



Design Optimization of Hypersonic Test Facility Nozzle Contours Using Splined Corrections

Frederick L. Shope
Aerospace Testing Alliance

March 2005

Final Report for Period 1 October 2001 — 30 September 2004

STATEMENT A: Approved for public release;
distribution unlimited.

**ARNOLD ENGINEERING DEVELOPMENT CENTER
ARNOLD AIR FORCE BASE, TENNESSEE
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE**

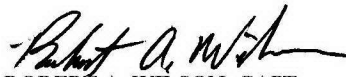
NOTICES
When U. S. Government drawings, specifications, or other data are used for any purpose other than a definitely related Government procurement operation, the Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.
Qualified users may obtain copies of this report from the Defense Technical Information Center.
References to named commercial products in this report are not to be considered in any sense as an endorsement of the product by the United States Air Force or the Government.

DESTRUCTION NOTICE

For unclassified, limited documents, destroy by any method that will prevent disclosure or reconstruction of the document.

APPROVAL STATEMENT

This report has been reviewed and approved.



ROBERT A. WILSON, CAPT
Applied Technology Division
Test Operations Directorate

Approved for publication:

FOR THE COMMANDER



ROBERT T. CROOK
Deputy Chief, Applied Technology Division
Test Operations Directorate

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.					
1. REPORT DATE (DD-MM-YYYY) 00-03-2005		2. REPORT TYPE Final Report		3. DATES COVERED (From - To) 1 October 2001 - 30 September 2004	
4. TITLE AND SUBTITLE Design Optimization of Hypersonic Test Facility Nozzle Contours Using Splined Corrections				5a. CONTRACT NUMBER F40600-03-C-0001	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Frederick L. Shope Aerospace Testing Alliance				5d. PROJECT NUMBER 9829	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) AEDC/DOT 1099 Avenue C Arnold AFB, TN 37389-9013				8. PERFORMING ORGANIZATION REPORT NUMBER AEDC-TR-04-2	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Office of Scientific Research 4015 Wilson Blvd, Room 713 Arlington, VA 22203-1954				10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/AFOSR	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT STATEMENT A: Approved for public release; distribution unlimited.					
13. SUPPLEMENTARY NOTES Available in Defense Technical Information Center (DTIC).					
14. ABSTRACT A procedure is presented to design and optimize the contour of a hypersonic wind tunnel nozzle with a goal of minimizing exit flow nonuniformity. The procedure uses a Navier-Stokes solver that admits chemical and vibrational nonequilibrium thermodynamics and high-pressure effects. The two-step optimization process is accomplished with a basic least-squares optimization (LSO) method. The first step of the design procedure begins with an existing inviscid irrotational method of characteristics (MOC) that is limited to thermally and calorically perfect gas (TCPG). MOC is used to design an inviscid contour, which is then corrected with a boundary-layer displacement thickness from an integral momentum formulation. The deleterious effects of the TCPG assumption are ameliorated by using an effective specific-heat ratio and an effective gas constant chosen so that the TCPG computation yields the same exit Mach number and velocity as a quasi-one-dimensional computation based on thermochemical equilibrium. The MOC-based contour is then formally optimized using the LSO method, treating various MOC program input variables as formal design parameters. The objective function is the square deviation of flow properties from target values at the nozzle exit, excluding the boundary layer, and is computed with the Navier-Stokes solver. The flow properties chosen for the objective function are the velocity components and the static pressure and density. After the MOC contour is optimized, the second step of the optimization procedure commences. In the second step, the contour is further perturbed by adding a small correction distribution represented as a cubic spline fitted to a limited number of nodes along the contour. The correction values of the nozzle radius are the formal design parameters for the next application of LSO. The splined correction procedure effectively divorces the number of nodes representing the contour from the number of design parameters. This affords freedom to limit the number of design parameters while maintaining enough contour nodes to accurately represent the contour. Optimization of the corrections allows consideration of physics not included in the MOC optimization. During development of the present design procedure, several computational tools were developed. These tools include a Navier-Stokes solver adapted by its developer, Prof. Graham Candler (U. MN), and validated for problems of interest at AEDC; the LSO code; the cubic spline correction code; and a code to trace characteristics from selected nozzle exit points to their origins upstream in the nozzle. Unfortunately, these tools do not include a single computer program or user interface that accomplishes the entire design optimization procedure without user intervention. This is because the present LSO-Newton iteration is not reliably convergent (effectively, a manually controlled direct search is performed, and this approach was found successful). However, the most labor-intensive portion of the design procedure has been coded in a program based on the Tcl scripting language. This script executes the various software components for a single LSO iteration, excluding the MOC optimization step. The design procedure is demonstrated by design of a nozzle for a Mach number 6 combustion-heated test facility.					
15. SUBJECT TERMS hypersonic nozzle contour design optimization					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT		18. NUMBER OF PAGES
a. REPORT	b. ABSTRACT	c. THIS PAGE	Unclassified		131
Unclassified	Unclassified	Unclassified			
			19a. NAMES OF RESPONSIBLE PERSON Capt. Robert A. Wilson		
			19b. TELEPHONE NUMBER (Include area code) (931) 454-5261		

PREFACE

The work reported herein was conducted by the Arnold Engineering Development Center (AEDC), Air Force Materiel Command (AFMC), at the request of the Air Force Office of Scientific Research (AFOSR). The results of the technology development effort were obtained by Aerospace Testing Alliance (ATA), the operations, maintenance, information management, and support contractor for AEDC, AFMC, Arnold Air Force Base, Tennessee, under Project Number 9829. The AFOSR Project Manager was Dr. John Schmisser, and the AEDC Air Force Project Manager was Captain Robert Wilson, AEDC/DOT. The development effort was completed on September 30, 2004, and the manuscript was submitted for publication on December 31, 2004.

