

LEVEL II

2
(3)

Communications Research Centre

AD A 056814

AD No. _____
DDC FILE COPY

6 AN ITERATIVE TECHNIQUE FOR
DESIGNING FINITE IMPULSE RESPONSE
CHEBYSHEV MTI FILTERS
WITH NONLINEAR DELAY

by

10 R.W. Herring

This document has been approved
for public release and sale; its
distribution is unlimited.

DDC
RECEIVED
JUL 28 1978
F

14 CRC-TN-691

DEPARTMENT OF COMMUNICATIONS
MINISTÈRE DES COMMUNICATIONS

9
CRC TECHNICAL NOTE NO. 691

This work was sponsored by the Department of National Defence,
Research and Development Branch under Project No. 33C69.

12 58 p.

11

CANADA 78 07 25 031

404 957

OTTAWA MAR 1978

(3)

COMMUNICATIONS RESEARCH CENTRE

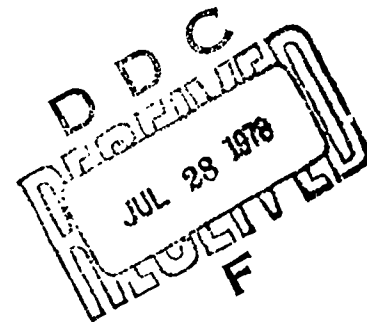
DEPARTMENT OF COMMUNICATIONS
CANADA

AN ITERATIVE TECHNIQUE FOR DESIGNING FINITE IMPULSE RESPONSE CHEBYSHEV MTI
FILTERS WITH NONLINEAR PHASE DELAY

by

R.W. Herring

(Radio and Radar Research Branch)



CRC TECHNICAL NOTE NO. 691

March 1978

OTTAWA

This work was sponsored by the Department of National Defence, Research and Development Branch under Project No. 33C69.

CAUTION

The use of this information is permitted subject to recognition of
proprietary and patent rights.

78 5 031

TABLE OF CONTENTS

ABSTRACT	1
1. INTRODUCTION	1
2. BENEFITS OF NONLINEAR PHASE	2
3. OUTLINE OF DESIGN PROCEDURE	9
4. DESIGNING THE PROTOTYPE FILTER	10
5. GAIN NORMALIZATION	13
6. COMPUTER PROGRAMS	13
7. SUMMARY	14
8. ACKNOWLEDGEMENT	14
9. REFERENCES	14
APPENDIX A - PROGRAM MPMTIPSF	16
APPENDIX B - PROGRAM MPMTIPSFO	25
APPENDIX C - SUBROUTINES	34

APPROPRIATE for	
NTIS	Wide Section ☒
DDC	Diff Section ☐
NOT DOCUMENTED	☐
JUSTIFICATION	
by	
DISTRIBUTION/AVAILABILITY NOTES	
by	SPECIAL
<i>A</i>	

AN ITERATIVE TECHNIQUE FOR DESIGNING FINITE IMPULSE RESPONSE CHEBYSHEV MTI FILTERS WITH NONLINEAR PHASE DELAY

by

R.W. Herring

ABSTRACT

A technique for designing finite-impulse-response (FIR) equiripple high-pass digital filters with nonlinear phase, suitable for use in radar moving-target-indication (MTI) systems, is described. The optimization of such filters in both the time and frequency domains is discussed, and comparisons are made between the spectral performance of nonlinear-phase filters and that of linear-phase filters designed to meet the same stopband bandwidth and attenuation specifications. In 92 out of the 94 cases examined, the transition bandwidth between the stopband is narrower for the nonlinear-phase filter. In the MTI application, this characteristic makes it possible to detect targets of lower radial velocity. Listings of Fortran programs for designing these nonlinear-phase filters are included.

1. INTRODUCTION

The object of this note is to describe a technique for designing finite-impulse-response (FIR) digital filters with nonlinear phase-delays for use in radar moving-target-indication (MTI) systems. The optimization of such filters in both the time and frequency domains is discussed, and comparisons are made between the spectral performance of nonlinear-phase filters and that of linear-phase filters designed to meet the same stopband bandwidth and attenuation specifications. In 92 out of 94 cases examined, the transition bandwidth between the filter stopband and passband is narrower for the nonlinear-phase filter. In the MTI application, this characteristic makes it possible to detect targets of lower radial velocity.

It is well known from digital signal theory (e.g., [1]) that delaying a sampled data sequence by a fixed amount imposes a shift or delay in the

relative phases of its spectral components which is linear with frequency. One of the great advantages of FIR filters is that they can be designed to have such a linear phase-delay, so that their only effects are to modify the magnitudes of the spectral components and to delay the sequence by a fixed amount.

In contrast, infinite-impulse-response (IIR) filters and nonlinear-phase FIR filters have nonlinear phase versus frequency characteristics. The use of nonlinear-phase filters may increase the complexity of any coherent post MTI signal processing, but since the phase characteristics of these filters are deterministic, any undesirable effects due to the phase nonlinearities can be compensated.

Houts and Burlage [2] have described the advantages to be derived by using Chebyshev equiripple FIR filters in MTI systems, and they have published computer programs [3] for the design and evaluation of Chebyshev filters having linear phase-delay. Their design procedure is based on a computer program for designing equiripple FIR linear phase digital filters [4] using the Remez exchange algorithm.

2. BENEFITS OF NONLINEAR PHASE

The removal of the linear-phase constraint can result in improved MTI performance in both the spectral and the time domains. The parameters of interest in MTI filter design and defined in Table 1 and depicted in Figure 1. The standard definitions of equiripple FIR filter parameters (e.g., [5]) are defined in Table 2 and depicted in Figure 2. Note that the standard definitions refer to a filter designed to have unity gain in the passband, whereas for MTI applications, it may be desirable to have non-unity passband gain in order to have 0 dB white noise power gain and/or 0 dB minimum passband gain.

The improved spectral performance obtainable from nonlinear-phase equiripple FIR filters relative to linear-phase equiripple FIR filters can be realized in three different ways. First, smaller ripples in either the stopband or passband ripples (or both) can be achieved for given values of f_{STOP} , f_{PASS} and N . Smaller passband ripples mean increased $A_{S\beta}$ and thus greater clutter suppression. Decreased $R_{p\beta}$ means that a higher detection threshold can be used without a loss of target visibility due to signal attenuation in the filter passband.

Second, the transition band, or the band of frequencies between f_{STOP} and f_{PASS} , can be made narrower. Such a narrowing of the transition band is useful when enhanced low-velocity target visibility is desired in the presence of stationary clutter of finite bandwidth.

Third, the performance of a given linear-phase filter can be approximated, except for phase, by a nonlinear-phase filter with smaller N . Thus enhanced incoherent integration gain can be achieved from a given fixed number of radar pulses, since a greater number of independent filtered output pulses are then available for integration [3].

TABLE 1
Definitions of Equiripple MTI Filter Parameters

N	number of weights in filter impulse response
$h(n)$	$(0 \leq n \leq N-1)$ filter impulse response
f	Doppler frequency (Hz)
$ H(f) $	absolute magnitude of the MTI filter response
$H(f)$	complex magnitude of the MTI filter response
f_{STOP}	frequency of upper edge of filter stopband (Hz)
f_{PASS}	frequency of lower edge of filter passband (Hz)
f_{PR}	radar pulse-repetition frequency (Hz)
R_{PB}	peak-to-peak gain ripple in filter passband (dB)
A_{SB}	minimum filter attenuation – measured from bottoms of passband ripples to peaks of stopband ripples (dB)
G_{WNP}	white-noise-power gain (dB)
G_{PBM}	minimum gain in filter passband (dB)

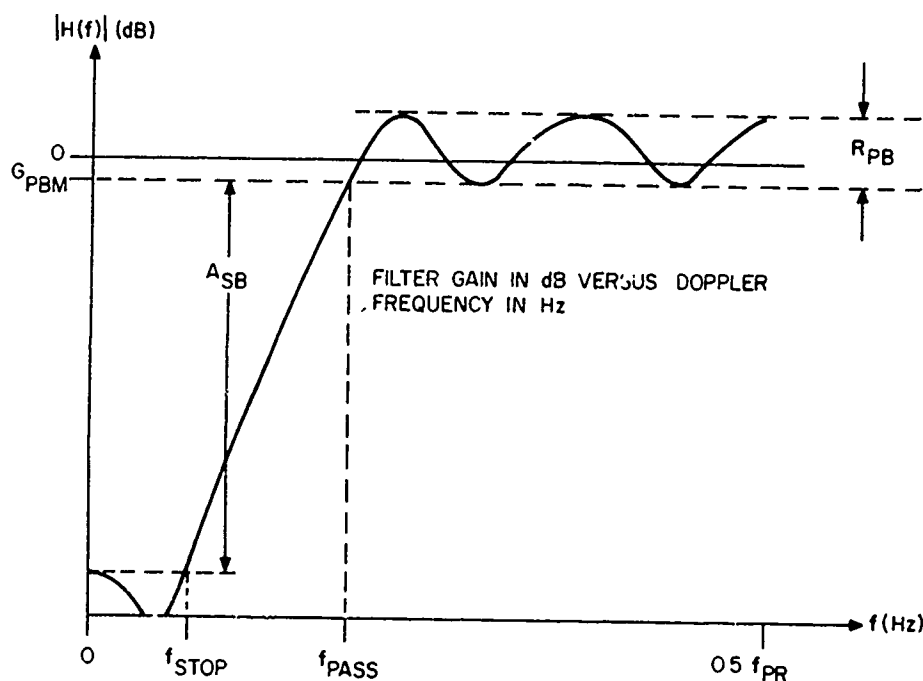


Figure 1. Definition of Equiripple MTI Filter Parameters

TABLE 2

ϵ_1 amplitude of passband ripple (linear)
 ϵ_2 amplitude of stopband ripple (linear)
 (other definitions as in Table 1)

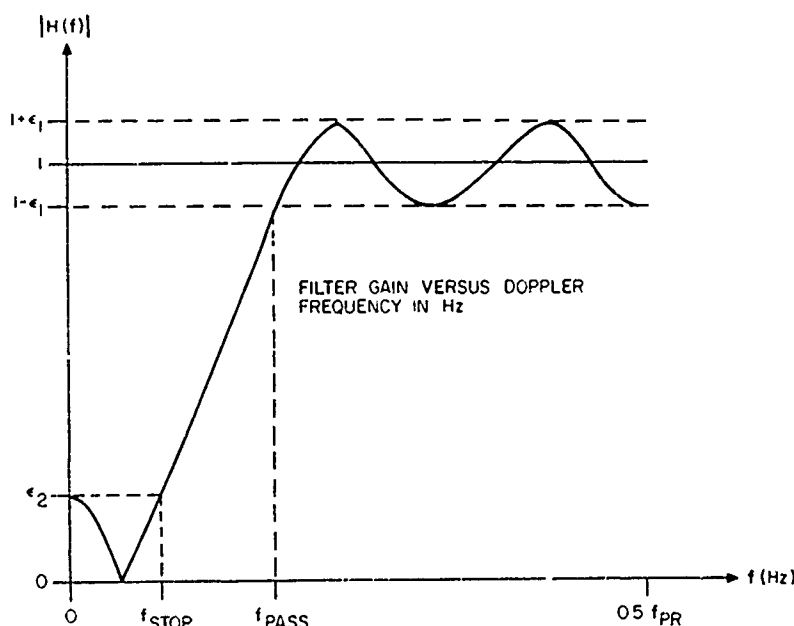


Figure 2. Standard Definition of Equiripple High-Pass Filter Parameters

In Tables 3 and 4 there can be found examples showing each of the three forms of benefit described above. The filters summarized in Table 3 were designed to have $A_{SB} = 50$ dB and values of f_{STOP} ranging from 50 Hz to 400 Hz. Those summarized in Table 4 were designed to have $A_{SB} = 30$ dB and values of f_{STOP} ranging from 100 Hz to 400 Hz. In both tables $f_{PR} = 2500$ Hz. For some sets of design parameters no results are shown, since such filters could not be designed subject to the constraints $G_{WNP} = 0$ dB and $G_{PBM} = 0$ dB (*vide* Tables 3a and 3e).

The relaxation of the linear-phase constraint also allows some latitude in the choice of the filter impulse-response function (IRF), because there are several possible IRFs corresponding to a particular spectral amplitude function. Each of these IRFs corresponds to a different phase-delay characteristic, but since phase delay is of no concern, it now becomes possible to select a particular IRF on the basis of its time domain characteristics. Three possible criteria for this selection are: (I) a mini-max criterion, which attempts to equalize the magnitudes of the IRF weights by selecting that IRF having the smallest value of the ratio of the largest to smallest weights; (II) a criterion which minimizes susceptibility to numerical overflow by selecting that IRF having the minimum absolute sum of its weights; or (III) another criterion which attempts to equalize the magnitude of the IRF weight by selecting that IRF having the minimum variance of the magnitudes of its weights. Criteria I and III should therefore select IRFs which

are less susceptible to disruption by impulsive noise, whereas Criterion II is based on minimizing susceptibility to numerical overflow. The evaluation of the suitability of these criteria or the proposal of others remains as a topic requiring further investigation.

