HP 98828A
Digital Filter Design

for the HP 9826 and 9836 Computers

HEWLETT
PACKARD

HP 98828A Digital Filter Design

for the HP 9826 and 9836 Computers

Manual Part No. 98828-14111

Disc Part No. 98828-14114

Hewlett-Packard Desktop Computer Division
3404 East Harmony Road, Fort Collins, Colorado 80525

Printing History

New editions of this manual will incorporate all material updated since the previous edition. Update packages may be issued between editions and contain replacement and additional pages to be merged into the manual by the user. Each updated page will be indicated by a revision date at the bottom of the page. A vertical bar in the margin indicates the changes on each page. Note that pages which are rearranged due to changes on a previous page are not considered revised.

The manual printing date and part number indicate its current edition. The printing date changes when a new edition is printed. (Minor corrections and updates which are incorporated at reprint do not cause the date to change.) The manual part number changes when extensive technical changes are incorporated.

July 1982...First Edition

Table of Contents

Preface

This software applications pack provides the following design capabilities:

- automated design of finite impulse response (FIR) and infinite impulse response (IIR) digital filters to meet frequency domain specifications,
- design of FIR filters to meet frequency sampling specifications,
- design of FIR filters from truncated impulse responses,
- least squares design of FIR filters,
- minmax design of FIR filters,
- transformations of analog systems to digital systems by the methods of impulse invariance, covariance invariance, and bilinear-z,
- least squares design of IIR digital filters.

Once designed, a filter may be analyzed by computing, tabulating, and graphing (on CRT graphics, thermal printer, or HP pen plotter) the following data:

- filter coefficients,
- filter impulse response,
- filter frequency response (magnitude and phase; linear and log scales; analog and digital overplot), and
- filter pole-zero locations.

Filter coefficients may be rounded to a specified number of (mantissa) significant digits and re-analyzed.

This pack also provides the following additional analysis features:

- impulse and frequency response analysis of user-specified autoregressive (AR), moving average (MA), and autoregressive moving average (ARMA) discrete-time systems,
- fast Fourier transform (FFT) analysis of real sequences of length 256, and
- factoring of polynomials for their zeros.

The pack is a menu-oriented, prompt-driven, interactive program that permits the user to select design and analysis options that exercise the capabilities enumerated above. To follow the instructions in this manual, you will need a working knowledge of the HP 9826 (or 9836) computer and rudimentary knowledge of analog or digital system theory.

Chapter 1
Introduction

The HP 9826/9836 Digital Filter Design software is a set of BASIC programs for the design, analysis, and study of digital filters and discrete-time systems. The set of programs is menu-oriented and prompt-driven for interactive design and analysis.

Input design and analysis parameters may be entered as frequency domain specifications, as analog filter specifications, or as digital filter specifications. In many cases the parameters may be entered from a data file, or manually from keyboard or CRT graphics. CRT, thermal printer, and pen plotter graphics of filter coefficients, impulse responses, and frequency responses are provided. All plots are normalized.

Design Options

Design options offered by this software include:

- automated design of finite impulse response (FIR) digital filters to meet frequency domain specifications,
- design of FIR filters to meet frequency sampling specifications,
- design of FIR filters from truncated impulse responses,
- least squares design of FIR filters,
- minmax design of FIR filters,
- transformation of analog systems by the methods of impulse-invariance, covariance-invariance, and bilinear-z,
- automated design of infinite impulse response (IIR) filters of the Butterworth and Tchebyshev variety to meet frequency domain specifications,
- least squares design of IIR digital filters.

Analysis Options

Once designed, filters may be analyzed by computing, tabulating, and graphing (on CRT graphics, thermal printer, or HP pen plotter) the following data:

- filter coefficients,
- filter impulse response,
- filter frequency response (magnitude and phase; linear or log scale; analog and digital overplot), and
- filter pole-zero locations.

Filter coefficients may be rounded to a desired number of (mantissa) significant digits and re-analyzed.

Additional Analysis Features

The following very useful analysis and utility functions are also provided in this software:

1. impulse and frequency response analysis of user-specified autoregressive (AR), moving average (MA), and autoregressive moving average (ARMA) discrete-time systems,
2. fast Fourier transform (FFT) analysis of real sequences of length 256, and
3. factoring of transfer function polynomials for their poles and zeros.

Functions 1 and 2 are exercised by making the software think it is analyzing a digital filter.

System Configuration

This package runs on an HP 9826 or 9836 Computer (256K bytes of available user R/W memory) and:

- a BASIC Software System – required,
- any HPGL plotter (i.e., 9872) – optional (e.g., 9876, 2671, etc.),
- any 9826/36 compatible printer – optional.

Comment of Terminology

Throughout this manual we use the terms digital filter, digital system, discrete-time filter, and discrete-time system interchangeably. All of these terms correctly suggest that time is discrete, as for example in sampled-data systems. However, strictly speaking, digital filters and systems operate with finite arithmetics, while discrete-time filters and systems may operate with infinite arithmetics.

Chapter 2
Theory and Application of Digital Filtering

Digital filtering is a computational process or algorithm that maps input sequences of numbers into output sequences of numbers. The input is often a sampled-data sequence generated by an A/D converter. The algorithm, itself, may implement a function corresponding to:

- lowpass filtering for smoothing and bandwidth reduction,
- bandpass filtering for control of adjacent channel interference,
- highpass filtering for rejection of low-frequency noise,
- bandstop and multiband filtering for noise suppression,
- differentiation for demodulation,
- integration for averaging, or
- interpolation or decimation for sampling rate modification.

The output sequence may be used as it stands, or it may be post-processed. Typical post-processing operations are:

- fast Fourier transform (FFT) processing for spectrum analysis,
- model fitting for data compression, and
- D/A conversion for generation of an analog control or communication signal.

These remarks are given a pictorial representation in Figure 1. In the figure an analog signal x(t) is A/D converted to generate the digital signal x_k. This signal is digitally filtered by the algorithm called H(z) to produce the output sequence y_k. The output may be D/A converted to produce the analog signal $\hat{y}(t)$. One of the key ideas in digital filtering is that functions normally achieved with analog filtering may often be achieved with digital filtering. That is, by properly designing the algorithm (or filter) H(z), we may make $\hat{y}(t)$ look much like y(t).

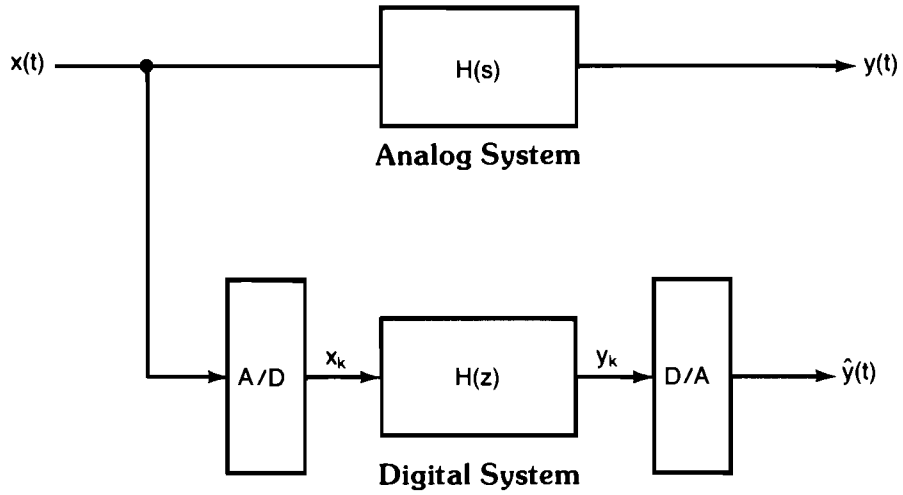


Figure 1. Digital Filtering Illustrated

Among the advantages offered by digital filtering over analog filtering, we list the following:

- insensitivity to thermal variations in component values,
- elimination of need to periodically calibrate,
- stability and repeatability of performance from unit to unit,
- programmability for design flexibility,
- precision limited only by computational word length, and
- small size, low power, and low cost with LSI and VLSI integration.

Example: Rejection of a 60 Hz Interference

Suppose you need to reject an energetic 60 Hz sinusoidal interference from a 10 Hz signal of interest. (You may need to estimate the amplitude and phase of the 10 Hz signal.) An obvious solution is to build a bandstop or notch filter that rejects the 60 Hz component while passing the 10 Hz signal with little or no distortion. A commonly used RC notch filter is the twin-T shown in Figure 2(a). This filter derives its notch characteristic from a pair of transmission zeros located in the s-plane on the imaginary axis at $s = j2\pi(60)$. It also has two poles on the negative real axis that only slightly affect the frequency response. See Figure 2(a). When an input waveform consisting of 60 Hz and 10 Hz components is applied to the input terminals of this filter, the twin-T rejects (in steady state) the 60 Hz component and passes the 10 Hz component as illustrated. To change the center frequency of the notch, we must change the resistor and capacitor values. The position of the notch is very sensitive to component values, so precision components must be used.

A digital filtering solution is illustrated in Figure 2(b). The input signal is sampled with an A/D converter at a rate of 500 samples/second. That is to say, the input signal is read at 2 millisecond intervals to produce an input sampled-data sequence to the digital filter. The digital filter is constructed from two registers (or memory elements), a digital multiplier, and a 3-input adder. The current digital output y_n in the output sequence is formed by adding the current input x_n with the previous inputs, $x_{n-1}*(-1.46)$ and x_{n-2}. When this filter is excited with a sequence of sampled-data that arrives at the rate of 500 samples/second, the 60 Hz component is eliminated (in steady state). The filter in Figure 2(b) derives its notch characteristics from a pair of z-plane zeros, located on the unit circle at $z = \exp(j2\pi(60/500))$. Figure 2(b) illustrates two cycles of the sampled input and the sampled output. Note the rejection of the 60 Hz component.

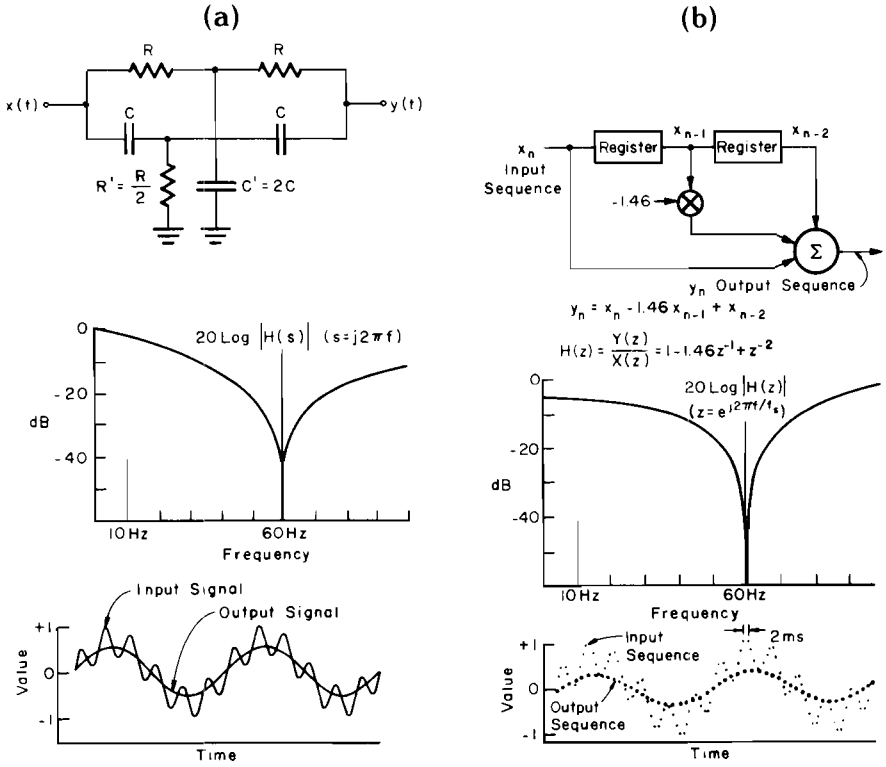


Figure 2. Analog and Digital Notch Filtering

The notch center frequency may be changed in two ways in a digital filter. The first way is to change the multiplication coefficient. By changing it from -1.46 to -1.62 we move the notch to 50 Hz. (This is important for your next trip to Europe.) The second way is to change the sampling rate. By reducing it to 417 samples/second, we also move the notch to 50 Hz.

One final comment on this example. The digital filter of Figure 2(b) works just like a FORTRAN DO Loop. To see this, write the input/output relation for the filter as follows:

$$y_n = x_n - 1.46x_{n-1} + x_{n-2}$$

This difference equation characterizes the filter. Assume this equation (or algorithm) is to be implemented on a computer for $n = 1,2,3,\ldots,1000$, with initial conditions $x_0 = 0$ and $x_{-1} = 0$. Here is a loop that does it:

```
      XMIN1 = 0
      XMIN2 = 0
      DO 3 N = 1,1000
      READ (5, ) XN
      YN = XN - 1.46*XMIN1 + XMIN2
      XMIN2 = XMIN1   (Exchange Statement)
      XMIN1 = XN   (Exchange Statement)
    3 CONTINUE
```

Can you see that the registers (or delay elements) in the circuit diagram of Figure 2(b) work just like the exchange statements in the FORTRAN code?

Classes of Filters

Digital filters are classified as lowpass, bandpass, highpass, and bandstop according to their frequency response characteristics. More than this, however, they are classified according to their algorithmic structure. To establish our algorithmic classification we need to consider the folowing general input/output description for a (finite-dimensional) digital filter:

$$y_n = -a_1 y_{n-1} - a_2 y_{n-2} - \ldots - a_N y_{n-N} + b_0 x_n + b_1 x_{n-1} + \ldots + b_M x_{n-M}$$

This difference equation tells us that current values of the output y_n are autoregressions (or feedbacks) over past outputs plus moving averages (or feedforwards) over current and past inputs.

Autoregressive (AR), All-Pole (AP), or Infinite Impulse Response (IIR)

When all the b_i coefficients in the general input/output equation are zero except for the b_0 term, we have the following special case:

$$y_n = -a_1 y_{n-1} - a_2 y_{n-2} - \ldots - a_N y_{n-N} + b_0 x_n$$

Such a digital filter is called an AR, AP, or IIR digital filter of order N. Two circuit diagrams for this filter are illustrated in Figures 3(a) and (b). In the figures, the boxes z^{-1} store values for one sample period so that if x_n is the input to the box, then x_{n-1} is the output. Thus, the z^{-1} are the registers discussed previously. The arrow paths a_i denote multiplication: if x_n passes through a path, it arrives at the next node as $a_i x_n$. The nodes are summing nodes.

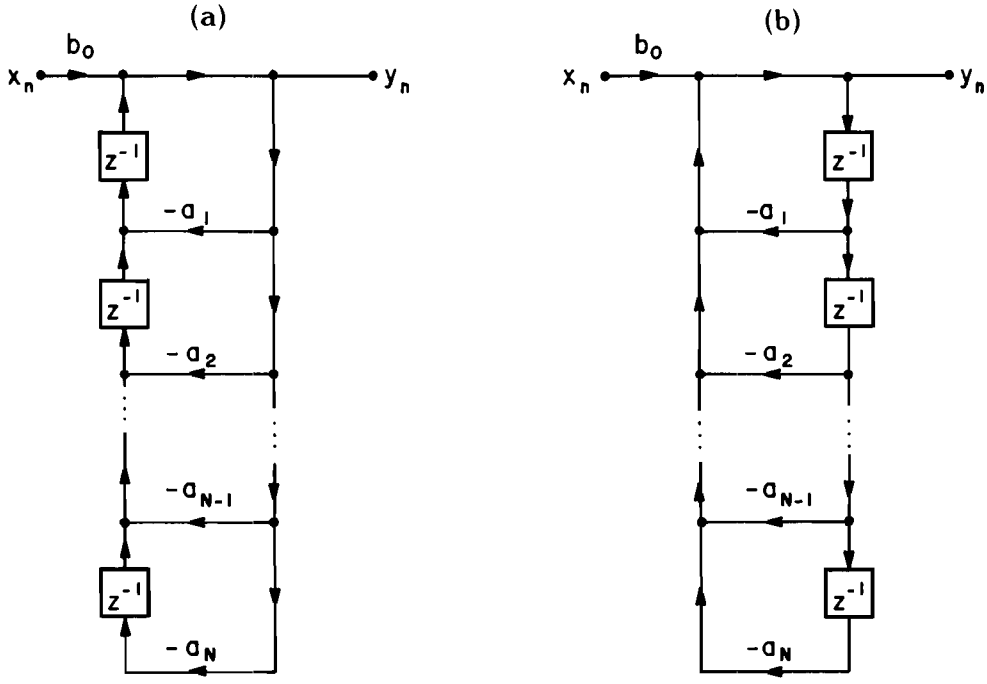


Figure 3. Circuit Diagrams for AR, AP, or IIR Digital Filters

Moving Average (MA), All-Zero (AZ), or Finite Impulse Response (FIR)

When all of the a_i coefficients in the general input/output equation are zero, we have the following special case:

$$y_n = b_0x_n + b_1x_{n-1} + \ldots + b_Mx_{n-M}$$

Such a digital filter is called an MA, AZ, or FIR digital filter of order M. Two circuit diagrams for this filter are illustrated in Figures 4(a) and (b).

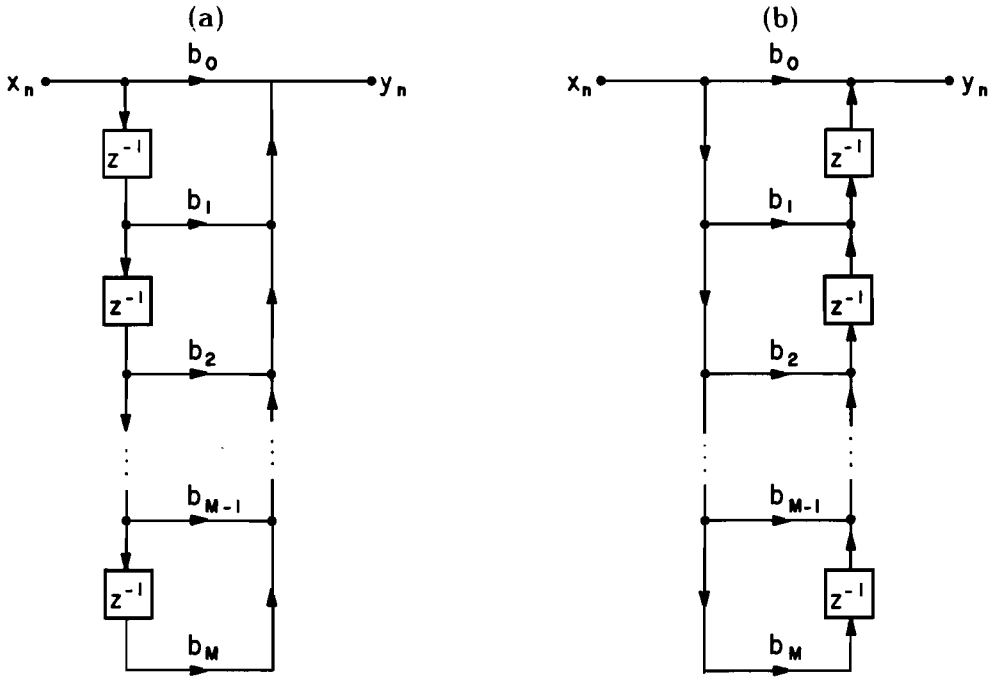


Figure 4. Circuit Diagrams for MA, AZ, or FIR Digital Filters

Autoregressive Moving Average (ARMA), Pole-Zero (PZ), or Infinite Impulse Response (IIR)

When at least a_N and b_M, and generally all of the other a_i and b_i coefficients are nonzero, we have the general input/output relation. Such a filter is called an ARMA, PZ, or IIR filter of order (N,M). The a_i coefficients are called AR or feedback coefficients and the b_i coefficients are called MA or feedforward coefficients. See Figures 5(a) and (b) for circuit diagrams.

