SIGAN

User Manual

Table of Contents

I INTRODUCTION

SIGAN is a software package for Digital Signal Analysis and Processing. SIGAN can be used to design both IIR and FIR filters, analyze spectral characteristics of discrete-time signals and convolution of signal with a filter impulse response, and synthesize composite signals imbedded in various types of noise.

Digital Filters can be designed to arbitrary frequency response specification or by truncating a discrete Fourier series. Design of Infinite-Impulse Response (IIR) filters is based on approximations of analog filters such as Butterworth, Chebyshev, and Elliptic. Finite-impulse Response (FIR) filters can be designed using Rabiner-McClennan-Parks (RMP) algorithm which uses linear programming approach for optimization or window method which uses truncation of discrete Fourier series. Types of windows supported are Rectangular, Triangular, Hamming, Generalized Hamming, Hann, Kaiser and Chebyshev.

SIGAN provides both time and frequency analysis capability. To obtain frequency domain information, fast Fourier transform (FFT) is used to convert time domain discrete samples to spectral samples. SIGAN time analysis capabilities include discrete convolution.

SIGAN can be used to generate a composite signal which consists of multiple sinusoids with different amplitudes, phases, and frequencies. In addition to the synthesis of this complex signal, SIGAN simulates Gaussian noise characteristics and superimposes on the composite signal. Instead of signal + noise, only noise can be simulated as a test data set. The types of noise distributions supported are uniform, Gaussian, log-normal, Weibull, and exponential.

SIGAN stores all the discrete sample information in ASCII files in current directory. The files can be printed within the DOS environment. Some of these data files become input/ output files for other modules of SIGAN program. For example, user can specify a data file as input time domain data to run the SPECTRUM (FFT) part of SIGAN program. Then SIGAN analyzes this time domain data based on the user selected FFT size, and stores spectral data in a different file. This spectral data file can be displayed with graphics automatically after spectrum computation, or can be invoked by a different module of SIGAN program to read the ASCII file and display with graphics.

One of the significant features of SIGAN is its integrated system capability. With this an overall digital signal processing system can be evaluated. For example, a complex time domain signal + noise can be synthesized using SIGAN to behave as a test data for a digital filter which is designed using a filter design module of SIGAN program and the output of the digital filter can be obtained with SIGAN time domain analysis (convolution) capability. The convoluted output of the digital filter can be displayed in time domain or with the SPECTRUM in frequency domain. This way, one can visualize the spectral characteristics of digital filter and its band rejection capability for a given test data.

II Getting Started

Contents of SIGAN diskettes:

Digital Signal Processing Files:

OPTFIL.EXE
WINFIL.EXE
SPECTRUM.EXE
IIR1FIL.EXE
IIR2FIL.EXE
CONVOL.EXE
SIGSIM.EXE
NOISIM.EXE

Graphics File:

GRAPH.EXE

System Requirements

Hardware Requirements

SIGAN runs on any IBM PC, XT, AT, or 100% compatible computer using DOS 2.1 or higher. Make sure your system has at least

512K RAM
A high density disk drive (1.2 Mb or 1.44 Mb) or 1 Mb Hard disk space.
Math co-processor is recommended but not required.
Requires either Hercules, EGA, or VGA graphics adapter.

Disks:

SIGAN contains one 3 1/2" diskette or two 5 1/4" diskettes.

Note:

For your protection, place a write-protect tab to the master disk(s) to prevent accidental erasure of the SIGAN files.

Installation

Installing SIGAN on a Floppy Disk System

SIGAN is not copy-protected. Before you begin using SIGAN, make a backup copy of the master disk(s). Please refer to license agreement for backup and copying policies.

If you have a 5 1/4" 360KB floppy drive, you will need 2 blank, formatted DS/DD diskettes.

Place the master disk in drive A: and a blank, formatted disk in drive B:. Type:

A> copy A:*.* B: and press the <ENTER> key.

All SIGAN files will be copied to the disk in drive B:. Remove the master disk and store it in a safe place.

If you have a second master disk in your package, repeat the above procedure.

Installing SIGAN on a Hard Disk System

If you have a hard disk system, always run SIGAN from the hard disk. You will want to create a special sub-directory for SIGAN files.

To create a sub-directory for SIGAN files: First, make sure you are at the DOS root directory by typing:

C> cd\ and press the <ENTER> key.

Next, type:

C> md sigan   and press the <ENTER> key.

Change to the SIGAN directory by typing:

C> cd sigan   and press the <ENTER> key.

Place the Master SIGAN disk in drive A: and type:

C> copy A:*.*   and press the <ENTER> key.

SIGAN files will be copied to the hard disk sub-directory. Remove the Master disk and store it in a safe place.

Repeat the above copy procedure if you have one more master disk in your package.

III An Overview of sigan program

In this chapter, an overview of SIGAN program is described. With SIGAN, a variety of digital signal processing (DSP) analysis, design, and synthesis tasks can be accomplished. You do not have to be expert on any of the DSP algorithms to use SIGAN. However, some basic understanding of discrete time sequences, convolution, spectrum analysis (FFT), Nyquist sampling theorem, and normalized frequencies is necessary. You may refer to any text book on DSP as outlined in Appendix B to comprehend these concepts.

You do have to follow a certain sequence in executing various modules of SIGAN. For example, if you attempt to run SPECTRUM (FFT) module with out SIGNAL.DAT or NOISE.DAT file in the current directory, the program aborts. Therefore, the current directory should have SIGNAL.DAT or NOISE.DAT file. These data files are created while executing SIGSIM or NOISIM modules. Similarly, if you try to run CONVOL module to do convolution with out FILTER.DAT file in the current directory, the program aborts. FILTER.DAT file which consists of FIR filter coefficients and SIGNAL.DAT file is necessary for convolution. Therefore, you should execute either OPTFIL or WINFIL module prior to running CONVOL module. There are no prerequisites for running IIR1FIL and IIR2FIL modules to design IIR filters. Similarly, you do not need any input data for running SIGSIM and NOISIM programs. And, finally GRAPH module can be invoked to display several graphs provided there are corresponding graphics data files in the current directory. For a summary of various executable modules of SIGAN, refer to Appendix A.

There are two options for running SIGAN modules. With the first option, you may simply run SIGAN, which in turn invokes SIGAN modules based on your selection. For first time users, this option is recommended. For example, to execute SIGAN, enter at the DOS prompt:

```
     > sigan    <CR>
```

You will first see logo. Press any key to continue. Then the following menu will appear on the screen.

```
Please enter one of the following options....

     1 -- FIR Filter Design (Remez Exchange method)
     2 -- FIR Filter Design (Window method)
     3 -- IIR Filter Design (Butterworth)
     4 -- IIR Filter Design (Chebyshev)
     5 -- Spectrum Analysis (FFT)
     6 -- Time-Domain Analysis (Convolution)
```

7 -- Signal Simulation

```
     8 -- Noise Simulation
     9 -- Graphics
     0 -- Quit the program

Enter 1,2,3,4,5,6,7,8,9, or 0 ......
```

Now, you may select an appropriate module of SIGAN.

With the second option, you may invoke any of the SIGAN modules directly by entering the module name at the DOS prompt. The following are the executable modules of SIGAN, corresponding to above options 1 to 9.

```
     OPTFIL
     WINFIL
     IIR1FIL
     IIR2FIL
     SPECTRUM
     CONVOL
     SIGSIM
     NOISIM
     GRAPH
```

Please note that throughout this manual, details are given only for the second option.

