On Digital Filtering

G-AE CONCEPTS SUBCOMMITTEE
C. M. RADER, Chairman

Abstract

Digital filtering is the process of spectrum shaping of signal waveforms, using digital components as the basic elements for implementation. This process is extensively used in the computer simulation of analog filters. The unmistakable trends toward increased speed and decreased cost and size of digital components make digital filtering especially attractive at this time. These trends promise to end the virtual monopoly of analog components for realizing real-time filters.

This paper attempts to set the stage for the companion papers on digital filtering to follow in this topical issue. After introducing the *z*-transform of a discrete-time series, the use of this transform in linear system analysis is considered. The relationship between discrete and continuous signals and systems is then discussed. Since all the papers of this issue are concerned with digital filter implementations in one form or another, only an overview of these implementations is given here. These include filter configurations, design methods, quantization effects, and the fast convolution method for implementing nonrecursive filters.

Manuscript received June 11, 1968.

The members of the subcommittee are: W. T. Cochran, Bell Telephone Laboratories, Inc., Holmdel, N. J.; J. W. Cooley, IBM Corporation, Yorktown Heights, N. Y.; H. D. Helms, Bell Telephone Laboratories, Inc., Whippany, N. J.; R. A. Kaenel, Bell Telephone Laboratories, Inc., Murray Hill, N. J.; J. F. Kaiser, Bell Telephone Laboratories, Inc., Murray Hill, N. J.; W. W. Lang, IBM Corporation, Poughkeepsie, N. Y.; G. C. Maling, IBM Corporation, Poughkeepsie, N. Y.; C. M. Rader, M.I.T. Lincoln Laboratory, Lexington, Mass. (operated with support from the U. S. Air Force); K. Steiglitz, Princeton University, Princeton, N. J.

I. Introduction

Linear network theory is based on the electrical properties of inductances, capacitances, and resistances. These lead, via Kirchoff's laws, to a description of the performance of a network by a set of linear differential equations. By contrast, a set of linear difference equations is used to describe a discrete linear system; these equations are realized (by manipulating numbers) in a special or general purpose digital computer. To realize a linear difference equation, the input signal must be composed of discrete samples, i.e., a sequence of numbers. All considerations here are based on uniformly spaced samples. Nonuniform spacing of samples lies outside the scope of this paper.

The discussion is based on a model whose input consists of discrete samples quantized in amplitude. The samples are then processed by digital logic, which performs the numerical operations required to realize the linear difference equation(s). Initially, it is assumed that the idealized digital logic manipulates the unquantized data with perfect accuracy. The effects of quantization will be considered later. In many practical cases, the effects of numerical error due to quantization may be treated as a noise superimposed on the ideal unquantized data.

An increasingly large number of examples can be identified in which digital filtering appears to be more practical than analog processing for performing such operations as interpolation, extrapolation, smoothing, and spectral decomposition. This is especially true when the data to be operated upon are generated in digital form, e.g., by a digital transducer. The unique advantages offered by digital techniques include the following: potentially small-size integrated circuit implementation; very predictable stable performance of arbitrarily high precision; absence of impedance-matching problems; no restrictions on the location of critical filter frequencies; greater flexibility, because of the ease with which the filter response can be changed by varying the proper coefficients; and the intrinsic possibility of time-sharing major implementation segments. These advantages together with larger scale circuit integration (LSI) promise to make the digital filtering technique eminently suitable for the exacting requirements of modern communications-oriented computing facilities. In fact, the rapid development of LSI has greatly increased the possibility of digital-filtering techniques, thus threatening to end the virtual monopoly of analog processing [3], [4].

The study of discrete-time systems can be approached from two directions: first, they can be viewed as approximations to continuous-time systems and second, they can be considered as existing without reference to any continuous-time systems. Both viewpoints offer advantages; we shall begin with the second and come back to the first.

II. Elements of the z-Transform

A. Definition of the z-Transform

The study of continuous linear dynamic systems is greatly facilitated by the introduction of the operational methods of the Laplace and Fourier transforms, and also by the use of network concepts. Equivalently, the study of linear discrete systems benefits from the introduction of the z-transform and the use of network concepts [2], [8]. To readers familiar with the continuous linear system approach, many of the theoretical developments for discrete linear systems will be familiar [1]. But it is important to remember that we have a discrete or digital framework within which new insights must be developed.

A discrete-time (i.e., time-sampled) signal will be represented by a sequence of numbers $f_0, f_1, \cdots, f_n$. When the function represents a time series, the index n will be referred to as the time parameter; however, the index may represent a space coordinate or some other discrete variable, depending on the nature of the series. What follows can be extended easily to include the case where signals are defined for negative time as well. A common example of a discrete-time signal is provided by the samples of a continuous-time signal $f(t)$ at times $t=nT$, $n=0, 1, 2, \cdots$. The gross national product of a country for successive years and the magnitude of sonar returns for successive pulses are examples of information that might be collected and stored as discrete-time signals.

The z-transform is a natural tool for the solution of linear constant-coefficient difference equations, just as the Laplace transform is appropriate for the solution of linear constant-coefficient differential equations. Given a discrete-time signal f, represented by the sequence of numbers f_n, its z-transform $F(z)$ is defined by the power series

$$F(z) \triangleq \sum_{n=0}^{\infty} f_n z^{-n}, \tag{1}$$

where z is interpreted as a complex transform variable. The variable z plays a role similar to that of the Laplace transform variable s. The operation of taking the z-transform of a sequence will be denoted by

$$F(z) = Z\{f_n\}. \tag{2}$$

The z-transform is a linear operation, so that

$$Z\{af_n + bg_n\} = aZ\{f_n\} + bZ\{g_n\}, \tag{3}$$

where a and b are constants.

An important property that can be derived from (1) is the z-transform of a delayed sequence:

$$Z\{f_{n-k}\} = z^{-k} \cdot Z\{f_n\}. \tag{4}$$

Equation (4) may be used to derive the important convolution property of the z-transform. The product of two z-transforms

$$F(z) = G(z)H(z) \tag{5}$$

is represented in the time domain as the discrete convolution

$$f_n = \sum_{j=0}^{\infty} h_j g_{n-j}, \tag{6}$$

where $F(z)=Z\{f_n\}$, $G(z)=Z\{g_n\}$, and $H(z)=Z\{h_n\}$.

B. Examples of the z-Transform

It will be useful at this point to calculate the z-transforms of some commonly encountered discrete-time signals. Consider first the constant signal

$$f_n = 1, \qquad n = 0, 1, 2, \cdots. \tag{7}$$

From (1) the z-transform is found simply to be the geometric series

$$Z\{1\} = \sum_{n=0}^{\infty} z^{-n} = \frac{1}{1 - z^{-1}}, \qquad |z| > 1. \tag{8}$$

Differentiating (8) with respect to z^{-1} and multiplying the result by z^{-1}, we obtain the z-transform of a signal linearly increasing in value,

$$Z\{n\} = \sum_{n=0}^{\infty} nz^{-n} = \frac{z^{-1}}{(1 - z^{-1})^2}. \tag{9}$$

The z-transform of higher-order polynomials can be obtained by further differentiation.

