NASA-CR-192834

Determination of Design and Operation Parameters for Upper Atmospheric Research Instrumentation to Yield Optimum Resolution with Deconvolution

GRANT
IN-46-CR
157639
P. 144

NASA Grant NAG 1-804

(NASA-CR-192834) DETERMINATION OF DESIGN AND OPERATION PARAMETERS FOR UPPER ATMOSPHERIC RESEARCH INSTRUMENTATION TO YIELD OPTIMUM RESOLUTION WITH DECONVOLUTION, APPENDIX 4 Final Report (New Orleans Univ.) 144 p
N93-24910
Unclas
G3/46 0157639

FINAL REPORT

APPENDIX 4

Dr. George E. Ioup, Principal Investigator
Dr. Juliette W. Ioup, Principal Investigator
Department of Physics
University of New Orleans
New Orleans, LA 70148

Determination of Design and Operation Parameters for Upper Atmospheric Research Instrumentation to Yield Optimum Resolution with Deconvolution

NASA Grant NAG 1-804

FINAL REPORT

APPENDIX 4

Dr. George E. Ioup, Principal Investigator
Dr. Juliette W. Ioup, Principal Investigator
Department of Physics
University of New Orleans
New Orleans, LA 70148

Least Squares Minimization for Spectral Regions of Combined Lag Windows

Submitted to the Graduate Faculty of the

University of New Orleans in Partial

Fulfillment of the Requirement for the Degree of

Master of Science

in the

Department of Physics

by

Michael Kent Broadhead

B.S., University of Southern Mississippi, 1979

December, 1989

ACKNOWLEDGEMENTS

I would like to express appreciation to Dr. George Ioup and Dr. Juliette Ioup of the UNO physics department for their patient instruction in Fourier and spectral analysis, and their guidance in this thesis research. This work truly would not have been possible without them. I would also like to thank committee members Dr. Charles Head of the UNO physics department and Dr. Grayson Rayborn of the University of Southern Mississippi physics department for their guidance in this research.

Next, I would like to extend thanks to Mr. Joseph Autin and the staff of the Computer Research Center for their assistance with the VAX mainframe. I would also like to thank Mr. N. B. Day and Mr. Ken H. Barnes of the UNO physics department for microcomputer support and Mr. Terry Morris and Mr. David Hasenkampf of the physics department for various forms of help.

Finally, I would like to express appreciation to Ms. Juanita Bordelon of the UNO physics department for general secretarial support.

TABLE OF CONTENTS

PAGE

Acknowledgements.....ii

List of Tables.....iv

List of Figures.....v

Abstract.....ix

Introduction.....1

Theory.....8

Results.....13

Conclusions and Future Considerations.....21

Tables.....24

Figures.....29

References.....109

Appendix A (A Relevant Proof).....111

Appendix B (Quadratic Programming).....113

Appendix C (Computer Code Listing).....116

Vita.....132

LIST OF TABLES

TABLE 1...PROGRAM-1 and PROGRAM-2 Lag Window Function Formulae

TABLE 2...Mean dB levels and Half Power Widths for the 1-2 Hz Test

TABLE 3...Mean dB levels and Half Power Widths for the 2-4 Hz Test

TABLE 4...Mean dB levels and Half Power Widths for the 2-8 Hz Test

TABLE 5...Mean dB levels and Half Power Widths for the 4-8 Hz Test

LIST OF FIGURES

FIG 1....Rectangular Spectral Window

FIG 2....Rectangular Lag Window

FIG 3....Parzen-2 Spectral Window

FIG 4....Parzen-2 Lag Window

FIG 5....Cosine-Tip Spectral Window

FIG 6....Cosine-Tip Lag Window

FIG 7....Bartlett-Like Spectral Window

FIG 8....Bartlett-Like Lag Window

FIG 9....Hann Spectral Window

FIG 10...Hann Lag Window

FIG 11...Hamming Spectral Window

FIG 12...Hamming Lag Window

FIG 13...Papoulis Spectral Window

FIG 14...Papoulis Lag Window

FIG 15...Blackman Spectral Window

FIG 16...Blackman Lag Window

FIG 17...Bartlett Spectral Window

FIG 18...Bartlett Lag Window

FIG 19...Sinc-Like Spectral Window

FIG 20...Sinc-Like Lag Window

FIG 21...Gaussian Spectral Window

FIG 22...Gaussian Lag Window

FIG 23...Cosine Spectral Window, L = 0

FIG 24...Cosine Lag Window, L = 0

FIG 25...Cosine Spectral Window, L = 1

FIG 26...Cosine Lag Window, L = 1

FIG 27...Cosine Spectral Window, L = 2

FIG 28...Cosine Lag Window, L = 2

FIG 29...Cosine Spectral Window, L = 3

FIG 30...Cosine Lag Window, L = 3

FIG 31...Cosine Spectral Window, L = 4

FIG 32...Cosine Lag Window, L = 4

FIG 33...Cosine Spectral Window, L = 5

FIG 34...Cosine Lag Window, L = 5

FIG 35...Cosine Spectral Window, L = 6

FIG 36...Cosine Lag Window, L = 6

FIG 37...Cosine Spectral Window, L = 7

FIG 38...Cosine Lag Window, L = 7

FIG 39...Cosine Spectral Window, L = 8

FIG 40...Cosine Lag Window, L = 8

FIG 41...Program-1, 1-2 Hz Test, Hybrid Spectral Window

FIG 42...Program-1, 1-2 Hz Test, Hybrid Lag Window

FIG 43...Program-2, 1-2 Hz Test, Hybrid Spectral Window

FIG 44...Program-2, 1-2 Hz Test, Hybrid Lag Window

FIG 45...Program-3, 3 Windows, 1-2 Hz Test, Hybrid Spectral Window

FIG 46...Program-3, 3 Windows, 1-2 Hz Test, Hybrid Lag Window

FIG 47...Program-3, 6 Windows, 1-2 Hz Test, Hybrid Spectral Window

FIG 48...Program-3, 6 Windows, 1-2 Hz Test, Hybrid Lag Window

FIG 49...Program-3, 9 Windows, 1-2 Hz Test, Hybrid Spectral Window

FIG 50...Program-3, 9 Windows, 1-2 Hz Test, Hybrid Lag Window

FIG 51...Program-1, 2-4 Hz Test, Hybrid Spectral Window

FIG 52...Program-1, 2-4 Hz Test, Hybrid Lag Window

FIG 53...Program-2, 2-4 Hz Test, Hybrid Spectral Window

FIG 54...Program-2, 2-4 Hz Test, Hybrid Lag Window

FIG 55...Program-3, 3 Windows, 2-4 Hz Test, Hybrid Spectral Window

FIG 56...Program-3, 3 Windows, 2-4 Hz Test, Hybrid Lag Window

FIG 57...Program-3, 6 Windows, 2-4 Hz Test, Hybrid Spectral Window

FIG 58...Program-3, 6 Windows, 2-4 Hz Test, Hybrid Lag Window

FIG 59...Program-3, 9 Windows, 2-4 Hz Test, Hybrid Spectral Window

FIG 60...Program-3, 9 Windows, 2-4 Hz Test, Hybrid Lag Window

FIG 61...Program-1, 2-8 Hz Test, Hybrid Spectral Window

FIG 62...Program-1, 2-8 Hz Test, Hybrid Lag Window

FIG 63...Program-2, 2-8 Hz Test, Hybrid Spectral Window

FIG 64...Program-2, 2-8 Hz Test, Hybrid Lag Window

FIG 65...Program-3, 3 Windows, 2-8 Hz Test, Hybrid Spectral Window

FIG 66...Program-3, 3 Windows, 2-8 Hz Test, Hybrid Lag Window

FIG 67...Program-3, 6 Windows, 2-8 Hz Test, Hybrid Spectral Window

FIG 68...Program-3, 6 Windows, 2-8 Hz Test, Hybrid Lag Window

FIG 69...Program-3, 9 Windows, 2-8 Hz Test, Hybrid Spectral Window

FIG 70...Program-3, 9 Windows, 2-8 Hz Test, Hybrid Lag Window

FIG 71...Program-1, 4-8 Hz Test, Hybrid Spectral Window

FIG 72...Program-1, 4-8 Hz Test, Hybrid Lag Window

FIG 73...Program-2, 4-8 Hz Test, Hybrid Spectral Window

FIG 74...Program-2, 4-8 Hz Test, Hybrid Lag Window

FIG 75...Program-3, 3 Windows, 4-8 Hz Test, Hybrid Spectral Window

FIG 76...Program-3, 3 Windows, 4-8 Hz Test, Hybrid Lag Window

FIG 77...Program-3, 6 Windows, 4-8 Hz Test, Hybrid Spectral Window

FIG 78...Program-3, 6 Windows, 4-8 Hz Test, Hybrid Lag Window

FIG 79...Program-3, 9 Windows, 4-8 Hz Test, Hybrid Spectral Window

FIG 80...Program-3, 9 Windows, 4-8 Hz Test, Hybrid Lag Window

ABSTRACT

The power spectrum for a stationary random process can be defined with the Wiener-Khintchine Theorem, which says that the power spectrum and the autocorrelation function are a Fourier transform pair. To implement this theorem for signals that are discrete and of finite length we can use the Blackman-Tuckey method. Blackman and Tukey (1958) show that a function $w(\tau)$, called a lag window, can be applied to the autocorrelation estimates to obtain power spectrum estimates that are statistically stable. The Fourier transform of $w(\tau)$ is called a spectral window.

Typical choices for spectral windows show a distinct trade-off between the mainlobe width and sidelobe strength. A new idea for designing windows by taking linear combinations of the standard windows to produce hybrid windows was introduced by Smith (1985). We implement Smith's idea to obtain spectral windows with narrow mainlobes and smaller (compared with typical windows) near sidelobes.

One of the main contributions of this thesis is that we show that Smith's problem is equivalent to a Quadratic Programming (QP) problem with linear equality and inequality constraints. A computer program was written to produce hybrid windows by setting up and solving the QP problem. We also developed and solved two variations of the original problem. The two variations involved changing the inequality constraints in both cases from nonnegativity on the combination coefficients to nonnegativity on the hybrid lag window itself. For the second variation, the window functions used to construct the hybrid window were changed to a frequency-variable

set of truncated cosinusoids.

A series of tests was run with the three computer programs to investigate the behavior of the hybrid spectral and lag windows. Emphasis was put on obtaining spectral windows with both relatively narrow mainlobes and the lowest possible (for these algorithms) near sidelobes. Some success was achieved for this goal. A 10 dB peak sidelobe reduction over the rectangular spectral window without significant mainlobe broadening was achieved. Also, average sidelobe levels of -117 dB were reached at a cost of doubling the mainlobe width (at the -3 dB point).

CHAPTER 1

INTRODUCTION

The definition of the power spectrum P(f) due to Wiener (Robinson, 1980) says that P(f) forms a Fourier transform pair with a quantity called the autocorrelation function. The autocorrelation of a stationary, ergodic realization of a random process (Blackman and Tukey, 1958) is

$$C(\tau) = \lim_{T \to \infty} 1/T \int_{-T/2}^{T/2} x(t)x(t+\tau)\, dt.$$

T is the length of a segment of x(t) and τ is a variable called the lag time. It corresponds to the amount of time shift between x(t) and $x(t+\tau)$ when computing the auto (self) correlation. The power spectrum P(f) is the Fourier transform of $C(\tau)$:

$$P(f) = \int_{-\infty}^{\infty} C(\tau)e^{-i2\pi f\tau}\, d\tau$$

where (f) is the frequency. To implement this definition of P(f) for a finite interval of a sampled time series, we need discrete estimators of $C(\tau)$ and the Fourier transform (Oppenheim and Schafer, 1975). For a real (N) point sequence x[n], we can write the autocorrelation function as

$$c[m] = 1/(N-|m|) \sum_{n=0}^{N-|m|-1} x[n]x[n+m].$$

and

$$(1.1) \qquad P[k] = \sum_{m=0}^{N-1} c[m]e^{-i2\pi kn/N}$$

for the discrete Fourier transform (DFT) of the autocorrelation (power spectrum). Blackman and Tukey (1958) realized that the application of the above two formulae leads to statistically unstable power spectrum estimates. The variance of the estimate can be quite large and does not decrease with increasing (N). Blackman and Tukey solved this problem by noticing that the later lags (m) of c[m] have a smaller number of products to average over, and hence, are less reliable estimates. They showed that spectrum estimates made by ignoring later lags became more stable. For Gaussian processes, they recommend keeping only about 10% of the autocorrelation lags. The theoretical properties of lag windows are more conveniently discussed for the continuous case so we will stay with the variable τ for the rest of this chapter.

Deleting later lags amounts to multiplying the autocorrelation estimate with a rectangle function $\Pi(\tau/L)$ of the appropriate length (L), where

$$\Pi(\tau/L) = \begin{cases} 1, & |\tau| < L/2 \\ 0, & |\tau| > L/2 \end{cases} .$$

Note that (Bracewell, 1978) $\Pi(\tau/L) \supset L\,\mathrm{sinc}(Lf) = \sin(\pi Lf)/(\pi f)$ (where $\supset$ denotes the Fourier Transform). According to the Convolution Theorem (Bracewell, 1978), this function, $\sin(\pi Lf)/(\pi f)$, will then be convolved with our spectral estimate. Figure 1 shows this function for L = 1 second. Note that the peak sidelobe is at about -13.5 dB. Our dB

convention uses the following definition

$$W(f)\ (dB) = 20 \log_{10} |\ W(f)/W_{max}|\ .$$

The convolution of the rectangle transform with our spectrum will reduce the variance of our estimate but will introduce other problems at the same time. First, note the main lobe width. This will fundamentally limit our resolution in the spectrum. We are especially concerned about this if we are trying to resolve closely spaced peaks. Also, the sidelobes themselves can mask weak signals in the presence of strong resonances, and generally distort the spectrum (Marple, 1987). Side lobe distortion is sometimes referred to as leakage. Hence, we need to pay close attention to these effects.

First, we make the observation that functions other than $\Pi(\tau/L)$ should also serve to reduce variance (Blackman and Tukey, 1958). Consider a function $w(\tau)$, called a lag window, which has the following properties.

1) $w(\tau) \geq 0$, for all τ

2) $w(\tau) = 0$, $|\tau| > L/2$

3) $w(\tau)$ piecewise continuous

4) $w(-\tau) = w(\tau)$ (symmetric about the origin)

5) $w(0) = 1$

Note that

$$C(0) = \lim_{T \to \infty} 1/T \int_{-T/2}^{T/2} [x(t)]^2\ dt.$$

C(0) is called the power of the signal. Property 5) simply insures that we preserve the power. Note also that the inverse

transform is

$$C(\tau) = \int_{-\infty}^{\infty} P(f)e^{i2\pi f\tau}\, df,$$

so

$$C(0) = \int_{-\infty}^{\infty} P(f)\, df.$$

Thus, property 5) preserves the area under the power spectrum P(f). Property 4) guarantees that P'(f), the Fourier transform of the windowed autocorrelation, is real, where

$$P'(f) = \int_{-\infty}^{\infty} C(\tau)w(\tau)e^{-i2\pi f\tau}\, d\tau.$$

Let $$w(\tau) \supset W(f).$$

We call W(f) the spectral window. If $w(\tau)$ is real and even, W(f) is real and even, since

$$W(f) = \int_{-\infty}^{\infty} w(\tau)\cos(2\pi f\tau)\, d\tau + i\int_{-\infty}^{\infty} w(\tau)\sin(2\pi\tau)\, d\tau.$$

An even function $w(\tau)$ multiplied by the odd function $\sin(2\pi f\tau)$ is odd, and the integral over τ vanishes. Now, $P'(f) = P(f)*W(f)$. The convolution of two real and even functions is real and even (P(f) is real and even for real x(t)).

Eleven typical spectral windows, common (except for window 4) in the literature, are shown in Figures 1-22 (Smith, 1985; Kreamer, 1988; Marple, 1987). Refer to Table 1 for the formulae. Each window has the following properties which are useful to summarize overall behavior

(Marple, 1987);

1) mainlobe width (typically referenced to the -3 dB point),

2) sidelobe decay rate (dB/octave),

and

3) the peak sidelobe (amplitude of the largest sidelobe).

When observing spectral window behavior in Figures 1-22, it is immediately obvious that there is a trade-off between mainlobe width and the peak sidelobe. Typically, having small (near) sidelobes and narrow mainlobes is a conflicting goal (Marple, 1987). One problem in spectral analysis where this trade-off is particularly bothersome is dicussed next.

Consider a signal x(t) that contains a large amplitude sinusoidal component of frequency f'. This "resonance" would ideally show up in the power spectrum as a peak at frequency f'. However, our spectral window W(f) will be convolved with the peak, and the sidelobes may mask weak signals at frequencies near f'. Our spectral window must have a mainlobe narrow enough to resolve the two signals, and needs small near sidelobes if we are to detect the weak spectral response. This creates a challenging window design problem in trying to overcome these apparently conflicting goals. The problem of designing such a spectral window will constitute the main emphasis of the rest of this work.

Smith (1985) posed a mathematical statement of a possible scheme to design the desired window. He suggested that we might be able to generate a better window than the standard literature windows if we take a linear combination of them. Let's define the windows in Table 1

as $w_i(\tau)$, where i=1,M (1 to 11 here). The linear combination will produce a "hybrid" window $w_H(\tau)$. Then,

$$w_H(\tau) = \sum_{i=1}^{M} a_i w_i(\tau).$$

To insure that $w_H(\tau)$ has property 1), we can impose a nonnegativity condition on the a_i, giving

$$a_i \geq 0.$$

To insure property 5) we need

$$\sum_{i=1}^{M} a_i = 1.$$

Note that if $w_H(\tau) \supset W_H(f)$,

then, since $W_H(\tau) = \Sigma\, a_i w_i(\tau)$,

we have $W_H(f) = \Sigma\, a_i W_i(f)$,

where $w_i(\tau) \supset W_i(f)$.

For a criterion to determine an appropriate set of a_i, we can use the least squares measure to insure that the spectral window is small over a desired frequency band f_1 to f_2. That is, choose a_i such that

$$\int_{f_1}^{f_2} | W_H(f) |^2 \, df$$

is a minimum. This problem, and two variations to be discussed later, form the basic subject of study in this work. A brief survey of the contents of the rest of this thesis follows next.

In Chapter 2 we proceed to solve the problem of how to determine the a_i so that w_H is a valid lag window while minimizing the above integral. We will also discuss two variations of the original problem. In Chapter 3 we discuss the algorithmic details of implementing (as computer codes) the solutions presented in Chapter 2, and discuss a series of tests that were run with the programs. The figures (1 through 80) that follow after Chapter 4 are also discussed. In Chapter 4 some general conclusions are drawn about algorithmic behavior from the tests that we discussed in Chapter 3.

CHAPTER 2

THEORY

In Chapter 1 we posed an optimization problem with the following form

$$\min F(a) = \int_{f_1}^{f_2} | W_H(f) |^2 \, df,$$

$$\text{subject to } a_i \geq 0$$

$$\text{and} \quad \sum_{i=1}^{M} a_i = 1,$$

where
$$W_H(f) = \sum_{i=1}^{M} a_i W_i(f)$$

and
$$a^T = [a_1, \ldots, a_M].$$

Since
$$w_i \supset W_i(f)$$

and the w_i are even, we have that the $W_i(f)$ are real. Thus,

$$F(a) = \int_{f_1}^{f_2} [W_H(f)]^2 \, df$$

$$= \int_{f_1}^{f_2} [\sum_i a_i W_i(f)]^2 \, df$$

$$= \sum_i \sum_j a_i \, a_j \int_{f_1}^{f_2} W_i(f) \, W_j(f) \, df$$

Now we know the $W_i(f)$ so, in principle, we could carry out the indicated integration. In practice, this would require numerical quadrature techniques. Another approach would be to evaluate $W_i(f)$ numerically with the discrete Fourier transform (DFT), and change the integral to a discrete sum over $W_i(k)W_j(k)$. Recall that the DFT was defined in Chapter 1 in Eq. No. (1.1). We will proceed in the following manner. Let

$$\int_{f_1}^{f_2} W_i(f)\ W_j(f)\ df \quad \rightarrow \quad \sum_{k=k_1}^{k_2} W_i(k)\ W_j(k)$$

or

$$\rightarrow \quad \sum_{k=k_1}^{k_2} W_{ki}\ W_{kj}$$

where k_1 and k_2 are the indices that correspond to the frequency points f_1 and f_2. That is,

$$f_n = (k_n - 1)\ \Delta f,$$

where $\Delta f = 1/(N\Delta\tau)$, and $\Delta\tau$ is the sample rate for $C(\tau)$ (to give c[m]). N is the total number of sample points.

Consider a matrix W with elements W_{ij}, where (j) is the window number and (i) is the frequency index. W is $(k_2 - k_1 + 1)$ X M in size. Then,

$$\sum_k W_{ki}\ W_{kj} = \sum_k W^T_{ik}\ W_{kj},$$

where W^T_{ik} is an element of W^T and $\sum_k W_{ki}\ W_{kj}$ is the ij^{th} element of W^TW. W^TW is square, M X M, and symmetric. Let $H = W^TW$ for convenience.

Then

$$F(\mathbf{a}) = \sum_i \sum_j a_i a_j \sum_k W_{ki} W_{kj} = \sum_i \sum_j a_i H_{ij} a_j$$
$$= \sum_i a_i (\sum_j H_{ij} a_j),$$

which can be written as

$$F(\mathbf{a}) = \mathbf{a}^T \mathbf{H} \mathbf{a} = \mathbf{a}^T (\mathbf{W}^T \mathbf{W}) \mathbf{a} = (\mathbf{W}\mathbf{a})^T (\mathbf{W}\mathbf{a})$$

In general, a quadratic form (Scales, 1985) is a function of $\mathbf{x}$ that can be written as

$$f(\mathbf{x}) = 1/2\ \mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c.$$

$\mathbf{A}$ is called the Hessian. Obviously, our objective function $F(\mathbf{a})$ is a quadratic form, where we identify

$$1/2\ \mathbf{A} = \mathbf{H}$$
$$\mathbf{b} = 0$$
$$c = 0.$$

We have immediately that $\mathbf{H}$ is symmetric, since

$$\mathbf{H} = \mathbf{W}^T \mathbf{W}$$
$$\mathbf{H}^T = (\mathbf{W}^T \mathbf{W})^T = \mathbf{W}^T \mathbf{W} = \mathbf{H}.$$

or

$$\mathbf{H} = \mathbf{H}^T.$$

Furthermore, $$F(a) = \int_{f_1}^{f_2} | W_H(f) |^2 \, df \geq 0,$$

so $F(\mathbf{a}) \geq 0$ for any choice of $\mathbf{a}$. By definition, a matrix $\mathbf{A}$ is nonnegative definite if and only if, for any $\mathbf{x}$, $\mathbf{x}^T \mathbf{A} \mathbf{x} \geq 0$. Hence, since

$$F(\mathbf{a}) = \mathbf{a}^T \mathbf{H} \mathbf{a} \geq 0$$

for any $\mathbf{a}$, we have that $\mathbf{H}$ is symmetric and nonnegative semidefinite. It is instructive to note that the vector $\mathbf{a}$ that minimizes $F(\mathbf{a})$ is $(0,\ldots,0)$, or the trivial solution. This is shown in Appendix A.

In summary, then, we wish to find an **a** that miminizes

$$\mathbf{a}^T\mathbf{H}\mathbf{a}$$

subject to $\sum_i a_i = 1$

and $a_i \geq 0,$

given that H is symmetric and nonnegative semidefinite. This is a special case of a more general mathematical problem in optimization theory call the Quadratic Programming problem (Bronson, 1982). The full formulation allows more general linear constraints and a nonzero **b** and c (although c does not affect the solution).

Quadratic programming (QP) is an effective way in general to deal with least squares problems if inequality constraints are needed. Various approaches for the solution of this problem are discussed by Bronson (1982), Lawson and Hanson (1974), Scales (1985), and Hillier and Lieberman (1967). The solution to the QP problem investigated in this work is a code, called DQPROG, from the IMSL MATH$^{T.M.}$/Library. The details of QP are beyond the scope of the work, but a brief description is given in Appendix B.

In addition to the QP problem we have just outlined, two variations were also studied in this work. We will refer to the above QP problem henceforth as Program-1. The first variation we will consider is a relaxation of the nonnegativity constraints on **a**. Instead, we will impose this condition on the hybrid lag window itself. The inequality constraints then have the following form:

$$\mathbf{B}\mathbf{a} \geq 0,$$

where B_{ij} is the lag window value for the ith time sample and the jth window. B is (N X M), where N is the number of lag window samples used

for the DFT. This new problem will be referred to as Program-2.

The second variation on Program-1 we wish to consider is that of retaining the constraints of Program-2, but changing the functions $w_i(\tau)$. The new functions are (Nuttall, 1981)

$$(2.1) \qquad w_{L+1} = \cos(2\pi L\tau/T), \; L = 0,1,\ldots,M$$

where T is the window length and L is an integer. They offer the advantage of being more general than those of Program-1, and are reminiscent of a Fourier construction. This variation will be called Program-3. Examples of the implementation of Programs-1, 2 and 3, with a discussion of results, are presented in Chapter 3.

CHAPTER 3

RESULTS

Introduction

A general outline of the algorithm for Program-1 is given first, followed by an introduction to the tests that were run. Program-2 and Program-3 follow this outline as well; only the constraints and/or the window functions differ.

1) The Discrete Fourier Transform (DFT) was discussed in Chapter One. An algorithm for efficient evaluation of the DFT is called the Fast Fourier Transform (FFT). An FFT (Claerbout, 1976) of the radix 2 type was used to evaluate W. The $w_i(\tau)$ (see Table 1 and Figures 1-22) were sampled at 256 points, and zero padded to 1024 points before computing the FFT. The zero padding was used to obtain a more densely sampled transform.

2) W was constructed by using the window FFT's over the chosen frequency band. The FFT over this band for window $w_i(\tau)$ becomes the ith column of W. Then H was constructed by $H = W^T W$.

H and the constraints were given to DQPROG (an IMSL quadratic programming subroutine), which returned a solution vector a.

3) The hybrid spectral window was constructed in the function domain using $\sum_i a_i w_i(\tau)$, and then sampled and the FFT computed in the same manner as in step 1 (except that the zero padding was to 2048 points).

Now we will consider the testing that was done using Programs-1, 2 and 3. These three programs were tested for performance by varying, in each case, the frequency band we wish to attenuate, computing two quantitative measures that relate to performance (discussed below),

and displaying the resulting hybrid spectral windows along with a graph of the rectangular spectral window for comparison. The two quantitative measures of performance are the mean dB level (MDB) over the desired attenuation band and the half power width (HPW). The half power width is the frequency (Hz) that corresponds to the -3 dB point on the spectral window of interest. The mean dB levels and half power widths are listed for each test in Table 2 through Table 5.

The testing strategy was as follows. For Program-1 and Program-2, the eleven lag windows listed in Table 1 and Figures 1 through 22 were used to compute the hybrid spectral windows. For Program-3, it was of interest to see how the results would vary with different numbers of the cosine windows, since the window number scheme (window number 1,2, etc.) corresponds to the frequency of the lag window. That is, increasing window numbers mean increasing frequencies (see Figures 23 through 40). The maximum number of cosine windows tested was 9 (corresponding to L = 8 in equation number 2.1).

For all three programs, the desired frequency band for attenuation was varied. Figures 41 through 50 show results for the frequency band 1 through 2 Hz. Figures 51 through 60 show results for the frequency band 2 through 4 Hz. Figures 61 through 70 show results for the frequency band 2 through 8 Hz. Figures 71 through 80 show results for the frequency band 4 through 8 Hz. The rationale for this testing scheme was to observe window behavior for the cases in which the frequency band was close to and farther away from the origin, and in which the frequency band width itself varied.

In the case of Program-3, for each frequency band tested, three different combinations of windows were used. The combinations were:

window numbers 1 through 3 (L = 0 through L = 2), window numbers 1 through 6 (L = 0 through L = 5), and window numbers 1 through 9 (L = 0 through L = 8).

Discussion of Tests

Test for 1 through 2 Hz

For the 1 - 2 Hz test (Figures 41 through 50), the MDB's range from -27 dB to -29 dB, and the HPW's range from 0.46 to 0.48 Hz (refer to Table 2). The average MDB is about -28 Hz and the average HPW is about 0.47 Hz. These values are fairly clustered, so to further judge performance, we can visually inspect the spectral and lag windows in Figures 41 through 50. For comparison purposes, the rectangular spectral window (also shown in Figure 1) is overplotted with dashed lines. Note that the HPW of the rectangular spectral window (RSW) is 0.44 Hz, and the peak sidelobe is at -13.5 dB.

For each case, we note that the HPW's of the hybrid spectral windows are relatively close to the RSW width, while the peak sidelobes of the hybrid windows inside the attenuation band are at about -23 dB. This is a 10 dB improvement over the RSW without much gain in window width (increases by a factor of 1.07), which is an encouraging result.

The relative performance of Programs 1, 2, and 3 (3 windows, 6 windows, and 9 windows) is discussed next. First, let's adopt a shorthand notation. Let Program-1 be represented by P1, Program-2 by P2, and Program-3, 3 windows, 6 windows, and 9 windows be respectively represented by P3;3, P3;6, and P3;9. We can summarize the relative window behavior, which we observe visually, by saying that the side-

lobe behavior, outside of the attenuation band, of P2 (Figure 43), P3;6 (Figure 47), and P3;9 (Figure 49) is relatively poor. On the other hand, P1 (Figure 41) and P3;3 (Figure 45) have sidelobe behavior outside of the attenuation band that is quite near that of the RSW. As we would expect, the lag windows for P1 and P3;3 (Figures 42 and 46) look quite similar, while the lag windows for P2, P3;6, and P3;9 (Figures 44, 48, and 50) are quite variable with respect to each other. It is interesting to compare the lag windows for P1 and P3;3 with the rectangular lag window (Figure 2). The hybrid lag windows start at 1.0, smoothly taper off to about 0.6, and then stay fairly level. This behavior is somewhat different from the standard lag windows (included in Figures 2 through 22).

Test for 2 through 4 Hz

For the 2 - 4 Hz test (Figures 51 through 60), the MDB's range from -54 dB to -64 dB, and the HPW's range from 0.62 through 0.66 Hz (refer to Table 3). The average MDB is about -58 dB and the average HPW is about 0.64 Hz. Note that there is more variability in the MDB's than was the case for the 1 - 2 Hz test. However, the HPW's do not significantly vary.

In moving the frequency band away from the origin (0 Hz), we have allowed the average HPW to increase over the RSW width (0.44 Hz). The increase is by a factor of 1.45. Also, by moving the attenuation band, we have improved our attenuation to an average dB level of -58 dB.

In comparing P1 (Figure 51) to P2 (Figure 53), we note that P2 has a slightly better attenuation level in the 2 - 4 Hz band, but larger sidelobes outside of the band. However, P2's behavior outside

of the attenuation band is better than in was in the 1 - 2 Hz case. P1 and P3;3 (Figure 54) have very similar behavior. P2 has slightly better behavior inside the attenuation band than P3;3, and slightly worse behavior outside of the band. P3;6 (Figure 57) and P3;9 (Figure 59) exhibit similar behavior in the 2 - 4 Hz band, and have comparable mainlobe widths, but P3;6 has much better sidelobe behavior outside of the attenuation band than does P3;9. Compared to the other windows (except P3;9) however, P3;6 has poor sidelobe behavior outside of the 2 - 4 Hz band. Note that the lag windows for P1 (Figure 52), P2 (Figure 54), and P3;3 (Figure 56) are quite similar. The lag windows for P3;6 (Figure 58) and P3;9 (Figure 60) are considerably different from each other and the other lag windows. The P3;9 lag window has an oscillatory behavior uncharacteristic of typical lag windows. However, P3;9 has the best attenuation characteristics for the 2 - 4 Hz case in the attenuation band.

Test for 2 through 8 Hz

For the 2 - 8 Hz test (Figures 61 through 70), the MDB's range from -55 dB to -57 dB, and the HPW's range from 0.62 Hz to 0.66 Hz (refer to Table 4). The average MDB is about -56 dB and the average HPW is about 0.67 Hz. It is evident from Figures 62 through 69 that the results for the spectral and lag windows are fairly uniform as compared with the last two tests. Going to a wider bandwith seems to have a stabilizing effect on the programs' behavior. It would be difficult to choose one window over the other from this test, except for the P3;9 (Figure 69) window, which has the poorest behavior and would be excluded. Towards the end of the frequency band, the

sidelobes of P3;9 come back to a higher level than for the other spectral windows.

In comparison to the 2 - 4 Hz case, the P1's (Figures 51 and 61) are about the same. For the P2's (Figures 53 and 63), a significant improvement in the behavior of the 2 - 8 Hz case can be noted. The sidelobes from 4 - 8 Hz are much lower than for the 2 - 4 Hz case. This makes a lot of sense, and shows that the algorithms are, fortunately, behaving in an intuitive and reasonable way. The results are similar but even more dramatic when comparing the P3;6 and P3;9 cases for the 2 - 8 Hz test (Figures 67 and 69) with the 2 - 4 Hz test (Figures 57 and 59). It seems safe to conclude that we gain more than we lose by opening up the bandwidth when possible.

Test for 4 through 8 Hz

For the 4 - 8 Hz test (Figures 71 through 80), the MDB's range from -83 dB to -117 dB, and the HPW's range from 0.81 Hz to 0.93 Hz (refer to Table 5). The average MDB is about -101 dB, and the mean HPW is about 0.88 Hz. The average MDB has almost doubled over the 2 - 8 Hz case, and the average HPW has increased by a factor of about 1.3. We note for this test that we have a broader range of results in terms of attenuation levels and mainlobe widths across the various programs.

Specifically, P3;6 (Figure 77) and P3;9 seem to be in a class by themselves with much higher attenuation levels (better than -100 dB), but broader main lobes. However, P3;6 and P3;9 are fairly comparable to each other. It is also interesting to note that sidelobe behavior outside the attenuation band is fairly good for all cases. The 4 - 8 Hz band, being farther from the origin, seems to allow more stable results. This is also reflected in the lag window behavior. The lag

windows for the 4 - 8 Hz case (Figures 72, 74, 76, 78, and 80) are all fairly comparable and well behaved (smoothly tapering to a low value).

The spectral windows for P1 (Figure 71), P2 (Figure 73), and P3;3 (Figure 75) are similar. Their behavior can be characterized in the following way. As we progress through the above sequence, the overall mean attenuation improves, but the attenuation nearest the mainlobe deteriorates. We can summarize the extremes of this test by saying that P1 has the narrowest mainlobe behavior, and P3;9 has the best sidelobe behavior. One would have to choose from them according to need.

Conclusions

We will summarize here some of the conclusions that are suggested by the above tests. The behavior differences between Programs-1, 2, and 3 can be significant. This is illustrated in the 1 - 2 Hz test by the instabilities (large sidelobes) exhibited for Program-3, and the differences between Program-1 and Program-3 results in the 4 - 8 Hz test. Program-1 and Program-2 show their greatest differences for cases where the attenuation band is close to the origin.

In general, Program-2 seems to have less stable sidelobe behavior outside the attenuation band than does Program-1. In most of the cases that we looked at, Program-1 and Program-3 (3 windows) seemed to be the most similar, but for the case away from the origin (4 - 8 Hz), Program-3 (6 windows) and Program-3 (9 windows) were very similar. In some cases, all the programs produced similar results (2 - 8 Hz test).

We can conclude that we will get narrow mainlobe widths (close to RSW) if we put the attenuation band close to the orign, but pay the price of low attenuation (-28 dB). Conversely, good attenuation (-100 dB) and broader mainlobes (twice that of RSW) are obtained for attenuation bands farther away from the origin. It also should be noted that opening up the bandwidth can have a stabilizing effect on the window's sidelobe behavior. A similar stability effect seems to be at work when we move the attenuation band away from the origin.

CHAPTER 4

CONCLUSIONS AND FUTURE CONSIDERATIONS

In this work we have reviewed the basic ideas involved with the Blackman-Tukey method of spectral estimation and, specifically, have investigated the problems associated with designing spectral lag windows. Three design ideas were implemented and tested. Program-1 used linear combinations of lag windows found in the literature. The combination coefficients were determined by solving a constrained least squares problem in which the objective function measured the spectral window response over a given frequency region. The constraints consisted of 1) nonnegativity on the combination coefficients and 2) that the combination coefficients sum to unity. For Program-2, the constraints were changed so that the hybrid window itself was nonnegative. In Program-3, the constraints of Program-2 were used with a different class of window functions (frequency variable cosinusoids).

Each of the three techniques was tested for performance over four different frequency bands, 1 - 2 Hz, 2 - 4 Hz, 2 - 8 Hz, and 4 - 8 Hz. In the case of Program-3, the number of windows was varied to include the first three cosine windows, then the first six, and finally, the first nine. Based on these tests, we can tentatively conclude that:

1) Relaxing the nonnegativity constraint on the a_i did not offer the advantages that were hoped for. The results for Program-2 were at best comparable to Program-1, and in some cases, not as good.

2) The difference in behavior for Programs-1, 2 and 3, for attenuation

bands close to the origin and of a narrow bandwidth, is largely in how the sidelobes behave outside the attenuation band. The best results for this case were from Program-1 and Program-3 (3 windows).

3) Using the cosine windows in Program-3 to try to improve over the original idea of Program-1 was a mixed success. Program-3 seems mainly to offer advantages over Program-1 in the case where the attenuation band is farther away from the origin (e.g., 4 - 8 Hz). For this case, Program-3 achieved much better attenuation levels (at the cost of mainlobe width) provided we use enough windows.

4) The level of attenuation improves and the main lobes get broader for all three programs as the attenuation band moves away from the origin.

5) Increasing the attenuation bandwidth improves sidelobe behavior in general. We do not have to give up much in terms of attenuation levels and mainlobe widths.

6) The number of cosine windows appropriate to use in Program-3 varies as follows: for attenuation bands close to the origin, use fewer windows; for bands farther away, use more. Of course, the mainlobe widths will increase for the latter case.

For future efforts along the line of research presented in this thesis, several things could be done.

1) Implementation of the algorithm in which H is evaluated by numerical integration rather than by matrix multiplication (W^TW) may offer advantages.

2) The three programs need to be studied more thoroughly in terms of their behavior with respect to various combinations of windows and spectral attenuation bands.

3) Modeling needs to be done to quantify the performance of the programs. For example, if we take two sinusoids of some relative magnitude at some frequency spacing, and truncate them to some specific length, we can then try to design a window that will resolve them. The relative strengths and frequency spacings could then be varied.

4) In this study we ignored the sign of the hybrid spectral windows. The degree to which these windows have negative lobes and their characterization should be investigated. This should provide another criterion for judging window performance. It is desirable to have no negative sidelobes in the spectral window, but several typical window functions have some. The severity of this problem depends to some extent on the applications in mind.

TABLE 1

Program-1 and Program-2 Lag Window
Functions (defined on an interval [-1/2, 1/2] secs)

WINDOW NO.	NAME	FORMULA $w_i(\tau)$
1	Rectangular	1
2	Parzen-2	$1 - 4\tau^2$
3	Cosine-tip	$\cos(\pi\tau)$
4	Bartlett-like	$1 + 2\lvert\tau\rvert$
5	Hann	$.5 + .5\cos(2\pi\tau)$
6	Hamming	$.54 + .46\cos(2\pi\tau)$
7	Papoulis	$1 - 2\lvert\tau\rvert\cos(2\pi\tau) + (1/\pi)\lvert\sin(2\pi\tau)\rvert$
8	Blackman	$.42 + .5\cos(2\pi\tau) + .08\cos(4\pi\tau)$
9	Bartlett	$1 - 2\lvert\tau\rvert$
10	Sinc-like	$\sin(2\pi\tau)/(2\pi\tau)$
11	Gaussian ($\alpha = 2.5$)	$\exp(-.5[2\alpha\tau]^2)$

TABLE 2

Mean dB Levels (MDB) and Half Power Widths (HPW)
for the 1 - 2 Hz Test

	MDB (db)	HPW (Hz)
P1	-27	0.48
P2	-28	0.46
P3;3	-28	0.47
P3;6	-29	0.46
P3;9	-29	0.46

TABLE 3

Mean dB Levels (MDB) and Half Power Widths (HPW)
for the 2 - 4 Hz Test

	MDB (db)	HPW (Hz)
P1	-54	0.66
P2	-58	0.64
P3;3	-55	0.66
P3;6	-60	0.63
P3;9	-64	0.62

TABLE 4

Mean dB Levels (MDB) and Half Power Widths (HPW)
for the 2 - 8 Hz Test

	MDB (db)	HPW (Hz)
P1	-56	0.67
P2	-57	0.67
P3;3	-55	0.68
P3;6	-57	0.67
P3;9	-57	0.67

TABLE 5

Mean dB Levels (MDB) and Half Power Widths (HPW)
for the 4 - 8 Hz Test

	MDB (db)	HPW (Hz)
P1	-83	0.81
P2	-99	0.85
P3;3	-97	0.85
P3;6	-110	0.94
P3;9	-117	0.93

