

**Acoustic Boundary Condition Estimation in a  
Near-Scale Gas-Turbine Combustor**

by

Andrew D. Wright

Thesis submitted to the Faculty of the  
Virginia Polytechnic Institute and State University  
in partial fulfillment of the requirements for the degree of

Master of Science

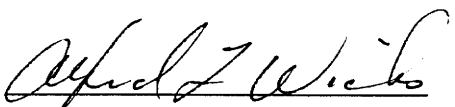
in

Mechanical Engineering

APPROVED:

  
\_\_\_\_\_  
Dr. J. R. Mahan, Chairman

  
\_\_\_\_\_  
Dr. E. P. Scott

  
\_\_\_\_\_  
Dr. A. L. Wicks

February, 1996

Blacksburg, Virginia

Key Words: Acoustics, Combustion Stability, Parameter Estimation

C.2

LD  
5655  
V855  
1996  
W754  
C.2

**Acoustic Boundary Condition Estimation in a  
Near-Scale Gas-Turbine Combustor**

by

Andrew D. Wright

J. Robert Mahan, Chairman

Mechanical Engineering

(ABSTRACT)

A method for determining the specific acoustic admittance of the inlet and outlet ports of a combustion chamber is presented. Parameter estimation techniques of Gauss linearization and genetic algorithms are employed to recover the acoustic boundary conditions. These techniques are used with a combination of two resources: dynamic pressure measurements obtained during combustor operation, and a finite element method-based model of the combustion chamber.

Results of a theoretical analysis are presented which show that the method is capable of accomplishing its mission. An observation of particular significance is the lack of sensitivity of the pressure mode shape to relatively large changes in the acoustic boundary conditions.

# Table of Contents

- 1.0 Introduction . . . . . 1**
  - 1.1 Introduction . . . . . 1
  - 1.2 Literature Review . . . . . 3
    - 1.2.1 Combustion Instability . . . . . 3
    - 1.2.2 Acoustic Boundary Conditions . . . . . 4
    - 1.2.3 Parameter Estimation . . . . . 7
  
- 2.0 Theoretical Background . . . . . 10**
  - 2.1 Combustion Instability . . . . . 10
  - 2.2 Acoustic Boundary Conditions . . . . . 19
  - 2.3 Parameter Estimation . . . . . 22
    - 2.3.1 The Gauss Method . . . . . 24
    - 2.3.2 Genetic Algorithms . . . . . 28
  - 2.4 Finite Element Model: DYNAMITE . . . . . 32
    - 2.4.1 Theory . . . . . 33



2.4.2	Modifications to DYNAMODE	35
<b>3.0</b>	<b>Approach</b>	<b>38</b>
3.1	Equipment	38
3.1.1	The METC Combustor	40
3.1.2	Program ESTIMAT	42
3.2	Analysis	50
<b>4.0</b>	<b>Results and Discussion</b>	<b>53</b>
<b>5.0</b>	<b>Conclusions and Recommendations</b>	<b>60</b>
	<b>References</b>	<b>127</b>
	<b>Appendix A. Finite Element Model Derivation</b>	<b>131</b>
	<b>Appendix B. Computer Code Listing</b>	<b>141</b>
	<b>Vita</b>	<b>209</b>

## List of Tables

Table 1.	Parameters Used to Determine the Theoretical Target Pressure Vectors . . . . .	63
Table 2.	Estimated Admittance Values for Cases 1 and 10. . . . .	64
Table 3.	Estimated Admittance Values for Cases 2 and 11. . . . .	65
Table 4.	Estimated Admittance Values for Cases 3 and 12. . . . .	66
Table 5.	Estimated Admittance Values for Cases 4 and 13. . . . .	67
Table 6.	Estimated Admittance Values for Cases 5 and 14. . . . .	68
Table 7.	Estimated Admittance Values for Cases 6 and 15. . . . .	69
Table 8.	Estimated Admittance Values for Cases 7 and 16. . . . .	70
Table 9.	Estimated Admittance Values for Cases 8 and 17. . . . .	71
Table 10.	Estimated Admittance Values for Cases 9 and 18. . . . .	72

## List of Illustrations

Fig. 1	Graphical Representation of Rayleigh's Criterion: (a) Amplification and (b) Damping. . . . .	12
Fig. 2	An Idealized Lean, Premixed Gas Turbine Combustion System . . . . .	15
Fig. 3	Stability Map for Premixed, Can-Type Combustors. . . . .	17
Fig. 4	Location of the Flame Front Due to Velocity Conditions. . . . .	18
Fig. 5	Collapse of One Side of a Six-Sided Element to Form an Element Suitable for Use on the Combustor Centerline. . . . .	37
Fig. 6	METC Combustor Contained in Pressure Vessel . . . . .	39
Fig. 7	Location of Pressure Probes in METC Combustor . . . . .	41
Fig. 8	Logic for Program ESTIMAT . . . . .	43
Fig. 9	Logic for Subroutine GAUSS . . . . .	45
Fig. 10	Logic for Subroutine GENETIC . . . . .	47
Fig. 10	Example of a Random Population Used in Subroutine GENETIC . . . . .	70
Fig. 11	Estimation Results of Case 1 Parameters . . . . .	73
Fig. 12	Estimation Results of Case 2 Parameters . . . . .	75
Fig. 13	Estimation Results of Case 3 Parameters . . . . .	77

Fig. 14	Estimation Results of Case 4 Parameters	. . . . .	79
Fig. 15	Estimation Results of Case 5 Parameters	. . . . .	81
Fig. 16	Estimation Results of Case 6 Parameters	. . . . .	83
Fig. 17	Estimation Results of Case 7 Parameters	. . . . .	85
Fig. 18	Estimation Results of Case 8 Parameters	. . . . .	87
Fig. 19	Estimation Results of Case 9 Parameters	. . . . .	89
Fig. 20	Estimation Results of Ideal Case 1 Parameters	. . . . .	91
Fig. 21	Estimation Results of Ideal Case 2 Parameters	. . . . .	92
Fig. 22	Estimation Results of Ideal Case 3 Parameters	. . . . .	93
Fig. 23	Estimation Results of Ideal Case 4 Parameters	. . . . .	94
Fig. 24	Estimation Results of Ideal Case 5 Parameters	. . . . .	95
Fig. 25	Estimation Results of Ideal Case 6 Parameters	. . . . .	96
Fig. 26	Estimation Results of Ideal Case 7 Parameters	. . . . .	97
Fig. 27	Estimation Results of Ideal Case 8 Parameters	. . . . .	98
Fig. 28	Estimation Results of Ideal Case 9 Parameters	. . . . .	99
Fig. 29	Estimation Results of Case 10 Parameters	. . . . .	100
Fig. 30	Estimation Results of Case 11 Parameters	. . . . .	102
Fig. 31	Estimation Results of Case 12 Parameters	. . . . .	104
Fig. 32	Estimation Results of Case 13 Parameters	. . . . .	106
Fig. 33	Estimation Results of Case 14 Parameters	. . . . .	108
Fig. 34	Estimation Results of Case 15 Parameters	. . . . .	110

Fig. 35	Estimation Results of Case 16 Parameters . . . . .	112
Fig. 36	Estimation Results of Case 17 Parameters . . . . .	114
Fig. 37	Estimation Results of Case 18 Parameters . . . . .	116
Fig. 38	Estimation Results of Ideal Case 10 Parameters . . . . .	118
Fig. 39	Estimation Results of Ideal Case 11 Parameters . . . . .	119
Fig. 40	Estimation Results of Ideal Case 12 Parameters . . . . .	120
Fig. 41	Estimation Results of Ideal Case 13 Parameters . . . . .	121
Fig. 42	Estimation Results of Ideal Case 14 Parameters . . . . .	122
Fig. 43	Estimation Results of Ideal Case 15 Parameters . . . . .	123
Fig. 44	Estimation Results of Ideal Case 16 Parameters . . . . .	124
Fig. 45	Estimation Results of Ideal Case 17 Parameters . . . . .	125
Fig. 46	Estimation Results of Ideal Case 18 Parameters . . . . .	126

## Nomenclature

$a_o$	thermodynamic speed of sound (ft/s) (m/s)
$A_c$	cross-sectional area of mixing chamber (ft <sup>2</sup> ) (m <sup>2</sup> )
$\mathbf{b}$	parameter estimate (ft <sup>3</sup> /lb <sub>r</sub> s) (m <sup>3</sup> /N·s)
$c$	thermodynamic speed of sound (ft/s) (m/s)
$E$	percent error (%)
$f$	frequency (Hz)
$F$	FEM forcing function matrix
$I_R$	value of the quantified form of Rayleigh's criterion
$j$	imaginary operator
$k$	iteration number
$K$	component of FEM stiffness matrix
$L_{cc}$	axial length of combustion chamber (ft) (m)
$L_{mc}$	axial length of mixing chamber (ft) (m)
$m$	number of experimental observations
$\dot{m}$	time averaged mass flow rate of fuel mixture (lb <sub>m</sub> /s) (kg/s)
$M$	component of FEM stiffness matrix
$\mathbf{n}$	unit vector
$p$	acoustic pressure (lb <sub>f</sub> /ft <sup>2</sup> ) (N/m <sup>2</sup> )
$P$	FEM acoustic pressure matrix (lb <sub>f</sub> /ft <sup>2</sup> ) (N/m <sup>2</sup> )
$\hat{P}$	predicted acoustic pressure (lb <sub>f</sub> /ft <sup>2</sup> ) (N/m <sup>2</sup> )
$q$	point heat source distribution (BTU/lb <sub>m</sub> ) (kJ/kg)
$r$	reflection coefficient

R	normalized radiation resistance
S	sum-of-squares error function ( $\text{lb}_f^2 \cdot \text{s} / \text{ft}^5$ ) ( $\text{N}^2 \cdot \text{s} / \text{m}^5$ )
$S_{ij}$	volume integral used in FEM model
t	time (s)
T	period (s)
u	acoustic particle velocity (ft/s) (m/s)
$U_{fm}$	mean velocity of fuel mixture (ft/s) (m/s)
$v_f$	flame velocity (ft/s) (m/s)
v	velocity (ft/s) (m/s)
X	normalized radiation reactance
$\mathbf{X}$	sensitivity matrix
y, $\mathbf{y}$	specific acoustic admittance ( $\text{ft}^3 / \text{lb}_f \cdot \text{s}$ ) ( $\text{m}^3 / \text{N} \cdot \text{s}$ )
$y_z^{\text{in}}$	axial component of the acoustic admittance at the burner inlet plane ( $\text{ft}^3 / \text{lb}_f \cdot \text{s}$ ) ( $\text{m}^3 / \text{N} \cdot \text{s}$ )
$y_z^{\text{out}}$	axial component of the acoustic admittance at the burner exhaust plane ( $\text{ft}^3 / \text{lb}_f \cdot \text{s}$ ) ( $\text{m}^3 / \text{N} \cdot \text{s}$ )
Y	experimentally observed pressure ( $\text{lb}_f / \text{in}^2$ ) ( $\text{N} / \text{m}^2$ )
z	specific acoustic impedance ( $\text{lb}_f \cdot \text{s} / \text{ft}^3$ ) ( $\text{N} \cdot \text{s} / \text{m}^3$ )
Z	normalized acoustic impedance

## Greek Symbols

$\alpha$	real component of specific acoustic admittance ( $\text{lb}_f / \text{ft}^2$ ) ( $\text{m}^3 / \text{N} \cdot \text{s}$ )
$\beta, \beta'$	parameter estimate ( $\text{lb}_f / \text{ft}^2$ ) ( $\text{m}^3 / \text{N} \cdot \text{s}$ )
$\gamma$	specific heat ratio
$\lambda$	wavelength (ft) (m)
$\rho_o$	mass density ( $\text{lb}_m / \text{ft}^3$ ) ( $\text{kg} / \text{m}^3$ )
$\tau$	critical time delay (s)
$\omega$	angular frequency (rad/s)
$\psi$	imaginary component of specific acoustic admittance ( $\text{lb}_f / \text{ft}^2$ ) ( $\text{m}^3 / \text{N} \cdot \text{s}$ )

## **1.0 Introduction**

### **1.1 Introduction**

Recognition of the harmful effects of some industrial combustion by-products has caused increasingly restrictive standards to be placed on CO and NO<sub>x</sub> emissions. Traditional approaches to meeting the requirements include water injection and emission scrubbers. These methods, however, can be very costly to implement. One relatively recent method of meeting the standards for industrial gas turbine engines is the use of lean, premixed combustion systems. Lean combustion systems operate at relatively low temperatures, discouraging the production of CO and NO<sub>x</sub> pollutants. Unfortunately, though, these systems have been found to be somewhat susceptible to fuel-feed instability.

Fuel-feed instability is a form of combustion instability that is caused by pressure fluctuations which accompany turbulent combustion. Under certain conditions these fluctuations can become sufficiently strong to actually modulate the equivalence ratio of the fuel-oxidizer mixture arriving at the flame front. When this occurs at a dangerous frequency (usually a resonant frequency of the burner), a fuel-feed instability may occur.



This instability, when present, results in excessive vibration and noise levels, which can lead to problems such as accelerated wear of seals and bearings, fatigue failure of sheet-metal components such as combustion liners and transition ducts, and even harmful physiological effects. In extreme cases the fuel-feed instability can lead to catastrophic failure of the combustion system.

Therefore it is desirable to eliminate the problem of fuel-feed instability in lean premixed combustion systems. One approach to accomplishing this is to investigate the dynamic behavior of combustion systems using mathematical models, and then to use this information, along with existing design standards, to design new combustors. By addressing the problem in the design stage, the need for costly prototypes and post-production modifications would be largely eliminated.

Existing combustion system dynamic models are difficult to apply, however, because of their dependence on knowledge of the acoustic boundary conditions of the combustion system. For practical combustion systems the acoustic boundary conditions are difficult, if not impossible, to determine theoretically. Therefore discovery of a method for determining the acoustic boundary conditions would signify a large step toward solving the problem of fuel-feed instability. Such a method could be used to study how impedance conditions of various practical combustors vary with frequency, and this knowledge could then be used in the design of combustion systems.

The objective of the current study is to develop a method for determining the acoustic boundary conditions of a combustion chamber using a mathematical model

based on the finite element method (FEM) with experimentally obtained dynamic pressure measurements. Introductory theory pertaining to combustion instability, acoustic boundary conditions, finite element modeling, and parameter estimation are first presented. Then an extensive discussion of the final method and sample results are given.

The current study lays the groundwork for future work in the prediction and elimination of fuel-feed instability. The method could be employed to produce scaling laws for the acoustic boundary conditions of various types of combustors. Accurate estimates of the boundary conditions obtained from such laws could then be utilized by designers, improving the reliability of mathematical simulations.

## **1.2 Literature Review**

The following is a review of literature pertinent to the study. Areas of interest include combustion instability, acoustic boundary conditions, and parameter estimation techniques.

### **1.2.1 Combustion Instability**

Combustion oscillations are characterized by a closed-loop cause-and-effect relationship between pressure and heat release oscillations in a combustion chamber. Lord Rayleigh [1] first recognized and qualitatively described this relationship in the nineteenth century. Since then combustion oscillations, and the possible subsequent

instabilities, have become recognized as a persistent problem in a variety of combustion applications, ranging from liquid-fueled rockets to gas turbines. Putnam [2] quantified Rayleigh's Criterion in 1971, and his approach has been the basis of many analyses of combustion instability.

Proposed methods for preventing combustion oscillations range from design criteria presented by Baade [3], to active control by cyclic fuel injection proposed by Richards et al. [4]. However, as Richards et al. state, "practical solutions to instability problems are often clouded by uncertainties over the specific mechanisms driving a given oscillation." Although the acoustic boundary conditions are not a direct driving mechanism, they do contribute indirectly by providing an environment in which instabilities can occur.

### **1.2.2 Acoustic Boundary Conditions**

The acoustic boundary condition is a significant phenomenon which occurs at any place where a traveling acoustic wave encounters a change in transmission medium. Such conditions play practical roles in systems including the flanged, open-ended ducts of musical instruments, the closed-ended ducts of organ pipes, and the sound deadening walls of an anechoic chamber.

Study of the acoustic boundary conditions of open-ended ducts has been of continuing interest for over a century. Interest stems from the many practical applications of ducted systems, ranging from industrial gas-turbine combustors to

residential ventilating systems. Open-ended ducts represent an important special case, and by studying open-ended duct impedance, much can be learned about the acoustic behavior of other duct terminations. The initial contribution to the field came in 1948 when Levine and Schwinger [5] presented their classic analytical treatment of open-ended duct terminating impedance. They addressed a system of plane-wave propagation in an open-ended circular duct of negligible wall thickness with no mean flow and in the absence of temperature gradients.

The first extension of this inaugural work came from Carrier [6] in 1956. He considered the same open-ended duct but introduced mean flow inside and outside of the duct. A major conclusion was that flow exiting the duct at low Mach numbers ( $< 0.3$ ) has little effect on radiated sound from the duct.

The effect of finite wall thickness on impedance was treated by Ando [7] in 1969. He, like Levine and Schwinger, assumed plane wave propagation with no mean flow and no temperature gradients. Ando's results indicate a significant influence on the acoustic impedance due to wall thickness ratio, and are in good agreement with experimental results.

Johnson and Ogimoto [7,8] present results concerning the effects of mean flow and frequency in 1980. As in reference 6, it was found that low Mach number flow has little or no impact. Frequency, on the other hand, was shown to have a strong influence on terminating impedance.

As systems of interest became less idealized (i.e. nonisentropic, elevated temperatures), strict analytical treatment gave way to experimental analyses. The first experimental study of the effects of heated flow on radiation impedance is given by Cummings [10] in 1977. In measuring the terminating impedance of an open-ended circular duct at elevated temperatures with low air flow, he showed that resistance (the real component of impedance) changes with temperature while reactance (the imaginary component) does not. Mahan, Cline, and Jones [11] substantiate this finding, and present a correlation for the resistance based upon the temperature difference between the duct exit plane and the ambient. Cline [12] goes on to experimentally show that the duct terminating impedance is dominated by the reactance at low frequencies, but tends toward a resistance domination at higher frequencies.

Many experimental analyses, of which references 10 through 12 are examples, use the standing wave tube technique to determine the normal acoustic impedance. This method is based on a mapping of the pressure maxima and minima in the standing wave pattern, and is well documented in Chapter 9 of Kinsler and Frey [13]. The newer two-microphone approach has been proven to be just as accurate as, and much faster than, the standing wave tube method in determining duct terminating impedance.

The two-microphone method is an empirical approach that allows for very rapid determination of acoustic impedance by application of Fast Fourier Transform analysis. Seybert and Ross [14] published the pioneering paper on the subject in 1977. Chung and Blaser [15,16] furthered the work by adding a third microphone to enhance the results of

low-coherence cases. Howard [17] uses the method to examine the impedance resulting from high-temperature flow in a circular duct. In addition to comparisons to the standing wave tube method, reference 17 contains extensive results of experimentally determined impedances.

The author knows of no previous attempt to estimate the acoustic boundary conditions of a combustion chamber using a finite element model in conjunction with dynamic pressure measurements.

### 1.2.3 Parameter Estimation

According to Beck and Arnold [18], formal discussions of parameter estimation were first presented by Legendre [19] and Gauss [20] in 1806 and 1809, respectively. In his work, Gauss described a least-squares method which he used to determine the orbits of planets about the sun. The method of least-squares is based on the minimization of an error function. Typically, the error function is the sum-of-squares of differences between experimentally determined and mathematically predicted values. By minimizing such a function, one can determine estimates of the unknown constants in the mathematical model. A general sum-of-squares function is given by Beck and Arnold [18] as

$$S = \sum_{i=1}^m (Y_i - \eta_i)^2 . \quad (1)$$

In this expression  $Y$  is an experimental observation,  $\eta$  is a corresponding predicted value,  $m$  is the number of observations, and  $S$  is the error function to be minimized.

Beck and Arnold go on to describe various methods of parameter estimation, including the Gauss linearization method. This method attempts to minimize the sum-of-squares function,  $S$ , by differentiating it with respect to the unknown parameters and then equating the result to zero. Modifications to the Gauss method proposed by Box and Kanemasu [21] and Bard [22] are also described by Beck and Arnold. Box and Kanemasu suggest a quadratic instead of linear local approximation of  $S$ , while Bard suggests that  $S$  should always decrease from one iteration to the next.

More recently, evolutionary algorithms, of which genetic algorithms are a subgroup, have been used for parameter estimation. Heitkoetter et al. [23] define evolutionary algorithms as “computer-based problem solving systems which use computational models of some of the known mechanisms of evolution as key elements in their design and implementation.” Heitkoetter goes on to give concise explanations of the background, theory, and common usage of the major evolutionary algorithms, including genetic algorithms, evolutionary programming, evolution strategies, classifier systems, and genetic programming.

The development and pioneering work in genetic algorithms was started by John Holland in the early 1960s at the University of Michigan. Holland’s three works in 1962 [24,25,26] represent the dawn of genetic algorithms in the United States. The next major breakthroughs for genetic algorithms came in 1975. Holland published his Adaptation in Natural and Artificial Systems [27], which is recognized as the pioneering textbook in

the field. Also, in his doctoral dissertation [28] at the University of Michigan, DeJong established the efficiency of using genetic algorithms for optimizing functions.

Today, numerous texts and technical papers may be found which describe methods for implementing genetic algorithms. Books by Goldberg [29] and Davis [30] present excellent explanations of the theory and application of genetic algorithms, as well as extensive reviews of the literature.



## **2.0 Theoretical Background**

In this chapter theoretical ideas that are fundamental to the study are introduced. These ideas include: combustion instability, acoustic boundary conditions, parameter estimation techniques, and FEM-based numerical modeling.

### **2.1 Combustion Instability**

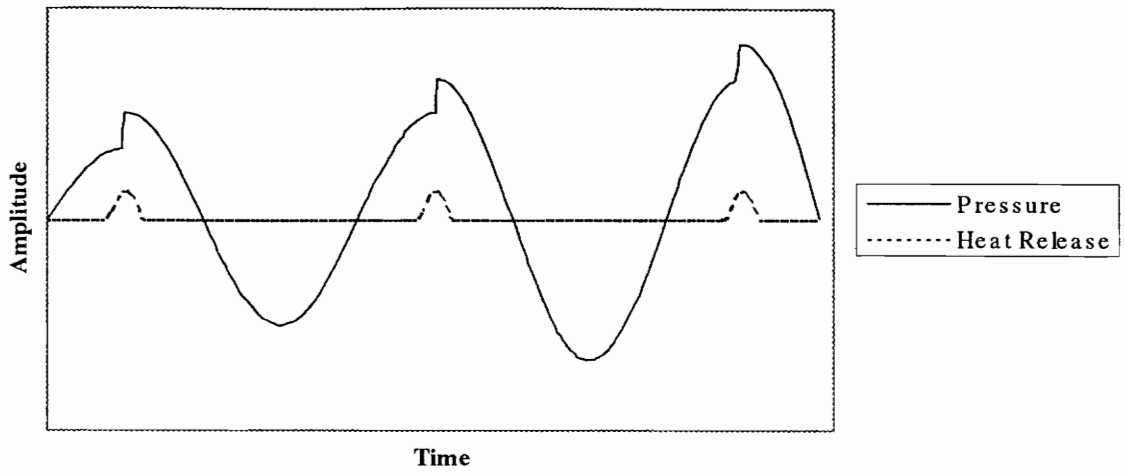
A standard for predicting fuel-feed instability has been derived by Mahan [31], and the discussion presented here closely follows his development.

A specific type of combustion instability, known as fuel-feed instability, is brought about by periodic combustion oscillations. Combustion oscillations are characterized by two separate but interrelated phenomena: the unsteady component of the burning rate of the flame, and the oscillation of pressure at the flame front. In certain circumstances, these two phenomena may be tied together in a closed-loop cause-and-effect relationship in which the pressure oscillations are both the cause of and are caused by the flame oscillations.

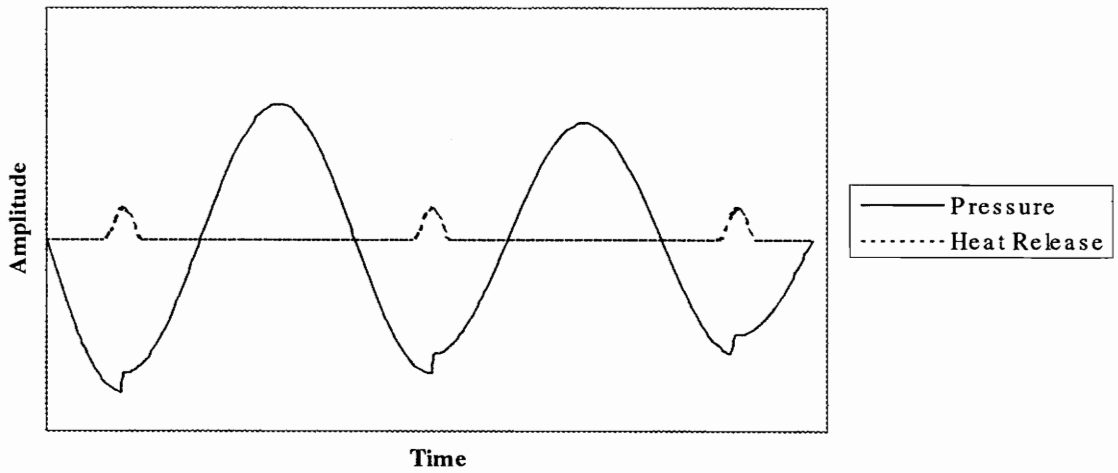
In the nineteenth century, Lord Rayleigh [1] recognized that periodic pressure oscillations could be encouraged by periodic heat release at the same frequency. This is due to the fact that heat release at constant volume leads to an increase in pressure. Furthermore, Rayleigh pointed out that the heat release in combustion chambers often tends to be periodic, or at least have a periodic component. Since the time of Rayleigh, it has been shown that this periodic heat release occurs at a broad range of frequencies in turbulent combustion. Depending on the geometry and temperature distribution of the burner, one of these frequencies may be picked out for amplification according to Rayleigh's observation.

The phase relationship between the pressure and heat release oscillations is critical to self-excited combustion oscillations. If the heat release occurs out of phase with the pressure wave (i.e. the heat release oscillation is in its positive half-cycle while the pressure oscillation is in its negative half-cycle) then energy is removed from the pressure wave. If, however, the heat release is in phase with the pressure wave (i.e. both are in their positive half-cycle during more than half of the cycle) then energy will be added to the pressure wave. These two conditions are illustrated in Figure 1. In the latter situation, the amplitude of the pressure oscillation will grow until a limit cycle is reached where further gains are negated by losses and nonlinearities.

In some cases the amplitude of the pressure fluctuations can become sufficiently large to actually modulate the fuel and/or air flow into the chamber, causing the mixture flow arriving at the flame front to oscillate between leaner and richer mixtures. If the



(a) Amplification:  $I_R = \int_0^T p(t)q(t + \tau)dt \gg 0$



(b) Damping:  $I_R = \int_0^T p(t)q(t + \tau)dt \ll 0$

**Figure 1. Graphical Representation of Rayleigh's Criterion: cases of (a) Amplification and (b) Damping.**

timing of this mixture oscillation provides a slightly richer mixture at the flame front in phase with the pressure oscillation, then the pressure amplitude is driven even higher at that frequency and the combustor is said to be unstable. It should be noted that the amplitude of the fluctuation in equivalence ratio can be as small as one-millionth or less of the steady value and still provoke an instability. It should also be emphasized that turbulent combustion provides a wide band of uncorrelated “noise” concentrated in the frequency band between 20 and 600 Hz, a frequency range that often includes the resonant modes of practical combustion chambers.

Rayleigh’s criterion can be quantified in the form of the integral

$$I_R = \int_0^T p(t)q(t + \tau)dt, \quad (2)$$

where  $p(t)$  represents the harmonic pressure oscillation,  $q(t)$  the harmonic heat release,  $T$  the common period of both, and  $\tau$  the time delay between the maximum of  $q(t)$  and the minimum of  $p(t)$ . In general, a combustion system becomes more unstable as the value of the integral becomes more positive. Therefore, a burner is inherently unstable when  $\tau = T/2$  (the value of  $I_R$  is maximized in the positive sense) and inherently stable when  $\tau = T$  (the value of  $I_R$  is maximized in the negative sense.)

For the most part, practical combustion chambers have the property that an instability is most likely to occur at the lowest acoustic mode. In long, can-type combustors, the lowest mode is usually the first axial mode. This case is presented here as an illustrative example, although the fundamental ideas apply equally well to more practical combustors and to radial and circumferential modes. The lowest axial mode for

a combustor which is “closed” at the head end and “open” at the exhaust end is the quarter-wave mode, for which

$$L_{cc} = \frac{1}{4}\lambda, \quad (3)$$

where  $L_{cc}$  is the length of the combustion chamber and  $\lambda$  is the wavelength of the standing pressure wave. This condition is depicted in Figure 2. From elementary physics, the frequency,  $f$ , is

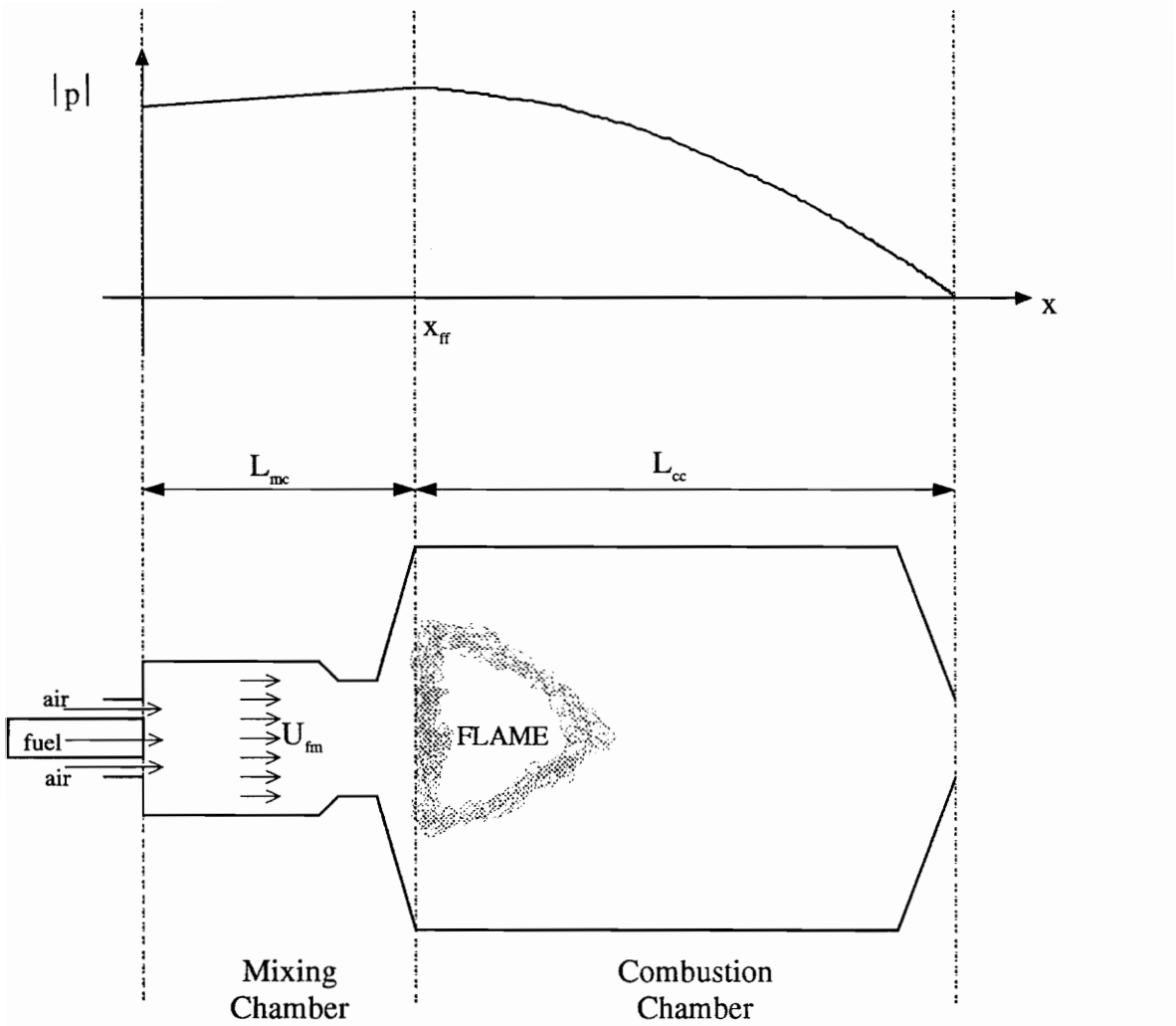
$$f = \frac{c}{\lambda} = \frac{c}{4L_{cc}}, \quad (4)$$

where  $c$  is the speed of sound in the burner. Then the period of the oscillation,  $T$ , can be expressed as

$$T = \frac{1}{f} = \frac{4L_{cc}}{c}. \quad (5)$$

The symbol  $\tau$  in Equation 2 represents the so-called “critical time delay,” which is the amount of time it takes an element of fuel mixture to be convected from the head end of the mixing chamber to the flame front. This time delay can then be expressed in terms of the effective mixing chamber length,  $L_{mc}$ , and the mean mixture convective velocity,  $U_{fm}$ , as

$$\tau = \frac{L_{mc}}{U_{fm}}. \quad (6)$$



**Figure 2. An Idealized Lean, Premixed Gas Turbine Combustion System.**

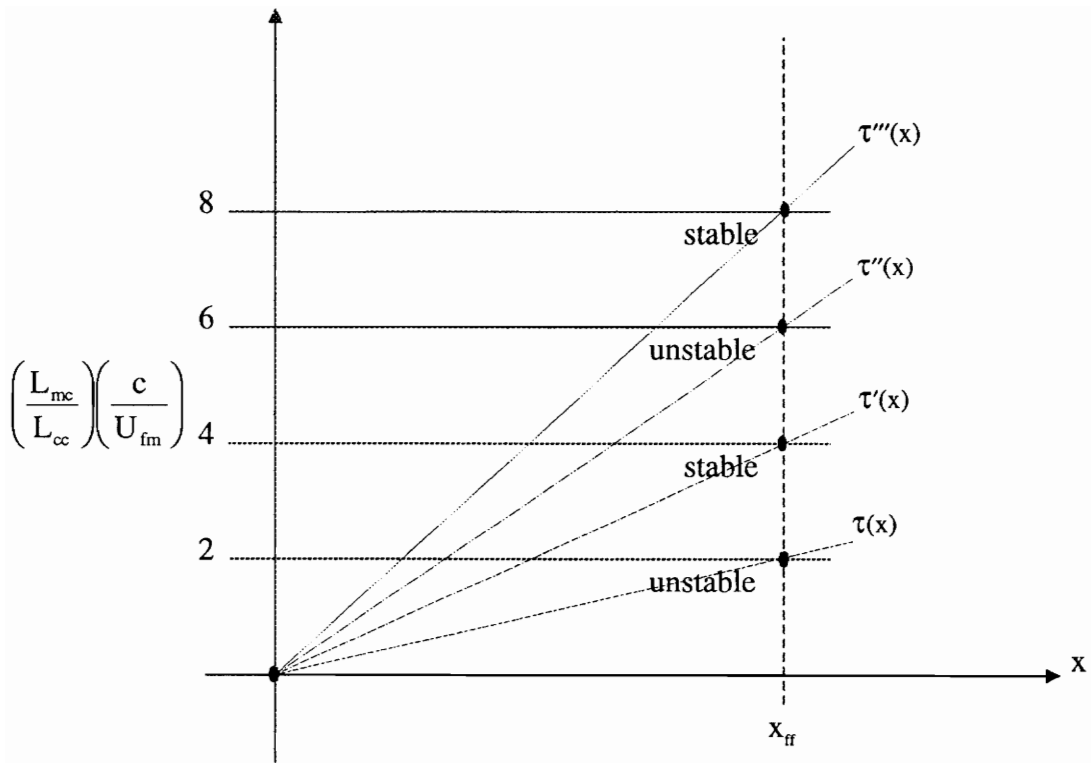
Combining the Equations 5 and 6 and the discussion of Equation 2, there results

$$\left(\frac{L_{mc}}{L_{cc}}\right)\left(\frac{c}{U_{fm}}\right) = 4 \quad \text{for stable operation, and} \quad (7)$$

$$\left(\frac{L_{mc}}{L_{cc}}\right)\left(\frac{c}{U_{fm}}\right) = 2 \quad \text{for unstable operation.} \quad (8)$$

A “stability map” can be constructed from these relations, and is shown in Figure 3. The abscissa represents the axial distance traveled from the head end of the mixing chamber, with the flame front ( $x_{ff}$ ) represented as a vertical line. The ordinate represents a nondimensional time  $(L_{mc}c)/(L_{cc}U_{fm})$ . The curves labeled with  $\tau(x)$  represent the time required by a fuel element to be convected from the head end of the mixing chamber to a given axial location,  $x$ . The stability map shows that three curves, the vertical line representing the flame front, the horizontal line representing odd-integer multiples of the first mode half-period, and the time-delay curve, must intersect to trigger a fuel-feed instability. This suggests that it is possible to avoid an instability by altering one or more of the curves.

The location of the flame front is determined by the convective velocity of the flame and the velocity of the fuel mixture; it is where these two quantities are equal, as seen in Figure 4. As the fuel mixture enters the combustion chamber (exits the mixing chamber), it decelerates to a point where its velocity is equal to that of the flame itself. As a result, the vertical line at  $x_{ff}$  on the stability map can be moved by changing either of these velocities.



$L_{cc} \equiv$  axial length of combustion chamber

$L_{mc} \equiv$  axial length of mixing chamber

$c \equiv$  thermodynamic speed of sound

$U_{fm} \equiv$  mean velocity of fuel mixture

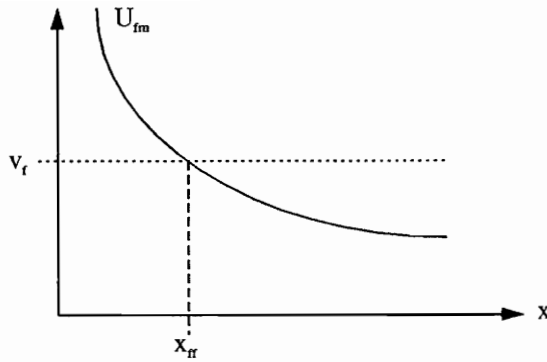
$x \equiv$  axial distance

$x_{ff} \equiv$  axial distance form fuel injector to flame front

$\tau(x) \equiv$  time for fuel mixture to travel axial distance  $x$

**Figure 3. Stability Map for Premixed, Can-Type Combustors.**





**Figure 4. Location of the Flame Front Due to Velocity Conditions**

The flame speed is a function of the fuel and the fuel temperature, and can therefore be modified by changing the fuel temperature. The velocity of the fuel mixture can be altered in a number of ways. One way is to modify the geometry of the combustion chamber to control the velocity of the fuel during its deceleration. Another is to change the velocity of the fuel before its deceleration. The velocity of the fuel mixture,  $U_{fm}$ , can be expressed as  $\dot{m}/(\rho A_c)$  where  $\dot{m}$  is the time averaged constant mass flow rate of the mix,  $\rho$  is the mean mass density, and  $A_c$  is the cross-section area of the mixing chamber. This suggest that one can alter the velocity of the fuel arriving at the flame front by changing any of these quantities.

The time delay curves represent another avenue for modifying the stability map, and can be changed in a number of ways. Since these curves represent the time it takes for the fuel mixture to travel from the injector to the flame front, it follows that their

shape is dependent upon the average velocity of the mixture and the length of the mixing chamber. Another approach concerning the time delay curves is to “spread” the fuel that is arriving at the flame front. This can be accomplished by releasing fuel at different locations, different times, or in different directions in the mixing chamber, and has the effect of “broadening” the width of the curves on the stability map.

Lastly, the stability map can be adjusted by changing the acoustics of the combustion chamber. The horizontal lines on the map represent integer multiples of the first longitudinal pressure mode half-period, and can be modified by altering this mode. One way of achieving this is to change the acoustic boundary conditions of the combustion chamber.

From the above discussion, it is seen that a number of parameters are available for controlling the stability of premixed combustors. These include (but may not be limited to): the length and diameter of the mixing chamber, the temperature of the fuel mixture, the injection conditions of the fuel, the geometry of the mouth of the mixing chamber, and the acoustics of the combustion chamber.

The above discussion can also be developed for circumferential and radial modes, and for annular-type combustors, by substituting the appropriate values.

## **2.2 Acoustic Boundary Conditions**

A traveling acoustic wave that encounters a change in transmission medium must satisfy certain conservation laws at the interface of the two media. These laws are

continuity of pressure (the pressures on both sides of the interface are equal) and continuity of normal velocity (each component of the particle velocity at the interface must add up to zero). Information governing how the wave will satisfy both continuity requirements, by reflecting some of its energy and transmitting the rest, is contained in a quantity known as the acoustic boundary condition. Precise values of boundary conditions depend on the source characteristics, geometry of the enclosure, and the external pressure field. The limiting cases of such boundary conditions depend upon the particular type of acoustic wave. For a standing wave, the limiting cases of the boundary conditions are a “hard” wall, a blocked condition which allows no transmission ( $u'=0$ ), and a “pressure release,” which allows complete transmission ( $p'=0$ ). For a traveling wave, the hard wall condition is the same, and its counterpart is a “matched” boundary.

Acoustic boundary conditions are complex-valued functions that are most commonly expressed as acoustic impedance, which is the complex ratio of complex pressure to complex particle velocity,

$$z = \frac{p}{u} . \quad (9)$$

The inverse of impedance, termed admittance, can also be used. Often times, acoustic impedance is normalized by what is known as the characteristic impedance. This characteristic impedance is the product of the mass density and the thermodynamic speed of sound of the medium,  $\rho_0 c$ , and is analogous to the index of refraction for light waves.

The normalized impedance can be expressed as

$$Z = \frac{z}{\rho_0 c} = \frac{1+r}{1-r}, \quad (10)$$

where  $r$  is the reflection coefficient calculated in the conventional standing wave tube technique, or in terms of the normalized radiation resistance and reactance,  $R$  and  $X$ ,

$$Z = R + jX. \quad (11)$$

As stated, the majority of acoustic boundary condition analyses report their results in the form of normalized impedance, either by reporting the reflection coefficient or the resistance and reactance. The current study, however, deals with the specific acoustic admittance. The specific admittance is the complex reciprocal of the specific impedance, and can be expressed in terms of  $R$  and  $X$  as

$$y = \frac{R}{\rho_0 c(R^2 + X^2)} - j \frac{X}{\rho_0 c(R^2 + X^2)}. \quad (12)$$

Expressing the real part as  $\alpha$  and the imaginary part as  $\psi$  gives  $y = \alpha - j\psi$ . (The symbol  $y$  used here represents acoustic admittance, and should not be confused with the target parameter vector  $\mathbf{Y}$  used later.) Physically,  $\alpha$  represents the dissipation of acoustic energy, and must be non-negative. The quantity  $\psi$  represents the storage of acoustic energy, and can be any real value. In the current study, a method for measuring  $\alpha$  and  $\psi$  is sought.

## 2.3 Parameter Estimation

Often times, one can create a numerical model of a physical process, but still not be able to evaluate that model because of a lack of required information. This information generally takes the form of parameters and/or boundary conditions.

Parameter estimation is a process in which values of unknown constants that appear in mathematical models are evaluated. The parameter estimation, or function optimization, problem is a type of value-based problem in which the task is to find the set(s) of parameters that maximize (or minimize) a function. In general, the parameter estimation problem can be approached in a number of ways. One may elect to search for a solution using an analytic method, a brute-force method, a calculus-based (gradient) strategy, an evolutionary strategy, or any combination of these.

For many practical problems of moderate complexity and parameter domain, analytic and brute-force methods can be quickly discarded as plausible solution techniques. Analytic techniques require extensive knowledge of the system and also require the objective function to be smooth and differentiable. Neither of these requirements are, for the most part, met by practical problems. Brute-force methods, such as enumerative or random searches, cannot efficiently cope with even a moderately sized search space. It simply takes too long to evaluate every possible solution.

This leaves the problem solver with two tools: gradient techniques and evolutionary computation. Gradient strategies are well suited for finding local extrema in an objective function, such as a sum-of-squares error function. These methods employ

the first partial derivatives of the objective function to “climb” (or “descend”) a nearby “peak” (or “valley.”) The benefit of such strategies is rapid convergence upon a local extreme.

The major drawback of gradient techniques is the inherently local scope of the search. These algorithms are extremely sensitive to starting positions, and will usually stop when anticipated improvements no longer occur. Random restart is often employed in an attempt to overcome this problem, but a large search space may cause this procedure to be inefficient or even ineffective.

Gradient techniques also depend upon the existence of derivatives. Practical optimization problems generally do not lend themselves to this notion of a smooth, differentiable objective function. Instead, users often locally linearize the model with a type of numerical approximation and differentiation, which necessarily introduces error and uncertainty into the method. And even if derivatives can be obtained, there is the possibility that the objective function may be poorly defined, or “flat,” which leads to convergence problems.

Evolutionary computation methods (or evolutionary algorithms), on the other hand, are well suited to optimization problems of a complex nature and sizable search space. Evolutionary algorithms achieve their robustness (balance between efficiency and effectiveness) by using search algorithms based on the mechanics of natural selection. More precisely, evolutionary algorithms maintain a population of individual structures that are allowed to evolve to an optimum state according to certain rules.

Evolutionary algorithms differ from traditional (analytical, enumerative, calculus-based) techniques in that they: (1) maintain and search from a population of points instead of a single point, (2) use objective function values (fitness) to guide the search instead of derivatives or other auxiliary knowledge, and (3) use probabilistic, rather than deterministic, transition rules. A key point that warrants emphasis is the fact that evolutionary algorithms do not rely on the existence of partial derivatives to complete their search. It should also be noted that evolutionary algorithms are not a random search for a solution. Randomized techniques are used, but the end result is distinctly nonrandom.

Benefits of evolutionary algorithms include computational simplicity and a very powerful, robust search tool. The prominent shortcoming is a sacrifice of peak performance for specific problems in exchange for relatively high performance throughout a large spectrum of problems. These algorithms are poorly suited for problems where efficient solution methods are already known to exist. As a result, evolutionary algorithms are generally used when all else fails.

A reasonable conclusion from these explanations of solution methods would be to use an evolutionary algorithm to determine the locale of the global minimum, and then use a gradient technique to zero in on it.

### **2.3.1 The Gauss Method**

Numerous techniques can be employed in the minimization of the objective function,  $S$ . One of the simplest and most effective is a gradient-based technique known

as the Gauss linearization, or just Gauss, method. For nonlinear models, the Gauss method is an iterative process that modifies an estimated parameter vector by specifying direction and size of corrections to that vector in each step. Redefining the sum-of-squares error function in Equation 1 yields

$$S = [\mathbf{Y} - \mathbf{P}(\boldsymbol{\beta})]^T [\mathbf{Y} - \mathbf{P}(\boldsymbol{\beta})] = \sum_{i=1}^m [Y_i - P_i(\boldsymbol{\beta})]^2, \quad (13)$$

where  $\mathbf{Y}$  and  $\mathbf{P}$  are vectors of observed and predicted values, respectively. In the current study, these values are dynamic pressure. For example,  $Y_1$  is the real component of the dynamic pressure observed at the first pressure probe,  $Y_2$  is the imaginary component of the pressure from the same probe,  $Y_3$  is the real component of the pressure from the second probe, and so on. Similarly,  $P_i$  represents the mathematically predicted value of  $Y_i$ . The  $\boldsymbol{\beta}$  term is a vector of the unknown parameters upon which  $\mathbf{P}$  is dependent. In the current research,  $\boldsymbol{\beta}$  is a vector of the four acoustic boundary conditions:

$$\begin{aligned} \beta_1 &= \text{Re}[y_z^{\text{in}}], \\ \beta_2 &= \text{Im}[y_z^{\text{in}}], \\ \beta_3 &= \text{Re}[y_z^{\text{out}}], \\ \beta_4 &= \text{Im}[y_z^{\text{out}}]. \end{aligned} \quad (14)$$

In the Gauss method, the minimization of  $S$  is achieved by setting the matrix derivative of  $S$  with respect to  $\boldsymbol{\beta}$  equal to zero, and then finding  $\boldsymbol{\beta}' = \boldsymbol{\beta}$  which best satisfies the resulting expression,

$$\nabla_{\boldsymbol{\beta}} S = 2[-\mathbf{X}^T(\boldsymbol{\beta}')] [\mathbf{Y} - \mathbf{P}(\boldsymbol{\beta}')] = 0. \quad (15)$$



In this expression,  $\mathbf{X}(\beta')$  is called the sensitivity matrix, and is defined as

$$\mathbf{X}(\beta') = [\nabla_{\beta} \mathbf{P}^T(\beta')]^T. \quad (16)$$

The sensitivity matrix is the matrix first derivative of the pressure vector with respect to the parameters being estimated. It represents the amount of change experienced by  $\mathbf{P}$  due to slight perturbations in the parameter values. Individual elements of  $\mathbf{X}$  are obtained using numerical differentiation, such as the forward difference technique,

$$X_{ij} = \frac{\partial P_i}{\partial \beta_j}, \quad (17a)$$

or

$$X_{ij} \approx \frac{P_i(b_1, \dots, b_j + \delta b_j, \dots, b_m) - P_i(b_1, \dots, b_j, \dots, b_m)}{\delta b_j}. \quad (17b)$$

The Gauss method requires that the sensitivity matrix be linearly independent, and a check must be included to verify that  $\det[\mathbf{X}^T \mathbf{X}] \neq 0$ .

Two approximations are used to solve for  $\beta'$ . First,  $\mathbf{X}(\beta')$  is replaced by  $\mathbf{X}(\mathbf{b})$ , where  $\mathbf{b}$  is an estimate of  $\beta'$ . Second,  $\mathbf{P}(\beta')$  is represented using the first two terms of a Taylor series expansion for  $\mathbf{P}(\beta')$  about  $\mathbf{b}$ . The resulting equation from these approximations,

$$\mathbf{X}^T(\mathbf{b})[\mathbf{Y} - \mathbf{P}(\mathbf{b}) - \mathbf{X}(\mathbf{b})(\beta' - \mathbf{b})] = 0, \quad (18)$$

is linear in  $\beta'$ , and can be rearranged to yield

$$\beta' = \mathbf{b} + [\mathbf{X}^T(\mathbf{b})\mathbf{X}(\mathbf{b})]^{-1}[\mathbf{X}^T(\mathbf{b})(\mathbf{Y} - \mathbf{P}(\mathbf{b}))]. \quad (19)$$

Theoretically, the above expression should give a new parameter vector,  $\beta'$ , that is a better approximation than the previous vector,  $\mathbf{b}$ , to the solution of Equation 15. A more compact notation facilitates the iterative process, and results in the Gauss linearization equation

$$\mathbf{b}^{(k+1)} = \mathbf{b}^{(k)} + [\mathbf{X}^{T(k)}\mathbf{X}^{(k)}]^{-1}[\mathbf{X}^{T(k)}(\mathbf{Y} - \mathbf{P}^{(k)})], \quad (20)$$

where  $k$  is the iteration number,  $\mathbf{b}^{(k)}$  is the  $k^{\text{th}}$  parameter estimate vector,  $\mathbf{b}^{(k+1)}$  is the  $(k+1)^{\text{th}}$  parameter estimate vector,  $\mathbf{X}^{(k)} = \mathbf{X}(\mathbf{b}^{(k)})$  is the sensitivity matrix corresponding to  $\mathbf{b}^{(k)}$ ,  $\mathbf{Y}$  is the experimentally observed pressure vector, and  $\mathbf{P}^{(k)} = \mathbf{P}(\mathbf{b}^{(k)})$  is the pressure vector corresponding to  $\mathbf{b}^{(k)}$ .

An initial estimate of the parameter vector, designated as  $\mathbf{b}^{(0)}$ , is needed to begin the iterative process. Once started, the process continues until a predetermined number of iterations is reached, or until any component of  $\mathbf{b}$  undergoes a negligible change from one estimate to the next. Beck and Arnold give a criterion for the latter as

$$\frac{|b_i^{(k+1)} - b_i^{(k)}|}{|b_i^{(k)}| + \delta_1} < \delta, \quad \text{for } i = 1, 2, \dots, m, \quad (21)$$

where  $\delta = 1 \times 10^{-4}$  and  $\delta_1 = 1 \times 10^{-10}$ .

The Gauss method is effective for both linear and nonlinear problems where the global minimum is well defined and the initial estimates are in the neighborhood of that minimum. Serious problems, i.e. nonconvergence, can be encountered, however, if these conditions are not met.

### 2.3.2 Genetic Algorithms

As stated earlier, evolutionary algorithms are search methods based on the mechanics of natural selection; a survival-of-the-fittest event in which parameters compete in the search space. Before embarking on a detailed theoretical explanation of genetic algorithms, it might be helpful to define certain terms that will be used, and how they relate to the current study. A *chromosome* is a single parameter, a single component of a  $\beta$  vector. An *individual* is a set of four chromosomes, a complete  $\beta$  vector. A *population* is a group of individuals; and a *generation* is a specific iteration step of the population. *Parents* are individuals in the current generation that *reproduce* to create a *child*, which is a new individual in the next generation.

In their basic form, evolutionary algorithms maintain a population of individuals and allow this population to evolve to an optimal state. An individual represents one possible solution to the problem, a set of parameters in the domain. Individuals are often represented using a “character string,” such as a binary number. Each individual has associated with it a “fitness,” which is a measure of the individual's ability to solve the problem. The evolution of the population is guided by rules called “search,” or “reproduction,” operators. These operators are designed with the knowledge that some individuals of a population have a better fitness than others, and that these better individuals should be more likely to survive and propagate their attractive qualities. In this way, evolutionary algorithms attempt to mimic the natural selection processes of biological systems.

The evolutionary algorithm used in the current study is a version of a genetic algorithm. Simple genetic algorithms influence the evolution of the population using three operators: reproduction, crossover, and mutation. Fitness-proportionate reproduction is a probabilistic selection process for parents of offspring in the next generation. It is based on the fitness of individuals in the parent generation, and ensures that more highly fit individuals have a higher number of offspring in the succeeding generation. Crossover is a process in which two selected parents exchange portions of their character strings to form new individuals. Mutation randomly changes the value of a bit in the character string of a new individual.

These three operators serve to perform specific functions in the evolutionary process. Any evolutionary process, whether biological or computer based, requires diversity and a selection routine. The selection requirement is fulfilled by the reproduction operator. Reproduction exploits the attractive qualities of highly fit individuals by focusing more attention on such individuals. The diversity requirement is fulfilled by the initial population and the crossover and mutation operators. A random initial population introduces a large amount of diversity at the start of the algorithm. Crossover and mutation explore new areas of the search space by perturbing the individuals in the population.

The evolutionary algorithm used in the present study is a slight deviation from a true genetic algorithm. While the base logic and operators are the same, the representation of individuals is different. Instead of using character strings to represent

individuals, the current study uses four real-valued functions. This alteration decreases the chances of finding the global minimum, but provides for very fast population convergence.

The first step in the genetic algorithm is to create an initial population. Each individual in the population consists of four real-valued variables: the real and imaginary parts of the inlet and outlet acoustic admittances. The search space is constrained in that the real part of the admittance must be non-negative. This stems from the fact that the real part of the admittance represents the dissipation of acoustic energy. Another constraint is also placed on the parameter search space, and is explained in Section 3.1.2.

Once the initial population is generated, a two-step iterative process is entered. The first step is to evaluate  $S$  for each individual in the population, as in Equation 13. This sum-of-squares error is used to determine the fitness of individuals. A highly fit individual provides a better solution (the parameter set results in a lower sum-of-squares error) than a less fit individual.

With the fitness of each individual determined, the second part of the iterative process can begin. This half consists of the three genetic operators: reproduction, crossover, and mutation. Reproduction begins by ranking the individuals according to their fitness, with a high fitness (low  $S$ ) being good and a low fitness (high  $S$ ) being bad. After the individuals are ranked each one is assigned a probability of being chosen as a parent. The probabilities are fitness-proportionate, with the sum of all equal to 1.0. For each child in the next generation two parents are selected using random numbers and the

probabilities. This process can be thought of as the spinning of a roulette wheel where space on the wheel is allocated according to fitness. Individuals of high fitness are given more space and, therefore, are more likely to be chosen as a parent. At this point, reproduction is over and crossover begins.

The crossover operation has been modified to accommodate the change in representation of individuals. Instead of each parent passing on a part of a character string to the child, the parents pass the real-valued functions. Each of the four values (chromosomes) in the offspring are derived from the chromosomes in the parents. The new chromosome can be an exact duplication of the corresponding chromosome in the first parent, it can be an exact duplication of the corresponding chromosome in the second parent, or it can be an average of the two. The three possibilities have an equal chance of occurring and are determined by a random number draw. Once the new chromosomes are determined, crossover ends and the chromosomes are checked for mutation.

Each chromosome in the offspring is checked for mutation, If mutation occurs, then a completely new value is assigned to the chromosome (in accordance with the constraints on the search space.)

The entire process, selection of parents and subsequent determination of the four offspring chromosomes, is repeated for as many children as will be in the next generation. In the current study an elitist strategy is employed in which the information in the best four individuals in the current populations passed directly to the next

generation. So with a population size of forty, thirty-six children must be inserted into the new generation along with the four best from the parent generation. (Some techniques allow for the creation of more children and then hold a fitness-based tournament to see which ones advance to the next generation.)

With the new generation complete, the fitness of each individual is once again determined and the stopping criteria are checked. If the stopping criteria are not met then the process returns to build a new generation. The stopping criteria in this case are a maximum number of generations and an acceptable lowest S value. If either of these criteria are met, then the process will not begin another iteration.

Throughout the entire process the average fitness is calculated and stored for every generation to track convergence. Also, the absolute lowest fitness and its corresponding individual parameters are stored.

## **2.4 Finite Element Model: DYNAMITE**

Parameter estimation techniques described in Section 2.3 require a mathematical model that predicts a measurable event. The simulation used in the current study is a finite-element-method (FEM)-based model that predicts the acoustic pressure field in a combustion chamber. It is called DYNAMITE and is a modification of a previously existing program called DYNAMODE, which was conceived and developed by Prof. J. R. Mahan [31] of Virginia Polytechnic Institute and State University.

### 2.4.1 Theory

Programs DYNAMODE and DYNAMITE are based on the same fundamental theory: an FEM-based model is used to provide an approximate solution to the nonhomogeneous acoustic wave equation. This equation describes the acoustic pressure field due to a point heat source distribution, and is given by [31] as

$$\frac{1}{a_0^2} \frac{\partial^2 p}{\partial t^2} - \nabla^2 p = \frac{\rho_0(\gamma - 1)}{a_0^2} \frac{\partial^2 q}{\partial t^2}. \quad (22)$$

The finite-element-method is able to provide a solution to such an equation by effectively reducing the order of the differentiation. As explained by Reddy [34], reduction of order is achieved by the introduction of a test function and the spreading of the differentiation equally over the test function and the independent variable (which, in the current study, is the acoustic pressure field,  $p(x,y,z,t)$ .) The finite-element-method allows the problem to be expressed in a matrix formulation as

$$(K_{ij} + M_{ij})P_j = F_i, \quad (23)$$

where  $P_j$  represents the acoustic pressure,  $F_i$  the heat source at the nodes, and  $(K_{ij} + M_{ij})$  represents the “stiffness” of the mesh. Once in this form, matrix inversion techniques can readily be used to solve for the pressure distribution,  $P_j$ .

The resulting solution is an approximation because the domain of the problem is discretized into a mesh of nodes and elements, and the governing equation is only solved at the nodal points. Solutions at other locations must be approximated, usually by interpolation between nodal values. A detailed derivation of the FEM model used in the current study is given in Appendix A.



It is in the stiffness matrix that the acoustic boundary conditions are found. From [31], the  $M_{ij}$  components can be expressed as

$$M_{ij} = -\omega\rho_o(\mathbf{y} \cdot \mathbf{S}_{ij}), \quad (24)$$

where

$$\mathbf{y} = \text{Re}[y_x\mathbf{n}_x + y_y\mathbf{n}_y + y_z\mathbf{n}_z] + j\text{Im}[y_x\mathbf{n}_x + y_y\mathbf{n}_y + y_z\mathbf{n}_z] \quad (25)$$

is the complex specific acoustic admittance. This admittance is assumed to be uniform over an element and is capable of having a nonzero value only at the bounding surfaces of the domain (i.e. the combustor walls and ports.) The term  $\mathbf{S}_{ij}$  in Equation 24 represents the value of volume integral in the FEM process, and has no effect on the boundary conditions.

Three significant assumptions have been placed upon the admittances in DYNAMODE and DYNAMITE for the purpose of simplifying the task at hand. The first is that the walls of the burner are “hard.” This means that the admittance in a wall element is zero, resulting in the component of the pressure gradient normal to the wall being zero. Justification for this assumption lies in the fact that the walls of the candidate burner are relatively rigid – no porous or flexible combustion liner, or similar device, that could allow for the damping of acoustic energy is present.

The second assumption is that the admittance is purely one-dimensional. This results in a non zero value in the longitudinal direction ( $y_z \neq 0$ ), and zero values in the remaining directions ( $y_x = y_y = 0$ .) This assumption is justified in that in the current

limited application only the low-frequency response of the candidate burner is being studied, and so only the longitudinal pressure modes are present.

The third, and final, assumption is that the admittance is spatially uniform over the element faces in the inlet and exhaust ports. This assumption is justified in that the range of frequencies being studied is well below the cut-on frequency for higher (non-longitudinal) modes; i.e. only longitudinal pressure wave shapes would be present even if the two admittances were allowed to have radial or circumferential distributions.