Next chapter covers more details on OPTFIL, WINFIL, IIR1FIL, and IIR2FIL modules. SIGSIM and NOISIM modules are explained in chapter V. Chapter VI deals with SPECTRUM and CONVOL modules. And finally, chapter VII provides more information on GRAPH module of SIGAN.

IV Designing Digital Filters

Digital filters have been widely replacing analog filters in various applications from home electronics to sophisticated electronic warfare systems. The main advantages of digital filters over analog filters are reliability, adaptability of frequency response, and versatility. Cost of digital filters is also becoming lower with rapid advances in digital integrated circuits. Nowadays, the entire digital function such as a filter can be implemented on a single integrated circuit (IC).

Design of digital filters involve the selection and optimization of filter coefficients to satisfy a given filter frequency response specification. Analog filter implementations for lowpass, highpass, bandpass, and bandstop filter functions will require resistor, capacitor, and inductor as circuit elements. In digital systems, incoming analog signal must be sampled to obtain a discrete-time signal. The sampling rate should be at least twice the maximum frequency of the incoming analog signal (Nyquist sampling theorem). One half of the sampling frequency is also called Nyquist frequency. The inverse of the sampling frequency is called the sampling period, which is the time between adjacent samples.

Filters can be classified as recursive or non-recursive. In recursive digital filters, there will be a feedback from outputs of digital filters to the inputs. It is also convenient to design recursive filters as Infinite-Impulse-Response (IIR) filters [Oppenheim]. On the other hand, if there is no feedback from the output of a digital filter to the input, the filter is said to be non-recursive. Non-recursive digital filters are also called as Finite-Impulse-Response (FIR) digital filters. FIR filters provide linear phase versus frequency response and can be implemented using fast convolution techniques for improving speed. The main advantage of an IIR filter is its steep frequency response for a given filter order. Typically, to get a similar frequency response, a FIR filter with an order equal to 10 times the order of the IIR filter is required.

The output of a FIR filter y(n) can be expressed as

$$y(n) = x(n)*h(0) + x(n-1)*h(1) + \ldots\ldots + x(n-N+1)*h(N-1)$$

where h(0),h(1),...., h(N-1) are filter coefficients and x(n), x(n-1), ..., x(n-N+1) are incoming signal samples delayed by the corresponding number of sampling times. The

multiply-accumulate operation provides the output (filtered sample) at time index n.

The output of an IIR filter y(n) can be expressed as

```
  y(n) = x(n)*a(0) + x(n-1)*a(1) + .............. +
               y(n-1)*b(1) + y(n-2)*b(2) + ...........
```

where a(0),a(1),..... and b(1),b(2),...... are feed-forward and feedback coefficients respectively. Obviously, more computations are needed for IIR filter of the same order. However, for the same filter order, IIR filters provide excellent frequency response. One of the disadvantages of an IIR filter is its instability due to feedback. The effect of this is to yield some noisy signal at the filter output even if the input signal is zero.

USING OPTFIL PROGRAM

Design of optimal Finite-Impulse-Response (FIR) digital filters can be done using OPTFIL program. Optimal digital filters can be designed using linear programming concepts [Rabiner] for high performance multiple band and highly selective digital filters. For theoretical understanding of this subject, refer to any Digital Signal Processing book. Some digital signal processing references are outlined in Appendix B.

With OPTFIL program, you will be able to design multiple-band digital filters, including simple lowpass, highpass, bandpass, and bandstop filters. Lowpass, Highpass, bandpass, and bandstop digital filters can also be designed using Window Method as explained in Using Window Program section.

The major advantage of OPTFIL program is that it can design a multiple band filter, for example a filter with two pass bands and two stop bands. Data requirements for this type of filter will become clear when we consider some examples.

How to run OPTFIL

At the DOS prompt, enter the following

```
     >OPTFIL
```

The program will be loaded into the memory and you will see a heading. Press any key to continue.

You should respond to all the prompts of the program. If your responses are not of predicted format, the system may crash and you may lose your data. Some responses do not require <CR> as will become clear when we go to the next section.

Data Entry for OPTFIL program

You should enter either 1,2,3, or 4 depending on whether you like to view Magnitude versus Frequency Plot, Phase versus Frequency Plot, Both Magnitude and Phase versus Frequency Plots, or No Frequency Plot.

Suppose if you enter either 1 or 3, the program prompts you for the type of magnitude versus frequency plot. Here you have a choice to select Linear Magnitude versus Frequency, Logarithm Magnitude versus Frequency, or both types of Magnitude plots.

The next prompt asks you whether you like to log the design parameters and filter coefficients in a file for future reference or for printing. It is a good idea to select this option by entering 1. The program creates a file OPTFIL.LOG in your current directory.

At the next prompt, you should enter 1 to design multiple band filter. You may enter 0 to quit the program. Following this, a filter order number should be entered. The lowest order is 3. As you increase the filter order, optimization of filter coefficients takes longer time. Typically, you should enter an integer between 10 and 50, for a reasonably good filter response.

Conventional filter design methods deal with one or two pass bands and one or two stop bands. There are certain rules about how you select the sequence in which these bands appear. For example in a bandpass filter, you will have a stop band followed by a pass band and one more stop band. This is how you will design filters using WINFIL program.

But in OPTFIL program, you should first enter the total number of bands. The maximum number of bands will be 5. For example you can design two bandpass filters separated by a stop band, all in one design. This is achieved by selecting bands in this sequence: stopband-passband-stopband-passband-stopband.

Unlike WINFIL program, where you will have a choice to select either a lowpass, highpass, bandpass, or a bandstop filter, here you should enter frequency band edges in normalized fractions. The maximum value of band edge is 0.5,

corresponding to one-half the sampling frequency. The minimum value is 0.0, corresponding to DC.

Suppose you want to design a filter with two bands. You should enter 4 values for band edges, because you will need lower and upper values for pass band and lower and upper values for stop band. You can not use upper band edge of a pass band as the same value of the lower band edge of stop band, because of non-ideal characteristics of realizable filters. If you enter the same value, the program optimization may not converge and the system may crash.

Let us assume we like to have filter the following edges.

```
     0.0
     0.2
     0.25
     0.5
```

Note that the first and last values of the band edges should always be 0.0 (DC) and 0.5 (Nyquist Frequency), respectively.

By entering normalized fractions instead of absolute frequency values, you can use the same filter coefficient set for designing at a different frequency band. This is achieved by maintaining a constant ratios between pass band and stop band frequencies and sampling frequency. For example, if you like to design a low pass filter with 2 KHz as pass band edge, 2.5 KHz as stop band edge, 10 KHz as sampling frequency, and 5 KHz as one-half the sampling frequency, corresponding normalized fractions shall be as follows.

```
     0.0/10.0  ----> 0.0
     2.0/10.0  ----> 0.2
     2.5/10.0  ----> 0.25
     5.0/10.0  ----> 0.5
```

The same filter coefficient set can be used at a different frequency band with parameters shown below.

pass band frequency edge	: 1.00 MHz
stop band frequency edge	: 1.25 MHz
sampling frequency	: 5.00 MHz
one-half sampling frequency	: 2.50 MHz

For above frequency values, the normalized fractions can be calculated as 0.0, 0.2, 0.25, and 0.5.

