Contractor Report

Propulsion Stability Codes for Liquid Propellant Propulsion Systems Developed for Use on a PC Computer

(5-32441)

G. B. Doane III
W. C. Armstrong

Contract No./Delivery Order No. NAS8-36955/95

June 1991

NASA
National Aeronautical and
Space Agency

Report Document Page

1. Report No.

2. Government Accession No.

3. Recipient's Catalog No.

4. Titile and Subtitle

Liquid Propellant Propulsion System Optimization

5. Report Due

July 11,1991

6. Performing Organization Code

Research Institute, UAH

7. Author(s)

George B. Doane III, Wilbur Armstrong

8. Performing Organization Report No.

D.O. 95

10. Work Unit No.

5-32441

9. Performing Organization Name and Address

UAH Research Institute
RI E-47
Huntsville, AL 35899

11. Contract or Grant No.

NAS8-36955, D.O. 95

12. Sponsoring Agency Name and Address

National Aeronautics and Space Administration
Washington, D.C. 20546-001
Marshall Space Flight Center, AL 35812

13. Type of report and Period covered

Final Technical Report July 1991

14. Sponsoring Agency Code

15. Supplementary Notes

16.Abstract

Research into component modeling and system synthesis leading to the analysis of the major types of propulsion system instabilities and the characterization of various component characteristics.

17. Key Words (Suggested by Authors(s))

Rocket propulsion stability, ducts, propellant pumps, valves, actuators

18. Distribution Statement

TBA

19. Security Class. (of this report)

Unclassified

20. Security Class. (of this page)

Unclassified

21. No. of pages

24 + Appendices

22. Price

Table of Contents

Summary

Last year, several programs designed to run on a PC computer were developed for MSFC. These codes covered the low, intermediate, and high frequency modes of oscillation of a liquid rocket propulsion system. No graphics were built into these programs and only simple piping layouts were supported. This year's effort has been to add run time graphics to the low and intermediate frequency codes, allow new types of piping elements (accumulators, pumps, and split pipes) in the low frequency code, and develop a new code for the PC to generate Nyquist plots.

Introduction

This year began with the computer programs at the stage described in NASA Contractor Report 5-32176, June 1990. The programs written for the Macintosh had plot capability, but were slow because of the interpretive language used. Programs for the PC were written in FORTRAN to increase the speed of execution. The PC programs discussed in Report 5-32176 contained no graphics.

This year, the PC programs were expanded to include graphics and to address more types of feedline elements. The effort this year was primarily in the low frequency area. In addition to the admittance calculations, the pressure transfer function was evaluated. A new PC program was written to generate the Nyquist plots already implemented on the Macintosh. Graphics were added to the intermediate mode program. Frequency may be input (and output) in either radians per second or in Hertz.

This report will trace the development of these enhancements. A summary of the working equations for impedance are presented first. Then, the equations are derived for each of the types of piping elements handled: straight piped, inline accumulator, tuned stub, Helmholtz resonator, parallel resonator, pumps, and split pipes . The bend is handled as an equivalent straight pipe based on the procedure presented in NASA Contractor Report 5-32176. All impedances are nondimensionalized by chamber pressure divided by chamber mass flow $(p_c/\dot{m}_c)$. In the split pipe case, this factor for one engine is multiplied by the number of engines $[m\cdot(p_c/\dot{m}_c)]$.

The Nyquist program is discussed next. The equations used are presented. In addition the Nyquist plots, phase-gain plots have been added.

The primary modifications to the intermediate mode program concern simplifying the operation and the plotting of the n vs τ curves.

There were no modifications to the high frequency program made this year. However, the code was used to study the stability of a couple of engines (see Appendix A).

Feedline Program

The feedline program has undergone extensive enhancements. The addition of graphics allows the user to run a case, look at the results, interactively modify the input, and repeat the cycle. All this may be done with one running of the code. Also, the input was rearranged into a more useful form for this type interactive operation.

The addition of graphics made it feasible to add the pressure transfer function to the code. This required restructuring the logic of the program. The original program was only required to compute the admittance looking toward the tank. The calculation of the pressure transfer function required the computation of impedance looking toward the engine.

Major changes to the code were required to accommodate more complex pipe layouts. The most complex addition was allowing a line to split into m identical lines. This calculation requires an iteration to determine the impedances. The addition of four types of accumulators was more straight forward. Inline accumulators, tuned stubs, Helmholtz resonators, and parallel resonators may be handled by the program. A pump also may be included in the piping layout.

The first graphics incorporated into the program displays the piping layout in the upper half of the screen and the admittance vs frequency curve in the lower half of the screen. A split pipe is represented by only one of the m identical lines. Accumulators are all shown as on the upper part of the pipe. The drawing of the pump has not been added to the graph.

A surface plot and a contour plot were added to display the pressure transfer function vs frequency and distance. The surface plot may be displayed from any viewpoint and as a solid surface or a wire-frame drawing. The contour plot displays nine contour lines with the values of lines 1, 5, and 9 displayed.

All aspects of the plots are under the control of the user. Defaults are set by the program, but these are easily changed. The colors used may be changed and these remain in effect until changed again. Colors are assigned separately to the three graphs. The surface plot and contour plot may be bypassed. The pipe layout - admittance graph is always displayed, but the admittance curve may be plotted as the calculations are made or after they are finished.

These enhancements to the feedline program will be illustrated by a series of runs. The four type of accumulators will be compared to the same layout without an accumulator. The results for the basic configuration are shown in Figure 1. The pipe layout and admittance vs frequency curve are in Figure 1a, the surface plot of the pressure transfer function vs frequency and location is in Figure 1b, and a contour plot of the pressure transfer function is in figure 1c. The peak pressure appears to occur after the second bend from the tank. The accumulators will be inserted at this point.

It should be noted that a coarse grid may underestimate the peak. In all cases run, the finest grid available was run to obtain the peak, then a coarser grid with the same peak was run to produce the plots. For example, the surface shown in Figure 1b was generated using 33 frequencies between 1 and 30 Hertz. The code was run again using 34 frequencies over the same interval giving the surface in Figure 2. The user must be aware of this problem and act accordingly.

An Inline Accumulator was inserted and the code rerun. The accumulator was 2 ft. long with a diameter of 4 ft. No attempt was made to minimize the peak, only to reduce it significantly. The results are given in Figure 3 which shows a drop in the peak pressure of 80%.

Next, a Tuned Stub was used. It was 10.5 ft. long and had a 0.74 ft. diameter. Figure 4 presents the results. The reduction in peak pressure was 70% for this configuration.

A Helmholtz Resonator with a 0.001 ft. diameter stem 0.4 ft. long leading to a volume of 5 ft^3 was run (Fig. 5). This reduced the peak pressure by 72%.

The last accumulator was a Parallel Resonator 1 ft. long with a diameter of 0.05 ft. It bypassed a volume of 1 ft$_3$. Figure 6 shows the results of the run. This configuration reduced the peak pressure by only 47%. Remember, this configuration was not fine tuned as only a reduction in the peak was desired.

The effect of splitting a pipe into three identical lines going to identical engines was investigated by first running a case where the pipe is unsplit, but has an area equivalent to the three pipes. The results of the unsplit pipe are shown in Figure 7. Then the split pipe case was run giving the results shown in Figure 8. These figures show that a split pipe cannot be properly analyzed using an equivalent single pipe.

Summary of Equations for Impedance

In the following equations, $n = s/a$.

1. Straight Pipe

$$Z_t(I) = Z_o(I)\cdot\left[\frac{Z_t(I-1) + Z_o(I)\cdot\tanh(n\cdot l)}{Z_o(I) + Z_t(I-1)\cdot\tanh(n\cdot l)}\right]$$

$$Z_g(I) = \{e^{n\cdot l_1}\cdot[Z_o(I+1) + Z_g(I+1)]\cdot(1 - N\cdot M\cdot e^{-2\cdot n\cdot l})$$

$$- Z_o(I+1)\cdot(1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1})\}/(1 + N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1})$$

where $N = [Z_o(I+1) - Z_t(I-1)]/[Z_o(I+1) + Z_t(I-1)]$

$M = [Z_o(I+1) - Z_g(I+1)]/[Z_o(I+1) + Z_g(I+1)]$

$l = L(I) + L(I+1)$

$l_1 = L(I+1)$

2. Inline Accumulator

$Z_{\bullet}=1/(C\cdot s)$

$Z_t(I) = Z_{\bullet}\cdot Z_t(I-1)/[Z_t(I-1) + Z_{\bullet}]$

$Z_g(I) = Z_{\bullet}\cdot Z_g(I+1)/[Z_g(I+1) + Z_{\bullet}]$

3. Tuned Stub

$Z_{\bullet} = Z_o/\tanh(n\cdot l)$

$Z_t(I) = Z_{\bullet}\cdot Z_t(I-1)/[Z_t(I-1) + Z_{\bullet}]$

$Z_g(I) = Z_{\bullet}\cdot Z_g(I+1)/[Z_g(I+1) + Z_{\bullet}]$

4. Helmholtz Resonator

$Z_{\bullet} = (1 + L\cdot C\cdot s^2)/(C\cdot s)$

$Z_t(I) = Z_{\bullet}\cdot Z_t(I-1)/[Z_t(I-1)+Z_{\bullet}]$

$Z_g(I) = Z_{\bullet}\cdot Z_g(I+1)/[Z_g(I+1)+Z_{\bullet}]$

5. Parallel Resonator

$$Z_e = L\cdot s/(1 + L\cdot C\cdot s^2)$$

$$Z_t(I) = Z_t(I-1) + Z_e$$

$$Z_g(I) = Z_g(I+1) + Z_e$$

6. Pump

$$Z_p = \frac{\partial p}{\partial \dot{m}}$$

$$Z_t(I) = \{Z_t(I-1) + (Z_p + L\cdot s)\cdot[1 + Z_t(I-1)\cdot C\cdot s]\}/[1 + Z_t(I-1)\cdot C\cdot s]$$

$$Z_g(I) = [L\cdot s - Z_p + Z_g(I+1)]/\{1 + C\cdot s\cdot[L\cdot s - Z_p + Z_g(I+1)]\}$$

7. Split Pipe

$$Z_e = Z_g(I-1)\cdot Z_t(I-1)/[(m-1)\cdot Z_t(I-1) + Z_g(I-1)]$$

$$Z_t(I) = Zo(I)\cdot\left[\frac{Z_e + Zo(I)\cdot\tanh(n\cdot l)}{Zo(I) + Z_e\cdot\tanh(n\cdot l)}\right]$$

$$Z_g(I) = \{e^{n\cdot l}{}_1\cdot[Zo(I+1) + Z_g(I+1)]\cdot(1 - N\cdot M\cdot e^{-2\cdot n\cdot l}) - Zo(I+1)$$

$$\cdot(1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l}{}_1)\}/[m\cdot(1 + N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l}{}_1)]$$

where $N = [Zo(I+1) - Z_t(I-1)]/[Zo(I+1) + Z_t(I-1)]$

$M = [Zo(I+1) - Z_g(I+1)]/[Zo(I+1) + Z_g(I+1)]$

$l = L(I) + L(I+1)$

$l_1 = L(I+1)$

Straight Pipe

The equation for the pressure at any point in a pipe is derived on page 25 of NASA Contractor Report 5-32176.

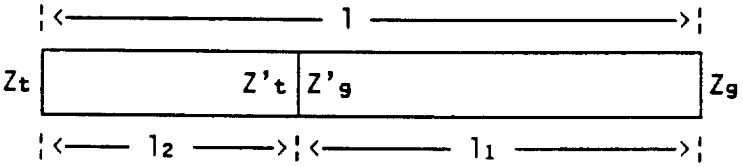
$$\frac{p(x,s)}{p_g(s)} = \left(\frac{Z_o}{Z_o + Z_g}\right)\cdot\left[\frac{e^{-n\cdot x} - N\cdot e^{-n\cdot(2\cdot l-x)}}{1 - N\cdot M\cdot e^{-2\cdot n\cdot l}}\right]$$

where $n = s/a$

where $$N = \frac{Z_o - Z_t}{Z_o + Z_t}$$

$$M = \frac{Z_o - Z_g}{Z_o + Z_g}$$

Consider the case where the pipe is divided into two sections:



Case 1. Solve for Z_t. Z_g is the same for l and l_1

$$\left(\frac{Z_o}{Z_o + Z_g}\right)\cdot\left[\frac{e^{-n\cdot x} - N\cdot e^{-n\cdot(2\cdot l-x)}}{1 - N\cdot M\cdot e^{-2\cdot n\cdot l}}\right]$$

$$= \left(\frac{Z_o}{Z_o + Z_g}\right)\cdot\left[\frac{e^{-n\cdot x} - N'\cdot e^{-n\cdot(2\cdot l_1-x)}}{1 - N'\cdot M\cdot e^{-2\cdot n\cdot l_1}}\right]$$

evaluate at $x = l_1$

$$\left(\frac{e^{-n\cdot l_1} - N\cdot e^{-n\cdot(2\cdot l-l_1)}}{1 - N\cdot M\cdot e^{-2\cdot n\cdot l}}\right) = \left[\frac{e^{-n\cdot l_1} - N'\cdot e^{-n\cdot(2\cdot l_1-l_1)}}{1 - N'\cdot M\cdot e^{-2\cdot n\cdot l_1}}\right]$$

$$\left(\frac{e^{-n\cdot l_1} - N\cdot e^{-2\cdot n\cdot l}\cdot e^{n\cdot l_1}}{1 - N\cdot M\cdot e^{-2\cdot n\cdot l}}\right) = \left(\frac{e^{-n\cdot l_1} - N'\cdot e^{-n\cdot l_1}}{1 - N'\cdot M\cdot e^{-2\cdot n\cdot l_1}}\right)$$

$$\left(\frac{1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}}{1 - N\cdot M\cdot e^{-2\cdot n\cdot l}}\right) = \left(\frac{1 - N'}{1 - N'\cdot M\cdot e^{-2\cdot n\cdot l_1}}\right)$$

$$(1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1})\cdot(1 - N'\cdot M\cdot e^{-2\cdot n\cdot l_1})$$

$$= (1 - N')\cdot(1 - N\cdot M\cdot e^{-2\cdot n\cdot l})$$

$$1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1} - N'\cdot M\cdot e^{-2\cdot n\cdot l_1} + N\cdot N'\cdot M\cdot e^{-2\cdot n\cdot l}$$

$$= 1 - N' - N\cdot M\cdot e^{-2\cdot n\cdot l} + N\cdot N'\cdot M\cdot e^{-2\cdot n\cdot l}$$

$$(1 - M\cdot e^{-2\cdot n\cdot l_1})\cdot N' = (e^{2\cdot n\cdot l_1} - M)\cdot N\cdot e^{-2\cdot n\cdot l}$$

$$N' = N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}$$

but, $l_2 = l - l_1$, therefore

$$N' = N\cdot e^{-2\cdot n\cdot l_2} = N\cdot[\cosh(2\cdot n\cdot l_2) - \sinh(2\cdot n\cdot l_2)]$$

$$N' = N\cdot[\cosh^2(n\cdot l_2) + \sinh^2(n\cdot l_2) - 2\cdot\cosh(n\cdot l_2)\cdot\sinh(n\cdot l_2)]^2$$

$$N' = N\cdot[\cosh(n\cdot l_2) - \sinh(n\cdot l_2)]^2$$

$$N' = N\cdot\left\{\frac{1}{\mathrm{sqrt}[1 - \tanh^2(n\cdot l_2)]} - \frac{\tanh(n\cdot l_2)}{\mathrm{sqrt}[1 - \tanh^2(n\cdot l_2)]}\right\}^2$$

$$N' = N\cdot\left\{\frac{[1 - \tanh(n\cdot l_2)]^2}{1 - \tanh^2(n\cdot l_2)}\right\} = N\cdot\left[\frac{1 - \tanh(n\cdot l_2)}{1 + \tanh(n\cdot l_2)}\right]$$

let $l_2 = l$ and expand N and N'

$$\left(\frac{Z_o - Z'_t}{Z_o + Z'_t}\right) = \left(\frac{Z_o - Z_t}{Z_o + Z_t}\right)\cdot\left[\frac{1 - \tanh(n\cdot l)}{1 + \tanh(n\cdot l)}\right]$$

$$(Z_o - Z'_t)\cdot(Z_o + Z_t)\cdot[1 + \tanh(n\cdot l)]$$

$$= (Z_o - Z_t)\cdot(Z_o + Z'_t)\cdot[1 - \tanh(n\cdot l)]$$

$$(Z_o^2 + Z_o \cdot Z_t - Z_o \cdot Z'_t - Z_t \cdot Z'_t) \cdot [1 + \tanh(n \cdot l)]$$

$$= (Z_o^2 - Z_o \cdot Z_t + Z_o \cdot Z'_t - Z_t \cdot Z'_t) \cdot [1 - \tanh(n \cdot l)]$$

$$Z_o \cdot (Z_t - Z'_t) + (Z_o^2 - Z_t \cdot Z'_t) \cdot \tanh(n \cdot l)$$

$$= -Z_o \cdot (Z_t - Z'_t) - (Z_o^2 - Z_t \cdot Z'_t) \cdot \tanh(n \cdot l)$$

$$Z_o \cdot (Z_t - Z'_t) + (Z_o^2 - Z_t \cdot Z'_t) \cdot \tanh(n \cdot l) = 0$$

$$[Z_o + Z_t \cdot \tanh(n \cdot l)] \cdot Z'_t = Z_o \cdot [Z_t + Z_o \cdot \tanh(n \cdot l)]$$

$$Z'_t = Z_o \cdot \left[\frac{Z_t + Z_o \cdot \tanh(n \cdot l)}{Z_o + Z_t \cdot \tanh(n \cdot l)}\right]$$

or,

$$Z_t(I) = Z_o(I) \cdot \left[\frac{Z_t(I-1) + Z_o(I) \cdot \tanh(n \cdot l)}{Z_o(I) + Z_t(I-1) \cdot \tanh(n \cdot l)}\right]$$

Case 2. Solve for Z_g. Z_t is the same for l and l_2

$$\left(\frac{Z_o}{Z_o + Z_g}\right) \cdot \left[\frac{e^{-n \cdot x} - N \cdot e^{-n \cdot (2 \cdot l - x)}}{1 - N \cdot M \cdot e^{-2 \cdot n \cdot l}}\right]$$

$$= \left(\frac{Z_o}{Z_o + Z'_g}\right) \cdot \left[\frac{e^{-n \cdot x} - N \cdot e^{-n \cdot (2 \cdot l_2 - x)}}{1 - N \cdot M' \cdot e^{-2 \cdot n \cdot l_2}}\right]$$

evaluate at $x = l_1$ for l and $x = 0$ for l_2

$$\left(\frac{1}{Z_o + Z_g}\right) \cdot \left[\frac{e^{-n \cdot l_1} - N \cdot e^{-n \cdot (2 \cdot l - l_1)}}{1 - N \cdot M \cdot e^{-2 \cdot n \cdot l}}\right]$$

$$= \left(\frac{1}{Z_o + Z'_g}\right) \cdot \left(\frac{1 - N \cdot e^{-2 \cdot n \cdot l_2}}{1 - N \cdot M' \cdot e^{-2 \cdot n \cdot l_2}}\right)$$

substitute $l - l_1$ for l_2

$$\left(\frac{1}{Zo + Z_g}\right)\cdot\left(\frac{e^{-n\cdot l_1} - N\cdot e^{-2\cdot n\cdot l}\cdot e^{n\cdot l_1}}{1 - N\cdot M\cdot e^{-2\cdot n\cdot l}}\right)$$

$$= \left(\frac{1}{Zo + Z'_g}\right)\cdot\left(\frac{1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}}{1 - N\cdot M'\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}}\right)$$

$$\left(\frac{e^{-n\cdot l_1}}{Zo + Z_g}\right)\cdot\left(\frac{1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}}{1 - N\cdot M\cdot e^{-2\cdot n\cdot l}}\right)$$

$$= \left(\frac{1}{Zo + Z'_g}\right)\cdot\left(\frac{1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}}{1 - N\cdot M'\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}}\right)$$

$$\left(\frac{e^{-n\cdot l_1}}{Zo + Z_g}\right)\cdot\left(\frac{1}{1 - N\cdot M\cdot e^{-2\cdot n\cdot l}}\right)$$

$$= \left(\frac{1}{Zo + Z'_g}\right)\cdot\left(\frac{1}{1 - N\cdot M'\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}}\right)$$

$$(Zo + Z'_g)\cdot e^{-n\cdot l_1}\cdot(1 - N\cdot M'\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1})$$

$$= (Zo + Z_g)\cdot(1 - N\cdot M\cdot e^{-2\cdot n\cdot l})$$

$$(Zo + Z'_g)\cdot e^{-n\cdot l_1}\cdot\left[1 - N\cdot\left(\frac{Zo-Z'_g}{Zo+Z'_g}\right)\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}\right]$$

$$= (Zo + Z_g)\cdot(1 - N\cdot M\cdot e^{-2\cdot n\cdot l})$$

$$(Zo + Z'_g)\cdot e^{-n\cdot l_1} - N\cdot(Zo-Z'_g)\cdot e^{-2\cdot n\cdot l}\cdot e^{n\cdot l_1}$$

$$= (Zo + Z_g)\cdot(1 - N\cdot M\cdot e^{-2\cdot n\cdot l})$$

$$Zo + Z'_g - N\cdot(Zo-Z'_g)\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}$$

$$= e^{n\cdot l_1}\cdot(Zo + Z_g)\cdot(1 - N\cdot M\cdot e^{-2\cdot n\cdot l})$$

$$Zo\cdot(1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}) + Z'_g\cdot(1 + N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1})$$

$$= e^{n\cdot l_1}\cdot(Zo + Z_g)\cdot(1 - N\cdot M\cdot e^{-2\cdot n\cdot l})$$

$$Z'_g\cdot(1 + N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1}) = e^{n\cdot l_1}\cdot(Zo + Z_g)\cdot(1 - N\cdot M\cdot e^{-2\cdot n\cdot l})$$

$$- Zo\cdot(1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1})$$

$$Z'_g = [e^{n\cdot l_1}\cdot(Zo + Z_g)\cdot(1 - N\cdot M\cdot e^{-2\cdot n\cdot l}) - Zo\cdot(1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1})]$$

$$/(1 + N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1})$$

or,

$N = [Zo(I+1) - Z_t(I-1)]/[Zo(I+1) + Z_t(I-1)]$

$M = [Zo(I+1) - Z_g(I+1)]/[Zo(I+1) + Z_g(I+1)]$

$l = L(I) + L(I+1)$

$l_1 = L(I+1)$

$$Z_g(I) = \{e^{n\cdot l_1}\cdot[Zo(I+1) + Z_g(I+1)]\cdot(1 - N\cdot M\cdot e^{-2\cdot n\cdot l})$$

$$- Zo(I+1)\cdot(1 - N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1})\}/(1 + N\cdot e^{-2\cdot n\cdot l}\cdot e^{2\cdot n\cdot l_1})$$

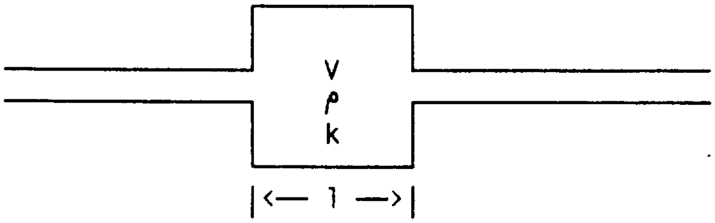
Accumulators

Four types of accumulators will be considered: inline (manifold), tuned stub, Helmholtz, and parallel. For all these accumulators, the equations hold for either direction (Z_t and Z_g). For the tuned stub and Helmholtz resonator, the admittance seen by the next element is the sum of the admittance of the preceding element and the admittance of the accumulator.

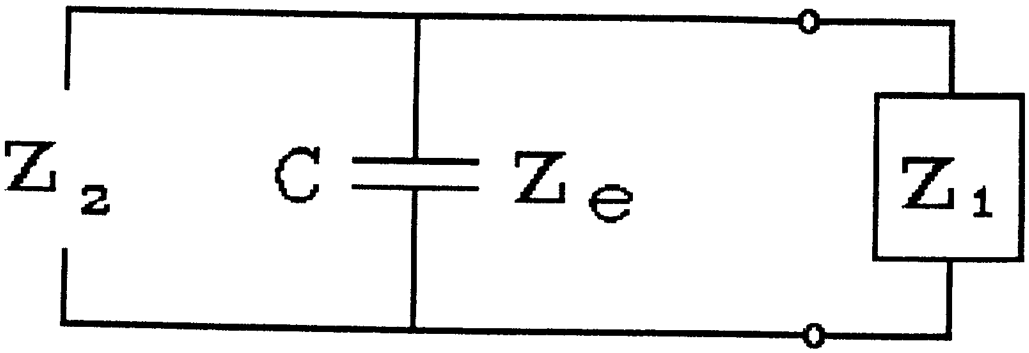
The following equations hold for each of the types of accumulators.

$A = \pi \cdot d^2/4$	ft^2
$a = \sqrt{g_c \cdot k/\rho}$	ft/sec
$C = (V/a^2)\cdot(p_c/\dot{m}_c) = (\rho\cdot V/k)\cdot(p_c/\dot{m}_c)$	sec
$L = [1/(g_c A)]/(p_c/\dot{m}_c)$	sec
$V = l \cdot A$	ft^3
$y = C \cdot s$	nd
$z = L \cdot s$	nd
$Z_o = \sqrt{z/y} = \sqrt{L/C}$	nd
$\sqrt{z \cdot y} = s \cdot \sqrt{L \cdot C}$	nd

1. Inline accumulator



The inline accumulator is analogous to a manifold which is a capacitor circuit.



$Z_e = 1/(C \cdot s)$

$1/Z_2 = 1/Z_e + 1/Z_1$

$1/Z_2 = (Z_1 + Z_e)/(Z_1 \cdot Z_e)$

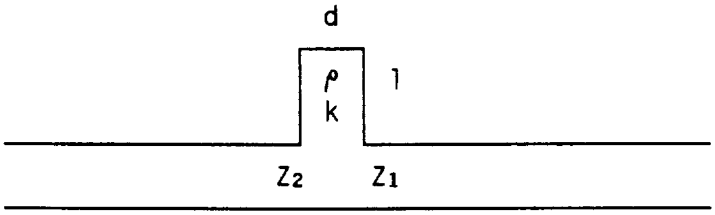
$Z_2 = Z_1 \cdot Z_e/(Z_e + Z_1)$

or,

$Z_t(I) = Z_t(I-1) \cdot Z_e/[Z_e + Z_t(I-1)]$

$Z_g(I) = Z_g(I+1) \cdot Z_e/[Z_e + Z_g(I+1)]$

2. Tuned Stub



The tuned stub considered has no net flow through it. Thus the termination impedance -> ∞ and the impedance of a pipe becomes

$Z_e = Z_o/\tanh(n \cdot l)$

$1/Z_2 = 1/Z_e + 1/Z_1$

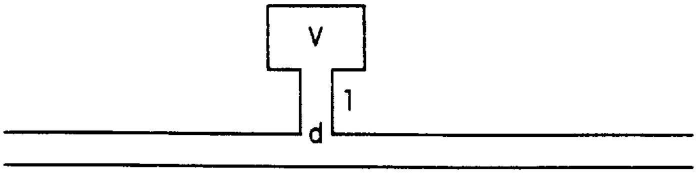
$Z_2 = Z_e \cdot Z_1/(Z_1 + Z_e)$

or,

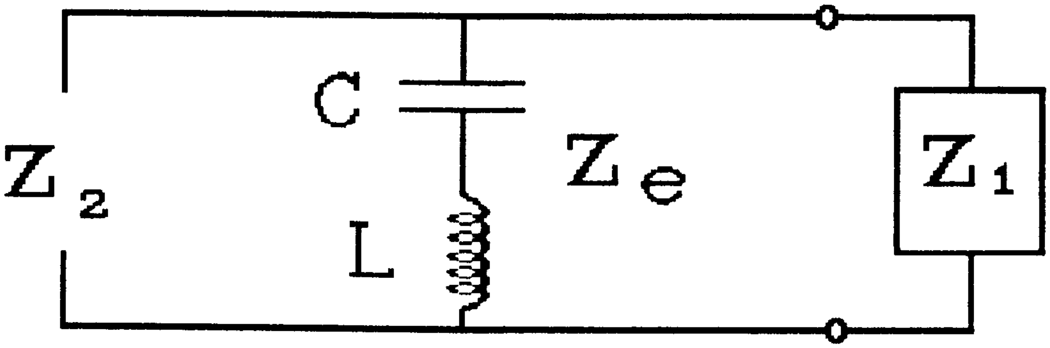
$Z_t(I) = Z_e \cdot Z_t(I-1)/[Z_t(I-1) + Z_e]$

$Z_g(I) = Z_e \cdot Z_g(I+1)/[Z_g(I+1) + Z_e]$

3. Helmholtz Resonator



The Helmholtz resonator is analogous to a series resonant circuit.



where L is based on the dimensions of the small pipe, and C is based on the large cavity, thus

$Z_e = L \cdot s + 1/(C \cdot s)$

$Z_e = (1 + L \cdot C \cdot s^2)/(C \cdot s)$

$1/Z_2 = 1/Z_e + 1/Z_1$

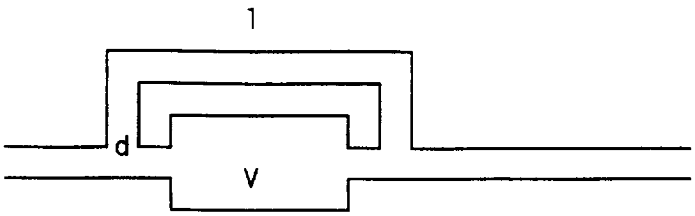
$Z_2 = Z_e \cdot Z_1/(Z_1 + Z_e)$

or,

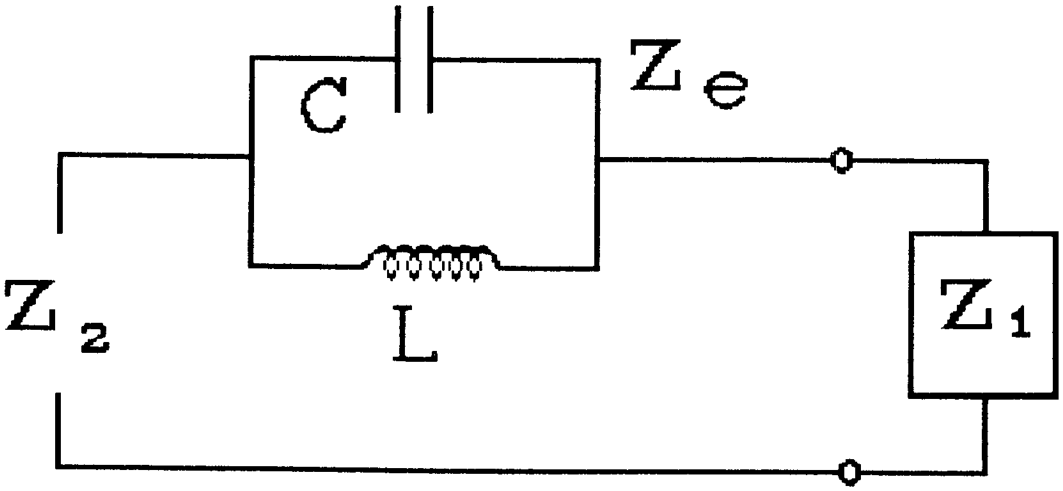
$Z_t(I) = Z_e \cdot Z_t(I-1)/[Z_t(I-1) + Z_e]$

$Z_g(I) = Z_e \cdot Z_g(I+1)/[Z_g(I+1) + Z_e]$

4. Parallel Resonator



The parallel resonator is analogous to a parallel resonant circuit.



where L is based on the dimensions of the bypass line, and C is based on the dimensions of the volume bypassed

$1/Z_e = 1/L \cdot s + C \cdot s$

$Z_e = L \cdot s/(1 + L \cdot C \cdot s^2)$

$Z_2 = Z_1 + Z_e$

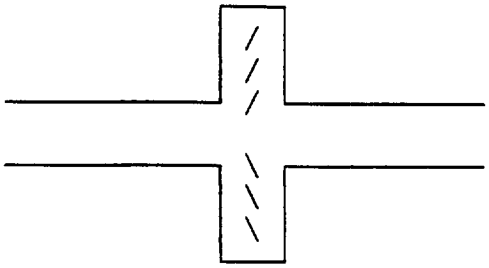
$Z_2 = Z_1 + L \cdot s/(1 + L \cdot C \cdot s^2)$

or,

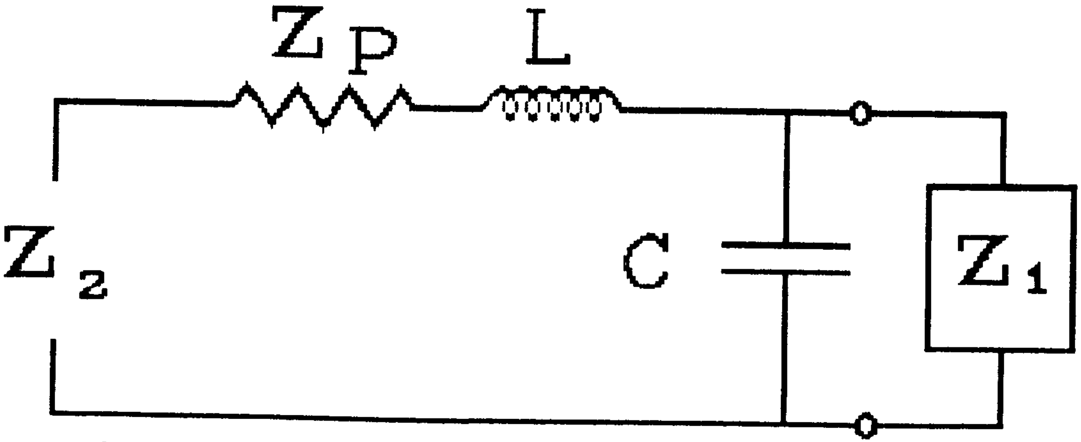
$Z_t(I) = Z_t(I-1) + L \cdot s/(1 + L \cdot C \cdot s^2)$

$Z_g(I) = Z_g(I+1) + L \cdot s/(1 + L \cdot C \cdot s^2)$

Pumps



The pump is analogous to the following circuit.



$$Z_p = \frac{\partial p}{\partial \dot{m}}$$

$$Z_2 = Z_p + L \cdot s + 1/(C \cdot s + 1/Z_1)$$

$$Z_2 = [Z_1 + (Z_p + L \cdot s) \cdot (Z_1 \cdot C \cdot s + 1)]/(1 + Z_1 \cdot C \cdot s)$$

or,

$$Z_t(I) = \frac{Z_t(I-1) + (Z_p + L \cdot s) \cdot [1 + Z_t(I-1) \cdot C \cdot s]}{1 + Z_t(I-1) \cdot C \cdot s}$$

When computing the impedance looking toward the engine use the negative of the slope.

$$1/Z_1 = 1/(Z_2 - Z_p + L\cdot s) + C\cdot s$$

$$Z_1 = \frac{L\cdot s - Z_p + Z_2}{1 + C\cdot s\cdot(L\cdot s - Z_p + Z_2)}$$

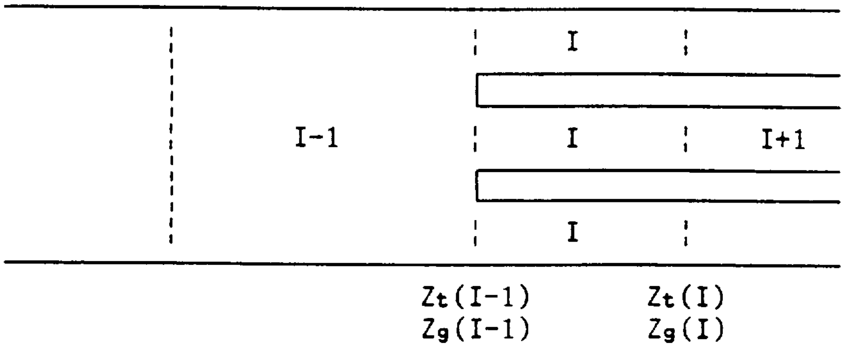
or,

$$Z_g(I) = \frac{L\cdot s - Z_p + Z_g(I+1)}{1 + C\cdot s\cdot[L\cdot s - Z_p + Z_g(I+1)]}$$

Split Piping

Often a main pipe from a fuel or LOX tank splits into several pipes, each going to a different engine. This analysis is for the case where the pipe is split into m identical lines going to m identical engines.

Case I. Finding the impedance looking toward the tank (Z_t).



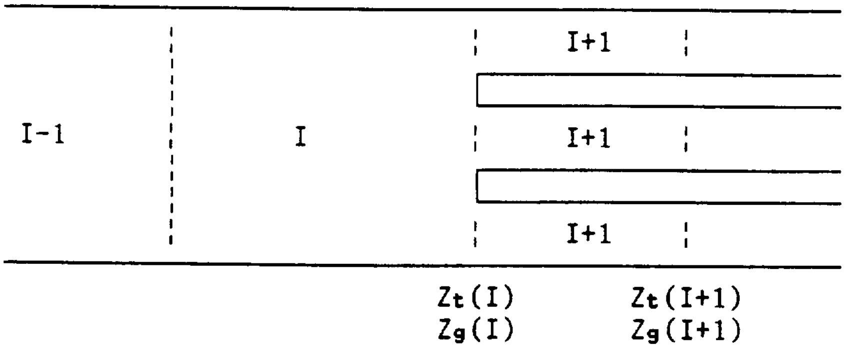
Section I, looking toward the tank sees $Z_t(I-1)$ and (m-1) $Z_g(I-1)$'s in parallel. Therefore the effective Z_e it sees is

$$\frac{1}{Z_e} = \frac{m-1}{Z_g(I-1)} + \frac{1}{Z_t(I-1)}$$

$$Z_e = Z_g(I-1) \cdot Z_t(I-1)/[(m-1) \cdot Z_t(I-1) + Z_g(I-1)]$$

This Z_e is used in the equations for Z_t instead of $Z_t(I-1)$.

Case II. Finding the impedance looking toward the engine (Z_g).



Section I, looking toward the engine sees m sections I+1 in parallel. Therefore the effective $Z_g(I)$ is 1/m of that for one pipe. Thus, compute Z_g using one pipe and then divide by m to obtain $Z_g(I)$.

Nyquist Program

The Nyquist equations presented in NASA Contractor Report 5-32176 were programmed for the PC. The equations used in the Nyquist program are a function of the admittances G_{ox} and G_f. The code was written to plot the Nyquist curves for the four cases: neither admittance used, G_{ox} only, G_f only, and both admittances used.

On page 47 of the report the following equation is derived

$$\frac{e^{-\tau\cdot s}}{(1+\theta_c\cdot s)}\cdot\{[1+\frac{(1+\bar{r})}{C^*}\cdot(\frac{\partial C^*}{\partial r})]\cdot G_{ox}+[1-\frac{\bar{r}\cdot(1+\bar{r})}{C^*}\cdot(\frac{\partial C^*}{\partial r})]\cdot G_f\} = -1.$$

In order to simplify the notation, the following definitions are used:

$$K_1 = \frac{e^{-\tau\cdot s}}{(1+\theta_c\cdot s)}$$

$$A_1 = [1+\frac{(1+\bar{r})}{C^*}\cdot(\frac{\partial C^*}{\partial r})]$$

$$A_2 = [1-\frac{\bar{r}\cdot(1+\bar{r})}{C^*}\cdot(\frac{\partial C^*}{\partial r})]$$

Thus, the equation may be expressed as $K_1\cdot(A_1\cdot G_{ox} + A_2\cdot G_f) = -1$.

The equations used are

$K(j\omega) = 2\cdot K_1$ neither admittance used,

$K(j\omega,G_{ox}) = K_1\cdot A_1$ G_{ox} used,

$K(j\omega,G_f) = K_1\cdot A_2$ G_f used,

$K(j\omega,G_{ox},G_f) = K_1\cdot(A_1+A_2)$ both admittances used.

In addition to the Nyquist plots of these four equations, Phase-Gain plots are also available.

The program will run when there is no data available for either or both of the feedlines. When a line is missing, the user is only allowed to request plots that are available. The admittance calculations include all the variations in the feedline program: split pipes, accumulators, and pumps.

Example plots are given in Figures 9 - 17. Figure 9 shows the fuel and LOX piping layouts used in the example. Figures 10 and 11 give the Nyquist plot and Phase-Gain plot for $K(j\omega)$. Similar plots are shown for $K(j\omega,G_{ox})$ in Figures 12 and 13, $K(j\omega,G_{f})$ in Figures 14 and 15, and $K(j\omega,G_{ox},G_{f})$ in Figures 16 and 17. Note that the curves for $K(j\omega,G_{ox})$ and $K(j\omega,G_{f})$ are similar, but out of phase. This is evident in the curves for $K(j\omega,G_{ox},G_{f})$.

Intermediate Mode

Graphics was added to the intermediate mode program and it was modified to run a range of frequencies and a range of τ's (sensitive time lag). After the range of τ's for a given frequency have been run and the n's displayed on the screen, the user may request a plot of n vs τ for that frequency (Fig. 18). After the range of frequencies have been run, n vs τ is plotted on one graph for each of the frequencies (Fig. 19).

Recommendations

Feedline Program

1. Speed up iteration for split pipe. A study of the convergence will have to be made to determine the best approach.

2. Generalize the split pipe to allow splits into non-identical pipes. This will require changing the logic of the program.

Mitchell's Program

1. Make it easier to use.
   a. Reduce number of input files. Seven are now used.
   b. Use dimensioned variables on input and output. Currently the program requires the user to nondimensionalize the data before it is input.

2. Add plots to the output. The code now outputs a file with n and τ to be used by another program for plotting.

Intermediate Frequency Program

1. Add split pipe and accumulators. Since these are already developed for the feedline codes, adding them will be fairly simple.

Nomenclature

a	speed of sound	ft/sec
A	area	ft^2
C	capacitance	sec
C	capacitance per unit length	sec/ft
d	diameter	ft
g_c	gravitational constant	$\mathrm{lbm\text{-}ft/lbf\text{-}sec}^2$
G	admittance	nd
k	bulk modulus	$\mathrm{lbf/ft}^2$
l	length	ft
L(I)	length of I^{th} pipe	ft
L	inductance	sec
L	inductance per unit length	sec/ft
m	no. of split lines	nd
$\dot{m}$	mass flow	lbm/sec
n	pressure interaction index	nd
n	pressure interaction factor	1/ft
p	pressure	$\mathrm{lbf/ft}^2$
s	complex frequency	1/sec
V	volume	ft^3
x	distance along pipe	ft
y	admittance	nd
z	impedance	nd
Z	impedance	nd
ρ	density	$\mathrm{lbm/ft}^3$
ω	imaginary part of frequency	rad/sec

Subscripts

c	combustion chamber	(e.g. p_c)
t	looking toward tank	(e.g. G_t)
g	looking toward engine	(e.g. Z_g)
0	lossless line	(e.g. Z_o)

List of Figures

Figure Description Page

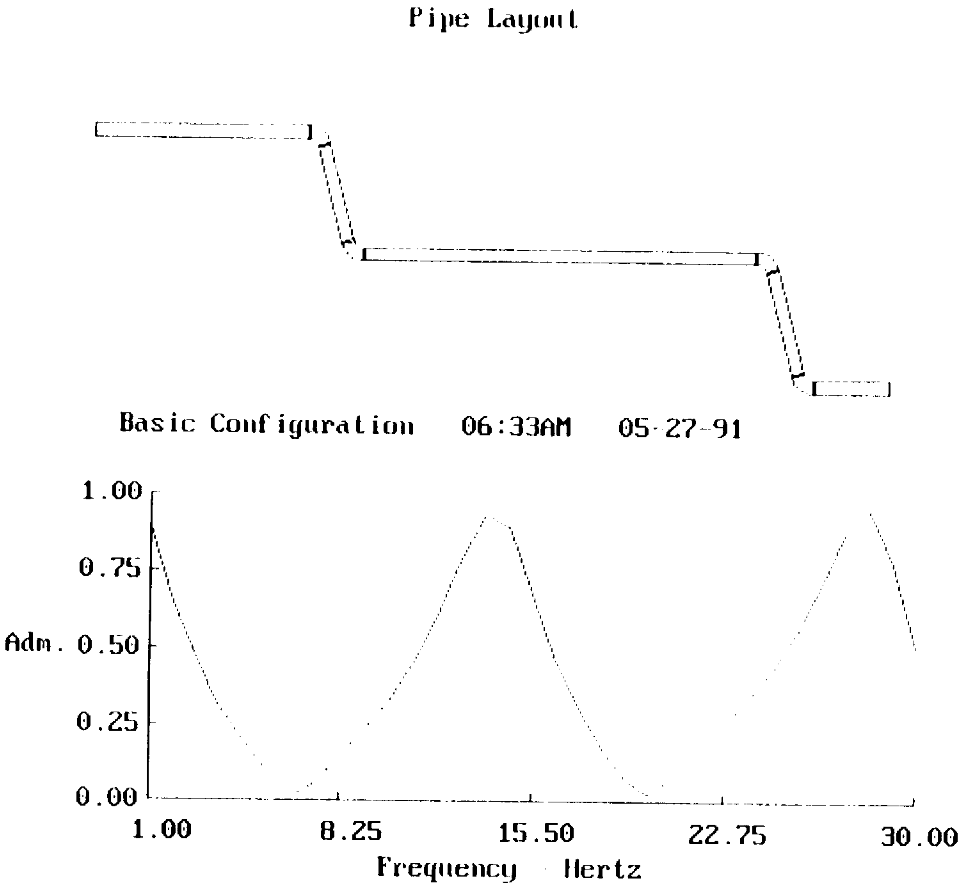

Pipe Layout
Basic Configuration 06:33AM 05-27-91
1.00
0.75
Adm. 0.50
0.25
0.00
1.00
8.25
15.50
22.75
30.00
Frequency Hertz

Figure 1a

Basic Configuration 06:33AM 05-27-91
Pressure Transfer Function = f(freq(Hertz),distance(ft))

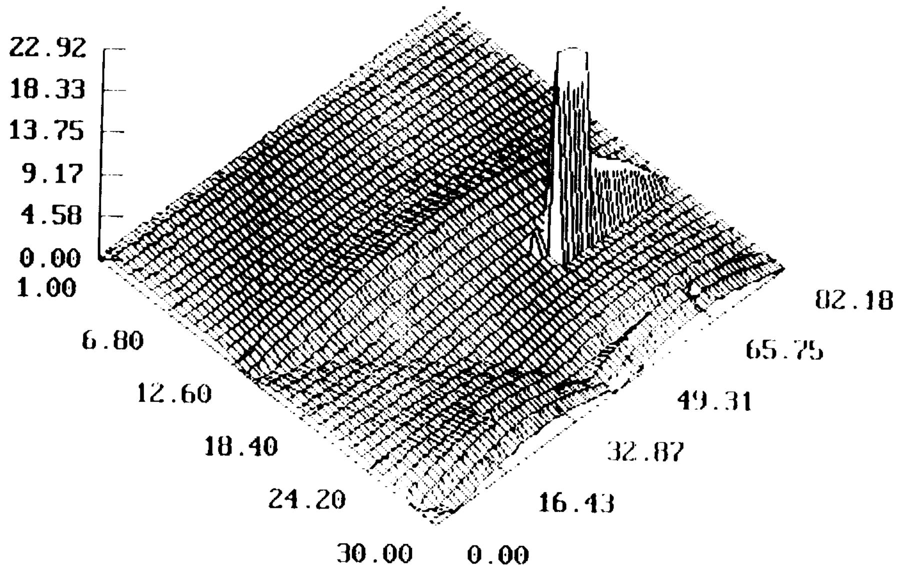


Figure 1b

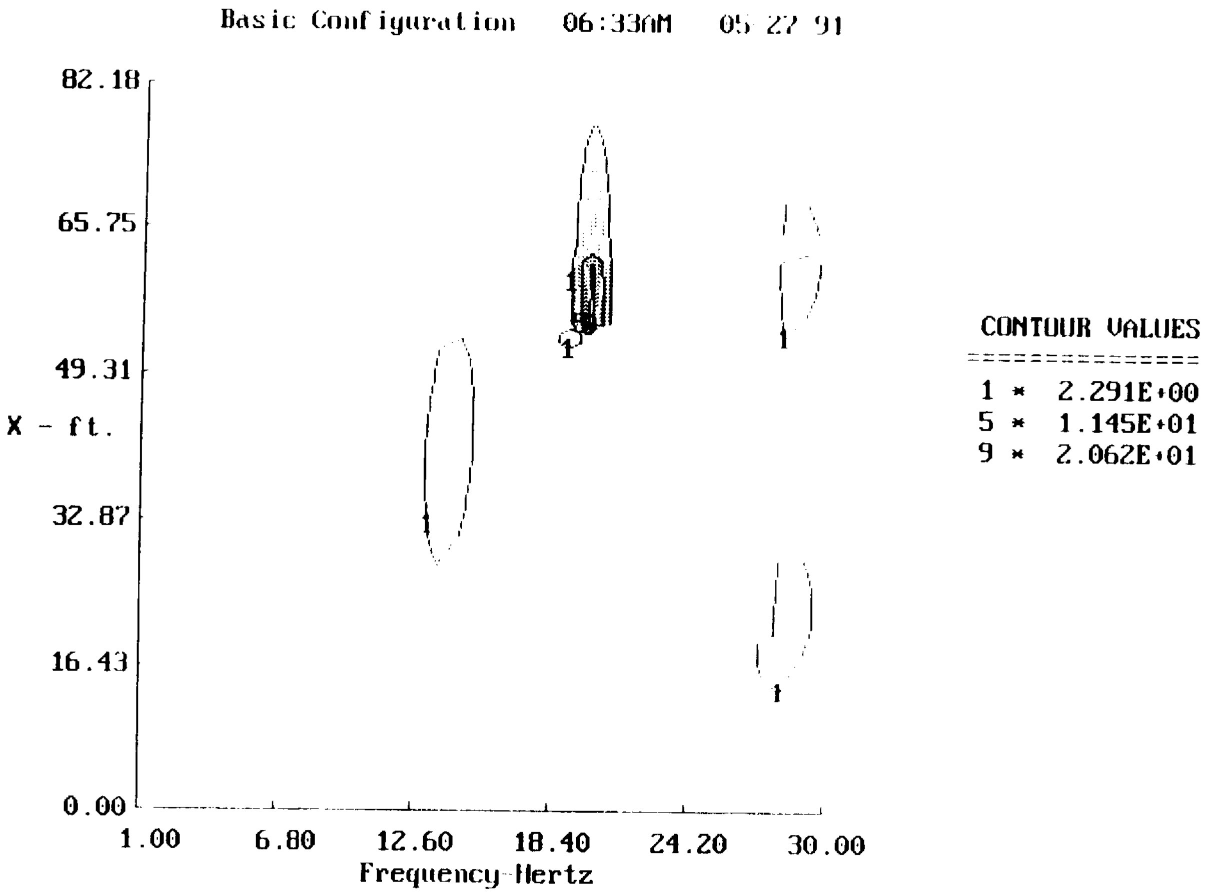

Basic Configuration 06:33AM 05 27 91
82.18
65.75
49.31
32.87
16.43
0.00
X - ft.
1.00
6.80
12.60
18.40
24.20
30.00
Frequency-Hertz
1
5
9
CONTOUR VALUES
1 * 2.291E+00
5 * 1.145E+01
9 * 2.062E+01

Figure 1c

Basic Configuration   06:33AM   05 27 91
Pressure Transfer Function = f(freq(Hertz),distance(ft))

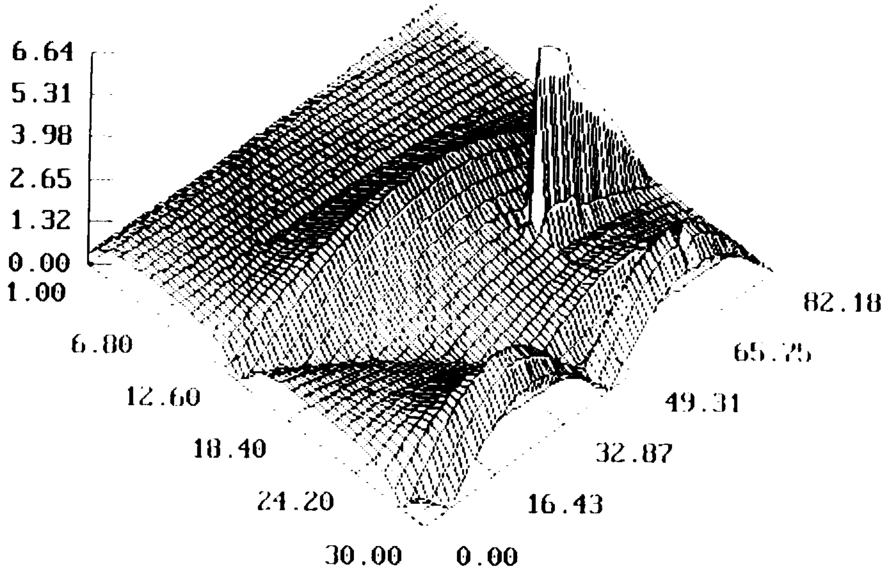
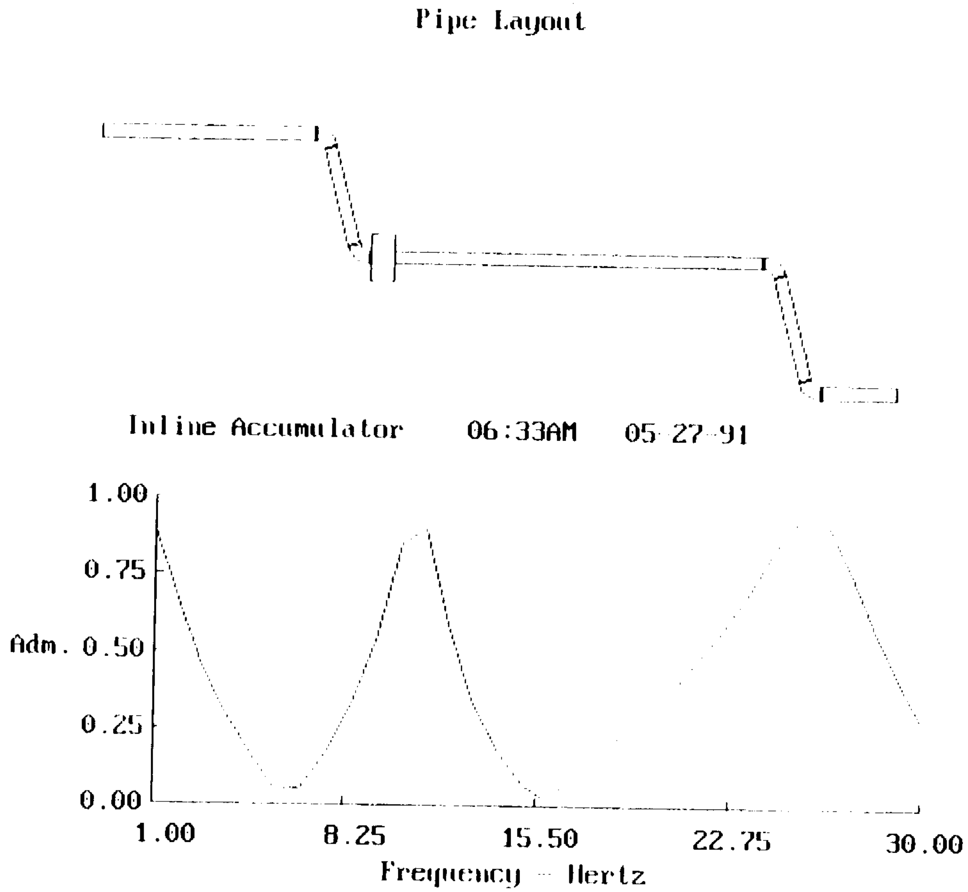


Figure 2



Pipe Layout
Inline Accumulator 06:33AM 05-27-91
1.00
0.75
Adm. 0.50
0.25
0.00
1.00
8.25
15.50
22.75
30.00
Frequency -- Hertz

Figure 3a

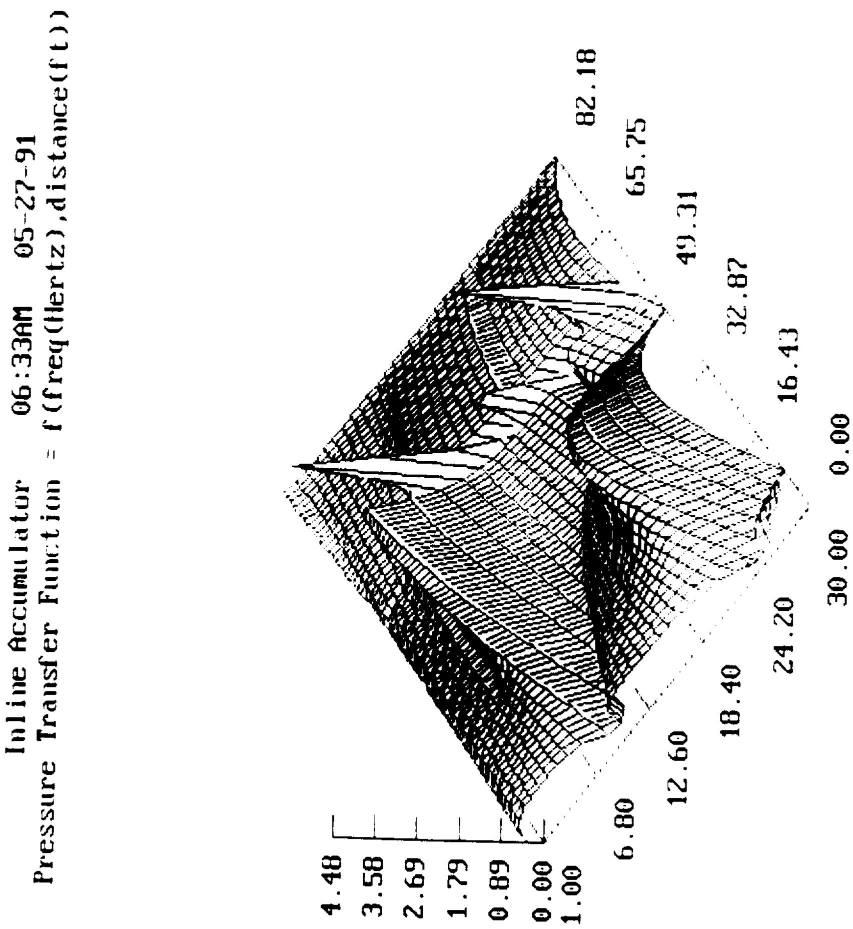

Inline Accumulator 06:33AM 05-27-91
Pressure Transfer Function = f(freq(Hertz),distance(ft))
4.48
3.58
2.69
1.79
0.89
0.00
1.00
6.80
12.60
18.40
24.20
30.00
0.00
16.43
32.87
49.31
65.75
82.18

Figure 3b

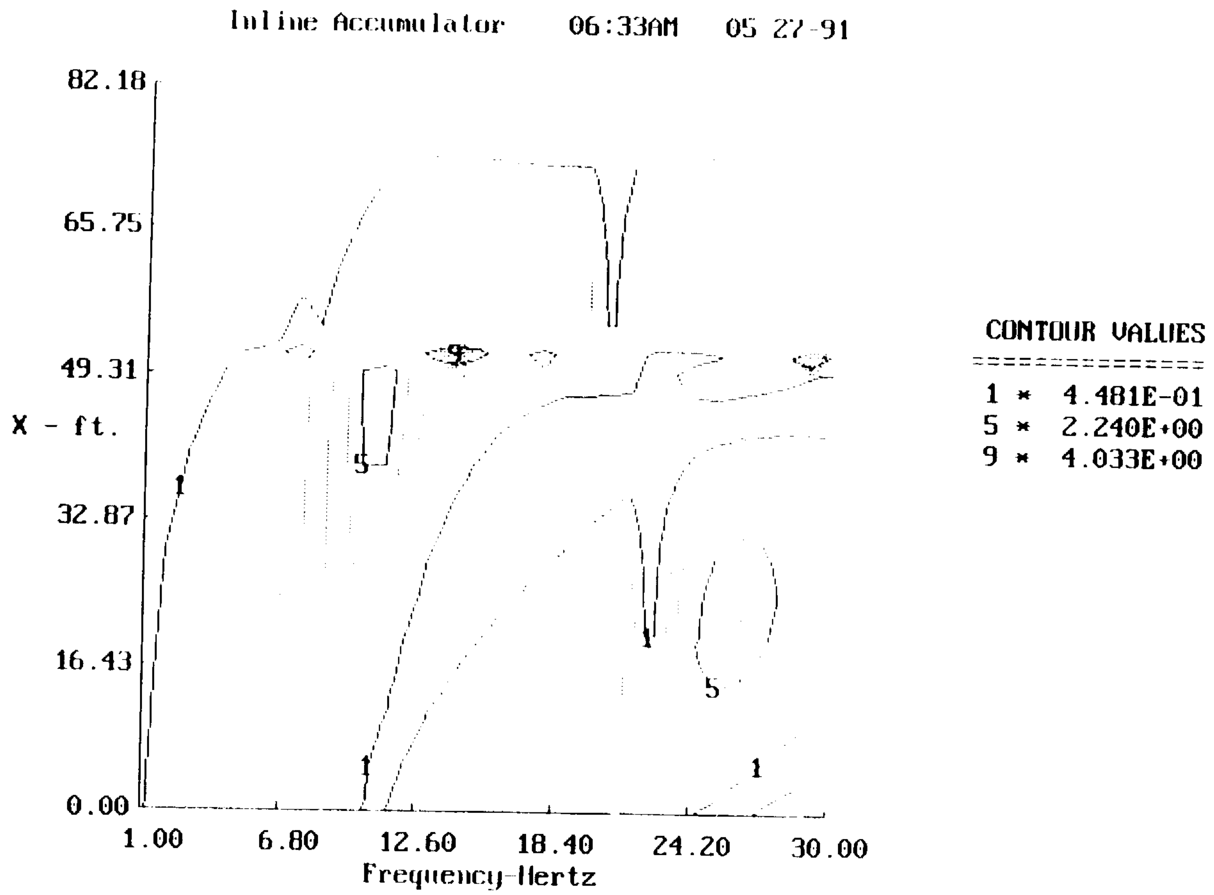


Figure 3c

ORIGINAL PAGE IS
OF POOR QUALITY

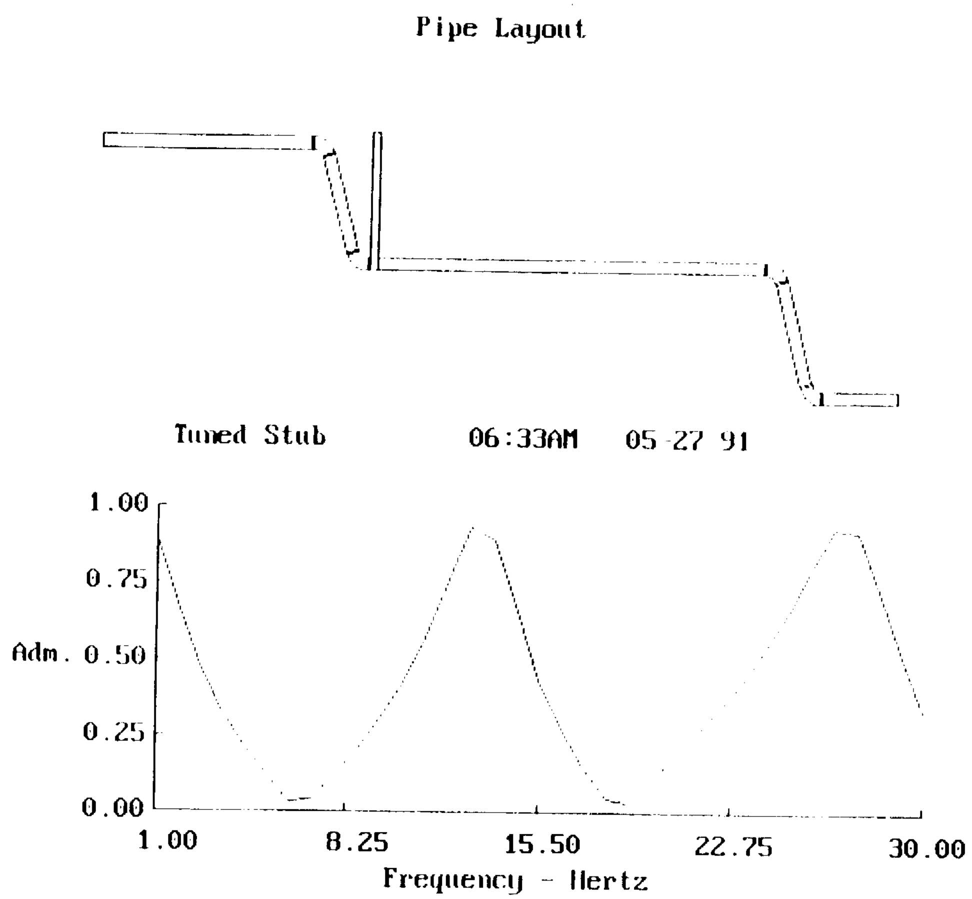


Figure 4a

ORIGINAL PAGE IS
OF POOR QUALITY

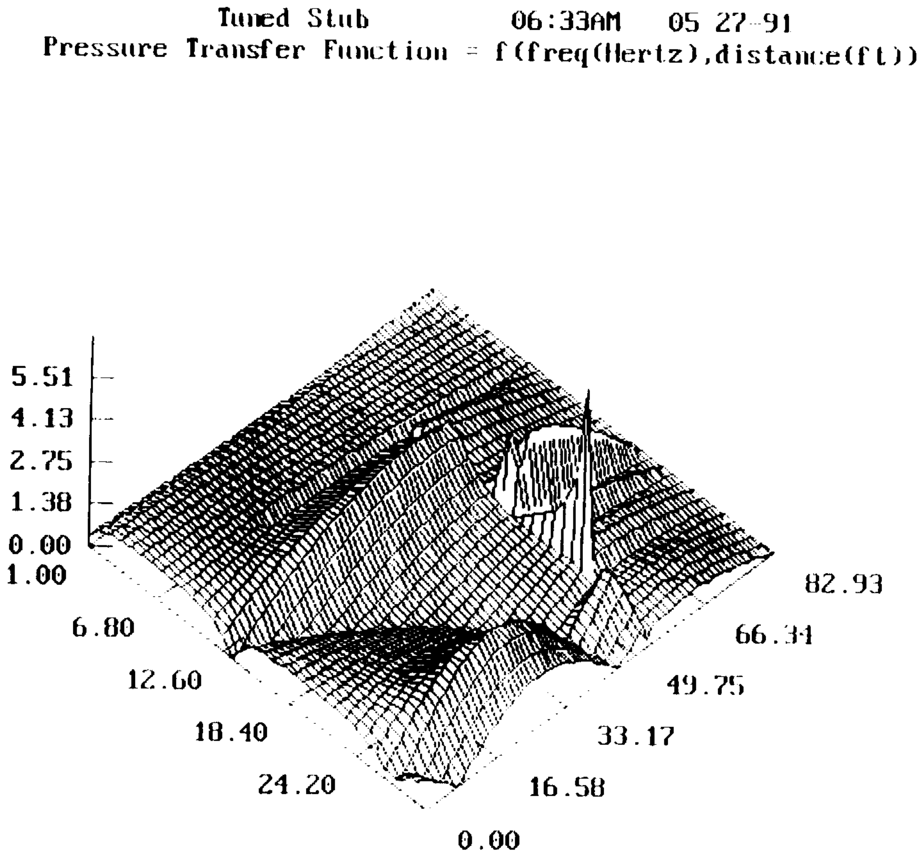

Tuned Stub 06:33AM 05 27 91
Pressure Transfer Function = f(freq(Hertz),distance(ft))
5.51
4.13
2.75
1.38
0.00
1.00
6.80
12.60
18.40
24.20
0.00
16.58
33.17
49.75
66.34
82.93

