume further that in the next step of the algorithm that A_1 will be replaced by A_{N+2} thus creating the new set of vectors $\{A_2, \dots, A_{N+2}\}$. Associated with the new set of vectors is a new solution vector y' and error ϵ' .

From the choice of A_{N+2} it is known that,

$$|r_{N+2}(y)| > |r_2(y)| \quad \text{and} \quad |r_{N+2}(y')| = |r_2(y')|. \quad (4.32)$$

Thus $y \neq y'$ or equivalently

$$y - y' \neq 0. \quad (4.33)$$

For $n = 2, \dots, N+1$ it is true that

$$\begin{aligned} \langle \sigma_n A_n, y - y' \rangle &= [\langle \sigma_n A_n, y \rangle - b_n] - [\langle \sigma_n A_n, y' \rangle - b_n] \\ &= \sigma_n r_n(y) - \sigma_n r_n(y') = \varepsilon - \varepsilon'. \end{aligned} \quad (4.34)$$

Thus by (4.33) and (4.34)

$$\varepsilon - \varepsilon' \neq 0 \quad (4.35)$$

For $n = N+2$,

$$\begin{aligned} \langle \sigma_{N+2} A_{N+2}, y - y' \rangle &= [\langle \sigma_{N+2} A_{N+2}, y \rangle - b_{N+2}] - [\langle \sigma_{N+2} A_{N+2}, y' \rangle - b_{N+2}] \\ &= \sigma_{N+2} r_{N+2}(y) - \sigma_{N+2} r_{N+2}(y') > \varepsilon - \varepsilon'. \end{aligned} \quad (4.36)$$

If $\varepsilon - \varepsilon' > 0$ then it must be so that $\langle \sigma_n A_n, y - y' \rangle > 0$ for $n = 2, \dots, N+2$ by (4.34) and (4.36). However, it will now be shown that if $\varepsilon - \varepsilon' > 0$ then zero does not belong to the convex hull of the set $\{A_2, \dots, A_{N+2}\}$ which contradicts the condition that the set have a minimax solution.

If zero belongs to the convex hull of the set $\{A_2, \dots, A_{N+2}\}$ then there exist θ_n such that

$$\theta_n \geq 0 \quad \text{and} \quad \sum_{n=2}^{N+2} \theta_n = 1 \quad (4.37)$$

and,

$$0 = \sum_{n=2}^{N+2} \theta_n \sigma_n A_n \quad (4.38)$$

(see (2.14)). Since $y - y'$ is a constant vector then by (4.38).

$$\sum_{n=2}^{N+2} \theta_n \langle \sigma_n A_n, y - y' \rangle = \langle \sum_{n=2}^{N+2} \theta_n \sigma_n A_n, y - y' \rangle = 0. \quad (4.39)$$

Equation (4.39) must be violated if $\langle \sigma_n A_n, y - y' \rangle > 0$ for all $n=2, \dots, N+2$ since by (4.37) the θ_n are all greater than or equal to zero. Thus it has been demonstrated that $\langle \sigma_n A_n, y - y' \rangle$ cannot be greater than zero for all θ_n and it may be concluded that the supposition $\varepsilon - \varepsilon' > 0$ is false. Due to (4.35) the only remaining possibility is that $\varepsilon' - \varepsilon > 0$ or equivalently $\varepsilon' > \varepsilon$. Thus, for each iteration of the algorithm the value of the error must increase and this fact proves the algorithm's effectiveness.

4.5 Restrictions on the Ascent Algorithm

There is an underlying assumption which makes the ascent algorithm operational. It concerns the condition of the set of linear equations, the extent to which these equations are linearly independent of each other. For the algorithm to work, it is necessary to perform matrix inversions and to manipulate matrix inverses. The algorithm proceeds on the

assumption that for each matrix and inverse matrix it works with that each one will be non-singular.

A non-singular matrix is a matrix whose determinant is non-zero which is equivalent to the condition that the rows of the matrix be linearly independent and that the columns of the matrix be linearly independent. Therefore, if the algorithm is to operate on matrices and their inverses, it must be guaranteed that the inverses exist. This is equivalent to requiring that every subset of $N+1$ from the set of M vectors be linearly independent.

This condition is known as the Haar condition* and the ascent algorithm will, in general, only be successful if the rows of the matrix in (2.2) satisfy the Haar condition.

5. USING THE TRANSVERSAL FILTER AS A CONTINUOUS TIME EQUALIZER

5.1 Statement of The Equalization Problem

When the system or channel to be equalized has a continuous time impulse response $h(t)$, then the convolution between $h(t)$ and the impulse response of a transversal filter $f(t)$ is

$$g(t) = \sum_{n=1}^N f_n h(t-nT) \quad (5.1)$$

where $g(t)$ is the desired equalized response, the f_n are the tap weights of the filter, and T is a unit delay (see Fig. 1.3). For $h(t)$ measured and $g(t)$ specified, the objective of equalization is to solve (5.1) for the value of the tap weights, the f_n . As previously discussed in section 1.3, (5.1) has no exact solution unless $g(t)$ can be written as a

* Cheney, E.W., Introduction to Approximation Theory, McGraw Hill, 1966.

linear combination of the $h(t-nT)$ for $n=1, \dots, N$. In general, the solutions to (5.1) are approximate solutions and to gauge the quality of these solutions an error function $r(t)$ is established as in (1.8).

$$r(t) = g(t) - \sum_{n=1}^N f_n h(t-nT). \quad (5.2)$$

Now, as discussed in section 1.4 the norm of $r(t)$ will serve as a measure of closeness between the approximation $\sum f_n h(t-nT)$ and the desired function $g(t)$. The Chebyshev norm will be the specific distance measure explored here. The Chebyshev norm of $r(t)$ defined on an interval $[a, b]$ is defined as

$$||r(t)|| = ||g(t) - \sum_{n=1}^N f_n h(t-nT)|| = \max_{a \leq t \leq b} |g(t) - \sum_{n=1}^N f_n h(t-nT)| \quad (5.3)$$

In Chapter 2 the discussion centered on how to find the Chebyshev norm of an overspecified system of linear equations and how to minimize the norm of such a system. This process and its results are known as the minimax approximation and it is desired to find a minimax approximation to the problem of (5.1) by minimizing the norm of $r(t)$ as defined in (5.3).

The path to follow which leads to a minimax solution of (5.1) is not readily apparent since the practical problem of finding the minimum point of $||r(t)||$ has only been discussed in terms of a system of linear equations. Section 5.2 will present a method for solving (5.1) in the minimax sense.

5.2 Linearization of the Continuous Error Function

The function $r(t)$ can be seen as an error equation of a linear equation in N unknowns if (5.1) is evaluated at a single point in time $t = t_1$,

$$g(t_1) = \sum_{n=1}^N f_n h(t_1 - nT) . \quad (5.4)$$

Equation (5.4) is of the form of a linear equation

$$b = \sum_{n=1}^N a_n x_n \quad (5.5)$$

where the values of b and a_n are determined by the functions $g(t)$ and $h(t)$ evaluated at the points t_1 and $t_1 - nT$. Specifically, $b = g(t_1)$ and $a_n = h(t_1 - nT)$.

Any interval $[a, b]$ of the time axis has infinitely many points belonging to it and each of these points may be evaluated in (5.4) to generate a linear equation in N unknowns. Thus (5.1) can be considered an overspecified system of linear equations where the number of equations is infinite and the number of unknowns is N .

From section 2.4 it is known that the minimax approximation is the minimum point of the function $||r||$ and that the minimum point is located by the intersection of some subset of $N+1$ error equations from the total set of M . The object of the next chapter will be to locate the $N+1$ points on the time axis which when used in (5.2) will generate the minimax approximation to (5.1).

6. REMEZ ALGORITHM

6.1 Process of the Remez Algorithm

As demonstrated in Chapter 2, it is always possible to find the minimax approximation to a finite overspecified system of linear equations in N unknowns. The minimax approximation is located by some subset of $N+1$ equations and since there are only a finite number of subsets containing $N+1$ equations the location of the minimax approximation is guaranteed simply by testing all possible subsets.

When the set of equations is infinite, the number of subsets containing $N+1$ equations is also infinite. Thus, it is not possible to test all possible subsets of the system. What is sought is an algorithm which sequentially examines subsets of $N+1$ equations such that the sequence converges to the optimum subset of $N+1$ equations. One algorithm which accomplishes this is called the Second Algorithm of Remez^{*} which is analogous to the ascent algorithm described in Chapter 4.

In the time domain the error function $r(t)$ (as defined in (5.2)) is of the form

$$r(t) = g(t) - \sum_{n=1}^N f_n h(t-nT) \quad a \leq t \leq b \quad (6.1)$$

for some interval $[a,b]$. As discussed in section 5.2, (6.1) is equivalent to an overspecified system of linear equations (with an infinite number of equations) in N unknowns.

The Remez algorithm begins by choosing an arbitrary subset of $N+1$ points from the time interval $[a,b]$. These $N+1$ points define a subset of $N+1$ linear equations in N unknowns (see section 5.1) and it is possible

^{*}Cheney, L.W., Introduction to Approximation Theory, McGraw Hill, 1964.

to find the minimax approximation to this subset of $N+1$ equations using the ascent algorithm as described in Chapter 4.

The set of tap weights $\{f_1, \dots, f_N\}$ and the minimax error ϵ is computed from the subset of $N+1$ equations. For $\{f_1, \dots, f_N\}$ to be the global minimax approximation to $r(t)$ on $[a, b]$, the magnitude of $r(t)$ must be bounded by ϵ . This condition may be determined by inserting the computed set $\{f_1, \dots, f_N\}$ into the continuous expression for $r(t)$ in (6.7). If $|r(t)| > \epsilon$ for any t belonging to $[a, b]$ then the set $\{f_1, \dots, f_N\}$ does not constitute the global minimax approximation to $r(t)$.

