# Optimal Design of FIR Digital Filters with Monotone Passband Response

KENNETH STEIGLITZ, SENIOR MEMBER, IEEE

*Abstract*—The application of linear programming to the design of FIR digital filters with constraints on the derivative of the frequency response is described. Numerical considerations in the implementation are discussed and a program is given with examples for the design of filters with optional monotone response in passbands. The method provides the user with an additional degree of flexibility over the Remez exchange algorithm.

Manuscript received June 22, 1978; revised April 30, 1979. This work was supported by the National Science Foundation under Grant GK-42048 and the U.S. Army Research Office (Durham, NC) under Grant DAAG29-75-G-0192.

The author is with the Department of Electrical Engineering, Princeton University, Princeton, NJ 08544.

## I. Introduction

ONE important advantage of linear programming for designing FIR filters [1]-[4], as opposed to the Remez exchange algorithms [9], is the inherent generality of the formulation. Rabiner in [2] points out that constraints on both the frequency and the time response of a filter can be imposed, and gives an example of a design with constraints on the step response. In this paper we show the possibility of adding extra constraints in the frequency domain and apply the technique to the design of FIR filters whose response in passbands is constrained to be monotonically decreasing or

0096-3518/79/1200-0643$00.75 

increasing. The approximate design of such monotonic FIR filters is described in [5]-[8].

We also discuss some important numerical considerations in the implementation of the revised simplex algorithm for filter design, such as the resolution of ties in the cases of degeneracy which arise in this problem, and the choice of tolerance for sign tests. The design of very high-degree filters (length greater than 100) can be made faster by using the FFT in the pricing operation, and we describe this idea and give a timing comparison.

Finally, we give a program and some examples which illustrate the additional flexibility allowed by linear programming.

## II. Basic Formulation of the Problem

We restrict our attention to the case of an odd-length symmetric FIR filter, whose frequency response $F$ is

$$F_j = \sum_{i=0}^{M-1} h_i \cos ij \frac{2\pi}{N} \qquad j = 0, \cdots, N/2 \tag{1}$$

where $M$ is the number of free coefficients (the number of variables), $h_i$ are the coefficients, and there are $N/2 + 1$ grid points from 0 to $\pi$ radians/sample. We will usually use a grid density of 16, by which we mean that $N = 16(2M - 1)$ where $(2M - 1)$ is the *filter length NFILT.*

Let $C_j$ be the desired frequency response at grid point $j$ and let $T_j$ be the desired tolerance at that point. Then the minimax formulation of the design problem is to choose the $h_i$ so that the real number $R$ in

$$|F_j - C_j| \leq RT_j \qquad j = 0, \cdots, N/2 \tag{2}$$

is as small as possible. If the minimum value of $R$ is no greater than 1, then the design achieves the specifications determined by the $C_j$ and $T_j$ in a minimax sense. We write the constraints (2) together with the cost criterion min $R$ as the linear program

$$\begin{aligned} &\max(-R) \\ &F_j + (-R)T_j \leq C_j \\ &-F_j + (-R)T_j \leq -C_j \end{aligned} \tag{3}$$

which we call the *primal problem* with unconstrained variables $h_i, i = 0, \cdots, M - 1$, and $-R$.

The dual of (3) is a linear program in standard form [10], the form usually used for numerical solution [11]. This dual has one equality constraint for each of the unconstrained variables $h_i$ and $-R$ in (3), and one nonnegative variable for each of the inequality constraints in (3), $2N + 2$ altogether. It is usual in these problems to solve the dual by the revised simplex algorithm [10], which we now outline.

The tableau of the dual is shown below

$$\begin{array}{r|c||c|c|l} \text{row } 0 & 0 & \cos ij\, 2\pi/N & -\cos ij\, 2\pi/N & \\ & \vdots & i = 0, \cdots, M-1 & i = 0, \cdots, M-1 & \\ \text{row } M-1 & 0 & j = 0, \cdots, N/2 & j = 0, \cdots, N/2 & . \\ \hline & 1 & T_j & T_j & \\ \hline & -z & C_j & -C_j & \leftarrow \text{cost row} \\ & \uparrow & & & \\ & \text{constant column} & & & \end{array} \tag{4}$$

The solution to this program will have at most $M + 1$ positive dual variables, each corresponding to a column; and each positive dual variable and its column will correspond to a constraint in (3) which holds with equality. In general, therefore, the solution response will have $M + 1$ ripples (but may have more).

The revised simplex algorithm proceeds at each pivot as follows (for more details, see [10]):

1) The columns are *priced* by calculating a relative cost $\bar{c}_j$ for each column. If no $\bar{c}_j$ is negative we have reached optimality; otherwise, the most negative $\bar{c}_j$ determines the column $k$ to enter the basis.

