

AD-754 263

CIRCUIT AND RADIATION EFFECTS ANALYSIS
BY ITERATED PROPAGATION OF BIVARIABLE
RESPONSE FUNCTIONS

Robert G. Stewart, et al

Stewart Research Enterprises

Prepared for:

Air Force Weapons Laboratory

November 1972

DISTRIBUTED BY:

NTIS

National Technical Information Service
U. S. DEPARTMENT OF COMMERCE
5285 Port Royal Road, Springfield Va. 22151

AD754263



CIRCUIT AND RADIATION EFFECTS ANALYSIS BY ITERATED PROPAGATION OF BIVARIABLE RESPONSE FUNCTIONS

Robert G. Stewart

Peter S. Castro

Stewart Research Enterprises

TECHNICAL REPORT NO. AFWL-TR-72-76

November 1972

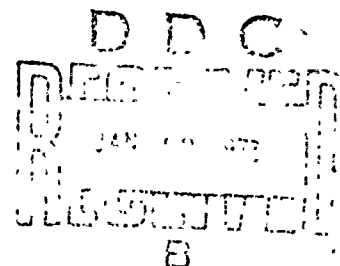
AIR FORCE WEAPONS LABORATORY

Air Force Systems Command

Kirtland Air Force Base

New Mexico

Reproduced by
**NATIONAL TECHNICAL
INFORMATION SERVICE**
U S Department of Commerce
Springfield VA 22151



Approved for public release; distribution unlimited.

ACCESSION for	
NTIS	White Section ☒
DDC	Ball Section ☐
UNCLASSIFIED	☐
JUSTIFICATION	
BY	
UNCLASSIFIED AVAILABLE COPIES	
DATE	
A	

AIR FORCE WEAPONS LABORATORY
Air Force Systems Command
Kirtland Air Force Base
New Mexico 87117

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related Government procurement operation, the Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

DO NOT RETURN THIS COPY. RETAIN OR DESTROY.

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Stewart Research Enterprises 23376 Belvoir Drive Los Altos, California 94022		2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	
		2b. GROUP	
3. REPORT TITLE CIRCUIT AND RADIATION EFFECTS ANALYSIS BY ITERATED PROPAGATION OF BIVARIABLE RESPONSE FUNCTIONS			
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) February 1971 through April 1972			
5. AUTHOR(S) (First name, middle initial, last name) Robert G. Stewart and Peter S. Castro			
6. REPORT DATE November 1972		7a. TOTAL NO. OF PAGES 258 260	7b. NO. OF REFS ---
8a. CONTRACT OR GRANT NO F29601-71-C-0039		9a. ORIGINATOR'S REPORT NUMBER(S) AFWL-TR-72-76	
b. PROJECT NO 5710			
c. Subtask TB027		9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	
d.			
10. DISTRIBUTION STATEMENT Approved for public release; distribution unlimited.			
11. SUPPLEMENTARY NOTES Details of illustrations in this document may be better studied on microfiche		12. SPONSORING MILITARY ACTIVITY AFWL (ELT) Kirtland AFB, NM 87117	
13. ABSTRACT (Distribution Limitation Statement A) Radiation effects analysis of complex electronic circuits by computers presently is limited by core storage and long execution times. A new method, entitled Iterated Propagation of Bivariable Response Functions, treats components in the block sense using experimentally obtained time domain response or transfer functions. These functions are described in a three dimensional space of the input stimuli, output response, and time, by means of fitting the experimental data with cubic polynomials matched in the spline sense at interstices of the fitting grid. The basic data was obtained using conventional oscilloscopic techniques at the Kirtland AFB, flash X-ray facility and the Sandia pulse reactor. A computer program called SAP, adapted to work with SCEPTRE, uses a modified convolution process to calculate the electrical and the gamma and neutron radiation responses from the surface descriptions. These responses are then superimposed to obtain the combined response. Such a process is not rigorous but rather an engineering class of approximation. The results achieved ranged from fair to excellent, as applied to a 741 operational amplifier and a 9704 dual NAND gate. The three dimensional surfaces provide a vivid grasp of device performance and radiation response over the entire range of the stimulus variables.			

DD FORM 1 NOV 65 1473

UNCLASSIFIED

Security Classification

CIRCUIT AND RADIATION EFFECTS ANALYSIS BY ITERATED PROPAGATION OF
BIVARIABLE RESPONSE FUNCTIONS

Robert G. Stewart
Peter S. Castro

Stewart Research Enterprises

TECHNICAL REPORT NO. AFWL-TR-72-76

Approved for public release; distribution unlimited.

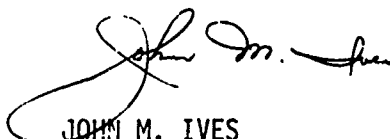
FOREWORD

This report was prepared by Stewart Research Enterprises, Los Altos, California, under Contract F29601-71-C-0039. The research was performed under Program Element 61102H, Project 5710, Subtask TB027, and was funded by the Defense Nuclear Agency (DNA) under MIPR 555-70.

Inclusive dates of research were February 1971 through April 1972. The report was submitted 1 September 1972 by the Air Force Weapons Laboratory Project Officer, Captain John M. Ives (ELT).

Dr. Robert G. Stewart was principal investigator. Dr. Peter S. Castro participated in all aspects of the work. The original technical motivation for this effort evolved from discussions with Dr. Julian Nichols and Captain John Anderson of AFWL. The radiation exposure jigs used at the Kirtland flash X ray and the Sandia pulse reactor were designed and built by Mr. Robert Youden. Mr. Robert Marshall of Fairchild Semiconductor and Captain Patrick Vail of AFWL helped in planning the device radiation exposures. The tests were conducted by Dr. Stewart and Mr. Youden with the assistance of Captain John Ives, AFWL project officer. Mr. Siu Kee Lee participated with Dr. Stewart and Dr. Castro in the tedious work of making the first three-dimensional plots. Mr. David Schrodi worked diligently in programming and debugging the computer codes. Mr. David A. Johnson lent valuable assistance with some of the more difficult programming problems. The principal investigator expresses here his appreciation to all of the above named individuals, and also to those at the Air Force Weapons Laboratory whose support made this research possible.

This technical report has been reviewed and is approved.



JOHN M. IVES
Captain, USAF
Project Officer



CARL F. PORTER
Lt Colonel, USAF
Chief, Transient Radiation Effects
Branch



CARL F. DAVIS
Colonel, USAF
Chief, Electronics Division

ABSTRACT

(Distribution Limitation Statement A)

Radiation effects analysis of complex electronic circuits by computers presently is limited by core storage and long execution times. A new method, entitled Iterated Propagation of Bivariable Response Functions, treats components in the block sense using experimentally obtained time domain response or transfer functions. These functions are described in a three dimensional space of the input stimuli, output response, and time, by means of fitting the experimental data with cubic polynomials matched in the spline sense at interstices of the fitting grid. The basic data was obtained using conventional oscilloscopic techniques at the Kirtland AFB flash X-ray facility and the Sandia pulse reactor. A computer program called SAP, adapted to work with SCEPTRE, uses a modified convolution process to calculate the electrical and the gamma and neutron radiation responses from the surface descriptions. These responses are then superimposed to obtain the combined response. Such a process is not rigorous but rather an engineering class of approximation. The results achieved ranged from fair to excellent, as applied to a 741 operational amplifier and a 9704 dual NAND gate. The three dimensional surfaces provide a vivid grasp of device performance and radiation response over the entire range of the stimulus variables.

CONTENTS

<u>Section</u>		<u>Page</u>
I	INTRODUCTION	1
II	BASIC THEORY OF BIVARIABLE RESPONSE FUNCTION PROPAGATION	4
III	GATHERING THE EXPERIMENTAL DATA	40
IV	PROCESSING OF EXPERIMENTAL DATA	46
V	SAP COMPUTER PROGRAM	119
VI	RESULTS COMPUTED BY SAP USING TRANSFER FUNCTION SURFACES	167
VII	MERGING SAP WITH SCEPTRE	210
VIII	CONCLUSIONS	240
	Appendix - TECHNIQUES USED FOR REACTOR TESTS OF DEVICES	243

ILLUSTRATIONS

<u>Figure</u>		<u>Page</u>
1	Voltage Transfer Function of a Series 1000 ohm Resistor	9
2	Time Domain Voltage Transfer Function of the 709 Op Amp	19
3	Step Pulse Voltage Transfer Function Surface for the 741 Op Amp	20
4	Computer Fit to the 709 Op Amp Time Domain Voltage Transfer Function Surface	22
5	Computer Generated Three Dimensional Perspective Plot of the 741 Voltage Transfer Function Surface	23
6	Oscilloscope Photograph of Op Amp Voltage Waveforms	24
7	Storage Time Surface of the 709 Op Amp	26
8	Turn Off Response Surface of the 709 Op Amp	28
9	SAP Determination of 709 Pulse Response	35
10	SAP Determination of 709 Turn Off Response	37
11	SAP Determination of 709 Turn Off Response with Storage	38
12	Experimental Block Diagrams for Op Amp Measurements	42
13	Circuits Used for 9704 NAND Gate Measurements	43
14	Voltage Transfer Function of the 741 Op Amp due to FXR Irradiation	48
15	Radiation Transfer Function Surface for the 741 Op Amp due to SPR Neutron Irradiation	50
16	Flowchart for Computer Program FIT3D	55
17	Listing of FIT3D Data Read Statements	57
18	Data Deck for FIT3D	58
19	Orthographic Projection Used for 3D Plots	60
20	Computer Plot of 741 Voltage Transfer Function	62
21	Radiation Transfer Function of 741 due to FXR Exposure	63
22	Cross Section Through Maximum and Minimum TPHI	66
23	Voltage Response Function of 741 due to SPR Neutrons	69
24	Cross Section Through Maximum and Minimum TPHI	72
25	Electrical Characterization Circuitry for 9704 Gate	75
26	Voltage Response Function Surface of the 9704 NAND Gate	76
27	Current Neutron Response Function of the 9704 NAND Gate	80
28	Voltage Neutron Response Function of the 9704 NAND Gate	81
29	Current Neutron Function of the 9704, Run 322	83
30	Current Neutron Radiation Response of the 9704, Run 328	84

ILLUSTRATIONS
continued

<u>Figure</u>		<u>Page</u>
31	Current Neutron Radiation Response of 9704, Run 404	85
32	Voltage Neutron Radiation Response of 9704, Run 342	86
33	Voltage Neutron Radiation Response of 9704, Run 471	91
34	Current Neutron Radiation Response of 9704, Run 362	94
35	Current Neutron Radiation Response of 9704, Run 362	95
36	Current Neutron Radiation Response of 9704 with Increasing Number of Smoothing Passes, Front View	96
37	Current Neutron Radiation Response of 9704 with Increasing Number of Smoothing Passes, Rear View	102
38	FXR Voltage Radiation Transfer Function Surface of 9704	109
39	FXR Voltage Radiation Transfer Function Surface of 9704 with Increasing Number of Smoothing Passes	110
40	FXR Current Radiation Transfer Function of 9704	117
41	FXR Current Radiation Transfer Function of 9704 with Logarithmic Time Base	118
42	Flowchart of Computer Program SAP	120
43	MAIN Listing	122
44	RADGEN Listing	131
45	COMMON Listing	133
46	VOLTIM Listing	135
47	DRIVER Listing	138
48	Root Finding Strategies of DRIVER	145
49	FE Listing	146
50	FI Listing	149
51	FEPhi Listing	151
52	CONVOL Listing	154
53	TSURF Listing	160
54	DAMAGE Listing	160
55	PARLEL Listing	161
56	PLOTEM Listing	164
57	SAP Determination of 741 Response to Step Pulse	168
58	741 Sine Wave Response, Run 335	173
59	741 Response to a Saturating Pulse	181
60	Schematic Computational Diagram Indicating Method of Radiation Response Calculation	187

ILLUSTRATIONS
continued

<u>Figure</u>		<u>Page</u>
61	SAP Determination of 741 FXR Response	190
62	SAP Determination of 741 SPR Response with Step Voltage Pulse	192
63	741 Response to FXR with Dose = 6.8×10^7	195
64	741 Response to SPR without Convolution	196
65	9704 Response without Irradiation, Run 529	197
66	XY Recorder Tracing of 9704 Sampling Oscilloscope Waveforms	198
67	9704 SAP Response without Irradiation Run 541	200
68	9704 SAP Response to FXR Pulse, Rerun 447	202
69	9704 SAP Response to SPR Pulse, Run 555	203
70	9704 SAP Response to SPR Pulse, Run 570	204
71	9704 SAP Response to SPR Pulse with Input Low	206
72	9704 SAP Response to SPR Pulse with Input High	207
73	9704 SAP Response to SPR Pulse with Input Delayed	209
74	FOV741 and FOJ741 Listings	212
75	FVOLT2 Listing	214
76	MAIN2 Listing	216
77	RADGN2 Listing	220
78	SIMTR Listing	225
79	SIMUL8 Listing	226
80	SCEPTRE Plot of Input Voltage versus Time	228
81	SAP Printout during SCEPTRE Execution	230
82	SCEPTRE Printout during SCEPTRE Execution	235
83	Sandia Pulse Reactor Room	244
84	Styrofoam Block for Holding Samples for SPR Exposure	245
85	Pulse Reactor Timing Sequence	247
86	Effect of Neutrons on 741 Input-Output Characteristic	247
87	Effect of Neutrons on 741 Pulse Response	248

TABLES

<u>Table</u>		<u>Page</u>
I	Voltage and Current Transfer Functions for Simple Circuit Elements	10
II	KSURF Array	153

SECTION I

INTRODUCTION

This report describes the ideas behind a block oriented computer analysis technique termed Bivariable Response Function Propagation and how these ideas were applied to the electrical and radiation responses of a linear integrated circuit operational amplifier and a digital NAND gate. The dynamical response of the circuits is represented using three-dimensional surfaces which were fit to experimental data acquired from neutron, gamma ray, and electrical pulse stimuli. The block response in a subsequent circuit or system analysis problem is calculated using these surfaces by a process which may be termed convolution in the nonlinear sense; hence, the process constitutes an approximate rather than a rigorous method. Nevertheless, computed results show reasonable agreement with those obtained experimentally. Further, the response function surfaces provide a rich and different insight into the effects of radiation on devices.

1. PRESENT PROBLEMS

The presently existing transient radiation effect codes have been extremely effective in enabling device and circuit designers to assess the validity of their designs prior to actually fabricating or building them. By developing physical understanding of the electrical characteristics of the device, and then learning how radiation alters these physical properties, one can:

- extrapolate test data to other dosages,
- determine best device geometry and circuit layout,
- alter physical parameters to improve device characteristics,
- ascertain radiation testing required and what measurements are needed.

Thus the existing programs such as SCEPTRE, TRAC, and NET have indeed been useful in meeting the primary goals for which they were devised.

Several aspects of the present computer modeling approaches are unacceptable, namely:

- core limitation on nodes,
- excessive execution time, say due to very small time constants,

- convergence problems with certain eigenvalues. This depends on whether implicit or explicit integration is used in the program.
- Package effects are not included in the equivalent circuit. These and EMP effects can cause major differences between predicted and observed radiation responses.
- expenses at the Nevada test site are very high. This type of testing is presently needed since there are no suitable computer programs for predicting performance of high complexity circuits. Note also that such test procedures only yield one point in the parameter space of the radiation stimulus and its effect.

2. EXISTING TECHNIQUES FOR MODELING

The primary method employed to determine circuit behavior today is to set up Kirchoff's laws for the nodes and meshes using topological trees, build the equivalent device models into a set of simultaneous algebraic and differential equations, and solve these by standard matrix inversion techniques. Generally sparse matrix procedures can be used to minimize core storage requirements. Implicit integration can be used to reduce solution time, and enhance convergence likelihood.

For more complex systems, state variable formulations, physical differential equations, Linvill's pseudo-linearization models, and analog-hybrid simulations are the primary procedures available. It is important to note these methods may require heavy experience on the part of the user with the modeling process. The accuracy of an analog or hybrid simulation is not so much a question of the precision of the analog computer per se, but rather the validity of the model itself. Sometimes, supposedly linear elements may actually be non-linear in the transient sense. Consequently, a modeling approach based on the incorrect assumptions might appear satisfactory for slow phenomena but fail for fast events.

Time stepping is an inherent part of most transient computer codes. It is essential to realize that this is a fundamental restriction on the analysis technique employed. The human brain does not analyze a system by a time stepping process. Its memorized knowledge of the various response functions, and their sensitivities to various phenomena, enables it to sweep through a mass of detail and make reasonably valid conclusions about the true system performance. Needless to say this process is as much an art as a science. And moreover digital

computers up to now have found it difficult to emulate.

Laplace transformation or s variable analysis is probably the most useful tool engineers have available for treating linear systems or circuits. Unfortunately digital circuits and most radiation effects are nonlinear, and hence, this tool is not generally of help for them. The wide use of s variable analysis has acted to minimize the engineers' familiarity with time domain approaches, which is the way computer programs step through the solution. This is another reason for the dichotomy between thinking patterns and computer solution techniques.

Sensitivity analysis to a certain extent is in accord with the rough strategy the mind uses in evaluating various factors and how they affect a solution.

Radiation simulation by flash Xray, linear accelerators, lasers, reactors, Marx generators, cobalt sources, etc., has been widely used to provide radiation effects information on circuits and systems. The validity of the result depends on the pulse spectrum, time duration and shape, intensity, and secondary effects such as ground loops, shielding, input and output loading or fan in/out. For laser simulation, the presence of overlying metallizations can affect the result.

SECTION II

BASIC THEORY OF BIVARIABLE RESPONSE FUNCTION PROPAGATION

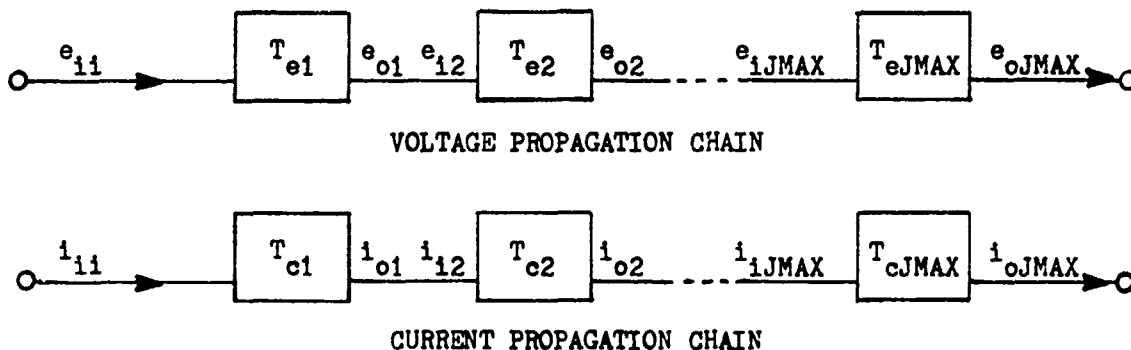
1. GENERAL PROCEDURE

As circuit and system complexity levels continue to expand, it seems to us mandatory to further develop techniques which treat circuit or system elements as blocks rather than a collection of parts. Note that this is a permissible simplification for the user compared with a designer of an integrated circuit or IC. The user of an IC can not alter the chip itself. So why should he necessarily be required to know the circuit diagram of it simply to judge its use as a block element in a more complex circuit or system? With this crucial aspect deleted from the analysis requirement, then transfer or response functions seems a permissible approach.

A very important consideration is that loading effects be taken into account. This immediately forbids simply treating voltage transfer functions alone. To show the elements of the general computation algorithm, let us now consider the case of simple serial blocks.

a. Serial Blocks

Suppose there are JMAX blocks connected in series with an input voltage and current applied at the input node of the first or $J = 1$ block, which is then propagated through it to the next so that its output voltage and current becomes the input voltage and current for the next $J = 2$ block, and so forth down the sequence to the last or JMAX block. Such a procedure is readily accomplished in a FORTRAN program by a DO loop on the index J.



The operators which yield the output given the input stimulus for a given block we will label T_e and T_c , the transfer or response functions on the voltage and current, respectively. In general, they are functions of time, and of the applied voltages and currents, as well as the circuit operating voltages, radiation degradation, etc. We will show later what some transfer functions look like, in equation and numerical form for typical circuit elements. But first the specific method for solving the system problem must be discussed.

Initially the various voltages and currents in the circuit are unknown. Thus an algorithm is needed to solve the currents for a specified input voltage and circuit configuration. To do so, note that the output voltages and currents are related to the input voltages and currents by

$$\begin{aligned} e_o &= T_{e1} T_{e2} T_{e3} \cdot \cdot \cdot T_{eJMAX} e_i \\ &= \left\{ \prod_{j=1}^{JMAX} T_{ej} \right\} e_i \\ &= P_e e_i \end{aligned} \tag{1}$$

and

$$\begin{aligned} i_o &= T_{c1} T_{c2} T_{c3} \cdot \cdot \cdot T_{cJMAX} i_i \\ &= P_c i_i \end{aligned} \tag{2}$$

where the repeated product of the block voltage and current transfer functions are abbreviated by P_e and P_c , the system response functions.

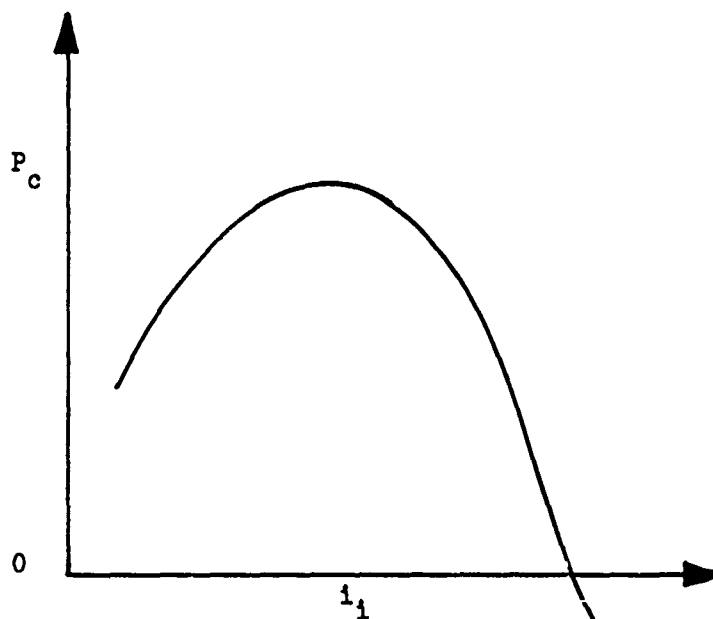
To solve the total system problem, suppose that a high-impedance load that effectively draws no current is at the last node. Hence

$$i_o = 0 \tag{3}$$

But from (2), $i_i \neq 0$, consequently the proper solution is that which drives the repeated product of the current transfer functions P to zero,

$$P_c = 0 \tag{4}$$

The system current response function is an explicit function of the input variables, and particularly the input current. Thus, if P_c is plotted versus the input current, we might have a functional dependence as shown on next page.

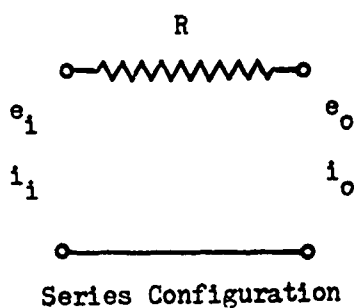


Sketch of Product of Current Transfer Functions versus Adjusted Input Variable

The computational algorithm to solve the system current and voltage problem is thus basically a zero finding process. In this case the function is regarded as a numerical form as compared with an equation or a polynomial as is the case with many system analysis techniques. The correct value of i_1 is that which yields P_c equal zero.

b. Examples of Block Transfer Functions

To establish the context of thinking of simple circuit elements as block elements, consider a resistor connected in series between the input and output terminals as shown below. The output current is equal to the input current



so that,

$$i_o = i_1$$

$$T_c = \frac{i_o}{i_1} = 1 \quad (5)$$

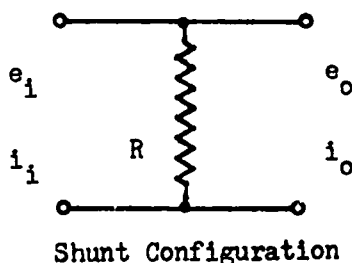
Similarly,

$$e_o = e_1 - i_1 R$$

hence

$$T_e = 1 - \frac{i_1 R}{e_1} \quad (6)$$

Whereas, for a shunt connected resistor, the input and output voltages are equal, but the output current is reduced by that shunted through the resistor to ground,



so,

$$i_o = i_i - \frac{e_i}{R}$$

$$T_e = 1 \quad (7)$$

$$T_c = 1 - \frac{e_i}{i_i R} \quad (8)$$

Thus, the same circuit element can have different transfer functions depending on its circuit configuration. For an integrated circuit being used in a larger circuit or system the chip configuration is fixed; hence its transfer function is fixed.

The above equations for the series and shunt transfer function of a resistor are very simple analytical relations. However if we want to be able to scale complexity levels by computer analysis without completely necessitating the interaction of a skilled engineer, it pays at this point to consider an alternative mathematical representation.

Let eq. (5) for the case of a 1K ohm resistor be plotted with respect to the space of the input variables and the transfer function. It then represents a three-dimensional surface as shown in Figure 1. Suppose we are dealing with a block element for which we do not have a suitable analytical expression. By developing a numerically defined surface, and fitting it with suitable surface approximants, such as three-dimensional splines, we can still perform a system analysis. Such a surface can be deduced by experimental measurements or by operating a component level computer analysis program such as SCEPTRE and determining the response function over the space of the input variables by direct computation.

While it is certainly much more conservative of core to use an equation if that is available, we are not stopped provided we have a numerical representation of the transfer function.

To see how such a surface would be used in the computer analysis program (and actually has been in the code SAP which will be described later), a propagating DO loop brings values of input voltages and currents to a particular block, calls in the subroutines for the voltage and current transfer functions that do look-ups on the surface, and return back the value to the calling program that performs the propagation.

The current transfer function for the shunt connected resistor is essentially the dual of the one shown in Figure 1, with the axes being interchanged and the resistance becoming the conductance.

A list of analytical expressions for the voltage and current transfer functions for some common circuit elements is presented in Table I.

c. Iteration Algorithm

A solution of the system problem can be obtained by fixing one of the input variables, e_i , and one of the output variables, i_o . For purpose of nomenclature, let these be referred to as the fixed input or output variables. The solution strategy of the computer algorithm varies the nonfixed or adjusted input variable in a manner which forces the system current response function P_c to zero. We make an arbitrary guess at the initial value of the adjusted variable, and call it x in general, or for a specific iterate, a . This iterate is then "propagated" through the sequence of blocks.

The result of the propagation yields a value of the fixed output variable y which is probably not the desired value y_o corresponding to our constraint condition imposed (i.e. $i_o = 0$.) We need then to correct our guess at x which will bring y closer to y_o . These words can be expressed in functional form as:

$$y = Px + y_o \quad (9)$$

adjusted output variable

desired value of adjusted output variable

proper response function

adjusted input variable

Newton devised a tactic three centuries ago for solving this type of problem. Given an iterate, a , an improved guess or second iterate a' for x which drives

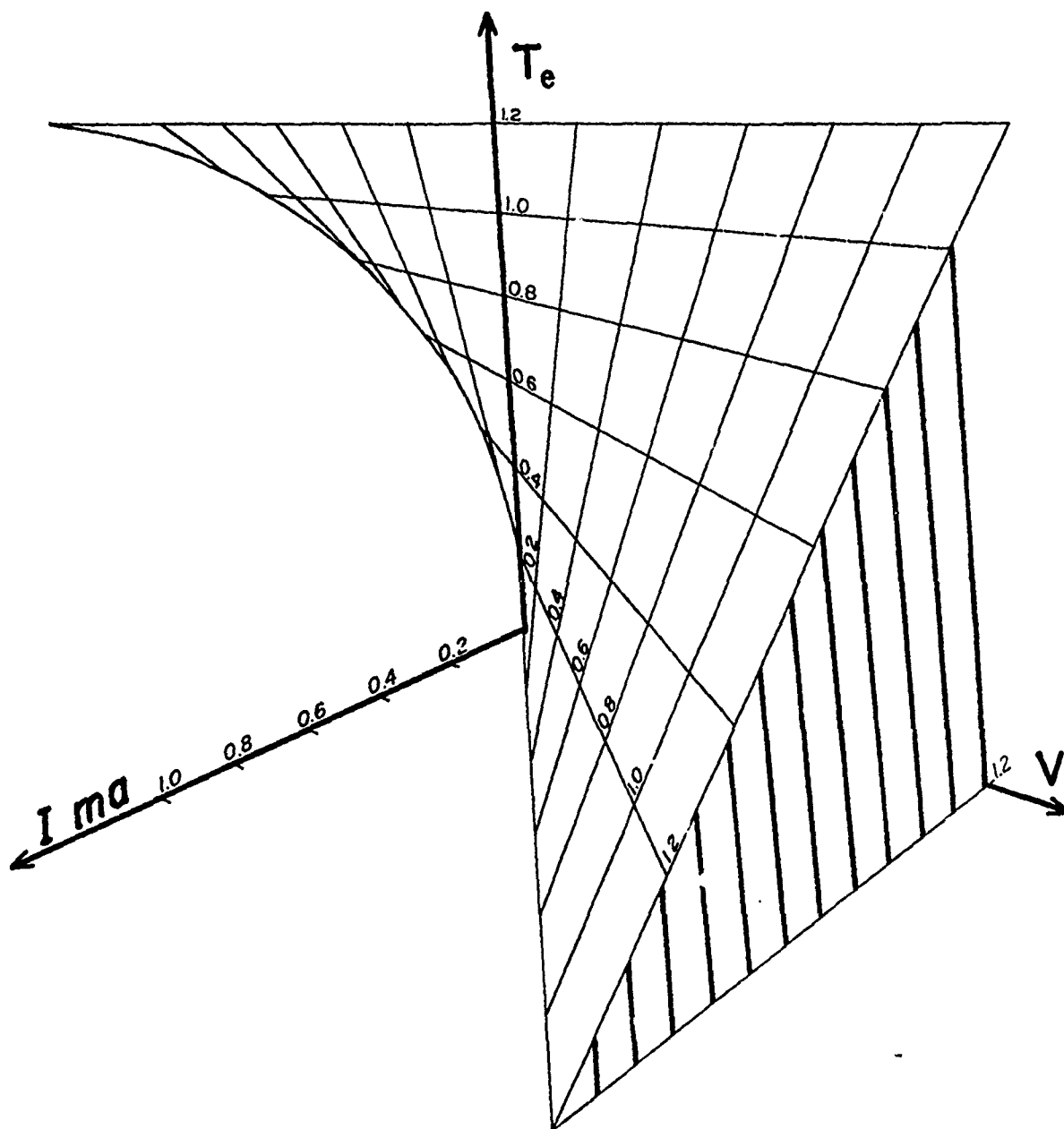
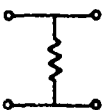
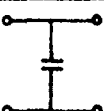
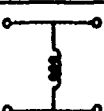
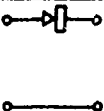
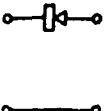
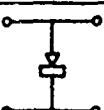
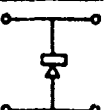


Figure 1. Voltage Transfer Function of a Series 1000 ohm Resistor

Table I

COMPONENT	CONFIGURATION	BLOCK DIAGRAMS	T_e	T_c
Resistor	Series		$1 - \frac{i_1 R}{e_1}$	1
Resistor	Shunt		1	$1 - \frac{e_1}{i_1 R}$
Capacitor	Series		$1 - \frac{1}{e_1 C} \int i_1 dt$	1
Capacitor	Shunt		1	$1 - \frac{C}{i_1} \frac{de_1}{dt}$
Inductance	Series		$1 - \frac{L}{e_1} \frac{di_1}{dt}$	1
Inductance	Shunt		1	$1 - \frac{1}{Li_1} \int e_1 dt$
Diode	Series Forward Biased		$1 - \frac{kT}{qe_1} \ln \left(\frac{i_1}{i_c} + 1 \right)$	1
Diode	Series Reverse Biased		$1 + \frac{kT}{qe_1} \ln \left(\frac{i_1}{i_c} + 1 \right)$	1
Diode	Shunt Forward Biased		1	$1 - \frac{i_c}{i_1} \left(e^{e_1 q/kT} - 1 \right)$
Diode	Shunt Reverse Biased		1	$1 - \frac{i_c}{i_1} \left(e^{-e_1 q/kT} - 1 \right)$

y to zero, or $y - y_0$ to zero, is given by

$$a' = a - \frac{f(a)}{f'(a)} \quad (10)$$

wherein $f(a) = y$. Now since $f(a) = Px + y_0$,

$$f'(a) = \left. \frac{dy}{dx} \right|_{x=a} = P + a \left. \frac{dP}{dx} \right|_a$$

Thus

$$\begin{aligned} a' &= a - \frac{(aP + y_0)}{P + a \left. \frac{dP}{dx} \right|_a} \\ &= \frac{a^2 \left. \frac{dP}{dx} \right|_a - y_0}{P + a \left. \frac{dP}{dx} \right|_a} \end{aligned} \quad (11)$$

which is the basic iteration equation utilized in the computer program SAP. It should be noted that the derivative of the response function with respect to the adjusted input variable will be of concern to the numerical solution of the problem. The numerical results which will be shown later have been calculated using eq. (11). To obtain the derivative initially, the second iterate is set at some fixed ratio of the first iterate. Typically this ratio would be $a'/a = 0.999$.

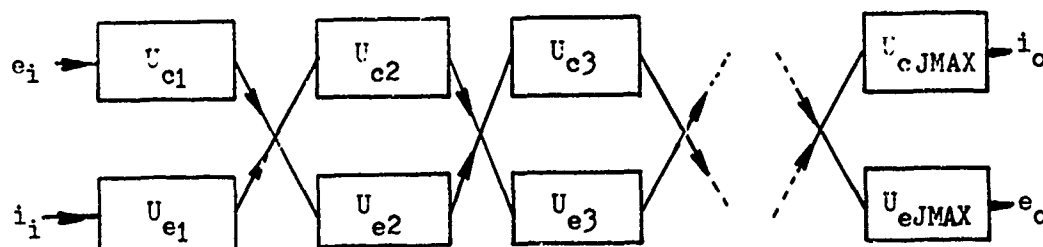
If multiple zeroes are contained in the response function, by dividing P by the first one found, the next one can be found without refinding the first one.

In a time stepping operation, the last converged value of the adjusted input variable can be extrapolated using the relative variation of the new fixed input variable compared to the last value. By this means the iteration convergence is enhanced. Using SAP for the 741 op amp, typically three to five passes are required to converge the output to 10 microvolts out of a 10-volt level. If the input impedance is linear the extrapolation may be sufficiently good to converge on the first pass.

d. Crossed Variables

In some situations the output variable which is fixed must be the same variable which is fixed in the input. In the case of the series block sequence discussed earlier, that was not the case. There the voltage was fixed at the input while the current was fixed at the output. Consequently, we could iterate on the adjusted input variable so as to drive the fixed output variable to its

proper value. It is easy to visualize this process as separate propagation of the voltage along the voltage transfer function chain, and the currents along the current transfer function chain. To help the understanding for the new 'crossed' situation, the propagation process can be looked on as cross-interacting the variables after each block as shown in the sketch below.



Mathematically eq.(9) now becomes

$$e_o = P_c^x i_i + e_{o,fix} \quad (12)$$

where the "voltage cross response function" is defined by

$$P_e^x = \frac{e_{oJMAX}}{i_{i1}} = U_{e1} U_{c2} U_{e3} \cdots U_{eJMAX} \quad (\text{for } JMAX \text{ odd}) \quad (13)$$

$$= U_{e1} U_{c2} U_{e3} \cdots T_{eJMAX} \quad (\text{for } JMAX \text{ even}) \quad (14)$$

in which the U s are "cross transfer function" on the voltage or current, and are defined by

$$U_{ej} = e_{oj}/i_{ij} \quad (15)$$

$$U_{cj} = i_{oj}/e_{ij} \quad (16)$$

in which the subscript j denotes the j^{th} block. A "current cross response function" can similarly be defined as

$$P_c^x = i_{oJMAX}/e_{i1} \quad (17)$$

The equation for P_c^x is essentially the same as (13) or (14) with the e 's and c 's interchanged.

Previously the voltages and currents were seemingly uncoupled as the computation propagates along the blocks. Now each variable cross interacts after each block as the cross-transfer functions are called into play. While seemingly complicated, this is actually easy to incorporate in a FORTRAN program using a

sequence of computed GO TOs, or many-way switches. The remainder of the formalism is essentially unchanged and the basic iteration and convergence aspects are the same.

The cross transfer functions do not essentially introduce any new information since they are deducible from the regular transfer functions by a simple transformation, namely,

$$U_{ej} = T_{ej} \frac{e_i}{i_i} \quad (18)$$

It should be pointed out that in the course of the iteration that the ratio e_i/i_i is not the input impedance of the block in the general sense. Rather we are using the voltages and currents as independent variables. Only after the solution has converged correctly as demanded by the constraint of eq.(4) does the e_i/i_i ratio represent the physically correct input impedance as seen at that block element.

Thus, the iteration process can be looked upon as finding numerically that coupling of the voltage and current variables which correctly satisfies the physical problem.

e. Parallel Blocks

So far the circuit or system has been treated as a sequence of blocks connected in series. The computational algorithm is a DO loop that simply propagates the current and voltage variables from $J = 1$ to $J = JMAX$. This DO is nested in the outer iteration DO which acts upon the adjusted input variable. Suppose now that a number N of blocks are connected in parallel, presumably located in the previous series sequence. Let us now see if we can devise an effective algorithm for solving this problem by a transfer function method.

Presume that the input current in branch j is some fraction F_j of the total available input current, and that on the first iteration pass the current is assumed to split equally, so that $F_j(1) = 1/N$. The current into the j^{th} block is then

$$i_{ij} = F_j(1) i_i \quad (19)$$

Knowing these input currents and the value of the input voltage, the output voltage calculated using their voltage cross transfer functions U_{ej} as given by (15), can be used to calculate the average output voltage of all the parallel blocks as

$$\bar{e}_o = \frac{1}{N} \sum_{j=1}^N U_{ej} i_{ij} \quad (20)$$

and the deviation of the individual blocks from that computed average,

$$\Delta e_j = e_{oj} - \bar{e}_o \quad (21)$$

Next a relation indicating how a block output voltage can be related to a change in block input current is obtainable by differentiating eq.(15), yielding

$$\Delta e_j = U_{ej} + i_{1j} \frac{\partial U_{ej}}{\partial i_{1j}} \Delta i_{1j} \quad (22)$$

By equating the left hand side of (22) to the deviation (21) incurred on a given iteration pass, the change in the current which should be propagated through a block in order to drive it closer to the average is then obtained from (22). However, if all the new input currents are calculated in this manner, it would seem unlikely that their sum would still equal the available input current i_1 coming from the prior block in the series sequence. Suppose that the sum of the new input currents obtained using the increments calculated from (21) and (22) is

$$i_1' = \sum_{j=1}^N i_{1j} \quad (23)$$

and that the ratio of this sum to the available input current i_1 is some fraction f ,

$$f = \frac{i_1'}{i_1} \quad (24)$$

If all the tentative block input currents are divided by f , then Kirchoff's current law at the input node is again satisfied. These new input currents are then propagated through their blocks and a new floating average output voltage is computed as before, and the whole cycle is repeated until $\Delta e_j = 0$.

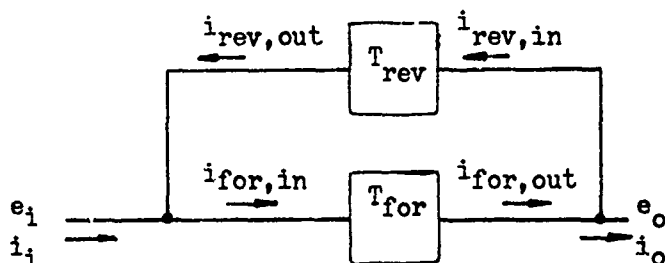
It is important to note that the above algorithm retains the advantages of the sequential analysis, namely:

- complete symmetry in the computational sense with respect to all block elements,
- no necessary restrictions to linearity, and
- no matrices involved in the solution.

The parallel block algorithm is a generalization on Newton's method where the fixed output constraint is changed to convergence to an average output voltage which floats during the iteration process.

f. Feedback Loops

Up to now the blocks have been considered as propagating input variables in one directed sense from the input node to the output node of the circuit or system. In this subsection a feedback loop will be considered with the transfer functions presumed to have a directionality. Surprisingly, the voltage transfer functions are not equal in this case, as was true for the simple parallel case.



Rather since

$$e_o = T_{e,for} e_i$$

and

$$e_i = T_{e,rev} e_o$$

the result is

$$T_{e,for} = \frac{1}{T_{e,rev}} \quad (25)$$

The computational algorithm for this case is somewhat like the parallel case described previously. To initiate the calculation apply e_o and some fraction of i_o to the reverse block and propagate to find $e_{rev,out}$ and $i_{rev,out}$. Next define

$$\Delta e = e_{rev,out} - e_i \quad (26)$$

Use Kirchoff's law and set

$$i_{for,in} = i_i + i_{rev,out}$$

Calculate a new e_o using the voltage cross transfer function, and $i_{for,out}$ using the current direct transfer function for the forward block. The voltage out of the reverse block is

$$e_{rev,out} = U_{e,rev} i_{in}$$

This equation for the voltage out of the feedback block can be differentiated to yield

$$\Delta e_{rev,out} = \left\{ U_{e,rev} + i_{rev,in} \frac{\partial U_{e,rev}}{\partial i_{rev,in}} \right\} \Delta i_{rev,in} \quad (27)$$

Equation (27) can then be solved for the new value of $\Delta i_{rev,in}$ using the prior value of Δe which will drive the voltage difference Δe toward zero. The condition $\Delta e = 0$ corresponds to the desired result, since Kirchoff's voltage law is then obeyed. The above process is recycled until that condition is obtained.

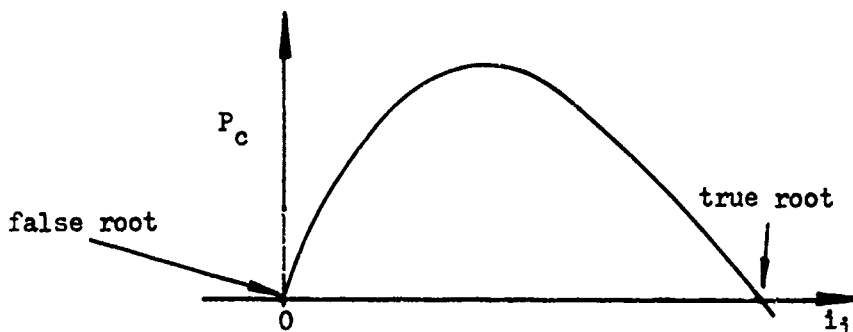
g. False Zeroes

The basic condition used in this iterative method for solving the total system aspect of the problem is that the repeated product of the block transfer functions, P_c , drives to zero when the output constraint is properly satisfied. Normally this will happen when

$$T_{c,JMAX} = 0 \quad (28)$$

For linear circuits no problem will arise if, during the iteration process, an iterate drives the block into an extreme condition. If the block is non-linear the same statement can not in general be made. As a result an intermediate T_{cj} may drive to a false zero. There may be physical cases when this might occur validly. However, before the iteration process is completed the variables are uncoupled and the zero is improper. In such a case, logic must be used in the block subroutine to check whether such is the case, and if so alter the value propagated on to a satisfactory non-zero value. Generally the logic condition we use in this test is that only the last block has a normal zero value of the output current when the input to the block is non-zero.

A more typical false zero arises from the fact that there is an automatic zero of the P function if the input current to the first block, $CIN(1)$, is driven to zero by the iteration process. In other words, the function P has at least one false and one true zero for a given value of the fixed input variable. Such a condition is shown in the following sketch.



Thus, the iteration driver routine needs to have more sophisticated zero finding strategies than a simple Newton-Raphson method which steers the iterate only on the basis of the derivative. Bounds must be set to restrict the iteration from driving to the false zero. The P function must also be free of other anomalies such as wild derivative variations in the neighborhood of the root; and if

the bounding technique is to work, free also of sliding of the root with the iterate as the iteration process homes in on the final value.

2. NONLINEAR RESPONSE FUNCTIONS

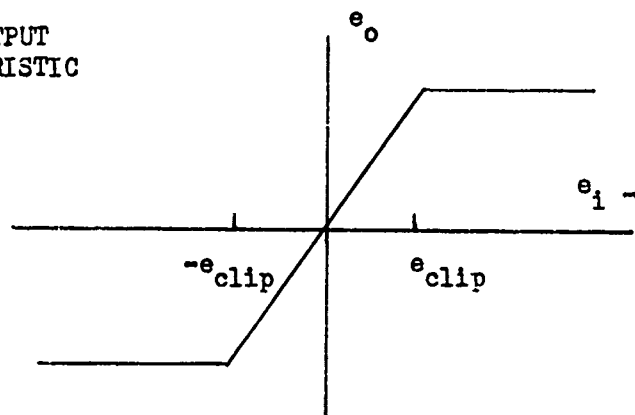
In this section we will get into the heart of modeling problems relating to nonlinear, complex, circuits. The prior section has shown that using ideas such as bivariable propagation, specific algorithms can be devised which correctly solve the case of circuits consisting of interdependent blocks. This is essential if loading effects such as fan-in and fan-out are to be properly taken into account. Thus, we can affirmatively say that it is at least permissible to consider using transfer or response functions for the block elements and still be able to perform valid system analysis. The prime questions which yet remain to be answered relate to the ability to properly take into account general classes of nonlinearities. The results obtained so far are encouraging as the results presented will show.

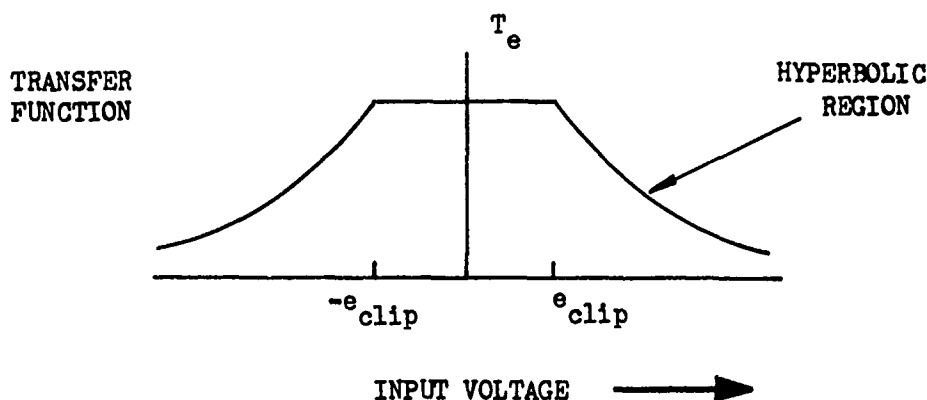
a. The 709 and 741 Operational Amplifiers

To provide an understanding of the applicability of the prior formalism to nonlinear blocks, two operational amplifiers will be treated and compared with one another to see the universality and the differences which accrue to separate devices. The 709 is an externally compensated monolithic integrated circuit op amp with a low-frequency gain of about 25,000. It is reasonably linear over a range of plus and minus 400 μv , and then saturates hard. The 741 is similar to the 709, with a somewhat higher low-frequency gain of 28,000; the prime difference between the two is that the 741 contains its own compensation circuit, and has internal balance points for offset adjustment. Neither the 709 nor the 741 are radiation hardened.

The general type of nonlinearity which these op amps have is shown in the following sketches.

INPUT-OUTPUT
CHARACTERISTIC





Let the input voltage at which the device starts to clip hard be referred to as e_{clip} . The corresponding transfer function T_e then has the form shown above. The voltage gain is constant over the linear range. Outside that range, the output is constant while the input continues to increase, thus the voltage transfer function has the form of a constant divided by an independent variable, the equation of a hyperbola. Hence, beyond e_{clip} the transfer function is a segment of a hyperbolic sheet. This is the mode of presentation of saturation class of nonlinearities as it is usually presented in the textbooks. There is much more to it than that.

To determine the time domain transfer functions, a step pulse was applied to the input, and the resultant response of the device photographed on an oscilloscope. By varying the magnitude of the step height, the time domain "turn on" transfer functions shown in Figure 2 for the 709 and Figure 3 for the 741 were determined. The height of the transfer function when multiplied by the magnitude of the pulse step indicates directly the time development of the output waveform. The shape of the 709 surface has some interesting surprises.

A time delay or propagation delay is evident for very small pulse amplitudes that gives rise to a valley in the supposedly linear range of the 709 device characteristic. This valley persists for up to 18 ms after device turn on. Note that the T_e surface for the 741 does not show this effect. Three important conclusions are:

1. the above result shows that nominally linear devices can be nonlinear in the transient sense,
2. the boundary region between linear and nonlinear modes of operation is not constant at a value of input voltage

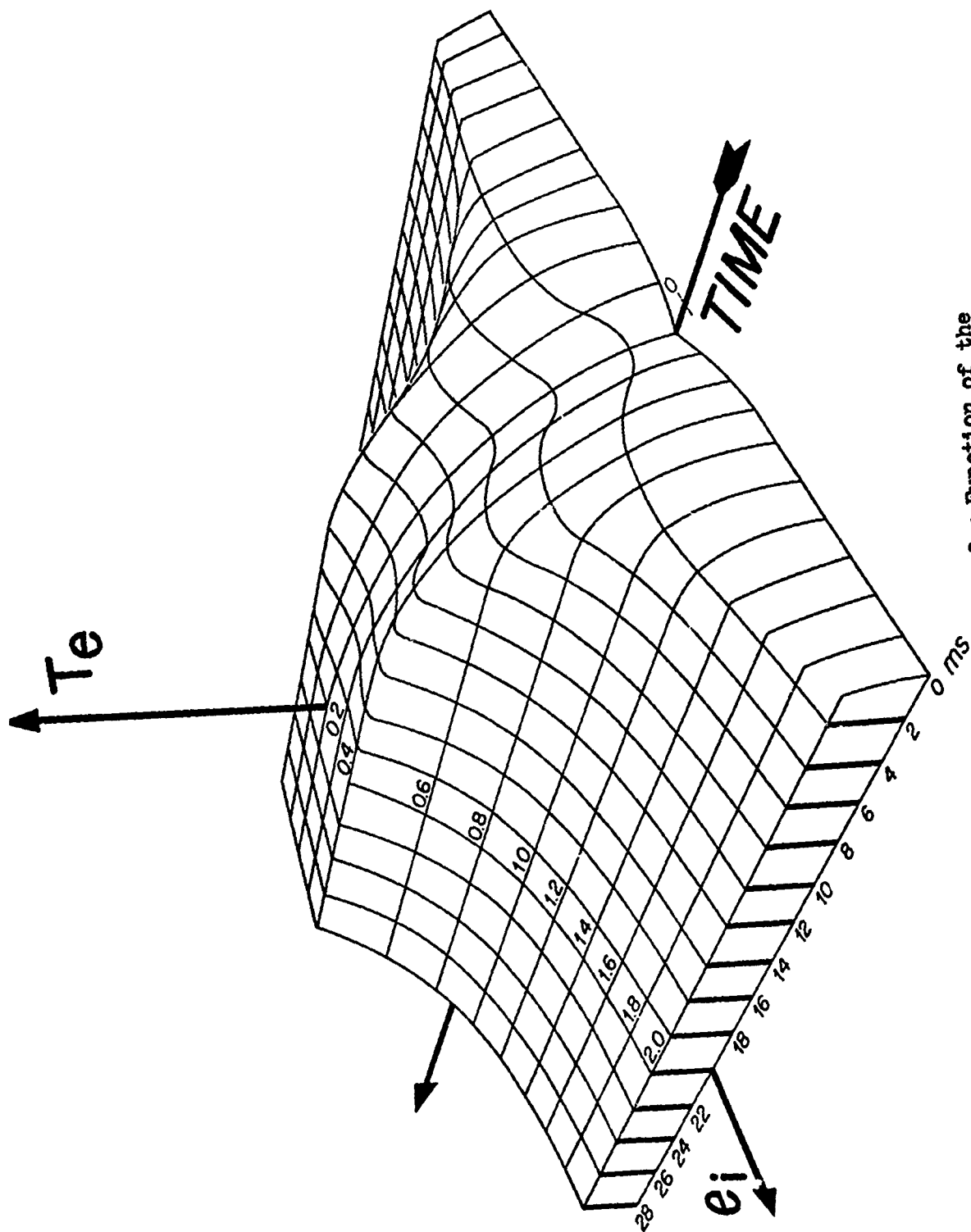


Figure 2. Time Domain Voltage Transfer Function of the 709 Op Amp, Response to Voltage Step Turn-on

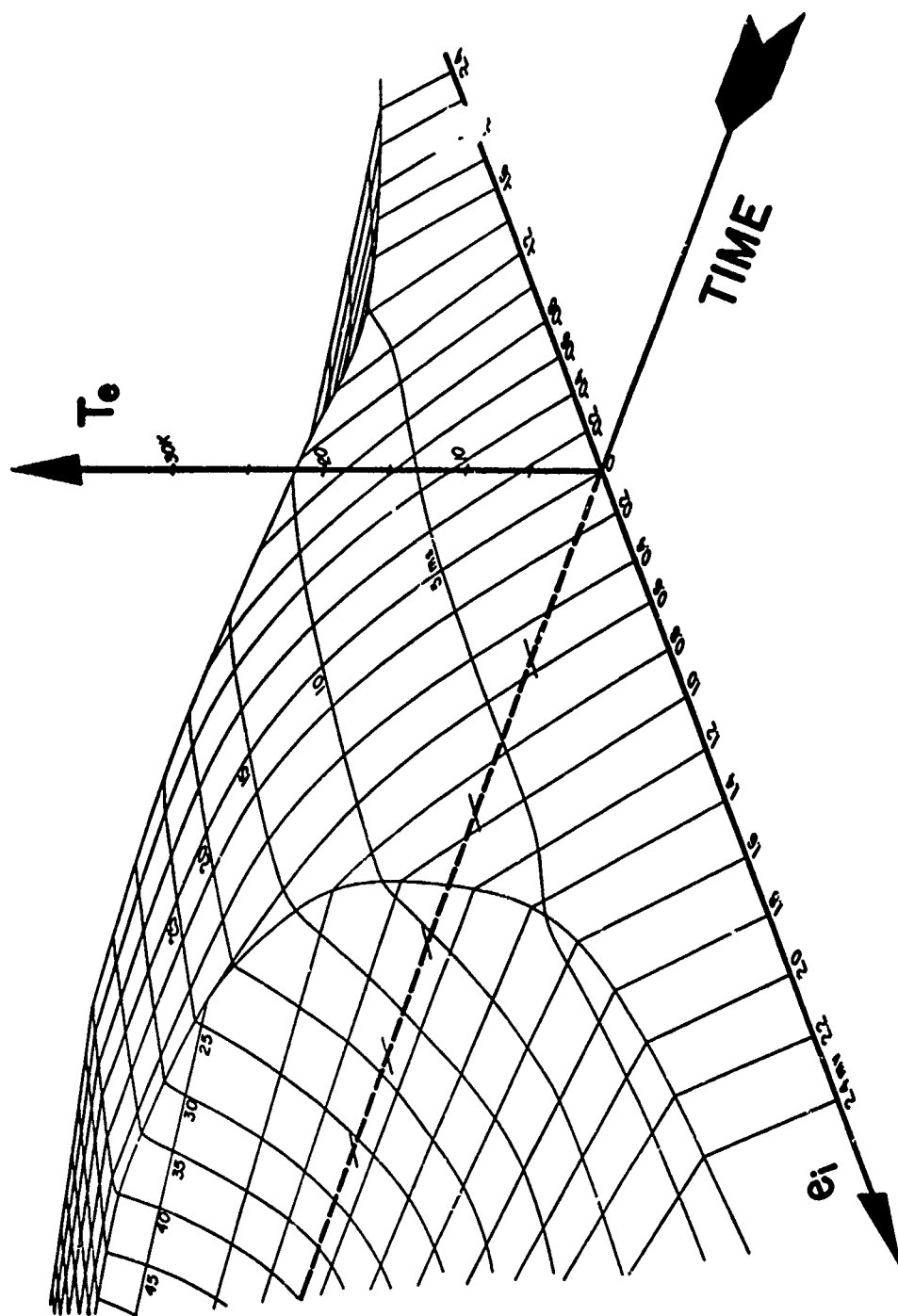


Figure 3. Time Domain Voltage Transfer Function of the 741 Op Amp, Response to Voltage Step Turn-on

- e_{clip} but rather swings out to higher input voltages at early values of time. Roughly the shape of this transition region is hyperbolic in the (e_1, t) plane,
3. the step transfer function rises up to a maximum value and remains flat or constant at long periods of time after step turn on.

It is clear by inspection of the time domain transfer function that a far more meaningful understanding of the device characteristics is obtainable from it than from the simple DC input-output characteristic. The physical sense of inertia is obvious. Hit the device with a stimulus and it takes up to 50 ms for the 741 to respond. This is the obvious cause of phase shift in the device. Also we see that the harder we hit the device (which corresponds to moving out further along the e_1 axis), the faster the device responds. It does not respond any harder once the nonlinear range is reached, but it does respond faster and faster until the response is slew-rate limited and thereafter the rise is constant.

While this surface was obtained by experimental observation of the device response on an oscilloscope and plotting the results using perspective drawing techniques, such a surface could have been equally well obtained by using a component level analysis computer program such as SCEPTRE. Three-dimensional computer plotting routines can generate a perspective plot rather easily. The most important question that must be asked is whether or not such a surface is of use in performing transient analysis at higher complexity levels. If so, then we have a strictly numerical way of staging complexity levels in system analysis. Our results for the combination of electrical and radiation stimuli on the 741 are encouraging.

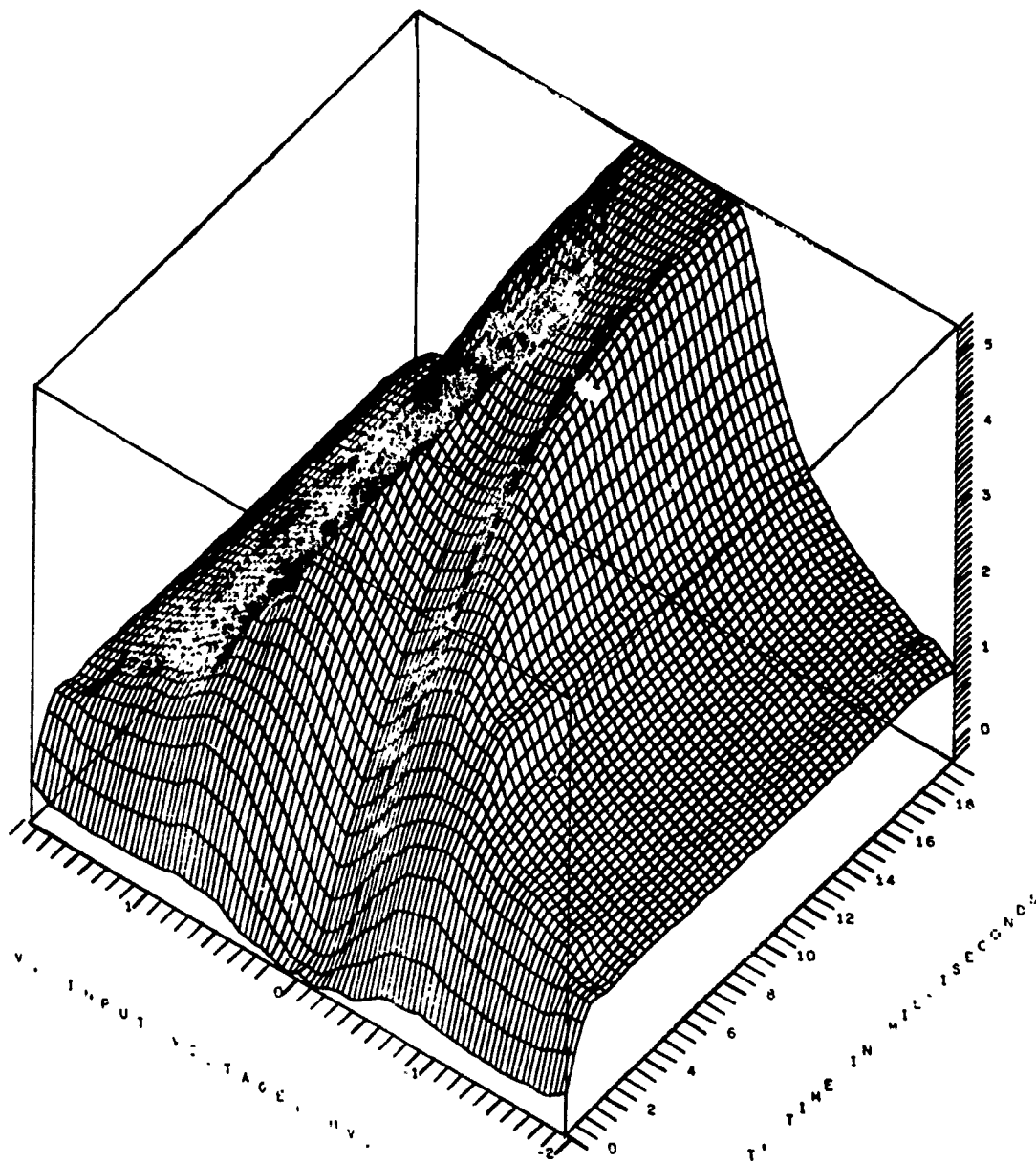
A computer generated approximation to the experimental surfaces are shown in Figures 4 and 5. This approximation uses three-dimensional splines fit in the least square sense.

The time domain response function of the 709 demonstrates quite nicely three general types of nonlinearity:

- saturation or clipping,
- distortion, and
- propagation delay.

There are other classes of nonlinearity which Figure 2 does not show, namely,

- storage,



3-FF RESPONSE FUNCTION SURFACE OF INTEGRATED CIRCUIT 709 OP AMP

Figure 4. Time Domain Voltage Transfer
Function of the 709 IC Op Amp
Response to Step Pulse Turn-On

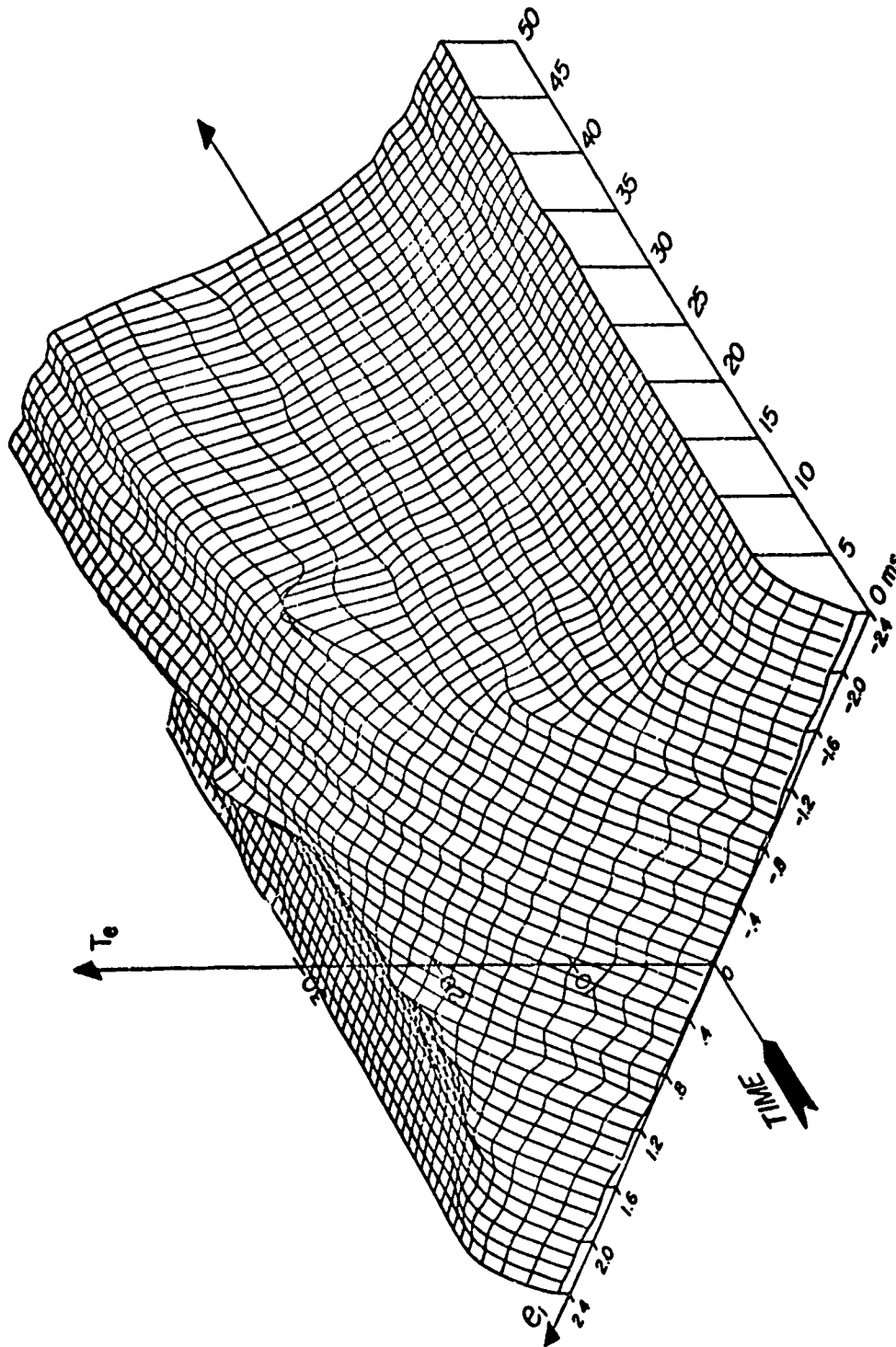


Figure 5. Computer Generated Three Dimensional Computer Plot of the Step Pulse Voltage Transfer Function of the 741 Op Amp. This plot is the result of the first pass at a computer fit and should be compared with that shown in Figure 20.

- deadzone,
- hysteresis, and
- instabilities, such as latch-up.

Of these four, deadzone is very easy to model since that simply is represented by a region in the three dimensional surface where the vertical height T is zero. Storage will be discussed in detail as it applies to the 709. Hysteresis is not a major effect with the 709 but could be for some digital circuits, such as a Schmitt trigger. Latch-up is a serious instability induced in semiconductor integrated circuits by radiation. It is generally a form of positive feedback in the internal circuit permitted to happen as the isolation junction to the substrate becomes conductive due to the photocurrent.

b. Storage Time Surface of the 709

The transfer function surface given in Figures 2 and 3 were obtained by photographing the response to a pulse as presented on an oscilloscope. A typical waveform is shown in Figure 6. The input pulse is in the linear range.

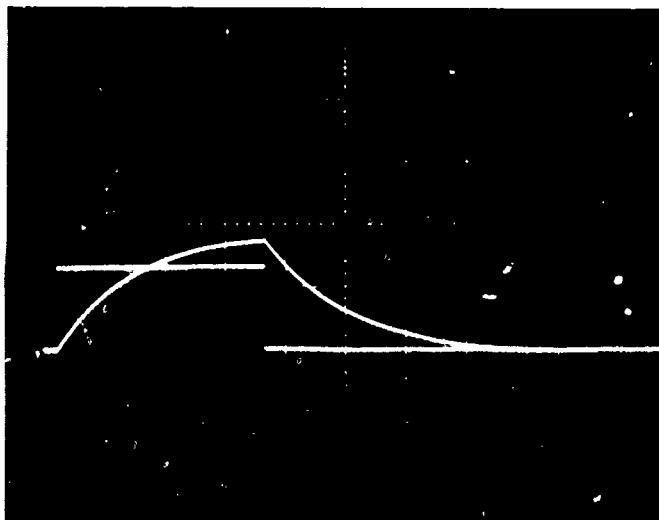


Figure 6. Polaroid Oscilloscope Photograph of Input and Output Voltage Waveforms of 709 Op Amp

The important point to note is that the output immediately starts to drop as the pulse turns off. In other words there is no storage in the linear range.

The surface as given in Figure 2 represents the rising part of the output waveform. In the linear region, it is seen that the falling part is the same as the rising.

When the input signal drives into the nonlinear range, the output signal quickly rises to the maximum allowable swing, 13 v, and clips hard. When the input is turned off, the output holds up at the same maximum value for some time, which we will term the storage time, τ_s , after that turnoff. Then the output drops with the same shape that it fell in the linear range. This aspect of the device characteristic is not contained in the turn transfer function surface of Figure 2.

The length of time that the output remains up after turn off of the input drive increases as

- the magnitude of the drive pulse increases,
- the length of the drive pulse increases,

up to a maximum value of storage time.

By experimentally observing the correlation of the storage time with the pulse amplitude and length, the three-dimensional surface presented in Figure 7 was determined.

The horizontal axis of the three-dimensional drawing is the magnitude of the electrical input signal applied to the inverting input of the 709 operational amplifier, the axis running backward is the pulse length in milliseconds applied to the input, while the vertical axis is the storage time in milliseconds.

The drawing itself was initially prepared using three point perspective grids approximately 3 feet square. It is necessary to use such large grids in the initial drawing in order to retain any semblance of the precision inherent in the data. The use of the grids is nicely described in the book by McCartney*.

*T. O. McCartney, Precision Perspective Drawing, McGraw-Hill Book Company, New York, 1963. The grids are obtainable from Perspective, Inc., 4400 Seventh Avenue, Seattle, Washington.

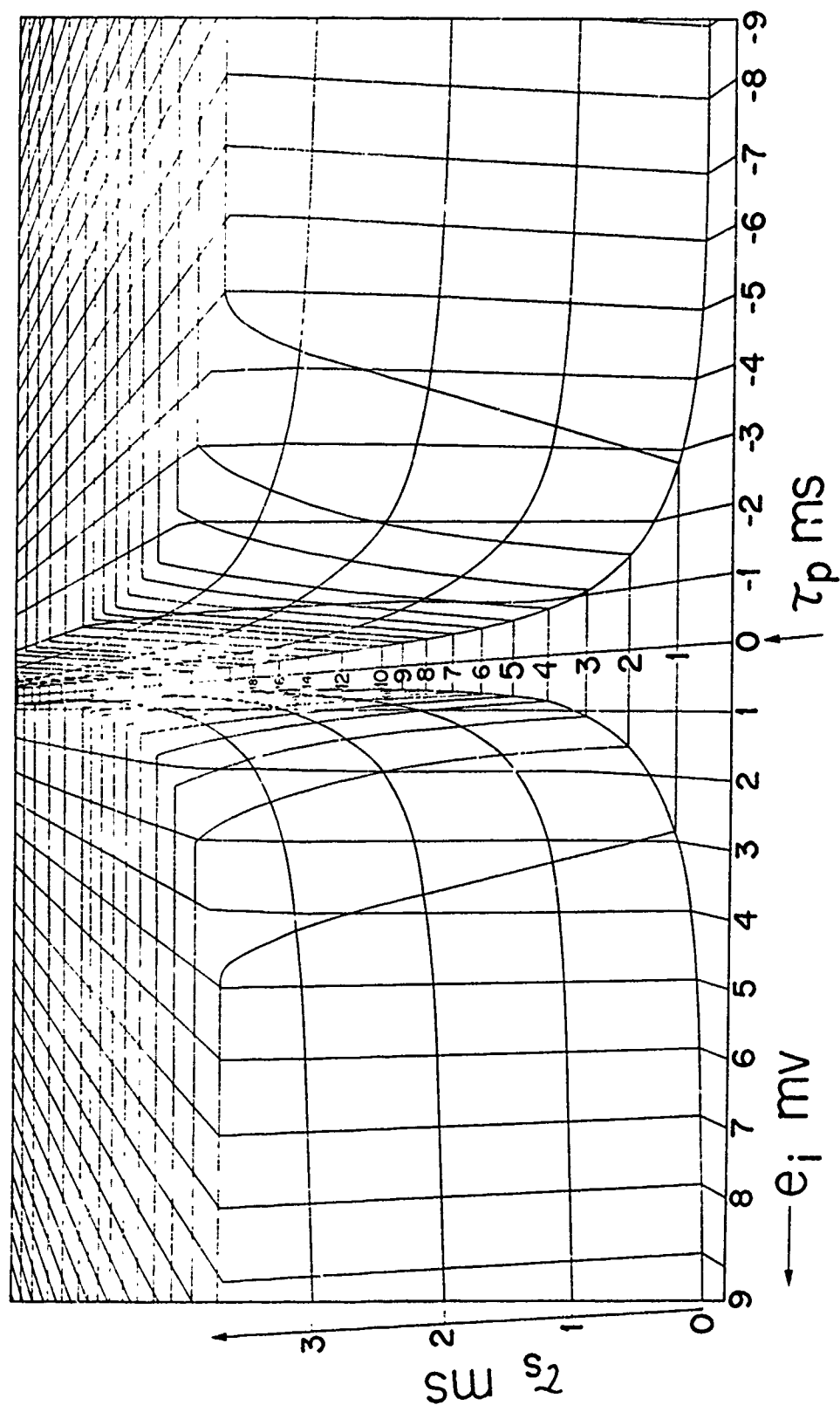


Figure 7. Storage Time Surface of the 709 Integrated Circuit Op Amp

Figure 7 presents at a glance the dependence on the input voltage e_i and pulse length τ_p of the storage time τ_s . The especially interesting things to note at this point are:

- for short pulses, large signals can be applied without storage resulting,
- for long pulses, only a small excursion beyond the linear range results in maximum storage,
- the roughly hyperbolic shape of the iso-storage curves or cuts through the surface parallel to the (e_i, τ_p) plane.

The first point is of theoretical importance since it relates to the equivalence of step pulse and impulse analysis of a circuit.

The storage time obtained from the storage time surface is actually used in conjunction with what we will term the "turn off response surface," a surface representing the decay characteristic of the output waveform after pulse turnoff. This surface is represented in Figure 8. The surface has three essential zones:

- exponential cylindrical - the linear decay region,
- hyperbolic cylindrical - the region where the pulse continues out to the storage time τ_s at full amplitude before starting to drop, and
- hyperbolic exponential - the drop from saturation for input stimuli exceeding the linear range.

The information obtained from Figure 7 is utilized in the computer program SAP to locate the line of demarcation between the hyperbolic cylindrical and the hyperbolic exponential zones. In other words, that line of demarcation is a moving line rather than a fixed unchanging line for a given input pulse. The particular location given in Figure 8 corresponds approximately to the case of maximum storage time at the signal level $e_i = 2$ mv. At a signal level $e_i = 0.8$ mv it is seen that the storage time $\tau_s = 2$ ms. Then by inspection, a storage time of $\tau_s = 2$ ms corresponds to a pulse length $\tau_p = 4$ ms. Thus the turn off response surface has been drawn for the case of an input pulse 4 ms long. What the computer look-up on the the storage time surface does is provide a displacement of the line of demarcation to the proper position corresponding to the particular applied waveform. The degree of success obtained by this technique is shown later in the computer output waveforms of Figure 11.

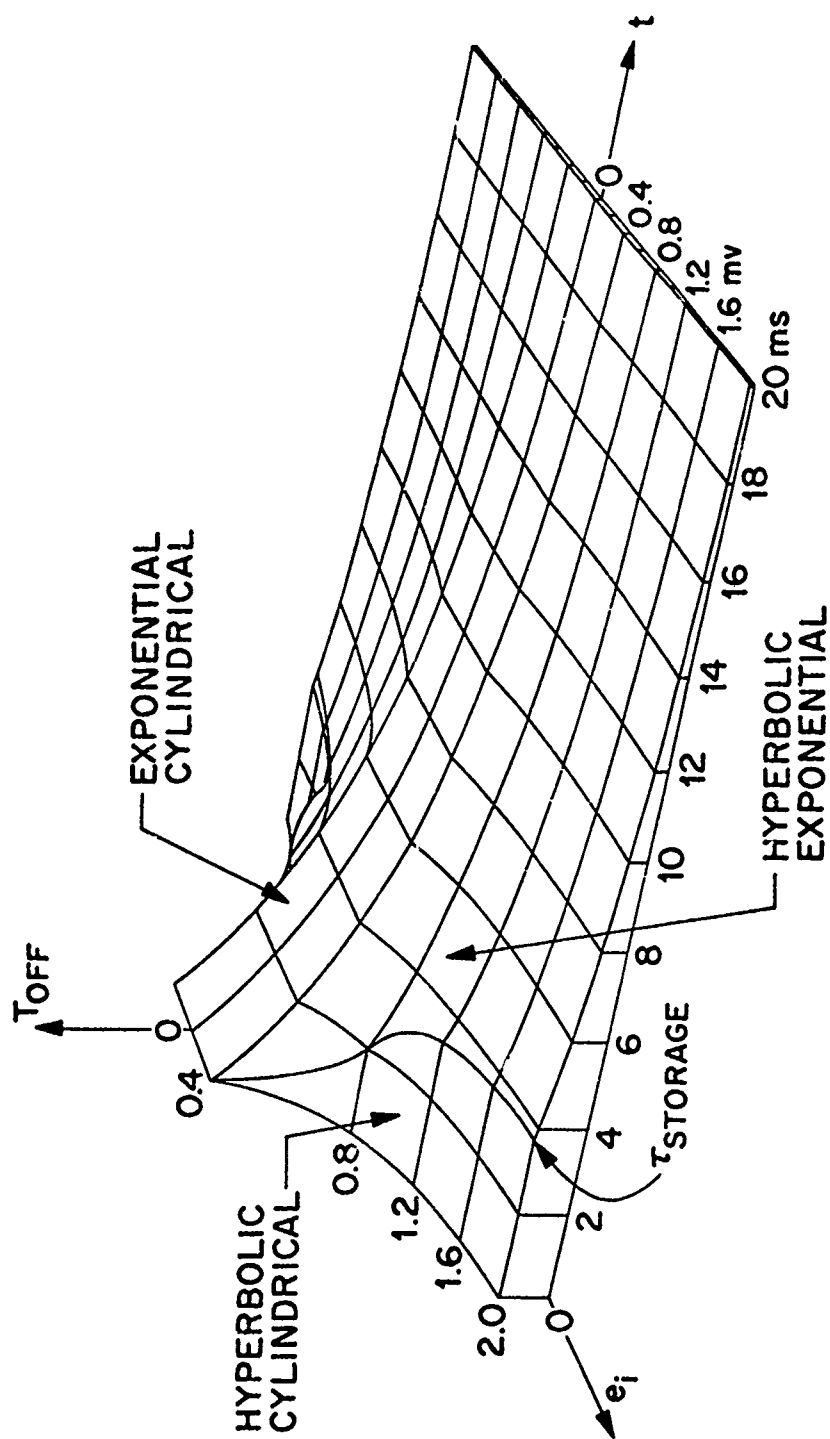


Figure 8. 709 Turn Off Response Surface

c. Artificial Terms in Transfer Functions

One attribute of the bivariable propagation technique that might not be obvious is that it requires a functional dependence of the output variable on the input variable. This is needed to steer the iteration process which drives the adjusted input variable to the proper value determined by the total system configuration. Now for digital circuits, the output is constant over a wide range of variation of the input voltage, such that a plot of P_c , the total system current response function, might be expected to have regions which are essentially parallel to the adjusted input variable axis. Consequently a Newton's method extrapolation to the next iterate would not operate. A linear interpolation method might be satisfactory since it can broaden the search range.

Another way to proceed shows unexpected and extremely important useful, and in fact mandatory, aspects of our time domain transient calculations using iteration on a time stepping basis with transfer functions.

(1) Current Transfer Functions

Up to now only surfaces or drawings showing the voltage transfer function have been presented. Yet the bivariable propagation method also needs current transfer functions. The current transfer function holds the key to establishing a satisfactory iteration convergence mechanism. A first guess at an output current would set $i_o = e_o/R_L$, where e_o is the output voltage established by T_e , the voltage transfer function, and R_L is the load resistance. For the sake of simplicity of the argument, suppose that the transition process of a flip-flop does not depend on e_i . In that case

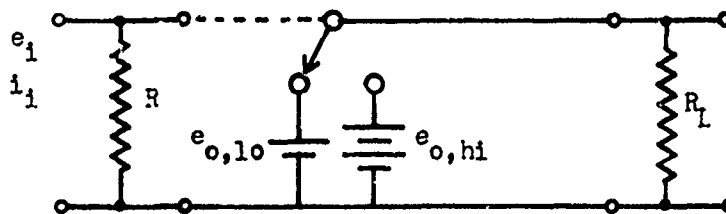
$$T_c = \frac{i_o}{i_i} = \frac{T_e e_o}{i_i R_L}$$

where

$$T_e = \frac{e_{o,hi}}{e_i} \quad \text{or} \quad T_e = \frac{e_{o,lo}}{e_i}$$

However, now there is no explicit dependence of the output variables on any change of the input variables except for the high-low transition. As a remedy, presume that the flip-flop has an input resistance R , and that we think of it as an external shunt resistor block. Connect the output node of the input resistor to the output node of the flip-flop by a dotted line that passes current but not

voltage. Then if the new value of i_1 is not the correct value the given input voltage e_1 and the resistor R , the current difference Δi will pass the shunt



resistor into the flip-flop and the output so that the output current is

$$i_o = \Delta i + \frac{e_o}{R_L} \quad (29)$$

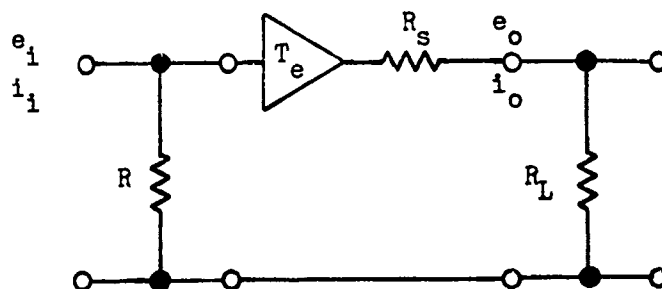
This is not the correct current physically unless $\Delta i = 0$. However, the output current does now depend explicitly on the input current, and hence, the bivariable propagation and iteration procedure can again operate regarding current as the adjusted input variable. If we imagine the constraint to be a high-impedance voltmeter connected across R_L , then the iteration process will drive the error current Δi to zero. Then the current transfer function is the physically correct transfer function for the flip-flop, and the correct current feeds the load resistor. Thus the transfer functions may contain an artificial component to assist numerical convergence so long as that term drives to zero when the correct solution is obtained.

(2) Dynamical Delay

Here let us explore one further crucial but not obvious aspect of trying to iterate around a block element containing dynamical time delay. Suppose that an input stimulus is applied and that the correct coupling of the input variables has not yet been obtained. For example, suppose that an input voltage within the linear range of the 741 op amp is applied. By inspection of Figure 3, the T_e surface, it is evident that a significant time passes before the output responds to the stimulus. This time may be several milliseconds. Suppose that the time step interval is say a fraction of a millisecond. Then many time steps pass before the output or condition of the circuit has been changed by the iteration process at the input of the device. How then can the basic iteration process work?

We believe the answer again lies in the use of two rather than one chain of transfer functions to propagate information. For the case of the op amps,

the current transfer function is obtained using a similar procedure to that used above for the flip-flop. An input resistor R acts as a shunt block element. The T_e transfer function provides the gain and the dynamics, and a source impedance R_s is in series with the load resistance R_L .



Suppose that the iteration process has not yet provided the correct coupling of the input variables to the block. Then an error current Δi as before will drive past the input resistor, through the gain element, and through to the output. Thus, the output current is as before

$$i_o = \Delta i + \frac{e_o}{R_s + R_L} \quad (30)$$

where

$$\Delta i = i_1 - \frac{e_i}{R} \quad (31)$$

and

$$e_o = T_e e_i$$

The key result is contained in (30) since the second term contains the block dynamics from the time domain transfer function surface and is delayed properly; however, the error current Δi is not delayed and propagates immediately through the block and steers the iteration process.

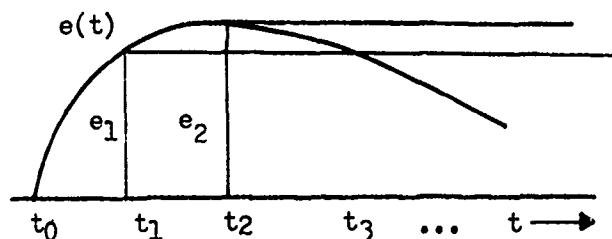
3. INPUT WAVEFORM DECOMPOSITION AND OUTPUT SUPERPOSITION

The major problem relating to nonlinear system analysis is loss of the validity of the principle of superposition. Yet as we will show shortly, it is possible to obtain reasonably accurate results using the type of nonlinear time domain transfer functions presented earlier for certain classes of nonlinearity. For other types of nonlinearities, the results are less accurate.

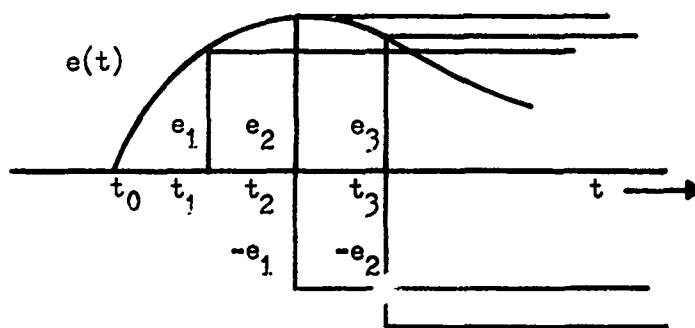
a. Step Pulse Decomposition

Let an arbitrary waveform $e(t)$ be applied to a block. Suppose that the

time stepping of the computer program samples that waveform at times $t_0, t_1, t_2,$ etc., obtaining voltages $e_0, e_1, e_2,$ etc. If just a step pulse of magnitude e_1 say had been applied, then the output at subsequent time steps could be obtained directly from the time domain transfer function surface T_e by evaluating its value at time increments measured from time of turn-on t_1 and multiplying that by the



magnitude of the step e_1 . Then the same procedure could be followed for the next step pulse e_2 and so forth. To prevent overdriving the input, since the earlier step would also be on, it is necessary to turn off step e_1 at the next time step t_2 . This is done by applying the turn-on of the negative e_1 , and add its response to the output. The decomposition of the input waveform would look as sketched below.



Taking $e_0 = 0$, the output at time step t_2 would be

$$e_o(t_2) = T_e(e_1, t_2 - t_1) e_1 - T_e(e_1, t_2 - t_2) e_1 + T_e(e_2, t_2 - t_2) e_2$$

Similarly at time step t_3 ,

$$\begin{aligned} e_o(t_3) = & T_e(e_1, t_3 - t_1) e_1 - T_e(e_1, t_3 - t_2) e_1 + \\ & T_e(e_2, t_3 - t_2) e_2 - T_e(e_2, t_3 - t_3) e_2 + \\ & T_e(e_3, t_3 - t_3) e_3 \end{aligned} \quad (32)$$

By inspection of the T_e surface for the 709 or the 741, it is seen that at time zero the vertical height is zero. Thus, the last term in (32) is zero, or at least very small. Further, it can be seen that the other terms in (32) have the input voltage at one time step multiplying the difference in the value of the transfer function evaluated one time step apart. That, however, is just the value of the time derivative of the transfer function surface evaluated at the value of the input voltage and the time since step turn on, multiplied by the magnitude of the time step. Hence, (32) becomes

$$e_o(t_3) = e_1 \left. \frac{\partial T_e}{\partial t} \right|_{e_1, t_3-t_1} dt + e_2 \left. \frac{\partial T_e}{\partial t} \right|_{e_2, t_3-t_2} dt \quad (33)$$

Or, passing to the general case,

$$e_o(t) = \sum_j^J e_j \left. \frac{\partial T_e}{\partial t} \right|_{e_j, (J-j)dt} dt \quad (34)$$

where J is the latest time step index value. Equation (34) is the basic form of the convolution principle or the principle of superposition which has been used in the program SAP to compute the output waveforms which will be presented herein.

While the above mathematics is mandatory in order to perform a computer analysis, it does not give us any direct insight into what is happening on the transfer function surface. It is not difficult to obtain an understanding using that surface which is very helpful in treating transient problems.

Suppose (34) is interpreted in terms of either the 709 or the 741 voltage transfer function surface, Figure 2 or 3. Let the first pulse decomposition element e_1 be referred to as a partition. Then the output in time following its turn on is mentally visualized by coming out laterally on the e_1 axis to the value of input voltage e_1 and then imagining the partition moving on isovoltage portion of the surface forward in time parallel to the time axis. That partition continues to contribute to the output until it reaches the portion of the surface where the time derivative of the surface goes to zero.

The next partition moves back along that isovoltage part of the surface corresponding to e_2 and again contributes until the time derivative of that part of the surface vanishes. Convolution is just the continual running summation of successive partitions contributions to the output. Such a process is theoretically justifiable only for linear elements.

b. Computed Results for the 709 and the 741

With the last statement firmly in mind, let us now examine some results. First with a 1.0 mv step pulse applied, and using 0.1 ms time steps, the computed output voltage is shown in Figure 9. The input waveform is shown in the top half, and the computed output in the bottom half. From the figure it is immediately evident that superposition can be used with reasonably good validity for saturation class of nonlinearities.

To most easily understand the way the constant output limit is achieved in the calculation, think how the partitions moving back along isovoltage trajectories on the T_e surface contribute to the output. By inspection of the surface in Figure 2, it is seen that for $e_i = 1.0$ mv it takes about 4 ms for the output to saturate and for $\dot{T}_e \rightarrow 0$. The input waveform in Figure 9 is not a step but takes about 1 ms to turn on. The partitions for $0 \leq e_i \leq 1.0$ mv travel along the valley portion of the surface. Those in the linear range contribute for as long as 20 ms from time zero. The partitions above the linear range contribute for shorter time intervals. The reason the output holds up at a nice, relatively constant value is that as the first partitions reach the flat hyperbolic sheet and cease to contribute, the fact that the waveform is still turned on means that subsequent partitions are starting to climb up the surface thereby maintaining the output constant in the steady state.

The variation in the output voltage in Figure 9 is about $\frac{1}{2}$ v in 10 v; thus the superposition principle as applied using (34) is valid to 5% for this case.

c. Computation of Storage Effects

Earlier the fact that storage phenomena are not included in a turn on surface, led to the portrayal of the storage time surface and the turn off surface in Figures 7 and 8 for the 709 op amp. Here, the way this is taken into account relevant to the superposition tactics of eq.(34) will be presented.