The subset of $N+1$ equations is then replaced by a different subset of $N+1$ equations. This is accomplished by choosing a new set of $N+1$ points from the interval $[a, b]$. The procedure differs here from the ascent algorithm as described in section 4.1 since more than one equation is replaced for each iteration. Generally, all $N+1$ equations are replaced by $N+1$ new equations in each iteration of the algorithm.

A graphical example of exchanging more than one equation per iteration is shown in Fig. 6.1. The ascent algorithm described in

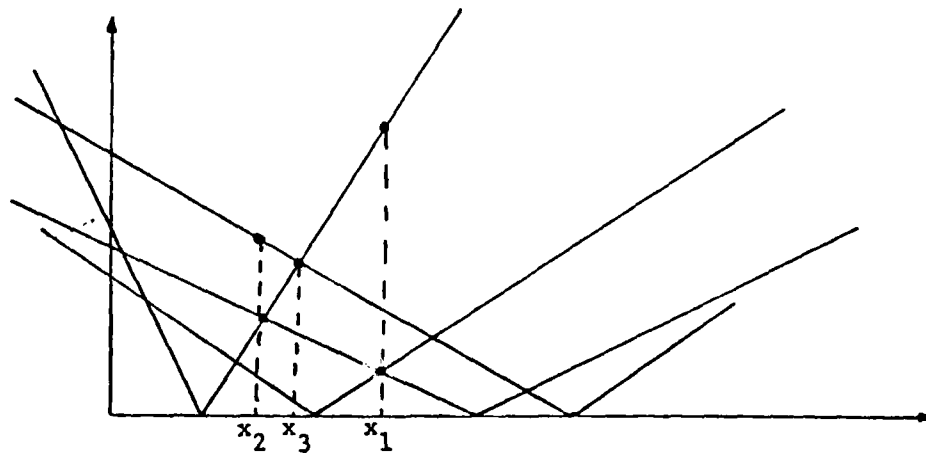


Figure 6.1 Example of Multiple Exchanges in Ascent Algorithm

Chapter 4 determines the global minimax approximation by computing a succession of solutions for subsets of $N+1$ equations. Since Fig. 6.1 is an example of a system in one unknown the subsets will contain two lines and the point x_1 is a relative minimax solution since it is located at the intersection of two lines which have slopes of opposite sign. Furthermore, when evaluated at x_1 there are two lines of greater magnitude than the magnitude of the lines which intersect at x_1 .

The Ascent algorithm described in Chapter 4 proceeds by replacing one of the two intersecting lines at x_1 by the single line of greatest magnitude evaluated at x_1 . The intersection of the two new lines locates a relative minimax solution at x_2 . Repeating this process once again locates the global minimax solution x_3 .

At x_1 in Fig. 6.1 if both intersecting lines which were used in locating x_1 are replaced in one iteration by the two lines of greatest magnitude at x_1 the algorithm converges faster. The intersection of the two new lines immediately locates the global minimax solution at x_3 and bypasses the examination of the relative minimax solution at x_2 .

In the Remez algorithm every set of $N+1$ equations is replaced by a new set of $N+1$ equations. The algorithm sequentially computes sets of tap weights (one set for each set of $N+1$ equations) which in the limit converge to optimum values. Because the algorithm must eventually terminate, the criterion for ending the algorithm depends on the percent difference between two consecutive computations of $||r(t)||$. When the percent difference is smaller than some specified percentage the algorithm ends.

6.2 Operational Details of the Remez Algorithm

The framework of the algorithm has been described, what remains are the operational details. There are three parts of the Remez algorithm, they are: (1) arriving at the relative minimax approximation for the $N+1$ equations generated by part (3); (2) evaluation of the merit of the approximation in part (1); and (3) choosing a set of $N+1$ points from the time interval $[a,b]$ for use in part (1).

The first set of $N+1$ points can be arbitrary, and it is assumed that the algorithm has a set of $N+1$ points t_n such that $a \leq t_1 < \dots < t_{N+1} \leq b$. These points are used to generate an overspecified system of $N+1$ linear equations in N unknowns by evaluating (5.1) at each of the points t_n .

The minimax solution for f_n is found by the ascent algorithm which was described in Chapter 4. The only essential difference is that the portion of the ascent algorithm which performs manipulations on the inverse matrix (see section 4.3) is invalid since more than one equation is exchanged in each iteration of the algorithm. As in the first step of the ascent algorithm a set of $N+1$ signs σ_n must be computed for the $N+1$ equations and then the $(N+1) \times (N+1)$ matrix of signs and vector coefficients is inverted to find the minimax approximation to this particular subset of $N+1$ equations. An inversion of a matrix to compute the signs σ_n is unnecessary due to an alternation property of the minimax approximation for continuous time signals. This property is described by a theorem called the Alternation Theorem^{*} and the property is a consequence of the requirement that zero belong to the convex hull of each subset of $N+1$ equations. Specifically, the alternation theorem states that the minimax approximation generated by the set of points

^{*}Cheney, E.W., Introduction to Approximation Theory, McGraw Hill, 1966.

$a \leq t_1 < \dots < t_{N+1} \leq b$ has the property that $r(t_n) = -r(t_{n+1})$ for $n=1, \dots, N$.^{*} Knowing that the magnitude of the error function will alternate in sign at the $N+1$ points t_n is equivalent to knowing *a priori* that the signs σ_n will alternate in sign since $\sigma_n = \text{sgn} \{r(t_n)\}$.

Each minimax approximation computed by the Remez algorithm is for a subset of $N+1$ equations generated by $N+1$ points from the interval $[a, b]$. For each iteration the algorithm tests how good an approximation it has computed. Later in this chapter it will be proven that the Remez algorithm achieves linear convergence. Using this knowledge it can be shown that each successive tap-weight vector is closer to the optimum tap-weight vector than the same vector from the previous iteration. Termination of the algorithm can be controlled by monitoring the distance between successive tap-weight vectors or by monitoring the percent difference between successive values of $\|r(t)\|$.

If the algorithm has finished computing the minimax approximation to a particular subset of $N+1$ equations and the algorithm does not terminate, then it replaces the $N+1$ points of the past iteration. The description of how the algorithm makes this replacement refers to Fig. 6.2.

The central idea in this part of the algorithm is to exchange one subset of $N+1$ points for another subset such that the error function evaluated at the new subset of points has magnitudes unbounded by the minimax error of the previous iteration. It is known that the previous iteration generated a minimax approximation for the $N+1$ points $a \leq t_1 < \dots < t_{N+1} \leq b$. From the characteristics of the minimax approximation

^{*} See section A.4.

it is known that $|r(t_1)| = |r(t_2)| = \dots = |r(t_{N+1})| = \epsilon$, and that any value of t in $[a, b]$ such that $|r(t)| > \epsilon$ is an error unbounded by the minimax error.

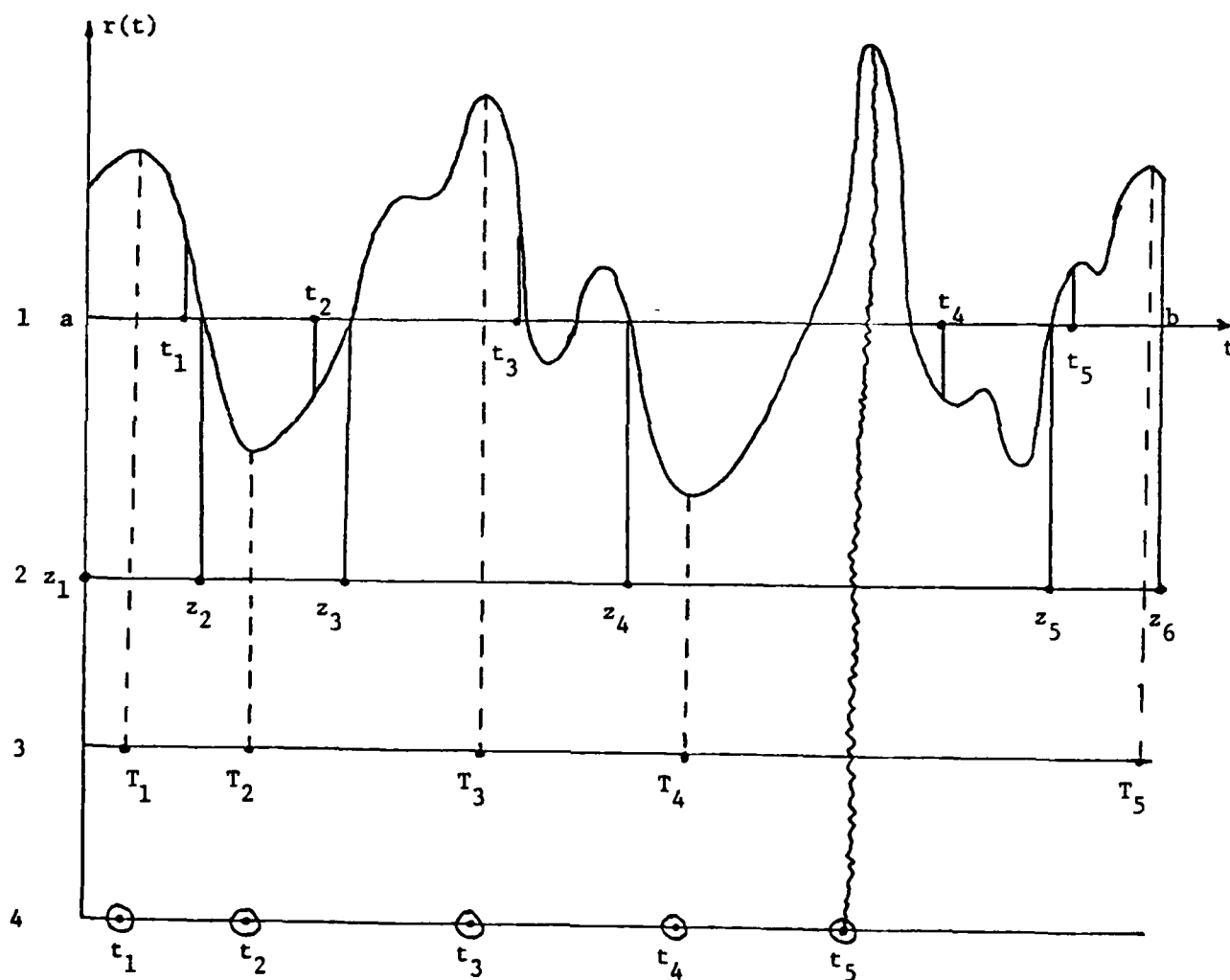


Figure 6.2 Details of Exchanging Subsets of $N+1$ Points in the Remez Algorithm (for $N+4$).