2) Column $k$ is *generated* by using an $m \times m$ inverse-basis matrix which is carried along from pivot to pivot (array CARRY in the program).

3) The usual ratio test is used to choose a row $l$, which then determines the column which is to leave the basis.

4) The pivoting operation is used to update the inverse-basis matrix and the *shadow* prices $\pi_j$, which are used to price the columns in Step 1.

We will use the usual two-phase method, where phase 1 is used to obtain a feasible solution to the dual, and phase 2 is used to drive that solution to optimality.

The simplex algorithm thus applied can be viewed as a "single exchange algorithm," as mentioned in [9]. That is, at each pivot one extremum of the frequency response error (column of the dual) is traded for another. In the Remez exchange algorithms, all extrema are changed at each iteration, but such an iteration step may be more time-consuming than a simplex pivot. In general, however, the Remez algorithms are faster for the basic minimax problem.

## III. Additional Constraints on the Frequency Response

Any constraints which are linear in the parameters $h_i$ and $R$ can be added very conveniently. We describe here the introduction of constraints on the derivative of the frequency response. Equation (1) can be thought of as a function of the continuous variable $\omega$

$$F(\omega) = \sum_{i=0}^{M-1} h_i \cos i\omega \tag{5}$$

and therefore,

$$\frac{dF(\omega)}{d\omega} = \sum_{i=0}^{M-1} h_i(-i \sin i\omega). \tag{6}$$

Thus, the constraints

$$\pm \frac{dF(\omega)}{d\omega} \leq 0 \tag{7}$$

can be added to the original set in (3) at any subset of the points $\omega = 2\pi j/N$, $j = 0, \cdots, N/2$. The effect is to add *columns* to the dual in (4), but no more rows. The only effect on the algorithm itself is to increase the number and variety of columns that need to be priced and that may have to be generated to enter the basis. In this case the columns are of the form $(\mp i \sin ij\, 2\pi/N)$, and the corresponding entries $T_j$ and $C_j$ are zero. It is easy to see, however, how we could add constraints of the form

$$\left| \frac{d^r F(\omega)}{d\omega^r} - C(\omega) \right| \leq T(\omega), \tag{8}$$

or even more general constraints, in arbitrary regions of the frequency axis [13].

In the program given here we can choose one of (7), or no derivative constraint at all, in each of the specification bands.

## IV. Numerical Considerations

A very important problem that arises in this program stems from the fact that the original solution at the start of phase 1 in the solution of the dual is highly degenerate—that is, $M$ out of $(M + 1)$ variables have the value zero, as can be seen from (4). This implies that there will be many ties in step 3 of the simplex algorithm, the ratio test. If no care is taken in resolving these ties, we are doing the equivalent of solving a set of simultaneous equations using Gauss reduction with no pivot selection. We found that the following algorithm works well: from among the rows for which a tie occurs in the ratio test, choose one that corresponds to the largest pivot.

There are two important places in the simplex algorithm where quantities must be tested against zero: in the pricing operation and in the ratio test. One must choose small numbers to use here; if these numbers are too large, valid pivots will be overlooked and suboptimal solutions produced; if they are too small, accumulated roundoff errors will induce invalid pivots. We used double precision arithmetic on an IBM 360 series computer (64 bit floating-point numbers), and we found that a threshold of EPS = $10^{-8}$ in both places was reliable. On some problems with a filter length greater than 100, however, there tends to be a long "tail" of convergence, and slightly suboptimal solutions may be obtained. This is typical of such linear programming applications.

TABLE I
Summary of Results in Example 1

| Filter Length | Constraints | Time: s | Deviation: dB | Pivots: phase 1 | Total |
|---|---|---|---|---|---|
| 33 | none | 1.24 | -15.63/-55.64 | 25 | 70 |
| 33 | monotone in passband | 2.59 | -10.05/-50.05 | 28 | 101 |

A good measure of how well phase 1 does is the cost at its end. In theory, since the artificial variables are all driven out, we should terminate with $z = 0$. In practice it is usually of the order of $10^{-11}$ or smaller in magnitude.

## V. Efficiency Considerations—FFT Pricing

The usual care must be exercised to produce reasonable run times. Thus, tables of cosines and sines are kept for column generation, and these are filled once at the start of execution. Also, a logical array KSTAR is kept which tells us when a column can be skipped in the pricing—either because it doesn't correspond to a band constraint we wish to impose, or because it is in the basis already.

The pricing operation is the most time-consuming part of the simplex algorithm. It involves computing an inner product of the current shadow prices with a column, which is either a cosine or sine vector. Thus, the pricing operation is essentially a cosine or sine transform, and can be accomplished with an FFT algorithm [12]. The direct method involves $0(MP)$ operations, where $P$ is the number of columns; and FFT pricing involves $0(P \log P)$ operations [14]. For sufficiently large $M$ we would therefore expect FFT pricing to be profitable. Experiments on the IBM 360/91 computer using Fortran H show that the break-even point is roughly between NFILT = 65 and 127, with not much difference between the two methods at these filter lengths. The idea was therefore not used in the program, but may be attractive when special FFT hardware is available.