Suppose that a waveform is applied which drives into the nonlinear range above the input clipping voltage e_{clip} . As before pulse decomposition is applied and a sequence of partitions travels up the turn on surface. Now however when the waveform turns off, logic in the computer program determines how long the voltage had held up at the levels corresponding to the values at which the partitions

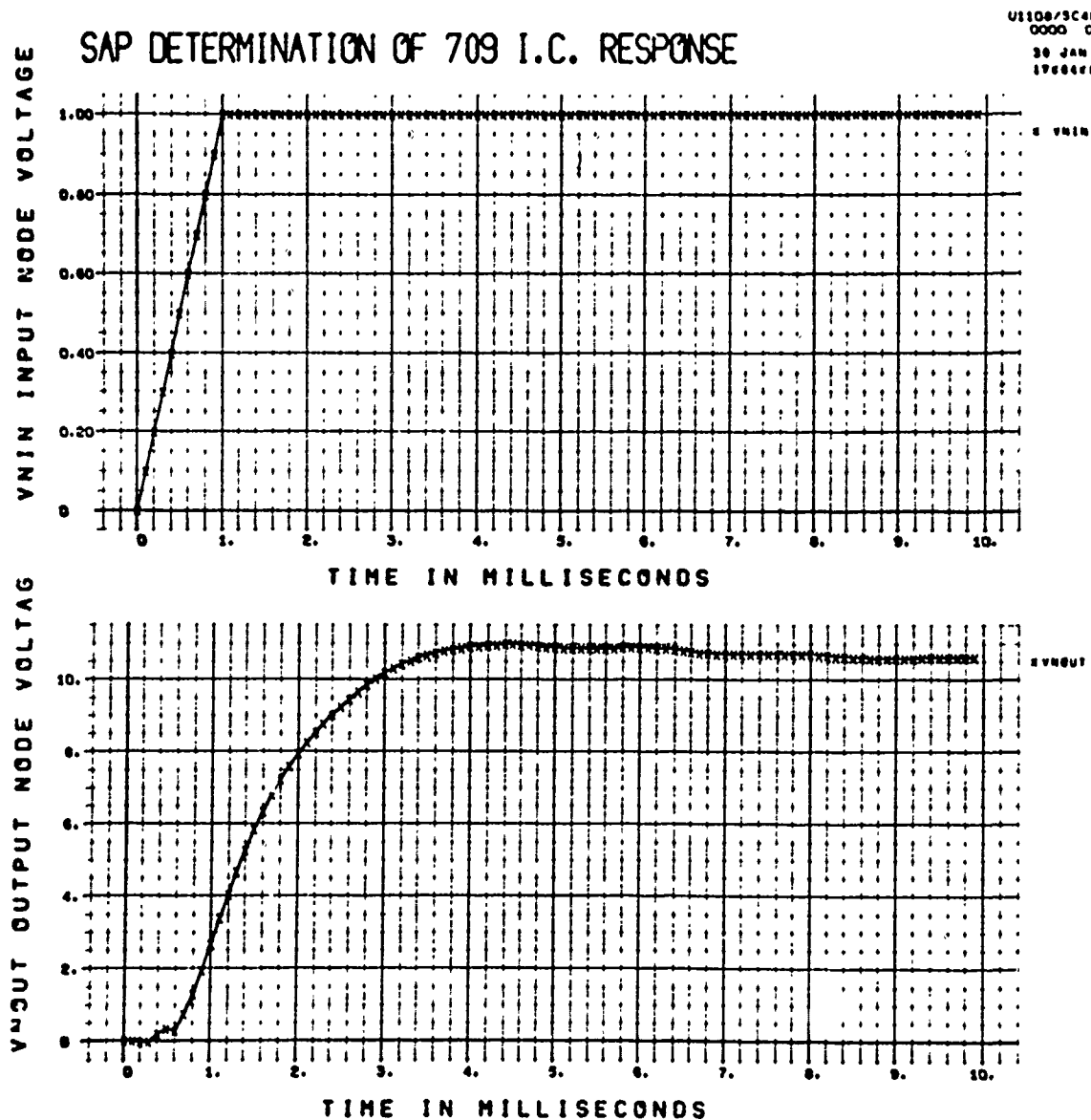
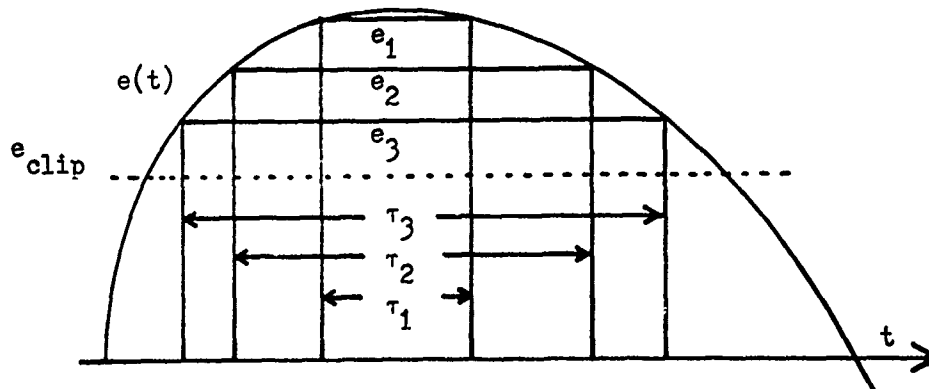


Figure 9. Calculation of Response of a 709 Op Amp to Step Pulse Using Bivariable Propagation with a Nonlinear Response Surface

were generated at the various time steps. Knowing these equivalent varying amplitude, varying length pulses, which constitute the input waveform as shown below:



We then are in a position to calculate the total storage time by the following computer procedure:

- decompose input into separate saturating pulses,
- look up storage time for each pulse (this depends on pulse amplitude and length),
- calculate turn off transfer function,
- combine individual responses using

$$e_o(t) = \sum_j \{ e_j - e_{j+1} \} T_{off}(\tau_s, e_j, t - t_j) \quad (35)$$

time step increments
storage time
time since pulse turn off

The reasoning behind eq.(35) is somewhat different than simple superposition by virtue of the presence of the e_{j+1} term in the multiplier of the transfer function. The easiest way to understand the term $e_j - e_{j+1}$ is to regard it as a linear interpolation of the effect of different pulse heights. For example, if e_{j+1} is small, say about zero, then we want the maximum charge storage term to be brought in from the saturating pulse e_j ; whereas, if $e_{j+1} \approx e_j$, we need only a small contribution from that pulse.

To see how successful or otherwise this approach is, compare the results shown in Figure 10 calculated using eq.(34) alone with those in Figure 11 which was calculated using both (34) and (35). In other words, Figure 10 used just the turn on transfer function surface while Figure 11 used the storage time surface and the turn off surface in addition.

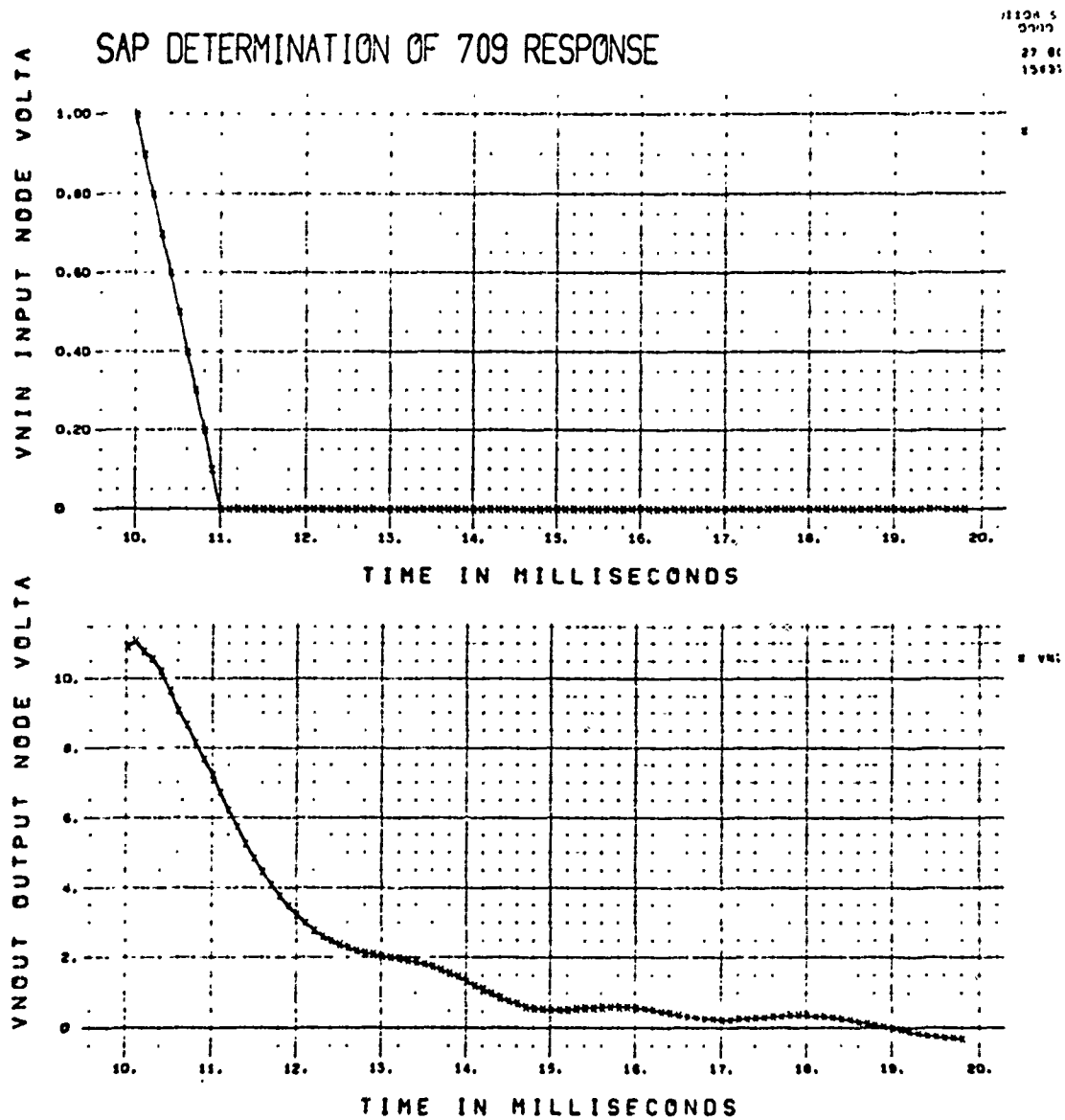


Figure 10. Turn off Portion of 709 Response Calculated Without use of Turn off Surface for Storage Effect

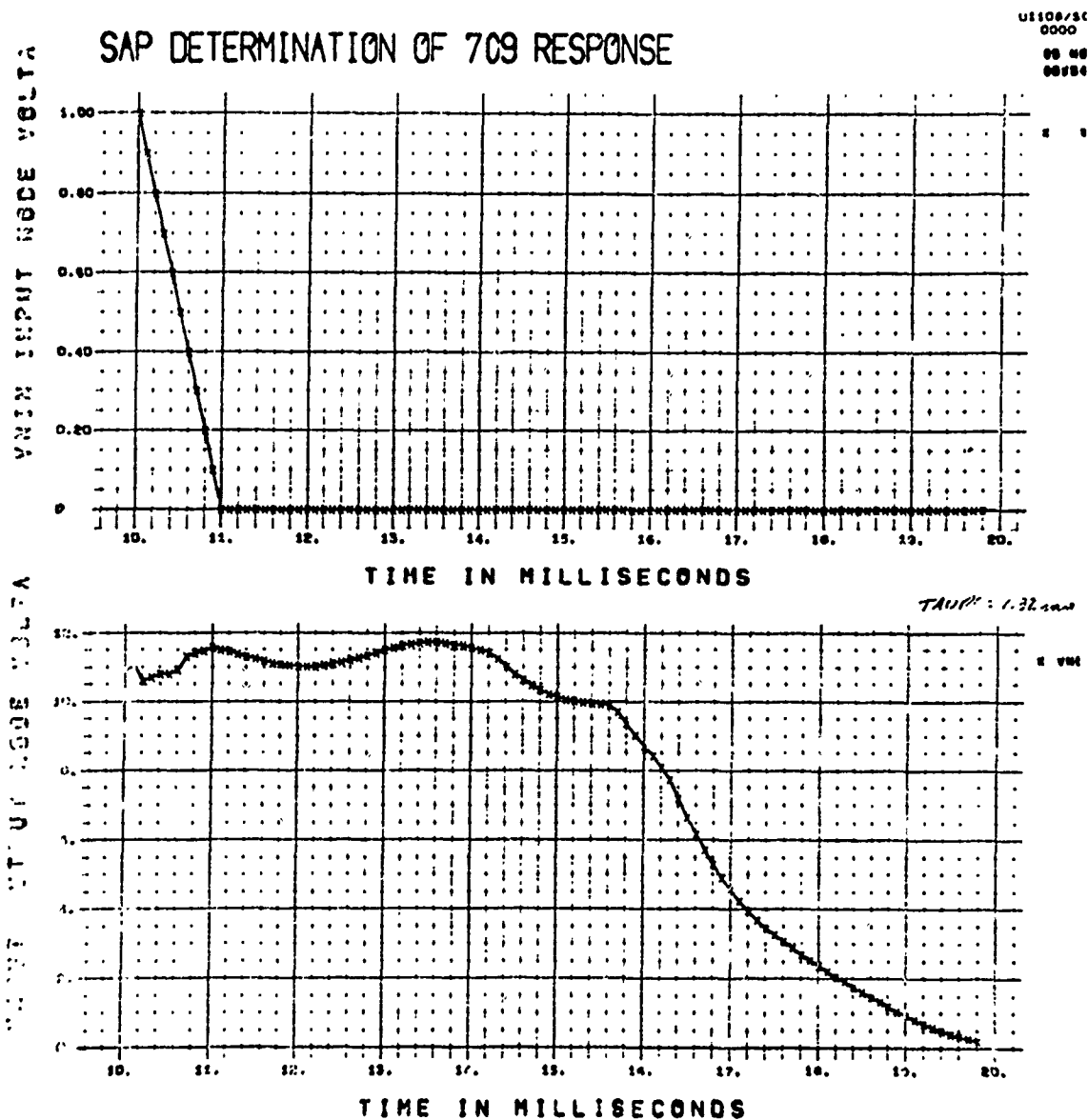


Figure 11. Turn off Portion of 709 Response Calculated
Using the Turn Off and Storage Time Surfaces

It is apparent from Figure 11 that the storage effect has been calculated with a modicum of success. The variation of the output in the supposedly constant saturation region is ± 1 v out of about 10 v. It thus appears possible to use a version of the principle of superposition and obtain results of about 10% accuracy, for charge storage class of nonlinearity.

The results which are shown in Figure 11 used the computer generated fit to the storage time surface that is shown in Figure 7. Figures 10 and 11 show the calculated turn off of a 1.0 mv pulse whose rise is given in Figure 9.

SECTION III

GATHERING THE EXPERIMENTAL DATA

The experimental information on the radiation sensitivity of the devices was taken during three trips to Kirtland Air Force Base and Sandia Base in Albuquerque, New Mexico, during April and May, 1971.

The fixtures used for the flash X-ray exposures at AFWL consisted of sockets affixed on one side of one-half of a lead brick. A hole was drilled diagonally through the brick large enough for the cable circuit cable drivers and Tektronix CT-2 current transformers which were mounted in a Pomona chassis box on the rear of the brick. Thus, the lead length was approximately $2\frac{1}{2}$ inches to the output coupling circuitry. The second NAND gate in the 9704 structure was used as the load for the gate under test. Thus, the load is simultaneously irradiated in good simulation of a weapon burst.

The non-irradiated electrical measurements on the devices were taken at Stewart Research Enterprises in Los Altos, California using the following measurement equipment ensemble:

Pulse Generators - Data Pulse 101, E-H 131, G.R. 1217-C

Oscilloscopes - Tektronix 531 and 535 with CA and 1S1
(Sampling) plug-ins.

Digital Voltmeter - HP3459A

X-Y Recorder - Moseley Model 2

For the fast pulse measurements on the gate circuits, the X-Y recorder was used to record the DC voltages from the 1S1 sampling plug-in. For the slower pulse measurements on the operational amplifiers, Polaroid pictures were taken of the oscilloscope waveforms.

The conditions used during the 741 operational amplifier measurements at the Kirtland flash Xray (FXR) were:

$$V_{cc} = \pm 15.00 \text{ volts}$$

$$R_1 = 820 \text{ ohms (on both inverting and non-inverting inputs)}$$

$$R_F = 50 \text{ megohm}$$

$$R_L = 2 \text{ kilohm}$$

During the measurements at the Sandia pulse reactor (SPR) the long cable runs necessitated lowering the feedback resistor in order to achieve stability.

1. OPERATIONAL AMPLIFIER TEST CIRCUITS

a. Sine Wave - Non-Radiation Transfer Characteristic

The electrical input - output transfer characteristics were made using the circuitry given in the block diagram of Figure 12a, and specific connections to the op amp shown in Figure 12b. The test sequence utilized was:

1. Insert op amp in test socket.
2. Adjust offset null potentiometer for zero volts output.
3. Increase oscillator output until op amp output displays limiting.
4. Photograph transfer characteristic with Polaroid camera.

b. Op Amp Response

The pulse response of the op amps were made with the block diagram shown in Figure 12c. The test sequence consisted of the following steps:

1. Adjust offset null potentiometer for zero volts at output.
2. Adjust attenuator and pulse generator to provide desired input signal, and
3. Observe and photograph during reactor pulse the waveform of the output signal.

2. DIGITAL LOGIC GATE TEST CIRCUITS

a. Transfer Characteristics - Before and After Reactor Pulse

The transfer characteristics of the gates were obtained using a curve tracer connected as shown in Figure 13a. The measurement steps are:

1. Note that the oscilloscope is referenced to +5 v for this measurement.
2. Display Base voltage (0.5 v/div) vs Collector voltage (0.5 v/div) as a plot of e_o vs e_i .

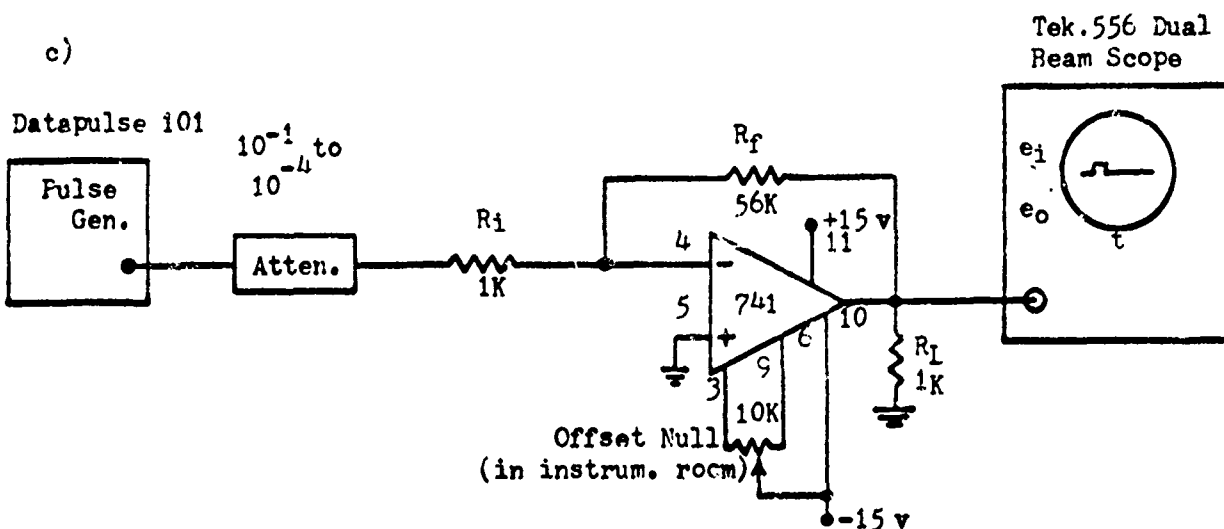
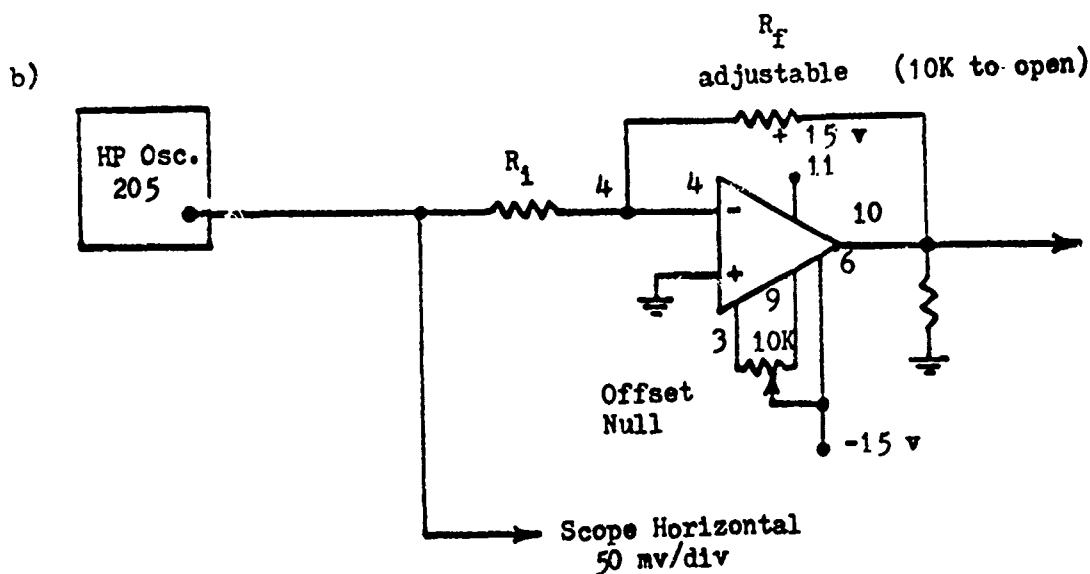
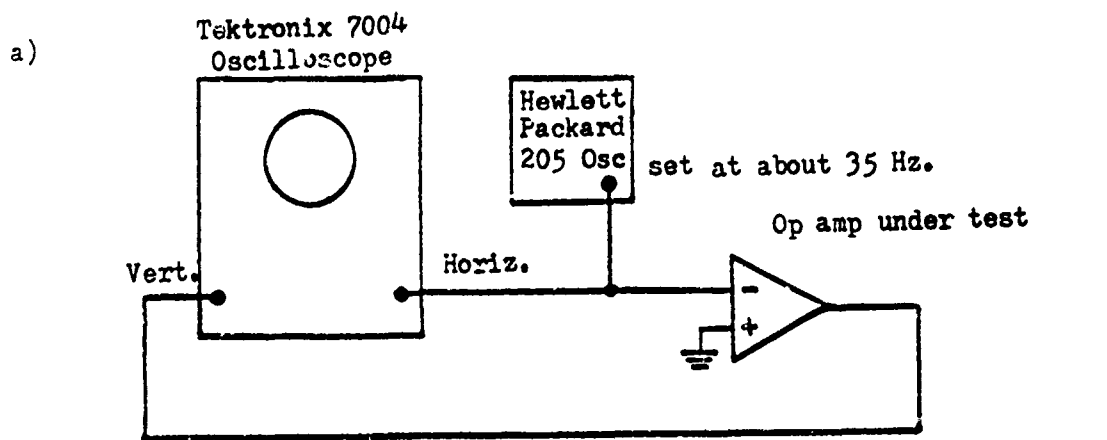


Figure 12. Block and Circuit Diagrams Used for 741 Op Amp Input-Output and Step Pulse Response Determinations

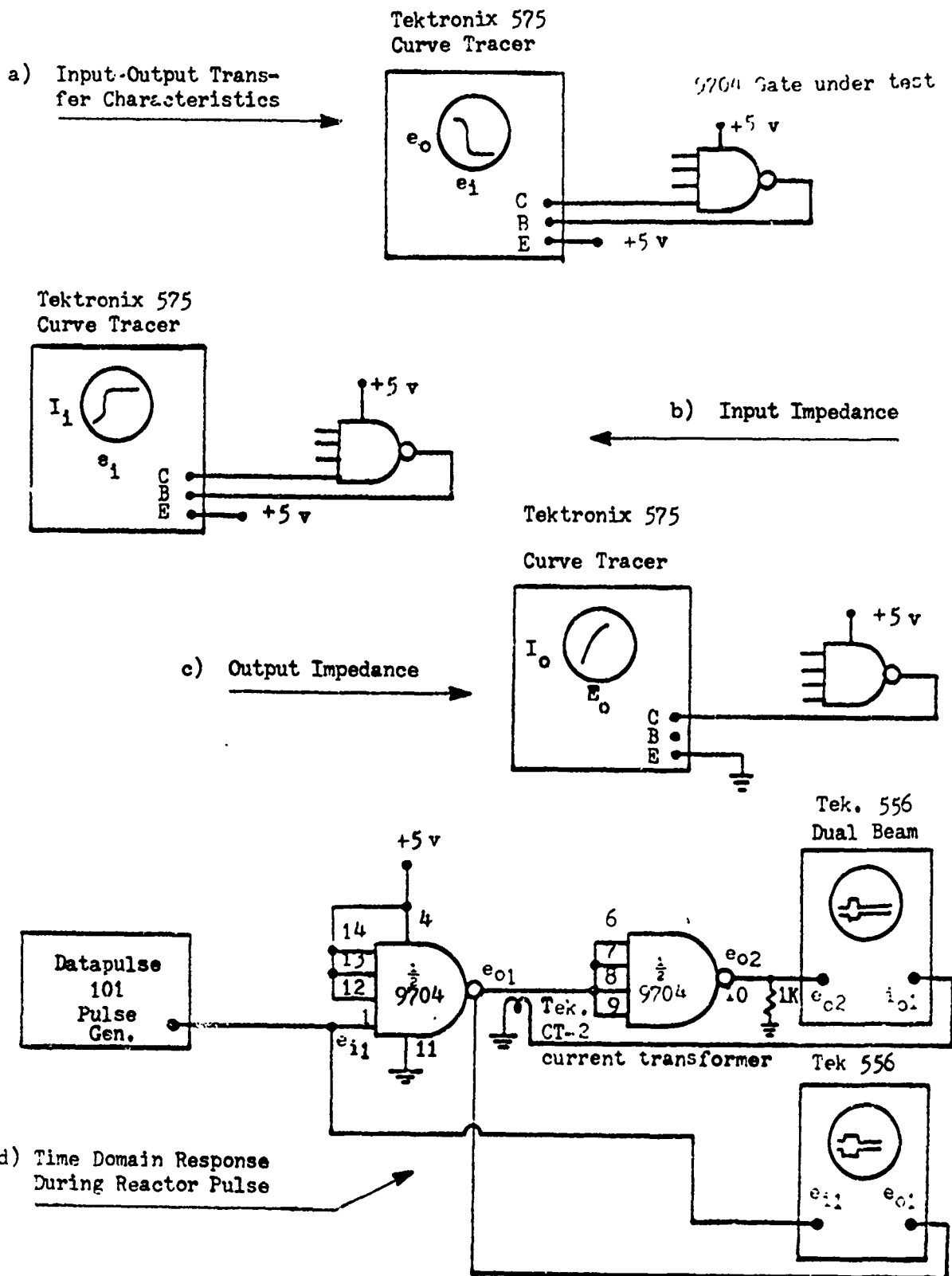


Figure 13. Circuits Used for Measurement of 9704 NAND Gate

3. Set collector sweep at minimum and negative polarity with the dissipation limiting resistor set at 100 ohms,
4. Hold the "zero current" switch in the open or zero current position with a rubber band (this disables the normal base step generator).
5. Increase the collector sweep till e_1 sweeps from +5 v to zero, and
6. The transfer characteristics is now observed showing shifts in the gate ON or OFF voltage, and the transition voltage.

b. Input Impedance

The gate input impedance was obtained with the circuit shown in Figure 13b, and the following measurement steps:

1. For this measurement, the oscilloscope is again referenced to +5 v.
2. Change the vertical display to collector ma., (0.2 ma/div)
3. The display is now input current vs. input voltage.
4. Look for excessive leakage current near +5 v input, shifts in transition voltage, and current drawn at zero input voltage.

c. Output Impedance

The impedance test sequence used the connections shown in Figure 13c, and the following measurement steps:

1. The oscilloscope is now referenced at ground instead of + 5 v.
2. Set Collector Sweep at minimum required with positive polarity.
3. Set the zero to the lower left and bottom of the display with the positioning controls.

4. Change the vertical display to 20 ma/div.
5. Increase the Collector Sweep until e_o sweeps from zero to approximately $1\frac{1}{2}$ or 2 v, taking care that the pattern does not change due to device overheating, and
6. Observe the display for change in saturation voltage.

d. Gate Time Domain Response During Reactor Pulse

Circuit configuration used for the gate pulse response during irradiation is given in Figure 13d.

SECTION IV

PROCESSING OF EXPERIMENTAL DATA

The techniques used for processing the raw data into the final function surfaces underwent an evolutionary change during the course of the research. The first method utilized hand made perspective drawings, and was applied to the electrical and radiation response data for the 741 op amp. This technique offers one principal advantage-- the creation of the final surface is an act of decision between noisy data from different test devices, and different experimental situations. Judgement can be used to weigh the significance of a local anomaly as compared to a real trend. As a result the initial surfaces which were evolved looked "good" in the subjective sense. However, the large amount of time required to process the data by this approach makes an alternative mandatory.

Thus, the decision was made to resort to computer processing techniques exclusively for the response surfaces of the 9704 NAND gate. This required extension of the initial machine programs which used least square fitting of three dimensional cubics over each segment of a basic 20 by 20 fitting grid. The cubics are matched in terms of function and derivative value in the spline sense at the grid interstices. The extension required writing a linear interpolation routine to fill in regions of the data space which lacked sufficient experimental data. Then other routines were adapted that performed wild pointing and/or smoothing on the data set. The number of smoothing passes varied in the processing. The effect it has on the result will be shown later.

While machine processing is certainly much easier and faster when the routines that do the real work are written and available, judgement and choice really can not be forgotten. Just how much smoothing is "right" in a certain sense is an esthetic question, rather than a technical one. Let us now see why as we look at different methods and their results.

1. RESPONSE FUNCTION SURFACES OF THE 741

Figure 3 shows the voltage transfer function surface of the 741 with respect to step pulse inputs. This surface is much better behaved than the similar surface for the 709 which was presented as Figure 2 earlier in this report. Yet there are obvious similarities between the two. This drawing shows the faster response of

the device as it is driven hard into the non-linear range by the boundary between the hyperbolic sheet and the rising front portion moving to earlier and earlier times at high signal amplitude. This surface was taken with the device mounted in the fixture made for testing at the Kirtland FXR. The conditions of the measurements are:

$$V_{cc} = \pm 15.00$$

$$R_1 = 820 \text{ ohms (on both inverting and non-inverting inputs)}$$

$$R_f = 50 \text{ megohm}$$

$$R_L = 2 \text{ kilohm}$$

The waveforms were photographed on a Tektronix 535 oscilloscope. The pulses were supplied by a General Radio 1217-C pulser driven at about 5 Hz pulse repetition frequency. Of significant interest to this modeling activity is the very small storage of the 741 as compared with the 709, at least prior to neutron irradiation. After neutron irradiation, storage is evident as shown in the Figures 86 and 87 of Appendix , where the top-hat or butterfly effect of the irradiated 741s is discussed. We will use a three-dimensional computer fit of splined cubics to the T_e surface.

Figure 14 is the radiation transfer function of the 741 as measured using the AFWL Febetron. This surface is more of an impulse response surface, as compared to step response of Figure 3. The vertical axis, labeled T_ϕ , is the ratio of the output voltage to the $\log_{10} \phi$, where ϕ is essentially $\dot{\psi}$, in rad/sec. Logarithmic time and dosage axis are used because of the large ranges in these variables. It is evident that the 741 is a fast device indeed in terms of radiation effects; whereas, the response to electrical stimulus is relatively slow. The radiation effects are seen on 50 ns time scales, whereas the T_e surface shows phenomena on millisecond or tens of millisecond time scales. This is significant in assessing the effects of inverse feedback on the radiation response. From the form of the two surfaces, we expect inverse feedback to hardly reduce the fast nanosecond transient at all, but to significantly reduce the long time hang-up of the device due to radiation.

It is evident by inspection of Figure 14 that the radiation response does not scale simply with dosage. At low dosages, say less than 3×10^8 , the response increases until the positive peak is saturated at the supply voltage. It takes another

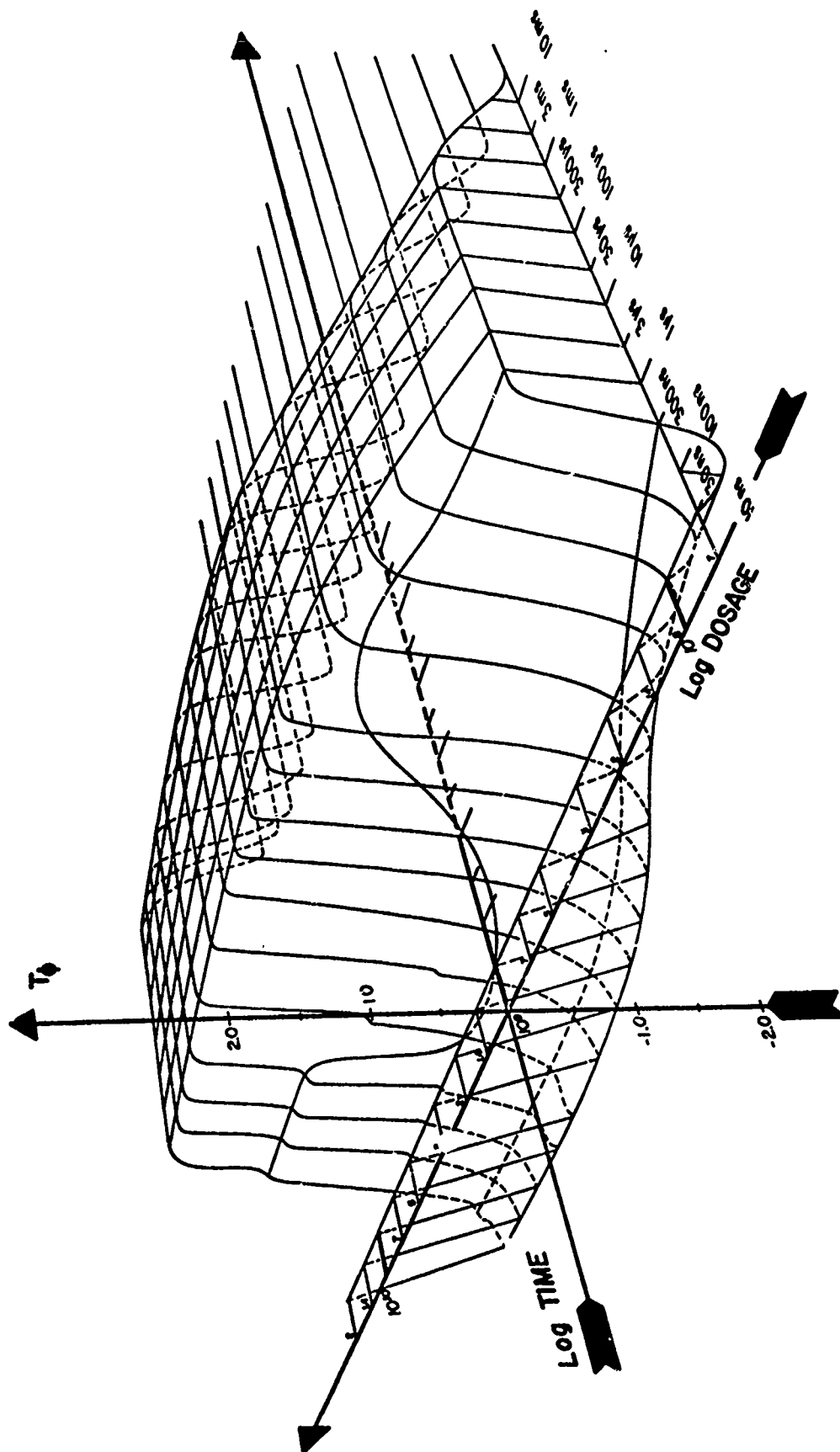


Figure 14. Voltage Transfer Function of the 741 Op Amp due to FXR Irradiation

order of magnitude of radiation to saturate the negative peak. Also interesting is the way the surface recedes on the rising cliff part at intermediate dosages. This shows up most clearly in the drawing by the way the $1\ \mu\text{s}$ isochronal (constant time line) plunges downward. The pulse hang-up time also increases at large dosages. Observe this by noticing how the far back top edge angles backward with respect to the $300\ \mu\text{s}$ and $1\ \text{ms}$ and $3\ \text{ms}$ isochronals. A small flat porch-like structure is evident on the rising cliff at high dosage. This may be due to structure in the FXR pulse, although that would seem unlikely so far in time along the response surface.

The final response function surface for the 741 is based on data taken in May 1971 at the Sandia pulse reactor in Albuquerque. In order to show the phenomena occurring at high dosage, three separate sections have been drawn. Again log time and log dosage axis have been used. The vertical axis, T_{ϕ} , is the same as for Figure 14. The dosage is integrated neutron flux with energy more than 10 kev. The pulse length of the reactor is about 50 - 100 ms, and a typical pulse is shown in Figure 85 of Appendix I. A phenomenon evident in Figure 15 is the dramatic destruction of the device at high dosages, as evidenced by the fall off in T_{ϕ} above 10^{13} . Thus, we see that this type of representation of a devices radiation transient response at the same time tells us something about the permanent degradation.

Figure 15a shows the front portion of the high dosage transient response. Note that at high dosage the device responds faster, as shown by the isochronals climbing upward on the front cliff. Above about $1.4 \times 10^{12}\ \text{nvt}$, a valley occurs after the rising cliff part of the waveform. This valley increases in depth, reaches a minimum, and then starts coming up again, both with respect to time and dosage.

Figure 15b shows the dropping part of the transient waveform toward the minimum of the valley. Note the porch like structure evident at about $150\ \mu\text{s}$. At high dosage, degradation is also clearly evident on the minimum of the valley.

Figure 15c shows the rising positive portion of the transient response after the valley, and how this is effected at high dosage by the device destruction or degradation. Above about 5×10^{12} , the pulse hang-up time increases greatly (remember this is a log time scale), extending out to about 40 ms. The extent of

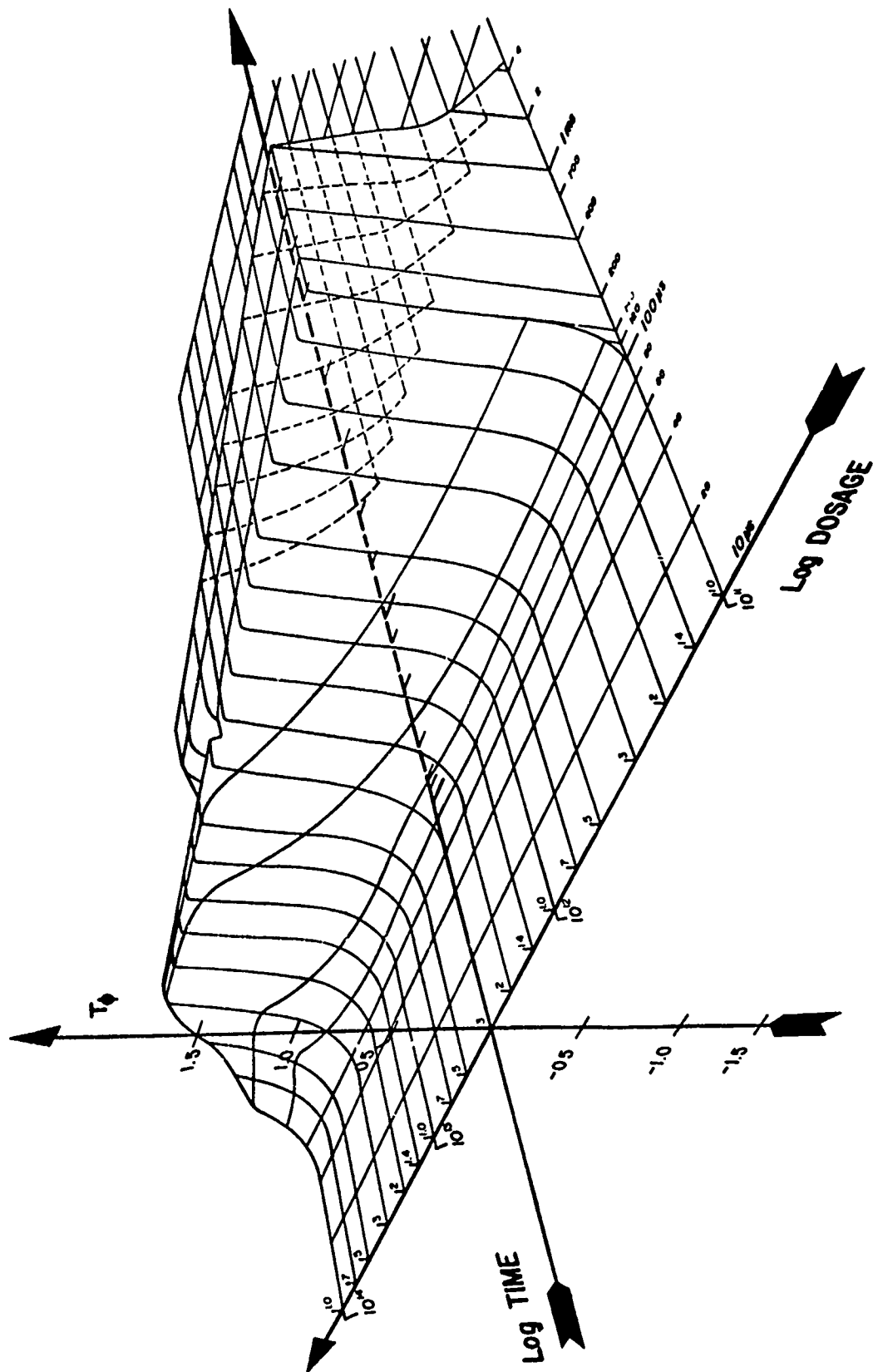


Figure 15a. Radiation Transfer Function Surface for the 741 Operational Amplifier as Obtained with the Sandia Pulse Reactor. Note Degradation at High Dosage.

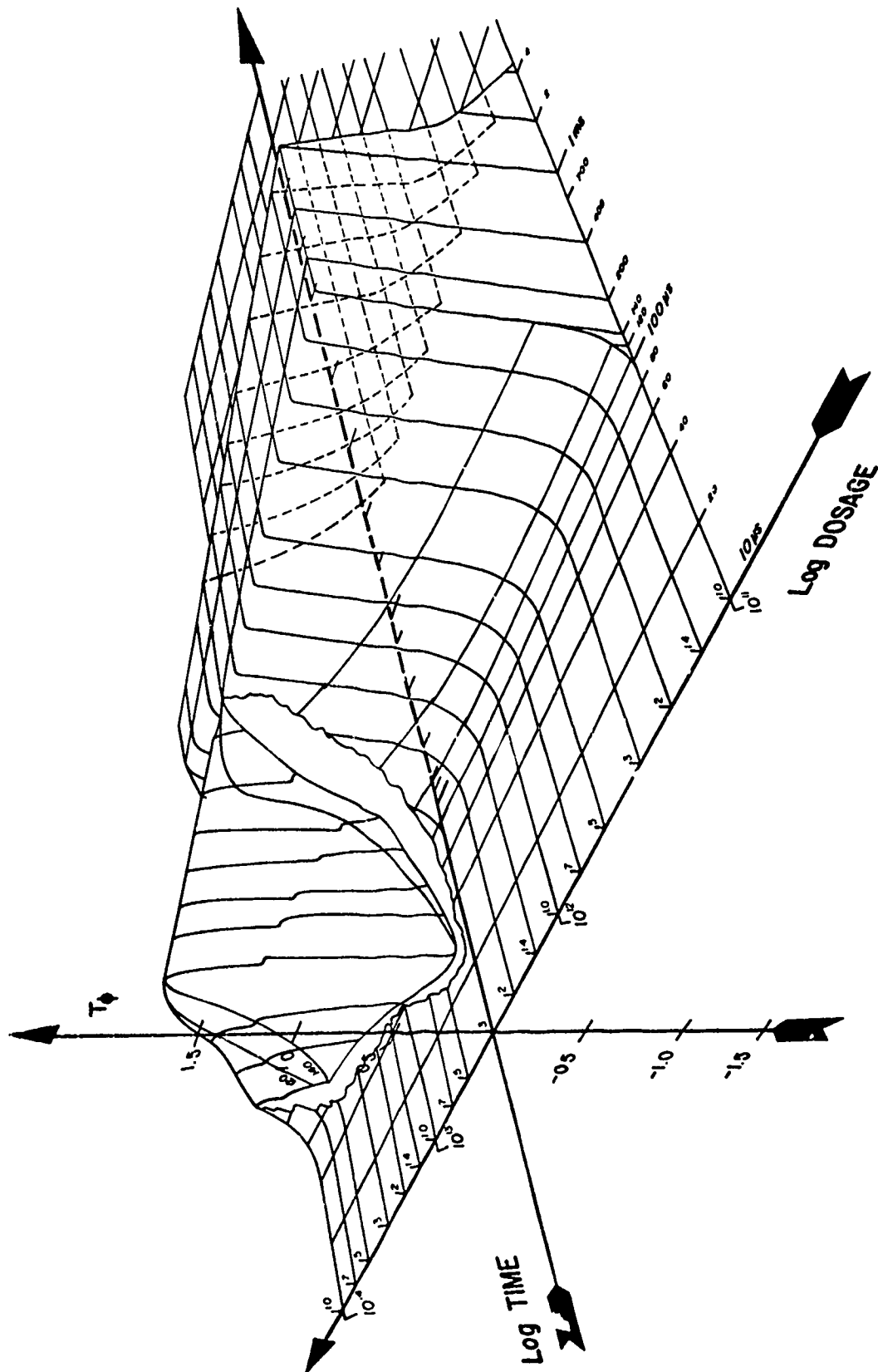


Figure 15b. Radiation Transfer Function Surface of the 741 Op Amp Taken at the SPR with the Front Surface Cutaway Showing Surface Dropping Toward a Valley. Note Degradation at High Dosage.

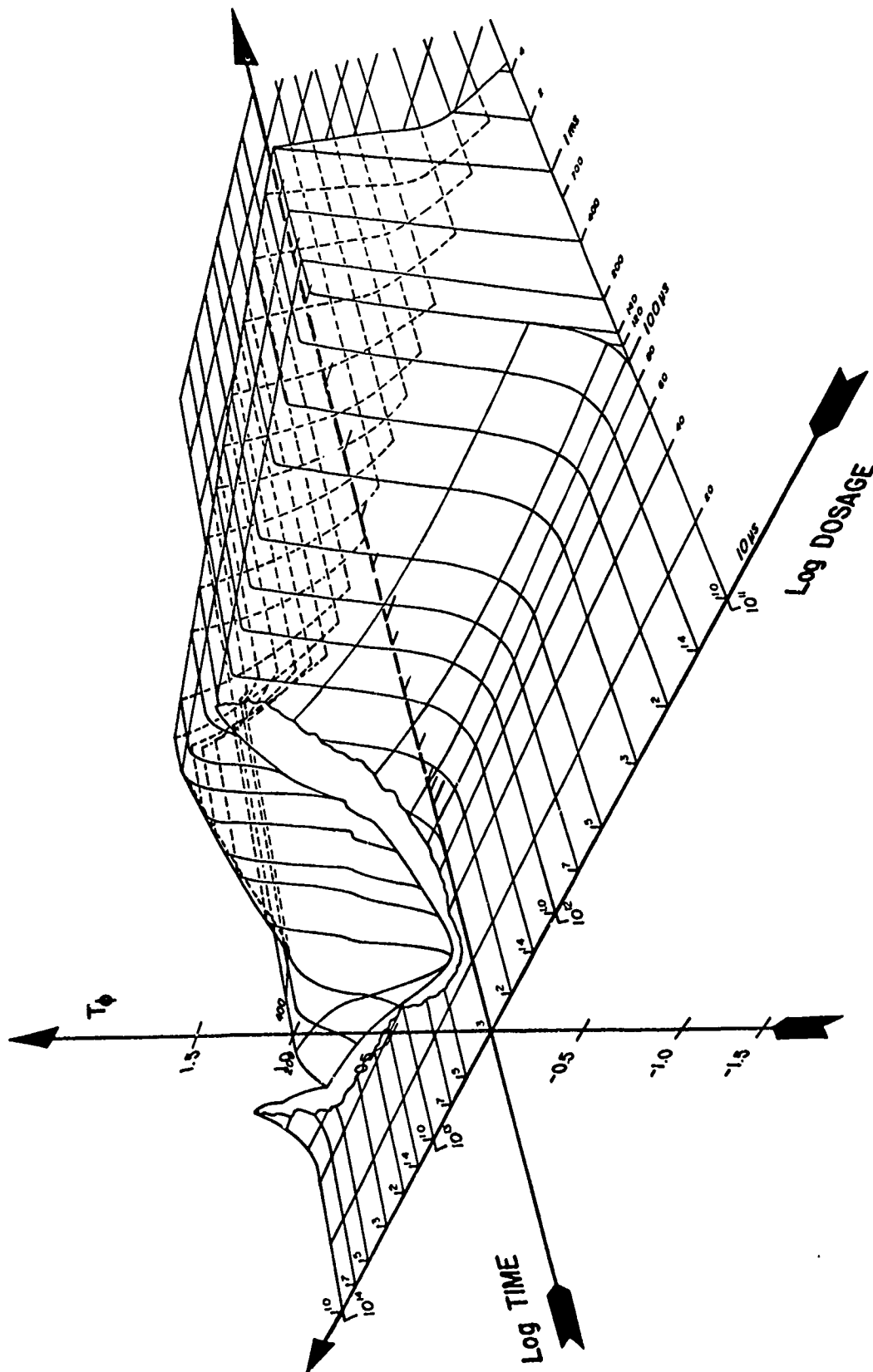


Figure 15c. Radiation Transfer Function Surface of the 741 Op Amp Taken at the SPR with the Front Ridge Cutaway Showing the Rise from the Valley to the Final Ridge. Note Degradation at High Dosage There Also.

this increase in the pulse hang-up time will be more evident in the later computer plots where the perspective view is rotated so that rear portion is visible.

2. HOW THE SURFACE DRAWINGS WERE MADE

The drawings of Figure 15 were not made by a computer. They were drawn using three point perspective grids at four times normal size. The actual dimensions of the original artwork of the drawing is 3 feet high by 4 feet wide. This permits reasonable accuracy to be obtained in plotting the experimental data. Normally five straight-edges are required to translate a point from the coordinate plane to a point in space on the drawing. The data from the original pictures usually does not correspond to nice even values of time or dosage. Hence, the isochronals and isodose lines are created by interpolation between the actual experimental results. To a certain extent this act of interpolating is an exercise in judgement. This is beneficial since nuances of individual devices can be weighed one upon another. However, in actuality there was much more commonality than difference amongst them. Probably the most variable aspect was the offset drift of the op amps in time. For the sake of these drawings, random offset drift was zeroed out.

While considerable effort was required to draw these response surfaces by hand, it is conceivable to couple an on-line computer to the instrumentation during the data acquisition phase at the FXR and the SPR, and create these surfaces automatically using standard three-dimensional surface fitting and projection routines. With the basic data stored, different projection angles can be chosen to optimize the perspective seen. We did not have that advantage for example in drawing Figure 15, where only after most of it had been drawn did we realize that the dosage axis reversed would show the valley behind the first cliff more readily.

The basic data for computer fitting the surfaces for the 741 were obtained from the hand drawings by means of three-dimensional perspective parallelograms. These are used to carry the positional data inherent in some point on the surface back to intersection points along the three fundamental perspective axes. This process is just as time consuming as the original data plot operation was initially. Also it is difficult to avoid errors in the positioning and sequencing of the straight-edges. About 200 data points are the minimum required to represent a surface if symmetry is present and two to three times that if the surface is complex with no symmetry.

3. COMPUTER PROGRAM FIT3D

The basic strategy used to fit and plot the raw data is to establish a two-dimensional grid in the x-y plane of the data set. Normally we will adopt the following labeling convention for the axis, and take them as a right-handed coordinate set:

- x-axis - time or log(time),
- y-axis - input stimulus variable, such as voltage or log(dosage).
- z-axis - transfer or response function.

The dimensionality of the fitting grid is taken as NX by NY, where in all cases that were treated, NX = NY = 20. The dimensionality of the plotting, NGX by NGY, grid was increased to the range of 40 to 50 along each axis in order to achieve reasonably valid plots since the plotted line segments are straight line segments. The fitting grid itself can be coarser since the fitting functions are three-dimensional cubics. The question of how large or small the fitting grid really must be is determined by the degree of deviation which is acceptable between the experimental data, and the machine generated fit. This will be pointed out later in comparing the computer fits to the hand-drawn surfaces.

The computer program FIT3D is a collection of 41 subroutines that create the three-dimensional fitting functions and then plot the resultant surface as an orthographic projection. This file of programs was continually improved and extended during the course of the research program. The main program is FITIT, and the sequence of functions which it calls out are shown in the flowchart of Figure 16.

Three card decks can be punched out by FITIT if desired. These are called ZDATA, ZCOEF, and ZCALC and represent

- ZDATA is a re-ordered version of the inputted data
- ZCOEF is a set of NX x NY cards containing the triplet of numbers $(z, \frac{\partial z}{\partial x}, \frac{\partial z}{\partial y})$ at each interstice of the fitting grid.
- ZCALC is a set of (NGX x NGY / 5) cards with z for each interstice of the plotting grid.

The ZCOEF deck or the ZCALC deck may be used to create plots subsequently without the necessity of running through the entire fitting process again. The ZCOEF deck is used in the subsequent program SAP and is hence more versatile, and since it is smaller than the ZCALC deck perhaps preferable for plotting. In some cases it does require driving through the interpolation routine SURFB again, but this only requires five seconds of machine time, which is not excessive.

The logic path taken in FITIT, and the control of the punch operations on the above decks, is determined by the data cards first read in by FITIT. Since the basic input data to FITIT can be brought in from the ZCOEF deck or the ZCALC deck, choice of which one is read is determined by the fourth data card. This card also controls the calling of FILLZ, which performs the creation of least square adjusted data points on the interstices of the fitting grid, or the calling of NICER which sets up the smoothing operations, or the calling of INTERP which can insert additional data points into regions which are deficient in terms of actual experimental data by using linear interpolation between valid data points.

A listing of the initial card reading statements of FITIT is contained in Figure 17. A typical data card deck makeup is presented in Figure 18.

The interpolation of values of z given (x,y) coordinates of a plot point is performed by the routines SURFB and SURFC using the information contained in the ZCOEF array created by SURFA. SURFB locates the input plot point within the proper one of the 400 grid rectangles, and performs interpolation in the three-dimensional cubics using the procedure of Ferguson*. If the plot point happens to lie in the same grid cell as the point of the prior call, the Ferguson coefficients are not recomputed, rather SURFB drops through calling SURFC which uses the Ferguson coefficients of the last pass. These interpolation routines are reasonably fast, performing approximately 500 interpolations per second of Univac 1102 CPU time.

Initially the three-dimensional plots were made using a University Computing Company routine PROJECT. The routine had some operational features which were not satisfactory (such as only plotting positive z values), and did not label and scale the axes of the figure. So, instead the routine DRAW3D was adapted to be compatible

J. Ferguson, J.A.C.M. 11, #2, April 1964, p. 221-228.

```

340 C      GO TO ZCALC, FOR PLOTTING BY PLOTSD AND DRAWSD.
350 C
360 C
370 C
380 C
390 C
400 C
410 C
420 C
430 C
440 C
450 C
460 C
470 C
480 C
490 C
500 C
510 C
520 C
530 C
540 C
550 C
560 C
570 C
580 C
590 C
600 C
610 C
620 C
630 C
640 C
650 C
660 C
670 C
680 C
690 C
700 C
710 C
720 C
730 C
740 C
750 C
760 C
770 C
780 C
790 C
800 C
810 C
820 C
830 C
840 C
850 C
860 C
870 C
880 C
890 C
900 C
910 C
920 C
930 C
940 C
950 C
960 C
970 C
980 C
990 C
1000 C
1010 C
1020 C
1030 C
1040 C
1050 C
1060 C
1070 C
1080 C
1090 C
1100 C
1110 C
1120 C
1130 C
1140 C
1150 C
1160 C
1170 C
1180 C
1190 C
1200 C
1210 C
1220 C
1230 C
1240 C
1250 C
1260 C
1270 C
1280 C
1290 C
1300 C
1310 C
1320 C
1330 C
1340 C
1350 C
1360 C
1370 C
1380 C
1390 C
1400 C
1410 C
1420 C
1430 C
1440 C
1450 C
1460 C
1470 C
1480 C
1490 C
1500 C
1510 C
1520 C
1530 C
1540 C
1550 C
1560 C
1570 C
1580 C
1590 C
1600 C
1610 C
1620 C
1630 C
1640 C
1650 C
1660 C
1670 C
1680 C
1690 C
1700 C
1710 C
1720 C
1730 C
1740 C
1750 C
1760 C
1770 C
1780 C
1790 C
1800 C
1810 C
1820 C
1830 C
1840 C
1850 C
1860 C
1870 C
1880 C
1890 C
1900 C
1910 C
1920 C
1930 C
1940 C
1950 C
1960 C
1970 C
1980 C
1990 C
2000 C
2010 C
2020 C
2030 C
2040 C
2050 C
2060 C
2070 C
2080 C
2090 C
2100 C
2110 C
2120 C
2130 C
2140 C
2150 C
2160 C
2170 C
2180 C
2190 C
2200 C
2210 C
2220 C
2230 C
2240 C
2250 C
2260 C
2270 C
2280 C
2290 C
2300 C
2310 C
2320 C
2330 C
2340 C
2350 C
2360 C
2370 C
2380 C
2390 C
2400 C
2410 C
2420 C
2430 C
2440 C
2450 C
2460 C
2470 C
2480 C
2490 C
2500 C
2510 C
2520 C
2530 C
2540 C
2550 C
2560 C
2570 C
2580 C
2590 C
2600 C
2610 C
2620 C
2630 C
2640 C
2650 C
2660 C
2670 C
2680 C
2690 C
2700 C
2710 C
2720 C
2730 C
2740 C
2750 C
2760 C
2770 C
2780 C
2790 C
2800 C
2810 C
2820 C
2830 C
2840 C
2850 C
2860 C
2870 C
2880 C
2890 C
2900 C
2910 C
2920 C
2930 C
2940 C
2950 C
2960 C
2970 C
2980 C
2990 C
3000 C
3010 C
3020 C
3030 C
3040 C
3050 C
3060 C
3070 C
3080 C
3090 C
3100 C
3110 C
3120 C
3130 C
3140 C
3150 C
3160 C
3170 C
3180 C
3190 C
3200 C
3210 C
3220 C
3230 C
3240 C
3250 C
3260 C
3270 C
3280 C
3290 C
3300 C
3310 C
3320 C
3330 C
3340 C
3350 C
3360 C
3370 C
3380 C
3390 C
3400 C
3410 C
3420 C
3430 C
3440 C
3450 C
3460 C
3470 C
3480 C
3490 C
3500 C
3510 C
3520 C
3530 C
3540 C
3550 C
3560 C
3570 C
3580 C
3590 C
3600 C
3610 C
3620 C
3630 C
3640 C
3650 C
3660 C
3670 C
3680 C
3690 C
3700 C
3710 C
3720 C
3730 C
3740 C
3750 C
3760 C
3770 C
3780 C
3790 C
3800 C
3810 C
3820 C
3830 C
3840 C
3850 C
3860 C
3870 C
3880 C
3890 C
3900 C
3910 C
3920 C
3930 C
3940 C
3950 C
3960 C
3970 C
3980 C
3990 C
4000 C
4010 C
4020 C
4030 C
4040 C
4050 C
4060 C
4070 C
4080 C
4090 C
4100 C
4110 C
4120 C
4130 C
4140 C
4150 C
4160 C
4170 C
4180 C
4190 C
4200 C
4210 C
4220 C
4230 C
4240 C
4250 C
4260 C
4270 C
4280 C
4290 C
4300 C
4310 C
4320 C
4330 C
4340 C
4350 C
4360 C
4370 C
4380 C
4390 C
4400 C
4410 C
4420 C
4430 C
4440 C
4450 C
4460 C
4470 C
4480 C
4490 C
4500 C
4510 C
4520 C
4530 C
4540 C
4550 C
4560 C
4570 C
4580 C
4590 C
4600 C
4610 C
4620 C
4630 C
4640 C
4650 C
4660 C
4670 C
4680 C
4690 C
4700 C
4710 C
4720 C
4730 C
4740 C
4750 C
4760 C
4770 C
4780 C
4790 C
4800 C
4810 C
4820 C
4830 C
4840 C
4850 C
4860 C
4870 C
4880 C
4890 C
4900 C
4910 C
4920 C
4930 C
4940 C
4950 C
4960 C
4970 C
4980 C
4990 C
5000 C
5010 C
5020 C
5030 C
5040 C
5050 C
5060 C
5070 C
5080 C
5090 C
5100 C
5110 C
5120 C
5130 C
5140 C
5150 C
5160 C
5170 C
5180 C
5190 C
5200 C
5210 C
5220 C
5230 C
5240 C
5250 C
5260 C
5270 C
5280 C
5290 C
5300 C
5310 C
5320 C
5330 C
5340 C
5350 C
5360 C
5370 C
5380 C
5390 C
5400 C
5410 C
5420 C
5430 C
5440 C
5450 C
5460 C
5470 C
5480 C
5490 C
5500 C
5510 C
5520 C
5530 C
5540 C
5550 C
5560 C
5570 C
5580 C
5590 C
5600 C
5610 C
5620 C
5630 C
5640 C
5650 C
5660 C
5670 C
5680 C
5690 C
5700 C
5710 C
5720 C
5730 C
5740 C
5750 C
5760 C
5770 C
5780 C
5790 C
5800 C
5810 C
5820 C
5830 C
5840 C
5850 C
5860 C
5870 C
5880 C
5890 C
5900 C
5910 C
5920 C
5930 C
5940 C
5950 C
5960 C
5970 C
5980 C
5990 C
6000 C
6010 C
6020 C
6030 C
6040 C
6050 C
6060 C
6070 C
6080 C
6090 C
6100 C
6110 C
6120 C
6130 C
6140 C
6150 C
6160 C
6170 C
6180 C
6190 C
6200 C
6210 C
6220 C
6230 C
6240 C
6250 C
6260 C
6270 C
6280 C
6290 C
6300 C
6310 C
6320 C
6330 C
6340 C
6350 C
6360 C
6370 C
6380 C
6390 C
6400 C
6410 C
6420 C
6430 C
6440 C
6450 C
6460 C
6470 C
6480 C
6490 C
6500 C
6510 C
6520 C
6530 C
6540 C
6550 C
6560 C
6570 C
6580 C
6590 C
6600 C
6610 C
6620 C
6630 C
6640 C
6650 C
6660 C
6670 C
6680 C
6690 C
6700 C
6710 C
6720 C
6730 C
6740 C
6750 C
6760 C
6770 C
6780 C
6790 C
6800 C
6810 C
6820 C
6830 C
6840 C
6850 C
6860 C
6870 C
6880 C
6890 C
6900 C
6910 C
6920 C
6930 C
6940 C
6950 C
6960 C
6970 C
6980 C
6990 C
7000 C
7010 C
7020 C
7030 C
7040 C
7050 C
7060 C
7070 C
7080 C
7090 C
7100 C
7110 C
7120 C
7130 C
7140 C
7150 C
7160 C
7170 C
7180 C
7190 C
7200 C
7210 C
7220 C
7230 C
7240 C
7250 C
7260 C
7270 C
7280 C
7290 C
7300 C
7310 C
7320 C
7330 C
7340 C
7350 C
7360 C
7370 C
7380 C
7390 C
7400 C
7410 C
7420 C
7430 C
7440 C
7450 C
7460 C
7470 C
7480 C
7490 C
7500 C
7510 C
7520 C
7530 C
7540 C
7550 C
7560 C
7570 C
7580 C
7590 C
7600 C
7610 C
7620 C
7630 C
7640 C
7650 C
7660 C
7670 C
7680 C
7690 C
7700 C
7710 C
7720 C
7730 C
7740 C
7750 C
7760 C
7770 C
7780 C
7790 C
7800 C
7810 C
7820 C
7830 C
7840 C
7850 C
7860 C
7870 C
7880 C
7890 C
7900 C
7910 C
7920 C
7930 C
7940 C
7950 C
7960 C
7970 C
7980 C
7990 C
8000 C
8010 C
8020 C
8030 C
8040 C
8050 C
8060 C
8070 C
8080 C
8090 C
8100 C
8110 C
8120 C
8130 C
8140 C
8150 C
8160 C
8170 C
8180 C
8190 C
8200 C
8210 C
8220 C
8230 C
8240 C
8250 C
8260 C
8270 C
8280 C
8290 C
8300 C
8310 C
8320 C
8330 C
8340 C
8350 C
8360 C
8370 C
8380 C
8390 C
8400 C
8410 C
8420 C
8430 C
8440 C
8450 C
8460 C
8470 C
8480 C
8490 C
8500 C
8510 C
8520 C
8530 C
8540 C
8550 C
8560 C
8570 C
8580 C
8590 C
8600 C
8610 C
8620 C
8630 C
8640 C
8650 C
8660 C
8670 C
8680 C
8690 C
8700 C
8710 C
8720 C
8730 C
8740 C
8750 C
8760 C
8770 C
8780 C
8790 C
8800 C
8810 C
8820 C
8830 C
8840 C
8850 C
8860 C
8870 C
8880 C
8890 C
8900 C
8910 C
8920 C
8930 C
8940 C
8950 C
8960 C
8970 C
8980 C
8990 C
9000 C
9010 C
9020 C
9030 C
9040 C
9050 C
9060 C
9070 C
9080 C
9090 C
9100 C
9110 C
9120 C
9130 C
9140 C
9150 C
9160 C
9170 C
9180 C
9190 C
9200 C
9210 C
9220 C
9230 C
9240 C
9250 C
9260 C
9270 C
9280 C
9290 C
9300 C
9310 C
9320 C
9330 C
9340 C
9350 C
9360 C
9370 C
9380 C
9390 C
9400 C
9410 C
9420 C
9430 C
9440 C
9450 C
9460 C
9470 C
9480 C
9490 C
9500 C
9510 C
9520 C
9530 C
9540 C
9550 C
9560 C
9570 C
9580 C
9590 C
9600 C
9610 C
9620 C
9630 C
9640 C
9650 C
9660 C
9670 C
9680 C
9690 C
9700 C
9710 C
9720 C
9730 C
9740 C
9750 C
9760 C
9770 C
9780 C
9790 C
9800 C
9810 C
9820 C
9830 C
9840 C
9850 C
9860 C
9870 C
9880 C
9890 C
9900 C
9910 C
9920 C
9930 C
9940 C
9950 C
9960 C
9970 C
9980 C
9990 C
10000 C

```

Figure 17. Listing of FTTT Read Statements

Column Number →
 1 10 20 30 40 50 60 70 80
:.....:.....:.....:.....:.....:.....:.....:.....:

8 FITIT

51 51 NGX NGY
 1.0 8.5 7.83333 10.333 XLO XHI YLO YHI
 ODD ISYM.

PLOT3D

RZCOEF

NO PUNCH

20 20 NX NY

-.3287712-01 -.2340861-00 .3319865-00 -.5044898-01 .6748608-01 .2943779-00

198 MORE CARDS AT THIS POINT FROM THE ZCOEF DECK.

-.5309451-01 -.1425200-00 -.3379903-00 -.1812959-01 .4699074-00 -.1475317-00

0.8 SIZE

LOG(TIME IN NANOSECONDS)

LOG(GAMMA DOSAGE)

TPHIVX

RADIATION TRANSFER FUNCTION OF THE 741 OP AMP DUE TO FXR EXPOSURE R 577

TRUE BYPASS

0 0 0 IXL IYL IZL

55 20 0 0 OFALSE0.0 0.0 0.9 1.1 0.9

MORE

170 290 0 0 OFALSE0.0 0.0 0.9 1.1 0.9

MORE

190 250 0 0 OFALSE0.0 0.0 0.9 1.1 0.9

NOMORE

Column Number →
 1 10 20 30 40 50 60 70 80
:.....:.....:.....:.....:.....:.....:.....:.....:

Figure 18. Data Card Deck for F I T I T and P L O T 3 D
 Which Contains Total of 220 Cards

with the Calcomp type calls used by the UCC plotter. When completed the results were rather good, in terms of generating good scales and labels for both the three dimensional plots and the plots created by SAP. Additional effort was devoted to incorporating a hidden line removal algorithm in DRAW3D.

The transformation which DRAW3D uses in making the plots is orthographic projection. The scale is constant over the drawing. The transformation equations from the three-dimensional data to the two-dimensional drawing are:

$$\bar{x}(x,y) = x S_x \cos \alpha - y S_y \cos \beta$$

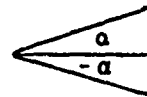
$$\bar{y}(x,y,z) = z S_z - x S_x \sin \alpha - y S_y \sin \beta$$

where S_x , S_y , and S_z are scale factors.

The various views of the surfaces are plotted with values of the angles alpha and beta having the meaning as shown in Figure 19 which is in accord with the previously given definition of the coordinate system and where:

- alpha is the angle measured counter clockwise from the positive horizontal line to the x axis,
- beta is the angle measured clockwise from the negative horizontal line to the y axis.

Predetermining the correct choice of the projection angles α and β was greatly simplified by taking a transparent plastic "picture cube", and marking the x, y, and z axes on it. This was then held up against a 360° protractor of 6 inches diameter in such a manner that the z axis was vertical as viewed; this means the z axis could be tilted to and fro, but not side to side. Then the intersection of the axes along the periphery of the protractor immediately provides the correct choice of alpha and beta to yield the desired perspective. It should be noted that when the z axis is not held vertical during the above operation the resulting plot does not look at all satisfactory, since the orthographic projection of DRAW3D will orient the z axis vertically regardless of α and β .


$$\bar{Y} = Z + X \sin \alpha + Y \sin \beta$$

$$\cos -\alpha = \cos \alpha$$

60

4. COMPUTER FITS TO THE 741 ELECTRICAL AND RADIATION RESPONSE

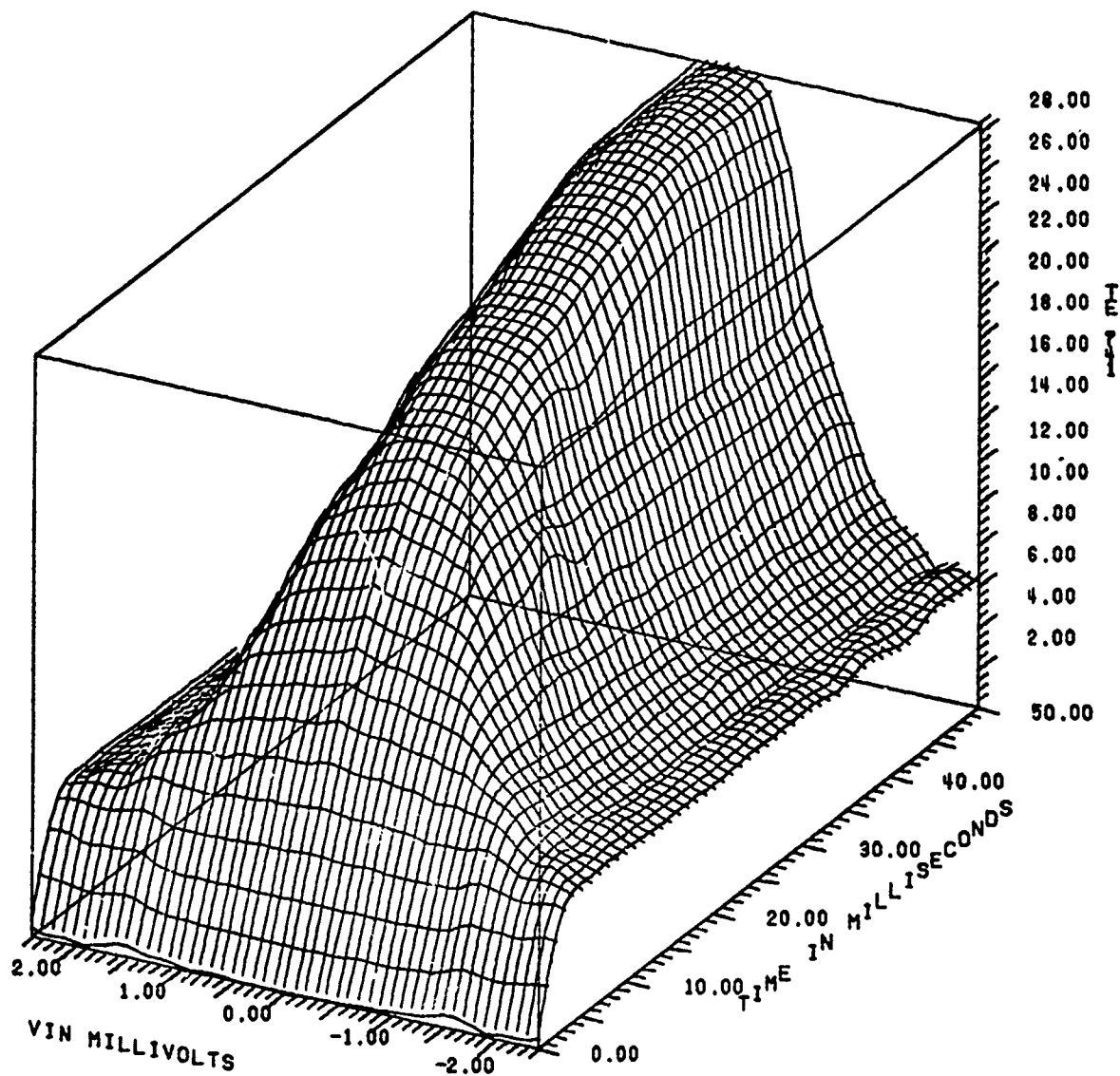
Because the 741 operational amplifiers surfaces were originally created by hand drawn perspective drawings, the subsequent machine processing was relatively simple compared to the strategies applied later to the digital gate data. In Figure 5 is shown the first fit attempted to the step voltage transfer function surface. This plot was made by the UCC routine PROJECT, with scales and labels added later by LEROY lettering. The hand drawing from which the data were obtained was shown earlier as Figure 3. There are some obvious flaws in the computer fit to the data available to it, namely the "ears" at about $e_i = \pm 0.6$ mv and $t = 23$ ms. These protuberances result from the relative sparseness of data points in that portion of the surface, combined with the fact that there is a sharp corner in the data itself. The "roundness" of polynomial fitting causes an over and undershoot in such regions. To a certain extent this can be minimized by supplying an abundance of data which tightens up the tolerances allowed by the least square fitting process of FILLZ.

Another area showing deficiencies is in the central front rising section where the first four or five isochronals have an accordion like pleating. Again this is due to a sparseness of raw data, such that the fitting cubics in a given fitting square are not adequately constrained, and so the spline conditions attempt to improve the fit in those areas where the supply of basic data is greater. The second pass at this surface is shown in Figure 20, as plotted by the routine DRAW3D. Note how adding additional data points eliminated the pleating at the initial rise, and reduced to some extent the overshoot or ears on the upper portion. The top portion at the back is also smoother and flatter.

Three views of the gamma radiation response surface of the 741 generated by FIT3D from the original surface drawing of Figure 14, are shown in Figure 21a, b, and c. It is evident that the overall fitting is rather good. The only place where there is a significant difference occurs on the rising wall at high dosage following the dip to negative saturation. The experimental plot has a shelf which is not reproduced due to the coarseness of the 20 by 20 fitting grid in the machine plots. Two-dimensional cross-sectional plots through maximum and minimum are given for this surface in Figure 22 a and b.

Again note the dramatic difference in the response speed of the device to radiation when compared to electrical stimulus as may be seen by comparing the

VOLTAGE TRANSFER FUNCTION OF 741 OP AMP



ALPHA= 32 ,BETA= 10

Figure 20. Computer Plot of 741 Voltage Transfer Function

RADIATION TRANSFER FUNCTION OF THE 741 OP AMP DUE TO FXR EXPOSURE R#577

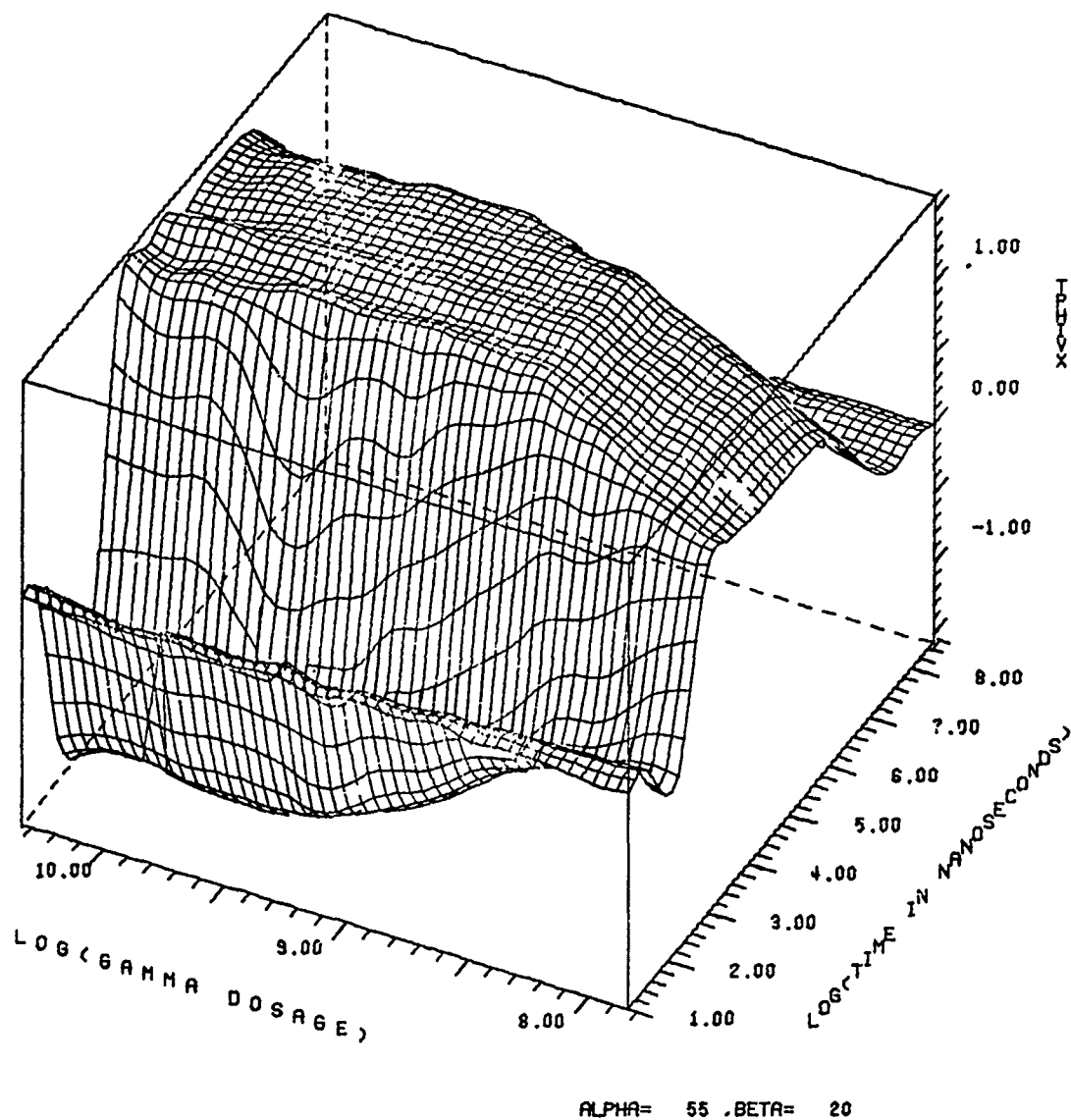


Figure 21. Radiation Transfer Function of 741 due to FXR Exposure

Figure 21a.

RADIATION TRANSFER FUNCTION OF THE 741 OP AMP DUE TO FXR EXPOSURE R#577

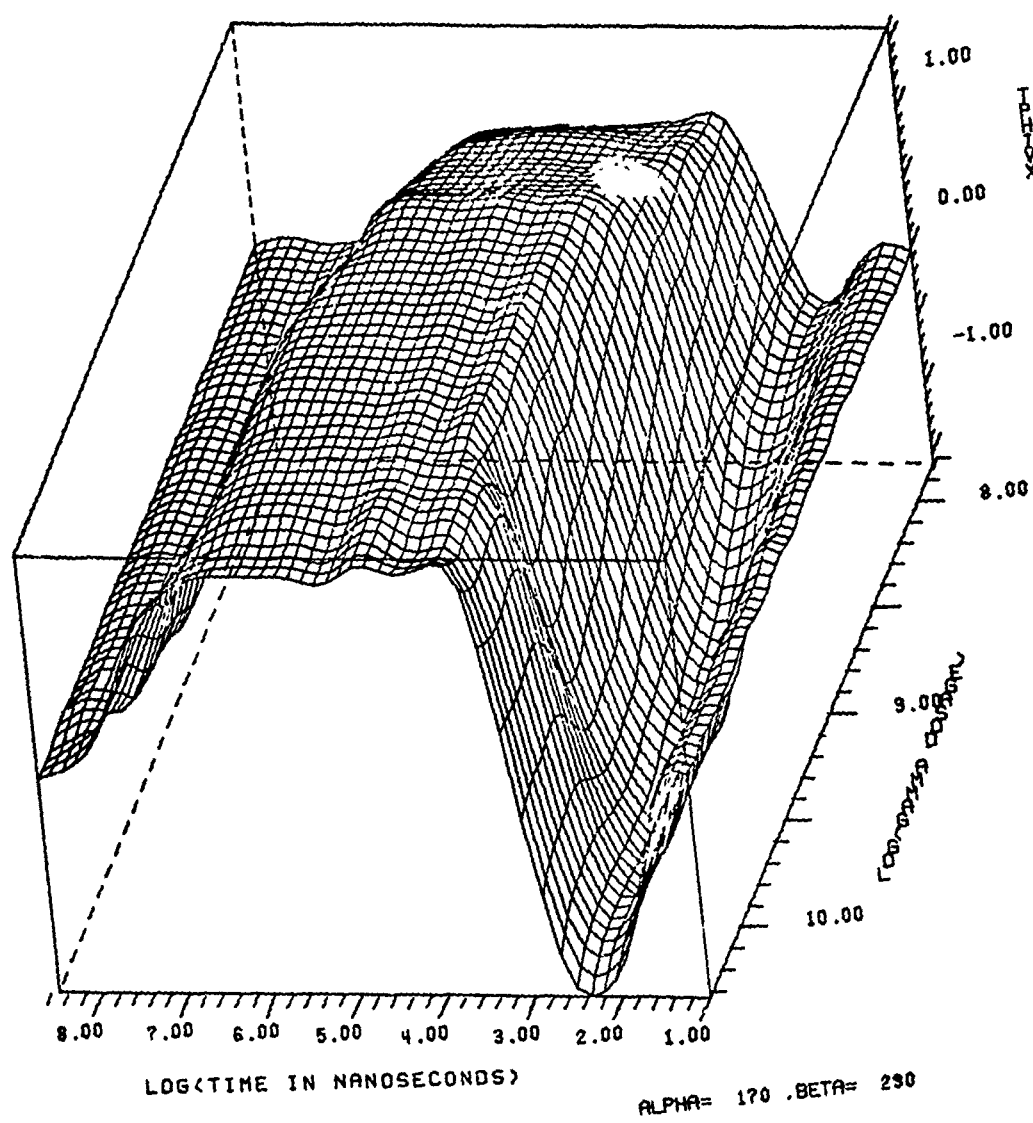


Figure 21b.

RADIATION TRANSFER FUNCTION OF THE 741 OP AMP DUE TO FXR EXPOSURE RM577

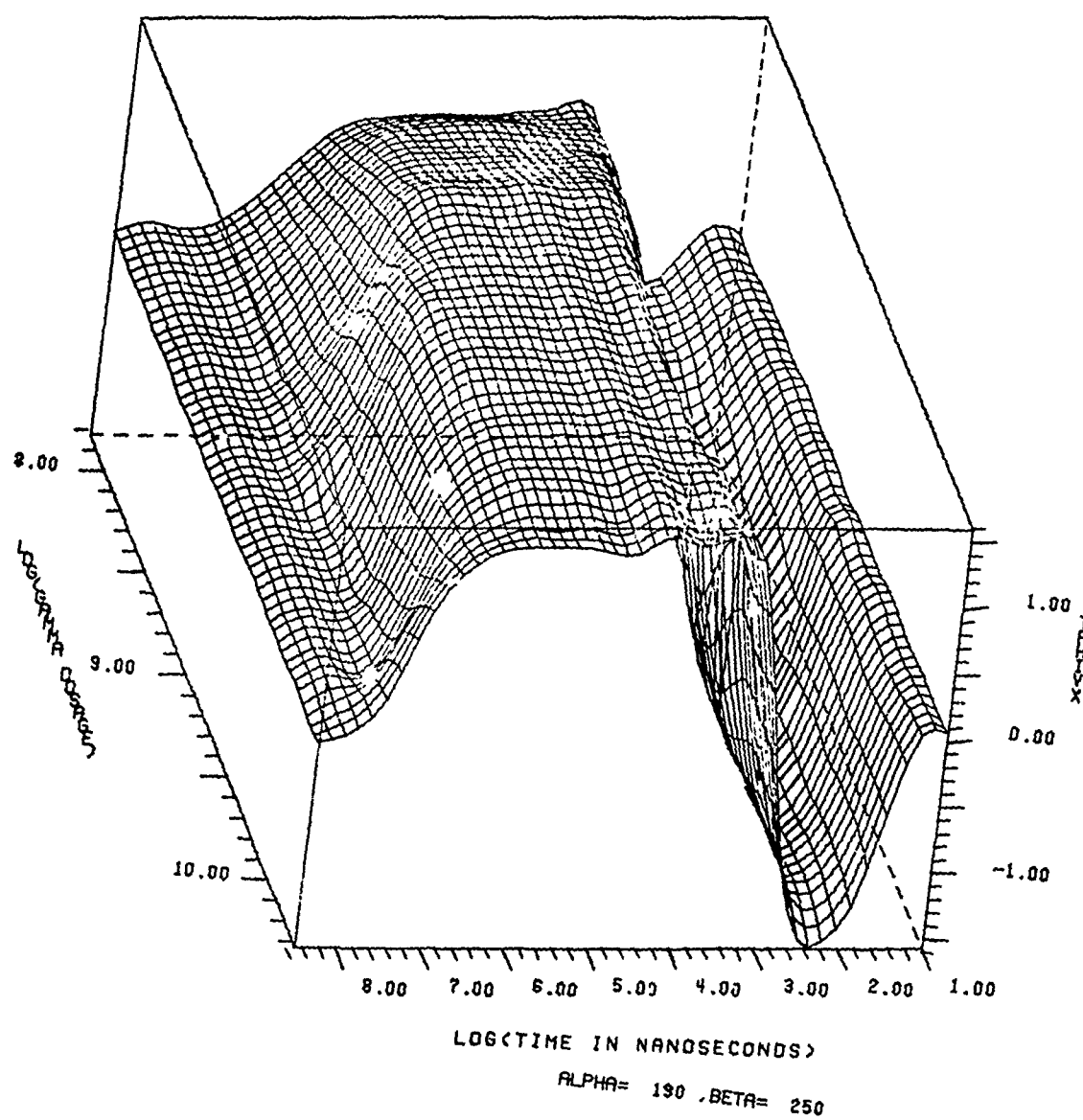


Figure 21c.

STEWART RESEARCH ENTERPRISES

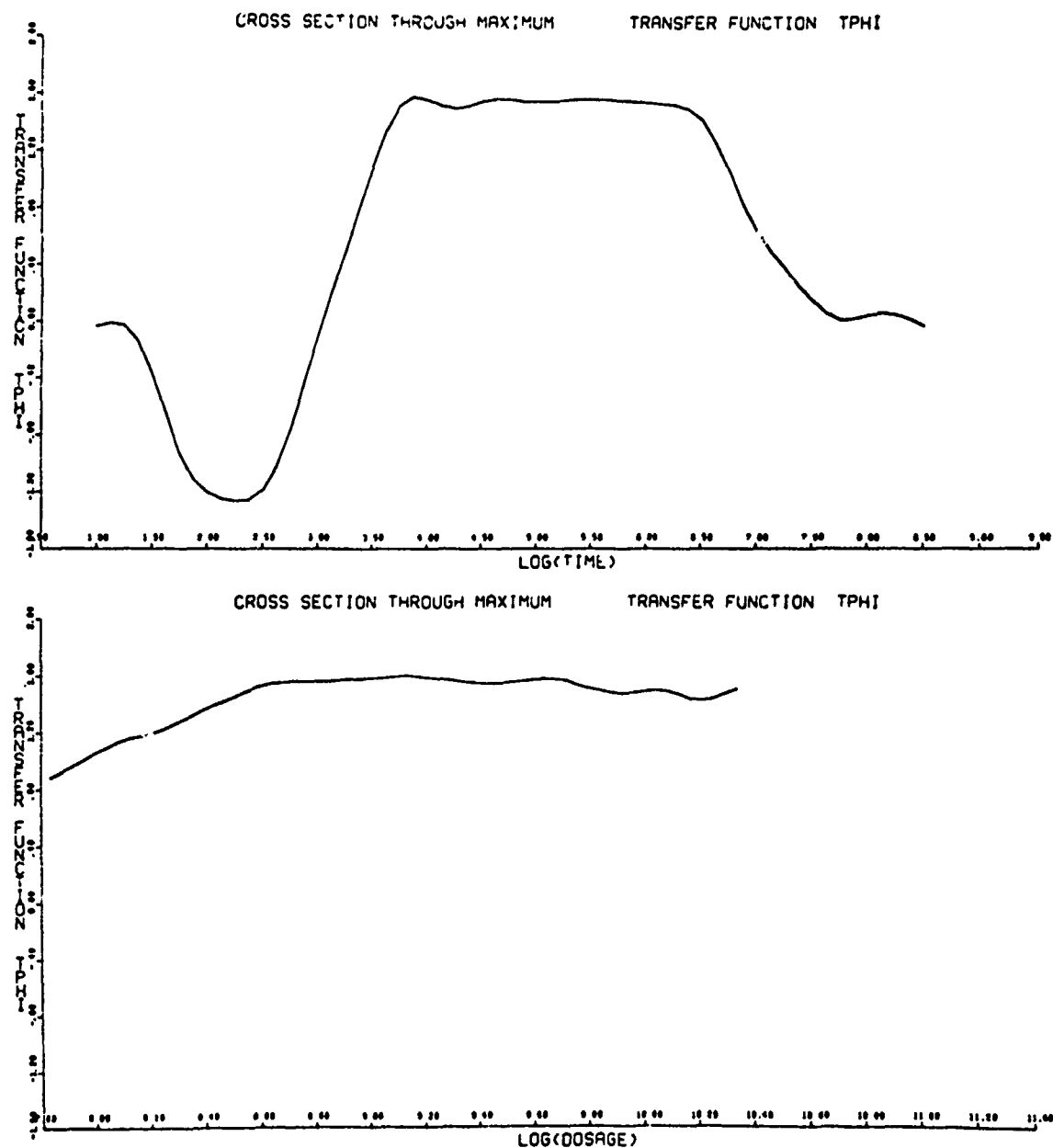


Figure 22a. Cross Section Through Maximum Transfer Function Value of the 741 FXR Surface of Figure 21

STEWART RESEARCH ENTERPRISES

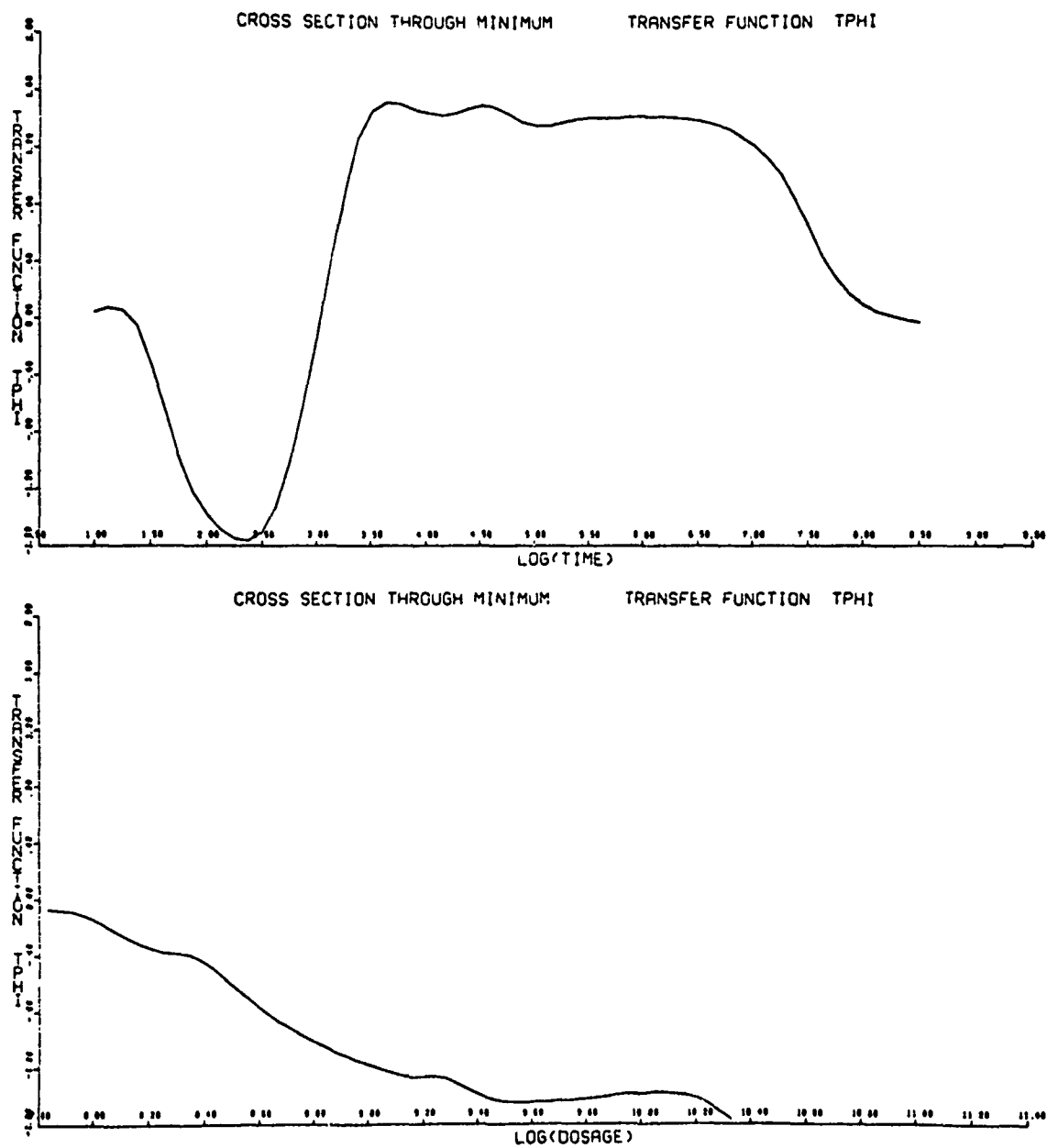


Figure 22b. Cross Section Through Minimum Transfer Function Value of the 741 FXR Surface of Figure 21

time axes of the vacuum cleaner surface, Figure 20, with that of the FXR surface, Figure 21.