Figure 1 RECTANGULAR

SPECTRAL WINDOW

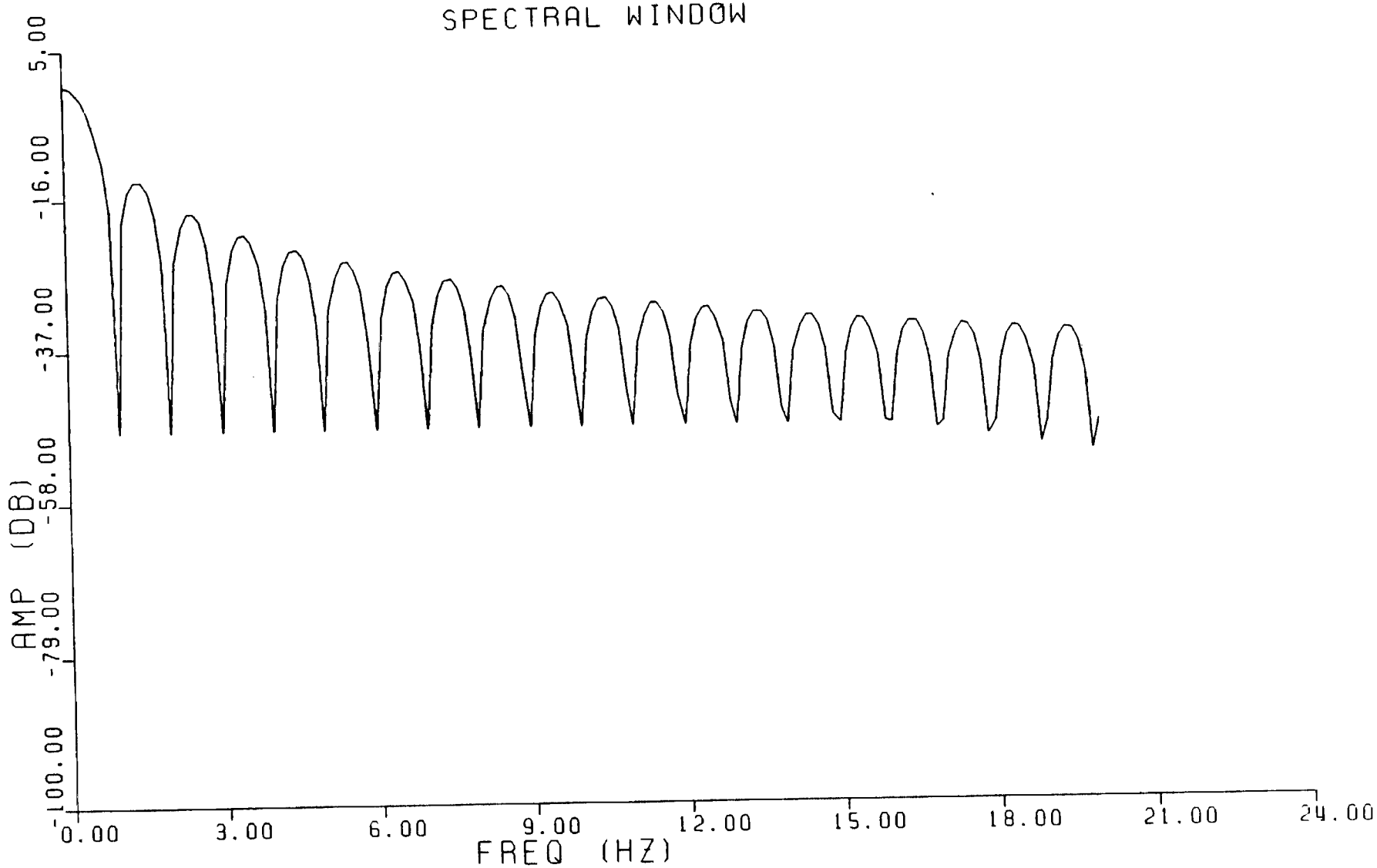


Figure 2 RECTANGULAR LAG WINDOW

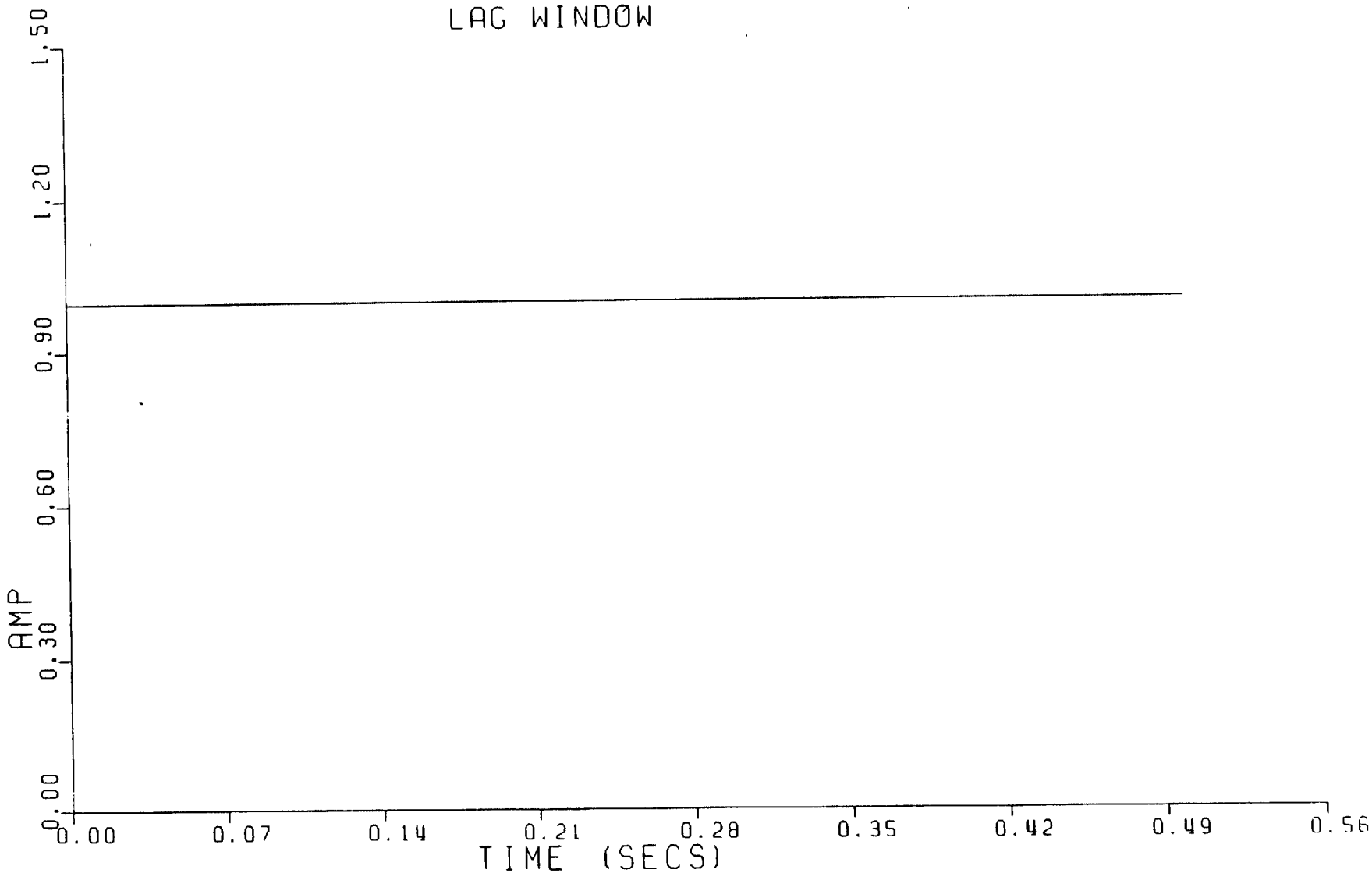


Figure 3 PARZEN-2

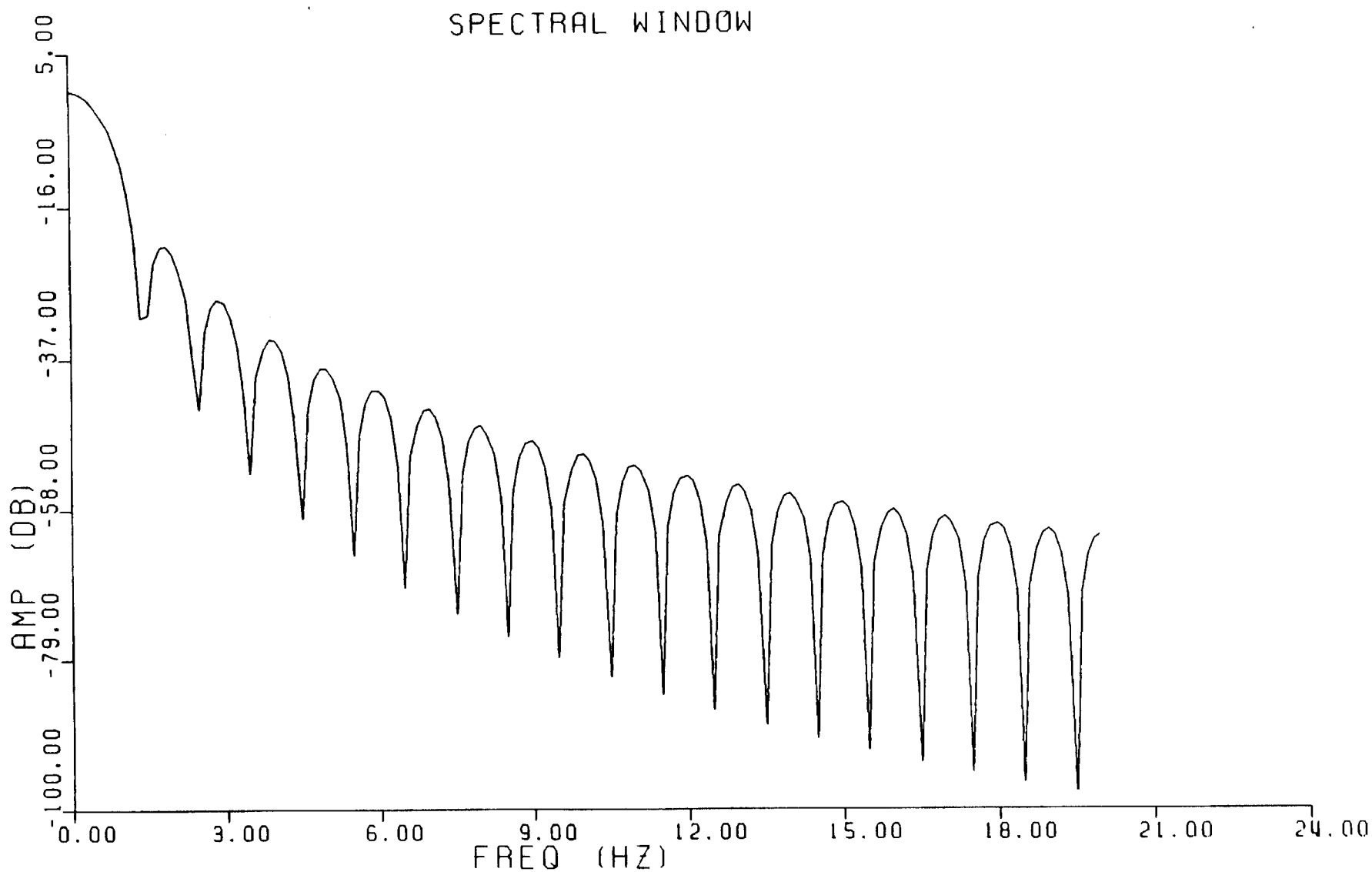


Figure 4 PARZEN-2

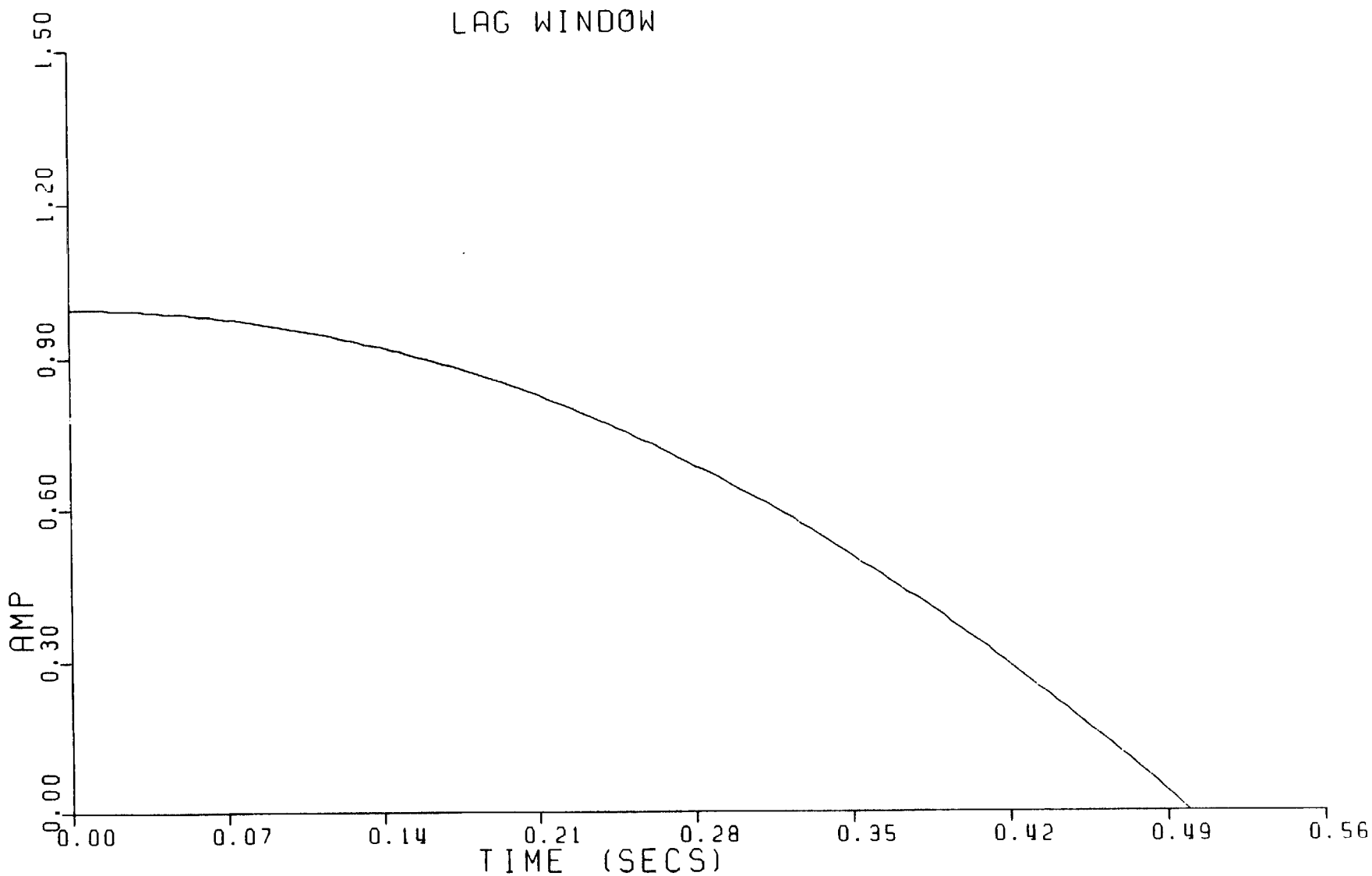


Figure 5

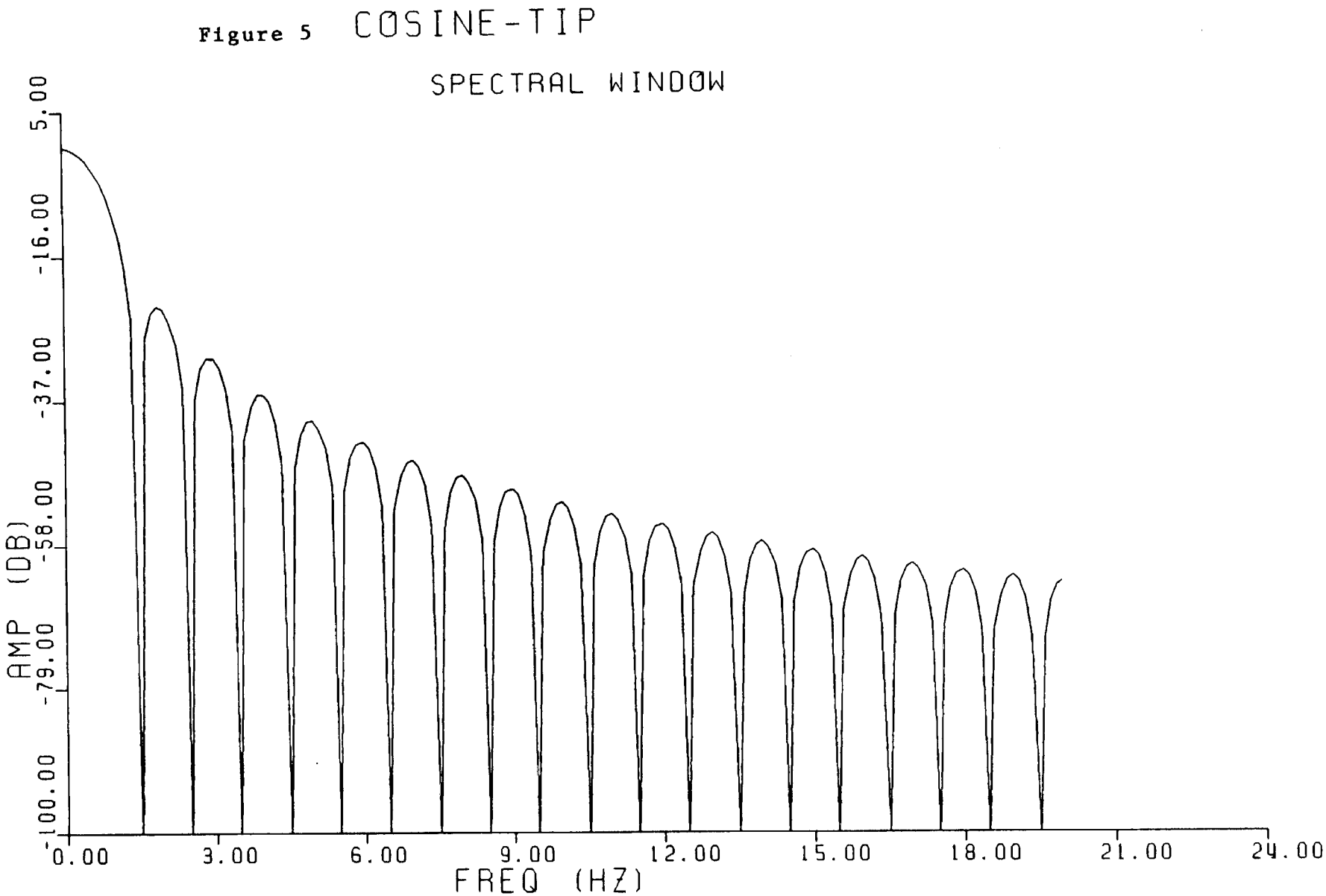

COSINE-TIP
SPECTRAL WINDOW
AMP (DB)
5.00
-16.00
-37.00
-58.00
-79.00
-100.00
0.00
3.00
6.00
9.00
12.00
15.00
18.00
21.00
24.00
FREQ (HZ)

Figure 6

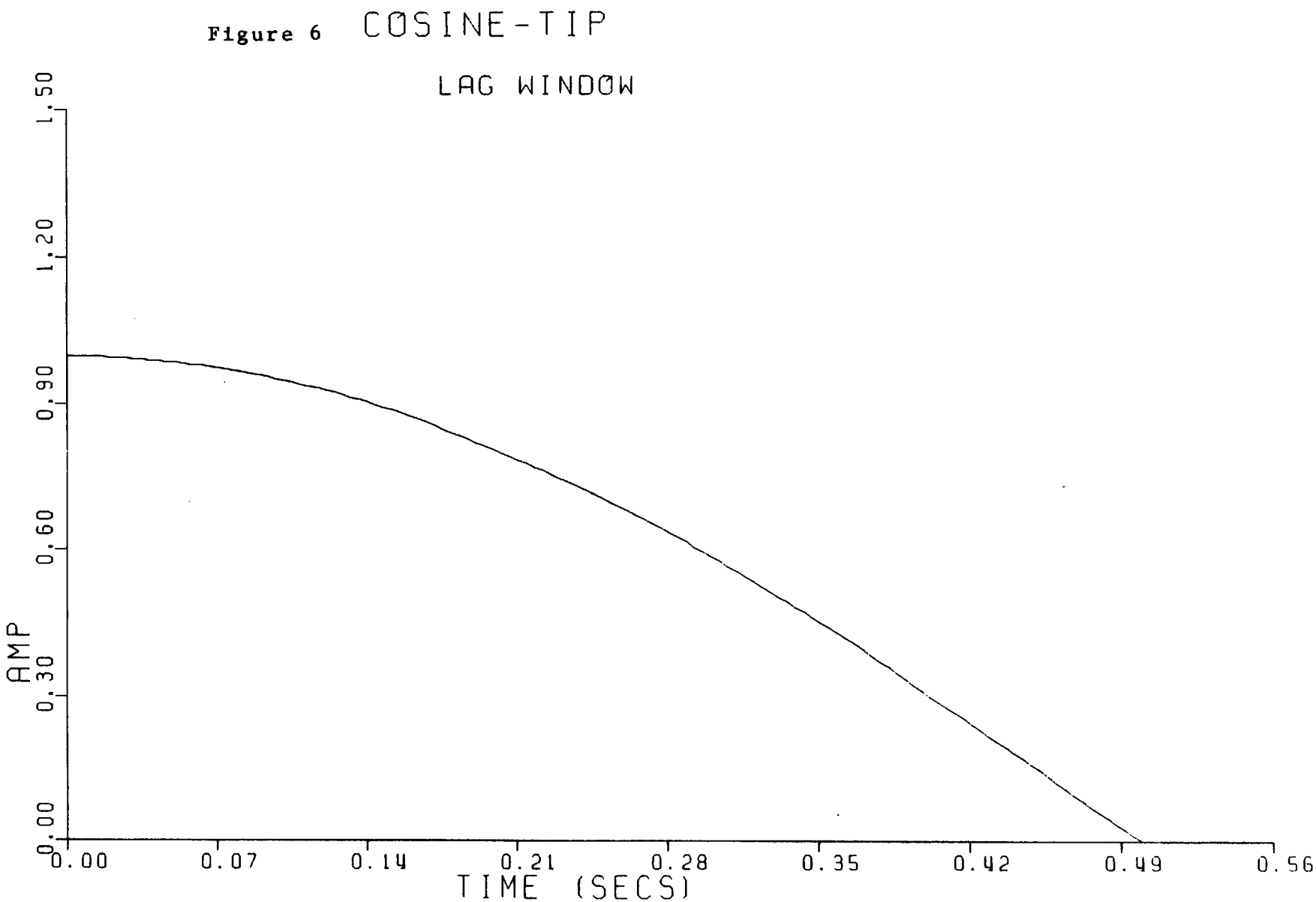

COSINE-TIP
LAG WINDOW
AMP
0.00
0.30
0.60
0.90
1.20
1.50
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
0.56
TIME (SECS)

Figure 7 BARTLETT-LIKE SPECTRAL WINDOW

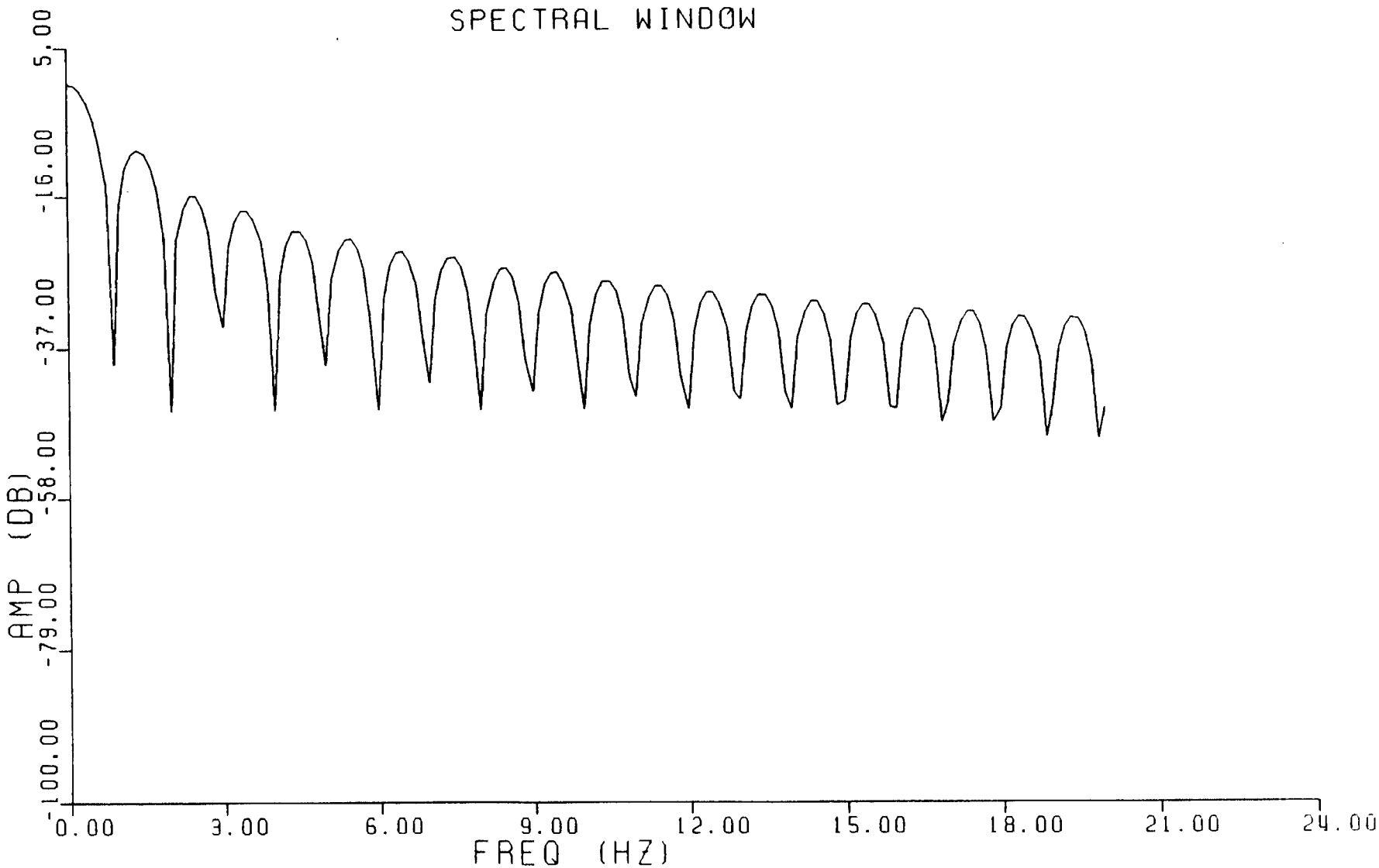


Figure 8 BARTLETT-LIKE

LAG WINDOW

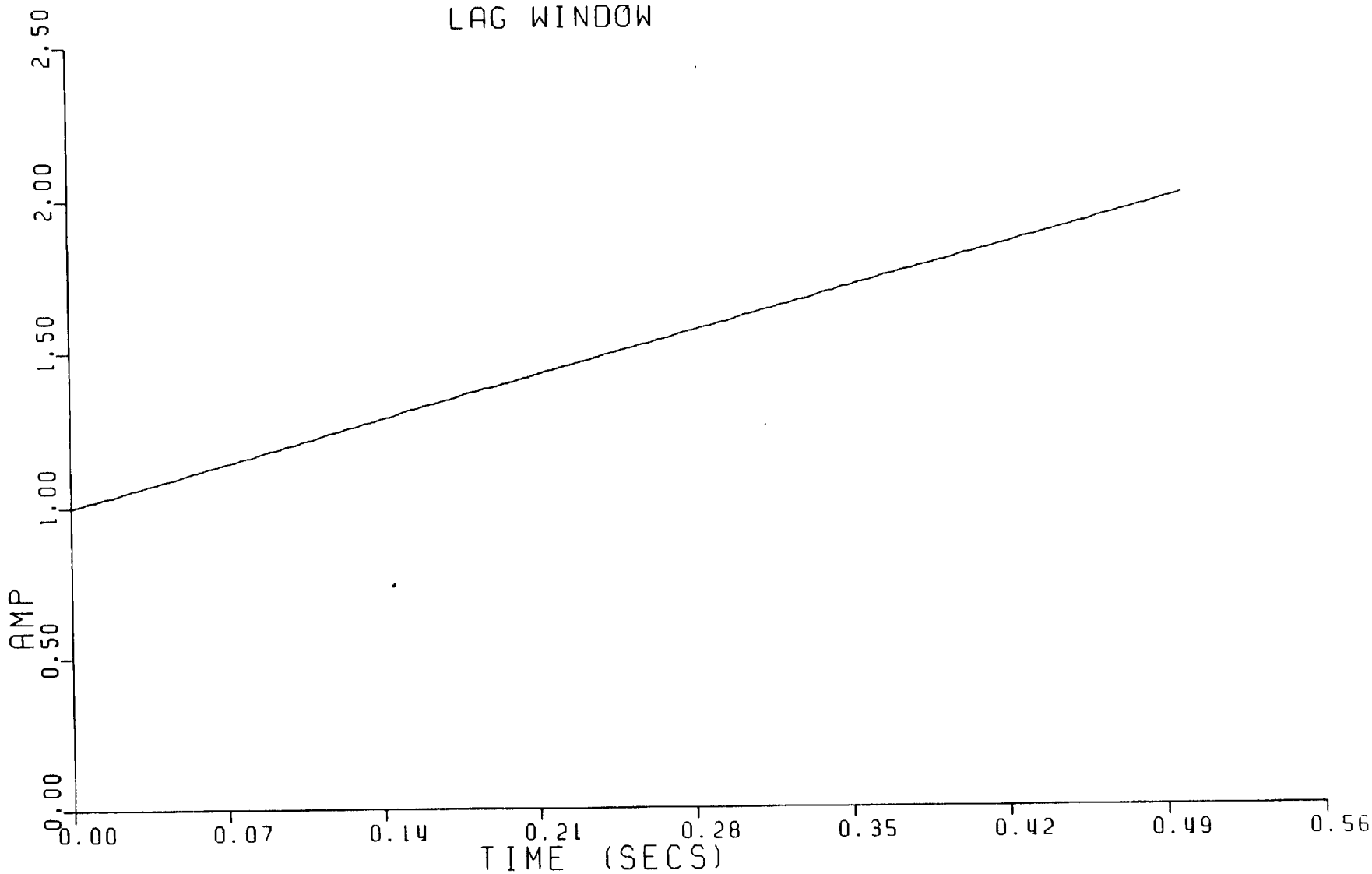


Figure 9 HANN

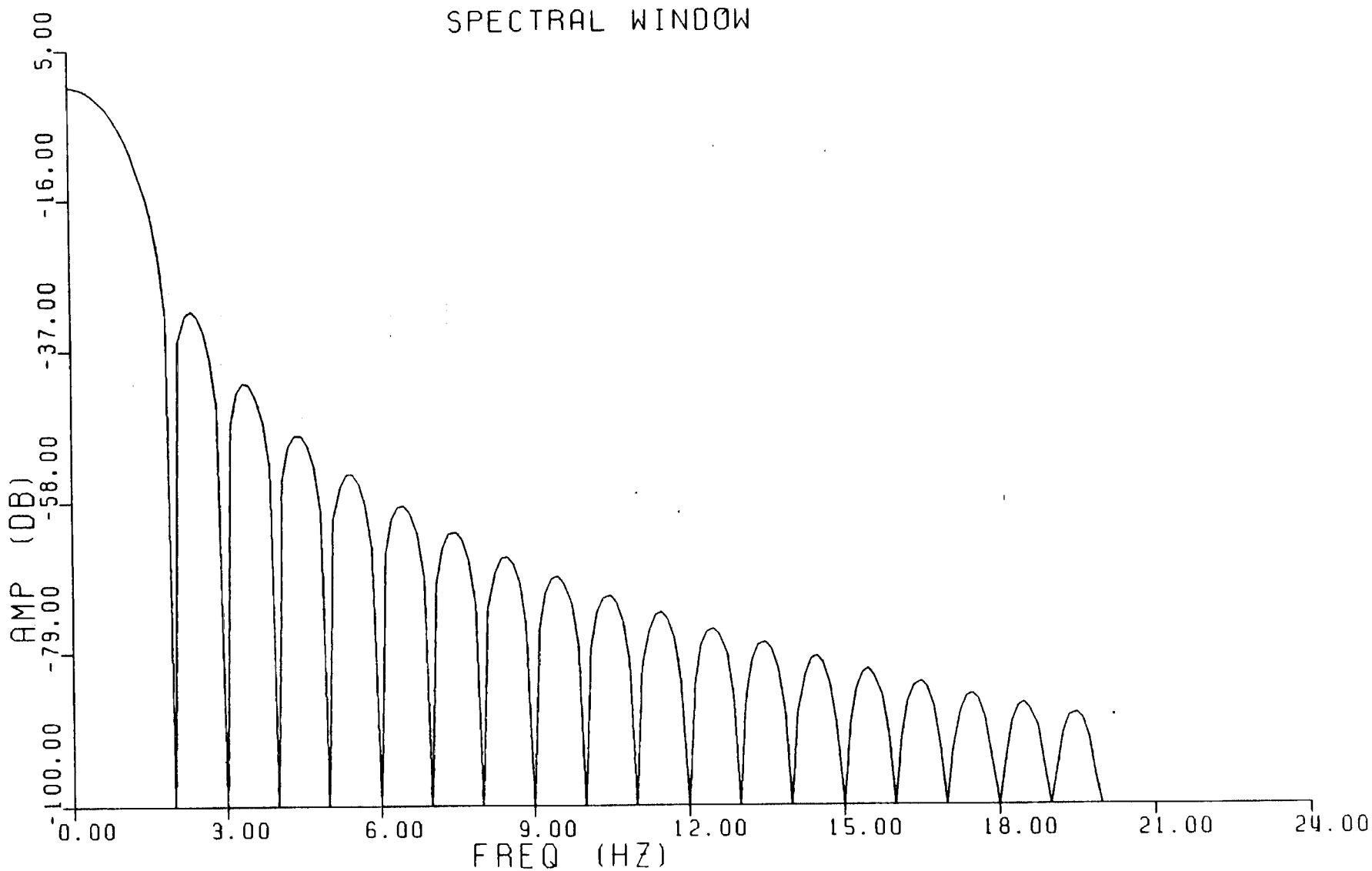


Figure 10 HANN

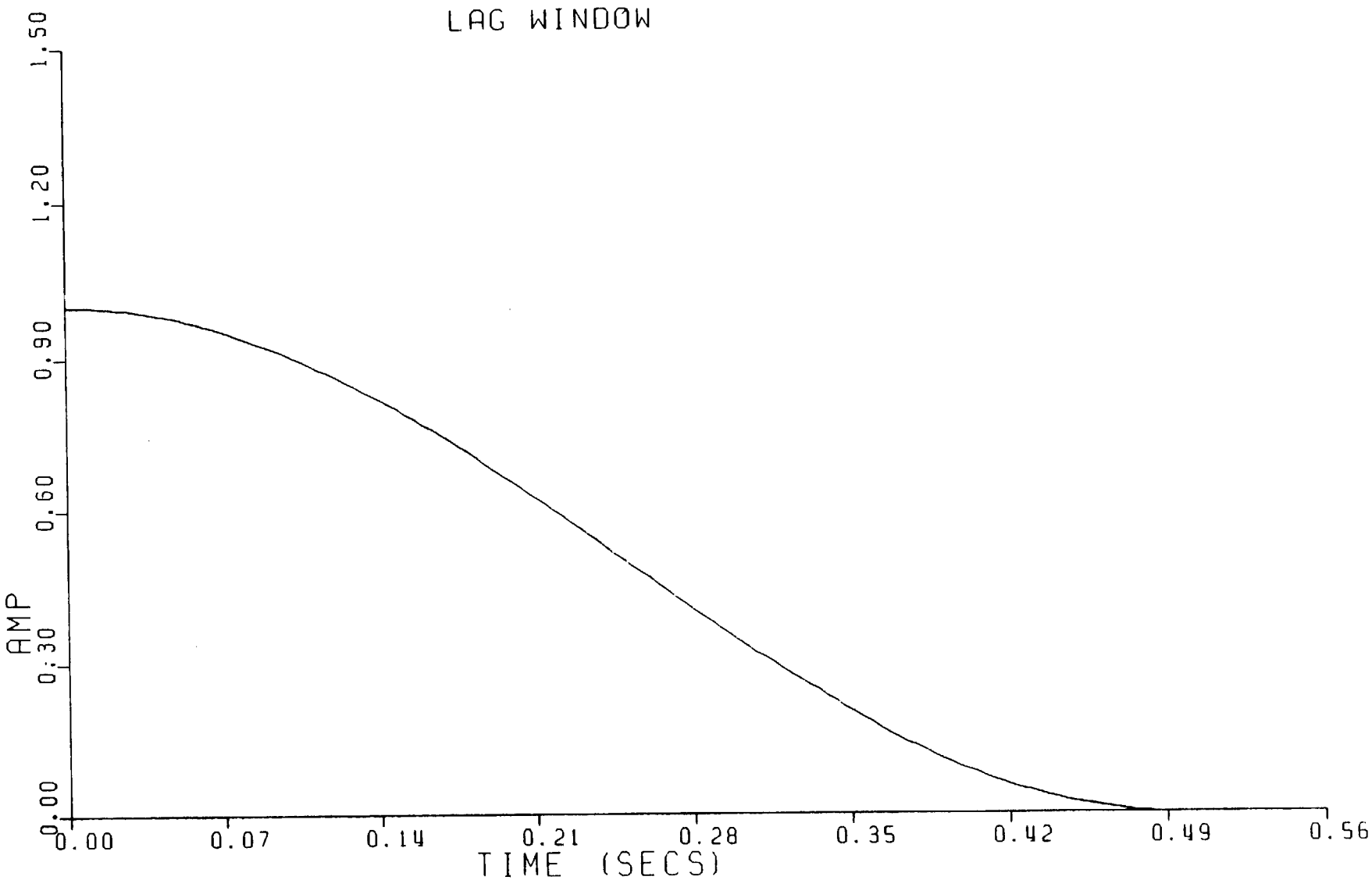


Figure 11 HAMMING

SPECTRAL WINDOW

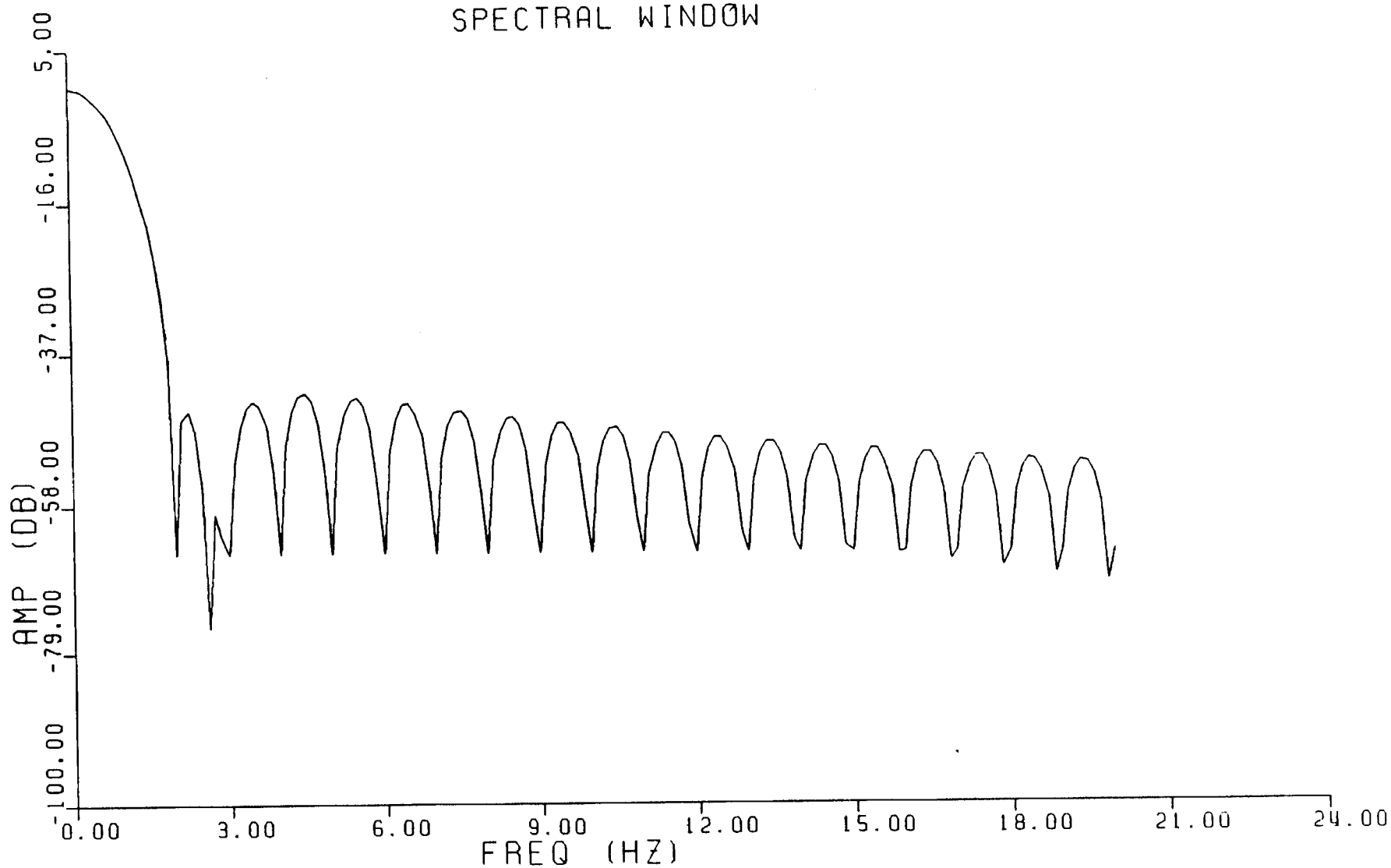


Figure 12

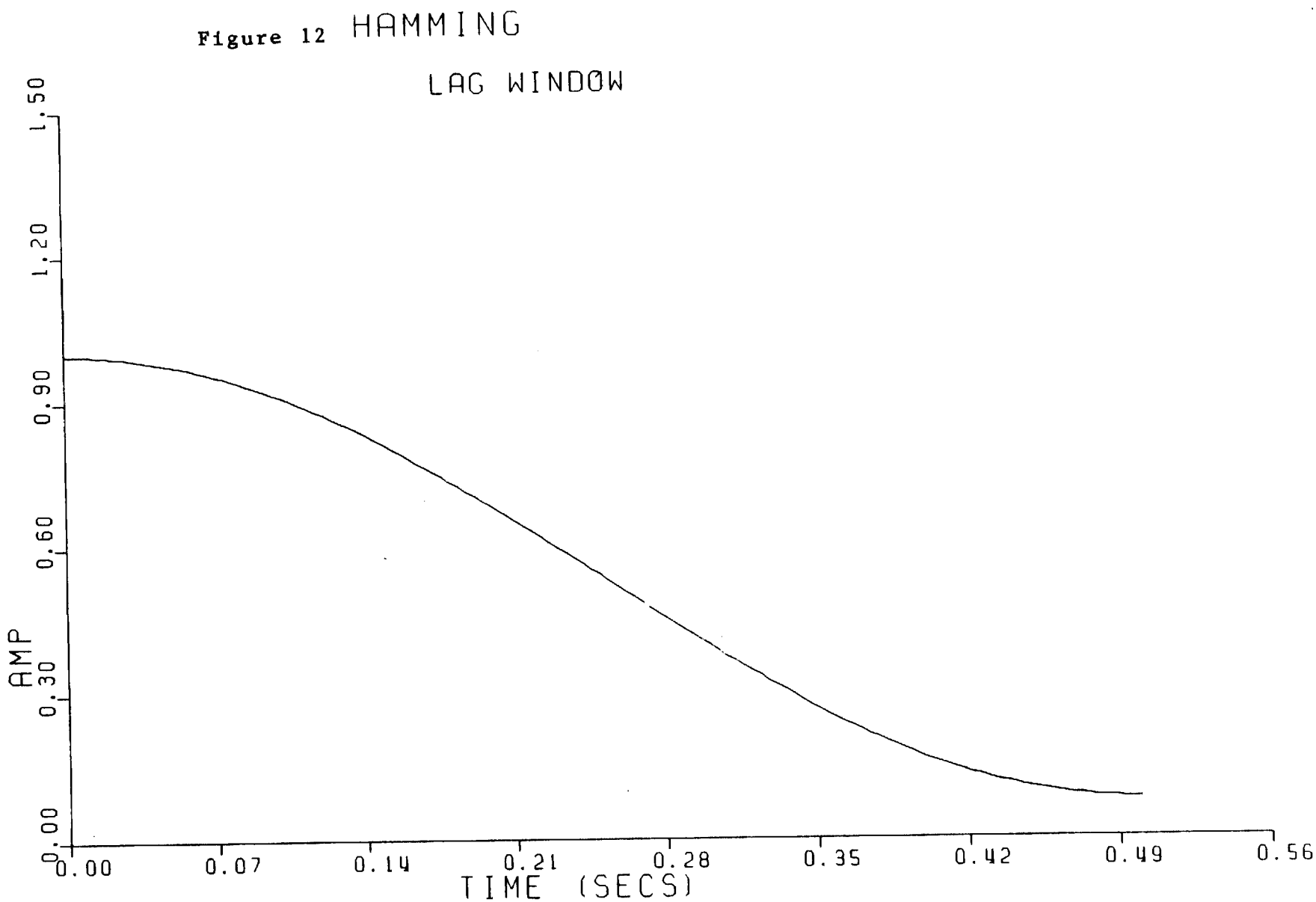

HAMMING
LAG WINDOW
AMP
0.00
0.30
0.60
0.90
1.20
1.50
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
0.56
TIME (SECS)

Figure 13

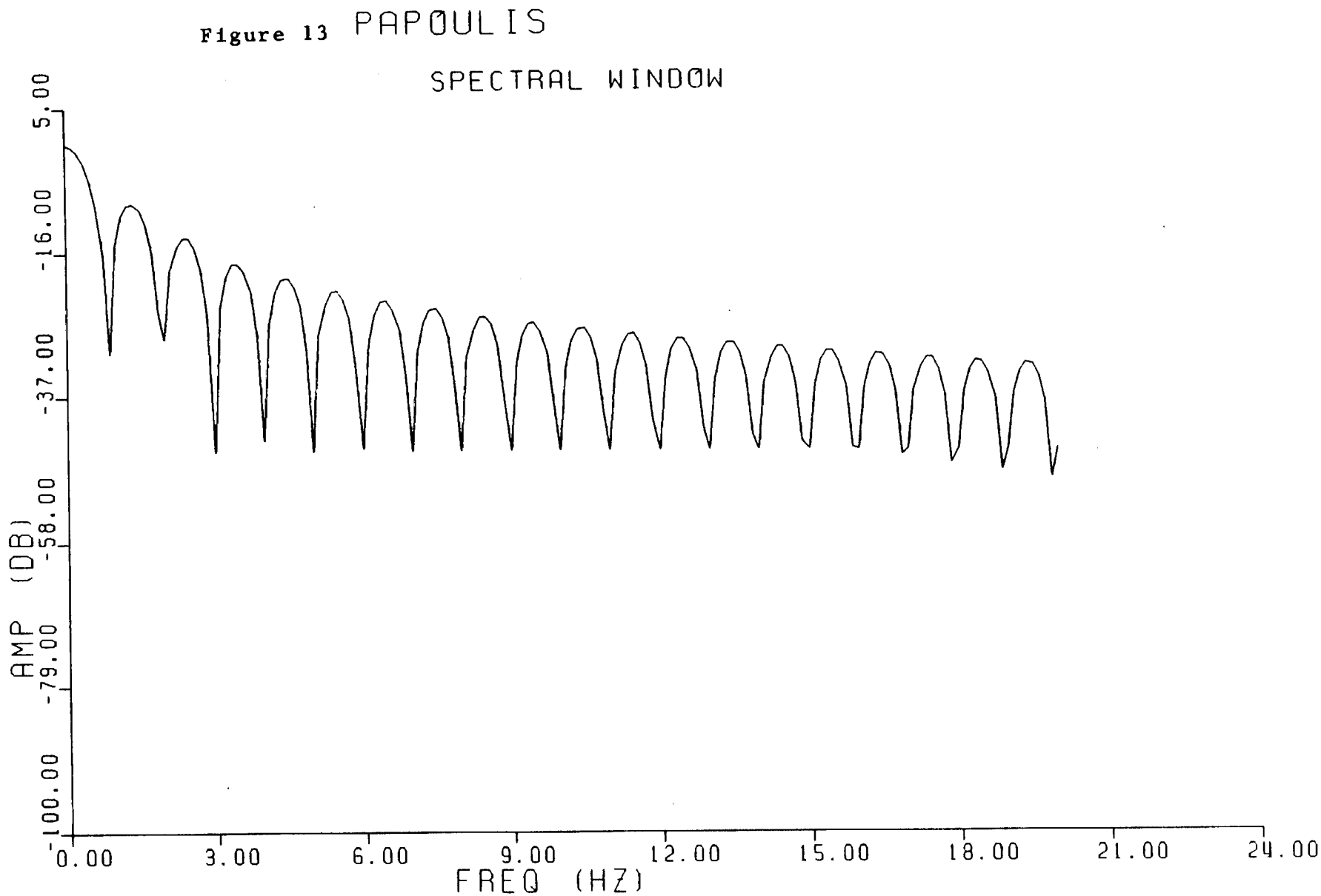

PAPOULIS
SPECTRAL WINDOW
AMP (DB)
5.00
-16.00
-37.00
-58.00
-79.00
-100.00
0.00
3.00
6.00
9.00
12.00
15.00
18.00
21.00
24.00
FREQ (HZ)

Figure 14

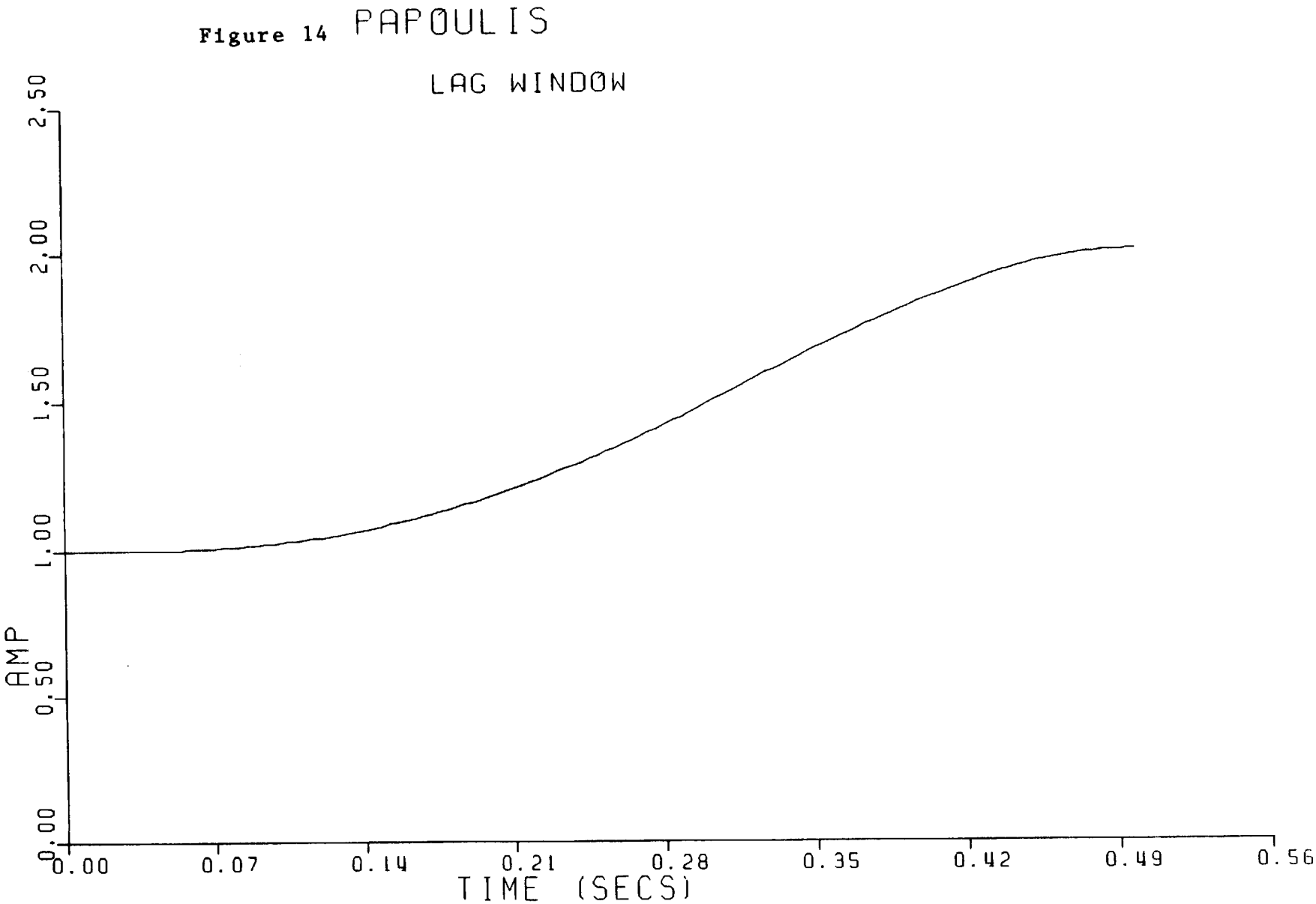

PAPOULIS
LAG WINDOW
AMP
0.00
0.50
1.00
1.50
2.00
2.50
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
0.56
TIME (SECS)