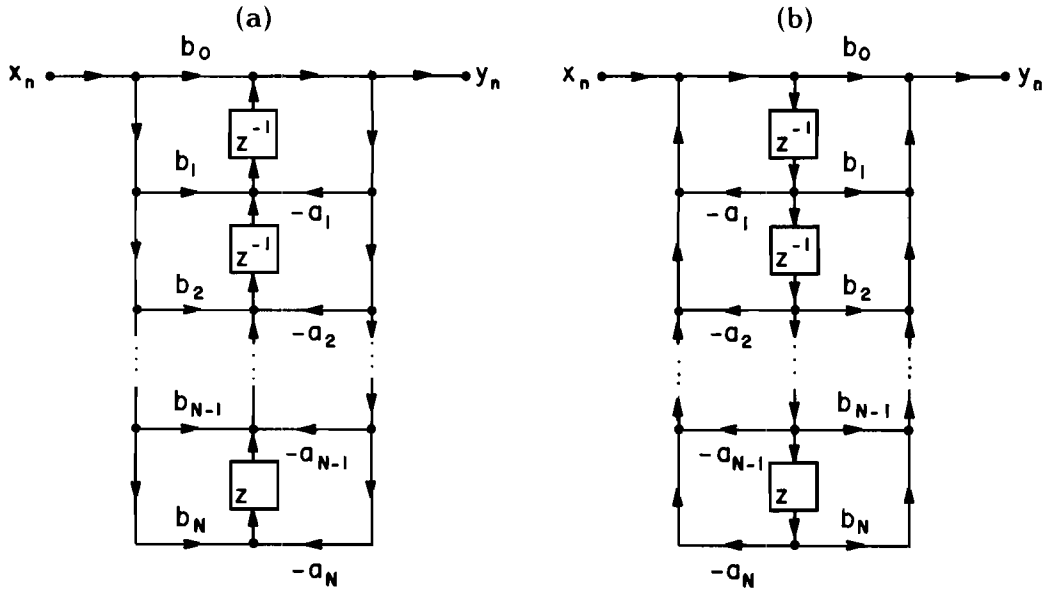


Figure 5. Circuit Diagrams for ARMA, PZ, or IIR Digital Filters

Trade-offs Between IIR and FIR Digital Filters

When scaling in finite wordlength applications is thought to be a problem, then FIR filters are often preferred over IIR filters. When machine wordlengths are short, FIR filters perform closer to their infinite wordlength counterparts than do IIR filters. However, when speed is an issue, then IIR filters are often preferred over FIR filters because they can be designed to meet specifications with fewer filter coefficients. This means that fewer multiplies and adds are required per input sample; consequently, higher throughput rates (or processing bandwidths) may be achieved.

Some applications, such as LPC coding of speech, lead naturally to FIR filters as the appropriate whiteners or predictors. Other applications, such as the approximation of analog systems, lead naturally to IIR filters. Thus, the choice is highly problem and constraint dependent, and the designer must proceed with his choice accordingly. Often a choice is made only after trying several designs of each type. This pack is ideally suited for interactive assaying of alternatives.

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

Analysis of Digital Filters

There are three very special input sequences in the study of digital filters. By analyzing the response of a filter to one or more of these sequences, we gain a great deal of insight into the properties of the filter when excited by more general sequences. In this software, we provide capabilities for the analysis of impulse responses and frequency responses for digital filters. In addition, by making the software think you are entering FIR filter coefficients, you may actually discrete Fourier transform (DFT) real signal sequences.

Unit Impulse Sequence

This input sequence has the value 1 at n = 0 and the value zero at all other values of n. Think of n as time. This sequence is to digital filter theory what the Dirac delta function (or unit impulse) is to analog filter theory. We denote the unit impulse sequence as follows:

$$x_n = \delta_n = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases}$$

This sequence is illustrated in Figure 6.

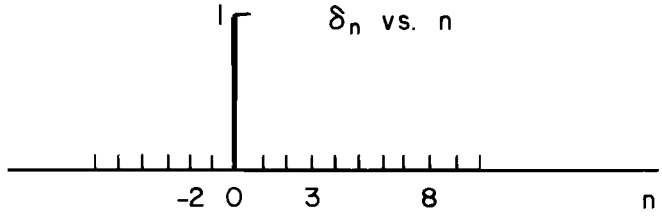


Figure 6. Unit Impulse Sequence

Power Series Sequence

This is the input sequence:

$$x_n = z^n, \text{ for all } n$$

with z a complex number. As we shall see, this sequence reproduces itself when applied to a digital filter. The power series sequence is illustrated in Figure 7 for z real and $0 < z < 1$.

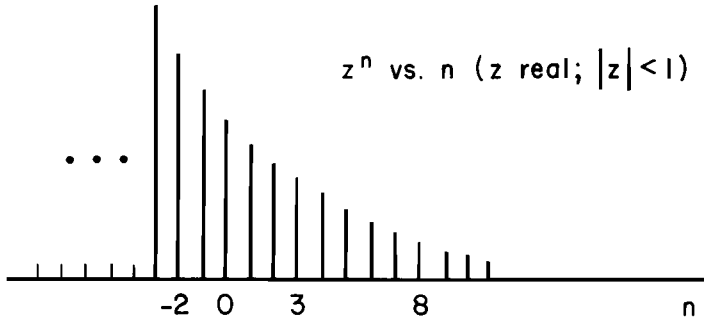


Figure 7. Power Series Sequence

Complex Exponential Sequence

When z takes on the special value $z = \exp(j\theta)$, and $j = \sqrt{-1}$, then the power series sequence becomes the following complex exponential sequence:

$x_n = \exp(jn\theta)$, for all n.

The discrete-time sine and cosine sequences are obtained by taking the imaginary and real parts of this complex exponential sequence:

$\sin(n\theta) = \text{Im}(\exp(jn\theta))$, for all n

$\cos(n\theta) = \text{Re}(\exp(jn\theta))$, for all n

See Figure 8 for an illustration of these sequences.

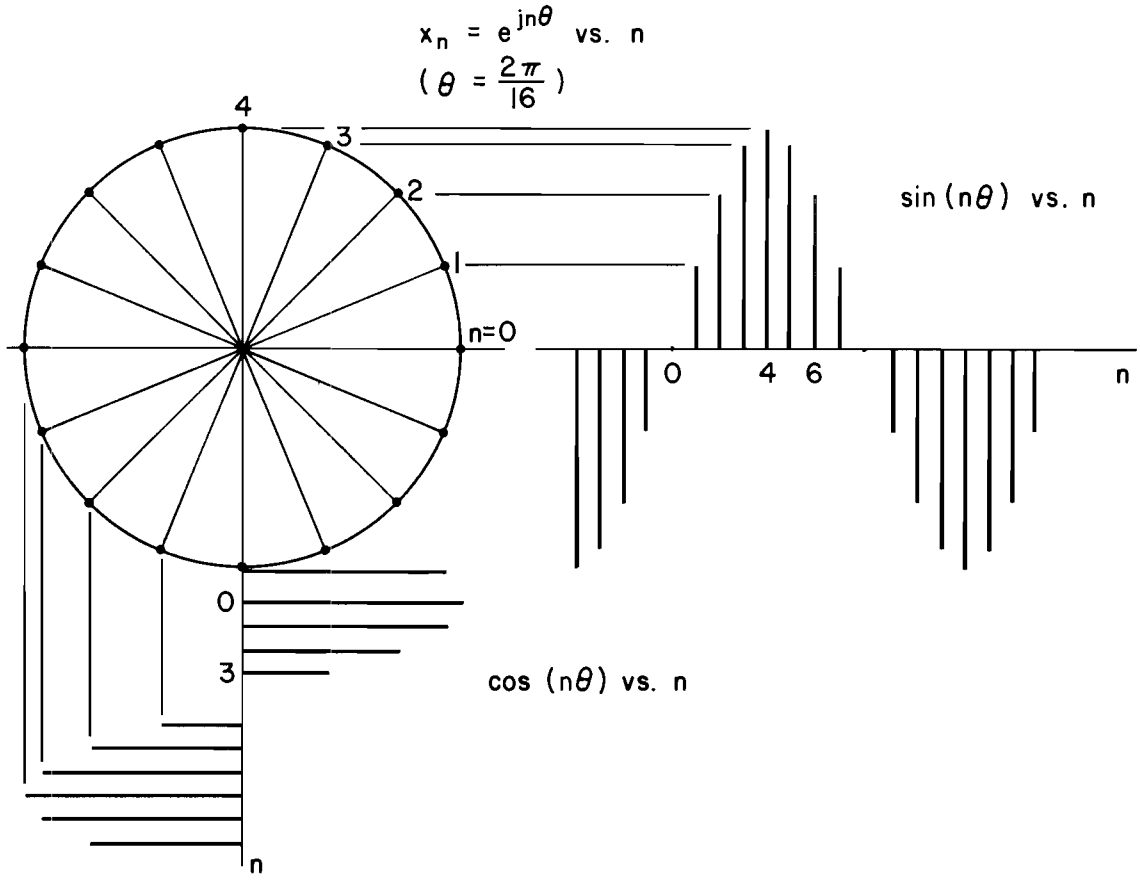


Figure 8. Complex Exponential, Sine, and Cosine Sequences

Perhaps most importantly of all, from the point of view of physical interpretation, the complex exponential sequence models the sampled-data version of the analog complex exponential. Here's the idea. Begin with the analog complex exponential:

$$x(t) = \exp(j2\pi ft) \text{ , for all } t$$

and its imaginary and real parts:

$$\sin(2\pi ft) = \text{Im}\,(\exp(j2\pi ft)) \text{ , for all } t$$
$$\cos(2\pi ft) = \text{Re}\,(\exp(j2\pi ft)) \text{ , for all } t$$

These latter two analog signals are recognized as sines and cosines of (angular) frequency f Hz. If the analog complex exponential, or equivalently the analog sine or cosine, is sampled at the rate:

$$f_s = 1/T$$

where T is the sampling interval, the following digital sequence results:

$$x_n = \exp(jn\theta)$$
$$\theta = 2\pi fT = 2\pi f/f_s$$

So once the frequency f (in Hz) of the analog signal has been specified and a sampling rate f_s (in Hz) – or sampling interval T (in seconds) – has been selected, the value of θ is pinned down. If the sampling rate remains fixed and the frequency f changes, then so does θ. See Figure 9 for a sampling illustration.

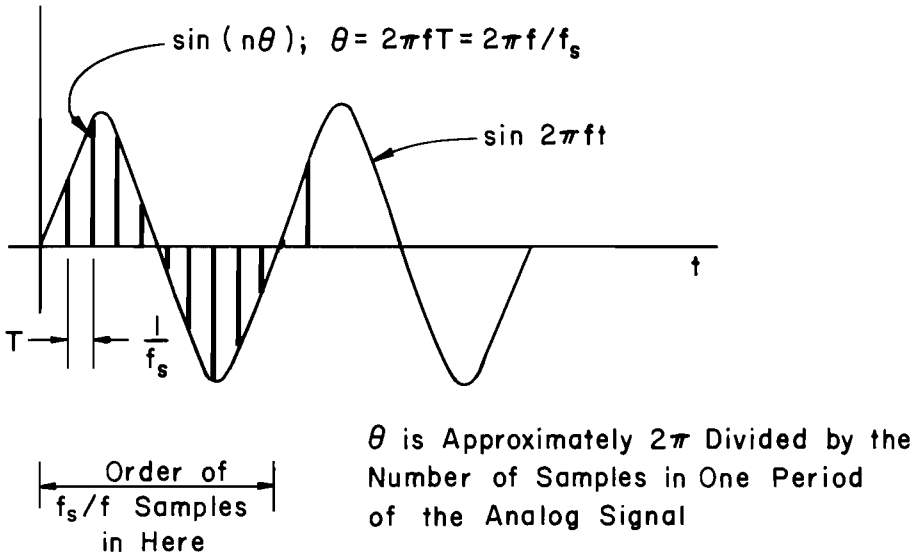


Figure 9. Sampling Illustrated

We now turn to a discussion of digital filter response to these input sequences.

Impulse Response and Convolution

The response of a digital filter to an arbitrary input x_n is described by the convolutional sum:

$$y_n = \sum_{k=0}^{\infty} h_k x_{n-k}$$

The sequence h_k is called the (unit) impulse response because when the input x_n is the unit impulse sequence δ_n the output is:

$$y_n = h_n$$

The impulse response is fundamental because it characterizes filter response to arbitrary inputs. A typical impulse response is depicted in Figure 10.

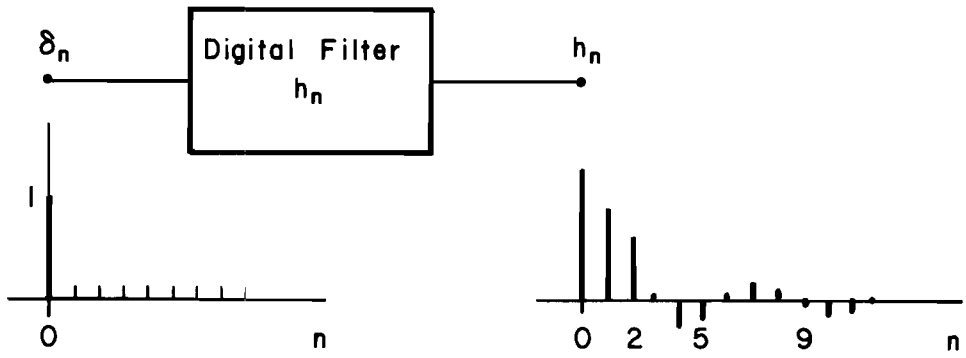


Figure 10. Impulse Response Illustrated

Writing out a few terms of the convolutional sum, we can see how convolution works:

$$y_n = h_0 x_n + h_1 x_{n-1} + \ldots + h_k x_{n-k} + \ldots + h_n x_0 + \ldots$$

The scaling of x_k before it shows up in the output y_n depends on the difference (k) between the time(n − k) when the input was applied and the time (n) when its effect on the output is read. When the digital filter is autoregressive, moving average, then the impulse response obeys the ARMA difference equation:

$$h_n = -a_1 h_{n-1} - a_2 h_{n-2} - \ldots - a_N h_{n-N} + b_0 \delta_n + \ldots + b_M \delta_{n-M}$$

This is, in fact, the difference equation evaluated in the software pack to get the impulse response.

Substitution of the expression for h_n into the convolutional sum reproduces the general input/output difference equation:

$$y_n = -a_1 y_{n-1} - \ldots - a_N y_{n-N} + b_0 x_n + \ldots + b_M x_{n-M}$$

Try it.

Z-Transform and Transfer Function

Consider the impulse response sequence h_n, $n = 0,1,2,\ldots$. Define the z-transform of this sequence to be the infinite power series

$$H(z) = \sum_{k=0}^{\infty} h_k z^{-k}$$

We assume this series is finite for some values of z, and give it the very descriptive name, Transfer Function. Let's see if the name is deserved.

To begin, consider the response of a digital filter to a power series input. Using the convolution formula, it is not hard to show that the response is[1]:

$$y_n = H(z)x_n \text{ , for all n}$$
$$x_n = z^n$$

Thus the transfer function tells how the power series sequence gets transferred from input to output, where it shows up again as a power series sequence scaled by H(z).

More generally, we may write from the convolution sum that:

$$Y(z) = H(z)\, X(z)$$

where X(z) and Y(z) are z-transforms of the input and output:

$$Y(z) = \sum_{k=-\infty}^{\infty} y_k z^{-k}$$

$$X(z) = \sum_{k=-\infty}^{\infty} x_k z^{-k}$$

The transfer function H(z) works multiplicatively in the z-transform domain. This is illustrated in Figure 11.

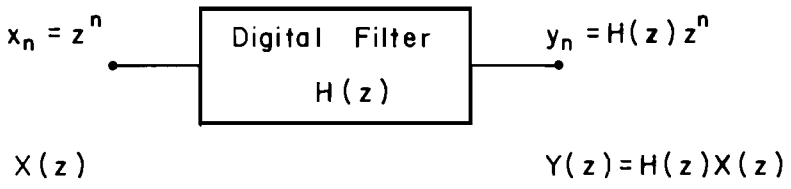


Figure 11. Transfer Function

1 Don't be thrown by a result like this. But be warned that it only holds for $x_n = z^n$.

When the impulse response sequence obeys the ARMA difference equation discussed in the previous section, then the transfer function takes on the special form:

$$H(z) = \frac{B(z)}{A(z)}$$

$$B(z) = b_0 + b_1 z^{-1} + b_2 z^{-2} + \ldots + b_M z^{-M}$$

$$A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_N z^{-N}$$

This is the transfer function of an ARMA filter. We may, therefore, write:

$$A(z)\,Y(z) = B(z)\,X(z)$$

This is the z-transform domain version of the input/output difference equation we started with in the Classes of Filters section.

If the digital filter is AR, then the transfer function takes the so-called AR or all-pole form:

$$H(z) = \frac{b_0}{A(z)}$$

If it is MA, then the transfer function is the MA or all-zero form:

$$H(z) = B(z)$$

Complex Exponential and Frequency Response

Consider a digital filter with transfer function H(z). Let the input sequence be the complex exponential sequence exp(jnθ). The output sequence is the scaled and phased complex exponential:

$$y_n = H(e^{j\theta})e^{jn\theta}\text{, for all n}$$

The function $H(e^{j\theta})$ is called the complex exponential response and is defined as follows:

$$H(e^{j\theta}) = \sum_{n=0}^{\infty} h_n e^{-jn\theta}$$

Of course h_n is the impulse response of the digital filter. We may think of h_n as a Fourier series coefficient of the periodic function $H(e^{j\theta})$, or of $H(e^{j\theta})$ as the Fourier transform of the discrete-time sequence $\{h_n\}$.

We often use the magnitude and phase representation of the complex exponential response to write:

$$y_n = |H(e^{j\theta})|e^{j(n\theta + \arg H(e^{j\theta}))}$$

$$H(e^{j\theta}) = |H(e^{j\theta})|e^{j\,\arg H(e^{j\theta})}$$

This form shows that the output is the same as the input, but scaled by the magnitude function $|H(e^{j\theta})|$ and phased by the phase function arg $H(e^{j\theta})$. If the digital filter has real coefficients, the response to the cosinusoidal input:

$$x_n = \cos(n\theta)\text{, for all n}$$

is consinusoidal also:

$$y_n = |H(e^{j\theta})|\cos(n\theta + \arg H(e^{j\theta}))$$

When the input complex exponential arises as a sampled-data complex exponential, we have the following input and output sequences:

$$x_n = e^{j2\pi fnT}, \text{ for all } n$$
$$y_n = H(e^{j2\pi fT})e^{j2\pi fnT}, \text{ for all } n$$

Now we call $H(e^{j2\pi fT})$ the complex frequency response because it is an explicit function of the underlying temporal frequency f(Hz) – and of course the sampling interval T. See Figure 12 for an illustration of how the complex frequency response works.

