N73-31145

FLOATING-POINT SYSTEM QUANTIZATION ERRORS IN DIGITAL CONTROL SYSTEMS

PREPARED BY

SAMPLED-DATA CONTROL SYSTEMS GROUP

AUBURN UNIVERSITY

C. L. Phillips, Project Leader

Final Technical Report

June, 1973

CONTRACT NAS8-28262

GEORGE C. MARSHALL SPACE FLIGHT CENTER

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

HUNTSVILLE, ALABAMA

Approved By:

J. David Irwin
Associate Professor and Head
Electrical Engineering

Submitted By:

C. L. Phillips
Professor
Electrical Engineering

72

FOREWORD

Auburn Research Foundation submitted a proposal which resulted in Contract NAS8-28262 being awarded on March 16, 1972. The contract was awarded to the Auburn University Engineering Experiment Station by the George C. Marshall Space Flight Center, National Aeronautics & Space Administration, Huntsville, Alabama, and was active until June 15, 1973.

This report is the final technical report of the work accomplished by the Electrical Engineering Department, Auburn University, in the performance of the contract.

SUMMARY

This report contains the results of research into the effects on system operation of signal quantization in a digital control system. The investigation considered digital controllers (filters) operating in floating-point arithmetic in either open-loop or closed-loop systems. An error analysis technique is developed, and is implemented by a digital computer program that is based on a digital simulation of the system. As an output the program gives the programing form required for minimum system quantization errors (either maximum or rms errors), and the maximum and rms errors that appear in the system output for a given bit configuration. The program can be integrated into existing digital simulations of a system.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. INTRODUCTION

The introduction of a digital controller (filter) into a continuous data system presents problems to the design engineer that do not exist with the use of analog controllers. A major problem in the design of digital control systems is the determination of the effects, on system performance, of signal quantization within the digital controller. This report presents the results of an investigation into the determination of the quantization errors, for filters using floating-point arithmetic, and the development of design techniques to minimize these errors. Throughout this report the terms digital filter and digital controller will be used interchangeably.

A problem in the implementation of a digital filter is the choice of the programing form (method of programing) used to realize the filter. Generally the use of different programing forms leads to different system error magnitudes, caused by signal quantization within the digital filter. A considerable amount of research has been published on this topic [See References and Bibliography]. In [2], a technique was reported for choosing programing forms for filters using fixed-point arithmetic. This report presents a technique for choosing programing forms for filters using floating-point arithmetic. The research listed in the References and Bibliography is concerned generally with digital filters in an open-loop configuration. The error analysis techniques require the use of transfer functions, which is not a problem for low-order open-loop filters. However, for high-order digital control systems, the development of the required transfer functions can be a major undertaking. The technique developed in this report is based on a digital simulation of the closed-loop system, and thus the pulse transfer function of the continuous parts of the system are not required.

II. FLOATING-POINT ARITHMETIC

In this chapter quantization errors that result from arithmetic performed in a floating-point format in digital devices are investigated. In Chapter III, the results will be applied to the analysis of digital control systems to determine system errors resulting from the quantization.

Floating-Point Format

In floating-point arithmetic [1], a number x_m is represented as the product of two terms,

$$x_m = E \cdot F \tag{2-1}$$

where a part of the bit configuration of the computer word is used to represent E, and the remainder to represent F. The term E is the exponent and is of the form 2^γ for a base 2 computer, 16^γ for a base 16 computer, etc., where γ is a signed integer. The bit configuration for E yields the value of γ. The term F is the fraction, and is normally set such that

$$1/2 \leq F < 1, \tag{2-2}$$

for a base 2 computer. If the base of the computer is 2^k,

$$1/2^k \leq F < 1 \tag{2-3}$$

The number zero is a special representation. The bit configuration for F yields F directly, where the first bit in F represents 1/2, the second bit 1/4, etc.

Let s be the number of bits assigned to the exponent, and t the number assigned to the fraction (excluding the sign bit for the fraction). Also let

$$s + t = n$$

Thus there is a total of n + 1 bits in the computer word configuration, with the additional bit used to give the sign of the number represented. The maximum magnitude of the exponent for a base 2 computer is

$$E_m = 2^{[2^{(s-1)}-1]} \quad , \tag{2-4}$$

and, for a base 2^k computer, is

$$E_m = 2^{k[2^{(s-1)}-1]} \tag{2-5}$$

The factor (s-1) appears since one bit of s must be used to give the sign of the exponent. The maximum magnitude of the fraction, F, is

$$F_m = 1 - 1/2^t \tag{2-6}$$

Thus the maximum magnitude that can be represented by the n bits is

$$M_{f\ell} = 2^{k[2^{(s-1)}-1]}[1-1/2^t] \approx 2^{k[2^{(s-1)}-1]} \tag{2-7}$$

for a base 2^k computer. Table 2-1 lists the numbers that can be represented in a base 2 computer by a bit configuration with n equal to five. In this configuration, s is equal to three, and is the first three bits. Then t is two, and F is represented by the last two bits. If F satisfies (2-2), the fourth bit from the left in the configuration is always 1. These values are indicated by an asterisk. The bit representation for zero is also shown by an asterisk. The truncation quantization characteristic is shown in Figure 2-1 for this case.

Quantization Errors

The characteristics of the quantization errors will be determined in this section [3]. Let x_m be the floating-point machine representation of x. Then

$$x_m = Q_{f\ell}[x] = 2^{k\gamma} \cdot F, \tag{2-8}$$

where $Q_{f\ell}[\cdot]$ indicates the floating-point representation (which is quantized) of the number. Suppose that truncation, as shown

TABLE 2-1
EXAMPLE OF QUANTIZATION

bit configuration	floating-point number
11111	6*
11110	4*
11101	2
11100	0
11011	3*
11010	2*
11001	1
11000	0
10111	3/2*
10110	2/2*
10101	1/2
10100	0
10011	3/4*
10010	2/4*
10001	1/4
10000	0
01111	3/32*
01110	2/32*
01101	1/32
01100	0
01011	3/16*
01010	2/16*
01001	1/16
01000	0
00111	3/8*
00110	2/8*
00101	1/8
00100	0
00011	3/4*
00010	2/4*
00001	1/4
00000	0*

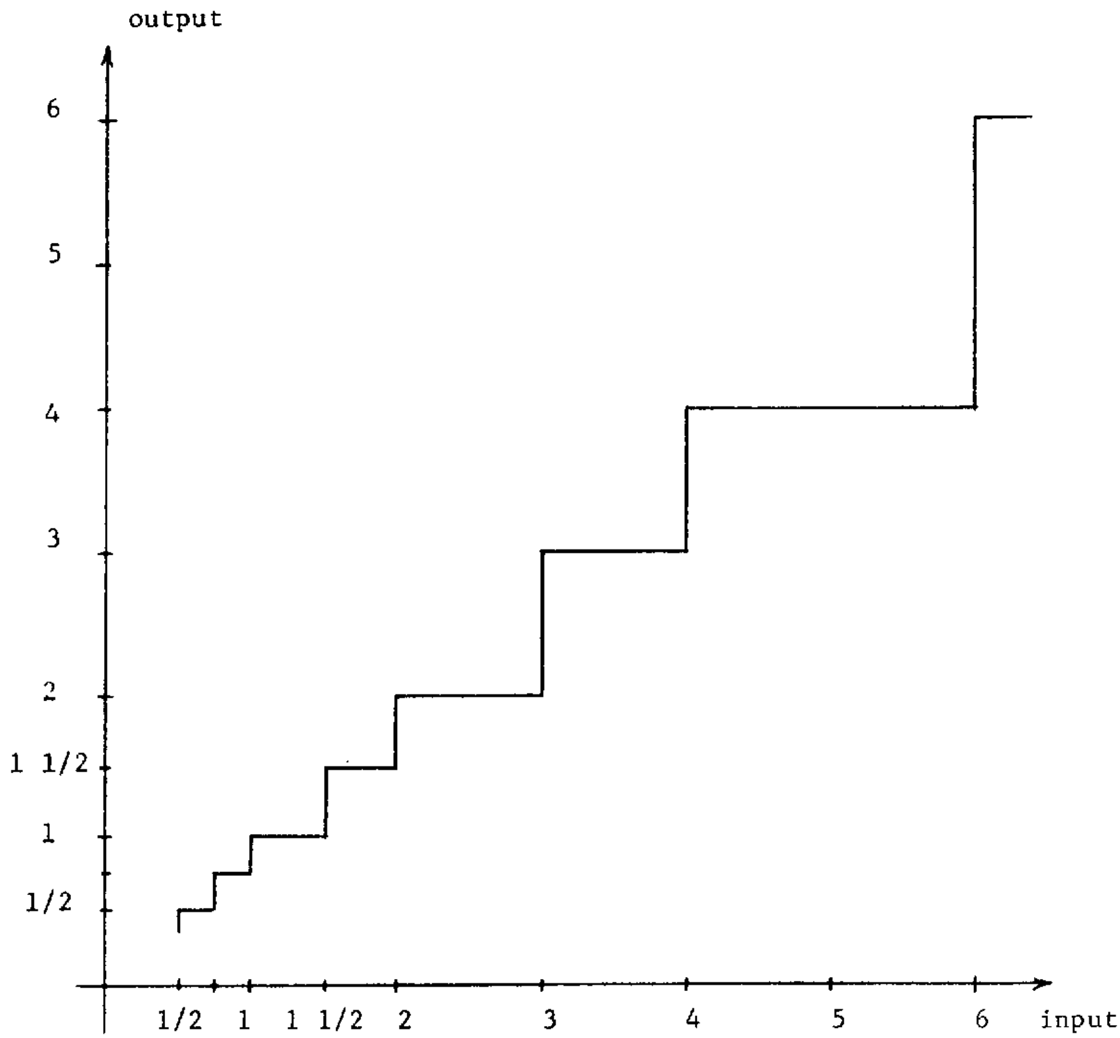


Figure 2-1. Truncation quantization for floating-point arithmetic.

in Figure 2-1, is used in quantizing x. Then the maximum magnitude of the quantization error is seen to be

$$e_m = 2^{k\gamma}(1/2^t) \tag{2-9}$$

and this error is always negative. For roundoff quantization, the maximum error is one-half that of (2-9). Then, from (2-8) and (2-9),

$$e_m = x_m \cdot 2^{-t}/F \tag{2-10}$$

Thus the quantization error is maximum if F is minimum. For a base 2 computer, from (2-2),

$$e_m = x_m \cdot 2^{-(t-1)} \tag{2-11}$$

For a base 16 computer,

$$e_m = x_m \cdot 2^{-(t-4)} \tag{2-12}$$

Thus for a base 2^k computer, where

$$1/2^k \le F < 1, \tag{2-13}$$

then

$$e_m = x_m \cdot 2^{-(t-k)} \tag{2-14}$$

For round-off quantization, the maximum error is one-half that given in (2-14). It is necessary that x and x_m be approximately equal, or else all calculations are meaningless. Thus x_m may be replaced with x in (2-14), with very little resulting error. Thus

$$e_m = x \cdot 2^{-(t-k)} \tag{2-15}$$

In general, the error introduced by truncation quantization is

$$e = 2^{k\gamma}\beta \quad ; \qquad \beta < 1/2^t \tag{2-16}$$

as is seen from Figure 2.1. Thus, from (2-8),

$$e = x_m \cdot \beta/F = x_m \cdot \delta \tag{2-17}$$

where

$$\delta < 2^{-(t-k)} \tag{2-18}$$

for truncation, and

$$-2^{-(t-k+1)} < \delta < 2^{-(t-k+1)} \tag{2-19}$$

for round-off. Generally in this type of analysis, it is assummed that all errors in the ranges given by (2-18) and (2-19) are equally likely to occur. This approach was first taken in this investigation, but the results were grossly in error when compared to actual errors obtained from simulation. Thus it was found to be necessary to use a more accurate representation. For floating-point quantization, the probability density functions for round-off and for truncation are given in Figure 2-2 [3]. These density functions will be used in the analysis developed in Chapter III.

Signal quantization errors occur during four operations in a digital filter. These four operations are: (1) analog-to-digital conversion, (2) digital-to-analog conversion, (3) multiplication, and (4) addition. The errors in (1) and (2) are described above. For multiplication and addition [1],

$$Q_{f\ell}[xy] = (xy)(1 + \delta), \tag{2-20}$$

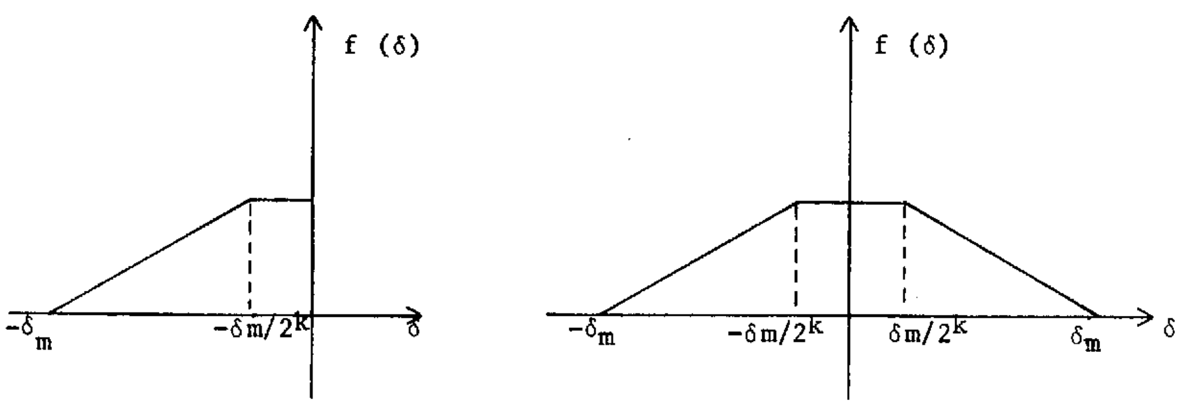
and

$$Q_{f\ell}[x + y] = (x + y)(1 + \delta), \tag{2-21}$$

where δ is given by (2-18) and (2-19). However, care must be exercised [3] in the utilization of (2-21). Consider the addition of two numbers in base 10, with the numbers limited to three decimal places.

$$\begin{array}{lcl} .115 & \times & 10^3 \\ .173 & \times & 10^3 \\ \hline .288 & \times & 10^3 \end{array} \qquad \begin{array}{lcl} .115 & \times & 10^2 \\ .173 & \times & 10^3 \\ \hline .184{\vdots}5 & \times & 10^3 \end{array}$$

In the first case, with the exponents equal, these is no error in the addition. In the second case, with the exponents unequal, there is a quantization error. Thus, in a digital filter, if the exponents of the numbers being added are equal, there will probably be no error in the addition. If the exponents are unequal, the density functions for the errors are given in Figure 2-2.