Figure 4b

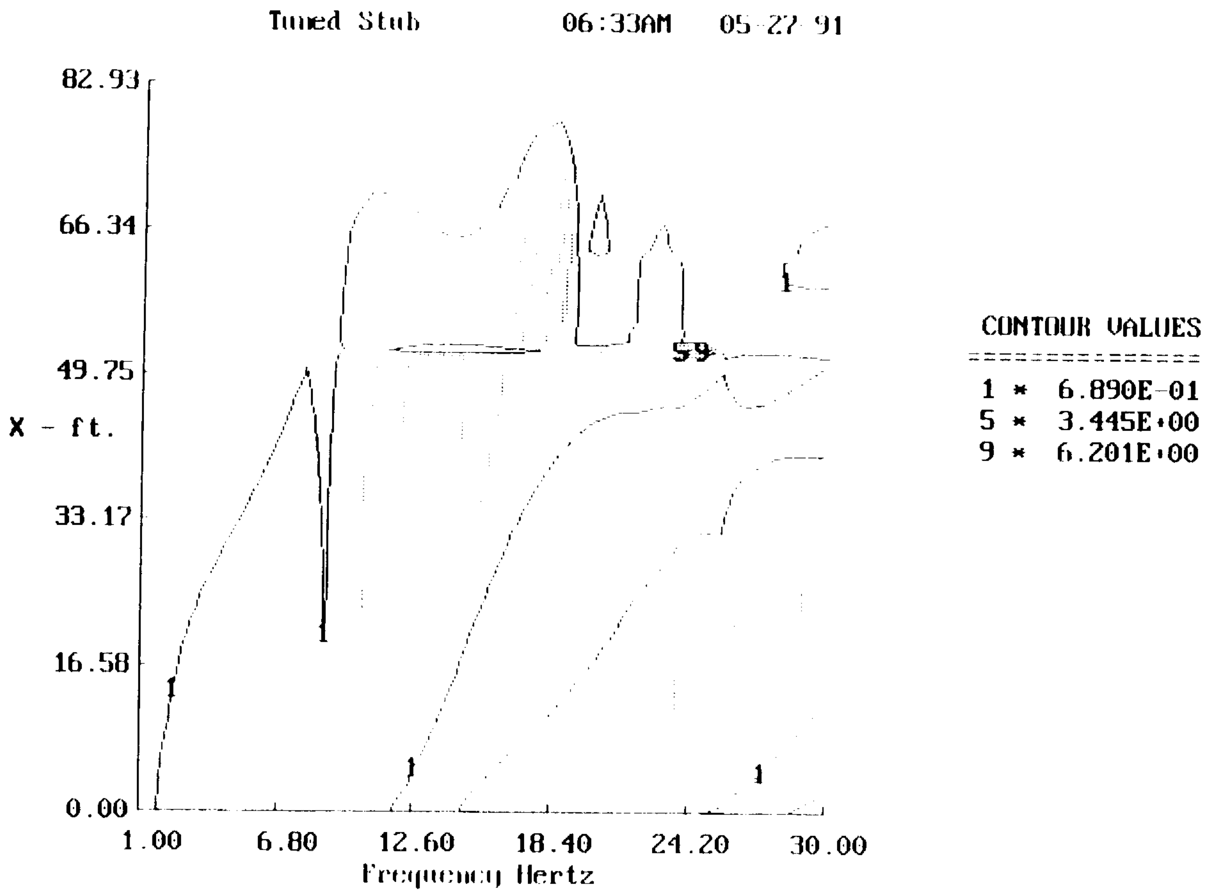

Tuned Stub
06:33AM
05-27-91
82.93
66.34
49.75
33.17
16.58
0.00
X - ft.
1.00
6.80
12.60
18.40
24.20
30.00
Frequency Hertz
1
1
1
1
1
59
CONTOUR VALUES
1 * 6.890E-01
5 * 3.445E+00
9 * 6.201E+00

Figure 4c

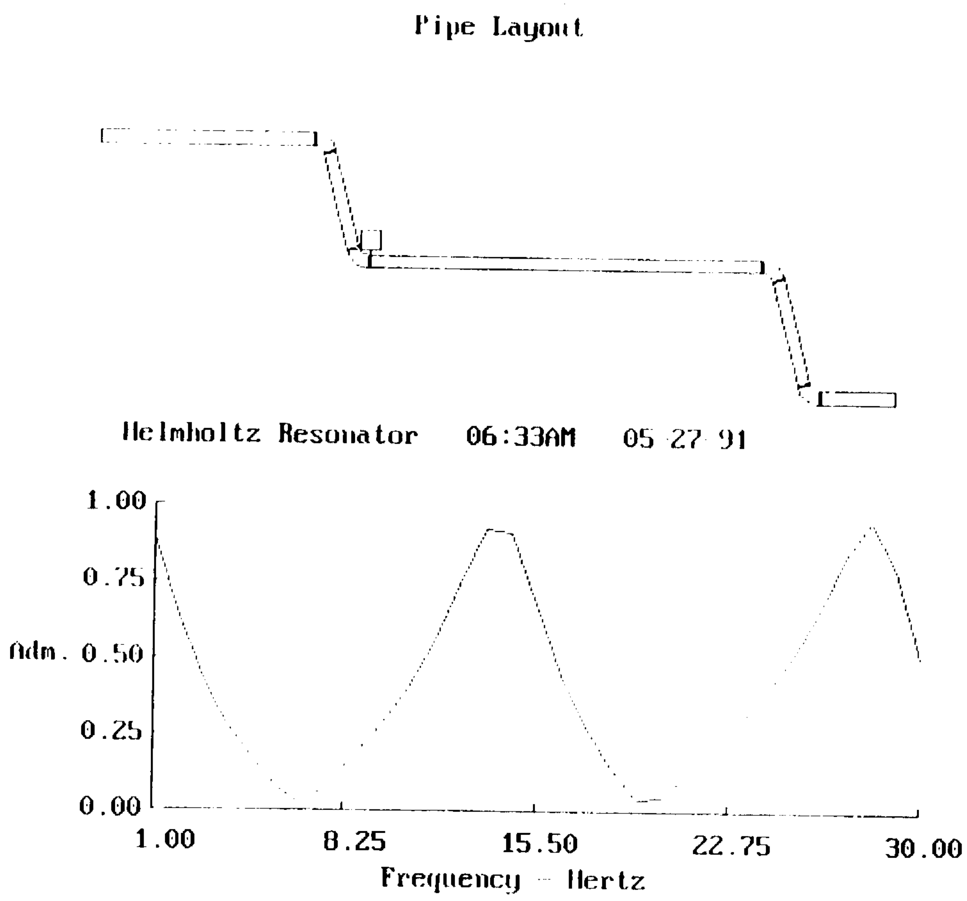

Pipe Layout
Helmholtz Resonator 06:33AM 05-27-91
1.00
0.75
Adm. 0.50
0.25
0.00
1.00
8.25
15.50
22.75
30.00
Frequency — Hertz

Figure 5a

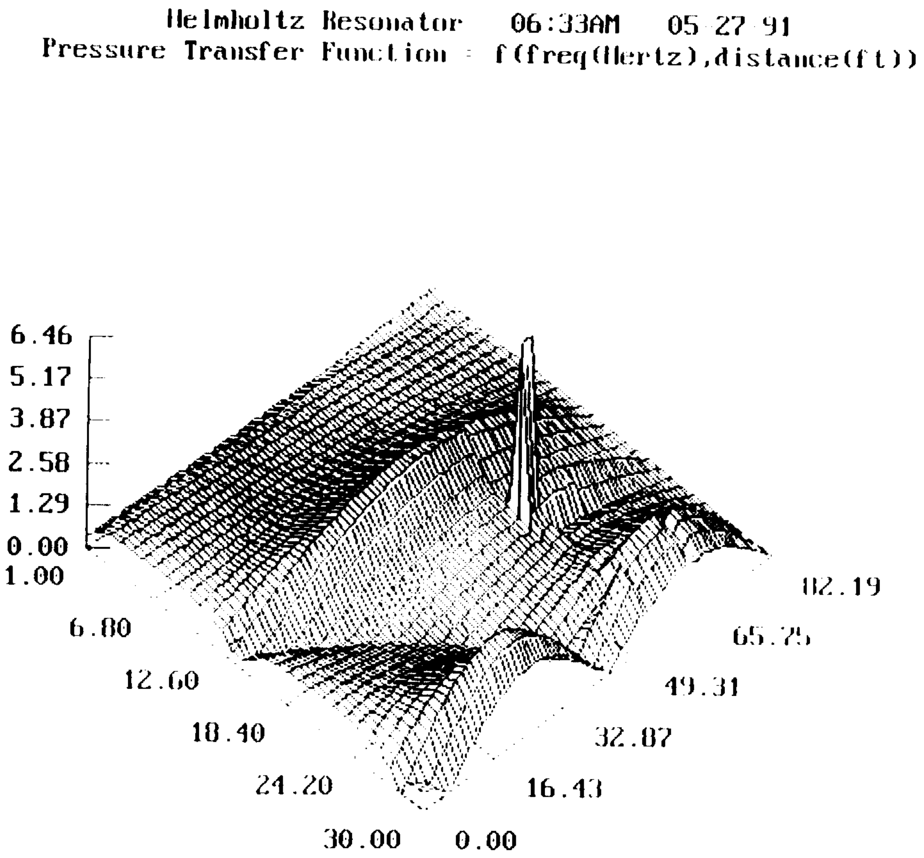

Helmholtz Resonator 06:33AM 05 27 91
Pressure Transfer Function = f(freq(Hertz),distance(ft))
6.46
5.17
3.87
2.58
1.29
0.00
1.00
6.80
12.60
18.40
24.20
30.00
0.00
16.43
32.87
49.31
65.75
82.19

Figure 5b

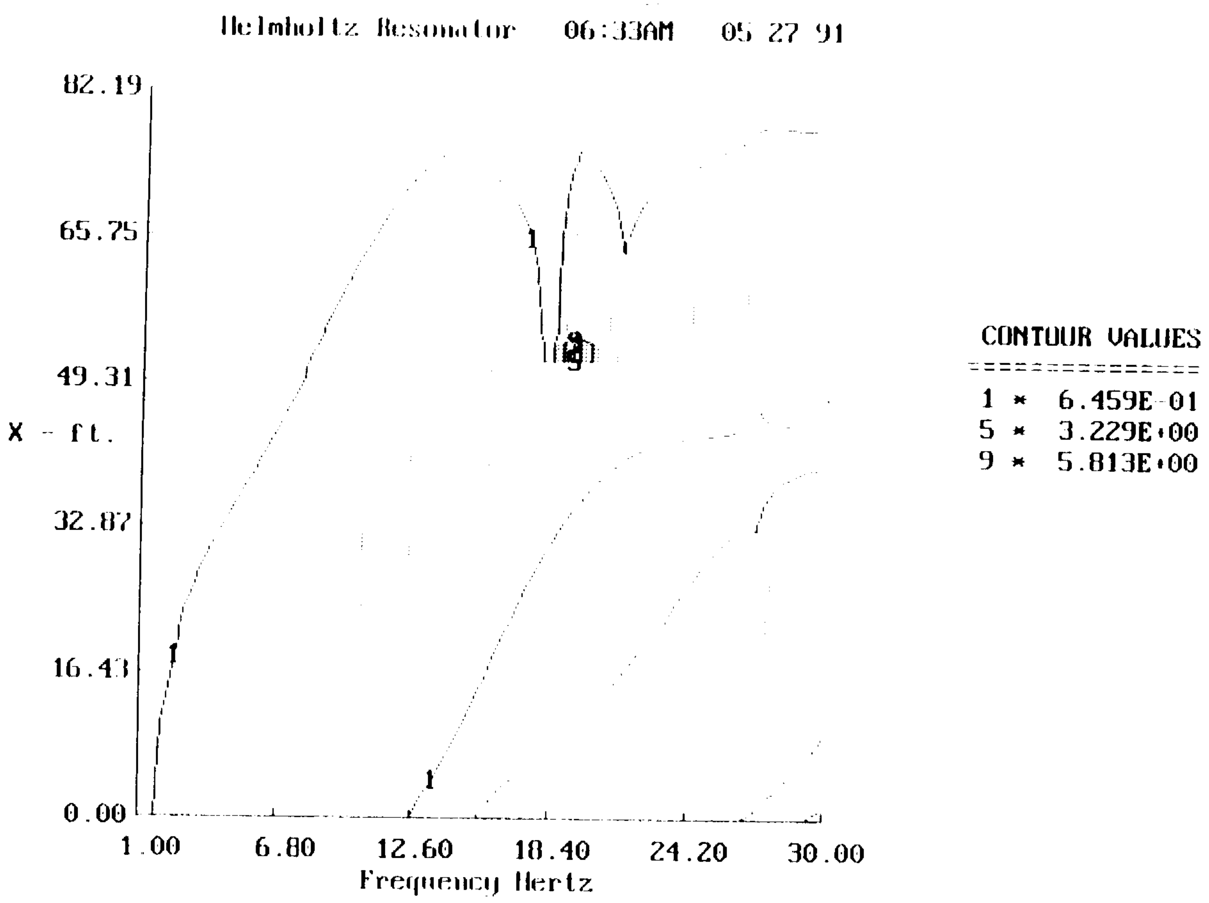


Figure 5c

ORIGINAL PAGE IS
OF POOR QUALITY

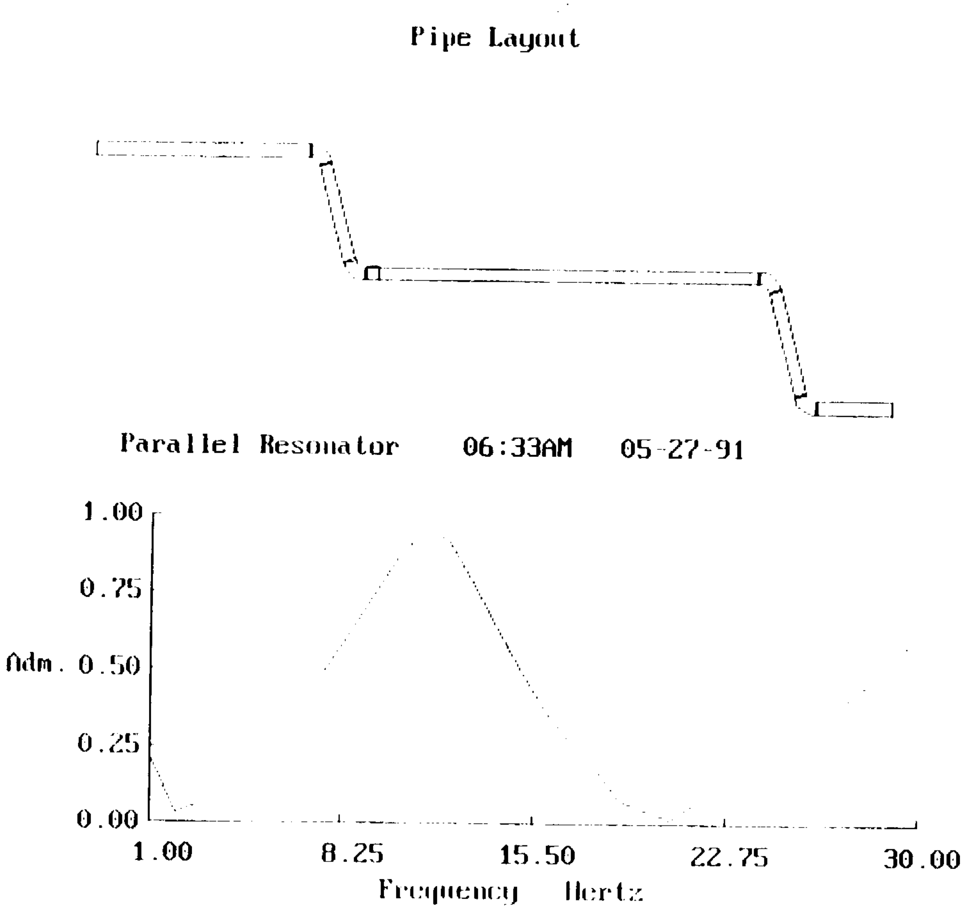
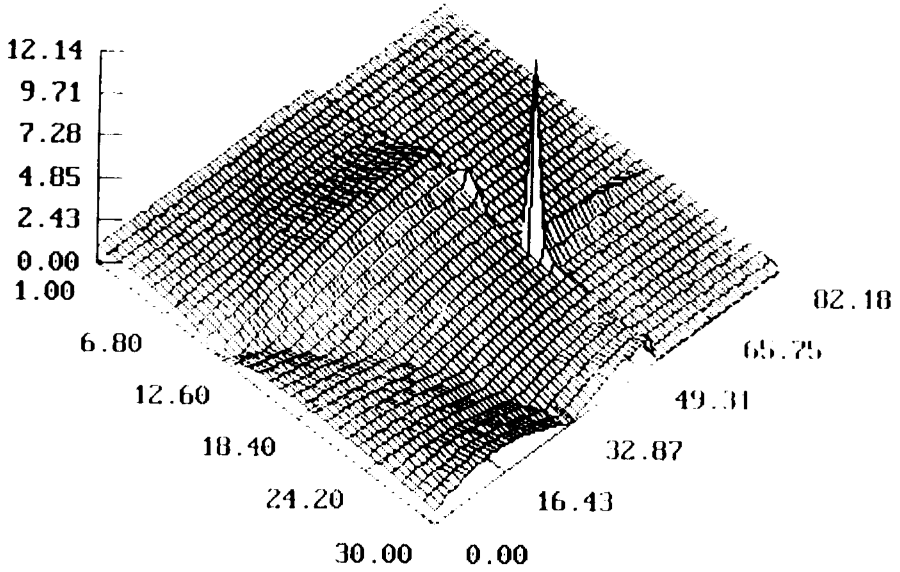


Figure 6a

ORIGINAL PAGE IS
OF POOR QUALITY



Parallel Resonator 06:33AM 05 27-91
Pressure Transfer Function = f(freq(Hertz),distance(ft))

Figure 6b

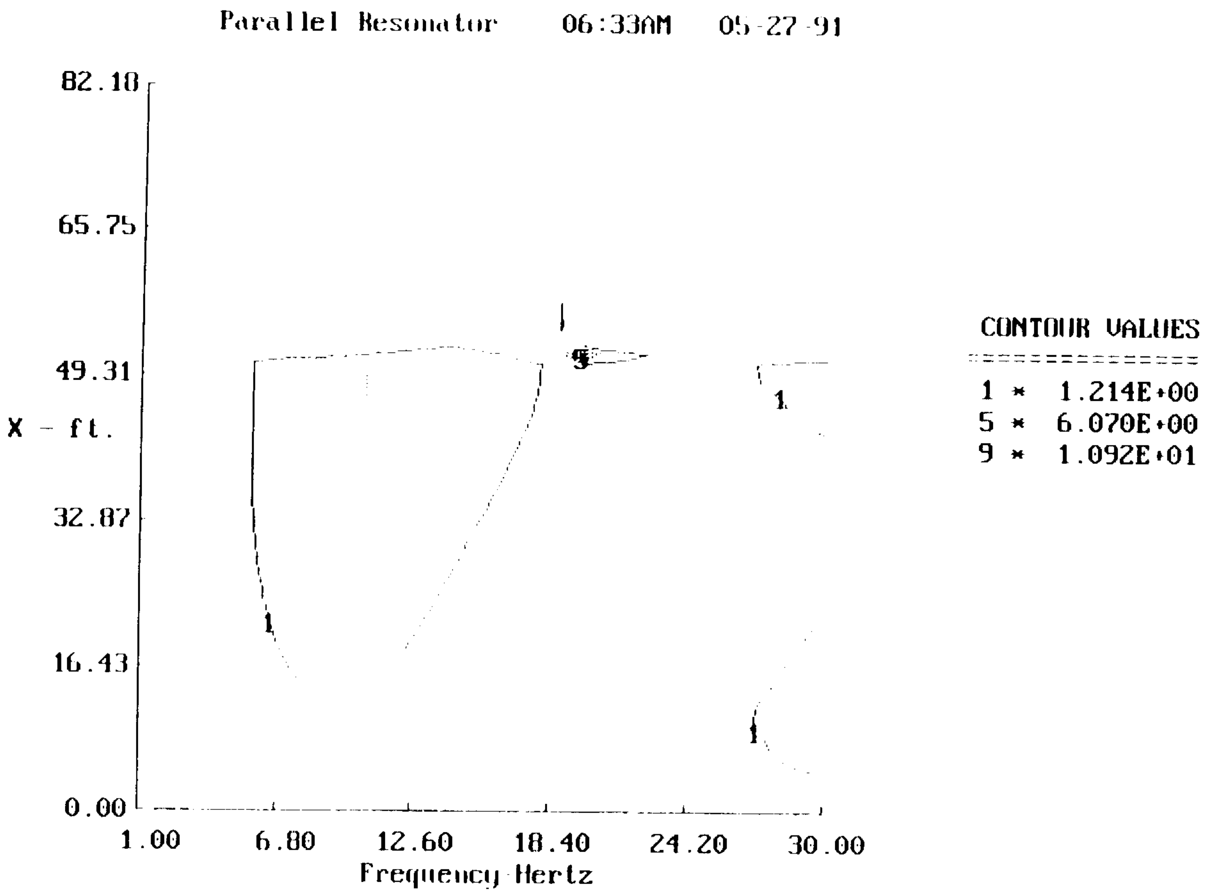

Parallel Resonator 06:33AM 05-27-91
82.18
65.75
49.31
32.87
16.43
0.00
X - ft.
1.00
6.80
12.60
18.40
24.20
30.00
Frequency Hertz
1
5
9
1
1
CONTOUR VALUES
1 * 1.214E+00
5 * 6.070E+00
9 * 1.092E+01

Figure 6c

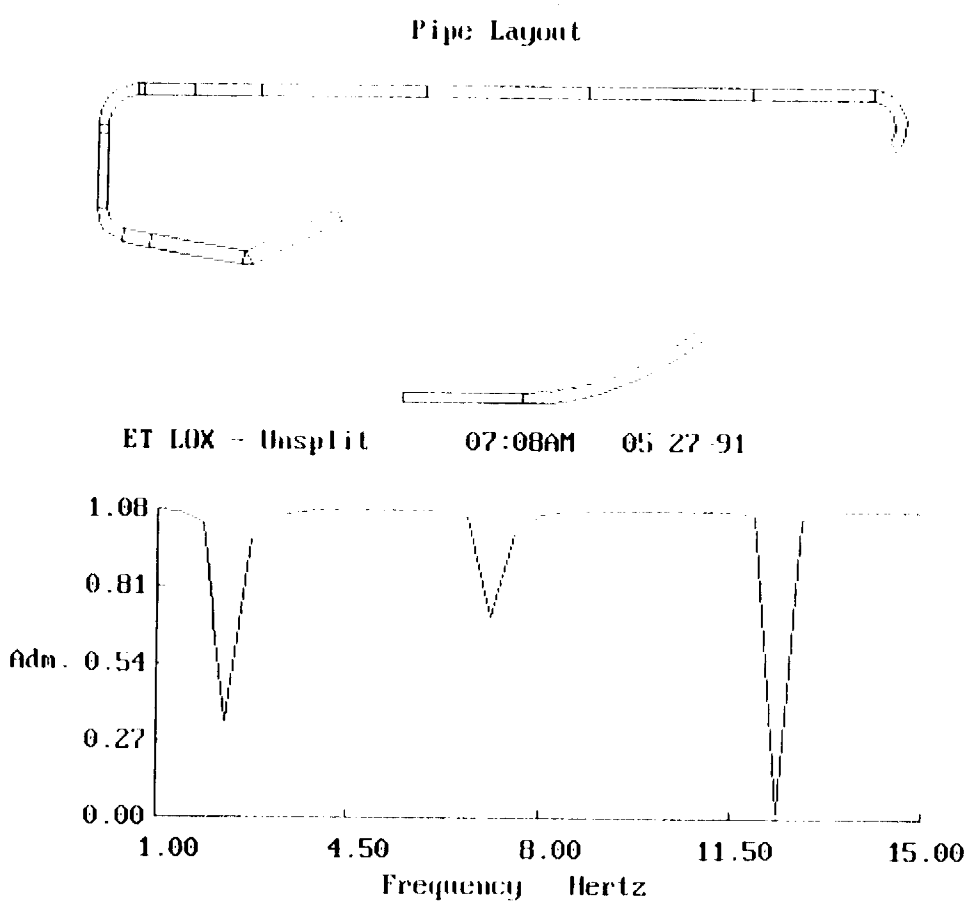

Pipe Layout
ET LOX - Unsplit 07:08AM 05 27 91
Adm.
1.08
0.81
0.54
0.27
0.00
1.00
4.50
8.00
11.50
15.00
Frequency Hertz

Figure 7a

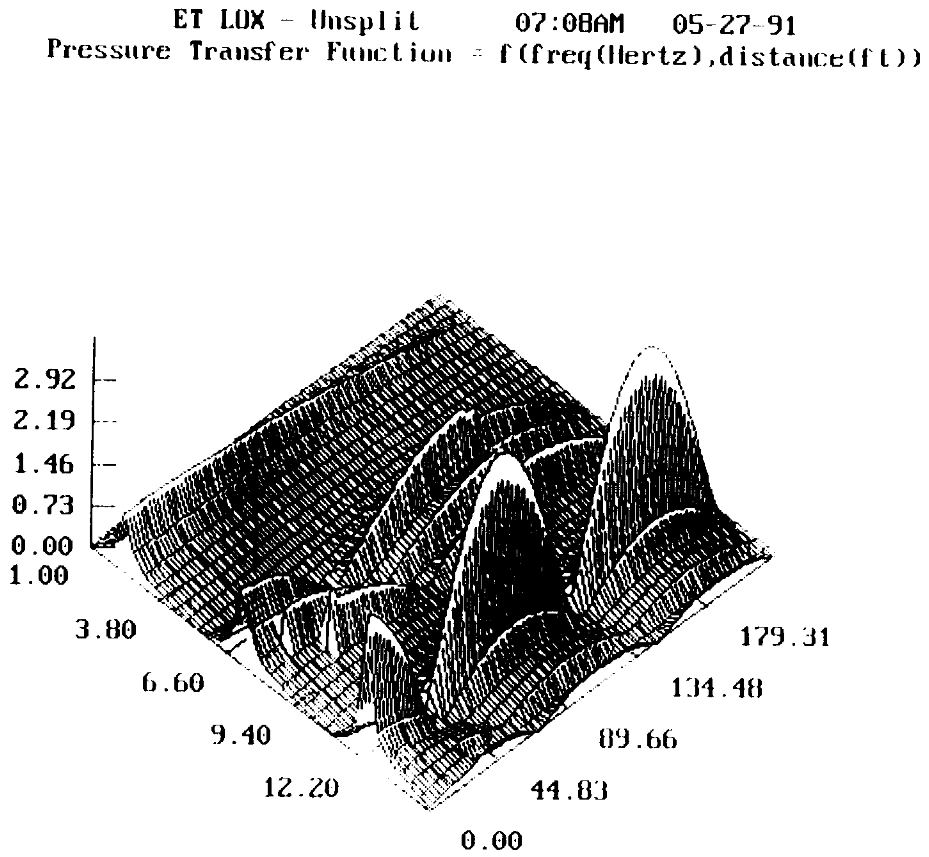

ET LOX - Unsplit 07:08AM 05-27-91
Pressure Transfer Function = f(freq(Hertz),distance(ft))
2.92
2.19
1.46
0.73
0.00
1.00
3.80
6.60
9.40
12.20
0.00
44.83
89.66
134.48
179.31

Figure 7b

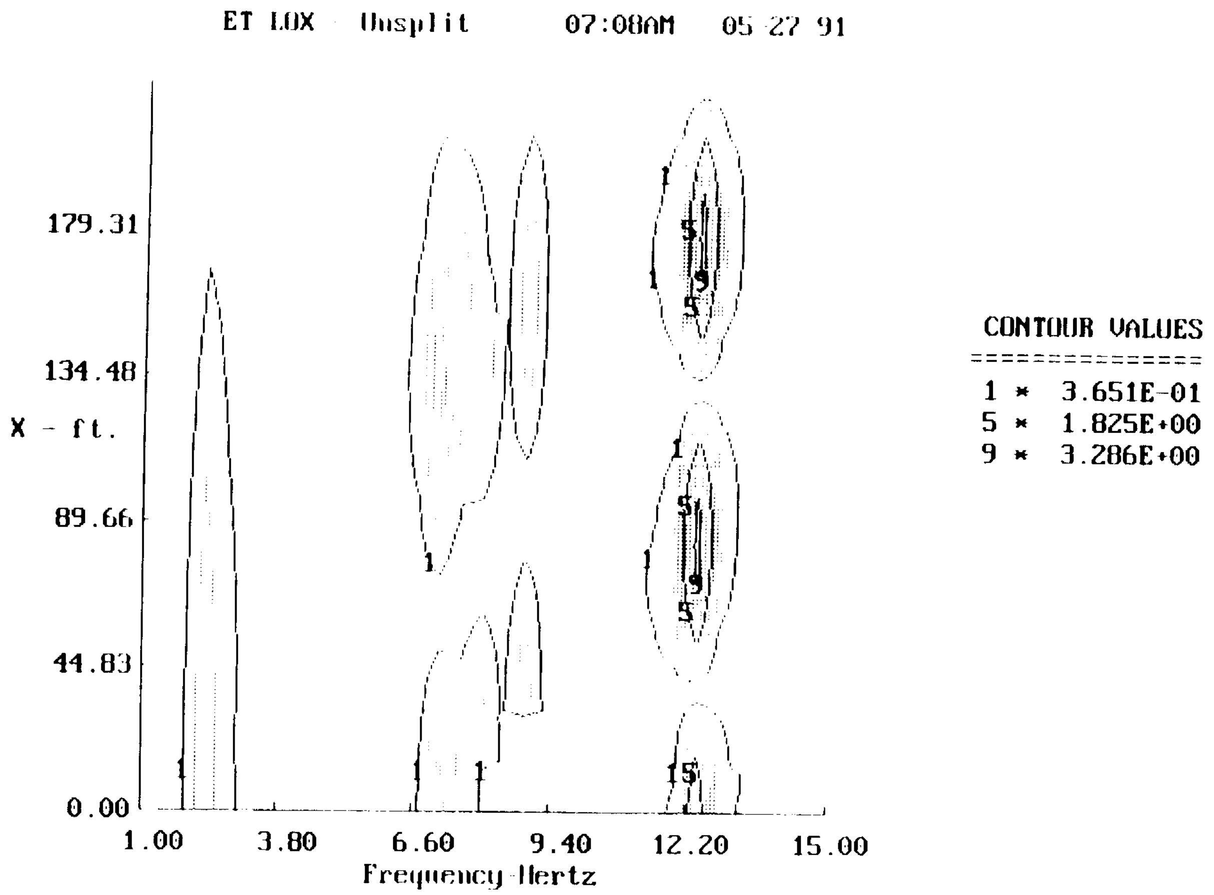

ET LOX Unsplit 07:08AM 05 27 91
179.31
134.48
X - ft.
89.66
44.83
0.00
1.00
3.80
6.60
9.40
12.20
15.00
Frequency Hertz
1
5
9
CONTOUR VALUES
1 * 3.651E-01
5 * 1.825E+00
9 * 3.286E+00

Figure 7c

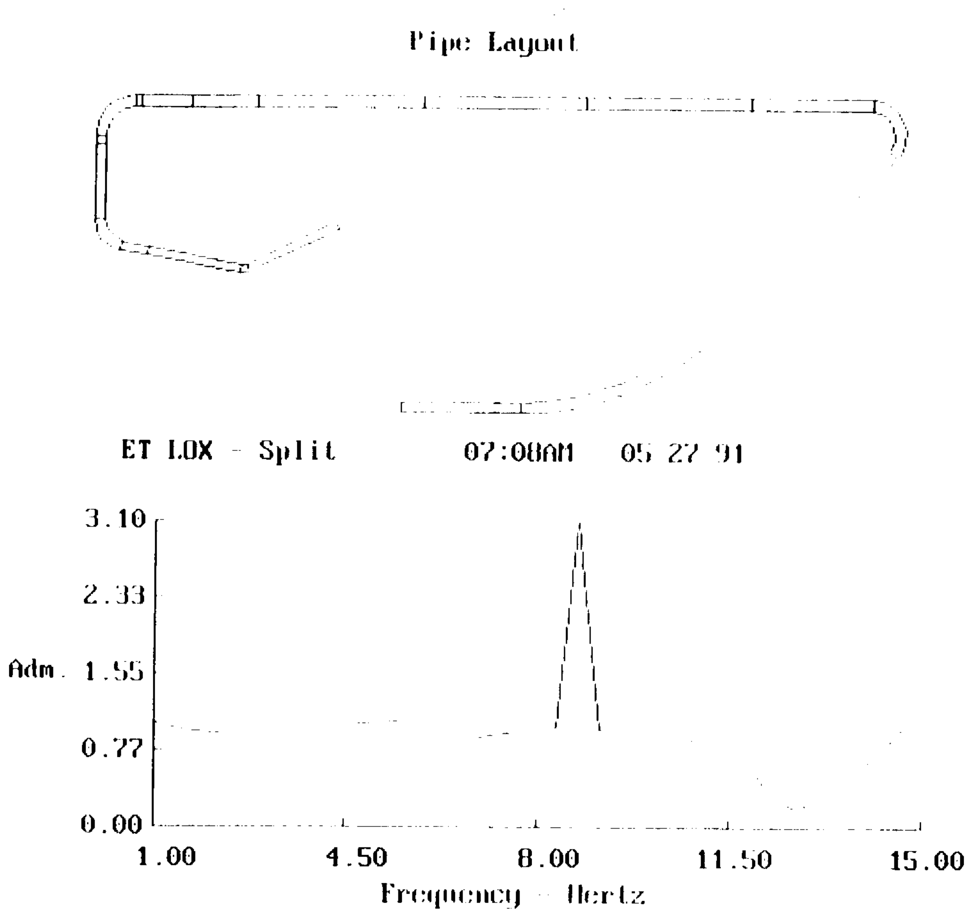

Pipe Layout
ET LOX - Split
07:08AM 05 27 91
3.10
2.33
Adm. 1.55
0.77
0.00
1.00
4.50
8.00
11.50
15.00
Frequency - Hertz

Figure 8a

ET LOX - Split 07:08AM 05-27-91
Pressure Transfer Function = f(freq(Hertz),distance(ft))

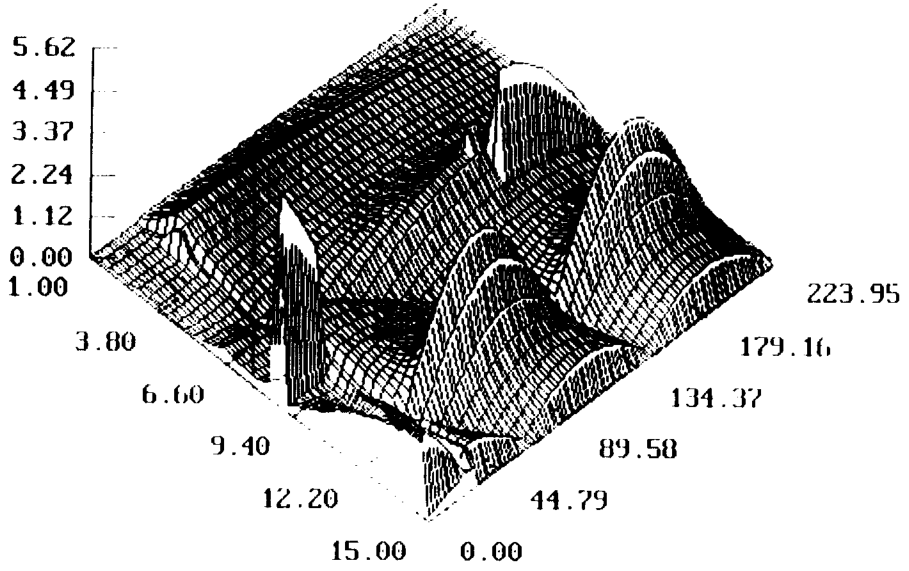


Figure 8b

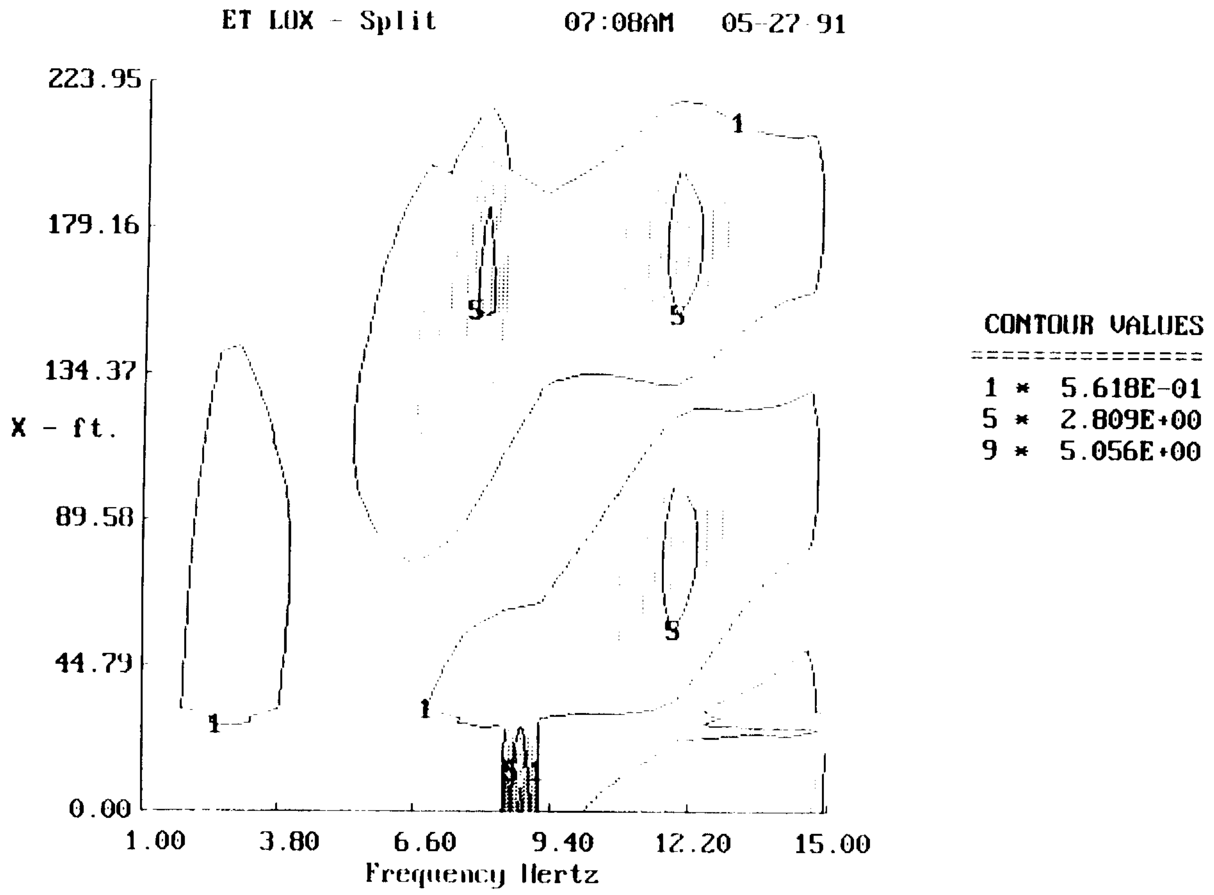
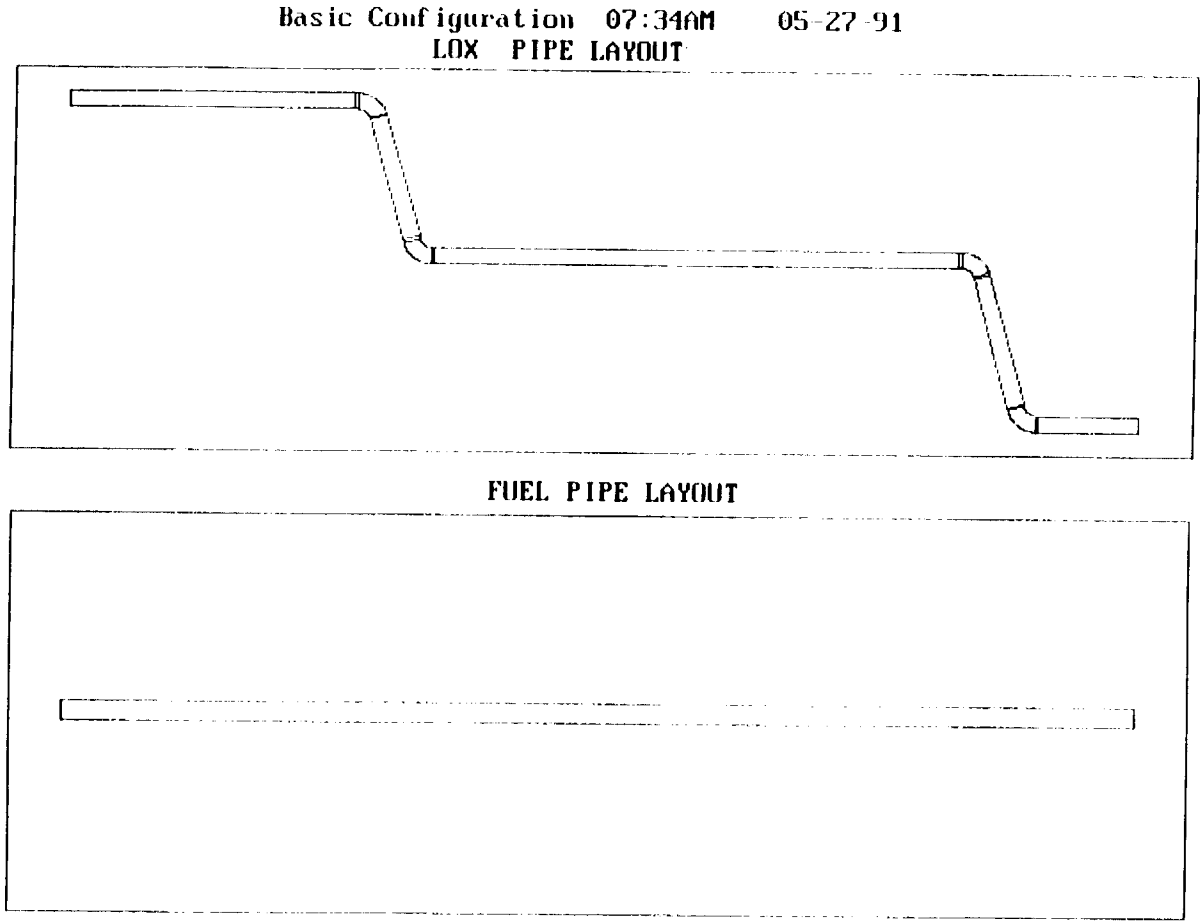

ET LOX - Split 07:08AM 05-27-91
223.95
179.16
134.37
X - ft.
89.58
44.79
0.00
1.00
3.80
6.60
9.40
12.20
15.00
Frequency Hertz
1
5
CONTOUR VALUES
==============
1 * 5.618E-01
5 * 2.809E+00
9 * 5.056E+00

Figure 8c



Basic Configuration 07:34AM 05-27-91
LOX PIPE LAYOUT
FUEL PIPE LAYOUT

Figure 9

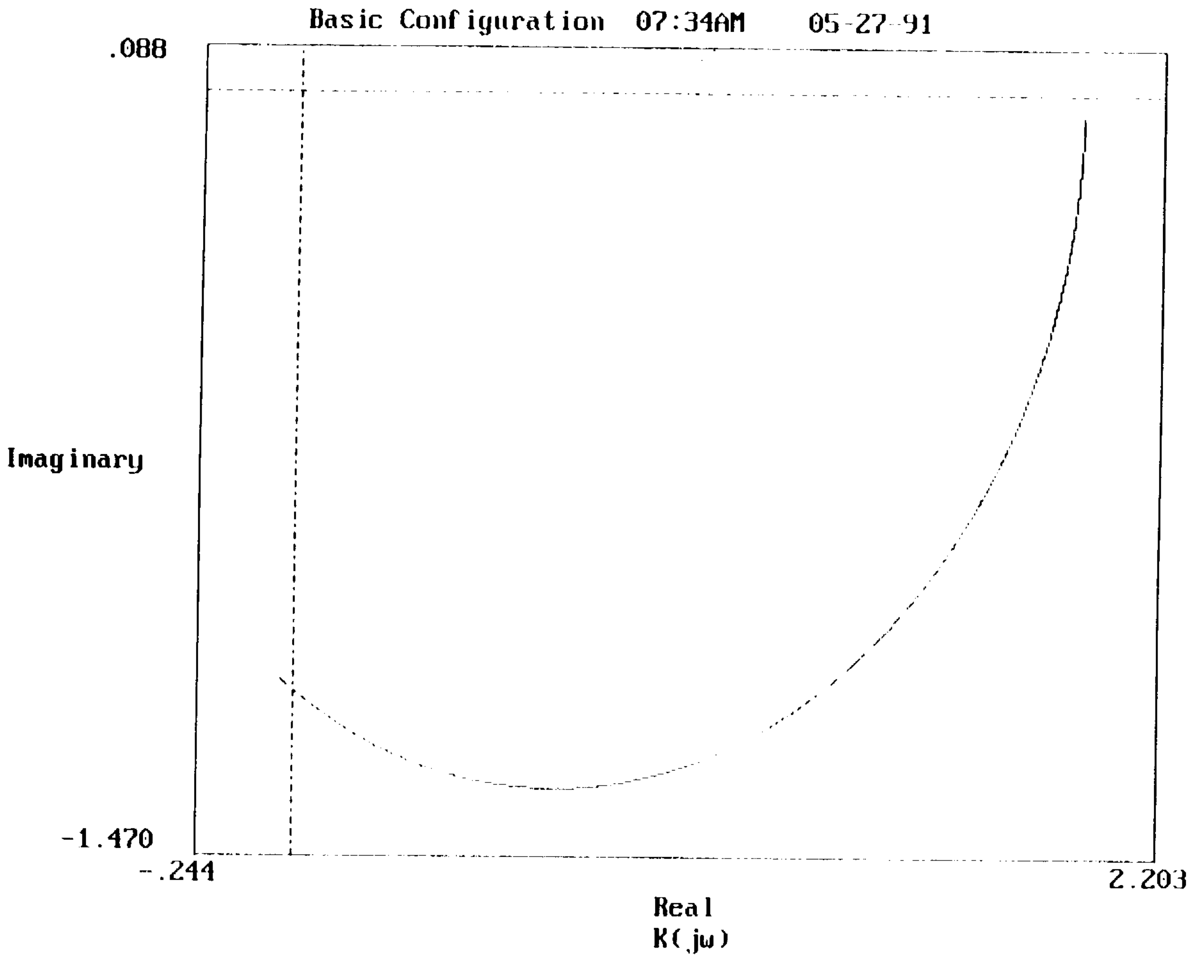

Basic Configuration 07:34AM 05-27-91
.088
Imaginary
-1.470
-.244
2.203
Real
K(jw)

Figure 10

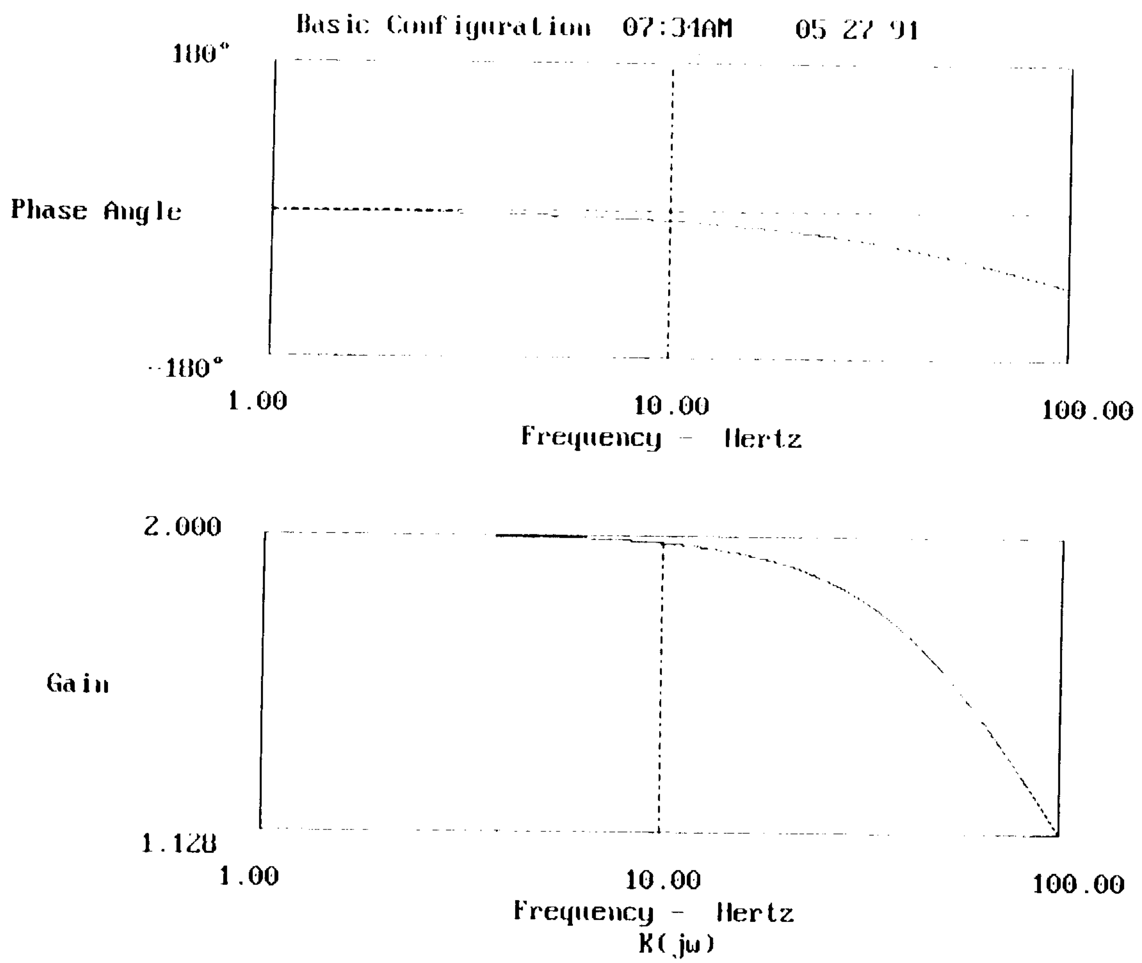

Basic Configuration 07:34AM 05 27 91
180°
Phase Angle
-180°
1.00
10.00
100.00
Frequency - Hertz
2.000
Gain
1.128
1.00
10.00
100.00
Frequency - Hertz
K(jw)

Figure 11

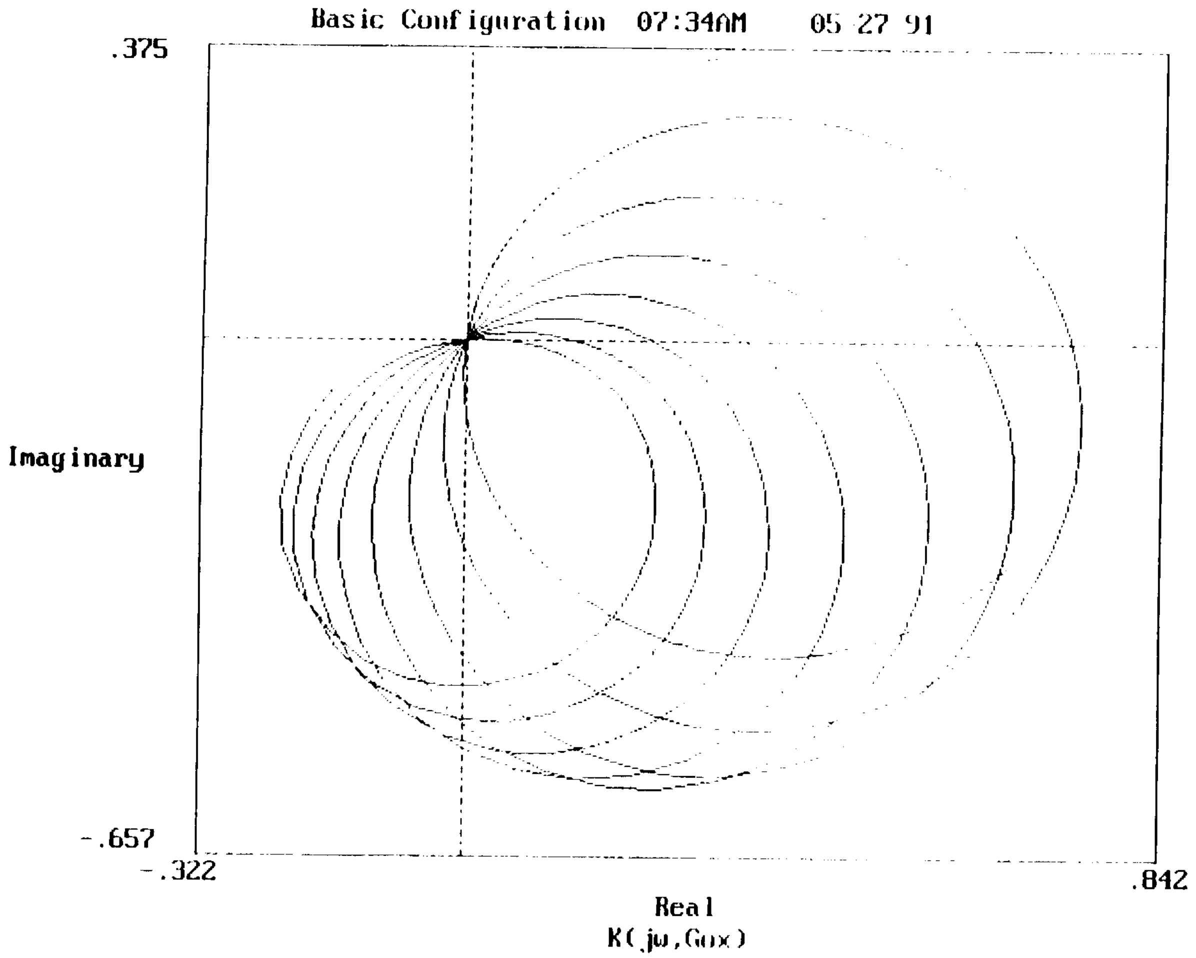

Basic Configuration 07:34AM 05 27 91
.375
-.657
Imaginary
-.322
.842
Real
K(jω,Gox)

Figure 12

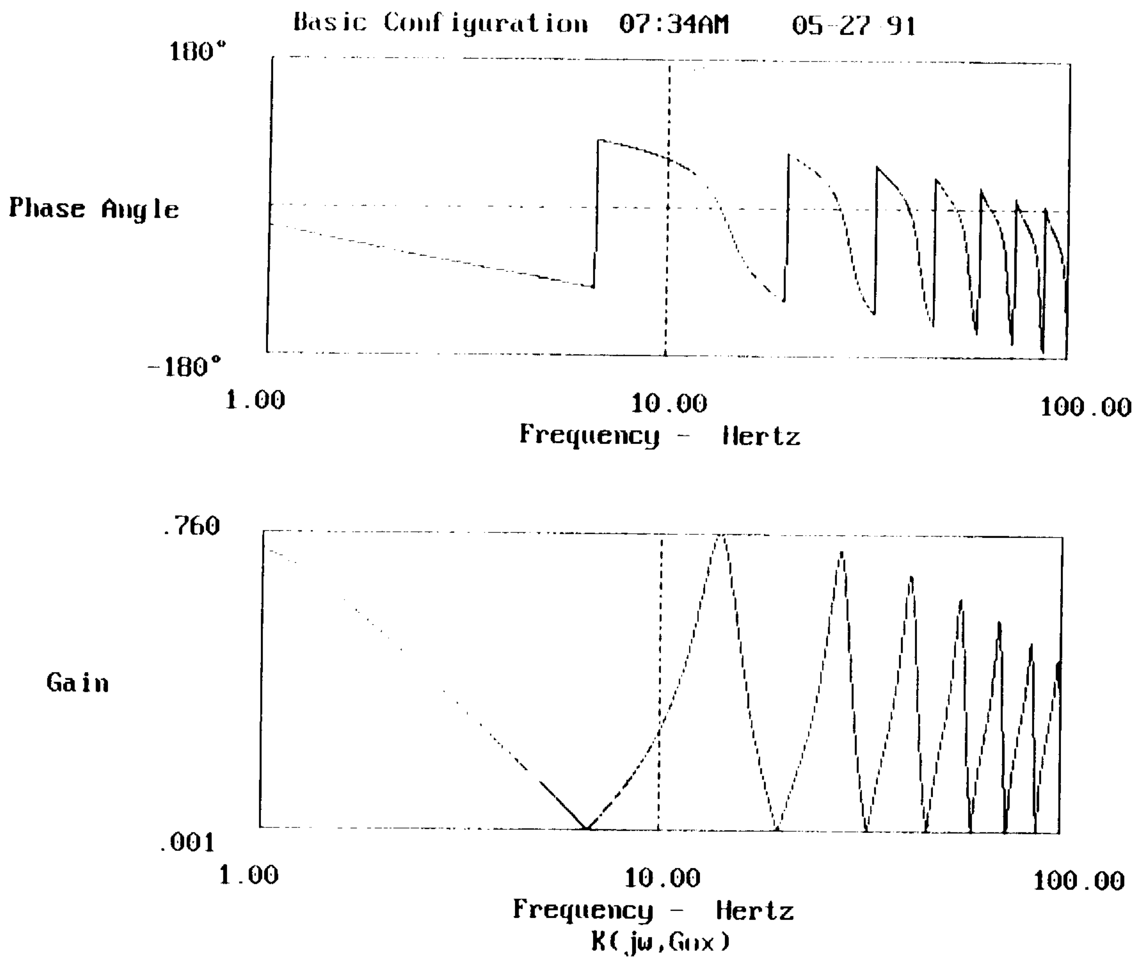

Basic Configuration 07:34AM 05-27-91
180°
Phase Angle
-180°
1.00
10.00
100.00
Frequency - Hertz
.760
Gain
.001
1.00
10.00
100.00
Frequency - Hertz
K(jw,Gox)

Figure 13

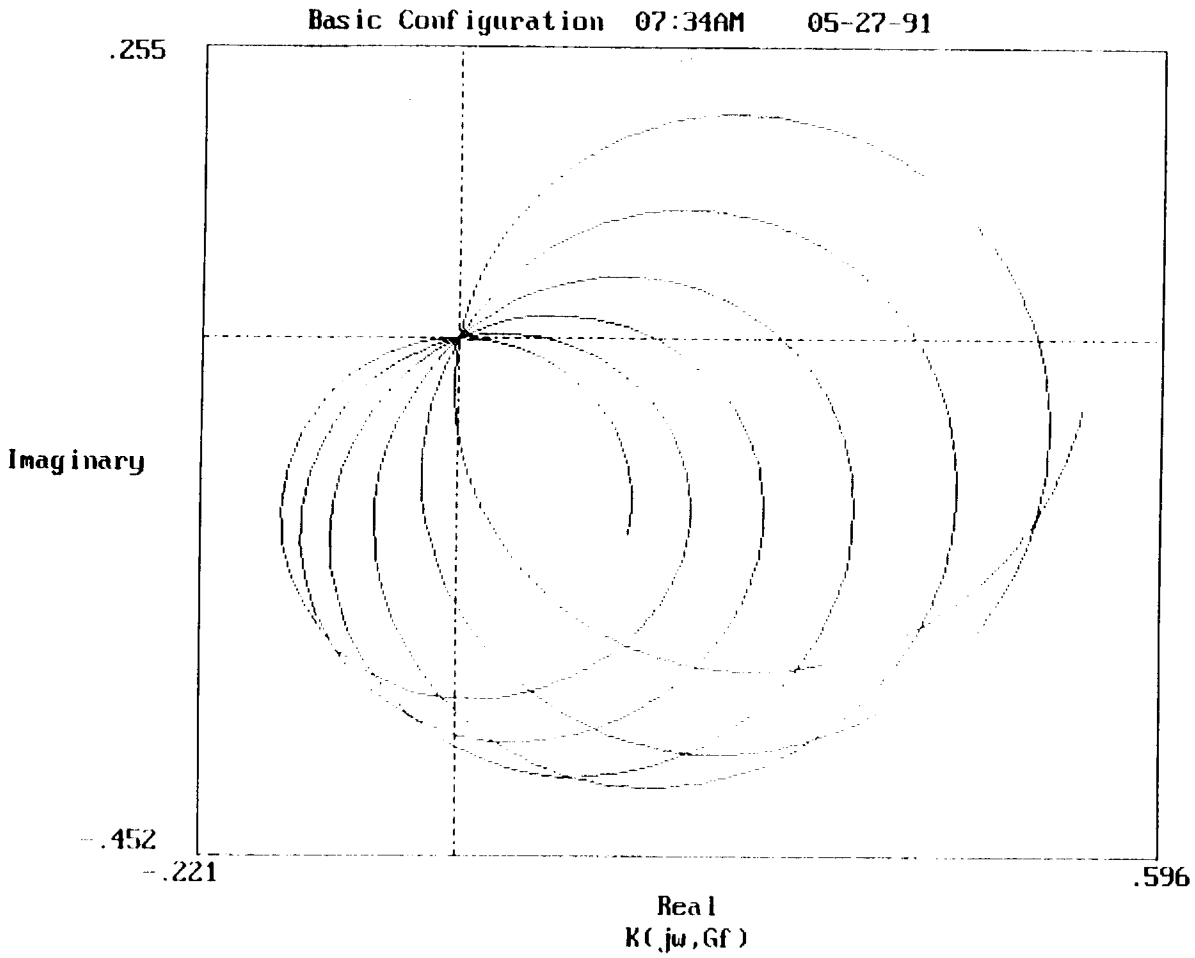

Basic Configuration 07:34AM 05-27-91
.255
Imaginary
-.452
-.221
.596
Real
K(jw,Gf)

Figure 14

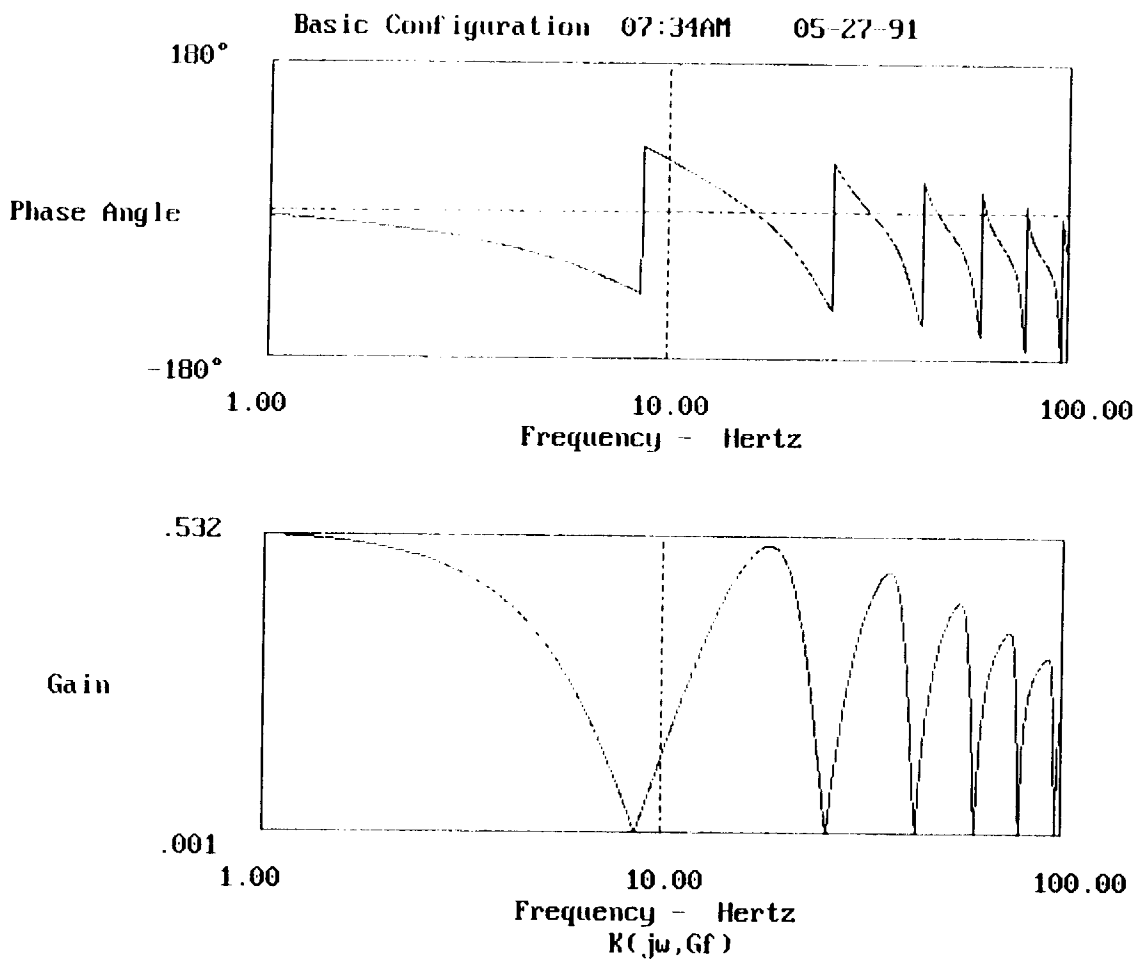

Basic Configuration 07:34AM 05-27-91
180°
Phase Angle
-180°
1.00
10.00
100.00
Frequency - Hertz
.532
Gain
.001
1.00
10.00
100.00
Frequency - Hertz
K(jw,Gf)

Figure 15

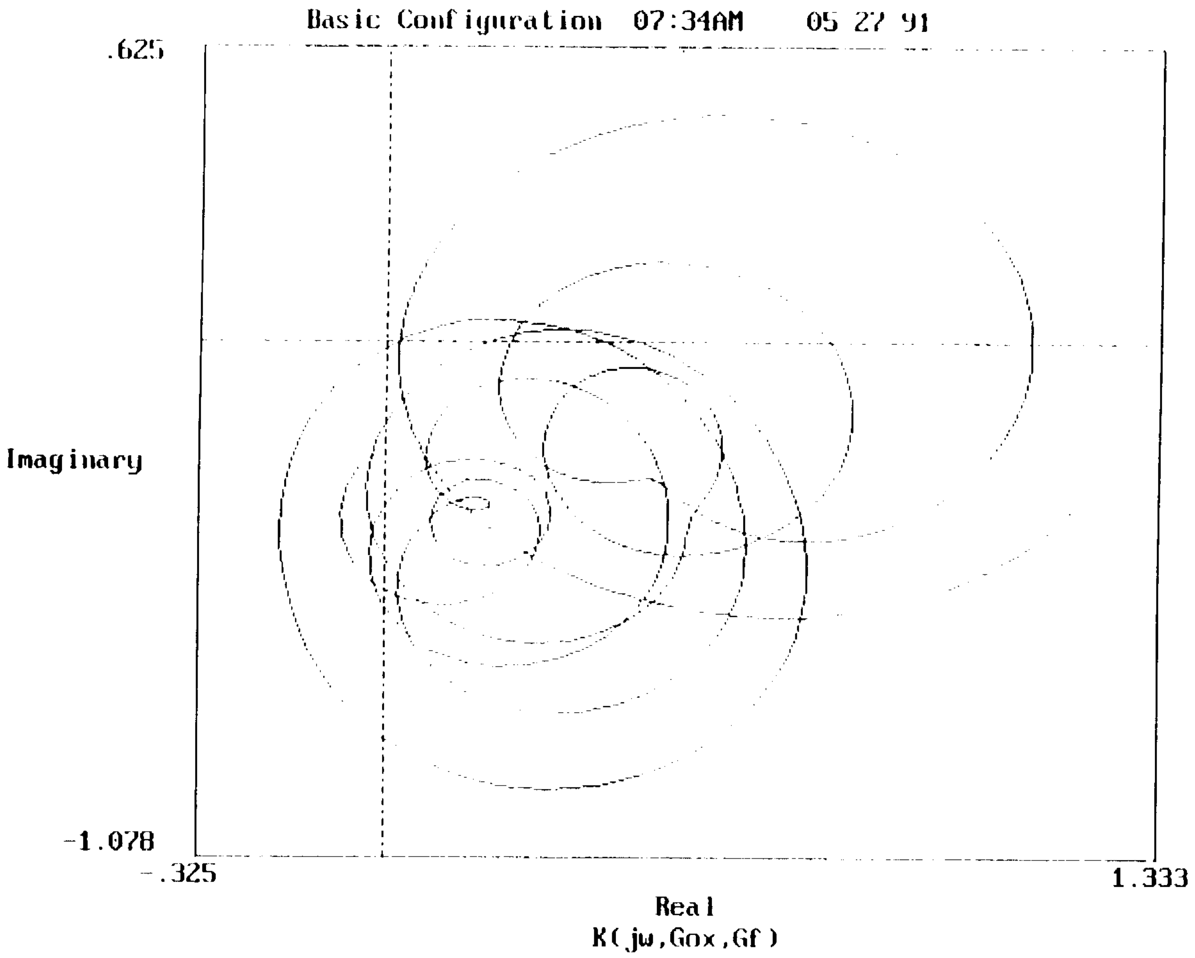

Basic Configuration 07:34AM 05 27 91
.625
Imaginary
-1.078
-.325
1.333
Real
K(jw,Gox,Gf)

Figure 16

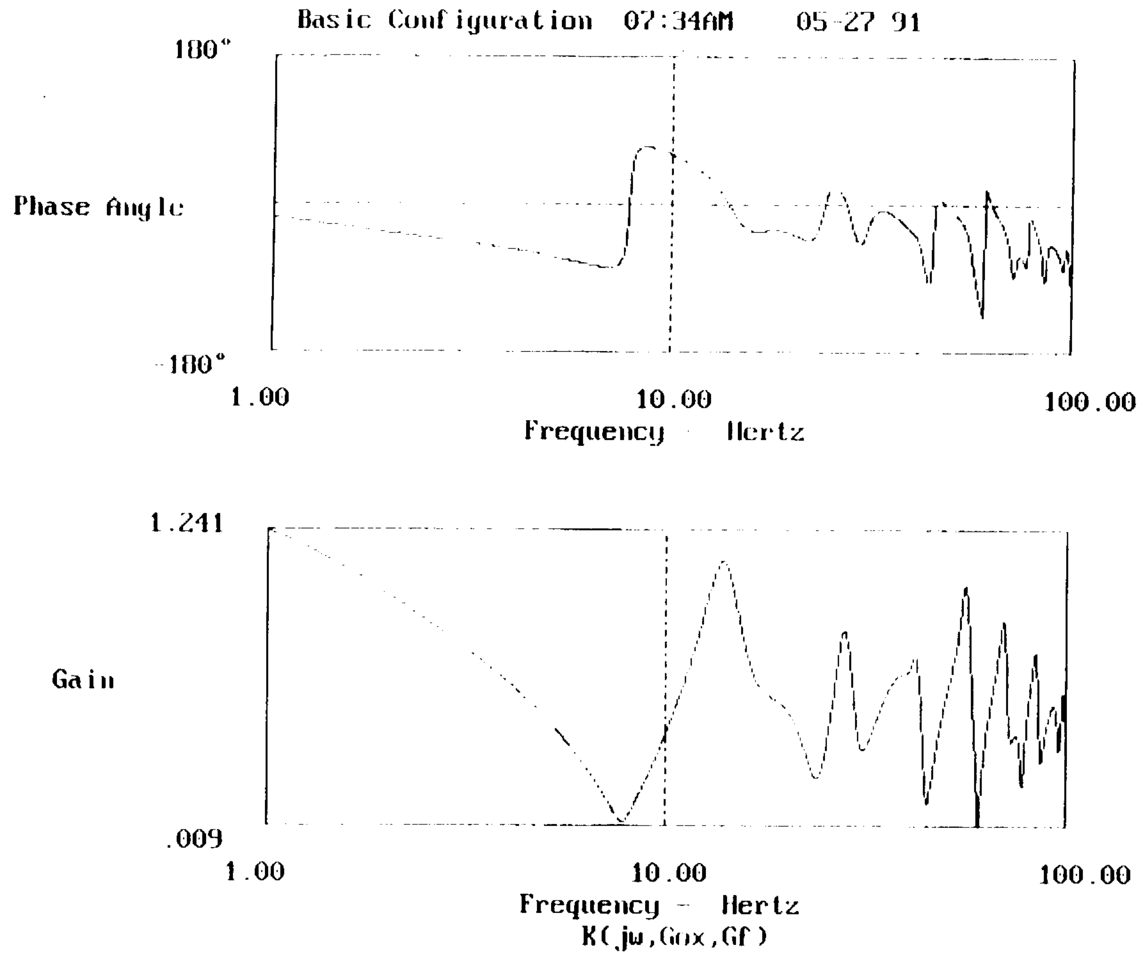

Basic Configuration 07:34AM 05-27 91
180°
Phase Angle
-180°
1.00
10.00
100.00
Frequency - Hertz
1.241
Gain
.009
1.00
10.00
100.00
Frequency - Hertz
K(jw,Gox,Gf)

Figure 17

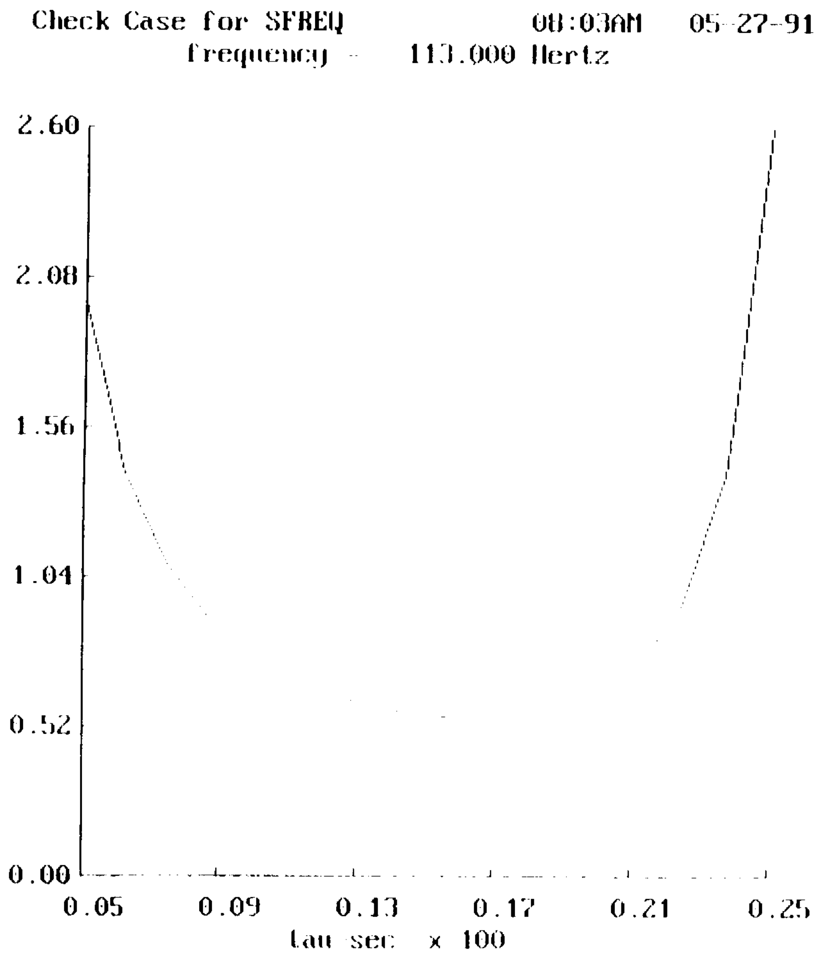

Check Case for SFREQ
08:03AM 05-27-91
frequency - 113.000 Hertz
2.60
2.08
1.56
1.04
0.52
0.00
0.05
0.09
0.13
0.17
0.21
0.25
tau sec x 100

Figure 18

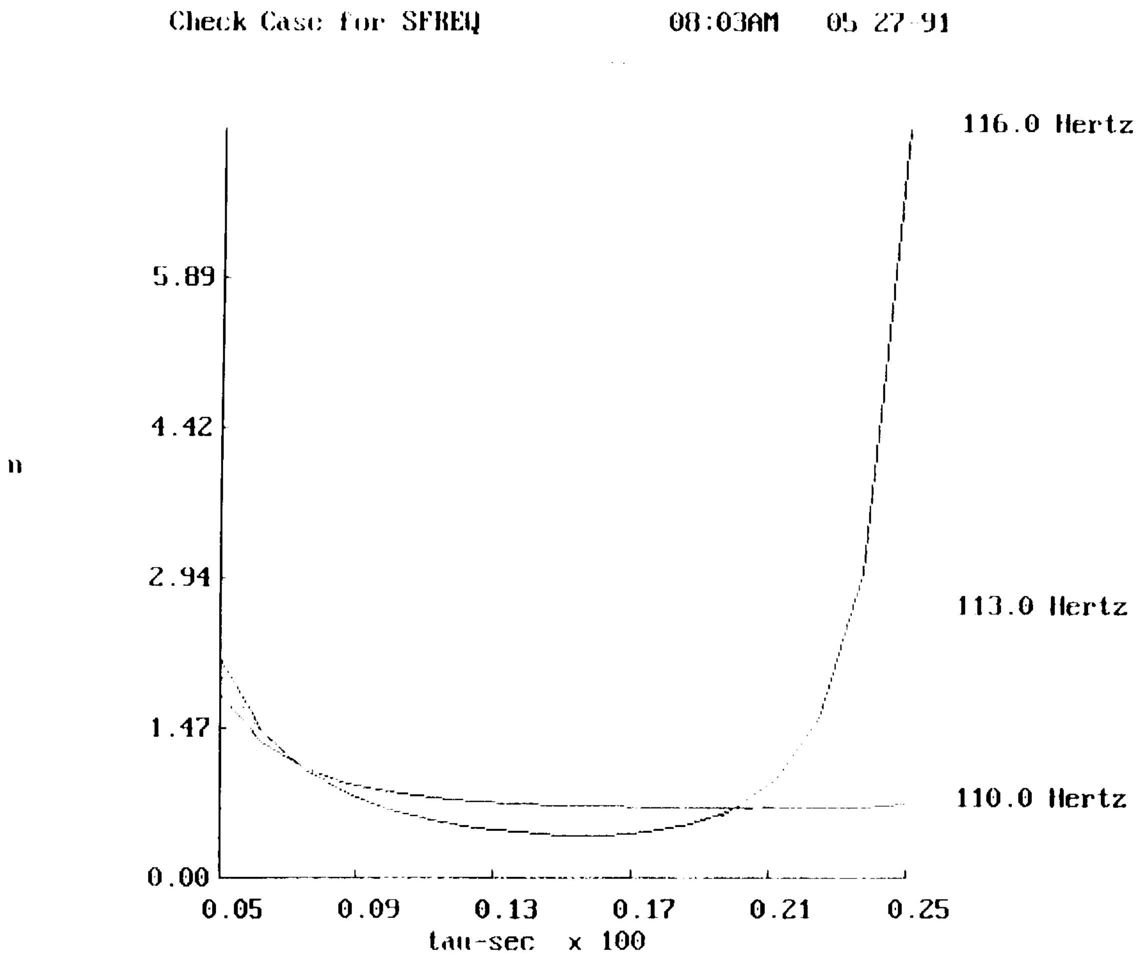

Check Case for SFREQ
08:03AM 05 27 91
116.0 Hertz
113.0 Hertz
110.0 Hertz
n
5.89
4.42
2.94
1.47
0.00
0.05
0.09
0.13
0.17
0.21
0.25
tau-sec x 100

Figure 19

Appendix A

Applications of High Frequency Code

FDORC

The high frequency code (FDORC) was applied to a couple of actual engine designs. The first engine to be studied is described in "Predicted Combustion Stability Characteristics for the TRW Advanced Booster Application Engine", C. W. Johnson and G. R. Nickerson, Software and Engineering Associates, Inc., SEA SN109, May 1990. The 750K engine and the 1st tangential - 1st longitudinal were used as a check-point.

