ALGORITHMIC PROPERTIES OF THE BILINEAR TRANSFORM

by

DAVID NICHOLAS HEIN

B.S., Kansas State University, 1974

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1976

Approved by:

M. Ahmed

Major Professor

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

The bilinear transformation (BT) plays an important role in the design of infinite impulse response (IIR) filters. It enables one to readily design digital filters with characteristics that are quite similar to their analog counterparts. In addition, the BT can be used to good advantage to obtain digital computer simulations of analog systems, prior to their hardware implementation.

The main objective of this paper is to develop a general algorithm for computing the BT associated with a given transfer function H(s), which is in the form of a rational function; i.e.

$$H(s) = \frac{a_o + a_1 s + a_2 s^2 + \ldots + a_N s^N}{b_o + b_1 s + b_2 s^2 + \ldots + b_M s^M} \tag{1}$$

We wish to obtain the corresponding digital transfer function H(z), which is given by

$$\hat{H}(z) = H(s)\Big|_{s = \frac{z-1}{z+1}} \tag{2}$$

A previous attempt to obtain a general solution to the above problem met with little success [1]. Any attempt to consider transfer functions beyond the fourth or fifth order case resulted in unwieldly expressions for relating the coefficients of a given H(s) to those of the desired $\hat{H}(z)$.

In this paper, we develop a recurrence relation in the form of a linear partial difference equation, which readily enables one to evaluate $\hat{H}(z)$ in Eq. (2) for a given H(s) defined in Eq. (1). This recurrence relation yields an algorithm which is implemented in software using FORTRAN as the programming language. A closed-form solution for the corresponding linear partial difference equation is also obtained using a Z-transform approach.

II. DEVELOPMENT OF THE ALGORITHM

Substitution of $s = \frac{z-1}{z+1}$ in (1) results in

$$\hat{H}(z) = \frac{\sum_{n=0}^{N} a_n[(1-z^{-1})/(1+z^{-1})]^n}{\sum_{m=0}^{M} b_m[(1-z^{-1})/(1+z^{-1})]^m} \tag{3}$$

From (3) it follows that

$$\hat{H}(z) = \frac{\sum_{n=0}^{N} a_n[(1-z^{-1})/(1+z^{-1})]^n(1+z^{-1})^\alpha}{\sum_{m=0}^{M} b_m[(1-z^{-1})/(1+z^{-1})]^m(1+z^{-1})^\alpha} \tag{4}$$

where $\alpha = \max\{M,N\}$.

Rewriting (4) we have

$$\hat{H}(z) = \frac{\sum_{n=0}^{N} a_n(1-z^{-1})^n(1+z^{-1})^{\alpha-n}}{\sum_{m=0}^{M} b_m(1-z^{-1})^m(1+z^{-1})^{\alpha-m}} \tag{5}$$

Now, we restrict our attention to the terms $(1-z^{-1})^n(1+z^{-1})^{\alpha-n}$ and $(1-z^{-1})^m(1+z^{-1})^{\alpha-m}$ in (5). For the time being, we shall assume that these terms can be expressed in the form of a power series expansion as follows:

$$(1-z^{-1})^r(1+z^{-1})^{\alpha-r} = \sum_{k=0}^{\alpha} c_{k,r}z^{-k} \tag{6}$$

where r = m or n.

Combining (5) and (6), we obtain

$$\hat{H}(z) = \frac{\sum_{n=0}^{N} a_n \sum_{k=0}^{\alpha} c_{k,n} z^{-k}}{\sum_{m=0}^{N} b_m \sum_{k=0}^{\alpha} c_{k,m} z^{-k}} \tag{7}$$

An interchange of the order of summation in the numerator and denominator of (7) leads to:

$$\hat{H}(z) = \frac{\sum_{k=0}^{\alpha} [\sum_{n=0}^{N} a_n c_{k,n}] z^{-k}}{\sum_{k=0}^{\alpha} [\sum_{m=0}^{M} b_m c_{k,m}] z^{-k}} \tag{8}$$

That is

$$\hat{H}(z) = \frac{\hat{a}_o + \hat{a}_1 z^{-1} + \hat{a}_2 z^{-2} + \ldots + \hat{a}_\alpha z^{-\alpha}}{\hat{b}_o + \hat{b}_1 z^{-1} + \hat{b}_2 z^{-2} + \ldots + \hat{b}_\alpha z^{-\alpha}} \tag{9}$$

where

$$\hat{a}_k = \sum_{n=0}^{N} a_n c_{k,n}$$

and

$$\hat{b}_k = \sum_{m=0}^{M} b_m c_{k,m} \tag{10}$$

We now show that the set of coefficients $\{c_{k,r}\}$ in (10) can be computed recursively via a linear partial difference equation. Clearly, once these coefficients are known, the desired $\hat{H}(z)$ in (9) can be readily obtained.