The author wishes to gratefully acknowledge AFOSR and Dr. Schmisser for support of this project. The very important development work on the flow solver (reported elsewhere) by Prof. Graham Candler, University of Minnesota, is also acknowledged. The author would like to thank Mr. Kenneth E. Tatum and Mr. William E. Milam for accomplishing the Peer Review of the manuscript. Finally, Dr. Gregory A. Movik is acknowledged for conceiving of the project and laying the initial groundwork for its undertaking. The author greatly appreciates the contributions of these individuals.

CONTENTS

	<u>Page</u>
PREFACE	1
1.0 INTRODUCTION	7
1.1 REQUIREMENTS	7
1.2 GENERAL APPROACH	8
2.0 PREVIOUS WORK	8
2.1 DIRECT-DESIGN TECHNIQUES	9
2.2 DESIGN-BY-ANALYSIS TECHNIQUES	11
3.0 APPROACH	14
3.1 SOFTWARE	16
3.2 DESIGN PROCEDURE	20
4.0 RESULTS	26
4.1 CONTOUR INITIALIZATION	27
4.2 MOC CONTOUR OPTIMIZATION	29
4.3 OPTIMIZATION OF SPLINED CORRECTIONS	31
4.4 CRITIQUE OF METHOD AND RESULTS	33
5.0 SUMMARY AND CONCLUSIONS	34
REFERENCES	34

ILLUSTRATIONS

Figure

1. Notation and Design Procedure Used by Sivells	38
2. Sivells' Option Chosen for Present Contour Design Problem	39
3. Left-Running Characteristics for an AEDC Arc Heater Nozzle	40
4. Design Procedure Based on Least-Squares Optimization	41
5. Flow Chart of Nozzle Contour Design Procedure	42
6. Characteristic Net and Characteristics Traced Upstream from Nozzle Exit	44
7. Traced Characteristics in the Inflection Region of Nozzle	45
8. Traced Characteristics in Throat Region	46
9. Pressure Distribution for Three Gas Models	47
10. Temperature Distribution for Three Gas Models	48
11. Density Distribution for Three Gas Models	49
12. Velocity Distribution for Three Gas Models	50
13. Difference Between Cubic Spline Representation and Original MOC Contour	51
14. Nozzle Contour with Exaggerated Correction Distribution	50
15. Candidate Viscous MOC Contours for APTU Mach 6	53
16. DPLR Exit Axial Velocity Profiles Compared to Equilibrium Values for APTU Mach 6 Candidate Contours	54
17. DPLR Exit Axis-Normal Velocity Profiles for APTU Mach 6 Candidate Contours	55
18. DPLR Exit Flow Angle Profiles for APTU Mach 6 Candidate Contours	56
19. DPLR Exit Static Pressure Profiles Compared to Equilibrium Values for APTU Mach 6 Candidate Contours	57

<u>Figure</u>	<u>Page</u>
20. DPLR Exit Static Density Profiles Compared to Equilibrium Values for APTU Mach 6 Candidate Contours	58
21. DPLR Exit Static Temperature Profiles Compared to Equilibrium Values for APTU Mach 6 Candidate Contours	59
22. DPLR Exit Static Vibrational Temperature Profiles Compared to Equilibrium Values for APTU Mach 6 Candidate Contours	60
23. Centerline Streamwise Velocity for APTU Mach 6 Candidate Contours	61
24. Centerline Static Pressure for APTU Mach 6 Candidate Contours	62
25. Centerline Static Density for APTU Mach 6 Candidate Contours	63
26. Centerline Static Translational/Rotational Temperature for APTU Mach 6 Candidate Contours	64
27. Centerline Static Vibrational Temperature for APTU Mach 6 Candidate Contours	65
28. Centerline Species Mass Fraction Distribution for Design1	66
29. Centerline Species Mass Fraction Distribution for Design2	67
30. Centerline Species Mass Fraction Distribution for Design3	68
31. Centerline Species Mass Fraction Distribution for Design4	69
32. Deviation of Exit Axial Velocity from Equilibrium as a Percent of CEA96 Equilibrium Velocity, Designs1-4	70
33. Exit Axis-Normal Velocity as a Percent of CEA96 Equilibrium Velocity, Designs1-4	71
34. Deviation of Exit Static Pressure from Equilibrium as a Percent of CEA96 Equilibrium Velocity, Designs1-4	72
35. Deviation of Exit Static Density from Equilibrium as a Percent of CEA96 Equilibrium Velocity, Designs1-4	73
36. Deviation of Exit Static Temperature from Equilibrium as a Percent of CEA96 Equilibrium Velocity, Designs1-4	74
37. Deviation of Exit Vibrational Temperature from Equilibrium as a Percent of CEA96 Equilibrium Velocity, Designs1-4	75
38. Objective Function vs Design Parameter RC	76
39. Axial Velocity Component of Objective Function for Three Iterations of MOC Contour Optimization	77
40. Axis-Normal Velocity Component of Objective Function for Three Iterations of MOC Contour Optimization	78
41. Static Pressure Component of Objective Function for Three Iterations of MOC Contour Optimization	79
42. Static Density Component of Objective Function for Three Iterations of MOC Contour Optimization	80
43. Comparison of All Four Components of Objective Function for Three Iterations of MOC Contour Optimization	81
44. Objective Function History for LSO	82
45. Axial Velocity Component of Objective Function for LSO	83
46. Axis-Normal Velocity Component of Objective Function for LSO	84
47. Static Pressure Component of Objective Function for LSO	85
48. Static Density Component of Objective Function for LSO	86