## VI. Computational Results and Examples

We will give two design examples, the first simple and the second more complicated. A grid density of 16 was used throughout, and the times are for the Fortran H compiler on an IBM 360/91 computer.

*Example 1:* This is a straightforward low-pass problem with the passband from 0 to 128/512 and the stopband from 152/512 to 1/2. The tolerances were chosen to be 1.0 in the passband and 0.01 in the stopband. The results are summarized in Table I and the response curves are shown in Figs. 1 and 2. As expected, there is a difference of 40 dB between the deviation in the pass- and stopbands because of the difference in tolerance. Also, as one might guess, we sacrifice some rejection for the monotonic response in the passband, in this case about 5.6 dB. What is surprising, perhaps, is the fact that the monotone design is very flat over most of the passband, using most of its slack at the edge of the band closest to the transition region. This effect can be anticipated in the specifications, and there is a great deal of freedom available to the designer in choosing band edges and tolerances.

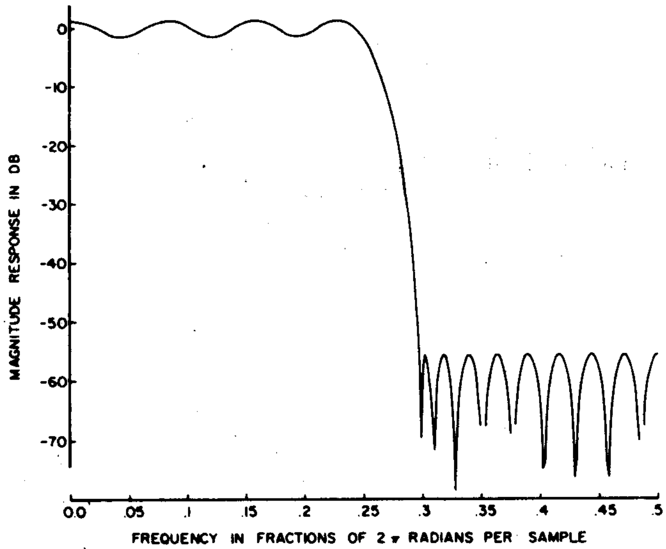


Fig. 1. Magnitude frequency response of the conventional design for Example 1.

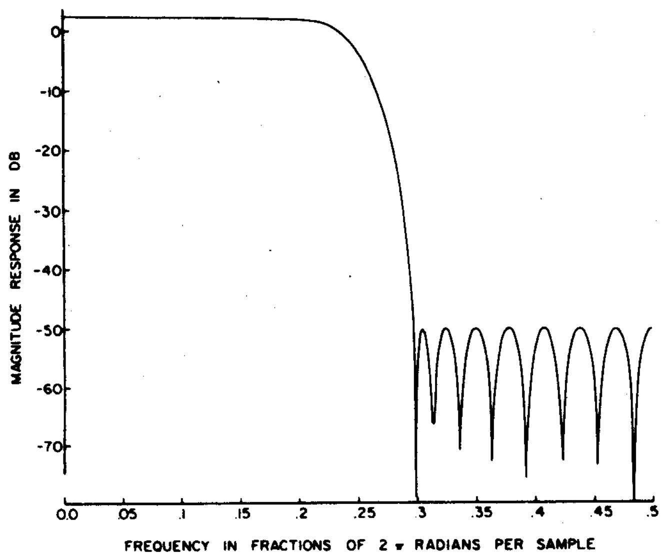


Fig. 2. Magnitude frequency response of the monotone passband design for Example 1.

For purposes of comparison, the program of [9] took 0.49 s on the same machine for a design which checks that shown in Fig. 1. We note also that a length 65 monotone design took 4.59 s, and a length 127 monotone design took 34.81 s.

*Example 2:* This is a bandstop specification with a passband from 0 to 140/512, a stopband from 160/512 to 190/512, and another passband from 210/512 to 1/2. The tolerances were chosen to be 1.0 in the passbands, and 0.01 in the stopband, as before, so that the passband ripple is visible without scale change on the drawing. The results are summarized in Table II and the response curves are shown in Figs. 3 and 4. The results are entirely as expected, with a sacrifice of about 7.8 dB rejection for the flat passbands (or an equivalent increase in filter order). Also as before, there is about a two-fold increase in execution time with the addition of the derivative constraints, caused not by a comparable increase in the number of pivots, but by the increased cost of pricing.