Difference Equation Derivation. For convenience, let $x = z^{-1}$. Then (6) can be rewritten as

$$(1-x)^r (1+x)^{\alpha-r} = \sum_{k=0}^{\alpha} c_{k,r} x^k \tag{11}$$

From the left-hand side of (11) it follows that

$$(1-x)^r(1+x)^{\alpha-r} = \frac{1-x}{1+x}\left[(1-x)^{r-1}(1+x)^{\alpha-r+1}\right]$$

$$= \frac{1-x}{1+x}\sum_{k=0}^{\alpha} c_{k,r-1}x^k \tag{12}$$

Thus, combining (11) and (12) we obtain

$$\sum_{k=0}^{\alpha} c_{k,r}x^k = \frac{1-x}{1+x}\sum_{k=0}^{\alpha} c_{k,r-1}x^k \tag{13}$$

Again, a Taylor series expansion of the term $(1-x)/(1+x)$ in (13) leads to

$$\frac{1-x}{1+x} = -1 + 2\sum_{i=0}^{\infty}(-x)^i \tag{14}$$

Substitution of (14) in (13) and subsequent manipulation of terms results in

$$\sum_{k=0}^{\alpha} c_{k,r}x^k = -\sum_{k=0}^{\alpha}\{c_{k,r-1} - 2\sum_{i=0}^{k}(-1)^i c_{k-i,r-1}\}x^k ,$$

which implies that

$$c_{k,r} = -c_{k,r-1} + 2\sum_{i=0}^{k}(-1)^i c_{k-i,r-1} \quad , \quad k = 0,1,\ldots,\alpha \tag{15}$$

Now, for the special case when k=0, (15) yields the recurrence relation

$$c_{0,r} = c_{0,r-1} \tag{16}$$

Thus we restrict our attention to $k = 1,2,\ldots,\alpha$ in (15) to obtain

$$c_{k-1,r} = -c_{k-1,r-1} + 2\sum_{i=1}^{k}(-1)^{i+1} c_{k-i,r-1} \quad , \quad k = 1,2,\ldots,\alpha \tag{17}$$

Again, from (15) and (17) we have

$$c_{k,r} + c_{k-1,r} = -c_{k,r-1} - c_{k-1,r-1} + 2\left\{\sum_{i=0}^{k}(-1)^i c_{k-i,r-1}\right.$$

$$\left. + \sum_{i=1}^{k}(-1)^{i+1} c_{k-i,r-1}\right\} ,$$

which yields

$$c_{k,r} + c_{k-1,r} = -c_{k,r-1} - c_{k-1,r-1} + 2c_{k,r-1} + 2 \sum_{i=1}^{k} [(-1)^i + (-1)^{i+1}]c_{k-i,r-1}$$

$$= c_{k,r-1} - c_{k-1,r-1} \quad , \tag{18}$$

since the quantity $[(-1)^i + (1)^{i+1}] \equiv 0$ for all i. Hence (16) and (18) yield the desired difference equation for the coefficients $c_{k,r}$ to be as follows:

$$c_{k,r} = c_{k,r-1} - c_{k-1,r} - c_{k-1,r-1} \quad , \quad k,r = 1,2,\ldots,\alpha \tag{19}$$

The boundary conditions associated with (19) are obtained by equating coefficients of x^0 and x^k on both sides of (11). This process results in the boundary conditions

$$c_{0,r} = 1$$

and $$c_{k,0} = \binom{\alpha}{k} = \frac{\alpha!}{k!(\alpha-k)!} \quad , \quad k,r = 0,1,\ldots,\alpha \tag{20}$$

From the above discussion it is apparent that the transfer function $\hat{H}(z)$ in (9) corresponding to a given H(s) in (1) can be computed via a simple algorithm which consists of the following steps.

1. Find $\alpha = \max(M,N)$ where M and N are the degrees of the numerator and denominator polynomials of H(s), respectively.