Consider next the exponential function

$$f_n = c^n, \qquad n = 0, 1, 2, \cdots, \tag{10}$$

where c is some real number. The z-transform is a geometric series, which is readily summed.

$$Z\{c^n\} = \frac{1}{1 - cz^{-1}}, \qquad |z| > |c|. \tag{11}$$

Differentiating with respect to z^{-1} and multiplying the result by z^{-1} as before, we obtain

$$Z\{nc^n\} = \frac{cz^{-1}}{(1 - cz^{-1})^2}, \qquad |z| > |c|, \tag{12}$$

which, of course, includes the first result when $c=1$. Differentiating again, we get

$$Z\{n^2c^n\} = \frac{cz^{-1} + cz^{-2}}{(1 - cz^{-1})^3}, \qquad |z| > |c| \tag{13}$$

and so on.

Although discrete-time signals will usually take on real values, we may for the moment consider the exponential signal c^n for complex c in order to derive further z-transforms. Write

$$c = ae^{jb}, \tag{14}$$

where $a>0$ and $j=\sqrt{-1}$; a is often called the damping factor and b the phase factor. Then the transform (11) is still valid. Taking the real part of (11) with z as the variable, we obtain

$$Z\{a^n \cos nb\} = \frac{1 - az^{-1}\cos b}{1 - 2az^{-1}\cos b + a^2z^{-2}} = \frac{1 - az^{-1}\cos b}{(1 - ae^{jb}z^{-1})(1 - ae^{-jb}z^{-1})}, \quad |z| > a, \tag{15}$$

which is thus the z-transform of an exponentially damped or growing sinusoidal discrete-time signal. Notice that this z-transform has a pair of complex poles in the z-plane at a radius depending on the damping factor and at an angle depending on the phase factor b. The angle of the poles reaches a maximum when b reaches π and is taken modulo 2π for b greater than π. This fact will be important when we discuss the sampling of continuous-time signals. The imaginary part of (11) yields in a similar manner

$$Z\{a^n \sin nb\} = \frac{az^{-1}\sin b}{1 - 2az^{-1}\cos b + a^2z^{-2}} \tag{16}$$

$$= \frac{az^{-1}\sin b}{(1 - ae^{jb}z^{-1})(1 - ae^{-jb}z^{-1})}. \tag{17}$$

By taking the real and imaginary parts of expressions such as (12) and (13), we may finally obtain z-transforms for any function of the form

$$n^k a^n \sin(nb + c) \tag{18}$$

and finite linear combinations of these. The result will always be a rational function of z^{-1} and, as we shall see, any rational function of z may be broken down into finite sums of such functions. This class of signals corresponds to the class of continuous-time signals with rational Laplace transforms and has much the same importance in engineering problems.

An even simpler class of discrete-time signals consists of those that are zero after a finite time N. The z-transform of such a signal is then a finite series in z^{-1} and is analytic for all values of $z \neq 0$.

$$F(z) = \sum_{n=0}^{N} f_n z^{-n}, \qquad z \neq 0. \tag{19}$$

These signals, having polynomial z-transforms, may be considered as belonging to the class of rational functions.

C. Definition of the z-Transform Inverse

Since the z-transform is a power series in z^{-1}, it represents a Taylor series expansion of $F(z)$ about the point $z^{-1}=0$ (the point at infinity in the z-plane) and hence

$$f_n = \frac{1}{n!}\left[\left(\frac{\partial}{\partial z^{-1}}\right)^n F(z)\right]_{z=\infty}. \tag{20}$$

Another expression for the inverse z-transform can be obtained by multiplying (1) by z^{m-1} and integrating on a contour C, which is entirely within the region of convergence of the series (1) and which encircles the origin once.

$$\frac{1}{2\pi j}\int_C z^{m-1}F(z)dz = \frac{1}{2\pi j}\int_C \sum_{n=0}^{\infty} f_n z^{m-n-1}dz = \sum_{n=0}^{\infty} f_n \frac{1}{2\pi j}\int_C z^{m-n-1}dz = f_m, \tag{21}$$

where the interchange of orders of integration and summation is justified by the uniform convergence of the power series.

The two formulas (20) and (21) for the inverse z-transform are general and may be applied to any transform that has a region of convergence. When dealing with a rational function of z^{-1}, however, there are much simpler methods of obtaining the signal values f_n. Perhaps the simplest method is long division, which consists of dividing the numerator by the denominator to obtain a power series in z^{-1}. To illustrate, consider the transform (9)

$$\begin{array}{r@{}l}
 & z^{-1} + 2z^{-2} + 3z^{-3} + \cdots \\
1 - 2z^{-1} + z^{-2}\,\big) & z^{-1} \\
 & z^{-1} - 2z^{-2} + z^{-3} \\
\hline
 & 2z^{-2} - z^{-3} \\
 & 2z^{-2} - 4z^{-3} + 2z^{-4} \\
\hline
 & 3z^{-3} - 2z^{-4} \\
 & \cdots
\end{array} \tag{22}$$

This is easily implemented on the digital computer and has the advantage that the locations of the poles of $F(z)$ need not be found. This long division is sensitive to round-off error, however, and is generally useful only when the poles are well inside the unit circle (large damping) [18].

The second method of z-transform inversion is applicable to a proper rational function of z^{-1}, with L poles.

$$F(z) = \frac{\sum_{i=0}^{L-1} a_i z^{-i}}{1 + \sum_{i=1}^{L} b_i z^{-i}} = z\left[\frac{\sum_{i=0}^{L-1} a_i z^{L-1-i}}{z^L + \sum_{i=1}^{L} b_i z^{L-i}}\right]. \tag{23}$$

Assume for simplicity that the poles p_i are distinct and expand the bracketed expression in a partial fraction expansion:

$$F(z) = z\sum_{i=1}^{L}\frac{A_i}{z - p_i} = \sum_{i=1}^{L}\frac{A_i}{1 - p_i z^{-1}}. \tag{24}$$

Each of the L terms in (21) has a known inverse z-transform given by (11), so that

$$f_n = \sum_{i=1}^{L} A_i p_i^n, \tag{25}$$

where the p_i may be complex but will appear in complex conjugate pairs.

The partial fraction expansion of (23) and the formation of (25) require the use of a polynomial root-finding routine. This is not the case if the denominator of $F(z)$ is already known in factored form. This second method has the advantage that the value of the signal at time n can be computed without computing any other values.

