A Computer Program for Designing Digital Elliptic Filters

AUGUSTINE H. GRAY, JR., MEMBER, IEEE, AND JOHN D. MARKEI, MEMBER, IEEE

Abstract—A computer program is presented for designing digital elliptic filters of the four most common types: low-pass, high-pass, bandpass, and band-stop. The program is presented in Fortran IV. The various numerical algorithms required for implementing the elliptic functions are referenced in the program and are obtained from readily available tables of mathematical functions. Examples for each filter type are presented.

I. INTRODUCTION

Elliptic filters1 have been widely used to achieve restrictive frequency domain requirements. They can simultaneously provide small passband ripple, large stopband attenuation, and small transition bandwidths, with equal ripple characteristics in both the passband and the stopband. Based on the early work of Cauer [1], Norton [2], and Darlington [3], a large amount of information is already in print. For example, the work of Grossman [4] is based upon Darlington’s early notes. Textbooks such as Guillemin [5], Storer [6], and Gold and Rader [7] devote considerable space to elliptic filters. Daniels [8] presents a detailed discussion of theoretical issues and approximation techniques.

For the design of elliptic filters, there are tables such as in Zverev [9], Christian and Eisenmann [10], and Saal [11], monographs such as in Henderson [12] and Szentirmai [13], and design charts such as illustrated in Rabiner and Gold [14]. Many of the algorithms needed can be found in computer form such as in [8], but a complete user-oriented computer program for the design of digital elliptic filters does not appear to exist in the currently available literature.

A computer program for the design of finite-duration impulse response (FIR) linear phase filters with equal ripple in the passband and stopband has been presented by McClellan, Parks, and Rabiner [15]. If linear phase is not of major importance, considerably fewer coefficients are usually necessary with an infinite-duration impulse response (IIR) digital elliptic filter design to match the specifications of an FIR equal ripple filter [14]. In this paper, a computer program for the design of digital elliptic filters is presented. The program is written in Fortran IV and allows for the design of low-pass, high-pass, bandpass, and band-reject or bandstop filters. The program output provides computed filter specifications (based upon the input parameters) and the transfer function coefficients. Numerical algorithms with rapid convergence and high accuracy are chosen so that very restrictive conditions can be met as long as sufficient computer accuracy is used. All algorithms for the Jacobi elliptic functions are taken from Abramowitz and Stegun [16] and each algorithm is identified within the computer program. The digital elliptic filter program and its associated subroutines are presented in Appendix A. A brief description of the subroutines and an outline of the main program are presented in Appendix B.

II. PROTOTYPE FILTER

All filter designs are based upon a normalized low-pass elliptic filter of order \( N \) in the \( s \)-plane, as illustrated in Fig. 1,
in terms of the magnitude squared response $|T|^2$. The prototype filter is normalized to have a passband edge frequency of unity. The stopband edge frequency $\omega_s$ will depend upon the filter specifications to be described in the examples to follow. Using assorted mappings, the normalized low-pass filter can be transformed into various unnormalized filter types such as those described by Grossman [4] and Constantinides [17]. An equivalent approach is to use the bilinear and biquadratic transformations to obtain discrete low-pass and bandpass filters from the prototype low-pass filter in the $s$-plane. Replacing $s$ by $1/s$ interchanges the stop and passbands and results in the high-pass and band-reject or bandstop filters.

Filter design tables [9]-[11] generally use as parameters a reflection coefficient $\rho$ and an angle $\theta$ where, in terms of the variables of Fig. 1,

$$\rho = e^{i\sqrt{1 + \epsilon^2}}$$

and

$$\theta = \sin^{-1} \left( 1/\omega_s \right).$$

These parameters are also given as outputs by the computer program, with $\theta$ expressed in degrees.