APPENDIXES

<u>Appendix</u>	<u>Page</u>
A. AUTOMATED NOZZLE DESIGN OPTIMIZATION (ANDO.t and ANDO2.t)	89
B. LEAST-SQUARES OPTIMIZATION PROGRAM (lso.f).	100
C. SPLINED CORRECTION PROGRAM (splinenc.f)	103
D. NODE PERTURBATION PROGRAM (nodepert.f)	105
E. OBJECTIVE FUNCTION EVALUATOR PROGRAM (obj.f)	108
F. DPLR CODE PACKAGE	109
G. CHARACTERISTIC TRACING PROGRAM (RCFROMLC.FOR)	113
H. PROGRAM FOR EFFECTIVE CALORICALLY PERFECT GAS MODEL (EFFCPG.FOR)	116
I. NODE LOCATOR PROGRAM (NODELOCS.FOR)	118
J. SINGLE-NODE PERTURBATION PROGRAM (NOZCOR.FOR)	119
NOMENCLATURE.	122

1.0 INTRODUCTION

Envisioned future hypersonic flight vehicles require a close-coupled integration of all vehicle subsystems to achieve the high performance level needed to make such vehicles practical. Since the needed performance levels clearly approach practical limits of anticipated technology, the requirements for accuracy and uncertainty of design information become severe. The source of much of the design information is the nation's complement of hypersonic wind tunnels, which accordingly acquire increasingly stringent flow quality requirements. An important factor affecting flow quality is the design of the nozzle contour. For nearly three generations, the standard design procedure for hypersonic nozzle contours has used an ingenious application of the inviscid method of characteristics originally proposed by Ludwig Prandtl and Adolf Busemann in 1929. While clearly successful, the Prandtl-Busemann method becomes inaccurate as the boundary-layer thickness becomes large—as it does in high Mach number tunnels. Although many important variations of the original idea have been proposed and successfully implemented, there were no fundamental advances beyond the original idea until John Korte of NASA Langley published a Navier-Stokes-based design-by-analysis technique. Korte's approach of coupling a viscous flow solver with an optimization technique offered the first significant quantitative improvement over the classical design by the method of characteristics. The method of the present report is a design-by-analysis approach based on a *reacting* Navier-Stokes solver.

This report documents the development and demonstration of a procedure to design the contour of a hypersonic wind tunnel nozzle in which the flow may be undergoing thermochemical nonequilibrium processes. The remainder of the introduction defines the requirements for the test facility nozzles and outlines the general approach to accomplishing the design. Next, the results of a literature review of previous nozzle design methods are presented. Then the present design procedure is presented in detail, and a demonstration design is discussed. Finally, conclusions drawn during the course of the effort are summarized. Appendixes A through I give detailed information to assist in using the software developed here.

1.1 REQUIREMENTS

The requirement of this project is to accomplish the research necessary to enable the design of hypersonic wind tunnel nozzle contours containing the flow of viscous gases in thermochemical nonequilibrium and at high pressure and temperature. Required nozzle geometries are axisymmetric and frequently have severe length constraints. Nozzles are required to deliver uniform and parallel hypersonic flow at a specified speed, pressure, and temperature. Test media of interest include air, nitrogen, and products of hydrocarbon combustion. Arnold Engineering Development Center (AEDC) test facilities that require this capability include the Aerodynamic and Propulsion Test Unit (APTU); the Hypervelocity Wind Tunnel 9 Facility (Tunnel 9); and the High Enthalpy Ablation Test (HEAT) units H1, H2, and H3. Flow nonuniformity limits have been established as ± 0.25 percent for aerodynamic testing (all flow properties) and ± 2 percent for aeropropulsion testing. Since the nozzle contour may not be able to

eliminate flow nonuniformity existing upstream of the throat, these requirements are taken to mean that the nozzle should not add more than this level of nonuniformity if the nozzle is provided with perfectly uniform inlet flow.

1.2 GENERAL APPROACH

The approach taken here is to seek values of geometric parameters defining the nozzle contour that minimize exit flow nonuniformity. The minimization process uses least-squares optimization (LSO). Flow nonuniformity is measured by computing the deviation of selected flow variables from mean values or target values at grid points along a radial line at the nozzle exit, excluding the boundary-layer region. The exit flow values are obtained from a Navier-Stokes solution that admits thermochemical nonequilibrium effects. The design procedure has two steps. First, a contour is designed using a classical method of characteristics (MOC) with a boundary-layer correction. This MOC contour is optimized using LSO, where the design parameters are selected input variables to the MOC program. Second, a small perturbation defined by cubic splines is applied to the MOC contour. The perturbation values at the nodes of the splines are the design parameters for a second LSO iteration. The second step of the design procedure allows investigators to optimize the contour by considering additional physics modeled in the Navier-Stokes solver but not in the MOC code, such as more complex thermodynamics.

The MOC code is that developed by Sivells (code name CONTUR, Refs. 1 and 2). The Navier-Stokes solver is the Data Parallel Line Relaxation (DPLR) method of Wright and Candler (Ref. 3). Additional software developed during the present effort includes a least-squares optimization code (LOC), a characteristic tracing code, a splining code, and two scripts to automate part of the nozzle design optimization procedure.