III. Use of the z-Transform in Linear System Analysis

A. Definition of the Digital Transfer Function

For a linear discrete system, the input (x_n) and output (y_n) signals are related by linear difference equations with constant coefficients of the form

$$y_n + b_1 y_{n-1} + \cdots + b_M y_{n-M} = a_0 x_n + \cdots + a_N x_{n-N}. \tag{26}$$

That is, at time n the output value can be computed from the current input and a linear combination of past inputs and outputs. If we take the z-transform of this equation, term by term, we obtain

$$Y(z)\left(1 + \sum_{i=1}^{N} b_i z^{-i}\right) = X(z) \sum_{i=0}^{N} a_i z^{-i}, \tag{27}$$

where i is used in place of n in (1). Then

$$Y(z) = H(z)\, X(z), \tag{28}$$

where $H(z)$ is a proper rational function of z^{-1}. Thus, if the values of the discrete-time signal x are known, $X(z)$, $Y(z)$, and hence the values of the discrete-time signal y can be found by the methods described above. In particular, if $X(z)$ is a polynomial or a rational function of z, $Y(z)$ will be a rational function of z. Its inverse will consist of exponential and sinusoidal terms like those discussed in Section II-B. The function $H(z)$ is called the transfer function relating x to y.

B. Time-Invariant Linear Operators: Digital Filters

By an operator on discrete-time signals, we mean a rule by which a discrete-time output signal is determined from a discrete-time input signal. We denote the input signal by x_n, the output signal by y_n, and the operator by $\boldsymbol{H}(\ )$, and write

$$Y_n = \boldsymbol{H}(x_n). \tag{29}$$

By a linear operator, we mean one for which

$$\boldsymbol{H}(ax_n + bw_n) = a\boldsymbol{H}(x_n) + b\boldsymbol{H}(w_n) \tag{30}$$

for all constants a and b, and all signals x_n and w_n. By a time-invariant operator, we mean one for which

$$Y_{n-k} = \boldsymbol{H}(x_{n-k}) \tag{31}$$

for all time translations k.

Time-invariant linear operators on discrete-time signals are normally called digital filters [1]. They can be characterized as follows. Let the digital filter $\boldsymbol{H}$ respond to a signal that is unity at $n=0$ and zero at all other times, with the output signal h_n. We shall make the additional assumption that $\boldsymbol{H}$ does not respond before an input appears, so that h_n is a legitmate one-sided discrete-time signal, the impulse response of $\boldsymbol{H}$. The response of $\boldsymbol{H}$ to a discrete-time signal x_n is therefore, from (27) and (28),

$$y_n = \sum_{j=0}^{\infty} x_j h_{n-j}. \tag{32}$$

This result will be recognized as a discrete convolution of the signal x_n with the impulse response h_n of the filter $\boldsymbol{H}$. The z-transform of both sides of (32) may be taken, the order of summation on the right-hand side reversed, and the delayed sequence relation (4) used. The result is

$$Y(z) = H(z)X(z). \tag{33}$$

This shows that the z-transform of the impulse response of $\boldsymbol{H}$ can be interpreted as a multiplicative transfer function in much the same way as the Fourier transform of the impulse response of a continuous-time filter is interpreted.

Digital filters that can be realized on a digital computer include those with rational transfer functions (or as a special case, polynomial transfer functions). These can be implemented conveniently by the recursive relation obtained from (26).

$$y_n = a_0 x_n + \cdots + a_N x_{n-N} - b_1 y_{n-1} - \cdots - b_L y_{n-M}. \tag{34}$$

The output at any time depends not only on a finite number of past inputs, but also on the present input and a finite number of past outputs. The reader should note that (34) is equivalent to one step of the long-division method of inverting a z-transform. Other implementations of the same transfer function, employing a partial fraction expansion of $H(z)$, may be preferable because of the effect of finite computer word length.

C. Definition of the Frequency Response of Digital Filters

Just as the frequency response of a continuous-time filter is determined by the values of its transfer function on the imaginary axis, the frequency response of a digital filter is determined by its values on the unit circle ($|z| = 1$). This can be seen in much the same way. Consider an input discrete-time signal sin nb, with the z-transform given by (17) as

$$Z\{\sin nb\} = \frac{z^{-1}\sin b}{(1 - e^{jb}z^{-1})(1 - e^{-jb}z^{-1})}. \tag{35}$$

This function has complex conjugate poles on the unit circle at $z = e^{\pm jb}$ and has a time response whose envelope neither grows in magnitude nor decays. This is completely analogous to a sinusoid that is continuous in time. Now consider a digital filter $H(z)$ with a rational transfer function and all its poles inside the unit circle; this requirement guarantees that the digital filter is strictly stable. The z-transform of the output y_n is also a rational function of z^{-1} and hence it has a partial fraction expansion of the following form:

$$Y(z) = H(z)X(z) = \frac{H(e^{jb})/(2j)}{1 - e^{jb}z^{-1}} + \frac{H(e^{-jb})/(-2j)}{1 - e^{-jb}z^{-1}} + \text{decaying terms.} \tag{36}$$

Putting the two steady-state terms over a common denominator, they become

$$\frac{\operatorname{Re}\{H(e^{jb})\}z^{-1}\sin b + \operatorname{Im}\{H(e^{jb})\}(1 - z^{-1}\cos b)}{(1 - e^{jb}z^{-1})(1 - e^{-jb}z^{-1})}, \tag{37}$$

which by (15) and (17) has the inverse z-transform

$$\begin{aligned} y_n &= \operatorname{Re}\{H(e^{jb})\}\sin nb + \operatorname{Im}\{H(e^{jb})\}\cos nb \\ &= |H(e^{jb})| \sin[nb + \operatorname{Arg} H(e^{jb})]. \end{aligned} \tag{38}$$

Thus the magnitude of $H(z)$ on the unit circle represents the transmission factor of a steady-state sinusoid at that frequency and the phase angle represents the phase shift of a steady-state sinusoid at that frequency, in complete analogy with the continuous-time case.

The frequency variable b is naturally limited to the range between $-\pi$ and π $(-\pi < b \leq \pi)$, in contrast with the continuous-time case. A discrete-time signal whose z-transform has a pole at $z = -1$ has a steady-state term of the form $a(-1)^n$. If this is interpreted as representing equally-spaced samples of a continuous-time signal, this term represents samples of a sinusoid with the frequency $1/2T$ Hz, where T is the sampling period. Thus we may think of the frequency circle in the z-plane and the frequency axis in the s-plane as related by

$$f = \frac{b}{\pi}\left(\frac{1}{2T}\right) \text{Hz} \tag{39}$$

whenever uniform sampling of a continuous-time signal is involved. The frequency $1/2T$ Hz is called the Nyquist frequency and represents the highest frequency that can be represented unambiguously by samples spaced every T seconds.