### III. Input Parameters

The parameters entered into the program are the filter order, the passband edge frequency or frequencies, and a single stopband edge frequency (or the stopband attenuation). The way in which these parameters are entered determines the type of filter—low-pass, high-pass, bandpass, or bandstop. The program requests an input by typing the line

$$N, DBR, FS, F1, F2, F3 OR DBDOWN.$$  

The variable $N$ is the order of the filter in the $s$-plane. It will equal the order of the filter in the $s$-plane for the low-pass and high-pass filters. For bandpass and bandstop filters, the filter order in the $s$-plane will equal $2N$.

A minimum value for $N$ can be estimated using the approximation [6]

$$N \approx \frac{2}{\pi^2} \ln \left( \frac{4A}{\epsilon} \right) \ln \left[ \frac{8}{(\omega_s - 1)} \right]$$  

where $\omega_s$ is the $s$-plane prototype filter stopband edge as shown in Fig. 1. The input variable $DBR$ is the dB ripple in the passband, related to $\epsilon$ of Fig. 1 by

$$DBR = 10 \log_{10} \left( 1 + \epsilon^2 \right).$$  

The variable $FS$ is the sampling frequency, used to normalize the other frequency entries. The variables $F1$ and $F2$ define passband edge frequencies as described in the following sections. The variable $F3 OR DBDOWN$ defines one stopband edge frequency if it is positive or the stopband attenuation if it is negative. For example, entering 30 for $F3 OR DBDOWN$ implies a stopband edge frequency of 30 (in the same units as $FS$), while entering -30 for $F3 OR DBDOWN$ implies a stopband attenuation of 30 dB, i.e.,

$$20 \log_{10} (A) = 30.$$  

### V. Low-Pass Filters

A low-pass filter is identified by $F1$ being either zero or negative. $F2$ then represents the passband edge frequency, which is mapped into $\omega_s = 1$ or $s = j$ in the $s$-plane by using the bilinear transformation

$$s = \frac{1}{c} \left( \frac{z - 1}{z + 1} \right)$$  

where

$$c = \tan \left( \frac{\pi F2}{FS} \right).$$

$F3 OR DBDOWN$ is the stopband edge frequency or the dB attenuation in the stopband for $F3 OR DBDOWN$ positive or negative, respectively.

For $F3 OR DBDOWN > 0$, the prototype stopband edge frequency is computed from (3) as

$$\omega_s = \tan \left( \frac{\pi F3}{FS} \right) \tan \left( \frac{\pi F2}{FS} \right).$$  

As a first example, the filter used by Gold and Rader [7] is illustrated where the original specifications were a minimum stopband attenuation of 60 dB, a maximum passband ripple of 0.5 dB, a sampling frequency of 10 kHz, a passband edge frequency of 2 kHz, and a stopband edge frequency of 2.2 kHz. Using (5) and the approximation (1), a minimum of 7.8 is found [7]. Since $N$ must then be at least 8, one or more of the requirements can be improved upon. The approach outlined by Gold and Rader improves the requirements on both the stopband attenuation and the passband ripple. The program presented here can improve on either the stopband attenuation or the stopband edge frequency, as illustrated by the two examples to follow.

Fig. 2 shows the program input and output for the specifications $N = 8$, $FS = 10$, $F1 = 0$, $F2 = 2$, $F3 = 2.2$, while Fig. 3 shows the filter response. Fig. 4 shows the program input and output for the specifications $N = 8$, $FS = 10$, $F1 = 10$, $F1 = 0$, $F2 = 2$, and $F3 OR DBDOWN = -60$. The corresponding filter response is not shown since differences between the two cases are visually negligible. The only difference in the input data for the two cases lies in the entry $F3 OR DBDOWN$.