TABLE 3a

Comparative Examples of Equiripple Finite-Impulse-Response MTI Filter Characteristics

Number of Weights N	LINEAR PHASE		NONLINEAR PHASE		Reduction of Passband Lower Edge (Hz)	Reduction in Passband Ripple (dB)
	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}		
5	477	2.71	—	—	—	—
6	537	3.30	390	2.12	147	1.18
7	392	2.48	376	2.07	16	0.41
8	421	2.52	347	1.93	74	0.59
9	386	2.16	319	1.79	67	0.37
10	350	2.05	292	1.64	58	0.41
11	352	2.00	270	1.51	82	0.49
12	301	1.73	251	1.40	50	0.33
13	317	1.82	234	1.30	83	0.52
14	265	1.51	219	1.22	46	0.29
15	286	1.65	206	1.14	80	0.51
16	237	1.34	195	1.08	42	0.26

$$f_{STOP} = 25 \text{ Hz}$$

$$f_{PR} = 2500 \text{ Hz}$$

$$A_{SB} = 50 \text{ dB}$$

$$G_{WNP} = 0 \text{ dB}$$

$$G_{PBM} = 0 \text{ dB}$$

TABLE 3b

Comparative Examples of Equiripple Finite-Impulse-Response MTI Filter Characteristics

Number of Weights N	LINEAR PHASE		NONLINEAR PHASE		Reduction of Passband Lower Edge (Hz)	Reduction in Passband Ripple (dB)
	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}		
5	718	4.30	576	3.48	142	0.82
6	539	3.33	482	2.87	57	0.46
7	556	3.44	416	2.45	140	0.99
8	423	2.54	367	2.14	56	0.40
9	455	2.80	332	1.90	123	0.90
10	352	2.07	300	1.72	52	0.35
11	386	2.35	282	1.60	104	0.75
12	303	1.76	280	1.60	23	0.16
13	337	2.03	274	1.57	63	0.46
14	288	1.66	263	1.52	25	0.14
15	300	1.79	252	1.46	48	0.33
16	284	1.65	240	1.39	44	0.26

$$f_{STOP} = 50 \text{ Hz}$$

$$f_{PR} = 2500 \text{ Hz}$$

$$A_{SB} = 50 \text{ dB}$$

$$G_{WNP} = 0 \text{ dB}$$

$$G_{PBM} = 0 \text{ dB}$$

TABLE 3c

Comparative Examples of Equiripple Finite-Impulse-Response MTI Filter Characteristics

Number of Weights N	LINEAR PHASE		NONLINEAR PHASE		Reduction of Passband Lower Edge (Hz)	Reduction in Passband Ripple (dB)
	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}		
5	793	5.38	591	3.67	202	1.71
6	542	3.37	532	3.37	10	0.00
7	583	3.78	511	3.13	72	0.65
8	537	3.33	462	2.84	75	0.49
9	469	2.96	417	2.56	62	0.40
10	477	3.01	380	2.31	97	0.70
11	397	2.47	349	2.11	48	0.36
12	419	2.64	332	2.01	87	0.53
13	346	2.13	330	2.00	16	0.13
14	372	2.33	321	1.95	51	0.28
15	340	2.09	308	1.88	32	0.21
16	335	2.09	293	1.79	42	0.30

 $f_{STOP} = 100 \text{ Hz}$ $f_{PR} = 2500 \text{ Hz}$ $A_{SB} = 50 \text{ dB}$ $G_{WNP} = 0 \text{ dB}$ $G_{PBM} = 0 \text{ dB}$

TABLE 3d

Comparative Examples of Equiripple Finite-Impulse-Response MTI Filter Characteristics

Number of Weights N	LINEAR PHASE		NONLINEAR PHASE		Reduction of Passband Lower Edge (Hz)	Reduction in Passband Ripple (dB)
	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}		
5	810	5.65	807	5.61	3	0.04
6	818	5.76	648	4.28	170	1.40
7	654	3.55	634	4.16	20	-0.61
8	627	4.26	580	3.81	47	0.45
9	617	4.14	522	3.40	95	0.74
10	519	3.45	479	3.30	40	0.15
11	536	3.60	479	3.10	57	0.50
12	493	3.25	458	2.97	35	0.28
13	471	3.14	431	2.80	40	0.34
14	472	3.14	407	2.64	65	0.50
15	423	2.80	405	2.62	18	0.18
16	436	2.90	397	2.58	39	0.32

 $f_{STOP} = 200 \text{ Hz}$ $f_{PR} = 2500 \text{ Hz}$ $A_{SB} = 50 \text{ dB}$ $G_{WNP} = 0 \text{ dB}$ $G_{PBM} = 0 \text{ dB}$

TABLE 3e
Comparative Examples of Equiripple Finite-Impulse-Response MTI Filter Characteristics

Number of Weights N	LINEAR PHASE		NONLINEAR PHASE		Reduction of Passband Lower Edge (Hz)	Reduction in Passband Ripple (dB)
	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}		
5	—	—	—	—	—	—
6	—	—	—	—	—	—
7	—	—	871	6.66	—	—
8	873	6.65	789	4.89	84	1.76
9	815	5.58	754	5.50	61	0.08
10	716	5.28	684	5.58	32	-0.30
11	723	5.35	686	4.92	37	0.43
12	703	5.16	646	4.61	57	0.55
13	636	5.39	636	4.52	0	0.87
14	647	4.74	618	4.39	29	0.35
15	640	4.68	596	4.07	44	0.61
16	607	3.97	593	4.20	14	-0.23

 $f_{STOP} = 400$ Hz $f_{PR} = 2500$ Hz $A_{SB} = 50$ dB $G_{WNP} = 0$ dB $G_{PBM} = 0$ dB

TABLE 4a
Comparative Examples of Equiripple Finite-Impulse-Response MTI Filter Characteristics

Number of Weights N	LINEAR PHASE		NONLINEAR PHASE		Reduction of Passband Lower Edge (Hz)	Reduction in Passband Ripple (dB)
	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}		
5	492	2.65	489	2.78	3	-0.13
6	534	3.27	455	2.61	79	0.66
7	485	2.79	407	2.35	78	0.44
8	422	2.53	365	2.12	57	0.41
9	428	2.53	330	1.91	38	0.62
10	353	2.09	302	1.75	51	0.34
11	374	2.23	279	1.62	95	0.61
12	306	1.80	270	1.56	36	0.24
13	332	1.98	268	1.55	64	0.43
14	278	1.49	262	1.53	16	-0.04
15	298	1.78	253	1.48	45	0.30
16	275	1.61	243	1.43	32	0.18

 $f_{STOP} = 100$ Hz $f_{PR} = 2500$ Hz $A_{SB} = 30$ dB $G_{WNP} = 0$ dB $G_{PBM} = 0$ dB

TABLE 4b
Comparative Examples of Equiripple Finite-Impulse-Response MTI Filter Characteristics

Number of Weights N	LINEAR PHASE		NONLINEAR PHASE		Reduction of Passband Lower Edge (Hz)	Reduction in Passband Ripple (dB)
	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}		
5	766	4.97	594	3.71	172	1.26
6	549	3.46	503	3.64	46	-0.18
7	479	3.73	505	3.12	74	0.61
8	521	3.28	469	2.91	53	0.37
9	474	3.03	428	2.67	46	0.36
10	479	3.05	394	2.47	85	0.58
11	409	2.61	387	2.42	22	0.19
12	428	2.74	380	2.39	48	0.35
13	395	2.52	366	2.31	29	0.21
14	386	2.48	349	2.21	37	0.27
15	384	2.46	334	2.12	50	0.34
16	353	2.28	331	2.11	22	0.17

 $f_{STOP} = 200 \text{ Hz}$ $f_{PR} = 2500 \text{ Hz}$ $A_{SB} = 30 \text{ dB}$ $G_{WNP} = 0 \text{ dB}$ $G_{PBM} = 0 \text{ dB}$

TABLE 4c
Comparative Examples of Equiripple Finite-Impulse-Response MTI Filter Characteristics

Number of Weights N	LINEAR PHASE		NONLINEAR PHASE		Reduction of Passband Lower Edge (Hz)	Reduction in Passband Ripple (dB)
	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}	Passband Lower Edge (Hz) f_{PASS}	Passband Ripple (dB) R_{PB}		
5	822	5.85	822	5.84	0	0.01
6	829	5.95	759	4.66	70	1.29
7	773	5.06	697	4.85	76	0.21
8	600	4.66	630	4.97	-30	-0.31
9	669	4.75	630	4.35	39	0.40
10	645	4.55	593	4.10	52	0.45
11	588	4.17	579	4.00	9	0.17
12	597	4.24	567	3.92	30	0.32
13	584	4.15	543	3.77	41	0.38
14	548	3.91	542	3.75	6	0.16
15	555	3.96	530	3.68	25	0.28
16	548	3.90	518	3.55	30	0.35

 $f_{STOP} = 400 \text{ Hz}$ $f_{PR} = 2500 \text{ Hz}$ $A_{SB} = 30 \text{ dB}$ $G_{WNP} = 0 \text{ dB}$ $G_{PBM} = 0 \text{ dB}$

3. OUTLINE OF THE DESIGN PROCEDURE

The technique for designing nonlinear-phase FIR filters is based on a procedure first suggested in [6]. This procedure involves the design of a prototype linear-phase FIR filter having an IRF $h_p(n)$ of length $(2N-1)$ and an amplitude response in the frequency domain $H_p(f)$ equal to the square of the magnitude of the desired frequency response. It is shown below that this prototype filter cannot be designed directly from the desired parameters of the nonlinear-phase filter, but that an iterative technique must be used.

A property of linear-phase FIR filters with real-valued IRFs is that if z_0 is a zero of the IRF, then so are z_0^{-1} , z_0^* and $(z_0^{-1})^*$ where $*$ denotes complex conjugate ([1]; page 159). Note that if z_0 is real, then $z_0 = z_0^*$ and also, if $|z_0| = 1$, then $z_0^{-1} = z_0^*$, so that roots can be real and single, or occur in conjugate pairs (if $|z_0| = 1$) or in reciprocal pairs (if z_0 is real), or in conjugate-reciprocal quartets (if $|z_0| \neq 1$ and z_0 complex). In general, of the $2(N-1)$ zeroes of the $(2N-1)$ -length prototype filter, there are k pairs of reciprocal real zeroes, l quartets of conjugate reciprocal zeroes and $m = (N-k-2l-1)$ pairs of double zeroes on the unit circle. To extract an IRF $h(n)$ of length N , it is necessary to discard one zero of each of the m pairs of double zeroes, one zero of each of the k pairs of reciprocal zeroes, and one pair of conjugate zeroes from each of the l quartets of conjugate-reciprocal zeroes. This leaves a set of $(N-1)$ zeroes which is expanded to produce a real-valued IRF of length N .

A set of M IRFs of length N can thus be derived from the $(2N-1)$ -length prototype filter, where

$$M = 2^{(k+l-1)}$$

Note that one zero or pair of conjugate zeroes can be arbitrarily chosen to lie inside or outside the unit circle, since the only effect of this choice is to reverse the IRFs in the time domain; i.e., $h(n)$ is replaced by $h(N-1-n)$ for $0 \leq n \leq N-1$.

An upper bound on M as a function of N is given by

$$M_{UB} = 2^{\{[(N-2)/2]-1\}}$$

where $[x]$ denotes the largest integer less than x , so it can be seen that the set of IRFs to be examined can contain of order 2^{10} for $N=24$. Hence the optimization techniques described above can become quite time-consuming for filters of such length. This limitation, however, should not preclude the use of these procedures for designing filters of the lengths usually considered for MTI applications (e.g., $N \leq 20$; so that $M \leq 256$).

4. DESIGNING THE PROTOTYPE FILTER

In order to make use of the standard filter design algorithms, it is first necessary to define the filter ripple parameters (R_{PB}, A_{SB}) of interest to radar MTI designers in terms of the standard linear ripple parameters

(ϵ_1, ϵ_2) . It can easily be shown from Figures 1 and 2 that

$$\epsilon_1 = \frac{10^{(R_{PB}/20)}}{10^{(R_{PB}/20)} + 1} \cdot 1 \quad (1)$$

and

$$\epsilon_2 = \frac{1 - \epsilon_1}{10^{(A_{SB}/20)}} \cdot \quad (2)$$

In the standard design procedure for linear-phase FIR filters [3], if N , f_{STOP} , f_{PASS} and f_{PR} are specified, then only the ratio

$$W = \epsilon_1 / \epsilon_2 \quad (3)$$

can be specified as a free parameter. This restriction can be circumvented by allowing f_{PASS} to be varied in an iterative manner until that value for f_{PASS} is found which gives the desired values for ϵ_2 and hence ϵ_1 [3].

A similar iterative procedure is necessary in the design of nonlinear-phase FIR filters. For the design of the linear-phase prototype filter $H_O(f)$ (see Figures 3(a)-3(c)), it is necessary to specify $2N-1$, f_{STOP} , f_{PR} , δ_1 and δ_2 , where δ_1 and δ_2 have to be specified in terms of ϵ_1 and ϵ_2 . The outline of the scheme for relating the δ 's and ϵ 's is pictured in Figures 3(a)-3(c) and is similar to that of [6] except for one detail pointed out below. For clarity the scheme is described progressing from the original prototype filter $H_O(f)$ (Figure 3(a)) to the intermediate filter $H_I(f)$ (Figure 3(b)) to the final prototype filter $H_P(f) = |H(f)|$ (Figure 3(c)). In practice, the actual progression is from specifying $H_P(f)$ to specifying $H_O(f)$ in terms of $H_P(f)$, since $H_O(f)$ is the filter which is actually designed using the algorithm of [4].

