AN INTERACTIVE COMPUTER ANALYSIS OF THE AORTIC EJECTION CLICK AND FIRST HEART SOUND

by

Samuel Joseph Showalter

Thesis submitted to the Graduate Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE
in
Electrical Engineering

APPROVED:

E. A. Manus, Chairman

A. A. Sarkady

P. H. Wiley

August, 1978

Blacksburg, Virginia

ACKNOWLEDGMENTS

I would like to express my appreciation to the following members of my graduate committee for their instruction, advice, and encouragement in helping me complete this thesis: , chairman, , and . I am grateful to for allowing me to use his recordings of phonocardiogram data.

I also wish to express apprciation to the National Science Foundation for its financial support (proposal grant no. ENG76-09846) of this work.

Many thanks go to for her encouragement and efficient typing.

Finally, I would like to thank each member of my family as well as and , , and for their moral and spiritual support.

INTRODUCTION

The rapid technological advancement of small scale computers and their growing acceptance in many areas has opened new possibilities in medical diagnosis. These possibilities are furthered by modern signal processing techniques and data aquisition hardware.

The intent of this thesis is to provide a mathematical description of the aortic ejection click and its characteristics relative to the first heart sound. The aortic ejection click is considered to be a reliable indicator of the presence of the heart disease valvar aortic stenosis. However, it is sometimes difficult to aurally distinguish the click from a loud first heart sound (a normal occurrence). It is hoped that sufficient research will lead to a reliable way of detecting clinically the presence of aortic stenosis without the need for catheterization (an investigative operation to determine the presence and severity of aortic stenosis).

A FORTRAN computer program was developed to perform the required data analysis. It was a flexible interactive program capable of computing averages, aligned averages, power spectra of selected segments, and envelograms. It also could be used to generate deterministic signals, store and retrieve intermediate results, and plot data at the users terminal.

The chapters which follow discuss the background physiology and pathology, the data base, signal processing techniques, the analysis program, and the results and conclusions.

CHAPTER I

PHYSIOLOGY OF THE NORMAL AND ABNORMAL HEART

1.1 FUNCTION AND OPERATION OF THE NORMAL HEART

The function of the heart within the physiological system is to circulate the blood. The blood then provides the body with oxygen, removes carbon dioxide and other wastes, and is a transport mechanism for many chemicals. Oxygen is particularily important since body cells will start to die within a few minutes if it is not provided. For this reason the heart must operate without interruption for life. It must also be able to adjust the amount of blood pumped to match the needs of the body.

Schematically, the heart is composed of four chambers which function within the circulatory system as shown in Fig. 1-1. Within the heart there is no flow of blood between the right and left sides. For this reason the two sides are referred to as the right heart and the left heart. The right heart, composed of the chambers right atrium and right ventricle, receives oxygen poor blood from the body and pumps it to the lungs where it is oxygenated. Similarly, the left heart takes oxygen rich blood from the lungs and pumps it to the body. Flow operated one-way valves are located at the outlet of each chamber.

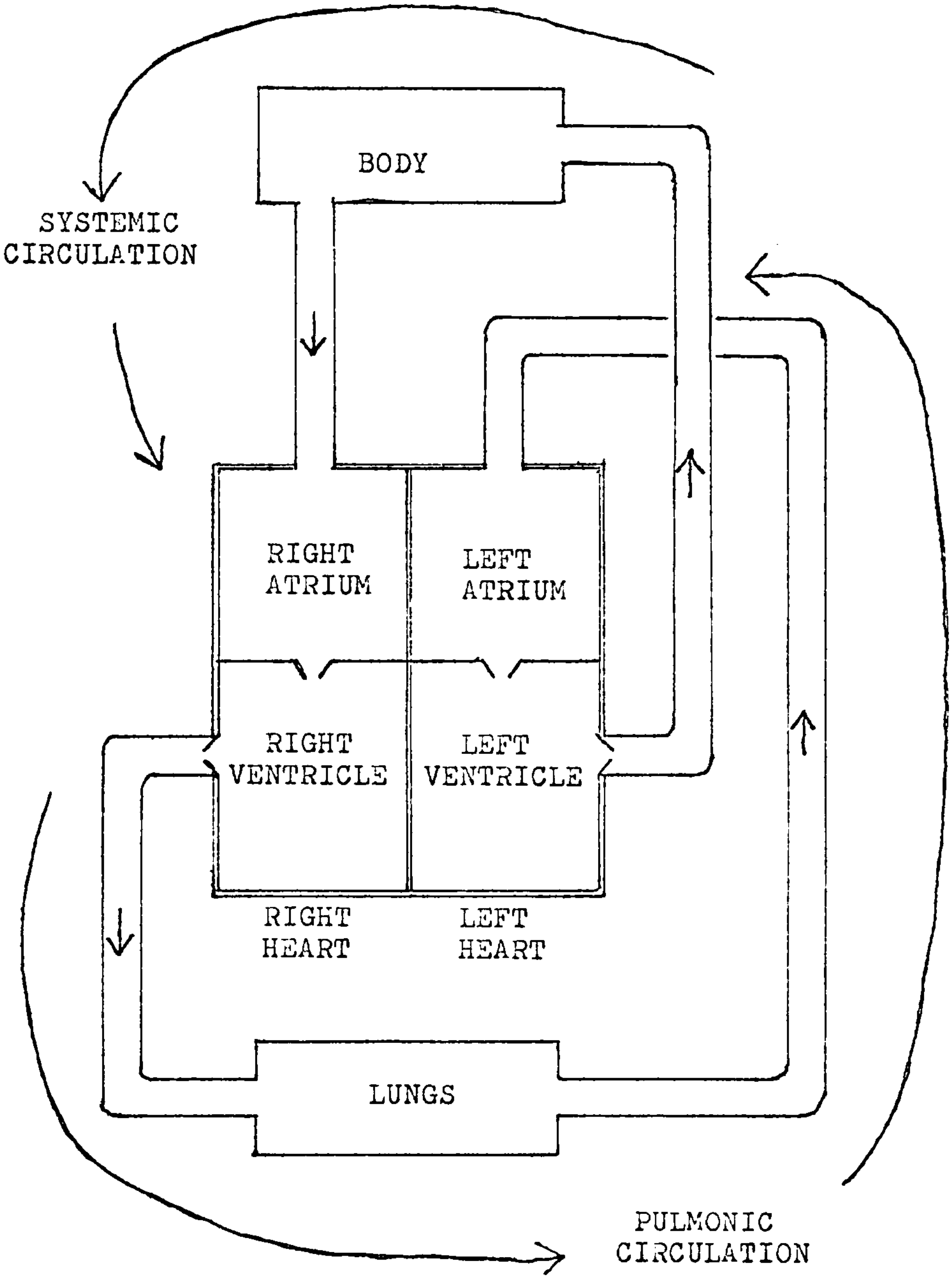
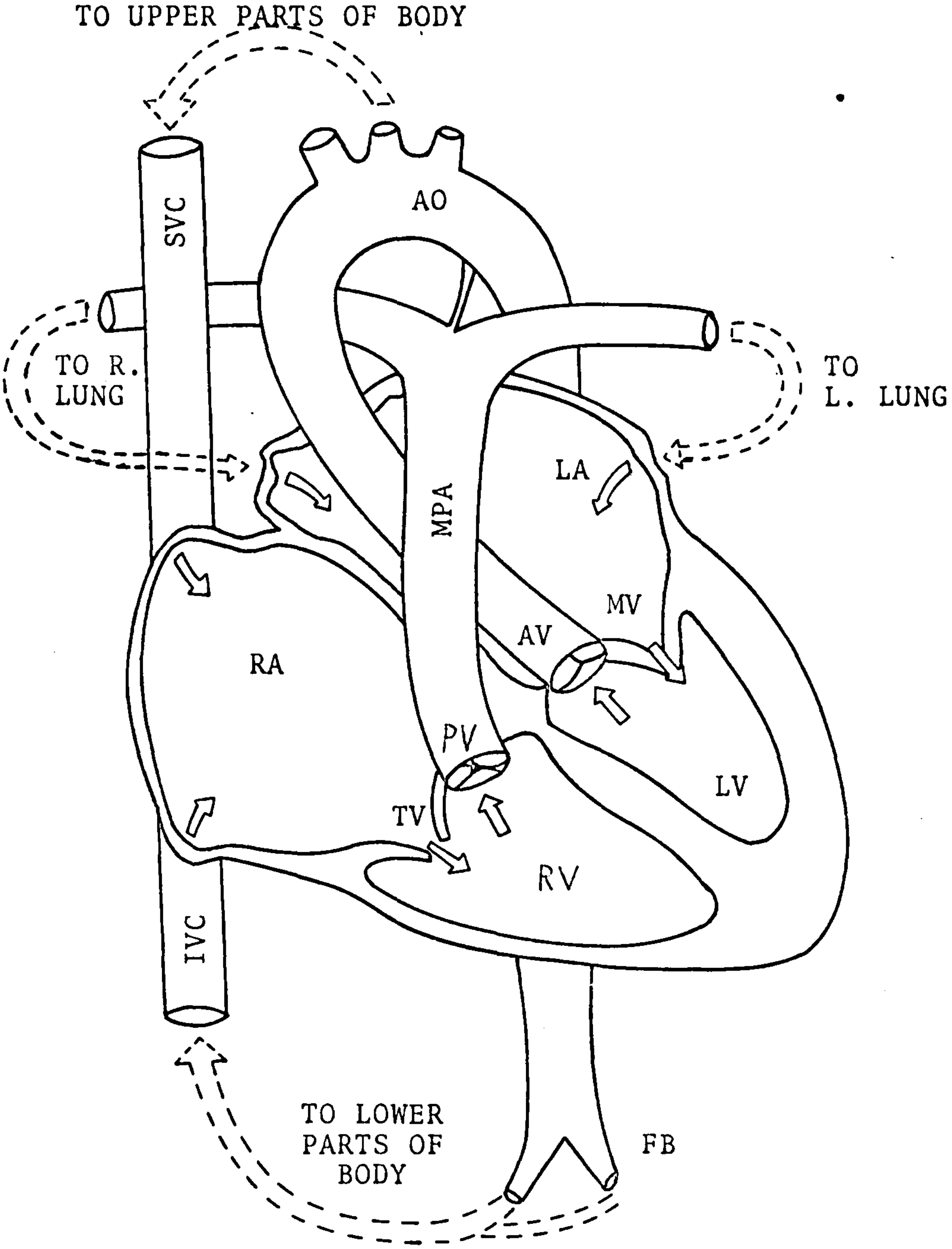


Fig. 1-1 The overall circulation system

Anatomically, the heart is built around a fibrous skeleton which surrounds the four heart valves [1]. The atria and ventricles, as well as the valves, are attached to this structure. Figure 1-2 shows an anterior view of the basic heart anatomy and the surrounding great vessels. The walls of the heart are made of a special type of muscle called myocardium. This muscle is relatively thin in the walls of the atria, being much thicker in the ventricles, especially the left. The atria and ventricles are separated, left from right, by the interatrial septum and the interventricular septum respectively.

Blood enters the right atrium from the upper and lower parts of the body through the superior vena cava and inferior vena cava respectively. It then flows through the tricuspid valve into the right ventricle from which it is pumped out through the pulmonary artery to the lungs. Blood returning from the lungs enters the left atrium and flows through the mitral valve to the left ventricle. From there it is pumped through the aortic valve and the aorta to the body.

The aorta functions as a reservor and distribution junction. It has a flexible wall which allows it to expand when blood is pumped rapidly into it from the left ventricle and then contract as blood continues to flow out into the arteries. This action prevents extremely high pressures



AO-aorta, AV-aortic valve, IVC-inferior vena cava
LA-left atrim, LV-left ventricle, MPA, main pulmonary artery
MV-mitral valve, PV-pulmonary valve, RV-right atrium
RV-right ventricle, SVC-superior vena cava
TR-tricuspid valve, FB-femoral bifurcation

Fig. 1-2 Normal heart of a child

from developing during ventricular contraction and helps prevent the arterial pressure from dropping to zero between contractions [2,3].

Normal heart action for one cardiac cycle is shown graphically in Fig. 1-3 from three viewpoints: (1) blood pressures associated with the left heart; (2) the electrocardiogram (ECG) from lead 2; and (3) the phonocardiogram (PCG). The terms systole and diastole refer in general to the periods of time when the heart is pumping and filling respectively [4]. Specific definitions have varied but here left heart systole was taken to be the time between mitral value closure and aortic valve closure with left heart diastole being the remaining time during the cardiac cycle.

The ECG is a recording of body surface potential differences taken between several points, typically the right arm, left arm, and left leg. These voltages result from the electrical activity of the heart which causes muscular contraction for each heart beat. The P wave is associated with atrial systole while the QRS wave complex is associated with ventricular systole. The T wave occurs near the end of electrical activity for a given cycle [4,5]. The ECG is particularly useful as a reference for timing cardiac events. The Q point on the ECG was used here to establish a zero time reference point for each cardiocycle.

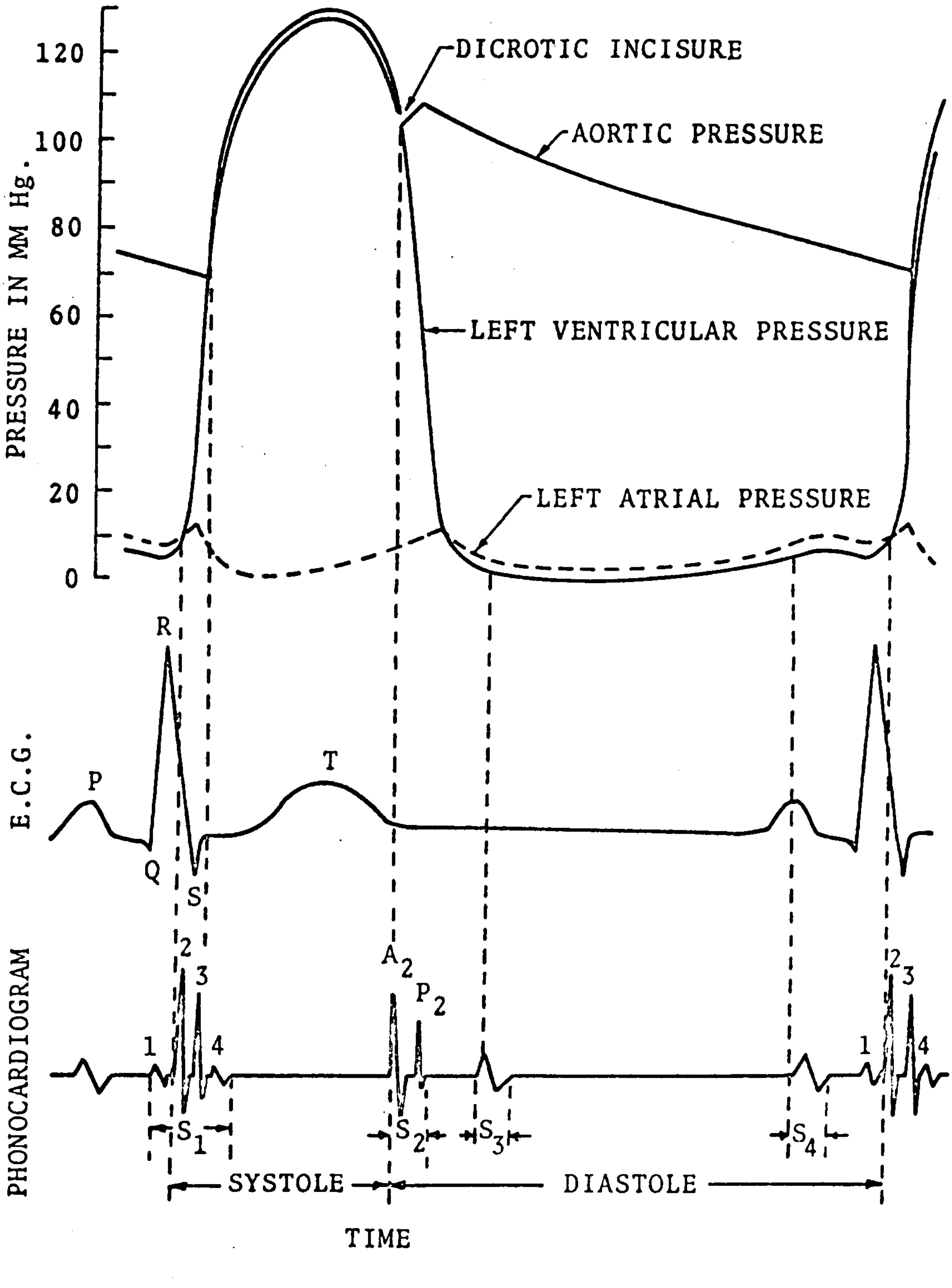


Fig. 1-3 A typical normal cardiac cycle

The three pressure wave forms of Fig. 1-3 show the hemodynamics of a normal left heart. This action begins with atrial contraction (P wave on the ECG) during which time final filling of the ventricle occurs. When ventricular systole begins, the rising ventricular pressure closes the mitral valve as soon as it rises above atrial pressure. The ventricle then undergoes a short period of isovolumetric contraction during which time the pressure rises rapidly. As soon as the ventricular pressure exceeds the aortic pressure, the aortic valve opens and blood is ejected at an initially high rate into the aorta. During this ejection phase the aortic pressure remains only slightly lower than the pressure in the ventricle. Part way through the ejection phase the ventricle begins to relax which allows the pressure to drop and the flow rate to decrease. When ventricular pressure drops below aortic pressure, the aortic valve is closed by a slight retrograde blood flow. This action causes the dicrotic incisure, or dicrotic notch, in the aortic pressure waveform and ends systole. The ventricle then undergoes a short period of isovolumetric pressure reduction until ventricular pressure drops below atrial pressure. At this point the mitral valve opens and the ventricle is rapidly filled with blood from the atrium. The heart is then essentially at rest until another P wave on the ECG starts another cycle [2,3,5].

The phonocardiogram signal is shown here to indicate its synchronization with cardiac events and is described in the following chapter.

1.2 THE NORMAL AORTIC VALVE

The aortic valve is composed of three symetrical valve leaflets or cusps which are attached to the fibrous heart skeleton. A normal aortic valve is shown in Fig. 1-4 as it would appear if opened laterally following a verticle incisure down the front of the aorta and left ventricle. The cusps are cup shaped with the concavity facing the aorta. Their edges are somewhat thickened (nodules) and have small flaps (lunules) which insure a complete seal when closed. Behind the valve cusps are the sinuses of valsalva. These are outward pouches in the walls of the aorta which allow blood to circulate behind the open cusps. The purpose of this is to prevent the cusps from sealing off the coronary ostia (openings into the coronary arteries) and to help float the cusps back to a closed position [6,7,8,9].

The valve is operated entirely by blood flow, there being no muscles or tendons attached to the cusps. When open, the valve orifice is approximately triangular and of sufficient size to allow laminar blood flow with a negligible pressure drop across the valve. When blood flow stops, the cusps return to a partially closed position from which they are completely closed by a slight reverse flow.

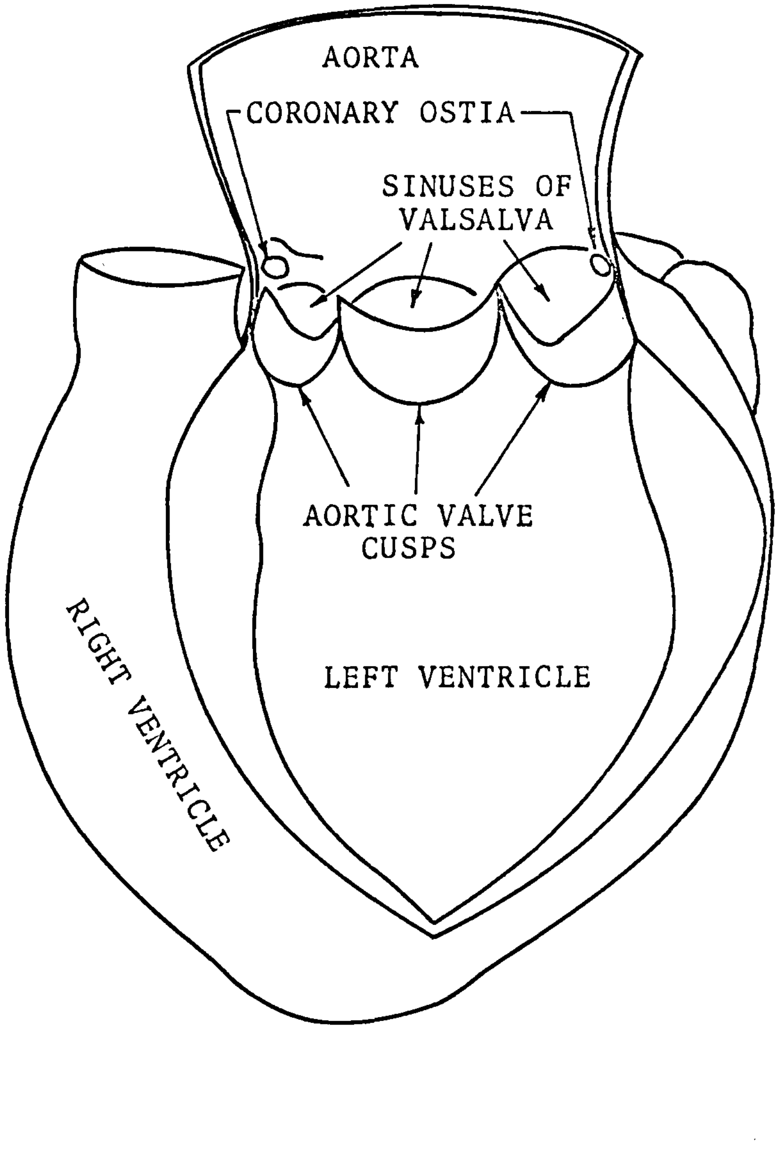
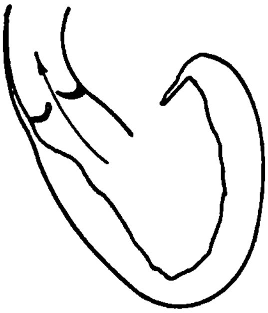


Fig. 1-4 The normal aortic valve

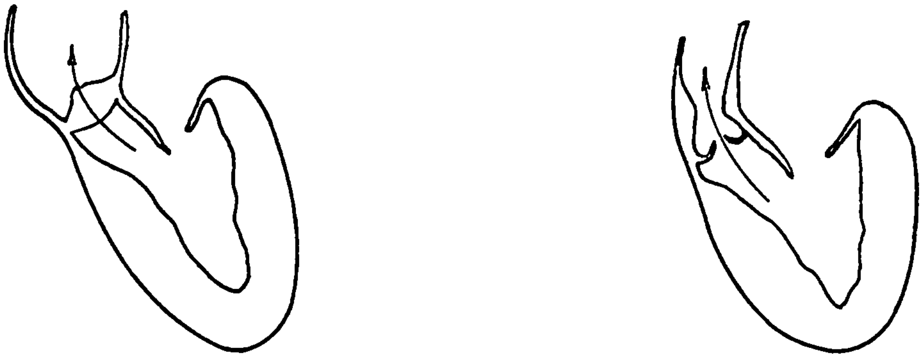
1.3 AORTIC STENOSIS

Anatomically, aortic stenosis is a constriction in the blood flow tract between the left ventricle and the aorta. This constriction can occur below the aortic valve, within the valve, or above it, giving rise to subvalvar, valvar, or supravalvar aortic stenosis respectively. Subaortic stenosis is further divided into disrete and idiopathic cases. These four anomolies along with a normal heart are shown in Fig. 1-5. Supravalvar stenosis and subvalvar idiopathic stenosis results from a deformed blood flow tract while subvalvar discrete stenosis is caused by a fibromuscular band just below the aortic valve [4,5,10].

Valvar aortic stenosis occurs approximatelly ten times more often than nonvalvar aortic stenosis andcan be of three types as shown in Fig. 1-6. If two of the valve cusps are fused together, the anomally is termed bicuspid valvar aortic stenosis. When all three cusps are partially fused around the perimiter of the valve, it is called tricuspid valvar aortic stenosis. In both of these cases valve action is maintained but the open valve area is reduced. In the case of a severe congenital defect, all valve action is lost and the valve becomes cnly an obstruction with equal resistance to both directions of flow.



NORMAL



VALVAR AORTIC
STENOSIS

SUPRAVALVAR
AORTIC STENOSIS

IDIOPATHIC HYPERTROPHIC
SUBAORTIC STENOSIS

DISCRETE SUBVALVAR
AORTIC STENOSIS

Fig. 1-5 Normal and aortic stenosed hearts

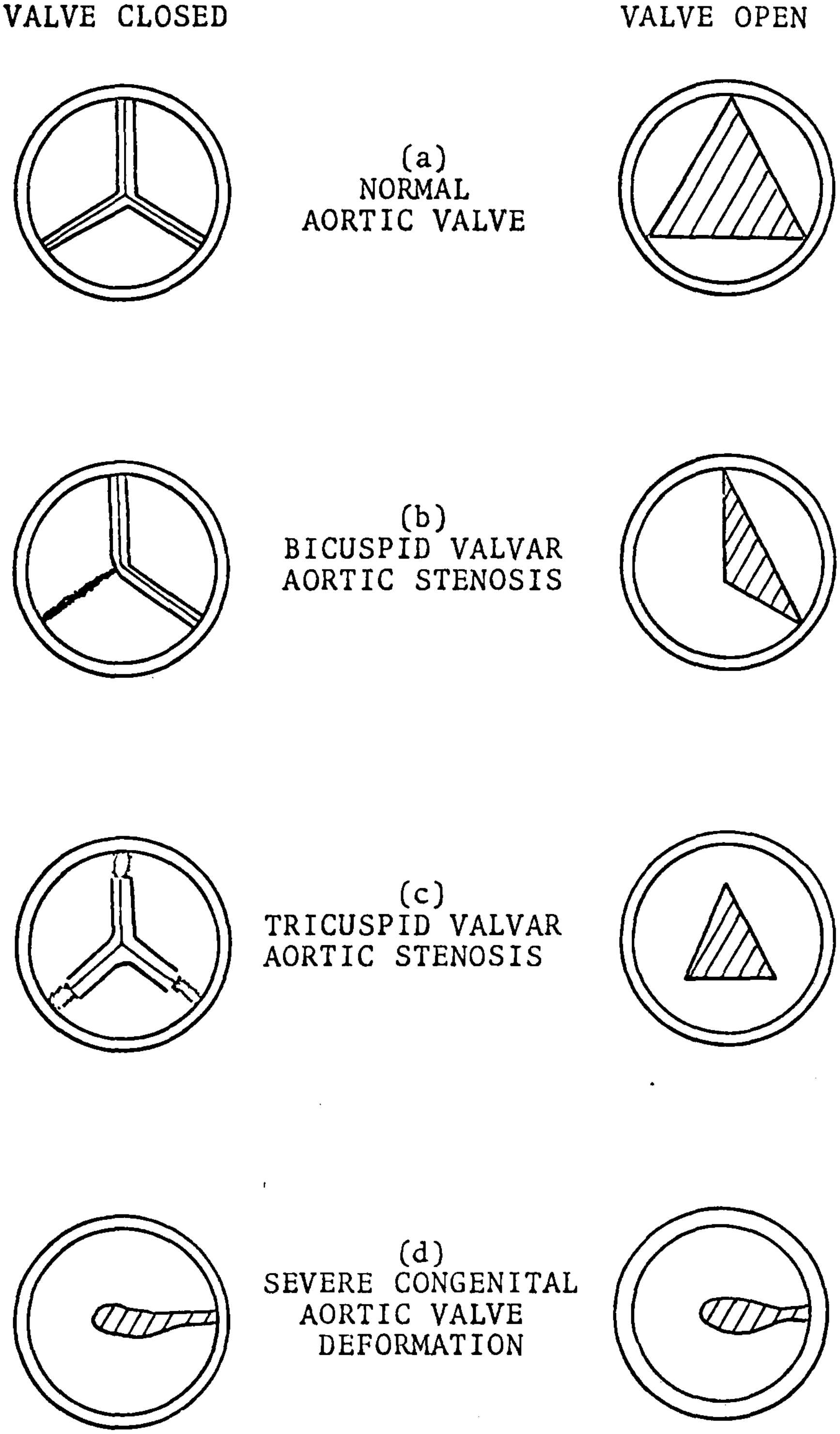


Fig. 1-6 Cross-sectional views of normal and stenosed valves

The normal area of the open valve orfice is 2.5 to 3.5 cm^2. Valvar aortic stenosis reduces this area and when it reaches 0.5 to 1.0 cm^2 the heart can no longer compensate and clinical symptoms develop [4,10].

A physiological definition of aortic stenosis is: a condition which causes a significant systolic pressure drop between the left ventricle and the aorta. In severe cases the peak pressure drop can exceed 100 mmHg. It is accompanied by a slower rise in the aortic systolic pressure wave and a reduction in cardiac output during exercise.

The small valve area in valvar aortic stenosis causes the formation of a jet of blood in the aorta above the valve. This has been shown in angiocardiographic studies [4], occurs throughout systole, and is accompanied by turbulent blood flow in the rising aorta.

The importance of early detection of aortic stenosis is indicated by the permenent effects the desease has on the ventricular heart muscle. The increased resistance of a stanotic valve forces the heart to work harder than normal. This increased work load causes the ventricular muscle mass to increase which reduces ventricular volume. Reduced volume decreases the efficiency of the heart, thus requiring it to work even harder which further increases muscle mass. This action compounds itself and will eventually reach a point where eventual death of the patient will occur even though the valve itself is replaced.

CHAPTER II

CARDIOVASCULAR SOUNDS

2.1 GENERATION OF HEART SOUNDS

Heart sounds result from vibrations in the heart and the surrounding great vessels [4,12,13,14]. These vibrations are caused by acceleration and deceleration of the blood, eddie currents in the blood, or by turbulant blood flow. In general, normal heart sounds are caused by acceleration and deceleration of the blood, innocent murmurs result from eddies, and pathological murmurs are produced by tubulent blood flow [15,16,17].

In the past, many different parts of the heart have been named as the cause of heart sounds. But a more realistic approach to considering the cause of a particular heart sound is to see the cardiohemic system as a highly interactive one being stimulated by various hemodynamic events during the cardiac cycle [1,2]. Since the heart is completely filled with blood, it can be considered as a fluid filled elastic system. Therefore, any acceleration or deceleration of the blood at one point in the system will cause the entire system to oscillate. The amplitude of these ocsillations is largely dependent on the rate of acceleration or deceleration of the blood [4,18,19,20].

Their frequency is determined by the mass and stiffness of the system. As the ventricles contract, the mass should decrease while the stiffness increases, resulting in a higher frequency of oscillation. A heart sound generated by a particular event lasts only a few cycles, thus indicating high damping.

Vibrations from the heart are conducted to the surface of the chest at velocities which result in a wavelength less than the distance traveled [21]. Therefore, all material in the transmission path tends to vibrate together. Vibrations are strongest on the surface of the chest where the transmission path is shortest since attenuation is rapid, especially through soft tissues like the lungs or fatty layers. Once on the surface, the vibrations spread as transverse waves with a velocity which is proportional to the square root of the frequency. The velocity is about 15 m/s at 100 Hz [22].

2.2 AUSCULTATION

Auscultation, the procedure of listening for sounds within the body, has long been established as an effective clinical method of determining the state of the heart. It can reveal abnormalities which still would be undetectable by other clinical procedures. The four main auscultatory areas for the heart, aortic, pulmonary, tricuspid, and mitral, are shown in Fig. 2-1. The sounds generated by the

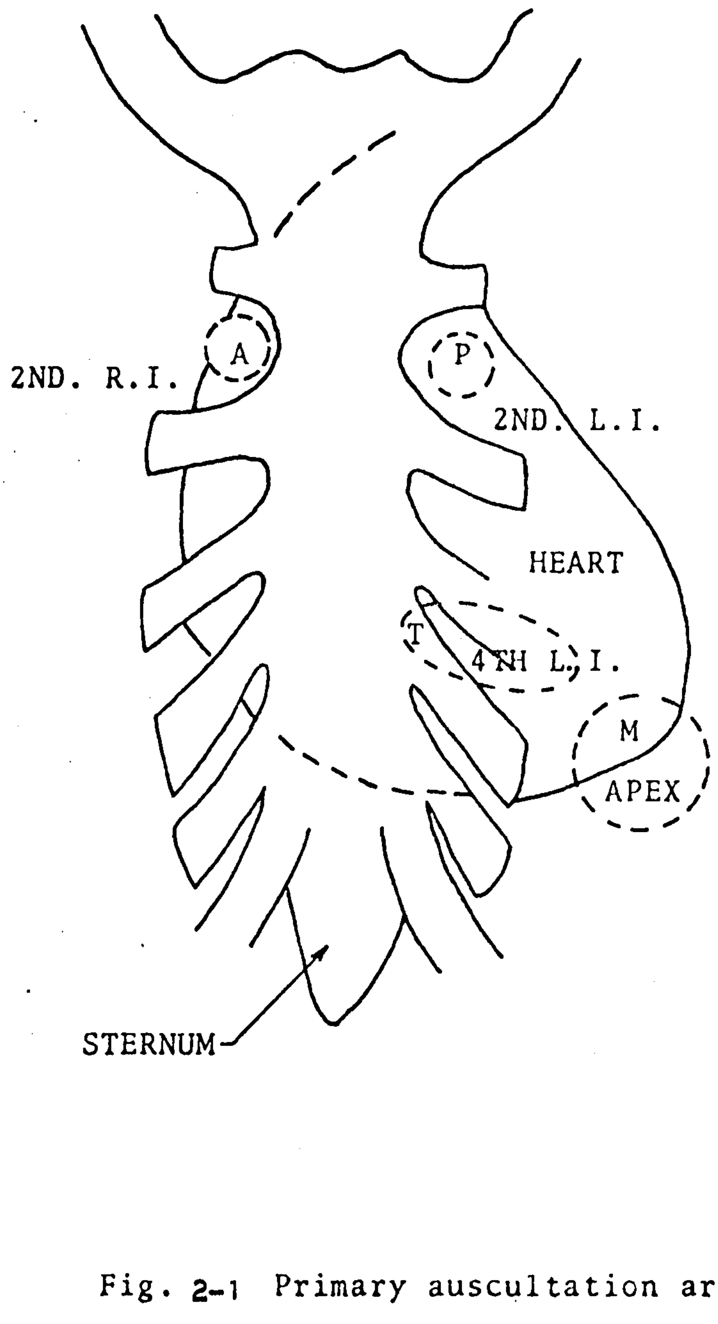


Fig. 2-1 Primary auscultation areas

action of a given valve are generally the loudest at the area of that name and it has been shown that this is the area where the sounds first reach the surface [22]. The aortic and pulmonary areas are located at the second right intercostal space (2nd R.I.S.) and second left intercostal space (2nd L.I.S.) respectively next to the sternum border. These are the areas where the aorta and main pulmonary artery are closest to the surface. The tricuspid area is located at the fourth left intercostal space (4th L.I.S.) and the mitral area is over the apex of the heart. These two areas are not located at the nearest points to their respective valves, but rather where the right and left ventricles are closest to the surface.

The range of frequencies observed on the human chest is from about 1 to 1000 Hz with most of the energy concentrated below 100 Hz [22]. Figure 2-2 shows the intensity of chest vibrations versus frequency along with the mean threshold of hearing. As can be seen, the range of spectroscopic auscultation is limited to about 40 to 700 Hz [5].

A time domain plot of heart sounds and murmurs is called a phonocardiogram. A typical cycle for a patient with valvar aortic stenosis is shown in Fig. 2-3. S1 and S2 are normal heart sounds occurring at the beginning and end of systole while the aortic ejection click and the ejection murmur are results of the disease.

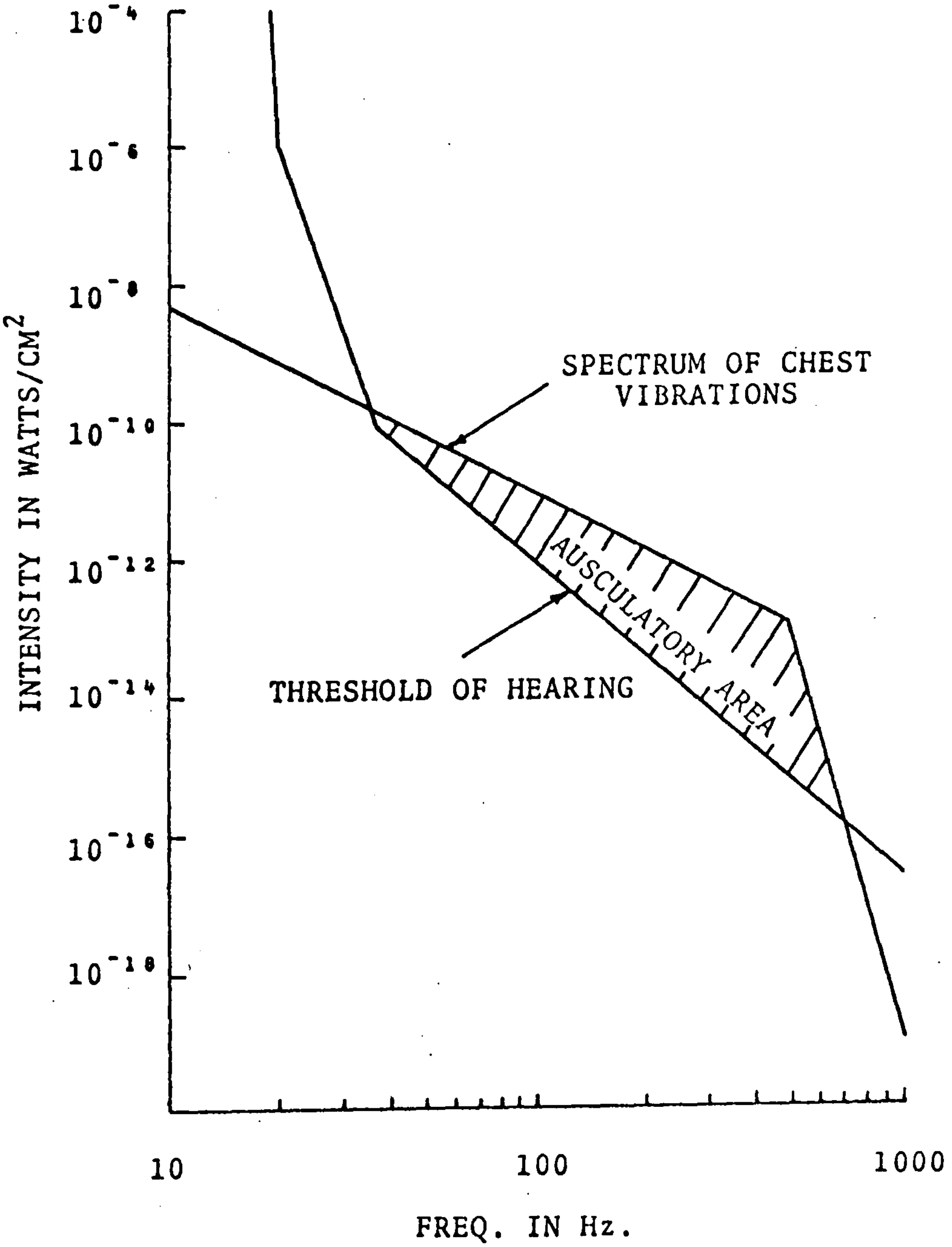


Fig. 2-2 Spectrum of chest vibrations and threshold of hearing.