Figure 15 BLACKMAN

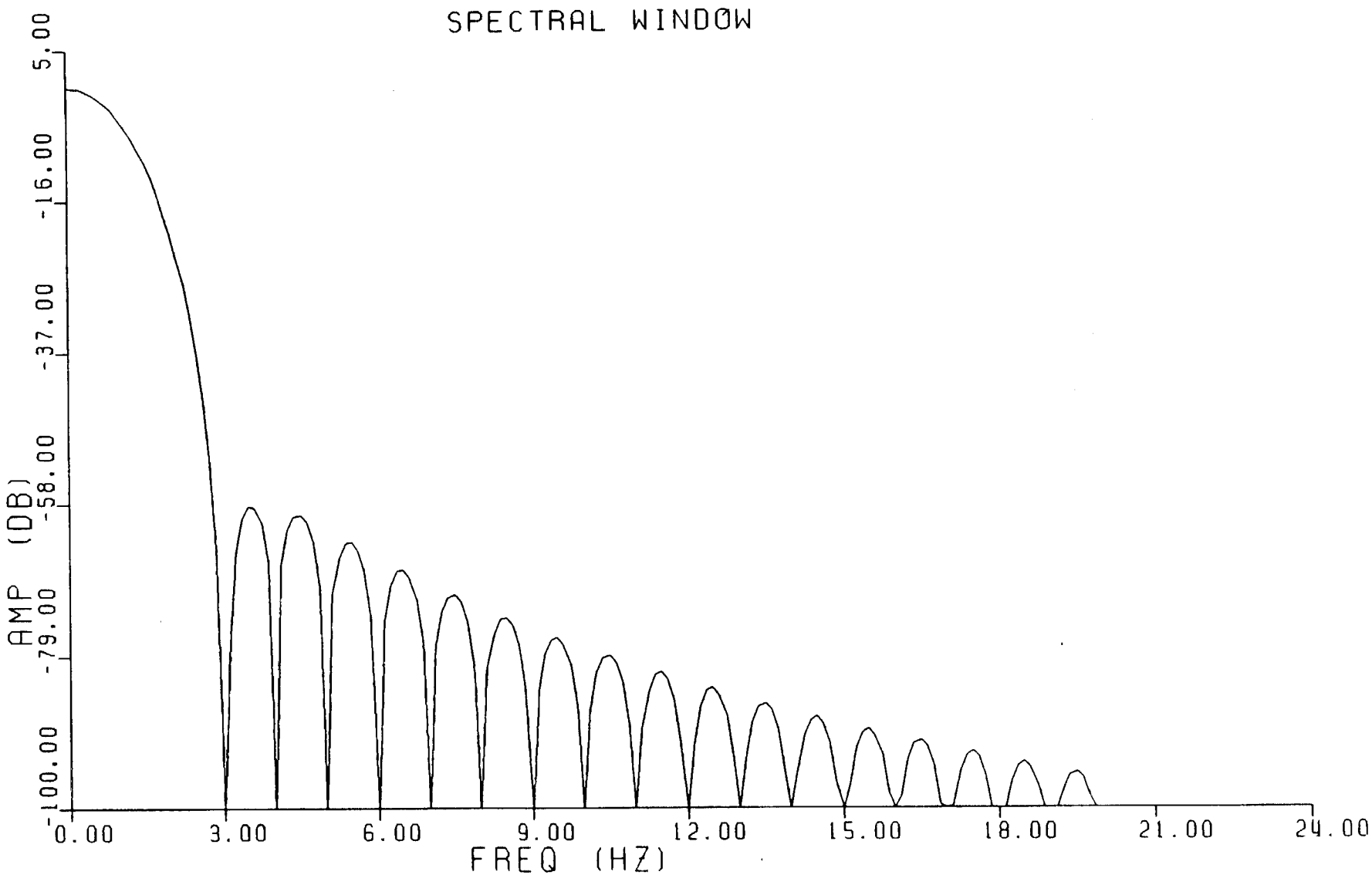


Figure 16 BLACKMAN

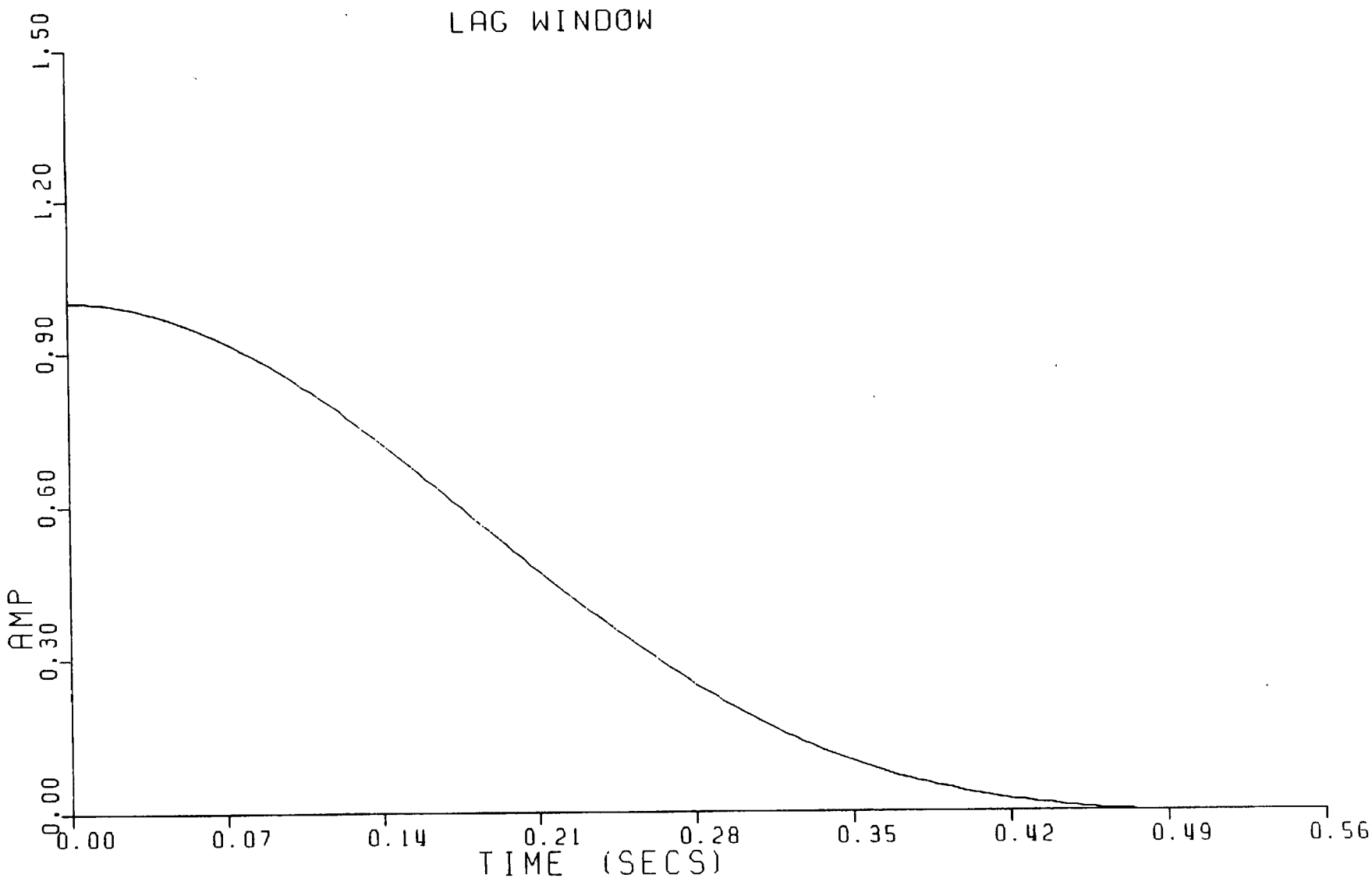


Figure 17

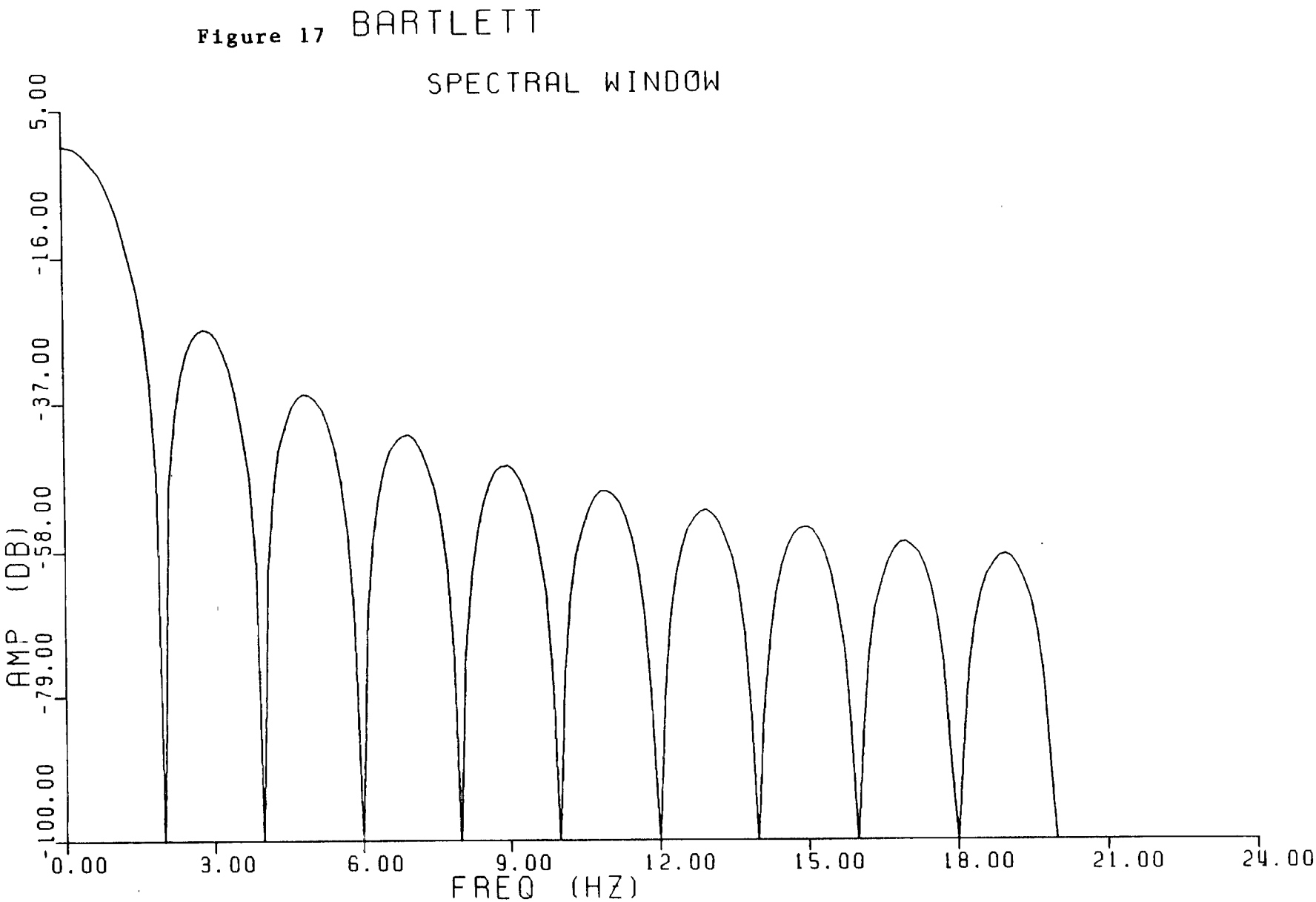

BARTLETT
SPECTRAL WINDOW
AMP (DB)
5.00
-16.00
-37.00
-58.00
-79.00
-100.00
0.00
3.00
6.00
9.00
12.00
15.00
18.00
21.00
24.00
FREQ (HZ)

Figure 18 BARTLETT

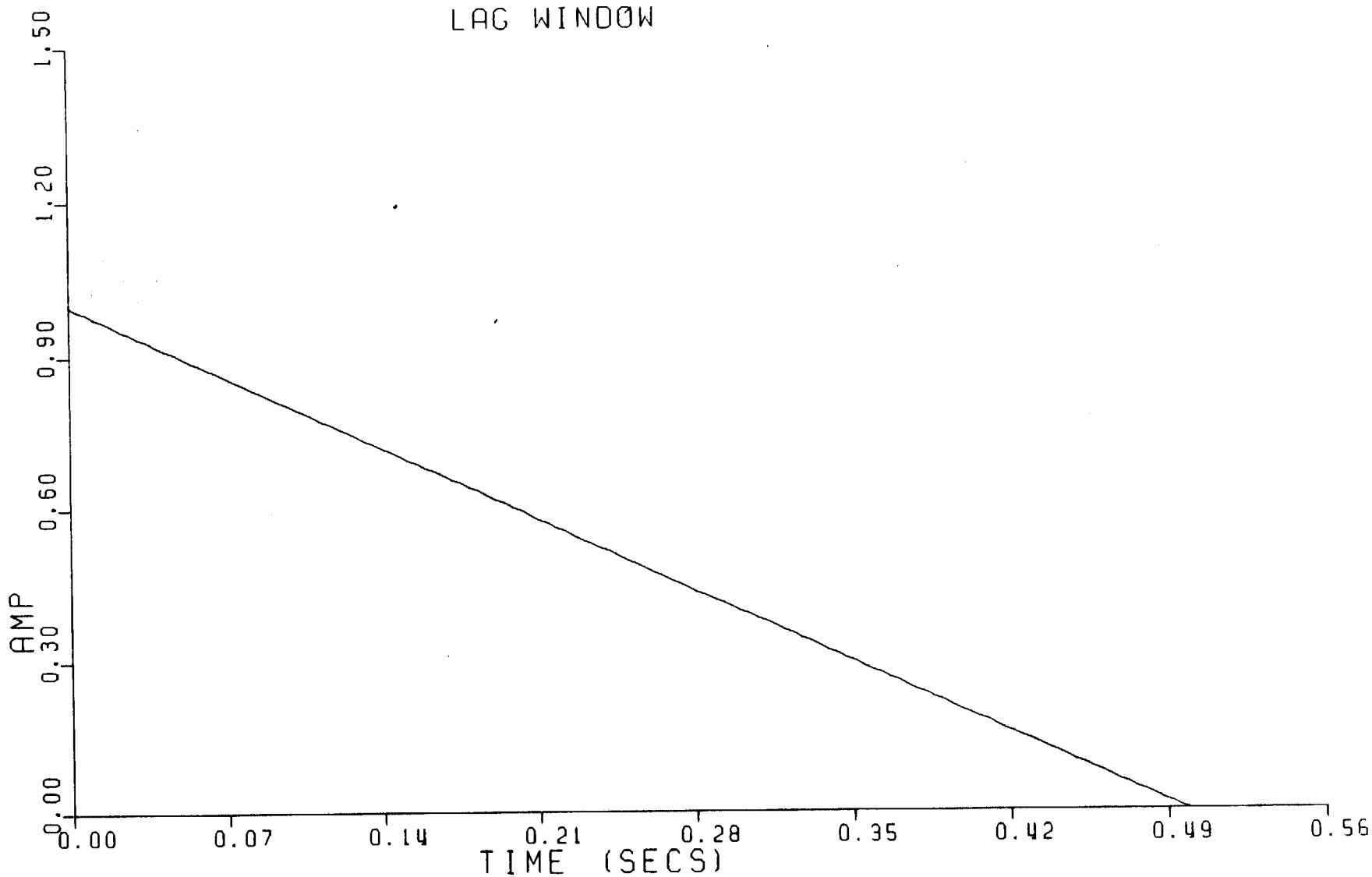


Figure 19 SINC-LIKE

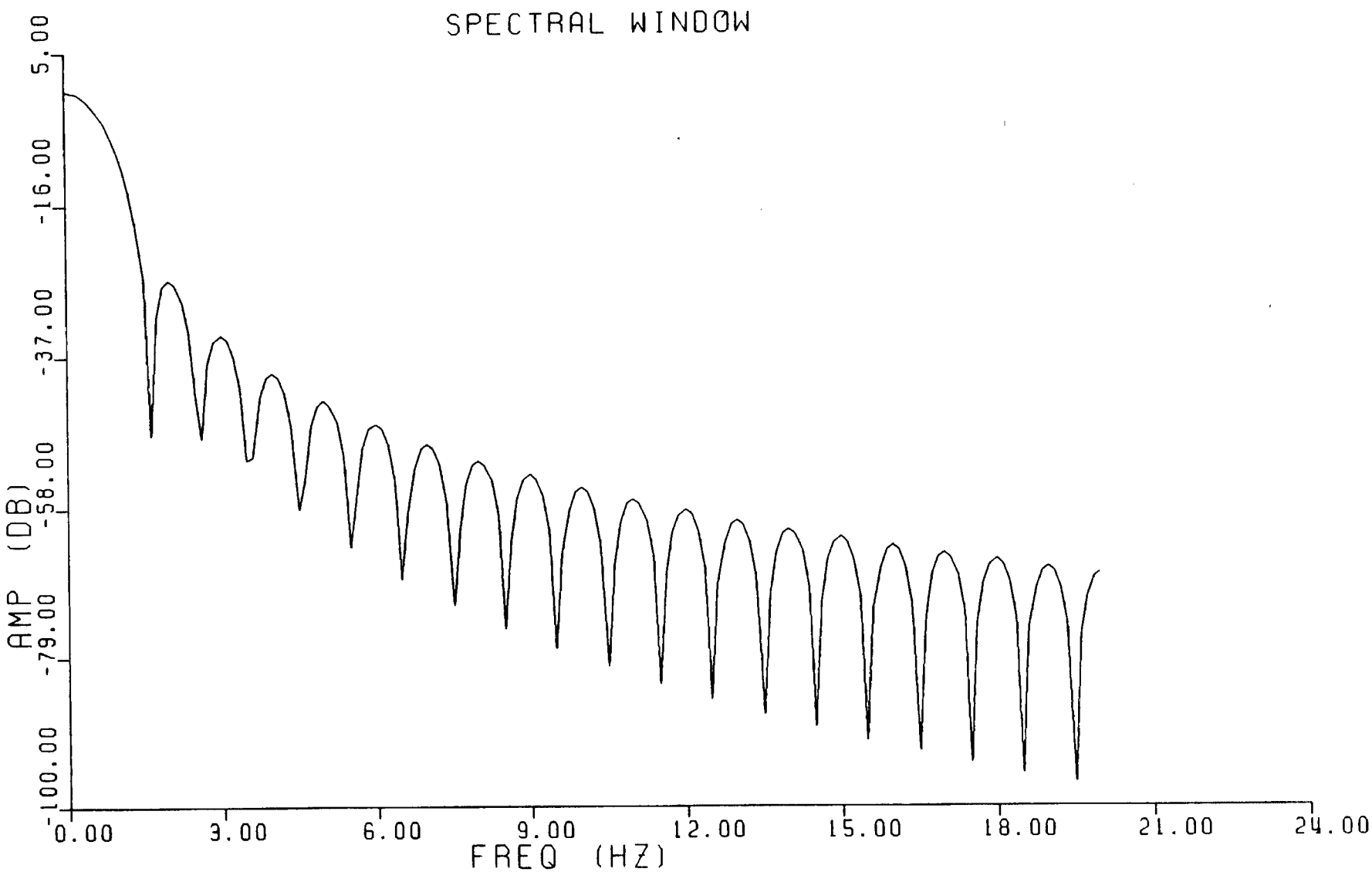


Figure 20 SINC-LIKE

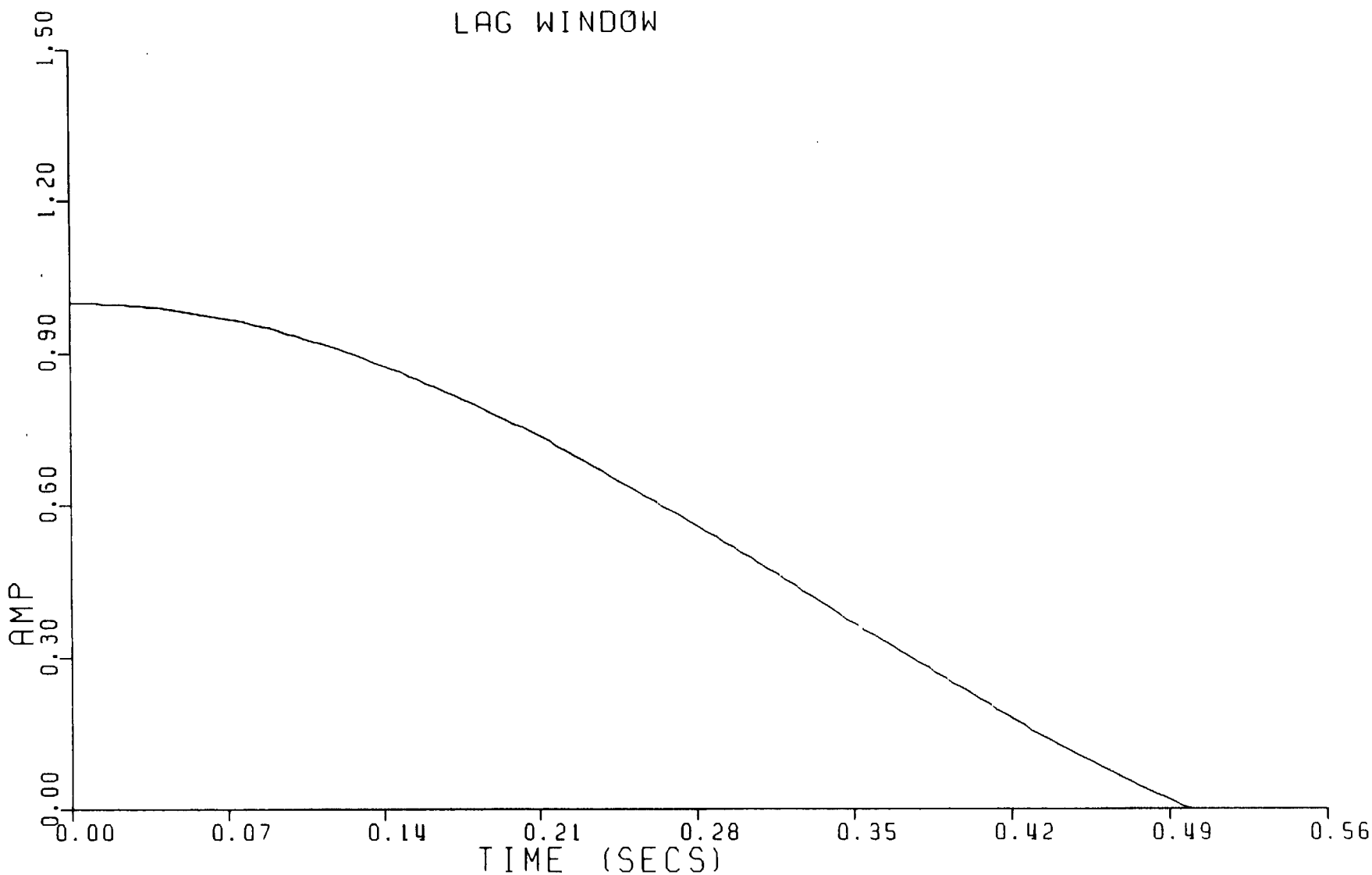


Figure 21

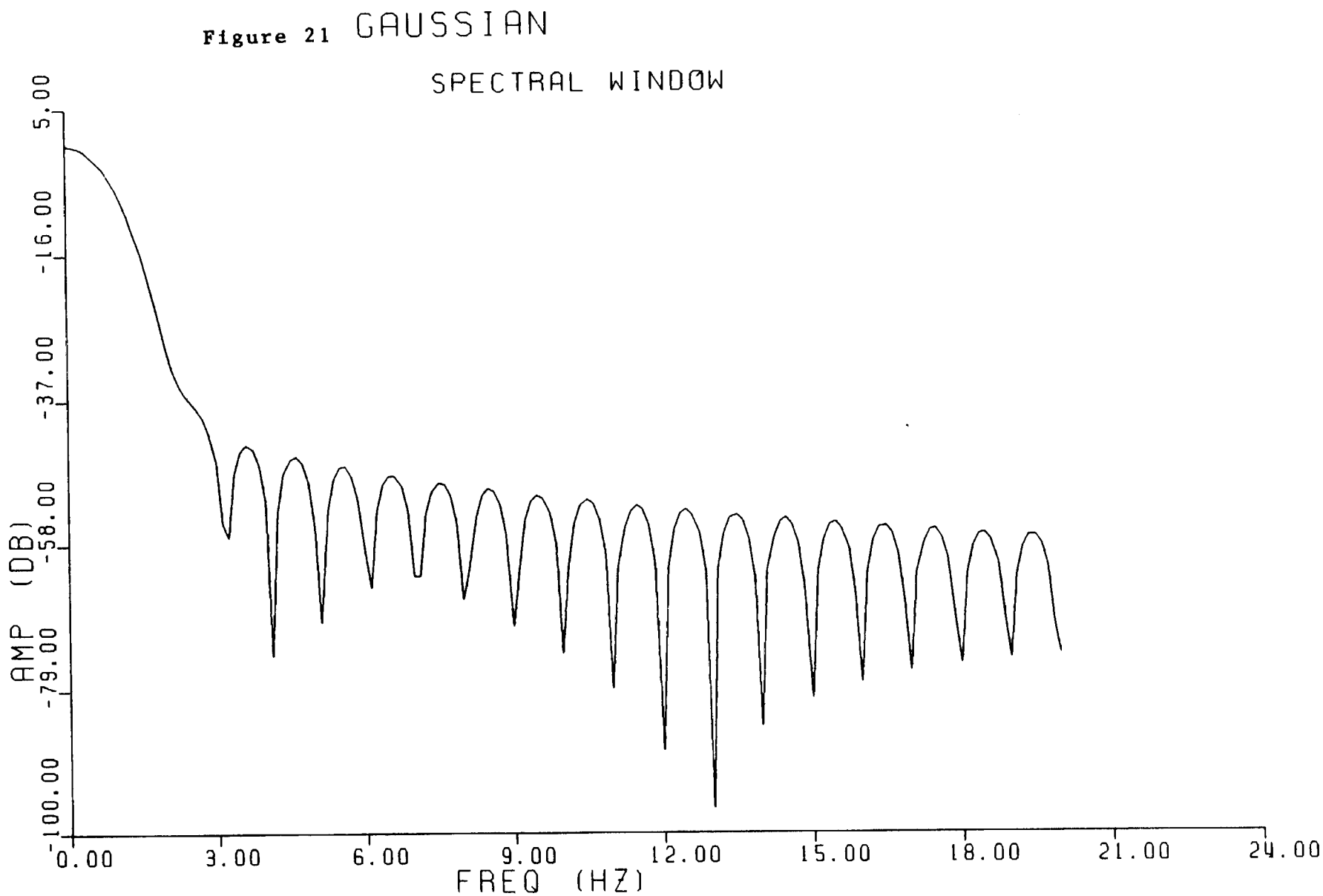

GAUSSIAN
SPECTRAL WINDOW
AMP (DB)
5.00
-16.00
-37.00
-58.00
-79.00
-100.00
0.00
3.00
6.00
9.00
12.00
15.00
18.00
21.00
24.00
FREQ (HZ)

Figure 22

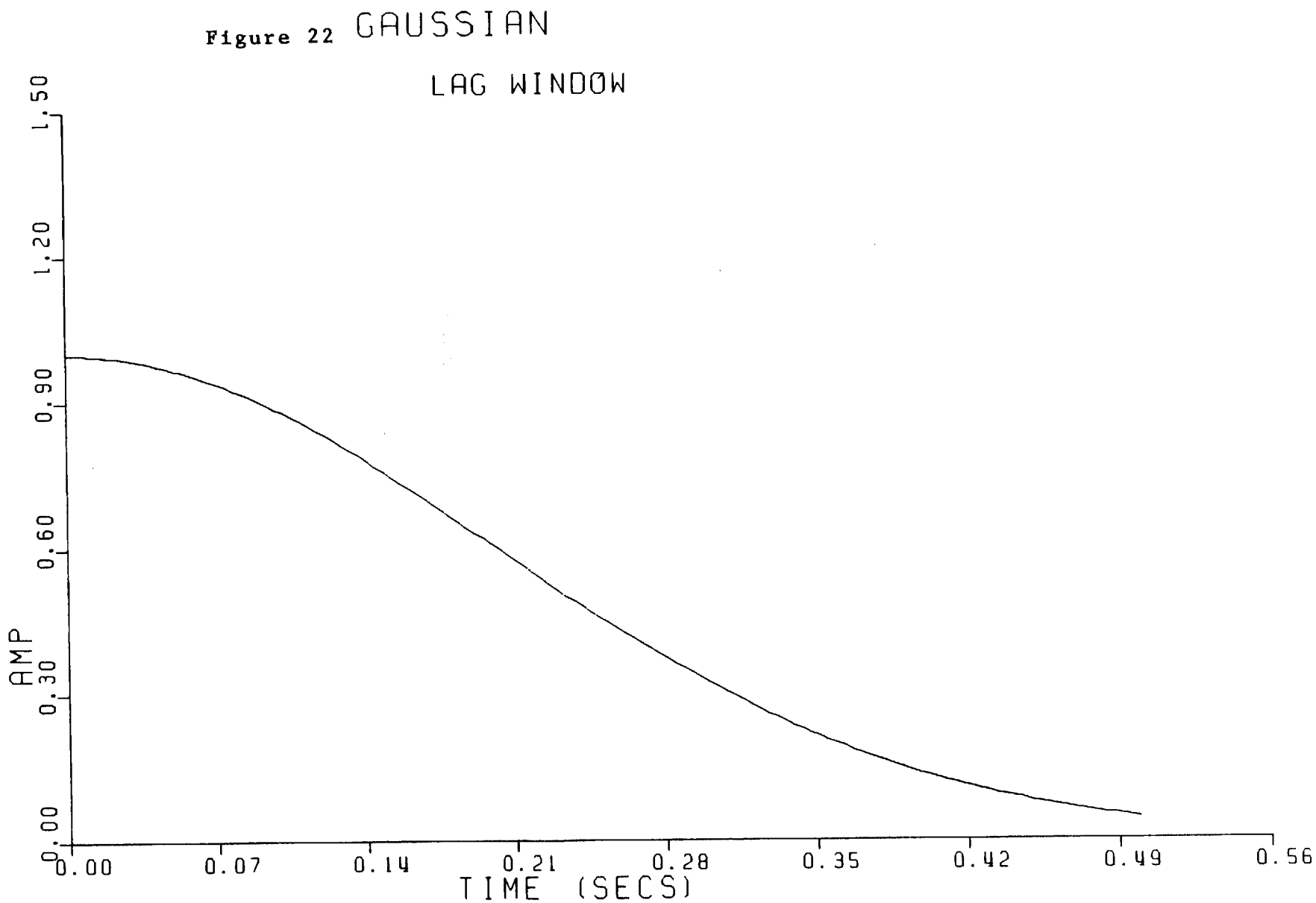

GAUSSIAN
LAG WINDOW
AMP
0.00
0.30
0.60
0.90
1.20
1.50
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
0.56
TIME (SECS)

Figure 23 COSINE SPEC WINDOW

L = 0

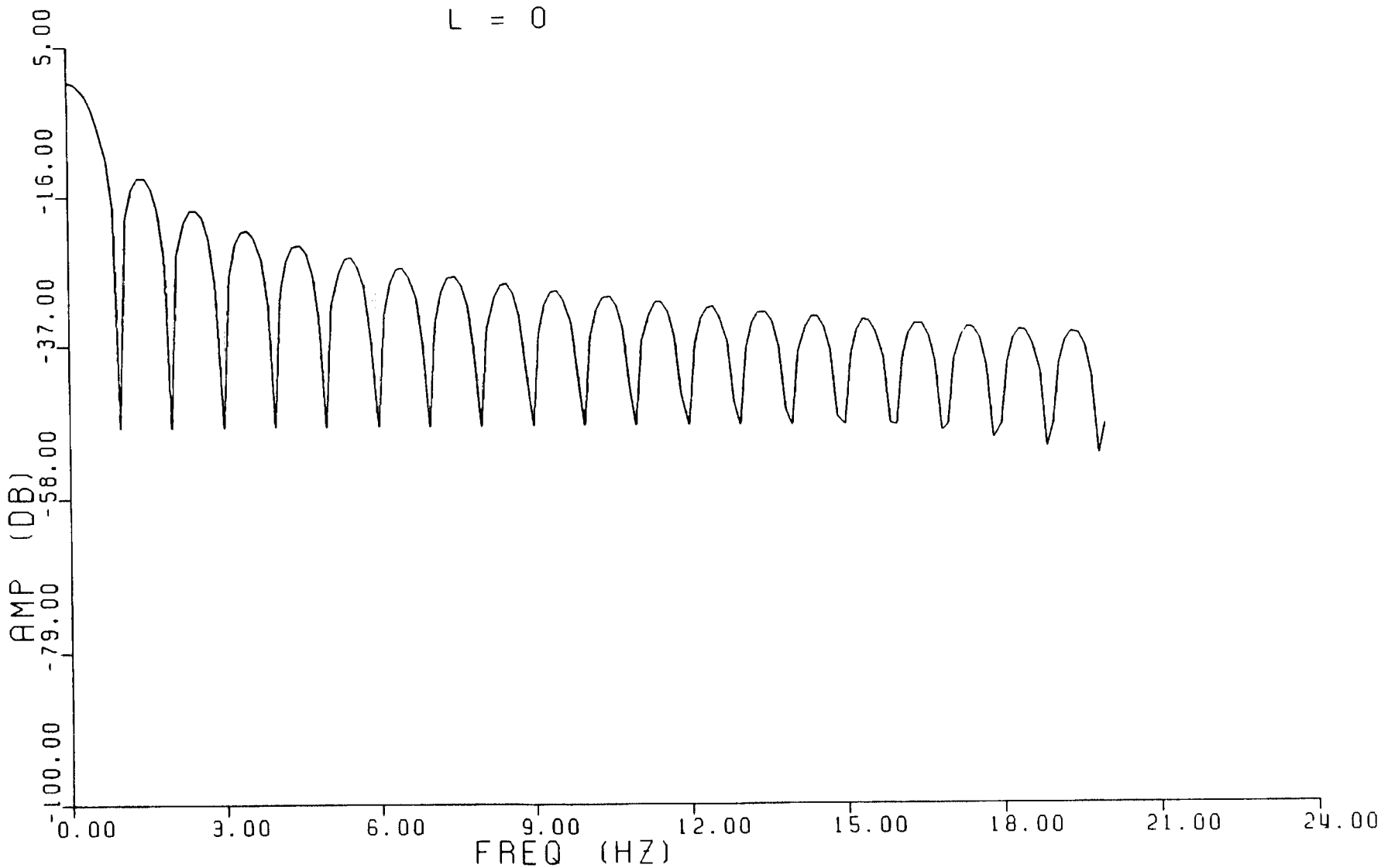


Figure 24 COSINE LAG WINDOW

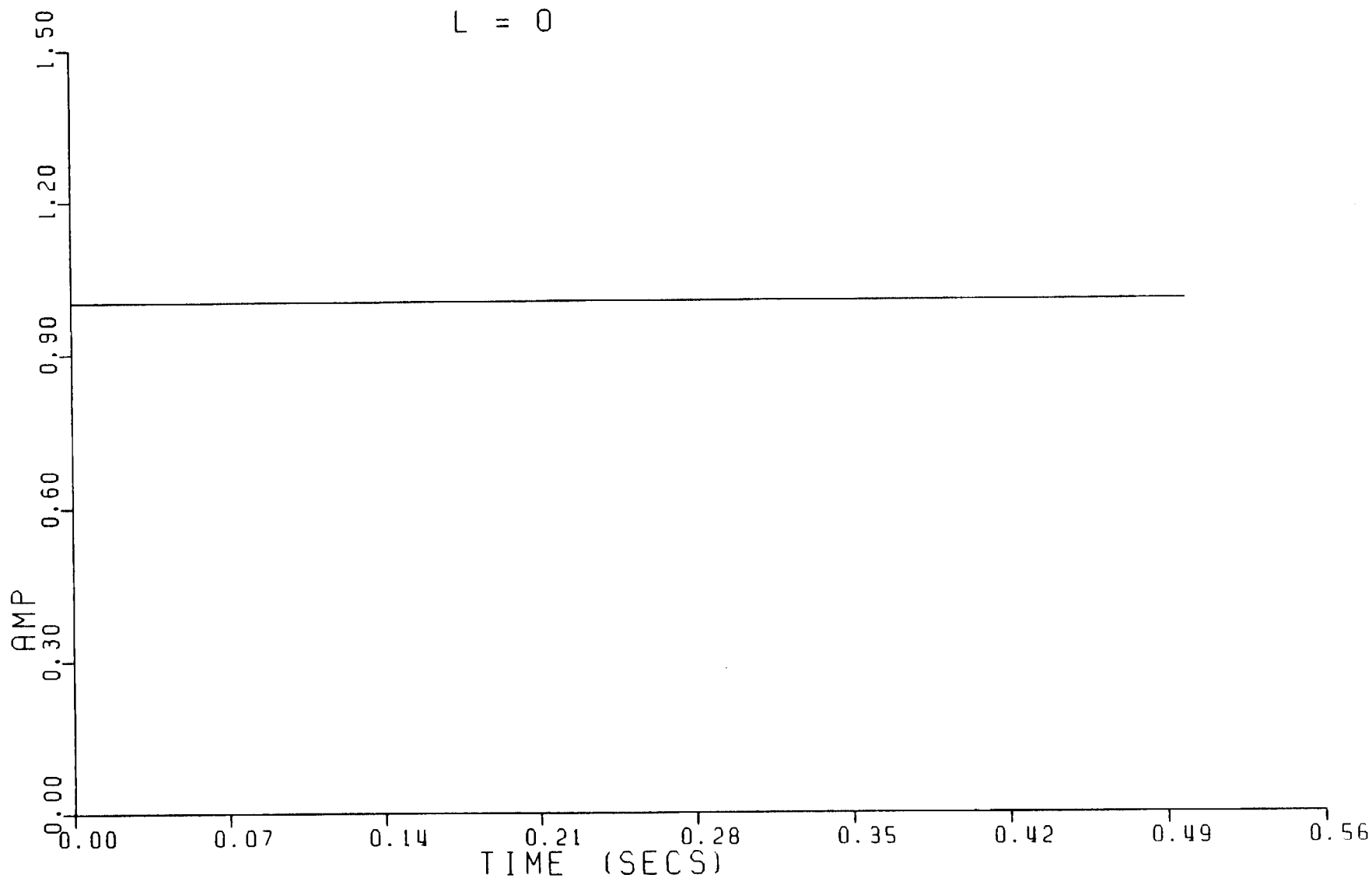


Figure 25 COSINE SPEC WINDOW

L = 1

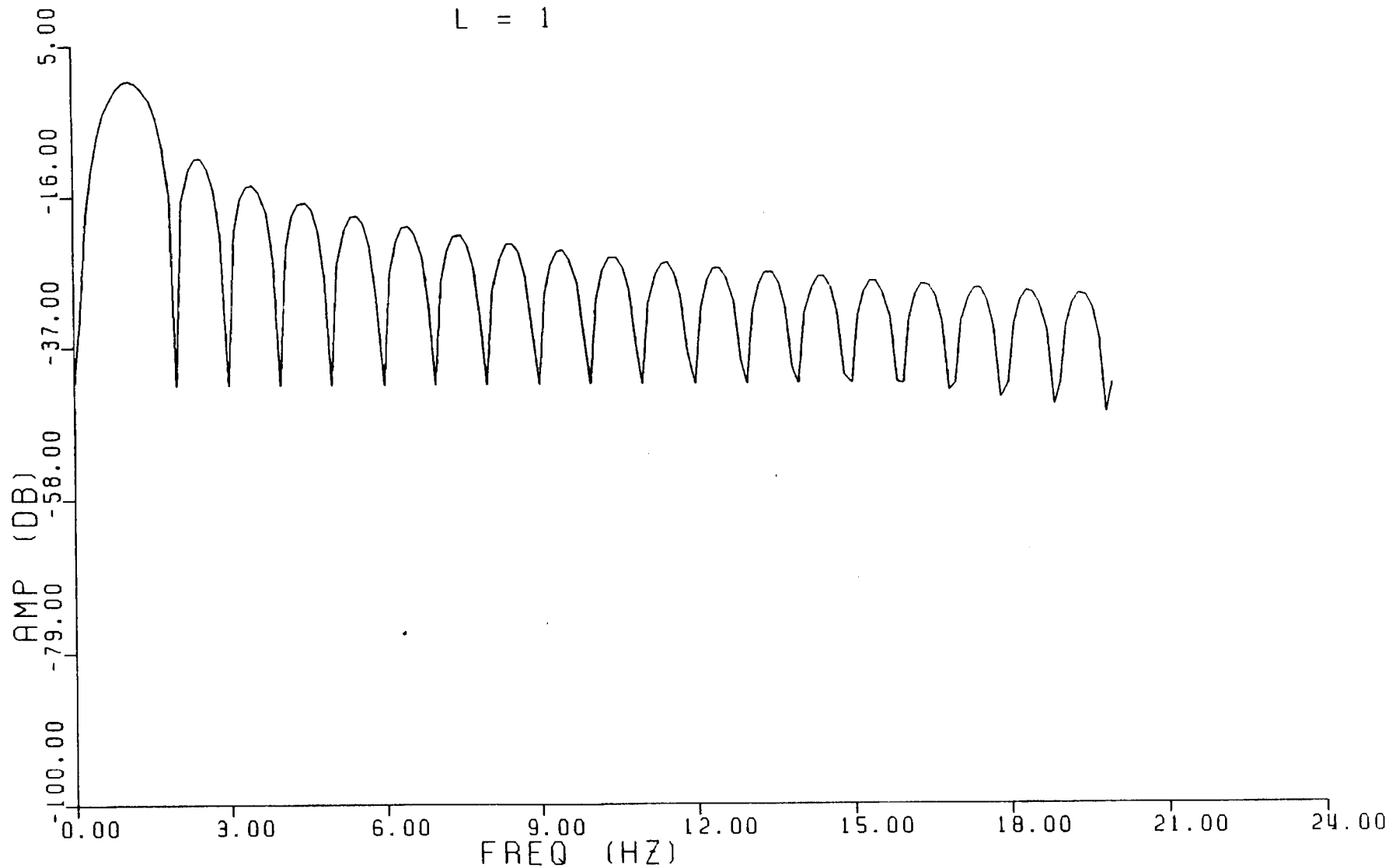


Figure 26 COSINE LAG WINDOW

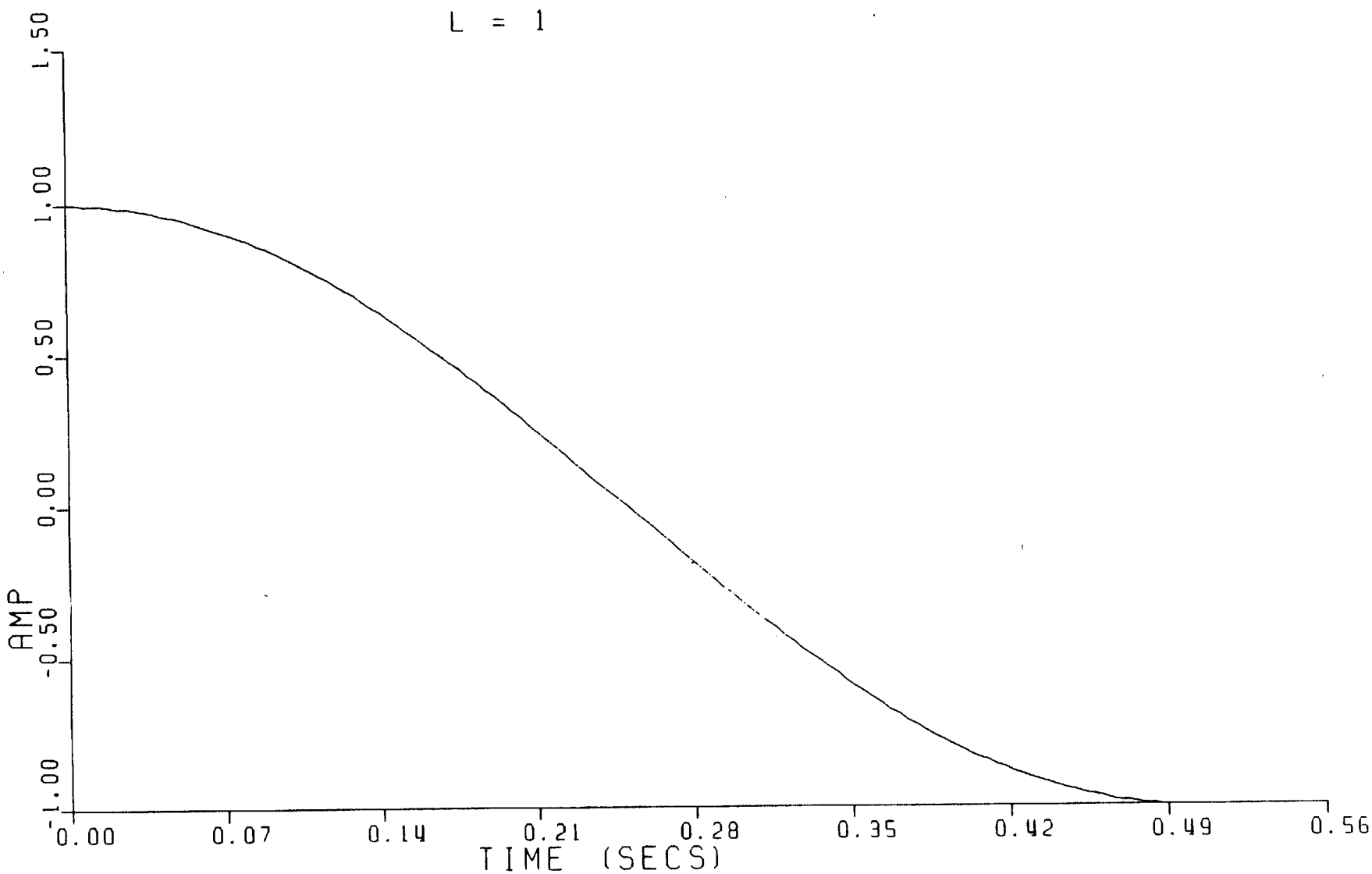


Figure 27

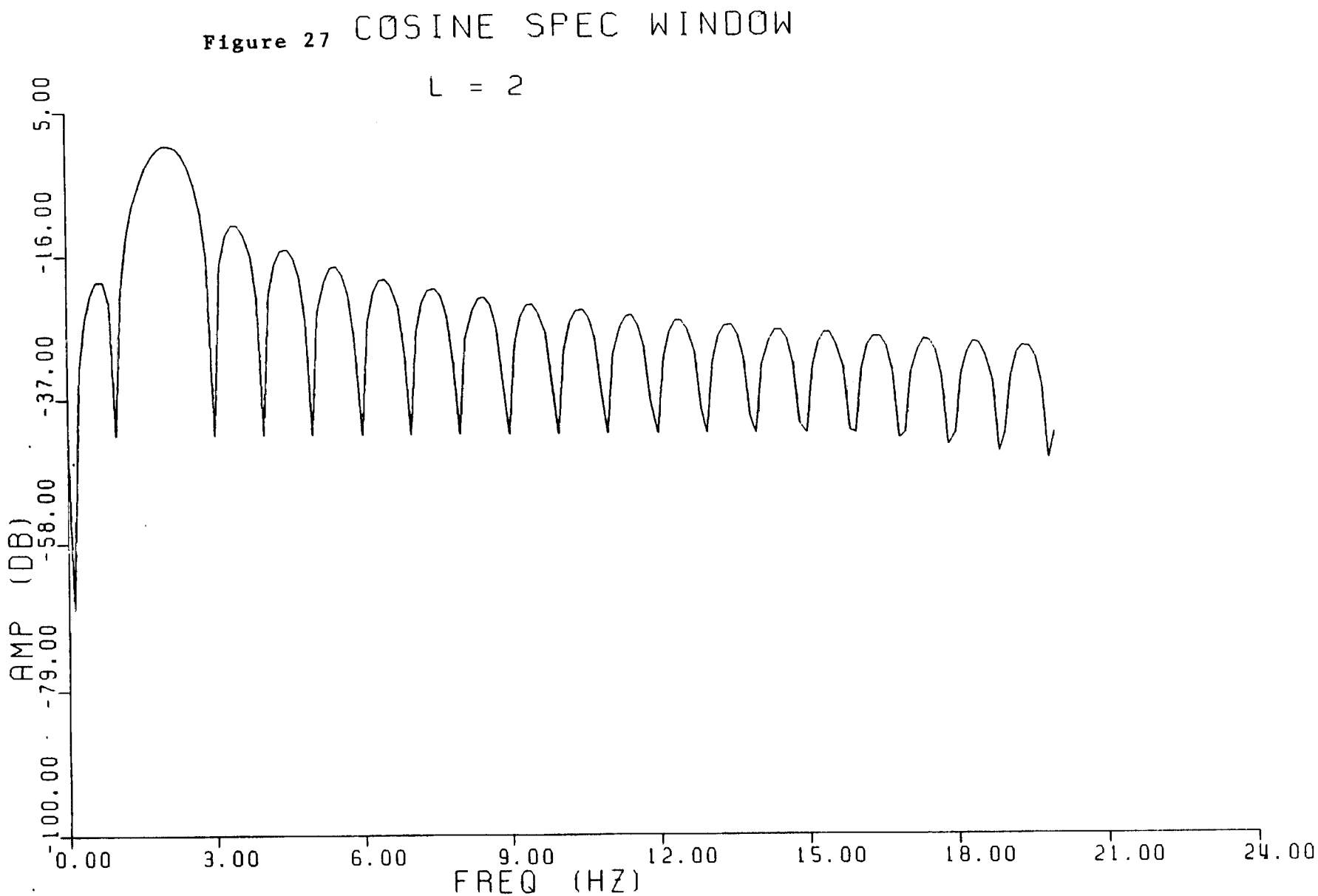

COSINE SPEC WINDOW
L = 2
AMP (DB)
5.00
-16.00
-37.00
-58.00
-79.00
-100.00
0.00
3.00
6.00
9.00
12.00
15.00
18.00
21.00
24.00
FREQ (HZ)