IV. Digital Implementation

A. Digital Filter Configurations

Assume that a digital filter has been designed in the sense that the transfer function $H(z)$ has been chosen. $H(z)$ is, at most, a ratio of polynominals in z^{-1}, and is finite outside and on the circle $|z| = 1$ in the z-plane.

$$H(z) = \frac{\sum_{i=0}^{N} a_i z^{-i}}{1 + \sum_{i=1}^{M} b_i z^{-i}}. \tag{40}$$

$H(z)$ might be written in some other form that can be manipulated into the form (40), for example,

$$H(z) = H_1(z) + H_2(z) + \cdots, \tag{41}$$

where $H_1(z)$ and $H_2(z)$ are ratios of polynomials, or

$$H(z) = \frac{(z - z_1)(z - z_2) \cdots (z - z_n)}{(z - p_1)(z - p_2) \cdots (z - p_n)}. \tag{42}$$

Although (41) or (42) can be manipulated into the form (40) by simple algebra, it is not generally wise to do so in practice for reasons of numerical accuracy in the realization.

In the analog world, the realization of a given system function is a moderately difficult problem that has received considerable attention. For digital filters, the implementation of a difference equation or system of difference equations to realize a given $H(z)$ is almost trivial. Suppose the input is x_n with transform $X(z)$ and the output is y_n with transform $Y(z)$. Then from (40)

$$Y(z) = \sum_{i=0}^{N} a_i z^{-i} X(z) - \sum_{i=1}^{M} b_i z^{-i} Y(z) \tag{43}$$

or, in the time domain,

$$y_n = \sum_{i=0}^{N} a_i x_{n-i} - \sum_{i=1}^{M} b_i y_{n-i}, \tag{44}$$

which is, in fact, the difference equation that realizes $H(z)$ directly. That is, (44) gives a straightforward rule for computing y_n in terms of the $N+1$ most recent samples of the input x_n and the M previous values of y, already computed. Once y_n has been computed and as soon as a new input sample x_{n+1} is found, we can proceed to compute y_{n+1}. In this way, an entire input sequence of indefinite duration can be filtered by (44) to produce an output sequence of the same length.

It is often helpful to have pictures to describe waveform processing operations. A diagram to describe (44) is shown in Fig. 1. The rectangle with a constant written inside represents multiplication of a variable by a constant and the rectangle with z^{-1} inscribed represents a one-sample delay. The circle with a Σ inscribed is, of course, a summing point. The interpretation of Fig. 1, in terms of a digital machine, is thus as follows.

At $t = nT$, x_n becomes available. The quantities x_{n-1}, $x_{n-2}, \cdots, x_{n-N}, y_{n-1}, \cdots, y_{n-M}$ at the outputs of the delay elements have been remembered. Thus all the variables are available for the computation of y_n. When this computation is complete, x_{n-M} and y_{n-N} are discarded and the other quantities are saved. They will be needed for

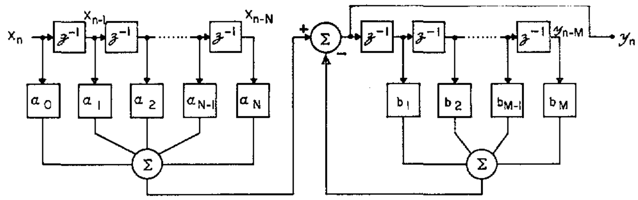


Fig. 1. Pictorial representation of (44).

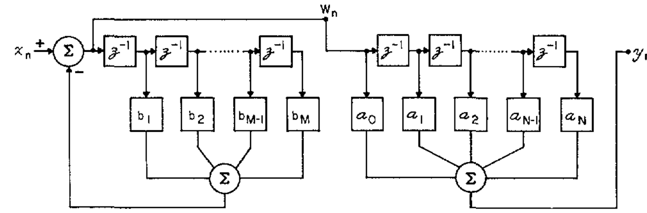


Fig. 2. The same filter as Fig. 1, but with the subsystem interchanged.

the next computation. By counting the delays, we can obtain a minimum estimate of the number of storages involved in realizing (44) and by counting rectangles with nontrivial associated constants, we can see how many multiplications are required per sample.

Equation (44), corresponding to Fig. 1, is not the only possible way to realize a given digital filter function $H(z)$. As a trivial example, suppose in Fig. 1 the sum of weighted delayed inputs is considered as one filter $N(z)$ and the remainder of the network is considered as a second filter $1/D(z)$. $N(z)$ is the numerator of $H(z)$ and $D(z)$ is the denominator of $H(z)$. But these are linear systems in cascade. Thus the same overall transfer function $H(z)$ is obtained if the order of the subsystems is reversed, as in Fig. 2. An intermediate variable w_n is introduced and the single difference equation is replaced by a pair of equations, but with no additional computation.

$$
\begin{aligned}
w_n &= x_n - \sum_{i=1}^{M} b_i w_{n-i} \\
y_n &= \sum_{i=0}^{N} a_i w_{n-i}.
\end{aligned}
\tag{45}
$$

An advantage of (45) over (44) is the smaller memory requirement. We are required to save only N or M previous values of w_n, depending on which is greater. This may be illustrated by redrawing Fig. 2 as Fig. 3, in which the delay elements having the same output have been replaced by a single delay element. Fig. 3 is drawn for $N = M$.

Figs. 1 and 3 are called direct forms. It is to be emphasized that the constants a_i and b_1 in the network are the same as the constants in the transfer function. Continuous filter realization would be simple if the values of resistors, inductors, and capacitors in a network were so easily related to the transfer function of a continuous filter.

Despite the simplicity of the direct forms that realize $H(z)$, they are undesirable for high-order difference equations for reasons of numerical accuracy. But there are other forms. Suppose $H(z)$ is expressed in the form (41). Then the output y_n is the sum of the outputs of several smaller filters, $H_1(z)$, $H_2(z)$, $\cdots$. Each of these can be realized in either of the direct forms. Thus such representation of $H(z)$ leads to the picture in Fig. 4. In the extreme, $H(z)$ could be expressed in a partial fraction expansion so

Fig. 3. The system of Fig. 2 with redundant delay elements eliminated.

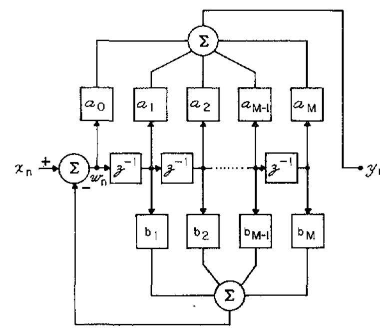
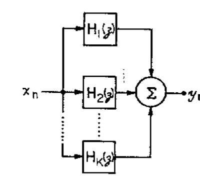


Fig. 4. The pictorial representation of (41).



that each of the terms in (41) would be a ratio of first- or second-order polynomials in z^{-1}. Then Fig. 4 becomes the parallel form of a digital filter. The parallel form tends to be not nearly as sensitive to quantization effects as the direct forms. Each of the subfilters in the parallel form is realized in one of the two direct forms.