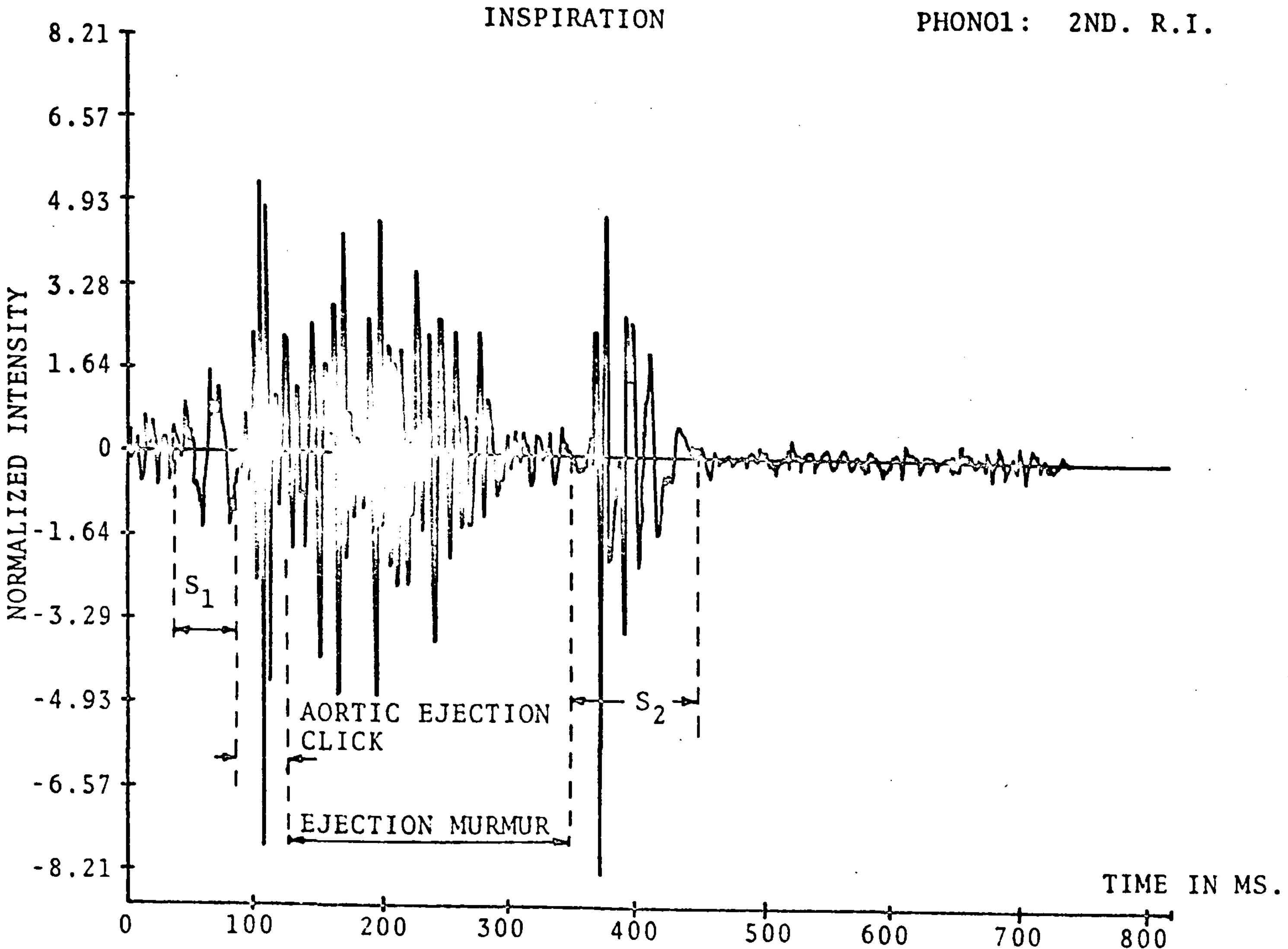


Fig. 2-3 A typical V.A.S. phonocardiogram cycle

2.3 THE FIRST HEART SOUND

The first heart sound is associated with the beginning of systole. The hemodynamic events causing it are, in chronological order: (1) the onset of left ventricular contraction; (2) the onset of right ventricular contraction; (3) the onset of right ventricular ejection; and (4) the onset of left ventricular ejection. Due to the rapid occurence of these events, oscillations caused by one may merge with oscillations caused by a following one.

Most researchers hold the opinion that the first heart sound is composed of four components [5]. The first component is thought to be caused by the initial contraction of the left ventricle before closure of the mitral valve. This component is of low amplitude and frequency. The larger amplitude, higher frequency (approximatelly 50 Hz) second component occurs with the closure of the mitral valve. This is the main component of the first heart sound and occurs approximately 20 ms after the first. The third component is also of high frequency and is most likely caused by the beginning of ejection into the great vessels. It follows mitral valve closure by an average of 30 ms. The fourth component is associated with the acceleration of blood into the great vessels and is of low amplitude and frequency. These components are often merged into one more continuous sound. The total length of the first heart sound is typically 100 to 120 ms [4,5,18,20].

2.4 THE AORTIC EJECTION CLICK

The aortic ejection click is one of two ejection sounds; the other being an accentuated component of the first heart sound (S1). The accentuated component of S1 is not pathological and so it may occur in both normal and abnormal patients. The aortic ejection click, however, is always associated with valvar aortic stenosis and is therefore considered a reliable indicator of this disease [5,10]. For these reasons it is important to be able to detect the presence of an ejection click independent of whether or not an accentuated component of S1 is present.

When present, the accentuated component of S1 occurs when ejection begins (start of the aortic pressure rise), whereas the ejection click occurs 24 to 40 ms later, coincident with the peak of the anacrotic notch. It is generally agreed that the aortic ejection click is caused principally by the sudden tensing of the aortic valve when it reaches its opening limit during the early stages of systole. Evidence for this is that the click coincides with the full opening of the valve and the fact that calcified (immobile) valves do not click. An additional contribution can occur when the turbulent jet of blood from the valve impinges on the upper wall of the aorta [11].

2.5 THE CAROTID PULSE

The carotid pulse is obtained from the carotid artery in the neck and is thus a reflection of aortic pressure. It is useful in helping to distinguish between the first heart sound and an ejection click. The initial systolic rise of the carotid pulse is normally seen to coincide with an ejection click or to follow it by less than 30 ms. If the transmission delay from the aorta to the neck is taken into account, the rising carotid pulse is seen to precede the ejection sound by approximately 20 ms.

CHAPTER III

DESCRIPTION OF THE DATA BASE

3.1 DATA PROCUREMENT AND INITIAL SELECTION

The phonocardiogram data used here was recorded at Children's Hospital, Boston, Massachusetts and the initial selection was done by Antal A. Sarkady [23]. Data was obtained from thirteen catheterization documented patients ranging in age from 8 to 19 years and six normal subjects 10 to 15 years old. Data from this age group are desirable since children have fewer arterial diseases, such as arteriosclerosis, than adults. The result is a data base less complicated by factors not under investigation.

Recordings were made with Cambridge type 53616 adult size crystal microphones, Cambridge type 72352 amplifier-filters with the filter switch set to the L position, and a four channel PI 6200 analog tape recorder set for FM recording. Before each patient recording was made, a short recording was made with no input to provide background noise information and another recording from a 400Hz acoustical calibration source. This tone was later used to normalize the phono data as described in Chapter IV. Two of the recorder channels were used for phono data from two listening sites, one was used for lead 2 ECG, and the last was used for either carotid pulse or respiration.

An 8 bit analog to digital converter was used to obtain digital data. The sampling rate was 2.5 kHz for the phono recordings and 625 Hz for the ECG, carotid pulse, and respiration recordings. The highest significant frequency components expected were 400 Hz for the phonocardiogram signals and 60 Hz for the other three. The sampling rates for either case were more than three times the Nyquist sampling rate. The highest frequency components were therefore attenuated less than 2 decibels.

The initial selection of data involved selecting cardiocycles of nearly the same length and timing them to begin with the Q point on the ECG. This was done with a precision (Q wave onset jitter) of approximately ±1.6 ms. Cardiocycles were selected to have a Q-Q interval variation of no more than 10 percent. It was found that within the same respiration phase the onset jitter of events associated with systole (the first and second heart sounds and the ejection click) was approximastely ±4 ms. Thus, most of the Q-Q interval variation occurred during diastole. Inspiration (and expiration) cardiocycles were selected as those in which the maximum (and minimum) value of the respiration signal occurred at the middle of the cardiocycle. These cardiocycles therefore occur at approximately maximum (and minimum) lung volumes.

3.2 FINAL SELECTION AND ALIGNMENT

From the data base described above, two groups of cardiocycles were selected for analysis. They were data recorded at mid-inspiration and data recorded at mid-expiration. For each group, phonocardiogram data from the 2nd R.I.S. was used. This auscultation sight was selected since it provides maximum intensity signals from the aortic value area.

Individual cardiocycles could have been analyzed but noise and random artifacts would have caused a low degree of statistical reliability. For this reason it was necessary to improve the signal to noise ratio by generating aligned average cardiocycles. This was done by shifting each cardiocycle so that a particular signal feature (e.g., a strong positive peak in the aortic ejection click) was always aligned with that same feature in all other cardiocycles being averaged. An aligned average cardiocycle provided data in which noise and random artifacts were reduced in amplitude. In addition, it provided some reduction in the amplitude of normal hemodynamic events other than the one used for alignment if there was some variation in time between the two (e.g., a reduction in the first heart sound when the ejection click was used for alignment).

3.3 DATA STORAGE FORMAT

The data for each patient was stored in a separate file which was then given a particular FORTRAN two-digit unit number, XX. The analysis program then obtained data from a particular file by reading from unit XX. The files consisted of 80 byte records (8 bits per byte) with the total number of records determined by the amount of data per patient. (Each record was equivalent to one 80 column card.) Each file consisted of eight contiguous sections, where the first four were fixed-length while the last four could vary in length depending on the number of cardiocycles available for analysis. This file structure is shown in Table 3-1.

Section one consisted of one record of alphanumeric information about the patient, such as name and hospital number. This record was read by the program and printed at the beginning of all patient data plots. However, this information has been deleted from all plots reproduced in this thesis in order to maintain privacy.

The one record in section two contained three numbers which told the program how many cardiocycles where in sections five through seven. (Section eight had the same number of cardiocycles as section seven). These numbers were supplied by the user when the file was created. The format of this record is shown at the bottom of Table 3-1.

TABLE 3-1

PATIENT DATA FILE STRUCTURE

Section	Data Set	Number of Records	Contents of Section
1	-	1	Patient identification
2[1]	-	1	Length information on data sets 3, 4, and 5.
3	1	8	Calibration data
4	2	8	Noise data
5	3	8 times XXX[2]	Phonocardiogram data
6	4	8 times YYY[2]	Phonocardiogram data
7	5	8 times ZZZ[2]	Phonocardiogram data
8	6	2 times ZZZ[2]	Carotid pulse data

[1] Record format:

Column

1	2	3	4	5	6	7	8	9
X	X	X	Y	Y	Y	Z	Z	Z

[2] where:
XXX = No. of cardiocycles in data set 3.
YYY = No. of cardiocycles in data set 4.
ZZZ = No. of cardiocycles in data set 5.

For sections three through eight, each byte contained one data sample. The time between samples was 0.4 ms for sections three through seven and 1.6 ms for section eight.

Section three contained eight records of a calibration signal which was recorded at the time of data collection. The analysis program used the RMS value of this signal to normalize the data, thereby eliminating variations in equipment gain between different recording times. From the program, this section was referenced as data set one.

Section four contained eight records of background noise from the phono recording channel with no input signal. It therefore contained the noise generated by the complete recording process. This section was data set two.

Sections five, six, and seven contained phonocardiogram data. Each section could have any number of cardiocycles with each cardiocycle being eight records (250 ms) in length. The first six samples of each cardiocycle were not used by the analysis program as data. They were reserved for reference information about the recording. These sections were referred to as data sets three, four, and five when called for from the analysis program.

The final section contained carotid pulse or electrocardiogram data recorded at the same time as the phono data in section seven. Due to the longer sample interval, only two records per cardiocycle were needed. This was data set six.

3.4 PATIENT DATA

Personal and catheterization data for the thirteen valvar aortic stenosis patients is given in Table 3-2. The first catheterization data column gives the maximum pressure drop across the open valve while the second column gives the relative severity of the disease as diagnosed by the physician. The final information given is the severity of reverse flow through the closed valve.

Data for the six normal patients is given in Table 3-3. They were included to provide a basis for comparison.

All of the patients in this study had normal body temperatures and no chest deformities.

TABLE 3-2

VALVAR AORTIC STENOSIS PATIENT DATA

-----Personal Data-----				-------Catheterization Data-------		
Patient Number	Age	Sex	Chest Wall	P.S.E.G. (mmHg)	Cath. Diagnosis	Aortic Regurgitation
1	9	M	Thin	5-9	Triv.	Triv.
2	9	M	Thin-Med.	45	Mod.	----
3	14	M	Thin	39	Mild	----
4	12	F	Thin	45	Mod.	----
5	11	M	Med.	61-68	Mod.-Sev.	Mild
6	10	M	Thin	6-8	Triv.	Triv.
7	19	M	Med.	45	Mild-Mod.	Mild
8	15	F	Med.	70-90	Mod.-Sev.	Mild
9	10	M	Med.	16-24	Mild	----
10	8	F	Med.	23	Mild	Triv.
11	16	M	Thick	75	Mod.	----
12	10	M	Thin	16	Triv.	Mild
13	10	M	Thin	9-18	Mild	----

TABLE 3-3

NORMAL PATIENT DATA

Patient Number	Age	Sex	Chest Wall
14	13	F	Med.
15	15	M	Med.
16	10	M	Thin
17	10	M	Thin
18	10	M	Thin
19	13	F	Thin

CHAPTER IV

SIGNAL PROCESSING TECHNIQUES

4.1 INTRODUTION

This chapter describes and mathematically defines the signal processing techniques used in the Interactive Phonocardiogram Analysis Program described in the following chapter. Since digital computers must operate with discrete digital data values, continuous analog data must be converted to digital form by a sampling process.

The first part of this chapter discusses this process and the various results of its use. The following section defines the discrete Fourier transform (DFT) and its inverse (IDFT) and also discusses a computationally efficient way of calculating these, the fast Fourier transform (FFT). The FFT was the principal computational tool used in this work. The next two sections define the power spectrum and envelogram analyses while the final section is a discussion of signal averaging.

4.2 IMPULSE SAMPLING

Consider a continous function of time fc(t) which is to be converted to a sampled function fs(t). This is done by multiplying the continuous function by a sampling function s(t) [24]. Thus:

$$fs(t) = fc(t)s(t) \qquad (4.2.1)$$

The sampling signal is made up of a series of unit impulses spaced T seconds apart. Thus:

$$s(t) = \sum_{n=-\infty}^{+\infty} d(t-nT) \qquad (4.2.2)$$

where: d(t-nT) is a unit impluse at the time nT.

For $Fc(j\omega)$, the Fourier transform of the continuous signal, the Fourier transform of the sampled signal becomes:

$$Fs(j\omega) = (1/T)\sum_{k=-\infty}^{+\infty} Fc[\,j(\omega-k\,\omega_0)\,] \qquad (4.2.3)$$

where: $\omega_0 = 2\pi/T$. From this equation it can be seen that the Fourier transform of the sampled signal is made up of the continuous signal transform reproduced at the interval ω_0. Thus, sampling in the time domain produces a periodic signal in the frequency domain. Since in digital signal processing the Fourier transform must also be represented in sampled form, it is also a sampled signal thereby implying periodicity in the time domain.

For a bandwidth limited time signal it has been shown [25] that the sampled signal completely defines the continuous signal if the sampling rate is sufficiently high. This is known as the sampling theorem which if stated mathematically says that if:

$$T \le 1/2fm \qquad (4.2.4)$$

where: T is the sample period in seconds and

fm is the maximum frequency component in Hertz, then the continuous signal is completely defined.

4.3 THE FOURIER TRANSFORM

The continuous Fourier transform pair can be expressed as:

$$Fc(f) = \int_{-\infty}^{\infty} fc(t)\exp(-j2\pi ft)\,dt \qquad (4.3.1)$$

$$fc(t) = \int_{-\infty}^{\infty} Fc(t)\exp(+j2\pi ft)\,df \qquad (4.3.2)$$

where: Fc(f) is the continuous frequency domain function,

fc(t) is the continuous time domain function, and

j is the square root of -1.

The analogous pair for discrete signals can be expressed as [26]:

$$Fs(m) = (1/N)\sum_{n=0}^{N-1} fs(n)\exp(-j2\pi mn/N) \qquad (4.3.3)$$

$$fs(n) = \sum_{m=0}^{N-1} Fs(m)\exp(j2\pi mn/N) \qquad (4.3.4)$$

where: $m = 0,1,\ldots,N-1$, and

$n= 0,1,\ldots,N-1$.

A direct calculation of the discrete Fourier transform by (4.3.3) would require nearly N^2 complex operations, an excessively large number for practical use. The development of an algorithm by Cooley and Tukey [27] known as the fast Fourier transform (FFT) greatly reduces he number of calculations required to obtain a discrete Fourier transform. When N is a power of 2, only (N/2) $\log_2 N$ complex multiplications, additions, and subtractions are needed. For a 1024 point transform this is a computational savings of over 200 to 1 [26].

The FFT subroutine used here [28] is based on the original Cooley-Tukey algorithm in that it allows N to have factors other than 2. This particular implementation allows N to have prime factors up through 23.

Frequently it is desired to obtain the FFT of a time series which is shorter than the length of the transform. In this case the remaining points can be set to zero. The result of this procedure is a broadening of the frequency components of the time series due to the time data window being shorter than the transform. The Fourier transform of a rectangular window is a Sin(X)/X function which when convolved with the time series spectra in the frequency domain produces the broadening of each frequency component. The narrower the time window, the broader each component becomes. For a time window Tw seconds wide, the resolution fo in the frequency domain will be [29]:

$$fo = 1/Tw \qquad (4.3.5)$$

4.4 POWER SPECTRA

The discrete power spectral estimate Ps(j) of a time series is obtained from its Fourier transform by means of:

$$Ps(m) = (1/Tw)[Fs(m)Fs^*(m)] \qquad (4.4.1)$$

where: Fs(j) is the Fourier transform of the time series,

$*$ denotes complex conjugate, and

$m = 0, 1, \ldots, (N/2)-1$.

Only the first N/2 terms are used since the remaining ones are the negative frequency terms.

The complete procedure used for obtaining a power spectral estimate from a phonocardiogram consisted of the following steps:

1. Remove any DC bias present in the time series data.
2. Normalize the data by dividing it with the rms value of the calibration signal.
3. Select the desired portion of the data with a rectangular window.
4. Compute the complex FFT for N points.
5. Compute the power spectral estimate by (4.4.1) for the first N/2 points.

4.5 ENVELOGRAM

The envelogram estimate of a time series signal is a positive valued function which gives a high resolution graph of the signal amplitude. It is actually the magnitude of the analytic signal $v(n)$ which is [30]:

$$v(n) = fs(n) + jH[fs(n)] \tag{4.5.1}$$

where: $H[*]$ is the Hilbert transform operator.

An envelogram can be calculated by the following procedure [31]:

1. Compute the DFT of an N point time series fs(t).
2. Set the terms from $n = N/2$ through $n = N-1$ (i.e., the negative frequency terms) to zero.
3. Multiply the terms from $n = 1$ through $n = (N/2)-1$ (i.e., the positive frequency terms) by 2. Note that the $n=0$ (i.e., DC) term is not changed.
4. Compute the IDFT.
5. Compute the magnitude of the complex result.

4.6 AVERAGING

Due to noise and random artifacts in phonocardiogram data, a power spectral estimate or envelogram estimate made from a single time series record is not statistically reliable. This situation can be improved by adding a number of records together point by point and dividing by the number added together. This averaging process can be done either before or after the power spectrum (or envelogram) is computed.

When averaging is done first, a further reduction in unwanted signals can be achieved by performing an aligned averaging operation in which each record is shifted so that a particular signal feature occurs at the same point in time when the records are added together. This technique has the ability to suppress signal features which are not synchronized in time with the signal feature used for alignment.

Averaging reduces the variance of power spectra and envelogram estimates by the square root of the number of records averaged [31].

CHAPTER V

INTERACTIVE PHONOCARDIOGRAM ANALYSIS PROGRAM

5.1 OVERALL PROGRAM STRUCTURE

The FORTRAN program developed for this research is described and instructions for its use are given in this chapter. The program was run on an IBM Virtual-Machine/370 (VM/370) under the IBM conversational Monitor System (CMS). It was compiled with the FORTRAN H-extended compiler. The program (listed in Appendix 1) was divided into four separate CMS files as shown in Table 5-1.

The interactive character of this program allows the operator to process data and obtain immediate results in the form of plots at his terminal. This quick turnaround time helps in developing the proper analysis sequence and it encourages more investigation of the data. The program prompts the user for all required entries by stating what is to be entered next or asking a question. There are also warnings and explanations given at several points.

The flow of data into, within, and out of the program is depicted in Fig. 5-1. The X and Y buffers are both 1000 samples long and store both real and imaginary data components. During execution of the RUN command, individual cardiocycles are input from one of the patient files to the

TABLE 5-1

PHONOCARDIOGRAM PROGRAM FILE DIVISIONS

CMS File Name[1]	File Length[2]	Content
HRT	734	Main program; subroutines RUN, CINPT, INPUT, IO, SYNCH, ANLYZ, RUNAN, CPLOT, TRNSF, GATE, ICCON, CLEAR, DIVIDE, NORM, RDTTY, RDANL, PWRSPM, ANLSIG, MAG, and pseg; BLOCK DATA program.
FFT	627	FFT subroutine
TEST	171	TEST, NOISE, RANDU, and GAUSS subroutines
TPLOT	261	TPLOT subroutine

[1] All files had a CMS file type of FORTRAN.

[2] Number of 80 byte FORTRAN records (or cards).

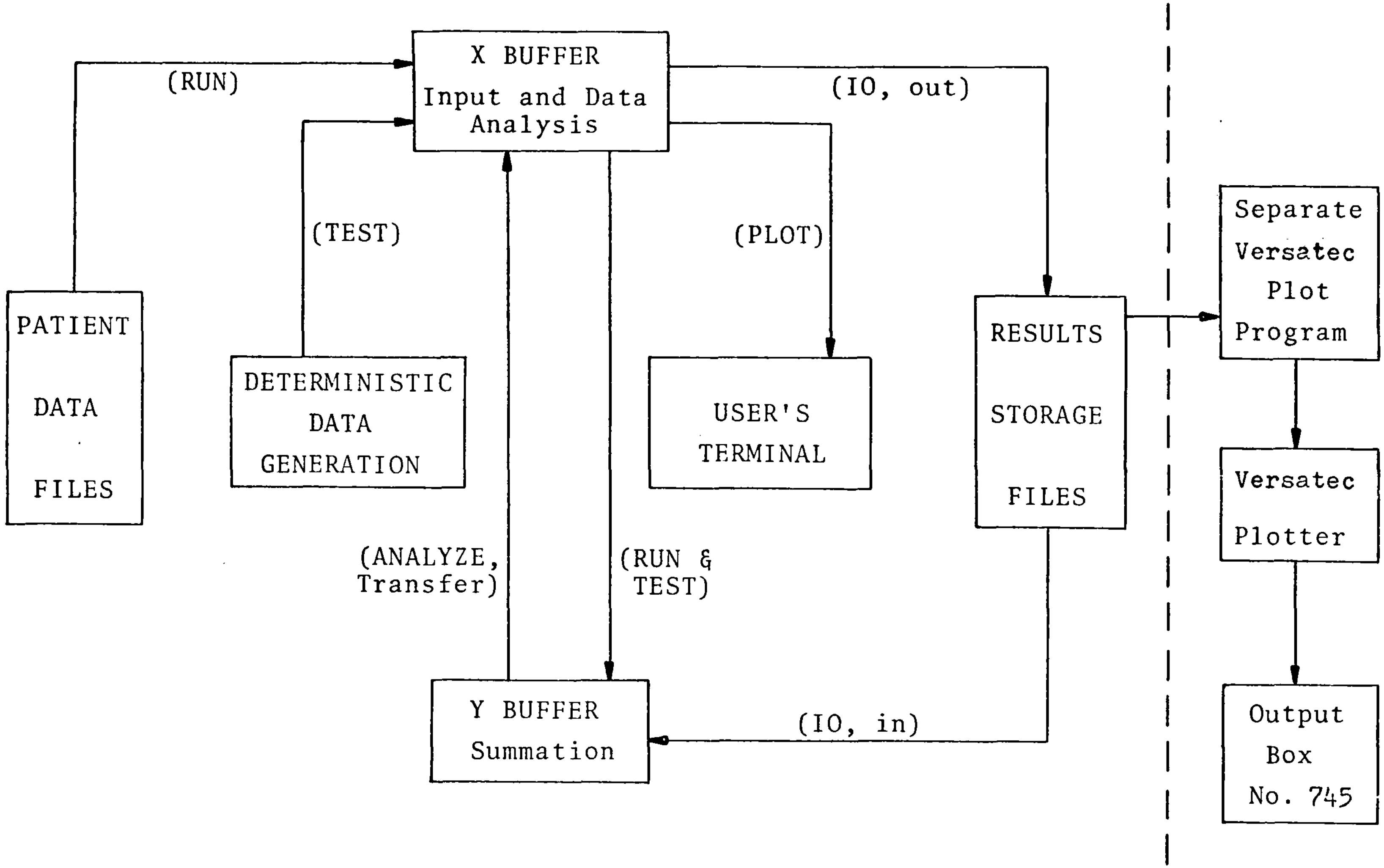


Fig. 5-1 Overall Program Data Flow

X buffer where they are analyzed and summed into the Y buffer. Alternatively, deterministic signals can be generated under the TEST command and input in like manner. The resulting sum in the Y buffer can then be transferred back to the X buffer for further analysis, from where it can be sent to the terminal as a plot with PLOT command or output to a disc storage file under the INPUT-OUTPUT (IO) command. Data in storage can then be returned to the Y buffer under the IO command for further analysis at a later time. The Calcomp Plot Program (listed in Appendix 2) was a completely separate program which could be submitted to the batch processor from CMS to obtain high resolution plots. The storage file to be plotted was appended to this program as data.

Operator responses to the program are of two types, textual and numeric. Textual responses are commands or answers to questions and may be preceded by any number of blanks. Numeric responses are read with a free format so they may be spaced in any manner, entered with or without decimal points or exponents, and have up to seven significant figures. However, each number must be separated by a space or comma, all items requested must be furnished, and the complete response must be on one line. When responding, the operator types a response and presses the "return" key on the terminal.

The overall command structure of the program is shown in Fig. 5-2. Input-output blocks (parallelagrams) indicate interaction between the program and the terminal. Output from the program is given in quotes while operator responses are indicated with an R. At the start of the program the seed for random number generation is initialized and the Y buffer and imaginary component of the X buffer are cleared. (The real component of the X buffer is effectively cleared when data is input). The program then enters the main command loop with the prompt "Enter Command". When entered, the command is decoded and executed, and the program returns to request another command. If the response was undecodable, "Unknown Command" is printed before returning to the beginning of the loop. Six of the commands require further interaction which is shown in the additional figures indicated. Square boxes with an X indicate execution of the command. The following sections describe each command, explaining any required responses and the operation performed.

All of the commands may be abbreviated, and four of them must be abbreviated to a certain extent. Table 5-2 shows the four required abbreviations and the minimum abbreviations.

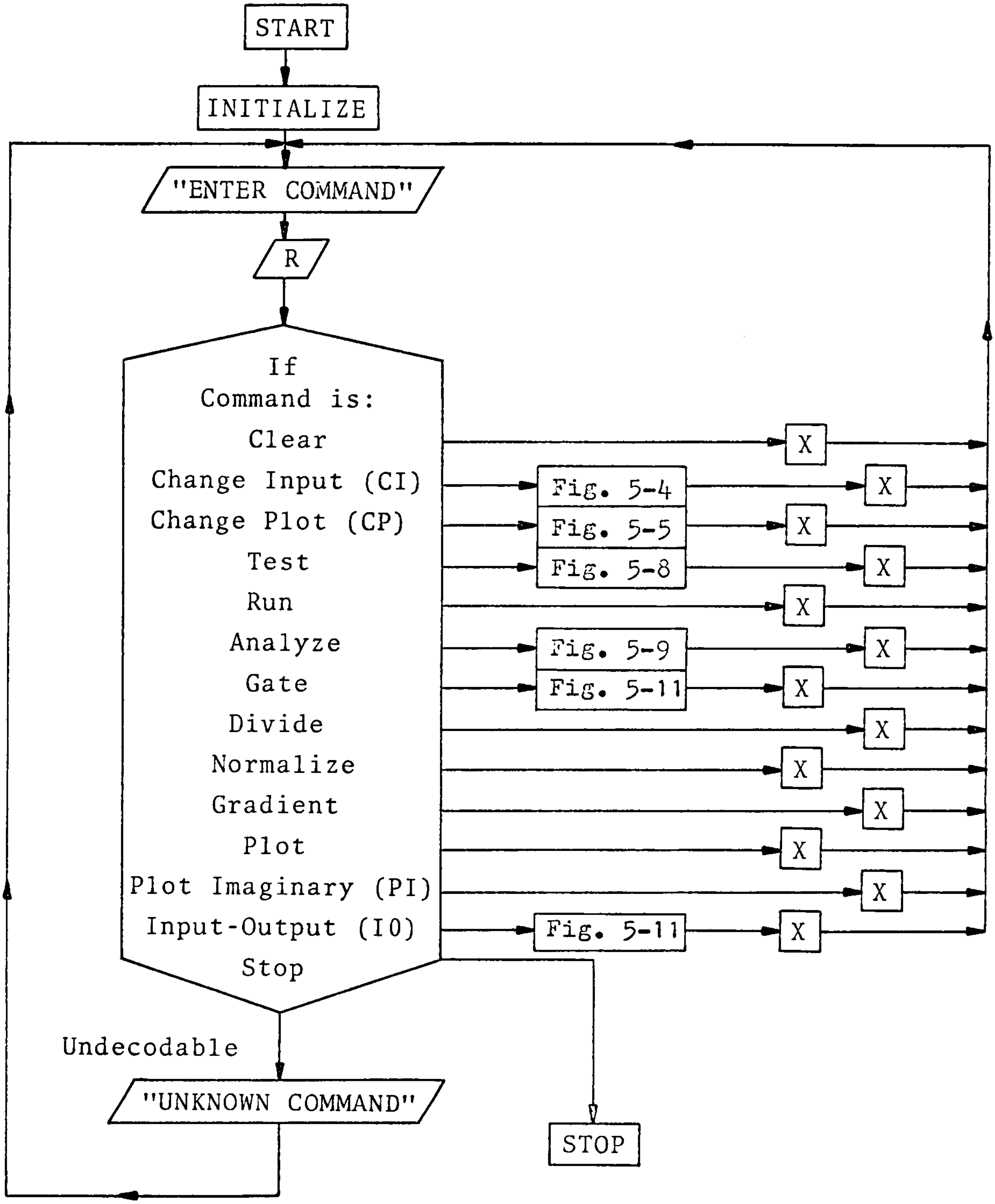


Fig. 5-2 Overall Program Flow Chart

TABLE 5-2

MAIN COMMAND ABREVIATIONS

Command	Required Abbreviation	Minimum Abbreviation
CLEAR	None	CL
CHANGE INPUT	CINPUT	CI
CHANGE PLOT	CPLOT	CP
TEST	None	T
RUN	None	R
ANALYZE	None	A
GATE	None	G
DIVIDE	None	D
NORMALIZE	None	N
GRADIENT	None	GR
PLOT	None	P
PLOT IMAGINARY	PIMAGINARY	PI
INPUT-OUTPUT	IOUTPUT	IO
STOP	None	S

5.2 MAIN COMMANDS

5.2.1 CLEAR

The CLEAR command completely clears the Y buffer. This function is also performed automatically when the program first begins and when the RUN command is executed. The CLEAR command is normally used before the TEST command.

5.2.2 CHANGE INPUT

The CHANGE INPUT command is used to set up the input parameter list. The program then follows this list when inputing patient data. Figure 5-3 shows the sequence of events when patient data is input under the run command. There is an overall patient loop which allows the same analysis to be performed on more than one patient, and an inner cardiocycle loop which analyzes cardiocycles individually as they are input.

To begin, the program selects the data file of the first patient, clears the Y buffer, reads the patient header, and determines the normalization constant. It then enters the cardiocycle loop where each cycle is read in, analyzed, and summed into the Y buffer. After the specified number of cycles have been processed, a final analysis on the sum is performed and the program either loops back for the next patient or returns to the main command loop.

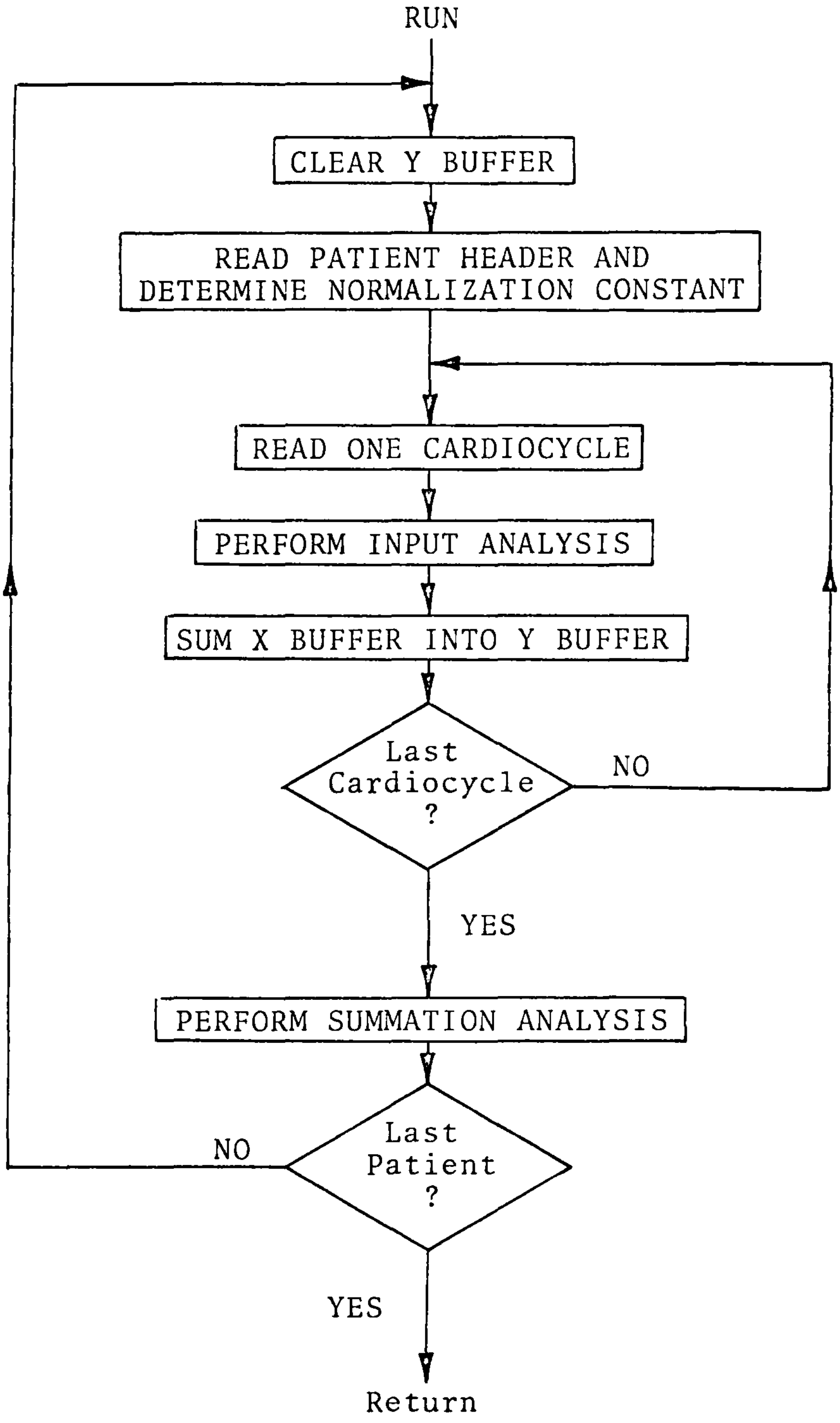


Fig. 5-3 Run Command Flow Chart

With this in mind, the interaction required for the change input command, as shown in Fig. 5-4, can be explained. The response following the prompt "enter first and last patient" consists of two numbers which tell the programs the first and last patients in a sequence to be analyzed. (The first patient file must be FORTRAN unit number eleven with the remaining files following sequentially). If only one patient is to be analyzed, that patient number must be entered twice. The program then requests five numerical items: the data set from which data will be taken (Table 5-2), the first and last times (in milliseconds) of the interval to be input, the number of repetitions (i.e., the number of cardiocycles to be processed), and the total desired length (number of samples) of the analysis. Zeroes will be padded if necessary. The analysis length parameter defines the length of the Fourier transform, so a value should be chosen which gives an appropriate frequency scale (500 is good). The only limitations on the analysis length is that it must not be greater than 1000 (the array length) or have any prime factor greater than 23. (This last requirement comes from the FFT subroutine) [28]. The times of data input have a zero reference at the beginning of each cardiocycle as stored in the patient file.