## **2.4.2 Modifications to DYNAMODE**

In the creation of DYNAMITE, two major modifications were applied to DYNAMODE: (1) the geometry of the domain was altered to represent the candidate combustor, and (2) the acoustic pressure and heat source function are treated as complex-valued rather than real-valued functions. The original geometry of DYNAMODE is that of an annular-type combustor with uniform inner and outer radii. This is changed in DYNAMITE to the METC geometry of a can-type combustor having two different outer radii. The change is accomplished by definition of the finite element mesh and its corresponding connectivity array in the subroutines COORDS and CONNECT. Worth noting is that the elements along the central axis are still six-sided brick elements, but with the length of one side collapsed to zero. This modification is suggested by Bathe [32] and is demonstrated in Figure 5.

The conversion to complex-valued variables is achieved by data-type declaration, and by recognizing and addressing a slight change in the  $M_{ij}$  term. In DYNAMODE the real component of the acoustic admittance is neglected, resulting in the expression

$$M_{ij} = -\omega\rho_o(\text{Im}[\mathbf{y}] \cdot \mathbf{S}_{ij}) , \quad (26)$$

but the complex derivation in DYNAMITE yields

$$M_{ij} = \omega\rho_o(j\mathbf{y} \cdot \mathbf{S}_{ij}). \quad (27)$$

An example illustrates the difference. If  $\mathbf{y}$  is expressed as

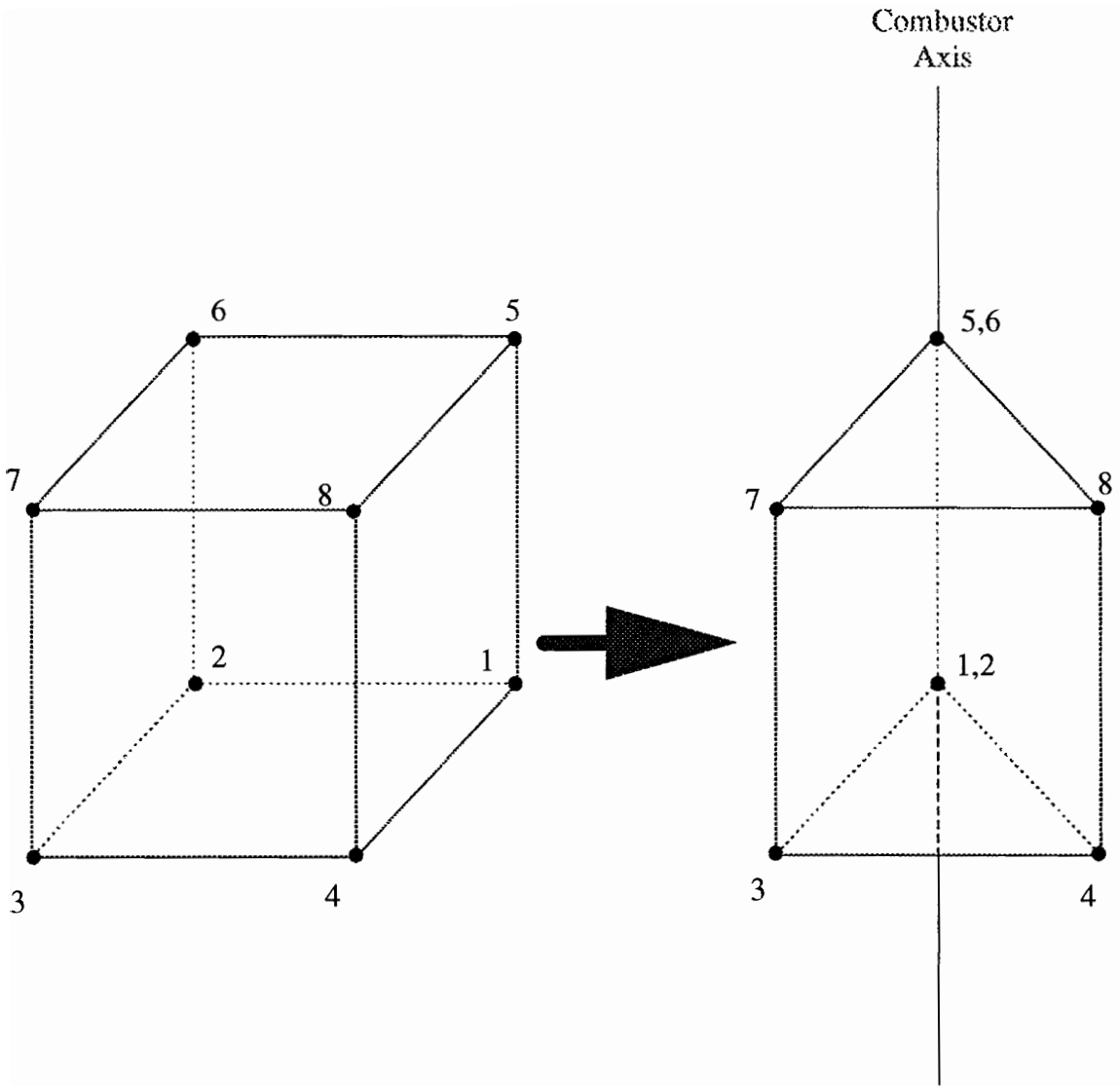
$$\mathbf{y} = \alpha + j\psi, \quad (28)$$

then  $j\mathbf{y}$  must be

$$j\mathbf{y} = -\psi + j\alpha. \quad (29)$$

This transformation must be recognized and the code changed accordingly.

These modifications allow DYNAMITE to return real and imaginary parts (or magnitude and phase angle) of the dynamic pressure at each of the specified node locations in the candidate combustor.



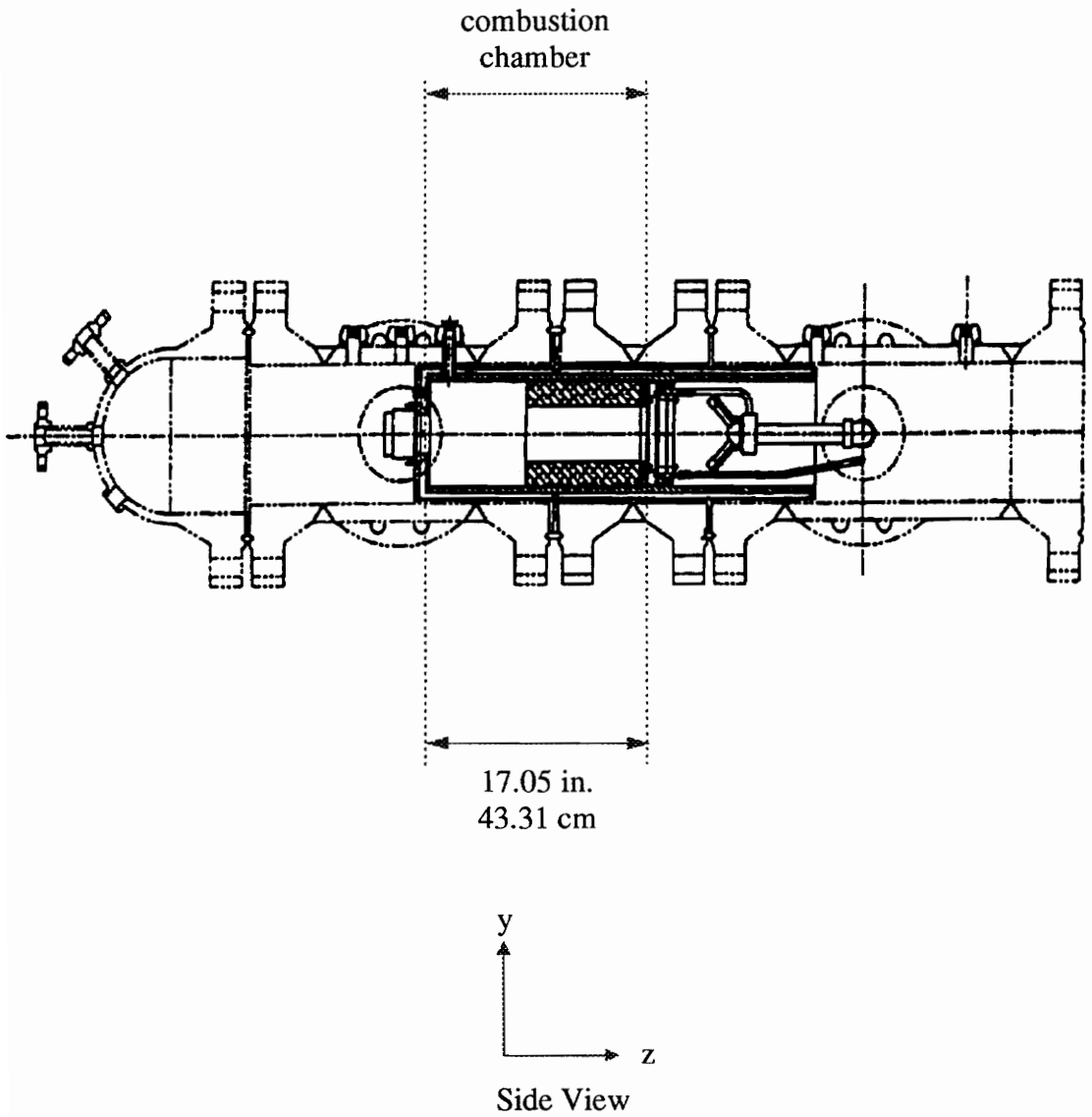
**Figure 5. Collapse of One Side of a Six-Sided Element to Form an Element Suitable for Use on the Combustor Axis.**

## 3.0 Approach

The underlying theory of the approach presented here is that experimentally observed pressures can be used in conjunction with a numerical model to determine the acoustic boundary conditions. For theoretical analyses, the pressure vector,  $\mathbf{Y}$ , is obtained by sending a realistic parameter vector to DYNAMITE. In this way the user knows the true target parameter vector,  $\beta$ , and comparisons between it and the final estimate can be made. In experimental analyses,  $\mathbf{Y}$  is obtained from a data file that contains actual measurements of pressure variations detected by probes in a candidate combustor.

### 3.1 Equipment

The equipment used in the analysis is the Program ESTIMAT and a near-scale gas-turbine combustor, shown in Figure 6. The combustor is provided and operated by the United States Department of Energy Morgantown Energy Technology Center (METC), located in Morgantown, West Virginia.

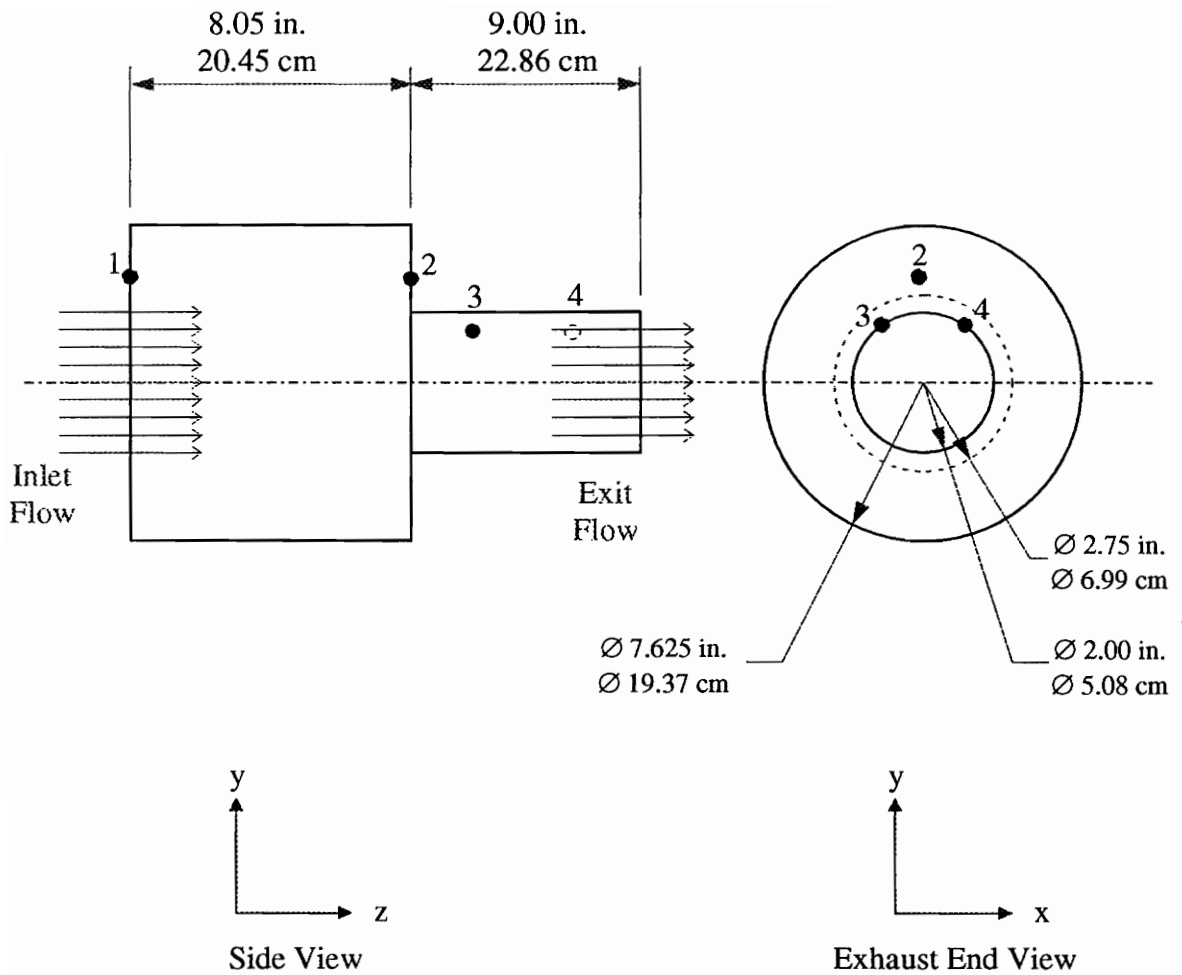


**Figure 6. METC Combustor Contained in Pressure Vessel.**

### 3.1.1 The METC Combustor

The candidate combustor, shown in Figure 5, is a near-scale, can-type burner with two distinct sections. The first section has an axial length of 8.05 in. (20.447 cm) and a constant inner radius of 7.625 in. (19.3675 cm), while the second section has an axial length of 9.0 in. (22.86 cm) and a constant inner radius of 2.0 in. (5.08 cm). The entire apparatus is constructed of stainless steel and is water-cooled. Combustion is fueled by a mixture of natural gas and air that is mixed by a swirl vane. The fuel mixture enters the burner through a 2.75 in. (6.985 cm) diameter port in the first section, and the products of combustion exit through a 2.0 in. (5.08 cm) diameter port in the second section. The mean operating pressure is 149.7 psi (1.0322 MPa), and the temperature ranges from 3680°F (2027°C) in the head end to 2690°F (1477°C) in the exhaust plane.

Four dynamic pressure probes are installed in the combustor, as shown in Figure 7. These probes return a time series of the dynamic pressure which is then converted to the frequency domain (magnitude and phase angle) via FFT. The magnitude and phase values can then be converted to the real and imaginary components of the pressure deviation above the mean value, resulting in eight independent measurements at any particular frequency. These eight measurements constitute the target vector,  $\mathbf{Y}$ , that is sent to program ESTIMAT. As explained in Section 2.3.1,  $Y_1$  is the real component of the dynamic pressure observed at the first probe,  $Y_2$  is the imaginary component of the pressure from the same probe,  $Y_3$  is the real component of the pressure from the second probe, and so on.



**Figure 7. Location of Pressure Probes in METC Combustor.**



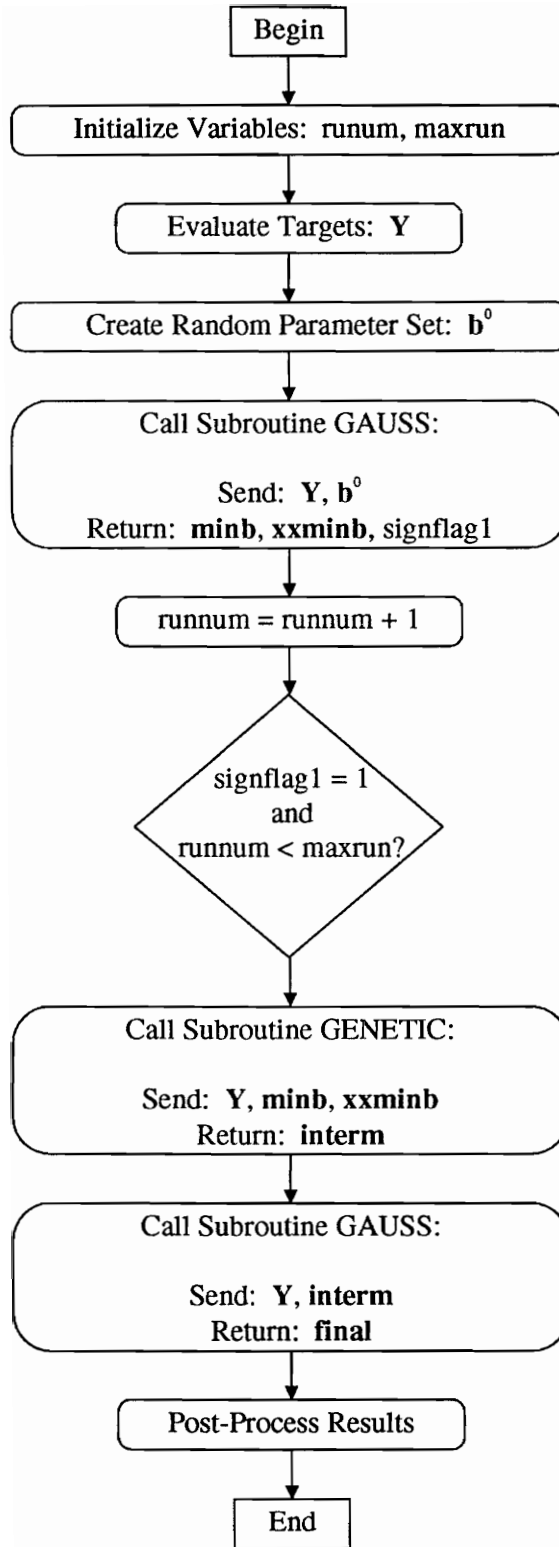
### 3.1.2 Program ESTIMAT

The estimation method developed in the current study is a FORTRAN-77 computer code called ESTIMAT. It is a combination of the gradient-based Gauss method and an evolutionary genetic algorithm. By exploiting the advantages of both techniques, it is possible for ESTIMAT to arrive at accurate estimates of the acoustic boundary conditions for nearly any candidate system. ESTIMAT uses the FEM model DYNAMITE in its numerical simulation of the dynamic behavior of the combustor.

Five major steps comprise the main body of ESTIMAT: (1) determination of the targets, (2) an initial search using the Gauss method, (3) a search using the genetic algorithm, (4) a second search with the Gauss method, and (5) post-processing of the results. These steps, along with additional logic, are shown in Figure 8.

As mentioned above, the target pressure vector,  $\mathbf{Y}$ , can be obtained theoretically or experimentally. For the current study  $\mathbf{Y}$  is a  $1 \times 8$  vector made up of four real and four imaginary components of measured pressure at a given frequency.

With the target vector determined, ESTIMAT enters a two-step loop consisting of the creation of a random starting position and the subsequent Gaussian estimation from this position. The purpose of this loop is to find the general location of the global minimum of the error function,  $S$ . The general location is defined, in this case, as an area in the search space where the sign (+/-) of every component of the estimated vector,  $\mathbf{P}$ , is the same as the sign of the corresponding component of the target vector,  $\mathbf{Y}$ . This looping process searches for such an area by comparing the signs of  $\mathbf{P}$  and  $\mathbf{Y}$  during



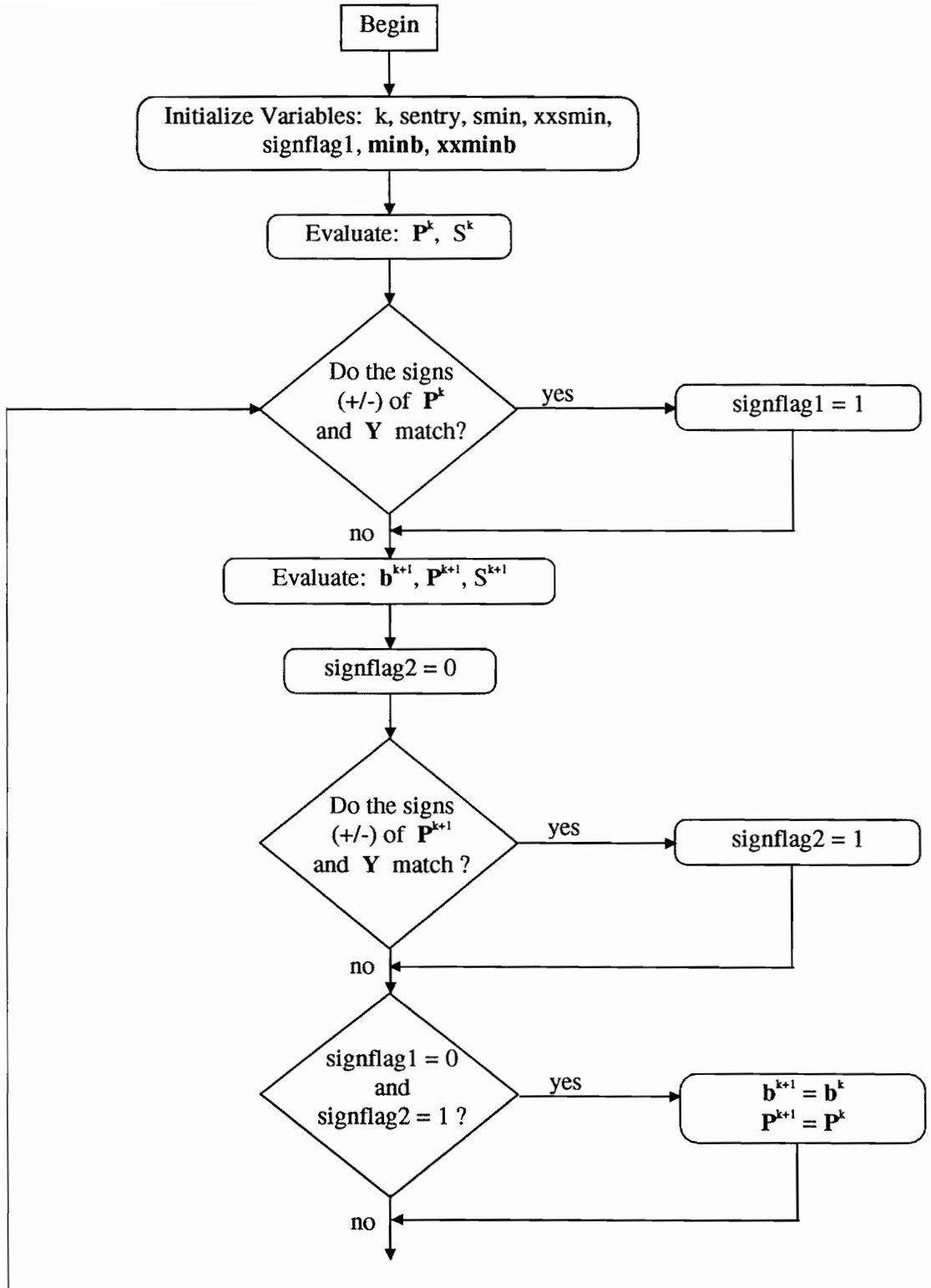
**Figure 8. Logic for Program ESTIMAT.**

Gaussian estimations from random starting positions in the search space. The loop continues until an appropriate parameter set,  $\mathbf{b}^*$ , is found, or until a preset maximum number of restarts is surpassed.

The Gauss method itself, the logic of which is shown in Figure 9, is limited to eight iterations per restart. Also, the least-square error,  $S$ , must decrease from iteration to iteration. The subroutine GAUSS terminates and a random restart occurs if either of these conditions is violated. During the execution of GAUSS the parameter set with the smallest sum-of-squares error, regardless of sign, is stored in the vector  $\mathbf{xxminb}$ . If the signs do happen to match during an iteration, then the parameter set with the smallest sum-of-squares error is stored in  $\mathbf{minb}$ . These two vectors,  $\mathbf{minb}$  and  $\mathbf{xxminb}$ , are used to prime the next estimation procedure, the genetic algorithm.

The subroutine GENETIC uses a genetic algorithm to focus more sharply on the global minimum of the objective function,  $S$ . Its probabilistic rules allow it to overcome any ill-conditioning that might cause problems for the Gauss method. The subroutine, the logic of which is shown in Figure 10, uses  $\mathbf{minb}$  and  $\mathbf{xxminb}$  as seeds for a random population, and returns an improved parameter estimate vector,  $\mathbf{interm}$ .

The variety provided by the randomized initial population is the first essential part of the genetic algorithm. GENETIC achieves its variety in the following manner. The first two individuals of the forty-member population are exact duplications of  $\mathbf{minb}$  and  $\mathbf{xxminb}$ . The next thirteen individuals are created with the constraint that each parameter must not be higher in absolute value than ten times the corresponding parameter in  $\mathbf{minb}$ .



**Figure 9. Logic for Subroutine GAUSS.**

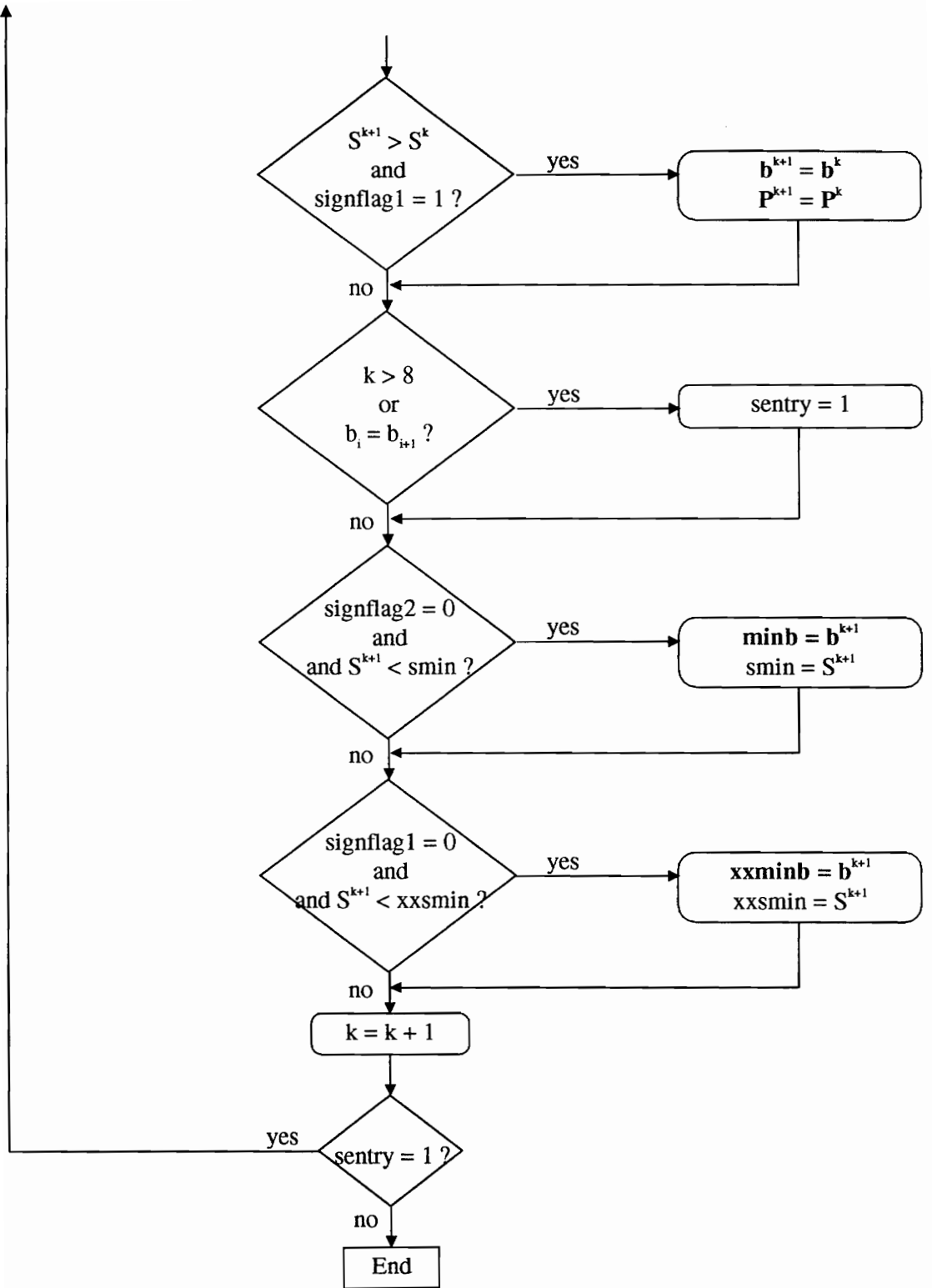
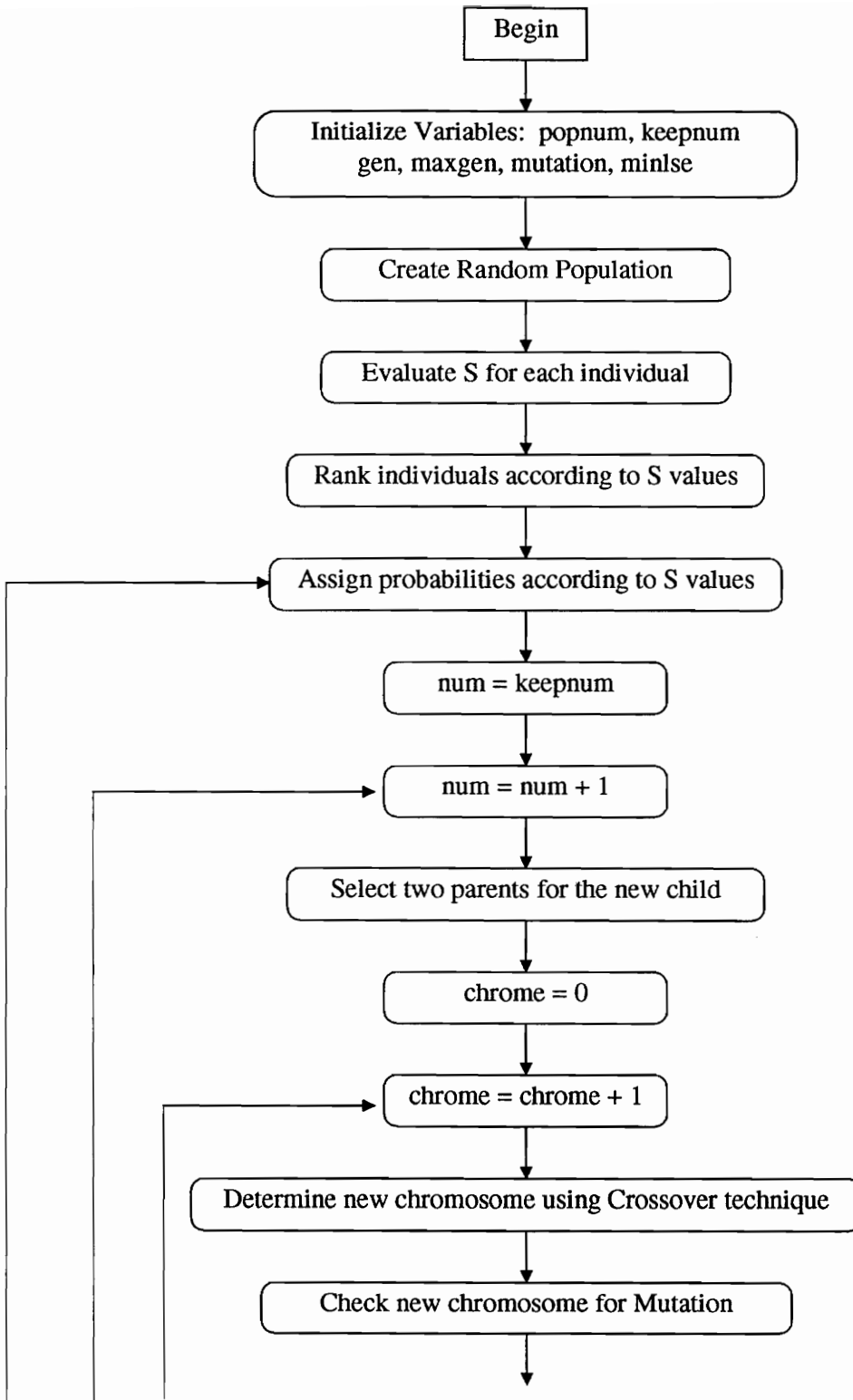
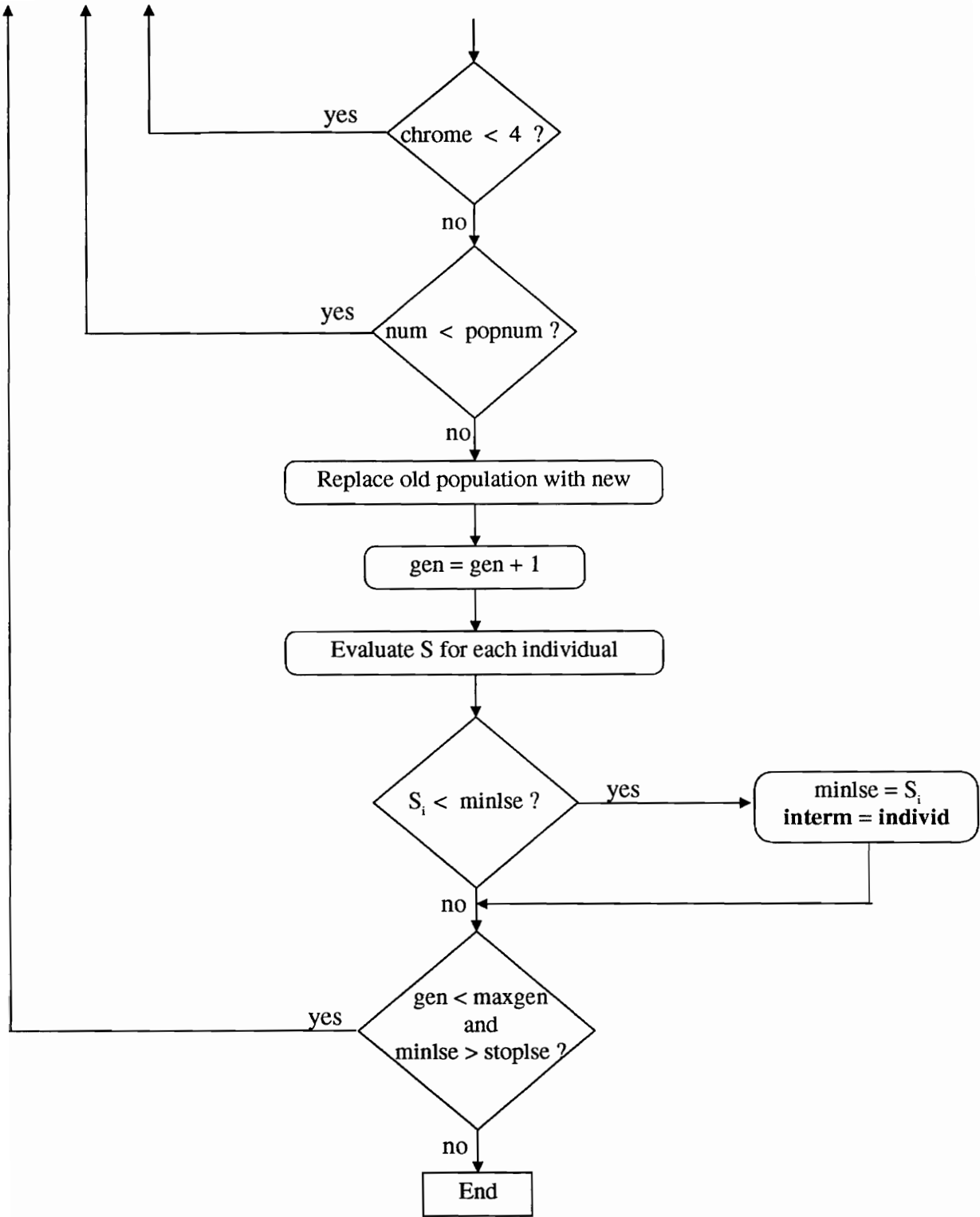


Figure 9. (continued) Logic for Subroutine GAUSS.



**Figure 10. Logic for Subroutine GENETIC.**



**Figure 10. (continued) Logic for Subroutine GENETIC.**

Individual parameters, or chromosomes, are created by drawing a random number from a uniform distribution between 0.0 and 1.0, and scaling this number according to the particular constraint. For example, if **minb** were equal to (0.979,-0.032,1.587,-6.549), then the first parameter would be constrained between 0.0 and 9.97, the second between -0.32 and 0.32, the third between 0.0 and 15.87, and the fourth between -65.49 and 65.49. (Remember that the first and third parameters represent the real part of the acoustic admittance, and must be non-negative.) Individuals sixteen through twenty-eight are determined using **xxminb** in a similar fashion. The last twelve individuals are determined by randomly multiplying or dividing a parameter from **minb** or **xxminb** by a random number between 0.0 and 1.0. It should be noted that the population size of forty and the initial distribution of 2-13-13-12 are products of the author's discretion. Both were chosen because of their apparent ability to balance effectiveness with reasonable computing time.

A scheme for fitness-proportionate reproduction is the second essential part of the genetic algorithm. This is accomplished by GENETIC with a two-step iterative process. The first step is the evaluation of the sum-of-squares error for each individual, and subsequent ranking of the population according these S values. After being ranked, the best twenty individuals of the forty-member population are selected as possible parents. The first four children of the next generation are exact duplicates of the best four parents. For each of the remaining thirty-six children, two parents are selected and genetic information is passed. Each of the four parameters in the offspring can be either an exact



duplication of the chromosome in the first parent, an exact duplication of that of the second parent, or an average of the two. Once a new chromosome is determined it is checked for mutation, which, if it occurs, randomly changes the value of the chromosome. The mutation rate in GENETIC is set at 80 occurrences out of 1000, which is considered high by genetic algorithm standards. After the new generation is complete, the process either repeats itself or terminates. The stopping criteria are a minimum sum-of-squares error ( $1.0 \times 10^{-25}$  psi<sup>2</sup> ( $4.754 \times 10^{-18}$  Pa)), or a maximum number of generations (20). When either stopping criteria is met, GENETIC ends and sends the improved estimate, **interm**, back to the main code. Once again, it should be noted that the proportions of these search parameters (number of parents to chose from, number of exact duplicates, crossover technique, mutation rate, and stopping criteria) are formed at the discretion of the author. It is quite possible that different values will work better for different conditions.

The fourth step in ESTIMAT is another Gaussian estimation, using **interm** as the starting point. This estimation is used to pinpoint the global minimum of the least-square error function, and returns the final parameter estimate, **final**. This parameter set and its corresponding pressure vector are then post-processed in the last step of ESTIMAT.

### 3.2 Analysis

The analysis in the current study is purely theoretical; at the time of this writing data is not available from the METC combustion rig. The analysis consists of

estimations based on a variety of conditions, summarized in Table 1, using two different geometries: the long, open-ended pipe used by Howard [17], and the METC burner. The nine sets of acoustic boundary conditions in Table 1 were obtained from [17], and each can be used to create a unique target pressure vector,  $\mathbf{Y}$ , using program DYNAMITE. The objective of the analysis, then, is to see if ESTIMAT can use  $\mathbf{Y}$  to recover the known parameter set which was used to create  $\mathbf{Y}$ .

To simulate the uncertainties associated with physical dynamic pressure measurements, the components of  $\mathbf{Y}$  are normally perturbed about their exact values before the estimation begins. This is accomplished using the target pressure value as the mean, an assumed standard deviation on the pressure measurement, and a uniform random number generator. The result is a random, normally distributed perturbation of each individual value in  $\mathbf{Y}$ .

For the cases involving the simple open-ended pipe geometry, uncertainties of zero, one, and five percent are applied to  $\mathbf{Y}$ . These values were chosen because a one-percent uncertainty is common in reported measurements, and zero and five percent may be considered to be limiting cases in either direction from one percent. For the cases involving the METC geometry, only uncertainties of zero and one percent are applied. The use of a five-percent uncertainty is omitted since this limiting case is studied in the straight, open-ended pipe geometry.

Therefore, each of the nine target parameter sets has five different theoretical situations associated with it: three values of uncertainty for the simple geometry, and

two values for the METC geometry. Furthermore, the admittances for each case are estimated twice, with the two estimates beginning from different starting points in the search space. The starting points are created randomly, and examining  $Y$  can provide a priori knowledge about how to construct the random bounds. For example, acoustic theory dictates that a low admittance be associated with a pressure anti-node, and high admittance with a pressure node. In the current analysis, randomly generated admittances that correspond to a pressure anti-node location are within the range of  $\pm 5 \text{ ft}^3/\text{lb}_f\cdot\text{s}$  ( $\pm 0.0318 \text{ m}^3/\text{N}\cdot\text{s}$ ); and admittances that correspond to a pressure node are in the range of  $\pm 20 \text{ ft}^3/\text{lb}_f\cdot\text{s}$  ( $\pm 0.1273 \text{ m}^3/\text{N}\cdot\text{s}$ ). These multiple estimates are performed to study the sensitivity of the process to starting position.

Overall, 45 estimates are performed based on the theoretical analysis. They are labeled as Cases 1 through 18, where Cases 1 through 9 involve the simple geometry and Cases 10 through 18 involve the METC geometry. The results of the estimates are analyzed and discussed in the next chapter.

## 4.0 Results and Discussion

In this chapter the results of the analysis are presented, explained, and discussed. The estimated complex acoustic admittance values for Cases 1 through 18 are presented in Tables 2 through 10. Two major points can be made from these results. First, the process performs very well in estimates involving zero uncertainty on the target pressure values. Of the 36 estimates of this type, 24 return admittance values that are nearly identical to the targets. More significantly, only three times out of eighteen did neither of the two estimates for a particular case return a perfect estimate. Secondly, it can be seen that the head-end admittances are consistently estimated closer to the targets than the exhaust-end admittances. This, coupled with the fact that the head end is the location of a pressure anti-node and the exhaust a pressure node, is very interesting.

Results in a different form are presented in Figures 11 through 46. These figures show the idealized target pressure fields, which are produced by the parameters in Table 1 and are not perturbed. They also show the error of the estimated pressure field relative to the targets for each theoretical case. The organization of the figures is as follows:

Figures 11 through 19. These contain the results for the simple geometry cases involving one and five percent uncertainty on the target pressures. Each figure has four subfigures (labeled a, b, c, and d). Subfigure a is the real component of the nonperturbed target pressure and subfigure b is the relative error between the target and estimated pressure fields. Subfigure b shows two curves for each of one- and five-percent uncertainty because two estimation processes are conducted for each; the two estimation processes are begun at different starting positions in the search space. Subfigures c and d are the same information for the imaginary component of the pressure.

Figures 20 through 28. These figures contain the results for the simple geometry cases involving zero uncertainty on the target pressures. The target pressure fields are the same as those in Figures 11 through 19 and, therefore, are not replotted in these figures. What is shown is the relative error curves for the real and imaginary components of pressure. These results are plotted separately from the one- and five-percent estimates due to the vast differences in magnitude of the relative error. Again, there are two curves labeled as zero-percent uncertainty because two estimation processes were conducted from different starting points.

Figures 29 through 37. These figures contain the results for the METC geometry cases involving one-percent uncertainty on the target pressures. The format is the

same as that for Figures 11 through 19, except that no five-percent uncertainty estimates are reported.

Figures 38 through 46. These figures contain the results for the METC geometry case involving zero uncertainty on the target pressures. The format is the same as that for Figures 20 through 28.

The reason for reporting the real and imaginary components of the acoustic pressure should be emphasized. Normally, results involving pressure are not presented in this fashion; however, the current study uses the two components as separate observations (or targets) in the process that estimates the complex-valued acoustic admittance. Therefore, the two components, and the relative error between the targets and estimates, are reported separately.

In the results, the relative error plots are based on how well a set of estimated parameters replicates the ideal (nonperturbed) target pressure field. The ideal pressure field, which is produced from the parameters in Table 1, is used because the objective of this part of the analysis is to see how accurately ESTIMAT can recover a target parameter set—not how closely it can match a slightly perturbed pressure field. With that in mind, the formula for determining the relative error is

$$E_i = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\% , \quad (30)$$

where  $E_i$  is the relative error, in percent, at location  $i$ ,  $Y_i$  is the target pressure at  $i$ , and  $P_i$  is the corresponding real or imaginary component of the estimated pressure field. The pressure fields for Cases 1 through 18 are sampled at [FEM] nodes along the centerline of the burner. While the estimation process is conducted using only four nodal points of observation, the results are reported for the entire length of the combustor (which consists of 11 nodes for the simple geometry and 12 for the more complicated METC geometry.)

In examining Figures 11 through 46, some observations are common to every case. First, the magnitude of the real component of the pressure is greater than that of the imaginary component, generally by two or three orders of magnitude. This implies that the complex-valued acoustic pressure is dominated by the real component. It can be seen in the results that the estimation process “recognizes” this, because the relative error of the real component is almost always less than that of the imaginary component at the same location. In other words, the estimation process pays more attention to the dominant term, i.e. the real component of the pressure. Second, the worst (highest) relative error invariably coincides with the location of pressure nodes (places where  $p' \approx 0$ ). This is to be expected, given the extremely small pressure magnitudes at these points.

Figures 11 through 19 reveal that, for the majority of the cases, one or both of the estimates involving one-percent uncertainty are better than those involving five-percent uncertainty, as seen in Figures 11, 14, and 19. Sometimes, however, the larger uncertainty estimates yield better results, as in Figure 17. This is made possible by poor

definition of the global minimum of the error function,  $S$ . Poor definition, which can be thought of as flatness in the error function, can allow a set of parameters to meet the stopping criteria of the estimation process well before the true minimum is found. This leads to a sensitivity of the process to starting position; that is, estimates which approach from different directions reach different results.

Also seen in these figures is the fact that the relative error of the real component of the pressure, for both one- and five-percent estimates, consistently lies below a value of ten percent at locations that do not coincide with a pressure node. Only two of the 36 estimates have a relative error value higher than ten percent along the entire length of the burner. These are a one-percent uncertainty estimate in Case 3 and a five-percent uncertainty estimate in Case 9, shown in Figures 13b and 19b, respectively. More significant is the fact that the real component of the estimates involving one-percent uncertainty generally have a relative error value below unity, i.e. below one percent. In all but Case 7, shown in figure 17b, at least one of these estimates returns relative error values less than unity along the entire length of the burner (except at pressure nodes).

The imaginary component of the pressure in Figures 11 through 19, being of a significantly smaller magnitude than the real component, generally has a higher relative error associated with it. While the error values of a few estimates lie entirely below one percent, as seen in Figure 11d, most of the estimates possess error values in the range of 10 to 1000 percent.



Figures 20 through 28 contain the error plots of the zero uncertainty estimates for Cases 1 through 9. Both the real and imaginary components of pressure have much lower (better) error values than those for the corresponding estimates involving one- and five-percent uncertainty. Only for one estimate of Case 8, shown in Figure 27a, did the relative error of the dominant real component remain above a value of one percent for the entire length of the burner.

Results of the estimation processes for Cases 10 through 18, which involve the METC geometry, are depicted in Figures 29 through 46. One very noticeable difference between this and the previous set of figures is in the target pressure mode shapes. Although Cases 10 through 18 use the same parameter sets as Cases 1 through 9, the METC geometry constrains the longitudinal mode shape to that of a quarter-wave in all cases, while in the simple long-pipe geometry the mode shape changes with frequency.

The most significant result seen in Figures 29 through 37 is the fact that the relative error values of the real component of estimated pressure for one-percent uncertainty estimates is generally around 0.1. The importance of this is not limited to the ability to predict the target pressure field; it also gives insight into the ability of the process to predict types of mode shapes. These results suggest that, in its current form, the estimation process is more adept at recovering parameters from lower-order mode shapes.

A final result lies in the dependence of the pressure field on the locations of (1) the pressure nodes and anti-nodes, and (2) the acoustic boundary conditions. The

pressure field is more sensitive to boundary conditions that are coincident with pressure anti-nodes, while it is less sensitive to boundary conditions that are coincident with pressure nodes. Comparing the tabulated admittance results and the plotted pressure results reveals that the relative error of the pressure field is affected more by the head-end admittance, where a pressure anti-node occurs, than by the exhaust-end admittance. Low-error cases almost always perfectly estimate the head-end admittance, while the exhaust-end admittance may be slightly erroneous. This can be seen in Case 6 in the one-percent uncertainty estimates: from Table 7, both estimates missed the exhaust-end admittances by roughly the same amount but, as seen in Figure 16b, the estimate that more closely matches the head-end admittances has a much lower relative error.

## **5.0 Conclusions and Recommendations**

### **5.1 Conclusions**

Based on the work presented in this thesis, it may be concluded that:

1. Even though the current study deals only with longitudinal mode shapes, radial and circumferential modes could also be treated with the process. The accuracy of the estimation process, however, is expected to decrease as the complexity of the pressure mode shape increases.
2. The poor definition of the global minimum of the error function can result in premature stoppage of the estimation process and, therefore, can produce poor parameter estimates.
3. The poor definition of the global minimum of the error function can result in a sensitivity of the estimation process to starting position in the search space.

4. The pressure field is less sensitive to boundary conditions that are located at pressure nodes, and more sensitive to boundary conditions that are located at pressure anti-nodes.
5. The pressure mode shape is generally not extremely sensitive to the boundary conditions; the general mode shape can be recovered even when there is a relatively large error associated with the boundary conditions.

## 5.2 Recommendations

Based on the work presented and the conclusions drawn, it is recommended that:

1. For work with higher order longitudinal, as well as with radial and circumferential pressure mode shapes, the FEM mesh should be refined and more pressure observations/targets should be used. It is recommended that more detailed studies be conducted in which the optimal mesh definition and number of observations be determined for a specific mode shape. Both specifications would necessarily depend upon the number of and spacing between the pressure maxima and minima in the standing wave pressure distribution at the particular frequency.
2. The stopping criteria of the process should be made more stringent to prevent poor estimates.

3. Multiple estimates, each with different starting positions, should be made for any single burner condition. The results of the multiple estimates (the final sum-of-squares error and the pressure mode shape) can then be compared, and the best of the group can be identified.
4. Finally, it is strongly recommended that an analysis be performed using experimental data for verification purposes. An independent measure of the boundary conditions, such as a two-microphone technique measurement, could be compared to estimated values.

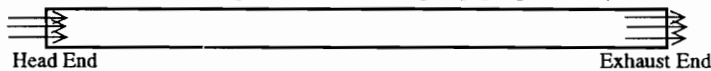
**Table 1. Parameters Used to Create Target Pressure Vectors**

	Frequency	Head-End Temperature	Exhaust-End Temperature	Head-End Admittance (Re,Im)	Exhaust-End Admittance (Re,Im)
	Hz	°F °C	°F °C	ft <sup>3</sup> /lb <sub>f</sub> ·s m <sup>3</sup> /N·s × 10 <sup>3</sup>	ft <sup>3</sup> /lb <sub>f</sub> ·s m <sup>3</sup> /N·s × 10 <sup>3</sup>
Cases 1 and 10	300	1456 791	1056 569	(0.0371, 0.0046) (0.2362, 0.0293)	(2.4212, -10.6078) (15.4131, -67.5283)
Cases 2 and 11	400	1456 791	1056 569	(0.0305, -0.0098) (0.1942, -0.0624)	(2.4581, -7.5764) (14.7877, -45.4033)
Cases 3 and 12	500	1456 791	1056 569	(0.0240, -0.0050) (0.1528, -0.0318)	(1.9279, -6.0829) (12.2728, -38.7232)
Cases 4 and 13	300	1580 860	1180 638	(0.0371, 0.0046) (0.2362, 0.0293)	(1.8658, -11.2969) (11.8775, -71.9151)
Cases 5 and 14	400	1580 860	1180 638	(0.0305, -0.0098) (0.1942, -0.0624)	(1.2631, -8.5203) (8.0410, -54.2395)
Cases 6 and 15	500	1580 860	1180 638	(0.0240, -0.0050) (0.1528, -0.0318)	(1.8874, -6.4044) (12.0150, -40.7698)
Cases 7 and 16	300	1950 1066	1550 843	(0.0371, 0.0046) (0.2362, 0.0293)	(1.5750, -14.6644) (10.0263, -93.3523)
Cases 8 and 17	400	1950 1066	1550 843	(0.0305, -0.0098) (0.1942, -0.0624)	(1.6701, -9.5784) (10.6317, -61.1026)
Cases 9 and 18	500	1950 1066	1550 843	(0.0240, -0.0050) (0.1528, -0.0318)	(1.3754, -7.1523) (8.7557, -45.5309)

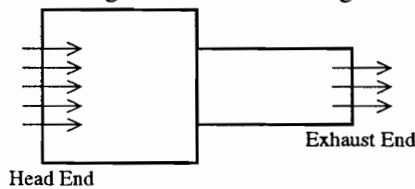
All cases use:

Heat Release = 1200.5 BTU/lb<sub>m</sub> (2792.4 kJ/kg) @ nodes 1 through 76  
 Mean Pressure = 14.699 lb/in<sup>2</sup> (1.013 × 10<sup>5</sup> Pa)

Cases 1 through 9 use the straight pipe geometry:



Cases 10 through 18 use the METC geometry:



**Table 2. Estimated Admittance Values for Cases 1 and 10.**

	Case Number	Head-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>	Exhaust-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>
<b>TARGETS</b>	1, 10	(0.0371, 0.0046) (0.2362, 0.0293)	(2.4212, -10.6078) (15.4131, -67.5283)
straight pipe, 0% uncertainty	1	(0.0371, 0.0046) (0.2362, 0.0293)	(2.4212, -10.6078) (15.4131, -67.5283)
straight pipe, 0% uncertainty	1	(0.0371, 0.0046) (0.2362, 0.0293)	(2.4241, -10.6127) (15.4316, -67.5595)
straight pipe, 1% uncertainty	1	(0.0365, 0.0014) (0.2323, 0.0088)	(1.7175, -13.3450) (10.9332, -84.9531)
straight pipe, 1% uncertainty	1	(0.0364, 0.0005) (0.2316, 0.0029)	(3.4984, -9.1452) (22.2707, -58.2175)
straight pipe, 5% uncertainty	1	(0.0342, -0.0111) (0.2180, -0.0705)	(3.7306, -29.2718) (23.7485, -186.3414)
straight pipe, 5% uncertainty	1	(0.0339, -0.0154) (0.2159, -0.0983)	(4.8305, -6.5015) (30.7508, -41.3880)
METC geometry, 0% uncertainty	10	(0.0523, 0.0042) (0.3328, 0.0265)	(4.2997, -2.7168) (27.3714, -17.2948)
METC geometry, 0% uncertainty	10	(0.0033, -0.0917) (0.0210, -0.5838)	(0.1629, 3.5783) (1.0370, 22.7788)
METC geometry, 1% uncertainty	10	(0.0350, 0.0044) (0.2229, 0.0279)	(3.1633, -22.4794) (20.1370, -143.1017)
METC geometry, 1% uncertainty	10	(0.0188, -0.0108) (0.1196, -0.0688)	(7.4691, 12.1579) (47.5475, 77.3964)

Frequency = 300 Hz  
 Head-End Temperature = 1456°F (791°C)  
 Exhaust-End Temperature = 1056°F (569°C)

**Table 3. Estimated Admittance Values for Cases 2 and 11.**

	Case Number	Head-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>F</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>	Exhaust-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>F</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>
<b>TARGETS</b>	2, 11	(0.0305, -0.0098) (0.1942, -0.0624)	(2.4581, -7.5764) (15.6480, -48.2307)
straight pipe, 0% uncertainty	2	(0.0305, -0.0098) (0.1942, -0.0624)	(2.3230, -7.1323) (14.7877, -45.4033)
straight pipe, 0% uncertainty	2	(0.0305, -0.0098) (0.1942, -0.0624)	(2.4510, -8.0158) (15.6027, -51.0280)
straight pipe, 1% uncertainty	2	(0.0313, -0.0097) (0.1990, -0.0615)	(3.5250, -5.6059) (22.4397, -35.6868)
straight pipe, 1% uncertainty	2	(0.0451, -0.0288) (0.2872, -0.1832)	(3.0575, 1.2822) (19.4637, 8.1621)
straight pipe, 5% uncertainty	2	(0.0341, -0.0092) (0.2174, -0.0586)	(3.7153, -2.9548) (23.6514, -18.8099)
straight pipe, 5% uncertainty	2	(0.0459, -0.0282) (0.2920, -0.1797)	(3.0679, 1.1982) (19.5299, 7.6276)
METC geometry, 0% uncertainty	11	(0.0305, -0.0098) (0.1942, -0.0624)	(2.4581, -7.5764) (15.6480, -48.2307)
METC geometry, 0% uncertainty	11	(0.0305, -0.0098) (0.1942, -0.0624)	(2.4588, -7.5729) (15.6524, -48.2086)
METC geometry, 1% uncertainty	11	(0.0266, -0.0138) (0.1691, -0.0881)	(3.4123, -5.9428) (21.7222, -37.8311)
METC geometry, 1% uncertainty	11	(0.0305, -0.0098) (0.1942, -0.0623)	(3.7735, -4.2710) (24.0219, -27.1885)

Frequency = 400 Hz  
 Head-End Temperature = 1456°F (791°C)  
 Exhaust-End Temperature = 1056°F (569°C)



**Table 4. Estimated Admittance Values for Cases 3 and 12.**

	Case Number	Head-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>	Exhaust-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>
<b>TARGETS</b>	3, 12	(0.0240, -0.0050) (0.1528, -0.0318)	(1.9279, -6.0829) (12.2728, -38.7232)
straight pipe, 0% uncertainty	3	(0.0240, -0.0050) (0.1528, -0.0318)	(1.9279, -6.0829) (12.2728, -38.7232)
straight pipe, 0% uncertainty	3	(0.0240, -0.0050) (0.1528, -0.0318)	(0.3754, -6.0322) (2.3898, -38.4005)
straight pipe, 1% uncertainty	3	(0.0227, -0.0048) (0.1446, -0.0308)	(0.6011, -11.4766) (3.8268, -73.0592)
straight pipe, 1% uncertainty	3	(0.0117, 0.0774) (0.0743, 0.4930)	(0.3766, 6.3521) (2.3972, 40.4369)
straight pipe, 5% uncertainty	3	(0.0236, -0.0049) (0.1503, -0.0313)	(1.9443, -5.7369) (12.3774, -36.5203)
straight pipe, 5% uncertainty	3	(0.0442, -0.0144) (0.2813, -0.0918)	(4.3383, -1.6605) (27.6172, -10.5705)
METC geometry, 0% uncertainty	12	(0.0240, -0.0050) (0.1528, -0.0318)	(1.9279, -6.0829) (12.2728, -38.7232)
METC geometry, 0% uncertainty	12	(0.0240, -0.0050) (0.1528, -0.0318)	(1.9279, -6.0829) (12.2729, -38.7232)
METC geometry, 1% uncertainty	12	(0.0238, -0.0054) (0.1517, -0.0342)	(2.9899, -3.7026) (19.0334, -23.5705)
METC geometry, 1% uncertainty	12	(0.0244, -0.0042) (0.1554, -0.0270)	(2.9491, -3.8930) (18.7738, -24.7824)

Frequency = 500 Hz  
 Head-End Temperature = 1456°F (791°C)  
 Exhaust-End Temperature = 1056°F (569°C)

**Table 5. Estimated Admittance Values for Cases 4 and 13.**

	Case Number	Head-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>	Exhaust-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>
<b>TARGETS</b>	4, 13	(0.0371, 0.0046) (0.2362, 0.0293)	(1.8658, -11.2969) (11.8775, -71.9151)
straight pipe, 0% uncertainty	4	(0.0371, 0.0046) (0.2362, 0.0293)	(1.8658, -11.2969) (11.8775, -71.9151)
straight pipe, 0% uncertainty	4	(0.0371, 0.0046) (0.2362, 0.0293)	(1.8658, -11.2969) (11.8775, -71.9151)
straight pipe, 1% uncertainty	4	(0.0375, 0.0049) (0.2384, 0.0314)	(1.0290, -17.9906) (6.5502, -114.5265)
straight pipe, 1% uncertainty	4	(0.0339, 0.0008) (0.2158, 0.0049)	(4.7972, -7.8649) (30.5385, -50.0671)
straight pipe, 5% uncertainty	4	(0.0388, 0.0063) (0.2472, 0.0399)	(1.7415, -60.4562) (11.0861, -384.8588)
straight pipe, 5% uncertainty	4	(0.0290, -0.0056) (0.1848, -0.0354)	(6.7741, -9.6224) (43.1234, -61.2554)
METC geometry, 0% uncertainty	13	(0.0371, 0.0046) (0.2362, 0.0293)	(1.8546, -11.2954) (11.8063, -71.9053)
METC geometry, 0% uncertainty	13	(0.0230, -0.0183) (0.1465, -0.1162)	(10.3532, 10.2514) (65.9077, 65.2593)
METC geometry, 1% uncertainty	13	(0.0447, -0.0050) (0.2848, -0.0318)	(9.6326, -2.8089) (61.3201, -17.8814)
METC geometry, 1% uncertainty	13	(0.0079, -0.0359) (0.0502, -0.2286)	(3.8613, 12.3177) (24.5805, 78.4134)

Frequency = 300 Hz  
 Head-End Temperature = 1580°F (860°C)  
 Exhaust-End Temperature = 1180°F (638°C)