Next, the program prompts for amplitude factors for all the bands. Typically, these values are either 1 or 0. If you enter 1 for band 1, and 0 for band 2, it is a lowpass characteristic or else, if you enter 0 followed by 1, it is a highpass characteristic. Let us enter 1.0 followed by 0.0 for this example.

Finally, you should enter weighting factors for all bands. Typically, you will enter 1 for all pass bands, and enter a value between 10.0 and 100.0 for all stop bands. The higher the value, the higher the attenuation of the corresponding stop band. Let us enter 1.0 followed by 20.0 for this example.

Now you have entered all the required parameters. The program displays Design Number and displays "Program Running". The amount of time for execution depends on the order of filter you have selected. Typically, for a 20th order filter on an IBM XT type machine with 8087 math-co-processor, it will take about 30 seconds.

If you have selected Plot option, you will see graphs on your CRT screen. Press any key to continue. You will see the menu again to select a type of filter for another design. You need not have to enter the type of frequency plots or whether you want to log the output in a file for subsequent designs.

Four design examples are given for lowpass, highpass, bandpass, and bandstop filters. Some graphs are shown for magnitude versus frequency and logarithm-magnitude versus frequency.

Numerical Entries for a Lowpass Filter Design Example

```
Type of Filter:
         Multiple Bandpass/bandstop Filter


Number of Filter coefficients:
            16

Normalized Band frequencies:
             .0000
             .2000
             .3000
             .5000

Amplitude Factors:
            1.0000
             .0000
```

Weighting Factors:

```
            1.0000
          100.0000

Filter Coefficients:
             .9131418000E-02
             .4238094000E-01
             .6548731000E-01
             .1533496000E-01
            -.7906209000E-01
            -.5528451000E-01
             .1721322000E+00
             .4202265000E+00
             .4202265000E+00
             .1721322000E+00
            -.5528451000E-01
            -.7906209000E-01
             .1533496000E-01
             .6548731000E-01
             .4238094000E-01
             .9131418000E-02


Numerical Entries for a Highpass Filter Design Example


Type of Filter:
         Multiple Bandpass/bandstop Filter

Number of Filter coefficients:
            23

Normalized Band frequencies:
             .0000
             .1000
             .3000
             .5000

Amplitude Factors:
             .0000
            1.0000

Weighting Factors:
           40.0000
            1.0000

Filter Coefficients:
            -.1311560000E-03
            -.1421681000E-02
             .1199367000E-02
             .6837445000E-02
             .6737543000E-04
```

```
-.2062974000E-01
```

```
             -.1400813000E-01
              .4278709000E-01
              .6533150000E-01
             -.6496078000E-01
             -.3022962000E+00
              .5744332000E+00
             -.3022962000E+00
             -.6496078000E-01
              .6533150000E-01
              .4278709000E-01
             -.1400813000E-01
             -.2062974000E-01
              .6737543000E-04
              .6837445000E-02
              .1199367000E-02
             -.1421681000E-02
             -.1311560000E-03
```

Numerical Entries for a Bandpass Filter Design Example

```
Type of Filter:
         Multiple Bandpass/bandstop Filter

Number of Filter coefficients:
            26

Normalized Band frequencies:
             .0000
             .1000
             .2000
             .3000
             .4000
             .5000

Amplitude Factors:
             .0000
            1.0000
             .0000

Weighting Factors:
           20.0000
            1.0000
           40.0000

Filter Coefficients:
            -.2559202000E-03
            -.3858224000E-02
             .8892963000E-02
             .1727045000E-01
            -.2580966000E-01
```

```
-.2714020000E-01
```

```
             .1966726000E-01
            -.6939387000E-02
             .4990217000E-01
             .1054282000E+00
            -.1678277000E+00
            -.2164198000E+00
             .2468225000E+00
             .2468225000E+00
            -.2164198000E+00
            -.1678277000E+00
             .1054282000E+00
             .4990217000E-01
            -.6939387000E-02
             .1966726000E-01
            -.2714020000E-01
            -.2580966000E-01
             .1727045000E-01
             .8892963000E-02
            -.3858224000E-02
            -.2559202000E-03
```

Numerical Entries for a Bandstop Filter Design Example

```
Type of Filter:
        Multiple Bandpass/bandstop Filter

Number of Filter coefficients:
            32

Normalized Band frequencies:
             .0000
             .1000
             .1500
             .3000
             .3500
             .5000

Amplitude Factors:
            1.0000
             .0000
            1.0000

Weighting Factors:
            1.0000
          100.0000
            1.0000

Filter Coefficients:
            -.5129008000E-01
             .2889319000E-01
```

```
-.1355764000E+00
```

```
 .4849811000E-01
-.1463406000E+00
 .2662247000E-01
-.8959413000E-01
-.8932173000E-02
-.3724927000E-01
-.2126408000E-01
-.4140139000E-01
 .1213274000E+00
-.1101784000E+00
 .3479352000E+00
-.1803756000E-01
 .2773846000E+00
 .2773846000E+00
-.1803756000E-01
 .3479352000E+00
-.1101784000E+00
 .1213274000E+00
-.4140139000E-01
-.2126408000E-01
-.3724927000E-01
-.8932173000E-02
-.8959413000E-01
 .2662247000E-01
-.1463406000E+00
 .4849811000E-01
-.1355764000E+00
 .2889319000E-01
-.5129008000E-01
```

USING WINFIL PROGRAM

In this section, digital filter design using Window method is discussed. For technical details about various window based designs, refer to any Digital Signal Processing book.

Design of simple digital filters, such as lowpass, highpass, bandpass, and bandstop filters can be performed using WINFIL program. With OPTFIL program, you can design these filters to any arbitrary frequency response specification. You may refer to the previous section for actual design with OPTFIL program.

How to Run WINFIL

Unlike OPTFIL program, the data entry requirements for this program are simple.

At the DOS prompt, enter the following.

```
>WINFIL
```

Data Entry for WINFIL Program

You should enter either 1,2,3, or 4 depending on whether you like to view Magnitude versus Frequency Plot, Phase versus Frequency Plot, Both Magnitude and Phase versus Frequency Plots, or No Frequency Plot.

Suppose if you enter either 1 or 3, the program prompts you for the type of magnitude versus frequency plot. Here you have a choice to select Linear Magnitude versus Frequency, Logarithm Magnitude versus Frequency, or both types of Magnitude plots.

The next prompt asks you whether you like to log the design parameters and filter coefficients in a file for future reference or for printing. It is a good idea to select this option by entering 1. The program creates a file WINFIL.LOG in your current directory.

Next, you will select a type of digital filter by entering 1,2,3, or 4 corresponding to Lowpass, Highpass, Bandpass, or Bandstop filter design. By entering 0, you can quit the program.

Once you have made a choice of filter, then the next step is to select the type of window for the filter design. Here you have a choice of 7 windows. Enter 1,2,3,4,5,6, or 7 corresponding to Rectangular, Triangular, Hamming, Generalized Hamming, Hann, kaiser, or Chebyshev window designs.

Next you should enter a number for filter coefficients. This number is typically between 3 and 128. Depending on the type of filter you have selected, you should enter cutoff band edges as normalized fractions. For more details on normalized fractions, refer to Data Entry for OPTFIL Program section. For lowpass or highpass filter designs, you need to enter one cutoff band edge. For bandpass or bandstop filters, you should enter both lower and upper cutoff band edges.