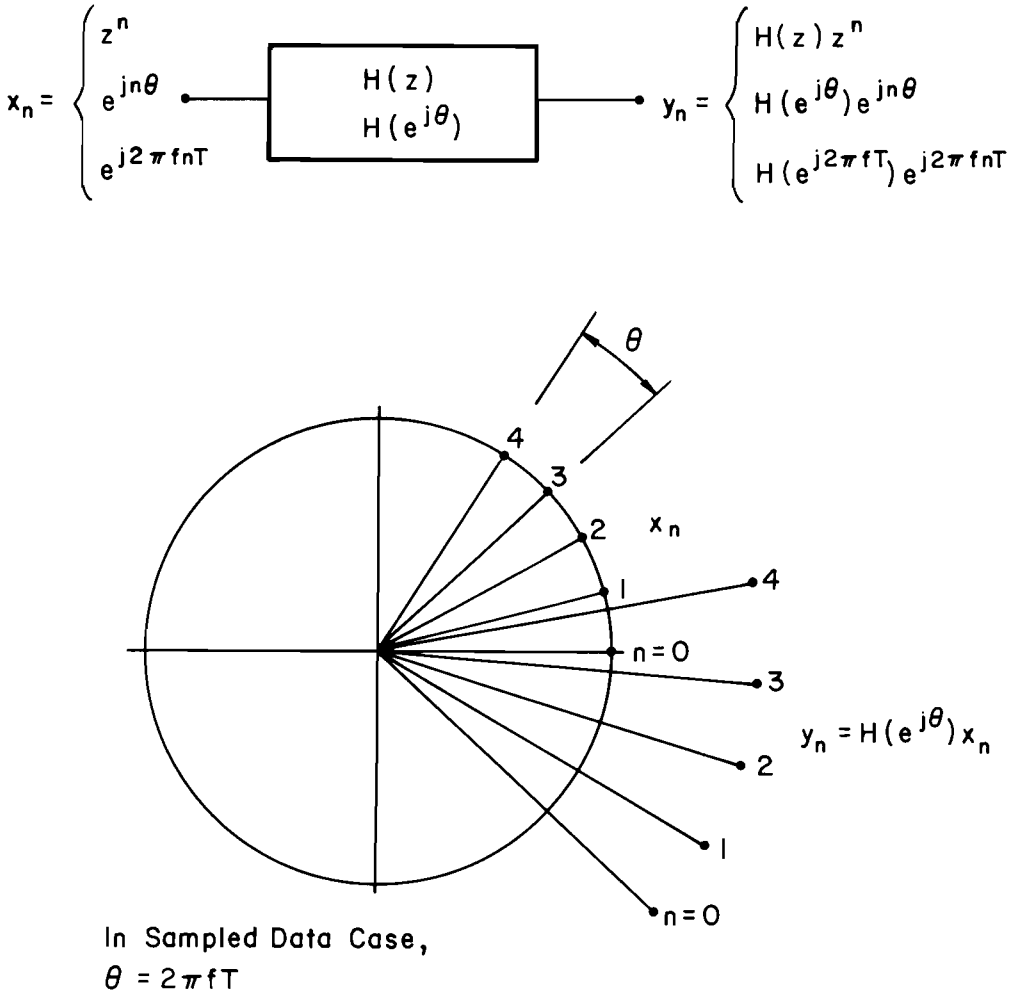


Figure 12. Complex Exponential and Complex Frequency Response

Computation of Complex Frequency Response and the Role of the Unit Circle

For digital filters with real coefficients, it is easy to show that the complex frequency response $H(e^{j2\pi fT})$ has the following symmetries:

$$|H(e^{j2\pi fT})| = |H(e^{j2\pi(1/T - f)T})|,\ 0 < f < 1/T$$

$$\arg H(e^{j2\pi fT}) = -\arg H(e^{j2\pi(1\ T - f)T})$$

$$H(e^{j2\pi(f \pm r\ T)T}) = H(e^{j2\pi fT}),\ r = 0,1,2, \ldots$$

The last of these symmetries says the digital filter doesn't know the frequency $f \pm r/T$ (or $f \pm rf_s$) from the frequency f. So the sampling frequency f_s places an upper limit on the discernible frequency in sampled-data systems. The first two symmetries lend a special significance to the foldover-frequency 1/2T (or $f_s/2$). There is no information in a plot of the complex frequency response for values of frequency greater than the foldover frequency $f_s/2$. This is illustrated in Figure 13.

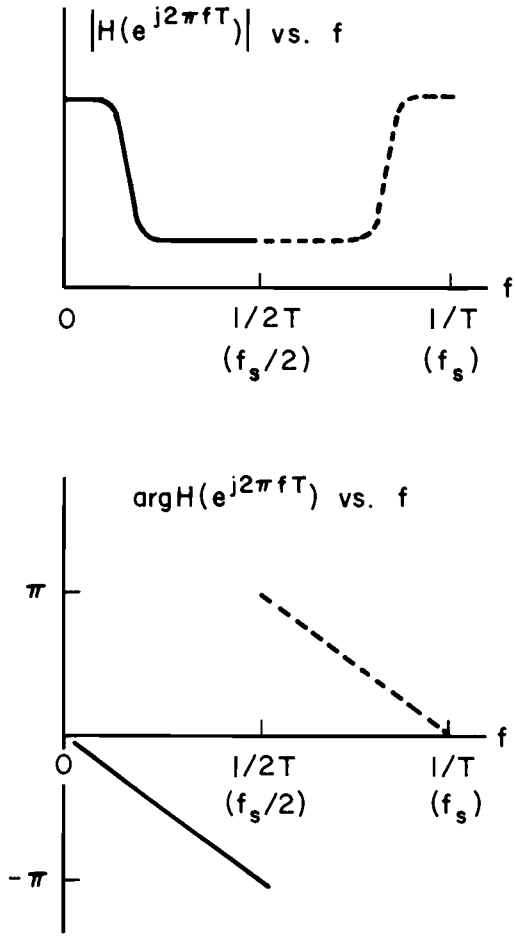


Figure 13. Magnitude and Phase of Complex Frequency Response

In this software pack, we evaluate the magnitude and phase of the complex frequency response for frequency values between f = 0 and f = $f_s/2$.

It should be clear that the complex frequency response $H(e^{j2\pi fT})$ is simply the transfer function H(z) evaluated at $z = e^{j2\pi fT}$. When we evaluate the complex frequency response for frequency values f between f = 0 and f = 1/2T, we are simply reading the transfer function H(z) on the upper part of the unit circle. This is illustrated in Figure 14.

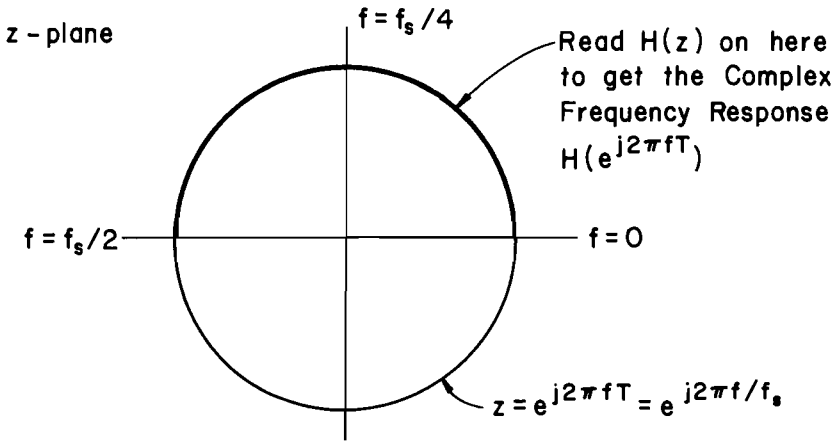


Figure 14. Role of the Unit Circle in the Calculation of Complex Frequency Response

For ARMA digital filters with M + 1 feedforward coefficients $b_0,b_1,\ldots,b_M$ and N feedback coefficients $a_1,a_2,\ldots,a_N$, the transfer function H(z) is

$$H(z) = \frac{b_0 + b_1z^{-1} + \ldots + b_Mz^{-M}}{1 + a_1z^{-1} + \ldots + a_Nz^{-N}}$$

and the complex frequency response is given by:

$$H(e^{j2\pi fT}) = \frac{b_0 + b_1e^{-j2\pi fT} + \ldots + b_Me^{-j2\pi fMT}}{1 + a_1e^{-j2\pi fT} + \ldots + a_Ne^{-j2\pi fNT}}$$

This is a ratio of polynomials in $e^{-j2\pi fT}$. We may brute force evaluate it at Q equally spaced frequency points between f = 0 and f = 1/2T by defining the discrete frequencies:

$$f_q = q/QT \quad , \quad q = 0,1,\ldots,Q/2$$

At these frequencies, the complex frequency response is

$$H_q = H(e^{j2\pi\frac{q}{QT}})$$

$$= \frac{b_0 + b_1e^{-j2\pi q/Q} + \ldots b_Me^{-j2\pi Mq/Q}}{1 + a_1e^{-j2\pi q/Q} + \ldots a_Ne^{-j2\pi Nq/Q}}$$

If we construct zero-padded sequences:

$$b = (b_0, b_1, \ldots, b_M, \underbrace{0, \ldots, 0}_{Q-1})$$
$$M+1$$

$$a = (1, a_1, \ldots, a_N, \underbrace{0, \ldots, 0}_{Q-1})$$
$$N+1$$

and let Q be $Q = 2^p$ (say 256) then the complex frequency response H_q may be evaluated as the ratio of discrete Fourier transforms (DFTs) of the zero-padded sequences a and b:

$$H_q = \frac{DFT(b)}{DFT(a)}$$

This is the way complex frequency response is evaluated in this software pack.

Poles and Zeros

As in analog filtering theory, poles and zeros play an important role in the analysis of digital filters. The concept of poles and zeros arises as follows. Factor the numerator and denominator polynomials of the transfer function H(z) to write:

$$H(z) = \frac{b_0(1 - z_1z^{-1})(1 - z_2z^{-1}) \ldots (1 - z_Mz^{-1})}{(1 - p_1z^{-1})(1 - p_2z^{-1}) \ldots (1 - p_Nz^{-1})}$$

$$b_M = z_1z_2 \ldots z_Mb_0(-1)^M$$

$$a_N = p_1p_2 \ldots p_N(-1)^N$$

The impulse response h_n may be written as a linear combination of complex modes of the form:

$$h_n = \sum_{i=1}^{N} A_ip_i^n, \quad n = 0,1, \ldots$$

$$= \sum_{i=1}^{N} A_i|p_i|^n e^{jn \arg p_i}$$

So the poles determine modal or natural responses, and the zeros (plus the poles) determine the scaling and phasing of modes. For stable filters, all poles are inside the unit circle:

$$|p_i| < 1, \; i = 1,2, \ldots, N$$

The magnitude $|p_i|$ of the pole determines how heavily damped the mode is and the angle, arg p_i, determines the frequency of oscillation. Poles near the unit circle correspond to lightly damped modes. Poles at large angles correspond to high-frequency modes. This is illustrated in Figure 15.

In this software pack, poles and zeros for any digital filter of numerator and denominator orders less than or equal to 16 may be calculated and plotted.

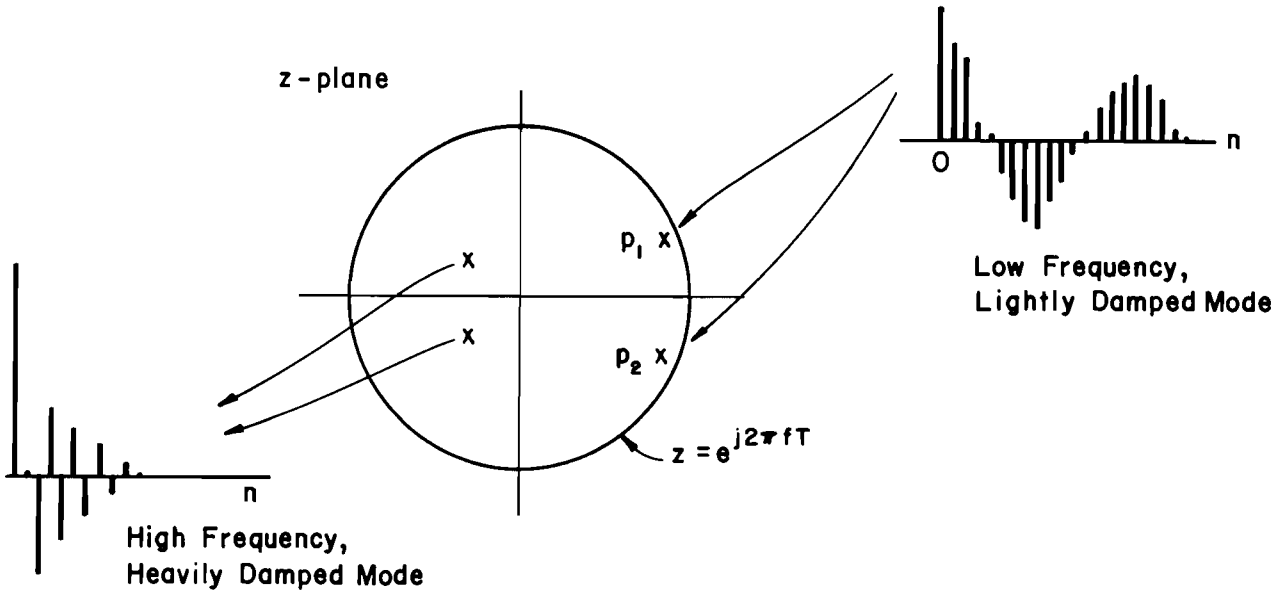


Figure 15. Poles and Zeros and Modal Responses

For purposes of visualizing complex frequency response, one may evaluate the factored transfer function at $z = e^{j2\pi fT}$ as follows:

$$H(e^{j2\pi fT}) = \frac{b_0(1 - z_1e^{-j2\pi fT}) \ldots (1 - z_Me^{-j2\pi fT})}{(1 - p_1e^{-j2\pi fT}) \ldots (1 - p_Ne^{-j2\pi fT})}$$

$$= \frac{b_0e^{-j2\pi MfT}(e^{j2\pi fT} - z_1) \ldots (e^{j2\pi fT} - z_M)}{e^{-j2\pi NfT}(e^{j2\pi fT} - p_1) \ldots (e^{j2\pi fT} - p_N)}$$

So, for a given value of f, the complex vectors $(e^{j2\pi fT} - z_i)$ and $(e^{j2\pi fT} - p_i)$ determine magnitudes and phases of the complex frequency response. When a pole p_i is near the unit circle, the length $(e^{j2\pi fT} - p_i)$ is small and the complex frequency response tends to be large in magnitude. When a zero z_i is near the unit circle, then $(e^{j2\pi fT} - z_i)$ is small and the complex frequency response tends to be small.

A Note on Sampling: Aliasing

Let H(s) denote an analog system with impulse response h(t) and complex frequency response $H(j2\pi f)$. Approximate this system with a digital system whose unit pulse response is a sampled version of h(t):

$$h_k = h(t = kT)$$

The complex frequency response of the digital filter is then:

$$H(e^{j2\pi fT}) = 1/T \sum_{k=-\infty}^{+\infty} H(j2\pi f + j2\pi k/T)$$

This equation says that the frequency response for the digital system is a folded response composed of an infinite sum of shifted versions of the frequency response for the analog system. If the analog frequency response is zero for $|f| > 1/2T$, the digital signal will have a frequency response which is an undistorted version of the analog frequency response for $|f| < 1/2T$. If the analog frequency response is not zero for $|f| > 1/2T$, then distortion (aliasing) occurs. This is illustrated in Figure 16. Since no analog system can be truly bandlimited, some aliasing always occurs. The sampling frequency, $f_s = 1/T$(Hz), should be chosen so that it is at least twice, and preferably 5 to 10 times as high as the highest significant frequency in the frequency response for the analog system.

This completes our discussion of digital filter and discrete-time system theory.

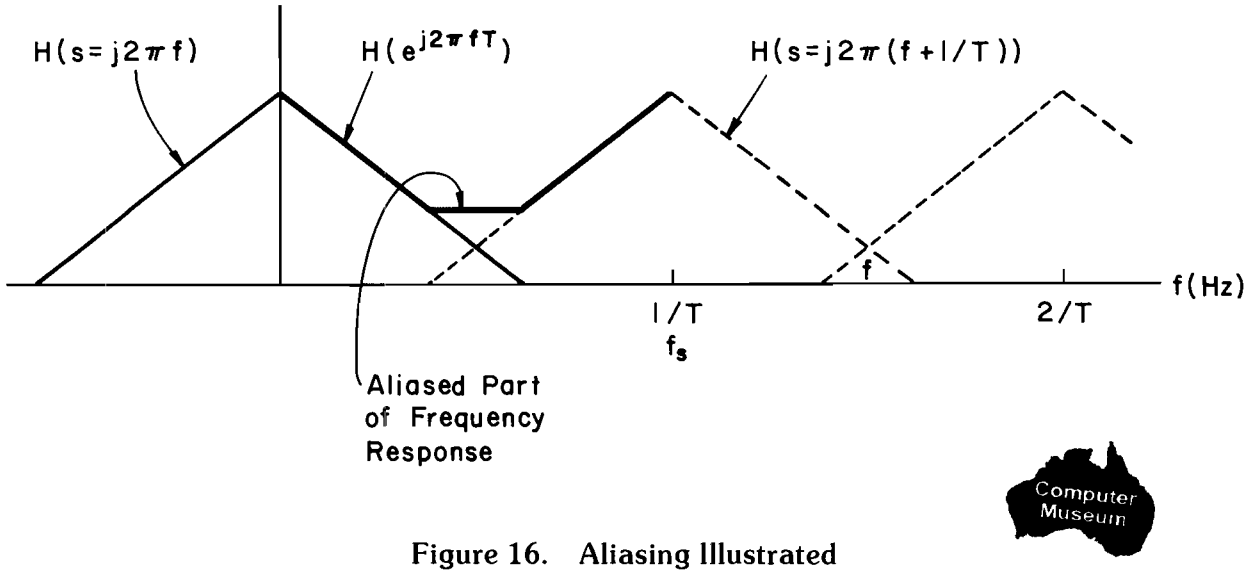


Figure 16. Aliasing Illustrated

Chapter 3
General Instructions and Considerations

In this chapter, we record general rules of the game for selecting menu options, for answering prompts, and for effectively using the data file and graphics features of the software. Read this material carefully before proceeding to the design and analysis instructions of Chapter 4.

Getting Started

To get started, you insert the applications disc #1 into the disc drive, load the program file "AUTOST" into the computer, and run it:

```
LOAD "AUTOST",1 (EXECUTE)
```

From here on, required files will be loaded into the computer automatically when they are needed. You will simply answer prompts to move from point to point in the software.

Note

Immediately upon executing "AUTOST", a prompt will appear asking for the length of the timeout. This is the amount of time the machine will wait for an external device to respond before abandoning the attempt. (If a timeout occurs, the program will automatically recover itself.)

Answering Prompts and Prods

If a Question Mark Appears

This is a prompt. Type in your answer and press: (CONTINUE).

If the answer to the prompt is improper, the question will be repeated. Many times a parenthesized list of candidate responses will appear. Examples are:

```
(Y OR N)
(1), (2)
(L), (H), (BP), (BS)
```

In this case, you must respond with an answer from the list.

If the Words "SELECT KEY" Appear

This is a prod to select one of the ten special function keys.

Data Files

Whenever you reach a point in a program where a data file may be read from or written into, you will receive a reminder on the CRT to put a disc into the disc drive. You will then be reminded when to reinsert the program disc. We suggest you leave the write protect on at all times on your program disc and use a separate disc for storing, reading and writing data files.

When the computer needs to read from, or write into a data file, it asks you for the name of the file. After you enter the name, you are likely to generate one of several error conditions. The computer will ignore your response and ask again for the name of the data file. Here is a list of common errors to watch for:

- If you are writing into a data file, the disc may not have enough storage space to accommodate the new file (the software always tries to create a new data file).
- If you are writing, the write protect may be on, thereby preventing you from writing.
- There may already exist a data file by the same name as the one you are trying to write into.
- The computer may be trying to read more values from a data record than are stored in your data file.

There are four places in the software where data may be read from a data file. The following table shows how many numbers the computer expects to find in the data file. The data files should be zero-padded to the required length.

Expected Data File Sizes

Description of Data	Number of Data Values Expected	Format of Data
Coefficients of a digital filter (MA, AR, or ARMA)	2 Records 1st record – Numerator order (N) 2nd record – Denominator order (M)	2 Records 1st record – $b_0, b_1, \ldots b_N$ 2nd record – $a_0, a_1, \ldots, a_M$
Frequency response samples	1 Record 258 values	mag. of point 0, phase (rad) of point 0, ⋮ mag. of point N, phase of point N, 0, 0, ⋮ 0 (258 total)
Covariance sequences	1 Record 258 values	$r_0, r_1, \ldots r_{N-1}, 0, \ldots, 0$ (258 total)

The External Plotter (Optional)

After you indicate you want to generate a plot, you will be prompted as follows:

```
PLOTTER IS X,Y$ (DEFAULT = 3,"INTERNAL")?
```

If you wish to plot to the CRT, press CONTINUE for DEFAULT. You may then press SHIFT DUMP GRAPHICS to print to a thermal printer. If you wish to plot to an external plotter, give the select code (X) and string identifier (y$). (For example: 705, "HPGL".)

Note

Never insert an interface into the computer while it is executing instructions. Wait until the computer has halted execution and is waiting for you to press CONTINUE.

CRT Scrolling for Table Viewing

After you have generated a table on the CRT, it may disappear before you have had a chance to look at it. Don't worry. The table is still on the CRT. You just have to scroll the CRT down to see it. Remember though, only 57 lines are remembered by the computer, so some long tables may not fit completely on the CRT.