$H_I(f)$ is related to $H_O(f)$ by the transformation

$$H_I(f) = H_O(f) + \delta_2. \quad (4)$$

In the time domain this is equivalent to

$$h_I(n) = \begin{cases} h_O(n), & 1 \leq |n| \leq N-1 \\ h_O(n) + \delta_2, & n = 0. \end{cases} \quad (5)$$

$H_P(f)$ is in turn related to $H_I(f)$ by the transformation

$$H_P(f) = K H_I(f) \quad (6)$$

which becomes in the time domain

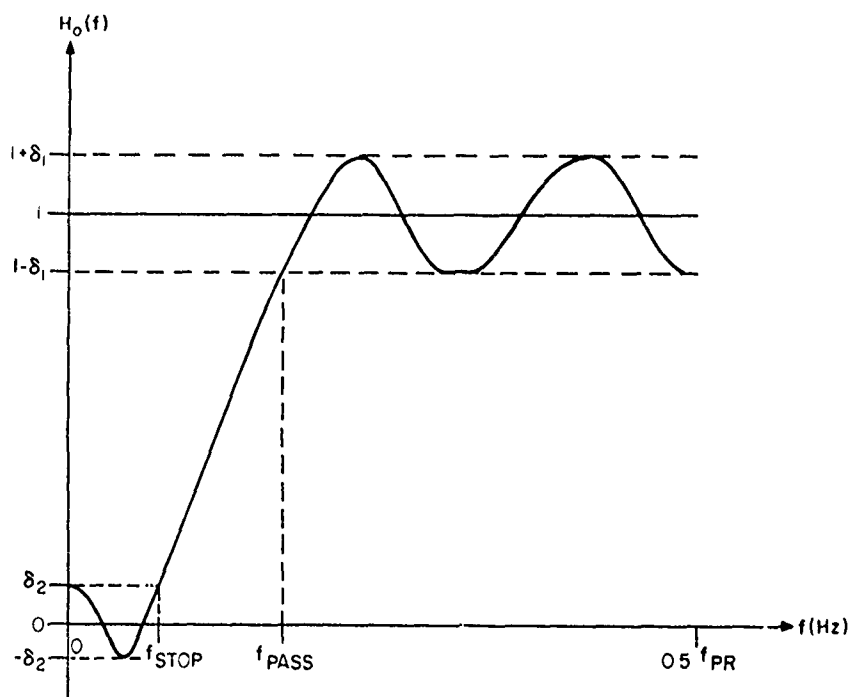


Figure 3(a). Original Linear-Phase Prototype Filter Frequency Response

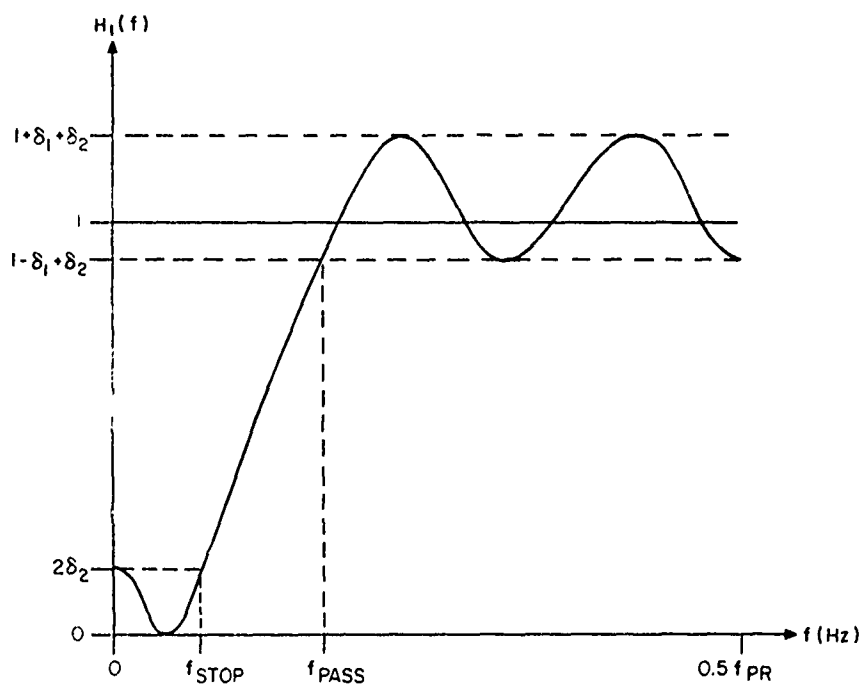


Figure 3(b) Intermediate Linear-Phase Prototype Filter Frequency Response

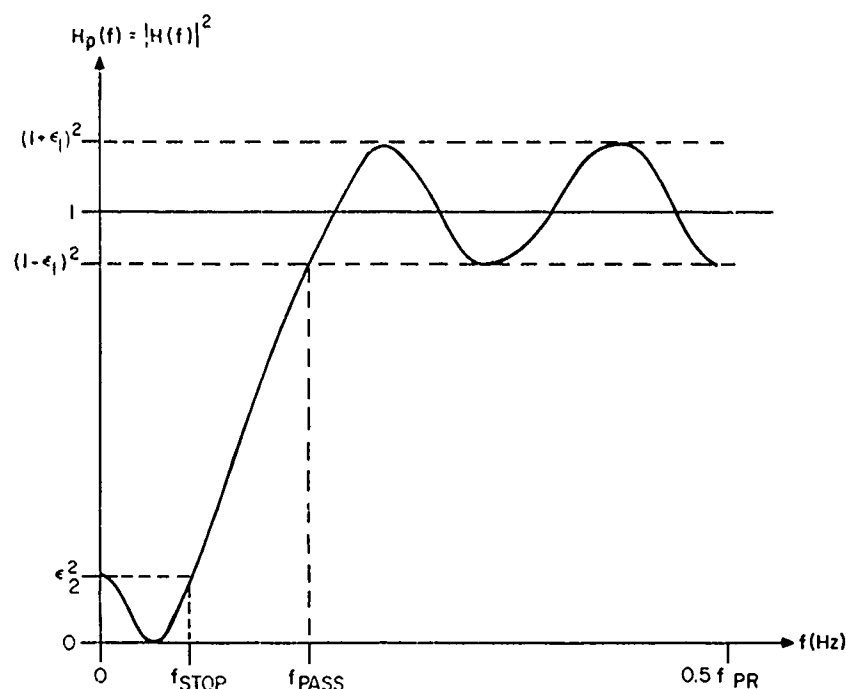


Figure 3(b). Linear-Phase Prototype Filter Frequency Response

$$h_p(n) = K h_I(n), \quad -N+1 \leq n \leq N-1 \quad (7)$$

where K is a scaling factor. It is in the determination of the factor K that this development differs from that of [6], for it can be noted that a scaling by the factor $(1+\delta_2)^{-1}$ does not produce a filter $|H(f)|$ with equal (linear) magnitude ripple excursions above and below unity gain.

Since it is the parameters R_{PB} and A_{SB} and thus ϵ_1 and ϵ_2 which are specified by the filter designer, it is necessary to invert the above functional relationship to derive expressions for δ_1 and δ_2 in terms of ϵ_1 and ϵ_2 . This being done, it is a straightforward matter to use the linear-phase FIR filter algorithm to determine the $(2N-1)$ -length IRF $h_o(n)$.

It can be seen from eqn. (6) and Figures 3(b) and 3(c) that

$$(1 + \epsilon_1)^2 = K(1 + \delta_1 + \delta_2) \quad (8)$$

$$(1 - \epsilon_1)^2 = K(1 - \delta_1 + \delta_2) \quad (9)$$

and

$$\epsilon_2^2 = 2K \delta_2 \quad (10)$$

Some algebraic manipulation shows that

$$\delta_1 = \frac{4\epsilon_1}{2 + 2\epsilon_1^2 - \epsilon_2^2} \quad (11)$$

$$\delta_2 = \frac{\epsilon_2^2}{2 + 2\epsilon_1^2 - \epsilon_2^2} \quad (12)$$

and

$$K = 1 + \epsilon_1^2 - \frac{1}{2} \epsilon_2^2. \quad (13)$$

It is now a simple matter to derive $h_p(n)$ from $h_o(n)$ in terms of these factors as outlined above, and then to derive a nonlinear-phase N-length IRF $h(n)$.

5. GAIN NORMALIZATION

It is often useful to design MTI filters which have 0 dB white-noise gain. This means that the white-noise power in the filtered signal is the same as that in the unfiltered input signal, so that in the absence of clutter, the false-alarm probability P_{FA} is not altered. Such normalization is easily accomplished [3] by invoking Parseval's theorem

$$g^2 = \sum_{n=0}^{N-1} |h(n)|^2 = 2 \int_0^{0.5 f_{PR}} |H(f)|^2 df \quad (14)$$

to compute the white noise power gain g^2 by means of a simple summation. Scaling $h(n)$ by g^{-1} produces a filter with unity white noise power gain ($G_{WNP} = 0$ dB) and leaves A_{SB} and R_{PB} unaltered. Note that this procedure is valid for any FIR filter.

It can also be useful to design a filter which has both $G_{WNP} = 0$ dB and a minimum gain of unity in its passband ($G_{PBM} = 0$ dB). This means that the probability of detection P_D in the passband will not be reduced by filter attenuation in these passband ripples. This goal can be achieved by an iterative procedure. In this case it is the passband ripple that is altered until the desired characteristics are obtained. The procedure is to specify N , A_{SB} and f_{SB} , select an arbitrary reasonable value for R_{PB} and design a filter using the iterative procedure described above. Based on whether this filter has G_{PBM} greater or less than unity gain, the value for R_{PB} is increased or decreased respectively and another iteration is carried out. This procedure is repeated until convergence to $G_{PBM} = 0$ dB is achieved to within an acceptable tolerance.

6. COMPUTER PROGRAMS

The program MPMTIPSF generates nonlinear-phase FIR MTI filters for specified values of N , f_{STOP} , f_{PR} , A_{SB} and R_{PB} . Parameters to specify the

grid density for the design of the prototype filter [4] and to select the impulse-response design criterion must also be provided. This program is listed in Appendix A, and the detailed operating instructions are given there. Typical times required to design a filter using a Xerox Sigma 9 computer range from 2.2 sec for $N=5$ to 28.4 sec for $N=16$.

The program MPMTIPSFO generates nonlinear-phase FIR MTI filters with $G_{PBM} = 0$ dB. The same parameters as for the program MPMTIPSF must be specified by the user, but here the parameter R_{PB} is used only as an initial value which is then modified by the program to converge to $G_{PBM} = 0$ dB. Hence the choice of a starting value for R_{PB} near its final value can significantly reduce the time required to design a filter. This program is listed in Appendix B. Typical running times on the Sigma 9 can be 1 to 10 or more times as long as for the program MPMTIPSF, depending on how good an estimate for the initial value of R_{PB} is used.

Appendix C contains the listings of the subroutines required by the programs MPMTIPSF and MPMTIPSFO. The subroutine REMEZ and its ancillary subroutines have not been included, since they are readily accessible elsewhere [4], but note the modifications required in the dimensions of the COMMON block variables.

7. SUMMARY

It has been shown in Tables 3 and 4 that, for the typically narrow stopbands used in radar MTI filters, nonlinear-phase filters can be designed which offer superior visibility for low-velocity targets, relative to that offered by linear-phase filters designed to the same stopband specifications. It has also been pointed out that the time-domain response of such nonlinear-phase MTI filters can be optimized without altering the filter power response in the frequency domain. The details of the design procedure have been described, and listings of Fortran programs for implementing the procedure have been provided.

8. ACKNOWLEDGEMENT

This work is supported by the Department of National Defence under Research and Development Branch Project No 33C69.

9. REFERENCES

1. Oppenheim, A.V. and R.W. Schaffer, *Digital Signal Processing*. Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1975.
2. Houts, R.C. and R.W. Burlage, *Maximizing the Usable Bandwidth of MTI Signal Processors*, IEEE Trans. Aerospace Electronic Systems AES-13, No. 1, January 1977, 48-55.

3. Burlage, D.W. and R.C. Houts, *Design Technique for Improved Bandwidth Moving Target Indicator Processors in Surface Radars*. Report No. RE-TR-75-35, US Army Missile Command, Redstone Arsenal, Alabama, June 1975.
4. McClellan, J.H., T.W. Parks and L.R. Rabiner, *A Computer Program for Designing Optimum FIR Linear Phase Digital Filters*. IEEE Trans. Audio Electroacoust. AU-21, No. 6, December 1973, 506-526.
5. Herrmann, O., *Design of Nonrecursive Digital Filters with Linear Phase*. Electronics Letters 6, No. 11, 28 May 1970, 328-329.
6. Herrmann, O. and W. Schuessler, *Design of Nonrecursive Digital Filters with Minimum Phase*. Electronics Letters 6, No. 11, 28 May 1970.