The printouts show the input data, the prototype filter
LP FILTER, 0.3800000+00 DB RIPPLE
PASS BAND EDGE 2.000000+01
STOP BAND EDGE 2.200000+01
SAMPLE FREQUENCY 8.100000+02
THETA= 0.624839+02, RHO= 0.329771+00
-0.690000+02 DB DOWN
\theta = ORDER
P(j)
A(j)
1 0.10880000+01 0.21680000+01
2 -0.35390000+01 0.51560000+01
3 0.713428+01 0.385604+01
4 -0.962960+01 0.445104+01
5 0.186313+01 0.658363+00
6 -0.738765+00 0.454104+01
7 0.387815+00 0.385604+01
8 -0.195114+01 0.154825+01
9 0.240371+00 0.185363+00
POLES QUADRATIC FACTORS
-0.3908238+00 -0.5375618+00 0.864874+00
0.3436233+00 -0.857146+00 0.861491+00
0.4548583+00 -0.857146+00 0.684326+00
0.5396718+00 -0.115145+00 0.425798+00
ZEROS QUADRATIC FACTORS
0.1788625+00 0.3577258+00 0.10000000+00
0.9421333+00 -0.1856258+00 0.10000000+00
-0.791514+00 0.3578258+00 0.10000000+00
-0.899536+00 0.816875+00 0.10000000+00
GAIN TERM FOR CASCADE FORM P(j) = 0.894470+00

Fig. 2. A low-pass filter example with stopband edge frequency specified.

Fig. 3. Filter response for the low-pass filter example of Fig. 2.

parameters \( \theta \) and \( \rho \) as \( \text{THETA} \) and \( \text{RHO} \), followed by a listing of the filter parameters. First, for use in direct form implementation or as a starting point in the design of orthogonal polynomial filter structures [18], [19], the denominator and numerator coefficients are listed for the transfer function \( T(z) \) where

\[
T(z) = \sum_{i=1}^{M+1} P(i) z^{-i} + \sum_{i=1}^{M+1} A(i) z^{i-1}
\]

For example, from Fig. 4, the transfer function with rounded coefficients is of the form

\[
T(z) = 0.0122 + 0.0159 z^{-1} + \cdots + 0.0122 z^{-8} + 1. - 3.36 z^{-1} + \cdots + 0.241 z^{-8}
\]

Next the real and imaginary parts of the complex \( z \)-plane poles and zeros and their quadratic forms for cascade implementation are printed out. Only those poles and zeros on or above the real axis are printed, for they occur in complex conjugate pairs. Thus, for examples, the first line for POLES and QUADRATIC FORMS in Fig. 4 indicates a complex pole pair at 0.3009 + 0.9356 through the printout of the real and imaginary parts as \( p = 0.3009 + j 0.9356 \), and a quadratic form

\[
(1 - pz^{-1})(1 - p^*z^{-1}) = 1 - 2 \text{ Real}(p) z^{-1} + |p|^2 z^{-2}
\]

For the case of real zeros on the unit circle (at \( z = \pm 1 \)), the quadratic form results are expressed in a degenerate linear manner, as can be seen in the example of Fig. 5 where there is a zero at \( z = \pm 1 \), resulting in the quadratic form

\[
1 - z^{-1} + z^{-2} = 1 - z^{-1}
\]

Following the printout of the poles, zeros, and their quadratic form, the gain term for a cascade implementation \( P(1) \) is printed.

If the filter is to be implemented in a parallel form, the user must supply his own subroutine to evaluate the residues.
VI. HIGH-PASS FILTERS

High-pass filters are identified by an entry $F_2$ which is equal to or larger than the folding frequency $F_S/2$. $F_1$ represents the passband edge frequency. The entry $F_3$ or DBDOWN defines the stopband edge frequency, if positive, or the dB attenuation, if negative. The mapping for this case also uses the bilinear transformation of (3), but with $s$ replaced by $1/s$ in the $s$-plane.

A fifth-order high-pass filter design example is shown in Figs. 5 and 6. Other input parameters are a sampling frequency of 6.5 kHz, a bandpass ripple of 0.3 dB, a bandpass edge frequency of 2 kHz, and a stopband attenuation of 40 dB. The interpretation of the output is identical to that discussed for the low-pass filter.

As the passband edge frequency is greater than one quarter of the sampling frequency, all of the poles have negative real parts.