The resulting design procedure is not a numerically well-behaved, fully automated process that will design a contour without significant user involvement. Experience here and elsewhere has shown that the LSO must use fairly small relaxation factors (0.1 or less) and thus serves only to define a search direction. In effect, this procedure results in a direct search process where the user must examine the values of the objective function and predicted parameter corrections after each LSO step and decide how much of the correction to accept. Despite that level of difficulty, however, it should be understood that the initial MOC contour will produce good uniformity (i.e., within several percent) and that the LSO will yield further significant improvements in both stages of the design procedure. The designer must judge whether the additional labor required to accomplish a Navier-Stokes-based LSO is justified by the improvement in flow quality.

2.0 PREVIOUS WORK

The present review concentrates on the aerodynamic design of ground test facility nozzles and generally ignores the design of propulsion nozzles. Facility nozzles are intended to yield uniform and parallel flow, while propulsion nozzles usually seek to maximize thrust-to-weight

ratio. Also not specifically considered here are design methods for airfoils or other air vehicle components, or the vast area of multidisciplinary design. However, the general methods, such as the sensitivity equation method and the adjoint method, are reviewed briefly. The more modern design techniques, whether for nozzles or otherwise, couple an optimization technique with a flow solver. While such design techniques from other technical problems might be applicable to facility nozzles, the primary intent here is to briefly review methods that have been used to design facility nozzles.

2.1 DIRECT-DESIGN TECHNIQUES

The literature on direct design of nozzle contours, primarily that on design utilizing MOC-based methods, is extensive. Accordingly, the intent here is not to review all publications but to summarize direct-design methods and some of the most important variations. Many more publications were examined than are mentioned here. The references given are representative but not comprehensive.

Direct-design MOC-based methods have perhaps culminated in the method of Sivells (Ref. 1). Figure 1 illustrates how Sivells applied MOC to the contour design problem. His code contains many design options, but the most complex is covered by this figure. The most prominent difference from earlier approaches is the presence of the radial flow region. For a thermally and calorically perfect gas (TCPG), the radial flow region is exactly described by an analytical solution for compressible source flow. A radial flow region was originally included in the design code for computational efficiency (a characteristics solution is not needed to compute the radial flow region), but it is also important in expanding the designer's control over the contour. If a radial flow region is to be present, then two centerline Mach number (or velocity) distributions must also be specified as boundary conditions for the MOC process. One centerline distribution extends from the throat characteristic to the upstream end of the radial flow region, and the other distribution is located from the downstream end of the radial flow region to the start of the test rhombus. However, the radial flow region is not generally used for high-temperature nozzle designs because it tends to produce very long nozzles. Instead, a single, centerline Mach number distribution is specified between the throat characteristic and the test rhombus as described by Fig. 2. The necessity of the radial flow region is a point of contention among designers, and additional discussion is given in Section 3.2.1 below.

As illustrated in Fig. 2, Sivells' design method for a single, centerline distribution proceeds as follows. The exit Mach number, exit radius, and throat radius ratio (ratio of wall radius of curvature to throat radius) are specified along with various gas properties. The throat characteristic, TI, is constructed from an analytical series solution for transonic flow in a circular-arc throat. The centerline Mach number or velocity distributions can be either supplied by the user in tabular form or computed internally by choosing polynomials of degree 3 to 5. For internally generated distributions, Sivells provides matching of various-order derivatives of these polynomials to the test rhombus and transonic solution, which results in a high degree of

continuity. With these boundaries established, the characteristic net in region TICD (Fig. 2) can be constructed, marching upstream. The location of the contour is computed by integrating mass flux along the characteristics, finishing the inviscid design. Reference 1 gives the details of the various mathematical models. A typical set of left-running characteristics is shown in Fig. 3 for an arc nozzle of Mach number 3.5.

It is worth noting that the design model assembled by Sivells is very intricate. The combination of inviscid, irrotational, and calorically perfect flow through a nozzle with a circular arc throat and radial flow region constitutes a model in which the various components are closely interdependent. The consequence is that if, for example, an equilibrium thermodynamic equation of state is deemed essential, then the analytical transonic and radial flow solutions are immediately lost. Thus, some effort has gone into finding ways to apply Sivells' code beyond its immediately obvious range of applicability.

Because of the importance of the thermodynamic model for hypersonic applications, several MOC-based design methods using more advanced thermodynamics have been reported (Refs. 4 through 7). All of these assume equilibrium thermodynamics and employ either tabular or analytical equations of state. The MOC becomes more complex when TCPG is abandoned (for 2D TCPG flow, the Riemann invariants are simply $\theta \pm v = \text{constant}$ along characteristics, where θ is the flow angle and v is the Prandtl-Meyer angle). Reference 6 gives more detail on real-gas MOC.

Several additional variations of the MOC-based nozzle design technique should be mentioned. One is the sharp-throat nozzle (Refs. 8 through 10), which produces a minimum length nozzle by effectively eliminating the entire region between the throat and the nozzle inflection point. This is accomplished with a sharp turn in which the flow is imagined to undergo an isentropic, Prandtl-Meyer-type expansion from the sonic throat directly to the nozzle inflection angle. For wind tunnel applications, the sharp throat nozzle will minimize nozzle cost by minimizing the length, amount of material, and machining time. However, there remains concern about flow separation and survival of the corner in a high-temperature flow.

A second important MOC variation is the design of nozzles of arbitrary cross section (Refs. 11 and 12). This variation is based on the simple idea that any arbitrary cross section nozzle can be designed as a subset of a larger axisymmetric nozzle. The desired exit shape is traced out as a closed curve on the axis-normal exit plane of the larger axisymmetric nozzle. Points along this curve define streamlines that can be traced upstream to the throat through the characteristic net, thus defining the contour. Note the remarkable fact that the noncircular nozzle contains an axisymmetric flow field. This is the approach taken to designing the four-sided contoured nozzle used in the NASA Langley quiet tunnel. It must be noted that Haddad and Moss (Ref. 12) appear to be unaware of the much earlier work (1952) of Beckwith et al., who, in turn, credit a Morton Cooper study (apparently unpublished) with the first application of the method in 1947.