The computer representation of the neutron response function surface of the 741 as obtained with the Sandia Pulse Reactor (SPR) is shown in different projection views in Figure 23 a, b, and c. Here we note the great advantage of the computer in being able to rotate the view angle so that we can peer right down the valley, which was so laboriously treated by successive overlays in Figure 15. Two-dimensional cross-sectional views for this neutron surface are given in Figure 24 a and b.

Of particular significance to this work is the cross-sectional plot on the bottom of Figure 24 b, where TPFI versus log (dosage). This curve essentially shows the destruction of the device at very high dose. It seems to represent permanent degradation, in addition to the basic transient phenomenon; few devices survive close shots at the SPR. Thus by defining an appropriate isochronal at a time of approximately 700 microseconds, the permanent damage to a device for a given accumulated dose would appear to be calculable from the same surface that depicts the transient response. This should not be expected to be true in all cases. The way this damage isochronal is used in the circuit analysis code SAP will be discussed later.

The neutron surface shows again the general validity obtainable with the 20 by 20-fitting grid. The results are good in the "big" sense. Note that the fit is not quite a faithful representation because of the absence of a shelf on the rising and falling portions of the valley, as may be seen by comparing Figure 5 with Figure 23.

A last aspect of the radiation transfer function surfaces for the 741 op amp which is of fundamental significance is the logarithmic scale for both the dosage and time axes. This is not only of great significance in terms of the machine fitting process itself, but also will be of high value later in the system analysis operations in SAP in considerably reducing the time constant and execution time problem for certain classes of simulation. However, a logarithmic time axis does not prove acceptable for some of the digital gate response surfaces. That is, the complexity of the response must reduce with time in order for a logarithmic time base to be acceptable. The 741 surfaces show that this does happen for some integrated circuits.

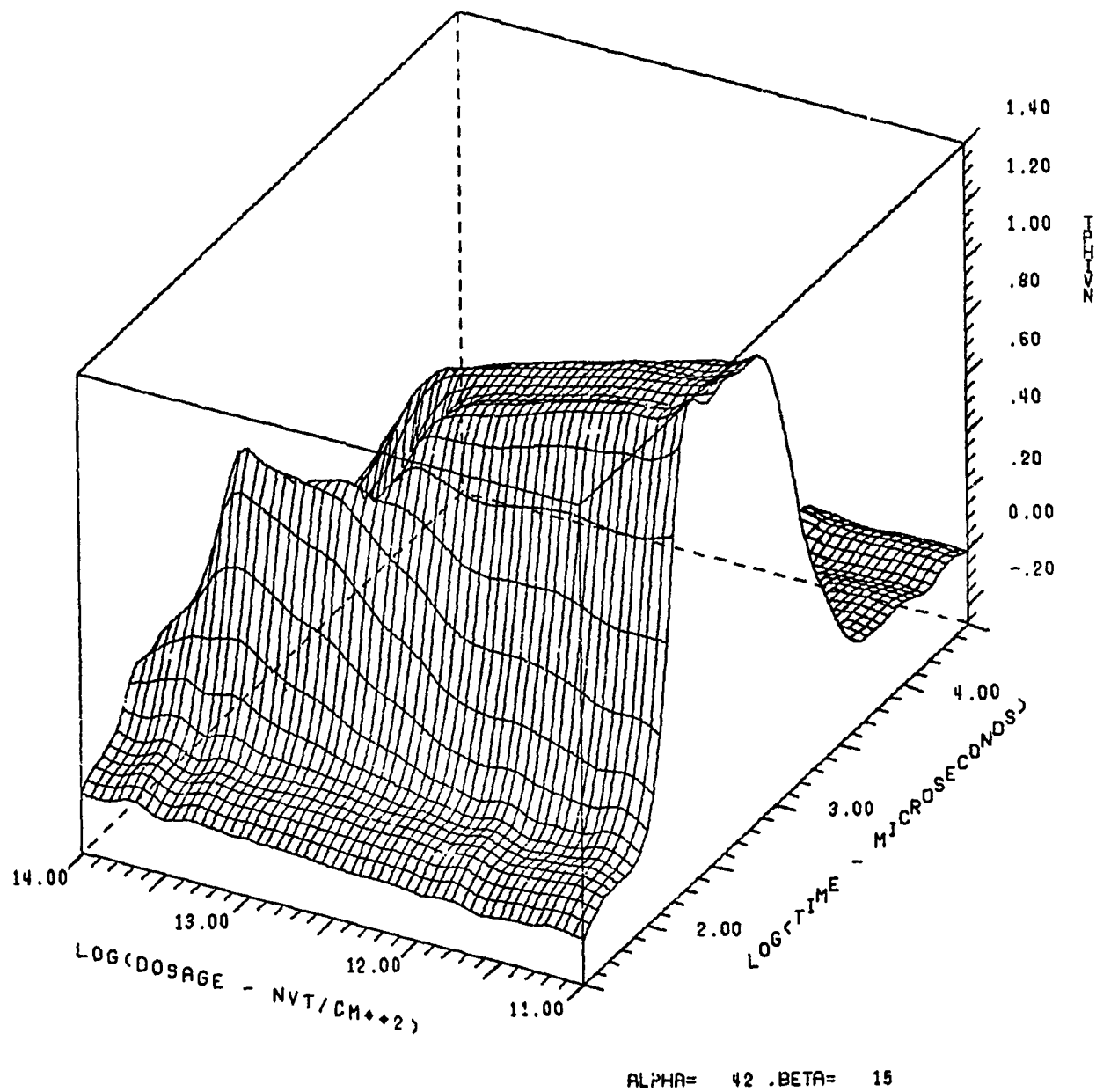


Figure 23a.

Figure 23. Voltage Response Function of 741 due to SPR Neutrons

VOLTAGE RESPONSE FUNCTION OF 741 WRTC SPR NEUTRONS RUN#429

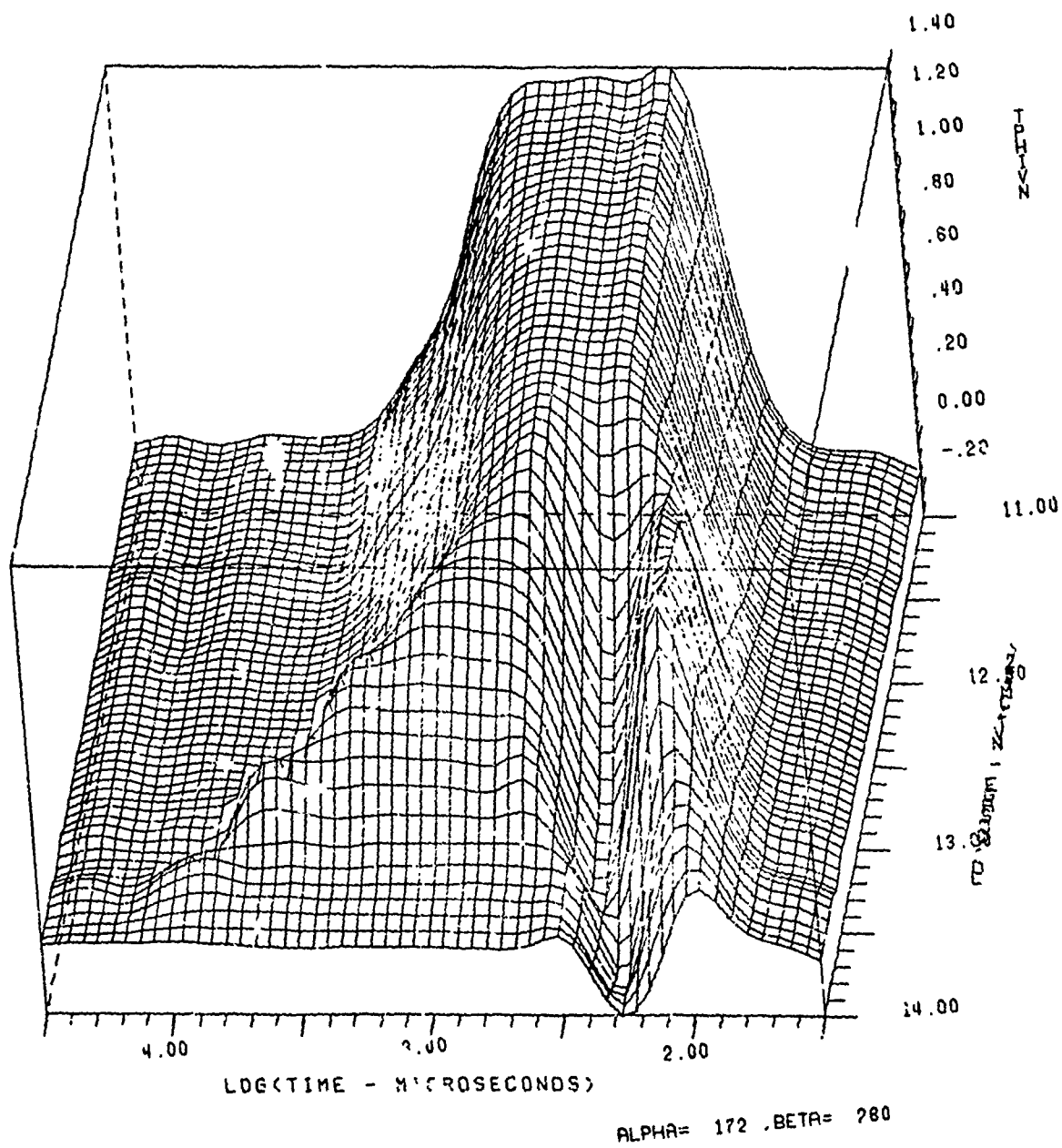


Figure 23b.

VOLTAGE RESPONSE FUNCTION OF 741 WRTO SPR NEUTRONS RUN#429

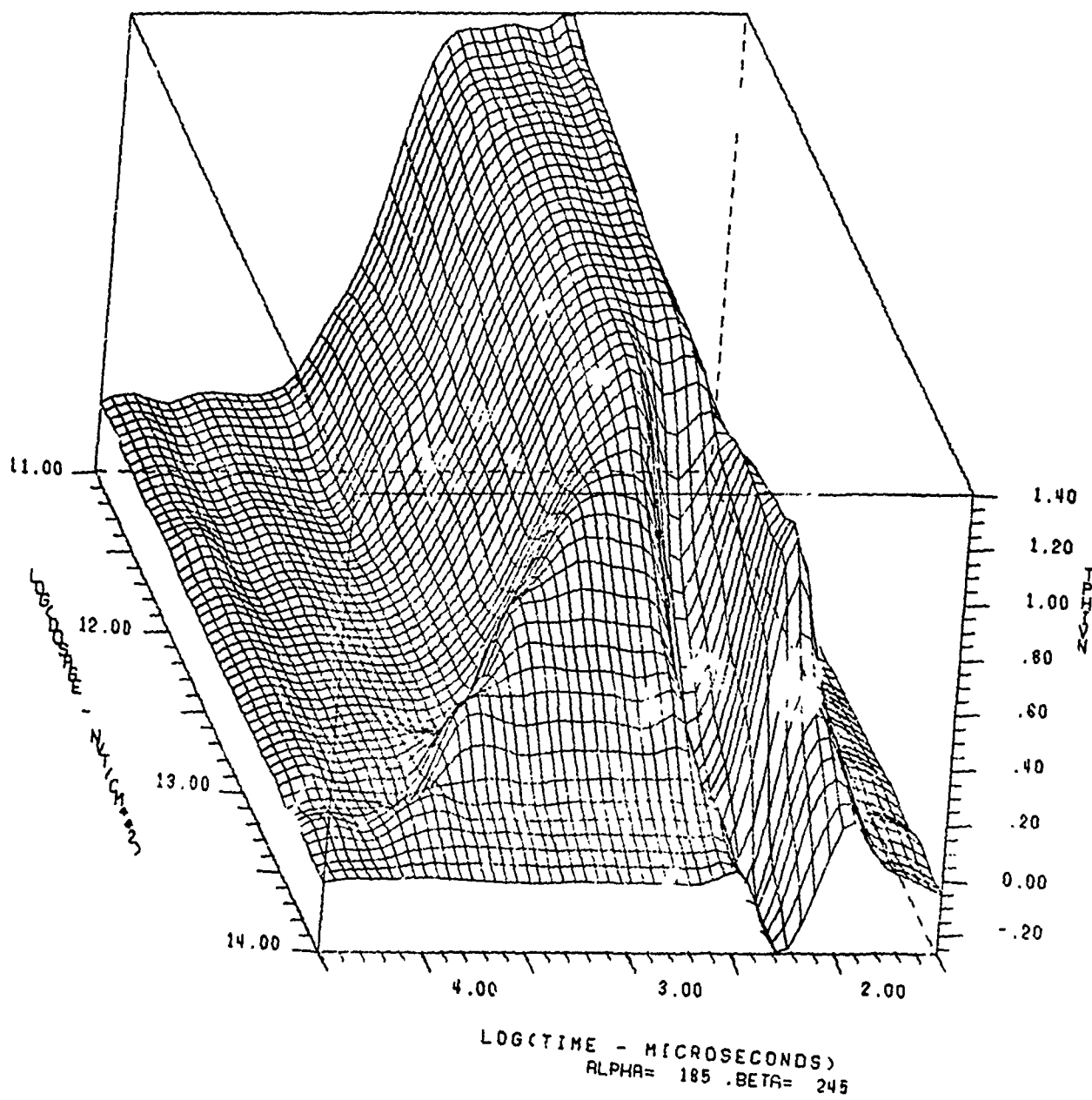


Figure 23c.

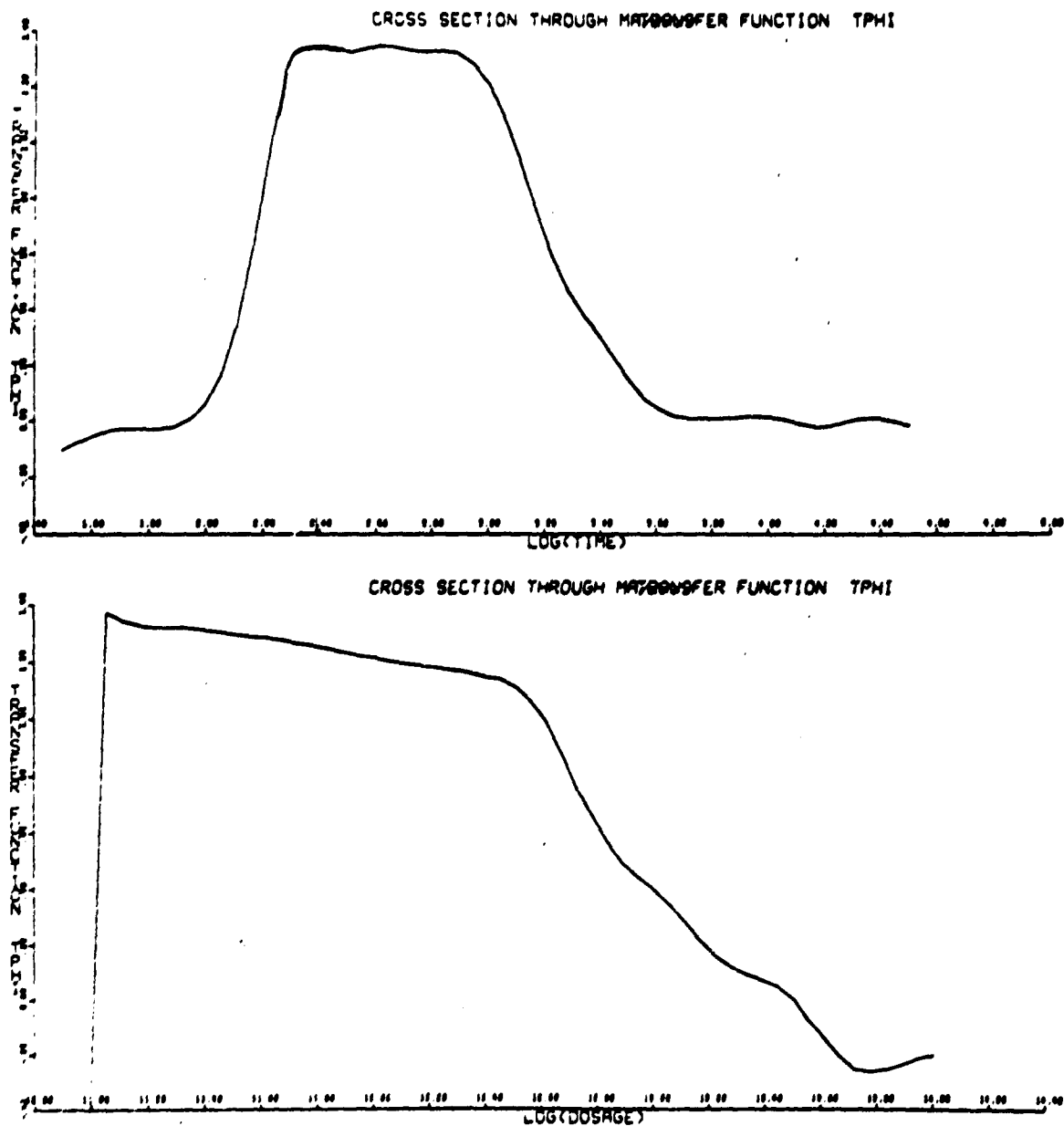


Figure 24a. Cross Section Through Maximum Transfer Function Value of the 741 SPR Surface of Figure 23

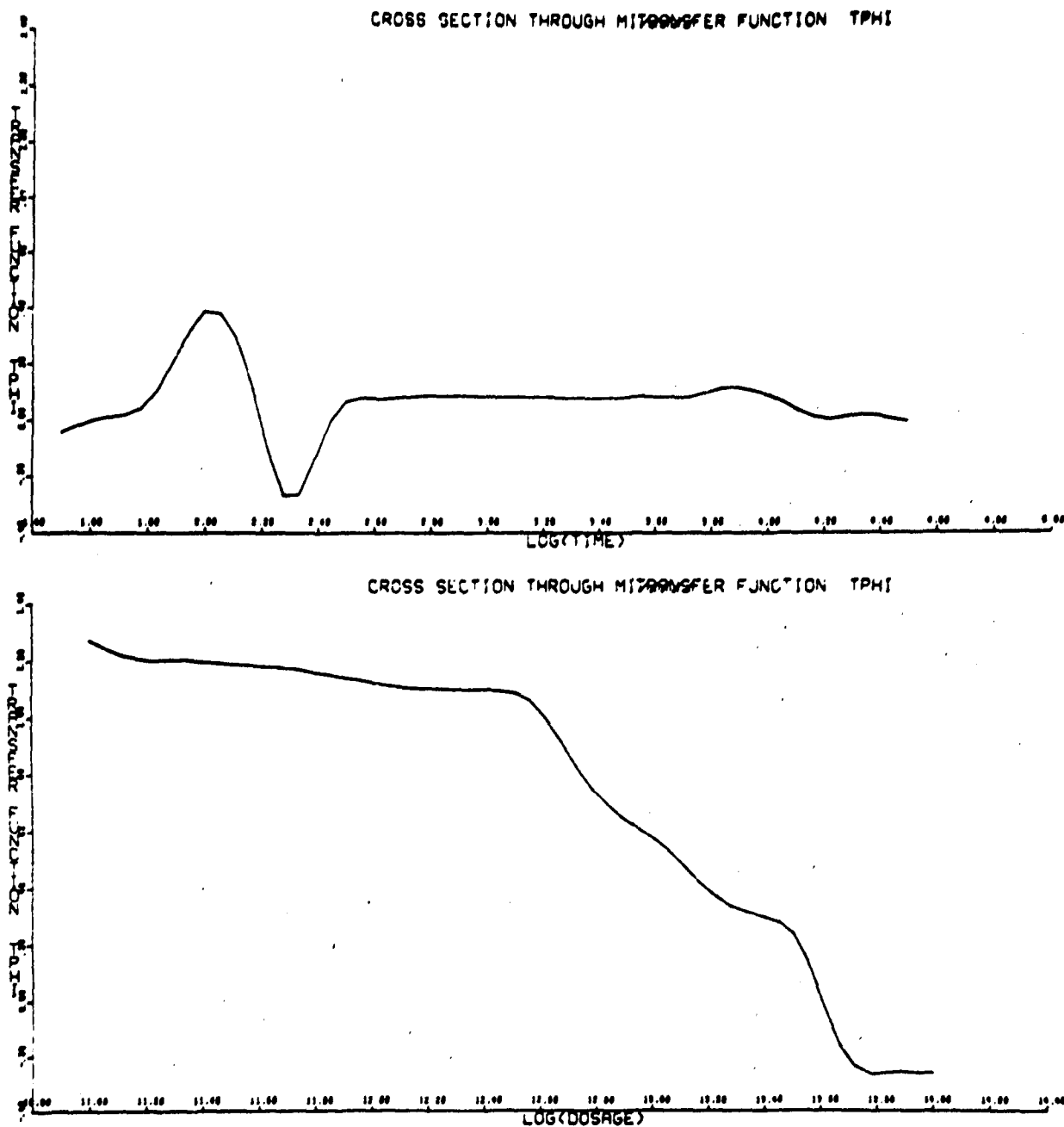


Figure 24b. Cross Section Through Minimum Transfer Function Value of the 741 SPR Surface of Figure 23

5. COMPUTER FITS TO THE 9704 ELECTRICAL AND RADIATION RESPONSE SURFACES

As discussed earlier, no hand drawings were prepared for the 9704 digital gate surfaces. The resolution of conflicting or sparse data available from the experiments was one problem in selecting and developing proper computer based strategies.

a. Electrical response surface

The gate electrical characterization was obtained with the circuit configuration shown in Figure 25, with data coming out as an x-y recorder tracing of the output voltage versus time for input pulses of various amplitudes and a fixed length of 300 ns. This 300 ns pulse length was sufficient for the steady state to be reached, particularly for the long time constant phenomenon that occur in the transition range. Then the output tracings were digitized and the data key punched as the ZDATA deck for FIT3D, with the results that are shown in Figure 26a, b, and c.

Some ringing is evident in the turn-on and turn-off of the device. The shortest possible lead-lengths in the test fixture were unable to eliminate the ringing. The values of circuit capacitance were far smaller than those required to explain the observed ringing frequency, and hence must be due to internal device capacitance. The figure shown made use of the linear interpolation option in FITIT to call INTERP so as to provide a denser packing of data points to control the "rolliness" of the cubics. This was needed in the fast transition regions. A further step necessary was to provide a line of data points along the $t = 0$ axis, to "tie down" the end of the surface. The experimental data supplies a multiple redundancy of points that are all located at the same spot, and hence the fitting polynomials are unconstrained elsewhere.

The essential manner in which the subroutine INTERP operates to provide additional data points to help in packing the data plane fully, is to first label legitimate data strings as .TRUE., and the new strings to be interpolated in between them as .FALSE. Next, a DO loop marches from one end of the y axis to the other interrogating the logical variable TYPE to find out if the string position is .FALSE. in which case it then locates the neighboring .TRUE. strings on either side. Next another DO loop moves along that .FALSE. string in the x direction by the desired amount. Then it finds four bracketing valid data points on the neighboring .TRUE. strings, and first interpolates along their x direction to find the

VOLTAGE RESONSE FUNCTION OF THE 9704 NAND GATE RUN #578

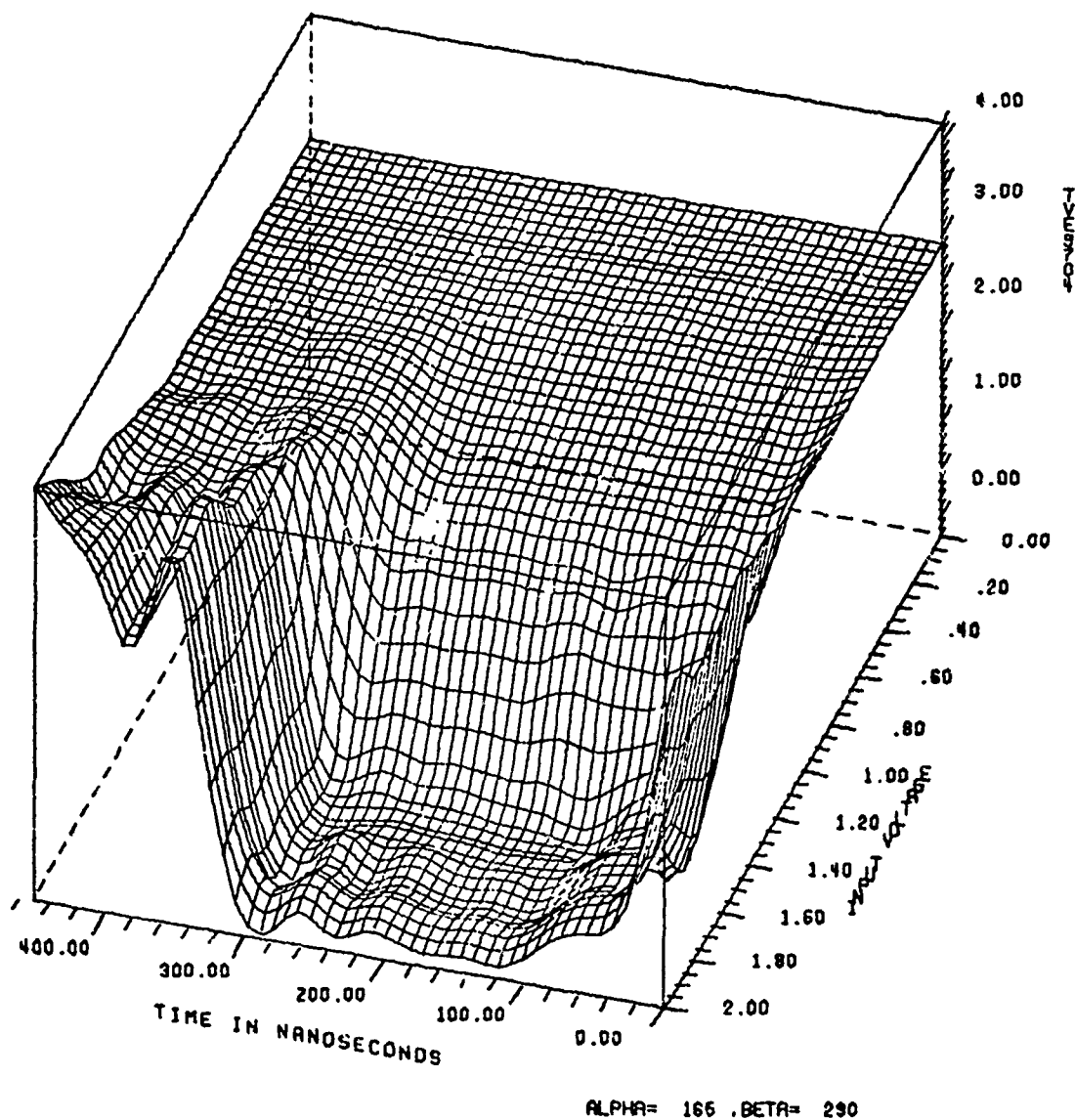


Figure 26. Voltage Response Function Surface of the 9704 NAND Gate
Figure 26a.

VOLTAGE RESONSE FUNCTION OF THE 9704 NAND GATE RUN #578

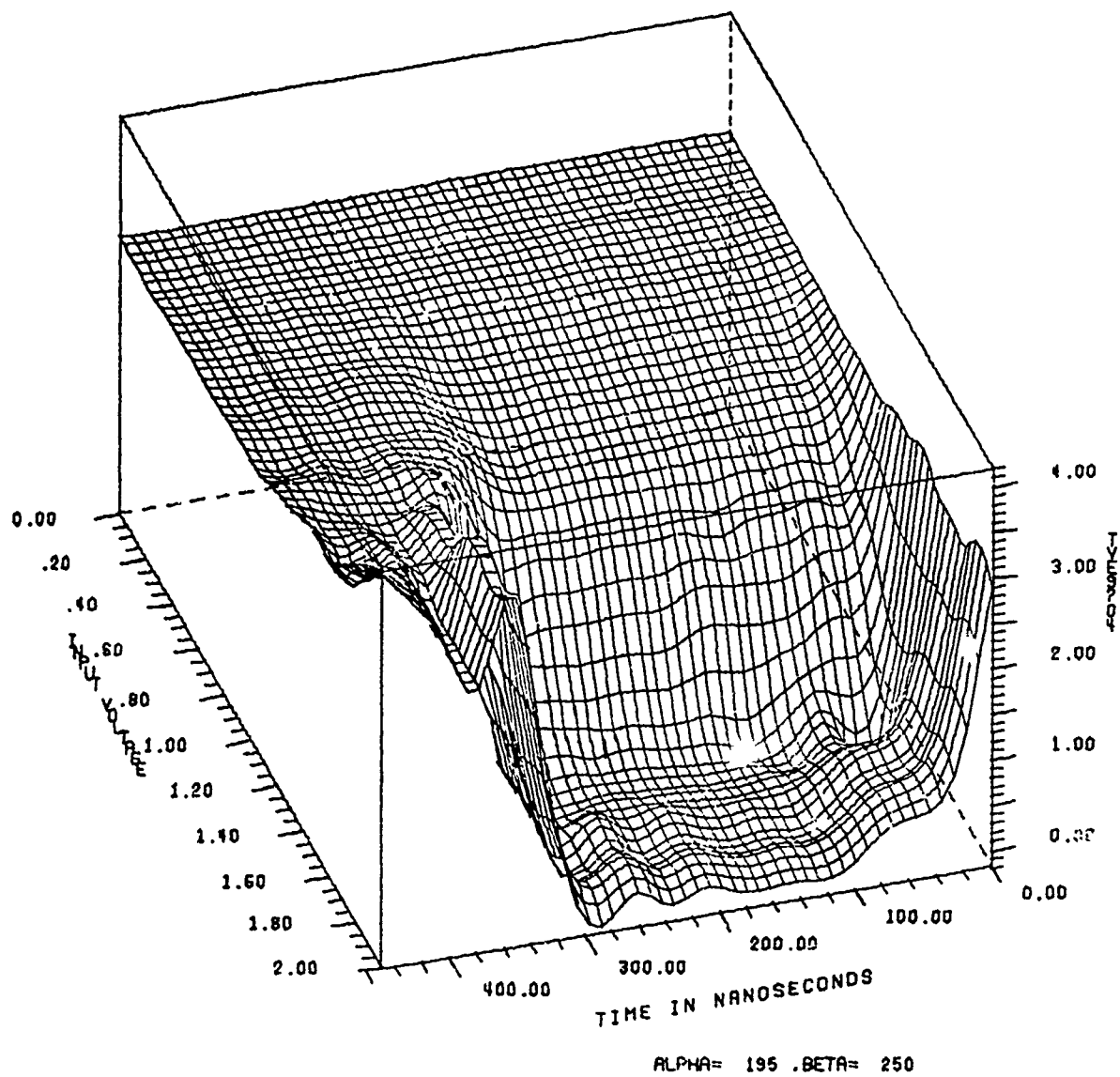


Figure 26b.

VOLTAGE RESPONSE FUNCTION OF THE 9704 NAND GATE RUN #578

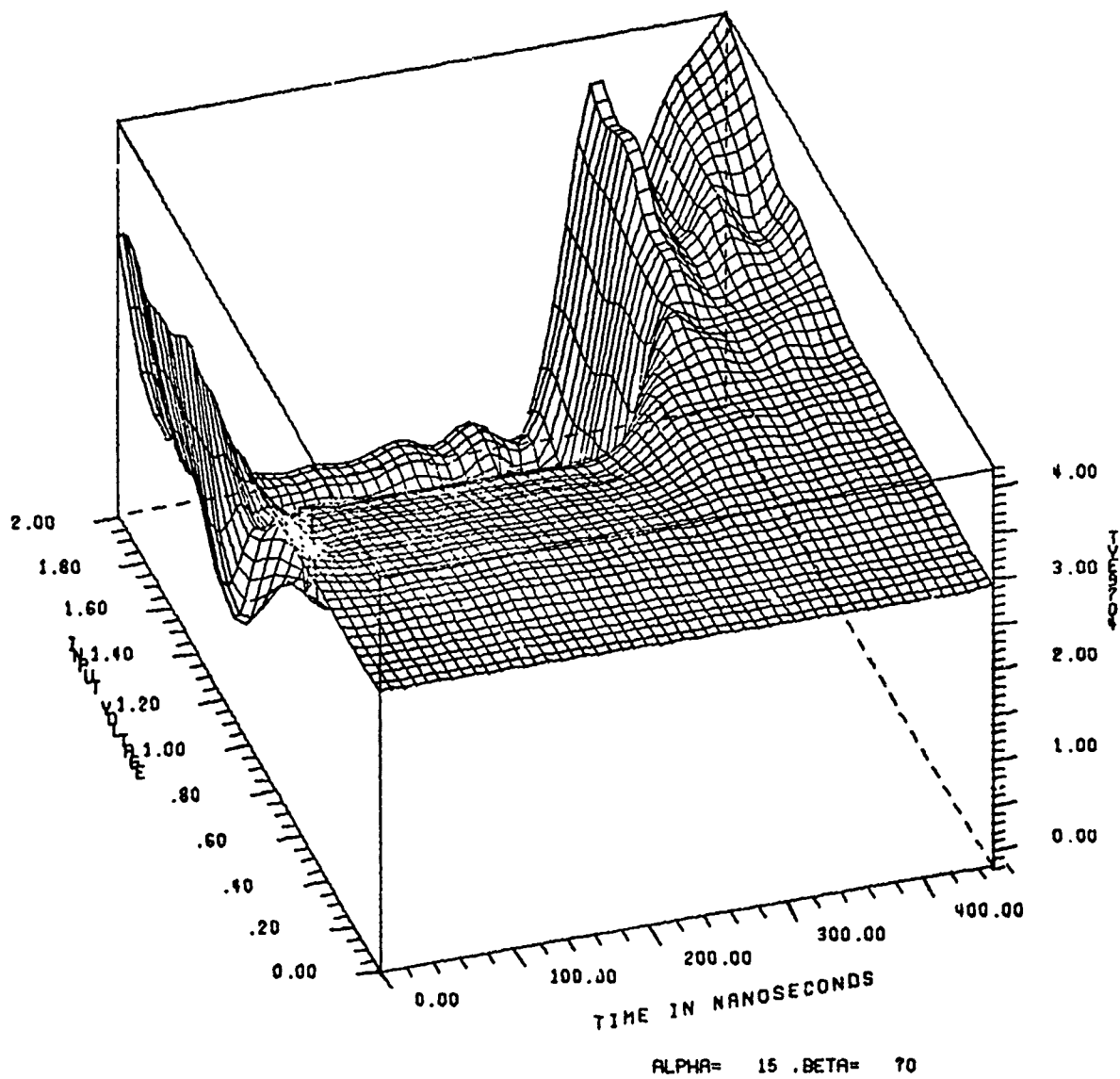
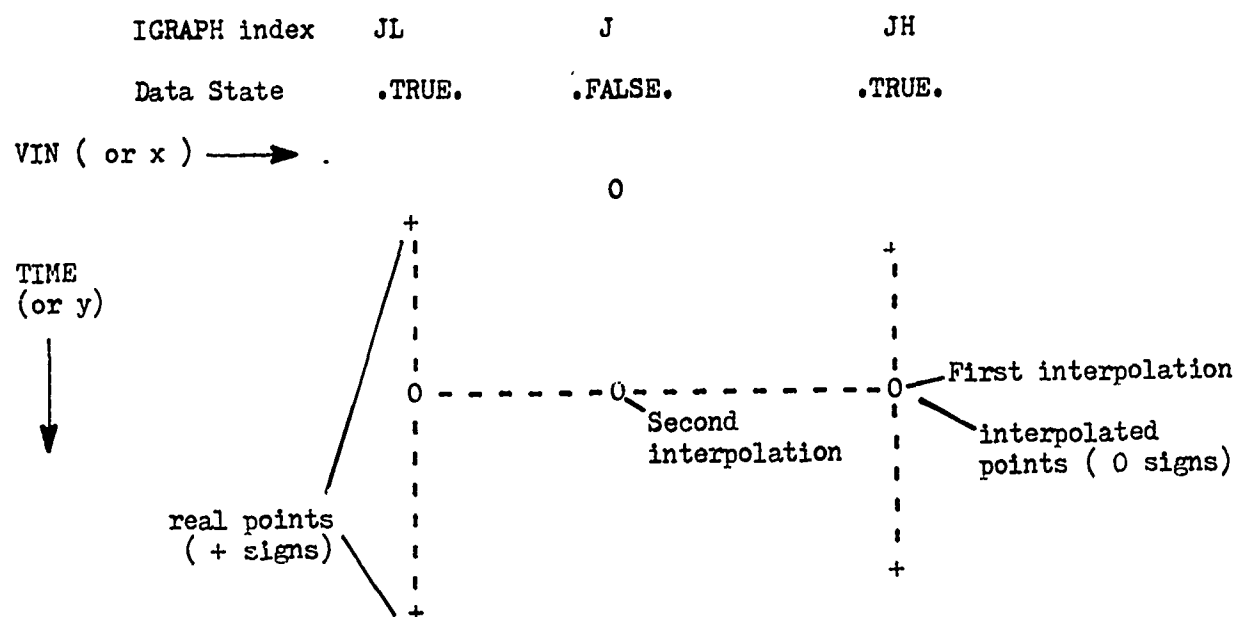


Figure 26c.

z value at the same x coordinate as the new .FALSE. point. This finds two points on either side of the new point. Then linear interpolation is used again between these two points on the .TRUE. strings to find the z value at the new .FALSE. point. The DO then continues on to the next desired .FALSE. point and repeats the process. If the data could be instantaneously processed as it is acquired, the need for this type of processing would be minimized, since the sparseness of data is very evident in the results of the fitting process.



Sketch of Double Interpolation Method Used for Filling Data Plane

b. Neutron response surfaces

The voltage and current response surfaces of the gate presented a new class of problems in the process of deriving a satisfactory result. To fully describe these problems, we shall include herein the full sequence of plots from the first try to the last. We will not delete our mistakes since it is from those that the successful strategies frequently are formed. Both the voltage and current radiation response surfaces will be discussed concurrently rather than separately to show how they presented similar problems.

The initial fit and plot of the radiation neutron current transfer function for the gate is shown in Figure 27, while that of the voltage surface is shown in Figure 28. Very evident in both of them is the large, smooth region at very

CURRENT NEUTRON FUNCTION OF 9704

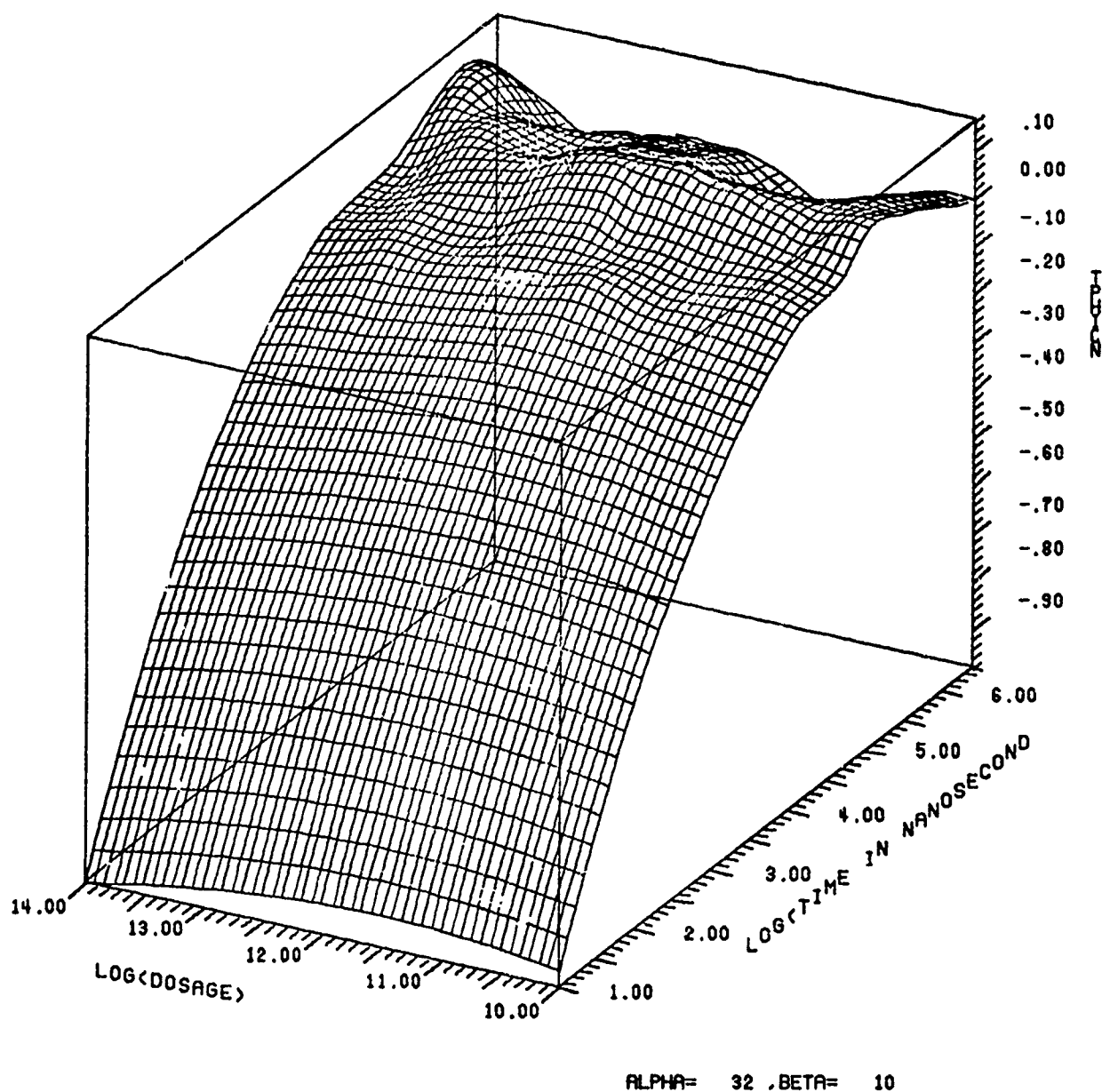


Figure 27. Current Neutron Function of 9704 NAND Gate,
First Fitting of Basic Data by Computer

VOLTAGE NEUTRON RADIATION RESPONSE OF 9704 RUN 333

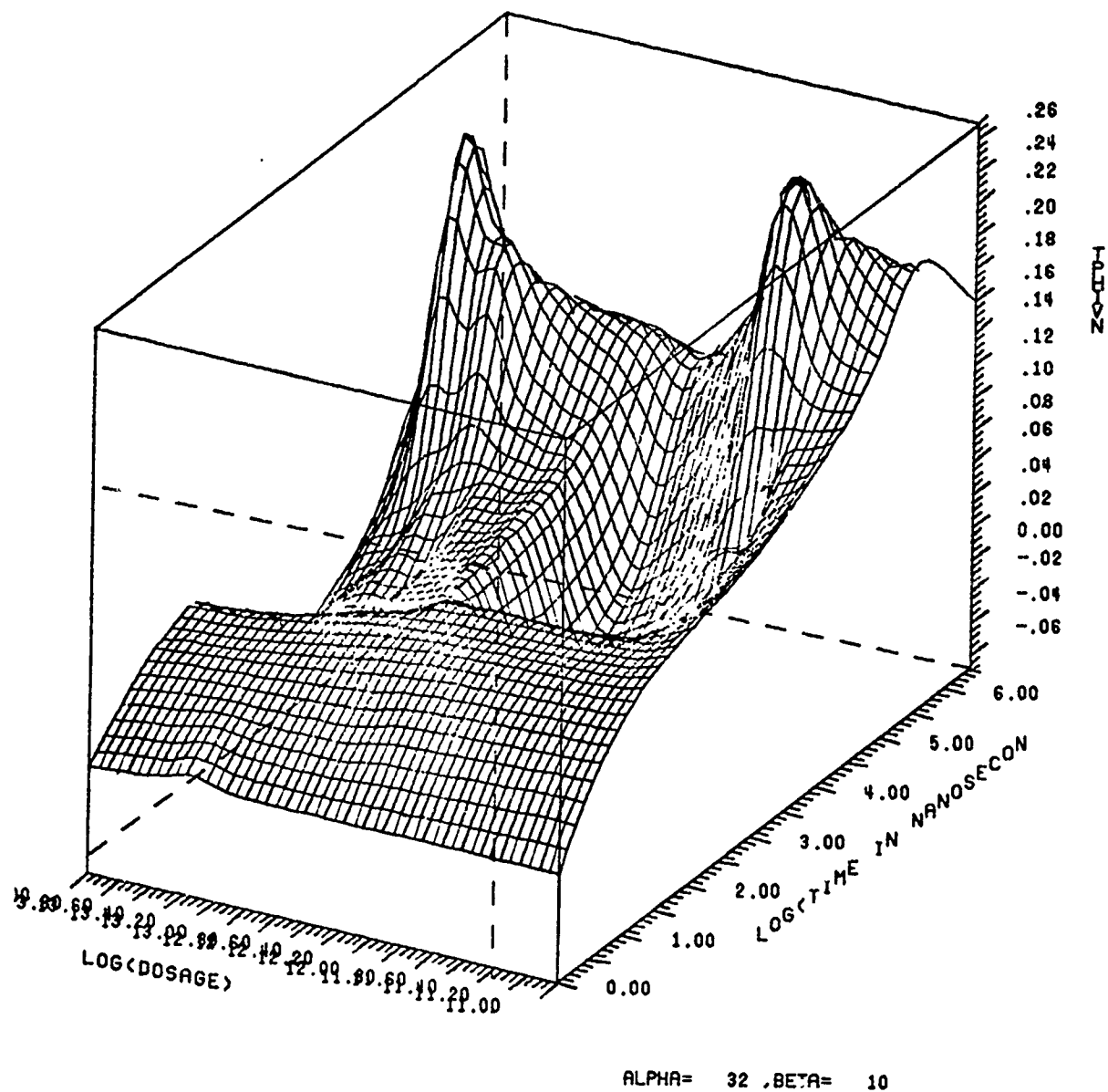


Figure 28. Voltage Neutron Radiation Response of 9704,
First Fitting of Basic Data by Computer

small time values. In both cases this is entirely spurious, and is the result of a mistake in specifying the lower limit for the time axis over which the routine FITIT was to generate the machine fit. The initial value of time in the experimental data (Polaroid photographs of CRT tracings) is the trigger of the reactor core and the oscilloscope sweep. As discussed in the Appendix the build-up of fission processes in the reactor core occurs exponentially in time reaching a maximum the order of a hundred microseconds after trigger. Meaningful excursions from the vertical zero on the scope tracing do not begin until about 15 or 20 ms after trigger. The error in the fit was to use a value of the variable XLOW which was too small, 0.0 or 1.0, on the log time scale, rather than 3.0 (which corresponds to 1 ms). Hence, a very large portion of the basic x-y fitting grid plane had no data points to constrain locally the fitting function, which was then free to minimize residuals in the spline sense in those areas containing data. The result is the long downward sweep in the current surface which is entirely erroneous.

The next try at the current neutron transfer function is shown in the results of run 322 in Figure 29. Despite all the lumps and wiggles, this is not a bad representation of the real data. The one spurious aspect is the negative excursion somewhere, as evidenced by the fact that the surface seems to be sitting in mid-air in the plot cube. So we turned the surface around in the view of run 328 as seen in Figure 30, which shows the backside of the hill. The negative excursion at a dosage of 10^{11} and 10^{13} is due to an absence of constraining data in those areas. By examining the basic data, filling in additional data points by hand where needed, and running through INTERP, the results of run 404 were obtained as shown in Figure 31. Still the surface does not seem really acceptable in that we don't expect experimental functions to be that rough.

The next try at the neutron-voltage transfer function gave the plot of run 342 shown in Figure 32a, where INTERP was used to fill in regions in the right hand data space. The vast difference between this figure and Figure 28, came about by deleting one valid data string from the collection of data. Of seven pictures representing dosages less than 10^{13} , only one showed any response, the rest were zero. Thus, it was necessary to use judgement and arbitrarily delete that one. Here it is evident that there is a threshold type of radiation response for the gate. Except for the one device, there was no effect of the radiation on

CURRENT NEUTRON FUNCTION OF 9704 RUN 322

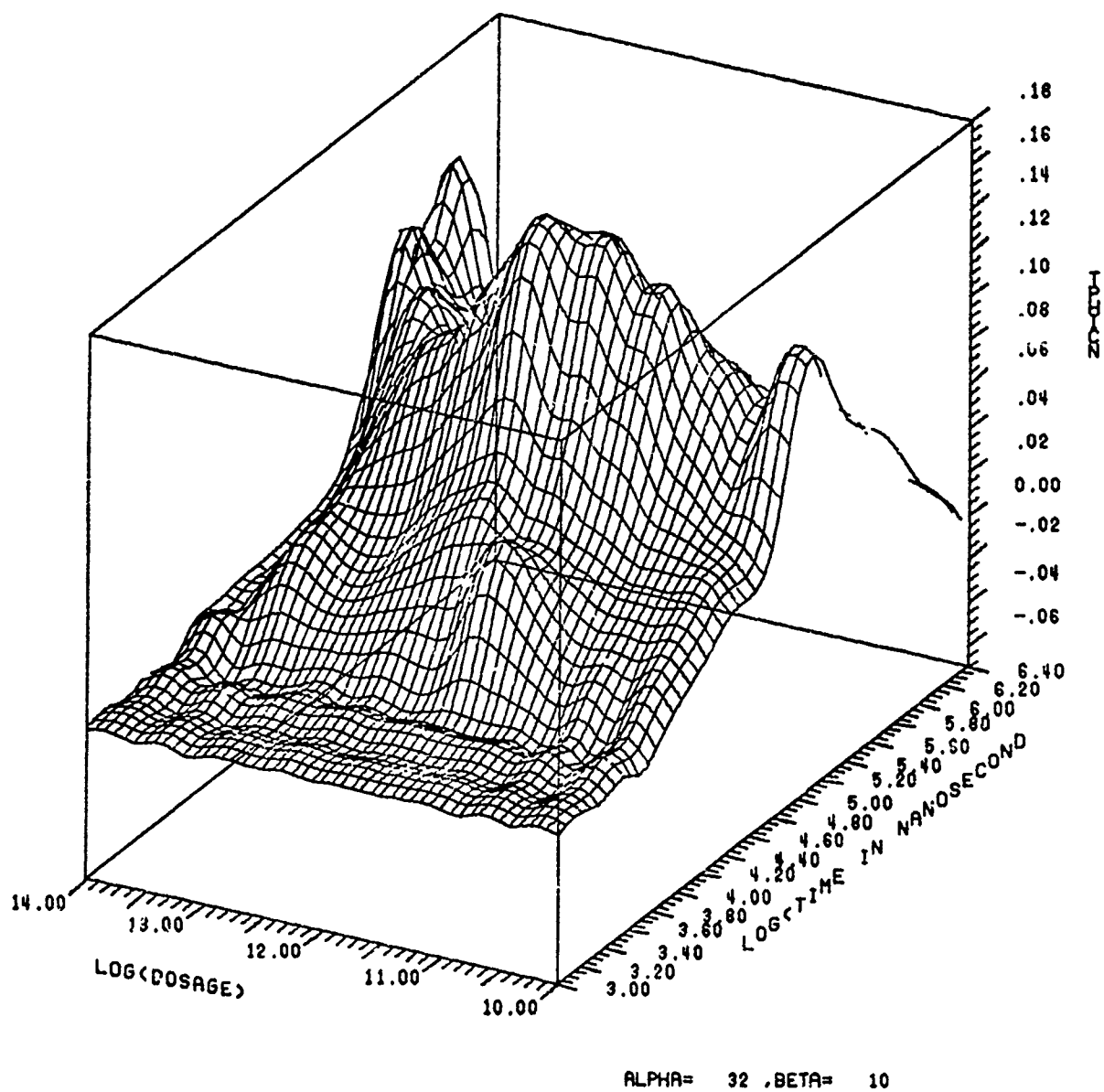


Figure 29. Front View of Current Neutron Function of 9704,
Second Fitting by Computer

CURRENT NEUTRON RADIATION RESPONSE OF 9704 RUN 328

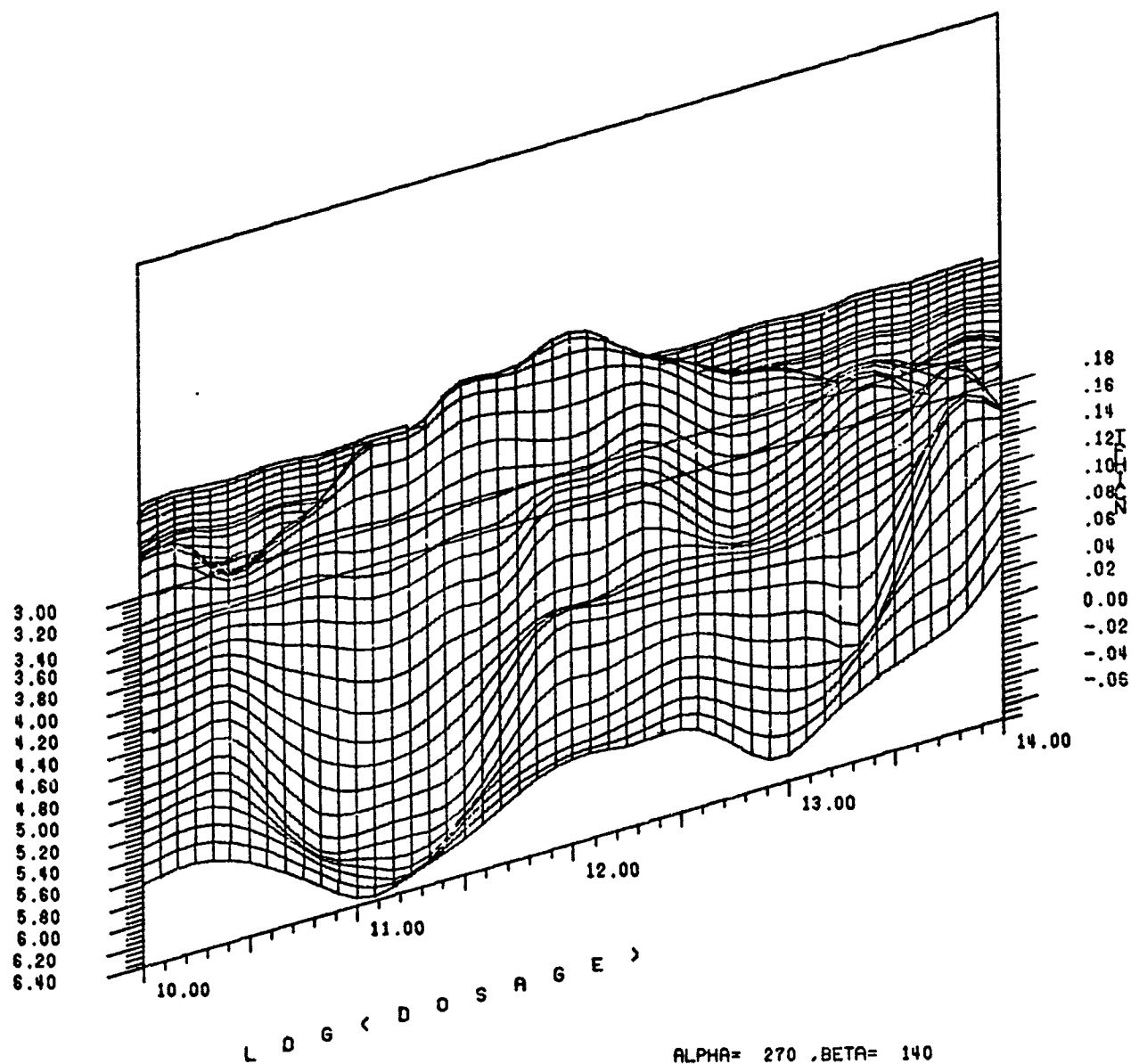


Figure 30. Rear View of Current Neutron Function of 9704, Second Fitting by Computer

CURRENT NEUTRON RADIATION RESPONSE OF 9704 RUN 404

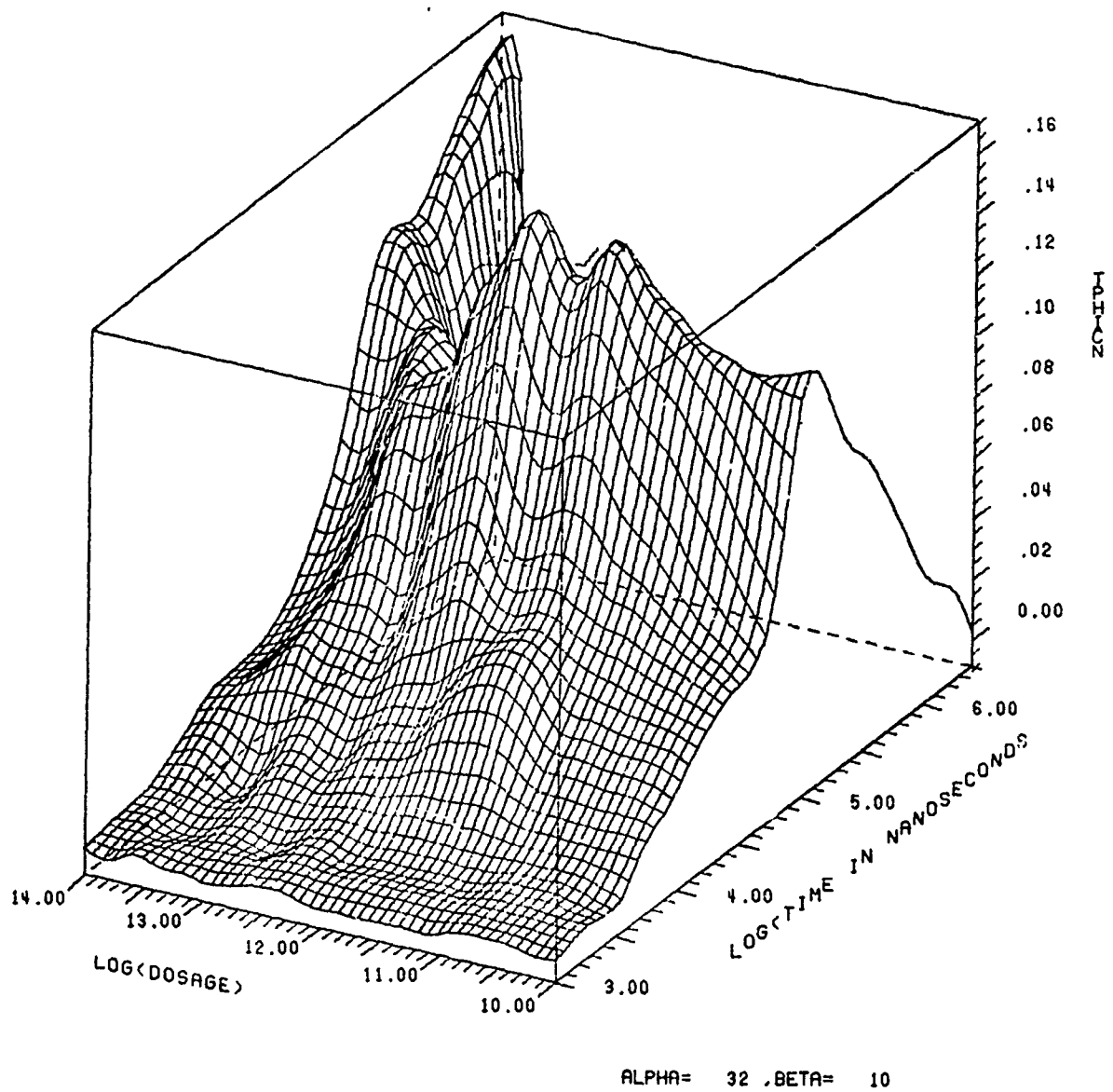


Figure 31. Front View of Current Neutron Function of 9704. Third Fitting by Computer

VOLTAGE NEUTRON RADIATION RESPONSE OF 9704 RUN 342

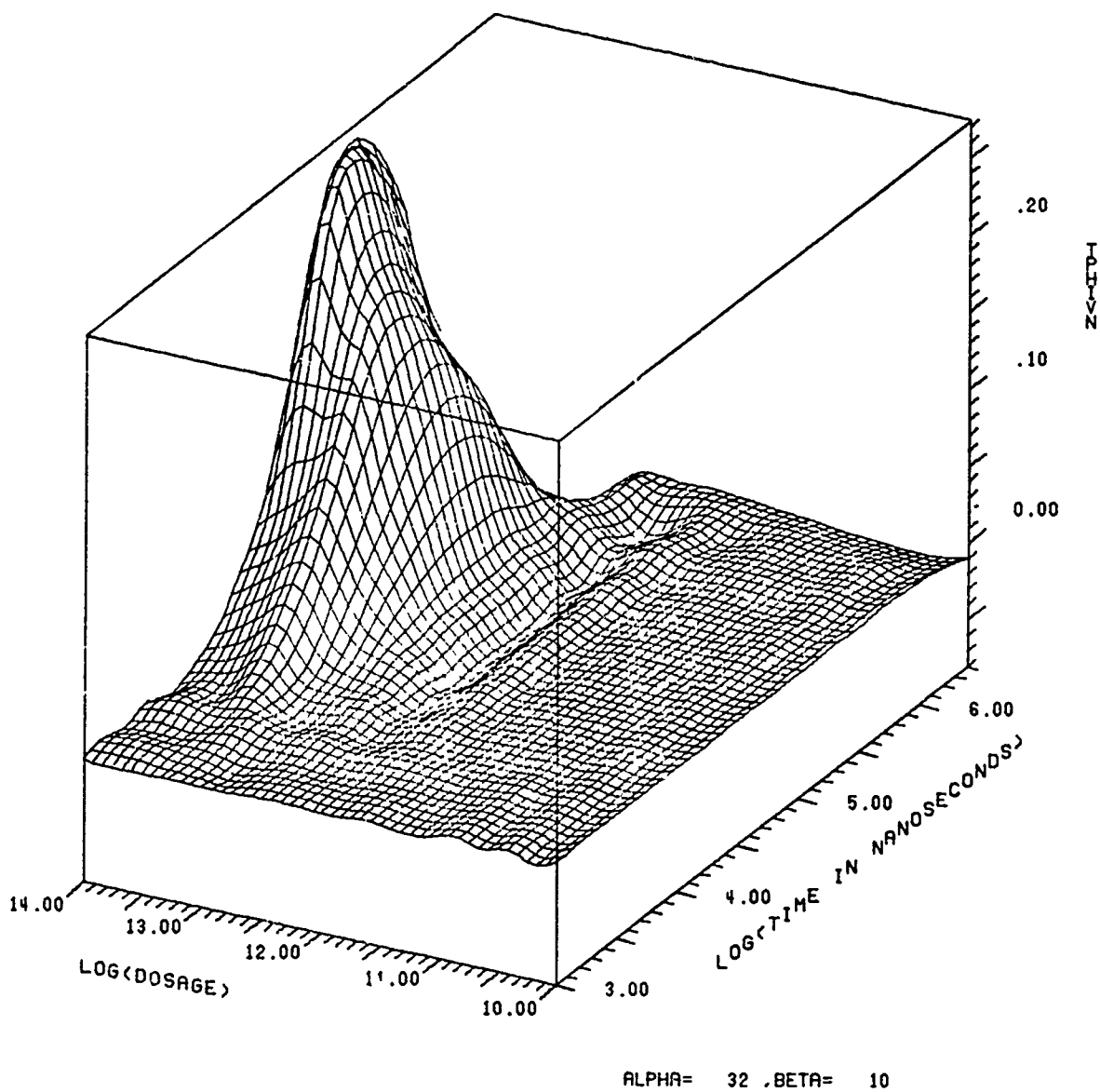


Figure 32a. Front View of Voltage Neutron Radiation Response of 9704. Second Computer Fitting

the output voltage until the dose exceeded about 10^{13} nvt/cm². The fact that the surface is sitting up in the plot cube shows again a negative swing is present somewhere. The other perspective views of Figure 32 b, c, and d, indicate a negative swing at maximum dose and time. This again was due to a sparseness of real data at that location. The final version of the surface after filling the corner with sufficient data values is shown in the results of runs 467 and 471 in Figure 33 a, b, and c. It should be realized that the radiation pulse drives the output voltage up, even though the output may already be in the electrically high digital state. Moreover, the output can swing higher than the positive supply voltage.

Now let us return to the current neutron transfer function surface to see how the smoothing routines were applied to try and even out the local inhomogeneities. The smoothing operations are an optional path in FITIT involving calls to the following subroutines:

NICER - sets up smoothing calls. First smooth on x, stepping along y; then smooth on y, stepping along x. Next, average the z values for the two smoothing directions. The number of passes is NTIME. NICER then calls

SMOOTH - handles setting up wild-pointing, differentiating, smoothing and array interchanges for

DUZ2 - performs actual operations. DUZ2 has options for anchoring of end points.

The starting point for the operations was the surface as shown in Figure 31 from run 362 and Figure 35 from run 362. Successive odd numbered passes on the front of this surface are shown in Figure 36 a through f, and the rear portion in Figure 37 a through f. The number of the smoothing pass is indicated by the SP abbreviation in the caption. The results were plotted to see actually what eleven smoothing passes mean to a surface. It is evident that a reasonably acceptable surface is achieved after three passes, and certainly by five. After that the smoothing hardly changes the basic structure. It is interesting to note how the deep valley on the back portion of the surface at $\log(\text{dosage}) = 13.55$ is eventually eliminated by successive smoothing passes. The criteria telling us when enough smoothing has been performed is essentially subjective. However a more

VOLTAGE NEUTRON RADIATION RESPONSE OF 9704 RUN J42

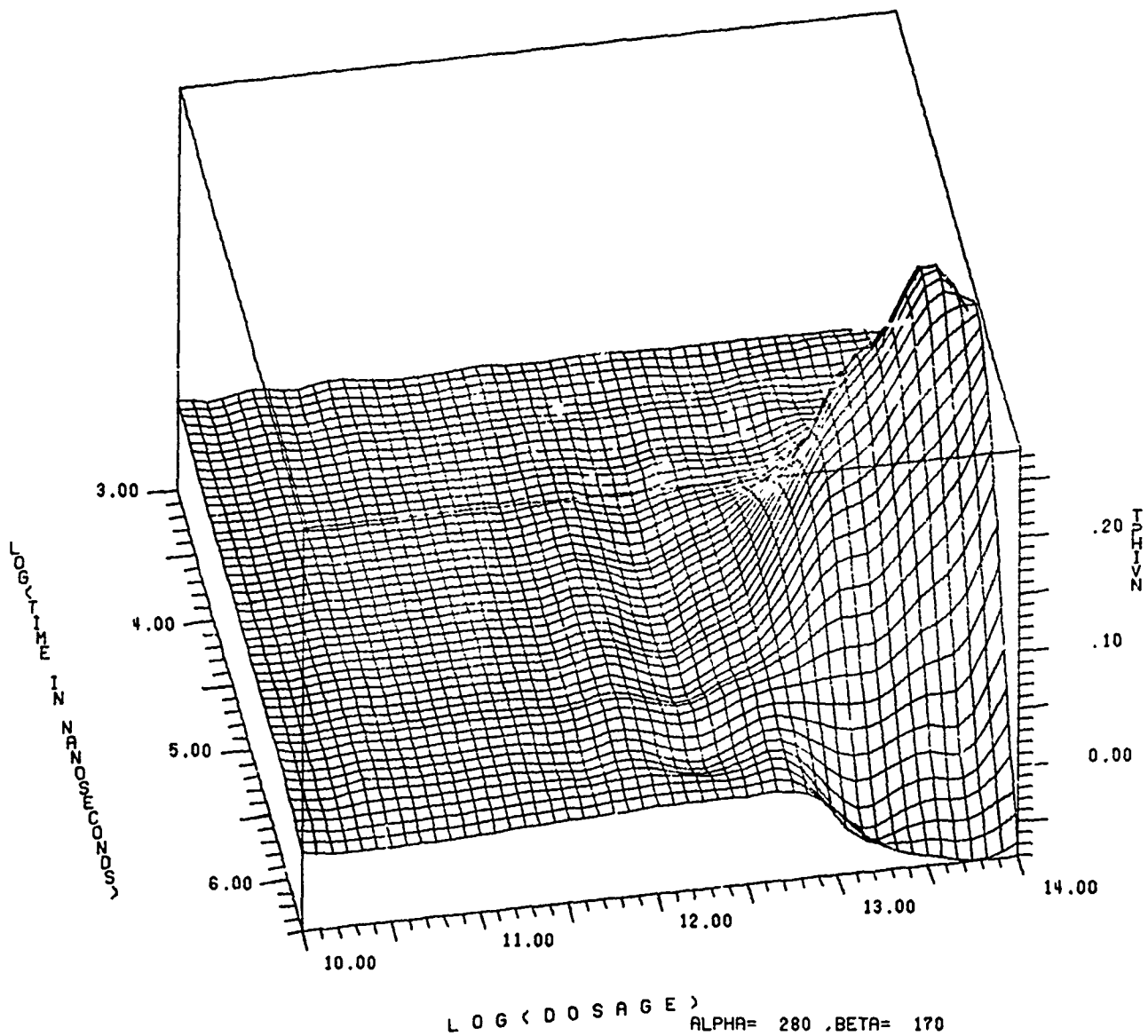


Figure 32b. Rear View of Voltage Neutron Radiation Response of 9704, Second Computer Fitting

VOLTAGE NEUTRON RADIATION RESPONSE OF 9704 RUN 342

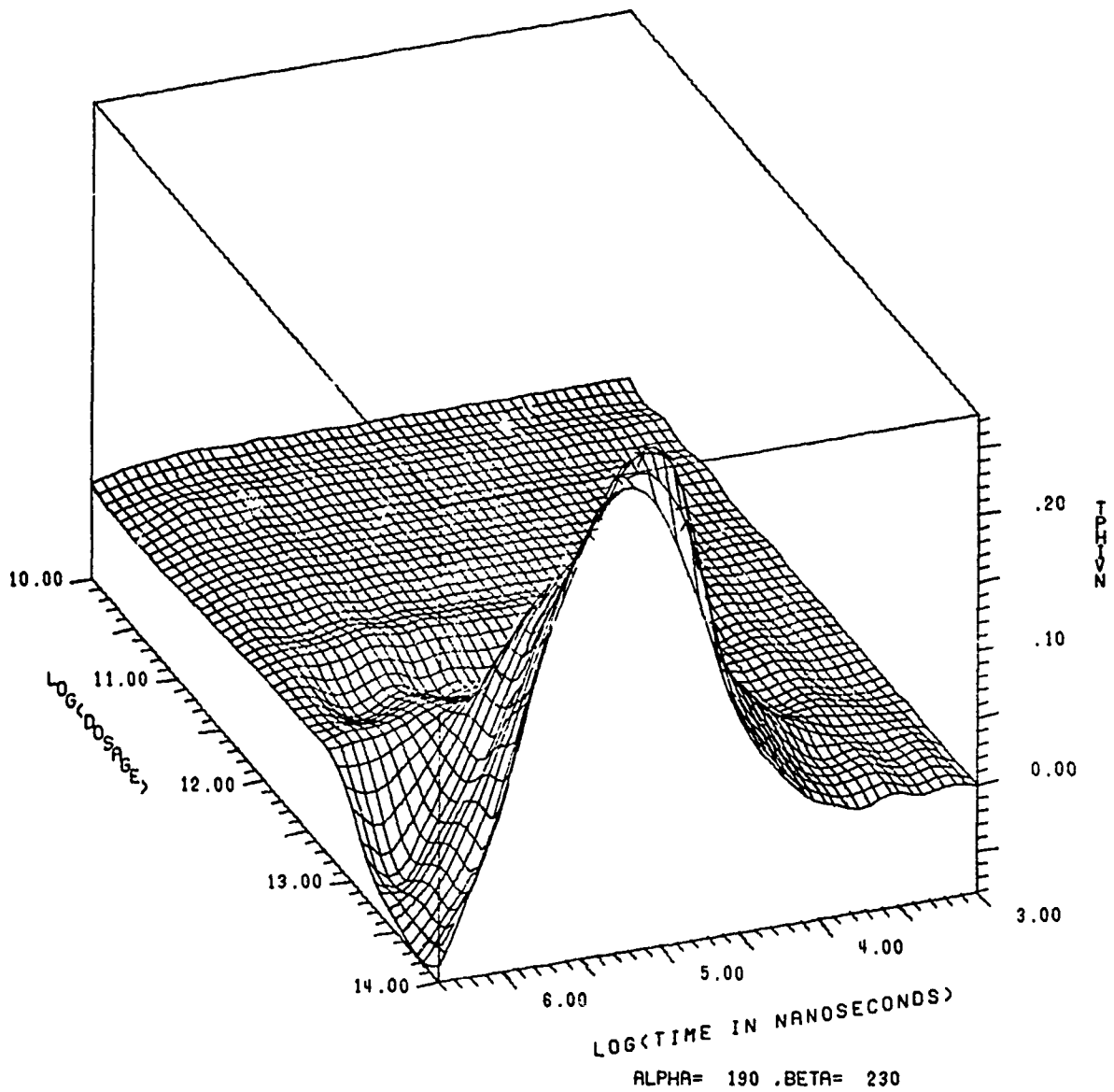


Figure 32c. Back Side View of Voltage Neutron Radiation Response of 9704. Second Computer Fitting

VOLTAGE NEUTRON RADIATION RESPONSE OF 9704 RUN 342

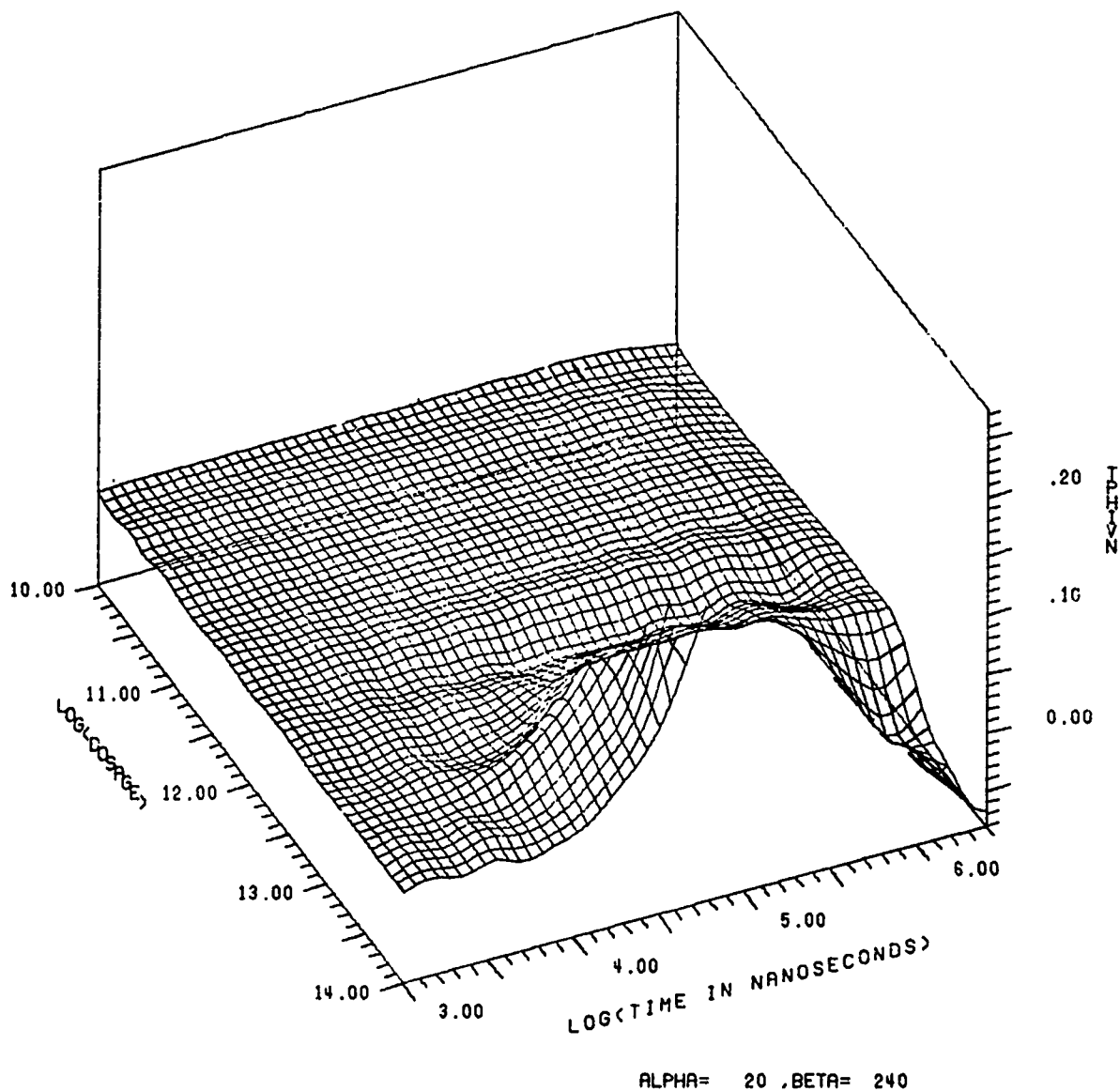


Figure 32d. Bottom View of Voltage Neutron Radiation Response of 9704, Second Computer Fitting

VOLTAGE NEUTRON RADIATION RESPONSE OF 9704 RUN #71

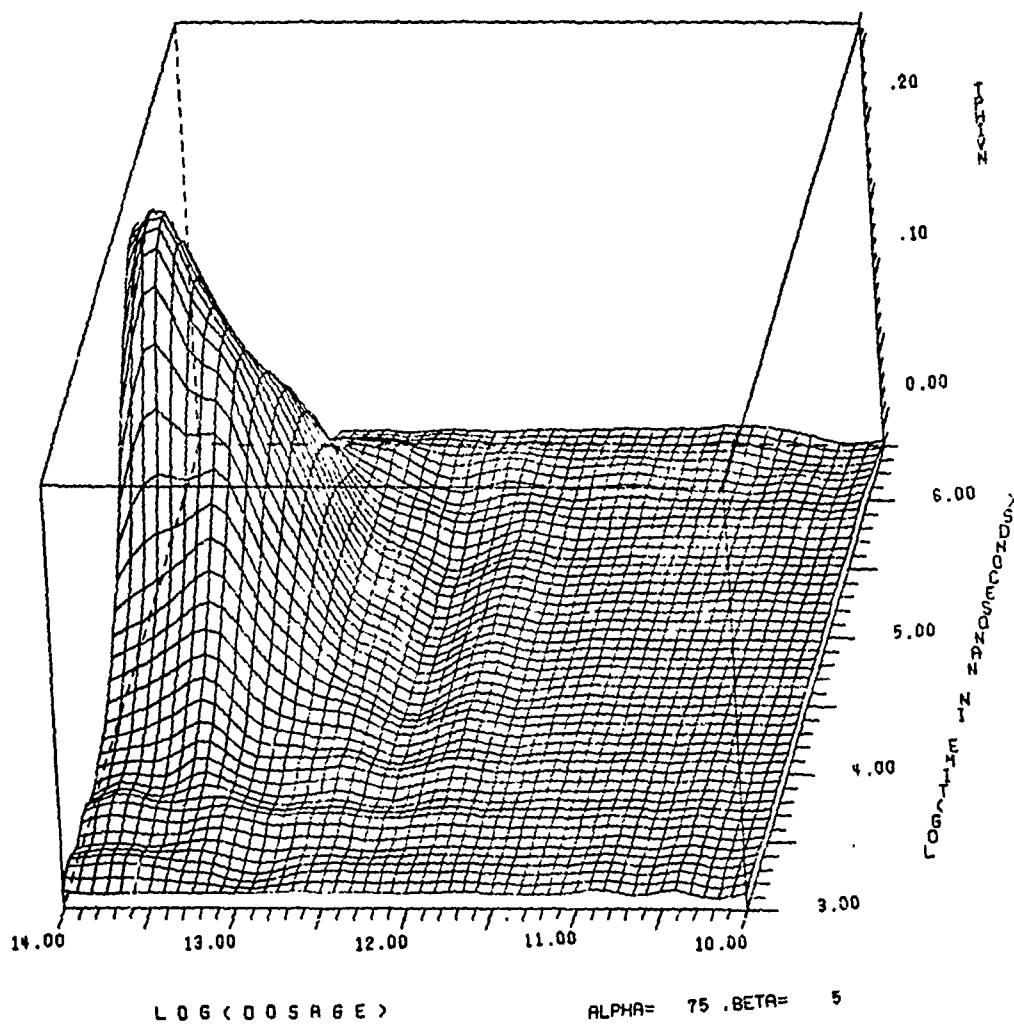


Figure 33a. Front View of Voltage Neutron Radiation Response of 9704, Third Fitting by Computer

VOLTAGE NEUTRON RADIATION RESPONSE OF 9704 RUN 467

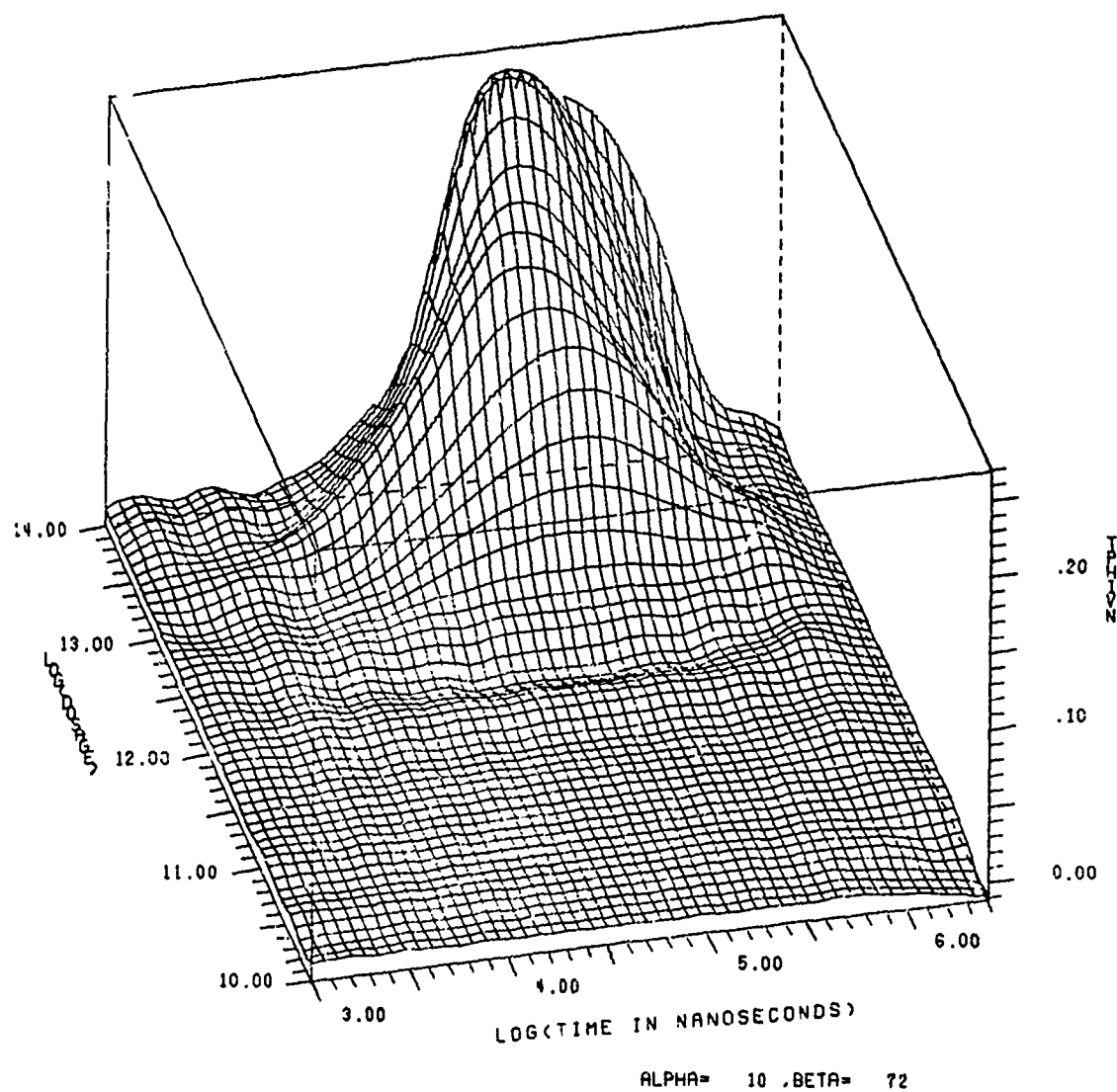


Figure 33b. Side View of Voltage Neutron Radiation Response of 9704, Third Fitting by Computer

VOLTAGE NEUTRON RADIATION RESPONSE OF 9704 RUN 467

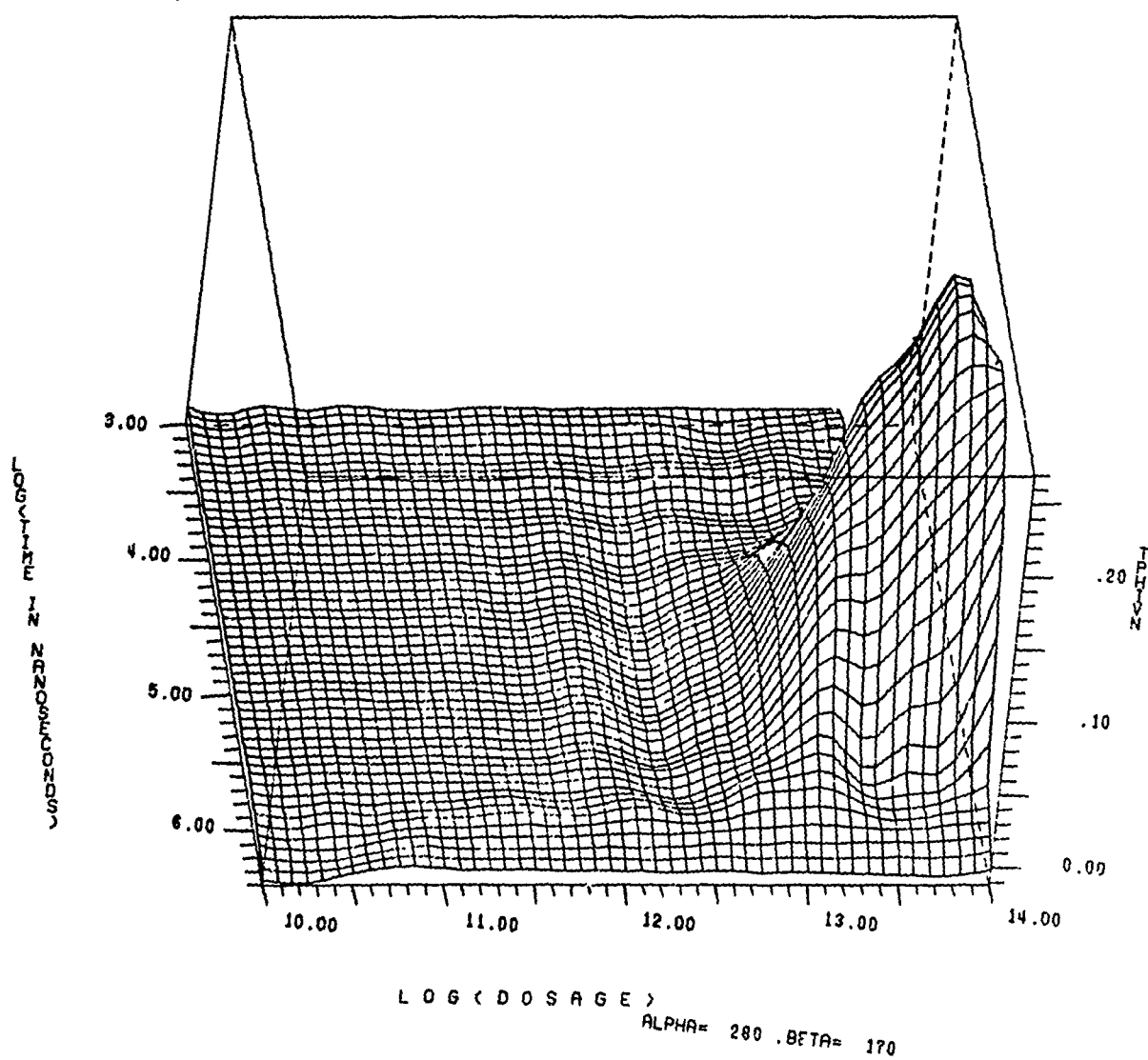


Figure 33c. Back View of Voltage Neutron Radiation Response of 9704. Third Fitting by Computer

CURRENT NEUTRON FUNCTION OF 9704 RUN 362

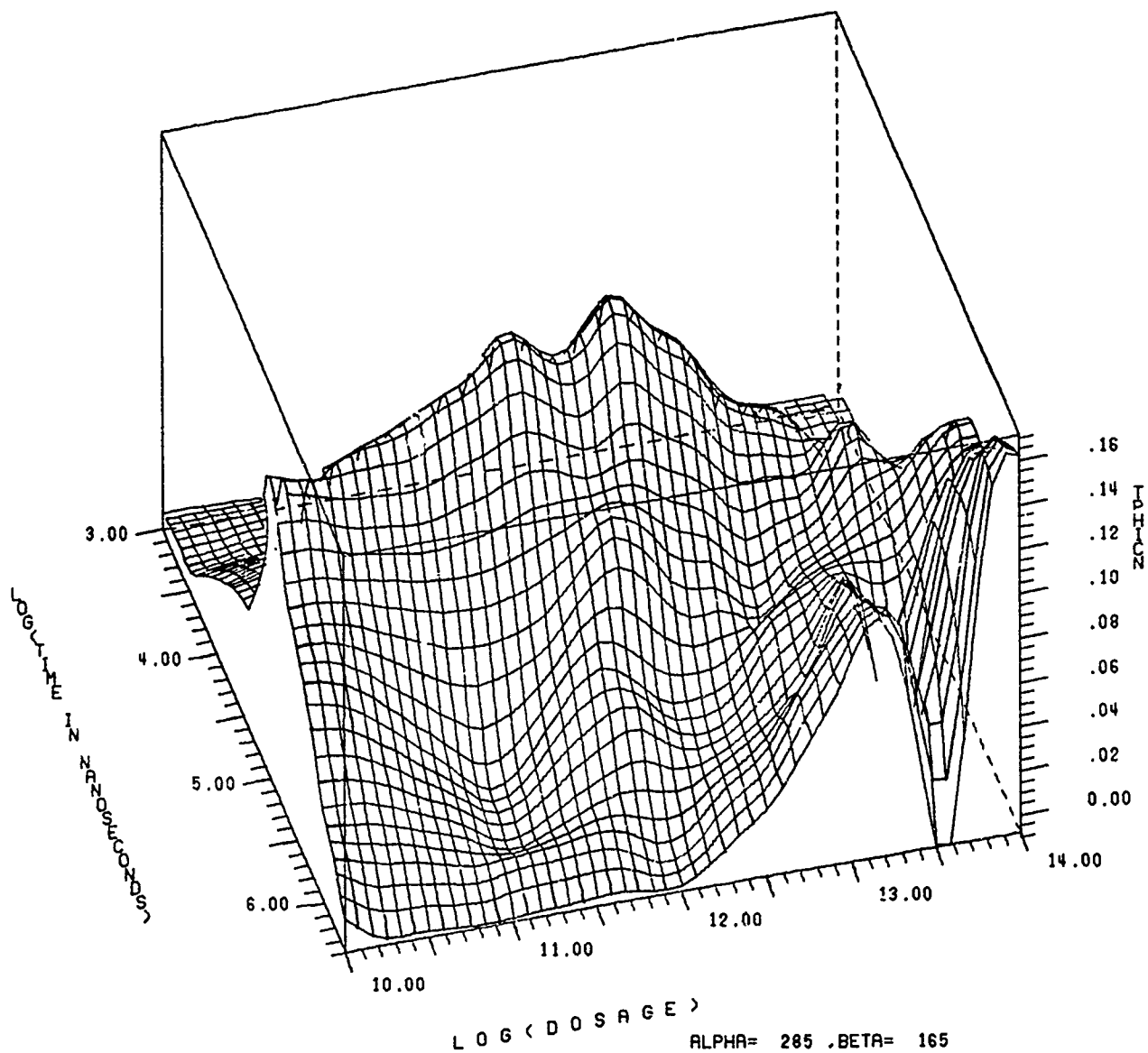


Figure 34. Rear View of Current Neutron Function of 9704.
Second Fitting by Computer