Data for 750K Engine Analysis (LOX RP1)

Gamma	= 1.202
Temperature	= 6400° F
Pressure	= 660 psi
Chamber radius	= 2.375'
Chamber length	= 1.4885'
Throat radius	= 1.28'
Radius RC	= 1.67'
Radius RE	= 1.67'
Angle	= 30°
Speed of sound	= 2861 ft/sec

The results are summarized in the following table:

Item	Value	Source
n	0.3087	SEA SN109
tau	0.0007182	SEA SN109
frequency	1046 Hz	SEA SN109
acoustic frequency	1024 Hz	FDORC
n - neutral stab. for 1046 Hz	6.6062	FDORC
tau - neut. stab. for 1046 Hz	0.0001514	FDORC
n - neutral stab. for 1024 Hz	6.2223	FDORC
tau - neut. stab. for 1024 Hz	0.0001573	FDORC
frequency for n=0.3087, tau=0.0007182	845.3 Hz	FDORC
damping for n=0.3087, tau=0.0007182	2.3642	FDORC

note: in FDORC's notation, a positive value for damping means there is positive damping.

Data for the $n - \tau$ curve for this case was generated using FDORC. The $n - \tau$ curve and n, τ for the 750K engine are shown in Figure A-1. Results from SEA SN109 lie well below the neutral stability curve. Thus, the two analysises agree that the engine is stable in the 1st tangential - 1st longitudinal mode.

The code also was used to study a new engine proposal. The data for this engine is given in the following table.

Data for New Engine Analysis (LOX H_2)

Gamma	= 1.22
Temperature	= 6000° F
Pressure	= 360 psi
Chamber radius	= 23.21"
Chamber length	= 18"
Throat radius	= 16.4"
Radius RC	= 24.63"
Radius RE	= 24.63"
Angle	= 20°
Speed of sound	= 3676 ft/sec

Several modes of oscillation were run for this engine on the FDORC code. The location of the minimum points on the n - τ curves are given in the following table.

Mode of Oscillation			Minimum of n-τ Curve Occurs at	
radial	tangential	axial	n	τ (sec)
1	1	0	0.565	0.000789
2	1	0	0.497	0.000301
1	2	0	0.537	0.000500
2	2	0	0.505	0.000249
2	2	1	1.806	0.000950

The n - τ curve for the 1^{st} transverse mode (1,1,0) is shown in Figure A-2. The engine will be stable in this mode if n for the engine falls below the curve.

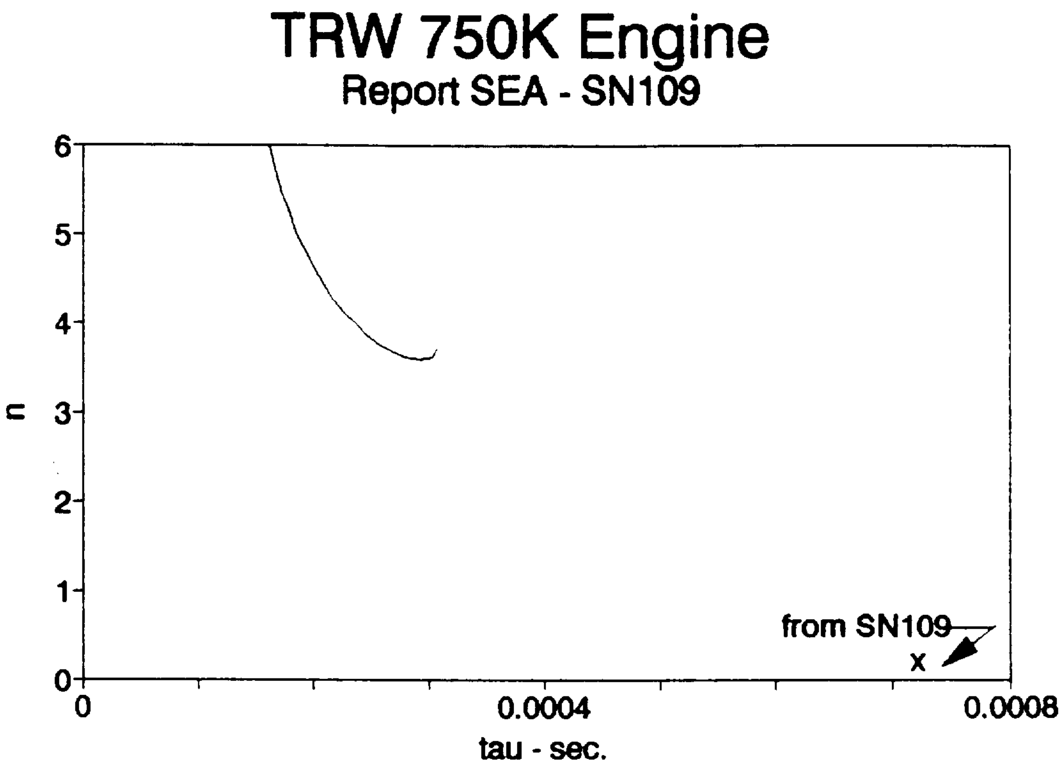

TRW 750K Engine
Report SEA - SN109
6
5
4
3
2
1
0
n
0
0.0004
0.0008
tau - sec.
from SN109
x

Figure A-1

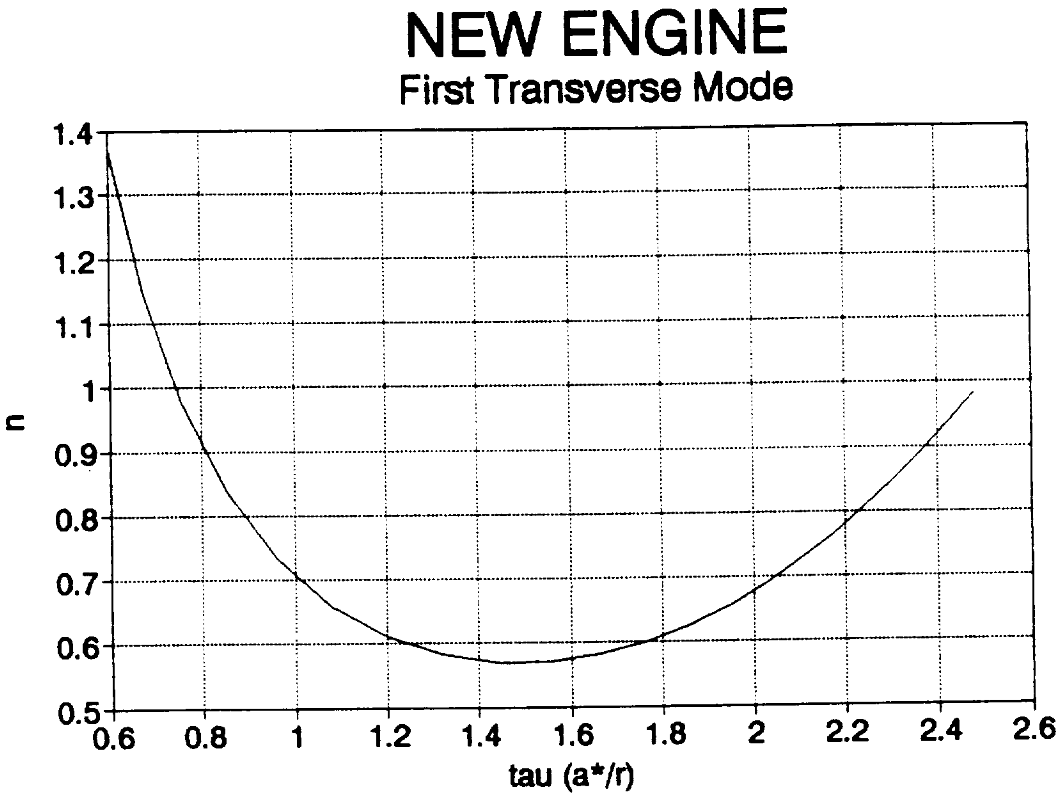

NEW ENGINE
First Transverse Mode
1.4
1.3
1.2
1.1
1
0.9
0.8
0.7
0.6
0.5
n
0.6
0.8
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
tau (a*/r)