Figure 28 COSINE LAG WINDOW

L = 2

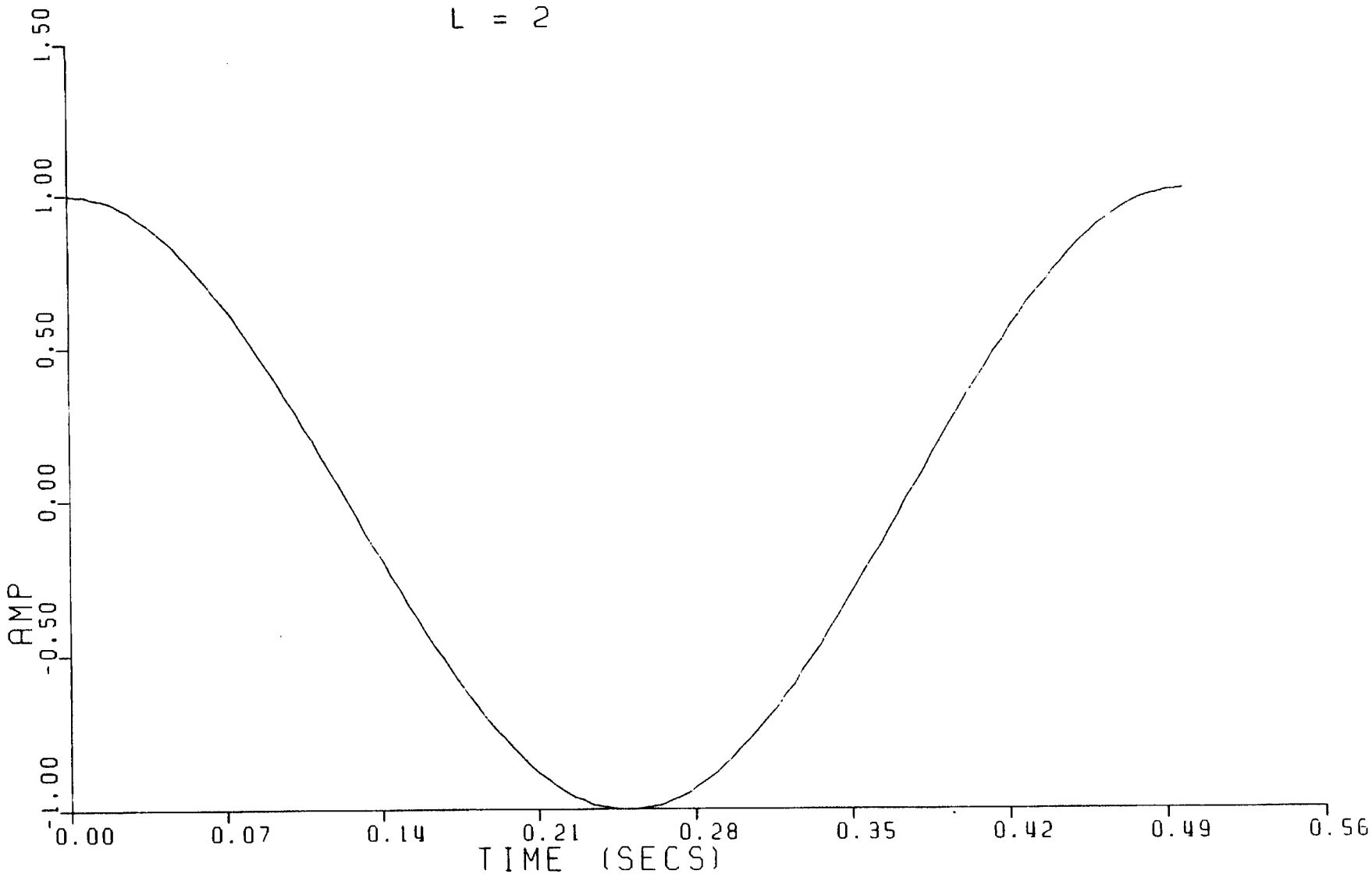


Figure 29

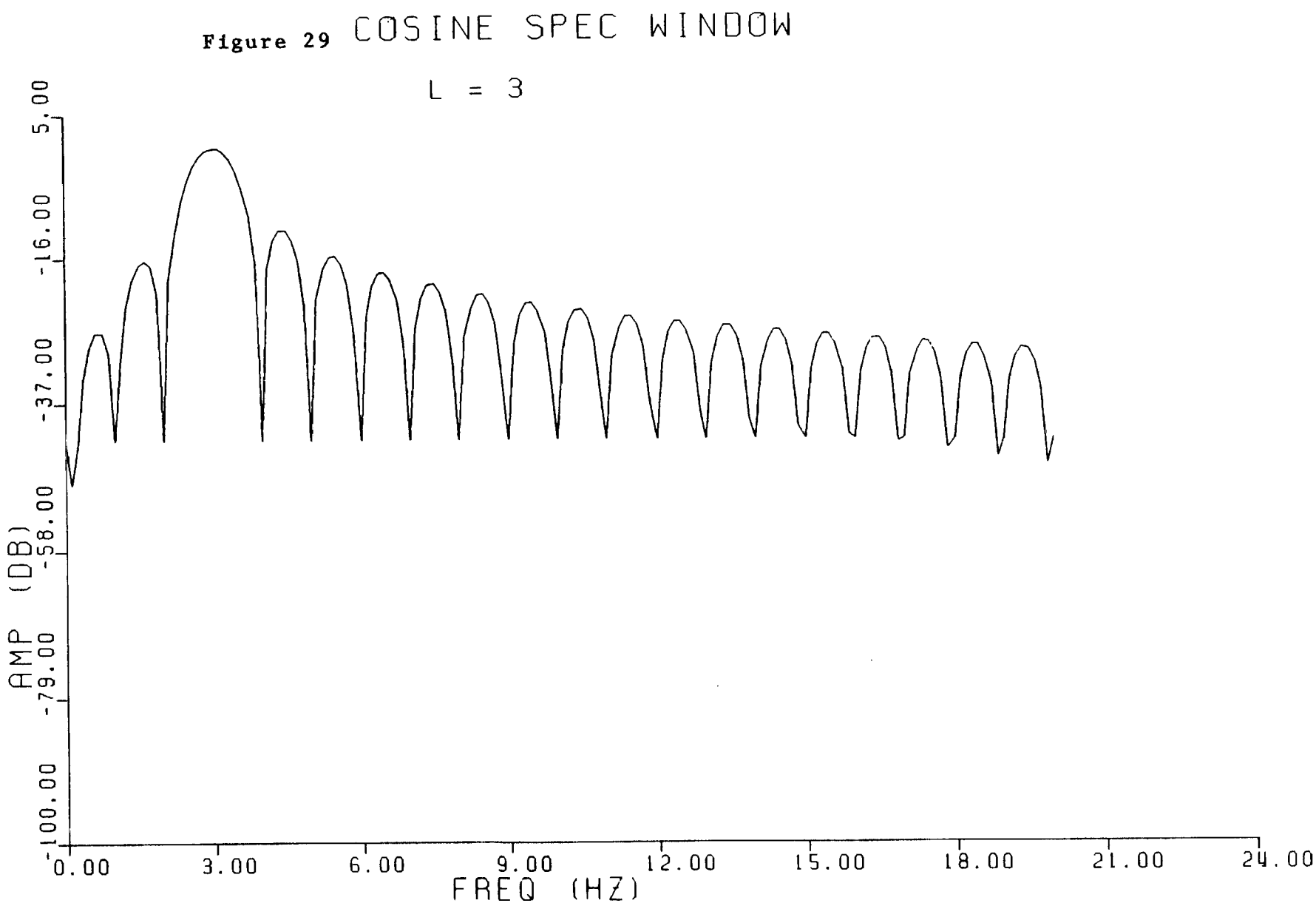

COSINE SPEC WINDOW
L = 3
AMP (DB)
5.00
-16.00
-37.00
-58.00
-79.00
-100.00
0.00
3.00
6.00
9.00
12.00
15.00
18.00
21.00
24.00
FREQ (HZ)

Figure 30 COSINE LAG WINDOW

L = 3

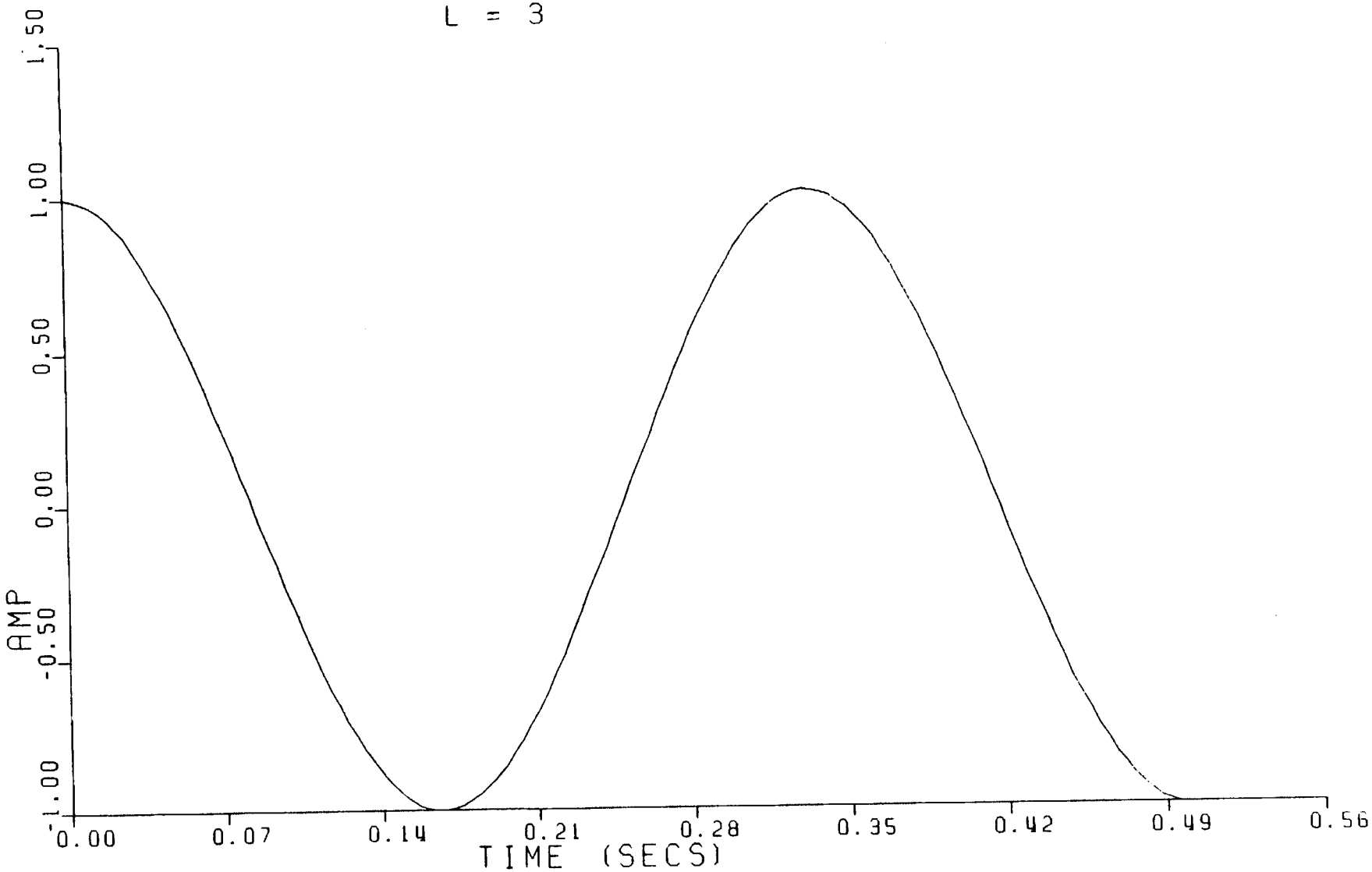


Figure 31

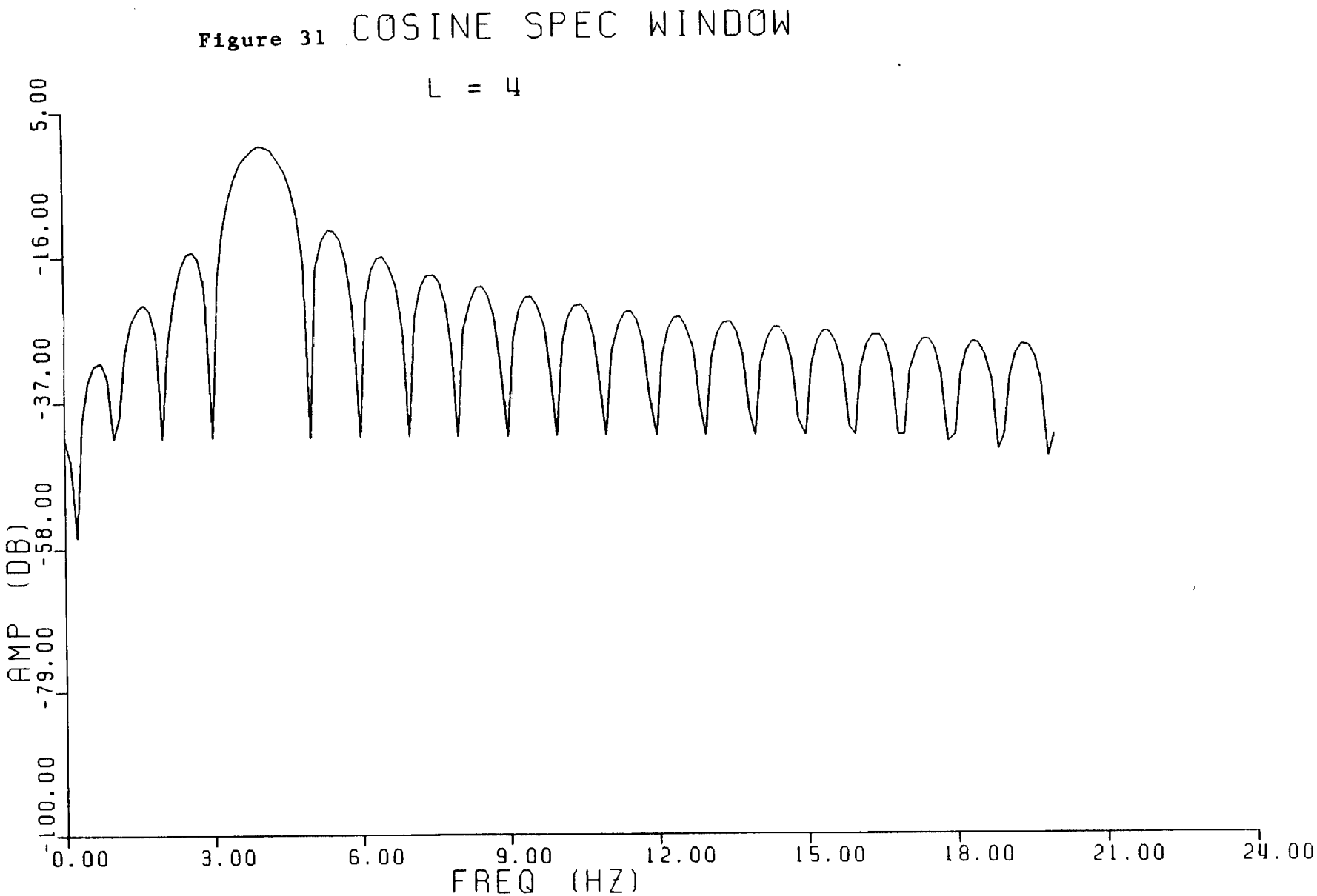

COSINE SPEC WINDOW
L = 4
AMP (DB)
5.00
-16.00
-37.00
-58.00
-79.00
-100.00
0.00
3.00
6.00
9.00
12.00
15.00
18.00
21.00
24.00
FREQ (HZ)

Figure 32

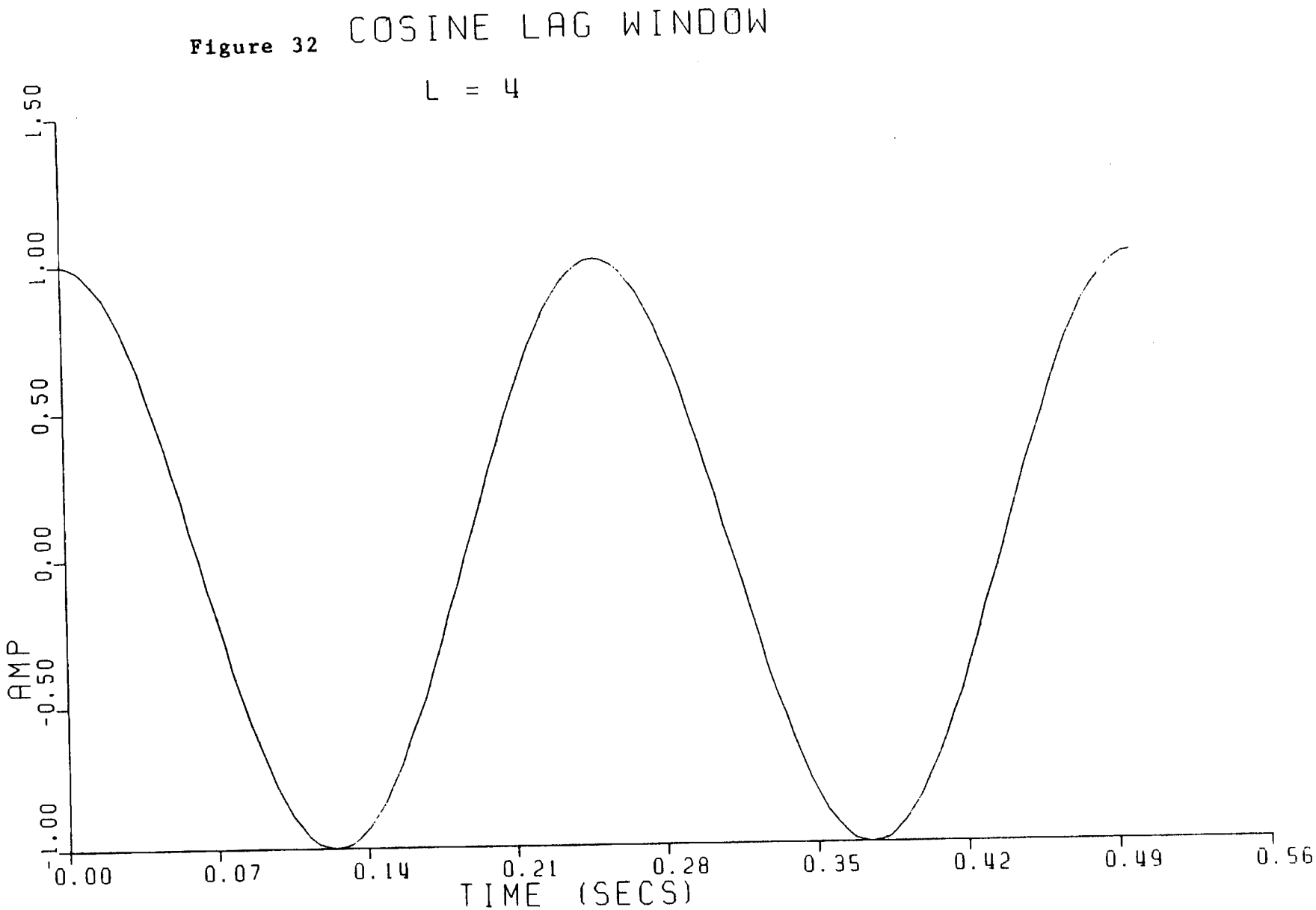

COSINE LAG WINDOW
L = 4
AMP
1.50
1.00
0.50
0.00
-0.50
-1.00
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
0.56
TIME (SECS)

Figure 33

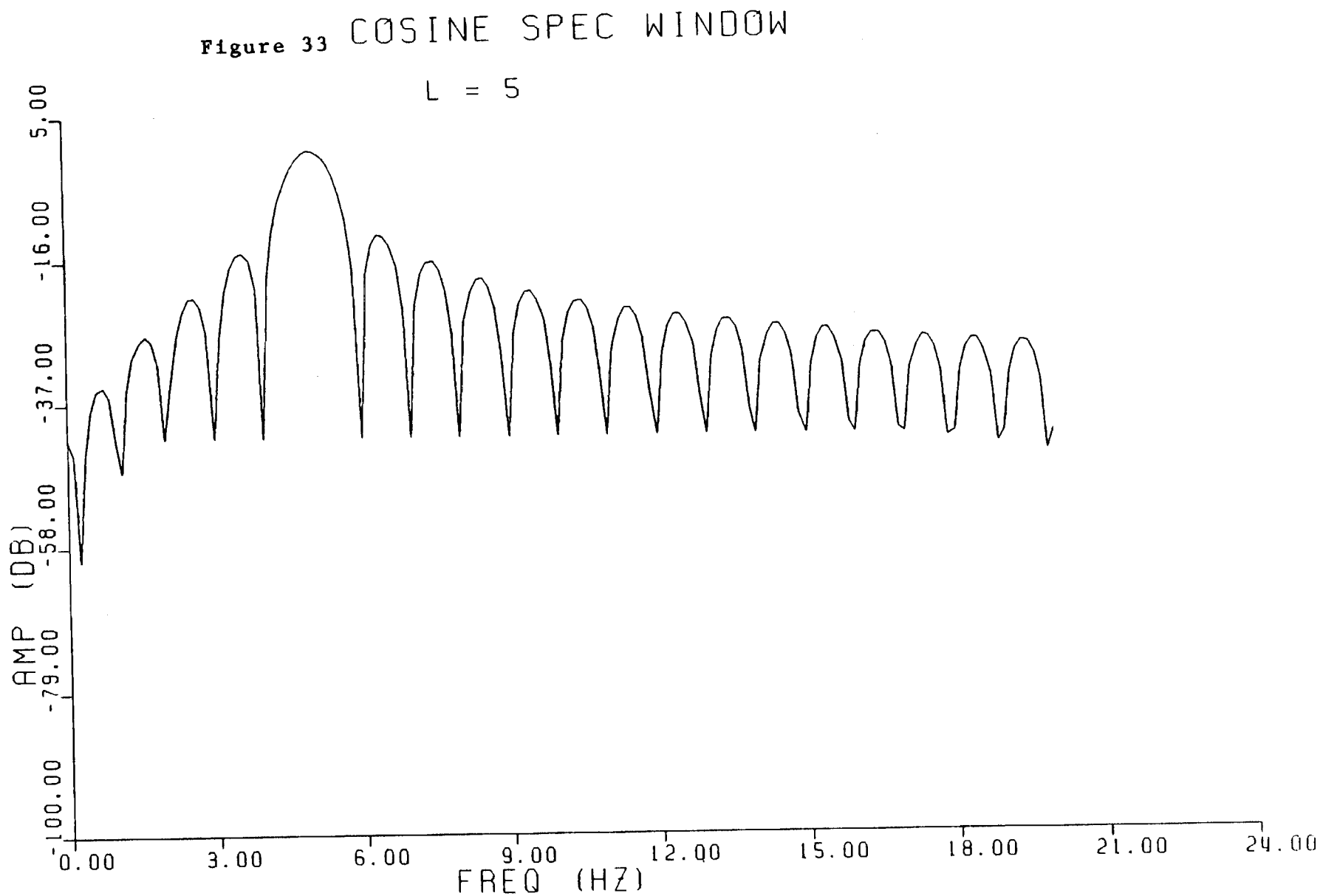

COSINE SPEC WINDOW
L = 5
AMP (DB)
5.00
-16.00
-37.00
-58.00
-79.00
-100.00
0.00
3.00
6.00
9.00
12.00
15.00
18.00
21.00
24.00
FREQ (HZ)

Figure 34

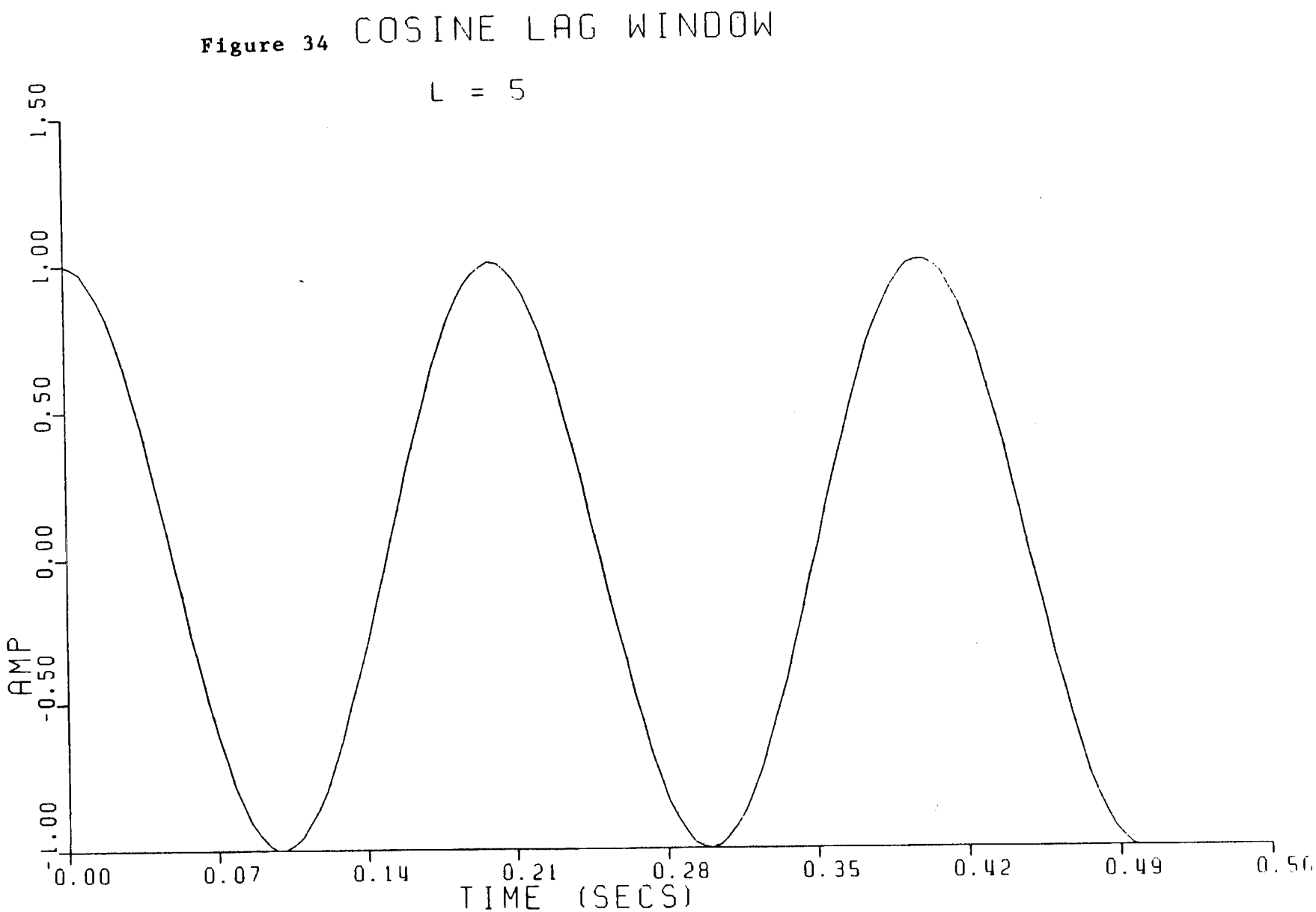

COSINE LAG WINDOW
L = 5
AMP
1.50
1.00
0.50
0.00
-0.50
-1.00
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
TIME (SECS)

Figure 35

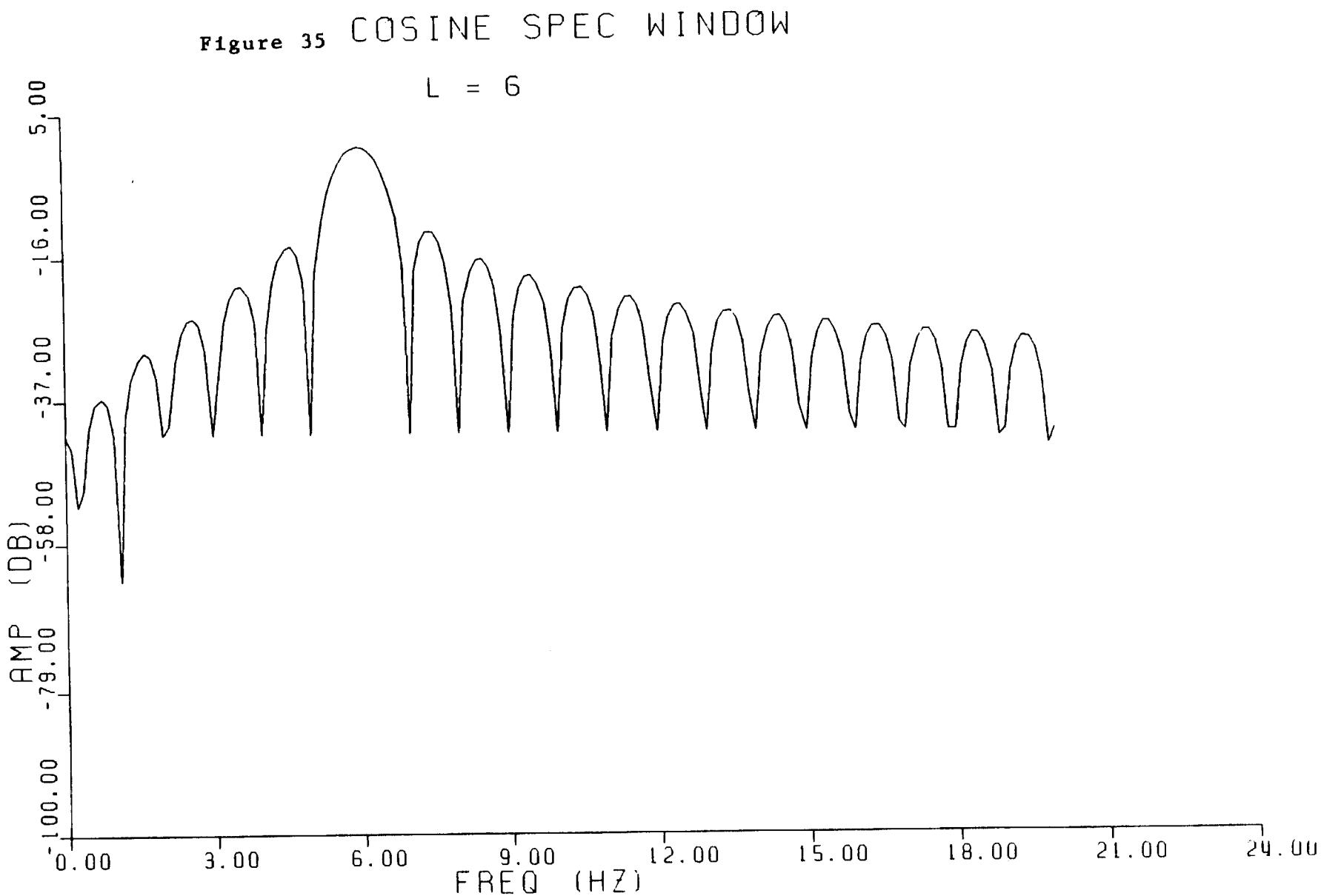

COSINE SPEC WINDOW
L = 6
AMP (DB)
5.00
-16.00
-37.00
-58.00
-79.00
-100.00
0.00
3.00
6.00
9.00
12.00
15.00
18.00
21.00
24.00
FREQ (HZ)

Figure 36

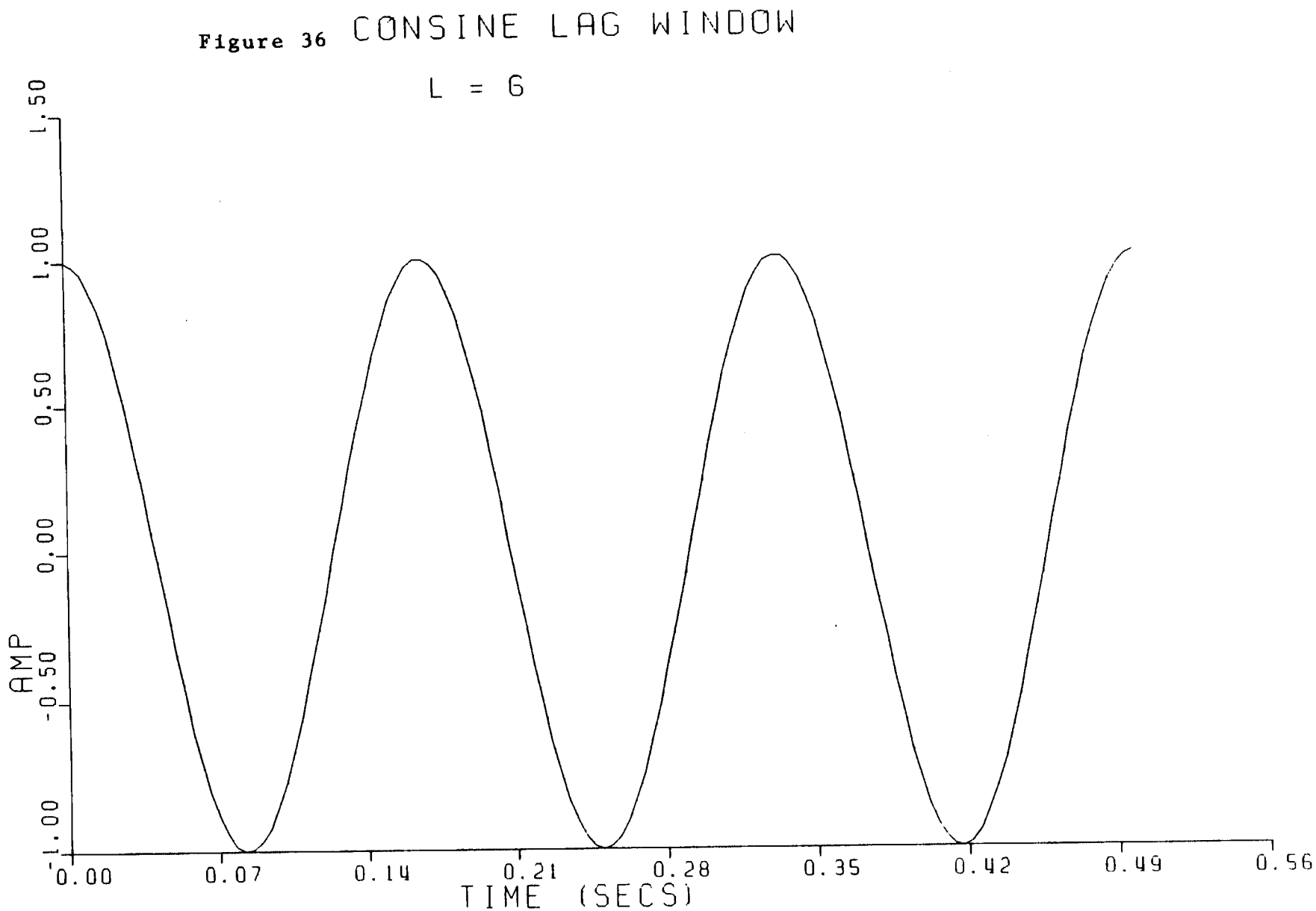

CONSINE LAG WINDOW
L = 6
AMP
1.50
1.00
0.50
0.00
-0.50
-1.00
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
0.56
TIME (SECS)

Figure 37 COSINE SPEC WINDOW

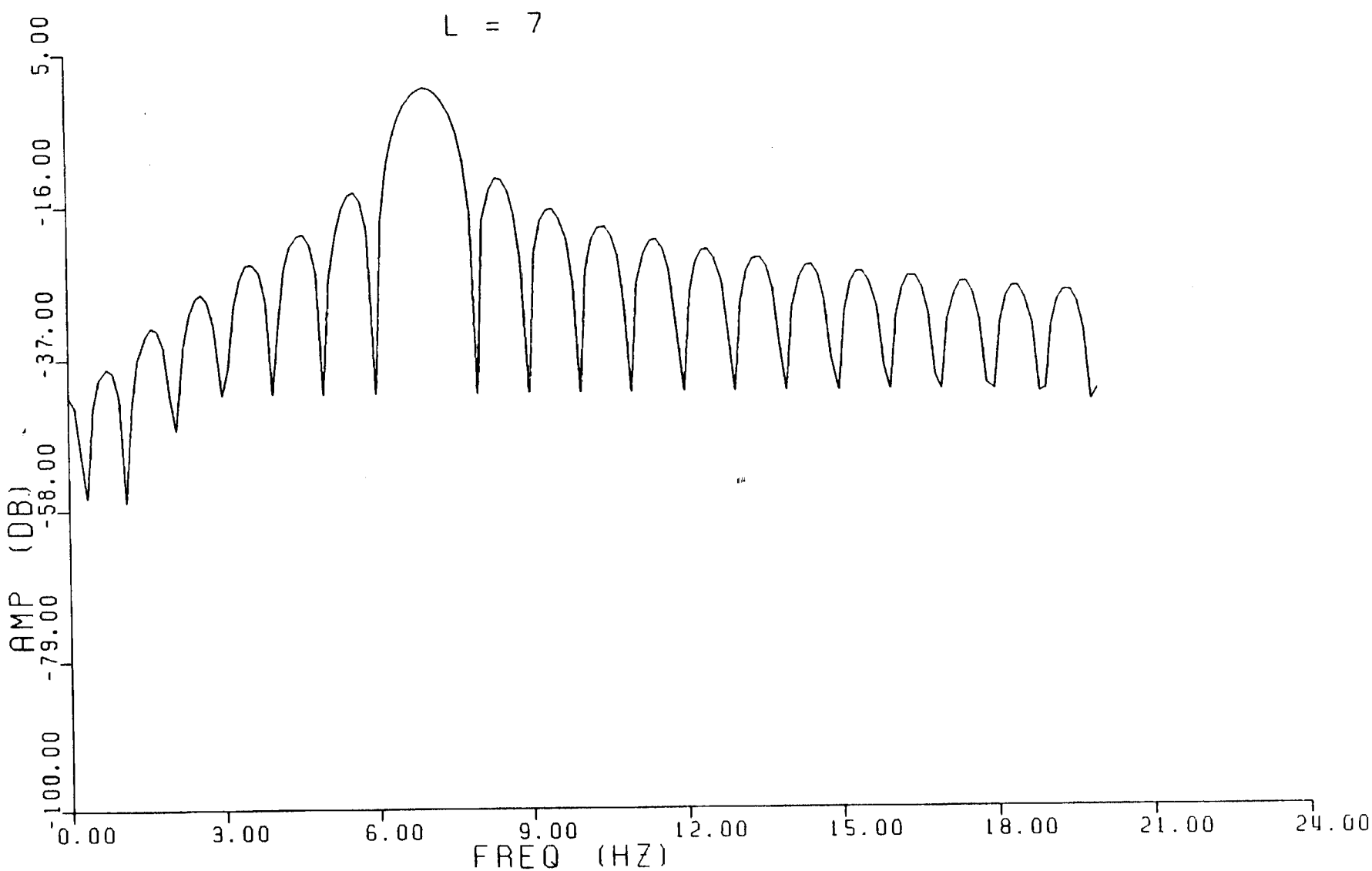


Figure 38 COSINE LAG WINDOW

L = 7

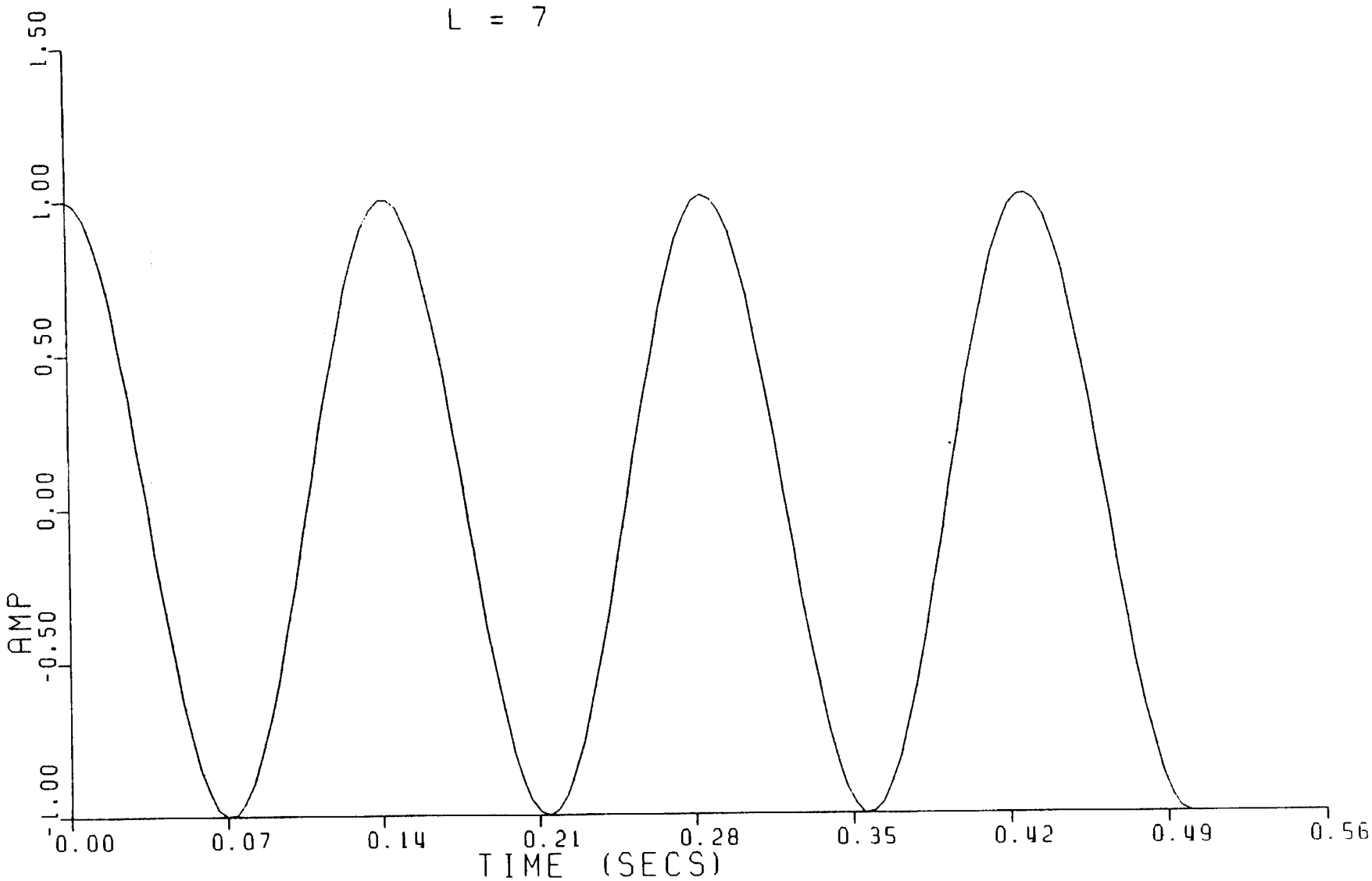


Figure 39

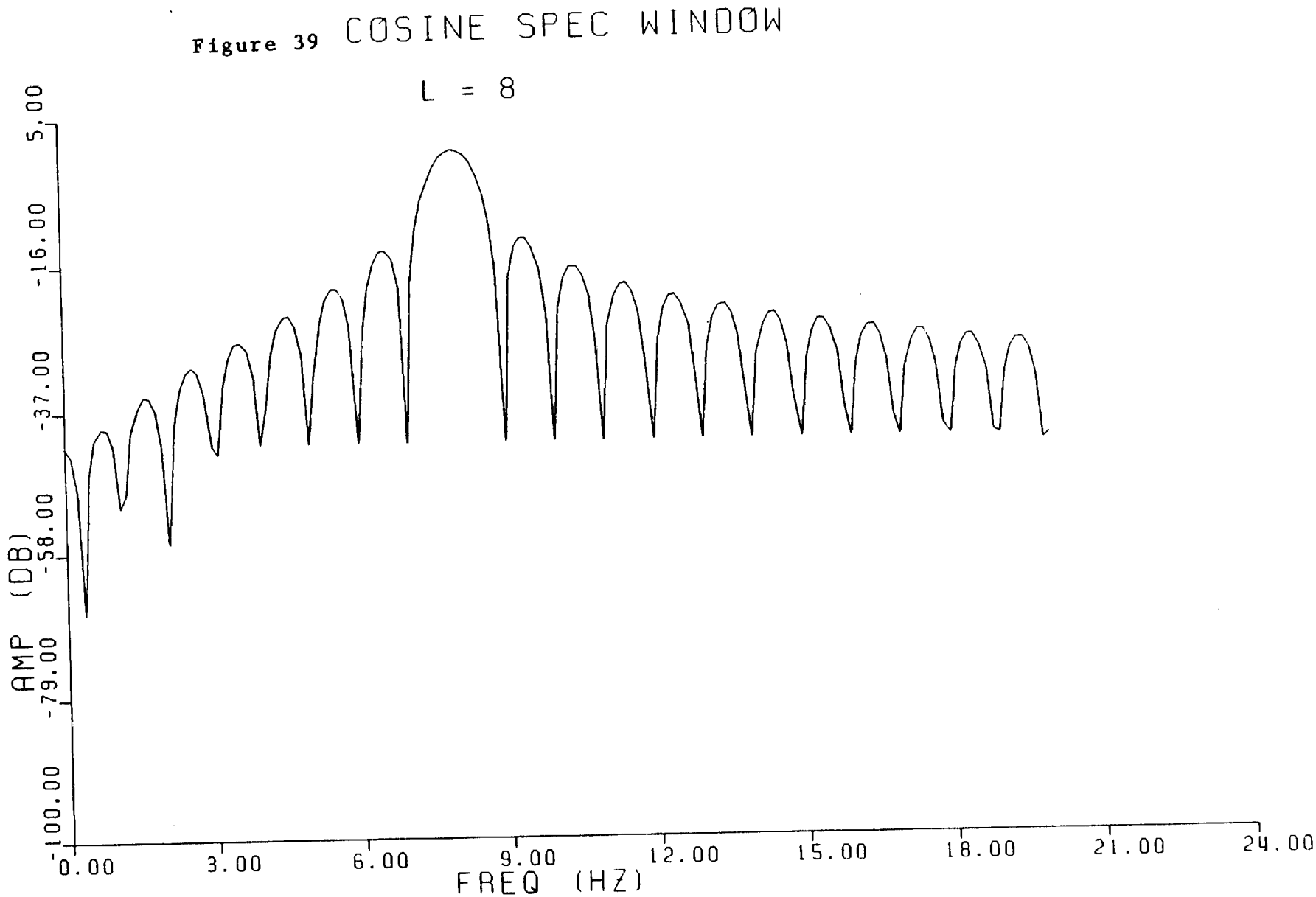

COSINE SPEC WINDOW
L = 8
AMP (DB)
5.00
-16.00
-37.00
-58.00
-79.00
-100.00
0.00
3.00
6.00
9.00
12.00
15.00
18.00
21.00
24.00
FREQ (HZ)