CURRENT NEUTRON FUNCTION OF 9704 RUN 362

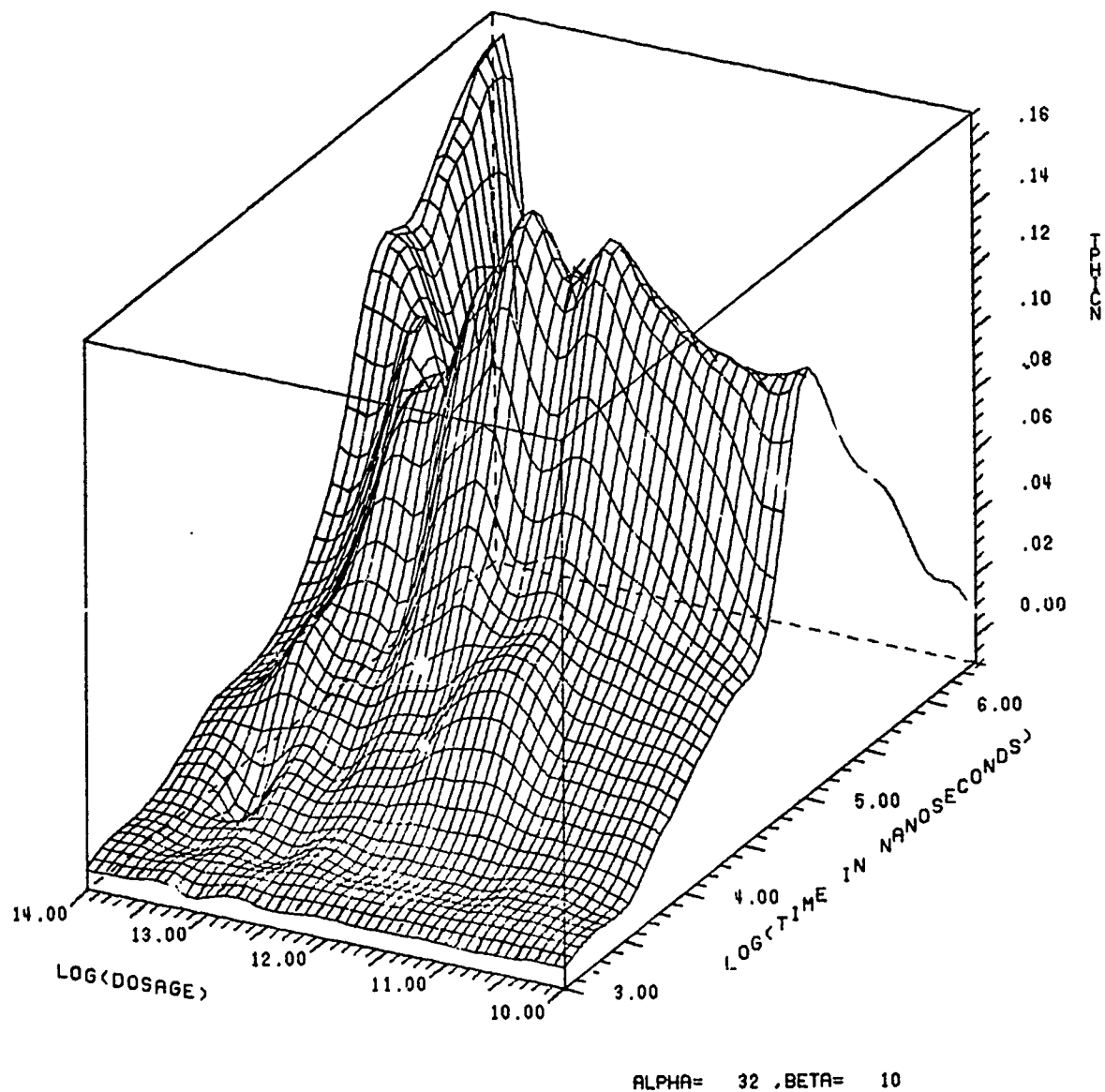


Figure 35. Front View of Current Neutron : action of 9704.
Prior to Computer Smoothing

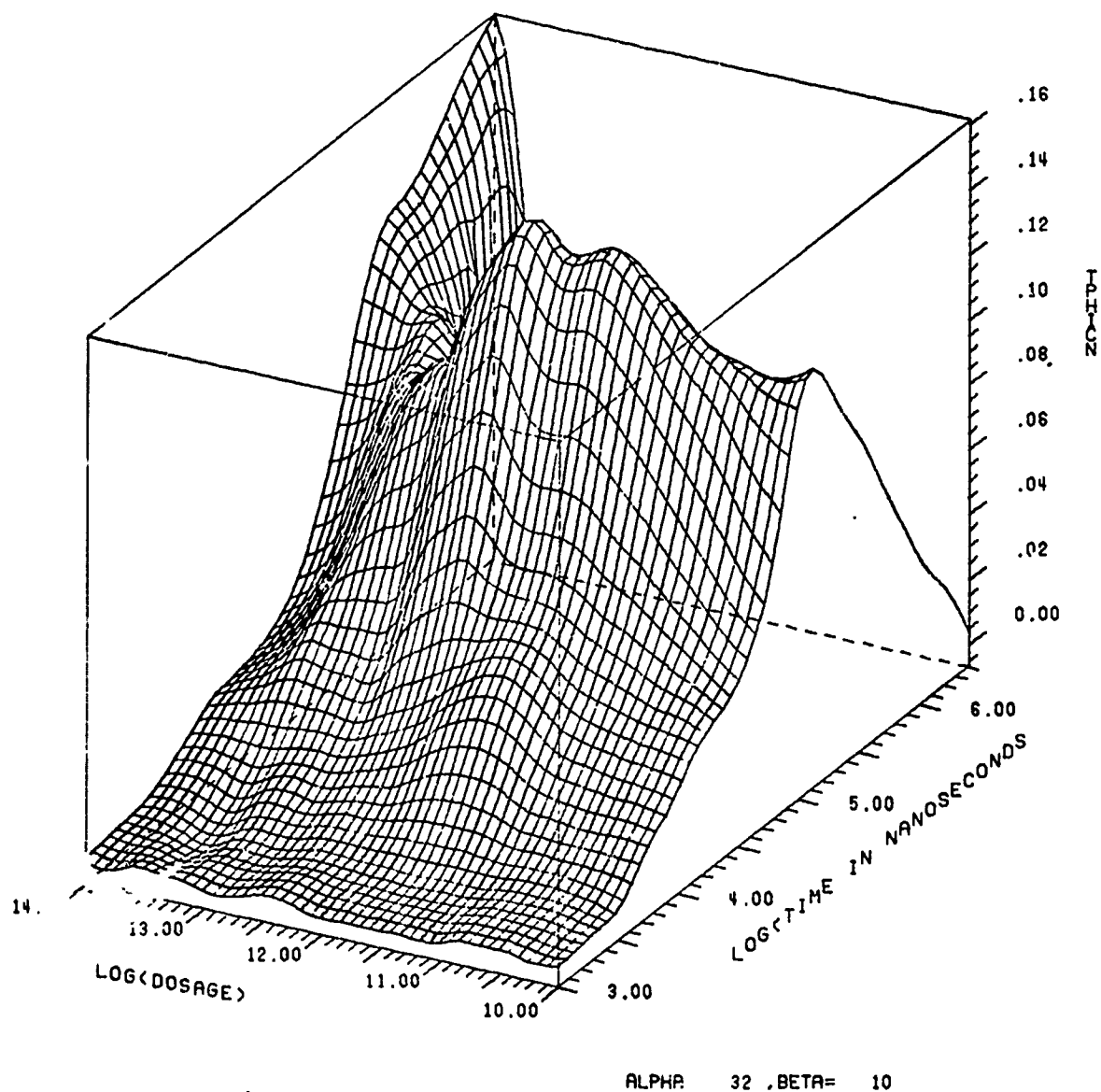


Figure 30a. Front View of Current Neutron Radiation Response of 9704, After One Smoothing Pass

CURRENT NEUTRON RADIATION RESPONSE OF 9704 RUN 415 3 SP

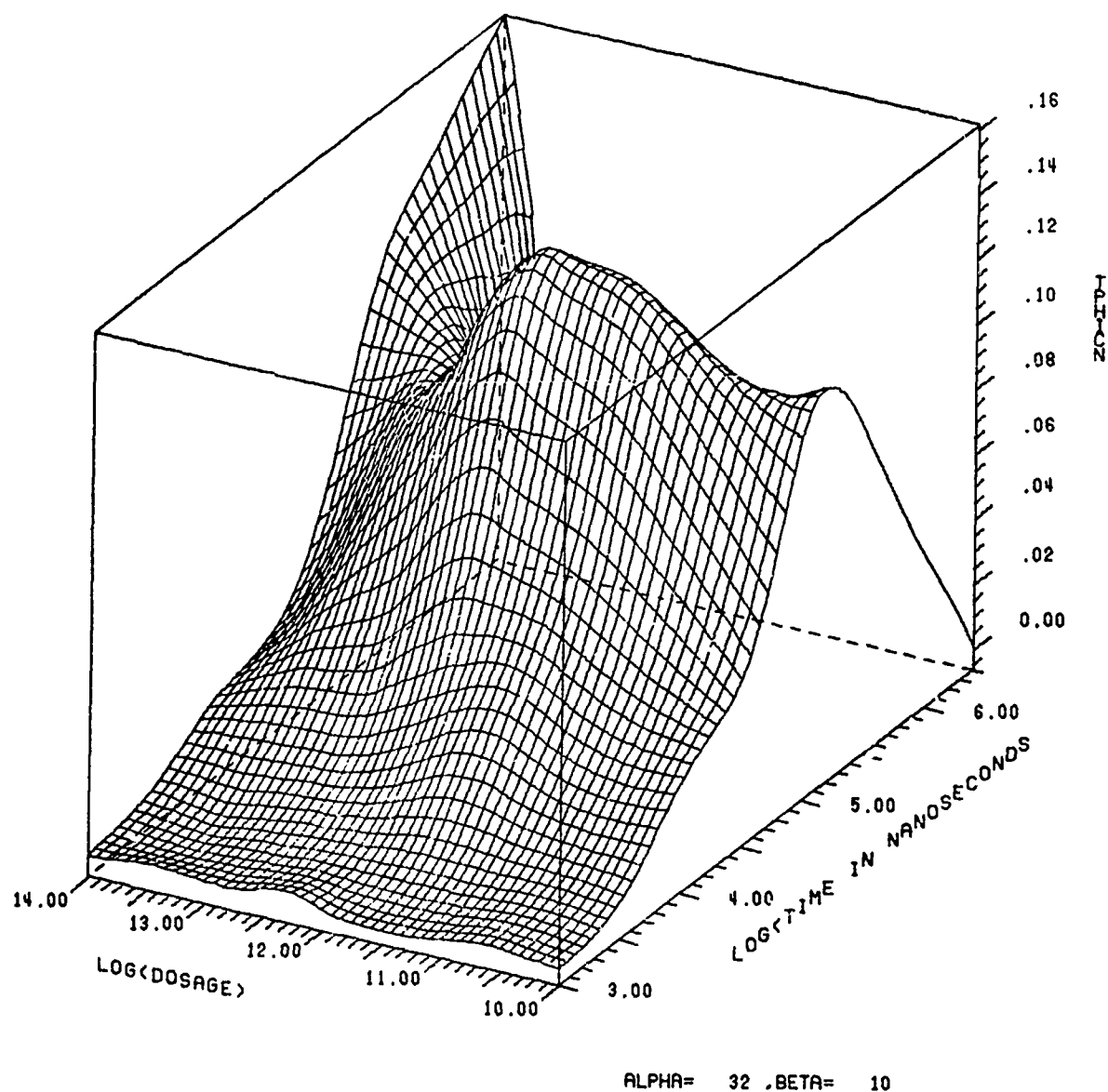


Figure 36b. Front View of Current Neutron Radiation Response of 9704, After Three Smoothing Passes

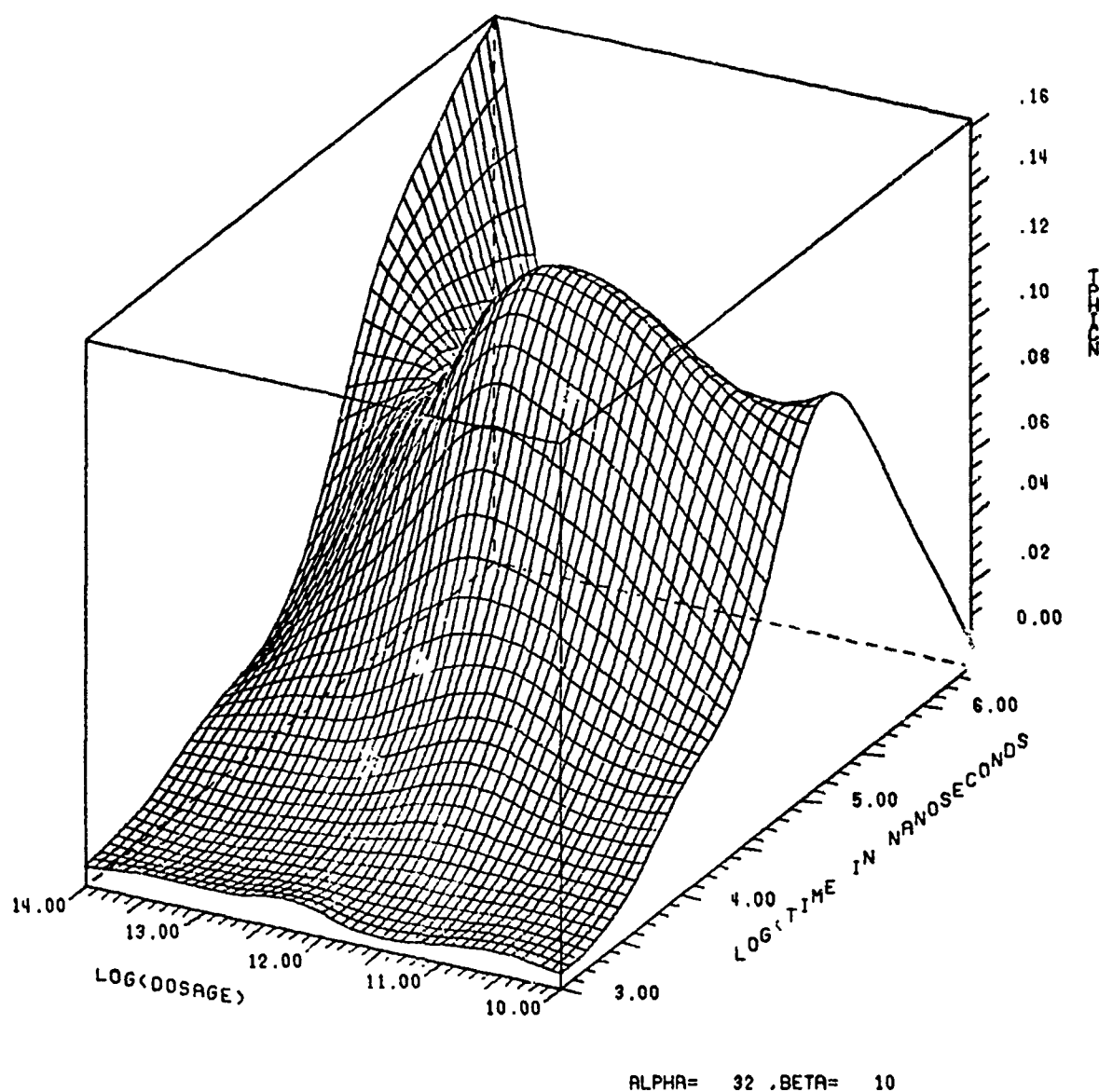


Figure 36c. Front View of Current Neutron Radiation Response of 9704, After Five Smoothing Passes

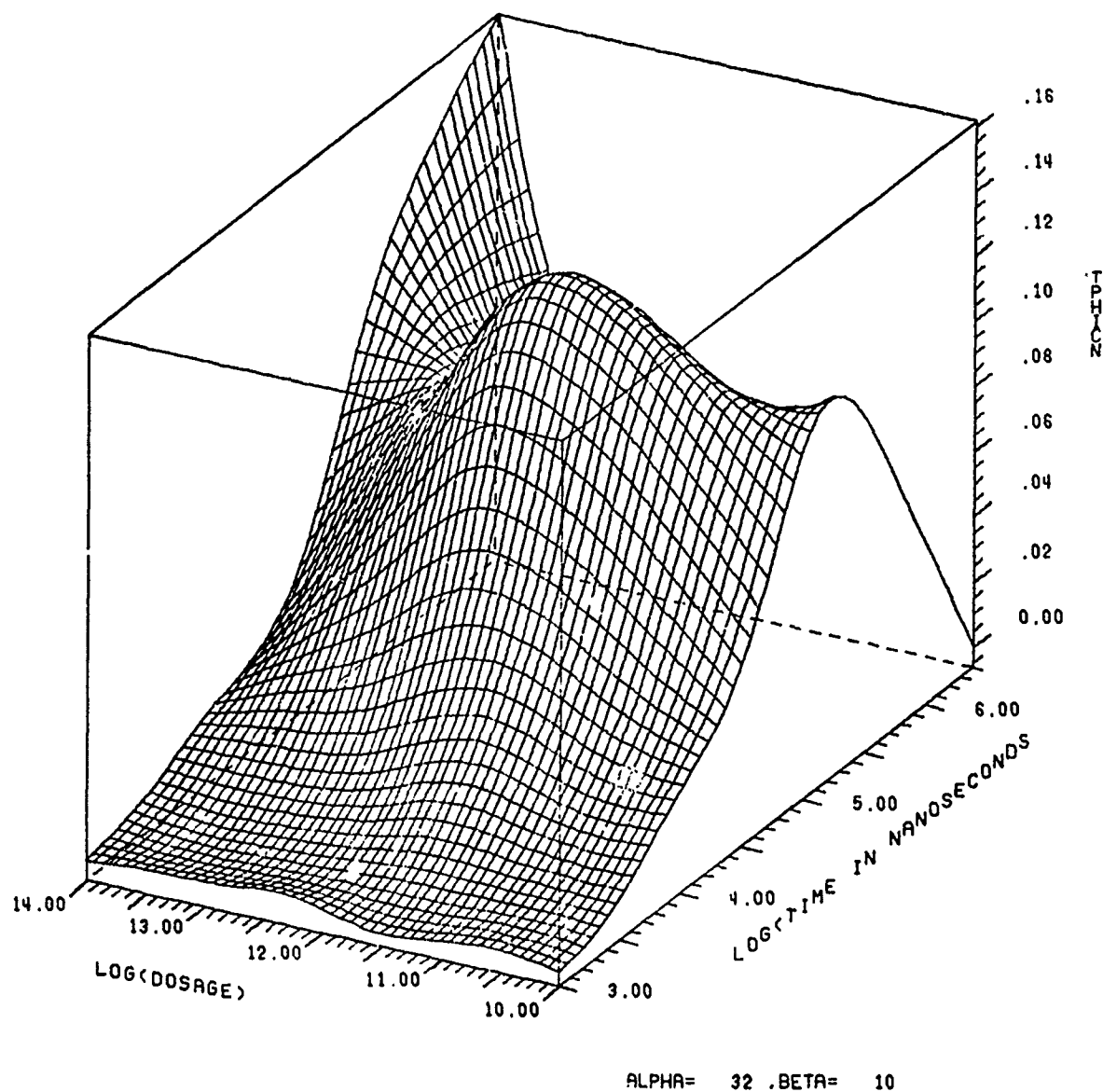


Figure 36d. Front View of Current Neutron Radiation Response of 9704, After Seven Smoothing Passes

CURRENT NEUTRON RADIATION RESPONSE OF 9704 RUN 420 9 SP

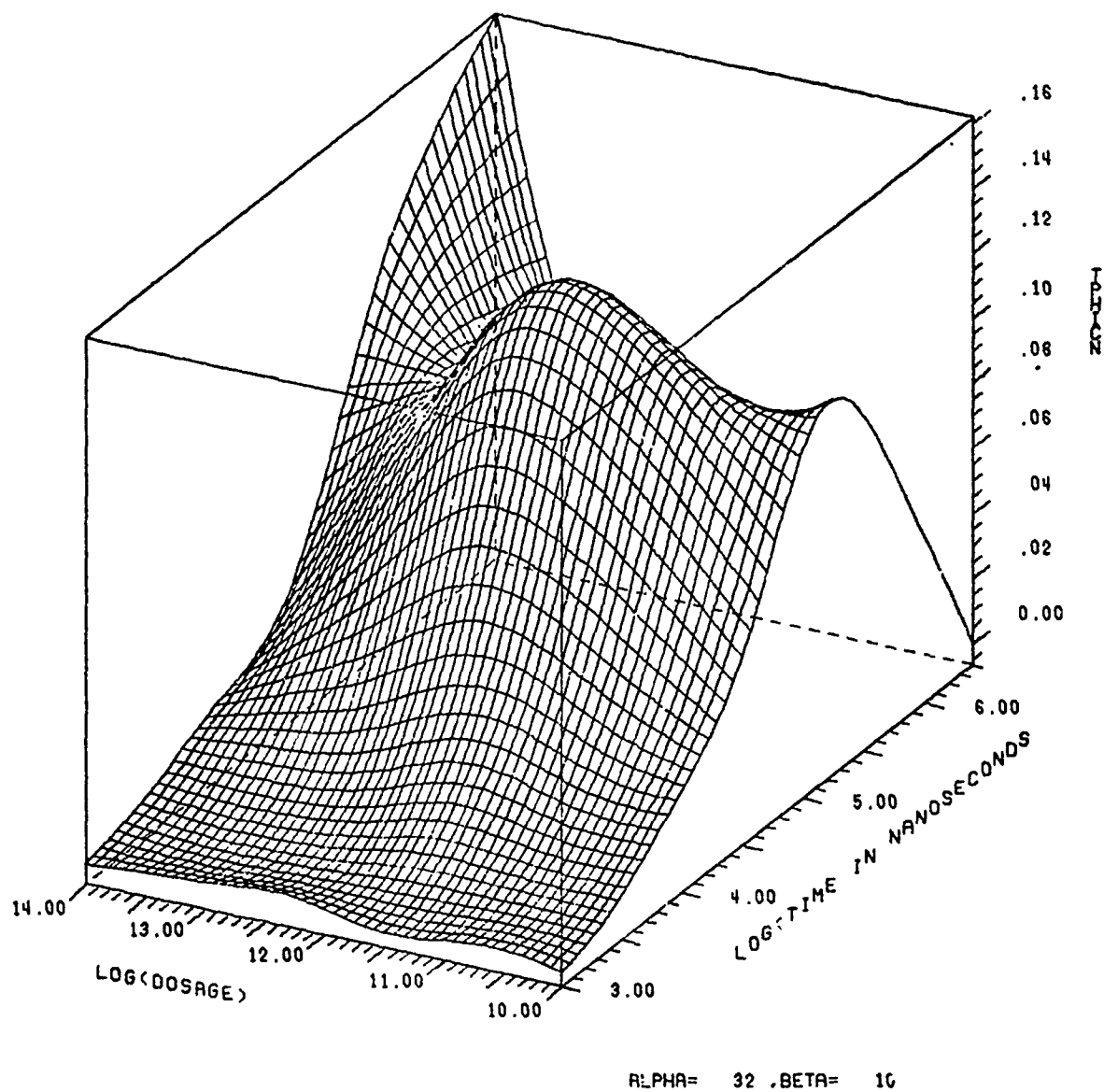


Figure 36e. Front View of Current Neutron Radiation Response of 9704, After Nine Smoothing Passes

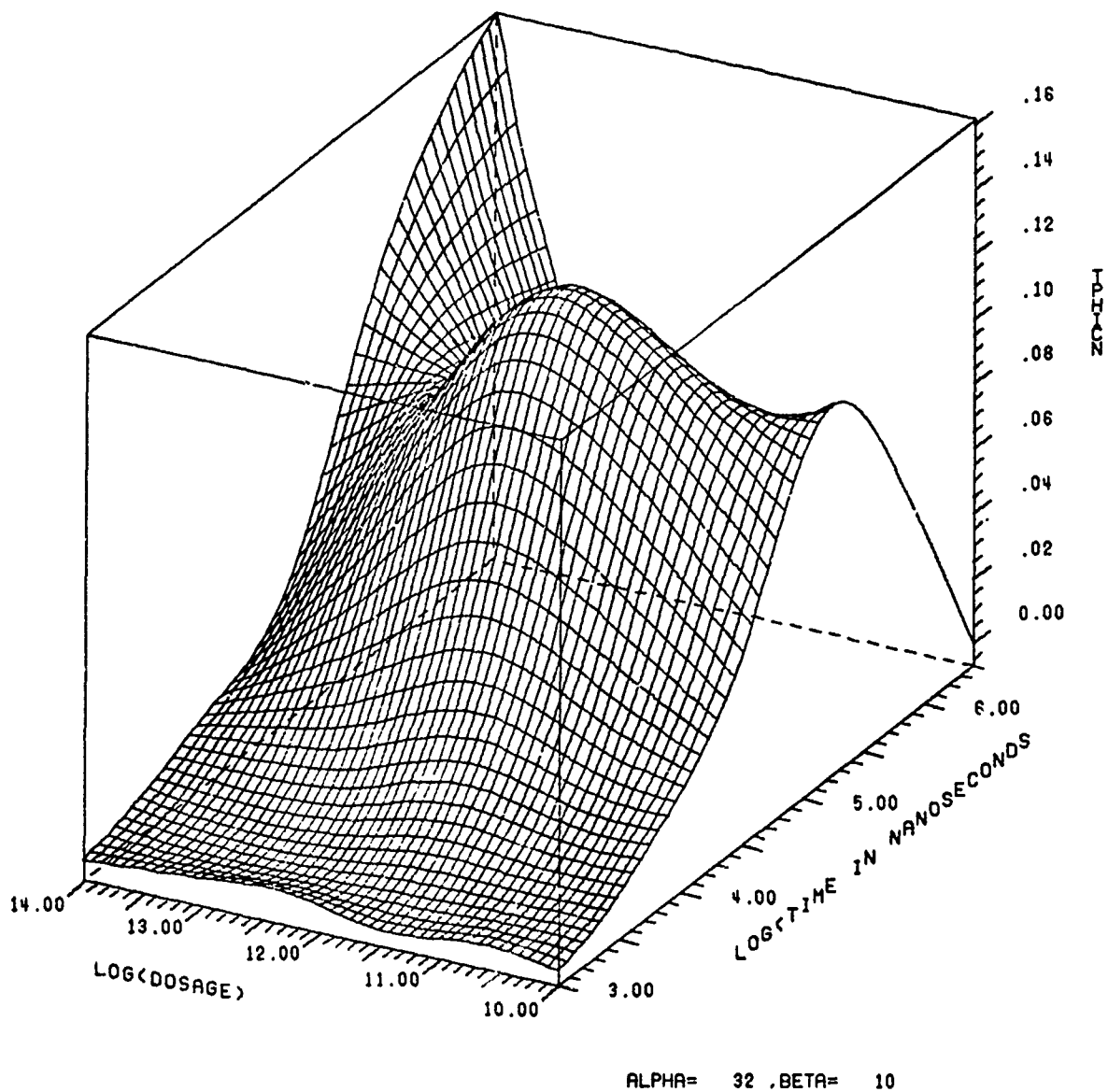


Figure 36f. Front View of Current Neutron Radiation Response of 9704, After Eleven Smoothing Passes

CURRENT NEUTRON RADIATION RESPONSE OF 9704 RUN 408

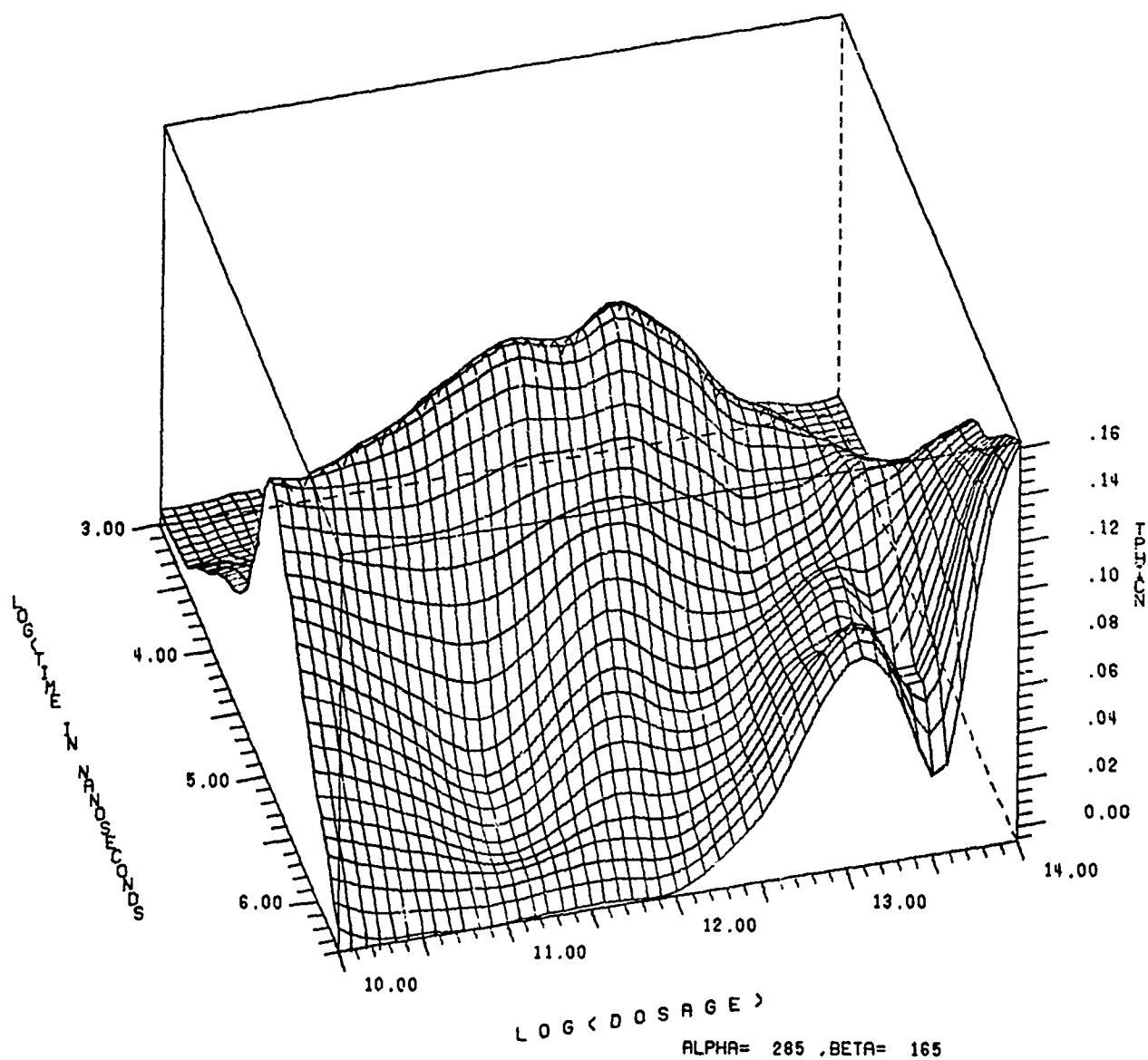


Figure 37a. Rear View of Current Neutron Radiation Response of 9704, After One Smoothing Pass

CURRENT NEUTRON RADIATION RESPONSE OF 9704 RUN 415 3 SP

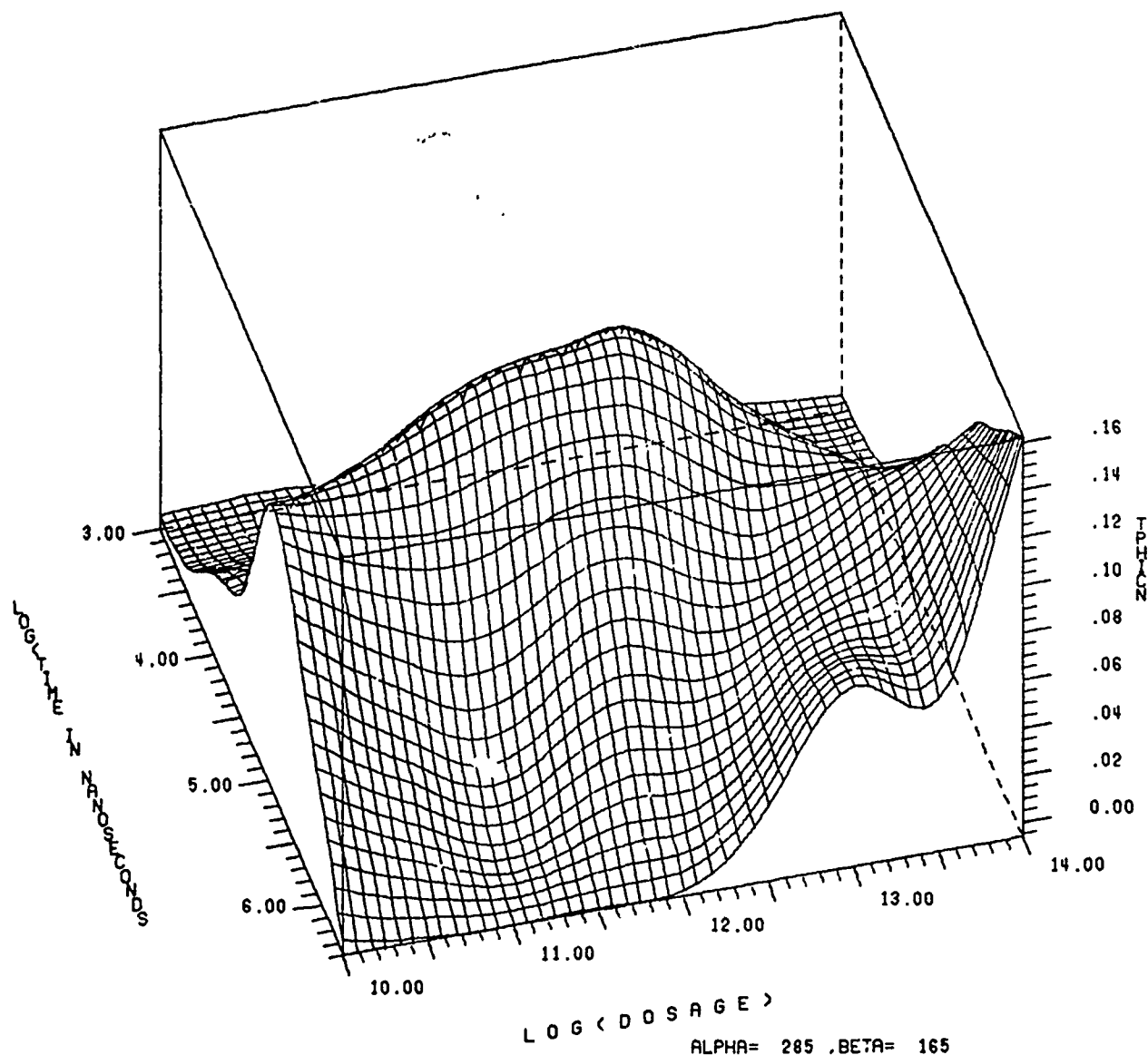


Figure 37b. Rear View of Current Neutron Radiation Response of 9704, After Three Smoothing Passes

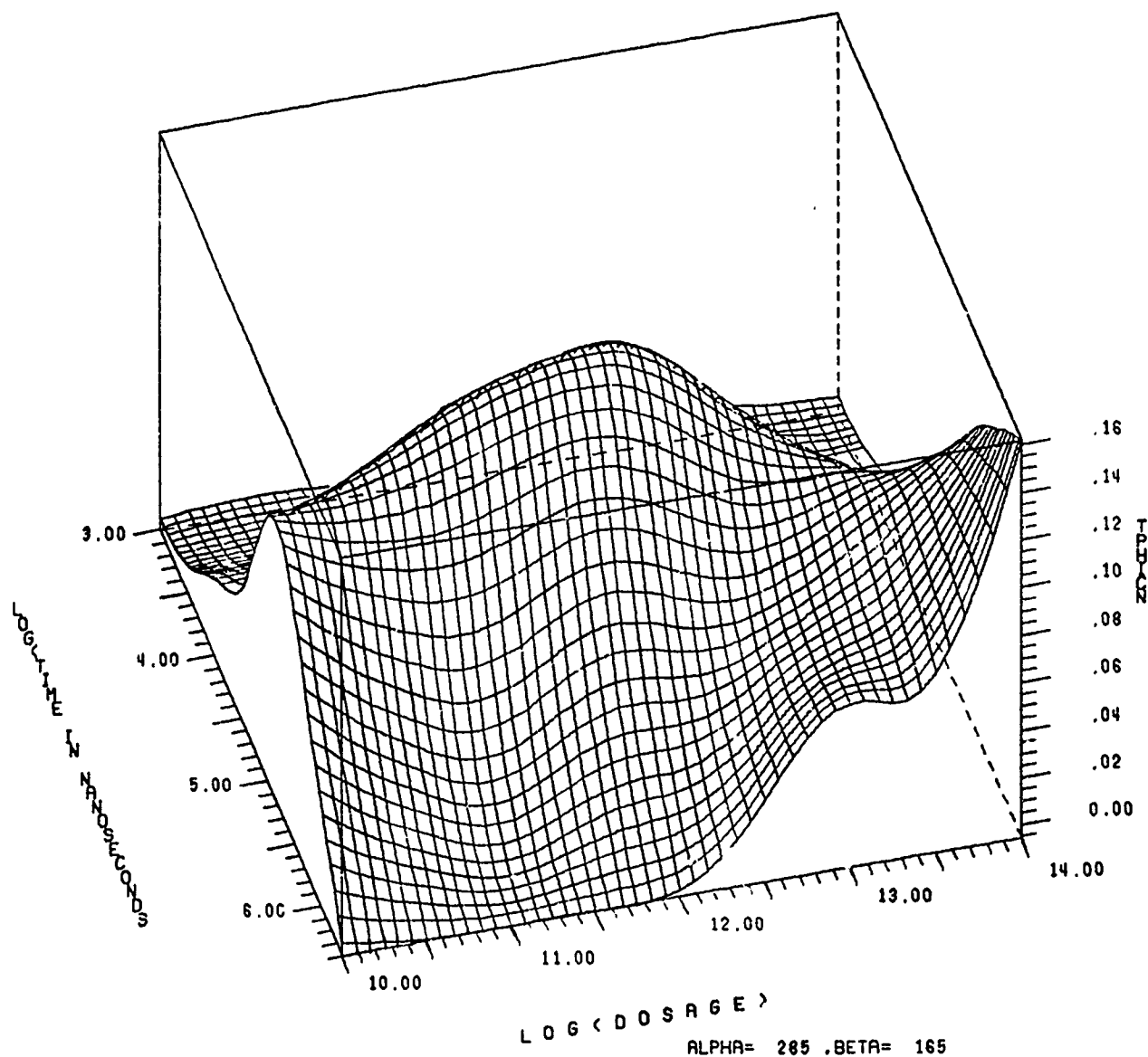


Figure 37c. Rear View of Current Neutron Radiation Response of 9704. After Five Smoothing Passes

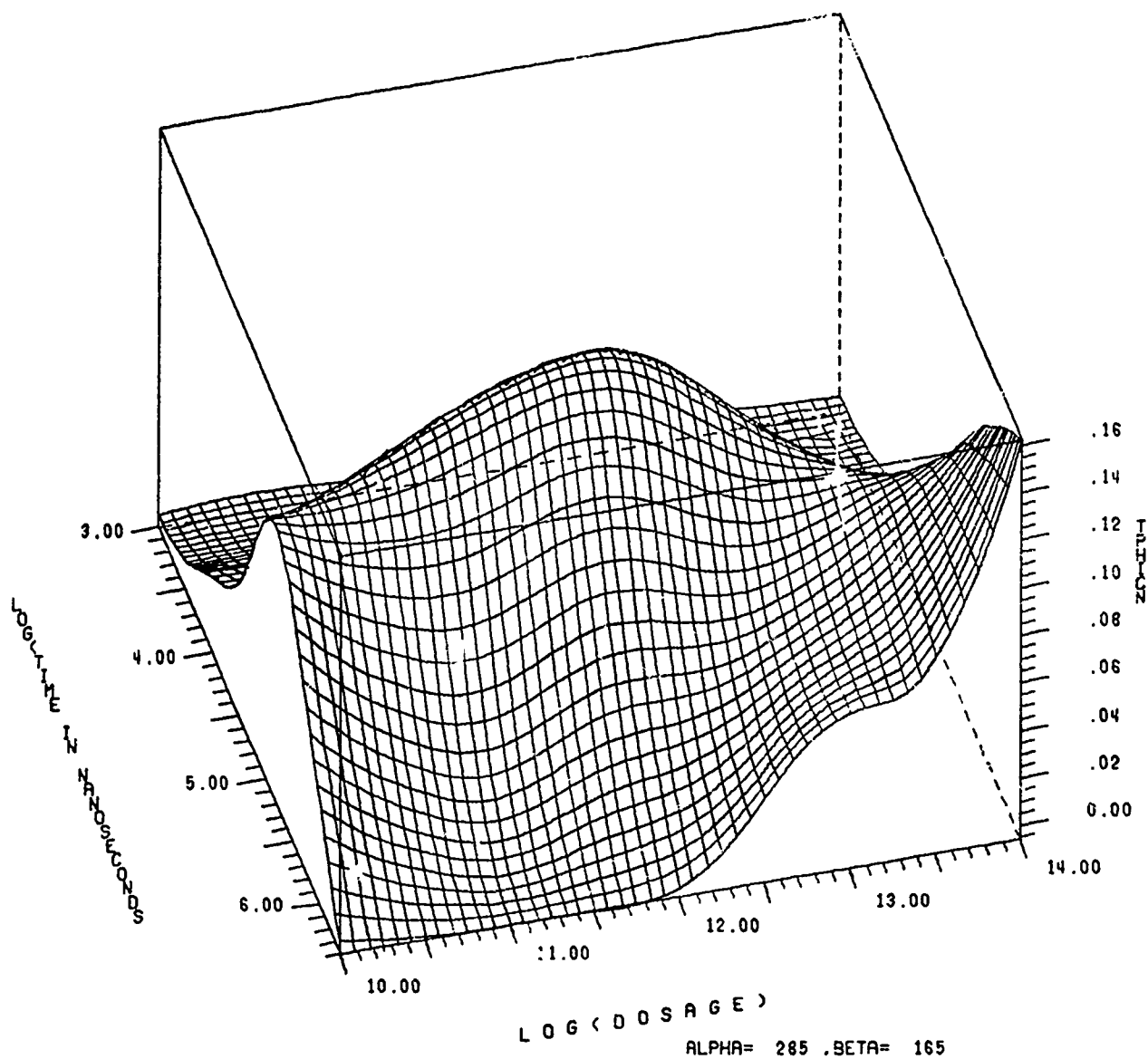


Figure 37d. Rear View of Current Neutron Radiation Response of 9704, After Seven Smoothing Passes

CURRENT NEUTRON RADIATION RESPONSE OF 9704 RUN 420 9 SP

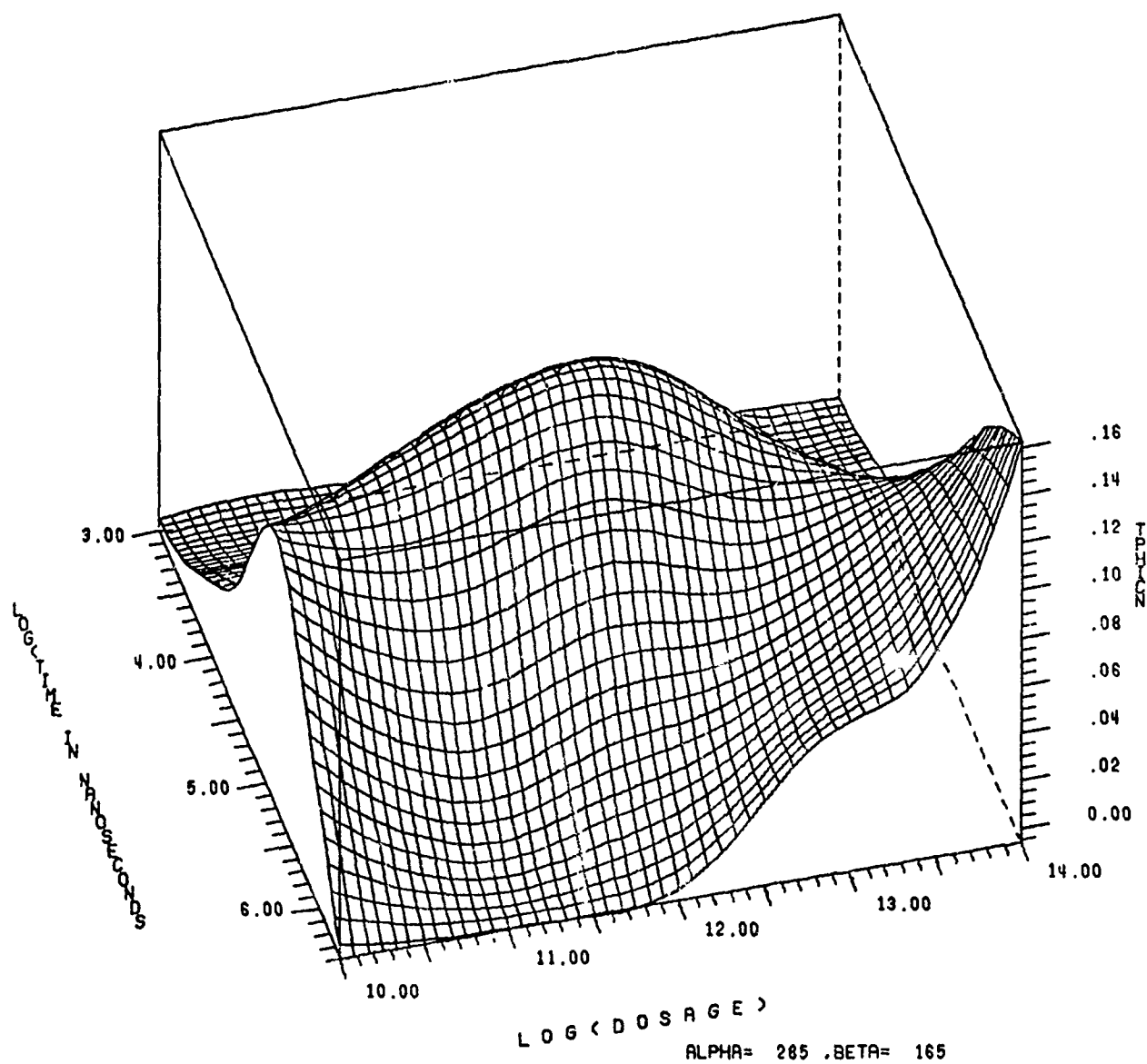


Figure 37e. Rear View of Current Neutron Radiation Response of 9704, After Nine Smoothing Passes

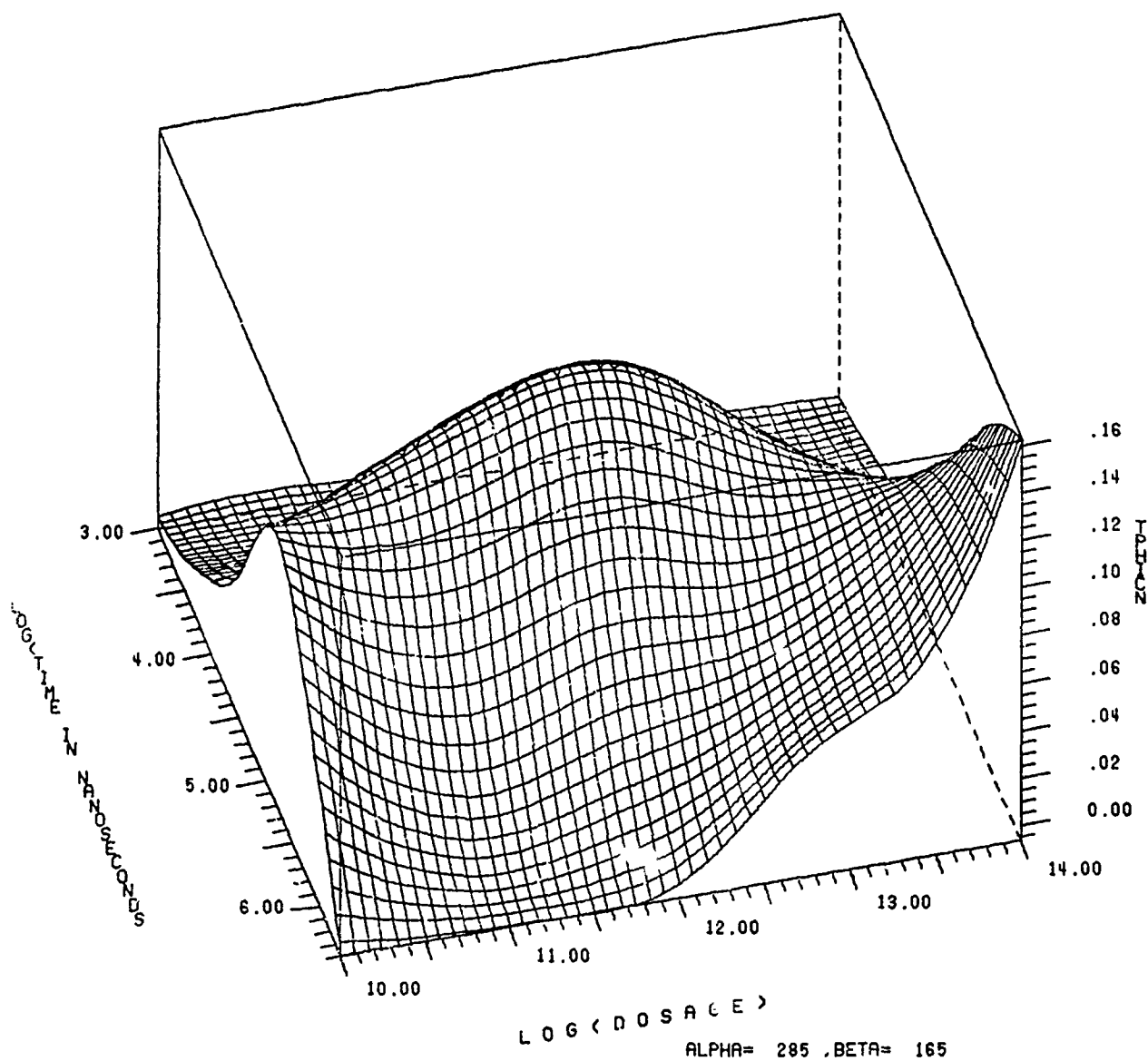


Figure 37f. Rear View of Current Neutron Radiation Response of 9704, After Eleven Smoothing Passes

quantitative criteria could probably be developed based on the amount of relative shifting that takes place on successive passes. Thus when the basic structure is reached, the relative shifts from the last pass presumably drop rapidly.

The prime conclusion however from the above example is that an appropriately applied combination of adding and deleting certain data points or strings, followed by astute choice of linear interpolation packing of the data plane, together with judicious application of smoothing and wild-pointing, can yield satisfactory computer representations of the basic experimental data without the necessity of first making a hand perspective drawing. However judgement is still required during the machine processing steps, particularly when treating the results of variability amongst different devices relating to threshold class of responses.

c. Gate Flash X-ray Response Surfaces

The first fitting of the voltage radiation transfer function of the 9704 NAND gate yielded the plot shown in Figure 38. While an initial impression might be that the surface doesn't tell much, on reconsideration it clearly shows the presence of a threshold type of radiation sensitivity, just as the neutron voltage response surface did. It leads to speculation that perhaps the neutron-gamma ratio of the fission burst is not so high that some of the observed neutron results are likely due to the incident gammas. Taking an $n-\gamma$ ratio of 5×10^8 , then a neutron dose of 5×10^{13} gives a gamma dose of 10^5 rad. The time scaling of the FXR is about 5000 times less than that of the SPR, hence, the dose rate or $\dot{\gamma}$ is the order of 5×10^8 rad/sec. The radiation sensitivity threshold for the 9704 gate at the FXR was about 3×10^{10} for a transition in the output voltage logic state level. However, photocurrent contribution were clearly evident at dose levels two or three orders of magnitude less in the current response surfaces.

Additional projection plots of the voltage response function for FXR exposure are shown in Figure 39 a and b. The vertical scale problem evident in Figure 38 was corrected yielding a larger range on the z axis.

Smoothing was applied to this surface with the results of run 502 shown in Figure 39 c, d, e, and f. It is apparent by inspection of the corner of the surface at maximum dose and minimum time, that the smoothing is driving the corner data points downward in an erroneous manner. This is readily observed by comparing Figure 39 b with Figure 39 d. The smoothing does reduce some of the bumps elsewhere

FXR VOLTAGE RADIATION TRANSFER FUNCTION OF 9704

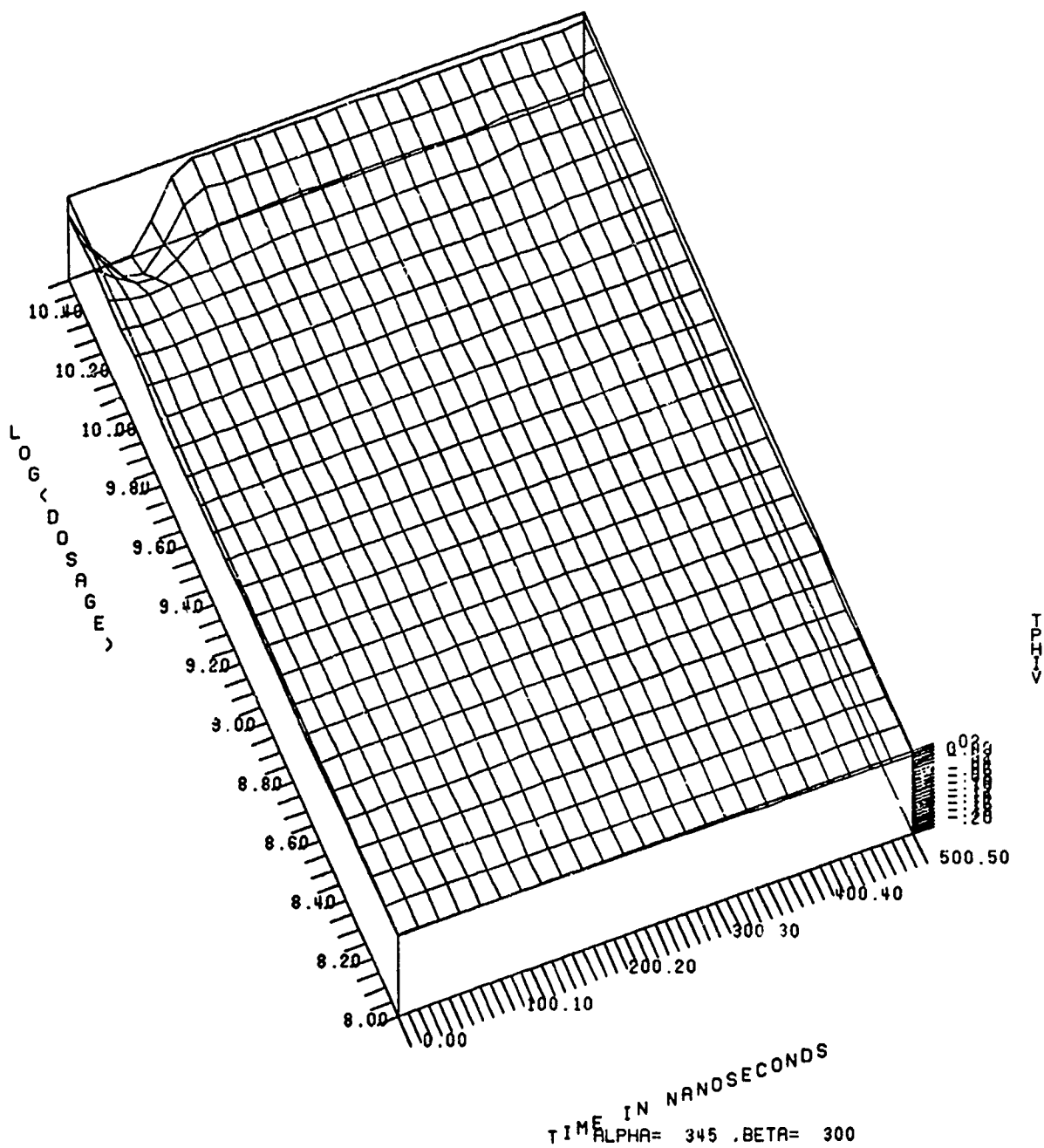


Figure 38.

FXR VOLTAGE RADIATION TRANSFER FUNCTION OF 9704 RUN #502

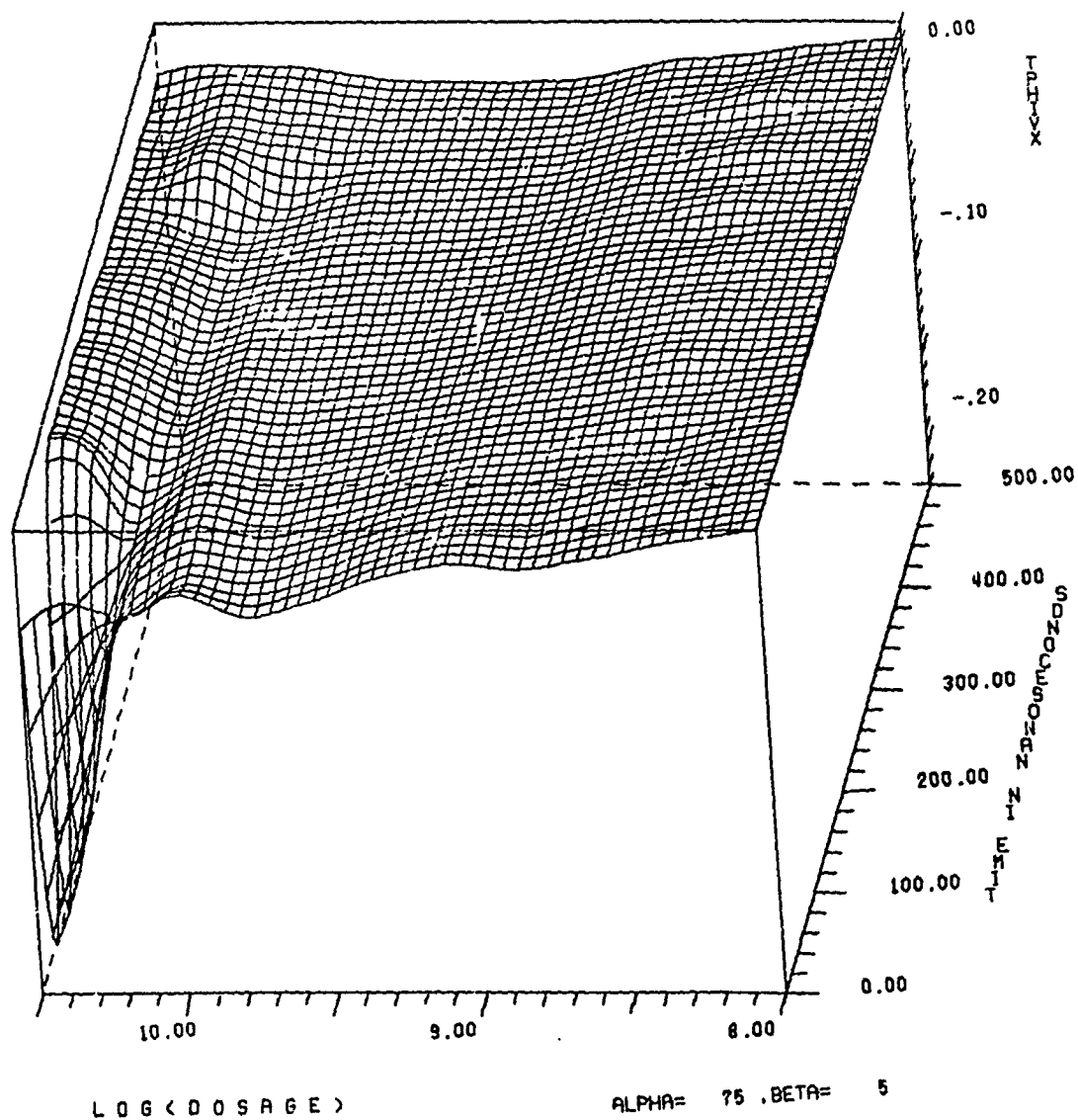


Figure 39a. Front View of FXR Voltage Radiation Transfer Function of 9704 Gate, Note Existence of a Threshold

FXR VOLTAGE RADIATION TRANSFER FUNCTION OF 9704 RUN #502

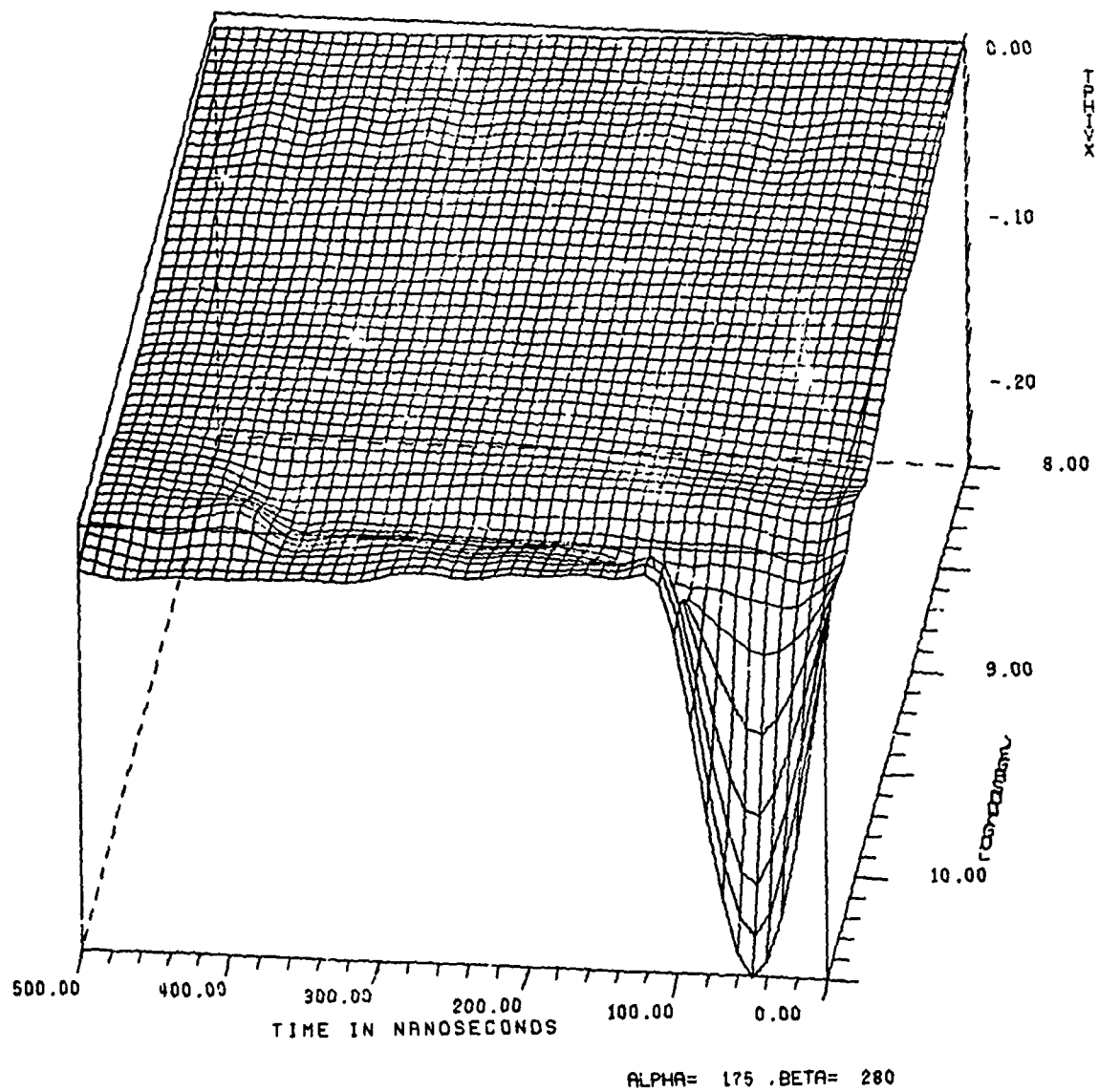


Figure 39b. Side View of FXR Voltage Radiation Transfer Function of 9704 Gate

FXR VOLTAGE RADIATION TRANSFER FUNCTION OF 9704 RUN #502

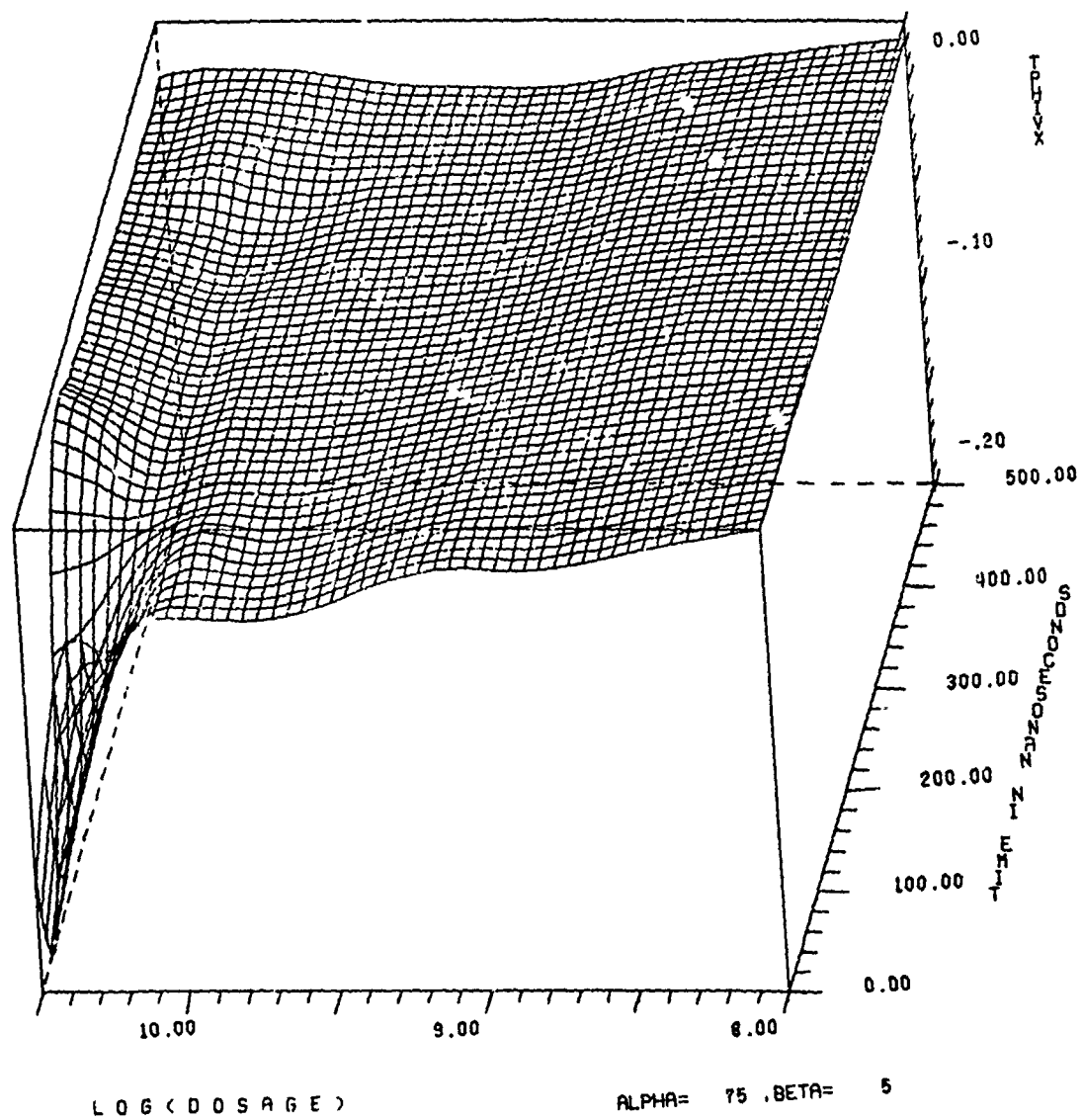


Figure 39c. Front View of FXR Voltage Radiation Transfer Function of 9704 After One Smoothing Pass

FXR VOLTAGE RADIATION TRANSFER FUNCTION OF 9704 RUN #502

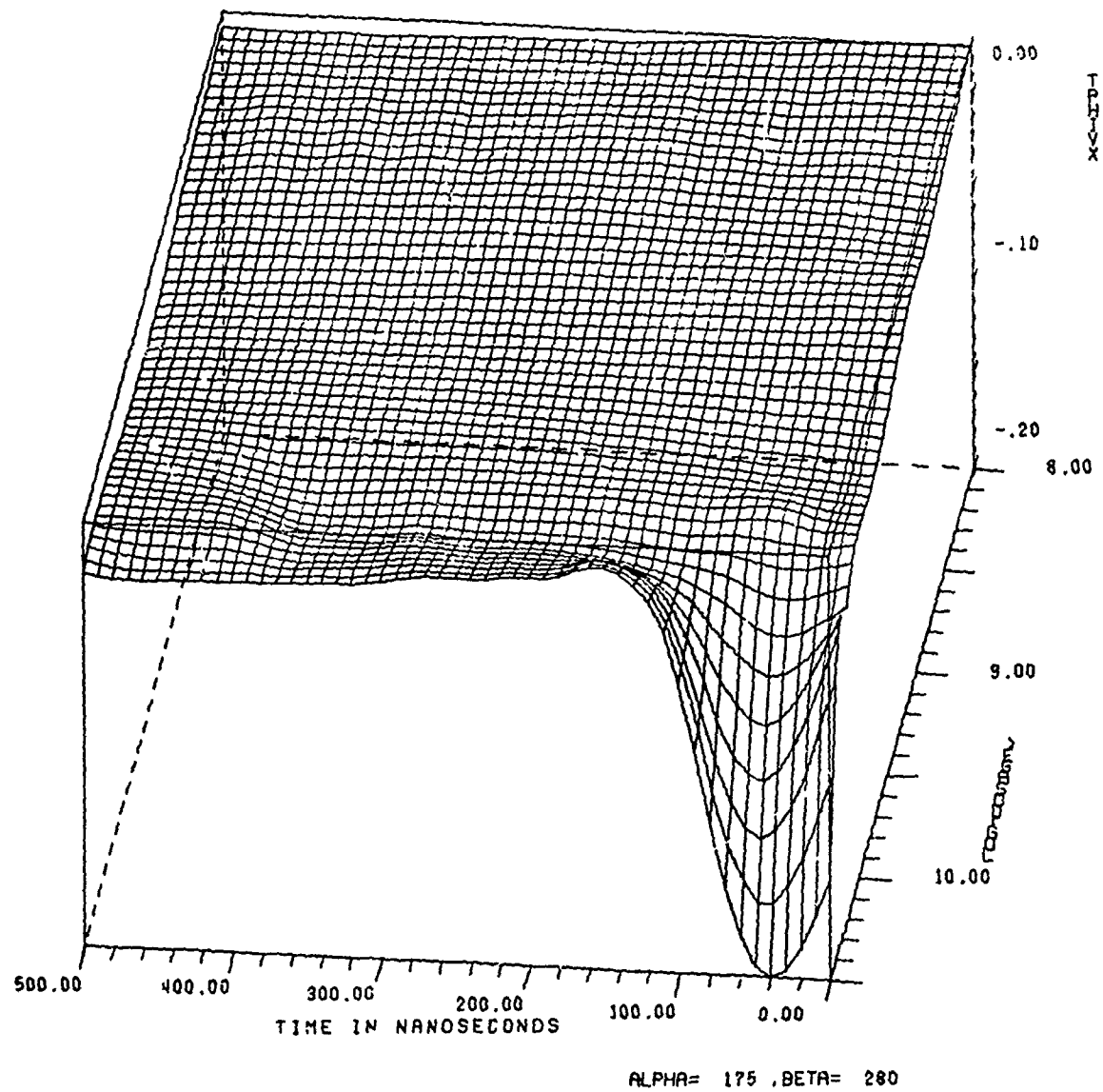


Figure 39d. Side View of FXR Voltage Radiation Transfer Function of 9704 After One Smoothing Pass

FXR VOLTAGE RADIATION TRANSFER FUNCTION OF 9704 RUN #502

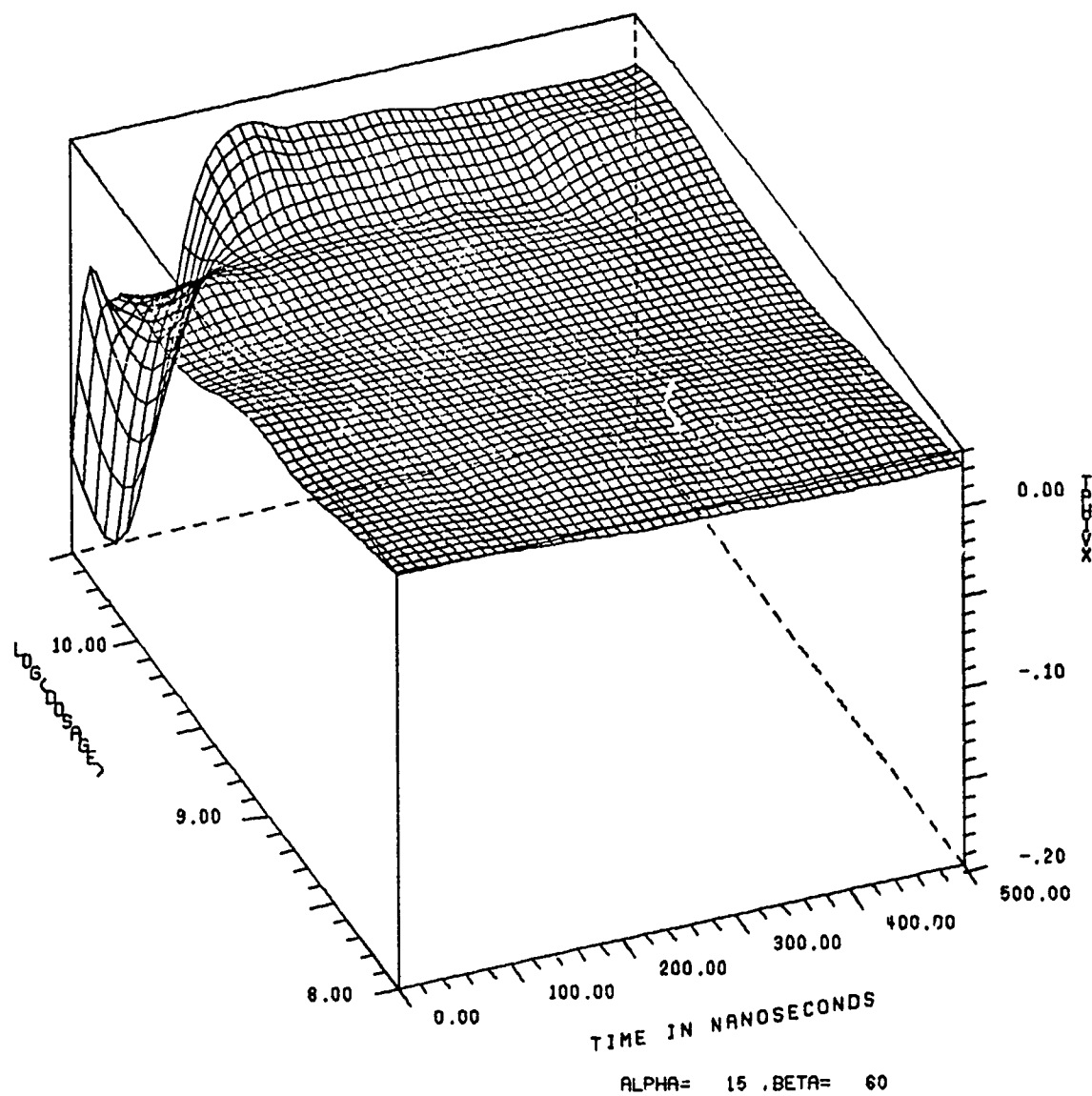


Figure 39e. Oblique View of FXR Voltage Radiation Transfer Function of 9704 After One Smoothing Pass

FXR VOLTAGE RADIATION TRANSFER FUNCTION OF 9704 RUN #502

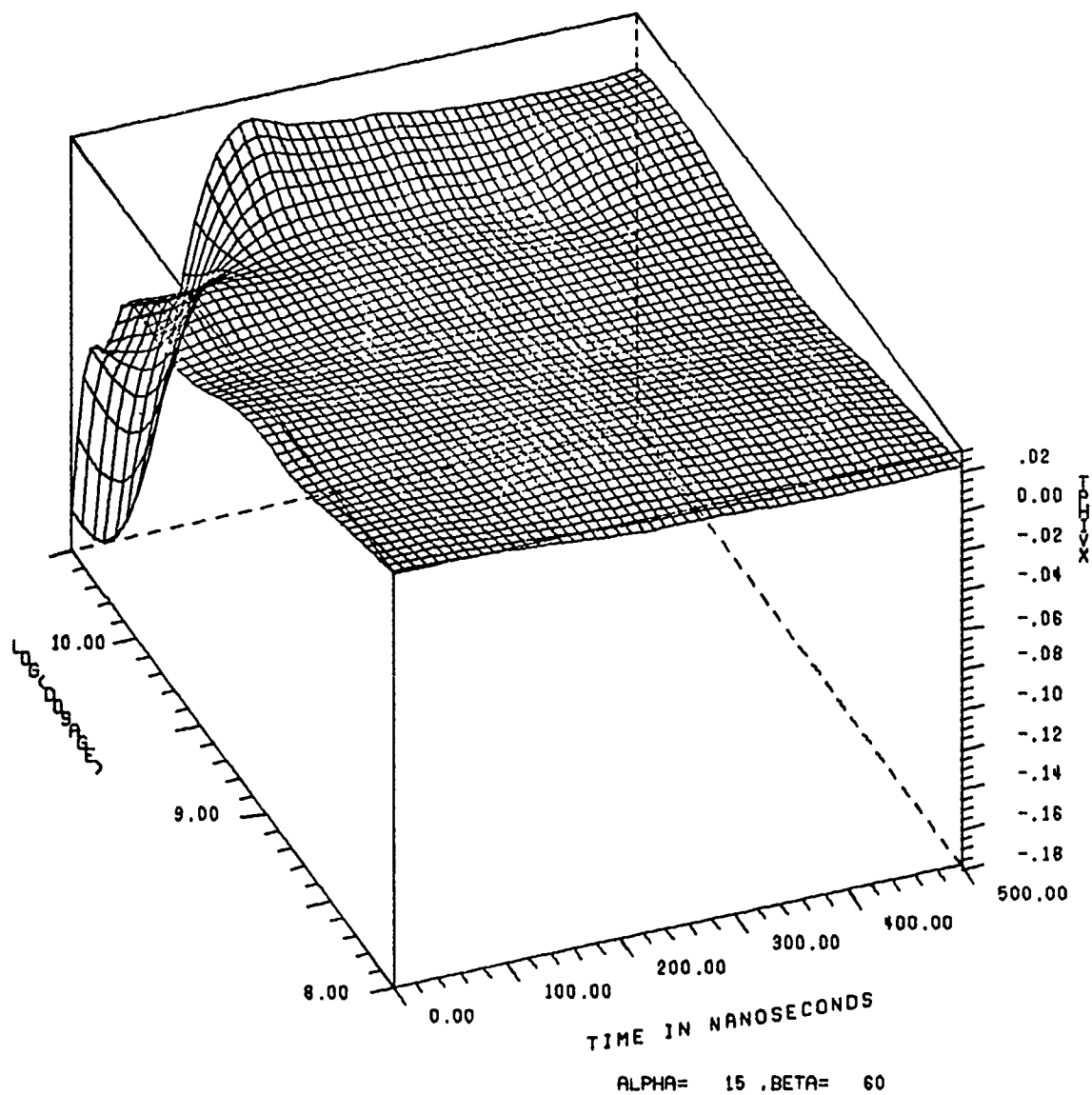


Figure 39f. Oblique View of FXR Voltage Radiation Transfer Function of 9704 After Two Smoothing Passes

in a way which is acceptable. However, the need for anchoring of some data points to hold them fixed with respect to the smoothing operations is clearly mandatory in this case. The present sequence of calls from FITIT to NICER, SMOOTH, and DUZ2 do not allow this option, and should be modified in the future.

The FXR-induced current output of the gate, as loaded with the four inputs of the following gate in the dual NAND package, is shown in Figure 40. The oscillations present in the output are very evident at low dose levels. The oscillation frequency is approximately 10 MHz. The behavior of the surface at higher dose levels shows portions of this ringing frequency also. The rather disturbed appearance at high dose levels is the result of fitting a number of data strings from different shots and devices, with not only their own inconsistencies, but also the error components arising from trigger differences, and those of the hand digitization of the Polaroid photographs. The amplitude of the signal from the Tektronix CT-2 current transformers is in the millivolt range so that the maximum gain of the Tektronix 7704 oscilloscope was required in order to observe the traces. Noise and drift of the signal were consequently severe.

The FXR response surfaces for the gate as presented above were plotted vs. time itself as the independent variable along the x axis, rather than the logarithm of time as was the case with the neutron response surfaces and the 741 response surfaces. This posed some problems in the later analysis program SAP, so an attempt was made to fit the gate FXR surfaces similarly with a log (time) representation along the x axis. The results for the current radiation surface from run 498 appear in Figure 41. The effect of the linear to logarithmic time transformation is to devote about one-third of the fitting function to the time interval between 1 and 10 ns after trigger, during which interval very little happens, and to essentially eliminate any of the detail in the ringing after about 70 ns. The resulting fit is certainly not satisfactory. Thus, it is evident that a log(time) axis will be satisfactory only if the fundamental structure of the surface decreases sufficiently rapidly with time. It would appear that a circuit or system whose impulse response function continues with rapid variations for long times, such as an oscillator, should not be fit with a log(time) representation.

FXR CURRENT RADIATION TRANSFER FUNCTION OF 9704

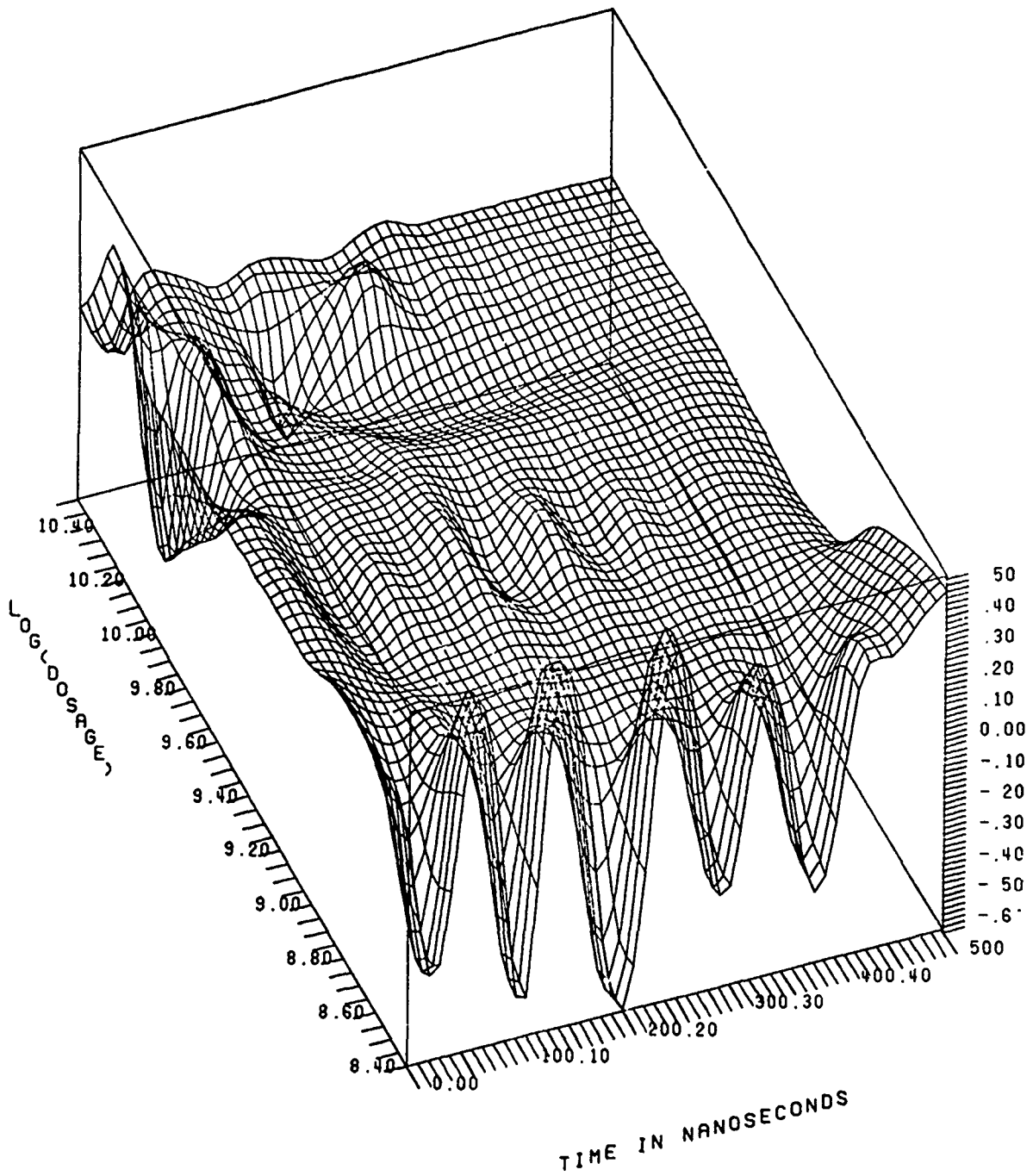


Figure 40. Oblique View of FXR Current Radiation Transfer Function of 9704 with Linear Representation along Time Axis

FXR CURRENT RADIATION TRANSFER FUNCTION OF 9704 RUN 498

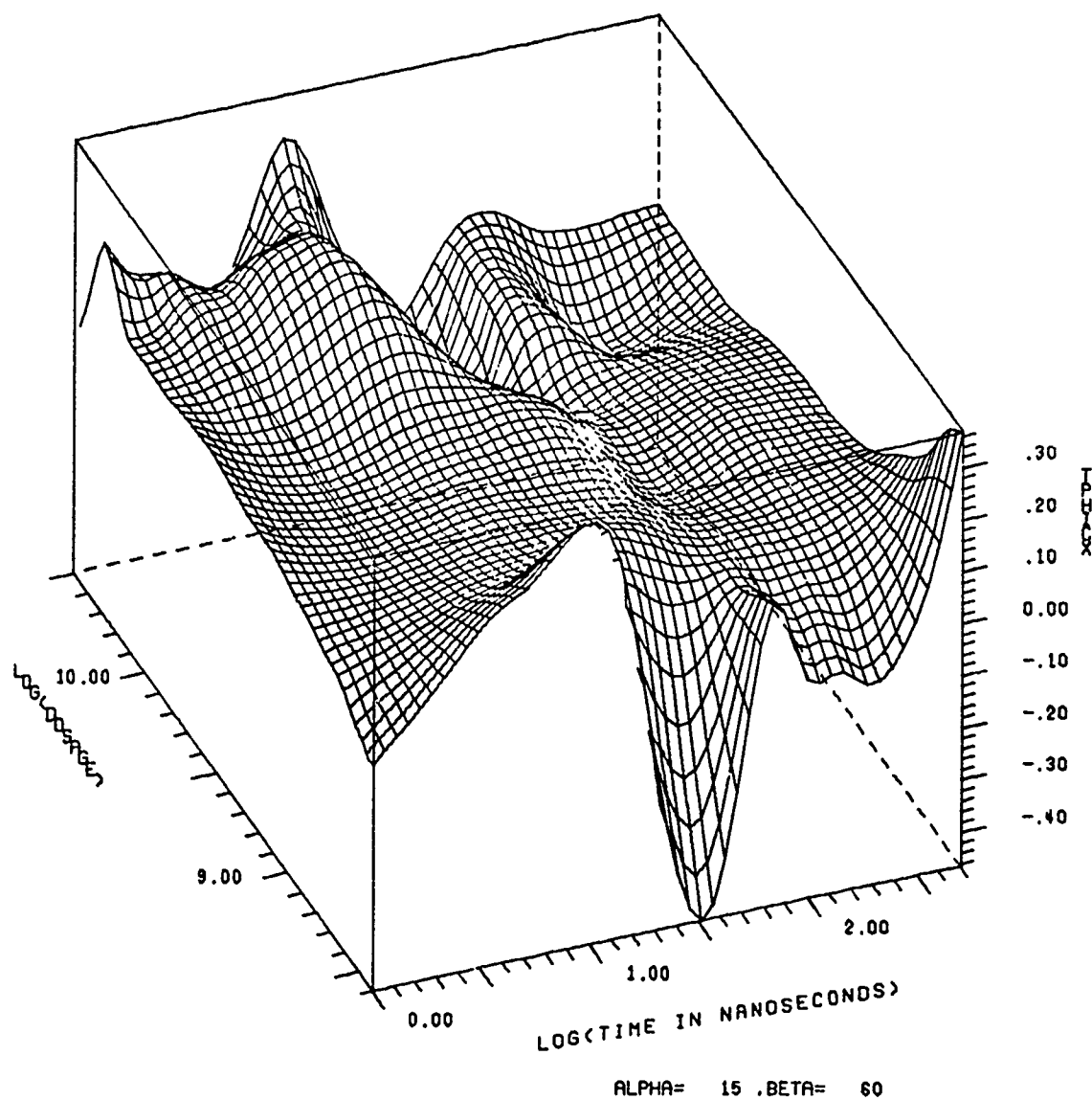


Figure 41. Oblique View of FXR Current Radiation Transfer Function of 9704 with Logarithmic Representation along Time Axis

SECTION V

SAP COMPUTER PROGRAM

1. GENERAL ORGANIZATION

The overall logic flow in the computer program SAP is shown in the flowchart presented in Figure 42. The first seven steps are essentially initialization procedures which bring in the surface ZCOEF files from drum storage; read data cards describing the circuit configuration, radiation forcing function, and voltage waveform; and then the control and title cards for the plots.

The execution of the computational strategy discussed in detail in Section II is initiated at step 8 which is the DO loop driving the time stepping and runs to step 16. The time stepping is a simple constant increment determined by the data card stream. The time base can be either linear or logarithmic. The mode of the time base affects the convolution process, as will be discussed below. Next, the subroutines RADGEN and VOLTIM are called to obtain the radiation and voltage forcing functions at a given time step.

At step 10 in the flow chart, the adjusted input variable is obtained by extrapolating the value at the last time step by the ratio of the new and old fixed input variables. If the input element is close to linear, the new iterate can be reasonably close to the correct value. Then the iteration routine DRIVER is called to initiate the Newton-Raphson iteration cycle. The iterate is returned to MAIN. Step 12 starts a DO loop which runs down through step 14, and acts to propagate the adjusted and fixed input variables through the chain of blocks to the output node.

As the above DO loop drives along the chain of blocks, successive calls to FE and FI are made to obtain the voltage and current transfer functions of the blocks. If the block element is simple, such that an equation suffices for the transfer function, FE or FI will return that result to MAIN. If surface look-up and convolution are required, then FE and FI will drive into CONVOL, TSURF, and SURFC. CONVOL basically handles the convolving of past time steps dynamical responses, and the interrogation of the surface routines for evaluation of the contribution of the current time steps present iterate to the output. Note that the convolution operation is needed only once for each time step, and that the

evaluation of the present iterate's contribution involves only one surface look-up rather than a look-up for each term in the convolution sum. Thus, the execution speed of convolutional dynamical evaluations can be rather high. The routine TSURF handles scaling of function values for the surfaces, and takes appropriate actions if the input variables are out of the range of the surface. SURFB and SURFC use the ZCOEF arrays read in by MAIN to perform surface evaluations by interpolating within the grid squares. The execution time is approximately 2 ms per surface look-up.