If $H(z)$ is expressed in the form (42), we can express it as

$$
H(z) = H_1(z) \times H_2(z) \times \cdots \times H_k(z), \tag{46}
$$

where each of the subfilters includes a subset of the poles and zeros of $H(z)$. Since these transfer functions are mulplied, the filters are in cascade. Thus Fig. 5 is a description of a realization of $H(z)$. In the case where all the $H_i(z)$ are chosen to be simple ratios of first- or second-order polynomials, we have the cascade form of $H(z)$. The cascade form is also preferable to the direct forms for numerical reasons.

There is an infinity of other possible forms of networks to realize a given $H(z)$. An example of further generality is given by the second-order system of equations

$$
\begin{aligned}
y_n &= a y_{n-1} + b w_{n-1} + c x_n \\
w_n &= d y_{n-1} + e w_{n-1} + f x_n,
\end{aligned}
\tag{47}
$$

where, in general, each of the present states y_n, w_n, is a weighted sum of all the previous states y_{n-1}, w_{n-1}, and the input. These coupled equations tend to require more multiplications to realize a given $H(z)$ than the direct, parallel,

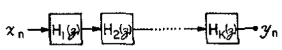


Fig. 5. The pictorial representation of (46).

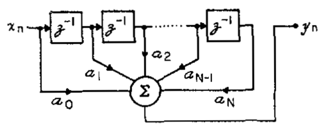


Fig. 6. A nonrecursive filter, pictorially.

or cascade realizations, but the increase in flexibility afforded thereby may be enough to overcome numerical accuracy problems in certain cases. The coupled forms do find use, especially in computer simulations.

The realizations given here assumed that $H(z)$ was a ratio of polynomials. These are called recursive digital filters. They are distinguished by feedback of delayed outputs or intermediate computational variables. They have transfer functions with poles at locations other than the trivial $z=0$. However, some design methods yield $H(z)$ a polynomial in z^{-1} rather than a ratio of polynomials. These are called nonrecursive digital filters. Of the realizations proposed so far, Figs. 1, 2, and 3 all become the same, namely a tapped delay line with a weighted sum of the signals at the equally spaced taps, as shown in Fig. 6. This realization has also been called a transversal filter. Fig. 4, the parallel form, has no particular meaning for a nonrecursive filter; while Fig. 5, the cascade form, is possible but not in common use, because it is usually very hard to factor the high-order polynomials $H(z)$ that arise in practice and also because there is no particular advantage to Fig. 5 over Fig. 1.

An important difference between recursive and nonrecursive digital filters exists in the range of values of M and/or N encountered in typical applications. Recursive digital filters usually meet the kinds of specifications arising in practice (bandpass or bandstop filters, for example) with at most 10 or 20 coefficients. Thus the computation required to produce each output, given a new input, is of the order of 10 to 20 multiplications and additions per sample point. In contrast, nonrecursive digital filters, when used to realize complex-shaped frequency responses, may require several hundred coefficients (even though there are no poles except at $z=0$).

B. Nonrecursive Filter Implementation by High-Speed Convolution

In this section, a computationally efficient method for obtaining the output of a nonrecursive filter will be presented. The nonrecursive filter is characterized by the absence of feedback; that is, past values of the output sequence are not used in computation of the current value of y_j. The nonrecursive filter relationship may be written as

$$y_j = \sum_{n=0}^{M-1} h_n x_{j-n}. \tag{48}$$

When M is large enough, it is computationally efficient to implement the filter by means of the technique called high-speed convolution [11], [12], [14]. This technique is based upon three observations. 1) The discrete convolution of (48) may be replaced by multiplication of the z-transforms of y_j, h_n, and x_n (see (5) and (6), and [10]). 2) The z-transforms may be evaluated at uniformly spaced points on the unit circle in the z-plane. The resulting transform is called the discrete Fourier transform (DFT). 3) The DFT and inverse discrete Fourier transform (IDFT) may be computed by means of the fast Fourier transform (FFT) algorithm, which requires approximately $L \log_2 L$ operations (multiply-adds). Here, L is the number of samples in the array being transformed. The DFT is defined by (49).

$$V_k = \sum_{n=0}^{L-1} v_n e^{-j2\pi nk/L}, \qquad k = 0, 1, \cdots, L-1. \tag{49}$$

From the properties of the exponential function, it can be shown that the IDFT is given by

$$v_n = \frac{1}{L} \sum_{k=0}^{L-1} V_k e^{+j2\pi nk/L}, \qquad n = 0, 1, \cdots, L-1. \tag{50}$$

We can take the input sequence x_n and convolve it with the aperiodic finite length impulse response h_n by using the FFT as follows.

We form a succession of short sequences $x_n^{(k)}$, by taking $L-M+1$ samples at a time to form successive sections of x, which abut but do not overlap, and by appending $M-1$ zero-valued samples to the end of each $x_n^{(k)}$, to form L point sequences. The optimum size for L has been discussed by Stockham [11] and Helms [12]. For example, if $M=128$, the optimum value for L is 1024. Variations about this optimum value do not, however, produce large increases in the required computation time.

The impulse response h_n is then used to form an L point sequence h_n by appending $L-M$ zero-valued samples to the end. Using the FFT, we compute the DFT's of each of these L point sequences and multiply the DFT of h_n with the DFT of each $x_n^{(k)}$. Then the IDFT of each product is computed using the FFT. The result of the process is a succession of L point sequences, which are the convolutions of each of the sections of x with the impulse response h. The periodic nature of the convolution as computed by this technique was avoided by putting enough zeros on the end of each sequence that the periodic convolution contained the complete aperiodic convolution in each period. This is always possible when $L>M$. Each of these results must then be added to each of the others with the appropriate delay to form the output of the filter. Usually, L is chosen to be a power of two, for which the fast Fourier transform is very efficient. This method is called the overlap-add method of high-speed convolution.

There is a second method, called the select-save method, in which the input sequences are formed from overlapping pieces of x and the outputs are selected so that only valid results are used as filter output. Both methods have been described in detail in the literature [11], [12], [14]. The important feature to note is the dependence of the computation time for both methods on $\log_2 M$, rather than M, per output point. This means that once the filter is complicated enough to have required the use of the FFT algorithm, almost any additional complexity may be attained at very small cost (except in terms of memory). So far, the high-speed convolution technique has found application only when the digital filtering was performed by a general-purpose computer program. For a given operation speed, the high-speed method is faster for M greater than roughly 30. While special-purpose hardware designed to perform the direct sum-of-products approach to convolution has been in use for several years, there has been slower progress in the hardware implementation of the fast Fourier transform. Arithmetic and logical units designed to perform the FFT algorithm are, however, beginning to appear.