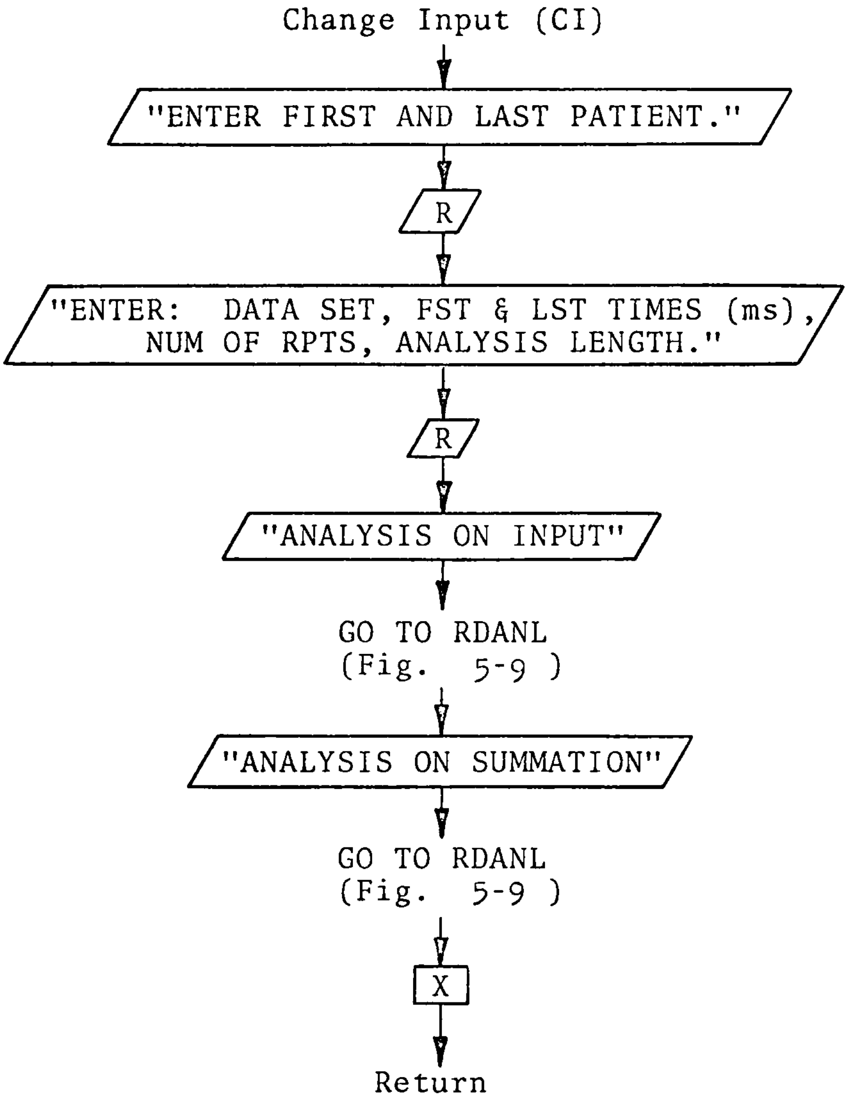


Fig. 5-4 Interaction of the Change-Input Command

The program then states "analysis on input" and branches to the read-analysis (RDANL) subroutine which requests the sequence of analyses to be done on each cardiocycle as it is input, i.e., before summation into the Y buffer. The interaction of this subroutine and the analyses available will be discussed under the ANALYZE command. The statement "analysis on summation" is made and a second branch to RDANL requests the analyses to be performed on the summation.

5.2.3 CHANGE PLOT

The CHANGE PLOT command is used to change the parameters which control the plotting of data, the destination of the plot, and any comments to be printed at the beginning of the plot. Actual plotting is done with the PLOT command. As shown in Fig. 5-5, the first item requested is the plot index, an integer number that controls which points from the array will be plotted. A one will plot all points, a two every other point, etc. When not plotting all points, the program still searches them all when determining the vertical scale to be used. In this way the maximum data value in the range plotted is still indicated. The program then asks where the plot is to be sent; to the user's terminal or a remote printer. The response is "terminal" or "remote", with a minimum

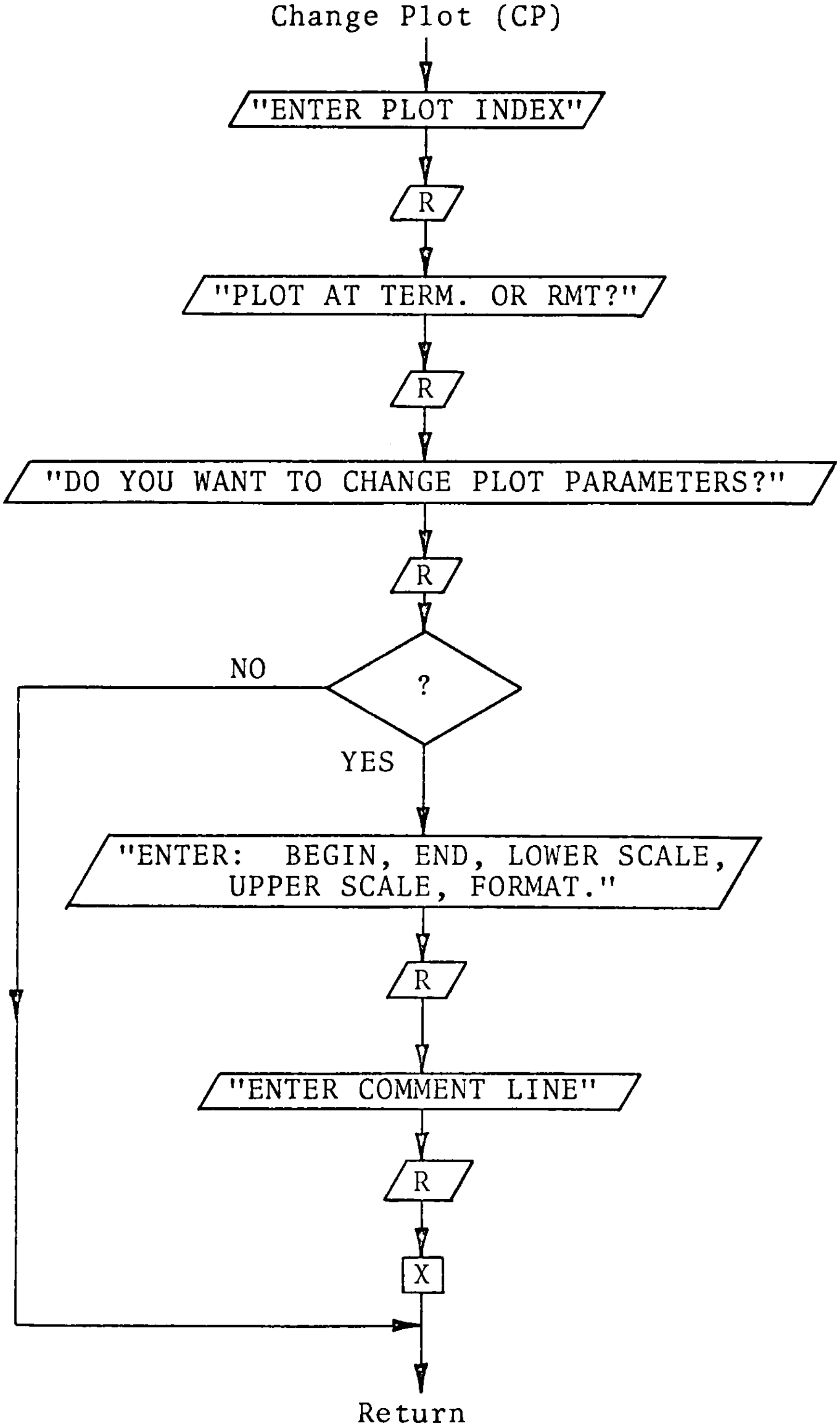


Fig. 5-5 Interaction of the Change Plot Command

abbreviation of "t" or "r" respectively. The program then asks if the operator wants to change the remaining parameters. The response is "yes" or "no" with a minimum abbreviation of "y" or "n". If the response is negative, the remaining parameters remain unchanged and the program returns to the main command loop. For a "yes" response, the program requests five plot parameters. At this point the various types of plots must be discussed.

For convenience, the type of plots will be referred to by a three character code; the first two being letters, the last a number. The first letter indicates the polarity of the data (P, N, or B), the second letter indicates one of three options pertaining to the vertical scale values (A, B, or C), while the number indicates one of three options for fill-in characters (1, 2, or 3). The use of an X in any position indicates irrelevance of that position to the discussion.

The plot subroutine determines the maximum and minimum values of the data in the range to be plotted. From these values the program decides on the polarity and the peak values of the data. If the data is unipolar, it is plotted with zero at the bottom and a full scale value equal to the peak data value. The type of plot which results is PAX for positive data or NAX for negative data. If bipolar, the data is plotted with a zero axis down the middle of the graph, and a full scale value (each side of zero) equal to the larger of the positive and negative maxima. This results in a type BAX plot. As an option, the user can specify the full scale value to be used. The program then selects only between unipolar and bipolar plotting based on the data. The result is a type XBX plot. A second option is that the user can specify the literal values to be used for the upper and lower scale limits, the result being a type CX plot. (The first letter in the plot-type code becomes irrelevant for a type C plot).

Regarding fill-in characters, the user can specify a type XX1 plot which uses no fill-in, a type XX2 plot which places cross marks at five unit intervals, or a type XX3 plot which gives cross marks plus full dash-mark fill-in.

Returning to Fig. 5-5, the five parameters requested are a beginning number, an ending number, lower and upper scale values, and a format (plot type) control number. The beginning number tells at which point in the array plotting is to begin, the ending number where it is to end. These numbers refer to actual samples in the array, not time values. The upper and lower scale values along with the sign of the format number specify type XAX, XBX, or CX plots. For all plots, the magnitude (1, 2, or 3) of the format number controls the type of fill-in, giving type XX1, XX2, or XX3 plots.

To produce type XAX plots, the lower and upper scale values must be zero and the sign of the format number positive. The program then automatically scales the plot and selects the appropriate plot polarity. For type XBX plots, the lower scale value must be zero, the upper scale value must equal the desired full scale value, and the sign of the format number must be positive. The program still decides, from the data, what polarity form the plot will take. For type CX plots, the sign of the format number must be negative. The values then given to the lower and upper scale values specify explicitly the lower and upper vertical scale limits. A summary of the various types of plots is given in Table 5-3, and several examples are shown in Fig. 5-6.

After entering the five parameters discussed above, the program requests a comment line which will be printed at the beginning of the plot. This line may contain up to 128 characters. If the line is to be left unchanged, a null line may be entered. This completes the CHANGE PLOT command; the program returns to the main command loop.

5.2.4 TEST

The TEST command is used to generate deterministic signals for testing the program, for experimentation, or for comparison with patient data. Any number of padding zeroes

TABLE 5-3

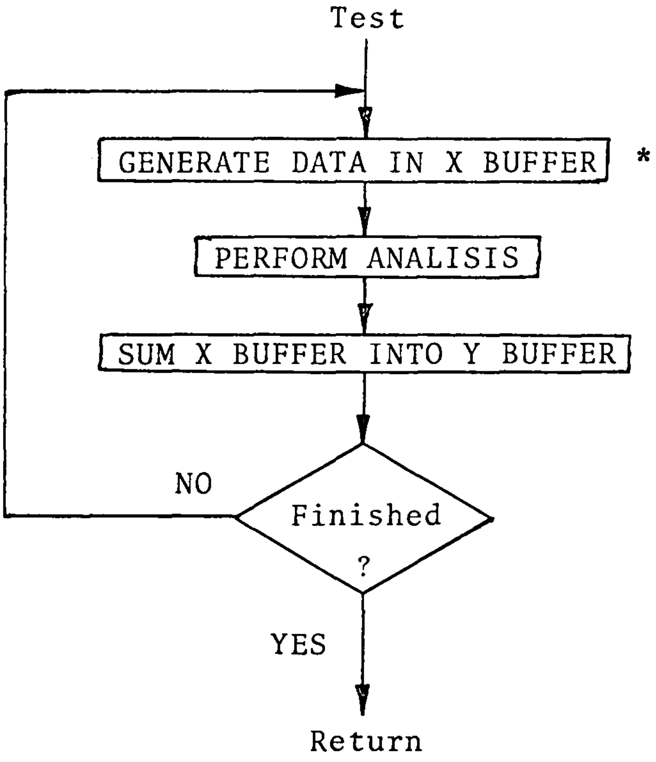
PLOT TYPE SUMMARY

(An X indicates irrelevance)

Polarity	Lower Scale	Upper Scale	Format Number	Plot Type
+	0	X	+X	PXX
-	0	X	+X	NXX
±	0	X	+X	BXX
X	0	0	+X	XAX
X	0	>0	+X	XBX
X	X	X	-X	CX
X	X	X	±1	XX1
X	X	X	±2	XX2
X	X	X	±3	XX3

can be added before or after the signal so long as a total length of 1000 samples is not exceeded. The equation used to generate each sample is given at the bottom of Fig. 5-7. It contains four terms multiplied together plus a noise term added to the final product. The first four terms consist of a constant, K, a decaying exponential with a time constant, tau, a cosine of frequency f1 and phase phi1, and a second cosine of frequency f2 and phase phi2. Pseudorandom noise can be added if desired with the value for each sample either uniformaly distributed in amplitude between specified limits or Gaussianly distributed with a given mean and standard deviation. The entire signal can be regenerated any number of times for tests involving noise. The CLEAR command must be executed before using the TEST command since the deterministic signals are added to what is already in the Y buffer. This arrangement allows for complex signals to be synthesized by repeated executions of the TEST command.

When the TEST command is given, the program requests eleven numerical items as stated at the top of Fig. 5-8. Values for the constant, tau, freql, phil, freq2, and phi2 go into the equation described. If the value for tau or either frequency is given as zero, the associated term is deleted from the equation. The sample interval is the amount of time between successive samples, T in the



$*X_n=(K)\exp\{-(nT)/\tau\}\mathrm{Cos}\{2\pi f_1(nT)+\phi_1\}\mathrm{Cos}\{2\pi f_2(nT)+\phi_2\}+\mathrm{Noise}$

$K, T, \tau, f_1, \phi_1, f_2, \phi_2$, and the parameters for noise generation are input by the user.

Fig. 5-7 Test Command Flow Chart

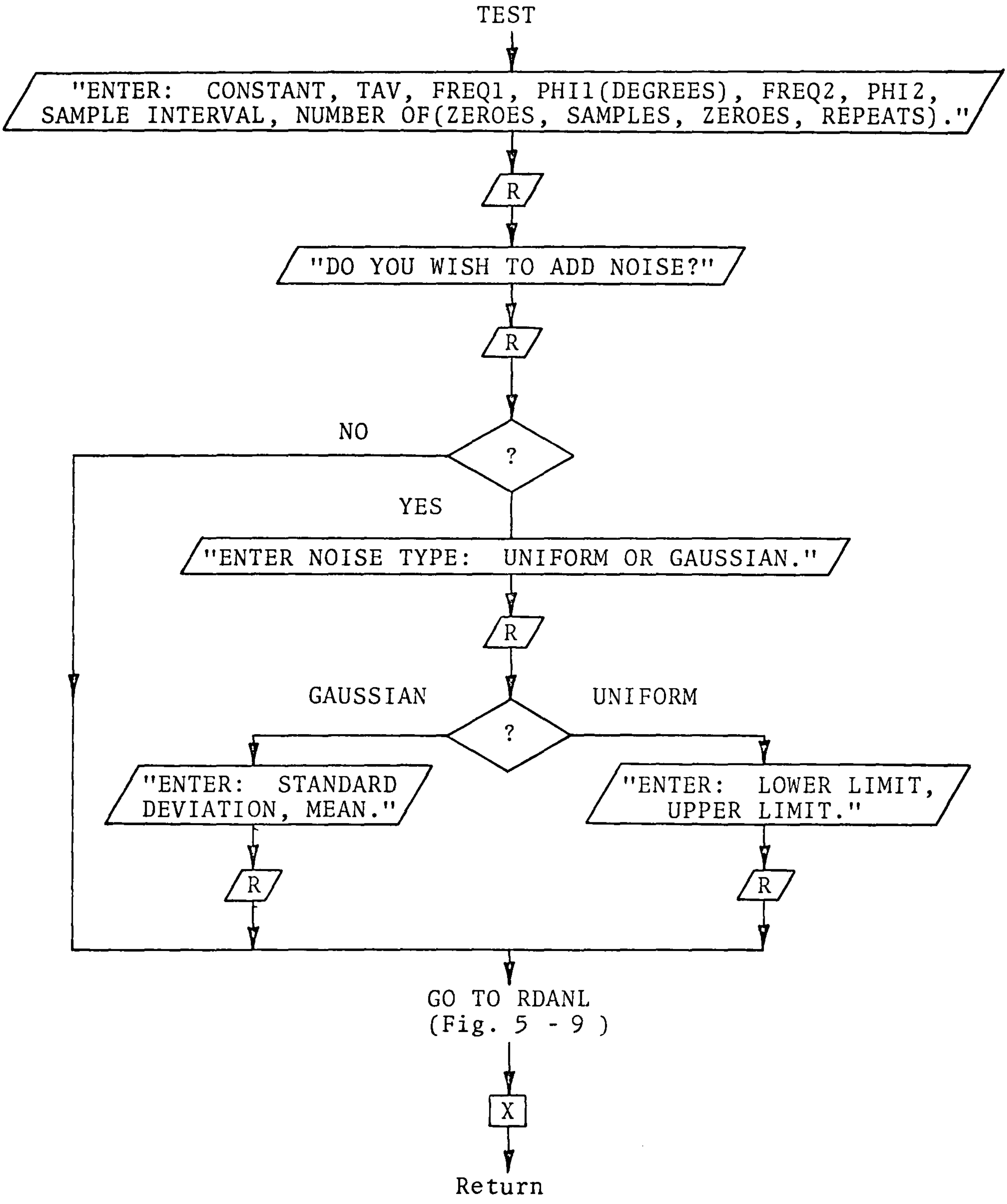


Fig. 5-8 Interaction of the Test Command

equation. The four remaining numbers specify in order, the number of zeroes preceding the test signal, the number of signal samples, the number of zeroes following the test signal, and the number of times the entire signal is to be generated. If this last value is given as zero, the program returns immediately to the main command loop without executing the TEST command. All units are assumed to be in seconds, hertz, or degrees as appropriate. The first signal sample is calculated for nT = 0 whether or not leading zeroes are present. The program then asks if the operator wishes to add noise. The response is "yes" or "no" with a minimum abbreviation of "y" or "n". If the response is "yes", the program requests the type of noise desired, uniform or Gaussian. The operator replies; the minimum abbreviation being "u" or "g". The program then requests the upper and lower limits for uniform noise, or the standard deviation and mean for Gaussian noise. These two values are entered by the operator and the program flow joins that from a "no" response to noise generation. A branch is then made to the RDANL subroutine to request the analyses to be performed on each test signal before it is summed. The three analyses available are Spectrum, Analytic Signal, and Magnitude. These are described under the ANALYZE command. After the TEST command has executed, additional analyses can be performed with the ANALYZE command.

5.2.5 RUN

The RUN command performs the actual inputing and individual cardiocycle analysis of patient data. It is controlled by the input parameter list as set up with the CHANGE INPUT command. The action of the RUN command has been described under the CHANGE INPUT command with reference to Fig. 5-4. There is no interaction required unless the Gating analysis was specified. In this case the gating times will be requested, as described under the GATE command, when that subroutine is executed. However, the program does indicate the beginning of analysis for every fifth cardiocycle by printing cycle 5, cycle 10, etc. This gives the operator an indication of how fast the analysis is proceeding.

5.2.6 ANALYZE

The ANALYZE command can be used to transfer data from the Y buffer to the X buffer, perform any available analysis on it, or output it to disc storage or to the terminal as a plot. All analysis of data after it has been input is performed in the X buffer. When this command is given, the program branches to the read-analysis (RDANL) subroutine for which an interaction flow chart is given in Fig. 5-9. This subroutine requests the series of analysis the operator desires to have performed. It consists of a loop which

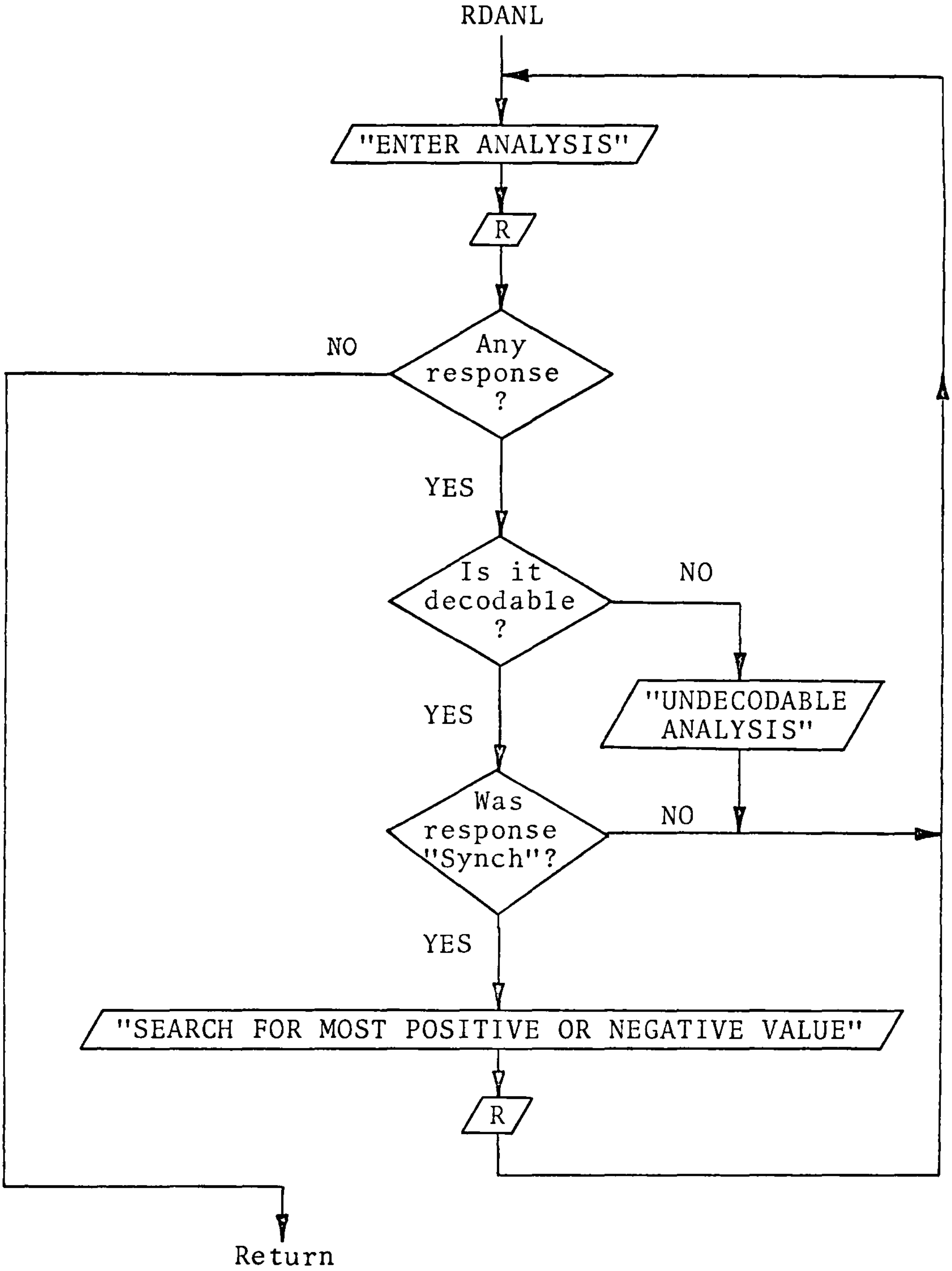


Fig. 5-9 Interaction of the Analyze Command

repeatedly requests an analysis until the operator enters a null line which signifies the end of an analysis sequence. The program states "enter analysis", after which one of the analyses listed in Table 5-4 may be entered. If no analysis (i.e., a null line) is entered, the program returns to execute those already entered. (Execution occurs in the order in which the analyses were given.) When an analysis is entered, it is added to the analysis sequence and the program loops back to request another analysis. If an entry was undecodable, "undecodable analysis" is printed before looping back. A request for the Synch analysis results in the question "search for most positive or negative value?" after which the operator replies with "positive" or "negative"; first letters being permissible as minimum abbreviations. A maximum of ten analyses may be requested each time the ANALYZE command is executed.

The "analyses" Gate, Divide, Gradient, Plot, Plot Imaginary, and Input-Output are actually the main commands of the same name, included under the ANALYZE command for convenience. The Spectrum analysis calculates the power spectrum of a time series signal by means of a fast Fourier transform (FFT) subroutine. The FFT used here [28] allows the time series to have a sample length other than a power of two which can result in convenient whole number frequency scales. The Analytic Signal analysis computes the envelope of the signal; the result being complex. Therefore, this

TABLE 5-4

ANALYSES AVAILABLE FOR THE ANALYZE COMMAND

Analysis	Minimum Abbreviation
Spectrum	SP
Analytic Signal	A
Magnitude	M
[1]Gate	G
[1]Divide	D
[1]Gradient	GR
[1]Plot	P
[1]Plot Imaginary	PI[2]
[1]Input-Output	IO

[1] Also available as main commands.

[2] Must at least be abbreviated to PImaginary.

analysis is normally followed immediately by the Magnitude analysis which computes the magnitude of the complex X array and places the result in the real part of the X array. The Transfer analysis simply transfers the contents of the Y array to the X array. Normally this is always done immediately after inputing data in order to return the summation to the X array where all analyses are performed. The program keeps track of all analyses performed and this list is printed before all data plots for reference.

5.2.7 GATE

The GATE command is used to select a portion of the X buffer for further analysis, setting the remainder to zero. The portion selected is placed at the beginning of the buffer. Figure 5-10 shows the function of the GATE command within the overall program. A section of data (e.g., 60 ms to 200 ms) from a patient file was input to the X buffer by the RUN command and Transfer subcommand. A small portion (e.g., 120 ms to 150 ms) of this data was then selected for later analysis (typically Spectrum) by the GATE command. The interaction for this command is shown in Fig. 5-11. The program requests the beginning and ending times of the interval to be gated out. These are entered (in milliseconds) and the command is executed.

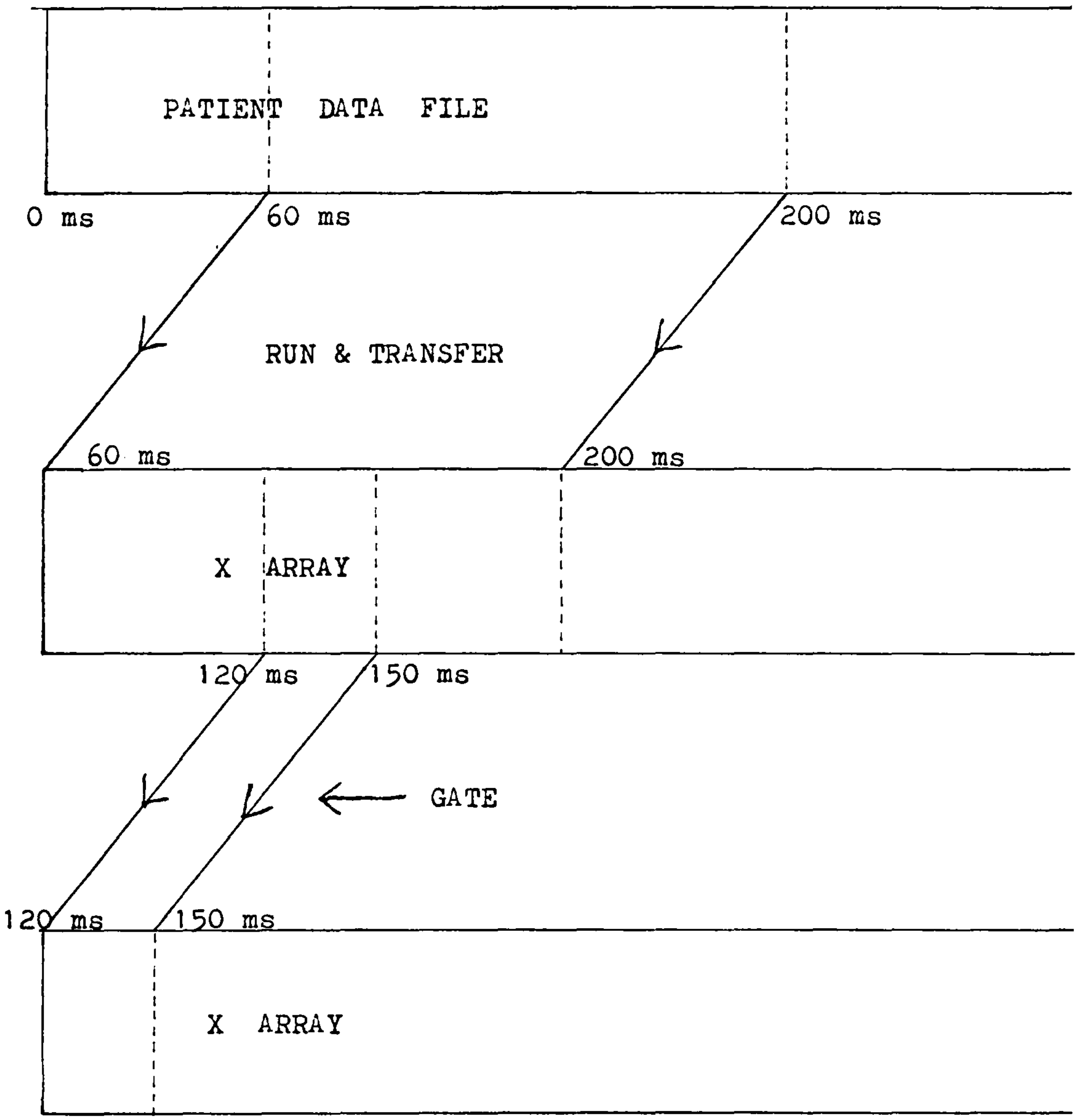


Fig. 5-10 Action of the Gate Command

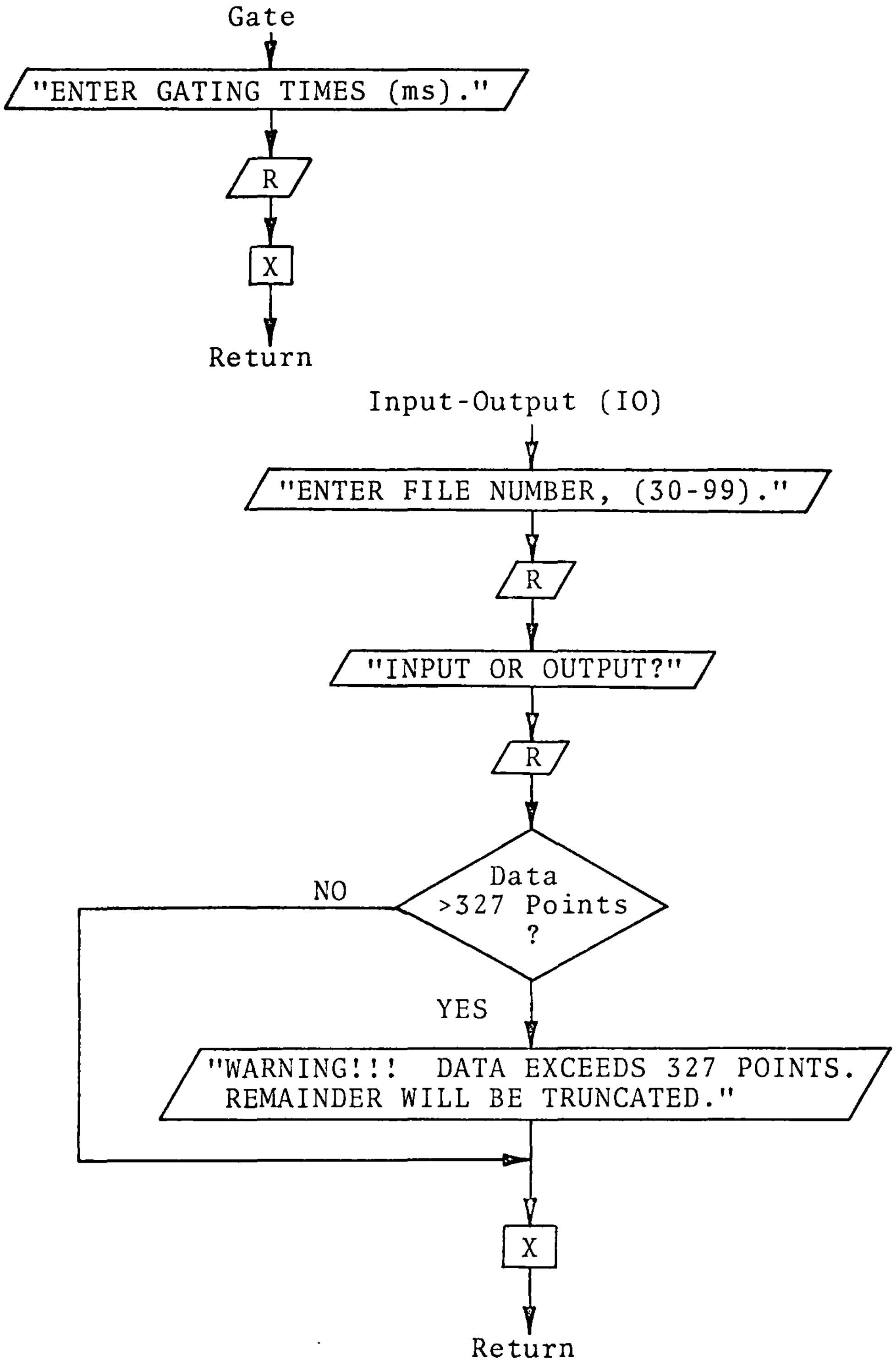


Fig. 5-11 Interaction of the Gate and Input-Output Commands

5.2.8 DIVIDE

The DIVIDE command is used to divide the summation in the Y buffer by the number of cardiocycles in the summation. The Y buffer then contains the average cardiocycle. Normally, this command must always be given after inputing data with the RUN command. No operator interaction is required.

5.2.9 NORMALIZE

The NORMALIZE command can be used to set the constant which normalizes patient data as it is input. This constant is determined from the calibration record of the patient to be analyzed as specified during the last execution of the CHANGE INPUT command. During normal use of the program this command need not be explicitly executed as it is performed automatically under the RUN command.

5.2.10 GRADIENT

Execution of the GRADIENT command generates an estimate of the peak systolic ejection gradient (P.S.E.G.) across the aortic value, based on the murmur power spectrum. It has been shown [24] that a high correlation exists between catheterization data for the P.S.E.G. and the first moment of the mean murmur power spectrum. The program calculates the P.S.E.G. from the equation:

$$P.S.E.G. = 0.634\ f - 46.0$$

where: P.S.E.G. is in millimeters of mercury
f is the first moment of the mean murmer
power spectrum, in hertz.