APPENDIX A

PROGRAM MPMTIPSF

```

1 - C
2 - C
3 - C
4 - C
5 - C
6 - C
7 - C
8 - C
9 - C
10 - C
11 - C
12 - C
13 - C
14 - C
15 - C
16 - C
17 - C
18 - C
19 - C
20 - C
21 - C
22 - C
23 - C
24 - C
25 - C
26 - C
27 - C
28 - C
29 - C
30 - C
31 - C
32 - C
33 - C
34 - C
35 - C
36 - C
37 - C
38 - C
39 - C
40 - C

PROGRAM MFMTIPSF
WRITTEN BY R.W. HERRING, COMMUNICATIONS RESEARCH CENTRE, OTTAWA
MODIFIED FROM LINEAR PHASE MTI FILTER DESIGN PROGRAM
WRITTEN BY D.W. BURLAGE AND R.C. HOUTS, MICOM.

*****
THIS PROGRAM IS USED TO DESIGN MIXED PHASE FIR DIGITAL FILTERS **
FOR USE IN MTI RADARS AND IS A MODIFIED VERSION OF THE MCCLELLAN**
PROGRAM WHICH WAS PUBLISHED IN THE DECEMBER, 1973 ISSUE OF THE **
IEEE TRANSACTIONS ON AUDIO AND ELECTROACOUSTICS. **
MODIFICATIONS INCLUDE: **
1. LIMITED TO HIGHPASS FILTER DESIGN. **
2. NORMALIZES COEFFICIENTS W.R.T. SUM H(1)**2 (PGN = 0 DR). **
3. COEFFICIENTS EQUALIZED FOR OPTIMUM NOISE-REJECTION **
PERFORMANCE. **
IN F U T D A T A C O N S I S T S O F O N E C A R D **
I N F R E E F O R M A T (VARIABLES SEPARATED BY COMMAS) : **
1. MFILT - NUMBER OF FILTER WEIGHTS (.LE. 75). **
2. LGRID - GRID DENSITY -- LGRID*MFILT .LE. 1200 . **
IF LGRID = 0, PROGRAM DEFAULTS TO MAXIMUM VALUE. **
IF LGRID = 1, PROGRAM USES MINIMUM VALUE FOR LGRID **
WHICH LEADS TO CONVERGENCE. **
PROGRAM CHECKS VALUE FOR LGRID AND ENSURES THAT **
AT LEAST TWO GRID POINTS LIE IN STOP BAND. **
3. STOPF - UPPER EDGE OF STOPBAND (HZ). **
4. PRF - PULSE REPETITION FREQUENCY (HZ). **
5. ASB - STOPBAND ATTENUATION (DB). **
6. RPB - PASSBAND RIPPLE (DB). **
7. MODE - SELECTS RULE FOR CHOOSING PARTICULAR FILTER **
IMPULSE RESPONSE: **
IF MODE = 1, MINI-MAX RATIO RULE IS USED. **
IF MODE = 2, MINIMUM ABSOLUTE SUM RULE IS USED. **
IF MODE = 3, MINIMUM VARIANCE RULE IS USED. **
*****
COMMON /HPFC/ PI,FCU,FUP,WTX,RATIO,ESDEL1,ESDEL2,
* NFILT,NEG,NODD,LGRID
COMMON PI2,AD,DEV,X,Y,GRID,DES,WT,ALPHA,IEXT,NFCNS,NGRID
DIMENSION H(150),ROOTK(150),ROOTI(150)

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

41 - DIMENSION IEXT(79),AD(79),ALPHA(79),X(79),Y(79)
42 - DIMENSION EXTF(79)
43 - DIMENSION Z(150)
44 - DIMENSION DES(1200),GRID(1200),WT(1200)
45 - DOUBLE PRECISION PI2,PI
46 - DOUBLE PRECISION AD,DEV,X,Y
47 - DOUBLE PRECISION H,ROOTR,ROOTI,DTEMP,Z,ZNORM,
48 - * DABSUM,SCALE
49 - LOGICAL SPTFLG
50 - DATA RATIO/1./,NEG/O/,NODD/1/
51 - PI = DATAN2(O.DO,-1.DO)
52 - PI2 = 2.DO*PI
53 - 10 J = NTIMER(O)
54 - SPTFLO = .FALSE.
55 - C
56 - C
57 - C
58 - PROGRAM INPUT SECTION.
59 - INPUT MFILT, LGRID, STOPF, PRF, ASB, RPB, MODE
60 - IF (MFILT .LE. 1) STOP
61 - IF (MFILT .LE. 75) GO TO 20
62 - WRITE (108,15)
63 - 15 FORMAT(/, MFILT SET TO MAX ALLOWED VALUE -- MFILT = 75/)
64 - MFILT = 75
65 - C
66 - C
67 - C
68 - FIND NORMALIZED STOP FREQUENCY, LINEAR RIPPLES AND WEIGHT.
69 - FCU = STOPF/PRF
70 - CRP = 10.*(RPB/20.)
71 - E1 = (CRP-1.)/(CRP+1.)
72 - E2 = 10.*(ASB/20.)*(1.-E1)
73 - WEIGHT = E1/E2
74 - DENOM = 2.+(2.*E1*E1-E2*E2)
75 - D1 = 4.*E1/DENOM
76 - D2 = E2*E2/DENOM
77 - WTX = (4.*E1)/(E2*E2)
78 - NFILT = 2*MFILT-1
79 - NFCNS = MFILT
80 - LGRIDM = 1200/NFCNS
PROGRAM DESIGN MAX 75 TAPS / LGRID = 1200/MFILT

```

```

81 - C
82 -
83 -
84 - C
85 -
86 -
87 -
88 -
89 -
90 -
91 - C
92 - C
93 - C
94 -
95 - C
96 - C
97 - C
98 -
99 -
100 -
101 -
102 -
103 -
104 - C
105 - C
106 - C
107 -
108 -
109 -
110 -
111 -
112 -
113 -
114 -
115 -
116 -
117 -
118 -
119 -
120 -

IF (LGRID .LE. 0) LGRID = 1 GRJDM
ITEST = LGRID*NFCNS
ENSURE AT LEAST 2 GRID POINTS IN STOPBAND.
IF (2./FLOAT(ITEST) .LE. FCU) GO TO 30
LGRID = 2./(FCU*FLOAT(NFCNS))+1.
ITEST = LGRID*NFCNS
25
30 IF (ITEST .LE. 1200) GO TO 31
LGRID = LGRIDM
31 IF (SPTFLG) GO TO 50

ESTIMATE PASSBAND FREQUENCY

IF (FCU .GE. 0.04) GO TO 35

CHERYSHV LOWER BOUND ESTIMATE FOR FCU .LT. 0.04

XT = (1.+D1)/D2
YT = (1.-D1)/D2
COSHIX = ALOG(XT+SQRT((XT+1.)*(XT-1.)))
COSHIX = ALOG(YT+SQRT((YT+1.)*(YT-1.)))
DELF=(COSHIX-SQRT((COSHIX+COSHIX)*(COSHIX-COSHIX)))/(PI*(NFILT-1))
GO TO 45

HERRMANN ESTIMATE (BSTJ) FOR FCU .GE. 0.04

35 D1L = ALOG10(D1)
D2L = ALOG10(D2)
FK = 11.01217 + 0.51244 * ALOG10(WTX)
DINF = ((0.005309*D1L+0.07114)*D1L-0.4761)*D2L
* ((0.00266*D1L+0.5941)*D1L-0.4278
DELF = (NFILT-1)*(SQRT(1.+4.*FK*DINF/(NFILT-1)**2) 1.)/(2.*FK)
45 FUP = FCU+DELF
IF (FUP .GE. 0.4) FUP = 0.4
FST = 0.4*DELF
ITER = 0
ELAST = 0.
FLAST = 0.
DEL = 0.001*D2
IF (FUP .LE. 0.45) GO TO 50

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDG

```

121 - WRITE(108,40) FUP
122 - FORMAT(1H0,'++UNSUCCESSFUL DESIGN BECAUSE PASSF/PRF = ',F6.4,
123 - * ', WHICH IS GREATER THAN 0.4 ++')
124 - GO TO 500
125 - CALL HPFH
126 - ITER = ITER+1
127 - C
128 - C CHECK FOR CONVERGENCE
129 - C
130 - C ERROR = D2-ESDEL2
131 - IF (ABS(ERROR) .LE. DEL) GO TO 70
132 - FTEMP = FUP
133 - FUP = FLAST-((FLAST-FUP)*ELAST/(ELAST-ERROR))
134 - FLAST = FTEMP
135 - ELAST = ERROR
136 - IF (FUP) 60,60,50
137 - FUP = FLAST-SIGN(FST,ERROR)
138 - GO TO 50
139 - PASSF = FUP*PRF
140 - C
141 - C CALCULATE IMPULSE RESPONSE OF
142 - C SQUARED FILTER WITH
143 - C LINEAR PHASE.
144 - C
145 - NM1 = NFCNS-1
146 - NZ = NFCNS+1
147 - DO 305 J = 1,NM1
148 - H(J) = 0.5*ALPHA(NZ-J)
149 - H(NFILT+1-J) = H(J)
150 - H(NFCNS) = ALPHA(1)+DEV/WT(1)
151 - C
152 - C SOLVE FOR MIX'D PHASE
153 - C IMPULSE RESPONSE.
154 - C
155 - CALL RTSQF(H,NFILT,FCU,ROOTR,ROOTI,MON,MOFF,NCOMB,NBIG)
156 - C
157 - C EXPAND ROOTS TO DERIVE MAXIMUM PHASE IMPULSE RESPONSE.
158 - C
159 - C DTEMP = DABS(H(1))
160 - CALL EXPAND(MON+MOFF,ROOTR,ROOTI,MTERMS,Z,H)

```

```

161 - IF (MTERMS.EQ. MFILT) GO TO 340
162 - IF (LGRID.EQ. LGRIDM) GO TO 333
163 - C
164 - C
165 - C
166 - C
167 - C
168 - C
169 - C
170 - C
171 - C
172 - C
173 - C
174 - C
175 - C
176 - C
177 - C
178 - C
179 - C
180 - C
181 - C
182 - C
183 - C
184 - C
185 - C
186 - C
187 - C
188 - C
189 - C
190 - C
191 - C
192 - C
193 - C
194 - C
195 - C
196 - C
197 - C
198 - C
199 - C
200 - C

IF (MTERMS.EQ. MFILT) GO TO 340
IF (LGRID.EQ. LGRIDM) GO TO 333

IF WE GET HERE, SPLIT DOUBLE ROOTS FOUND ON UNIT CIRCLE.
DOUBLE THE GRID DENSITY AND TRY AGAIN.

LGRID = LGRID*2
SPTFLG = .TRUE.
GO TO 25

333 WRITE (108,335) MFILT,MTERMS,LGRIDM
335 FORMAT(/, ERROR: MTERMS,NE, MFILT, MFILT =',I4,', MTERMS =',
* I4,', USING MAXIMUM ALLOWED VALUE FOR LGRID:',I5,',')
GO TO 350

FIND THE OPTIMUM MIXED PHASE
FILTER.

340 CALL OPTMPF(MODE,MON,MUFF,ROOTR,ROOTI,MTERMS,Z)

PROGRAM OUTPUT SECTION.

350 WRITE(108,359)
359 FORMAT(1X)
CALL NEWPAGE(108)
WRITE(108,360)
360 FORMAT(1H1, 82(1H*)//5X,'FINITE IMPULSE RESPONSE (FIR)',
* 'OPTIMUM MIXED PHASE DIGITAL FILTER DESIGN', / 10X,
* 'FOR REMOVING GROUND CLUTTER IN MTI RADAR SIGNAL PROCESSOR'//)
WRITE (108,361) MFILT,LGRID,ITER
361 FORMAT(23X,I4,' IAP FILTER',
* '(LGRID =',I4,')//23X,'CONVERGENCE AFTER',J3,
* 'ITERATIONS'//)
IF (MODE.EQ. 1) WRITE(108,365)
365 FORMAT(/31X,'MINJ-MAX FILTER'//)
IF (MODE.EQ. 2) WRITE(108,366)
366 FORMAT(/25X,'MINIMUM ABSOLUTE SUM FILTER'//)
IF (MODE.EQ. 3) WRITE(108,367)
367 FORMAT(/23X,'MINIMUM ABSOLUTE VARIANCE FILTER'//)
SCALE = 2.00/(DSQRT(1.00+(FSDEL1+FSDEL2)))
* +DSQRT(1.00-(FSDEL1-ESDEL2)))

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDG

```

201 - ZNORM = SCALE*DSQRT(DTEMP)
202 - E1C = 0.5*SCALE*
203 - * (USQRT(1.0+(ESDEL1+ESDEL2))-DSQRT(1.0-(ESDEL1-ESDEL2)))
204 - E2C = SCALE*DSQRT(2.0*ESDEL2)
205 - WRITE (108,390)
206 - 390 FORMAT(/31H BAND LOWER EDGE UPPER EDGE ,5X,
207 - * 'WEIGHT',5X,'RIPPLE',5X,'RIPPLE(DB)'/2X,66(1H--))
208 - HPRF = PRF/2.00
209 - DBSTOP = 20.* ALOG10(E2C/(1.-E1C))
210 - DBPASS = 20.* ALOG10( (1.+E1C)/(1.-E1C) )
211 - WRITE (108,410) STOPF, WEIGHT, E2C, DBSTOP
212 - 410 FORMAT(2X,'STOPF' 0.00',2(F12.2),F11.5,F14.2/)
213 - WRITE (108,420) PASSF, HPRF, E1C, DBPASS
214 - 420 FORMAT (2X,'PASSF',2(F12.2),8X,'1.00',F11.5,F14.2
215 - * /2X,66(1H--)//)
216 - C
217 - C
218 - C
219 - C
220 - PKGN = 20.*ALOG10(1.+E1C)
221 - SMGN = 20.*ALOG10(1.-E1C)
222 - SUM = 0.00
223 - DO 370 K = 1,MFILT
224 - Z(K) = Z(K)*ZNORM
225 - SUM = SUM+Z(K)*Z(K)
226 - PCN = 10.00*ALOG10(SUM)
227 - SKTSUM = SQRT(SUM)
228 - WRITE (108,373) PCN,PKGN,SMGN,Z(J),J = 1,MFILT)
229 - 373 FORMAT(1H0,' ORIGINAL TAP GAINS: NOISE POWER GAIN = ',F7.3,
230 - * ' DB',//23X,'MAX GAIN IN PASSEBAND =',F7.3,
231 - * ' DB',//23X,'MIN GAIN IN PASSEBAND =',F7.3,' DB',/(//2X,10F10.5))
232 - C
233 - C
234 - C
235 - PKGN = PCN-SKTSUM
236 - SMGN = SMGN-PCN
237 - SUM = 0.
238 - ZMAX = 0.
239 - TMIN = 7.237E75
240 - DARSUM = 0.00