2. Compute the sets of coefficients

$$\{c_{k,n}\} = \{c_{k,0}, c_{k,1}, \ldots, c_{k,N}\}$$

and

$$\{c_{k,m}\} = \{c_{k,0}, c_{k,1}, \ldots, c_{k,M}\} ,$$

recursively using the following relation [see (19) and (20)]:

$$c_{k,r} = c_{k,r-1} - c_{k-1,r} - c_{k-1,r-1} \quad , \quad k,r = 1,2,\ldots,\alpha$$

where

$$c_{0,r} = 1$$

and $$c_{k,0} = \binom{\alpha}{k} \quad , \quad k,r = 0,1,\ldots,\alpha \tag{21}$$

3. Use (10) to compute the coefficients $\hat{a}_k$ and $\hat{b}_k$; i.e.

$$\hat{a}_k = \sum_{n=0}^{N} a_n c_{k,n}$$

and $$\hat{b}_k = \sum_{m=0}^{M} b_n c_{k,m}$$

4. Obtain the desired $\hat{H}(z)$ which is given by [see (9)]

$$\hat{H}(z) = \frac{\hat{a}_0 + \hat{a}_1 z^{-1} + \hat{a}_2 z^{-2} + \ldots + \hat{a}_N z^{-\alpha}}{\hat{b}_0 + \hat{b}_1 z^{-1} + \hat{b}_2 z^{-2} + \ldots + \hat{b}_M z^{-\alpha}}$$

For the purpose of illustration, we consider an example. Let a given H(s) be as follows:

$$H(s) = \frac{1 + s^2}{1 + s + s^2} \tag{22}$$

Then, from (1) it follows that

$$a_i = b_j = 1 \quad , \quad i = 0, 2 \quad , \quad j = 0,1,2$$

and

$$\alpha = \max(2,2) = 2 \tag{23}$$

Next, from (21) one obtains

$$c_{0,r} = 1 \quad , \quad r = 0,1,2$$

$$c_{1,0} = 2 \quad , \quad c_2,0 = 1$$

and

$$c_{1,1} = 0 \quad , \quad c_{1,2} = -2 \quad , \quad c_{2,1} = -1 \quad , \quad c_{2,2} = 1 \, . \tag{24}$$

Substitution of the information in (23) and (24) in (10) leads to

$$\hat{a}_0 = 2 \quad , \quad \hat{a}_1 = 0 \quad , \quad \hat{a}_2 = 2$$

and

$$\hat{b}_0 = 3 \quad , \quad \hat{b}_1 = 0 \quad , \quad \hat{b}_2 = 1$$

Thus (9) yields the desired transfer function to be

$$\hat{H}(z) = \frac{2 + 2z^{-2}}{3 + z^{-2}} \, . \tag{25}$$

III. CLOSED-FORM SOLUTION CONSIDERATIONS

In this section a closed-form solution for the difference equation developed in the previous section is obtained.

We rewrite the difference equation in (19) as

$$c_{k+1,r+1} - c_{k+1,r} + c_{k,r+1} + c_{k,r} = 0 \quad , \quad k,r = 1,2,\ldots,\alpha \tag{26}$$

Taking the z-transform of (26) with respect to k, we obtain

$$(z+1)C_{r+1}(z) + (1-z)C_r(z) + z(c_{0,r} - c_{0,r+1}) = 0 \tag{27}$$

where

$$Z\{c_{k+1,r}\} = zC_r(z) - zc_{0,r}$$

Now, from the boundary condition $c_{0,r} = 1$ and (16) it follows that

$$c_{0,r} = c_{0,r+1} = 1$$

Thus (27) simplifies to yield

$$C_{r+1}(z) - \left(\frac{z-1}{z+1}\right)C_r(z) = 0 \, , \tag{28}$$

which is a simple first-order linear difference equation with respect to the variable r. Hence the solution to (28) is given by

$$C_r(z) = A(z)\ \left(\frac{z-1}{z+1}\right)^r \quad , \quad r = 0,1,2,\ldots,\alpha \tag{29}$$

where A(z) is yet to be determined. To this end we consider the case $r = 0$. Then, (29) yields

$$C_0(z) = A(z) \tag{30}$$

However, by definition, we have

$$C_0(z) = Z\{c_{k,0}\}$$

where $c_{k,0} = \binom{\alpha}{k}$ by virtue of the boundary condition in (21). Thus

$$C_o(z) = \sum_{k=0}^{\alpha} \binom{\alpha}{k} z^{-k}$$

$$= (1 + z^{-1})^{\alpha} \tag{31}$$

Hence from (29), (30) and (31) it follows that

$$C_r(z) = z^{-\alpha}(z-1)^r(z+1)^{\alpha-r} \quad , \quad r = 0,1,\ldots,\alpha \tag{32}$$