**Table 6. Estimated Admittance Values for Cases 5 and 14.**

	Case Number	Head-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>	Exhaust-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>
<b>TARGETS</b>	5, 14	(0.0305, -0.0098) (0.1942, -0.0624)	(1.2631, -8.5203) (8.0410, -54.2395)
straight pipe, 0% uncertainty	5	(0.0305, -0.0098) (0.1942, -0.0624)	(1.2631, -8.5203) (8.0410, -54.2395)
straight pipe, 0% uncertainty	5	(0.0305, -0.0098) (0.1942, -0.0624)	(1.2618, -8.5191) (8.0326, -54.2320)
straight pipe, 1% uncertainty	5	(0.0152, -0.0331) (0.0970, -0.2104)	(3.1129, 6.1549) (19.8167, 39.1816)
straight pipe, 1% uncertainty	5	(0.0193, -0.0029) (0.1226, -0.0185)	(1.5477, 18.6889) (9.8528, 118.9716)
straight pipe, 5% uncertainty	5	(0.0152, -0.0331) (0.0970, -0.2104)	(7.9876, 4.9462) (50.8482, 31.4870)
straight pipe, 5% uncertainty	5	(0.0593, -0.0217) (0.3778, -0.1383)	(2.1065, -0.1768) (13.4100, -1.1255)
METC geometry, 0% uncertainty	14	(0.0306, -0.0104) (0.1948, -0.0664)	(3.7389, -23.6019) (23.8013, -150.2478)
METC geometry, 0% uncertainty	14	(0.0305, -0.0098) (0.1942, -0.0624)	(1.1345, -8.5084) (7.2219, -54.1638)
METC geometry, 1% uncertainty	14	(0.0261, -0.0144) (0.1661, -0.0914)	(5.2912, -38.2223) (33.6834, -243.3196)
METC geometry, 1% uncertainty	14	(0.0169, 0.0459) (0.1075, 0.2920)	(0.9253, -2.9286) (5.8903, -18.6432)

Frequency = 400 Hz  
 Head-End Temperature = 1580°F (860°C)  
 Exhaust-End Temperature = 1180°F (638°C)

**Table 7. Estimated Admittance Values for Cases 6 and 15.**

	Case Number	Head-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>	Exhaust-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>
<b>TARGETS</b>	6, 15	(0.0240, -0.0050) (0.1528, -0.0318)	(1.8874, -6.4044) (12.0150, -40.7698)
straight pipe, 0% uncertainty	6	(0.0240, -0.0050) (0.1528, -0.0318)	(1.8892, -6.3836) (12.0265, -40.6372)
straight pipe, 0% uncertainty	6	(0.0240, -0.0050) (0.1528, -0.0318)	(1.8934, -6.3931) (12.0534, -40.6977)
straight pipe, 1% uncertainty	6	(0.0090, 0.0673) (0.0575, 0.4282)	(0.1207, -0.9947) (0.7682, -6.3319)
straight pipe, 1% uncertainty	6	(0.0229, -0.0054) (0.1455, -0.0342)	(1.2243, -11.0995) (7.7935, -70.6586)
straight pipe, 5% uncertainty	6	(0.0032, -0.0058) (0.0205, -0.0370)	(0.2268, -2.4129) (1.4441, -15.3601)
straight pipe, 5% uncertainty	6	(0.0032, -0.0058) (0.0205, -0.0370)	(0.2268, -2.4129) (1.4441, -15.3601)
METC geometry, 0% uncertainty	15	(0.0240, -0.0050) (0.1528, -0.0318)	(1.8874, -6.4044) (12.0150, -40.7698)
METC geometry, 0% uncertainty	15	(0.0240, -0.0050) (0.1528, -0.0318)	(1.8874, -6.4044) (12.0150, -40.7698)
METC geometry, 1% uncertainty	15	(0.0272, -0.0011) (0.1728, -0.0071)	(30.0516, 77.7716) (191.3057, 495.0868)
METC geometry, 1% uncertainty	15	(0.0227, -0.0139) (0.1448, -0.0885)	(2.3373, -16.0055) (14.8789, -101.8895)

Frequency = 500 Hz  
 Head-End Temperature = 1580°F (860°C)  
 Exhaust-End Temperature = 1180°F (638°C)

**Table 8. Estimated Admittance Values for Cases 7 and 16.**

	Case Number	Head-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>F</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>	Exhaust-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>F</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>
<b>TARGETS</b>	7, 16	(0.0371, 0.0046) (0.2362, 0.0293)	(1.5750, -14.6644) (10.0263, -93.3523)
straight pipe, 0% uncertainty	7	(0.0371, 0.0046) (0.2362, 0.0293)	(1.5699, -14.6913) (9.9936, -93.5235)
straight pipe, 0% uncertainty	7	(0.0371, 0.0046) (0.2362, 0.0293)	(1.5750, -14.6644) (10.0263, -93.3523)
straight pipe, 1% uncertainty	7	(0.0024, 0.1217) (0.0152, 0.7747)	(0.2423, 18.5800) (1.5422, 118.2788)
straight pipe, 1% uncertainty	7	(0.0384, 0.2154) (0.2444, 1.3709)	(3.2195, -2.9630) (20.4951, -18.8624)
straight pipe, 5% uncertainty	7	(0.0024, 0.1217) (0.0152, 0.7747)	(0.2985, 12.7158) (1.9000, 80.9474)
straight pipe, 5% uncertainty	7	(0.0525, -0.0193) (0.3345, -0.1228)	(5.0673, -18.5129) (32.2581, -117.8517)
METC geometry, 0% uncertainty	16	(0.0080, 0.0054) (0.0511, 0.0343)	(1.5241, 75.1735) (9.7024, 478.5476)
METC geometry, 0% uncertainty	16	(0.0049, 0.0172) (0.0314, 0.1097)	(1.1417, 20.5133) (7.2680, 130.5856)
METC geometry, 1% uncertainty	16	(0.0167, 0.0216) (0.1064, 0.1375)	(42.2469, 15.1175) (268.9400, 96.2368)
METC geometry, 1% uncertainty	16	(0.0136, 0.0299) (0.0866, 0.1902)	(0.4566, -5.0021) (2.9064, -31.8431)

Frequency = 300 Hz  
 Head-End Temperature = 1950°F (1066°C)  
 Exhaust-End Temperature = 1550°F (843°C)

**Table 9. Estimated Admittance Values for Cases 8 and 17.**

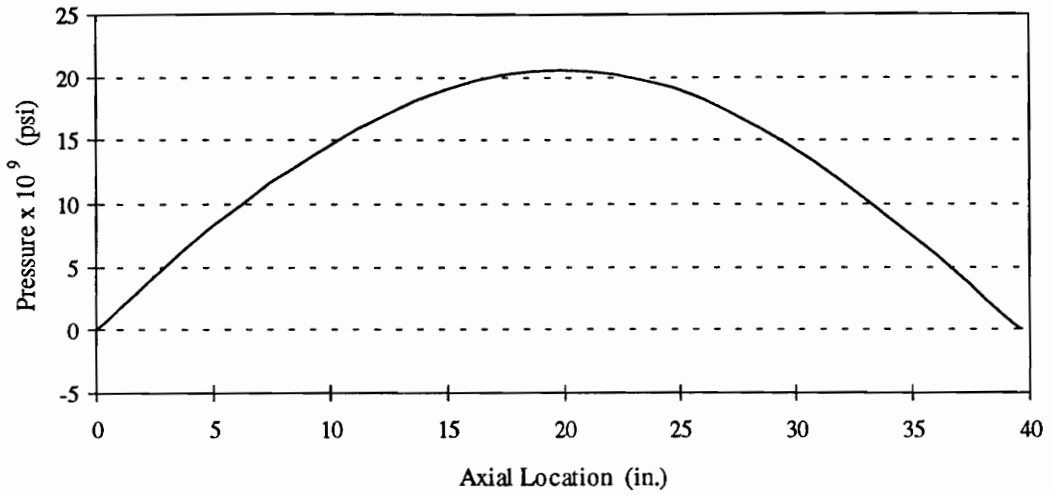
	Case Number	Head-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>	Exhaust-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>r</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>
<b>TARGETS</b>	8, 17	(0.0305, -0.0098) (0.1942, -0.0624)	(1.6701, -9.5984) (10.6317, -61.1026)
straight pipe, 0% uncertainty	8	(0.0007, 0.0508) (0.0046, 0.3231)	(2.1330, 6.4384) (13.5782, 40.9864)
straight pipe, 0% uncertainty	8	(0.0305, -0.0098) (0.1942, -0.0624)	(7.6400, -32.5993) (48.6354, -207.5240)
straight pipe, 1% uncertainty	8	(0.0050, 0.0465) (0.0319, 0.2961)	(1.3072, 7.5415) (8.3212, 48.0083)
straight pipe, 1% uncertainty	8	(0.0304, -0.0098) (0.1937, -0.0622)	(4.4245, -6.8669) (28.1663, -43.7142)
straight pipe, 5% uncertainty	8	(0.0323, -0.0046) (0.2056, -0.0292)	(5.6543, 1.9243) (35.9949, 12.2497)
straight pipe, 5% uncertainty	8	(0.0301, -0.0097) (0.1917, -0.0616)	(4.5836, -3.2892) (29.1787, -20.9391)
METC geometry, 0% uncertainty	17	(0.0306, -0.0097) (0.1948, -0.0615)	(47.9336, 9.0932) (305.1409, 57.8867)
METC geometry, 0% uncertainty	17	(0.0305, -0.0098) (0.1942, -0.0624)	(1.6458, -9.8193) (10.4771, -62.5091)
METC geometry, 1% uncertainty	17	(0.0324, -0.0108) (0.2065, -0.0690)	(4.8360, -12.6856) (30.7853, -80.7554)
METC geometry, 1% uncertainty	17	(0.0290, -0.0110) (0.1849, -0.0703)	(3.7449, -8.1358) (23.8397, -51.7915)

Frequency = 400 Hz  
 Head-End Temperature = 1950°F (1066°C)  
 Exhaust-End Temperature = 1550°F (843°C)

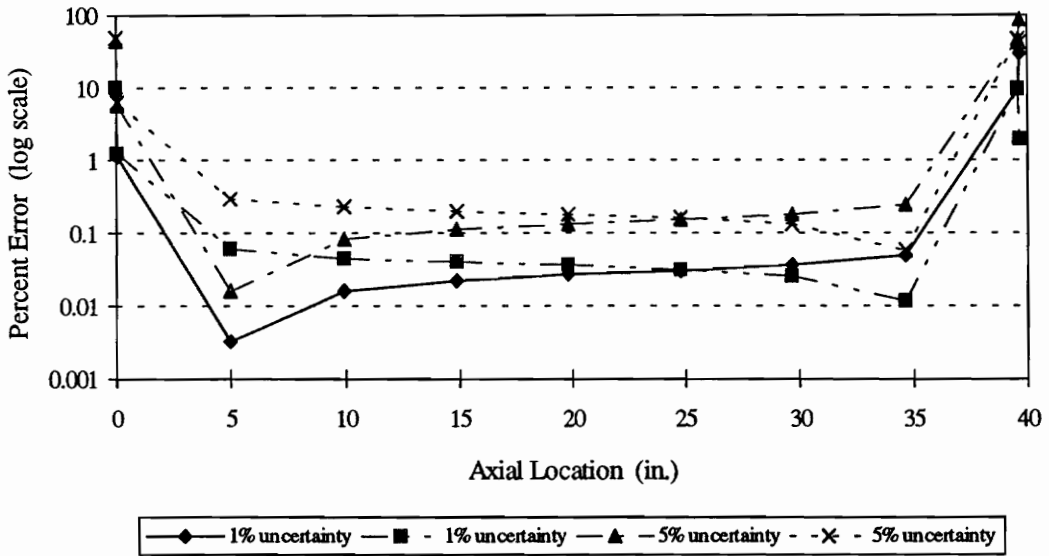
**Table 10. Estimated Admittance Values for Cases 9 and 18.**

	Case Number	Head-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>F</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>	Exhaust-End Admittance (Re, Im) ft <sup>3</sup> /lb <sub>F</sub> s m <sup>3</sup> /N·s × 10 <sup>3</sup>
<b>TARGETS</b>	<b>9, 18</b>	<b>(0.0240, -0.0050)</b> <b>(0.1528, -0.0318)</b>	<b>(1.3754, -7.1523)</b> <b>(8.7557, -45.5309)</b>
straight pipe, 0% uncertainty	9	(0.0240, -0.0049) (0.1526, -0.0314)	(3.2207, -6.0660) (20.5029, -38.6157)
straight pipe, 0% uncertainty	9	(0.0240, -0.0050) (0.1528, -0.0318)	(2.0490, -6.8244) (13.0438, -43.4437)
straight pipe, 1% uncertainty	9	(0.0264, -0.0046) (0.1680, -0.0294)	(3.8159, -4.2352) (24.2915, -26.9609)
straight pipe, 1% uncertainty	9	(0.0237, -0.0051) (0.1507, -0.0323)	(0.0015, -12.9565) (0.0096, -82.4802)
straight pipe, 5% uncertainty	9	(0.0363, -0.0028) (0.2312, -0.0176)	(2.6566, -3.9334) (16.9114, -25.0395)
straight pipe, 5% uncertainty	9	(0.5907, 0.0014) (3.7606, 0.0090)	(5.3695, -3.8539) (34.1819, -24.5335)
METC geometry, 0% uncertainty	18	(0.0240, -0.0050) (0.1528, -0.0318)	(1.3754, -7.1523) (8.7557, -45.5309)
METC geometry, 0% uncertainty	18	(0.0240, -0.0050) (0.1528, -0.0318)	(1.3754, -7.1523) (8.7557, -45.5309)
METC geometry, 1% uncertainty	18	(0.0259, -0.0026) (0.1647, -0.0165)	(0.0000, -12.1331) (0.0002, -77.2385)
METC geometry, 1% uncertainty	18	(0.0238, -0.0047) (0.1515, -0.0302)	(1.5230, -18.8316) (9.6954, -119.8805)

Frequency = 500 Hz  
 Head-End Temperature = 1950°F (1066°C)  
 Exhaust-End Temperature = 1550°F (843°C)



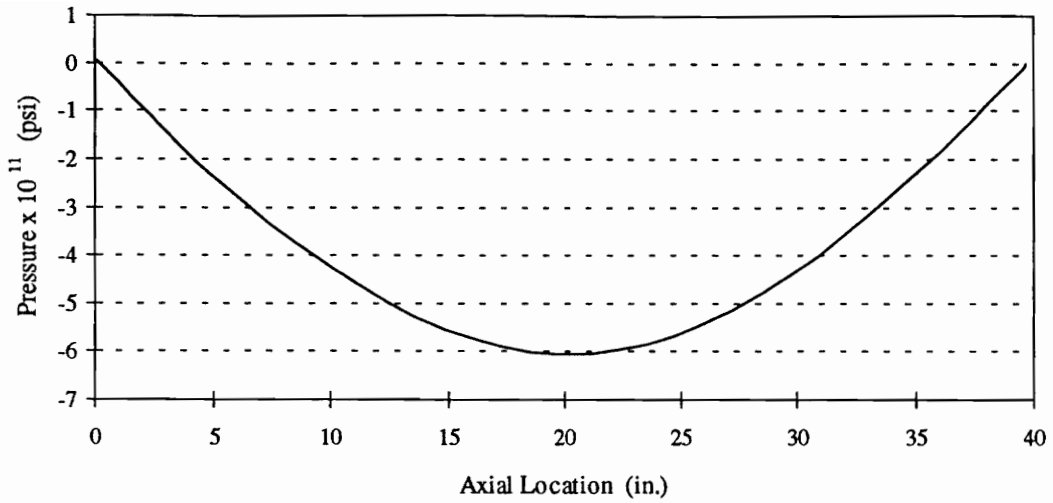
**Figure 11a. The Real Component of the Target Pressure, Case 1.**



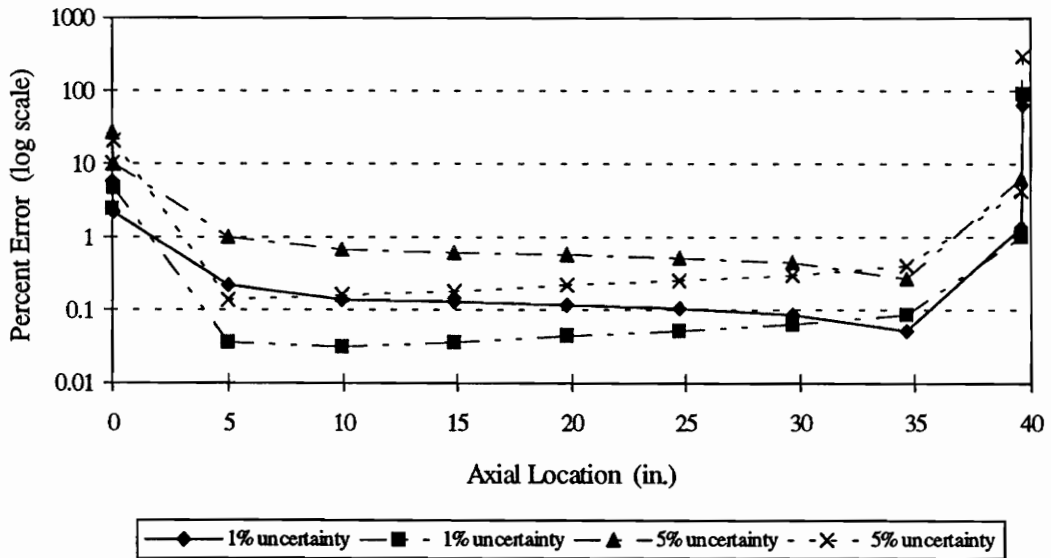
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 11b. Percent Error of the Real Component of the Estimated Pressure, Case 1.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)



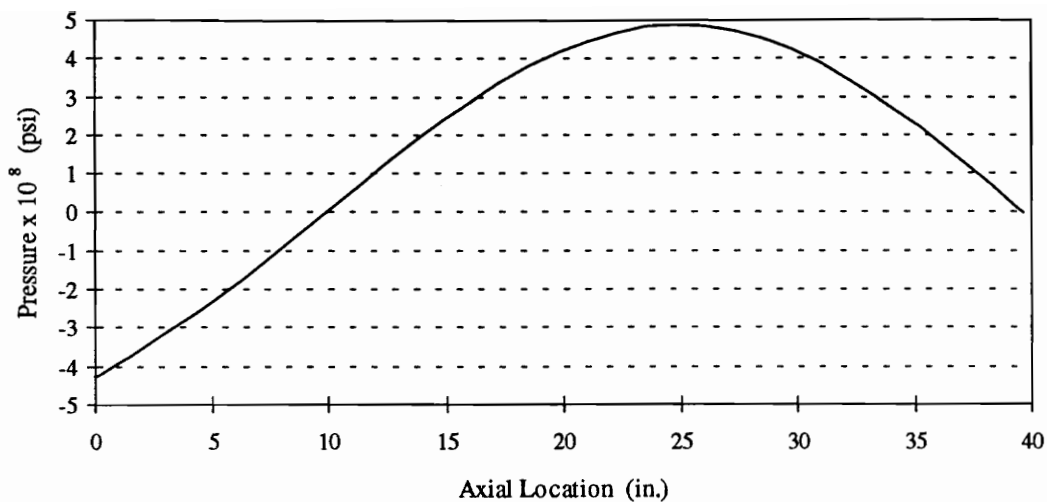


**Figure 11c. The Imaginary Component of the Target Pressure, Case 1.**

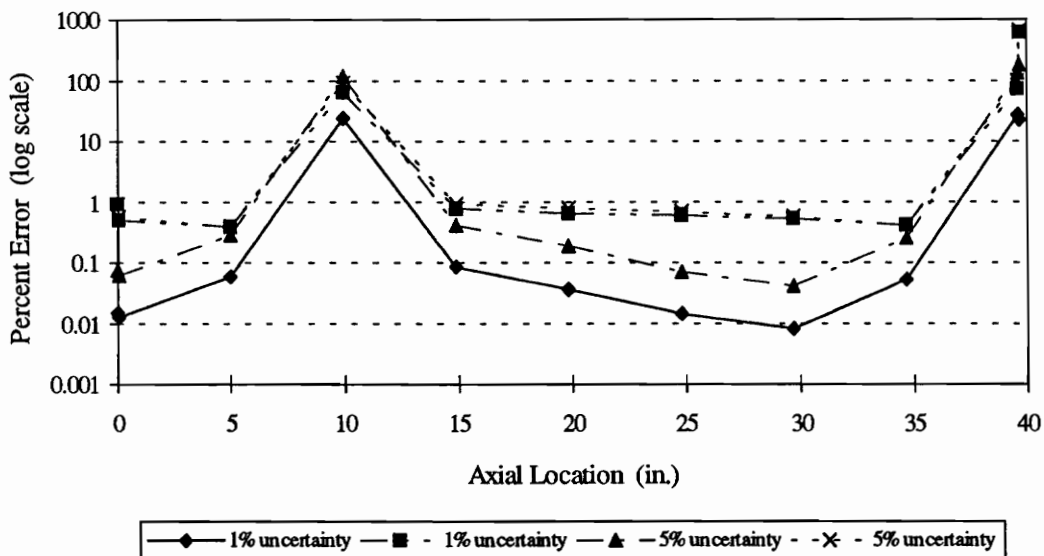


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 11d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 1.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

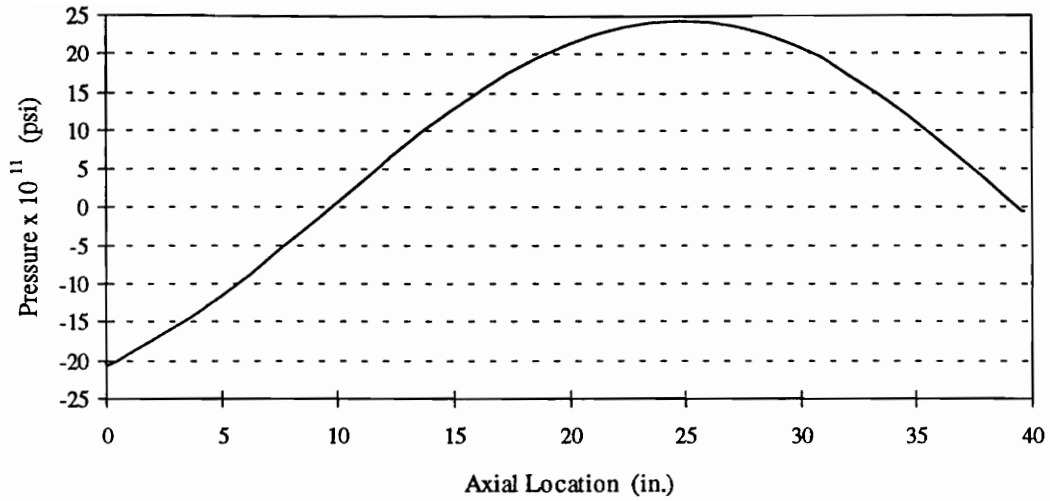


**Figure 12a. The Real Component of the Target Pressure, Case 2.**

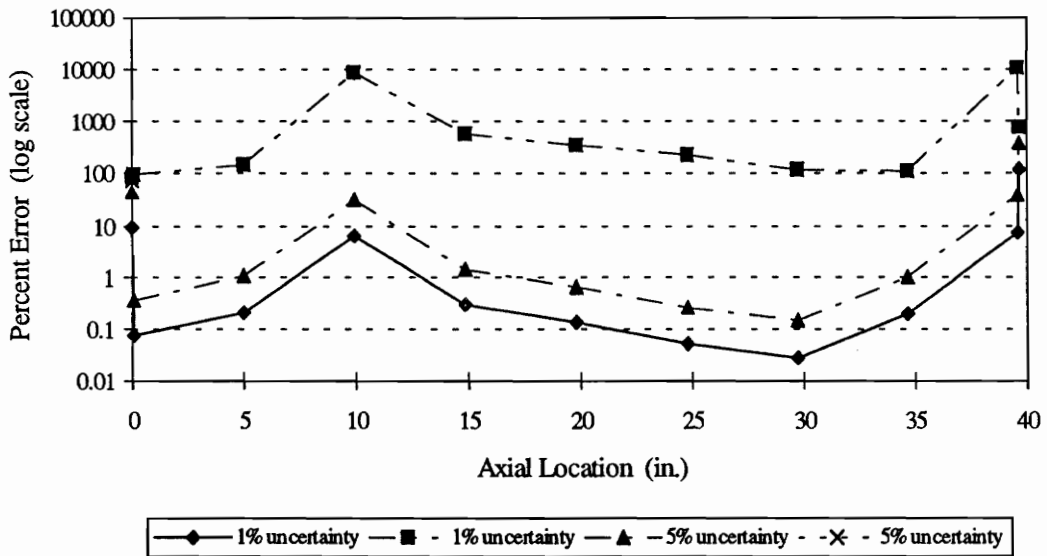


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 12b. Percent Error of the Real Component of the Estimated Pressure, Case 2.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

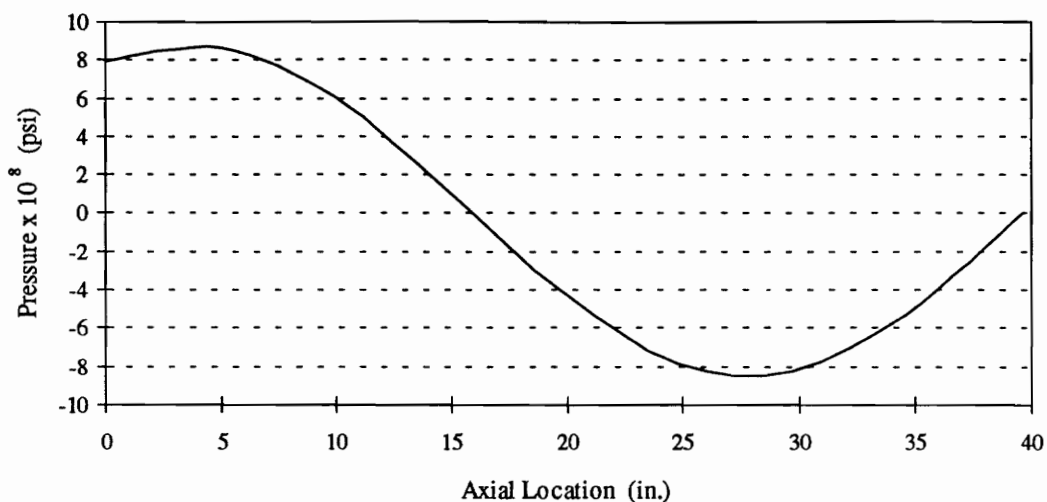


**Figure 12c. The Imaginary Component of the Target Pressure, Case 2.**

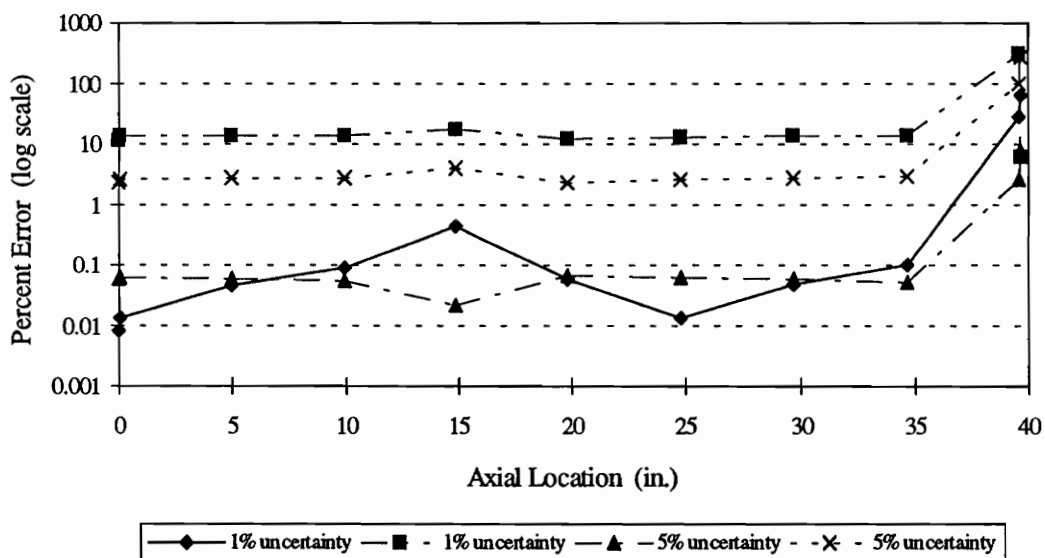


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 12d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 2.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

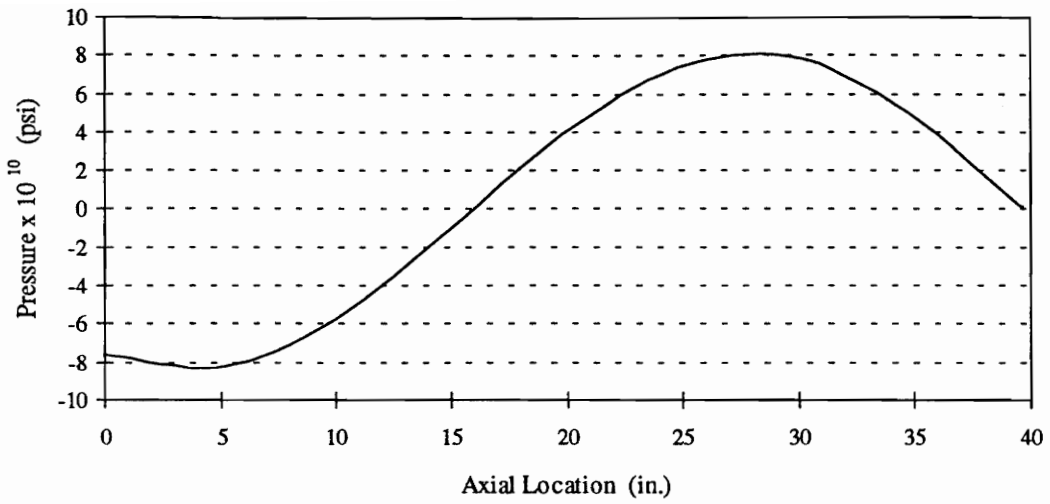


**Figure 13a. The Real Component of the Target Pressure, Case 3.**

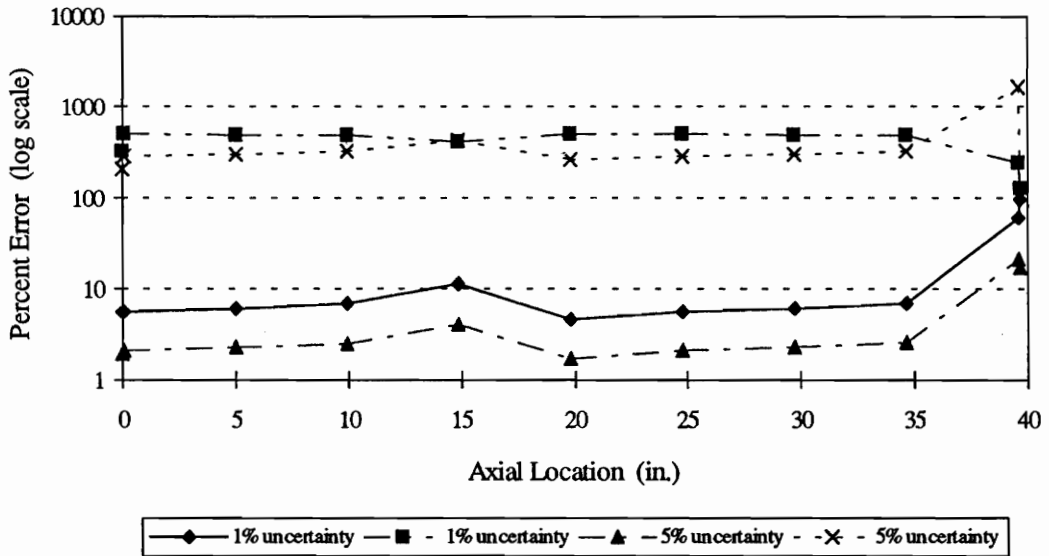


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 13b. Percent Error of the Real Component of the Estimated Pressure, Case 3.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

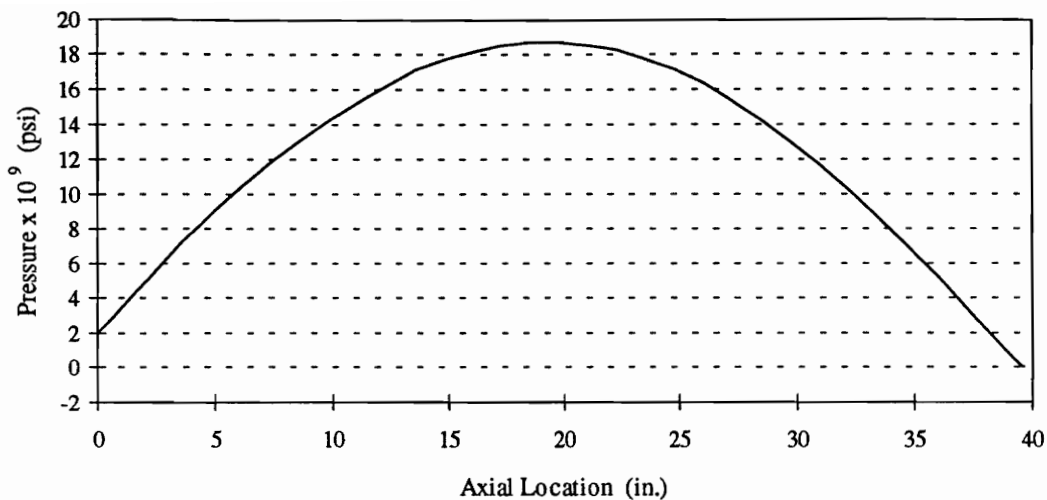


**Figure 13c. The Imaginary Component of the Target Pressure, Case 3.**

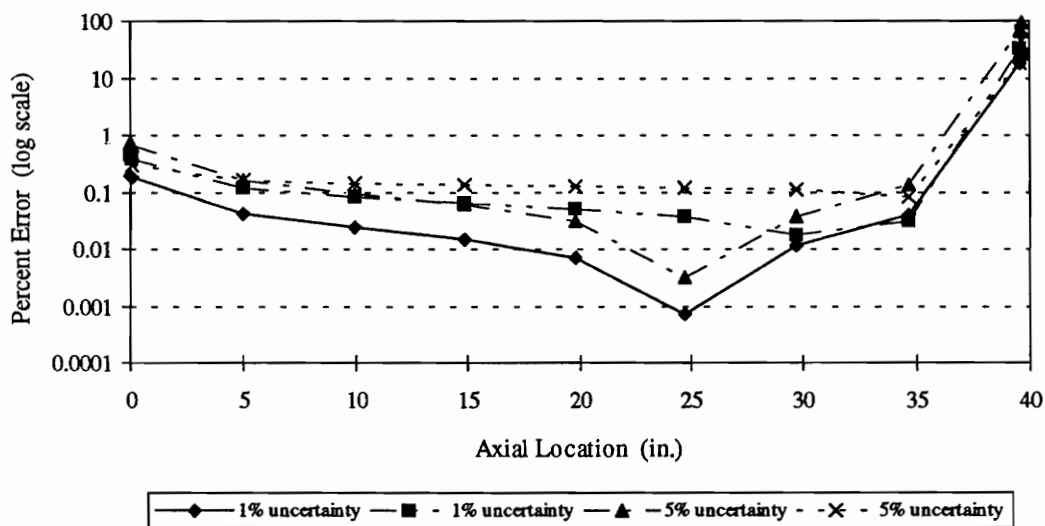


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 13d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 3.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

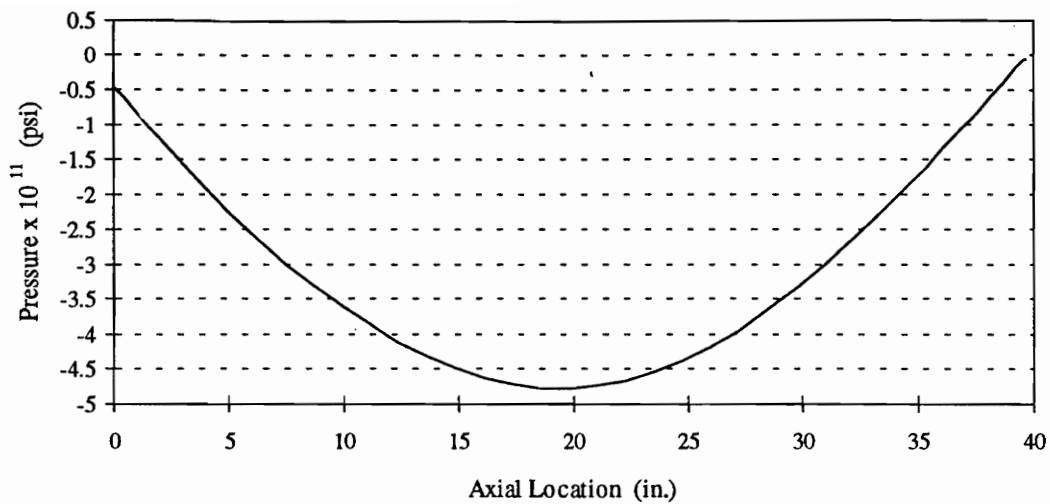


**Figure 14a. The Real Component of the Target Pressure, Case 4.**

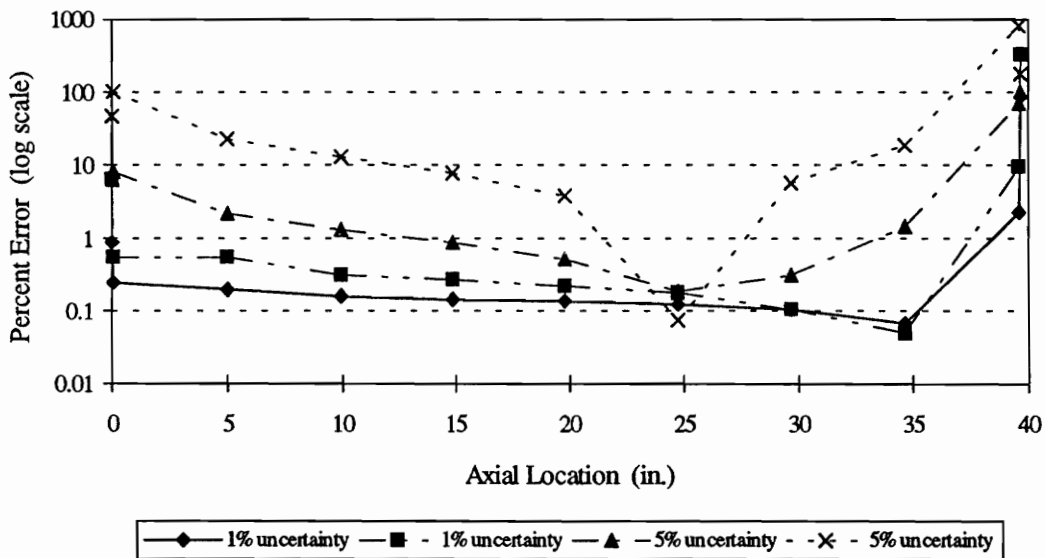


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 14b. Percent Error of the Real Component of the Estimated Pressure, Case 4.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

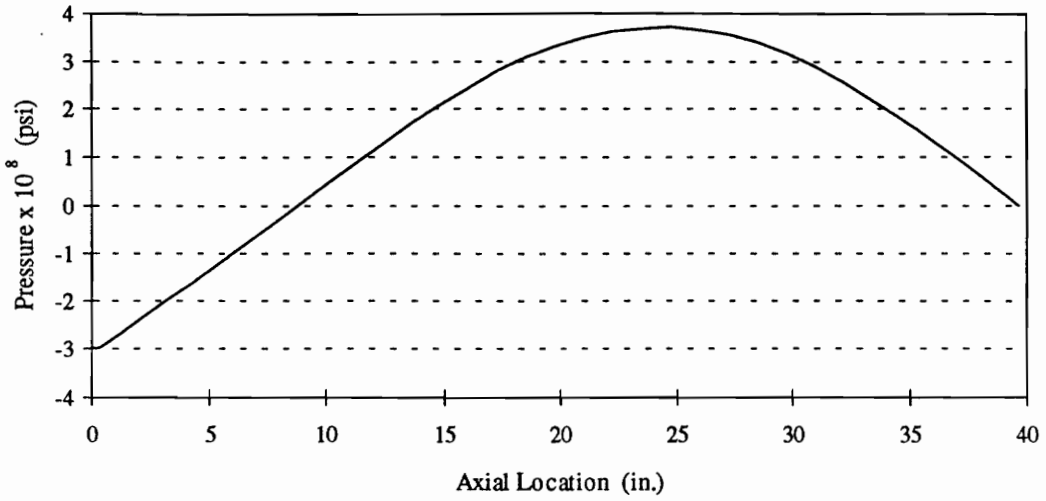


**Figure 14c. The Imaginary Component of the Target Pressure, Case 4.**

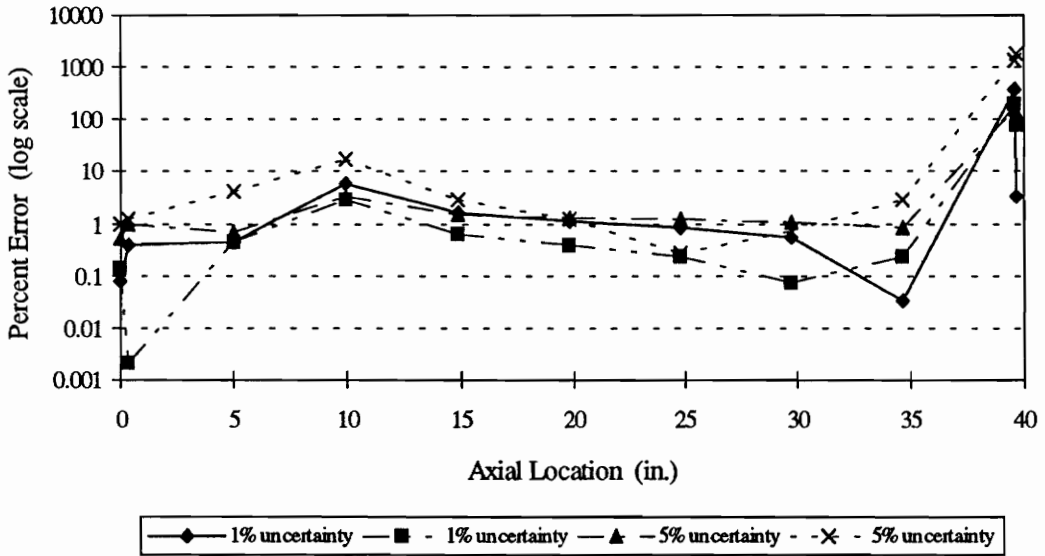


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 14d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 4.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)



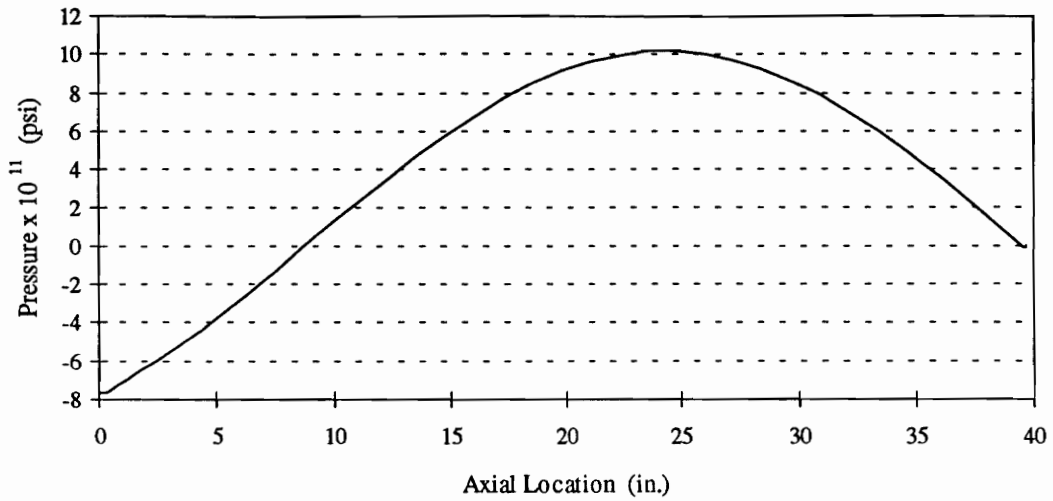
**Figure 15a. The Real Component of the Target Pressure, Case 5.**



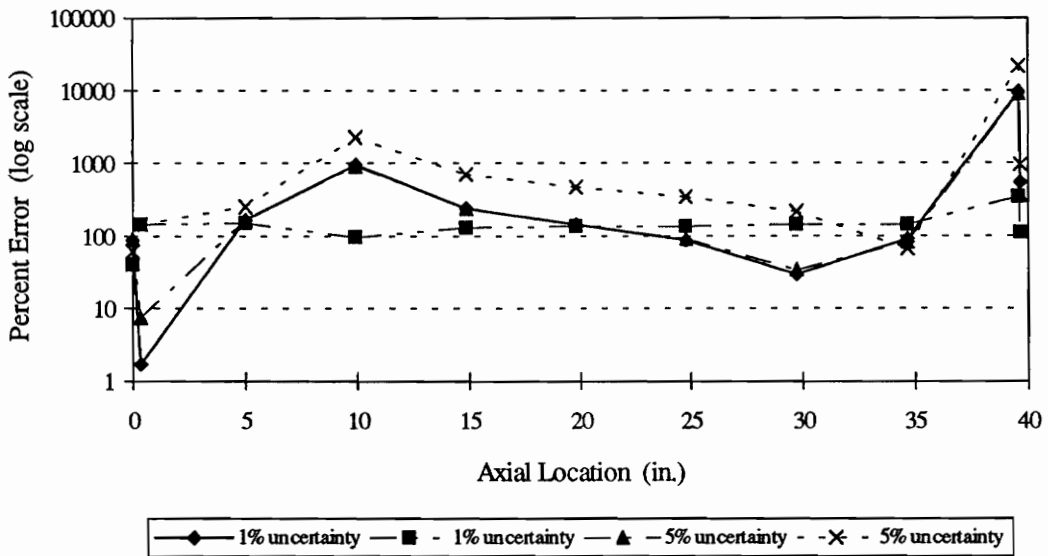
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 15b. Percent Error of the Real Component of the Estimated Pressure, Case 5.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)



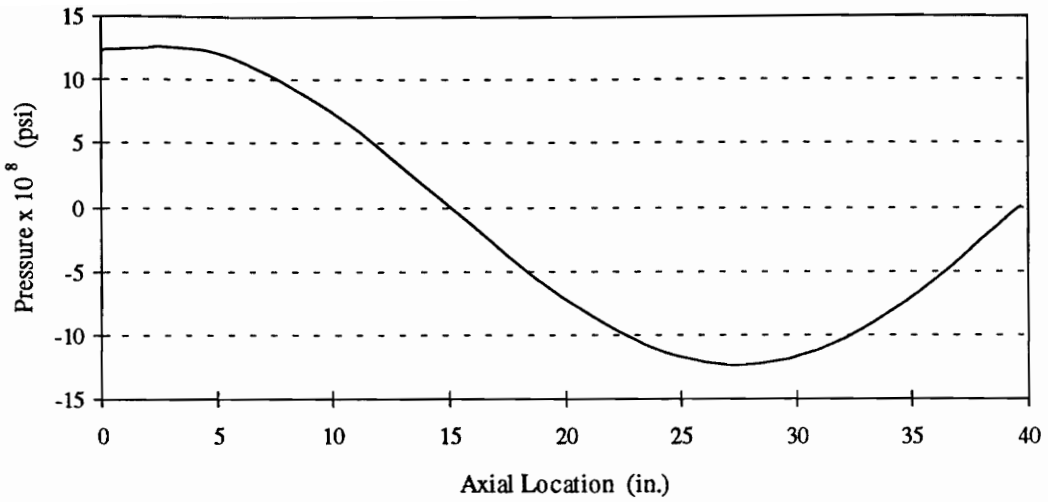


**Figure 15c. The Imaginary Component of the Target Pressure, Case 5.**

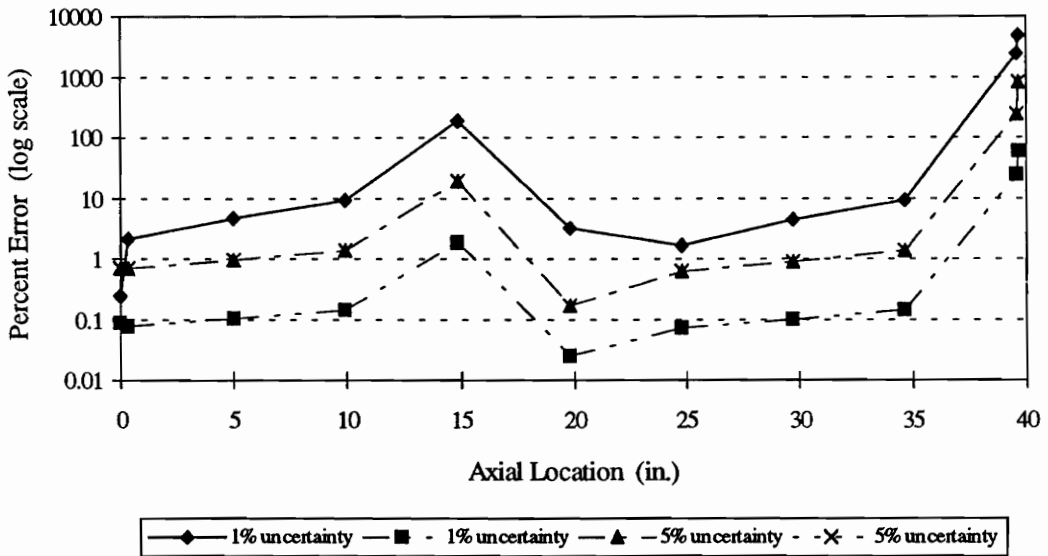


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 15d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 5.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

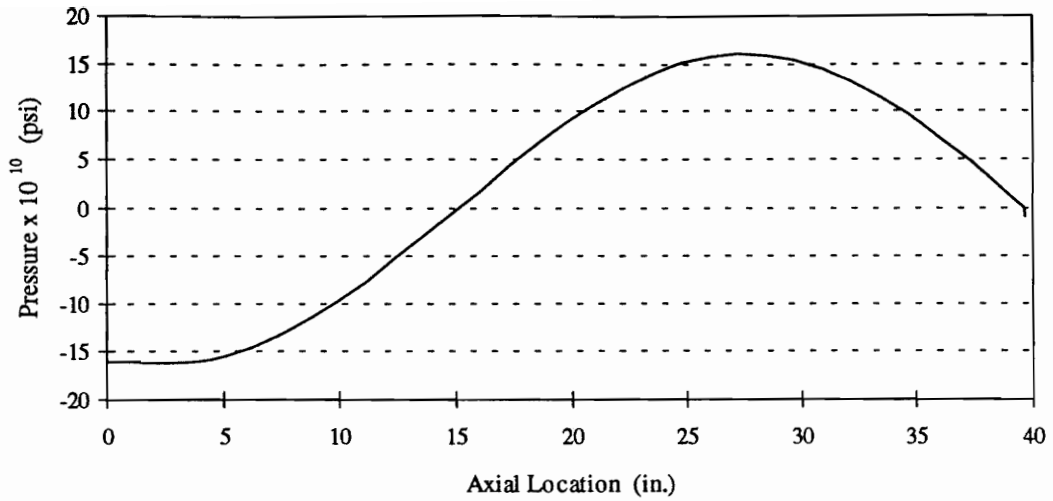


**Figure 16a. The Real Component of the Target Pressure, Case 6.**

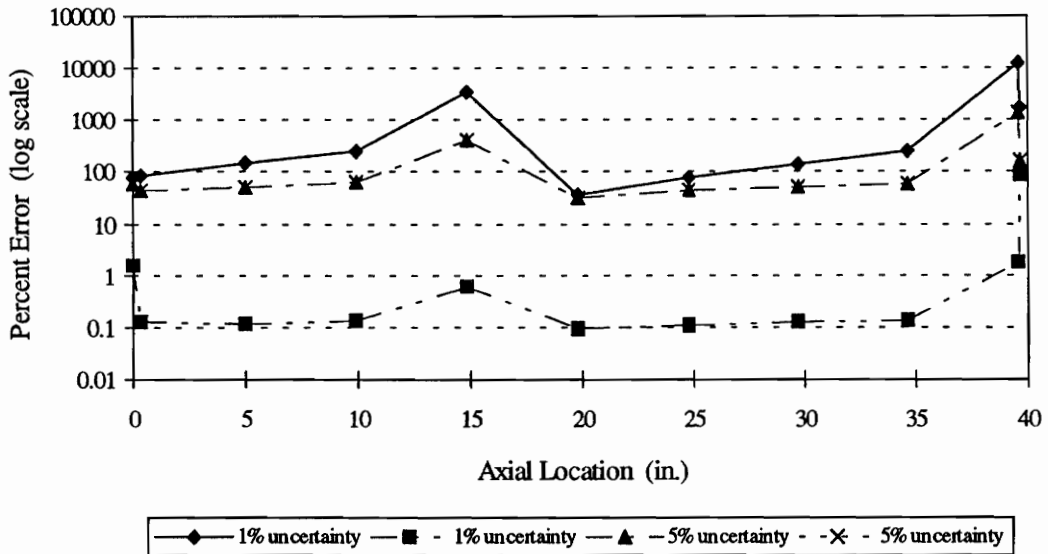


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 16b. Percent Error of the Real Component of the Estimated Pressure, Case 6.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

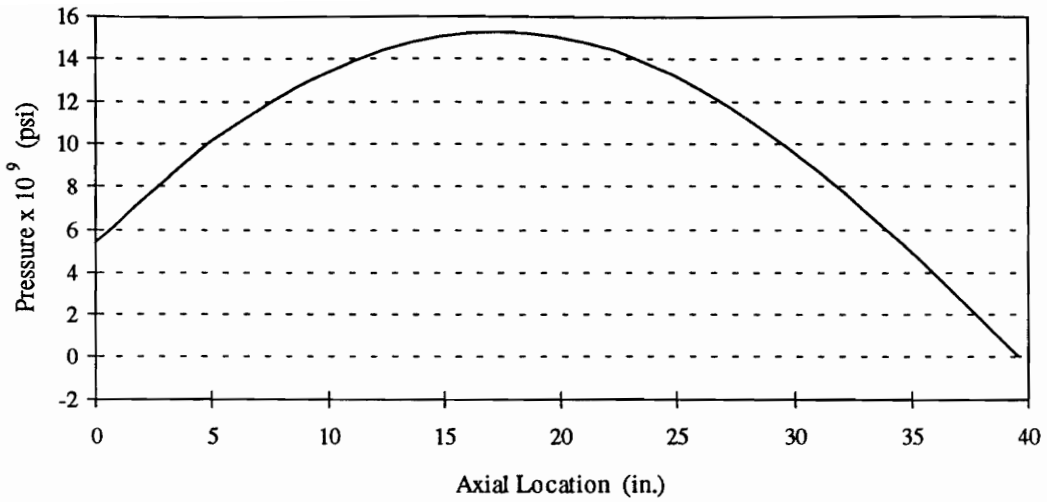


**Figure 16c. The Imaginary Component of the Target Pressure, Case 6.**

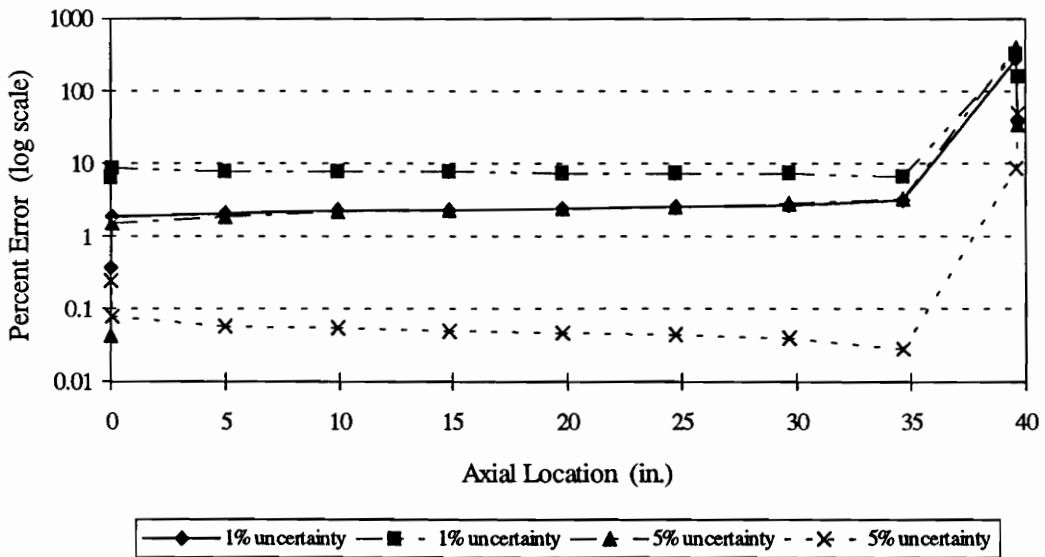


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 16d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 6.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

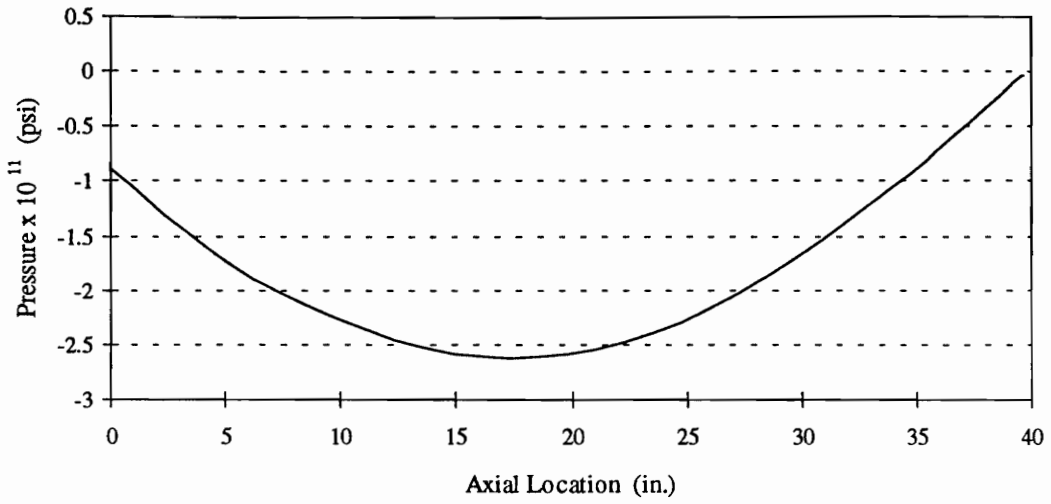


**Figure 17a. The Real Component of the Target Pressure, Case 7.**

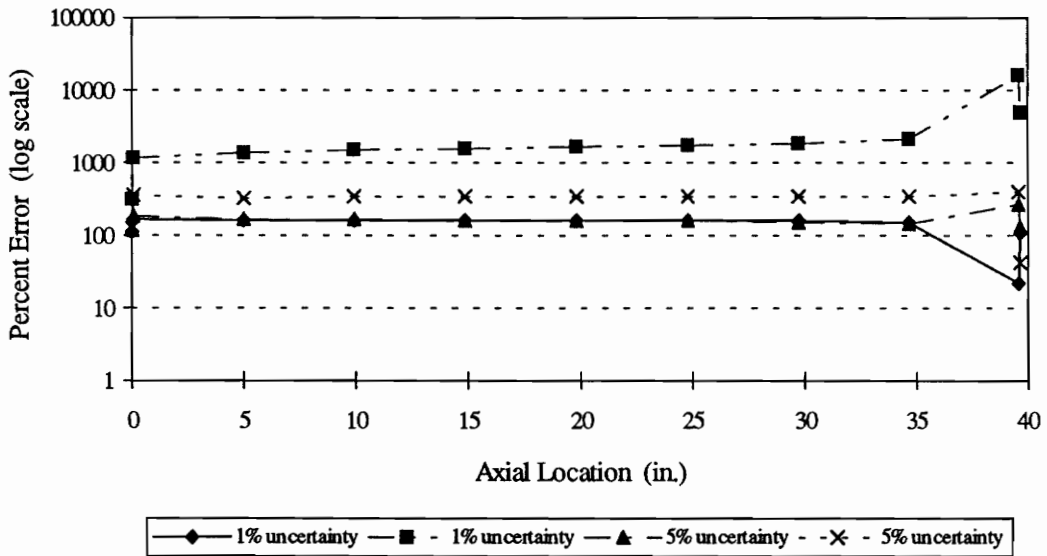


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 17b. Percent Error of the Real Component of the Estimated Pressure, Case 7.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