Figure 40

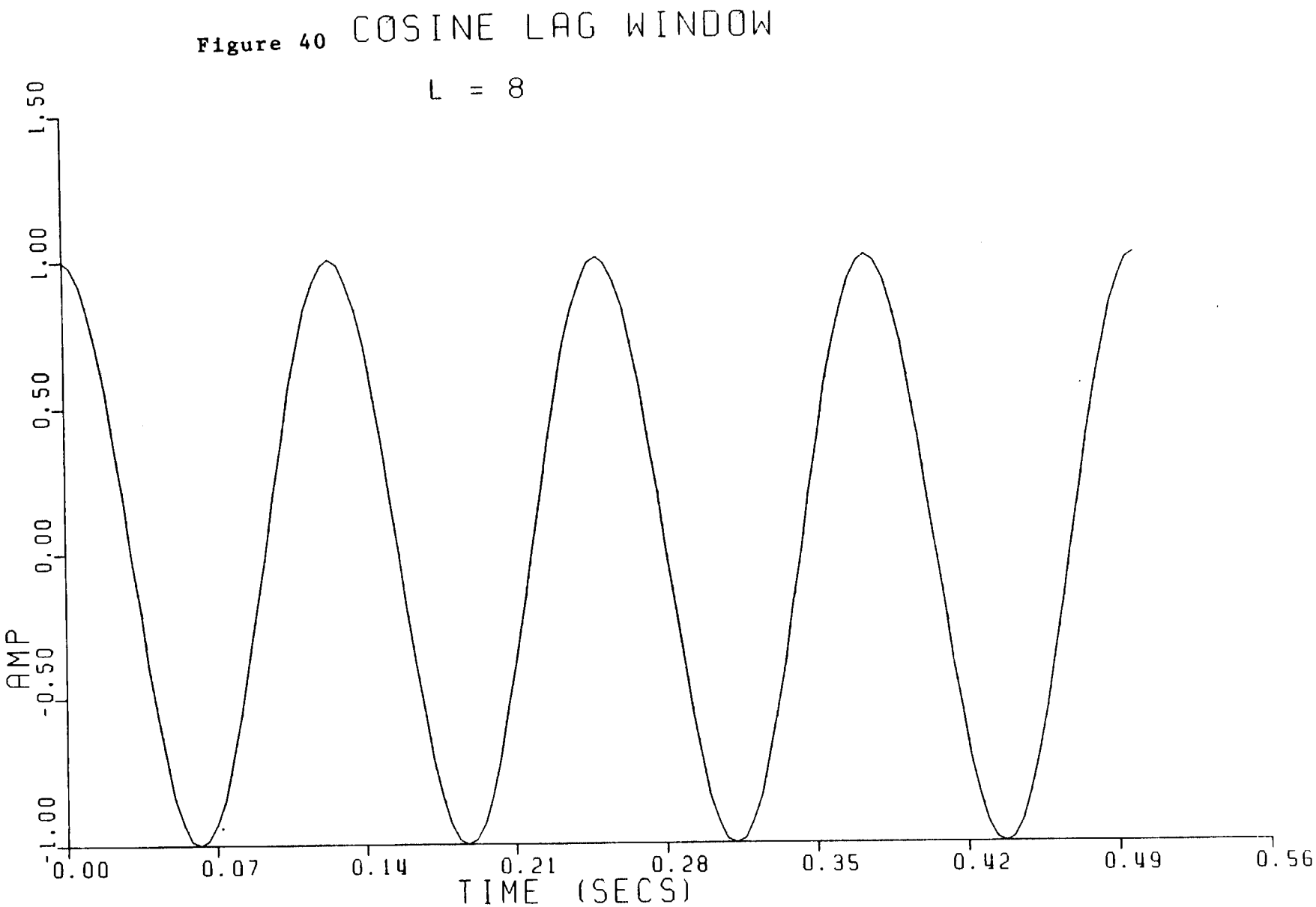

COSINE LAG WINDOW
L = 8
AMP
1.50
1.00
0.50
0.00
-0.50
-1.00
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
0.56
TIME (SECS)

Figure 41 PRGRM-1, 1-2 HZ

HYBRID SPEC WINDOW

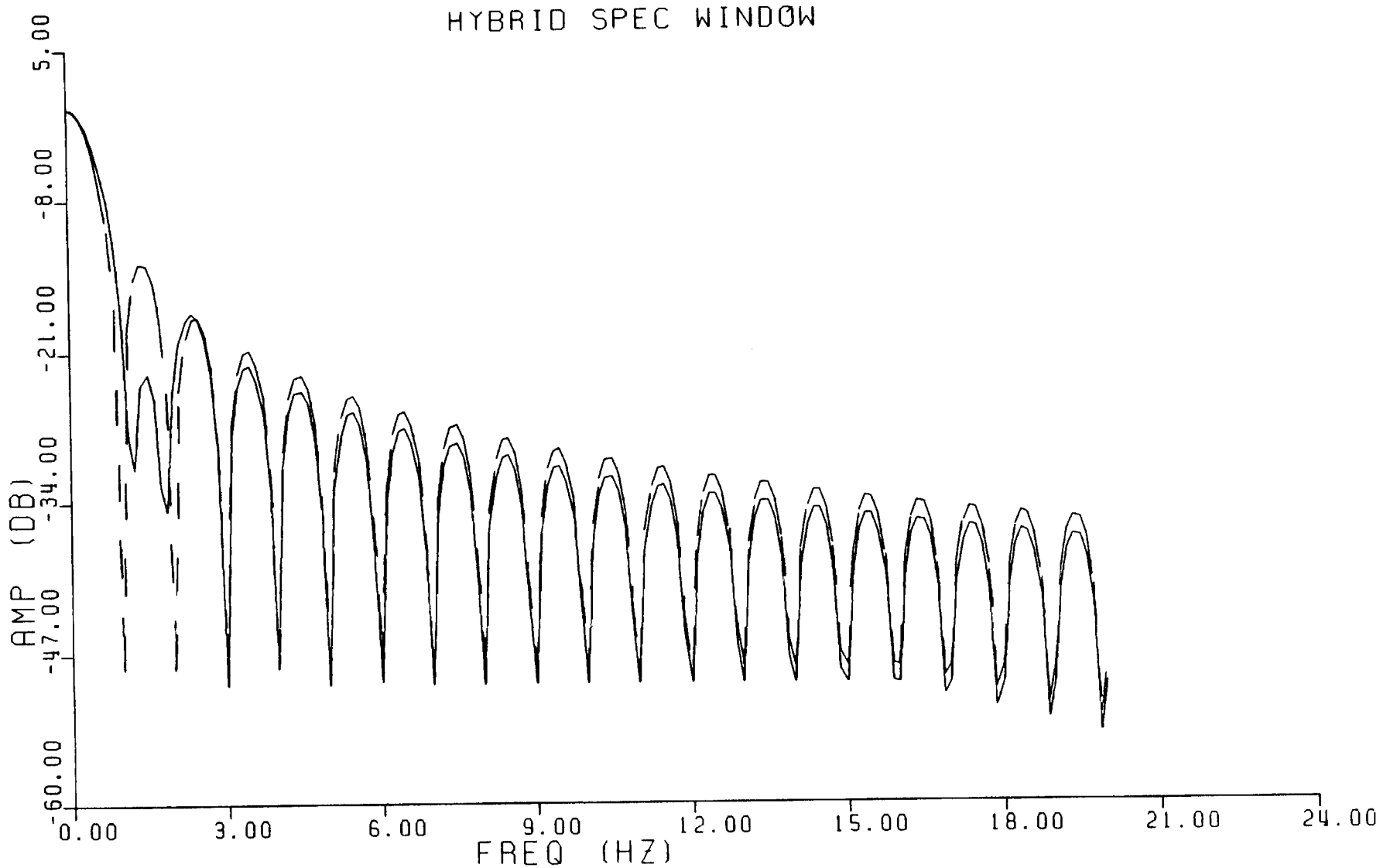


Figure 42

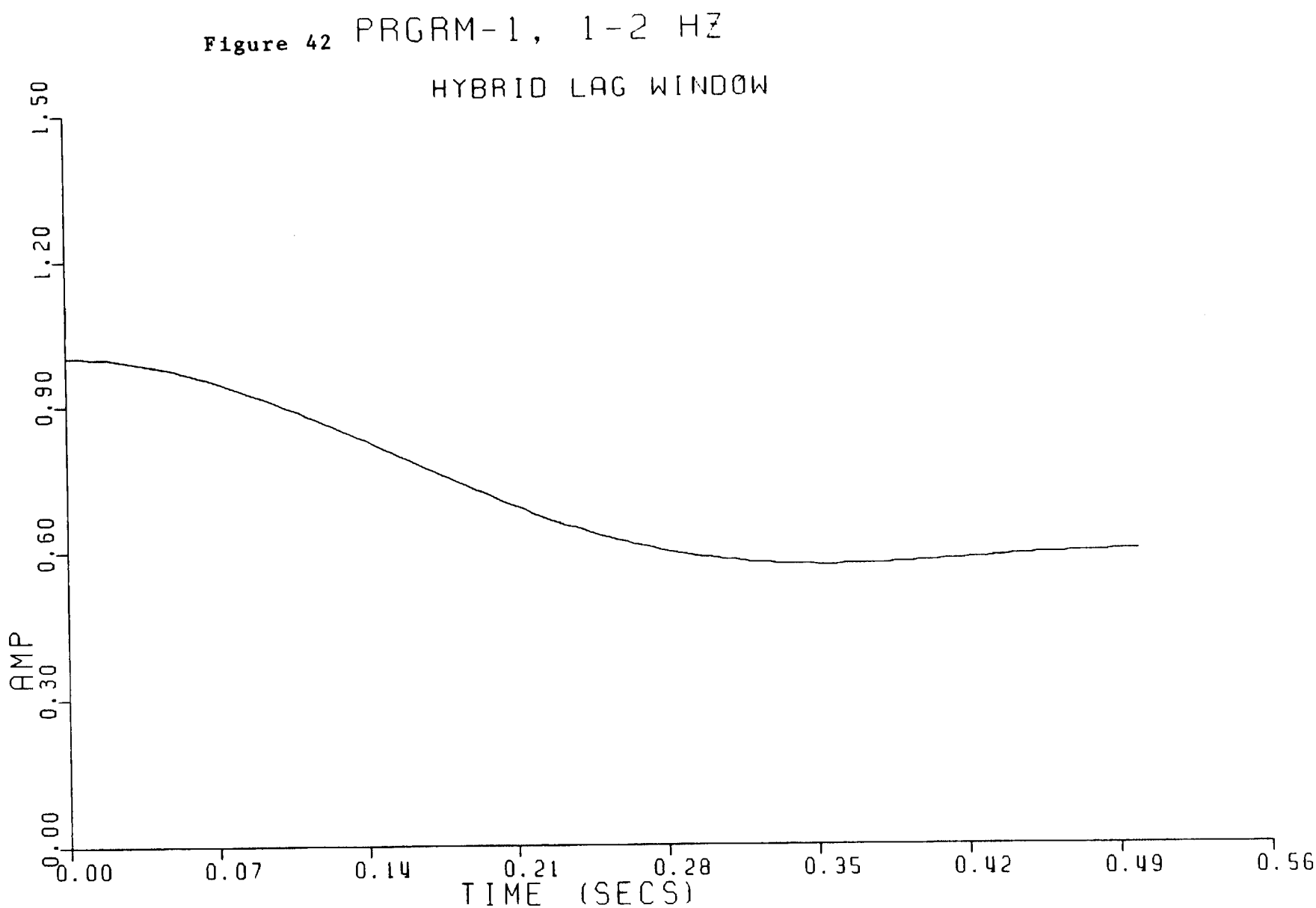

PRGRM-1, 1-2 HZ
HYBRID LAG WINDOW
AMP
0.00
0.30
0.60
0.90
1.20
1.50
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
0.56
TIME (SECS)