C. Choice of the Transfer Function $H(z)$

There are a great many methods of choosing $H(z)$ to meet a given filtering requirement and the choice of an appropriate method depends on the requirement. The simplest case by far is when a desired impulse response is to be duplicated. Suppose that a measurement has yielded a sequence of M numbers, which are successive samples of an impulse response. A digital filter, whose impulse response is the samples of the given impulse response, is a nonrecursive (tapped delay line) filter for which the weight applied to the kth tap is the kth sample of the impulse response. Note that in practical situations, this will lead to a nonrecursive filter with a very large number of coefficients.

In the common case where the desired impulse response is that of an analog filter of the classical type (an RLC filter), the impulse response of the filter is known to be of the form of (25), assuming that all the poles are distinct, and we know that we can find a recursive filter with this impulse response by z-transforming (25). Even though the impulse response (25) has an infinite number of terms, the recursive digital filter typically requires only a few coefficients [3], [4], [6], [7], [9].

Both methods based on impulse response must be examined in the frequency domain. Because the impulse response is sampled, the frequency response exhibits folding or aliasing [see (61)]. The frequency response of the digital filter is thus a poor imitation of the frequency response of the analog prototype, if the latter has significant frequency response beyond $1/2T$ Hz.

To overcome this limitation in the design of recursive filters, one can use the bilinear z-transform or z-form, given in (53). This maps the entire left half of the s-plane into the interior of the unit circle. The poles and zeros of an analog filter can thus be located in the z-plane in such a way that the resulting digital filter has a frequency response that goes through the same values, in the same order (as the unit circle is traversed from 0 to π) as the analog frequency response displayed from 0 to ∞. The price paid is in the nonlinear warping of the frequency scale and the distortion of the impulse response. The nonlinear warping effect can be overcome by a technique called predistortion, in which an analog filter is designed that will give the desired frequency response after warping of the frequency scale. This technique is practical only when digital filters whose frequency response is piecewise constant—not piecewise linear—are being designed [3]–[7].

When a nonrecursive filter is designed to meet an arbitrary frequency response criterion, a common approach is the Fourier series technique [4], [7], [14]. The frequency response (usually chosen to be real and even) is expanded as a Fourier series (since it is a periodic function with period 2π). The Fourier coefficients a_n form the impulse response of an ideal filter meeting the specifications exactly.

$$H^*(z) = a_0 + \frac{1}{2} \sum_{n=1}^{\infty} a_n(z^n + z^{-n}). \tag{51}$$

This filter has an infinite number of terms and is unrealizeable unless a finite approximation is made. Simply eliminating all a_n, $n \geq M$ for some suitably chosen M will work nicely if the Fourier series is rapidly convergent. However, a result of poor convergence is the so-called Gibbs phenomenon, which typically produces an overshoot of about 9 percent at any discontinuity in the desired frequency response. An approach to the elimination of the Gibbs phenomenon is to multiply the a_n by a window function w_n. The effect of a window function is to smooth the frequency response and thereby attenuate the overshoot.

There are other methods [4], [13] of design of both recursive and nonrecursive filters, but the above summary gives an idea of the problems to be encountered and the trade-offs involved.

D. Quantization Effects in Digital Filters

Up to this point, it has been tacitly assumed that the sampling operation and the arithmetic operations indicated in (40) are performed with infinite accuracy. In the actual implementation of (40), either by a computation subroutine for a digital computer or by the construction of special-purpose digital hardware, infinite accuracy of representation and computation are not possible. There are three primary sources of error that arise from the use of a finite word length computer.

One source of error is incurred when the input to the filter is quantized to a finite number of bits. This quantization creates an additive noise, which may be treated as random if the quantization is fine enough and if the signal varies sufficiently, relative to the sampling rate and the number of quantization levels.

The second source of error arises in the evaluation of the arithmetic products and their sum as indicated in (40). For the nonrecursive filter ($b_j=0, j=1, 2, \cdots, n$) the magnitude of the error incurred by using finite arithmetic can be quickly estimated by approximating the action of truncation and round-off with a noise source (which can be considered random in most cases). For the recursive filter the calculation of the errors is more difficult as a result of the feedback inherent in the b_j terms. For one thing, while there is no absolute necessity to round or truncate the products in a nonrecursive filter, in the recursive filter the sums of products that are fed back must be rounded or truncated, since after a multiplication of two quantities represented by k_1 and k_2 bits, respectively, the product contains k_1+k_2 bits. If it were fed back without rounding, the next stage would generate numbers requiring still more bits, etc. Again each truncation or rounding operation adds a small noise term, which can be considered random in most cases, and these terms are passed through a digital filter consisting of part or all of the required digital filter [3], [4], [6], [15]–[17], [19]. Obviously, in a cascade realization, the noise generated in the kth stage cannot be seen by any of the earlier stages. A similar effect causes the noise in some of the direct realizations to pass through those portions of the filter that realize the poles of $H(z)$ and not through those portions that realize the zeros.

A related effect also may occur in recursive filters as a result of round-off error when the round-off noise is highly correlated with the signal or highly correlated with itself from iteration to iteration. This is the so-called dead-band effect [6], [21]. This is best illustrated by an example. Suppose the digital filter is described by

$$y_n = .99y_{n-1} + x_n, \tag{52}$$

but is implemented with products rounded to the nearest integer. Then with the input zero, the output would be expected to decay to zero. However, any output in the range -50 to 50 causes the error due to quantization to exactly balance the decay per iteration, so that the erroneous output is maintained.

The third source of error arises in the representation of each of the digital filter coefficients by a fixed number of bits. This effect is analogous to that encountered in continuous filters when the components called for by the design are unavailable [4], [6], [17], [18], [20]. If a design calls for a coefficient of 0.95, the best 6-bit (i.e., 5 bits for the fraction and a sixth bit for the sign) approximation we can make is 0.9375. The best 7-bit approximation we can make is 0.953125, and so on. It must be realized that certain forms of digital filter realization are extremely sensitive to these errors in coefficients. For the nonrecursive filter, the magnitude of this coefficient accuracy problem can be quickly estimated by simply looking at the relative magnitudes of the coefficients making up the weighting sequence. For the recursive filter with its inherent feedback the results are not so simple, as the stability of the filter itself may be affected by coefficient round-off. This problem is most severe when using the direct form for realization of the recursive filter. In general, the direct form should be avoided for fourth and higher order recursive filters because of this effect.