The program also computes an estimate for the standard deviation of f (   ) and from that the standard deviation of the P.S.E.G. This information is then printed in the form:

P.S.E.G. = XXX.X mmHg ±XX.X
F Bar = XXX.X cps ±XX.X

5.2.11 PLOT and PLOT IMAGINARY

The PLOT and PLOT IMAGINARY commands result in the plotting of the real or imaginary components, respectively, of the X buffer. Plotting is immediate; the form being controlled by the parameters set up previously by the CHANGE PLOT command. Two different headers can precede the plot, depending on whether patient or deterministic data is being plotted. Figure 5-12 shows a plot of deterministic data. The header gives the constants used to generate the data, the analyses performed before and after summing the data, the comment line, and the plot index. If more than one execution of the TEST command was used to generate the data, this will be indicated by more than one "end" statement on the "analysis before summation" line. However, the data

```
CONSTANT        TAU       FREQ1       PHI1       FREQ2       PHI2       SMPL INT    ZERO  SMPL  ZERO  RPTS  NSTP   NOISE PARAMATERS
   .2000       .0         70.00      -90.00      .0          .0          .1000E-02    0     35    0     1     1    .0            .0

ANALYSIS BEFORE SUMMING:    END  END


PLOTTING INDEX =  1
```

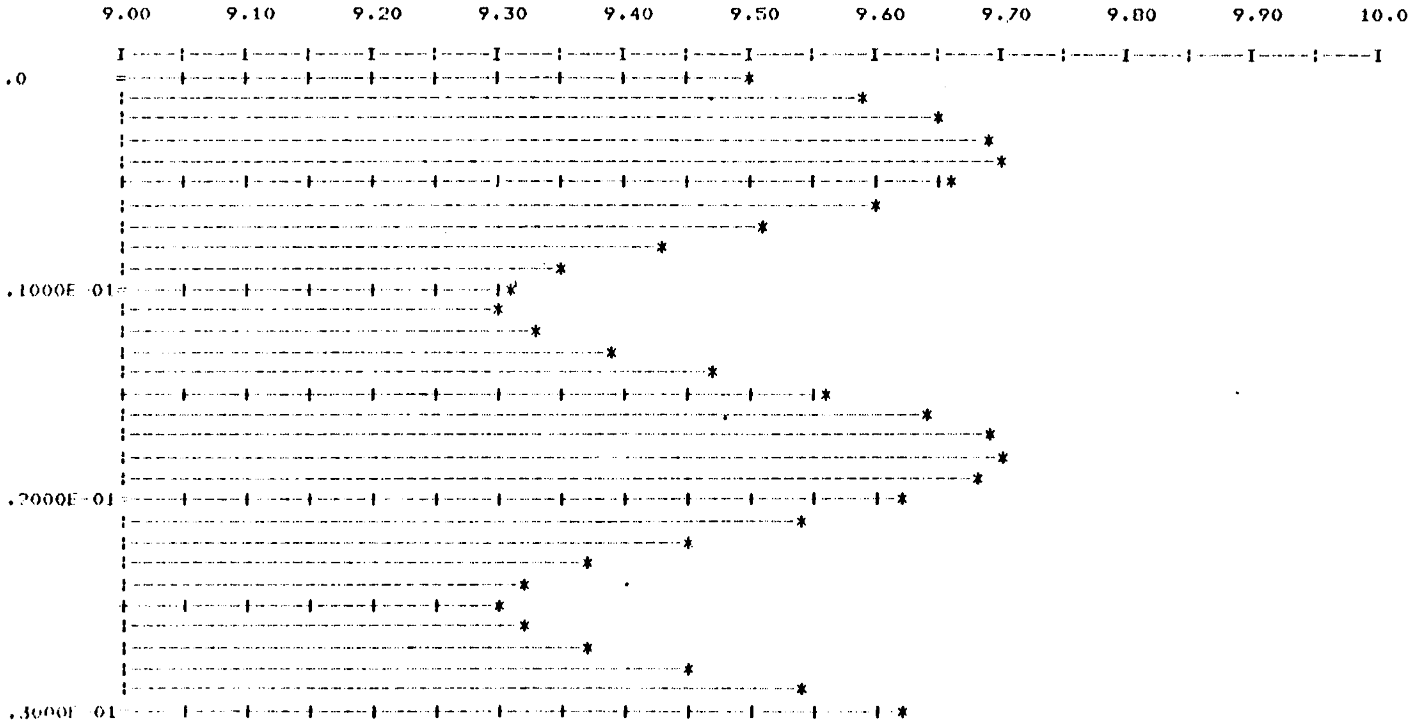
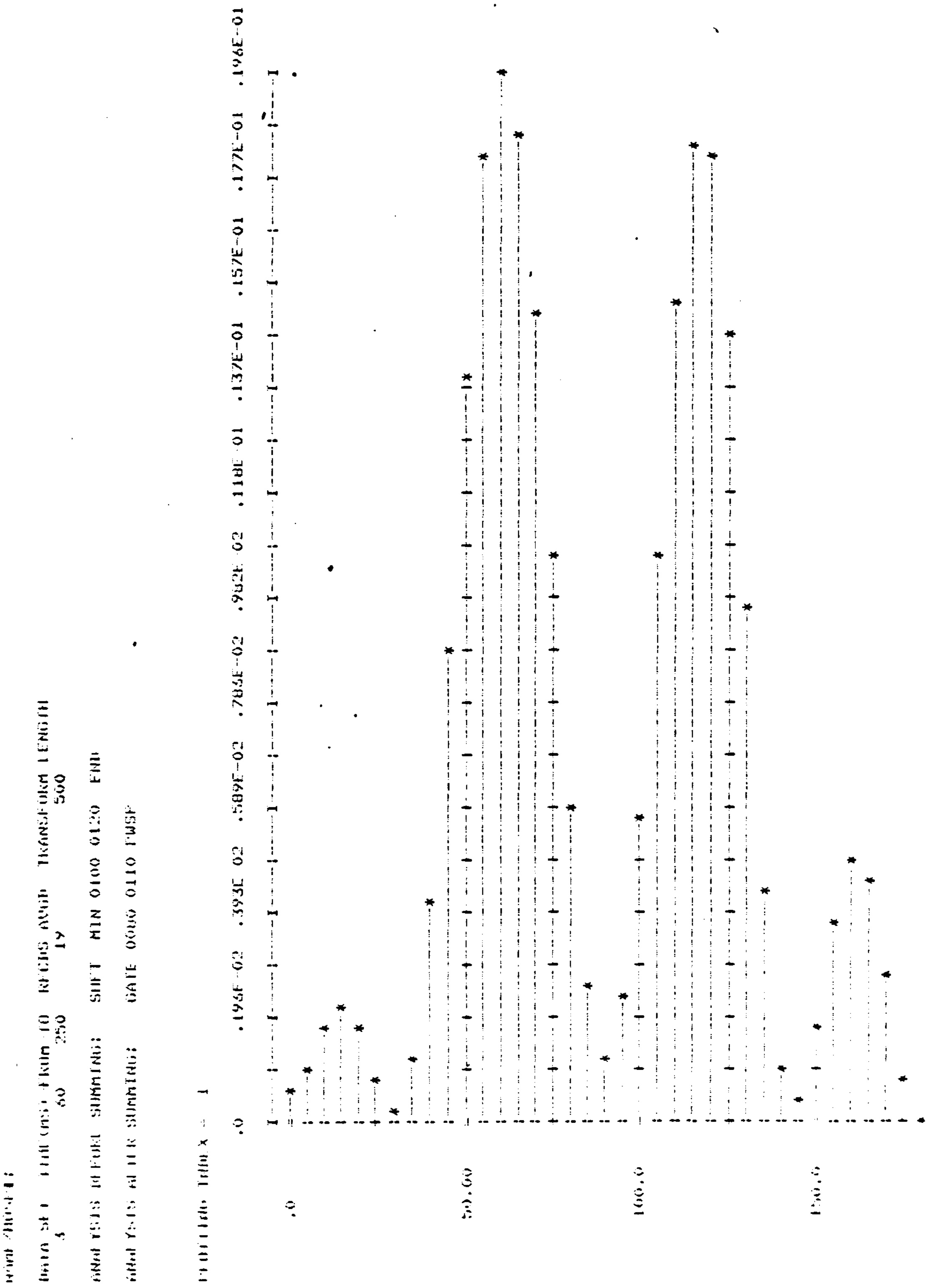


Fig. 5-12 Deterministc Data Plot

generation constants will be given for the last execution of the TEST command only. Figure 5-13 shows a plot of patient data. The plot header gives the patient header, the parameters used for inputing the data, the analyses performed before and after summing, and the plot index. The "begin time" from the header must be added to the time scale values to find the actual cardiocycle times.

5.2.12 INPUT-OUTPUT

The INPUT-OUTPUT (IO) command allows the operator to store (Output) part of the X buffer in a disc file, or retrieve (Input) a previously written file to the Y buffer. Figure 5-11 shows the interaction required. The first item requested is the file number which can be any number from 30 to 99 inclusive. (Lower numbers are reserved for patient data files and computer peripherals.) If this command is given by mistake, it can be aborted by entering a file number of zero. The program then requests whether input or output is to be performed. The user replies as desired with "i" and "o" being the minimum abbreviations. This command can also be aborted at this point by entering an "a". On output, the first 327 data points of the real X array are stored. If the length of data is greater than this, a warning is given as shown. The sequence of analyses performed on the data before it is output is stored with the



```
DATA SET  TIME(MS) FROM TO  RECDS AVGD  TRANSFORM LENGTH
    3               60   250     19              500

ANALYSIS BEFORE SUMMING:   SHFT  MIN 0100 0120  END

ANALYSIS AFTER SUMMING:    GATE 0080 0110 FWSF


PLOTTING INDEX =  1
```

Fig. 5-13 Patient Data Plot

data. Thus, the program keeps a complete list of all analyses done on the data. On input, the data is returned to the Y buffer. From there it can be moved to the X buffer with the Transfer function of the ANALYZE command for further analysis.

5.2.13 STOP

The STOP command causes an execution of the FORTRAN Stop statement, terminating execution of the program.

5.3 ALIGNED AVERAGING

An aligned average is obtained by a two pass procedure as follows. First, the input parameter list is set up (CHANGE-INPUT command) to perform the Synch analysis on input (over the desired search interval) and the RUN command is given. Then the input parameter list is set ip for the Shift analysis on input and the RUN command is given again.

CHAPTER VI

RESULTS AND CONCLUSIONS

6.1 APPROACH TO THE PROBLEM

Throughout this project, separate analyses were performed on inspiration and expiration data. This was done for two reasons. First, it would allow comparisons to be made between inspiration and expiration data, keeping in mind the physiological fact that events associated with left heart systole are not significantly affected by respiration. Therefore, a large difference in results between inspiration and expiration would indicate that that event was probably not occurring in the left heart. Secondly, two separate data sets would provide greater statistical confidence.

The first step was to determine the onset time (To) of the ejection click. Initial observations of time series data seemed to indicate that the ejection click was of relatively high frequency and short duration. In several patients this high frequency wavelet was much larger in amplitude than any other signal feature, with its onset time being easy to determine by visual inspection of the time series record. In other patients the beginning of the click had to be determined by considering the time series, the average envelogram of the time series, and the carotid pulse together. This determination of the onset time was done by looking for a high frequency wavelet in the time record which coincided with a significant peak on the average envelogram and also occurred approximately 20 ms before the upward rise of the carotid pulse.

Once the onset time was determined, an aligned average time record was obtained by performing the alignment on a selected local maximum or minimum which occurred approximately 5 to 15 ms after the beginning of the click. The aligned average records for inspiration and expiration were then compared to determine the time difference between the occurrence of the click in each. This correlation was made by overlaying the inspiration and expiration printouts and visually aligning them to obtain the best fit during approximately the first 20 ms of the click.

Power spectra of the click were then made from the aligned average records after gating out the following intervals:

1. To through To + 20 ms
2. To + 20 ms through To + 40 ms
3. To through To + 30 ms
4. To through To + 40 ms

Intervals 1 and 2 were used to investigate the frequency characteristics of the click versus time, while intervals 3 and 4 were used to obtain overall power spectra for longer periods of time.

An aligned average was then made on a point within S1, and power spectra were obtained over the following intervals:

1. To - 50 ms through To
2. To - 50 ms through To - 20 ms
3. To - 30 ms through To

In several cases the above intervals were shifted by an amount no greater than approximately 5 ms.

6.2 AN EXAMPLE PATIENT

The following plots show the complete procedure given above for patient number 7. Figures 6-1 and 6-2 are plots of a single time series cardiocycle for inspiration and expiration respectively. Aligned average records are given in Figs. 6-3 (inspiration) and 6-4 (expiration). The peak used for alignment is indicated by a circle. Figures 6-5 through 6-12 show the power spectra for all four window intervals and both respiration phases as indicated.

The aligned average records done for S1 on inspiration and expiration are shown in Figs. 6-13 and 6-14 respectively. Again, the points used for alignment are indicated by a circle. Power spectra of S1 for all three window intervals and both respiration phases are given in Figs. 6-15 through 6-20 as indicated.