VII. BANDPASS AND BANDSTOP (BAND-REJECT) FILTERS

The bandpass digital elliptic filter is obtained by using the biquadratic transformation [7, p. 761 to get to the prototype $s$-plane filter,

$$s = \frac{c z^2 - 2 z \cos(\gamma) + 1}{z^2 - 1}. \quad (6)$$

The bandstop filters are found by replacing $s$ by $1/s$ in (6). The constants $c$ and $\gamma$ are chosen to map passband edge frequencies $F_1$ and $F_2$ into $s = \pm j$ or $\omega = \pm 1$. For the bandpass filter, this gives

$$c = \tan(\Delta \theta/2) \quad (7a)$$

and

$$\cos(\gamma) = \cos(\bar{\theta}/\cos(\Delta \theta/2) \quad (7b)$$

where $\bar{\theta}$ and $\Delta \theta$ are the normalized center frequency and bandwidths

$$\bar{\theta} = (F_1 + F_2) n/F_S \quad (8a)$$

and

$$\Delta \theta = 2n(F_2 - F_1)/F_S. \quad (8b)$$

If the input data are entered with $0 < F_1 < F_2 < F_S/2$, a bandpass filter is defined. If the input data are entered with $0 < F_2 < F_1 < F_S/2$, a bandstop filter is defined. Since $N$ is the order of the filter in the $s$-plane, $2N$ is the order in the $z$-plane. The final data entry is $F_3$ or DBDOWN.

Although relatively low-order and nonstringent elliptic filter design examples have been presented here for simplicity, care has been exercised in the choice of numerical algorithms to assure that the complexity or stringency of the design is limited only by the computer accuracy used.
GRAY AND MARKEL: COMPUTER PROGRAM FOR DESIGNING DIGITAL ELLIPTIC FILTERS

Fig. 10. Frequency response of the band-reject filter design example.

APPENDIX A

COMPUTER PROGRAM

DOUBLE PRECISION DIGITAL ELLIPTIC FILTER DESIGN PROGRAM

BY

A. H. GRAY, JR. AND J. D. MARKEL

SPEECH COMMUNICATIONS RESEARCH LABORATORY

833A MIRAMONTE DRIVE

SANTA BARBARA, CA. 93109

LOADING OF DATA -- MAIN PROGRAM

INPUTS:

N= ORDER OF FILTER (IN S-PLANE)

DDB= PASS BAND RIPPLE IN DB

F* = SAMPLE FREQUENCY

F1,F2= PASS BAND EDGES (SEE OPTIONS)

F3 = STOP BAND EDGE OR DISSON (SEE OPTIONS)

OPTIONS

IN ALL CASES, IF A NEGATIVE VALUE FOR F3 IS ENTERED, IT IS TREATED AS THOUGH IT IS THE STOP BAND ATTENUATION IN DB. THE DESCRIPTORS BELOW ARE FOR POSITIVE ENTRIES.

LOW PASS---- ENTER F1 AS LESS THAN OR EQUAL TO ZERO.

HIGH PASS---- ENTER F2 AS GREATER THAN OR EQUAL TO F1/F3.

BAND PASS---- ENTER F1 AND F2 AS PASS BAND EDGES.

BAND REJECT---- ENTER F1 AS LESS THAN OR EQUAL TO F2 AND F3 AS STOP BAND EDGE.

FOR SINGLE PRECISION, REMOVE THE DOUBLE PRECISION STATEMENTS, BEING SURE TO PUT IN DIMENSION STATEMENTS FOR DIFFERENT PRECISIONS.

LOADING OF DATA: CHANGE O FORMATS FOR SINGLE PRECISION, REMOVE THE DCDOUBLE PRECISION STATEMENTS. BEING SURE TO PUT IN DIMENSION STATEMENTS FOR DIFFERENT PRECISIONS.

FOR STANDARD FORMAT STATEMENT ON THE INPUT TO BE READ, THIS SHOULD BE ALTERED ON MOST MACHINES.

IN S-PLANE ALL FILTERS HAVE PASS BAND EDGE AT WR/2. STOP BAND EDGE AT WR.