TABLE II
SUMMARY OF RESULTS IN EXAMPLE 2

| Filter Length | Constraints | Time: s | Deviation: dB | Pivots: phase 1 | Total |
|---|---|---|---|---|---|
| 65 | none | 6.89 | -29.96/-69.96 | 46 | 162 |
| 65 | [mono. down / none / mono. up] | 13.73 | -22.15/-62.15 | 47 | 164 |

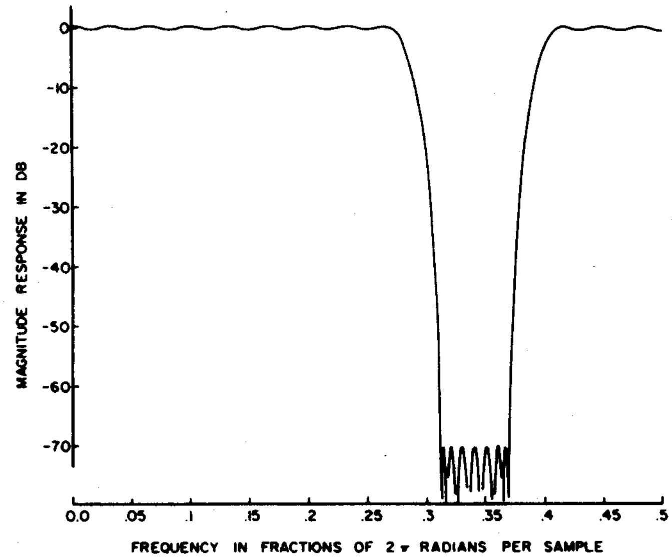


Fig. 3. Magnitude frequency response of the conventional design for Example 2.

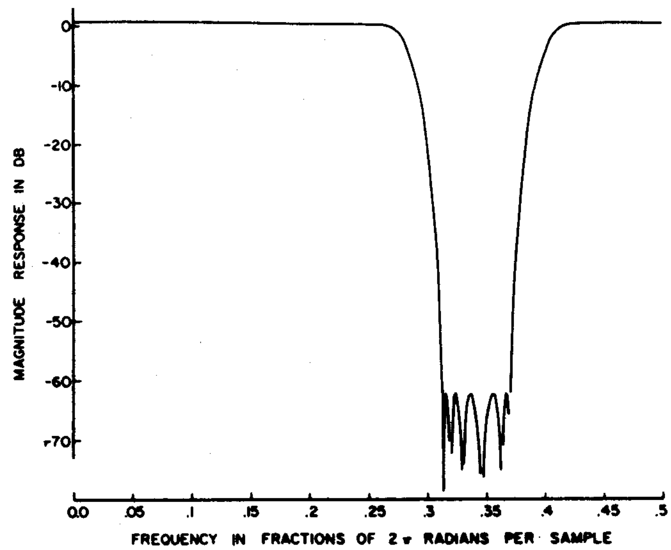


Fig. 4. Magnitude frequency response of the monotone passband design for Example 2.

For comparison, the program of [9] took 1.35 s to produce the same equiripple design as that shown in Fig. 3.