The alternation theorem requires $r(t_n) = -r(t_{n+1})$ for $n=1, \dots, N$. Thus, for each pair of indices $(n, n+1)$, $r(t)$ must have a root in the interval $[t_n, t_{n+1}]$. For every interval $[t_n, t_{n+1}]$ the algorithm then locates this root and labels it z_{n+1} . The set of roots z_n also contains the endpoints a and b so that, as can be seen in Fig. 6.2 abscissa number 2, the $N+2$ roots are ordered $a = z_1 < \dots < z_{N+2} = b$. For each pair of roots the interval $[z_n, z_{n+1}]$ contains a single member of the set $\{t_1, \dots, t_{N+1}\}$.

Using the set of roots $\{z_1, \dots, z_{N+2}\}$ a test set of new points will be generated for the next iteration of the algorithm. It is a test set because the choices for the set may be modified before being passed to the next iteration of the algorithm. The test set is found by choosing a point T_n from each interval $[z_n, z_{n+1}]$ for $n = 1, \dots, N+1$ such that $\sigma_n r(T_n)$ is a maximum in the interval. The coefficient is the sign of the error function evaluated at t_n (i.e., $\sigma_n = \text{sign} \{r(t_n)\}$).

Abcissa number 3 in Fig. 6.2 shows the test set $\{T_1, \dots, T_{N+1}\}$. This set is modified only if the value t' for which $|r(t')|$ is a maximum in the interval $[a, b]$ is not a member of the set $\{T_1, \dots, T_{N+1}\}$. This condition is equivalent to $||r(t')| > |r(T_n)|$ for all n . If t' does not belong to the set $\{T_1, \dots, T_{N+1}\}$ it is desired to insert t' into the set and to remove an appropriate T_n . This is done by first inserting t' into its appropriate ordered position within the set $\{T_1, \dots, T_{N+1}\}$. Then one of the T_n will be adjacent to t' and will be such that $\text{sgn} \{r(T_n)\} = \text{sgn} \{r(t')\}$. If t' is less than T_1 or greater than T_{N+1} then both T_1 and T_{N+1} are considered adjacent to t' . It is the T_n which satisfies these conditions which is removed from the set. The remaining T_n and the point t' constitute a new set of t_n (see Fig. 6.2 abscissa number 4).

Figure 6.2 shows how the new set of t_n replaces the old set. Note that T_5 is the adjacent point to t' which has the property $\text{sgn } r(T_5) = \text{sgn } r(t')$. The algorithm now can begin the next iteration by finding the minimax solution to the set of $N+1$ equations generated by the new set of $N+1$ points. Figure 6.3 shows a flow chart for the Remez algorithm.

6.3 Convergence of the Remez Algorithm

It remains to be shown that the Remez algorithm converges to a best tap-weight vector f_n^* . A best tap-weight vector in the minimax sense is one for which the associated minimax error ϵ^* bounds the magnitude of the error function $r(t)$ when $r(t)$ is evaluated using f_n^* ,

$$r(t) = g(t) - \sum_{n=1}^N f_n^* h(t-nT). \quad (6.2)$$

For each iteration of the algorithm there is an ordered set of points

$$a \leq t_1 < \dots < t_{N+1} \leq b \quad (6.3)$$

for which a minimax approximation f_n and associated minimax error ϵ has been computed. Thus,

$$\epsilon = |r(t_1)| = |r(t_2)| = \dots = |r(t_{N+1})|. \quad (6.4)$$

Furthermore, for the same iteration a new set of ordered points

$$a \leq t_1' < \dots < t_{N+1}' \leq b \quad (6.5)$$

is chosen such that for each point t_n' , $|r(t_n')| > \epsilon$. Let

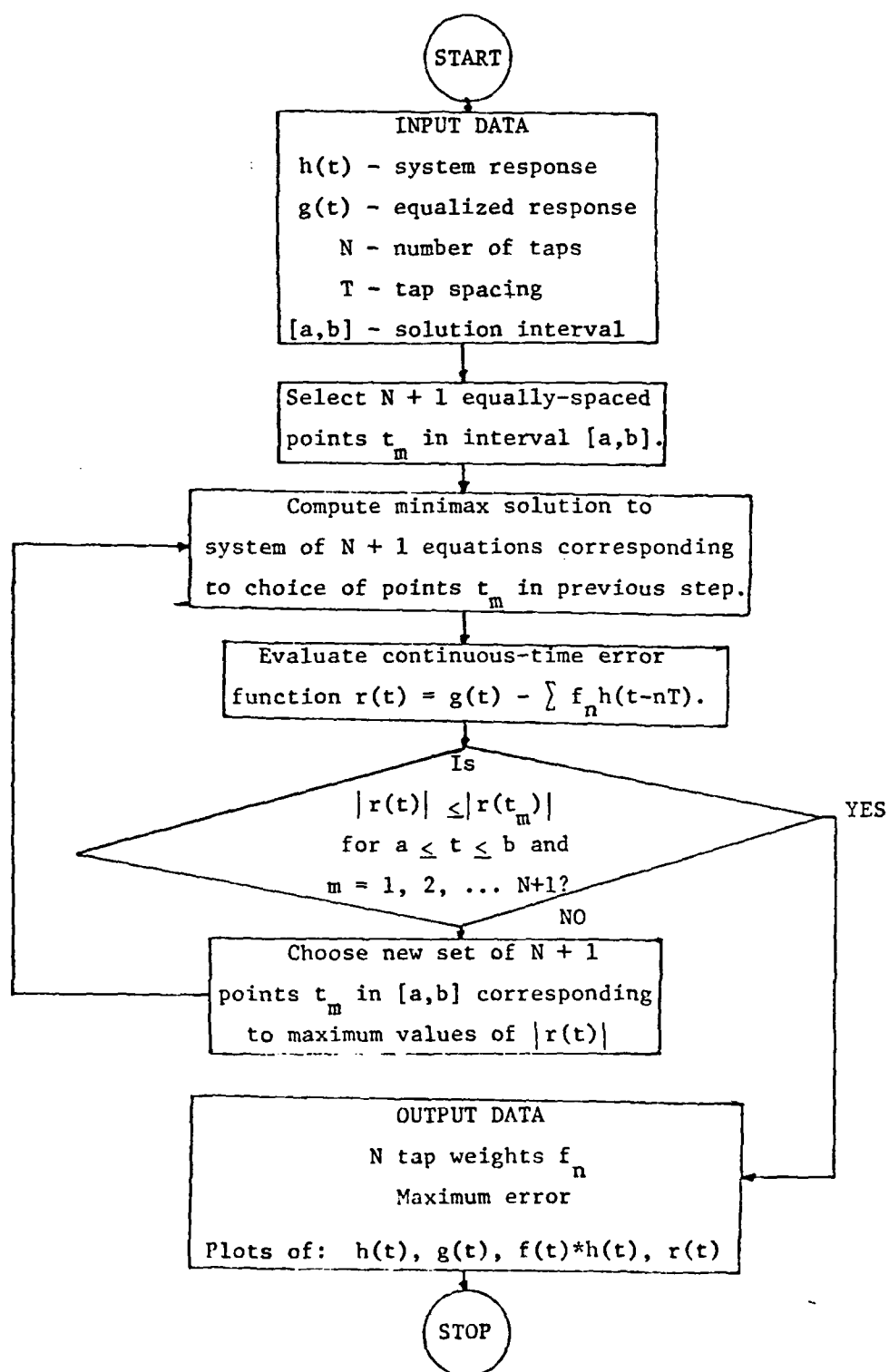


FIGURE 6.3 Flow Chart for the Remez Algorithm

$$\alpha = \min_{1 \leq n \leq N+1} |r(t_n')|, \quad \beta = \max_{1 \leq n \leq N+1} |r(t_n')| \quad (6.6)$$

So that by the theorem of de la Vallée Poussin* and by (6.4) and (6.6)

$$\varepsilon \leq \alpha \leq \varepsilon^* \leq \beta. \quad (6.7)$$

The proof of the fact that the Remez algorithm converges to a best tap-weight vector f_n^* is made by showing that both β and α converge uniformly to ε^* .

Letting ε' , α' , and β' denote the quantities defined in (6.4) and (6.6) for the next iteration of the algorithm, the minimax approximation to the set of equations generated by (6.5) is such that

$$(-1)^n \varepsilon' + \sum_{k=1}^N f_k h(t_n' - kT) = g(t_n') \quad (6.8)$$

$$n = 1, \dots, N+1$$

The error ε' can be solved for using Cramer's rule,

$$\varepsilon' = \frac{\begin{vmatrix} g(t_1') & h(t_1'-T) & \dots & h(t_1'-NT) \\ g(t_2') & h(t_2'-T) & \dots & h(t_2'-NT) \\ \vdots & \vdots & & \vdots \\ g(t_{N+1}') & h(t_{N+1}'-T) & \dots & h(t_{N+1}'-NT) \end{vmatrix}}{\begin{vmatrix} 1 & h(t_1'-T) & \dots & h(t_1'-NT) \\ -1 & h(t_2'-T) & \dots & h(t_2'-NT) \\ \vdots & \vdots & & \vdots \\ (-1)^{N+2} & h(t_{N+1}'-T) & \dots & h(t_{N+1}'-NT) \end{vmatrix}} \quad (6.9)$$

* Cheney, E.W., Introduction to Approximation Theory, McGraw-Hill, 1966.