Chapter 4
Design and Analysis Instructions

In this chapter we outline the design and analysis techniques available in this software and give detailed instructions for using them. Examples follow in Chapter 5.

Outline of Design Techniques

The regular reader of the IEEE Transactions on Acoustics, Speech, and Signal Processing [1], and well-known texts and publications such as [2] and [3] is aware of legion numbers of design techniques for digital filters. We have selected for inclusion in this software, several techniques found most useful in research, development, and manufacturing activities.

For the design of FIR (or MA) filters, we have selected five techniques:

1. automated design of FIRs from idealized frequency response specifications for lowpass, bandpass, highpass, or bandstop filters [4],
2. frequency sampling design from arbitrary frequency response specifications [2],
3. design of FIR filters from truncated impulse responses,
4. least squares design of lowpass, bandpass, highpass, and bandstop filters [6], and
5. minimax design of FIR filters [8].

For technique 1, order selection is automatic. For techniques 2 thru 5, the user iteratively selects orders that meet his or her complexity and performance requirements.

For the design of IIR (or ARMA) filters, we have selected techniques for:

6. transformations from all-pole or pole-zero analog filters to ARMA digital filters by the methods of impulse-invariance [2], covariance-invariance [5], and bilinear-z [2],
7. automated design of IIRs from idealized frequency response specifications for lowpass, bandpass, highpass, and bandstop filters [4], and
8. least squares design of IIR filters [9].

In technique 6, the supposition is that the user is armed with an analog filter transfer function for which he wants a digital equivalent. In technique 7, filter order selection and all other aspects of the design are automatic. For technique 8, order is user selectable.

Software Overview

This software may be exercised in three modes to design and analyze digital filters, discrete-time systems, and discrete-time signals. Refer to Figure 17. Beginning at (i), you may:

(i) enter frequency domain digital filter specifications, design a digital filter which meets these specifications, and analyze the impulse and frequency response of the resulting design.

Beginning at (ii), you may:

(ii) enter the transfer function of an analog filter, transform it to an impulse-invariant, covariance-invariant, or bilinear-z digital filter equivalent, and analyze its impulse and frequency response.

Finally, exercising only the analysis features of the software you may begin at (iii) to:

(iii) enter a digital filter or discrete-time transfer function and analyze the impulse and frequency response. You may enter a discrete-time sequence as coefficients of an FIR filter to trick the software into thinking it is discrete Fourier transforming (DFTing) your filter. Entry in this mode may be from keyboard or data file.

Results of analysis may be tabulated or plotted on CRT graphics, thermal printer, or pen plotter.

Figure 18 is a road map that shows how to get from menu to menu in the software. All routes pass through the MAIN menu. So to get from OUTPUT to DESIGN or DESIGN to OUTPUT, you first return to MAIN by a heavy lined route and then go to your desired destination by pressing a special function key.

Within menus, you select special function keys to exercise options.

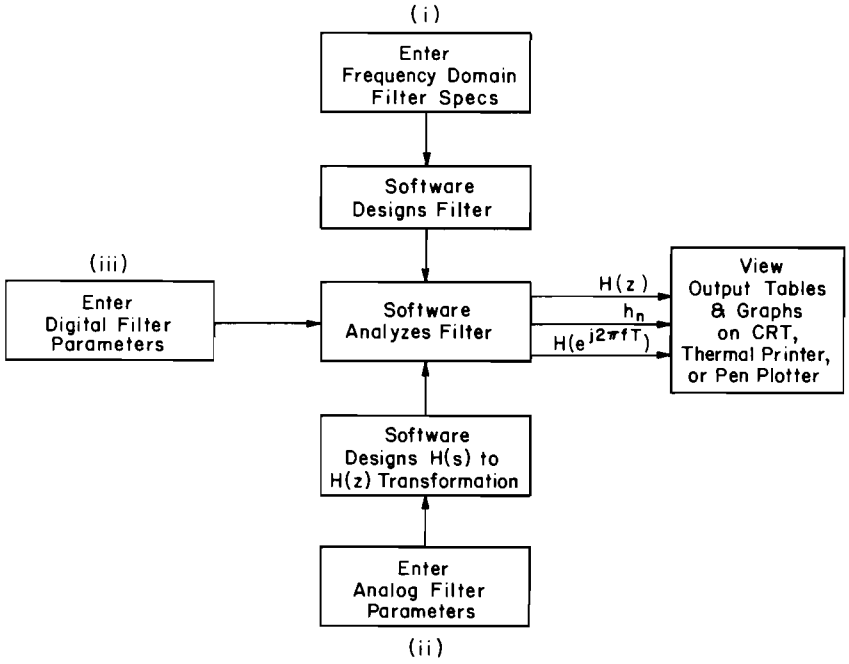


Figure 17. Design and Analysis Modes

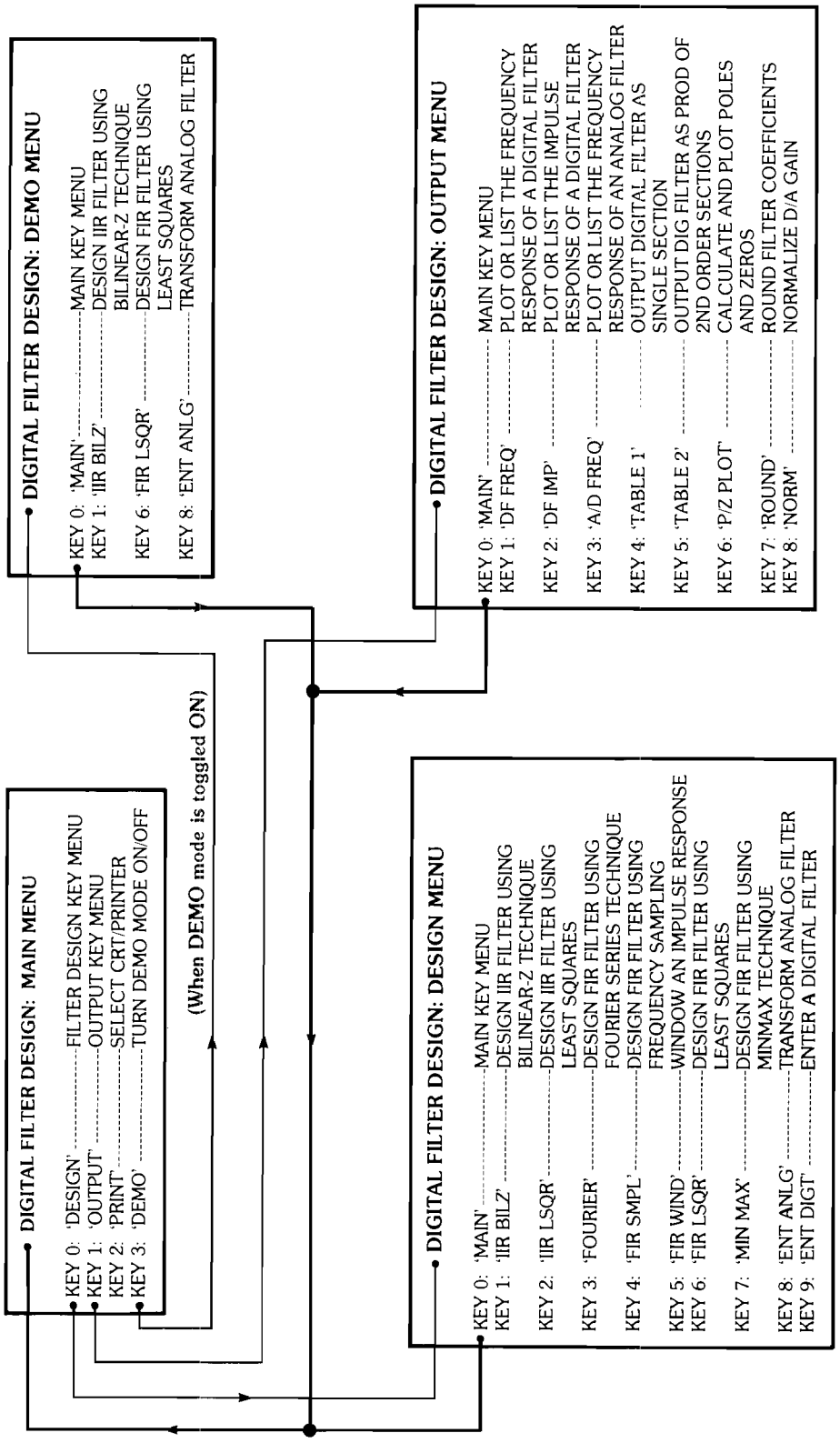


Figure 18. Software Road Map

MAIN MENU

After loading "AUTOST" according to the instructions of Chapter 3, you see the MAIN MENU on the CRT. Here is a copy of it:

DIGITAL FILTER DESIGN: MAIN MENU

KEY 0: 'DESIGN' ----------FILTER DESIGN KEY MENU
KEY 1: 'OUTPUT' ----------OUTPUT KEY MENU
KEY 2: 'PRINT' ----------SELECT CRT PRINTER
KEY 3: 'DEMO' ----------TURN DEMO MODE ON/OFF

SELECT KEY

DESIGN	OUTPUT	PRINT	DEMO	

Note the SELECT KEY. Press [k0], DESIGN, to get to the DESIGN MENU to design digital filters. Press [k1], OUTPUT, to get to the OUTPUT MENU for analyzing digital filters. Press [k2], PRINT, to select the print device and [k3], DEMO, to run the demonstration program. These keys are self-explanatory.

DESIGN MENU

If you are in the MAIN MENU and you press [k0], DESIGN, you see the DESIGN MENU:

DIGITAL FILTER DESIGN: DESIGN MENU

KEY 0: 'MAIN' ----------MAIN KEY MENU
KEY 1: 'IIR BILZ' ----------DESIGN IIR FILTER USING BILINEAR-Z TECHNIQUE
KEY 2: 'IIR LSQR' ----------DESIGN IIR FILTER USING LEAST SQUARES
KEY 3: 'FOURIER' ----------DESIGN FIR FILTER USING FOURIER SERIES TECHNIQUE
KEY 4: 'FIR SMPL' ----------DESIGN FIR FILTER USING FREQUENCY SAMPLING
KEY 5: 'FIR WIND' ----------WINDOW AN IMPULSE RESPONSE
KEY 6: 'FIR LSQR' ----------DESIGN FIR FILTER USING LEAST SQUARES
KEY 7: 'MIN MAX' ----------DESIGN FIR FILTER USING MINMAX TECHNIQUE
KEY 8: 'ENT ANLG' ----------TRANSFORM ANALOG FILTER
KEY 9: 'ENT DIGT' ----------ENTER A DIGITAL FILTER

SELECT KEY

MAIN	IIR BILZ	IIR LSQR	FOURIER	FIR SMPL
FIR WIND	FIR LSQR	MIN MAX	ENT ANLG	ENT DIGT

Note the SELECT KEY. You may return to MAIN by pressing [k0], MAIN, or you may design a digital filter according to a particular option by selecting [k1] through [k9]. The following paragraphs give a key-by-key description of data entry for each design option.

k1: IIR BILZ (Design IIR Filter Using Bilinear-z Technique)

This design procedure starts with your magnitude response specifications. These determine the digital filter order and certain transformation parameters. A normalized, lowpass, Butterworth or Chebyshev analog filter is then designed. Two transformations are made simultaneously on this analog design. The first one produces a prewarped analog filter of the proper type (lowpass, highpass, etc.). The second transformation (the bilinear-z transformation) replaces the analog transfer function with a digital transfer function whose frequency response meets your magnitude response specifications. The design flow is shown below.

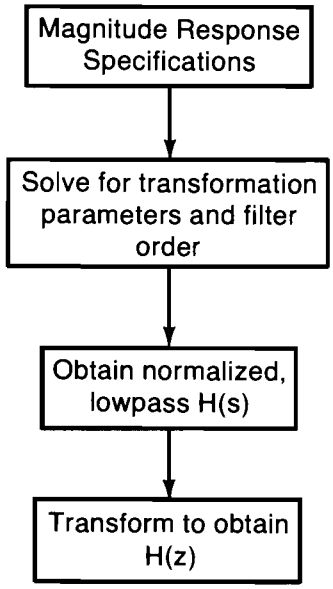


Figure 19

The program carries out these steps automatically in response to your specifications. For more information about this design, see [4].

Inputs

The inputs for this design are illustrated below.

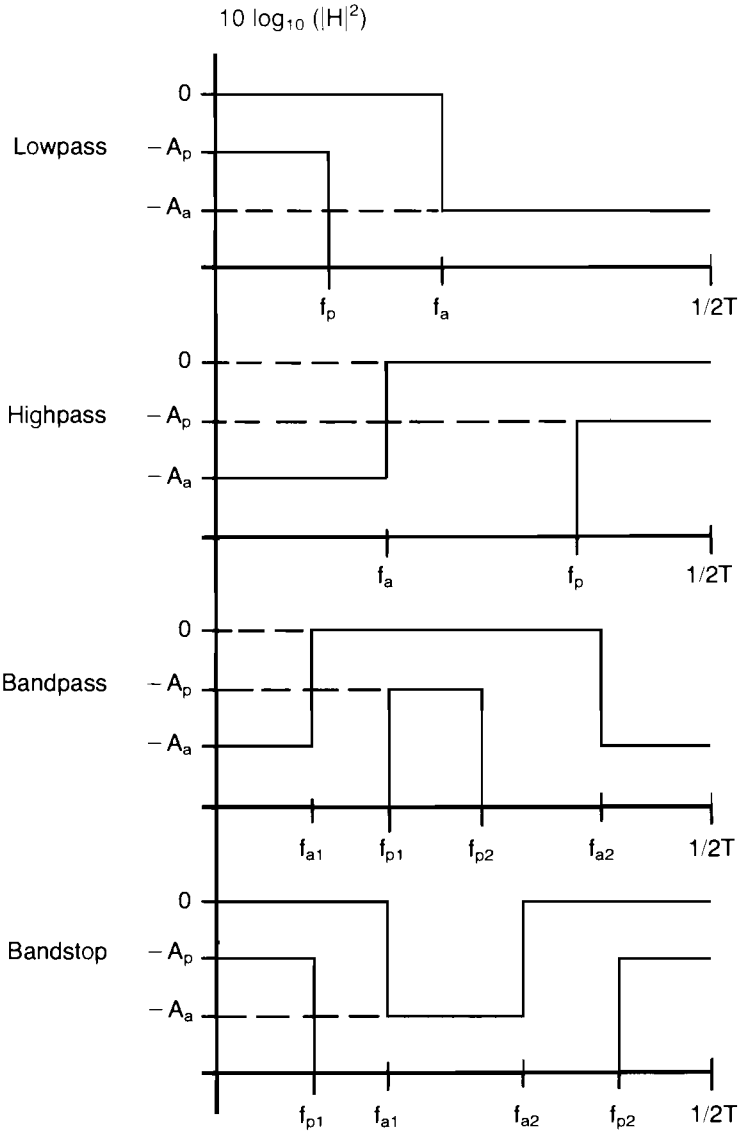


Figure 20

You are prompted by the program to enter these parameters. The program accepts input parameters in the following ranges:

T = sampling period; $T>0$
A_p = maximum decibel ripple in passband; $0<A_p<A_a$
A_a = minimum decibel rejection in stopband; $A_p<A_a\leq 100$

Lowpass: $0<f_p<f_a<1/2T$
Highpass: $0<f_a<f_p<1/2T$
Bandpass: $0<f_{a1}<f_{p1}<f_{p2}<f_{a2}<1/2T$
Bandstop: $0<f_{p1}<f_{a1}<f_{a2}<f_{p2}<1/2T$

Remarks

1. The program chooses the lowest filter order which meets the desired specifications.
2. The maximum decibel loss in the passband will be A_p.
3. The magnitude of the decibel loss in the stopband will usually be greater than A_a. In other words, the frequency response will be at least as good in the stopband as what you require.
4. The numerator and denominator orders of the resulting filter will be equal.
5. If the required filter order is greater than 40, an error message is given telling you that the magnitude specifications must be relaxed.
6. Given specifications lead to lower order Chebyshevs than Butterworths.

k2: IIR LSQR (Design IIR Filter Using Least Squares)

The least-squares design technique is used to obtain ARMA (N,M) filters of specified numerator and denominator order. You begin with a covariance sequence r_k, which is used to design an AR filter of order $N - 1$ whose covariance sequence is the same as the original covariance sequence for the first N values. The impulse response of the AR $(N- 1)$ filter, denoted h_k, is generated next. h_k and r_k are used to obtain the coefficients of an ARMA (N,M) filter whose frequency response approximates the frequency response of the AR $(N - 1)$ filter and whose numerator order M and denominator order N you specify.

The method of modified least-squares is used to obtain the ARMA (N,M) filter, This algorithm minimizes the modified squared error between the frequency response of the AR filter, $H(e^{j2\pi fT})$, and the frequency response of the ARMA filter, $\hat{H}(e^{j2\pi fT})$. This error, denoted ϵ^2, is expressed as:

$$\epsilon^2 = T\int_{-1/2T}^{1/2T} |\hat{D}(e^{j2\pi fT})|\,|H(e^{j2\pi fT}) - \hat{H}(e^{j2\pi fT})|^2\,df$$

$\hat{D}(z)$ is the denominator polynomial of $\hat{H}(z)$.

The covariance sequence r_k is obtained by specifying an idealized frequency response like the one shown below.

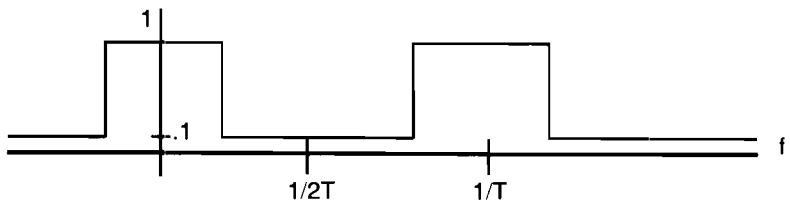


Figure 21

The first N Fourier series coefficients of the ideal frequency response are used as the first N values of r_k. The resulting AR $(N-1)$ filter will have a frequency response which approximates the original ideal frequency response in a least-squares sense.

The design flow is shown below:

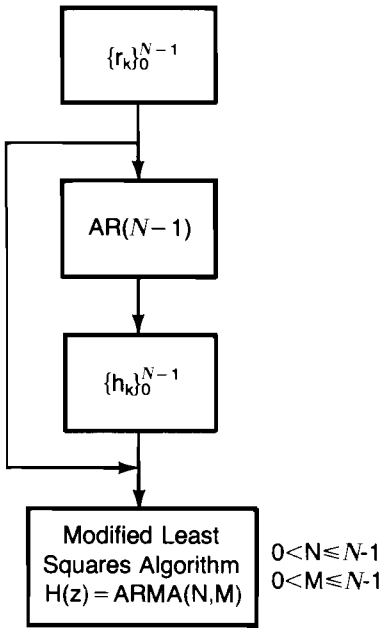
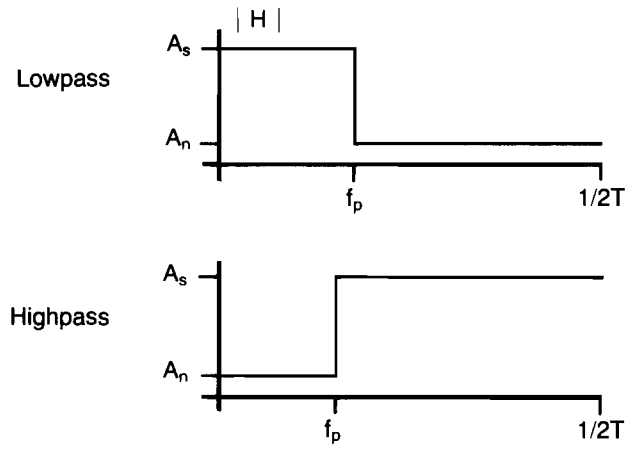


Figure 22

The program carries out these steps automatically in response to your specifications. For more information on this design, see [9].