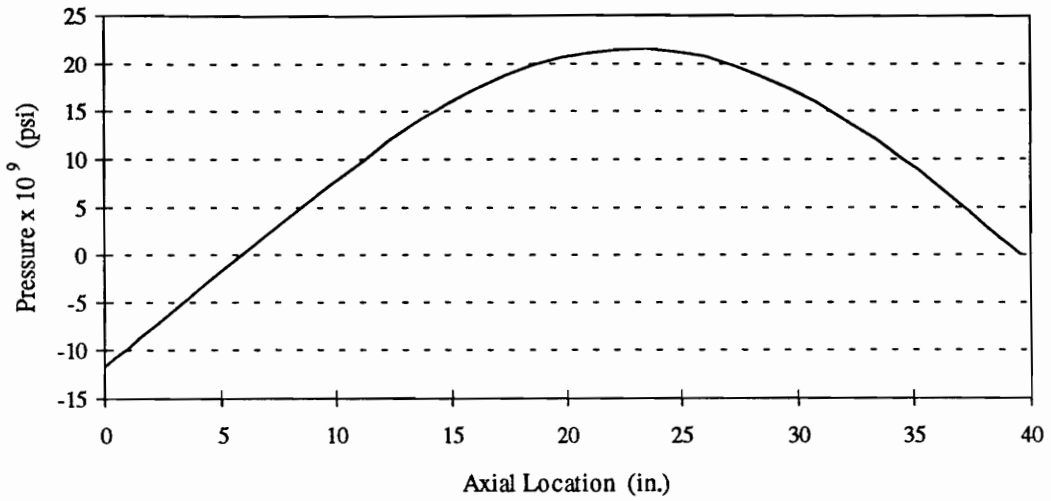


**Figure 17c. The Imaginary Component of the Target Pressure, Case 7.**

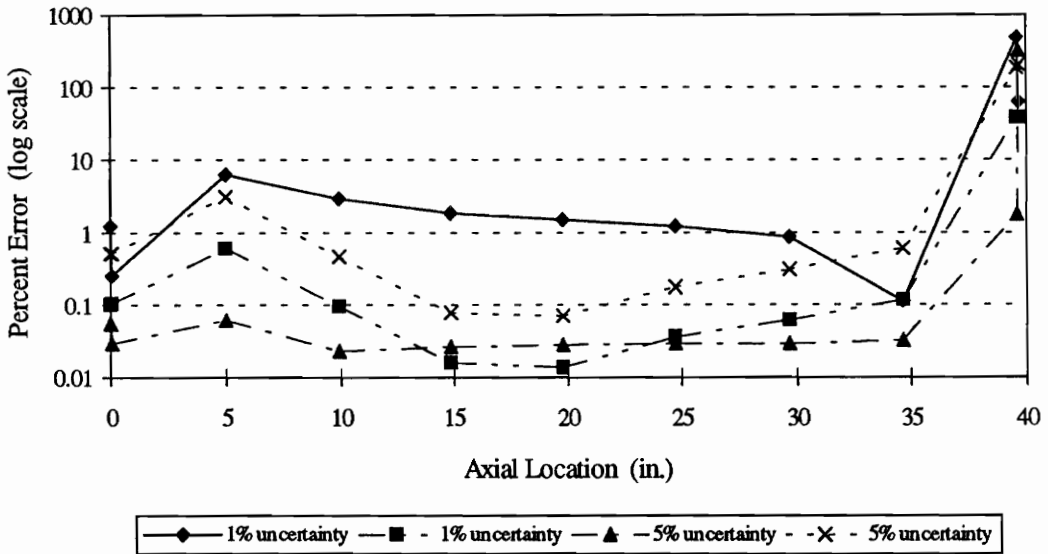


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 17d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 7.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

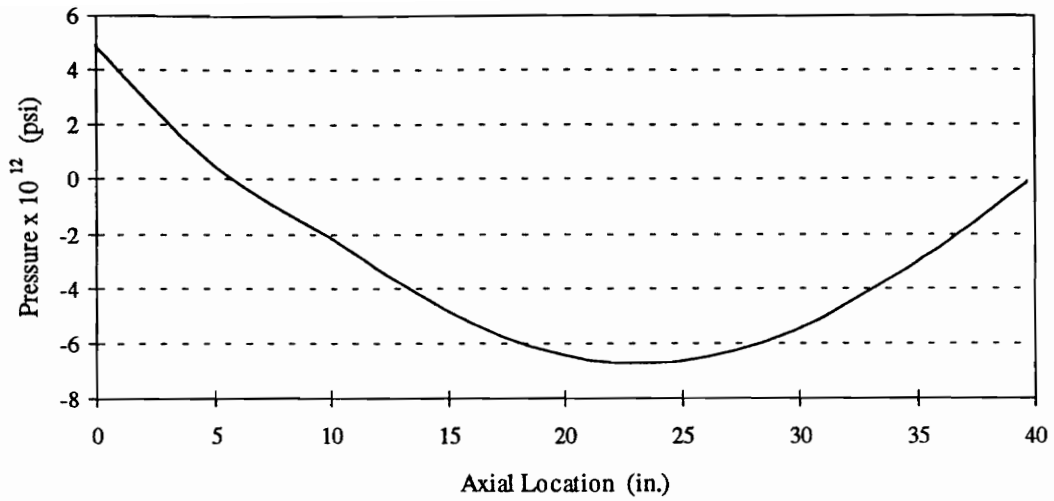


**Figure 18a. The Real Component of the Target Pressure, Case 8.**

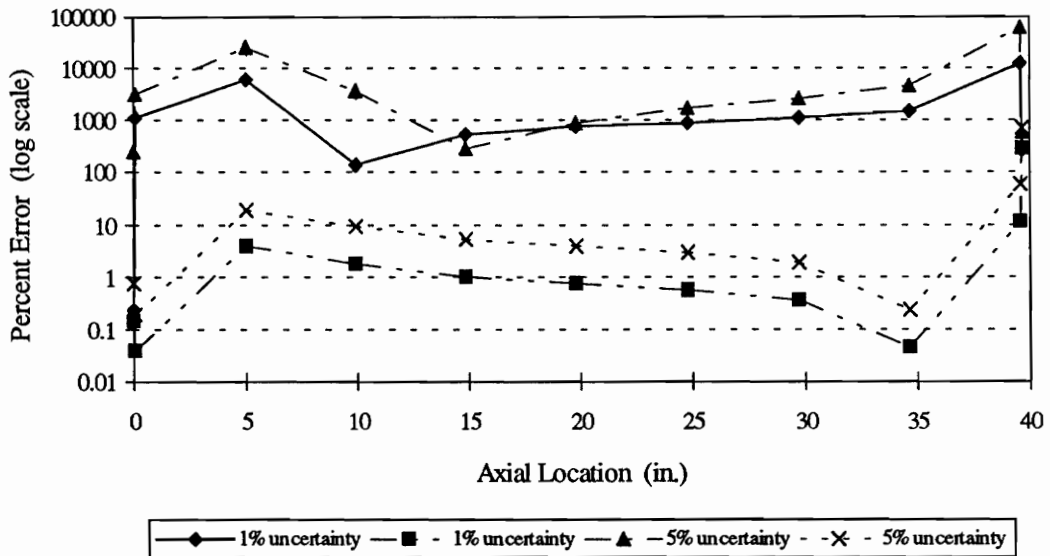


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 18b. Percent Error of the Real Component of the Estimated Pressure, Case 8.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)

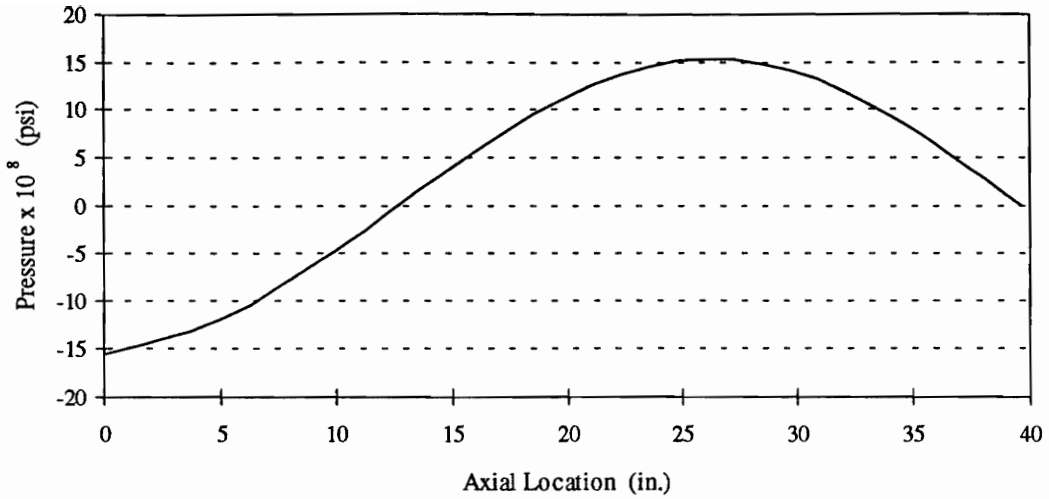


**Figure 18c. The Imaginary Component of the Target Pressure, Case 8.**

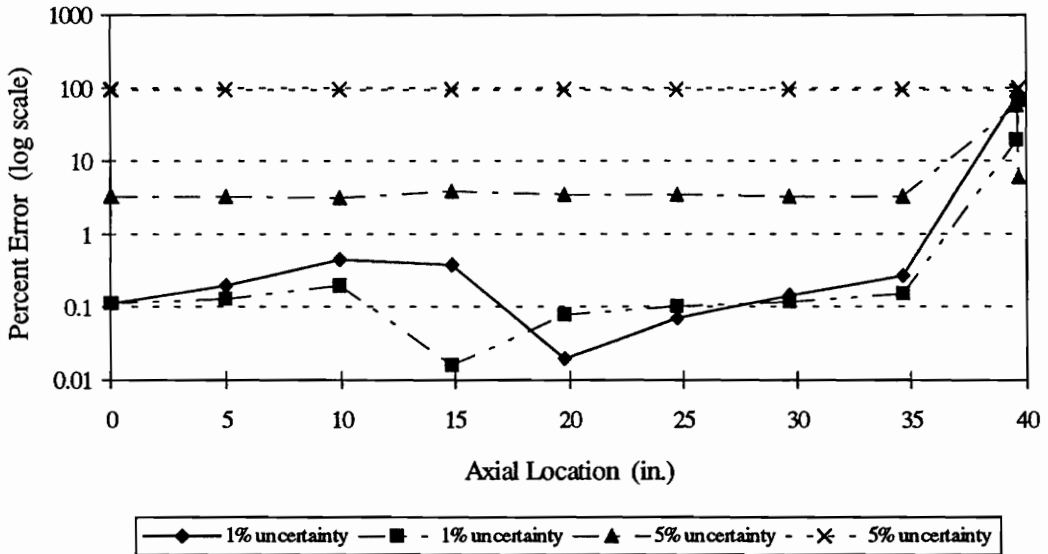


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 18d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 8.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)



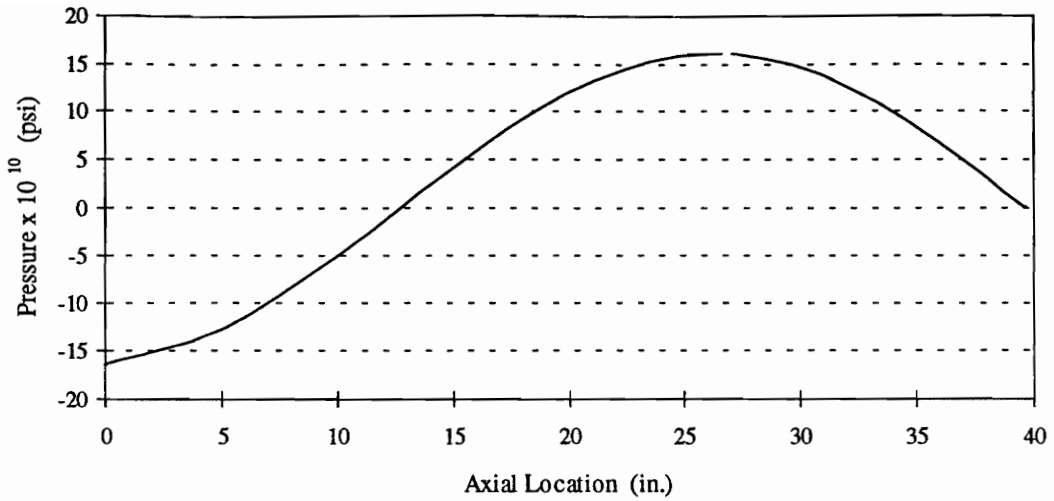
**Figure 19a. The Real Component of the Target Pressure, Case 9.**



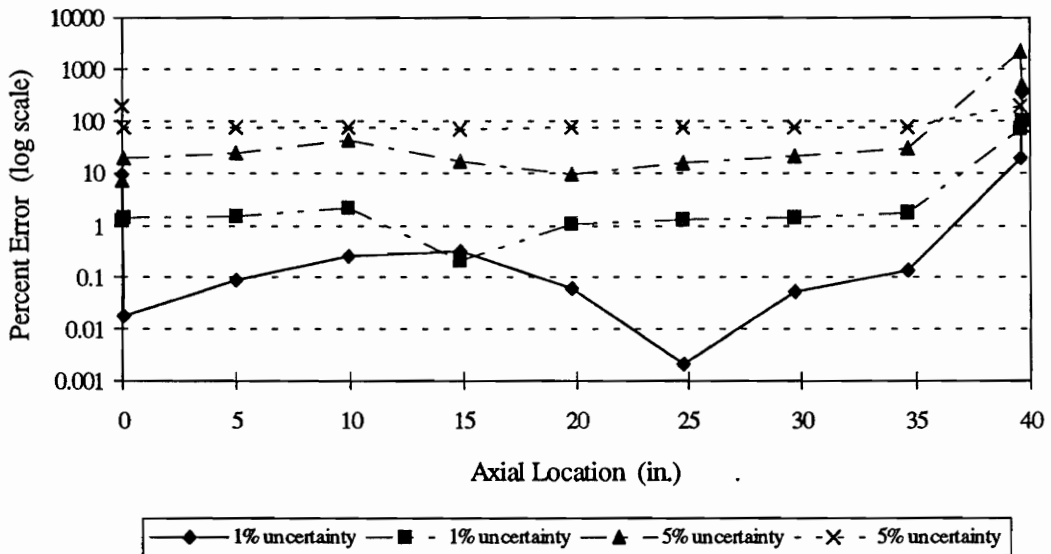
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 19b. Percent Error of the Real Component of the Estimated Pressure, Case 9.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)



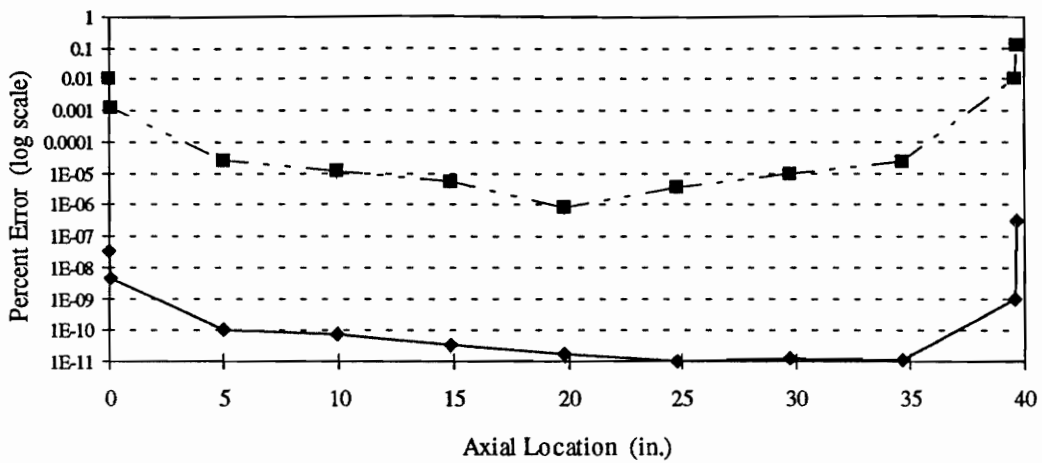


**Figure 19c. The Imaginary Component of the Target Pressure, Case 9.**



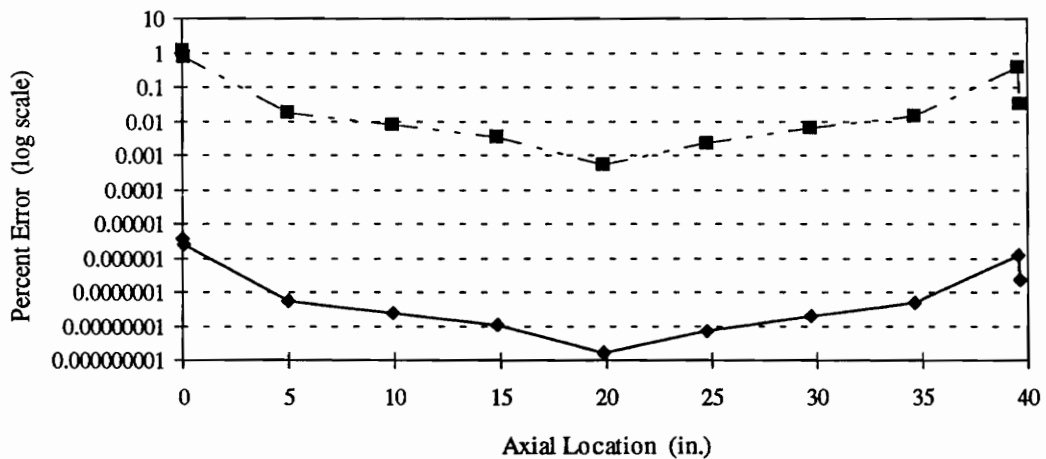
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 19d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 9.**  
 (Two Different Starting Positions for Both One- and Five-Percent Uncertainty)



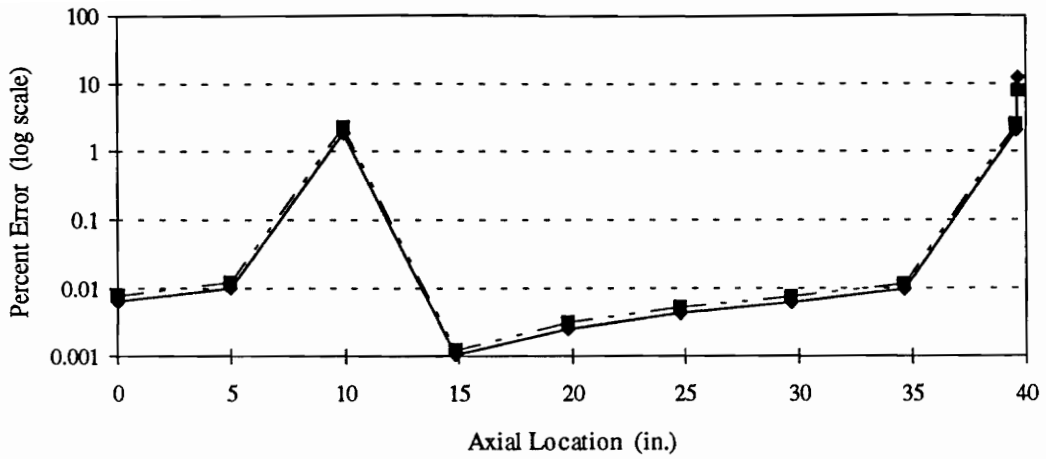
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 20a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 1.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



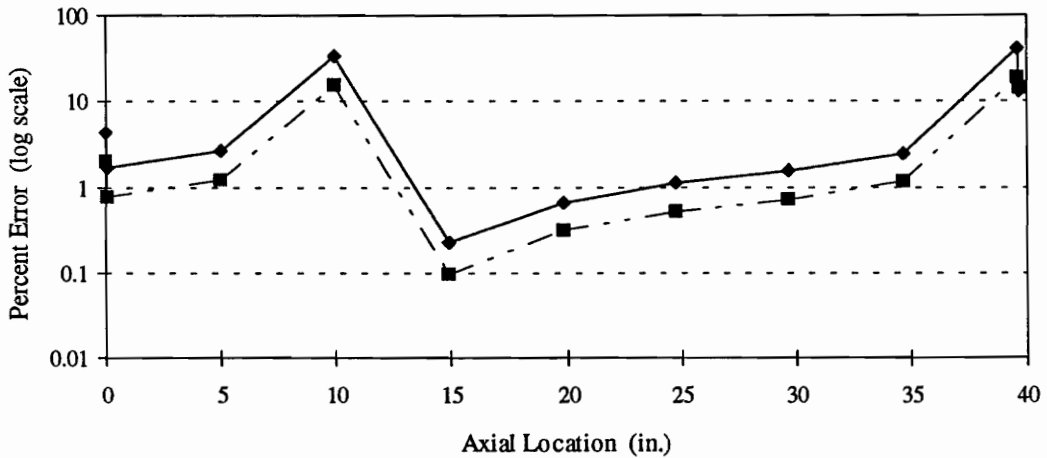
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 20b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 1.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



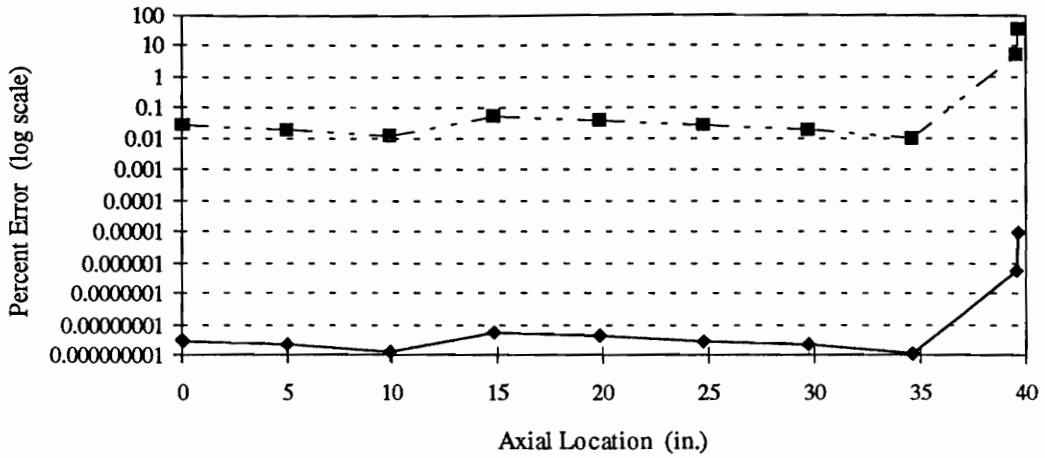
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 21a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 2.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



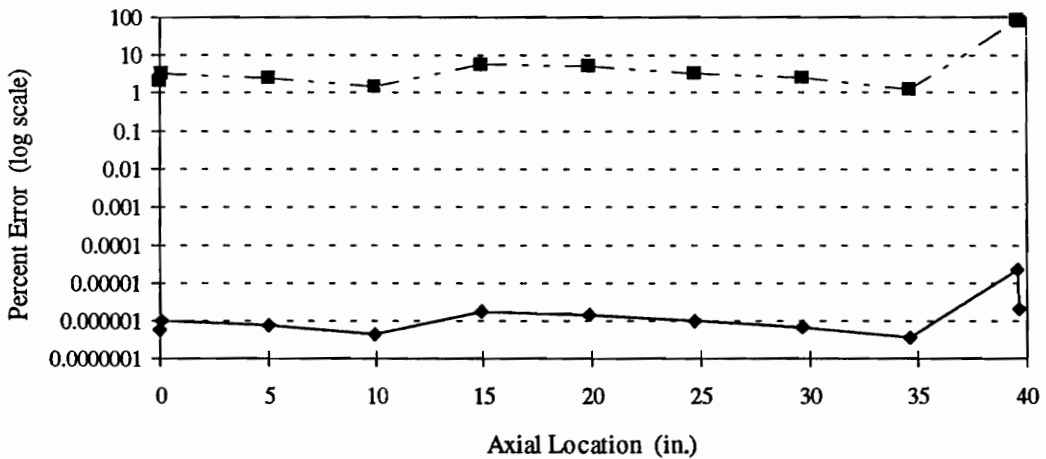
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 21b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 2.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



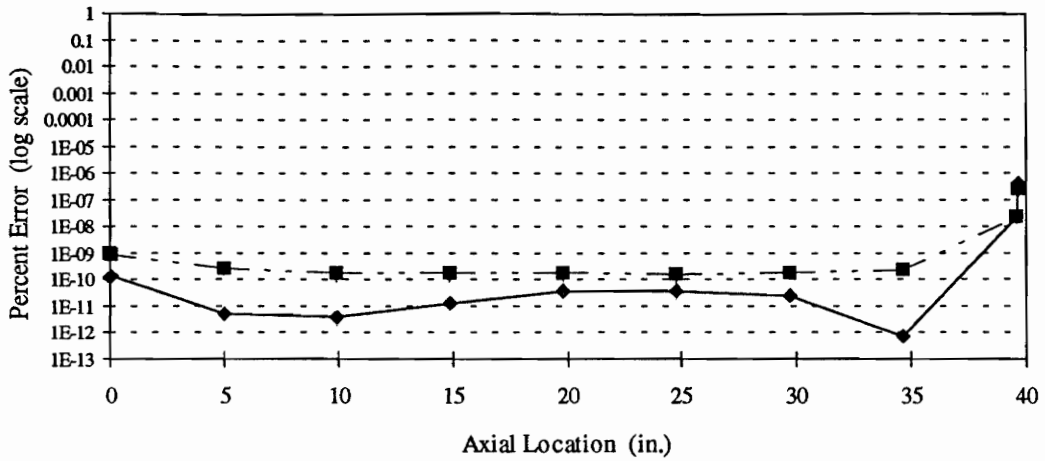
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 22a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 3.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



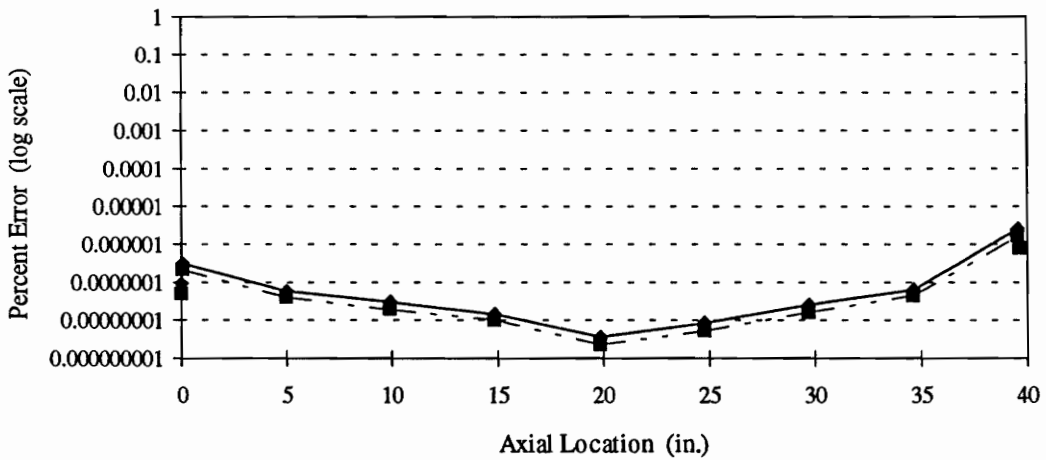
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 22b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 3.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



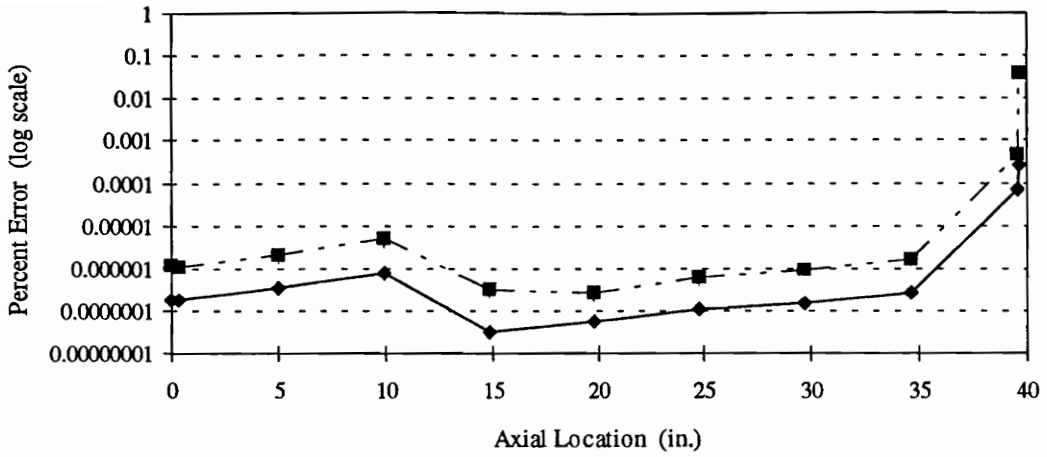
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 23a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 4.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



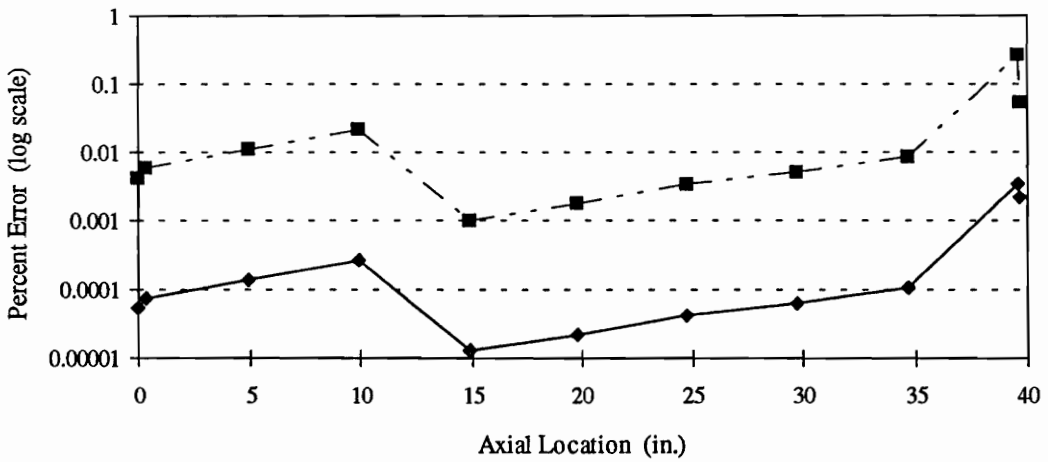
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 23b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 4.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



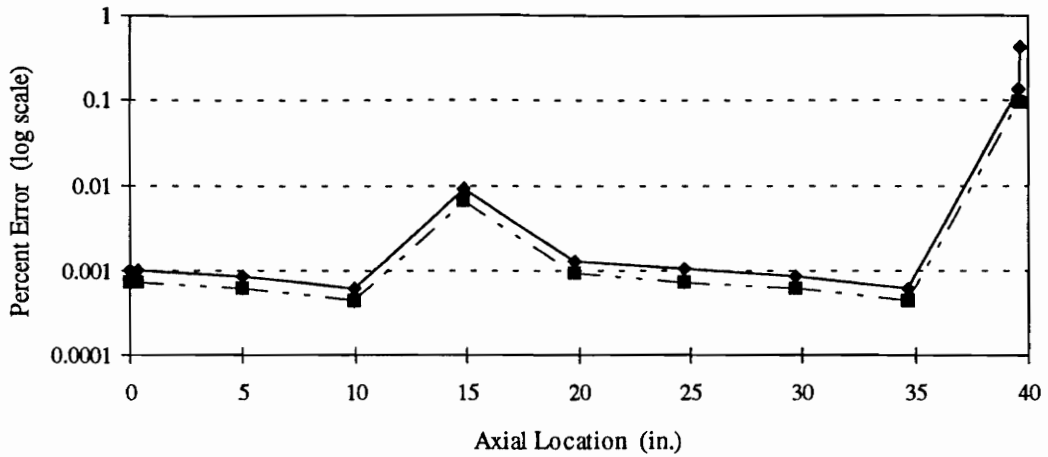
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 24a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 5.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



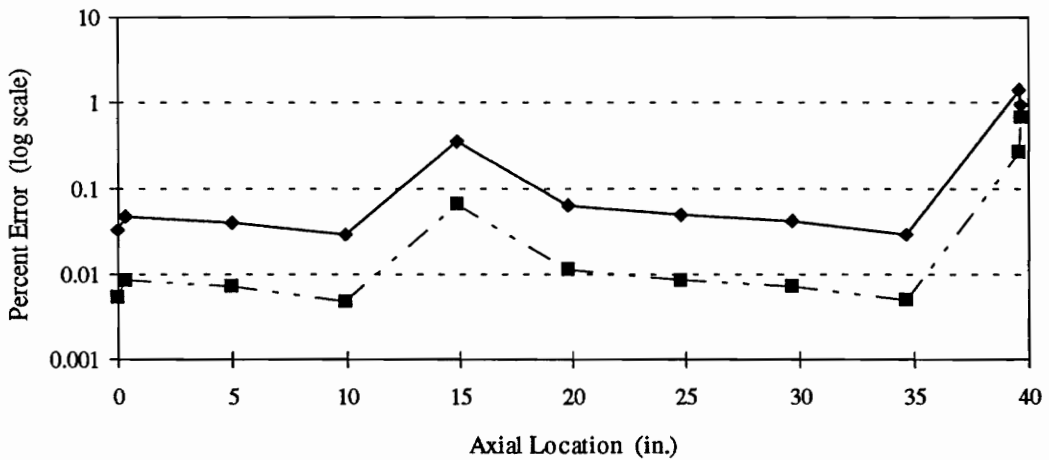
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 24b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 5.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



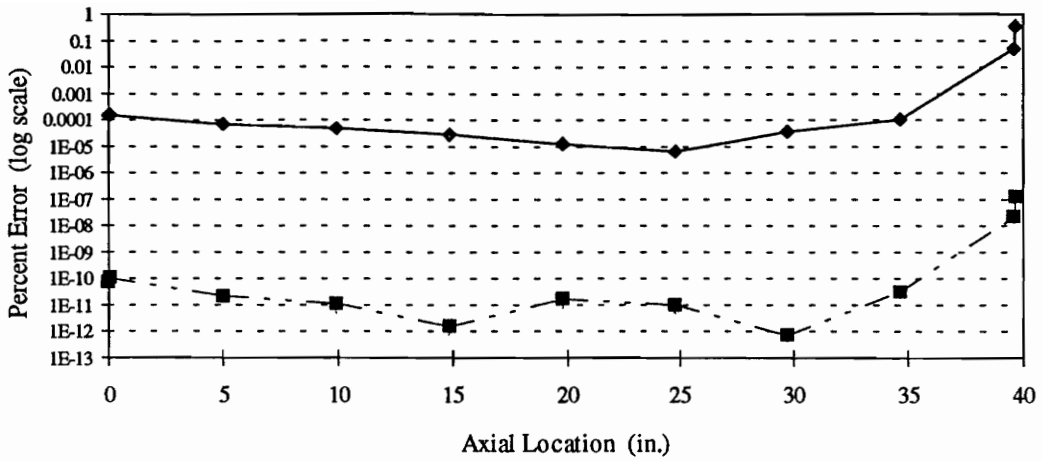
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 25a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 6.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



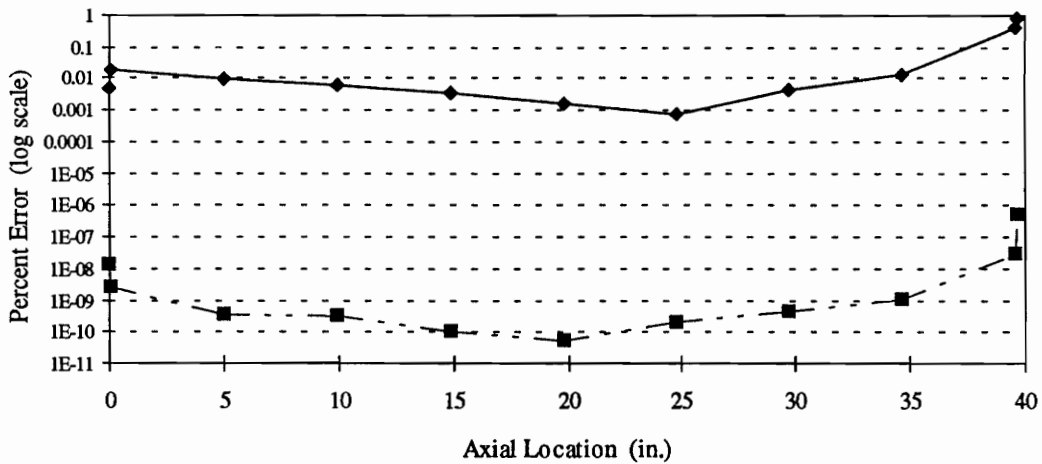
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 25b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 6.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

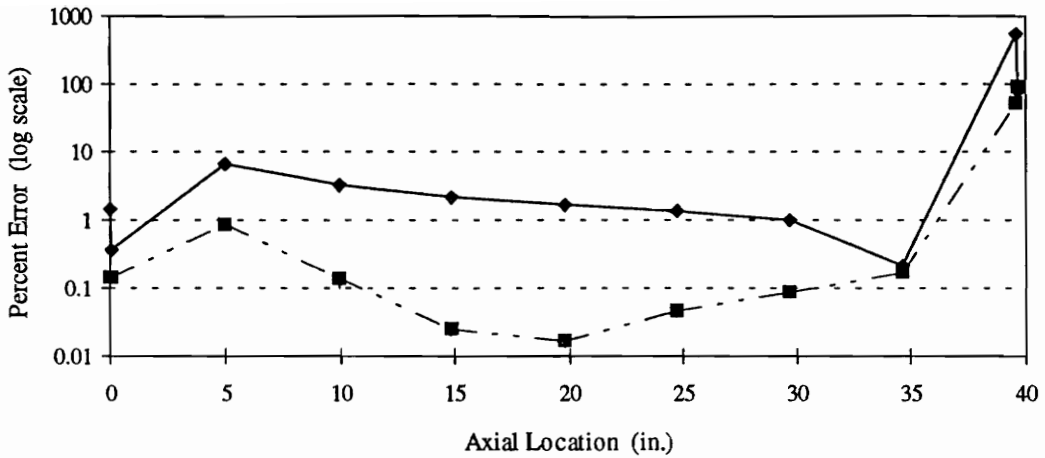
**Figure 26a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 7.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

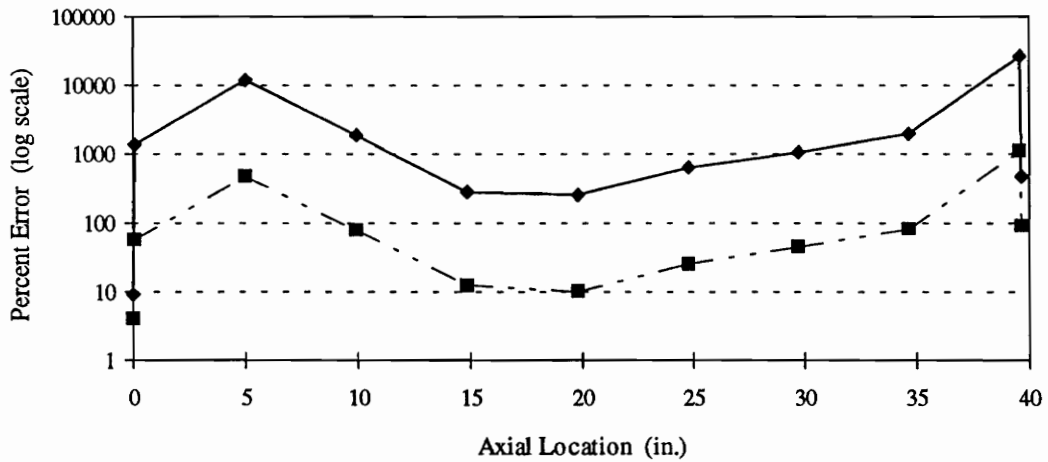
**Figure 26b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 7.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)





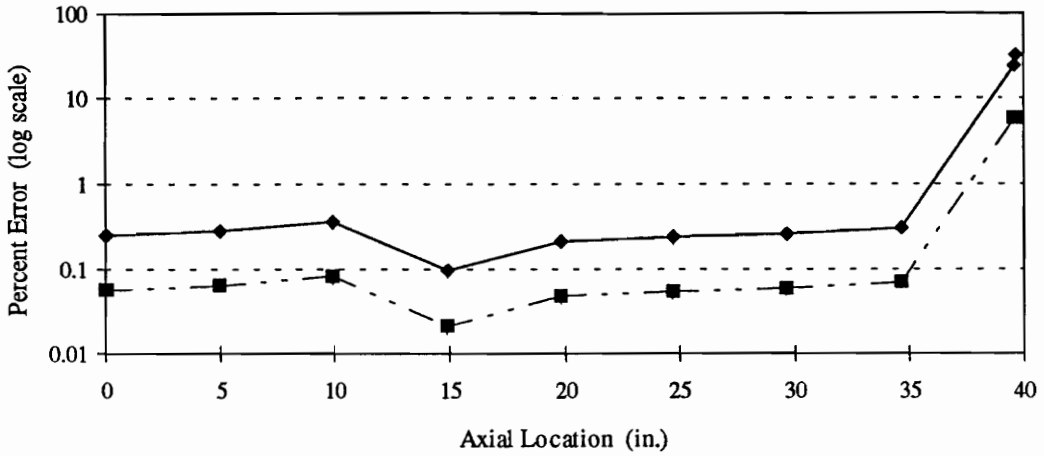
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 27a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 8.**  
 (Two Different Starting Positions for Zero-Percent Uncertainty)



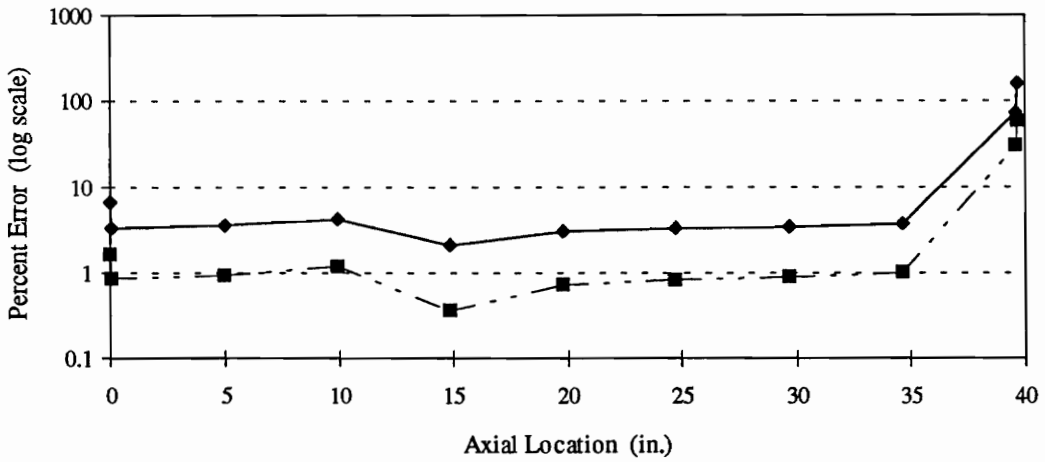
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 27b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 8.**  
 (Two Different Starting Positions for Zero-Percent Uncertainty)



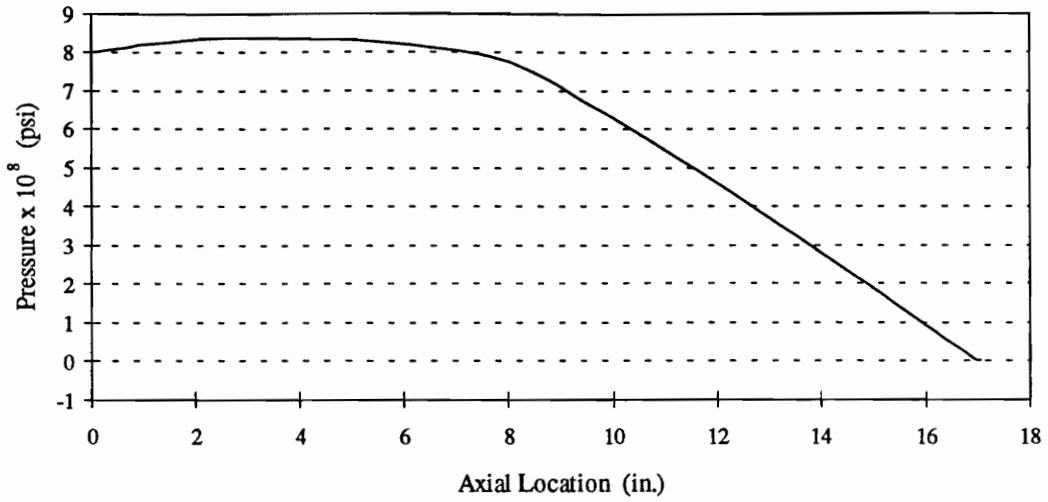
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 28a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 9.**  
 (Two Different Starting Positions for Zero-Percent Uncertainty)

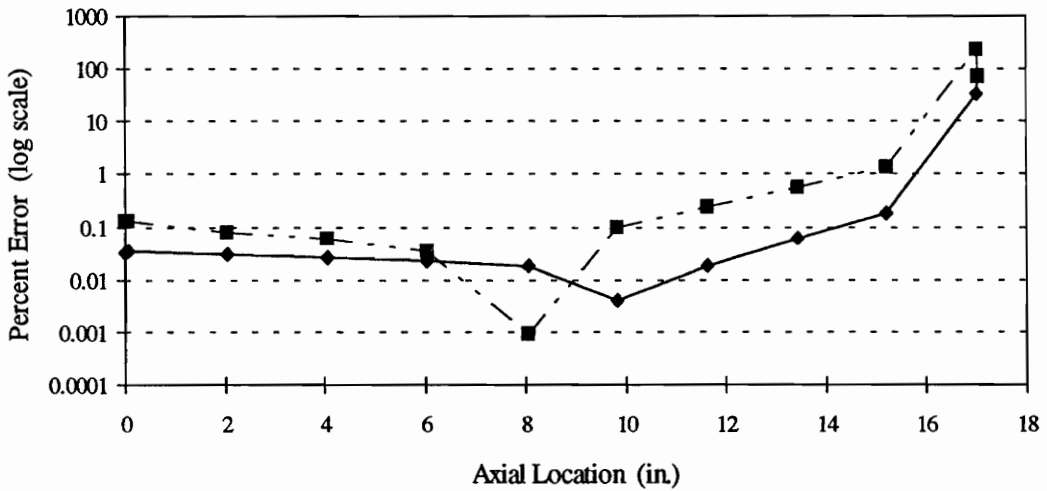


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 28b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 9.**  
 (Two Different Starting Positions for Zero-Percent Uncertainty)

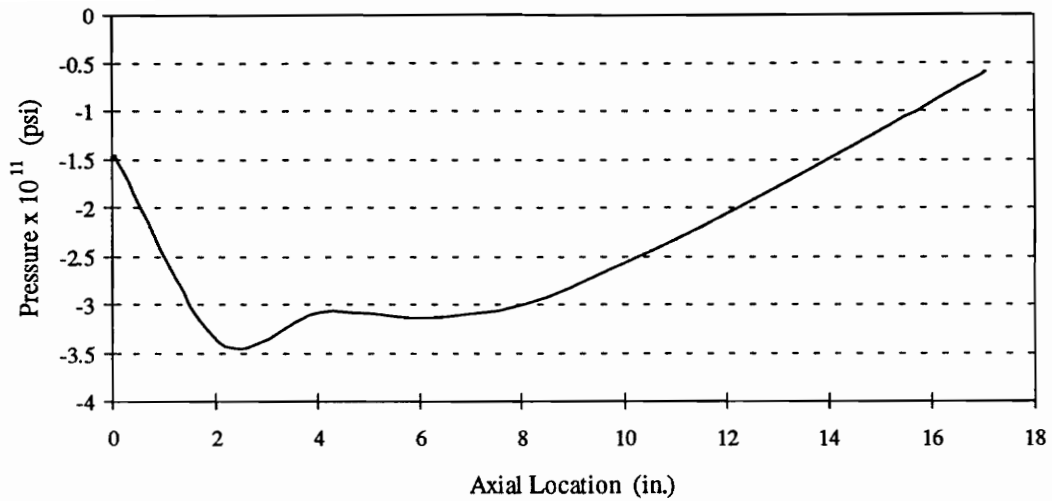


**Figure 29a. The Real Component of the Target Pressure, Case 10.**

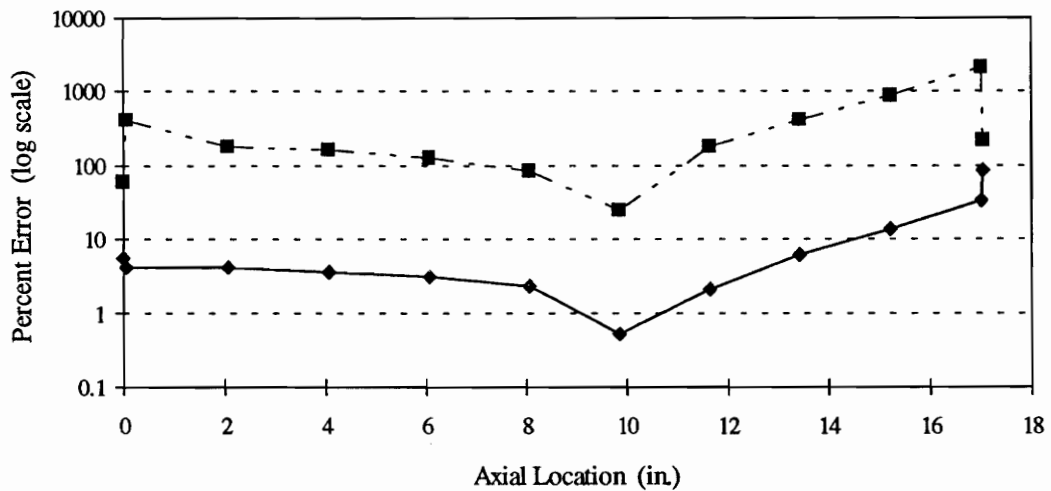


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 29b. Percent Error of the Real Component of the Estimated Pressure, Case 10.**  
 (Two Different Starting Positions for One-Percent Uncertainty)

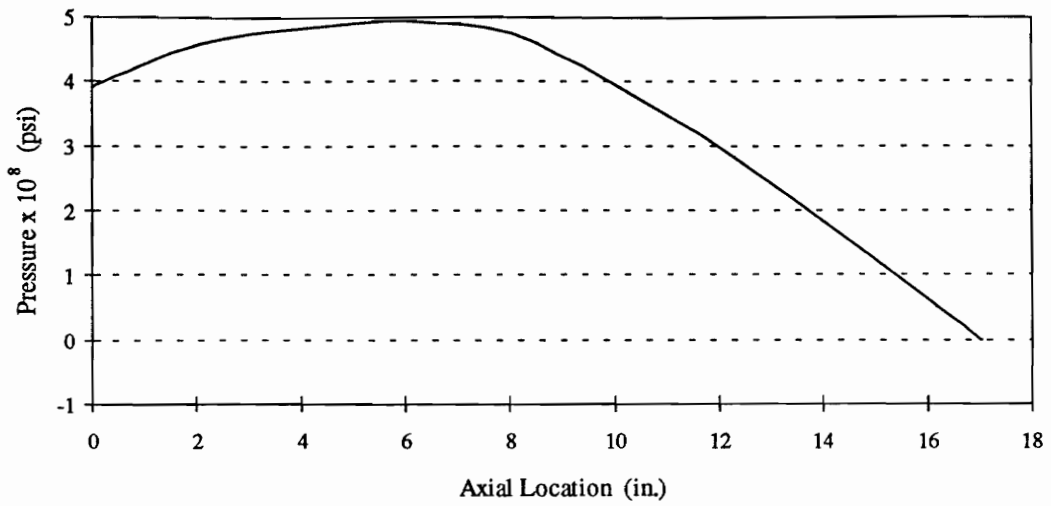


**Figure 29c. The Imaginary Component of the Target Pressure, Case 10.**

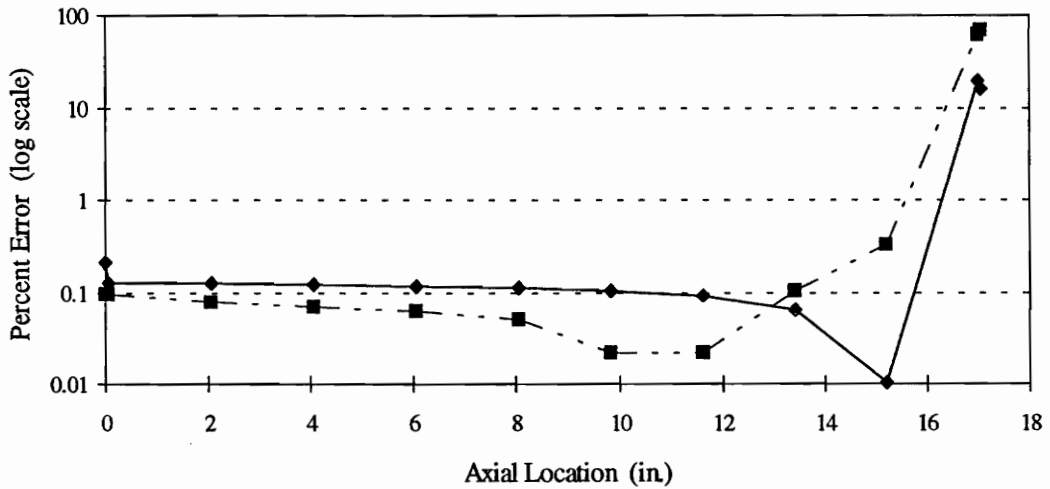


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 29d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 10.**  
 (Two Different Starting Positions for One-Percent Uncertainty)

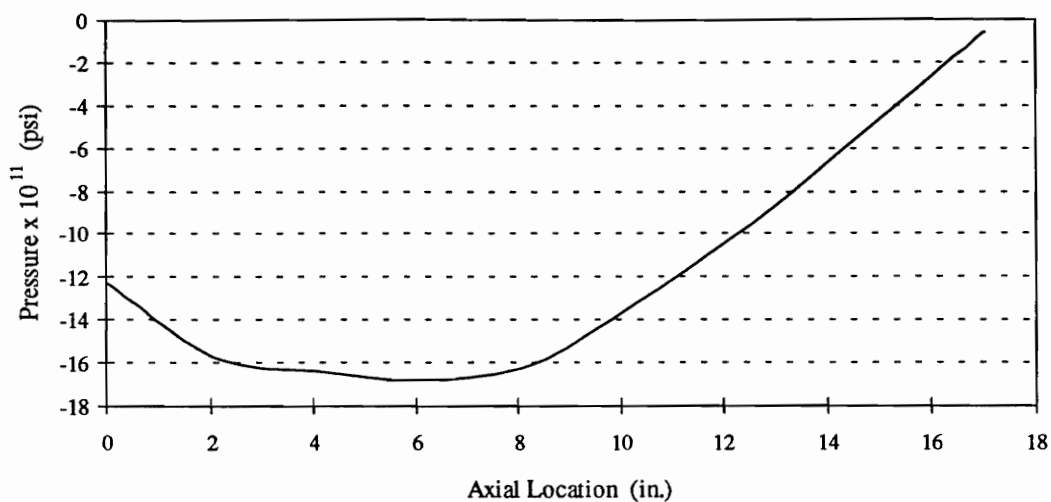


**Figure 30a. The Real Component of the Target Pressure, Case 11.**

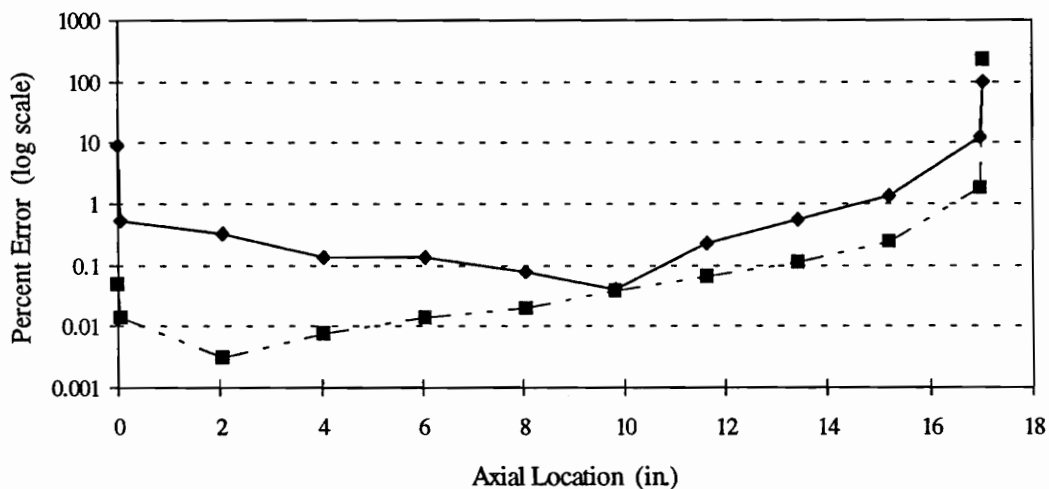


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 30b. Percent Error of the Real Component of the Estimated Pressure, Case 11.**  
 (Two Different Starting Positions for One-Percent Uncertainty)

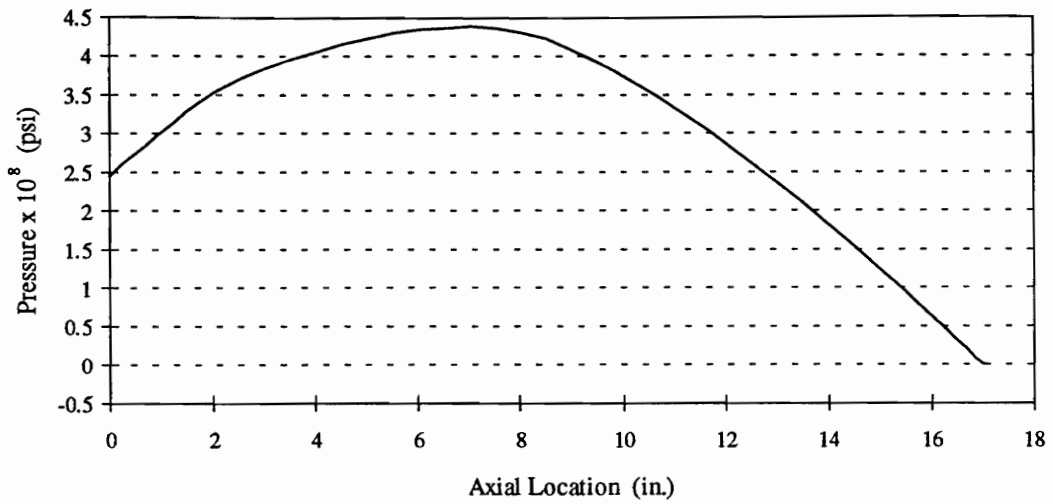


**Figure 30c. The Imaginary Component of the Target Pressure, Case 11.**

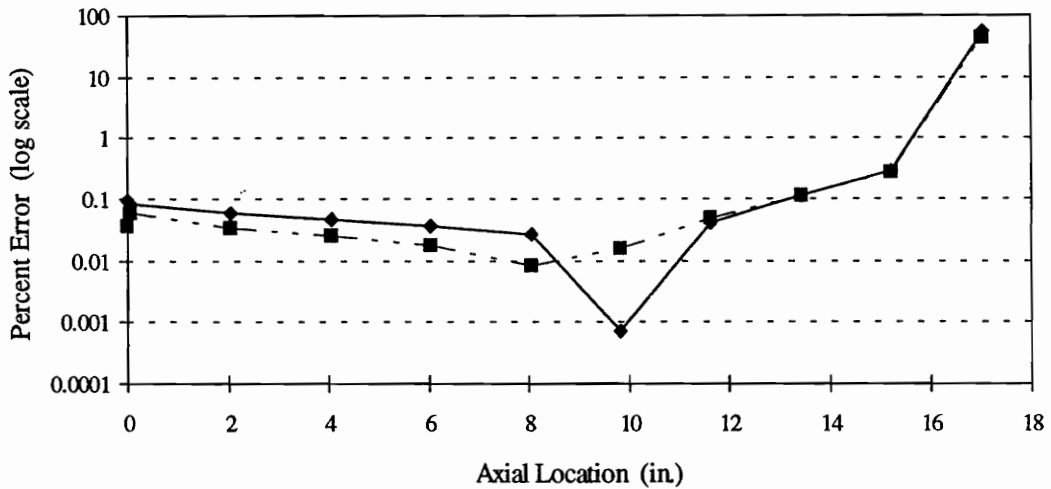


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 30d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 11.**  
(Two Different Starting Positions for One-Percent Uncertainty)

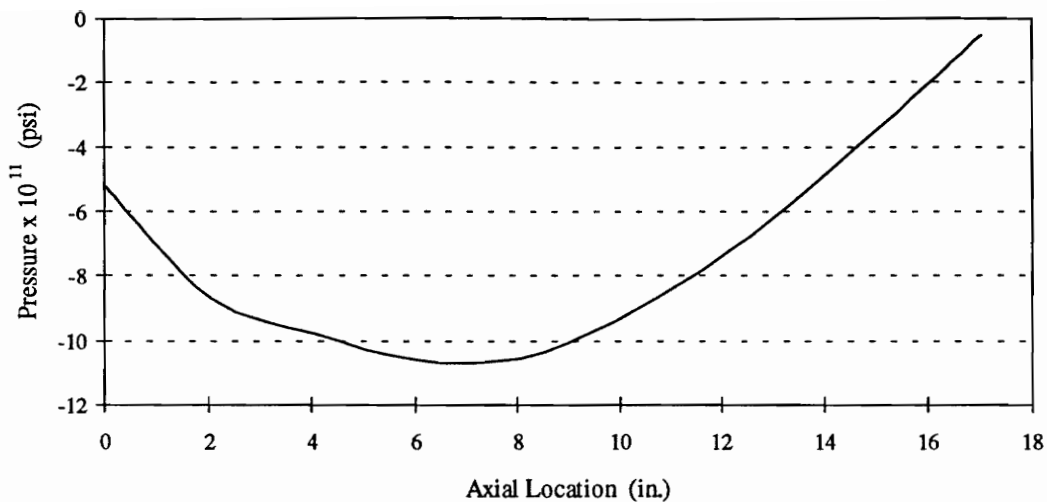


**Figure 31a. The Real Component of the Target Pressure, Case 12.**

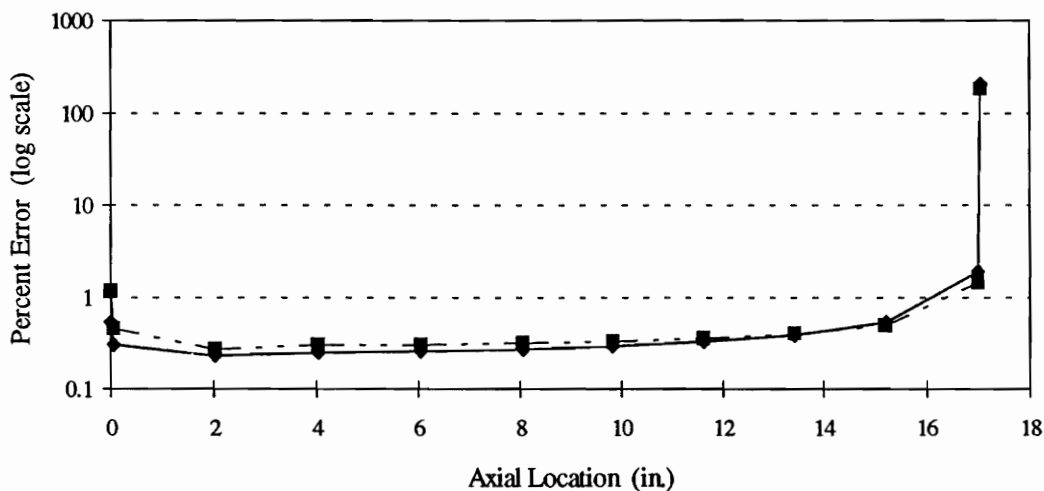


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 31b. Percent Error of the Real Component of the Estimated Pressure, Case 12.**  
 (Two Different Starting Positions for One-Percent Uncertainty)