The self-consistency loop at step 14 essentially attempts to compensate for a computational delay inherent in the calculation if the voltage transfer function depends on the output current, at a given time step and a given iterate. Since the voltage is calculated first through FE, the first result will be using the current from the last time step. The next statement in MAIN updates the current for the present iterate. The self-consistency loop then goes back and recalculates the voltage for the block with the new current and tests for self-consistency with the values of the last pass. When the voltages and currents agree within some desired value with those of the last pass, the self-consistency criteria are satisfied and control passes to the next step in the program.

At step 15 the routine DRIVER is called thereby providing it with the function value of the fixed output variable. DRIVER then tests for satisfaction of the convergence criteria, and either forms a new iterate and returns to step 12, or is satisfied and continues to step 16. At 16 the results for the time step are printed out, and PLOT6 is called to enter them into the plot arrays. This then completes one time step, and the program returns to the beginning of the DO at step 8.

After completion of the time step DO, at step 17, PLOT6 is again entered to initiate printer and machine plots through PLOT6, PLOTOP, the plot subroutine package, and the UCC machine plot routines. Finally, at step 18 a data card is read which returns to step 3 if it says MORE, or calls PLOT6 and PLOT6 to terminate the plot tape and then calls EXIT if the card says NOMORE.

2. PROGRAM MAIN

Since the overall logic of MAIN was described in the section above relating to the flow chart, here we shall just describe some of the details of the data card decks. The listing of MAIN is given in Figure 43. The surface ZCOEF decks are

W ELT MAIN,1,720302, 47795 , 1

```

000001      C      PROGRAM MAIN
000002      INCLUDE SAP.LIST
000003      INCLUDE TIME.LIST
000004      INCLUDE XYZ.LIST
000005      COMMON / DEBUG / IDEBUG
000006      LOGICAL IDEBUG
000007      DIMENSION NIN(IX),NOUT(IX),LBLOCK(IX,IX),ISURF(NSURF),LTIME(2)
000008      1,JPARAL(IX),FACTOR(IX),CNOUT(IX),VNOUT(IX),CNJN(IX),VNIN(IX)
000009      2,LMEX(IX)
000010      LOGICAL IRADX,IRADN,INITAL,ISYM,ISYML
000011      C      ISYM = .TRUE. IMPLIES SURFACE IS SYMMETRIC WITH RESPECT TO Y = 0.
000012      C      KMAX = MAXIMUM NUMBER OF ITERATIONS.
000013      DATA TINY/1.E-20/, SMALL/1.E-12/, SAP1/4HSAP1/, STAR/4H****/
000014      DATA EPSLN2/.01/,EPSLN3/1.0E-8/,IN/5/,IT/6/
000015      DATA NOMORE/6HNOMORE/, DAMAGX/1.0/, DAMAGN/1.0/
000016
000017      C      IF THE ROUTINE DAMAGE IS CALLED UPON, THE SURFACE ZVN741,
000018      C      MUST BE READ IN BY MAIN. THIS CORRESPONDS TO LSURF = 5.
000019      C
000020      C
000021      C      ZEROING BLANK COMMON SINCE OVERLAID BY THE LOADER.
000022      C      LENGTH = TOTAL LENGTH OF BLANK COMMON.
000023      C
000024      LENGTH = 792
000025      DO 1001 I = 1, LENGTH
000026      1001 VIN(I,1) = 0.0
000027      C
000028      C      CALL EKEY($600,MAXTIM)
000029      C      INITAL = .TRUE.
000030      IXBASE = 0
000031      IYBASE = 0
000032      IZBASE = 0
000033      IDEBUG = .TRUE.
000034      LIMIT0 = 10
000035      VTEST = .05
000036      CTEST = .05
000037      C
000038      WRITE(IT,1) STAR,SAP1,STAR,SAP1,STAR,SAP1,STAR,SAP1,STAR,SAP1,STAR
000039      1 FORMAT(1H1, //,10X,5(A4,4X,A4,4X),A4 //)
000040      READ(IN,2) (ISURF(L),L=1,NSURF)
000041      2 FORMAT(12I5)
000042      WRITE(IT,3) (L,ISURF(L),L=1,NSURF)
000043      3 FORMAT(/ 2X,8(3X, 6HISURF(,I2,4H) = ,I1) /)
000044      C
000045      CALLING INPUT DATA FROM SUBROUTINE DATA.
000046      CALL DATA
000047      C
000048      CALLING SURFACE INFORMATION FROM DRUM FILES.
000049      DO 13 L = 1, NSURF
000050      IF (ISURF(L).EQ.0) GO TO 13
000051      INPUT = L + 17
000052      READ(INPUT,4) XLO,XHI,YLO,YHI,IXDIMN,IYDIMN,LL,ISYML,(SCALE(L,J)
000053      1,J=1,3)
000054      4 FORMAT(4E8.4,3I4,L6,3E10.5)
000055      IXDIM(L) = IXDIMN
000056      IYDIM(L) = IYDIMN
000057      ISYM(L) = ISYML
000058      XLOW(L) = .9999998 * XLO

```

Figure 43a. M A I N Listing

```

000059      YLOW(L) = .9999998 * YLO
000060      XHIGH(L) = 1.0000002 * XHI
000061      YHIGH(L) = 1.0000002 * YHI
000062      XDELTA = (XHIGH(L)-XLOW(L))/(IXDIM(L)-1)
000063      YDELTA = (YHIGH(L)-YLOW(L))/(IYDIM(L)-1)
000064      WRITE(IT,5) L,XLO,XHI,YLO,YHI,XDELTA,YDELTA,IXDIMN,IYDIMN,LL,ISYML
000065      1,(L,J,SCALE(L,J),J=1,3)
000066      5 FORMAT(/13H FOR SURFACE ,I2,7H, XLO =,G11.4,7H, XHI =,G11.4,7H, YL
000067      10 =,G11.4,7H, YHI =,G11.4,10H, XDELTA =,G11.4,10H, YDELTA =,G11.4,
000068      2/9H IXDIM =,I3,9H, IYDIM =,I3,5H, L =,I2,9H, ISYM = ,L1,5X,3(8H
000069      3SCALE(,I1,1H, ,I1,3H) =,G10.2)/)
000070      6 FORMAT(/ 10X,8HIYBASE =,I3,5X,8HIYBASE = ,I3,5X,8HIZBASE =,I5 /)
000071      C
000072      CREATING X AND Y ARRAYS FROM INPUTTED INFORMATION.
000073      C
000074      IPONTR(L,1) = IXBASE + 1
000075      8 FORMAT(/ 5X,3(7HIPONTR(,I2,1H,,I2,3H) =,I5,1H,5X)/)
000076      DO 9 I = 1, IXDIMN
000077      9 X(I + IXBASE) = XLO + (I-1)*XDELTA
000078      IXBASE = IXBASE + IXDIMN
000079      IPONTR(L,2) = IYBASE + 1
000080      DO 10 J = 1, IYDIMN
000081      10 Y(J+IYBASE) = YLO + (J-1)*YDELTA
000082      IYBASE = IYBASE + IYDIMN
000083      IPONTR(L,3) = IZBASE + 1
000084      WRITE(IT, 8) (L,J,IPONTR(L,J),J=1,3)
000085      MINC = IXDIMN * IYDIMN
000086      MINC2 = MINC + MINC
000087      JHI = IYDIMN - 1
000088      DO 12 I = 1, IXDIMN
000089      DO 12 J = 1, JHI, 2
000090      MS1 = I + (J-1)*IXDIMN + IZBASE
000091      MS2 = MS1 + IXDIMN
000092      ML1 = MS1 + MINC2
000093      ML2 = MS2 + MINC2
000094      READ(INPUT,11) (Z(KK),KK=MS1,ML1,MINC), (Z(KKK),KKK=MS2,ML2,MINC)
000095      11 FORMAT(6E13.7)
000096      12 CONTINUE
000097      IZBASE = IZBASE + 3*MINC
000098      13 CONTINUE
000099      WRITE(IT,6) IXBASE,IYBASE,IZBASE
000100      IF(IXBASE.GT.IXLIM.OR.IYBASE.GT.IYLIM.OR.IZBASE.GT.IZLIM) STOP
000101      C
000102      14 CONTINUE
000103      C
000104      CALLING IN INPUT DATA
000105      READ(IN,15) MORE
000106      15 FORMAT(A6)
000107      WRITE(IT,17) MORE
000108      IF(MORE.EQ.NOMORE)CALL PLOTEM(3,VNIN,CNIN,VNOUT,CNOUT,NMAX,INITAL)
000109      PHIXL = EPSLON
000110      PHINL = FPSLON
000111      DAMAGX = 1.0
000112      DAMAGN = 1.0
000113      WRITE(IT,16)
000114      16 FORMAT(43H INPUT THE VALUE OF ISAME, USE FORMAT I3. /)
000115      17 FORMAT(10X,A6)
000116      READ(IN,22) ISAME
000117      WRITE(IT,22) ISAME
000118      IF(ISAME.EQ.1) GO TO 75
000119      WRITE(IT,20)

```

Figure 43b. M A I N Listing (cont.)

```

000119      20 FORMAT(41H INPUT THE VALUE OF JMAX, USE FORMAT I3. /)
000120      READ(IN,22) JMAX
000121      22 FORMAT(I3)
000122      WRITE(IT,24) JMAX
000123      24 FORMAT(37H THE VALUE OF JMAX AS READ IN IS , I3 /)
000124      WRITE(IT,30)
000125      30 FORMAT(71H INPUT THE VALUES IN SEQUENCE OF J, CONFIG,NIN,NOUT,R,C,
000126      1 L. 4I3,3F12.4 /)
000127      DO 32 J = 1, JMAX
000128      READ(IN,34) I, ITABLE(J,1) ,NIN(J),NOUT(J),R(J),C(J),FL(J)
000129      32 WRITE(IT,34) I, ITABLE(J,1) ,NIN(J),NOUT(J),R(J),C(J),FL(J)
000130      CALL SETUP( NIN,NOUT,LBLOCK,JMAX,NMAX,JPARAL,LMEX)
000131      33 CONTINUE
000132      34 FORMAT(4I3, 3F12.4)
000133
C
000134      IF INKALL.NE. 1, PRINTING IS SUPPRESSED, EXCEPT FOR EVERY MMODTH
C
000135      TIME STEP.
000136      WRITE(IT,41)
000137      41 FORMAT(41H INPUT THE VALUE OF INKALL AND MMOD, 2I3. /)
000138      READ(IN,42) INKALL,MMOD
000139      42 FORMAT(2I3)
000140      WRITE(IT,44) INKALL,MMOD
000141      44 FORMAT( 9H INKALL =,I2,10H,   MMOD =,I3   /)
000142
C
000143      READ(IN,46) EPSLON
000144      46 FORMAT(F10.6)
000145      WRITE(IT,48) EPSLON
000146      48 FORMAT(37H THE ITERATION CRITERIA,EPSLON =, G14.8 /)
000147
C
000148      WRITE(IT,54)
000149      54 FORMAT(54H INPUT 1 IF INPUT VOLTAGE FIXED, 2 IF CURRENT, I3 FRMT/)
000150      READ(IN,56) IFIXIN
000151      56 FORMAT(I3)
000152      WRITE(IT,56) IFIXIN
000153      WRITE(IT,58)
000154      58 FORMAT(52H INPUT 1 IF VOUT IS FIXED, 2 IF COUT. USE FORMAT I3./)
000155      READ(IN,60) IFIXOT
000156      60 FORMAT(I3)
000157      WRITE(IT,56) IFIXOT
000158      KROSS = 2 - IABS(IFIXIN - IFIXOT)
000159      WRITE(IT,61) IFIXIN, IFIXOT, KROSS
000160      61 FORMAT(/8H IFIXIN=,I1,9H, IFIXOT=,I1,8H, KROSS=,I1,/)
000161      WRITE(IT,62)
000162      62 FORMAT(53H INPUT THE VALUE OF THE FIXED OUTPUT VARIABLE, F12.4./)
000163      GO TO(68,64), IFIXOT
000164      64 READ(IN,65) COUT(NMAX)
000165      WRITE(IT,65) COUT(NMAX)
000166      65 FORMAT(F12.4)
000167      Y0 = COUT(NMAX)
000168      GO TO 70
000169      68 READ(IN,65) VNOUT(NMAX)
000170      WRITE(IT,65) VNOUT(NMAX)
000171      Y0 = VNOUT(NMAX)
000172      70 CONTINUE
000173      WRITE(IT,72)
000174      72 FORMAT(58H INPUT THE INITIAL VALUES OF VIN(1), CIN(1), USING 2F12.
000175      14 )
000176      READ(IN,74) VNIN(1), CNIN(1)
000177      WRITE(IT,74) VNIN(1),CNIN(1)
000178      74 FORMAT(2F12.6)

```

Figure 43c. M A I N Listing (cont.)

```

000179 C          READ FORCING FUNCTION DATA.
000180 C
000181 C      THE NEXT THREE STATEMENTS CAUSE INPUTTING OF DATA CARDS BY THE
000182 C      ROUTINES RADGEN, VOLTIM, AND PLOTEM. IN ADDITION PLOT6 WILL ALSO
000183 C      BE CALLED BY PLOTEM AND WILL READ ONE DATA CARD.
000184 C      HOWEVER, IF THE RUN IS A SECONDS PASS OF DATA SETS, THIS CARD WILL
000185 C      NOT, REPEAT NOT, BE READ AND SO SHOULD NOT BE PLACED IN THE SECOND
000186 C      AND FOLLOWING DATA SETS.
000187 C
000188 C      75 CALL RADGEN(0,STIMUL,PHI,IRADX,IRADN)
000189 C      DUMMY = VOLTIM(0)
000190 C      IF(ISAME.NE.1) CALL PLOTEM(-1,VNIN,CNIN,VNOUT,CNOUT,NMAX,INITAL)
000191 C      NEXTAK=1
000192 C      GO TO (80,82), IFIXIN
000193 C      80 F1 = VNIN(1)
000194 C      G1 = CNIN(1)
000195 C      GO TO 90
000196 C      82 F1 = CNIN(1)
000197 C      G1 = VNIN(1)
000198 C
000199 C      CONVOCKING THE SYSTEM ELAPSED TIME ROUTINE TO START THE CLOCK
000200 C      90 CALL ELT1
000201 C
000202 C      THE 90 DO LOOP STEPS TIME.
000203 C      DO 500 M = 1, MMAX
000204 C      MTRUN = M/MMOD
000205 C      MODM = MTRUN*MMOD/M
000206 C      GO TO (92,94), IFIXIN
000207 C      92 FOLD = VNIN(1)
000208 C      G = CNIN(1)
000209 C      CALL RADGEN(1,VNIN(1),PHI,IRADX,IRADN)
000210 C      F = VNIN(1)
000211 C      GO TO 96
000212 C      94 FOLD = CNIN(1)
000213 C      G = VNIN(1)
000214 C      CALL RADGEN(1,CNIN(1),PHI,IRADX,IRADN)
000215 C      F = CNIN(1)
000216 C      96 IF(IRADX.AND.PHI(1).GT.0.) PHIXL = ALOG10(PHI(1))
000217 C      IF(IRADN.AND.PHI(2).GT.0.) PHINL = ALOG10(PHI(2))
000218 C      IF(.NOT.(INKALL.EQ.1.OR.MODM.EQ.1)) GO TO 100
000219 C      WRITE(IT,97) M,TIME(MPRIME,1)
000220 C      97 FORMAT(/,6H0 M =,15,10X,6HTIME =,G14.2)
000221 C      WRITE(IT,98)
000222 C      98 FORMAT( 76H K VIN CIN VOUT COUT
000223 C      1 P DPDX )
000224 C
000225 C      F = CURRENT VALUE OF FIXED INPUT VARIABLE
000226 C      G = CURRENT VALUE OF ADJUSTED INPUT VARIABLE.
000227 C      V = CURRENT VALUE OF ADJUSTED OUTPUT VARIABLE.
000228 C      Y0= DESIRED VALUE OF ADJUSTED OUTPUT VARIABLE.
000229 C      P = PRODUCT OF TRANSFER FUNCTIONS OF ADJUSTED OUTPUT VARIABLE.
000230 C      AK = CUPRENT GUESS OF ADJUSTED INPUT VARIABLE.
000231 C      AK1= PREVIOUS GUESS OF ADJUSTED INPUT VARIABLE
000232 C      AK2= PAST PREVIOUS GUESS OF ADJUSTED INPUT VARIABLE
000233 C
000234 C      100 IF(M.GT.1) GO TO 110
000235 C      DO 105 N1 = 1, NMAX
000236 C      IF(JPARAL(N1)) 103, 105, 103
000237 C      103 N2 = LMEX(N1)
000238 C      DO 104 N3 = 1, N2

```

Figure 43d. M A I N Listing (cont.)

```

000239      LB = LRLOCK(N1,N3)
000240      104 FACTOR(LB) = 1.0/N2
000241      105 CONTINUE
000242
000243      C
000244      CALCULATING FIRST GUESS OF ADJUSTED INPUT VARIABLE AS LINEAR EXTRAPOLA-
000245      C TION OF LAST CONVERGED VALUE BY RATIO OF NEW AND OLD FIXED INPUT
000246      C VARIABLE.
000247      110 IF(NEXTAK.NE.0) GO TO 112
000248      NEXTAK = 1
000249      111 AK = F * G1/F1
000250      GO TO 120
000251      112 AK = G
000252      IF (F) 114, 116, 114
000253      114 IF(FOLD) 118, 111, 118
000254      116 AK = 0.0
000255      LPATH = 2
000256      NFXAK = 0
000257      GO TO 160
000258      118 AK = G * F/FOLD
000259      120 IF(M.EQ.1.OR.LPATH.EQ.3) GO TO 140
000260      A = 0.50 * AK
000261      B = 1.50 * AK
000262      GO TO 145
000263      140 A = AK/EPSLON
000264      B = AK * EPSLON
000265      145 BNDLO = AMIN1(A,B)
000266      BNDHI = AMAX1(A,B)
000267      ITERAT = 0
000268
000269      C
000270      150 CALL DRIVER (AK,V,BNDLO,BNDHI,INKALL,ITERAT,LPATH,Y0,F,EPSLON)
000271      C 150 CALL NEWTON (AK,V,BNDLO,BNDHI,INKALL,ITERAT,LPATH,Y0,F,EPSLON)
000272      160 GO TO(170,180), IFIXIN
000273      170 CNIN(1)=AK
000274      GO TO 185
000275      180 VNIN(1) = AK
000276      185 GO TO(200,190,190), LPATH
000277      190 DO 192 J = 1, JMAX
000278      VOUT(J,2) = VOUT(J,1)
000279      192 COUT(J,2) = COUT(J,1)
000280
000281      C
000282      C THE 200 DO LOOP CORRESPONDS TO INPUT NODES.
000283      C
000284      200 CONTINUE
000285      DO 300 N = 1, NMAX
000286      IF(JPARAL(N)) 235,206,235
000287      206 LN = LMEX(N)
000288      J = LRLOCK(N,LN)
000289      IF(ABS(VNIN(N)).LT.TINY) VNIN(N) = SMALL
000290      IF(ABS(CNIN(N)).LT.TINY) CNIN(N) = SMALL
000291      VIN(J,1) = VNIN(N)
000292      CIN(J,1) = CNIN(N)
000293      ITRSAV = ITERAT
000294      NITER = 0
000295      207 NITER = NITER + 1
000296      ITERAT = ITRSAV * NITER
000297      209 VOLD = VOUT(J,1)
000298      COLD = COUT(J,1)
000299      IF(ABS(VOLD).LT.TINY) VOLD = SMALL
000300      IF(ABS(COLD).LT.TINY) COLD = SMALL
000301      GO TO (210,220), KROSS

```

Figure 43e. M A I N Listing (cont.)

```

000299      210 VOUT(J,1) = VIN(J,1) * FE(J)
000300      COUT(J,1) = CIN(J,1) * FI(J)
000301      GO TO 224
000302      220 VOUT(J,1) = CIN(J,1) * FE(J)
000303      COUT(J,1) = VIN(J,1) * FI(J)
000304      224 VTST = ABS ( 1.0 - VOUT(J,1)/VOLD )
000305      CTST = ABS ( 1.0 - COUT(J,1)/COLD )
000306      IF(.NOT.(ABS(VOLD).GT.0.)) VTST = 0.
000307      IF(.NOT.(ABS(COLD).GT.0.)) CTST = 0.
000308      226 IF ((VTST .GT. VTEST .OR. CTST .GT. CTEST) .AND. NITER.LE.LIMITO
000309      1) GO TO 207
000310      ITERAT = ITRSAV
000311      230 IF ( ITERAT .GT. 1 ) GO TO 2301
000312      IF(IRADX) VOUT(J,3) = PHIXL * FEPHI(J,1)
000313      IF(IRADN) VOUT(J,3) = PHINL * FEPHI(J,2)
000314      IF(IRADX) COUT(J,3) = PHIXL * FIPHI(J,1)
000315      IF(IRADN) COUT(J,3) = PHINL * FIPHI(J,2)
000316      IF(IRADX) DAMAGX = DAMAGE(J,1)
000317      IF(IRADN) DAMAGN = DAMAGE(J,2)
000318      DAMAGT = DAMAGX * DAMAGN
000319      2301 IF(LPATH.EQ.1) GO TO 231
000320      IF(.NOT.(IRADX.OR.IRADN)) GO TO 231
000321      VOUT(J,1) = VOUT(J,1)*DAMAGT + VOUT(J,3)
000322      COUT(J,1) = COUT(J,1)*DAMAGT + COUT(J,3)
000323      231 IF(JPARAL(N)) 232,232,250
000324      232 VNOUT(N) = VOUT(J,1)
000325      CNOUT(N) = COUT(J,1)
000326      GO TO 250
000327      235 CALL PARLEL(N,LBLOCK,LMEX,FACTOR,CNOUT,VNOUT,VNIN,CNIN,
000328      I INKALL,JPARAL)
000329      GO TO 230
000330      250 VNIN(N+1) = VNOUT(N)
000331      CNIN(N+1) = CNOUT(N)
000332      300 CONTINUE
000333      C
000334      GO TO (301,440,440), LPATH
000335      301 GO TO (302,304), KROSS
000336      302 FEPROD = VNOUT(NMAX)/VNIN(1)
000337      FIPROD = CNOUT(NMAX)/CNIN(1)
000338      GO TO 308
000339      304 FEPROD = VNOUT(NMAX)/CNIN(1)
000340      FIPROD = CNOUT(NMAX)/VNIN(1)
000341      C
000342      308 GO TO (310,314), IFIXOT
000343      310 P = FEPROD
000344      V = VNOUT(NMAX)
000345      GO TO 320
000346      314 P = FIPROD
000347      V = CNOUT(NMAX)
000348      320 IF(ITERAT-2) 322,324,324
000349      322 DPDX = 0.0
000350      AK1 = AK
000351      GO TO 328
000352      324 DP = P - POLD
000353      DX = AK - AK1
000354      IF(ABS(DX).LT.TINY) DX = SMALL
000355      DPDX = DP/DX
000356      328 IF(.NOT.(INKALL.EQ.1.OR.MODM.EQ.1)) GO TO 342
000357      330 WRITE(17,340)ITERAT,VNIN(1),CNIN(1),VNOUT(NMAX),CNOUT(NMAX),P,DPDX
000358      340 FORMAT(1X,I3,1X,G11.4,1X,G13.6,1X,G11.4,1X,G12.5,1X,G10.3,1X,G11.4

```

Figure 43f. M A I N Listing (cont.)

```

000359      1)
000360      342 AK2 = AK1
000361          AK1 = AK
000362          POLD = P
000363      COMPLETION OF FUNCTION EVALUATION FOR ITERATION DRIVER.
000364          GO TO 150
000365      C
000366      440 IF (.NOT. (INKALL.EQ.1.OR.MODM.EQ.1)) GO TO 450
000367          WRITE(IT,441)
000368      441 FORMAT(// 128H J   VIN(J)      CIN(J)      VOUT(J,1)   COUT(J,1)
000369          1)   VDIFF      CDIFF      VOUT(J,2)   COUT(J,2)   VOUT(J,3)   C
000370          2OUT(J,3)   )
000371      444 DO 445 J = 1, JMAX
000372          VDIFF = VIN(J,1)-VOUT(J,1)
000373          CDIFF = CIN(J,1)-COUT(J,1)
000374      445 WRITE(IT,446)J,VIN(J,1),CIN(J,1),VOUT(J,1),COUT(J,1),VDIFF,CDIFF
000375          1,VOUT(J,2),COUT(J,2),VOUT(J,3),COUT(J,3)
000376      446 FORMAT(I3,10(1X,611.3))
000377      450 DO 480 J = 1, JMAX
000378          DO 460 JIG = 2, JT
000379              JOG = JT - JIG + 2
000380              VIN(J,JOG) = VIN(J,JOG-1)
000381              CIN(J,JOG) = CIN(J,JOG-1)
000382              VINTGL(J,JOG) = VINTGL(J,JOG-1)
000383              CINTGL(J,JOG) = CINTGL(J,JOG-1)
000384      460 CONTINUE
000385          DO 470 JIG = 2, JD
000386              JOG = JD - JIG + 2
000387              VINDT1(J,JOG) = VINDT1(J,JOG-1)
000388              CINDT1(J,JOG) = CINDT1(J,JOG-1)
000389      470 CONTINUE
000390      480 CONTINUE
000391          CALL PLOTEM(0,VNIN,CNIN,VNOUT,CIN,IT,NMAX,INITAL)
000392      500 CONTINUE
000393      600 CALL ELT3(LTIME)
000394          WRITE(IT,610) LTIME
000395      610 FORMAT(// 33H THE ELAPSED TIME REQUIRED WAS....2A6. //)
000396          CALL PLOTEM(1,VNIN,CNIN,VNOUT,CNOUT,NMAX,INITAL)
000397          CALL PLOTEM(2,VNIN,CNIN,VNOUT,CNOUT,NMAX,INITAL)
000398          CALL RADGEN(2,STIMUL,FHI,IRADX,IRADN)
000399          GO TO 14
000400      END

```

Figure 43g. M A I N Listing (cont.)

brought in from drum files using a sequence of read statements which essentially treats the arrays as linear rather than triply dimensioned so that the different surfaces can have different dimensions if desired. Pointers are calculated and entered into the XYZ common block for use later by TSURF and SURFB. The first data card has the surface number; the array dimensions; the x, y, and z scaling factors; and the symmetry type, i.e. whether even or odd. Then follow the $(z, \partial z / \partial x, \partial z / \partial y)$ cards totaling NX times NY (or 200) in number.

Only those surface arrays are read from the drum files which are to be used in a given simulation. The specific arrays are designated by the first read, Format number 2. If a specific surface is needed in the computation and has not been read in, then computed GO TO's may go out of range during execution, and correct pointers will not exist for SURFB.

The data read statements that extend from the statement 14 to statement 74 bring in the information on the circuit needed for a given run. ISAME is a rerun variable for repeating the same circuit; zero means false or a new circuit, one means true or a rerun of the same circuit. JMAX is the number of blocks in the circuit. The read at Format 34 calls in J - the block sequence number ITABLE(J,1) indicates the circuit element as described by the comment cards in FE and FI. NIN(J) and NOUT(J) are the input and output node numbers for a block, while R(J), C(J), and FL(J) represent resistance, capacitance, or inductance values.

The following statement calls SETUP which ascertains the circuit configuration from the node numbers, and tags these blocks in parallel branches for treatment later during propagation by the parallel algorithm in PARLEL.

The following READ statements are self-explanatory as indicated by the comments in the WRITE FORMATS. IFIXIN and IFIXOT indicate to the program whether the voltage or current is the fixed input and output variables. From these the variable KROSS is calculated, which is used in many subroutines in computed GO TOs for switching from the crossed to uncrossed propagation strategy. The normal value for IFIXIN is one and that for IFIXOT is two. The usual value for CNOUT(NMAX) is zero. VNIN(1) and CNIN(1) are used on the first time step to initiate the iteration process. The remainder of the data card stream is read in by calls to RADGEN, VOLTIM, and PLOTEN.

3. SUBROUTINE RADGEN (N, STIMUL, PHI, IRADX, IRADN)

RADGEN creates the radiation stimulus applied to the device during the simulation. The listing of RADGEN is contained in Figure 44. Four classes of forcing functions can be used - constant, Gaussian, Weibull, or piecewise linear. Both neutron and X-ray stimulus functions can be generated simultaneously. Since the radiation response surfaces really were created by plotting response characteristics from experimental radiation facilities whose instantaneous dose varies during the radiation pulse, an "isodosal" line actually ranges in dose in the same manner as did the experimental pulse. In other words, the basic data were not deconvolved to true instantaneous dose.

The significance of this representation then is that a constant stimulus function will then represent a simulation of the radiation pulse from the accelerator or reactor which generated the basic data set itself. So a constant stimulus then moves back along an isodosal of the surface, and the resultant calculation is portrayal of the experimental oscilloscope waveform whose peak dosage value corresponds to the isodosal being sampled.

A still more important aspect of this representation is that this type of simulation does not require convolution of the radiation stimulus function in time since only one partition element need move back along the surface. If a different stimulus function is used, a family of partitions is created, one at each time step, and convolving of running sums at each time step is required. These partitions range over a dosage interval on the surface, sampling the structure over a large section of the surface. Hence, the result of the computation is an approximation to that which would be obtained if the original data set were deconvolved initially. In the next subsection we will show two calculations for the same dosage level with and without convolution. The results are surprisingly close to each other in general structure. The main result from the viewpoint of a computer oriented radiation simulation strategy is that simulation of the radiation facility type of exposure reduces the surface look-ups required by a factor of 25 to 50, and hence is fast in execution on the computer.

The above discussion provides a basis for understanding the logical variable KONVOL used in RADGEN and passed through the TIMEX COMMON block for use by the convolution subroutine CONVOL. The COMMON block listings are given in Figure 45.

ELT RADGEN,1,720302, 47814 , 1

```

000001      SUBROUTINE RADGEN (N,STIMUL,PHI,IRADX,IRADN)
000002      INCLUDE TIME,LIST
000003      DIMENSION RH01(25),RH02(25),T1(25),T2(25),PHI(2),XPHI(100,2),
000004      IYPHI(100,2)
000005      DIMENSION IYPE1(2),IYPE2(2),NAME1(2),NAME2(2),IWEIBL(2),
000006      IGAUSS(2),IPIECE(2),IXRAY(2),NEUTRN(2),NOPT(2)
000007      DIMENSION LABEL7(48),ILABEL(6,8),IPOINT(7),TITLE(12),
000008      IORIGN(7),SCALE(7),LAWG(7),IPAR(2),IPOYNT(7)
000009      LOGICAL IRADX1,IRADN1,IRADX2,IRADN2,IRADX,IRADN,KONVOL
000010      DATA IN,11/5,6/
000011      DATA IWEIBL(1)/6HWEIBUL/,IWEIBL(2)/6HL      /,IGAUSS(1)/6HGAUSSI/,
000012      1      IGAUSS(2)/6HAN      /,IXRAY(1)/6HXRAY      /,IXRAY(2)/6H      /,
000013      2      NEUTRN(1)/6HNEUTRO/,NEUTRN(2)/6HN      /,IPIECE(1)/6HPIECEW/,
000014      3      IPIECE(2)/6HISE      /,IBLANK/6H      /
000015      DATA IFXR/6HFXR      /,ISPR/6HSPR      /
000016      DATA IPOYNT(1)/1H1/,IPOYNT(2)/1H2/,IPOYNT(3)/1H3/,IPOYNT(4)/1H4/,
000017      1      IPOYNT(5)/1H5/,IPOYNT(6)/1H6/,IPOYNT(7)/1HX/
000018      C THE DISTRIBUTION FORCING FUNCTIONS ARE CALCULATED BY USING THE
000019      C FOLLOWING STATEMENT FUNCTIONS.
000020      PWEIBL(TYME,ALPHA,BETA,GAMMA) = (BETA * ALPHA ** (-BETA)*(TYME -
000021      1      GAMMA) ** (BETA-1))** (((TYME-GAMMA)/ALPHA) ** BETA)
000022      PGAUSS(TYME,A,TAU,SIGMA)=(A/(SIGMA * (2*3.14159) ** 0.5))**
000023      1      (( TYME - TAU) ** 2 / (2 * SIGMA ** 2))
000024      PICWIS (RI,RIP,TI,TIP1,PHIMAX,TYME) = ((RIP1 - RI) * (TYME - TI)
000025      1      / (( TIP1 - TI ) + RI) * PHIMAX
000026      C
000027      C DATA ARE READ WHEN N = 0 .
000028      C
000029      IF (N - 1) 5, 100, 200
000030      C INITIALIZATION
000031      5 IRADX1 = .FALSE.
000032      IRADN1 = .FALSE.
000033      IRADX2 = .FALSE.
000034      IRADN2 = .FALSE.
000035      IRADX = .FALSE.
000036      IRADN = .FALSE.
000037      KONVOL(1) = .TRUE.
000038      KONVOL(2) = .TRUE.
000039      RADTIM = 0.0
000040      ILO = 1
000041      KOUNT1 = 0
000042      KOUNT2 = 0
000043      NUMBER = 0
000044      C
000045      READ(IN,10) IYPE1(1),IYPE1(2),NAME1(1),NAME1(2),B1,B2,B3
000046      10 FORMAT (4A6,6X,3F10.5)
000047      IF ( NAME1(1).EQ. IFXR .OR.NAME1(1).EQ.ISPR) RADTIM = B2
000048      IF (RADTIM .GT. 0.0) RADTLG = ALOG10(RADTIM)
000049      IF(IYPE1(1).NE.IBLANK.AND.NAME1(1).NE.IFXR.AND.NAME1(1).NE.ISPR)
000050      1 NUMBER = NUMBER + 1
000051      IF (NAME1(1) .NE. IPIECE(1)) GO TO 35
000052      NSTEPS = B1
000053      WRITE (IT,15) NSTEPS
000054      15 FORMAT (9HNSFPS = ,I3)
000055      DO 25 I = 1, NSTEPS
000056      READ(IN,20) RH01(I),T1(I)
000057      20 FORMAT (2F10.5)
000058      25 WRITE(IT,30) I,RH01(I),I,T1(I)

```

Figure 44a. R A D G E N Listing

```

000059      30 FORMAT( 6HRHO1( ,I3,5H ) = ,G12.5,5X,4HT1( ,I3,5H ) = ,G12.5)
000060      NSTEP1 = NSTEPS - 1
000061      35 READ(IN,10) ITYPE2(1),ITYPE2(2),NAME2(1),NAME2(2),C1,C2,C3
000062      IF ( NAME2(1).EQ. IFXR .OR.NAME2(1).EQ.ISPR) RADTIM = C2
000063      IF(ITYPE2(1).NE.IBLANK.AND.ITYPE1(1).NE.ITYPE2(1).AND.NAME2(1)
000064      1.NE.IFXR.AND.NAME2(1).NE.ISPR) NUMBER = NUMBER + 1
000065      IF(NAME2(1) .NE. IPIECE(1)) GO TO 60
000066      NSTEPS = C1
000067      WRITE (IT,15) NSTEPS
000068      DO 40 I = 1, NSTEPS
000069      READ(IN,20) RHO2(I), T2(I)
000070      40 WRITE(IT,30) I, RHO2(I), 1, T2(I)
000071      NSTEP2 = NSTEPS - 1
000072      60 WRITE(IT,45) ITYPE1(1),ITYPE1(2),NAME1(1),NAME1(2),B1,B2,B3
000073      WRITE(IT,65) ITYPE2(1),ITYPE2(2),NAME2(1),NAME2(2),C1,C2,C3
000074      65 FORMAT(18H RADIATION TYPE = , 2A6, 5X, 15HDISTRIBUTION =
000075      1 2A6 / 5X
000076      2 32HTHE 3 RESPECTIVE PARAMETERS ARE ,3(4X,G12.5))
000077      C
000078      IF(NUMBER - 1) 90, 660, 660
000079      660 READ(IN,670) (TITLE(J), J=1,12)
000080      670 FORMAT(8X,12A6)
000081      READ(IN,675) SIZE, (LABLE7(J),J=1,48), IORIGN(7), SKALE(7), NUMBER,
000082      1 LAWG(7)
000083      675 FORMAT(A5,48A1,I3,3X,E12.5,I3,3X,I3)
000084      DO 688 I = 1, NUMBER
000085      688 READ(IN,680) IPOINT(I), (ILABEL(I,J),J=1,8), IORIGN(I), SKALE(I),
000086      1 LAWG(I)
000087      680 FORMAT(A1,4X,8A6,I3,3X,E12.5,6X,I3)
000088      C
000089      90 IF (ITYPE1(1) .EQ. IXRAY(1)) IRADX1 = .TRUE.
000090      IF (ITYPE1(1) .EQ. NEUTRN(1)) IRADN1 = .TRUE.
000091      IF (ITYPE2(1) .EQ. IXRAY(1)) IRADX2 = .TRUE.
000092      IF (ITYPE2(1) .EQ. NEUTRN(1)) IRADN2 = .TRUE.
000093      IF (IRADX1 . OR. IRADX2) IRADX = .TRUE.
000094      IF (IRADN1 . OR. IRADN2) IRADN = .TRUE.
000095      C
000096      C THE MEANING OF THE SUBSCRIPTS ON PHI AND KONVOL ARE AS FOLLOW
000097      C 1 = X-RAY IRRADIATION
000098      C 2 = NEUTRON IRRADIATION
000099      C
000100      IF ((NAME1(1) .EQ. IFXR .AND. ITYPE2(1) .EQ. IXRAY(1)) .OR.
000101      1 (NAME2(1) .EQ. IFXR .AND. ITYPE1(1) .EQ. IXRAY(1)) .OR.
000102      2 (NAME1(1) .EQ. ISPR .AND. ITYPE2(1) .EQ. NEUTRN(1)) .OR.
000103      3 (NAME2(1) .EQ. ISPR .AND. ITYPE1(1) .EQ. NEUTRN(1)))GO TO 900
000104      IF(NAME1(1).EQ.IFXR.OR.NAME2(1).EQ.IFXR) KONVOL(1) = .FALSE.
000105      IF(NAME1(1).EQ.ISPR.OR.NAME2(1).EQ.ISPR) KONVOL(2) = .FALSE.
000106      RETURN
000107      C WHEN N = 1, DIFFERENT FORCING FUNCTIONS ARE CALCULATED AND RETURN
000108      C THE RESPECTIVE RESULTS TO THE CALLING PROGRAM.
000109      100 NN = N
000110      STIMUL = VOLTIM(NN)
000111      TIM = TIME(MPRIME,1)
000112      K=1
000113      IF (IRADX1) GO TO 110
000114      IF (.NOT.IRADN1) GO TO 150
000115      K=2
000116      110 IF (NAME1(1) .EQ. IWEIBL(1)) PHI(K) = PWEIBL(TIM ,B1,B2,B3)
000117      IF (NAME1(1) .EQ. IGAUSS(1)) PHI(K) = PGAUSS(TIM ,B1,B2,B3)
000118      IF (NAME1(1).EQ.IFXR.OR.NAME1(1).EQ.ISPR) PHI(K) = B1

```

Figure 44b. R A D G E N Listing (cont.)

```

000119      IF (NAME2(1) .EQ. IWEIBL(1)) PHI(L) = PWEIBL(TIM ,C1,C2,C3)
000120      1 + DELTA * PHI(K)
000121      IF (NAME2(1) .EQ. IGAUSS(1)) PHI(L) = PGAUSS(TIM ,C1,C2,C3)
000122      1 + DELTA * PHI(K)
000123      IF (NAME2(1) .EQ. IFXR .OR. NAME2(1) .EQ. ISPR) PHI(L) = C1
000124      IF (NAME2(1) .NE. IPIECE(1)) GO TO 200
000125      DO 180 I = ILO, NSTEP2
000126      IF ((TIM .GT. T2(I) .OR. .NOT. TIM .LT. T2(I)) .AND. TIM .LT. T2(I+1)) GO TO 190
000127      180 CONTINUE
000128      I = I - 1
000129      190 ILO = I
000130      IP1 = I + 1
000131      PHI(L) = PICWIS ( RH02(I), RH02(IP1), T2(I), T2(IP1), C2, TIM)
000132      1 + DELTA * PHI(K)
000133      200 RETURN
000134      900 WRITE (IT,910)
000135      910 FORMAT(55H  TERMINATED BY RADGN2 DUE TO INCONSISTENT DATA CARDS.)
000136      STOP
000137      END

```

Figure 44c. R A D G E N listing (cont.)

* ELT COMMON,7,720302, 47791 , 4

```

000001      SAP* FCOPY
000002      C
000003      PARAMETER IX=10,JT=3,JD=3,IS=8
000004      COMMON/SAP/VIN(IX,JT),CIN(IX,JT),VINDT1(IX,JD),CINDT1(IX,JD),R(IX)
000005      1 ,FL(IX),C(IX),ITABLE(16,2),TKQ,CDIND,RSDIOD,RPDIOD,EPSON,S
000006      2 ,JEFLAG(IX),JIFLAG(IX),KROSS,VOUT(IX,3),COUT(IX,3)
000007      3 ,VINTGL(IX,JT),CINTGL(IX,JT),PHI(2),KSURF(16,IS),PHIXL
000008      4 ,PHINL,ITERAT
000009      C
000010      END
000011      TIME* FCOPY
000012      C
000013      PARAMETER LMAX = 100
000014      COMMON/TIME/MAX,M,MPRIME,DT,DLOGT,RADTIM,TRAD,TRADLG,KONVOL(2)
000015      1,TIME(LMAX,2),DTL(LMAX,2),TL(LMAX,2),MODULO,MODE
000016      C
000017      END
000018      XYZ* FCOPY
000019      C
000020      PARAMETER IXLIM=100,IYLIM=100,IZLIM=6006,NSURF=12
000021      COMMON/XYZ/X(IXLIM),Y(IYLIM),Z(IZLIM),IPONTR(NSURF,3),ISYM(NSURF)
000022      1,XLOW(NSURF),XHIGH(NSURF),YLOW(NSURF),YHIGH(NSURF),IXDIM(NSURF)
000023      2,IYDIM(NSURF),SCALE(NSURF,3)
000024      C
000025      END

```

PROCEDURE NAME TABLE

SAP	00000000	TIME	00000214	XYZ	00000356
-----	----------	------	----------	-----	----------

Figure 45. C O M M O N Listing

If the radiation data cards read in by RADGEN indicate that a SPR or FXR simulation is to be made, then KONVOL is set .FALSE. for the appropriate neutron or X-ray term. KONVOL is dimensioned two, with the first element denoting the X-ray term, and the second element denoting the neutron term.

The entries in the call list of RADGEN have the following meanings: N = 0 is an initialization call and causes data cards to be read, N = 1 is the normal entry on subsequent calls. STIMUL is the value of the voltage stimulation function obtained by a call to VOLTIM. PHI is an array of dimension two which returns the radiation forcing function values to MAIN. PHI(1) is the X-ray and PHI(2) the neutron dosage values. The logical variables IRADX and IRADN are set .TRUE. if a particular run requires X-ray or neutron simulation, and are passed back to MAIN through the call.

The actual READ statements of RADGEN, using FORMAT 10, as shown in Figure 44, read in the control variables ITYPE and NAME which denote type of radiation, X-ray or neutron, and the name of the distribution function by which it is to be represented as given above. IFXR or ISPR is the NAME applied for those simulations. The constants B1, B2, and B3 are used to bring in the peak dose level, radiation distribution time, location parameter, and the distribution width parameter.

If a piecewise linear radiation function is called for, then pairs of (dosage, time) points are read in. If the radiation dose level is generated by the distribution function or the piecewise representation, plot title and control information is read in by statements 660 through 680. This will cause PLOT6 to make printer and machine plots of the radiation forcing function.

4. FUNCTION SUBPROGRAM VOLTIM

Data cards are read in by VOLTIM as indicated by the listing of Figure 46. These permit a sinusoidal or piecewise linear voltage forcing function to be created by VOLTIM. In addition one data card is read in at statement 40 which reads the start and stop times, the radiation start time, the number of time steps, the option setting and the mode of the time base - that is whether linear or logarithmic.

An important function performed by VOLTIM is to load the TIME and DTL arrays which are used in the convolution calculations by KONVOL. The first columns in these arrays contain the values of time and differential time dt in a linear

ELT VOLTIM,1,720302, 50484 , 1

```

000001      FUNCTION VOLTIM(N)
000002      INCLUDE TIME,LIST
000003      DIMENSION V(12), T(12)
000004      DIMENSION TUNIT(3), TSCALE(3)
000005      INTEGER OPTION,OPTLIN,OPTLOG,TUNIT,UNIT
000006      DATA IN,IT/5,6/
000007      DATA NANSEC/6HNANO 5/,MICSEC/6HMICROS/,MILSEC/6HMILLIS/
000008      DATA OPTLOG/6HLOG  /,OPTLIN/6HLINEAR/
000009
000010      C      IF (N.NE.0) GO TO 50
000011      ILO = 1
000012      MODULO = 0
000013      OLDTIM = 0.0
000014      OLDLOG = 0.0
000015      TRADLG = 0.0
000016      READ (IN,10)IOPT
000017      10 FORMAT(I3)
000018      WRITE (IT,17)IOPT
000019      17 FORMAT(26H0FORCING FUNCTION OPTION =,I3)
000020      GO TO (18,30), IOPT
000021      18 READ (IN,12)A,FREQ
000022      12 FORMAT(2(E15.8,5X))
000023      WRITE (IT,19)A,FREQ
000024      19 FORMAT(34H0A=E15.8,5X,6H FREQ=,E15.8)
000025      OMEGA = 2.0*3.141592654*FREQ
000026      GO TO 40
000027      30 READ(IN,10) NSTEPS
000028      WRITE(IT,31) NSTEPS
000029      31 FORMAT(/69H A PIECEWISE LINEAR FORCE FUNCTION IS USED WITH THE NU
000030      MBER OF STEPS = ,I3,/ 31H FUNCTION      TIME      UNIT )
000031      DO 36 I = 1, NSTEPS
000032      READ(IN,34) V(I),T(I), UNIT
000033      WRITE(IT,35) V(I),T(I), UNIT
000034      34 FORMAT(2E12.8,A6)
000035      35 FORMAT(2X,2G12.4,A6)
000036      IF(UNIT.EQ.NANSEC) TSC = 1.0
000037      IF(UNIT.EQ.MICSEC) TSC = 1.0E+3
000038      IF(UNIT.EQ.MILSEC) TSC = 1.0E+6
000039      IF(UNIT.NE.NANSEC.AND.UNIT.NE.MICSEC.AND.UNIT.NE.MILSEC)
000040      1 WRITE(IT,38)
000041      36 T(I) = T(I) * TSC
000042      38 FORMAT(5X, 50H THE UNIT MULTIPLIER IS NOT CORRECT FOR THE TIME. )
000043      NSTEP1 = NSTEPS - 1
000044      40 READ(IN,41) ISTART,TUNIT(1), ISTOP, TUNIT(2), IRADTM, TUNIT(3),
000045      IMMAX, OPTION
000046      41 FORMAT(5X,3(I6,1X,A6,2X),5X,I5,5X,A6)
000047      WRITE(IT,41)ISTART,TUNIT(1), ISTOP, TUNIT(2), IRADTM, TUNIT(3),
000048      IMMAX, OPTION
000049      DO 42 K = 1,3
000050      IF (TUNIT(K) .EQ. NANSEC) TSCALE(K) = 1.0
000051      IF (TUNIT(K) .EQ. MICSEC) TSCALE(K) = 1.0E+3
000052      IF (TUNIT(K) .EQ. MILSEC) TSCALE(K) = 1.0E+6
000053      IF(TUNIT(K).NE.NANSEC.AND.TUNIT(K).NE.MICSEC.AND.TUNIT(K).NE.MILSE
000054      1C) WRITE(IT,38)
000055      42 CONTINUE
000056      TSTART = ISTART * TSCALE(1)
000057      TSTART = AMAX1(TSTART,1.)
000058      TSTOP = ISTOP * TSCALE(2)

```

Figure 46a. V O L T I M Listing

```

000059      RADTIM = IRADTM * TSCALE(3)
000060      IF (OPTION .EQ. OPTLIN) MODE = 1
000061      IF (OPTION .EQ. OPTLOG) MODE = 2
000062      WRITE (IT,46) OPTION
000063      IF (OPTION.NE.OPTLIN.AND.OPTION.NE.OPTLOG) STOP
000064 46  FORMAT(10X,A6,21H TIME STEPS ARE USED. / )
000065      GO TO (47,48), MODE
000066 47  A1 = TSTART
000067      A2 = (TSTOP - TSTART) / (MMAX - 1)
000068      RETURN
000069 48  A1 = ALOG10(TSTART)
000070      A2 = (ALOG10(TSTOP/TSTART))/(MMAX - 1)
000071      RETURN
000072  C
000073  C  ENTRY POINT ON ALL BUT FIRST CARD READ OPERATION
000074  C  MODE = 1 CORRESPONDS TO A LINEAR TIME BASE.
000075  C  MODE = 2 CORRESPONDS TO LOGARITHMIC TIME BASE.
000076  C
000077      50  MPRIME = M - MODULO * LMAX
000078      IF (MPRIME.EQ.LMAX) MODULO = MODULO + 1
000079      GO TO (52,54), MODE
000080 52  TIME(MPRIME,1) = AMAX1((A1+(M-1)*A2),1.)
000081      TIME(MPRIME,2) = ALOG10(TIME(MPRIME,1))
000082      GO TO 55
000083 54  TIME(MPRIME,2) = A1 + (M-1) * A2
000084      TIME(MPRIME,1) = 10.0**TIME(MPRIME,2)
000085 55  IF (M-1) 1055, 1055, 4055
000086 1055 GO TO(2055,3055), MODE
000087 2055 D1 = A2
000088      DLOGT = ALOG10(DT)
000089      GO TO 5055
000090 3055 DLOGT = A2
000091      DT = 10.0**DLOGT
000092      GO TO 5055
000093 4055 DT = TIME(MPRIME,1) - OLDTIM
000094      DLOGT = TIME(MPRIME,2) - OLDLOG
000095 5055 OLDTIM = TIME(MPRIME,1)
000096      OLDLOG = TIME(MPRIME,2)
000097      DTL(MPRIME,1) = DT
000098      DTL(MPRIME,2) = DLOGT
000099      TRAD = TIME(MPRIME,1) - RADTIM
000100      TRADLG = ALOG10(AMAX1(TRAD,1.))
000101      IHI = MPRIME
000102      IF (MODULO.GT.0) IHI = LMAX
000103      DO 56 I = 1, IHI
000104          TL(I,1) = TIME(MPRIME,1) - TIME(I,1)
000105 56  TL(I,2) = TIME(MPRIME,2) - TIME(I,2)
000106  C
000107 58  GO TO(59,60), IOPT
000108 59  VOLTIM = A * SIN(OMEGA*TIME(MPRIME,1))
000109      RETURN
000110 60  DO 70 I = ILO, NSTEP1
000111      IF ((TIME(MPRIME,1).GT.T(I)).OR..NOT.TIME(MPRIME,1).LT.T(I)).AND.
000112          1 TIME(MPRIME,1).LT. T(I+1)) GO TO 80
000113 70  CONTINUE
000114      I = I - 1
000115 80  ILO = I
000116      VOLTIM = (V(I+1)-V(I))*(TIME(MPRIME,1)-T(I))/(T(I+1)-T(I)) + V(I)
000117 200  RETURN
000118      END

```

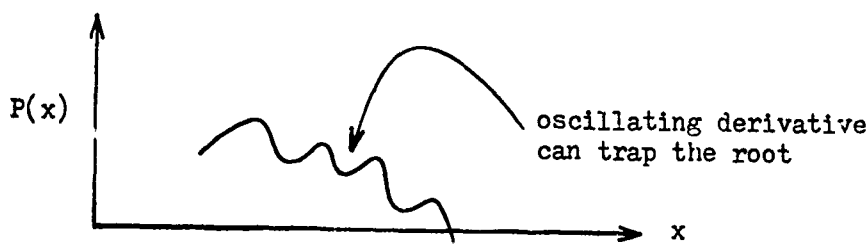
Figure 46b. V O L T I M Listing (cont.)

representation, while the second column contains the same in a logarithmic basis. These arrays are moduloed down to a maximum size of LMAX if the number of time steps exceeds LMAX. An interesting aspect of convolutional computational methods is that the response of old stimuli fade away as the partitions reach the flat portions of the surface; hence, a restricted prior time span is usually quite sufficient. Thus, LMAX does not need to be more than a few hundred.

5. SUBROUTINE DRIVER

DRIVER is essentially a Newton-Raphson iteration routine which has added features to attempt to cope with some of the problems that occur in a zero-finding problem such as this. The routine looks for sign changes on the function P versus variations in the iterate, which is the adjusted input variable x . When a sign change occurs, an upper or lower bound is moved in to contain the iterate. This is normally a wise strategy provided the root does not slide with the iterate, as may happen in some non-linear problems. Root sliding can also occur if the iteration process has terms carrying over from prior iterates or time steps (the computational delay problem). The listing of DRIVER is given in Figure 47.

Another difficult class of problem for an iteration driver routine is the case of an oscillating derivative accompanied by no zero crossing. The iterate is then trapped at a local minimum, and cannot escape. The strategy employed here is to allow so many trials which do not yield a sign change, and then to initiate a step and search procedure which expands the iteration zone by an arbitrary factor.



The expansion of the search zone can lead to an instability in the iteration process. The situation is improved if the function P is relatively free of pathological dependencies on the computational process. Care in arranging statements that compute transfer functions proved of significance in this regard, so that an intrinsic computational delay does not lead to oscillations on the next iterate.

```

000001      SUBROUTINE DRIVER(XITER,/UNCT,LWRBND,UPRBND,IPRINT,ITERAT,LPATH, DRI
000002      1 YO,F,EPSLON)
000003      C
000004      C      STEWART RESEARCH ENTERPRISES, 23376 BELVOIR DR., LOS ALTOS, CALIF.
000005      C
000006      C      ITERATION DRIVER FOR MAIN.
000007      C
000008      DIMENSION ROOTS(3)
000009      REAL LWRBND
000010      LOGICAL SKIP
000011      DATA SKIP/.TRUE./,ITNMAX/35/,RATIO/1000./,ETA/2.E-6/
000012      DATA IT/5/,CHANGE/.5/
000013      DATA MAXR/3/,NS/0/,ROOTS(1)/0.0/,ROOTS(2)/0.0/,NFOUND/2/,NTOTAL/1/
000014      DATA TINY/1.E-20/, SMALL/1.E-12/
000015      C
000016      IF(ITERAT.NE.0) GO TO 243
000017      C
000018      C      FIRST PASS. USE GIVEN ITERATE.
000019      C
000020      NHLP = 0
000021      5 DELTA=(UPRBND-LWRBND)/(RATIO*XITER)
000022      GR = XITER/(1.+DELTA)
000023      NRTS=MIN0(NTOTAL,MAXR)
000024      WRITE(IT,7)
000025      7 FORMAT(1H+,80X,4H M,9X,3HRLX,9X,3HRUA,10X,4HPOLE)
000026      IF (NRTS.GT.0) GO TO 9
000027      WRITE (6,8)NFOUND,NTOTAL,MAXR
000028      8 FORMAT(22H0ZERO ROOTS REQUESTED.,3I6)
000029      GO TO 300
000030      9 N4=NFOUND
000031      NXG=1
000032      DX=(UPRBND-LWRBND)/20.
000033      DX=DX*1.0000001
000034      IPOLE=0
000035      POLE=LWRBND
000036      RLX=LWRBND
000037      RUX=UPRBND
000038      10 GR=AMAX1(GR,LWRBND)
000039      XITER=GR*(1.0+DELTA)
000040      ITN=0
000041      NBK=0
000042      IFV=0
000043      M=1
000044      GO TO 240
000045      C
000046      C      2ND PASS, ITERATE IS RATIO OF FIRST DEPENDING ON BOUNDS.
000047      C
000048      15 X1=XITER
000049      Y1=FUNCT
000050      L1=NS
000051      XITER=GR
000052      M=2
000053      GO TO 240
000054      C
000055      C      3RD PASS. USE NEWTONS METHOD.
000056      C
000057      20 X2=XITER
000058      Y2=FUNCT

```

NEWRO
NEWRO

NEWRO

NEWRO

NEWRO
NEWRO
NEWRO
NEWRO

NEWRO
NEWRO
NEWRO
NEWRO
NEWRO

NEWRO
NEWRO

Figur. 47a. D R I V E R Listing

000059	L2=NS	NEWRO
000060	GO TO 90	NEWRO
000061	C	
000062	C 4TH AND FOLLOWING PASSES. LOOK FOR ZERO CROSSINGS.	
000063	C	
000064	25 IF (NBK.EQ.1) GO TO 65	NEWRO
000065	IF (FUNCT*Y2) 35,35,30	NEWRO
000066	30 IF (FUNCT*Y1) 35,35,65	NEWRO
000067	C IF FIND ONE, SET NBK = 1	
000068	35 NBK=1	NEWRO
000069	IF (X1.LT.X2) GO TO 40	NEWRO
000070	X=X1	NEWRO
000071	X1=X2	NEWRO
000072	X2=X	NEWRO
000073	Y=Y1	NEWRO
000074	Y1=Y2	NEWRO
000075	Y2=Y	NEWRO
000076	L=L1	NEWRO
000077	L1=L2	NEWRO
000078	L2=L	NEWRO
000079	40 IF (X1.LT.XITER) GO TO 45	NEWRO
000080	X=XITER	NEWRO
000081	XITER=X1	NEWRO
000082	X1=X	NEWRO
000083	Y=FUNCT	NEWRO
000084	FUNCT=Y1	NEWRO
000085	Y1=Y	NEWRO
000086	L=NS	NEWRO
000087	NS=L1	NEWRO
000088	L1=L	NEWRO
000089	45 IF (X2.LT.XITER) GO TO 50	NEWRO
000090	X=X2	NEWRO
000091	X2=XITER	NEWRO
000092	XITER=X	NEWRO
000093	Y=Y2	NEWRO
000094	Y2=FUNCT	NEWRO
000095	FUNCT=Y	NEWRO
000096	L=L2	NEWRO
000097	L2=NS	NEWRO
000098	NS=L	NEWRO
000099	50 IF (Y1*Y2) 55,55,60	NEWRO
000100	C	
000101	C MOVE BOUNDS, RLX AND RUX, IF HAVE CROSSING.	
000102	C MOVING BOUNDS AS CLOSE IN ON CROSSING CAN BE BAD IF FUNCTION IS	
000103	C PATHOLOGICAL SUCH THAT ROOT SLIDES WITH ITERATE.	
000104	C	
000105	55 RLX=X1	NEWRO
000106	RLY=Y1	NEWRO
000107	NRL=L1	NEWRO
000108	RUX=X2	NEWRO
000109	RUY=Y2	NEWRO
000110	NRU=L2	NEWRO
000111	GO TO 90	NEWRO
000112	60 RLX=X2	NEWRO
000113	RLY=Y2	NEWRO
000114	NRL=L2	NEWRO
000115	RUX=XITER	NEWRO
000116	RUY=FUNCT	NEWRO
000117	NRU=NS	NEWRO
000118	X1=RLX	NEWRO

Figure 47b. DRIVER Listing (cont.)

```

000119          Y1=RLY                      NEWRO
000120          L1=NRL                      NEWRO
000121          X2=RUX                      NEWRO
000122          Y2=RU Y                    NEWRO
000123          L2=NRU                      NEWRO
000124          GO TO 90                    NEWRO
000125      65  X1=X2                      NEWRO
000126          Y1=Y2                      NEWRO
000127          L1=L2                      NEWRO
000128          X2=XITER                    NEWRO
000129          Y2=FUNCT                    NEWRO
000130          L2=NS                       NEWRO
000131          IF (NRK.EQ.0) GO TO 90        NEWRO
000132          IF (FUNCT) 70,70,80          NEWRO
000133      70  IF (RLY) 75,75,85            NEWRO
000134      75  RLX=XITER                    NEWRO
000135          GO TO 90                    NEWRO
000136      80  IF (RU Y) 75,75,85            NEWRO
000137      85  RUX=XITER                    NEWRO
000138      C  90  ITN=ITN+1                  NEWRO
000139          T1=Y1                      NEWRO
000140          T2=Y2                      NEWRO
000141          IF (SKIP) GO TO 94            NEWRO
000142          I=MAX0(L1,L2)                 NEWRO
000143          T1=Y1*2.** (L1-I)              NEWRO
000144          T2=Y2*2.** (L2-I)              NEWRO
000145      94  M=3                           NEWRO
000146          IF (T2-T1) 100,95,100          NEWRO
000147      95  XITER=X1+.5*(X2-X1)            NEWRO
000148          GO TO 105                     NEWRO
000149      C                                     NEWRO
000150      100 X2MX1 = X2 - X1                 NEWRO
000151          T2MT1 = T2 - T1                 NEWRO
000152          IF (ABS(T2MT1).LT.TINY.AND.ABS(X2MX1).GT.(T2MT1/SMALL)) T2MT1=SMALL
000153      C                                     NEWRO
000154      C  BASIC ITERATION EQUATION.
000155          XITER = X1 - T1 * X2MX1/T2MT1
000156      C                                     NEWRO
000157          IF (NRK.EQ.1) GO TO 110
000158      105 IF (XITER.LT.LWRBND.OR.XITER.GT.UPRBND) GO TO 200
000159      110 NTEST=2                          NEWRO
000160          IF (ABS(XITER-X2).LT.ETA*ABS(XITER)) GO TO 125
000161          IF (ABS(FUNCT).LT.EPSLON*ABS(XITER).AND.ITERAT.GE.2) GO TO 135
000162          IF (NRK.EQ.0) GO TO 130
000163          IF (RLX-XITER) 115,240,120
000164      115 IF (XITER-RUX) 240,240,120
000165      C                                     NEWRO
000166      C  RESETTING ITERATE IF OUTSIDE OF BOUNDS.
000167      C                                     NEWRO
000168      120 XITER=(RLX+RUX)/2.               NEWRO
000169          GO TO 240                      NEWRO
000170      125 IF (ABS(XITER-X1).LT.ETA * ABS(XITER)) GO TO 135
000171          IF ((T1*T2).LT.0.) XITER=X1+.8*(X2-X1)
000172      130 M=3                              NEWRO
000173          IF (ITN=15 ) 240,240,200
000174      135 IF (ABS(FUNCT).GT.ABS(XITER)) GO TO 200
000175      C                                     NEWRO
000176      C  ROOT FOUND.
000177      C                                     NEWRO
000178          N4=MIN0(N4+1,MAXR)

```

Figure 47c. D R I V E R Listing (cont.)

```

000179      ROOTS(N4)=XITER
000180      GR=AMAX1(.8*ABS(XITER),1.5*LWRBND)
000181      IF(IPRINT.NE.0)
000182      1WRITE (6,140)XITER,ITN,IFV,NTEST
000183      140 FORMAT(7H ROOT =,E14.8,5X,6HITNS =,I3,5X,6HEVAL =,I3,5X,8HTEST NO.
000184      1,12,60X,3H***)
000185      IFV=0
000186      IPOLE=0
000187      RLX=0.
000188      RUX=0.
000189      NRK=0
000190      ITN=0
000191      IF (NRTS-N4) 145,145,10
000192      C
000193      C      SORT THE ROOTS IN ASCENDING ORDER.
000194      C
000195      1-5 N7=N4-1
000196      IF (N7.LE.0) GO TO 155
000197      DO 150 I=1,N7
000198      K=I+1
000199      DO 150 J=K,N4
000200      IF (ROOTS(I).LT.ROOTS(J)) GO TO 150
000201      T1=ROOTS(I)
000202      ROOTS(I)=ROOTS(J)
000203      ROOTS(J)=T1
000204      150 CONTINUE
000205      155 IF (N4.GE.MAXR) GO TO 295
000206      C
000207      C      CHECK INTERVALS BETWEEN ROOTS FOR SIGN CROSSINGS.
000208      C
000209      ITN=0
000210      I=NXG-1
000211      M=6
000212      NBK=1
000213      160 I=1
000214      N4'-I
000215      IF (I.GT.N4) ROOTS(I)=UPRBND
000216      IF (I.EQ.1) GO TO 165
000217      XITER=ROOTS(I-1)*1.001
000218      X=(ROOTS(I)-ROOTS(I-1))/9.99
000219      IF (X.LT..01) GO TO 195
000220      IF (M-6) 240,240,175
000221      165 X=(ROOTS(I)-LWRBND)/9.99
000222      IF (X.LT..01) GO TO 195
000223      XITER=LWRBND
000224      GO TO 240
000225      170 X=XITER
000226      Y2=FUNCT
000227      L2=NS
000228      175 DO 190 J=1,10
000229      XITER=AMIN1(XITER+X,UPRBND)
000230      M=7
000231      GO TO 240
000232      180 IF (Y2*FUNCT) 60,60,185
000233      185 X2=XITER
000234      Y2=FUNCT
000235      L2=NS
000236      190 CONTINUE
000237      195 IF (1.GT.N4) GO TO 295
000238      IF (I-NRTS) 160,295,295

```

Figure 47d. D R I V E R Listing (cont.)

000239	C		NEWRO
000240	C	SEARCH FOR A MORE SUITABLE GUESS.	NEWRO
000241	C		NEWRO
000242		200 IF (NHELP .GT.4) GO TO 145	NEWRO
000243		NHELP = NHELP + 1	
000244		ITN=0	
000245		NBK = 0	
000246	C		
000247	C	MAYBE TRAPPED BY SLIDING ROOF. FIRST TRY MOVING BOUNDS OUT.	
000248	C		
000249		CHANGE = CHANGE **NHELP	
000250		IF(XITER) 202,202,203	
000251		202 LWRBND = LWRBND / CHANGE	
000252		UPRBND = UPRBND * CHANGE	
000253		GO TO 204	
000254		203 LWRBND = LWRBND * CHANGE	
000255		UPRBND = UPRBND / CHANGE	
000256		204 RLX = LWRBND	
000257		RUX = UPRBND	
000258		POLE= LWRBND	
000259		DX= (UPRBND - LWRBND)/20.0	
000260		IF(NHELP.LE.2) GO TO 65	
000261	C		
000262	C	NOW TRY SYSTEMATIC STEPPING. (PUSH THE PANIC BUTTON).	
000263	C		
000264		XITER=POLE	NEWRO
000265		IF (IPOLE.NE.0) GO TO 201	
000266		IPOLE=1	
000267		M=4	NEWRO
000268		GO TO 240	NEWRO
000269		201 FUNCT=RFUNCT	
000270		NS=NNR	
000271		205 X2=XITER	NEWRO
000272		Y2=FUNCT	NEWRO
000273		L2=NS	NEWRO
000274		M=5	NEWRO
000275		210 POLE=AMIN1(POLE+DX,UPRBND)	NEWRO
000276		XITER=POLE	NEWRO
000277		GO TO 240	NEWRO
000278		215 X1=X2	NEWRO
000279		Y1=Y2	NEWRO
000280		L1=L2	NEWRO
000281		X2=XITER	NEWRO
000282		Y2=FUNCT	NEWRO
000283		L2=NS	NEWRO
000284		RFUNCT=FUNCT	
000285		NNR=NS	
000286		IF (Y1*Y2) 25,25,220	NEWRO
000287		220 IF (L2-L1) 90,225,230	NEWRO
000288		225 IF (ABS(Y2)-ABS(Y1)) 90,230,230	NEWRO
000289		230 IF (XITER-UPRBND) 210,235,235	NEWRO
000290		235 NHELP=1	NEWRO
000291		GO TO 145	NEWRO
000292	C		NEWRO
000293	C	COMPUTE FUNCTION AND DIVIDED FUNCTION.	NEWRO
000294	C		NEWRO
000295		240 IF (XITER.GT.UPRBND) GO TO 200	NEWRO
000296		NTEST=1	NEWRO
000297		IFV=IFV+1	NEWRO
000298		ITN = ITN + 1	NEWRO

Figure 47e. DRIVER Listing (cont.)

```

000299          ITERAT=ITERAT + 1
000300          LPATH=1
000301          RETURN
000302      C      RETURN FOR FUNCTION EVALUATION.
000303      C
000304      C      ENTRY POINT WITH NEW FUNCTION VALUE.
000305      C
000306      243 IF(ITERAT.GE.ITNMAX) GO TO 296
000307          WRITE(IT,244) M,RLX,RUX,POLE
000308      244 FORMAT(1H+,83X,11,4X,3(G12.4))
000309          T1=FUNCT
000310          N2=NS
000311          IF(FUNCT.EQ.0.0.AND.XITER.NE.0.0) GO TO 135
000312      245 IF (N4.EQ.0) GO TO 255
000313      C
000314      C      SCALER IS NEEDED WHEN FUNCTION VALUES TAKE ON EXTREME VALUES DUE
000315      C      TO DIVIDING BY ROOTS ALREADY FOUND, IN CASES OF MULTIPLE ROOTS.
000316      C
000317      C      IF SCALER IS REQUIRED, SET SKIP = .FALSE.
000318          IF(SKIP) GO TO 275
000319          DO 250 L=1,N4
000320          FUNCT=FUNCT/(XITER-ROOTS(L))
000321          CALL SCALER(FUNCT,J)
000322      250 NS=NS+J
000323      255 IF(IPRINT) 260,270,260
000324          260 WRITE (6,265)XITER,FUNCT,NS,T1,N2,M,RLX,RUX
000325          265 FORMAT(E20.8,5X,2(E9.3,14),18,5X,2E15.8)
000326          270 IF (FUNCT) 275,260,275
000327      C
000328      C      BASIC SYEERING SWITCH TO DIFFERENT STRATEGIES.
000329      C
000330      275 GO TO (15,20,25,205,215,170,180),M
000331      280 NTEST=6
000332          GO TO 135
000333      295 NTOTAL=N4
000334      296 IF(ITERAT.GE.ITNMAX) WRITE(IT,298)
000335      298 FORMAT(36H DRIVER ITERATION MAXIMUM EXCEEDED.)
000336      300 LPATH=2
000337          IF(ITERAT.GE.ITNMAX) LPATH=3
000338          RETURN
000339      C 300 RETURN
000340      C      TERMINATION TESTS
000341      C          1 = NORMAL ROOT ACCEPTANCE, F(X)=0.
000342      C          2 = NORMAL CONVERGENCE TO EPSILON
000343      C          3 = MAXIMUM ITERATION COUNT (100) EXCEEDED.
000344      C          6 = DIVIDED RESIDUAL WENT TO ZERO (CAUSED BY MULTIPLE ROOTS).
000345      C      END

```

Figure 47f. D R I V E R Listing (cont.)

For certain types of zero-finding problems, such as polynomial evaluations, more than one root may exist over the range of variation of the iterate. In such cases DRIVER divides the original function P by the root already found, creating a new function which does not have the old root inherently contained within the iteration zone. Such a situation is shown in Figure 48. Thus, the standard iteration strategy can now be employed to find the remaining roots without just refinding the old ones. The only difficulty which is encountered by this process, is that the new function P can take on extremely large values if the root is small (close to zero). DRIVER then can call a separate subroutine SCALER which separates the power of ten of a number and carries it separately through the computation in DRIVER so that overflow within the computer does not occur.

6. FUNCTION SUBPROGRAMS FE and FI

These two subroutines return the value of a block transfer or response function to MAIN. In the case of some elements this is a trivially simple calculation. In cases where some surface operation is required, FE and FI prepare the call to CONVOL. The initial sections of FE and FI are computed GO TO's which take control to the proper subsection for the particular device class as determined by ITABLE(J,1). The listings for these routines are given in Figure 49 and Figure 50.

The COMMENT cards in the listing indicate the device class treated therein. Generally rather simple expressions have been used in our work so far. Note that in sections 7 and 8, s variable transfer functions are given for linear elements. The basic computational method can equally well be applied in the s variable domain.

7. FUNCTION SUBPROGRAM FEPHI(J,IRRAD)

FEPHI returns to MAIN the radiation voltage response as calculated by CONVOL; the listing of FEPHI is given in Figure 51. The prime task performed by FEPHI is to ascertain the correct surface number LSURF knowing only the block number J. First J is used as a look-up entry in ITABLE to find the device class number. This number, set to INDEX, is used together with the variable IRRAD coming in through the call, to find LSURF by look-up in the small table KSURF. IRRAD has the value one for X-ray and the value two for neutron irradiation. The contents

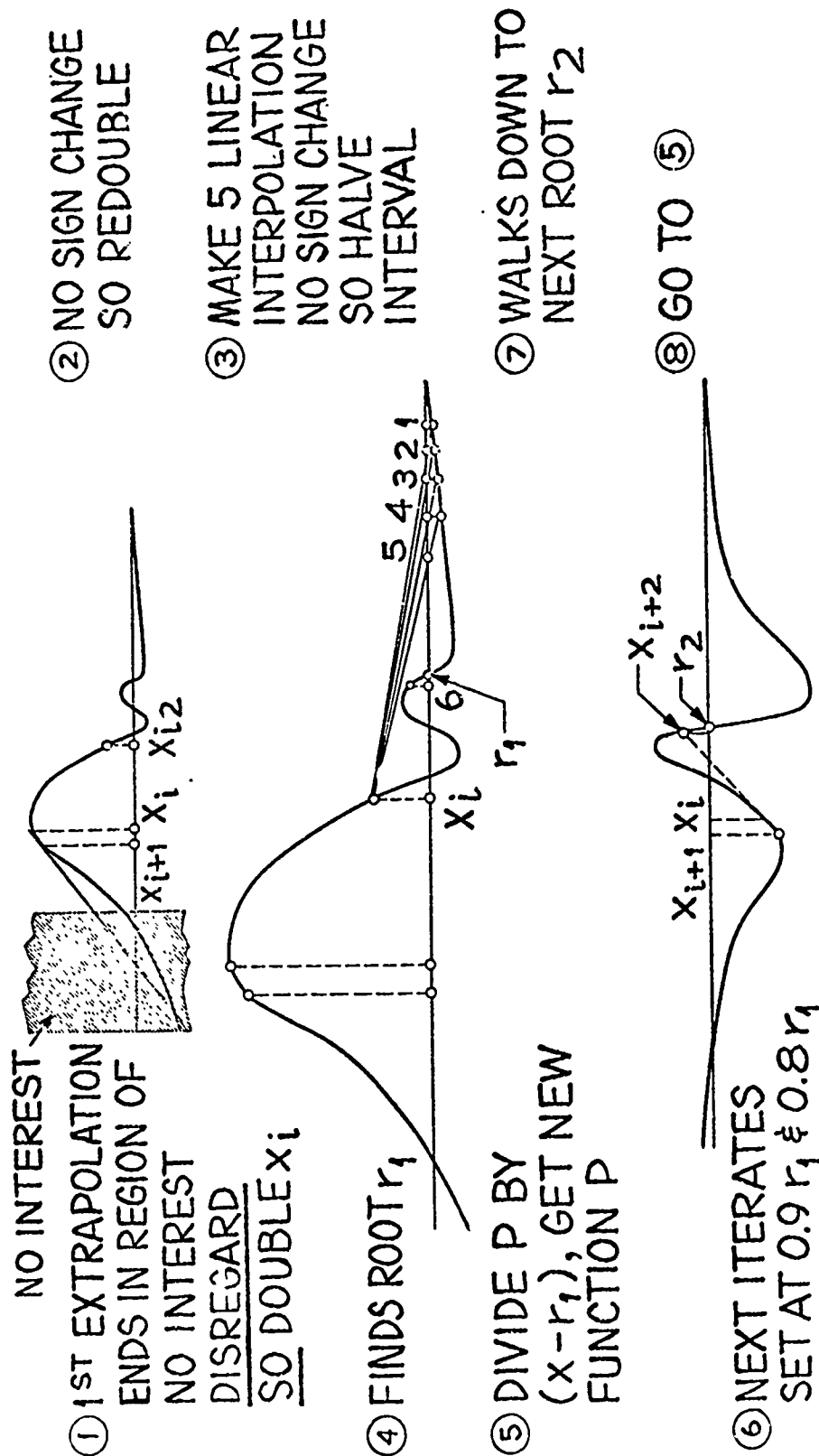


Figure 48. Strategies Employed in Linear Interpolation
Root Finding Routine DRIVER

ELT FE,1,720302, 47804 , 1

```

000001      FUNCTION FE(J)
000002      INCLUDE SAP,LIST
000003      DATA IN/5/,IT/6/
000004      INDEX = 1TABLE(J,1)
000005      GO TO(1,2,3,4,5,6,7,8,9,10,11,12,13,14) , INDEX
000006      1  FE = FEE01(J)
000007      RETURN
000008      2  FE = FEE02(J)
000009      RETURN
000010      3  FE = FEE03(J)
000011      RETURN
000012      4  FE = FEE04(J)
000013      RETURN
000014      5  FE = FEE05(J)
000015      RETURN
000016      6  FE = FEE06(J)
000017      RETURN
000018      7  FE = FEE07(J)
000019      RETURN
000020      8  FE = FEE08(J)
000021      RETURN
000022      9  FE = FEE09(J)
000023      RETURN
000024      10 FE = FEE10(J)
000025      RETURN
000026      11 FE = FEE11(J)
000027      RETURN
000028      12 FE = FEE12(J)
000029      RETURN
000030      13 FE = FEE13(J)
000031      RETURN
000032      14 FE = FEE14(J)
000033      RETURN
000034      C
000035      C      01 = RESISTOR, SERIES CONFIGURATION.
000036      FUNCTION FEE01(J)
000037      GO TO (10,20), KROSS
000038      10 FEE01 = 1. - CIN(J,1)*R(J)/VIN(J,1)
000039      RETURN
000040      20 FEE01 = VIN(J,1)/CIN(J,1) - R(J)
000041      RETURN
000042      C
000043      C      02 = RESISTOR, SHUNT CONFIGURATION.
000044      FUNCTION FEE02(J)
000045      GO TO (10,20), KROSS
000046      10 FEE02 = 1.
000047      RETURN
000048      20 FEE02 = VIN(J,1)/CIN(J,1)
000049      RETURN
000050      C
000051      C      03 = DIODE, SERIES, FORWARD BIASED.
000052      FUNCTION FEE03(J)
000053      ARG = CIN(J,1)/CDIODE + 1.0
000054      IF(ARG) 10,10,20
000055      10 FEE03 = 1.0 - (TKQ/VIN(J,1))*(ARG - 1.0)
000056      RETURN
000057      20 FEE03 = 1.0 - (TKQ/VIN(J,1))*ALOG(ARG)
000058      RETURN

```

Figure 49a. F E Listing

```

000059      C
000060      C      04 = DIODE, SERIES, REVERSE BIASED.
000061      FUNCTION FEE04(J)
000062      ARG = - CIN(J,1)/CDIODE + 1.0
000063      IF(ARG) 10,10,20
000064      10 FEE04 = (TKO/VIN(J,1))*(ARG - 1.0)
000065      RETURN
000066      20 FEE04 = 1.0 + (TKO/VIN(J,1)) * ALOG(ARG)
000067      RETURN
000068      C
000069      C      05 = DIODE, SHUNT, FORWARD BIASED.
000070      FUNCTION FEE05(J)
000071      FEE05 = 1.0
000072      RETURN
000073      C
000074      C      06 = DIODE, SHUNT, REVERSE BIASED.
000075      FUNCTION FEE06(J)
000076      FEE06 = 1.0
000077      RETURN
000078      C
000079      C      07 = CAPACITOR, SERIES, S VARIABLE ANALYSIS.
000080      FUNCTION FEE07(J)
000081      CINTGL(J,1) = ENTGRL(DT,J,CIN,IX,JT,M) + CINTGL(J,3)
000082      FEE07 = 1.0 - CINTGL(J,1)/(C(J)*VIN(J,1))
000083      RETURN
000084      C
000085      C      08 = CAPACITOR, SHUNT, S VARIABLE ANALYSIS
000086      FUNCTION FEE08(J)
000087      FEE08 = 1.0
000088      RETURN
000089      C
000090      C      09 = INDUCTANCE, SERIES, S VARIABLE ANALYSIS.
000091      FUNCTION FEE09(J)
000092      CINDT1(J,1) = DIFFER(DT,J,CIN,IX,JT,M)
000093      FEE09 = 1.0 - (FL(J)/VIN(J,1))* CINDT1(J,1)
000094      RETURN
000095      C
000096      C      10 = INDUCTANCE, SHUNT, S VARIABLE ANALYSIS.
000097      FUNCTION FEE10(J)
000098      FEE10 = 1.0
000099      RETURN
000100      C
000101      C      11 = INTEGRATED CIRCUIT 709 LINEAR OP AMP.
000102      FUNCTION FEE11(J)
000103      DATA ROUT/0.1/
000104      LSURF = 1
000105      ISTEER = 1
000106      V = VIN(J,1)
000107      IF(V.EQ.0.0) V = 1.E-6
000108      FEE11 = CONVOL(J,LSURF,ISTEER) - OUT(J,2) * ROUT/V
000109      RETURN
000110      C
000111      C      12 = FLIP-FLOP
000112      FUNCTION FEE12(J)
000113      FEE12 = 0.0
000114      RETURN
000115      C
000116      C      13 = INTEGRATED CIRCUIT 9704 DUAL NAND GATE
000117      FUNCTION FEE13(J)
000118      DATA LSURF/6/, ISTEER/1/

```

Figure 49b. F E Listing (cont.)

```

000119      V = VIN(J,1)
000120      IF(V.EQ.0.0) V = 1.E-6
000121      IF(ITERAT.EQ.1.OR.V.NE.VOLD) X = CONVOL(J,I SURF,ISTEER)
000122      FEE13 = X
000123      VOLD = V
000124      GO TO(10,20), KROSS
000125  10 RETURN
000126  20 FEE13 = FEE13 * V/CIN(J,1)
000127      RETURN
000128  C
000129  C      14 = INTEGRATED CIRCUIT 741 LINEAR OP AMP
000130      FUNCTION FEE14(J)
000131      DATA RIN/500.0/,ROUT/0.1/
000132      LSURF = 3
000133      ISTEER = 1
000134      V = VIN(J,1)
000135      IF(V.EQ.0.0) V = 1.E-6
000136      IF(ITERAT.EQ.1.OR.V.NE.VOLD) X = -CONVOL(J,LSURF,ISTEER)
000137      FEE14 = X - (COUT(J,1)*CIN(J,1)-V/RIN)*ROUT/V
000138      VOLD = V
000139      GO TO (10,20), KROSS
000140  10 RETURN
000141  20 FEE14 = FEE14 * V/CIN(J,1)
000142      RETURN
000143      END

```

Figure 49c. F E Listing (cont.)

P ELT FI,1,720302, 47807 , 1

```

000001      FUNCTION FI(J)
000002      INCLUDE SAP,LIST
000003      INDEX = ITABLE(J,1)
000004      GO TO(1,2,3,4,5,6,7,8,9,10,11,12,13,11), INDEX
000005      1  FI = FII01(J)
000006      RETURN
000007      2  FI = FII02(J)
000008      RETURN
000009      3  FI = FII03(J)
000010      RETURN
000011      4  FI = FII04(J)
000012      RETURN
000013      5  FI = FII05(J)
000014      RETURN
000015      6  FI = FII06(J)
000016      RETURN
000017      7  FI = FII07(J)
000018      RETURN
000019      8  FI = FII08(J)
000020      RETURN
000021      9  FI = FII09(J)
000022      RETURN
000023      10 FI = FII10(J)
000024      RETURN
000025      11 FI = FII11(J)
000026      RETURN
000027      12 FI = FII12(J)
000028      RETURN
000029      13 FI = FII13(J)
000030      RETURN
000031      C
000032      C      01 = RESISTOR, SERIES CONFIGURATION.
000033      FUNCTION FII01(J)
000034      GO TO (10,20), KROSS
000035      10 FII01 = 1.
000036      RETURN
000037      20 FII01 = CIN(J,1)/VIN(J,1)
000038      RETURN
000039      C
000040      C      02 = RESISTOR, SHUNT CONFIGURATION.
000041      FUNCTION FII02(J)
000042      GO TO (10,20), KROSS
000043      10 FII02 = 1. - VIN(J,1)/(CIN(J,1)*R(J))
000044      RETURN
000045      20 FII02 = CIN(J,1)/VIN(J,1) - 1.0/R(J)
000046      RETURN
000047      C
000048      C      03 = DIODE, SERIES, FORWARD BIASED.
000049      FUNCTION FII03(J)
000050      FII03 = 1.0
000051      RETURN
000052      C
000053      C      04 = DIODE, SERIES, REVERSE BIASED.
000054      FUNCTION FII04(J)
000055      FII04 = 1.0
000056      RETURN
000057      C
000058      C      05 = DIODE, SHUNT, FORWARD BIASED.

```

Figure 50a. F I Listing

```

000059      FUNCTION FII05(J)
000060      IF(VIN(J,1)/TKQ-40.0) 20,10,10
000061      10 VIN(J,1)=TKQ * 12.0 *ALOG(ABS(CIN(J,1)*(1.0E-12)/CDIODE))
000062      GO TO 50
000063      20 IF(VIN(J,1))30,40,50
000064      30 FII05 = 1.0 - VIN(J,1)/(CIN(J,1) * RPD10D)
000065      RETURN
000066      40 FII05 = 1.0
000067      RETURN
000068      50 FII05 = 1.0 -(CDIODE/CIN(J,1))*(EXP(VIN(J,1)/TKQ) - 1.0)
000069      RETURN
000070
000071      C      06 = DIODE, SHUNT, REVERSE BIASED.
000072      C      FUNCTION FII06(J)
000073      FII06 = 1.0 - (CDIODE/CIN(J,1))*(EXP(-VIN(J,1)/TKQ) - 1.0)
000074      RETURN
000075
000076      C      07 = CAPACITOR, SERIES, S VARIABLE ANALYSIS.
000077      C      FUNCTION FII07(J)
000078      FII07 = 1.0
000079      RETURN
000080
000081      C      08 = CAPACITOR, SHUNT, S VARIABLE ANALYSIS.
000082      C      FUNCTION FII08(J)
000083      VINDT1(J,1) = DIFFER(DT,J,VIN,IX,JT,M)
000084      FII08 = 1.0 - C(J)/CIN(J,1) * VINDT1(J,1)
000085      RETURN
000086
000087      C      09 = INDUCTANCE, SERIES, S VARIABLE ANALYSIS.
000088      C      FUNCTION FII09(J)
000089      FII09 = 1.0
000090      RETURN
000091
000092      C      10 = INDUCTANCE, SHUNT, S VARIABLE ANALYSIS.
000093      C      FUNCTION FII10(J)
000094      VINTGL(J,1) = ENTGRL(DT,J,VIN,IX,JT,M) + VINTGL(J,3)
000095      FII10 = 1.0 - VINTGL(J,1)/(FL(J)*CIN(J,1))
000096      RETURN
000097
000098      C
000099      C      11 = INTEGRATED CIRCUIT 709 LINEAR OP AMP.
000100      C      FUNCTION FII11(J)
000101      DATA RIN/500.0/, CGAIN/1.E+4/
000102      CUROUT = CGAIN*(CIN(J,1) -VIN(J,1)/
000103      1 RIN) + VOUT(J,1)/R(J+1)
000104      GO TO (10,20), KROSS
000105      10 FII11 = CUROUT/CIN(J,1)
000106      RETURN
000107      20 FII11 = CUROUT/VIN(J,1)
000108      RETURN
000109
000110      C
000111      C      12 = FLIP-FLOP
000112      C      FUNCTION FII12(J)
000113      GO TO(10,20), KROSS
000114      10 FII12 = 1.0*VOUT(J,2)/(CIN(J,1)*R(J+1))
000115      RETURN
000116      20 FII12 = (CIN(J,1)+VOUT(J,2)/R(J+1))/VIN(J,1)
000117      RETURN
000118      C

```

Figure 50b. F I Listing (cont.)

```

000119      C      13 = 9704 DUAL NAND GATE
000120      FUNCTION FII13(J)
000121      DATA CGAIN/1.E+4/
000122      CUROUT = CGAIN * CIN(J,1) + VOUT(J,1)/R(J+1)
000123      GO TO (10,20), KROSS
000124      10 FII13 = CUROUT/CIN(J,1)
000125      RETURN
000126      20 FII13 = CUROUT/VIN(J,1)
000127      RETURN
000128      END

```

Figure 50c. F I Listing (cont.)

```

      # ELT FEPHI,1,720302, 47805      . 1

000001      FUNCTION FEPHI(J,IRRAD)
000002      INCLUDE SAP,LIST
000003      INDEX = ITABLE(J,1)
000004      LSURF = KSURF(INDEX,IRRAD+3)
000005      IF(LSURF) 10,10,20
000006      10 FEPHI = 0.0
000007      RETURN
000008      20 ISTEER = IRRAD + 2
000009      FEPHI = CONVOL(J,LSURF,ISTEER)
000010      RETURN
000011      END

```

Figure 51. F E P H I Listing

of the array KSURF are shown in Table II. The columns of the array represent the type of transfer function surface upon which SURFB will have to perform an interpolation. Column one denotes the total number of surfaces available for a given device class. The rows of KSURF denote the device class as indexed by ITABLE, and corresponds to the sequence as indicated by comment cards in FE. The subprogram FIPHI is similar to FEPHI, with the column index variable increased by two compared to that in FEPHI so as to pick out the proper columns in KSURF.

8. FUNCTION SUBPROGRAM CONVOL(J,LSURF,ISTEER)

Superposition or convolution of the running sums created by the partitions driving back along the surface is the main operation performed by CONVOL. The listing of CONVOL is given in Figure 52. The variable J in the call is the block number. LSURF is the surface number being dealt with, and is passed on by CONVOL to TSURF. ISTEER tells CONVOL what type of stimulus function applies-- that is either voltage, current, X-ray dosage, or neutron dosage, and takes on values from one to four, respectively.

On the first time step, CONVOL calculates the length of the running sums needed by taking and dividing the length of the surface in the time direction by the initial time step. If this number is larger than the number of storage locations LMAX set aside for the running sums, a message is printed. The first time that CONVOL is called on a given time step from any of the routines which can call it, the section of program starting at statement 90 is entered. The DO loop limits are calculated using the time step counter M as moduloed down within the range LMAX, together with the IRANGE variable calculated earlier. The running sums move along in a fixed storage length and then fold back when the maximum is reached, as indicated by the COMMENT cards in the listing of Figure 52b.

With the limits on the DO established, the running sum is convolved. Four essential pieces of information are needed:

1. TIMER which is the elapsed time since a partition was created, until the current time step. The second column in the TIMER array contains that value in log form. TIMER is a look-up variable for surface interpolation.