(a) Truncation (b) Round-off

Figure 2-2. Probability density function for δ.

III. SYSTEM ERROR ANALYSIS

In this chapter, the error models derived in Chapter II will be utilized to develop a technique for determining system errors due to quantization in a digital control system. The technique will be implemented using a computer simulation of the control system.

System Errors

It was shown in Chapter II that the quantized representation of a number in floating-point format can be written as

$$Q_{f\ell}(x) = x(1 + \delta) \tag{3-1}$$

or

$$x = Q_{f\ell}(x) - x\delta \tag{3-2}$$

Thus the quantization operation may be modeled as shown in Figure 3-1.

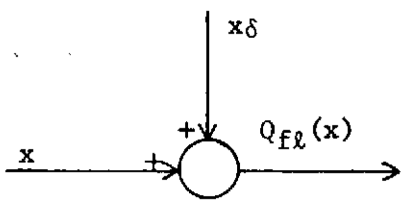


Figure 3-1. Model of floating-point quantization.

Consider now an otherwise linear discrete system that contains a single quantizer. Let the

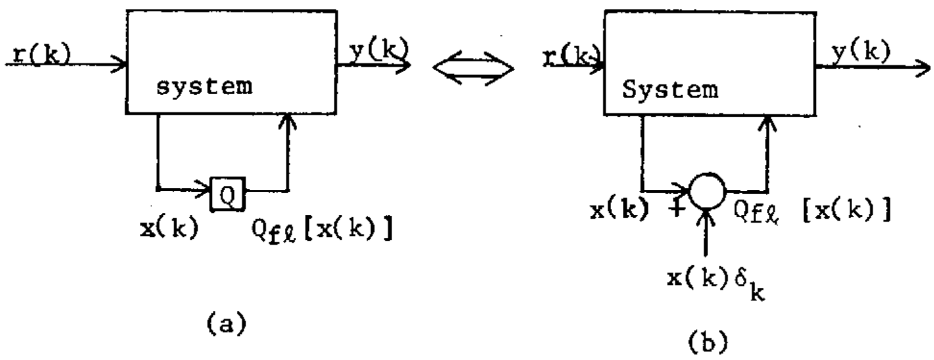


Figure 3-2. Discrete system containing quantization.

system be represented as shown in Figure 3-2. The system (b) of Figure 3-2 can be considered to be linear if the sequence $x(k)\delta_k$, the error at the k^{th} sampling instant, is determined in advance. The output y(k) then can be expressed as

$$y(k) = y_r(k) + y_q(k), \tag{3-3}$$

where $y_r(k)$ is the response from the input r(k), and $y_q(k)$ is the response from $x(k)\delta_k$. Thus $y_q(k)$ is the error in the output due to the quantization. Let G(z) be the transfer function from r(k) to x(k). Then

$$X(z) = R(z)G(z) = x(0) + x(1)z^{-1} + \dots \tag{3-4}$$

Let

$$X_\delta(z) = \delta_0 x(0) + \delta_1 x(1) z^{-1} + \delta_2 x(2) z^{-2} + \ldots, \tag{3-5}$$

Also let H(z) be the transfer function from the error source $x(k)\delta_k$ to the output y(k). Then

$$\begin{aligned} Y_q(z) &= H(z)\, X_\delta(z) \\ &= \delta_0 x(0) H(z) + \delta_1 x(1) z^{-1} H(z) + \delta_2 x(2) z^{-2} H(z) + \ldots \end{aligned} \tag{3-6}$$

At the k^{th} sampling instant,

$$y_q(k) = \delta_0 x(0) h(k) + \delta_1 x(1) h(k-1) + \ldots + \delta_k x(k) h(0), \tag{3-7}$$

where {h(k)} is the impulse response from the error source to the output, given by

$$H(z) = h(0) + h(1) z^{-1} + h(2) z^{-2} + \ldots \tag{3-8}$$

Assume that the quantization is round-off. Then δ_i is given by (2-19), which is repeated as (3-9) below.

$$-2^{-(t-k+1)} < \delta < 2^{-(t-k+1)} \tag{3-9}$$

Thus it is seen from (3-7) that the maximum magnitude of the error in the output is given by

$$|y_{qmax}(k)| = \delta_{max}[|x(0)h(k)| + |x(1)h(k-1)| + \ldots + |x(k)h(0)|], \tag{3-10}$$

where

$$\delta_{max} = 2^{-(t-k+1)} \tag{3-11}$$

For a base 2 machine,

$$\delta_{max} = 2^{-t} \tag{3-12}$$

Consider now truncation quantization. For truncation, the error is alway negative or zero. Thus, in (3-7), each δ_i is either negative or zero. The maximum possible value for $y_q(k)$ is obtained from (3-7) by allowing δ_i to assume its maximum magnitude, and first summing all positive terms, and then summing all negative terms. The maximum possible magnitude of $y_q(k)$ is then the larger of the two sums, in magnitude. It is seen that larger of the two sums is at least one-half the sum of (3-10). However, it is to be recalled that δ_{max} for truncation is twice that for round-off, for a given t and k.

To determine the mean-square error, consider equation (3-7). The output $y_q(k)$ can be considered to be the output of a system whose input is given by

$$\Delta(z) = \delta_0 + \delta_1 z^{-1} + \delta_2 z^{-2} + \ldots , \tag{3-13}$$

and whose tranfer function is given by

$$\begin{aligned} F(z) &= h(0)x(k) + h(1)x(k-1)z^{-1} + h(2)x(k-2)z^{-2} + \ldots \\ &= f(0) + f(1)z^{-1} + f(2)z^{-2} + \ldots \end{aligned} \tag{3-14}$$

The δ_i of (3-13) are assumed to be independent, and to have the density functions given in Figure 2-2. The following development will be for both round-off and truncation quantization. Let m_1 be the expected value of δ_i, and m_2 be the second moment,[1] i.e.,

$$E[\delta_i] = m_1; \qquad E[\delta_i^2] = m_2; \text{ for all } i \tag{3-15}$$

Consider now (3-7), (3-13), and (3-14). The expected value of the output error resulting from a single quantizer is

$$\begin{aligned} E[y_q(k)] &= E[\delta_0 f(0) + \delta_1 f(1) + \ldots + \delta_k f(k)] \\ &= m_1 \sum_{i=0}^{k} f(i) \end{aligned} \tag{3-16}$$

[1] m_1 and m_2 are derived in Appendix A.

The mean-square output error is given by

$$E[y_q^2(k)] = E[\{\delta_0 f(0) + \delta_1 f(1) + \ldots + \delta_k f(k)\}^2] \tag{3-17}$$

Thus

$$E[y_q^2(k)] = E[\delta_0^2]f^2(0) + \ldots + E[\delta_k^2]f^2(k) + 2f(0)f(1)E[\delta(0)]E[\delta(1)]$$
$$+ 2f(0)f(2)E[\delta(0)]E[\delta(2)] + \ldots + 2f(k-1)f(k)E[\delta(k-1)]E[\delta(k)] \tag{3-18}$$

Or

$$E[y_q^2(k)] = m_2 \sum_{i=0}^{k} f^2(i) + 2m_1^2 \sum_{i=0}^{k-1} \sum_{j=i+1}^{k} f(i)f(j) \tag{3-19}$$

But

$$2\sum_{i=0}^{k-1} \sum_{j=i+1}^{k} f(i)f(j) = [\ \sum_{i=1}^{k} f(i)\ ]^2 - \sum_{i=1}^{k} f^2(i) \tag{3-20}$$

Then (3-19) becomes

$$E[y_q^2(k)] = \sum_{i=0}^{k} f^2(k)[m_2 - m_1^2] + [\ \sum_{i=1}^{k} f(i)\ ]^2 m_1^2 \tag{3-21}$$

Consider now the case that n quantization points contribute to the output error. Let $y_t(k)$ be the total error in the output, and let $y_{qi}(k)$ be that part of $y_t(k)$ due to the i^{th} quantization point. Then, the mean output error is

$$E[y_t(k)] = E[y_{q1}(k) + y_{q2}(k) + \ldots + y_{qn}(k)]$$

$$= E[y_{q1}(k)] + E[y_{q2}(k)] + \ldots + E[y_{qn}(k)], \qquad (3\text{-}22)$$

where $E[y_{qi}(k)]$ is given by (3-16). The mean-square output error is given by

$$E[y_t^2(k)] = E[\{y_{q1}(k) + \ldots + y_{qn}(k)\}^2] = E[\{y_{q1}(k)\}^2] + \ldots$$
$$+ E[\{y_{qn}(k)\}^2] + 2E[y_{q1}(k)]E[y_{q2}(k)] + \ldots$$
$$+ 2E[y_{q(n-1)}(k)]E[y_{qn}(k)] \qquad (3\text{-}23)$$

The terms of (3-23) are given by either (3-21) or (3-16).

Error Calculations by Digital Simulation

Both the maximum error of (3-10) and the mean-square error of (3-21) can be obtained from digital simulations of the system. First, with the input of the system of Figure 3-2 equal to the desired input $\{r(k)\}$ and all initial condition set of zero, the sequence $\{x(k)\}$ can be calculated and stored. Next, with zero input and zero initial conditions, the impulse response sequence $\{h(k)\}$ can be obtained by applying are input $\delta_0 x(0) = 1$, and $\delta_i x(i) = 0$, $i > 0$. The system output sequence is then the desired impulse response sequence $\{h(k)\}$. The indicated sums of (3-10) and (3-21) can then be evaluated, yielding the maximum error and the mean-square error for the system. The total maximum error is then the sum of the maximum errors from each quantizer, and the total mean-square error is given by (3-23).

Computer Evaluation of Errors

A computer program has been developed for determining system quantization errors in a digital control system, with the digital filter realized by various programing forms. The utilization of this program will allow the choice of the programing form that will yield the smallest system error due to quantization. This computer program will be described in this section.

All programing forms in the computer program realize the transfer function

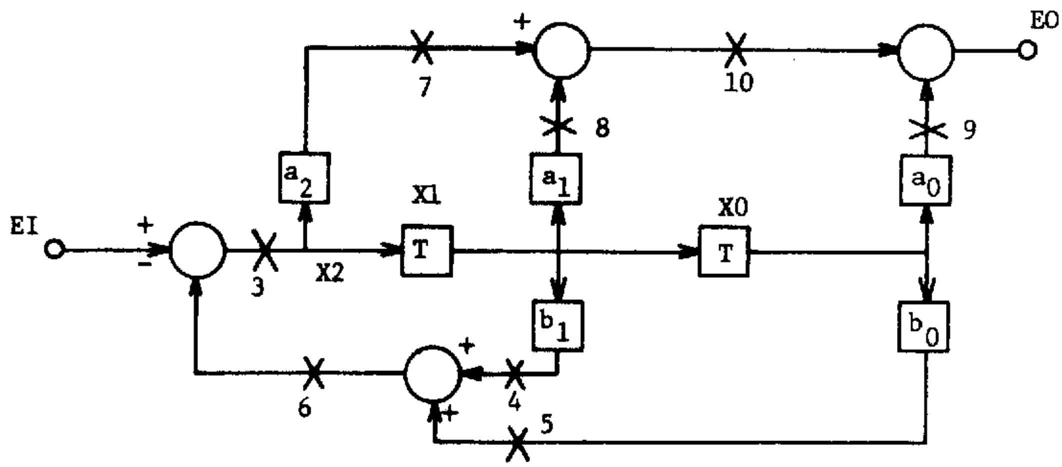
$$D(z) = \frac{a_2 z^2 + a_1 z + a_0}{z^2 + b_1 z + b_0}$$

Consider first the canonical programing form, shown in Figure 3-3. It is assumed that this filter form is connected in a digital control system. Each point at which quantization occurs is indicated by an χ. The input and output quantization points are omitted, and are considered separately, since the errors from these points are the same for all programming forms. Let X2(z) be the signal at quantization point 1 as indicated in

Figure 3-3. Further, let $H_1(z)$ be the transfer function from the error source at quantization point 1 to the system output. Then the errors resulting from quantization point 1 are obtained using X2(z) and $H_1(z)$ in (3-10) and (3-21). The errors resulting from quantization point 2 are obtained using $b_1z^{-1}X2(z)$ and $-H_1(z)$ in (3-10) and (3-21). All necessary signals and transfer functions to determine system errors from the eight points of quantization are given in Figure 3-3.