Inputs

You enter the specifications for an ideal frequency response by answering prompts. These specifications are shown below:



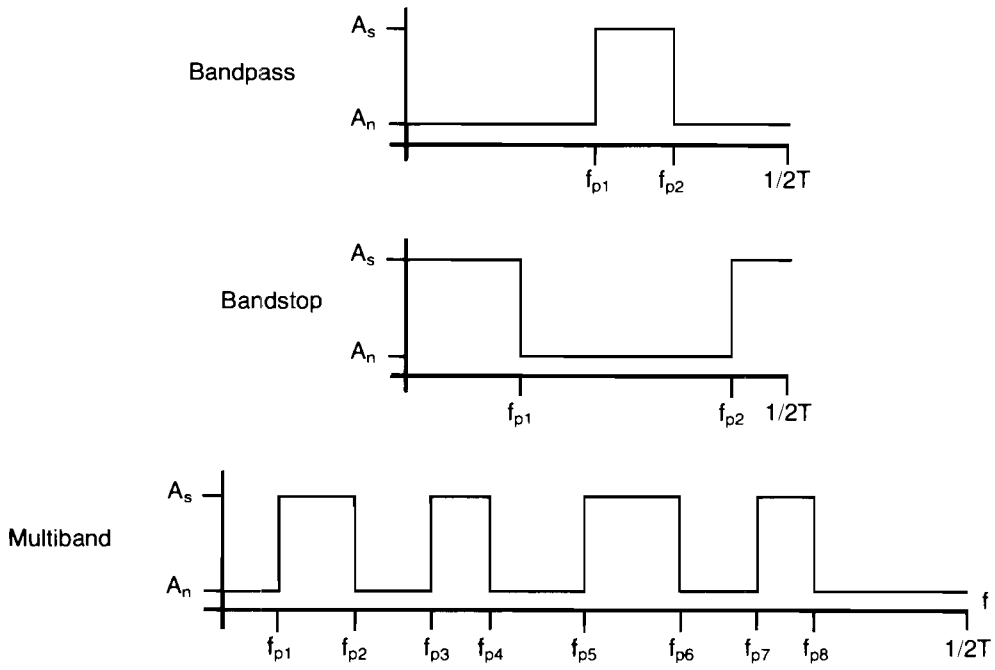


Figure 23

The program accepts input parameters in the following ranges:

T = sampling period; $T>0$
A_s = signal height; $A_s>0$
A_n = noise height; $A_n>0$

Lowpass: $0<f_p<1/2T$
Highpass: $0<f_p<1/2T$
Bandpass: $0<f_{p1}<f_{p2}<1/2T$
Bandstop: $0<f_{p1}<f_{p2}<1/2T$

Multiband (4 passbands): $0<f_{p1}<f_{p2}<f_{p3}<f_{p4}<f_{p5}<f_{p6}<f_{p7}<f_{p8}<1/2T$
Multiband (3 passbands): $0<f_{p1}<f_{p2}<f_{p3}<f_{p4}<f_{p7}<f_{p8}<1/2T$
Multiband (2 passbands): $0<f_{p1}<f_{p2}<f_{p7}<f_{p8}<1/2T$

The AR ($N-1$) and ARMA (N,M) filter orders must be chosen so that:

$\text{Max}(2,N,M)\leq N-1\leq 255$

$0<N\leq 40$
$0\leq M\leq 40$

You may also enter your own covariance sequence. The length of the sequence, N, sets the AR order at $N-1$. This covariance sequence may be entered manually or from a data file. (Store r_k in an OPTION BASE 0 array of size 258 like this: $r_k = R(k)$, $k = 0,1, \ldots N-1$; $R(k) = 0$, $k = N, N+1, \ldots, 257$. These numbers should be full-precision.)

You can obtain as output either the ARMA (N,M) coefficients or the AR ($N-1$) coefficients, but not both.

Remarks

1. You may want to design an AR (p) filter using this technique. Simply enter a covariance sequence of length N. Then specify the numerator order of the ARMA (N,M) as zero, and the denominator order as p. This produces an ARMA (p,0) design for an AR (p) filter.
2. The order of the AR ($N - 1$) filter should be chosen as high as possible to obtain a good approximation. However, the higher the order, the longer the design takes.
3. M may be larger than N in the ARMA (N,M). This provides flexibility not available in the other IIR design methods.
4. The frequency response of the ARMA (N,M) or of the AR ($N - 1$) is only an approximation to ideal specifications. The only specification which is met exactly is the filter order.

k3: FOURIER (Design FIR Filter Using Fourier Series Technique)

This design procedure starts with your magnitude response specifications which are then used to determine the filter order and to design a Kaiser window. The Kaiser window is multiplied with the Fourier Series coefficients of an idealized magnitude response to produce the coefficients of an FIR filter that meets the original magnitude response specifications. The design flow is shown below.

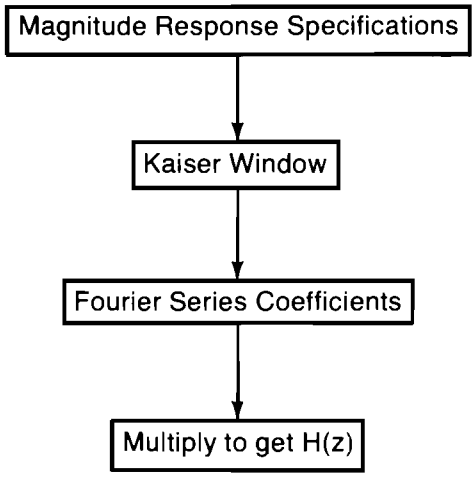


Figure 24

The program carries out these steps automatically in response to your specifications. For more information about this design procedure, see [4].

Inputs

The inputs for this design are shown below.

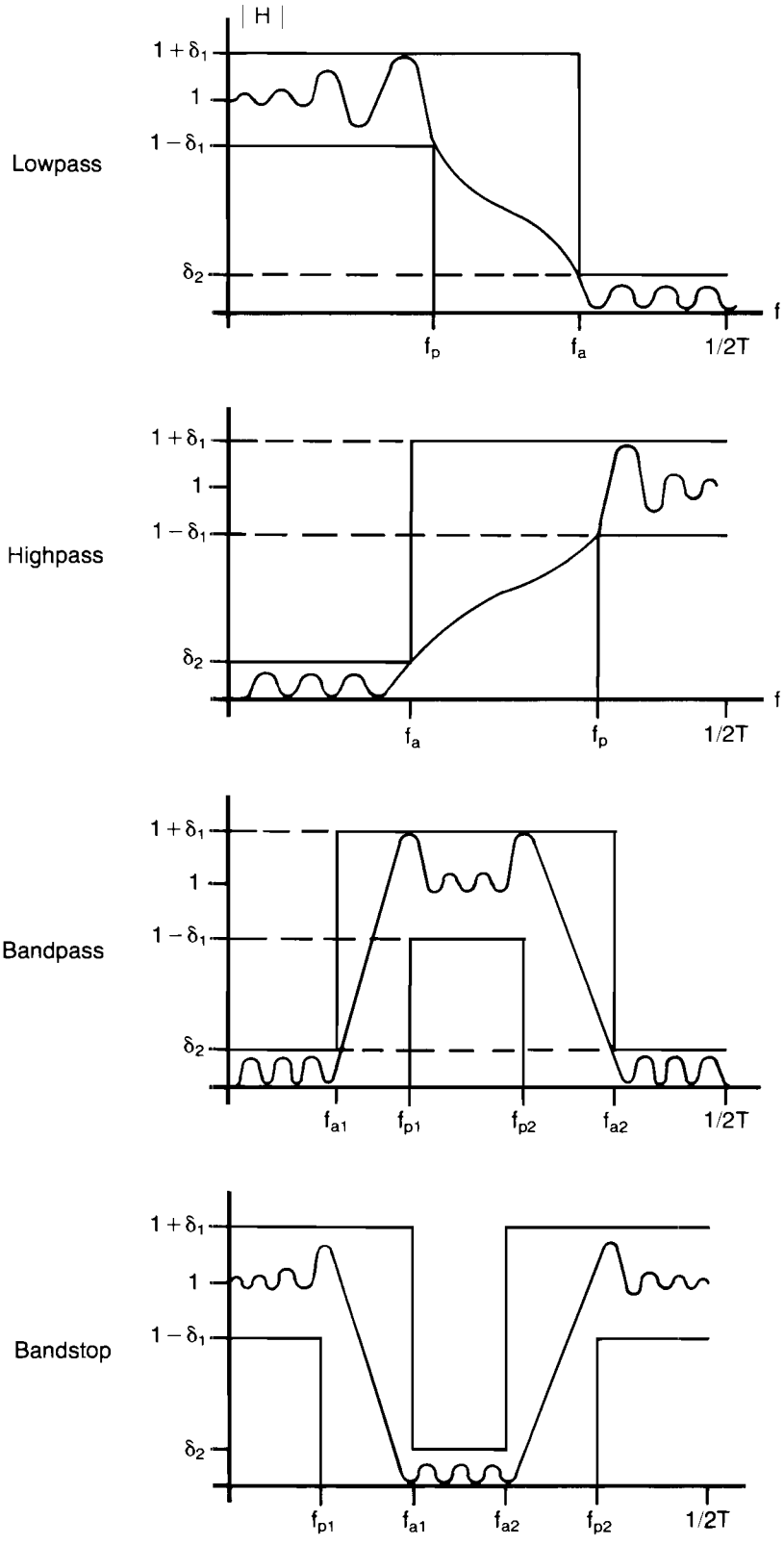


Figure 25

The program accepts input parameters in the following ranges:

T = sampling period; $T>0$

$A_p = 20\log_{10}\left(\frac{1+\delta_1}{1-\delta_1}\right)$; $0<A_p<A_a$

$A_a = -20\log_{10}\delta_2$; $A_p<A_a\leq 100$

Lowpass: $0<f_p<f_a<1/2T$
Highpass: $0<f_a<f_p<1/2T$
Bandpass: $0<f_{a1}<f_{p1}<f_{p2}<f_{a2}<1/2T$
Bandstop: $0<f_{p1}<f_{a1}<f_{a2}<f_{p2}<1/2T$

Remarks

1. The program chooses the lowest filter order which meets the desired specifications.
2. Either A_p or A_a will be improved. That is, either the actual ripple in the passband will be less than A_p or the actual rejection in the stopband will be better than A_a.
3. If the required filter order is greater than 255, an error message notifies you that your specifications must be relaxed.
4. The filters will have a linear phase response.

k4: FIR SMPL (Design FIR Filter Using Frequency Sampling)

In this design mode, you design FIR filters from samples of a desired frequency response. These samples may arise as experimental data. The frequency response of the filter designed passes exactly through the samples of the desired frequency response.

Inputs

Illustrated in Figure 26 are typical magnitude and phase frequency responses for some desired complex frequency response. The following samples of these responses are entered for an FIR filter of order N (with N + 1 filter coefficients b_i):

$$\begin{matrix}|H(e^{j2\pi f_n T})| \\ \arg H(e^{j2\pi f_n T})\end{matrix} : f_n = \begin{cases} n/(N+1)T\,,\ n=0,1,\ldots,(N+1)/2 & (N+1 \text{ even}) \\ n/(N+1)T\,,\ n=0,1,\ldots,N/2 & (N+1 \text{ odd}) \end{cases}$$

These points can be entered manually or from a data file.

You must select N. The following parameter constraints are in force:

(i) T = sampling interval ; $T > 0$

(ii) N = filter order ; $1 \leq N \leq 127$

($N+1$ = Number of FIR filter coefficients ; $2 \leq N+1 \leq 128$)

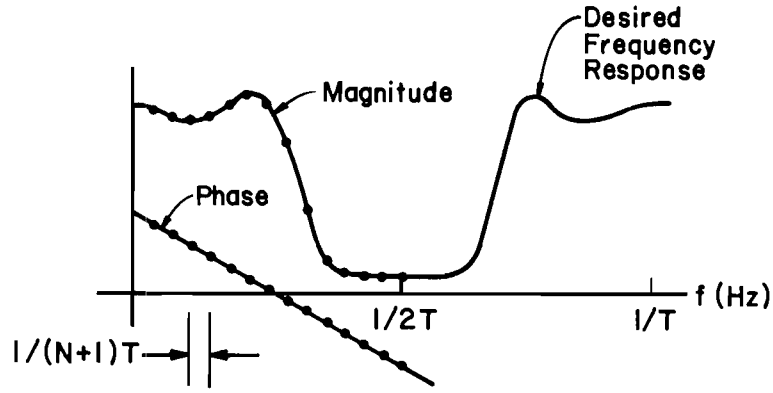


Figure 26. Frequency Samples for the Frequency Sampling Designs

Remarks

1. In this design mode you enter only half the samples ((N + 1)/2 or N/2) of the frequency response between f = 0 and f = 1/T. This is because the second half of the sampled complex frequency response must contain samples that are complex conjugates of those from the first half if real FIR filter coefficients are to be obtained. If you want to enter only magnitudes $|H(e^{j2\pi f_n T})|$, you tell the program to add a phase function that will cause the resulting FIR filter to have linear phase.
2. To achieve a linear phase for N odd (N + 1 even), you must have the magnitude response equal to zero at the foldover frequency:

$$|H(e^{j2\pi f_n T})| = 0, \quad n = (N+1)/2$$
$$f_{(N+1)/2} = \frac{(N+1)/2}{(N+1)T} = 1/2T$$

For real filters $\arg H(e^{j2\pi f_n T}) = 0$ for n = 0.

[k5]: FIR WIND (Window an Impulse Response)

This option allows you to window your impulse response data with a rectangular window. The windowed points are used as the coefficients of an FIR filter. This procedure enables you to obtain an FIR approximation of an FIR or IIR filter. For more information about windowing, consult a digital filter design text.

Inputs

The inputs for this design option are the first and last points of the window.

Remarks

None.

[k6]: FIR LSQR (Design FIR Filter Using Least Squares)

The least squares design starts with the specification of an ideal frequency response, denoted $H_d(e^{j2\pi fT})$. A real-valued frequency domain weighting function is selected and denoted R(f). The frequency response of the FIR filter, $H(e^{j2\pi fT})$, minimizes the following weighted squared error:

$$\epsilon^2 = T\int_{-1/2T}^{1/2T} R(f)\,|H(e^{j2\pi fT}) - H_d(e^{j2\pi fT})|^2\,df$$

Note that the use of a weighting function penalizes you more for errors in one part of the frequency band than in others. You can improve the response over selected frequency bands where you want little error at the expense of a poorer response over bands where you can tolerate large errors.

Let $\{r_k\}_0^N$ be the first N+1 Fourier series coefficients for the weighting function R(f) and let $\{p_k\}_0^N$ be the first N+1 Fourier series coefficients for the product of $H_d(e^{j2\pi fT})$ and R(f). The filter coefficients, $\{a_k\}$ are found using the following system of equations:

$$\begin{bmatrix} r_0 & r_1 & \cdots & r_N \\ r_1 & r_0 & r_1 & \vdots \\ \vdots & & \ddots & r_1 \\ r_N & \cdots & & r_0 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} p_0 \\ p_1 \\ \vdots \\ p_N \end{bmatrix}$$

The FIR filter is formed as:

$$H(z) = \sum_{i=0}^{N} a_i z^{-1}$$

The design flow is shown below.

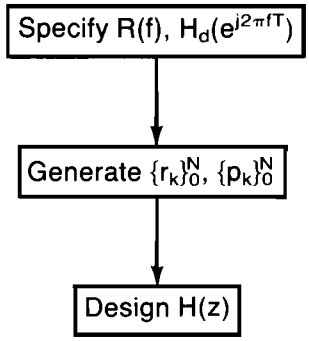


Figure 27

The program carries out these steps automatically in response to your specifications. For more information about this design procedure, see [6] and [7].

Inputs

The inputs for this design option are shown below.

Key { ——— = $H_d(e^{j2\pi fT})$: desired frequency response
- - - = R(f): weighting function (completely specified by A_s and A_n.)

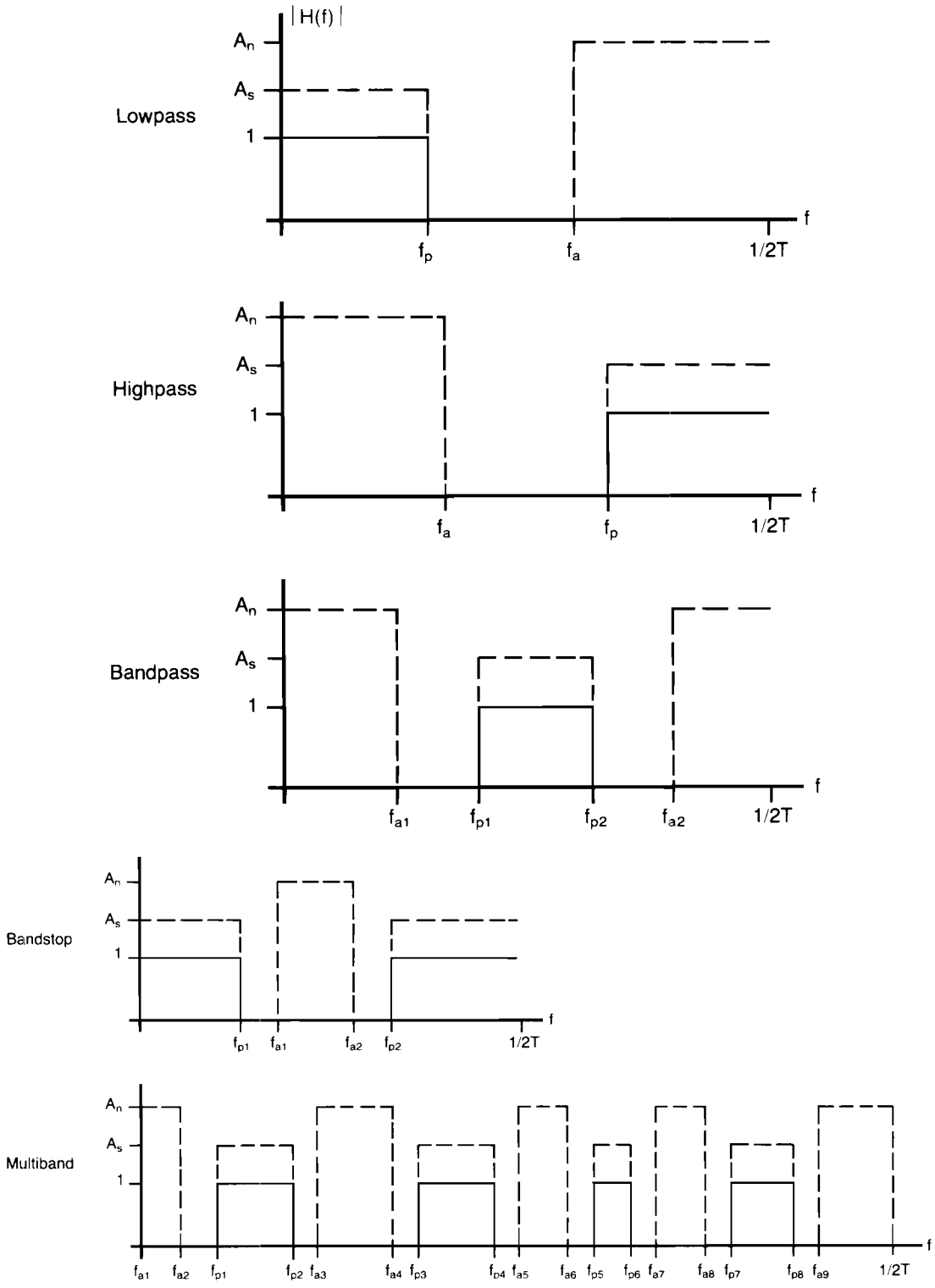


Figure 28

The program accepts input parameters in the following ranges:

T = sampling period: $T>0$
N = filter order $2 \leq N \leq 255$
A_n = stopband weighting; $A_n>0$
A_s = passband weighting; $A_s>0$

Lowpass: $0<f_p<f_a>1/2T$
Highpass: $0<f_a<f_p<1/2T$
Bandpass: $0<f_{a1}<f_{p1}<f_{p2}<f_{a2}<1/2T$
Bandstop: $0<f_{p1}<f_{a1}<f_{a2}<f_{p2}<1/2T$

Multiband (4 passbands):
$0<f_{a2}<f_{p1}<f_{p2}<f_{a3}<f_{a4}<f_{p3}<f_{p4}<f_{a5}<f_{a6}<f_{p5}<f_{p6}<f_{a7}<f_{a8}<f_{p7}<f_{p8}<f_{a9}<1/2T$

Multiband (3 passbands):
$0<f_{a2}<f_{p1}<f_{p2}<f_{a3}<f_{a4}<f_{p3}<f_{p4}<f_{a5}<f_{a6}<f_{p7}<f_{p8}<f_{a9}<1/2T$

Multiband (2 passbands):
$0<f_{a2}<f_{p1}<f_{p2}<f_{a3}<f_{a4}<f_{p7}<f_{p8}<f_{a9}<1/2T$

If the first passband of a multiband filter is a lowpass band (that is, $f_{p1} = 0$), then $f_{a1} = f_{a2} = f_{p1} = 0$.