Figure A-2

Appendix B

Listing of Feedline Program

ACCUM

```
C
C      PROGRAM ACCUM
C
C         Program to compute and plot admittance coefficients, pipe layout,
C                    and pressure transfer function
C
C               VARIABLE DIMENSION VERSION 06-27-91
C
C         This program will handle the following type elements
C
C               Straight pipes
C               Bends
C               Split pipes (into identical lines)
C               Inline accumulators
C               Tuned stub accumulators
C               Helmholtz resonators
C               Parallel resonators
C               Pumps
C
C
C                       Variables in Commons
C
C                             /ADMCOL/
C   ADMBAC           INTEGER*2  maximum value of admittance for plot
C   ADMLIN           INTEGER*2  line color of admittance plot
C
C                             /ARCCON/
C   XC               REAL*4     x coordinate of curve center
C   YC               REAL*4     y coordinate of curve center
C   RAD              REAL*4     radius of bend
C   ANG              REAL*4     angle of bend in radians
C   ANGLE            REAL*4     angle of bend in degrees
C
C                             /FACTOR/
C   SFAC             REAL*4     factor for frequency
C
C                             /FREQ/
C   S                COMPLEX*8  complex frequency
C   ZT(0:76)         COMPLEX*8  impedance looking toward tank
C   ZO(76)           REAL*4     characteristic impedance
C   ZG(76)           COMPLEX*8  impedance looking toward engine
C
C                             /INTVAL/
C   SECT             INTEGER*2  current pipe section type
C   SECTN(75)        INTEGER*2  pipe section type
C   SEGMN            INTEGER*2  number of pipe sections
C   NSEC(75)         INTEGER*2  no. of integration segments of a pipe section
C   NPTS             INTEGER*2  number of x points for plot
C   LOPEND           INTEGER*2  maximum number of iterations for split pipe
C   LOPOLD           INTEGER*2  previous maximum number of iterations
C
C                             /NOCOL/
```

```
C    MODE             INTEGER*2  graphics mode of monitor
C    MODET            INTEGER*2  text mode of monitor
C    NTROWS           INTEGER*2  number of text rows for graphics
C    NTCOLS           INTEGER*2  number of text columns for graphics
C    NPROWS           INTEGER*2  number of pixel rows for graphics
C    NPCOLS           INTEGER*2  number of pixel columns for graphics
C
C                               /PIPPXY/
C    X                REAL*4       x location of current centerline
C    XH               REAL*4       x location of current upper pipe
C    XL               REAL*4       x location of current lower pipe
C    Y                REAL*4       y location of current centerline
C    YH               REAL*4       y location of current upper pipe
C    YL               REAL*4       y location of current lower pipe
C    XMIN             REAL*4       minimum x value of piping layout
C    XMAX             REAL*4       maximum x value of piping layout
C    YMIN             REAL*4       minimum y value of piping layout
C    YMAX             REAL*4       maximum y value of piping layout
C    SINA             REAL*4       sine of current pipe direction
C    COSA             REAL*4       cosine of current pipe direction
C
C                               /RELVAL/
C    A                REAL*4       speed of sound in the fluid (ft/sec)
C    AREA(75)         REAL*4       area of pipe section (ft^2)
C    AREAB            REAL*4       area of current pipe section (ft^2)
C    CMAN             REAL*4       manifold capacitance
C    CTANK            REAL*4       tank capacitance
C    DENS             REAL*4       density of fluid (lbm/ft^3)
C    DIA(75)          REAL*4       diameter of pipe section (ft)
C    DIME             REAL*4       diameter of current pipe section (ft)
C    DPROR            REAL*4       pressure drop across orfices (lbf/ft^2)
C    L(75)            REAL*4       length of pipe section (ft)
C    PCHMB            REAL*4       chamber pressure (lbf/ft^2)
C    PIPE1(75)        REAL*4       first parameter of pipe description
C    PIPE2(75)        REAL*4       second parameter of pipe description
C    PIPE3(75)        REAL*4       third parameter of pipe description
C    PIPE4(75)        REAL*4       fourth parameter of pipe description
C    PIPE5(75)        REAL*4       fifth parameter of pipe description
C    TFLOW            REAL*4       total flow rate of engine (lbm/sec)
C    VALUE            REAL*4       used for passing different values
C    VOL              REAL*4       volume of tank (ft^3)
C    VOLMF            REAL*4       volume of manifold (ft^3)
C    PMRAT            REAL*4       chamber pressure/total mass flow
C    SPLIT            REAL*4       number of lines from pipe split
C    PCAP(75)         REAL*4       capacitance of pipe section
C    PIND(75)         REAL*4       inductance of pipe section
C    KMAN             REAL*4       bulk modulus of manifold (lbf/ft^2)
C    KTANK            REAL*4       bulk modulus of tank (lbf/ft^2)
C    LFLOW            REAL*4       flow rate through pipe (lbm/sec)
C
C                               /WCAOUT
C    NAMLIN           CHAR*24      name of file containing pipe description
```

```
C
C                                  /WCAPAS/
C    IFRST              INTEGER*2  flag for admittance plot
C
C                                  /WCATIT/
C    TITLE              CHAR*40    title for plots
C    TITLF              CHAR*20    title from pipe file
C    IHR                INTEGER*2  hour code run
C    IMIN               INTEGER*2  minute code run
C    AP                 CHAR*2     AM or PM
C    IYR                INTEGER*2  yesr code run
C    IMON               INTEGER*2  month code run
C    IDAY               INTEGER*2  day code run
C
C
C    PROGRAM ACCUM
C        Determines maximum array sizes
C
C                  Local Variables
C    I                  INTEGER*4  do loop index
C    IERR               INTEGER*2  error flag for ALLOCATE
C    IXMAX              INTEGER*4  maximum number of frequencies
C    IYMAX              INTEGER*4  maximum number of points along piping
C    X(IXMAX,IYMAX)     REAL*4     frequency array for plotting
C    XF(IXMAX)          REAL*4     frequency array
C    Y(IXMAX,IYMAX)     REAL*4     location array for plotting
C    YF(IYMAX)          REAL*4     location array
C    Z(IXMAX,IYMAX)     REAL*4     gain array for plotting
C    ZF(IXMAX,IYMAX)    REAL*4     gain array
C
C
C    SUBROUTINE MAINP(X,Y,Z,XF,YF,ZF,IXMAX,IYMAX)
C        Logic portion of code
C
C    Commons FACTOR    FREQ    INTVAL  RELVAL  WCAOUT  WCATIT
C                      Variables in Argument List
C    IXMAX              INTEGER*4  maximum number of frequencies
C    IYMAX              INTEGER*4  maximum number of points along piping
C    X(IXMAX,IYMAX)     REAL*4     frequency array for plotting
C    XF(IXMAX)          REAL*4     frequency array
C    Y(IXMAX,IYMAX)     REAL*4     location array for plotting
C    YF(IYMAX)          REAL*4     location array
C    Z(IXMAX,IYMAX)     REAL*4     gain array for plotting
C    ZF(IXMAX,IYMAX)    REAL*4     gain array
C                      Local Variables
C    ADMMAX             REAL*4     maximum value of admittance for plot
C    AM                 CHAR*2     'AM'
C    ANS                CHAR*1     response to question
C    AVGK               REAL*4     average bulk modulus (lbf/ft^2)
C    CAPM               COMPLEX*8  intermediate variable
C    CAPN               COMPLEX*8  intermediate variable
C    CFAC               COMPLEX*8  intermediate variable
```

```
C    ERRP              REAL*4      error in gain calculation
C    G(0:76)           COMPLEX*8   admittance looking toward tank
C    GRAV              REAL*4      gravitational constant (lbm-ft/lbf-sec^2)
C    G1                COMPLEX*8   admittance starting at G(0)+1
C    HFREQ             REAL*4      maximum frequency requested
C    I                 INTEGER*2   do loop index
C    IOPEN             INTEGER*2   flag indicating if SURF.ERR is open
C    IPLT              INTEGER*2   flag indicating when admittance is plotted
C    ISEC              INTEGER*2   second code run
C    ISIZ              INTEGER*2   counter for number of integration segments
C    I100              INTEGER*2   hundredth of second code run
C    K                 INTEGER*2   do loop index
C    KLOOP             INTEGER*2   do loop index
C    LFREQ             REAL*4      minimum frequency requested
C    MAG               REAL*4      magnitude of G at orfice
C    MAG1              REAL*4      magnitude of G1 at orfice
C    NAMFUL            CHAR*24     name of fuel file (if used)
C    NAMLOX            CHAR*24     name of lox file (if used)
C    PI                REAL*4      mathematical constant
C    PM                CHAR*2      'PM'
C    PTS               INTEGER*2   number of frequencies
C    RHS               COMPLEX*8   intermediate variable
C    RSPON             INTEGER*2   flag to MODIFY subroutine
C    SSIZE             REAL*4      frequency step size
C    TL                REAL*4      length/speed of sound
C    TLT               REAL*4      total lenthe of piping
C    W                 REAL*4      oscillatory part of frequency
C    WN                REAL*4      normalized W
C    WVAL              REAL*4      maximum gain
C    ZGEFF             COMPLEX*8   effective impedance for calculations
C    ZOEFF             COMPLEX*8   effective ZO for calculations
C    ZOR               REAL*4      intermediate variable
C    ZTOP              REAL*4      intermediate variable
C
C
C    SUBROUTINE ADMGRAPH(LFREQ,HFREQ,ADMMAX)
C        Plots admittance curve
C
C    Commons FACTOR    NOCOL   WCATIT
C                      Variables in Argument List
C    ADMMAX             REAL*4     maximum value of admittance for plot
C    HFREQ              REAL*4     maximum frequency requested
C    LFREQ              REAL*4     minimum frequency requested
C                      Local Variables
C    XMAJ               REAL*4     distance between tick marks on x axis
C    XMAX               REAL*4     maximum value of x
C    XMIN               REAL*4     mimimum value of x
C    YMAJ               REAL*4     distance between tick marks on y axis
C    YMAX               REAL*4     maximum value of y
C    YMIN               REAL*4     mimimum value of y
C
C
```

```
C    SUBROUTINE ALLPT(X,Y,PTS)
C        Supervises plot of admittance after calculations
C
C                         Variables in Argument List
C    PTS                   INTEGER*2  number of frequencies
C    X(PTS)                REAL*4     frequency array
C    Y(PTS)                REAL*4     admittance array
C                         Local Variables
C    ADMMAX                REAL*4     maximum value of admittance for plot
C    I                     INTEGER*2  do loop index
C
C
C    SUBROUTINE BENDS(PIPE1,PIPE2,PIPE3,PIPE4,VALUE,DIME)
C        Computes effective straight pipe for bend
C
C                         Variables in Argument List
C    DIME                  REAL*4     effective diameter (ft)
C    PIPE1                 REAL*4     radius of bend (ft)
C    PIPE2                 REAL*4     angle of bend (degrees)
C    PIPE3                 REAL*4     diameter of bend (ft)
C    PIPE4                 REAL*4     length of end straight segments (ft)
C    VALUE                 REAL*4     effective length (ft)
C                         Local Variables
C    AREAB                 REAL*4     effective area of bend
C    ARBND                 REAL*4     area of bend
C    BENDR                 REAL*4     bend angle in radians
C    GAMMA                 REAL*4     intermediate variable
C    INERT                 REAL*4     intermediate variable
C    INRAD                 REAL*4     inside radius of bend
C    LBND                  REAL*4     intermediate variable
C    LPRME                 REAL*4     intermediate variable
C    NEWLN                 REAL*4     intermediate variable
C    OTRAD                 REAL*4     outside radius of bend
C    RATIO                 REAL*4     intermediate variable
C    X                     REAL*4     intermediate variable
C    Y                     REAL*4     intermediate variable
C
C
C    SUBROUTINE BNSECT(J,ITYPE,POINT,PIPE1,PIPE2,PIPE3,PIPE4)
C        Computes plot coordinates for a bend
C
C    Commons ARCCON   PIPPXY
C                         Variables in Argument List
C    ITYPE(200)            INTEGER*2  type plot element
C    J                     INTEGER*2  pointer to element
C    PIPE1                 REAL*4     first parameter of pipe description
C    PIPE2                 REAL*4     second parameter of pipe description
C    PIPE3                 REAL*4     third parameter of pipe description
C    PIPE4                 REAL*4     fourth parameter of pipe description
C    POINT(8,200)          REAL*4     description of plot element
C                         Local Variables
C    DIA                   REAL*4     intermediate variable
```

```
C   HOLD            REAL*4     intermediate variable
C   RANG            REAL*4     intermediate variable
C   SLENTH          REAL*4     intermediate variable
C   X0              REAL*4     intermediate variable
C   X1              REAL*4     intermediate variable
C   X2              REAL*4     intermediate variable
C   X3              REAL*4     intermediate variable
C   Y0              REAL*4     intermediate variable
C   Y1              REAL*4     intermediate variable
C   Y2              REAL*4     intermediate variable
C   Y3              REAL*4     intermediate variable
C
C
C   COMPLEX FUNCTION CCOSH(S)
C       Evaluates the complex hyperbolic cosine
C
C                  Variables in Argument List
C   S               COMPLEX*8  complex frequency
C                  Local Variables
C   COSHI           REAL*4     intermediate variable
C   COSHR           REAL*4     intermediate variable
C   LAMDA           REAL*4     real part of complex frequency
C   MU              REAL*4     imaginary part of complex frequency
C
C
C   COMPLEX FUNCTION CSINH(S)
C       Evaluates the complex hyperbolic sine
C
C                  Variables in Argument List
C   S               COMPLEX*8  complex frequency
C                  Local Variables
C   LAMDA           REAL*4     intermediate variable
C   MU              REAL*4     intermediate variable
C   SINHI           REAL*4     real part of complex frequency
C   SINHR           REAL*4     imaginary part of complex frequency
C
C
C   COMPLEX FUNCTION CTANH(S)
C       Evaluates the complex hyperbolic tangent
C
C                  Variables in Argument List
C   S               COMPLEX*8  complex frequency
C
C
C   SUBROUTINE ENDPLT
C       Closes plot routines
C
C   Commons NOCOL   WCAPAS
C                  Local Variables
C   IEXTEN          INTEGER*2  extension of key hit
C   IKEY            INTEGER*2  code of key hit
C
```

```
C
C   SUBROUTINE FREQRS(YF,ZF,K,IXMAX,IYMAX,KLOOP,ERRP,WVAL)
C       Computes pressure transfer function
C
C   Commons FREQ    INTVAL  RELVAL
C                   Variables in Argument List
C   ERRP             REAL*4     error in gain calculation
C   IXMAX            INTEGER*4  maximum number of frequencies
C   IYMAX            INTEGER*4  maximum number of points along piping
C   K                INTEGER*2  frequency pointer
C   KLOOP            INTEGER*2  loop pointer
C   WVAL             REAL*4     maximum gain
C   YF(IYMAX)        REAL*4     location array
C   ZF(IXMAX,IYMAX)  REAL*4     gain array
C                   Local Variables
C   BOTTOM           COMPLEX*8  intermediate variable
C   CAPM             COMPLEX*8  intermediate variable
C   CAPN             COMPLEX*8  intermediate variable
C   DX               REAL*4     x increment
C   ERRN             REAL*4     local error
C   I                INTEGER*2  do loop index
C   J                INTEGER*2  do loop index
C   LITTLN           COMPLEX*8  intermediate variable
C   LSEC             INTEGER*2  number of segments of pipe section
C   M                INTEGER*2  locatioon pointer
C   PRAT             COMPLEX*8  pressure ratio
C   PRATN            REAL*4     absolute value of pressure ratio
C   PRATO(2,75)      REAL*4     previous pressure ratio
C   SUMX             REAL*4     distance from orfice
C   TOP              COMPLEX*8  intermediate variable
C   X                REAL*4     distance along pipe section
C   ZFAC             COMPLEX*8  intermediate variable
C
C
C   SUBROUTINE GINERT(BEND,X,Y)
C       Evaluates curve fit of inertance of bends
C
C                   Variables in Argument List
C   BEND             REAL*4     angle of bend (degrees)
C   X                REAL*4     ratio of inner to outer radius
C   Y                REAL*4     inertance
C                   Local Variables
C   A                REAL*4     intermediate variable
C   B(3)             REAL*4     coefficient array for inertance fit
C
C
C   SUBROUTINE HHSECT(J,ITYPE,POINT,LEN,DIA,VOL)
C       Computes plot coordinates for Helmholtz resonator
C
C   Common  PIPPXY
C                   Variables in Argument List
C   DIA              REAL*4     diameter of opening (ft)
```

```
C   ITYPE(200)       INTEGER*2  type plot element
C   J                INTEGER*2  pointer to element
C   LEN              REAL*4     length of opening (ft)
C   POINT(8,200)     REAL*4     description of plot element
C   VOL              REAL*4     volume of reservoir (ft^3)
C                   Local Variables
C   COSOLD           REAL*4     intermediate variable
C   DIAM             REAL*4     intermediate variable
C   SIDE             REAL*4     intermediate variable
C   SINOLD           REAL*4     intermediate variable
C   XC               REAL*4     intermediate variable
C   XHOLD            REAL*4     intermediate variable
C   XLOLD            REAL*4     intermediate variable
C   XOLD             REAL*4     intermediate variable
C   YC               REAL*4     intermediate variable
C   YHOLD            REAL*4     intermediate variable
C   YLOLD            REAL*4     intermediate variable
C   YOLD             REAL*4     intermediate variable
C
C
C   SUBROUTINE LOWERW(LFREQ,HFREQ,ADMMAX)
C       Sets up lower plotting window
C
C   Commons ADMCOL  NOCOL
C                   Variables in Argument List
C   ADMMAX           REAL*4     maximum value of admittance for plot
C   HFREQ            REAL*4     maximum frequency requested
C   LFREQ            REAL*4     minimum frequency requested
C                   Local Variables
C   ASPECT           REAL*4     aspect ratio of monitor screen
C   IOPT             INTEGER*2  intermediate variable
C   JCOL1            INTEGER*2  starting column for admittance window
C   JCOL2            INTEGER*2  ending column for afdmittance window
C   JROW1            INTEGER*2  starting row for admittance window
C   JROW2            INTEGER*2  ending row for admittance window
C   XLEN             REAL*4     intermediate variable
C   XMAX             REAL*4     maximum x value for admittance plot
C   XMIN             REAL*4     minimum x value for admittance plot
C   XORG             REAL*4     x origin for admittance plot
C   YLEN             REAL*4     intermediate variable
C   YMAX             REAL*4     maximum y value for admittance plot
C   YMIN             REAL*4     minimum y value for admittance plot
C   YOVERX           REAL*4     intermediate variable
C   YORG             REAL*4     y origin for admittance plot
C
C
C   SUBROUTINE MODIFY(RSPON)
C       Allows modifications to input data
C
C   Commons INTVAL  RELVAL  WCAOUT  WCATIT
C                   Variables in Argument List
C   RSPON            INTEGER*2  flag for path to be taken
```

```
C                     Local Variables
C   ANS                CHAR*1     response to question
C   AVGK               REAL*4     average bulk modulus (lbf/ft^2)
C   GRAV               REAL*4     gravitational constant (lbm-ft/lbf-sec^2)
C   I                  INTEGER*2  pointer
C   II                 INTEGER*2  do loop index
C   III                INTEGER*2  do loop index
C   ICHG               INTEGER*2  change flag
C   ISEGMN             INTEGER*2  intermediate variable
C   NAME               CHAR*8     variable name
C   PI                 REAL*4     mathematical constant
C   VARL(9)            CHAR*8     array of variable names (lower case)
C   VARU(9)            CHAR*8     array of variable names (upper case)
C   VARVAL(9)          CHAR*8     array of variable names for printout
C
C
C   SUBROUTINE NEXPT(WN,MAG1)
C       Supervises plot of admittance while computing
C
C   Common  WCAPAS
C                     Variables in Argument List
C   MAG1               REAL*4     admittance
C   WN                 REAL*4     frequency
C                     Local Variables
C   X(2)               REAL*4     print line (frequency)
C   Y(2)               REAL*4     print line (admittance)
C
C
C   SUBROUTINE PIPPLOT(SEGMN,SECTN,PIPE1,PIPE2,PIPE3,PIPE4)
C       Supervises plot of piping layout
C
C   Commons ARCCON  PIPPXY
C                     Variables in Argument List
C   PIPE1(75)          REAL*4     first parameter of pipe description
C   PIPE2(75)          REAL*4     second parameter of pipe description
C   PIPE3(75)          REAL*4     third parameter of pipe description
C   PIPE4(75)          REAL*4     fourth parameter of pipe description
C   SECTN(75)          INTEGER*2  segment types
C   SEGMN              INTEGER*2  number of pipe segments
C                     Local Variables
C   I                  INTEGER*2  do loop index
C   ITYPE(200)         INTEGER*2  type plot element
C   J                  INTEGER*2  pointer to element
C   POINT(8,200)       REAL*4     description of plot element
C   XP(2)              REAL*4     x plot array
C   XRANGE             REAL*4     range of x values
C   X0                 REAL*4     intermediate variable
C   X1                 REAL*4     intermediate variable
C   X2                 REAL*4     intermediate variable
C   X3                 REAL*4     intermediate variable
C   YP(2)              REAL*4     y plot array
C   YRANGE             REAL*4     range of y values
```

```
C   Y0               REAL*4     intermediate variable
C   Y1               REAL*4     intermediate variable
C   Y2               REAL*4     intermediate variable
C   Y3               REAL*4     intermediate variable
C
C
C   SUBROUTINE PLOTSU(X,Y,Z,XF,YF,ZF,JPTS,IPTS,IXMAX,IYMAX)
C       Supervises the surface plot
C
C   Commons FACTOR  WCATIT
C                   Variables in Argument List
C   IPTS             INTEGER*2  actual number of frequencies
C   IXMAX            INTEGER*4  maximum number of frequencies
C   IYMAX            INTEGER*4  maximum number of points along piping
C   JPTS             INTEGER*2  actual number of points along pipe
C   X(IPTS,JPTS)     REAL*4     frequency array for plotting
C   XF(IXMAX)        REAL*4     frequency array
C   Y(IPTS,JPTS)     REAL*4     location array for plotting
C   YF(IYMAX)        REAL*4     location array
C   Z(IPTS,JPTS)     REAL*4     gain array for plotting
C   ZF(IXMAX,IYMAX)  REAL*4     gain array
C                   Local Variables
C   ANS              CHAR*1     response to question
C   ASPECT           REAL*4     aspect ratio of monitor
C   I                INTEGER*2  do loop index
C   IBOARD           INTEGER*2  type graphics board installed
C   ICOLR            INTEGER*2  background color
C   IEXTEN           INTEGER*2  extension of key hit
C   IFIL             INTEGER*2  fill color
C   IGO              INTEGER*2  flag for changes
C   IKEY             INTEGER*2  code of key hit
C   ILIN             INTEGER*2  line color
C   IWIRE            INTEGER*2  flag for wire-frame or filled
C   IWR              INTEGER*2  temporary flag for wire-frame or filled
C   IWRK1(640)       INTEGER*2  work array for plot routine
C   IWRK2(640)       INTEGER*2  work array for plot routine
C   J                INTEGER*2  do loop index
C   LEGEND           CHAR*45    legend for CGA monitor
C   LEGENDH          CHAR*58    legend for EGA or VGA monitor (Hertz)
C   LEGENDR          CHAR*58    legend for EGA or VGA monitor (rad/sec)
C   MODE             INTEGER*2  graphics mode
C   MODET            INTEGER*2  text mode
C   NCOLT            INTEGER*2  number of columns in text mode
C   P                REAL*4     phi rotation angle (degrees)
C   T                REAL*4     theta rotation angle (degrees)
C   XFAC             REAL*4     intermediate variable
C   XINV             REAL*4     intermediate variable
C   XLEN             REAL*4     length of x axis
C   XMAJ             REAL*4     distance between tick marks on x axis
C   XMAX             REAL*4     maximum value for x axis
C   XMIN             REAL*4     minimum value for x axis
C   XYZLEN           REAL*4     intermediate variable
```

```
C   YFAC               REAL*4     intermediate variable
C   YINV               REAL*4     intermediate variable
C   YLEN               REAL*4     length of y axis
C   YMAJ               REAL*4     distance between tick marks on y axis
C   YMAX               REAL*4     maximum value for y axis
C   YMIN               REAL*4     minimum value for y axis
C   ZFAC               REAL*4     intermediate variable
C   ZINV               REAL*4     intermediate variable
C   ZLEN               REAL*4     length of z axis
C   ZMAJ               REAL*4     distance between tick marks on z axis
C   ZMAX               REAL*4     maximum value for z axis
C   ZMIN               REAL*4     minimum value for z axis
C
C
C   SUBROUTINE PLSECT(J,ITYPE,POINT,LEN,DIA,VOL)
C       Computes plot coordinates for parallel resonator
C
C   Commons ARCCON  PIPPXY
C                   Variables in Argument List
C   DIA              REAL*4     diameter of parallel segment (ft)
C   ITYPE(200)       INTEGER*2  type plot element
C   J                INTEGER*2  pointer to element
C   LEN              REAL*4     length of parallel segment (ft)
C   POINT(8,200)     REAL*4     description of plot element
C   VOL              REAL*4     volume of bypassed segment (ft^3)
C                   Local Variables
C   ANGOLD           REAL*4     intermediate variable
C   ANGSAV           REAL*4     intermediate variable
C   COSOLD           REAL*4     intermediate variable
C   DIAM             REAL*4     intermediate variable
C   PDIA             REAL*4     intermediate variable
C   PLEN             REAL*4     intermediate variable
C   RADIUS           REAL*4     intermediate variable
C   SIDE             REAL*4     intermediate variable
C   SINOLD           REAL*4     intermediate variable
C   TURN             REAL*4     intermediate variable
C   XHC              REAL*4     intermediate variable
C   XHOLD            REAL*4     intermediate variable
C   XHSAV            REAL*4     intermediate variable
C   XLC              REAL*4     intermediate variable
C   XLOLD            REAL*4     intermediate variable
C   XLSAV            REAL*4     intermediate variable
C   XOLD             REAL*4     intermediate variable
C   XSAV             REAL*4     intermediate variable
C   YHC              REAL*4     intermediate variable
C   YHOLD            REAL*4     intermediate variable
C   YHSAV            REAL*4     intermediate variable
C   YLC              REAL*4     intermediate variable
C   YLOLD            REAL*4     intermediate variable
C   YLSAV            REAL*4     intermediate variable
C   YOLD             REAL*4     intermediate variable
C   YSAV             REAL*4     intermediate variable
```

```
C
C
C   SUBROUTINE PLTCON(X,Y,Z,XF,YF,ZF,JPTS,IPTS,IXMAX,IYMAX)
C       Supervises plot of contour plot
C
C   Commons FACTOR  WCATIT
C                   Variables in Argument List
C   IPTS             INTEGER*2  actual number of frequencies
C   IXMAX            INTEGER*4  maximum number of frequencies
C   IYMAX            INTEGER*4  maximum number of points along piping
C   JPTS             INTEGER*2  actual number of points along pipe
C   X(IPTS)          REAL*4     frequency array for plotting
C   XF(IXMAX)        REAL*4     frequency array
C   Y(JPTS)          REAL*4     location array for plotting
C   YF(IYMAX)        REAL*4     location array
C   Z(IPTS,JPTS)     REAL*4     gain array for plotting
C   ZF(IXMAX,IYMAX)  REAL*4     gain array
C                   Local Variables
C   ANS              REAL*4     response to question
C   ASPECT           REAL*4     aspect ratio of monitor
C   CONS(10)         REAL*4     array for values of contour lines
C   I                INTEGER*2  do loop index
C   IBOARD           INTEGER*2  type graphics board installed
C   ICOLR            INTEGER*2  background color
C   IDEF             INTEGER*2  flag for plot routine
C   IEXTEN           INTEGER*2  extension of key hit
C   IFIL             INTEGER*2  fill color
C   IKEY             INTEGER*2  code of key hit
C   ILIN             INTEGER*2  line color
C   IOPT             INTEGER*2  flag for plot routine
C   J                INTEGER*2  do loop index
C   JCOL1            INTEGER*2  starting column for contour plot window
C   JCOL2            INTEGER*2  ending column for contour plot window
C   JROW1            INTEGER*2  starting row for contour plot window
C   JROW2            INTEGER*2  ending row for contour plot window
C   LABL(10)         INTEGER*2  flags for labeling contours
C   MODE             INTEGER*2  graphics mode
C   MODET            INTEGER*2  text mode
C   NCOLT            INTEGER*2  number of columns in text mode
C   XMAJ             REAL*4     distance between tick marks on x axis
C   XMAX             REAL*4     maximum value for x axis
C   XMIN             REAL*4     minimum value for x axis
C   XORG             REAL*4     origin of x axis
C   YMAJ             REAL*4     distance between tick marks on y axis
C   YMAX             REAL*4     maximum value for y axis
C   YMIN             REAL*4     minimum value for y axis
C   YORG             REAL*4     origin of y axis
C   YOVERX           REAL*4     intermediate variable
C   ZLEN             REAL*4     intermediate variable
C   ZMAX             REAL*4     maximum value for z
C   ZMIN             REAL*4     minimum value for z
C
```

```
C
C   SUBROUTINE SETPLT
C       Sets up the plot environment
C
C   Commons ADMCOL  NOCOL   WCAPAS
C                   Local Variables
C   ANS              CHAR*1      response to question
C   IBOARD           INTEGER*2   type graphics board installed
C   ITIM             INTEGER*2   flag for initialization
C   NCOLT            INTEGER*2   number of columns in text mode
C
C
C   SUBROUTINE STSECT(J,ITYPE,POINT,LEN,DIA)
C       Computes plot coordinates for a straight section
C
C   Common  PIPPXY
C                   Variables in Argument List
C   DIA              REAL*4      diameter of segment (ft)
C   ITYPE(200)       INTEGER*2   type plot element
C   J                INTEGER*2   pointer to element
C   LEN              REAL*4      length of segment (ft)
C   POINT(8,200)     REAL*4      description of plot element
C
C
C   SUBROUTINE TSSECT(J,ITYPE,POINT,LEN,DIA)
C       Computes plot coordinates for a tuned stub
C
C   Common  PIPPXY
C                   Variables in Argument List
C   DIA              REAL*4      diameter of tuned stub (ft)
C   ITYPE(200)       INTEGER*2   type plot element
C   J                INTEGER*2   pointer to element
C   LEN              REAL*4      length of tuned stub
C   POINT(8,200)     REAL*4      description of plot element
C                   Local Variables
C   DIAM             REAL*4      intermediate variable
C
C
C   SUBROUTINE UPPERW(X0,Y0,X1,Y1)
C       Sets up upper plotting window
C
C   Commons ADMCOL  NOCOL
C                   Variables in Argument List
C   X0               REAL*4      minimum value of x for piping layout window
C   X1               REAL*4      maximum value of x for piping layout window
C   Y0               REAL*4      minimum value of y for piping layout window
C   Y1               REAL*4      maximum value of y for piping layout window
C                   Local Variables
C   ASPECT           REAL*4      aspect ratio of monitor
C   CHANGE           REAL*4      intermediate variable
C   IOPT             INTEGER*2   flag for plot routine
C   JCOL1            INTEGER*2   starting column for pipe layout plot window
```

```
C   JCOL2            INTEGER*2  ending column for pipe layout plot window
C   JROW1            INTEGER*2  starting row for pipe layout plot window
C   JROW2            INTEGER*2  ending row for pipe layout plot window
C   XMAX             REAL*4     maximum value for x axis
C   XMIN             REAL*4     minimum value for x axis
C   XORG             REAL*4     origin of x axis
C   YMAX             REAL*4     maximum value for x axis
C   YMAXO            REAL*4     intermediate variable
C   YMIN             REAL*4     minimum value for x axis
C   YORG             REAL*4     origin of x axis
C   YOVERX           REAL*4     intermediate variable
C
C
C   SUBROUTINE WINDOW(MODE,XSCALE,XST,XFIN,YST,YFIN,ZST,ZFIN)
C       Sets up window for surface plot
C
C                   Variables in Argument List
C   MODE             INTEGER*2  graphics mode
C   XFIN             REAL*4     final x value
C   XSCALE           REAL*4     aspect ratio of monitor
C   XST              REAL*4     starting x value
C   YFIN             REAL*4     final y value
C   YST              REAL*4     starting y value
C   ZFIN             REAL*4     final z value
C   ZST              REAL*4     starting z value
C                   Local Variables
C   ASPECT           REAL*4     aspect ratio of monitor
C   IOPT             INTEGER*2  flag for plot routine
C   JCOL1            INTEGER*2  starting column for surface plot window
C   JCOL2            INTEGER*2  ending column for surface plot window
C   JROW1            INTEGER*2  starting row for surface plot window
C   JROW2            INTEGER*2  ending row for surface plot window
C   XMAX             REAL*4     maximum value for x axis
C   XMIN             REAL*4     minimum value for x axis
C   XORG             REAL*4     origin of x axis
C   YMAX             REAL*4     maximum value for y axis
C   YMIN             REAL*4     minimum value for y axis
C   YORG             REAL*4     origin of y axis
C   YOVERX           REAL*4     intermediate variable
C
C
C   FUNCTION XFUN(T)
C       Parametric function for plotting of bends
C
C   Common  ARCCON
C                   Variables in Argument List
C   T                REAL*4     angle in radians
C
C
C   FUNCTION YFUN(T)
C       Parametric function for plotting of bends
C
```

```
C   Common  ARCCON
C                   Variables in Argument List
C   T                REAL*4      angle in radians
C
C
C   SUBROUTINE ZREAD(NAME,VALUE)
C       Reads input for input modification
C
C                   Variables in Argument List
C   NAME(8)          CHAR*1      name of input variable
C   VALUE            REAL*4      value of input variable
C                   Local Variables
C   BLK              CHAR*1      ' '
C   CARD(80)         CHAR*1      card image
C   CEND(3)          CHAR*1      'E','N','D'
C   COMMA            CHAR*1      ','
C   CTIT(5)          CHAR*1      'T','I','T','L','E'
C   DCARD            CHAR*80     card image
C   E                CHAR*1      'E'
C   FRACT            REAL*4      fractional part of number
C   I                INTEGER*2   do loop index
C   ICOUNT           INTEGER*2   position counter
C   ID               INTEGER*2   position counter
C   II               INTEGER*2   position counter
C   J                INTEGER*2   do loop index
C   JJ               INTEGER*2   position counter
C   LE               CHAR*1      'e'
C   LEND(3)          CHAR*1      'e','n','d'
C   LTIT(5)          CHAR*1      't','i','t','l','e'
C   MINUS            CHAR*1      '-'
C   NUMBER(10)       CHAR*1      '0','1','2','3','4','5','6','7','8','9'
C   PERIOD           CHAR*1      '.'
C   PLUS             CHAR*1      '+'
C   POUND            CHAR*1      '#'
C   QUEST            CHAR*1      '?'
C   SIGN             REAL*4      sign of number or exponent
C   WHOLE            REAL*4      WHOLE PART OF NUMBER
C
      INTERFACE TO SUBROUTINE
     1            clearscreen[FAR,C,ALIAS:"__clearscreen"] (area)
      INTEGER*2 area
      END
      INTEGER*4 IXMAX,IYMAX,I
      REAL X[ALLOCATABLE](:,:),Y[ALLOCATABLE](:,:),Z[ALLOCATABLE](:,:),
     *    XF[ALLOCATABLE](:),YF[ALLOCATABLE](:),ZF[ALLOCATABLE](:,:)
      EXTERNAL CLEARSCREEN
      DO 20 I=150,1,-1
       IXMAX=I
       IYMAX=I
       IERR=0
       ALLOCATE(X(IXMAX,IYMAX),Y(IXMAX,IYMAX),Z(IXMAX,IYMAX),STAT=IERR)
       ALLOCATE(XF(IXMAX),YF(IYMAX),ZF(IXMAX,IYMAX),STAT=IERR)
       IF(IERR.EQ.0)  GO TO 21
       DEALLOCATE(X,Y,Z,XF,YF,ZF,STAT=IERR)
   20 CONTINUE
      STOP
   21 CONTINUE
      CALL CLEARSCREEN(0)
      WRITE(*,'(10X,A)')
     *'╔═══════════════════════════════════════════════════════════════╗'
      WRITE(*,'(10X,A)')
     *'║                                                               ║'
      WRITE(*,'(10X,A)')
     *'║        Welcome to ACCUM - a Feedline Analysis Program         ║'
      WRITE(*,'(10X,A)')
     *'║                                                               ║'
      WRITE(*,'(10X,A)')
     *'║                 To send a plot to the printer                 ║'
      WRITE(*,'(10X,A)')
     *'║                                                               ║'
      WRITE(*,'(10X,A)')
     *'║             The computer MUST be in GRAPHICS mode             ║'
      WRITE(*,'(10X,A)')
     *'║                                                               ║'
      WRITE(*,'(10X,A)')
     *'║      Hit PrScn to send the current plot to the printer        ║'
      WRITE(*,'(10X,A)')
     *'║                                                               ║'
      WRITE(*,'(10X,A)')
     *'╚═══════════════════════════════════════════════════════════════╝'
      WRITE(*,*)' '
      WRITE(*,'(20X,A,I4)')'Maximum no. of frequencies = ',IXMAX
      WRITE(*,'(20X,A,I4)')'Maximum points along pipe  = ',IYMAX
      WRITE(*,*)' '
      CALL MAINP(X,Y,Z,XF,YF,ZF,IXMAX,IYMAX)
      STOP
      END
      SUBROUTINE MAINP(X,Y,Z,XF,YF,ZF,IXMAX,IYMAX)
C       Logic portion of code
      INTEGER*4 IXMAX,IYMAX
      COMPLEX G(0:76),CTANH,G1,S,ZT(0:76),ZG(76),RHS,CFAC,CAPN,CAPM
      COMPLEX ZGEFF,ZOEFF
      REAL AREA(75),DIA(75),L(75),PIPE1(75),PIPE2(75),PIPE3(75),
     *     PIPE4(75),PIPE5(75),ZO(76),PCAP(75),PIND(75)
      REAL KMAN,KTANK,LFLOW,LFREQ,MAG,MAG1
      REAL X(IXMAX,IYMAX),Y(IXMAX,IYMAX),Z(IXMAX,IYMAX)
      REAL XF(IXMAX),YF(IYMAX),ZF(IXMAX,IYMAX)
      INTEGER*2 SECTN(75),PTS,RSPON,SECT,SEGMN
      CHARACTER ANS*1
      CHARACTER*24 NAMLIN,NAMFUL,NAMLOX
      COMMON /WCAOUT/NAMLIN
      COMMON /RELVAL/A,AREA,AREAB,CMAN,CTANK,DENS,DIA,DIME,DPROR,KMAN,
     *               KTANK,L,LFLOW,PCHMB,PIPE1,PIPE2,PIPE3,PIPE4,PIPE5,
     *               TFLOW,VALUE,VOL,VOLMF,PMRAT,SPLIT,PCAP,PIND
```

```
      COMMON /INTVAL/SECT,SECTN,SEGMN,NSEC(75),NPTS,LOPEND,LOPOLD
      COMMON /FREQ/S,ZT,ZG,ZO
      INTEGER*2 IHR,IMIN,ISEC,I100,IYR,IMON,IDAY
      CHARACTER*2 AM,PM,AP
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /FACTOR/SFAC
      DATA AM/'AM'/,PM/'PM'/
      DATA GRAV/32.2/,PI/3.1415927/
      DATA NAMFUL/'FUEL.INP                '/
      DATA NAMLOX/'LOX.INP                 '/
      DATA IOPEN/0/
    1 FORMAT(E15.6)
    2 FORMAT(I5,5E15.6)
    3 FORMAT(1P4E15.6)
    4 FORMAT(1PE13.5,'  (',E12.5,',',E12.5,')  (',E12.5,',',E12.5,')')
    5 FORMAT(/'       FREQ',8X,'FREQ-NORM',9X,'G(R)',11X,'G(I)'/)
    6 FORMAT(/2X,'"    FREQ"',7X,'"FREQ-NORM"',5X,'"   /G1/"',6X,
     *         '"   /G/"'/)
    7 FORMAT('"',A,'"')
    8 FORMAT(I5,1P3E15.6)
   10 FORMAT(A20,2X,I2.2,':',I2.2,A2,3X,I2.2,'-',I2.2,'-',I2.2)
      SFAC=1.0
      WRITE(*,*)' If you want frequency in rad/sec, hit enter.'
      WRITE(*,'(A\)')' If you want it in Hertz, enter "H". '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'H'.OR.ANS.EQ.'h')  SFAC=6.283185
      LOPOLD=20
      CALL GETTIM(IHR,IMIN,ISEC,I100)
      CALL GETDAT(IYR,IMON,IDAY)
      IYR=IYR-1900
      IF(IHR.LT.12)  THEN
       AP=AM
      ELSE
       AP=PM
       IF(IHR.GT.12)  IHR=IHR-12
      ENDIF
   20 CONTINUE
      WRITE(*,'(A\)')' Is this setup for FUEL or OXIDIZER? Enter F or O
     *. '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'F'.OR.ANS.EQ.'f')  THEN
       WRITE(*,'(A\)')'  Is the name of the I/O file FUEL.INP? Y or N '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'N'.OR.ANS.EQ.'n')  THEN
        WRITE(*,*)' Enter name of I/O file'
        READ(*,'(A)')NAMLIN
       ELSE
        NAMLIN=NAMFUL
       ENDIF
      ELSEIF(ANS.EQ.'O'.OR.ANS.EQ.'o')  THEN
```

```
       WRITE(*,'(A\)')'  Is the name of the I/O file LOX.INP? Y or N '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'N'.OR.ANS.EQ.'n')  THEN
        WRITE(*,*)' Enter name of I/O file'
        READ(*,'(A)')NAMLIN
       ELSE
        NAMLIN=NAMLOX
       ENDIF
      ELSE
       WRITE(*,*)' You did not enter F or O.  Try again'
       GO TO 20
      ENDIF
      OPEN(UNIT=11,FILE=NAMLIN)
      OPEN(UNIT=12,FILE='SURF.DAT')
      WRITE(*,'(A\)')' If there is data stored enter Y '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'N'.OR.ANS.EQ.'n')  THEN
       RSPON=4
       GO TO 24
      ENDIF
   21 CONTINUE
      SPLIT=1.0
      LOPEND=1
C        TITLE
      READ(11,'(A)')TITLF
      WRITE(TITLE,10)TITLF,IHR,IMIN,AP,IMON,IDAY,IYR
C        TANK CONDITIONS
      READ(11,1)VOL
      READ(11,1)LFLOW
      READ(11,1)KTANK
C        MANIFOLD CONDITIONS
      READ(11,1)DENS
      READ(11,1)TFLOW
      READ(11,1)VOLMF
      READ(11,1)KMAN
      READ(11,1)PCHMB
C        ORFICE CONDITION
      READ(11,1)DPROR
      A=SQRT(GRAV*KTANK/DENS)
      CTANK=DENS*VOL/KTANK
      CMAN=DENS*VOLMF/KMAN
      PMRAT=PCHMB/TFLOW
      AVGK=0.5*(KTANK+KMAN)
      READ(11,2)SEGMN
      DO 22 I=1,SEGMN
      READ(11,2)SECTN(I),PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),PIPE5(I)
      IF(SECTN(I).EQ.0)  THEN
C             BEND IN PIPE
       CALL BENDS(PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),VALUE,DIME)
       AREAB=0.785398*DIME**2
       L(I)=VALUE
       AREA(I)=AREAB
       DIA(I)=DIME
      ELSEIF(SECTN(I).EQ.1.OR.SECTN(I).EQ.9)  THEN
C             STRAIGHT SECTION  OR SPLIT
       VALUE=PIPE1(I)
       DIME=PIPE2(I)
       AREAB=0.785398*DIME**2
       L(I)=VALUE
       AREA(I)=AREAB
       DIA(I)=DIME
       IF(SECTN(I).EQ.9)  THEN
C             SPLIT PIPE
        SPLIT=PIPE3(I)
        WRITE(*,'(A,I3)')' Maximun no. of iterations is set at ',LOPOLD
        WRITE(*,'(A\)')' Do you wish to change it? '
        READ(*,'(A)')ANS
        IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
         WRITE(*,'(A\)')' Enter maximum no. of iterations '
         READ(*,*)LOPOLD
        ENDIF
        LOPEND=LOPOLD
       ENDIF
      ELSEIF(SECTN(I).EQ.2)  THEN
C             INLINE ACCUMULATOR
C          PIPE1 - LEN   - L
C          PIPE2 - DIA   - DIA
C          PIPE3 - DEN
C          PIPE4 - K
       L(I)=PIPE1(I)
       DIA(I)=PIPE2(I)
       AREA(I)=0.25*PI*PIPE2(I)**2
       IF(PIPE3(I).EQ.0.0)  PIPE3(I)=DENS
       IF(PIPE4(I).EQ.0.0)  PIPE4(I)=AVGK
       PCAP(I)=PIPE3(I)*L(I)*AREA(I)*PMRAT/PIPE4(I)
      ELSEIF(SECTN(I).EQ.3)  THEN
C             TUNED STUB ACCUMULATOR
C               SUPPRESSES OMEGA = (PI/2)/(L*SQRT(PIND*PCAP))
C          PIPE1 - LEN   - L
C          PIPE2 - DIA   - DIA
C          PIPE3 - DEN
C          PIPE4 - K
       L(I)=PIPE1(I)
       DIA(I)=PIPE2(I)
       AREA(I)=0.25*PI*DIA(I)**2
       IF(PIPE3(I).EQ.0.0)  PIPE3(I)=DENS
       IF(PIPE4(I).EQ.0.0)  PIPE4(I)=AVGK
       PCAP(I)=PIPE3(I)*L(I)*AREA(I)*PMRAT/PIPE4(I)
       PIND(I)=L(I)/(AREA(I)*GRAV*PMRAT)
      ELSEIF(SECTN(I).EQ.4.OR.SECTN(I).EQ.5)  THEN
C             HELMHOLTZ RESONATOR ACCUMULATOR
C             PARALLEL RESONATOR ACCUMULATOR
C               SUPPRESSES OMEGA = 1/SQRT(PIND*PCAP)
C          PIPE1 - LEN   - L
```

```
C          PIPE2 - DIA   - DIA
C          PIPE3 - VOL   - AREA
C          PIPE4 - DEN
C          PIPE5 - K
       L(I)=PIPE1(I)
       DIA(I)=PIPE2(I)
       AREA(I)=PIPE3(I)
       IF(PIPE4(I).EQ.0.0)  PIPE4(I)=DENS
       IF(PIPE5(I).EQ.0.0)  PIPE5(I)=AVGK
       PCAP(I)=PIPE4(I)*AREA(I)*PMRAT/PIPE5(I)
       PIND(I)=L(I)/(0.25*PI*DIA(I)**2*GRAV*PMRAT)
      ELSEIF(SECTN(I).EQ.6)  THEN
C             PUMP
C          PIPE1 - LEN   - L
C          PIPE2 - DIA   - DIA
C          PIPE3 - DP/DM - AREA
C          PIPE4 - IND   - PIND
C          PIPE5 - CAP   - PCAP
       L(I)=PIPE1(I)
       DIA(I)=PIPE2(I)
       AREA(I)=PIPE3(I)
       PCAP(I)=PIPE4(I)*PMRAT
       PIND(I)=PIPE5(I)/PMRAT
      ENDIF
   22 CONTINUE
C
C      The first stage in this program is to define the parameters then
C      we will begin the initial calculations. Because these parameters
C      are as likely to change as not, a provision is made to update the
C      parameters if necessary.
C
      WRITE(12,*)' '
      WRITE(12,*)TITLE
      WRITE(12,*)' '
      WRITE(12,*)'PRESENT CONDITIONS ARE AS FOLLOWS:'
      WRITE(12,*)'FUEL TANK VOLUME=',VOL
      WRITE(12,*)'LINE FLOW RATE=',LFLOW
      WRITE(12,*)'BULK MOD. OF FUEL TANK=',KTANK
      WRITE(12,*)'VELOCITY OF SOUND IN FLUID=',A
      WRITE(12,*)'CAPACITANCE OF FUEL TANK=',CTANK
      WRITE(12,*)'DENS=',DENS
      WRITE(12,*)'TOTAL FLOW RATE=',TFLOW
      WRITE(12,*)'MANIFOLD VOLUME=',VOLMF
      WRITE(12,*)'BULK MOD. OF MANIFOLD=',KMAN
      WRITE(12,*)'ENGINE CHAMBER PRESSURE=',PCHMB
      WRITE(12,*)'CAPACITANCE OF MANIFOLD=',CMAN
      WRITE(12,*)'PRESSURE DROP ACROSS ORIFICE=',DPROR
      WRITE(12,*)' STATUS   LENGTH          AREA          DIAMETER'
      WRITE(12,8)(SECTN(I),L(I),AREA(I),DIA(I),I=1,SEGMN)
      WRITE(12,*)' '
      WRITE(*,*)' '
      WRITE(*,*)TITLE
      WRITE(*,*)' '
      WRITE(*,*)' PRESENT CONDITIONS ARE AS FOLLOWS:'
      WRITE(*,*)' FUEL TANK VOLUME=',VOL
      WRITE(*,*)' LINE FLOW RATE=',LFLOW
      WRITE(*,*)' BULK MOD. OF FUEL TANK=',KTANK
      WRITE(*,*)' VELOCITY OF SOUND IN FLUID=',A
      WRITE(*,*)' CAPACITANCE OF FUEL TANK=',CTANK
      WRITE(*,*)' DENS=',DENS
      WRITE(*,*)' TOTAL FLOW RATE=',TFLOW
      WRITE(*,*)' MANIFOLD VOLUME=',VOLMF
      WRITE(*,*)' BULK MOD. OF MANIFOLD=',KMAN
      WRITE(*,*)' ENGINE CHAMBER PRESSURE=',PCHMB
      WRITE(*,*)' CAPACITANCE OF MANIFOLD=',CMAN
      WRITE(*,*)' PRESSURE DROP ACROSS ORIFICE=',DPROR
      WRITE(*,*)' STATUS   LENGTH          AREA          DIAMETER'
      WRITE(*,8)(SECTN(I),L(I),AREA(I),DIA(I),I=1,SEGMN)
      WRITE(*,*)' If revisions on the design have been made'
      WRITE(*,*)' (changes in fuel, pipe length, diameter, bends, etc.)'
      WRITE(*,'(A\)')' Please enter yes for revisions or no to continue.
     * '
      READ(*,'(A)')ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  GO TO 25
   23 CONTINUE
      RSPON=0
   24 CONTINUE
      CALL MODIFY(RSPON)
C
C       THIS SECTION COMPUTES THE NEW ADMITTANCE OVER VARYING FREQUENCIES.
C
   25 CONTINUE
      IF(SFAC.EQ.1.0)  THEN
       WRITE(*,*)' Enter range of frequencies in rad/sec '
      ELSE
       WRITE(*,*)' Enter range of frequencies in Hertz '
      ENDIF
      WRITE(*,*)' Low freq=1 high freq=2 #pts=10'
      READ(*,*)LFREQ,HFREQ,PTS
      IF(PTS.LT.1)  GO TO 29
C
C       THIS SECTION WILL COMPUTE THE ADMITTANCE RATIO FOR THE FUEL TANK
C       AND THEN IT WILL COMPUTE THE ADMITTANCE RATIOS FOR EACH SEGMENT,
C       SINCE THERE ARE L(I) I=1,SEGMN LENGTHS, THEN THERE WILL BE AT LEAST
C       AS MANY ADMITTANCE RATIOS, THEREFORE I AM SETTING UP AN ARRAY FOR
C       EACH LENGTH L(I) HAVING AN ADMITTANCE RATIO G(I).
C
      IPLT=0
      IF(PTS.GT.IXMAX)  THEN
       WRITE(*,*)' Maximum number of points for this option is IXMAX =',
     *          IXMAX
       WRITE(*,*)'    Do you want PTS reduced to IXMAX? Y or N'
       READ(*,'(A)')ANS
       IF(ANS.EQ.'N'.OR.ANS.EQ.'n')  GO TO 29
       PTS=IXMAX
      ENDIF
      IF(LFREQ.EQ.0.0)  LFREQ=1.0E-5
      WRITE(*,*)' Do you wish to plot ADMITTANCE as it is calculated? Y
     *or N '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
       WRITE(*,*)' Enter estimated maximum value of admittance '
       READ(*,*)ADMMAX
       IPLT=1
      ENDIF
      SSIZE=0.0
      IF(PTS.NE.1)  SSIZE=(HFREQ-LFREQ)/(PTS-1)
      ZTOP=A/(GRAV*PMRAT)
      ZOR=2.0*DPROR/(LFLOW*PMRAT)
  252 CONTINUE
      TLT=0.0
      ISIZ=0
      DO 26 I=1,SEGMN
       IF(SECTN(I).EQ.3.OR.SECTN(I).EQ.4)  THEN
        TLT=TLT+DIA(I)
       ELSE
        TLT=TLT+L(I)
       ENDIF
       IF(SECTN(I).LE.1.OR.SECTN(I).EQ.9)  THEN
        ZO(I)=ZTOP/AREA(I)
        WRITE(*,*)' This section is ',L(I),' ft. long'
        WRITE(*,*)'     How many segments should it be broken into? '
        READ(*,*)NSEC(I)
        IF(NSEC(I).LE.1)  NSEC(I)=2
       ELSEIF(SECTN(I).EQ.2)  THEN
        ZO(I)=ZTOP/AREA(I)
        NSEC(I)=2
       ELSE
        ZO(I)=SQRT(PIND(I)/PCAP(I))
        NSEC(I)=2
       ENDIF
       ISIZ=ISIZ+NSEC(I)
       IF(ISIZ.GT.IYMAX)  THEN
        WRITE(*,*)' Too many segments ',ISIZ
        WRITE(*,*)' Maximun is IYMAX =',IYMAX,' Try again.'
        GO TO 252
       ENDIF
   26 CONTINUE
      TLT=TLT/(PI*A)
C       PLOT PIPE LAYOUT IN WINDOW 1