Expanding each determinant along its first column and denoting the $N+1$ minors M_n for $n = 1, \dots, N+1$, the expression for ϵ' becomes

$$\epsilon' = \left[\sum_{n=1}^{N+1} g(t_n') (-1)^{n+1} M_n \right] / \sum_{n=1}^{N+1} M_n \quad (6.10)$$

If $g(t) = \sum f_n h(t-nT)$ then $\epsilon' = 0$ and so $g(t)$ can be replaced by $r(t) = g(t) - \sum f_n h(t-nT)$ in (6.10). Furthermore, since $r(t_n') = -r(t'_{n+1})$ and since the minors M_n all have the same sign*, ϵ' becomes

$$\epsilon' = \sum_{n=1}^{N+1} |M_n| |r(t_n')| / \sum_{n=1}^{N+1} |M_n|. \quad (6.11)$$

Choosing θ_n ,

$$\theta_n = |M_n| / \sum_{k=1}^{N+1} |M_k| \quad (6.12)$$

then

$$\sum \theta_n = 1 \quad \text{and} \quad 0 < \theta_n < 1 \quad (6.13)$$

for $n = 1, \dots, N+1$. By (6.13)

$$\epsilon' = \sum \theta_n |r(t_n')| \geq \min_{1 \leq n \leq N+1} |r(t_n')| = \alpha. \quad (6.14)$$

Knowing that $\epsilon' \geq \alpha$ allows the inequality

$$\begin{aligned} \alpha' - \alpha &\geq \epsilon' - \alpha = \sum_{n=1}^{N+1} \theta_n |r(t_n')| - \alpha \sum_{n=1}^{N+1} \theta_n \\ &= \sum_{n=1}^{N+1} \theta_n (|r(t_n')| - \alpha) \end{aligned} \quad (6.15)$$

* See section A.4.

because $\alpha' \geq \varepsilon'$ and $\sum_n \theta_n = 1$ by (6.7) and (6.13). Assuming that there exists a θ such that $(1-\theta) \leq \min_{1 \leq n \leq N+1} (\theta_n)$ then by (6.6)

$$\alpha' - \alpha \geq (1-\theta)(\beta - \alpha) \geq (1-\theta)(\varepsilon^* - \alpha). \quad (6.16)$$

Since $\varepsilon^* \leq \beta$ (see (6.7)). Furthermore, by (6.16)

$$(\varepsilon^* - \alpha') = (\varepsilon^* - \alpha) - (\alpha' - \alpha) \leq (\varepsilon^* - \alpha) - (1-\theta)(\varepsilon^* - \alpha) = \theta(\varepsilon^* - \alpha). \quad (6.17)$$

Since it is known that $0 < \theta < 1$, (6.16), gives a relationship for the distance from ε^* to α for successive iterations. If the sequence of α 's are labelled $\alpha^{(1)}, \alpha^{(2)}, \dots$ then

$$\varepsilon^* - \alpha^{(k)} \leq \theta^k (\varepsilon^* - \alpha^{(1)}) = B\theta^k \quad (6.18)$$

For $B = (\varepsilon^* - \alpha^{(1)})$ a constant, the sequence $\alpha^{(k)}$ converges uniformly and monotonically to ε^* from below.

Furthermore, since $\alpha' - \alpha \geq (1-\theta)(\beta - \alpha)$ by (6.16) for each term of the sequence $\beta^{(1)}, \beta^{(2)}, \dots$

$$\begin{aligned} \beta^{(k)} - \varepsilon^* &\leq \beta^{(k)} - \alpha^{(k)} \leq (1-\theta)^{-1} [\alpha^{(k+1)} - \alpha^{(k)}] \\ &\leq (1-\theta)^{-1} (\varepsilon^* - \alpha^{(k)}) \leq (1-\theta)^{-1} B\theta^k = C\theta^k. \end{aligned} \quad (6.19)$$

The sequence $\beta^{(k)}$ also converges uniformly and monotonically to ε^* , however, it does so from above. Thus, the algorithm converges to the best

7. Numerical Examples and Suggested Future Research

7.1 Examples

Two examples are presented. The first example shows a Gaussian function being equalized to $\sin(t)/t$. Specifically,

$$\begin{aligned} h(t) &= .337 \exp(-t^2/27.6) \\ g(t) &= \sin(t)/t. \end{aligned} \quad (7.1)$$

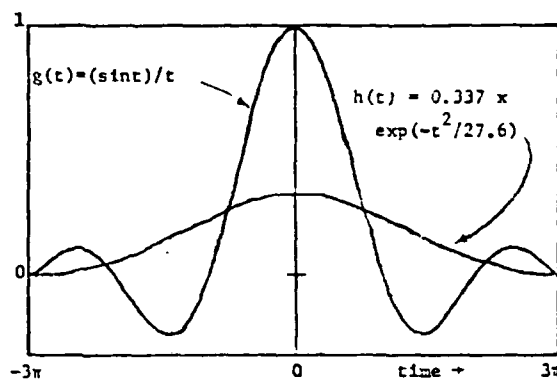
Figure 7.1(a) shows a plot of $h(t)$ and $g(t)$ in the interval $[-3\pi, 3\pi]$. Figures 7.1(b) and 7.1(c) show plots of $g(t)$ and the equalized versions of $h(t)$, $f_6 * h$ for a six tap filter and $f_8 * h$ for an eight tap filter, respectively. The tap spacings for f_6 and f_8 are $T=\pi$ and $T=3\pi/4$, respectively.

The set of tap-weights computed for f_6 and f_8 are shown with their associated minimax error in Table 7.1.

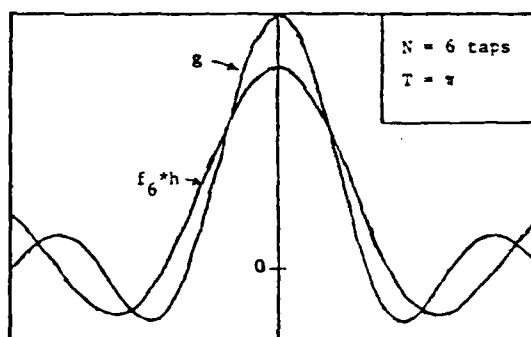
f_6		f_8	
ϵ	.208	ϵ	.021
$f(1)$	3.038	$f(1)$	-13.637
$f(2)$	-5.828	$f(2)$	39.696
$f(3)$	3.783	$f(3)$	-52.256
$f(4)$	3.776	$f(4)$	25.830
$f(5)$	-5.823	$f(5)$	25.811
$f(6)$	3.036	$f(6)$	-52.241
		$f(7)$	39.688
		$f(8)$	-13.635

TABLE 7.1 Tap-Weights and Error for f_6 and f_8

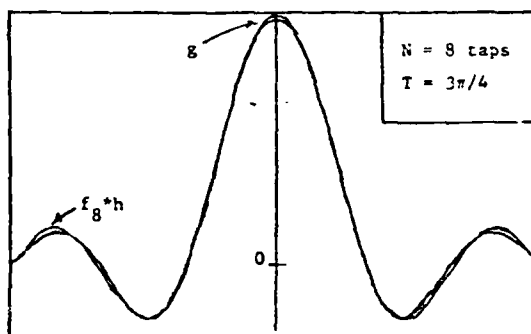
7.1(a)



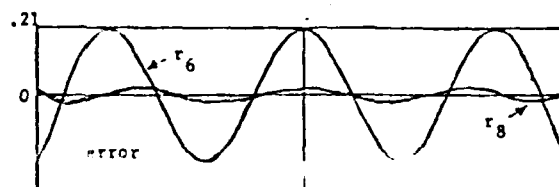
7.1(b)



7.1(c)



7.1(d)

Figure 7.1 Equalization of Gaussian Function to $\sin(t)/t$.

Note that the error for the eight tap filter is an order of magnitude smaller than the error for the six tap filter. The number of iterations to find f_6 and f_8 was five.

Figure 7.1(d) shows the error functions r_6 and r_8 for both the six and eight tap filters when $r(t)$ is evaluated at the values of f_6 and f_8 shown in table 7.1.

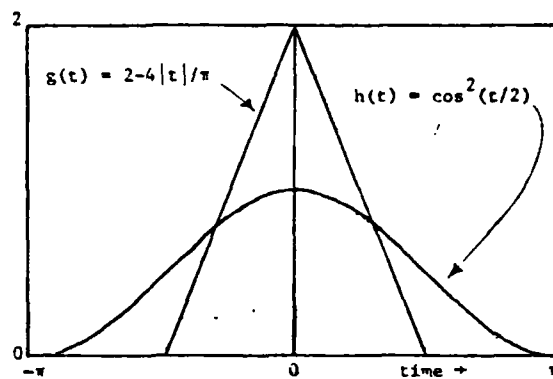
The second example shows a raised cosine function being equalized to a triangular pulse. In this case

$$\begin{aligned} h(t) &= \cos^2(t/2) \\ g(t) &= 2-4|t|/\pi. \end{aligned} \quad (7.2)$$

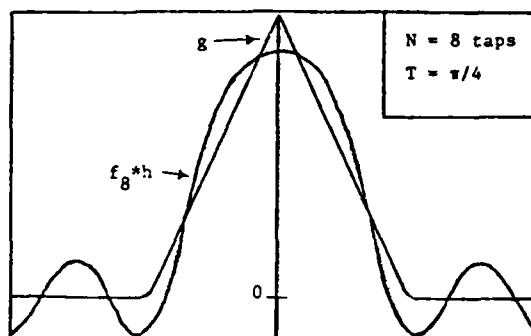
Figure 7.2(a) shows a plot of $h(t)$ and $g(t)$ in the interval $[-\pi, \pi]$. Figures 7.2(b) and 7.2(c) show plots of $g(t)$ and the equalized versions of $h(t)$, $f_8 * h$ for an eight tap filter and $f_{12} * h$ for a twelve tap filter, respectively. The tap spacings for f_8 and f_{12} are $T = \pi/4$ and $T = \pi/6$, respectively.