A third important MOC variation is perhaps the first formal multidisciplinary facility nozzle design technique, which was developed by Varner (Ref. 13). In this application, Varner was designing a two-dimensional, jack-driven, flexible plate nozzle. He coupled an MOC-based contour design procedure with the structural equations for a deflecting beam, in the process optimizing the plate thickness distribution and the jack locations to minimize flow nonuniformity and the number of jacks required. The procedure used a formal optimization technique.

One unique design approach is that attributed to Brodsky (Ref. 14). Rather than an MOC approach, Brodsky developed a finite-difference formulation, which he marched outward from the nozzle centerline, on which he specified a Mach number distribution. As he marched outward, the solution front looked much like left- and right-running characteristics. An attempt was made to apply the method in the subsonic region of the flow; and, though numerical oscillations were encountered, the method was not without some success. Reference 14 does not include results, but good agreement with a previous conventional MOC design is claimed.

2.2 DESIGN-BY-ANALYSIS TECHNIQUES

Documented design-by-analysis (DBA) methods for nozzle design date back little more than 15 years. The first such method known to the present author is that reported by Peter Hoffman in November 1989, a work not published outside of AEDC. Hoffman used a linear combination of contours and solved for the coefficients with a Newton method. His flow solver was a space-marching Euler method. As a demonstration, he designed a Mach 4 nozzle starting from an initial contour based on a cubic polynomial. His method designed a contour that yielded Mach number uniformity of less than one percent in three iterations.

Shortly thereafter, John Korte of NASA Langley reported a Navier-Stokes DBA nozzle design technique (also not an open literature publication). The design parameters for this early method were the actual nozzle coordinates. A much more advanced approach was reported by Korte et al. in Ref. 15. For this technique, the objective function included not only the test rhombus Mach number but also the centerline Mach number distribution, which distribution target values were determined by Sivells' method of polynomials and a radial flow region. The design parameters were the slopes rather than the coordinates of the contour. Use of the slopes was said to greatly improve the convergence. The flow solver was a parabolized Navier-Stokes (PNS) code started from a time-dependent Navier-Stokes (NS) solution for a fixed throat contour. The flow solver assumed TCPG thermodynamics. The Korte procedure is named CFD-Based Aerodynamic Nozzle Design and Optimization (CAN-DO) and has been used successfully in several nozzle design and redesign efforts. This procedure is the first major advance in nozzle contour design technology since the MOC, which is attributed to Prandtl and Busemann in 1929.

Another DBA approach was that reported by Keeling in Ref. 16, a paper awarded the AIAA Best Paper in Thermophysics for 1993. Keeling followed the approach of Barger and Moitra (Ref. 17) in defining the contour as a linear combination of trial contours but added the all-important

constraint that the combination be convex (coefficients must be positive). He also specified that the coefficients must sum to one. These decisions led to two breakthrough conclusions: 1) the numerical method did not require computation of derivatives (i.e., computation of the very expensive Jacobian matrix was not needed), and 2) convergence of the iteration could be guaranteed. As a consequence of this breakthrough, Keeling was able to accomplish his demonstration design without supercomputer-level resources; computations were performed on a single serial workstation. Keeling used the reacting PNS solver attributed to Molvik and Merkle (Ref. 18). A criticism of the method is that the final solution must be bounded within the initial contours and is limited accordingly.

A more recent and simplified DBA variation is that reported by Tolle (Ref. 19). Tolle applied Response Surface Methodology (RSM) to the optimization problem. In RSM, one first computes the objective function for several sets of values of the design parameters, with each objective function evaluation requiring a Navier-Stokes solution. The objective function (the response surface) is then modeled as a least-squares curve fit versus the design parameters. In this fit, however, the objective function values from the PNS solutions are held fixed, and the values of the design parameters are altered to minimize the least-square error. The fit is a multivariable linear regression (i.e., a plane in n -space). The direction of steepest descent identifies the search direction to select the next iteration. Tolle represents the nozzle contour as a circular arc from the throat to the inflection point and as a rotated parabola from the inflection point to the exit. His method has only two design parameters: the “attachment angle,” where the circle and parabola meet, and the wall slope at the exit. His objective function is the deviation from uniformity of the Mach number at the nozzle exit plane and along the *entire* nozzle centerline. He uses weighting functions to deemphasize the importance of the Mach number error in the boundary layer and in the upstream, rapidly expanding region of the centerline. He applies his method to redesign an existing nozzle that was originally designed with an MOC and boundary-layer approach. The computed flow quality of the redesigned contour is not obviously better than the original design, probably because of the limited flexibility of the assumed contour functions and the attempt to drive the entire centerline to a constant Mach number.