      CALL SETPLT
      CALL PIPPLOT(SEGMN,SECTN,PIPE1,PIPE2,PIPE3,PIPE4)
      IF(IPLT.EQ.1)  THEN
C       PLOT ADMITTANCE IN WINDOW 2
       CALL LOWERW(LFREQ,HFREQ,ADMMAX)
       CALL ADMGRAPH(LFREQ,HFREQ,ADMMAX)
      ENDIF
      WRITE(12,5)
      IF(IOPEN.NE.0.AND.LOPEND.NE.1)  THEN
       WRITE(13,*)' '
       WRITE(13,*)' '
       WRITE(13,*)TITLE
       WRITE(13,*)' '
      ENDIF
      DO 28 K=1,PTS
       W=LFREQ+SSIZE*(K-1)
       XF(K)=W
       S=CMPLX(0.0,W*SFAC)
       G(0)=CTANK*PMRAT*S
       G(0)=G(0)/SPLIT
       ZT(0)=1.0/G(0)
      DO 281 KLOOP=1,LOPEND
       G1=G(0)+1.0
       DO 27 I=1,SEGMN
        ZGEFF=G(I-1)
        IF(SECTN(I).LE.1.OR.SECTN(I).EQ.9)  THEN
C             BEND IN PIPE OR STRAIGHT SECTION
         TL=L(I)/A
         IF(KLOOP.NE.1.AND.SECTN(I).EQ.9)  THEN
          ZGEFF=G(I-1)+(SPLIT-1.0)/ZG(I-1)
         ENDIF
         G(I)=(1.0+CTANH(S*TL)/(ZGEFF*ZO(I)))/(1.0+ZGEFF*ZO(I)*
     *          CTANH(S*TL))
        ELSEIF(SECTN(I).EQ.2)  THEN
C             INLINE RESONATOR ACCUMULATOR
         G(I)=1.0+PCAP(I)*S/ZGEFF
        ELSEIF(SECTN(I).EQ.3)  THEN
C             TUNED STUB ACCUMULATOR
         G(I)=1.0+CTANH(S*SQRT(PIND(I)*PCAP(I)))/(ZO(I)*ZGEFF)
        ELSEIF(SECTN(I).EQ.4)  THEN
C             HELMHOLTZ RESONATOR ACCUMULATOR
         G(I)=S*PCAP(I)/(1.0+PIND(I)*PCAP(I)*S**2)
         G(I)=1.0+G(I)/ZGEFF
        ELSEIF(SECTN(I).EQ.5)  THEN
C             PARALLEL RESONATOR ACCUMULATOR
         G(I)=PIND(I)*PCAP(I)*S**2+1.0
         G(I)=G(I)/(G(I)+PIND(I)*S*ZGEFF)
        ELSEIF(SECTN(I).EQ.6)  THEN
C             PUMP
         G(I)=(1.0+PCAP(I)*S/ZGEFF)/(1.0+(PIND(I)*S+AREA(I))*
     *         (PCAP(I)*S+ZGEFF))
        ENDIF
        G1=G1*G(I)
        G(I)=G(I)*ZGEFF
        ZT(I)=1.0/G(I)
   27 CONTINUE
       G(SEGMN+1)=1.0+CMAN*PMRAT*S/G(SEGMN)
       G1=G1*G(SEGMN+1)
      G(SEGMN+1)=G(SEGMN+1)*G(SEGMN)
      G(SEGMN+2)=1.0/(1.0+ZOR*G(SEGMN+1))
      G1=G1*G(SEGMN+2)
      G(SEGMN+2)=G(SEGMN+2)*G(SEGMN+1)
      ZG(SEGMN)=ZOR/(ZOR*CMAN*PMRAT*S+1.0)
      IF(SEGMN.NE.1)  THEN
       DO 271 I=SEGMN-1,1,-1
        ZGEFF=ZG(I+1)
        ZOEFF=ZO(I+1)
        IF(SECTN(I+1).LE.1.OR.SECTN(I+1).EQ.9)  THEN
C            BEND IN PIPE OR STRAIGHT SECTION
         TL=(L(I)+L(I+1))/A
         CAPN=(ZOEFF-ZT(I-1))/(ZOEFF+ZT(I-1))
         CAPM=(ZOEFF-ZGEFF)/(ZOEFF+ZGEFF)
         CFAC=CEXP(-2.0*S*TL)
         RHS=(ZOEFF+ZGEFF)*(1.0-CAPN*CAPM*CFAC)*CEXP(S*L(I+1)/A)
         CFAC=CAPN*CFAC*CEXP(2.0*S*L(I+1)/A)
         ZG(I)=(RHS-ZOEFF*(1.0-CFAC))/(1.0+CFAC)
         IF(SECTN(I+1).EQ.9)  THEN
          ZG(I)=ZG(I)/SPLIT
         ENDIF
        ELSEIF(SECTN(I+1).EQ.2)  THEN
C            INLINE RESONATOR ACCUMULATOR
         ZG(I)=ZGEFF/(ZGEFF*PCAP(I+1)*S+1.0)
        ELSEIF(SECTN(I+1).EQ.3)  THEN
C            TUNED STUB ACCUMULATOR
         ZG(I)=ZOEFF/CTANH(S*SQRT(PIND(I+1)*PCAP(I+1)))
         ZG(I)=(ZG(I)*ZGEFF)/(ZG(I)+ZGEFF)
        ELSEIF(SECTN(I+1).EQ.4)  THEN
C            HELMHOLTZ RESONATOR ACCUMULATOR
         ZG(I)=(1.0+PIND(I+1)*PCAP(I+1)*S**2)/(PCAP(I+1)*S)
         ZG(I)=(ZG(I)*ZGEFF)/(ZG(I)+ZGEFF)
        ELSEIF(SECTN(I+1).EQ.5)  THEN
C            PARALLEL RESONATOR ACCUMULATOR
         ZG(I)=ZGEFF+PIND(I+1)*S/(PIND(I+1)*PCAP(I+1)*S**2+1.0)
        ELSEIF(SECTN(I+1).EQ.6)  THEN
C            PUMP
         ZG(I)=ZGEFF+PIND(I+1)*S-AREA(I+1)
         ZG(I)=ZG(I)/(1.0+ZG(I)*PCAP(I+1)*S)
        ENDIF
  271 CONTINUE
      ENDIF
      CALL FREQRS(YF,ZF,K,IXMAX,IYMAX,KLOOP,ERRP,WVAL)
      IF(KLOOP.GT.1.AND.ERRP.LT.0.001)  GO TO 282
  281 CONTINUE
      IF(LOPEND.EQ.1)  GO TO 282
      IF(IOPEN.EQ.0)  THEN
       OPEN(UNIT=13,FILE='SURF.ERR')
       WRITE(13,*)' '
       WRITE(13,*)' '
       WRITE(13,*)TITLE
       WRITE(13,*)' '
```

```
        IOPEN=1
       ENDIF
       WRITE(13,'('' jw ='',F8.1,''  after'',I3,'' iterations'',
     *            '' has error of'',F8.3,''% out of'',F8.3)')
     *            W,LOPEND,100.0*ERRP,WVAL
  282 CONTINUE
       MAG=CABS(G(SEGMN+2))
       MAG1=CABS(G1)
       WN=W*TLT
       WRITE(12,3)W,WN,G(SEGMN+2)
       IF(IPLT.EQ.0)  THEN
        X(K,1)=W
        Y(K,1)=MAG
       ELSE
        CALL NEXPT(W,MAG)
       ENDIF
   28 CONTINUE
      IF(IPLT.EQ.0)  THEN
       CALL ALLPT(X,Y,PTS)
      ENDIF
      CALL ENDPLT
      WRITE(*,'(A\)')' Do you wish to plot the surface? '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
       CALL PLOTSU(X,Y,Z,XF,YF,ZF,NPTS,PTS,IXMAX,IYMAX)
      ENDIF
      WRITE(*,'(A\)')' Do you wish to plot contours? '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
       CALL PLTCON(X,Y,Z,XF,YF,ZF,NPTS,PTS,IXMAX,IYMAX)
      ENDIF
   29 CONTINUE
      WRITE(*,'(A\)')' Enter E to exit, F to run new frequency range, or
     * C to run a new case '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'F'.OR.ANS.EQ.'f')  GO TO 25
      IF(ANS.EQ.'E'.OR.ANS.EQ.'e')  RETURN
      IF(ANS.EQ.'C'.OR.ANS.EQ.'c')  THEN
       WRITE(*,'(A\)')' Do you wish to use old data with changes? Y or N
     * '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  GO TO 23
       WRITE(*,'(A\)')' Does INPUT file need to be rewound? Y or N '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  REWIND 11
       GO TO 21
      ENDIF
      WRITE(*,*)' You did not enter E, F, or C. Try again.'
      GO TO 29
      END
      SUBROUTINE ADMGRAPH(LFREQ,HFREQ,ADMMAX)
C       Plots admittance curve
```

```
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /NOCOL/MODE,MODET,NTROWS,NTCOLS,NPROWS,NPCOLS
      COMMON /FACTOR/SFAC
      REAL LFREQ
    1 FORMAT(F6.3)
      XMIN=LFREQ
      XMAX=HFREQ
      YMIN=0.0
      YMAX=ADMMAX
      XMAJ=0.25*(XMAX-XMIN)
      YMAJ=0.25*(YMAX-YMIN)
      IF(MODE.NE.18)  THEN
       CALL QPTXT(40,TITLE,7,17,11)
      ELSE
       CALL QPTXT(40,TITLE,7,17,14)
      ENDIF
      CALL QXAXIS(XMIN,XMAX,XMAJ,0,-1,2)
      IF(SFAC.EQ.1)  THEN
       CALL QPTXTA(20,'Frequency - rad/sec ',7)
      ELSE
       CALL QPTXTA(20,' Frequency - Hertz  ',7)
      ENDIF
      CALL QYAXIS(YMIN,YMAX,YMAJ,0,0,0)
      CALL QPTXTD(8,'Adm.    ',7)
      CALL QYAXIS(YMIN,YMAX,YMAJ,0,-1,2)
      RETURN
      END
      SUBROUTINE ALLPT(X,Y,PTS)
C       Supervises plot of admittance after calculations
      INTEGER*2 PTS
      REAL X(PTS),Y(PTS)
      ADMMAX=Y(1)
      DO 21 I=2,PTS
       IF(Y(I).GT.ADMMAX)  ADMMAX=Y(I)
   21 CONTINUE
      CALL LOWERW(X(1),X(PTS),ADMMAX)
      CALL ADMGRAPH(X(1),X(PTS),ADMMAX)
      CALL QTABL(1,PTS,X,Y)
      RETURN
      END
      SUBROUTINE BENDS(PIPE1,PIPE2,PIPE3,PIPE4,VALUE,DIME)
C       Computes effective straight pipe for bend
      REAL LBEND,INRAD,INERT,LPRME,NEWLN
      BENDR=0.0174533*ABS(PIPE2)
      LBEND=PIPE1*BENDR
      ARBND=0.785398*PIPE3**2
      INRAD=PIPE1-0.5*PIPE3
      OTRAD=PIPE1+0.5*PIPE3
      RATIO=INRAD/OTRAD
      X=RATIO
      CALL GINERT(ABS(PIPE2),X,Y)
      INERT=(Y*(OTRAD-INRAD))/ARBND
      LPRME=LBEND/ARBND
      NEWLN=LPRME+INERT
      GAMMA=NEWLN/LPRME
      VALUE=GAMMA*(LBEND+2.0*PIPE4)
      AREAB=ARBND/SQRT(GAMMA)
      DIME=2.0*SQRT(AREAB/3.1415927)
      RETURN
      END
      SUBROUTINE BNSECT(J,ITYPE,POINT,PIPE1,PIPE2,PIPE3,PIPE4)
C       Computes plot coordinates for a bend
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
      REAL POINT(8,200)
      INTEGER*2 ITYPE(200)
C          FIRST STRAIGHT SECTION OF BEND
      IF(PIPE4.NE.0.0)  CALL STSECT(J,ITYPE,POINT,PIPE4,PIPE3)
C          CURVED SECTION OF BEND
      IF(PIPE2.GE.0.0)  THEN
       XC=X-SINA*PIPE1
       YC=Y+COSA*PIPE1
       DIA= 0.5
      ELSE
       XC=X+SINA*PIPE1
       YC=Y-COSA*PIPE1
       DIA=-0.5
      ENDIF
      J=J+1
      ITYPE(J)=0
      POINT(1,J)=XC
      POINT(2,J)=YC
      POINT(3,J)=ANG
      ANG=ANG+0.01745329*PIPE2
      ANGLE=ANGLE+0.5*PIPE2
      RANG=0.01745329*ANGLE
      COSA=COS(RANG)
      SINA=SIN(RANG)
      RAD=PIPE1-DIA*PIPE3
      POINT(4,J)=ANG
      POINT(5,J)=RAD
      X0=XC-RAD
      Y0=YC+RAD
      X1=XC+RAD
      Y1=YC-RAD
      X2=XH
      Y2=YH
      SLENTH=2.0*RAD*SIN(0.00872665*ABS(PIPE2))
      XH=X2+COSA*SLENTH
      YH=Y2+SINA*SLENTH
      X3=XH
      Y3=YH
      IF(DIA.LT.0.0)  THEN
       HOLD=X2
       X2=X3
       X3=HOLD
       HOLD=Y2
       Y2=Y3
       Y3=HOLD
      ENDIF
      RAD=PIPE1+DIA*PIPE3
      X0=XC-RAD
      Y0=YC+RAD
      X1=XC+RAD
      Y1=YC-RAD
      X2=XL
      Y2=YL
      SLENTH=2.0*RAD*SIN(0.00872665*ABS(PIPE2))
      XL=X2+COSA*SLENTH
      YL=Y2+SINA*SLENTH
      X3=XL
      Y3=YL
      IF(DIA.LT.0.0)  THEN
       HOLD=X2
       X2=X3
       X3=HOLD
       HOLD=Y2
       Y2=Y3
       Y3=HOLD
      ENDIF
      J=J+1
      ITYPE(J)=0
      POINT(1,J)=POINT(1,J-1)
      POINT(2,J)=POINT(2,J-1)
      POINT(3,J)=POINT(3,J-1)
      POINT(4,J)=POINT(4,J-1)
      POINT(5,J)=RAD
      SLENTH=2.0*PIPE1*SIN(0.00872665*ABS(PIPE2))
      X=X+COSA*SLENTH
      Y=Y+SINA*SLENTH
      XMIN=AMIN1(X,XL,XH,XMIN)
      XMAX=AMAX1(X,XL,XH,XMAX)
      YMIN=AMIN1(Y,YL,YH,YMIN)
      YMAX=AMAX1(Y,YL,YH,YMAX)
C          LAST STRAIGHT SECTION OF BEND
      ANGLE=ANGLE+0.5*PIPE2
      RANG=0.01745329*ANGLE
      COSA=COS(RANG)
      SINA=SIN(RANG)
      J=J+1
      ITYPE(J)=1
      POINT(1,J)=XH
      POINT(2,J)=YH
      POINT(3,J)=XL
      POINT(4,J)=YL
      X=X+COSA*PIPE4
      XH=X-0.5*SINA*PIPE3
      XL=X+0.5*SINA*PIPE3
      Y=Y+SINA*PIPE4
      YH=Y+0.5*COSA*PIPE3
      YL=Y-0.5*COSA*PIPE3
      POINT(5,J)=XH
      POINT(6,J)=YH
      POINT(7,J)=XL
      POINT(8,J)=YL
      XMIN=AMIN1(X,XL,XH,XMIN)
      XMAX=AMAX1(X,XL,XH,XMAX)
      YMIN=AMIN1(Y,YL,YH,YMIN)
      YMAX=AMAX1(Y,YL,YH,YMAX)
      RETURN
      END
      COMPLEX FUNCTION CCOSH(S)
C       Evaluates the complex hyperbolic cosine
      COMPLEX S
      REAL LAMDA, MU
      LAMDA=REAL(S)
      MU=AIMAG(S)
      COSHR=COSH(LAMDA)*COS(MU)
      COSHI=SINH(LAMDA)*SIN(MU)
      CCOSH=CMPLX(COSHR,COSHI)
      RETURN
      END
      COMPLEX FUNCTION CSINH(S)
C       Evaluates the complex hyperbolic sine
      COMPLEX S
      REAL LAMDA, MU
      LAMDA=REAL(S)
      MU=AIMAG(S)
      SINHR=SINH(LAMDA)*COS(MU)
      SINHI=COSH(LAMDA)*SIN(MU)
      CSINH=CMPLX(SINHR,SINHI)
      RETURN
      END
      COMPLEX FUNCTION CTANH(S)
C       Evaluates the complex hyperbolic tangent
      COMPLEX CCOSH,CSINH,S
      CTANH=CSINH(S)/CCOSH(S)
      RETURN
      END
      SUBROUTINE ENDPLT
C       Closes plot routines
      COMMON /WCAPAS/IFRST
      COMMON /NOCOL/MODE,MODET,NTROWS,NTCOLS,NPROWS,NPCOLS
   21 CONTINUE
```

```
      CALL QONKEY(IKEY)
      IF(IKEY.EQ.0)  GO TO 21
      CALL QINKEY(IEXTEN,IKEY)
      IF(IKEY.EQ.80.OR.IKEY.EQ.112)  CALL QPSCRN
      CALL QSMODE(MODET)
      RETURN
      END
      SUBROUTINE FREQRS(YF,ZF,K,IXMAX,IYMAX,KLOOP,ERRP,WVAL)
C       Computes pressure transfer function
      COMPLEX S,ZT(0:76),ZG(76),LITTLN,CAPM,CAPN,ZFAC,TOP,BOTTOM,PRAT
      REAL AREA(75),DIA(75),L(75),PIPE1(75),PIPE2(75),PIPE3(75),
     *     PIPE4(75),PIPE5(75),ZO(76),PIND(75),PCAP(75)
      REAL KMAN,KTANK,LFLOW
      INTEGER*2 SECTN(75),SECT,SEGMN
      COMMON /RELVAL/A,AREA,AREAB,CMAN,CTANK,DENS,DIA,DIME,DPROR,KMAN,
     *               KTANK,L,LFLOW,PCHMB,PIPE1,PIPE2,PIPE3,PIPE4,PIPE5,
     *               TFLOW,VALUE,VOL,VOLMF,PMRAT,SPLIT,PCAP,PIND
      COMMON /INTVAL/SECT,SECTN,SEGMN,NSEC(75),NPTS,LOPEND,LOPOLD
      COMMON /FREQ/S,ZT,ZG,ZO
      INTEGER*4 IXMAX,IYMAX
      REAL YF(IYMAX),ZF(IXMAX,IYMAX),PRATO(2,75)
      LITTLN=S/A
      SUMX=0.0
      M=1
      ERRP=0.0
      DO 22 I=SEGMN,1,-1
       CAPN=(ZO(I)-ZT(I-1))/(ZO(I)+ZT(I-1))
       CAPM=(ZO(I)-ZG(I))/(ZO(I)+ZG(I))
       ZFAC=ZO(I)/(ZO(I)+ZG(I))
       LSEC=NSEC(I)
       DX=0.0
       IF(SECTN(I).EQ.3.OR.SECTN(I).EQ.4)  THEN
        DX=DIA(I)/(LSEC-1)
       ELSE
        DX=L(I)/(LSEC-1)
       ENDIF
       BOTTOM=1.0-CAPM*CAPN*CEXP(-2.0*LITTLN*L(I))
       DO 21 J=1,LSEC
        X=DX*(J-1)
        IF(SECTN(I).GT.1.AND.SECTN(I).LT.6)  THEN
         IF(J.EQ.LSEC)  PRAT=ZT(I-1)/(ZT(I-1)+ZG(I))
        ELSE
         TOP=CEXP(-LITTLN*X)-CAPN*CEXP(-LITTLN*(2.0*L(I)-X))
         PRAT=ZFAC*TOP/BOTTOM
        ENDIF
        IF(J.NE.1)  THEN
         SUMX=SUMX+DX
         M=M+1
         ZF(K,M)=CABS(PRAT)
         IF(K.EQ.1)  YF(M)=SUMX
        ELSE
         IF(I.EQ.SEGMN)  THEN
```

```
          ZF(K,M)=CABS(PRAT)
          IF(K.EQ.1)  YF(M)=SUMX
         ENDIF
        ENDIF
       IF(J.NE.1.AND.J.NE.LSEC)  GO TO 21
        PRATN=CABS(PRAT)
        IF(KLOOP.NE.1)  THEN
         IF(J.EQ.1)  THEN
          ERRN=ABS((PRATN-PRATO(1,I))/PRATN)
         ELSE
          ERRN=ABS((PRATN-PRATO(2,I))/PRATN)
         ENDIF
         ERRP=AMAX1(ERRP,ERRN)
         IF(ERRP.EQ.ERRN)  WVAL=PRATN
        ENDIF
        IF(J.EQ.1)  PRATO(1,I)=PRATN
        IF(J.EQ.LSEC)  PRATO(2,I)=PRATN
   21 CONTINUE
   22 CONTINUE
      IF(K.EQ.1)  NPTS=M
      RETURN
      END
      SUBROUTINE GINERT(BEND,X,Y)
C       Evaluates curve fit of inertance of bends
      DIMENSION B(3)
      DATA B/0.0,0.7877014E-02,-0.2814679E-04/
      A=B(1)+(B(2)+B(3)*BEND)*BEND
      Y=A*(X-1.0)**2
      RETURN
      END
      SUBROUTINE HHSECT(J,ITYPE,POINT,LEN,DIA,VOL)
C       Computes plot coordinates for Helmholtz resonator
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      REAL LEN,POINT(8,200)
      INTEGER*2 ITYPE(200)
      XOLD=X
      XHOLD=XH
      XLOLD=XL
      YOLD=Y
      YHOLD=YH
      YLOLD=YL
      SINOLD=SINA
      COSOLD=COSA
      DIAM=SQRT((XH-XL)**2+(YH-YL)**2)
      CALL TSSECT(J,ITYPE,POINT,LEN,DIA)
      XC=0.5*(XOLD+X)
      YC=0.5*(YOLD+Y)
      XOLD=X
      YOLD=Y
      SINA=COSOLD
      COSA=-SINOLD
      X=XC+COSA*(LEN+0.5*DIAM)
      Y=YC+SINA*(LEN+0.5*DIAM)
      SIDE=VOL**0.3333333
      CALL STSECT(J,ITYPE,POINT,SIDE,SIDE)
      X=XOLD
      Y=YOLD
      SINA=SINOLD
      COSA=COSOLD
      DIAM=SQRT((XHOLD-XLOLD)**2+(YHOLD-YLOLD)**2)
      XH=X-0.5*SINA*DIAM
      XL=X+0.5*SINA*DIAM
      YH=Y+0.5*COSA*DIAM
      YL=Y-0.5*COSA*DIAM
      RETURN
      END
      SUBROUTINE LOWERW(LFREQ,HFREQ,ADMMAX)
C       Sets up lower plotting window
      COMMON /NOCOL/MODE,MODET,NTROWS,NTCOLS,NPROWS,NPCOLS
      COMMON /ADMCOL/ADMBAC,ADMLIN
      INTEGER ADMBAC,ADMLIN
      REAL LFREQ
      XMIN=LFREQ
      XMAX=HFREQ
      YMIN=0.0
      YMAX=ADMMAX
      XORG=XMIN
      YORG=YMIN
      XLEN=0.01*(XMAX-XMIN)
      YLEN=0.01*(YMAX-YMIN)
      XMIN=XMIN-XLEN
      XMAX=XMAX+XLEN
      YMIN=YMIN-YLEN
      YMAX=YMAX+YLEN
      JCOL1=150
      JCOL2=550
      IF(MODE.EQ.6)  THEN
       JROW1=20
       JROW2=79
      ELSE
       JROW1=40
       IF(MODE.EQ.16)  JROW2=134
       IF(MODE.EQ.18)  JROW2=199
      ENDIF
      YOVERX=1.0
      IOPT=0
      ASPECT=1.35
      CALL QPLOT(JCOL1,JCOL2,JROW1,JROW2,XMIN,XMAX,YMIN,YMAX,
     *           XORG,YORG,IOPT,YOVERX,ASPECT)
      IF(MODE.NE.6)  THEN
       CALL QPREG(0,ADMBAC)
      ENDIF
       CALL QSETUP(0,ADMLIN,-2,ADMLIN)
      RETURN
      END
      SUBROUTINE MODIFY(RSPON)
C       Allows modifications to input data
      REAL AREA(75),DIA(75),L(75),PIPE1(75),PIPE2(75),PIPE3(75),
     *     PIPE4(75),PIPE5(75),PIND(75),PCAP(75)
      REAL KMAN,KTANK,LFLOW
      INTEGER*2 SECTN(75),RSPON,SECT,SEGMN
      CHARACTER ANS*1
      CHARACTER*8 VARVAL(9),VARU(9),VARL(9),NAME
      CHARACTER*24 NAMLIN
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /RELVAL/A,AREA,AREAB,CMAN,CTANK,DENS,DIA,DIME,DPROR,KMAN,
     *               KTANK,L,LFLOW,PCHMB,PIPE1,PIPE2,PIPE3,PIPE4,PIPE5,
     *               TFLOW,VALUE,VOL,VOLMF,PMRAT,SPLIT,PCAP,PIND
      COMMON /INTVAL/SECT,SECTN,SEGMN,NSEC(75),NPTS,LOPEND,LOPOLD
      COMMON /WCAOUT/NAMLIN
      DATA GRAV/32.2/,PI/3.141593/
      DATA VARVAL/'  DENS =',' DPROR =','  KMAN =',
     *            ' KTANK =',' LFLOW =',' PCHMB =',' TFLOW =',
     *            '   VOL =',' VOLMF ='/
      DATA VARU/'DENS    ','DPROR   ','KMAN    ',
     *          'KTANK   ','LFLOW   ','PCHMB   ','TFLOW   ',
     *          'VOL     ','VOLMF   '/
      DATA VARL/'dens    ','dpror   ','kman    ',
     *          'ktank   ','lflow   ','pchmb   ','tflow   ',
     *          'vol     ','volmf   '/
    1 FORMAT(1PE15.6)
    2 FORMAT(I5,1P5E15.6)
    3 FORMAT(I5,1P3E15.6)
    4 FORMAT(' This segment is a bend of',1PE13.5,' deg and radius of',
     *       E13.5)
    5 FORMAT(' This segment is straight ',1PE13.5,' diameter pipe ',
     *       E13.5,' ft. long')
    6 FORMAT(A8,1PE13.5,10X,A8,E13.5)
    7 FORMAT(' TITLE = ',A20)
   10 FORMAT(A20,2X,I2.2,':',I2.2,A2,3X,I2.2,'-',I2.2,'-',I2.2)
   11 FORMAT(' This segment is ',I2,' way split ',1PE13.5,' dia.',
     *       ' pipe ',E13.5,' ft. long')
   12 FORMAT(' This segment is a pump with length =',1PE13.5,' dia =',
     *       E13.5/5X,'dp/dm =',E13.5,' capacitance =',E13.5,
     *       ' inductance =',E13.5)
   13 FORMAT(' This segment is a tuned pipe ',1PE13.5,' long & dia =',
     *       E13.5)
   14 FORMAT(' This segment is a Helmholtz resonator with'/5X,'length ='
     *       ,1PE13.5,' dia =',E13.5,' and vol =',E13.5)
   15 FORMAT(' This segment is a parallel resonator with'/5X,'length =',
     *       1PE13.5,' dia =',E13.5,' and vol =',E13.5)
   16 FORMAT(' This segment is a',1PE13.5,' long inline acc. with',
  *      ' diameter of',E13.5)
   AVGK=0.5*(KTANK+KMAN)
   ICHG=0
   IF(RSPON.EQ.4)  GO TO 21
   WRITE(*,*)' Do you wish to change engine & fluid parameters '
   READ(*,'(A)')ANS
   IF(ANS.NE.'Y'.AND.ANS.NE.'y')  GO TO 29
   WRITE(*,*)' Do you wish to change all of the parameters?'
   READ(*,'(A)')ANS
   IF(ANS.NE.'Y'.AND.ANS.NE.'y')  ICHG=1
21 CONTINUE
   IF(ICHG.EQ.0)  THEN
    WRITE(*,'(A\)')' Enter TITLE (20 characters max.) '
    READ(*,'(A)')TITLF
    WRITE(TITLE,10)TITLF,IHR,IMIN,AP,IMON,IDAY,IYR
    WRITE(*,'(A\)')' Enter FUEL TANK VOLUME (ft^3)'
    READ(*,*)VOL
    WRITE(*,'(A\)')' Enter FLOW RATE inside LINE (lbm/sec)'
    READ(*,*)LFLOW
    WRITE(*,'(A\)')' Enter BULK MODULUS of fluid inside TANK (lb /ft^
  *2)'
    READ(*,*)KTANK
    WRITE(*,'(A\)')' Enter FUEL DENSITY (lbm/ft^3)'
    READ(*,*)DENS
    WRITE(*,'(A\)')' Enter TOTAL FLOW RATE inside ENGINE (lbm/sec)'
    READ(*,*)TFLOW
    WRITE(*,'(A\)')' Enter MANIFOLD VOLUME (ft^3)'
    READ(*,*)VOLMF
    WRITE(*,'(A\)')' Enter BULK MODULUS of fluid inside MANIFOLD (lb
  */ft^2)'
    READ(*,*)KMAN
    WRITE(*,'(A\)')' Enter CHAMBER PRESSURE in ENGINE (lbf/ft^2)'
    READ(*,*)PCHMB
    WRITE(*,'(A\)')' Enter PRESSURE DROP across ORIFICE (lbf/ft^2)'
    READ(*,*)DPROR
    A=SQRT(GRAV*KTANK/DENS)
    CTANK=DENS*VOL/KTANK
    CMAN=DENS*VOLMF/KMAN
    PMRAT=PCHMB/TFLOW
   ELSE
    GO TO 24
22  CONTINUE
    WRITE(*,*)'   VARIABLE NAMES AND DESCRIPTIONS'
    WRITE(*,*)' '
    WRITE(*,*)'   TITLE - title (20 characters max.)             '
    WRITE(*,*)'    DENS - density of fluid (lbm/ft^3)            '
    WRITE(*,*)'   DPROR - pressure drop across orfices (lbf/ft^2)'
    WRITE(*,*)'    KMAN - bulk modulus in manifold (lbf/ft^2)    '
    WRITE(*,*)'   KTANK - bulk modulus in tank (lbf/ft^2)        '
    WRITE(*,*)'   LFLOW - mass flow rate of fluid (lbm/sec)      '
    WRITE(*,*)'   PCHMB - chamber pressure (lbf/ft^2)            '
    WRITE(*,*)'   TFLOW - total mass flow inside engine (lbm/sec)'
    WRITE(*,*)'     VOL - volume of storage tank (ft^3)           '
    WRITE(*,*)'   VOLMF - volume of manifold (ft^3)               '
    WRITE(*,*)' '
    GO TO 25
23  CONTINUE
    WRITE(*,*)'   VARIABLE NAMES AND VALUES'
    WRITE(*,*)' '
    WRITE(*,7)TITLF
    WRITE(*,6)VARVAL( 1), DENS,VARVAL( 2),DPROR,
  *           VARVAL( 3), KMAN,VARVAL( 4),KTANK,VARVAL( 5),LFLOW,
  *           VARVAL( 6),PCHMB,VARVAL( 7),TFLOW,VARVAL( 8),  VOL,
  *           VARVAL( 9),VOLMF
24  CONTINUE
    WRITE(*,*)' '
    WRITE(*,*)' Enter ? to print variable names & descriptions'
    WRITE(*,*)'       # to print variable names & values'
    WRITE(*,*)'       TITLE to enter new title'
    WRITE(*,*)'       END when all changes have been made'
    WRITE(*,*)' '
25  CONTINUE
    WRITE(*,'(A\)')'   Enter variable name and new value, END, ?, or
  * # '
    CALL ZREAD(NAME,VALUE)
    IF(NAME.EQ.'?')  GO TO 22
    IF(NAME.EQ.'#')  GO TO 23
    IF(NAME.EQ.'END'.OR.NAME.EQ.'end')  GO TO 28
    IF(NAME.EQ.'TITLE'.OR.NAME.EQ.'title')  THEN
     WRITE(*,'(A\)')' Enter new TITLE (20 characters max.) '
     READ(*,'(A)')TITLF
     WRITE(TITLE,10)TITLF,IHR,IMIN,AP,IMON,IDAY,IYR
     GO TO 25
    ENDIF
    DO 26 II=1,9
     I=II
     IF(NAME.EQ.VARU(I).OR.NAME.EQ.VARL(I))  GO TO 27
26  CONTINUE
    WRITE(*,*)'     Invalid name, try again'
    GO TO 22
27  CONTINUE
    IF(I.EQ. 1)  DENS=VALUE
    IF(I.EQ. 2)  DPROR=VALUE
    IF(I.EQ. 3)  KMAN=VALUE
    IF(I.EQ. 4)  KTANK=VALUE
    IF(I.EQ. 5)  LFLOW=VALUE
    IF(I.EQ. 6)  PCHMB=VALUE
    IF(I.EQ. 7)  TFLOW=VALUE
    IF(I.EQ. 8)  VOL=VALUE
    IF(I.EQ. 9)  VOLMF=VALUE
    GO TO 25
   ENDIF
28 CONTINUE
   A=SQRT(GRAV*KTANK/DENS)
   CTANK=DENS*VOL/KTANK
   CMAN=DENS*VOLMF/KMAN
   PMRAT=PCHMB/TFLOW
29 CONTINUE
   ICHG=0
   IF(RSPON.EQ.4)  GO TO 30
   WRITE(*,*)' Do you wish to change the pipe layout? '
   READ(*,'(A)')ANS
   IF(ANS.NE.'Y'.AND.ANS.NE.'y')  GO TO 36
   WRITE(*,*)' Do you wish to change all of the pipe segments?'
   READ(*,'(A)')ANS
   IF(ANS.NE.'Y'.AND.ANS.NE.'y')  THEN
    ICHG=1
    GO TO 30
   ENDIF
    SPLIT=1.0
    LOPEND=1
   WRITE(*,'(A\)')' How many segments is the pipe broken into? '
   READ(*,*)SEGMN
30 CONTINUE
   WRITE(12,*)'            NEW PIPE LAYOUT'
   WRITE(12,*)' STATUS   LENGTH          AREA          DIAMETER'
   I=0
   ISEGMN=SEGMN
   DO 35 II=1,SEGMN
    I=I+1
    IF(ICHG.EQ.1)  THEN
     IF(SECTN(I).EQ.0)  THEN
      WRITE(*,4)PIPE2(I),PIPE1(I)
     ELSEIF(SECTN(I).EQ.1)  THEN
      WRITE(*,5)PIPE2(I),PIPE1(I)
     ELSEIF(SECTN(I).EQ.2)  THEN
      WRITE(*,16)PIPE1(I),PIPE2(I)
     ELSEIF(SECTN(I).EQ.3)  THEN
      WRITE(*,13)PIPE1(I),PIPE2(I)
     ELSEIF(SECTN(I).EQ.4)  THEN
      WRITE(*,14)PIPE1(I),PIPE2(I),PIPE3(I)
     ELSEIF(SECTN(I).EQ.5)  THEN
      WRITE(*,15)PIPE1(I),PIPE2(I),PIPE3(I)
     ELSEIF(SECTN(I).EQ.6)  THEN
      WRITE(*,12)PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),PIPE5(I)
     ELSEIF(SECTN(I).EQ.9)  THEN
      WRITE(*,11)INT(PIPE3(I)),PIPE2(I),PIPE1(I)
     ENDIF
     WRITE(*,*)' You may keep (K), modify (Y), delete (D),',
  *            ' add before (B), or add after (A)?'
     READ(*,'(A)')ANS
     IF(ANS.EQ.'A'.OR.ANS.EQ.'a')  THEN
      I=I+1
      DO 31 III=ISEGMN,I,-1
       PIPE1(III+1)=PIPE1(III)
       PIPE2(III+1)=PIPE2(III)
       PIPE3(III+1)=PIPE3(III)
       PIPE4(III+1)=PIPE4(III)
       PIPE5(III+1)=PIPE5(III)
       L(III+1)=L(III)
       DIA(III+1)=DIA(III)
       AREA(III+1)=AREA(III)
       PCAP(III+1)=PCAP(III)
       PIND(III+1)=PIND(III)
       SECTN(III+1)=SECTN(III)
31 CONTINUE
      ISEGMN=ISEGMN+1
      GO TO 34
     ELSEIF(ANS.EQ.'B'.OR.ANS.EQ.'b')  THEN
      DO 32 III=ISEGMN,I,-1
       PIPE1(III+1)=PIPE1(III)
       PIPE2(III+1)=PIPE2(III)
       PIPE3(III+1)=PIPE3(III)
       PIPE4(III+1)=PIPE4(III)
       PIPE5(III+1)=PIPE5(III)
       L(III+1)=L(III)
       DIA(III+1)=DIA(III)
       AREA(III+1)=AREA(III)
       PCAP(III+1)=PCAP(III)
       PIND(III+1)=PIND(III)
       SECTN(III+1)=SECTN(III)
32 CONTINUE
      ISEGMN=ISEGMN+1
      GO TO 34
     ELSEIF(ANS.EQ.'D'.OR.ANS.EQ.'d')  THEN
      DO 33 III=I,ISEGMN
       PIPE1(III)=PIPE1(III+1)
       PIPE2(III)=PIPE2(III+1)
       PIPE3(III)=PIPE3(III+1)
       PIPE4(III)=PIPE4(III+1)
       PIPE5(III)=PIPE5(III+1)
       L(III)=L(III+1)
       DIA(III)=DIA(III+1)
       AREA(III)=AREA(III+1)
       PCAP(III)=PCAP(III+1)
       PIND(III)=PIND(III+1)
       SECTN(III)=SECTN(III+1)
33 CONTINUE
      I=I-1
      ISEGMN=ISEGMN-1
      GO TO 35
     ELSEIF(ANS.NE.'Y'.AND.ANS.NE.'y')  THEN
      GO TO 35
     ENDIF
    ENDIF
34 CONTINUE
    WRITE(*,*)' Specify 0 for BEND,           1 for STRAIGHT pipe,'
    WRITE(*,*)'         2 for INLINE ACCUM.,  3 for TUNED STUB,'
```

```
      WRITE(*,*)'          4 for HELMHOLTZ RES., 5 for PARALLEL RES.'
      WRITE(*,*)'          6 for PUMP,           9 for SPLIT'
      READ(*,*) SECT
      IF(SECT.LT.0.OR.SECT.GT.6.AND.SECT.NE.9)  GO TO 34
      SECTN(I)=SECT
      IF(SECT.EQ.0)  THEN
C            BEND IN PIPE
       WRITE(*,*)' RADIUS of bend along CL (ft), ANGLE of bend (deg),'
       WRITE(*,*)' DIAMETER (ft), and LENGTH (ft) beyond bend of pipe'
       READ(*,*)PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I)
       CALL BENDS(PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),VALUE,DIME)
       AREAB=0.785398*DIME**2
       L(I)=VALUE
       AREA(I)=AREAB
       DIA(I)=DIME
       PIPE5(I)=0.0
      ELSEIF(SECT.EQ.1)  THEN
C            STRAIGHT SECTION
       WRITE(*,*)' Specify LENGTH (ft) and DIAMETER (ft) of segment'
       READ(*,*) PIPE1(I),PIPE2(I)
       VALUE=PIPE1(I)
       DIME=PIPE2(I)
       PIPE3(I)=0.0
       PIPE4(I)=0.0
       PIPE5(I)=0.0
       AREAB=0.785398*DIME**2
       L(I)=VALUE
       AREA(I)=AREAB
       DIA(I)=DIME
      ELSEIF(SECT.EQ.2)  THEN
C            INLINE ACCUMULATOR
       WRITE(*,*)' Specify LENGTH (ft) & DIAMETER (ft) of accumulator '
       READ(*,*) PIPE1(I),PIPE2(I)
       L(I)=PIPE1(I)
       DIA(I)=PIPE2(I)
       AREA(I)=0.25*PI*PIPE2(I)**2
       PCAP(I)=DENS*0.785398*L(I)*DIA(I)**2*PMRAT/AVGK
       PIPE3(I)=0.0
       PIPE4(I)=0.0
       PIPE5(I)=0.0
      ELSEIF(SECT.EQ.3)  THEN
C            TUNED STUB ACCUMULATOR
       WRITE(*,*)' Specify LENGTH (ft) & DIAMETER (ft) of tuned stub'
       READ(*,*)PIPE1(I),PIPE2(I)
       L(I)=PIPE1(I)
       DIA(I)=PIPE2(I)
       AREA(I)=0.25*PI*PIPE2(I)**2
       PCAP(I)=DENS*L(I)*AREA(I)*PMRAT/AVGK
       PIND(I)=L(I)/(AREA(I)*GRAV*PMRAT)
       PIPE3(I)=0.0
       PIPE4(I)=0.0
       PIPE5(I)=0.0
       ELSEIF(SECT.EQ.4)  THEN
C             HELMHOLTZ RESONATOR ACCUMULATOR
        WRITE(*,*)' Specify LENGTH (ft), DIAMETER (ft) ,VOLUME (ft^3)',
     *            ' of Helmholtz Resonator'
        READ(*,*)PIPE1(I),PIPE2(I),PIPE3(I)
        L(I)=PIPE1(I)
        DIA(I)=PIPE2(I)
        AREA(I)=PIPE3(I)
        PCAP(I)=DENS*L(I)*AREA(I)*PMRAT/AVGK
        PIND(I)=L(I)/(0.25*PI*DIA(I)**2*GRAV*PMRAT)
        PIPE4(I)=0.0
        PIPE5(I)=0.0
       ELSEIF(SECT.EQ.5)  THEN
C             PARALLEL RESONATOR ACCUMULATOR
        WRITE(*,*)' Specify LENGTH (ft), DIAMETER (ft) ,VOLUME (ft^3)',
     *            ' of Parallel Resonator'
        READ(*,*)PIPE1(I),PIPE2(I),PIPE3(I)
        L(I)=PIPE1(I)
        DIA(I)=PIPE2(I)
        AREA(I)=PIPE3(I)
        PCAP(I)=DENS*L(I)*AREA(I)*PMRAT/AVGK
        PIND(I)=L(I)/(0.25*PI*DIA(I)**2*GRAV*PMRAT)
        PIPE4(I)=0.0
        PIPE5(I)=0.0
       ELSEIF(SECT.EQ.6)  THEN
C             PUMP
        WRITE(*,*)' Specify LENGTH (ft), DIAMETER (ft) ,dp/dm, CAP.',
     *            ' & IND. of pump'
        READ(*,*)PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),PIPE5(I)
        L(I)=PIPE1(I)
        DIA(I)=PIPE2(I)
        AREA(I)=PIPE3(I)
        PCAP(I)=PIPE4(I)*PMRAT
        PIND(I)=PIPE5(I)/PMRAT
       ELSEIF(SECTN(I).EQ.9)  THEN
C             SPLIT PIPE
        WRITE(*,*)' Specify LENGTH (ft), DIAMETER (ft), and no. of',
     *            ' segments'
        READ(*,*) PIPE1(I),PIPE2(I),PIPE3(I)
        VALUE=PIPE1(I)
        DIME=PIPE2(I)
        SPLIT=PIPE3(I)
        WRITE(*,'(A,I3)')' Maximun no. of iterations is set at ',LOPOLD
        WRITE(*,'(A\)')' Do you wish to change it? '
        READ(*,'(A)')ANS
        IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
         WRITE(*,'(A\)')' Enter maximum no. of iterations '
         READ(*,*)LOPOLD
        ENDIF
        LOPEND=LOPOLD
        AREAB=0.785398*DIME**2
        L(I)=VALUE
        AREA(I)=AREAB
        DIA(I)=DIME
        PIPE4(I)=0.0
        PIPE5(I)=0.0
       ENDIF
       WRITE(12,3)SECTN(I),L(I),AREA(I),DIA(I)
   35 CONTINUE
      SEGMN=ISEGMN
   36 CONTINUE
      WRITE(*,'(A\)')' Do you wish to save these changes? Y or N '
      READ(*,'(A)')ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  RETURN
      WRITE(*,'(A,A,A\)')' Do you wish to use file ',NAMLIN,'? Y or N '
      READ(*,'(A)')ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  THEN
       WRITE(*,'(A\)')' Enter name of file to use '
       READ(*,'(A)')NAMLIN
       CLOSE(UNIT=11)
       OPEN(UNIT=11,FILE=NAMLIN)
      ELSE
       WRITE(*,'(A,A,A\)')' Do you wish to rewind ',NAMLIN,'? Y or N '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  REWIND 11
      ENDIF
      WRITE(11,'(A)')TITLF
      WRITE(11,1)VOL
      WRITE(11,1)LFLOW
      WRITE(11,1)KTANK
      WRITE(11,1)DENS
      WRITE(11,1)TFLOW
      WRITE(11,1)VOLMF
      WRITE(11,1)KMAN
      WRITE(11,1)PCHMB
      WRITE(11,1)DPROR
      WRITE(11,2)SEGMN
      WRITE(11,2)(SECTN(I),PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),PIPE5(I),
     *            I=1,SEGMN)
      RETURN
      END
      SUBROUTINE NEXPT(WN,MAG1)
C       Supervises plot of admittance while computing
      COMMON /WCAPAS/IFRST
      REAL MAG1,X(2),Y(2)
      X(2)=WN
      Y(2)=MAG1
      IF(IFRST.NE.0)  CALL QTABL(1,2,X,Y)
      X(1)=WN
      Y(1)=MAG1
      IFRST=1
      RETURN
      END
      SUBROUTINE PIPPLOT(SEGMN,SECTN,PIPE1,PIPE2,PIPE3,PIPE4)
```

```
C       Supervises plot of piping layout
      COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      EXTERNAL XFUN,YFUN
      INTEGER*2 SEGMN,SECTN(75),ITYPE(200)
      REAL PIPE1(75),PIPE2(75),PIPE3(75),PIPE4(75)
      REAL POINT(8,200),XP(2),YP(2)
      ANG=0.0
      ANGLE=0.0
      COSA=1.0
      SINA=0.0
      X=0.0
      XH=0.0
      XL=0.0
      Y=0.0
      IF(SECTN(1).EQ.0)  THEN
       YH=Y+0.5*PIPE3(1)
       YL=Y-0.5*PIPE3(1)
      ELSEIF(SECTN(1).GE.3.AND.SECTN(1).LE.5)  THEN
       IF(SECTN(2).EQ.0)  THEN
        YH=Y+0.5*PIPE3(2)
        YL=Y-0.5*PIPE3(2)
       ELSE
        YH=Y+0.5*PIPE2(2)
        YL=Y-0.5*PIPE2(2)
       ENDIF
      ELSE
       YH=Y+0.5*PIPE2(1)
       YL=Y-0.5*PIPE2(1)
      ENDIF
      J=0
      XMIN=0.0
      XMAX=0.0
      YMIN=AMIN1(Y,YL,YH)
      YMAX=AMAX1(Y,YL,YH)
      DO 21 I=1,SEGMN
       IF(SECTN(I).EQ.0)  THEN
C         BEND
        CALL BNSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I))
       ELSEIF(SECTN(I).EQ.1.OR.SECTN(I).EQ.9)  THEN
C         STRAIGHT SECTION
        CALL STSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ELSEIF(SECTN(I).EQ.2)  THEN
C         INLINE ACCUMULATOR
        CALL STSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ELSEIF(SECTN(I).EQ.3)  THEN
C         TUNED STUB ACCUMULATOR
        CALL TSSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ELSEIF(SECTN(I).EQ.4)  THEN
C         HELMHOLTZ RESONATOR
        CALL HHSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I),PIPE3(I))
       ELSEIF(SECTN(I).EQ.5)  THEN
```

```
C         PARALLEL RESONATOR
        CALL PLSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I),PIPE3(I))
       ELSEIF(SECTN(I).EQ.6)  THEN
C         PUMP
        CALL STSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ENDIF
   21 CONTINUE
      XRANGE=XMAX-XMIN
      YRANGE=YMAX-YMIN
      XMIN=XMIN-0.05*XRANGE
      XMAX=XMAX+0.05*XRANGE
      YMIN=YMIN-0.05*YRANGE
      YMAX=YMAX+0.05*YRANGE
      CALL UPPERW(XMIN,YMIN,XMAX,YMAX)
      DO 24 I=1,J
       IF(ITYPE(I).EQ.0)  THEN
C         BEND
        XC=POINT(1,I)
        YC=POINT(2,I)
        X1=POINT(3,I)
        Y1=POINT(4,I)
        RAD=POINT(5,I)
        IF(X1.GT.Y1)  THEN
         X1=3.14159+X1
         Y1=3.14159+Y1
         CALL QCURV(XFUN,YFUN,Y1,X1)
        ELSE
         CALL QCURV(XFUN,YFUN,X1,Y1)
        ENDIF
       ELSE
C         ALL EXCEPT BEND
        X0=POINT(1,I)
        Y0=POINT(2,I)
        X1=POINT(3,I)
        Y1=POINT(4,I)
        X2=POINT(5,I)
        Y2=POINT(6,I)
        X3=POINT(7,I)
        Y3=POINT(8,I)
        XP(1)=X0
        YP(1)=Y0
        XP(2)=X1
        YP(2)=Y1
        CALL QTABL(1,2,XP,YP)
        XP(1)=X2
        YP(1)=Y2
        XP(2)=X3
        YP(2)=Y3
        CALL QTABL(1,2,XP,YP)
        XP(1)=X0
        YP(1)=Y0
        XP(2)=X2
        YP(2)=Y2
        CALL QTABL(1,2,XP,YP)
        XP(1)=X1
        YP(1)=Y1
        XP(2)=X3
        YP(2)=Y3
        CALL QTABL(1,2,XP,YP)
       ENDIF
   24 CONTINUE
      RETURN
      END
      SUBROUTINE PLOTSU(X,Y,Z,XF,YF,ZF,JPTS,IPTS,IXMAX,IYMAX)
C       Supervises the surface plot
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /FACTOR/SFAC
      INTEGER*4 IXMAX,IYMAX
      REAL XF(IXMAX),YF(IYMAX),ZF(IXMAX,IYMAX)
      REAL X(IPTS,JPTS),Y(IPTS,JPTS),Z(IPTS,JPTS)
      INTEGER*2 IWRK1(640),IWRK2(640)
      CHARACTER*1 ANS
      CHARACTER*45 LEGEND
      CHARACTER*58 LEGENDR,LEGENDH
      DATA LEGEND/'Pressure Transfer Function = f(freq,distance)'/
      DATA LEGENDR/'Pressure Transfer Function = f(freq(rad/sec),distanc
     *e(ft))'/
      DATA LEGENDH/' Pressure Transfer Function = f(freq(Hertz),distance
     *(ft)) '/
      DATA ASPECT/1.35/
      DATA ICOLR/4/,IFIL/3/,ILIN/1/
    1 FORMAT(' Current view is PHI =',F8.3,'   THETA =',F8.3)
    2 FORMAT(' Current BACKGROUD COLOR = ',I2,' LINE COLOR = ',I2,
     *       ' FILL COLOR = ',I2)
      CALL QRMODE(MODET,NCOLT)
      CALL QVIDBD(IBOARD)
      IF(IBOARD.LT.1.OR.IBOARD.GT.3)  THEN
       WRITE(*,*)' Graphics board not installed!'
       RETURN
      ENDIF
      IF(IBOARD.EQ.1)  MODE=6
      IF(IBOARD.EQ.2)  MODE=16
      IF(IBOARD.EQ.3)  MODE=18
      IWIRE=0
      IF(IBOARD.NE.1)  THEN
       WRITE(*,'(A\)')' Do you want a wire-frame drawing? '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  IWIRE=1
      ENDIF
      XMIN=XF(1)
      XMAX=XF(IPTS)
      YMIN=YF(1)
      YMAX=YF(JPTS)
      ZMIN=ZF(1,1)
      ZMAX=ZF(1,1)
      DO 20 J=1,JPTS
      DO 20 I=1,IPTS
       IF(ZMIN.GT.ZF(I,J))  ZMIN=ZF(I,J)
       IF(ZMAX.LT.ZF(I,J))  ZMAX=ZF(I,J)
   20 CONTINUE
      YLEN=YF(JPTS)-YF(1)
      XLEN=XF(IPTS)-XF(1)
      ZLEN=ZMAX-ZMIN
      XYZLEN=AMAX1(XLEN,YLEN,ZLEN)
      XFAC=XYZLEN/XLEN
      XINV=1.0/XFAC
      YFAC=XYZLEN/YLEN
      YINV=1.0/YFAC
      ZFAC=XYZLEN/ZLEN
      ZINV=1.0/ZFAC
      DO 21 J=1,JPTS
      DO 21 I=1,IPTS
       X(I,J)=XF(I)*XFAC
       Y(I,J)=YF(J)*YFAC
       Z(I,J)=ZF(I,J)*ZFAC
   21 CONTINUE
      XMIN=XMIN*XFAC
      XMAX=XMAX*XFAC
      YMIN=YMIN*YFAC
      YMAX=YMAX*YFAC
      ZMIN=ZMIN*ZFAC
      ZMAX=ZMAX*ZFAC
      XMAJ=0.2*(XMAX-XMIN)
      YMAJ=0.2*(YMAX-YMIN)
      ZMAJ=0.2*(ZMAX-ZMIN)
      P=-45.0
      T=30.0
      CALL Q3DROT(X,Y,Z,IPTS,JPTS,P,T)
   22 CONTINUE
      CALL QSMODE(MODE)
      IF(IBOARD.NE.1)  CALL QPREG(0,ICOLR)
      CALL WINDOW(MODE,ASPECT,XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX)
      CALL Q3DXAX(XMIN,XMAX,XMAJ,0,-1,2,YMIN,YMAX,ZMIN,XINV)
      CALL Q3DYAX(YMIN,YMAX,YMAJ,0,-1,2,XMAX,XMIN,ZMIN,YINV)
      CALL Q3DZAX(ZMIN,ZMAX,ZMAJ,0,-1,2,XMIN,YMIN,ZINV)
      IF(MODE.EQ.6)  THEN
       CALL QPTXT(40,TITLE,7,17,23)
       CALL QPTXT(45,LEGEND,7,15,22)
      ELSEIF(MODE.EQ.16)  THEN
       CALL QPTXT(40,TITLE,7,17,23)
       IF(SFAC.EQ.1.0)  THEN
        CALL QPTXT(58,LEGENDR,7,8,22)
       ELSE
        CALL QPTXT(58,LEGENDH,7,8,22)
       ENDIF
      ELSE
       CALL QPTXT(40,TITLE,7,17,27)
       IF(SFAC.EQ.1.0)  THEN
        CALL QPTXT(58,LEGENDR,7,8,26)
       ELSE
        CALL QPTXT(58,LEGENDH,7,8,26)
       ENDIF
      ENDIF
      IF(IBOARD.EQ.1.OR.IWIRE.EQ.1)  THEN
       CALL Q3DSTK(X,Y,IPTS,JPTS,IWRK1,IWRK2,640,1)
      ELSE
       CALL Q3DFIL(X,Y,IPTS,JPTS,IFIL,ILIN)
      ENDIF
   23 CONTINUE
      CALL QONKEY(IKEY)
      IF(IKEY.EQ.0)  GO TO 23
      CALL QINKEY(IEXTEN,IKEY)
      IF(IKEY.EQ.80.OR.IKEY.EQ.112)  CALL QPSCRN
      CALL QSMODE(MODET)
   25 CONTINUE
      IGO=0
      WRITE(*,1)P,T
      WRITE(*,'(A\)')' Do you wish another view? '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
       WRITE(*,'(A\)')'  Enter new viewing angles PHI & THETA. '
       READ(*,*)P,T
       CALL Q3DINV(X,Y,Z,IPTS,JPTS)
       CALL Q3DROT(X,Y,Z,IPTS,JPTS,P,T)
       IGO=1
      ENDIF
      IF(IBOARD.NE.1)  THEN
       WRITE(*,2)ICOLR,ILIN,IFIL
       WRITE(*,'(A\)')' Do you wish another color? '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
        WRITE(*,*)' Enter color number (0-63) for BACKGROUND, LINE,
     * and FILL '
        WRITE(*,*)'   4,1,3 will give the default colors '
        WRITE(*,'(A\)')'    0,7,0 will give black & white '
        READ(*,*)ICOLR,ILIN,IFIL
        IGO=1
       ENDIF
       IWR=0
       IF(IWIRE.EQ.0)  THEN
        WRITE(*,'(A\)')' Do you want a wire-frame drawing? '
        READ(*,'(A)')ANS
        IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
         IWR=1
         IGO=1
        ENDIF
       ELSE
        WRITE(*,'(A\)')' Do you want a filled drawing? '
        READ(*,'(A)')ANS
        IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
         IWR=2
         IGO=1
        ENDIF
       ENDIF
       IF(IWR.EQ.1)  IWIRE=1
       IF(IWR.EQ.2)  IWIRE=0
      ENDIF
      IF(IGO.NE.0)  GO TO 22
      RETURN
      END
      SUBROUTINE PLSECT(J,ITYPE,POINT,LEN,DIA,VOL)
C       Computes plot coordinates for parallel resonator
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
      REAL LEN,POINT(8,200)
      INTEGER*2 ITYPE(200)
      XOLD=X
      XHOLD=XH
      XLOLD=XL
      YOLD=Y
      YHOLD=YH
      YLOLD=YL
      ANGOLD=ANG
      ANGSAV=ANGLE
      SINOLD=SINA
      COSOLD=COSA
      DIAM=SQRT((XH-XL)**2+(YH-YL)**2)
      CALL STSECT(J,ITYPE,POINT,DIA,DIAM)
      XC=0.5*(XHOLD+XH)
      XHC=XHOLD
      XLC=XL
      YC=0.5*(YHOLD+YH)
      YHC=YHOLD
      YLC=YL
      PLEN=LEN-2.0*DIA
      PDIA=(VOL-2.0*DIA*DIAM)/PLEN
      CALL STSECT(J,ITYPE,POINT,PLEN,PDIA)
      CALL STSECT(J,ITYPE,POINT,DIA,DIAM)
      XSAV=X
      XHSAV=XH
      XLSAV=XL
      YSAV=Y
      YHSAV=YH
      YLSAV=YL
      SINA=COSOLD
      COSA=-SINOLD
      RADIUS=DIA
      TURN=-90.0
      SIDE=LEN-5.0*DIA
      ANG=ANG+1.5708
      ANGLE=ANGLE+90.0
      X=XC
      Y=YC
      XH=XHC
      XL=XLC
      YH=YHC
      YL=YLC
      CALL BNSECT(J,ITYPE,POINT,RADIUS,TURN,DIA,DIA)
      CALL STSECT(J,ITYPE,POINT,SIDE,DIA)
      CALL BNSECT(J,ITYPE,POINT,RADIUS,TURN,DIA,DIA)
      X=XSAV
      Y=YSAV
      XH=XHSAV
      XL=XLSAV
      YH=YHSAV
      YL=YLSAV
      ANG=ANGOLD
      ANGLE=ANGSAV
      SINA=SINOLD
      COSA=COSOLD
      RETURN
      END
      SUBROUTINE PLTCON(X,Y,Z,XF,YF,ZF,JPTS,IPTS,IXMAX,IYMAX)
C       Supervises plot of contour plot
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /FACTOR/SFAC
      INTEGER*4 IXMAX,IYMAX
      REAL XF(IXMAX),YF(IYMAX),ZF(IXMAX,IYMAX)
      REAL X(IPTS),Y(JPTS),Z(IPTS,JPTS),CONS(10)
      INTEGER*2 LABL(10)
      DATA ASPECT/1.35/
      DATA LABL/1,0,0,0,1,0,0,0,1,0/
      DATA ICOLR/4/,IFIL/3/,ILIN/1/
    2 FORMAT(' Current BACKGROUD COLOR = ',I2,' LINE COLOR = ',I2,
     *       ' FILL COLOR = ',I2)
      CALL QRMODE(MODET,NCOLT)
      CALL QVIDBD(IBOARD)
      IF(IBOARD.LT.1.OR.IBOARD.GT.3)  THEN
       WRITE(*,*)' Graphics board not installed!'
       RETURN
      ENDIF
      IF(IBOARD.EQ.1)  MODE=6
      IF(IBOARD.EQ.2)  MODE=16
      IF(IBOARD.EQ.3)  MODE=18
      XMIN=XF(1)
      XMAX=XF(IPTS)
      YMIN=YF(1)
      YMAX=YF(JPTS)
      ZMIN=ZF(1,1)
      ZMAX=ZF(1,1)
      DO 21 J=1,JPTS
       Y(J)=YF(J)
      DO 21 I=1,IPTS
       IF(J.EQ.1)  X(I)=XF(I)
       Z(I,J)=ZF(I,J)
       IF(ZMIN.GT.Z(I,J))  ZMIN=Z(I,J)
       IF(ZMAX.LT.Z(I,J))  ZMAX=Z(I,J)
   21 CONTINUE
      ZLEN=0.1*(ZMAX-ZMIN)
      DO 22 I=1,9
       CONS(I)=I*ZLEN
   22 CONTINUE
      XMAJ=0.2*(XMAX-XMIN)
      YMAJ=0.2*(YMAX-YMIN)
   20 CONTINUE
      CALL QSMODE(MODE)
      IDEF=2
      IF(IBOARD.NE.1)  THEN
       IDEF=2
       CALL QPREG(0,ICOLR)
      ENDIF
      CALL QCTRDE(MODE,ILIN,IFIL,ILIN,1)
      JCOL1=100
      JCOL2=450
      JROW1=40
      IF(MODE.EQ.6)  JROW1=60
      JROW2=169
      IF(MODE.EQ.16)  JROW2=319
      IF(MODE.EQ.18)  JROW2=409
      XORG=XMIN
      YORG=YMIN
      YOVERX=1.0
      IOPT=0
      IF(MODE.NE.18)  THEN
       CALL QPTXT(40,TITLE,7,17,23)
      ELSE
       CALL QPTXT(40,TITLE,7,17,27)
      ENDIF
      CALL QPLOT(JCOL1,JCOL2,JROW1,JROW2,XMIN,XMAX,YMIN,YMAX,
     *           XORG,YORG,IOPT,YOVERX,ASPECT)
      CALL QXAXIS(XMIN,XMAX,XMAJ,0,-1,2)
      CALL QYAXIS(YMIN,YMAX,YMAJ,0,-1,2)
      IF(SFAC.EQ.1)  THEN
       CALL QPTXTA(17,'Frequency-rad/sec',7)
      ELSE
       CALL QPTXTA(17,' Frequency-Hertz ',7)
      ENDIF
      CALL QPTXTD(7,'X - ft.',7)
      CALL QCNTOU(ASPECT,X,Y,Z,CONS,LABL,IPTS,JPTS,9,IDEF)
   23 CONTINUE
      CALL QONKEY(IKEY)
      IF(IKEY.EQ.0)  GO TO 23
      CALL QINKEY(IEXTEN,IKEY)
      IF(IKEY.EQ.80.OR.IKEY.EQ.112)  CALL QPSCRN
      CALL QSMODE(MODET)
      IF(IBOARD.NE.1)  THEN
       WRITE(*,2)ICOLR,ILIN,IFIL
       WRITE(*,'(A\)')' Do you wish another color? '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
        WRITE(*,*)' Enter color number (0-63) for BACKGROUND, LINE,
     * and FILL '
        WRITE(*,*)'   4,1,3 will give the default colors '
        WRITE(*,'(A\)')'    0,7,7 will give black & white '
        READ(*,*)ICOLR,ILIN,IFIL
        GO TO 20
       ENDIF
      ENDIF
   25 CONTINUE
      RETURN
      END
      SUBROUTINE SETPLT
C       Sets up the plot environment
      COMMON /WCAPAS/IFRST
      COMMON /NOCOL/MODE,MODET,NTROWS,NTCOLS,NPROWS,NPCOLS
      COMMON /ADMCOL/ADMBAC,ADMLIN
      INTEGER ADMBAC,ADMLIN
      CHARACTER*1 ANS
      DATA ITIM/0/
      IF(ITIM.EQ.0)  THEN
       ITIM=1
       ADMBAC=4
       ADMLIN=1
      ENDIF
      CALL QRMODE(MODET,NCOLT)
      CALL QVIDBD(IBOARD)
      IF(IBOARD.LT.1.OR.IBOARD.GT.3)  THEN
       WRITE(*,*)' Graphics board not installed!'
       RETURN
      ENDIF
      IF(IBOARD.EQ.1)  THEN
       MODE=6
       NPROWS=200
       NTROWS=25
      ENDIF
      IF(IBOARD.EQ.2)  THEN
       MODE=16
       NPROWS=350
       NTROWS=25
      ENDIF
      IF(IBOARD.EQ.3)  THEN
       MODE=18
       NPROWS=480
       NTROWS=25
      ENDIF
      IFRST=0
      NTCOLS=NCOLT
      NPCOLS=640
      IF(MODE.NE.6)  THEN
       WRITE(*,'(A\)')' Do you wish change colors of admittance? '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
        WRITE(*,*)' Enter no. of background color and no. of line color'
        WRITE(*,*)'   4,1 will give the default colors '
        WRITE(*,'(A\)')'    0,7 will give black & white '
        READ(*,*)ADMBAC,ADMLIN
       ENDIF
      ENDIF
      CALL QSMODE(MODE)
      RETURN
      END
      SUBROUTINE STSECT(J,ITYPE,POINT,LEN,DIA)
C       Computes plot coordinates for a straight section
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      REAL LEN,POINT(8,200)
      INTEGER*2 ITYPE(200)
      J=J+1
      ITYPE(J)=1
      XH=X-0.5*SINA*DIA
      XL=X+0.5*SINA*DIA
      YH=Y+0.5*COSA*DIA
      YL=Y-0.5*COSA*DIA
      POINT(1,J)=XH
      POINT(2,J)=YH
      POINT(3,J)=XL
      POINT(4,J)=YL
      X=X+COSA*LEN
      XH=X-0.5*SINA*DIA
      XL=X+0.5*SINA*DIA
      Y=Y+SINA*LEN
      YH=Y+0.5*COSA*DIA
      YL=Y-0.5*COSA*DIA
      POINT(5,J)=XH
      POINT(6,J)=YH
      POINT(7,J)=XL
      POINT(8,J)=YL
      XMIN=AMIN1(X,XL,XH,XMIN)
      XMAX=AMAX1(X,XL,XH,XMAX)
      YMIN=AMIN1(Y,YL,YH,YMIN)
      YMAX=AMAX1(Y,YL,YH,YMAX)
      RETURN
      END
      SUBROUTINE TSSECT(J,ITYPE,POINT,LEN,DIA)
C       Computes plot coordinates for a tuned stub
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      REAL LEN,POINT(8,200)
      INTEGER*2 ITYPE(200)
      J=J+1
      ITYPE(J)=1
      DIAM=SQRT((XH-XL)**2+(YH-YL)**2)
      XH=X-SINA*(LEN+0.5*DIAM)
      YH=Y+COSA*(LEN+0.5*DIAM)
      POINT(1,J)=XH
      POINT(2,J)=YH
      POINT(3,J)=XL
      POINT(4,J)=YL
      X=X+COSA*DIA
      XH=X-SINA*(LEN+0.5*DIAM)
      XL=XL+COSA*DIA
      Y=Y+SINA*DIA
      YH=Y+COSA*(LEN+0.5*DIAM)
      YL=YL+SINA*DIA
      POINT(5,J)=XH
      POINT(6,J)=YH
      POINT(7,J)=XL
      POINT(8,J)=YL
      XMIN=AMIN1(X,XL,XH,XMIN)
      XMAX=AMAX1(X,XL,XH,XMAX)
      YMIN=AMIN1(Y,YL,YH,YMIN)
      YMAX=AMAX1(Y,YL,YH,YMAX)
      RETURN
      END
      SUBROUTINE UPPERW(X0,Y0,X1,Y1)
C       Sets up upper plotting window
      COMMON /NOCOL/MODE,MODET,NTROWS,NTCOLS,NPROWS,NPCOLS
      COMMON /ADMCOL/ADMBAC,ADMLIN
      INTEGER ADMBAC,ADMLIN
      XMIN=X0
      XMAX=X1
      YMIN=Y0
      YMAX=Y1
      JCOL1=100
      JCOL2=550
      IF(MODE.EQ.6)  THEN
       JROW1=100
       JROW2=179
      ELSEIF(MODE.EQ.16)  THEN
       JROW1=214
       JROW2=309
      ELSEIF(MODE.EQ.18)  THEN
       JROW1=244
       JROW2=449
      ENDIF
      XORG=XMIN
      YORG=YMIN
      YOVERX=1.0
      IOPT=1
      ASPECT=1.35
      YMAXO=YMAX
      CALL QPLOT(JCOL1,JCOL2,JROW1,JROW2,XMIN,XMAX,YMIN,YMAX,
     *           XORG,YORG,IOPT,YOVERX,ASPECT)
      IF(IOPT.GE.0)  GO TO 21
      IOPT=1
      CHANGE=(YMAX-YMIN)/(YMAXO-YMIN)
      JCOL2=JCOL1+0.98*CHANGE*(JCOL2-JCOL1)
      YMAX=YMAXO
      CALL QPLOT(JCOL1,JCOL2,JROW1,JROW2,XMIN,XMAX,YMIN,YMAX,
     *           XORG,YORG,IOPT,YOVERX,ASPECT)
   21 CONTINUE
      IF(MODE.NE.6)  THEN
       CALL QPREG(0,ADMBAC)
      ENDIF
       CALL QSETUP(0,ADMLIN,-2,ADMLIN)
      IF(MODE.NE.18)  THEN
       CALL QPTXT(11,'Pipe Layout',7,35,23)
      ELSE
       CALL QPTXT(11,'Pipe Layout',7,35,27)
      ENDIF
      RETURN
      END
      SUBROUTINE WINDOW(MODE,XSCALE,XST,XFIN,YST,YFIN,ZST,ZFIN)
C       Sets up window for surface plot
      CALL Q3DWIN(XST,XFIN,YST,YFIN,ZST,ZFIN,XMIN,XMAX,YMIN,YMAX)
      JCOL1=100
      JCOL2=450
      JROW1=40
      JROW2=169
      IF(MODE.EQ.16)  JROW2=319
      IF(MODE.EQ.18)  JROW2=409
      XORG=XMIN
      YORG=YMIN
      YOVERX=1.0
      IOPT=0
      ASPECT=XSCALE
      CALL QPLOT(JCOL1,JCOL2,JROW1,JROW2,XMIN,XMAX,YMIN,YMAX,
     *           XORG,YORG,IOPT,YOVERX,ASPECT)
      RETURN
      END
      FUNCTION XFUN(T)
C       Parametric function for plotting of bends
      COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
      XFUN=XC+RAD*SIN(T)
      RETURN
      END
      FUNCTION YFUN(T)
C       Parametric function for plotting of bends
      COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
      YFUN=YC-RAD*COS(T)
      RETURN
      END
      SUBROUTINE ZREAD(NAME,VALUE)
C       Reads input for input modification
      CHARACTER*1 NAME(8)
      CHARACTER*1 CARD(80),PLUS,MINUS,PERIOD,LE,E,NUMBER(10)
      CHARACTER*1 LEND(3),CEND(3),POUND,QUEST,BLK,COMMA
      CHARACTER*1 LTIT(5),CTIT(5)
      CHARACTER*80 DCARD
      EQUIVALENCE (CARD(1),DCARD)
      DATA PLUS/'+'/,MINUS/'-'/,PERIOD/'.'/,LE/'e'/,E/'E'/,BLK/' '/
      DATA NUMBER/'0','1','2','3','4','5','6','7','8','9'/,COMMA/','/
      DATA LEND/'e','n','d'/,CEND/'E','N','D'/,POUND/'#'/,QUEST/'?'/
      DATA LTIT/'t','i','t','l','e'/,CTIT/'T','I','T','L','E'/
    1 FORMAT(A)
      DO 21 I=1,8
       NAME(I)=BLK
   21 CONTINUE
      READ(*,1)DCARD
      IF(CARD(1).EQ.POUND)  THEN
       NAME(1)=POUND
       RETURN
      ENDIF
      IF(CARD(1).EQ.QUEST)  THEN
       NAME(1)=QUEST
       RETURN
      ENDIF
      DO 22 I=1,3
       IF(CARD(I).NE.LEND(I).AND.CARD(I).NE.CEND(I))  GO TO 220
       NAME(I)=CEND(I)
   22 CONTINUE
      RETURN
  220 CONTINUE
      DO 221 I=1,5
       IF(CARD(I).NE.LTIT(I).AND.CARD(I).NE.CTIT(I))  GO TO 23
       NAME(I)=CTIT(I)
  221 CONTINUE
      RETURN
   23 CONTINUE
      DO 24 I=1,8
       II=I
       IF(CARD(I).EQ.BLK.OR.CARD(I).EQ.COMMA)  GO TO 25
       NAME(I)=CARD(I)
   24 CONTINUE
   25 CONTINUE
      DO 26 I=II,80
       ID=I
       IF(CARD(I).NE.BLK.AND.CARD(I).NE.COMMA)  GO TO 27
```

```
26 CONTINUE
   VALUE=0.0
   WRITE(*,*)'   No value given, ZERO assumed'
   RETURN
27 CONTINUE
   SIGN=1.0
   IF(CARD(ID).EQ.MINUS)  THEN
    SIGN=-1.0
    ID=ID+1
   ELSEIF(CARD(ID).EQ.PLUS)  THEN
    ID=ID+1
   ENDIF
   WHOLE=0.0
   DO 30 I=ID,80
    II=I
    IF(CARD(I).EQ.PERIOD)  GO TO 31
    IF(CARD(I).EQ.PLUS)  GO TO 36
    IF(CARD(I).EQ.MINUS)  GO TO 36
    IF(CARD(I).EQ.E.OR.CARD(I).EQ.LE)  GO TO 35
    DO 28 J=1,10
     JJ=J-1
     IF(CARD(I).EQ.NUMBER(J))  GO TO 29
28 CONTINUE
    VALUE=SIGN*WHOLE
    IF(CARD(I).EQ.BLK)  RETURN
    WRITE(*,*)'  Input error, value set to ZERO'
    VALUE=0.0
    RETURN
29 CONTINUE
    WHOLE=WHOLE*10.0+JJ
30 CONTINUE
   VALUE=SIGN*WHOLE
   RETURN
31 CONTINUE
   ID=II+1
   FRACT=0.0
   ICOUNT=0
   DO 34 I=ID,80
    ICOUNT=ICOUNT+1
    II=I
    IF(CARD(I).EQ.PERIOD)  THEN
     WRITE(*,*)'  Input error, value set to ZERO'
     VALUE=0.0
     RETURN
    ENDIF
    IF(CARD(I).EQ.PLUS)  GO TO 36
    IF(CARD(I).EQ.MINUS)  GO TO 36
    IF(CARD(I).EQ.E.OR.CARD(I).EQ.LE)  GO TO 35
    DO 32 J=1,10
     JJ=J-1
     IF(CARD(I).EQ.NUMBER(J))  GO TO 33
32 CONTINUE
```

```
    VALUE=SIGN*(WHOLE+FRACT)
    IF(CARD(I).EQ.BLK)  RETURN
    WRITE(*,*)'  Input error, value set to ZERO'
    VALUE=0.0
    RETURN
33 CONTINUE
    FRACT=FRACT+JJ/10.0**ICOUNT
34 CONTINUE
   VALUE=SIGN*(WHOLE+FRACT)
   RETURN
35 CONTINUE
   II=II+1
36 CONTINUE
   VALUE=SIGN*(WHOLE+FRACT)
   SIGN=1.0
   IF(CARD(II).EQ.MINUS)  THEN
    SIGN=-1.0
    II=II+1
   ELSEIF(CARD(II).EQ.PLUS)  THEN
    II=II+1
   ENDIF
   WHOLE=0.0
   DO 39 I=II,80
    DO 37 J=1,10
     JJ=J-1
     IF(CARD(I).EQ.NUMBER(J))  GO TO 38
37 CONTINUE
    VALUE=VALUE*10.0**(SIGN*WHOLE)
    IF(CARD(I).EQ.BLK)  RETURN
    WRITE(*,*)'  Input error, value set to ZERO'
    VALUE=0.0
    RETURN
38 CONTINUE
    WHOLE=WHOLE*10.0+JJ
39 CONTINUE
   VALUE=VALUE*10.0**(SIGN*WHOLE)
   RETURN
   END
```