The sums indicated in (3-10) and (3-21) are evaluated using the computer program given in Appendix B of this report. The value of k in these equations is N1 of the program, and must be given in the main program. The canonical programing form is simulated in subroutine FIL2. Three different simulations of the system are required. In the first simulation (JJ = 1), the sequence {x2(k)} of Figure 3-3 is obtained and stored. Note that the system input is R, and is a unit step in this case. In the second simulation (JJ = 2), the impulse response $\{h_1(k)\}$ is obtained, and $\Sigma f(i)$ and $\Sigma f^2(i)$ are evaluated for quantization points 1, 2, 3, and 4. In the third simulation (JJ = 3), the impulse response $\{h_2(k)\}$ is obtained, and the required sums are evaluated for quantization points 5,6,7and 8.

The quantization error at any point is assumed to be statistically independent of that at any other point. Thus the maximum system error for the filter form is the sum of the maximum system errors for each quantization point, and the mean-square system error is given by (3-23). The total system errors are evaluated using these relationships. The



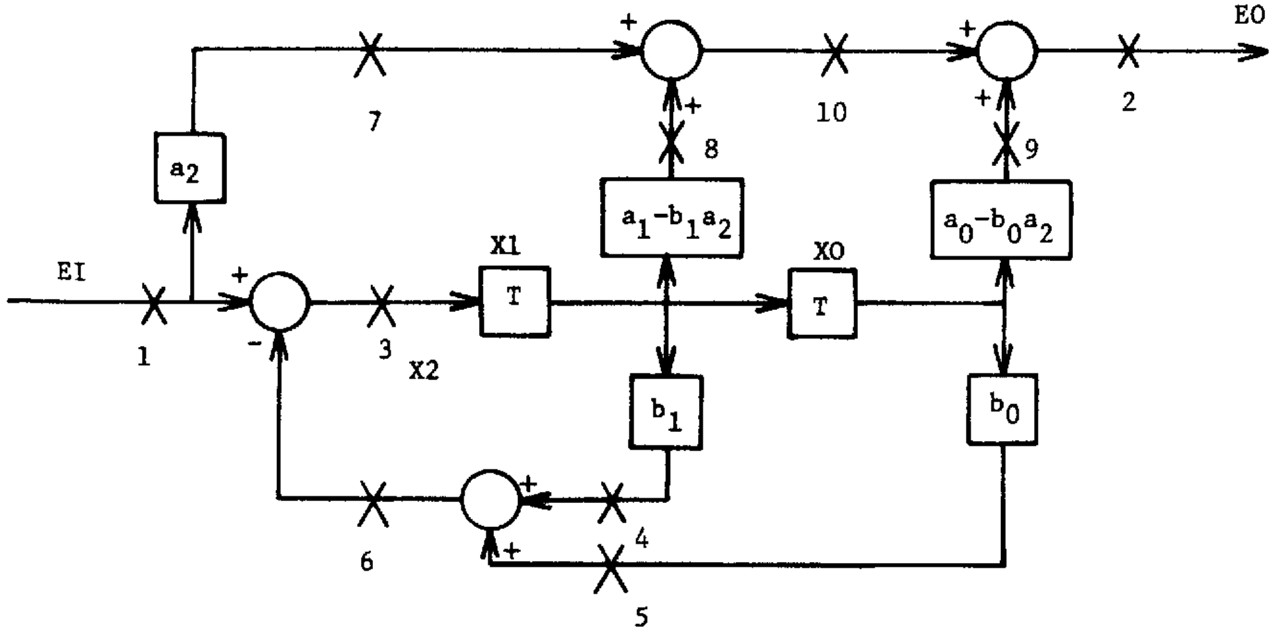
error point	signal at error point	transfer function to output
3	$X2(z)$	$H_1(z)$
4	$b_1z^{-1}X2(z)$	$-H_1(z)$
5	$b_0z^{-2}X2(z)$	$-H_1(z)$
6	$(b_1z^{-1}+b_0z^{-2})X2(z)$	$-H_1(z)$
7	$a_2X2(z)$	$H_2(z)$
8	$a_1z^{-1}X2(z)$	$H_2(z)$
9	$a_0z^{-2}X2(z)$	$H_2(z)$
10	$(a_2+a_1z^{-1})X2(z)$	$H_2(z)$

Figure 3-3. Canonical form.

program prints out the maximum error and the root-mean-square error for each filter form with each divided by δ_{max} of (3-10) and (3-19). Since no errors occur in multiplication if a signal is multiplied by a coefficient equal to unity, logic is included in the program to zero the multiplication errors for this case. To obtain the errors for a given system, the results of the computer program must be multiplied by δ_{max}.

Consider now the modified canonical programing form. This form is shown in Figure 3-4, and is realized as subroutine FIL1 in the computer program. This form is also used in determining the errors from both input and output quantization. For this programing form the sequences $\{X2(k)\}$ and $\{EI(k)\}$ are calculated and stored for JJ = 1. For JJ = 2, the impulse response from X2 to the system output is obtained. For JJ = 3, the impulse response from the filter output is obtained; and for JJ = 4, the impulse response from the filter input is obtained. These responses are used in the manner indicated in Figure 3-4 to calculate system errors.

All filter forms consider are listed in Table 3-1. It is noted that the direct form gives the same system errors as does the canonical form [4], and thus is not considered separately. The parallel form can realize filters containing only real poles, and the XI and XII, forms can realize filters containing only complex poles. Logic is included in the program to insure that these forms are considered only at the appropriate times.



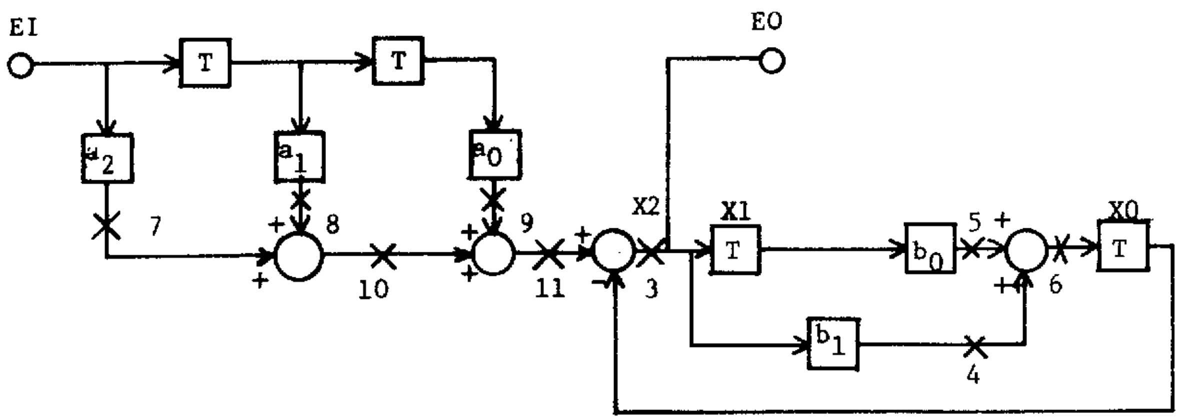
error point	signal at error point	transfer function to output
3	$X2(z)$	$H_1(z)$
4	$b_1z^{-1}X2(z)$	$-H_1(z)$
5	$b_0z^{-2}X2(z)$	$-H_1(z)$
6	4 + 5	$-H_1(z)$
7	$a_2EI(z)$	$H_2(z)$
8	$(a_1-b_1a_2)z^{-1}X2(z)$	$H_2(z)$
9	$(a_0-b_0a_2)z^{-2}X2(z)$	$H_2(z)$
10	7 + 8	$H_2(z)$
2	9 + 10	$H_2(z)$
1	$EI(z)$	$H_3(z)$

Figure 3-4. Modified canonical form

TABLE 3-1

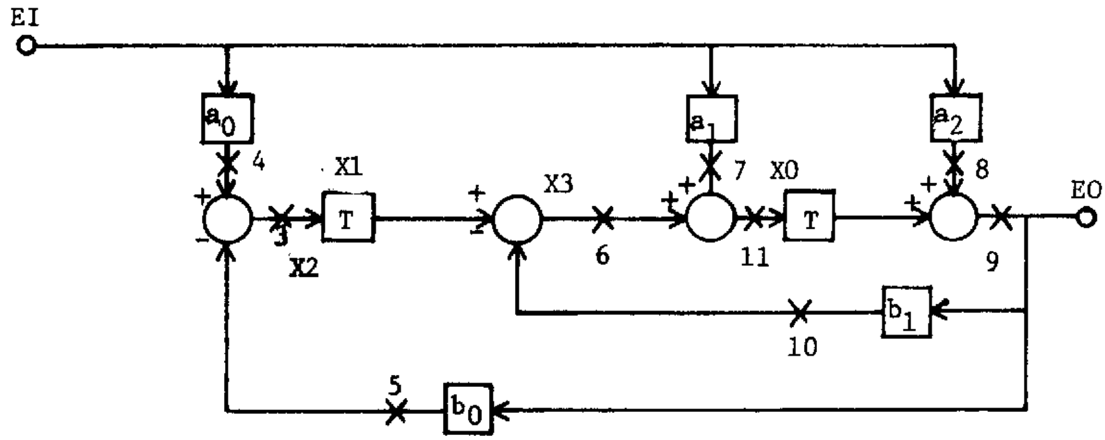
FILTER FORMS

filter form	Figure	Subroutine
modified canonical	3-4	F1L1
canonical	3-3	F1L2
modified direct	3-5	F1L3
direct	errors same as canonical	
modified standard	3-6	F1L4
standard	3-7	F1L5
parallel	3-8	F1L6
XI	3-9	F1L7
XII	3-10	F1L8



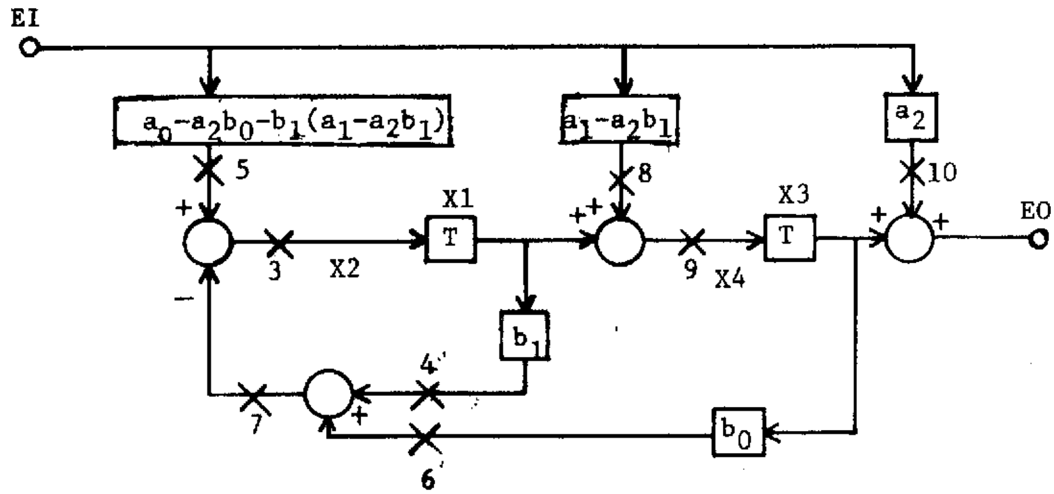
error point	signal at error point	transfer function to output
3	$X2(z)$	$H_1(z)$
4	$b_1X2(z)$	$-z^{-1}H_1(z)$
5	$b_0z^{-1}X2(z)$	$-z^{-1}H_1(z)$
6	4 + 5	$-z^{-1}H_1(z)$
7	$a_2EI(z)$	$H_1(z)$
8	$a_1z^{-1}EI(z)$	$H_1(z)$
9	$a_0z^{-2}EI(z)$	$H_1(z)$
10	7 + 8	$H_1(z)$
11	9 + 10	$H_1(z)$

Figure 3-5. Modified direct form.



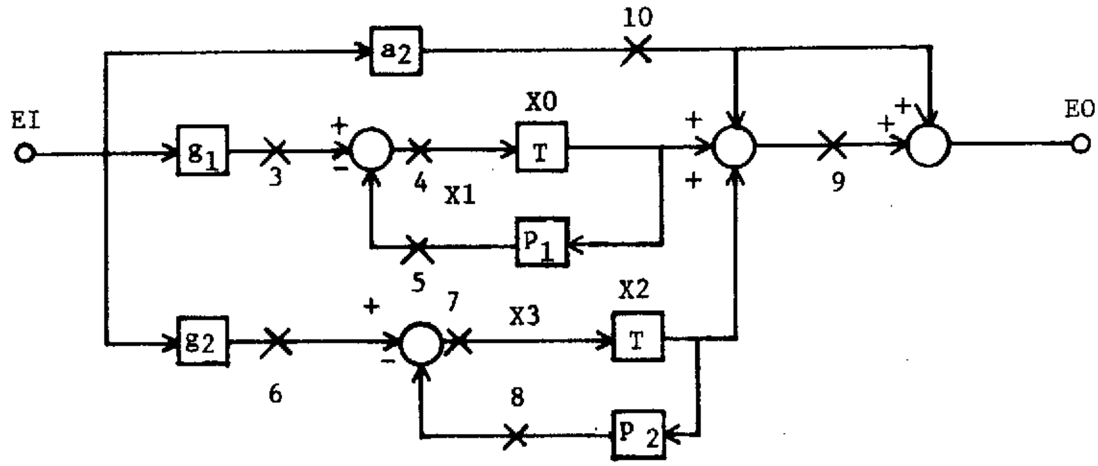
error point	signal at error point	transfer function to output
3	$X2(z)$	$H_1(z)$
4	$a_0EI(z)$	$H_1(z)$
5	$b_0\{z^{-1}[X3(z)+a_1EI(z)]+a_2EI(z)\}$	$-H_1(z)$
6	$X3(z)$	$H_2(z)$
7	$a_1EI(z)$	$H_2(z)$
8	$a_2EI(z)$	$z^{-1}H_2(z)$
9	$z^{-1}[X3(z) + a_1EI(z)] + a_2EI(z)$	$z^{-1}H_2(z)$
10	$b_1\{z^{-1}[X3(z) + a_1EI(z)] + a_2EI(z)\}$	$-H_2(z)$
11	6 + 7	$H_2(z)$

Figure 3-6. Modified standard form.