```

NORMALIZE Z(K) W.R.T. PCN AND COMPUTE NEW PCN = 0 DB

```

241 DO 380 K = 1,MFILF
242 Z(K) = Z(K)/SRTSUM
243 DTEMP = DABS(Z(K))
244 DABSUM = DABSUM+DTEMP
245 IF (DTEMP.LE. ZMAX) GO TO 375
246 ZMAX = DTEMP
247 KMAX = K
248 IF (DTEMP.GE. ZMIN) GO TO 380
249 ZMIN = DTEMP
250 KMIN = K
251 SUM = SUM+Z(K)*Z(K)
252 PGN = 10.*ALD610(SUM)
253 WRITE (108,385) PGN,PGN,SMGN,(Z(I),J = 1,MFILF)
254 385 FORMAT(/1H0,' NORMALIZED TAP GAINS: NOISE POWER GAIN - ',F7.3,
255 * ' DB, '//25X,' MAX GAIN IN PASSBAND = ',F7.3,
256 * ' DB, '//25X,' MIN GAIN IN PASSBAND = ',F7.3, ' DB, '//25X,' 10F10.5))
257 1APRAT = ZMAX/ZMIN
258 WRITE (108,387) KMAX,KMIN,TAPRAT,DABSUM,NCOMB,NRIG
259 387 FORMAT(/1H0 TAP',I3,' HAS GREATEST MAGNITUDE, '
260 * '//0 TAP',I3,' HAS SMALLEST MAGNITUDE, '
261 * '//0 RATIO OF GREATEST TO SMALLEST TAP WEIGHTS IS',F8.2
262 * '//0 ABSOLUTE SUM OF TAP MAGNITUDES IS',F7.3
263 * '//0',I2,' PAIRS OF ROOTS COMBINED, '
264 * '//0',I2,' LARGE ROOTS DISCARDED, '//
265 DO 450 J = 1,NZ
266 EXIF(J) = GRIN(EXT(J))*PRF
267 WRITE(108,455) (EXTF(I),J = 1,NZ)
268 455 FORMAT(/ 1X,EXTREMAL FREQUENCIES (HZ) ,
269 * // (10F10.3/))
270 WRITE(108,460)
271 460 FORMAT(1H0,B2(1H*))
272 C
273 C
274 C
275 C
276 C
277 * DBSTOP,DBPASS,PGN,PGN,MODE
278 480 FORMAT(13,11,14,2F8.3,F5.0,F5.3,5F8.3,11)
279 WRITE(106,490) (Z(I),J = 1,MFILF)
280 490 FORMAT(10F8.5)

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```
281 -  
282 -  
283 -  
284 -  
285 -  
286 -  
287 -  
500 CONTINUE  
J = NTIMER(1)  
TEMP = FLOAT(J)/500.  
WRITE (108,510) TEMP  
510 FORMAT(/' COMPUTATION REQUIRED',F7.2,' SECONDS OF CPU TIME.'//)  
GO TO 10  
END
```

APPENDIX B**PROGRAM MPMTIPSFO**

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

1 - C
2 - C
3 - C
4 - C
5 - C
6 - C
7 - C
8 - C*****
9 - C** THIS PROGRAM IS USED TO DESIGN MIXED PHASE FIR DIGITAL FILTERS **
10 - C** FOR USE IN MTI RADARS AND IS A MODIFIED VERSION OF THE MCCLELLAN**
11 - C** PROGRAM WHICH WAS PUBLISHED IN THE DECEMBER, 1973 ISSUE OF THE **
12 - C** IEEE TRANSACTIONS ON AUDIO AND ELECTROACOUSTICS. **
13 - C** MODIFICATIONS INCLUDE: **
14 - C** 1. LIMITED TO HIGHPASS FILTER DESIGN. **
15 - C** 2. NORMALIZES COEFFICIENTS W.R.T. SUM H(I)**2 (PGN = 0 DB). **
16 - C** 3. MINIMUM GAIN OF 0 DB IN PASSBAND. **
17 - C** 4. COEFFICIENTS EQUALIZED FOR OPTIMUM NOISE-REJECTION **
18 - C** PERFORMANCE. **
19 - C** INPUT DATA CONSISTS OF ONE CARD **
20 - C** INPUT FORMAT (VARIABLES SEPARATED BY COMMAS) : **
21 - C** 1. MFILT - NUMBER OF FILTER WEIGHTS (.LE. 75). **
22 - C** 2. LGRID - GRID DENSITY -- LGRID*MFILT .LE. 1200. **
23 - C** IF LGRID = 0, PROGRAM DEFAULTS TO MAXIMUM VALUE. **
24 - C** IF LGRID = 1, PROGRAM USES MINIMUM VALUE FOR LGRID **
25 - C** WHICH LEADS TO CONVERGENCE. **
26 - C** PROGRAM CHECKS VALUE FOR LGRID AND ENSURES THAT **
27 - C** AT LEAST TWO GRID POINTS LIE IN STOP BAND. **
28 - C** 3. STOPF - UPPER EDGE OF STOPBAND (HZ). **
29 - C** 4. PRF - PULSE REPETITION FREQUENCY (HZ). **
30 - C** 5. ASB - STOPBAND ATTENUATION (DB). **
31 - C** 6. RPB - PASSBAND RIPPLE (DB). **
32 - C** 7. MODE - SELECTS RULE FOR CHOOSING PARTICULAR FILTER **
33 - C** IMPULSE RESPONSE: **
34 - C** IF MODE = 1, MINI-MAX RATIO RULE IS USED. **
35 - C** IF MODE = 2, MINIMUM ABSOLUTE SUM RULE IS USED. **
36 - C** IF MODE = 3, MINIMUM VARIANCE RULE IS USED. **
37 - C*****
38 - C** COMMON /HPFC/ PI,FCU,FUP,WTX,RATIO,ESDEL1,ESDEL2,
39 - C** NFILT,NEG,NOOD,LGRID
40 - C** COMMON PI2,AD,DEV,X,Y,GRID,DES,WT,ALPHA,TEXT,NFCNS,NGRID

```

```

41 - DIMENSION H(150),ROOTR(150),ROOTI(150)
42 - DIMENSION IEXT(79),AD(79),ALPHA(79),X(79),Y(79)
43 - DIMENSION EXTF(79)
44 - DIMENSION Z(150)
45 - DIMENSION DES(1200),GRID(1200),WT(1200)
46 - DOUBLE PRECISION FI2,FI
47 - DOUBLE PRECISION AD,DEV,X,Y
48 - DOUBLE PRECISION H,ROOTR,ROOTI,UTEMP,Z,ZNOR,I
49 - * DABSUM,SCALE
50 - LOGICAL SPTFLG
51 - DATA RATIO/1./,NEG/O/,NODD/1/
52 - FI = DATAN2(O.DO,-1.DO)
53 - FI2 = 2.DO*PI
54 - 10 J = NTIMER(O)
55 - SPTFLG = .FALSE.
56 - ITER = 0
57 - C
58 - C
59 - C
60 - INPUT MFILT, LGRID, STOPF, PRF, ASB, RFB, MODE
61 - IF (MFILT .LE. 1) STOP
62 - IF (MFILT .LE. 75) GO TO 20
63 - WRITE (108,15)
64 - 15 FORMAT(/, MFILT SET TO MAX ALLOWED VALUE -- MFILT = 75//)
65 - MFILT = 75
66 - C
67 - C
68 - C
69 - FIND NORMALIZED STOP FREQUENCY, LINEAR RIPPLES AND WEIGHT.
70 - FCU = STOPF/PRF
71 - CRP = 10.**((RFB/20.))
72 - E1 = (CRP-1.)/(CRP+1.)
73 - E2 = 10.**((-ASB/20.))*(1.-E1)
74 - WEIGHT = E1/E2
75 - DENOM = 2.+(2.*E1*E1-E2*E2)
76 - D1 = 4.*E1/DENOM
77 - D2 = E2*E2/DENOM
78 - WTX = (4.*E1)/(E2*E2)
79 - IF (ITER .NE. 0) GO TO 50
80 - NFILT = 2*MFILT-1
    NFCNS = MFILT

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDG

```

81 - LGRIDM = 1200/NFCNS
82 - C
83 - PROGRAM DESIGN MAX 75 TAPS / LGRID = 1200/MFILT
84 - C
85 - IF (LGRID .LE. 0) LGRID = LGRIDM
86 - ITEST = LGRID*NFCNS
87 - C
88 - ENSURE AT LEAST 2 GRID POINTS IN STOPBAND.
89 - IF (2./FLOAT(ITEST) .LE. FCU) GO TO 30
90 - LGRID = 2./FCU*FLOAT(NFCNS))+1.
91 - ITEST = LGRID*NFCNS
92 - IF (ITEST .LE. 1200) GO TO 31
93 - LGRID = LGRIDM
94 - IF (SPTFLG) GO TO 50
95 - C
96 - ESTIMATE PASSBAND FREQUENCY
97 - IF (FCU .GE. 0.04) GO TO 35
98 - C
99 - CHERYSHEV LOWER BOUND ESTIMATE FOR FCU .11. 0.04
100 - C
101 - XT = (1.+D1)/D2
102 - YT = (1.-D1)/D2
103 - COSHIX = ALOG(XT+SQRT((XT+1.)*(XT-1.)))
104 - COSHIY = ALOG(YT+SQRT((YT+1.)*(YT-1.)))
105 - DELF=(COSHIX-SQRT((COSHIX+COSHIY)*(COSHIX-COSHIY)))/(PI*(NFILT-1))
106 - GO TO 45
107 - C
108 - C HERRMANN ESTIMATE (BSTJ) FOR FCU .GE. 0.04
109 - C
110 - 35 D1L = ALOG10(D1)
111 - D2L = ALOG10(D2)
112 - FK = 11.01217 + 0.51244 * ALOG10(WTX)
113 - D1NF = ((0.005309*D1L+0.07114)*D1L-0.4761)*D2L
114 - * -(0.00266*D1L+0.5941)*D1L-0.4278
115 - DELF = (NFILT-1)*(SQRT(1.+4.*FK*D1NF/(NFILT-1)**2)-1.)/(2.*FK)
116 - FUP = FCU+DELF
117 - IF (FUP .GE. 0.4) FUP = 0.4
118 - FST = 0.4*DELF
119 - ELAST = 0.
120 - FLAST = 0.

```

```

121 - DEL = 0.0001*DI2
122 - IF (FUP,LE,0.45) GO TO 50
123 - WRITE(108,40) FUP
124 - FORMAT(1H0,'++UNSUCCESSFUL DESIGN BECAUSE PASSF/PRF = ',F6.4,
125 - *', WHICH IS GREATER THAN 0.4 ++')
126 - GO TO 500
127 - CALL HPFH
128 - ITER = ITER+1
129 - C
130 - C
131 - C
132 - CHECK FOR CONVERGENCE
133 - ERROR = D2-FCNFI2
134 - IF (ABS(ERROR) .LE. DEL) GO TO 300
135 - FTEMP = FUP
136 - FUP = FLAST-((FLAST-FUP)*ELAST/(ELAST-ERROR))
137 - FLAST = FTEMP
138 - ELAST = ERROR
139 - IF (FUP) 60,60,49
140 - FUP = FLAST-SIGN(FST,ERROR)
141 - GO TO 50
142 - C
143 - C
144 - C
145 - C
146 - CALCULATE IMPULSE RESPONSE OF
147 - SQUARED FILTER WITH
148 - LINEAR PHASE.
149 -
150 - 300 NM1 = NFCNS-1
151 - NZ = NFCNS+1
152 - DO 305 J = 1,NM1
153 - H(J) = 0.5*ALPHA(NZ-J)
154 - H(NFILT+1-J) = H(J)
155 - H(NFCNS) = ALPHA(1)+DEV/WT(1)
156 - C
157 - C
158 - C
159 - C
160 - SOLVE FOR MIXED PHASE
IMPULSE RESPONSE.
CALL RTSQF(H,NFILT,FCU,ROOTR,ROOTI,MUN,MOFF,NCOMB,NBIC)
EXPAND ROOTS TO DERIVE MAXIMUM PHASE IMPULSE RESPONSE.
DTEMP = DABS(H(1))

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

161 - CALL EXPAND(MON+MOFF,ROOTR,ROOTI,MTERMS,Z,H)
162 - IF (MTERMS.EQ. MFILT) GO TO 340
163 - IF (LGRID.EQ. LGRIDM) GO TO 333
164 - C
165 - C
166 - C
167 - C
168 - LGRID = LGRID*2
169 - SPTFLG = .TRUE.
170 - GO TO 25
171 - C
172 - C
173 - C
174 - C
175 - C
176 - C
177 - C
178 - C
179 - C
180 - C
181 - C
182 - C
183 - C
184 - C
185 - C
186 - C
187 - C
188 - C
189 - C
190 - C
191 - C
192 - C
193 - C
194 - C
195 - C
196 - C
197 - C
198 - C
199 - C
200 - C

333 WRITE (108,335) MFILT,MTERMS,LGRIDM
335 FORMAT(' ERROR: MTERMS .NE. MFILT. MFILT =',I4,' MTERMS =',
* I4,' USING MAXIMUM ALLOWED VALUE FOR LGRID:',I5,'')
GO TO 350

FIND THE OPTIMUM MIXED - PHASE
FILTER.

340 CALL OPTMPF(MODE,MON,MOFF,ROOTR,ROOTI,MTERMS,Z)

PROGRAM OUTPUT SECTION.

350 SCALE = 2.DO/(DSQRT(1.DO+(ESDEL1+ESDEL2))
* +DSQRT(1.DO-(ESDEL1-ESDEL2)))
ZNORM = SCALE*DSQRT(DTEMP)
E1C = 0.5*SCALE*
* (DSQRT(1.DO+(ESDEL1+ESDEL2))-DSQRT(1.DO-(ESDEL1-ESDEL2)))
E2C = SCALE*DSQRT(2.DO*ESDEL2)
PASSF = FUP*PRF
HPRF = PRF/2.00
DBSTOP = -20.* ALOG10(E2C/(1.-E1C))
DBPASS = 20.* ALOG10( (1.+E1C)/(1.-E1C) )

NORMALIZE FILTER TO UNITY AMPLITUDE GAIN IN PASSBAND AND
CALCULATE NOISE POWER GAIN (PGN) IN DB

PGN = 20.*ALOG10(1.+E1C)
SMGN = 20.*ALOG10(1.-E1C)
SUM = 0.00
DO 370 K = 1,MFILT