Appendix C

Listing of Nyquist Program

NYQ

```
C
C     PROGRAM NYQ
C
C         Program to calculate fuel and lox lines admittance
C         as input to routines for a Nyquist plot
C
C
C                       Variables in Commons
C
C                             BLANK
C   SCREEN          CHAR*22    screen atributes for plotting
C
C                             /ARCCON/
C   XC              REAL*4       x coordinate of curve center
C   YC              REAL*4       y coordinate of curve center
C   RAD             REAL*4     radius of bend
C   ANG             REAL*4     angle of bend in radians
C   ANGLE           REAL*4     angle of bend in degrees
C
C                             /FACTOR/
C   SFAC            REAL*4     factor for frequency
C
C                             /NOCOL/
C   NCOLS           INTEGER*2  number of text columns
C   NMODE           INTEGER*2  graphics mode
C
C                             /PIPPXY/
C   X               REAL*4     x location of current centerline
C   XH              REAL*4     x location of current upper pipe
C   XL              REAL*4     x location of current lower pipe
C   Y               REAL*4     y location of current centerline
C   YH              REAL*4     y location of current upper pipe
C   YL              REAL*4     y location of current lower pipe
C   XMIN            REAL*4     minimum x value of piping layout
C   XMAX            REAL*4     maximum x value of piping layout
C   YMIN            REAL*4     minimum y value of piping layout
C   YMAX            REAL*4     maximum y value of piping layout
C   SINA            REAL*4     sine of current pipe direction
C   COSA            REAL*4     cosine of current pipe direction
C
C                             /WCAOUT/
C   NAMLIN(2)       CHAR*24    name of files containing pipe description
C   IUNIT           INTEGER*2  unit number of current file (fuel or lox)
C
C                             /WCAPAS/
C   IFRST           INTEGER*2  flag for admittance plot
C
C                             /WCATIT/
C   TITLE           CHAR*40    title for plots
C   TITLF           CHAR*20    title from pipe file
C   IHR             INTEGER*2  hour code run
C   IMIN            INTEGER*2  minute code run
```

```
C   AP              CHAR*2      AM or PM
C   IYR             INTEGER*2   yesr code run
C   IMON            INTEGER*2   month code run
C   IDAY            INTEGER*2   day code run
C
C                                         /WORKIT/
C   WORK(12)        REAL*4      EQUIVALENCE(WORK(1),A)
C   A               REAL*4      speed of sound in the fluid (ft/sec)
C   CMAN            REAL*4      manifold capacitance
C   CTANK           REAL*4      tank capacitance
C   DENS            REAL*4      density of fluid (lbm/ft^3)
C   LFLOW           REAL*4      flow rate through pipe (lbm/sec)
C   KTANK           REAL*4      bulk modulus of tank (lbf/ft^2)
C   KMAN            REAL*4      bulk modulus of manifold (lbf/ft^2)
C   TFLOW           REAL*4      total flow rate of engine (lbm/sec)
C   VOL             REAL*4      volume of tank (ft^3)
C   VOLMF           REAL*4      volume of manifold (ft^3)
C   PCHMB           REAL*4      chamber pressure (lbf/ft^2)
C   DPROR           REAL*4      pressure drop across orfices (lbf/ft^2)
C
C
C   PROGRAM NYQ
C       Logic portion of code
C
C   Commons FACTOR  NOCOL   WCAOUT  WCATIT
C                   Local Variables
C   AM              CHAR*2      'AM'
C   ANS             CHAR*1      response to question
C   CHOICE          INTEGER*2   flag for type plot requested
C   CSTAR           REAL*4      characteristic rocket velocity (ft/sec)
C   DCDR            REAL*4      change in velocity with mixture ratio (ft/sec)
C   GF              COMPLEX*8   admittance of fuel line looking toward tank
C   GOX             COMPLEX*8   admittance of lox line looking toward tank
C   HFREQ           REAL*4      maximum frequency requested
C   IFUEL           INTEGER*2   flag indicating presence of fuel line
C   IGONE           INTEGER*2   flag for FUEL & LOX routines
C   ILOX            INTEGER*2   flag indicating presence of lox line
C   ISEC            INTEGER*2   second code run
C   I100            INTEGER*2   hundredth of second code run
C   K               INTEGER*2   do loop index
C   KW(1001)        REAL*4      frequency array
C   K1C(1001)       REAL*4      complex part of K(jw)
C   K1R(1001)       REAL*4      real part of K(jw)
C   K2C(1001)       REAL*4      complex part of K(jw,Gox)
C   K2R(1001)       REAL*4      real part of K(jw,Gox)
C   K3C(1001)       REAL*4      complex part of K(jw,Gf)
C   K3R(1001)       REAL*4      real part of K(jw,Gf)
C   K4C(1001)       REAL*4      complex part of K(jw,Gox,Gf)
C   K4R(1001)       REAL*4      real part of K(jw,Gox,Gf)
C   LFREQ           REAL*4      minimum frequency requested
C   NPTS            INTEGER*2   intermediate variable
C   PM              CHAR*2      'PM'
```

```
C    PTS               INTEGER*2  number of frequencies
C    PIPEA1(75)        REAL*4     first parameter of fuel pipe description
C    PIPEA2(75)        REAL*4     second parameter of fuel pipe description
C    PIPEA3(75)        REAL*4     third parameter of fuel pipe description
C    PIPEA4(75)        REAL*4     fourth parameter of fuel pipe description
C    PIPEB1(75)        REAL*4     first parameter of lox pipe description
C    PIPEB2(75)        REAL*4     second parameter of lox pipe description
C    PIPEB3(75)        REAL*4     third parameter of lox pipe description
C    PIPEB4(75)        REAL*4     fourth parameter of lox pipe description
C    RBAR              REAL*4     mixture ratio
C    S                 COMPLEX*8  complex frequency
C    SECTNA(75)        INTEGER*2  fuel pipe section types
C    SECTNB(75)        INTEGER*2  lox pipe section types
C    SEGMNA            INTEGER*2  number of fuel pipe sections
C    SEGMNB            INTEGER*2  number of lox pipe sections
C    SSIZE1            REAL*4     parameter to pack frequencies toward low end
C    SSIZE2            REAL*4     parameter to pack frequencies toward low end
C    SSIZE3            REAL*4     parameter to pack frequencies toward low end
C    TAUT              REAL*4     transport lag (sec)
C    THETAC            REAL*4     characteristic time constant (sec)
C    VARI              CHAR*24    name of input file
C    W                 REAL*4     oscillatory part of frequency
C
C
C    SUBROUTINE ADMIT(S,GADM,A,AREA,CMAN,CTANK,DPROR,L,LFLOW,PMRAT,
C                      SEGMN,SECTN,SPLIT,LOPEND,PCAP,PIND)
C        determines admittance looking toward tank
C
C    Common  WCATIT
C                     Variables in Argument List
C    A                 REAL*4     speed of sound in the fluid (ft/sec)
C    AREA(75)          REAL*4     area of pipe section (ft^2)
C    CMAN              REAL*4     manifold capacitance
C    CTANK             REAL*4     tank capacitance
C    DPROR             REAL*4     pressure drop across orfices (lbf/ft^2)
C    GADM              COMPLEX*8  admittance of line looking toward tank
C    L(75)             REAL*4     length of pipe section (ft)
C    LFLOW             REAL*4     flow rate through pipe (lbm/sec)
C    LOPEND            INTEGER*2  maximum number of iterations for split pipe
C    PCAP(75)          REAL*4     capacitance of pipe section
C    PIND(75)          REAL*4     inductance of pipe section
C    PMRAT             REAL*4     chamber pressure/total mass flow
C    S                 COMPLEX*8  complex frequency
C    SECTN(75)         INTEGER*2  pipe section types
C    SEGMN             INTEGER*2  number of pipe sections
C    SPLIT             REAL*4     number of lines from pipe split
C                     Local Variables
C    CAPM              COMPLEX*8  intermediate variable
C    CAPN              COMPLEX*8  intermediate variable
C    CFAC              COMPLEX*8  intermediate variable
C    ERRP              REAL*4     error in gain calculation
C    G(0:75)           COMPLEX*8  admittance looking toward tank
```

```
C   GDIF             REAL*4     distance between new and old admittance
C   GOLD(0:75)       COMPLEX*8  previous admittance calculated
C   GRAV             REAL*4     gravitational constant (lbm-ft/lbf-sec^2)
C   G1               COMPLEX*8  admittance starting at G(0)+1
C   I                INTEGER*2  do loop index
C   IOPEN            INTEGER*2  flag indicating if SURF.ERR is open
C   KLOOP            INTEGER*2  do loop index
C   RHS              COMPLEX*8  intermediate variable
C   TL               REAL*4     length/speed of sound
C   ZG(75)           COMPLEX*8  impedance looking toward engine
C   ZGEFF            COMPLEX*8  effective impedance for calculations
C   ZOEFF            COMPLEX*8  effective ZO for calculations
C   ZO(75)           REAL*4     characteristic impedance
C   ZOR              REAL*4     intermediate variable
C   ZT(0:75)         COMPLEX*8  impedance looking toward tank
C   ZTOP             REAL*4     intermediate variable
C
C
C   SUBROUTINE ALLPT(WHOLD,GHOLD,PTS,ITYPE)
C       Supervises Nyquist plot
C
C                   Variables in Argument List
C   GHOLD(1001)      REAL*4     imaginary part of K()
C   ITYPE            INTEGER*2  which K()
C   PTS              INTEGER*2  number of values to plot
C   WHOLD(1001)      REAL*4     real part of K()
C                   Local Variables
C   DUMWIL           INTEGER*2  intermediate variable
C   I                INTEGER*2  do loop index
C   IMAX             REAL*8     maximum value of complex part
C   IMMIN            REAL*8     minimum value of complex part
C   RMAX             REAL*8     maximum value of real part
C   RMIN             REAL*8     minimum value of real part
C   X                REAL*8     x value of point to be plotted
C   XY               CHAR*16    intermediate variable
C   Y                REAL*8     y value of point to be plotted
C
C
C   SUBROUTINE BENDS(PIPE1,PIPE2,PIPE3,PIPE4,VALUE,DIME)
C       Computes effective straight pipe for bend
C
C                   Variables in Argument List
C   DIME             REAL*4     effective diameter (ft)
C   PIPE1            REAL*4     radius of bend (ft)
C   PIPE2            REAL*4     angle of bend (degrees)
C   PIPE3            REAL*4     diameter of bend (ft)
C   PIPE4            REAL*4     length of end straight segments (ft)
C   VALUE            REAL*4     effective length (ft)
C                   Local Variables
C   ARBND            REAL*4     area of bend
C   AREAB            REAL*4     effective area of bend
C   BENDR            REAL*4     bend angle in radians
```

```
C   GAMMA            REAL*4     intermediate variable
C   INERT            REAL*4     intermediate variable
C   INRAD            REAL*4     inside radius of bend
C   LBEND            REAL*4     intermediate variable
C   LPRME            REAL*4     intermediate variable
C   NEWLN            REAL*4     intermediate variable
C   OTRAD            REAL*4     outside radius of bend
C   RATIO            REAL*4     intermediate variable
C   X                REAL*4     intermediate variable
C   Y                REAL*4     intermediate variable
C
C
C   SUBROUTINE BNSECT(J,ITYPE,POINT,PIPE1,PIPE2,PIPE3,PIPE4)
C       Computes plot coordinates for a bend
C
C   Commons ARCCON  PIPPXY
C                   Variables in Argument List
C   ITYPE(200)       INTEGER*2  type plot element
C   J                INTEGER*2  pointer to element
C   PIPE1            REAL*4     first parameter of pipe description
C   PIPE2            REAL*4     second parameter of pipe description
C   PIPE3            REAL*4     third parameter of pipe description
C   PIPE4            REAL*4     fourth parameter of pipe description
C   POINT(8,200)     REAL*4     description of plot element
C                   Local Variables
C   DIA              REAL*4     intermediate variable
C   HOLD             REAL*4     intermediate variable
C   RANG             REAL*4     intermediate variable
C   SLENTH           REAL*4     intermediate variable
C   X0               REAL*4     intermediate variable
C   X1               REAL*4     intermediate variable
C   X2               REAL*4     intermediate variable
C   X3               REAL*4     intermediate variable
C   Y0               REAL*4     intermediate variable
C   Y1               REAL*4     intermediate variable
C   Y2               REAL*4     intermediate variable
C   Y3               REAL*4     intermediate variable
C
C
C   COMPLEX FUNCTION CCOSH(S)
C       Evaluates the complex hyperbolic cosine
C
C                   Variable  in Argument List
C   S                COMPLEX*8  complex frequency
C                   Local Variables
C   COSHI            REAL*4     intermediate variable
C   COSHR            REAL*4     intermediate variable
C   LAMDA            REAL*4     real part of complex frequency
C   MU               REAL*4     imaginary part of complex frequency
C
C
C   COMPLEX FUNCTION CSINH(S)
```

```
C       Evaluates the complex hyperbolic sine
C
C                       Variable  in Argument List
C   S                    COMPLEX*8  complex frequency
C                       Local Variables
C   LAMDA                REAL*4     real part of complex frequency
C   MU                   REAL*4     imaginary part of complex frequency
C   SINHI                REAL*4     intermediate variable
C   SINHR                REAL*4     intermediate variable
C
C
C   COMPLEX FUNCTION CTANH(S)
C       Evaluates the complex hyperbolic tangent
C
C                       Variable  in Argument List
C   S                    COMPLEX*8  complex frequency
C
C
C   SUBROUTINE CURV(A1,A2)
C       Draws circular arc
C
C   Common  ARCCON
C                       Variables in Argument List
C   A1                   REAL*8     starting angle for arc
C   A2                   REAL*8     ending angle for arc
C                       Local Variables
C   ANG1                 REAL*4     starting angle for arc
C   ANG2                 REAL*4     ending angle for arc
C   DA                   REAL*4     incremental angle for plot
C   DTH                  REAL*4     total angle to plot
C   DUMWIL               INTEGER*2  intermediate variable
C   I                    INTEGER*2  do loop index
C   N                    INTEGER*2  number of points to plot
C   T                    REAL*4     current angle
C   XP                   REAL*8     x location of point to plot
C   XY                   CHAR*16    intermediate variable
C   YP                   REAL*8     y location of point to plot
C
C
C   SUBROUTINE ENDPLT
C       Closes plot routines
C
C                       Local Variable
C   DUMMY                INTEGER*2  intermediate variable
C
C
C   LOGICAL FUNCTION fourcolors()
C       Determines type of graphics monitor
C
C   Common  BLANK
C                       Local Variable
C   DUMMY                INTEGER*2  intermediate variable
```

```
C
C
C   SUBROUTINE FUEL(S,GF,PIPEA1,PIPEA2,PIPEA3,PIPEA4,SEGMNA,SECTNA,IGONE)
C       Handles fuel piping logic
C
C   Commons WCAOUT  WORKIT
C                   Variables in Argument List
C   GF               COMPLEX*8  admittance of fuel line looking toward tank
C   IGONE            INTEGER*2  flag for path to be taken
C   SECTNA(75)       INTEGER*2  pipe section types
C   SEGMNA           INTEGER*2  number of pipe sections
C   PIPEA1(75)       REAL*4     first parameter of fuel pipe description
C   PIPEA2(75)       REAL*4     second parameter of fuel pipe description
C   PIPEA3(75)       REAL*4     third parameter of fuel pipe description
C   PIPEA4(75)       REAL*4     fourth parameter of fuel pipe description
C   S                COMPLEX*8  complex frequency
C                   Local Variables
C   A                REAL*4     speed of sound in the fluid (ft/sec)
C   ANS              CHAR*1     response to question
C   AREA(75)         REAL*4     area of pipe section (ft^2)
C   CMAN             REAL*4     manifold capacitance
C   CTANK            REAL*4     tank capacitance
C   DENS             REAL*4     density of fluid (lbm/ft^3)
C   DIA(75)          REAL*4     diameter of pipe section (ft)
C   DPROR            REAL*4     pressure drop across orfices (lbf/ft^2)
C   FUELIN           CHAR*24    name of file containing fuel piping data
C   IMORE            INTEGER*2  internal flag
C   ISTRT            INTEGER*2  internal flag
C   KMAN             REAL*4     bulk modulus of manifold (lbf/ft^2)
C   KTANK            REAL*4     bulk modulus of tank (lbf/ft^2)
C   L(75)            REAL*4     length of pipe section (ft)
C   LFLOW            REAL*4     flow rate through pipe (lbm/sec)
C   LOPEND           INTEGER*2  maximum number of iterations for split pipe
C   LOPOLD           INTEGER*2  previous value of LOPEND
C   PCAP(75)         REAL*4     capacitance of pipe section
C   PCHMB            REAL*4     chamber pressure (lbf/ft^2)
C   PIND(75)         REAL*4     inductance of pipe section
C   PIPEA5(75)       REAL*4     fifth parameter of fuel pipe description
C   PMRAT            REAL*4     chamber pressure/total mass flow
C   SECTA            INTEGER*2  intermediate variable
C   SPLIT            REAL*4     number of lines from pipe split
C   TFLOW            REAL*4     total flow rate of engine (lbm/sec)
C   TITLF            CHAR*20    title from fuel file
C   VOL              REAL*4     volume of tank (ft^3)
C   VOLMF            REAL*4     volume of manifold (ft^3)
C
C
C   SUBROUTINE GINERT(BEND,X,Y)
C       Evaluates curve fit of inertance of bends
C
C                   Variables in Argument List
C   BEND             REAL*4     angle of bend (degrees)
```

```
C   X                REAL*4     ratio of inner to outer radius
C   Y                REAL*4     inertance
C                   Local Variables
C   A                REAL*4     intermediate variable
C   B(3)             REAL*4     coefficient array for inertance fit
C
C
C   SUBROUTINE HHSECT(J,ITYPE,POINT,LEN,DIA,VOL)
C       Computes plot coordinates for Helmholtz resonator
C
C   Common  PIPPXY
C                   Variables in Argument List
C   DIA              REAL*4     diameter of opening (ft)
C   ITYPE(200)       INTEGER*2  type plot element
C   J                INTEGER*2  pointer to element
C   LEN              REAL*4     length of opening (ft)
C   POINT(8,200)     REAL*4     description of plot element
C   VOL              REAL*4     volume of reservoir (ft^3)
C                   Local Variables
C   COSOLD           REAL*4     intermediate variable
C   DIAM             REAL*4     intermediate variable
C   SIDE             REAL*4     intermediate variable
C   SINOLD           REAL*4     intermediate variable
C   XC               REAL*4     intermediate variable
C   XHOLD            REAL*4     intermediate variable
C   XLOLD            REAL*4     intermediate variable
C   XOLD             REAL*4     intermediate variable
C   YC               REAL*4     intermediate variable
C   YHOLD            REAL*4     intermediate variable
C   YLOLD            REAL*4     intermediate variable
C   YOLD             REAL*4     intermediate variable
C
C
C   SUBROUTINE LABANG(XMIN,XMAX,YMIN,YMAX)
C       Labels phase angle plot
C
C   Commons BLANK  FACTOR  NOCOL   WCATIT
C                   Variables in Argument List
C   XMAX             REAL*8     maximum x value for phase angle plot
C   XMIN             REAL*8     minimum x value for phase angle plot
C   YMAX             REAL*8     maximum y value for phase angle plot
C   YMIN             REAL*8     minimum y value for phase angle plot
C                   Local Variables
C   DUMMY            REAL*4     intermediate variable
C   DUMWIL           INTEGER*2  intermediate variable
C   HI               REAL*4     intermediate variable
C   I                INTEGER*2  do loop index
C   IDEL             INTEGER*2  intermediate variable
C   IHI              INTEGER*2  intermediate variable
C   ILO              INTEGER*2  intermediate variable
C   ILOC             INTEGER*2  intermediate variable
C   IMAX             INTEGER*2  intermediate variable
```

```
C   ROW              INTEGER*2  intermediate variable
C   ROWS             INTEGER*2  intermediate variable
C   S                CHAR*4     intermediate variable
C   XHI              CHAR*7     label for x tick marks
C   XP               REAL*8     x point for plot
C   XY               CHAR*16    intermediate variable
C   YHI              CHAR*6     '  180'' upper phase angle label
C   YLO              CHAR*6     ' -180'' lower phase angle label
C   YP               REAL*8     y point for plot
C
C
C   SUBROUTINE LABGAIN(XMIN,XMAX,YMIN,YMAX,ITYPE)
C       Labels gain plot
C
C   Commons BLANK  FACTOR  NOCOL   WCATIT
C                   Variables in Argument List
C   ITYPE            INTEGER*2  which K()
C   XMAX             REAL*8     maximum x value for gain plot
C   XMIN             REAL*8     minimum x value for gain plot
C   YMAX             REAL*8     maximum y value for gain plot
C   YMIN             REAL*8     minimum y value for gain plot
C                   Local Variables
C   DUMMY            REAL*4     intermediate variable
C   DUMWIL           INTEGER*2  intermediate variable
C   HI               REAL*4     intermediate variable
C   I                INTEGER*2  do loop index
C   IDEL             INTEGER*2  intermediate variable
C   IHI              INTEGER*2  intermediate variable
C   ILO              INTEGER*2  intermediate variable
C   ILOC             INTEGER*2  intermediate variable
C   IMAX             INTEGER*2  intermediate variable
C   ROW              INTEGER*2  intermediate variable
C   ROWS             INTEGER*2  intermediate variable
C   S                CHAR*4     intermediate variable
C   XHI              CHAR*7     label for x tick marks
C   XP               REAL*8     x point for plot
C   XY               CHAR*16    intermediate variable
C   YHI              CHAR*6     '  180'' upper phase angle label
C   YLO              CHAR*6     ' -180'' lower phase angle label
C   YP               REAL*8     y point for plot
C
C
C   SUBROUTINE LOWERW(XMIN,XMAX,YMAX,YMIN)
C       Sets up lower plotting window
C
C   Commons BLANK  NOCOL
C                   Variables in Argument List
C   XMAX             REAL*8     maximum x value for Nyquist plot
C   XMIN             REAL*8     minimum x value for Nyquist plot
C   YMAX             REAL*8     maximum y value for Nyquist plot
C   YMIN             REAL*8     minimum y value for Nyquist plot
C                   Local Variables
```

```
C   COLS              INTEGER*2  number of text columns
C   DUMMY             INTEGER*2  intermediate variable
C   ROWS              INTEGER*2  number of text rows
C   XLEN              REAL*8     intermediate variable
C   XWIDTH            INTEGER*2  number of x pixels
C   YHEIGHT           INTEGER*2  number of y pixels
C   YLEN              REAL*8     intermediate variable
C
C
C   SUBROUTINE LOX(S,GOX,PIPEB1,PIPEB2,PIPEB3,PIPEB4,SEGMNB,SECTNB,IGONE)
C       Handles lox piping logic
C
C   Commons WCAOUT   WORKIT
C                    Variables in Argument List
C   GOX               COMPLEX*8  admittance of lox line looking toward tank
C   IGONE             INTEGER*2  flag for path to be taken
C   PIPEB1(75)        REAL*4     first parameter of lox pipe description
C   PIPEB2(75)        REAL*4     second parameter of lox pipe description
C   PIPEB3(75)        REAL*4     third parameter of lox pipe description
C   PIPEB4(75)        REAL*4     fourth parameter of lox pipe description
C   S                 COMPLEX*8  complex frequency
C   SECTNB(75)        INTEGER*2  pipe section types
C   SEGMNB            INTEGER*2  number of pipe sections
C                    Local Variables
C   A                 REAL*4     speed of sound in the fluid (ft/sec)
C   ANS               CHAR*1     response to question
C   AREA(75)          REAL*4     area of pipe section (ft^2)
C   CMAN              REAL*4     manifold capacitance
C   CTANK             REAL*4     tank capacitance
C   DENS              REAL*4     density of fluid (lbm/ft^3)
C   DIA(75)           REAL*4     diameter of pipe section (ft)
C   DPROR             REAL*4     pressure drop across orfices (lbf/ft^2)
C   IMORE             INTEGER*2  internal flag
C   ISTRT             INTEGER*2  internal flag
C   KMAN              REAL*4     bulk modulus of manifold (lbf/ft^2)
C   KTANK             REAL*4     bulk modulus of tank (lbf/ft^2)
C   L(75)             REAL*4     length of pipe section (ft)
C   LFLOW             REAL*4     flow rate through pipe (lbm/sec)
C   LOPEND            INTEGER*2  maximum number of iterations for split pipe
C   LOPOLD            INTEGER*2  previous value of LOPEND
C   LOXIN             CHAR*24    name of file containing lox piping data
C   PCAP(75)          REAL*4     capacitance of pipe section
C   PCHMB             REAL*4     chamber pressure (lbf/ft^2)
C   PIND(75)          REAL*4     inductance of pipe section
C   PIPEB5(75)        REAL*4     fifth parameter of fuel pipe description
C   PMRAT             REAL*4     chamber pressure/total mass flow
C   SECTB             INTEGER*2  intermediate variable
C   SPLIT             REAL*4     number of lines from pipe split
C   TFLOW             REAL*4     total flow rate of engine (lbm/sec)
C   TITLO             CHAR*20    title from lox file
C   VOL               REAL*4     volume of tank (ft^3)
C   VOLMF             REAL*4     volume of manifold (ft^3)
```

```
C
C
C   SUBROUTINE MODIFY(AREA,DIA,L,PIPE1,PIPE2,PIPE3,PIPE4,PIPE5,SECTN,
C                     SEGMN,SECT,PIND,PCAP,LOPEND,LOPOLD,SPLIT,PMRAT,R)
C       Allows modifications to input data
C
C   Commons WCAOUT  WCATIT  WORKIT
C                   Variables in Argument List
C   AREA(75)         REAL*4     area of pipe section (ft^2)
C   DIA(75)          REAL*4     diameter of pipe section (ft)
C   L(75)            REAL*4     length of pipe section (ft)
C   LOPEND           INTEGER*2  maximum number of iterations for split pipe
C   LOPOLD           INTEGER*2  previous value of LOPEND
C   PCAP(75)         REAL*4     capacitance of pipe section
C   PIND(75)         REAL*4     inductance of pipe section
C   PIPE1(75)        REAL*4     first parameter of pipe description
C   PIPE2(75)        REAL*4     second parameter of pipe description
C   PIPE3(75)        REAL*4     third parameter of pipe description
C   PIPE4(75)        REAL*4     fourth parameter of pipe description
C   PIPE5(75)        REAL*4     fifth parameter of pipe description
C   PMRAT            REAL*4     chamber pressure/total mass flow
C   R                CHAR*1     flag for fuel or lox
C   SECT             INTEGER*2  intermediate variable
C   SECTN(75)        INTEGER*2  pipe section types
C   SEGMN            INTEGER*2  number of pipe sections
C   SPLIT            REAL*4     number of lines from pipe split
C                   Local Variables
C   ANS              CHAR*1     response to question
C   AREAB            REAL*4     intermediate variable
C   AVGK             REAL*4     average bulk modulus
C   DIME             REAL*4     intermediate variable
C   GRAV             REAL*4     gravitational constant (lbm-ft/lbf-sec^2)
C   I                INTEGER*2  pointer
C   II               INTEGER*2  do loop index
C   III              INTEGER*2  do loop index
C   ICHG             INTEGER*2  change flag
C   ISEGMN           INTEGER*2  intermediate variable
C   NAME             CHAR*8     name of input variable
C   NAMNAM           INTEGER*2  flag for fuel or lox
C   PI               REAL*4     mathematical constant
C   VALUE            REAL*4     value of input variable
C   VARL(9)          CHAR*8     array of variable names (lower case)
C   VARU(9)          CHAR*8     array of variable names (upper case)
C   VARVAL(9)        CHAR*8     array of variable names for printout
C
C
C   SUBROUTINE NICEGRF(RMIN,RMAX,IMAX,IMMIN,ITYPE)
C       Plots Nyquist curve
C
C   Commons BLANK  FACTOR  NOCOL   WCATIT
C                   Variables in Argument List
C   IMAX             REAL*8     maximum value of complex part
```

```
C   IMMIN             REAL*8     minimum value of complex part
C   ITYPE             INTEGER*2  which K()
C   RMAX              REAL*8     maximum value of real part
C   RMIN              REAL*8     minimum value of real part
C                    Local Variables
C   DUMMY             REAL*4     intermediate variable
C   ROW               INTEGER*2  intermediate variable
C   ROWS              INTEGER*2  intermediate variable
C   S                 CHAR*4     intermediate variable
C   XHI               CHAR*6     label for maximum x value
C   XLO               CHAR*6     label for minimum x value
C   XMAX              REAL*8     maximum x value
C   XMIN              REAL*8     minimum x value
C   YHI               CHAR*6     label for maximum y value
C   YLO               CHAR*6     label for minimum y value
C   YMAX              REAL*8     maximum y value
C   YMIN              REAL*8     minimum y value
C
C
C   SUBROUTINE NYQUIS(GF,GOX,S,TAUT,CSTAR,RBAR,DCDR,THETAC,K,K1R,K2R,
C                     K3R,K4R,K1C,K2C,K3C,K4C,IFUEL,ILOX)
C       Computes the K()'s
C
C                    Variables in Argument List
C   CSTAR             REAL*4     characteristic rocket velocity (ft/sec)
C   DCDR              REAL*4     change in velocity with mixture ratio (ft/sec)
C   GF                COMPLEX*8  admittance of fuel line looking toward tank
C   GOX               COMPLEX*8  admittance of lox line looking toward tank
C   IFUEL             INTEGER*2  flag indicating presence of fuel line
C   ILOX              INTEGER*2  flag indicating presence of lox line
C   K                 INTEGER*2  index of current item
C   K1C(1001)         REAL*4     complex part of K(jw)
C   K1R(1001)         REAL*4     real part of K(jw)
C   K2C(1001)         REAL*4     complex part of K(jw,Gox)
C   K2R(1001)         REAL*4     real part of K(jw,Gox)
C   K3C(1001)         REAL*4     complex part of K(jw,Gf)
C   K3R(1001)         REAL*4     real part of K(jw,Gf)
C   K4C(1001)         REAL*4     complex part of K(jw,Gox,Gf)
C   K4R(1001)         REAL*4     real part of K(jw,Gox,Gf)
C   RBAR              REAL*4     mixture ratio
C   S                 COMPLEX*8  complex frequency
C   TAUT              REAL*4     transport lag (sec)
C   THETAC            REAL*4     characteristic time constant (sec)
C                    Local Variables
C   KG1               COMPLEX*8  K(jw)
C   KG2               COMPLEX*8  K(jw,Gox)
C   KG3               COMPLEX*8  K(jw,Gf)
C   KG4               COMPLEX*8  K(jw,Gox,Gf)
C
C
C   SUBROUTINE PIPPLOT(SEGMN,SECTN,PIPE1,PIPE2,PIPE3,PIPE4,ILOX,R)
C       Supervises plot of piping layout
```

```
C
C   Commons ARCCON  PIPPXY
C                   Variables in Argument List
C   ILOX             INTEGER*2  flag indicating presence of lox line
C   PIPE1(75)        REAL*4     first parameter of pipe description
C   PIPE2(75)        REAL*4     second parameter of pipe description
C   PIPE3(75)        REAL*4     third parameter of pipe description
C   PIPE4(75)        REAL*4     fourth parameter of pipe description
C   R                CHAR*1     flag indicating fuel or lox line
C   SECTN(75)        INTEGER*2  pipe section types
C   SEGMN            INTEGER*2  number of pipe sections
C                   Local Variables
C   DUMWIL           INTEGER*2  intermediate variable
C   I                INTEGER*2  do loop index
C   ITYPE(200)       INTEGER*2  type plot element
C   J                INTEGER*2  pointer to element
C   POINT(8,200)     REAL*4     description of plot element
C   XRANGE           REAL*4     range of x values
C   XY               CHAR*16    intermediate variable
C   X0               REAL*8     intermediate variable
C   X1               REAL*8     intermediate variable
C   X2               REAL*8     intermediate variable
C   X3               REAL*8     intermediate variable
C   YRANGE           REAL*4     range of y values
C   Y0               REAL*8     intermediate variable
C   Y1               REAL*8     intermediate variable
C   Y2               REAL*8     intermediate variable
C   Y3               REAL*8     intermediate variable
C
C
C   SUBROUTINE PLSECT(J,ITYPE,POINT,LEN,DIA,VOL)
C       Computes plot coordinates for parallel resonator
C
C   Commons ARCCON  PIPPXY
C                   Variables in Argument List
C   DIA              REAL*4     diameter of parallel segment (ft)
C   ITYPE(200)       INTEGER*2  type plot element
C   J                INTEGER*2  pointer to element
C   LEN              REAL*4     length of parallel segment (ft)
C   POINT(8,200)     REAL*4     description of plot element
C   VOL              REAL*4     volume of bypassed segment (ft^3)
C                   Local Variables
C   ANGOLD           REAL*4     intermediate variable
C   ANGSAV           REAL*4     intermediate variable
C   COSOLD           REAL*4     intermediate variable
C   DIAM             REAL*4     intermediate variable
C   PDIA             REAL*4     intermediate variable
C   PLEN             REAL*4     intermediate variable
C   RADIUS           REAL*4     intermediate variable
C   SIDE             REAL*4     intermediate variable
C   SINOLD           REAL*4     intermediate variable
C   TURN             REAL*4     intermediate variable
```

```
C   XHC              REAL*4     intermediate variable
C   XHOLD            REAL*4     intermediate variable
C   XHSAV            REAL*4     intermediate variable
C   XLC              REAL*4     intermediate variable
C   XLOLD            REAL*4     intermediate variable
C   XLSAV            REAL*4     intermediate variable
C   XOLD             REAL*4     intermediate variable
C   XSAV             REAL*4     intermediate variable
C   YHC              REAL*4     intermediate variable
C   YHOLD            REAL*4     intermediate variable
C   YHSAV            REAL*4     intermediate variable
C   YLC              REAL*4     intermediate variable
C   YLOLD            REAL*4     intermediate variable
C   YLSAV            REAL*4     intermediate variable
C   YOLD             REAL*4     intermediate variable
C   YSAV             REAL*4     intermediate variable
C
C
C   SUBROUTINE PNYQ(KR,KC,KW,PTS,ITYPE)
C       Plots gain and phase angle
C
C                   Variables in Argument List
C   ITYPE            INTEGER*2  which K()
C   KC(PTS)          REAL*4     complex part of K()
C   KR(PTS)          REAL*4     real part of ()
C   KW(PTS)          REAL*4     frequency
C   PTS              INTEGER*2  number of points
C                   Local Variables
C   DUMWIL           INTEGER*2  intermediate variable
C   I                INTEGER*2  do loop index
C   X(1001)          REAL*4     log of frequency (base 10)
C   XHI              REAL*8     intermediate variable
C   XLO              REAL*8     intermediate variable
C   XMAX             REAL*8     maximum x value
C   XMIN             REAL*8     minimum x value
C   XP               REAL*8     x point to plot
C   XY               CHAR*16    intermediate variable
C   YC(1001)         REAL*4     phase angle
C   YMAXC            REAL*8     maximum phase angle
C   YMAXR            REAL*8     maximum amplitude
C   YMINC            REAL*8     minimum phase angle
C   YMINR            REAL*8     minimum amplitude
C   YP               REAL*8     y point to plot
C   YR(1001)         REAL*4     amplitude
C
C
C   SUBROUTINE RLINE(TITL,PMRAT,SEGMN,SECTN,PIPE1,PIPE2,PIPE3,
C     PIPE4,PIPE5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,IUNIT)
C       Reads fuel or lox file
C
C   Common  WORKIT
C                   Variables in Argument List
```

```
C   AREA(75)           REAL*4      area of pipe section (ft^2)
C   DIA(75)            REAL*4      diameter of pipe section (ft)
C   IUNIT              INTEGER*2   unit number of current file (fuel or lox)
C   L(75)              REAL*4      length of pipe section (ft)
C   LOPEND             INTEGER*2   maximum number of iterations for split pipe
C   LOPOLD             INTEGER*2   previous value of LOPEND
C   PCAP(75)           REAL*4      capacitance of pipe section
C   PIND(75)           REAL*4      inductance of pipe section
C   PIPE1(75)          REAL*4      first parameter of pipe description
C   PIPE2(75)          REAL*4      second parameter of pipe description
C   PIPE3(75)          REAL*4      third parameter of pipe description
C   PIPE4(75)          REAL*4      fourth parameter of pipe description
C   PIPE5(75)          REAL*4      fifth parameter of pipe description
C   PMRAT              REAL*4      chamber pressure/total mass flow
C   SECTN(75)          INTEGER*2   pipe section types
C   SEGMN              INTEGER*2   number of pipe sections
C   SPLIT              REAL*4      number of lines from pipe split
C   TITL               CHAR*20     title from fuel or lox file
C                     Local Variables
C   ANS                REAL*4      response to question
C   AREAB              REAL*4      intermediate variable
C   AVGK               REAL*4      average bulk modulus
C   DIME               REAL*4      intermediate variable
C   GRAV               REAL*4      gravitational constant (lbm-ft/lbf-sec^2)
C   I                  INTEGER*2   do loop index
C   PI                 REAL*4      mathematical constant
C   VALUE              REAL*4      intermediate variable
C
C
C   SUBROUTINE SETPLT
C       Sets up the plot environment
C
C   Commons BLANK  NOCOL   WCAPAS
C
C
C   SUBROUTINE STSECT(J,ITYPE,POINT,LEN,DIA)
C       Computes plot coordinates for a straight section
C
C   Common  PIPPXY
C                     Variables in Argument List
C   DIA                REAL*4      diameter of segment (ft)
C   ITYPE(200)         INTEGER*2   type plot element
C   J                  INTEGER*2   pointer to element
C   LEN                REAL*4      length of segment (ft)
C   POINT(8,200)       REAL*4      description of plot element
C
C
C   SUBROUTINE TSSECT(J,ITYPE,POINT,LEN,DIA)
C       Computes plot coordinates for a tuned stub
C
C   Common  PIPPXY
C                     Variables in Argument List
```

```
C   DIA              REAL*4     diameter of tuned stub (ft)
C   ITYPE(200)       INTEGER*2  type plot element
C   J                INTEGER*2  pointer to element
C   LEN              REAL*4     length of tuned stub
C   POINT(8,200)     REAL*4     description of plot element
C                   Local Variables
C   DIAM             REAL*4     intermediate variable
C
C
C   SUBROUTINE UPPERW(X00,Y00,X11,Y11,ILOX,R)
C       Sets up upper plotting window
C
C   Commons BLANK  NOCOL   WCATIT
C                   Variables in Argument List
C   ILOX             INTEGER*2  flag indicating presence of lox line
C   R                CHAR*1     flag indicating fuel or lox
C   X00              REAL*4     minimum x value
C   X11              REAL*4     maximum x value
C   Y00              REAL*4     minimum y value
C   Y11              REAL*4     maximum y value
C                   Local Variables
C   ADDX             REAL*4     intermediate variable
C   ADDY             REAL*4     intermediate variable
C   COLS             INTEGER*2  number of text columns
C   DUMMY            INTEGER*2  intermediate variable
C   HALFY            REAL*4     intermediate variable
C   PICX             REAL*4     intermediate variable
C   PICY             REAL*4     intermediate variable
C   ROWS             INTEGER*2  number of text rows
C   S                CHAR*4     intermediate variable
C   XRANG            REAL*4     intermediate variable
C   XRAT             REAL*4     intermediate variable
C   XWIDTH           INTEGER*2  number ox x pixels
C   X0               REAL*8     minimum x value
C   X1               REAL*8     maximum x value
C   YHEIGHT          INTEGER*2  number of y pixels
C   YRANG            REAL*4     intermediate variable
C   YRAT             REAL*4     intermediate variable
C   Y0               REAL*8     minimum y value
C   Y1               REAL*8     maximum y value
C
C
C   SUBROUTINE WINDLO(XMIN,XMAX,YMIN,YMAX)
C       Sets up gain window
C
C   Commons BLANK  NOCOL
C                   Variables in Argument List
C   XMAX             REAL*8     maximum x value
C   XMIN             REAL*8     minimum x value
C   YMAX             REAL*8     maximum y value
C   YMIN             REAL*8     minimum y value
C                   Local Variables
```

```
C    COLS               INTEGER*2  number of text columns
C    DUMMY              INTEGER*2  intermediate variable
C    HALFY              INTEGER*2  intermediate variable
C    ROWS               INTEGER*2  number of text rows
C    XLEN               REAL*8     intermediate variable
C    XMAXP              REAL*8     maximum x value
C    XMINP              REAL*8     minimum x value
C    XWIDTH             INTEGER*2  number of x pixels
C    YHEIGHT            INTEGER*2  number of y pixels
C    YLEN               REAL*8     intermediate variable
C    YMAXP              REAL*8     maximum y value
C    YMINP              REAL*8     minimum y value
C
C
C    SUBROUTINE WINDUP(XMIN,XMAX,YMIN,YMAX)
C        Sets up phase angle window
C
C    Commons BLANK  NOCOL
C                      Variables in Argument List
C    XMAX               REAL*8     maximum x value
C    XMIN               REAL*8     minimum x value
C    YMAX               REAL*8     maximum y value
C    YMIN               REAL*8     minimum y value
C                      Local Variables
C    COLS               INTEGER*2  number of text columns
C    DUMMY              INTEGER*2  intermediate variable
C    HALFY              INTEGER*2  intermediate variable
C    ROWS               INTEGER*2  number of text rows
C    XLEN               REAL*8     intermediate variable
C    XMAXP              REAL*8     maximum x value
C    XMINP              REAL*8     minimum x value
C    XWIDTH             INTEGER*2  number of x pixels
C    YHEIGHT            INTEGER*2  number of y pixels
C    YLEN               REAL*8     intermediate variable
C    YMAXP              REAL*8     maximum y value
C    YMINP              REAL*8     minimum y value
C
C
C    SUBROUTINE WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,
C                   VOLMF,PCHMB,DPROR)
C        Moves arguments from common /WORKIT/
C
C    Common  WORKIT
C                      Variables in Argument List
C    A                  REAL*4     speed of sound in the fluid (ft/sec)
C    CMAN               REAL*4     manifold capacitance
C    CTANK              REAL*4     tank capacitance
C    DENS               REAL*4     density of fluid (lbm/ft^3)
C    DPROR              REAL*4     pressure drop across orfices (lbf/ft^2)
C    KMAN               REAL*4     bulk modulus of manifold (lbf/ft^2)
C    KTANK              REAL*4     bulk modulus of tank (lbf/ft^2)
C    LFLOW              REAL*4     flow rate through pipe (lbm/sec)
```

```
C   PCHMB            REAL*4     chamber pressure (lbf/ft^2)
C   TFLOW            REAL*4     total flow rate of engine (lbm/sec)
C   VOL              REAL*4     volume of tank (ft^3)
C   VOLMF            REAL*4     volume of manifold (ft^3)
C
C
C   SUBROUTINE WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,
C                     VOLMF,PCHMB,DPROR)
C       Moves arguments to common /WORKIT/
C
C   Common  WORKIT
C                   Variables in Argument List
C   A                REAL*4     speed of sound in the fluid (ft/sec)
C   CMAN             REAL*4     manifold capacitance
C   CTANK            REAL*4     tank capacitance
C   DENS             REAL*4     density of fluid (lbm/ft^3)
C   DPROR            REAL*4     pressure drop across orfices (lbf/ft^2)
C   KMAN             REAL*4     bulk modulus of manifold (lbf/ft^2)
C   KTANK            REAL*4     bulk modulus of tank (lbf/ft^2)
C   LFLOW            REAL*4     flow rate through pipe (lbm/sec)
C   PCHMB            REAL*4     chamber pressure (lbf/ft^2)
C   TFLOW            REAL*4     total flow rate of engine (lbm/sec)
C   VOL              REAL*4     volume of tank (ft^3)
C   VOLMF            REAL*4     volume of manifold (ft^3)
C
C
C   SUBROUTINE ZREAD(NAME,VALUE)
C       Reads input for input modification
C
C                   Variables in Argument List
C   NAME(8)          CHAR*1     name of input variable
C   VALUE            REAL*4     value of input variable
C                   Local Variables
C   BLK              CHAR*1     ' '
C   CARD(80)         CHAR*1     card image
C   CEND(3)          CHAR*1     'E','N','D'
C   COMMA            CHAR*1     ','
C   CTIT(5)          CHAR*1     'T','I','T','L','E'
C   DCARD            CHAR*80    card image
C   E                CHAR*1     'E'
C   FRACT            REAL*4     fractional part of number
C   I                INTEGER*2  do loop index
C   ICOUNT           INTEGER*2  position counter
C   ID               INTEGER*2  position counter
C   II               INTEGER*2  position counter
C   J                INTEGER*2  do loop index
C   JJ               INTEGER*2  position counter
C   LE               CHAR*1     'e'
C   LEND(3)          CHAR*1     'e','n','d'
C   LTIT(5)          CHAR*1     't','i','t','l','e'
C   MINUS            CHAR*1     '-'
C   NUMBER(10)       CHAR*1     '0','1','2','3','4','5','6','7','8','9'
```

```
C   PERIOD            CHAR*1     '.'
C   PLUS              CHAR*1     '+'
C   POUND             CHAR*1     '#'
C   QUEST             CHAR*1     '?'
C   SIGN              REAL*4     sign of number or exponent
C   WHOLE             REAL*4     WHOLE PART OF NUMBER
C
$LARGE
      INCLUDE 'FGRAPH.FI'
      INCLUDE 'FGRAPH.FD'
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 NCOLS,NMODE
      INTEGER*2 IHR,IMIN,ISEC,I100,IYR,IMON,IDAY
      CHARACTER*2 AM,PM,AP
      COMPLEX GF,GOX,S
      REAL K1R(1001),K2R(1001),K3R(1001),K1C(1001),K2C(1001),K3C(1001)
      REAL K4R(1001),K4C(1001),KW(1001)
      REAL PIPEA1(75),PIPEA2(75),PIPEA3(75),PIPEA4(75)
      REAL PIPEB1(75),PIPEB2(75),PIPEB3(75),PIPEB4(75)
      REAL LFREQ,TAUT,CSTAR,RBAR,THETAC,DCDR
      INTEGER SECTNA(75),SECTNB(75),SEGMNA,SEGMNB,PTS,CHOICE
      CHARACTER ANS*1
      CHARACTER*24 NAMLIN(2)
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      CHARACTER*24 VARI
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /WCAOUT/NAMLIN,IUNIT
      COMMON /FACTOR/SFAC
      DATA AM/'AM'/,PM/'PM'/
      DATA IFUEL/0/,ILOX/0/
    1 FORMAT(E15.6)
    2 FORMAT(I5,4E15.6)
    3 FORMAT(1P4E15.6)
    4 FORMAT(1PE13.5,E12.5,E12.5)
    5 FORMAT(/'       FREQ',8X,'FREQ-NORM',9X,'REALS',11X,'IMAGINARY'/)
    8 FORMAT(I5,1P3E15.6)
    9 FORMAT(E11.4,E11.4)
   10 FORMAT(A20,1X,I2.2,':',I2.2,A2,4X,I2.2,'-',I2.2,'-',I2.2)
      CALL GETTIM(IHR,IMIN,ISEC,I100)
      CALL GETDAT(IYR,IMON,IDAY)
      IYR=IYR-1900
      CALL CLEARSCREEN(0)
      WRITE(*,'(10X,A)')
     *'┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┓'
      WRITE(*,'(10X,A)')
     *'┃                                                              ┃'
      IF(IHR.LT.12)  THEN
       WRITE(*,'(10X,A)')
     *'┃               Good Morning and Welcome to NYQ!!              ┃'
       AP=AM
      ELSE
    WRITE(*,'(10X,A)')
  *'║              Good Afternoon and Welcome to NYQ!!             ║'
    AP=PM
    IF(IHR.GT.12)  IHR=IHR-12
   ENDIF
   WRITE(*,'(10X,A)')
  *'║                                                              ║'
   WRITE(*,'(10X,A)')
  *'║          Program NYQ provides stability predictions          ║'
   WRITE(*,'(10X,A)')
  *'║                    of feedline systems                       ║'
   WRITE(*,'(10X,A)')
  *'║                                                              ║'
   WRITE(*,'(10X,A)')
  *'║                 To send a plot to the printer                ║'
   WRITE(*,'(10X,A)')
  *'║                                                              ║'
   WRITE(*,'(10X,A)')
  *'║             The computer MUST be in GRAPHICS mode            ║'
   WRITE(*,'(10X,A)')
  *'║                                                              ║'
   WRITE(*,'(10X,A)')
  *'║      Hit PrScn to send the current plot to the printer       ║'
   WRITE(*,'(10X,A)')
  *'║                                                              ║'
   WRITE(*,'(10X,A)')
  *'╚══════════════════════════════════════════════════════════════╝'
   WRITE(*,*)' '
   SFAC=1.0
   WRITE(*,*)' If you want frequency in rad/sec, hit enter.'
   WRITE(*,'(A\)')' If you want it in Hertz, enter "H". '
   READ(*,'(A)')ANS
   IF(ANS.EQ.'H'.OR.ANS.EQ.'h')  SFAC=6.283185
20 CONTINUE
   OPEN(UNIT=13,FILE='CONST.DAT')
   WRITE(*,'(A\)')' Do you have FUEL data? '
   READ(*,'(A)')ANS
   IF(ANS .EQ.'N' .OR. ANS .EQ. 'n') THEN
    IFUEL=1
   ELSE
    IGONE=2
    CALL FUEL(S,GF,PIPEA1,PIPEA2,PIPEA3,PIPEA4,SEGMNA,SECTNA,IGONE)
   ENDIF
   WRITE(*,'(A\)')' Do you have LOX data? '
   READ(*,'(A)')ANS
   IF(ANS .EQ.'N' .OR. ANS .EQ. 'n') THEN
    ILOX=1
   ELSE
    IGONE=2
    CALL LOX(S,GOX,PIPEB1,PIPEB2,PIPEB3,PIPEB4,SEGMNB,SECTNB,IGONE)
   ENDIF
   IGONE=0
```

```
C       THIS SECTION COMPUTES THE NEW ADMITTANCE OVER VARYING FREQUENCIES.
   95 CONTINUE
      WRITE(*,*)' Enter 20 character title'
      READ(*,'(A)')TITLF
      WRITE(TITLE,10)TITLF,IHR,IMIN,AP,IMON,IDAY,IYR
      WRITE(*,*)'  Are the following variables in a file? (Y/N) '
      WRITE(*,*)' '
      WRITE(*,*)'      VARIABLES   '
      WRITE(*,*)' TRANSPORT LAG'
      WRITE(*,*)' CHARACTERISTIC ROCKET VELOCITY'
      WRITE(*,*)' MIXTURE RATIO '
      WRITE(*,*)' CHARACTERISTIC TIME CONSTANT '
      WRITE(*,*)' CHANGE IN VELOCITY WITH MIXTURE RATIO '
      WRITE(*,*)' '
      READ(*,'(A)')ANS
      IF(ANS .EQ. 'N' .OR. ANS .EQ. 'n') THEN
  101 CONTINUE
       WRITE(*,*)'Enter values for VARIABLES as listed above.'
       READ(*,*,ERR=100)TAUT,CSTAR,RBAR,THETAC,DCDR
       GOTO 102
  100 CONTINUE
       WRITE(*,*)'  Enter numeric values only.  Please try again !!'
       GOTO 101
  102 CONTINUE
       WRITE(13,*)TAUT
       WRITE(13,*)CSTAR
       WRITE(13,*)RBAR
       WRITE(13,*)THETAC
       WRITE(13,*)DCDR
       WRITE(13,*)'        VARIABLES  '
       WRITE(13,*)'  TAUT     = ',TAUT
       WRITE(13,*)' CSTAR     = ',CSTAR
       WRITE(13,*)'  RBAR     = ',RBAR
       WRITE(13,*)'THETAC     = ',THETAC
       WRITE(13,*)'  DCDR     = ',DCDR
       ELSE
       WRITE(*,*)'Is the name of the file CONST.DAT? (Y/N) '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'N'.OR.ANS.EQ.'n')  THEN
        WRITE(*,'(A\)')'  Enter name of file with VARIABLES data '
        READ(*,'(A)')VARI
        OPEN(UNIT=13,FILE=VARI)
       ENDIF
       REWIND 13
       READ(13,*)TAUT
       READ(13,*)CSTAR
       READ(13,*)RBAR
       READ(13,*)THETAC
       READ(13,*)DCDR
      ENDIF
   27 CONTINUE
  201 CONTINUE
```

```
      IF(SFAC.EQ.1.0)  THEN
       WRITE(*,*)' Enter range of frequencies in rad/sec '
      ELSE
       WRITE(*,*)' Enter range of frequencies in Hertz '
      ENDIF
      WRITE(*,*)' Low freq=1 high freq=2 #pts=10'
      WRITE(*,*)'   1001 = Maximum number of points'
      READ(*,*,ERR=200)LFREQ,HFREQ,PTS
      IF(LFREQ.LE.0.0)  LFREQ=1.0E-5
      IF(PTS.LE.1)  GO TO 30
      GO TO 202
  200 CONTINUE
      WRITE(*,*)'  Enter numeric values only. Please try again !!'
      GO TO 201
  202 CONTINUE
C   THIS SECTION CALCULATES THE ADMITTANCES FOR FUEL AND LOX, THEN
C   CALCULATES THE COMPLEX K(JW) IN THE "PREDICTION OF THE LINEAR
C   STABILITY BEHAVIOR OF LIQUID PROPELLANT PROPULSION SYSTEMS",
C   VOLUME 1, PAGE 47.
C
      NPTS=PTS/3
      IF(NPTS.GT.1)  THEN
       SSIZE1=0.1*(HFREQ-LFREQ)/(NPTS-1)
       SSIZE2=0.3*(HFREQ-LFREQ)/NPTS
       IF(3*NPTS.EQ.PTS)  THEN
        SSIZE3=0.6*(HFREQ-LFREQ)/NPTS
       ELSEIF(3*NPTS.EQ.PTS-1)  THEN
        SSIZE3=0.6*(HFREQ-LFREQ)/(NPTS+1)
       ELSEIF(3*NPTS.EQ.PTS-2)  THEN
        SSIZE3=0.6*(HFREQ-LFREQ)/(NPTS+2)
       ENDIF
      ELSE
       SSIZE1=(HFREQ-LFREQ)/(PTS-1)
       NPTS=PTS
      ENDIF
C       PLOT FUEL PIPE LAYOUT ON SCREEN 1
       CALL SETPLT
       IF(IFUEL.EQ.0)  CALL PIPPLOT(SEGMNA,SECTNA,PIPEA1,PIPEA2,
     *                              PIPEA3,PIPEA4,ILOX,'A')
       IF(ILOX.EQ.0)  CALL PIPPLOT(SEGMNB,SECTNB,PIPEB1,PIPEB2,
     *                             PIPEB3,PIPEB4,ILOX,'B')
       CALL clearscreen(0)
       WRITE(*,*)'  Please wait while computations proceed. '
      W=LFREQ
      DO 29 K=1,PTS
       IF(K.LE.NPTS)  THEN
        IF(K.GT.1)  W=W+SSIZE1
       ELSEIF(K.GT.2*NPTS)  THEN
        W=W+SSIZE3
       ELSE
        W=W+SSIZE2
       ENDIF
```

```
    IF(K.EQ.PTS)  THEN
     W=HFREQ
    ENDIF
    KW(K)=W
    S=CMPLX(0.0,SFAC*W)
    IF(IFUEL.EQ.0)  CALL FUEL(S,GF,PIPEA1,PIPEA2,PIPEA3,PIPEA4,
  *                            SEGMNA,SECTNA,IGONE)
    IF(ILOX.EQ.0)  CALL LOX(S,GOX,PIPEB1,PIPEB2,PIPEB3,PIPEB4,
  *                            SEGMNB,SECTNB,IGONE)
   CALL NYQUIS(GF,GOX,S,TAUT,CSTAR,RBAR,DCDR,THETAC,K,K1R,K2R,K3R,
  * K4R,K1C,K2C,K3C,K4C,IFUEL,ILOX)
29 CONTINUE
81 CONTINUE
   WRITE(*,*)'   Enter graph selection '
   WRITE(*,*)' '
   WRITE(*,*)'   1   Nyquist plot independent of fuel or lox. '
   IF(ILOX.EQ.0)
  * WRITE(*,*)'   2   Nyquist plot independent of fuel.'
   IF(IFUEL.EQ.0)
  * WRITE(*,*)'   3   Nyquist plot independent of lox.'
   IF(ILOX.EQ.0.AND.IFUEL.EQ.0)
  * WRITE(*,*)'   4   Nyquist plot with fuel and lox.'
   WRITE(*,*)'   5   Phase-Gain plot independent of fuel or lox. '
   IF(ILOX.EQ.0)
  * WRITE(*,*)'   6   Phase-Gain plot independent of fuel.'
   IF(IFUEL.EQ.0)
  * WRITE(*,*)'   7   Phase-Gain plot independent of lox.'
   IF(ILOX.EQ.0.AND.IFUEL.EQ.0)
  * WRITE(*,*)'   8   Phase-Gain plot with fuel and lox.'
   WRITE(*,*)'   9   End plots.'
   WRITE(*,*)' '
   READ(*,*)CHOICE
   IF(CHOICE.EQ.9)  GO TO 30
   IF(CHOICE.LT.1.OR.CHOICE.GT.8)  THEN
    WRITE(*,*)' Number must be between 1 and 9, TRY AGAIN'
    GO TO 81
   ENDIF
   IF(ILOX.EQ.1)  THEN
    IF(MOD(CHOICE,2).EQ.0)  THEN
     WRITE(*,*)' No LOX file, do not use 2,4,6,8'
     GO TO 81
    ENDIF
   ENDIF
   IF(IFUEL.EQ.1)  THEN
    IF(CHOICE.EQ.3.OR.CHOICE.EQ.4.OR.CHOICE.GE.7)  THEN
     WRITE(*,*)' No FUEL file, do not use 3,4,7,8'
     GO TO 81
    ENDIF
   ENDIF
   CALL SETPLT
   CALL GETTIM(IHR,IMIN,ISEC,I100)
   CALL GETDAT(IYR,IMON,IDAY)
```

```
      IYR=IYR-1900
      IF(IHR.LT.12)  THEN
       AP=AM
      ELSE
       AP=PM
       IF(IHR.GT.12)  IHR=IHR-12
      ENDIF
      IF(CHOICE.EQ.1) CALL ALLPT(K1R,K1C,PTS,1)
      IF(CHOICE.EQ.2) CALL ALLPT(K2R,K2C,PTS,2)
      IF(CHOICE.EQ.3) CALL ALLPT(K3R,K3C,PTS,3)
      IF(CHOICE.EQ.4) CALL ALLPT(K4R,K4C,PTS,4)
      IF(CHOICE.EQ.5) CALL PNYQ(K1R,K1C,KW,PTS,1)
      IF(CHOICE.EQ.6) CALL PNYQ(K2R,K2C,KW,PTS,2)
      IF(CHOICE.EQ.7) CALL PNYQ(K3R,K3C,KW,PTS,3)
      IF(CHOICE.EQ.8) CALL PNYQ(K4R,K4C,KW,PTS,4)
      CALL ENDPLT
      GO TO 81
   30 CONTINUE
      WRITE(*,*)' Enter E to exit,'
      WRITE(*,*)'       F to run new frequency range,'
      WRITE(*,*)'       C to run a new case, '
      WRITE(*,'(A\)')'        N to read new files. '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'F'.OR.ANS.EQ.'f')  GO TO 27
      IF(ANS.EQ.'E'.OR.ANS.EQ.'e')  STOP
      IF(ANS.EQ.'C'.OR.ANS.EQ.'c') THEN
       IF(IFUEL.EQ.0)  THEN
        IGONE=1
        CALL FUEL(S,GF,PIPEA1,PIPEA2,PIPEA3,PIPEA4,SEGMNA,SECTNA,IGONE)
       ENDIF
       IF(ILOX.EQ.0)  THEN
        IGONE=1
        CALL LOX(S,GOX,PIPEB1,PIPEB2,PIPEB3,PIPEB4,SEGMNB,SECTNB,IGONE)
       ENDIF
       IGONE=0
       GO TO 95
      ENDIF
      IF(ANS.EQ.'N'.OR.ANS.EQ.'n') THEN
       IFUEL=0
       ILOX=0
       GO TO 20
      ENDIF
      WRITE(*,*)' You did not enter E, F, C, or N. Try again.'
      GO TO 30
      END
      SUBROUTINE ADMIT(S,GADM,A,AREA,CMAN,CTANK,DPROR,L,LFLOW,PMRAT,
     *                 SEGMN,SECTN,SPLIT,LOPEND,PCAP,PIND)
C       determines admittance looking toward tank
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      INTEGER SEGMN,SECTN(75)
      REAL AREA(75),PCAP(75),PIND(75),L(75),LFLOW,ZO(75)
      COMPLEX G(0:75),ZT(0:75),ZG(75),GOLD(0:75),GADM,S,G1,ZGEFF,ZOEFF
      COMPLEX CTANH,RHS,CFAC,CAPN,CAPM
      DATA GRAV/32.2/
      ZTOP=A/(GRAV*PMRAT)
      ZOR=2.0*DPROR/(LFLOW*PMRAT)
      GOLD(0)=0.0
      DO 26 I=1,SEGMN
       GOLD(I)=0.0
       IF(SECTN(I).LE.1.OR.SECTN(I).EQ.9)  THEN
        ZO(I)=ZTOP/AREA(I)
       ELSEIF(SECTN(I).EQ.2)  THEN
        ZO(I)=ZTOP/AREA(I)
       ELSE
        ZO(I)=SQRT(PIND(I)/PCAP(I))
       ENDIF
   26 CONTINUE
      G(0)=CTANK*PMRAT*S
      G(0)=G(0)/SPLIT
      ZT(0)=1.0/G(0)
      DO 281 KLOOP=1,LOPEND
       G1=G(0)+1.0
       DO 27 I=1,SEGMN
        ZGEFF=G(I-1)
        IF(SECTN(I).LE.1.OR.SECTN(I).EQ.9)  THEN
C             BEND IN PIPE OR STRAIGHT SECTION
         TL=L(I)/A
         IF(KLOOP.NE.1.AND.SECTN(I).EQ.9)  THEN
          ZGEFF=G(I-1)+(SPLIT-1.0)/ZG(I-1)
         ENDIF
         G(I)=(1.0+CTANH(S*TL)/(ZGEFF*ZO(I)))/(1.0+ZGEFF*ZO(I)*
     *         CTANH(S*TL))
        ELSEIF(SECTN(I).EQ.2)  THEN
C             INLINE RESONATOR ACCUMULATOR
         G(I)=1.0+PCAP(I)*S/ZGEFF
        ELSEIF(SECTN(I).EQ.3)  THEN
C             TUNED STUB ACCUMULATOR
         G(I)=1.0+CTANH(S*SQRT(PIND(I)*PCAP(I)))/(ZO(I)*ZGEFF)
        ELSEIF(SECTN(I).EQ.4)  THEN
C             HELMHOLTZ RESONATOR ACCUMULATOR
         G(I)=1.0+S*PCAP(I)/(1.0+PIND(I)*PCAP(I)*S**2)/ZGEFF
        ELSEIF(SECTN(I).EQ.5)  THEN
C             PARALLEL RESONATOR ACCUMULATOR
         G(I)=PIND(I)*PCAP(I)*S**2+1.0
         G(I)=G(I)/(G(I)+PIND(I)*S*ZGEFF)
        ELSEIF(SECTN(I).EQ.6)  THEN
C             PUMP
         G(I)=(1.0+PCAP(I)*S/ZGEFF)/(1.0+(PIND(I)*S+AREA(I))*
     *        (PCAP(I)*S+ZGEFF))
        ENDIF
```

```
          G(I)=G(I)*ZGEFF
          G1=G1*G(I)
          ZT(I)=1.0/G(I)
   27 CONTINUE
        G(SEGMN+1)=1.0+CMAN*PMRAT*S/G(SEGMN)
        G1=G1*G(SEGMN+1)
        G(SEGMN+1)=G(SEGMN+1)*G(SEGMN)
        G(SEGMN+2)=1.0/(1.0+ZOR*G(SEGMN+1))
        G1=G1*G(SEGMN+2)
        G(SEGMN+2)=G(SEGMN+2)*G(SEGMN+1)
        IF(LOPEND.EQ.1)  GO TO 281
        ZG(SEGMN)=ZOR/(ZOR*CMAN*PMRAT*S+1.0)
        IF(SEGMN.NE.1)  THEN
         DO 271 I=SEGMN-1,1,-1
          ZGEFF=ZG(I+1)
          ZOEFF=ZO(I+1)
          IF(SECTN(I+1).LE.1.OR.SECTN(I+1).EQ.9)  THEN
C              BEND IN PIPE OR STRAIGHT SECTION
           TL=(L(I)+L(I+1))/A
           CAPN=(ZOEFF-ZT(I-1))/(ZOEFF+ZT(I-1))
           CAPM=(ZOEFF-ZGEFF)/(ZOEFF+ZGEFF)
           CFAC=CEXP(-2.0*S*TL)
           RHS=(ZOEFF+ZGEFF)*(1.0-CAPN*CAPM*CFAC)*CEXP(S*L(I+1)/A)
           CFAC=CAPN*CFAC*CEXP(2.0*S*L(I+1)/A)
           ZG(I)=(RHS-ZOEFF*(1.0-CFAC))/(1.0+CFAC)
           IF(SECTN(I+1).EQ.9)  THEN
            ZG(I)=ZG(I)/SPLIT
           ENDIF
          ELSEIF(SECTN(I+1).EQ.2)  THEN
C              INLINE RESONATOR ACCUMULATOR
           ZG(I)=ZGEFF/(ZGEFF*PCAP(I+1)*S+1.0)
          ELSEIF(SECTN(I+1).EQ.3)  THEN
C              TUNED STUB ACCUMULATOR
           ZG(I)=ZOEFF/CTANH(S*SQRT(PIND(I+1)*PCAP(I+1)))
           ZG(I)=(ZG(I)*ZGEFF)/(ZG(I)+ZGEFF)
          ELSEIF(SECTN(I+1).EQ.4)  THEN
C              HELMHOLTZ RESONATOR ACCUMULATOR
           ZG(I)=(1.0+PIND(I+1)*PCAP(I+1)*S**2)/(PCAP(I+1)*S)
           ZG(I)=(ZG(I)*ZGEFF)/(ZG(I)+ZGEFF)
          ELSEIF(SECTN(I+1).EQ.5)  THEN
C              PARALLEL RESONATOR ACCUMULATOR
           ZG(I)=ZGEFF+PIND(I+1)*S/(PIND(I+1)*PCAP(I+1)*S**2+1.0)
          ELSEIF(SECTN(I+1).EQ.6)  THEN
C              PUMP
           ZG(I)=ZGEFF+PIND(I+1)*S-AREA(I+1)
           ZG(I)=ZG(I)/(1.0+ZG(I)*PCAP(I+1)*S)
          ENDIF
  271 CONTINUE
        ENDIF
        IF(KLOOP.EQ.1)  GO TO 281
        ERRP=0.0
        DO 272 I=1,SEGMN
        GDIF=SQRT((REAL(G(I))-REAL(GOLD(I)))**2+(AIMAG(G(I))-
     *       AIMAG(GOLD(I)))**2)
        IF(GDIF.GT.ERRP)  ERRP=GDIF
  272 CONTINUE
       IF(ERRP.LT.0.001)  GO TO 282
  281 CONTINUE
       IF(LOPEND.EQ.1)  GO TO 282
       IF(IOPEN.EQ.0)  THEN
        OPEN(UNIT=14,FILE='SURF.ERR')
        WRITE(14,*)' '
        WRITE(14,*)' '
        WRITE(14,*)TITLE
        WRITE(14,*)' '
        IOPEN=1
       ENDIF
       WRITE(14,'('' jw ='',F8.1,''  after'',I3,'' iterations'',
     *           '' has error of'',F8.3,''% '')')
     *           AIMAG(S),LOPEND,100.0*ERRP
  282 CONTINUE
       GADM=G(SEGMN+2)
      RETURN
      END
      SUBROUTINE ALLPT(WHOLD,GHOLD,PTS,ITYPE)
C       Supervises Nyquist plot
      INCLUDE 'FGRAPH.FD'
      RECORD/WXYCOORD/XY
      INTEGER*2 DUMWIL
      REAL WHOLD(1001),GHOLD(1001)
      REAL*8 RMIN,RMAX,IMMIN,IMAX
      REAL*8 X,Y
      INTEGER PTS
      RMAX=WHOLD(1)
      RMIN=WHOLD(1)
      IMAX=GHOLD(1)
      IMMIN=GHOLD(1)
      DO 21 I=2,PTS
       IF(WHOLD(I).GT.RMAX)  RMAX=WHOLD(I)
       IF(WHOLD(I).LT.RMIN)  RMIN=WHOLD(I)
       IF(GHOLD(I).GT.IMAX)  IMAX=GHOLD(I)
       IF(GHOLD(I).LT.IMMIN)  IMMIN=GHOLD(I)
   21 CONTINUE
      CALL LOWERW(RMIN,RMAX,IMAX,IMMIN)