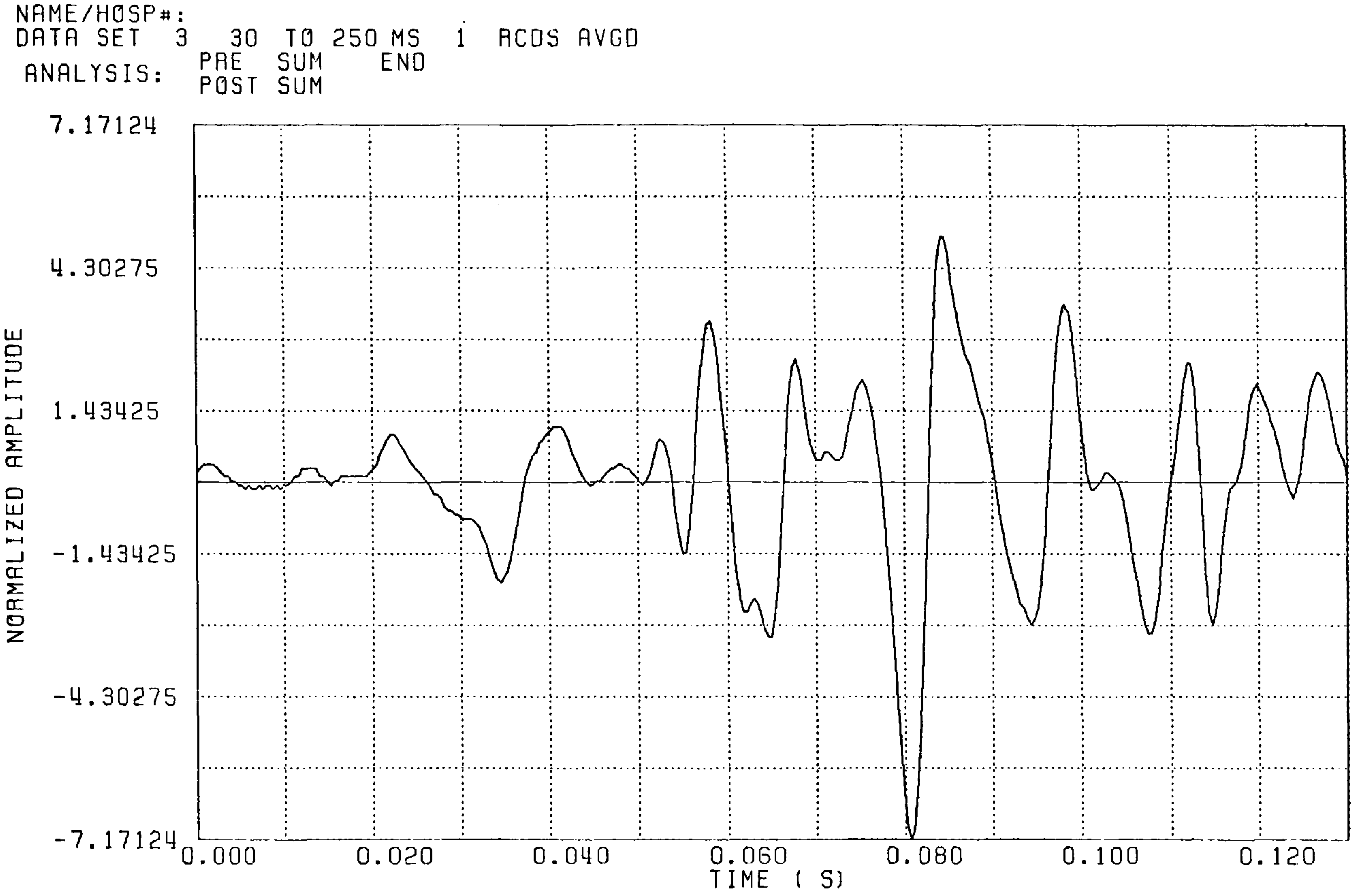


Fig. 6-1 Phonocardiogram, Patient 7, Inspiration

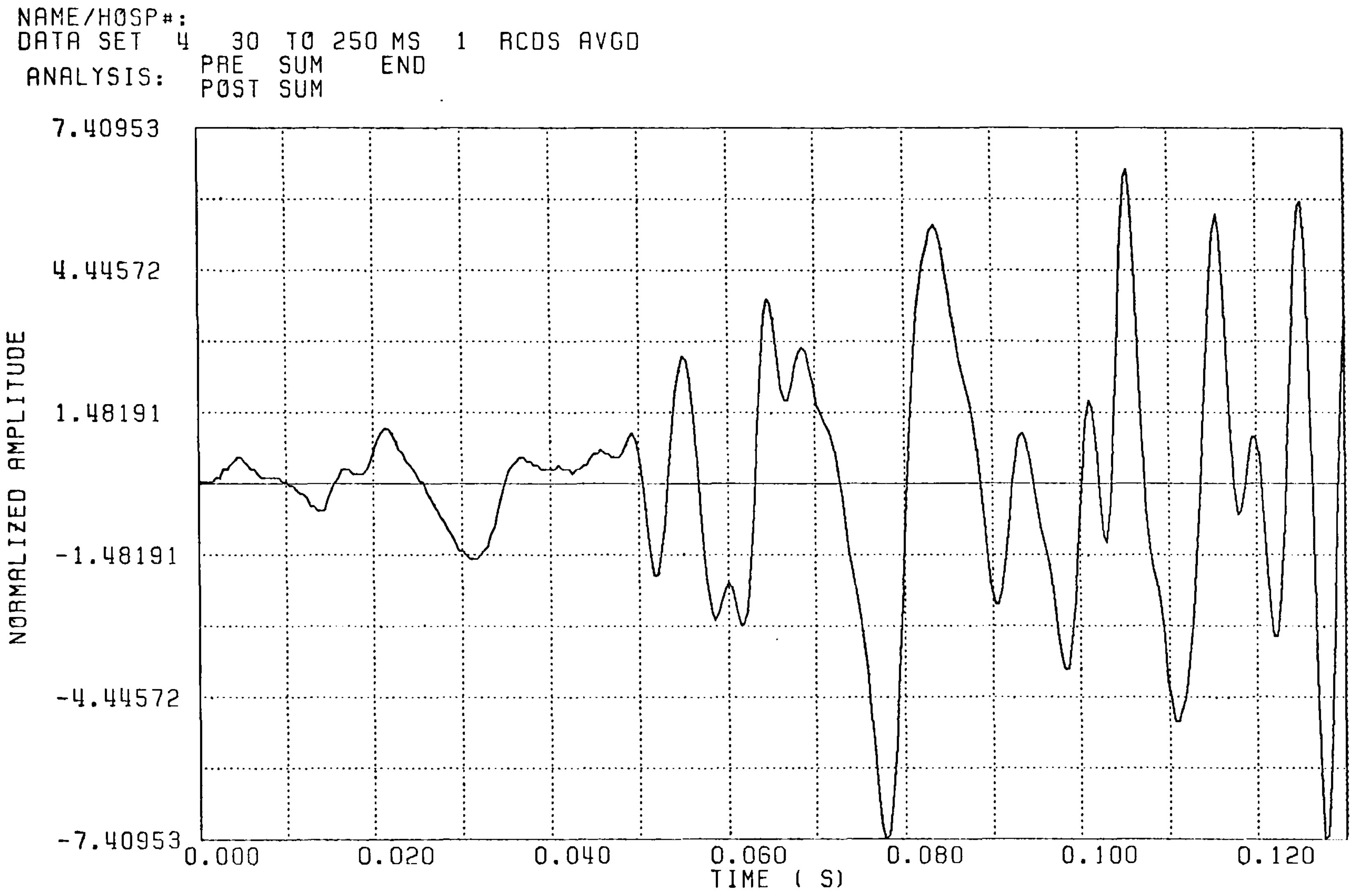


Fig. 6-2 Phonocardiogram, Patient 7, Expiration

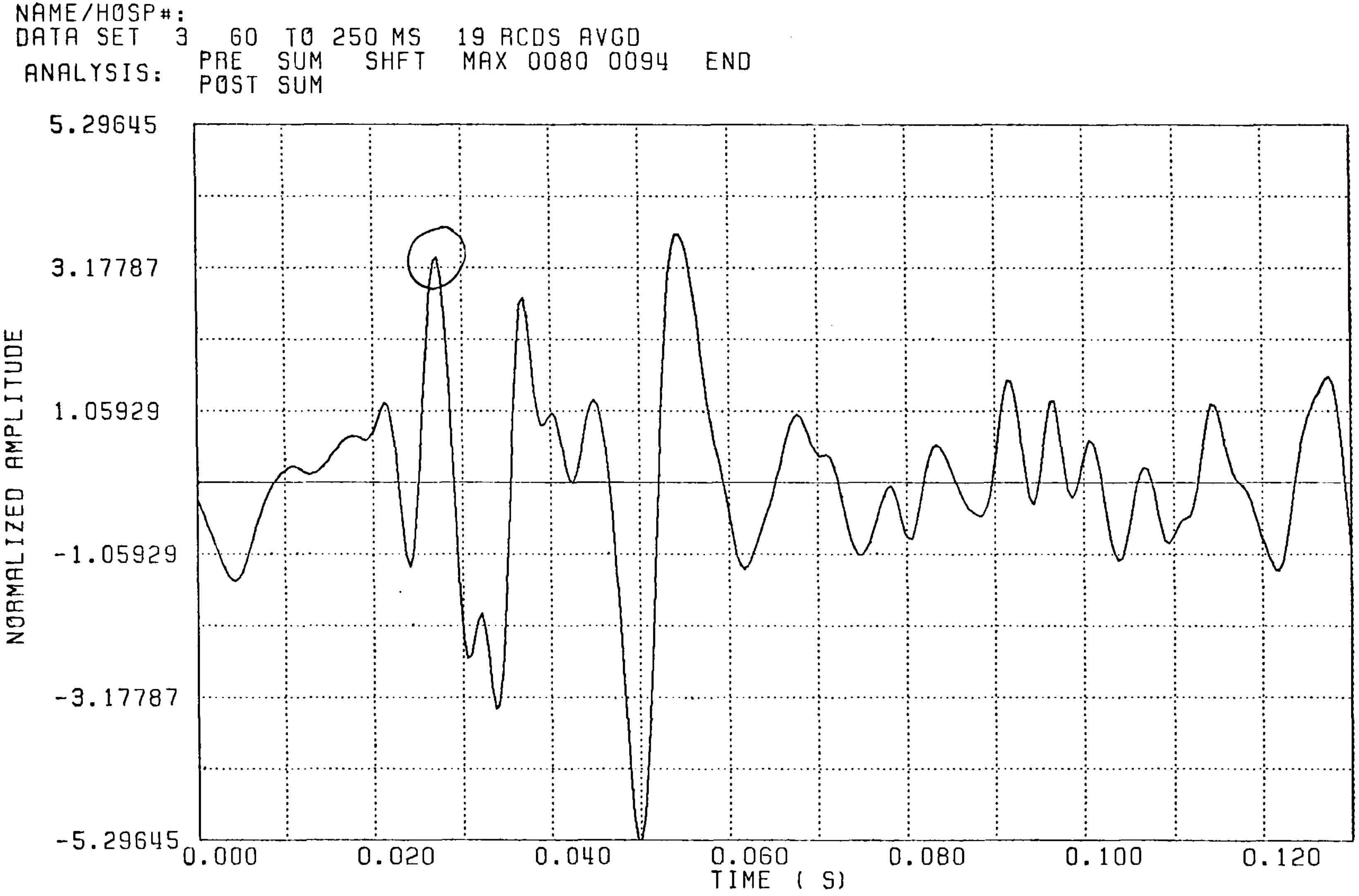


Fig. 6-3 Ejection Click Aligned Average, Patient 7, Inspiration

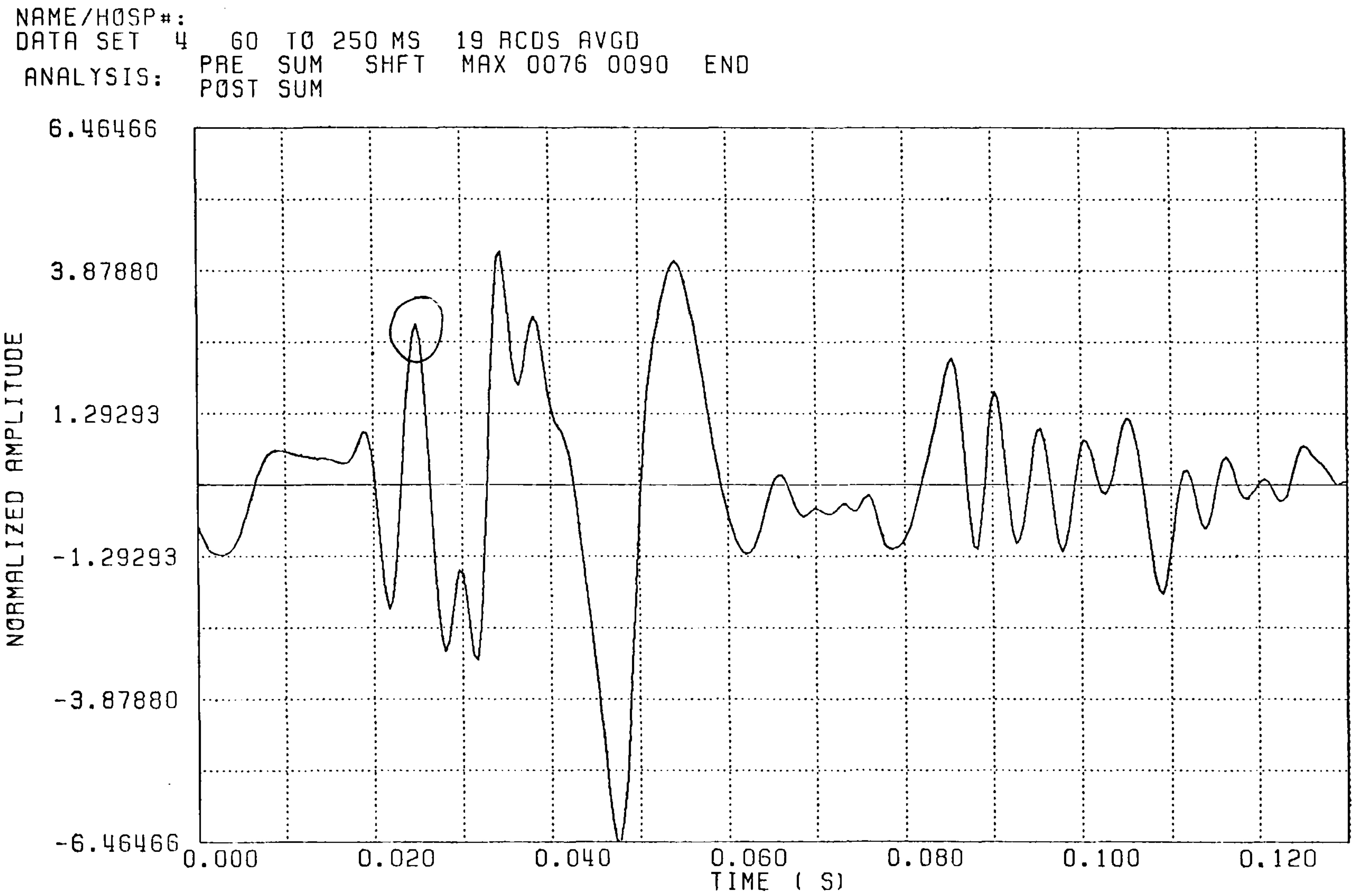


Fig. 6-4 Ejection Click Aligned Average, Patient 7, Expiration

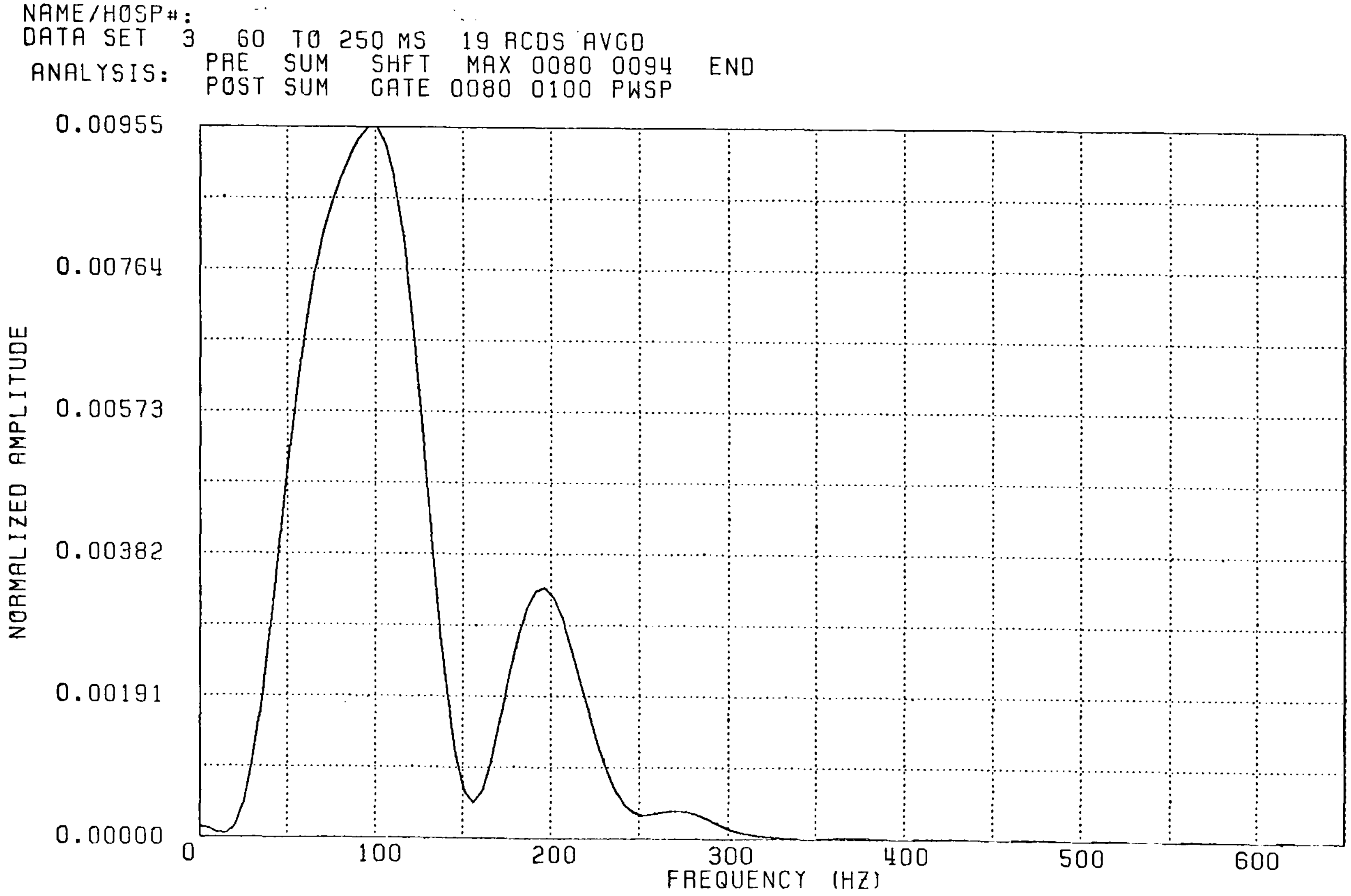


Fig. 6-5 Early 20 ms Power Spectrum, Patient 7, Inspiration

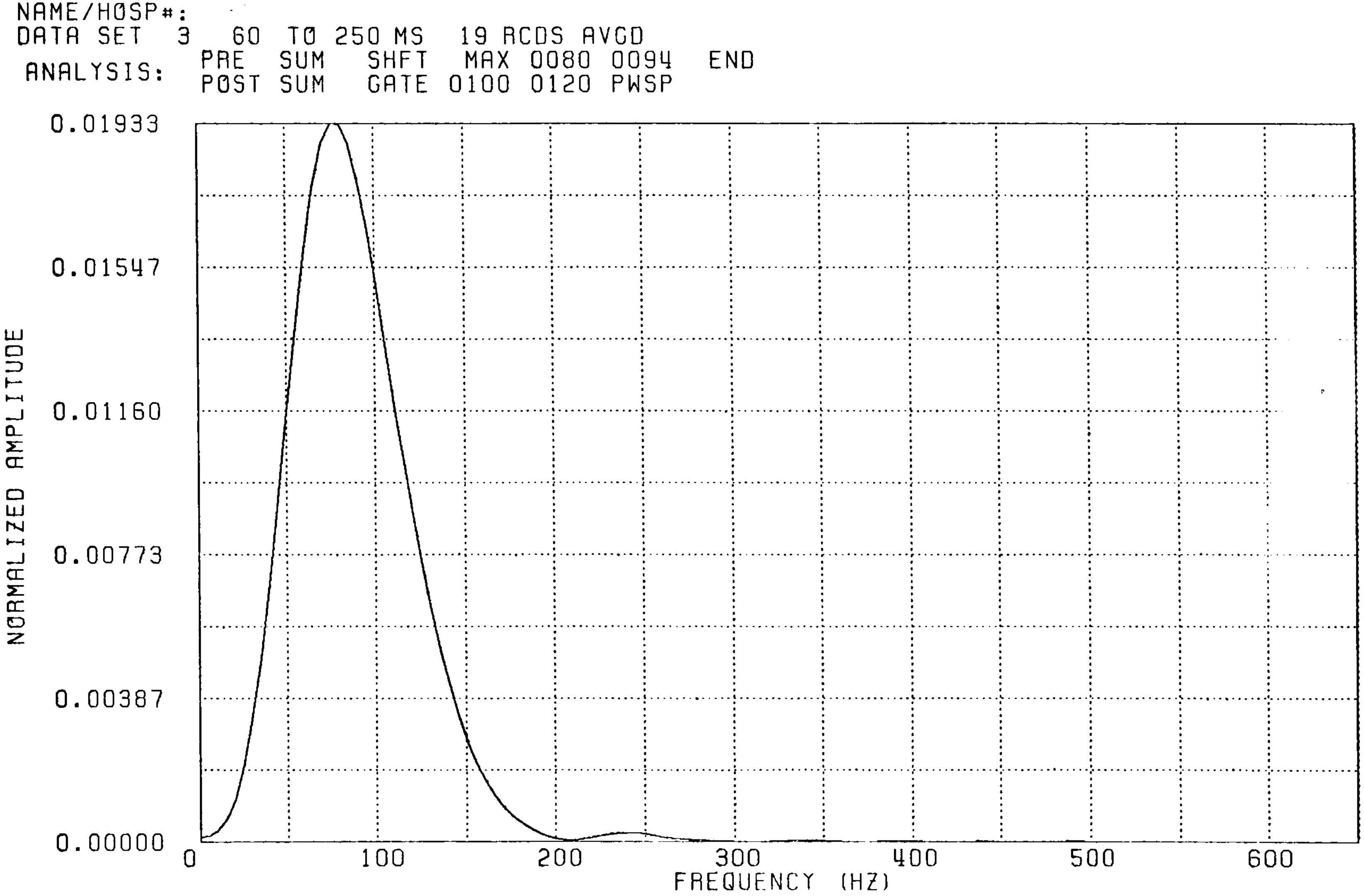


Fig. 6-6 Late 20 ms Power Spectrum, Patient 7, Inspiration

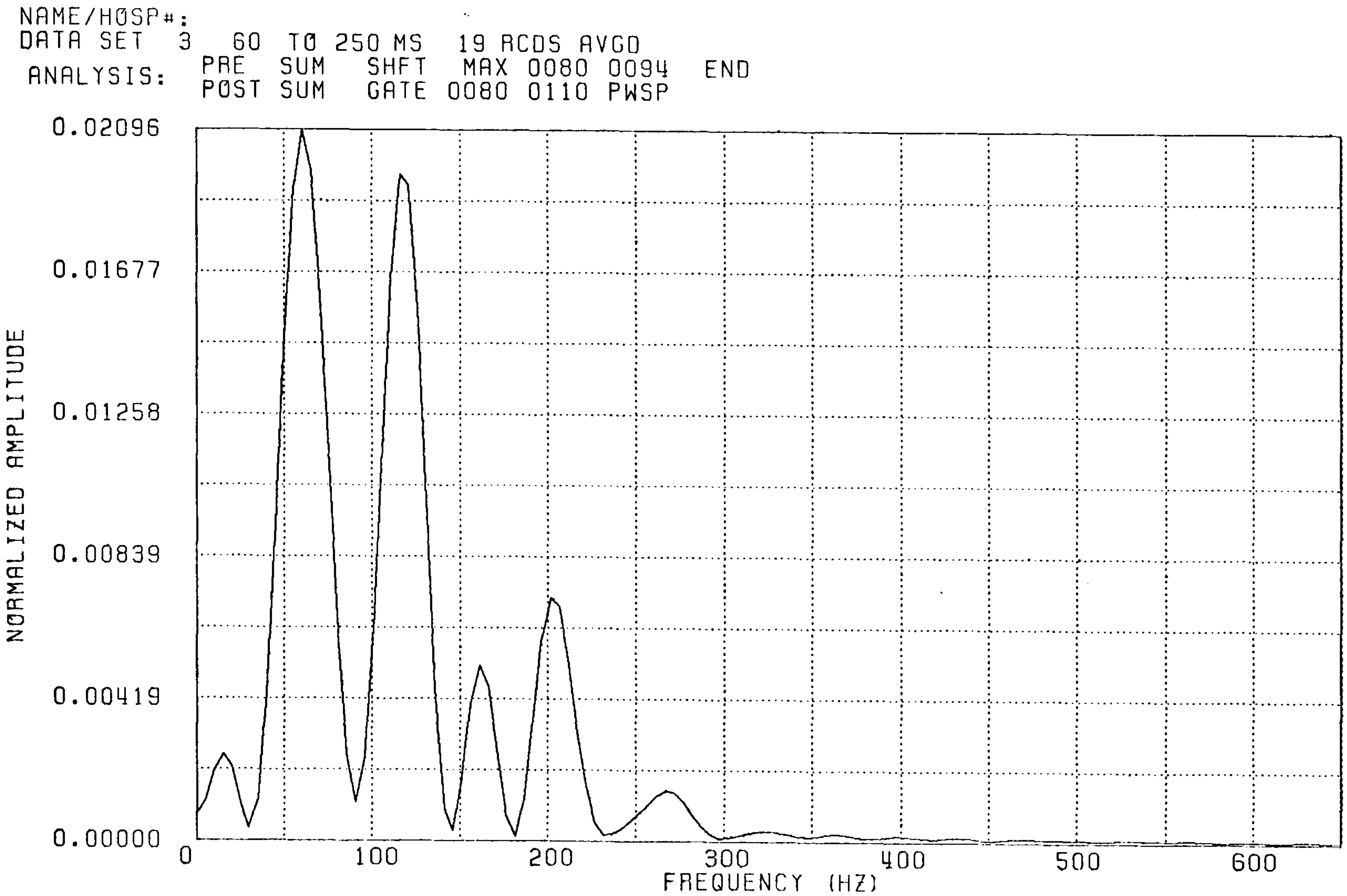


Fig. 6-7 30 ms Click Power Spectrum, Patient 7, Inspiration

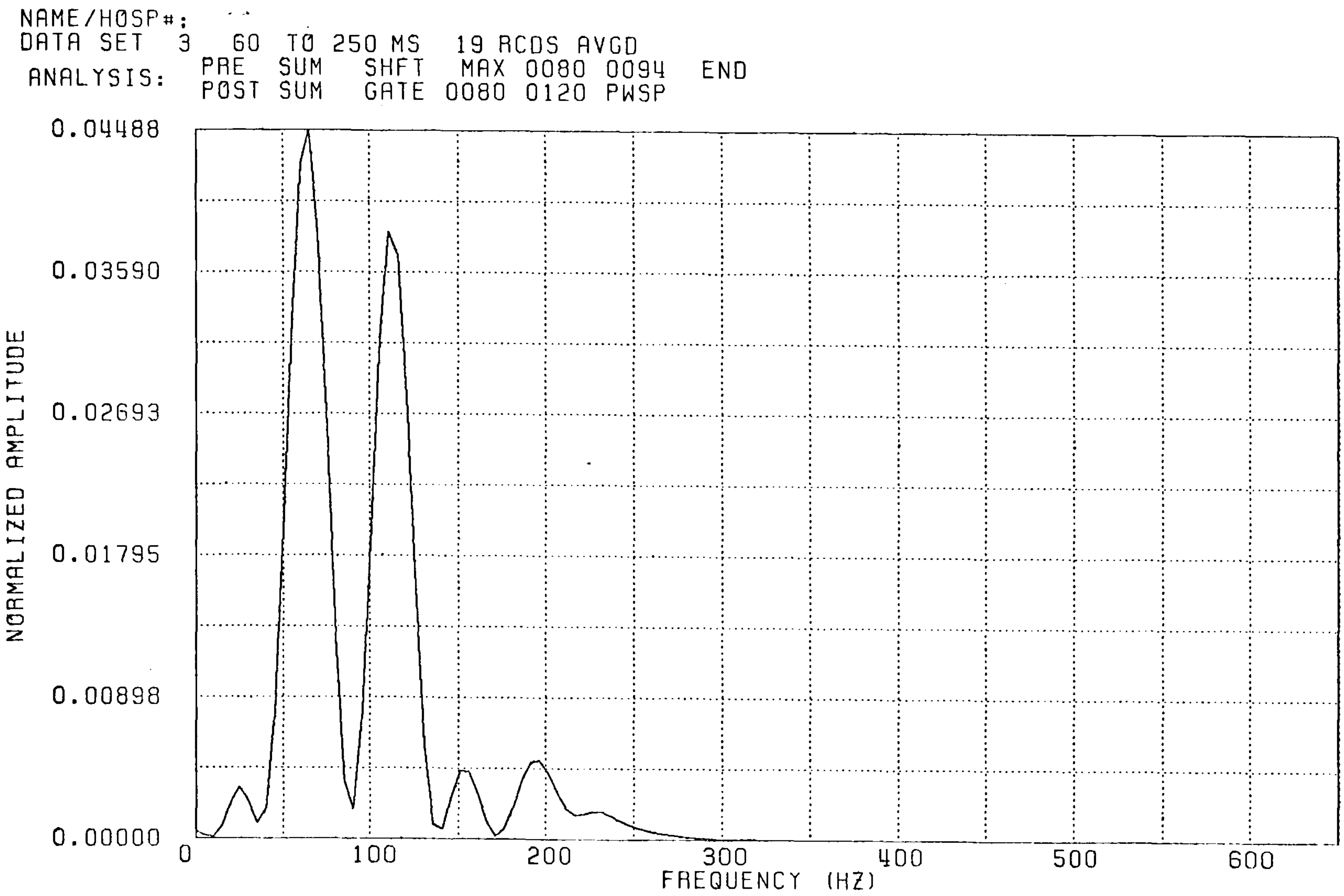


Fig. 6-8 40 ms **Click** Power Spectrum, Patient 7, Inspiration

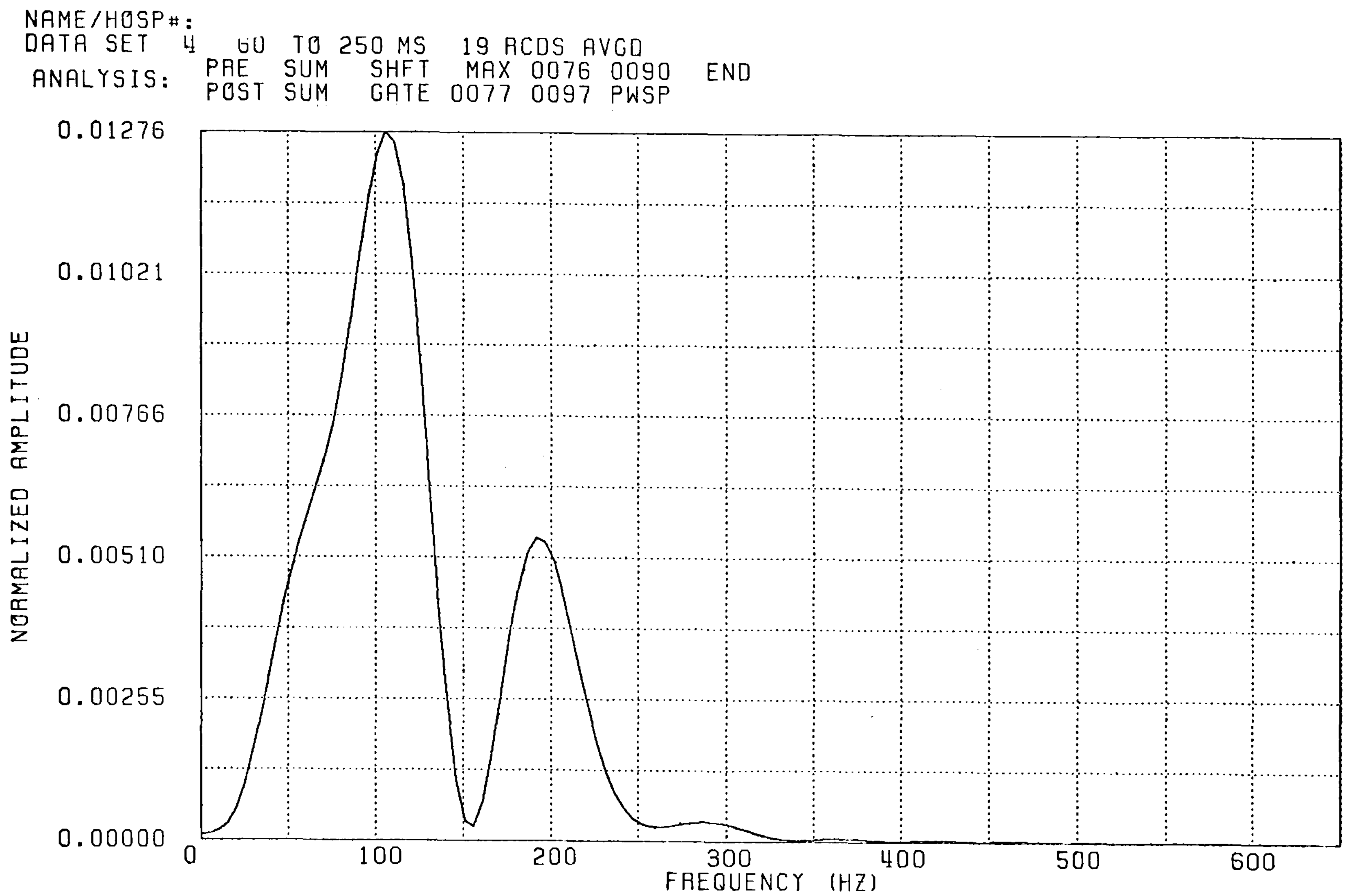


Fig. 6-9 Early 20 ms Power Spectrum, Patient 7, Expiration

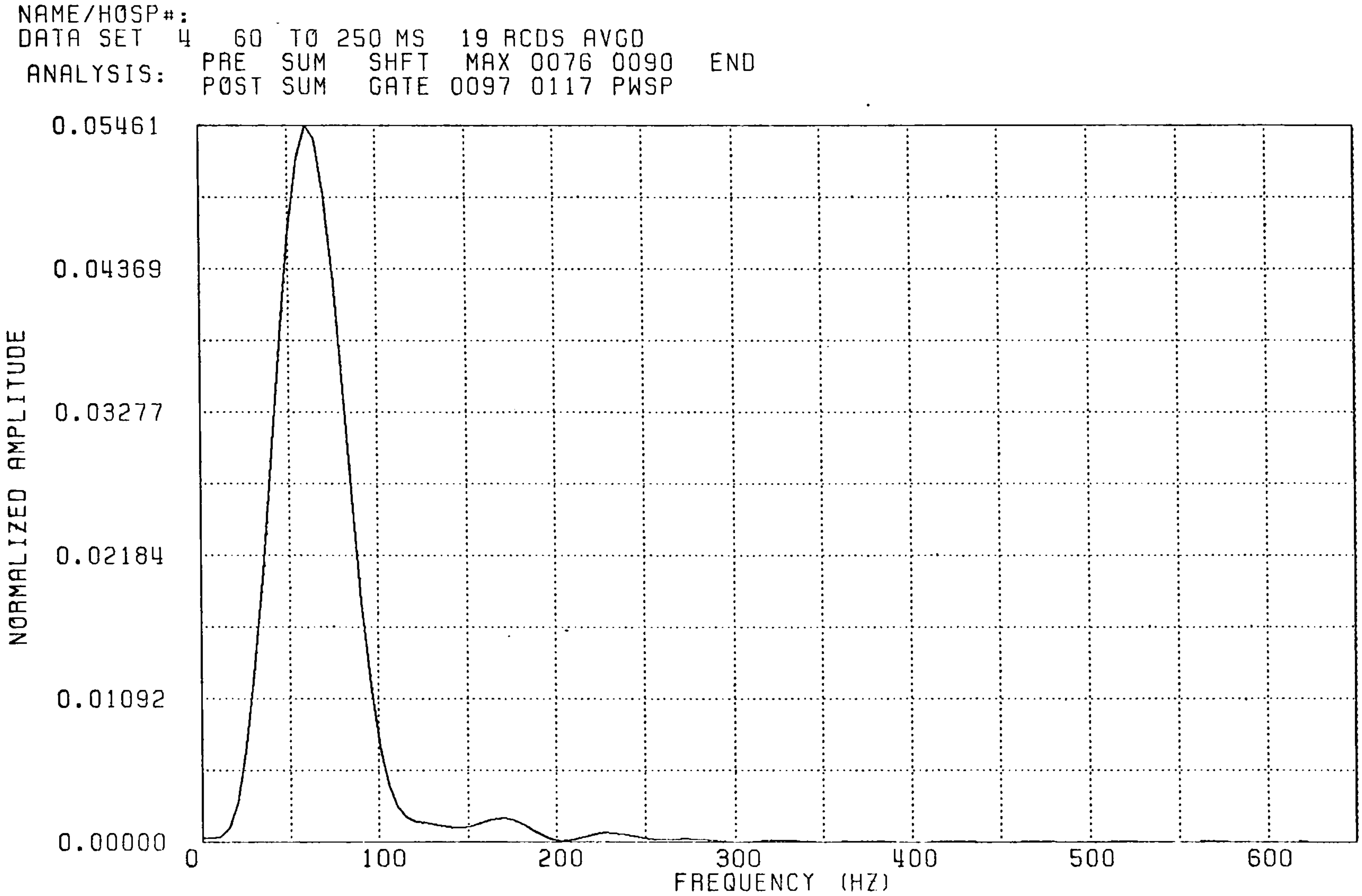


Fig. 6-10 Late 20 ms Power Spectrum, Patient 7, Expiration

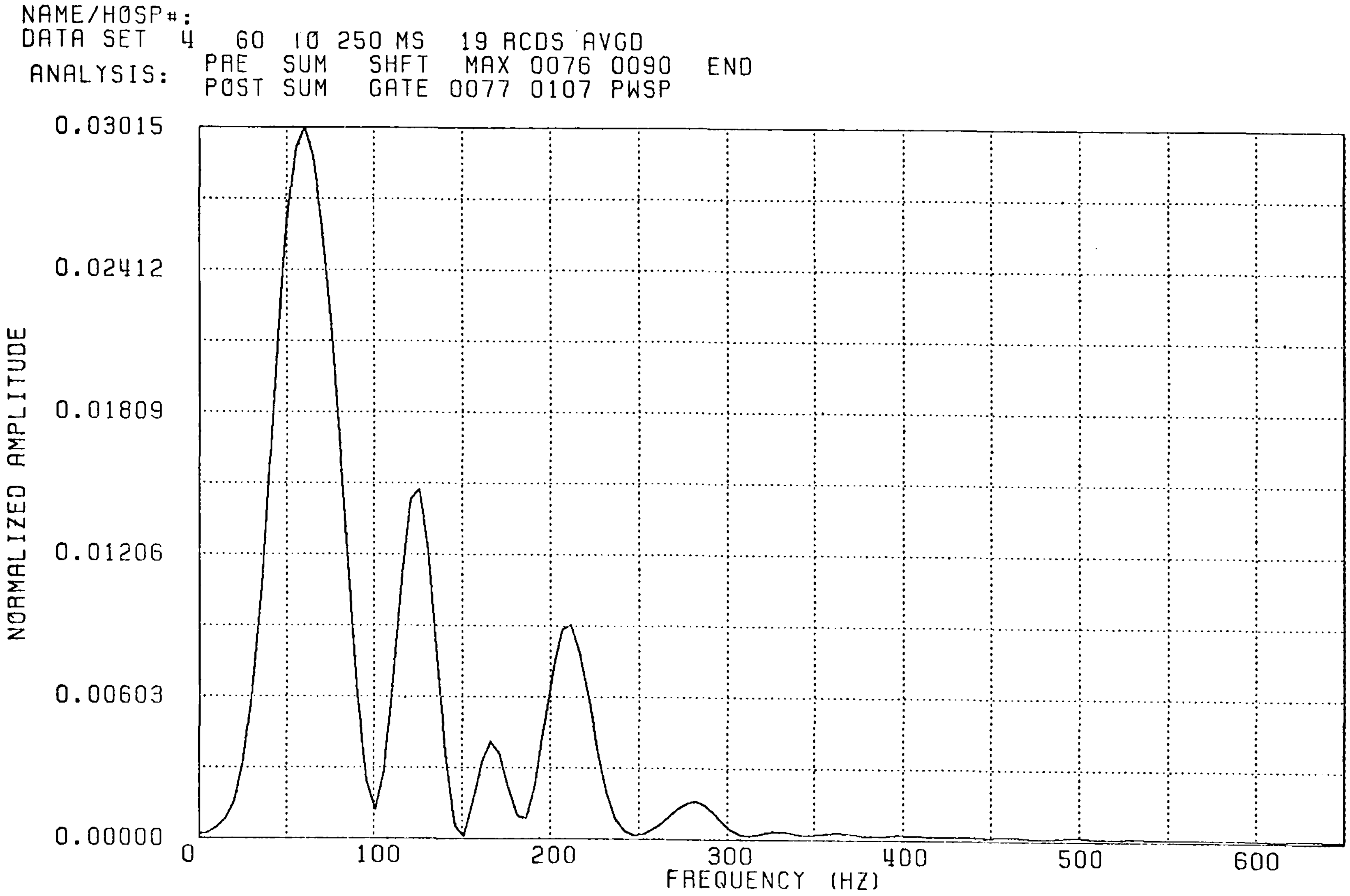


Fig. 6-11 30 ms Click Power Spectrum, Patient 7, Expiration

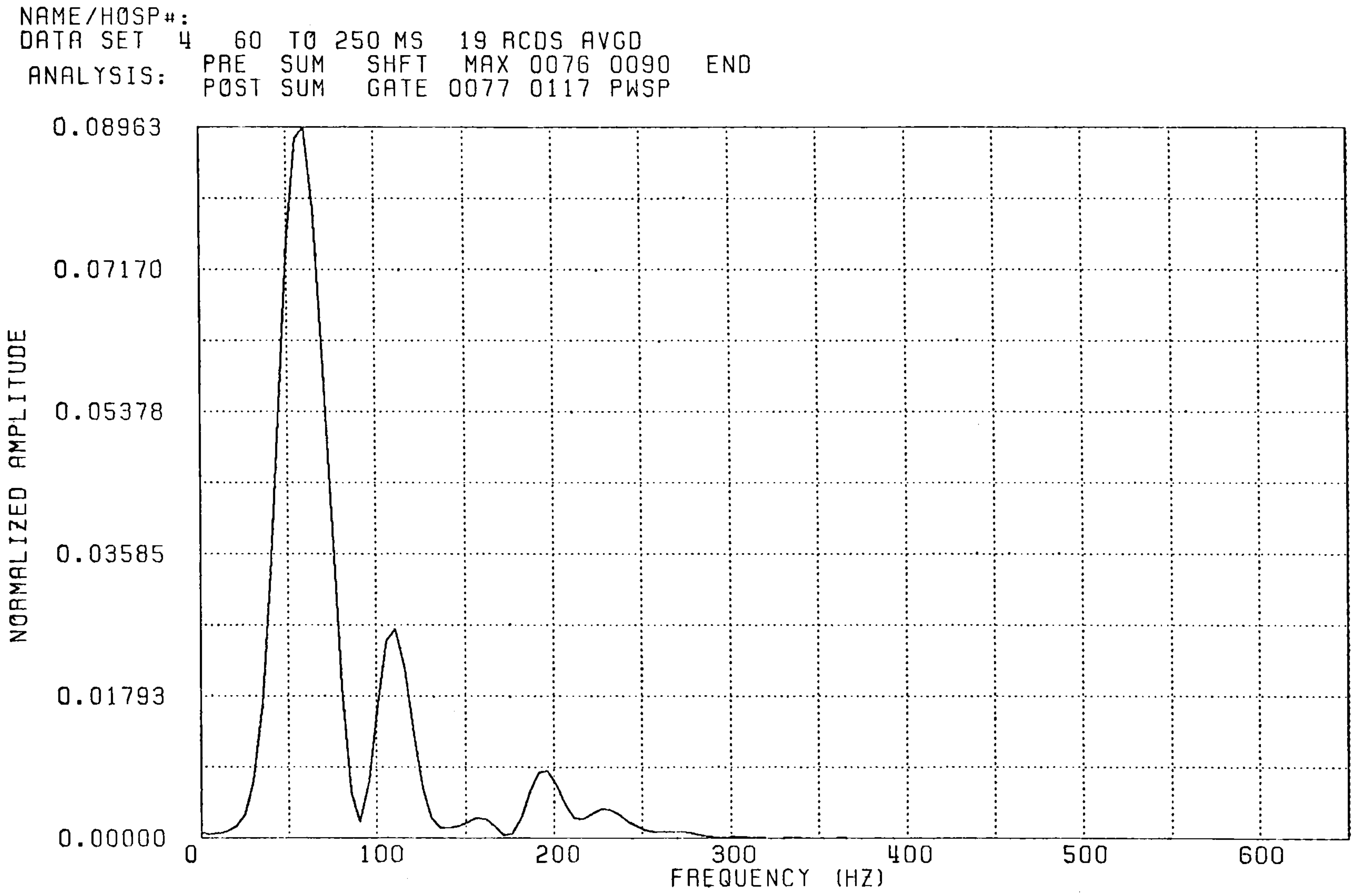


Fig. 6-12 40 ms Click Power Spectrum, Patient 7, Expiration

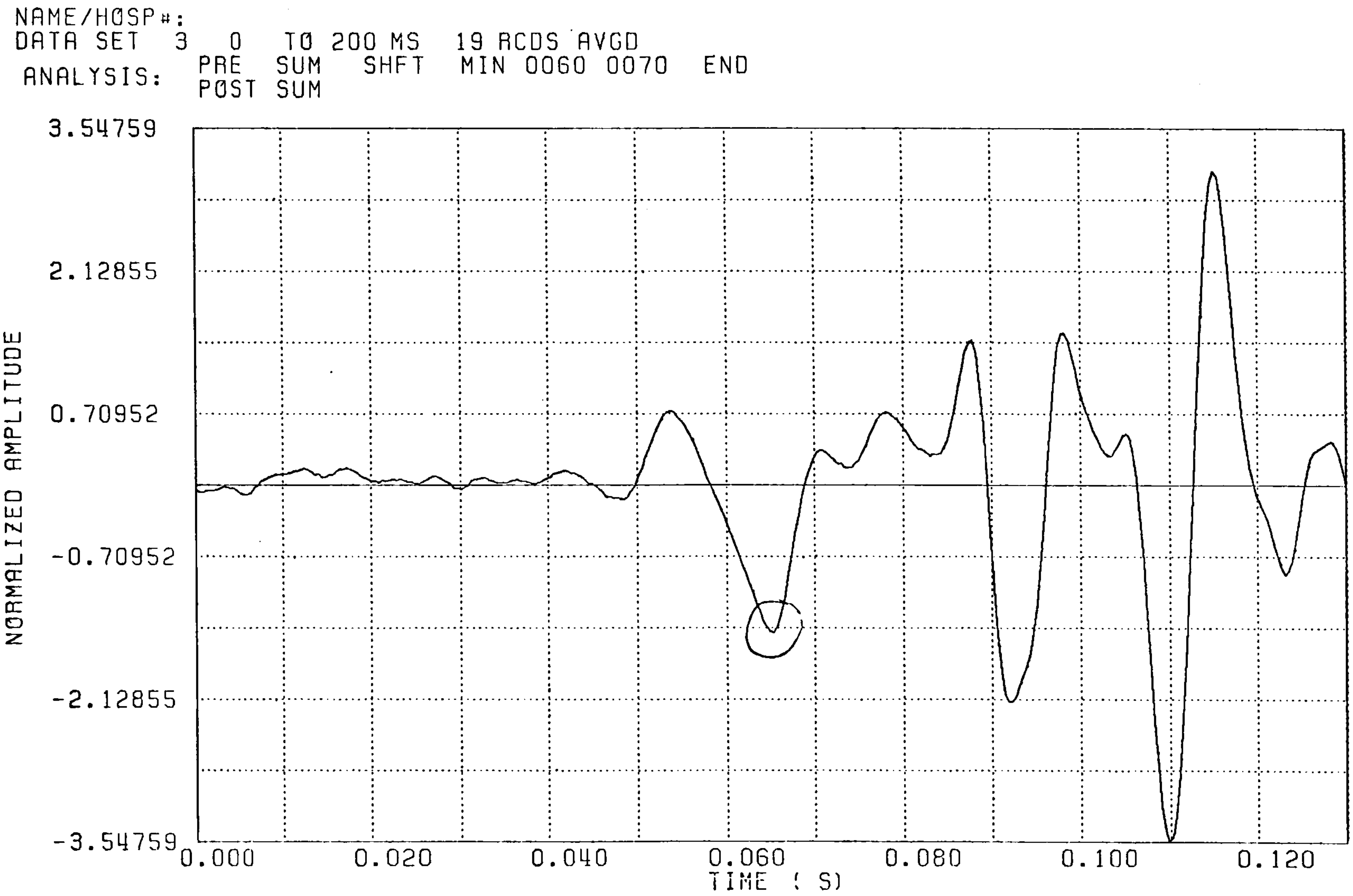


Fig. 6-13 S1 Aligned Average, Patient 7, Inspiration

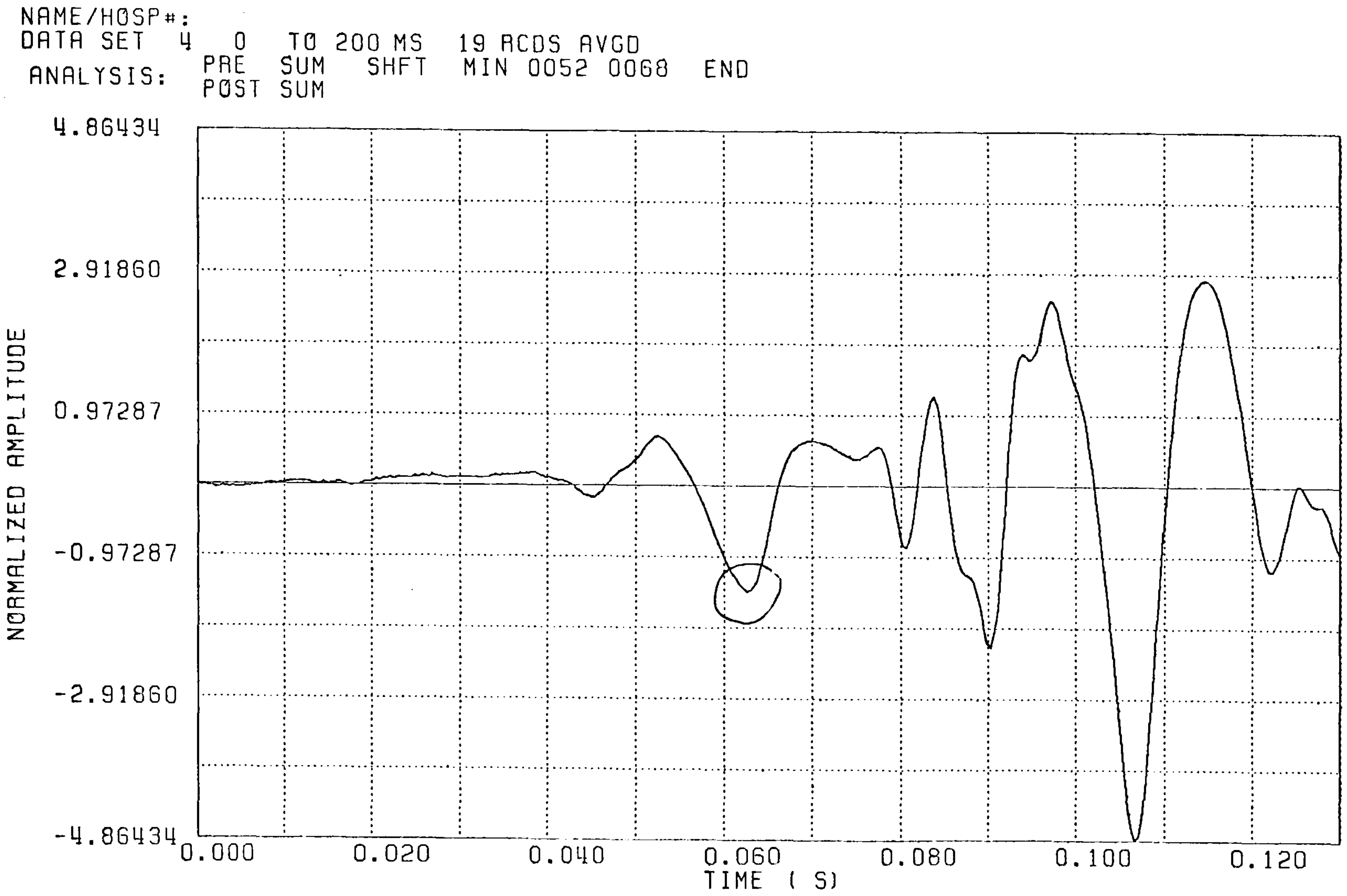


Fig. 6-14 S1 Aligned Average, Patient 7, Expiration

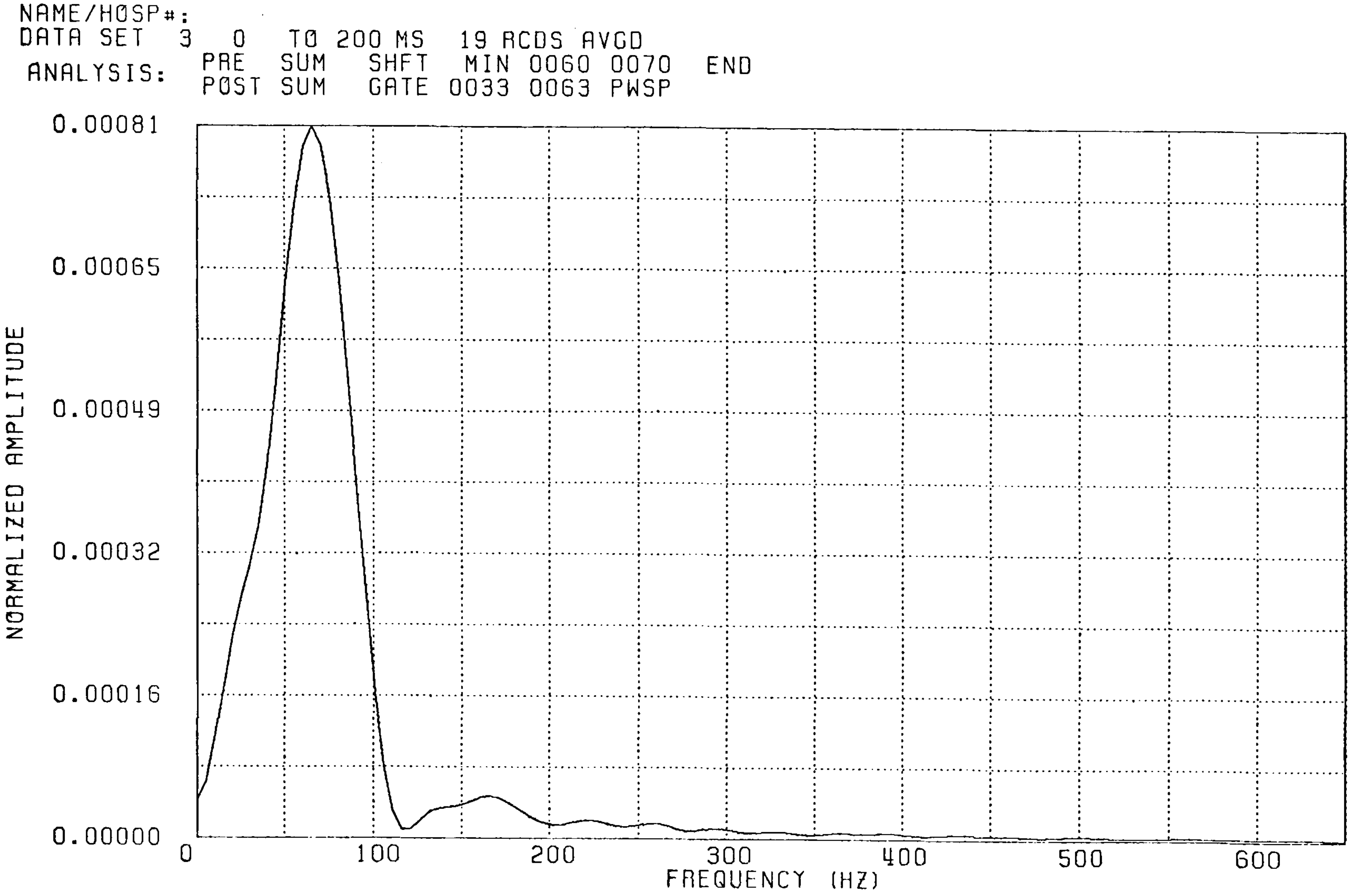


Fig. 6-15 Early 30 ms S1 Power Spectrum, Patient 7, Inspiration

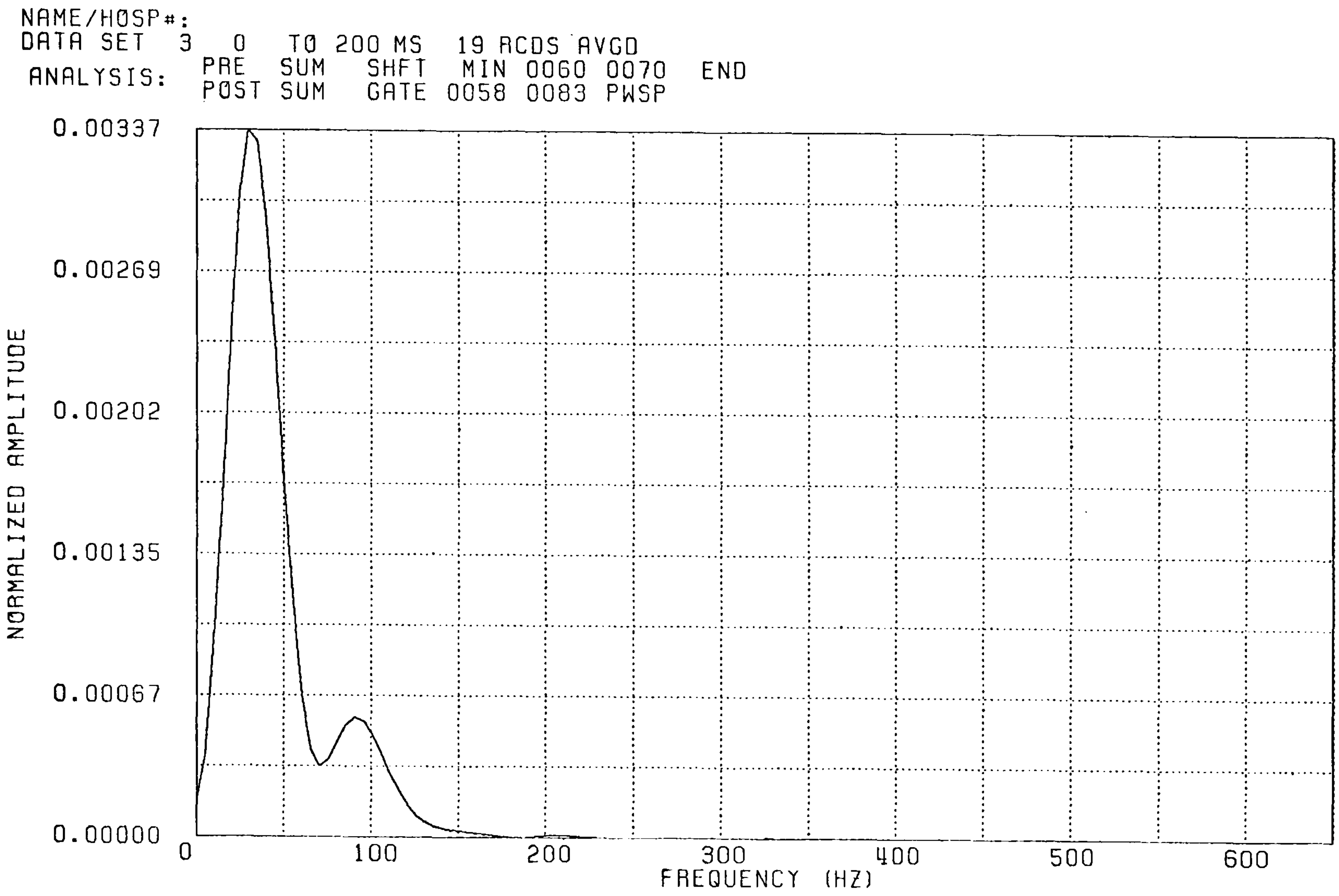


Fig. 6-16 Late 30 ms S1 Power Spectrum, Patient 7, Inspiration

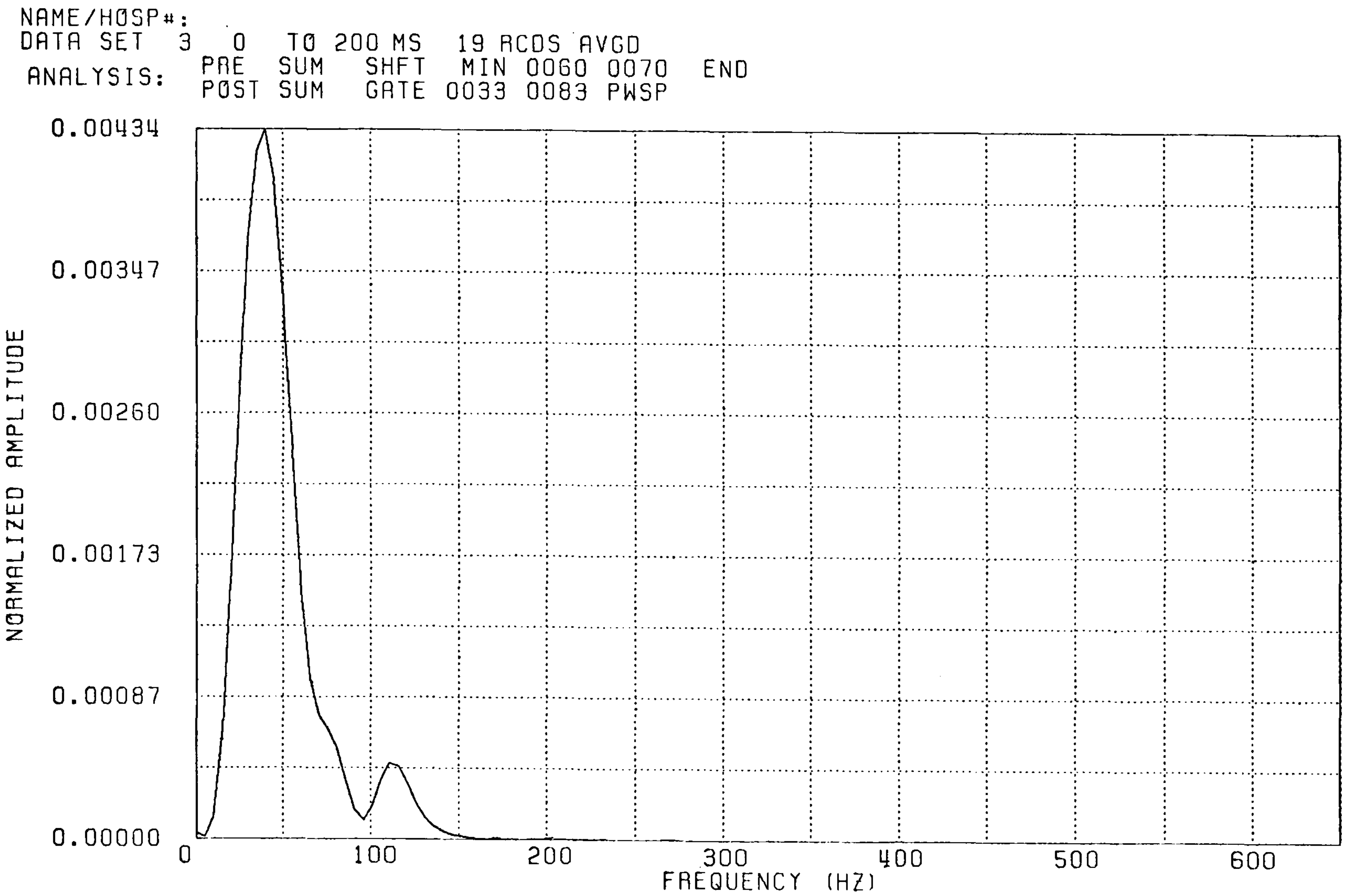


Fig. 6-17 50 ms S1 Power Spectrum, Patient 7, Inspiration

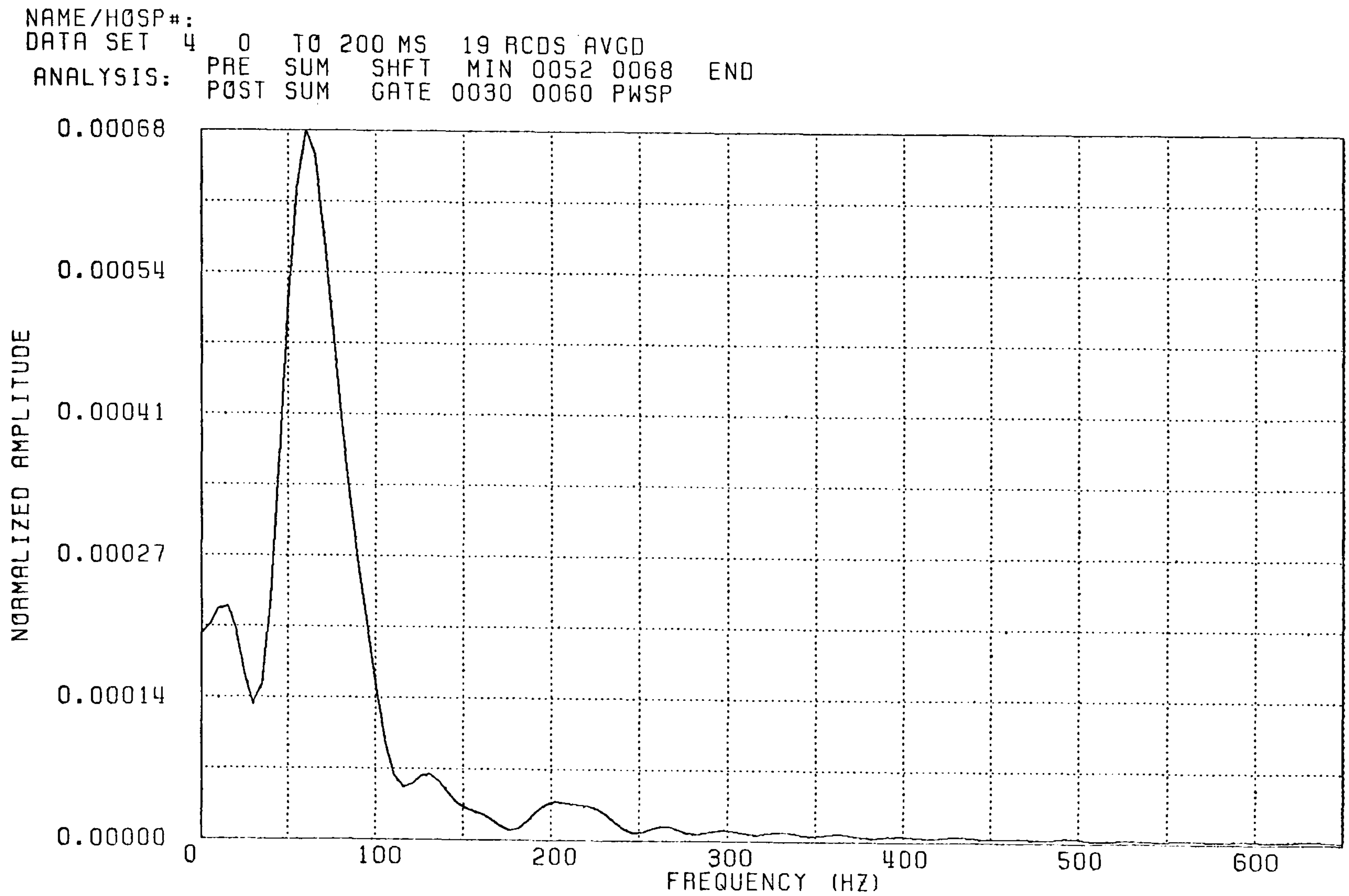


Fig. 6-18 Early 30 ms S1 Power Spectrum, Patient 7, Expiration

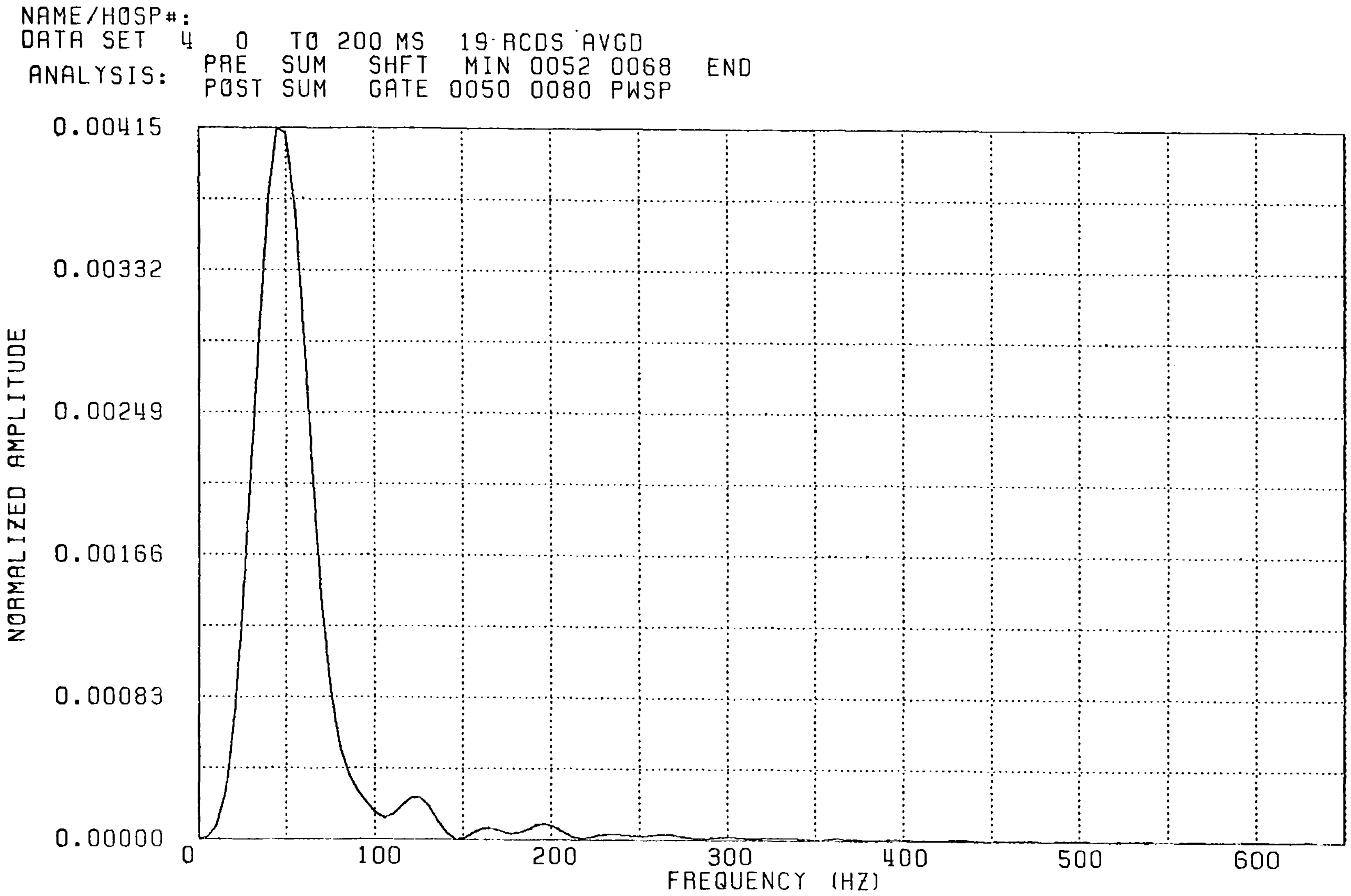


Fig. 6-19 Late 30 ms S1 Power Spectrum, Patient 7, Expiration

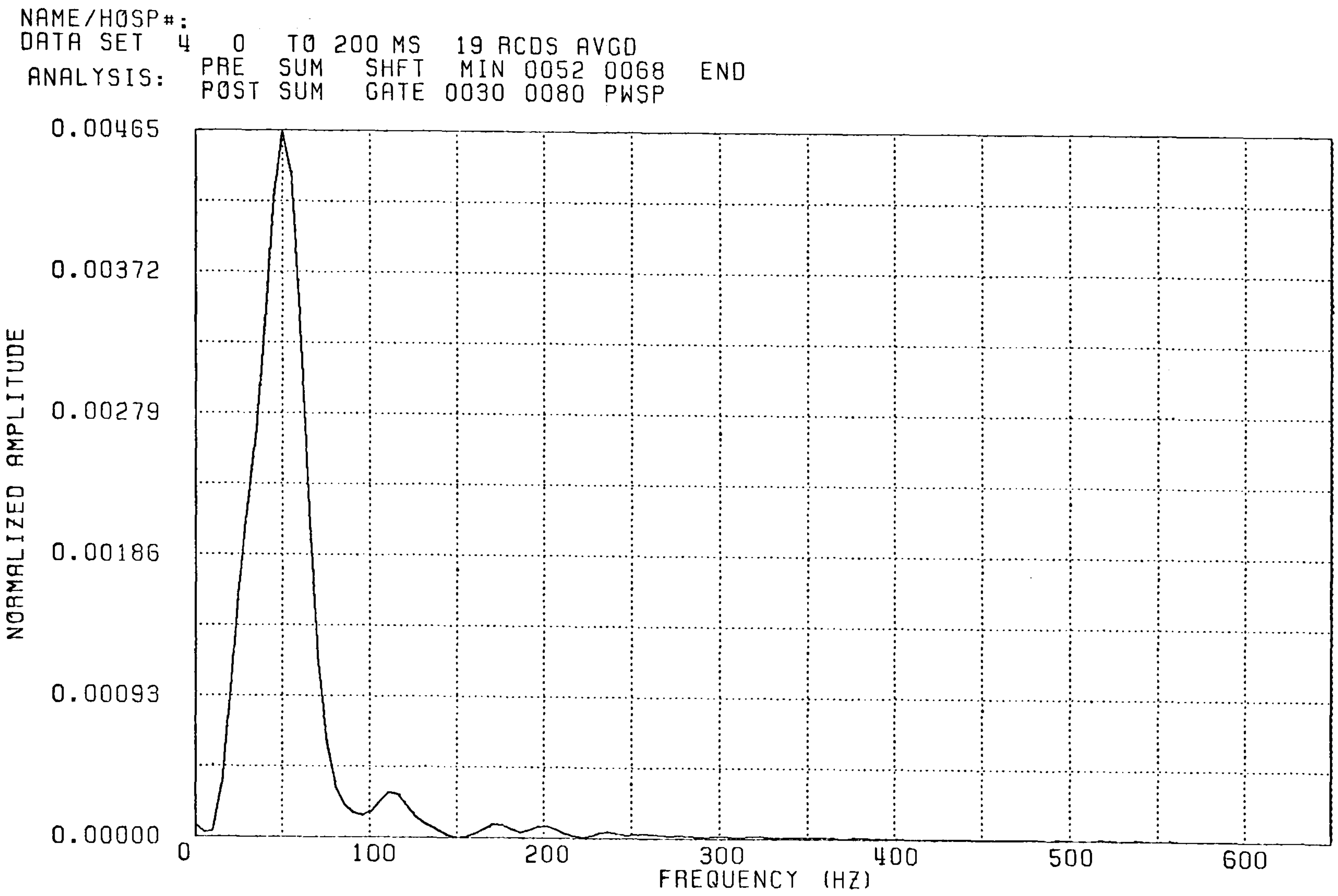


Fig. 6-20 50 ms S1 Power Spectrum, Patient 7, Expiration

6.3 RESULTS

6.3.1 CHANGE IN TIME OF OCCURRENCE OF THE CLICK WITH RESPIRATION

The difference between the time of the occurrence of the click in inspiration and expiration, Td, is defined to be positive if the click occurs earlier during expiration. Table 6-1 lists Td for all aortic stenosis patients and gives the number of records averaged for both stenosed and normal patients. For the patients analyzed, Td was always positive with a mean of 5 ms.

6.3.2 POWER SPECTRA OF THE CLICK

Power spectra for window interval 3 of the ejection click are summarized in Table 6-2 (inspiration) and Table 6-3 (expiration). Data for the major peaks are listed in the following form: frequency/normalized amplitude, where each frequency is in units of Hz and each amplitude is times 10^{-3}.

6.3.3 POWER SPECTRA OF THE CLICK VERSUS TIME

Table 6-4 (inspiration) and 6-5 (expiration) show the principal low and high frequency peaks occurring during the first 20 ms of the click and during the following 20 ms. The following considerations were used in taking these data

TABLE 6-1

Td AND NUMBER OF RECORDS AVERAGED

Patient	Td(ms.)	Inspiration	Expiration
1	10	12	13
2	5	19	24
3	2	20	20
4	7	16	20
5	6	20	19
6	6	21	16
7	3	19	19
8	4	12	14
9	2	15	14
10	4	25	17
11	7	20	25
12	8	12	24
13	-	14	20
14		14	16
15		14	19
16		16	17
17		11	8
18		17	12
19		14	13

TABLE 6-2

30ms. CLICK POWER SPECTRA, INSPIRATION

Patient	**Frequency/Amplitude X 10^{-3}**				
1	50/3.6	100/10		155/18	
2		75/9.8	150/11	205/19	
3	50/20	95/120		210/52	
4	25/19	85/15		200/13	
5	30/2.4	125/1.0		215/8.4	
6	35/3.3	85/12	135/4.5	210/3.9	
7	60/26	110/24	155/4.2	195/5.2	
8	40/1.3	85/6.6		180/1.7	
9	35/11	100/6.3	155/6.9	225/1.9	330/1.1
10	45/3.5	110/3.6		170/2.2	
11	35/7.5	75/11	120/9.9	215/1.3	
12	50/15	100/9.2	140/1.3	195/0.7	
13	45/18	---		---	

TABLE 6-3

30ms. CLICK POWER SPECTRA, EXPIRATION

Patient	Frequency/Amplitude X 10^{-3}				
1	45/4.6	105/5.2		160/33	
2		75/20	145/7.8	210/15	
3	45/30	85/62		215/34	
4	30/21	80/26		195/17	
5	70/5.8	115/3.7		225/12	
6	35/3.9	85/11		210/6.9	
7	55/48	115/21		195/7.2	
8		90/7.7		185/2.5	250/0.8
9	40/12	95/13		175/8.4	220/1.4
10	50/5.4	110/3.7		180/2.1	
11	30/9	75/9.6	120/4.1	200/1.0	
12	45/43	95/9.8		---	
13	45/40	155/2.4		---	

TABLE 6-4

CLICK POWER SPECTRA VERSUS TIME, INSPIRATION

Patient	Early		Late	
1	85/4.4	160/11	65/2.6	---
2	40/4.9	185/11	90/12	205/1.6
3	90/49	215/37	85/78	205/3.9
4	45/6.6	210/10	65/16	---
5	40/2.5	210/8.0	45/0.4	205/0.37
6	95/4.9	220/3.5	75/6.3	240/0.38
7	85/11	205/2.4	85/23	---
8	85/4.8	215/0.86	50/0.74	215/0.63
9	100/6.5	215/1.9	40/7.2	---
10	40/1.1	145/1.1	75/3.3	145/1.2
11	100/4.6	220/0.53	80/7.6	---
12	75/6.0	165/0.7	55/11	---
13	45/11	---	<50/?	---

TABLE 6-5

CLICK POWER SPECTRA VERSUS TIME, EXPIRATION

Patient	Early		Late	
1	65/2.9	165/24	45/2.6	125/1.9
2	70/11	210/12	95/8.3	210/0.8
3	70/8.7	215/28	90/27	---
4	70/9.5	200/15	70/11	190/1.5
5	60/2.9	215/13	55/1.7	200/0.30
6	95/6.8	205/6.7	85/7.3	---
7	80/13	205/5.5	60/41	---
8	95/5.2	210/1.1	110/1.5	205/0.72
9	100/7.8	175/3.7	45/8.7	160/0.96
10	45/1.8	135/1.2	70/3.1	145/0.91
11	90/2.5	205/0.94	90/5.6	205/0.68
12	55/18	---	45/12	---
13	45/19	---	55/12	215/2.0

from the power spectra graphs: (1) peaks below 25 Hz. were dropped; (2) peaks less than 5% of the maximum were dropped; (3) small amplitude peaks between larger ones or in the skirt of a larger one were generally dropped; and (4) small amplitude peaks which appeared to have resulted from only the window action were dropped.

6.3.4 POWER SPECTRA OF S1

The power spectra of S1 are summarized in Table 6-6 (inspiration) and Table 6-7 (expiration). The criteria used for tabulation were the same as were used for the ejection click power spectra. The 50 ms time window was used and this fact should be kept in mind when comparing the click to S1. Patients 14 through 19 are the normal ones.

6.3.5 COMPARISON BETWEEN S1 AND THE EJECTION CLICK

A comparison was made between the later 30 ms of S1 and the first 20 ms of the click with regard to high frequency power (150 to 250 Hz). These data are tabulated in Table 6-8 (inspiration) and Table 6-9 (expiration). The amplitude given is the sum of all peaks greater than 5% of the largest peak (over all frequencies) within the given frequency range. The last column gives the ratio of power in the click to power in S1 as an indication of the increase in high frequency power when the click begins. Data for the

TABLE 6-6

S1 POWER SPECTRA, INSPIRATION

Patient		Frequency/Amplitude X 10^{-3}			
1		35/3.1	80/8.6		
2		30/7.5	90/17	180/1.0	
3		45/52	110/5.7	145/4.2	
4		20/0.34	75/0.15	155/0.11	210/0.13
5		30/1.3	75/2.3	200/0.46	
6		50/35			
7		40/4.3	110/0.48		
8		65/6.0			
9	25/2.1	55/6.1	90/6.7	175/1.4	
10		35/1.5			
11		40/41			
12		40/5.7		180/0.46	
13		50/2.5	90/1.0	210/0.32	
14		40/23	100/15		
15		35/4.6	55/4.8	120/0.34	
16		35/13	70/13	130/1.4	
17		45/22	75/7.3	125/1.4	
18		25/4.2	55/15		
19		25/1.3	70/6.3		

TABLE 6-7

S1 POWER SPECTRA, EXPIRATION

Patient	Frequency/Amplitude X 10^{-3}				
1		35/1.1	80/6.0		
2		35/11	80/30		
3		50/41			
4		25/2.0	60/2.3	140/0.27	190/0.18
5			75/1.3	215/0.40	
6		40/21	70/40		
7		50/4.7			
8		60/5.0	100/1.9		
9	25/2.4	55/6.2	90/5.4	190/1.9	
10		35/0.88	60/0.41		
11		40/19			
12		45/4.5			
13	25/12	75/15	105/13	165/4.6	195/4.0
14		40/17	95/9.0		
15		30/6.4	60/16		
16			65/20	145/1.2	
17		40/11	85/7.9	150/1.0	
18		40/19	70/8.6		
19	25/1.1	55/2.1	105/0.55		

TABLE 6-8

S1 TO CLICK COMPARISON, INSPIRATION

Patient	Power In Click	Power In S1	Click/S1
1	11	0.1	110
2	11	0.7	16
3	45	3	15
4	10	0.2	50
5	7.8	0.7	11
6	5.8	0.3	19
7	2.4	<0.02	>100
8	0.9	0.03	30
9	5.1	1.9	2.7
10	2.1[1]	<0.01	>200
11	0.4	<0.15	>2.7
12	1	0.5	2
13	0	0.4	0
14	0	0	-
15	0.03	0.04	0.8
16	0	0	-
17	0.1	0.7	0.1
18	0	0	-
19	0	0	-

[1] Taken from the 30 ms power spectra.

TABLE 6-9

S1 TO CLICK COMPARISON, EXPIRATION

Patient	Power In Click	Power In S1	Click/S1
1	24	<0.1	>200