Table II. KSURF Array

J →

1 2 3 4 5 6 7

I ↓

1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11	2	1	2			
12						
13	5	6		8	9	10
14	3	3		4	5	
15						
16						
17						
18						
19						
20						
	No. of surf. for given block type	TE	TI or STOR	T E P H I X	T E P H I N	T I P H I N

entries = 0
if blank

```

000001      FUNCTION CONVOL(J,LSURF,ISTEER)
000002      INCLUDE SAP,LIST
000003      INCLUDE TIME,LIST
000004      INCLUDE XYZ,LIST
000005      COMMON/PARALL/LITER,JPAPL
000006      DIMENSION U(LMAX,IX),SUMOLD(IX),TERM(IX),IRANGE(IX)
000007      DIMENSION KSURFX(12)
000008      LOGICAL NOMORE,KONVOL,IBOTH,IPRINT
000009      DATA (KSURFX(I),I=1,12)/1,1,1,2,2,1,0,1,2,1,2,0/
000010      DATA IT/6/
000011      C
000012      C      LMAX = FIRST DIMENSION IN U ARRAY.
000013      C
000014      C      THE KSURFX ARRAY INDICATES WHETHER THE XAXIS OF THE SURFACE IS
000015      C      LINEAR OR LOGARITHMIC. 1 = LINEAR, 2 = LOG.
000016      C
000017      IPRINT = .TRUE.
000018      MNEW = M
000019      IF(MNEW.NE.MOLD) MMOD10 = ((M/10)*10)/M
000020      C
000021      C      SELECTING THE PROPER STIMULUS VARIABLE.
000022      GO TO (2,3,4,5), ISTEER
000023      2 DRIVE = VIN(J,1)
000024      GO TO 10
000025      3 DRIVE = CIN(J,1)
000026      GO TO 10
000027      4 DRIVE = PHIXL
000028      GO TO 10
000029      5 DRIVE = PHINL
000030      C
000031      C      HERE THE TAG SUBSCRIPT FOR A SPECIFIC ONE OF A SPECIFIC DEVICE
000032      C      TYPE IS ESTABLISHED.
000033      C
000034      10 INDEX = ITABLE(J,1)
000035      KOUNT = 0
000036      DO 20 KK = 2, IS
000037      KS = KSURF(INDEX,KK)
000038      IF(KS.GT.0) KOUNT = KOUNT + 1
000039      IF(LSURF.NE.KS) GO TO 20
000040      JJ = ITABLE(J,2) - KSURF(INDEX,1) + KOUNT
000041      GO TO 22
000042      20 CONTINUE
000043      22 LL = KSURFX(LSURF)
000044      C
000045      IGO = 1
000046      IF((ISTEER.EQ.3.AND..NOT.KONVOL(1)).OR.(ISTEER.EQ.4.AND..NOT.
000047      1 KONVOL(2))) IGO = 2
000048      IF(ITERAT.GT.1.OR.IGO.EQ.2) GO TO 400
000049      IF(JPARL.EQ.1.AND.LITER.GT.1) GO TO 400
000050      IF(M-1) 30,30,90
000051      30 XHS =ABS( XHIGH(LSURF)/SCALE(LSURF,1))
000052      IF(MODE.EQ.1.AND.LL.EQ.1) IRANGE(J) = XHS/DT
000053      IF(MODE.EQ.1.AND.LL.EQ.2) IRANGE(J) = (10.**XHS)/DT
000054      IF(MODE.EQ.2.AND.LL.EQ.1) IRANGE(J) = (ALOG10(XHS))/DLOGT
000055      IF(MODE.EQ.2.AND.LL.EQ.2) IRANGE(J) = XHS/DLOGT
000056      IF(IRANGE(J).LT.20) IRANGE(J)=20
000057      WRITE(IT,40) J, IRANGE(J)
000058      40 FORMAT(/ 5X,8H IRANGE(,12,3H) =,14 /)

```

Figure 52a. C O N V O L Listing

```

000059      60 IF (IRANGE(J).LE.LMAX) GO TO 75
000060      WRITE(IT,70)
000061      70 FORMAT(/49H TIME STEP DT TOO SMALL. WILL OVERFLOW U ARRAYS.  //)
000062      IRANGE(J) = LMAX
000063      75 DO 80 J1 = 1, IX
000064      SUMOLD(J1) = 0.0
000065      80 TERM(J1) = 0.0
000066      GO TO 400
000067      C
000068      C *****
000069      C *
000070      C * FOLDING OF ARRAYS
000071      C *
000072      C * 1. AT START
000073      C *
000074      C *      ILO1              IH11                      LMAX
000075      C *      1-----I
000076      C *              LATEST
000077      C *
000078      C * 2. LATER
000079      C *      ILO1              IRANGE(J)              IH11                      LMAX
000080      C *      1      I-----I
000081      C *              OLDEST                      LATEST
000082      C *
000083      C * 3. STILL LATER
000084      C *      ILO1              IH11                      ILO2              IH12
000085      C *      1-----I                      I-----I
000086      C *              LATEST                      OLDEST                      LMAX
000087      C *
000088      C *****
000089      C
000090      90 ILO1 = MPRIME - IRANGE(J)
000091      IF (ILO1.LT.1) ILO1 = 1
000092      IF (MPRIME.EQ.1) ILO1 = LMAX - IRANGE(J) + 1
000093      TERM(JJ) = 0.0
000094      IH11 = MPRIME - 1
000095      IF (IH11.LT.1) IH11 = LMAX
000096      IF (MPRIME.LE.IRANGE(J).AND.IH11.LT.IRANGE(J).AND.MODULO.GT.0)
000097      1 GO TO 100
000098      NOMORE = .TRUE.
000099      GO TO 200
000100      100 NOMORE = .FALSE.
000101      ILO2 = MPRIME + LMAX - IRANGE(J)
000102      IH12 = LMAX
000103      C
000104      200 SUM = 0.0
000105      C
000106      C      SETTING DO LOOP LIMITS FOR SUMMING LOWER PART OF RANGE.
000107      ILOW = ILO1
000108      IHIGH = IH11
000109      IBOTH = .FALSE.
000110      GO TO 220
000111      C
000112      C      SETTING DO LOOP LIMITS FOR SUMMING UPPER PART OF RANGE.
000113      210 ILOW = ILO2
000114      IHIGH = IH12
000115      IBOTH = .TRUE.
000116      C
000117      220 IF (MOD10.EQ.1.AND.IPRINT) WRITE(IT,222) M,MPRIME,ILOW,IHIGH,LSURF
000118      222 FORMAT(/ 5H M =,I3,11H, MPRIME =,I3, 8H ILOW =,I3,9H IHIGH =,

```

Figure 52b. C O N V O L Listing (cont.)

```

000119      1 I3, 9H LSURF =, I2,
000120      2 //2X, 64H L   TIMER      UIN      DTL      TEMP      SUM
000121      3   TSURF   )
000122      C
000123      CONVOLVING THE RUNNING SUMS BY DRIVING THE PARTITIONS ALONG THE SURFACE.
000124      DO 300 L = ILOW, IHIGH
000125      UIN = U(L,JJ)
000126      IF(UIN) 230, 300, 230
000127      230 TIMER = TL(L,LL)
000128      TSURFF = TSURF(TIMER,UIN,3,LSURF,LL)
000129      UMULT = UIN
000130      IF(LSURF.EQ.6) UMULT = 1.0
000131      TEMP = TSURFF * UMULT * DTL(L,LL)
000132      IF(MMOD10.EQ.1.AND.IPRINT) WRITE(IT,260) L,TIMER,UIN,DTL(L,LL),
000133      1 TEMP,SUM,TSURFF
000134      260 FORMAT(2X,I3,1X,6(G10.3,1X))
000135      SUM = SUM + TEMP
000136      IF(TIMER.GT.XHIGH(LSURF).AND..NOT.ABS(TEMP).GT.EPSLON) U(L,JJ)=0.0
000137      300 CONTINUE
000138      IF(.NOT.NOMORE.AND..NOT.IBOTH) GO TO 210
000139      SUMOLD(JJ) = SUM
000140      C
000141      COMPUTING CONTRIBUTION OF PRESENT TIME STEP TO RESPONSE SUM.
000142      C
000143      400 IF(ITEPAT.GT.1.AND.IG0.EQ.2) GO TO 480
000144      U(MPRIME,JJ) = DRIVE
000145      UIN = DRIVE
000146      UMULT = UIN
000147      IF(LSURF.EQ.6) UMULT = 1.0
000148      SUM = SUMOLD(JJ) - TERM(JJ)
000149      GO TO(430,440), MODE
000150      430 TIMER = DT
000151      GO TO 460
000152      440 GO TO (450,470), IG0
000153      450 TIMER = DLOGT
000154      460 TERM(JJ) = UMULT * TSURF(TIMER,UIN,1,LSURF,LL)
000155      SUMOLD(JJ) = SUM + TERM(JJ)
000156      GO TO 480
000157      470 TIMER = TRADLG
000158      IF(LL.NE.2) TIMER = 10.**TRADLG
000159      SUMOLD(JJ) = UMULT * TSURF(TIMER,UIN,1,LSURF,LL)
000160      480 SUM = SUMOLD(JJ)
000161      C
000162      500 IF(DRIVE.EQ.0) DRIVE = 1.E-6
000163      CONVOL = SUM/DRIVE
000164      MOLD = M
000165      RETURN
000166      END

```

Figure 52c. C O N V O L Listing (cont.)

2. UMULT is the stimulus value at the time the partition was created. For the case of the 9704 voltage surface, UMULT = 1 since the surface is a response rather than transfer function surface. UMULT is also a look-up variable for surface interpolation.
3. DTL is the time step interval that existed at the time steps being swept over by the running sum as the partition moves along the surface. The second column in the array is the logarithm of the value.
4. TSURFF is the properly scaled value of the surface z coordinate returned from SURFB by TSURF using TIMER as the x look-up variable and UMULT for the y look-up variable.

The contribution of a particular partition to the sum is the product

$$TEMP = TSURFF * UMULT * DTL(L,LL)$$

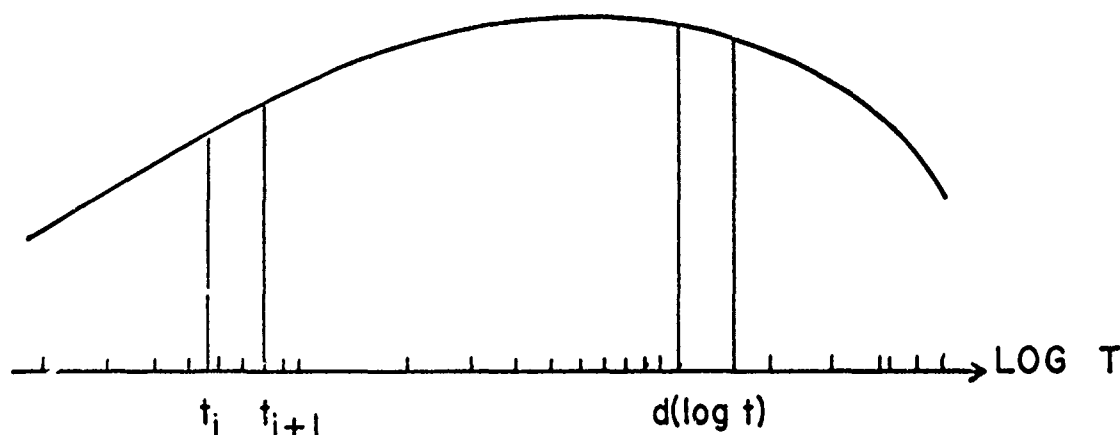
Every tenth time step during execution the running sums are printed for examination of the convolution process. When a particular term decreases below some value EPSLON, the multiplier UMULT is set zero for that term, so that the term is bypassed on subsequent summations.

The last section of CONVOL handles the contribution of the current time step to the total sum. Since the iteration process only affects this one term, after the first iteration pass only this section of CONVOL need be executed. The call which CONVOL sets up to TSURF is somewhat different here in that the call list contains a one rather than a three. This retrieves the z value of the surface rather than $\partial z / \partial x$, hence multiplication by DTL is not required. In the statement after 500, the value of the function CONVOL is computed as SUM/DRIVE, which casts the result into the sense of a transfer function, for use by FE et al.

One subtlety involved in CONVOL is the use of a doubly subscripted array for DTL which is dimensioned two in the second index, LL. LL indicates whether the time base is linear or logarithmic, taking the values one and two, respectively. If the surface time base is linear, then the linear DTL is required, whereas if

the basic mode of representation of the surface is logarithmic, then the time step

$$\text{TERM} = \text{UIN} \left. \frac{\partial T_u}{\partial \log t} \right|_{\text{UIN}(t_1), t-t_1} d(\log t_1)$$



Sketch Showing Equi-interval Convolution on a Log(time) Base

DTL must also be logarithmic since SURFB is returning $\partial z / \partial(\log t)$, rather than $\partial z / \partial t$. The setting of LL is done at the beginning of CONVOL by looking up the mode of the x axis of the surface in the array KSURFX.

9. FUNCTION SUBPROGRAM TSURF(TT,UU,II,L,LL)

TSURF is a relatively short program which interfaces CONVOL and SURFB, performing the task of rescaling the input and output information for internal compatibility. For example, the voltage transfer function surface of the 741, the vacuum cleaner, used milliseconds for the time axis and millivolts for the input voltage axis, while the SAP program uses nanoseconds for the time unit, and volts for the voltage unit. So TSURF multiplies the variable supplied by CONVOL, TT and UU, by SCALE for the surface number L, to obtain the proper values for use by SURFB.

TSURF finds the proper pointers for the X, Y, and Z arrays by look-up in the IPONTR array for the surface L. These pointers are set in the SURFB call to provide entry points in the array sequence because these same arrays are dimensioned differently in SURFB than in TSURF and MAIN. For example, Z is singly dimensioned

in the XYZ COMMON block, whereas it is triply dimensioned in SURFB.

Another major task performed by TSURF is decision making when the inputted look-up variables lie outside the domain of the surface. The actions taken have been kept simple, as can be seen by inspection of the listing of TSURF given in Figure 53.

The variable II selects the appropriate term in the ZA array returned by SURFB. For II = 1, ZA(1) = z, while for II = 3, ZA(3) = $\partial z / \partial x$.

10. FUNCTION SUBPROGRAM DAMAGE (J,IRRAD)

The routine DAMAGE is called by MAIN to calculate the permanent degradation of the device due to radiation exposure. A rather simple technique has been used which makes use of the 741 radiation voltage transfer function due to neutrons, TPHIVN. The isochronal cross-section occurring at $t = 875$ ms is used as the x axis look-up variable in a call to TSURF. This isochronal is approximately the same as that given in Figure 24b, bottom.

The accumulated dose is integrated by DAMAGE, as shown in the listing of Figure 54, using simple trapezoidal integration. The integrated dose is used as the y axis look-up variable for TSURF. Thus the surface interpolation returns the value of TPHIVN at the accumulated dose point along the isochronal. This is then divided by the lowest dose value of the function, TPHINO, to provide the ratio by which the maximum output has decreased. MAIN then uses this ratio, in the transfer function sense, to degrade the device output as calculated in the normal manner with electrical and radiation transient responses superimposed.

The variable J in the call list is the block number, and IRRAD indicates the irradiation type, one corresponding to Xrays, and two to neutrons. However, the same surface is used for the damage function (as a temporary expedient).

11. SUBROUTINE PARLEL(N,LELOCK,LMAX,FACTOR,CNOUT,VNOUT,VNIN,KPRINT,JPARAL)

This routine handles the parallel iteration algorithm, and its listing is given in Figure 55. The items in the call list represent the following:

- N is the input node number,
- LELOCK is an array which contains the block numbers for a given parallel sequence,

P ELT TSURF,1,720304, 74057 , 1

```

000001      FUNCTION TSURF(TT,UU,II,L,LL)
000002      INCLUDE XYZ,LIST
000003      DIMENSION ZA(8)
000004      LOGICAL ISYM
000005      DATA XHI6/250./
000006      DATA IT/6/
000007      FACTOR = 1.0
000008      T = TT * SCALF(L,1)
000009      U = UU * SCALF(L,2)
000010      IF(T.LT.XLOW(L)) GO TO 200
000011      XHI = XHIGH(L)
000012      IF(L.EQ.6) XHI = XHI6
000013      IF(T.GT.10.*XHI) GO TO 200
000014      IF(T.GT.XHI) T = XHI
000015      IF(U.LT.YLOW(L).AND.ISYM(L)) U = -U
000016      IF(U.LT.YLOW(L).AND..NOT.ISYM(L)) U = YLOW(L)
000017      IF(U.GT.YHIGH(L)) GO TO 300
000018      100 IP = IPONTR(L,1)
000019      JP = IPONTR(L,2)
000020      KP = IPONTR(L,3)
000021      IF(IP.EQ.0.OR.JP.EQ.0.OR.KP.EQ.0) GO TO 400
000022      CALL SURFB(X(IP),IXDIM(L),Y(JP),IYDIM(L),Z(KP),T,U,ZA,L)
000023      TSURF = ZA(II)*SCALE(L,3)*FACTOR
000024      IF(II.EQ.3) TSURF = TSURF * SCALE(L,1)
000025      RETURN
000026      200 TSURF = 0.0
000027      RETURN
000028      300 FACTOR = YHIGH(L)/U
000029      U = YHIGH(L)
000030      GO TO 100
000031      400 WRITE(IT,410)
000032      410 FORMAT(120H1  TERMINATED BY TSURF SINCE POINTER FOR SURFB IS ZERO.
000033      1  TSURF DATA CARD IS IN ERROR AND A NEEDED SURFACE IS NOT READ IN.)
000034      CALL EXIT
000035      END

```

Figure 53. T S U R F Listing

P ELT DAMAGE,1,720302, 47798 , 1

```

000001      FUNCTION DAMAGE(J,IRRAD)
000002      C AS PRESENTLY SET UP THIS ROUTINE REQUIRES THAT THE TPHIVN SURFACE
000003      C OF THE 741 BE READ IN AT EXECUTION TIME BY MAIN. LSURF = 5, ZVN741
000004      INCLUDE SAP,LIST
000005      INCLUDE TIME,LIST
000006      DATA TLOOK/5.75/,TPHINO/1.32/,DOSEN/.001/,DOSEX/.001/
000007      C
000008      INDEX = ITABLE(J,1)
000009      IF(KSURF(INDEX,IRRAD*3).EQ.0) GO TO 300
000010      C
000011      T = TIME(MPRIME,1)
000012      IF (.NOT.(T.GT.TOLD)) GO TO 100
000013      DOSEN = DOSEN + DT*10.**PHINL
000014      DOSEX = DOSEX + DT*10.**PHIXL
000015      DOSENL = ALOG10(DOSEN)
000016      DOSEXL = ALOG10(DOSEX)
000017      C
000018      100 GO TO (120,140), IRRAD
000019      120 DAMAGE = DOSEXL * TSURF(TLOOK,DOSEXL,1,5,1)/TPHINO
000020      GO TO 200
000021      140 DAMAGE = DOSENL * TSURF(TLOOK,DOSEN,1,5,1)/TPHINO
000022      200 TOLD = T
000023      RETURN
000024      300 DAMAGE = 1.0
000025      RETURN
000026      END

```

Figure 54. D A M A G E Listing

• ELT PARLEL.1,720302, 47809 • 1

```

000001      SUBROUTINE PARLEL(N,LBLOCK,LMAX,FACTOR,CNOUT,VNOUT,VNIN,CNIN,      PAR
000002      1 KPRINT,JPARAL)
000003      C
000004      C      DR. R. G. STEWART, AREA CODE 415, 941-6699
000005      C      STEWART RESEARCH ENTERPRISES, 23376 BELVOIR DR., LOS ALTOS, CALIF.
000006      C
000007      C      PARLEL HANDLES BALANCING OF CURRENTS IN PARALLEL CONNECTED BLOCKS.
000008      C
000009      INCLUDE SAP,LIST
000010      COMMON/PARALL/LITER,JPARL
000011      DIMENSION LBLOCK(IX,IX),LMAX(IX),FACTOR(IX),CNOUT(IX),VNOUT(IX)
000012      1      ,VNIN(IX),CNIN(IX),DELTAV(IX),JPARAL(IX),FEE(IX),FIT(IX)
000013      2      ,FEOLD(IX),DUDJ(IX),VAVRAG(IX)
000014      DATA IN/5,IT/6,KMOST/ 2/
000015      DATA VTEST/0.02/,CTEST/0.02/,LIMIT/ 4/
000016      C
000017      COMPUTING CURRENTS IN PARALLEL BRANCHES.
000018      C
000019      VTOTAL = VNIN(N)
000020      CTOTAL = CNIN(N)
000021      KSTATE = KROSS
000022      KROSS = 2
000023      C
000024      C
000025      C      K IS THE PARALLLEL ITERATION DO, L STEPS ALONG BRANCHES.
000026      C
000027      LHI = LMAX(N)
000028      DO 20 LL = 1, LHI
000029      LB = LBLOCK(N,LL)
000030      20 VIN(LB,1) = VTOTAL
000031      C
000032      C      ITERATING TO FIND NEW CURRENT FACTORS.
000033      C
000034      DO 100 K = 1, KMOST
000035      CSUM = 0.0
000036      LITER = K
000037      DO 30 LL = 1, LHI
000038      LB = LBLOCK(N,LL)
000039      CIN(LB,1) = CTOTAL * FACTOR(LB)
000040      30 CSUM = CSUM + CIN(LB,1)
000041      C
000042      FRACT = CSUM/CTOTAL
000043      DO 40 LL = 1, LHI
000044      LB = LBLOCK(N,LL)
000045      40 CIN(LB,1) = CIN(LB,1)/FRACT
000046      C
000047      C      NOW KIRCHOFFS CURRENT LAW IS SATISFIED AT INPUT NODE.
000048      C      NEXT PROPAGATE THRU BRANCHES TO FIND OUTPUT VOLTAGE USING
000049      C      CROSSED VOLTAGE TRANSFER FUNCTIONS.
000050      C
000051      VSUM = 0.0
000052      DO 60 LL = 1, LHI
000053      LB = LBLOCK(N,LL)
000054      JPARL = JPARAL(N)
000055      ITRSAV = ITERAT
000056      NITER = 0
000057      C
000058      C      SELF-CONSISTENCY ITERATION LOOP

```

Figure 55a. P A R L E L Listing

```

000059      C
000060      50 NITER = NITER + 1
000061          ITERAT = ITRSAV * NITER
000062          VOLD = VOUT(LB,1)
000063          COLD = COUT(LB,1)
000064          FEE(LB) = FE(LB)
000065          FII(LB) = FI(LB)
000066          VOUT(LB,1)=FEE(LB)*CIN(LB,1)
000067          COUT(LB,1)=FII(LB)*VIN(LB,1)
000068          TESTV = ABS(1.0-VOUT(LB,1)/VOLD)
000069          TESTC = ABS(1.0-COUT(LB,1)/COLD)
000070          IF(.NOT.(ABS(VOLD).GT.0.)) TESTV = 0.0
000071          IF(.NOT.(ABS(COLD).GT.0.)) TESTC = 0.0
000072          IF((TESTV.GT.VTEST.OR.TESTC.GT.CTEST).AND.NITER.LE.LIMIT) GO TO 50
000073          ITERAT=ITRSV
000074      C
000075      C      CONSISTENCY ACHIEVED
000076      C
000077      60 VSUM = VSUM + VOUT(LB,1)
000078      VAVRAG(N) = VSUM/LMAX(N)
000079      IF(ABS(VAVRAG(N)).GT.15.0) VAVRAG(N) = 15.0 * ISIGN(1,VAVRAG(N))
000080      KOUNT = 0
000081      C
000082      C      FINDING NEW CURRENT SPLITTING FACTORS.
000083      DO 80 LL = 1, LHI
000084          LB = LBLOCK(N,LL)
000085          DELTAV(LB) = VOUT(LB,1)-VAVRAG(N)
000086          DUDI(LB) = (FEE(LB)-FEOLD(LB))/(CIN(LB,1)-CIN(LB,2))
000087          IF(K.EQ.1) DUDI(LB) = 0.0
000088          DELTAI = DELTAV(LB)/(FEE(LB) + CIN(LB,1)*DUDI(LB))
000089          FEOLD(LB) = FEE(LB)
000090          CIN(LB,2) = CIN(LB,1)
000091          FACTOR(LB) = (CIN(LB,1)-DELTAI)/CTOTAL
000092          IF(ABS(DELTAV(LB)) - .00001) 76, 76, 80
000093      76 KOUNT = KOUNT + 1
000094      80 CONTINUE
000095      C
000096      IF(KPRINT) 82,90,82
000097      82 DO 84 LL = 1, LHI
000098          LB = LBLOCK(N,LL)
000099      84 WRITE(IT,86) K,LB,DUDI(LB),CIN(LB,1),FEE(LB),FII(LB),DELTAV(LB)
000100          1,FACTOR(LB),VOUT(LB,1),COUT(LB,1)
000101      86 FORMAT(3H K=,I3,3H L=,I2,6H DUDI=,G9.4,5H CIN=,G9.4,5H FEE=,G9.4,
000102          15H FII=,G9.4,8H DELTAV=,G9.4,8H FACTOR=,G9.4,6H VOUT=,G9.4,6H COUT
000103          2=,G9.4)
000104          LB = LBLOCK(N,1)
000105          WRITE(IT,88) VAVRAG(N),CTOTAL,VIN(LB,1)
000106      88 FORMAT(10X,7HVAVRAG=,G9.4,8H CTOTAL=,G9.4,5H VIN=,G9.4)
000107      C
000108      90 IF(KOUNT - LMAX(N)) 100, 200, 200
000109      COMPLETION OF ITERATION.
000110      100 CONTINUE
000111      C
000112      C
000113      200 CSUM = 0.0
000114      DO 220 LL = 1, LHI
000115          LB = LBLOCK(N,LL)
000116      220 CSUM = CSUM + COUT(LB,1)
000117          CNOUT(N) = CSUM
000118          VNOUT(N) = VAVRAG(N)
000119
000119      KROSS = KSTATE
000120      JPARL = 0
000121      IF(KPRINT) 400, 300, 400
000122      300 RETURN
000123      400 WRITE(IT,410)K,N,CNOUT(N),N,VNOUT(N)
000124      410 FORMAT( 6H AFTER,I4,19H ITERATIONS CNOUT(,I2,2H)=,G10.4,
000125          1 8H, VNOUT(,I2,2H)=,G10.4, /)
000126      RETURN
000127      END

```

Figure 55b. P A R L E L Listing (cont.)

- LMAX is the number of blocks in parallel in the sequence,
- FACTOR is the current splitting factor for a given block,
- CNOUT and VNOUT are the output voltages and currents returned to MAIN,
- VNIN and CNIN are the voltage and current at the input node as supplied by MAIN,
- KPRINT controls printing of the results of the parallel iteration, and
- JPARAL is an array created by SETUP which indicates whether a node requires the use of PARLEL; JPARAL = 1 implies yes.

The actions taken in PARLEL are indicated by the COMMENT cards in the listing of Figure 55. The input current is split into the blocks, and then propagated using the crossed voltage transfer functions. Since the Univac 1108 sets the result of dividing any number by zero equal to zero, logical IF tests set the variables TESTV and TESTC equal to zero if the denominators VOLD or COLD vanish. Otherwise the following consistency test would automatically drop through when it should not.

After self-consistency is attained, the new current splitting factors are calculated, the deviations from the floating average output voltage determined, and the iteration continued if the deviations exceed the test value. After the iteration loop is completed, the voltage and current at the output node are computed and returned to MAIN.

12. SUBROUTINE PLOTEM(KIND,VNIN,CNIN,VNOUT,CNOUT,NMAX,INITAL)

PLOTEM performs the tasks of reading plot title and control cards, and loading information into the plot arrays. The listing of PLOTEM is given in Figure 56. The entries in the call list perform the following tasks:

- KIND controls action taken - if negative, data cards are read; if zero, points are entered into plot arrays; if positive, PLOT6 is called to force plotting,
- VNIN, CNIN, VNOUT, and CNOUT are the arrays containing node voltages and currents as supplied by MAIN,
- NMAX is the final output node number,

ELT PLOTEM,1,720302, 47811 . 1

```

000001      SUBROUTINE PLOTEM(KIND,VNIN,CNIN,VNOUT,CNOUT,NMAX,INITAL)
000002      INCLUDE SAP,LIST
000003      INCLUDE TIME,LIST
000004      DIMENSION VNIN(IX),CNIN(IX),VNOUT(IX),CNOUT(IX)
000005      PARAMETER IDIM=100,JDIM=6
000006      DIMENSION X(IDIM,JDIM),Y(IDIM,JDIM),XX(IDIM,JDIM),YY(IDIM,JDIM)
000007      1,LABEL7(48),ILABEL(6,8),IPOINT(7),TITLE(12),IORIGN(7),SKALE(7)
000008      2,LAWG(7),NOXY(JDIM),IPAR(2)
000009      LOGICAL INITAL
000010      DIMENSION XSCALE(6),XSKALE(6),JPLOT(4),JIPLOT(4)
000011      DATA (XSCALE(I),I=1,6)/6HLOG(11,6HNANOSE,6HMICROS,6HMILLIS,
000012      16HSECOND,6H /, (XSKALE(I),I=1,6)/1.,1.,1.E-3,1.E-6,1.E-9,1.0/
000013      DATA ICURNT/1HI/,IVOLT/1HV/,IN/5/,IT/6/
000014      DATA ICRAD/2HIR/,IVRAD/2HVR/
000015      C
000016      5 FORMAT(8X,12A6)
000017      6 FORMAT(A5,48A1,I3,3X,E12.5,3X,3X,I3)
000018      7 FORMAT(A1,4X,8A6,I3,3X,E12.5,6X,I3)
000019      C
000020      IF(KIND) 200,400,800
000021      C
000022      CALLIN PLOT6 HERE WILL READ IN ONE DATA CARD. IF PLOTIT APPEARS IN THE
000023      C FIRST 6 COLUMNS, THEN CALCOMP PLOT IS MADE. IF NOT, ONLY PRINTER
000024      C PLOTS. THE PLOT TAPE IS ALSO INITIALIZED IF REQUIRED.
000025      C
000026      200 IF(INITIAL) CALL PLOT6(X,Y,NOXY,IPAR,IDIM,JDIM,-1,LABEL7,ILABEL
000027      1,IPOINT,TITLE,IORIGN,SKALE,LAWG,NUMBER,SIZE,IPAGE)
000028      C
000029      C      INITIALIZATION OF ROUTINE
000030      C      INITIAL =.FALSE.
000031      C
000032      C      READING THE PLOT ASSIGNMENT CARD. THIS DETERMINES WHICH BLOCK OUTPUT
000033      C      VOLTAGES OR CURRENTS ARE PLOTTED. VNIN(1) AND VNOUT(NMAX) ARE ALWAYS
000034      C      PLOTTED. NUMJ IS THE NUMBER OF ADDITIONAL BLOCK VOLTAGES AND CURRENTS.
000035      C      WHERE NUMJ.LE.4 .
000036      C
000037      READ(IN,26) NUMJ,(JPLOT(I),JIPLOT(I),I=1,4),TSCALE
000038      26 FORMAT(13,4(I3,A2),6X,A6)
000039      WRITE(IT,28)NUMJ,(JPLOT(I),JIPLOT(I),I=1,4),TSCALE
000040      28 FORMAT(46H THE PLOT ASSIGNMENT CARD AS READ IN SAYS.../,13,4(I3,
000041      1A2),6X,A6 /)
000042      NUMBER = NUMJ + 2
000043      KARDS = 0
000044      READ(IN,5) (TITLE(J),J=1,12)
000045      WRITE(IT,5) (TITLE(J),J=1,12)
000046      READ(IN,6) SIZE,(LABEL7(J),J=1,48),IORIGN(7),SKALE(7),LAWG(7)
000047      READ(IN, 7) IPOINT(1),(ILABEL(1,J),J=1,8),IORIGN(1),SKALE(1),LAWG(1
000048      1)
000049      DO 30 I = 1,6
000050      IF(XSCALE(I).NE.TSCALE) GO TO 30
000051      XSC = XSCALE(I)
000052      GO TO 32
000053      30 CONTINUE
000054      XSC = 1.0
000055      32 DO 564 I = 2, NUMBER
000056      564 READ(IN, 7) IPOINT(I),(ILABEL(1,J),J=1,8),IORIGN(I),SKALE(I),
000057      1 LAWG(I)
000058      RETURN

```

Figure 56a. P L O T E M Listing

```

000059      400 IF(M - 1) 404, 404, 410
000060      404 NUMPTS = 0
000061          NOPTS = 0
000062          MODPTS = IFIX(MMAX/100) + 1
000063      410 NUMPTS = NUMPTS + 1
000064          TIM = TIME(MPRIME,MODE) * XSC
000065          DO 420 I = 1, NUMBER
000066      420 X(NUMPTS,I) = TIM
000067          Y(NUMPTS,2) = VNIN(1)
000068          Y(NUMPTS,1) = VNOUT(NMAX)
000069          IF(NUMBER.LE.2) GO TO 500
000070          DO 460 I = 1, NUMJ
000071              II = JPLLOT(I)
000072              IF(JIPLLOT(I).EQ.ICRAD) Y(NUMPTS,I+1) = COUT(II,3)
000073              IF(JIPLLOT(I).EQ.IVRAD) Y(NUMPTS,I+2) = VOUT(II,3)
000074              IF(JIPLLOT(I).EQ.ICURNT) Y(NUMPTS,I+2) = COUT(II,1)
000075              IF(JIPLLOT(I).EQ.IVOLT) Y(NUMPTS,I+2) = VOUT(II,1)
000076      460 CONTINUE
000077      500 IF(MOD(NUMPTS,MODPTS)) 560,520,560
000078          NOPTS = NOPTS + 1
000079          DO 540 I = 1, NUMBER
000080      540 XX(NOPTS,I) = TIM
000081          YY(NOPTS,2) = VNIN(1)
000082          YY(NOPTS,1) = VNOUT(NMAX)
000083          IF(NUMBER.LE.2) GO TO 560
000084          DO 550 I = 1, NUMJ
000085              II = JPLLOT(I)
000086              IF(JIPLLOT(I).EQ.ICRAD) YY(NOPTS,I+2) = COUT(II,3)
000087              IF(JIPLLOT(I).EQ.IVRAD) YY(NOPTS,I+2) = VOUT(II,3)
000088              IF(JIPLLOT(I).EQ.ICURNT) YY(NOPTS,I+2) = COUT(II,1)
000089              IF(JIPLLOT(I).EQ.IVOLT) YY(NOPTS,I+2) = VOUT(II,1)
000090      550 CONTINUE
000091      560 IF(NUMPTS.NE.100.AND.KIND.EQ.0) RETURN
000092      570 DO 580 I = 1, NUMBER
000093      580 NOXY(I) = NUMPTS
000094          CALL PCNT(IPAGE)
000095          IPAGE = IPAGE + 1
000096          CALL PLOT6(X,Y,NOXY,IPAR,IDIM,JDIM,KARDS,LABEL7,ILABEL,IPOINT
000097      1,TITLE,IORIGN,SKALE,LAWG,NUMBER,SIZE,IPAGE)
000098          NUMPTS = 0
000099          RETURN
000100      800 IF(KIND - 2) 810,820,900
000101      810 IF(NUMPTS.LT.3) RETURN
000102          GO TO 570
000103      820 IF(MMAX.LT.103) RETURN
000104          DO 840 I = 1, NUMBER
000105      840 NOXY(I) = NOPTS
000106          CALL PCNT(IPAGE)
000107          IPAGE = IPAGE + 1
000108          CALL PLOT6(XX,YY,NOXY,IPAR,IDIM,JDIM,0,LABEL7,ILABEL,IPOINT
000109      1,TITLE,IORIGN,SKALE,LAWG,NUMBER,SIZE,IPAGE)
000110          NOPTS = 0
000111          RETURN
000112      900 CALL PLOT6(X,Y,NOXY,IPAR,IDIM,JDIM,-2,LABEL7,ILABEL,IPOINT
000113      1,TITLE,IORIGN,SKALE,LAWG,NUMBER,SIZE,IPAGE)
000114      C      THIS IS THE NORMAL TERMINATION OF THE ENTIRE SAP.
000115          CALL EXIT
000116          RETURN
000117          END

```

Figure 56b. P L O T E M Listing (cont.)

- INITIAL = .TRUE. causes PLOT6 to read one data card; if PLOTIT is contained in the first six columns, the plot tape is initialized, and machine plots will be made.

The FORMATS that read the plot title and control cards are numbers 5, 6, 7, 26, and 28. The COMMENT cards in the listing indicate the function of the plot assignment card. JPLOT is the block number whose output is to be plotted, JIPLLOT indicates whether the voltage, current, radiation induced voltage, or the radiation induced current is to be plotted, being tested against the entities in the last two DATA cards in the listing. TSCALE sets the scale factor used on the time axis of the plots, since nanoseconds is not the best unit if time intervals measured in seconds are being treated. SCALE is tested against the spelling indicated in the first DATA card.

The normal values for the quantities read in by FORMATS 6 and 7 are: SIZE = SMALL, IORIGN = -1, SCALE = -1.0, LAWG = 0 . PLOT6 can make large or small printer plots. If two page printer plots are desired, set SIZE = LARGE. PLOT6 can plot up to six dependent variables versus one independent variable in one plot.

The final completion of a run occurs in PLOTTEM with a CALL EXIT, after the call to PLOT6 with -2 in the seventh list position which causes proper plot tape termination.

SECTION VI

RESULTS COMPUTED BY SAP USING TRANSFER FUNCTION SURFACES

The concepts discussed in Section II, and the surface fitting techniques developed in Section IV, provide the tools needed to evaluate the usefulness of the nonlinear surface convolution method for simulating the effects of electrical and radiation stimuli on integrated circuits. The results presented in this section are new in the research sense. No prior efforts existed that treated the radiation effects on devices in this way. The results show that the ideas have promise in the sense of an approximation having an engineering level of precision. Further, the execution time on the Univac 1108 is reasonably low -- typically 23 seconds for 100 time steps. The core storage available on the 1108 of 65K decimal was sufficient to contain the SAP program, all the plot routines, and six surfaces without segmenting. If the plot routines had been segmented, since they are needed only at the end, at least ten or more additional surfaces could have been contained in core. Thus it appears that the method does indeed have sufficient computational characteristics to make genuine its usefulness for real applications.

1. 741 OP AMP - ELECTRICAL RESULTS

The computed results due to application of a 1.0 mv pulse to the 741 T_0 surface, Figure 20, are shown in Figure 57. The input pulse is shown in Figure 57a, the input and output voltages and the output current in Figure 57b, and the details of the leading and trailing edges of the pulse in Figure 57c and d. The plot points on the plots occur at twice the interval of computation, otherwise no line could be seen. The number of time steps was 100. Because the input pulse waveform is almost a square pulse, the convolution process amounts to almost an isovoltage sampling of the surface in this case, except for two points on the rising and falling edges which move back at half the peak pulse level and sample the surface at the 500 microvolt portion.

To ascertain the sine wave response obtained with the T_0 surface, six runs were made varying the frequency and the amplitude of the input waveform. These runs were made after the self-consistency loop had been inserted into MAIN, and the optimization of DRIVER's convergence tests was still under way. The frequencies were chosen

SAP DETERMINATION OF 741 RESPONSE

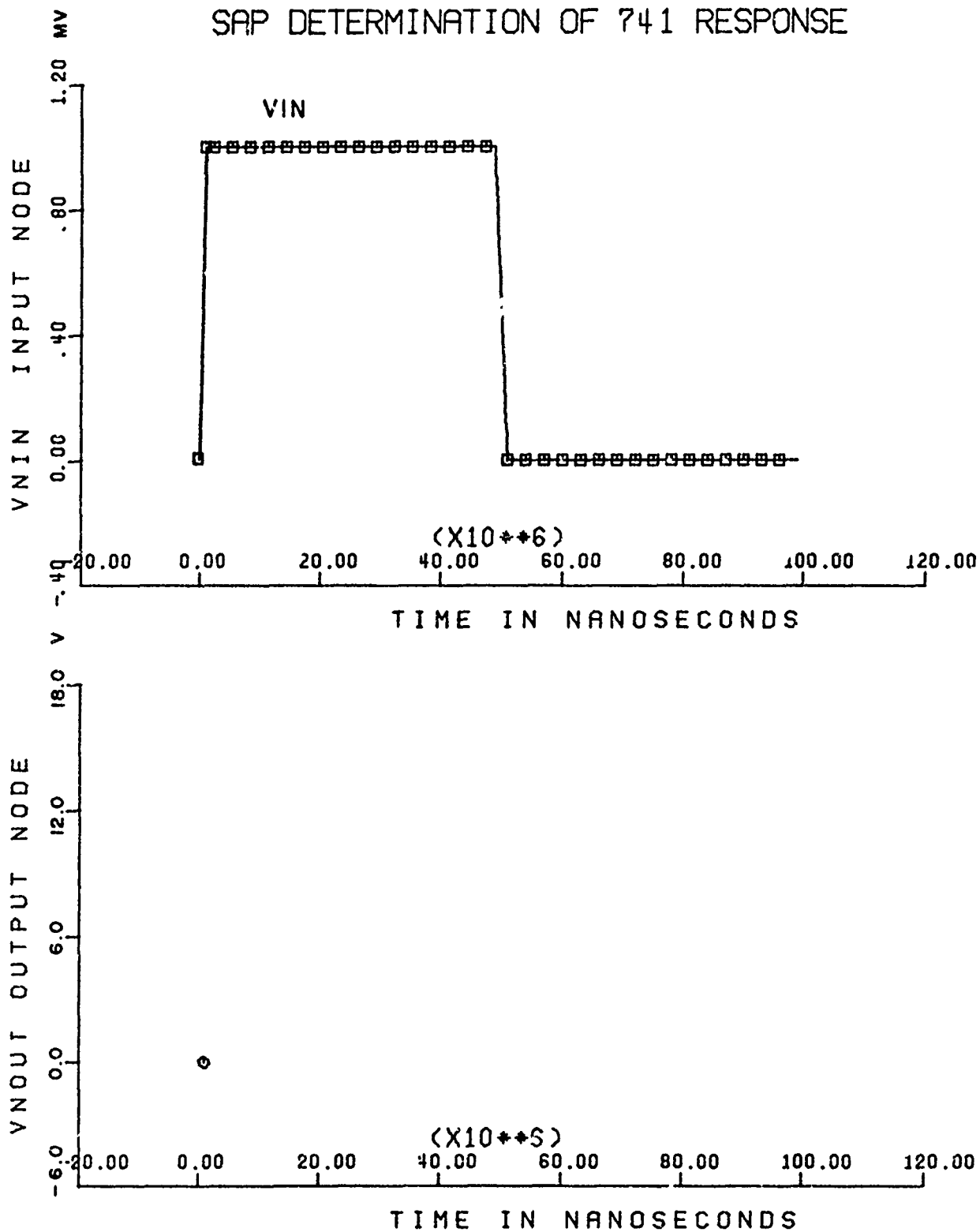


Figure 57a. 741 Op Amp Response to 50 ms Step Pulse.
 Input Waveform has 1 ms Rise and Fall Times.
 See Figure 57b for VOUT waveform.

SAP DETERMINATION OF 741 RESPONSE

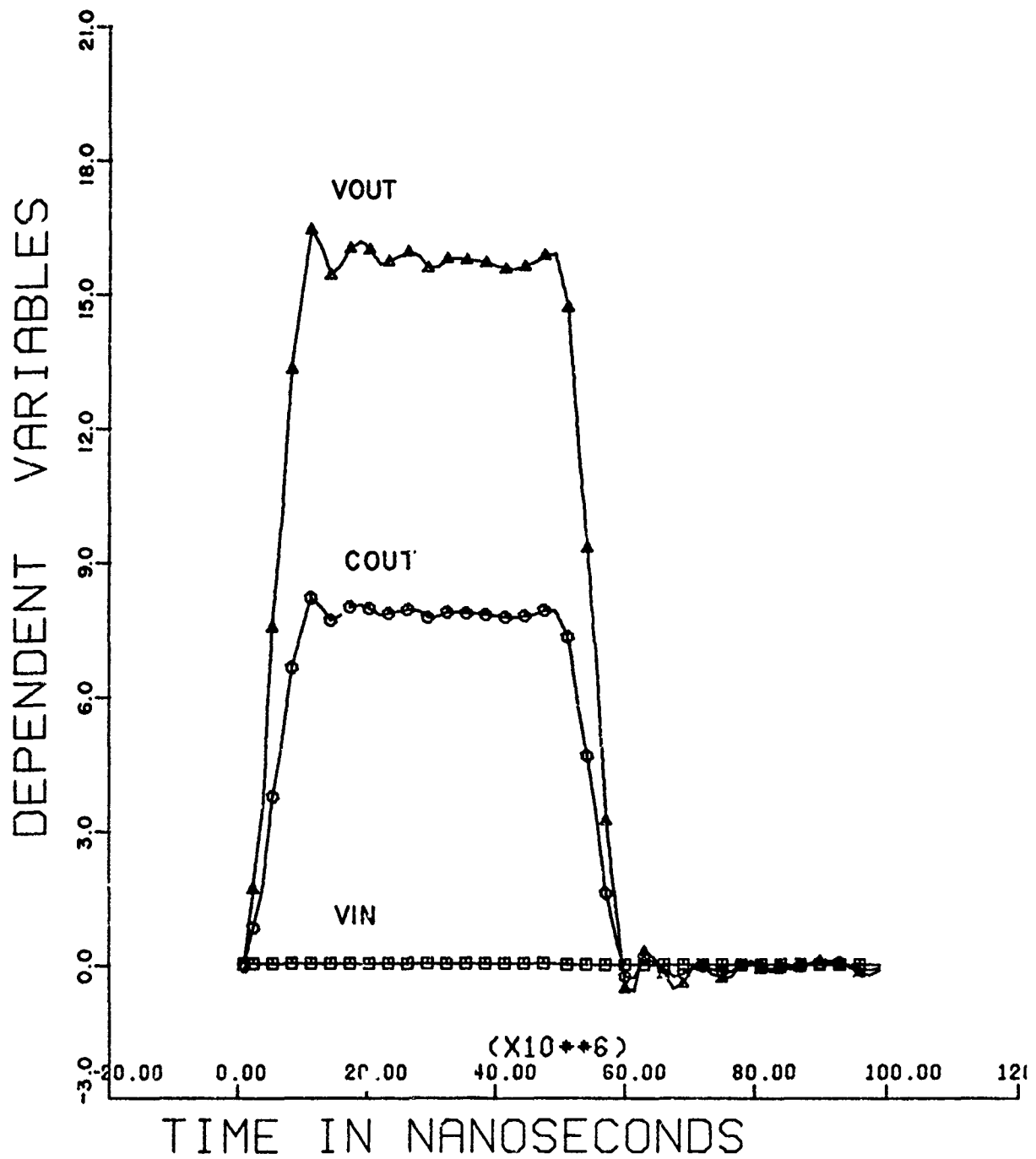


Figure 57b. 741 Op Amp Response to Step Pulse, VIN and VOUT in volts, COUT in ma.

SAP DETERMINATION OF 741 RESPONSE

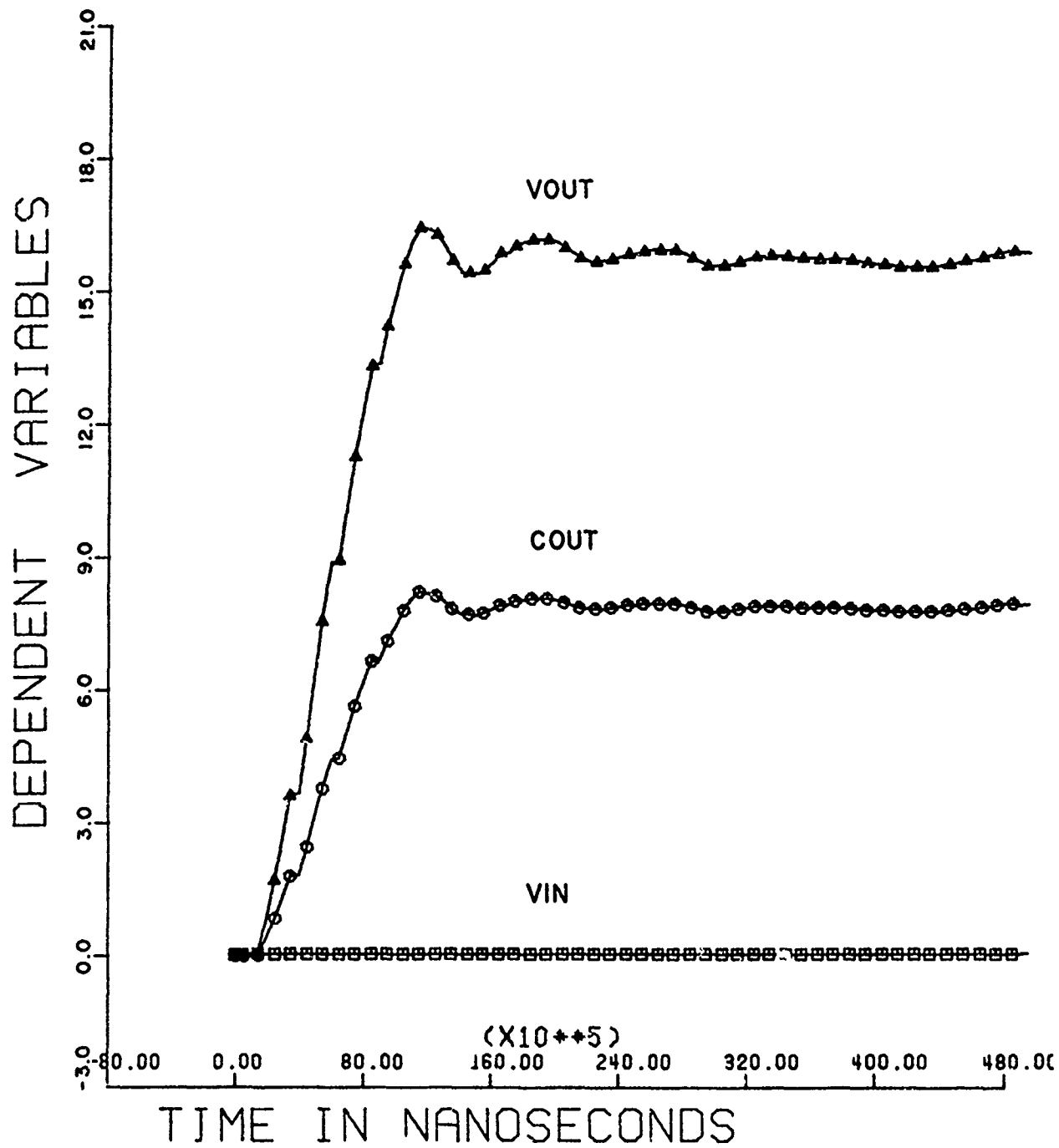


Figure 57c. 741 Op Amp Response to Step Pulse.
Leading Edge Portions of Waveforms.

SAP DETERMINATION OF 741 RESPONSE

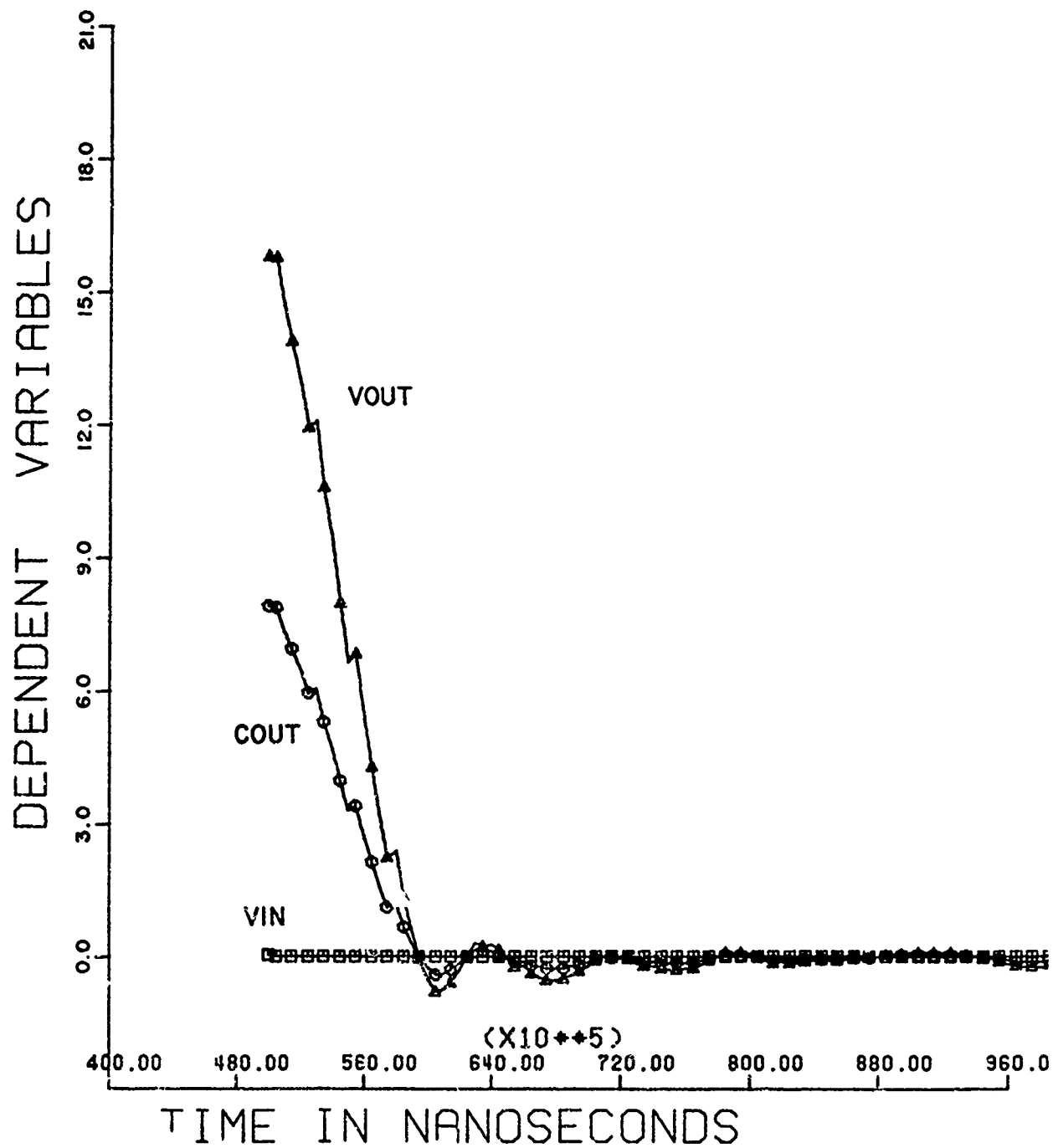


Figure 57d. 741 Op Amp Response to Step Pulse.
Trailing Edge Portions of Waveforms.

to be about an order of magnitude above, below, and approximately at the reciprocal of the open loop time constant. The peak input waveform amplitudes were chosen to lie in the linear and nonlinear operating regions. Since the stimulus is a sine wave, partitions are driven back along the surface at input voltages which range from plus to minus the peak amplitude of the sine wave. Thus a sine wave input reaching into the nonlinear region also samples the linear region of the device characteristic. This is the same situation as for a real device.

The machine plots of the six runs are contained in Figure 58. The summary of the conditions applying for the runs are:

Fig	Run	VIN	Phase	Freq	Comments
58a	335	0.2mv	0°	2 Hz	
58b	336	2.0		2	Time step too large for surface.
58c	337	2.0	37°	20	Lost iterate twice but came back in.
58d	338	0.2		20	
58e	340	0.2	78°	200	Note correct form of transient.
58f	341	2.0		200	" " " " "
58g	349	2.0		200	Return of 341 with CONVOL bug corrected; note only slight effect on results.

The phase given is that of the output lagging the input. The change of the phase with frequency yields $f_{\text{corner}} \approx 20$ Hz. In the case of the 2 Hz runs, the step size was so large that only two partitions were sampling the surface at any moment of the calculation. Now since the effective time constant of the T_e surface for the 741 decreases with increased excitation (until slew rate is reached) that small a number of sampling partitions yields an unsatisfactory result as can be seen in Figure 58b, for the 2.0 mv nonlinear excitation level. Compare that with the result contained in Figure 58a for a linear excitation whose effective time constant is about an order of magnitude greater, and it is seen that a fairly smooth output function is computed for the same time step size.

The curve for the 2.0 mv, 20 Hz, run shown in Figure 58c is interesting in that it demonstrates an interesting aspect of convolutional calculation methods, namely at about 150 milliseconds the iteration process lost stability. Yet, in two time steps the iteration routine DRIVER regained stability with only a very small effect remaining from the prior iteration loss. This results from the fact that convolution is the superposition of many terms, and unless a faulty iteration yields an excessively large term, the large number of the good terms minimizes the effect of

741 SINE WAVE RESPONSE RUN #335

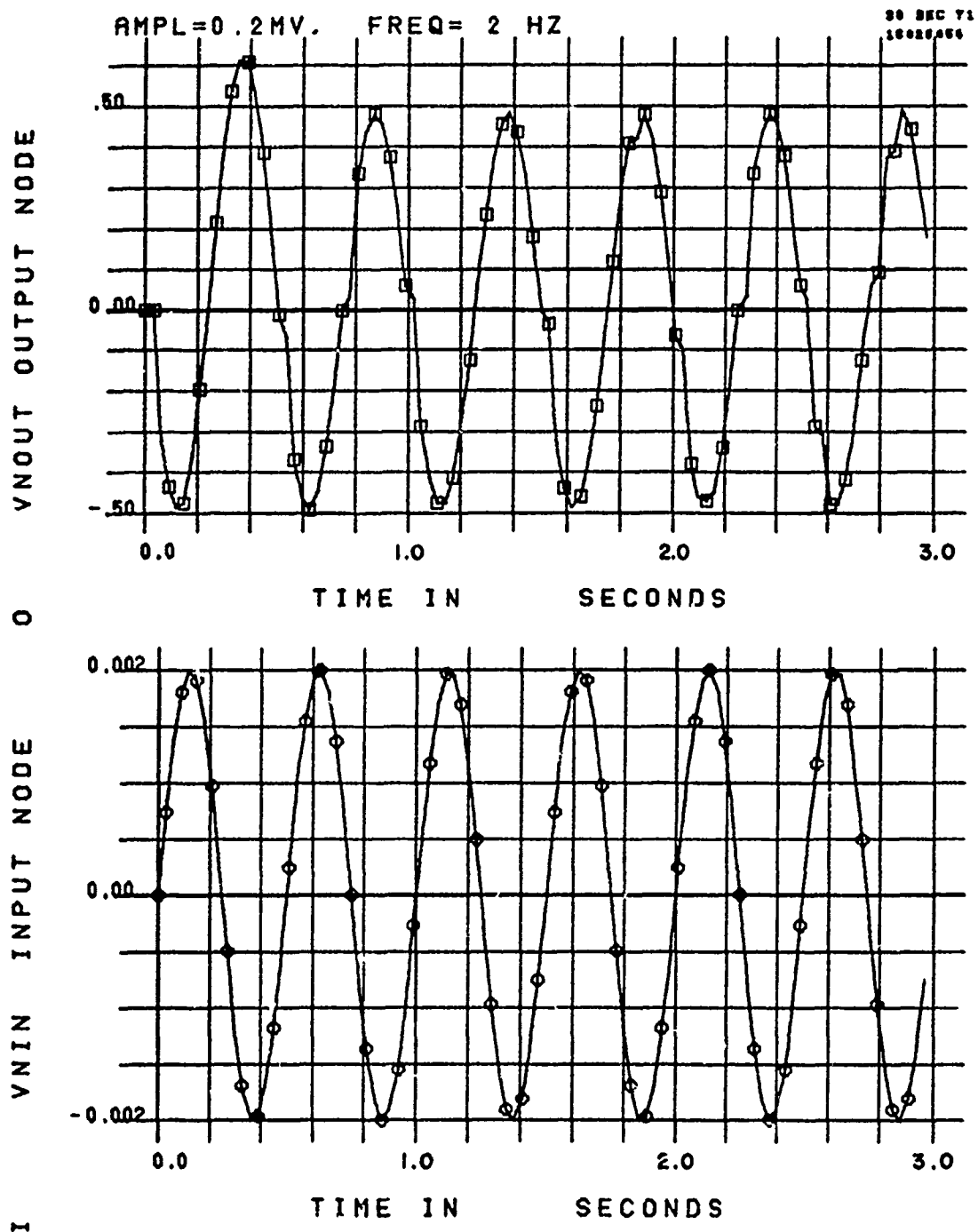


Figure 58a. 741 Response to 2 Hz Sine Wave, Op Amp Input Amplitude of 0.2 mv is One-Tenth Input Node Voltage, VNIN.

741 SINE WAVE RESPONSE RUN #336

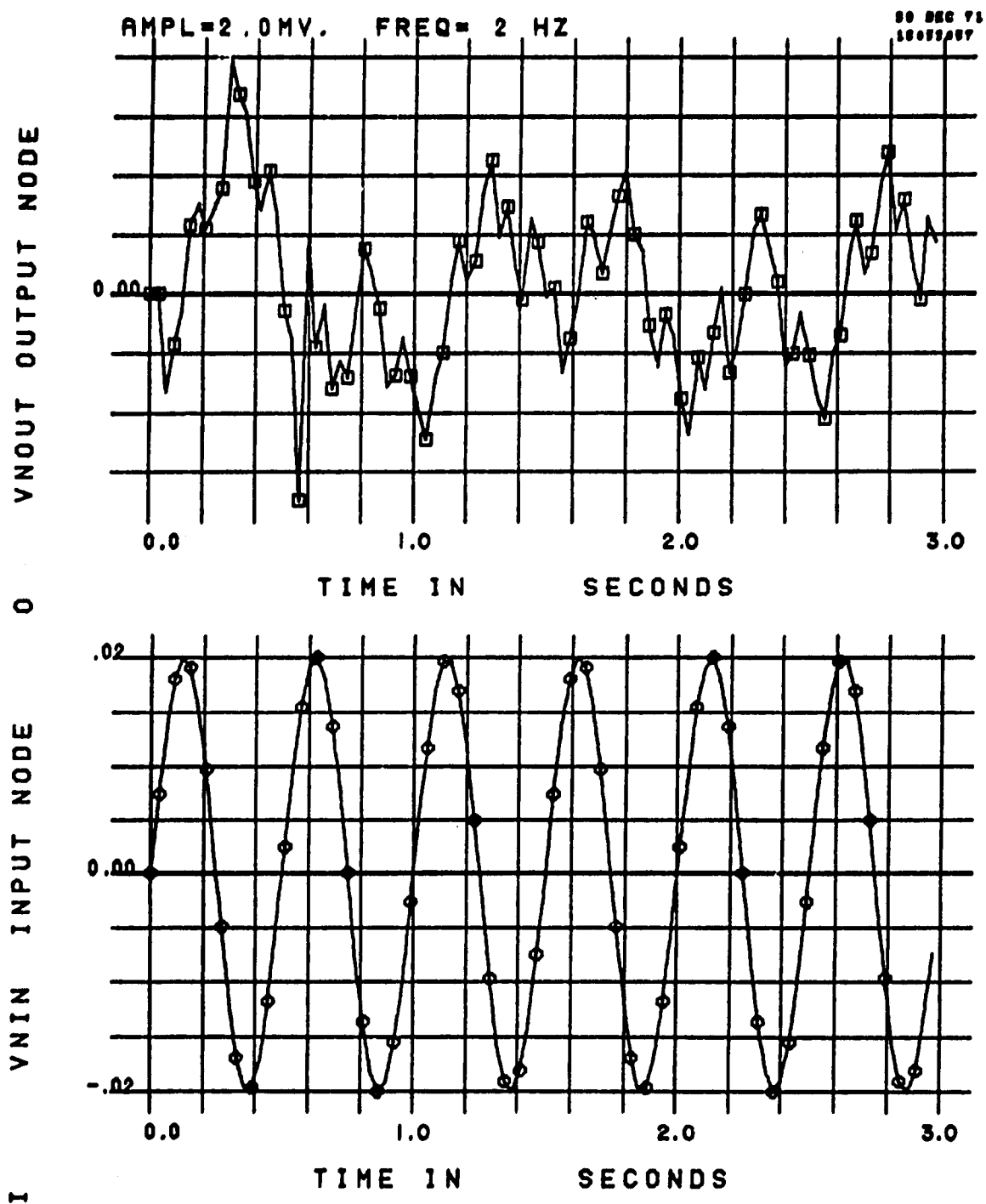


Figure 58b. 741 Response to 2 Hz, 2 mv Sine Wave

741 SINE WAVE RESPONSE RUN #337

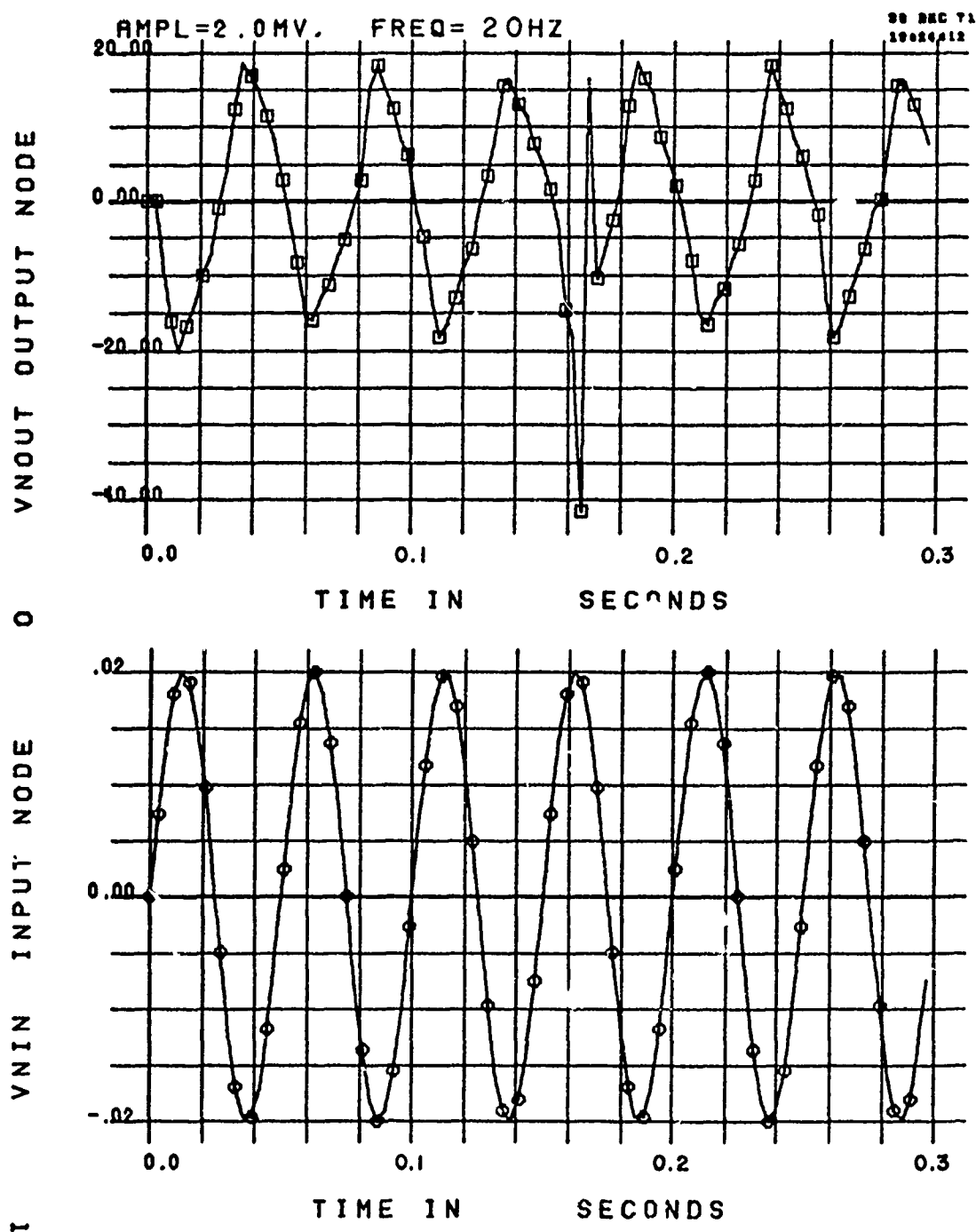


Figure 58c. 741 Response to 20 Hz, 2.0 mv Sine Wave

741 SINE WAVE RESPONSE RUN #338

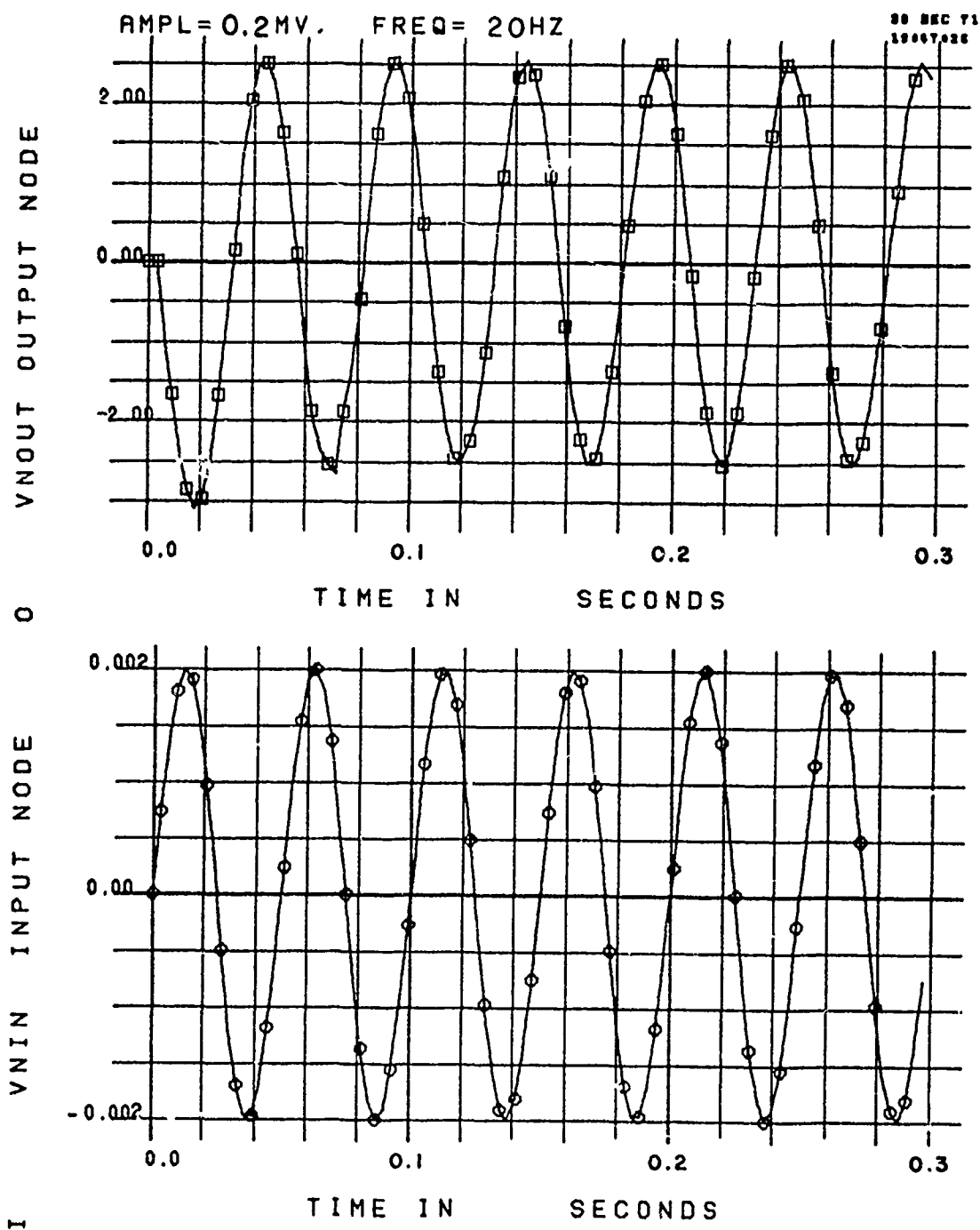


Figure 58d. 741 Response to 20 Hz, 0.2 mv Sine Wave

741 SINE WAVE RESPONSE RUN #340

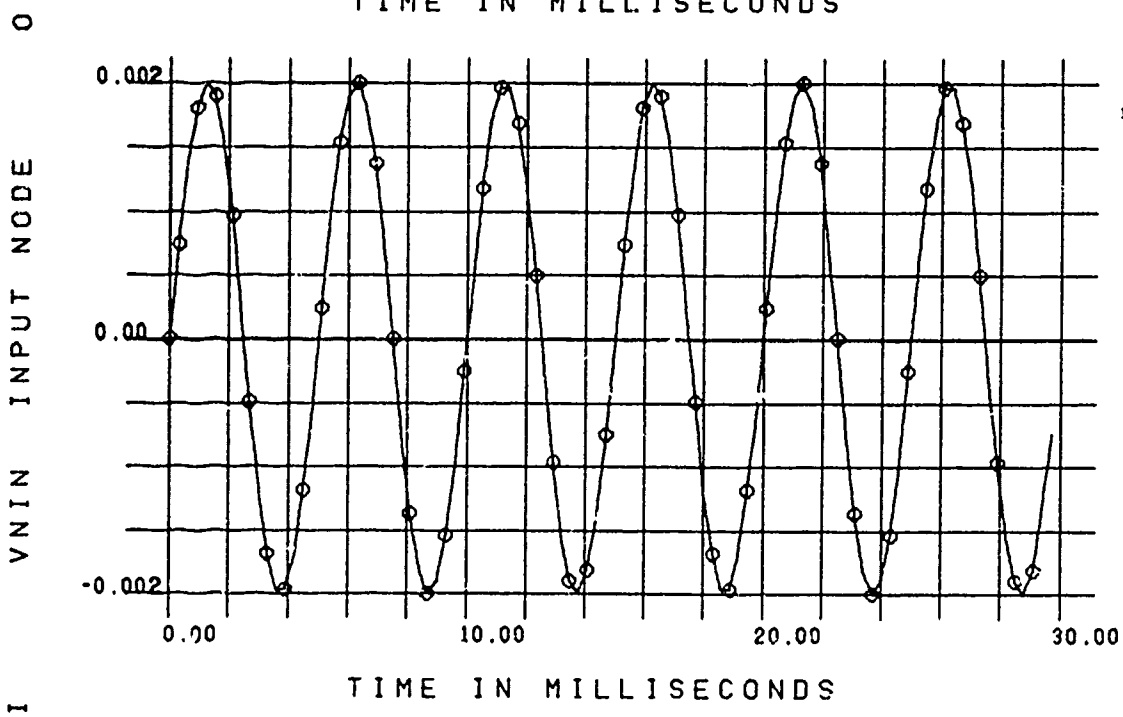
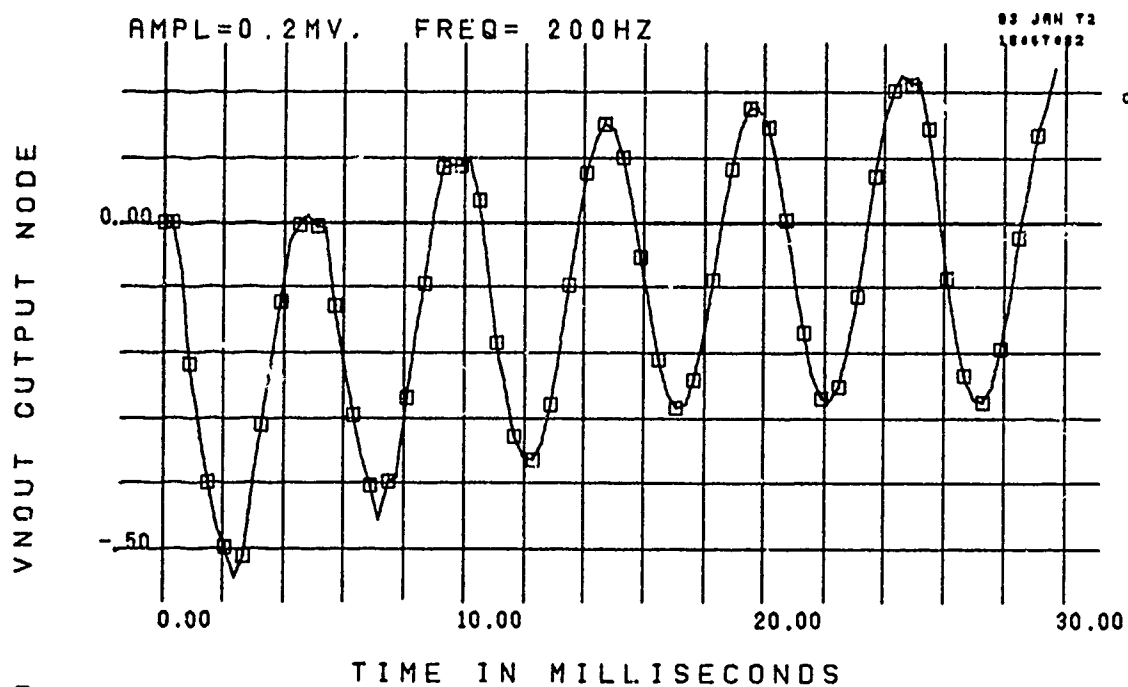


Figure 58e. 741 Response to 200 Hz, 0.2 mv Sine Wave

741 SINE WAVE RESPONSE RUN #341

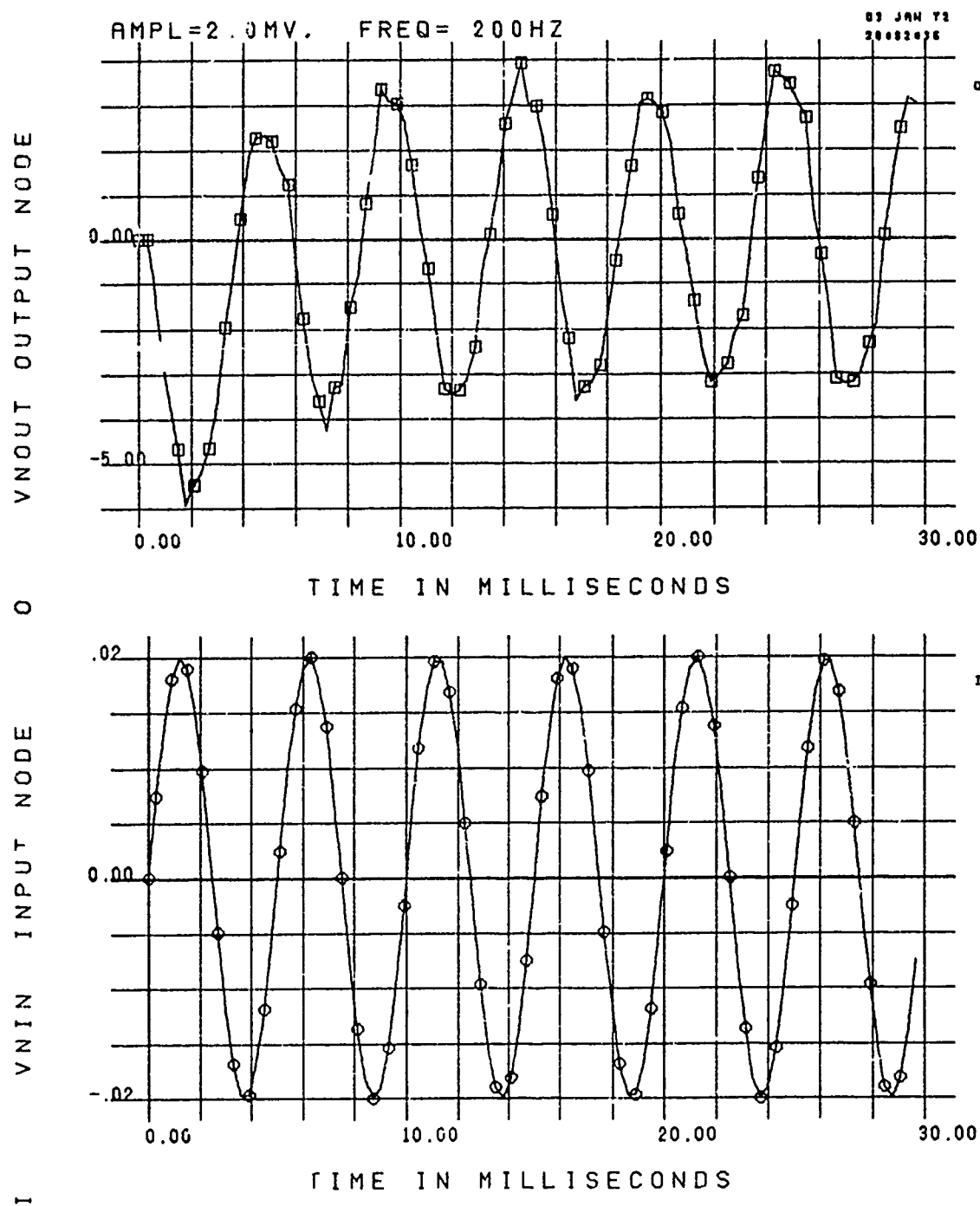


Figure 58f. 741 Response to 200 Hz, 2.0 mv Sine Wave

741 SINE WAVE RESPONSE RUN #349

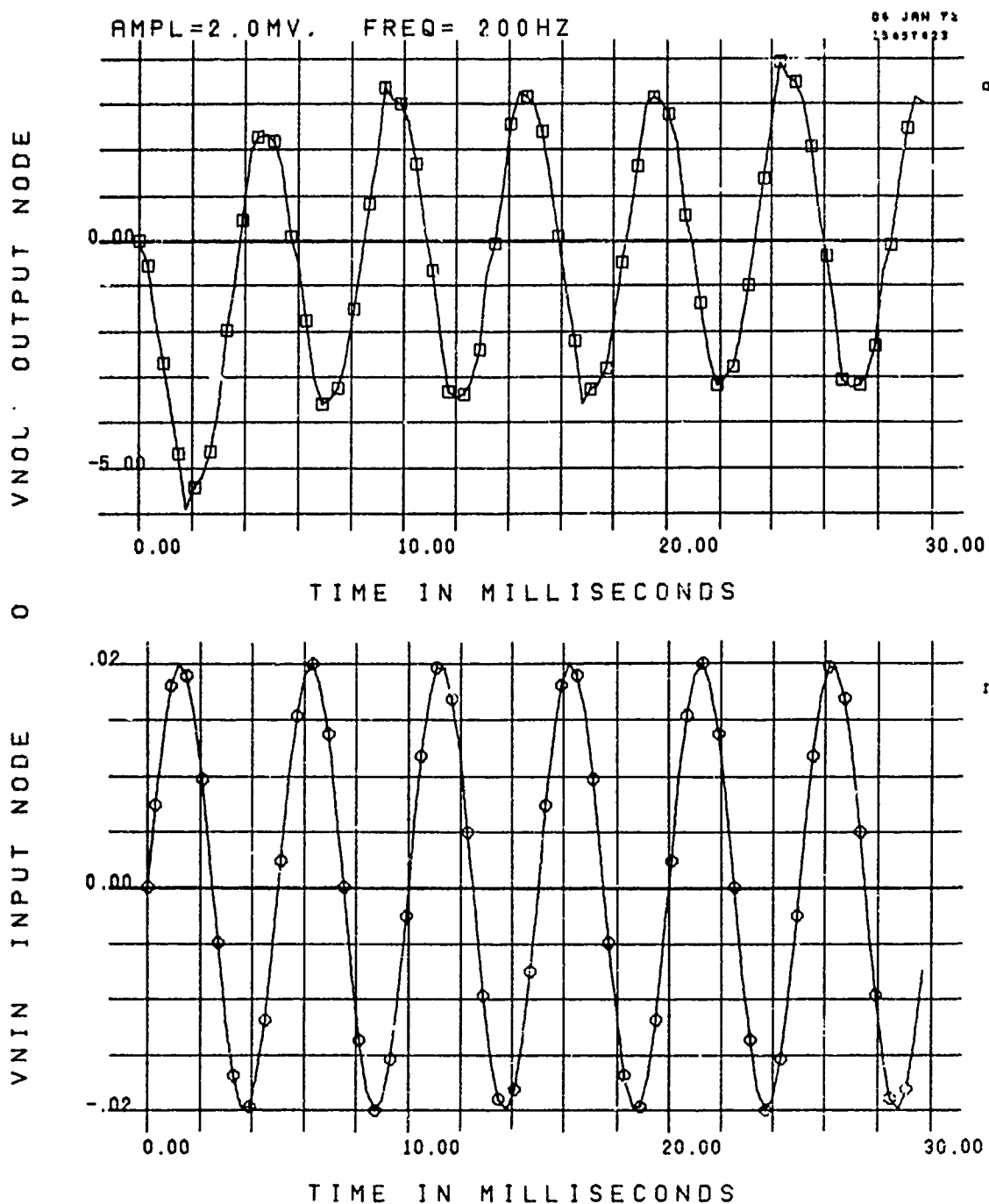


Figure 58g. 741 Response to 200 Hz, 2.0 mv Sine Wave. Rerun of Run 341 with CONVOL Corrected for Multiple Calls by Self-Consistency Loop. Compare with Figure 58f.

a few incorrect terms.

At higher frequencies, above the corner frequency, as evidenced in the results of Figure 58e and f, the initial transient due to turn on of the excitation yields an output response which is distinctly biased in the direction of the initial impulse. This is precisely the result expected by a rigorous theoretical analysis of a linear one pole circuit by Laplace transform techniques. Thus as would be really expected, convolution on the transfer function surface does provide the correct transient solution to the high frequency sine wave stimulus. As time progresses, the response returns to the usual steady state solution with the output swinging equally about the zero voltage axis.

Figure 58g is the rerun of the results presented in Figure 58f in which a bug in CONVOL was corrected that was coming into account when the self-consistency loop in MAIN drove into CONVOL after the first iteration. The interesting aspect to note is that the difference between the two figures is actually very slight, and shows again the stability of a convolutional calculation process against errors in only one term out of a large number. The solution is indeed determined by the entire surface shaping rather than just the instantaneous derivative at the current time step as in the case of a state variable solution.

A comparison of the 741 response to a pulse with and without feedback is contained in Figure 59. The response to the input pulse of Figure 59a, bottom, is shown in Figure 59a, top, and Figure 59b. The voltage at the input node was divided down by a 10k/1k voltage divider to the device input, hence the 741 input level was approximately 1 mv. The output level is seen to be somewhat over 15 volts, yielding a circuit gain of about 15,000. The input pulse rise time was 5 ms, whereas that of the output was 10 ms. This demonstrates again the slowness of the open loop 741.

The time step size used in the calculation for the above case was 5 milliseconds. The fact that a reasonably valid result can be achieved with such a large time constant is interesting of itself.

The pulse response as calculated using the parallel algorithm discussed in Section II for a feedback configuration is given in Figure 59c and d. The feedback resistor was 100K, while the resistance from inverting input to ground was 1K; hence the circuit gain should be 100. The input pulse magnitude was 0.1 at the device. The magnitude of the output is seen in Figure 59c, top, to be 10.0.

741 RESPONSE TO + SATURATING PULSE

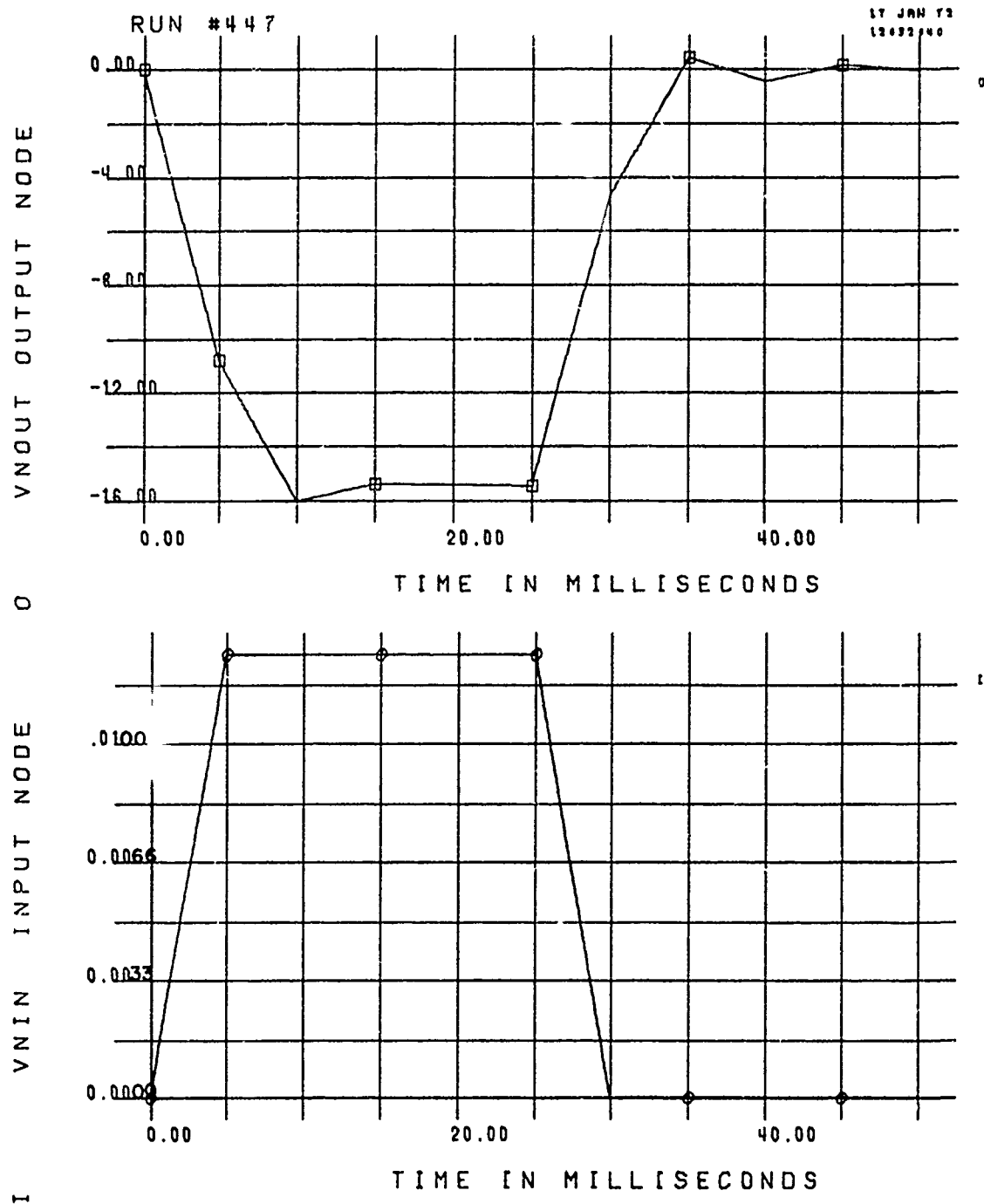


Figure 59a. 741 Response to a Saturating Pulse Without Feedback, Note Output Risettime is 10ms while Input Risettime is 5 ms.

741 RESPONSE TO + SATURATING PULSE

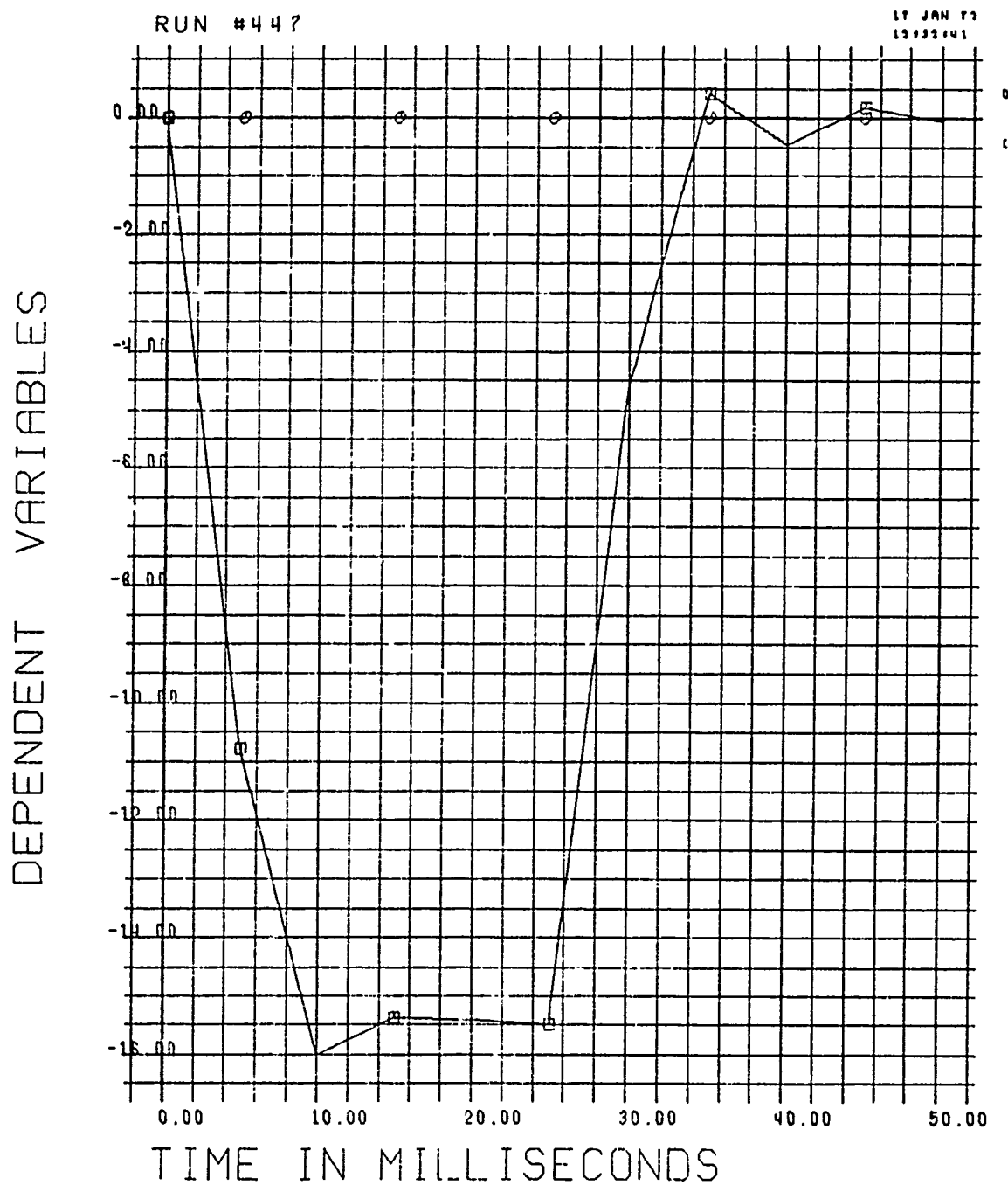


Figure 59b. 741 Response to a Saturating Pulse Without Feedback.