```

```

201 -
202 -
203 -
204 -
205 -
206 -
207 -
208 -
209 -
210 -
211 -
212 -
213 -
214 -
215 -
216 -
217 -
218 -
219 -
220 -
221 -
222 -
223 -
224 -
225 -
226 -
227 -
228 -
229 -
230 -
231 -
232 -
233 -
234 -
235 -
236 -
237 - C
238 - C
239 - C
240 -

      Z(K) = Z(K)*ZNORM
370 SUM = SUM+Z(K)*Z(K)
      PGN = 10.00*ALOG10(SUM)
      SRTSUM = SQRT(SUM)
      IF (ABS(SMGN-PGN) .LE. 0.001) GO TO 358
      RFB = PKGN-PGN
      GO TO 20

358 WRITE(108,359)
359 FORMAT(1X)
      CALL NEWPAGE(108)
      WRITE(108,360)
360 FORMAT(1H1, 82(1H*)//5X,'FINITE IMPULSE RESPONSE (FIR)',
      * , OPTIMUM MIXED PHASE DIGITAL FILTER DESIGN' / 10X,
      * , FOR REMOVING GROUND CLUTTER IN MTI RADAR SIGNAL PROCESSOR'//)
      WRITE (108,361) MFILT,LGRID,ITER
361 FORMAT(23X,I4,' TAP FILTER'
      * , (LGRID = ',I4,')'//23X,'CONVERGENCE AFTER',I3,
      * , ITERATIONS'//)
      IF (MODE.EQ. 1) WRITE(108,365)
365 FORMAT(/31X,'MINI-MAX FILTER'//)
      IF (MODE.EQ. 2) WRITE(108,366)
366 FORMAT(/25X,'MINIMUM ABSOLUTE SUM FILTER'//)
      IF (MODE.EQ. 3) WRITE(108,367)
367 FORMAT(/23X,'MINIMUM ABSOLUTE VARIANCE FILTER'//)
      WRITE (108,390)
390 FORMAT(/31H BAND LOWER EDGE UPPER EDGE ,5X,
      * ,WEIGHT',5X,'RIPPLE',5X,'RIPPLE(DB)'/2X,66(1H-))
      WRITE (108,410) STOPF, WEIGHT, E2C, DBSTOP
410 FORMAT(2X,'STOP
      * ,2(1H-))//)
      WRITE (108,420) PASSF, HPRF, E1C, DBPASS
420 FORMAT (2X,'PASS',2(F12.2),8X,'1.00',F11.5,F14.2
      * /2X,66(1H-))//)
      WRITE (108,373) PGN,PKGN,SMGN,(Z(J),J = 1,MFILT)
373 FORMAT(1H0,' ORIGINAL TAP GAINS: NOISE POWER GAIN = ',F7.3,
      * , DB.'//23X,'MAX GAIN IN PASSBAND =',F7.3,
      * , DB.'//23X,'MIN GAIN IN PASSBAND =',F7.3,' DB.'//2X,10F10.5))

      NORMALIZE Z(K) W.R.T PGN AND COMPUTE NEW PGN = 0 DB
      PKGN = PKGN-PGN

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

241 - SMGN = SMGN-PGN
242 - SUM = 0.
243 - ZMAX = 0.
244 - ZMIN = 7.237E75
245 - DABSUM = 0.00
246 - DO 380 K = 1,MFILT
247 - Z(K) = Z(K)/SRTSUM
248 - DTEMP = DABS(Z(K))
249 - DABSUM = DABSUM+DTEMP
250 - IF (DTEMP .LE. ZMAX) GO TO 375
251 - ZMAX = DTEMP
252 - KMAX = K
253 - 375 IF (DTEMP .GE. ZMIN) GO TO 380
254 - ZMIN = DTEMP
255 - KMIN = K
256 - 380 SUM = SUM+Z(K)*Z(K)
257 - PGN = 10.* ALOG10(SUM)
258 - WRITE (108,385) PGN,PKGN,SMGN,(Z(J),J = 1,MFILT)
259 - 385 FORMAT(/1H0,' NORMALIZED TAP GAINS: NOISE POWER GAIN = ',F7.3,
260 - * , DB,'//25X,'MAX GAIN IN PASSBAND = 'F7.3,
261 - * , DB,'//25X,'MIN GAIN IN PASSBAND = 'F7.3,' DB,'//2X,10F10.5))
262 - TAPRAT = ZMAX/ZMIN
263 - WRITE (108,387) KMAX,KMIN,TAPRAT,DABSUM,NCOMB,NBIG
264 - 387 FORMAT(/'0 TAP',I3,' HAS GREATEST MAGNITUDE,'
265 - * ,'0 TAP',I3,' HAS SMALLEST MAGNITUDE,'
266 - * ,'0 RATIO OF GREATEST TO SMALLEST TAP WEIGHTS IS',F8.2
267 - * ,'0 ABSOLUTE SUM OF TAP MAGNITUDES IS',F7.3
268 - * ,'0',I2,' PAIRS OF ROOTS COMBINED,'
269 - * ,'0',I2,' LARGE ROOTS DISCARDED.'//)
270 - DO 450 J = 1,NZ
271 - 450 EXT(J) = GRID(IEXT(J))*PRF
272 - WRITE(108,455) (EXT(J),J = 1,NZ)
273 - 455 FORMAT(/' EXTREMAL FREQUENCIES (HZ) '
274 - * // (10F10.3/ )
275 - WRITE(108,460)
276 - 460 FORMAT(1H0,82(1H*))
277 - C
278 - C OUTPUT FILTER PARAMETERS AND
279 - C WEIGHTS FOR PLOTTING.
280 - C

```

```

281 -
282 -
283 -
284 -
285 -
286 -
287 -
288 -
289 -
290 -
291 -
292 -

      WRITE(106,480) MFILT,2,LCRID,STOFF,PASSF,PRF,1,WEIGHT,
      *      DBSTOP,DBFPASS,FGN,PKGN,MODE
280  FORMAT(13,I1,14,2F8.3,F5.0,F5.3,5F8.3,I1)
      WRITE(106,490) (Z(J),J = 1,MFILT)
290  FORMAT(10F8.5)
      CONTINUE
      J = NTIMER(1)
      TEMP = FLOAT(J)/500.
      WRITE (108,510) TEMP
291  FORMAT(/, COMPUATION REQUIRED',F7.2,' SECONDS OF CPU TIME.'/)
      GO TO 10
      END

```

APPENDIX C

SUBROUTINES

```

1 - SUBROUTINE HPFH
2 - COMMON /HFFC/ F1,STOFF,PASSF,WEIGHT,RATIO,ESDEL1,ESDEL2,
3 - * NFILT,NEG,NOID,LGRID
4 - COMMON F12,AD,DEV,X,Y,GRID,DES,WT,ALPHA,IEXT,NFCNS,NGRID
5 - DIMENSION IEXT(79),AD(79),ALPHA(79),X(79),Y(79),EDGE(4),
6 - * DES(1200),GRID(1200),WT(1200)
7 - DOUBLE PRECISION F12,F1
8 - DOUBLE PRECISION AD,DEV,X,Y
9 - DATA EDGE(1)/0./,EDGE(4)/0.5/
10 - EDGE(2) = STOFF
11 - EDGE(3) = PASSF
12 - C** FIND THE DESIRED MAGNITUDE (DES(J)) AND WEIGHT (WT(J)) ON GRID.
13 - 140 GRID(1) = EDGE(1)
14 - DELF = 0.5 / FLOAT(LGRID * NFCNS)
15 - J=1
16 - FUP=EDGE(2)
17 - IF (NEG.EQ.0) GO TO 145
18 - GRID(1) = DELF
19 - 145 TEMP=GRID(J)
20 - DES(J) = 0.
21 - WT(J) = WEIGHT * (1.00-(1.00-RATIO)*TEMP/EDGE(2))
22 - J=J+1
23 - GRID(J)=TEMP+DELF
24 - IF (GRID(J).GT.FUP) GO TO 150
25 - GO TO 145
26 - 150 GRID(J-1)=FUP
27 - WT(J-1)= RATIO * WEIGHT
28 - GRID(J)= EDGE(3)
29 - FUP= 0.50
30 - 155 TEMP= GRID(J)
31 - DES(J)= 1.00
32 - WT(J)= 1.
33 - J= J + 1
34 - GRID(J)= TEMP + DELF
35 - IF (GRID(J).GT.0.50) GO TO 160
36 - GO TO 155
37 - 160 GRID(J-1)= 0.50
38 - NGRID= J-1
39 - C** SET UP APPROXIMATION PROBLEM. WEIGHT BY SIN(PI*GRID(J)) IF NFILT EVEN
40 - IF (NEG.EQ.1) GO TO 175

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

41 - IF(NODD, EQ, 1) GO TO 185
42 - C NFILT EVEN; SYMMETRY POS.
43 - IF(GRID(NGRID).GT.(0.5-DELF)) NGRID=NGRID-1
44 - DO 170 J=1,NGRID
45 - CHANGE=DCOS(PI*GRID(J))
46 - DES(J)=DES(J)/CHANGE
47 - 170 WT(J)=WT(J)*CHANGE
48 - GO TO 185
49 - C NFILT EVEN; SYMMETRY NEG.
50 - 175 DO 176 J=1,NGRID
51 - CHANGE=DSIN(PI*GRID(J))
52 - DES(J)=DES(J) / CHANGE
53 - 176 WT(J)= WT(J) * CHANGE
54 - C NFILT ODD; SYMMETRY POS.
55 - C INITIAL GUESS FOR EXTREMAL FREQUENCIES IS EQUALLY SPACED ON GRID
56 - 185 TEMP=FLOAT(NGRID-1)/FLOAT(NFCNS)
57 - DO 190 J=1,NFCNS
58 - IEXT(J)=(J-1)*TEMP+1
59 - IEXT(NFCNS+1)=NGRID
60 - CALL REMEZ(EDGE,2)
61 - EDEL1 = DEV
62 - EDEL2 = DEV / WEIGHT
63 - RETURN
64 - END

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDG

```

1 - SUBROUTINE RTSQF(H,NFILT,FCU,ROOTR,ROOTI,MON,MOFF,NCOMB,NBIG)
2 - HERRING 28 JULY, 1977.
3 -
4 - SUBROUTINE TO FIND AND SORT ROOTS OF SQUARED FILTER
5 - RESPONSE IN DESIGN OF MIXED-PHASE FILTERS.
6 -
7 - COMMON PI2,AD,DEV,X,Y,GRID,DES,WT,ALPHA,IEXT,NFCNS,NGRID
8 - DIMENSION H(150),ROOTR(150),ROOTI(150)
9 - DIMENSION IEXT(79),AD(79),ALPHA(79),X(79),Y(79)
10 - DIMENSION DES(1200),GRID(1200),WT(1200)
11 - DOUBLE PRECISION PI2
12 - DOUBLE PRECISION AD,DEV,X,Y
13 - DOUBLE PRECISION H,ROOTR,ROOTI,DTEMP,PSPLIT
14 -
15 - NCOMB = 0
16 - NBIG = 0
17 -
18 - FIND ROOTS OF SQUARED FILTER IMPULSE RESPONSE.
19 -
20 - NROOTS = NFILT-1
21 - CALL MULPOL(H,NROOTS,ROOTR,ROOTI)
22 -
23 - CONVERT ROOTS TO POLAR FORM AND RETAIN ONLY THOSE ROOTS
24 - ON UPPER HALF OF Z-PLANE AND ON OR OUTSIDE THE
25 - UNIT CIRCLE.
26 -
27 - NREAL = 0
28 - K = 0
29 - DO 320 J = 1,NROOTS
30 - ELIMINATE SPLITTING OF ROOTS NEAR REAL AXIS.
31 - IF (DABS(ROOTI(J)) .GT. 2.D-5) GO TO 300
32 - ROOTI(J) = 0.D0
33 - COUNT AS REAL ROOT IF IT LIES ON OR OUTSIDE UNIT CIRCLE.
34 - IF (DABS(ROOTR(J)) .GE. 0.99998D0) NREAL = NREAL+1
35 - DTEMP = ROOTR(J)*ROOTR(J)+ROOTI(J)*ROOTI(J)
36 - MOVE NEARBY ROOTS ONTO UNIT CIRCLE.
37 - IF (DABS(DTEMP-1.D0) .GT. 4.D-5) GO TO 310
38 - DTEMP = 1.D0
39 -
40 - ROOTI(J) = DATAN2(ROOTI(J),ROOTR(J))