Generalized Hamming window design requires in addition to the above data, a constant between 0.0 and 1.0. Similarly, for Kaiser and Chebyshev designs, you should enter stop band attenuation in dB (20 to 60 dB range). Chebyshev design also requires transition bandwidth typically 0.01 to 0.1 (normalized with respect to Nyquist Frequency).

Four design examples are given for lowpass, highpass, bandpass, and bandstop filters. Some graphs are shown for

magnitude versus frequency and logarithm-magnitude versus frequency.

Numerical Entries for a Lowpass Filter Design Example

```
Type of Filter:
   Lowpass Filter

Type of Window:
   Hamming Window

Cutoff Frequency:
             0.15

Number of Filter Coefficients:
             20

Filter Coefficients:
              .1216922000E-02
              .3880845000E-02
              .5311730000E-02
             -.2209371000E-02
             -.2202288000E-01
             -.3642808000E-01
             -.1031150000E-01
              .7666636000E-01
              .1979737000E+00
              .2872061000E+00
              .2872061000E+00
              .1979737000E+00
              .7666636000E-01
             -.1031150000E-01
             -.3642808000E-01
             -.2202288000E-01
             -.2209371000E-02
              .5311730000E-02
              .3880845000E-02
              .1216922000E-02
```

Numerical Entries for a Highpass Filter Design Example

```
Type of Filter:
   Highpass Filter

Type of Window:
   Hann Window

Cutoff Frequency:
             0.25

Number of Filter Coefficients:
```

```
            15

Filter Coefficients:
             .1730705000E-02
             .1852944000E-09
            -.1964979000E-01
            -.6956883000E-08
             .7335363000E-01
             .1187614000E-07
            -.3061949000E+00
             .5000000000E+00
            -.3061949000E+00
             .1187614000E-07
             .7335363000E-01
            -.6956883000E-08
            -.1964979000E-01
             .1852944000E-09
             .1730705000E-02


Numerical Entries for a Bandpass Filter Design Example

Type of Filter:
   Bandpass Filter

Type of Window:
   kaiser Window

Lower Cutoff Frequency:
             0.15

Upper Cutoff Frequency
             0.30

Attenuation in dB:
            30.0


Number of Filter Coefficients:
            20

Filter Coefficients:
            -.1702662000E-01
            -.2417747000E-01
             .7358778000E-02
            -.5094771000E-02
             .3628966000E-02
             .1008058000E+00
             .3810059000E-01
            -.2060948000E+00
            -.1413037000E+00
```

```
.2255410000E+00
```

```
             .2255410000E+00
            -.1413037000E+00
            -.2060948000E+00
             .3810059000E-01
             .1008058000E+00
             .3628966000E-02
            -.5094771000E-02
             .7358778000E-02
            -.2417747000E-01
            -.1702662000E-01
```

Numerical Entries for a Bandstop Filter Design Example

```
Type of Filter:
  Bandstop Filter

Type of Window:
   Chebyshev Window

Lower Cutoff Frequency:
             0.18

Upper Cutoff Frequency:
             0.32

Ripple in dB:
            40.0

Transition Band width:
             0.08

Number of Filter Coefficients:
            29

Filter Coefficients:
             -.1598363000E-03
             -.8760620000E-09
              .3857094000E-02
              .1040236000E-07
             -.1236784000E-01
             -.5262397000E-09
              .1109917000E-01
             -.1825648000E-08
              .3011758000E-01
              .2419775000E-07
             -.1241769000E+00
             -.2155147000E-08
              .2316956000E+00
              .1168151000E-07
              .7200000000E+00
```

```
.1168151000E-07
```

```
                  .2316956000E+00
                 -.2155147000E-08
                 -.1241769000E+00
                  .2419775000E-07
                  .3011758000E-01
                 -.1825648000E-08
                  .1109917000E-01
                 -.5262397000E-09
                 -.1236784000E-01
                  .1040236000E-07
                  .3857094000E-02
                 -.8760620000E-09
                 -.1598363000E-03
```

USING IIR1FIL PROGRAM

The major advantage of IIR (Infinite Impulse Response) digital filters, compared to FIR (Finite Impulse Response) digital filters is that for a given filter order N, highly selective recursive digital filters can be designed.

Butterworth filters have a monotonically decreasing response with respect to frequency. For theoretical understanding of this subject, refer to [Antoniou].

With IIR1FIL module, you will be able to design lowpass, highpass, bandpass, and bandstop filters. The major advantage of IIR1FIL over OPTFIL is simplicity. Orders of IIR filters are usually low compared to FIR filters. Typically, orders less than 10 are selected. Data requirements for this type of filter will become clear when we consider some examples.

How to run IIR1FIL Program

At DOS prompt, enter the following

```
     >IIR1FIL
```

The program will be loaded in to the memory and you will see a heading. Press any key to continue.

You should respond to all the prompts of the program. If your responses are not of predicted format, the system may crash and you may lose your data. Some responses do not require <CR> as will become clear when we go to the next section.

Data Entry for IIR1FIL Program

You should enter either 1,2,3,4 or 0 to select design of lowpass, highpass, bandpass, and bandstop filters or quit this module.

Suppose if you enter 1 for designing a lowpass filter, the program prompts you for stopband cutoff frequency in radians/second. Unlike OPTFIL or WINFIL, data entry for IIR filter design requires actual frequency edges for passband and stopband. Therefore, you should enter stopband cutoff frequency in radians/second. Similarly, you should enter passband cutoff frequency in radians/second. The stopband attenuation and passband ripple must be entered in dB. The sampling frequency in radians/second must be greater than twice the highest cutoff frequency (Nyquist theorem).

Four design examples of Butterworth filters are given for lowpass, highpass, bandpass, and bandstop responses. Number of second order filter sections for each design depend on input parameters. For a sharp frequency response, the filter order has to be higher. Several second order sections can be cascaded to realize an IIR filter.