2	12	0.4	30
3	28	1.0	28
4	15	0.4	38
5	13	0.4	33
6	6.8	<1.0	>6.8
7	5.5	<0.2	>28
8	2	<0.1	>20
9	5.4	2.0	2.7
10	2[1]	<0.02	>100
11	0.9	<0.4	>2.3
12	0.8	<0.02	>40
13	1	1.7	0.6
14	0.05	0.2	0.3
15	0	0	-
16	0	0.1	0
17	0.2	0.7	0.3
18	0.03	0.2	0.2
19	0.02	0.01	2

[1] Taken from the 30 ms power spectra.

normal patients were taken from the aligned average for S1, in the region where the click would be expected to appear. This region was taken to begin at a point approximately 50 ms after the onset of S1 and 10 to 20 ms before the upward rise of the carotid pulse.

6.4 CONCLUSIONS

The characteristics of the aortic ejection click are summarized by the following statements:

1. The click consists of a high frequency oscillation in the range of 160 to 230 Hz.
2. The power of this oscillation during the first 20 ms is typically (75% of the cases) at least 10 times its value during the second 20 ms.
3. During the first 30 ms of the click there are typically 3 frequency components. Their mean frequencies are approximately 45, 95, and 195 Hz.
4. Power in the frequency range of 150 to 250 Hz typically (80% of the cases) increases by a factor of 10 or more from the last 30 ms preceding the click to the first 20 ms of the click.
5. In 70% of the cases, the peak power in the frequency range of 0 to 150 Hz stays approximately the same or increases from the first 20 ms of the click to the second 20 ms.

6. The click occurs 2 to 10 ms earlier during expiration than during inspiration.

It is suggested that the short high frequency oscillation occurs when the aortic value reaches its opening limit and suddenly offers an increased resistance to the flow of blood. The lower frequencies are possibly either; (1) a continuation of the second component of S1, or a reexcitement of the oscillation modes of S1, in the cases where there is an increase in low frequency energy after the onset of the click.

It is thought that patient 13 probably does not have valvar aortic stenosis. This conclusion is based on the following observations: (1) there is very little energy in the 150 to 250 Hz region; (2) there is a significant change in high frequency energy between inspiration and expiration; (3) the amount of high frequency energy decreases from S1 to the region where the click should occur; and (4) the patient was diagnosed as having only a mild case of the disease.

The first heart sound is characterized by a low frequency oscillation at a mean frequency of 40 Hz and a higher oscillation at a mean frequency of 75 Hz in most cases. In some cases there is also a small third oscillation above 100 Hz. In some of the aortic stenosis patients, there is a small amount of power around 200 Hz. It is possible that this power comes from the beginning of the click.

6.5 SUGGESTIONS FOR FURTHER INVESTIGATION

There is a need for a more convenient method to obtain an aligned average record than the one used here. While the method of searching for the local maximum or minimum is computationally efficient and works well in many cases, it requires careful observations of time series data by the operator and fails when the greatest amplitude shifts from one peak to an adjacent one in different records. Some type of correlation technique is suggested.

An analytic procedure which would plot the amount of energy above 150 Hz versus time would be useful in detecting the aortic ejection click. This procedure could consist of low pass filtering the data, followed by an envelogram analysis. It may also be desirable to normalize this plot by the amount of low frequency energy. Then, a peak or sudden increase in the relative amount of high frequency energy at a point near the upward rise of the carotid pulse would indicate the probable existence of an aortic ejection click.

BIBLIOGRAPHY

1. V. Navaratnam, The Human Heart and Circulation. London: Academic Press, 1975, pp. 15.

2. E. O. Attinger, Pulsatile Blood Flow. New York: McGraw-Hill, 1964

3. R. N. Watts, "A Mathematical Model for Studying the Mechanical Properties of the Impaired Left Ventricle," Ph.D. Dissertation, Rutgers Univ., New Brunswick, N.J., January 1974.

4. R. F. Rushmer, Cardiovascular Dynamics. Philidelphia: W. B. Saunders Co, 1976, pp. 519.

5. M. E. Tavel, Clinical Phonocardiography and External Pulse Recording, 2nd Ed. Chicago. Year Book Medical Publishers, 1972, pp. 36,37.

6. B. J. Bellhouse and F. H. Bellhouse, "Mechanics of Closure of the Aortic Valve," Nature, Vol. 217, January 1978.

7. B. J. Bellhouse and F. H. Bellhouse, "Fluid Mechanics of the Aortic Root with Application to Coronary Flow," Nature, Vol. 219, September 1968.

8. B. J. Bellhouse and L. Talbot, "The Fluid Mechanics of the Aortic Valve," J. Fluid Mech., Vol. 35, Part 4, 1969.

9. B. J. Bellhouse, J. P. Tapas, J. A. Campbell, and P. R. Lurie, "The Roentgenographic Manifestation of Aortic Stenosis and Aortic Valvular Insufficiency, "Amer. J. Roent., Vol. 88, 1962.

10. E. Braunwald and W. F. Friedman, "Aortic Stenosis", in Paediatric Cardiology, Ed. by A. Nadas. Philadelphia: W. R. Saunders Co., 1963.

11. E. J. Epstein, J. M. Criley, E. B. Raftery, J. O'Neal Humphries, and R. S. Ross, "Cineradiographic Studies of the Early Systolic Click in Aortic Valve Stenosis", Circ. Res., Vol. 31, June 1965.

12. W. Dock, "The Forces Needed to Evoke Sounds from Cardiac Tissues and the Attenuation of Heart Sounds", Cir. Res., Vol. 19, March 1959.

13. J. E. Meismer and R. F. Rushmer,"Production of Sounds in Distensible Tubes", Circ. Res., Vol. 12, 1963.

14. J. Faber, M. B. Job, and A. C. Burton, "Biophysics of Heart Sounds and Its Application to Clinical Auscultation", Canac. Med. Assn. J., July 1964.

15. C. A. Caseres, The Innocent Murmur. Boston: Little, Brown and Co., 1967. lsp 1;.of 5

16. R. F. Rushmer and C. Morgan, "Meaning of Murmurs", Amer. J. Cardiol., Vol. 21, May 1968.

17. D. L. Bruns, "A General Theory of the Causes of Murmurs in the Cardiovascular System", Amer. J. Med., September 1959.

18. P. M. Shah, M. Mori, D. M. MacCanon, and A. A. Luisada, "Hemodynamic Correlates of the Various Components of the First Heart Sound", Circ. Res., Vol. 12, 1963.

19. T. E. Piemme, G. O. Barnett, and L. Dexter, "Relationship of Heart Sounds to Acceleration of Blood Flow", Circ. Res., Vol. 18, March 1966.

20. T. Sakamoto, R. Kusukawa, D. M. MacCanon, and A. A. Luisade, "Hemodynamic Determinants of the Amplitude of the First Heart Sound", Circ. Res., Vol. 16, January 1965.

21. R. Zalter, H. C. Hardy, and A. A. Luisada, "Acoustic Transmission Characteristics of the Thorax", Appl. Physiol., Vol. 18, 1963.

22. A. C. Burton, Physiology and Biophysics of the Circulation. Chicago: Year Book Medical Publishers, 1965, pp. 147.

23. A. A. Sarkady, "An Interactive Computer Analysis of Phonocardiograms," Ph.D. dissertation, University of New Hampshire, 1975.

24. J. R. Ragazzini, G. F. Franklin, Sampled-Data Control Systems. New York: McGraw-Hill, 1958, pp. 13-16.

25. B. M. Oliver, J. R. Pierce, and C. E. Shannon, "The Philosophy of Pulse Code Modulation", Proc. IRE, Vol. 36, No. 11, pp. 1324-1331, November 1948.

26. G. D. Bergland, "A guided tour of the fast Fourier transform," IEEE Spectrum, Vol. 6, pp. 41-52, July 1969.

27. J. W. Cooley, J. W. Turkey, "An Allgorithm for the Machine Calculation of Complex Fourier Series," Mathematics of Computation, Vol. 19, No. 90, pp. 297-301, 1965.

28. R. C. Singleton, "An Allgorithm for Computing the Mixed Radix Fast Fourier Transform", IEEE Trans. Audio Electroacoust., Vol. AV-17, pp. 297-301, June 1969.

29. G. W. Jenkins and D. G. Watts, *Spectral Analysis and Its Applications*. San Francisco: Holden-Day, 1968, pp. 50.

30. L. R. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing*. Englewood Cliffs: Prentice-Hall, 1975, pp. 67-77.

31. A. A. Sarkady, R. R. Clark, R. Williams, "Computer Analysis Technics for Phonocardiogram Diagnosis," Computers and Biomedical Research, vol. 9, pp. 349-363, 1976.