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

41 - ROOTR(J) = DTEMP
42 - IF (ROOTI(J) .LT. 0.00) GO TO 320
43 - IF (ROOTR(J) .LT. 1.00) GO TO 320
44 - K = K+1
45 - ROOTR(K) = DSQRT(ROOTR(J))
46 - ROOTI(K) = ROOTI(J)
47 - 320 CONTINUE
48 - C
49 - C
50 - C
51 - SORT ROOTS IN ORDER OF DESCENDING MAGNITUDE.
52 -
53 - JROOTS = K
54 - DO 330 J = 2, JROOTS
55 - KHI = JROOTS+1-J
56 - DO 330 K = 1, KHI
57 - IF (ROOTR(K) .GE. ROOTR(K+1)) GO TO 330
58 - DTEMP = ROOTR(K)
59 - ROOTR(K) = ROOTR(K+1)
60 - ROOTR(K+1) = DTEMP
61 - ROOTI(K) = ROOTI(K+1)
62 - ROOTI(K+1) = DTEMP
63 - 330 CONTINUE
64 - C
65 - C
66 - COUNT ROOTS ON UNIT CIRCLE.
67 -
68 - DO 340 J = 1, JROOTS
69 - J1 = JROOTS+1-J
70 - IF (ROOTR(J1) .NE. 1.00) GO TO 350
71 - 340 CONTINUE
72 - J1 = JROOTS-J1
73 - IF (J1 .LE. 1) GO TO 400
74 - C
75 - C
76 - SORT ROOTS ON UNIT CIRCLE IN ORDER OF
77 - ASCENDING PHASE.
78 -
79 - KLO = JROOTS+1-J1
80 - DO 360 J = 2, J1
    KHI = JROOTS+1-J
    DO 360 K = KLO, KHI
    IF (ROOTI(K) .LT. ROOTI(K+1)) GO TO 360

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

41 - ROOTR(J) = DTEMP
42 - IF (ROOTI(J) .LT. 0.00) GO TO 320
43 - IF (ROOTR(J) .LT. 1.00) GO TO 320
44 - K = K+1
45 - ROOTR(K) = DSQRT(ROOTR(J))
46 - ROOTI(K) = ROOTI(J)
47 - 320 CONTINUE
48 - C
49 - C
50 - C
51 - JROOTS = K
52 - DO 330 J = 2, JROOTS
53 - KHI = JROOTS+1-J
54 - DO 330 K = 1, KHI
55 - IF (ROOTR(K) .GE. ROOTR(K+1)) GO TO 330
56 - DTEMP = ROOTR(K)
57 - ROOTR(K) = ROOTR(K+1)
58 - ROOTR(K+1) = DTEMP
59 - DTEMP = ROOTI(K)
60 - ROOTI(K) = ROOTI(K+1)
61 - ROOTI(K+1) = DTEMP
62 - 330 CONTINUE
63 - C
64 - C
65 - C
66 - COUNT ROOTS ON UNIT CIRCLE.
67 - DO 340 J = 1, JROOTS
68 - J1 = JROOTS+1-J
69 - IF (ROOTR(J1) .NE. 1.00) GO TO 350
70 - 340 CONTINUE
71 - J1 = JROOTS-J1
72 - IF (J1 .LE. 1) GO TO 400
73 - C
74 - C
75 - C
76 - SORT ROOTS ON UNIT CIRCLE IN ORDER OF
77 - ASCENDING PHASE.
78 - KLO = JROOTS+1-J1
79 - DO 360 J = 2, J1
80 - KHI = JROOTS+1-J
81 - DO 360 K = KLO, KHI
82 - IF (ROOTI(K) .LT. ROOTI(K+1)) GO TO 360

```

```

81 - DTEMP = ROOTR(K)
82 - ROOTR(K) = ROOTR(K+1)
83 - ROOTR(K+1) = DTEMP
84 - DTEMP = ROOTI(K)
85 - ROOTI(K) = ROOTI(K+1)
86 - ROOTI(K+1) = DTEMP
87 - 360 CONTINUE
88 - C
89 - C
90 - C
91 - PSPLIT = PI2*FCU/FLOAT(2*J1+1)
92 - K = KLO
93 - KHI = JROOTS-1
94 - J = 0
95 - 370 IF (ROOTI(K+1)-ROOTI(K) .GE. 1)SPLIT) GO TO 380
96 - ROOTI(KLO+J) = 0.500*(ROOTI(K)+ROOTI(K+1))
97 - IF (ROOTI(KLO+J) .EQ. 0.00) NREAL = NREAL-1
98 - J = J+1
99 - K = K+1
100 - 380 K = K+1
101 - IF (K .LE. KHI) GO TO 370
102 - ROOTI(KLO+J) = ROOTI(K)
103 - JROOTS = JROOTS-J
104 - J1 = J1-J
105 - NCOMB = J
106 - C
107 - C
108 - C
109 - IF (2*(JROOTS-NREAL)+NREAL .EQ. NROOTS/2) GO TO 400
110 - C
111 - C
112 - C
113 - DO 390 J = 2, JROOTS
114 - ROOTR(J-1) = ROOTR(J)
115 - ROOTI(J-1) = ROOTI(J)
116 - 390 CONTINUE
117 - JROOTS = JROOTS-1
118 - NBIG = 1
119 - 400 MON = J1
120 - MOFF = JROOTS-J1

```

RETURN
END

121 - -
122 - -

```

1 - C
2 - C
3 - C
4 - C
5 - C
6 - C
7 - C
8 - C
9 - C
10 - C
11 - C
12 - C
13 - C
14 - C
15 -
16 -
17 -
18 -
19 -
20 -
21 -
22 - 19
23 - 7
24 -
25 -
26 -
27 -
28 - 9
29 - 10
30 -
31 -
32 -
33 -
34 -
35 -
36 -
37 - 11
38 -
39 -
40 -

```

SUBROUTINE MULPOL(COE,N1,ROOTR,ROOTI)

SUBROUTINE MULPOL FACTORS A POLYNOMIAL BY MULLER'S ALGORITHM

SOURCE : D.S. HUMPHREYS, 'THE ANALYSIS, DESIGN AND SYNTHESIS
OF ELECTRICAL FILTERS', PP 649-652,
PRENTICE-HALL, 1970.

DIMENSION COE(1),ROOTR(1),ROOTI(1)

FOR DOUBLE PRECISION VERSION REMOVE THE C IN COLUMN 1
OF THE FOLLOWING LINE:

DOUBLE PRECISION COE,ROOTR,ROOTI

DOUBLE PRECISION ALP1R,ALP1I,ALP2R,ALP2I,ALP3R,ALP3I,ALP4R,ALP4I,
* AXR,AXI,HELL,BELL,BET1R,BET1I,BET2R,BET2I,BET3R,BET3I,
* DE15,DE16,TEM,TEMR,TEMI,TEM2,TE1,TE2,TE3,TE4,TE5,TE6,
* TE7,TE8,TE9,TE10,TE11,TE12,TE13,TE14,TE15,TE16

N2 = N1+1
N4 = 0
I = N1+1
IF (COE(I)) 7,7,9
N4 = N4+1
ROOTR(N4) = 0.00
ROOTI(N4) = 0.00
J = J-1
IF (N4-N1) 19,37,19
CONTINUE
AXR = 0.800
AXI = 0.000
L = 1
N3 = 1
ALP1R = AXR
ALP1I = AXI
M = 1
GO TO 99
BET1R = TEMR
BET1I = TEMI
AXR = 0.8500
ALP2R = AXR

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

41 - ALP2I = AXI
42 - M = 2
43 - GO TO 99
44 - 12 BET2R = TEMR
45 - BET2I = TEMI
46 - AXR = 0.9D0
47 - ALP3R = AXR
48 - ALP3I = AXI
49 - M = 3
50 - GO TO 99
51 - 13 BET3R = TEMR
52 - BET3I = TEMI
53 - 14 TE1 = ALP1R-ALP3R
54 - IE2 = ALP1I-ALP3I
55 - TE5 = ALP3R-ALP2R
56 - TE6 = ALP3I-ALP2I
57 - TEM = TE5*TE5+TE6*TE6
58 - TE3 = (TE1*TE5+TE2*TE6)/TEM
59 - TE4 = (TE2*TE5-TE1*TE6)/TEM
60 - TE7 = TE3+1.0D0
61 - TE9 = TE3*TE3-TE4*TE4
62 - TE10 = 2.0D0*TE3*TE4
63 - DE15 = TE7*BET3R-TE4*BET3I
64 - DE16 = TE7*BET3I+TE4*BET3R
65 - TE11 = TE3*BET2R-TE4*BET2I+BET1R-DE15
66 - TE12 = TE3*BET2I+TE4*BET2R+BET1I-DE16
67 - TE7 = TE9-1.0D0
68 - TE1 = TE9*BET2R-TE10*BET2I
69 - TE2 = TE9*BET2I+TE10*BET2R
70 - TE13 = TE1-BET1R-TE7*BET3R+TE10*BET3I
71 - TE14 = TE2-BET1I-TE7*BET3I-TE10*BET3R
72 - TE15 = DE15*TE3-DE16*TE4
73 - TE16 = DE15*TE4+DE16*TE3
74 - TE1 = TE13*TE13-TE14*TE14-4.0D0*(TE11*TE15-TE12*TE16)
75 - TE2 = 2.0D0*TE13*TE14-4.0D0*(TE12*TE15+TE11*TE16)
76 - TEM = DSQRT(TE1*TE1+TE2*TE2)
77 - IF (TE1) 113,113,112
78 - 113 TE4 = DSQRT(.5D0*(TEM-TE1))
79 - TE3 = .5D0*TE2/TE4
80 - GO TO 111