SAP 741 WITH FEEDBACK

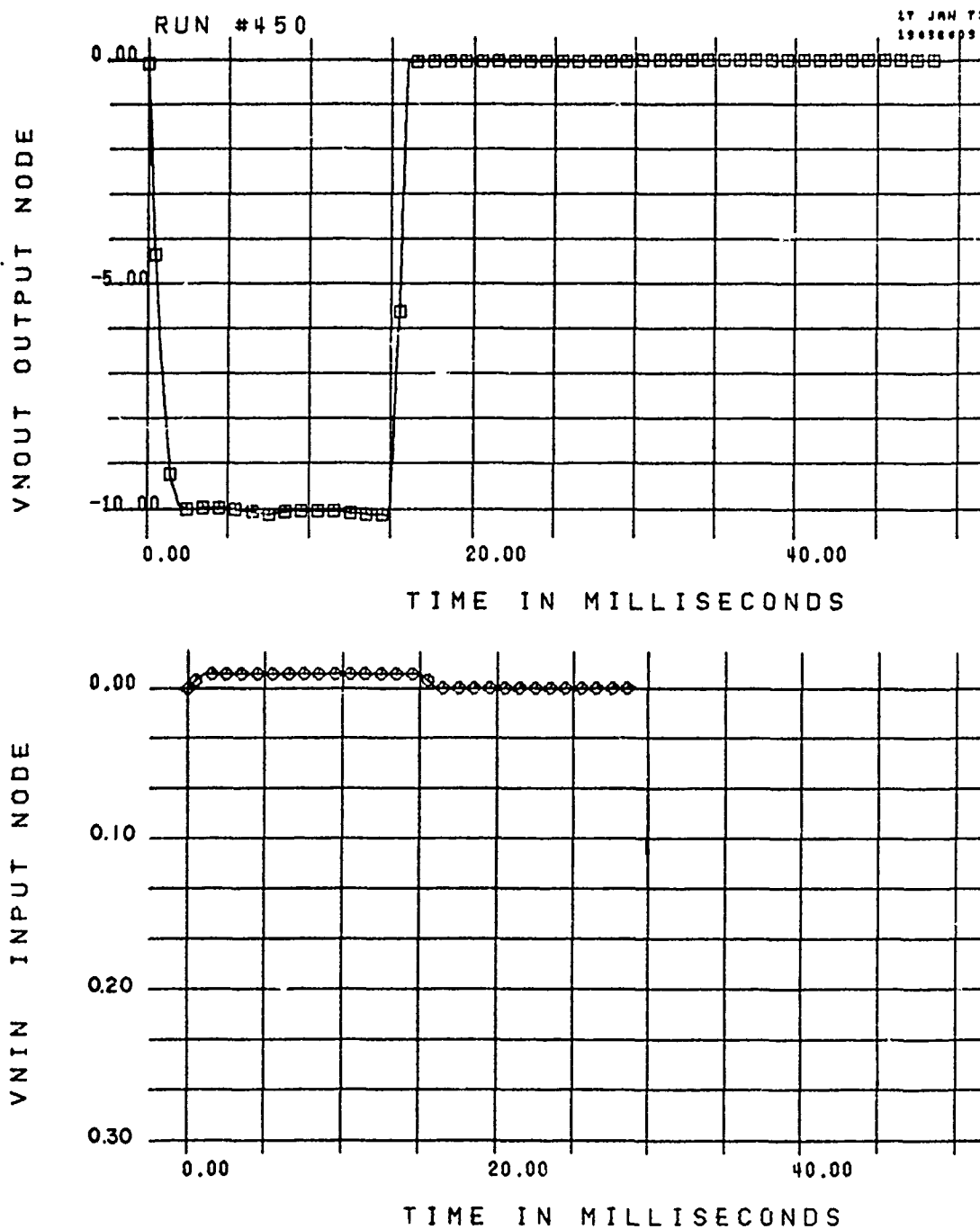


Figure 59c. 741 Response to Same Input Pulse With Feedback, Note Input Risettime is Reduced to 1 ms and the Output Risettime to 2 ms.

SAP 741 WITH FEEDBACK

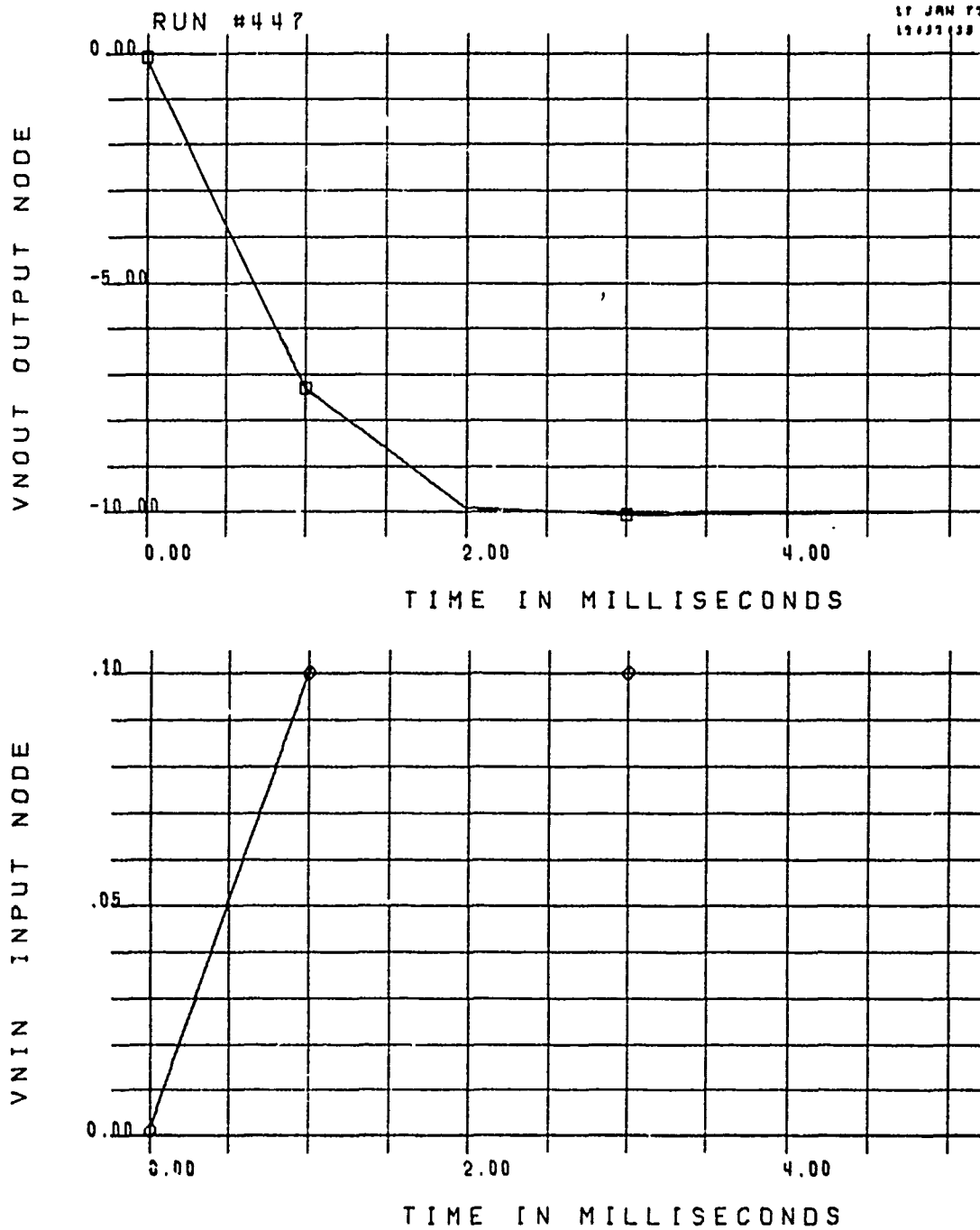


Figure 59d. 741 Response to Pulse With Feedback,
Early Time Portion of Waveform.

Hence, the computed circuit gain is in exact agreement with that expected theoretically. This demonstrates that the parallel iteration algorithm does work correctly. The critical aspect of the calculation for this case was the fact that the current splitting between the device input and the feedback resistor differed by a factor of $10^{1.2}$. The use of double precision arithmetic would seem advisable in such cases. Nevertheless, the results of the single precision calculation by PARLEL do seem satisfactory here.

Note also the higher response speed of the 741 in the feedback configuration as compared with openloop case. The output pulse rise time seen in Figure 59c, top, is 2 ms; which is five times faster than that obtained in Figure 59a, top, for the open loop case.

An important point is correctly selecting the EPSLON convergence criteria which the iteration loop in MAIN and the current splitting loop in PARLEL are driving towards. A relatively loose criteria is needed in the parallel iteration loop; in the case above 0.1. This seems rather large when compared with the parts per million convergence routinely achieved on the fixed output variable. On the other hand, a computational validity of 0.1 v out of 10.0 v is a relative precision of 1 percent. In experimental electrical engineering activities that degree of precision is regarded as acceptable.

The essential point deduced by detailed analysis of the printed listings of the parallel iterations runs was the inner parallel iteration loop should have a loose convergence criteria particularly at the beginning of the outer iteration, and then can be tightened up as the outer iterate is improved.

In a test run using linear resistors connected in parallel, the inner loop converged exactly on the second iterate. With a smaller spread of current splitting factors, the single precision computation is less a problem. However, the convergence criteria of the inner loop had to be about two orders of magnitude tighter in this case, or approximately the same as that of the outer loop as controlled by DRIVER.

2. 741 OP AMP - RADIATION RESULTS

In the foregoing, only two stimulus functions, the voltage and current driving into a block were taken into account by SAP. These were then propagated through

the system blocks by the successive block transfer functions. With radiation applied, one or perhaps two additional forcing functions are driving the system response, namely ϕ_x and ϕ_n , representing the X-ray and neutron dosage functions. The manner in which these forcing functions act on the system is different from the electrical stimuli in that they act simultaneously on all blocks. This difference is shown schematically in the block computational diagram as shown in Figure 60. The electrical response of the block to the radiation stimulus still propagates in the normal sense however.

The radiation voltage and current transfer functions are shown tied onto the radiation stimulus lines essentially in parallel, with the outputs summing into the block output terminals.

A basically interesting point now arises in relation to the fundamental iteration procedure that SAP uses to solve the system problem. Before with just electrical stimulus, one input variable was fixed and a modified Newton's method iteration algorithm applied to the adjusted input variable to meet the prescribed output constraint condition on the fixed output variable. Now with a radiation stimulus also acting, additional phenomena are occurring which are outside of the normal iteration loop. This is tantamount to the realization that the driving function is an independent variable for the case of radiation, but which is not true for the adjusted input variable. On this basis, the radiation terms were excluded from the iteration process taking place on the adjusted input variable.

The block input and output impedances are represented in Figure 60 as Z_{in} and Z_{out} . For a linear element, these would be very simple and perhaps best represented by an appropriate equation for the particular circuit element. For digital circuits, a machine fit to the experimental characteristics may be best. Such characteristics were taken at the SPR for the gates, and showed relatively small change with successive exposures.

Voltage and current radiation transfer functions are shown in Figure 60 for each block element. In our measurements at the Kirtland FXR or the Sandia Pulse Reactor (SPR), it seemed that one or the other are primarily important. The other may be small, or perhaps linearly related. Hence it does not seem always necessary to resort to a surface representation for both independently.

The radiation forcing functions programmed into RADGEN are piecewise linear, Gaussian, and Weibull distributions. The SPR radiation waveshape in time is

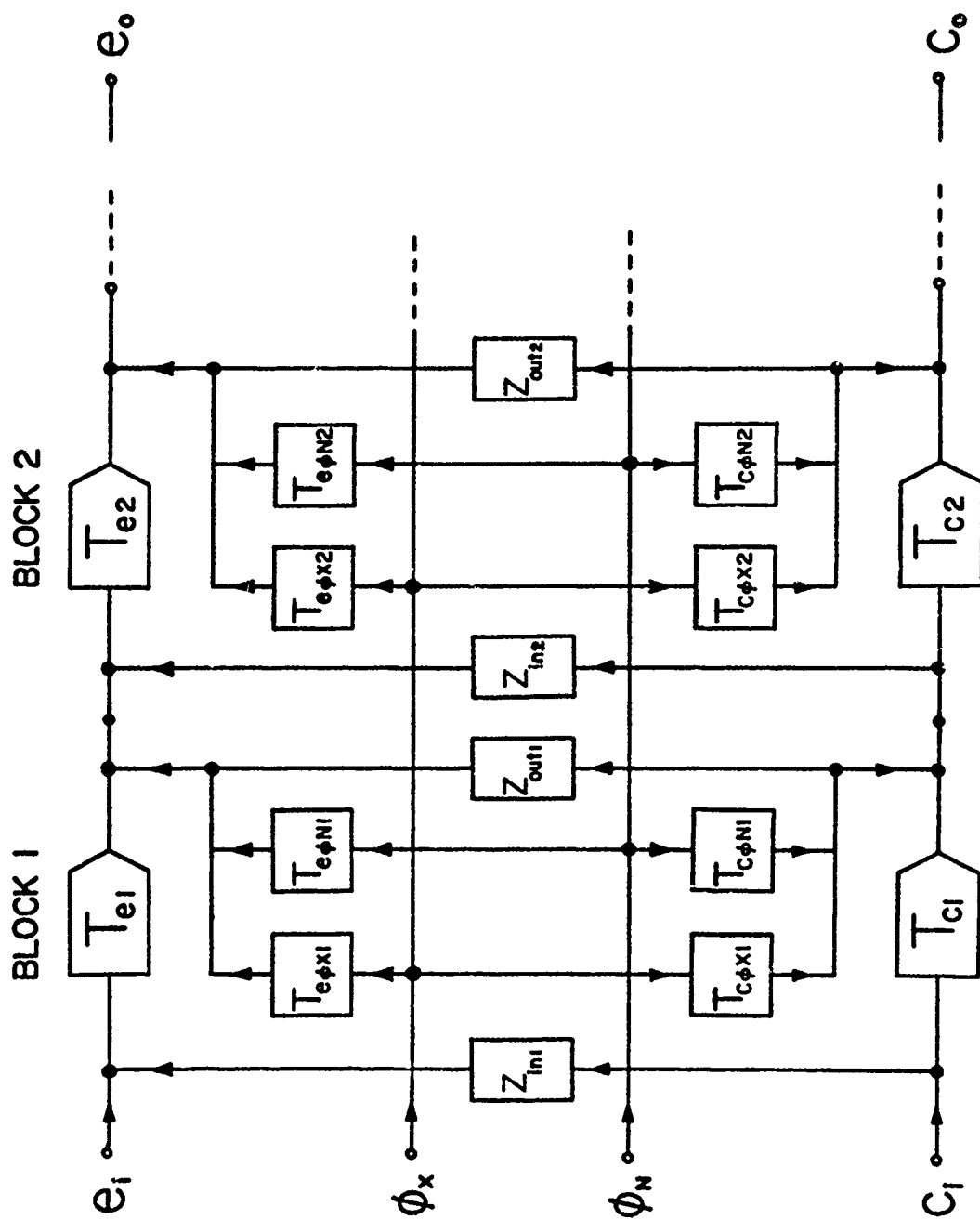


Figure 60. Schematic Computational Block Diagram Indicating How Radiation Drives Simultaneously into all Blocks.

approximately Gaussian; however, some asymmetry is evident in the shape relative to the peak. This type of asymmetry can not be achieved by a Gaussian distribution. The Weibull distribution is more flexible in that vastly different shapes can be generated by adjustment of the shape parameter. The equations for these two distributions are :

Gaussian

$$\rho(t) = \frac{A}{\sigma\sqrt{2\pi}} \exp - \frac{(t - \tau)^2}{2\sigma^2}$$

wherein A = amplitude, σ = width parameter, and τ = time of peak;

Weibull

$$\rho(t) = B \alpha^{-B} (t - \gamma)^{B-1} \exp - \left\{ \frac{t - \gamma}{\alpha} \right\}^B$$

in which α = time scaling factor, B = shape determining factor, and γ = time location parameter. The value of γ is usually zero. For the shape parameter B equal to a large positive number, say about ten, the Weibull distribution is a narrow peaked distribution like a Dirac delta function and α is the time at which the peak occurs. For $B = 1$, the Weibull distribution becomes an exponential. We judge that a value of B about 3 approximates the SPR radiation waveshape.

By inspecting the mode of calculation given in Figure 60, and neglecting for simplicity the input-output impedance terms, it is possible to write a closed form expression for the system output voltage and current in terms of the transfer functions and the input stimuli. Suppose for example, that we had three blocks and looked at the voltage calculation, the terms that enter into the machine computation are:

$$e_{o3} = T_{e3} (T_{e2} T_{e1} e_1 + [T_{e2} T_{ep1} + T_{ep2}] \varphi) + T_{ep3} \varphi$$

Or, in the general case

$$e_o = \left\{ \prod_{j=1}^{JMAX} T_{ej} \right\} e_1 + \left\{ \left[\sum_{j=1}^{JMAX-1} T_{epj} \prod_{k=j+1}^{JMAX} T_{ek} \right] + T_{epJMAX} \right\} \varphi$$

with a similar expression for the current. While the above expression is nice in terms of being able to reduce the operations into a form of transformations acting on the stimulus functions, the machine calculations more closely follow

the computational strategy contained in the block diagram format of Figure 60.

An interesting phenomena came to light in the machine computations, namely with small but non-zero values of the voltage stimulus variable on the input, the iteration procedure does not have sufficient signal upon which it can operate so that it can significantly effect the output. Hence if factors such as digital noise, or carryover from the radiation terms upset the output result, the iterate is too small to overcome and null them. This meant that those runs which had a simultaneous electrical signal applied during the radiation pulse yielded a more stable iteration pattern than runs where the electrical stimulus was small or non-existent. The solution to this problem consisted of isolating the radiation terms from the iteration process, and eliminating carry-over of prior computation elements by the self-consistency loop.

The very first runs that attempted to calculate the radiation response did have a concurrent electrical signal; consequently the iteration was very stable as the curves of Figure 61 and Figure 62 nicely show. Figure 61a (bottom) is the response to a FXR pulse of dose $\dot{\gamma} = 10^{10}$ and an electrical pulse of 1 mv, as shown in Figure 61a, (top). Figure 61b is the combined plot. A logarithmic time base was used in this calculation, covering seven orders of magnitude in time in 100 equi-interval log (time) steps.

The radiation first drives the output negative, and then up into positive saturation until a time of about 5 ms. The electrical input then becomes effective and drives the output down and dies away. Since the radiation simulation is that of an FXR burst, convolution of the radiation surface was not necessary; rather just one partition was driven back along an isodosal. The electrical response represents the result of convolution of many partitions along the T_e surface. Three surfaces were resident in core for the run -- TE741, TRX741, and TRN741.

The result of superimposing a SPR neutron burst with the same electrical pulse is shown in Figure 62. The initial positive and then negative voltage swing results from the valley present in the 741 SPR surface, Figure 23, at high dose. Note how the electrical term is more significant here than in the case of an FXR burst since the SPR burst is relatively slower in time by about three orders of magnitude, and at the high dose end of the SPR surface, degradation has reduced the long time radiation response. This run was executed prior to the incorporation of the routine DAMAGE into the computation.

SAP DETERMINATION OF 741 FXR RESPONSE

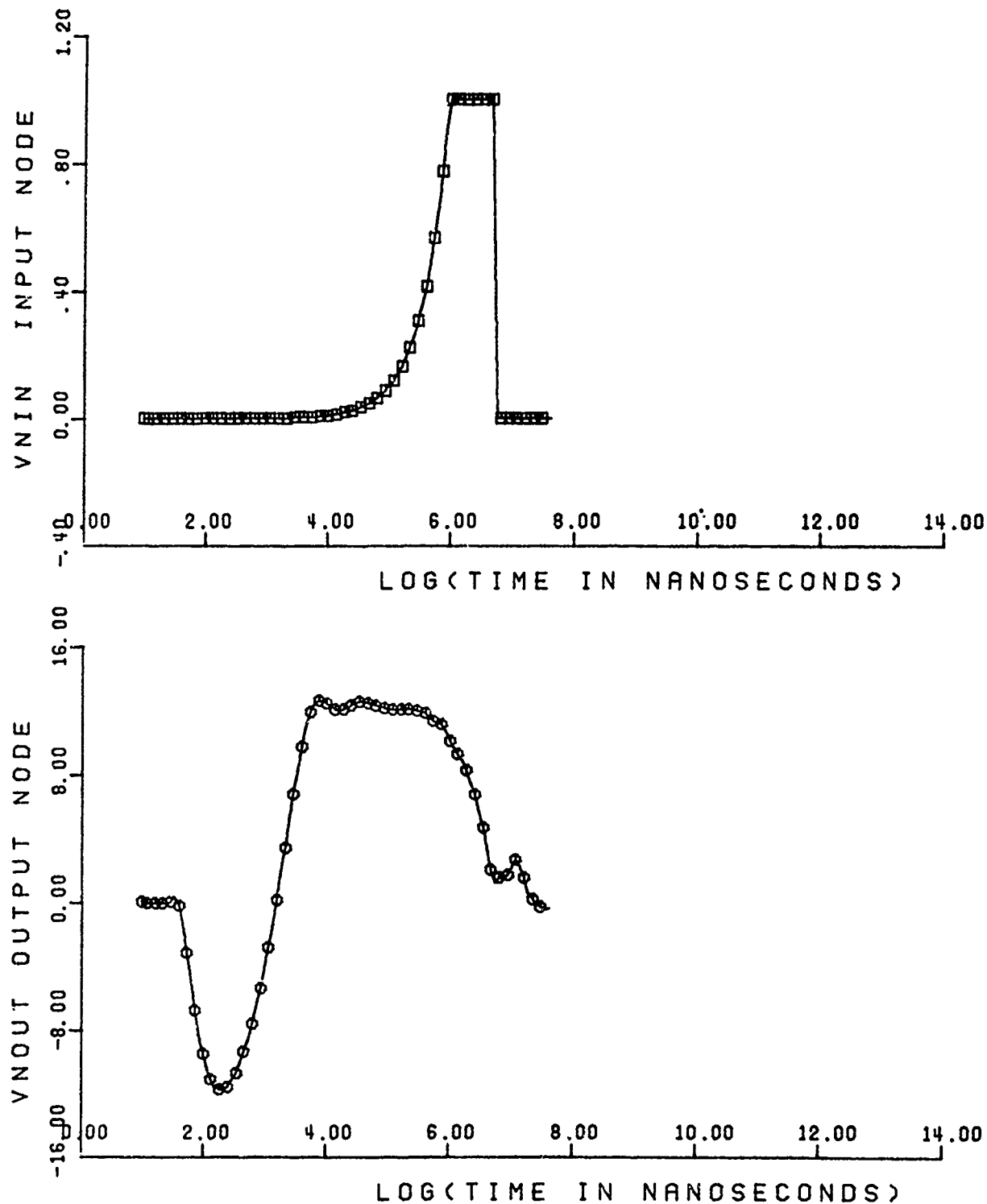


Figure 61a. 741 Response as Calculated by SAP to both Electrical and Radiation Stimulus, FXR Dose is $\dot{\gamma} = 1.0 \times 10^{10}$.

SAP DETERMINATION OF 741 FXR RESPONSE

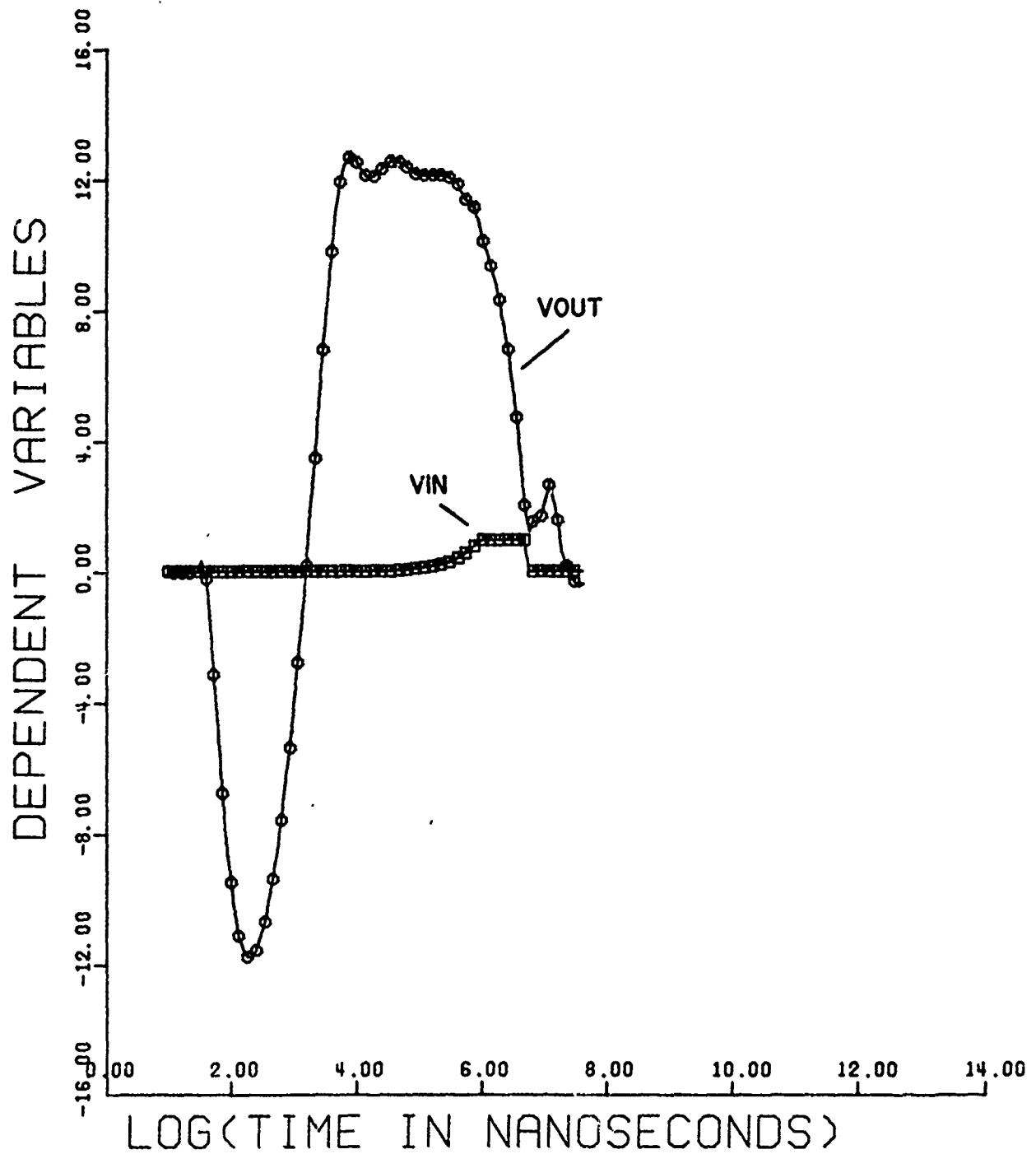


Figure 61b. 741 Response to Both Electrical and Radiation Stimulus. Note Log(Time) Base Covers Seven Orders of Magnitude in 100 Steps.

SAP DETERMINATION OF 741 SPR RESPONSE

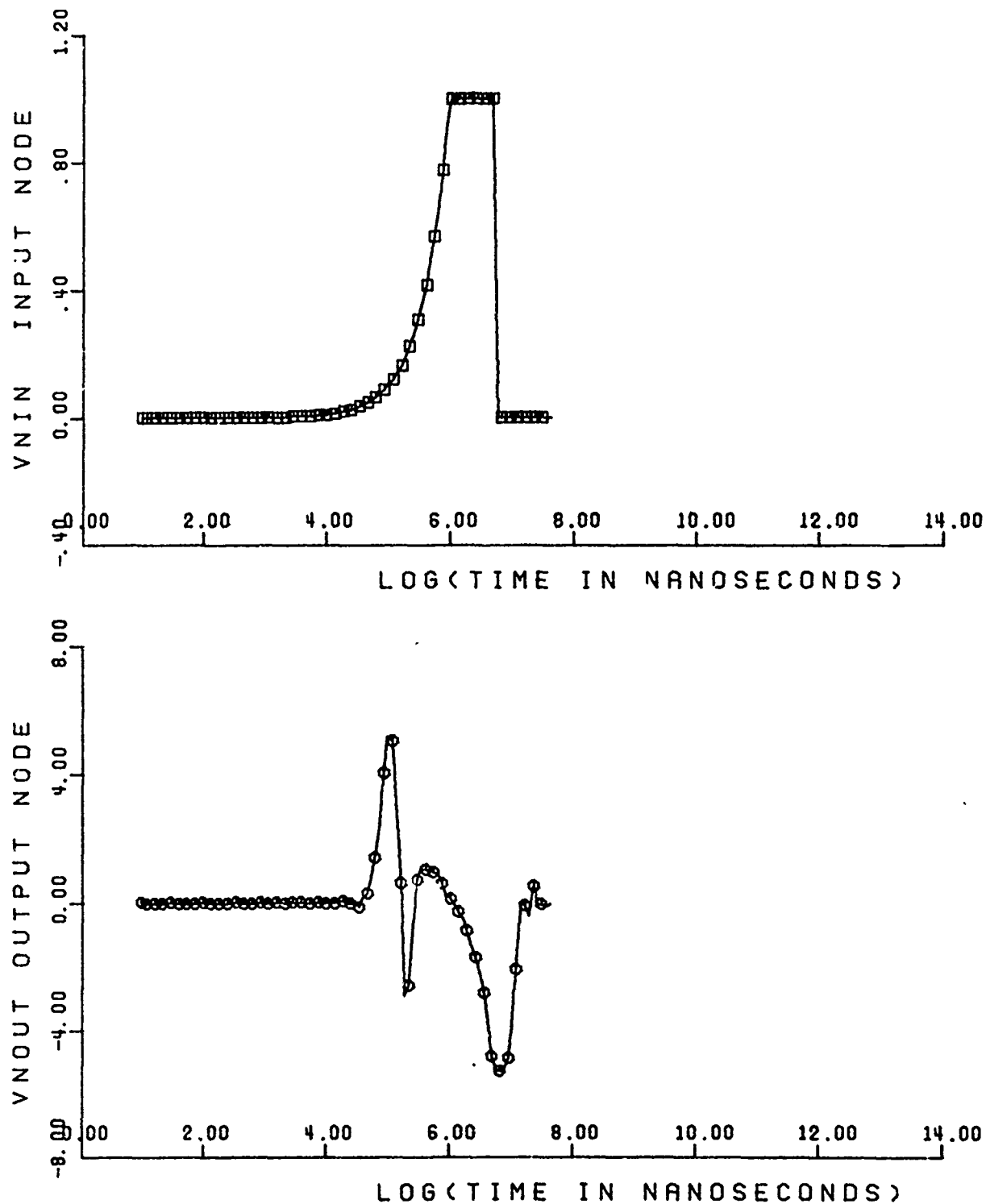


Figure 62a. 741 Response Calculated by SAP to both Electrical and SPR Radiation Stimulus, Neutron Dose is 7×10^{13} nvt/cm².

SAP DETERMINATION OF 741 SPR RESPONSE

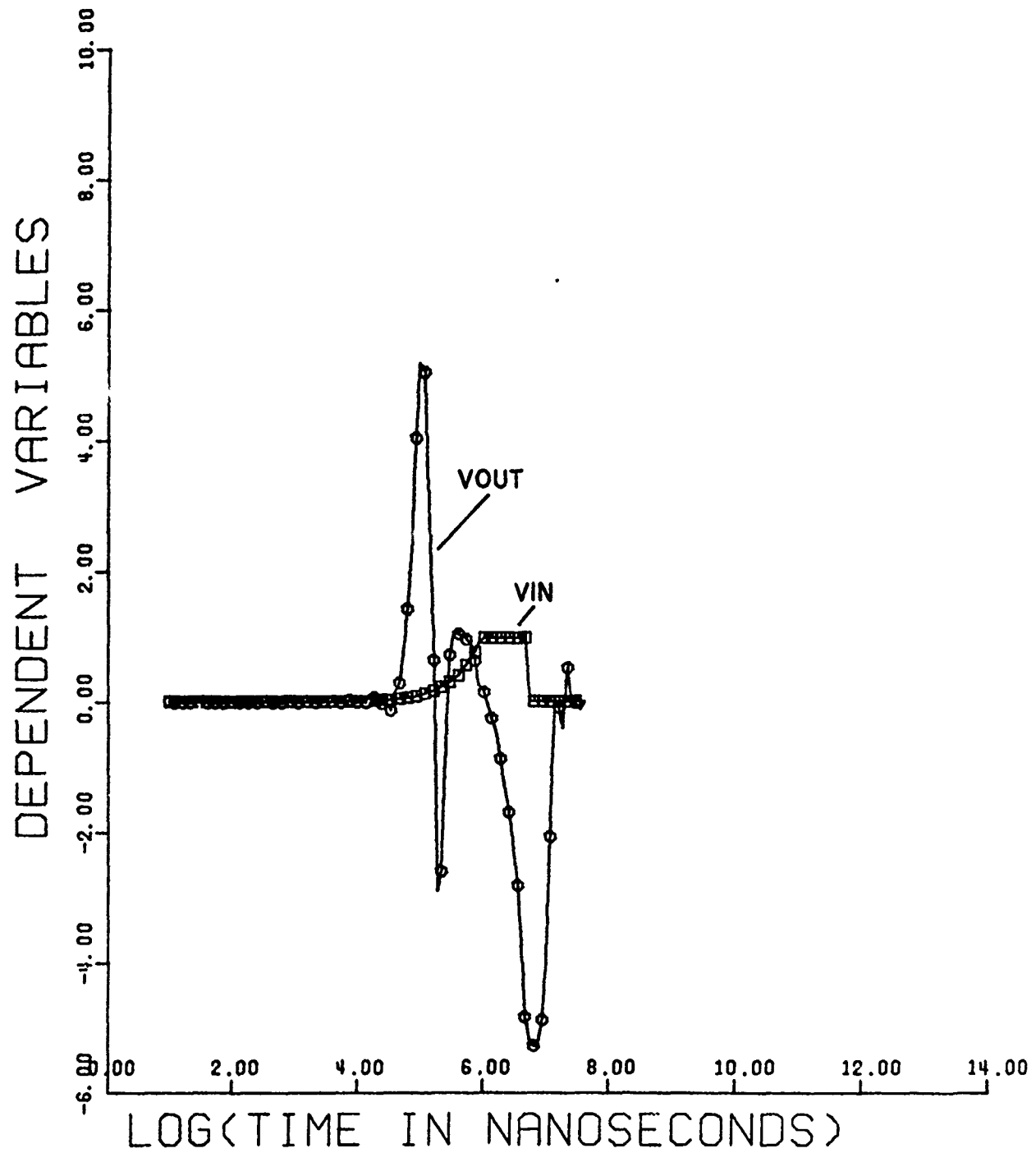


Figure 62b. 741 Response Calculated by SAP to both Electrical and Radiation Stimulus.

The results presented in Figure 61 and Figure 62 were obtained on the same job submission in about one minute execution time on the Univac 1108. File manipulation, and compilations if any, typically extend the execution time another 30-40 seconds. The computations are extending over seven to eight orders of magnitude in simulation time. The calculational time base and the plot time base were both logarithmic. If a linear plot scale is used, the fast early phenomena are invisible.

The FXR simulation of Figure 63 is for a somewhat lower dose level of $\dot{\gamma} = 6.8 \times 10^7$ and no electrical stimulus. There are actually four curves of significance on the figure, rather than the evident three. The top curve is the output current in milliamperes. The center curve is actually two curves lying on top of each other. The plus symbols denote the radiation response alone; the squares denote the superposition of that with the electrical output. Since the input, as given by the curve on the zero voltage line by diamonds, is zero, the degree of success of the iteration process in MAIN is indicated by the difference between the two center curves. A small difference means the process is working well, a large difference contrariwise. Figure 64 is a SPR simulation for a dose of 10^{13} .

3. 9704 NAND GATE ELECTRICAL RESPONSE

The electrical response of the 9704 gate is calculated using the surface depicted in Figure 26. Since the x coordinate of the surface directly returns the output magnitude, rather than a transfer function, the multiplier UMULT on the convolution terms in CONVOL is set to the value one rather than the stimulus variable UIN, as can be seen in the listing of Figure 52c. Also since the surface represents the results of first turning on, and then turning off the pulse, if a partition continuously moves back along the surface two output pulses are generated, one at the turn on transition, and one at the turn off transition. This yields a double pulse of opposite polarities, as is shown in Figure 65 (top). The input pulse that generated it is shown in Figure 65 (bottom). It is seen that the first pulse is a reasonably good approximation to the real device response as shown in the x-y recorder tracing of a sampling oscilloscope output given in Figure 66.

By simply setting the value of XHIGH for the surface of Figure 26 to a value of about 200 ms, TSURF returns the surface z value at XHIGH when the call from

741 RESPONSE TO FXR DOSE=6.80X10**7

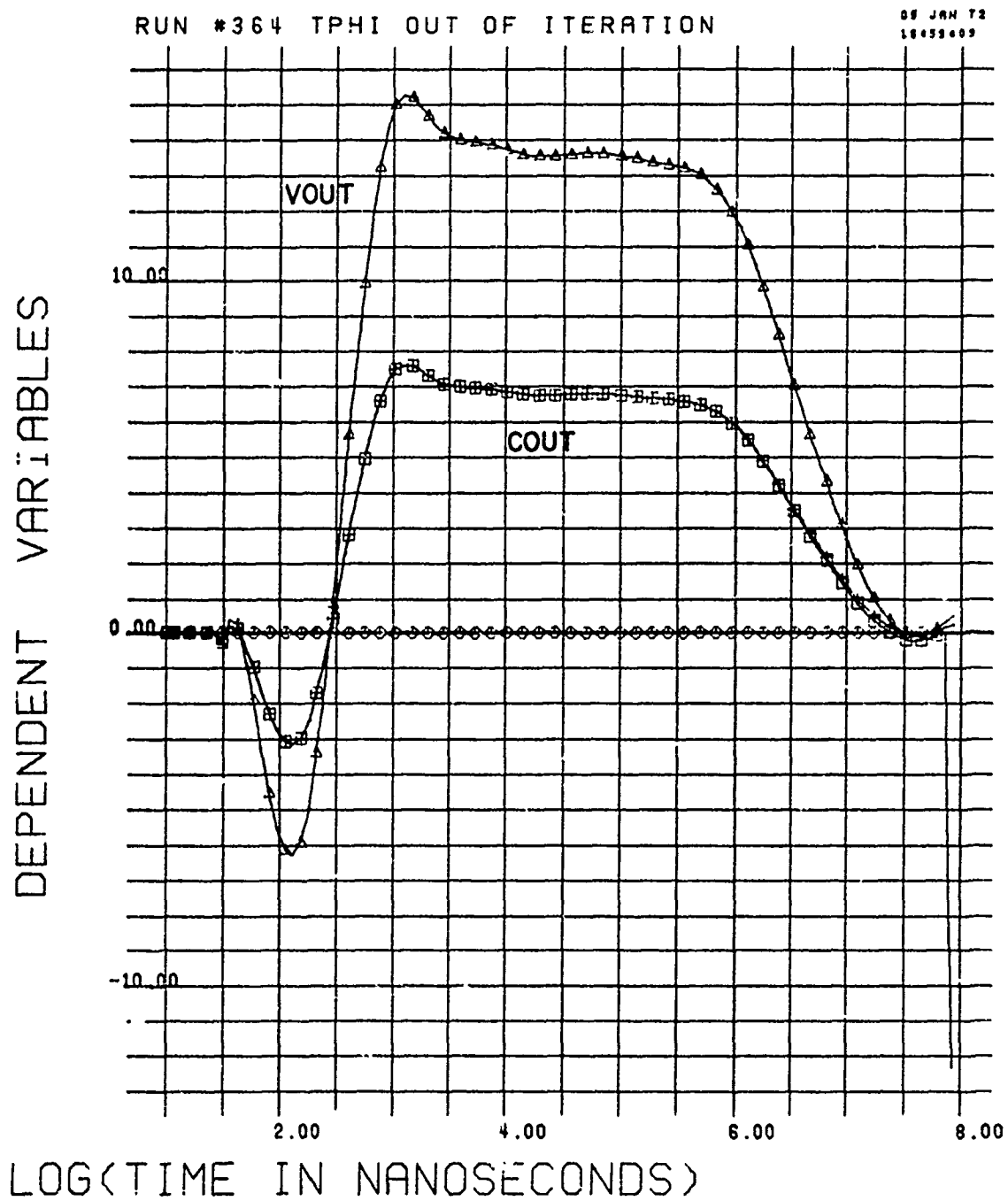


Figure 63. SAP Simulation of 741 Response to FXR Pulse.

741 RESPONSE TO SPR NOT CONVOLVED

DEPENDENT VARIABLES

RUN # 553 DOSE = 10++13

28 FEB 72
11:20:28

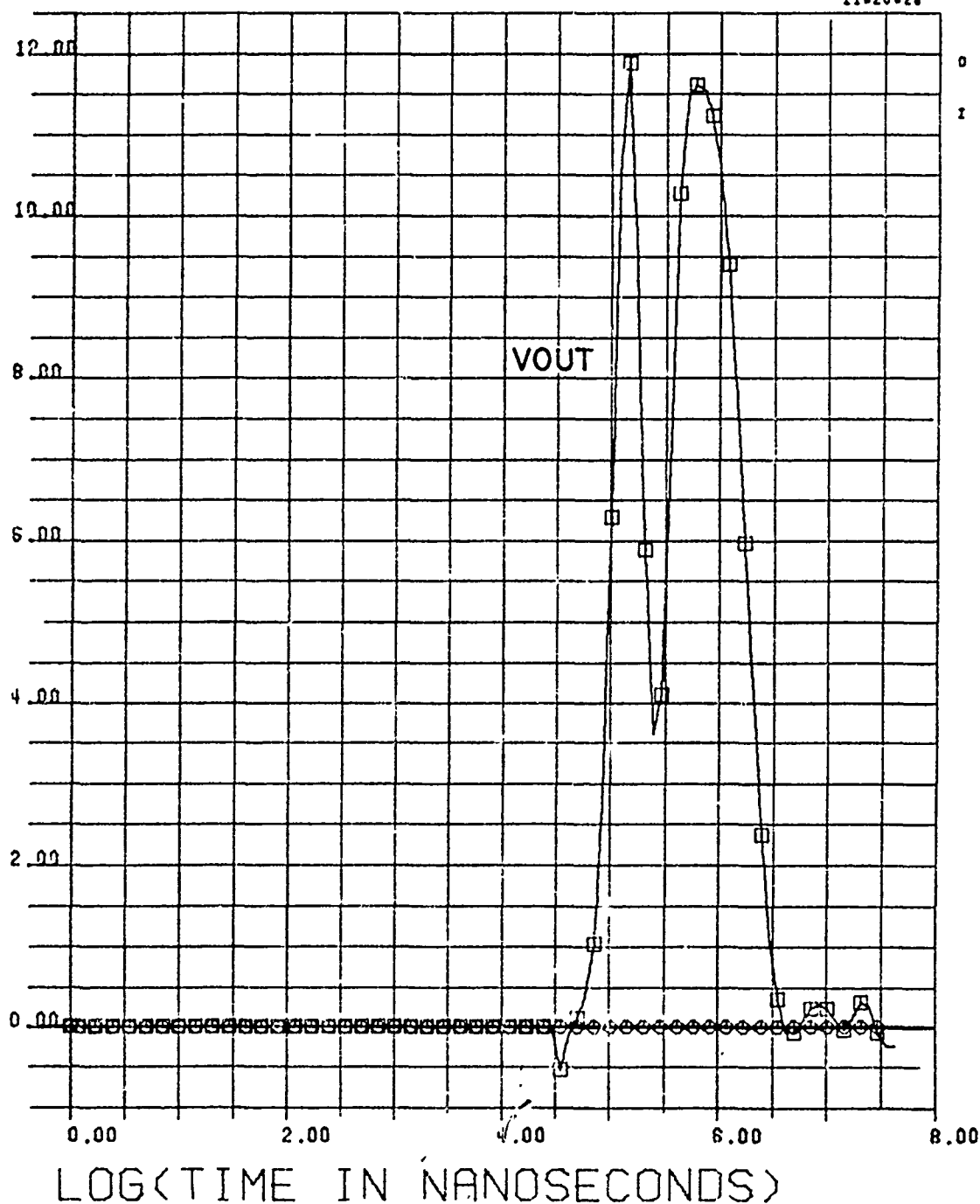


Figure 64. SAP Simulation of 741 Response to SPR Burst.

SAP 9704 NO IRRADIATION

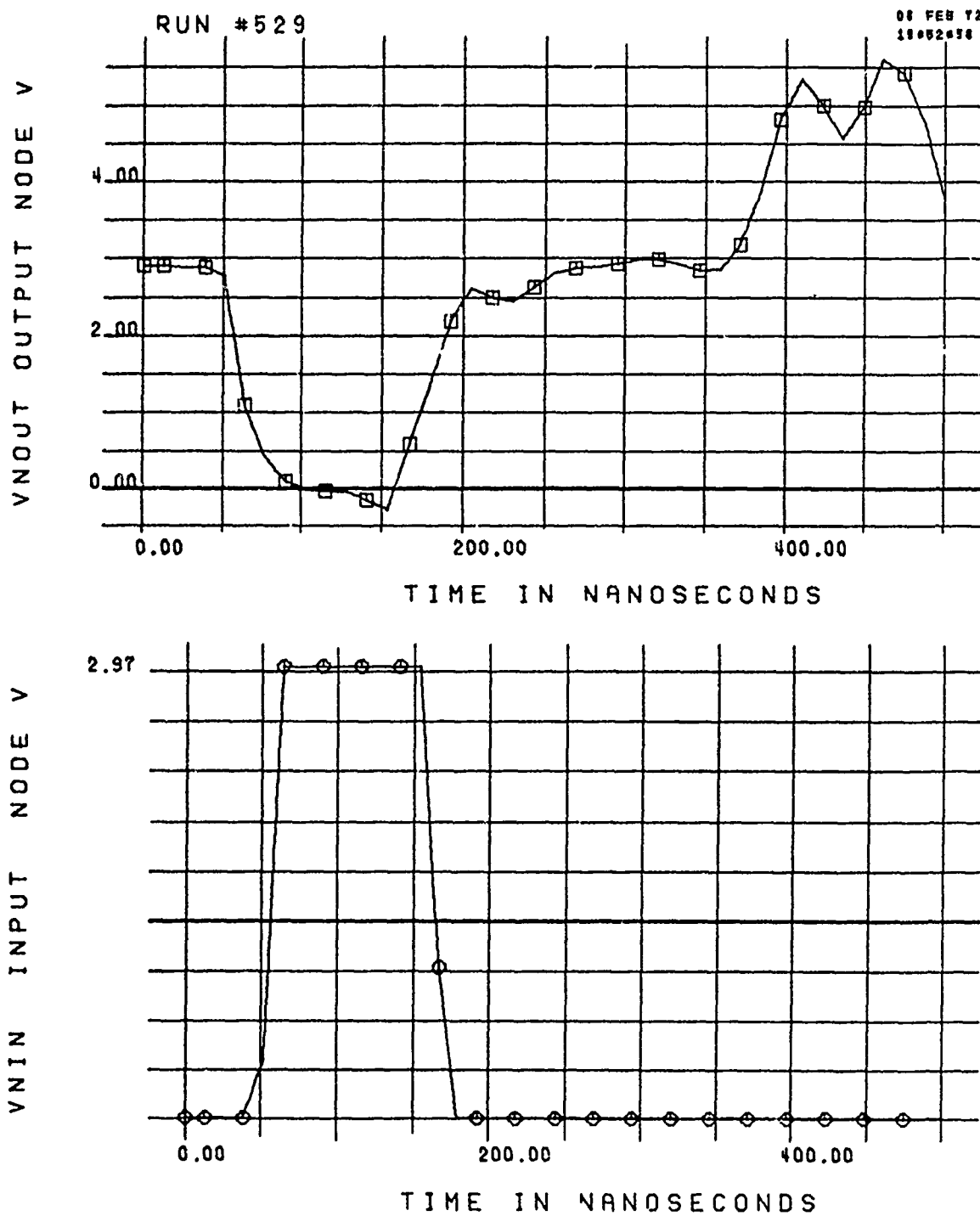


Figure 65. SAP Calculation of 9704 NAND Gate Response to Electrical Pulse.

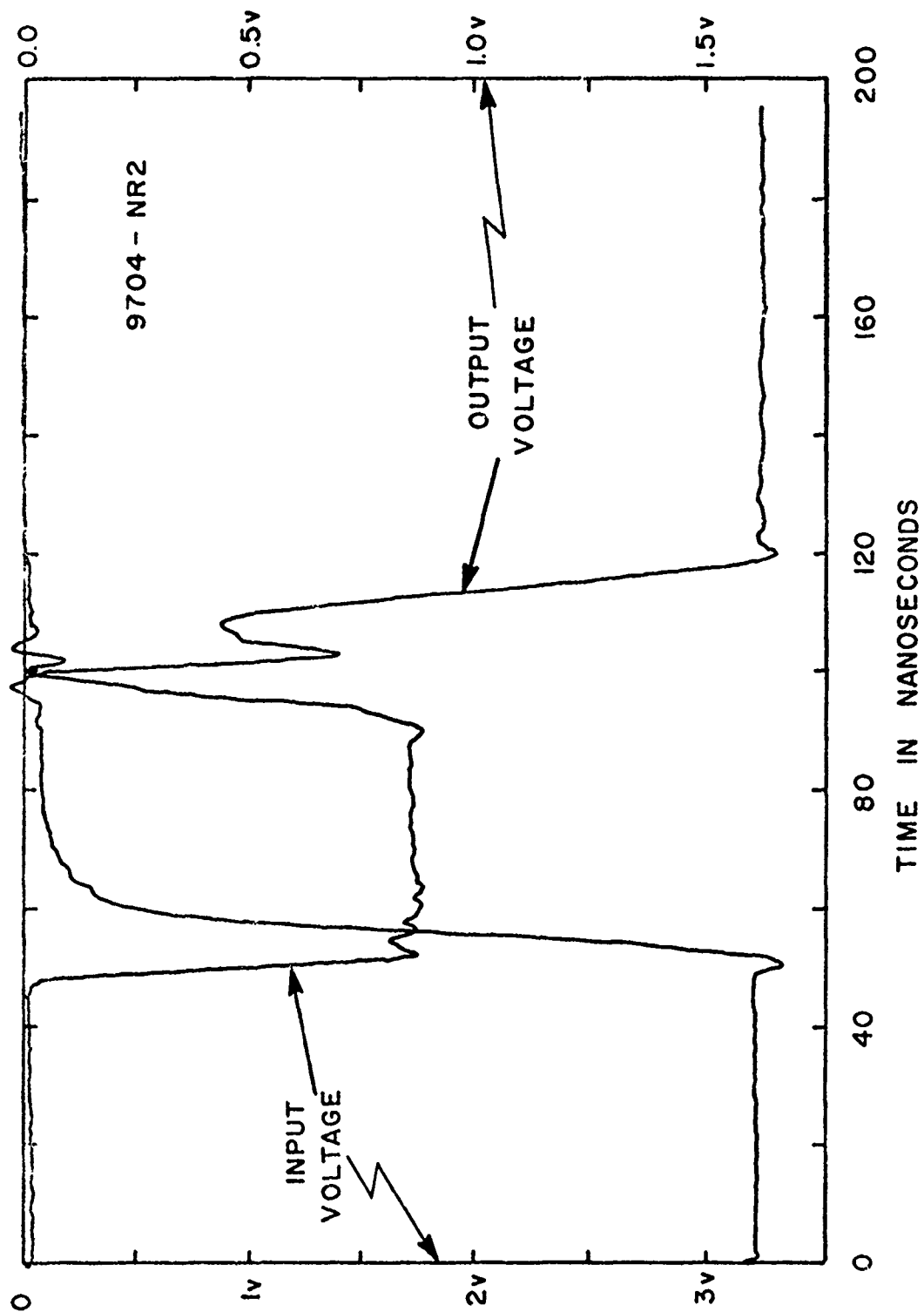


Figure 66. X-Y Recorder Tracing Made with a Sampling Oscilloscope of 9704 NAND Gate Response to a Pulse Input Waveform.

CONVOL has a time coordinate exceeding XHIGH. The result is to delete the second pulse as is shown in Figure 67. The rise after the pulse is resumably due to the surface structure just ahead of the new XHIGH value. Further computer experiments would consist of further reduction of XHIGH or smoothing of the surface in the low state. There is however a fairly good simulation of the rise and fall times of the real device. Thus even with the relatively preliminary nature of the results presented here, it appears that convolution on experimentally obtained input-output surfaces can yield dynamical representations of device performance which are fairly good.

One aspect of work on which effort was not expended because of the limited time span available was the modeling of multiple inputs to the gate. There are at least two ways of doing this. One is to use logical AND statements on the individual input signals to ascertain the logic state, and to obtain the dynamics then by taking the equivalent composite pulse and running it back along an isovoltage loci on the surface. Thus, convolution would not be required. This essentially is equivalent to presuming that the input pulse rise times are all very short, rather than like a slow ramp. The alternative procedure is to convolve each individual pulse along the surface and sum the outputs modulated down by the number of inputs N . Thus, the switching threshold would be N times the individual input threshold. If the sum is more than the N threshold, then the logic device switches.

The actual device has multiple emitters on one transistor. That transistor essentially forms a composite input waveform to the emitter-base junction. The junction really does not see separate waveforms coming in. The input transistor operates in grounded base configuration, and is fast since the alpha rather than the beta cut-off frequency is applicable. The slowness of the entire circuit is due to the fact that the following transistors operate grounded emitter. The logic function arises at the input since if any one emitter is down, it hogs all the base current and keeps the input transistor on. Only when all the emitters are high does the input transistor turn off, and so turn on the following stage. After the first input transistor, the circuit is the same as a single input device. The logic thus occurs right at the input as in other T^2L devices.

Thus, since the input device is fast, and the logic occurs immediately, it would appear reasonable to use a logical IF statement to perform the logic function, and the surface to generate the total dynamics from the composite waveform thus created.

SAP 9704 NO IRRADIATION

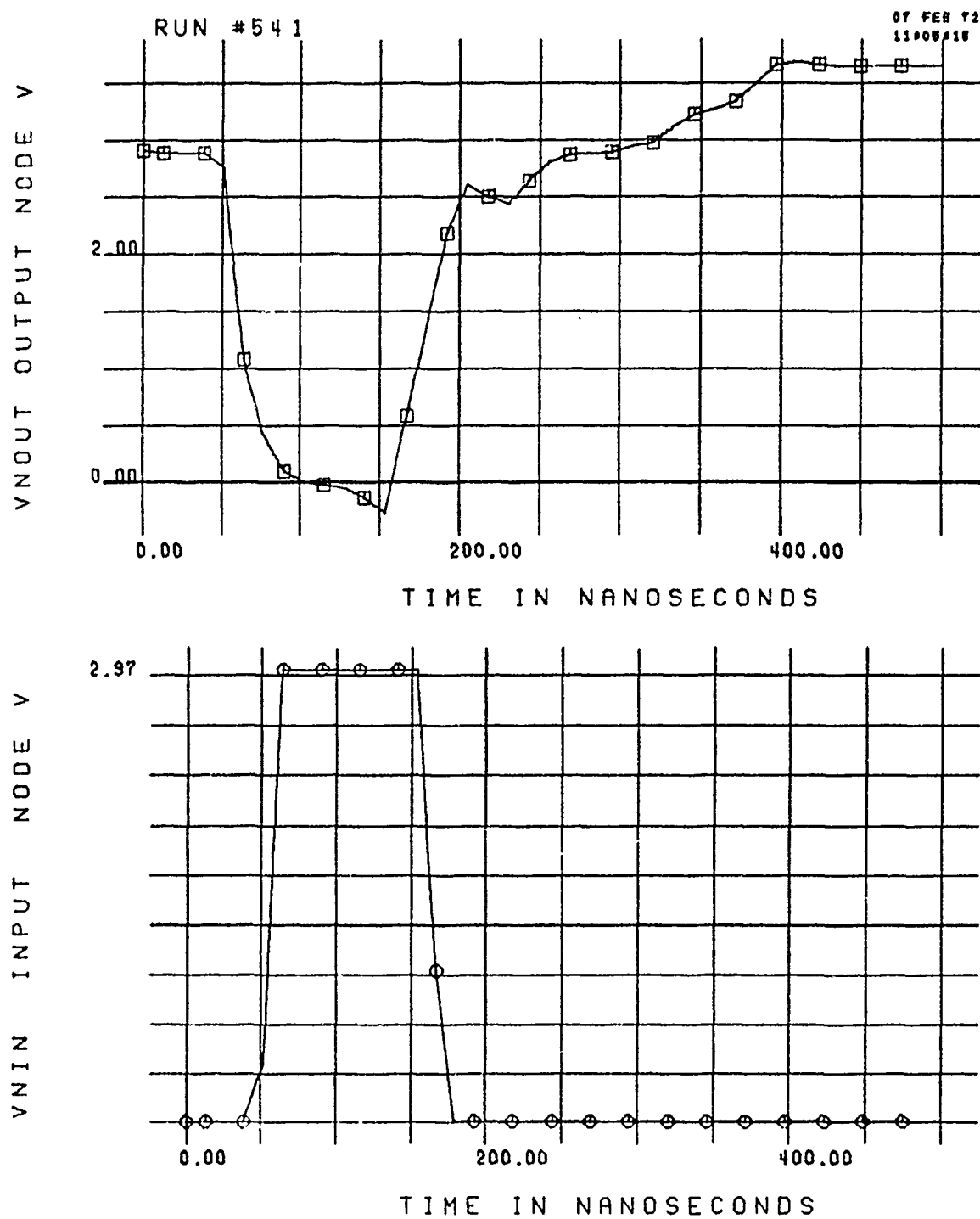


Figure 67. SAP Calculation of 9704 Response to an Electrical Pulse After XHIGH was Lowered.

The convolutional approach on the individual input waveforms would appear better in cases which are more apt to have input stimuli which are slow or have uncertain voltage levels, perhaps lying in the transitional range. Then the motion of the partitions on the surface would sample the slow, long-time constant, transitional region of the response surface.

The merit of these two methods should be ascertained in any future work with the general ideas discussed in this report.

4. 9704 NAND GATE RADIATION RESPONSE SIMULATION

The combination response to an FXR pulse and the voltage waveform shown in Figure 68 (bottom) is presented in Figure 68 (top). This represents the output as calculated using the voltage response surface (Figure 26); the radiation voltage response surface (Figure 39); and the radiation current response surface (Figure 40). The device was loaded with a 500-ohm resistor rather than another gate. The computed waveform seems to be primarily determined by the basic electrical response alone because the dosage $\dot{\gamma} = 1 \times 10^{10}$ is below the level at which a significant excursion occurs on $T_{\phi V9704}$ as in Figure 39. The magnitude of the radiation terms in Figure 68 (top) are only about 0.1v in magnitude and so do not dominate the electrical term.

The first attempt at the neutron simulation is shown in Figure 69. The plotted output became very large and negative near the end of the calculation. This was seen, by looking at the listings of the convolution sums, to be due to the continuing contribution of partitions on the T_{ϕ} surface which had reached the end of the surface and were continuing to contribute ever larger and larger values to the sum by virtue of the logarithmic increase of the time step multiplier coming into play as time went on. Since the surface terminates at 450 ns, a partition whose time step multiplier is in the millisecond range makes a very large contribution in the calculation even though its dynamic effects are long past.

The cure to this problem turned out to be relatively simple as shown in the TSURF listing of Figure 53, when the time entry look-up variable exceeds 10 times the XHIGH value for the surface, simply return the value zero to CONVOL. The resulting change in the calculated result is shown in Figure 70. For this particular run, four surfaces were resident in core and were being used simultaneously during the calculation. These surfaces are:

SAP 9704 FXR PULSE RERUN 447

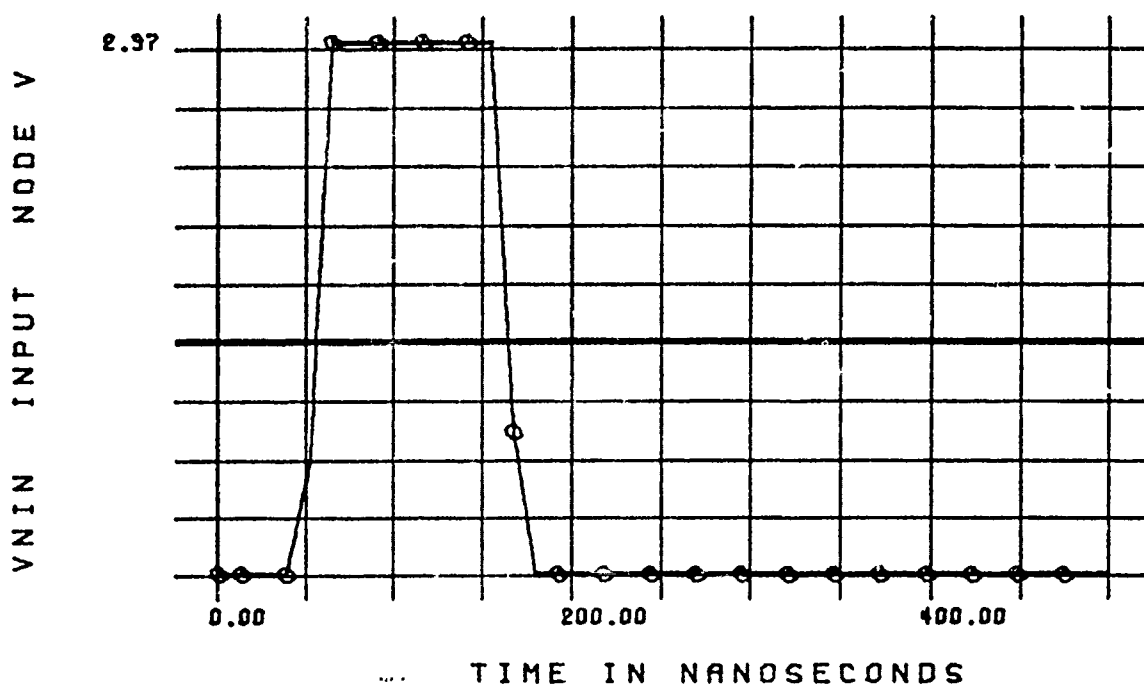
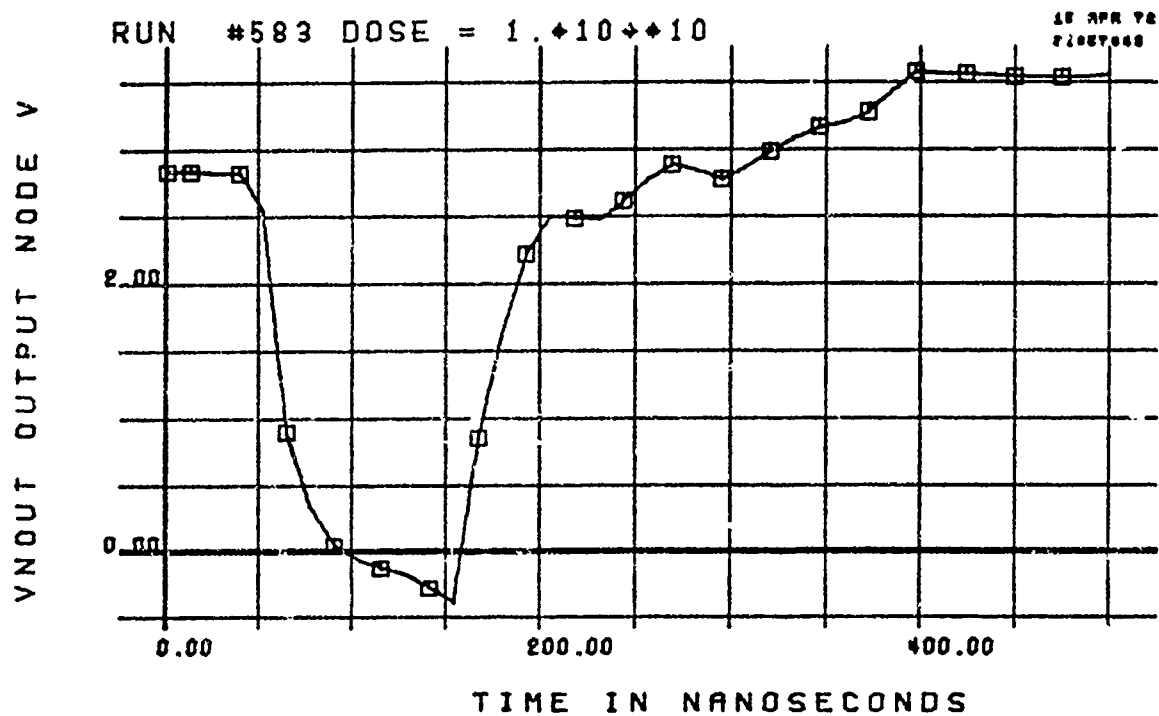


Figure 68. SAP Simulation of 9704 with Electrical Pulse and FXR Radiation Dose = 1.0×10^{10}

SAP 9704 SPR IRRADIATION

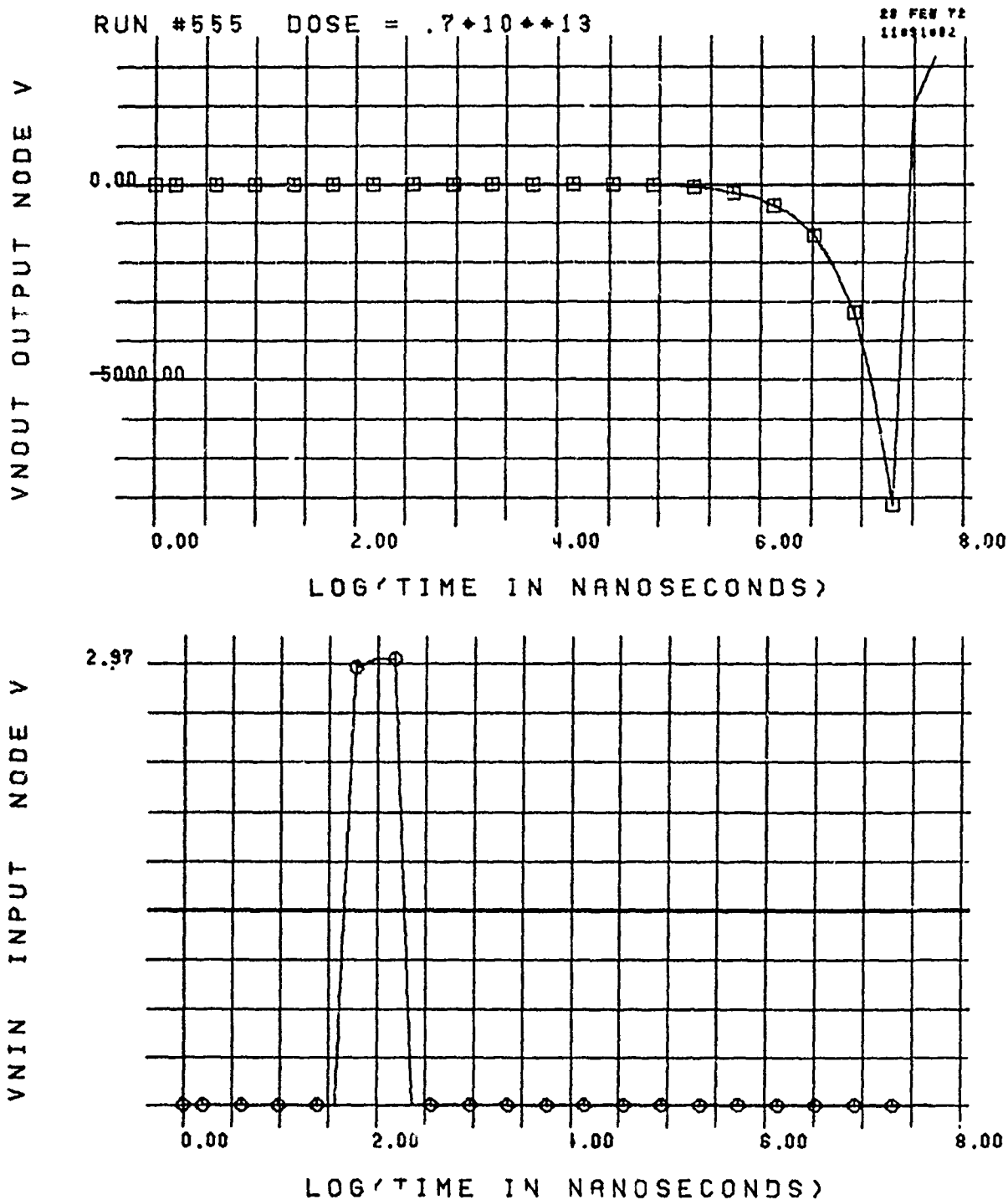


Figure 69. First SAP Simulation of 9704 Neutron Response with Electrical Pulse. Log(time) Base and Large XHIGH Caused Divergence.

SAP 9704 SPR IRRADIATION

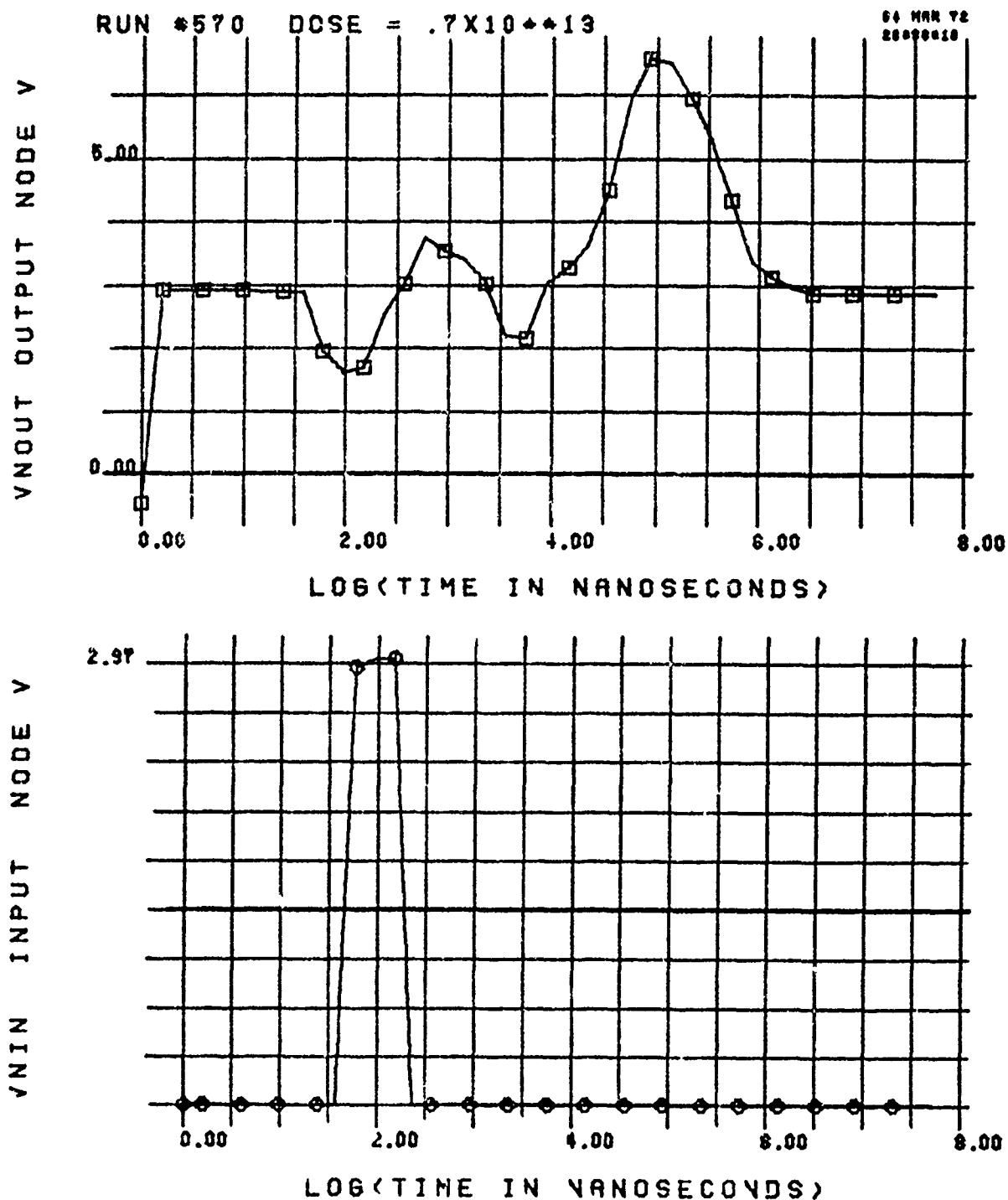


Figure 70. Second SAP Simulation of 9704 Neutron Response with Electrical Pulse. Reducing Value of XHIGH Stabilized Convolution Process.

- T_e Figure 26 9704 Voltage Response Surface,
- $T_{\phi NV}$ Figure 33 9704 SPR Voltage-Neutron Radiation Surface,
- $T_{\phi NC}$ Figure 36 9704 SPR Current-Neutron Radiation Surface, and
- $T_{\phi NV}$ Figure 23 741 SPR Voltage-Neutron Radiation Surface.

The latter surface was being used by DAMAGE to calculate the permanent degradation caused by the shot.

The rather interesting item of interest of the result shown in Figure 70 is the prediction that the output voltage rises above the supply voltage of 5 v to a value more than 6 v. Rechecking the Polaroid pictures taken during the experimental bursts at the SPR showed that this was indeed true, and the calculated value agreed very well with that observed.

The results of higher neutron radiation level are given in Figure 71, for the case of a simultaneous low electrical input state. The load again for the calculation is a 500-ohm resistor. The dominant term in the output voltage is the photocurrent from the radiation current transfer function surface, Figure 36. The radiation voltage term, the curve with small triangles, did not seem to have as much effect on the output as might have been expected. This might have arisen due to the manner in which the degradation effect was being handled by DAMAGE and MAIN.

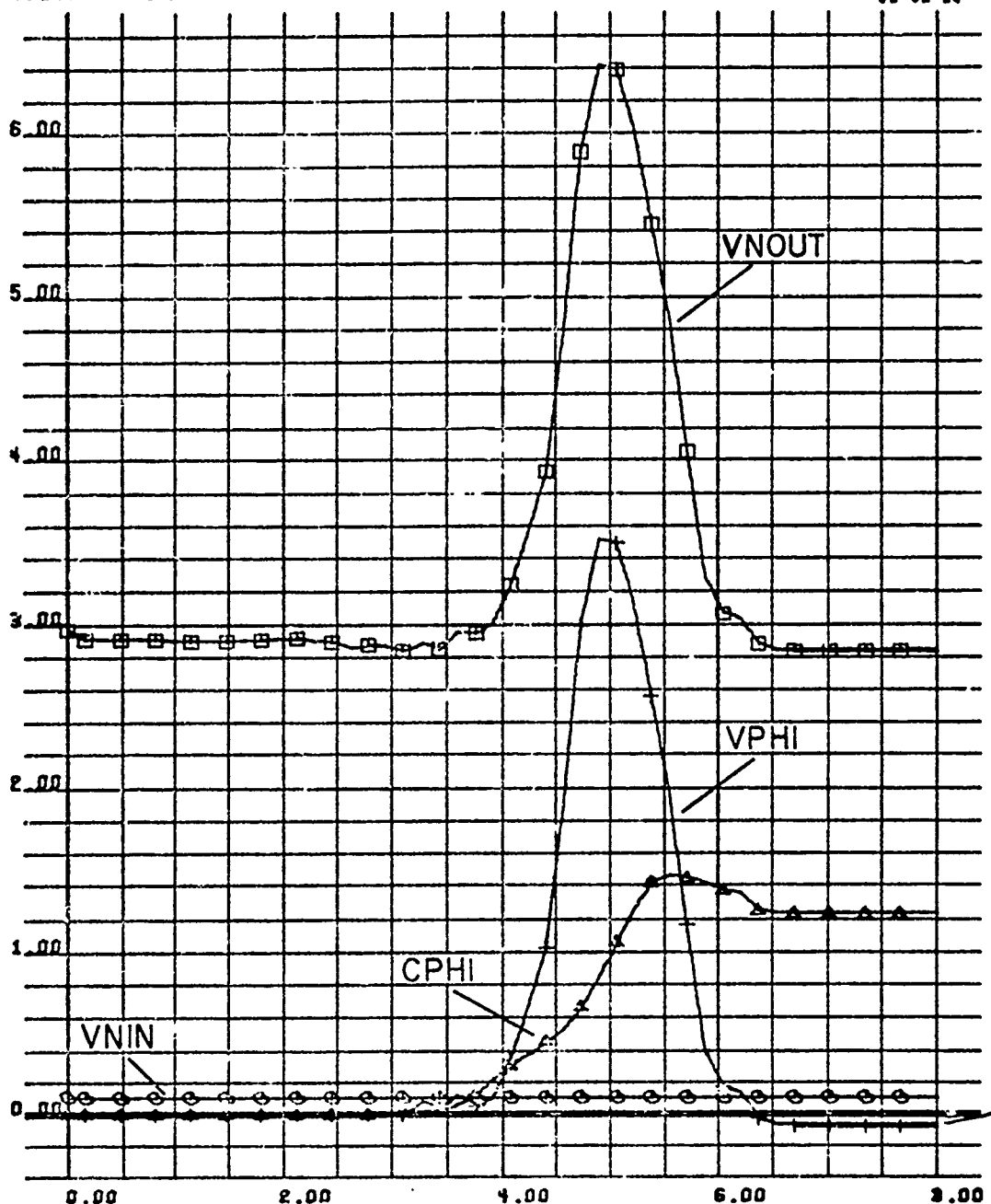
The case of a high electrical input on the gate, along with a high level SPR burst is shown in Figure 72. The first dip and peak seem to be spurious, perhaps due to the large time steps outweighing the convolution process. It might be wise to keep in mind that the convolution calculation is essentially the approximation to an integral, and when the differential time step size becomes comparable or larger than the length of the surface axis being integrated, one has to ask just how far such a concept can be stretched. Our approach in this particular run was to test on VIN, the gate input voltage, and DT, the time step interval. For time steps where successive values of VIN are the same, and DT is more than some large fraction of XHIGH, the surface length, then the output was calculated as simply the surface value itself evaluated at VIN and XHIGH. Thus, the output can be returned by one look up on the surface. Evidently our coding was imperfect on this run, so that further effort is needed to properly develop the response by such a procedure.

SAP 9704 SPR PULSE INPUT LOW

RUN #588 DOSE = 5.0×10^{13}

28 APR 72
08:32:00

DEPENDENT VARIABLES



LOG(TIME IN NANOSECONDS)

Figure 71. SAP Simulation of 9704 Response with Low State Electrical Input and High Dose SPR Neutron Radiation of 5.0×10^{13} .

SAP 9704 SPR PULSE INPUT HIGH

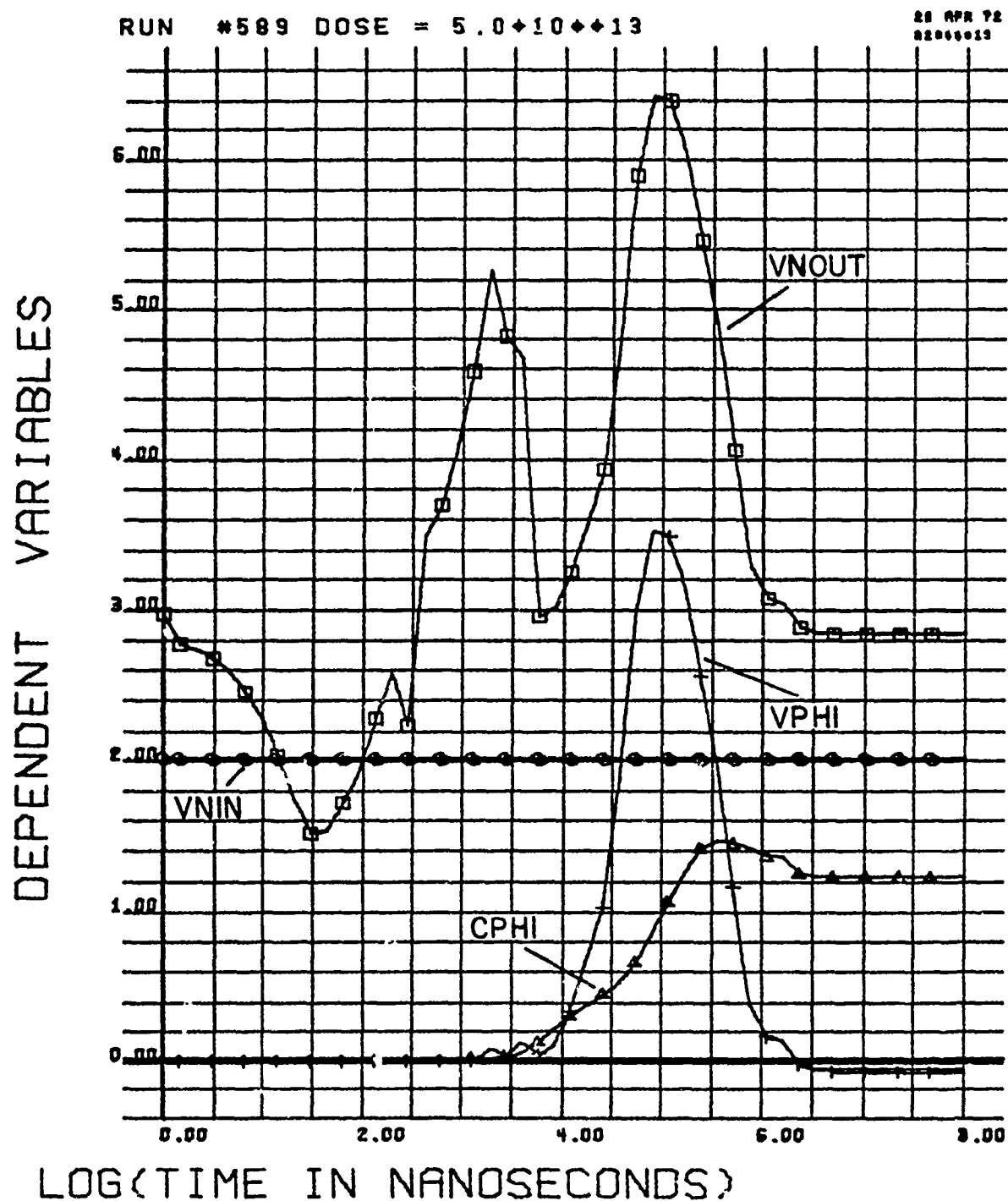


Figure 72. SAP Simulation of 9704 Response with High State Electrical Input and High Dose SPR Neutron Radiation of 5.0×10^{13} .

The last figure in the series, Figure 73, shows the calculation of the degradation of the device due to the neutron shot. This is evidenced by the fact that after the neutron burst there is no electrical output due to the turn on of an electrical input pulse at a time of 10 ms, corresponding to $\log(\text{time}) = 7$ along the plot. For convenience, the value of the Function DAMAGE as calculated during the run is also plotted on Figure 73. This should only be regarded as demonstrating a concept, rather than being actually valid for the 9704. The damage was calculated by moving out along an isochronal of the radiation-voltage transfer function of the 741, which is certainly not a hardened, dielectrically isolated device. The dosage integration and evaluation of the surface z coordinate are handled by DAMAGE, TSURF, and SURFB. The value of DAMAGE is then used as a multiplier in MAIN to destroy the gain.

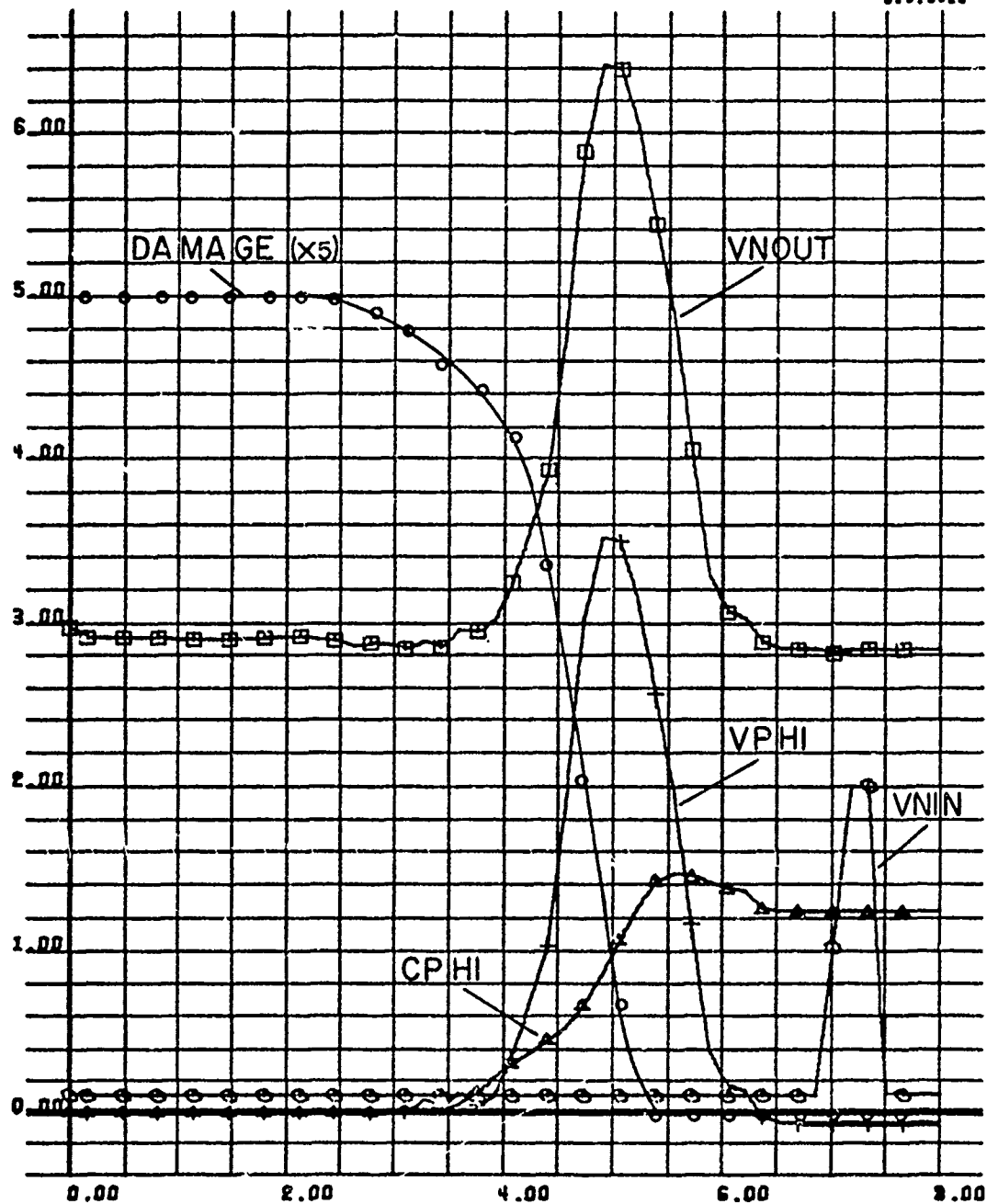
It appears then that reasonable success can be obtained with a computer based strategy which generates the system responses by retrieving the dynamical information by convolution and interpolation simultaneously on a number of multi-dimensional surfaces. The results presented in this section are the first attempted by such a strategy to calculate the effects of radiation on electrical devices. The computational characteristics are reasonably good. The execution time for run 590, the SPR simulation with delayed voltage pulse of Figure 73, was only 5.959 seconds for 50 time steps despite the fact that four surfaces were being processed during the calculation. While the effort expended in this program was only able to scratch the surface so to speak, the results are not discouraging.

SAP 9704 SPR PULSE INPUT DELAYED

RUN #590 DOSE = 5.0×10^{-13}

28 APR 78
08013020

DEPENDENT VARIABLES



LOG(TIME IN NANoseconds)

Figure 73. SAP Simulation of 9704 Response to SPR Neutron Pulse and Delayed Electrical Pulse. Absence of Response to Latter is due to DAMAGE Degradation Calculation Using 741 TPHIVN Surface of Figure 23.

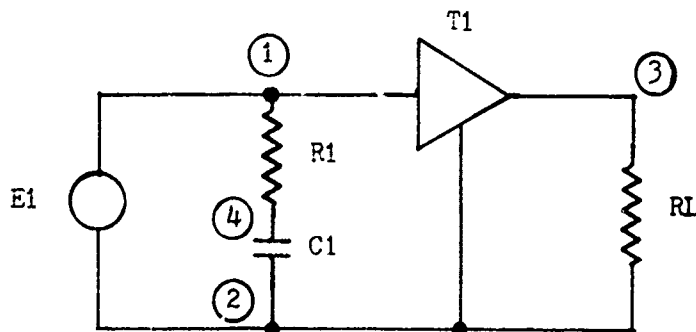
SECTION VII

MERGING SAP WITH SCEPTRE

In this section, the procedures are given by which the bivariable propagation program SAP is used with the circuit analysis program SCEPTRE. SCEPTRE was developed by AFWL under contracts AF29(601)-6489, AF29(601)-7852, F29601-67-C-0049, F29601-68-C-0117, and F29601-70-C-0038. User manuals are available from AFWL describing job deck makeups. The version of SCEPTRE which we used is an IBM 360 version modified by University Computing Company (UCC) to run on the UNIVAC 1108. This UCC version runs entirely from drum files and permits execution times typically of about 30 to 40 seconds for both Phase I and Phase II of SCEPTRE, for the cases described in the subsections below.

1. CIRCUIT MODEL

Because the surfaces contain the detailed performance information of the device, the equivalent circuit or model required to couple in with SCEPTRE is relatively simple, as shown in the sketch below.



The circuit description for SCEPTRE was:

ELEMENTS

E1,1-2 = FVOLT2(TIME)

R1,1-4 = 50.0

C1,4-2 = 1.E+10

T1,1-3-2 = MODEL 741A (TEMP)

RL,3-2 = 2.0

FVOLT2(TIME) is a function subprogram which is a modified version of VOLTIM and calculates the voltage forcing function E1 at each time step as SIMTR, the transient simulation routine written during SCEPTRE phase I, calls FVOLT2. The R1, C1 circuit on the input was mandatory simply to cause SCEPTRE to find time dependent state variables in the simulation. Otherwise the integration routines in SCEPTRE force a dummy return on all but the initialization pass, and no action is taken on subsequent time steps by SIMTR.

The model description for the 741 used was:

MODEL DESCRIPTION

MODEL 741A (TEMP) (I-O-G)

ELEMENTS

RI, I-G = 500.

JI, I - G = 0

RO, O-X = 0.1

RL, Y-G = 500.