That is

$$C_r(z) = \sum_{i=0}^{\alpha} g_i z^{-i} \tag{33}$$

where, it can be verified that

$$g_i = \frac{1}{(\alpha-i)!}\frac{d^{\alpha-i}}{dz^{\alpha-i}}\left\{(z-1)^r(z+1)^{\alpha-r}\right\}_{z=0}$$

$$= \frac{1}{(\alpha-i)!}\sum_{j=0}^{\alpha-i}(-1)^{j+r}\binom{\alpha-i}{j}(\alpha-r)^{(\alpha-i-j)}r^{(j)} ,$$

the notation $p^{(q)}$ denoting decending factorials; i.e.

$$p^{(q)} = p(p-1)(p-2)\ \ldots\ (p-q+1)$$

From (33) it is apparent that the g_i are merely the inverse Z-transform coefficients of $C_r(k)$ in (32). Thus the desired closed-form solution is given by

$$c_{k,r} = \frac{1}{(\alpha-k)!}\sum_{j=0}^{\alpha-k} (-1)^{j+r}\binom{\alpha-k}{j}(\alpha-r)^{(\alpha-k-j)}{}_r(j) \quad , \quad k = 0,1,\ldots,\alpha \qquad (34)$$

where r = m or n.

IV. ILLUSTRATIVE EXAMPLE

For the purposes of illustrating the use of the algorithm derived in Section II, we consider the third-order normalized lowpass Butterworth filter transfer function

$$H(s) = \frac{1}{s^3 + 2s^2 + 2s + 1}, \qquad (35)$$

and seek the corresponding digital bandpass filter with the following specifications:

Lower cutoff frequency = 4Hz

Upper cutoff frequency = 8Hz

Sampling rate = 62 sps

Using the standard lowpass-bandpass transformation, followed by the conventional design procedure for designing digital filters via the bilinear transformation [2-4], we obtain the following frequency-scaled transfer function [5]:

$$\hat{H}(s) = \frac{.010867s^3}{s^6 + .443s^5 + 3.58153s^4 + .087662s^3 + .031043s^2 + .003328s + .000651} \qquad (36)$$

Now, we compute the digital filter transfer function

$$H(z) = \hat{H}(s)\Big|_{s = \frac{z-1}{z+1}},$$

using the FORTRAN program given in Appendix I.

The input to the program consists of three or more data cards, the first of which contains the degree of H(s) in columns 1 and 2. The next set of cards consists of the coefficients of the ascending powers of the numerator polynomial of H(s), while the rest of the cards consist of the corresponding denominator coefficients. In each case, these cards are punched using an F10.5 format. The resulting digital transfer function is as given by $\hat{H}(z)$ in (9), with α=6, and,

$$\hat{a}_0 = -0.01087 \;,\; \hat{a}_2 = 0.0326 \;,\; \hat{a}_4 = -0.0326 \;,\; \hat{a}_6 = 0.01087 \;,$$

$$\hat{a}_i = 0 \;,\; i = 1,3,5$$

and

$$\hat{b}_0 = 0.85586 \;,\; \hat{b}_1 = -4.89163 \;,\; \hat{b}_2 = 12.6519 \;,\; \hat{b}_3 = -18.6785 \;,$$

$$\hat{b}_4 = 16.5892 \;,\; \hat{b}_5 = -8.4090 \;,\; \hat{b}_6 = 1.92384 \;.$$

The corresponding magnitude and phase characteristics of the above filter are as shown in Fig. 1, from which it is apparent that the filter does possess the desired bandpass characteristics.

V. CONCLUDING REMARKS

The algorithm presented in this paper enables one to readily compute the bilinear transform corresponding to a given transfer function H(s). It yields the numerator and denominator coefficients $\{\hat{a}_k, \hat{b}_k\}$ of the desired H(z) in terms of a linear combination of the corresponding coefficients $\{a_k, b_k\}$ associated with H(s). This linear combination is in terms of a set of <u>integer</u> coefficients $\{c_{k,r}\}$ which are obtained exactly using integer arithmetic, via a recurrence relation which is in the form of a linear partial difference equation. Hence it follows that the algorithm is amenable to a straightforward computer implementation using a convenient high level language such as FORTRAN. As such, this algorithm provides a very general and efficient means of computing the bilinear transform relative to the conventional approach which involves a direct substitution of $s = \frac{z-1}{z+1}$ in H(s), followed by a set of tedious algebraic manipulations.