## VII. Comments on the Program

The Fortran source is shown in Fig. 5. The program has no subroutines (besides the user-supplied EXIT and DRAW) and consists of an initialization segment (up to the comment "MAIN LOOP"), the main simplex algorithm loop (from there

```
C
C        INPUT DATA
C        CARD 1: NFILT= FILTER LENGTH, NFFT=GRID SIZE PARAMETER,
C             NORMALLY 16*NFILT, NBANDS=NO. OF BANDS, KMAX= MAX. NO.
C             OF PIVOTS, IEX IS EXPONENT OF EPS ( EPS=10**IEX), IPRINT
C             =1 IF FULL PRINTOUT IS DESIRED
C        CARD 2: BANDEDGES IN FRACTIONS OF 2*PI
C        CARD 3: DESIRED VALUES IN BANDS
C        CARD 4: DESIRED TOLERANCES IN BANDS
C        CARD 5: MONOTONICITY PARAMETERS OF BANDS: +1 IF MONO.
C             DECREASING, -1 IF MONO. INCREAS., 0 IF NO CONSTRAINT
C
C        ARRAY INFORMATION
C        PRESENT DIMS. ARE FOR NFILT=129, NFFT=2048
C        CARRY IS (MM+2)X(MM+2), WHERE MM=(NFILT+1)/2, NO. OF COEFFS.
C        ABAR IS (NFILT+1)-DIMENSIONAL
C        H,COLK ARE MM-DIMENSIONAL
C        C,D,KSTAR ARE NN-DIM., WHERE NN=3*N2=NO. OF COLS.
C        TOL IS N2-DIMENSIONAL, WHERE N2=N/2
C        QQ,SLOPE ARE N2-DIM.: TABC,TABS ARE NFFT-DIM.
C        BASIS IS M-DIMENSIONAL
C        D IS FIRST THE COST IN PHASE 1, THEN PHASE 2
C
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL PRINT,KSTAR(3075)
      INTEGER PHASE,BASIS(66),IFLAT(10)
      REAL*8 CARRY(67,67),ABAR(130),H(65),COLK(65),C(3075),D(3075)
      REAL*8 EDGE(20),FX(10),TOLBAN(10),TOL(1025)
      REAL*8 QQ(1025),TABC(2048),TABS(2048)
      REAL*8 SLOPE(1025)
C
C        NFILT=FILTER LENGTH (ODD)
C        NFFT= POWER OF 2 = 2*(N2-1), N2= # OF GRID POINTS FROM 0 TO PI
C
      HUGE=1.D32
      READ(5,700)NFILT,NFFT,NBANDS,KMAX,IEX,IPRINT
  700 FORMAT(10I5)
      EPS=10.**(IEX)
      PRINT=.FALSE.
      IF(IPRINT.EQ.1)PRINT=.TRUE.
      WRITE(6,705)
  705 FORMAT(1H1,70(1H*)//25X,'FINITE IMPULSE RESPONSE (FIR)'/
     125X,'LINEAR PHASE DIGITAL FILTER DESIGN'/
     225X,'TWO-PHASE LINEAR PROGRAMMING ALGORITHM'/)
      WRITE(6,706)NFILT,NFFT,IEX
  706 FORMAT(25X,'FILTER LENGTH=',I5,' NFFT=',I5,' EPS=10**',I4)
      IF(PRINT)WRITE(6,901)
      IF(.NOT.PRINT)WRITE(6,902)
  901 FORMAT(25X,'THE PRINT OPTION IS ON')
  902 FORMAT(25X,'THE PRINT OPTION IS OFF')
      WRITE(6,707)
  707 FORMAT(//,' ',70(1H*))
      JB=2*NBANDS
      READ(5,701)(EDGE(J),J=1,JB)
  701 FORMAT(8F10.5)
      READ(5,701)(FX(J),J=1,NBANDS)
      READ(5,701)(TOLBAN(J),J=1,NBANDS)
C        IFLAT=1 IF MONOTONICALLY DECREASING, -1 IF INCREASING
      READ(5,700)(IFLAT(J),J=1,NBANDS)
C
C        MM=# OF COEFFS., STARTING WITH SAMPLE 0
C
      MM=(NFILT+1)/2
      M=MM+1
      MP=M+1
      N=NFFT+2
      N2=N/2
      N2P=N/2+1
      NP=N+1
      NN=3*N2
      TPON=8.D0*DATAN2(1.D0,1.D0)/FLOAT(NFFT)
C
C        INITIALIZE CONSTANTS IN TABLES
C
      DO 300 I=1,NFFT
  300 TABC(I)=DCOS(FLOAT(I-1)*TPON)
      DO 311 I=1,NFFT
  311 TABS(I)=DSIN(FLOAT(I-1)*TPON)