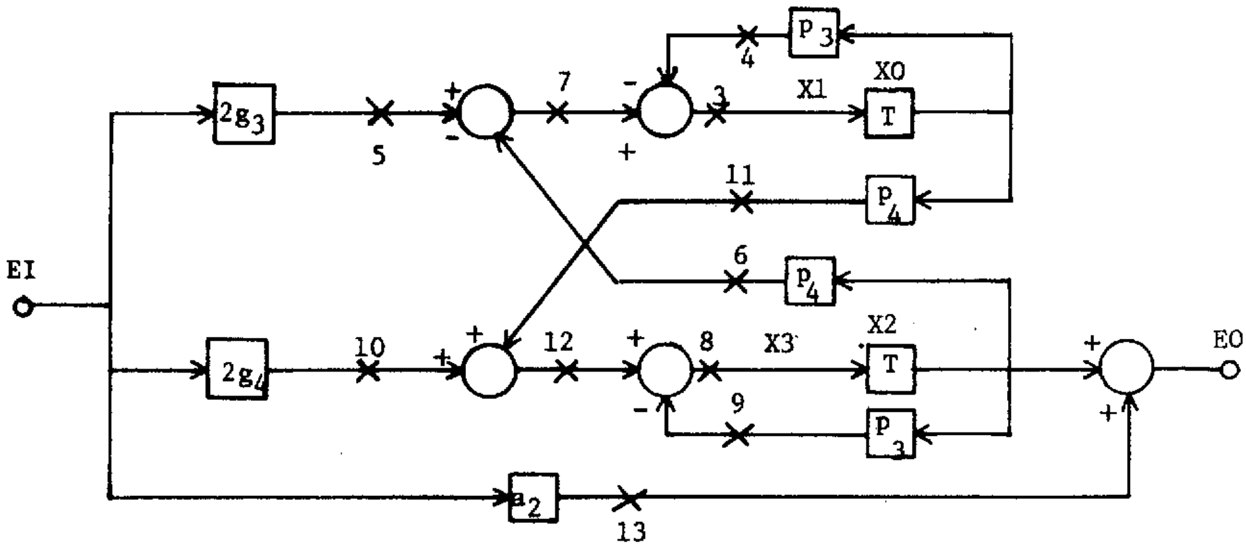
error point	signal at error point	transfer function to output
3	$X2(z)$	$H_1(z)$
4	$b_1 z^{-1} X_2(z)$	$-H_1(z)$
5	$[a_0 - a_2 b_0 - b_1(a_1 - a_2 b_1)]EI(z)$	$H_1(z)$
6	$b_0[(a_1 - a_2 b_1) z^{-1} EI(z) + z^{-2} X_2(z)]$	$-H_1(z)$
7	$4 + 6$	$-H_1(z)$
8	$(a_1 - a_2 b_1) EI(z)$	$H_2(z)$
9	$z^{-1} X2(z) + (a_1 - a_2 b_1) EI(z)$	$H_2(z)$
10	$a_2 EI(z)$	$H_3(z)$

Figure 3-7. Standard form.



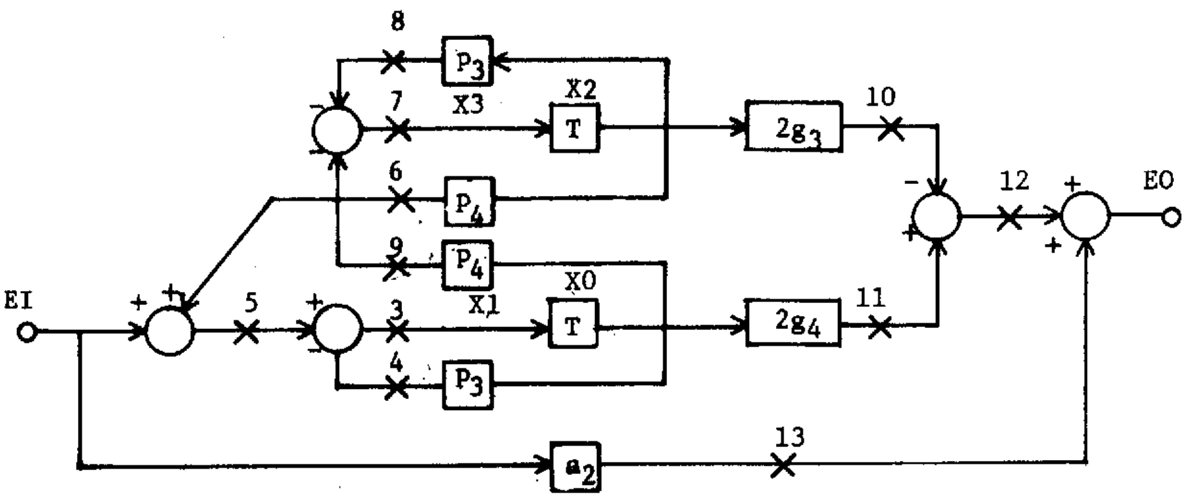
error point	signal at error point	transfer function to output
3	$g_1 EI(z)$	$H_1(z)$
4	$X1(z)$	$H_1(z)$
5	$p_1 z^{-1} X1(z)$	$-H_1(z)$
6	$g_2 EI(z)$	$H_2(z)$
7	$X3(z)$	$H_2(z)$
8	$p_2 z^{-1} X3(z)$	$-H_2(z)$
9	$z^{-1}X1(z) + z^{-1}X3(z)$	$H_3(z)$
10	$a_2 EI(z)$	$H_3(z)$

Figure 3-8. Parallel form.



error point	signal at error point	transfer function to output
3	$X1(z)$	$H_1(z)$
4	$p_3 z^{-1} Z1(z)$	$-H_1(z)$
5	$2g_3 EI(z)$	$H_1(z)$
6	$p_4 z^{-1} X3(z)$	$-H_1(z)$
7	5 - 6	$H_1(z)$
8	$X3(z)$	$H_2(z)$
9	$p_3 z^{-1} X3(z)$	$-H_2(z)$
10	$2g_4 EI(z)$	$H_2(z)$
11	$p_4 z^{-1} X1(z)$	$H_2(z)$
12	10 + 11	$H_2(z)$
13	$a_2 EI(z)$	$H_3(z)$

Figure 3-9. XI form.



error point	signal at error point	transfer function to output
3	$X1(z)$	$H_1(z)$
4	$p_3 z^{-1} X1(z)$	$-H_1(z)$
5	$EI(z) + P_4 z^{-1} X3(z)$	$H_1(z)$
6	$p_4 z^{-1} X3(z)$	$H_1(z)$
7	$X3(z)$	$H_2(z)$
8	$p_3 z^{-1} X3(z)$	$-H_2(z)$
9	$p_4 z^{-1} X1(z)$	$H_2(z)$
10	$2g_3 z^{-1} X3(z)$	$H_3(z)$
11	$2g_4 z^{-1} X1(z)$	$H_3(z)$
12	10 + 11	$H_3(z)$
13	$a_2 EI(z)$	$H_3(z)$

Figure 3-10. XII form.

Higher Order Filters

In the previous sections, it was assumed that the digital filter was of at most second order. In this section it will be shown that the program given in the appendix is applicable to higher order filters.

Consider, for example, a fourth-order filter, with all poles complex. This filter should be realized as two second-order sections, as shown in Figure 3-11, because of coefficient sensitivities. There are two questions to be answered with respect to this filter. First, which section should be placed first, and second, which programing form should be used for each section? To answer these questions, four different computer runs must be made. The filter section $D_1(z)$ must

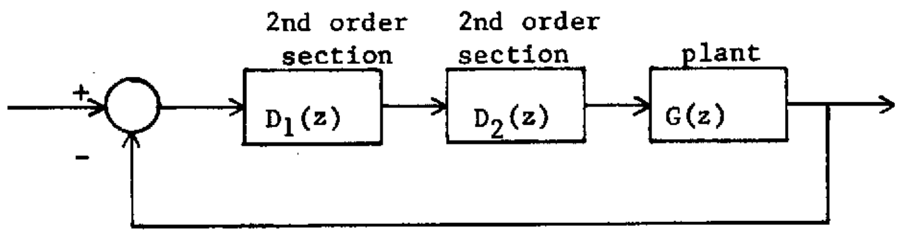


Figure 3-11. Digital control system.

be simulated in the program first as shown in Figure 3-11, and then with its position reversed with that of $D_2(z)$. Then the same simulations must be made with $D_2(z)$. An examination of the results of these runs will indicate not only the placement of $D_1(z)$ and $D_2(z)$, but also the filter forms required. For the system of Figure 3-11, an addition subroutine must be added to the program to simulate the additional filter section.

Example

As an example, consider the system of

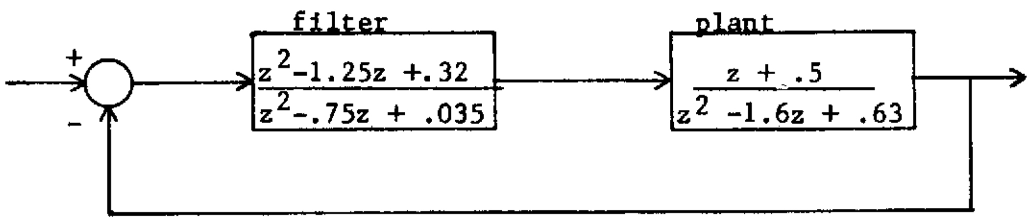


Figure 3-12. System for example

Figure 3-12. This system is simulated in the program of Appendix B, and the predicted errors from quantization were calculated. To check these results, the system was simulated with first the filter in single precision and the remainder of the system in extended precision and then with the entire system in extended precision on an IBM 360/50 computer. The difference in the outputs of these two simulations is then the system quantization error. The IBM 360/50 computer is a base 16 machine, has 24 bits in the fraction, and used tuncation quantization. Thus, in (2-18), t is equal to 24 and k is equal to 4. The maximum error at the point of quantization is

$$\delta_m = 2^{-20} \approx 0.95 \times 10^{-6} \qquad (3\text{-}24)$$

Consider first the modified canonical form. The predicted rms error for the system was found to be 0.80×10^{-6}, with k of (3-21) equal to 100 and the input a step function.[1] Sixteen different simulations with step-function inputs yielded a total rms error of 0.33×10^{-6}, which is lower than the predicted result. However, it is to be recalled that probably no error will occur in the addition of two numbers if the exponents of the two numbers are equal. Since the IBM 360/50 is a base 16 computer, two numbers can be different by a factor of almost 16 and still have the same exponent. To approximately compensate for this effect, the errors from all additions were set to zero, and the predicted rms error was calculated to be 0.28×10^{-6}. It is then seen that additions do contribute very little to the total system errors.

Next the canonical form was used in the simulations, with the resultant rms error of 0.20×10^{-6}. With addition errors included, the predicted rms error was 0.27×10^{-6}; and without additional errors, the predicted rms error was 0.24×10^{-6}.

With the standard form, the result of the simulations was an rms error of 0.135×10^{-6}. With addition errors included, the predicted rms error was 0.21×10^{-6}; and without addition errors, the predicted rms error was 0.125×10^{-6}.

From the above results it is seen that the developed technique yields accurate results. However, problems do occur with the errors

---

[1] Computer run time for the program of appendix B was approximately 30 seconds, using the WATFIV version of FORTRAN.

caused by addition. In a base 2 computer, these errors would be much more likely to occur, since the exponents of the numbers are not as likely to be equal. Additional research is needed to develop a better technique for the inclusion of errors due to addition.

IV Conclusions

In this report the problem of floating-point quantization errors in a digital control system is investigated. A technique is developed which yields the maximum possible system output error and the mean-square system output error caused by quatization in a digital controller operating with floating-point arithmetic. A deterministic system input is assumed. The technique is implemented by a digital computer program, which is based on a simulation of the control system. The program requires as input the number base of the digital controller, the digital controller coefficients, the system input function, and the sampling period for which the errors are desired. The system plant must be simulated in a subroutine of the program. As an output, the program gives the maximum error and the mean-square error in the system output for the digital controller realized by nine different programming forms. Thus the programming form can be chosen for the controller that yields the smallest errors.

REFERENCES

(1) J. H. Wilkerson, Rounding Errors in Algebraic Processes, Englewood Cliffs, N.J.: Prentice Hall, 1963.

(2) C. L. Phillips, "System Quantization Errors in Digital Control Systems," 22 Technical Report, Contract NAS8-11274, Auburn University, Auburn, Alabama, February, 1972.

(3) B. Liu and T. Kaneko, "Error Analysis of Digital Filters Realized with Floating-Point Arithmetic," Proc. of IEEE, Vol. 57, October, 1969, pp. 1735-1747.

(4) E.P.F. Kan and J.K. Aggarwal, "Error Analysis of Digital Filter Employing Floating-Point Arithmetic," IEEE Trans. on Circuit Theory, Vol. CT-18, November, 1971, pp. 678-686.