In considering quantization effects, it is not as necessary to compute the exact results of the effects, which is difficult, as to estimate the bounds on them as a guide to avoiding the effects that cannot be tolerated. The theory developed in the literature so far has concentrated on rough estimates, such as upper bounds and mean square errors.

V. Relationship Between Discrete and Continuous Signals and Systems

A. Formal Equivalence of Discrete-Time and Continuous-Time Filtering Theory

In the previous sections, the theory of discrete-time signals and digital filters has been developed without using continuous-time theory. A theory has been developed that is in every way similar to the continuous-time theory insofar as linear time-invariant filtering is concerned. The mathematical formulation of several operations on both discrete-time and continuous-time signals is shown in Table I.

It is also possible to rigorously establish a one-to-one correspondence between discrete-time signals and continuous-time signals in such a way that corresponding quantities are images of each other. This has been done in the axiomatic framework of Hilbert space [1], each continuous-time signal being mapped to a discrete-time signal via an orthonormal expansion. The one-to-one correspondence between the unit circle and the imaginary axis is, in this case, provided by the mapping

$$z = \frac{1 + j\omega}{1 - j\omega}, \tag{53}$$

which is a useful transformation in relating the filter design problems in the two domains [3]–[7], [9].

B. The Sampling Process

In contrast to the theoretical correspondence between discrete- and continuous-time signals described previously the usual process of sampling the values of a continuous-time signal at regular intervals does not yield a one-to-one correspondence between the discrete- and continuous-time signals. In fact, a sinusoid of one frequency is, after sampling, indistinguishable from sinusoids at frequencies

TABLE I

Correspondences Between Operations on Continuous-Time and Discrete-Time Signals

	Discrete-Time	Continuous-Time
Transform	$F(z) = \sum_{n=0}^{\infty} f(n)z^{-n}$	$F(s) = \int_0^{\infty} f(t)e^{-st}dt$
Inverse Transform	$f(n) = \frac{1}{2\pi j}\int_C F(z)z^n \frac{dz}{z}$	$f(t) = \frac{1}{2\pi j}\int_C F(s)e^{st}ds$
Inner Product	$\sum_{n=0}^{\infty} f(n)g(n) = \frac{1}{2\pi j}\int_C F(z)G(z^{-1})\frac{dz}{z}$	$\int_0^{\infty} f(t)g(t)dt = \frac{1}{2\pi j}\int_C F(s)G(-s)ds$
Frequency Line	unit circle	imaginary axis
Convolution Filter	$\sum_{j=0}^{\infty} f_j g_{n-j}$	$\int_0^{\infty} f(\tau)g(t-\tau)d\tau$
Operator Leading to Rational Transfer Function	linear constant-coefficient difference equation	linear constant-coefficient differential equation

differing from the original by an integer multiple of $1/T$ Hz, the sampling frequency. One approach to the problem is to consider band-limited signals and limit consideration to the baseband of frequencies between $-1/2T$ and $1/2T$ Hz. When this is done, the correspondence

$$z = e^{j\omega T}, \qquad |\omega| < \frac{\pi}{T}, \tag{54}$$

can be used to provide a one-to-one correspondence between the baseband and the unit circle, and, as is well known, the original signal can be recovered exactly from its samples. In a practical sense, however, the sampling process always destroys some information and operations on sampled signals can represent operations on the original continuous-time signals only approximately.

Uniform sampling can be represented by multiplication by a periodic train of impulses, producing a train of impulses weighted by the sample values of the continuous-time signal. We call this impulse train $f^*(t)$ and write

$$f^*(t) = f(t)\sum_{n=0}^{\infty}\delta(t - nT), \tag{55}$$

which can also be written

$$f^*(t) = \sum_{n=0}^{\infty} f(nT^+)\delta(t - nT), \tag{56}$$

where we have adopted the convention of taking the right-hand limit when sampling at a discontinuity. The Laplace transform of $f^*(t)$ is, therefore,

$$F^*(s) = \sum_{n=0}^{\infty} f(nT^+)e^{-nsT}, \tag{57}$$

which shows that the z-transform of the discrete-time sampled signal is obtained from the Laplace transform of the impulse train by a simple change of variable as follows:

$$F(z) = F^*(s)\Big|_{e^{sT} = z}. \tag{58}$$

The Laplace transform of the product (55) can be written as the convolution

$$F^*(s) = \frac{1}{2\pi j}\int_C \frac{F(p)}{1 - e^{-(s-p)T}}dp + \tfrac{1}{2}f(0^+), \tag{59}$$

where the contour C extends from the bottom to the top of the complex plane to the right of the singularities of $F(p)$ and to the left of the singularities of the impulse train. Assume for simplicity that $F(p)$ is a proper rational function with all poles in the left-half p-plane. Then the contour of integration can be closed to the left, yielding [2]

$$F^*(s) = \sum\left[\text{residues } \frac{F(p)}{1 - e^{-(s-p)T}}\right]_{\text{poles of } F(p)}, \tag{60}$$

or to the right, yielding

$$F^*(s) = \frac{1}{T}\sum_{k=-\infty}^{\infty} F\left[s + jk\frac{2\pi}{T}\right] + \tfrac{1}{2}f(0^+). \tag{61}$$

Equation (60) is equivalent to the partial fraction expansion of (24), while (61), the so-called aliasing formula, demonstrates the effect of frequency components outside the baseband. Formula (61) can be derived also by Fourier series methods.

C. The Impulse Transfer Function

If the impulse train (56) is applied to an ordinary continuous-time network with impulse response $h(t)$, there results the continuous-time signal

$$y(t) = \sum_{n=0}^{\infty} f(nT^+)h(t - nT). \tag{62}$$

Samples of $y(t)$ are, therefore, given by

$$y(mT^+) = \sum_{n=0}^{\infty} f(nT^+)h[(m - n)T^+], \tag{63}$$

which is a discrete convolution of the form (38). It follows that

$$Y(z) = F(z)H(z), \tag{64}$$

where $H(z)$ is the z-transform of the discrete-time signal, which is obtained by sampling the impulse response of the continuous-time filter H. This digital filter $H(z)$ is the effective discrete-time transfer function of a pulsed network and is useful in the analysis of sampled-data control systems.

D. Reconstruction Filters

The relation (62) represents the process of producing a continuous-time signal from a discrete-time signal and, as such, represents an operation that is the opposite of sampling. As noted before, however, this process will not in general provide an exact inverse to the sampling operation, since some information is destroyed by sampling.

The Laplace transform of (62) yields

$$Y(s) = F^*(s)H(s), \tag{65}$$

which shows that $H(s)$, if it is to be an effective reconstruction filter, should have a low-pass characteristic in order to select the baseband alias of $F^*(s)$, that is, the $k=0$ term in the aliasing formula (61).