C
C        DEFAULT DESIRED VALUE IS 0
C
      DO 702 J=1,NN
  702 C(J)=0.D0
C
C        DEFAULT TOLERANCE IS 8.D0
C
      DO 703 J=1,N2
  703 TOL(J)=8.D0
C
C        TURN OFF ALL COLUMNS TO START
C
      DO 401 K=1,NN
  401 KSTAR(K)=.TRUE.
      DO 704 J=1,NBANDS
C
C        ROUND LOWER BAND EDGE DOWN, UPPER UP
C
      J1=IDINT(FLOAT(NFFT)*EDGE(2*J-1))+1
      J2=IDINT(FLOAT(NFFT)*EDGE(2*J)-10000.D0)+10001
      BE1=FLOAT(J1-1)/FLOAT(NFFT)
      BE2=FLOAT(J2-1)/FLOAT(NFFT)
      WRITE(6,900)J1,J2,BE1,BE2
  900 FORMAT(' BANDEDGES ARE IN COLUMNS',2I5,' CORRESP. TO',2D20.10)
      DO 704 K=J1,J2
      IF(IFLAT(J).EQ.0)GOTO315
      KSTAR(K+N)=.FALSE.
      SLOPE(K)=FLOAT(IFLAT(J))
  315 CONTINUE
      TOL(K)=TOLBAN(J)
      KSTAR(K)=.FALSE.
      KSTAR(K+N2)=.FALSE.
      C(K)=FX(J)
  704 C(K+N2)=-C(K)
C
C        SET UP CARRY, D FOR PHASE 1
C
      DO 312 J=1,N2
      IF(KSTAR(J).AND.KSTAR(J+N2))GOTO325
      SUM=0.D0
      JM=J-1
      LL=1
      DO 307 I=1,MM
      SUM=SUM+TABC(LL)
      LL=LL+JM
  307 IF(LL.GT.NFFT)LL=LL-NFFT
      D(J)=-TOL(J)-SUM
      D(J+N2)=-TOL(J)+SUM
  325 IF(KSTAR(J+N))GOTO312
      SUM=0.D0
      JM=J-1
      LL=1
      DO 313 I=1,MM
      SUM=SUM-FLOAT(I-1)*TABS(LL)
      LL=LL+JM
  313 IF(LL.GT.NFFT)LL=LL-NFFT
      D(J+N)=-SUM*SLOPE(J)
  312 CONTINUE
      DO 25 I=1,MP
      DO 25 J=1,MP
   25 CARRY(I,J)=0.D0
      CARRY(1,1)=-1.D0
      CARRY(MP,1)=1.D0
      DO 15 I=2,MP
   15 CARRY(I,I)=1.D0
C
C        USE NEGATIVE INDICES FOR ARTIFICIAL VARIABLES
C
      DO 30 I=1,M
   30 BASIS(I)=-I
C
C
C        MAIN LOOP
C
      PHASE=1
      KOUNT=0
    1 KOUNT=KOUNT+1
  510 IF(KOUNT.GT.KMAX)GOTO750
C
C        PRICING OPERATION
C
      DO 600 I=2,M
  600 H(I-1)=CARRY(1,I)
      CMIN=HUGE
      R= CARRY(1,MP)
      DO 316 J=1,N2
      S=R*TOL(J)
      IF(KSTAR(J))GOTO601
      Q=0.D0
      JM=J-1
      LL=1
      DO 301 I=1,MM
      Q=Q+H(I)*TABC(LL)
      LL=LL+JM
  301 IF(LL.GT.NFFT)LL=LL-NFFT
      CBAR=D(J)+(Q    +S)
      IF(CBAR.GE.CMIN)GOTO601
      K=J
      CMIN=CBAR
  601 IF(KSTAR(J+N2))GOTO602
      IF(.NOT.KSTAR(J))GOTO302
      Q=0.D0
      JM=J-1
      LL=1
      DO 303 I=1,MM
      Q=Q+H(I)*TABC(LL)
      LL=LL+JM
  303 IF(LL.GT.NFFT)LL=LL-NFFT
  302 CONTINUE
      CBAR=D(J+N2)+(-Q    +S)
      IF(CBAR.GE.CMIN)GOTO602
      K=J+N2
      CMIN=CBAR
  602 CONTINUE
      IF(KSTAR(J+N))GOTO316
      Q=0.D0
      JM=J-1
      LL=1
      DO 317 I=1,MM
      Q=Q-H(I)*FLOAT(I-1)*TABS(LL)
      LL=LL+JM
  317 IF(LL.GT.NFFT)LL=LL-NFFT
      CBAR=D(J+N)+SLOPE(J)*Q
      IF(CBAR.GE.CMIN)GOTO316
      K=J+N
      CMIN=CBAR
  316 CONTINUE
      CBAR=CMIN
      IF(CBAR.LT.-EPS)GOTO56
C
C        OPTIMUM FOUND
C
      IF(PHASE.EQ.1)GOTO6
      GOTO7
C
C        COLUMN K FOUND, GENERATE COLUMN K
C
   56 IF(K.GT.N)GOTO318
C
C        FIRST N COLS.
C
      KP=K
      IF(K.GT.N2)KP=K-N2
      JM=KP-1
      LL=1
      DO 308 I=1,MM
      COLK(I)=TABC(LL)
      LL=LL+JM
```