BIBLIOGRAPHY

T. Kaneko and B. Liu, "Effect of Coefficient Rounding in Floating-Point Digital Filters," IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-7, September, 1971, pp. 995-1003.

B. Liu, "Effect of Finite Word Length on the Accuracy of Digital Filters - A Review, "IEEE Trans. on Circuit Theory, Vol. CT-18, November, 1971, pp. 670-677.

I. W. Sandberg, "Floating-Point Roundoff Accumulation in Digital-Filter Realizations, " Bell System Technical Journal,Vol. 46, October, 1967, pp. 1775-1791.

C. Weinstein and A.V. Oppenheim, "A Comparison of Roundoff Noise in Floating-Point and Fixed-Point Digital Filter Realizations," Proc. of IEEE, Vol. 57, June, 1969, pp. 1181-1183.

APPENDIX A

In this appendix the first and second moments of the probability density functions for both roundoff and truncation errors will be derived.

Consider first the roundoff density function

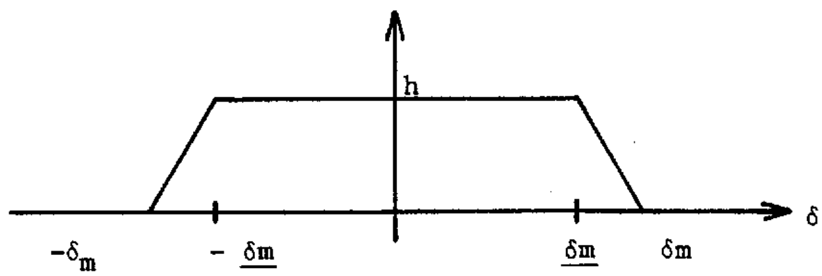


Figure A-1. Roundoff quantization

given in Figure A-1. Let $r = 2^k$ be the base of the computer. The first moment is zero. The second moment is

$$m_2 = \int_{-\infty}^{\infty} \delta^2 f(\delta)d\delta = 2\int_0^{\delta m} \delta^2 f(\delta)d\delta \tag{A-1}$$

Or

$$m_2 = 2\int_0^{\frac{\delta m}{r}} h\,\delta^2 d\delta + 2\int_{\frac{\delta m}{r}}^{\delta m} \frac{-h\ r}{\delta_m(r-1)} (\delta - \delta_m)\delta^2 d\delta \tag{A-2}$$

Since the area under the curve in Figure A-1 is equal to unity, then

$$h = \frac{r}{(r+1)_{\delta m}} \tag{A-3}$$

Substituting (A-3) into (A-2) and evaluating the integrals, we find that

$$m_2 = \frac{\delta_m^{\,2}(r^3 + r^2 + r + 1)}{6r^2(r + 1)} \qquad \text{(A-4)}$$

Consider now the truncation density function given in Figure A-2.

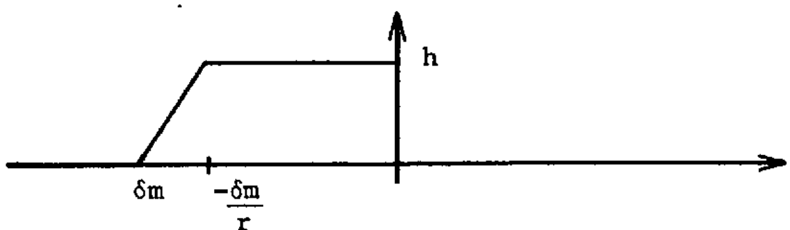


Figure A-2. Truncation quantization

The first moment is given by

$$m_1 = \int_{-\infty}^{\infty} \delta f(\delta)\, d\delta = \int_0^{-\frac{\delta_m}{r}} \frac{-h\, r\, \delta}{\delta_m(r-1)} (\delta - \delta_m)\, d\delta, \qquad \text{(A-5)}$$

where

$$h = \frac{2r}{(r+1)\delta_m} \qquad \text{(A-6)}$$

Evaluation of (A-5) yields

$$m_1 = \frac{\delta m(r^2 + r + 1)}{3r(r + 1)} \qquad \text{(A-7)}$$

The second moment is given by

$$m_2 = \int_0^{-\frac{\delta_m}{r}} \delta^2 h\, d\delta + \int_{-\frac{\delta_m}{r}}^{-\delta_m} \frac{-h\, r\, \delta^2}{\delta_m(r-1)} (\delta - \delta_m)\, d\delta \qquad \text{(A-8)}$$

Evaluation of (A-8) yields

$$m_2 = \frac{\delta_m{}^2(r^3 + r^2 + r + 1)}{6r^2\ (r + 1)} \tag{A-9}$$

Note that (A-4) and (A-9) are the same. However, δ_m for roundoff is one-half that for truncation.

APPENDIX B

TABLE B-1

SYMBOLS IN PROGRAM

Symbol	Description
BK	base of computer
BSR	use to determine if poles of filter are complex
E1(I)	signal amplitude at quantization point
EE(I)	$\sum x(N1-i)h(i)$
EER	used in calculating rms errors
EI	input to filter
EI1	used in finding impulse response
EMM	total maximum error for form
EM(I)	$\sum \lvert x(N1-i)h(i) \rvert$
EO	output of filter
ERMS	total RMS error for form
ERR	used to calculate total RMS error for form
ER(I)	$\sum [x(N1-i)h(i)]^2$
F(I)	$x(N1-i)h(i)$
IO	IO=0 zeros errors from input quantizer
I1	I1=0 zeros errors from output quantizer
JJ	number count for simulations for each form
JJJ	number of simulations for each form
JT	JT=0 for roundoff, JT=1 for truncation
KK	used to choose form
KK1	initial value of KK

Table B-1
(continued)

KK2	Final value of KK
MM	stops program at correct point for filter with complex poles
N1	number of iterations in each simulation
R	system input
XM	first moment (expected value) for errors
XM2	used to calculate rms errors
XM3	second moment for errors
YO	system output

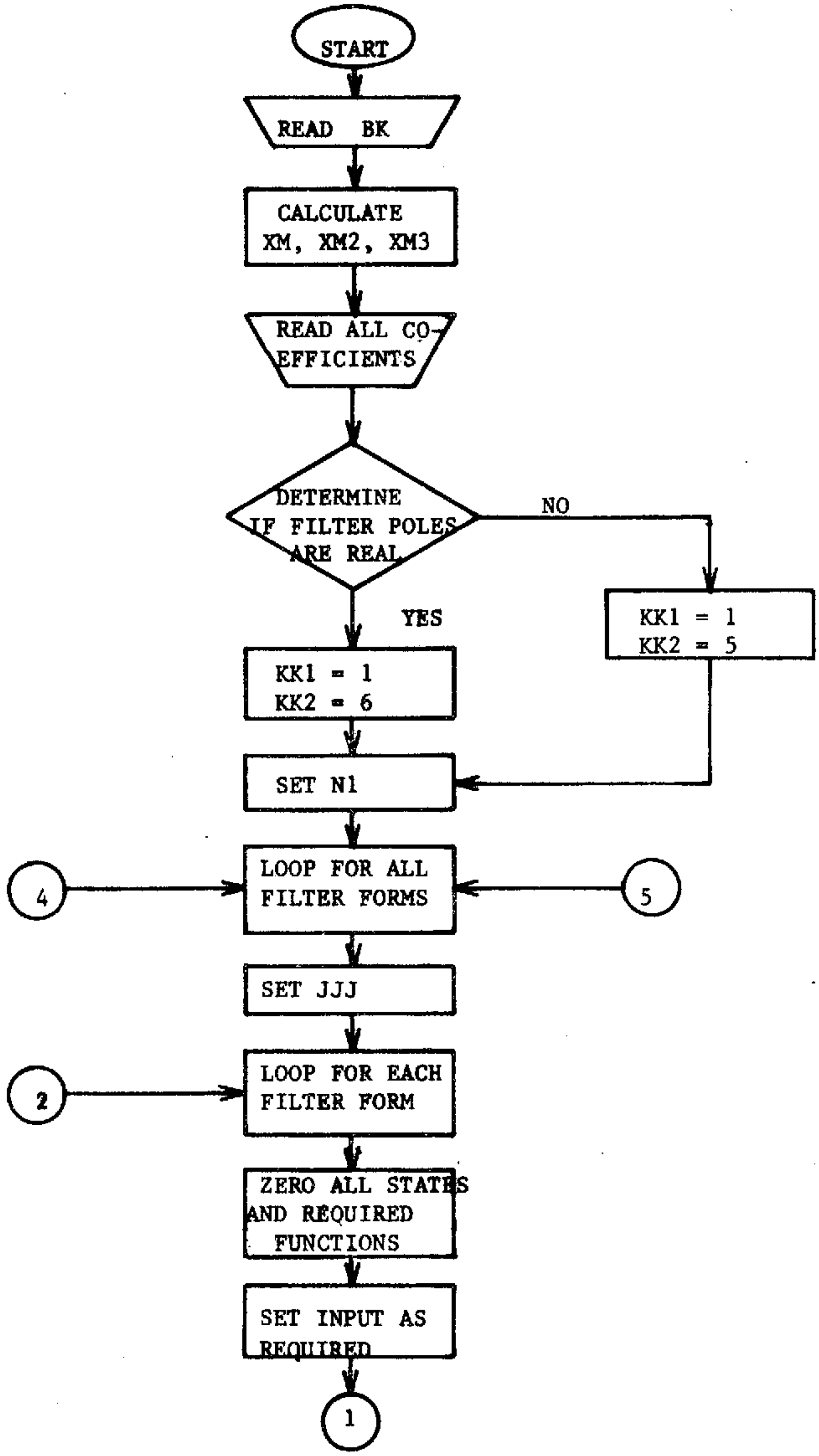


Figure B-1. Flow-chart for program

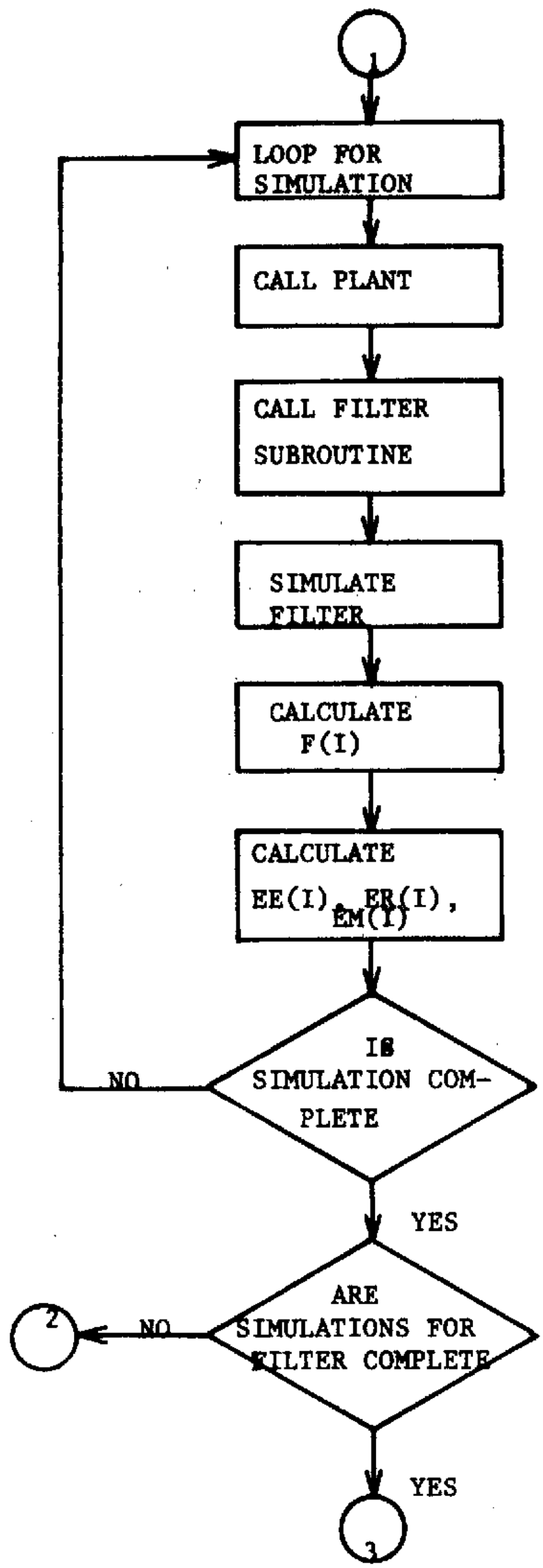


Figure B-1 (continued)