If the last passband of a multiband filter is a highpass band (that is, $f_{p8} = 1/2T$), then $f_{p8} = f_{a9} = 1/2T$.

Remarks

1. If the filter order is odd, the linear phase constraint imposed by the program requires that $H_2 = 0$ at $f = 1/2T$. Therefore, for specifications that violate this constraint (bandstop, highpass, etc.) and an odd filter order, there will be an unwanted "notch" in the magnitude response at $f = 1/2T$.
2. The filters will have a linear phase.

k7: MIN MAX (Design FIR Filter Using Minmax Technique)

This design option minimizes a weighted Chebyshev error to approximate an ideal frequency response. You can design three types of filters: passband/stopband, Hilbert transformers and differentiators. For more information, see [8].

The inputs for designing lowpass, highpass, bandpass and multiband filters are the same as those given for the least squares designs: N, A_s, A_n, T, f_a, f_p, etc. However, for this design $2 \leq N \leq 127$. For more unusual filters, the following discussion is relevant.

A Hilbert transformer is a filter with the following frequency response:

$$H(e^{j2\pi fT}) = \begin{cases} -j; & 0 \leq f \leq 1/2T \\ j; & 1/2T < f \leq 1/T \end{cases}$$

Inputs

The inputs for this option are shown below:

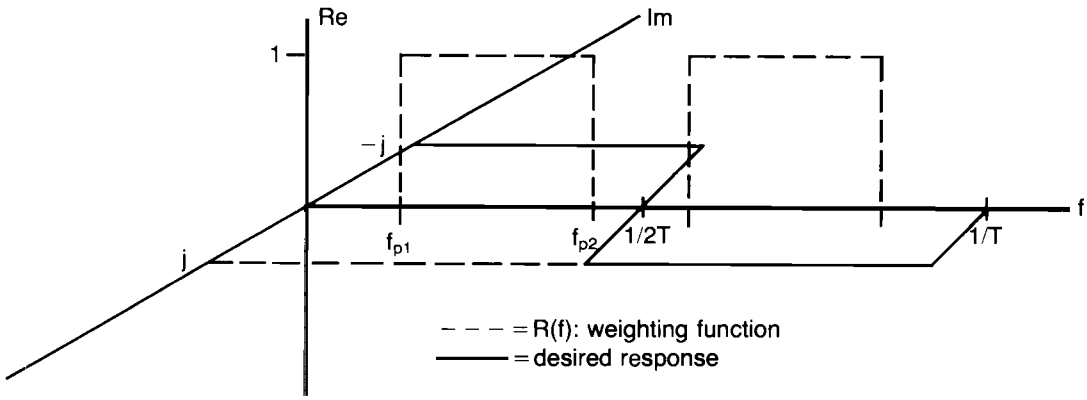


Figure 29

The program accepts $T>0$ and $0<f_{p1}<f_{p2}<1/2T$, and $N\leq 127$.

A differentiator is a filter with the following frequency response:

$$H(e^{j2\pi fT}) = (\text{Slope}) * f;\ 0\leq f\leq 1/2T$$

The inputs for this option are shown below:

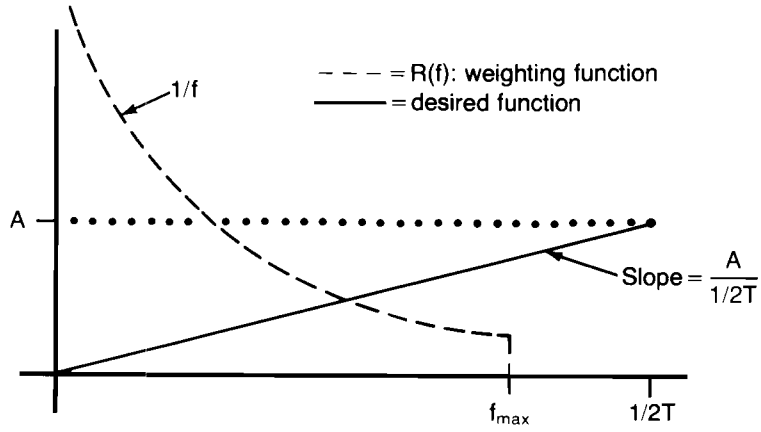


Figure 30

The software accepts $T>0$ and $0<f_p\leq 1/2T$.

Remarks

1. The numerical algorithm used to design Minmax FIR filters may not converge for very large values of N. Nevertheless, you may want to look at frequency response characteristics to see if the design is acceptable.
2. For Minmax FIR designs, $2\leq N\leq 127$.

[k8]: ENT ANLG (Transform Analog Filter)

Many times you have an analog transfer function H(s) and want to obtain a digital filter transfer function H(z) that approximates H(s) in some way. Three approximation techniques are available in this pack: bilinear-z, impulse invariance, and covariance invariance.

Bilinear-z and Prewarping

The bilinear-z transformation models the analog integrator, 1/s, with a digital approximation to this integrator. The transformation is:

$$\frac{1}{s} \rightarrow \frac{T}{2}\frac{1 + z^{-1}}{1 - z^{-1}}$$

The effect of this transformation on the frequency response is to map the function $H(j2\pi f)$ on $0 \leq f < \infty$ into $H(e^{j2\pi\zeta T})$, $0 \leq \zeta \leq 1/2T$. The mapping is nonlinear and produces an effect called frequency warping. Warping refers to the following phenomenon: if $H(j2\pi f)$ has a passband on $f_1 \leq f \leq f_2$, then $H(e^{j2\pi\zeta T})$ will exhibit this passband on:

$$\zeta_1 = (1/\pi T)\tan^{-1}(\pi f_1 T) \leq \zeta \leq (1/\pi T)\tan^{-1}(\pi f_2 T) = \zeta_2$$

(Note: $0 \leq \zeta_1 < \zeta_2 \leq 1/2T$)

The mapping relationship can be expressed as:

$$H(e^{j2\pi\zeta T}) = H(j2\pi f) \Big|_{f = (1/\pi T)\tan(\pi\zeta T)}$$

This relationship is illustrated below. Note that the equally spaced passbands in the analog filter are warped as they translate to the digital filter.

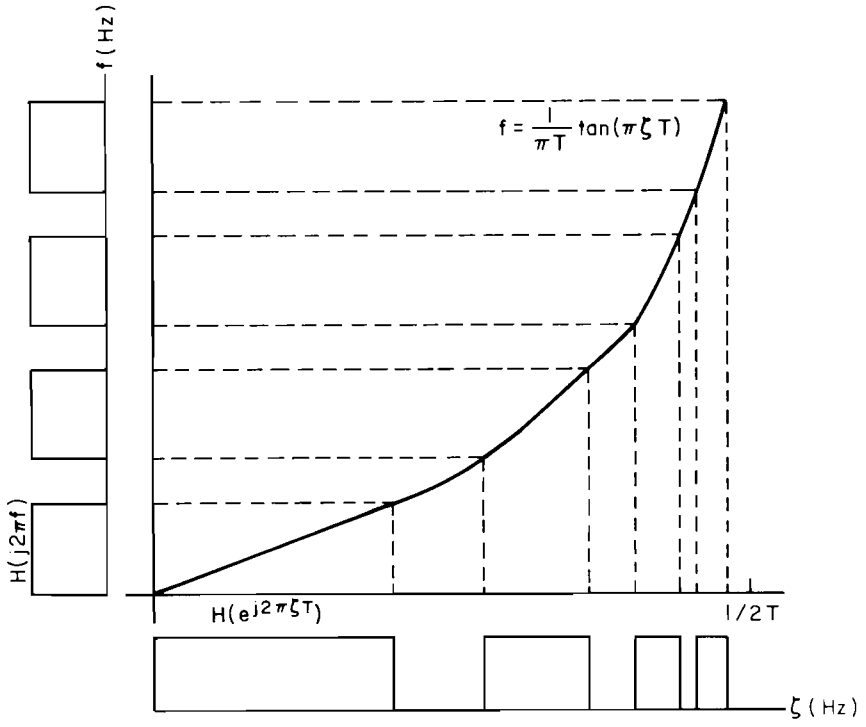


Figure 31

In this software, a prewarping parameter may be entered. This parameter replaces 2/T in the classical bilinear-z transformation with a constant c. Thus:

$$s \rightarrow c \frac{1 - z^{-1}}{1 + z^{-1}}$$

The prewarping parameter allows you to obtain $H(e^{j2\pi\zeta_0 T}) = H(j2\pi f_0)$ where you specify both ζ_0 and f_0. (See the examples in Chapter 5). c is determined by the relationship:

$$c = 2\pi f_0 \cot(\pi\zeta_0 T).$$

For more information on the bilinear-z transformation, see Reference 2 or a digital filter design text.

Impulse Invariance

The method of impulse invariance is based on the design of a digital filter whose unit pulse response is a sampled version of the analog filter impulse response. That is:

$$h_k = h(t = kT);\ k = 0,1,2, \ldots$$

This condition causes $H(e^{j2\pi fT})$ to look like an infinite sum of periodically extended versions of $H(j2\pi f)$:

$$H(e^{j2\pi fT}) = 1/T \sum_{n=-\infty}^{\infty} H(j2\pi f + j2\pi n/T)$$

Now, if $H(j2\pi f) = 0$ for $|f| > 1/2T$ (that is, $H(j2\pi f)$ is "bandlimited" by the foldover frequency), then:

$$H(e^{j2\pi fT}) = 1/T\ H(j2\pi f)$$

for $|f| < 1/2T$. The program automatically multiplies the resulting H(z) by T to normalize the gain. In practice, since no analog frequency response can be truly bandlimited, some aliasing always occurs.

For more information on impulse invariance, see Reference 2 or a digital filter design text.

Covariance Invariance

The method of covariance invariance is based on the design of a digital filter whose covariance sequence is a sampled version of the analog filter covariance function. That is:

$$r_k = R(\tau = kT);\ k = 0, \pm 1, \pm 2, \ldots$$

This property causes a second-order statistical equivalence to be established between the analog filter output and the digital filter output. The result is that the magnitude-squared frequency response $|H(e^{j2\pi fT})|^2$ is an infinite sum of periodically extended versions of $|H(j2\pi f)|^2$:

$$|H(e^{j2\pi fT})|^2 = 1/T \sum_{n=-\infty}^{+\infty} |H(j2\pi f + j2\pi n/T)|^2$$

For more information about the method of covariance invariance, see Reference 5.

Inputs

The inputs to the transformation may be either the numerator and denominator coefficients or the poles and the zeros of the analog transfer function. To enter the coefficients of the analog transfer function, the transfer function must be in the following form:

$$H(s) = (\text{gain constant}) \cdot \frac{1 + b_1 s + \ldots + b_M s^M}{1 + a_1 s + \ldots + a_N s^N}$$

Enter: gain constant, a_k, b_k, M, N

To enter the poles and zeros of the analog transfer function, the transfer function must be in the following form:

$$H(s) = (\text{gain constant}) \cdot \frac{\prod_{i=1}^{M} (s - (a_i + jb_i))}{\prod_{i=1}^{N} (s - (c_i + jd_i))}$$

Enter: gain constant, N, M, a_k, b_k, c_k, d_k

The poles and zeros must be entered in the following order:

1. complex pairs ($|a_k| \neq 0$, $|b_k| \neq 0$, for example)
2. real numbers ($|a_k| \neq 0$, $b_k = 0$, for example)
3. poles or zeros at zero ($a_k = 0$, $b_k = 0$, for example)

The analog filter must be of order less than 20.

Select the type of transformation you wish to use. Once an analog filter has been input (and transformed), it may be re-transformed using one of the other transformation techniques. To do this, re-press [k8] and answer the prompts appropriately.

Remarks

1. Unstable or marginally unstable analog filters (that is, filters with poles in the right half plane or on the imaginary axis) can be entered and transformed. However, the frequency response cannot be plotted for unstable filters.
2. A stable analog filter will produce a stable digital filter.
3. The covariance invariance and impulse invariance methods cannot be performed on analog filters whose numerator order is greater than or equal to the denominator order.
4. The covariance invariance and impulse invariance transformations cannot be performed on analog filters with multiple poles. That is, all analog poles must be unique.
5. The method of covariance invariance is preferred when magnitude-squared frequency response approximation is desired, or when random data is to be simulated with a digital filter. The method of impulse invariance is preferred when complex frequency response is to be approximated or when a digital filter is to be used for simulating a transient response. The method of bilinear-z always mitigates the effects of aliasing, but leads to designs which match neither the impulse response nor the covariance function.

k9: ENT DIGT (Enter a Digital Filter)

In this design mode, you may enter the transfer function of your own AR, MA, or ARMA digital filter. This permits you to analyze the filter for its time- and frequency- domain responses.

Inputs

You may enter your digital transfer function as a ratio of polynomials or as a ratio of products of poles and zeros:

(i) $H(z) = (\text{gain constant}) \cdot \dfrac{(1 - z^{-1}z_1)(1 - z^{-1}z_2)\cdot\ldots\cdot(1 - z^{-1}z_M)}{(1 - z^{-1}p_1)(1 - z^{-1}p_2)\cdot\ldots\cdot(1 - z^{-1}p_N)}$

(z_i and p_i are in general complex).

(ii) $H(z) = (\text{gain constant}) \cdot \dfrac{1 + b_1z^{-1} + \ldots + b_Mz^{-M}}{1 + a_1z^{-1} + \ldots + a_Nz^{-N}}$

(b_i and a_i are real).

If you enter your transfer function as a ratio of polynomials, then you may enter the b_i and a_i coefficients from keyboard or data file. If you enter it as a ratio of products of poles and zeros, then you may enter the z_i and p_i from keyboard or from the CRT using a built-in graphics routine for placing poles and zeros on the unit circle.

Remarks

None.

OUTPUT MENU

Whether you have designed a digital filter, transformed an analog filter, or entered a digital filter, you will want to analyze it. In the MAIN MENU, press [k1]: OUTPUT. Here is the output menu you see:

DIGITAL FILTER DESIGN: OUTPUT MENU

KEY 0: 'MAIN' ---------- MAIN KEY MENU
KEY 1: 'DF FREQ' ---------- PLOT OR LIST THE FREQUENCY RESPONSE OF A DIGITAL FILTER
KEY 2: 'DF IMP' ---------- PLOT OR LIST THE IMPULSE RESPONSE OF A DIGITAL FILTER
KEY 3: 'A D FREQ' ---------- PLOT OR LIST THE FREQUENCY RESPONSE OF AN ANALOG FILTER
KEY 4: 'TABLE 1' ---------- OUTPUT DIGITAL FILTER AS SINGLE SECTION
KEY 5: 'TABLE 2' ---------- OUTPUT DIG FILTER AS PROD OF 2ND ORDER SECTIONS
KEY 6: 'P Z PLOT' ---------- CALCULATE AND PLOT POLES AND ZEROS
KEY 7: 'ROUND' ---------- ROUND FILTER COEFFICIENTS
KEY 8: 'NORM' ---------- NORMALIZE D A GAIN

SELECT KEY

MAIN	DF FREQ	DF IMP	A/D FREQ	TABLE 1
TABLE 2	P/Z PLOT	ROUND	NORM	OUT MENU

Note the SELECT KEY. By pressing [k1] through [k8], you can exercise all of the OUTPUT (or analysis) features of the software. The following paragraphs contain key-by-key descriptions of the output options.

[k1]: DF FREQ (Plot or List the Frequency Response of a Digital Filter)

This option allows you to obtain lists or plots of magnitude and phase frequency responses for your digital filter using a variety of linear and db scales over a variety of frequency ranges. Frequency ZOOM computes the exact frequency response at arbitrary frequencies over an interval you specify. Frequency responses over 256 frequency points between 0 and FOLDOVER are computed exactly at 256 equally spaced FFT frequencies between 0 and FOLDOVER.

Prompts
Answer the following list of prompts:

Digital Filter Frequency Response

```
256 POINTS (0 TO FOLDOVER) (1) OR ZOOM
(TABULAR ONLY) (2), (DEFAULT=1) ?

?

PLOT (1), OR TABULAR (2), (DEFAULT=1) ?

?

MAGNITUDE (1), OR PHASE (2), (DEFAULT=1) ?

?

LOG PLOT (1), OR LINEAR PLOT (2), (DEFAULT=1) ?

?

SCALE DESIRED: 0db TO -100db (1), -10db (2),
-1db (3) (DEFAULT=1) ?

?

LABEL FOR GRAPH (MAX 38 CHAR, DEFAULT=NO LABEL) ?

BAR (B), OR LINE (L) GRAPH (DEFAULT=L) ?

PLOTTER IS X,Y$ (DEFAULT= 3 ,"INTERNAL") ?
```

k2: DF IMP (Plot or List the Impulse Response of a Digital Filter)

This option allows you to obtain lists and plots of the impulse response for your digital filter using several time scales.

Prompts

Answer the following list of prompts:

```
DIGITAL FILTER IMPULSE RESPONSE

PLOT (1), or TABULAR (2), (DEFAULT=1) ?

NUMBER OF SAMPLES (32,64,128,256 DEFAULT=32) ?
```
(This prompt appears with IIR filters only.)

```
USING SECOND ORDER SECTIONS (1) OR SINGLE SECTION (2)
(DEFAULT=1) ?
```
(This prompt appears with IIR filters only.)

```
LABEL FOR GRAPH (MAX 38 CHAR, DEFAULT=NO LABEL) ?

PLOTTER IS X,Y$ (DEFAULT=3,"INTERNAL") ?
```

[k3]: A/D FREQ (Plot or List the Frequency Response of an Analog Filter)

This output option is only available if you have entered an analog filter for transformation to a digital filter. This option is identical with [k1]: DF FREQ, except you may obtain analog or analog-digital overplots of frequency responses with this option.

Prompts

Answer these prompts:

Analog/Digital Filter Frequency Response

```
256 POINTS (0 TO FOLDOVER) (1) OR ZOOM
(TABULAR ONLY) (2), (DEFAULT=1) ?

?

PLOT (1), OR TABULAR (2), (DEFAULT=1) ?

?

MAGNITUDE (1), OR PHASE (2), (DEFAULT=1) ?

?

LOG PLOT (1), OR LINEAR PLOT (2), (DEFAULT=1) ?

?

ANALOG PLOT(1) OR ANALOG-DIGITAL OVERPLOT(2)
(DEFAULT=2) ?

?

SCALE DESIRED: 0db TO -100db (1), -10db(2),
-1db(3) (DEFAULT=1) ?

?

LABEL FOR GRAPH (MAX 38 CHAR, DEFAULT=NO LABEL) ?

PLOTTER IS X,Y$ (DEFAULT= 3 ,"INTERNAL") ?
```

[k4]: TABLE 1 (Output Digital Filter as Single Section)

This option allows you to tabulate and list numerator and denominator coefficients for your digital filter transfer function. Once the table of coefficients has been displayed, the following prompt will appear:

```
DO YOU WANT TO SAVE THESE COEFFICIENTS IN A DATA FILE (Y OR
N  DEFAULT=N)?
```

If you don't want to save the coefficients, press [CONTINUE]. To save the coefficients, input "Y" and answer the prompts that follow. It should be noted that:

1. Saved transfer functions can be re-entered using [k9], ENT DIGT, of the DESIGN menu,
2. The gain constant of MA, AR, and ARMA filters is stored with the coefficients, but the system type (MA, AR, or ARMA) is not stored, nor is the order, and
3. IIR filters will be stored in a single transfer function form only, and the second-order section form will not be available when the filter is re-entered using ENT DIGT.

[k5]: TABLE 2 (Output Digital Filter as a Product of Second Order Sections)

This option is only available if you have designed an automated IIR filter. It allows you to break your digital filter transfer function up into a product of second order sections and to list the numerator and denominator coefficients for each section. There are no prompts to be answered.