Examples:

```
 Butterworth IIR Filter Design:
 ==============================

 ****** FILTER DESIGN NUMBER  1 ******

 Stopband Cutoff Frequency:        400.0000000 rad/sec
 Passband Cutoff Frequency:        300.0000000 rad/sec
 Sampling Frequency       :       1000.0000000 rad/sec
 Stopband Attenuation     :         25.0000000 dB
 Passband Ripple          :          1.0000000 dB

 LOW_PASS FILTER COEFFICIENTS

 Filter Section Number:  1

 A0( 1) =   .1219794E-02
 A1( 1) =   .1219794E-02
 A2( 1) =   .0000000E+00
 B0( 1) =   .2234552E+00
 B1( 1) =   .1000000E+01
 B2( 1) =   .0000000E+00

 Filter Section Number:  2

 A0( 2) =   .1636352E-05
 A1( 2) =   .3272704E-05
 A2( 2) =   .1636352E-05
```

```
B0( 2) =   .1546896E+00
```

```
B1( 2) =   .4915010E+00
B2( 2) =   .1000000E+01

Filter Section Number:  3

A0( 3) =   .2214924E-05
A1( 3) =   .4429849E-05
A2( 3) =   .2214924E-05
B0( 3) =   .5629581E+00
B1( 3) =   .6652831E+00
B2( 3) =   .1000000E+01

****** FILTER DESIGN NUMBER  2 ******

Stopband Cutoff Frequency:       2000.0000000 rad/sec
Passband Cutoff Frequency:       2400.0000000 rad/sec
Sampling Frequency       :       5000.0000000 rad/sec
Stopband Attenuation     :         30.0000000 dB
Passband Ripple          :          2.0000000 dB

HIGH_PASS FILTER COEFFICIENTS

Filter Section Number:  1

A0( 1) =  -.6436902E-01
A1( 1) =   .6436902E-01
A2( 1) =   .0000000E+00
B0( 1) =   .8712620E+00
B1( 1) =   .1000000E+01
B2( 1) =   .0000000E+00

Filter Section Number:  2

A0( 2) =   .4408900E-02
A1( 2) =  -.8817800E-02
A2( 2) =   .4408900E-02
B0( 2) =   .8718295E+00
B1( 2) =   .1854194E+01
B2( 2) =   .1000000E+01

****** FILTER DESIGN NUMBER  3 ******

Low Stopband Cutoff Frequency :       1000.0000000 rad/sec
High Stopband Cutoff Frequency:       4000.0000000 rad/sec
Low Passband Cutoff Frequency :       2000.0000000 rad/sec
High Passband Cutoff Frequency:       3000.0000000 rad/sec
Sampling Frequency            :      10000.0000000 rad/sec
Stopband Attenuation          :         35.0000000 dB
Passband Ripple               :          1.0000000 dB

BAND_PASS FILTER COEEFICIENTS
```

```
Filter Section Number:  1

A0( 1) =  -.1347425E-03
A1( 1) =   .0000000E+00
A2( 1) =   .1347425E-03
B0( 1) =   .4849488E+00
B1( 1) =  -.2306461E+00
B2( 1) =   .1000000E+01

Filter Section Number:  2

A0( 2) =  -.9851272E-04
A1( 2) =   .0000000E+00
A2( 2) =   .9851272E-04
B0( 2) =   .4849490E+00
B1( 2) =   .2306461E+00
B2( 2) =   .1000000E+01

Filter Section Number:  3

A0( 3) =  -.1858947E-03
A1( 3) =   .0000000E+00
A2( 3) =   .1858947E-03
B0( 3) =   .7692499E+00
B1( 3) =  -.5976344E+00
B2( 3) =   .1000000E+01

Filter Section Number:  4

A0( 4) =  -.9201846E-04
A1( 4) =   .0000000E+00
A2( 4) =   .9201846E-04
B0( 4) =   .7692499E+00
B1( 4) =   .5976345E+00
B2( 4) =   .1000000E+01

****** FILTER DESIGN NUMBER  4 ******

Low Stopband Cutoff Frequency :        600.0000000 rad/sec
High Stopband Cutoff Frequency:       1000.0000000 rad/sec
Low Passband Cutoff Frequency :        400.0000000 rad/sec
High Passband Cutoff Frequency:       1200.0000000 rad/sec
Sampling Frequency            :       3000.0000000 rad/sec
Stopband Attenuation          :         28.0000000 dB
Passband Ripple               :          1.5000000 dB

BAND_STOP FILTER COEFFICIENTS

Filter Section Number:  1

A0( 1) =   .4959078E+00
```

```
A1( 1) =   .1549369E+00
```

```
 A2( 1) =   .4959078E+00
 B0( 1) =  -.8184366E-02
 B1( 1) =   .1549369E+00
 B2( 1) =   .1000000E+01

 Filter Section Number:  2

 A0( 2) =   .1154890E+01
 A1( 2) =   .3608230E+00
 A2( 2) =   .1154890E+01
 B0( 2) =   .2843860E+00
 B1( 2) =  -.6645700E+00
 B2( 2) =   .1000000E+01

 Filter Section Number:  3

 A0( 3) =   .2354213E+00
 A1( 3) =   .7355286E-01
 A2( 3) =   .2354213E+00
 B0( 3) =   .3714634E+00
 B1( 3) =   .9741737E+00
 B2( 3) =   .1000000E+01

 Filter Section Number:  4

 A0( 4) =   .1634024E+01
 A1( 4) =   .5105193E+00
 A2( 4) =   .1634024E+01
 B0( 4) =   .7009149E+00
 B1( 4) =  -.1056614E+01
 B2( 4) =   .1000000E+01

 Filter Section Number:  5

 A0( 5) =   .2299391E+00
 A1( 5) =   .7184004E-01
 A2( 5) =   .2299391E+00
 B0( 5) =   .7531760E+00
 B1( 5) =   .1365138E+01
 B2( 5) =   .1000000E+01
```

USING IIR2FIL PROGRAM

The major advantage of IIR (Infinite Impulse Response) digital filters, compared to FIR (Finite Impulse Response) digital filters is that for a given filter order N, highly selective recursive digital filters can be designed.

Chebyshev filters have a monotonically decreasing response with respect to frequency in the stop band and equi-ripple

amplitude response in the pass band. For theoretical understanding of this subject, refer to [Antoniou].

With IIR2FIL module, you will be able to design lowpass, highpass, bandpass, and bandstop filters. The major advantage of IIR2FIL over OPTFIL is simplicity. Orders of IIR filters are usually low compared to FIR filters. Typically, orders less than 10 are selected. Data requirements for this type of filter will become clear when we consider some examples.

How to run IIR2FIL Program

At DOS prompt, enter the following

```
     >IIR2FIL
```

The program will be loaded in to the memory and you will see a heading. Press any key to continue.

You should respond to all the prompts of the program. If your responses are not of predicted format, the system may crash and you may lose your data. Some responses do not require <CR> as will become clear when we go to the next section.

Data Entry for IIR2FIL Program

You should enter either 1,2,3,4 or 0 to select design of lowpass, highpass, bandpass, and bandstop filters or quit this module.

Suppose if you enter 3 for designing a bandpass filter, the program prompts you for lower and upper stopband and passband frequency edges in radians/second. Unlike OPTFIL or WINFIL, data entry for IIR filter design requires actual frequency edges for passband and stopband. Therefore, you should enter lower and upper stopband cutoff frequencies in radians/second. Similarly, you should enter lower and upper passband cutoff frequencies in radians/second. The stopband attenuation and passband ripple must be entered in dB. The sampling frequency in radians/second must be greater than twice the highest cutoff frequency (Nyquist theorem).

Four design examples of Chebyshev filters are given for lowpass, highpass, bandpass, and bandstop responses. Number of second order filter sections for each design depend on input parameters. For a sharp frequency response, the filter order has to be higher. Several second order sections can be cascaded to realize an IIR filter.