APPENDIX 1

INTERACTIVE PHONOCARDIOGRAM ANALYSIS PRORAM LISTING

In the folling listing, lines which were too long were broken in two by the text editor. The second part of any broken line begins two spaces to the left of FORTRAN column one. Lines which did not contain any embedded blanks at which to break were broken at the right margin with this being indicated by a dash "-" at the end of the line.

```
  C  INTERACTIVE PHONOCARDIOGRAM ANALYSIS PROGRAM,
VERSION 1.02 04/78
  C      BY S. J. SHOWALTER, VIRGINIA POLYTECHNIC
INSTITUTE & STATE
  C      UNIVERSITY, BLACKSBURG, VA.  APRIL 1978
  C
        INTEGER
LC(32),CH(136),QU,TT,II,AA,PP,RR,SS,GG,LL,DD,NN,CC
        REAL X(1000),XI(1000),Y(1000),YI(1000)
        COMMON/B1/ISEED,CAL
        COMMON/B2/IXL,XSI,X,XI
        COMMON/B3/IYL,YSI,IYSUM,Y,YI
        COMMON/B8/IPU,IGF,IGL,NB,NE,SL,SU,IFC,MD,LC
        DATA QU,TT,II,AA,PP,RR,SS,GG,LL,DD,NN,CC,BLK/
       *1H?,1HT,1HI,1HA,1HP,1HR,1HS,1HG,1HL,1HD,1HN,1H-
C,1H /
        CALL CLEAR
        ISEED=65539
        CAL=1.0
        DO 8 I=1,32
      8 LC(I)=BLK
        DO 9 I=1,1000
      9 XI(I)=0.0
     10 WRITE(6,11)
     11 FORMAT(' ENTER COMMAND')
        CALL RDTTY(CH,N)
        IF(CH(1).EQ.QU) GO TO 110
        IF(CH(1).EQ.TT) GO TO 112
        IF(CH(1).EQ.II) GO TO 114
        IF(CH(1).EQ.AA) GO TO 116
```

```
      IF(CH(1).NE.PP) GO TO 40
      IF(CH(2).EQ.II) GO TO 120
      GO TO 118
   40 IF(CH(1).EQ.RR) GO TO 122
      IF(CH(1).EQ.SS) GO TO 140
      IF(CH(1).NE.GG) GO TO 60
      IF(CH(2).EQ.RR) GO TO 124
      GO TO 138
   60 IF(CH(1).EQ.LL) GO TO 102
      IF(CH(1).EQ.DD) GO TO 128
      IF(CH(1).EQ.NN) GO TO 130
      IF(CH(1).NE.CC) GO TO 102
      IF(CH(2).EQ.II) GO TO 132
      IF(CH(2).EQ.PP) GO TO 134
      IF(CH(2).EQ.LL) GO TO 136
  102 WRITE(6,103)
  103 FORMAT(' UNKNOWN COMMAND')
      GO TO 10
  110 WRITE(6,111)
  111 FORMAT(' HI DUMMY!')
      GO TO 10
  112 CALL TEST
      GO TO 10
  114 CALL IO
      GO TO 10
  116 CALL ANLYZ
      GO TO 10
  118 CALL TPLOT(X)
      GO TO 10
  120 CALL TPLOT(XI)
      GO TO 10
  122 CALL RUN
      GO TO 10
  124 CALL PSEG
      GO TO 10
  128 CALL DIVIDE
      GO TO 10
  130 CALL NORM
      GO TO 10
  132 CALL CINPT
      GO TO 10
  134 CALL CPLOT
      GO TO 10
  136 CALL CLEAR
      GO TO 10
  138 CALL GATE
      GO TO 10
  140 STOP
      END
C
      SUBROUTINE RUN
```

```
        INTEGER HDR(20)
        INTEGER IANI(10),IANS(100)
        COMMON/B4/INI,IANI,INS,IANS
        COMMON/B5/HDR
        COMMON/B6/IU,NPF,NPL,NIN,NEX,NCR
        DO 100 I=NPF,NPL
        CALL CLEAR
        IU=I+10
        REWIND IU
        READ(IU,5) (HDR(J),J=1,20)
      5 FORMAT(20A4)
        READ(IU,7) NIN,NEX,NCR
      7 FORMAT(3I3)
        CALL NORM
        CALL INPUT
        CALL RUNAN(IANS,INS)
    100 CONTINUE
        RETURN
        END
  C
        SUBROUTINE CINPT
        INTEGER IANI(10),IANS(10)
        INTEGER CH(136),APS(10),AAS(10)
        COMMON/B4/INI,IANI,INS,IANS
        COMMON/B6/IU,NPF,NPL,NIN,NEX,NCR
        COMMON/B7/IDT,APS,AAS,NBS,NAS,NBM,IRS,ITF,ITL,-
IRN,IAL
        WRITE(6,11)
     11 FORMAT(' ENTER FIRST AND LAST PATIENT.')
        READ(5,*) PNB,PNE
        NPF=IFIX(PNB)
        NPL=IFIX(PNE)
        IU=NPF+10
        WRITE(6,21)
     21 FORMAT(' ENTER: DATA SET, FST & LST TIMES(MS),
NUM OF RPTS, ANALY        S
       *IS LENGTH.')
        READ(5,*) RS,TF,TL,RN,AL
        IF((RS.EQ.0.0).OR.(RN.EQ.0.0)) GO TO 100
        IRS=IFIX(RS)
        ITF=IFIX(TF)
        ITL=IFIX(TL)
        IRN=IFIX(RN)
        IAL=IFIX(AL)
        WRITE(6,31)
     31 FORMAT(' ANALYSIS ON INPUT')
        CALL RDANL(IANI,INI)
        WRITE(6,33)
     33 FORMAT(' ANALYSIS ON SUMMATION')
        CALL RDANL(IANS,INS)
    100 RETURN
```

```
      END
  C
      SUBROUTINE INPUT
      INTEGER
IAN(10),U(1000),ISHIFT(100),APS(10),AAS(10)
      INTEGER
PWSP,ANSG,IMAG,TRSF,SYNC,SHFT,END,MAX,MIN
      REAL X(1000),XI(1000),Y(1000),YI(1000)
      LOGICAL*1 W(4000),C
      COMMON/B1/ISEED,CAL
      COMMON/B2/IXL,XSI,X,XI
      COMMON/B3/IYL,YSI,IYSUM,Y,YI
      COMMON/B4/IN,IAN
      COMMON/B6/IU,NPF,NPL,NIN,NEX,NCR
      COMMON/B7/IDT,APS,AAS,NBS,NAS,NBM,IRS,ITF,ITL,-
IRN,IAL,CNST,TAU,FR        E
     *Q1,PHI1,FREQ2,PHI2,SV,ILZ,ISN,ITZ,INSW,RNP1,RN-
P2
      COMMON/B8/IPU,IGF,IGL,NB,NE
      COMMON/B9/MXMN,ISAV,ISHIFT
      DATA PWSP,ANSG,IMAG,TRSF,SYNC,SHFT,END,MAX,MIN/
     *4HPWSP,4HANSG,4H MAG,4HTRSF,4HSYNC,4HSHFT,4H
END,4H MAX,4H MIN/
      EQUIVALENCE(X(1),U(1),W(1))
      IF(IRN.LT.1)GO TO 110
      IF(IN.LT.1) GO TO 10
      DO 9 I=1,IN
      NBS=NBS+1
      IF(IAN(I).EQ.1) APS(NBS)=PWSP
      IF(IAN(I).EQ.2) APS(NBS)=ANSG
      IF(IAN(I).EQ.3) APS(NBS)=IMAG
      IF(IAN(I).EQ.4) APS(NBS)=TRSF
      IF(IAN(I).NE.5) GO TO 7
      APS(NBS)=SYNC
      ISTF=ITF
      ISTL=ITL
    7 IF(IAN(I).NE.6) GO TO 8
      APS(NBS)=SHFT
      NBS=NBS+1
      APS(NBS)=MAX
      IF(MXMN.LE.1) APS(NBS)=MIN
      NBS=NBS+1
      IT=ISTF
      CALL ICCON(IT)
      APS(NBS)=IT
      NBS=NBS+1
      IT=ISTL
      CALL ICCON(IT)
      APS(NBS)=IT
    8 IF(IAN(I).EQ.7) NBS=NBS-1
    9 CONTINUE
```

```
10 NBS=NBS+1
   APS(NBS)=END
   IDT=2
   ISF=IFIX(ITF*2.5)+1
   ISL=IFIX(ITL*2.5)+1
   GO TO (12,44),IRS
   GO TO 20
12 REWIND IU
   DO 14 I=1,2
14 READ(IU,15)
15 FORMAT(1X)
   GO TO 44
20 JUMP=8
   IF(IRS.GT.3) JUMP=JUMP+8*NIN
   IF(IRS.GT.4) JUMP=JUMP+8*NEX
   IF(IRS.GT.5) JUMP=JUMP+8*NCR
   DO 22 I=1,JUMP
22 READ(IU,15)
44 DO 100 K=1,IRN
   IXL=IAL
   IF(IXL.LT.NE) NE=IXL
   XSI=0.0004
   IF(IRS.EQ.6) XSI=0.0016
   DO 45 J=1,1000
45 U(J)=0.0
   IF(K/5*5.EQ.K)WRITE(6,46)K
46 FORMAT(' CYCLE',I3)
   IE=2536
   IF(IRS.GT.5) IE=616
   READ(IU,47,END=106)C,C,C,C,C,C,(W(J),J=4,IE,4)
47 FORMAT(80A1)
   IF(ISF.LT.1) ISF=1
   NSUM=0
   DO 57 I=ISF,ISL
57 NSUM=NSUM+U(I)
   BIAS=FLOAT(NSUM)/FLOAT(ISL-ISF+1)
   DO 58 I=ISF,ISL
58 X(I)=(FLOAT(U(I))-BIAS)/CAL
   IF(ISF.LE.1) GO TO 64
   IE=ISL-ISF+1
   DO 62 I=1,IE
62 X(I)=X(I+ISF-1)
64 IE=ISL-ISF+2
   IF(IE.GT.634) GO TO 68
   DO 66 I=IE,1000
66 X(I)=0.0
68 IF(IN.LT.1) GO TO 90
   DO 88 I=1,IN
   IGOTO=IAN(I)
```

```
      GO TO
(71,72,73,88,75,76,77,88,88,88,88,78,79),IGOTO
      GO TO 88
   71 CALL PWRSPM
      GO TO 88
   72 CALL ANLSIG
      GO TO 88
   73 CALL MAG
      GO TO 88
   75 CALL SYNCH(K)
      GO TO 88
   76 CALL SHIFT(K)
      GO TO 88
   77 CALL TPLOT(X)
      GO TO 88
   78 CALL FILTER
      GO TO 88
   79 CALL TPLOT(XI)
   88 CONTINUE
   90 DO 92 I=1,1000
      Y(I)=Y(I)+X(I)
   92 YI(I)=YI(I)+XI(I)
      IYSUM=IYSUM+1
  100 CONTINUE
      GO TO 110
  106 WRITE(6,107)
  107 FORMAT(' END OF FILE ENCOUNTERED IN SUBROUTINE
INPUT.')
  110 IYL=IXL
      YSI=XSI
      RETURN
      END
C
      SUBROUTINE IO
      INTEGER
CH(136),CAL,XSI,X(327),YSI,Y(327),N5(20),N6(6),N7(42),
     *AA,II,OO
      REAL YZ(1673)
      DATA AA,II,OO/1HA,1HI,1HO/
      COMMON/B1/ISEED,CAL
      COMMON/B2/IXL,XSI,X
      COMMON/B3/IYL,YSI,IYSUM,Y,YZ
      COMMON/B5/N5
      COMMON/B6/N6
      COMMON/B7/N7
      WRITE(6,3)
    3 FORMAT(' ENTER FILE NUMBER, (30-99).')
      READ(5,*)UN
      IU=IFIX(UN)
      IF(IU.EQ.0) GO TO 40
      REWIND IU
```

```
     10 WRITE(6,11)
     11 FORMAT(' INPUT OR OUTPUT?')
        CALL RDTTY(CH,N)
        IF(CH(1).EQ.II) GO TO 20
        IF(CH(1).EQ.OO) GO TO 30
        IF(CH(1).EQ.AA) GO TO 40
        GO TO 10
     20
READ(IU,23)ISEED,CAL,IYL,YSI,(Y(I),I=1,327),IYSUM,(N5(-
I),I=1,20),
       *(N6(I),I=1,6),(N7(I),I=1,42)
     23 FORMAT(20A4)
        IF(N7(1).GE.2) IYL=N7(29)
        DO 26 I=1,1673
     26 YZ(I)=0.0
        RETURN
     30 IXLO=IXL
        IF(IXL.LE.327) GO TO 34
        WRITE(6,33)
     33 FORMAT(' WARNING!!! DATA EXCEEDS 327 POINTS.
REMAINDER WILL BE TR        U
       *NCATED.')
        IXLO=327
     34
WRITE(IU,23)ISEED,CAL,IXLO,XSI,(X(I),I=1,327),IYSUM,(N-
5(I),I=1,20        )
       *,(N6(I),I=1,6),(N7(I),I=1,42)
     40 RETURN
        END
  C
        SUBROUTINE SYNCH(K)
        INTEGER ISHIFT(100)
        REAL X(1000)
        INTEGER APS(10),AAS(10)
        COMMON/B2/IXL,XSI,X
        COMMON/B7/IDT,APS,AAS,NBS,NAS,NBM,IRS,ITF,ITL,-
IRN
        COMMON/B9/MXMN,ISAV,ISHIFT
        IF(MXMN.EQ.2) GO TO 20
        ITEMP=10000
        DO 10 I=1,IXL
        IF(X(I).GE.ITEMP) GO TO 10
        ITEMP=X(I)
        ISHIFT(K)=I
     10 CONTINUE
        IF(K.EQ.IRN) GO TO 30
        RETURN
     20 ITEMP=-10000
        DO 25 I=1,IXL
        IF(X(I).LE.ITEMP) GO TO 25
        ITEMP=X(I)
```

```
      ISHIFT(K)=I
   25 CONTINUE
      IF(K.EQ.IRN) GO TO 30
      RETURN
   30 ITEMP=0
      DO 32 I=1,IRN
   32 ITEMP=ITEMP+ISHIFT(I)
      ITEMP=ITEMP/IRN
      WRITE(6,33)ITEMP
   33 FORMAT(' ITEMP =',I4)
      DO 34 I=1,IRN
   34 ISHIFT(I)=ITEMP-ISHIFT(I)
      WRITE(6,37) (ISHIFT(I),I=1,IRN)
   37 FORMAT(2(5(1X,I3),2X))
      ISAV=ITF+IFIX(FLOAT(ITEMP)*0.4)
      IF(MXMN.EQ.2) GO TO 38
      WRITE(6,35)ISAV
   35 FORMAT(' THE AVERAGE MINIMUN OCCURS AT',I4,'
MILLISECONDS.')
      RETURN
   38 WRITE(6,39)ISAV
   39 FORMAT(' THE AVERAGE MAXIMUN OCCURS AT',I4,'
MILLISECONDS.')
      RETURN
      END
C
      SUBROUTINE SHIFT(K)
      INTEGER ISHIFT(100)
      REAL X(1000)
      COMMON/B2/IXL,XSI,X
      COMMON/B9/MXMN,ISAV,ISHIFT
      IF(ISHIFT(K).GT.0) GO TO 10
      IF(ISHIFT(K).LT.0) GO TO 20
      RETURN
   10 IE=IXL-ISHIFT(K)
      DO 12 I=1,IE
      J=IXL-I+1
   12 X(J)=X(J-ISHIFT(K))
      IE=ISHIFT(K)
      DO 14 I=1,IE
   14 X(I)=0.0
      RETURN
   20 IE=IXL+ISHIFT(K)
      DO 22 I=1,IE
   22 X(I)=X(I-ISHIFT(K))
      IE=IE+1
      DO 24 I=IE,IXL
   24 X(I)=0.0
      RETURN
      END
C
```

```
      SUBROUTINE ANLYZ
      INTEGER IAN(10)
      CALL RDANL(IAN,IN)
      CALL RUNAN(IAN,IN)
      RETURN
      END
C
      SUBROUTINE RUNAN(IAN,IN)
      INTEGER APS(10),AAS(10),IAN(10),PWSP,ANSG,IMAG
      COMMON/B2/IXL,XSI,X,XI
      COMMON/B7/IDT,APS,AAS,NBS,NAS
      DATA PWSP,ANSG,IMAG/4HPWSP,4HANSG,4H MAG/
      IF(IN.LT.1) GO TO 32
      DO 30 I=1,IN
      NAS=NAS+1
      IGOTO=IAN(I)
      GO TO
(10,11,12,13,30,30,14,15,16,17,18,19,20),IGOTO
      GO TO 30
   10 CALL PWRSPM
      AAS(NAS)=PWSP
      GO TO 30
   11 CALL ANLSIG
      AAS(NAS)=ANSG
      GO TO 30
   12 CALL MAG
      AAS(NAS)=IMAG
      GO TO 30
   13 CALL TRNSF
      NAS=NAS-1
      GO TO 30
   14 CALL TPLOT(X)
      NAS=NAS-1
      GO TO 30
   15 CALL DIVIDE
      GO TO 30
   16 CALL PSEG
      GO TO 30
   17 CALL IO
      GO TO 30
   18 CALL GATE
      GO TO 30
   19 CALL FILTER
      GO TO 30
   20 CALL TPLOT(XI)
      NAS=NAS-1
   30 CONTINUE
   32 RETURN
      END
C
      SUBROUTINE CPLOT
```

```
        INTEGER CH(136),LC(32),TT,RR,NN,YY,BLK
        COMMON/B2/IXL,XSI
        COMMON/B8/IPU,IGF,IGL,NB,NE,SL,SU,IFC,MD,CL
        DATA TT,RR,NN,YY,BLK/1HT,1HR,1HN,1HY,1H /
        GO TO 2
      1 BACKSPACE 5
      2 WRITE(6,3)
      3 FORMAT(' ENTER PLOT INDEX')
        READ(5,*,END=1) RMD
        IF(RMD.LT.1.0) GO TO 40
        MD=IFIX(RMD)
      4 WRITE(6,5)
      5 FORMAT(' PLOT AT TERM. OR RMT ?')
        CALL RDTTY(CH,N)
        IF(CH(1).NE.TT.AND.CH(1).NE.RR) GO TO 4
        IPU=6
        IF(CH(1).EQ.RR) IPU=7
     10 WRITE(6,11)
     11 FORMAT(' DO YOU WANT TO CHANGE PLOT
PARAMETERS?')
        CALL RDTTY(CH,N)
        IF(CH(1).EQ.NN) GO TO 30
        IF(CH(1).EQ.YY) GO TO 14
        WRITE(6,13)
     13 FORMAT(' UNDECODABLE RESPONSE')
        GO TO 10
     14 WRITE(6,15)
     15 FORMAT(' ENTER: BEGIN, END, LOWER SCALE, UPPER
SCALE, FORMAT.')
        READ(5,*,END=18) RNB,RNE,SL,SU,FC
        GO TO 20
     18 BACKSPACE 5
        GO TO 14
     20 NB=IFIX(RNB)
        NE=IFIX(RNE)
        IFC=IFIX(FC)
     30 DO 32 I=1,32
     32 LC(I)=BLK
        WRITE(6,35)
     35 FORMAT(' ENTER COMMENT LINE.')
        READ(5,37,END=38)LC
     37 FORMAT(32A4)
        GO TO 40
     38 BACKSPACE 5
     40 RETURN
        END
  C
        SUBROUTINE TRNSF
        REAL X(1000),XI(1000),Y(1000),YI(1000)
        INTEGER APS(10),AAS(10)
        COMMON/B2/IXL,XSI,X,XI
```

```
      COMMON/B3/IYL,YSI,IYSUM,Y,YI
      COMMON/B7/IDT,APS,AAS,NBS,NAS,NBM
      COMMON/B8/IPU,IGF,IGL
      IGF=0
      NAS=NBM
      DO 10 I=1,IYL
      X(I)=Y(I)
   10 XI(I)=YI(I)
      IXL=IYL
      XSI=YSI
      RETURN
      END
C
      SUBROUTINE GATE
      INTEGER GT,APS(10),AAS(10)
      REAL X(1000),XI(1000)
      COMMON/B2/IXL,XSI,X,XI
      COMMON/B7/IDT,APS,AAS,NBS,NAS,NBM,IRS,ITF
      COMMON/B8/IPU,IGF,IGL
      DATA GT/4HGATE/
   10 WRITE(6,11)
   11 FORMAT(' ENTER GATING TIMES(MS).')
      READ(5,*) GF,GL
      IGF=IFIX(GF)
      IF(IGF.LT.ITF) GO TO 10
      IGL=IFIX(GL)
      IS=IFIX(FLOAT(IGF-ITF)*2.5)
      IF(IGF.LE.ITF) GO TO 20
      IE=IXL-IS+1
      DO 18 I=1,IE
      X(I)=X(I+IS)
   18 XI(I)=XI(I+IS)
   20 IE=IFIX(FLOAT(IGL-IGF+1)*2.5)
      IF(IE.GT.IXL) IE=IXL
      DO 28 I=IE,IXL
      X(I)=0.0
   28 XI(I)=0.0
      NAS=NAS+1
      AAS(NAS)=GT
      NAS=NAS+1
      IT=IGF
      CALL ICCON(IT)
      AAS(NAS)=IT
      NAS=NAS+1
      IT=IGL
      CALL ICCON(IT)
      AAS(NAS)=IT
      RETURN
      END
C
      SUBROUTINE ICCON(N)
```

```
  CONVERT INTEGER N, (0 TO 9999), TO IT'S EBCDIC
CHARACTER REPRESENTATION        .
        N4=N/1000
        N=N-N4*1000
        N3=N/100
        N=N-N3*100
        N2=N/10
        N=N-N2*10
        N=(240+N4)*2**24+(240+N3)*2**16+(240+N2)*256+2-
40+N
        RETURN
        END
  C
        SUBROUTINE CLEAR
        REAL Y(1000),YI(1000)
        INTEGER APS(10),AAS(10)
        COMMON/B3/IYL,YSI,IYSUM,Y,YI
        COMMON/B7/IDT,APS,AAS,NBS,NAS,NBM
        IYSUM=0
        IXL=0
        NAS=0
        NBS=0
        NBM=0
        DO 10 I=1,1000
        Y(I)=0.0
     10 YI(I)=0.0
        RETURN
        END
  C
        SUBROUTINE DIVIDE
        REAL Y(1000),YI(1000)
        INTEGER APS(10),AAS(10),DIV
        COMMON/B3/IYL,YSI,IYSUM,Y,YI
        COMMON/B7/IDT,APS,AAS,NBS,NAS
        DATA DIV/4H DIV/
        T=FLOAT(IYSUM)
        DO 10 I=1,1000
        Y(I)=Y(I)/T
     10 YI(I)=YI(I)/T
        NAS=NAS+1
        AAS(NAS)=DIV
        RETURN
        END
  C
        SUBROUTINE NORM
        INTEGER U(640)
        LOGICAL*1 W(2560),C
        COMMON/B1/ISEED,CAL
        COMMON/B6/IU
        EQUIVALENCE(U(1),W(1))
        DO 4 I=1,640
```

```
      4 U(I)=0
        READ(IU,11)C,C,C,C,C,C,(W(I),I=4,2536,4)
     11 FORMAT(80A1)
        IT=0
        DO 6 I=1,634
      6 IT=IT+U(I)
        IT=IT/634
        DO 8 I=1,634
      8 U(I)=U(I)-IT
        IT=0
        DO 12 I=1,634
     12 IT=IT+U(I)*U(I)
        CAL=FLOAT(IT)
        CAL=SQRT(CAL/634.0)
        WRITE(6,13)CAL
     13 FORMAT(' CALIBRATION VALUE ='G12.5)
        RETURN
        END
  C
        SUBROUTINE RDTTY(CH,N)
        INTEGER CH(136),DOL,BLK
        DATA DOL,BLK/1H$,1H /
        DO 10 I=1,133
     10 CH(I)=DOL
        READ(5,13,END=14) (CH(J),J=1,132)
     13 FORMAT(132A1)
        GO TO 16
     14 BACKSPACE 5
     16 DO 18 I=1,132
        N=I-1
        IF(CH(1).EQ.DOL) GO TO 20
     18 CONTINUE
     20 IF(CH(1).NE.BLK) GO TO 24
        DO 22 I=1,N
     22 CH(I)=CH(I+1)
        GO TO 20
     24 RETURN
        END
  C
        SUBROUTINE RDANL(IAN,IN)
        INTEGER
IAN(10),CH(136),DOL,SS,PP,AA,MM,TT,YY,HH,DD,GG,NN,II,R-
R,F       F
        COMMON/B9/MXMN
        DATA
DOL,SS,PP,AA,MM,TT,YY,HH,DD,GG,NN,II,RR,FF/
       *1H$,1HS,1HP,1HA,1HM,1HT,1HY,1HH,1HD,1HG,1HN,1H-
I,1HR,1HF/
        DO 200 I=1,10
        IN=I-1
     12 WRITE(6,13)
```

```
 13 FORMAT(' ENTER ANALYSIS')
    CALL RDTTY(CH,N)
    IF(CH(1).EQ.DOL) GO TO 202
    IF(CH(1).EQ.SS.AND.CH(2).EQ.PP) GO TO 20
    IF(CH(1).EQ.AA) GO TO 21
    IF(CH(1).EQ.MM) GO TO 22
    IF(CH(1).EQ.TT) GO TO 23
    IF(CH(1).EQ.SS.AND.CH(2).EQ.YY) GO TO 24
    IF(CH(1).EQ.SS.AND.CH(2).EQ.HH) GO TO 29
    IF(CH(1).NE.PP) GO TO 15
    IF(CH(2).EQ.II) GO TO 36
    GO TO 30
 15 IF(CH(1).EQ.DD) GO TO 31
    IF(CH(1).NE.GG) GO TO 17
    IF(CH(2).EQ.RR) GO TO 32
    GO TO 34
 17 IF(CH(1).EQ.II) GO TO 33
    IF(CH(1).EQ.FF) GO TO 35
    WRITE(6,19)
 19 FORMAT(' UNDECODABLE ANALYSIS')
    GO TO 12
 20 IAN(I)=1
    GO TO 200
 21 IAN(I)=2
    GO TO 200
 22 IAN(I)=3
    GO TO 200
 23 IAN(I)=4
    GO TO 200
 24 IAN(I)=5
 25 WRITE(6,26)
 26 FORMAT(' SEARCH FOR MOST POSITIVE OR NEGATIVE
VALUE?')
    CALL RDTTY(CH,N)
    IF(CH(1).EQ.PP) GO TO 27
    IF(CH(1).EQ.NN) GO TO 28
    GO TO 25
 27 MXMN=2
    GO TO 200
 28 MXMN=1
    GO TO 200
 29 IAN(I)=6
    GO TO 200
 30 IAN(I)=7
    GO TO 200
 31 IAN(I)=8
    GO TO 200
 32 IAN(I)=9
    GO TO 200
 33 IAN(I)=10
    GO TO 200
```

```
   34 IAN(I)=11
      GO TO 200
   35 IAN(I)=12
      GO TO 200
   36 IAN(I)=13
  200 CONTINUE
  202 RETURN
      END
C
      SUBROUTINE PWRSPM
      REAL X(1000),XI(1000)
      COMMON/B2/IXL,XSI,X,XI
      CALL FFT(X,XI,IXL,IXL,IXL,1)
      DO 10 I=1,IXL
      IF(ABS(X(I)).LT.1.0E-38)  X(I)=0.0
      IF(ABS(XI(I)).LT.1.E-38)  XI(I)=0.0
      X(I)=(X(I)*X(I)+XI(I)*XI(I))/(IXL*IXL)
   10 XI(I)=0.0
      XSI=1.0/(XSI*IXL)
      IXL=IXL/2
      RETURN
      END
C
      SUBROUTINE ANLSIG
      REAL X(1000),XI(1000)
      COMMON/B2/IXL,XSI,X,XI
      CALL FFT(X,XI,IXL,IXL,IXL,1)
      T=FLOAT(IXL)
      II=IXL/2+1
      DO 10 I=II,IXL
      XI(I)=0.0
   10 X(I)=0.0
      II=II-1
      X(1)=X(1)/T
      XI(1)=XI(1)/T
      DO 20 I=2,II
      X(I)=X(I)*2.0/T
   20 XI(I)=XI(I)*2.0/T
      CALL FFT(X,XI,IXL,IXL,IXL,-1)
      RETURN
      END
C
      SUBROUTINE MAG
      REAL X(1000),XI(1000)
      COMMON/B2/IXL,XSI,X,XI
      DO 10 I=1,IXL
      IF(ABS(X(I)).LT.1.E-38)  X(I)=0.
      IF(ABS(XI(I)).LT.1.0E-38) XI(I)=0.
      X(I)=SQRT(X(I)*X(I)+XI(I)*XI(I))
   10 XI(I)=0.0
      RETURN
```

```
         END
  C
         SUBROUTINE PSEG
         REAL Y(1000)
         COMMON/B2/IXL,XSI
         COMMON/B3/IYL,YSI,IYSUM,Y
         SN=0.0
         SD=0.0
         DO 10 I=1,IXL
         SN=SN+Y(I)*XSI*(I-1)
      10 SD=SD+Y(I)
         F=SN/SD
         SN=FLOAT(IXL-1)
         SD=1250.0/FLOAT(2*IXL)
         SF=SQRT((F*F/SN+SD*SD*((2.*SN*SN+3.*SN+1.)/(6.-
*SN))-SD*F*(SN+1.)/         S
        *N)/FLOAT(IYSUM))
         PS=0.634*F-46.0
         SN=0.634*SF
         WRITE(6,11)PS,SN,F,SF
      11 FORMAT(/1X,'P.S.E.G. =',F6.1,'MM HG  PLUS OR
MINUS',F5.1,'   F BA        R
        * =',F6.1,'CPS  PLUS OR MINUS',F5.1/)
         RETURN
         END
  C
         BLOCK DATA
         COMMON/B8/IPU,IGF,IGL,NB,NE,SL,SU,IFC,MD
         DATA
IPU,IGF,IGL,NB,NE,SL,SU,IFC,MD/6,0,0,1,100,0.0,0.0,3,1/
         END
  C
  C
  C     DETERMINISTIC SIGNAL GENERATION SUBROUTINES FOR
USE WITH:
  C  INTERACTIVE PHONOCARDIOGRAM ANALYSIS PROGRAM,
VERSION 1.02 04/78
  C      BY S. J. SHOWALTER, VIRGINIA POLYTECHNIC
INSTITUTE & STATE
  C      UNIVERSITY, BLACKSBURG, VA.  APRIL 1978
  C
         SUBROUTINE TEST
         INTEGER CH(136),APS(10),AAS(10),IAN(10)
         INTEGER NN,YY,UU,GG,PWSP,ANSG,IMAG,END
         REAL X(1000),XI(1000),Y(1000),YI(1000)
         COMMON/B2/IXL,XSI,X,XI
         COMMON/B3/IYL,YSI,IYSUM,Y,YI
         COMMON/B7/IDT,APS,AAS,NBS,NAS,NBM,IRS,ITF,ITL,-
IRN,IAL,CNST,TAU,FR         E
```

```
      *Q1,PHI1,FREQ2,PHI2,SV,ILZ,ISN,ITZ,INSW,RNP1,RN-
P2
       COMMON/B8/IPU,IGF,IGL,NB,NE
       DATA NN,YY,UU,GG/1HN,1HY,1HU,1HG/,
      *PWSP,ANSG,IMAG,END/4HPWSP,4HANSG,4H MAG,4H END/
       GO TO 10
     9 BACKSPACE 5
    10 WRITE(6,11)
    11 FORMAT(' ENTER: CONSTANT, TAU, FREQ1,
PHI1(DEGREES), FREQ2, PHI2,        '
      */'          SAMPLE INTERVAL, NUMBER OF(ZEROES,
SAMPLES, ZEROES,REPE         A
      *TS).')
       READ(5,*,END=9)CNST,TAU,FREQ1,PHI1,FREQ2,PHI2,-
SV,ZL,SN,ZT,RN
       IF(RN.EQ.0.0) GO TO 90
       ITSW=1
       I1SW=1
       I2SW=1
       INSW=1
       IF(TAU.NE.0.0) ITSW=2
       IF(FREQ1.NE.0.0) I1SW=2
       IF(FREQ2.NE.0.0) I2SW=2
       IF(TAU.EQ.0.0) GO TO 8
       XPT=SV/TAU
     8 OMEGA1=FREQ1*SV*6.2831853
       OMEGA2=FREQ2*SV*6.2831853
       PHR1=PHI1*0.01745329
       PHR2=PHI2*0.01745329
       IRN=IFIX(RN)
       ILZ=IFIX(ZL)
       ISN=IFIX(SN)
       ITZ=IFIX(ZT)
       XSI=SV
       RNP1=0.0
       RNP2=0.0
    12 WRITE(6,13)
    13 FORMAT(' DO YOU WISH TO ADD NOISE?')
       CALL RDTTY(CH,N)
       IF(CH(1).EQ.NN) GO TO 30
       IF(CH(1).EQ.YY) GO TO 16
       WRITE(6,15)
    15 FORMAT(' UNDECODABLE RESPONSE')
       GO TO 12
    16 WRITE(6,17)
    17 FORMAT(' ENTER NOISE TYPE: UNIFORM OR
GAUSSIAN.')
       CALL RDTTY(CH,N)
       IF(CH(1).EQ.UU) GO TO 20
       IF(CH(1).EQ.GG) GO TO 24
       WRITE(6,19)
```

```
 19 FORMAT(' UNDECODABLE NOISE TYPE')
    GO TO 16
 20 INSW=2
 22 WRITE(6,23)
 23 FORMAT(' ENTER: LOWER LIMIT, UPPER LIMIT.')
    READ(5,*,END=28) RNP1,RNP2
    GO TO 30
 24 INSW=3
 26 WRITE(6,27)
 27 FORMAT(' ENTER: STANDARD DEVIATION, MEAN.')
    READ(5,*,END=29) RNP1,RNP2
    GO TO 30
 28 BACKSPACE 5
    GO TO 22
 29 BACKSPACE 5
    GO TO 26
 30 IF(IRN.LT.1) GO TO 90
    CALL RDANL(IAN,IN)
    IF(IN.LT.1) GO TO 131
    DO 31 I=1,IN
    NBS=NBS+1
    IF(IAN(I).EQ.1) APS(NBS)=PWSP
    IF(IAN(I).EQ.2) APS(NBS)=ANSG
    IF(IAN(I).EQ.3) APS(NBS)=IMAG
 31 CONTINUE
131 NBS=NBS+1
    APS(NBS)=END
    IDT=1
    DO 78 I=1,IRN
    IF(ISN.LT.1) GO TO 39
    IF(ILZ.LT.1) GO TO 33
    DO 32 J=1,ILZ
    X(J)=0.0
 32 XI(J)=0.0
 33 IB=ILZ+1
    IE=ILZ+ISN
    DO 38 J=IB,IE
    X(J)=CNST
    XI(J)=0.0
    IF(ITSW.EQ.1) GO TO 34
    X(J)=X(J)*EXP(0.0-(J-1-ILZ)*XPT)
 34 IF(I1SW.EQ.1) GO TO 36
    X(J)=X(J)*COS((J-1-ILZ)*OMEGA1+PHR1)
 36 IF(I2SW.EQ.1) GO TO 38
    X(J)=X(J)*COS((J-1-ILZ)*OMEGA2+PHR2)
 38 CONTINUE
    IXL=ILZ+ISN+ITZ
 39 IF(ITZ.LT.1) GO TO 41
    IB=ILZ+ISN+1
    DO 40 J=IB,IXL
    X(J)=0.0
```

```
   40 XI(J)=0.0
   41 IF(INSW.LT.2) GO TO 42
      CALL NOISE(INSW,RNP1,RNP2)
   42 IF(IN.LT.1) GO TO 72
      DO 70 J=1,IN
      IGOTO=IAN(J)
      GO TO (44,45,46),IGOTO
      GO TO 70
   44 CALL PWRSPM
      GO TO 70
   45 CALL ANLSIG
      GO TO 70
   46 CALL MAG
      GO TO 70
   70 CONTINUE
   72 DO 74 J=1,IXL
      Y(J)=Y(J)+X(J)
   74 YI(J)=YI(J)+XI(J)
      IYSUM=IYSUM+1
   78 CONTINUE
   90 CONTINUE
      DO 100 I=1,IXL
      X(I)=Y(I)
  100 XI(I)=YI(I)
      IYL=IXL
      IF(IXL.LT.NE) NE=IXL
      YSI=XSI
      RETURN
      END
C
      SUBROUTINE NOISE(INSW,RNP1,RNP2)
      REAL X(1000),XI(1000)
      COMMON/B1/ISEED
      COMMON/B2/IXL,XSI,X,XI
      IF(INSW.EQ.3) GO TO 12
      DO 10 I=1,IXL
      CALL RANDU(ISEED,ISRET,RN)
      ISEED=ISRET
   10 X(I)=X(I)+RN*(RNP2-RNP1)+RNP1
      GO TO 16
   12 DO 14 I=1,IXL
      CALL GAUSS(ISEED,RNP1,RNP2,RN)
   14 X(I)=X(I)+RN
   16 RETURN
      END
C
      SUBROUTINE RANDU(IX,IY,YFL)
      IY=IX*65539
      IF(IY) 5,6,6
    5 IY=IY+2147483647+1
    6 YFL=IY
```

```
        YFL=YFL*0.4656613E-9
        RETURN
        END
  C
        SUBROUTINE GAUSS(IX,S,AM,V)
        A=0.0
        DO 50 I=1,12
        CALL RANDU(IX,IY,Y)
        IX=IY
     50 A=A+Y
        V=(A-6.0)*S+AM
        RETURN
        END
  C
  C
  C    TERMINAL PLOTTING SUBROUTINE FOR USE WITH:
  C  INTERACTIVE PHONOCARDIOGRAM ANALYSIS PROGRAM,
     VERSION 1.02 04/78
  C      BY S. J. SHOWALTER, VIRGINIA POLYTECHNIC
INSTITUTE
  C      & STATE UNIVERSITY, BLACKSBURG, VA.  APRIL
1978
  C    THIS SUBROUTINE REQUIRES A 132 COLUMN TERMINAL.
  C
        SUBROUTINE TPLOT(A)
        INTEGER NB,NE,FC,FCC,PC,T,Y,MD
        INTEGER APS(10),AAS(10),HDR(20),LC(32),VFT(5)
        INTEGER FMT(9)/4H),   ,4H('  ,4H  +',4H   ',4H('
',
       *4H('--,4H--+',4H---',4H('-'/
        INTEGER CH(19)
       */' ','G11.','11X,','4,','|,','+,','=,',' ','
',' ',
       *' ',' ','T18,',' ','T67,','
1H','*,','>,','<,'/
        INTEGER VF(25)
       */'(5X,',' ',' ','  1H',' ',' ',' ',' ',' ','
',' ',
       *' ',' ',' ',' ',' ',' ',' ',' ',' ',
       *'   T',' ',', 1H',' ','1X)'/
        INTEGER
DG(117)/'1','2','3','4','5','6','7','8','9','10','11',-
'12        '
       *,'13','14','15','16','17','18','19','20','21',-
'22','23','24','25'       ,
       *'26','27','28','29','30','31','32','33','34','-
35','36','37','38',       '
       *39','40','41','42','43','44','45','46','47','4-
8','49','50','51','       5
```

```
       *2','53','54','55','56','57','58','59','60','61-
','62','63','64','6       5
       *','66','67','68','69','70','71','72','73','74'-
,'75','76','77','78        '
       *,'79','80','81','82','83','84','85','86','87',-
'88','89','90','91'        ,
       *'92','93','94','95','96','97','98','99','100',-
'101','102','103','        1
       *04','105','106','107','108','109','110','111',-
'112','113','114','        1
       *15','116','117'/
        REAL
A(1000),SL,SU,SI,MAX,MIN,VS(11),HS,TEMP,SIGN
        COMMON/B2/IXL,SI
        COMMON/B5/HDR
        COMMON/B7/IDT,APS,AAS,NBS,NAS,NBM,IRS,ITF,ITL,-
IRN,IAL,CNST,TAU,FR        E
       *Q1,PHI1,FREQ2,PHI2,SV,ILZ,ISN,ITZ,INSW,RNP1,RN-
P2
        COMMON/B8/N,IGF,IGL,NBC,NEC,SLC,SUC,FCC,MDC,LC
        NB=NBC
        NE=NEC
        SL=SLC
        SU=SUC
        FC=FCC
        MD=MDC
        IHSOF=0
        HSOF=FLOAT(IHSOF)/1000
        CH(11)=FMT(1)
        CH(14)=FMT(5)
        T=IABS(FC)
        GO TO(80,60,70),T
     60 CH(8)=FMT(2)
        CH(9)=FMT(3)
        CH(10)=FMT(4)
        CH(12)=FMT(5)
        GO TO 80
     70 CH(8)=FMT(6)
        CH(9)=FMT(7)
        CH(10)=FMT(8)
        CH(12)=FMT(9)
     80 IF(IDT.GE.2) GO TO 90
        WRITE(W,81)CNST,TAU,FREQ1,PHI1,FREQ2,PHI2,SV,I-
LZ,ISN,ITZ,IRN,INSW        ,
       *RNP1,RNP2
     81 FORMAT('1'//' CONSTANT        TAU      FREQ1
PHI1      FREQ        2
       *       PHI2     SMPL INT    ZERO  SMPL  ZERO
RPTS  NSTP   NOISE         P
       *ARAMATERS'/2X,7(1X,G10.4),5(1X,I5),G12.4,G11.4)
        IF(NBS.LT.1) GO TO 84
```

```
        WRITE(N,83) (APS(I),I=1,NBS)
     83 FORMAT(/' ANALYSIS BEFORE SUMMING:  '21A5)
     84 IF(NAS.LT.1) GO TO 100
        WRITE(N,85) (AAS(I),I=1,NAS)
     85 FORMAT(/' ANALYSIS AFTER SUMMING:   '21A5)
        GO TO 100
     90 WRITE(N,91) (HDR(I),I=1,20)
     91 FORMAT('1'/1X,20A4)
        WRITE(N,93) IRS,ITF,ITL,IRN,IAL
     93 FORMAT(/' DATA SET  TIME(MS)-FROM-TO  RECDS
AVGD  TRANSFORM LENGT        H
       *'/5X,I1,I14,I7,I8,I15,I18)
        IF(NBS.LT.1) GO TO 94
        WRITE(N,83) (APS(I),I=1,NBS)
     94 IF(NAS.LT.1) GO TO 100
        WRITE(N,85) (AAS(I),I=1,NAS)
    100 WRITE(N,110) (LC(I),I=1,32)
    110 FORMAT(/' ',32A4/)
        WRITE (N,114) MD
    114 FORMAT(' PLOTTING INDEX =',I3/)
        MAX=-10.0**50
        MIN=10.0**50
        DO 120 I=NB,NE
        IF(A(I).GT.MAX) MAX=A(I)
        IF(A(I).LT.MIN) MIN=A(I)
    120 CONTINUE
        IF(MIN.EQ.0.0.AND.MAX.EQ.0.0) GO TO 900
        IF(FC.LT.0) GO TO 400
        IF(SL.EQ.0.0.AND.SU.EQ.0.0) GO TO 200
        IF(SL.EQ.0.0.AND.SU.NE.0.0) GO TO 300
        GO TO 950
    200 IF(MIN.LT.0.0.AND.MAX.LE.0.0-MIN/200.0)  GO TO
220
        IF(MIN.LT.0.0-MAX/200.0.AND.MAX.GT.0.0-MIN/200-
.0)  GO TO 210
        SL=0.0
        SH=MAX
        SIGN=1.0
        PC=1
        DO 208 I=1,5
    208 VFT(I)=CH(1)
        GO TO 800
    210 MIN=-MIN
        TEMP=AMAX1(MIN,MAX)
        SL=-TEMP
        SH=2.0*TEMP
        SIGN=1.0
        VFT(1)=CH(15)
        VFT(2)=CH(16)
        VFT(3)=CH(5)
        VFT(4)=CH(6)
```

```
        VFT(5)=CH(7)
        PC=2
        GO TO 800
    220 SL=0.0
        SH=-MIN
        SIGN=-1.0
        DO 224 I=1,5
    224 VFT(I)=CH(1)
        PC=1
        GO TO 800
    300 IF(MIN.LT.0.0.AND.MAX.LE.0.0-MIN/200.0)   GO TO
320
        IF(MIN.LT.0.0-MAX/200.0.AND.MAX.GT.0.0-MIN/200-
.0)  GO TO 310
        DO 304 I=1,5
    304 VFT(I)=CH(1)
        SL=0.0
        SH=SU
        SIGN=1.0
        PC=1
        GO TO 800
    310 SL=-SU/2.0
        SH=SU
        SIGN=1.0
        VFT(1)=CH(15)
        VFT(2)=CH(16)
        VFT(3)=CH(5)
        VFT(4)=CH(6)
        VFT(5)=CH(7)
        PC=2
        GO TO 800
    320 SL=0.0
        SH=SU
        SIGN=-1.0
        DO 324 I=1,5
    324 VFT(I)=CH(1)
        PC=1
        GO TO 800
    400 SH=ABS(SU-SL)
        SIGN=1.0
        IF(SL.GT.SU) SIGN=-1.0
        PC=1
        DO 404 I=1,5
    404 VFT(I)=CH(1)
    800 DO 802 I=1,11
    802 VS(I)=SL+SIGN*(I-1)*SH/10.0
        WRITE(N,804) (VS(I),I=1,11)
    804 FORMAT(13X,11G10.3//16X,'I',10('----|----I'))
        DO 805 J=6,16
    805 VF(J)=CH(1)
        NB=NB+(NE-NB)/MD
```

```
      FC=IABS(FC)
      VF(17)=VFT(1)
      VF(18)=VFT(2)
      DO 888 I=NB,NE
      TEMP=SIGN*(A(MD*(I-NB)+NB)-SL)/SH*100.0
      Y=IFIX(TEMP)
      IF(TEMP.GT.100.5) GO TO 807
      IF(TEMP.LT.-0.5) GO TO 808
      IF(TEMP-AINT(TEMP).GE.0.5.AND.Y.LT.100) Y=Y+1
      VF(22)=DG(Y+17)
      VF(24)=CH(17)
      GO TO 809
  807 VF(22)=DG(117)
      VF(24)=CH(18)
      Y=100
      GO TO 809
  808 VF(22)=DG(17)
      VF(24)=CH(19)
  809 VF(2)=CH(1)
      VF(3)=CH(3)
      IF(FC.EQ.2) GO TO 820
      IF(FC.EQ.1) GO TO 840
  810 IF(Y.LE.5) GO TO 812
      VF(6)=DG((Y-1)/5)
      VF(7)=CH(8)
      VF(8)=CH(10)
      VF(9)=CH(11)
      GO TO 814
  812 DO 813 J=6,9
  813 VF(J)=CH(1)
  814 IF(Y.LE.1.OR.(Y-1)/5*5.EQ.(Y-1)) GO TO 816
      VF(10)=DG(Y-(Y-1)/5*5-1)
      VF(11)=CH(12)
      VF(12)=CH(11)
      GO TO 840
  816 DO 818 J=10,12
  818 VF(J)=CH(1)
      GO TO 840
  820 IF(FC.EQ.1) GO TO 840
      IF(Y.GT.51) GO TO 830
      VF(6)=DG(10)
      VF(7)=CH(8)
      VF(8)=CH(10)
      VF(9)=CH(11)
      DO 821 J=10,12
  821 VF(J)=CH(1)
      IF(Y.LT.2) GO TO 822
      VF(13)=CH(13)
      VF(14)=DG(Y-1)
      VF(15)=CH(14)
      VF(16)=CH(11)
```

```
      GO TO 840
  822 DO 824 J=13,16
  824 VF(J)=CH(1)
      GO TO 840
  830 VF(13)=CH(13)
      VF(14)=DG(49)
      VF(15)=CH(14)
      VF(16)=CH(11)
      GO TO 810
  840 IF((I-1)/5*5.EQ.(I-1)) GO TO 850
      VF(5)=CH(5)
      VF(19)=VFT(3)
      GO TO 870
  850 IF(VF(8).EQ.CH(10)) VF(8)=CH(9)
      IF((I-1)/10*10.EQ.(I-1)) GO TO 860
      VF(5)=CH(6)
      VF(19)=VFT(4)
      GO TO 870
  860 VF(5)=CH(7)
      VF(19)=VFT(5)
      VF(2)=CH(2)
      VF(3)=CH(4)
      HS=(MD*(I-NB)+NB-1)*SI+HSOF
      IF(PC.EQ.1.OR.Y.GE.50) GO TO 868
      DO 866 J=17,20
      T=VF(J)
      VF(J)=VF(J+4)
      VF(J+4)=T
  866 CONTINUE
  868 WRITE(N,VF) HS
      GO TO 880
  870 IF(PC.EQ.1.OR.Y.GE.50) GO TO 878
      DO 876 J=17,20
      T=VF(J)
      VF(J)=VF(J+4)
      VF(J+4)=T
  876 CONTINUE
  878 WRITE(N,VF)
  880 CONTINUE
      IF(PC.EQ.1.OR.Y.GE.50) GO TO 888
      DO 882 J=17,20
      T=VF(J)
      VF(J)=VF(J+4)
      VF(J+4)=T
  882 CONTINUE
  888 CONTINUE
      GO TO 1000
  900 WRITE(N,902)
  902 FORMAT(' THE ARRAY CONTAINS ALL ZEROES IN
SUBROUTINE TPLOT.')
      GO TO 1000
```

```
    950 WRITE(6,952)
    952 FORMAT(' INVALID COMBINATION OF VALUES
SPECIFIED FOR SL, SU AND
       * IN SUBROUTINE TPLOT.')
   1000 CONTINUE
        RETURN
        END
  C
  C
        SUBROUTINE FFT(A,B,NTOT,N,NSPAN,ISN)
  C  MULTIVARIATE COMPLEX FOURIER TRANSFORM, COMPUTED
IN PLACE
  C    USING MIXED-RADIX FAST FOURIER TRANSFORM
ALGORITHM.
  C  BY R. C. SINGLETON, STANFORD RESEARCH INSTIUTE,
OCT. 1968
  C  ARRAYS A AND B ORIGINALLY HOLD THE REAL AND
IMAGINARY
  C    COMPONENTS OF THE DATA, AND RETURN THE REAL AND
  C    IMAGINAY COMPONENTS OF THE RESULTING FOURIER
COEFFICIENTS.
  C  MUTIVARIATE DATA IS INDEXD ACCORDING TO THE
FORTRAN
  C    ARRAY ELEMENT SUCCESSOR FUNCTION, WITHOUT LIMIT
  C    ON THE NUMBER OF IMPLIED MULTIPLE SUBSCRIPTS.
  C    THE SUBROUTINE IS CALLED ONCE FOR EACH VARIATE.
  C    THE CALLS FOR A MULTIVARIATE TRANSFORM MAY BE IN
ANY ORDER.
  C  NTOT IS THE TOTAL NUMBER OF COMPLEX DATA VALUES.
  C  N IS THE DIMENSION OF THE CURRENT VARIABLE.
  C  NSPAN/N IS THE SPACING OF CONSECUTIVE DATA VALUES
  C    WHILE INDEXING THE CURRENT VARIABLE.
  C  THE SIGN OF ISN DETERMINES THE SIGN OF THE COMPLEX
  C    EXPONENTIAL, AND THE MAGNITUDE OF ISN IS
NORMALLY ONE.
  C  A TRI-VARIATE TRANSFORM WITH A(N1,N2,N3),
B(N1,N2,N3)
  C    IS COMPUTED BY
  C      CALL FFT(A,B,N1*N2*N3,N1,N1,1)
  C      CALL FFT(A,B,N1*N2*N3,N2,N1*N2,1)
  C      CALL FFT(A,B,N1*N2*N3,N3,N1*N2*N3,1)
  C  FOR A SINGLE VARIATE TRANSFORM,
  C    NTOT = NSPAN = (NUMBER OF COMPLEX DATA VALUES),
E.G.
  C      CALL FFT(A,B,N,N,N,1)
  C  THE DATA MAY ALTERNATIVELY BE STORED IN A SINGLE
COMPLEX
  C    ARRAY A, THEN THE MAGNITUDE OF ISN CHANGED TO
TWO TO
```

```
  C    GIVE THE CORRECT INDEXING INCREMENT AND A(2)
USED TO
  C    PASS THE INITIAL ADDRESS FOR THE SEQUENCE OF
IMAGINARY
  C    VALUES, E.G.
  C      CALL FFT(A,A(2),NTOT,N,NSPAN,2)
  C  ARRAYS AT(MAXF), CK(MAXF), BT(MAXF), SK(MAXF), AND
NP(MAXP)
  C    ARE USD FOR TEMPORARY STORAGE. IF THE AVAILABLE
STORAGE
  C    IS INSUFFICIENT, THE PROGRAM IS TERMINATED BY A
STOP.
  C    MAXF MUST BE .GE. THE MAXIMUM PRIME FACTOR OF N.
  C    IN ADDITION, IF THE SQUARE-FREE PORTION K OF N
HAS TWO OR
  C    MORE PRIME FACTORS, THEN MAXP MUST BE .GE. K-1.
         DIMENSION A(N),B(N)
  C  ARRAY STORAGE IN NFAC FOR A MAXIMUM OF 11 FACTORS
OF N.
  C  IF N HAS MORE THAN ONE SQUARE-FREE FACTOR, THE
PRODUCT OF THE
  C    SQUARE-FREE FACTORS MUST BE .LE. 210
         DIMENSION NFAC(11),NP(209)
  C  ARRAY STORAGE FOR MAXIMUM PRIME FACTOR OF 23
         DIMENSION AT(23),CK(23),BT(23),SK(23)
         EQUIVALENCE (I,II)
  C  THE FOLLOWING TWO CONSTANTS SHOULD AGREE WITH THE
ARRAY DIMENSIONS.
         MAXF=23
         MAXP=209
         IF(N .LT. 2) RETURN
         INC=ISN
         RAD=8.0*ATAN(1.0)
         S72=RAD/5.0
         C72=COS(S72)
         S72=SIN(S72)
         S120=SQRT(0.75)
         IF(ISN .GE. 0) GO TO 10
         S72=-S72
         S120=-S120
         RAD=-RAD
         INC=-INC
      10 NT=INC*NTOT
         KS=INC*NSPAN
         KSPAN=KS
         NN=NT-INC
         JC=KS/N
         RADF=RAD*FLOAT(JC)*0.5
         I=0
         JF=0
  C  DETERMINE THE FACTORS OF N
```

```
        M=0
        K=N
        GO TO 20
     15 M=M+1
        NFAC(M)=4
        K=K/16
     20 IF(K-(K/16)*16 .EQ. 0) GO TO 15
        J=3
        JJ=9
        GO TO 30
     25 M=M+1
        NFAC(M)=J
        K=K/JJ
     30 IF(MOD(K,JJ) .EQ. 0) GO TO 25
        J=J+2
        JJ=J**2
        IF(JJ .LE. K) GO TO 30
        IF(K .GT. 4) GO TO 40
        KT=M
        NFAC(M+1)=K
        IF(K .NE. 1) M=M+1
        GO TO 80
     40 IF(K-(K/4)*4 .NE. 0) GO TO 50
        M=M+1
        NFAC(M)=2
        K=K/4
     50 KT=M
        J=2
     60 IF(MOD(K,J) .NE. 0) GO TO 70
        M=M+1
        NFAC(M)=J
        K=K/J
     70 J=((J+1)/2)*2+1
        IF(J .LE. K) GO TO 60
     80 IF(KT .EQ. 0) GO TO 100
        J=KT
     90 M=M+1
        NFAC(M)=NFAC(J)
        J=J-1
        IF(J .NE. 0) GO TO 90
  C  COMPUTE FOURIER TRANSFORM
    100 SD=RADF/FLOAT(KSPAN)
        CD=2.0*SIN(SD)**2
        SD=SIN(SD+SD)
        KK=1
        I=I+1
        IF(NFAC(I) .NE. 2) GO TO 400
  C  TRANSFORM FOR FACTOR OF 2 (INCLUDING ROTATION
FACTOR)
        KSPAN=KSPAN/2
        K1=KSPAN+2
```

```
    210 K2=KK+KSPAN
        AK=A(K2)
        BK=B(K2)
        A(K2)=A(KK)-AK
        B(K2)=B(KK)-BK
        A(KK)=A(KK)+AK
        B(KK)=B(KK)+BK
        KK=K2+KSPAN
        IF(KK .LE. NN) GO TO 210
        KK=KK-NN
        IF(KK .LE. JC) GO TO 210
        IF(KK .GT. KSPAN) GO TO 800
    220 C1=1.0-CD
        S1=SD
    230 K2=KK+KSPAN
        AK=A(KK)-A(K2)
        BK=B(KK)-B(K2)
        A(KK)=A(KK)+A(K2)
        B(KK)=B(KK)+B(K2)
        A(K2)=C1*AK-S1*BK
        B(K2)=S1*AK+C1*BK
        KK=K2+KSPAN
        IF(KK .LT. NT) GO TO 230
        K2=KK-NT
        C1=-C1
        KK=K1-K2
        IF(KK .GT. K2) GO TO 230
        AK=C1-(CD*C1+SD*S1)
        S1=(SD*C1-CD*S1)+S1
  C  THE FOLLOWING THREE STATEMENTS COMPENSATE FOR
TRUNCATION
  C    ERROR. IF ROUNDED ARITHMETIC IS USED, SUBSTITUTE
  C     C1=AK
        C1=0.5/(AK**2+S1**2)+0.5
        S1=C1*S1
        C1=C1*AK
        KK=KK+JC
        IF(KK .LT. K2) GO TO 230
        K1=K1+INC+INC
        KK=(K1-KSPAN)/2+JC
        IF(KK .LE. JC+JC) GO TO 220
        GO TO 100
  C  TRANSFORM FOR FACTOR OF 3 (OPTIONAL CODE)
    320 K1=KK+KSPAN
        K2=K1+KSPAN
        AK=A(KK)
        BK=B(KK)
        AJ=A(K1)+A(K2)
        BJ=B(K1)+B(K2)
        A(KK)=AK+AJ
        B(KK)=BK+BJ
```

```
      AK=-0.5*AJ+AK
      BK=-0.5*BJ+BK
      AJ=(A(K1)-A(K2))*S120
      BJ=(B(K1)-B(K2))*S120
      A(K1)=AK-BJ
      B(K1)=BK+AJ
      A(K2)=AK+BJ
      B(K2)=BK-AJ
      KK=K2+KSPAN
      IF(KK .LT. NN) GO TO 320
      KK=KK-NN
      IF(KK .LE. KSPAN) GO TO 320
      GO TO 700
C  TRANSFORM FOR FACTOR OF 4
  400 IF(NFAC(I) .NE. 4) GO TO 600
      KSPNN=KSPAN
      KSPAN=KSPAN/4
  410 C1=1.0
      S1=0
  420 K1=KK+KSPAN
      K2=K1+KSPAN
      K3=K2+KSPAN
      AKP=A(KK)+A(K2)
      AKM=A(KK)-A(K2)
      AJP=A(K1)+A(K3)
      AJM=A(K1)-A(K3)
      A(KK)=AKP+AJP
      AJP=AKP-AJP
      BKP=B(KK)+B(K2)
      BKM=B(KK)-B(K2)
      BJP=B(K1)+B(K3)
      BJM=B(K1)-B(K3)
      B(KK)=BKP+BJP
      BJP=BKP-BJP
      IF(ISN .LT. 0) GO TO 450
      AKP=AKM-BJM
      AKM=AKM+BJM
      BKP=BKM+AJM
      BKM=BKM-AJM
      IF(S1 .EQ. 0.0) GO TO 460
  430 A(K1)=AKP*C1-BKP*S1
      B(K1)=AKP*S1+BKP*C1
      A(K2)=AJP*C2-BJP*S2
      B(K2)=AJP*S2+BJP*C2
      A(K3)=AKM*C3-BKM*S3
      B(K3)=AKM*S3+BKM*C3
      KK=K3+KSPAN
      IF(KK .LE. NT) GO TO 420
  440 C2=C1-(CD*C1+SD*S1)
      S1=(SD*C1-CD*S1)+S1
```

```
  C  THE FOLLOWING THREE STATEMENTS COMPENSATE FOR
TRUNCATION
  C    ERROR. IF ROUNDED ARITHMETIC IS USED, SUBSTITUTE
  C     C1=C2
        C1=0.5/(C2**2+S1**2)+0.5
        S1=C1*S1
        C1=C1*C2
        C2=C1**2-S1**2
        S2=2.0*C1*S1
        C3=C2*C1-S2*S1
        S3=C2*S1+S2*C1
        KK=KK-NT+JC
        IF(KK .LE. KSPAN) GO TO 420
        KK=KK-KSPAN+INC
        IF(KK .LE. JC) GO TO 410
        IF(KSPAN .EQ. JC) GO TO 800
        GO TO 100
    450 AKP=AKM+BJM
        AKM=AKM-BJM
        BKP=BKM-AJM
        BKM=BKM+AJM
        IF(S1 .NE. 0.0) GO TO 430
    460 A(K1)=AKP
        B(K1)=BKP
        A(K2)=AJP
        B(K2)=BJP
        A(K3)=AKM
        B(K3)=BKM
        KK=K3+KSPAN
        IF(KK .LE. NT) GO TO 420
        GO TO 440
  C  TRANSFORM FOR FACTOR OF 5 (OPTIONAL CODE)
    510 C2=C72**2-S72**2
        S2=2.0*C72*S72
    520 K1=KK+KSPAN
        K2=K1+KSPAN
        K3=K2+KSPAN
        K4=K3+KSPAN
        AKP=A(K1)+A(K4)
        AKM=A(K1)-A(K4)
        BKP=B(K1)+B(K4)
        BKM=B(K1)-B(K4)
        AJP=A(K2)+A(K3)
        AJM=A(K2)-A(K3)
        BJP=B(K2)+B(K3)
        BJM=B(K2)-B(K3)
        AA=A(KK)
        BB=B(KK)
        A(KK)=AA+AKP+AJP
        B(KK)=BB+BKP+BJP
        AK=AKP*C72+AJP*C2+AA
```

```
      BK=BKP*C72+BJP*C2+BB
      AJ=AKM*S72+AJM*S2
      BJ=BKM*S72+BJM*S2
      A(K1)=AK-BJ
      A(K4)=AK+BJ
      B(K1)=BK+AJ
      B(K4)=BK-AJ
      AK=AKP*C2+AJP*C72+AA
      BK=BKP*C2+BJP*C72+BB
      AJ=AKM*S2-AJM*S72
      BJ=BKM*S2-BJM*S72
      A(K2)=AK-BJ
      A(K3)=AK+BJ
      B(K2)=BK+AJ
      B(K3)=BK-AJ
      KK=K4+KSPAN
      IF(KK .LT. NN) GO TO 520
      KK=KK-NN
      IF(KK .LE. KSPAN) GO TO 520
      GO TO 700
C  TRANSFORM FOR ODD FACTORS
  600 K=NFAC(I)
      KSPNN=KSPAN
      KSPAN=KSPAN/K
      IF(K .EQ. 3) GO TO 320
      IF(K .EQ. 5) GO TO 510
      IF(K .EQ. JF) GO TO 640
      JF=K
      S1=RAD/FLOAT(K)
      C1=COS(S1)
      S1=SIN(S1)
      IF(JF .GT. MAXF) GO TO 998
      CK(JF)=1.0
      SK(JF)=0.0
      J=1
  630 CK(J)=CK(K)*C1+SK(K)*S1
      SK(J)=CK(K)*S1-SK(K)*C1
      K=K-1
      CK(K)=CK(J)
      SK(K)=-SK(J)
      J=J+1
      IF(J .LT. K) GO TO 630
  640 K1=KK
      K2=KK+KSPAN
      AA=A(KK)
      BB=B(KK)
      AK=AA
      BK=BB
      J=1
      K1=K1+KSPAN
  650 K2=K2-KSPAN
```

```
      J=J+1
      AT(J)=A(K1)+A(K2)
      AK=AT(J)+AK
      BT(J)=B(K1)+B(K2)
      BK=BT(J)+BK
      J=J+1
      AT(J)=A(K1)-A(K2)
      BT(J)=B(K1)-B(K2)
      K1=K1+KSPAN
      IF(K1 .LT. K2) GO TO 650
      A(KK)=AK
      B(KK)=BK
      K1=KK
      K2=KK+KSPAN
      J=1
  660 K1=K1+KSPAN
      K2=K2-KSPAN
      JJ=J
      AK=AA
      BK=BB
      AJ=0.0
      BJ=0.0
      K=1
  670 K=K+1
      AK=AT(K)*CK(JJ)+AK
      BK=BT(K)*CK(JJ)+BK
      K=K+1
      AJ=AT(K)*SK(JJ)+AJ
      BJ=BT(K)*SK(JJ)+BJ
      JJ=JJ+J
      IF(JJ .GT. JF) JJ=JJ-JF
      IF(K .LT. JF) GO TO 670
      K=JF-J
      A(K1)=AK-BJ
      B(K1)=BK+AJ
      A(K2)=AK+BJ
      B(K2)=BK-AJ
      J=J+1
      IF(J .LT. K) GO TO 660
      KK=KK+KSPAN
      IF(KK .LE. NN) GO TO 640
      KK=KK-NN
      IF(KK .LE. KSPAN) GO TO 640
C  MULTIPLY BY ROTATION FACTOR (EXCOPT FOR FACTORS 2
AND 4)
  700 IF(I .EQ. M) GO TO 800
      KK=JC+1
  710 C2=1.0-CD
      S1=SD
  720 C1=C2
      S2=S1
```

```
        KK=KK+KSPAN
    730 AK=A(KK)
        A(KK)=C2*AK-S2*B(KK)
        B(KK)=S2*AK+C2*B(KK)
        KK=KK+KSPNN
        IF(KK .LE. NT) GO TO 730
        AK=S1*S2
        S2=S1*C2+C1*S2
        C2=C1*C2-AK
        KK=KK-NT+KSPAN
        IF(KK .LE. KSPNN) GO TO 730
        C2=C1-(CD*C1+SD*S1)
        S1=S1+(SD*C1-CD*S1)
  C  THE FOLLOWING THREE STATEMENTS COMPENSATE FOR
TRUNCATION
  C    ERROR. IF ROUNDED ARITHMETIC IS USED, THEY MAY
  C    BE DELETED.
        C1=0.5/(C2**2+S1**2)+0.5
        S1=C1*S1
        C2=C1*C2
        KK=KK-KSPNN+JC
        IF(KK .LE. KSPAN) GO TO 720
        KK=KK-KSPAN+JC+INC
        IF(KK .LE. JC+JC) GO TO 710
        GO TO 100
  C  PERMUTE THE RESUTS TO NORMAL ORDER---DONE IN TWO
STAGES
  C  PERMUTATION FOR SQUARE FACTORS OF N
    800 NP(1)=KS
        IF(KT .EQ. 0) GO TO 890
        K=KT+KT+1
        IF(M .LT. K) K=K-1
        J=1
        NP(K+1)=JC
    810 NP(J+1)=NP(J)/NFAC(J)
        NP(K)=NP(K+1)*NFAC(J)
        J=J+1
        K=K-1
        IF(J .LT. K) GO TO 810
        K3=NP(K+1)
        KSPAN=NP(2)
        KK=JC+1
        K2=KSPAN+1
        J=1
        IF(N .NE. NTOT) GO TO 850
  C  PERMUTATION FOR SINGLE-VARIATE TRANSFORM (OPTIONAL
CODE)
    820 AK=A(KK)
        A(KK)=A(K2)
        A(K2)=AK
        BK=B(KK)
```

```
      B(KK)=B(K2)
      B(K2)=BK
      KK=KK+INC
      K2=KSPAN+K2
      IF(K2 .LT. KS) GO TO 820
  830 K2=K2-NP(J)
      J=J+1
      K2=NP(J+1)+K2
      IF(K2 .GT. NP(J)) GO TO 830
      J=1
  840 IF(KK .LT. K2) GO TO 820
      KK=KK+INC
      K2=KSPAN+K2
      IF(K2 .LT. KS) GO TO 840
      IF(KK .LT. KS) GO TO 830
      JC=K3
      GO TO 890
C  PERMUTATION FOR MULTIVARIATE TRANSFORM
  850 K=KK+JC
  860 AK=A(KK)
      A(KK)=A(K2)
      A(K2)=AK
      BK=B(KK)
      B(KK)=B(K2)
      B(K2)=BK
      KK=KK+INC
      K2=K2+INC
      IF(KK .LT. K) GO TO 860
      KK=KK+KS-JC
      K2=K2+KS-JC
      IF(KK .LT. NT) GO TO 850
      K2=K2-NT+KSPAN
      KK=KK-NT+JC
      IF(K2 .LT. KS) GO TO 850
  870 K2=K2-NP(J)
      J=J+1
      K2=NP(J+1)+K2
      IF(K2 .GT. NP(J)) GO TO 870
      J=1
  880 IF(KK .LT. K2) GO TO 850
      KK=KK+JC
      K2=KSPAN+K2
      IF(K2 .LT. KS) GO TO 880
      IF(KK .LT. KS) GO TO 870
      JC=K3
  890 IF(2*KT+1 .GE. M) RETURN
      KSPNN=NP(KT+1)
C  PERMUTATION FOR SQUARE-FREE FACTORS OF N
      J=M-KT
      NFAC(J+1)=1
  900 NFAC(J)=NFAC(J)*NFAC(J+1)
```

```
      J=J-1
      IF(J .NE. KT) GO TO 900
      KT=KT+1
      NN=NFAC(KT)-1
      IF(NN .GT. MAXP) GO TO 998
      JJ=0
      J=0
      GO TO 906
  902 JJ=JJ-K2
      K2=KK
      K=K+1
      KK=NFAC(K)
  904 JJ=KK+JJ
      IF(JJ .GE. K2) GO TO 902
      NP(J)=JJ
  906 K2=NFAC(KT)
      K=KT+1
      KK=NFAC(K)
      J=J+1
      IF(J .LE. NN) GO TO 904
C  DETERMINE THE PERMUTATION CYCLES OF LENGTH GREATER
THAN 1
      J=0
      GO TO 914
  910 K=KK
      KK=NP(K)
      NP(K)=-KK
      IF(KK .NE. J) GO TO 910
      K3=KK
  914 J=J+1
      KK=NP(J)
      IF(KK .LT. 0) GO TO 914
      IF(KK .NE. J) GO TO 910
      NP(J)=-J
      IF(J .NE. NN) GO TO 914
      MAXF=INC*MAXF
C  REORDER A AND B, FOLLOWING THE PERMUTATION CYCLES
      GO TO 950
  924 J=J-1
      IF(NP(J) .LT. 0) GO TO 924
      JJ=JC
  926 KSPAN=JJ
      IF(JJ .GT. MAXF) KSPAN=MAXF
      JJ=JJ-KSPAN
      K=NP(J)
      KK=JC*K+II+JJ
      K1=KK+KSPAN
      K2=0
  928 K2=K2+1
      AT(K2)=A(K1)
      BT(K2)=B(K1)
```

```
        K1=K1-INC
        IF(K1 .NE. KK) GO TO 928
    932 K1=KK+KSPAN
        K2=K1-JC*(K+NP(K))
        K=-NP(K)
    936 A(K1)=A(K2)
        B(K1)=B(K2)
        K1=K1-INC
        K2=K2-INC
        IF(K1 .NE. KK) GO TO 936
        KK=K2
        IF(K .NE. J) GO TO 932
        K1=KK+KSPAN
        K2=0
    940 K2=K2+1
        A(K1)=AT(K2)
        B(K1)=BT(K2)
        K1=K1-INC
        IF(K1 .NE. KK) GO TO 940
        IF(JJ .NE. 0) GO TO 926
        IF(J .NE. 1) GO TO 924
    950 J=K3+1
        NT=NT-KSPNN
        II=NT-INC+1
        IF(NT .GE. 0) GO TO 924
        RETURN
  C  ERROR FINISH, INSUFFICIENT ARRAY STORAGE
    998 ISN=0
        WRITE(6,999)
        RETURN
    999 FORMAT(44H ARRAY BOUNDS EXCEEDED WITHIN
SUBROUTINE FFT)
        END
  C
  C
        SUBROUTINE REALTR(A,B,N,ISN)
  C  IF ISN=1, THIS SUBROUTINE COMPLETES THE FOURIER
TRANSFORM
  C    OF 2*N REAL DATA VALUES, WHERE THE ORIGINAL DATA
VALUES ARE
  C    STORED ALTERNATELY IN ARRAYS A AND B, AND ARE
FIRST
  C    TRANSFORMED BY A COMPLEX FOURIER TRANSFORM OF
DIMENSION N.
  C    THE COSINE COEFFICIENTS ARE IN
A(1),A(2),...A(N+1) AND
  C    THE SINE COEFFICIENTS ARE IN
B(1),B(2),...B(N+1).
  C    A TYPICAL CALLING SEQUENCE IS
  C      CALL FFT(A,B,N,N,N,1)
  C      CALL REALTR(A,B,N,1)
```

```
  C    THE RESULTS SHOULD BE MULTIPLIED BY 0.5/N TO
GIVE THE
  C    USUAL SCALING OF COEFFICIENTS.
  C  IF ISN=-1, THE INVERSE TRANSFORM IS DONE, THE
FIRST STEP
  C    IN EVALUATING A REAL FOURIER SERIES.
  C    A TYPICAL CALLING SEQUENCE IS
  C      CALL REALTR(A,B,N,-1)
  C      CALL FFT(A,B,N,N,N,-1)
  C    THE RESULTS SHOULD BE MULTIPLIED BY 0.5 TO GIVE
THE USUAL
  C    SCALING, AND THE TIME DOMAIN RESULTS ALTERNATE
IN ARRAYS A
  C    AND B, I.E. A(1),B(1),A(2),B(2),...A(N),B(N).
  C  THE DATA MAY ALTERNATIVELY BE STORED IN A SINGLE
COMPLEX
  C    ARRAY A, THEN THE MAGNITUDE OF ISN CHANGED TO
TWO TO
  C    GIVE THE CORRECT INDEXING INCREMENT AND A(2)
USED TO
  C    PASS THE INITIAL ADDRESS FOR THE SEQUENCE OF
IMAGINARY
  C    VALUES, E.G.
  C      CALL FFT(A,A(2),N,N,N,2)
  C      CALL REALTR(A,A(2),N,2)
  C    IN THIS CASE, THE COSINE AND SINE COEFFICIENTS
ALTERNATE IN A.
  C  BY R. C. SINGLETON, STANFORD RESEARCH INSTITUTE,
OCT. 1968
        DIMENSION A(1),B(1)
        REAL IM
        INC=IABS(ISN)
        NK=N*INC+2
        NH=NK/2
        SD=2.0*ATAN(1.0)/FLOAT(N)
        CD=2.0*SIN(SD)**2
        SD=SIN(SD+SD)
        SN=0.0
        IF(ISN .LT. 0) GO TO 30
        CN=1.0
        A(NK-1)=A(1)
        B(NK-1)=B(1)
     10 DO 20 J=1,NH,INC
        K=NK-J
        AA=A(J)+A(K)
        AB=A(J)-A(K)
        BA=B(J)+B(K)
        BB=B(J)-B(K)
        RE=CN*BA+SN*AB
        IM=SN*BA-CN*AB
        B(K)=IM-BB
```

```
        B(J)=IM+BB
        A(K)=AA-RE
        A(J)=AA+RE
        AA=CN-(CD*CN+SD*SN)
        SN=(SD*CN-CD*SN)+SN
  C  THE FOLLOWING THREE STATEMENTS COMPENSATE FOR
TRUNCATION
  C     ERROR. IF ROUNDED ARITHMETIC IS USED,
SUBSTITUTE
  C   20 CN=AA
        CN=0.5/(AA**2+SN**2)+0.5
        SN=CN*SN
     20 CN=CN*AA
        RETURN
     30 CN=-1.0
        SD=-SD
        GO TO 10
        END
```