The set of tap weights computed for f_8 and f_{12} are shown in Table 7.2 along with their associated minimax errors. Again, note that the error is substantially reduced for the longer length filter.

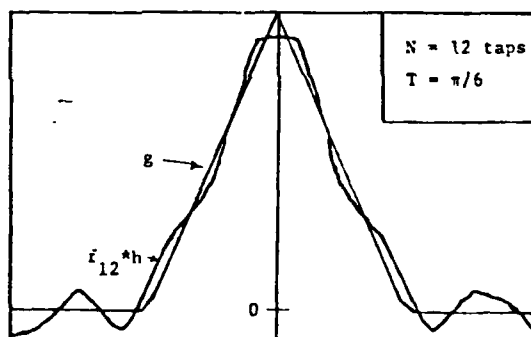
7.2(a)



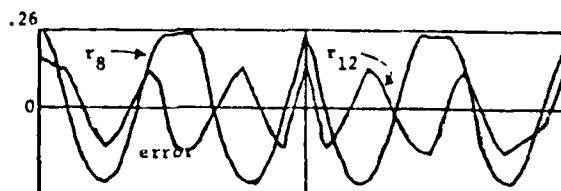
7.2(b)



7.2(c)



7.2(d)

Figure 7.2 Equalization of Raised Cosine Function to Triangular Pulse

f_8		f_{12}	
ϵ	.263	ϵ	.200
$f(1)$	4.281	$f(1)$	7.303
$f(2)$	-12.982	$f(2)$	-16.846
$f(3)$	15.664	$f(3)$	15.261
$f(4)$	-6.358	$f(4)$	-14.068
$f(5)$	-6.362	$f(5)$	18.958
$f(6)$	15.666	$f(6)$	-9.907
$f(7)$	-12.982	$f(7)$	-9.914
$f(8)$	4.281	$f(8)$	18.962
		$f(9)$	-14.069
		$f(10)$	15.262
		$f(11)$	-16.846
		$f(12)$	7.303

TABLE 7.2 Tap-Weights and Error for f_8 and f_{12}

Figure 7.2(d) shows the error functions r_8 and r_{12} for the eight and twelve tap filters when $r(t)$ is evaluated at the values of f_8 and f_{12} shown in Table 7.2. The number of iterations needed to find f_8 was four and the number needed for f_{12} was five.

7.2 Future Research

The algorithms discussed in this work are dependent upon the Haar condition being satisfied (see section 4.5). This is not to imply that the minimax solution does not exist when the Haar condition is not satisfied, but the algorithm fails to find the solution if this is the case. An algorithm which successfully operates independent of the Haar condition is necessary before minimax equalizers can be universally implemented.

Investigation of the stability of the Ascent and Remez algorithms in the presence of various types of noise is a topic of interest. Noise is a practical problem and a comparative study of the effects of noise on minimax equalizers with respect to other types of equalizers is important.

Finally, there seems to be a very strong potential for minimax equalizers in adaptive equalization. Adaptive equalization is the technique of varying the tap-weights of a transversal filter to compensate for variation in the communication channel or system being equalized. Since the Ascent algorithm only works with $N+1$ pieces of data at a time, the equalization of a system or channel impulse response is determined by a particular subset of $N+1$ pieces of the total input data. An adaptive equalizer in the minimax sense promises to be very fast for the following reasons: the input of new data never increases the difficulty of the problem and new pieces of data do not necessarily change the current solution. Only if the magnitude of the error curve associated with the new piece of data has error unbounded by the current solution is it necessary to recompute the solution.

BIBLIOGRAPHY

BIBLIOGRAPHY

- Cheney, E.W., Introduction to Approximation Theory, McGraw-Hill, 1966.
- Evans, R.J., Fortman, T.E., Cantoni, A., "Envelope-Constrained Filters, Part 1: Theory and Applications, Part 2: Adaptive Structures," IEEE Trans. on Information Theory, Vol. IT-23, no. 4, pp. 421-444, July 1977.
- Farden, D.C., Scharf, L.L., "Statistical Design of Nonrecursive Digital Filters," IEEE Trans. on Acoustics, Speech, and Signal Processing, June 1974.
- Guida, A., "Optimum Tapped Delay Line for Digital Signals," IEEE Trans. on Commun., vol. COM-21, pp. 277-283, April 1973.
- Hildebrand, F.C., Methods of Applied Mathematics, Prentice-Hall, New Jersey, 1965.
- Hunt, B.R., "Deconvolution of Linear Systems by Constrained Regression and its Relationship to Wiener Theory," IEEE Trans. Auto. Control, vol. AC-17, no. 5, pp. 703-705, Octo. 1972.
- Hunt, B.R., "A Theorem on the Difficulty of Numerical Deconvolution," IEEE Trans. on Audio and Electroacoustics, vol. AU-20, pp. 94-95, March 1972.
- McClellan, J.H., Parks, T.W., Rabiner, L.R., "A Computer Program for Designing Optimum FIR Linear-Phase Digital Filters," IEEE Trans. on Audio and Electroacoustics, vol. AU-21, no. 6, pp. 506-526, Dec. 1973.
- McClellan, J.H., Parks, T.W., "A Unified Approach to the Design of Optimum FIR Linear-Phase Digital Filters," IEEE Trans. on Circuit Theory, vol. CT-20, no. 6, pp. 697-701, Nov. 1973.
- Mersereau, R.M., Schafer, R.W., "Comparative Study of Iterative Deconvolution Algorithms," IEEE Conference Record, ICASS-78, pp. 192-194, April 1978.
- O'nan, M., Linear Algebra, Harcourt Brace Jovanovich, 1976.
- Oppenheim, A.V., Schafer, R.W., Digital Signal Processing, Prentice-Hall, New Jersey, 1975.
- Padulo, L., The Time-Domain Inversion of Convolution, Ph.D. Thesis, Georgia Institute of Technology, May 1966.
- Preis, D., "Audio Signal Processing with Transversal Filters," IEEE Conference Record, ICASSP-79, pp. 310-313, April 1979.

- Preis, D., "Linear Distortion," Journal of the Audio Engineering Society,
vo. 24, no. 5, pp. 346-367, June 1976.
- Preis, D., "Envelope-Constrained Time-Domain Deconvolution for Transversal-
Filter Equalizers," Electron. Lett., vol. 14, pp. 37-38, Jan. 1978.
- Preis, D., "Least-Squares Time-Domain Deconvolution for Transversal-Filter
Equalizers," Electron. Lett., vol. 13, pp. 356-357, June 1977.
- Proakis, J.G., "Advances in Equalization for Intersymbol Interference,"
Advances in Communication Systems, vol. 4, pp. 123-198,
Academic Press, 1975.
- Remez, E. Ya., General Computational Methods of Chebyshev Approximation.
The Problems with Linear Real Parameters, Book 1, U.S. Atomic
Energy Commission, AEC-tr-4491/1, May 1962
- Rice, J.R., The Approximation of Functions, Vol I.: Linear Theory,
Addison-Wesley, Reading, MA., 1964.
- Rudin, H., "Automatic Equalization Using Transversal Filters," IEEE Spectrum,
vol. 4, pp. 53-59, Jan. 1967.
- Widrow, B., "Adaptive Filters," Aspects of Network and System Theory, Part IV,
pp. 563-587, Holt Reinhart, and Winston, 1971.

APPENDIX

APPENDIX

A.1 Proof that the Convex Hull Contains Zero

For the overspecified system of linear equations as defined in (2.2),

$$\begin{bmatrix} a_{11} & \dots & a_{1N} \\ a_{21} & \dots & a_{2N} \\ . & & . \\ . & & . \\ . & & . \\ a_{M1} & \dots & a_{MN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ . \\ . \\ . \\ b_M \end{bmatrix} \quad (\text{A.1})$$

each row of the matrix can be represented as an error function

$$r_m(x) = \left| b_m - \sum_{n=1}^N a_{mn} x_n \right| \quad (\text{A.2})$$

which can be written in inner product form,

$$r_m(x) = \left| b_m - \langle A_m, x \rangle \right| \quad (\text{A.3})$$

where $A_m = (a_{m1}, a_{m2}, \dots, a_{mN})$ and $x = (x_1, \dots, x_N)$.

It will now be shown that the minimax solution, y , of the overspecified system in (A.1) is such that the set of vectors

$$\{A_m \mid r_m(y) = ||r(x)||\} \quad (\text{A.4})$$

contains zero in its convex hull.

The norm of $r(x)$ is defined as

$$||r(x)|| = \max_{1 \leq m \leq M} \{r_m(x)\} \quad (\text{A.5})$$

and an example of the norm of $r(x)$ for the case of one unknown is shown as the crosshatched portion of Fig. A.1.

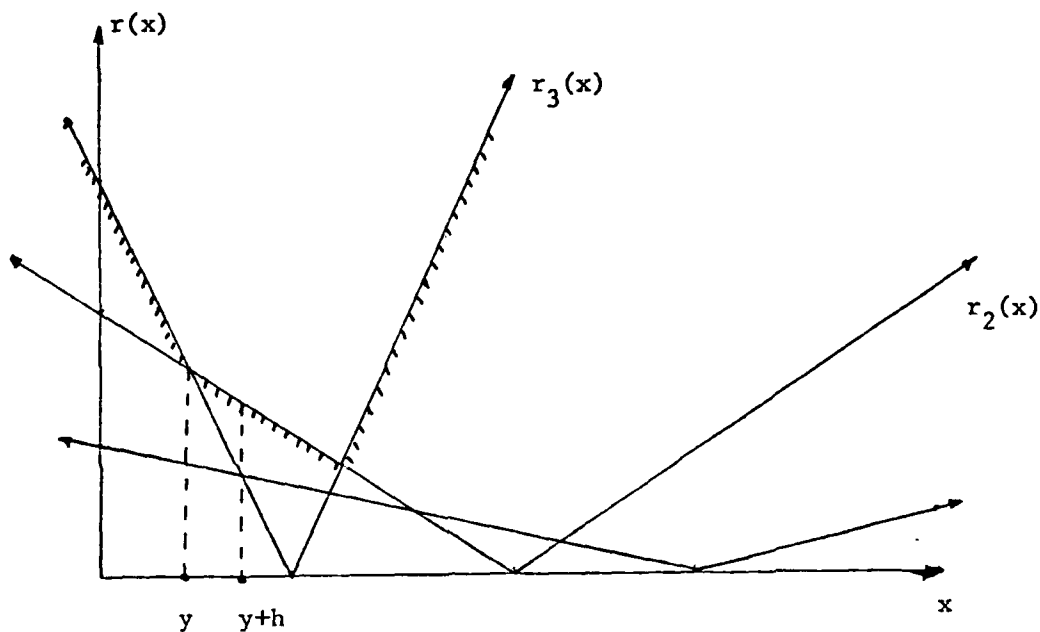


Figure A.1 Proof of the Convex Hull Containing Zero

Assuming y is not the minimax solution it will be shown that zero is not contained in the convex hull of the set in (A.4) which will constitute a proof by asserting the contrapositive.