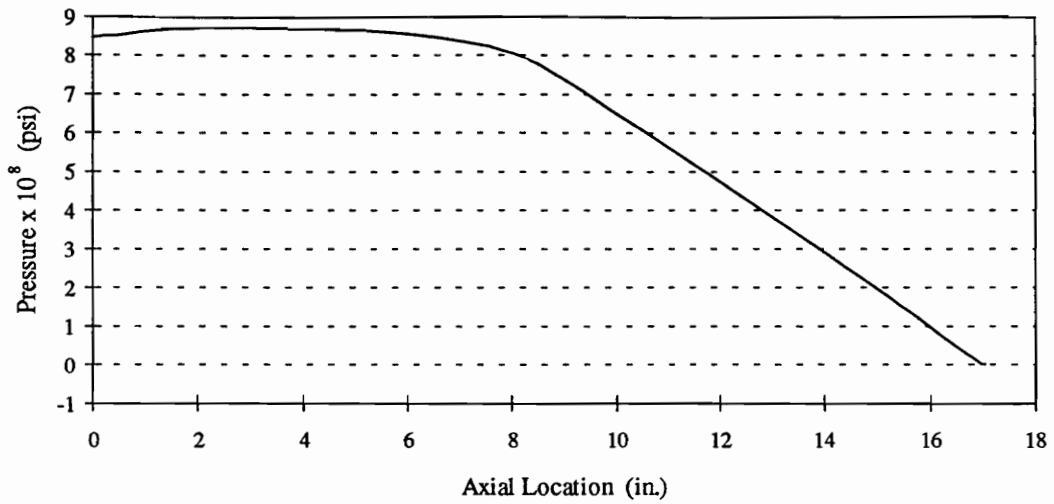
**Figure 31c. The Imaginary Component of the Target Pressure, Case 12.**



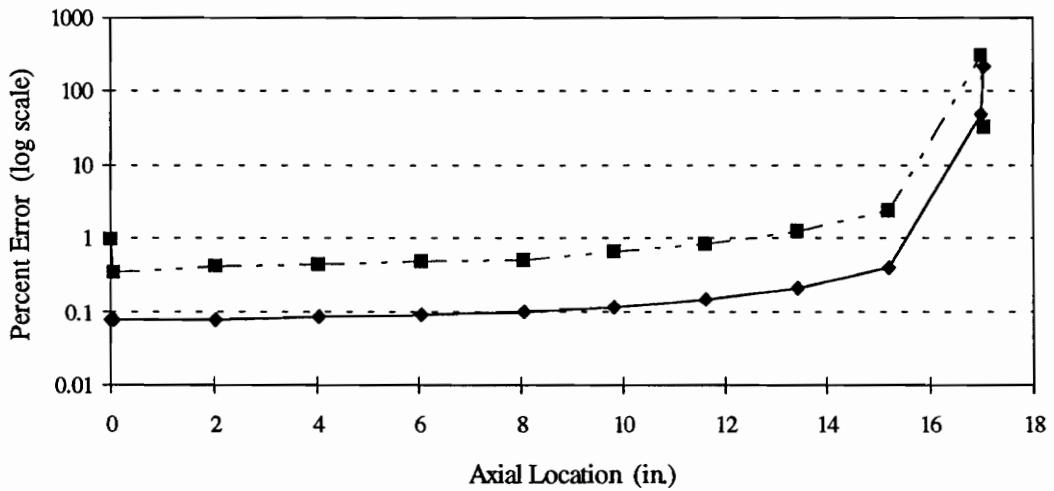
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 31d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 12.**  
 (Two Different Starting Positions for One-Percent Uncertainty)



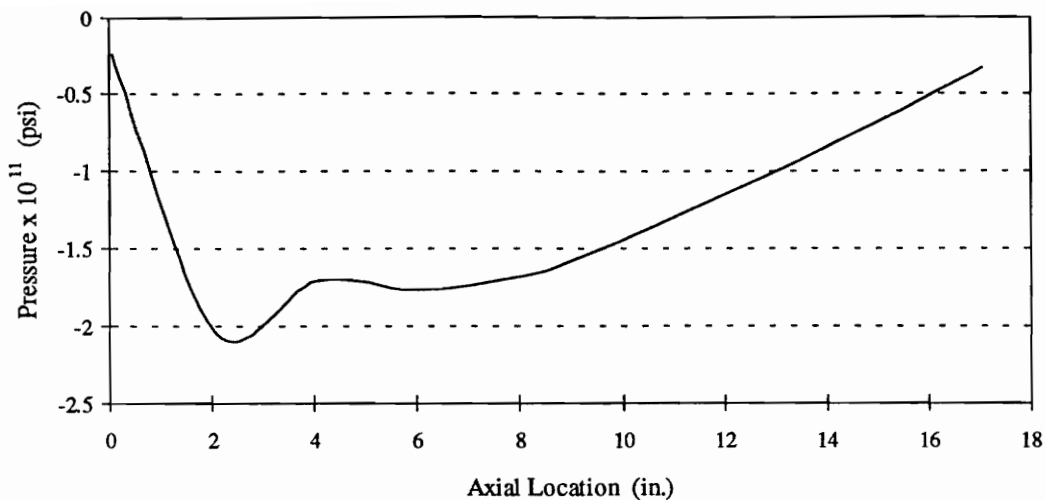


**Figure 32a. The Real Component of the Target Pressure, Case 13.**

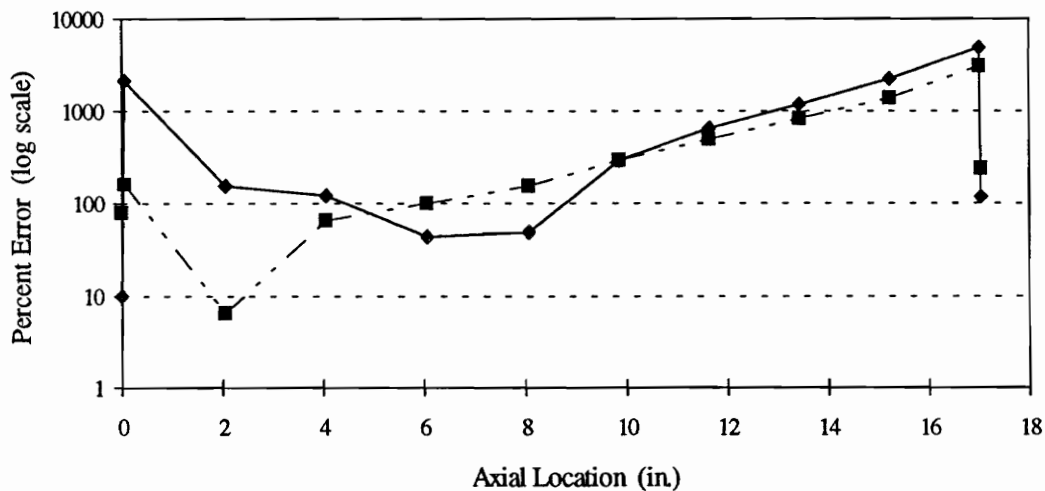


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 32b. Percent Error of the Real Component of the Estimated Pressure, Case 13.**  
 (Two Different Starting Positions for One-Percent Uncertainty)

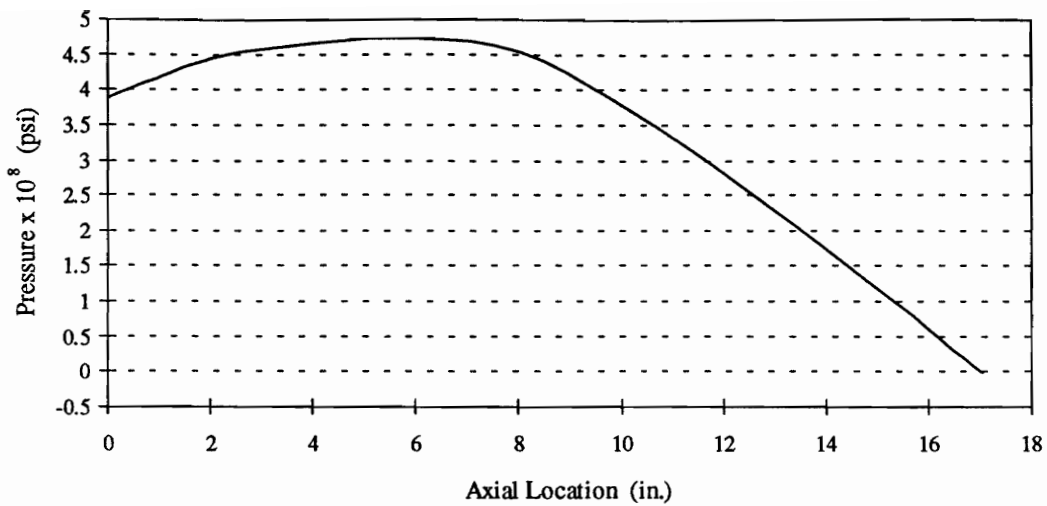


**Figure 32c. The Imaginary Component of the Target Pressure, Case 13.**

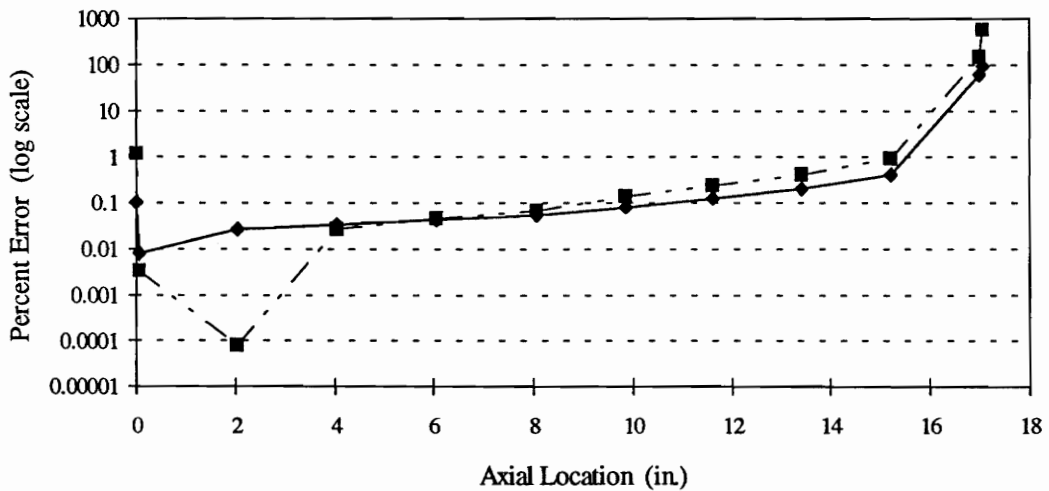


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 32d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 13.**  
 (Two Different Starting Positions for One-Percent Uncertainty)

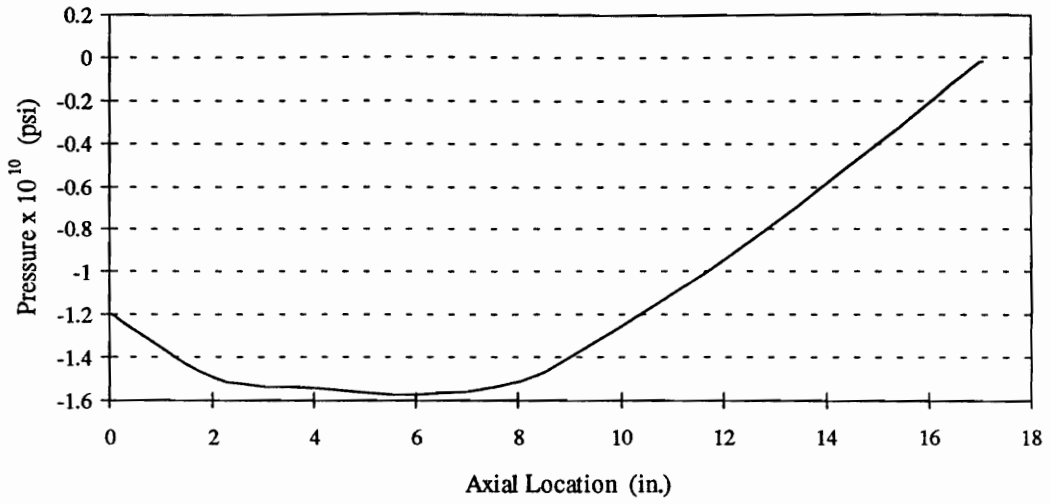


**Figure 33a. The Real Component of the Target Pressure, Case 14.**

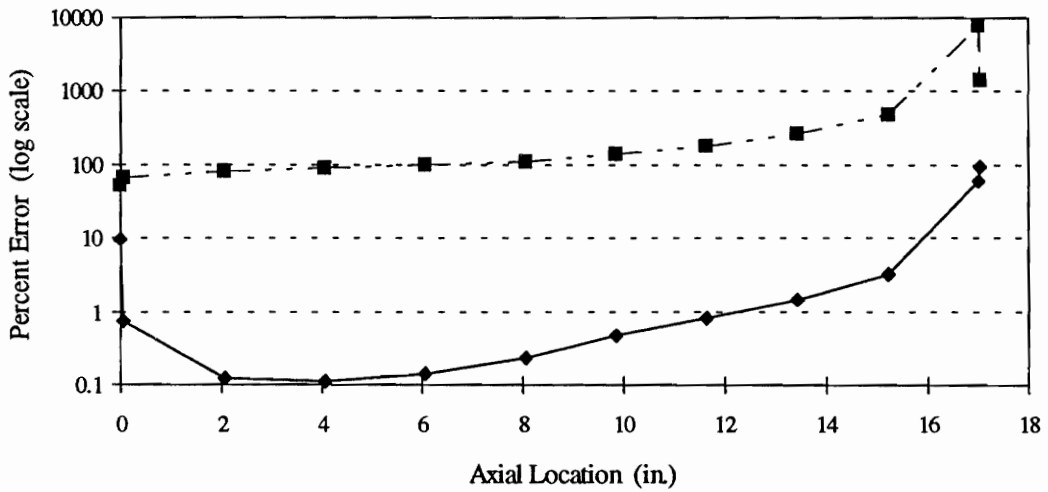


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 33b. Percent Error of the Real Component of the Estimated Pressure, Case 14.**  
 (Two Different Starting Positions for One-Percent Uncertainty)

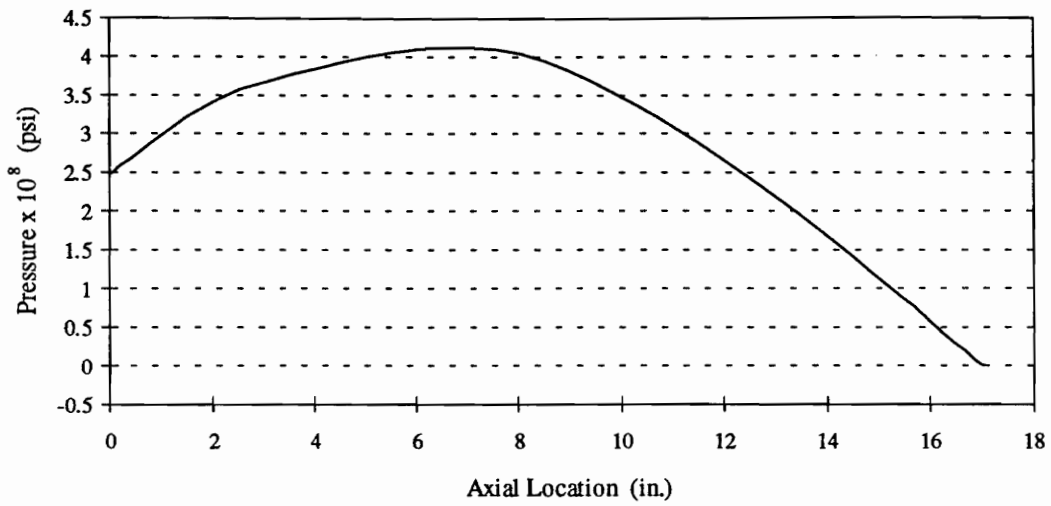


**Figure 33c. The Imaginary Component of the Target Pressure, Case 14.**

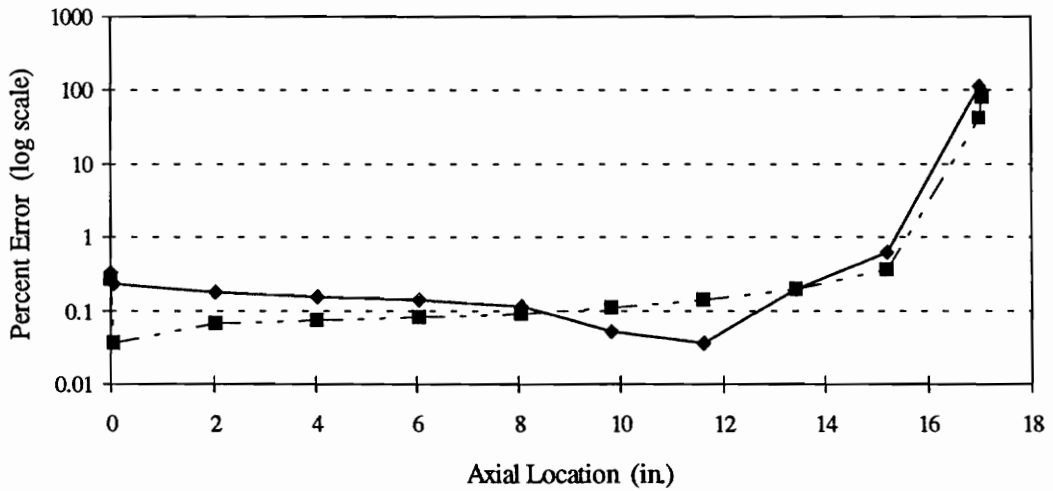


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 33d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 14.**  
 (Two Different Starting Positions for One-Percent Uncertainty)

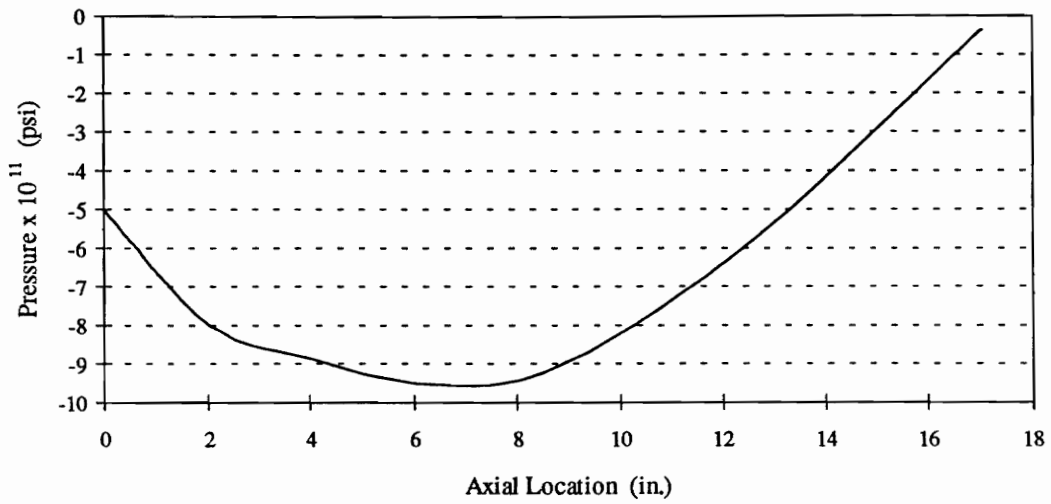


**Figure 34a. The Real Component of the Target Pressure, Case 15.**

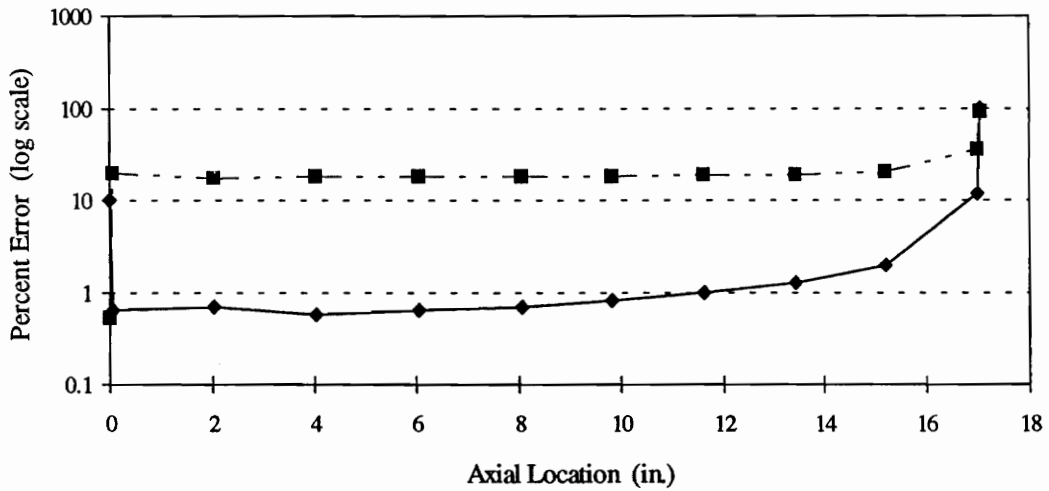


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 34b. Percent Error of the Real Component of the Estimated Pressure, Case 15.**  
(Two Different Starting Positions for One-Percent Uncertainty)

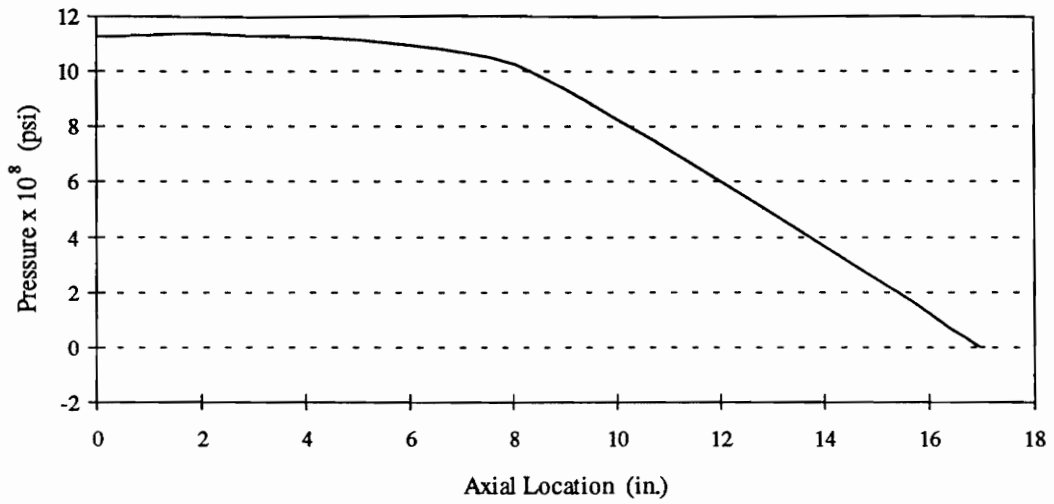


**Figure 34c. The Imaginary Component of the Target Pressure, Case 15.**

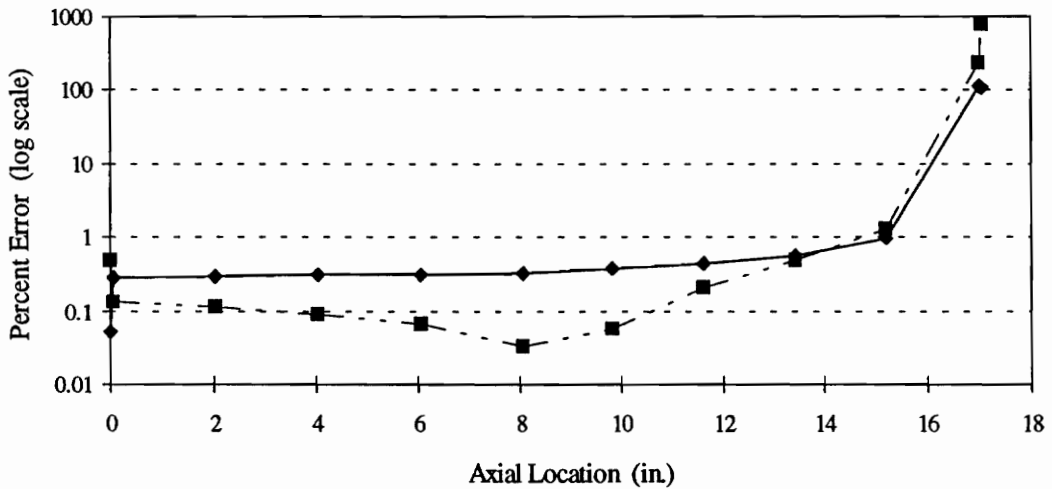


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 34d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 15.**  
 (Two Different Starting Positions for One-Percent Uncertainty)

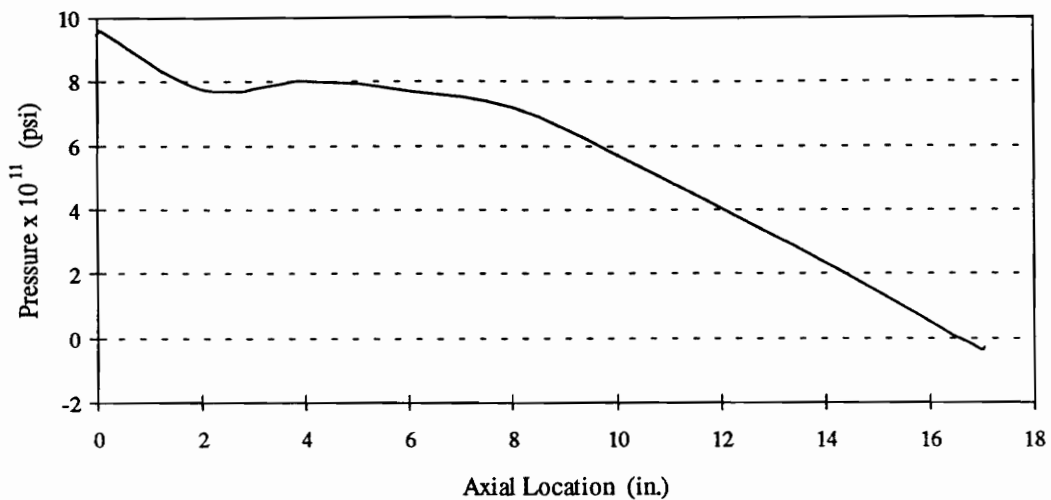


**Figure 35a. The Real Component of the Target Pressure, Case 16.**

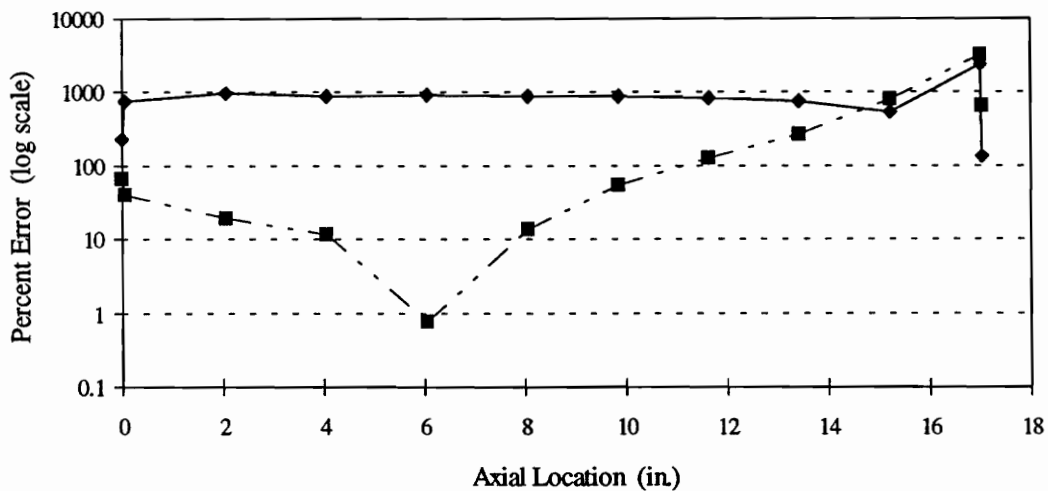


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 35b. Percent Error of the Real Component of the Estimated Pressure, Case 16.**  
 (Two Different Starting Positions for One-Percent Uncertainty)



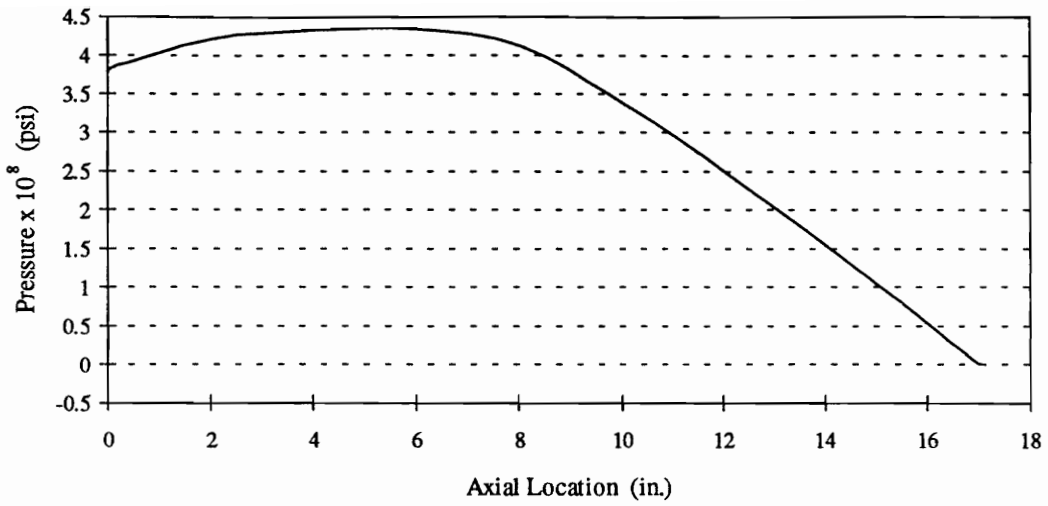
**Figure 35c. The Imaginary Component of the Target Pressure, Case 16.**



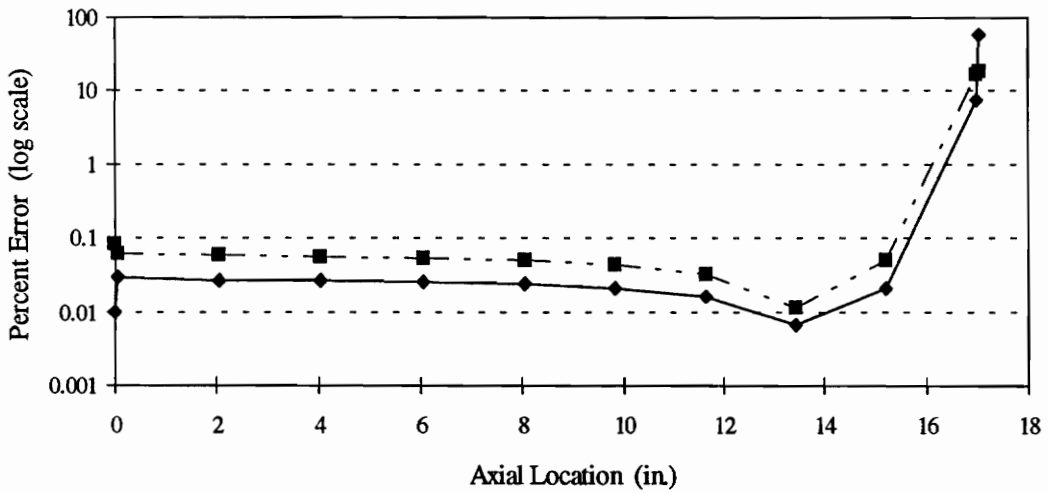
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 35d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 16.**  
 (Two Different Starting Positions for One-Percent Uncertainty)



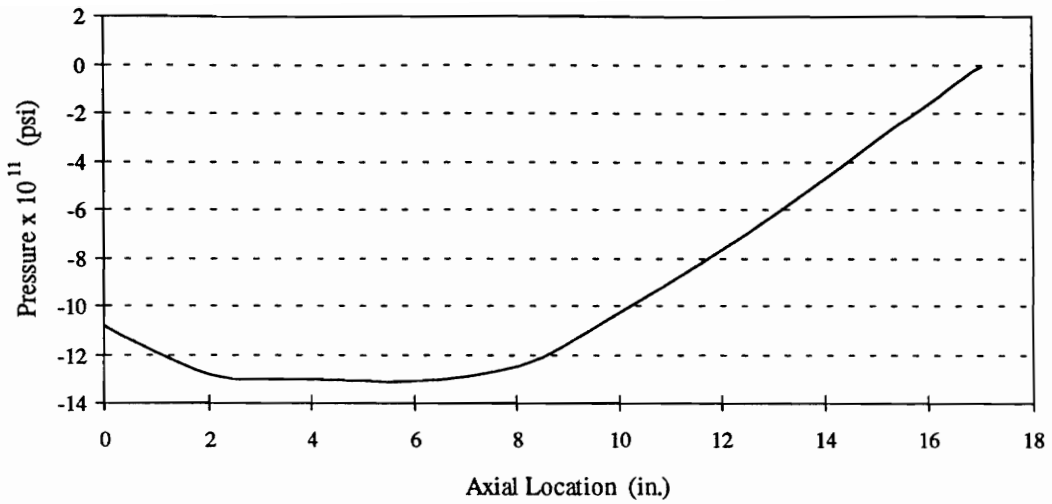


**Figure 36a. The Real Component of the Target Pressure, Case 17.**

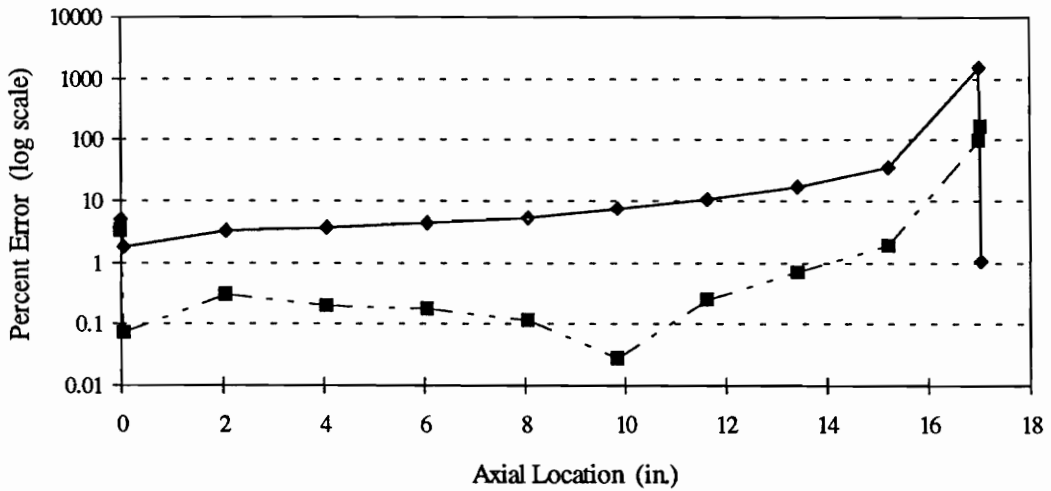


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 36b. Percent Error of the Real Component of the Estimated Pressure, Case 17.**  
(Two Different Starting Positions for One-Percent Uncertainty)

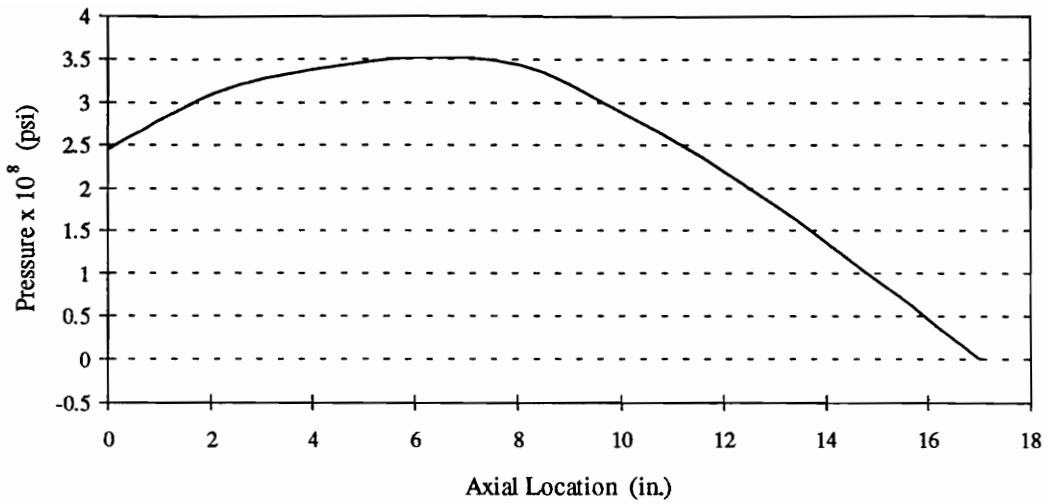


**Figure 36c. The Imaginary Component of the Target Pressure, Case 17.**

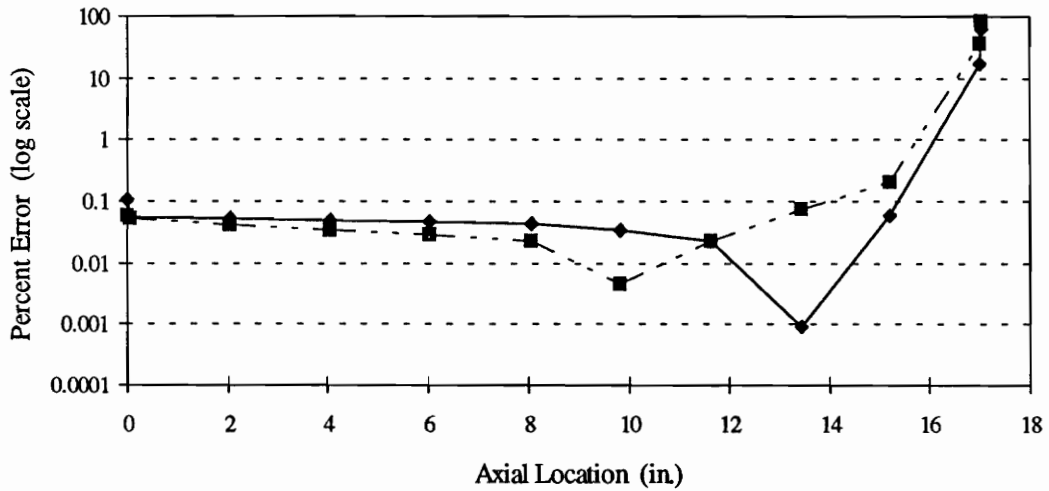


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 36d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 17.**  
 (Two Different Starting Positions for One-Percent Uncertainty)

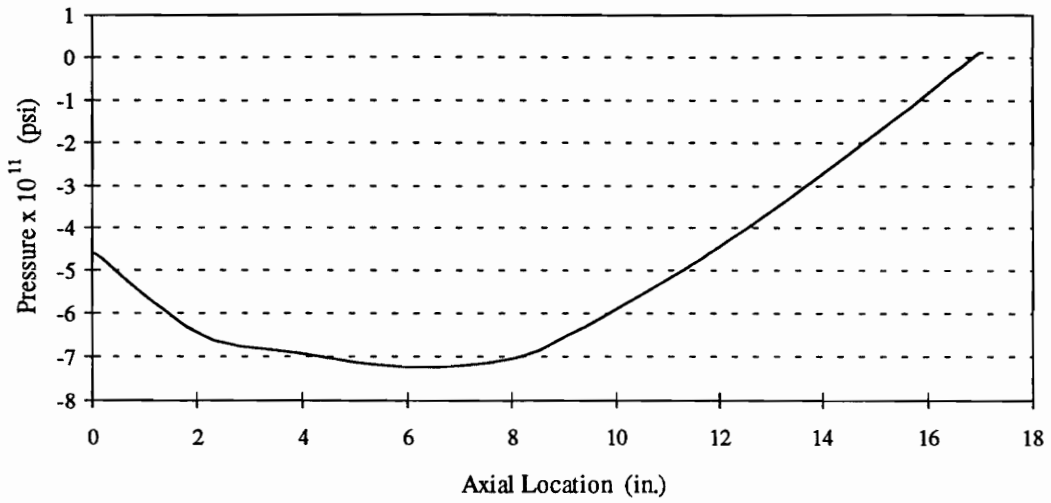


**Figure 37a. The Real Component of the Target Pressure, Case 18.**

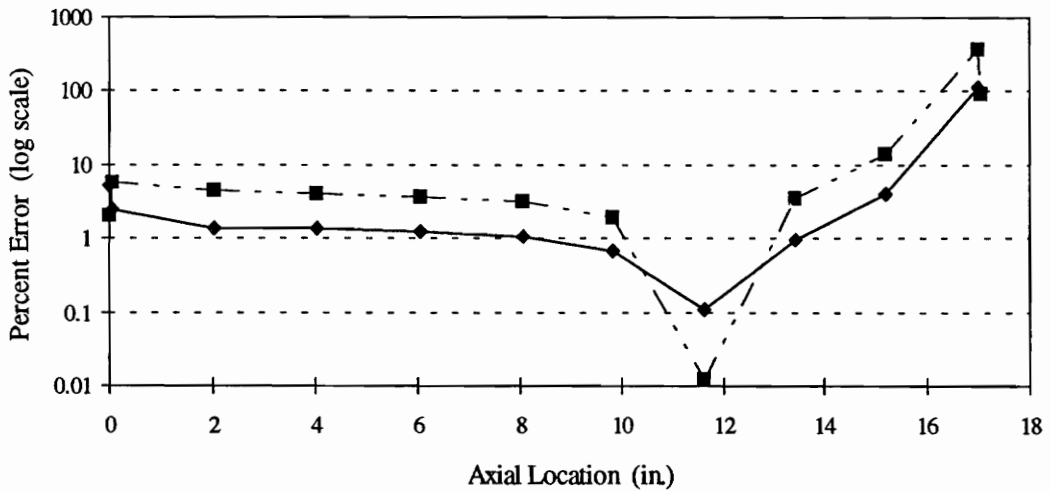


$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 37b. Percent Error of the Real Component of the Estimated Pressure, Case 18.**  
 (Two Different Starting Positions for One-Percent Uncertainty)

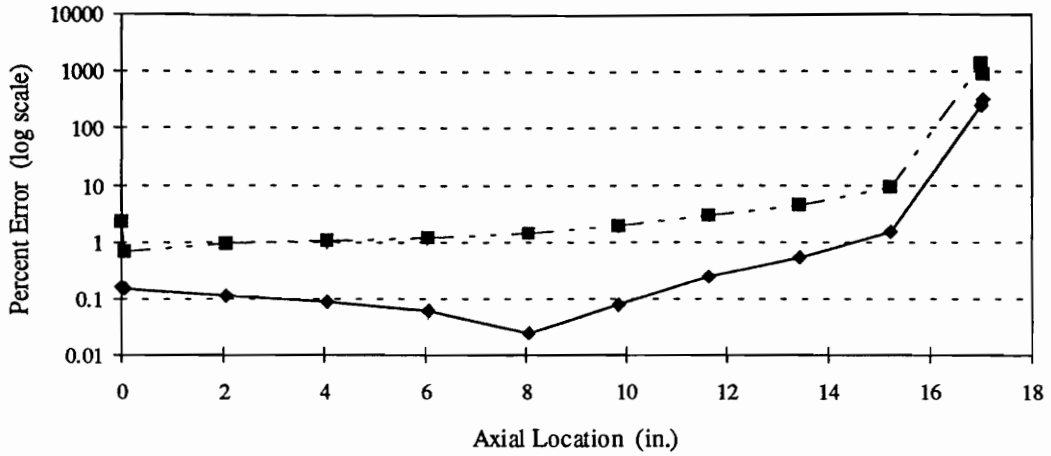


**Figure 37c. The Imaginary Component of the Target Pressure, Case 18.**



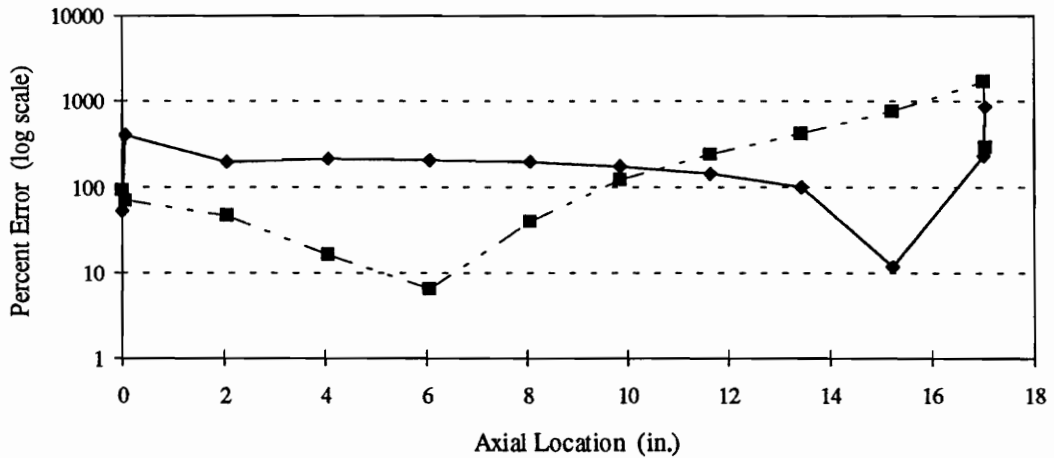
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 37d. Percent Error of the Imaginary Component of the Estimated Pressure, Case 18.**  
 (Two Different Starting Positions for One-Percent Uncertainty)



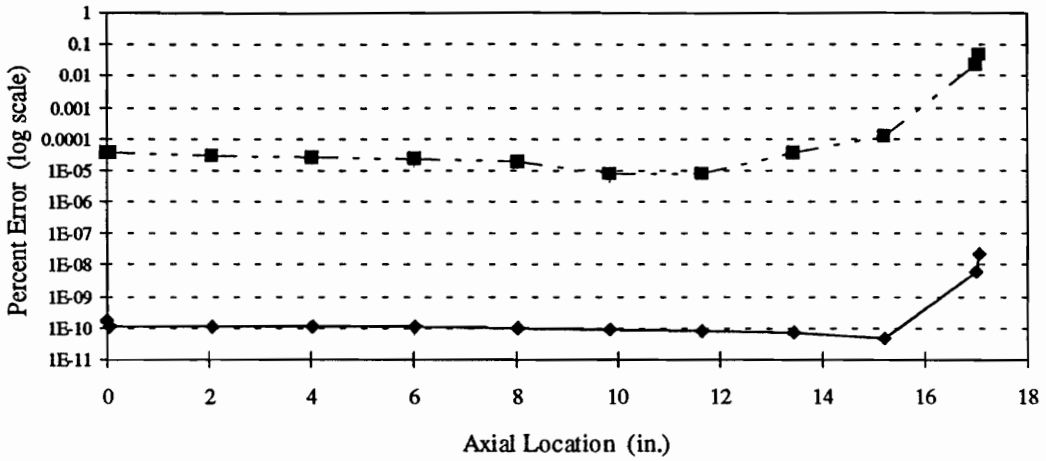
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 38a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 10.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



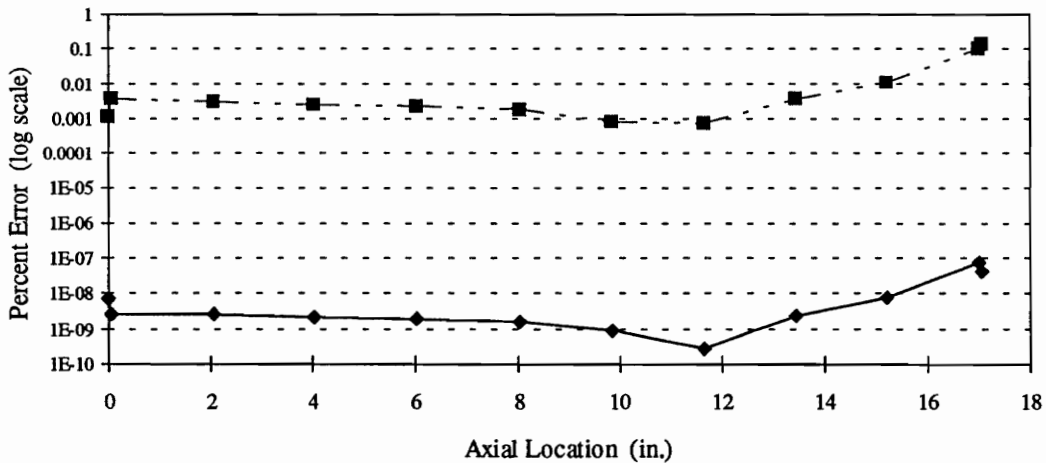
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 38b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 10.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



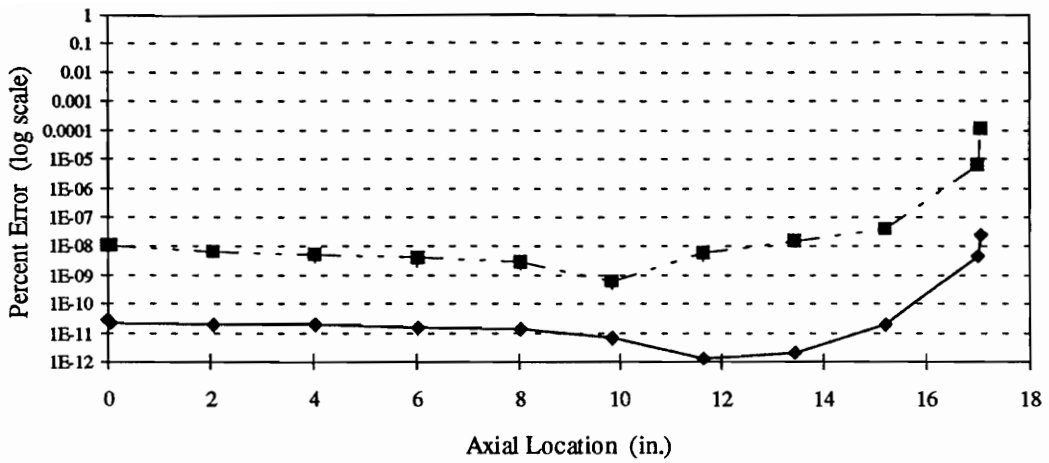
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 39a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 11.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



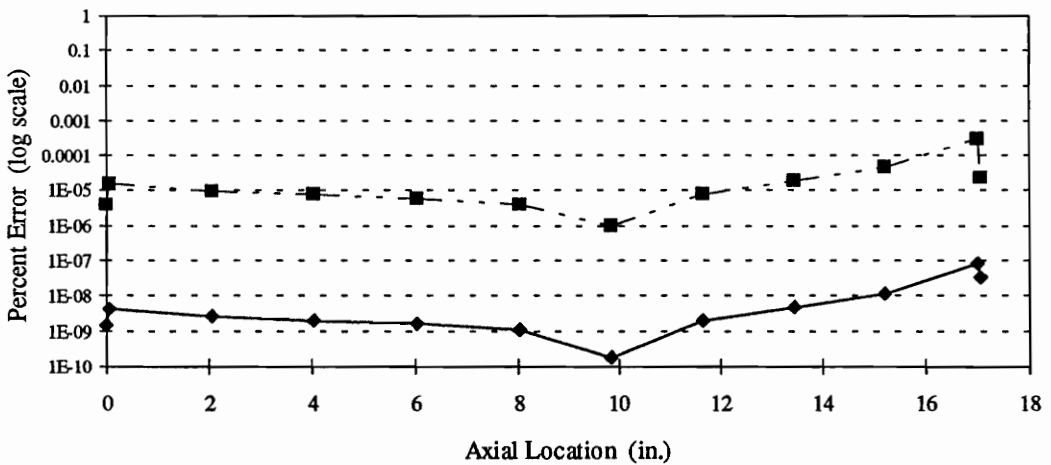
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 39b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 11.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



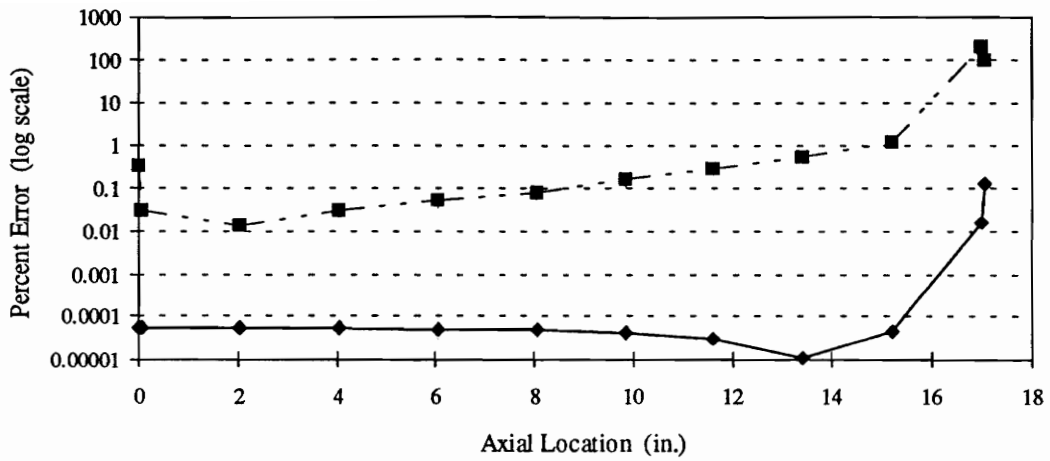
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 40a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 12.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



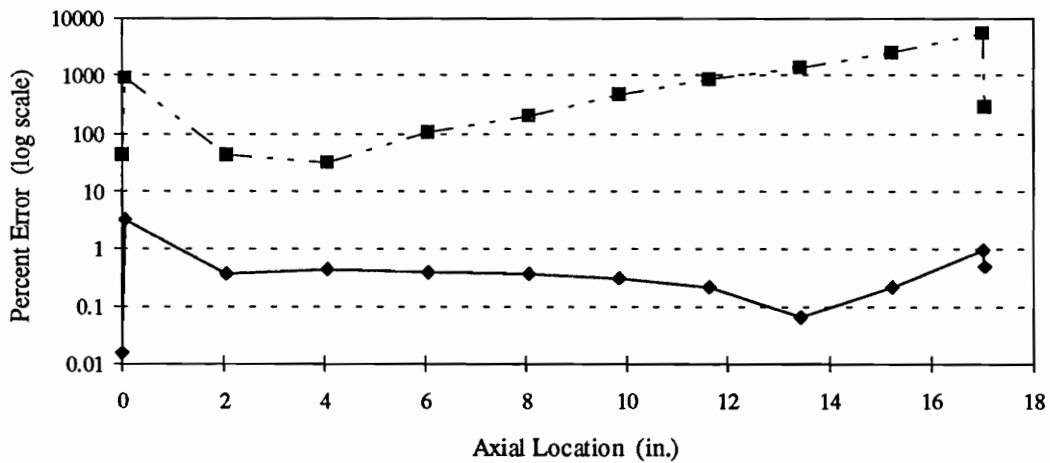
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 40b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 12.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

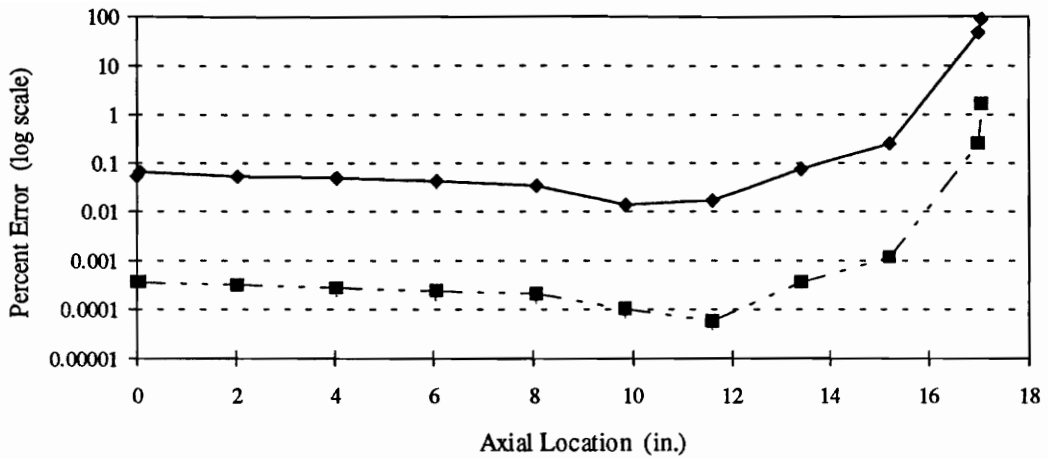
**Figure 41a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 13.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

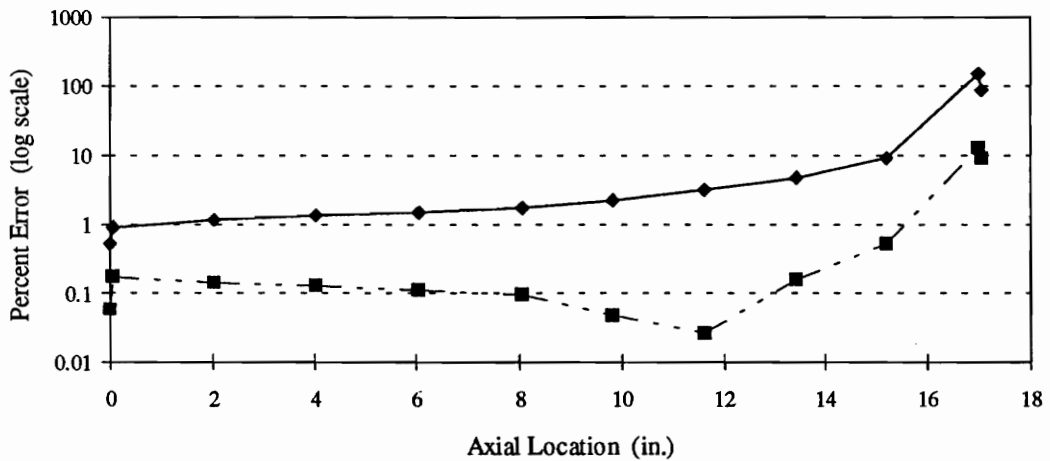
**Figure 41b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 13.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)





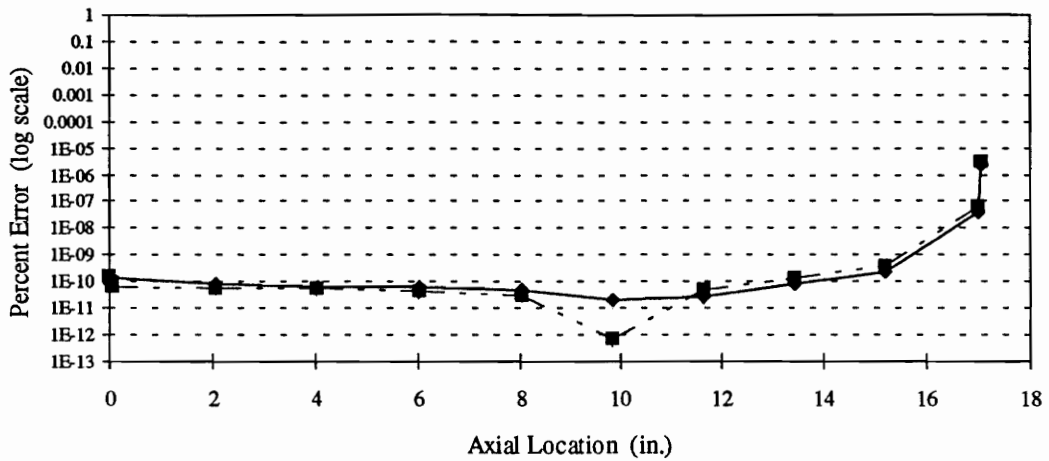
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 42a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 14.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



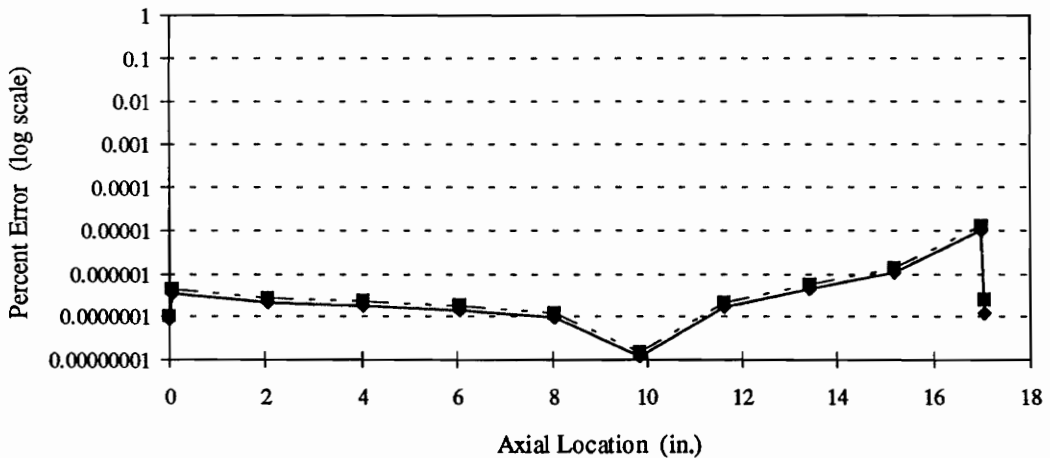
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 42b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 14.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