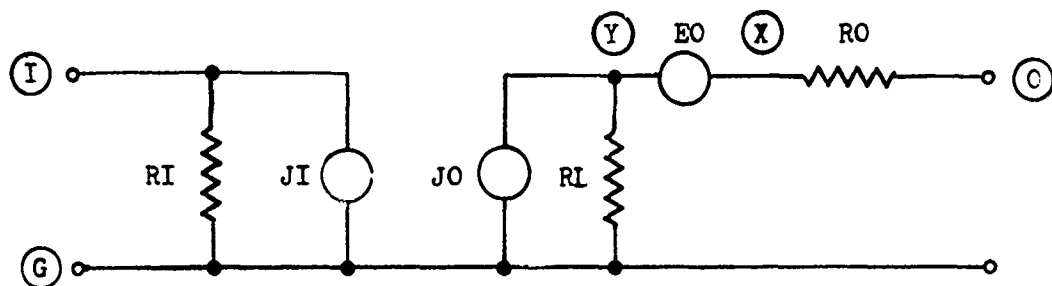
JO, Y-G = FOJ741(TIME,VJI,IRI,PJ1)

EO, X-Y = FOV741(TIME,VJI,IRI,PJ1)

DEFINED PARAMETERS

PJ1 = 1

The circuit configuration corresponding to this description is:



The function subprograms FOJ741 and FOV741 are small interfacing routines that pick up the input information from SCEPTRE, transfer it to SAP by a call to MAIN2, and are given back the output voltage and current to pass back to SIMTR in SCEPTRE. The listings of these routines are given in Figure 74. The purpose of the defined parameter in the model PJ1, is simply to identify a block number which SAP can use

P ELT FOV741,1,720130, 62163 , 1

```
000001      FUNCTION FOV741(TIME,VIN,CIN,JTAG)
000002      COMMON/VCOUT/VOUT,COUT
000003      JBLOCK = JTAG
000004      CALL MAINP(VIN,CIN,VOUTPT,COUTPT,JBLOCK,TIME)
000005      VOUT = VOUTPT
000006      COUT = COUTP
000007      FOV741 = VOUT
000008      RETURN
000009      END
```

P ELT FOJ741,1,720130, 62164 , 1

```
000001      FUNCTION FOJ741(TIME,VIN,CIN,JBLOCK)
000002      COMMON/VCOUT/VOUT,COUT
000003      FOJ741 = COUT
000004      RETURN
000005      END
```

Figure 74. F O V 7 4 1 and F O J 7 4 1 Listings

to tag the 741 block. The resistance RL in parallel with JO is needed to make SCEPTRE include the voltage generator EO in calculating the output, otherwise SCEPTRE Phase One deletes EO in writing the circuit equations.

2. PROGRAM MAKE-UP USED WITH SCEPTRE

Because of the limitations of the core size available on the UNIVAC 1108 (65K decimal) when SCEPTRE Phase Two is resident, most of the unnecessary programs were deleted. These were primarily the many plot packages and subroutines. Some functions are performed by SCEPTRE that were done by routines in SAP, and so the latter were redundant. An example of the latter is the iteration routine DRIVER, since SCEPTRE already contains iteration packages. Further, the time step itself is generated as a normal process in SCEPTRE hence those sections of VOLTIM which generated the time stepping could be deleted.

The additional routines which were written to work with SCEPTRE are:

MAIN2 - cut version of MAIN

FVOLT2 - cut version of VOLTIM, creates voltage waveform

RADGN2 - RADGEN altered for call compatibility as well as the the interface function subprograms FOJ741 and FOV741 given earlier. By deleting the plot routines, and cutting back the size of the XYZ common block which contains the surface files, all of the program code fit into about 7K decimal, and all of the data into about 10K decimal. This was small enough to be accommodated in core together with the UCC new version of SCEPTRE during Phase Two. The listings of these new routines, MAIN2, FVOLT2 and RADGN2, are given in Figures 75, 76, and 77.

Initially we attempted to adapt the United Aircraft version of SCEPTRE provided by AFWL. However the executive EXEC II on the Palo Alto Univac 1108 of UCC differed sufficiently to render the UA system routines inoperative. Similar problems arose with the tape obtained from Lockheed Missile and Space Company (LMSC) since the SLEUTH routines were available only in binary rather than symbolic form. Thus the approach we followed was to try and work with the UCC drum version of SCEPTRE despite the fact that UCC in Dallas refused to release the symbolics to the local UCC staff in Palo Alto. Thus when drum assignments had to be set to match our drum file designations for the surfaces, many phone calls were necessary between Palo Alto and Dallas to accomplish the most elementary tasks. Eventually the necessary

@ FLT FVOLT2,1,720302, 50408 . 1

```

000001      FUNCTION FVOLT2(TIN)
000002      INCLUDE TIME.LIST
000003      DIMENSION V(12), T(12)
000004      DIMENSION TUNIT(3), TSCALE(3)
000005      INTEGER OPTION,OPTLIN,OPTLOG,TUNIT,UNIT
000006      LOGICAL INITIAL
000007      DATA INITIAL/.TRUE./
000008      DATA IP,IT/5,6/
000009      DATA NANSEC/6NNANO S/,MICSEC/6MMICROS/,MILSEC/6HMILLIS/
000010      DATA OPTLOG/6HLOG /,OPTLIN/6HLINEAR/
000011
000012      C      IF(.NOT.INITIAL) GO TO 50
000013      INITIAL = .FALSE.
000014      ILO = 1
000015      MODULO = 0
000016      OLDTIM = -.001
000017      OLDLOG = 0.0
000018      TRADLG = 0.0
000019      READ (IN,10) IOPT
000020      10 FORMAT(I3)
000021      WRITE (IT,17) IOPT
000022      17 FORMAT(24H0FORCING FUNCTION OPTION =,I3)
000023      GO TO (18,30), IOPT
000024      18 READ (IA,12) A,FREQ
000025      12 FORMAT(2(E15.8,5X))
000026      WRITE (IT,19) A,FREQ
000027      19 FORMAT(34H0A=E15.8,5X,6H FREQ=,E15.8)
000028      OMEGA = 2.0*3.141592654*FREQ
000029      GO TO 40
000030      30 READ (IN,10) NSTEPS
000031      WRITE (IT,31) NSTEPS
000032      31 FORMAT(//49H A PIECEWISE LINEAR FORCE FUNCTION IS USED WITH THE NU
000033      IMBER OF STEPS = ,I3,/ 31H FUNCTION      TIME      UNIT )
000034      DO 36 I = 1, NSTEPS
000035      READ (IN,34) V(I),T(I), UNIT
000036      WRITE (IT,35) V(I),T(I), UNIT
000037      34 FORMAT(2F12.8,A6)
000038      35 FORMAT(2X,2G12.4,A6)
000039      IF (UNIT.EQ.NANSEC) TSC = 1.0
000040      IF (UNIT.EQ.MICSEC) TSC = 1.0E+3
000041      IF (UNIT.EQ.MILSEC) TSC = 1.0E+6
000042      IF (UNIT.NE.NANSEC.AND.UNIT.NE.MICSEC.AND.UNIT.NE.MILSEC)
000043      1 WRITE (IT,34)
000044      36 T(I) = T(I) * TSC
000045      38 FORMAT(5X, 50H THE UNIT MULTIPLIER IS NOT CORRECT FOR THE TIME. )
000046      NSTEP1 = NSTEPS - 1
000047      40 READ (IN,41) ISTART,TUNIT(1), ISTOP, TUNIT(2), IPADTM, TUNIT(3),
000048      1,MMAX, OPTION
000049      41 FORMAT(5X,3(I6,1X,A6,2X),5X,I5,5X,A6)
000050      WRITE (IT,41) ISTART,TUNIT(1), ISTOP, TUNIT(2), IPADTM, TUNIT(3),
000051      1,MMAX, OPTION
000052      DO 42 K = 1,3
000053      IF (TUNIT(K).EQ.NANSEC) TSCALE(K) = 1.0
000054      IF (TUNIT(K).EQ.MICSEC) TSCALE(K) = 1.0E+3
000055      IF (TUNIT(K).EQ.MILSEC) TSCALE(K) = 1.0E+6
000056      IF (TUNIT(K).NE.NANSEC.AND.TUNIT(K).NE.MICSEC.AND.TUNIT(K).NE.MILSE
000057      1C) WRITE (IT,34)
000058      42 CONTINUE

```

Figure 75a. F V O L T 2 Listing

```

000059      TSTART = ISTART * TSCALE(1)
000060      TSTART = AMAX1(TSTART,1.)
000061      TSTOP = ISTOP * TSCALE(2)
000062      RADTIM = IPADTM * TSCALE(3)
000063      IF (OPTION.EQ. OPTLIN) MODE = 1
000064      IF (OPTION.EQ. OPTLOG) MODE = 2
000065      WRITE (IT,46) OPTION
000066      IF (OPTION.NE. OPTLIN.AND. OPTION.NE. OPTLOG) STOP
000067  46 FORMAT('D',46,2) ' TIME STEPS ARE USED. / )
000068      GO TO (47,48), MODE
000069  47 A1 = TSTART
000070      A2 = (TSTOP - TSTART) / (MMAX - 1)
000071      GO TO 50
000072  48 A1 = ALOG10(TSTART)
000073      A2 = (ALOG10(TSTOP/TSTART))/(MMAX - 1)
000074  C
000075  C      ENTRY POINT ON ALL BUT FIRST CARD READ OPERATION
000076  C      MODE = 1 CORRESPONDS TO A LINEAR TIME BASE.
000077  C      MODE = 2 CORRESPONDS TO LOGARITHMIC TIME BASE.
000078  C
000079  50 TT = TIN
000080      IF (.NOT. (TT.GT. OLDTIM)) RETURN
000081      M = M + 1
000082      MPRIME = M - MODULO * LMAX
000083      IF (MPRIME.EQ. LMAX) MODULO = MODULO + 1
000084      TIME(MPRIME,1) = TT
000085      TIME(MPRIME,2) = ALOG10(AMAX1(TT,1.))
000086      DT = TIME(MPRIME,1) - OLDTIM
000087      DLOGT = TIME(MPRIME,2) - OLDLOG
000088      OLDTIM = TIME(MPRIME,1)
000089      OLDLOG = TIME(MPRIME,2)
000090      DTL(MPRIME,1) = DT
000091      DTL(MPRIME,2) = DLOGT
000092      TRAD = TIME(MPRIME,1) - RADTIM
000093      TRADLG = ALOG10(AMAX1(TRAD,1.))
000094      IHI = MPRIME
000095      IF (MODULO.GT. 0) IHI = LMAX
000096      DO 56 I = 1, IHI
000097          TL(I,1) = TIME(MPRIME,1) - TIME(I,1)
000098  56 TL(I,2) = TIME(MPRIME,2) - TIME(I,2)
000099  C
000100  58 GO TO (59,60), IOPT
000101  59 FVOLT2 = A * SIN(OMEGA*TIME(MPRIME,1))
000102      RETURN
000103  60 DO 70 I = ILO, NSTEP1
000104      IF ((TIME(MPRIME,1).GT. T(I).OR. .NOT. TIME(MPRIME,1).LT. T(I)).AND.
000105          1 TIME(MPRIME,1).LT. T(I+1)) GO TO 80
000106  70 CONTINUE
000107      I = I - 1
000108      ILO = I
000109      FVOLT2 = (V(I+1)-V(I))*(TIME(MPRIME,1)-T(I))/(T(I+1)-T(I)) + V(I)
000110  80 RETURN
000111  END

```

Figure 75b. F V O L T 2 Listing (cont.)

0 FLI MAIN2,1,720310, R37A , 1

```

000001      SUBROUTINE MAIN2(VINPUT,CINPUT,VOUTPT,COUTPT,JBLOCK,T)
000002      INCLUDE SAP,LIST
000003      INCLUDE TIME,LIST
000004      INCLUDE XYZ,LIST
000005      COMMON / IDEFUG / IDEFUG
000006      LOGICAL IDEFUG
000007      DIMENSION MNIN(IX),MOUT(IX),LHLOCK(IX,IX),LSURF(NSURF),LTIME(2)
000008      1, JPARAL(IX),FACTOR(IX),CROUT(IX),VNOUT(IX),CNIN(IX),VNIN(IX)
000009      2,LMEX(IX)
000010      LOGICAL IPADN,IPADN,INITAL,ISYM,ISYML
000011      C      ISYM = .TRUE. IMPLIES SURFACE IS SYMMETRIC WITH RESPECT TO Y = 0.
000012      C      KMAX = MAXIMUM NUMBER OF ITERATIONS.
000013      DATA EPSLN2/.01/,EPSLN3/1.0E-8/,IN/5/,IT/5/,KMAX/20/
000014      DATA LTIME/10/, VTEST/.05/, CTEST/.05/, M/0/
000015      DATA INITAL/.TRUE./,IDEFUG/.TRUE./,IXBASE/0/,IYBASE/0/,IZBASE/0/
000016      DATA NOMORE/64NOMORE/, TOLD/-.001/
000017      DATA LTIME(1)/0/, FACTOR(1)/0./
000018      DATA TIM/1.E-20/, SMALL/1.E-12/, SAP2/4HSAP2/, STAR/4H****/
000019      C
000020      C
000021      C
000022      IF(.NOT.INITAL) GO TO 80
000023      INITAL = .FALSE.
000024      C
000025      C      IF THE ROUTINE DAMAGE IS CALLED UPON, THE SURFACE ZVN741.
000026      C      MUST BE READ IN BY MAIN. THIS CORRESPONDS TO LSURF = 5.
000027      C
000028      WRITE(IT,1) STAR,SAP2,STAR,SAP2,STAR,SAP2,STAR,SAP2,STAR,SAP2,STAR
000029      1 FORMAT(1H, //,10X,5(A4,4X,A4,4X),A4 //)
000030      READ(IN,2) (LSURF(L),L=1,NSURF)
000031      2 FORMAT(12I5)
000032      WRITE(IT,3) (L,LSURF(L),L=1,NSURF)
000033      3 FORMAT(/ 2X,3(1X, 6HLSURF(,12,4H) = ,11) /)
000034      C
000035      CALLING INPUT DATA FROM SUBROUTINE DATA.
000036      CALL DATA
000037      C
000038      CALLING SURFACE INFORMATION FROM DRUM FILES.
000039      DO 13 I = 1, NSURF
000040      IF (LSURF(I).EQ.0) GO TO 13
000041      C      INPUT = I + 17
000042      INPUT = IN
000043      READ(INPUT,4) XLO,XHI,YLO,YHI,IXDIMN,IYDIMN,LL,ISYML,(SCALE(L,J)
000044      1,J=1,3)
000045      4 FORMAT(4F8.4,3I4,L6,3E10.5)
000046      IXDIM(L) = IXDIMN
000047      IYDIM(L) = IYDIMN
000048      ISYM(L) = ISYML
000049      XLOW(L) = .9999998 * XLO
000050      YLOW(L) = .9999998 * YLO
000051      XHIGH(L) = 1.0000002 * XHI
000052      YHIGH(L) = 1.0000002 * YHI
000053      XDELTA = (XHIGH(L)-XLOW(L))/(IXDIM(L)-1)
000054      YDELTA = (YHIGH(L)-YLOW(L))/(IYDIM(L)-1)
000055      WRITE(IT,5) L,XLO,XHI,YLO,YHI,XDELTA,YDELTA,IXDIMN,IYDIMN,LL,ISYML
000056      1,(L,J,SCALE(L,J),J=1,3)
000057      5 FORMAT(/13H FOR SURFACE ,12,7H, XLO =,G11.4,7H, XHI =,G11.4,7H, YL
000058      10 =,G11.4,7H, YHI =,G11.4,10H, XDELTA =,G11.4,10H, YDELTA =,G11.4,

```

Figure 76a. M A I N 2 Listing

```

000059      2/9H IXDIM =.13.9H, IYDIM =.13.5H, L =.12.9H, ISYM =.L1.5X.3(8H
000060      3SCALE(.11.1H,.11.3H) =.610.2)/)
000061      6 FORMAT(/ 10X,8HIXBASE =.13.5X,8HIYBASE =.13.5X,8HIZBASE =.15 /)
000062      C
000063      CREATING X AND Y ARRAYS FROM INPUTTED INFORMATION.
000064      C
000065      IPONTR(L,1) = IXBASE + 1
000066      8 FORMAT(/ 5X,3(7H1PONTR(.12.1H,.12.3H) =.15.1H.5X)/)
000067      DO 9 I = 1, IXDIMN
000068      9 X(I + IXBASE) = XLO + (I-1)*XDELTA
000069      IXBASE = IXBASE + IXDIMN
000070      IPONTR(L,2) = IYBASE + 1
000071      DO 10 J = 1, IYDIMN
000072      10 Y(J + IYBASE) = YLO + (J-1)*YDELTA
000073      IYBASE = IYBASE + IYDIMN
000074      IPONTR(L,3) = IZBASE + 1
000075      WRITE(IT, 8) (L,J,IPONTR(L,J),J=1,3)
000076      MINC = IXDIMN * IYDIMN
000077      MINC2 = MINC + MINC
000078      JHI = IYDIMN - 1
000079      DO 12 I = 1, IXDIMN
000080      DO 12 J = 1, JHI, 2
000081      MS1 = I + (J-1)*IXDIMN + IZBASE
000082      MS2 = MS1 + IXDIMN
000083      ML1 = MS1 + MINC2
000084      ML2 = MS2 + MINC2
000085      READ(INPUT,11) (Z(KK),KK=MS1,ML1,MINC), (Z(KKK),KKK=MS2,ML2,MINC)
000086      11 FORMAT(6F13.7)
000087      12 CONTINUE
000088      IZBASE = IZBASE + 3*MINC
000089      13 CONTINUE
000090      WRITE(IT,6) IXBASE,IYBASE,IZBASE
000091      IF(IXBASE.GT.IXLIM.OR.IYBASE.GT.IYLIM.OR.IZBASE.GT.IZLIM) STOP
000092      C
000093      DAMAGX = 1.0
000094      DAMAGN = 1.0
000095      C
000096      WRITE(IT,20)
000097      20 FORMAT(41H INPUT THE VALUE OF JMAX, USE FORMAT 13. /)
000098      READ(IN,22) JMAX
000099      22 FORMAT(I3)
000100      WRITE(IT,24) JMAX
000101      24 FORMAT(37H THE VALUE OF JMAX AS READ IN IS , I3 /)
000102      WRITE(IT,30)
000103      30 FORMAT(71H INPUT THE VALUES IN SEQUENCE OF J, CONFIG,NIN,NOUT,R,C,
000104      1 L, 4I3,3F12.4 /)
000105      DO 32 J = 1, JMAX
000106      READ(IN,34) I, ITABLE(J,1), ,NIN(J),NOUT(J),R(J),C(J),FL(J)
000107      32 WRITE(IT,34) I, ITABLE(J,1), ,NIN(J),NOUT(J),R(J),C(J),FL(J)
000108      CALL SETUP( NIN,NOUT,LXLOCK,JMAX,NMAX,JP4RAL,LMEX)
000109      33 CONTINUE
000110      34 FORMAT(4I3, 3F12.4)
000111      C
000112      IF INKALL.NE. 1, PRINTING IS SUPPRESSED. EXCEPT FOR EVERY MMODTH
000113      TIME STEP.
000114      C
000115      WRITE(IT,41)
000116      41 FORMAT(41H INPUT THE VALUE OF INKALL AND MMOD, 2I3. /)
000117      READ(IN,42) INKALL,MMOD
000118      42 FORMAT(2I3)
000119      WRITE(IT,44) INKALL,MMOD

```

Figure 76b. M A I N 2 Listing (cont.)

```

000119      44 FORMAT( 9H INKALL =.I2,10H,   MMOD =.I3   /)
000120      C
000121      READ(IN,44) EPSLON
000122      45 FORMAT(F10.6)
000123      WRITE(11,44) EPSLON
000124      48 FORMAT(32H THE ITERATION CRITERIA, EPSLON =. G14.8 /)
000125      C
000126      KROSS = 1
000127      75 CALL RADGN2(0, PHI, IRADN, IRADN)
000128      C
000129      NORMAL ENTRY ON ALL BUT INITIAL PASS.
000130      C
000131      80 CONTINUE
000132      MTRUN = M/MMOD
000133      MODM = MTRUN*MMOD/M
000134      J = JBLOCK
000135      N = NJIN(J)
000136      CALL RADGN2(1, PHI, IRADN, IRADN)
000137      C
000138      IF (IRADN.AND. PHI(1).GT.0.) PHIXL = ALOG10(PHI(1))
000139      IF (IRADN.AND. PHI(2).GT.0.) PHINL = ALOG10(PHI(2))
000140      C
000141      VNIN(N) = VININ
000142      CNIN(N) = CININ
000143      IF (ABS(VNIN(N)).LT. TINY) VNIN(N) = SMALL
000144      IF (ABS(CNIN(N)).LT. TINY) CNIN(N) = SMALL
000145      CIN(J,1) = CNIN(N)
000146      VIN(J,1) = VNIN(N)
000147      ITERAT = 1
000148      ITRSAV = ITERAT
000149      NITER = 0
000150      C
000151      207 NITER = NITER + 1
000152      ITERAT = ITRSAV + NITER
000153      209 VOLD = VOUT(J,1)
000154      COLD = COUT(J,1)
000155      IF (ABS(VOLD).LT. TINY) VOLD = SMALL
000156      IF (ABS(COLD).LT. TINY) COLD = SMALL
000157      210 VOUT(J,1) = VIN(J,1) * FE(J)
000158      COUT(J,1) = CIN(J,1) * FI(J)
000159      224 VTST = ABS ( 1.0 - VOUT(J,1)/VOLD )
000160      CTST = ABS ( 1.0 - COUT(J,1)/COLD )
000161      IF (.NOT. (ABS(VOLD).GT.0.)) VTST = 0.
000162      IF (.NOT. (ABS(COLD).GT.0.)) CTST = 0.
000163      226 IF ((VTST.GT. VTST .OR. CTST.GT. CTST) .AND. NITER.LE. LIMIT0
000164      1) GO TO 207
000165      ITERAT = ITRSAV
000166      230 IF ( ITERAT .GT. 1 ) GO TO 231
000167      IF (IRADN) VOUT(J,3) = PHIXL * FEPI(J,1)
000168      IF (IRADN) VOUT(J,3) = PHINL * FEPI(J,2)
000169      IF (IRADN) COUT(J,3) = PHIXL * FIPI(J,1)
000170      IF (IRADN) COUT(J,3) = PHINL * FIPI(J,2)
000171      IF (IRADN) DAMAGX = DAMAGE(J,1)
000172      IF (IRADN) DAMAGN = DAMAGE(J,2)
000173      DAMAGT = DAMAGX + DAMAGN
000174      IF (.NOT. (IRADN.OR. IRADN)) GO TO 231
000175      VOUT(J,1) = VOUT(J,1)*DAMAGT + VOUT(J,3)
000176      COUT(J,1) = COUT(J,1)*DAMAGT + COUT(J,3)
000177      231 CONTINUE
000178      232 VNOUT(N) = VOUT(J,1)

```

Figure 76c. M A I N 2 Listing (cont.)

```

000179      CNOUT(N) = COUT(J,1)
000180      VOUTPT = VOUT(J,1)
000181      COUTPT = COUT(J,1)
000182
000183      C
000184      440 IF(.NOT.(INKALL.EQ.1.OR.MODEM.EQ.1)) GO TO 450
000185      WRITE(IT,441)
000186      441 FORMAT(/ / 12H J   VIN(J)       CIN(J)       VOUT(J,1)   COUT(J,1)
000187      1)   VDIFF   CDIFF   VOUT(J,2)   COUT(J,2)   VOUT(J,3)   C
000188      2OUT(J,3)   )
000189      JLO = J
000190      JHI = J
000191      444 DO 445 J = JLO, JHI
000192      VDIFF = VIN(J,1)-VOUT(J,1)
000193      CDIFF = CIN(J,1)-COUT(J,1)
000194      445 WRITE(IT,446),J,VIN(J,1),CIN(J,1),VOUT(J,1),COUT(J,1),VDIFF,CDIFF
000195      1,VOUT(J,2),COUT(J,2),VOUT(J,3),COUT(J,3)
000196      446 FORMAT(13,10(1X,G11.3))
000197      450 DO 490 J = JLO, JHI
000198      DO 460 JIG = 2, JT
000199      JOG = JT - JIG + 2
000200      VIN(J,JOG) = VIN(J,JOG-1)
000201      CIN(J,JOG) = CIN(J,JOG-1)
000202      VINTGL(J,JOG) = VINTGL(J,JOG-1)
000203      CINTGL(J,JOG) = CINTGL(J,JOG-1)
000204      460 CONTINUE
000205      DO 470 JIG = 2, JD
000206      JOG = JD - JIG + 2
000207      VINDT1(J,JOG) = VINDT1(J,JOG-1)
000208      CINDT1(J,JOG) = CINDT1(J,JOG-1)
000209      470 CONTINUE
000210      480 CONTINUE
000211      RETURN
000212      END

```

Figure 76d. M A I N 2 Listing (cont.)

* ELT RADGN2,1,720302, 50500 * 1

```

000001      SUBROUTINE RADGN2 (N,PHI,IRADX,IRADN)
000002      INCLUDE TIME.LIST
000003      DIMENSION RH01(25),RH02(25),T1(25),T2(25),PHI(2)
000004      DIMENSION ITYPE1(2),ITYPE2(2),NAME1(2),NAME2(2),IWEIBL(2),
000005      IGAUSS(2),PIECE(2),IXRAY(2),NEUTRN(2)
000006      LOGICAL IRADX1,IRADN1,IRADX2,IRADN2,IRADX,IRADN,KONVOL
000007      DATA IN,11/5,6/
000008      DATA IWEIBL(1)/6HWFIMUL/,IWEIBL(2)/6HL      /,IGAUSS(1)/6HGAUSS1/,
000009      1      IGAUSS(2)/6HAN      /,IXRAY(1)/6HXRAY      /,IXRAY(2)/6H      /,
000010      2      NEUTRN(1)/6HNEUTRO/,NEUTRN(2)/6HN      /,PIECE(1)/6HPIECEW/,
000011      3      PIECE(2)/6HISE      /,IBLANK/6H      /
000012      DATA IFXR/6HFXR      /,ISPR/6HSPR      /
000013      C
000014      C
000015      C THE DISTRIBUTION FORCING FUNCTIONS ARE CALCULATED BY USING THE
000016      C FOLLOWING STATEMENT FUNCTIONS.
000017      PWEIBL(TYME,ALPHA,BETA,GAMMA) = (BETA * ALPHA ** (-BETA)*(TYME -
000018      1      GAMMA) ** (BETA-1))** (((TYME-GAMMA)/ALPHA) ** BETA)
000019      PGAUSS(TYME,A,TAU,SIGMA)=(A/(SIGMA * (2*3.14159) ** 0.5))**
000020      1      (( TYME - TAU) ** 2 / (2 * SIGMA ** 2)).
000021      PICWIS (RI,RIPI,TI,TIPI,PHIMAX,TYME) = ((RIPI - RI) * (TYME - TI)
000022      1      /(( TIPI - TI ) + RI) * PHIMAX
000023      C
000024      C DATA ARE READ WHEN N = 0 .
000025      C
000026      IF (N - 1) 5, 100, 200
000027      C INITIALIZATION
000028      5 IRADX1 = .FALSE.
000029      IRADN1 = .FALSE.
000030      IRADX2 = .FALSE.
000031      IRADN2 = .FALSE.
000032      IRADX = .FALSE.
000033      IRADN = .FALSE.
000034      KONVOL(1) = .TRUE.
000035      KONVOL(2) = .TRUE.
000036      RADTIM = 0.0
000037      ILO = 1
000038      KOUNT1 = 0
000039      KOUNT2 = 0
000040      NUMREK = 0
000041      C
000042      READ(IN,10) ITYPE1(1),ITYPE1(2),NAME1(1),NAME1(2),B1,B2,B3
000043      10 FORMAT (4A6,6X,3F10.5)
000044      IF ( NAME1(1).EQ. IFXR .OR.NAME1(1).EQ.ISPR) RADTIM = B2
000045      IF (RADTIM .GT. 0.0) RADTLG = ALOG10(RADTIM)
000046      IF(ITYPE1(1).NE.IBLANK.AND.NAME1(1).NE.IFXR.AND.NAME1(1).NE.ISPR)
000047      1 NUMBER = NUMBER + 1
000048      IF (NAME1(1) .NE. PIECE(1)) GO TO 35
000049      NSTEPS = 41
000050      WRITE (11,15) NSTEPS
000051      15 FORMAT (4H NSTEPS = ,I3)
000052      DO 25 I = 1, NSTEPS
000053      READ(IN,20) PH01(I),T1(I)
000054      20 FORMAT (2F10.5)
000055      25 WRITE(11,30) I,RH01(1),I,T1(I)
000056      30 FORMAT (4H RH01( ,I3,5H ) = ,G12.5,5X,4HT1( ,I3,5H ) = ,G12.5)
000057      NSTEP1 = NSTEPS - 1
000058      35 READ(IN,10) ITYPE2(1),ITYPE2(2),NAME2(1),NAME2(2),C1,C2,C3

```

Figure 77a. R A D G N 2 Listing

```

000059      IF (NAME2(1).EQ. IFXR .OR. NAME2(1).EQ. ISPR) RADTIM = C2
000060      IF (ITYPE2(1).NE. IRLANK. AND. ITYPE1(1).NE. ITYPE2(1). AND. NAME2(1)
000061      1.NE. IFXR. AND. NAME2(1).NE. ISPR) NUMBER = NUMBER + 1
000062      IF (NAME2(1).NE. IPIECE(1)) GO TO 60
000063      NSTEPS = C1
000064      WRITE (IT,15) NSTEPS
000065      DO 40 I = 1, NSTEPS
000066      READ (IN,20) RH02(I), T2(I)
000067      40 WRITE (IT,30) I, RH02(I), I, T2(I)
000068      NSTEP2 = NSTEPS - 1
000069      60 WRITE (IT,45) ITYPE1(1), ITYPE1(2), NAME1(1), NAME1(2), B1, B2, B3
000070      WRITE (IT,45) ITYPE2(1), ITYPE2(2), NAME2(1), NAME2(2), C1, C2, C3
000071      65 FORMAT (18H RADIATION TYPE = , 2A6, 5X, 15H DISTRIBUTION =
000072      1 2A6 / 5X
000073      2 324THF 3 RESPECTIVE PARAMETERS ARE ,3(4X,610.5))
000074      C
000075      C
000076      90 IF (ITYPE1(1).EQ. IXRAY(1)) IRADX1 = .TRUE.
000077      IF (ITYPE1(1).EQ. NEUTRN(1)) IRADN1 = .TRUE.
000078      IF (ITYPE2(1).EQ. IXRAY(1)) IRADX2 = .TRUE.
000079      IF (ITYPE2(1).EQ. NEUTRN(1)) IRADN2 = .TRUE.
000080      IF (IRADX1 .OR. IRADX2) IRADX = .TRUE.
000081      IF (IRADN1 .OR. IRADN2) IRADN = .TRUE.
000082      C
000083      C THE MEANING OF THE SUBSCRIPTS ON PHI AND KONVOL ARE AS FOLLOW
000084      C 1 = X-RAY IRRADIATION
000085      C 2 = NEUTRON IRRADIATION
000086      C
000087      IF ((NAME1(1).EQ. IFXR .AND. ITYPE2(1).EQ. IXRAY(1)) .OR.
000088      1 (NAME2(1).EQ. IFXR .AND. ITYPE1(1).EQ. IXRAY(1)) .OR.
000089      2 (NAME1(1).EQ. ISPR .AND. ITYPE2(1).EQ. NEUTRN(1)) .OR.
000090      3 (NAME2(1).EQ. ISPR .AND. ITYPE1(1).EQ. NEUTRN(1))) GO TO 900
000091      IF (NAME1(1).EQ. IFXR. OR. NAME2(1).EQ. IFXR) KONVOL(1) = .FALSE.
000092      IF (NAME1(1).EQ. ISPR. OR. NAME2(1).EQ. ISPR) KONVOL(2) = .FALSE.
000093      RETURN
000094      C WHEN N = 1, DIFFERENT FORCING FUNCTIONS ARE CALCULATED AND RETURN
000095      C THE RESPECTIVE RESULTS TO THE CALLING PROGRAM.
000096      100 NN = N
000097      TIM = TIME(MPRIME,1)
000098      K=1
000099      IF (IRADX1) GO TO 110
000100      IF (.NOT. IRADN1) GO TO 150
000101      K=2
000102      110 IF (NAME1(1).EQ. IWEIBL(1)) PHI(K) = PWEIBL(TIM ,B1,B2,B3)
000103      IF (NAME1(1).EQ. IGAUSS(1)) PHI(K) = PGAUSS(TIM ,B1,B2,B3)
000104      IF (NAME1(1).EQ. IFXR. OR. NAME1(1).EQ. ISPR) PHI(K) = B1
000105      IF (NAME1(1).NE. IPIECE(1)) GO TO 150
000106      DO 125 I = ILO,NSTEP1
000107      IF ((TIM.GT.T1(I).OR..NOT.TIM.LT.T1(I)).AND.TIM.LT.T1(I+1))GO TO130
000108      125 CONTINUE
000109      I = I - 1
000110      130 ILO = I
000111      IPI = I + 1
000112      PHI(K) = PICWTS ( RH01(I),RH01(IPI),T1(I),T1(IPI),B2,TIM)
000113      150 L = 1
000114      DELTA = 0.0
000115      IF (IRADX2) GO TO 175
000116      IF (.NOT. IRADN2) RETURN
000117      L = 2
000118      175 IF (L.EQ.K) DELTA = 1.0

```

Figure 77b. R A D G N 2 Listing (cont.)

```

000119      IF (NAME2(1) .EQ. IWEIHL(1)) PHI(L) = PWFIAL(TIM ,C1,C2,C3)
000120      1  + DELTA * PHI(K)
000121      IF (NAME2(1) .EQ. IGAUSS(1)) PHI(L) = PGAUSS(TIM ,C1,C2,C3)
000122      1  + DELTA * PHI(K)
000123      IF (NAME2(1) .EQ. IFXR .OR. NAME2(1) .EQ. ISPR) PHI(L) = C1
000124      IF (NAME2(1) .NE. IPIECE(1)) GO TO 200
000125      0) 180  I = ILO + NSTEP2
000126      IF ((TIM .GT. T2(I) .OR. .NOT. TIM .LT. T2(I)) .AND. TIM .LT. T2(I+1)) GO TO 190
000127      180 CONTINUE
000128      I = I + 1
000129      190 ILO = I
000130      IHI = I + 1
000131      PHI(L) = PICWIS ( RH02(I),RH02(IHI),T2(I),T2(IHI),C2,TIM)
000132      1  + DELTA * PHI(K)
000133      200 RETURN
000134      900 WRITE (IT,910)
000135      910 FORMAT(55H  TERMINATED BY RADGN2 DUE TO INCONSISTENT DATA CARDS.)
000136      STOP
000137      END

```

Figure 77c. R A D G N 2 Listing (cont.)

modifications were achieved with the results which will be given in a later section.

The loader or allocator on the Univac 1108 will automatically search the PCF (Program Complex File) for all necessary subroutines required by the main program, and load them at execution time. Unnecessary routines will not be loaded. All the routines required to execute SAP starting with program MAIN were stored in a FASTRAND drum file called SAP/SAP1. This included all the plot routines. The additional routines used to execute with SCEPTRE were stored in a file called SAP/SAP2. Both files were called into the PCF for execution with SCEPTRE, and the UNIVAC allocator loaded the needed routines.

In the case of operating with the CDC 6600, it was necessary to split the program into three files:

- 1) routines used in execution of SAP without SCEPTRE, and consisting primarily of plot subroutines,
- 2) routines required only while executing SAP with SCEPTRE, and
- 3) routines required both in execution of SAP alone, or of SAP in conjunction with SCEPTRE.

Thus if SAP alone were to be executed, the first and last program files would be loaded; while if SAP was to be executed with SCEPTRE, the last two files would be loaded. Those original SAP routines which were altered or rewritten for SCEPTRE thus would be in the first file, while the modified versions were in the second file. The routines which were unchanged, such as FE, FI, CONVOL, TSURF, SURFB, DAMAGE, SETUP, FEPHI, etc. were in the third file.

A major problem encountered in adapting FIT3D and SAP to the CDC 6600 arose due to the difference in the manner in which the UNIVAC 1108 and the CDC 6600 treat division by zero. The UNIVAC 1108 FORTRAN V compiler of UCC sets the result of dividing by zero equal to zero. The CDC compiler sets the result equal to an "indefinite" quantity. Whenever this indefinite result is subsequently used in a calculation, the new result is set indefinite, etc., etc. The propagation of the indefinite quantity throughout the rest of the entire computation usually does not take long. If overflow occurs, a similar phenomena takes place. Thus it is mandatory to test every denominator which may conceivably go to zero during a calculation, and reset it to some acceptable small value if zero does occur.

Since the basic computational strategy of SAP involves the ratio of an output to input stimuli in the transfer function, and a zero value of the input stimuli is

perfectly acceptable, such zeroes in the denominator can and did occur. The output is usually well defined in such a situation. For example for the 741 op amp, a zero input leads to a zero output. This happens to be precisely the action taken by the UNIVAC 1108 Fortran, and so we did not experience any difficulty, due to this class of problem. However, for machines which treat the case differently, it is wise to test every possible denominator which may go to zero in order to get around the action otherwise taken by the system.

3. RESULTS CALCULATED WITH SCEPTRE

Using the Model Description and Circuit Description given above, and with Outputs and Run Controls as follows:

OUTPUTS

E1, IE1, VRL, IRL, PLOT

RUN CONTROLS

STOP TIME = 4E+7

STOP

SCEPTRE Phase One generated the SIMUL8 and SIMTR programs given in the listings of Figures 78 and 79.

The output voltage as calculated by SAP using the voltage response surface TE741 of Figure 20 is plotted in Figure 80. A few pages of the output listing from SAP for the run is given in Figure 81, while the output as printed by SCEPTRE is given in Figure 82. The running sums of the convolution process are printed by CONVOL in the output of Figure 81 every tenth time step. The oldest partitions occur first with a contribution to the output given under the column heading marked TEMP. The value of the surface contribution for the partitions is given in the column labeled TSURF. The running sum is added successively along the column labeled SUM. The time that has passed since the arrival of the partition is given in the column TIMER. UIN is the value of the stimulus variable for the partition. DTL is the time step interval coming in from the iteration routines in SCEPTRE. Note that these steps are alternating in size, and are different than the time step interval printed out in the SCEPTRE printout of Figure 82.

The entries printed out by SAP at each time step are the block input/output voltages and currents. Since no radiation was acting, the $UT(J,3)$ terms are

```

0005 000005 X4220      0005 000003 XR550      0004 000042 XSAVE      0004 000001 XSTOP
0004 000040 XSTOP42    0004 000004 XTICSS      0004 000043 XTMON      0004 000041 XTRAN
0004 000024 XYTIME5

00101 1* SUBROUTINE SIMTR (KWAY,KSWIT)
00103 2* IMPLICIT REAL (A-J,L-M,O-Z). INTEGER (K,N)
00103 3* C SCRIPTURE/1108 CIRCUIT ANALYSIS PROGRAM
00104 4* COMMON /TAPES/INOUTP,NTAPE,NLITP,NOUTP,NEDITP,NSAVTP
00105 5* COMMON /CUTPOL/TIME,XSTOP,XMREF,XIR,XTICSS,XMVISS,XMXISS,XMNAIE,
00105 6* *XMXAIE,XIC,XVFPNO,XXPAS,XMURIE,XAXRUE,XVHISO,XMXICP,XICREF,XICAER
00105 7* *XMXOTP,XXTIMS,XYDIMS,XICPAS,XNUPG,XNTNO,XNMHAD,XNDFEQ,XERT,XNRE
00105 8* *PN,XCNTNU,XSTPNO,XPASNO,XRUVNO,XSTPSZ,XTRANS,XSAVE ,XTMON,SKIP(
00105 9* *4)
00106 10* DIMENSION CTPOLS(70)
00107 11* COMMON /NAVES/E1,FOT1,C1,ROT1,RL,R1,RT1,RLT1,JIT1,JOT1
00110 12* COMMON /VOLTAGS/VE1,VEOT1,VC1,VROT1,VRL,VR1,VRT1,VRLT1,VJIT1,VJOT1
00111 13* COMMON /CURRENTS/IE1,IFOT1,IC1,IROT1,IRL,IR1,IRIT1,IRLT1,IJIT1,IJOT1
00112 14* DIMENSION X4220( 3),XC440( 1),XR550( 2),XE7( 2),XJ8( 2)
00113 15* DIMENSION VP( 3),V4( 1),V5( 2),V7( 2),V8( 2)
00114 16* DIMENSION IP( 3),I4( 1),I5( 2),I7( 2),I8( 2)
00115 17* EQUIVALENCE (VP1,VP),(VC1,V4),(VROT1,V5),(VE1,V7),(VJIT1,V8)
00116 18* EQUIVALENCE (IR1,IP),(IC1,I4),(IROT1,I5),(IE1,I7),(IJIT1,I8)
00117 19* EQUIVALENCE (R1,X4220),(C1,XC440),(ROT1,XR550),(E1,XE7),(JIT1,XJ8)
00120 20* COMMON /SAVEIC/V4IC( 1)
00121 21* COMMON /SAVICI/V4ICT( 1)
00122 22* DIMENSION V4DOT( 1)
00123 23* COMMON /DEPVS/DC1
00124 24* COMMON /UTILITY/SAVETR(H, 1)
00125 25* EQUIVALENCE (DC1,V4DOT)
00126 26* DIMENSION PRAMTR( 1)
00127 27* COMMON /FRMTRS/PJIT1
00130 28* EQUIVALENCE (PRAMTR,PJIT1)
00131 29* COMMON /OUTPTS/OUTPOST(5, 4)
00132 30* COMMON /OUTPUT/TPINDX( 5)
00133 31* COMMON /OUTDTA/OUTREF( 5)
00134 32* EQUIVALENCE (CTPOLS,TIME)
00135 33* DATA NCLIC/ 0/,NCL3C/ 0/,NCL4C/ 1/,NCL6C/ 0/,NOPD/ 0/
00135 34* C TRANSIENT INITIALIZATION
00143 35* DIMENSION X4220( 3)
00144 36* DIMENSION V2P( 3)
00145 37* IF ( KSWIT.NE.0 ) GO TO 1
00147 38* X4220( 1)=1
00150 39* X4220( 2)=RT1
00151 40* X4220( 3)=RLT1+ROT1+RL
00151 41* C TRANSIENT EQUATIONS
00152 42* 1 CONTINUE
00152 43* C V2P=-.025*XR550*(.035*IP3+.045*IR1)+.024*V4-.027*E7
00153 44* F1=FVOT1*(T1+1)
00154 45* V2P( 1)=-VC1-F1
00155 46* V2P( 2)=-F1
00156 47* VJIT1=-F1
00157 48* FOT1=FOT1*(TIME,VJIT1,IRIT1,PJIT1)
00160 49* JOT1=JOT1*(TIME,VJIT1,IRIT1,PJIT1)
00161 50* V2P( 3)=-ROT1*(+JOT1)+RL*(-JOT1)+EOT1
00162 51* DO 10 N= 1 , 3
00165 52* 10 I2(N)=V2P(N)/X4220(N)
00167 53* DO 20 N= 1 , 3
00172 54* 20 V2(N)=I2(N)*X4220(N)
00172 55* C I5=.025*IP2+.035*IP3+.045*IR1
00172 56* I4OT1=+IRLT1+JOT1

```

Figure 78. S I M T R Listing

```

00175 57*      IPL=-IPLT1-JOT1
00176 58*      DO      30      N=      1      ,      2
00201 59*      30      VS(N)=1-(N)*XMS50(N)
00201 60*      C      I4=R141*11+.261*12+.0341*13+.4841*J8
00203 61*      IC1=+IR1
00204 62*      DO      40      N=      1      ,      1
00207 63*      40      V4DOT(1)=14(1)/XC440(N)
00211 64*      PF1000N
00212 65*      END

```

Figure 78b. S I M T R Listing (cont.)

```

00101 1*      SUBROUTINE SIMUL8(KWAY)
00103 2*      IMPLICIT REAL (A-J,L-M,O-Z), INTEGER (K,N)
00103 3*      C      SCRIPTURE/1108 CIRCUIT ANALYSIS PROGRAM
00104 4*      COMMON /TAPES/NOUITP,NTAPE,NLIBTP,NOUTP,NEOITP,NSAVTP
00105 5*      COMMON /CTRLS/TIME,XSTOPT,XMXERT,XIR,XTISSS,XMNISS,XMXISS,XMNAIE,
00105 6*      *XMXAIE,XIC,XREFNO,XMXPAS,XMNPIC,XMAXIE,XMNTSO,XMATIC,XICRER,XICAER
00105 7*      *XMXOTP,XNDIMS,XYDIMS,XICPAS,XNUPRO,XNTNDX,XNHEAD,XNDFEQ,XERT,XNRE
00105 8*      *RN,XCNTJU,XSTPNO,XPASNO,XRINNO,XSTPSZ,XTRANS,XSAVE      ,XTMON,SKIP(
00105 9*      *4)
00106 10*      DIMENSION CTROLS(70)
00107 11*      COMMON /NAMES/E1,EOT1,C1,ROT1,PL,R1,RIT1,RLT1,JIT1,JOT1
00110 12*      COMMON /VOLTGS/VE1,VEOT1,VC1,VROT1,VPL,VR1,VPIT1,VRLT1,VJIT1,VJOT1
00111 13*      COMMON /CURNTS/IE1,IEOT1,IC1,IROT1,IPL,IR1,IRIT1,IRLT1,IJIT1,IJOT1
00112 14*      DIMENSION XR220( 3),XC440( 1),XR550( 2),XE7( 2),XJ8( 2)
00113 15*      DIMENSION V2( 3),V4( 1),V5( 2),V7( 2),V8( 2)
00114 16*      DIMENSION I2( 3),I4( 1),I5( 2),I7( 2),I8( 2)
00115 17*      EQUIVALENCE (V2,V2),(VC1,V4),(VROT1,V5),(VE1,V7),(VJIT1,V8)
00116 18*      EQUIVALENCE (I2,I2),(IC1,I4),(IROT1,I5),(IE1,I7),(IJIT1,I8)
00117 19*      EQUIVALENCE (R1,XR220),(C1,XC440),(ROT1,XR550),(E1,XE7),(JIT1,XJ8)
00120 20*      COMMON /SAVEIC/V4ICT( 1)
00121 21*      COMMON /SAVEICT/V4ICT( 1)
00122 22*      DIMENSION V4DOT( 1)
00123 23*      COMMON /DEFIVS/DC1
00124 24*      COMMON /UT1(1)/SAVECTR(8, 1)
00125 25*      EQUIVALENCE (DC1,V4DOT)
00126 26*      DIMENSION PRAMTR( 1)
00127 27*      COMMON /PRAMTR/PJIT1
00130 28*      EQUIVALENCE (PRAMTR,PJIT1)
00131 29*      COMMON /OUTPTS/OTPOST(5, 4)
00132 30*      COMMON /OUTPUT/TPINDX( 5)
00133 31*      COMMON /OUTDTA/OUTREF( 5)
00134 32*      EQUIVALENCE (CTRLS,TIME)
00135 33*      DATA NCL1C/ 0/,NCL3C/ 0/,NCL4C/ 1/,NCL6C/ 0/,NDPD/ 0/
00135 34*      C      READ IN INPUT DATA
00143 35*      READ (NSAVTP,7000)XF7,XC440,XR550,XR220,XJ8
00171 36*      READ (NSAVTP,7000)V7,V4,V5,V2,V8
00217 37*      READ (NSAVTP,7000)I7,I4,I5,I2,I8
00245 38*      READ (NSAVTP,7000)V4ICT

```

Figure 79a. S I M U L 8 Listing

```

00253 39*      READ (NSAVTP,7000)V4DOT
00261 40*      READ (NSAVTP,7000)PRAMTR
00267 41*      READ (NSAVTP,7400)OTROST
00275 42*      READ (NSAVTP,7400)TPINDX
00303 43*      READ (NSAVTP,7400)ENDCPD
00306 44*      7000 FORMAT (5E14.7)
00307 45*      7100 FORMAT (12A6)
00310 46*      7200 FORMAT (1H0/(1X,1P2F14.7))
00311 47*      7400 FORMAT (12A6)
00312 48*      7500 FORMAT (1H0/(1X,12A6))
00313 49*      7600 FORMAT (1H0/(1X,12A6))
00314 50*      CALL TPFAD
00314 51*      C      LOAD APPROPRIATE INITIAL CONDITIONS DATA
00315 52*      CALL XLOAD(V1IC,V1ICT,V1,NCL1C,I3IC,I3ICT,I3,NCL3C,V4IC,V4ICT,V4,N
00315 53*      *CL4C,I6IC,I6ICT,I6,NCL6C, 0,NWAY)
00315 54*      C      TRANSIENT INITIALIZATION
00316 55*      CALL SIMTR(KWAY,0)
00317 56*      9000 XPASND=XPASND+1.
00320 57*      CALL SIMTR(KWAY,1)
00321 *DIAGNOSTIC* THE TEST FOR EQUALITY BETWEEN NON-INTEGERS MAY NOT BE MEANINGFUL.
00321 58*      IF (XSTPND.EQ.0.) GO TO 9050
00323 59*      9010 CALL INTGRR(NOGO,I3,NCL3C,V4,NCL4C,PRAMTR,NDPD)
00324 60*      IF (NOGO.EQ.0) GO TO 9000
00324 61*      C      LOAD OUTPUT REQUESTS
00326 62*      9050 OUTFR ( 1)=TIME
00327 63*      OUTFR ( 2)=E1
00330 64*      IE1=-IR1-J2IT1-JIT1
00331 65*      OUTFR ( 3)=IE1
00332 66*      OUTFR ( 4)=VRL
00333 67*      OUTFR ( 5)=IRL
00334 68*      CALL OUTP(KWAY)
00335 69*      IF (KWAY.EQ.0) GO TO 9010
00335 70*      C      WRITE OUT THE COMPUTED VALUES
00337 71*      9090 REWIND NEDITP
00340 72*      WRITE (NEDITP,7001)CTRULS
00346 73*      WRITE (NEDITP,7001)XE7,XC44D,XR55D,XR22D,XJB
00374 74*      WRITE (NEDITP,7001)V7,V4,V5,V2,V8
00422 75*      WRITE (NEDITP,7001)I7,I4,I5,I2,I8
00450 76*      WRITE (NEDITP,7001)V4IC
00456 77*      WRITE (NEDITP,7001)V4DOT
00464 78*      WRITE (NEDITP,7001)PRAMTR
00472 79*      WRITE (NEDITP,7400)OTRUST
00500 80*      WRITE (NEDITP,7400)TPINDX
00506 81*      WRITE (NEDITP,7400)ENDCPD
00511 82*      7001 FORMAT (1P5F14.7,10X)
00511 83*      C      LOAD APPROPRIATE TRANSIENT DATA
00512 84*      CALL XLOAD(V1IC,V1ICT,V1,NCL1C,I3IC,I3ICT,I3,NCL3C,V4IC,V4ICT,V4,N
00512 85*      *CL4C,I6IC,I6ICT,I6,NCL6C, 2,NWAY)
00512 86*      C      CHECK FOR PROGRAM SAVE
00513 87*      IF (KWAY.NE.9) GO TO 9100
00515 88*      CALL SAVFPD(1)
00516 89*      KWAY=0
00517 90*      GO TO 9010
00520 91*      9100 RETURN
00521 92*      FND

```

END OF UCC 1104 FORTRAN V COMPILATION. 1 *DIAGNOSTIC* MESSAGE(S)

PHASE 1 TIME 00100.276
 PHASE 2 TIME 00100.019

Figure 79b. S I M U L 8 Listing (cont.)

PLUT OF ϵ_1 VS TIME

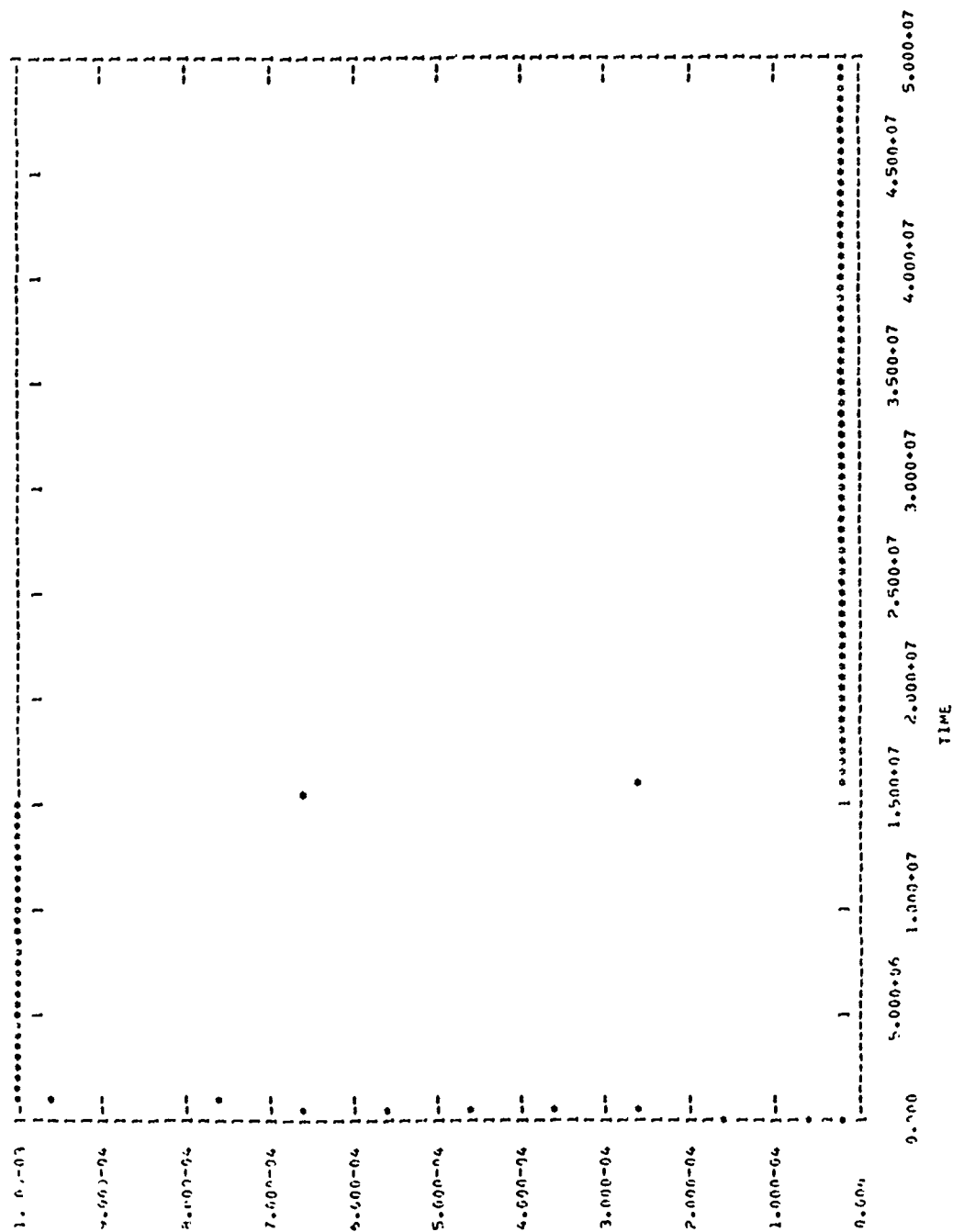


Figure 80a. SCEPTRE Plot of Input Voltage versus Time.

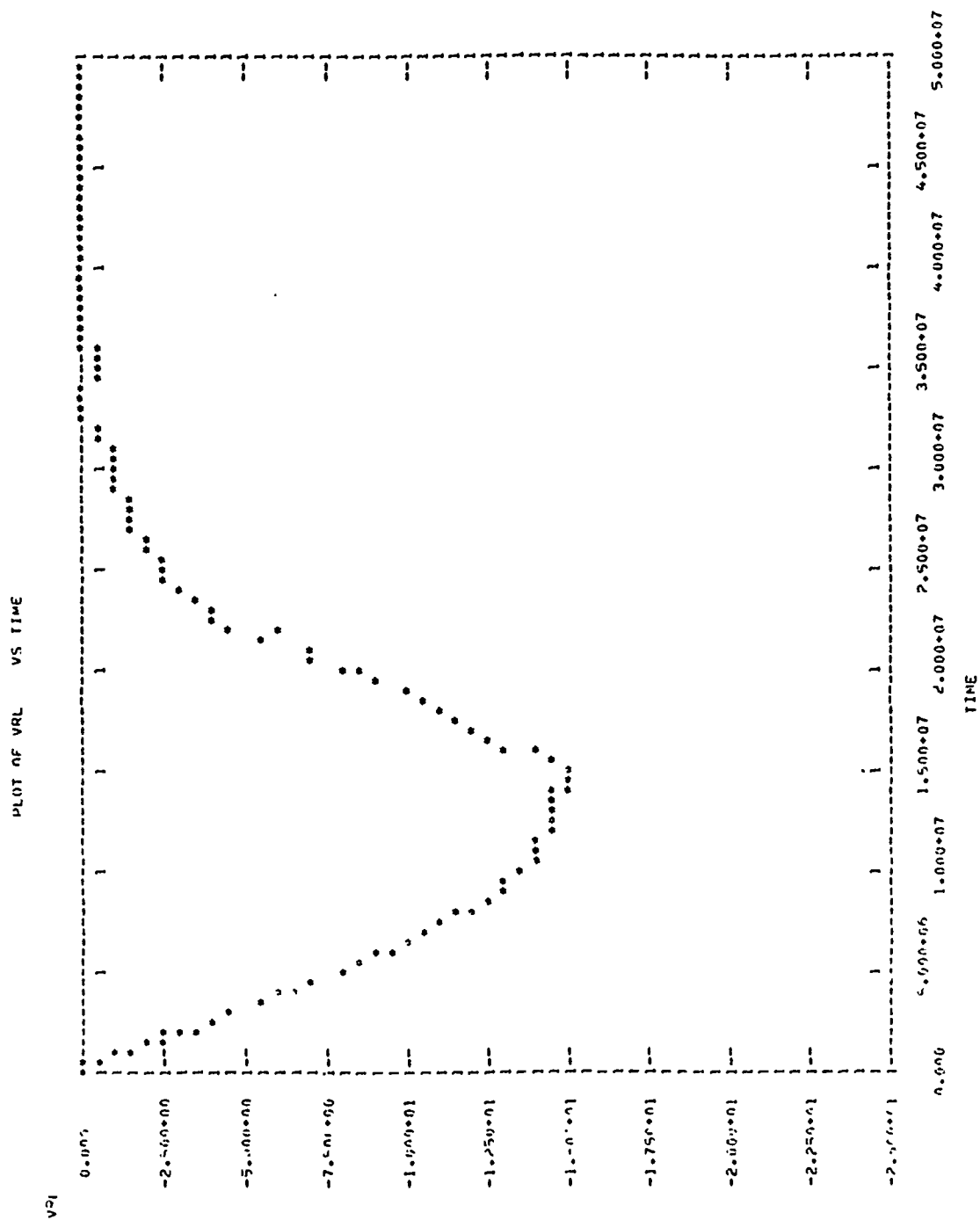


Figure 80b. SCEPTRE Plot of Output Voltage versus Time.


```

J  VT(J)  C1N(J)  VOUT(J,1)  COUT(J,1)  VOIFF  COIFF  VOUT(J,2)  COUT(J,2)  VOUT(J,3)  COUT(J,3)
1  -1.10E-04  .160-11  .142-02  .915-03  -.143-02  -.915-03  .000  .000  .000  .000
1-41-06(1) =====

```

TIME STEP NOT TOO SMALL. WILL OVERFLOW 11 ARRAYS.

```

J  VT(J)  C1N(J)  VOUT(J,1)  COUT(J,1)  VOIFF  COIFF  VOUT(J,2)  COUT(J,2)  VOUT(J,3)  COUT(J,3)
1  -1.10E-04  -.200-07  .144-02  .725-03  -.145-02  -.725-03  .000  .000  .000  .000

J  VT(J)  C1N(J)  VOUT(J,1)  COUT(J,1)  VOIFF  COIFF  VOUT(J,2)  COUT(J,2)  VOUT(J,3)  COUT(J,3)
1  -1.22E-04  -.200-07  .342-02  .216-02  -.344-02  -.216-02  .000  .000  .000  .000

J  VT(J)  C1N(J)  VOUT(J,1)  COUT(J,1)  VOIFF  COIFF  VOUT(J,2)  COUT(J,2)  VOUT(J,3)  COUT(J,3)
1  -5.45E-04  -.447-07  .142-01  .759-02  -.143-01  -.759-02  .000  .000  .000  .000

J  VT(J)  C1N(J)  VOUT(J,1)  COUT(J,1)  VOIFF  COIFF  VOUT(J,2)  COUT(J,2)  VOUT(J,3)  COUT(J,3)
1  -1.43E-03  -.113-06  .230-01  .121-01  -.231-01  -.121-01  .000  .000  .000  .000

J  VT(J)  C1N(J)  VOUT(J,1)  COUT(J,1)  VOIFF  COIFF  VOUT(J,2)  COUT(J,2)  VOUT(J,3)  COUT(J,3)
1  -1.15E-03  -.164-06  .625-01  .324-01  -.626-01  -.324-01  .000  .000  .000  .000

J  VT(J)  C1N(J)  VOUT(J,1)  COUT(J,1)  VOIFF  COIFF  VOUT(J,2)  COUT(J,2)  VOUT(J,3)  COUT(J,3)
1  -1.43E-03  -.317-06  .780-01  .397-01  -.782-01  -.397-01  .000  .000  .000  .000

J  VT(J)  C1N(J)  VOUT(J,1)  COUT(J,1)  VOIFF  COIFF  VOUT(J,2)  COUT(J,2)  VOUT(J,3)  COUT(J,3)
1  -2.57E-03  -.364-06  .134  .689-01  -.135  -.689-01  .000  .000  .000  .000

J  VT(J)  C1N(J)  VOUT(J,1)  COUT(J,1)  VOIFF  COIFF  VOUT(J,2)  COUT(J,2)  VOUT(J,3)  COUT(J,3)
1  -2.2E-03  -.815-06  .154  .777-01  -.154  -.777-01  .000  .000  .000  .000

J  VT(J)  C1N(J)  VOUT(J,1)  COUT(J,1)  VOIFF  COIFF  VOUT(J,2)  COUT(J,2)  VOUT(J,3)  COUT(J,3)
1  -1.54E-03  -.544-06  .219  .111  -.219  -.111  .000  .000  .000  .000

```

v = 10. uoajmc = 10 IL04 = 1 IHIGH = 0 LSURF = 3

```

L  Time2  UIN  DT1  TFAP  SUM  TSURF
2  .343-06  -.224-04  .125-05  -.530-03  .000  .120-02
3  .325-06  -.545-04  .374-05  -.422-02  -.530-03  .189-02
4  .300-06  -.442-04  .250-05  -.347-02  -.475-02  .189-02
5  .255-06  -.151-03  .150-05  -.223-01  -.472-02  .187-02
6  .200-06  -.111-03  .250-05  -.454-02  -.310-01  .186-02
7  .124-06  -.257-03  .750-05  -.344-01  -.394-01  .184-02
8  .100-06  -.272-03  .250-05  -.130-01  -.751-01  .184-02
9  .250-06  -.344-03  .750-05  -.441-01  -.481-01  .184-02

```

Figure 81b. SAP Printout during SCEPTRE run (cont.)

J	1	VIN(J)	CE(J)	VOUT(J,1)	COU(J,1)	VOIFF	COIFF	VOUT(J,2)	COU(J,2)	VOUT(J,3)	COU(J,3)
		-0.31-03	-0.717-06	.242	.122	-.242	-.122	.000	.000	.000	.000
J	1	VIN(J)	CE(J)	VOUT(J,1)	COU(J,1)	VOIFF	COIFF	VOUT(J,2)	COU(J,2)	VOUT(J,3)	COU(J,3)
		-0.455-03	-0.702-06	.372	.163	-.322	-.163	.000	.000	.000	.000
J	1	VIN(J)	CE(J)	VOUT(J,1)	COU(J,1)	VOIFF	COIFF	VOUT(J,2)	COU(J,2)	VOUT(J,3)	COU(J,3)
		-0.440-03	-0.911-06	.352	.177	-.352	-.177	.000	.000	.000	.000
J	1	VIN(J)	CE(J)	VOUT(J,1)	COU(J,1)	VOIFF	COIFF	VOUT(J,2)	COU(J,2)	VOUT(J,3)	COU(J,3)
		-0.554-03	-0.960-06	.457	.231	-.458	-.231	.000	.000	.000	.000
J	1	VIN(J)	CE(J)	VOUT(J,1)	COU(J,1)	VOIFF	COIFF	VOUT(J,2)	COU(J,2)	VOUT(J,3)	COU(J,3)
		-0.576-03	-0.111-05	.447	.250	-.497	-.250	.000	.000	.000	.000
J	1	VIN(J)	CE(J)	VOUT(J,1)	COU(J,1)	VOIFF	COIFF	VOUT(J,2)	COU(J,2)	VOUT(J,3)	COU(J,3)
		-0.553-03	-0.110-05	.527	.316	-.627	-.316	.000	.000	.000	.000
J	1	VIN(J)	CE(J)	VOUT(J,1)	COU(J,1)	VOIFF	COIFF	VOUT(J,2)	COU(J,2)	VOUT(J,3)	COU(J,3)
		-0.474-03	-0.131-05	.672	.339	-.672	-.339	.000	.000	.000	.000
J	1	VIN(J)	CE(J)	VOUT(J,1)	COU(J,1)	VOIFF	COIFF	VOUT(J,2)	COU(J,2)	VOUT(J,3)	COU(J,3)
		-0.752-03	-0.130-05	.803	.405	-.804	-.405	.000	.000	.000	.000
J	1	VIN(J)	CE(J)	VOUT(J,1)	COU(J,1)	VOIFF	COIFF	VOUT(J,2)	COU(J,2)	VOUT(J,3)	COU(J,3)
		-0.592-03	-0.150-05	.881	.443	-.881	-.443	.000	.000	.000	.000
J	1	VIN(J)	CE(J)	VOUT(J,1)	COU(J,1)	VOIFF	COIFF	VOUT(J,2)	COU(J,2)	VOUT(J,3)	COU(J,3)
		-0.450-03	-0.160-05	1.17	.591	-1.17	-.591	.000	.000	.000	.000

q = 20. voutwf = 20 l04 = 1 l4lgh = 10 (SURF = 3)

L	TIME	UIN	NTL	TEMP	SUM	TSURF
2	0.000-00	-0.224-04	.175-05	-.532-03	.000	.190-02
3	0.050-00	-0.507-04	.175-05	-.423-02	-.532-03	.190-02
4	0.100-00	-0.462-04	.250-05	-.309-02	-.476-02	.149-02
5	0.150-00	-0.150-03	.250-05	-.222-01	-.675-02	.107-02
6	0.200-00	-0.143-03	.250-05	-.044-02	-.310-01	.106-02
7	0.250-00	-0.257-03	.250-05	-.356-01	-.345-01	.144-02
8	0.300-00	-0.202-03	.250-05	-.130-01	-.752-01	.104-02
9	0.350-00	-0.346-03	.250-05	-.491-01	-.841-01	.104-02
10	0.400-00	-0.341-03	.250-05	-.175-01	-.137	.104-02
11	0.450-00	-0.457-03	.250-05	-.631-01	-.155	.104-02
12	0.500-00	-0.443-03	.250-05	-.722-01	-.218	.104-02
13	0.550-00	-0.550-03	.250-05	-.765-01	-.240	.104-02
14	0.600-00	-0.570-03	.250-05	-.660-01	-.317	.104-02
15	0.650-00	-0.630-03	.250-05	-.691-01	-.343	.102-02
16	0.700-00	-0.670-03	.250-05	-.307-01	-.432	.101-02
17	0.750-00	-0.752-03	.250-05	-.301	-.463	.100-02
18	0.800-00	-0.807-03	.400-05	-.722-01	-.564	.100-02

Figure 81c. SAP Printout during SCPTRE run (cont).

FUNCTION FUNCTION UNIT = 2

A PIECE-WISE LINEAR FORCE FUNCTION IS USED WITH THE NUMBER OF STEPS = 5

FUNCTION	TIME	UNIT
.1657-64	.0000	MMILIS
.1663-62	1.000	MMILIS
.1663-02	15.00	MMILIS
.1666-04	16.00	MMILIS
.1666-64	50.00	MMILIS
5 MILLIS	50 MILLIS	
LINEAR TIME STEPS ARE USED.		
	0 MILLIS	101 LINEAR

Figure 81f. SAP Printout during SCEPTRE Run (cont) .

TRANSIENT SOLUTION CONTROLS AFTER SIMULATION

INTEGRATION ROUTINE	XPO
CURRENT SIMULATION TIME	5.0150000+07
INTEGRATION STEP COUNTER	1.3600000+02
INTEGRATION PASS COUNTER	2.7100000+02
SOLUTION EXECUTION TIME (ELAPSED)	1.2433333+00
TERMINATION CONDITION	STOP TIME EXCEEDED (NORMAL STOP)

Figure 82a. SCEPTRE Printout during SCEPTRE Run.

TRANSIENT ANALYSIS RESULTS

TIME	0.0000	5.0000E-04	1.5000E-03	2.5000E-05	7.5000E-05	4.5000E-05	5.5000E-05
F1	1.0000E-05	5.0000E-05	1.5000E-04	2.5000E-06	7.5000E-06	4.5000E-06	5.5000E-06
F2	2.0000E-07	1.0000E-06	3.0000E-06	5.0000E-08	1.5000E-07	4.5000E-07	5.5000E-07
V1	-1.4400E-03	-1.5774E-02	-6.2667E-02	-1.3740E-01	-2.2262E-01	-3.2794E-01	-4.6204E-01
I1	-7.2424E-04	-7.3454E-03	-3.2837E-02	-6.8900E-02	-1.1131E-01	-1.5293E-01	-2.3104E-01
TIME	6.5000E-05	7.5000E-05	9.5000E-05	1.5000E-04	1.7500E-04	1.7500E-04	1.7500E-06
F1	6.5000E-06	7.5000E-06	9.5000E-06	1.5000E-05	1.7500E-05	1.7500E-05	1.7500E-03
F2	1.4377E-05	1.6550E-05	2.0110E-05	2.2600E-05	2.2600E-05	2.2600E-05	2.1999E-05
V1	-6.1210E-01	-8.6940E-01	-1.1017E-01	-1.5443E-01	-0.2932E-01	-2.2631E-01	-2.6358E-01
I1	-3.1665E-01	-4.0474E-01	-5.9045E-01	-7.8141E-01	-9.6459E-01	-1.1415E-01	-1.3179E-01
TIME	1.5000E-04	2.5000E-04	2.5000E-04	2.5000E-06	3.5000E-06	3.7500E-04	4.1500E-06
F1	1.5000E-05	2.5000E-05	2.5000E-05	2.5000E-07	3.5000E-07	3.7500E-07	4.1500E-03
F2	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05
V1	-2.5444E-01	-3.1665E-01	-4.0474E-01	-5.9045E-01	-7.8141E-01	-9.6459E-01	-1.1415E-01
I1	-1.2631E-01	-1.5443E-01	-1.8177E-01	-2.0110E-01	-2.2600E-01	-2.2600E-01	-2.2600E-01
TIME	4.5000E-04	4.5000E-04	5.5000E-04	5.5000E-04	5.5000E-04	6.5000E-04	6.5000E-06
F1	4.5000E-05	4.5000E-05	5.5000E-05	5.5000E-05	5.5000E-05	6.5000E-05	6.5000E-03
F2	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05
V1	-7.2137E-01	-7.2137E-01	-7.2137E-01	-7.2137E-01	-7.2137E-01	-7.2137E-01	-7.2137E-01
I1	-3.5045E-01	-3.5045E-01	-3.5045E-01	-3.5045E-01	-3.5045E-01	-3.5045E-01	-3.5045E-01
TIME	7.5000E-04	7.5000E-04	8.5000E-04	8.5000E-04	8.5000E-04	9.5000E-04	9.5000E-06
F1	7.5000E-05	7.5000E-05	8.5000E-05	8.5000E-05	8.5000E-05	9.5000E-05	9.5000E-03
F2	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05
V1	-1.1665E-01	-1.1665E-01	-1.1665E-01	-1.1665E-01	-1.1665E-01	-1.1665E-01	-1.1665E-01
I1	-5.5313E-01	-5.5313E-01	-5.5313E-01	-5.5313E-01	-5.5313E-01	-5.5313E-01	-5.5313E-01
TIME	1.0150E-07	1.0550E-07	1.0950E-07	1.1350E-07	1.1750E-07	1.2150E-07	1.2550E-07
F1	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
F2	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05
V1	-1.3474E-01	-1.3474E-01	-1.3474E-01	-1.3474E-01	-1.3474E-01	-1.3474E-01	-1.3474E-01
I1	-4.4392E-01	-4.4392E-01	-4.4392E-01	-4.4392E-01	-4.4392E-01	-4.4392E-01	-4.4392E-01
TIME	1.2550E-07	1.3500E-07	1.3750E-07	1.4150E-07	1.4550E-07	1.4950E-07	1.5350E-07
F1	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
F2	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05
V1	-1.4400E-01	-1.4400E-01	-1.4400E-01	-1.4400E-01	-1.4400E-01	-1.4400E-01	-1.4400E-01
I1	-7.2424E-01	-7.2424E-01	-7.2424E-01	-7.2424E-01	-7.2424E-01	-7.2424E-01	-7.2424E-01
TIME	1.5750E-07	1.6150E-07	1.6550E-07	1.6950E-07	1.7350E-07	1.7750E-07	1.8150E-07
F1	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
F2	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05	2.1999E-05
V1	-1.3474E-01	-1.3474E-01	-1.3474E-01	-1.3474E-01	-1.3474E-01	-1.3474E-01	-1.3474E-01
I1	-4.4392E-01	-4.4392E-01	-4.4392E-01	-4.4392E-01	-4.4392E-01	-4.4392E-01	-4.4392E-01

Figure 82b. SCEPTRE Printout during SCEPTRE Run (cont) .

[illegible]

Figure 82c. SCEPTRE Printout during SCEPTRE Run (cont).

TRANSIENT ANALYSIS RESULTS

TIME	4.03500+07	4.13500+07	4.17500+07	4.21500+07	4.25500+07	4.29500+07	4.33500+07
U1	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05
U2	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07
U3	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01
U4	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01
TIME	4.37500+07	4.41500+07	4.45500+07	4.49500+07	4.53500+07	4.57500+07	4.61500+07
U1	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05
U2	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07
U3	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01
U4	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01
TIME	4.65500+07	4.69500+07	4.73500+07	4.77500+07	4.81500+07	4.85500+07	4.89500+07
U1	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05
U2	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07
U3	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01
U4	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01
TIME	4.93500+07	4.97500+07	5.01500+07	5.05500+07	5.09500+07	5.13500+07	5.17500+07
U1	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05	1.00000-05
U2	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07	2.19374-07
U3	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01	-2.25716-01
U4	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01	-1.12458-01

Figure 82d. SCEPTRE Printout during SCEPTRE Run (cont).

zero throughout. Normally in a SAP execution the iteration cycle in MAIN is printed out during each time step, but the basic iteration strategy of SAP is out of action because SCEPTRE has the last output node. Hence iteration results are not printed in the listing of Figure 81 by SAP.

The printer plot output by SCEPTRE from the run is contained in Figure 80.

SECTION VIII

CONCLUSIONS

Emphasis has been given during this research program to finding out whether reasonable hope exists for treating radiation effects on electronic components using strictly numerical representations of time domain input-output response surfaces. It was seen that convolution works reasonably well for the class of non-linearity represented by a saturating op amp.

Probably a clear cut major advantage of the approach is the immediate grasp of the intricacies of the radiation response over the domains of the dose and time variables which the surface presentations provide. If suitable data acquisition systems and data reduction equipment are made on-line accessories at the major radiation facilities in the country, then surfaces such as presented here could be obtained while the data is being taken. If a picture is worth a thousand words, certainly a surface is worth several dozens of Polaroid pictures of single events. The surface representation is far more readily remembered.

The original surface representations for the 741 were rather laboriously drawn by hand using large three dimensional perspective grids. The one advantage of the technique was that human judgement was directly relied in resolving inconsistencies between data sets, smoothing and interpolating where needed. The first results thus looked good subjectively. The data sets for the 9704 gate were treated solely by computer processing methods. By judicious use of wild-pointing, smoothing, and interpolation routines, and by culling completely inconsistent data sets, surfaces could be obtained whose smoothness is again a matter of subjective choice. If appropriate data reduction equipment is available at the radiation site, the computer processing strategies are clearly the way to go.

The basic computational techniques involved in the Propagation of Bivariable Response Functions were tried for some relatively simple circuit configurations. A learning process was evident in treating the parallel block algorithm. If the transfer functions contain the output variables, then a self-consistency loop is required in the iteration process to produce a stable convergence. The precision of the solution was about 0.1 percent for the case of a feedback loop around an op amp. The gain calculated was precisely that expected. The response of the op amp to sine waves yielded expected values for the phase shift with frequency.

Care must be taken in very slow phenomena that the time steps are small enough that enough partitions are created to correctly sample the device dynamics. If the device is nonlinear, more time steps may be required in the nonlinear region than in the linear region.

The radiation response simulations have good computational characteristics so far as large scale computer processing is concerned. Runs were made that involved convolving or interpolating on four surfaces simultaneously, with execution times on the UNIVAC 1108 of six seconds per 50 time steps. All the surfaces treated here were fit with a 20 x 20 fitting grid, and thus required 1200 numbers to represent them. Some surfaces might be adequately fit with a smaller number. It was also pointed out that if the surface values are obtained by calculation based on a lower level system analysis rather than from experimental data, it is not necessary to least square justify the information. The computer time required to generate the three dimensional fit is then lowered drastically. Thus from the point of view of the important system analysis concept of staging of complexity levels, the machine execution time requirements are entirely reasonable.

Logarithmic time bases were seen to be advantageous in fitting certain classes of surfaces. If the device makes a rapid transition from a fast to a slow phenomena, then such a time base works well, both for the surface and the simulation. Problems can be encountered in using such a time base on a surface which is short relative to the time steps, as the time step changes during the computation. In this case a convolutional strategy will probably have to make a transition to a single partition look-up process.

The merging of the techniques of SAP with SCEPTRE seemed relatively straightforward. The amount of core space remaining available to the user hopefully might be increased when the matrix dimensions in SCEPTRE become adjustable. If a block technique is used, the nodal complexity should be reduced.

While it is conceivable that higher dimensionality surfaces than the three dimensional ones treated here might be required for general cases, no attention was directed to them in the work covered here. This is a question to be examined if the three dimensional surface concept does prove of value. Work is being done at LMSC in improving the computer aspects of dealing with surfaces of higher dimensionality.

The results obtained in this contract seem encouraging in terms of computational characteristics, and enlightening in terms of understanding and visualizing the effects of radiation on devices. Summarizing then:

Established Advantages or Conclusions

- Surfaces provide rich intuitive understanding of device electrical and radiation responses.
- On-line computer reduction of radiation effects data to the surface form seems practical.
- Engineering precision of output response prediction can be achieved by the bivariable response function propagation method.
- The computer execution time and storage required are entirely acceptable.
- A completely numerical representation of device dynamics can be used for circuit analysis.
- Staging of complexity levels in system analysis by computer calculation of response surfaces should have acceptable execution times for surface fitting.
- Logarithmic time bases and steps can be used for certain devices radiation simulation.
- SCEPTRE is adaptable to this method.

Areas of Suggested Further Study

- Best mix of iteration strategies in combination series and parallel circuits.
- Actual trial of staging of complexity levels in system analysis by this method.
- Transition between convolution and single partition surface look-up to calculate responses.
- Use of surfaces having dimensionality higher than three.
- Application to wider classes of devices and systems.
- Failure mechanism analysis of the permanent degradation observed on the 741s.

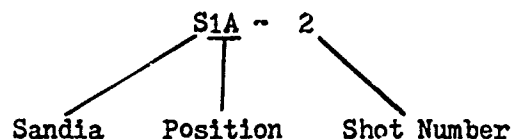
APPENDIX

TECHNIQUES USED FOR REACTOR TESTS OF DEVICES

The sample devices were held to styrofoam assemblies held at seven distances from the core centerline, as given in the Test Plan we had submitted earlier to White Sands. The overall situation in the reactor room is shown in Figure 83. The photograph shows five of the styrofoam blocks mounted to the pegboard table. Two other blocks are out of the field of view off to the left at distances of 7 and 12 feet from core centerline. Careful inspection of the drawing in the frame on the far wall of the cave shows a plan drawing of the reactor core itself. Note the "glory hole" down the core axis for maximum neutron dosages.

Since the different holders portray different angles relative to the lens axis, the way in which the devices are attached and positioned relative to the reactor core become evident. Affixed in the leftmost holder are two cables (laced with blue tape and orange tape) holding mother sockets containing one 9704 and one 741 respectively. (The 9704 is a flat-pak and doesn't stick out as far as the DIP 741 package.) The sockets are mounted on 3 x 4 in.-pieces of flat white plastic 1/16th inch thick. A black plastic ferrule, which can't be seen, fits around the cable on the back side of the plastic, and is potted with Silastic RTV to hold the wires to the socket better. The ferrule nests in a 1 in.-cut into the front face of the styrofoam block, as shown in Figure 84. Styrofoam is an easy material to work with, and has low density which is important from the point of view of minimizing interaction with the core fission characteristics.

The dosimetry employed during the shots consisted of sulfur in aluminum cans, which measures the integrated fast neutron flux, and TLD's or thermoluminescent detectors of LiF, which is responsive to the gamma flux. These were coded as follows:



The dosimeters were held in position on the styrofoam blocks by masking tape, so that rapid changes could be made between reactor bursts reducing exposure of personnel to the radiation from the walls of the reactor room.

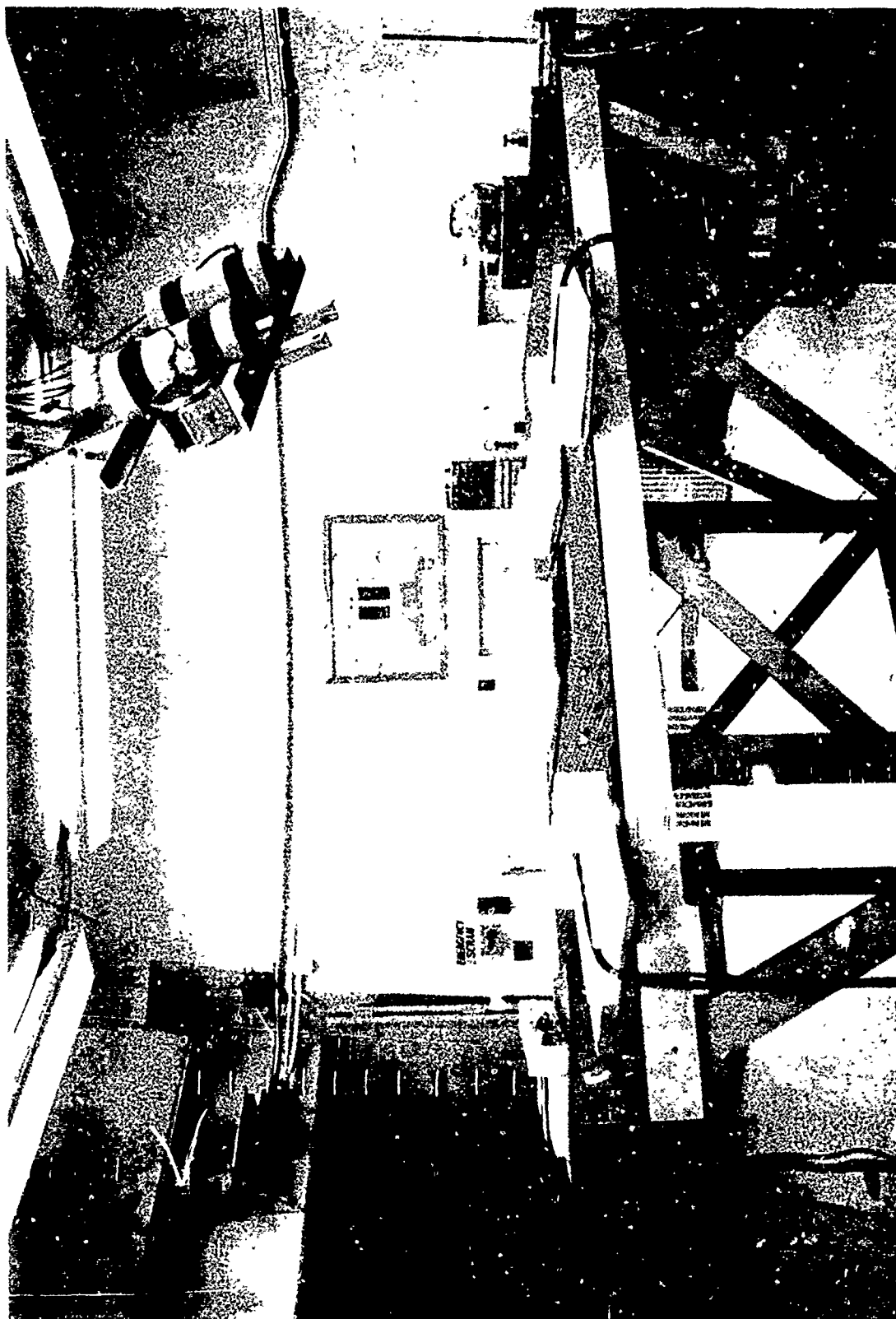
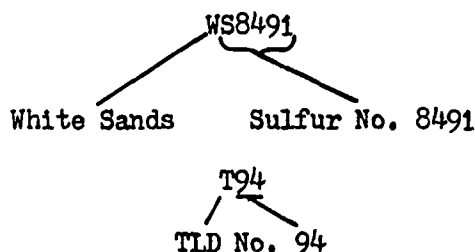


Figure 83. Sandia Pulse Reactor Room showing 4' x 8' Pegboard Table Attached to Aluminum Support Table, and with Styrofoam Blocks held to it and Holding Samples in Sockets. Note Dress of Cables.



Figure 84. Typical Styrofoam Block to Hold Devices. Radius of Inside Surface provides Equidistant Exposures for up to Three Devices. The Ferrule on the Back of a Socket Plate fits into a One Inch Hole in the Block. Cable Slips in Groove.

For the White Sands dosimeters mounted for cross-calibration puposes



These dosimeters were affixed to the face of the styrofoam blocks with masking tape. The devices project 1" closer to the core than the dosimeters from the face of the block, and the final dosage computations were corrected for this difference assuming inverse square intensity change with distance. No attempt was made to alter the fission $n - \gamma$ ratio by lead shielding. If lead is placed close to the reactor core, significant perturbations in the reactor pulse result.

Safety block and control rod adjustment controls are the means by which the desired burst pulse is achieved. This is most conveniently discussed in terms of the temperature rise of the core as measured by thermocouples, and is called ΔT and given in degrees centigrade. We aimed at a ΔT value of 320°C for all the bursts. The actual values achieved were:

Shot No.	Sandia Burst No.	ΔT	Date	Time	δt
				h m	h m
1	489	317°C	5/14/71	1935	
2	490	296	5/15/71	0932	
3	491	317	"	1227	3 55
4	492	322	"	1452	2 25
5	493	321	"	1653	2 01
6	494	315	"	1852	1 59
7	499	313	5/17/71	1817	
8	500	321	"	2031	2 14
9	506	317	5/18/71	1834	
10	507	318	"	2034	2 00
11	514	313	5/19/71	1831	
12	515	319	"	2030	1 59

The last column, δt , gives the time in hours and minutes between bursts. A separate test box was made up for the 741 transfer function measurements, so that a Tektronix 575 curve tracer could be used just for the 9704 gate measurements. A Tektronix 7004 scope was used for the 741 transfer function pictures. Twelve scopes were used in the measurements. A scope was used to be sure only one trigger was delivered to all the pulse generators and scope triggers, since

occasionally the Sandia trigger circuit would create a double pulse. A typical timing sequence for the pulse reactor is shown below in Figure 85.

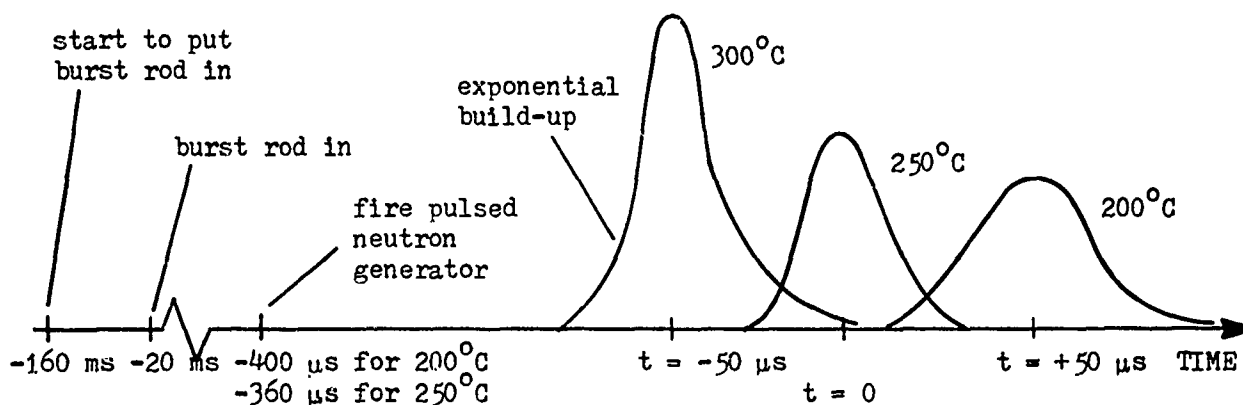
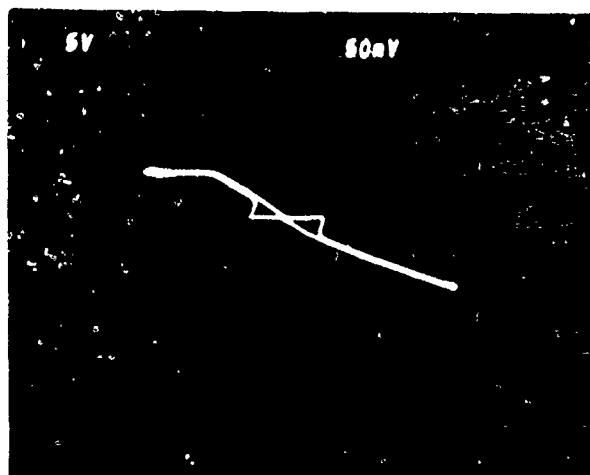


Figure 85. Pulse Reactor Timing Sequence

A type of radiation damage observed on the 741 op amps after neutron exposure at the closest position of the SPR is shown below in Figure 86. The input-output voltage transfer characteristic takes on a butterfly shape in the center. This is a permanent degradation of the device.

DATE 5/19/71 741-11

PHOTO
SOPELO
POST 12
#

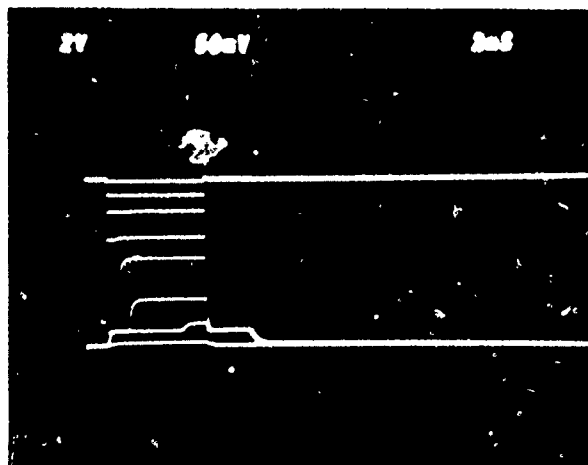


$$\begin{aligned} V_1 &= e_o = 5V / \text{DIV} & H &= e_i = 50mV / \text{DIV} \\ V_2 &= & R_f / R_i &= 68K / 1K & P_{oo} &= 1A \end{aligned}$$

Figure 86. Effect of Neutron Radiation on 741 Op Amp Input-Output Characteristic

If this degradation is observed in a time domain oscilloscope waveform as shown in Figure 87, the output appears to take on a "top hat" which builds and grows on top a low level pulse. With increased input signal the top hat becomes larger and initiates earlier.

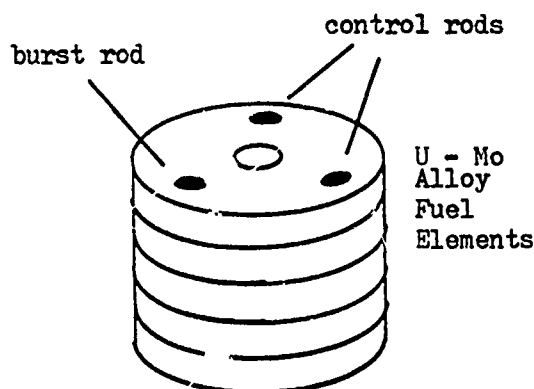
PHOTO
SCOPE 10
POST 12
#



$$\begin{aligned} V_1 = e_i &= 50 \text{ mV/DIV} & H = t &= 2 \text{ ms/DIV} \\ V_2 = e_o &= 2 \text{ V/DIV} & R_f/R_i &= 60 \text{ K/1K} \end{aligned}$$

Figure 87. Effect of Neutron Radiation on 741 Op Amp Pulse Response.

A sketch of the reactor core is shown below.



Schematic of Reactor Core

The Uranian-Molybdenum alloy fuel elements of the core expand 20 to 30 mil during burst, which is enough to stop the fission chain reaction process and terminate the pulse.