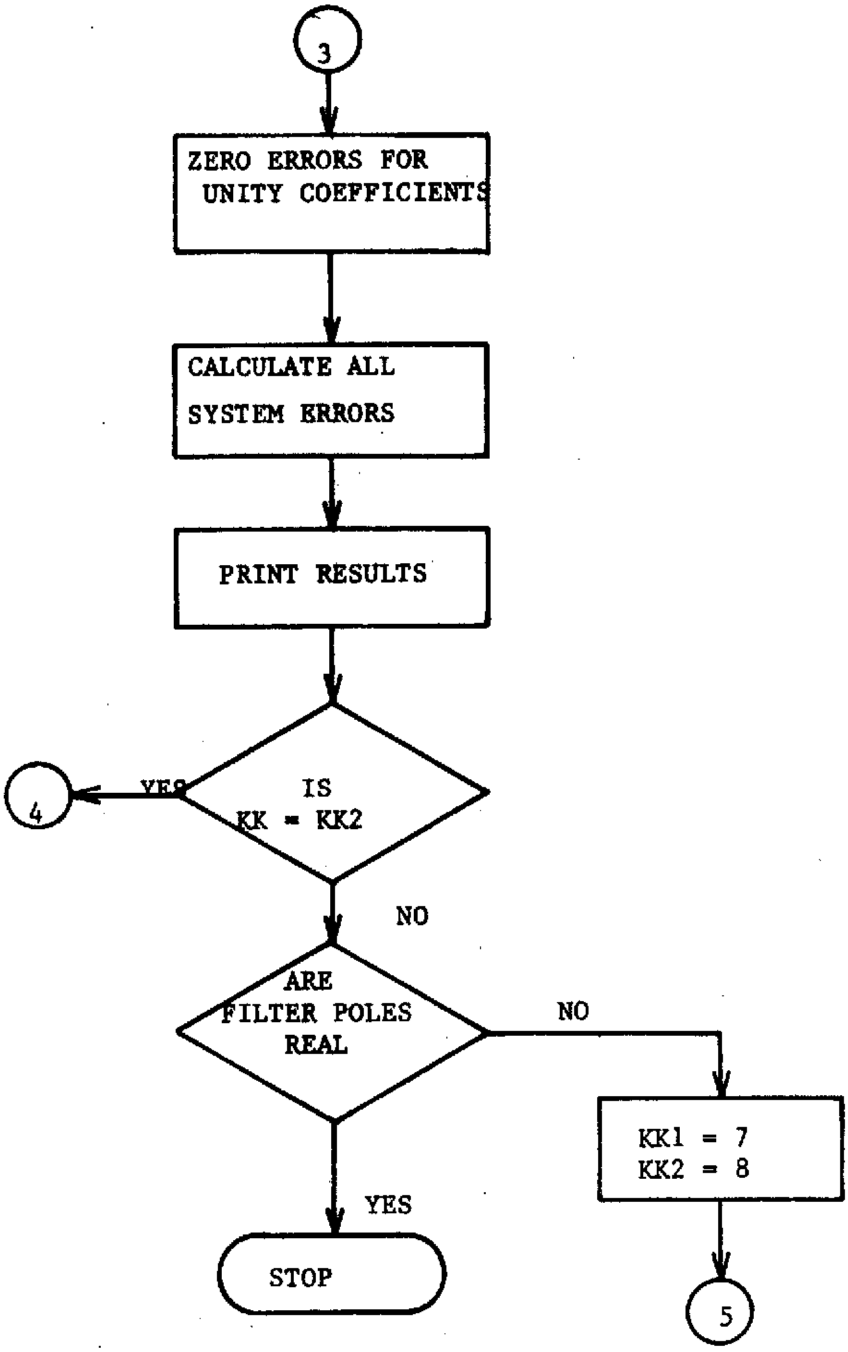


Figure B-1 (continued)

```
      DIMENSION E1(150),E2(150),E3(150),F(13),ER(13),EM(13),EE(13)
      COMMON/C1/A2,A1,A0,B1,B0,X0,X1,X2,X3,X4,X5,EI1,EI2,EI3
      COMMON/C2/E1,E2,E3,F,ER,EM,EE,II,JJ,KK,N1,Y0,JT,XM,XM2
      COMMON/C3/P1,P2,P3,P4,G1,G2,G3,G4
      COMMON/C4/A22,A11,A00,B22,B11,B00,G,X31,X41,X51
C     JT=0 FOR ROUNDOFF,JT=1 FOR TRUNCATION
      JT = 1
      JT = 0
C     BASE OF COMPUTER IS BK
      BK = 16.
C     XM IS FIRST MOMENT OF ERROR SOURCE
C     XM3 IS SECOND MOMENT OF ERROR SOURCE
C     XM2 IS USED IN CALCULATING RMS ERRORS
      XM=(BK**2+3.*BK+3.)/(BK**2+3.*BK+2.)/3.
      XM3=(BK**3+4.*BK**2+6.*BK+4.)/(BK**3+4.*BK**2+5.*BK+2.)/6.
      IF(JT.NE.1) XM=0.
      XM2 = XM3 - XM**2
C     COEFFICIENTS OF PLANT
      G = 1.0
      A22 = 1.
      A11 = 0.5
      A00 = 0.
      B22 = -1.6
      B11 = 0.63
      B00 = 0.
C     COEFFICIENTS OF FILTER
      A0 = 0.315
      A1 = -1.25
      A2=1.0
      B0 = 0.035
      B1 = -0.75
      MM = 1
C     TO DETERMINE IF FILTER POLES ARE COMPLEX
      BSR = (B1**2)/4. - B0
  600 IF(BSR.GT.0.) GO TO 601
C     FORMS FOR COMPLEX POLES
      KK1 = 1
      KK2 = 5
      GO TO 603
  601 CONTINUE
C     FORMS FOR REAL POLES
      BSSR = SQRT(BSR)
      P1 = B1/2. + BSSR
      P2 = B1/2. - BSSR
      G1 = (A1*P1 - A2*P1**2 - A0)/(P1 - P2)
      G2 = (A0-A1*P2 + A2*P2**2)/(P1-P2)
      KK1 = 1
      KK2 = 6
```

```
      GO TO 603
  604 IF(BSR.GE.0.) GO TO 606
      MM = 2
C     FORMS FOR COMPLEX POLES
      P3 = B1/2.
      P4 = SQRT(-BSR)
      G4 = A1/2. - P3*A2
      G3 = (A0 - A2*B0 - 2.*G4*P3)/(2.*P4)
      KK1 = 7
      KK2 = 8
  603 CONTINUE
C     N1 IS TOTAL ITERATIONS FOR SIMULATIONS
      N1 = 100
C     KK USED TO CHOOSE FILTER FORM
      DO 501 KK=KK1,KK2
C     JJJ SETS NUMBER OF SIMULATIONS REQUIRED FOR EACH FORM
      JJJ = 4
      IF(KK.EQ.2) JJJ=3
      IF(KK.EQ.3) JJJ=2
      IF(KK.EQ.4) JJJ=3
C     JJ COUNTS THE NUMBER OF SIMULATIONS FOR EACH FORM
      DO 50 JJ=1,JJJ
      Y0 = 0.
C     ZERO INITIAL CONDITIONS FOR FILTER
      X0 = 0.
      X1=0.
      X2=0.
      X3=0.
      X4=0.
      X5=0.
C     ZERO INITIAL CONDITIONS FOR PLANT
      X31 = 0.
      X41 = 0.
      X51 = 0.
C     EI1,EI2,EI3 USED TO CALCULATE IMPULSE RESPONSES
      EI1 = 0.
      EI2 = 0.
      EI3 = 0.
C     E1(I),E2(I),E3(I) ARE SIGNALS AT POINTS IN FILTER
      E1(1) = 0.
      E1(2) = 0.
      E2(1) = 0.
      E2(2) = 0.
      E3(1) = 0.
      E3(2) = 0.
C     JJ=2 YIELDS FIRST IMPULSE RESPONSE
C     JJ=3 YIELDS SECOND IMPULSE RESPONSE
C     JJ=4 YIELDS THIRD IMPULSE RESPONSE
```

```
      IF(JJ.EQ.2) EI1 = 1.
      IF(JJ.EQ.3) EI2 = 1.
      IF(JJ.EQ.4) EI3=1.
C     JJ=1 YIELDS SIGNAL AMPLITUDES IN FILTER
      IF(JJ.GT.2) GO TO 102
      IF(KK.GT.1) GO TO 10
C     ER(I) USED TO CALCULATE RMS ERROR
C     EE(I) USED TO CALCULATE RMS ERROR
C     EM(I) USED TO CALCULATE MAXIMUM ERROR
      DO 1 I=1,2
      EE(I) = 0.
      ER(I) = 0.
    1 EM(I) = 0.
   10 DO 2 I=3,13
      EE(I) = 0.
      ER(I) = 0.
    2 EM(I) = 0.
C     R IS SYSTEM INPUT
  102 R = 0.
      IF(JJ.EQ.1) R = 1.
      IF(JJ.EQ.2) N1 = N1 - 2
      DO 40 II=1,N1
      CALL PLANT(BI,EO,R)
   40 CONTINUE
   50 CONTINUE
  501 CONTINUE
      IF(MM.EQ.1) GO TO 604
  606 CONTINUE
      STOP
      END
```

```
      SUBROUTINE PLANT(EI,EO,R)
C     PLANT OF CONTROL SYSTEM
      DIMENSION E1(150),E2(150),E3(150),F(13),ER(13),EM(13),EE(13)
      COMMON/C2/E1,E2,E3,F,ER,EM,EE,II,JJ,KK,N1,YO,JT,XM,XM2
      COMMON/C4/A22,A11,A00,B22,B11,B00,G,X31,X41,X51
C     YO IS PLANT OUTPUT
      YO = A00*X31 + A11*X41 + A22*X51
      EI = R - YO
C     EI IS FILTER INPUT
C     EO IS FILTER OUTPUT
      IF(KK.EQ.1) CALL FIL1(EI,EO)
      IF(KK.EQ.2) CALL FIL2(EI,EO)
      IF(KK.EQ.3) CALL FIL3(EI,EO)
      IF(KK.EQ.4) CALL FIL4(EI,EO)
      IF(KK.EQ.5) CALL FIL5(EI,EO)
      IF(KK.EQ.6) CALL FIL6(EI,EO)
      IF(KK.EQ.7) CALL FIL7(EI,EO)
      IF(KK.EQ.8) CALL FIL8(EI,EO)
      X61 = G*EO - B00*X31 - B11*X41 - B22*X51
      X31 = X41
      X41 = X51
      X51 = X61
      RETURN
      END
```

```
      SUBROUTINE FILI(EI,EO)
C     MODIFIED CANONICAL FORM
      DIMENSION E1(150),E2(150),E3(150),F(13),ER(13),EM(13),EE(13)
      COMMON/C1/A2,A1,A0,B1,B0,X0,X1,X2,X3,X4,X5,EI1,EI2,EI3
      COMMON/C2/E1,E2,E3,F,ER,EM,EE,II,JJ,KK,N1,YO,JT,XM,XM2
C     EI IS FILTER INPUT
C     EI1 USED TO CALCULATE FIRST IMPULSE RESPONSE
C     EI2 USED TO CALCULATE SECOND IMPULSE RESPONSE
C     EI3 USED TO CALCULATE THIRD IMPULSE RESPONSE
      EI = EI + EI3
      X2 = EI - B1*X1 - B0*X0 + EI1
C     E1(I) IS SIGNAL AT A POINT IN FILTER
C     E2(I) IS SIGNAL AT A POINT IN FILTER
      IF(JJ.EQ.1) E1(II+2)=X2
      IF(JJ.EQ.1) E2(II+2) = EI
C     EO IS FILTER OUTPUT
      EO = A2*EI+(A1-B1*A2)*X1+(A0-B0*A2)*X0+EI2
      EI1 = 0.
      EI2 = 0.
      EI3 = 0.
      X0 = X1
      X1 = X2
      IF(JJ.EQ.1) GO TO 201
      IF(JJ.EQ.3) GO TO 202
      IF(JJ.EQ.4) GO TO 203
C     F(J) IS X(N1-I)H(I)
C     E1 IS X(N1-I)
C     YO IS H(I)
      F(3) = YO*E1(N1+3-II)
      F(4) = YO*E1(N1+2-II)*(-B1)
      F(5) = YO*E1(N1+1-II)*(-B0)
      F(6) = F(4) + F(5)
C     EE(I) USED TO CALCULATE RMS ERROR
C     ER(I) USED TO CALCULATE RMS ERROR
C     EM(I) USED TO CALCULATE MAXIMUM ERROR
      DO 2 I=3,6
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
    2 EM(I) = EM(I) + ABS(F(I))
      GO TO 201
  202 F(7) = YO*E2(N1+3-II)*A2
      F(8) = YO*E1(N1+2-II)*(A1-B1*A2)
      F(9) = YO*E1(N1+1-II)*(A0-B0*A2)
      F(10) = F(7) + F(8)
      F(2) = F(9) + F(10)
      DO 3 I=7,10
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
```

```
3     EM(I) = EM(I) + ABS(F(I))
      EE(2) = EE(2) + F(2)
      ER(2) = ER(2) + F(2)**2
      EM(2) = EM(2) + ABS(F(2))
      GO TO 201
  203 F(1) = YO*E2(N1+3-II)
      EE(1) = EE(1) + F(1)
      ER(1) = ER(1) + F(1)**2
      EM(1) = EM(1) + ABS(F(1))
      IF(II.LT.N1) GO TO 201
C     SETS MULTIPLICATION ERROR TO ZERO IF COEFFICIENT=1
      IF(B1.EQ.1.) EE(4)=0.
      IF(B1.EQ.1.) EM(4)=0.
      IF(B1.EQ.1.) ER(4)=0.
      IF(B0.EQ.1.) EE(5)=0.
      IF(B0.EQ.1.) EM(5)=0.
      IF(B0.EQ.1.) ER(5)=0.
      IF(A2.EQ.1.) EM(7)=0.
      IF(A2.EQ.1.) EE(7)=0.
      IF(A2.EQ.1.) ER(7)=0.
      IF((A1-B1*A2).EQ.1.) EE(8)=0.
      IF((A1-B1*A2).EQ.1.) EM(8)=0.
      IF((A1-B1*A2).EQ.1.) ER(8)=0.
      IF((A0-B0*A2).EQ.1.) EE(9)=0.
      IF((A0-B0*A2).EQ.1.) EM(9)=0.
      IF((A0-B0*A2).EQ.1.) ER(9)=0.
      DO 50 I=1,10
   50 ER(I) = ER(I)*XM2 + EE(I)**2*XM**2
      PRINT 300
  300 FORMAT('-','FORM',23X,'MAX ERROR',7X,'RMS ERROR')
C     CALCULATES ERRORS FOR INPUT QUANTIZER
      ERM1 = SQRT(ER(1)*XM2)
C     CALCULATES ERRORS FOR OUTPUT QUANTIZER
      ERM2 = SQRT(ER(2)*XM2)
      PRINT 304,EM(1),ERM1
  304 FORMAT('-','INPUT',15X,2F16.5)
      PRINT 305,EM(2),ERM2
  305 FORMAT('-','OUTPUT',14X,2F16.5)
C     SET I1=0 TO EXCLUDE INPUT ERROR POINT FROM TOTAL ERROR
      I1 = 1
      I1 = 0
C     SET IO=0 TO EXCLUDE OUTPUT ERROR POINT FROM TOTAL ERROR
      IO = 0
      IO = 1
      IF(I1.EQ.0) EE(1)=0.
      IF(I1.EQ.0) EM(1)=0.
      IF(I1.EQ.0) ER(1)=0.
      IF(IO.EQ.0) EE(2)=0.
```

```
      IF(IO.EQ.0) EM(2)=0.
      IF(IO.EQ.0) ER(2)=0.
C     EER IS USED IN CALCULATNG RMS ERROR FOR TRUNCATION
C     EMM IS TOTAL MAXIMUM ERROR FOR FILTER FORM
      EER = 0.
      EMM = 0.
      ERR = 0.
      DO 10 I=1,10
      EMM = EMM+ EM(I)
   10 ERR = ERR + ER(I)
      IF(JT.NE.1) GO TO 40
      DO 30 I=1,9
      K = I+1
      DO 30 J=K,10
   30 EER = EER + EE(I)*EE(J)