Figure 43 PRGRM-2, 1-2 HZ

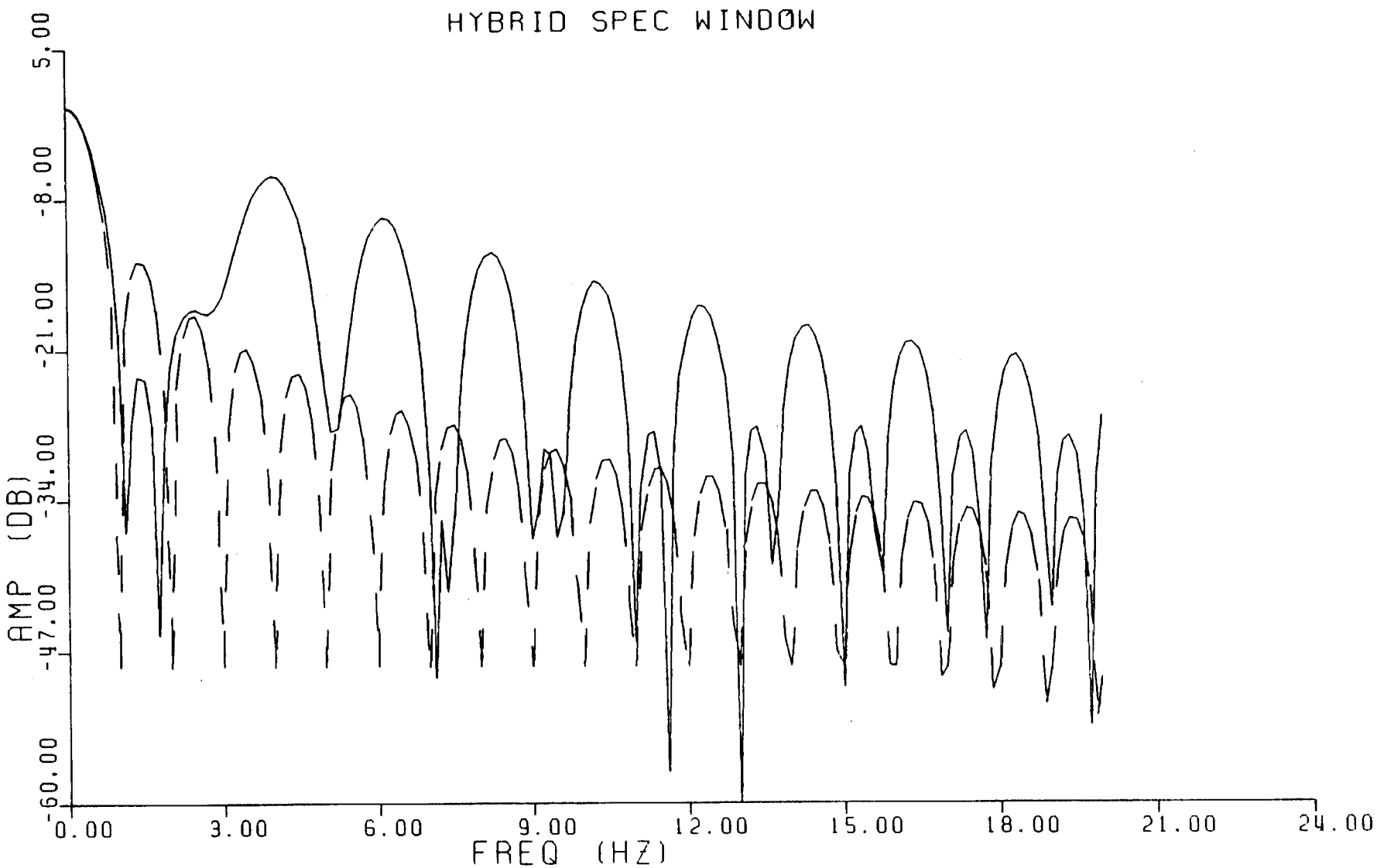


Figure 44 PRGRM-2, 1-2 HZ

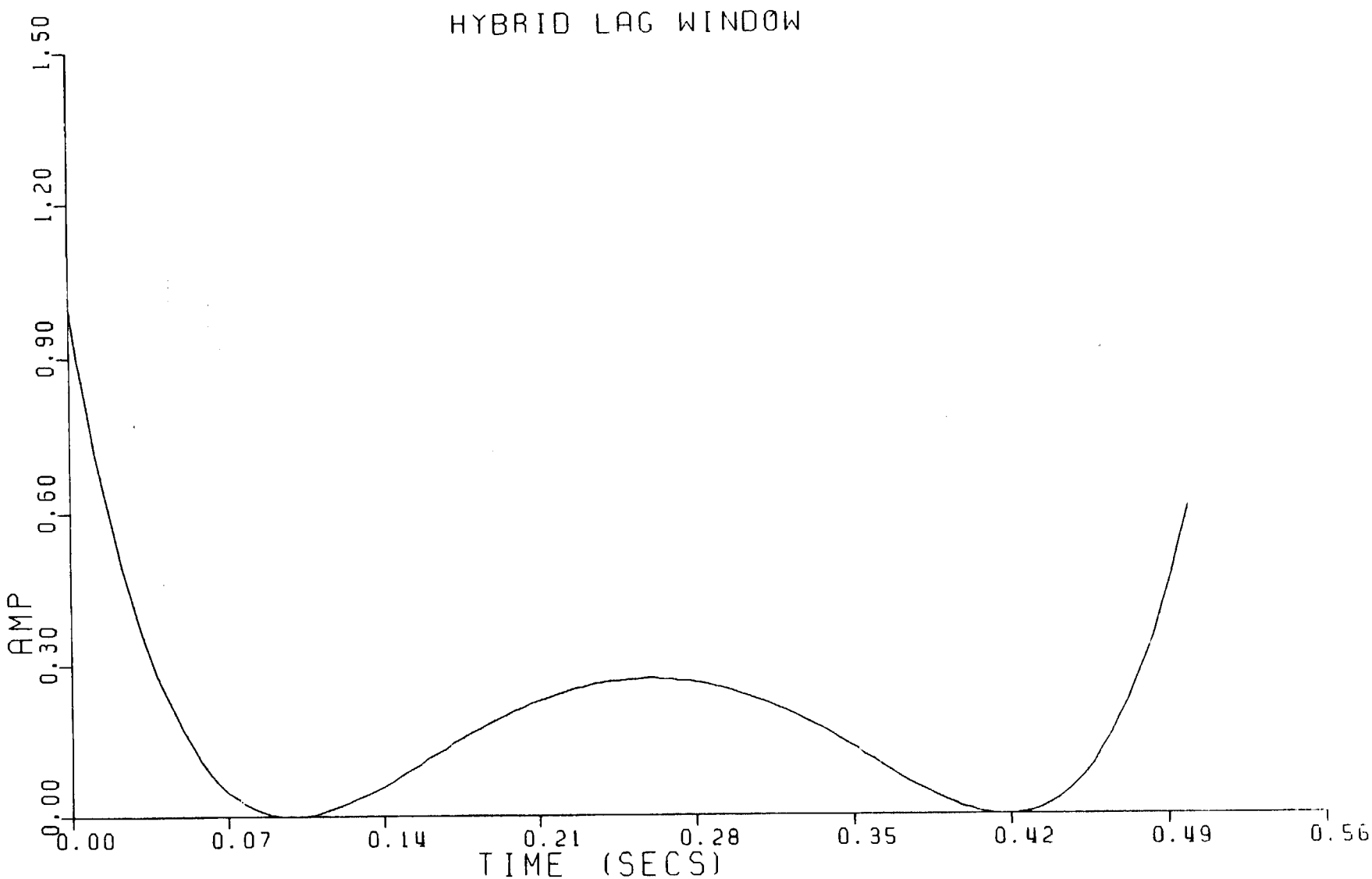


Figure 45 PRGRM-3, 1-2 HZ

HYBRID, 3 WINDOWS

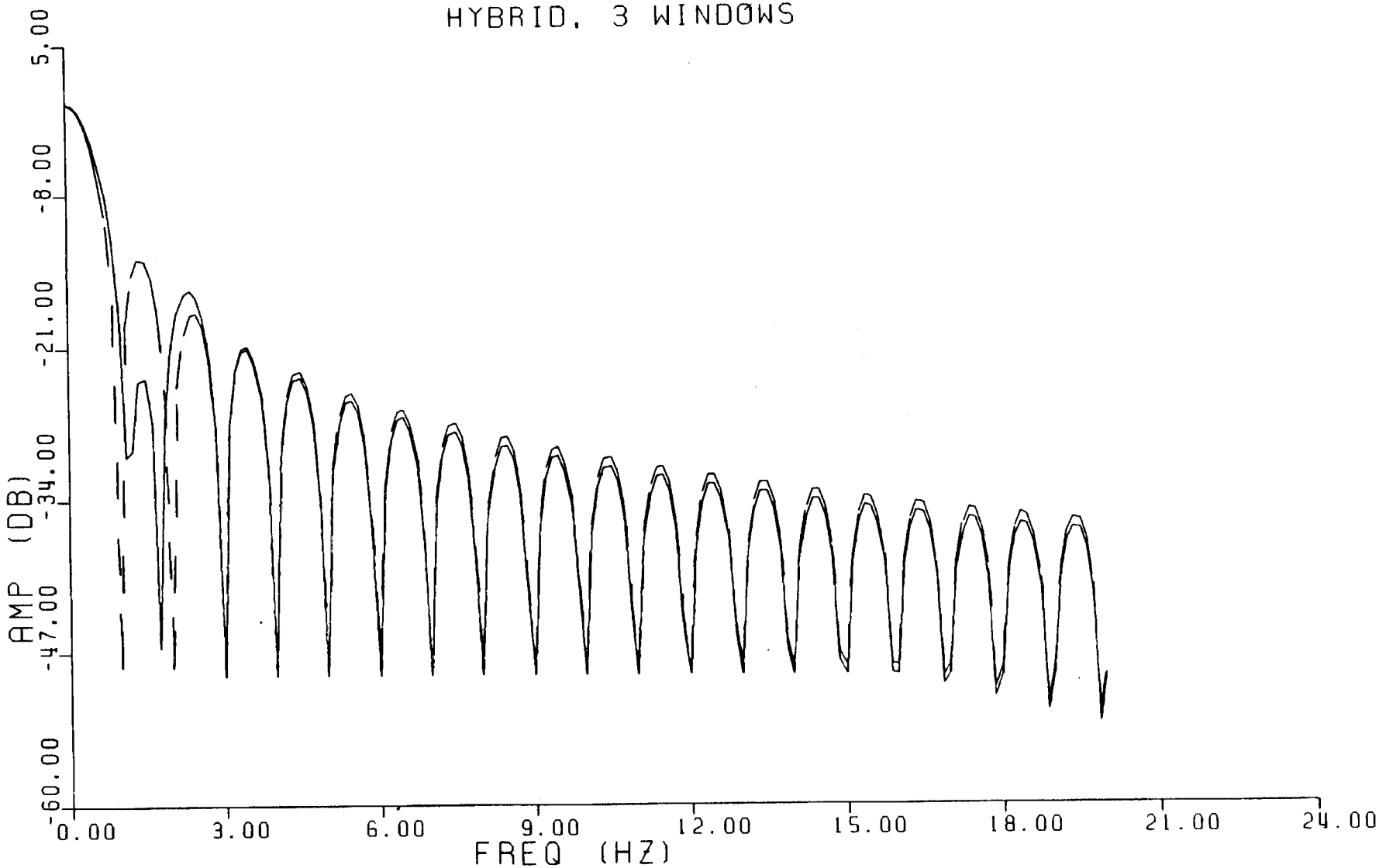


Figure 46 PRGRM-3, 1-2 HZ

LAG, 3 WINDOWS

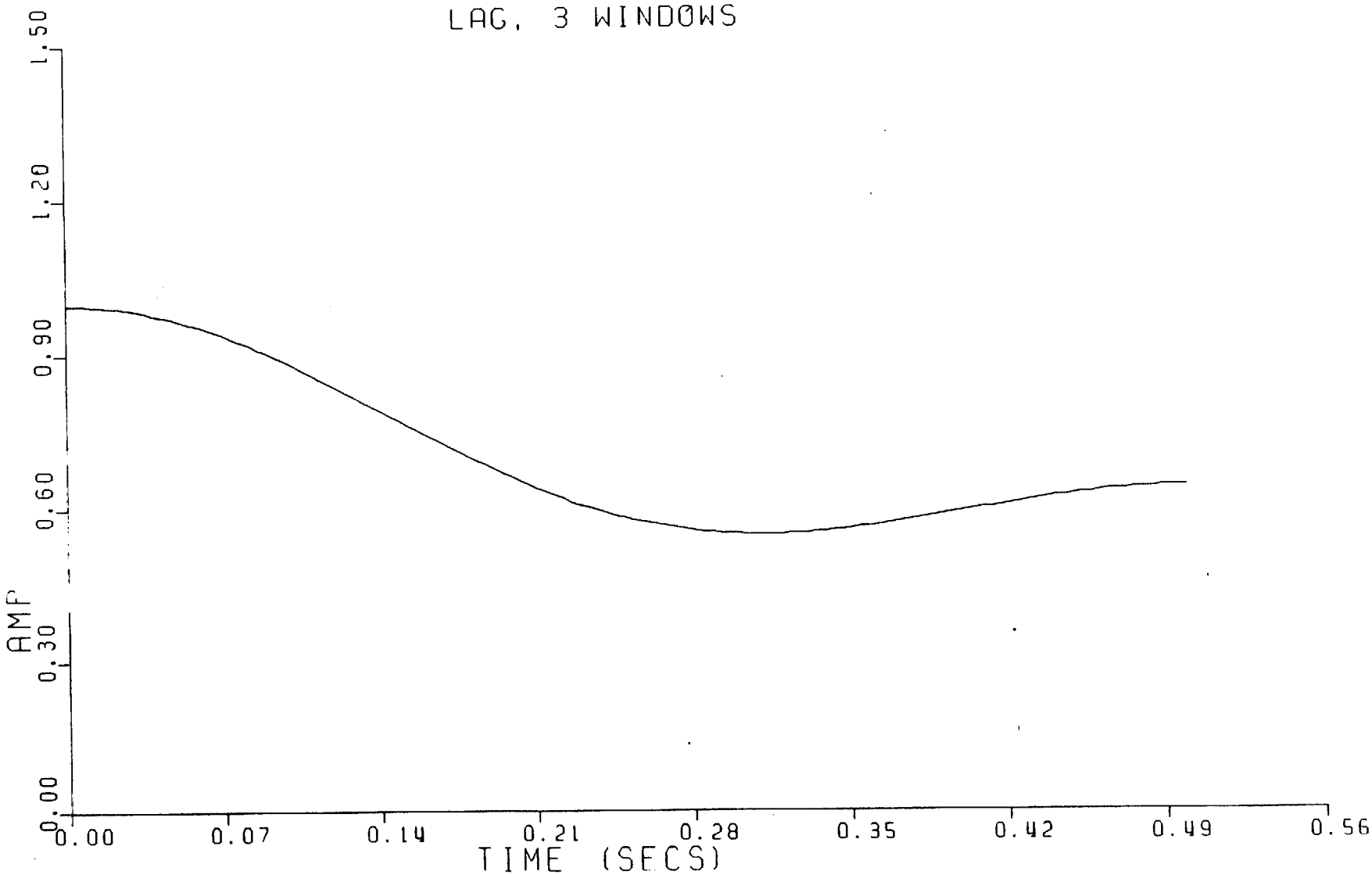


Figure 47 PRGRM-3, 1-2 HZ

HYBRID, 6 WINDOWS

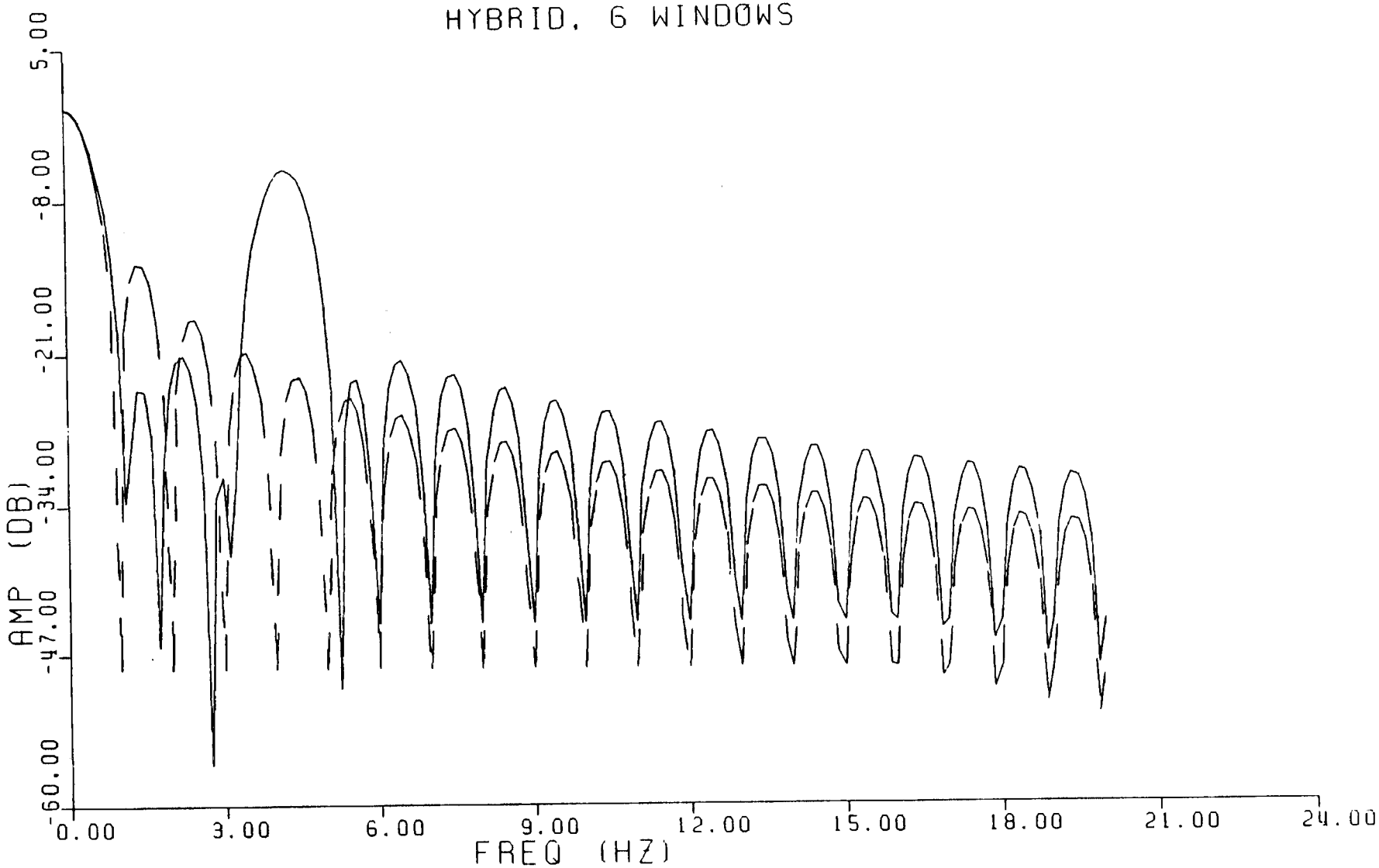


Figure 48 PRGRM-3, 1-2 HZ

LAG, 6 WINDOWS

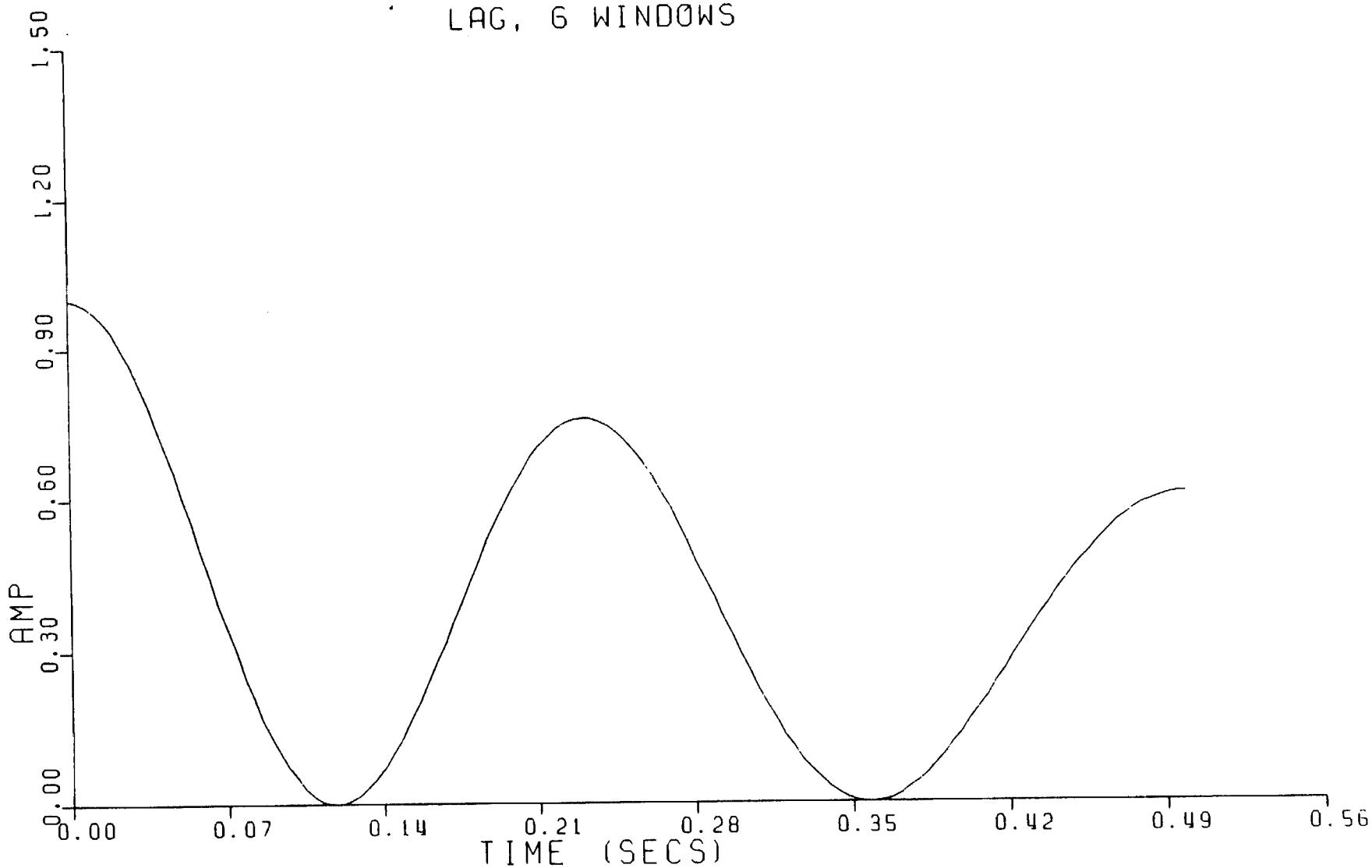


Figure 49 PRGRM-3, 1-2 HZ

HYBRID, 9 WINDOWS

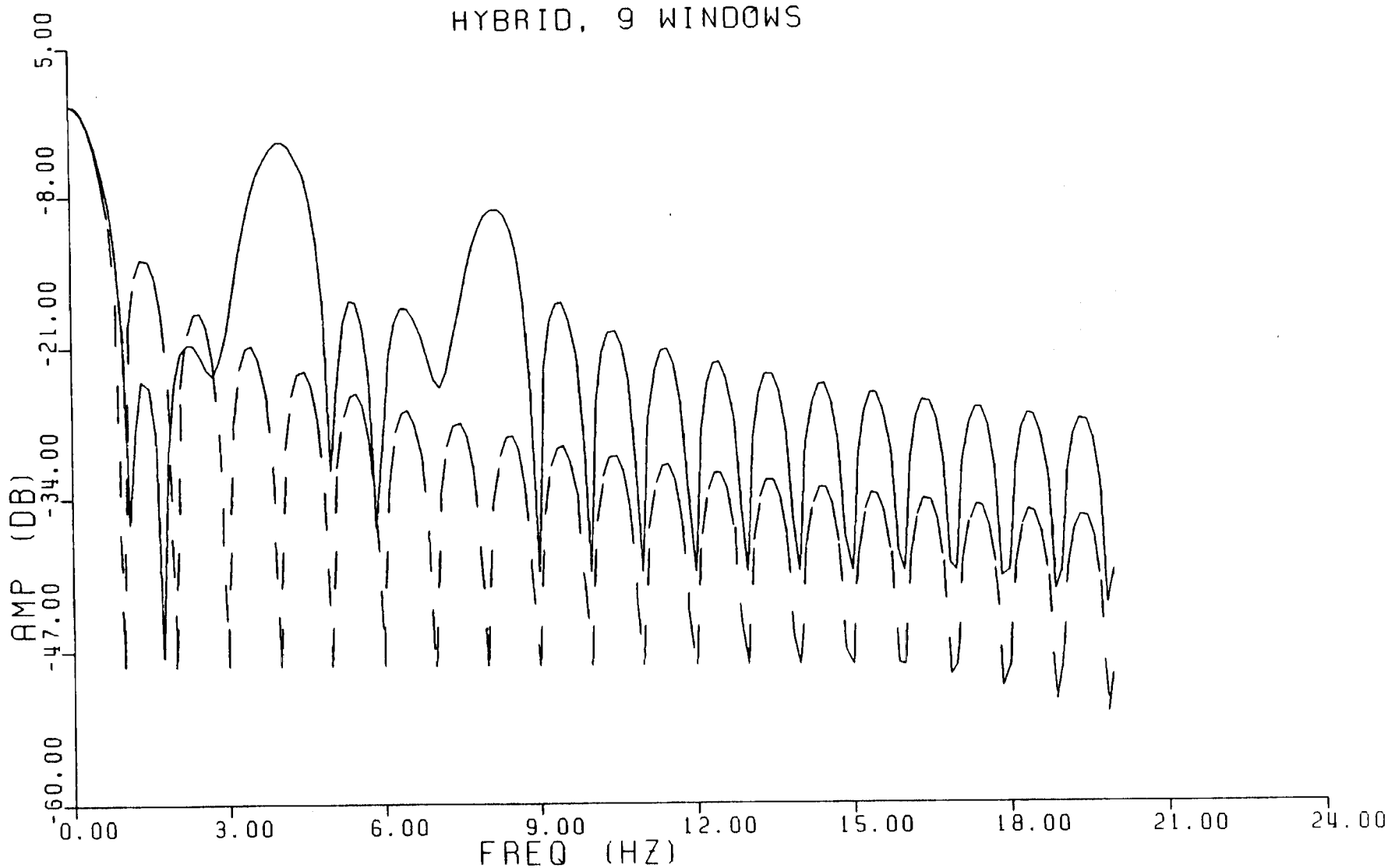


Figure 50 PRGRM-3, 1-2 HZ

LAG, 9 WINDOWS

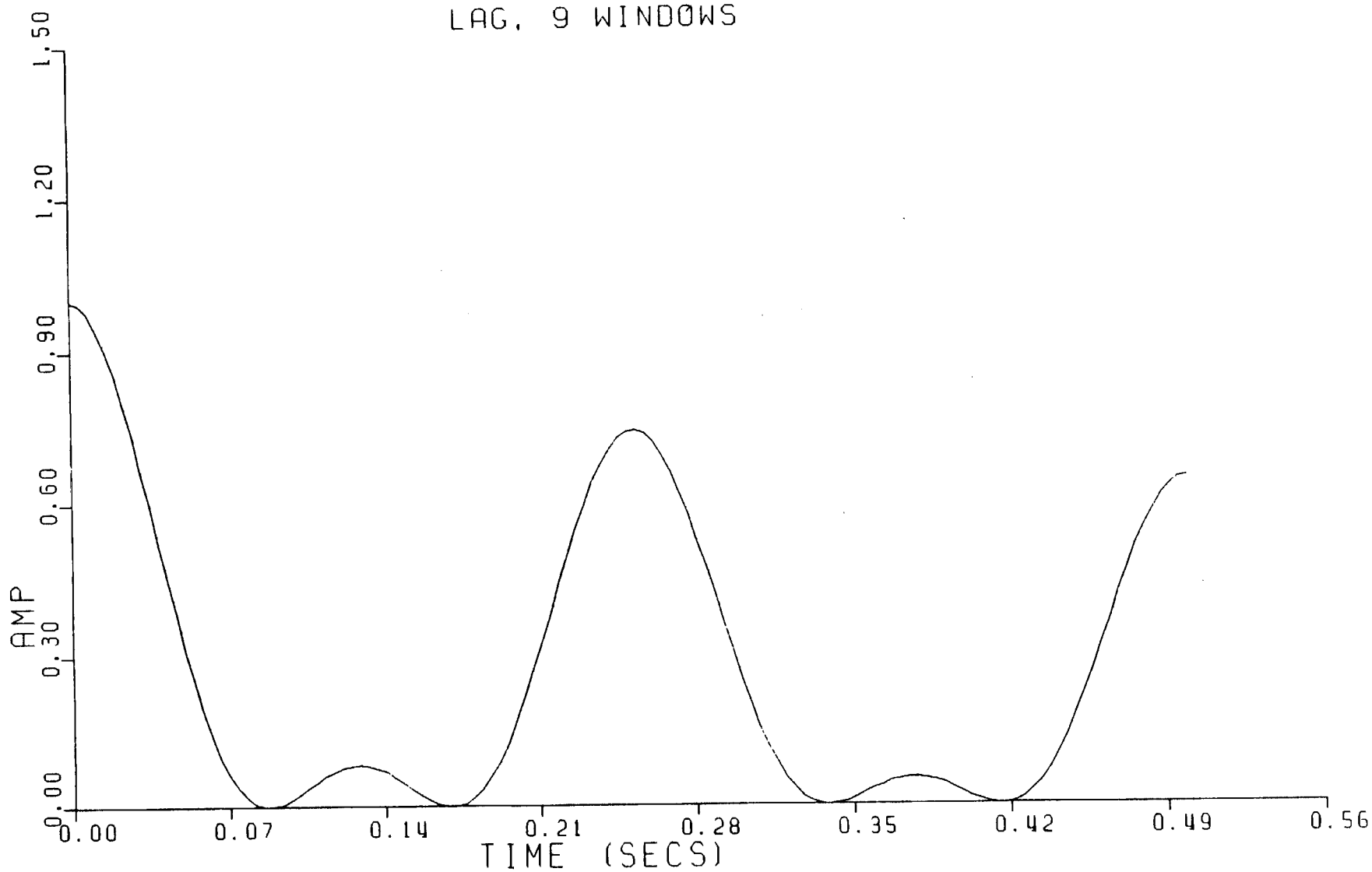


Figure 51 PRGRM-1, 2-4 HZ

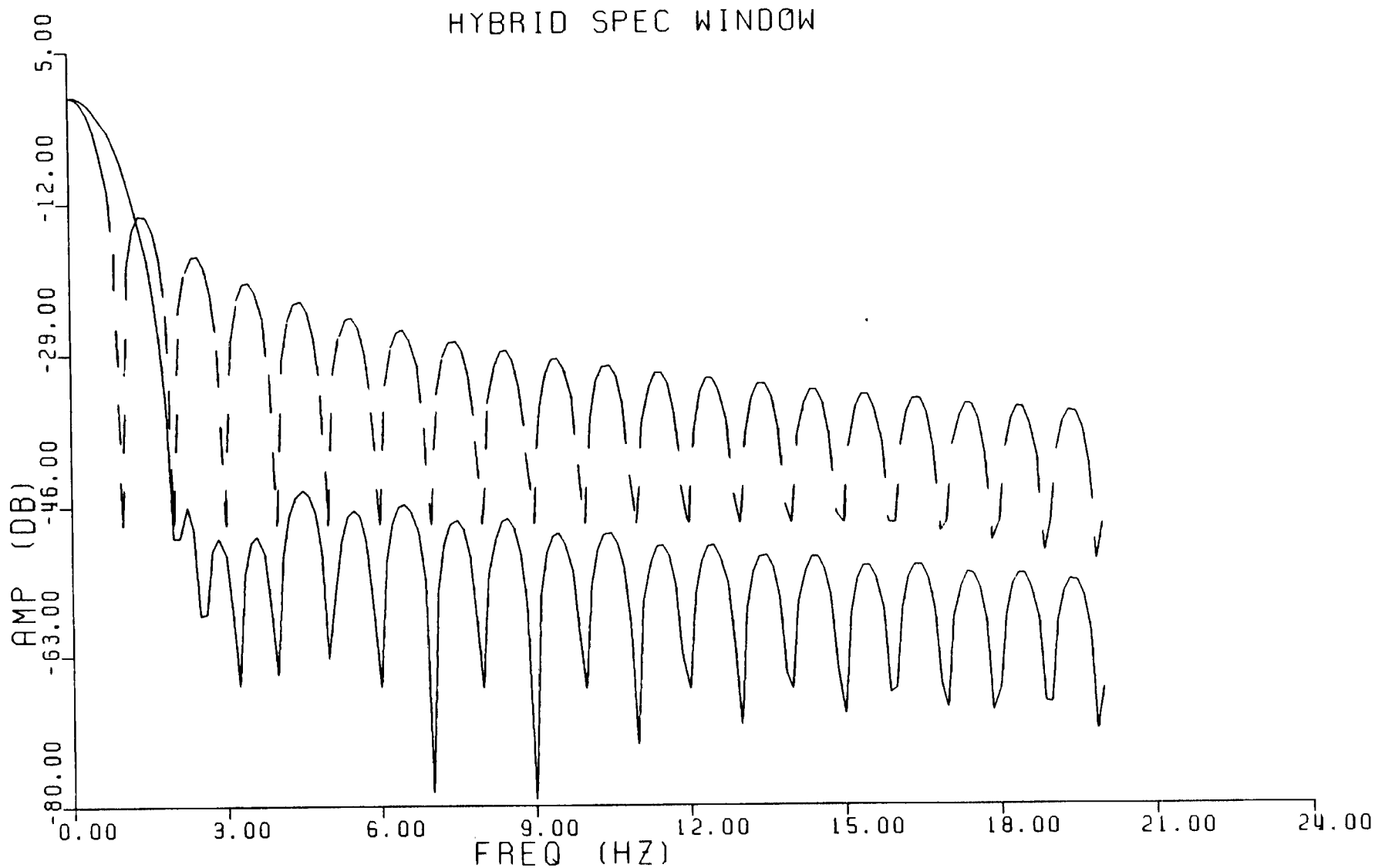


Figure 52 PRGRM-1, 2-4 HZ

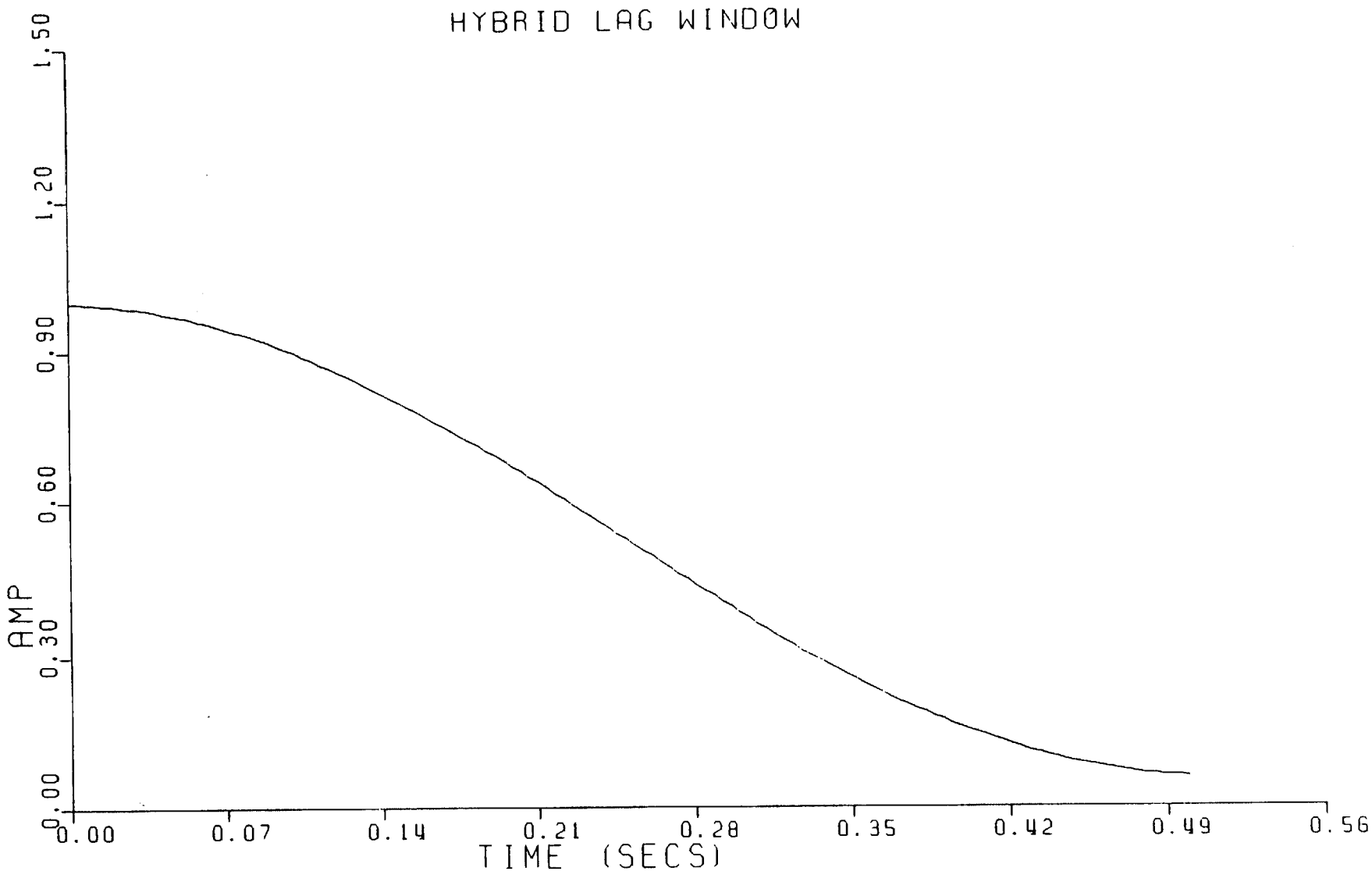


Figure 53 PRGRM-2, 2-4 HZ

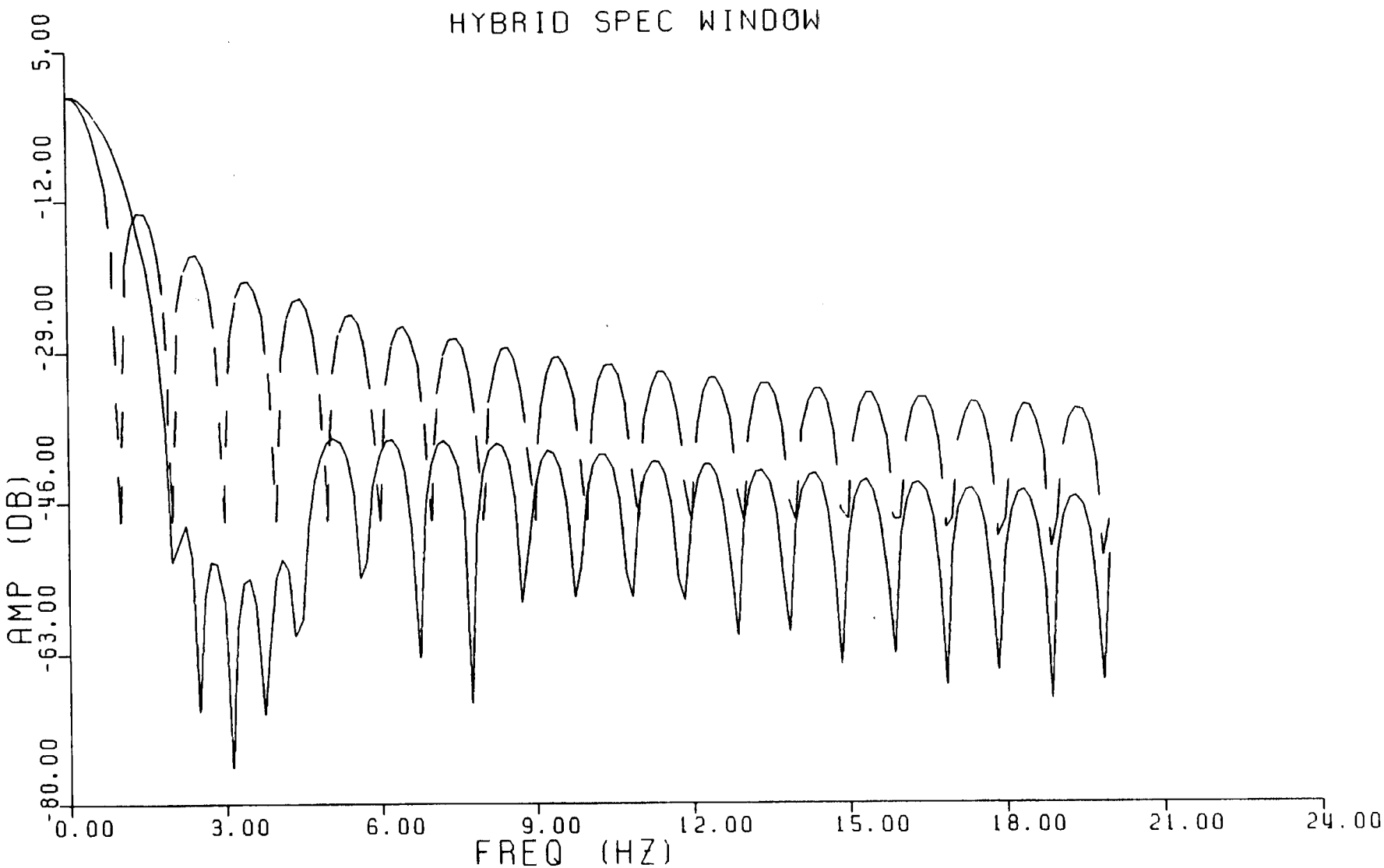


Figure 54 PRGRM-2, 2-4 HZ

HYBRID LAG WINDOW

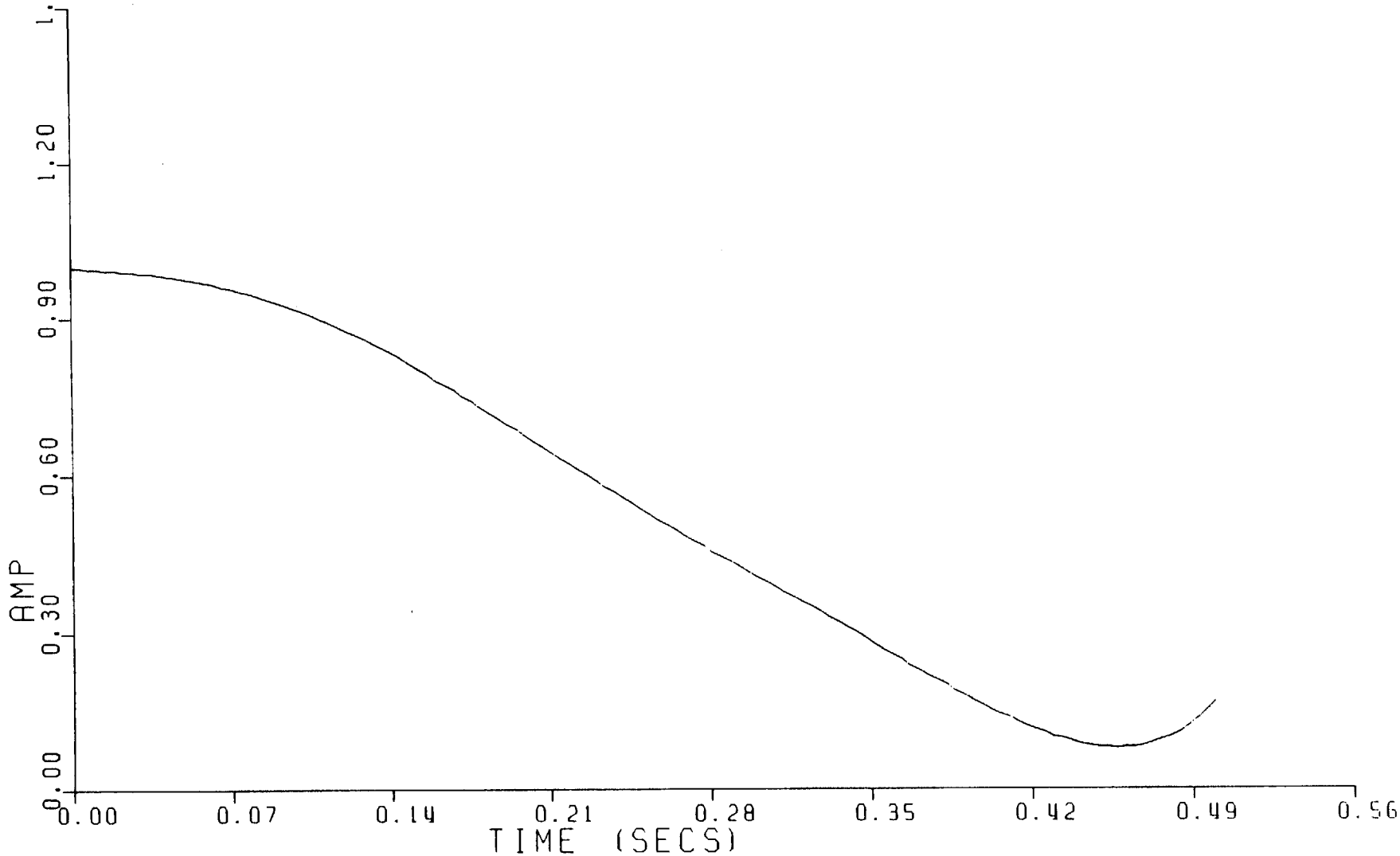


Figure 55 PRGRM-3, 2-4 HZ

HYBRID, 3 WINDOWS

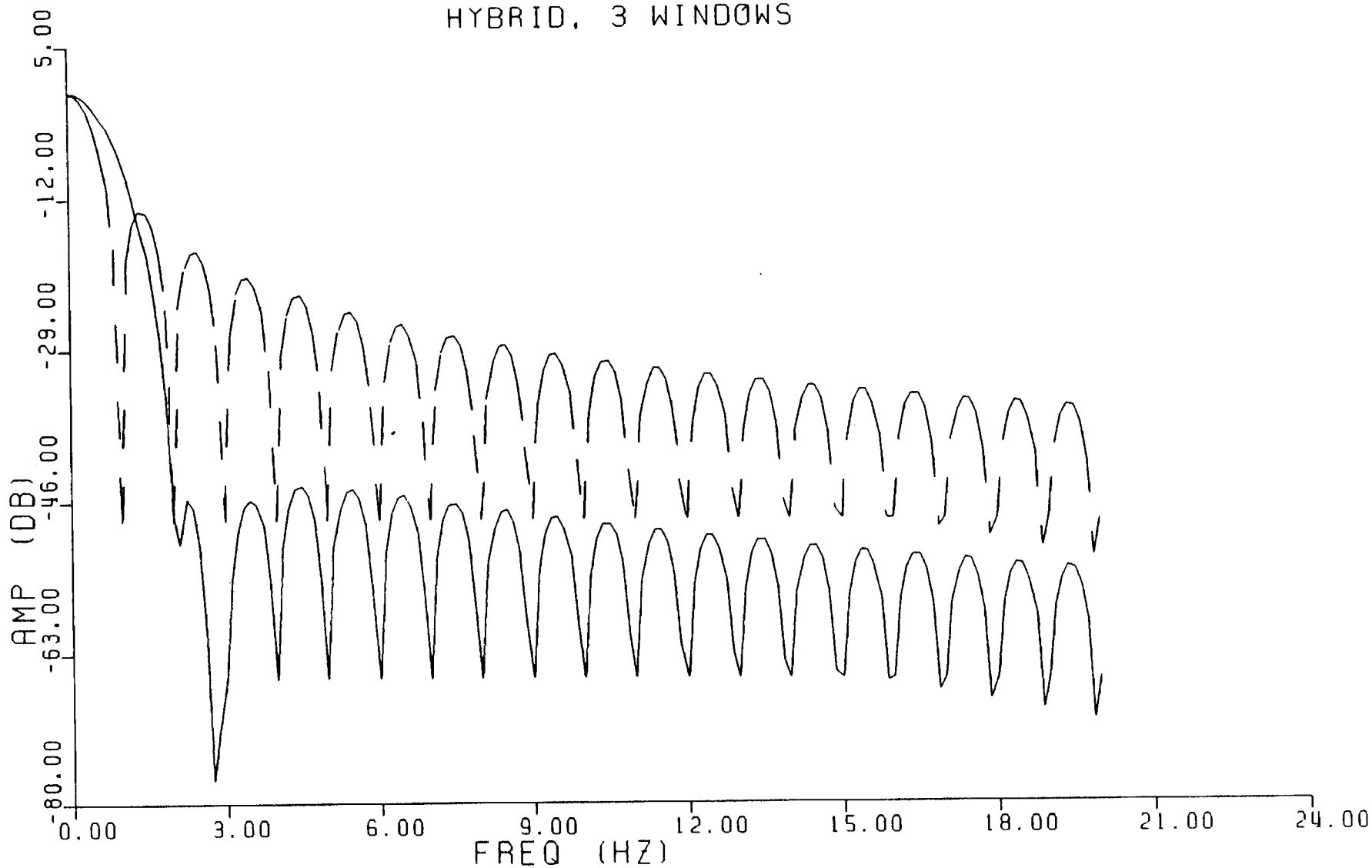


Figure 56 PRGRM-3, 2-4 HZ

LAG, 3 WINDOWS

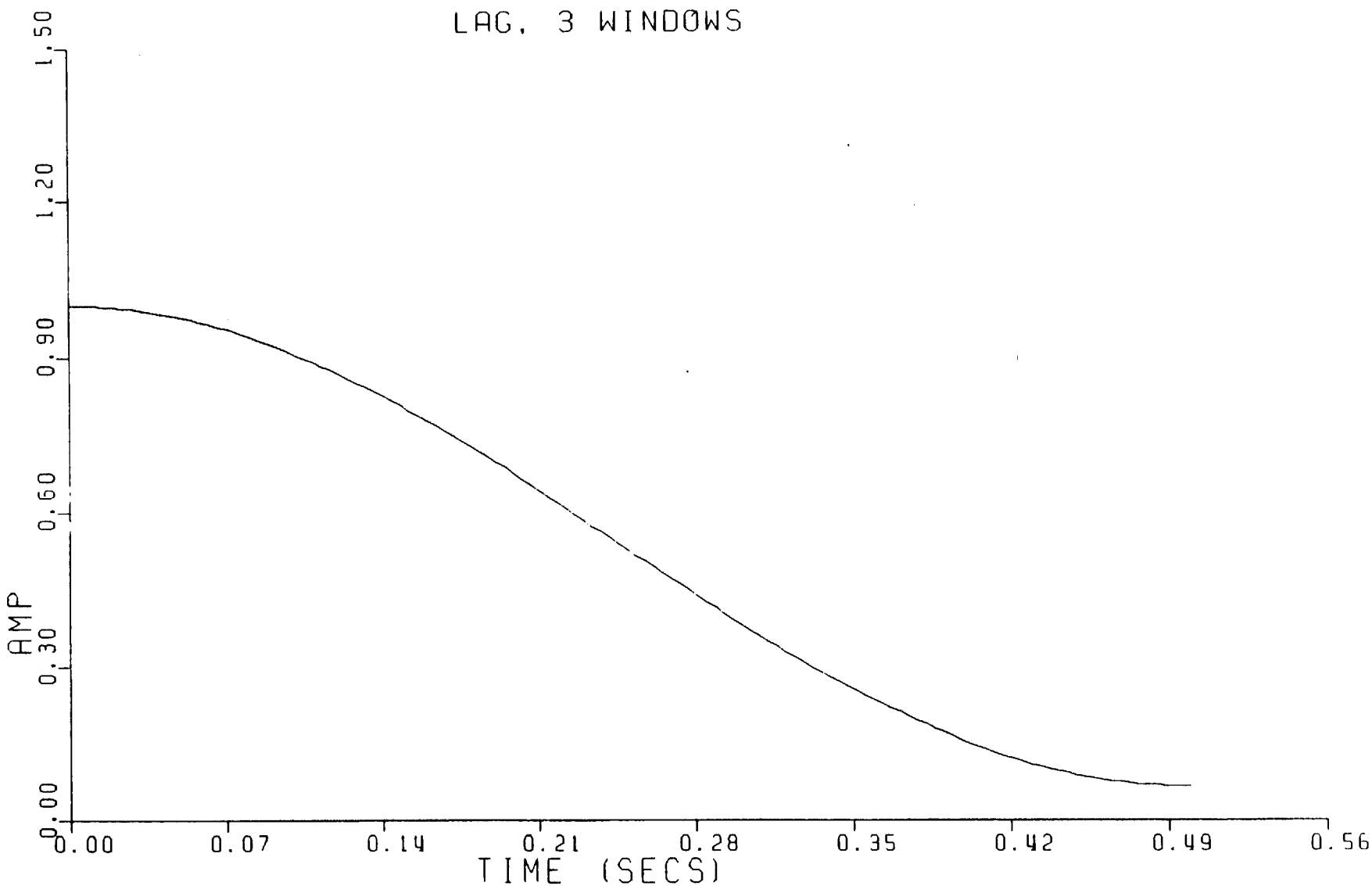
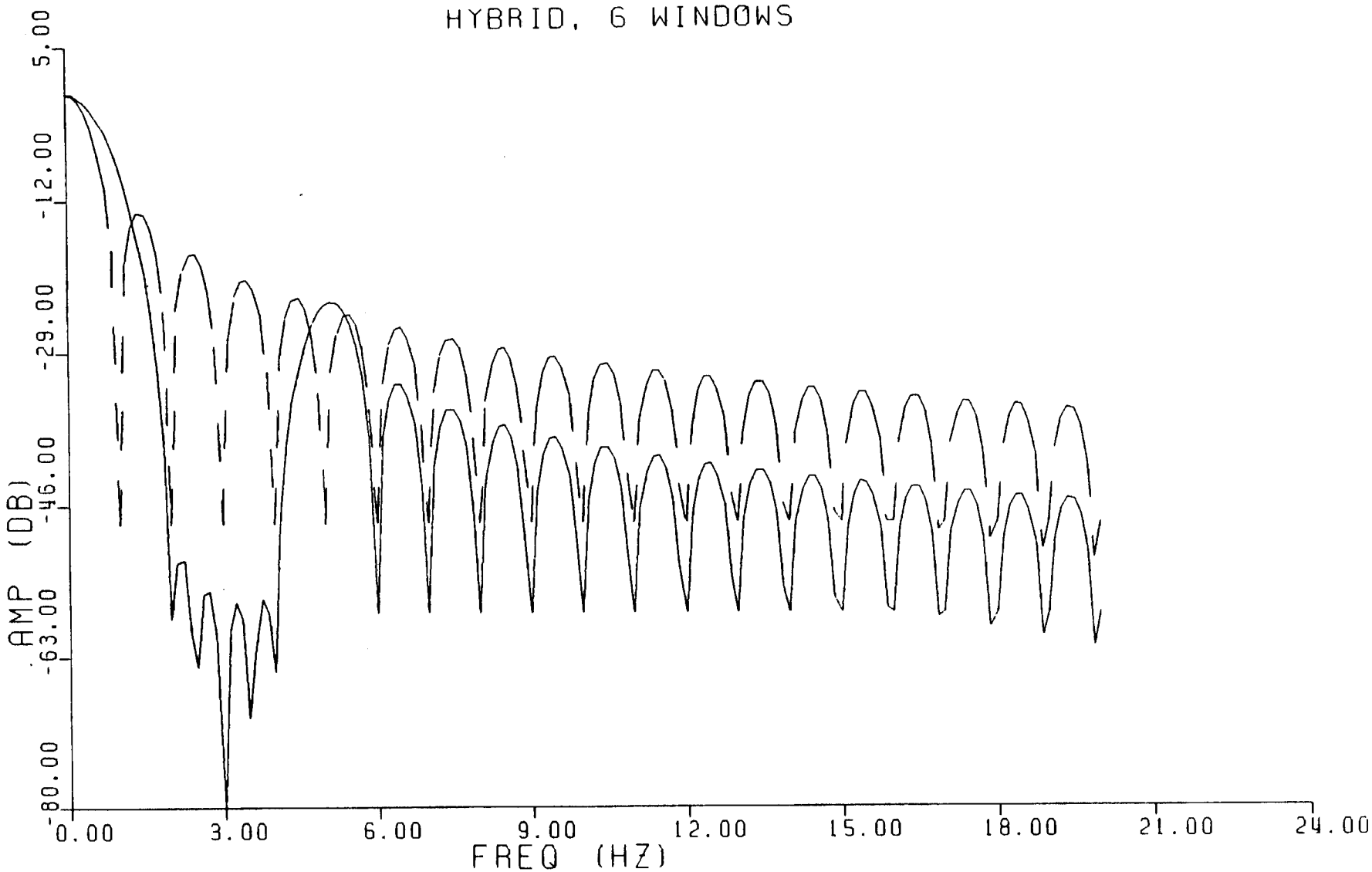