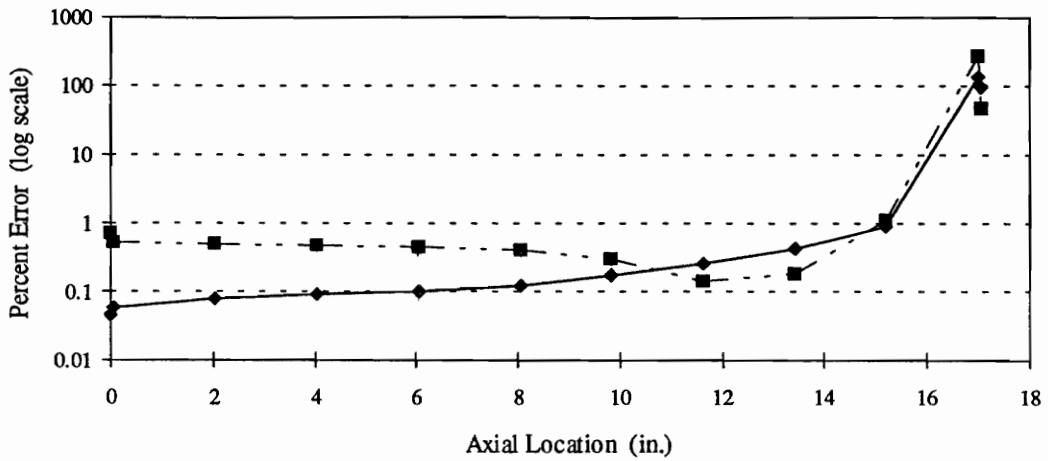
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 43a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 15.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



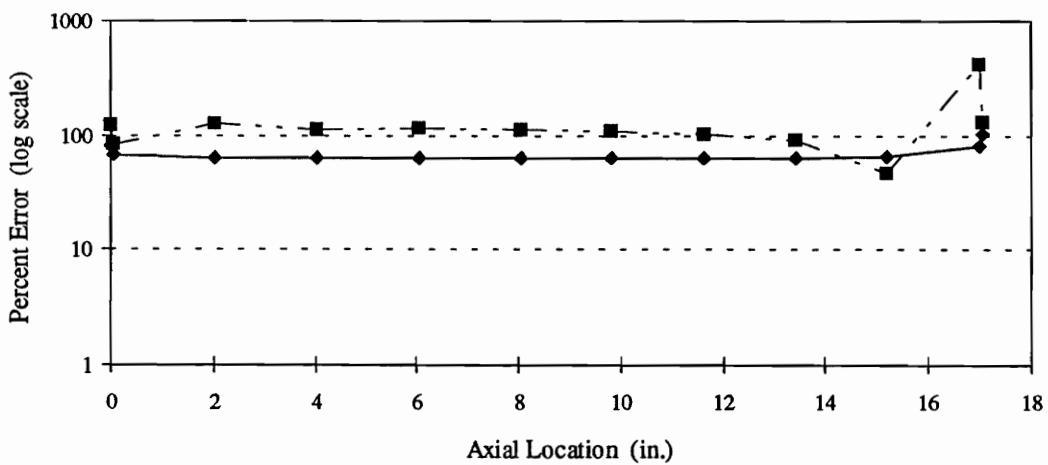
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 43b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 15.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



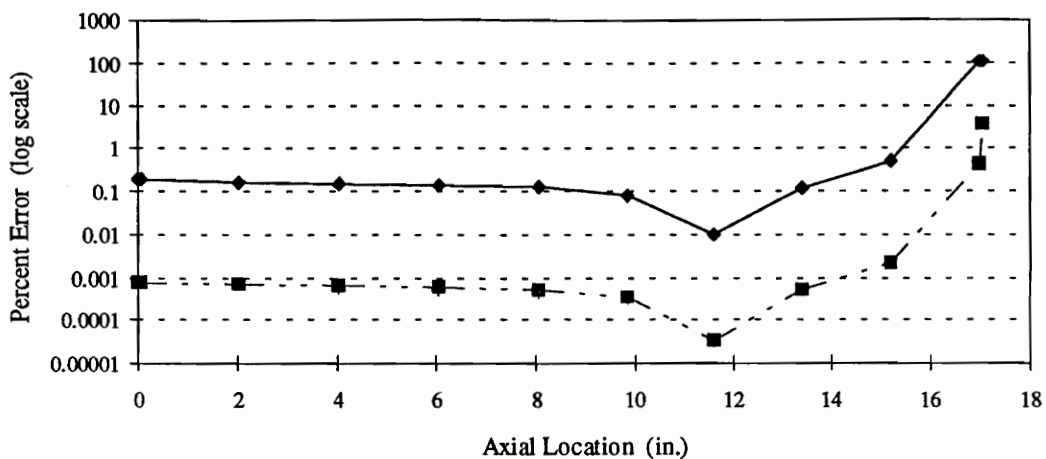
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 44a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 16.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



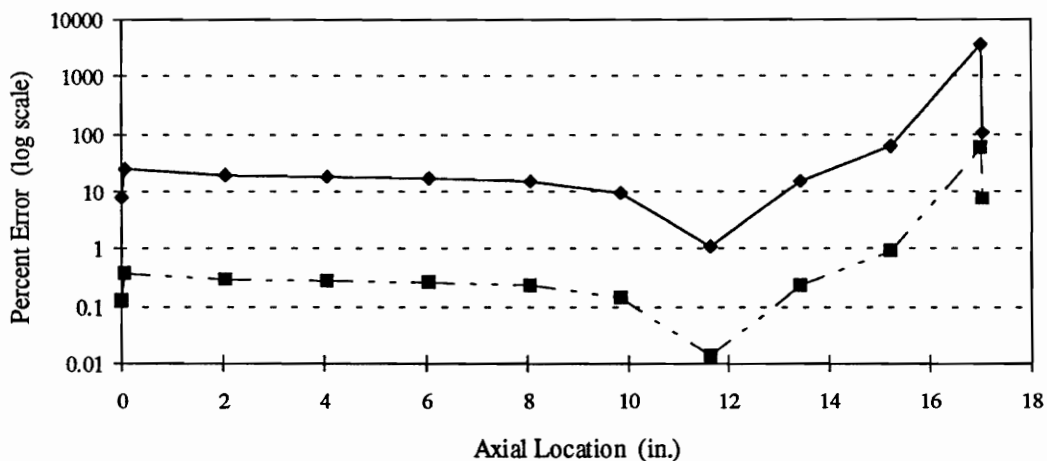
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 44b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 16.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



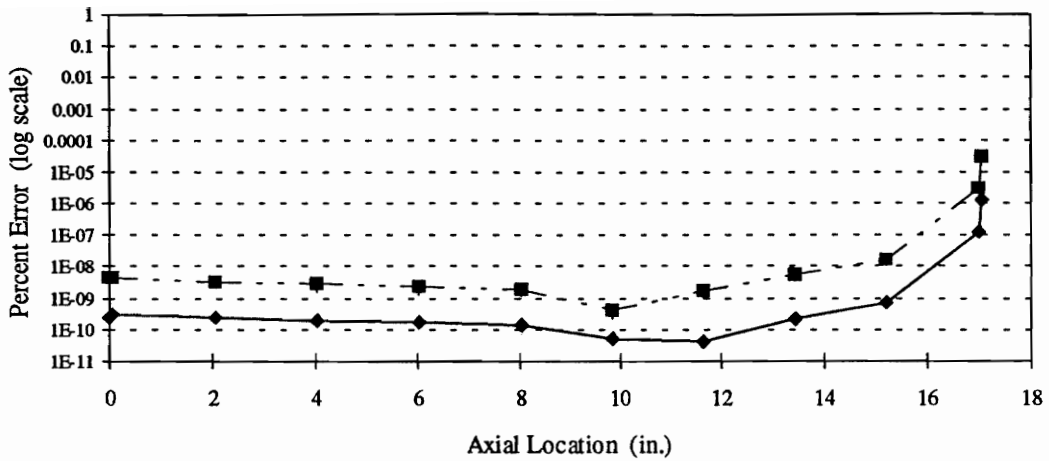
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 45a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 17.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



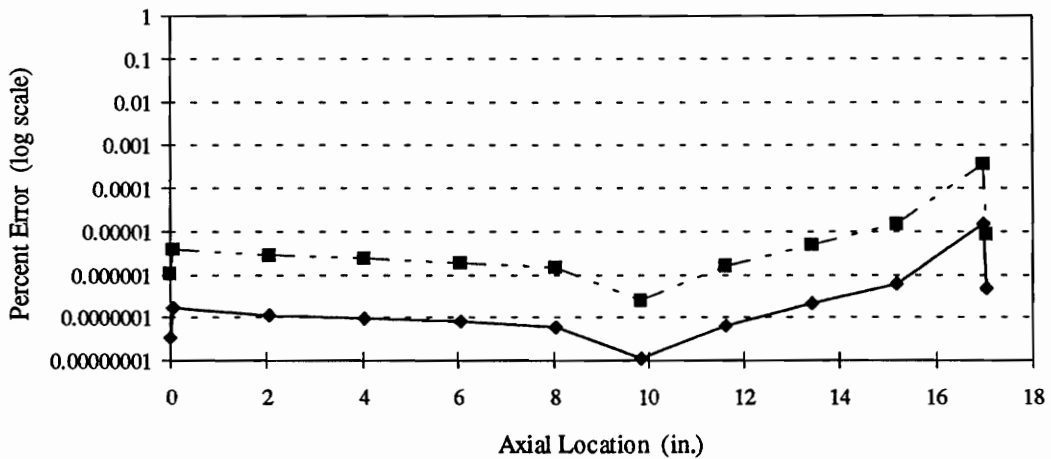
$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 45b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 17.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 46a. Percent Error of the Real Component of the Estimated Pressure, Ideal Case 18.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)



$$\text{Percent Error} = \left| \frac{Y_i - P_i}{Y_i} \right| \times 100\%$$

**Figure 46b. Percent Error of the Imaginary Component of the Estimated Pressure, Ideal Case 18.**  
(Two Different Starting Positions for Zero-Percent Uncertainty)

## References

1. Lord Rayleigh, The Theory of Sound, Vol. 2, reprinted by Dover Publications, Inc., New York, 1894/1945.
2. Putnam, A. A., Combustion Driven Oscillations in Industry, Elsevier, 1971.
3. Baade, P. K., "Design Criteria and Models for Preventing Combustion Oscillations," Paper presented at ASHRAE Symposium on Combustion Driven Oscillations, Atlanta, Georgia, January-February 1978.
4. Richards, G. A., M. J. Yip, E. Robey, L. Cowell, and D. Rawlins, "Combustion Oscillation Control by Cyclic Fuel Injection," Paper presented at the ASME Turbo Expo Conference, Houston, Texas, June 1995.
5. Levine, H. And J. Schwinger, "On the Radiation of Sound from an Unflanged Circular Pipe," Physical Review, Vol. 73, no. 4, pp 383-406, 1948.
6. Carrier, G. F., "Sound Transmission from a Tube with Flow," Quarterly of Applied Mathematics, Vol. 13, pp 453-461, 1956.
7. Ando, Y., "On the Sound Radiation from Semi-Infinite Circular Pipe of Certain Wall Thickness," Acustica, Vol. 22, pp 219-225, 1969/1970.
8. Johnston, G. W., and K. Ogimoto, "Sound Radiation from a Finite Length Unflanged Circular Duct with Uniform Axial Flow, Part I: Theoretical Analysis," Journal of the Acoustical Society of America, Vol. 68, pp 1858-1870, 1980.
9. Johnston, G. W., and K. Ogimoto, "Sound Radiation from a Finite Length Unflanged Circular Duct with Uniform Axial Flow, Part II: Computed Radiation Characteristics," Journal of the Acoustical Society of America, Vol. 68, pp 1871-1883, 1980.

10. Cummings, A., "High Temperature Effects on the Radiation Impedance of an Unflanged Duct Exit," Journal of Sound and Vibration, Vol. 52, no. 2, pp 299-304, 1977.
11. Mahan, J. R., J. G. Cline, and J. D. Jones, "A Temperature Correlation for the Radiation Resistance of a Thick-Walled Circular Duct Exhausting Hot Flow," Journal of the Acoustical Society of America, Vol 75, pp 63-71, 1984.
12. Cline, J. G., The Effects of Nonisentropic Flow on the Acoustic Radiation Impedance of an Unflanged Circular Tube, M.S. Thesis, Virginia Polytechnic Institute and State University, March, 1980.
13. Kinsler, L. E., A. R. Frey, A. B. Copens, and J. V. Sanders, Fundamentals of Acoustics, Third Edition, J. Wiley Publishers, 1982.
14. Seybert, A. F., and D. F. Ross, "Experimental Determination of Acoustical Properties Using Two-Microphone Random Excitation Technique," Journal of the Acoustical Society of America, Vol. 61, pp 1362-1370, 1977.
15. Chung, J. Y., and D. A. Blaser, "Transfer Function Method of Measuring In-Duct Acoustical Properties, Part I: Theory," Journal of the Acoustical Society of America, Vol. 68, pp 907-921, 1980.
16. Chung, J. Y., and D. A. Blaser, "Transfer Function Method of Measuring In-Duct Acoustical Properties, Part II: Experiment," Journal of the Acoustical Society of America, Vol. 68, pp 907-921, 1980.
17. Howard, R. E., Acoustical Behavior of a Turbulent, Ducted, Premixed, Hydrogen-Flame Burner, M.S. Thesis, Virginia Polytechnic Institute and State University, August, 1985.
18. Beck, J. V., and K. J. Arnold, Parameter Estimation in Engineering and Science, J. Wiley Publishers, New York, 1977.
19. Legendre, A. M., Nouvelles Méthodes Pour la Détermination des Orbites des Comètes, Paris, 1806.
20. Gauss, K. F., Theory of the Motion of Heavenly Bodies Moving about the Sun in Conic Sections, reprinted by Dover Publications, Inc., New York, 1809/1963.

21. Box, G. E. P., and H. Kanemasu, "Topics in Model Building, Part II, On Non-Linear Least Squares," Technical Report No. 321, University of Wisconsin, Dept. Of Statistics, Madison, Wis., Nov. 1972.
22. Bard, Y., Nonlinear Parameter Estimation, Academic Press, Inc., New York, 1974.
23. Heitkoetter, J., and D. Beasley, "The Hitchhiker's Guide to Evolutionary Computation," USENET: comp.ai.genetic.
24. Holland, J. H., "Concerning efficient Adaptive Systems," In M. C. Yovits, G. T. Jacobi, and G. D. Goldstein (Eds.), Self-organizing Systems, (pp.215-230), Spartan Books, Washington, 1962.
25. Holland, J. H., "Information Processing in Adaptive Systems," Information Processing in the Nervous System, Proceedings of the International Union of Physiological Sciences, Vol. 3, pp 330-339., 1962.
26. Holland, J. H., "Outline for a Logical Theory of Adaptive Systems," Journal of the Association for Computing Machinery, Vol. 9, no 3., pp 297-314, 1962.
27. Holland, J. H., Adaptation in Natural and Artificial Systems, The University of Michigan Press, Ann Arbor, Michigan, 1975.
28. DeJong, K. A., An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Doctoral Dissertation, University of Michigan, 1975.
29. Goldberg, D. E., Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Co., Inc., Reading, Massachusetts, 1989.
30. Davis, L. (Ed.), Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, 1991.
31. Mahan, J. R., "Recovery of Acoustical Boundary Conditions in Gas Turbine Combustors from Dynamic Pressure Measurements Using a Finite-Element-Based Modal Analysis Code," Research Proposal Submitted to United States Department of Energy Morgantown Energy Technology Center, December 1994.
32. Bathe, K. J., Finite Element Procedures in Engineering Analysis, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.



33. Tira, N. E., Dynamic simulation of Solar Calibration of the Total Earth Viewing Channel of the Earth Radiation Budget Experiment (ERBE), Master of Science Thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, December 1987.
34. Reddy, J. N., An Introduction to the Finite Element Method, McGraw-Hill Book Company, New York, 1984.

## Appendix A

Contained here is a derivation of the finite element model used by Program DYNAMODE and Subroutine DYNAMITE. It is adapted, nearly verbatim, from a derivation written by Dr. J. R. Mahan [31] in 1994 as part of a research proposal.

### A.1 Theoretical Development

The nonhomogeneous acoustic wave equation describing the acoustic pressure field  $p(x,y,z,t)$  (lb/ft<sup>2</sup>) due to a point heat source distribution  $q(x,y,z,t)$  (ft·lb/lb<sub>m</sub>) is

$$\frac{1}{a_o^2} \frac{\partial^2 p}{\partial t^2} - \nabla^2 p = \frac{\rho_o (\gamma - 1)}{a_o^2} \frac{\partial^2 q}{\partial t^2}, \quad (\text{A1})$$

where  $a_o$  (ft/s) is the steady-flow speed of sound,  $\rho_o$  (lb<sub>m</sub>/ft<sup>3</sup>) is the steady-flow mass density, and  $\gamma$  is the specific heat ratio,  $c_p/c_v$ . The derivation of equation A1 assumes that the fluid is an ideal gas undergoing an isentropic process. Consistent with this assumption,

$$a_o = \sqrt{\gamma \mathcal{R} \mathcal{T}_o g_c}, \quad (\text{A2})$$

where  $\mathcal{R}$  (ft·lb/lb<sub>m</sub>·R) is the gas constant of the medium,  $\mathcal{T}_o$  (R) is the local steady-flow absolute temperature, and  $g_c$  (=32.2 ft·lb<sub>m</sub>/lb<sub>f</sub>·s<sup>2</sup>) is the required dimensional constant; and

$$\rho_o = \frac{\mathcal{P}_o}{\mathcal{R} \mathcal{T}_o}, \quad (\text{A3})$$

where  $\mathcal{P}_o$  (lb/ft<sup>2</sup>) is the local steady-flow pressure. Due to the high temperatures and large temperature variations in a gas turbine combustor, temperature-dependent specific heats should be used in computing the local specific heat ratio  $\gamma$ . Equation A1 also

assumes that  $p/P_o \ll 1.0$ , i.e. small perturbations are assumed. Equation A1 can be used to find the pressure field within an enclosure, assuming that the acoustic impedance conditions are known at the walls. The finite-element method (FEM) is used to solve equation A1.

## **A.2 Solution Using the Finite-Element Method**

The development is begun by multiplying equation A1 by a test function  $\lambda$  and then integrating over a volume element in the physical  $x,y,z$  coordinate system,

$$\int_V \left( \frac{\lambda}{a_o^2} \frac{\partial^2 p}{\partial t^2} - \lambda \frac{\partial^2 p}{\partial x^2} - \lambda \frac{\partial^2 p}{\partial y^2} - \lambda \frac{\partial^2 p}{\partial z^2} - \frac{\lambda \rho_o (\gamma - 1)}{a_o^2} \frac{\partial^2 q}{\partial t^2} \right) dx dy dz = 0. \quad (A4)$$

Note that the analysis is being developed in rectangular coordinates even though the geometry of a cylindrical burner would seem to require development in cylindrical coordinates. This is because the solution space will be approximated using six-sided prismatic volume elements which are capable of being mapped into a rectangular master element. The analysis will be formulated on the basis of the physical  $x,y,z$  coordinate system, and then at the appropriate time the analysis will be mapped into the  $\xi,\eta,\zeta$  coordinates of the master element. This mapping allows the FEM to conform to irregular geometries.

The analysis is converted to the frequency domain by assuming that the acoustic pressure field can be represented by

$$p(x,y,z,t) = P(x,y,z)e^{j\omega t}, \quad (A5)$$

where  $j = \sqrt{-1}$ . This separation of variables leads to a solution for the mode structure in the burner; that is, the local magnitude of  $P(x,y,z)$  of the acoustic pressure corresponding to a source distribution  $Q(x,y,z)$ , where

$$q(x,y,z,t) = Q(x,y,z)e^{j\omega t}, \quad (A6)$$

Subject to this assumed form of the solution, the first term of the integrand in equation A13 may be written

$$\int_{\mathcal{V}} \frac{-\omega^2 \lambda}{a_o^2} \mathcal{P} e^{j\omega t} d\chi dy dz, \quad (\text{A7})$$

and the last term of the integrand may be written

$$\int_{\mathcal{V}} \frac{\omega^2 \lambda \rho_o (\gamma - 1)}{a_o^2} Q e^{j\omega t} d\chi dy dz, \quad (\text{A8})$$

Focusing on the middle three terms of the integrand in equation A3, application of the identity

$$g \frac{\partial f}{\partial \chi} = -f \frac{\partial g}{\partial \chi} + \frac{\partial (fg)}{\partial \chi}, \quad (\text{A9})$$

where  $f = \lambda$  and  $g = \partial \mathcal{P} / \partial \chi$  (where  $\chi$  represents  $x$ ,  $y$ , or  $z$ ), yields for these terms

$$\int_{\mathcal{V}} \left( \frac{\partial \lambda}{\partial \chi} \frac{\partial \mathcal{P}}{\partial \chi} + \frac{\partial \lambda}{\partial y} \frac{\partial \mathcal{P}}{\partial y} + \frac{\partial \lambda}{\partial z} \frac{\partial \mathcal{P}}{\partial z} \right) e^{j\omega t} d\chi dy dz, \quad (\text{A10a})$$

$$- \int_{\mathcal{V}} \left[ \frac{\partial}{\partial \chi} \left( \lambda \frac{\partial \mathcal{P}}{\partial \chi} \right) + \frac{\partial}{\partial y} \left( \lambda \frac{\partial \mathcal{P}}{\partial y} \right) + \frac{\partial}{\partial z} \left( \lambda \frac{\partial \mathcal{P}}{\partial z} \right) \right] e^{j\omega t} d\chi dy dz. \quad (\text{A10b})$$

Note that all of the terms of equation A4, when rewritten using expressions A7, A8, and A10, contain  $e^{j\omega t}$  as a factor. Thus, this common factor can be divided out of equation A4.

Note that in the first integral of equation A10 the differentiation has been spread equally over the test function  $\lambda$  and the dependent variable  $\mathcal{P}$  so that the differentiation of the dependent variable has been reduced to first order. However, the second integral in the expression still contains second-order derivatives. The normal continuation in FEM at this point is to apply the divergence theorem to the second volume integral in equation A10 to convert it to a surface integral containing only first derivatives. But taking this approach in the case of the acoustic wave equation misrepresents the acoustic boundary conditions in the analysis. However, for reasons explained below, the order of differentiation of the second integral of equation A10 still needs to be reduced.

The approach, which is believed to be a new contribution to the FEM applied to acoustic analysis, is to introduce Euler's equation,

$$\nabla p = -\rho_o(\partial \mathbf{u} / \partial t), \quad (\text{A11})$$

where  $\mathbf{u}(\chi, y, z, t) = \mathbf{U}(\chi, y, z)e^{j\omega t}$  is the acoustic particle velocity. With the introduction of the acoustic admittance,  $\mathbf{y} = \mathbf{u}/p = \mathbf{U}/\mathcal{P}$ , Euler's equation can be written

$$\nabla \mathcal{P} = -j\omega \rho_o \mathbf{y} \mathcal{P} e^{j\omega t}, \quad (\text{A12})$$

where it is noted that equation A12 is a vector equation; that is, like  $\nabla \mathcal{P}$ ,  $\mathbf{y}$  is a vector quantity having components in the  $\chi$ ,  $y$ , and  $z$  directions. Introducing equation A12 into the second integral of expression A10 (from which the factor  $e^{j\omega t}$  has been dropped) yields

$$j\omega \rho_o \int_{\nu} \left( y_\chi \frac{\partial(\lambda \mathcal{P})}{\partial \chi} + y_y \frac{\partial(\lambda \mathcal{P})}{\partial y} + y_z \frac{\partial(\lambda \mathcal{P})}{\partial z} \right) d\chi dy dz, \quad (\text{A13})$$

where the components of the acoustic admittance,  $y_\chi$ ,  $y_y$ , and  $y_z$ , are treated as constant coefficients under the differentiation. *These admittances represent the acoustic boundary conditions and so have nonzero values only at bounding surfaces of the burner.* Finally, it is recognized that the acoustic admittance is a complex number,

$$\mathbf{y} = \text{Re}\{\mathbf{y}\} + j \text{Im}\{\mathbf{y}\}. \quad (\text{A14})$$

This completes the reduction of order of differentiation while correctly including the acoustic boundary conditions.

It should be noted that, other than geometry, here lies the major difference between DYNAMODE and DYNAMITE. In DYNAMODE, the real, or dissipative, component of the acoustic admittance is neglected, in which case equation A13 becomes

$$-\omega \rho_o \int_{\nu} \left( \text{Im}\{y_\chi\} \frac{\partial(\lambda \mathcal{P})}{\partial \chi} + \text{Im}\{y_y\} \frac{\partial(\lambda \mathcal{P})}{\partial y} + \text{Im}\{y_z\} \frac{\partial(\lambda \mathcal{P})}{\partial z} \right) d\chi dy dz. \quad (\text{A13b})$$

In DYNAMITE, the real part of the admittance is not neglected, leaving the imaginary  $j$  term in the formulation. To accommodate this, the admittance defined by the user must be multiplied by  $j$

$$\dot{\mathbf{y}} = -\text{Im}\{\mathbf{y}\} + j \text{Re}\{\mathbf{y}\} \quad (\text{A14b})$$

before it is inserted into the analysis.

As stated above, the problem is formulated in the physical  $x,y,z$  coordinate system and then mapped onto a rectangular master element. The master element and related nomenclature are shown in Figure A1. It is assumed that the dependent variable  $\mathcal{P}(\xi,\eta,\zeta)$  may be represented in the master element by the approximate function

$$\mathcal{P}(\xi, \eta, \zeta) = \sum_{j=1}^{\mathcal{N}} \mathcal{P}_j \phi_j(\xi, \eta, \zeta), \quad (\text{A15})$$

where  $\mathcal{N}$  is the number of nodes per element (in this case 8) and the  $\phi_j(\xi,\eta,\zeta)$  are basis functions. In the current analysis the  $\phi_j$  are represented as linear functions:

$$\phi_1 = \frac{1}{8}(1-\xi)(1-\eta)(1-\zeta), \quad (\text{A16})$$

$$\phi_2 = \frac{1}{8}(1+\xi)(1-\eta)(1-\zeta), \quad (\text{A17})$$

$$\phi_3 = \frac{1}{8}(1+\xi)(1+\eta)(1-\zeta), \quad (\text{A18})$$

$$\phi_4 = \frac{1}{8}(1-\xi)(1+\eta)(1-\zeta), \quad (\text{A19})$$

$$\phi_5 = \frac{1}{8}(1-\xi)(1-\eta)(1+\zeta), \quad (\text{A20})$$

$$\phi_6 = \frac{1}{8}(1+\xi)(1-\eta)(1+\zeta), \quad (\text{A21})$$

$$\phi_7 = \frac{1}{8}(1+\xi)(1+\eta)(1+\zeta), \quad (\text{A22})$$

$$\phi_8 = \frac{1}{8}(1-\xi)(1+\eta)(1+\zeta). \quad (\text{A23})$$

Equations 26 through 33 meet the requirements that: (1)  $\mathcal{P}$  and its first partial derivatives can be evaluated everywhere in the master element, and (2)  $\phi_i = 1$  at node  $i$  of the element. This shows clearly why it was important to reduce the order of derivatives in the formulation.

The test function  $\lambda$  is arbitrary as long as it can conform to the boundary conditions. It is convenient and usual to use the same polynomial functions to represent  $\lambda$  as those used for  $\mathcal{P}$ . When expressions A7, A8, A10, A13, and A15 are combined and introduced into equation A14 there results

$$[\mathcal{K}_y + \mathcal{M}_y] \mathcal{P}_j = \mathcal{F}_r \quad (\text{A24})$$

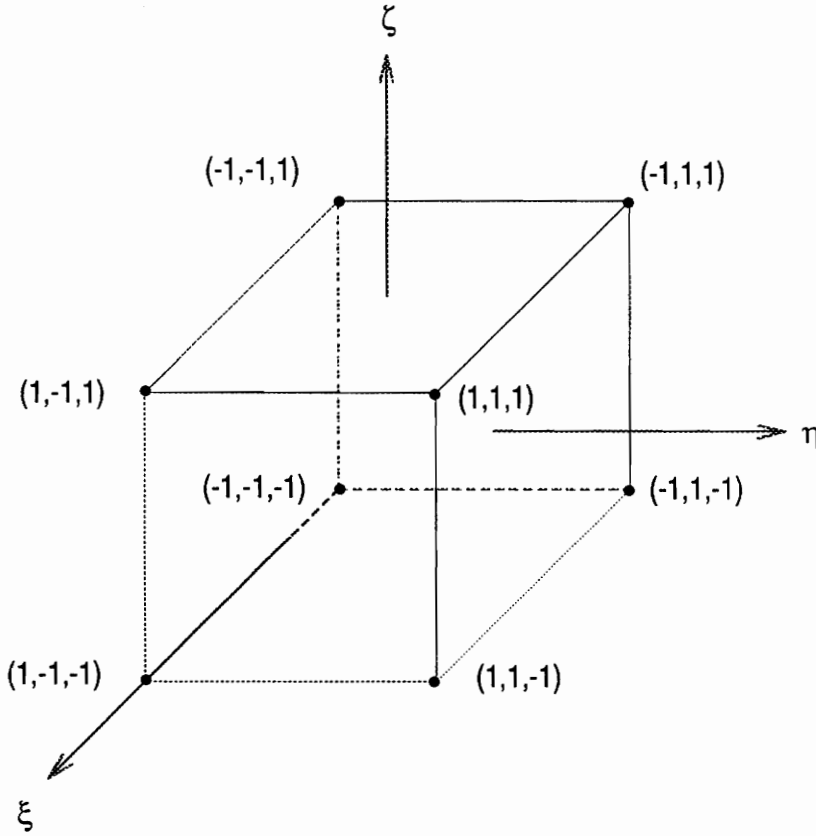
where

$$\mathcal{K}_{ij} = \int_{\nu} \left( \frac{\partial \phi_i}{\partial \chi} \frac{\partial \phi_j}{\partial \chi} + \frac{\partial \phi_i}{\partial y} \frac{\partial \phi_j}{\partial y} + \frac{\partial \phi_i}{\partial z} \frac{\partial \phi_j}{\partial z} \right) d\chi dy dz, \quad (\text{A25})$$

$$\mathcal{M}_{ij} = j\omega \rho_o (y_x S_{ij}^x + y_y S_{ij}^y + y_z S_{ij}^z), \quad (\text{A26})$$

and

$$\mathcal{F}_i = -\kappa^2 \rho_o (\gamma - 1) Q \int_{\nu} \phi_i d\chi dy dz. \quad (\text{A27})$$



**Figure A1.** The Master Element and its Coordinate System.

In Equations A25 and A27 the quantity  $\kappa = \omega/a_0$  is the wavenumber, and in equation A26

$$S_{ij}^x = \int_{\nu} \left( \phi_i \frac{\partial \phi_j}{\partial x} + \phi_j \frac{\partial \phi_i}{\partial x} \right) dx dy dz, \quad (\text{A28})$$

$$S_{ij}^y = \int_{\nu} \left( \phi_i \frac{\partial \phi_j}{\partial y} + \phi_j \frac{\partial \phi_i}{\partial y} \right) dx dy dz, \quad (\text{A28})$$

and

$$S_{ij}^z = \int_{\nu} \left( \phi_i \frac{\partial \phi_j}{\partial z} + \phi_j \frac{\partial \phi_i}{\partial z} \right) dx dy dz. \quad (\text{A28})$$

Note that a version of equation A13 exists for every element in the problem domain. These elements must be assembled to form an approximation of the physical space. When this is done a new version of A24 exists for which the  $ij$  subscripts now refer to the total number of nodes in the finite element model (mesh). In the current version of the model of a can-type gas turbine combustor, symmetry can be used to model only a  $180^\circ$  sector of the combustor. Still, the half-combustor model contains 1416 elements and 1662 nodes. This means that, using standard matrix techniques, a  $1662 \times 1662$  matrix must be manipulated and stored. To avoid this problem the  $\mathcal{K}_y + \mathcal{M}_y$  matrix is converted to banded form, resulting in a  $1662 \times \text{NHBW}$  matrix, where NHBW is the narrow half-bandwidth of the banded matrix. The value of NHBW can be minimized by judicious numbering of the nodes, and in the current study has a value of 203. Subroutine INTER3 and Subroutine SOLVE, which put matrix  $\mathcal{K}_y + \mathcal{M}_y$  in banded form and solve equation B24, are based on Subroutine INTER from Noura Tira's M.S. Thesis [33] and Subroutine SOLVE from Reddy's [34] book on the finite element method.

The next step in the analysis is to evaluate the volume integrals in equations A25, A27, A28, A29, and A30 and assemble the results to obtain the matrices  $\mathcal{K}_y$  and  $\mathcal{M}_y$  and the vector  $\mathcal{F}_i$  in the assembled version of equation A24. However, before proceeding it is useful to discuss the physical significance of these quantities. The eigenvalues of the matrix  $\mathcal{K}_y$  are the resonant frequencies of the burner for the special case of a hard-walled



burner (all admittances  $y$  equal to zero.) Thus, in the special case of a hard-walled burner, the assembled version of equation A24 becomes

$$[\mathcal{K}_y]\mathcal{P}_j = \mathcal{F}_j \quad (\text{A31})$$

and its solution gives the three-dimensional acoustic pressure mode shapes in the hard-walled burner. Similarly, the eigenvalues of the matrix  $\mathcal{K}_y + \mathcal{M}_y$  are the resonant frequencies of the burner with specified acoustic wall admittances. The vector  $\mathcal{F}_j$  represents the “forcing function” to which the combustor responds.

The careful reader will have already recognized that the integrals in equations A25, A27, A28, A29, and A30 need work before they can be evaluated because they are written in terms of two different coordinate systems: the physical  $x, y, z$  coordinate system describing the burner geometry, and the  $\xi, \eta, \zeta$  coordinate system of the master element. Clearly one of these systems must be expressed in terms of the other before the integrals can be evaluated. Transformations between coordinate systems are normally accomplished using the Jacobian matrix,

$$[j] = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix}. \quad (\text{A32})$$

Then,

$$dx dy dz = |j| d\xi d\eta d\zeta, \quad (\text{A33})$$

where  $|j|$  is the determinant of the Jacobian matrix, and similarly

$$\begin{bmatrix} \frac{\partial \phi_i}{\partial x} \\ \frac{\partial \phi_i}{\partial y} \\ \frac{\partial \phi_i}{\partial z} \end{bmatrix} = \begin{bmatrix} \frac{\partial \xi}{\partial x} & \frac{\partial \eta}{\partial x} & \frac{\partial \zeta}{\partial x} \\ \frac{\partial \xi}{\partial y} & \frac{\partial \eta}{\partial y} & \frac{\partial \zeta}{\partial y} \\ \frac{\partial \xi}{\partial z} & \frac{\partial \eta}{\partial z} & \frac{\partial \zeta}{\partial z} \end{bmatrix} \begin{bmatrix} \frac{\partial \phi_i}{\partial \xi} \\ \frac{\partial \phi_i}{\partial \eta} \\ \frac{\partial \phi_i}{\partial \zeta} \end{bmatrix}, \quad (\text{A34})$$

where the matrix is the inverse of the Jacobian matrix,  $|j|^{-1}$ . Equations A33 and A34 are used to eliminate  $x$ ,  $y$ , and  $z$  from the integrals in equations A25, A27, A28, A29, and

A30, after which the integrals can be evaluated. For example, equation A28 can now be written

$$\begin{aligned}
 S_{ij}^x &= \int_{-1}^1 \int_{-1}^1 \int_{-1}^1 \left( \phi_i \frac{\partial \phi_j}{\partial \xi} + \phi_j \frac{\partial \phi_i}{\partial \xi} \right) g_{11}^{-1} |g| d\xi d\eta d\zeta \\
 &+ \int_{-1}^1 \int_{-1}^1 \int_{-1}^1 \left( \phi_i \frac{\partial \phi_j}{\partial \eta} + \phi_j \frac{\partial \phi_i}{\partial \eta} \right) g_{12}^{-1} |g| d\xi d\eta d\zeta \\
 &+ \int_{-1}^1 \int_{-1}^1 \int_{-1}^1 \left( \phi_i \frac{\partial \phi_j}{\partial \zeta} + \phi_j \frac{\partial \phi_i}{\partial \zeta} \right) g_{13}^{-1} |g| d\xi d\eta d\zeta.
 \end{aligned} \tag{A35}$$

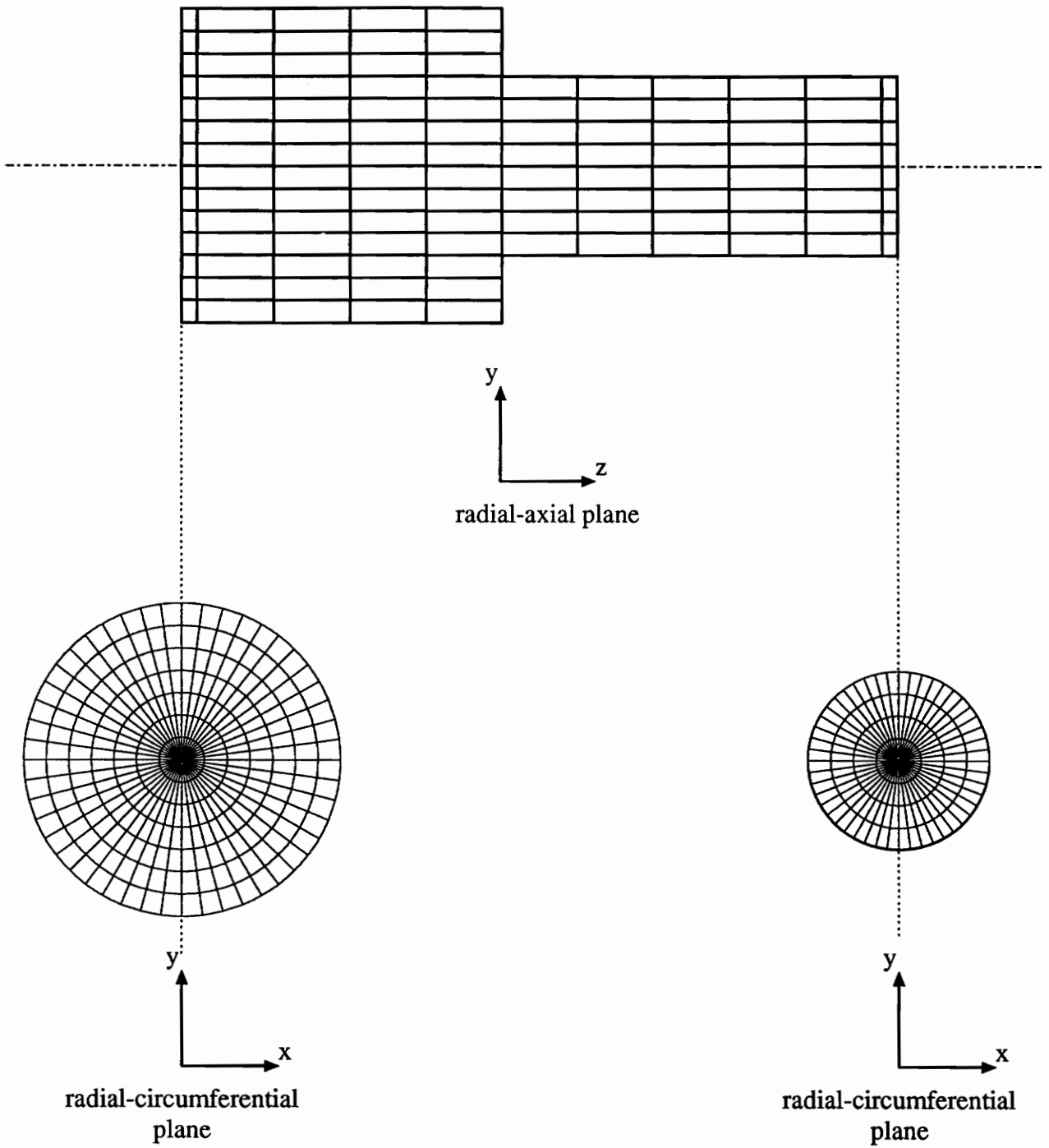
The actual integrations are carried out using Gauss quadrature. In Gauss quadrature the value of an integral like those in equation A35 is approximated as

$$\int_{-1}^1 \int_{-1}^1 \int_{-1}^1 f(\xi, \eta, \zeta) d\xi d\eta d\zeta = \sum_{k=1}^{\mathcal{N}_{GP}} f(\xi_k, \eta_k, \zeta_k) \mathcal{W}_k, \tag{A36}$$

where the  $\xi_k, \eta_k, \zeta_k$  are the Gauss points,  $\mathcal{N}_{GP}$  is the number of Gauss points, and the  $\mathcal{W}_k$  are the corresponding weights. The Gauss points and weights for various situations can be found in [34].

### **A.3 Finite Element Description of the Combustor Geometry**

Central to FEM is the discretization of the problem domain. In the current study, a mesh of nodes and six-sided “brick” elements is used to describe the combustion chamber. The geometry is defined using a combination of: (1) the number of divisions (elements) in the axial, radial, and circumferential directions, (2) the coordinate definition of the nodal points, and (3) the arrangement of the elements in the mesh. Figure A2 shows a radial-axial plane and two radial-circumferential planes of the FEM mesh used for the METC can-type burner. To model other geometries, one must change the mesh definition (as well as other parameters), and this process is described in Appendix B.



**Figure A2. FEM Grid for METC Burner**

## **Appendix B. User's Guide to Program ESTIMAT**

### **B.1 Changing the Burner Geometry**

Currently, Program ESTIMAT is set up for the METC can-type combustor geometry. Other types of burners, such as a straight-can or an annular geometry, can be modeled by simply altering the FEM mesh definition. Below is a list of changes that are required alter the geometry in ESTIMAT. The line numbers alluded to in these changes correspond to the line numbers of the documented code in Appendix B.3. Also, explanations of the variables used in ESTIMAT are given in the documentation.

#### List of Alterations for Changing the Burner Geometry in ESTIMAT:

1. The user must change the value of the user-defined variables in lines 1916 through 1949 of ESTIMAT. These variables include IFULL, NEM, NTHETA, NR, NZ, NQ1, NQ2, N1, N2, N3, N4, R(), AND ZPLANE(). Also, the assignment of the NGEOM() array to geometric variables in lines 1969-1972 and 1977-1988 needs attention.
2. The sizes of arrays in variable declaration in Subprogram DYNAMITE must be changed according to the new geometry. If computing space is not a restraint, the user may want to oversize these arrays and then never worry about them again. However, if the computing system won't handle oversized arrays, then the

declarations can be divided into two types depending upon nodal or elemental variables. The following variables need an array the size of which is at least as big as the number of elements: X, Y, Z, GAMMA, RHO, WN, QQ, MC. The following variables need an array the size of which is at least as big as the number of nodes: XX, YY, ZZ, GF, GK, P, Q, T. The user must be sure to correctly declare variables in all of the subroutines where they are used. Also, be aware that the NHBW may change due to a change in the mesh, and GK must be dimensioned to handle it!

3. Lines 2085 through 2112 are used to assign the initial values of P, Q, and T to the nodes. The method of looping through the nodes is important here; a change in geometry may change this looping process.

4. Subroutine COORDS located in lines 2448 through 2558 defines the x,y,z coordinates of the nodes. The same looping process used in step 3 can often be employed here. The task here is to come up with a simple logic for assigning the nodal coordinates.

5. Subroutine CONNECT located in lines 2571 through 2755 defines how the elements are connected in the global FEM mesh. Judicious numbering and arrangement of the elements can minimize the NHBW and reduce computing time. An extremely good way to become familiar with the process of constructing the connectivity array is to go through a few examples by hand.

6. Finally, redefining the geometry will usually require the location of the pressure observations (for the parameter estimation) to be changed. The definition of these locations is located in lines 2418 through 2431.

## **B.2 Other Changes**

Other desired modifications may include: changing the stopping criteria of the estimation process, changing the number of experimental observations, changing the number of parameters to estimate, and other specific alterations to the estimation processes.

The stopping criteria of the Gaussian estimation can be changed in Subroutine STOPPER (lines 660 through 691.) These criteria include the number of iterations from a particular starting position, and the definition of negligible change in any estimated parameter.

Changing the number of experimental observations requires many modifications to the code. Obviously variables such as Y, PRESS, ETA and, EETTAA must be declared accordingly. But one must also be sure that the matrix operations and variable declarations in subroutines GAUSS, SENSTV, PARAM, LSTSQR, GENETIC reflect any changes.

Changing the number of parameters to be estimated is similar to changing the number of experimental observations. Variables must be declared correctly and matrix operations must be changed in the main program as well as in subroutines GAUSS, STOPPER, SENSTV, PARAM, INVERT, GENETIC, AND DYNAMITE.

Specific changes to the estimation process, other than those described, can also be implemented by the user. There are many possibilities for change, ranging from a new population size in the genetic algorithm, to a different numerical differentiation technique in the Gauss linearization, to a different formulation for the specific heat in DYNAMITE.

### B.3 Documentation of ESTIMAT

```

PROGRAM ESTIMAT

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
5  c    This code was written by Andrew D. Wright while he was c
c    a student at Virginia Polytechnic Institute and state University. c
c    It was written as part of a research project that was conducted c
c    with the United States Department of Energy, Morgantown c
c    Energy Technology Center. c
10 c    ESTIMAT can be used to determine the acoustic boundary c
c    conditions of a combustion system. The input to ESTIMAT is ac
c    target pressure vector, which contains the real and imaginary c
c    parts of the acoustic pressure obtained from four locations c
c    during combustor operation. The user also must specify the c
15 c    burner geometry in subroutine DYNAMITE, and parameters c
c    such as heat release, temperature, and mean pressure must be c
c    provided. c
c    ESTIMAT is written to perform both theoretical and c
c    experimental analyses. If so desired, one can input a set of c
20 c    known boundary conditions, and the code will produce a target c
c    pressure vector and then try to recover the original boundary c
c    conditions. Or one can just input a target pressure vector, and c
c    let the code go from there. c
c    Output from the code is written to a data file, the name of c
25 c    which can be specified in the subroutine POSTPRO. c
c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c                                Nomenclature c
30 c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c
c    Integer Quantities: c
35 c
c    I, J    counters used in loops c
c    K      number of iterations from a starting point during c
c          Gauss linearization estimation. c
c    COL    column number used in inversion of matrix c
40 c    CHANGE index used in ranking of objective function c
c          values in genetic algorithm c
c    CHROME counter used to track the parameter number c
c          in the genetic algorithm c
c    ELITIST index used in the creation of the initial random c
45 c          in the GA; specifies which seed to use. c
c    FN     output file number used in psot-processing c
c    GEN    generation number in genetic algorithm (GA) c

```

c	KEEPNUM	number of individuals if GA that will be directly	c
c		passed into next generation	c
50	MAXGEN	maximum number of generations in the GA	c
c	MAXCOL	number of columns in matrix to invert	c
c	MAXROW	number of rows in matrix to invert	c
c	MAXRUN	maximum number of random restarts in the	c
c		first Gauss linearization process	c
55	MINK	stored value of K where lowest sum-of-squares error	c
c		occurs during the Gauss linearization (correct signs)	c
c	NEWRANK	temporary storage location used in ranking the	c
c		individuals of the GA	c
c	NCOL	column counter in matrix inversion	c
60	NROW	row counter used in matrix inversion	c
c	NUM	index used to loop through individuals in many parts of GA	c
c	NUMETA	number of pressure observations, used in GA	c
c	PARENT1	index for the first individual chosen for mating to	c
c		create a single child in the next generation of the GA	c
65	PARENT2	index for the second individual chosen for mating to	c
c		create a single child in the next generation of the GA	c
c	POPNUM	size of population (# of individuals) in GA	c
c	RANK	rank of individual in GA according to error function	c
c	ROW	row index used in matrix inversion	c
70	RUNNUM	counter for the number of random restarts that occur	c
c		during the first Gauss estimation	c
c	SAMEFLAG	index used to indicate that two individuals are	c
c		identical while evaluating sum-of-squares error	c
c		in the GA	c
75	SEED1, SEED2	seeds sent to the random number generator	c
c	SENTRY	index used to stop Gauss estimation; SENTRY is	c
c		assigned a value of 1 if stopping criteria are met	c
c	SET	counter used in ranking individuals in GA	c
80	SIGNFLAG1	index equals 1 if the signs of the target and estimate	c
c		pressures don't match before parameter modification	c
c	SIGNFLAG2	index equals 1 if the signs of the target and estimate	c
c		pressures don't match after parameter modification	c
c	STCL	starting column for row-reduction in matrix inversion	c
c	STRW	starting row for row-reduction in matrix inversion	c
85	XXMINK	stored value of K where lowest sum-of-square error	c
c		occurs during the Gauss linearization (wrong signs)	c
c			c
c	Real Quantities		c
c			c
90	BMINUSA	range of pressure about the mean value in which	c
c		a perturbation must be (lbf/in <sup>2</sup> )	c
c	DEL	stopping criteria for negligible change in parameter	c
c		in the Gauss estimation (ft <sup>3</sup> /lbf*s)	c
c	DEL1	differential number to avoid division by zero	c
95	DELB	differential change in a parameter value (ft <sup>3</sup> /lbf*s)	c
c	ERROR1	percent error in the real part of either pressure	c
c		or admittance, used in post-processing	c
c	ERROR2	percent error in the imaginary part of either pressure	c
c		or admittance, used in post-processing	c
100	CHECKER	value to compare against DEL (ft <sup>3</sup> /lbf*s)	c
c	MEAN	mean pressure value used to determine the	c
c		perturbed value (lbf/in <sup>2</sup> )	c



	c	MINLSE	the lowest sum-of-square error function value of c	
	c		the entire generation in the GA (lbf/in2) <sup>2</sup>	c
105	c	MUTATION	the mutation rate (out of 100) in the GA	c
	c	NEWNUM	number used in determining perturbed pressure	c
	c	ORIGNAL	temporary storage during numerical differentiation	c
	c		(ft <sup>3</sup> /lbf*s)	c
	c	RANDOM	random number obtained by calling RANMAR	c
110	c	RANMAR	random number generating function; generates a	c
	c		uniformly distributed number between 0 and 1	c
	c	RANSUM	number used to perturb pressure target	c
	c	REALSEED	number used to automatically send different seeds	c
	c		to the random number generator	c
115	c	RESMAG	magnitude of estimated pressure at a probe location	c
	c		used in post-processing (lbf/in2)	c
	c	SMIN	lowest sum-of-squares error value for pressure	c
	c		vector with correct signs in Gaussian estimation	c
	c		(lbf/in2) <sup>2</sup>	c
120	c	XXSMIN	lowest sum-of-squares error value for pressure	c
	c		vector with incorrect signs in Gaussian estimation	c
	c		(lbf/in2) <sup>2</sup>	c
	c	S	sum-of-squares error (lbf/in2) <sup>2</sup>	c
	c	SNEW	sum-of-squares error for new parameter estimates	c
125	c		of iteration K during the Gauss method (lbf/in2) <sup>2</sup>	c
	c	SOLD	sum-of-squares error for old parameter estimates	c
	c		of iteration K during the Gauss method (lbf/in2) <sup>2</sup>	c
	c	STOPLSE	acceptable stopping sum-of-squares error used	c
	c		as a stopping criterion in the GA (lbf/in2) <sup>2</sup>	c
130	c	SUM1	value used to determine probabilities of being	c
	c		chosen as a parent in the GA	c
	c	SUM2	value used to determine probabilities of being	c
	c		chosen as a parent in the GA	c
	c	TARMAG	magnitude of target pressure at a probe location	c
135	c		used in post-processing (lbf/in2)	c
	c	TOTALLSE	summation of all sum-of-square error of a gener-	c
	c		ation of individuals in the GA; used to determine	c
	c		the average error for the generation (lbf/in2) <sup>2</sup>	c
	c	TWOSIGMA	uncertainty on the mean value in terms of a part	c
140	c		of the mean (lbf/in2)	c
	c	UNCERTAIN	assumed uncertainty (5%=0.05) on the mean value	c
	c		used in perturbing the target pressure field	c
	c			c
	c	Vector Quantities		c
145	c			c
	c	AVGLSE(I)	average sum-of-square error for the generations	c
	c		of the GA (lbf/in2) <sup>2</sup>	c
	c	BB(I)	estimated acoustic admittances, (ft <sup>3</sup> /lbf*s)	c
	c		BB(1)=Re(Inlet Admittance)	c
150	c		BB(2)=Im(Inlet Admittance)	c
	c		BB(3)=Re(Exhaust Admittance)	c
	c		BB(4)=Im(Exhaust Admittance)	c
	c	BBNEW(I)	newly estimated admittances during iteration in the	c
	c		Gauss method (ft <sup>3</sup> /lbf*s)	c
155	c	ETA(I)	pressure field resulting from sending BB to sub-	c
	c		program dynamite, (lbf/in2)	c
	c		ETA(1)=Re(Pressure at location #1)	c

	c		ETA(2)=Im(Pressure at location #1)	c
	c		ETA(3)=Re(Pressure at location #2)	c
160	c		↓	c
	c		ETA(8)=Im(Pressure at location #4)	c
	c	ETANEW(I)	pressure field resulting from BBNEW (lbf/in2)	c
	c	DELETA(I)	pressure field used in numerical differentiation to	c
	c		determine the sensitivity matrix (lbf/in2)	c
165	c	FINAL(I)	final parameter estimates (ft3/lbf*s)	c
	c	HIGH(I)	limits used in creating the initial population in the	c
	c		GA; this is obtained by multiplying XXMINB(I)	c
	c		by 10.0 (ft3/lbf*s)	c
	c	INTERM(I)	parameter estimates [acosutic admittances] between	c
170	c		the steps of the GA and second Gauss estimation	c
	c		(ft3/lbf*s)	c
	c	LOW(I)	limits used in creating the initial population in the	c
	c		GA; this is obtained by multiplying MINB(I)	c
	c		by 10.0 (ft3/lbf*s)	c
175	c	LSE(I)	sum-of-square error for every individual in the	c
	c		current generation of the GA (lbf/in2)	c
	c	MINB(I)	parameter estimates [acosutic admittances] after the	c
	c		first Gauss estimation; these estimates give the	c
	c		correct signs on the pressures (ft3/lbf*s)	c
180	c	NEWCHILD(I)	parameter set used in ranking the individuals of the	c
	c		current GA generation (ft3/lbf*s)	c
	c	PRESS(I)	pressures resulting from FINAL (lbf/in2)	c
	c	PROB(I)	the probability of indiviula I in the current gener-	c
	c		ation of the GA to be chosen as a parent; summed	c
185	c		over all I, this variable equals 1.0	c
	c	PROBDIV(I)	different representation of PROB(I) used to	c
	c		fascilitate logical IF statement	c
	c	PROPS(I)	properties used by subroutine DYNAMITE;	c
	c		PROPS(1)=HeatRelease (BTU/lbm)	c
190	c		PROPS(2)=MeanPressure (lbf/in2)	c
	c		PROPS(3)=InletTemperature (°F)	c
	c		PROPS(4)=ExhaustTemperature (°F)	c
	c		PROPS(5)=Frequency (Hz)	c
	c	RNKLSE(I)	ranked sum-of-square error function for the current	c
195	c		generation; ranked from lowest to highest (lbf/in2) <sup>2</sup>	c
	c	TARGET(I)	Target pressure vector used in GA (same as Y(I))	c
	c	TARGETB(I)	Target admittances used in the theoretical	c
	c		analysis (ft3/lbf*s)	c
	c	TMPCHILD(I)	parameter set used in ranking the individuals of the	c
200	c		current GA generation (ft3/lbf*s)	c
	c	Y(I)	Target pressure vector (lbf/in2)	c
	c		Y(1)=Re(Target Pressure at location #1)	c
	c		Y(2)=Im(Target Pressure at location #1)	c
	c		Y(3)=Re(Target Pressure at location #2)	c
205	c		↓	c
	c		Y(8)=Im(Target Pressure at location #4)	c
	c	XXMINB(I)	parameter estimates [acosutic admittances] after the	c
	c		first Gauss estimation; these estimates give the	c
	c		incorrect signs on the pressures (ft3/lbf*s)	c
210	c			c
	c	Matrix Quantities		c
	c			c

```

c      A(I,J)      Matrix that gets inverted in INVERT      c
c      ANEW(I,J)   Form of A(I,J) in inversion process      c
215  c      CHECK(I,J) Matrix used to check that the matrix inversion      c
c      is working correctly. This matrix should be the      c
c      equal to the identity matrix.                        c
c      CHILD(I,J)  Individual I consisting of parameters (J1, J2, J3, J4)      c
c      in the current generation of the GA (ft3/lbf*s)      c
220  c      ELITE(I,J) Storage location for MINB(I) and XXMINB(I)      c
c      in the GA (ft3/lbf*s)                                c
c      NEWCHILD(I,J) New Individual I consisting of parameters (J1, J2, J3,      c
c      J4) in the next generation of the GA (ft3/lbf*s)      c
225  c      RNKCHILD(I,J) Ranked population of the current generation in      c
c      the GA (ft3/lbf*s)                                    c
c      X(I,J)       Sensitivity matrix used in Gauss method      c
c      XT(I,J)      Transpose of sensitivity matrix            c
c      XTETA(I,J)   YYETA(I,J) matrix premultiplied by XT(I,J) c
c      YYETA(I,J)   Vector difference of Y(I) and ETA(I)      c
230  c      YYETAT(I,J) Transpose of YYETA(I,J)                c
c      PXTETA(I,J)  XTETA(I,J) matrix premultiplied by P(I,J) c
c      P(I,J)       Result of the inversion of the PINV(I,J) matrix c
c      PINV(I,J)    Matrix product of X(I,J) and XT(I,J)      c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
235  c
c      Declare variables. Real and Complex types are double precision. c
c
c      INTEGER  RUNNUM, MAXRUN, SEED1, SEED2, SIGNFLAG1
240  &      REAL*8  RANDOM, RANMAR, REALSEED, SMIN, XXSMIN,
&      UNCERTAIN, MEAN, TWOSIGMA, BMINUSA, RANSUM,
&      NEWNUM
&      REAL*8  BB(4), Y(8), MINB(4), XXMINB(4), INTERM(4),
&      FINAL(4), PROPS(5), TARGETB(4), PRESS(8)

245  c
c      It is essential that the user define the PROPS() vector, which      c
c      contains the properties of HEAT, PMEAN, T1, T2, and FREQ,          c
c      which are needed by SubProgram DYNAMITE. These are defined        c
c      here to fascillitate ease of change.                              c
250  c
c      PROPS(1) = DREAL(1200.50d0)
c      PROPS(2) = DREAL(14.699d0)
c      PROPS(3) = DREAL(1950.0d0)
255  c      PROPS(4) = DREAL(1550.0d0)
c      PROPS(5) = DREAL(300.0d0)

c
c      Initialize Random Number Generator, using values of PROPS()      c
c      for seeds. The seeds must be integers between 1 and 30000, so      c
c      the values must be scaled accordingly.                              c
260  c
c      REALSEED = DABS(DSIN(DREAL(PROPS(1)/PROPS(2))))*1.0D+7
20  REALSEED = REALSEED/10.0D0
c      IF (REALSEED.GT.30000.0D0) GOTO 20
c      SEED1 = INT(REALSEED)

265  c
c      REALSEED = DABS(DCOS(DREAL(PROPS(4)/PROPS(5))))*1.0D+7

```

```

22  REALSEED = REALSEED/10.0D0
    IF (REALSEED.GT.30000.0D0) GOTO 22
    SEED2 = INT(REALSEED)
270  c
    CALL RMARIN (SEED1,SEED2)
    DO 24 NUM = 1,(ABS(SEED1-SEED2)),1
        RANDOM = RANMAR()
275  24  CONTINUE
    c
    ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
    c   The following Commented block of logic is used for validation           c
    c   purposes. It allows the user to define target admittances,             c
    c   TARGETB(), then creates a target pressure vector, Y(), then           c
280  c   goes through the estimation process to see if the code can           c
    c   actually recover the user-defined target admittances.                 c
    c                                                                           c
    c   Assign target admittance vector, TARGET().                            c
    c                                                                           c
285  TARGETB(1) = DREAL(0.0371d0)
    TARGETB(2) = DREAL(0.0046d0)
    TARGETB(3) = DREAL(1.5750d0)
    TARGETB(4) = DREAL(-14.6644d0)
    c                                                                           c
290  c   Evaluate Targets. Here the target pressure vector, Y(), that         c
    c   results from the user-defined admittances is evaluated. The          c
    c   vector TARGET() is sent to SubProgram DYNAMITE,                       c
    c   and Y() returns.                                                       c
    c                                                                           c
295  write(*,*) 'Target Admittances:'
    write(*,*) targetb(1),targetb(2)
    write(*,*) targetb(3),targetb(4)
    call dynamite(targetb,y,props)
    write(*,*) 'TARGETS:'
300  do 37 i = 1,7,2
        write(*,*) y(i),y(i+1)
    37  continue
    c                                                                           c
    c   Perturb Targets with a random normal distribution. Here, the          c
305  c   target pressure vector, Y(), is "jostled" using a random             c
    c   generation of normally distributed values about the mean. The         c
    c   mean value is the individual pressure value, and the user            c
    c   defines the uncertainty. (The uncertainty should be defined          c
    c   as a fraction, e.g. +/-1% = 0.01).                                    c
    c                                                                           c
310  c
    do 38 i = 1,8,1
        uncertain = 0.00d0
        mean = y(i)
        twosigma = dabs(mean*uncertain)
315  sigmasq = (twosigma/2.0d0)*(twosigma/2.0d0)
        bminusa = dsqrt(12.0d0*sigmasq)
        ransum = 0.0d0
        do 39 j = 1,11,1
            newnum = ranmar()*bminusa + (mean-(bminusa/2.0d0))