      EER = 2.*EER
C     ERMS IS TOTAL RMS ERROR FOR FILTER FORM
   40 ERMS = SQRT(ERR + EER*XM**2)
      PRINT 302,EMM,ERMS
  302 FORMAT('-','MODIFIED CANONICAL',2X,2F16.5)
  201 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FIL2(EI,EO)
C     CANONICAL FORM
      DIMENSION E1(150),E2(150),E3(150),F(13),ER(13),EM(13),EE(13)
      COMMON/C1/A2,A1,A0,B1,B0,X0,X1,X2,X3,X4,X5,EI1,EI2,EI3
      COMMON/C2/E1,E2,E3,F,ER,EM,EE,II,JJ,KK,N1,Y0,JT,XM,XM2
      X2 = EI - B1*X1 - B0*X0 + EI1
      EI1 = 0.
      IF(JJ.EQ.1) E1(II+2) = X2
      EO = A2*X2 + A1*X1 + A0*X0 + EI2
      EI2 = 0.
      X0 = X1
      X1 = X2
      IF(JJ.EQ.1) GO TO 201
      IF(JJ.EQ.3) GO TO 202
      F(3) = Y0*E1(N1+3-II)
      F(4) = Y0*E1(N1+2-II)*(-B1)
      F(5) = Y0*E1(N1+1-II)*(-B0)
      F(6) = F(4) + F(5)
      DO 2 I=3,6
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
    2 EM(I) = EM(I) + ABS(F(I))
      GO TO 201
  202 F(7) = Y0*E1(N1+3-II)*A2
      F(8) = Y0*E1(N1+2-II)*A1
      F(9) = Y0*E1(N1+1-II)*A0
      F(10) = F(7) + F(8)
      DO 3 I=7,10
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
    3 EM(I) = EM(I) + ABS(F(I))
      IF(II.LT.N1) GO TO 201
      IF(B1.EQ.1.) EE(4) = 0.
      IF(B1.EQ.1.) EM(4) = 0.
      IF(B1.EQ.1.) ER(4) = 0.
      IF(B0.EQ.1.) EE(5) = 0.
      IF(B0.EQ.1.) EM(5) = 0.
      IF(B0.EQ.1.) ER(5) = 0.
      IF(A2.EQ.1.) EE(7) = 0.
      IF(A2.EQ.1.) EM(7) = 0.
      IF(A2.EQ.1.) ER(7) = 0.
      IF(A1.EQ.1.) EE(8) = 0.
      IF(A1.EQ.1.) EM(8) = 0.
      IF(A1.EQ.1.) ER(8) = 0.
      IF(A0.EQ.1.) EE(9) = 0.
      IF(A0.EQ.1.) EM(9) = 0.
      IF(A0.EQ.1.) ER(9) = 0.
      DO 50 I=1,10
```

```
50  ER(I) = ER(I)*XM2 + EE(I)**2*XM**2
    EER = 0.
    EMM = 0.
    ERR = 0.
    DO 10 I=1,10
    EMM = EMM + EM(I)
10  ERR = ERR + ER(I)
    IF(JT.NE.1) GO TO 40
    DO 30 I=1,9
    K = I+1
    DO 30 J=K,10
30  EER = EER + EE(I)*EE(J)
    EER = 2.*EER
40  ERMS = SQRT(ERR + EER*XM**2)
    PRINT 302,EMM,ERMS
302 FORMAT('-','CANONICAL',11X,2F16.5)
201 CONTINUE
    RETURN
    END
```

```
      SUBROUTINE FIL3(EI,EO)
C     MODIFIED DIRECT FORM
      DIMENSION E1(150),E2(150),E3(150),F(13),ER(13),EM(13),EE(13)
      COMMON/C1/A2,A1,A0,B1,B0,X0,X1,X2,X3,X4,X5,EI1,EI2,EI3
      COMMON/C2/E1,E2,E3,F,ER,EM,EE,II,JJ,KK,N1,YO,JT,XM,XM2
      X2 = A2*EI + A1*X5 + A0*X4 - X3+EI1
      EI1 = 0.
      IF(JJ.EQ.1) E1(II+2) = X2
      IF(JJ.EQ.1) E2(II+2)=EI
      EO = X2
      X3 = B1*X2 + BO*X1
      X1 = X2
      X4 = X5
      X5 = EI
      IF(JJ.EQ.1) GO TO 201
      F(3) = YO*E1(N1+3-II)
      F(4) = YO*E1(N1+2-II)*(-B1)
      F(5) = YO*E1(N1+1-II)*(-B0)
      F(6) = F(4) + F(5)
      F(7) = YO*E2(N1+3-II)*A2
      F(8) = YO*E2(N1+2-II)*A1
      F(9) = YO*E2(N1+1-II)*A0
      F(10) = F(7) + F(8)
      F(11) = F(9) + F(10)
      DO 2 I=3,11
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
    2 EM(I) = EM(I) + ABS(F(I))
      IF(II.LT.N1) GO TO 201
      IF(B1.EQ.1.) EE(4)=0.
      IF(B1.EQ.1.) EM(4)=0.
      IF(B1.EQ.1.) ER(4)=0.
      IF(B0.EQ.1.) EE(5)=0.
      IF(B0.EQ.1.) EM(5)=0.
      IF(B0.EQ.1.) ER(5)=0.
      IF(A2.EQ.1.) EE(7)=0.
      IF(A2.EQ.1.) EM(7)=0.
      IF(A2.EQ.1.) ER(7)=0.
      IF(A1.EQ.1.) EE(8)=0.
      IF(A1.EQ.1.) EM(8)=0.
      IF(A1.EQ.1.) ER(8)=0.
      IF(A0.EQ.1.) EE(9)=0.
      IF(A0.EQ.1.) EM(9)=0.
      IF(A0.EQ.1.) ER(9)=0.
      DO 50 I=1,10
   50 ER(I) = ER(I)*XM2 + EE(I)**2*XM**2
      EER = 0.
      EMM = 0.
```

```
      ERR = 0.
      DO 10 I=1,11
      EMM = EMM + EM(I)
 10   ERR = ERR + ER(I)
      IF(JT.NE.1) GO TO 40
      DO 30 I=1,10
      K = I+1
      DO 30 J=K,11
 30   EER = EER + EE(I)*EE(J)
      EER = 2.*EER
 40   ERMS = SQRT(ERR + EER*XM**2)
      PRINT 302,EMM,ERMS
 302  FORMAT('-','MODIFIED DIRECT',5X,2F16.5)
 201  CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FIL4(EI,EO)
C     MODIFIED STANDARD FORM
      DIMENSION E1(150),E2(150),E3(150),F(13),ER(13),EM(13),EE(13)
      COMMON/C1/A2,A1,A0,B1,B0,X0,X1,X2,X3,X4,X5,EI1,EI2,EI3
      COMMON/C2/E1,E2,E3,F,ER,EM,EE,II,JJ,KK,N1,Y0,JT,XM,XM2
      EO = A2*EI + X0
      X2 = A0*EI - B0*EO + EI1
      X3 = A1*EI + X1 - B1*EO + EI2
      EI1 = 0.
      EI2 = 0.
      IF(JJ.EQ.1) E1(II+2) = X2
      IF(JJ.EQ.1) E2(II+2) = EI
      IF(JJ.EQ.1) E3(II+2) = X1 - B1*EO
      X0 = X3
      X1 = X2
      IF(JJ.EQ.1) GO TO 201
      IF(JJ.EQ.3) GO TO 202
      F(3) = Y0*E1(N1+3-II)
      F(4) = Y0*E2(N1+3-II)*A0
      F(5) = Y0*(A1*E2(N1+2-II)+E3(N1+2-II)+A2*E2(N1+3-II))*(-B0)
      DO 2 I=3,5
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
    2 EM(I) = EM(I) + ABS(F(I))
      GO TO 201
  202 F(6) = Y0*E3(N1+3-II)
      F(7) = Y0*E2(N1+3-II)*A1
      F(8) = Y0*E2(N1+2-II)*A2
      F(9) = Y0*(E3(N1+1-II) + A1*E2(N1+1-II)+A2*E2(N1+2-II))
      F(10) = Y0*(A1*E2(N1+2-II)+E3(N1+2-II)+A2*E2(N1+3-II))*(-B1)
      F(11) = F(6) + F(7)
      DO 3 I=6,11
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
    3 EM(I) = EM(I) + ABS(F(I))
      IF(II.LT.N1) GO TO 201
      IF(A0.EQ.1.) EE(4)=0.
      IF(A0.EQ.1.) EM(4)=0.
      IF(A0.EQ.1.) ER(4)=0.
      IF(B0.EQ.1.) EE(5)=0.
      IF(B0.EQ.1.) EM(5)=0.
      IF(B0.EQ.1.) ER(5)=0.
      IF(A1.EQ.1.) EE(7)=0.
      IF(A1.EQ.1.) EM(7)=0.
      IF(A1.EQ.1.) ER(7)=0.
      IF(A2.EQ.1.) EE(8)=0.
      IF(A2.EQ.1.) EM(8)=0.
      IF(A2.EQ.1.) ER(8)=0.
```

```
      IF(B1.EQ.1.) EE(10)=0.
      IF(B1.EQ.1.) EM(10)=0.
      IF(B1.EQ.1.) ER(10)=0.
      DO 50 I=1,11
50    ER(I) = ER(I)*XM2 + EE(I)**2*XM**2
      EER = 0.
      EMM = 0.
      ERR = 0.
      DO 10 I=1,11
      EMM = EMM + EM(I)
10    ERR = ERR + ER(I)
      IF(JT.NE.1) GO TO 40
      DO 30 I=1,10
      K = I+1
      DO 30 J=K,11
30    EER = EER + EE(I)*EE(J)
      EER = 2.*EER
40    ERMS = SQRT(ERR + EER*XM**2)
      PRINT 302,EMM,ERMS
302   FORMAT('-','MODIFIED STANDARD',3X,2F16.5)
201   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FIL5(EI,EO)
C     STANDARD FORM
      DIMENSION E1(150),E2(150),E3(150),F(13),ER(13),EM(13),EE(13)
      COMMON/C1/A2,A1,A0,B1,B0,X0,X1,X2,X3,X4,X5,EI1,EI2,EI3
      COMMON/C2/E1,E2,E3,F,ER,EM,EE,II,JJ,KK,N1,YO,JT,XM,XM2
      X2 = (A0-A2*B0-B1*(A1-A2*B1))*EI-B1*X1-B0*X3+EI1
      X4 = X1+(A1-A2*B1)*EI+EI2
      EO = X3+A2*EI+EI3
      EI1 = 0.
      EI2 = 0.
      EI3 = 0.
      IF(JJ.EQ.1) E1(II+2)=X2
      IF(JJ.EQ.1) E2(II+2)=EI
      X1 = X2
      X3 = X4
      IF(JJ.EQ.1) GO TO 201
      IF(JJ.EQ.3) GO TO 202
      IF(JJ.EQ.4) GO TO 203
      F(3) = YO*E1(N1+3-II)
      F(4) = YO*E1(N1+2-II)*(-B1)
      F(5) = YO*E2(N1+3-II)*(A0-A2*B0-B1*(A1-A2*B1))
      F(6) = YO*((A1-A2*B1)*E2(N1+2-II)+E1(N1+1-II))*(-B0)
      F(7) = F(4) + F(6)
      DO 2 I=3,7
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
  2   EM(I) = EM(I) + ABS(F(I))
      GO TO 201
  202 F(8) = YO*E2(N1+3-II)*(A1-A2*B1)
      F(9) = F(8) + YO*E1(N1+2-II)
      DO 3 I=8,9
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
  3   EM(I) = EM(I) + ABS(F(I))
      GO TO 201
  203 F(10) = YO*E2(N1+3-II)*A2
      EE(10) = EE(10) + F(10)
      ER(10) = ER(10) + F(10)**2
      EM(10) = EM(10) + ABS(F(10))
      IF(II.LT.N1) GO TO 201
      IF(B1.EQ.1.) EE(4)=0.
      IF(B1.EQ.1.) EM(4)=0.
      IF(B1.EQ.1.) ER(4)=0.
      IF((A0-A2*B0-B1*(A1-A2*B1)).EQ.1.) EE(5)=0.
      IF((A0-A2*B0-B1*(A1-A2*B1)).EQ.1.) EM(5)=0.
      IF((A0-A2*B0-B1*(A1-A2*B1)).EQ.1.) ER(5)=0.
      IF((A1-A2*B1).EQ.1.) EE(8)=0.
      IF((A1-A2*B1).EQ.1.) EM(8)=0.
```

```
      IF((A1-A2*B1).EQ.1.) ER(8)=0.
      IF(B0.EQ.1.) EE(6)=0.
      IF(B0.EQ.1.) EM(6)=0.
      IF(B0.EQ.1.) ER(6)=0.