Examples:

```
Chebyshev IIR Filter Design:
============================

****** FILTER DESIGN NUMBER  1 ******

Stopband Cutoff Frequency:       1000.0000000 rad/sec
Passband Cutoff Frequency:        850.0000000 rad/sec
Sampling Frequency       :       2500.0000000 rad/sec
Stopband Attenuation     :         20.0000000 dB
Passband Ripple          :          1.0000000 dB

LOW_PASS FILTER COEFFICIENTS

Filter Section Number:  1

A0( 1) =   .5013161E-06
A1( 1) =   .1002632E-05
A2( 1) =   .5013161E-06
B0( 1) =   .2218830E+00
B1( 1) =  -.4796542E-01
B2( 1) =   .1000000E+01

Filter Section Number:  2

A0( 2) =   .3309369E-06
A1( 2) =   .6618739E-06
A2( 2) =   .3309369E-06
B0( 2) =   .7872339E+00
B1( 2) =   .9489608E+00
B2( 2) =   .1000000E+01

****** FILTER DESIGN NUMBER  2 ******

Stopband Cutoff Frequency:        700.0000000 rad/sec
Passband Cutoff Frequency:        900.0000000 rad/sec
Sampling Frequency       :       2000.0000000 rad/sec
Stopband Attenuation     :         25.0000000 dB
Passband Ripple          :          2.0000000 dB

HIGH_PASS FILTER COEFFICIENTS

Filter Section Number:  1

A0( 1) =  -.1496410E+00
A1( 1) =   .1496410E+00
A2( 1) =   .0000000E+00
B0( 1) =   .8895916E+00
B1( 1) =   .1000000E+01
B2( 1) =   .0000000E+00

Filter Section Number:  2
```

```
A0( 2) =   .2321330E-01
A1( 2) =  -.4642660E-01
A2( 2) =   .2321330E-01
B0( 2) =   .8918627E+00
B1( 2) =   .1809586E+01
B2( 2) =   .1000000E+01

****** FILTER DESIGN NUMBER  3 ******

Low Stopband Cutoff Frequency :        100.0000000 rad/sec
High Stopband Cutoff Frequency:        500.0000000 rad/sec
Low Passband Cutoff Frequency :        150.0000000 rad/sec
High Passband Cutoff Frequency:        300.0000000 rad/sec
Sampling Frequency            :       2000.0000000 rad/sec
Stopband Attenuation          :         30.0000000 dB
Passband Ripple               :          1.5000000 dB

BAND_PASS FILTER COEEFICIENTS

Filter Section Number:  1

A0( 1) =  -.1359556E-02
A1( 1) =   .0000000E+00
A2( 1) =   .1359556E-02
B0( 1) =   .8864421E+00
B1( 1) =  -.1575640E+01
B2( 1) =   .1000000E+01

Filter Section Number:  2

A0( 2) =  -.1250562E-02
A1( 2) =   .0000000E+00
A2( 2) =   .1250562E-02
B0( 2) =   .8576697E+00
B1( 2) =  -.1326862E+01
B2( 2) =   .1000000E+01

Filter Section Number:  3

A0( 3) =  -.1454657E-02
A1( 3) =   .0000000E+00
A2( 3) =   .1454657E-02
B0( 3) =   .9613284E+00
B1( 3) =  -.1742924E+01
B2( 3) =   .1000000E+01

Filter Section Number:  4

A0( 4) =  -.1210995E-02
A1( 4) =   .0000000E+00
A2( 4) =   .1210995E-02
```

```
B0( 4) =    .9332067E+00
```

```
B1( 4) =  -.1150567E+01
B2( 4) =   .1000000E+01

****** FILTER DESIGN NUMBER  4 ******

Low Stopband Cutoff Frequency :      10000.0000000 rad/sec
High Stopband Cutoff Frequency:      14000.0000000 rad/sec
Low Passband Cutoff Frequency :       8000.0000000 rad/sec
High Passband Cutoff Frequency:      16000.0000000 rad/sec
Sampling Frequency            :      40000.0000000 rad/sec
Stopband Attenuation          :         40.0000000 dB
Passband Ripple               :          1.0000000 dB

BAND_STOP FILTER COEFFICIENTS

Filter Section Number:  1

A0( 1) =   .2849243E+00
A1( 1) =   .2176629E+00
A2( 1) =   .2849243E+00
B0( 1) =  -.4301513E+00
B1( 1) =   .2176629E+00
B2( 1) =   .1000000E+01

Filter Section Number:  2

A0( 2) =   .2004279E+01
A1( 2) =   .1531133E+01
A2( 2) =   .2004279E+01
B0( 2) =   .6176122E+00
B1( 2) =  -.8598118E+00
B2( 2) =   .1000000E+01

Filter Section Number:  3

A0( 3) =   .1650818E+00
A1( 3) =   .1261113E+00
A2( 3) =   .1650818E+00
B0( 3) =   .7697626E+00
B1( 3) =   .1565710E+01
B2( 3) =   .1000000E+01

Filter Section Number:  4

A0( 4) =   .2014136E+01
A1( 4) =   .1538663E+01
A2( 4) =   .2014136E+01
B0( 4) =   .8991532E+00
B1( 4) =  -.5904548E+00
B2( 4) =   .1000000E+01
```

Filter Section Number: 5

```
A0( 5) =   .2980795E+00
A1( 5) =   .2277125E+00
A2( 5) =   .2980795E+00
B0( 5) =   .9365134E+00
B1( 5) =   .1568067E+01
B2( 5) =   .1000000E+01
```

V SIGNAL SYNTHESIS

One of the important requirements of characterizing digital signal processing systems such as digital filters or FFT processors is to have test data which closely represents real data. Therefore, signal synthesis is an important task of generating discrete-time data.

SIGAN provides discrete-time signal synthesis capability. User initially selects a number of samples. Multiple sinusoidal signals of different amplitudes, frequencies, and phases can be generated and combined to produce a composite test signal. As per Nyquist theorem, if sampling frequency is greater than twice the maximum frequency of the signal, the signal can be completely reconstructed from discrete-time samples. Therefore, sampling frequency will be set at twice the maximum frequency of the signal. In addition, user selects an over- sampling factor for better graphics display. This factor is typically between 10.0 and 30.0. Note that when you enter a large factor, samples stored in in SIGNAL.DAT are closely spaced. Hence, you can see a few cycles of the signal cleary on the screen. But spectrum analysis on such a signal set will have spectral lines closely spaced in near DC. Therefore, there is a tradeoff between a good signal graph and a good spectrum graph.

Noise is an important part of realistic test data. Gaussian noise closely represents noise arising in physical systems. Noise of certain amplitude, mean, and variance can be added to the composite signal to simulate any signal-to-noise ratio in the system. If you want to simulate different types of noise, you may refer to Noise Simulation section.

Signal Simulation

Discrete-time signals can be synthesized for various amplitudes, frequencies, phase shifts, and noise distributions. SIGSIM program will synthesize composite waveforms using multiple sinusoids. The user inputs the number of sinusoidal signals to be synthesized, their frequencies and amplitudes. The synthesized signal + noise samples can be stored in a file for later processing such as SPECTRUM, Convolution, and Graphics.

Complex waveforms can be synthesized by selecting multiple frequencies and amplitudes. For example, a square wave can be synthesized, if the multiple frequencies are harmonically related with appropriate relation between their amplitudes.

Composite signal/noise can be displayed as a graph if user selects this option. The signal/noise samples can be stored in SIGNAL.DAT file. The data generated with this module is useful for interfacing with SPECTRUM (FFT) and time-domain convolution (CONVOL) modules of SIGAN program. For example, SPECTRUM can read the discrete-time samples stored in SIGNAL.DAT file, perform spectrum analysis, and create SPECTRUM.DAT and FFT.DAT files.

If you select noise option while running SIGSIM, Gaussian noise will be generated for user selectable mean, variance, and noise scaling factors and added to signal samples. You will have an option to normalize the composite signal/noise samples in the range -1.0 to +1.0.