CASE 1- LOW PASS FILTER

CASE 2- BAND PASS FILTER

CASE 3- HIGH PASS FILTER

CASE 4- BAND REJECT FILTER

IN S-PLANE ALL FILTERS HAVE PASS BAND EDGE AT WR/2. STOP BAND EDGE AT WR.

CASE 1- LOW PASS FILTER

CASE 2- BAND PASS FILTER

CASE 3- HIGH PASS FILTER

CASE 4- BAND REJECT FILTER

FOR DIMENSIONED VARIABLES, CHANGE O FORMATS FOR SINGLE PRECISION, REMOVE THE DCDOUBLE PRECISION STATEMENTS. BEING SURE TO PUT IN DIMENSION STATEMENTS FOR DIFFERENT PRECISIONS.

READ THE FOLLOWING STATEMENTS TO THE MACHINE.

DO 138 J=1,2
DOUBLE PRECISION THETA, U, U0, V, W, C, W, R, X, X0, Y, Z.

WRITE(IP, 1651) FS

T-TEST

COMMON ALPHA, ALPHAT, ARITHMEIC-GEOMETRIC MEAN, ALGORITHM FOR
CALCULATION OF ELLIPTIC FUNCTIONS. TAKEN FROM ABRAHAM J. WAD.

SINGLE PRECISION X, X0, Y, Z.

DOUBLE PRECISION EPS0 = 0.00001.

CALL SCN(X, Y, Z, Z0, IP, JP).

DO 260  LPI = 1, NP

DO 250  LPI = 1, NZ

DO 240  LPI = 1, NZ

DO 230  LPI = 1, NZ

DO 220  LPI = 1, NZ

DO 210  LPI = 1, NZ

DO 200  LPI = 1, NZ

DO 190  LPI = 1, NZ

DO 180  LPI = 1, NZ

DO 170  LPI = 1, NZ

DO 160  LPI = 1, NZ

DO 150  LPI = 1, NZ

DO 140  LPI = 1, NZ

DO 130  LPI = 1, NZ

DO 120  LPI = 1, NZ

DO 110  LPI = 1, NZ

DO 100  LPI = 1, NZ

DO 90  LPI = 1, NZ

DO 80  LPI = 1, NZ

DO 70  LPI = 1, NZ

DO 60  LPI = 1, NZ

DO 50  LPI = 1, NZ

DO 40  LPI = 1, NZ

DO 30  LPI = 1, NZ

DO 20  LPI = 1, NZ

DO 10  LPI = 1, NZ

DO 0  LPI = 1, NZ

WRITE(IP, 157) T-TEST.

RETURN
The computer program of Appendix A implements the necessary transformations between the z-plane and s-plane, designs a prototype filter in the s-plane, and gives the output transfer function parameters in the z-plane. The necessary equations for the prototype design are summarized in [6] and [7] with only minor differences in notation between the two references. Algorithms for evaluation of all of the necessary elliptic functions are taken from 1161 and specific references to sections and equations are presented in the program.

The program is portable to the extent that at most three statements should have to be changed in order to run on most Fortran machines. The read and write device numbers are specified in lines 520 and 530, with IRD being the read device and IP being the write device. The input data is read with FORMAT statement number 35, line 640, which presently is a free format not standard on many machines.

Rather than give a flow chart for the functioning of the program, we present an outline description since there are few conditional branching statements other than those used to determine the filter type from the input data. This outline refers to the lines numbers printed with the program. The main program uses a set of 11 subroutines or functions given by

```
APPENDIX B

The computer program of Appendix A implements the necessary transformations between the z-plane and s-plane, designs a prototype filter in the s-plane, and gives the output transfer function parameters in the z-plane. The necessary equations for the prototype design are summarized in [6] and [7] with only minor differences in notation between the two references. Algorithms for evaluation of all of the necessary elliptic functions are taken from 1161 and specific references to sections and equations are presented in the program.

The program is portable to the extent that at most three statements should have to be changed in order to run on most Fortran machines. The read and write device numbers are specified in lines 520 and 530, with IRD being the read device and IP being the write device. The input data is read with FORMAT statement number 35, line 640, which presently is a free format not standard on many machines.

Rather than give a flow chart for the functioning of the program, we present an outline description since there are few conditional branching statements other than those used to determine the filter type from the input data. This outline refers to the lines numbers printed with the program. The main program uses a set of 11 subroutines or functions given by