If y is not the solution (refer to Fig. A.1) then y is not the minimum value of $||r(x)||$ and for some vector h ,

$$||r(y+h)|| < ||r(y)|| \quad (\text{A.6})$$

For any error function $r_m(x)$ which is part of the function $r(x)$ at the point y ,

$$r_m(y+h) \leq ||r(y+h)|| < ||r(y)|| = r_m(y) \quad (A.7)$$

or by (A.3)

$$|b_m - \langle A_m, y+h \rangle| < |b_m - \langle A_m, y \rangle| \quad (A.8)$$

Rewriting (A.8)

$$|b_m - \langle A_m, y \rangle - \langle A_m, h \rangle| < |b_m - \langle A_m, y \rangle| \quad (A.9)$$

and for (A.9) to be true

$$\langle A_m, h \rangle > 0. \quad (A.10)$$

However, for zero to belong to the convex hull of the A_m belonging to the set in (A.4)

$$\sum \theta_m A_m = 1 \quad (A.11)$$

where

$$0 \leq \theta_m < 1 \quad \text{and} \quad \sum \theta_m = 1 \quad (A.12)$$

and by (A.11)

$$\sum \theta_m \langle A_m, h \rangle = \langle \sum \theta_m A_m, h \rangle = \langle 1, h \rangle = 0 \quad (A.13)$$

which is not possible if all the inner products $\langle A_m, h \rangle$ are greater than zero as in (A.10).

Therefore, it has been shown by contradiction that if the set in (A.4) contains zero in its convex hull, then y must be the minimum point of the function $||r(x)||$.

A.2 Property of the Convex Hull of a Set

A set S is convex if for any two vectors y and z which belong to S the linear combination

$$\theta y + (1-\theta)z, \quad 0 \leq \theta \leq 1 \quad (\text{A.14})$$

also belongs to S . It will be shown that for any set of vectors $\{f_1, \dots, f_N\}$ which belong to a convex set S , the linear combination

$$\sum_{n=1}^N \theta_n f_n \quad \text{for} \quad \sum_{n=1}^N \theta_n = 1, \quad 0 \leq \theta_n \leq 1 \quad (\text{A.15})$$

also belongs to S .

Rewriting (A.15)

$$\sum_{n=1}^N \theta_n f_n = \sum_{n=1}^{N-1} \theta_n f_n + \theta_N f_N \quad (\text{A.16})$$

and defining L ,

$$L = \sum_{n=1}^{N-1} \theta_n \quad (\text{A.17})$$

equation (A.16) can be rewritten

$$\sum_{n=1}^N \theta_n f_n = L \sum_{n=1}^{N-1} \frac{\theta_n f_n}{L} + \theta_N f_N. \quad (\text{A.18})$$

Since

$$0 \leq \frac{\theta_n}{L} \leq 1, \quad \sum_{n=1}^{N-1} \frac{\theta_n}{L} = 1, \quad L + \theta_N = 1 \quad (\text{A.19})$$

then by induction on $N-1$, (A.18), is proved to be a member of the set S .

A.3 Proof that the Minimax Solutions are Determined by Subsets of $N+1$ Elements

It will now be shown that the number of elements contained in the set (A.4) need not be greater than $N+1$. From the discussion in section A.1 it is known that zero is contained in the convex hull of the set,

$$\{A_m | r_m(y) = ||r(x)||\} . \quad (\text{A.20})$$

The proof will be made by showing that a vector Z which belongs to the convex hull of a set S is expressible as a convex linear combination of $N+1$ or fewer elements of S .

Thus, Z is expressed as

$$Z = \sum_{n=0}^K \theta_n V_n \quad (\text{A.21})$$

where

$$n = 0, 1, \dots, K, \quad \sum_{n=0}^K \theta_n = 1 \quad (\text{A.22})$$

and the V_n belong to S . If K is minimal then each θ_n in (A.22) is strictly greater than zero.

Defining a set of vectors Z_n ,

$$Z_n = V_n - Z \quad (A.23)$$

for $n = 1, \dots, K$ the Z_n are linearly dependent because

$$\sum_{n=0}^K \theta_n Z_n = \sum_{n=0}^K \theta_n (V_n - Z) = \sum_{n=0}^K \theta_n V_n - Z \sum_{n=0}^K \theta_n = Z - Z = 0 \quad (A.24)$$

Furthermore, if $K > N$ then the Z_n are linearly dependent since there are more than N vectors in a space of dimension N .

If Z_0 is removed from (A.24) then suppose a set of not all zero coefficients α_n exists such that

$$\sum_{n=1}^K \alpha_n Z_n = 0. \quad (A.25)$$

Defining $\alpha_0 = 0$, for all λ

$$\sum_{n=0}^K (\theta_n + \lambda \alpha_n) Z_n = \sum_{n=0}^K \theta_n Z_n + \lambda \sum_{n=0}^K \alpha_n Z_n = 0 + (0)\lambda = 0. \quad (A.26)$$

Now λ is chosen so that $|\lambda|$ is a minimum and so that one of the coefficients in (A.26) disappears, i.e.

$$\theta_n + \lambda \alpha_n = 0. \quad (A.27)$$

The remaining coefficients in (A.26) are all non-negative since $|\lambda|$ is a minimum and at least one is non-zero since $\theta_0 + \lambda \alpha_0 = \theta_0 > 0$. Using this λ a term in (A.26) drops out. Replacing Z_n with (A.23) in (A.26) yields

$$\sum_{n=0}^K (\theta_n + \lambda \alpha_n) Z_n = \sum_{n=0}^K (\theta_n + \lambda \alpha_n) (V_n - Z) = 0. \quad (\text{A.28})$$

Equation (A.28) implies

$$Z \sum_{n=0}^K (\theta_n + \lambda \alpha_n) = \sum_{n=0}^K (\theta_n + \lambda \alpha_n) V_n \quad (\text{A.29})$$

which may be written

$$Z = \left[\sum_{n=0}^K (\theta_n + \lambda \alpha_n) V_n \right] / \left[\sum_{n=0}^K (\theta_n + \lambda \alpha_n) \right] \quad (\text{A.30})$$

which is valid since the coefficient of Z on the left-hand side of (A.29) is non-zero.

Since one of the terms in (A.30) is zero Z has been expressed as a sum of K terms instead of $K+1$ terms as in (A.21). Thus the assumed minimality of K has been shown false and it must be that $K \leq N$. For $K \leq N$ the number of terms in (A.21) is $N+1$ or fewer.

A.4 Alternation Theorem

For the continuous equalization case the error function is defined as in (5.2)

$$r(t) = g(t) - \sum_{n=1}^N f_n h(t-nT). \quad (\text{A.31})$$

For the norm of $r(t)$ to be minimized it will be shown that for a certain set of $N+1$ points $t_1 < \dots < t_{N+1}$ from the interval $[a, b]$ that

$$r(t_n) = -r(t_{n+1}) = ||r(t)||. \quad (\text{A.32})$$

For an interval $[a,b]$ a set of $N+1$ distinct points $t_1 < \dots < t_{N+1}$ is a vector t

$$t = (t_1, \dots, t_{N+1}) \quad (\text{A.33})$$

which has an associated vector $\hat{t}$

$$\hat{t}_n = (h(t_n - T), \dots, h(t_n - NT)) \quad (\text{A.34})$$

where t_n in (A.34) is a component of the vector t in (A.33).

The Haar condition (see section 4.5) states that every set of N vectors of the form (A.34) must be linearly independent. This is the same as requiring that the determinant

$$D[t_1, \dots, t_{N+1}] = \begin{vmatrix} h(t_1 - T) & \dots & h(t_1 - NT) \\ h(t_2 - T) & & h(t_2 - NT) \\ \cdot & & \cdot \\ \cdot & & \cdot \\ h(t_{N+1} - T) & \dots & h(t_{N+1} - NT) \end{vmatrix} \quad (\text{A.35})$$

be non-zero.

Since determinants are continuous functions (determinants are defined by polynomial expansions) it can be shown that all determinants of the form (A.35) have the same sign. Suppose that

$$D[t_1, \dots, t_{N+1}] < 0 < D[u_1, \dots, u_{N+1}] \quad (\text{A.36})$$

for $u_1 < \dots < u_{N+1}$. Since the determinant is a continuous operator there

exists a λ such that $0 \leq \lambda \leq 1$ and

$$D[\lambda t_1 + (1-\lambda)u_1, \dots, \lambda t_{N+1} + (1-\lambda)u_{N+1}] = 0 \quad (\text{A.37})$$

If the determinant in (A.37) is zero then for some distinct m and n

$$\lambda t_n + (1-\lambda)u_n = \lambda t_m + (1-\lambda)u_m \quad (\text{A.38})$$

or

$$\lambda(t_n - t_m) = (1-\lambda)(u_m - u_n) \quad (\text{A.39})$$

From (A.39) $t_n - t_m$ is of opposite sign from $u_n - u_m$ which is a contradiction of the assumption that the t_n and u_n are similarly ordered.