An alternative to the conventional finite-difference-based DBA methods is those referred to as sensitivity equation methods (Refs. 20 through 23). Sensitivity equation methods are based on a simple and powerful observation that is developed as follows. The goal is to find a nozzle contour, e.g., $w(x,y,z,\alpha) = 0$, that minimizes some measure of flow nonuniformity by varying the design parameters α , where α is a vector that might represent, for example, coefficients in a polynomial representation of the contour. Given an initial guess at w , one wishes to model the flow in the nozzle with the Navier-Stokes equations,

$$\frac{\partial q}{\partial t} + \frac{\partial E}{\partial x} + \frac{\partial F}{\partial y} + \frac{\partial G}{\partial z} = 0 \quad (1)$$

This is the vector form of the partial-differential equations (PDE). An error function is formed, e.g., $\varepsilon(q)$, that might represent the deviation of the flow for the current contour estimate

from perfectly uniform flow over a surface at the nozzle exit. For example, we might choose $\varepsilon(q) = \sum (q - q^*)^2$ where the sum is over grid points on the entrance to the test rhombus and q^* is the desired flow profile, presumably uniform. To minimize the nonuniformity, one therefore seeks values of α which yield

$$\frac{\partial \varepsilon}{\partial \alpha} = 0 \quad (2)$$

By the chain rule,

$$\frac{\partial \varepsilon}{\partial \alpha} = \frac{\partial \varepsilon}{\partial q} \frac{\partial q}{\partial \alpha} \quad (3)$$

The key to proceeding with the design is to find the matrix $q' = \partial q / \partial \alpha$. For the sensitivity method one makes the observation that the operation

$$\frac{\partial}{\partial \alpha} \left(\frac{\partial q}{\partial t} + \frac{\partial E}{\partial x} + \frac{\partial F}{\partial y} + \frac{\partial G}{\partial z} \right) = 0 \quad (4)$$

yields the result that

$$\frac{\partial q'}{\partial t} + \frac{\partial E'}{\partial x} + \frac{\partial F'}{\partial y} + \frac{\partial G'}{\partial z} = 0 \quad (5)$$

where prime denotes differentiation with respect to α . The important observation is that this equation is identical in form to the original Navier-Stokes equations. This has several remarkably favorable consequences.

- First, if one has a code that solves Navier-Stokes, then one also has a code—with some modifications—that solves for the sensitivities q' .
- Second, in deriving the formulas for q' , E' , ..., one finds that these derivatives do not depend explicitly on α and that the resulting PDEs are linear.
- Third, since the nozzle geometry is fixed in time, the time derivative may be dropped from the sensitivity PDE.

The total effect of these observations is that the usual Jacobian matrix for the Newton iteration q' can be computed with one flow-solver evaluation and a steady, noniterative solution of the sensitivity PDEs (one linear solution for each design parameter). This is a potentially large reduction of computation time compared to a conventional Newton method, where each column of the Jacobian matrix requires a full time-dependent flow-solver solution. Note that the sensitivity equations could be solved with a separate code or simultaneously with Navier-Stokes in the existing code. In this latter alternative, the arrays for the solution variable would be extended for storage of the additional sensitivity variables. When the Navier-Stokes solution converged in time, the sensitivity variables and the contour design would also be converged and complete.

There is also an interesting extension of the sensitivity method based on Lagrange multipliers. Here, the error function is minimized with respect to the design parameters subject to a side condition. The "side condition" is, in fact, the Navier-Stokes equations. In other words, one solves the equation system $\nabla_{\alpha}\varepsilon + \lambda\nabla_{\alpha}g = 0$, where g is the system of Navier-Stokes equations and λ is the Lagrange multiplier vector. To visualize this in purely geometric terms, consider that $\nabla_{\alpha}\varepsilon$ is a vector normal to the surface $\varepsilon(\alpha)$, and $\nabla_{\alpha}g$ is a vector normal to $g(\alpha)$. Surfaces $\varepsilon(\alpha)$ and $g(\alpha)$ (presumably) intersect, forming a curve along which we want to find a minimum point on ε . We will accomplish this if the two vectors are coplanar (emanate from the same point). They will be coplanar if there exist constants λ , not all zero, such that $\nabla_{\alpha}\varepsilon + \lambda\nabla_{\alpha}g = 0$ (i.e., the vectors are not linearly independent). Note that this is equivalent to finding a point on the surface $\varepsilon(\alpha)$ where the directional derivative along the intersection curve is zero. This is referred to as the adjoint method.

Sensitivity-based methods appear to have significant advantages over conventional Newton methods, but there are also restrictive issues to consider.

1. A new code to generate the sensitivity matrix must be created. This could be a limited modification of the original code. The modifications include different definitions of the solution vector and different boundary conditions.
2. Although the code modification is often touted as "easy," it would seem to require an intimate familiarity with the original code. Since modern flow solvers can extend to 100,000 lines of code, this may not be a trivial requirement.
3. With the conventional Newton method, changing to another flow solver, for example, to one with more complete physics, is relatively simple, whereas the sensitivity method will require a new sensitivity code to be developed to accompany each new flow solver. The required depth of code familiarity for implementing a conventional Newton method would seem to be significantly less than that required for the sensitivity method.
4. The conventional Newton method has been validated for hypersonic nozzle contour design (Refs. 24 and 25), whereas sensitivity-based methods have been applied only to relatively simple nozzle problems.

3.0 APPROACH

The present approach to nozzle contour design is DBA. An optimization technique is loosely coupled to a reacting Navier-Stokes solver. The sensitivity derivatives in the Jacobian matrix are computed with finite differences, with a flow-solver solution computed for each design parameter. The LSO procedure is illustrated in Fig. 4. An initial contour that is a function of a vector of design parameters α is assumed to exist. The design parameters are each, one at a time, perturbed slightly, and a contour $y(x, \alpha)$ is generated for each perturbation $\delta\alpha$. For each contour, a Navier-Stokes solution is computed, and each solution generates a set of nozzle exit flow profiles. Next, the LSO is performed (mathematical details are given in Section 3.1.2, below) and proceeds

as follows. A difference vector ε between the actual flow profile and the target value is computed for the initial unperturbed contour. The same computation is repeated for each of the perturbed contours. The Jacobian matrix J is then computed from finite-difference derivatives as $\delta\varepsilon/\delta\alpha$. The least-squares linear system [Eq. (11), below] is then solved for the corrections to the α -vector (i.e., $\Delta\alpha$) that will reduce the value of the objective function (i.e., that will reduce the flow nonuniformity). In practice, only a fraction, f , of the computed correction is likely to be used. In any event, a new set of design parameter values, α , is chosen, which then defines a new initial contour. A flow-solver solution is computed for this new contour, and the value of the objective function [Eq. (6), below] is computed. The designer then decides whether to proceed with another set of perturbations.