It may be required that the reconstructed signal agree with the discrete-time signal at sample points, in which case the reconstruction filter $H(s)$ is called an interpolation filter. From (64), this means that

$$F(z) = F(z)H(z) \tag{66}$$

for every $F(z)$ or

$$H(z) = 1. \tag{67}$$

A simple example of an interpolation filter is the zero-order hold, which produces the piecewise constant reconstruction

$$y(t) = f_n, \qquad nT \leq t < (n + 1)T. \tag{68}$$

In this case, the impulse response is

$$H(t) = \begin{cases} 1 & 0 \leq t < T. \\ 0 & \text{elsewhere} \end{cases} \tag{69}$$

for which (67) can be checked.

VI. Conclusion

In this paper, a summary of techniques that may be used for digital processing of signals has been presented. Digital filtering is based on the z-transform, much as analog filter theory is based on the Laplace transform. The concepts of impulse and frequency response have their digital counterparts. The impulse response of a digital filter is defined as its response to a sequence $1, 0, 0, \cdots$, as input. The z-transform of the impulse response is the transfer function of the filter and if the transfer function is evaluated along the unit circle, the frequency response of the filter is obtained. An important consequence is that frequency response is a continuous but periodic function of frequency.

For those impulse responses corresponding to sums of exponentially decaying polynomials and sinusoidal sequences, the z-transform can be expressed in a closed form as a rational function of z, reminiscent of the transfer function of an RLC filter. Such an impulse response leads to the recursive digital filter, whose implementation is in terms of difference equations. Impulse responses of finite duration are usually realized as weighted sums of the output of tapped delay lines and are called nonrecursive filters. There are various forms of digital filters that realize the same transfer function. There are practical techniques for designing both types of filters. Currently, recursive filters are more practical for meeting hardware requirements for many bandpass, bandstop, low-pass, and high-pass filters, but both recursive and nonrecursive filters are finding wide application in waveform processing by computer, where very complicated frequency response functions are required.

One possible advantage of digital filtering over analog filtering may be the high precision obtainable. The sources of error due to finite length arithmetic are somewhat understood and can be minimized in many cases by analytical considerations. It is expected that the technique of digital filtering will grow rapidly in importance in the future, especially as the size and cost of discrete components continues to decrease.

REFERENCES

[1] K. Steiglitz, "The equivalence of digital and analog signal processing," *Information and Control*, vol. 8, pp. 455–467, October 1965.

[2] H. Freeman, *Discrete-Time Systems.* New York: Wiley, 1965, ch. 4.

[3] C. M. Rader and B. Gold, "Digital filter design techniques in the frequency domain," *Proc. IEEE*, vol. 55, pp. 149–171, February 1967.

[4] J. F. Kaiser, "Digital filters," in *System Analysis by Digital Computer*, F. F. Kuo and J. F. Kaiser, Eds. New York: Wiley, 1966, ch. 7.

[5] R. M. Golden and J. F. Kaiser, "Design of wideband sampled-data filters," *Bell Sys. Tech. J.*, vol. 43, pt. 2, pp. 1533–1546, July 1964.

[6] B. Gold and C. M. Rader, *Digital Processing of Signals.* New York: McGraw-Hill, chs. 2–4 (in press).
[7] J. F. Kaiser, "Design methods for sampled-data filters," *1963 Proc. 1st Allerton Conf. on Circuit and System Theory* (Monticello, Ill., November 1963), pp. 221–236.
[8] W. K. Linvill, "Sampled-data control systems studied through a comparison of sampling with amplitude modulation," *Trans. AIEE*, vol. 70, pt. 2, pp. 1779–1788, 1951.
[9] R. M. Golden, "Digital filter synthesis by sampled-data transformation," this issue, pp. 321–329.
[10] W. T. Cochran *et al.*, "What is the fast Fourier transform?" *IEEE Trans. Audio and Electroacoustics*, vol. AU-15, pp. 45–66, June 1967.
[11] T. G. Stockham, "High-speed convolution and correlation," *1966 Spring Joint Computer Conf., AFIPS Proc.*, vol. 28, pp. 229–233.
[12] H. D. Helms, "Fast fourier transform method for computing difference equations and simulating filters," *IEEE Trans. Audio and Electroacoustics*, vol. AU-15, pp. 85–90, June 1967.
[13] R. Boxer and S. Thaler, "A simplified method of solving linear and nonlinear systems," *Proc. IRE*, vol. 44, pp. 89–101, January 1956.
[14] H. D. Helms, "Nonrecursive digital filters: design methods for achieving specifications on frequency response," this issue, pp. 336–342.
[15] J. B. Knowles and R. Edwards, "Effect of a finite-word length computer in a sampled-data feedback system," *Proc. IEE* (London), vol. 112, pp. 1147–1207, June 1965.
[16] ——, "Complex cascade programming and associated computational errors," *Electronics Lett.*, vol. 1, pp. 160–161, August 1965.
[17] J. B. Knowles and E. M. Olcayto, "Coefficient accuracy and digital filter response," *IEEE Trans. Circuit Theory*, vol. CT-15, pp. 31–41, March 1968.
[18] J. F. Kaiser, "Some practical considerations in the realization of linear digital filters," *Proc. 3rd Allerton Conf. on Circuit and System Theory* (Monticello, Ill., October 1965), pp. 621–633.
[19] B. Gold and C. M. Rader, "Effects of quantization noise in digital filters," *1966 Spring Joint Computer Conf., AFIPS Proc.*, vol. 28, pp. 213–219.
[20] ——, "Effects of parameter quantization on the poles of a digital filter," *Proc. IEEE*, vol. 55, pp. 688–689, May 1967.
[21] R. B. Blackman, *Linear Data-Smoothing and Prediction in Theory and Practice.* Reading, Mass.: Addison-Wesley, 1965, pp. 75–81.

GENERAL BOOKS

[22] J. R. Ragazzini and G. F. Franklin, *Sampled-Data Control Systems.* New York: McGraw-Hill, 1958.
[23] E. I. Jury, *Sampled-Data Control Systems.* New York: Wiley, 1958.
[24] J. Tou, *Digital and Sampled-Data Control Systems.* New York: McGraw-Hill, 1959.
[25] B. Kuo, *Analysis and Synthesis of Sampled-Data Control Systems.* Englewood Cliffs, N. J.: Prentice-Hall, 1963.
[26] E. I. Jury, *Theory and Application of the Z-Transform Method.* New York: Wiley, 1964.
[27] D. P. Lindorff, *Theory of Sampled-Data Control Systems.* New York: Wiley, 1965.
[28] A. J. Monroe, *Digital Processes for Sampled-Data Systems.* New York: Wiley, 1962.