Fig. 5. The Fortran source program for linear programming design with monotonicity constraints.

```
  308 IF(LL.GT.NFFT)LL=LL-NFFT
      ABAR(1)=CBAR
      DO 66 I=2,MP
      SUM=0.D0
      DO 65 J=2,M
   65 SUM=SUM+CARRY(I,J)*COLK(J-1)
      IF(K.LE.N2)ABAR(I)=SUM+TOL(K)*CARRY(I,MP)
      IF(K.GT.N2)ABAR(I)=-SUM+TOL(K-N2)*CARRY(I,MP)
   66 CONTINUE
      GOTO319
C
C        LAST N2 COLS.
C
  318 KP=K-N
      JM=KP-1
      LL=1
      DO 320 I=1,MM
      COLK(I)=-FLOAT(I-1)*TABS(LL)*SLOPE(KP)
      LL=LL+JM
  320 IF(LL.GT.NFFT)LL=LL-NFFT
      ABAR(1)=CBAR
      DO 321 I=2,MP
      SUM=0.D0
      DO 322 J=2,M
  322 SUM=SUM+CARRY(I,J)*COLK(J-1)
  321 ABAR(I)=SUM
C
C        LOOK FOR ROW

  319 RMIN=HUGE
      L=0
      DO 70 I=2,MP
      IF(ABAR(I).LE.EPS)GOTO70
      RATIO=CARRY(I,1)/ABAR(I)
      IF(RATIO.GT.RMIN)GOTO70
      IF(RATIO.LT.RMIN)GOTO69
      IF(ABAR(I).LE.ABAR(L))GOTO70
   69 RMIN=RATIO
      L=I
   70 CONTINUE
C
C        TEST FOR UNBOUNDED SOLUTION
C
      IF(L.EQ.0)GOTO751
C
C        PIVOT
C
      LM=L-1
      KOLD=BASIS(LM)
      IF(KOLD.GT.0)KSTAR(KOLD)=.FALSE.
      BASIS(LM)=K
      KSTAR(K)=.TRUE.
      PIVOT=ABAR(L)
      DO 80 J=1,MP
   80 CARRY(L,J)=CARRY(L,J)/PIVOT
      DO 85 I=1,MP
      IF(I.EQ.L)GOTO85
      P=ABAR(I)
      DO 90 J=1,MP
   90 CARRY(I,J)=CARRY(I,J)-CARRY(L,J)*P
   85 CONTINUE
      Z=-CARRY(1,1)
      IF(PRINT)WRITE(6,400)KOUNT,LM,KOLD,K,CBAR,PIVOT,Z
  400 FORMAT(' ',I5,' PIVOT',I5,' CHANGED',I5,' OUT',I5,' IN CBAR=',
     1D20.10,' PIV=',D20.10,' Z=',D20.10)
      GOTO1
C
C        END OF MAIN LOOP; EXITS FOLLOW
C
C        KOUNT EXCEEDS KMAX
C
  750 WRITE(6,800)
  800 FORMAT(' ***** MAXIMUM NO. OF PIVOTS EXCEEDED *****')
      CALL EXIT
C
C        PHASE 1 OPTIMUM
C
    6 IF(Z.LE.EPS)GOTO8
      WRITE(6,108)Z
  108 FORMAT(' PHASE 1 ENDS WITH Z =',D14.7,':PROBLEM INFEASIBLE')
      CALL EXIT
    8 WRITE(6,106)Z
  106 FORMAT(' PHASE 1 SUCCESSFULLY COMPLETED; Z=',D14.7)
      PHASE=2
C
C        CHANGE ROW 1 OF CARRY FOR PHASE 2
C
      DO 95 J=1,NN
   95 D(J)=C(J)
      DO 96 J=1,MP
      CARRY(1,J)=0.D0
      DO 96 I=2,MP
      L=BASIS(I-1)
      IF(L.LE.0)GOTO96
      CARRY(1,J)=CARRY(1,J)-C(L)*CARRY(I,J)
   96 CONTINUE
      GOTO1
C
C        UNBOUNDED SOLUTION
C
  751 WRITE(6,111)PHASE
  111 FORMAT(' ***** UNBOUNDED SOLUTION FOUND IN PHASE ',I2,' *****')
      CALL EXIT
C
C        DONE
C
    7 WRITE(6,112)
  112 FORMAT(' OPTIMUM SOLUTION FOUND DURING PHASE 2')
C
C        FINAL OUTPUT SECTION
C
    2 WRITE(6,110)(BASIS(I),I=1,M)
  110 FORMAT(' FINAL BASIS IS'//16(' ',I4))
      DO 500 I=2,M
  500 H(I-1)=-CARRY(1,I)
      WRITE(6,109)(  I,H(I) ,I=1,MM)
  109 FORMAT(' COEFFICIENTS'//4(' ',I5,D20.10))
      DO 306 J=1,N2
      Q=0.D0
      JM=J-1
      LL=1
      DO 305 I=1,MM
      Q=Q+H(I)*TABC(LL)
      LL=LL+JM
  305 IF(LL.GT.NFFT)LL=LL-NFFT
  306 QQ(J)=Q
      IF(PRINT)WRITE(6,98)(J,QQ(J),J=1,N2)
   98 FORMAT(' FREQUENCY RESPONSE'//4(' ',I5,D20.10))
      DO 801 J=1,NBANDS
      J1=IDINT(FLOAT(NFFT  )*EDGE(2*J-1))+1
      J2=IDINT(FLOAT(NFFT  )*EDGE(2*J)-10000.D0)+10001
      FDES=FX(J)
      ERRMAX=0.D0
      DO 803 I=J1,J2
      ERROR=DABS(QQ(I)-FDES)
      IF(ERROR.LE.ERRMAX)GOTO803
      ERRMAX=ERROR
  803 CONTINUE
      DEVDB=20.D0*DLOG10(ERRMAX)
      WRITE(6,807)J,EDGE(2*J-1),EDGE(2*J),FX(J),TOLBAN(J)
  807 FORMAT('0BAND',I3,' EDGE1=',D20.10,' EDGE2=',D20.10,' FX=',D20.10,
     1' TOLBAND=',D20.10)
      WRITE(6,804)ERRMAX,DEVDB
  804 FORMAT(' MAX. DEV.=',D20.10,' MAX. DEV. IN DB =',D20.10)
  801 CONTINUE
C
C        DRAW IS A USER-SUPPLIED PLOT ROUTINE
C
      CALL DRAW(QQ,N2,NFFT,NFILT)
      STOP
      END
```