```

```

320          ransum = ransum + newnum
    39      continue
           y(i) = ransum/11.0d0
    38      continue
c
c
325      Display the perturbed targets.
c
c
           write(*,*) 'PERTURBED TARGETS:'
           do 40 i = 1,7,2
               write(*,*) y(i),y(i+1)
330    40      continue
c
c      This end the block of logic used to theoretically
c      determine a target pressure vector, Y(). From this point
c      forward, the code is the same for both the theoretical
335      and experimental analysis.
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c
c      Initialize Variables. These variables must be initialized
340      every time a random restart occurs.
c
           RUNNUM = 0
           MAXRUN = 15
           SIGNFLAG1 = 1
345           SMIN = 10.0D0
           XXSMIN = 10.0D0
           MINB(1) = DREAL(0.50D0)
           MINB(2) = DREAL(2.0D0)
           MINB(3) = DREAL(0.50D0)
350           MINB(4) = DREAL(2.0D0)
           XXMINB(1) = DREAL(0.50D0)
           XXMINB(2) = DREAL(2.0D0)
           XXMINB(3) = DREAL(0.50D0)
           XXMINB(4) = DREAL(2.0D0)
355      c
c      Create Random Starting Position. A position is defined by a set
c      of parameters, BB(). Here, a random set of parameters is
c      generated. The user can define the constraints on these,
c      and also influence the sign (these controls allow the user to
360      make use of any a priori knowledge about the boundaries that
c      he/she might have.)
c
    5      RUNNUM = RUNNUM + 1
           RANDOM = RANMAR()
365           BB(1) = RANDOM*5.0D0
           RANDOM = RANMAR()
           BB(2) = RANDOM*5.0D0
           RANDOM = RANMAR()
370           IF (RANDOM.LE.0.75D0) THEN
               BB(2) = BB(2) * -1.0D0
           END IF
           RANDOM = RANMAR()
           BB(3) = RANDOM*20.0D0

```

```

375      RANDOM = RANMAR()
      BB(4) = RANDOM*20.0D0
      RANDOM = RANMAR()
      IF (RANDOM.LE.0.75D0) THEN
          BB(4) = BB(4) * -1.0D0
      END IF

380      c
      c      Call Subroutine GAUSS, which performs a Gauss linearization
      c      type of parameter estimation. This initial estimation is
      c      done to get in the general area of the error function global
      c      minimum. This is accomplished by trying to match the signs
385      c      (+/-) of Y() and the estimated pressure vector. The subroutine
      c      returns two sets of parameter estimates: MINB and XXMINB.
      c
      CALL GAUSS(Y,BB,MINB,XXMINB,SIGNFLAG1,PROPS,SMIN,XXSMIN)
      c

390      c      Check stopping Criteria. If the stopping Criteria are not met,
      c      then the estimation process begins from a new random location
      c      (where a set of parameters BB() defines a location).
      c
      IF ((SIGNFLAG1.EQ.1).AND.(RUNNUM.LE.MAXRUN)) GOTO 5

395      c
      c      Call Subroutine GENETIC, which performs a genetic algorithm-
      c      based parameter estimation. This step is used to focus more
      c      sharply on the error function minimum by overcoming any
      c      ill-conditioning that the Gauss method could not deal with.
400      c      The MINB and XXMINB vectors obtained from Subroutine GAUSS
      c      are sent to GENETIC as seeds for the initial random population.
      c      When complete, GENETIC returns an vector of improved
      c      parameter estimates called INTERM.
      c
405      CALL GENETICS(Y,MINB,XXMINB,PROPS,INTERM)
      c
      c      Call Subroutine GAUSS, which performs a second Gauss
      c      linearization; this time to pinpoint the parameters.
      c      The vector interm obtained from GENETIC is used as the
410      c      starting position in this last estimation process. The final
      c      parameter estimates are returned in the vector called FINAL.
      c
      FINAL(1) = DREAL(INTERM(1))
      FINAL(2) = DREAL(INTERM(2))
415      FINAL(3) = DREAL(INTERM(3))
      FINAL(4) = DREAL(INTERM(4))
      CALL GAUSS(Y,INTERM,FINAL,XXMINB,SIGNFLAG1,PROPS,SMIN,XXSMIN)
      c
      c      Call Subroutine POSTPRO, which post-processes the search
420      c      results. Output can be tailored to the user's liking.
      c      DYNAMITE is called on last time with FINAL to get a pressure
      c      field for the final parameter estimates.
      c
      CALL DYNAMITE(FINAL,PRESS,PROPS)
425      CALL POSTPRO(Y,TARGETB,FINAL,PRESS,PROPS)
      c

```

```

c                                                                    c
      FND                                                                    c
c                                                                    c
430  c                                                                    c
      ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c                                                                    c
c              SUBROUTINE POSTPRO                                                                    c
c                                                                    c
435  c                                                                    c
      ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c          This subroutine is used to post process the search results.    c
c          Output format can be tailored here by the user.                c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c                                                                    c
440  SUBROUTINE POSTPRO(Y,TARGETB,FINAL,PRESS,PROPS)
c                                                                    c
c          Declaration of variables.                                        c
c                                                                    c
      INTEGER          I, FN
445  REAL*8 TARMAG, RESMAG, ERROR1, ERROR2
      REAL*8 Y(8), PRESS(8), TARGETB(4), FINAL(4), PROPS(5)
      CHARACTER*14 NAME
c                                                                    c
c          Open a data file to write results to.                        c
450  c                                                                    c
      FN = 1
      FILENAME = 'ESTIMAT.DAT'
      OPEN(UNIT=FN,FILE=NAME)
c                                                                    c
455  c          Begin writing results.                                     c
c          Report the properties used by DYNAMITE.                       c
c                                                                    c
      WRITE(FN,*) 'DYNAMITE PROPERTIES:'
460  WRITE(FN,*) ' HEAT = ',PROPS(1)
      WRITE(FN,*) ' PMEAN = ',PROPS(2)
      WRITE(FN,*) ' T1 = ',PROPS(3)
      WRITE(FN,*) ' T2 = ',PROPS(4)
      WRITE(FN,*) ' FREQ = ',PROPS(5)
      WRITE(FN,*)
465  c                                                                    c
c          Report the target admittances. This is used if the user has
c          pre-defined the targets, i.e. theoretical analysis. Also
c          reported is the percent error between the estimated and
c          target admittances.
c                                                                    c
470  c                                                                    c
      * WRITE(FN,*) 'TARGET ADMITTANCES:'
      * WRITE(FN,*) TARGETB(1),TARGETB(2)
      * WRITE(FN,*) TARGETB(3),TARGETB(4)
      * WRITE(FN,*) 'RESULT ADMITTANCES:'
475  * WRITE(FN,*) FINAL(1),FINAL(2)
      * WRITE(FN,*) FINAL(3),FINAL(4)
      * WRITE(FN,*) '%ERROR IN ADMITTANCE VALUES:'
      * WRITE(FN,*) (DABS(TARGETB(1)-FINAL(1))/TARGETB(1))*100.0D0
      * WRITE(FN,*) (DABS(TARGETB(2)-FINAL(2))/TARGETB(2))*100.0D0
480  * WRITE(FN,*) (DABS(TARGETB(3)-FINAL(3))/TARGETB(3))*100.0D0

```

```

*      WRITE(FN,*) (DABS(TARGETB(4)-FINAL(4))/TARGETB(4))*100.0D0
      WRITE(FN,*)
c
c      Report the target pressure vector, Y(), and the resulting
485 c      estimated pressure vector, PRESS(). Also report the percent
c      error for the result pressure estimates.
c
      WRITE(FN,*) 'TARGET PRESSURES:'
      DO 25 I = 1,7,2
490      WRITE(FN,*) Y(I), Y(I+1)
      CONTINUE
      WRITE(FN,*) 'RESULT PRESSURES:'
      DO 27 I = 1,7,2
495      WRITE(FN,*) PRESS(I), PRESS(I+1)
      CONTINUE
      WRITE(FN,*) '%ERROR IN PRESSURE VALUES:'
      DO 29 I = 1,7,2
500      ERROR1 = (DABS(Y(I)-PRESS(I))/Y(I))*100.0D0
      ERROR2 = (DABS(Y(I+1)-PRESS(I+1))/Y(I+1))*100.0D0
      WRITE(FN,*) ERROR1, ERROR2
      CONTINUE
c
c      Report the target and result pressure magnitudes, and the
505 c      corresponding perCent error.
c
      WRITE(FN,*) 'TARGET AND RESULT PRESSURE MAGNITUDES:'
      DO 40 I = 1,7,2
      TARMAG = DSQRT(Y(I)*Y(I) + Y(I+1)*Y(I+1))
      RESMAG = DSQRT(PRESS(I)*PRESS(I) + PRESS(I+1)*PRESS(I+1))
510      WRITE(FN,*) TARMAG,RESMAG
      CONTINUE
      WRITE(FN,*) '%ERROR IN PRESSURE MAGNITUDES:'
      DO 41 I = 1,7,2
515      TARMAG = DSQRT(Y(I)*Y(I) + Y(I+1)*Y(I+1))
      RESMAG = DSQRT(PRESS(I)*PRESS(I) + PRESS(I+1)*PRESS(I+1))
      WRITE(FN,*) (DABS(TARMAG-RESMAG)/TARMAG)*100.0
      CONTINUE
c
c      Close the output data file, and terminate the subroutine.
520 c
      CLOSE (FN)
      END
c

525 ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c      SUBPROGRAM GAUSS
c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
530 c      This subprogram is used to perform a Gauss-Linearization type
c      parameter estimation. It requires the target vector Y(), an
c      initial set of parameters BB(), and the properties of the
c      system, PROPS(). It returns parameter estimates in MINB()
c      and XXMINB().
535 ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

```



```

c SUBROUTINE GAUSS(Y,BB,MINB,XXMINB,SIGNFLAG1,PROPS,SMIN,XXSMIN) c
c Declaration of variables. c
540 c INTEGER K, SENTRY, SIGNFLAG1, SIGNFLAG2, I, MINK, XXMINK c
REAL*8 SOLD, SNEW, SMIN, XXSMIN c
REAL*8 ETA(8), Y(8), BB(4), BBNEW(4), YYETA(8,1) c
REAL*8 X(8,4), XT(4,8), PINV(4,4), P(4,4), ETANEW(8) c
545 REAL*8 PROPS(5), MINB(4), XXMINB(4) c
c Initialize the variables K, SENTRY, and SNEW. c
c K = 0 c
550 SENTRY = 0 c
SNEW = 5.0D0 c
c Call subroutine DYNAMITE with the initial parameter set. c
c CALL DYNAMITE(BB,ETA,PROPS) c
555 c Begin the iterative estimation process, and Continue until c
one of the stopping criteria are met. c
c DO 106 WHILE (SENTRY.EQ.0) c
560 c Call subroutine LSTSQR, which calculates the least square c
error between the target pressures Y(), and the estimated c
pressures ETA(). c
565 CALL LSTSQR(Y,ETA,YYETA,SOLD) c
c Compare the signs (+/-) of ETA() to those of Y(). c
c SIGNFLAG1 = 0 c
570 DO 121 I = 1,8,1 c
IF ((Y(I)*ETA(I)).LT.0.0D0) SIGNFLAG1 = 1 c
121 CONTINUE c
c Call subroutine SENSTV, which calculates the sensitivity c
575 matrix for the current set of parameters. c
CALL SENSTV(X,XT,PINV,BB,ETA,PROPS) c
c Call subroutine INVERT, which inverts the sensitivity matrix. c
580 CALL INVERT(PINV,P) c
c Call subroutine PARAM, which uses the sensitivity matrix, its c
585 inverse, and the Current parameter estimates to Create a new c
parameter estimate BBNEW(). c
CALL PARAM(BB,P,XT,YYETA,BBNEW) c
c

```

```

590 c      Call subroutine DYNAMITE, which evaluates the pressure vector      c
c      for the new parameter estimates.                                     c
c      CALL DYNAMITE(BBNEW,ETANEW,PROPS)                                  c
c
595 c      Call subroutine LSTSQR, which evaluates the least square error      c
c      for the new parameter estimates.                                   c
c      CALL LSTSQR(Y,ETANEW,YYETA,SNEW)                                  c
c
600 c      Compare the signs of the new pressure vector ETANEW() to those      c
c      of the target vector Y().                                         c
c
c      SIGNFLAG2 = 0
c      DO 122 I = 1,8,1
605 c      IF ((Y(I)*ETANEW(I)).LT.0.0D0) SIGNFLAG2 = 1
122 c      CONTINUE
c
c      If the signs of ETANEW() are correct and if the least square
c      error is lower than that of the previous set with correct
610 c      signs, then assign MINB() the values of BBNEW().
c
c      IF ((SIGNFLAG2.EQ.0).AND.(SNEW.LT.SMIN)) THEN
c      SMIN = SNEW
c      MINK = K
615 c      DO 115 I = 1,4,1
115 c      MINB(I) = DREAL(BBNEW(I))
c      CONTINUE
c      END IF
c
620 c      If the signs of ETANEW() are incorrect and if the least square
c      error is lower than that of the previous set with incorrect
c      signs, then assign XXMINB() the values of BBNEW().
c
c      IF ((SIGNFLAG2.EQ.1).AND.(SNEW.LT.XXSMIN)) THEN
625 c      XXSMIN = SNEW
c      XXK = K
c      DO 117 I = 1,4,1
117 c      XXMINB(I) = DREAL(BBNEW(I))
c      CONTINUE
630 c      END IF
c
c      If the signs of ETA() are correct but the signs of ETANEW()
c      are incorrect, then terminate the iteration.
c
635 c      IF ((SIGNFLAG1.EQ.0).AND.(SIGNFLAG2.EQ.1)) THEN
c      SENTRY = 1
c      END IF
c
c      If the signs of ETANEW() are incorrect and if the least square
640 c      error increased from ETA() to ETANEW(), then terminate the
c      iteration.
c
c      IF ((SNEW.GT.SOLD).AND.(SIGNFLAG2.EQ.1)) THEN

```

```

                SENTRY = 1
645      END IF
c
c      Call subroutine STOPPER, which determines if any of the
c      stopping criteria have been met.
c
650      CALL STOPPER(BB,BBNEW,K,SENTRY)
c
c      Change the old parameters to the new, in preparation for the
c      next iteration.
c
655      DO 110 I = 1,4,1
        BB(I) = BBNEW(I)
110     CONTINUE
        DO 111 I = 1,8,1
          ETA(I) = ETANEW(I)
660     111     CONTINUE
            DO 112 I = 1,8,1
              IF ((Y(I)*ETA(I)).LT.0.0D0) SIGNFLAG1 = 1
112     CONTINUE
c
665     c      Increment the iteration counter, K.
c
c      K = K+1
c
c      End the iterative process.
670     c
106    CONTINUE
c
c      Return values of MINB, XXMINB, and SIGNFLAG1, and terminate
c      the subprogram.
675     c
        RETURN
        END
c
680     cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c              SUBROUTINE STOPPER
c
c      cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
685     c      This subroutine examines the stopping criteria, and terminates
c      the iterative process in GAUSS if either has been met.
c
c      cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
        SUBROUTINE STOPPER(BB,BBNEW,K,SENTRY)
690     c
c      Declaration of variables.
c
c              INTEGER      I, K, SENTRY
695     REAL*8        CHECKER, DEL, DEL1
        REAL*8        BB(4), BBNEW(4)
c
c      Assign values to Constants.

```

```

c
700  DEL = 1.0D-7
      DEL1 = 1.0D-38
c
c      Check the number of iterations stopping Criteria.
c
705  IF (K.GE.7) THEN
      SENTRY = 1
      END IF
c
c      Check the negligible change in parameters stopping Criteria.
c
710  DO 145 I = 1,4,1
      CHECKER = (DABS(BBNEW(I)-BB(I)))/(DABS(BB(I))+DEL1)
      IF (CHECKER.LT.DEL) THEN
          SENTRY = 1
          END IF
715  145  CONTINUE
c
c      Return value of SENTRY and terminate subroutine.
c
      RETURN
720  END
c

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
725  SUBROUTINE SENSTV
c
c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c      This subroutine calculates the sensitivity matrix for the Gauss
c      method estimation.
730  ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
      SUBROUTINE SENSTV(X,XT,PINV,BB,ETA,PROPS)
c
c      Declaration of variables.
735  c
      INTEGER          I,J,KOUNT
      REAL*8 ORIGINAL, DELB
      REAL*8 X(8,4), XT(4,8), PINV(4,4), BB(4), PROPS(5)
      REAL*8 ETA(8), DELETA(8)
740  c
c      Calculate the sensitivity matrix using the forward difference
c      technique of numerical differentiation.
c
c
745  DO 80 I = 1,4,1
      ORIGINAL = BB(I)
      DELB = 0.0001*BB(I)
      IF (I.GE.3) THEN
          DELB = 0.005*BB(I)
      END IF
750  BB(I) = BB(I) + DELB
      CALL DYNAMITE(BB,DELETA,PROPS)

```



```

      PXTETA(J,1) = 0.0D0
      DO 60 I=1,4,1
          PXTETA(J,1) = PXTETA(J,1)+(P(J,I)*XTETA(I,1))
810    60    CONTINUE
        50    CONTINUE
      c
      c      Add PXTETA() to the parameter estimates BB() to get the new
      c      parameter estimates BBNEW().
815    c
          DO 70 J = 1,4,1
              BBNEW(J) = BB(J) + PXTETA(J,1)
        70    CONTINUE
      c
820    c      Constrain the parameters according to physical meaning. The
      c      real components, BB(1) and BB(3), cannot be negative. Also,
      c      all of the parameters are kept away from zero to keep the
      c      estimation process from hitting a singularity.
      c
825    c      IF (BBNEW(1).LE.0) BBNEW(1)=BB(1)/5.0D0
      c      IF (BBNEW(2).EQ.0) BBNEW(2)=BB(2)/5.0D0
      c      IF (BBNEW(3).LE.0) BBNEW(3)=BB(3)/5.0D0
      c      IF (BBNEW(4).EQ.0) BBNEW(4)=BB(4)/5.0D0
      c
830    c      Constraint he parameters to a reasonable range to keep the
      c      iterative process from going unstable.
      c
          DO 71 I = 1,4,1
              IF (DABS(BBNEW(I)).GT.100.0D0) THEN
835    c          BBNEW(I) = BB(I)/5.0D0
              END IF
        71    CONTINUE
      c
840    c      Return value of BBNEW(), and terminate the subroutine.
      c
          RETURN
          END
      c
845    ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      c
      c          SUBROUTINE LSTSQR
      c
      ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
850    c      This subroutine evaluates the least square error between the
      c      taeget pressure vector, Y(), and the estimated pressure vector.
      ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      c
          SUBROUTINE LSTSQR(Y,ETA,YYETA,S)
855    c
      c      Declaration of variables.
      c
          INTEGER      I
          REAL*8 Y(8), ETA(8), YYETA(8,1), YYETAT(1,8), S
860    c

```

```

c      Calculate the difference vector, YYETA(), and the least square      c
c      error, S.                                                         c
c                                                                                   c
865      S = 0.0D0
      DO 20 I=1,8,1
          YYETA(I,1) = Y(I) - ETA(I)
          YYETAT(1,I) = YYETA(I,1)
          S = S + YYETAT(1,I)*YYETA(I,1)
20      CONTINUE
870      RETURN values of YYETA() and S, and terminate the subroutine.      c
c                                                                                   c
c      RETURN                                                             c
      END
875      c                                                                                   c

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c                                                                                   c
c      SUBROUTINE INVERT                                                  c
880      c                                                                                   c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c      This subroutine uses row elimination techniques to invert the      c
c      PINV matrix and return the P matrix. The PINV matrix is row-      c
c      reduced to the identity matrix. By performing the same oper-      c
885      c      ations on a separate identity matrix as those performed on      c
c      PINV, the matrix inverse, P, is found. Locally, PINV is          c
c      referred to as A, and P as INV.                                     c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c                                                                                   c
890      SUBROUTINE INVERT(A,INV)
c                                                                                   c
c      Declaration of variables.                                          c
c                                                                                   c
895      INTEGER ROW, NROW, MAXROW, STRW, COL, NCOL, MAXCOL, STCL
      REAL*8  A(4,4), ANEW(4,4), INV(4,4), INVNEW(4,4)
      REAL*8  CHECK(4,4), ORIG(4,4), PIVOT
c                                                                                   c
c      Assign the dimensions of the matrices.                             c
c                                                                                   c
900      MAXROW = 4
      MAXCOL = 4
c                                                                                   c
c      Initialize the INV() matrix as the identity matrix, and assign      c
c      values of A() to ORIG(), which will be used as a check at the      c
905      c      end of the inversion process.                               c
c                                                                                   c
      DO 10 ROW = 1, MAXROW, 1
      DO 20 COL = 1, MAXCOL, 1
          ORIG(ROW,COL) = A(ROW,COL)
          INV(ROW,COL) = 0.0
910      20  CONTINUE
          INV(ROW,ROW) = 1.0
      10  CONTINUE
c                                                                                   c

```

```

915  c      Scale the values of A().
      c
      DO 13 ROW = 1,MAXROW,1
        DO 15 COL = 1,MAXCOL,1
          A(ROW,COL)=A(ROW,COL)*1.0D5
920  15  CONTINUE
      13  CONTINUE
      c
      c      Row-reduce the lower trinagle of the A() matrix.
      c
925  DIVISOR = A(1,1)
      IF ((DIVISOR.NE.1.0).AND.(DIVISOR.NE.0.0))THEN
        DO 25 COL = 1,MAXCOL,1
          A(1,COL) = A(1,COL)/DIVISOR
          INV(1,COL) = INV(1,COL)/DIVISOR
930  25  CONTINUE
        END IF
      c
      DO 30 COL = 1, MAXCOL, 1
        ANEW(1,COL) = A(1,COL)
        INVNEW(1,COL) = INV(1,COL)
935  30  CONTINUE
      c
      STRW = 2
      c
940  DO 40 STCL = 1, (MAXCOL-1), 1
        DO 50 ROW = STRW, MAXROW, 1
          PIVOT = A(STRW-1,STCL)*A(STRW,STCL+1)
          &      -A(STRW,STCL)*A(STRW-1,STCL+1)
          IF (PIVOT.NE.0) THEN
945  DO 60 COL = STCL, MAXCOL, 1
            ANEW(ROW,COL) = A(STRW-1,STCL)*A(ROW,COL)
            &      -A(ROW,STCL)*A(STRW-1,COL)
            ANEW(ROW,COL) = ANEW(ROW,COL)/PIVOT
          60  CONTINUE
          DO 70 COL = 1, MAXCOL, 1
            INVNEW(ROW,COL) = A(STRW-1,STCL)*INV(ROW,COL)
            &      -A(ROW,STCL)*INV(STRW-1,COL)
            INVNEW(ROW,COL) = INVNEW(ROW,COL)/PIVOT
950  70  CONTINUE
          END IF
          50  CONTINUE
          DO 80 NROW = 1,MAXROW,1
            DO 90 NCOL = 1,MAXCOL,1
              A(NROW,NCOL) = ANEW(NROW,NCOL)
              INV(NROW,NCOL) = INVNEW(NROW,NCOL)
960  90  CONTINUE
          80  CONTINUE
          STRW = STRW+1
          40  CONTINUE
965  c
      c      Row-reduce the upper trinagle of the A() matrix.
      c
      STRW = MAXROW-1
      DO 100 STCL = MAXCOL,1,-1

```



```

970      DO 110 ROW = STRW,1,-1
          PIVOT = A(STRW+1,COL)*A(ROW,STCL-1)
          &      -A(ROW,STCL)*A(STRW+1,STCL-1)
          DO 120 COL = STCL,1,-1
          ANEW(ROW,COL) = A(STRW+1,STCL)*A(ROW,COL)
975      &      -A(ROW,STCL)*A(STRW+1,COL)
          ANEW(ROW,COL) = ANEW(ROW,COL)
120      CONTINUE
          DO 125 COL = MAXCOL,1,-1
          INVNEW(ROW,COL) = A(STRW+1,STCL)*INV(ROW,COL)
980      &      -A(ROW,STCL)*INV(STRW+1,COL)
          INVNEW(ROW,COL) = INVNEW(ROW,COL)
125      CONTINUE
110      CONTINUE
          DO 130 NROW=1,MAXROW,1
985      DO 140 NCOL = 1,MAXCOL,1
          A(NROW,NCOL) = ANEW(NROW,NCOL)
          INV(NROW,NCOL) = INVNEW(NROW,NCOL)
140      CONTINUE
130      CONTINUE
990      STRW = STRW-1
100      CONTINUE
c
c      Since A() is always symmetric, its inverse will be symmetric.
c      Here the INV() matrix is made symmetric in case any round-off
995      error occurred during the row-reduction.
c
          DO 143 COL = 3,1,-1
          DO 144 ROW = 1,COL,1
          INV(ROW,COL+1)=INV(COL+1,ROW)
1000      144      CONTINUE
          143      CONTINUE
c
c      Scale the inverse matrix.
c
1005      DO 14 ROW = 1,MAXROW,1
          DO 16 COL = 1,MAXCOL,1
          INV(ROW,COL)=INV(ROW,COL)*1.0D5
16      CONTINUE
14      CONTINUE
1010      c
c      Check to see if the original matrix multiplied by the inverse
c      returns the identity matrix.
c
1015      DO 170 ROW = 1, MAXROW,1
          DO 180 COL = 1,MAXCOL,1
          CHECK(ROW,COL)= 0.0
          DO 181 K = 1,4,1
          CHECK(ROW,COL)=CHECK(ROW,COL)+(ORIG(ROW,K)*INV(K,COL))
181      CONTINUE
1020      180      CONTINUE
          170      CONTINUE
c
c      Return the value of P(), and terminate the subroutine.
c

```



```

1080 13 CONTINUE
c
c Initialize variables and constants. Here is where the user can
c define the population size, the number of individuals kept
c from one generation to the next, the mutation rate, the
1085 c maximum number of generations, and the acceptable least square
c error.
c
POPNUM = 40
KEEPNUM = INT(POPNUM*0.08)
1090 MUTATION = 0.080D0
MAXGEN = 20
STOPLSE = 1.0E-25
MINLSE = 10.0D0
NUMETA = 8
1095 GEN = 1
c
c Create the bounds for a random population using ELITE().
c
DO 15 CHROME = 1,4,1
1100 HIGH(CHROME) = 10.0D0*DABS(ELITE(1,CHROME))
LOW(CHROME) = 10.0D0*DABS(ELITE(2,CHROME))
15 CONTINUE
c
c Create the initial, random population. Parameter values are
1105 c determined using a uniform random number generator and the
c pre-determined bounds. The imaginary components, the second
c and fourth parameters, can be influenced towards positive
c or negative values here.
c The first two individuals in the population are MINB and XXMINB.
1110 c
DO 19 RANK = 1,2,1
DO 20 CHROME = 1,4,1
CHILD(RANK,CHROME) = ELITE(RANK,CHROME)
20 CONTINUE
1115 19 CONTINUE
c
c The next 16 individuals are created using the HIGH bounds.
c
DO 21 RANK = 3,18,1
1120 DO 22 CHROME = 1,4,1
RANDOM = RANMAR()
CHILD(RANK,CHROME) = RANDOM*HIGH(CHROME)
IF ((CHROME.EQ.2).OR.(CHROME.EQ.4)) THEN
RANDOM = RANMAR()
1125 IF (RANDOM.LE.0.75D0) THEN
CHILD(RANK,CHROME)=CHILD(RANK,CHROME)*-1.0D0
END IF
END IF
22 CONTINUE
1130 21 CONTINUE
c
c The next 12 individuals are created using the LOW bounds.
c

```

```

1135      DO 234 RANK = 19,30,1
          DO 235 CHROME = 1,4,1
              RANDOM = RANMAR()
              CHILD(RANK,CHROME) = RANDOM*LOW(CHROME)
              IF ((CHROME.EQ.2).OR.(CHROME.EQ.4)) THEN
1140                  RANDOM = RANMAR()
                      IF (RANDOM.LE.0.75D0) THEN
                          CHILD(RANK,CHROME)=CHILD(RANK,CHROME)*-1.0D0
                      END IF
                  END IF
          CONTINUE
1145      235 CONTINUE
          234 CONTINUE
c
c      The last 10 individuals are created using random multiplication
c      or division by a random number. This is done to introduce more
c      diversity into the population.
1150      c
          DO 237 RANK = 31,POPNUM,1
              DO 238 CHROME = 1,4,1
                  RANDOM = RANMAR()
1155                  IF (RANDOM.LE.0.50D0) THEN
                      ELITIST = 1
                  ELSE
                      ELITIST = 2
                  END IF
                  RANDOM = RANMAR()
1160                  IF (RANDOM.LE.0.50D0) THEN
                      CHILD(RANK,CHROME) = ELITE(ELITIST,CHROME)*RANMAR()
                  ELSE IF (RANDOM.LE.1.00D0) THEN
                      CHILD(RANK,CHROME) = ELITE(ELITIST,CHROME)/RANMAR()
                  END IF
1165                  IF ((CHROME.EQ.2).OR.(CHROME.EQ.4)) THEN
                      RANDOM = RANMAR()
                      IF (RANDOM.LT.0.50D0) THEN
                          CHILD(RANK,CHROME) = -CHILD(RANK,CHROME)
                      END IF
                  END IF
1170                  CONTINUE
          238 CONTINUE
          237 CONTINUE
c
c      Next, the least square error is evaluated for each individual
1175      c      in the population. If identical individuals exist, computing
c      time is saved by only calling DYNAMITE once per unique set.
c      Also, the average least square error for the generation is
c      calculated.
c
1180      TOTALLSE = 0.0D0
          DO 25 RANK = 1,POPNUM,1
              ADMITTIN = DCMLPX(CHILD(RANK,1),CHILD(RANK,2))
              ADMITOUT = DCMLPX(CHILD(RANK,3),CHILD(RANK,4))
              SAMEFLAG = 0
1185              IF (RANK.GT.1) THEN
                  DO 26 NUM = 1,RANK-1,1
                      TSTIN=DCMLPX(CHILD(NUM,1),CHILD(NUM,2))

```

```

TSTOUT=DCMPLX(CHILD(NUM,3),CHILD(NUM,4))
IF ((ADMITIN.EQ.TSTIN).AND.(ADMITOUT.EQ.TSTOUT)) THEN
1190     SAMEFLAG = 1
        LSE(RANK) = LSE(NUM)
        END IF
26     CONTINUE
    END IF
1195     IF (SAMEFLAG.NE.1) THEN
        BB(1) = CHILD(RANK,1)
        BB(2) = CHILD(RANK,2)
        BB(3) = CHILD(RANK,3)
        BB(4) = CHILD(RANK,4)
1200     CALL DYNAMITE(BB,ETA,PROPS)
        LSE(RANK) = 0.0D0
        DO 27 NUM = 1,NUMETA,1
            LSE(RANK)=LSE(RANK)+(TARGET(NUM)-ETA(NUM))*
1205     & (TARGET(NUM)-ETA(NUM))
27     CONTINUE
        END IF
25     TOTALLSE = TOTALLSE + LSE(RANK)
    CONTINUE
    AVGLSE(GEN) = TOTALLSE/DREAL(POPNUM)
1210     c
    c     Call subroutine RANKER, which ranks the individuals according
    c     to their least square error value, with the lowest being best.
    c
29     CALL RANKER(CHILD,LSE,POPNUM)
1215     c
    c     If the least square error of the best individual is lower than
    c     the previous best, then assign the values of that individual
    c     to INTERM().
    c
1220     LOWLSE(GEN) = LSE(1)
    IF (LSE(1).LT.MINLSE) THEN
        DO 299 CHROME = 1,4,1
            INTERM(CHROME) = CHILD(1,CHROME)
299     CONTINUE
1225     MINLSE = LSE(1)
    END IF
    c
    c     Assign a probability of being Chosen as a parent to the best
    c     twenty individuals. The probabilities are proportionate to
1230     the least square error values.
    c
    DO 34 RANK = 1,POPNUM,1
        PROB(RANK) = 0.0D0
        PROBDIV(RANK) = 0.0D0
1235     34     CONTINUE
        POPNUM = 20
    c
    SUM1 = 0.0D0
    DO 30 RANK = 1,POPNUM,1
        SUM1 = SUM1+LSE(RANK)
1240     30     CONTINUE
    c

```

```

SUM2 = 0.0D0
DO 35 RANK = 1,POPNUM,1
1245     SUM2 = SUM2+(SUM1/LSE(RANK))
        35 CONTINUE
        c
        DO 40 RANK = 1,POPNUM,1
        PROB(RANK) = (SUM1/LSE(RANK))/SUM2
1250     40 CONTINUE
        c
        POPNUM = 40
        DO 45 RANK = 1,POPNUM,1
        PROBDIV(RANK)=0.0D0
1255     DO 47 NUM = 1,RANK,1
        PROBDIV(RANK)=PROBDIV(RANK)+PROB(NUM)
        47 CONTINUE
        45 CONTINUE
        c
1260     c Generate the new population by choosing parents for a child,
        c and then performing the crossover operation, and then checking
        c for mutation.
        c
        c Loop through for each child in the next generation.
1265     c
        DO 50 NUM = KEEPNUM+1,POPNUM,1
        c
        c Choose the first parent for the child.
        c
1270     RANDOM = RANMAR()
        IF (RANDOM.LT.PROBDIV(1)) THEN
            PARENT1 = 1
        ELSE IF (RANDOM.LT.PROBDIV(2)) THEN
            PARENT1 = 2
1275     ELSE IF (RANDOM.LT.PROBDIV(3)) THEN
            PARENT1 = 3
        ELSE IF (RANDOM.LT.PROBDIV(4)) THEN
            PARENT1 = 4
1280     ELSE IF (RANDOM.LT.PROBDIV(5)) THEN
            PARENT1 = 5
        ELSE IF (RANDOM.LT.PROBDIV(6)) THEN
            PARENT1 = 6
        ELSE IF (RANDOM.LT.PROBDIV(7)) THEN
            PARENT1 = 7
1285     ELSE IF (RANDOM.LT.PROBDIV(8)) THEN
            PARENT1 = 8
        ELSE IF (RANDOM.LT.PROBDIV(9)) THEN
            PARENT1 = 9
1290     ELSE IF (RANDOM.LT.PROBDIV(10)) THEN
            PARENT1 = 10
        ELSE IF (RANDOM.LT.PROBDIV(11)) THEN
            PARENT1 = 11
        ELSE IF (RANDOM.LT.PROBDIV(12)) THEN
            PARENT1 = 12
1295     ELSE IF (RANDOM.LT.PROBDIV(13)) THEN
            PARENT1 = 13
        ELSE IF (RANDOM.LT.PROBDIV(14)) THEN

```

```

        PARENT1 = 14
1300     ELSE IF (RANDOM.LT.PROBDIV(15)) THEN
        PARENT1 = 15
        ELSE IF (RANDOM.LT.PROBDIV(16)) THEN
        PARENT1 = 16
        ELSE IF (RANDOM.LT.PROBDIV(17)) THEN
1305     PARENT1 = 17
        ELSE IF (RANDOM.LT.PROBDIV(18)) THEN
        PARENT1 = 18
        ELSE IF (RANDOM.LT.PROBDIV(19)) THEN
        PARENT1 = 19
1310     ELSE IF (RANDOM.LT.PROBDIV(20)) THEN
        PARENT1 = 20
        END IF

```

c  
c  
c

Choose the second parent for the child.

c  
c  
c

```

1315     RANDOM = RANMAR()
        IF (RANDOM.LT.PROBDIV(1)) THEN
        PARENT2 = 1
        ELSE IF (RANDOM.LT.PROBDIV(2)) THEN
1320     PARENT2 = 2
        ELSE IF (RANDOM.LT.PROBDIV(3)) THEN
        PARENT2 = 3
        ELSE IF (RANDOM.LT.PROBDIV(4)) THEN
1325     PARENT2 = 4
        ELSE IF (RANDOM.LT.PROBDIV(5)) THEN
        PARENT2 = 5
        ELSE IF (RANDOM.LT.PROBDIV(6)) THEN
        PARENT2 = 6
        ELSE IF (RANDOM.LT.PROBDIV(7)) THEN
1330     PARENT2 = 7
        ELSE IF (RANDOM.LT.PROBDIV(8)) THEN
        PARENT2 = 8
        ELSE IF (RANDOM.LT.PROBDIV(9)) THEN
        PARENT2 = 9
1335     ELSE IF (RANDOM.LT.PROBDIV(10)) THEN
        PARENT2 = 10
        ELSE IF (RANDOM.LT.PROBDIV(11)) THEN
        PARENT2 = 11
        ELSE IF (RANDOM.LT.PROBDIV(12)) THEN
1340     PARENT2 = 12
        ELSE IF (RANDOM.LT.PROBDIV(13)) THEN
        PARENT2 = 13
        ELSE IF (RANDOM.LT.PROBDIV(14)) THEN
        PARENT2 = 14
1345     ELSE IF (RANDOM.LT.PROBDIV(15)) THEN
        PARENT2 = 15
        ELSE IF (RANDOM.LT.PROBDIV(16)) THEN
        PARENT2 = 16
        ELSE IF (RANDOM.LT.PROBDIV(17)) THEN
        PARENT2 = 17
1350     ELSE IF (RANDOM.LT.PROBDIV(18)) THEN
        PARENT2 = 18
        ELSE IF (RANDOM.LT.PROBDIV(19)) THEN

```

```

        PARENT2 = 19
ELSE IF (RANDOM.LT.PROBDIV(20)) THEN
1355     PARENT2 = 20
        END IF
c
c     Pass the genetic information from the parents to the child.
c     The new parameters in a child can be exact duplicates of the
1360 c     parents, or an averga of the two. The user can define the
c     likelihood of each here.
c
        DO 55 CHROME = 1,4,1
1365     RANDOM = RANMAR()
        IF (RANDOM.LT.0.30D0) THEN
            NEWCHILD(NUM,CHROME)=CHILD(PARENT1,CHROME)
        ELSE IF (RANDOM.LT.0.60D0) THEN
            NEWCHILD(NUM,CHROME)=CHILD(PARENT2,CHROME)
1370     ELSE IF (RANDOM.LT.1.0D0) THEN
            NEWCHILD(NUM,CHROME)=(CHILD(PARENT1,CHROME)+
&     CHILD(PARENT2,CHROME))/2.0D0
        END IF
c
c     Check the new parameter in the child for mutation. If mutation
1375 c     occurs, Then create a completely new chromosome/parameter.
c     The user can define new bounds for the parameters here, or use
c     the original bounds of HIGH and LOW.
c
        RANDOM = RANMAR()
1380     IF (RANDOM.LE.MUTATION) THEN
            RANDOM = RANMAR()
            IF (CHROME.EQ.1) THEN
                NEWCHILD(NUM,CHROME)=RANDOM*5.0D0
1385     ELSE IF (CHROME.EQ.2) THEN
                NEWCHILD(NUM,CHROME)=RANDOM*5.0D0
                RANDOM = RANMAR()
                IF (RANDOM.LE.0.75D0) THEN
                    NEWCHILD(NUM,CHROME)=NEWCHILD(NUM,CHROME)*-1.0D0
1390     ELSE IF (CHROME.EQ.3) THEN
                    NEWCHILD(NUM,CHROME)=RANDOM*20.0D0
                ELSE IF (CHROME.EQ.4) THEN
                    NEWCHILD(NUM,CHROME)=RANDOM*20.0D0
1395     RANDOM = RANMAR()
                    IF (RANDOM.LE.0.75D0) THEN
                        NEWCHILD(NUM,CHROME)=NEWCHILD(NUM,CHROME)*-1.0D0
                    END IF
                END IF
            END IF
1400     55 CONTINUE
        50 CONTINUE
c
c     Place the children into the population.
c
1405     DO 60 RANK = KEEPNUM+1,POPNUM,1
        DO 65 CHROME = 1,4,1

```



```

                                CHILD(RANK,CHROME) = NEWCHILD(RANK,CHROME)
65      CONTINUE
60      CONTINUE
1410   c
      c      Increment the generation counter.
      c
      GEN = GEN + 1
      c
1415   c      Evaluate the least square error, same as above.
      c
      c
      TOTALLSE = 0.0D0
      DO 71 RANK = 1,KEEPNUM,1
1420   71      TOTALLSE = TOTALLSE + LSE(RANK)
      CONTINUE
      DO 70 RANK = KEEPNUM+1,POPNUM,1
      ADMITTIN = DCMLPX(CHILD(RANK,1),CHILD(RANK,2))
      ADMITOUT = DCMLPX(CHILD(RANK,3),CHILD(RANK,4))
      SAMEFLAG = 0
1425   IF (RANK.GT.1) THEN
      DO 73 NUM = 1,RANK-1,1
      TSTIN= DCMLPX(CHILD(NUM,1),CHILD(NUM,2))
      TSTOUT= DCMLPX(CHILD(NUM,3),CHILD(NUM,4))
      IF ((ADMITIN.EQ.TSTIN).AND.(ADMITOUT.EQ.TSTOUT)) THEN
1430   SAMEFLAG = 1
      LSE(RANK) = LSE(NUM)
      END IF
73      CONTINUE
      END IF
1435   IF (SAMEFLAG.NE.1) THEN
      BB(1) = CHILD(RANK,1)
      BB(2) = CHILD(RANK,2)
      BB(3) = CHILD(RANK,3)
      BB(4) = CHILD(RANK,4)
1440   CALL DYNAMITE(BB,ETA,PROPS)
      LSE(RANK) = 0.0D0
      DO 75 NUM = 1,NUMETA,1
      LSE(RANK)=LSE(RANK)+(TARGET(NUM)-ETA(NUM))*
1445   &      (TARGET(NUM)-ETA(NUM))
      75      CONTINUE
      END IF
      TOTALLSE = TOTALLSE + LSE(RANK)
70      CONTINUE
      AVGLSE(GEN) = TOTALLSE/DREAL(POPNUM)
1450   c
      c      Examine the stopping criteria. If the generation number is less
      c      then the maximum allowed, and if the lowest least square error
      c      is higher than the highest allowed, then go back to the step
      c      of ranking the population.
1455   c
      c      IF ((GEN.LT.MAXGEN).AND.(MINLSE.GT.STOPLSE)) GOTO 29
      c
      c      Return the value of INTERM(), and terminate the subprogram.
      c
1460   RETURN
      END

```

```

c
c
c
c
1465 c SUBROUTINE RANKER c
c
c
c
1470 c This subroutine ranks the individuals in the population accord- c
c ing to their least square error values, with a lower value c
c being better. c
c c
c SUBROUTINE RANKER(CHILD,LSE,POPNUM) c
1475 c Declaration of variables. c
c
c INTEGER SET, RANK, NEWRANK, CHROME, POPNUM
REAL*8 CHILD(50,4),RNKCHILD(50,4),TMPCHILD(4),
1480 & NEWCHILD(4), LSE(50), RNKLSE(50), TEMP, NEW c
c Rank the individuals and store them in RNKCHILD(). c
c
c DO 20 SET = 1,POPNUM,1
1485 NEW = LSE(SET)
NEW RANK = SET
RNKLSE(NEWRANK) = LSE(SET)
DO 25 CHROME = 1,4,1
NEWCHILD(CHROME) = CHILD(SET,CHROME)
1490 RNKCHILD(NEWRANK,CHROME) = CHILD(SET,CHROME)
25 CONTINUE
DO 30 RANK = (SET-1),1,-1
IF (NEW.LT.RNKLSE(RANK)) THEN
1495 TEMP = RNKLSE(RANK)
RNKLSE(RANK) = NEW
RNKLSE(NEWRANK) = TEMP
DO 34 CHROME = 1,4,1
TMPCHILD(CHROME) = RNKCHILD(RANK,CHROME)
34 CONTINUE
1500 DO 35 CHROME = 1,4,1
RNKCHILD(RANK,CHROME) = NEWCHILD(CHROME)
35 CONTINUE
DO 36 CHROME = 1,4,1
RNKCHILD(NEWRANK,CHROME) = TMPCHILD(CHROME)
1505 36 CONTINUE
NEWRANK = RANK
END IF
30 CONTINUE
20 CONTINUE
1510 c Assign the values of RNKCHILD() to CHILD(). c
c
c
c
1515 c DO 40 RANK = 1,POPNUM,1
LSE(RANK) = RNKLSE(RANK)
DO 45 CHROME = 1,4,1
CHILD(RANK,CHROME) = RNKCHILD(RANK,CHROME)

```

```
45      CONTINUE
40      CONTINUE

1520    c                                     c
c       Return the value of CHILD(), and terminate the subroutine.    c
c                                                     c
       RETURN
       END

1525    c                                     c

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c                                             c
c                               SUBROUTINE RMARIN                      c
1530    c                                     c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c       This subroutine initializes a uniform random number generator.  c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c                                             c
1535    SUBROUTINE RMARIN(IJ, KL)
c
c THIS IS THE INITIALIZATION ROUTINE FOR THE RANDOM NUMBER
c GENERATOR RANMAR().
c
1540    C NOTE: THE SEED VARIABLES CAN HAVE VALUES BETWEEN:
c      0 <= IJ <= 31328
c      0 <= KL <= 30081
c
c THIS RANDOM NUMBER GENERATOR CAN CREATE 900 MILLION DIFFERENT
1545    C SUBSEQUENCES -- WITH EACH SUBSEQUENCE HAVING A LENGTH OF
c APPROXIMATELY 10^30.
c
c USE IJ = 1802 & KL = 9373 TO TEST THE RANDOM NUMBER GENERATOR.
c THE SUBROUTINE RANMAR SHOULD BE USED TO GENERATE 20000 RANDOM
1550    C NUMBERS. THEN DISPLAY THE NEXT SIX RANDOM NUMBERS GENERATED
c MULTIPLIED BY 4096*4096. IF THE RANDOM NUMBER GENERATOR IS
c WORKING PROPERLY, THE RANDOM NUMBERS SHOULD BE:
c
c      6533892.0 14220222.0 7275067.0
1555    c      6172232.0 8354498.0 10633180.0
c
c      Declaration of variables.
c
c      REAL*8 U(97), C, CD, CM, S, T
1560    INTEGER II, I, J, IJ, JJ, K, KL, L, M, I97, J97
c      LOGICAL TEST
c      COMMON /RASET1/ U, C, CD, CM, I97, J97, TEST
c      TEST = .FALSE.
c
1565    IF( IJ .LT. 0 .OR. IJ .GT. 31328 .OR.
1     KL .LT. 0 .OR. KL .GT. 30081 ) THEN
c      WRITE (*, *) ' THE FIRST RANDOM NUMBER SEED MUST HAVE A '
c      WRITE (*, *) ' VALUE BETWEEN 0 AND 31328.'
c      WRITE (*, *)
1570    WRITE (*, *) ' THE SECOND SEED MUST HAVE A VALUE BETWEEN 0 '
c      WRITE (*, *) ' AND 30081.'
```

```

    WRITE (*, *) 'STOPPING...'
    STOP
  ENDIF
1575  C
    I = MOD(IJ/177, 177) + 2
    J = MOD(IJ , 177) + 2
    K = MOD(KL/169, 178) + 1
    L = MOD(KL, 169)
1580  C
    DO 2 II = 1, 97
      S = 0.0
      T = 0.5
      DO 3 JJ = 1, 24
1585      M = MOD(MOD(I*J, 179)*K, 179)
          I = J
          J = K
          K = M
          L = MOD(53*L+1, 169)
1590      IF (MOD(L*M, 64) .GE. 32) THEN
          S = S + T
          ENDIF
          T = 0.5 * T
        3  CONTINUE
1595      U(II) = S
        2  CONTINUE
    C
      C = 362436.0 / 16777216.0
      CD = 7654321.0 / 16777216.0
1600      CM = 16777213.0 / 16777216.0
    C
      I97 = 97
      J97 = 33
    C
1605      TEST = .TRUE.
    C
      Return and terminate the subroutine. c
    C c
    RETURN
1610  END
    c c

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c c
1615  c          FUNCTION RANMAR c
    c c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c          This function returns a random number when called. c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
1620  c c
      REAL*8 FUNCTION RANMAR()
    C
    C THIS IS THE RANDOM NUMBER GENERATOR PROPOSED BY GEORGE MARSAGLIA
    C IN FLORIDA STATE UNIVERSITY REPORT: FSU-SCRI-87-50
1625  C
    c c

```

```

c      Declaration of variables.                                c
c                                                                 c
1630  REAL*8 U(97), UNI, C, CD, CM
      INTEGER I97, J97
      LOGICAL TEST
      COMMON /RASET1/ U, C, CD, CM, I97, J97, TEST
C
1635  IF(.NOT.TEST) THEN
      WRITE (*, *)
      WRITE (*, *) 'RANMAR ERROR: YOU MUST CALL THE'
      WRITE (*, *) 'INITIALIZATION ROUTINE RMARIN BEFORE'
      WRITE (*, *) 'CALLING RANMAR.'
1640  WRITE (*, *) 'STOPPING...'
      STOP
      ENDIF
C
1645  UNI = U(I97) - U(J97)
      IF(UNI .LT. 0.0) UNI = UNI + 1.0
      U(I97) = UNI
      I97 = I97 - 1
      IF(I97 .EQ. 0) I97 = 97
      J97 = J97 - 1
      IF(J97 .EQ. 0) J97 = 97
1650  C = C - CD
      IF(C .LT. 0.0) C = C + CM
      UNI = UNI - C
      IF(UNI .LT. 0.0) UNI = UNI + 1.0
C
1655  RANMAR = UNI
c                                                                 c
c      Return the value RANMAR(), and terminate the function.    c
c                                                                 c
1660  RETURN
      END
c                                                                 c

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
1665  c                                                                 c
      c      Subprogram DYNAMITE                                c
      c                                                                 c
      c      This program computes the acoustic field in a gas turbine
      c      combustor of user defined: type (can or annular), dimensions
      c      steady-flow gas temperature distribution, acoustic boundary
1670  c      conditions, and source distribution at a specified frequency.
      c                                                                 c
      c      Written by Andrew D. Wright, Graduate Student of Mechanical
      c      Engineering, Virginia Polytechnic Institute and State
      c      University, Blacksburg, Virginia. Written for The United
1675  c      States Department of Energy, Morgantown Energy Technology
      c      Center, Morgantown, West Virginia, April-August, 1995.
      c                                                                 c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
1680  c                                                                 c
      c      Main SubProgram                                    c
      c                                                                 c

```



c local node of element N. c  
 c NODE Index number of the I,<sup>1</sup>,Kth node; value computed c  
 c in the "40" DO-Loop. c  
 1740 c NPE Number of nodes per element (= 8). c  
 c NQ1 Node number of lowest numbered node with heat c  
 c input. c  
 c NQ2 Node number of highest numbered node with heat c  
 c input. c  
 1745 c NR Number of radial layers of elements. c  
 c NR1 NR + 1. c  
 c NTHETA Number of circumferential sectors of elements c  
 c in entire combustor. c  
 c NW Auxiliary variable used in computing NHBW. c  
 1750 c NZ Number of axial blocks of elements. c  
 c NZ1 NZ + 1. c  
 c  
 c Real Quantities: c  
 c  
 1755 c A Nondimensional factor representing the relative c  
 c thermoacoustic efficiency at the frequency being c  
 c considered (-). User specified between 0 and 1.0. c  
 c ANGLE Auxiliary variable representing the angular c  
 c position in the burner (rad), used to compute c  
 1760 c the angular distribution of heat input in the c  
 c heat end. c  
 c COEFF Nondimensional coefficient used in Gauss quadra- c  
 c ture integration to locate Gauss points (-). c  
 c DZ Physical distance between fuel injector interface c  
 1765 c with combustion liner and downstream limit of c  
 c first block of elements (ft). User specified. c  
 c ETA Nondimensional dimension of Gauss point used in c  
 c Gauss quadrature integration (-). c  
 c FREQ Frequency (Hz); user specified. c  
 1770 c HEAT Heat release per unit mass of mixture (Btu/lbm). c  
 c OMEGA Angular frequency; 2\*PI\*fhertz (rad/sec). c  
 c PMEAN Uniform steady-state pressure in burner (psi). c  
 c User specified. c  
 c RR Radial location in the burner (ft). Computed c  
 1775 c and used for output purposes. c  
 c THETA Angle in burner measured clockwise (looking c  
 c downstream) from arbitrary zero reference (deg). c  
 c T1 Maximum combustor temperature (F). User specified c  
 c (Assumed to be in head end of the burner). c  
 1780 c T2 Minimum combustor temperature (F). User specified c  
 c (Assumed to be at exhaust end of the burner). c  
 c TMEAN Mean combustor temperature (F). Linear variation c  
 c is assumed from head end to exhaust end. c  
 c XI Nondimensional dimension of Gauss point used in c  
 1785 c Gauss quadrature integration (-). c  
 c ZETA Nondimensional dimension of Gauss point used in c  
 c Gauss quadrature integration (-). c  
 c  
 c Vector Quantities: c  
 1790 c  
 c BC(NODE) Natural (pressure gradient) boundary condition c

	c	at global node NODE (psi/ft).	c
	c	GAMMA(N) Steady specific heat ratio Cp/Cv averaged over	c
	c	element N (-).	c
1795	c	R(I) Eight user-specified radii used to automatically	c
	c	determine the global node coordinants (ft).	c
	c	RHO(N) Steady mass density averaged over element N	c
	c	(lbm/ft <sup>3</sup> )	c
1800	c	S(I) Element-level coefficient vector for the (heat)	c
	c	source term corresponding to Gauss point K. Com-	c
	c	puted in Subroutine INTER3 (ft <sup>3</sup> ). I = 1, 2, ..., 8.	c
	c	SSI(I) Element-level coefficient vector for the (heat)	c
	c	source term summed over all NGP Gauss points of	c
	c	the master element (ft <sup>3</sup> ).	c
1805	c	T(NODE) Local steady temperature at global node NODE (F).	c
	c	User specified.	c
	c	WN(N) Steady wave number averaged over element N	c
	c	(1/ft).	c
1810	c	XX(NODE) x-coordinant of global node NODE (ft). Auto-	c
	c	matically computed.	c
	c	YY(NODE) y-coordinant of global node NODE (ft). Auto-	c
	c	matically computed.	c
	c	ZZ(NODE) z-coordinant of global node NODE (ft). Auto-	c
	c	matically computed.	c
1815	c	ZPLANE(I) z-coordinant of the nine planes forming the eight	c
	c	axial blocks of elements (ft).	c
	c		c
	c	Matrix Quantities:	c
	c		c
1820	c	SS(I,J) Element-level t-component of the "stiffness"	c
	c	matrix for node pair I,J of the master element	c
	c	corresponding to Gauss point K; computed in Sub-	c
	c	routine INTER3 (ft). I,J = 1, 2, ..., 8.	c
1825	c	SSIJ(I,J) Element-level t-component of the "stiffness"	c
	c	matrix for node pair I,J of the master element	c
	c	summed over all NGP Gauss points (unassembled)	c
	c	(ft). I,J = 1, 2, ..., 8.	c
	c	SX(I,J) Element-level x-component of the admittance	c
	c	matrix for node pair I,J of the master element	c
1830	c	corresponding to Gauss point K; computed in	c
	c	Subroutine INTER3 (ft <sup>2</sup> ). I,J = 1, 2, ..., 8.	c
	c	SY(I,J) Element-level y-component of the admittance	c
	c	matrix for node pair I,J of the master element	c
	c	corresponding to Gauss point K; computed in	c
1835	c	Subroutine INTER3 (ft <sup>2</sup> ). I,J = 1, 2, ..., 8.	c
	c	SZ(I,J) Element-level z-component of the admittance	c
	c	matrix for node pair I,J of the master element	c
	c	corresponding to Gauss point K; computed in	c
	c	Subroutine INTER3 (ft <sup>2</sup> ). I,J = 1, 2, ..., 8.	c
1840	c	SSX(I,J) Element-level x-component of the admittance	c
	c	matrix for node pair I,J of the master element	c
	c	summed over all NGP Gauss points (unassembled)	c
	c	(ft <sup>2</sup> ). I,J = 1, 2, ..., 8.	c
	c	SSY(I,J) Element-level y-component of the admittance	c
1845	c	matrix for node pair I,J of the master element	c
	c	summed over all NGP Gauss points (unassembled)	c



```

c          (ft^2). I,J = 1, 2, ..., 8. c
c      SSZ(I,J) Element-level z-component of the admittance c
c          matrix for node pair I,J of the master element c
1850 c          summed over all NGP Gauss points (unassembled) c
c          (ft^2). I,J = 1, 2, ..., 8. c
c      SSXX(I,J) Element-level x-component of the "stiffness" c
c          matrix for node pair I,J of the master element c
1855 c          summed over all NGP Gauss points (unassembled) c
c          (ft). I,J = 1, 2, ..., 8. c
c      SSYY(I,J) Element-level y-component of the "stiffness" c
c          matrix for node pair I,J of the master element c
c          summed over all NGP Gauss points (unassembled) c
c          (ft). I,J = 1, 2, ..., 8. c
1860 c      SSZZ(I,J) Element-level z-component of the "stiffness" c
c          matrix for node pair I,J of the master element c
c          summed over all NGP Gauss points (unassembled) c
c          (ft). I,J = 1, 2, ..., 8. c
c      SXX(I,J) Element-level x-component of the "stiffness" c
1865 c          matrix for node pair I,J of the master element c
c          corresponding to Gauss point K; computed in Sub- c
c          routine INTER3 (ft). I,J = 1, 2, ..., 8. c
c      SYY(I,J) Element-level y-component of the "stiffness" c
1870 c          matrix for node pair I,J of the master element c
c          corresponding to Gauss point K; computed in Sub- c
c          routine INTER3 (ft). I,J = 1, 2, ..., 8. c
c      SZZ(I,J) Element-level z-component of the "stiffness" c
c          matrix for node pair I,J of the master element c
c          corresponding to Gauss point K; computed in Sub- c
1875 c          routine INTER3 (ft). I,J = 1, 2, ..., 8. c
c      X(N,NN) x-coordinant of the NNth node of the Nth element (ft) c
c      Y(N,NN) y-coordinant of the NNth node of the Nth element (ft) c
c      Z(N,NN) z-coordinant of the NNth node of the Nth element (ft) c
1880 c          Complex Quantities: c
c          ADMITIN Acoustic admittance at head end of burner c
c          (ft^3/lbf-sec). User specified. c
1885 c          ADMINOUT Acoustic admittance at exhaust end of burner c
c          (ft^3/lbf-sec). User specified. c
c          ADX Magnitude of acoustic admittance at a hypo- c
c          theoretical boundary normal to the circumferential c
c          direction (ft^3/lbf-sec); always zero in this c
c          analysis because of circumferential symmetry. c
1890 c          ADY Magnitude of acoustic admittance at one of the c
c          radial boundaries of the burner (ft^3/lbf-sec); c
c          zero in current version of program (hard-wall c
c          condition). c
1895 c          ADZ Magnitude of acoustic admittance at one end of c
c          the burner (ft^3/lbf-sec). c
c          FACTOR Non-Dimensional scaling factor used in SOLVE. c
c          PNEW Temporary storage sight for pressure (lbf/in^2) c
c          at node NODE, used during output. c
1900 c          EF(I) Element-level vector representing the sources c
c          and boundary conditions at node I of the master c
c          element; corresponds to the right-hand side of c

```

```

c          the system being solved (lbf/ft). c
c          GF(NODE) Assembled global vector in banded-matrix form c
c          representing the sources and boundary conditions c
1905 c          at node NODE; the right-hand side of the system c
c          being solved (lbf/ft). After execution of Sub- c
c          routine SOLVE, acoustic pressure (lbf/ft^2). c
c          P(NODE) Local steady pressure at global node NODE (psi). c
c          User specified. c
1910 c          Q(NODE) Local steady heat release per unit mass of mix- c
c          ture at global node NODE (Btu/lbm of mixture). c
c          User specified. c
c          QQ(N) Amplitude of unsteady heat release (source func- c
c          tion) averaged over element N (ft-lbf/lbm mixture). c
1915 c          EF(I) Element-level vector representing the sources c
c          and boundary conditions at node I of the master c
c          element; corresponds to the right-hand side of c
c          the system being solved (lbf/ft). c
c          EK(I,J) Element-level "stiffness" matrix for node pair c
1920 c          I,J of the master element; corresponds to the c
c          coefficient matrix of the system being solved (ft). c
c          GK(NROW, Assembled global "stiffness" matrix in banded- c
c          NCOL) matrix form for use in Subroutine SOLVE (ft). c
c c
1925 ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c          SUBROUTINE DYNAMITE(BB,EETTA,PROPS) c
c
c          Specify double precision for all real and complex variables, c
1930 c          then dimension all vector and matrix quantities. c
c
c          INTEGER IFULL, NEM, NTHETA, NR, NZ, NPE, NQ1, NQ2,
& N1, N2, N3, N4, NTHETA1, NR1, NZ1, NEQ,
& NODE, NODESA, CNODE, I,J,K, N, NN, NHBW,
1935 & NW, NROW, NCOL, NGP,
& NRA, NZA, NRB, NZB, NZTOT,
& NRA1,NZA1,NRB1,NZB1,NZTOT1
c          INTEGER MC(1700,8), NGEOM(4) c
c
1940 c          REAL*8 A, COEFF, DZ, XI,ETA,ZETA, FREQ, HEAT, OMEGA,
& PMEAN, RR, THETA, T1, T2, TMEAN, ANGLE
c          REAL*8 X(1700,8), Y(1700,8), Z(1700,8), SX(8,8),
& SY(8,8), SZ(8,8), XX(2400), YY(2400), ZZ(2400),
& SSX(8,8), SSY(8,8), SSZ(8,8), SSXX(8,8),
1945 & SSYY(8,8), SSZZ(8,8), SSIJ(8,8), SSI(8),
& SXX(8,8), SYY(8,8), SZZ(8,8), SS(8,8), S(8),
& ZPLANE(20), R(8), T(2400),
& GAMMA(1700), RHO(1700), WN(1700),
& BB(4), EETTA(8), PROPS(5) c
1950 c          COMPLEX*16 ADMITIN, ADMITOUT, ADX, ADY, ADZ, PNEW c
c          COMPLEX*16 GF(2400), GK(2400,400), EF(8), EK(8,8),
& P(2400), Q(2400), QQ(1700) c
c
1955 c          Initialize values of IFULL, NEM, NTHETA, NR, NZ, NPE, NQ1,
c          NQ2, DZ, A, N1, N2, N3, N4, R, and ZPLANE. c