      IF(A2.EQ.1.) EE(10)=0.
      IF(A2.EQ.1.) EM(10)=0.
      IF(A2.EQ.1.) ER(10)=0.
      DO 50 I=1,10
50    ER(I) = ER(I)*XM2 + EE(I)**2*XM**2
      EER = 0.
      EMM = 0.
      ERR = 0.
      DO 10 I=1,10
      EMM = EMM + EM(I)
10    ERR = ERR + ER(I)
      IF(JT.NE.1) GO TO 40
      DO 30 I=1,9
      K = I+1
      DO 30 J=K,10
30    EER = EER + EE(I)*EE(J)
      EER = 2.*EER
40    ERMS = SQRT(ERR + EER*XM**2)
      PRINT 302,EMM,ERMS
302   FORMAT('-','STANDARD',12X,2F16.5)
201   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FIL6(EI,EO)
C     PARALLEL FORM
      DIMENSION E1(150),E2(150),E3(150),F(13),ER(13),EM(13),EE(13)
      COMMON/C1/A2,A1,A0,B1,B0,X0,X1,X2,X3,X4,X5,EI1,EI2,EI3
      COMMON/C2/E1,E2,E3,F,ER,EM,EE,II,JJ,KK,N1,YO,JT,XM,XM2
      COMMON/C3/P1,P2,P3,P4,G1,G2,G3,G4
      X1 = G1*EI - P1*X0 + EI1
      X3 = G2*EI - P2*X2 + EI2
      EO = A2*EI + X0 + X2 + EI3
      EI1 = 0.
      EI2 = 0.
      EI3 = 0.
      IF(JJ.EQ.1) E1(II+2) = X1
      IF(JJ.EQ.1) E2(II+2) = X3
      IF(JJ.EQ.1) E3(II+2) = EI
      X0 = X1
      X2 = X3
      IF(JJ.EQ.1) GO TO 201
      IF(JJ.EQ.3) GO TO 202
      IF(JJ.EQ.4) GO TO 203
      F(3) = YO*E3(N1+3-II)*G1
      F(4) = YO*E1(N1+3-II)
      F(5) = YO*E1(N1+2-II)*(-P1)
      DO 2 I=3,5
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
  2   EM(I) = EM(I) + ABS(F(I))
      GO TO 201
  202 F(6) = YO*E3(N1+3-II)*G2
      F(7) = YO*E2(N1+3-II)
      F(8) = YO*E2(N1+2-II)*(-P2)
      DO 3 I=6,8
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
  3   EM(I) = EM(I) + ABS(F(I))
      GO TO 201
  203 F(9) = YO*(E1(N1+2-II)+E2(N1+2-II))
      F(10) = YO*E3(N1+3-II)*A2
      DO 8 I=9,10
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
  8   EM(I) = EM(I) + ABS(F(I))
      IF(II.LT.N1) GO TO 201
      IF(G1.EQ.1.) EE(3)=0.
      IF(G1.EQ.1.) EM(3)=0.
      IF(G1.EQ.1.) ER(3)=0.
      IF(P1.EQ.1.) EE(5)=0.
      IF(P1.EQ.1.) EM(5)=0.
```

```
      IF(P1.EQ.1.) ER(5)=0.
      IF(G2.EQ.1.) EE(6)=0.
      IF(G2.EQ.1.) EM(6)=0.
      IF(G2.EQ.1.) ER(6)=0.
      IF(P2.EQ.1.) EE(8)=0.
      IF(P2.EQ.1.) EM(8)=0.
      IF(P2.EQ.1.) ER(8)=0.
      IF(A2.EQ.1.) EE(10)=0.
      IF(A2.EQ.1.) EM(10)=0.
      IF(A2.EQ.1.) ER(10)=0.
      DO 50 I=1,11
50    ER(I) = ER(I)*XM2 + EE(I)**2*XM**2
      EER = 0.
      EMM = 0.
      ERR = 0.
      DO 10 I=1,10
      EMM = EMM + EM(I)
10    ERR = ERR + ER(I)
      IF(JT.NE.1) GO TO 40
      DO 30 I=1,9
      K = I+1
      DO 30 J=K,10
30    EER = EER + EE(I)*EE(J)
      EER = 2.*EER
40    ERMS = SQRT(ERR + EER*XM**2)
      PRINT 302,EMM,ERMS
302   FORMAT('-','PARALLEL',12X,2F16.5)
201   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FIL7(EI,EO)
C     XI FORM
      DIMENSION E1(150),E2(150),E3(150),F(13),ER(13),EM(13),EE(13)
      COMMON/C1/A2,A1,A0,B1,B0,X0,X1,X2,X3,X4,X5,EI1,EI2,EI3
      COMMON/C2/E1,E2,E3,F,ER,EM,EE,II,JJ,KK,N1,Y0,JT,XM,XM2
      COMMON/C3/P1,P2,P3,P4,G1,G2,G3,G4
      X1 = 2.*G3*EI - P3*X0 - P4*X2 + EI1
      X3 = 2.*G4*EI - P3*X2 + P4*X0 + EI2
      EO = X2 + A2*EI + EI3
      EI1 = 0.
      EI2 = 0.
      EI3 = 0.
      IF(JJ.EQ.1) E1(II+2) = X1
      IF(JJ.EQ.1) E2(II+2) = X3
      IF(JJ.EQ.1) E3(II+2) = EI
      X0 = X1
      X2 = X3
      IF(JJ.EQ.1) GO TO 201
      IF(JJ.EQ.3) GO TO 202
      IF(JJ.EQ.4) GO TO 203
      F(3) = Y0*E1(N1+3-II)
      F(4) = Y0*E1(N1+2-II)*(-P3)
      F(5) = Y0*E3(N1+3-II)*2.*G3
      F(6) = Y0*E2(N1+2-II)*(-P4)
      F(7) = F(5) + F(6)
      DO 2 I=3,7
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
 2    EM(I) = EM(I) + ABS(F(I))
      GO TO 201
 202  F(8) = Y0*E2(N1+3-II)
      F(9) = Y0*E2(N1+2-II)*(-P3)
      F(10) = Y0*E3(N1+3-II)*2.*G4
      F(11) = Y0*E1(N1+2-II)*P4
      F(12) = F(10) + F(11)
      DO 3 I=8,12
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
 3    EM(I) = EM(I) + ABS(F(I))
      GO TO 201
 203  F(13) = Y0*E3(N1+3-II)*A2
      EE(13) = EE(13) + F(13)
      ER(13) = ER(13) + F(13)**2
      EM(13) = EM(13) + ABS(F(13))
      IF(II.LT.N1) GO TO 201
      IF(P3.EQ.1.) EE(4)=0.
      IF(P3.EQ.1.) EM(4)=0.
      IF(P3.EQ.1.) ER(4)=0.
```

```
      IF((2.*G3).EQ.1.) EE(5)=0.
      IF((2.*G3).EQ.1.) EM(5)=0.
      IF((2.*G3).EQ.1.) ER(5)=0.
      IF(P4.EQ.1.) EE(6)=0.
      IF(P4.EQ.1.) EM(6)=0.
      IF(P4.EQ.1.) ER(6)=0.
      IF(P3.EQ.1.) EE(9)=0.
      IF(P3.EQ.1.) EM(9)=0.
      IF(P3.EQ.1.) ER(9)=0.
      IF((2.*G4).EQ.1.) EE(10)=0.
      IF((2.*G4).EQ.1.) EM(10)=0.
      IF((2.*G4).EQ.1.) ER(10)=0.
      IF(P4.EQ.1.) EE(11)=0.
      IF(P4.EQ.1.) EM(11)=0.
      IF(P4.EQ.1.) ER(11)=0.
      IF(A2.EQ.1.) EE(13)=0.
      IF(A2.EQ.1.) EM(13)=0.
      IF(A2.EQ.1.) ER(13)=0.
      DO 50 I=1,13
50    ER(I) = ER(I)*XM2 + EE(I)**2*XM**2
      EER = 0.
      EMM = 0.
      ERR = 0.
      DO 10 I=1,13
      EMM = EMM + EM(I)
10    ERR = ERR + ER(I)
      IF(JT.NE.1) GO TO 40
      DO 30 I=1,12
      K = I+1
      DO 30 J=K,13
30    EER = EER + EE(I)*EE(J)
      EER = 2.*EER
40    ERMS = SQRT(ERR + EER*XM**2)
      PRINT 302,EMM,ERMS
302   FORMAT('-','XI',18X,2F16.5)
201   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FIL8(EI,EO)
C     XII FORM
      DIMENSION E1(150),E2(150),E3(150),F(13),ER(13),EM(13),EE(13)
      COMMON/C1/A2,A1,A0,B1,B0,X0,X1,X2,X3,X4,X5,EI1,EI2,EI3
      COMMON/C2/E1,E2,E3,F,ER,EM,EE,II,JJ,KK,N1,Y0,JT,XM,XM2
      COMMON/C3/P1,P2,P3,P4,G1,G2,G3,G4
      X1 = EI - P3*X0 + P4*X2 + EI1
      X3 = -P3*X2 - P4*X0 + EI2
      EO = 2.*G4*X0 - 2.*G3*X2 + A2*EI + EI3
      EI1 = 0.
      EI2 = 0.
      EI3 = 0.
      IF(JJ.EQ.1) E1(II+2) = X1
      IF(JJ.EQ.1) E2(II+2) = X3
      IF(JJ.EQ.1) E3(II+2) = EI
      X0 = X1
      X2 = X3
      IF(JJ.EQ.1) GO TO 201
      IF(JJ.EQ.3) GO TO 202
      IF(JJ.EQ.4) GO TO 203
      F(3) = Y0*E1(N1+3-II)
      F(4) = Y0*E1(N1+2-II)*(-P3)
      F(5) = Y0*(E3(N1+3-II)+P4*E2(N1+2-II))
      F(6) = Y0*E2(N1+2-II)*P4
      DO 2 I=3,6
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
  2   EM(I) = EM(I) + ABS(F(I))
      GO TO 201
  202 F(7) = Y0*E2(N1+3-II)
      F(8) = Y0*E2(N1+2-II)*(-P3)
      F(9) = Y0*E1(N1+2-II)*P4
      DO 3 I=7,9
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
  3   EM(I) = EM(I) + ABS(F(I))
      GO TO 201
  203 F(10) = Y0*E2(N1+2-II)*2.*G3
      F(11) = Y0*E1(N1+2-II)*2.*G4
      F(12) = F(10) + F(11)
      F(13) = Y0*E3(N1+3-II)*A2
      DO 8 I=10,13
      EE(I) = EE(I) + F(I)
      ER(I) = ER(I) + F(I)**2
  8   EM(I) = EM(I) + ABS(F(I))
      IF(II.LT.N1) GO TO 201
      IF(P3.EQ.1.) EE(4)=0.
      IF(P3.EQ.1.) EM(4)=0.
```

```
      IF(P3.EQ.1.) ER(4)=0.
      IF(P4.EQ.1.) EE(6)=0.
      IF(P4.EQ.1.) EM(6)=0.
      IF(P4.EQ.1.) ER(6)=0.
      IF(P3.EQ.1.) EE(8)=0.
      IF(P3.EQ.1.) EM(8)=0.
      IF(P3.EQ.1.) ER(8)=0.
      IF(P4.EQ.1.) EM(9)=0.
      IF(P4.EQ.1.) EE(9)=0.
      IF(P4.EQ.1.) ER(9)=0.
      IF((2.*G3).EQ.1.) EE(10)=0.
      IF((2.*G3).EQ.1.) EM(10)=0.
      IF((2.*G3).EQ.1.) ER(10)=0.
      IF((2.*G4).EQ.1.) EE(11)=0.
      IF((2.*G4).EQ.1.) EM(11)=0.
      IF((2.*G4).EQ.1.) ER(11)=0.
      IF(A2.EQ.1.) EE(13)=0.
      IF(A2.EQ.1.) EM(13)=0.
      IF(A2.EQ.1.) ER(13)=0.
      DO 50 I=1,13
50    ER(I) = ER(I)*XM2 + EE(I)**2*XM**2
      EER = 0.
      EMM = 0.
      ERR = 0.
      DO 10 I=1,13
      EMM = EMM + EM(I)
10    ERR = ERR + ER(I)
      IF(JT.NE.1) GO TO 40
      DO 30 I=1,12
      K = I+1
      DO 30 J=K,13
30    EER = EER + EE(I)*EE(J)
      EER = 2.*EER
40    ERMS = SQRT(ERR + EER*XM**2)
      PRINT 302,EMM,ERMS
302   FORMAT('-','XII',17X,2F16.6)
201   CONTINUE
      RETURN
      END
```