[k6]: P/Z PLOT (Calculate and Plot Poles and Zeros)

This option allows you to calculate, list, and plot the poles and zeros of a digital filter transfer function. The plots of poles and zeros are always constructed by reflecting poles and zeros outside the unit circle to their reciprocal values inside.

Note

For linear phase filters, zeros which lie inside the unit circle always accompany their own reciprocal zeros outside the unit circle. Therefore, the plotting routine will graph these zeros on top of each other inside the unit circle. Keep this in mind if you don't see enough zeros on the pole/zero plot of a linear phase filter. (You can always check the tabulated listing of poles and zeros to see all the zeros.)

Prompts

Answer the following prompts for a plot:

```
WOULD YOU LIKE TO TEST THESE ROOTS (Y OR N)?
```

(This prompt is not given for IIR filters stored in second-order sections.)

```
WOULD YOU LIKE A POLE/ZERO PLOT (Y OR N)?
```

(Note: Pole/zero calculations can only be obtained for filters of order 16 or less.)

[k7]: ROUND (Round Filter Coefficients)

With this option you can round the mantissa of all digital filter transfer function coefficients.

Prompts

Answer this prompt:

```
NUMBER OF SIGNIFICANT DIGITS (1,2,...,9)?
```

[k8]: NORM (Normalize D/A Gain)

This output option is available only if you have entered an analog filter. If this option is not used, the analog-digital frequency response magnitude overplots are plotted with both responses normalized to the greatest magnitude found on the plot (be it from the analog or digital response). If this option is used, each of the frequency responses is normalized according to its own maximum magnitude. The gain constant for the digital filter is adjusted so the maxima for both frequency responses are the same. There are no prompts to be answered.

Chapter 5
Examples

In this chapter we work through three examples that illustrate the use of the pack. In the first example, magnitude response specifications are automatically met with an FIR filter. Several analysis options are exercised. In the second example, a bandstop filter is designed and the frequency zoom function is illustrated. In the third example, an analog transfer function is transformed into a digital transfer function using the bilinear-z transformation technique.

Example 1: Design of Lowpass FIR Filter

Example 1 begins with the magnitude response specifications for a lowpass (1) filter to be approximated with an FIR digital filter using the Fourier Series technique. Refer to Figure 25 in Chapter 4.

Press [k3]: FOURIER in the DESIGN MENU and answer a prompt to carry out a lowpass (L) design. Enter these design parameters:

SAMPLING PERIOD = .001
f_p = 250
f_a = 375
A_p = 1
A_a = 35

The filter is automatically designed.

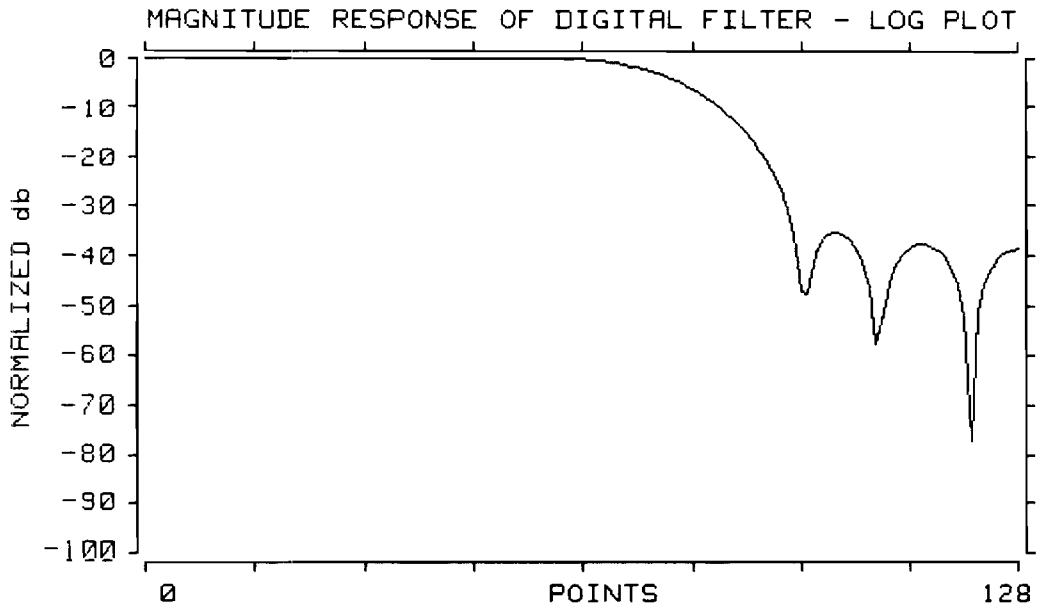
To analyze the filter, press [k0]: MAIN. Then press [k1]: OUTPUT. In the OUTPUT MENU, press [k4]: TABLE 1. You will see the following table of filter coefficients:

FILTER SPECIFICATIONS	VALUE
TYPE	LOWPASS
SAMPLING PERIOD	1.0000E-03
PASSBAND CORNER FREQUENCY	2.5000E+02
STOPBAND CORNER FREQUENCY	3.7500E+02
MAX. db RIPPLE IN PASSBAND	3.0895E-01
db REJECTION IN STOPBAND	3.5000E+01

Z TO POWER	COEFFICIENT	Z TO POWER	COEFFICIENT
0	5.940442835E-18	-1	1.544652219E-02
-2	-1.869810363E-02	-3	-1.530776257E-02
-4	5.961572571E-02	-5	-3.463291439E-02
-6	-1.049637587E-01	-7	2.890424314E-01
-8	6.250000000E-01	-9	2.890424314E-01
-10	-1.049637587E-01	-11	-3.463291439E-02
-12	5.961572571E-02	-13	-1.530776257E-02
-14	-1.869810363E-02	-15	1.544652219E-02
-16	5.940442835E-18		

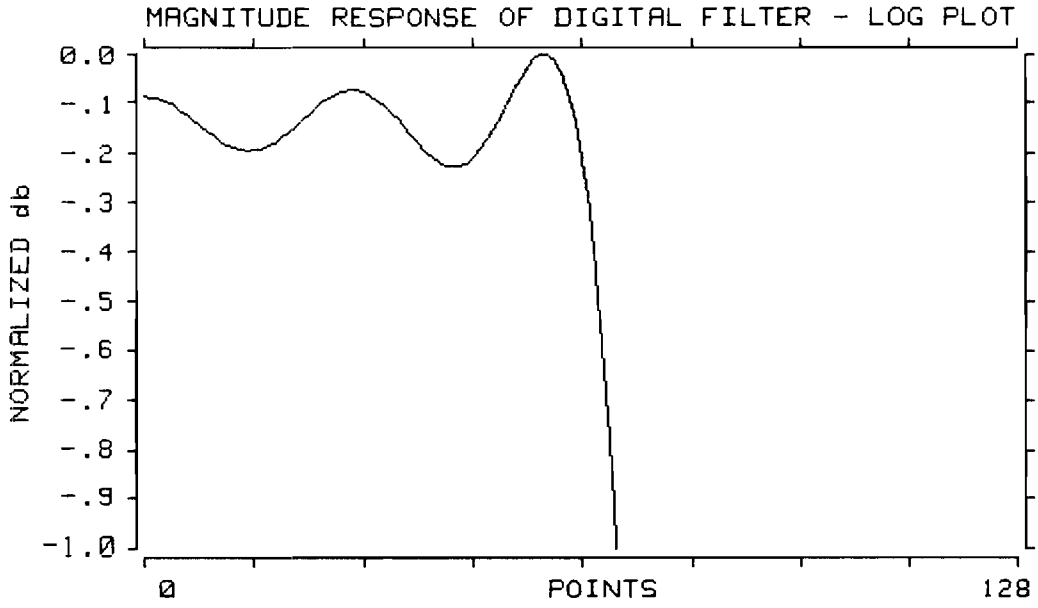
Note that the ripple in the passband is .309 rather than 1. This is an improvement on the frequency response specifications.

Press [k1]: DF FREQ in the OUTPUT MENU. Choose (1) 256 POINTS (0 to FOLDOVER), PLOT (1), MAGNITUDE (1), LOG PLOT (1), and for scale desired, select 0db to −100db (1). Label plot as you wish. Choose LINE (L) and PLOTTER is 3, "INTERNAL". Here is the frequency response you see:



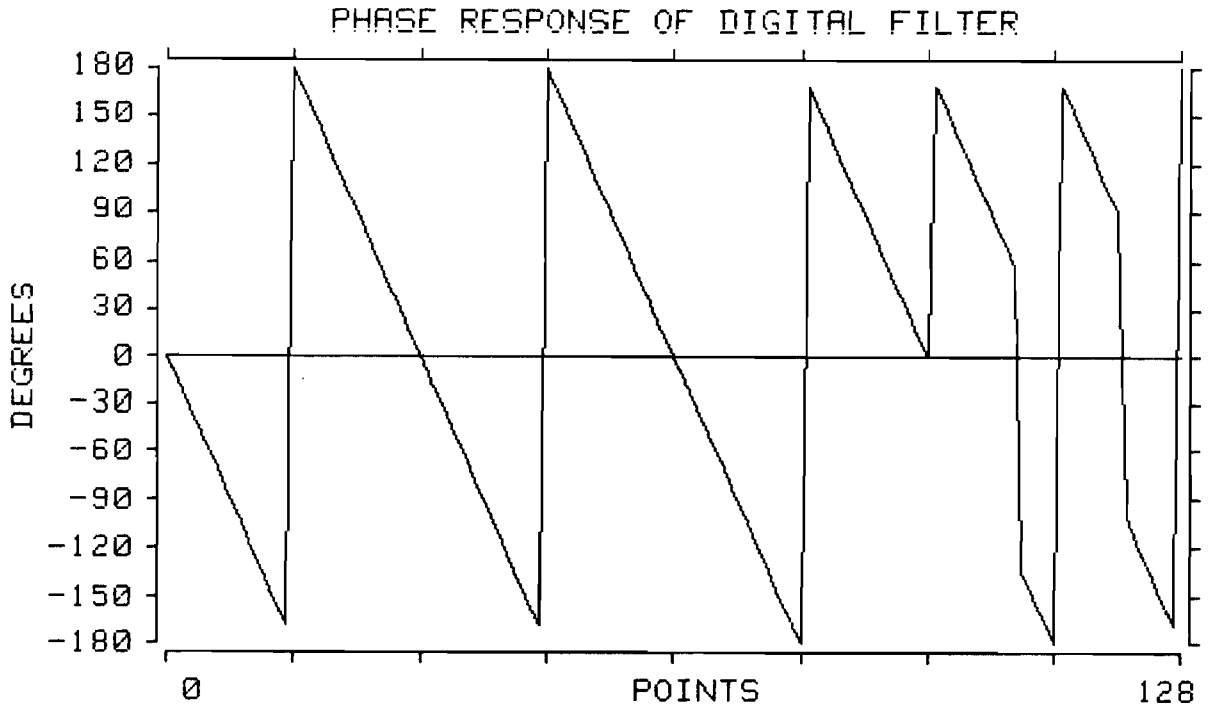

```
MAXIMUM MAGNITUDE: 0= 1.3932E-01 db
FREQUENCY RESOLUTION = 3.9063E+00 Hz
FOLDOVER FREQUENCY = 5.0000E+02 Hz = POINT  128
```

Press [k1]: DF FREQ again. For scale desired, select 0db to −1db (3). You will see the following plot of the passband ripple:




```
MAXIMUM MAGNITUDE: 0= 1.3932E-01 db
FREQUENCY RESOLUTION = 3.9063E+00 Hz
FOLDOVER FREQUENCY = 5.0000E+02 Hz = POINT  128
```

Press [k1]: DF FREQ again. Choose PHASE (2) to view this phase spectrum.

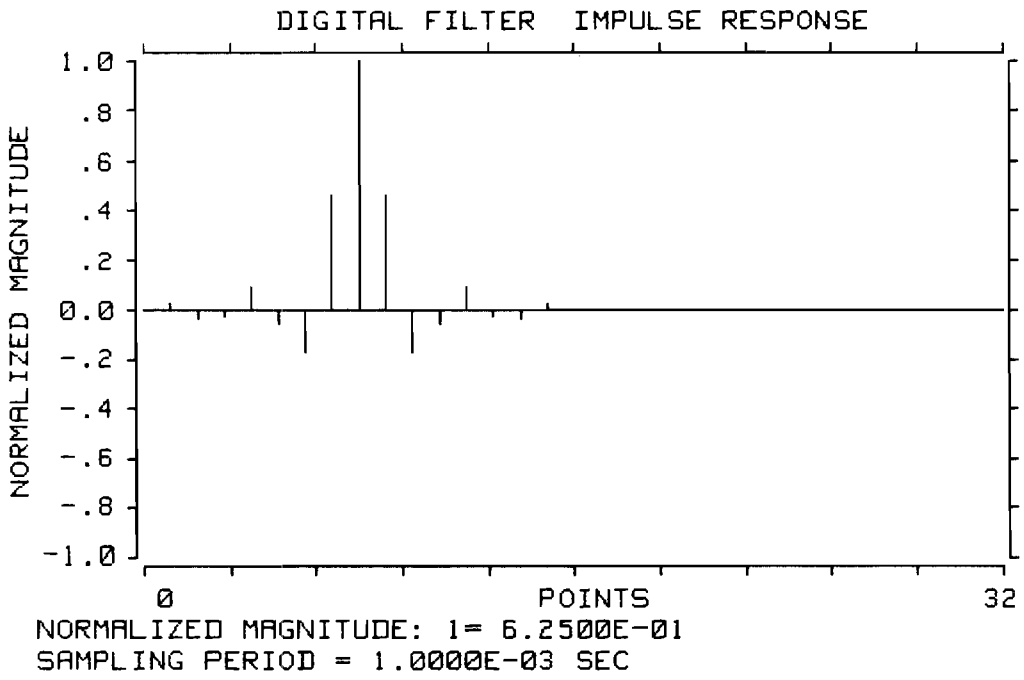



```
FREQUENCY RESOLUTION = 3.9063E+00 Hz
FOLDOVER FREQUENCY = 5.0000E+02 Hz = POINT  128
```

Press [k1]: DF FREQ again. Choose TABULAR (2) and specify points 0,9 to obtain this listing of frequency response values:

```
DIGITAL FREQUENCY RESPONSE
**************************************************
POINT    MAG.    20*LGT(MAG) 20*LGT(NORM) PHASE(DEG)
**************************************************
  0  1.006E+00   5.200E-02  -8.732E-02   0.000E+00
  1  1.006E+00   5.090E-02  -8.842E-02  -1.125E+01
  2  1.005E+00   4.764E-02  -9.168E-02  -2.250E+01
  3  1.005E+00   4.235E-02  -9.697E-02  -3.375E+01
  4  1.004E+00   3.525E-02  -1.041E-01  -4.500E+01
  5  1.003E+00   2.661E-02  -1.127E-01  -5.625E+01
  6  1.002E+00   1.679E-02  -1.225E-01  -6.750E+01
  7  1.001E+00   6.195E-03  -1.331E-01  -7.875E+01
  8  9.995E-01  -4.750E-03  -1.441E-01  -9.000E+01
  9  9.982E-01  -1.559E-02  -1.549E-01  -1.013E+02
```

Press [k2]: DF IMP to generate an impulse response for your filter. Choose PLOT (1) and label as you wish. Default to PLOTTER is 3, "INTERNAL". Here is what you see:



Press [k2]: DF IMP again. Choose TABULAR (2) and specify FIRST AND LAST POINTS as 0,16. Here is the listing of impulse response values you see:

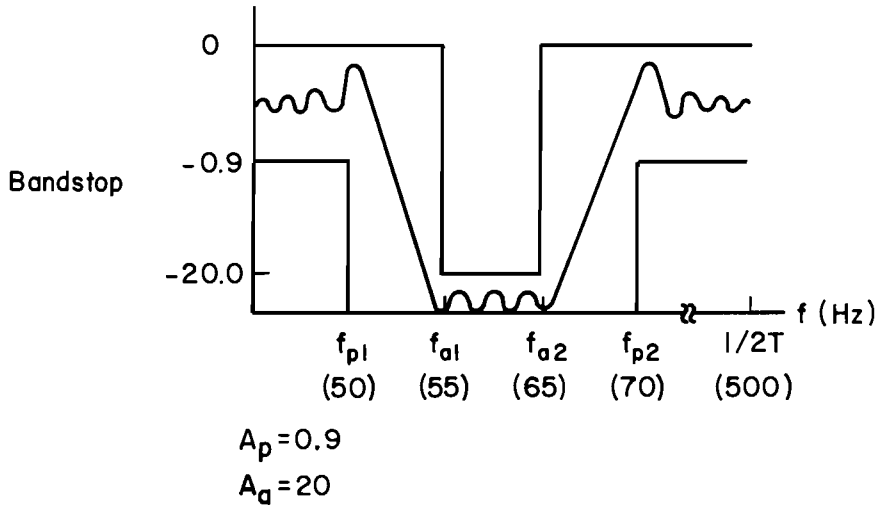
```
IMPULSE RESPONSE
*******************************
 POINT                VALUE
*******************************
     0            5.94044E-18
     1            1.54465E-02
     2           -1.86981E-02
     3           -1.53078E-02
     4            5.96157E-02
     5           -3.46329E-02
     6           -1.04964E-01
     7            2.89042E-01
     8            6.25000E-01
     9            2.89042E-01
    10           -1.04964E-01
    11           -3.46329E-02
    12            5.96157E-02
    13           -1.53078E-02
    14           -1.86981E-02
    15            1.54465E-02
    16            5.94044E-18
```

Notice that, for the special case of an FIR filter, the values of the impulse response equal the coefficients of the filter.

You may want to round the filter coefficients to a specified number of digits in order to see what effect this has on the frequency and impulse responses. This allows you to analyze a finite-word-length approximation to your design. Press [k7]: ROUND from the OUTPUT MENU to do this. Proceed as you wish to analyze the rounded filter.

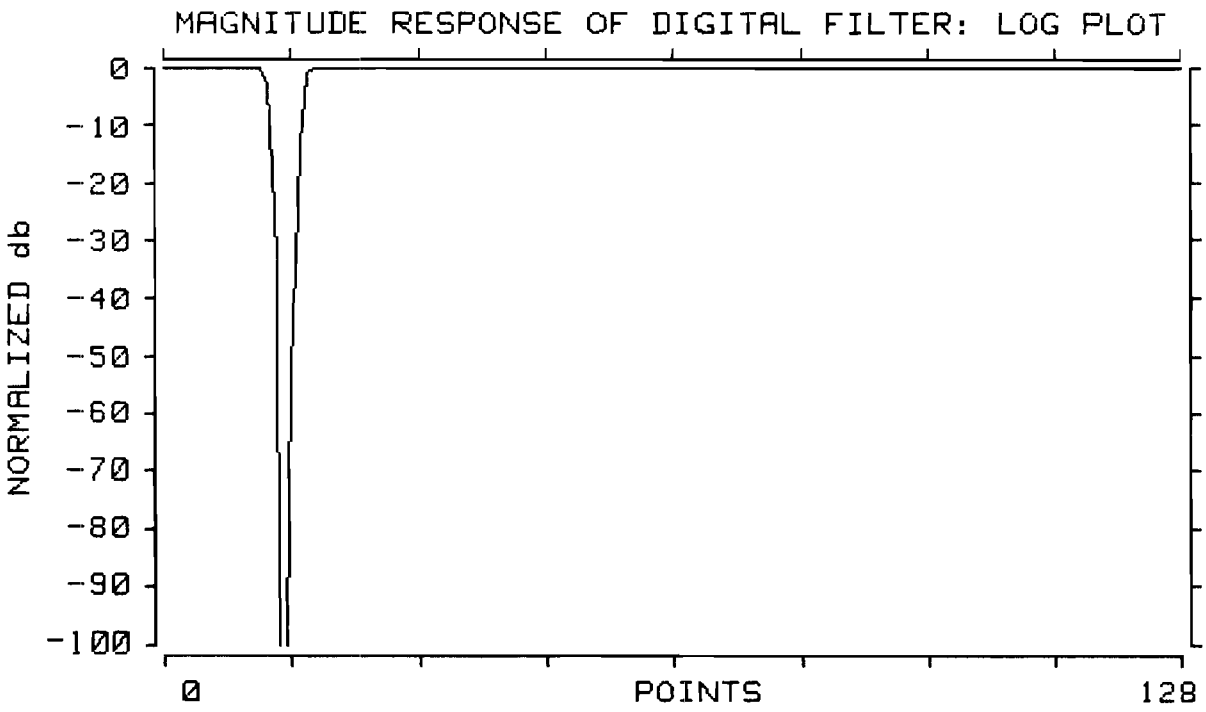
Example 2: Frequency Response Zoom

This example begins with the following magnitude response specifications for an IIR, Butterworth, bandstop filter to be designed with the bilinear-z technique:



Note that the bandstop region is very narrow compared with the range of frequencies between dc and foldover.