With SIGSIM, you may generate up to 1024 samples for each run. If you need more than 1024 samples of signal, you should run SIGSIM separately and rename SIGNAL.DAT generated for each run as a different file name. After sufficient number of runs, merge these data files at the DOS environment to get a composite file and rename it as SIGNAL.DAT.

Example:

```
Number of frequencies simulated: 4

     (1) frequency : 3000.0 Hz
         Amplitude : 1.0
         Phase     : 90 degrees
     (2) frequency : 6000.0 Hz
         Amplitude : 2.0
         Phase     : 0 degrees
     (3) frequency : 12000.0 Hz
         Amplitude : 0.5
         Phase     : 270 degrees
     (4) frequency : 18000.0 Hz
         Amplitude : 0.25
         Phase     : 180 degrees

Frequency Over-Sampling Factor   :  30.0
Number of Signal Samples        :  600
Normalization                   :  NO
```

Noise Simulation

If applications require only noise data without any sinusoidal frequencies, then NOISIM program can be used to generate noise samples with any combinations of above noise distributions and store them in a file called NOISE.DAT.

NOISIM module of SIGAN can be used as a stand-alone program to generate different types of noise. If you want to generate a composite signal with noise, you may refer to the previous chapter. However, with that option, only Gaussian noise can be generated and added to the signal. If you want to generate multiple noises of different types, NOISIM module of SIGAN provides this capability.

User inputs parameters such as number of noise sources, types of noise distributions, means, and variances, and noise type dependent parameters. Noise samples can be generated and combined to provide a composite noise test data. This data is stored in NOISE.DAT file. You will have an option to normalize the composite signal/noise samples in the range -1.0 to +1.0.

The noise data can be viewed graphically if user selects that option. NOISE.DAT can be useful for interfacing with other modules of SIGAN program.

With NOISIM, you may generate up to 1024 samples for each run. If you need more than 1024 samples of noise, you should run NOISIM separately and rename NOISE.DAT generated for each run as a different file name. After sufficient number of runs, merge these data files at the DOS environment to get a composite file and rename it as NOISE.DAT.

Example:

```
Number of Noise Sources Simulated: 3

     (1) Uniform Distribution:
          mean                 : 0.0
          standard deviation   : 5.0
          amplitude factor     : 0.2
     (2) Weibull Distribution:
          mean                 : 10.0
          standard deviation   : 3.0
          amplitude factor     : 0.05
     (3) Exponential Distribution:
          mean                 : 0.0
          amplitude factor     : 0.5

Number of Noise Samples        : 600
Normalization                  :  NO
```

VI SIGNAL ANALYSIS

Discrete-time signals can be analyzed both in time and frequency domains using SIGAN. SPECTRUM (FFT) module of SIGAN can be used to convert time domain data stored in SIGNAL.DAT file to a frequency domain data and store in SPECTRUM.DAT file. The spectrum of a signal can be displayed with GRAPH program.

Sometimes, signal analysis in time domain provide useful information. Convolution is a method of time-domain filtering. For example, an input signal which is stored in SIGNAL.DAT file is convolved with FILTER.DAT which is a data set of filter coefficients, then the output is a filtered signal which is be stored in CONVOL.DAT file.

Digital Spectral Analysis:

In this section, Fast Fourier Transform (FFT) techniques can be applied for digital spectral analysis. Spectrum of a signal/noise can be obtained after transformation of time-domain data to the frequency-domain data. FFT is an algorithm for faster computation of discrete Fourier transform (DFT). DFT it self is an approximation of continuous Fourier transform. DFT is suitable for processing sampled data.

The continuous Fourier transform of a signal x(t) can be expressed as

$$X(f) = x(t)\ e\text{-}jwt$$

where w is radial frequency and X(f) is spectrum of a signal x(t).

The DFT of a sampled time domain sequence x(n) can be expressed as

$$X(k) = \sum_{n=0}^{N-1} x(n)\ e\text{-}j2\pi nk/N$$

where N is the number of samples and X(k) represents spectrum of a sequence x(n).

How to run SPECTRUM program

To invoke the SPECTRUM module, enter at a DOS prompt, the following:

```
     > spectrum    <CR>
```

The SPECTRUM program generates a series of prompts for data input. You have a choice to select either SIGNAL.DAT, NOISE.DAT, SIGNAL.DAT+NOISE.DAT, and CONVOL.DAT as input for spectrum analysis. SPECTRUM module of SIGAN can combine samples of SIGNAL.DAT (due to SIGSIM) and NOISE.DAT (due to NOISIM) and store them in SAMPLE.DAT file and perform spectrum analysis. Also, the values of SAMPLE.DAT can be viewed as a composite test signal with GRAPHICS module. With CONVOL option, you can view the spectrum of a convolved signal.

Next, you will be asked to enter the number of samples. Remember the number you enter should be a highly composite number such as 2,4,8,16,....1024. For real input data (i.e., there is no imaginary data set), the spectrum shall be symmetrical over the center frequency. Therefore, SPECTRUM provides spectral samples of DC + 1/2 of N, where N is the number of FFT points. The other half of the spectrum can be obtained as mirror image of the spectral samples 1 to N/2. Therefore, SPECTRUM computes magnitudes of only one-half spectral samples and store them in SPECTRUM.DAT file.

If we assume that a sequence x(n) is generated by sampling at a rate of fsamp (sampling interval Tsamp = 1/fsamp), then an N point FFT will provide a spectral resolution of fdelta as follows.

$$\text{fdelta} = \frac{1}{\text{N Tsamp}}$$

The SPECTRUM module of SIGAN reads a file for input data and after performing FFT calculations, store the real and imaginary parts of output spectrum as well as its magnitude. The magnitude of the spectrum is stored in SPECTRUM.DAT file. All these data are stored in files in an ASCII readable form. FFT.DAT file contains both real and imaginary parts of spectrum output in each line separated by a space.

The SPECTRUM.DAT file will be used by GRAPH program to display spectrum output, if you have selected such an option. The spectrum can later be displayed without running the SPECTRUM again, by running the graphics program. For more information on these graphics programs, refer to GRAPHICS chapter.

Example:

Signal/Noise Characteristics:

Frequency	: 10 KHz

```
Amplitude                 : 1.0
Phase                     : 0 degrees
Over-Sampling Factor      : 1.5
Gaussian Noise Added      : YES
 Noise Mean               : 0.0
 Noise standard Deviation     : 10.0
Noise Scaling Factor      : 0.5
Number of FFT Points      : 1024
```

Digital Time-series Analysis:

SIGAN provides time domain signal processing technique such as convolution. Convolution is a linear filtering operation. If we represent x(n) as an input sequence and h(n) as an impulse response of a linear system such as a filter, then the system (filter) output is the convolution of x(n) and h(n). Symbolically, this is represented as

$$y(n) = x(n) * h(n)$$

How to run CONVOL Program

To invoke the CONVOL module, enter at a DOS prompt, the following:

```
> convol  <CR>
```