```
subroutine ELLIP lines 01570-0120
function AKK lines 02160-02430
subroutine ASN lines 02440-02730
subroutine SCN lines 02740-02940
subroutine SING lines 02950-03390
function CAY lines 03300-03480
subroutine DBDOWN lines 03490-03790
subroutine ZPLN lines 03800-05590
function DISIN lines 05600-05700
function DICOS lines 05710-05790
subroutine QUAD lines 05800-05900
```

The function of each of these subroutines and functions is described in the outline.

Main Program (Lines 00010-01560)

The main program takes the input specifications for the filter and uses them to obtain the specifications for the prototype s-plane filter by first finding the appropriate mapping between the z-plane and the s-plane. There are no calculations involving the Jacobi elliptic functions that take place in the main program.

00010-00620 Comments and initialization.
00630-00640 Request for input data.
00650-01060 Reading of input data, comments, logic to test for type of filter, and logic to test for input data errors.
01070-01110 Generation of the parameters needed for the mapping as in (4) and (7).
01120-01150 Comments and test for input of dB attenuation in stopband rather than stopband edge. If the data entry for F3 OR DBDOWN is negative, then that entry is treated as stopband attenuation, and the subroutine DBDOWN is called.
01160-01530 The mapping parameters and filter type are used to obtain the prototype filter stopband edge \( \omega_s \). For the case of bandpass and bandstop filters, both stopband edges (in the z-plane) are evaluated for computer printout.
01540-01560 The subroutine ELLIP is called for filter design, and command returns to the request for input data to design a new filter.

Subroutine ELLIP

This subroutine, combined with the other subroutines and functions that it calls, carries out the design of the prototype filter in the s-plane.

01570-01880 Comments and evaluation of \( e \) of Fig. 1 and the reflection coefficient

\[
\rho = \frac{e}{\sqrt{1 + e^2}}.
\]

01890-01920 Evaluates the modulus \( k \) and the complete elliptic integrals \( K(k) \) and \( K'(k) = K(\sqrt{1 - k^2}) \) needed to be used with [6, eq. (30.21a)] or [7, eq. (3.27)]. The modulus
Function AKK

Subroutine SING

GRAY AND MARKEL: COMPUTER PROGRAM FOR DESIGNING DIGITAL ELLIPTIC FILTERS

Subroutine ASN

01930-01950 A variable SCALE is evaluated to be used later to insure that the peak magnitude filter response is one. In addition, [6, eq. (20.21a)] or [7, eq. (3.27)] is used to evaluate the ratio of \( K(k_1)/K'(k_1) \). The function CAY is then used to evaluate \( k_1 \) from the ratio where

\[
k_1 = e^{\sqrt{A^2 - 1}}. \tag{10}
\]

01960-02000 The modulus \( k_1 \) is used to evaluate the complete elliptic integral \( K'(k_1) \), called \( V \) in the program.

02010-02030 Equation (10) is used to give the value of \( A \), and thus the stopband attenuation, from the values of \( k_1 \) and \( e \).

02040-02110 After comments and assorted print statements, the variable \( u_0 \) of [6] and [7] is obtained for use in the evaluation of pole locations. This is done through the function ASN by using the unnumbered equation at the top of [6, p. 296] or [7, eq. (3.30)].

02120 Subroutine SING is called, which actually finds the locations of the poles and zeros in the \( s \)-plane for the prototype filter.

02130 Subroutine ZPLN is called to transform the poles and zeros from the \( s \)-plane to the \( z \)-plane, and to find the digital filter parameters.

02140-02150 Termination of the subroutine.

Function AKK

02160-02430 This function implements the arithmetic-geometric mean (AGM) table of [16] and obtains the complete elliptic integral. Specific algorithms from [16] are referenced in comment statements.

Subroutine ASN

02440-02730 This subroutine is used to evaluate the inverse elliptic sine function of an imaginary argument as needed for line number 02110. It is based upon [16] and uses the AGM table generated by function AKK.