```

```

81 - 112 TE3 = DSQRT(.5D0*(TEM+TE1))
82 - IF (TE2) 110,200,200
83 - TE3 = -TE3
84 - 200 TE4 = .5D0*TE2/TE3
85 - 111 TE7 = TE13+TE3
86 - TE8 = TE14+TE4
87 - TE9 = TE13-TE3
88 - TE10 = TE14-TE4
89 - TE11 = 2.D0*TE15
90 - TE2 = 2.D0*TE16
91 - IF (TE7*TE7+TE8*TE8-TE9*TE9-TE10*TE10) 204,204,205
92 - 204 TE7 = TE9
93 - TE8 = TE10
94 - 205 TEM = TE7*TE7+TE8*TE8
95 - TE3 = (TE1*TE7+TE2*TE8)/TEM
96 - TE4 = (TE2*TE7-TE1*TE8)/TEM
97 - AXR = ALP3R+TE3*TE5-TE4*TE6
98 - AXI = ALP3I+TE3*TE6+TE4*TE5
99 - ALP4R = AXR
100 - ALP4I = AXI
101 - M = 4
102 - GO TO 99
103 - 15 N4 = 1
104 - 38 IF (DABS(HELL)+DABS(HELL)-1.D-20) 18,18,16
105 - 16 TE7 = DABS(ALP3R-AXR)+DABS(ALP3I-AXI)
106 - IF (TE7/(DABS(AXR)+DABS(AXI))-1.D-12) 18,18,17
107 - 17 N3 = N3+1
108 - ALP1R = ALP2R
109 - ALP1I = ALP2I
110 - ALP2R = ALP3R
111 - ALP2I = ALP3I
112 - ALP3R = ALP4R
113 - ALP3I = ALP4I
114 - BET1R = BET2R
115 - BET1I = BET2I
116 - BET2R = BET3R
117 - BET2I = BET3I
118 - BET3R = TEMR
119 - BET3I = TEMI
120 - IF (N3-100) 14,18,18

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

121 - 18      N4 = N4+1
122 -      ROOTR(N4) = ALP4R
123 -      ROOTI(N4) = ALP4I
124 -      N3 = 0
125 - 41      IF (N4-N1) 30,37,37
126 - 37      RETURN
127 - 30      IF (DABS(ROOTI(N4))-1.0-5) 10,10,31
128 - 31      GO TO (32,10),L
129 - 32      AXR = ALP1R
130 -      AXI = -ALP1I
131 -      ALP1I = -ALP1I
132 -      M = 5
133 -      GO TO 99
134 - 33      BET1R = TEMR
135 -      BET1I = TEMI
136 -      AXR = ALP2R
137 -      AXI = -ALP2I
138 -      ALP2I = -ALP2I
139 -      M = 6
140 -      GO TO 99
141 - 34      BET2R = TEMR
142 -      BET2I = TEMI
143 -      AXR = ALP3R
144 -      AXI = -ALP3I
145 -      ALP3I = -ALP3I
146 -      L = 2
147 -      M = 3
148 - 99      TEMR = COE(1)
149 -      TEMI = 0.00
150 -      DO 100 I = 1,N1
151 -      TE1 = TEMR*AXR-TEMI*AXI
152 -      TEMI = TEMI*AXR+TEMR*AXI
153 - 100      TEMR = TE1+COE(I+1)
154 -      HELL = TEMR
155 -      BELL = TEMI
156 - 42      IF (N4) 102,103,102
157 - 102      DO 101 I = 1,N4
158 -      TEM1 = AXR-ROOTR(1)
159 -      TEM2 = AXI-ROOTI(1)
160 -      TE1 = TEM1*TEM1+TEM2*TEM2

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```
161 - TE2 = (TEMR*TEM1+TEMI*TEM2)/TE1  
162 - TEMI = (TEMI*TEM1-TEMR*TEM2)/TE1  
163 - TEMR = TE2  
164 - GO TO (11,12,13,15,33,34),M  
165 - END
```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDG

```

1 - SUBROUTINE OPTMPF(MODE, NON, NOFF, ROOTM, ROOTP, NTERMS, Z)
2 - C
3 - C
4 - C
5 - HERRING 18 FEB, 1977. REVISED 05 AUG, 1977.
6 - C
7 - COMMON P12, AD(79), DEV, X(79), Y(79), GKID(1200), ROOTR(75), ROOTT(75),
8 - * ROOTI(75), ZON(75), ZOFF(75), ZTEMP(75), WORK(150)
9 - * DOUBLE PRECISION P12, AD, DEV, X, Y, ROOTR, ROOTT, ROOTI, ZON, ZOFF, ZTEMP, WORK
10 - * DOUBLE PRECISION ROOTM(NROOTS), ROOTP(NROOTS), 7(1),
11 - * DTEMP, SIGZ2M, SIGZ2, ZTNORM, RATION,
12 - * RATIO, ZMIN, ZMAX, AZTEMP, AZSUM, AZSUMM, SIGN
13 - C
14 - C
15 - MOVE ROOTS TO LOCAL STORAGE.
16 - C
17 - C
18 - NROOTS = NON+NOFF
19 - DO 100 J = 1, NROOTS
20 - ROOTR(J) = ROOTM(J)
21 - 100 ROOTI(J) = ROOTP(J)
22 - C
23 - C
24 - EXPAND ZEROES ON UNIT CIRCLE
25 - C
26 - CALL EXPAND(NON, ROOTR(NOFF+1), ROOTI(NOFF+1), NZON, ZON, WORK)
27 - C
28 - CALCULATE NUMBER OF COMBINATIONS TO BE TRIED.
29 - C
30 - NTRY = 2** (NOFF-1)
31 - C
32 - TRY THEM.
33 - C
34 - JFLAG = 1
35 - C
36 - SET FILTER INDEX TO 0.
37 - C
38 - J = 0
39 - C
40 - SET INITIAL VALUES TO + INFINITY.
41 - C
42 - SIGZ2M = 1.075
43 - RATION = 1.075
44 - AZSUMM = 1.075

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

41 - C      DECIDE WHETHER TO SET EACH ZERO INSIDE OR OUTSIDE
42 - C      UNIT CIRCLE.
43 - C
44 - C      130 DO 160 K = 1,NOFF
45 - C      J1 = 1+MOD(J/(2*(K-1)),2)
46 - C      GO TO (150,140),J1
47 - C      140 ROOTT(K) = 1.00/ROOTR(K)
48 - C      GO TO 160
49 - C      150 ROOTT(K) = ROOTR(K)
50 - C      160 CONTINUE
51 - C
52 - C      EXPAND ZEROES OFF UNIT CIRCLE.
53 - C
54 - C      CALL EXPAND(NOFF,ROOTT,ROOTI,NZOFF,ZOFF,WORN)
55 - C
56 - C      GENERATE COMPLETE FILTER IMPULSE RESPONSE OF LENGTH
57 - C      NZTEMP = NZOFF+NZON-1.
58 - C
59 - C      CALL POLYMULT(ZTEMP,NZTEMP,ZON,NZON,ZOFF,NZOFF)
60 - C
61 - C      CALCULATE GAIN OF UNNORMALIZED FILTER.
62 - C
63 - C      ZTNORM = 1.00/DSQRT(DARS(ZTEMP(NZTEMP)))
64 - C      GO TO (170,200),JFLAG
65 - C
66 - C      INITIALIZATION.
67 - C
68 - C      170 SIGZ2 = 0.00
69 - C      DTEMP = 0.00
70 - C      ZMIN = 1.075
71 - C      ZMAX = 0.00
72 - C
73 - C      DO 180 K = 1,NZTEMP
74 - C
75 - C      NORMALIZE IMPULSE RESPONSE TAP WEIGHTS.
76 - C
77 - C      ZTEMP(K) = ZTEMP(K)*ZTNORM
78 - C      AZTEMP = DARS(ZTEMP(K))
79 - C
80 - C      FIND AND SAVE MAXIMUM AND MINIMUM WEIGHTS

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

81 - C      IN EACH GENERATED IMPULSE RESPONSE.
82 - C
83 - C      IF (AZTEMP .LE. ZMAX) GO TO 173
84 - C      ZMAX = AZTEMP
85 - C      KMAX = K
86 - C      173 IF (AZTEMP .GE. ZMIN) GO TO 176
87 - C      ZMIN = AZTEMP
88 - C      KMIN = K
89 - C      ACCUMULATE SUM OF ABSOLUTE VALUES OF TAP WEIGHTS.
90 - C
91 - C      176 DTEMP = DTEMP+AZTEMP
92 - C
93 - C      ACCUMULATE SUM OF SQUARES OF TAP WEIGHTS.
94 - C
95 - C      SIGZ2 = SIGZ2+AZTEMP*AZTEMP
96 - C
97 - C      180 CONTINUE
98 - C
99 - C      CALCULATE (NZTEMP-1)*ESTIMATED VARIANCE OF
100 - C      MAGNITUDES OF TAP WEIGHTS.
101 - C
102 - C      SIGZ2 = SIGZ2-(DTEMP*NZTEMP)/NZTEMP
103 - C
104 - C      CALCULATE RATIO OF LARGEST TO SMALLEST TAP WEIGHTS.
105 - C
106 - C      RATIO = DABS(ZTEMP(KMAX))/ZTEMP(KMIN))
107 - C
108 - C      RECORD INDEX NUMBER OF FILTER IF THIS IS SMALLEST
109 - C      RATIO YET ENCOUNTERED.
110 - C
111 - C      IF (RATIO .GE. RATIO0) GO TO 183
112 - C      RATIO0 = RATIO
113 - C      MRATIO = J
114 - C
115 - C      RECORD INDEX NUMBER OF FILTER IF THIS IS SMALLEST
116 - C      ABSOLUTE SUM YET ENCOUNTERED.
117 - C
118 - C      183 AZSUM = DTEMP/NZTEMP
119 - C      IF (AZSUM .GE. AZSUM0) GO TO 187
120 - C

```

```

121 - AZSUM = AZSUM
122 - JSUM = J
123 -
124 - 187 CONTINUE
125 - C
126 - RECORD INDEX NUMBER OF FILTER IF THIS IS SMALLEST
127 - VARIANCE YET ENCOUNTERED.
128 -
129 - IF (SIGZ2 .GT. SIGZ2M) GO TO 190
130 - SIGZ2M = SIGZ2
131 - JMIN = J
132 - C
133 - INCREMENT INDEX.
134 -
135 - 190 J = J+1
136 - IF (J .LT. NTRY) GO TO 130
137 - C
138 - RECOMPUTE OPTIMUM IMPULSE RESPONSE.
139 -
140 - GO TO (191,192,193),MODE
141 - 191 J = MRATIO
142 - GO TO 195
143 - 192 J = JSUM
144 - GO TO 195
145 - 193 J = JMIN
146 - 195 JFLAG = 2
147 - GO TO 130
148 - C
149 - CHECK THAT ALTERNATING SUM IS GREATER THAN 0.
150 -
151 - 200 DTEMP = 0.00
152 - SIGN = (-1.00)**(NZTEMP/2)
153 - DO 210 K = 1,NZTEMP
154 - DTEMP = DTEMP+SIGN*ZTEMP(K)
155 - 210 SIGN = -SIGN
156 - IF (DTEMP .LT. 0.00) ZTNORM = -ZTNORM
157 - C
158 - TRANSFER RESULTS TO OUTPUT ARRAY.
159 -
160 - NTERMS = NZTEMP
      DO 220 K = 1,NZTEMP

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDG

161 - 220 Z(K) = ZTEMP(K)*ZTNORM
162 - RETURN
163 - END

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

1 - C
2 - C
3 - C
4 - C
5 - C
6 - C
7 - C
8 - C
9 - C
10 - C
11 - C
12 - C
13 - C
14 - C
15 - C
16 - C
17 - C
18 - C
19 - C
20 - C
21 - C
22 - C
23 - C
24 - C
25 - C
26 - C
27 - C
28 - C
29 - C
30 - C
31 - C
32 - C
33 - C
34 - C
35 - C
36 - C
37 - C
38 - C
39 - C
40 - C

SUBROUTINE EXPAND(NKROOTS,ROOTM,ROOTP,NTERMS,Z,WORK)
HERRING 04 JAN, 1976.
SUBROUTINE TO EXPAND SET OF ROOTS INTO A POLYNOMIAL
WITH REAL COEFFICIENTS.
ARGUMENTS:
INPUT:
NKROOTS : NUMBER OF ROOTS GIVEN IN ARRAYS ROOTM, ROOTP.
KROOTM : ARRAY OF MAGNITUDES OF ROOTS.
KROOTP : ARRAY OF PHASE ANGLES (IN RADIANS) OF ROOTS.
IF KROOTP(J) .NE. 0. OR PI, THE J'TH ROOT IS TREATED AS
ONE OF A PAIR OF COMPLEX CONJUGATE ROOTS.
OUTPUT:
NTERMS : NUMBER OF COEFFICIENTS IN EXPANDED POLYNOMIAL.
Z : ARRAY OF POLYNOMIAL COEFFICIENTS, IN ORDER OF
ASCENDING POWER.
WORK : TEMPORARY STORAGE ARRAY OF DIMENSION .GE. 2*NKROOTS.
DIMENSION ROOTM(1),ROOTP(1),Z(1),WORK(1),Y(3)
DOUBLE PRECISION ROOTM,ROOTP,Z,WORK,Y,PI
DATA Y(1)/1.DO/
PI = DATAN2(0.DO,-1.DO)
NTERMS = 1
Z(1) = 1.DO
DO 500 J = 1,NKROOTS
MOVE PARTIALLY EXPANDED POLYNOMIAL TO WORK AREA.
DO 100 K = 1,NTERMS
100 WORK(K) = Z(K)
I1W = NTERMS
DECIDE WHETHER SINGLE ROOT OR CONJUGATE PAIR.
IF (KROOTP(J) .EQ. 0.DO) GO TO 200
IF (KROOTP(J) .EQ. PI) GO TO 300
MUST BE CONJUGATE PAIR.

```

```

41 - Y(2) = -2.D0*ROOTM(J)*DCOS(ROOTF(J))
42 - Y(3) = ROOTM(J)*ROOTM(J)
43 - IDY = 3
44 - GO TO 500
45 - C ROOT ON POSITIVE REAL AXIS.
46 - 200 Y(2) = -ROOTM(J)
47 - GO TO 400
48 - C ROOT ON NEGATIVE REAL AXIS.
49 - 300 Y(2) = ROOTM(J)
50 - 400 IDY = 2
51 - 500 CALL POLYMULT(Z, NTERMS, WORK, IUW, Y, IDY)
52 - RETURN
53 - END

```

```

1 - C
2 - C
3 - C
4 - C
5 - C
6 - C
7 - C
8 - C
9 - C
10 - C
11 - C
12 - C
13 - C
14 - C
15 - C
16 - C
17 - C
18 - C
19 - C
20 - C
21 - C
22 - C
23 - C
24 - C
25 - C
26 - C
27 - C
28 - C
29 - C
30 - C
31 - C
32 - C
33 - C
34 - C
35 - C
36 - C
37 - C
38 - C
39 - C
40 - C
41 - C
42 - C
43 - C

SUBROUTINE POLYMULT
PURPOSE
MULTIPLY TWO POLYNOMIALS

USAGE
CALL POLYMULT(Z, IDIMZ, X, IDIMX, Y, IDIMY)

DESCRIPTION OF PARAMETERS
Z - VECTOR OF RESULTANT COEFFICIENTS, ORDERED
FROM SMALLEST TO LARGEST POWER
IDIMZ - DIMENSION OF Z ( CALCULATED )
X - VECTOR OF COEFFICIENTS FOR FIRST POLYNOMIAL,
ORDERED FROM SMALLEST TO LARGEST POWER
IDIMX - DIMENSION OF X (DEGREE IS IDIMX-1)
Y - VECTOR OF COEFFICIENTS FOR SECOND POLYNOMIAL,
ORDERED FROM SMALLEST TO LARGEST POWER
IDIMY - DIMENSION OF Y (DEGREE IS IDIMY-1)

REMARKS
Z CANNOT BE IN THE SAME LOCATION AS X
Z CANNOT BE IN THE SAME LOCATION AS Y

SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
NONE

METHOD
DIMENSION OF Z IS CALCULATED AS IDIMX+IDIMY-1
THE COEFFICIENTS OF Z ARE CALCULATED AS SUM OF PRODUCTS
OF COEFFICIENTS OF X AND Y, WHOSE EXPONENTS ADD UP TO
THE CORRESPONDING EXPONENT OF Z

SUBROUTINE POLYMULT(Z, IDIMZ, X, IDIMX, Y, IDIMY)
DIMENSION X(1), Y(1), Z(1)
DOUBLE PRECISION X, Y, Z
IF(IDIMX*IDIMY) 10, 10, 20
10 IDIMZ=0
GOTO 50

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDG

44
45
46
47
48
49
50
51
52

```

41 - 20 IDIMZ=IDIMX+IDIMY-1
42 - DO 30 I=1, IDIMZ
43 - 30 Z(I)=0
44 - DO 40 I=1, IDIMX
45 - DO 40 J=1, IDIMY
46 - K=I+J-1
47 - Z(K)=X(I)*Y(J)+Z(K)
48 - 40 RETURN
49 - 50 END

```

```

1 - SUBROUTINE NEWPAGE(IUNIT)
2 - C
3 - C
4 - C
5 - C
6 - C
7 - C
8 - DIMENSION I(21)
9 - DATA 1/8Z0B404040,19*(8740404040),8Z40404007/
10 - IUN = JUNIT
11 - IF (IUN .LE. 0) IUN = 108
12 - WRITE (IUN,100) 1
13 - RETURN
14 - 100 FORMAT(21R4)
15 - END

```