For $\|r(t)\|$ to be a minimum it is known from sections A.1 and A.3 that zero must belong to the convex hull of some subset of $N+1$ or fewer vectors $\{\sigma_1 \hat{t}_1, \dots, \sigma_{N+1} \hat{t}_{N+1}\}$. Thus, from section A.2

$$\sum_{n=1}^{N+1} \theta_n \sigma_n \hat{t}_n = 0 \quad (\text{A.40})$$

for

$$0 \leq \theta_n \leq 1, \quad \sum_{n=1}^{N+1} \theta_n = 1. \quad (\text{A.41})$$

Rewriting (A.40)

$$\hat{t}_1 = \sum_{n=2}^{N+1} -\frac{\theta_n \sigma_n}{\theta_1 \sigma_1} \hat{t}_n \quad (\text{A.42})$$

and solving (A.42) using Cramer's rule

$$-\frac{\theta_n \sigma_n}{\theta_1 \sigma_1} = \frac{D[t_2, \dots, t_{n-1}, t_1, t_{n+1}, \dots, t_{N+1}]}{D[t_2, \dots, t_{N+1}]} \quad (\text{A.43})$$

Using the property of determinants which states that a column interchange changes the sign of the determinant (A.43) becomes

$$\frac{-\theta_n \sigma_n}{\theta_1 \sigma_1} = \frac{D[t_1, \dots, t_{n-1}, t_{n+1}, \dots, t_{N+1}]}{D[t_2, \dots, t_{N+1}]} (-1)^{n-1} \quad (\text{A.44})$$

and since both determinants in (A.44) are similarly ordered their signs are the same. Thus,

$$\text{sgn} \left(\frac{-\theta_n \sigma_n}{\theta_1 \sigma_1} \right) = (-1)^{n-1} \quad (\text{A.45})$$

and because each θ_n is positive.

$$\text{sgn } \sigma_n = (-1)^n \text{sgn } \sigma_1. \quad (\text{A.46})$$

Equation (A.46) shows that the σ_n must alternate in sign thus proving (A.32).

A.5 Program Listings

The following is a listing of the ascent algorithm and the Remez algorithm. The subroutines MINV and RIMI used in the algorithms are IBM scientific subroutine programs and are not listed here. Both algorithms are programmed in Fortran IV on a DEC-10 computer.

A.5.1 Ascent Algorithm

```

C -----
C PROGRAM ASCENT
C
C PURPOSE
C   TO OBTAIN THE BEST SET OF TAP-WEIGHTS, F(J),
C   IN THE MINIMAX SENSE FOR AN OVERSPECIFIED
C   SYSTEM OF LINEAR EQUATIONS GENERATED BY CON-
C   VOLUTION SUM. HX.DAT AND GX.DAT ARE THE UN-
C   EQUALIZED AND DESIRED RESPONSES, RESPECTIVELY,
C   AND MUST BE SPECIFIED BY THE USER.
C
C USAGE
C   .EX ASCENT, MINV
C
C DESCRIPTION OF PARAMETERS
C
C   MINV -SUBROUTINE WHICH FINDS THE INVERSE OF
C         AN (N X N) MATRIX PROVIDED THE MATRIX
C         IS NON-SINGULAR.
C
C REMARKS
C   THE PROCEDURE ASSUMES THAT THE ROWS OF THE
C   MATRIX REPRESENTING THE OVERSPECIFIED SYSTEM
C   OF LINEAR EQUATIONS SATISFY THE HAAR CONDITION.
C
C METHOD
C   THE SOLUTION IS FOUND BY SUCCESSIVE SOLUTIONS
C   TO SUBSYSTEMS OF N+1 EQUATIONS. SINCE THERE
C   ARE A FINITE NUMBER OF SUCH SUBSYSTEMS AND
C   SINCE THE ERROR GROWS FOR EACH ITERATION THE
C   ALGORITHM IS GUARANTEED TO FIND THE SOLUTION
C   IN A FINITE NUMBER OF ITERATIONS.
C
C -----
C INITIALIZATION
C -----
C   DIMENSION A(8,2),B(3,3),THETA(3),Y(3)
C   DIMENSION BR(8),R(8),AMBDA(3),JS(3),JL(3),M(3)
C   REAL MU
C   DATA MR,MC,ME/8,2,3/
C   CALL IFILE(21,'HX.DAT')
C   CALL IFILE(22,'GX.DAT')
C   DO 10 I=1,ME
C     JS(I)=I
10

```

```

C -----
C   CREATE SYSTEM MATRIX
C   FROM HX.DAT DATA FILE
C -----

      J=1
      DO 30 I=1,MR
      IF(I.GT.MR-MC+1) GO TO 20
      READ(21,1)A(I,J)
      GO TO 30
20    A(I,J)=0.
30    CONTINUE
      DO 50 I=1,MR
      DO 50 J=2,MC
      K=I-J+1
      IF(K.LE.0) TO TO 40
      IF(K.GT.MR-MC+1) GO TO 40
      A(I,J)=A(I-1,J-1)
      GO TO 50
40    A(I,J)=0.
50    CONTINUE
      DO 55 I=1,MR
C -----
C   COMPUTE SIGNS WHICH FORCE ZERO
C   TO LIE IN THE CONVEX HULL OF THE N+1 VECTORS
C -----
55    READ(22,1)BR(I)
      DO 60 I=1, ME
60    B(1,I)=1.
      DO 70 K=2,ME
      DO 70 L=1,ME
      N=JS(L)
70    B(K,L)=A(N,K-1)
      IHOLD=ME
      CALL MINV(B,IHOLD,DUM,JL,M)
      DO 80 I=1,ME
80    THETA(I)=B(I,1)/ABS(B(I,1))
C -----
C   INITIAL COMPUTATION OF MINIMAX
C   SOLUTION FOR SUBSYSTEM OF N+1 EQUATIONS
C -----

      DO 90 I=1,ME
90    B(I,1)=THETA(I)
      DO 100 K=2,ME
      DO 100 L=1,ME
      N=JS(L)
100   B(L,K)=A(N,K-1)
      IHOLD=ME
      CALL MINV(B,IHOLD,DUM,JL,M)
      IF(ABS(DUM).EQ.0.) WRITE(5,2)
110   DO 120 K=1,ME
      Y(K)=0.
      DO 120 L=1,ME
      N=JS(L)
120   Y(K)=Y(K)+B(K,L)*BR(N)

```

```

C -----
C EVALUATE M ERROR EQUATIONS AT
C REALTIVE MINIMAX SOLUTION
C -----

DO 140 K=1,MR
R(K)=0.
DO 130 L=2,ME
130 R(K)=R(K)+A(K,L-1)*Y(L)
140 R(K)=R(K)-BR(K)

C -----
C FIND THE ERROR EQUATION WITH MAXIMUM
C MAGNITUDE AND DENOTE ITS INDEX AS ALPHA
C -----

JALPHA=1
X=ABS(R(1))
DO 150 I=2,MR
IF(X.GT.ABS(R(I))) GO TO 150
X=ABS(R(I))
JALPHA=I
150 CONTINUE

C -----
C HAS A GLOBAL MINIMAX SOLUTION BEEN FOUND?
C -----

DO 160 I=1,ME
160 IF(JALPHA.EQ.JS(I)) GO TO 220
IF(X.LE.ABS(Y(1))) TO TQ 220

C -----
C THE GLOBAL MINIMAX SOLUTION HAS NOT BEEN
C FOUND. EXPRESS THE VECTOR WITH INDEX JALPHA AS
C A LINEAR COMBINATION OF THE N+1 OLD VECTORS
C -----

MU=R(JALPHA)/X
DO 180 K=1,ME
AMBDA(K) = 0.
DO 170 L=1,MC
170 AMBDA(K)=AMBDA(K)+A(JALPHA,L)*B(L+1,K)
180 AMBDA(K)=AMBDA(K)+MU*B(1,K)

C -----
C DETERMINE WHICH OF THE N+1 OLD VECTORS
C WILL BE EXCHANGED FOR VECTOR WITH INDEX JALPHA
C -----

Q=MU*AMBDA(1)/B(1,1)
NBETA=1
DO 190 I=2,ME
IF(Q.GT.MU*AMBDA(I)/B(1,I)) GO TO 190
Q=MU*AMBDA(I)/B(1,I)
NBETA=I
190 CONTINUE

```

```

C -----
C REVERSE INVERSE MATRIX
C -----

      DO 200 K=1,ME
200   B(K,NBETA)=B(K,NBETA)/AMBDA(NBETA)
      DO 210 J=1,ME
      IF(J.EQ.NBETA) GO TO 210
      DO 210 K=1,ME
      B(K,J)=B(K,J)-AMBDA(J)*B(K,NBETA)
210   CONTINUE

C -----
C RE-INITIALIZE
C -----

      JS(NBETA)=JALPHA
      GO TO 110

C -----
C INPUT-OUTPUT
C -----

220   DO 230 I=1,ME
230   WRITE(5,3)Y(I)
1     FORMAT(F)
2     FORMAT(1X,'DET OF MATRIX IS ZERO')
3     FORMAT(1X,1F10.8)

C -----
C END OF PROGRAM
C -----

      CALL EXIT
      END

```

A.5.2 Remez Algorithm

```

C -----
C
C PROGRAM REMEZ
C
C PURPOSE
C
C   TO OBTAIN THE BEST SET OF TAP WEIGHTS, F(J), IN THE
C   MINIMAX SENSE FOR THE APPROXIMATION  $G(X)=F(1)H(X-T)+$ 
C    $\dots+F(N)H(X-NT)$ . THIS IS WHERE H(X) AND G(X)
C   ARE DEFINED FUNCTIONS AND THE COEFFICIENTS
C   F(1),...,F(N) ARE TO BE DETERMINED. THE APPROXIMATION
C   IS OVER A USER DEFINED INTERVAL [A,B] and T IS A USER
C   DEFINED CONSTANT.