Press [k1] : IIR BILZ and answer prompts to design a Butterworth filter to these specifications. Proceed as outlined in the previous example to obtain the magnitude frequency response on a 0db to −100db scale:




```
MAXIMUM MAGNITUDE: 0= 1.9287E-15 db
FREQUENCY RESOLUTION = 3.9063E+00 Hz
FOLDOVER FREQUENCY = 5.0000E+02 Hz = POINT  128
```

This plot does not contain much useful information about the frequency region of interest between 50 Hz and 70 Hz. You may "ZOOM IN" on this stopband region by pressing [k1]: DF FREQ again and choosing ZOOM (2). Answer a prompt for 25 values on the interval [50,70] to obtain this listing.

```
DIGITAL FREQUENCY RESPONSE
**************************************************
FREQUENCY(HZ)   MAGNITUDE   20*LGT(MAG)   PHASE(DEG)
**************************************************
 5.0000E+01    9.0157E-01   -9.0000E-01   1.3244E+02
 5.0833E+01    7.4572E-01   -2.5484E+00   9.6786E+01
 5.1667E+01    4.9322E-01   -6.1391E+00   5.6672E+01
 5.2500E+01    2.5805E-01   -1.1766E+01   1.9057E+01
 5.3333E+01    1.1398E-01   -1.8863E+01  -1.3371E+01
 5.4167E+01    4.3617E-02   -2.7207E+01  -4.1777E+01
 5.5000E+01    1.4076E-02   -3.7030E+01  -6.7596E+01
 5.5833E+01    3.5712E-03   -4.8944E+01  -9.1714E+01
 5.6667E+01    6.2060E-04   -6.4144E+01  -1.1465E+02
 5.7500E+01    5.4310E-05   -8.5302E+01  -1.3672E+02
 5.8333E+01    9.1492E-07   -1.2077E+02  -1.5816E+02
 5.9167E+01    2.9115E-15   -2.9072E+02  -1.7916E+02
 6.0000E+01    5.2112E-07   -1.2566E+02   1.6012E+02
 6.0833E+01    3.6321E-05   -8.8797E+01   1.3955E+02
 6.1667E+01    4.1487E-04   -6.7642E+01   1.1897E+02
 6.2500E+01    2.2919E-03   -5.2796E+01   9.8242E+01
 6.3333E+01    8.5328E-03   -4.1378E+01   7.7167E+01
 6.4167E+01    2.4780E-02   -3.2118E+01   5.5502E+01
 6.5000E+01    6.0600E-02   -2.4351E+01   3.2883E+01
 6.5833E+01    1.3019E-01   -1.7709E+01   8.7595E+00
 6.6667E+01    2.5010E-01   -1.2038E+01  -1.7593E+01
 6.7500E+01    4.2661E-01   -7.3993E+00  -4.6643E+01
 6.8333E+01    6.2993E-01   -4.0142E+00  -7.7424E+01
 6.9167E+01    7.9869E-01   -1.9524E+00  -1.0696E+02
 7.0000E+01    9.0157E-01   -9.0000E-01  -1.3244E+02
```

Example 3: Transform an Analog Filter

This example begins with an analog transfer function:

$$H(s) = 1 * \frac{1}{(s - (-\sin\frac{\pi}{8} - j\cos\frac{\pi}{8}))} * \frac{1}{(s - (-\sin\frac{\pi}{8} + j\cos\frac{\pi}{8}))} * \frac{1}{(s - (-\cos\frac{\pi}{8} - j\sin\frac{\pi}{8}))} * \frac{1}{(s - (-\cos\frac{\pi}{8} + j\sin\frac{\pi}{8}))}$$

This is a Butterworth transfer function. We are looking for a digital equivalent of it.

Press [k8]: ENT ANLG. Answer the prompt for SAMPLING PERIOD (in seconds):

1.2

Answer the prompt to enter H(s) in FORM B(B), which is a ratio of products of poles and zeros. Specify the numerator degree as 0 and the denominator degree as 4. Enter the following pole parameters:

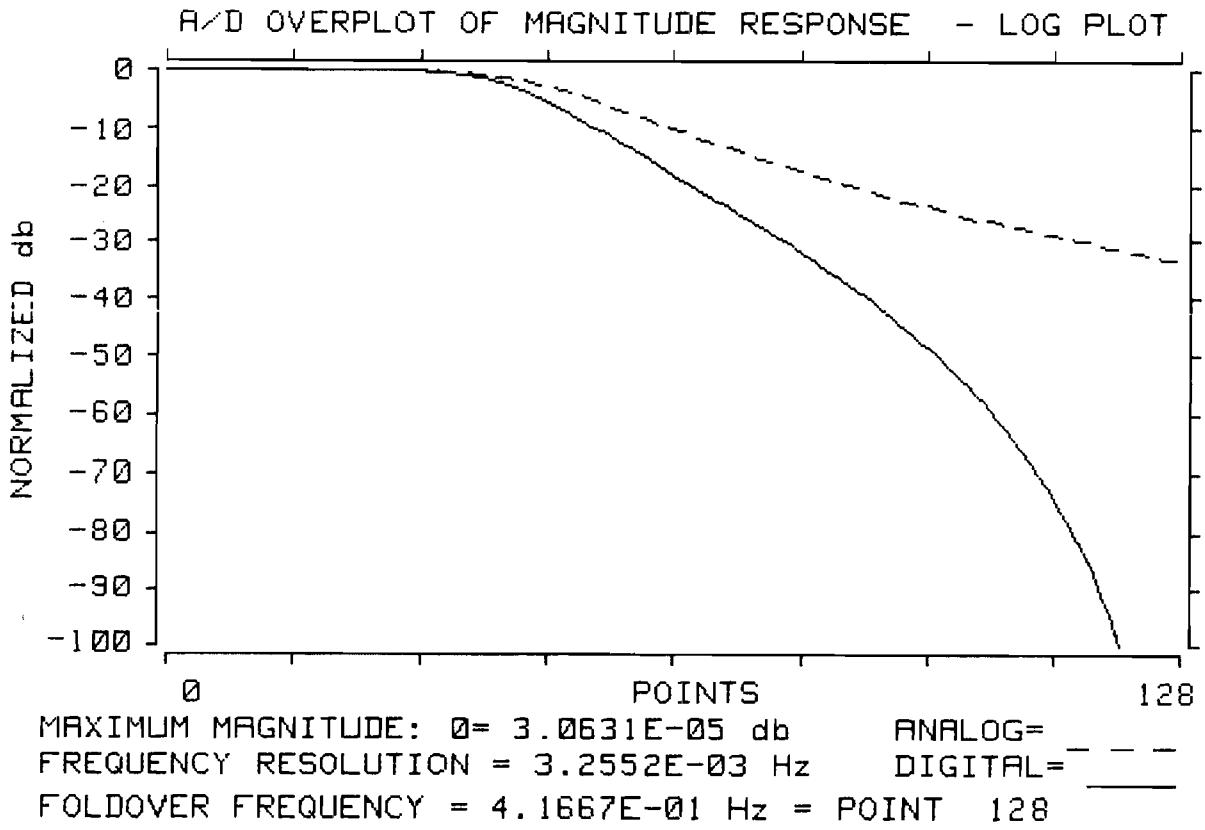
REAL PART OF POLE 1: $-\sin(\pi/8) = -0.38268$
IMAG. PART OF POLE 1: $-\cos(\pi/8) = -0.92388$
REAL PART OF POLE 3: $-\cos(\pi/8) = -0.92388$
IMAG. PART OF POLE 3: $-\sin(\pi/8) = -0.38268$

Note you are not prompted for poles 2 and 4. These are complex conjugates of poles 1 and 3. Thus they are added automatically. Choose the gain constant to be 1. You will see this graphic summary of the analog filter you have entered:

```
| POLE NUMBER |     REAL PART     | IMAGINARY PART   |
|      1      |  -3.8268E-01      |   -9.2388E-01    |
|      2      |  -3.8268E-01      |    9.2388E-01    |
|      3      |  -9.2388E-01      |   -3.8268E-01    |
|      4      |  -9.2388E-01      |    3.8268E-01    |
|_____________|___________________|__________________|
GAIN CONSTANT = 1.00000E+00
SAMPLING PERIOD= 1.20000E+00SEC.



WOULD YOU LIKE TO MAKE ANY CHANGES ?
(Y OR N  DEFAULT=N)

?
```

Answer no (N). Select BILINEAR-z (B) as your desired transform and choose the default prewarping parameter by pressing [CONTINUE].

Return to MAIN MENU. Press [k1]: OUTPUT and then [k3]: A/D FREQ. Select 256 points (1). Choose PLOT (1), MAGNITUDE (1), LOG PLOT (1), and ANALOG/DIGITAL OVERPLOT (2). Select 0db to −100db (1) as the desired scale. Label as you wish and default to internal plotter. Here is the frequency domain overplot you see:



The 3-decibel cutoff frequency for the analog filter is $1/2\pi$ Hz. Notice that, because of frequency warping, the 3-decibel point for the digital filter is not at 1/2Hz. To adjust the digital filter so that the 3-decibel point translates exactly, compute the prewarping parameter:

$$c = 2\pi f_0 \cot(\pi \zeta_0 T)$$

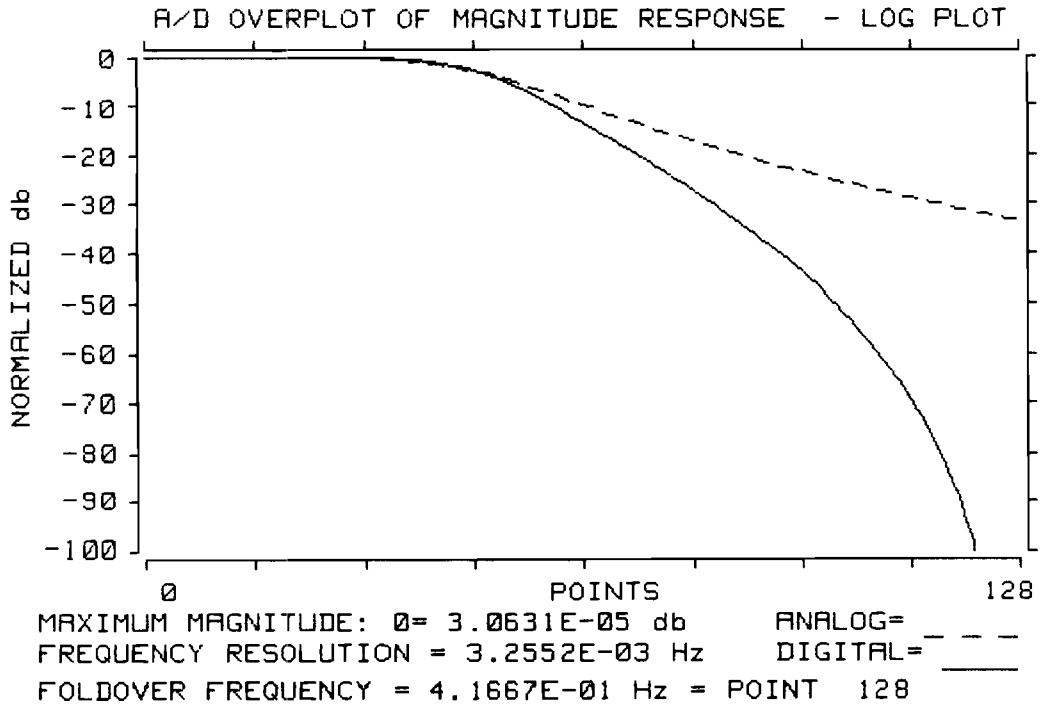
$$c = 2\pi(1/2\pi)\cot(\pi 1/2\pi)(1.2))$$

$$c = \cot(.6) = 1.4617$$

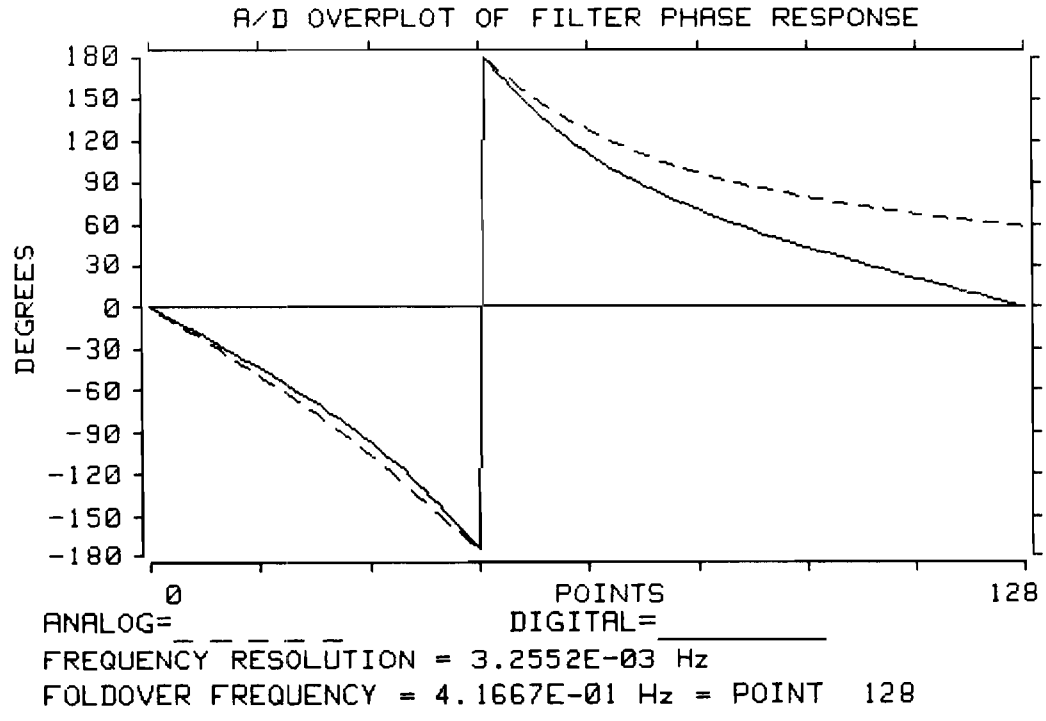
Return to MAIN MENU. Press [k0]: DESIGN and then [k8]: ENT ANLG to retransform the filter. Answer the prompt indicating you want to TRANSFORM THE PRESENT FILTER (1). Proceed to the prompt for the prewarping parameter and enter:

1.4617

Return to MAIN MENU and then to the OUTPUT MENU. Proceed as before to obtain this overplot of the analog and digital frequency responses. Note the new cutoff frequency for the digital filter.

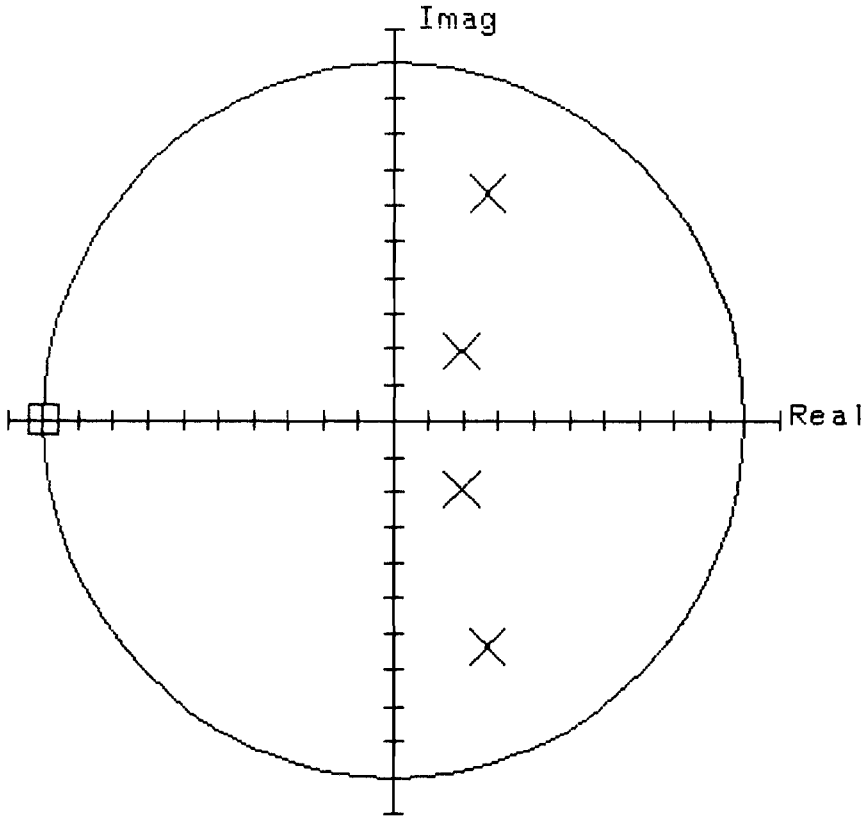


Press [k3]: A/D FREQ and answer prompts to get this phase plot:



Finally, press [k6]: P/Z PLOT and answer prompts to obtain this listing and plot digital filter poles and zeros. Note that a bilinear-z designed digital filter always has N (=4) poles and N (=4) zeros when the analog filter has N (=4) poles. The zeros of the bilinear-z design all lie on top of each other at $z = -1$.

```
ZERO#   MAGNITUDE   PHASE(DEG)     REAL      IMAG.
**************************************************
  1:   1.000E+00   1.800E+02 -1.000E+00   0.000E+00
  2:   1.000E+00   1.800E+02 -1.000E+00   0.000E+00
  3:   1.000E+00   1.800E+02 -1.000E+00   0.000E+00
  4:   1.000E+00   1.800E+02 -1.000E+00   0.000E+00

POLE#   MAGNITUDE   PHASE(DEG)     REAL      IMAG.
**************************************************
  1:   6.886E-01   6.718E+01   2.671E-01   6.347E-01
  2:   6.886E-01  -6.718E+01   2.671E-01  -6.347E-01
  3:   2.732E-01   4.455E+01   1.947E-01   1.916E-01
  4:   2.732E-01  -4.455E+01   1.947E-01  -1.916E-01

GAIN CONSTANT =   4.0258E-02
```



References

[1] *IEEE Trans. on Acoustics, Speech, and Signal Processing*, (published bi-monthly).

[2] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1975.

[3] *Programs for Digital Signal Processing*, edited by the Digital Signal Processing Committee of the IEEE Acoustics, Speech, and Signal Proc. Society, IEEE Press 1979.

[4] A. Antoniou, *Digital Filter: Analysis and Design*, McGraw-Hill, New York, 1979.

[5] J. Perl and L. L. Scharf, "Covariance-Invariant Digital Filtering," *IEEE Trans. Acoustics, Speech, and Signal Proc.*, 25, pp. 143-151 (April 1977).

[6] D. C. Farden and L. L. Scharf, "Statistical Design of Nonrecursive Digital Filters," *IEEE Trans. Acoustics, Speech, and Signal Proc.*, 22, pp. 188-196 (June 1974).

[7] H. Clergeot and L. L. Scharf, "Connections Between Classical and Statistical Methods of FIR Digital Filter Design," *IEEE Trans. Acoustics, Speech, and Signal Proc.*, 26, pp. 463-465 (Oct. 1978).

[8] J. H. McClellan and T. W. Parks, "Chebyshev Approximation for Nonrecursive Digital Filters with Linear Phase," *IEEE Trans. on Circuit Theory*, 19, pp. 188-194, (March 1972).

[9] L. L. Scharf and J. C. Luby, "Statistical Design of Autoregressive Moving Average Digital Filters," *IEEE Trans. Acoustics, Speech, and Signal Processing*, ASSP-27, No. 3, (June 1979).