      CALL NICEGRF(RMIN,RMAX,IMAX,IMMIN,ITYPE)
      CALL SETLINESTYLE(62268)
      X=0.0
      Y=IMMIN
      CALL MOVETO_W(X,Y,XY)
      Y=IMAX
      DUMWIL=LINETO_W(X,Y)
      Y=0.0
      X=RMIN
      CALL MOVETO_W(X,Y,XY)
      X=RMAX
      DUMWIL=LINETO_W(X,Y)
      CALL SETLINESTYLE(65535)
      X=WHOLD(1)
      Y=GHOLD(1)
      CALL MOVETO_W(X,Y,XY)
      DO 25 I=2,PTS
       X=WHOLD(I)
       Y=GHOLD(I)
       DUMWIL=LINETO_W(X,Y)
   25 CONTINUE
      RETURN
      END
      SUBROUTINE BENDS(PIPE1,PIPE2,PIPE3,PIPE4,VALUE,DIME)
C       Computes effective straight pipe for bend
      REAL LBEND,INRAD,INERT,LPRME,NEWLN
      BENDR=0.0174533*ABS(PIPE2)
      LBEND=PIPE1*BENDR
      ARBND=0.785398*PIPE3**2
      INRAD=PIPE1-0.5*PIPE3
      OTRAD=PIPE1+0.5*PIPE3
      RATIO=INRAD/OTRAD
      X=RATIO
      CALL GINERT(ABS(PIPE2),X,Y)
      INERT=(Y*(OTRAD-INRAD))/ARBND
      LPRME=LBEND/ARBND
      NEWLN=LPRME+INERT
      GAMMA=NEWLN/LPRME
      VALUE=GAMMA*(LBEND+2.0*PIPE4)
      AREAB=ARBND/SQRT(GAMMA)
      DIME=2.0*SQRT(AREAB/3.1415927)
      RETURN
      END
      SUBROUTINE BNSECT(J,ITYPE,POINT,PIPE1,PIPE2,PIPE3,PIPE4)
C       Computes plot coordinates for a bend
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
      REAL POINT(8,200)
      INTEGER*2 ITYPE(200)
C         BEND
C          FIRST STRAIGHT SECTION OF BEND
      IF(PIPE4.NE.0.0)  CALL STSECT(J,ITYPE,POINT,PIPE4,PIPE3)
C          CURVED SECTION OF BEND
      IF(PIPE2.GE.0.0)  THEN
       XC=X-SINA*PIPE1
       YC=Y+COSA*PIPE1
       DIA= 0.5
      ELSE
       XC=X+SINA*PIPE1
       YC=Y-COSA*PIPE1
       DIA=-0.5
      ENDIF
```

```
J=J+1
ITYPE(J)=0
POINT(1,J)=XC
POINT(2,J)=YC
POINT(3,J)=ANG
ANG=ANG+0.01745329*PIPE2
ANGLE=ANGLE+0.5*PIPE2
RANG=0.01745329*ANGLE
COSA=COS(RANG)
SINA=SIN(RANG)
RAD=PIPE1-DIA*PIPE3
POINT(4,J)=ANG
POINT(5,J)=RAD
X0=XC-RAD
Y0=YC+RAD
X1=XC+RAD
Y1=YC-RAD
X2=XH
Y2=YH
SLENTH=2.0*RAD*SIN(0.00872665*ABS(PIPE2))
XH=X2+COSA*SLENTH
YH=Y2+SINA*SLENTH
X3=XH
Y3=YH
IF(DIA.LT.0.0)  THEN
 HOLD=X2
 X2=X3
 X3=HOLD
 HOLD=Y2
 Y2=Y3
 Y3=HOLD
ENDIF
RAD=PIPE1+DIA*PIPE3
X0=XC-RAD
Y0=YC+RAD
X1=XC+RAD
Y1=YC-RAD
X2=XL
Y2=YL
SLENTH=2.0*RAD*SIN(0.00872665*ABS(PIPE2))
XL=X2+COSA*SLENTH
YL=Y2+SINA*SLENTH
X3=XL
Y3=YL
IF(DIA.LT.0.0)  THEN
 HOLD=X2
 X2=X3
 X3=HOLD
 HOLD=Y2
 Y2=Y3
 Y3=HOLD
ENDIF
```

```
      J=J+1
      ITYPE(J)=0
      POINT(1,J)=POINT(1,J-1)
      POINT(2,J)=POINT(2,J-1)
      POINT(3,J)=POINT(3,J-1)
      POINT(4,J)=POINT(4,J-1)
      POINT(5,J)=RAD
      SLENTH=2.0*PIPE1*SIN(0.00872665*ABS(PIPE2))
      X=X+COSA*SLENTH
      Y=Y+SINA*SLENTH
      XMIN=AMIN1(X,XL,XH,XMIN)
      XMAX=AMAX1(X,XL,XH,XMAX)
      YMIN=AMIN1(Y,YL,YH,YMIN)
      YMAX=AMAX1(Y,YL,YH,YMAX)
C         LAST STRAIGHT SECTION OF BEND
      ANGLE=ANGLE+0.5*PIPE2
      RANG=0.01745329*ANGLE
      COSA=COS(RANG)
      SINA=SIN(RANG)
      J=J+1
      ITYPE(J)=1
      POINT(1,J)=XH
      POINT(2,J)=YH
      POINT(3,J)=XL
      POINT(4,J)=YL
      X=X+COSA*PIPE4
      XH=X-0.5*SINA*PIPE3
      XL=X+0.5*SINA*PIPE3
      Y=Y+SINA*PIPE4
      YH=Y+0.5*COSA*PIPE3
      YL=Y-0.5*COSA*PIPE3
      POINT(5,J)=XH
      POINT(6,J)=YH
      POINT(7,J)=XL
      POINT(8,J)=YL
      XMIN=AMIN1(X,XL,XH,XMIN)
      XMAX=AMAX1(X,XL,XH,XMAX)
      YMIN=AMIN1(Y,YL,YH,YMIN)
      YMAX=AMAX1(Y,YL,YH,YMAX)
      RETURN
      END
      COMPLEX FUNCTION CCOSH(S)
C       Evaluates the complex hyperbolic cosine
      COMPLEX S
      REAL LAMDA, MU
      LAMDA=REAL(S)
      MU=AIMAG(S)
      COSHR=COSH(LAMDA)*COS(MU)
      COSHI=SINH(LAMDA)*SIN(MU)
      CCOSH=CMPLX(COSHR,COSHI)
      RETURN
      END
      COMPLEX FUNCTION CSINH(S)
C       Evaluates the complex hyperbolic sine
      COMPLEX S
      REAL LAMDA, MU
      LAMDA=REAL(S)
      MU=AIMAG(S)
      SINHR=SINH(LAMDA)*COS(MU)
      SINHI=COSH(LAMDA)*SIN(MU)
      CSINH=CMPLX(SINHR,SINHI)
      RETURN
      END
      COMPLEX FUNCTION CTANH(S)
C       Evaluates the complex hyperbolic tangent
      COMPLEX CCOSH,CSINH,S
      CTANH=CSINH(S)/CCOSH(S)
      RETURN
      END
      SUBROUTINE CURV(A1,A2)
C       Draws circular arc
      INCLUDE 'FGRAPH.FD'
      RECORD/WXYCOORD/XY
      INTEGER*2 DUMWIL
      COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
      REAL*8 XP,YP,A1,A2
      ANG1=A1
      ANG2=A2
      DTH=ANG2-ANG1
      IF(DTH.LT.0.0)  DTH=6.283185+DTH
      N=57.29578*DTH
      DA=DTH/(N-1)
      XP=XC+RAD*SIN(ANG1)
      YP=YC-RAD*COS(ANG1)
      CALL MOVETO_W(XP,YP,XY)
      DO 21 I=1,N-1
      T=ANG1+I*DA
      XP=XC+RAD*SIN(T)
      YP=YC-RAD*COS(T)
      DUMWIL=LINETO_W(XP,YP)
   21 CONTINUE
      RETURN
      END
      SUBROUTINE ENDPLT
C       Closes plot routines
      INCLUDE  'FGRAPH.FD'
      INTEGER*2            dummy
      READ (*,*)         ! Wait for ENTER key to be pressed
      dummy = setvideomode( $DEFAULTMODE )
      RETURN
      END
      LOGICAL FUNCTION fourcolors()
C       Determines type of graphics monitor
      INCLUDE  'FGRAPH.FD'
      INTEGER*2            dummy
      RECORD /videoconfig/ screen
      COMMON               screen
C
C     Set to maximum number of available colors.
C
      CALL getvideoconfig( screen )
      SELECT CASE( screen.adapter )
         CASE( $CGA, $OCGA )
            dummy = setvideomode( $MRES4COLOR )
         CASE( $EGA, $OEGA )
            dummy = setvideomode( $ERESCOLOR )
         CASE( $VGA, $OVGA )
            dummy = setvideomode( $VRES16COLOR )
         CASE DEFAULT
            dummy = 0
      END SELECT
      CALL getvideoconfig( screen )
      fourcolors = .TRUE.
      IF( dummy .EQ. 0 ) fourcolors = .FALSE.
      END
      SUBROUTINE FUEL(S,GF,PIPEA1,PIPEA2,PIPEA3,PIPEA4,SEGMNA,SECTNA,
     *                IGONE)
C       Handles fuel piping logic
      COMMON /WORKIT/WORK(12)
      COMPLEX GF,S
      REAL AREA(75),DIA(75),L(75),KMAN,PIND(75),PCAP(75)
      REAL DENS,A,LFLOW,KTANK,CMAN,CTANK,VOL,VOLMF
      REAL PIPEA1(75),PIPEA2(75),PIPEA3(75),PIPEA4(75),PIPEA5(75)
      INTEGER SEGMNA,SECTNA(75),SECTA
      CHARACTER*24 FUELIN,NAMLIN(2)
      COMMON /WCAOUT/NAMLIN,IUNIT
      CHARACTER*20 TITLF
      CHARACTER*1 ANS
      DATA ISTRT/0/
    1 FORMAT(E15.6)
    2 FORMAT(I5,4E15.6)
      IMORE=0
      IF(IGONE.EQ.2) THEN
       WRITE(*,'(A\)')'  Is fuel line data in a file? (Y/N) '
       READ(*,'(A)')ANS
       IF(ANS.NE.'N'.AND.ANS.NE.'n')  THEN
        WRITE(*,'(A\)')' Is the file name FUEL.INP? (Y/N) '
        READ(*,'(A)')ANS
        IF(ANS.NE.'N'.AND.ANS.NE.'n')  THEN
         OPEN(UNIT=11,FILE='FUEL.INP')
         NAMLIN(1)='FUEL.INP'
        ELSE
         WRITE(*,'(A\)')'  Enter name of file with fuel line data '
         READ(*,'(A)')FUELIN
         OPEN(11,FILE=FUELIN)
         NAMLIN(1)=FUELIN
     ENDIF
     IMORE=1
    ENDIF
     IGONE=0
   ENDIF
65 CONTINUE
   IF(ISTRT .EQ.0.AND. IGONE.EQ.0) THEN
    ISTRT=1
    IF(IMORE.EQ.1) GO TO 66
    CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
  *             PCHMB,DPROR)
    CALL MODIFY(AREA,DIA,L,PIPEA1,PIPEA2,PIPEA3,PIPEA4,PIPEA5,
  *   SECTNA,SEGMNA,SECTA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,PMRAT,'A')
    CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
  *             PCHMB,DPROR)
    IF(IUNIT.EQ.0) THEN
     WRITE(*,*)' You do not have any data stored, please re-read'
     WRITE(*,*)' the questions and answer carefully.'
     ISTRT=0
     WRITE(*,*)' '
     GOTO 65
    ENDIF
    REWIND 11
66  CONTINUE
    CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
  *             PCHMB,DPROR)
    CALL RLINE(TITLF,PMRAT,SEGMNA,SECTNA,PIPEA1,PIPEA2,
  *  PIPEA3,PIPEA4,PIPEA5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,
  *  SPLIT,11)
    CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
  *             PCHMB,DPROR)
    WRITE(*,*)'  For changes in fuel line data enter Y,'
    WRITE(*,'(A\)')'  if not, press enter key.'
    READ(*,'(A)')ANS
    WRITE(*,*)' '
    IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
     CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
  *             PCHMB,DPROR)
     CALL MODIFY(AREA,DIA,L,PIPEA1,PIPEA2,PIPEA3,PIPEA4,PIPEA5,
  *   SECTNA,SEGMNA,SECTA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,PMRAT,'A')
     CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
  *             PCHMB,DPROR)
    ENDIF
    RETURN
   ELSEIF(ISTRT .EQ. 1.AND. IGONE .EQ.0) THEN
    CALL ADMIT(S,GF,A,AREA,CMAN,CTANK,DPROR,L,LFLOW,PMRAT,SEGMNA,
  *            SECTNA,SPLIT,LOPEND,PCAP,PIND)
    RETURN
   ELSEIF(ISTRT .EQ. 1 .AND. IGONE .EQ. 1) THEN
    WRITE(*,'(A\)')' Do you wish to modify current fuel line data? '
    READ(*,'(A)')ANS
    IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y')  THEN
```

```
        CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *            PCHMB,DPROR)
        CALL MODIFY(AREA,DIA,L,PIPEA1,PIPEA2,PIPEA3,PIPEA4,PIPEA5,
     *   SECTNA,SEGMNA,SECTA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,PMRAT,'A')
        CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *            PCHMB,DPROR)
       ELSE
        WRITE(*,'(A\)')' Do you wish to rewind fuel line file? '
        READ(*,'(A)')ANS
        IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y')  REWIND 11
        CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *            PCHMB,DPROR)
        CALL RLINE(TITLF,PMRAT,SEGMNA,SECTNA,PIPEA1,PIPEA2,
     *   PIPEA3,PIPEA4,PIPEA5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,
     *   SPLIT,11)
        CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *            PCHMB,DPROR)
        WRITE(*,*)'  For changes in fuel line data enter Y,'
        WRITE(*,'(A\)')'  if not, press enter key.'
        READ(*,'(A)')ANS
        WRITE(*,*)' '
        IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
         CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *            PCHMB,DPROR)
        CALL MODIFY(AREA,DIA,L,PIPEA1,PIPEA2,PIPEA3,PIPEA4,PIPEA5,
     *    SECTNA,SEGMNA,SECTA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,PMRAT,'A')
         CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *            PCHMB,DPROR)
        ENDIF
       ENDIF
       IGONE=0
      ENDIF
      RETURN
      END
      SUBROUTINE GINERT(BEND,X,Y)
C       Evaluates curve fit of inertance of bends
      DIMENSION B(3)
      DATA B/0.0,0.7877014E-02,-0.2814679E-04/
      A=B(1)+(B(2)+B(3)*BEND)*BEND
      Y=A*(X-1.0)**2
      RETURN
      END
      SUBROUTINE HHSECT(J,ITYPE,POINT,LEN,DIA,VOL)
C       Computes plot coordinates for Helmholtz resonator
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      REAL LEN,POINT(8,200)
      INTEGER*2 ITYPE(200)
      XOLD=X
      XHOLD=XH
      XLOLD=XL
      YOLD=Y
      YHOLD=YH
      YLOLD=YL
      SINOLD=SINA
      COSOLD=COSA
      DIAM=SQRT((XH-XL)**2+(YH-YL)**2)
      CALL TSSECT(J,ITYPE,POINT,LEN,DIA)
      XC=0.5*(XOLD+X)
      YC=0.5*(YOLD+Y)
      XOLD=X
      YOLD=Y
      SINA=COSOLD
      COSA=-SINOLD
      X=XC+COSA*(LEN+0.5*DIAM)
      Y=YC+SINA*(LEN+0.5*DIAM)
      SIDE=VOL**0.3333333
      CALL STSECT(J,ITYPE,POINT,SIDE,SIDE)
      X=XOLD
      Y=YOLD
      SINA=SINOLD
      COSA=COSOLD
      DIAM=SQRT((XHOLD-XLOLD)**2+(YHOLD-YLOLD)**2)
      XH=X-0.5*SINA*DIAM
      XL=X+0.5*SINA*DIAM
      YH=Y+0.5*COSA*DIAM
      YL=Y-0.5*COSA*DIAM
      RETURN
      END
      SUBROUTINE LABANG(XMIN,XMAX,YMIN,YMAX)
C       Labels phase angle plot
      INCLUDE  'FGRAPH.FD'
      RECORD/WXYCOORD/XY
      RECORD /videoconfig/ screen
      COMMON               screen
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /NOCOL/NCOLS,NMODE
      COMMON /FACTOR/SFAC
      INTEGER*2 NCOLS
      INTEGER*2 row,rows
      INTEGER*2 DUMWIL
      RECORD/RCCOORD/S
      REAL*8 XMIN, XMAX, YMIN, YMAX, XP, YP
      CHARACTER*6 YLO,YHI
      CHARACTER*7 XHI
      DATA YLO/' -180°'/
      DATA YHI/'  180°'/
    1 FORMAT(F6.3)
    2 FORMAT(F7.2)
      rows    = screen.numtextrows
      IF(NMODE.EQ.6)  THEN
```

```
 CALL settextposition( 1, 1, s)
ELSE
 CALL settextposition( 0, 20, s)
ENDIF
CALL OUTTEXT(TITLE)
dummy = rectangle_w( $GBORDER, XMIN, YMIN, XMAX, YMAX )
row=rows/4
CALL SETTEXTPOSITION(row,1,s)
IF(NCOLS.LE.40)  THEN
 CALL OUTTEXT('Angle')
ELSE
 CALL OUTTEXT(' Phase Angle')
ENDIF
IF(NMODE.EQ.6)  THEN
 CALL SETTEXTPOSITION(rows/2-1,18,s)
 CALL OUTTEXT('freq')
ELSE
 CALL SETTEXTPOSITION(rows/2-1,35,s)
 IF(SFAC.EQ.1.0)  THEN
  CALL OUTTEXT('Frequency - rad/sec')
 ELSE
  CALL OUTTEXT('Frequency -  Hertz ')
 ENDIF
ENDIF
CALL GETTEXTPOSITION(s)
IF(NMODE.EQ.6)  THEN
 CALL SETTEXTPOSITION(3,1,s)
 CALL OUTTEXT(YHI)
 CALL SETTEXTPOSITION(s.row-3,1,s)
 CALL OUTTEXT(YLO)
 CALL GETTEXTPOSITION(s)
 ILOC=4
 IMAX=26
ELSEIF(NMODE.EQ.16)  THEN
 CALL SETTEXTPOSITION(2,10,s)
 CALL OUTTEXT(YHI)
 CALL SETTEXTPOSITION(s.row-2,10,s)
 CALL OUTTEXT(YLO)
 CALL GETTEXTPOSITION(s)
 ILOC=13
 IMAX=54
ELSE
 CALL SETTEXTPOSITION(2,10,s)
 CALL OUTTEXT(YHI)
 CALL SETTEXTPOSITION(s.row-2,10,s)
 CALL OUTTEXT(YLO)
 CALL GETTEXTPOSITION(s)
 ILOC=13
 IMAX=54
ENDIF
ILO=XMIN
IHI=XMAX
```

```
      IDEL=IMAX/(IHI-ILO)
      row=s.row+1
      DO 21 I=ILO,IHI
       HI=10.0**I
       WRITE(XHI,2)HI
       CALL SETTEXTPOSITION(row,ILOC,s)
       CALL OUTTEXT(XHI)
       ILOC=ILOC+IDEL
       IF(I.EQ.ILO.OR.I.EQ.IHI)  GO TO 21
       CALL SETLINESTYLE(62268)
       XP=I
       YP=YMIN
       CALL MOVETO_W(XP,YP,XY)
       YP=YMAX
       DUMWIL=LINETO_W(XP,YP)
       CALL SETLINESTYLE(65535)
   21 CONTINUE
      RETURN
      END
      SUBROUTINE LABGAIN(XMIN,XMAX,YMIN,YMAX,ITYPE)
C       Labels gain plot
      INCLUDE  'FGRAPH.FD'
      RECORD/WXYCOORD/XY
      RECORD /videoconfig/ screen
      COMMON               screen
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /NOCOL/NCOLS,NMODE
      COMMON /FACTOR/SFAC
      INTEGER*2 NCOLS
      INTEGER*2 row,rows
      INTEGER*2 DUMWIL
      RECORD/RCCOORD/S
      REAL*8 XMIN, XMAX, YMIN, YMAX, XP, YP
      CHARACTER*6 YLO,YHI
      CHARACTER*7 XHI
    1 FORMAT(F6.3)
    2 FORMAT(F7.2)
      rows    = screen.numtextrows
      dummy = rectangle_w( $GBORDER, XMIN, YMIN, XMAX, YMAX )
      row=rows/4
      CALL SETTEXTPOSITION(row,5,s)
      CALL OUTTEXT('Gain ')
      IF(NMODE.EQ.6)  THEN
       CALL SETTEXTPOSITION(rows/2-1,18,s)
       CALL OUTTEXT('freq')
       CALL SETTEXTPOSITION(rows,16,s)
      ELSE
       CALL SETTEXTPOSITION(rows/2-1,35,s)
```

```
 IF(SFAC.EQ.1.0)  THEN
  CALL OUTTEXT('Frequency - rad/sec')
 ELSE
  CALL OUTTEXT('Frequency -  Hertz ')
 ENDIF
 CALL SETTEXTPOSITION(rows,39,s)
ENDIF
IF(ITYPE.EQ.1)  CALL OUTTEXT('    K(jw)    ')
IF(ITYPE.EQ.2)  CALL OUTTEXT(' K(jw,Gox)   ')
IF(ITYPE.EQ.3)  CALL OUTTEXT('  K(jw,Gf)   ')
IF(ITYPE.EQ.4)  CALL OUTTEXT('K(jw,Gox,Gf)')
WRITE(YLO,1)YMIN
WRITE(YHI,1)YMAX
CALL GETTEXTPOSITION(s)
IF(NMODE.EQ.6)  THEN
 CALL SETTEXTPOSITION(3,1,s)
 CALL OUTTEXT(YHI)
 CALL SETTEXTPOSITION(s.row-3,1,s)
 CALL OUTTEXT(YLO)
 CALL GETTEXTPOSITION(s)
 ILOC=4
 IMAX=26
ELSEIF(NMODE.EQ.16)  THEN
 CALL SETTEXTPOSITION(3,10,s)
 CALL OUTTEXT(YHI)
 CALL SETTEXTPOSITION(s.row-4,10,s)
 CALL OUTTEXT(YLO)
 CALL GETTEXTPOSITION(s)
 ILOC=13
 IMAX=54
ELSE
 CALL SETTEXTPOSITION(2,10,s)
 CALL OUTTEXT(YHI)
 CALL SETTEXTPOSITION(s.row-3,10,s)
 CALL OUTTEXT(YLO)
 CALL GETTEXTPOSITION(s)
 ILOC=13
 IMAX=54
ENDIF
ILO=XMIN
IHI=XMAX
IDEL=IMAX/(IHI-ILO)
row=s.row+1
DO 21 I=ILO,IHI
 HI=10.0**I
 WRITE(XHI,2)HI
 CALL SETTEXTPOSITION(row,ILOC,s)
 CALL OUTTEXT(XHI)
 ILOC=ILOC+IDEL
 IF(I.EQ.ILO.OR.I.EQ.IHI)  GO TO 21
 CALL SETLINESTYLE(62268)
 XP=I
```

```
       YP=YMIN
       CALL MOVETO_W(XP,YP,XY)
       YP=YMAX
       DUMWIL=LINETO_W(XP,YP)
       CALL SETLINESTYLE(65535)
   21 CONTINUE
      RETURN
      END
      SUBROUTINE LOWERW(XMIN,XMAX,YMAX,YMIN)
C       Sets up lower plotting window
      INCLUDE  'FGRAPH.FD'
      INTEGER*2            dummy
      INTEGER*2            xwidth, yheight, cols, rows
      RECORD /videoconfig/ screen
      COMMON               screen
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 NCOLS,NMODE
      REAL*8 XMIN, XMAX, YMIN, YMAX, XLEN, YLEN
      XLEN=0.1*(XMAX-XMIN)
      YLEN=0.1*(YMAX-YMIN)
      XMIN=XMIN-XLEN
      XMAX=XMAX+XLEN
      YMIN=YMIN-YLEN
      YMAX=YMAX+YLEN
      xwidth  = screen.numxpixels
      yheight = screen.numypixels
      cols    = screen.numtextcols
      rows    = screen.numtextrows
C
C     window
C
      IF(NMODE.EQ.6)  THEN
       CALL setviewport( 50, yheight - 30, xwidth - 20, 10 )
      ELSE
       CALL setviewport( 100, yheight - 50, xwidth - 50, 20 )
      ENDIF
       CALL settextwindow(  0, 1, rows, cols)
      dummy = setwindow(.TRUE.,XMIN,YMIN,XMAX,YMAX)
      CALL clearscreen( $GWINDOW )
      RETURN
      END
      SUBROUTINE LOX(S,GOX,PIPEB1,PIPEB2,PIPEB3,PIPEB4,SEGMNB,SECTNB,
     *               IGONE)
C       Handles lox piping logic
      COMMON /WORKIT/WORK(12)
      COMPLEX GOX,S
      REAL AREA(75),DIA(75),L(75),PIND(75),PCAP(75)
      REAL DENS,A,LFLOW,KTANK,KMAN,CMAN,CTANK,VOL,VOLMF
      REAL PIPEB1(75),PIPEB2(75),PIPEB3(75),PIPEB4(75),PIPEB5(75)
      INTEGER SEGMNB,SECTNB(75),SECTB
      CHARACTER*24 LOXIN,NAMLIN(2)
      COMMON /WCAOUT/NAMLIN,IUNIT
      CHARACTER*20 TITLO
      CHARACTER*1 ANS
      DATA ISTRT/0/
    1 FORMAT(E15.6)
    2 FORMAT(I5,4E15.6)
      IMORE=0
      IF(IGONE.EQ.2) THEN
       WRITE(*,'(A\)')'  Is the lox line data in a file? (Y/N) '
       READ(*,'(A)')ANS
       IF(ANS.NE.'N'.AND.ANS.NE.'n') THEN
        WRITE(*,'(A\)')' Is the file with lox line data LOX.INP? (Y/N)'
        READ(*,'(A)')ANS
        IF(ANS.NE.'N'.AND.ANS.NE.'n') THEN
         OPEN(UNIT=10,FILE='LOX.INP')
         NAMLIN(2)='LOX.INP'
        ELSE
         WRITE(*,'(A\)')'  Enter name of file with lox line data '
         READ(*,'(A)')LOXIN
         OPEN(10,FILE=LOXIN)
         NAMLIN(2)=LOXIN
        ENDIF
        IMORE=1
       ENDIF
       IGONE=0
      ENDIF
   65 CONTINUE
      IF(ISTRT .EQ. 0.AND.IGONE.EQ.0) THEN
       ISTRT=1
       IF(IMORE.EQ.1)  GO TO 66
       CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
       CALL MODIFY(AREA,DIA,L,PIPEB1,PIPEB2,PIPEB3,PIPEB4,PIPEB5,
     *   SECTNB,SEGMNB,SECTB,PIND,PCAP,LOPEND,LOPOLD,SPLIT,PMRAT,'B')
       CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
       IF(IUNIT.EQ.0) THEN
        WRITE(*,*)' You do not have any data stored, please re-read'
        WRITE(*,*)' the questions and answer carefully.'
        ISTRT=0
        WRITE(*,*)' '
        GOTO 65
       ENDIF
       REWIND 10
   66  CONTINUE
       CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
       CALL RLINE(TITLO,PMRAT,SEGMNB,SECTNB,PIPEB1,PIPEB2,
     *  PIPEB3,PIPEB4,PIPEB5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,
     *  SPLIT,10)
       CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
       WRITE(*,*)'  For changes in lox line data enter Y,'
       WRITE(*,'(A\)')'  if not, press enter key.'
       READ(*,'(A)')ANS
       WRITE(*,*)' '
       IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
        CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
        CALL MODIFY(AREA,DIA,L,PIPEB1,PIPEB2,PIPEB3,PIPEB4,PIPEB5,
     *   SECTNB,SEGMNB,SECTB,PIND,PCAP,LOPEND,LOPOLD,SPLIT,PMRAT,'B')
        CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
       ENDIF
       RETURN
      ELSEIF(ISTRT .EQ. 1 .AND.IGONE.EQ.0) THEN
       CALL ADMIT(S,GOX,A,AREA,CMAN,CTANK,DPROR,L,LFLOW,PMRAT,SEGMNB,
     *            SECTNB,SPLIT,LOPEND,PCAP,PIND)
      ELSEIF(ISTRT.EQ.1.AND.IGONE.EQ.1) THEN
       WRITE(*,'(A\)')' Do you wish to modify current LOX line data? '
       READ(*,'(A)')ANS
       IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y')  THEN
        CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
        CALL MODIFY(AREA,DIA,L,PIPEB1,PIPEB2,PIPEB3,PIPEB4,PIPEB5,
     *   SECTNB,SEGMNB,SECTB,PIND,PCAP,LOPEND,LOPOLD,SPLIT,PMRAT,'B')
        CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
       ELSE
        WRITE(*,'(A\)')' Do you wish to rewind LOX line file? '
        READ(*,'(A)')ANS
        IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y')  REWIND 10
        CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
        CALL RLINE(TITLO,PMRAT,SEGMNB,SECTNB,PIPEB1,PIPEB2,
     *   PIPEB3,PIPEB4,PIPEB5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,
     *   SPLIT,10)
        CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
        WRITE(*,*)'  For changes in lox line data enter Y,'
        WRITE(*,'(A\)')'  if not, press enter key.'
        READ(*,'(A)')ANS
        WRITE(*,*)' '
        IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
         CALL WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
        CALL MODIFY(AREA,DIA,L,PIPEB1,PIPEB2,PIPEB3,PIPEB4,PIPEB5,
     *    SECTNB,SEGMNB,SECTB,PIND,PCAP,LOPEND,LOPOLD,SPLIT,PMRAT,'B')
         CALL WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,VOLMF,
     *             PCHMB,DPROR)
        ENDIF
       ENDIF
       IGONE=0
      ENDIF
      RETURN
      END
      SUBROUTINE MODIFY(AREA,DIA,L,PIPE1,PIPE2,PIPE3,PIPE4,PIPE5,SECTN,
     *           SEGMN,SECT,PIND,PCAP,LOPEND,LOPOLD,SPLIT,PMRAT,R)
C       Allows modifications to input data
      REAL AREA(75),DIA(75),L(75),PIPE1(75),PIPE2(75),PIPE3(75),
     *     PIPE4(75),PIPE5(75),PIND(75),PCAP(75)
      REAL KMAN,KTANK,LFLOW
      INTEGER*2 SECTN(75),SECT,SEGMN
      COMMON /WORKIT/A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,
     *               VOLMF,PCHMB,DPROR
      CHARACTER*1 ANS,R
      CHARACTER*8 VARVAL(9),VARU(9),VARL(9),NAME
      CHARACTER*24 NAMLIN(2)
      COMMON /WCAOUT/NAMLIN,IUNIT
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      DATA GRAV/32.2/,PI/3.141593/
      DATA VARVAL/'  DENS =',' DPROR =','  KMAN =',
     *            ' KTANK =',' LFLOW =',' PCHMB =',' TFLOW =',
     *            '   VOL =',' VOLMF ='/
      DATA VARU/'DENS    ','DPROR   ','KMAN    ',
     *          'KTANK   ','LFLOW   ','PCHMB   ','TFLOW   ',
     *          'VOL     ','VOLMF   '/
      DATA VARL/'dens    ','dpror   ','kman    ',
     *          'ktank   ','lflow   ','pchmb   ','tflow   ',
     *          'vol     ','volmf   '/
    1 FORMAT(1PE15.6)
    2 FORMAT(I5,1P5E15.6)
    3 FORMAT(I5,1P3E15.6)
    4 FORMAT(' This segment is a bend of',1PE13.5,' deg and radius of',
     *       E13.5)
    5 FORMAT(' This segment is straight ',1PE13.5,' diameter pipe ',
     *       E13.5,' ft. long')
    6 FORMAT(A8,1PE13.5,10X,A8,E13.5)
    7 FORMAT(' TITLE = ',A20)
   10 FORMAT(A20,2X,I2.2,':',I2.2,A2,3X,I2.2,'-',I2.2,'-',I2.2)
   11 FORMAT(' This segment is ',I2,' way split ',1PE13.5,' dia.',
     *       ' pipe ',E13.5,' ft. long')
   12 FORMAT(' This segment is a pump with length =',1PE13.5,' dia =',
     *       E13.5/5X,'dp/dm =',E13.5,' capacitance =',E13.5,
     *       ' inductance =',E13.5)
   13 FORMAT(' This segment is a tuned pipe ',1PE13.5,' long & dia =',
     *       E13.5)
   14 FORMAT(' This segment is a Helmholtz resonator with'/5X,'length ='
     *       ,1PE13.5,' dia =',E13.5,' and vol =',E13.5)
   15 FORMAT(' This segment is a parallel resonator with'/5X,'length =',
     *       1PE13.5,' dia =',E13.5,' and vol =',E13.5)
   16 FORMAT(' This segment is a',1PE13.5,' long inline acc. with',
     *       ' diameter of',E13.5)
```

```
      IF(R.EQ.'A')  THEN
       IUNIT=11
       NAMNAM=1
      ELSE
       IUNIT=10
       NAMNAM=2
      ENDIF
      AVGK=0.5*(KTANK+KMAN)
      ICHG=0
      WRITE(*,*)' Do you wish to change engine & fluid parameters '
      READ(*,'(A)')ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  GO TO 29
      WRITE(*,*)' Do you wish to change all of the parameters?'
      READ(*,'(A)')ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  ICHG=1
   21 CONTINUE
      IF(ICHG.EQ.0)  THEN
       WRITE(*,'(A\)')' Enter TITLE (20 characters max.) '
       READ(*,'(A)')TITLF
       WRITE(TITLE,10)TITLF,IHR,IMIN,AP,IMON,IDAY,IYR
       WRITE(*,'(A\)')' Enter FUEL TANK VOLUME (ft^3)'
       READ(*,*)VOL
       WRITE(*,'(A\)')' Enter FLOW RATE inside LINE (lbm/sec)'
       READ(*,*)LFLOW
       WRITE(*,'(A\)')' Enter BULK MODULUS of fluid inside TANK (lb /ft^
     *2)'
       READ(*,*)KTANK
       WRITE(*,'(A\)')' Enter FUEL DENSITY (lbm/ft^3)'
       READ(*,*)DENS
       WRITE(*,'(A\)')' Enter TOTAL FLOW RATE inside ENGINE (lbm/sec)'
       READ(*,*)TFLOW
       WRITE(*,'(A\)')' Enter MANIFOLD VOLUME (ft^3)'
       READ(*,*)VOLMF
       WRITE(*,'(A\)')' Enter BULK MODULUS of fluid inside MANIFOLD (lb
     */ft^2)'
       READ(*,*)KMAN
       WRITE(*,'(A\)')' Enter CHAMBER PRESSURE in ENGINE (lbf/ft^2)'
       READ(*,*)PCHMB
       WRITE(*,'(A\)')' Enter PRESSURE DROP across ORIFICE (lbf/ft^2)'
       READ(*,*)DPROR
       A=SQRT(GRAV*KTANK/DENS)
       CTANK=DENS*VOL/KTANK
       CMAN=DENS*VOLMF/KMAN
       PMRAT=PCHMB/TFLOW
       AVGK=0.5*(KTANK+KMAN)
      ELSE
       GO TO 24
   22  CONTINUE
       WRITE(*,*)'   VARIABLE NAMES AND DESCRIPTIONS'
       WRITE(*,*)' '
       WRITE(*,*)'   TITLE - title (20 characters max.)             '
       WRITE(*,*)'    DENS - density of fluid (lbm/ft^3)            '
      WRITE(*,*)'   DPROR - pressure drop across orfices (lbf/ft^2)'
      WRITE(*,*)'    KMAN - bulk modulus in manifold (lbf/ft^2)    '
      WRITE(*,*)'   KTANK - bulk modulus in tank (lbf/ft^2)        '
      WRITE(*,*)'   LFLOW - mass flow rate of fluid (lbm/sec)      '
      WRITE(*,*)'   PCHMB - chamber pressure (lbf/ft^2)            '
      WRITE(*,*)'   TFLOW - total mass flow inside engine (lbm/sec)'
      WRITE(*,*)'     VOL - volume of storage tank (ft^3)          '
      WRITE(*,*)'   VOLMF - volume of manifold (ft^3)              '
      WRITE(*,*)' '
      GO TO 25
   23 CONTINUE
      WRITE(*,*)'   VARIABLE NAMES AND VALUES'
      WRITE(*,*)' '
      WRITE(*,7)TITLF
      WRITE(*,6)VARVAL( 1), DENS,VARVAL( 2),DPROR,
     *          VARVAL( 3), KMAN,VARVAL( 4),KTANK,VARVAL( 5),LFLOW,
     *          VARVAL( 6),PCHMB,VARVAL( 7),TFLOW,VARVAL( 8),  VOL,
     *          VARVAL( 9),VOLMF
   24 CONTINUE
      WRITE(*,*)' '
      WRITE(*,*)' Enter ? to print variable names & descriptions'
      WRITE(*,*)'       # to print variable names & values'
      WRITE(*,*)'       TITLE to enter new title'
      WRITE(*,*)'       END when all changes have been made'
      WRITE(*,*)' '
   25 CONTINUE
      WRITE(*,'(A\)')'   Enter variable name and new value, END, ?, or
     * # '
      CALL ZREAD(NAME,VALUE)
      IF(NAME.EQ.'?')  GO TO 22
      IF(NAME.EQ.'#')  GO TO 23
      IF(NAME.EQ.'END'.OR.NAME.EQ.'end')  GO TO 28
      IF(NAME.EQ.'TITLE'.OR.NAME.EQ.'title')  THEN
       WRITE(*,'(A\)')' Enter new TITLE (20 characters max.) '
       READ(*,'(A)')TITLF
       WRITE(TITLE,10)TITLF,IHR,IMIN,AP,IMON,IDAY,IYR
       GO TO 25
      ENDIF
      DO 26 II=1,9
       I=II
       IF(NAME.EQ.VARU(I).OR.NAME.EQ.VARL(I))  GO TO 27
   26 CONTINUE
      WRITE(*,*)'     Invalid name, try again'
      GO TO 22
   27 CONTINUE
      IF(I.EQ. 1)  DENS=VALUE
      IF(I.EQ. 2)  DPROR=VALUE
      IF(I.EQ. 3)  KMAN=VALUE
      IF(I.EQ. 4)  KTANK=VALUE
      IF(I.EQ. 5)  LFLOW=VALUE
      IF(I.EQ. 6)  PCHMB=VALUE
      IF(I.EQ. 7)  TFLOW=VALUE
```

```
    IF(I.EQ. 8)  VOL=VALUE
    IF(I.EQ. 9)  VOLMF=VALUE
    GO TO 25
   ENDIF
28 CONTINUE
   A=SQRT(GRAV*KTANK/DENS)
   CTANK=DENS*VOL/KTANK
   CMAN=DENS*VOLMF/KMAN
   PMRAT=PCHMB/TFLOW
   AVGK=0.5*(KTANK+KMAN)
29 CONTINUE
   ICHG=0
   WRITE(*,*)' Do you wish to change the pipe layout? '
   READ(*,'(A)')ANS
   IF(ANS.NE.'Y'.AND.ANS.NE.'y')  GO TO 36
   WRITE(*,*)' Do you wish to change all of the pipe segments?'
   READ(*,'(A)')ANS
   IF(ANS.NE.'Y'.AND.ANS.NE.'y')  THEN
    ICHG=1
    GO TO 30
   ENDIF
    SPLIT=1.0
    LOPEND=1
    LOPOLD=20
   WRITE(*,'(A\)')' How many segments is the pipe broken into? '
   READ(*,*)SEGMN
30 CONTINUE
   I=0
   ISEGMN=SEGMN
   DO 35 II=1,SEGMN
    I=I+1
    IF(ICHG.EQ.1)  THEN
     IF(SECTN(I).EQ.0)  THEN
      WRITE(*,4)PIPE2(I),PIPE1(I)
     ELSEIF(SECTN(I).EQ.1)  THEN
      WRITE(*,5)PIPE2(I),PIPE1(I)
     ELSEIF(SECTN(I).EQ.2)  THEN
      WRITE(*,16)PIPE1(I),PIPE2(I)
     ELSEIF(SECTN(I).EQ.3)  THEN
      WRITE(*,13)PIPE1(I),PIPE2(I)
     ELSEIF(SECTN(I).EQ.4)  THEN
      WRITE(*,14)PIPE1(I),PIPE2(I),PIPE3(I)
     ELSEIF(SECTN(I).EQ.5)  THEN
      WRITE(*,15)PIPE1(I),PIPE2(I),PIPE3(I)
     ELSEIF(SECTN(I).EQ.6)  THEN
      WRITE(*,12)PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),PIPE5(I)
     ELSEIF(SECTN(I).EQ.9)  THEN
      WRITE(*,11)INT(PIPE3(I)),PIPE2(I),PIPE1(I)
     ENDIF
     WRITE(*,*)' You may keep (K), modify (Y), delete (D),',
  *            ' add before (B), or add after (A)?'
     READ(*,'(A)')ANS
     IF(ANS.EQ.'A'.OR.ANS.EQ.'a')  THEN
     I=I+1
      DO 31 III=ISEGMN,I,-1
       PIPE1(III+1)=PIPE1(III)
       PIPE2(III+1)=PIPE2(III)
       PIPE3(III+1)=PIPE3(III)
       PIPE4(III+1)=PIPE4(III)
       PIPE5(III+1)=PIPE5(III)
       L(III+1)=L(III)
       DIA(III+1)=DIA(III)
       AREA(III+1)=AREA(III)
       PCAP(III+1)=PCAP(III)
       PIND(III+1)=PIND(III)
       SECTN(III+1)=SECTN(III)
31 CONTINUE
      ISEGMN=ISEGMN+1
      GO TO 34
     ELSEIF(ANS.EQ.'B'.OR.ANS.EQ.'b')  THEN
      DO 32 III=ISEGMN,I,-1
       PIPE1(III+1)=PIPE1(III)
       PIPE2(III+1)=PIPE2(III)
       PIPE3(III+1)=PIPE3(III)
       PIPE4(III+1)=PIPE4(III)
       PIPE5(III+1)=PIPE5(III)
       L(III+1)=L(III)
       DIA(III+1)=DIA(III)
       AREA(III+1)=AREA(III)
       PCAP(III+1)=PCAP(III)
       PIND(III+1)=PIND(III)
       SECTN(III+1)=SECTN(III)
32 CONTINUE
      ISEGMN=ISEGMN+1
      GO TO 34
     ELSEIF(ANS.EQ.'D'.OR.ANS.EQ.'d')  THEN
      DO 33 III=I,ISEGMN
       PIPE1(III)=PIPE1(III+1)
       PIPE2(III)=PIPE2(III+1)
       PIPE3(III)=PIPE3(III+1)
       PIPE4(III)=PIPE4(III+1)
       PIPE5(III)=PIPE5(III+1)
       L(III)=L(III+1)
       DIA(III)=DIA(III+1)
       AREA(III)=AREA(III+1)
       PCAP(III)=PCAP(III+1)
       PIND(III)=PIND(III+1)
       SECTN(III)=SECTN(III+1)
33 CONTINUE
      I=I-1
      ISEGMN=ISEGMN-1
      GO TO 35
     ELSEIF(ANS.NE.'Y'.AND.ANS.NE.'y')  THEN
      GO TO 35
```

```
        ENDIF
       ENDIF
   34 CONTINUE
       WRITE(*,*)' Specify 0 for BEND,           1 for STRAIGHT pipe,'
       WRITE(*,*)'         2 for INLINE ACCUM.,  3 for TUNED STUB,'
       WRITE(*,*)'         4 for HELMHOLTZ RES., 5 for PARALLEL RES.'
       WRITE(*,*)'         6 for PUMP,           9 for SPLIT'
       READ(*,*) SECT
       IF(SECT.LT.0.OR.SECT.GT.6.AND.SECT.NE.9)  GO TO 34
       SECTN(I)=SECT
       IF(SECT.EQ.0)  THEN
C             BEND IN PIPE
        WRITE(*,*)' RADIUS of bend along CL (ft), ANGLE of bend (deg),'
        WRITE(*,*)' DIAMETER (ft), and LENGTH (ft) beyond bend of pipe'
        READ(*,*)PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I)
        CALL BENDS(PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),VALUE,DIME)
        AREAB=0.785398*DIME**2
        L(I)=VALUE
        AREA(I)=AREAB
        DIA(I)=DIME
        PIPE5(I)=0.0
       ELSEIF(SECT.EQ.1)  THEN
C             STRAIGHT SECTION
        WRITE(*,*)' Specify LENGTH (ft) and DIAMETER (ft) of segment'
        READ(*,*) PIPE1(I),PIPE2(I)
        VALUE=PIPE1(I)
        DIME=PIPE2(I)
        PIPE3(I)=0.0
        PIPE4(I)=0.0
        PIPE5(I)=0.0
        AREAB=0.785398*DIME**2
        L(I)=VALUE
        AREA(I)=AREAB
        DIA(I)=DIME
       ELSEIF(SECT.EQ.2)  THEN
C             INLINE ACCUMULATOR
        WRITE(*,*)' Specify LENGTH (ft) & DIAMETER (ft) of accumulator '
        READ(*,*) PIPE1(I),PIPE2(I)
        L(I)=PIPE1(I)
        DIA(I)=PIPE2(I)
        AREA(I)=0.25*PI*PIPE2(I)**2
        PCAP(I)=DENS*0.785398*L(I)*DIA(I)**2*PMRAT/AVGK
        PIPE3(I)=0.0
        PIPE4(I)=0.0
        PIPE5(I)=0.0
       ELSEIF(SECT.EQ.3)  THEN
C             TUNED STUB ACCUMULATOR
        WRITE(*,*)' Specify LENGTH (ft) & DIAMETER (ft) of tuned stub'
        READ(*,*)PIPE1(I),PIPE2(I)
        L(I)=PIPE1(I)
        DIA(I)=PIPE2(I)
        AREA(I)=0.25*PI*PIPE2(I)**2
        PCAP(I)=DENS*L(I)*AREA(I)*PMRAT/AVGK
        PIND(I)=L(I)/(AREA(I)*GRAV*PMRAT)
        PIPE3(I)=0.0
        PIPE4(I)=0.0
        PIPE5(I)=0.0
       ELSEIF(SECT.EQ.4)  THEN
C             HELMHOLTZ RESONATOR ACCUMULATOR
        WRITE(*,*)' Specify LENGTH (ft), DIAMETER (ft) ,VOLUME (ft^3)',
     *            ' of Helmholtz Resonator'
        READ(*,*)PIPE1(I),PIPE2(I),PIPE3(I)
        L(I)=PIPE1(I)
        DIA(I)=PIPE2(I)
        AREA(I)=PIPE3(I)
        PCAP(I)=DENS*L(I)*AREA(I)*PMRAT/AVGK
        PIND(I)=L(I)/(0.25*PI*DIA(I)**2*GRAV*PMRAT)
        PIPE4(I)=0.0
        PIPE5(I)=0.0
       ELSEIF(SECT.EQ.5)  THEN
C             PARALLEL RESONATOR ACCUMULATOR
        WRITE(*,*)' Specify LENGTH (ft), DIAMETER (ft) ,VOLUME (ft^3)',
     *            ' of Parallel Resonator'
        READ(*,*)PIPE1(I),PIPE2(I),PIPE3(I)
        L(I)=PIPE1(I)
        DIA(I)=PIPE2(I)
        AREA(I)=PIPE3(I)
        PCAP(I)=DENS*L(I)*AREA(I)*PMRAT/AVGK
        PIND(I)=L(I)/(0.25*PI*DIA(I)**2*GRAV*PMRAT)
        PIPE4(I)=0.0
        PIPE5(I)=0.0
       ELSEIF(SECT.EQ.6)  THEN
C             PUMP
        WRITE(*,*)' Specify LENGTH (ft), DIAMETER (ft) ,dp/dm, CAP.',
     *            ' & IND. of pump'
        READ(*,*)PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),PIPE5(I)
        L(I)=PIPE1(I)
        DIA(I)=PIPE2(I)
        AREA(I)=PIPE3(I)
        PCAP(I)=PIPE4(I)
        PIND(I)=PIPE5(I)
       ELSEIF(SECTN(I).EQ.9)  THEN
C             SPLIT PIPE
        WRITE(*,*)' Specify LENGTH (ft), DIAMETER (ft), and no. of',
     *            ' segments'
        READ(*,*) PIPE1(I),PIPE2(I),PIPE3(I)
        VALUE=PIPE1(I)
        DIME=PIPE2(I)
        SPLIT=PIPE3(I)
        WRITE(*,'(A,I3)')' Maximun no. of iterations is set at ',LOPOLD
        WRITE(*,'(A\)')' Do you wish to change it? '
        READ(*,'(A)')ANS
        IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
         WRITE(*,'(A\)')' Enter maximum no. of iterations '
         READ(*,*)LOPOLD
        ENDIF
        LOPEND=LOPOLD
        AREAB=0.785398*DIME**2
        L(I)=VALUE
        AREA(I)=AREAB
        DIA(I)=DIME
        PIPE4(I)=0.0
        PIPE5(I)=0.0
       ENDIF
 35   CONTINUE
      IF(ICHG.EQ.0)  THEN
       WRITE(*,*)'            NEW PIPE LAYOUT'
       WRITE(*,*)' STATUS   LENGTH          AREA          DIAMETER'
       DO 351 II=1,SEGMN
        WRITE(*,3)SECTN(I),L(I),AREA(I),DIA(I)
351    CONTINUE
      ENDIF
      SEGMN=ISEGMN
 36   CONTINUE
      WRITE(*,'(A\)')' Do you wish to save these changes? Y or N '
      READ(*,'(A)')ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  RETURN
      WRITE(*,'(A,A,A\)')' Do you wish to use file ',NAMLIN(NAMNAM),
     *                   '? Y or N '
      READ(*,'(A)')ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  THEN
       WRITE(*,'(A\)')' Enter name of file to use '
       READ(*,'(A)')NAMLIN(NAMNAM)
       CLOSE(UNIT=IUNIT)
       OPEN(UNIT=IUNIT,FILE=NAMLIN(NAMNAM))
      ELSE
       WRITE(*,'(A,A,A\)')' Do you wish to rewind ',NAMLIN(NAMNAM),
     *                    '? Y or N '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  REWIND IUNIT
      ENDIF
      WRITE(IUNIT,'(A)')TITLF
      WRITE(IUNIT,1)VOL
      WRITE(IUNIT,1)LFLOW
      WRITE(IUNIT,1)KTANK
      WRITE(IUNIT,1)DENS
      WRITE(IUNIT,1)TFLOW
      WRITE(IUNIT,1)VOLMF
      WRITE(IUNIT,1)KMAN
      WRITE(IUNIT,1)PCHMB
      WRITE(IUNIT,1)DPROR
      WRITE(IUNIT,2)SEGMN
      WRITE(IUNIT,2)(SECTN(I),PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),
     *               PIPE5(I),I=1,SEGMN)
      RETURN
      END
      SUBROUTINE NICEGRF(RMIN,RMAX,IMAX,IMMIN,ITYPE)
C       Plots Nyquist curve
      INCLUDE  'FGRAPH.FD'
      RECORD /videoconfig/ screen
      COMMON               screen
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /NOCOL/NCOLS,NMODE
      COMMON /FACTOR/SFAC
      INTEGER*2 NCOLS,NMODE
      INTEGER*2 row,rows
      RECORD/RCCOORD/S
      REAL*8 IMMIN,IMAX,RMIN,RMAX
      REAL*8 XMIN, XMAX, YMIN, YMAX
      CHARACTER*6 YLO,YHI,XLO,XHI
    1 FORMAT(F6.3)
      rows    = screen.numtextrows
      XMIN=RMIN
      XMAX=RMAX
      YMIN=IMMIN
      YMAX=IMAX
      IF(NMODE.EQ.6)  THEN
       CALL settextposition( 0, 1, s)
       CALL OUTTEXT(TITLE)
      ELSE
       CALL settextposition( 0, 20, s)
       CALL OUTTEXT(TITLE)
      ENDIF
      dummy = rectangle_w( $GBORDER, XMIN, YMIN, XMAX, YMAX )
      row=rows/2
      CALL SETTEXTPOSITION(row,1,s)
      IF(NMODE.EQ.6)  THEN
       CALL OUTTEXT('Imag')
       CALL SETTEXTPOSITION(rows-1,16,s)
       CALL OUTTEXT('    Real')
       CALL SETTEXTPOSITION(rows,16,s)
      ELSE
       CALL OUTTEXT('Imaginary')
       CALL SETTEXTPOSITION(rows-1,39,s)
       CALL OUTTEXT('    Real')
       CALL SETTEXTPOSITION(rows,39,s)
      ENDIF
      IF(ITYPE.EQ.1)  CALL OUTTEXT('    K(jw)   ')
      IF(ITYPE.EQ.2)  CALL OUTTEXT(' K(jw,Gox)  ')
      IF(ITYPE.EQ.3)  CALL OUTTEXT('  K(jw,Gf)  ')
      IF(ITYPE.EQ.4)  CALL OUTTEXT('K(jw,Gox,Gf)')
      WRITE(YLO,1)YMIN
      WRITE(YHI,1)YMAX
      WRITE(XLO,1)XMIN
      WRITE(XHI,1)XMAX
      CALL GETTEXTPOSITION(s)
      IF(NMODE.EQ.6)  THEN
       CALL SETTEXTPOSITION(s.row-3,1,s)
       CALL OUTTEXT(YLO)
       CALL GETTEXTPOSITION(s)
       CALL SETTEXTPOSITION(s.row+1,4,s)
       CALL OUTTEXT(XLO)
       CALL GETTEXTPOSITION(s)
       CALL SETTEXTPOSITION(s.row,35,s)
       CALL OUTTEXT(XHI)
       CALL SETTEXTPOSITION(3,1,s)
       CALL OUTTEXT(YHI)
      ELSE
       CALL SETTEXTPOSITION(s.row-3,5,s)
       CALL OUTTEXT(YLO)
       CALL GETTEXTPOSITION(s)
       CALL SETTEXTPOSITION(s.row+1,9,s)
       CALL OUTTEXT(XLO)
       CALL GETTEXTPOSITION(s)
       CALL SETTEXTPOSITION(s.row,71,s)
       CALL OUTTEXT(XHI)
       CALL SETTEXTPOSITION(2,5,s)
       CALL OUTTEXT(YHI)
      ENDIF
      RETURN
      END
      SUBROUTINE NYQUIS(GF,GOX,S,TAUT,CSTAR,RBAR,DCDR,THETAC,K,K1R,K2R,
     *K3R,K4R,K1C,K2C,K3C,K4C,IFUEL,ILOX)
C       Computes the K()'s
      COMPLEX GF,GOX,KG1,KG2,KG3,KG4,S
      REAL THETAC,RBAR,CSTAR,DCDR,TAUT
      REAL K1R(1001),K2R(1001),K3R(1001),K1C(1001),K2C(1001),K3C(1001)
      REAL K4R(1001),K4C(1001)
      KG1=2.0*CEXP(-S*TAUT)/(THETAC*S +1.0)
      K1C(K)=AIMAG(KG1)
      K1R(K)=REAL(KG1)
      IF(ILOX.EQ.0)  THEN
       KG2=0.5*KG1*((1.0+(1.0+RBAR)*DCDR/CSTAR)*GOX)
       K2C(K)=AIMAG(KG2)
       K2R(K)=REAL(KG2)
      ENDIF
      IF(IFUEL.EQ.0)  THEN
       KG3=0.5*KG1*((1.0-RBAR*(1.0+RBAR)*DCDR/CSTAR)*GF)
       K3C(K)=AIMAG(KG3)
       K3R(K)=REAL(KG3)
      ENDIF
      IF(ILOX.EQ.0.AND.IFUEL.EQ.0)  THEN
       KG4=KG2+KG3
       K4C(K)=AIMAG(KG4)
       K4R(K)=REAL(KG4)
      ENDIF
      RETURN
      END
      SUBROUTINE PIPPLOT(SEGMN,SECTN,PIPE1,PIPE2,PIPE3,PIPE4,ILOX,R)
C       Supervises plot of piping layout
      INCLUDE 'FGRAPH.FD'
      RECORD/WXYCOORD/XY
      INTEGER*2 DUMWIL
      COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      INTEGER*2 SEGMN,SECTN(75),ITYPE(200)
      REAL PIPE1(75),PIPE2(75),PIPE3(75),PIPE4(75)
      REAL*8 X0,X1,X2,X3,Y0,Y1,Y2,Y3
      REAL POINT(8,200)
      CHARACTER*1 R
      ANG=0.0
      ANGLE=0.0
      COSA=1.0
      SINA=0.0
      X=0.0
      XH=0.0
      XL=0.0
      Y=0.0
      IF(SECTN(1).EQ.0)  THEN
       YH=Y+0.5*PIPE3(1)
       YL=Y-0.5*PIPE3(1)
      ELSEIF(SECTN(1).GE.3.AND.SECTN(1).LE.5)  THEN
       IF(SECTN(2).EQ.0)  THEN
        YH=Y+0.5*PIPE3(2)
        YL=Y-0.5*PIPE3(2)
       ELSE
        YH=Y+0.5*PIPE2(2)
        YL=Y-0.5*PIPE2(2)
       ENDIF
      ELSE
       YH=Y+0.5*PIPE2(1)
       YL=Y-0.5*PIPE2(1)
      ENDIF
      J=0
      XMIN=0.0
      XMAX=0.0
      YMIN=AMIN1(Y,YL,YH)
      YMAX=AMAX1(Y,YL,YH)
      DO 21 I=1,SEGMN
       IF(SECTN(I).EQ.0)  THEN
C          BEND
        CALL BNSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I))
       ELSEIF(SECTN(I).EQ.1.OR.SECTN(I).EQ.9)  THEN
C          STRAIGHT SECTION
        CALL STSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ELSEIF(SECTN(I).EQ.2)  THEN
C          INLINE ACCUMULATOR
        CALL STSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ELSEIF(SECTN(I).EQ.3)  THEN
C         TUNED STUB ACCUMULATOR
        CALL TSSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ELSEIF(SECTN(I).EQ.4)  THEN
C         HELMHOLTZ RESONATOR
        CALL HHSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I),PIPE3(I))
       ELSEIF(SECTN(I).EQ.5)  THEN
C         PARALLEL RESONATOR
        CALL PLSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I),PIPE3(I))
       ELSEIF(SECTN(I).EQ.6)  THEN
C         PUMP
        CALL STSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ENDIF
   21 CONTINUE
      XRANGE=XMAX-XMIN
      YRANGE=YMAX-YMIN
      XMIN=XMIN-0.05*XRANGE
      XMAX=XMAX+0.05*XRANGE
      YMIN=YMIN-0.05*YRANGE
      YMAX=YMAX+0.05*YRANGE
      CALL UPPERW(XMIN,YMIN,XMAX,YMAX,ILOX,R)
      DO 24 I=1,J
       IF(ITYPE(I).EQ.0)  THEN
C         BEND
        XC=POINT(1,I)
        YC=POINT(2,I)
        X1=POINT(3,I)
        Y1=POINT(4,I)
        RAD=POINT(5,I)
        IF(X1.GT.Y1)  THEN
         X1=3.14159+X1
         Y1=3.14159+Y1
         CALL CURV(Y1,X1)
        ELSE
         CALL CURV(X1,Y1)
        ENDIF
       ELSE
C         ALL EXCEPT BEND
        X0=POINT(1,I)
        Y0=POINT(2,I)
        X1=POINT(3,I)
        Y1=POINT(4,I)
        X2=POINT(5,I)
        Y2=POINT(6,I)
        X3=POINT(7,I)
        Y3=POINT(8,I)
        CALL MOVETO_W(X0,Y0,XY)
        DUMWIL=LINETO_W(X1,Y1)
        CALL MOVETO_W(X2,Y2,XY)
        DUMWIL=LINETO_W(X3,Y3)
        CALL MOVETO_W(X0,Y0,XY)
        DUMWIL=LINETO_W(X2,Y2)
        CALL MOVETO_W(X1,Y1,XY)
        DUMWIL=LINETO_W(X3,Y3)
       ENDIF
   24 CONTINUE
      IF(R.EQ.'A')  THEN
       IF(ILOX.EQ.0)  RETURN
      ENDIF
      READ(*,*)
      RETURN
      END
      SUBROUTINE PLSECT(J,ITYPE,POINT,LEN,DIA,VOL)
C       Computes plot coordinates for parallel resonator
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
      REAL LEN,POINT(8,200)
      INTEGER*2 ITYPE(200)
      XOLD=X
      XHOLD=XH
      XLOLD=XL
      YOLD=Y
      YHOLD=YH
      YLOLD=YL
      ANGOLD=ANG
      ANGSAV=ANGLE
      SINOLD=SINA
      COSOLD=COSA
      DIAM=SQRT((XH-XL)**2+(YH-YL)**2)
      CALL STSECT(J,ITYPE,POINT,DIA,DIAM)
      XC=0.5*(XHOLD+XH)
      XHC=XHOLD
      XLC=XL
      YC=0.5*(YHOLD+YH)
      YHC=YHOLD
      YLC=YL
      PLEN=LEN-2.0*DIA
      PDIA=(VOL-2.0*DIA*DIAM)/PLEN
      CALL STSECT(J,ITYPE,POINT,PLEN,PDIA)
      CALL STSECT(J,ITYPE,POINT,DIA,DIAM)
      XSAV=X
      XHSAV=XH
      XLSAV=XL
      YSAV=Y
      YHSAV=YH
      YLSAV=YL
      SINA=COSOLD
      COSA=-SINOLD
      RADIUS=DIA
      TURN=-90.0
      SIDE=LEN-5.0*DIA
      ANG=ANG+1.5708
      ANGLE=ANGLE+90.0
      X=XC
      Y=YC
      XH=XHC
      XL=XLC
      YH=YHC
      YL=YLC
      CALL BNSECT(J,ITYPE,POINT,RADIUS,TURN,DIA,DIA)
      CALL STSECT(J,ITYPE,POINT,SIDE,DIA)
      CALL BNSECT(J,ITYPE,POINT,RADIUS,TURN,DIA,DIA)
      X=XSAV
      Y=YSAV
      XH=XHSAV
      XL=XLSAV
      YH=YHSAV
      YL=YLSAV
      ANG=ANGOLD
      ANGLE=ANGSAV
      SINA=SINOLD
      COSA=COSOLD
      RETURN
      END
      SUBROUTINE PNYQ(KR,KC,KW,PTS,ITYPE)
C       Plots gain and phase angle
      INCLUDE 'FGRAPH.FD'
      INTEGER PTS
      REAL KR(PTS),KC(PTS),KW(PTS),X(1001),YR(1001),YC(1001)
      RECORD/WXYCOORD/XY
      INTEGER*2 DUMWIL
      REAL*8 XMIN,XMAX,YMINR,YMAXR,YMINC,YMAXC,XP,YP,XLO,XHI
      DO 20 I=1,PTS
       YR(I)=SQRT(KR(I)**2+KC(I)**2)
       YC(I)=57.29578*ATAN2(KC(I),KR(I))
       X(I)=ALOG10(KW(I))
   20 CONTINUE
      YMINR=YR(1)
      YMAXR=YR(1)
      YMINC=-180.0
      YMAXC= 180.0
      XMIN=X(1)
      XMAX=X(1)
      DO 21 I=2,PTS
       IF(X(I).LT.XMIN)  XMIN=X(I)
       IF(X(I).GT.XMAX)  XMAX=X(I)
       IF(YR(I).LT.YMINR)  YMINR=YR(I)
       IF(YR(I).GT.YMAXR)  YMAXR=YR(I)
   21 CONTINUE
      XLO=XMIN
      XHI=XMAX
      DO 22 I=1,10
       IF(XMIN.GE.I)  XLO=I
       IF(XMAX.GE.I)  XHI=I
   22 CONTINUE
      IF(XMAX.NE.XHI)  XHI=XHI+1.0
      IF(XLO.EQ.XHI)  THEN
       XLO=XMIN
       XHI=XMAX
      ENDIF
      CALL WINDLO(XLO,XHI,YMINR,YMAXR)
      CALL LABGAIN(XLO,XHI,YMINR,YMAXR,ITYPE)
      CALL SETLINESTYLE(62268)
      IF(XMIN.LE.0.0.AND.XMAX.GE.0.0)  THEN
       XP=0.0
       YP=YMINR
       CALL MOVETO_W(XP,YP,XY)
       YP=YMAXR
       DUMWIL=LINETO_W(XP,YP)
      ENDIF
      IF(YMINR.LE.0.0.AND.YMAXR.GE.0.0)  THEN
       YP=0.0
       XP=XLO
       CALL MOVETO_W(XP,YP,XY)
       XP=XHI
       DUMWIL=LINETO_W(XP,YP)
      ENDIF
      CALL SETLINESTYLE(65535)
      XP=X(1)
      YP=YR(1)
      CALL MOVETO_W(XP,YP,XY)
      DO 23 I=2,PTS
       XP=X(I)
       YP=YR(I)
       DUMWIL=LINETO_W(XP,YP)
   23 CONTINUE
      CALL WINDUP(XLO,XHI,YMINC,YMAXC)
      CALL LABANG(XLO,XHI,YMINC,YMAXC)
      CALL SETLINESTYLE(62268)
      IF(XMIN.LE.0.0.AND.XMAX.GE.0.0)  THEN
       XP=0.0
       YP=YMINC
       CALL MOVETO_W(XP,YP,XY)
       YP=YMAXC
       DUMWIL=LINETO_W(XP,YP)
      ENDIF
      IF(YMINC.LE.0.0.AND.YMAXC.GE.0.0)  THEN
       YP=0.0
       XP=XLO
       CALL MOVETO_W(XP,YP,XY)
       XP=XHI
       DUMWIL=LINETO_W(XP,YP)
      ENDIF
      CALL SETLINESTYLE(65535)
      XP=X(1)
      YP=YC(1)
      CALL MOVETO_W(XP,YP,XY)
      DO 24 I=2,PTS
```

```
       XP=X(I)
       YP=YC(I)
       DUMWIL=LINETO_W(XP,YP)
   24 CONTINUE
      RETURN
      END
      SUBROUTINE RLINE(TITL,PMRAT,SEGMN,SECTN,PIPE1,PIPE2,PIPE3,
     *  PIPE4,PIPE5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,IUNIT)
C       Reads fuel or lox file
      REAL AREA(75),DIA(75),L(75),PIND(75),PCAP(75)
      REAL LFLOW,KTANK,KMAN
      REAL PIPE1(75),PIPE2(75),PIPE3(75),PIPE4(75),PIPE5(75)
      INTEGER SEGMN,SECTN(75)
      COMMON /WORKIT/A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,
     *             VOLMF,PCHMB,DPROR
      CHARACTER*20 TITL
      DATA GRAV/32.2/,PI/32.2/
    1 FORMAT(E15.6)
    2 FORMAT(I5,5E15.6)
C        TITLE
      READ(IUNIT,'(A)')TITL
C        TANK CONDITIONS
      READ(IUNIT,1)VOL
      READ(IUNIT,1)LFLOW
      READ(IUNIT,1)KTANK
C        MANIFOLD CONDITIONS
      READ(IUNIT,1)DENS
      READ(IUNIT,1)TFLOW
      READ(IUNIT,1)VOLMF
      READ(IUNIT,1)KMAN
      READ(IUNIT,1)PCHMB
C        ORFICE CONDITION
      READ(IUNIT,1)DPROR
      A=SQRT(GRAV*KTANK/DENS)
      CTANK=DENS*VOL/KTANK
      CMAN=DENS*VOLMF/KMAN
      PMRAT=PCHMB/TFLOW
      AVGK=0.5*(KTANK+KMAN)
      SPLIT=1.0
      LOPOLD=20
      LOPEND=1
C        PIPING
      READ(IUNIT,2)SEGMN
      DO 21 I=1,SEGMN
       READ(IUNIT,2)SECTN(I),PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),
     *              PIPE5(I)
       IF(SECTN(I).EQ.0)  THEN
        CALL BENDS(PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),VALUE,DIME)
        AREAB=0.785398*DIME**2
        L(I)=VALUE
        AREA(I)=AREAB
        DIA(I)=DIME
      ELSEIF(SECTN(I).EQ.1.OR.SECTN(I).EQ.9)  THEN
C            STRAIGHT SECTION  OR SPLIT
       VALUE=PIPE1(I)
       DIME=PIPE2(I)
       AREAB=0.785398*DIME**2
       L(I)=VALUE
       AREA(I)=AREAB
       DIA(I)=DIME
       IF(SECTN(I).EQ.9)  THEN
        SPLIT=PIPE3(I)
        WRITE(*,'(A,I3)')' Max. no. of iterations is set at ',LOPOLD
        WRITE(*,'(A\)')' Do you wish to change it? '
        READ(*,'(A)')ANS
        IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
         WRITE(*,'(A\)')' Enter maximum no. of iterations '
         READ(*,*)LOPOLD
        ENDIF
        LOPEND=LOPOLD
       ENDIF
      ELSEIF(SECTN(I).EQ.2)  THEN
C            INLINE ACCUMULATOR
C          PIPE1 - LEN   - L
C          PIPE2 - DIA   - DIA
C          PIPE3 - DEN
C          PIPE4 - K
       L(I)=PIPE1(I)
       DIA(I)=PIPE2(I)
       AREA(I)=0.25*PI*PIPE2(I)**2
       IF(PIPE3(I).EQ.0.0)  PIPE3(I)=DENS
       IF(PIPE4(I).EQ.0.0)  PIPE4(I)=AVGK
       PCAP(I)=PIPE3(I)*L(I)*AREA(I)*PMRAT/PIPE4(I)
      ELSEIF(SECTN(I).EQ.3)  THEN
C            TUNED STUB ACCUMULATOR
C              SUPPRESSES OMEGA = (PI/2)/(L*SQRT(PIND*PCAP))
C          PIPE1 - LEN   - L
C          PIPE2 - DIA   - DIA
C          PIPE3 - DEN
C          PIPE4 - K
       L(I)=PIPE1(I)
       DIA(I)=PIPE2(I)
       AREA(I)=0.25*PI*DIA(I)**2
       IF(PIPE3(I).EQ.0.0)  PIPE3(I)=DENS
       IF(PIPE4(I).EQ.0.0)  PIPE4(I)=AVGK
       PCAP(I)=PIPE3(I)*L(I)*AREA(I)*PMRAT/PIPE4(I)
       PIND(I)=L(I)/(AREA(I)*GRAV*PMRAT)
      ELSEIF(SECTN(I).EQ.4.OR.SECTN(I).EQ.5)  THEN
C            HELMHOLTZ RESONATOR ACCUMULATOR
C            PARALLEL RESONATOR ACCUMULATOR
C              SUPPRESSES OMEGA = 1/SQRT(PIND*PCAP)
C          PIPE1 - LEN   - L
C          PIPE2 - DIA   - DIA
C          PIPE3 - VOL   - AREA
```

```
C          PIPE4 - DEN
C          PIPE5 - K
        L(I)=PIPE1(I)
        DIA(I)=PIPE2(I)
        AREA(I)=PIPE3(I)
        IF(PIPE4(I).EQ.0.0)  PIPE4(I)=DENS
        IF(PIPE5(I).EQ.0.0)  PIPE5(I)=AVGK
        PCAP(I)=PIPE4(I)*AREA(I)*PMRAT/PIPE5(I)
        PIND(I)=L(I)/(0.25*PI*DIA(I)**2*GRAV*PMRAT)
       ELSEIF(SECTN(I).EQ.6)  THEN
C             PUMP
C          PIPE1 - LEN   - L
C          PIPE2 - DIA   - DIA
C          PIPE3 - DP/DM - AREA
C          PIPE4 - IND   - PIND
C          PIPE5 - CAP   - PCAP
        L(I)=PIPE1(I)
        DIA(I)=PIPE2(I)
        AREA(I)=PIPE3(I)
        PCAP(I)=PIPE4(I)*PMRAT
        PIND(I)=PIPE5(I)/PMRAT
       ENDIF
   21 CONTINUE
      RETURN
      END
      SUBROUTINE SETPLT
C       Sets up the plot environment
      INCLUDE 'FGRAPH.FD'
      RECORD /videoconfig/ screen
      COMMON               screen
      COMMON /WCAPAS/IFRST
      LOGICAL  fourcolors
      EXTERNAL fourcolors
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 NCOLS,NMODE
      IFRST=0
      IF( .NOT.fourcolors() ) THEN
       WRITE (*,*) ' This program requires a CGA, EGA, or',
     +              ' VGA graphics card.'
       STOP
      END IF
      NCOLS    = screen.numtextcols
      NMODE    = screen.mode
      RETURN
      END
      SUBROUTINE STSECT(J,ITYPE,POINT,LEN,DIA)
C       Computes plot coordinates for a straight section
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      REAL LEN,POINT(8,200)
      INTEGER*2 ITYPE(200)
      J=J+1
      ITYPE(J)=1
      XH=X-0.5*SINA*DIA
      XL=X+0.5*SINA*DIA
      YH=Y+0.5*COSA*DIA
      YL=Y-0.5*COSA*DIA
      POINT(1,J)=XH
      POINT(2,J)=YH
      POINT(3,J)=XL
      POINT(4,J)=YL
      X=X+COSA*LEN
      XH=X-0.5*SINA*DIA
      XL=X+0.5*SINA*DIA
      Y=Y+SINA*LEN
      YH=Y+0.5*COSA*DIA
      YL=Y-0.5*COSA*DIA
      POINT(5,J)=XH
      POINT(6,J)=YH
      POINT(7,J)=XL
      POINT(8,J)=YL
      XMIN=AMIN1(X,XL,XH,XMIN)
      XMAX=AMAX1(X,XL,XH,XMAX)
      YMIN=AMIN1(Y,YL,YH,YMIN)
      YMAX=AMAX1(Y,YL,YH,YMAX)
      RETURN
      END
      SUBROUTINE TSSECT(J,ITYPE,POINT,LEN,DIA)
C       Computes plot coordinates for a tuned stub
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      REAL LEN,POINT(8,200)
      INTEGER*2 ITYPE(200)
      J=J+1
      ITYPE(J)=1
      DIAM=SQRT((XH-XL)**2+(YH-YL)**2)
      XH=X-SINA*(LEN+0.5*DIAM)
      YH=Y+COSA*(LEN+0.5*DIAM)
      POINT(1,J)=XH
      POINT(2,J)=YH
      POINT(3,J)=XL
      POINT(4,J)=YL
      X=X+COSA*DIA
      XH=X-SINA*(LEN+0.5*DIAM)
      XL=XL+COSA*DIA
      Y=Y+SINA*DIA
      YH=Y+COSA*(LEN+0.5*DIAM)
      YL=YL+SINA*DIA
      POINT(5,J)=XH
      POINT(6,J)=YH
      POINT(7,J)=XL
      POINT(8,J)=YL
      XMIN=AMIN1(X,XL,XH,XMIN)
      XMAX=AMAX1(X,XL,XH,XMAX)
      YMIN=AMIN1(Y,YL,YH,YMIN)
      YMAX=AMAX1(Y,YL,YH,YMAX)
      RETURN
      END
      SUBROUTINE UPPERW(X00,Y00,X11,Y11,ILOX,R)
C       Sets up upper plotting window
      INCLUDE  'FGRAPH.FD'
      RECORD/RCCOORD/S
      INTEGER*2            dummy
      INTEGER*2            xwidth, yheight, cols, rows
      RECORD /videoconfig/ screen
      COMMON               screen
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 NCOLS,NMODE
      CHARACTER*2 AP
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      REAL*8 X0, X1, Y0, Y1
      CHARACTER*1 R
      xwidth  = screen.numxpixels
      yheight = screen.numypixels
      cols    = screen.numtextcols
      rows    = screen.numtextrows
      halfy   = yheight/2
      X0=X00
      Y0=Y00
      X1=X11
      Y1=Y11
      PICX=XWIDTH-20
      PICY=HALFY-30
      IF(NCOLS.LE.40)  PICY=HALFY-20
      XRANG=DABS(X1-X0)
      YRANG=DABS(Y1-Y0)
      XRAT=PICX/XRANG
      YRAT=PICY/YRANG
      IF(XRAT.LT.YRAT)  THEN
       YRAT=PICY/XRAT
       ADDY=0.5*(YRAT-YRANG)
       Y0=Y0-ADDY
       Y1=Y1+ADDY
      ELSE
       XRAT=PICX/YRAT
       ADDX=0.5*(XRAT-XRANG)
       X0=X0-ADDX
       X1=X1+ADDX
      ENDIF
C
C     window
C
      IF(R .EQ. 'A') THEN
       IF(NMODE.EQ.6)  THEN
        CALL setviewport( 10, halfy + 10, xwidth - 10, yheight - 10 )
        dummy = setwindow( .TRUE., X0-1.0, Y0-1.0, X1+1.0, Y1+1.0 )
        CALL settextwindow( (rows / 2 ) + 1, 1, rows, cols)
       ELSE
        CALL setviewport( 10, halfy + 10, xwidth - 10, yheight - 10 )
        dummy = setwindow( .TRUE., X0-1.0, Y0-1.0, X1+1.0, Y1+1.0 )
        CALL settextwindow( (rows / 2 ) + 1, 5, rows, cols - 5)
       ENDIF
       CALL clearscreen( $GWINDOW )
       IF(ILOX.EQ.0)  dummy = rectangle_w( $GBORDER, X0, Y0, X1, Y1 )
       IF(NMODE.EQ.6)  THEN
        CALL SETTEXTPOSITION(1,15,S)
       ELSE
        CALL SETTEXTPOSITION(1,30,S)
       ENDIF
       CALL OUTTEXT('FUEL PIPE LAYOUT')
      ENDIF
      IF(R.EQ.'B'.OR.ILOX.EQ.1)  THEN
       IF(NMODE.EQ.6)  THEN
        CALL setviewport( 10, 20, xwidth - 10, halfy  )
        dummy = setwindow( .TRUE., X0-1.0, Y0-1.0, X1+1.0, Y1+1.0 )
        CALL settextwindow(0 , 1, (rows / 2 ) , cols)
       ELSE
        CALL setviewport( 10, 25, xwidth - 10, halfy - 5 )
        dummy = setwindow( .TRUE., X0-1.0, Y0-1.0, X1+1.0, Y1+1.0 )
        CALL settextwindow(0 , 1, (rows / 2 ) , cols - 5)
       ENDIF
       CALL clearscreen( $GWINDOW )
       dummy = rectangle_w( $GBORDER, X0, Y0, X1, Y1 )
       IF(NMODE.EQ.6)  THEN
        CALL SETTEXTPOSITION(0,1,S)
       ELSE
        CALL SETTEXTPOSITION(0,20,S)
       ENDIF
       CALL OUTTEXT(TITLE)
       IF(NMODE.EQ.6)  THEN
        CALL SETTEXTPOSITION(2,15,S)
       ELSE
        CALL SETTEXTPOSITION(2,30,S)
       ENDIF
       IF(ILOX.EQ.0)  CALL OUTTEXT('LOX  PIPE LAYOUT')
      ENDIF
      RETURN
      END
      SUBROUTINE WINDLO(XMIN,XMAX,YMIN,YMAX)
C       Sets up gain window
      INCLUDE  'FGRAPH.FD'
      INTEGER*2            dummy
      INTEGER*2            xwidth, yheight, cols, rows, halfy
      RECORD /videoconfig/ screen
      COMMON               screen
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 NCOLS
      REAL*8 XMIN, XMAX, YMIN, YMAX, XLEN, YLEN
      REAL*8 XMINP, XMAXP, YMINP, YMAXP
      XLEN=0.1*(XMAX-XMIN)
      YLEN=0.1*(YMAX-YMIN)
      XMINP=XMIN-XLEN
      XMAXP=XMAX+XLEN
      YMINP=YMIN-YLEN
      YMAXP=YMAX+YLEN
      xwidth  = screen.numxpixels
      yheight = screen.numypixels
      cols    = screen.numtextcols
      rows    = screen.numtextrows
      halfy   = yheight/2
C
C     window
C
      IF(NCOLS.LE.40)  THEN
       CALL setviewport( 50, halfy + 10, xwidth - 20, yheight - 30 )
      ELSE
       CALL setviewport( 100, halfy + 10, xwidth - 50, yheight - 50 )
      ENDIF
       CALL settextwindow( (rows / 2 ) + 1, 1, rows, cols - 1)
      dummy = setwindow(.TRUE.,XMINP,YMINP,XMAXP,YMAXP)
      CALL clearscreen( $GWINDOW )
      RETURN
      END
      SUBROUTINE WINDUP(XMIN,XMAX,YMIN,YMAX)
C       Sets up phase angle window
      INCLUDE  'FGRAPH.FD'
      INTEGER*2            dummy
      INTEGER*2            xwidth, yheight, cols, rows, halfy
      RECORD /videoconfig/ screen
      COMMON               screen
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 NCOLS
      REAL*8 XMIN, XMAX, YMIN, YMAX, XLEN, YLEN
      REAL*8 XMINP, XMAXP, YMINP, YMAXP
      XLEN=0.1*(XMAX-XMIN)
      YLEN=0.1*(YMAX-YMIN)
      XMINP=XMIN-XLEN
      XMAXP=XMAX+XLEN
      YMINP=YMIN-YLEN
      YMAXP=YMAX+YLEN
      xwidth  = screen.numxpixels
      yheight = screen.numypixels
      cols    = screen.numtextcols
      rows    = screen.numtextrows
      halfy   = yheight/2
C
C     window
C
      IF(NCOLS.LE.40)  THEN
       CALL setviewport( 50, 10, xwidth - 20, halfy - 30 )
      ELSE
       CALL setviewport( 100, 10, xwidth - 50, halfy - 50 )
      ENDIF
      CALL settextwindow( 1, 1, (rows / 2 ) - 1, cols - 1)
      dummy = setwindow(.TRUE.,XMINP,YMINP,XMAXP,YMAXP)
      CALL clearscreen( $GWINDOW )
      RETURN
      END
      SUBROUTINE WORKFR(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,
     *                  VOLMF,PCHMB,DPROR)
C       Moves arguments from common /WORKIT/
      COMMON /WORKIT/WORK(12)
      REAL KMAN,KTANK,LFLOW
      A=WORK(1)
      CMAN=WORK(2)
      CTANK=WORK(3)
      DENS=WORK(4)
      KMAN=WORK(5)
      KTANK=WORK(6)
      LFLOW=WORK(7)
      TFLOW=WORK(8)
      VOL=WORK(9)
      VOLMF=WORK(10)
      PCHMB=WORK(11)
      DPROR=WORK(12)
      RETURN
      END
      SUBROUTINE WORKTO(A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,TFLOW,VOL,
     *                  VOLMF,PCHMB,DPROR)
C       Moves arguments to common /WORKIT/
      COMMON /WORKIT/WORK(12)
      REAL KMAN,KTANK,LFLOW
      WORK(1)=A
      WORK(2)=CMAN
      WORK(3)=CTANK
      WORK(4)=DENS
      WORK(5)=KMAN
      WORK(6)=KTANK
      WORK(7)=LFLOW
      WORK(8)=TFLOW
      WORK(9)=VOL
      WORK(10)=VOLMF
      WORK(11)=PCHMB
      WORK(12)=DPROR
      RETURN
      END
      SUBROUTINE ZREAD(NAME,VALUE)
C       Reads input for input modification
      CHARACTER*1 NAME(8)
      CHARACTER*1 CARD(80),PLUS,MINUS,PERIOD,LE,E,NUMBER(10)
      CHARACTER*1 LEND(3),CEND(3),POUND,QUEST,BLK,COMMA
      CHARACTER*1 LTIT(5),CTIT(5)
      CHARACTER*80 DCARD
      EQUIVALENCE (CARD(1),DCARD)
      DATA PLUS/'+'/,MINUS/'-'/,PERIOD/'.'/,LE/'e'/,E/'E'/,BLK/' '/
      DATA NUMBER/'0','1','2','3','4','5','6','7','8','9'/,COMMA/','/
      DATA LEND/'e','n','d'/,CEND/'E','N','D'/,POUND/'#'/,QUEST/'?'/
      DATA LTIT/'t','i','t','l','e'/,CTIT/'T','I','T','L','E'/
    1 FORMAT(A)
      DO 21 I=1,8
       NAME(I)=BLK
   21 CONTINUE
      READ(*,1)DCARD
      IF(CARD(1).EQ.POUND)  THEN
       NAME(1)=POUND
       RETURN
      ENDIF
      IF(CARD(1).EQ.QUEST)  THEN
       NAME(1)=QUEST
       RETURN
      ENDIF
      DO 22 I=1,3
       IF(CARD(I).NE.LEND(I).AND.CARD(I).NE.CEND(I))  GO TO 220
       NAME(I)=CEND(I)
   22 CONTINUE
      RETURN
  220 CONTINUE
      DO 221 I=1,5
       IF(CARD(I).NE.LTIT(I).AND.CARD(I).NE.CTIT(I))  GO TO 23
       NAME(I)=CTIT(I)
  221 CONTINUE
      RETURN
   23 CONTINUE
      DO 24 I=1,8
       II=I
       IF(CARD(I).EQ.BLK.OR.CARD(I).EQ.COMMA)  GO TO 25
       NAME(I)=CARD(I)
   24 CONTINUE
   25 CONTINUE
      DO 26 I=II,80
       ID=I
       IF(CARD(I).NE.BLK.AND.CARD(I).NE.COMMA)  GO TO 27
   26 CONTINUE
      VALUE=0.0
      WRITE(*,*)'   No value given, ZERO assumed'
      RETURN
   27 CONTINUE
      SIGN=1.0
      IF(CARD(ID).EQ.MINUS)  THEN
       SIGN=-1.0
       ID=ID+1
      ELSEIF(CARD(ID).EQ.PLUS)  THEN
       ID=ID+1
      ENDIF
   WHOLE=0.0
   DO 30 I=ID,80
    II=I
    IF(CARD(I).EQ.PERIOD)  GO TO 31
    IF(CARD(I).EQ.PLUS)  GO TO 36
    IF(CARD(I).EQ.MINUS)  GO TO 36
    IF(CARD(I).EQ.E.OR.CARD(I).EQ.LE)  GO TO 35
    DO 28 J=1,10
     JJ=J-1
     IF(CARD(I).EQ.NUMBER(J))  GO TO 29
28 CONTINUE
    VALUE=SIGN*WHOLE
    IF(CARD(I).EQ.BLK)  RETURN
    WRITE(*,*)'  Input error, value set to ZERO'
    VALUE=0.0
    RETURN
29 CONTINUE
    WHOLE=WHOLE*10.0+JJ
30 CONTINUE
   VALUE=SIGN*WHOLE
   RETURN
31 CONTINUE
   ID=II+1
   FRACT=0.0
   ICOUNT=0
   DO 34 I=ID,80
    ICOUNT=ICOUNT+1
    II=I
    IF(CARD(I).EQ.PERIOD)  THEN
     WRITE(*,*)'  Input error, value set to ZERO'
     VALUE=0.0
     RETURN
    ENDIF
    IF(CARD(I).EQ.PLUS)  GO TO 36
    IF(CARD(I).EQ.MINUS)  GO TO 36
    IF(CARD(I).EQ.E.OR.CARD(I).EQ.LE)  GO TO 35
    DO 32 J=1,10
     JJ=J-1
     IF(CARD(I).EQ.NUMBER(J))  GO TO 33
32 CONTINUE
    VALUE=SIGN*(WHOLE+FRACT)
    IF(CARD(I).EQ.BLK)  RETURN
    WRITE(*,*)'  Input error, value set to ZERO'
    VALUE=0.0
    RETURN
33 CONTINUE
    FRACT=FRACT+JJ/10.0**ICOUNT
34 CONTINUE
   VALUE=SIGN*(WHOLE+FRACT)
   RETURN
35 CONTINUE
   II=II+1
```

```
36 CONTINUE
   VALUE=SIGN*(WHOLE+FRACT)
   SIGN=1.0
   IF(CARD(II).EQ.MINUS)  THEN
    SIGN=-1.0
    II=II+1
   ELSEIF(CARD(II).EQ.PLUS)  THEN
    II=II+1
   ENDIF
   WHOLE=0.0
   DO 39 I=II,80
    DO 37 J=1,10
     JJ=J-1
     IF(CARD(I).EQ.NUMBER(J))  GO TO 38
37 CONTINUE
    VALUE=VALUE*10.0**(SIGN*WHOLE)
    IF(CARD(I).EQ.BLK)  RETURN
    WRITE(*,*)'  Input error, value set to ZERO'
    VALUE=0.0
    RETURN
38 CONTINUE
    WHOLE=WHOLE*10.0+JJ
39 CONTINUE
   VALUE=VALUE*10.0**(SIGN*WHOLE)
   RETURN
   END
```