Subroutine SCN

02740-02940 This subroutine uses the AGM table generated by function AKK and algorithms of [16] to evaluate the elliptic functions \( sn(\cdot), cn(\cdot), \) and \( dn(\cdot) \).

Subroutine SING

02950-03290 This subroutine evaluates the locations of

the poles and zeros in the \( s \)-plane for the prototype filter. It is based on [6, eq. (30.18) and (30.15)] or equivalently, [7, figs. 3.11 and 3.12 and eq. (3.29)].

Function CAY

03300-03480 This function evaluates a modulus, \( k \) or \( k_1 \), from a ratio of complete elliptic integrals \( K(k)/K'(k) \) or \( K(k_1)/K'(k_1) \). Its input is called a nome, \( Q \) defined by [16, eq. (17.3.17)] as

\[
Q = \exp \left[ -\pi K'(\cdot)/K(\cdot) \right]. \tag{11}
\]

Subroutine DBDOWN

03490-03790 This subroutine is used when the final data entry \( F3 \) or \( DBDOWN \) is negative, so that it is taken as a stopband attenuation. For ease of programming, this subroutine was designed to take that entry and obtain a value for \( F3 \), a discrete stopband edge frequency so that the remainder of the calculations carry on as though that \( F3 \) value had been entered. The stopband attenuation is used to give \( A \) of Fig. 1. The modulus \( k_1 \) is found from (11), and the elliptic integrals \( K(k_1) \) and \( K'(k_1) \) by using function AKK. Equation (30.21a) of [6] or [7, eq. (3.27)] are then used to give the ratio \( K(k)/K'(k) \), and the function CAY gives the modulus \( k \) and from (11) the prototype stopband edge frequency. Using the mapping parameters, this stopband edge frequency is transformed back to the \( z \)-plane to give the equivalent stopband edge frequency entry which would have produced the required stopband attenuation.

Subroutine ZPLN

03800-04720 The type of mapping and appropriate transformation is used to map the \( s \)-plane poles and zeros into the \( z \)-plane, and obtains the final transfer function.

04730-04850 Any \( s \)-plane zeros at infinity are mapped into the \( z \)-plane. The net result is a total of \( N \) poles and \( N \) zeros for low-pass and highpass filters and \( 2N \) poles and \( 2N \) zeros for bandpass and bandstop filters.

04860-05040 The poles and zeros are used to form numerator and denominator polynomials for the \( z \)-plane transfer function, each with unity leading coefficient, so they are not yet normalized.
Roundoff Noise in Digital Filters: Frequency Transformations and Invariants

CLIFFORD T. MULLIS, MEMBER, IEEE, AND RICHARD A. ROBERTS, MEMBER, IEEE

Abstract—The family of filters \( \{ H(F(z)) : F(z) \} \) a frequency transformation, generated from a prototype filter \( H(z) \) is shown to possess certain common properties. These are coordinate-free quantities (called second-order modes) which are invariant under frequency transformation. The invariance is significant in the design of low-noise fixed-point digital filter structures since these second-order modes characterize the minimum attainable noise. Filter structures (including parallel, cascade, and ladder configurations) are studied whose output noise is essentially independent of bandwidth and center frequency. An analysis of direct form structures (whether isolated or as one section within a cascade or parallel configuration) results in an expression giving the dominant term in the output noise as a function of the parameter in the low-pass-low-pass transformation. This noise term approaches infinity as bandwidth approaches zero. Thus, for narrowband filters, a difference of several orders of magnitude in the output noise can exist between a scaled direct form (having six multiplications per two-pole section) and the optimal form (having nine multiplications per two-pole section).

I. INTRODUCTION

A COMMONLY held opinion, based on experience, is that roundoff noise in fixed-point digital filters increases in severity as the width of the passband is made