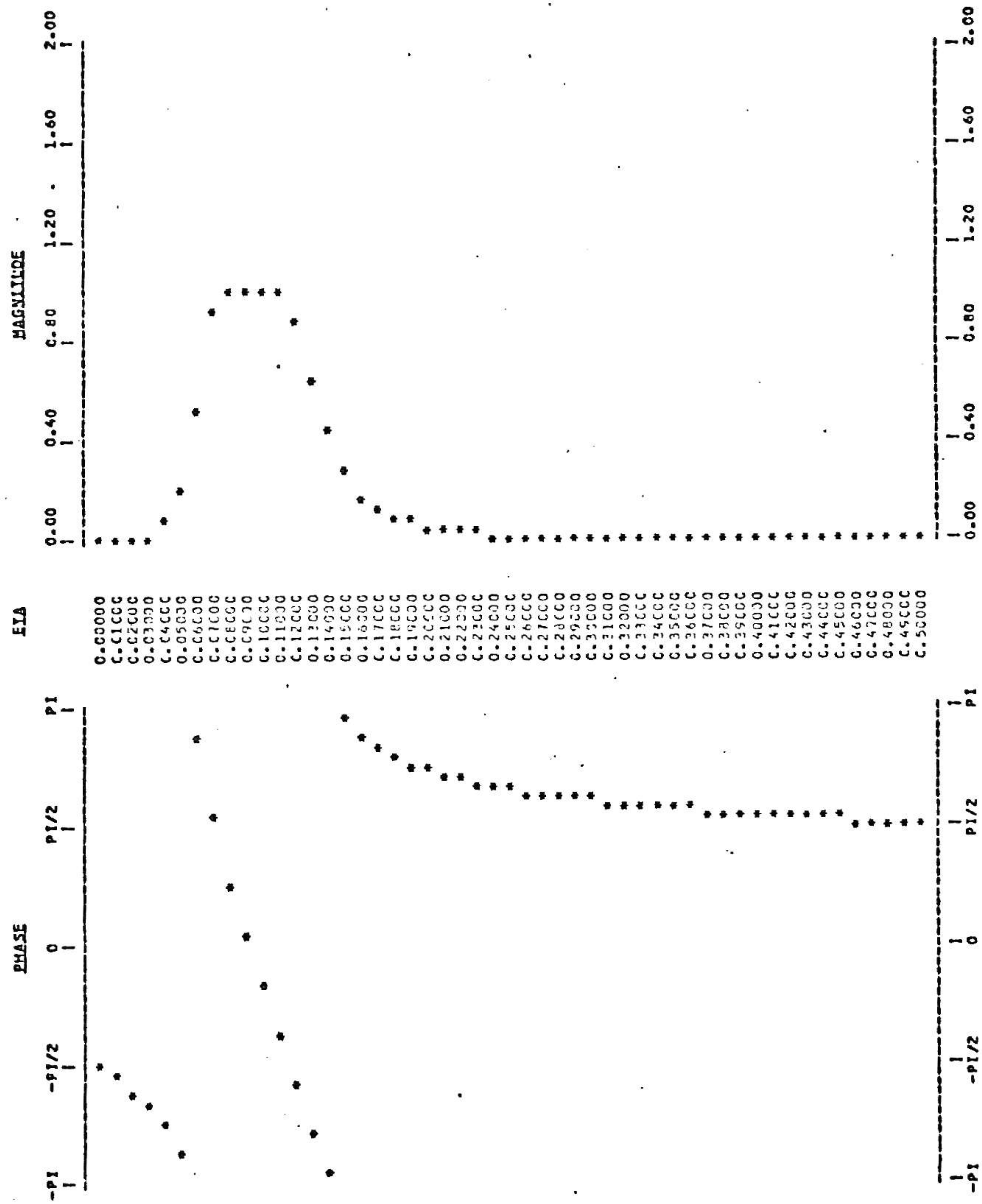

Figure 1

Phase and magnitude response of a sixth-order
Butterworth digital bandpass filter

VI. REFERENCES

[1] W. D. Stanley, Digital Signal Processing, Restow Publishing Co., Inc., Chapter 7.