```

c	***NOTE*** These values are user defined! ***	c
c		
1960	IFULL = 0 NEM = 2832 NTHETA = 48 NR = 7 NZ = 5 NPE = 8	
1965	NQ1 = 1 NQ2 = 76 DZ = 0.60 A = 1.0D0 N1 = 1 N2 = 72 N3 = 1321 N4 = 1416 R(1) = 0.0D0 R(2) = 1.0D0*(1.417D0/2.0D0)/7.0D0	
1970	R(3) = 2.0D0*(1.417D0/2.0D0)/7.0D0 R(4) = 3.0D0*(1.417D0/2.0D0)/7.0D0 R(5) = 4.0D0*(1.417D0/2.0D0)/7.0D0 R(6) = 5.0D0*(1.417D0/2.0D0)/7.0D0 R(7) = 6.0D0*(1.417D0/2.0D0)/7.0D0	
1975	R(8) = 1.417D0/2.0D0 ZPLANE(1) = 0.0d0 ZPLANE(2) = 0.05d0 ZPLANE(3) = 2.05d0 ZPLANE(4) = 4.05d0	
1980	ZPLANE(5) = 6.05d0 ZPLANE(6) = 8.05d0 ZPLANE(7) = 9.84d0 ZPLANE(8) = 11.63d0 ZPLANE(9) = 13.42d0 ZPLANE(10) = 15.21d0 ZPLANE(11) = 17.00d0 ZPLANE(12) = 17.05d0	
1985		
1990		
c		c
1995	Initialize values of HEAT, PMEAN, T1, T2, FREQ. ***NOTE*** These values are user defined in ESTIMAT!!***	c
c		c
2000	HEAT = PROPS(1) PMEAN = PROPS(2) T1 = PROPS(3) T2 = PROPS(4) FREQ = PROPS(5)	
c		c
2005	Initialize values of ADMITIN and ADMITOUT. ***NOTE*** These values are sent by ESTIMAT!!***	c
c		c
2010	ADMITIN = DCMLX(BB(1),BB(2)) ADMITOUT = DCMLX(BB(3),BB(4))	
c		c
2010	Initialize NGEOM vector using NR, NZ, and two other user-defined values.	c
c		c

```

                NGEOM(1) = NR
                NGEOM(2) = NZ
2015          NGEOM(3) = 4
                NGEOM(4) = 6
c
c          Initialize NR1,NZ1,NRA,NRA1,NZA,NZA1,NRB,NRB1,NZB,NZB1,
c          NZTOT, and NZTOT1 using the NGEOM vector.
c
2020          NRA = NGEOM(1)
                NZA = NGEOM(2)
                NRB = NGEOM(3)
                NZB = NGEOM(4)
                NZTOT = NGEOM(2)+NGEOM(4)
2025          NR1 = NGEOM(1)+1
                NZ1 = NGEOM(2)+1
                NRA1 = NRA+1
                NZA1 = NZA+1
                NRB1 = NRB+1
2030          NZB1 = NZB+1
                NZTOT1 = NZTOT+1
c
                TMEAN = (T1 + T2)/2.0D0
                IF(IFULL.EQ.1) THEN
2035          N1 = 1
                N2 = 145
                N3 = 1686
                N4 = 1878
                ENDIF
2040          c
c          If the analysis is over only one-half of the Combustor
c          volume (IFULL = 0), then divide NEM and NTHETA by two.
c
                IF(IFULL.EQ.0) THEN
2045          NEM = NEM/2
                NTHETA = NTHETA/2
                ENDIF
c
c          Convert all physical dimensions into feet. Also compute
2050          the angular frequency OMEGA = 2*pi*FREQ.
c
                DZ = DZ/12.0D0
                DO 140 I = 1, NR1
                    R(I) = R(I)/12.0D0
2055          140 CONTINUE
                DO 150 I = 1, NZTOT1
                    ZPLANE(I) = ZPLANE(I)/12.0D0
                150 CONTINUE
                OMEGA = 2.0D0*(DACOS(-1.0D0))*FREQ
2060          c
                NTHETA1 = NTHETA
                IF(IFULL.EQ.0) NTHETA1 = NTHETA + 1
c
c          Assign nodal values of T, P, and Q.
c
2065          c
c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

```

```

c          Below is the logic for assignment of T, P, and Q, for an          c
c          annular-type Combustor.                                         c
c                                                                              c
2070  cc    NTHETA1 = NTHETA                                                cc
cc    IF(IFULL.EQ.0) NTHETA1 = NTHETA + 1                                  cc
cc    NR1 = NR + 1                                                         cc
cc    NZ1 = NZ + 1                                                         cc
cc    DO 40 I = 1, NZ1                                                     cc
2075  cc    DO 40 J = 1, NR1                                               cc
cc    DO 40 K = 1, NTHETA1                                                 cc
cc    NODE = (I-1)*NTHETA1*NR1 + (J-1)*NTHETA1 + K                       cc
c                                                                              c
c          Default values of T, P, and Q (temporary).                       c
2080  c                                                                              c
cc    T(NODE) = T1 - (T1 - T2)*ZPLANE(I)/ZPLANE(9)                         cc
cc    P(NODE) = PMEAN                                                       cc
cc    Q(NODE) = 0.0D0                                                       cc
cc    IF((NODE.GE.NQ1).AND.(NODE.LE.NQ2)) Q(NODE) = HEAT                 cc
2085  c                                                                              c
c          The following logic allows for an angular distribution of heat.   c
c                                                                              c
cc    IF(NODE.EQ.53) ANGLE = 0.261799388D0                                 cc
cc    IF(NODE.EQ.57) ANGLE = 0.785398163D0                                 cc
2090  cc    IF(NODE.EQ.61) ANGLE = 1.308996939D0                           cc
cc    IF(NODE.EQ.65) ANGLE = 1.832595715D0                                 cc
cc    IF(NODE.EQ.69) ANGLE = 2.35619449D0                                 cc
cc    IF(NODE.EQ.73) ANGLE = 2.879793266D0                                 cc
cc    Q(NODE) = HEAT*DCOS(ANGLE)                                           cc
2095  cc 40 CONTINUE                                                         cc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c                                                                              c
c          Below is the logic for the assignment of P, T, and Q, for a      c
c          straight-pipe Can-type Combustor.                                c
2100  c                                                                              c
c          do 40 i = 1,nztot1,1                                           c
c          cnode = (((ntheta1*nr)+1)*(i-1))+1                               c
c          node = cnode                                                     c
2105  c          t(node) = t1-(t1-t2)*zplane(i)/zplane(nztot1)           c
c          p(node) = pmean                                                 c
c          q(node) = 0.0d0                                                 c
c          if ((node.ge.nq1).and.(node.le.nq2)) then                       c
c          q(node) = heat                                                 c
c          end if                                                         c
2110  c          do 42 j = 2,nr1,1                                         c
c          do 44 k = 1,ntheta1,1                                           c
c          node = cnode+((j-2)*ntheta1)+k                                   c
c          t(node) = t1-(t1-t2)*zplane(i)/zplane(nztot1)                 c
c          p(node) = pmean                                                 c
2115  c          q(node) = 0.0d0                                           c
c          if ((node.ge.nq1).and.(node.le.nq2)) then                       c
c          q(node) = heat                                                 c
c          end if                                                         c
c          44 continue                                                     c
2120  c          42 continue                                               c

```

```

c 40    Continue
c
c
c
2125  c    Below is the logic for the assignment of P, T, and Q, for a
c        can-type combustor (METC geometry).
c
c
        NODESA = ((NTHETA1*NRA)+1)*NZA1
2130  DO 40 I = 1,NZTOT1,1
c        IF (I.LE.NZA1) THEN
c          CNODE = (((NTHETA1*NR)+1)*(I-1))+1
c        ELSE IF (I.GT.NZA1) THEN
c          CNODE = NODESA + (((NTHETA1*NRB)+1)*(I-(NZA1+1)))+1
c          NR1 = NRB1
2135  END IF
c        NODE = CNODE
c        T(NODE) = T1-(T1-T2)*ZPLANE(I)/ZPLANE(NZTOT1)
c        P(NODE) = PMEAN
c        Q(NODE) = 0.0D0
2140  IF ((NODE.GE.NQ1).AND.(NODE.LE.NQ2)) THEN
c        Q(NODE) = HEAT
c        END IF
c        DO 42 J = 2,NR1,1
c          DO 44 K = 1,NTHETA1,1
2145  NODE = CNODE+((J-2)*NTHETA1)+K
c        T(NODE) = T1-(T1-T2)*ZPLANE(I)/ZPLANE(NZTOT1)
c        P(NODE) = PMEAN
c        Q(NODE) = 0.0D0
2150  IF ((NODE.GE.NQ1).AND.(NODE.LE.NQ2)) THEN
c        Q(NODE) = HEAT
c        END IF
44    CONTINUE
42    CONTINUE
        NR1 = NR+1
2155  40    CONTINUE
c
c    Assign the total number of nodes to NEQ.
c
c
2160  c    NEQ = NODE
c
c    Call Subroutine COORDS, which computes the global x,y,z
c    coordinates, XX(NODE),YY(NODE), and ZZ(NODE), of node NODE.
c
c
2165  CALL COORDS(IFULL, NTHETA, NGEOM, DZ, R, ZPLANE, XX, YY, ZZ)
c
c    Call Subroutine CONNECT, which computes the connectivity
c    matrix, MC(N,NN), where N is the element index, NN is the
c    local node index, and MC is the global node index.
c
c
2170  CALL CONNECT(IFULL, NTHETA, NGEOM, MC)
c
c    Compute the narrow half-bandwidth, NHBW, of the banded matrix.
c
c
        NHBW = 0

```

```

2175      DO 120 N = 1, NEM
          DO 120 I = 1, NPE
          DO 120 J = 1, NPE
          NW = IABS(MC(N,I) - MC(N,J)) + 1
          IF(NHBW.LT.NW) NHBW = NW
2180      120 CONTINUE
          c
          c          Initialize the global stiffness matrix and force vector to zero.
          c
          DO 70 I = 1, NEQ
          GF(I) = 0.0D0
          DO 70 J = 1, NHBW
          GK(I,J) = 0.0D0
          70 CONTINUE
          c
2190      c          Loop through the elements (there are NEM elements; N is the
          c          index of the current element). Within this loop, the element-
          c          level stiffness matrix and force vector for element N is
          c          evaluated, and then inserted into the global matrix and vector.
          c
2195      DO 100 N = 1, NEM
          c
          c          Establish the x,y,z coordinates of the global nodes in terms
          c          of the connectivity matrix and local nodes.
          c
2200      DO 60 NN = 1, NPE
          NODE = MC(N,NN)
          X(N,NN) = XX(NODE)
          Y(N,NN) = YY(NODE)
          Z(N,NN) = - ZZ(NODE)
2205      60 CONTINUE
          c
          c          Call Subroutine WEIGHT, which computes the node-weighted
          c          steady-flow coefficient values for each element.
          c
2210      CALL WEIGHT(N,NPE,OMEGA,A,T,P,Q,RHO,GAMMA,WN,QQ,X,Y,Z,MC)
          c
          c          Initialize the values of the coefficient matrices for the
          c          current element (element N).
          c
2215      DO 10 I = 1, NPE
          SSI(I) = 0.0D0
          DO 10 J = 1, NPE
          SSXX(I,J) = 0.0D0
          SSYY(I,J) = 0.0D0
          SSZZ(I,J) = 0.0D0
          SSIJ(I,J) = 0.0D0
          SSX(I,J) = 0.0D0
          SSY(I,J) = 0.0D0
          SSZ(I,J) = 0.0D0
2220      10 CONTINUE
          c
          c          Compute values of the coefficient matrix for the current
          c          element (element N). This assigns the magnitude of the
          c

```

```

2230 c      Gauss Quadrature coordinates. This will change if the current      c
c      scheme of quadrature integration is changed (usually due to      c
c      a different number of Gauss points (usually due to a diff-      c
c      erent type of master element.))                                  c
c
2235 c      COEFF = 1.0D0/DSQRT(3.0D0)                                          c
c
c      Set the number of Gauss Quadrature points equal to eight      c
c      and then loop through all eight points.                          c
c
2240 c      NGP = 8
c      DO 50 K = 1, NGP
c
c      Set the coordinates of the current Gauss point.                  c
c
2245 c      IF(K.EQ.1) THEN
c      XI = - COEFF
c      ETA = - COEFF
c      ZETA = - COEFF
c      ENDIF
c
2250 c      IF(K.EQ.2) THEN
c      XI = COEFF
c      ETA = - COEFF
c      ZETA = - COEFF
c      ENDIF
c
2255 c      IF(K.EQ.3) THEN
c      XI = COEFF
c      ETA = COEFF
c      ZETA = - COEFF
2260 c      ENDIF
c
c      IF(K.EQ.4) THEN
c      XI = - COEFF
c      ETA = COEFF
2265 c      ZETA = - COEFF
c      ENDIF
c
c      IF(K.EQ.5) THEN
c      XI = - COEFF
c      ETA = - COEFF
2270 c      ZETA = COEFF
c      ENDIF
c
c      IF(K.EQ.6) THEN
c      XI = COEFF
c      ETA = - COEFF
2275 c      ZETA = COEFF
c      ENDIF
c
2280 c      IF(K.EQ.7) THEN
c      XI = COEFF
c      ETA = COEFF
c      ZETA = COEFF

```



```

2285   ENDIF
c
IF(K.EQ.8) THEN
    XI = - COEFF
    ETA = COEFF
    ZETA = COEFF
2290   ENDIF
c
c      Call Subroutine INTER3, which computes the Jacobian matrix,
c      its determinant, and its inverse, all of which are functions
c      of the current element, N, and the current Gauss point, K.
2295   CALL INTER3(N,XI,ETA,ZETA,X,Y,Z,SX,SY,SZ,SXX,SY,SZZ,SS,S)
c
c      Perform Gauss Quadrature integration by adding the current
c      values of the Coefficient matrix, which are evaluated at the
2300   current Gauss point, to the Current sum.
c
DO 30 I = 1, NPE
    SSI(I) = SSI(I) + S(I)
DO 30 J = 1, NPE
2305   SSXX(I,J) = SSXX(I,J) + SXX(I,J)
    SSYY(I,J) = SSYY(I,J) + SY(I,J)
    SSZZ(I,J) = SSZZ(I,J) + SZZ(I,J)
    SSLJ(I,J) = SSLJ(I,J) + SS(I,J)
    SSX(I,J) = SSX(I,J) + SX(I,J)
2310   SSY(I,J) = SSY(I,J) + SY(I,J)
    SSZ(I,J) = SSZ(I,J) + SZ(I,J)
30 CONTINUE
50 CONTINUE
c
2315   c      Compute the element-level stiffness matrix, EK and force
c      vector, EF.
c
    ADX = 0.0D0
    ADY = 0.0D0
2320   ADZ = 0.0D0
c
c      This is where the inclusion of BOTH components of the
c      admittance calls for a multiplication by j=sqrt(-1).
c
2325   IF((N.GE.N1).AND.(N.LE.N2)) THEN
    ADZ = DCMLPX((-DIMAG(ADMITIN)),DREAL(ADMITIN))
    END IF
    IF((N.GE.N3).AND.(N.LE.N4)) THEN
2330   ADZ = DCMLPX((-DIMAG(ADMITOUT)),DREAL(ADMITOUT))
    END IF
    DO 80 I = 1, NPE
    EF(I) = - WN(N)*WN(N)*RHO(N)*(GAMMA(N)-1.0D0)*QQ(N)*SSI(I)
    DO 80 J = 1, NPE
2335   EK(I,J) = SSXX(I,J)+SSYY(I,J)+SSZZ(I,J)-(WN(N)*WN(N))*SSLJ(I,J)+
    & OMEGA*RHO(N)*(ADX*SSX(I,J)+ADY*SSY(I,J)+ADZ*SSZ(I,J))
    80 CONTINUE
c

```

```

c      At this point the evaluation of the elemental stiffness matrix      c
c      and force vector is complete for element N. The next step is      c
2340 c      to insert these into the global matrix and vector.              c
c                                                                           c
      DO 90 I = 1, NPE
      NROW = MC(N,I)
      GF(NROW) = GF(NROW) + EF(I)
2345 DO 90 J = 1, NPE
      NCOL = MC(N,J) - NROW + 1
      IF(NCOL) 95, 95, 110
      110 CONTINUE
      GK(NROW,NCOL) = GK(NROW,NCOL) + EK(I,J)
2350 95 CONTINUE
      90 CONTINUE
      100 CONTINUE
c
c      The 100 CONTINUE statement marks the end of the loop that      c
2355 c      covered every element. With the global stiffness matrix and      c
c      force vector determined, it is now time to solve the NEQ      c
c      equations using subroutine SOLVE.                                  c
c                                                                           c
      CALL SOLVE(NEQ, NHBW, GK, GF)
2360 c
c      Write out the pressure as a function of location.                cc
cc                                                                           cc
cc      OPEN(UNIT=5,FILE='OUTPUT')                                     cc
cc      IF(IFULL.EQ.1) THEN                                           cc
2365 cc      WRITE(5,5)                                                 cc
cc      ELSE                                                           cc
cc      WRITE(5,6)                                                     cc
cc      ENDIF                                                         cc
cc      WRITE(*,7) PMEAN, TMEAN                                       cc
2370 cc      WRITE(*,1) FREQ                                           cc
cc      DO 130 I = 1, NZ1                                             cc
cc      DO 130 J = 1, NR1                                             cc
cc      DO 130 K = 1, NTHETA1                                         cc
cc      NODE = (I-1)*NTHETA1*NR1 + (J-1)*NTHETA1 + K                cc
2375 cc      IF(K.EQ.1) THEN                                           cc
cc      RR = DSQRT(XX(NODE)*XX(NODE) + YY(NODE)*YY(NODE))           cc
cc      WRITE(5,2)                                                     cc
cc      WRITE(5,8) RR, ZPLANE(I)                                       cc
cc      WRITE(5,12)                                                    cc
2380 cc      WRITE(5,2)                                                 cc
cc      ENDIF                                                         cc
cc      THETA = (K-1)*7.5D0                                           cc
cc      PNEW = GF(NODE)/144.0D0                                         cc
cc      WRITE(5,9) NODE, THETA, PNEW                                    cc
2385 cc 130 CONTINUE                                                  cc
cc      CLOSE (5)                                                     cc
cc                                                                           cc
cc      FORMAT statements.                                             cc
cc                                                                           cc
2390 cc 1 FORMAT(11X, 'Frequency =', F8.3, ' Hz')                       cc
cc 2 FORMAT(5x, ' ')                                                  cc

```

```

cc 4 FORMAT(5X, I5, 5X, 3F10.3, 5X, E12.5)
cc 5 FORMAT(12X, 'Full Burner Analysis')
cc 6 FORMAT(12X, 'Half Burner Analysis')
2395 cc 7 FORMAT(4X, 'Pmean = ', F5.1, ' psi', 3X, 'Tmean = ', F6.1, ' F')
cc 8 FORMAT(5X, 'Radius = ', F5.3, ' ft', 5X, 'Z = ', F5.3, ' ft')
cc 9 FORMAT(5X, I5, 5X, F5.1, 5X, E12.5)
cc 12 FORMAT(7X, 'Node', 2X, 'Theta (deg)', 2X, 'Press (psi)')
cc 14 FORMAT(10X, F8.3, 5X, E12.5)
2400 c
c
c      This commented section can be used to write the entire
c      calculated complex pressure field to an output file.
c
2405 c      OPEN(UNIT=5,FILE='OUTPUT.DAT')
c      IF(IFULL.EQ.1) THEN
c        WRITE(5,5)
c      ELSE
c        WRITE(5,7)
2410 c      ENDIF
c      WRITE(5,9) PMEAN, TMEAN
c      WRITE(5,1) FREQ
c      DO 130 I = 1,NZTOT1,1
c        IF (I.LE.NZA1) THEN
2415 c          CNODE = (((NTHETA1*NR)+1)*(I-1))+1
c        ELSE IF (I.GT.NZA1) THEN
c          CNODE = NODESA + (((NTHETA1*NRB)+1)*(I-(NZA1+1)))+1
c          NR1 = NRB1
c        END IF
2420 c      NODE = CNODE
c      PNEW = GF(NODE)/144.0D0
c      ANGLE = ATAN2(DIMAG(PNEW),DREAL(PNEW))*(180.0/DACOS(-1.0D0))
c      WRITE(5,15) ZPLANE(I)
c      WRITE(5,17) NODE,PNEW
2425 c      DO 132 J = 2,NR1,1
c        DO 134 K = 1,NTHETA1,1
c          NODE = CNODE+((J-2)*NTHETA1)+K
c          IF(K.EQ.1) THEN
2430 c            RR = DSQRT(XX(NODE)*XX(NODE) + YY(NODE)*YY(NODE))
c            WRITE(5,3)
c            WRITE(5,11) RR, ZPLANE(I)
c            WRITE(5,13)
c            WRITE(5,3)
c          ENDIF
2435 c          THETA = (K-1)*7.5D0
c          PNEW = GF(NODE)/144.0D0
c          ANGLE=ATAN2(DIMAG(PNEW),DREAL(PNEW))*(180.0/DACOS(-1.0D0))
c          WRITE(5,19) NODE,THETA,PNEW
c 134      CONTINUE
2440 c 132      CONTINUE
c          NR1 = NR+1
c 130      CONTINUE
c
c      Format Statements
2445 c
c 1 FORMAT(11X, 'Frequency =', F8.3, ' Hz')

```



```

c      Can-type geometry.
c
c      INTEGER      NRA, NZA, NRB, NZB, NZTOT,
2505 &      NRA1,NZA1,NRB1,NZB1,NZTOT1, NODESA
c
c      Assign values to Constants.
c
c      PI = DACOS(-1.0D0)
2510 PI180 = PI/180.0D0
c      DZ = 0.0
c
c      Determine whether the analysis is half- or full-combustor,
c      and adjust NTHETA1 accordingly.
c
2515 NTHETA1 = NTHETA
c      IF(IFFULL.EQ.0) NTHETA1 = NTHETA + 1
c
c
c
c
c      Commented below is the logic for the x,y,z coordinates of a
2520 c      straight, constant-radius, can-type burner (no-annulus).
c
c      NR = NGEOM(1)
c      NZ = NGEOM(2)
c      NR1 = NR+1
2525 c      NZ1 = NZ+1
c      NZTOT1 = NZ1
c      DO 41 I = 1,NZTOT1,1
c      CNODE = (((NTHETA1*NR)+1)*(I-1))+1
c      NODE = CNODE
2530 c      X(NODE) = 0.0
c      Y(NODE) = 0.0
c      Z(NODE) = ZZ(I)
c      DO 43 J = 2,NR1,1
c      DO 45 K = 1,NTHETA1,1
2535 c      NODE = CNODE+((J-2)*NTHETA1)+K
c      THETA = (K-1)*PI180*7.5D0
c      X(NODE) = R(J)*DSIN(THETA)
c      Y(NODE) = R(J)*DCOS(THETA)
c      Z(NODE) = ZZ(I)
2540 c 45 CONTINUE
c 43 CONTINUE
c 41 CONTINUE
c
c
c
c
c      Below is the logic for the x,y,z coordinates of the METC
2545 c      burner. It is a can-type with two different radii (from
c      outside it resembles a soda Can sitting on a Coffee tin, with
c      both sharing the Common Central axis.)
c
c
2550 c
c      Initialize NR1,NZ1,NRA,NRA1,NZA,NZA1,NRB,NRB1,NZB,NZB1,
c      NZTOT, and NZTOT1 using the NGEOM vector. These values are
c      needed for the METC geometry.
c
c

```

```

2555   NRA = NGEOM(1)
      NZA = NGEOM(2)
      NRB = NGEOM(3)
      NZB = NGEOM(4)
      NZTOT = NGEOM(2)+NGEOM(4)
2560   NR1 = NGEOM(1)+1
      NZ1 = NGEOM(2)+1
      NRA1 = NRA+1
      NZA1 = NZA+1
      NRB1 = NRB+1
2565   NZB1 = NZB+1
      NZTOT1 = NZTOT+1
      c
      c   Compute the number of nodes in the first cylindrical part of
      c   the METC geometry.
2570   c
      NODESA = ((NTHETA1*NRA)+1)*NZA1
      c
      c   Loop through all nodes and determine the x,y,z coordinates.
      c
2575   DO 41 I = 1,NZTOT1,1
      IF (I.LE.NZA1) THEN
        CNODE = (((NTHETA1*NRA)+1)*(I-1))+1
      ELSE IF (I.GT.NZA1) THEN
        CNODE = NODESA + (((NTHETA1*NRB)+1)*(I-(NZA1+1)))+1
2580   NR1 = NRB1
      END IF
      NODE = CNODE
      X(NODE) = 0.0d0
      Y(NODE) = 0.0d0
2585   Z(NODE) = ZZ(I)
      DO 43 J = 2,NR1,1
        DO 45 K = 1,NTHETA1,1
          NODE = CNODE+((J-2)*NTHETA1)+K
          THETA = (K-1)*PI180*7.5D0
2590   X(NODE) = R(J)*DSIN(THETA)
          Y(NODE) = R(J)*DCOS(THETA)
          Z(NODE) = ZZ(I)
45    CONTINUE
43    CONTINUE
2595   NR1 = NGEOM(1)+1
41    CONTINUE
      c
      c   Return to calling routine.
      c
2600   RETURN
      END
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      c
      c   SUBROUTINE CONNECT
      c
2605   c
      ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      c
      c   This subroutine automatically generates the Connectivity
      c   matrix, MC(N,K), where N is the element index, K is the

```

```

2610  c      local node index and MC is the global node index.          c
      c
      ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      c
      SUBROUTINE CONNECT(IFULL, NTHETA, NGEOM, MC)
2615  c
      c      Declare integer values needed for the connectivity array.    c
      c
      INTEGER              IFULL, N, NTHETA, NR, NZ, NTHETA1
      &                    IR, IZ, ITHETA, INDEX
2620  &                    INTEGER              NGEOM(4), MC(1700,8)
      c
      c      Declare integer variables that are needed for the METC      c
      c      can-type geometry.                                          c
      c
2625  INTEGER              NRA, NZA, NRB, NZB, NZTOT,
      &                    NRA1, NZA1, NRB1, NZB1, NZTOT1,
      &                    NELEMA, NODESA1, NODESA2, M1, M2, M3, M4, M5
      c
      c      Determine whether the analysis is half- or full-combustor,   c
2630  c      and adjust NTHETA1 accordingly.                                c
      c
      NTHETA1 = NTHETA
      IF(IFULL.EQ.0) NTHETA1 = NTHETA + 1
      ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
2635  cc      Double commented is the logic for the connectivity array for cc
      cc      a constant radius, annular-type burner.                    cc
      cc
      cc      NR = NGEOM(1)                                               cc
      cc      NZ = NGEOM(2)                                               cc
2640  cc      NTHETA1 = NTHETA                                             cc
      cc      IF(IFULL.EQ.0) NTHETA1 = NTHETA + 1                         cc
      cc      DO 20 IZ = 1, NZ                                             cc
      cc      DO 20 IR = 1, NR                                             cc
      cc      DO 20 ITHETA = 1, NTHETA                                     cc
2645  cc      INDEX = 1                                                    cc
      cc      IF((IFULL.EQ.1).AND.(ITHETA.EQ.NTHETA)) INDEX = INDEX - NTHETA cc
      cc      N = (IZ-1)*NR*NTHETA + (IR-1)*NTHETA + ITHETA              cc
      cc      MC(N,1) = (NR+1)*(NTHETA1)*(IZ-1)+(NTHETA1)*(IR-1)+ITHETA  cc
2650  cc      MC(N,2) = (NR+1)*(NTHETA1)*(IZ-1)+(NTHETA1)*(IR-1)+ITHETA+INDEX cc
      cc      MC(N,3) = (NR+1)*(NTHETA1)*(IZ-1)+(NTHETA1)*IR+ITHETA+INDEX cc
      cc      MC(N,4) = (NR+1)*(NTHETA1)*(IZ-1)+(NTHETA1)*IR+ITHETA      cc
      cc      MC(N,5) = (NR+1)*(NTHETA1)*IZ+(NTHETA1)*(IR-1)+ITHETA      cc
      cc      MC(N,6) = (NR+1)*(NTHETA1)*IZ+(NTHETA1)*(IR-1)+ITHETA+INDEX cc
      cc      MC(N,7) = (NR+1)*(NTHETA1)*IZ+(NTHETA1)*IR+ITHETA+INDEX  cc
2655  cc      MC(N,8) = (NR+1)*(NTHETA1)*IZ+(NTHETA1)*IR+ITHETA        cc
      cc      20 CONTINUE                                                 cc
      c
      ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
2660  c      Commented below is the logic for the connectivity array for a c
      c      straight, constant-radius, can-type burner (no annulus).    c
      c
      c      NR = NGEOM(1)                                               c
      c      NZ = NGEOM(2)                                               c

```

```

c DO 20 IZ = 1,NZ,1 c
2665 c DO 22 IR = 1,NR,1 c
c DO 24 ITHETA = 1,NTHETA,1 c
c INDEX = 1 c
c IF ((IFULL.EQ.1).AND.(ITHETA.EQ.NTHETA)) THEN c
c INDEX = INDEX-NTHETA c
2670 c END IF c
c N = (IZ-1)*NR*NTHETA + (IR-1)*NTHETA + ITHETA c
c IF (IR.EQ.1) THEN c
c MC(N,1)=(((NTHETA1*NR)+1)*(IZ-1))+1 c
c MC(N,2)=(((NTHETA1*NR)+1)*(IZ-1))+1 c
2675 c MC(N,3)=(((NTHETA1*NR)+1)*(IZ-1))+1+ITHETA+INDEX c
c MC(N,4)=(((NTHETA1*NR)+1)*(IZ-1))+1+ITHETA c
c MC(N,5)=(((NTHETA1*NR)+1)*(IZ))+1 c
c MC(N,6)=(((NTHETA1*NR)+1)*(IZ))+1 c
c MC(N,7)=(((NTHETA1*NR)+1)*(IZ))+1+ITHETA+INDEX c
2680 c MC(N,8)=(((NTHETA1*NR)+1)*(IZ))+1+ITHETA c
c ELSE IF (IR.GT.1) THEN c
c MC(N,1)=(((NTHETA1*NR)+1)*(IZ-1))+1+((IR-2)*NTHETA1)+ITHETA c
c MC(N,2)=(((NTHETA1*NR)+1)*(IZ-1))+1+((IR-2)*NTHETA1)+ITHETA+INDEX c
c MC(N,3)=(((NTHETA1*NR)+1)*(IZ-1))+1+((IR-1)*NTHETA1)+ITHETA+INDEX c
2685 c MC(N,4)=(((NTHETA1*NR)+1)*(IZ-1))+1+((IR-1)*NTHETA1)+ITHETA c
c MC(N,5)=(((NTHETA1*NR)+1)*(IZ))+1+((IR-2)*NTHETA1)+ITHETA c
c MC(N,6)=(((NTHETA1*NR)+1)*(IZ))+1+((IR-2)*NTHETA1)+ITHETA+INDEX c
c MC(N,7)=(((NTHETA1*NR)+1)*(IZ))+1+((IR-1)*NTHETA1)+ITHETA+INDEX c
c MC(N,8)=(((NTHETA1*NR)+1)*(IZ))+1+((IR-1)*NTHETA1)+ITHETA c
2690 c END IF c
c 24 CONTINUE c
c 22 CONTINUE c
c 20 CONTINUE c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
2695 c c
c Below is the logic for the Connectivity array of the METC c
c burner. It is a Can-type with two different radii (from c
c outside it resembles a soda Can sitting on a Coffee tin, with c
c both sharing the Common Central axis.) c
2700 c c
c Initialize NR1,NZ1,NRA,NRA1,NZA,NZA1,NRB,NRB1,NZB,NZB1, c
c NZTOT, and NZTOT1 using the NGEOM vector. These values are c
c needed for the METC geometry. c
c
2705 NRA = NGEOM(1)
NZA = NGEOM(2)
NRB = NGEOM(3)
NZB = NGEOM(4)
NZTOT = NGEOM(2)+NGEOM(4)
2710 NR1 = NGEOM(1)+1
NZ1 = NGEOM(2)+1
NRA1 = NRA+1
NZA1 = NZA+1
NRB1 = NRB+1
2715 NZB1 = NZB+1
NZTOT1 = NZTOT+1

```



```

c      Compute the number of elements in the first cylinder of the      c
c      METC can-type combustor.                                         c
2720  c      NELEMA = NTHETA*NRA*NZA                                       c
c
c      Compute the number of nodes in the first cylinder of the        c
c      can-type combustor (the nodes in the first NZA ZPlanes.)      c
2725  c      NODESA1 = ((NTHETA1*NRA)+1)*(NZA)                             c
c
c      Compute the number of nodes in the first cylinder of the        c
c      can-type combustor (the nodes in the first NZA+1 ZPlanes.)    c
2730  c      NODESA2 = ((NTHETA1*NRA)+1)*(NZA1)                          c
c
c      DO 20 IZ = 1,NZTOT,1
c      IF (IZ.LE.NZA) THEN
2735  c      M1 = 0
c      M2 = 0
c      M3 = 0
c      M4 = 0
c      M5 = 0
2740  c      NR = NRA
c      ELSE IF (IZ.EQ.NZA1) THEN
c      M1 = NZA
c      M2 = NELEMA
c      M3 = NODESA1
2745  c      M4 = NODESA2
c      M5 = 1
c      NR = NRB
c      ELSE IF (IZ.GT.NZA1) THEN
c      M1 = NZA1
2750  c      M2 = NELEMA+(NRB*NTHETA)
c      M3 = NODESA2
c      M4 = NODESA2
c      M5 = 0
c      NR = NRB
2755  c      END IF
c
c      DO 22 IR = 1,NR,1
c      DO 24 ITHETA = 1,NTHETA,1
c      INDEX = 1
2760  c      IF ((IFULL.EQ.1).AND.(ITHETA.EQ.NTHETA)) THEN
c      INDEX = INDEX-NTHETA
c      END IF
c      N=(IZ-(M1+1))*NR*NTHETA+(IR-1)*NTHETA+ITHETA+M2
c      IF (IR.EQ.1) THEN
2765  c      MC(N,1)=(((NTHETA1*NR)+1)*(IZ-(M1+1)))+1+M3
c      MC(N,2)=(((NTHETA1*NR)+1)*(IZ-(M1+1)))+1+M3
c      MC(N,3)=(((NTHETA1*NR)+1)*(IZ-(M1+1)))+1+ITHETA+INDEX+M3
c      MC(N,4)=(((NTHETA1*NR)+1)*(IZ-(M1+1)))+1+ITHETA+M3
c      MC(N,5)=(((NTHETA1*NR)+1)*(IZ-(M1+M5)))+1+M4
2770  c      MC(N,6)=(((NTHETA1*NR)+1)*(IZ-(M1+M5)))+1+M4
c      MC(N,7)=(((NTHETA1*NR)+1)*(IZ-(M1+M5)))+1+ITHETA+INDEX+M4

```

```

        MC(N,8)=(((NTHETA1*NR)+1)*(IZ-(M1+M5)))+1+ITHETA+M4
        ELSE IF (IR.GT.1) THEN
2775    & MC(N,1)=(((NTHETA1*NR)+1)*(IZ-(M1+1)))+1+((IR-2)*NTHETA1)+
        ITHETA+M3
        & MC(N,2)=(((NTHETA1*NR)+1)*(IZ-(M1+1)))+1+((IR-2)*NTHETA1)+
        ITHETA+INDEX+M3
        & MC(N,3)=(((NTHETA1*NR)+1)*(IZ-(M1+1)))+1+((IR-1)*NTHETA1)+
        ITHETA+INDEX+M3
2780    & MC(N,4)=(((NTHETA1*NR)+1)*(IZ-(M1+1)))+1+((IR-1)*NTHETA1)+
        ITHETA+M3
        & MC(N,5)=(((NTHETA1*NR)+1)*(IZ-(M1+M5)))+1+((IR-2)*NTHETA1)+
        ITHETA+M4
        & MC(N,6)=(((NTHETA1*NR)+1)*(IZ-(M1+M5)))+1+((IR-2)*NTHETA1)+
2785    & ITHETA+INDEX+M4
        & MC(N,7)=(((NTHETA1*NR)+1)*(IZ-(M1+M5)))+1+((IR-1)*NTHETA1)+
        ITHETA+INDEX+M4
        & MC(N,8)=(((NTHETA1*NR)+1)*(IZ-(M1+M5)))+1+((IR-1)*NTHETA1)+
        ITHETA+M4
2790    & END IF
        24 CONTINUE
        22 CONTINUE
        20 CONTINUE
c
2795    c       Return to calling routine.    c
        c
        RETURN
        END
2800    ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
        c
        c                SUBROUTINE WEIGHT                c
        c
        ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
        c
2805    c       This subroutine Computes the node-weighted steady flow     c
        c       coefficient values RHO(N), GAMMA(N), WN(N), AND QQ(N),    c
        c       for each element N.                                        c
        c
        ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
2810    c       SUBROUTINE WEIGHT(N,NPE,OMEGA,A,T,P,Q,RHO,GAMMA,WN,QQ,X,Y,Z,MC)
        c
        c       Declare all non-integer variables as double precision.    c
        c
2815    c       IMPLICIT REAL*8 (A-H,O-Z)
        c       INTEGER N, NPE,>NNLIM, NN, NLOOP,
        c       & N1, N2, N3, N4, N5, N6, N7, N8, N9
        c       REAL*8 OMEGA, A, VOL, DENSE, CPCV, GAMMART, QHEAT
        c       REAL*8 T(2400), RHO(1700), GAMMA(1700),
2820    & WN(1700), X(1700,8), Y(1700,8), Z(1700,8),
        c       & TX(8), TY(8), TZ(8), V(9)
        c       COMPLEX*16 P(2400), Q(2400), QQ(1700)
        c
        c       First establish the global Coordinants of the 8 corner
2825    c       nodes of the element or sub-element whose volume is to

```

```

c      be computed. Loop through the 8 corner nodes but also      c
c      one step must be added for the element itself; that is,    c
c      there are nine volumes to calculate: one for the entire    c
c      element and one each for the 8 sub-elements which compose c
2830  c      it.                                                    c
c
     >NNLIM = NPE + 1
     >DO 10 NN = 1,>NNLIM
     >NLOOP = NN - 1
2835  IF(NLOOP.LE.1) THEN
      N1 = 1
      N2 = 2
      N3 = 3
2840  N4 = 4
      N5 = 5
      N6 = 6
      N7 = 7
      N8 = 8
     >ENDIF
2845  IF(NLOOP.EQ.2) THEN
      N1 = 2
      N2 = 6
      N3 = 7
      N4 = 3
2850  N5 = 1
      N6 = 5
      N7 = 8
      N8 = 4
     >ENDIF
2855  IF(NLOOP.EQ.3) THEN
      N1 = 3
      N2 = 7
      N3 = 8
      N4 = 4
2860  N5 = 2
      N6 = 6
      N7 = 5
      N8 = 1
     >ENDIF
2865  IF(NLOOP.EQ.4) THEN
      N1 = 4
      N2 = 3
      N3 = 7
      N4 = 8
2870  N5 = 1
      N6 = 2
      N7 = 6
      N8 = 5
     >ENDIF
2875  IF(NLOOP.EQ.5) THEN
      N1 = 5
      N2 = 1
      N3 = 4
      N4 = 8
2880  N5 = 6

```

```

      N6 = 2
      N7 = 3
      N8 = 7
ENDIF
2885 IF(NLOOP.EQ.6) THEN
      N1 = 6
      N2 = 5
      N3 = 8
      N4 = 7
2890 N5 = 2
      N6 = 1
      N7 = 4
      N8 = 3
ENDIF
2895 IF(NLOOP.EQ.7) THEN
      N1 = 7
      N2 = 3
      N3 = 2
      N4 = 6
2900 N5 = 8
      N6 = 4
      N7 = 1
      N8 = 5
ENDIF
2905 IF(NLOOP.EQ.8) THEN
      N1 = 8
      N2 = 4
      N3 = 3
      N4 = 7
2910 N5 = 5
      N6 = 1
      N7 = 2
      N8 = 6
ENDIF
2915 IF(NLOOP.EQ.0) THEN
      TX(1) = X(N,N1)
      TY(1) = Y(N,N1)
      TZ(1) = Z(N,N1)
      TX(2) = X(N,N2)
2920 TY(2) = Y(N,N2)
      TZ(2) = Z(N,N2)
      TX(3) = X(N,N3)
      TY(3) = Y(N,N3)
      TZ(3) = Z(N,N3)
2925 TX(4) = X(N,N4)
      TY(4) = Y(N,N4)
      TZ(4) = Z(N,N4)
      TX(5) = X(N,N5)
      TY(5) = Y(N,N5)
2930 TZ(5) = Z(N,N5)
      TX(6) = X(N,N6)
      TY(6) = Y(N,N6)
      TZ(6) = Z(N,N6)
      TX(7) = X(N,N7)
2935 TY(7) = Y(N,N7)

```

```

TZ(7) = Z(N,N7)
TX(8) = X(N,N8)
TY(8) = Y(N,N8)
TZ(8) = Z(N,N8)
2940 ELSE
TX(1) = X(N,N1)
TY(1) = Y(N,N1)
TZ(1) = Z(N,N1)
TX(2) = (X(N,N1)+X(N,N2))/2.0D0
2945 TY(2) = (Y(N,N1)+Y(N,N2))/2.0D0
TZ(2) = (Z(N,N1)+Z(N,N2))/2.0D0
TX(3) = (X(N,N1)+X(N,N2)+X(N,N3)+X(N,N4))/4.0D0
TY(3) = (Y(N,N1)+Y(N,N2)+Y(N,N3)+Y(N,N4))/4.0D0
TZ(3) = (Z(N,N1)+Z(N,N2)+Z(N,N3)+Z(N,N4))/4.0D0
2950 TX(4) = (X(N,N1)+X(N,N4))/2.0D0
TY(4) = (Y(N,N1)+Y(N,N4))/2.0D0
TZ(4) = (Z(N,N1)+Z(N,N4))/2.0D0
TX(5) = (X(N,N1)+X(N,N5))/2.0D0
TY(5) = (Y(N,N1)+Y(N,N5))/2.0D0
2955 TZ(5) = (Z(N,N1)+Z(N,N5))/2.0D0
TX(6) = (X(N,N1)+X(N,N2)+X(N,N5)+X(N,N6))/4.0D0
TY(6) = (Y(N,N1)+Y(N,N2)+Y(N,N5)+Y(N,N6))/4.0D0
TZ(6) = (Z(N,N1)+Z(N,N2)+Z(N,N5)+Z(N,N6))/4.0D0
TX(7) = (X(N,N1)+X(N,N2)+X(N,N3)+X(N,N4)+
2960 & X(N,N5)+X(N,N6)+X(N,N7)+X(N,N8))/8.0D0
TY(7) = (Y(N,N1)+Y(N,N2)+Y(N,N3)+Y(N,N4)+
& Y(N,N5)+Y(N,N6)+Y(N,N7)+Y(N,N8))/8.0D0
TZ(7) = (Z(N,N1)+Z(N,N2)+Z(N,N3)+Z(N,N4)+
& Z(N,N5)+Z(N,N6)+Z(N,N7)+Z(N,N8))/8.0D0
2965 TX(8) = (X(N,N1)+X(N,N4)+X(N,N5)+X(N,N8))/4.0D0
TY(8) = (Y(N,N1)+Y(N,N4)+Y(N,N5)+Y(N,N8))/4.0D0
TZ(8) = (Z(N,N1)+Z(N,N4)+Z(N,N5)+Z(N,N8))/4.0D0
ENDIF
c
2970 c Now call Subroutine TETRA, which actually computes the c
c volume of the element or sub-element by computing and c
c summing the volumes of the five tetrahedrons which compose c
c it. c
c
2975 CALL TETRA(TX, TY, TZ, VOL)
V(NN) = VOL
10 CONTINUE
V(1) = V(2)+V(3)+V(4)+V(5)+V(6)+V(7)+V(8)+V(9)
c
2980 c Finally, Compute the node-volume-weighted values of RHO(N), c
c GAMMA(N), WN(N), and QQ(N) for element N. c
c
CALL DENSITY(N, V, T, P, MC, DENSE)
RHO(N) = DENSE
2985 CALL CPCVAO(N, V, T, MC, CPCV, GAMMART)
GAMMA(N) = CPCV
WN(N) = OMEGA/DSQRT(GAMMART)
CALL HEAT(N, A, V, Q, MC, QHEAT)
QQ(N) = QHEAT
2990 c

```

```

c      Return to calling routine.                                c
c                                                                 c
      RETURN                                                    c
      END
2995 ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c                                                                 c
c      SUBROUTINE TETRA                                          c
c                                                                 c
3000 ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c                                                                 c
c      This subroutine computes the volume of a six-sided      c
c      solid based on knowledge of the global coordinants of   c
c      its eight corners. It accomplishes this by finding the  c
c      volume of the five tetrahedrons which compose it.      c
3005 c                                                                 c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c      SUBROUTINE TETRA(TX, TY, TZ, VOL)                        c
c                                                                 c
3010 c      Dimension vectors and declare double precision for all c
c      variables.                                               c
c                                                                 c
      INTEGER          NT, I, J, K, L                          c
      REAL*8 A1, B1, C1, VOL                                  c
3015 REAL*8 TX(8), TY(8), TZ(8)                                c
c                                                                 c
c      Initialize the volume to zero.                            c
c                                                                 c
      VOL = 0.0D0                                             c
3020 c                                                                 c
c      Loop through the five tetrahedrons that make up th solid. c
c                                                                 c
      DO 10 NT = 1, 5
3025 IF(NT.EQ.1) THEN
      I = 7
      J = 3
      K = 6
      L = 8
      ENDIF
3030 IF(NT.EQ.2) THEN
      I = 4
      J = 1
      K = 3
      L = 8
3035 ENDIF
      IF(NT.EQ.3) THEN
      I = 2
      J = 1
      K = 6
3040 L = 3
      ENDIF
      IF(NT.EQ.4) THEN
      I = 5
      J = 1

```

```

3045     K = 8
        L = 6
        ENDIF
        IF(NT.FQ.5) THEN
3050     I = 3
        J = 1
        K = 6
        L = 8
        ENDIF
3055     A1 = (TY(J)-TY(L))*(TZ(K)-TZ(L))-(TY(K)-TY(L))*(TZ(J)-TZ(L))
        B1 = (TX(K)-TX(L))*(TZ(J)-TZ(L))-(TX(J)-TX(L))*(TZ(K)-TZ(L))
        C1 = (TX(J)-TX(L))*(TY(K)-TY(L))-(TX(K)-TX(L))*(TY(J)-TY(L))
        VOL = VOL + A1*(TX(I)-TX(L)) + B1*(TY(I)-TY(L)) + C1*(TZ(I)-TZ(L))
10 CONTINUE
        VOL = VOL/6.0D0
3060     c                                     c
        c         Return to calling routine.         c
        c                                     c
        RETURN
        END
3065     cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
        c                                     c
        c             SUBROUTINE DENSITY              c
        c                                     c
3070     cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
        c                                     c
        c         This subroutine computes the volume-weighted mass density         c
        c         element n based on the mass densities at the eight corners         c
        c         of the element and the relative volumes occupied by the         c
        c         eight corner nodes.                                                 c
3075     c                                     c
        c         Ideal gas behavior is assumed. Thus,                             c
        c                                     c
        c             RHO = P/R*T,                                                  c
        c                                     c
3080     c         where R (= 53.3 ft-lbf/lbm-r) is the gas constant of air.         c
        c         then RHO is in lbf/ft^3, P is in lbf/ft^2, and T is in R.         c
        c         Pressure and temperature enter the subroutine with units         c
        c         psi and F, respectively. Thus coefficients must be added         c
        c         convert these to lbf/ft^2 and R.                                   c
3085     c                                     c
        cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
        c                                     c
        SUBROUTINE DENSITY(N, V, T, P, MC, RHO)
        c                                     c
3090     c         Declare all real and complex variables as double precision,         c
        c         and dimension the vectors and matrices.                             c
        c                                     c
        INTEGER      N, NN, NNN, NODE, MC(1700,8)
        REAL*8 R, C1, C2, RHO
3095     REAL*8 V(9), T(2400)
        COMPLEX*16  P(2400)
        c                                     c
        c         Establish values of constants.                                     c

```

```

c
3100    R = 53.3D0
        C1 = 144.0D0
        C2 = 460.0D0
c
c      Initialize the mass density to zero.
3105
c      RHO = 0.0D0
c
c      Loop through the eight corner nodes to compute the volume-
c      weighted mass density.
3110
c      DO 10 NN = 2, 9
        NNN = NN - 1
        NODE = MC(N,NNN)
cc      RHO = RHO + V(NN)*P(NODE)*C1/(R*(T(NODE)+C2))
3115      RHO = RHO + V(NN)*DREAL(P(NODE))*C1/(R*(T(NODE)+C2))
        10 CONTINUE
c
c      Divide by the total volume of the element.
c
3120      RHO = RHO/V(1)
c
c      Return to the calling routine.
c
        RETURN
3125      END
cc
c
c      SUBROUTINE CPCVAO
3130
c
c      This subroutine computes the volume-weighted specific
c      heat ratio, GAMMA = Cp/Cv, and the speed of sound,
c      Ao = SQRT(GAMMA*R*T), for air behaving as an ideal gas
3135      with variable specific heats. We use the curve-fit
c      equations from "Engineering Thermodynamics, an Intro-
c      ductory Textbook (Second Edition)," by Jones and Hawkins
c      John Wiley, p. 783). That is, Cp/R is assumed to have
c      the form
3140
c      Cp/R = a + bT + cT^2 + dT^3 + eT^4 ,
c
c      where the values of the coefficients a, b, c, d and e
c      depend on the temperature range. Then
3145
c      GAMMA = Cp/Cv = Cp/R / (Cp/R - 1).
c
c      Note that temperature is read in in F and so must be
c      converted to R in the routine.
3150
c
cc
c

```



```

SUBROUTINE CPCVAO(N, V, T, MC, GAMMA, GAMMART)
c
3155 c      Declare all real variables as double precision, and dimension
c      the vectors and matrices.
c
c      INTEGER      NN, NNN, NODE, MC(1700,8)
3160 c      REAL*8 GAMMA, GAMMAT, GAMMART, R, TR, A,B,c,D,E, CPR
c      REAL*8 V(9), T(2400)
c
c      Initialize the specific heat ratio GAMMA and the product
c      GAMMAT = GAMMA*T to zero. Also set value of the gas constant
c      for air, R.
3165 c
c      GAMMA = 0.0D0
c      GAMMAT = 0.0D0
c      R = 53.3D0
c
3170 c      Loop through the eight corner nodes to Compute the volume-
c      weighted specific heat ratio.
c
c      DO 10 NN = 2, 9
c      NNN = NN - 1
3175 c      NODE = MC(N,NNN)
c
c      Establish the value of the constants a,b,c,d,e depending on
c      the temperature range.
c
3180 c      TR = T(NODE) + 460.0D0
c      IF(TR.LE.1800.0D0) THEN
c      A = 3.653D0
c      B = - 0.7428D-03
c      c = 1.017D-06
3185 c      D = - 0.3280D-09
c      E = 0.02632D-12
c      ELSE
c      A = 3.045D0
c      B = 0.7428D-03
3190 c      c = - 0.1506D-06
c      D = 0.01466D-09
c      E = - 5.426D-16
c      ENDIF
c      CPR = A + B*TR + c*TR*TR + D*TR*TR*TR + E*TR*TR*TR*TR
3195 c      GAMMA = GAMMA + (CPR/(CPR - 1.0D0))*V(NN)
c      GAMMAT = GAMMAT + (CPR/(CPR - 1.0D0))*TR*V(NN)
c      10 CONTINUE
c
c      Divide by the total volume of the element.
3200 c
c      GAMMA = GAMMA/V(1)
c      GAMMART = 32.2D0*R*GAMMAT/V(1)
c
c      Return to the calling routine.
3205 c
c      RETURN

```





```

3315      DSF(2,1) = + COEFF*(1.0D0 - ETA)*(1.0D0 - ZETA)
          DSF(2,2) = - COEFF*(1.0D0 + XI)*(1.0D0 - ZETA)
          DSF(2,3) = - COEFF*(1.0D0 + XI)*(1.0D0 - ETA)
c
3320      DSF(3,1) = + COEFF*(1.0D0 + ETA)*(1.0D0 - ZETA)
          DSF(3,2) = + COEFF*(1.0D0 + XI)*(1.0D0 - ZETA)
          DSF(3,3) = - COEFF*(1.0D0 + XI)*(1.0D0 + ETA)
c
3325      DSF(4,1) = - COEFF*(1.0D0 + ETA)*(1.0D0 - ZETA)
          DSF(4,2) = + COEFF*(1.0D0 - XI)*(1.0D0 - ZETA)
          DSF(4,3) = - COEFF*(1.0D0 - XI)*(1.0D0 + ETA)
c
3330      DSF(5,1) = - COEFF*(1.0D0 - ETA)*(1.0D0 + ZETA)
          DSF(5,2) = - COEFF*(1.0D0 - XI)*(1.0D0 + ZETA)
          DSF(5,3) = + COEFF*(1.0D0 - XI)*(1.0D0 - ETA)
c
3335      DSF(6,1) = + COEFF*(1.0D0 - ETA)*(1.0D0 + ZETA)
          DSF(6,2) = - COEFF*(1.0D0 + XI)*(1.0D0 + ZETA)
          DSF(6,3) = + COEFF*(1.0D0 + XI)*(1.0D0 - ETA)
c
3340      DSF(7,1) = + COEFF*(1.0D0 + ETA)*(1.0D0 + ZETA)
          DSF(7,2) = + COEFF*(1.0D0 + XI)*(1.0D0 + ZETA)
          DSF(7,3) = + COEFF*(1.0D0 + XI)*(1.0D0 + ETA)
c
3345      DSF(8,1) = - COEFF*(1.0D0 + ETA)*(1.0D0 + ZETA)
          DSF(8,2) = + COEFF*(1.0D0 - XI)*(1.0D0 + ZETA)
          DSF(8,3) = + COEFF*(1.0D0 - XI)*(1.0D0 + ETA)
c
c          Initialize the Jacobian matrix to zero.
c
3350      DO 10 II = 1, 3
          DO 10 JJ = 1, 3
          GJ(II,JJ) = 0.0D0
10 CONTINUE
c
3355      Compute the Jacobian matrix for the current Gauss point.
c
          DO 20 K = 1, 8
          GJ(1,1) = GJ(1,1) + DSF(K,1)*X(N,K)
          GJ(1,2) = GJ(1,2) + DSF(K,1)*Y(N,K)
3360      GJ(1,3) = GJ(1,3) + DSF(K,1)*Z(N,K)
          GJ(2,1) = GJ(2,1) + DSF(K,2)*X(N,K)
          GJ(2,2) = GJ(2,2) + DSF(K,2)*Y(N,K)
          GJ(2,3) = GJ(2,3) + DSF(K,2)*Z(N,K)
3365      GJ(3,1) = GJ(3,1) + DSF(K,3)*X(N,K)
          GJ(3,2) = GJ(3,2) + DSF(K,3)*Y(N,K)
          GJ(3,3) = GJ(3,3) + DSF(K,3)*Z(N,K)
20 CONTINUE
c
3370      Compute the determinant of the Jacobian matrix for the current
c          Gauss point.
c
          DET = GJ(1,1)*(GJ(2,2)*GJ(3,3) - GJ(2,3)*GJ(3,2))
          DET = DET - GJ(1,2)*(GJ(2,1)*GJ(3,3) - GJ(2,3)*GJ(3,1))
          DET = DET + GJ(1,3)*(GJ(2,1)*GJ(3,2) - GJ(2,2)*GJ(3,1))

```

```

3370      c
      c      Compute the cofactor matrix of the Jacobian matrix in prep-
      c      aration for computing the inverse of the Jacobian matrix.
      c
      CF(1,1) = GJ(2,2)*GJ(3,3) - GJ(2,3)*GJ(3,2)
3375      CF(1,2) = GJ(2,3)*GJ(3,1) - GJ(2,1)*GJ(3,3)
      CF(1,3) = GJ(2,1)*GJ(3,2) - GJ(2,2)*GJ(3,1)
      CF(2,1) = GJ(1,3)*GJ(3,2) - GJ(1,3)*GJ(3,3)
      CF(2,2) = GJ(1,1)*GJ(3,3) - GJ(1,3)*GJ(3,1)
      CF(2,3) = GJ(1,2)*GJ(3,1) - GJ(1,1)*GJ(3,2)
3380      CF(3,1) = GJ(1,2)*GJ(2,3) - GJ(1,3)*GJ(2,2)
      CF(3,2) = GJ(1,3)*GJ(2,1) - GJ(1,1)*GJ(2,3)
      CF(3,3) = GJ(1,1)*GJ(2,2) - GJ(1,2)*GJ(2,1)
      c
      c      Compute the inverse of the Jacobian matrix for the current
3385      c      Gauss point, by using the cofactor matrix and determinant.
      c
      GINV(1,1) = CF(1,1)/DET
      GINV(1,2) = CF(2,1)/DET
      GINV(1,3) = CF(3,1)/DET
3390      GINV(2,1) = CF(1,2)/DET
      GINV(2,2) = CF(2,2)/DET
      GINV(2,3) = CF(3,2)/DET
      GINV(3,1) = CF(1,3)/DET
      GINV(3,2) = CF(2,3)/DET
3395      GINV(3,3) = CF(3,3)/DET
      c
      c      Compute the coefficient matrix for the current element, N,
      c      corresponding to the current Gauss point, K.
      c
3400      DO 40 J = 1, 8
      DO 30 I = 1, 8
      c
      SX(I,J) = GINV(1,1)*(SF(I)*DSF(J,1)+SF(J)*DSF(I,1))
3405      & + GINV(1,2)*(SF(I)*DSF(J,2)+SF(J)*DSF(I,2))
      & + GINV(1,3)*(SF(I)*DSF(J,3)+SF(J)*DSF(I,3))
      SX(I,J) = SX(I,J)*DET
      c
      SY(I,J) = GINV(2,1)*(SF(I)*DSF(J,1)+SF(J)*DSF(I,1))
3410      & + GINV(2,2)*(SF(I)*DSF(J,2)+SF(J)*DSF(I,2))
      & + GINV(2,3)*(SF(I)*DSF(J,3)+SF(J)*DSF(I,3))
      SY(I,J) = SY(I,J)*DET
      c
      SZ(I,J) = GINV(3,1)*(SF(I)*DSF(J,1)+SF(J)*DSF(I,1))
3415      & + GINV(3,2)*(SF(I)*DSF(J,2)+SF(J)*DSF(I,2))
      & + GINV(3,3)*(SF(I)*DSF(J,3)+SF(J)*DSF(I,3))
      SZ(I,J) = SZ(I,J)*DET
      c
      SXX(I,J) = GINV(1,1)*DSF(I,1)
3420      & + GINV(1,2)*DSF(I,2)
      & + GINV(1,3)*DSF(I,3)
      SXX(I,J) = SXX(I,J)*(GINV(1,1)*DSF(J,1)
      & + GINV(1,2)*DSF(J,2)
      & + GINV(1,3)*DSF(J,3))

```



```

c
3480  MEQ = NEQ - 1
      DO 30 NPIV = 1, MEQ
      NPIVOT = NPIV + 1
      LSTSUB = NPIV + NHBW - 1
      IF(LSTSUB.GT.NEQ) LSTSUB = NEQ
      DO 20 NROW = NPIVOT, LSTSUB
3485  c
      c      Invert rows and columns for row factor.
      c
      NCOL = NROW - NPIV + 1
      FACTOR = GK(NPIV,NCOL)/GK(NPIV,1)
3490  DO 10 NCOL = NROW, LSTSUB
      ICOL = NCOL - NROW + 1
      JCOL = NCOL - NPIV + 1
      GK(NROW,ICOL) = GK(NROW,ICOL) - FACTOR*GK(NPIV,JCOL)
3495  10 CONTINUE
      GF(NROW) = GF(NROW) - FACTOR*GF(NPIV)
      20 CONTINUE
      30 CONTINUE
c
c      Back-substitution.
c
3500  DO 70 IJK = 2, NEQ
      NPIV = NEQ - IJK + 2
      GF(NPIV) = GF(NPIV)/GK(NPIV,1)
      LSTSUB = NPIV - NHBW + 1
3505  IF(LSTSUB.LT.1) LSTSUB = 1
      NPIVOT = NPIV - 1
      DO 60 JKI = LSTSUB, NPIVOT
      NROW = NPIVOT - JKI + LSTSUB
      NCOL = NPIV - NROW + 1
3510  FACTOR = GK(NROW,NCOL)
      GF(NROW) = GF(NROW) - FACTOR*GF(NPIV)
      60 CONTINUE
      70 CONTINUE
      GF(1) = GF(1)/GK(1,1)
3515  c
      c      Return to the calling routine.
      c
      c
      RETURN
      END

```

## **Vita**

Andrew D. Wright was born in Charleston, West Virginia, on October 6, 1972. Spending the majority of his formative years there, he graduated from George Washington High School in 1990.

Upon graduation he entered Virginia Polytechnic Institute and State University to pursue an undergraduate degree in Mechanical Engineering. Deciding to complete his studies elsewhere, he earned his Bachelor of Science from the University of Tennessee, Knoxville, graduating magna cum laude in the May of 1994. Returning to Virginia Tech for his graduate studies, he earned his Master of Science in February of 1996.