APPENDIX 2

VERSATEC PLOTTING PROGRAM LISTING

In the following listing, lines which were too long were broken in two by the text editor. The second part of any broken line begins two spaces to the left of FORTRAN column one.

```
C     VERSATEC PLOTTING PROGRAM FOR USE WITH:
C   INTERACTIVE PHONOCARDIOGRAM ANALYSIS PROGRAM,
VERSION 1.02 04/78
C      BY S. J. SHOWALTER, VIRGINIA POLYTECHNIC
INSTITUTE
C      & STATE UNIVERSITY, BLACKSBURG, VA.  APRIL
1978
        INTEGER CAL,N5(20),N6(6),N7(42)
        DATA LMASK1,LMASK2/ZFFFFFFFF,Z03030303/
        REAL D(327),X(327),Y(4),VS(6),HS(7)
        EX=1.0
        CALL PLOTS(0,0,0)
C   FOR 35% REDUCTION USE A FACTOR OF 1.53846
        CALL FACTOR(1.0)
        CALL PLOT(0.0,0.0,-3)
        CALL PLOT(0.0,7.0,2)
    10 READ(5,11,END=80)ISEED,CAL,IXL,XSI,
      *(D(I),I=1,327),IYSUM,(N5(I),I=1
      *,20),(N6(I),I=1,6),(N7(I),I=1,42)
     11 FORMAT(20A4)
        XSI=XSI/EX
        NP=325.0/EX
        CALL GRID(2.5,1.75,1,7.25,1,4.5,LMASK1)
        CALL GRID(2.5,1.75,13,0.558,10,0.45,LMASK2)
C
        DH=-10000.0
        DL=10000.0
        DO 34 I=1,NP
        IF(D(I).GT.DH) DH=D(I)
     34 IF(D(I).LT.DL) DL=D(I)
        IF(DL.GT.-0.01*DH) GO TO 40
C
        DS=2.0*DH
        IF(ABS(DL).GT.ABS(DH)) DS=2.0*ABS(DL)
        DO 36 I=1,NP
     36 D(I)=D(I)/DS*4.5+4.0
```

```
      DO 38 I=1,6
    38 VS(I)=(FLOAT(I)-3.5)*0.2*DS
      CALL PLOT(2.5,4.0,3)
      CALL PLOT(9.75,4.0,2)
      GO TO 50
  C
    40 CONTINUE
      DO 42 I=1,NP
    42 D(I)=D(I)/DH*4.5+1.75
      DO 44 I=1,6
    44 VS(I)=FLOAT(I-1)*DH/5.0
  C
    50 CONTINUE
      DX=7.25/FLOAT(NP-1)
      DO 52 I=1,NP
    52 X(I)=FLOAT(I-1)*DX+2.5
      DO 54 I=1,7
    54 HS(I)=FLOAT(I-1)*XSI*50.0
      CALL LINE(X,D,NP,1)
  C
      H=0.1
      XX=2.4
      ND=-1
      IF(HS(2).LT.1.0) ND=3
      DO 62 I=1,7
      CALL NUMBER(XX,1.6,H,HS(I),0.0,ND)
    62 XX=XX+1.112
      IF(HS(2).LT.1.0) GO TO 64
      CALL SYMBOL(5.48,1.45,H,14HFREQUENCY
(HZ),0.0,14)
      GO TO 66
    64 CALL SYMBOL(5.73,1.45,H,9HTIME (MS),0.0,9)
    66 CONTINUE
      YY=1.7
      DO 68 I=1,6
      CALL NUMBER(1.6,YY,H,VS(I),0.0,5)
    68 YY=YY+0.9
      CALL SYMBOL(1.4,3.0,H,20HNORMALIZED
AMPLITUDE,90.0,20)
  C
      CALL SYMBOL(1.4,6.9,H,N5(1),0.0,40)
      YY=6.75
      CALL SYMBOL(1.4,YY,H,8HDATA SET,0.0,8)
      FN=FLOAT(N7(25))
      CALL NUMBER(2.4,YY,H,FN,0.0,-1)
      FN=FLOAT(N7(26))
      CALL NUMBER(2.75,YY,H,FN,0.0,-1)
      CALL SYMBOL(3.1,YY,H,2HTO,0.0,2)
      FN=FLOAT(N7(27))
      CALL NUMBER(3.4,YY,H,FN,0.0,-1)
      CALL SYMBOL(3.77,YY,H,2HMS,0.0,2)
```

```
      FN=FLOAT(N7(28))
      CALL NUMBER(4.17,YY,H,FN,0.0,-1)
      CALL SYMBOL(4.45,YY,H,9HRCDS AVGD,0.0,9)
      CALL SYMBOL(1.45,6.525,H,9HANALYSIS:,0.0,9)
      CALL SYMBOL(2.55,6.6,H,8HPRE  SUM,0.0,8)
      XX=3.6
      IE=N7(22)
      DO 72 I=1,IE
      CALL SYMBOL(XX,6.6,H,N7(I+1),0.0,4)
   72 XX=XX+0.51
      CALL SYMBOL(2.55,6.45,H,8HPOST SUM,0.0,8)
      IF(N7(23).LT.1) GO TO 76
      XX=3.6
      IE=N7(23)
      DO 74 I=1,IE
      CALL SYMBOL(XX,6.45,H,N7(I+11),0.0,4)
   74 XX=XX+0.51
C
   76 CALL PLOT(11.0,10.0,3)
      CALL PLOT(11.0,9.5,2)
      CALL PLOT(11.0,0.25,3)
      CALL PLOT(11.0,0.0,2)
      CALL PLOT(11.0,0.0,-3)
      GO TO 10
   80 CALL PLOT(0.0,0.0,999)
      STOP
      END
C
      SUBROUTINE LINE(X,Y,N,K)
      DIMENSION X(1),Y(1)
C  CALCULATE TOTAL NUMBER OF FULL WORDS
      J=N*IABS(K)
C  DETERMINE INITIAL PEN POSITION (UP OR DOWN)
      IF(K) 1,1,2
C  INITIAL PEN POSITION IS UP
2     CALL PLOT(X(1),Y(1),3)
1     IPEN=2
      KK=IABS(K)
C  DRAW THE LINE
      DO 10 I=1,J,KK
      CALL PLOT(X(I),Y(I),IPEN)
10    CONTINUE
      RETURN
      END
/*
//GO.SYSIN  DD  *
```

AN INTERACTIVE COMPUTER ANALYSIS
OF THE AORTIC EJECTION CLICK AND FIRST HEART SOUND

by

Samuel Joseph Showalter

(ABSTRACT)

The aortic ejection click and the first heart sound (S1) are described in the time and frequency domains from computer processed phonocardiogram data obtained from 13 valvar aortic stenosis patients and 6 normal ones. The interactive FORTRAN program developed was capable of the following: (1) computing averages, aligned averages, power spectra of selected segments, envelograms, and determinstic signals; (2) storing and retrieving intermediate results; and (3) plotting data. The program is listed and its interaction is fully described in this thesis. The following steps were used to analyze the data: (1) the determination of the click onset times; (2) the generation of aligned average cardiocycles for both the click and S1; and (3) the calculation of power spectra for selected sgments of the click and S1. The tabulated results typically show 2 frequencies in S1 and 3 in the click, with the click being distinguished by an oscillation between 160 and 230 Hz. Characteristics of the click versus time are also tabulated.