Figure 57 PRGRM-3, 2-4 HZ

HYBRID, 6 WINDOWS



C-2

Figure 58 PRGRM-3, 2-4 HZ

LAG, 6 WINDOWS

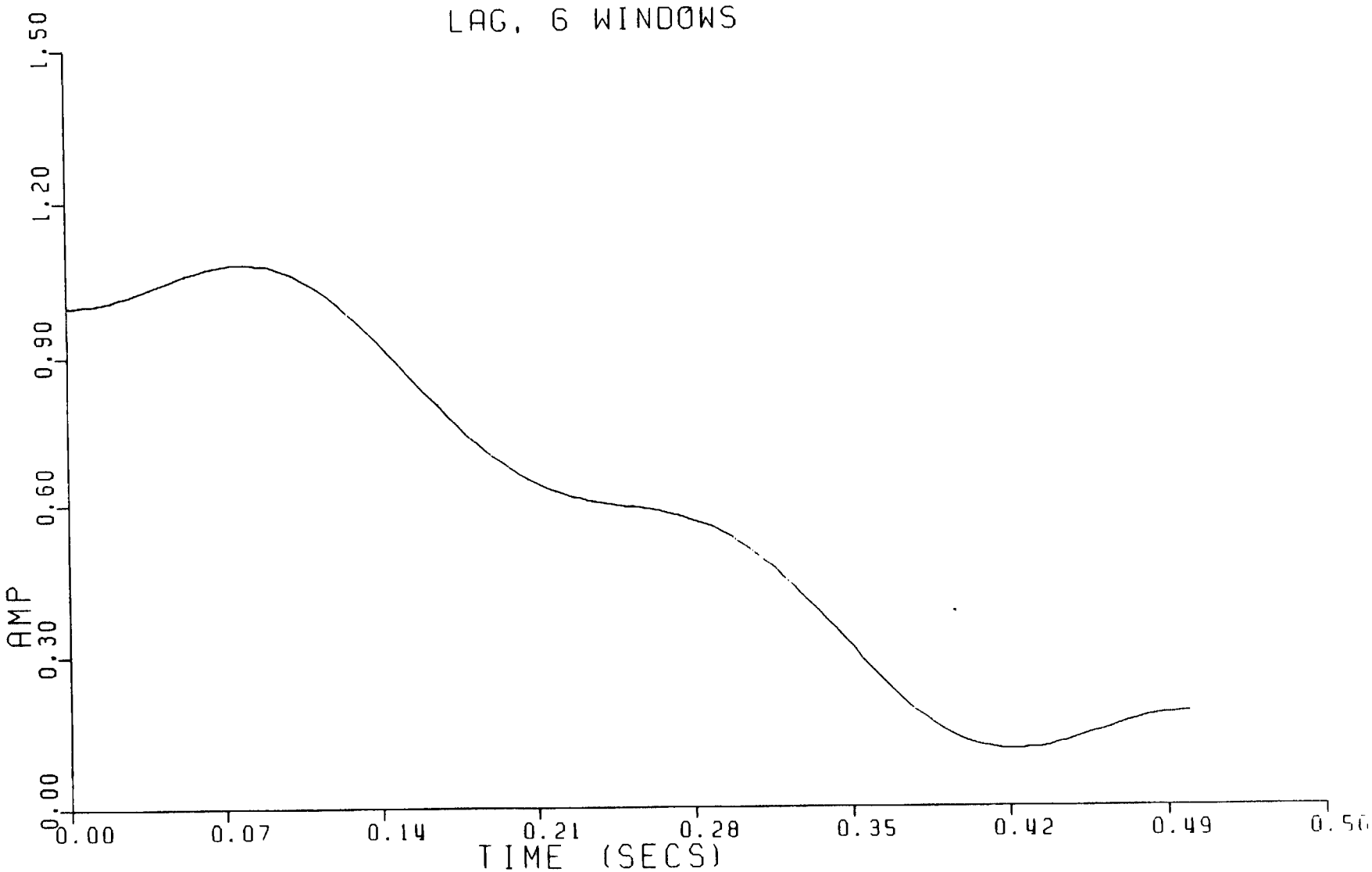


Figure 59 PRGRM-3, 2-4 HZ

HYBRID, 9 WINDOWS

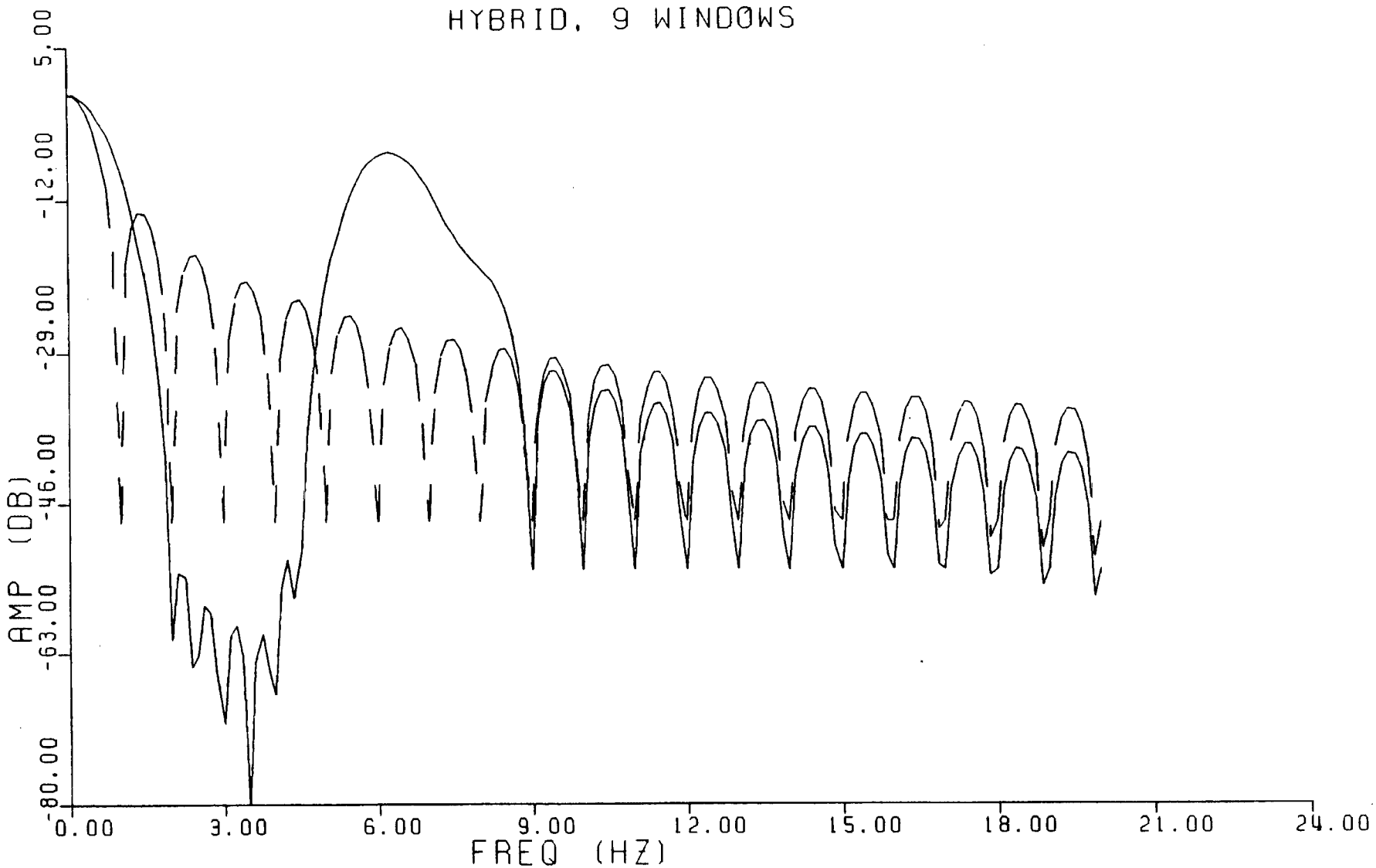


Figure 60 PRGRM-3, 2-4 HZ

LAG, 9 WINDOWS

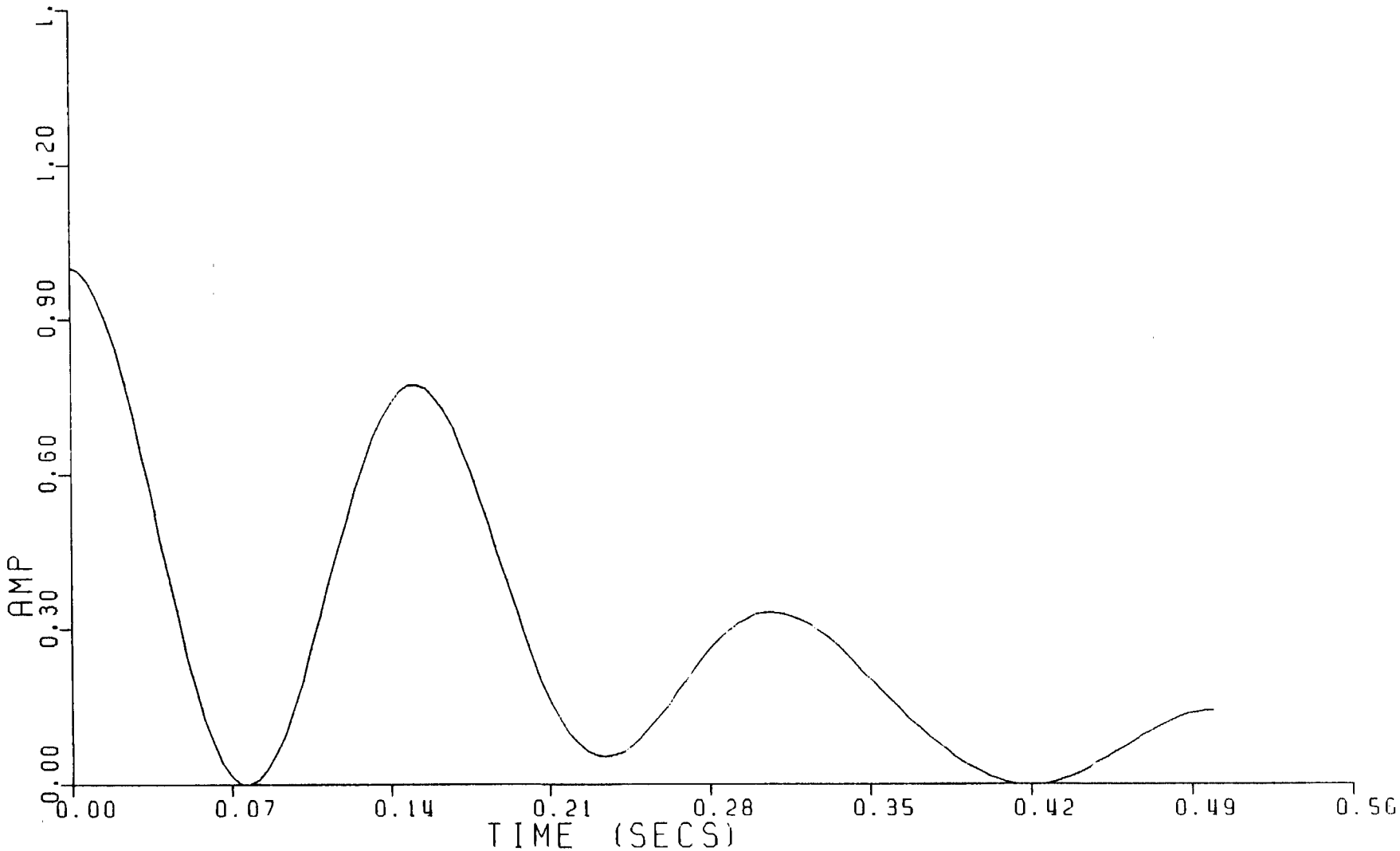


Figure 61 PRGRM-1, 2-8 HZ

HYBRID SPEC WINDOW

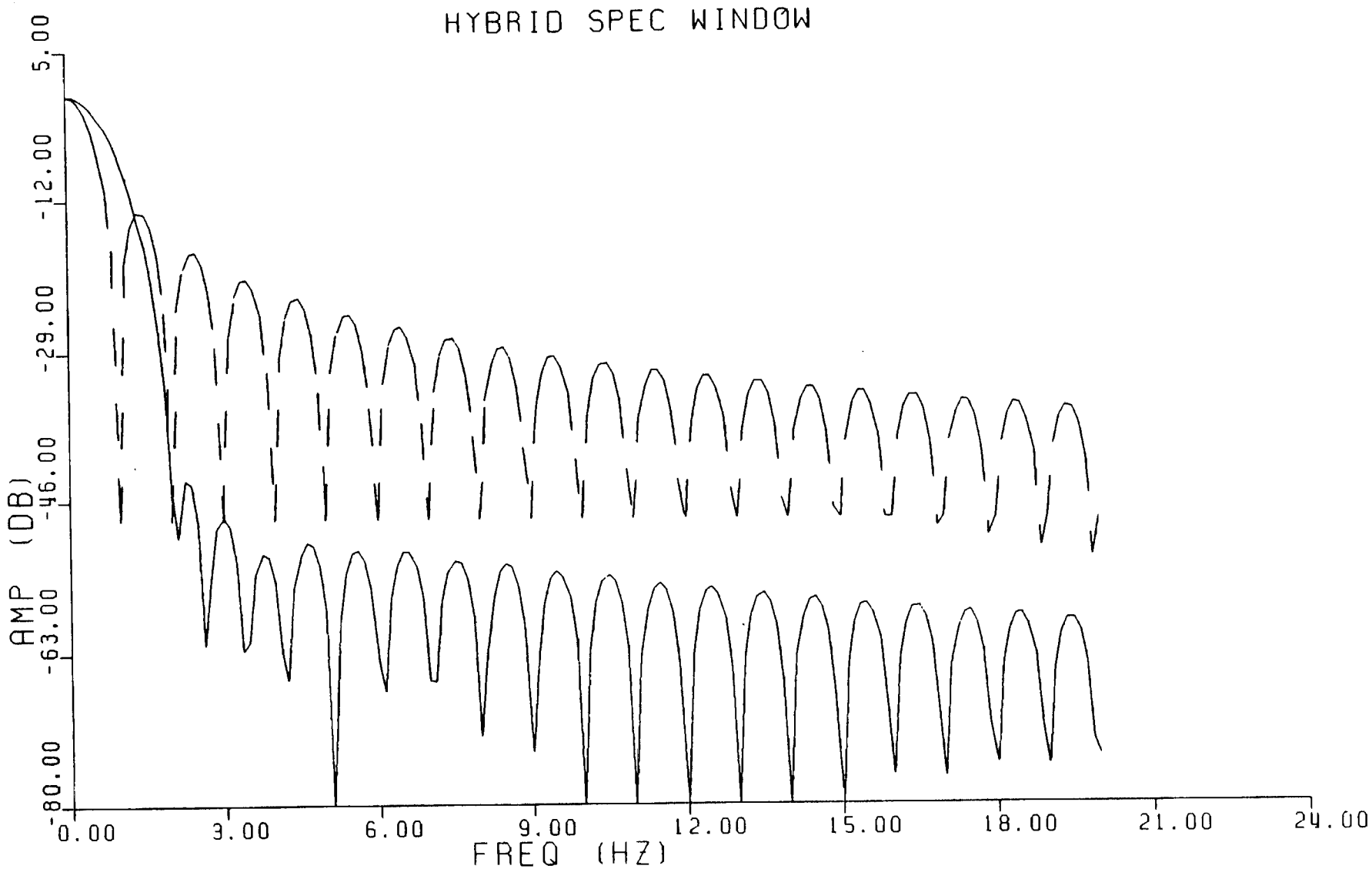


Figure 62 PRGRM-1, 2-8 HZ

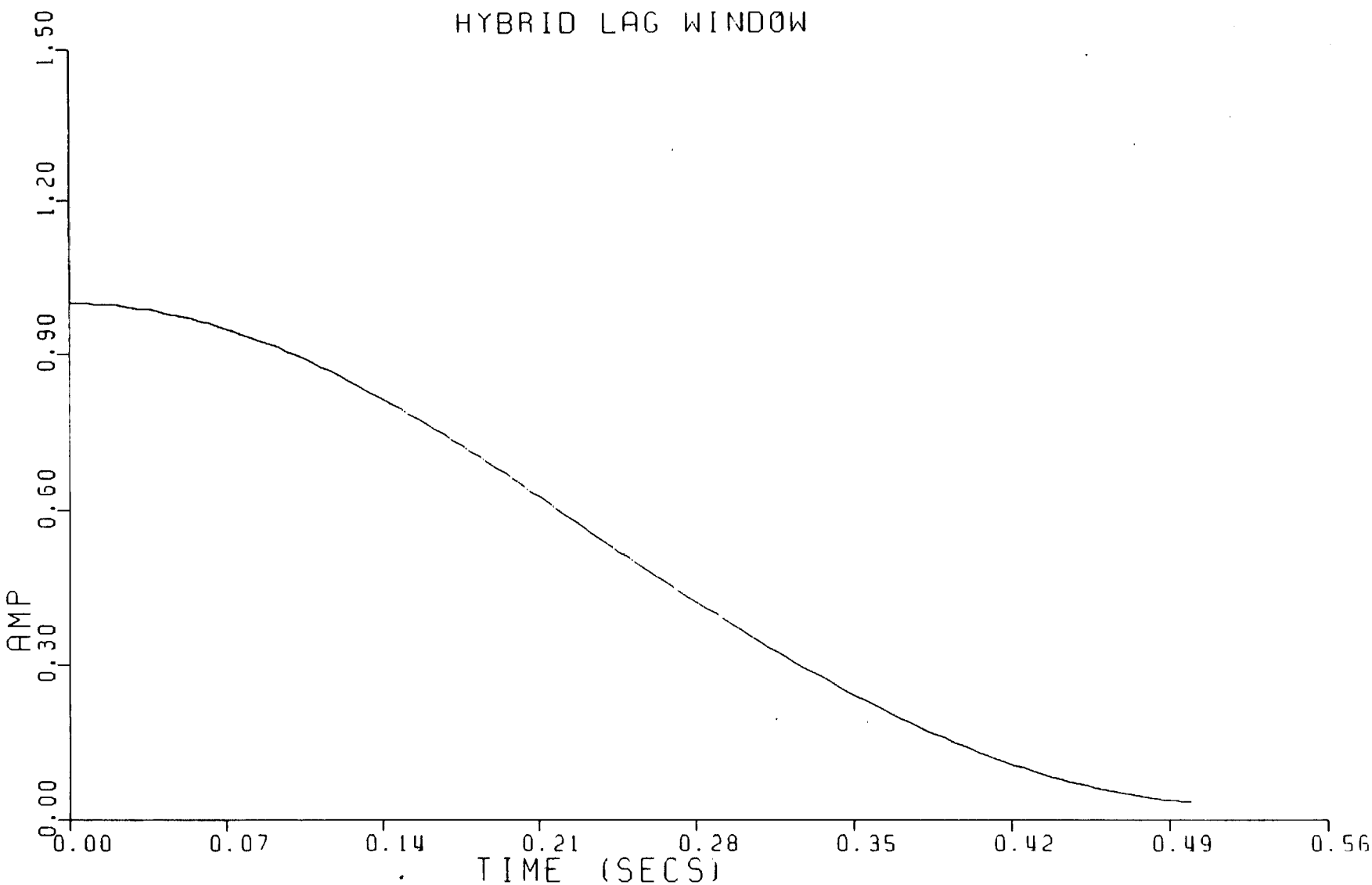


Figure 63 PRGRM-2, 2-8 HZ

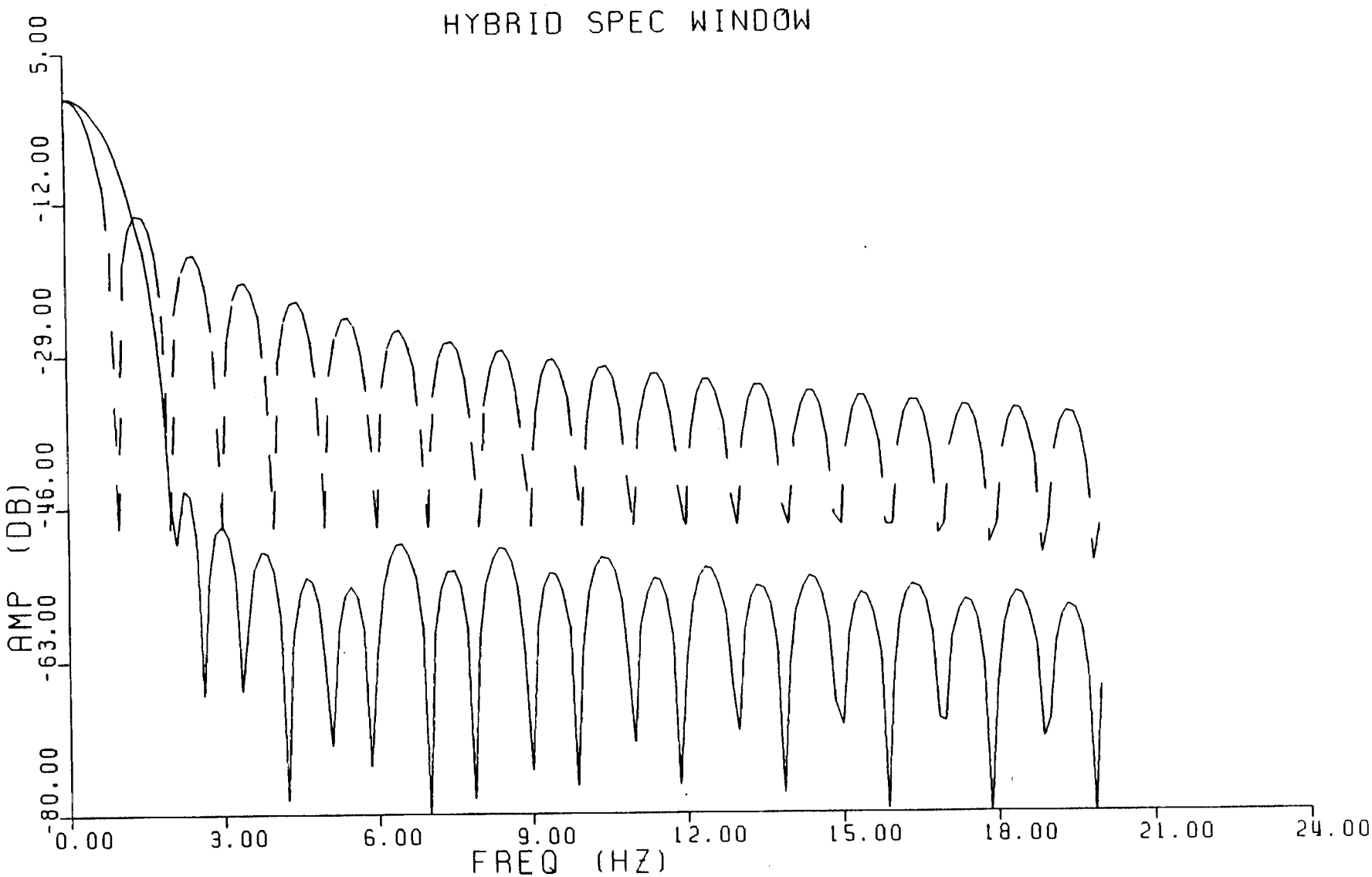


Figure 64 PRGRM-2, 2-8 HZ

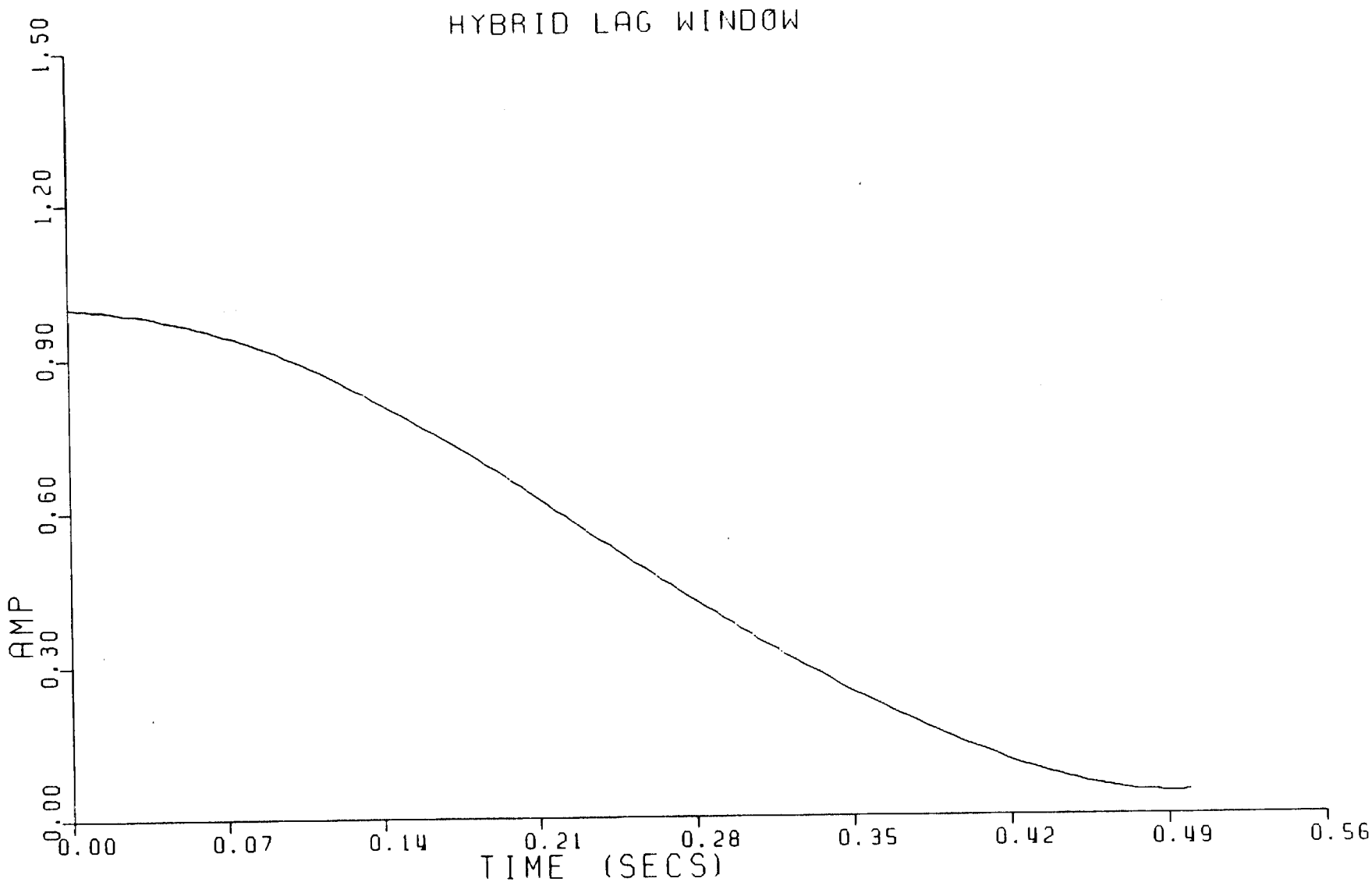


Figure 65 PRGRM-3, 2-8 HZ

HYBRID, 3 WINDOWS

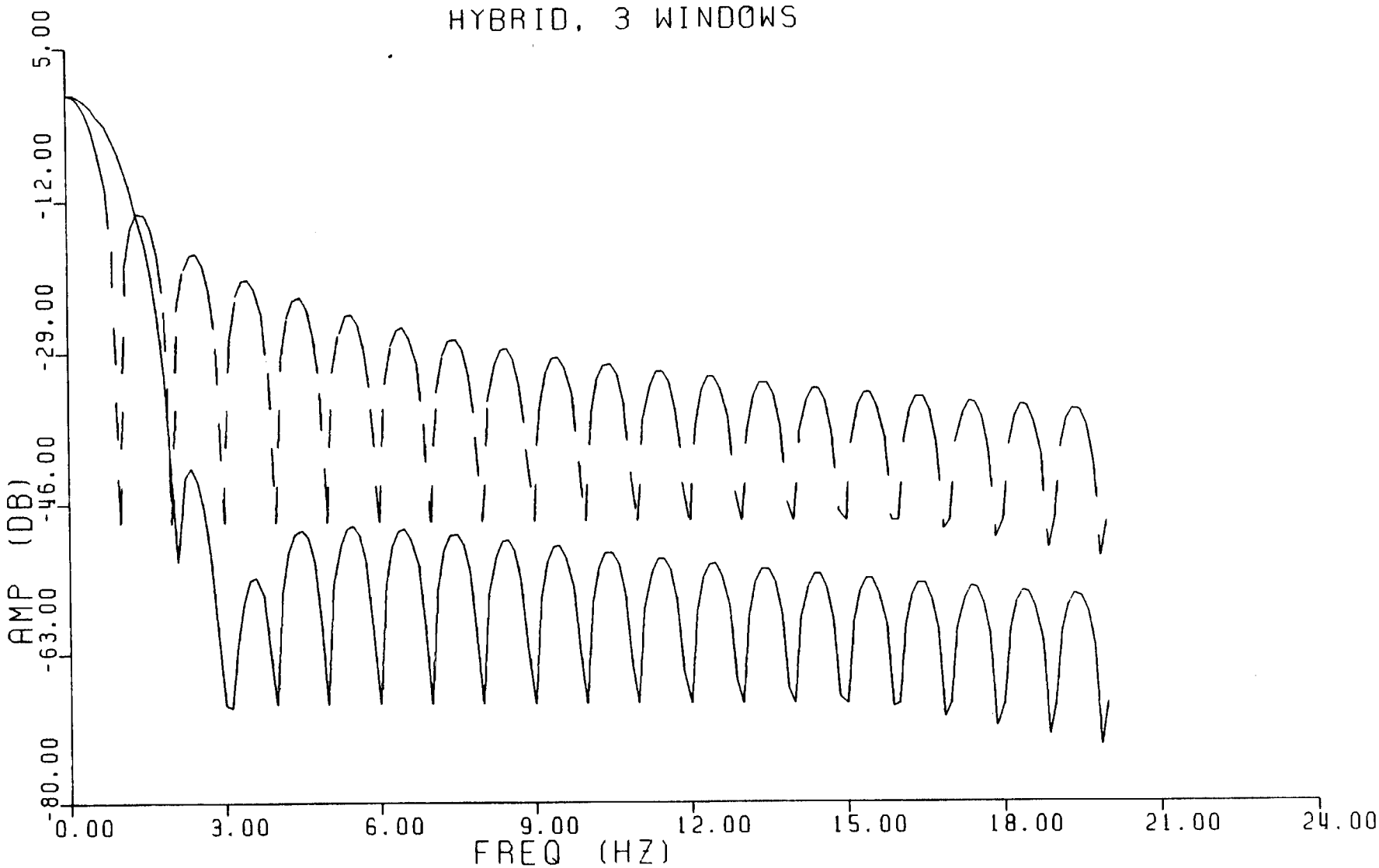


Figure 66 PRGRM-3, 2-8 HZ

LAG, 3 WINDOWS

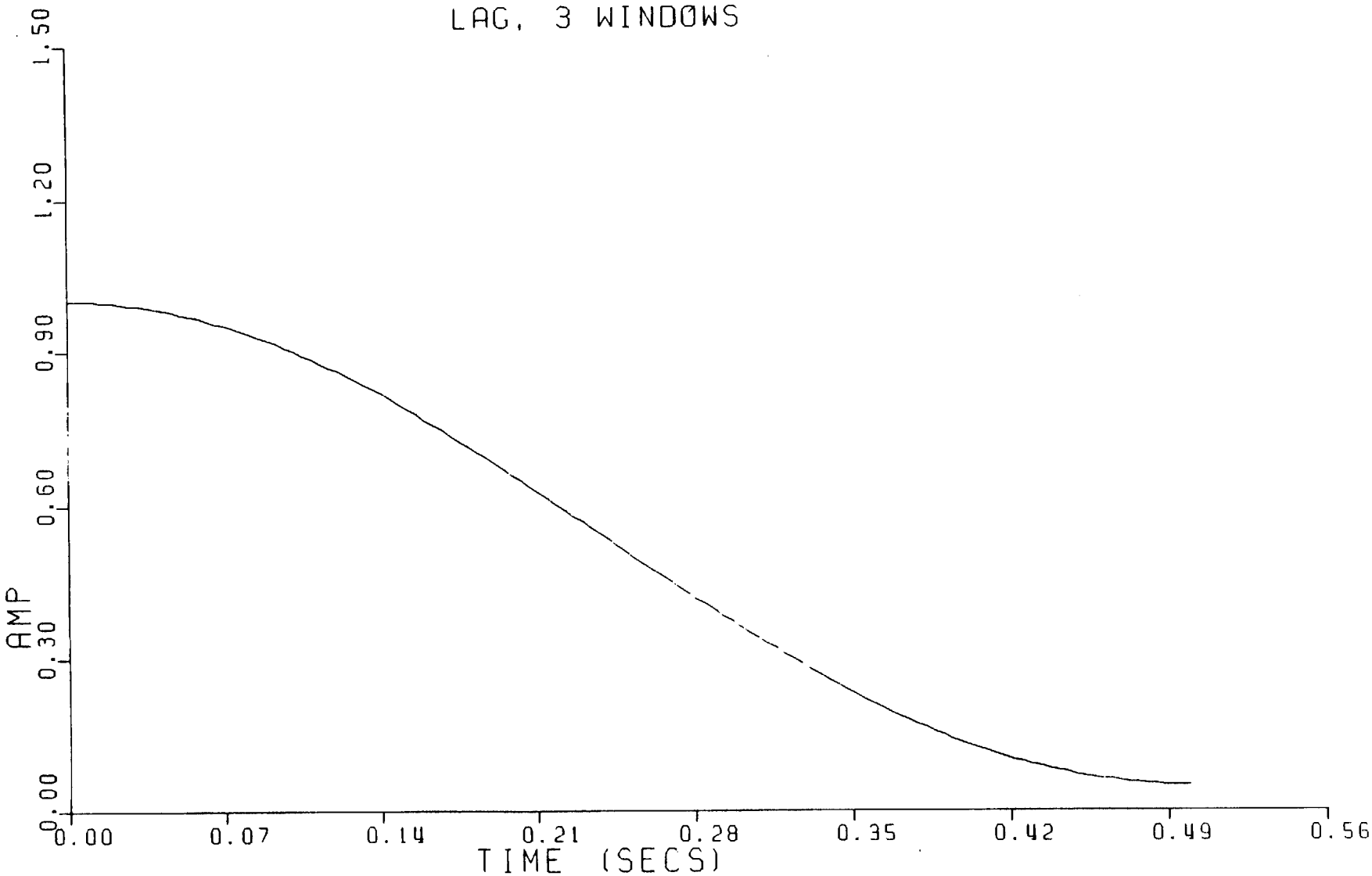


Figure 67 PRGRM-3, 2-8 HZ

HYBRID, 6 WINDOWS

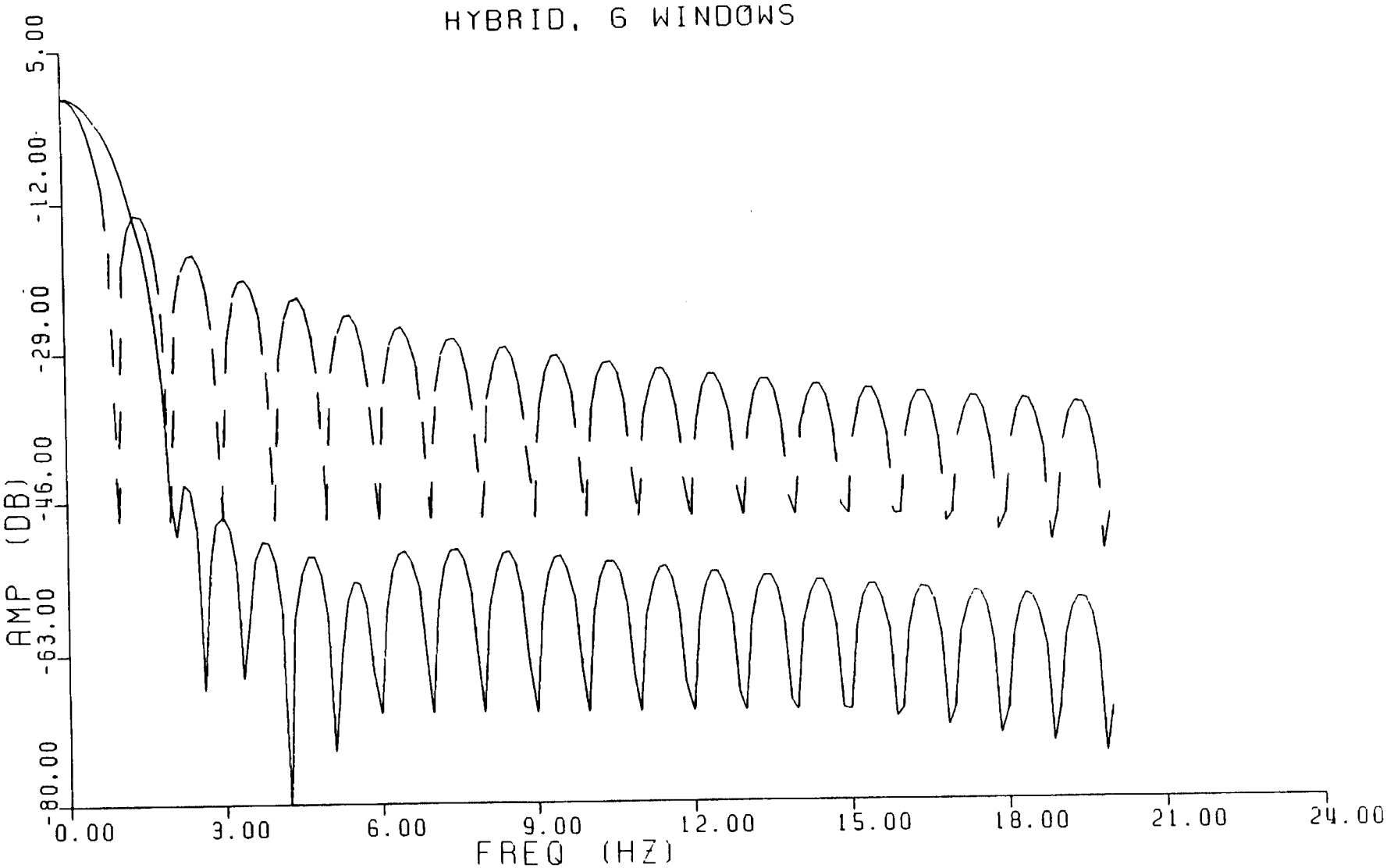


Figure 68 PRGRM-3, 2-8 HZ

LAG, 6 WINDOWS

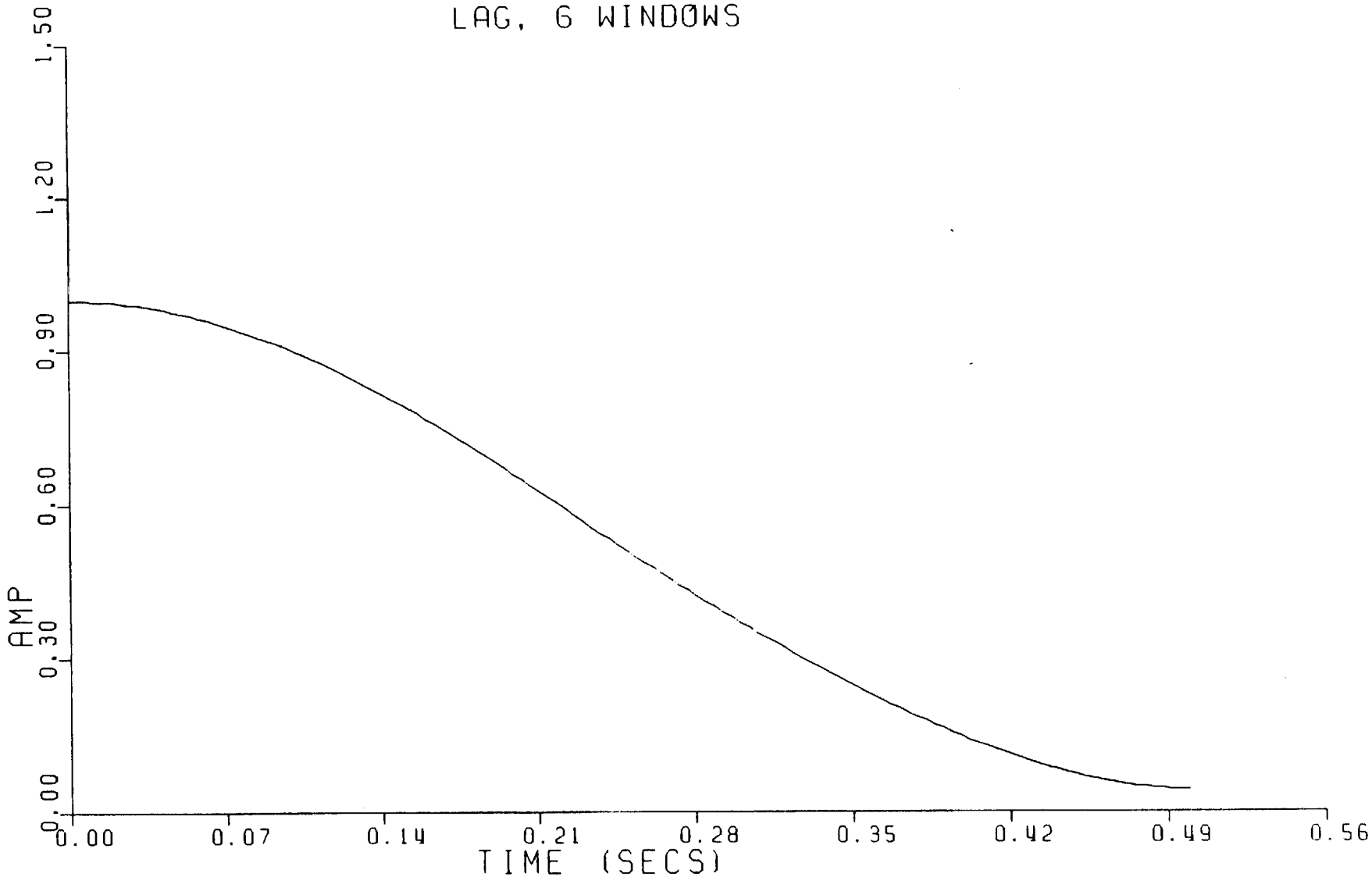


Figure 69 PRGRM-3, 2-8 HZ

HYBRID, 9 WINDOWS

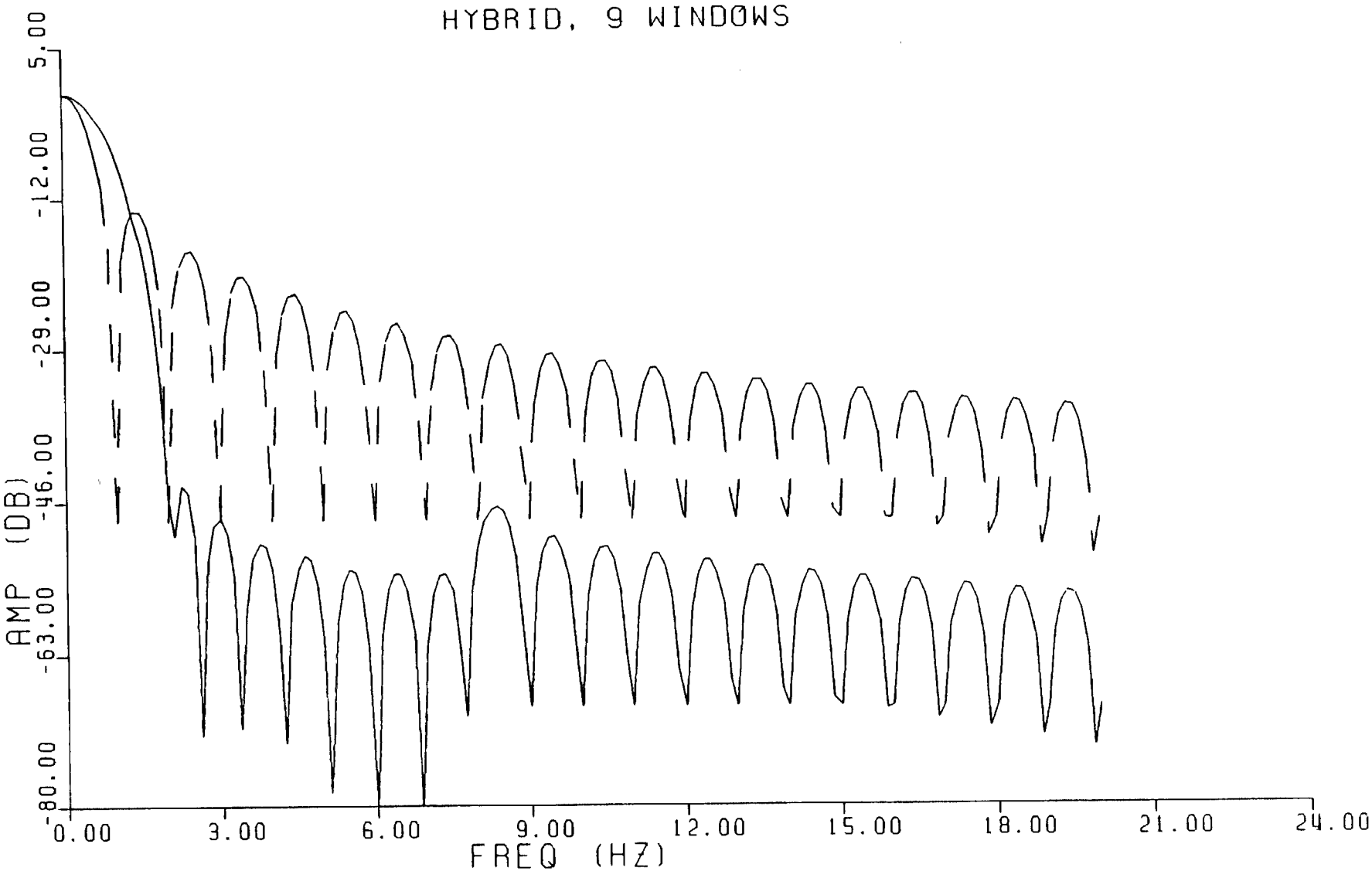


Figure 70

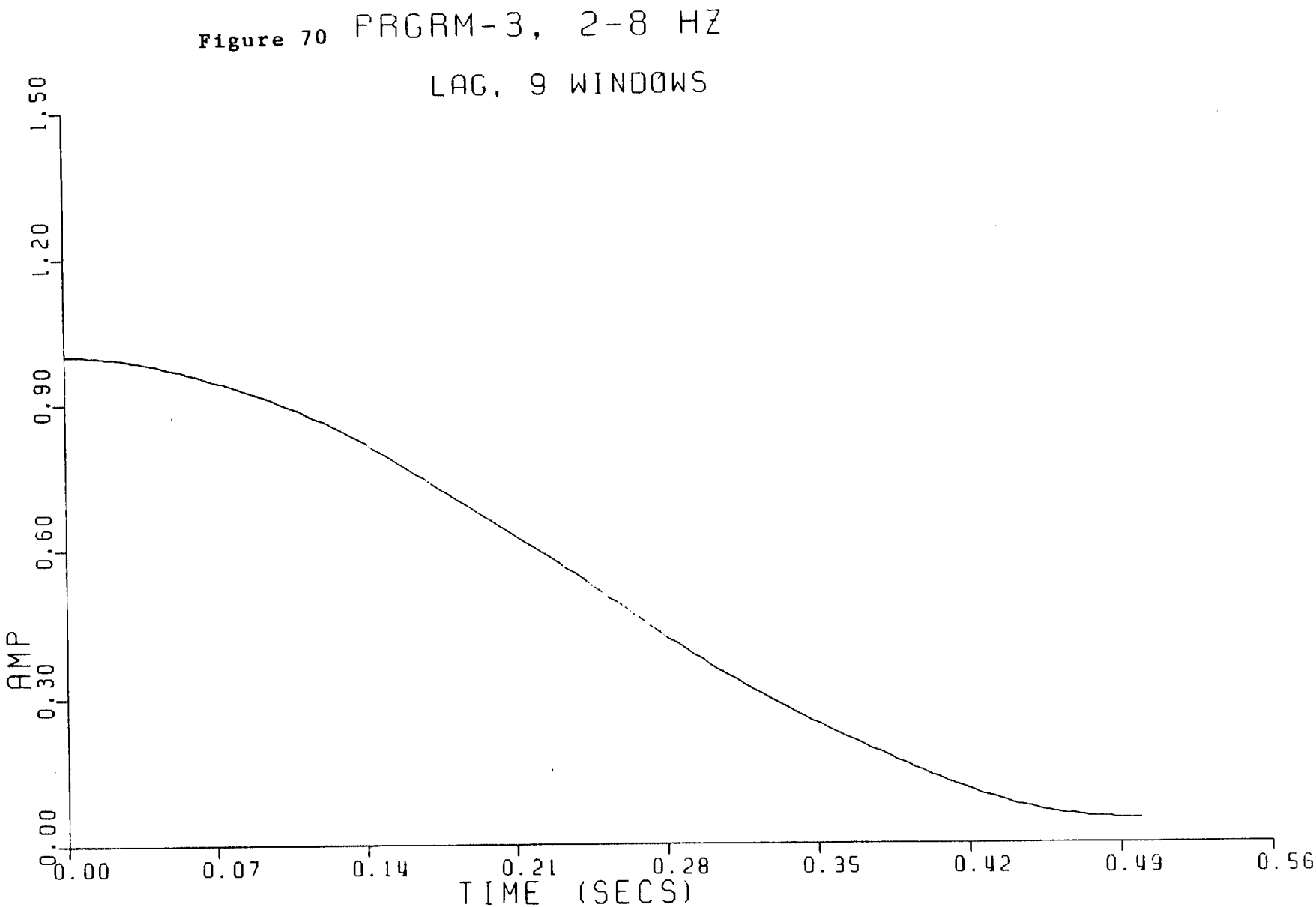

PRGRM-3, 2-8 HZ
LAG, 9 WINDOWS
AMP
0.00
0.30
0.60
0.90
1.20
1.50
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
0.56
TIME (SECS)

Figure 71 PRGRM-1, 4-8 HZ

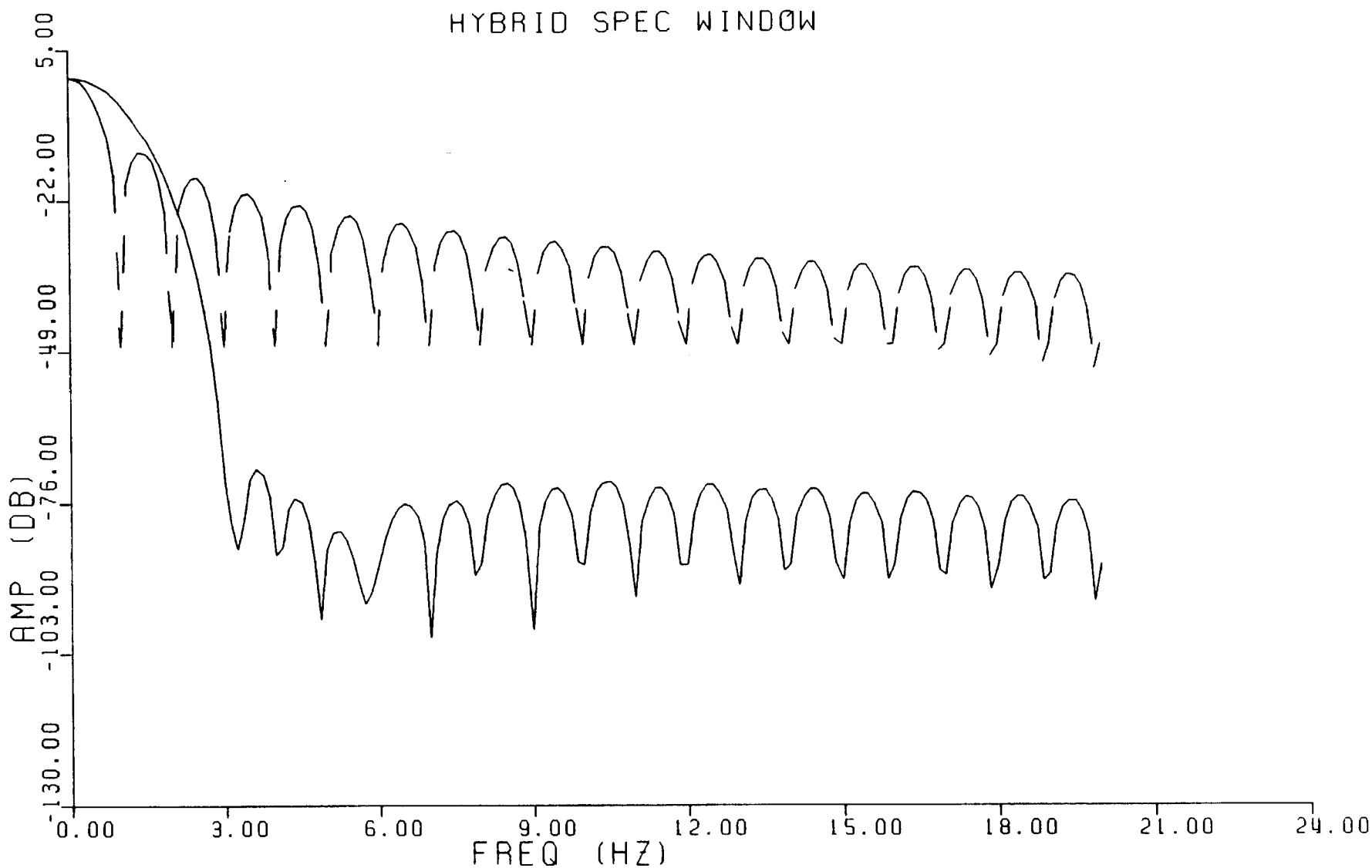


Figure 72 PRGRM-1. 4-8 HZ

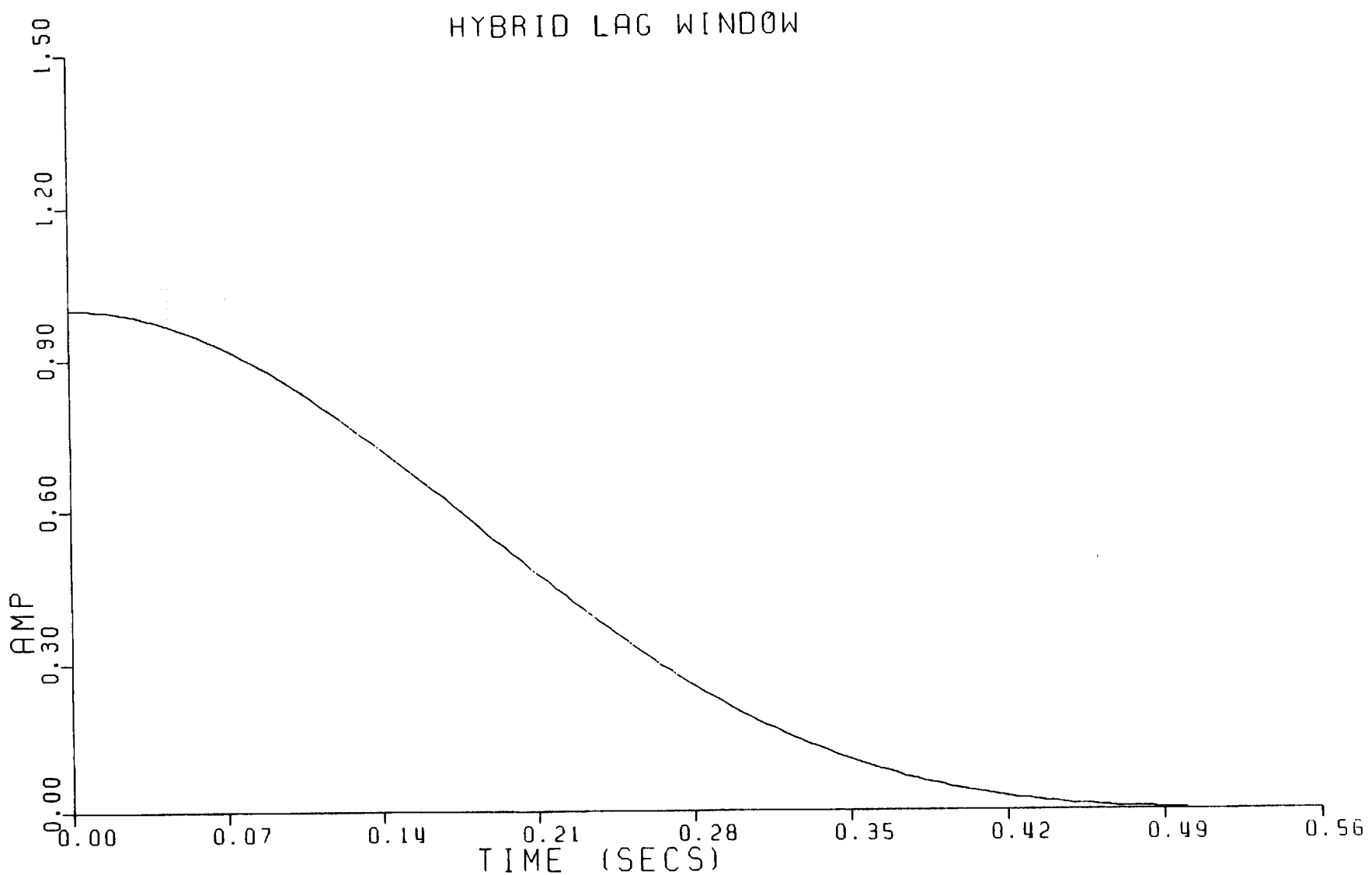


Figure 73

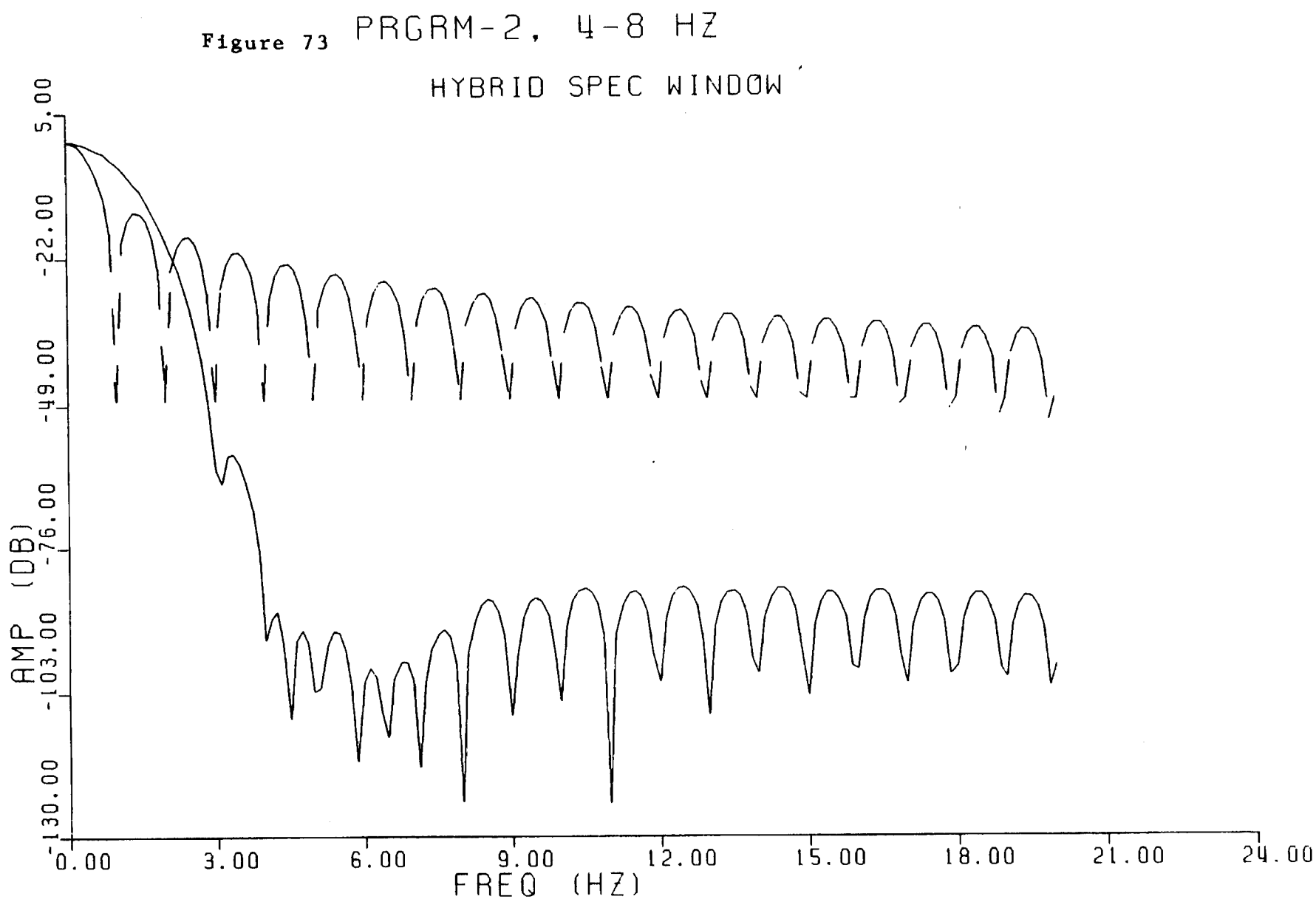

PRGRM-2, 4-8 HZ
HYBRID SPEC WINDOW
AMP (DB)
5.00
-22.00
-49.00
-76.00
-103.00
-130.00
0.00
3.00
6.00
9.00
12.00
15.00
18.00
21.00
24.00
FREQ (HZ)

Figure 74 PRGRM-2, 4-8 HZ

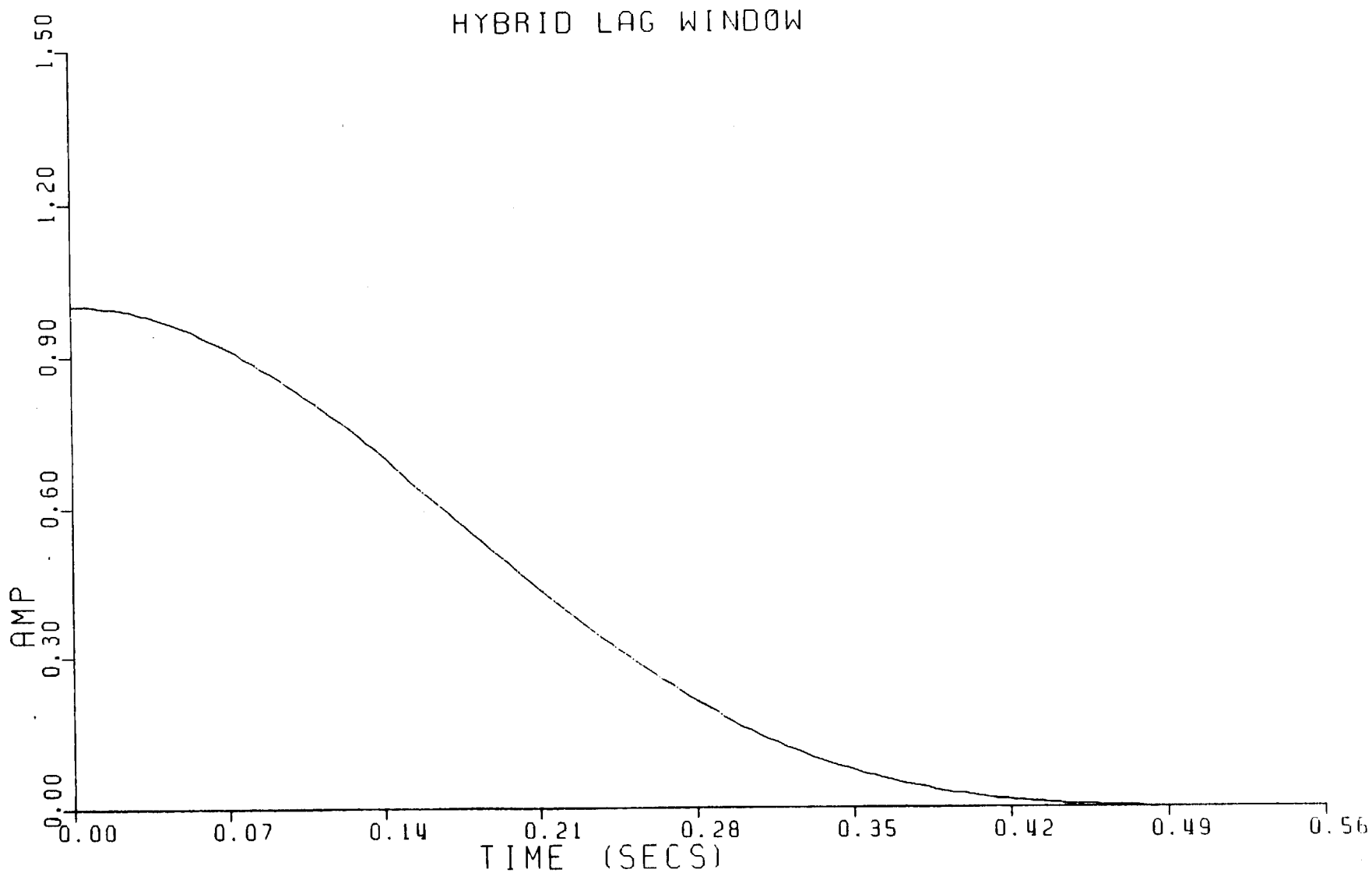
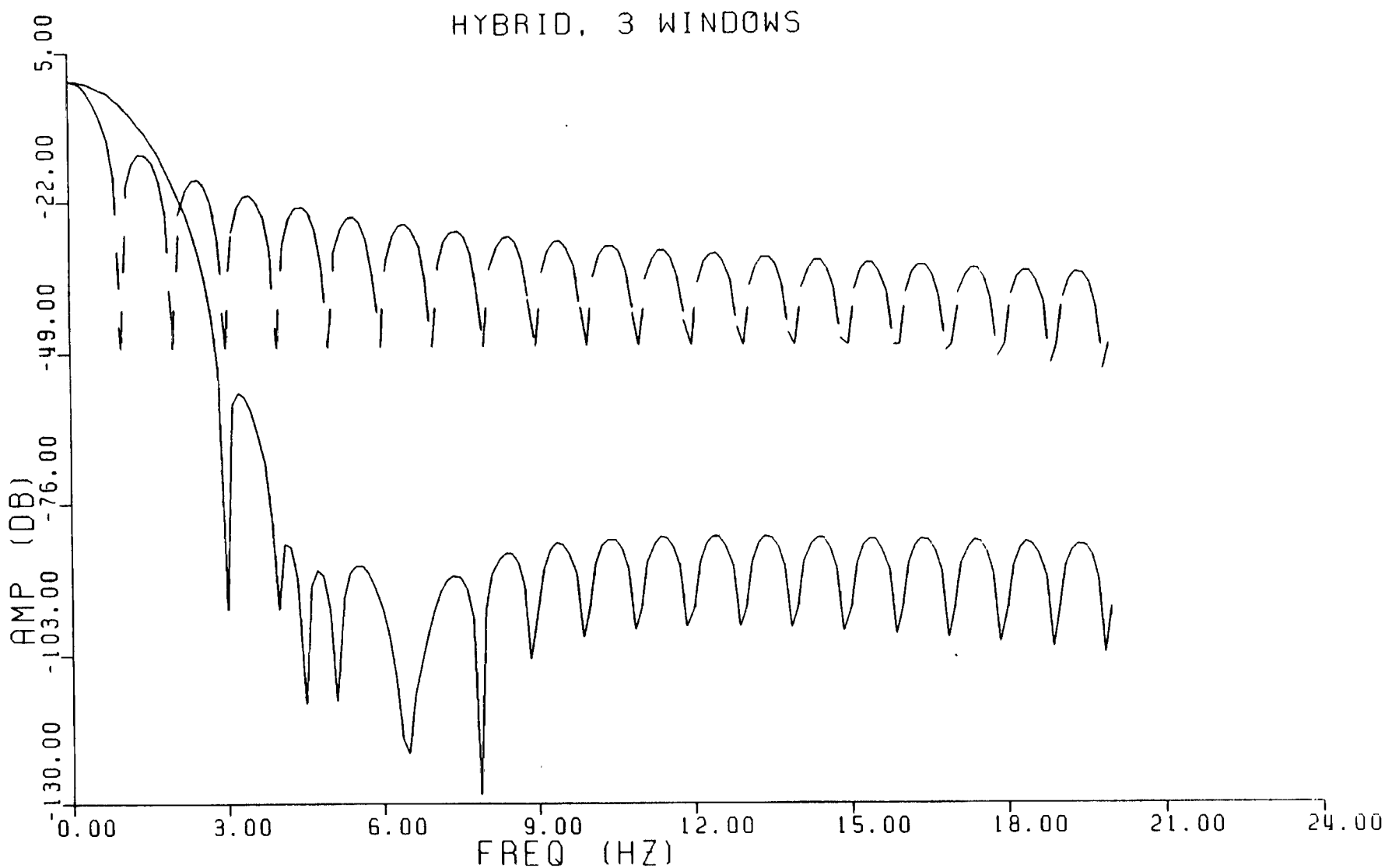


Figure 75 PRGRM-3, 4-8 HZ



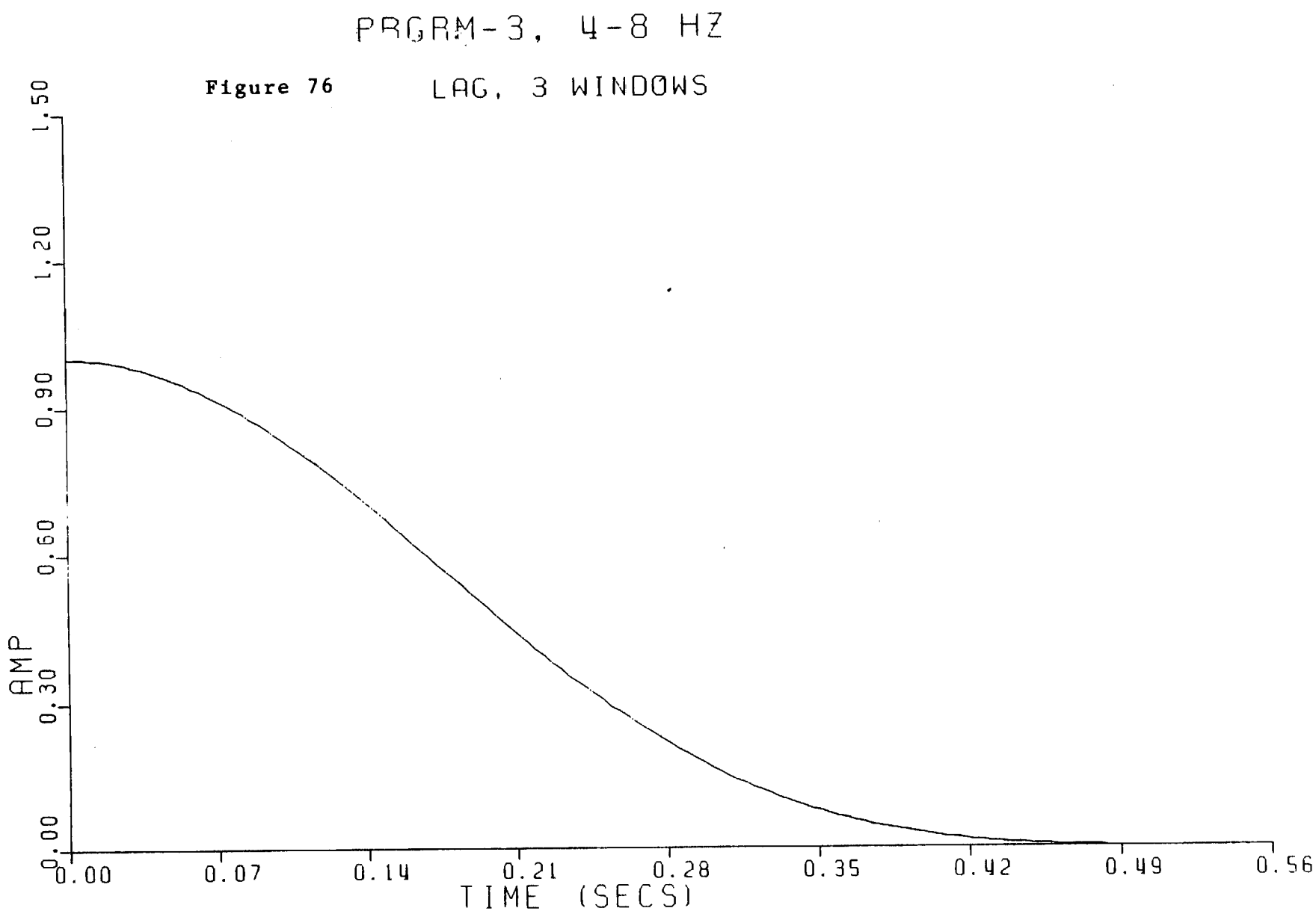

PRGRM-3, 4-8 HZ
LAG, 3 WINDOWS
AMP
0.00
0.30
0.60
0.90
1.20
1.50
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
0.56
TIME (SECS)