Fig. 5. (Continued.)

to "END OF MAIN LOOP; EXITS FOLLOW"), error exists (from there to "DONE"), and the output section (from there to the END). The user should provide a subroutine EXIT for terminations other than the normal STOP, and may supply a plotting subroutine DRAW, or remove its call.

The program has always been run in double precision, and this is probably necessary for any reasonably sized problem. More sophisticated measures to control roundoff error in linear programs, such as periodic basis reinversion and replacement of small quantities by zero, were not found necessary for the design of filters of lengths up to 127, but may be necessary for larger problems.

## VIII. Conclusions

We have shown that the linear programming formulation can be used to impose monotonicity constraints in the design of FIR digital filters, and provides the designer with much more flexibility than the usual Remez algorithms. It is hoped that the inclusion of the program will encourage others to experiment further with the flexibility of this design approach.

## Acknowledgment

The author would like to thank Dr. J. W. Cooley, Dr. H. D. Helms, and Dr. J. F. Kaiser for their helpful discussions concerning the topics of this paper.

## References

[1] H. D. Helms, "Digital filters with equiripple or minimax responses," *IEEE Trans. Audio Electroacoust.*, vol. AU-19, pp. 87-93, Mar. 1971.

[2] L. R. Rabiner, "Linear programming design of finite impulse response (FIR) digital filters," *IEEE Trans. Audio Electroacoust.*, vol. AU-20, pp. 280-288, Oct. 1972.

[3] D. W. Tufts, D. W. Rorabacher, and W. E. Mosier, "Designing simple, effective digital filters," *IEEE Trans. Audio Electroacoust.*, vol. AU-18, pp. 142-158, June 1970.

[4] R. K. Cavin, III, C. H. Ray, and V. T. Rhyne, "The design of

optimal convolutional filters via linear programming," *IEEE Trans. Geosci. Electron.*, vol. GE-7, pp. 142-145, July 1969.

[5] O. Herrmann, "On the approximation problem in nonrecursive digital filter design," *IEEE Trans. Circuit Theory*, vol. CT-18, pp. 411-413, May 1971. (Reprinted in *Digital Signal Processing*, L. R. Rabiner and C. M. Rader, Eds. New York, NY: IEEE Press, 1972.)

[6] J. F. Kaiser, "Design subroutine (MXFLAT) for symmetric FIR low-pass digital filters with maximally-flat pass- and stopbands," to appear.

[7] J. F. Kaiser and W. A. Reed, "Data smoothing using lowpass digital filters," *Rev. Sci. Instrum.*, vol. 48, pp. 1447-1457, Nov. 1977.

[8] S. Darlington, "Filters with Chebyshev stopbands, flat passbands, and impulse responses of finite duration," in *Proc. 1978 IEEE Int. Symp. Circuits Syst.*, New York, NY, pp. 40-44.

[9] J. H. McClellan, T. W. Parks, and L. R. Rabiner, "A computer program for designing optimum FIR linear phase digital filters," *IEEE Trans. Audio Electroacoust.*, vol. AU-21, pp. 506-526, Dec. 1973.

[10] M. Simonnard, *Linear Programming* (translated from the French by W. S. Jewell). Englewood Cliffs, NJ: Prentice-Hall, 1966.

[11] I. Barrodale and A. Young, "Algorithms for best $L_1$ and $L_\infty$ linear approximation on a discrete set," *Numerische Mathematik*, vol. 8, pp. 295-306, 1966.

[12] J. W. Cooley, P. A. W. Lewis, and P. D. Welch, "The finite Fourier transform," *IEEE Trans. Audio Electroacoust.*, vol. AU-17, pp. 77-85, June 1969. (Reprinted in *Digital Signal Processing*, L. R. Rabiner and C. M. Rader, Eds. New York, NY: IEEE Press, 1972.)

[13] K. Steiglitz, "Optimal design of digital Hilbert transformers with a concavity constraint," in *Proc. 1979 IEEE Int. Conf. Acoust., Speech, Signal Processing*, Washington, DC, Apr. 2-4, 1979, pp. 824-827.

[14] H. D. Helms has suggested that a chirp transform taking $0(P \log M)$ time may be useful here; personal communication.