[2] A. V. Oppenheim and R. W. Schafer, Digital Signal Processing, Prentice-Hall, 1975, Chapter 5.

[3] S. D. Stearns, Digital Signal Analysis, Hayden Publishing Co., 1975, Chapter 12.

[4] L. Rabiner and B. Gold, Theory and Application of Digital Signal Processing, Prentice-Hall, 1975, Chapter 4.

[5] N. Ahmed, et al., "On the Digital Processing of EEG Data," Proc. 1976 Int. Symp. on Electromag. Compat., July 1976.

VII. ACKNOWLEDGMENTS

I would like to take this opportunity to thank all those who assisted me during my study. I would especially like to thank Dr. Nasir Ahmed for his encouragement and guidance, along with Dr. Donald R. Hummels and Dr. Donald H. Lenhert for their assistance. I also wish to thank the Department of Electrical Engineering and the Engineering Experiment Station for their financial support.

Finally, I would like to thank Dakshesh Parikh for his assistance and encouragement.

```
      REAL AS(21),BS(21),AZ(21),BZ(21)
      INTEGER A,A1
      WRITE(6,7000)
C     READ THE DATA AND ECHO IT.
 1000 READ(5,3000,END=2000) A                16
      A1=A+1
      READ(5,4000) (AS(K),K=1,A1)
      READ(5,4000) (BS(K),K=1,A1)
      WRITE(6,5000) (AS(K),K=1,A1)
      WRITE(6,5000) (BS(K),K=1,A1)
      CALL BILINE(AS,BS,AZ,BZ,A)
C     PRINT THE RESULTS.
      WRITE(6,5000) (AZ(K),K=1,A1)
      WRITE(6,5000) (BZ(K),K=1,A1)
      GO TO 1000
 2000 WRITE(6,6000)
      STOP
 3000 FORMAT(I2)
 4000 FORMAT(8F10.5)
 5000 FORMAT(' ',8E15.6)
 6000 FORMAT('1    END OF INPUT  ')
 7000 FORMAT('1')
      END
```

```
      SUBROUTINE BILINE(AS,BS,AZ,BZ,A)
C     THIS SUBROUTINE PERFORMS THE BILINEAR TRANSFORMATION ON A
C     TRANSFER FUNCTION OF DEGREE A.  THE NUMERATOR COEFFICIENTS
C     ARE IN AS AND THE COEFFICIENTS OF THE DENOMINATOR ARE IN BS.
C     THE TRANSFORMED COEFFICIENTS ARE LEFT IN AZ AND BZ.
      INTEGER A,A1,COEF(21,21)
      REAL AS(21),BS(21),AZ(21),BZ(21)
      A1=A+1
C     CALCULATE THE BOUNDARY CONDITIONS.
      DO 1000 K=1,A1
 1000 COEF(1,K)=1
      IFACT=1
      DO 2000 K=2,A1
      IFACT=IFACT*(A-K+2)/(K-1)
 2000 COEF(K,1)=IFACT
C     GENERATE THE TRANSFORMATION COEFFICIENTS.
      DO 3000 K=2,A1
      DO 3000 J=2,A1
 3000 COEF(K,J)=COEF(K,J-1)-COEF(K-1,J)-COEF(K-1,J-1)
      DO 5000 K=1,A1
C     DO THE SUMMATIONS.
      ASUM=0
      BSUM=0
      DO 4000 J=1,A1
      ASUM=ASUM+COEF(K,J)*AS(J)
 4000 BSUM=BSUM+COEF(K,J)*BS(J)
      AZ(K)=ASUM
 5000 BZ(K)=BSUM
      RETURN
      END
```

ALGORITHMIC PROPERTIES OF THE BILINEAR TRANSFORM

by

DAVID NICHOLAS HEIN

B.S., Kansas State University, 1974

AN ABSTRACT OF A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1976

A general algorithm for computing the bilinear transform corresponding to a given rational function H(s) is developed. The algorithm transforms an H(s) in the form of a ratio of two polynomials into an H(z) of the same form. This algorithm is due to a recurrence relation which is in the form of a linear partial difference equation, a closed-form solution for which is also presented. In addition, an efficient software implementation for the algorithm is included using FORTRAN as the programming language. The FORTRAN program uses the difference equation rather than its closed-form solution in performing the bilinear transformation.