Appendix D

Listing of Intermediate Frequency Program

SFREQ

```
C
C      PROGRAM SFREQ
C
C                  Intermediate Mode Oscillations
C
C                   Modified for n vs tau plots
C
C
C                       Variables in Commons
C
C                             /CMPVAL/
C   CVAR(17)         COMPLEX*8  equivalence(CVAR(1),X1)
C   X1               COMPLEX*8  first order term of x
C   Y1               COMPLEX*8  first order term of y
C   Z1               COMPLEX*8  first order term of z
C   W1               COMPLEX*8  first order term of w
C   M1               COMPLEX*8  first order term of m
C   P0               COMPLEX*8  zeroth order term of pressure
C   P1               COMPLEX*8  first order term of pressure
C   U0               COMPLEX*8  zeroth order term of velocity
C   U1               COMPLEX*8  first order term of velocity
C   RFH              COMPLEX*8  combustion response function for mixture ratio
C   RFK              COMPLEX*8  compustion response function for mass flow
C   RFP              COMPLEX*8  combustion response function for pressure
C   S                COMPLEX*8  lamda + mu I - perturbation oscillation
C   GF               COMPLEX*8  admittance of fuel line looking toward tank
C   GOX              COMPLEX*8  admittance of lox line looking toward tank
C   RFA              COMPLEX*8  nozzle pressure admittance coefficient
C   RFC              COMPLEX*8  nozzle entropy admittance coefficient
C
C                             /DIMVAL/
C   AJUNK1(8)        REAL*4     equivalence(AJUNK1(1),ND)
C   HOLDD(20)        REAL*4     equivalence(HOLDD(1),ND)
C   ND               REAL*4     pressure interaction index
C   TAUD             REAL*4     sensitive time lag (sec)
C   DTAUD            REAL*4     delta time lag (sec)
C   NRD              REAL*4     enthalpy interaction index
C   LAMDAD           REAL*4     damping of perturbation
C   MUD              REAL*4     frequency of perturbation (rad/sec)
C   CDIAM            REAL*4     chamber diameter (ft)
C   TDIAM            REAL*4     throat diameter (ft)
C   XLCD             REAL*4     x location of chamber-nozzle interface (ft)
C   AJUNK2(161)      REAL*4     equivalence(AJUNK2(1),GAMMAD)
C   GAMMAD           REAL*4     ratio of specific heats
C   RGAS             REAL*4     gas constant (ft^2/sec^2/°R)
C   P00D             REAL*4     maximum pressure at injection face (lbf/ft^2)
C   MBARD            REAL*4     mean combustion response function (lbm/sec)
C   RBARD            REAL*4     mean mixture ratio
C   DCSDRD           REAL*4     d(cstar)/d(mixture ratio) (ft/sec)
C   DHLDRD           REAL*4     d(enthalpy/d(mixture ratio) (ft^2/sec^2)
C   RHOLOD           REAL*4     mass of liquid per unit chamber vol (lbm/ft^3)
C   ULOD             REAL*4     axial component of liquid velocity (ft/sec)
```

```
C   PCHMB            REAL*4     chamber pressure (lbf/ft^2)
C   TCHMB            REAL*4     chamber temperature (°R)
C   XBARD(50)        REAL*4     x locations along axis (ft)
C   PBAR(50)         REAL*4     pressure along axis (lbf/ft^2)
C   TBAR(50)         REAL*4     temperature along axis (°R)
C
C                             /FFACT/
C   FFAC             REAL*4     factor for frequency
C
C                             /NVAL/
C   NVAL             INTEGER*2  number of input points along axis
C
C                             /PIPES/
C   PFACE            REAL*4     pressure at injector face (lbf/ft^2)
C   TFACE            REAL*4     mean combustion response function (lbm/sec)
C   ASTAR            REAL*4     speed of sound at injector face (ft/sec)
C
C                             /RELVAL/
C   RVAR(13)         REAL*4     equivalence(RVAR(1),N)
C   N                REAL*4     pressure interaction index
C   TAU              REAL*4     sensitive time lag
C   DTAU             REAL*4     delta time lag
C   NR               REAL*4     enthalpy interaction index
C   RBAR             REAL*4     mean mixture ratio
C   MBAR             REAL*4     mean combustion response function
C   GAMMA            REAL*4     ratio of specific heats
C   P00              REAL*4     maximum pressure at injection face
C   DHLDR            REAL*4     d(enthalpy)/d(mixture ratio)
C   CSTAR            REAL*4     characteristic velocity at combustor exit
C   DCSDR            REAL*4     d(cstar)/d(mixture ratio)
C   RHOLO            REAL*4     mass of liquid per unit chamber volume
C   ULO              REAL*4     axial component of liquid velocity
C   LAMDA            REAL*4     damping of perturbation
C   MU               REAL*4     frequency of perturbation
C   TAUT             REAL*4     total time lag
C   UBAR(50)         REAL*4     velocity along axis
C   XBAR(50)         REAL*4     x locations along axis
C   XLC              REAL*4     x location of chamber-nozzle interface
C
C                             /RESULT/
C   PP               COMPLEX*8  P' = P0 + P1
C   UP               COMPLEX*8  U' = U0 + U1
C   SIGP             COMPLEX*8  SIG' = SIG0 + SIG1
C   FUNB             COMPLEX*8  boundary function U' + RFA * P' + RFC * SIG'
C
C                             /TITL/
C   TITLE            CHAR*60    title for plots including date andd time
C   TITLF            CHAR*40    input title
C   IHR              INTEGER*2  hour code run
C   IMIN             INTEGER*2  minute code run
C   AP               CHAR*2     AM or PM
C   IYR              INTEGER*2  yesr code run
```

```
C   IMON               INTEGER*2  month code run
C   IDAY               INTEGER*2  day code run
C
C
C   PROGRAM SFREQ
C       Logic portion of code
C
C   Commons CMPVAL  DIMVAL  FFACT   INTVAL  RELVAL  RESULT  TITL
C                   Local Variables
C   AM               CHAR*2     'AM'
C   ANS              CHAR*1     response to question
C   DELF             REAL*4     intermediate variable
C   DELVAL           REAL*4     intermediate variable
C   FREQ(50)         REAL*4     array of frequencies
C   I                INTEGER*2  do loop index
C   ID               INTEGER*2  flag for dependent variable
C   II               INTEGER*2  flag for independent variable
C   ISEC             INTEGER*2  seconds at start
C   I100             INTEGER*2  hundreds of seconds at start
C   J                INTEGER*2  do loop index
C   NOF              INTEGER*2  maximum number of frequencies
C   NOT              INTEGER*2  maximum number of tau's
C   NPTF             INTEGER*2  number of frequencies
C   NPTS             INTEGER*2  number of tau's
C   PM               CHAR*2     'PM'
C   RADHER(2)        CHAR*8     labels
C   ROCIN            CHAR*24    input file name
C   ROCOUT           CHAR*24    output file name
C   ROCVAR           CHAR*24    file name for frequencies or tau's
C   STARTF           REAL*4     starting frequency
C   STARTV           REAL*4     starting tau
C   STOPF            REAL*4     ending frequency
C   STOPV            REAL*4     ending tau
C   TAULST(200)      REAL*4     array of tau's
C   TOL              REAL*4     convergence criteria
C   YP(200,50)       REAL*4     array of n's
C   VARP(3)          CHAR*8     labels
C   VAR1             REAL*4     intermediate variable
C
C
C   SUBROUTINE ADMIT(S,GADM,A,AREA,CMAN,CTANK,DPROR,L,LFLOW,PCHMB,SEGMN,TFLOW)
C       determines admittance looking toward tank
C
C   Commons DIMVAL  PIPES
C                   Variables in Argument List
C   A                REAL*4     speed of sound in the fluid
C   AREA(75)         REAL*4     area of pipe section
C   CMAN             REAL*4     manifold capacitance
C   CTANK            REAL*4     tank capacitance
C   DPROR            REAL*4     pressure drop across orfices
C   GADM             COMPLEX*8  admittance of line looking toward tank
C   L(75)            REAL*4     length of pipe section
```

```
C   LFLOW                REAL*4     flow rate through pipe
C   PCHMB                REAL*4     chamber pressure
C   S                    COMPLEX*8  complex frequency
C   SEGMN                INTEGER*2  number of pipe sections
C   TFLOW                REAL*4     total flow rate of engine
C                       Local Variables
C   G(76)                COMPLEX*8  admittance looking toward tank
C   GRAV                 REAL*4     gravitational constant (lbm-ft/lbf-sec^2)
C   I                    INTEGER*2  do loop index
C   TL                   REAL*4     intermediate variable
C   W                    COMPLEX*8  normalized frequency
C   ZLINE                REAL*4     intermediate variable
C   ZOR                  REAL*4     intermediate variable
C   ZTOP                 REAL*4     intermediate variable
C
C
C   SUBROUTINE BENDS(PIPE1,PIPE2,PIPE3,PIPE4,VALUE,DIME)
C       Computes effective straight pipe for bend
C
C                       Variables in Argument List
C   DIME                 REAL*4     effective diameter (ft)
C   PIPE1                REAL*4     radius of bend (ft)
C   PIPE2                REAL*4     angle of bend (degrees)
C   PIPE3                REAL*4     diameter of bend (ft)
C   PIPE4                REAL*4     length of end straight segments (ft)
C   VALUE                REAL*4     effective length (ft)
C                       Local Variables
C   ARBND                REAL*4     area of bend
C   AREAB                REAL*4     effective area of bend
C   BENDR                REAL*4     bend angle in radians
C   GAMMA                REAL*4     intermediate variable
C   INERT                REAL*4     intermediate variable
C   INRAD                REAL*4     inside radius of bend
C   LBEND                REAL*4     intermediate variable
C   LPRME                REAL*4     intermediate variable
C   NEWLN                REAL*4     intermediate variable
C   OTRAD                REAL*4     outside radius of bend
C   RATIO                REAL*4     intermediate variable
C   X                    REAL*4     intermediate variable
C   Y                    REAL*4     intermediate variable
C
C
C   SUBROUTINE BOUND(PP,UP,SIGP,FUNB)
C       Evaluates the boundary function
C
C   Commons CMPVAL      INTVAL  RELVAL
C                       Variables in Argument List
C   FUNB                 COMPLEX*8  boundary function U' + RFA * P' + RFC * SIG'
C   PP                   COMPLEX*8  P' = P0 + P1
C   SIGP                 COMPLEX*8  SIG' = SIG0 + SIG1
C   UP                   COMPLEX*8  U' = U0 + U1
C
```

```
C
C   COMPLEX FUNCTION CCOSH(S)
C       Evaluates the complex hyperbolic cosine
C
C                   Variables in Argument List
C   S                COMPLEX*8  complex frequency
C                   Local Variables
C   COSHI            REAL*4     intermediate variable
C   COSHR            REAL*4     intermediate variable
C   LAMDA            REAL*4     real part of complex frequency
C   MU               REAL*4     imaginary part of complex frequency
C
C
C   COMPLEX FUNCTION CSINH(S)
C       Evaluates the complex hyperbolic sine
C
C                   Variables in Argument List
C   S                COMPLEX*8  complex frequency
C                   Local Variables
C   LAMDA            REAL*4     real part of complex frequency
C   MU               REAL*4     imaginary part of complex frequency
C   SINHI            REAL*4     intermediate variable
C   SINHR            REAL*4     intermediate variable
C
C
C   COMPLEX FUNCTION CTANH(S)
C       Evaluates the complex hyperbolic tangent
C
C                   Variables in Argument List
C   S                COMPLEX*8  complex frequency
C                   Local Variables
C   CTAND            COMPLEX*8  hyperbolic sine
C   CTANN            COMPLEX*8  hyperbolic cosine
C
C
C   SUBROUTINE EVAL(X)
C       Evaluates parameters at a given x location
C
C   Commons CMPVAL  INTVAL  RELVAL
C                   Variables in Argument List
C   X                REAL*4     axial location
C                   Local Variables
C   I                INTEGER*2  do loop index
C   FAC              REAL*4     intermediate variable
C   UB               REAL*4     intermediate variable
C
C
C   COMPLEX FUNCTION FP1(XL)
C       Evaluates P1
C
C   Commons CMPVAL  INTVAL  RELVAL
C                   Variables in Argument List
```

```
C   XL               REAL*4     length of chamber
C                   Local Variables
C   DX               REAL*4     integration increment
C   I                INTEGER*2  do loop variable
C   VINT             COMPLEX*8  intermediate variable
C   X                REAL*4     current x location
C
C
C   COMPLEX FUNCTION FSIGP(XL)
C       Evaliates SIG'
C
C   Commons CMPVAL  INTVAL  RELVAL
C                   Variables in Argument List
C   XL               REAL*4     length of chamber
C                   Local Variables
C   DX               REAL*4     integration increment
C   FAC              REAL*4     intermediate variable
C   FCON             COMPLEX*8  intermediate variable
C   FSIG2            COMPLEX*8  intermediate variable
C   I                INTEGER*2  do loop index
C   II               INTEGER*2  do loop index
C   J                INTEGER*2  do loop index
C   UB(51)           REAL*4     intermediate variable array
C   VINT(51)         COMPLEX*8  intermediate variable array
C   VVINT(51)        COMPLEX*8  intermediate variable array
C   X                REAL*4     current x location
C
C
C   SUBROUTINE FUEL(S,GF)
C       Handles fuel piping logic
C
C   Common  PIPES
C                   Variables in Argument List
C   GF               COMPLEX*8  admittance of fuel line looking toward tank
C   S                COMPLEX*8  complex frequency
C                   Local Variables
C   A                REAL*4     speed of sound in the fluid (ft/sec)
C   ANS              CHAR*1     response to question
C   AREA(75)         REAL*4     area of pipe section (ft^2)
C   AREAB            REAL*4     intermediate variable
C   CMAN             REAL*4     manifold capacitance
C   CTANK            REAL*4     tank capacitance
C   DENS             REAL*4     density of fluid
C   DIA(75)          REAL*4     diameter of pipe section
C   DIME             REAL*4     intermediate variable
C   DPROR            REAL*4     pressure drop across orfices (lbf/ft^2)
C   FLOWL            REAL*4     intermediate variable
C   FUELIN           CHAR*24    name of file containing fuel piping data
C   GRAV             REAL*4     gravitational constant (lbm-ft/lbf-sec^2)
C   I                INTEGER*2  do loop index
C   ISTRT            INTEGER*2  flag
C   KMAN             REAL*4     bulk modulus of manifold
```

```
C   KTANK             REAL*4     bulk modulus of tank
C   L(75)             REAL*4     length of pipe section
C   LFLOW             REAL*4     flow rate through pipe
C   PCHMB             REAL*4     chamber pressure
C   PIPE1(75)         REAL*4     first parameter of fuel pipe description
C   PIPE2(75)         REAL*4     second parameter of fuel pipe description
C   PIPE3(75)         REAL*4     third parameter of fuel pipe description
C   PIPE4(75)         REAL*4     fourth parameter of fuel pipe description
C   SECTN(75)         INTEGER*2  pipe section types
C   SEGMN             INTEGER*2  number of pipe sections
C   TFLOW             REAL*4     total flow rate of engine
C   TITLF             CHAR*20    title from fuel file
C   VALUE             REAL*4     intermediate variable
C   VOL               REAL*4     volume of tank
C   VOLMF             REAL*4     volume of manifold
C
C
C   COMPLEX FUNCTION FU1(XL)
C       Evaluates U1
C
C   Commons CMPVAL  INTVAL  RELVAL
C                  Variables in Argument List
C   XL                REAL*4     length of chamber
C                  Local Variables
C   DX                REAL*4     integration increment
C   I                 INTEGER*2  do loop index
C   VINT              COMPLEX*8  intermediate variable
C   X                 REAL*4     current x location
C
C
C   SUBROUTINE GINERT(BEND,X,Y)
C       Evaluates curve fit of inertance of bends
C
C                  Variables in Argument List
C   BEND              REAL*4     angle of bend (degrees)
C   X                 REAL*4     ratio of inner to outer radius
C   Y                 REAL*4     inertance
C                  Local Variables
C   A                 REAL*4     intermediate variable
C   B(3)              REAL*4     coefficient array for inertance fit
C
C
C   SUBROUTINE ITER(ID,TOL)
C       Iterates for dependent variable
C
C   Commons CMPVAL  INTVAL  RELVAL  RESULT
C                  Variables in Argument List
C   ID                INTEGER*2  flag for dependent variable
C   TOL               REAL*4     convergence criteria
C                  Local Variables
C   FUN               REAL*4     intermediate variable
C   FUN1              REAL*4     intermediate variable
```

```
C   FUN2              REAL*4     intermediate variable
C   I                 INTEGER*2  do loop index
C   VAL               REAL*4     intermediate variable
C   VAL1              REAL*4     intermediate variable
C   VAL2              REAL*4     intermediate variable
C
C
C   SUBROUTINE LOX(S,GOX)
C       Handles lox piping logic
C
C   Common  PIPES
C                    Variables in Argument List
C   GOX               COMPLEX*8  admittance of lox line looking toward tank
C   S                 COMPLEX*8  complex frequency
C                    Local Variables
C   A                 REAL*4     speed of sound in the fluid (ft/sec)
C   ANS               CHAR*1     response to question
C   AREA(75)          REAL*4     area of pipe section (ft^2)
C   AREAB             REAL*4     intermediate variable
C   CMAN              REAL*4     manifold capacitance
C   CTANK             REAL*4     tank capacitance
C   DENS              REAL*4     density of fluid
C   DIA(75)           REAL*4     diameter of pipe section
C   DIME              REAL*4     intermediate variable
C   DPROR             REAL*4     pressure drop across orfices (lbf/ft^2)
C   FLOWL             REAL*4     intermediate variable
C   GRAV              REAL*4     gravitational constant (lbm-ft/lbf-sec^2)
C   I                 INTEGER*2  do loop index
C   ISTRT             INTEGER*2  flag
C   KMAN              REAL*4     bulk modulus of manifold
C   KTANK             REAL*4     bulk modulus of tank
C   L(75)             REAL*4     length of pipe section
C   LFLOW             REAL*4     flow rate through pipe
C   LOXIN             CHAR*24    name of file containing lox piping data
C   PCHMB             REAL*4     chamber pressure
C   PIPE1(75)         REAL*4     first parameter of fuel pipe description
C   PIPE2(75)         REAL*4     second parameter of fuel pipe description
C   PIPE3(75)         REAL*4     third parameter of fuel pipe description
C   PIPE4(75)         REAL*4     fourth parameter of fuel pipe description
C   SECTN(75)         INTEGER*2  pipe section types
C   SEGMN             INTEGER*2  number of pipe sections
C   TFLOW             REAL*4     total flow rate of engine
C   TITLO             CHAR*20    totle from lox file
C   VALUE             REAL*4     intermediate variable
C   VOL               REAL*4     volume of tank
C   VOLMF             REAL*4     volume of manifold
C
C
C   SUBROUTINE NONDIM(HOLD)
C       Nondimensionalizes variables
C
C   Commons CMPVAL  DIMVAL  INTVAL  PIPES   RELVAL  TITL
```

```
C                    Variables in Argument List
C    HOLD(20)         REAL*4      array for transferring variables
C                    Local Variables
C    CAREA            REAL*4      area of chamber
C    CSTARD           REAL*4      intermediate variable
C    FAC              REAL*4      intermediate variable
C    GC               REAL*4      gravitational constant (lbm-ft/lbf-sec^2)
C    I                INTEGER*2   do loop index
C    PEXIT            REAL*4      exit pressure
C    PI               REAL*4      mathematical constant
C    RFAR             REAL*4      intermediate variable
C    RHOBAR(50)       REAL*4      intermediate variable array
C    TAREA            REAL*4      throat area
C    UBARD(50)        REAL*4      intermediate variable array
C    VAR(13)          CHAR*8      names of nondimensional variables
C    VARD(20)         CHAR*8      names of dimensional variables
C
C
C    SUBROUTINE PLTALL(X,Y,NOT,NOF,N,M,LABLX,LABLY,FREQ)
C        Plots n vs τ for all frequencies
C
C    Commons FFACT   TITL
C                    Variables in Argument List
C    FREQ(NOF)        REAL*4      frequency array
C    LABLX            CHAR*8      label for x axis
C    LABLY            CHAR*8      label for y axis
C    M                INTEGER*2   number of frequencies
C    N                INTEGER*2   number of tau's
C    NOF              INTEGER*2   maximum number of frequencies
C    NOT              INTEGER*2   maximum number of tau's
C    X(NOT)           REAL*4      tau array
C    Y(NOT,NOF)       REAL*4      n array
C                    Local Variables
C    ASPECT           REAL*4      intermediate variable
C    FREQL            CHAR*16     label for frequency
C    I                INTEGER*2   do loop index
C    IBOARD           INTEGER*2   flag for type of graphics board used
C    ICOLR            INTEGER*2   color flag
C    IEXTEN           INTEGER*2   extension of key hit
C    IFIL             INTEGER*2   color flag
C    IKEY             INTEGER*2   code of key hit
C    ILIN             INTEGER*2   color flag
C    IOPT             INTEGER*2   intermediate variable
C    IXLAB            INTEGER*2   intermediate variable
C    IXPIX            INTEGER*2   intermediate variable
C    IYLAB            INTEGER*2   intermediate variable
C    IYPIX            INTEGER*2   intermediate variable
C    J                INTEGER*2   do loop index
C    JCOL1            INTEGER*2   starting plot column
C    JCOL2            INTEGER*2   ending plot column
C    JROW1            INTEGER*2   starting plot row
C    JROW2            INTEGER*2   ending plot row
```

```
C     LABFAC(7)         CHAR*8      labels
C     MODE              INTEGER*2   graphics mode
C     MODET             INTEGER*2   text mode
C     NCOLT             INTEGER*2   number oc text columns
C     RADHER(2)         CHAR*8      labels
C     XFAC              REAL*4      intermediate variable
C     XLABL(2)          CHAR*8      label
C     XMAJC             REAL*4      intermediate variable
C     XMAX              REAL*4      maximum x value for plot
C     XMIN              REAL*4      minimum x value for plot
C     XORG              REAL*4      plot x origin
C     YFAC              REAL*4      intermediate variable
C     YLABL(2)          CHAR*8      label
C     YMAJ              REAL*4      intermediate variable
C     YMAX              REAL*4      maximum y value for plot
C     YMIN              REAL*4      minimum y value for plot
C     YORG              REAL*4      plot y origin
C     YOVERX            REAL*4      intermediate variable
C
C
C     SUBROUTINE PLTVAR(X,Y,N,LABLX,LABLY,FREQ)
C         Plots n vs τ for a single frequency
C
C     Commons FFACT    TITL
C                      Variables in Argument List
C     FREQ              REAL*4      frequency
C     LABLX             CHAR*8      label for x axis
C     LABLY             CHAR*8      label for y axis
C     N                 INTEGER*2   number of tau's
C     X(N)              REAL*4      tau array
C     Y(N)              REAL*4      n array
C                      Local Variables
C     ASPECT            REAL*4      intermediate variable
C     FREQL             CHAR*29     label for frequency
C     I                 INTEGER*2   do loop index
C     IBOARD            INTEGER*2   flag for type of graphics board used
C     ICOLR             INTEGER*2   color flag
C     IEXTEN            INTEGER*2   extension of key hit
C     IFIL              INTEGER*2   color flag
C     IKEY              INTEGER*2   code of key hit
C     ILIN              INTEGER*2   color flag
C     IOPT              INTEGER*2   intermediate variable
C     IXLAB             INTEGER*2   intermediate variable
C     IYLAB             INTEGER*2   intermediate variable
C     JCOL1             INTEGER*2   starting plot column
C     JCOL2             INTEGER*2   ending plot column
C     JROW1             INTEGER*2   starting plot row
C     JROW2             INTEGER*2   ending plot row
C     LABFAC(7)         CHAR*8      labels
C     MODE              INTEGER*2   graphics mode
C     MODET             INTEGER*2   text mode
C     NCOLT             INTEGER*2   number oc text columns
```

```
C    RADHER(2)        CHAR*8      labels
C    XFAC             REAL*4      intermediate variable
C    XLABL(2)         CHAR*8      label
C    XMAJ             REAL*4      intermediate variable
C    XMAX             REAL*4      maximum x value for plot
C    XMIN             REAL*4      minimum x value for plot
C    XORG             REAL*4      plot x origin
C    YFAC             REAL*4      intermediate variable
C    YLABL(2)         CHAR*8      label
C    YMAJ             REAL*4      intermediate variable
C    YMAX             REAL*4      maximum y value for plot
C    YMIN             REAL*4      minimum y value for plot
C    YORG             REAL*4      plot y origin
C    YOVERX           REAL*4      intermediate variable
C
C
C    SUBROUTINE READIN
C        Reads input data
C
C    Commons CMPVAL  DIMVAL  INTVAL  RELVAL  TITL
C                    Local Variables
C    ANS              CHAR*1      response to question
C    CDIAM            REAL*4      chamber diameter (ft)
C    DCSDRD           REAL*4      d(cstar)/d(mixture ratio) (ft/sec)
C    DHLDRD           REAL*4      d(enthalpy)/d(mixture ratio) (ft/sec)^2
C    DTAUD            REAL*4      delta time lag (sec)
C    GAMMAD           REAL*4      ratio of specific heats
C    HOLD(20)         REAL*4      equivalenced to dimensioned variables
C    I                INTEGER*2   do loop index
C    IGO              INTEGER*2   path flag
C    II               INTEGER*2   do loop index
C    LAMDAD           REAL*4      real part of complex frequency
C    MBARD            REAL*4      mean combustion response function (lbm/sec)
C    MUD              REAL*4      imaginary part of complex frequency
C    NAME             CHAR*8      name of input parameter
C    ND               REAL*4      pressure interaction index
C    NRD              REAL*4      enthalpy interaction index
C    PCHMB            REAL*4      chamber pressure (lbf/ft^2)
C    POOD             REAL*4      maximum pressure at injection face
C    RBARD            REAL*4      mean mixture ratio
C    RGAS             REAL*4      gas constant (ft^2/sec^2/°R)
C    RHOLOD           REAL*4      mass of liquid per unit chamber vol (lbm/ft^3)
C    TAUD             REAL*4      sensitive time lag (sec)
C    TCHMB            REAL*4      chamber temperature (°R)
C    TDIAM            REAL*4      throat diameter (ft)
C    ULOD             REAL*4      axial component of liquid velocity (ft/sec)
C    VALUE            REAL*4      value of input parameter
C    VAR(20)          CHAR*8      names of variables for printout
C    VARL(20)         CHAR*8      names of variables (lower case)
C    VARP(20)         CHAR*8      names of variables (upper case)
C    XLCD             REAL*4      x location of chamber-nozzle interface (ft)
C
```

```
C
C   SUBROUTINE SETVAL(VAL,ID)
C       Sets value from iterated variable
C
C   Common  DIMVAL
C                   Variables in Argument List
C   ID               INTEGER*2  pointer to variable
C   VAL              REAL*4     value of variable
C
C
C   SUBROUTINE SETVAR(VAL,ID)
C       Sets iterated variable from value
C
C   Commons CMPVAL  DIMVAL  INTVAL  RELVAL  RESULT
C                   Variables in Argument List
C   ID               INTEGER*2  pointer to variable
C   VAL              REAL*4     value of variable
C                   Local Variables
C   ASTAR            REAL*4     speed of sound at injector face
C   CAREA            REAL*4     area of chamber
C   CSTARD           REAL*4     intermediate variable
C   FAC              REAL*4     intermediate variable
C   GC               REAL*4     gravitational constant (lbm-ft/lbf-sec^2)
C   I                INTEGER*2  do loop index
C   PI               REAL*4     mathematical constant
C   RHOBAR           REAL*4     intermediate variable
C   RHOB1            REAL*4     intermediate variable
C   TAREA            REAL*4     throat area
C   UBARD            REAL*4     intermediate variable
C
C
C   SUBROUTINE ZREAD(NAME,VALUE)
C       Reads input for input modification
C
C                   Variables in Argument List
C   NAME(8)          CHAR*1     name of input variable
C   VALUE            REAL*4     value of input variable
C                   Local Variables
C   BLK              CHAR*1     ' '
C   CARD(80)         CHAR*1     card image
C   CEND(3)          CHAR*1     'E','N','D'
C   COMMA            CHAR*1     ','
C   DCARD            CHAR*80    card image
C   E                CHAR*1     'E'
C   FRACT            REAL*4     fractional part of number
C   I                INTEGER*2  do loop index
C   ICOUNT           INTEGER*2  position counter
C   ID               INTEGER*2  position counter
C   II               INTEGER*2  position counter
C   J                INTEGER*2  do loop index
C   JJ               INTEGER*2  position counter
C   LE               CHAR*1     'e'
```

```
C   LEND(3)          CHAR*1     'e','n','d'
C   MINUS            CHAR*1     '-'
C   NUMBER(10)       CHAR*1     '0','1','2','3','4','5','6','7','8','9'
C   PERIOD           CHAR*1     '.'
C   PLUS             CHAR*1     '+'
C   POUND            CHAR*1     '#'
C   QUEST            CHAR*1     '?'
C   SIGN             REAL*4     sign of number or exponent
C   WHOLE            REAL*4     WHOLE PART OF NUMBER
C
      INTERFACE TO SUBROUTINE
     1             clearscreen[FAR,C,ALIAS:"__clearscreen"] (area)
      INTEGER*2 area
      END
      EXTERNAL CLEARSCREEN
      COMMON /CMPVAL/X1,Y1,Z1,W1,M1,P0,P1,U0,U1,RFH,RFK,RFP,
     *               S,GF,GOX,RFA,RFC
      COMMON /RELVAL/N,TAU,DTAU,NR,RBAR,MBAR,GAMMA,P00,DHLDR,CSTAR,
     *        DCSDR,RHOLO,ULO,LAMDA,MU,TAUT,UBAR(50),XBAR(50),XLC
      COMMON /RESULT/PP,UP,SIGP,FUNB
      COMMON /INTVAL/NVAL
      COMMON /DIMVAL/HOLDD(20),XBARD(50),PBAR(50),TBAR(50)
      COMMON /TITL/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /FFACT/FFAC
      INTEGER*2 IHR,IMIN,ISEC,I100,IYR,IMON,IDAY
      CHARACTER*2 AM,PM,AP
      CHARACTER*60 TITLE
      CHARACTER*40 TITLF
      REAL YP(200,50),FREQ(50),TAULST(200)
      REAL MBAR,N,NR,LAMDA,MU,RVAR(13)
      COMPLEX S,X1,Y1,Z1,W1,M1,P0,P1,U0,U1,GF,GOX,RFH,RFK,RFP,RFA,RFC
      COMPLEX PP,UP,SIGP,FUNB,CVAR(17)
      EQUIVALENCE (N,RVAR(1)),(X1,CVAR(1))
      CHARACTER*8 VARP(3)
      CHARACTER*1 ANS
      CHARACTER*24 ROCIN,ROCOUT,ROCVAR
      CHARACTER*8 RADHER(2)
      DATA RADHER/' rad/sec',' Hertz  '/
      DATA AM/'AM'/,PM/'PM'/
      DATA VARP/'   n    ','tau-sec ','   MU   '/
      DATA TOL/.0001/
      DATA NOT/200/,NOF/50/
      DATA II/2/,ID/1/
    1 FORMAT(A8,1PE13.5,2X,A8,E13.5,'   FUNB=',2E13.5)
    2 FORMAT(A)
    3 FORMAT(/3X,A8,5X,A8,5X,' FUNB(R)',5X,' FUNB(I)'/)
    4 FORMAT(1P6E13.5)
    5 FORMAT(1H1/'  FREQUENCY =',1PE13.5,A)
    6 FORMAT('"',A,'"')
    7 FORMAT(2X,'"',A8,'"',3X,'"',A8,'"')
   10 FORMAT(A40,2X,I2.2,':',I2.2,A2,3X,I2.2,'-',I2.2,'-',I2.2)
      CALL GETTIM(IHR,IMIN,ISEC,I100)
      CALL GETDAT(IYR,IMON,IDAY)
      IYR=IYR-1900
      IF(IHR.LT.12)  THEN
       AP=AM
      ELSE
       AP=PM
       IF(IHR.GT.12)  IHR=IHR-12
      ENDIF
      CALL CLEARSCREEN(0)
      WRITE(*,'(10X,A)')
     *'┌──────────────────────────────────────────────────────────────┐'
      WRITE(*,'(10X,A)')
     *'│                                                              │'
      WRITE(*,'(10X,A)')
     *'│        Welcome to SFREQ - an Intermediate Mode Program       │'
      WRITE(*,'(10X,A)')
     *'│                                                              │'
      WRITE(*,'(10X,A)')
     *'│                 To send a plot to the printer                │'
      WRITE(*,'(10X,A)')
     *'│                                                              │'
      WRITE(*,'(10X,A)')
     *'│             The computer MUST be in GRAPHICS mode            │'
      WRITE(*,'(10X,A)')
     *'│                                                              │'
      WRITE(*,'(10X,A)')
     *'│      Hit PrScn to send the current plot to the printer       │'
      WRITE(*,'(10X,A)')
     *'│                                                              │'
      WRITE(*,'(10X,A)')
     *'└──────────────────────────────────────────────────────────────┘'
      FFAC=1.0
      WRITE(*,*)' '
      WRITE(*,*)' If you want frequency in rad/sec, hit enter.'
      WRITE(*,'(A\)')' If you want it in Hertz, enter "H". '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'H'.OR.ANS.EQ.'h')  FFAC=6.283185
      WRITE(*,*)' '
      WRITE(*,*)' Are the files you are using'
      WRITE(*,*)'    IMODE.INP - input data'
      WRITE(*,*)'    IMODE.OUT - output data'
      WRITE(*,'(A\)')'         Enter Y or N '
      READ(*,2)ANS
      IF(ANS.NE.'N'.AND.ANS.NE.'n')  THEN
       OPEN(15,FILE='IMODE.INP')
       OPEN(16,FILE='IMODE.OUT')
      ELSE
       WRITE(*,'(A\)')'  Enter name of file containing input '
       READ(*,2)ROCIN
       OPEN(15,FILE=ROCIN)
       WRITE(*,'(A\)')'  Enter name of file for output '
       READ(*,2)ROCOUT
      OPEN(16,FILE=ROCOUT)
     ENDIF
     XLC=1.0
     WRITE(*,*)' '
     WRITE(*,*)' '
     WRITE(*,*)' '
     WRITE(*,*)' '
     WRITE(*,*)' '
     WRITE(*,*)' '
     WRITE(*,*)'                    Welcome to IMODE'
     WRITE(*,*)' '
     WRITE(*,*)'         Intermediate Mode Rocket Stability Aide'
     WRITE(*,*)' '
     WRITE(*,*)'  There are three types of input, rocket parameters,'
     WRITE(*,*)'  Oxidizer feed parameters, and fuel feed parameters,'
     WRITE(*,*)'   Each may be read from files or from the keyboard'
     WRITE(*,*)' '
     WRITE(*,*)'          File Name                 Input'
     WRITE(*,*)' '
     WRITE(*,*)'   IMODE.INP or NAME read in   Rocket Parameters  '
     WRITE(*,*)'          LOX.INP              Oxidizer Parameters'
     WRITE(*,*)'          FUEL.INP             Fuel Parameters    '
     WRITE(*,*)' '
     WRITE(*,*)' If keyboard entry, you will be prompted for values'
     GO TO 21
  20 CONTINUE
     WRITE(*,*)' '
     WRITE(*,'(A\)')' Do you want to run another case? Enter Y or N '
     READ(*,2)ANS
     IF(ANS.EQ.'N'.OR.ANS.EQ.'n')  STOP
  21 CONTINUE
     CALL READIN
  22 CONTINUE
     WRITE(*,*)' '
 231 CONTINUE
     WRITE(*,*)' Specify how frequency will be input -'
     WRITE(*,*)'   Enter R for a range of values'
     WRITE(*,*)'   Enter F for values in a file'
     WRITE(*,*)'   Enter K (end with -999) to enter values ',
    *          'from keyboard'
     READ(*,2)ANS
     IF(ANS.EQ.'R'.OR.ANS.EQ.'r')  THEN
2310 CONTINUE
      IF(FFAC.EQ.1.0)  THEN
       WRITE(*,*)' Enter first and last values of frequency ',
    *           'in rad/sec and no. of points.'
      ELSE
       WRITE(*,*)' Enter first and last values of frequency ',
    *           'in hertz and no. of points.'
      ENDIF
      READ(*,*)STARTF,STOPF,NPTF
      IF(NPTF.GT.NOF)  THEN
```

```
        WRITE(*,*)' No. of points must be <',NOF
        GO TO 2310
       ENDIF
       IF(STOPF.EQ.0.0)  STOPF=STARTF
       IF(NPTF.EQ.0)  NPTF=1
       IF(NPTF.EQ.1) THEN
        DELF=0.0
       ELSE
        DELF=(STOPF-STARTF)/(NPTF-1)
       ENDIF
       DO 232 I=1,NPTF
       FREQ(I)=STARTF+DELF*(I-1)
  232  CONTINUE
       GO TO 23
      ENDIF
      IF(ANS.EQ.'F'.OR.ANS.EQ.'f')  THEN
       WRITE(*,*)' Is the frequency on IMODE.FRQ?'
       WRITE(*,'(A\)')'        Enter Y or N '
       READ(*,2)ANS
       IF(ANS.NE.'N'.AND.ANS.NE.'n')  THEN
        OPEN(19,FILE='IMODE.FRQ')
       ELSE
        WRITE(*,'(A\)')'  Enter name of file for frequency '
        READ(*,2)ROCVAR
        OPEN(19,FILE=ROCVAR)
       ENDIF
       READ(19,*)NPTF
       IF(NPTF.GT.NOF)  THEN
        WRITE(*,*)' Too many points for program'
        GO TO 231
       ENDIF
       DO 233 I=1,NPTF
       READ(19,*)FREQ(I)
  233  CONTINUE
       GO TO 23
      ENDIF
      IF(ANS.EQ.'K'.OR.ANS.EQ.'k')  THEN
       NPTF=0
  234  CONTINUE
       READ(*,*)VAR1
       IF(VAR1.EQ.-999)  GO TO 23
       NPTF=NPTF+1
       FREQ(NPTF)=VAR1
       IF(NPTF.EQ.NOF)  GO TO 23
       GO TO 234
      ELSE
       WRITE(*,*)'  R, F, or K not entered, try again!'
       GO TO 231
      ENDIF
   23 CONTINUE
      WRITE(*,*)' Specify how tau will be input -'
      WRITE(*,*)'   Enter R for a range of values'
   WRITE(*,*)'   Enter F for values in a file'
   WRITE(*,*)'   Enter K to enter values from keyboard'
   READ(*,2)ANS
   IF(ANS.EQ.'R'.OR.ANS.EQ.'r')  GO TO 24
   IF(ANS.EQ.'F'.OR.ANS.EQ.'f')  GO TO 26
   IF(ANS.EQ.'K'.OR.ANS.EQ.'k')  GO TO 28
   WRITE(*,*)'  R, F, or K not entered, try again!'
   GO TO 23
24 CONTINUE
   WRITE(*,*)' Enter first and last values of tau ',
  *          'and no. of points.'
   READ(*,*)STARTV,STOPV,NPTS
   IF(NPTS.GT.NOT)  THEN
    WRITE(*,*)' No. of points must be <',NOT
    GO TO 24
   ENDIF
   IF(STOPV.EQ.0.0)  STOPV=STARTV
   IF(NPTS.EQ.0)  NPTS=1
   IF(NPTS.EQ.1) THEN
    DELVAL=0.0
   ELSE
    DELVAL=(STOPV-STARTV)/(NPTS-1)
   ENDIF
   DO 25 I=1,NPTS
    TAULST(I)=STARTV+(I-1)*DELVAL
25 CONTINUE
   GO TO 30
26 CONTINUE
   WRITE(*,*)' Is tau on IMODE.TAU?'
   WRITE(*,'(A\)')'        Enter Y or N '
   READ(*,2)ANS
   IF(ANS.NE.'N'.AND.ANS.NE.'n')  THEN
    OPEN(18,FILE='IMODE.TAU')
   ELSE
    WRITE(*,'(A\)')'  Enter name of file for tau '
    READ(*,2)ROCVAR
    OPEN(18,FILE=ROCVAR)
   ENDIF
   READ(18,*)NPTS
   IF(NPTS.GT.NOT)  THEN
    WRITE(*,*)' Too many points for program'
    GO TO 23
   ENDIF
   DO 27 I=1,NPTS
   READ(18,*)TAULST(I)
27 CONTINUE
   GO TO 30
28 CONTINUE
   NPTS=0
29 CONTINUE
   WRITE(*,'(A\)')
  *  ' Enter new value for independent variable (-999 to stop) '
```

```
      READ(*,*,END=99)VAR1
      IF(VAR1.EQ.-999.0)  GO TO 30
      NPTS=NPTS+1
      TAULST(I)=VAR1
      IF(NPTS.EQ.NOT)  GO TO 30
      GO TO 29
   30 CONTINUE
      DO 32 J=1,NPTF
      WRITE(16,2)TITLE
      WRITE(16,3)VARP(II),VARP(ID)
      IF(FFAC.EQ.1.0)  THEN
       WRITE(16,5)FREQ(J),RADHER(1)
       WRITE(*,5)FREQ(J),RADHER(1)
      ELSE
       WRITE(16,5)FREQ(J),RADHER(2)
       WRITE(*,5)FREQ(J),RADHER(2)
      ENDIF
      WRITE(*,3)VARP(II),VARP(ID)
       VAR1=FFAC*FREQ(J)
       CALL SETVAR(VAR1,6)
       DO 31 I=1,NPTS
        VAR1=TAULST(I)
        CALL SETVAR(VAR1,II)
        CALL ITER(ID,TOL)
        WRITE(16,4)HOLDD(II),HOLDD(ID),FUNB
        WRITE(*,4)HOLDD(II),HOLDD(ID),FUNB
        YP(I,J)=HOLDD(ID)
   31 CONTINUE
       WRITE(*,'(A\)')
     *         ' Do you wish to see n vs tau for this frequency? '
       READ(*,2)ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
        CALL PLTVAR(TAULST,YP(1,J),NPTS,VARP(II),VARP(ID),FREQ(J))
       ENDIF
   32 CONTINUE
      CALL PLTALL(TAULST,YP,NOT,NOF,NPTS,NPTF,VARP(II),VARP(ID),FREQ)
      GO TO 20
   99 CONTINUE
      STOP
      END
      SUBROUTINE ADMIT(S,GADM,A,AREA,CMAN,CTANK,DPROR,L,LFLOW,PCHMB,
     *                 SEGMN,TFLOW)
C       determines admittance looking toward tank
      COMPLEX CTANH,G(76),GADM,S,W
      REAL AREA(75),L(75),LFLOW
      INTEGER SEGMN
      COMMON /DIMVAL/AJUNK1(8),XLCD,AJUNK2(161)
      COMMON /PIPES/PFACE,TFACE,ASTAR
      DATA GRAV/32.2/
      W=S*ASTAR*2.0/XLCD
      G(1)=CTANK*W
      GADM=G(1)+1.0
      ZTOP=A*TFLOW/(GRAV*PCHMB)
      ZOR=2.0*DPROR*TFLOW/(LFLOW*PCHMB)
      DO 21 I=2,SEGMN+1
        ZLINE=ZTOP/AREA(I-1)
        TL=L(I-1)/A
        G(I)=(1.0+CTANH(W*TL)/(G(I-1)*ZLINE))/(1.0+G(I-1)*ZLINE*
     *       CTANH(W*TL))
       GADM=GADM*G(I)
       G(I)=G(I)*G(I-1)
   21 CONTINUE
       G(SEGMN+2)=1.0+CMAN*W/G(SEGMN+1)
       GADM=GADM*G(SEGMN+2)
       G(SEGMN+2)=G(SEGMN+2)*G(SEGMN+1)
       G(SEGMN+3)=1.0/(1.0+ZOR*G(SEGMN+2))
       GADM=GADM*G(SEGMN+3)
       G(SEGMN+3)=G(SEGMN+3)*G(SEGMN+2)
       GADM=G(SEGMN+3)
      RETURN
      END
      SUBROUTINE BENDS(PIPE1,PIPE2,PIPE3,PIPE4,VALUE,DIME)
C       Computes effective straight pipe for bend
      REAL LBEND,INRAD,INERT,LPRME,NEWLN
      BENDR=0.0174533*ABS(PIPE2)
      LBEND=PIPE1*BENDR
      ARBND=0.785398*PIPE3**2
      INRAD=PIPE1-0.5*PIPE3
      OTRAD=PIPE1+0.5*PIPE3
      RATIO=INRAD/OTRAD
      X=RATIO
      CALL GINERT(ABS(PIPE2),X,Y)
      INERT=(Y*(OTRAD-INRAD))/ARBND
      LPRME=LBEND/ARBND
      NEWLN=LPRME+INERT
      GAMMA=NEWLN/LPRME
      VALUE=GAMMA*(LBEND+2.0*PIPE4)
      AREAB=ARBND/SQRT(GAMMA)
      DIME=2.0*SQRT(AREAB/3.1415927)
      RETURN
      END
      SUBROUTINE BOUND(PP,UP,SIGP,FUNB)
C       Evaluates the boundary function
      COMMON /CMPVAL/X1,Y1,Z1,W1,M1,P0,P1,U0,U1,RFH,RFK,RFP,
     *               S,GF,GOX,RFA,RFC
      COMMON /RELVAL/N,TAU,DTAU,NR,RBAR,MBAR,GAMMA,P00,DHLDR,CSTAR,
     *        DCSDR,RHOLO,ULO,LAMDA,MU,TAUT,UBAR(50),XBAR(50),XLC
      COMMON /INTVAL/NVAL
      REAL MBAR,N,NR,LAMDA,MU
      COMPLEX S,X1,Y1,Z1,W1,M1,P0,P1,U0,GF,GOX,U1,RFH,RFK,RFP,RFA,RFC
      COMPLEX FP1,FU1,FSIGP,PP,UP,SIGP,FUNB,CSINH,CCOSH
C         EVALUATE PP,UP,SIGP, AND FUNB
      P1=FP1(XLC)
      U1=FU1(XLC)
      P0=P00*CCOSH(S*XLC)
      U0=-(1.0/GAMMA)*P00*CSINH(S*XLC)
      PP=P0+P1
      UP=U0+U1
      SIGP=FSIGP(XLC)
      FUNB=UP+RFA*PP+RFC*SIGP
      RETURN
      END
      COMPLEX FUNCTION CCOSH(S)
C       Evaluates the complex hyperbolic cosine
      COMPLEX S
      REAL LAMDA, MU
      LAMDA=REAL(S)
      MU=AIMAG(S)
      COSHR=COSH(LAMDA)*COS(MU)
      COSHI=SINH(LAMDA)*SIN(MU)
      CCOSH=CMPLX(COSHR,COSHI)
      RETURN
      END
      COMPLEX FUNCTION CSINH(S)
C       Evaluates the complex hyperbolic sine
      COMPLEX S
      REAL LAMDA, MU
      LAMDA=REAL(S)
      MU=AIMAG(S)
      SINHR=SINH(LAMDA)*COS(MU)
      SINHI=COSH(LAMDA)*SIN(MU)
      CSINH=CMPLX(SINHR,SINHI)
      RETURN
      END
      COMPLEX FUNCTION CTANH(S)
C       Evaluates the complex hyperbolic tangent
      COMPLEX S,CTANN,CTAND,CSINH,CCOSH
      CTANN=CSINH(S)
      CTAND=CCOSH(S)
      CTANH=(0.0,0.0)
      IF(CTAND.NE.0.0)  CTANH=CTANN/CTAND
      RETURN
      END
      SUBROUTINE EVAL(X)
C       Evaluates parameters at a given x location
      COMMON /CMPVAL/X1,Y1,Z1,W1,M1,P0,P1,U0,U1,RFH,RFK,RFP,
     *               S,GF,GOX,RFA,RFC
      COMMON /RELVAL/N,TAU,DTAU,NR,RBAR,MBAR,GAMMA,P00,DHLDR,CSTAR,
     *        DCSDR,RHOLO,ULO,LAMDA,MU,TAUT,UBAR(50),XBAR(50),XLC
      COMMON /INTVAL/NVAL
      REAL MBAR,N,NR,LAMDA,MU
      COMPLEX S,X1,Y1,Z1,W1,M1,P0,P1,U0,U1,GF,GOX,RFH,RFK,RFP,RFA,RFC
      COMPLEX CSINH,CCOSH
C         EVALUATE EVERYTHING EXCEPT PP,UP,SIGP
      IF(NVAL.EQ.1)  THEN
       UB=UBAR(1)
       GO TO 23
      ENDIF
      DO 21 I=2,NVAL
       IF(X.LE.XBAR(I))  GO TO 22
   21 CONTINUE
      UB=UBAR(NVAL)
      GO TO 23
   22 CONTINUE
      FAC=(X-XBAR(I-1))/(XBAR(I)-XBAR(I-1))
      UB=UBAR(I-1)+FAC*(UBAR(I)-UBAR(I-1))
   23 CONTINUE
      RFH=(1.0+RBAR)*((RBAR/CSTAR)*DCSDR-NR*S*TAU)*(GOX
     *    -RBAR*GF)/RBAR
      RFK=(1.0+S*TAUT)*(GOX+GF)
      RFP=N*(1.0-CEXP(S*TAU))
      P0=P00*CCOSH(S*X)
      U0=-(1.0/GAMMA)*P00*CSINH(S*X)
      X1=(GAMMA-1.0)*UB*U0+(1.0+RBAR)*DHLDR*(MBAR/S)
     *   *CEXP(-S*TAUT)*(GOX-RBAR*GF)*P00
      Y1=-UB*P0
      Z1=(1.0/GAMMA)*UB*P0+RHOLO*ULO
      W1=2.0*UB*U0
      M1=MBAR*(CEXP(-S*TAUT)*(RFK+RFH)*P00-RFP*P0)
      RETURN
      END
      COMPLEX FUNCTION FP1(XL)
C       Evaluates P1
      COMMON /CMPVAL/X1,Y1,Z1,W1,M1,P0,P1,U0,U1,RFH,RFK,RFP,
     *               S,GF,GOX,RFA,RFC
      COMMON /RELVAL/N,TAU,DTAU,NR,RBAR,MBAR,GAMMA,P00,DHLDR,CSTAR,
     *        DCSDR,RHOLO,ULO,LAMDA,MU,TAUT,UBAR(50),XBAR(50),XLC
      COMMON /INTVAL/NVAL
      REAL MBAR,N,NR,LAMDA,MU
      COMPLEX S,X1,Y1,Z1,W1,M1,P0,P1,U0,U1,GF,GOX,RFH,RFK,RFP,RFA,RFC
      COMPLEX CSINH,CCOSH
      COMPLEX VINT
C         EVALUATE P1
      DX=XL/50.0
      FP1=CMPLX(0.0,0.0)
      DO 23 I=1,51
       X=(I-1)*DX
       CALL EVAL(X)
       VINT=(S*(W1-X1)+M1)*CSINH(S*(XL-X))
     *      +S*(Y1+Z1)*CCOSH(S*(XL-X))
       IF(I.EQ.1.OR.I.EQ.51)  THEN
        FP1=FP1+0.5*VINT*DX
       ELSE
        FP1=FP1+VINT*DX
       ENDIF
   23 CONTINUE
      FP1=-GAMMA*(W1+FP1)
      RETURN
      END
      COMPLEX FUNCTION FSIGP(XL)
C       Evaliates SIG'
      COMMON /CMPVAL/X1,Y1,Z1,W1,M1,P0,P1,U0,U1,RFH,RFK,RFP,
     *             S,GF,GOX,RFA,RFC
      COMMON /RELVAL/N,TAU,DTAU,NR,RBAR,MBAR,GAMMA,P00,DHLDR,CSTAR,
     *       DCSDR,RHOLO,ULO,LAMDA,MU,TAUT,UBAR(50),XBAR(50),XLC
      COMMON /INTVAL/NVAL
      REAL MBAR,N,NR,LAMDA,MU
      COMPLEX S,X1,Y1,Z1,W1,M1,P0,P1,U0,U1,GF,GOX,RFH,RFK,RFP,RFA,RFC
      REAL UB(51)
      COMPLEX VINT(51),VVINT(51),FSIG2,FCON
C         EVALUATE FSIGP  (INTEGRATION NOT CHANGED YET)
      DX=XL/50.0
      DO 23 I=1,51
       X=(I-1)*DX
       IF(NVAL.EQ.1)  THEN
        UB(I)=UBAR(1)
        GO TO 23
       ENDIF
       DO 21 II=2,NVAL
        IF(X.LE.XBAR(II))  GO TO 22
   21 CONTINUE
       II=NVAL
   22 CONTINUE
       FAC=(X-XBAR(II-1))/(XBAR(II)-XBAR(II-1))
       UB(I)=UBAR(II-1)+FAC*(UBAR(II)-UBAR(II-1))
   23 CONTINUE
      DO 24 I=1,51
       X=(I-1)*DX
       CALL EVAL(X)
       VINT(I)=((GAMMA-1.0)/GAMMA)*P0
       VVINT(I)=1.0/UB(I)
   24 CONTINUE
      FCON=(1.0+RBAR)*DHLDR*(GOX-RBAR*GF)*P00
     *     *CEXP(-S*TAUT)
      DO 26 I=1,51
       FSIG2=CMPLX(0.0,0.0)
       DO 25 J=I,51
        IF(J.EQ.I.OR.J.EQ.51)  THEN
         FSIG2=FSIG2+0.5*VVINT(J)*DX
        ELSE
         FSIG2=FSIG2+VVINT(J)*DX
        ENDIF
   25 CONTINUE
       FSIG2=CEXP(-S*FSIG2)
       VINT(I)=(VINT(I)+FCON)*MBAR*FSIG2
   26 CONTINUE
      FSIGP=CMPLX(0.0,0.0)
      DO 27 I=1,51
       IF(I.EQ.1.OR.I.EQ.51)  THEN
        FSIGP=FSIGP+0.5*VINT(I)*DX
       ELSE
        FSIGP=FSIGP+VINT(I)*DX
       ENDIF
   27 CONTINUE
      FSIGP=-FSIGP/UB(51)
      RETURN
      END
      SUBROUTINE FUEL(S,GF)
C       Handles fuel piping logic
      COMMON /PIPES/PFACE,TFACE,ASTAR
      COMPLEX GF,S
      REAL AREA(75),DIA(75),L(75),KMAN,KTANK,LFLOW
      REAL PIPE1(75),PIPE2(75),PIPE3(75),PIPE4(75)
      INTEGER SEGMN,SECTN(75)
      CHARACTER*24 FUELIN
      CHARACTER*20 TITLF
      CHARACTER*1 ANS
      DATA ISTRT/0/,GRAV/32.2/
    1 FORMAT(E15.6)
    2 FORMAT(I5,4E15.6)
      IF(ISTRT.EQ.0)  THEN
       ISTRT=1
       WRITE(*,*)' Is the file with fuel line data FUEL.INP?'
       WRITE(*,'(A\)')'        Enter Y or N '
       READ(*,'(A)')ANS
       IF(ANS.NE.'N'.AND.ANS.NE.'n')  THEN
        OPEN(UNIT=11,FILE='FUEL.INP')
       ELSE
        WRITE(*,'(A\)')'  Enter name of file with fuel line data '
        READ(*,'(A)')FUELIN
        OPEN(11,FILE=FUELIN)
       ENDIF
C        FUEL TITLE
       READ(11,'(A)')TITLF
C        TANK CONDITIONS
       READ(11,1)VOL
       READ(11,1)LFLOW
       READ(11,1)KTANK
C        MANIFOLD CONDITIONS
       READ(11,1)DENS
       READ(11,1)TFLOW
       READ(11,1)VOLMF
       READ(11,1)KMAN
       READ(11,1)PCHMB
C        ORFICE CONDITION
       READ(11,1)DPROR
       A=SQRT(GRAV*KTANK/DENS)
       CTANK=(DENS*VOL*PCHMB)/(KTANK*TFLOW)
       CMAN=(DENS*VOLMF*PCHMB)/(KMAN*TFLOW)
C        PIPING
       READ(11,2)SEGMN
       DO 21 I=1,SEGMN
```

```
        READ(11,2)SECTN(I),PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I)
        IF(SECTN(I).EQ.0)  THEN
         CALL BENDS(PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),VALUE,DIME)
       ELSE
        VALUE=PIPE1(I)
        DIME=PIPE2(I)
       ENDIF
       AREAB=0.785398*DIME**2
       L(I)=VALUE
       AREA(I)=AREAB
       DIA(I)=DIME
   21 CONTINUE
      ENDIF
      FLOWL=LFLOW*TFACE/TFLOW
      CTANK=(DENS*VOL*PFACE)/(KTANK*TFACE)
      CMAN=(DENS*VOLMF*PFACE)/(KMAN*TFACE)
      CALL ADMIT(S,GF,A,AREA,CMAN,CTANK,DPROR,L,FLOWL,PFACE,
     *           SEGMN,TFACE)
      RETURN
      END
      COMPLEX FUNCTION FU1(XL)
C       Evaluates U1
      COMMON /CMPVAL/X1,Y1,Z1,W1,M1,P0,P1,U0,U1,RFH,RFK,RFP,
     *               S,GF,GOX,RFA,RFC
      COMMON /RELVAL/N,TAU,DTAU,NR,RBAR,MBAR,GAMMA,P00,DHLDR,CSTAR,
     *        DCSDR,RHOLO,ULO,LAMDA,MU,TAUT,UBAR(50),XBAR(50),XLC
      COMMON /INTVAL/NVAL
      REAL MBAR,N,NR,LAMDA,MU
      COMPLEX S,X1,Y1,Z1,W1,M1,P0,P1,U0,U1,GF,GOX,RFH,RFK,RFP,RFA,RFC
      COMPLEX CSINH,CCOSH
      COMPLEX VINT
C         EVALUATE U1
      DX=XL/50.0
      FU1=CMPLX(0.0,0.0)
      DO 23 I=1,51
       X=(I-1)*DX
       CALL EVAL(X)
       VINT=(S*(W1-X1)+M1)*CCOSH(S*(XL-X))
     *      +S*(Y1+Z1)*CSINH(S*(XL-X))
       IF(I.EQ.1.OR.I.EQ.51)  THEN
        FU1=FU1+0.5*VINT*DX
       ELSE
        FU1=FU1+VINT*DX
       ENDIF
   23 CONTINUE
      FU1=Y1+FU1
      RETURN
      END
      SUBROUTINE GINERT(BEND,X,Y)
C       Evaluates curve fit of inertance of bends
      DIMENSION B(3)
      DATA B/0.0,0.7877014E-02,-0.2814679E-04/
      A=B(1)+(B(2)+B(3)*BEND)*BEND
      Y=A*(X-1.0)**2
      RETURN
      END
      SUBROUTINE ITER(ID,TOL)
C       Iterates for dependent variable
      COMMON /CMPVAL/X1,Y1,Z1,W1,M1,P0,P1,U0,U1,RFH,RFK,RFP,
     *               S,GF,GOX,RFA,RFC
      COMMON /RELVAL/N,TAU,DTAU,NR,RBAR,MBAR,GAMMA,P00,DHLDR,CSTAR,
     *        DCSDR,RHOLO,ULO,LAMDA,MU,TAUT,UBAR(50),XBAR(50),XLC
      COMMON /INTVAL/NVAL
      COMMON /RESULT/PP,UP,SIGP,FUNB
      REAL MBAR,N,NR,LAMDA,MU,RVAR(13)
      COMPLEX S,X1,Y1,Z1,W1,M1,P0,P1,U0,U1,GF,GOX,RFH,RFK,RFP,RFA,RFC
      COMPLEX PP,UP,SIGP,FUNB,CVAR(17)
      EQUIVALENCE (N,RVAR(1)),(X1,CVAR(1))
      CALL SETVAL(VAL1,ID)
      CALL BOUND(PP,UP,SIGP,FUNB)
      FUN1=REAL(FUNB)
      IF(ABS(FUN1).LE.TOL)  GO TO 22
      VAL2=1.01*VAL1
      IF(VAL1.EQ.0)  VAL2=0.01
      CALL SETVAR(VAL2,ID)
      CALL BOUND(PP,UP,SIGP,FUNB)
      FUN2=REAL(FUNB)
      IF(ABS(FUN2).LE.TOL)  GO TO 22
      IF(FUN1.EQ.FUN2)  THEN
       VAL=VAL1+VAL2
      ELSE
       VAL=VAL1-FUN1*(VAL2-VAL1)/(FUN2-FUN1)
      ENDIF
       IF(ABS(FUN2).LT.ABS(FUN1))  THEN
        FUN=FUN2
        FUN2=FUN1
        FUN1=FUN
        VAL=VAL2
        VAL2=VAL1
        VAL1=VAL
       ENDIF
      DO 21 I=1,20
       CALL SETVAR(VAL,ID)
       CALL BOUND(PP,UP,SIGP,FUNB)
       FUN=REAL(FUNB)
       IF(ABS(FUN).LE.TOL)  GO TO 22
       IF(ABS(FUN).LT.ABS(FUN1))  THEN
        FUN2=FUN1
        FUN1=FUN
        VAL2=VAL1
        VAL1=VAL
       ELSE
        FUN2=FUN
        VAL2=VAL
       ENDIF
       IF(FUN1.EQ.FUN2)  THEN
        IF(VAL1.EQ.VAL2)  THEN
         VAL=VAL1+VAL2
        ELSE
         VAL=0.5*(VAL1+VAL2)
        ENDIF
       ELSE
        VAL=VAL1-FUN1*(VAL2-VAL1)/(FUN2-FUN1)
       ENDIF
   21 CONTINUE
      WRITE(*,*)' FAILED TO CONVERGE after 20 iterations'
   22 CONTINUE
      RETURN
      END
      SUBROUTINE LOX(S,GOX)
C       Handles lox piping logic
      COMMON /PIPES/PFACE,TFACE,ASTAR
      COMPLEX GOX,S
      REAL AREA(75),DIA(75),L(75),KMAN,KTANK,LFLOW
      REAL PIPE1(75),PIPE2(75),PIPE3(75),PIPE4(75)
      INTEGER SEGMN,SECTN(75)
      CHARACTER*24 LOXIN
      CHARACTER*20 TITLO
      CHARACTER*1 ANS
      DATA ISTRT/0/,GRAV/32.2/
    1 FORMAT(E15.6)
    2 FORMAT(I5,4E15.6)
      IF(ISTRT.EQ.0)  THEN
       ISTRT=1
       WRITE(*,*)' Is the file with lox line data LOX.INP?'
       WRITE(*,'(A\)')'        Enter Y or N '
       READ(*,'(A)')ANS
       IF(ANS.NE.'N'.AND.ANS.NE.'n')  THEN
        OPEN(UNIT=10,FILE='LOX.INP')
       ELSE
        WRITE(*,'(A\)')'  Enter name of file with lox line data '
        READ(*,'(A)')LOXIN
        OPEN(10,FILE=LOXIN)
       ENDIF
C        LOX TITLE
       READ(10,'(A)')TITLO
C        TANK CONDITIONS
       READ(10,1)VOL
       READ(10,1)LFLOW
       READ(10,1)KTANK
C        MANIFOLD CONDITIONS
       READ(10,1)DENS
       READ(10,1)TFLOW
       READ(10,1)VOLMF
       READ(10,1)KMAN
       READ(10,1)PCHMB
```

```
C        ORFICE CONDITION
       READ(10,1)DPROR
       A=SQRT(GRAV*KTANK/DENS)
       CTANK=(DENS*VOL*PCHMB)/(KTANK*TFLOW)
       CMAN=(DENS*VOLMF*PCHMB)/(KMAN*TFLOW)
C        PIPING
       READ(10,2)SEGMN
       DO 21 I=1,SEGMN
        READ(10,2)SECTN(I),PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I)
        IF(SECTN(I).EQ.0)  THEN
         CALL BENDS(PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I),VALUE,DIME)
        ELSE
        VALUE=PIPE1(I)
        DIME=PIPE2(I)
       ENDIF
       AREAB=0.785398*DIME**2
       L(I)=VALUE
       AREA(I)=AREAB
       DIA(I)=DIME
   21 CONTINUE
      ENDIF
      FLOWL=LFLOW*TFACE/TFLOW
      CTANK=(DENS*VOL*PFACE)/(KTANK*TFACE)
      CMAN=(DENS*VOLMF*PFACE)/(KMAN*TFACE)
      CALL ADMIT(S,GOX,A,AREA,CMAN,CTANK,DPROR,L,FLOWL,PFACE,
     *           SEGMN,TFACE)
      RETURN
      END
      SUBROUTINE NONDIM(HOLD)
C       Nondimensionalizes variables
      COMMON /CMPVAL/X1,Y1,Z1,W1,M1,P0,P1,U0,U1,RFH,RFK,RFP,
     *               S,GF,GOX,RFA,RFC
      COMMON /RELVAL/N,TAU,DTAU,NR,RBAR,MBAR,GAMMA,P00,DHLDR,CSTAR,
     *        DCSDR,RHOLO,ULO,LAMDA,MU,TAUT,UBAR(50),XBAR(50),XLC
      COMMON /INTVAL/NVAL
      COMMON /DIMVAL/HOLDD(20),XBARD(50),PBAR(50),TBAR(50)
      COMMON /PIPES/PFACE,TFACE,ASTAR
      COMMON /TITL/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      CHARACTER*60 TITLE
      CHARACTER*40 TITLF
      REAL MBAR,N,NR,LAMDA,MU,RVAR(15)
      REAL MBARD,ND,NRD,LAMDAD,MUD
      REAL HOLD(20),UBARD(50),RHOBAR(50)
      COMPLEX S,X1,Y1,Z1,W1,M1,P0,P1,U0,U1,GF,GOX,RFH,RFK,RFP,RFA,RFC
      COMPLEX CVAR(17)
      CHARACTER*8 VAR(13),VARD(20)
      EQUIVALENCE (N,RVAR(1)),(X1,CVAR(1))
      EQUIVALENCE
     *    (ND,HOLDD(1)),(TAUD,HOLDD(2)),(DTAUD,HOLDD(3)),
     *    (NRD,HOLDD(4)),(LAMDAD,HOLDD(5)),(MUD,HOLDD(6)),
     *    (CDIAM,HOLDD(7)),(TDIAM,HOLDD(8)),(XLCD,HOLDD(9)),
     *    (GAMMAD,HOLDD(10)),(RGAS,HOLDD(11)),(P00D,HOLDD(12)),
     *    (MBARD,HOLDD(13)),(RBARD,HOLDD(14)),(DCSDRD,HOLDD(15)),
     *    (DHLDRD,HOLDD(16)),(RHOLOD,HOLDD(17)),(ULOD,HOLDD(18)),
     *    (PCHMB,HOLDD(19)),(TCHMB,HOLDD(20))
      DATA VAR/'      N=','    TAU=','   DTAU=','     NR=','   RBAR=',
     *         '   MBAR=','  GAMMA=','    P00=','  DHLDR=','  CSTAR=',
     *         '  DCSDR=','  RHOLO=','    ULO='/
      DATA VARD/'     N =','    TAU =','   DTAU =','     NR =',' LAMDA =',
     *          '    MU =',' CDIAM =',' TDIAM =','    XLC =',' GAMMA =',
     *          '  RGAS =','    P00 =','   MBAR =','   RBAR =',' DCSDR =',
     *          ' DHLDR =',' RHOLO =','    ULO =',' PCHMB =',' TCHMB ='/
      DATA PI/3.141593/,GC/32.174/
    1 FORMAT(A)
    2 FORMAT(A8,1PE13.5,2X,A8,E13.5,2X,A8,E13.5)
    3 FORMAT(' ')
C
C          N     - HOLD(1)
C          TAU   - HOLD(2)
C          DTAU  - HOLD(3)
C          NR    - HOLD(4)
C          LAMDA - HOLD(5)
C          MU    - HOLD(6)
C          CDIAM - HOLD(7)
C          TDIAM - HOLD(8)
C          XLC   - HOLD(9)
C          GAMMA - HOLD(10)
C          RGAS  - HOLD(11)
C          P00   - HOLD(12)
C          MBAR  - HOLD(13)
C          RBAR  - HOLD(14)
C          DCSDR - HOLD(15)
C          DHLDR - HOLD(16)
C          RHOLO - HOLD(17)
C          ULO   - HOLD(18)
C          PCHMB - HOLD(19)
C          TCHMB - HOLD(20)
C          PBAR  - PBAR
C          TBAR  - TBAR
C          XBAR  - XBARD
C
C          PCHMB = PBAR(1)
C          TFLOW = LFLOW(LOX) + LFLOW(FUEL)
C          LFLOW = LINE FLOW OF LOX OR FUEL
C
      DO 21 I=1,20
       HOLDD(I)=HOLD(I)
   21 CONTINUE
      IF(PCHMB.NE.PBAR(1))  THEN
       FAC=PCHMB/PBAR(1)
       DO 22 I=1,NVAL
        PBAR(I)=FAC*PBAR(I)
```

```
22 CONTINUE
   ENDIF
   IF(TCHMB.NE.TBAR(1))  THEN
    FAC=TCHMB/TBAR(1)
    DO 23 I=1,NVAL
     TBAR(I)=FAC*TBAR(I)
23 CONTINUE
   ENDIF
   CAREA=0.25*PI*CDIAM**2
   WRITE(16,3)
   WRITE(16,*)'  CAREA=',CAREA
   TAREA=0.25*PI*TDIAM**2
   WRITE(16,*)'  TAREA=',TAREA
   PFACE=PBAR(1)
   PEXIT=PBAR(NVAL)
   TFACE=MBARD
   ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
   WRITE(16,*)'  ASTAR=',ASTAR
   CSTARD=PEXIT*TAREA*GC/MBARD
   WRITE(16,*)' CSTARD=',CSTARD
   DO 24 I=1,NVAL
    RHOBAR(I)=PBAR(I)*GC/(RGAS*TBAR(I))
   WRITE(16,*)' RHOBAR=',RHOBAR(I)
    UBARD(I)=MBARD/(RHOBAR(I)*CAREA)
   WRITE(16,*)'  UBARD=',UBARD(I)
24 CONTINUE
   N=ND
   TAU=TAUD*ASTAR/XLCD
   DTAU=DTAUD*ASTAR/XLCD
   TAUT=TAU+DTAU
   NR=NRD
   RBAR=RBARD
   MBAR=MBARD/(RHOBAR(1)*ASTAR*CAREA/XLCD)
   GAMMA=GAMMAD
   POO=POOD/PBAR(1)
   DHLDR=DHLDRD
   CSTAR=CSTARD/ASTAR
   DCSDR=DCSDRD/ASTAR
   RHOLO=RHOLOD/RHOBAR(1)
   ULO=ULOD/ASTAR
   LAMDA=LAMDAD*XLCD/ASTAR
   MU=MUD*XLCD*PI/ASTAR
   XLC=1.0
   DO 25 I=1,NVAL
    XBAR(I)=XBARD(I)/XLCD
    UBAR(I)=UBARD(I)/ASTAR
25 CONTINUE
   S=CMPLX(LAMDA,MU)
   CALL FUEL(S,GF)
   CALL LOX(S,GOX)
   RFAR=(GAMMA-1.0)*UBAR(1)/(2.0*GAMMA)
   RFA=CMPLX(RFAR,0.0)
```

```
      RFC=CMPLX(0.0,0.0)
      WRITE(*,*)' '
      WRITE(*,1)TITLE
      WRITE(*,*)'                     DIMENSIONAL VARIABLES'
      WRITE(*,'(''   NVAL='',I5)')NVAL
      WRITE(*,'(''   XBAR='',1P4E13.5/(8X,4E13.5))')(XBARD(I),I=1,NVAL)
      WRITE(*,'(''   UBAR='',1P4E13.5/(8X,4E13.5))')(UBARD(I),I=1,NVAL)
      WRITE(*,2)(VARD(I),HOLDD(I),I=1,20)
      WRITE(16,3)
      WRITE(16,1)TITLE
      WRITE(16,3)
      WRITE(16,*)'                     DIMENSIONAL VARIABLES'
      WRITE(16,'(''   NVAL='',I5)')NVAL
      WRITE(16,'(''   XBAR='',1P4E13.5/(8X,4E13.5))')(XBARD(I),I=1,NVAL)
      WRITE(16,'(''   UBAR='',1P4E13.5/(8X,4E13.5))')(UBARD(I),I=1,NVAL)
      WRITE(16,2)(VARD(I),HOLDD(I),I=1,20)
      WRITE(*,*)'                  NON-DIMENSIONAL VARIABLES'
      WRITE(*,'(''   NVAL='',I5)')NVAL
      WRITE(*,'(''   XBAR='',1P4E13.5/(8X,4E13.5))')(XBAR(I),I=1,NVAL)
      WRITE(*,'(''   UBAR='',1P4E13.5/(8X,4E13.5))')(UBAR(I),I=1,NVAL)
      WRITE(*,'(''      S='',1P2E13.5)')LAMDA,MU
      WRITE(*,2)(VAR(I),RVAR(I),I=1,13)
      WRITE(*,'(''     GF='',1P2E13.5,5X,''    GOX='',2E13.5)')GF,GOX
      WRITE(*,'(''    RFA='',1P2E13.5,5X,''    RFC='',2E13.5)')RFA,RFC
      WRITE(16,3)
      WRITE(16,*)'                  NON-DIMENSIONAL VARIABLES'
      WRITE(16,'(''   NVAL='',I5)')NVAL
      WRITE(16,'(''   XBAR='',1P4E13.5/(8X,4E13.5))')(XBAR(I),I=1,NVAL)
      WRITE(16,'(''   UBAR='',1P4E13.5/(8X,4E13.5))')(UBAR(I),I=1,NVAL)
      WRITE(16,'(''      S='',1P2E13.5)')LAMDA,MU
      WRITE(16,2)(VAR(I),RVAR(I),I=1,13)
      WRITE(16,'(''     GF='',1P2E13.5,5X,''    GOX='',2E13.5)')GF,GOX
      WRITE(16,'(''    RFA='',1P2E13.5,5X,''    RFC='',2E13.5)')RFA,RFC
      WRITE(*,'(A\)')' Hit ENTER to continue '
      READ(*,*)
      RETURN
      END
      SUBROUTINE PLTALL(X,Y,NOT,NOF,N,M,LABLX,LABLY,FREQ)
C       Plots n vs τ for all frequencies
      DIMENSION X(NOT),Y(NOT,NOF),FREQ(NOF)
      CHARACTER*8 LABLX,LABLY,LABFAC(7)
      CHARACTER*8 XLABL(2),YLABL(2)
      CHARACTER*16 FREQL
      COMMON /TITL/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      CHARACTER*60 TITLE
      CHARACTER*40 TITLF
      COMMON /FFACT/FFAC
      CHARACTER*8 RADHER(2)
      DATA RADHER/' rad/sec',' Hertz  '/
      DATA LABFAC/'        ',' x 10   ',' x 100  ',' x 1000 ',
  *                       ' x-10   ',' x-100  ',' x-1000 '/
   DATA ASPECT/1.35/
 1 FORMAT(F8.1,A)
   CALL QRMODE(MODET,NCOLT)
   CALL QVIDBD(IBOARD)
   IF(IBOARD.LT.1.OR.IBOARD.GT.3)  THEN
    WRITE(*,*)' Graphics board not installed!'
    RETURN
   ENDIF
   IF(IBOARD.EQ.1)  MODE=6
   IF(IBOARD.EQ.2)  MODE=16
   IF(IBOARD.EQ.3)  MODE=18
   YMIN=Y(1,1)
   YMAX=Y(N,1)
   XMIN=X(1)
   XMAX=X(N)
   DO 21 I=1,N
    IF(XMIN.GT.X(I))  XMIN=X(I)
    IF(XMAX.LT.X(I))  XMAX=X(I)
   DO 21 J=1,M
    IF(YMIN.GT.Y(I,J))  YMIN=Y(I,J)
    IF(YMAX.LT.Y(I,J))  YMAX=Y(I,J)
21 CONTINUE
   IF(YMIN.GT.0.0)  YMIN=0.0
   IXLAB=1
   IF(XMAX.LT.0.1)  IXLAB=2
   IF(XMAX.LT.0.01)  IXLAB=3
   IF(XMAX.LT.0.001)  IXLAB=4
   IF(XMAX.GT.10.0)  IXLAB=5
   IF(XMAX.GT.100.0)  IXLAB=6
   IF(XMAX.GT.1000.0)  IXLAB=7
   IYLAB=1
   IF(YMAX.LT.0.1)  IYLAB=2
   IF(YMAX.LT.0.01)  IYLAB=3
   IF(YMAX.LT.0.001)  IYLAB=4
   IF(YMAX.GT.10.0)  IYLAB=5
   IF(YMAX.GT.100.0)  IYLAB=6
   IF(YMAX.GT.1000.0)  IYLAB=7
   IF(IXLAB.NE.1)  THEN
    IF(IXLAB.EQ.2)  XFAC=10.0
    IF(IXLAB.EQ.3)  XFAC=100.0
    IF(IXLAB.EQ.4)  XFAC=1000.0
    IF(IXLAB.EQ.5)  XFAC=0.01
    IF(IXLAB.EQ.6)  XFAC=0.001
    IF(IXLAB.EQ.7)  XFAC=0.0001
    XMIN=XMIN*XFAC
    XMAX=XMAX*XFAC
    DO 22 I=1,N
     X(I)=X(I)*XFAC
22 CONTINUE
   ENDIF
   IF(IYLAB.NE.1)  THEN
```

```
       IF(IYLAB.EQ.2)  YFAC=10.0
       IF(IYLAB.EQ.3)  YFAC=100.0
       IF(IYLAB.EQ.4)  YFAC=1000.0
       IF(IYLAB.EQ.5)  YFAC=0.01
       IF(IYLAB.EQ.6)  YFAC=0.001
       IF(IYLAB.EQ.7)  YFAC=0.0001
       YMIN=YMIN*YFAC
       YMAX=YMAX*YFAC
       DO 23 J=1,M
       DO 23 I=1,N
        Y(I,J)=Y(I,J)*YFAC
   23 CONTINUE
      ENDIF
      XLABL(1)=LABLX
      XLABL(2)=LABFAC(IXLAB)
      YLABL(1)=LABLY
      YLABL(2)=LABFAC(IYLAB)
      XMAJ=0.2*(XMAX-XMIN)
      YMAJ=0.2*(YMAX-YMIN)
      ICOLR=4
      IFIL=3
      ILIN=1
      CALL QSMODE(MODE)
      IF(IBOARD.NE.1)  THEN
       CALL QPREG(0,ICOLR)
      ENDIF
      JCOL1=150
      JCOL2=500
      JROW1=40
      IF(MODE.EQ.6)  JROW1=60
      JROW2=149
      IF(MODE.EQ.16)  JROW2=299
      IF(MODE.EQ.18)  JROW2=419
      XORG=XMIN
      YORG=YMIN
      YOVERX=1.0
      IOPT=0
      IF(MODE.NE.18)  THEN
       CALL QPTXT(60,TITLE,7,5,23)
      ELSE
       CALL QPTXT(60,TITLE,7,5,29)
      ENDIF
      CALL QPTXT(8,YLABL(1),7,2,15)
      CALL QPTXT(8,YLABL(2),7,2,14)
      CALL QPLOT(JCOL1,JCOL2,JROW1,JROW2,XMIN,XMAX,YMIN,YMAX,
     *           XORG,YORG,IOPT,YOVERX,ASPECT)
      CALL QSETUP(0,ILIN,-2,IFIL)
      CALL QXAXIS(XMIN,XMAX,0.0,0,0,0)
      CALL QPTXTA(16,XLABL,7)
      CALL QXAXIS(XMIN,XMAX,XMAJ,0,-1,2)
      CALL QYAXIS(YMIN,YMAX,YMAJ,0,-1,2)
      DO 24 J=1,M
```

```
       IF(FFAC.EQ.1.0)  THEN
        WRITE(FREQL,1)FREQ(J),RADHER(1)
       ELSE
        WRITE(FREQL,1)FREQ(J),RADHER(2)
       ENDIF
       IF(MOD(J,2).EQ.0)  THEN
        CALL QSETUP(0,ILIN+1,-2,IFIL)
       ELSE
        CALL QSETUP(0,ILIN,-2,IFIL)
       ENDIF
       CALL QTABL(1,N,X,Y(1,J))
       CALL QRTOI(X(N),Y(N,J),IXPIX,IYPIX)
       IYPIX=IYPIX-5
       IXPIX=IXPIX+2
       CALL QGTXT(16,FREQL,7,IXPIX,IYPIX,0)
   24 CONTINUE
   25 CONTINUE
      CALL QONKEY(IKEY)
      IF(IKEY.EQ.0)  GO TO 25
      CALL QINKEY(IEXTEN,IKEY)
      CALL QSMODE(MODET)
      IF(IXLAB.NE.1)  THEN
       DO 31 I=1,N
        X(I)=X(I)/XFAC
   31 CONTINUE
      ENDIF
      IF(IYLAB.NE.1)  THEN
       DO 32 J=1,M
       DO 32 I=1,N
        Y(I,J)=Y(I,J)/YFAC
   32 CONTINUE
      ENDIF
      RETURN
      END
      SUBROUTINE PLTVAR(X,Y,N,LABLX,LABLY,FREQ)
C       Plots n vs τ for a single frequency
      DIMENSION X(N),Y(N)
      CHARACTER*8 LABLX,LABLY,LABFAC(7)
      CHARACTER*8 XLABL(2),YLABL(2)
      COMMON /TITL/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /FFACT/FFAC
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      CHARACTER*60 TITLE
      CHARACTER*40 TITLF
      CHARACTER*29 FREQL
      CHARACTER*8 RADHER(2)
      DATA RADHER/' rad/sec',' Hertz  '/
      DATA LABFAC/'        ',' x 10   ',' x 100  ',' x 1000 ',
     *                       ' x-10   ',' x-100  ',' x-1000 '/
      DATA ASPECT/1.35/
    1 FORMAT('frequency =',F10.3,A)
```

```
      CALL QRMODE(MODET,NCOLT)
      CALL QVIDBD(IBOARD)
      IF(IBOARD.LT.1.OR.IBOARD.GT.3)  THEN
       WRITE(*,*)' Graphics board not installed!'
       RETURN
      ENDIF
      IF(IBOARD.EQ.1)  MODE=6
      IF(IBOARD.EQ.2)  MODE=16
      IF(IBOARD.EQ.3)  MODE=18
      XMIN=X(1)
      XMAX=X(N)
      YMIN=Y(1)
      YMAX=Y(N)
      DO 21 I=1,N
       IF(XMIN.GT.X(I))  XMIN=X(I)
       IF(XMAX.LT.X(I))  XMAX=X(I)
       IF(YMIN.GT.Y(I))  YMIN=Y(I)
       IF(YMAX.LT.Y(I))  YMAX=Y(I)
   21 CONTINUE
      IF(YMIN.GT.0.0)  YMIN=0.0
      IXLAB=1
      IF(XMAX.LT.0.1)  IXLAB=2
      IF(XMAX.LT.0.01)  IXLAB=3
      IF(XMAX.LT.0.001)  IXLAB=4
      IF(XMAX.GT.10.0)  IXLAB=5
      IF(XMAX.GT.100.0)  IXLAB=6
      IF(XMAX.GT.1000.0)  IXLAB=7
      IYLAB=1
      IF(YMAX.LT.0.1)  IYLAB=2
      IF(YMAX.LT.0.01)  IYLAB=3
      IF(YMAX.LT.0.001)  IYLAB=4
      IF(YMAX.GT.10.0)  IYLAB=5
      IF(YMAX.GT.100.0)  IYLAB=6
      IF(YMAX.GT.1000.0)  IYLAB=7
      IF(IXLAB.NE.1)  THEN
       IF(IXLAB.EQ.2)  XFAC=10.0
       IF(IXLAB.EQ.3)  XFAC=100.0
       IF(IXLAB.EQ.4)  XFAC=1000.0
       IF(IXLAB.EQ.5)  XFAC=0.01
       IF(IXLAB.EQ.6)  XFAC=0.001
       IF(IXLAB.EQ.7)  XFAC=0.0001
       XMIN=XMIN*XFAC
       XMAX=XMAX*XFAC
       DO 22 I=1,N
        X(I)=X(I)*XFAC
   22 CONTINUE
      ENDIF
      IF(IYLAB.NE.1)  THEN
       IF(IYLAB.EQ.2)  YFAC=10.0
       IF(IYLAB.EQ.3)  YFAC=100.0
       IF(IYLAB.EQ.4)  YFAC=1000.0
       IF(IYLAB.EQ.5)  YFAC=0.01
       IF(IYLAB.EQ.6)  YFAC=0.001
       IF(IYLAB.EQ.7)  YFAC=0.0001
       YMIN=YMIN*YFAC
       YMAX=YMAX*YFAC
       DO 23 I=1,N
        Y(I)=Y(I)*YFAC
   23 CONTINUE
      ENDIF
      XLABL(1)=LABLX
      XLABL(2)=LABFAC(IXLAB)
      YLABL(1)=LABLY
      YLABL(2)=LABFAC(IYLAB)
      XMAJ=0.2*(XMAX-XMIN)
      YMAJ=0.2*(YMAX-YMIN)
      ICOLR=4
      IFIL=3
      ILIN=1
      CALL QSMODE(MODE)
      IF(IBOARD.NE.1)  THEN
       CALL QPREG(0,ICOLR)
      ENDIF
      JCOL1=150
      JCOL2=500
      JROW1=40
      IF(MODE.EQ.6)  JROW1=60
      JROW2=149
      IF(MODE.EQ.16)  JROW2=299
      IF(MODE.EQ.18)  JROW2=419
      XORG=XMIN
      YORG=YMIN
      YOVERX=1.0
      IOPT=0
      IF(FFAC.EQ.1.0)  THEN
       WRITE(FREQL,1)FREQ,RADHER(1)
      ELSE
       WRITE(FREQL,1)FREQ,RADHER(2)
      ENDIF
      IF(MODE.NE.18)  THEN
       CALL QPTXT(60,TITLE,7,5,23)
       CALL QPTXT(29,FREQL,7,25,22)
      ELSE
       CALL QPTXT(60,TITLE,7,5,29)
       CALL QPTXT(29,FREQL,7,25,28)
      ENDIF
      CALL QPTXT(8,YLABL(1),7,2,15)
      CALL QPTXT(8,YLABL(2),7,2,14)
      CALL QPLOT(JCOL1,JCOL2,JROW1,JROW2,XMIN,XMAX,YMIN,YMAX,
     *           XORG,YORG,IOPT,YOVERX,ASPECT)
      CALL QSETUP(0,ILIN,-2,IFIL)
      CALL QXAXIS(XMIN,XMAX,0.0,0,0,0)
      CALL QPTXTA(16,XLABL,7)
      CALL QXAXIS(XMIN,XMAX,XMAJ,0,-1,2)
      CALL QYAXIS(YMIN,YMAX,YMAJ,0,-1,2)
      CALL QTABL(1,N,X,Y)
   24 CONTINUE
      CALL QONKEY(IKEY)
      IF(IKEY.EQ.0)  GO TO 24
      CALL QINKEY(IEXTEN,IKEY)
      CALL QSMODE(MODET)
   25 CONTINUE
      IF(IXLAB.NE.1)  THEN
       DO 31 I=1,N
        X(I)=X(I)/XFAC
   31 CONTINUE
      ENDIF
      IF(IYLAB.NE.1)  THEN
       DO 32 I=1,N
        Y(I)=Y(I)/YFAC
   32 CONTINUE
      ENDIF
      RETURN
      END
      SUBROUTINE READIN
C       Reads input data
      COMMON /CMPVAL/X1,Y1,Z1,W1,M1,P0,P1,U0,U1,RFH,RFK,RFP,
     *               S,GF,GOX,RFA,RFC
      COMMON /RELVAL/N,TAU,DTAU,NR,RBAR,MBAR,GAMMA,P00,DHLDR,CSTAR,
     *        DCSDR,RHOLO,ULO,LAMDA,MU,TAUT,UBAR(50),XBAR(50),XLC
      COMMON /INTVAL/NVAL
      COMMON /DIMVAL/HOLDD(20),XBARD(50),PBAR(50),TBAR(50)
      COMMON /TITL/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      CHARACTER*60 TITLE
      CHARACTER*40 TITLF
      REAL MBAR,N,NR,LAMDA,MU,RVAR(15)
      REAL MBARD,ND,NRD,LAMDAD,MUD,HOLD(20)
      COMPLEX S,X1,Y1,Z1,W1,M1,P0,P1,U0,U1,GF,GOX,RFH,RFK,RFP,RFA,RFC
      COMPLEX CVAR(17)
      EQUIVALENCE (N,RVAR(1)),(X1,CVAR(1))
      EQUIVALENCE (ND,HOLD(1)),(TAUD,HOLD(2)),(DTAUD,HOLD(3)),
     *            (NRD,HOLD(4)),(LAMDAD,HOLD(5)),(MUD,HOLD(6)),
     *            (CDIAM,HOLD(7)),(TDIAM,HOLD(8)),(XLCD,HOLD(9)),
     *            (GAMMAD,HOLD(10)),(RGAS,HOLD(11)),(P00D,HOLD(12)),
     *            (MBARD,HOLD(13)),(RBARD,HOLD(14)),(DCSDRD,HOLD(15)),
     *            (DHLDRD,HOLD(16)),(RHOLOD,HOLD(17)),(ULOD,HOLD(18)),
     *            (PCHMB,HOLD(19)),(TCHMB,HOLD(20))
      CHARACTER*8 VAR(20),VARP(20),VARL(20),NAME
      CHARACTER*1 ANS
      DATA IGO/0/
      DATA VAR /'    ND =','  TAUD =',' DTAUD =','   NRD =','LAMDAD =',
     *          '   MUD =',' CDIAM =',' TDIAM =','  XLCD =','GAMMAD =',
     *          '  RGAS =','  P00D =',' MBARD =',' RBARD =','DCSDRD =',
     *          'DHLDRD =','RHOLOD =','  ULOD =',' PCHMB =',' TCHMB ='/
```

```
      DATA VARP/'ND      ','TAUD    ','DTAUD   ','NRD     ','LAMDAD  ',
     *          'MUD     ','CDIAM   ','TDIAM   ','XLCD    ','GAMMAD  ',
     *          'RGAS    ','P00D    ','MBARD   ','RBARD   ','DCSDRD  ',
     *          'DHLDRD  ','RHOLOD  ','ULOD    ','PCHMB   ','TCHMB   '/
      DATA VARL/'nd      ','taud    ','dtaud   ','nrd     ','lamdad  ',
     *          'mud     ','cdiam   ','tdiam   ','xlcd    ','gammad  ',
     *          'rgas    ','p00d    ','mbard   ','rbard   ','dcsdrd  ',
     *          'dhldrd  ','rholod  ','ulod    ','pchmb   ','tchmb   '/
    1 FORMAT(16I5)
    2 FORMAT(4E15.6)
    3 FORMAT(3E15.6)
    4 FORMAT(A)
    5 FORMAT(' Enter X (ft), P (lbf/ft^2), and T (°R) for point ',
     *        I3,'  ')
    6 FORMAT(1P4E15.6)
    7 FORMAT(2X,A8,2X,A8,2X,A8,2X,A8,2X,A8)
    8 FORMAT(2X,A8,1PE13.5,2X,A8,E13.5,2X,A8,E13.5)
    9 FORMAT(1P3E15.6)
   10 FORMAT(A40,2X,I2.2,':',I2.2,A2,3X,I2.2,'-',I2.2,'-',I2.2)
      IF(IGO.EQ.1)  THEN
       WRITE(*,'(A\)')' Do you wish to use old data with or without chan
     *ges? Y or N '
       READ(*,4)ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  GO TO 24
      ENDIF
      IGO=1
      WRITE(*,*)' '
      WRITE(*,'(A\)')' Is your rocket input on file? Y OR N '
      READ(*,4)ANS
      IF(ANS.NE.'N'.AND.ANS.NE.'n')  THEN
       WRITE(*,'(A\)')' Does the file need to be rewound? Y OR N '
       READ(*,4)ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  REWIND 15
       READ(15,4,END=99)TITLF
       WRITE(TITLE,10)TITLF,IHR,IMIN,AP,IMON,IDAY,IYR
       READ(15,1,END=99)NVAL
       IF(NVAL.EQ.0)  GO TO 99
       READ(15,3)(XBARD(I),PBAR(I),TBAR(I),I=1,NVAL)
       PCHMB=PBAR(1)
       TCHMB=TBAR(1)
       READ(15,2)ND,TAUD,DTAUD,NRD
       READ(15,2)LAMDAD,MUD
       READ(15,2)CDIAM,TDIAM,XLCD
       READ(15,2)GAMMAD,RGAS,P00D
       READ(15,2)MBARD,RBARD
       READ(15,2)DCSDRD,DHLDRD,RHOLOD,ULOD
      ELSE
       WRITE(*,'(A\)')' How many points along centerline? '
       READ(*,*,END=99)NVAL
       IF(NVAL.EQ.0)  GO TO 99
       DO 21 I=1,NVAL
        WRITE(*,5)I
     READ(*,*)XBARD(I),PBAR(I),TBAR(I)
21 CONTINUE
    PCHMB=PBAR(1)
    TCHMB=TBAR(1)
    WRITE(*,*)' Enter Title'
    READ(*,4)TITLF
    WRITE(TITLE,10)TITLF,IHR,IMIN,AP,IMON,IDAY,IYR
    WRITE(*,*)' Enter N (pressure interaction index) and NR',
  *          ' (enthalpy interaction index)'
    READ(*,*)ND,NR
    WRITE(*,*)' Enter TAU (sensitive time lag - sec) and DTAU',
  *          ' (invarient time lag - sec)'
    READ(*,*)TAUD,DTAUD
    WRITE(*,*)' Enter LAMDA and MU (real and imaginary parts',
  *          ' of frequency'
    READ(*,*)LAMDAD,MUD
    WRITE(*,*)' Enter XLCD (length of combustion chamber - ft)'
    READ(*,*)XLCD
    WRITE(*,*)' Enter CDIAM (chamber diameter - ft) and TDIAM',
  *          ' (throat diameter - ft)'
    READ(*,*)CDIAM,TDIAM
    WRITE(*,*)' Enter GAMMA (ratio of specific heats), RGAS',
  *          ' (gas constant - ft^2/sec^2/'R)'
    READ(*,*)GAMMAD,RGAS
    WRITE(*,*)' Enter P00 (maximum overpressure - lbf/ft^2)'
    READ(*,*)P00D
    WRITE(*,*)' Enter MBAR (mean combustion response function -',
  *          ' lbm/sec)'
    WRITE(*,*)'   and RBAR (mean mixture ratio)'
    READ(*,*)MBARD,RBARD
    WRITE(*,*)' Enter DCSDR (dc*/dr - ft/sec) and DHLDR',
  *          ' (dh/dr - ft^2/sec^2)'
    READ(*,*)DCSDRD,DHLDRD
    WRITE(*,*)' Enter RHOLO (mass of liquid/unit chamber vol -',
  *          'lbm/ft^3)'
    WRITE(*,*)'   and ULO (axial component of liquid velocity',
  *          ' - ft/sec)'
    READ(*,*)RHOLOD,ULOD
    WRITE(15,4)TITLF
    WRITE(15,1)NVAL
    WRITE(15,9)(XBARD(I),PBAR(I),TBAR(I),I=1,NVAL)
    WRITE(15,6)ND,TAUD,DTAUD,NR
    WRITE(15,6)LAMDAD,MUD
    WRITE(15,6)CDIAM,TDIAM,XLCD
    WRITE(15,6)GAMMAD,RGAS,P00D
    WRITE(15,6)MBARD,RBARD
    WRITE(15,6)DCSDRD,DHLDRD,RHOLOD,ULOD
   ENDIF
   CALL NONDIM(HOLD)
   RETURN
24 CONTINUE
   WRITE(*,'(A\)')'  are there any changes? Y or N '
```

```
      READ(*,4)ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  THEN
       CALL NONDIM(HOLD)
       RETURN
      ENDIF
      WRITE(*,'(A\)')'  Do you wish to change title? Y or N '
      READ(*,4)ANS
      IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
       WRITE(*,*)' Enter Title'
       READ(*,4)TITLF
       WRITE(TITLE,10)TITLF,IHR,IMIN,AP,IMON,IDAY,IYR
      ENDIF
      GO TO 29
   27 CONTINUE
      WRITE(*,*)'   VARIABLE NAMES AND DESCRIPTIONS'
      WRITE(*,*)' '
      WRITE(*,*)' ND     - pressure interaction index'
      WRITE(*,*)' TAUD   - sensitive time lag                     sec'
      WRITE(*,*)' DTAUD  - invarient time lag                     sec'
      WRITE(*,*)' NRD    - enthalpy interaction index'
      WRITE(*,*)' LAMDAD - damping of perturbation'
      WRITE(*,*)' MUD    - frequency of perturbation'
      WRITE(*,*)' CDIAM  - chamber diameter                       ft'
      WRITE(*,*)' TDIAM  - throat diameter                        ft'
      WRITE(*,*)' XLCD   - length of combustion chamber           ft'
      WRITE(*,*)' GAMMAD - ratio of specific heats'
      WRITE(*,*)' RGAS   - gas constant                           ',
     *          '(ft/sec)^2/°R'
      WRITE(*,*)' POOD   - maximum pressure                       ',
     *          'lbf/ft^2'
      WRITE(*,*)' MBARD  - mean combustion response funct.        ',
     *          'lbm/sec'
      WRITE(*,*)' RBARD  - mean mixture ratio'
      WRITE(*,*)' DCSDRD - d(c*)/d(mixture ratio)                 ft/sec'
      WRITE(*,*)' DHLDRD - d(enthalpy)/d(mixture ratio)           ',
     *          'ft^2/sec^2'
      WRITE(*,*)' RHOLOD - mass of liquid/unit chamber volume     ',
     *          'lbm/ft^3'
      WRITE(*,*)' ULOD   - axial component of liquid velocity     ft/sec'
      WRITE(*,*)' PCHMB  - chamber pressure at injector           ',
     *          'lbf/ft^2'
      WRITE(*,*)' TCHMB  - chamber temperature                    °R'
      WRITE(*,*)' '
      GO TO 30
   28 CONTINUE
      WRITE(*,*)'   VARIABLE NAMES AND VALUES'
      WRITE(*,*)' '
      WRITE(*,8)(VAR(I),HOLD(I),I=1,20)
   29 CONTINUE
      WRITE(*,*)' '
      WRITE(*,*)' Enter ? to print variable names & descriptions'
      WRITE(*,*)'       # to print variable names & values'
      WRITE(*,*)'       END when all changes have been made'
      WRITE(*,*)' '
   30 CONTINUE
      WRITE(*,'(A\)')'   Enter variable name and new value, END, ?, or #
     * '
      CALL ZREAD(NAME,VALUE)
      IF(NAME.EQ.'?')  GO TO 27
      IF(NAME.EQ.'#')  GO TO 28
      IF(NAME.EQ.'END'.OR.NAME.EQ.'end')  THEN
       CALL NONDIM(HOLD)
       RETURN
      ENDIF
      DO 31 II=1,20
       I=II
       IF(NAME.EQ.VARP(I).OR.NAME.EQ.VARL(I))  GO TO 32
   31 CONTINUE
      WRITE(*,*)'     Invalid name, try again'
      GO TO 27
   32 CONTINUE
      HOLD(I)=VALUE
      GO TO 30
   99 CONTINUE
      STOP
      END
      SUBROUTINE SETVAL(VAL,ID)
C       Sets value from iterated variable
      COMMON /DIMVAL/HOLDD(20),XBARD(50),PBAR(50),TBAR(50)
      VAL=HOLDD(ID)
      RETURN
      END
      SUBROUTINE SETVAR(VAL,ID)
C       Sets iterated variable from value
      COMMON /CMPVAL/X1,Y1,Z1,W1,M1,P0,P1,U0,U1,RFH,RFK,RFP,
     *               S,GF,GOX,RFA,RFC
      COMMON /RELVAL/N,TAU,DTAU,NR,RBAR,MBAR,GAMMA,P00,DHLDR,CSTAR,
     *        DCSDR,RHOLO,ULO,LAMDA,MU,TAUT,UBAR(50),XBAR(50),XLC
      COMMON /RESULT/PP,UP,SIGP,FUNB
      COMMON /INTVAL/NVAL
      COMMON /DIMVAL/HOLDD(20),XBARD(50),PBAR(50),TBAR(50)
      REAL MBAR,N,NR,LAMDA,MU,RVAR(13)
      REAL MBARD,ND,NRD,LAMDAD,MUD
      COMPLEX S,X1,Y1,Z1,W1,M1,P0,P1,U0,U1,GF,GOX,RFH,RFK,RFP,RFA,RFC
      COMPLEX PP,UP,SIGP,FUNB,CVAR(17)
      EQUIVALENCE (N,RVAR(1)),(X1,CVAR(1))
      EQUIVALENCE
     *    (ND,HOLDD(1)),(TAUD,HOLDD(2)),(DTAUD,HOLDD(3)),
     *    (NRD,HOLDD(4)),(LAMDAD,HOLDD(5)),(MUD,HOLDD(6)),
     *    (CDIAM,HOLDD(7)),(TDIAM,HOLDD(8)),(XLCD,HOLDD(9)),
     *    (GAMMAD,HOLDD(10)),(RGAS,HOLDD(11)),(P00D,HOLDD(12)),
     *    (MBARD,HOLDD(13)),(RBARD,HOLDD(14)),(DCSDRD,HOLDD(15)),
     *    (DHLDRD,HOLDD(16)),(RHOLOD,HOLDD(17)),(ULOD,HOLDD(18)),
     *    (PCHMB,HOLDD(19)),(TCHMB,HOLDD(20))
```

```
      DATA PI/3.141593/,GC/32.174/
      HOLDD(ID)=VAL
      IF(ID.EQ.1)  THEN
C                          ND
       N=ND
       RETURN
      ENDIF
      IF(ID.EQ.2)  THEN
C                          TAUD
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       TAU=TAUD*ASTAR/XLCD
       TAUT=TAU+DTAU
       RETURN
      ENDIF
      IF(ID.EQ.3)  THEN
C                          DTAUD
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       DTAU=DTAUD*ASTAR/XLCD
       TAUT=TAU+DTAU
       RETURN
      ENDIF
      IF(ID.EQ.4)  THEN
C                          NRD
       NR=NRD
       RETURN
      ENDIF
      IF(ID.EQ.5)  THEN
C                          LAMDAD
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       LAMDA=LAMDAD*XLCD/ASTAR
       S=CMPLX(LAMDA,MU)
       RETURN
      ENDIF
      IF(ID.EQ.6)  THEN
C                          MUD
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       MU=MUD*XLCD*PI/ASTAR
       S=CMPLX(LAMDA,MU)
       RETURN
      ENDIF
      IF(ID.EQ.7)  THEN
C                          CDIAM
       CAREA=0.25*PI*CDIAM**2
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       DO 21 I=1,NVAL
        RHOBAR=PBAR(I)*GC/(RGAS*TBAR(I))
        UBARD=MBARD/(RHOBAR*CAREA)
        UBAR(I)=UBARD/ASTAR
   21 CONTINUE
       RETURN
      ENDIF
      IF(ID.EQ.8)  THEN
```

```
C                        TDIAM
       TAREA=0.25*PI*TDIAM**2
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       CSTARD=PBAR(NVAL)*TAREA*GC/MBARD
       CSTAR=CSTARD/ASTAR
       RETURN
      ENDIF
      IF(ID.EQ.9)  THEN
C                        XLCD
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       TAU=TAUD*ASTAR/XLCD
       DTAU=DTAUD*ASTAR/XLCD
       TAUT=TAU+DTAU
       LAMDA=LAMDAD*XLCD/ASTAR
       MU=MUD*XLCD*PI/ASTAR
       S=CMPLX(LAMDA,MU)
       DO 22 I=1,NVAL
        XBAR(I)=XBARD(I)/XLCD
   22 CONTINUE
       RETURN
      ENDIF
      IF(ID.EQ.10)  THEN
C                        GAMMAD
       GAMMA=GAMMAD
       CAREA=0.25*PI*CDIAM**2
       TAREA=0.25*PI*TDIAM**2
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       TAU=TAUD*ASTAR/XLCD
       DTAU=DTAUD*ASTAR/XLCD
       TAUT=TAU+DTAU
       LAMDA=LAMDAD*XLCD/ASTAR
       MU=MUD*XLCD*PI/ASTAR
       S=CMPLX(LAMDA,MU)
       ULO=ULOD/ASTAR
       DCSDR=DCSDRD/ASTAR
       RHOB1=PBAR(1)*GC/(RGAS*TBAR(1))
       MBAR=MBARD/(RHOB1*ASTAR*CAREA/XLCD)
       CSTARD=PBAR(NVAL)*TAREA*GC/MBARD
       CSTAR=CSTARD/ASTAR
       DO 23 I=1,NVAL
        RHOBAR=PBAR(I)*GC/(RGAS*TBAR(I))
        UBARD=MBARD/(RHOBAR*CAREA)
        UBAR(I)=UBARD/ASTAR
   23 CONTINUE
       RETURN
      ENDIF
      IF(ID.EQ.11)  THEN
C                        RGAS
       CAREA=0.25*PI*CDIAM**2
       TAREA=0.25*PI*TDIAM**2
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       TAU=TAUD*ASTAR/XLCD
       DTAU=DTAUD*ASTAR/XLCD
       TAUT=TAU+DTAU
       LAMDA=LAMDAD*XLCD/ASTAR
       MU=MUD*XLCD*PI/ASTAR
       S=CMPLX(LAMDA,MU)
       ULO=ULOD/ASTAR
       DCSDR=DCSDRD/ASTAR
       RHOB1=PBAR(1)*GC/(RGAS*TBAR(1))
       RHOLO=RHOLOD/RHOB1
       MBAR=MBARD/(RHOB1*ASTAR*CAREA/XLCD)
       CSTARD=PBAR(NVAL)*TAREA*GC/MBARD
       CSTAR=CSTARD/ASTAR
       DO 24 I=1,NVAL
        RHOBAR=PBAR(I)*GC/(RGAS*TBAR(I))
        UBARD=MBARD/(RHOBAR*CAREA)
        UBAR(I)=UBARD/ASTAR
   24 CONTINUE
       RETURN
      ENDIF
      IF(ID.EQ.12)  THEN
C                          POOD
       POO=POOD/PCHMB
       RETURN
      ENDIF
      IF(ID.EQ.13)  THEN
C                          MBARD
       CAREA=0.25*PI*CDIAM**2
       TAREA=0.25*PI*TDIAM**2
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       RHOB1=PBAR(1)*GC/(RGAS*TBAR(1))
       MBAR=MBARD/(RHOB1*ASTAR*CAREA/XLCD)
       CSTARD=PBAR(NVAL)*TAREA*GC/MBARD
       CSTAR=CSTARD/ASTAR
       DO 25 I=1,NVAL
        RHOBAR=PBAR(I)*GC/(RGAS*TBAR(I))
        UBARD=MBARD/(RHOBAR*CAREA)
        UBAR(I)=UBARD/ASTAR
   25 CONTINUE
       RETURN
      ENDIF
      IF(ID.EQ.14)  THEN
C                          RBARD
       RBAR=RBARD
       RETURN
      ENDIF
      IF(ID.EQ.15)  THEN
C                          DCSDRD
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       DCSDR=DCSDRD/ASTAR
       RETURN
      ENDIF
      IF(ID.EQ.16)  THEN
```

```
C                         DHLDRD
       DHLDR=DHLDRD
       RETURN
      ENDIF
      IF(ID.EQ.17)  THEN
C                         RHOLOD
       RHOB1=PBAR(1)*GC/(RGAS*TBAR(1))
       RHOLO=RHOLOD/RHOB1
       RETURN
      ENDIF
      IF(ID.EQ.18)  THEN
C                         ULOD
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       ULO=ULOD/ASTAR
       RETURN
      ENDIF
      IF(ID.EQ.19)  THEN
C                         PCHMB
       CAREA=0.25*PI*CDIAM**2
       TAREA=0.25*PI*TDIAM**2
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       FAC=PCHMB/PBAR(1)
       DO 26 I=1,NVAL
        PBAR(I)=FAC*PBAR(I)
        RHOBAR=PBAR(I)*GC/(RGAS*TBAR(I))
        UBARD=MBARD/(RHOBAR*CAREA)
        UBAR(I)=UBARD/ASTAR
   26 CONTINUE
       CSTARD=PBAR(NVAL)*TAREA*GC/MBARD
       CSTAR=CSTARD/ASTAR
       RHOB1=PBAR(1)*GC/(RGAS*TBAR(1))
       RHOLO=RHOLOD/RHOB1
       MBAR=MBARD/(RHOB1*ASTAR*CAREA/XLCD)
       POO=POOD/PCHMB
       RETURN
      ENDIF
      IF(ID.EQ.20)  THEN
C                         TCHMB
       DO 27 I=1,NVAL
        TBAR(I)=FAC*TBAR(I)
   27 CONTINUE
       CAREA=0.25*PI*CDIAM**2
       TAREA=0.25*PI*TDIAM**2
       ASTAR=SQRT(GAMMAD*RGAS*TBAR(1))
       FAC=TCHMB/TBAR(1)
       DO 28 I=1,NVAL
        RHOBAR=PBAR(I)*GC/(RGAS*TBAR(I))
        UBARD=MBARD/(RHOBAR*CAREA)
        UBAR(I)=UBARD/ASTAR
   28 CONTINUE
       CSTARD=PBAR(NVAL)*TAREA*GC/MBARD
       CSTAR=CSTARD/ASTAR
       RHOB1=PBAR(1)*GC/(RGAS*TBAR(1))
       RHOLO=RHOLOD/RHOB1
       MBAR=MBARD/(RHOB1*ASTAR*CAREA/XLCD)
      ENDIF
      RETURN
      END
      SUBROUTINE ZREAD(NAME,VALUE)
C       Reads input for input modification
      CHARACTER*1 NAME(8)
      CHARACTER*1 CARD(80),PLUS,MINUS,PERIOD,LE,E,NUMBER(10)
      CHARACTER*1 LEND(3),CEND(3),POUND,QUEST,BLK,COMMA
      CHARACTER*80 DCARD
      EQUIVALENCE (CARD(1),DCARD)
      DATA PLUS/'+'/,MINUS/'-'/,PERIOD/'.'/,LE/'e'/,E/'E'/,BLK/' '/
      DATA NUMBER/'0','1','2','3','4','5','6','7','8','9'/,COMMA/','/
      DATA LEND/'e','n','d'/,CEND/'E','N','D'/,POUND/'#'/,QUEST/'?'/
    1 FORMAT(A)
      DO 21 I=1,8
       NAME(I)=BLK
   21 CONTINUE
      READ(*,1)DCARD
      IF(CARD(1).EQ.POUND)  THEN
       NAME(1)=POUND
       RETURN
      ENDIF
      IF(CARD(1).EQ.QUEST)  THEN
       NAME(1)=QUEST
       RETURN
      ENDIF
      DO 22 I=1,3
       IF(CARD(I).NE.LEND(I).AND.CARD(I).NE.CEND(I))  GO TO 23
       NAME(I)=CEND(I)
   22 CONTINUE
      RETURN
   23 CONTINUE
      DO 24 I=1,8
       II=I
       IF(CARD(I).EQ.BLK.OR.CARD(I).EQ.COMMA)  GO TO 25
       NAME(I)=CARD(I)
   24 CONTINUE
   25 CONTINUE
      DO 26 I=II,80
       ID=I
       IF(CARD(I).NE.BLK.AND.CARD(I).NE.COMMA)  GO TO 27
   26 CONTINUE
      VALUE=0.0
      WRITE(*,*)'   No value given, ZERO assumed'
      RETURN
   27 CONTINUE
      SIGN=1.0
      IF(CARD(ID).EQ.MINUS)  THEN
       SIGN=-1.0
    ID=ID+1
   ELSEIF(CARD(ID).EQ.PLUS)  THEN
    ID=ID+1
   ENDIF
   WHOLE=0.0
   DO 30 I=ID,80
    II=I
    IF(CARD(I).EQ.PERIOD)  GO TO 31
    IF(CARD(I).EQ.PLUS)  GO TO 36
    IF(CARD(I).EQ.MINUS)  GO TO 36
    IF(CARD(I).EQ.E.OR.CARD(I).EQ.LE)  GO TO 35
    DO 28 J=1,10
     JJ=J-1
     IF(CARD(I).EQ.NUMBER(J))  GO TO 29
28 CONTINUE
    VALUE=SIGN*WHOLE
    IF(CARD(I).EQ.BLK)  RETURN
    WRITE(*,*)'  Input error, value set to ZERO'
    VALUE=0.0
    RETURN
29 CONTINUE
    WHOLE=WHOLE*10.0+JJ
30 CONTINUE
   VALUE=SIGN*WHOLE
   RETURN
31 CONTINUE
   ID=II+1
   FRACT=0.0
   ICOUNT=0
   DO 34 I=ID,80
    ICOUNT=ICOUNT+1
    II=I
    IF(CARD(I).EQ.PERIOD)  THEN
     WRITE(*,*)'  Input error, value set to ZERO'
     VALUE=0.0
     RETURN
    ENDIF
    IF(CARD(I).EQ.PLUS)  GO TO 36
    IF(CARD(I).EQ.MINUS)  GO TO 36
    IF(CARD(I).EQ.E.OR.CARD(I).EQ.LE)  GO TO 35
    DO 32 J=1,10
     JJ=J-1
     IF(CARD(I).EQ.NUMBER(J))  GO TO 33
32 CONTINUE
    VALUE=SIGN*(WHOLE+FRACT)
    IF(CARD(I).EQ.BLK)  RETURN
    WRITE(*,*)'  Input error, value set to ZERO'
    VALUE=0.0
    RETURN
33 CONTINUE
    FRACT=FRACT+JJ/10.0**ICOUNT
34 CONTINUE
```

```
   VALUE=SIGN*(WHOLE+FRACT)
   RETURN
35 CONTINUE
   II=II+1
36 CONTINUE
   VALUE=SIGN*(WHOLE+FRACT)
   SIGN=1.0
   IF(CARD(II).EQ.MINUS)  THEN
    SIGN=-1.0
    II=II+1
   ELSEIF(CARD(II).EQ.PLUS)  THEN
    II=II+1
   ENDIF
   WHOLE=0.0
   DO 39 I=II,80
    DO 37 J=1,10
     JJ=J-1
     IF(CARD(I).EQ.NUMBER(J))  GO TO 38
37 CONTINUE
    VALUE=VALUE*10.0**(SIGN*WHOLE)
    IF(CARD(I).EQ.BLK)  RETURN
    WRITE(*,*)'  Input error, value set to ZERO'
    VALUE=0.0
    RETURN
38 CONTINUE
    WHOLE=WHOLE*10.0+JJ
39 CONTINUE
   VALUE=VALUE*10.0**(SIGN*WHOLE)
   RETURN
   END
```