```

```

C      USAGE
C      .EX REMEZ,GX,HX,RTMI,MINV
C
C      DESCRIPTION OF PARAMETERS
C      GX  -SUBROUTINE WHICH RETURNS A SINGLE OUTPUT
C           VALUE FOR A SINGLE INPUT VALUE ACCORDING
C           TO THE FUNCTION G(X).  THE FIRST LINE OF
C           SUBROUTINE GX MUST BE:
C
C           SUBROUTINE GX(Y,Z)
C
C           WHERE Y IS THE INPUT VALUE AND Z IS THE
C           RETURN VALUE.
C      HX  -SUBROUTINE WHICH RETURNS A SINGLE OUTPUT
C           VALUE FOR A SINGLE INPUT VALUE ACCORDING
C           TO THE FUNCTION H(X).  THE FIRST LINE OF
C           SUBROUTINE HX MUST BE:
C
C           SUBROUTINE HX(Y,Z)
C
C           WHERE Y IS THE INPUT VALUE AND Z IS THE
C           RETURN VALUE.
C      RTMI -SUBROUTINE WHICH WHEN GIVEN A FUNCTION R(X)
C           FINDS A ROOT BETWEEN TWO VALUES OF THE
C           FUNCTION R(X1) AND R(X2).  R(X1) AND R(X2)
C           MUST BE OF OPPOSITE SIGNS.
C      MINV -SUBROUTINE WHICH TAKES A (N X N) MATRIX AND
C           FINDS ITS INVERSE ASSUMING THAT THE DETERMINANT
C           OF THE MATRIX IS NONZERO.
C
C      REMARKS
C      THE PROCEDURE ASSUMES THAT THE GENERATED FUNCTIONS
C      H(X-T),...,H(X-NT) SATISFY THE HAAR CONDITION.  THE
C      CONDITION REQUIRES THAT EVERY SET OF N DISTINCT
C      VECTORS H(X(I)-T),...,H(X(I)-NT) FOR I=1,...,N
C      BE INDEPENDENT.
C
C      SUBROUTINES USED
C      GX AND HX ARE SUBROUTINES WHICH MUST BE WRITTEN
C      BY THE USER.  THEY ARE INPUT SUBROUTINES.
C      RTMI AND MINV ARE STANDARD SCIENTIFIC SUBROUTINES
C      WHICH CAN BE FOUND ON MOST COMPUTING SYSTEMS.
C
C      METHOD
C      SOLUTION OF APPROXIMATION FOR MINIMAX COEFFICIENTS
C      IS ACCOMPLISHED BY AN ITERATIVE PROCESS KNOWN
C      AS THE REMEZ ALGORITHM.  THE ALGORITHM SUCCESSIVELY
C      MINIMIZES THE LARGEST N+1 DISCRETE ERRORS ON THE
C      INTERVAL [A,B] UNTIL IT CONVERGES TO A BEST SOL-
C      TION.  FOR REFERENCE SEE: INTRODUCTION TO
C      APPROXIMATION THEORY, E.W. CHENEY, MCGRAW HILL 1966.
C      -----

```

```

C
C      INITIALIZE PROGRAM
C
      DIMENSION X(9),G(9),R(9),SIGMA(9),Z(10),H(9,8)
      COMMON TAU,N,F(9),ME,MC,MR
      REAL NORM
      EXTERNAL RX
      DATA N/8/,MR,MC,ME/9,8,9/
      DATA EPS/.001/,IEND/100/,STOP/0./
      CALL OFILE(21,'GDAT.DAT')
      CALL OFILE(22,'ADAT.DAT')
      CALL OFILE(23,'RDAT.DAT')
      CALL OFILE(24,'HDAT.DAT')

C
C      INPUT PARAMETERS, CREATE INITIAL X-VECTOR, AND COMPUTE
C      VALUES IN FUNCTION MATRICES
C
      WRITE(5,1)
      READ(5,2)A,B,TAU
      SINT=(B-A)/(N+2)
      DO 10 I=1,N+1
      AI=I*SINT
10      X(I)=A+AI
20      WRITE(5,3)
      WRITE(5,5)X
      DO 30 I=1,N+1
      XI=X(I)
      CALL GX(XI,GXI)
      G(I)=GX
      DO 30 J=1,N
      CNTRJ=(2*J-N-1)/2.
      XS=XI-CNTRJ*TAU
      CALL HX(XS,HXS)
30      H(I,J)=HXS

C
C      COMPUTE F-VECTOR AND THE INCURRED ERRORS
C
      CALL CHEBY(G,H,F)
      WRITE(5,4)
      WRITE(5,5)F
      DO 40 I=1,N+1
      XI=X(I)
40      R(I)=RX(XI)
      WRITE(5,6)
      WRITE(5,5)R

C
C      HAS A SOLUTION BEEN FOUND?
C
      TEST=1.-(STOP/F(1))
      IF(ABS(TEST).LT..001) GO TO 50
      STOP=F(1)
      GO TO 70
50      TINT=(B-A)/50.
      WRITE(5,5)
      WRITE(5,5)

```

```

WRITE(5,9)
WRITE(5,5)
DO 60 K=1,51
V=A+(K-1)*TINT
CALL GX(V,GV)
CALL HX(V,HTV)
RV=RX(V)
HV=GV-RV
WRITE(5,7)V,GV,HV,RV
WRITE(21,7)V,GV
WRITE(22,7)V,HV
WRITE(23,7)V,RV
WRITE(24,7)V,HTV
60  CONTINUE
GO TO 210

C
C  LOCATE ZEROES BETWEEN ERROR VALUES
C
70  DO 100 I=1,N
XI=X(I)
XIPO=X(I+1)
HOLD=IEND
SAVE=EPS
80  CALL RTMI(ZIPO,DUM,RX,XI,XIPO,EPS,IEND,IER)
IF(IER-1)100,90,100
90  IEND=2*IEND
EPS=EPS*2.
IF(IER.EQ.2)WRITE(5,81)
81  FORMAT(1X,'ERROR')
GO TO 80
100 Z(I+1)=ZIPO
IEND=HOLD
EPS=SAVE
Z(1)=A
Z(N+2)=B
WRITE(5,8)
WRITE(5,5)Z

C
C  EDIT X=VECTOR
C
DO 110 I=1,N+1
XI=X(I)
U=RX(XI)
110 SIGMA(I)=U/ABS(U)
NORM=0.
DO 130 I=1,N+1
ZI=Z(I)
ZIPO=Z(I+1)
STEP (ZIPO-ZI)/1000.
HOLD=0.
PMAZ=ZI
DO 130 J=1,1000

```

```

      T=ZI+J*STEP
      VAL=RX(T)
      SVAL=VAL*SIGMA(I)
      IF(SVAL.LE.HOLD) GO TO 120
      HOLD=SVAL
      PMAX=T
120   IF(ABS(VAL).LE.NORM) GO TO 130
      Y=T
      NORM=ABS(VAL)
130   X(I)=PMAX

C
C      RE-EDIT X-VECTOR
C
      DO 140 K=1,N+1
      XK=X(K)
      RXVAL=RX(XK)
140   IF(NORM.LE.ABS(RXVAL))K1=1
      IF(K1.EQ.1) GO TO 20
      X1=X(1)
      IF(Y.GT.X1) GO TO 160
      DO 150 K=2,N+1
      J=N+3-K
150   X(J)=X(J-1)
      X(1)=Y
      GO TO 20
160   DO 170 I=1,N+1
      IF(Y.LT.Z(I)) GO TO 190
170   IF(Y.LT.X(I)) GO TO 200
      DO 180 K=1,N
      X(K)=X(K+1)
      X(N+1)=Y
      GO TO 20
190   X(I)=Y
      GO TO 20
200   X(I-1)=Y
      GO TO 20

C
C      INPUT-OUTPUT
C
1   FORMAT(1X,'PLEASE INPUT A,B,TAU')
2   FORMAT(3F)
3   FORMAT(1X,'X-VECTOR:')
4   FORMAT(1X,'F-VECTOR:')
5   FORMAT(1X,10F10.3)
6   FORMAT(1X,'R(T) EVALUATED AT X:')
7   FORMAT(1X,3F10.3,6X,1F10.3)
8   FORMAT(1X,'Z-VECTOR:')
9   FORMAT(6X,'X          G(X)          H(X)*F(X)          R(X)')

C
C      END OF PROGRAM
C
210  CALL EXIT
      END

```

```

C -----
C
C COMPUTE THE ERROR FUNCTION:  R=G-<H,F>
C

```

```

      FUNCTION RX(XIN)
      COMMON TAU,N,F(9)
      XOUT=0.
      DO 220 J=1,N
      CNTRJ=(2*J-N-1)/2.
      B=XIN-CNTRJ*TAU
      CALL HX(B,C)
220    XOUT=XOUT+C*F(J+1)
      CALL GX(XIN,D)
      RX=D-XOUT
      RETURN
      END

```

```

C -----
C
C FIND THE MINIMAX SOLUTION TO SET OF N+1 LINEAR
C EQUATIONS IN N UNKNOWNNS
C

```

```

      SUBROUTINE CHEBY(BR,A,Y)
      DIMENSION A(9,8),B(9,9),THETA(9),Y(9),BR(9),JL(9),M(9)
      COMMON TAU,N,F(9),ME,MC,MR
      MO=0
230    DO 240 I=1,ME
240    B(I,1)=(-1)**(I+MO)
      DO 250 K=2,ME
      DO 250 L=1,ME
250    B(L,K)=A(L,K-1)
      IHOLD=ME
      CALL MINV(B,IHOLD,DUM,JL,M)
      DO 260 K=1,ME
      Y(K)=0.
      DO 260 L=1,ME
260    Y(K)=Y(K)+B(K,L)*BR(L)
      IF(Y(1).GE.0.) GO TO 270
      MO=1
      GO TO 230
270    RETURN
      END

```