Figure 76

Figure 77 PRGRM-3, 4-8 HZ

HYBRID, 6 WINDOWS

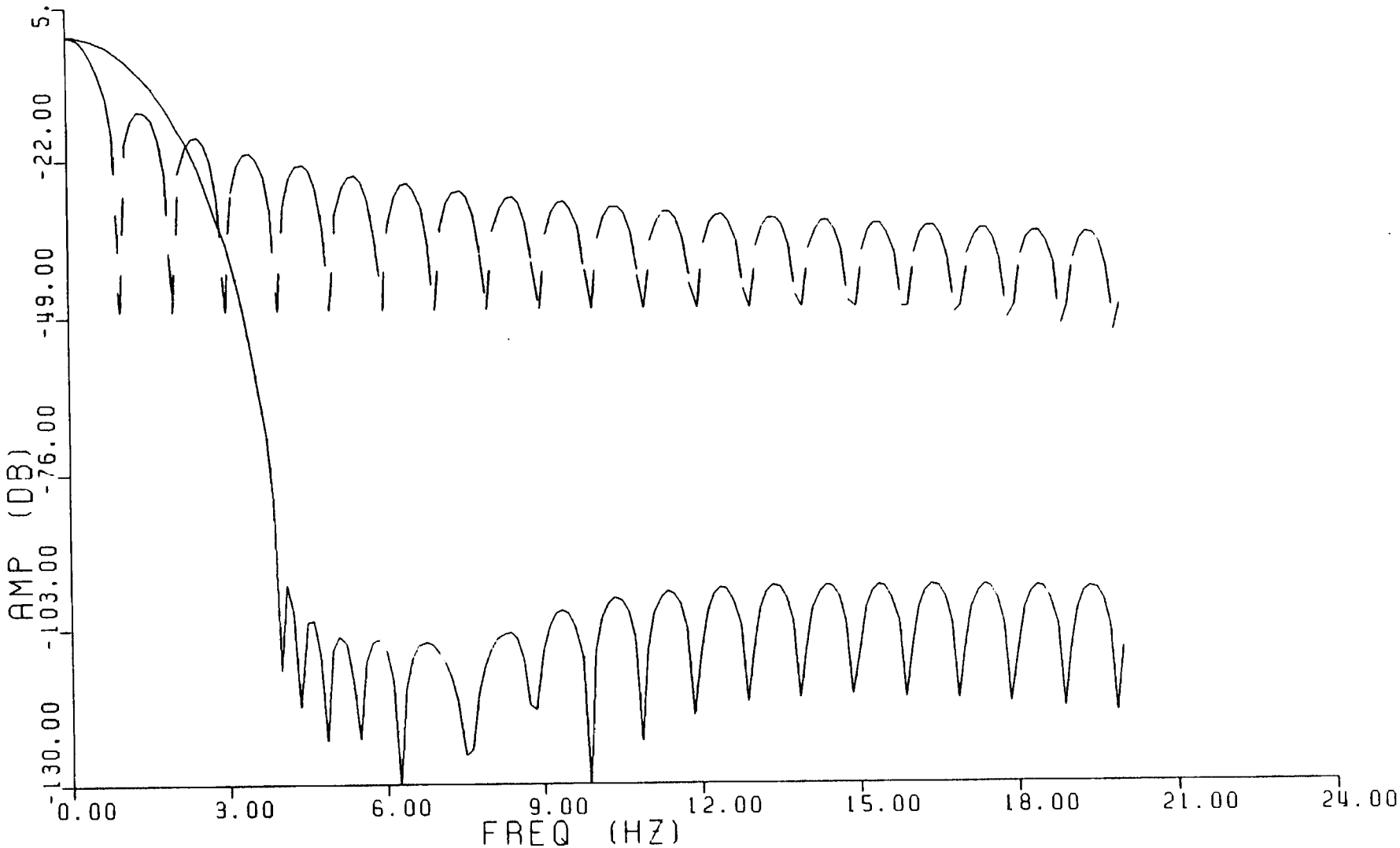


Figure 78 PRGRM-3, 4-8 HZ

LAG, 6 WINDOWS

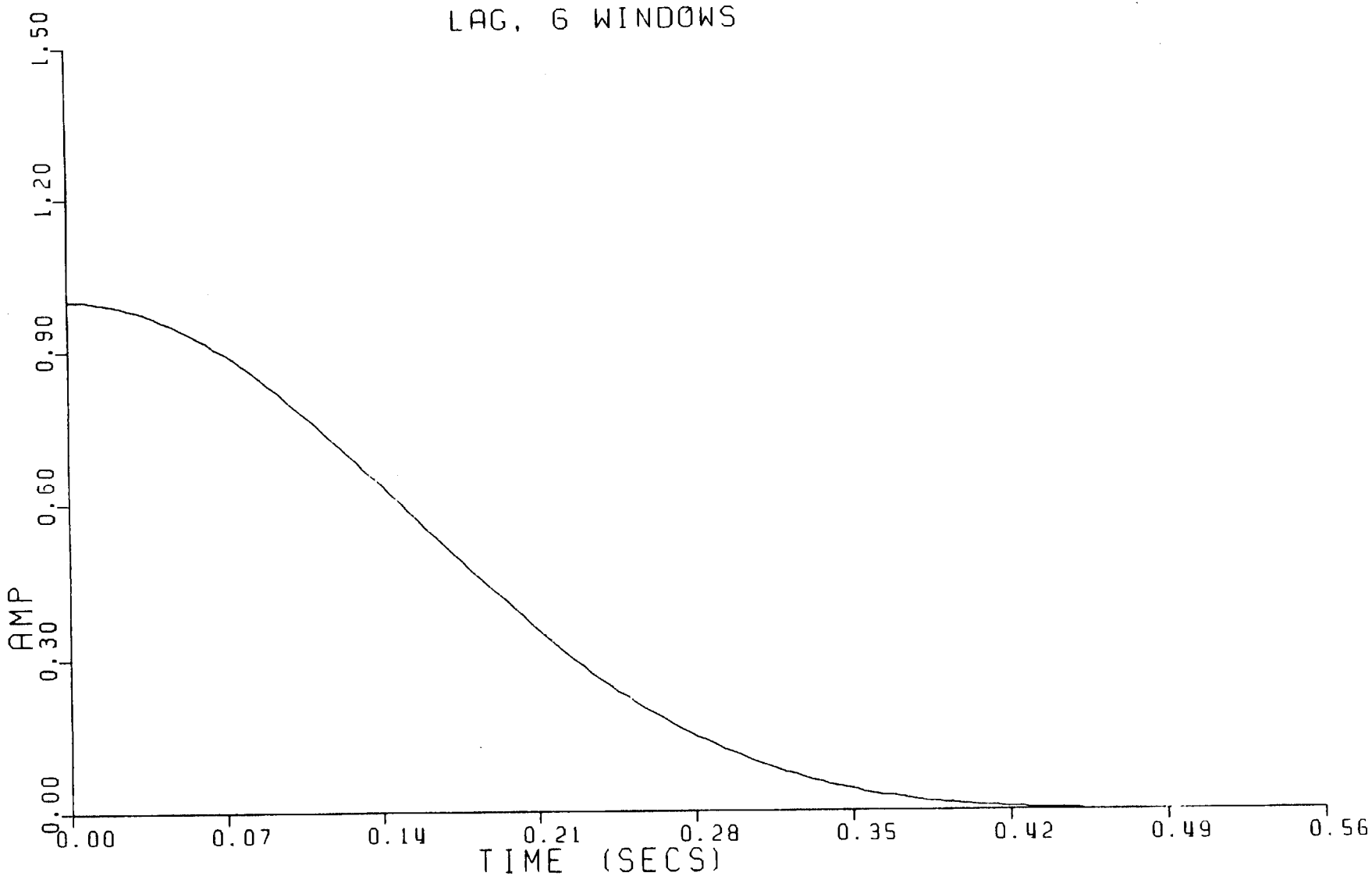


Figure 79 PRGRM-3, 4-8 HZ

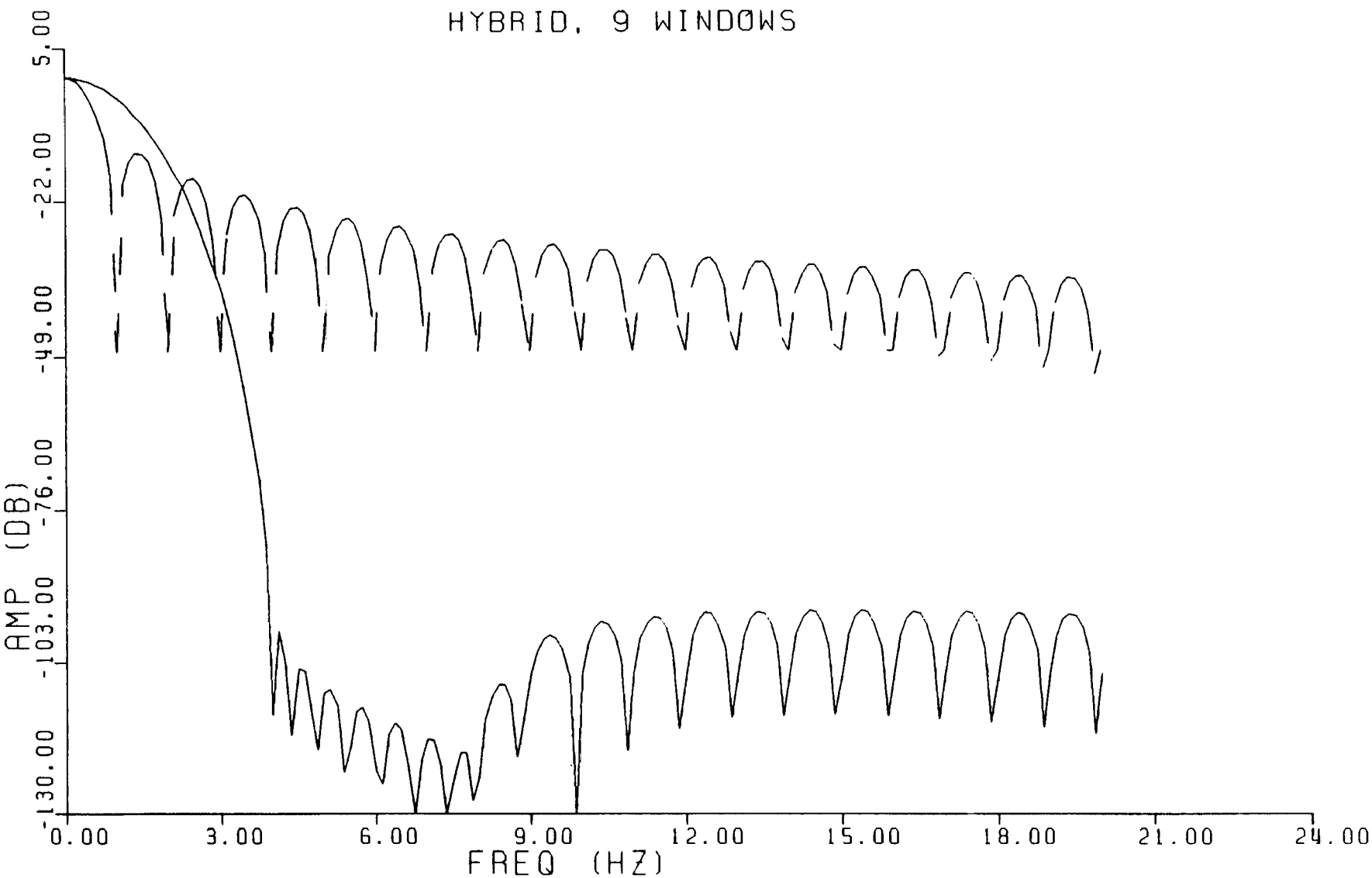
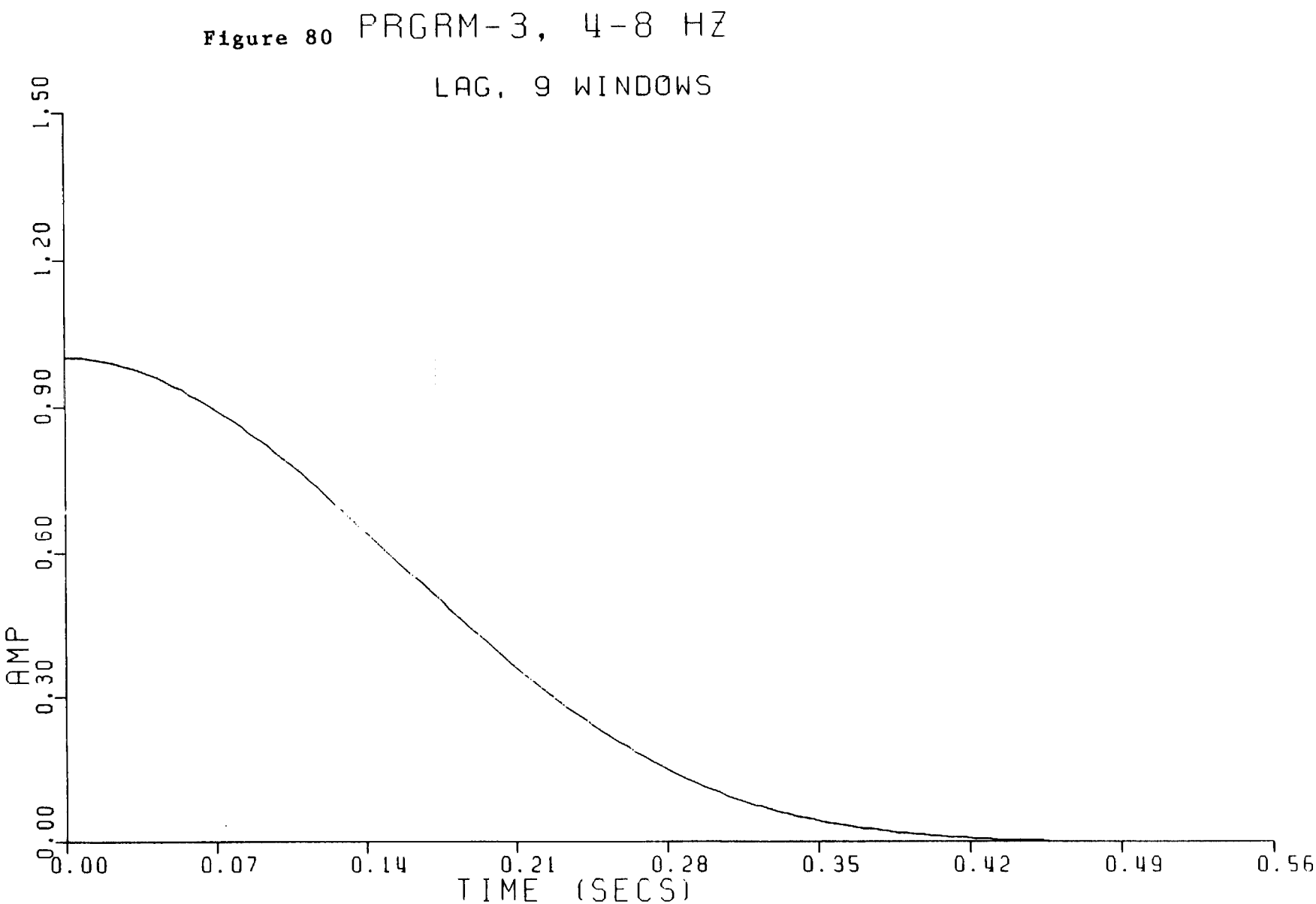


Figure 80


PRGRM-3, 4-8 HZ
LAG, 9 WINDOWS
AMP
0.00
0.30
0.60
0.90
1.20
1.50
0.00
0.07
0.14
0.21
0.28
0.35
0.42
0.49
0.56
TIME (SECS)

REFERENCES

Blackman, R. B., and Tukey, J. W. (1958), Measurement of Power Spectra, (Dover: New York).

Bracewell, R. N. (1978), The Fourier Transform and Its Applications, (New York: McGraw-Hill).

Bronson, R. (1982), Schaum's Outline of Theory and Problems of Operations Research, (New York: McGraw-Hill).

Claerbout, J. F. (1976), Fundamentals of Geophysical Data Processing, (New York: McGraw-Hill).

Goldfarb, D., and Idnani, A. (1983), "A Numerically Stable Dual Method for Solving Strictly Convex Quadratic Programs", Mathematical Programming, 27, 1-33.

Hillier, F. S., and Lieberman, G. J. (1967), Introduction to Operations Research, (San Francisco: Holden-Day, Inc).

Kreamer, J. L. (1988), Expanded Analysis of Combined Widow Spectral Estimate Channel Clearing, Master of Science Thesis, Dept. of Physics: University of New Orleans, Louisiana.

Lawson, C. L., and Hanson, R. J. (1974), Solving Least Squares Problems, (New Jersey: Prentice-Hall, Inc.).

Marple, S. L. (1987), Digital Spectral Analysis, (New Jersey: Prentice-Hall, Inc.).

Nuttall, A. H. (1981), "Some Windows With Very Good Sidelobe Behavior", IEEE V. ASSP-29, No. 1, 84-91.

Oppenheim, A. V. and, Schafer, R. W. (1975), Digital Signal Processing, (New Jersey: Prentice-Hall Inc.).

Powell, M. J. D. (1983a), "On the Quadratic Programming Algorithm of Goldfarb and Idnani", DAMTP Report NA19, Cambridge, England.

Powell, M. J. D. (1983b), "ZQPCVX - A FORTRAN Subroutine for Convex Quadratic Programming", DAMTP Report NA17, Cambridge, England.

Robinson, E. A. (1980), Physical Applications of Stationary Time Series, (New York: McMillan Publishing).

Scales, L. E. (1985), Introduction to Non-Linear Optimization, (New York: Springer-Verlag).

Smith, H. M. (1985), Creating Clear Channels by Minimizing Sidelobes in Power Spectral Estimates Using Linear Combinations Of Autocorrelation Windows, Master of Science Thesis, Dept. of Physics: University of New Orleans, Louisiana.

APPENDIX A

In this appendix, we will show that the vector **a** that minimizes the unconstrained function $\mathbf{a}^T\mathbf{H}\mathbf{a}$ is $\mathbf{a}^* = 0$. The treatment is after Scales (1985). Consider a Taylor expansion for $F(\mathbf{x})$

$$(1) \qquad F(\mathbf{x} + \Delta\mathbf{x}) = F(\mathbf{x}) + \Delta\mathbf{x}^T\mathbf{g}(\mathbf{x}) + 1/2\ \Delta\mathbf{x}^T G(\mathbf{x})\Delta\mathbf{x} + \ldots,$$

where the gradient is $\mathbf{g}(\mathbf{x}) = \begin{bmatrix} \partial F/\partial x_1 \\ \vdots \\ \partial F/\partial x_n \end{bmatrix}$

and the Hessian is $G(x) = \begin{bmatrix} \partial^2 F/\partial x_1 \partial x_1 & \ldots & \partial^2 F/\partial x_1 \partial x_n \\ \vdots & & \vdots \\ \partial^2 F/\partial x_n \partial x_1 & \ldots & \partial^2 F/\partial x_n \partial x_n \end{bmatrix}$.

Now consider a quadratic form

$$(2) \qquad F(\mathbf{x}) = 1/2\ \mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{b}^T\mathbf{x} + c.$$

Then $\quad F(\mathbf{x} + \Delta\mathbf{x}) = 1/2\ (\mathbf{x} + \Delta\mathbf{x})^T\mathbf{A}(\mathbf{x} + \Delta\mathbf{x}) + \mathbf{b}^T(\mathbf{x} + \Delta\mathbf{x}) + c$

$$= F(\mathbf{x}) + \Delta\mathbf{x}^T(\mathbf{A}\mathbf{x} + \mathbf{b}) + 1/2\ \Delta\mathbf{x}^T\mathbf{A}\Delta\mathbf{x}.$$

Comparing (2) with (1), we have that

$$\mathbf{g}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b}$$

and

$$G(\mathbf{x}) = \mathbf{A}.$$

The condition for **x** to be a stationary point is that

$$g(x^*) = 0$$

or

$$Ax^* = -b.$$

Casting this result in the notation of our problem gives

$$Ha^* = -b.$$

But,

$$b = 0,$$

so

$$Ha^* = 0.$$

This linear system is consistent, with a solution $a^* = 0$. Since H is square, if it is nonsingular, the solution is unique, and 0 is the solution. If H is positive definite it is nonsingular. We actually require H to be positive definite although we only showed in Chapter 2 that it was nonnegative semidefinite. DQPROG will perturb the diagonals so that H becomes positive definite if it is not.

APPENDIX B

In this appendix we will first give some useful basic information about the specific quadratic programming (QP) subroutine we used to generate the thesis results, and then we will give a broad outline of the typical steps to solving a QP problem.

In all three programs discussed in the body of this thesis the heart of each code was a subroutine from the IMSL MATH$^{T.M.}$/LIBRARY called DQPROG. DQPROG is the double precision version of QPROG which is dicussed next.

QPROG is based on M. J. D. Powell's implementation of the Goldfarb and Idnani (1983) quadratic programming algorithm for QP problems subject to general linear equality and inequality constraints. The matrix **H** (discussed in Chapter 2) is required to be positive definite. If it is not, then a positive definite perturbation is used in place of H. For more details, see Powell (1983a, 1983b).

The following broad outline of quadratic programming is after Bronson (1982). The solution of many mathematical programming problems begins with the Kuhn-Tucker conditions. Consider the following general nonlinear programming problem

given that	$\mathbf{x} = [x_1, \ldots, x_n]^T$,	
maximize	$z = f(\mathbf{x})$	
subject to	$g_i \leq 0$,	$i = 1, m$
with	$\mathbf{x} \geq 0$.	

To solve this program, rewrite the nonnegativity constraints as

$$-x_i \leq 0,$$

and add slack variables to the left hand sides of the constraints. This converts the inequality constraints into equality constraints. Now form the Lagrangian function

$$L = f(\mathbf{x}) - \sum_{i=1}^{m} \lambda_i [g_i(\mathbf{x}) + (x_{n+1})^2] - \sum_{i=m+1}^{m+n} \lambda_i [-x_i + (x_{n+i})^2],$$

where the λ_i are Lagrange multipliers. Finally, solve the system

$$\partial L/\partial x_j = 0, \qquad j = 1, 2n+m$$

$$\partial L/\partial \lambda_i = 0, \qquad i = 1, m+n$$

$$\lambda_i \geq 0, \qquad i = 1, m+n .$$

These last three expressions constitute the Kuhn-Tucker conditions. They are important because the set of their solutions contains the solution to our original nonlinear programming problem.

The QP problem is a special case of the general nonlinear programming problem. Let's write the QP problem in the following form (note, seeking the maximum extremum is quite general in that the maxima of $f(\mathbf{x})$ are the minima of $-f(\mathbf{x})$)

maximize $\qquad z = \mathbf{x}^T C \mathbf{x} + D^T \mathbf{x},$

subject to $\qquad \mathbf{A}\mathbf{x} \leq \mathbf{B},$

with $\qquad \mathbf{x} \geq 0.$

After applying the Kuhn-Tucker conditions to the QP problem, we find that the solution must satisfy the new matrix equation

$$A'Y = B',$$

where

$$A' = \begin{bmatrix} A & I_1 & 0_1 & 0_2 \\ -2C & 0_3 & -I_2 & A^T \end{bmatrix},$$

$$B' = \begin{bmatrix} B \\ D \end{bmatrix},$$

and

$$Y' = \begin{bmatrix} x \\ S \\ U \\ v \end{bmatrix}.$$

S is a slack variable vector and U and V are Lagrange multiplier vectors. We also have the condition that

$$Y' \geq 0.$$

Bronson (1982) discussed the use of the method of Frank and Wolfe to solve these equations. This method is rather complicated and beyond the scope of this thesis. However, in a few words, we can generally describe what takes place. A subproblem of our Kuhn-Tucker conditions is a modified linear programming (LP) problem with the inequality constraints of our QP problem. A well known method of solving LP problems is the Simplex method. A series of iterations that involves application of the Simplex method will give the desired solution if it exists.

APPENDIX C

```
      PROGRAM PRGRAM1

C Author: Kent Broadhead (12/89).

C****************************
      INTEGER STRINGLEN,SIZE
      PARAMETER (IFFTMAX=2048,IMAXCHNL=50)
      INTEGER LDA,LDH,NCONMAX,NCON,NEQ,NVARMAX,NVAR
      PARAMETER (NCONMAX=12,NEQ=1,NVARMAX=11
     1           ,LDA=NCONMAX,LDH=NVARMAX)
      CHARACTER*20,FILEOUT,DIGNSTC,TIME
      CHARACTER QUESTN*1
      DIMENSION IARRAY(11)
      REAL*8 AME(IMAXCHNL,NVARMAX),
     1           AMET(NVARMAX,IMAXCHNL)
      DIMENSION SCALE(NVARMAX),SUMARRY(IFFTMAX)
     1          ,SUMDB(IFFTMAX)
      REAL*8 A(LDA,NVARMAX),ALAMDA(NVARMAX)
     1       ,B(NCONMAX),DIAG,G(NVARMAX),
     1        H(LDH,LDH),SOL(NVARMAX)
      COMPLEX   CW(IFFTMAX)
      INTEGER BEGCHNL,ENDCHNL
      INTEGER IACT(NVARMAX),K,NACT,NOUT
      EXTERNAL DQPROG,UMACH
C**********************************

      PRINT *,'Enter the output file name:'
      READ (*,'(A20)')FILEOUT
      OPEN(5,FILE=FILEOUT,STATUS='NEW')

C------------Window only option----------------------
      PRINT *,'Do you just  want a window plot?
     1 (Y or RETURN):'
      READ(*,'(A1)'),QUESTN
      IF (QUESTN .EQ. 'Y' .OR. QUESTN .EQ.'y') THEN
        CALL WNDONLY(CW,SUMDB,ILAG1,SUMARRY)
        GO TO 1050
      ENDIF
C----------------------------------------

C-------------------------------------------------------
C Get the main input params. No values for the arguments
C are sent. Both returned by GETLIST.
      CALL GETLIST(IARRAY,NVAR)
C      WRITE(*,'(1X,I2,4X,I2)'), (IARRAY(I),I,I=1,NVAR)
C -------------------------------------------------------


C      PRINT *, 'Enter no. of windows to use (2-11):'
C      ACCEPT *, NVAR
```

```
C SCALE(i) sums to unity constraint
      DO 302 J=1,NVAR
302      A(1,J)=1.0

C Open diagnostics file;
      SIZE=STRINGLEN(FILEOUT,LEN(FILEOUT))
      DIGNSTC = FILEOUT(1:SIZE)//'.DIAG'
      OPEN(4,FILE=DIGNSTC,STATUS='NEW')
      WRITE(4,*),'Diagnositics File'
      WRITE(4,*) ' '
      WRITE(4,*),'VAX filename = PRGRM1.FOR'
      WRITE(4,*) ' '
      WRITE(4,444),FILEOUT
444   FORMAT(' Data filename = ',A20)
      WRITE(4,*) ' '

C Windows range from -.5 to .5
C Get 0 to .5 first, then use symm to pad
C out neg side (in FFT format).


C Initialize
      ICNT = 1

      PRINT *, 'No. smples for lag wndw (desgn)'
          READ(*,*),ilag1
      PRINT *, 'No. smples for spec wndw (desgn)'
          READ(*,*),ifftnm1

      XLAG1 = ILAG1
      XFFTNM1 = IFFTNM1
      DELTAF1 = XLAG1/XFFTNM1

      PRINT *, 'No. smples for lag wndw (dsply)'
          READ(*,*),ilag2
      PRINT *, 'No. smples for spec wndw (dsply)'
          READ(*,*),ifftnm2

      XLAG2 = ILAG2
      XFFTNM2 = IFFTNM2

      DELTAF2 = XLAG2/XFFTNM2

      ihalf1 = ilag1/2 + 1
      ihalf2 = ilag2/2 + 1
      NYQst1 = ifftnm1/2 + 1
      NYQST2  = ifftnm2/2 + 1

      WRITE(4,1010),ilag1
1010  FORMAT(1X,'No. smples for lag wndw (desgn)',I4)
      WRITE(4,1011),ifftnm1
1011  FORMAT(1X,'No. smples for spec wndw (desgn)',I4)
```

```
      WRITE(4,1020),ilag2
1020  FORMAT(1X,'No. smples for lag wndw (dsply)',I4)
      WRITE(4,1021),ifftnm2
1021  FORMAT(1X,'No. smples for spec wndw (dsply)',I4)
      WRITE(4,*) ' '

      PRINT *,'Enter the begining and ending freqs:'
c      PRINT *, 'Note: max no. of channels
C     1 presently is 50.'
      ACCEPT *,F1,F2
C Cnvert F1 and F2 to fft points.
      CALL CNVRT(BEGCHNL,ENDCHNL,F1,F2,DELTAF1)

      WRITE(4,1000),F1,F2
1000  FORMAT(1X,'Frq atten. band ',F10.3,' to 'F10.3)
      WRITE(4,*) ' '
      WRITE(4,1023),BEGCHNL,ENDCHNL
1023  FORMAT(1X,'Chnl band ',i4,' to 'i4)
      WRITE(4,*) ' '

      NCON = NVAR + 1

C Set up constraints;
      DO 305 ICLR=1,NVARMAX
305      G(ICLR)=0.0D0

      B(1) = 1.0D0
      DO 299 I=2,NCON
299      B(I) = 0.0D0
      DO 301 I=1,NCON
         DO 301 J=1,NVAR
301         A(I,J)=0.0D0
      DO 307 J=1,NVAR
307      A(1,J)=1.0D0
      DO 303 I=1,NVAR
303      A(I+1,I)=1.0D0

C Begin window calcs:
      xlag1 = ilag1
      DO 20 IOUTER=1,NVAR
          CALL CCLEAR(CW,ifftnm1)
C         Compute a window.
          DO 10 I=1,ihalf1
             XI = I
             ARG = (XI-1.)/xlag1
10           CW(I)=WINDOW(ARG,IARRAY(ICNT))
C         Fourier transform the window.
          CALL FFT(CW,ifftnm1)
C         Load the window transform as a column of AME
          DO 40 IL=BEGCHNL,ENDCHNL
             INDEX = IL-BEGCHNL+1
40           AME(INDEX,ICNT) = CW(IL)
          ICNT = ICNT + 1
```

```
20    CONTINUE

C Define M.
      M = ENDCHNL-BEGCHNL+1

C Take transpose.
      DO 12 K=1,M
          DO 50 L=1,NVAR
50          AMET(L,K) = AME(K,L)
12        CONTINUE

C Multiply transpose by orig matrix and store in ATA.
      DO 100 I=1,NVAR
         DO 100 J=1,NVAR
            SUM=0.
            DO 102 K=1,M
102            SUM = SUM + AMET(I,K)*AME(K,J)
100      H(I,J)= 2.*SUM

C       DO 13 K=1,NVAR
C          WRITE(6,'(1X,8F10.5)') (H(K,L), L=1,NVAR)
C13        CONTINUE


C Call DQPROG to solve for the hybrid window scale factors;

C  IMSL Name:  QPROG/DQPROG (Single/Double precision version)
C  Computer:   VAX/SINGLE
C  Revised:    October 15, 1985
C  Purpose:    Solve a quadratic programming problem subject
C              to linear equality/inequality constraints.
C  Usage:      CALL QPROG (NVAR, NCON, NEQ, A, LDA, B, G, H,
C                          LDH,DIAG, SOL, NACT, IACT, ALAMDA)
C  Arguments:
C  NVAR    - The number of variables. (Input)
C  NCON    - The number of linear constraints. (Input)
C  NEQ     - The number of linear equality constraints. (Input)
C  A       - Real NCON by NVAR matrix. (Input)
C            The matrix contains the equality contraints in the
C            first NEQ rows, followed by the inequality
C            constraints.
C  LDA     - Leading dimension of A exactly as specified in the
C            dimension statement of the calling program. (Input)
C  B       - Real vector of length NCON containing right-hand
C            sides of the linear constraints. (Input)
C  G       - Real vector of length NVAR containing the
C            coefficients of the linear term of the objective
C            function.  (Input)
C  H       - Real NVAR by NVAR matrix containing the Hessian
C            matrix of the objective function.  (Input)
C            H should be symmetric positive definite; if H
C            is not positive definite, the algorithm attempts
C            to solve the QP problem with H replaced by a
```

```
C            H + DIAG*I such that H + DIAG*I is positive
C            definite - See Remark 3.
C  LDH     - Leading dimension of H exactly as specified in the
C            dimension statement of the calling program. (Input)
C  DIAG    - Real scalar equal to the multiple of the
C            identity matrix added to H to give a positive
C            definite matrix. (Output)
C  SOL     - Real vector of length NVAR containing solution.
C            (Output)
C  NACT    - Final number of active constraints.  (Output)
C  IACT    - Integer vector of length NVAR containing the indices
C            of the final active constraints in the first
C            NACT positions. (Output)
C  ALAMDA - Real vector of length NVAR containing the
C            Lagrange multiplier estimates of the final
C            active constraints in the first NACT positions.
C            (Output)

      CALL DQPROG(NVAR,NCON,NEQ,A,LDA,B,G,H,LDH,DIAG,SOL,
     1            NACT,IACT,ALAMDA)

C      CALL UMACH(2,NOUT)
      PRINT *,' '
       WRITE(*,*) (SOL(K),K=1,NVAR)
C      WRITE(*,99999) (SOL(K),K=1,NVAR)
C99999 FORMAT(1X,8F10.5)
      SUM=0.
      DO 432 I=1,NVAR
         SCALE(I) = SOL(I)
432      SUM = SUM + SOL(I)
      WRITE(*,'(1X,4HSUM=,F10.5)') SUM

C Put some info in the diag. file;
      WRITE(4,*),'Optimum Window Wts'
      WRITE(4,*) (SCALE(I),I=1,NVAR)
      WRITE(4,*) ' '

      WRITE(4,*) 'The program uses the following percentages
     1 for the given windows;'
      DO 452 IPRCNT=1,NVAR
         PERCNT = SCALE(IPRCNT)*100.
         WRITE(4,453),IARRAY(IPRCNT),PERCNT
453   FORMAT(' WINDOW NO.',I3,' = ',F10.2,' PER CNT')
452   CONTINUE
      WRITE(4,*) ' '

C-----------------------------------------------------
C Now that we have the lin. comb. wts, we need to get
C  the spectrum of the hybrid window. To do this, let's
C  recompute the window functions, scale them,
C  stack them, and FFT.

C Initialize
```

```
      ICNT = 1
      CALL CLEAR(SUMARRY,ifftnm2)

C Begin window calcs:
      xlag2 = ilag2
      DO 22 IOUTER=1,NVAR
C        Compute, scale, and stack a window.
         DO 11 I=1,ihalf2
            XI = I
            ARG = (XI-1.)/xlag2
11          SUMARRY(I)=SUMARRY(I)+SCALE(ICNT)*
     1                  WINDOW(ARG,ICNT)
         ICNT = ICNT + 1
22    CONTINUE
C Put hybrid window (time domain) out to a plot file.
      SIZE=STRINGLEN(FILEOUT,LEN(FILEOUT))
      TIME = FILEOUT(1:SIZE)//'T.DAT'
      OPEN(3,FILE=TIME,STATUS='NEW')

        THALF = 0.5
        XHALF2 = IHALF2
        DELTAT = THALF/(XHALF2 - 1.)
        SUM = 0.0
        T = 0.0
        WRITE(3,*) T,SUMARRY(1)
        DO 405 J5=2,ihalf2
           SUM = SUM + DELTAT
           T = SUM
           WRITE(3,*) T,SUMARRY(J5)
405     CONTINUE


C Fourier transform the window.

      CALL CCLEAR(CW,ifftnm2)
      DO 23 IMOV=1,ihalf2
23       CW(IMOV)=SUMARRY(IMOV)
      CALL FFT(CW,Ifftnm2)
      CALL CLEAR(SUMARRY,ifftnm2)
      DO 24 IMOV=1,NYQST2
24       SUMARRY(IMOV)=CW(IMOV)

C Output Stage:
C-----------Basic set up--------------------------------
      XMAX = SUMARRY(1)
      DO 400 IDB=1,NYQST2
         ARG = ABS((SUMARRY(IDB)/XMAX))
         IF (ARG.EQ.0.0) THEN
           PRINT *,'A zero value was encountered in
     1 the hybrid spectrum window.'
           PRINT *,'The dB value has been set to -100
     1 0000000. I recommend that'
           PRINT *,'you go with the clipping preset on
```

```
     1 the next question.'
          SUMDB(IDB) = -1000000000.
        ELSE
          SUMDB(IDB) = 20.*ALOG10(ARG)
        ENDIF
400   CONTINUE
C-----------------------------------------------

C Find mean of attenuated zone;
C   First, get chnnel pts. for given freq band;
      CALL CNVRT(BEGCHNL,ENDCHNL,F1,F2,DELTAF2)
      XSUM = 0.
      DO 402 IMEAN=BEGCHNL,ENDCHNL
402      XSUM = XSUM + SUMDB(IMEAN)
      XNUMBR = ENDCHNL - BEGCHNL + 1
      XMEANDB = XSUM / XNUMBR
      WRITE(4,406) XMEANDB
406   FORMAT(1X,'XMEANDB = ',G20.10)
      WRITE(4,*) ' '

C Find Half Power Width:

      CALL GETHFPW(SUMDB,HFPW,DELTAF2)

      WRITE(4,873),HFPW
873    FORMAT(1X,'Half Power Freq = ',F6.2,1X,'Hz')
      WRITE(4,*) ' '

C------------Plot file details------------------
      PRINT *, 'Clipping? (N or Return):'
      READ(*,'(A1)'),QUESTN
      IF (QUESTN .EQ. 'N' .OR. QUESTN .EQ.'n') THEN
C       Output data as is.
        DO 404 J=1,NYQST2
        FREQ = (J-1)*DELTAF2
           WRITE(5,*) FREQ,SUMDB(J)
404     CONTINUE
      ELSE
C       Output data with clipping
        CLIP = -160.0
      PRINT *, 'Enter Y (change clipping value)
     lor Return (for -160 preset):'
      READ(*,'(A1)'),QUESTN
      IF (QUESTN .EQ. 'Y' .OR. QUESTN .EQ.'y') THEN
         PRINT *,' Enter new clipping value:'
         READ(*,*),CLIP
      ENDIF
        DO 401 J=1,NYQST2
           FREQ = (J-1)*DELTAF2
           IF(SUMDB(J).GE.CLIP) THEN
             WRITE(5,*) FREQ,SUMDB(J)
           ELSE
             WRITE(5,*) FREQ,CLIP
```

```
          ENDIF
401     CONTINUE

      ENDIF
C-----------------------------------------------


C Put some info in the diag. file;
      WRITE(4,*),'The first few values'
      WRITE (4,*),'of the  plot file:'

      DO 403 J2=1,10
         XJ2 = J2-1
         WRITE(4,*)XJ2,SUMDB(J2)
403   CONTINUE

C END OF MAIN PROGRAM
1050  CONTINUE
      END


C=============================================
C BEGIN  SUBROUTINES

      SUBROUTINE GETHFPW(X,F2,DELTAF)
      DIMENSION X(1)

C Find dB pt. just past -3dB.

      DO 10 I=1,10000
        IF(X(I) .LE. -3.) THEN
          IF(X(I) .EQ. -3.) THEN
            I2 = I
            CALL CNVRT2(I,F2,DELTAF)
            GO TO 20
          ELSE
            I1 = I-1
            I3 = I
            CALL CNVRT2(I1,F1,DELTAF)
            CALL CNVRT2(I3,F3,DELTAF)
            CALL INTERP(F1,F2,F3,X(I1),X(I3))
            GO TO 20
          ENDIF
        ENDIF
 10    CONTINUE
 20    CONTINUE

       RETURN
       END

       SUBROUTINE CNVRT2(I,F,DELTAF)
```

```
      F = (I-1)*DELTAF

      RETURN
      END

      SUBROUTINE INTERP(F1,F2,F3,DB1,DB3)

      DB2 = -3.

C Convert dB to ratio;

      R1 =( 10.**(DB1/20.))
      R2 =( 10.**(DB2/20.))
      R3 =( 10.**(DB3/20.))
C Linearly interpolate;

      F2 = F1 + (F3-F1)*(R2-R1)/(R3-R1)

      RETURN
      END

      SUBROUTINE CNVRT(BEGCHNL,ENDCHNL,F1,F2,DELTAF)
      INTEGER BEGCHNL,ENDCHNL

      BEGCHNL = (F1/DELTAF) + 1.5
      ENDCHNL = (F2/DELTAF) + 1.5

      IWIDTH = ENDCHNL - BEGCHNL + 1
      IF (IWIDTH .GT. 50) THEN
       PRINT *, 'Channel width .GT. 50'
       STOP
      ENDIF
      RETURN
      END

      SUBROUTINE FFT(CW,LX)
      COMPLEX CW(1)
      CALL FFTWRAP(CW,LX)
      CALL FORK(LX,CW,1.0)
      RETURN
      END

      SUBROUTINE FFTWRAP(CW,LX)
      COMPLEX CW(1)
         NYQUIST=LX/2 + 1
         IEND=LX-NYQUIST
         DO 10 I=1,IEND
 10          CW(NYQUIST+I) = CW(NYQUIST-I)
      RETURN
      END

      FUNCTION WINDOW(ARG,ICNT)
      PI = 3.14159
```

```
      IF(ICNT.EQ.1) THEN
        GO TO 1
      ELSEIF(ICNT.EQ.2) THEN
        GO TO 2
       ELSEIF(ICNT.EQ.3) THEN
        GO TO 3
        ELSEIF(ICNT.EQ.4) THEN
         GO TO 4
         ELSEIF(ICNT.EQ.5) THEN
          GO TO 5
          ELSEIF(ICNT.EQ.6) THEN
           GO TO 6
           ELSEIF(ICNT.EQ.7) THEN
            GO TO 7
            ELSEIF(ICNT.EQ.8) THEN
             GO TO 8
             ELSEIF(ICNT.EQ.9) THEN
              GO TO 9
              ELSEIF(ICNT.EQ.10) THEN
               GO TO 10
               ELSEIF(ICNT.EQ.11) THEN
                GO TO 11
                 ELSE
                  GO TO 100
      ENDIF

1     CONTINUE
C Rectangular
      WINDOW = 1.
      GO TO 100

2     CONTINUE
C Parzen-2
      WINDOW = 1. - 4. * (ARG**2)
      GO TO 100

3     CONTINUE
C Cosine-Tip
      WINDOW = COS(ARG*PI)
      GO TO 100

4     CONTINUE
C Bartlett-like, with sign change
      WINDOW = 1. + 2.*ABS(ARG)
      GO TO 100

5     CONTINUE
C Hann
      WINDOW = .5 + .5*COS(2.*ARG*PI)
      GO TO 100

6     CONTINUE
C Hamming
```

```
      WINDOW = .54 + .46*COS(2.*ARG*PI)
      GO TO 100

7     CONTINUE
C Papoulis
      WINDOW = 1. - 2.*ABS(ARG)*COS(2.*ARG*PI)
     1            + (1./PI)*ABS(SIN(2.*PI*ARG))
      GO TO 100

8     CONTINUE
C Blackman
      WINDOW = .42 +.5*COS(2.*ARG*PI)+.08*COS(4.*ARG*PI)
      GO TO 100

9     CONTINUE
C Triangle
      WINDOW = 1. - 2.*ABS(ARG)
      GO TO 100

10    CONTINUE
      IF(ARG .EQ. 0.) THEN
        WINDOW = 1.
      ELSE
        WINDOW = SIN(2.*PI*ARG)/(2.*PI*ARG)
      ENDIF
      GO TO 100

11    CONTINUE
C Gaussian
      ALPHA = 2.5
      ARG2 = -.5*((2.*ALPHA*ARG)**2)
      WINDOW = EXP(ARG2)
      GO TO 100

100   CONTINUE

      RETURN
      END

      SUBROUTINE FORK(LX,CX,SIGNI)
C From Claerbout's FGDP (first ed.), pg 12. Typed in by MKB.

      COMPLEX CX(LX),CARG,CEXP,CW,CTEMP
      J=1
      SC=SQRT(1./LX)
      DO 30 I=1,LX
      IF(I.GT.J) GO TO 10
      CTEMP=CX(J)*SC
      CX(J)=CX(I)*SC
      CX(I)=CTEMP
10    M=LX/2
20    IF(J.LE.M) GO TO 30
      J=J-M
```

```
      M=M/2
      IF(M.GE.1) GO TO 20
30    J=J+M
      L=1
40    ISTEP=2*L
      DO 50 M=1,L
      CARG=(0.,1.)*(3.14159265*SIGNI*(M-1))/L
      CW=CEXP(CARG)
      DO 50 I=M,LX,ISTEP
      CTEMP=CW*CX(I+L)
      CX(I+L)=CX(I)-CTEMP
50    CX(I)=CX(I)+CTEMP
      L=ISTEP
      IF(L.LT.LX) GO TO 40
      RETURN
      END

      SUBROUTINE CCLEAR(X,LEN)
      COMPLEX X(1)
      DO 10 I=1,LEN
10        X(I) = (0.,0.)
      RETURN
      END

      SUBROUTINE CLEAR(X,LEN)
      DIMENSION X(1)
      DO 10 I=1,LEN
10        X(I) = 0.
      RETURN
      END

       integer function stringlen(string,length)
      integer length
      character*(*) string
      logical done
      integer i
      i=length
      done=.false.
      do while (.not. done)
         done=(string(i:i).ne.' ').or.(i.le.1)
         i=i-1
         end do

c     check for "empty" string; i.e., having only blanks
       if(i.eq.1 .and. string(1:1).eq.' ') i=0

       stringlen=i
       if(i.gt.0) stringlen=i+1
       return
       end

      SUBROUTINE WNDONLY(CW,SUMDB,ILAG1,SUMARRY)
```

```
      COMPLEX CW(1)
      DIMENSION WIND(1024),SUMDB(1),SUMARRY(1)
      CHARACTER OPTION*4
      CHARACTER QUESTN*1
      PRINT *,'Enter window no.:'
      READ(*,*),IWINDNO
C     Calculate window.
      PRINT *, 'No. smples for lag wndw (desgn)'
          READ(*,*),ilag1
      CALL CALC(WIND,IWINDNO,ILAG1)
C Prepare output.
      PRINT *,'Enter FUNC or FREQ:'
      READ(*,10),OPTION
10    FORMAT(A4)
      IF(OPTION .EQ. 'FUNC') THEN
C       Output function domain window values for plotting.
        CALL FUNCOUT(WIND,ILAG1)
      ELSE
C       Compute FFT and output freq. domain window values
C       for plotting.
      PRINT *, 'No. smples for spec wndw (desgn)'
          READ(*,*),ifftnm1
        CALL FREQOUT(WIND,CW,SUMDB,SUMARRY
     1 ,ILAG1,IFFTNM1)
      ENDIF

      RETURN
      END

      SUBROUTINE CALC(WIND,IWINDNO,ILAG1)
      DIMENSION WIND(1)
      xlag1 = ilag1
      ihalf1 = ilag1/2 + 1
C         Compute a window.
          DO 10 I=1,ihalf1
             XI = I
             ARG = (XI-1.)/xlag1
10        WIND(I)=WINDOW(ARG,IWINDNO)

      RETURN
      END

      SUBROUTINE FUNCOUT(WIND,ILAG)
      DIMENSION WIND(1)
        IHALF2 = ILAG/2 + 1
        THALF = 0.5
        XHALF2 = IHALF2
        DELTAT = THALF/(XHALF2 - 1.)
        SUM = 0.0
        T = 0.0
        WRITE(5,*) T,WIND(1)
        DO 405 J5=2,ihalf2
           SUM = SUM + DELTAT
```

```
          T = SUM
          WRITE(5,*) T,WIND(J5)
405     CONTINUE

      RETURN
      END

      SUBROUTINE FREQOUT(WIND,CW,SUMDB,SUMARRY
     1                   ,ILAG2,IFFTNM2)
      DIMENSION WIND(1),SUMARRY(1),SUMDB(1)
      COMPLEX CW(1)
      CHARACTER QUESTN*1
      NYQST2 = IFFTNM2/2 + 1
      XLAG2 = ILAG2
      XFFTNM2 = IFFTNM2
      DELTAF2 = XLAG2/XFFTNM2
      IHALF2 = ILAG2/2 + 1

      CALL CCLEAR(CW,ifftnm2)
      DO 23 IMOV=1,ihalf2
23       CW(IMOV)=WIND(IMOV)
      CALL FFT(CW,Ifftnm2)
      CALL CLEAR(SUMARRY,ifftnm2)
      DO 24 IMOV=1,NYQST2
24       SUMARRY(IMOV)=CW(IMOV)


C Output Stage:

C-----------Basic set up--------------------------------
      XMAX = SUMARRY(1)
      DO 400 IDB=1,NYQST2
         ARG = ABS((SUMARRY(IDB)/XMAX))
         IF (ARG.EQ.0.0) THEN
           PRINT *,'A zero value was encountered in
     1 the hybrid spectrum window.'
           PRINT *,'The dB value has been set to -100
     1 0000000. I recommend that'
           PRINT *,'you go with the clipping preset on
     1 the next question.'
           SUMDB(IDB) = -1000000000.
         ELSE
           SUMDB(IDB) = 20.*ALOG10(ARG)
         ENDIF
400   CONTINUE
C-----------------------------------------------


C Find Half Power Width:

      CALL GETHFPW(SUMDB,HFPW,DELTAF2)

      WRITE(*,873),HFPW
```

```
873    FORMAT(1X,'Half Power Freq = ',F6.2,1X,'Hz')

C------------Plot file details------------------
      PRINT *, 'Clipping? (N or Return):'
      READ(*,'(A1)'),QUESTN
      IF (QUESTN .EQ. 'N' .OR. QUESTN .EQ.'n') THEN
C       Output data as is.
        DO 404 J=1,NYQST2
        FREQ = (J-1)*DELTAF2
           WRITE(5,*) FREQ,SUMDB(J)
404     CONTINUE
      ELSE
C       Output data with clipping
        CLIP = -160.0
      PRINT *, 'Enter Y (change clipping value)
     1or Return (for -160 preset):'
      READ(*,'(A1)'),QUESTN
      IF (QUESTN .EQ. 'Y' .OR. QUESTN .EQ.'y') THEN
         PRINT *,' Enter new clipping value:'
         READ(*,*),CLIP
      ENDIF
        DO 401 J=1,NYQST2
           FREQ = (J-1)*DELTAF2
           IF(SUMDB(J).GE.CLIP) THEN
             WRITE(5,*) FREQ,SUMDB(J)
           ELSE
             WRITE(5,*) FREQ,CLIP
           ENDIF
401     CONTINUE
      ENDIF

      RETURN
      END

      SUBROUTINE GETLIST(IARRAY,NUM)
      DIMENSION IARRAY(1)
      INTEGER NUM

      PRINT *,' '
      PRINT *,'Welcome to PRGRM1, an alternative window
     1 design program'

      PRINT *,' '
      PRINT *,'Later below, you will be asked for a list
     1 of numbers corresponding to'
      PRINT *,'the windows you want to use. The window options
     1 are given here with'
      PRINT *,'their respective identifying numbers.'
      PRINT *,' '
      PRINT *, 'Window options:'
      PRINT *, 'Window No. 1 -  Rectangular'
      PRINT *, 'Window No. 2 -  Parzen-2'
```

```
      PRINT *, 'Window No. 3 -  Cosine-Tip'
      PRINT *, 'Window No. 4 -  Bartlett-like, with sign change'
      PRINT *, 'Window No. 5 -  Hann'
      PRINT *, 'Window No. 6 -  Hamming'
      PRINT *, 'Window No. 7 -  Papoulis'
      PRINT *, 'Window No. 8 -  Blackman'
      PRINT *, 'Window No. 9 -  Bartlett (Triangle)'
      PRINT *, 'Window No. 10 - Sinc like'
      PRINT *, 'Window No. 11 - Gaussian'
      PRINT *,' '
      PRINT *, 'Enter the total number of windows you are'
      PRINT *, 'going to use (between 2 and 11 inclusive):'
      ACCEPT *, NUM

C Do some editing.
        IF (NUM .GE. 2 .AND. NUM .LE. 11) THEN
        ELSE
         PRINT *,'The number of windows you can use must'
         PRINT *, 'be between 2 and 11 (inclusive).'
         STOP
        ENDIF

      PRINT *,' '
      PRINT *, 'Enter window no.s here (list of integers, each on '
      PRINT *, 'a separate line):'

      READ(*,100), (IARRAY(I),I=1,NUM)
100   FORMAT(I2)

C Do some editing.
      DO 30 I=1,NUM
        IF (IARRAY(I) .GE. 1 .AND. IARRAY(I) .LE. 11) THEN
        ELSE
         PRINT *,'One or more of the window no.s you gave is outside'
         PRINT *, 'the acceptable range (1-11).'
         STOP
        ENDIF
30    CONTINUE

      END
```

VITA

The author, Michael Kent Broadhead, was born [redacted] His undergraduate work was done at the University of Southern Mississippi and he was awarded a Bachelor of Science degree in Mathematics in May, 1979. He was employed by Texaco, Inc. from 1979 through 1989, and is currently a candidate for a Master of Science degree in Physics from the University of New Orleans.