CONVOL program reads information stored in SIGNAL.DAT and FILTER.DAT files and after performing convolution (time-domain filtering) calculations, store the results in CONVOL.DAT file. You should enter the number of signal samples and the number of filter coefficients for convolution. One important requirement is that you should enter the same number for filter coefficients as in FILTER.DAT file. If you don't, you are convolving with a distorted filter impulse response. This is not mandatory for signal samples, because signal sample set is usually larger╨than filter coefficient set. If you select a graphics option, the output (convoluted sum) can be displayed. Otherwise, samples of CONVOL.DAT file can be displayed using GRAPH CONVOL routine. You can compare signal characteristics before and after convolution (filtering) by successively executing the following commands.

```
> GRAPH SIGNAL
> CONVOL       (with graphics option)
```

Example:

In this example, a Kaiser window based FIR filter is used for convolution. The filter parameters are as follows.

```
Type of Filter:
   Lowpass Filter

Type of Window:
   kaiser Window

Cutoff Frequency:
             0.02

Attenuation in dB:
            30.0

Number of Filter Coefficients:
            13

Filter Coefficients:

             1.461163E-002
             2.069804E-002
             2.669076E-002
             3.206482E-002
             3.632312E-002
             3.905746E-002
             4.000000E-002
             3.905746E-002
             3.632312E-002
             3.206482E-002
             2.669076E-002
             2.069804E-002
             1.461163E-002
```

The signal samples are stored in SIGNAL.DAT file. In this example, we are using same signal as that of an example in chapter V. The input parameters for CONVOL are as follows.

```
Number of Filter Coefficients:
                    13
Number of Signal Samples:
                    600
```

Note that in CONVOL.DAT file, you will have (600 + 13 - 1) = 612 samples. But the GRAPH program uses only 600 samples for display.

VII GRAPHICS

Several modules of SIGAN can be used for independent graphics. You do not have to use any of the modules of SIGAN to view the graphics. Some of the executable files of SIGAN can be invoked directly with appropriate data set to view the graph.

With LINPLOT option, GRAPH can be used to display a filter frequency response with linear magnitude and normalized frequency as parameters. When executed, GRAPH program will read a data file called LINPLOT.DAT to obtain sample values and display on the screen.

Other options for GRAPH include LOGPLOT, PHAPLOT, SIGNAL, NOISE, CONVOL, SPECTRUM, and SAMPLE. LOGPLOT option will display logarithmic-magnitude versus Normalized Frequency plot of samples of LOGPLOT.DAT file. Similarly PHAPLOT option can be used to display phase versus magnitude characteristic of a FIR filter of samples stored in PHAPLOT.DAT file.

SIGNAL option is used to view signal samples stored in SIGNAL.DAT file. Similarly, SPECTRUM option can be used to view spectral samples stored in SPECTRUM.DAT file. Note that SPECTRUM.DAT is created automatically whenever SPECTRUM routine is executed. Similarly, CONVOL option is used to view output of convoluted sum.

How to use GRAPH Program

At the DOS prompt enter the following:

```
> GRAPH  < function >
```

where function can be any of the following

```
LINPLOT
LOGPLOT
PHAPLOT
SIGNAL
NOISE
SAMPLE
CONVOL
SPECTRUM
```

For example, to display spectrum samples stored in SPECTRUM.DAT file, enter the following:

```
> graph spectrum
```

When graphs are displayed on CRT screen, you will see certain range of values on vertical axis. Please note that these values are normalized with respect to a maximum value found in a relevant data set. You actual data set may contain different range of values, especially if you do not normalize them while running the program. But when a graph is displayed, it shall be normalized just for graph only. The actual data set remain unchanged.

For SIGNAL, NOISE, SAMPLE, and CONVOL graphs, up to 600 samples are shown on display. For SPECTRUM, LINPLOT, LOGPLOT, and PHAPLOT graphs, up to 512 points (one-half the spectrum of 1024 point FFT) are displayed.

APPENDIX A SUMMARY OF SIGAN (DSP) MODULES

OPTFIL Program

This program is used for designing optimal FIR filters using Remez exchange algorithm. This is a stand-alone program.

This program produces LINPLOT.DAT, LOGPLOT.DAT, and PHAPLOT.DAT files. Digital filter coefficients are stored in OPTFIL.LOG file.

Upon user selection, this module invokes GRAPH program to display filter responses.

User should enter all parameters at prompts.

WINFIL Program

This program is used for designing window based FIR filters using several window types. This is a stand-alone program.

This program produces LINPLOT.DAT, LOGPLOT.DAT, and PHAPLOT.DAT files. Digital filter coefficients are stored in WINFIL.LOG file.

Upon user selection, this module invokes GRAPH program to display filter responses.

User should enter all parameters at prompts.

IIR1FIL Program

This program is used for designing Butterworth Infinite Impulse Response (IIR) filters. This is a stand-alone program.

No input files are needed for this program. However, user should input several parameters at prompts. The filter coefficients are stored in IIR1FIL.LOG file.

IIR2FIL Program

This program is used for designing Chebyshev Infinite Impulse Response (IIR) filters. This is a stand-alone program.

No input files are needed for this program. However, user should input several parameters at prompts. The filter coefficients are stored in IIR2FIL.LOG file.

SPECTRUM Program

This program implements a complex Fast Fourier Transform (FFT) program of size up to 1024 points. User should enter the number of FFT points.

SPECTRUM module reads data from either SIGNAL.DAT, NOISE.DAT, SIGNAL.DAT+NOISE.DAT, or CONVOL.DAT depending on user option and calculates spectral magnitudes and store them in an output file SPECTRUM.DAT. Real and imaginary parts of FFT output are stored in FFT.DAT file. If you respond "yes" to display option, then a graph will be displayed. Note that up to a maximum of 512 points are displayed on a graph.

CONVOL Program

This program implements convolution on data stored in SIGNAL.DAT and FILTER.DAT files. The output is stored in a CONVOL.DAT file. This is a stand-alone program.

User should input number of samples in each input data file.

Upon user selection, this module invokes GRAPH program to display CONVOL.DAT. The output of this program can also be displayed using GRAPH module with CONVOL option.

SIGSIM Program

This program is used to synthesize multiple sinusoidal signals with different amplitudes, frequencies, and phase shifts and can be superimposed to create a composite test signal.

This is a stand-alone program which accepts user inputs of different signal parameters and also an over-sampling factor. If necessary, noise can be generated and added to this composite signal. If this option is selected, Gaussian noise is generated with user selectable mean, standard deviation, and noise scaling factor. These noise samples will be added to the samples of multiple sinusoidal components and are stored in SIGNAL.DAT file.

This program can invoke GRAPH module with SIGNAL option to display a composite Signal/Noise graph.

NOISIM Program

This program is used to generate a composite noise data for testing purposes. This is a stand-alone program.

The user inputs types of statistical distributions for noise, several noise parameters, and number of samples.

Samples of several distributions can be added to synthesize a composite noise data. This data is stored in a NOISE.DAT file.

This program can invoke GRAPH module with NOISE option to display the composite noise graph.

APPENDIX B Digital Signal Processing References

[1] Rabiner and Gold, "Theory and Application of Digital Signal Processing", Prentice-Hall, 1975.

[2] Oppenheim and Schafer, "Digital Signal Processing", Prentice-Hall, 1975.

[3] Antoniou, "Digital Filters - Analysis and Design", McGraw-Hill, 1979.

[4] IEEE ASSP Committee, "Digital Signal Processing Vol. 1&2, IEEE 1975, 1977.

[5] Elliott, "Handbook of Digital Signal Processing - Engineering and Applications", Academic Press, 1987.