In the present design procedure, the process described above is applied in two separate steps, which will be referred to as “Step 1” and “Step 2” as defined in Fig. 5. First, an initial nozzle contour is designed using an MOC (Sivells' method herein, Ref. 1). For hypersonic nozzles it is recommended that deviations from TCPG be included in this initial contour. This could be accomplished by using a code that actually admits more complex thermodynamics, such as Ref. 6, or by computing an effective TCPG model that better mimics the actual thermodynamics. A procedure to compute an effective TCPG model is given in Section 3.2.1, below. An initial contour is then computed with Sivells' code. There are several possible design options that may be chosen with Sivells' code. The present recommendation is to use the option for a single centerline Mach number distribution rather than two separate distributions separated by a radial flow region. Additional detail is given in section 3.2.1, below. When an acceptable initial MOC contour has been designed, the contour should then be further optimized (Step 1) following the procedure in Fig. 4. The design parameters are selected input variables to Sivells' code; each input variable/design parameter is perturbed one at a time, and a flow-solver solution is computed for each perturbation. The LSO procedure proceeds as above. The end result is an MOC contour optimized for the full thermodynamic model included in the Data Parallel Line Relaxation (DPLR) code. However, the contour so obtained remains a purely MOC-based design. This completes the first of two steps in the design procedure.

Step 2 of the design procedure (Fig. 5) is to further optimize the MOC contour but to go beyond the limitations of MOC. The essence of the second step (and, in fact, a primary contribution of the present work) is to represent the contour as a table of coordinates plus a splined correction. That is, the spline is passed through a small correction of the contour defined at only a few points along the contour, but the spline is not used to represent the contour itself. (This approach is justified and described in more detail in Section 3.2.2, below.) The corrections values at those few nodes along the contour are the design parameters for Step 2 of the design procedure. The LSO process is repeated with this set of design parameters to obtain a further improvement in flow quality. Software has been developed to automate a portion of the second step. The automated portion is identified in the broken-line box in Fig. 4.

The software developed under this effort is discussed in Section 3.1. Then certain aspects of the design procedure are presented in more detail in Section 3.2.

3.1 SOFTWARE

Several new software items have been developed under the present effort. The most important is an adaptation of the DPLR code to AEDC applications by Prof. Graham V. Candler. Code was also developed to accomplish the LSO, tracing of characteristics, and automation of the design procedure. Each of these is discussed below. The DPLR code is documented in files accompanying the software, and the remaining software is documented in the appendixes to the present report.

3.1.1 DPLR Flow Solver (aptu_nozzle.f)

The flow-solver development was accomplished by Prof. Candler at the University of Minnesota. The flow solver is referred to as DPLR, for Data Parallel Line Relaxation, which term describes the solution algorithm tailored to efficient implementation on parallel computers. The present code is derived from that described in Refs. 3 and 26 and is documented in Ref. 27. DPLR is a structured grid, time-dependent, finite-volume Navier-Stokes solver with chemical and vibrational nonequilibrium capabilities. The present and final form of the code may be described as follows:

- Axisymmetric or two-dimensional internal flow
- Viscous turbulent flow with the Spalart-Allmaras turbulence model
- Finite-rate chemistry for air, nitrogen, and products of combustion of isobutane with air (used in the AEDC APTU facility)
- Vibrational model based on Landau-Teller with a special correction for carbon dioxide
- High-pressure, thermally imperfect equation of state based on the excluded volume concept
- Erickson water condensation model for hydrocarbon combustion products
- Variable wall temperature
- Highly adapted for efficient parallel processing via message-passing interface protocol (MPI)

The DPLR code package includes the computer programs listed in the table below. Software details are given in a user manual that accompanies the flow-solver package. Additional information related to the present application is given in Appendix F, and the flow solver is described in Ref. 27. The software package has restricted distribution. Programs included in the DPLR flow-solver package are shown below.

Program/File Name	Function
aptu_nozzle.f	Flow solver for air, nitrogen, and APTU combustion products
nozgrid.f	Elliptic grid generator
aireq.f	Equilibrium air stagnation composition
n2eq.f	Equilibrium nitrogen stagnation composition
choneq.f	Equilibrium APTU gas stagnation composition
read.f	Post processor to generate plot files and objective function

3.1.2 Least-Squares Optimization (lso.f)

To accomplish the contour optimization, an objective function Q is chosen that measures flow nonuniformity and is to be minimized over a space whose coordinate directions are the design parameters α . In LSO, the objective function is written as the sum of square errors ε ,

$$Q(\alpha) = \sum_{grid} \varepsilon^2(\alpha) \quad (6)$$

where the error is defined as the difference between the current value of the flow property and the target value; that is,

$$\varepsilon \equiv (q - q^*) / q^* \quad (7)$$

where q is a vector of flow values at selected grid points in the test rhombus and q^* is the vector of target values. Since the q vector might include velocities, pressures, densities, etc., the differences are scaled by the target values (must be nonzero). The LSO equation is derived in Ref. 28. A simpler (perhaps nonrigorous) derivation is as follows:

Pass a multidimensional plane through $\varepsilon(\alpha)$:

$$\varepsilon = A\alpha + b \quad (8)$$

Find a correction, $\Delta\alpha$, to the current values of the design parameters so that

$$0 = A(\alpha + \Delta\alpha) + b \quad (9)$$

Subtracting these two relations gives

$$A\Delta\alpha = -\varepsilon, \quad (10)$$

which is a system of linear equations. The matrix A is actually a Jacobian matrix of the form $\partial\varepsilon/\partial\alpha = \partial q/\partial\alpha \equiv J$. Thus the system to solve is $J\Delta\alpha = -\varepsilon$. Unfortunately, there are typically many

more values for q than there are design parameters. Therefore, J is not square, and there are, in fact, many more equations than unknowns in the system. However, premultiplication by J^T , i.e.,

$$J^T J \Delta \alpha = -J^T \varepsilon \quad (11)$$

yields a coefficient matrix $J^T J$ which is square and converts the system to that for a least-squares fit. This is the well-known LSO equation that finds that α -vector for which Q is a local minimum (or possibly—but one hopes not—a saddle point). An iteration with counter v is then set up as

$$\alpha^{v+1} = \alpha^v + f \Delta \alpha \quad (12)$$

with $f \leq 1$, a relaxation factor to facilitate convergence.

A computer code, lso.f, was written to read in the q -vectors, compute the error vector and Jacobian matrix, and then solve the linear system for $\Delta \alpha$. The linear system is solved with code from Ref. 29 (pp. 129-133). Information on program usage is given in Appendix B.

3.1.3 Characteristic Tracing (RCFROMLC.FOR)

For the present design procedure of applying corrections over selected regions of a nozzle contour, there is a need to identify contour regions whose modification could reduce the exit flow nonuniformity. It is known that flow nonuniformities can sometimes be caused by surface anomalies on the nozzle contour. To trace possible sources for such nonuniformities, a computer program, RCFROMLC.FOR, was developed. This program allows the user to choose specific points at the nozzle exit and to trace characteristics back upstream to the contour, including through multiple wall and centerline reflections. The details of how this program operates and how to use it are given in Appendix G.

Typical results from RCFROMLC.FOR are shown in Figs. 6 through 8. Figure 6 shows a nozzle contour with a complete characteristic net composed of both right- and left-running characteristics. Notice that the characteristics extend outside of the nozzle coordinates. For this computation, four points were selected along the radius line at station 160. The left and right characteristics passing nearest to these points are then traced back upstream until they impinge on the wall or the centerline, at which point their reflection is traced further upstream. Figure 7 is an enlargement of the region between the throat and the inflection region of the contour, and Fig. 8 expands the immediate region of the throat. By carefully tracing these characteristics, the designer may associate any point in the nozzle exit flow with one or more points on the contour that may have influenced the exit flow.

3.1.4 Automated Nozzle Design Optimization (ANDO.t and ANDO2.t)

Two small software items were developed to automate a portion of the LSO process. The programs are ANDO.t and ANDO2.t. Their purpose is to automate the most tedious and labor-

intensive tasks of the LSO process. These programs are written in Tool Control Language (Tcl), Ref. 30, which is similar to Unix scripting but with many extensions. Although there are some machine dependencies in these two programs, Tcl is available for a wide variety of machines. These programs automate only Step 2 of the design procedure – splined correction of the contour – not Step 1 – optimization of the MOC contour. The MOC optimization was not automated because of the many more possible options to consider in the programming. Further, the complete iterative LSO process is not fully automated; only a single LSO step is processed, after which the designer must examine the computed corrections to the design parameters and decide which corrections and what fractions to apply to the next iteration. This partial automation is presently necessitated by the convergence difficulties encountered.

Before executing these procedures, the designer must prepare the flow solver as described in Appendix F and Ref. 27, including setting dimensions, compilation, computation of stagnation species, and setup of the input data files. The designer must also have created an initial contour, selected a distribution of small corrections (i.e., the design parameters), and selected perturbations to the corrections. The designer then interactively executes ANDO.t on the machine where the flow-solver solutions are to be computed. The ANDO.t program proceeds to create directories for each flow-solver solution, including one for the corrected baseline and one for each of the perturbations. ANDO.t copies the needed files to each of these directories, spawns a perturbed contour, prepares the batch job control file, and executes the grid generator. Note that the grid generator must be able to function without user intervention beyond initially preparing the input files. ANDO.t then submits a batch job for each flow-solver solution and terminates.

As a manual interim step, the designer must observe when the batch jobs have all finished and whether they ran satisfactorily. When ready to proceed, the designer executes ANDO2.t. This program proceeds to set up a directory for the LSO computation and copies the needed software and some of the data files. It then executes the postprocessor read.f for each flow-solver solution and copies the flow profile files to the LSO directory. Then the remaining input files for the LSO computation are prepared, and the LSO computation is accomplished. The program then terminates.

At this point the designer should examine the computed corrections to the design parameters to decide if they are reasonable. The designer can easily investigate the effect of varying the relaxation parameter f [Eq. (12)] by executing lso.f manually. If further corrections to the design parameters are to be pursued, the designer should run a flow-solver solution to compute the new value of the objective function to ensure that the chosen corrections to the design parameters actually reduce the objective function.

The programs ANDO.t and ANDO2.t are discussed in greater detail in Appendix A.

3.1.5 Other Software

Several additional programs are also documented in the appendixes. Briefly, these include a program (splinenc.f, Appendix C) to accomplish the spline fitting. The program nodepert.f (Appendix D) applies perturbation distributions to the input contour. The program obj.f (Appendix E) computes the objective function for a single DPLR solution, whereas lso.f described above, computes values of the objective function for multiple flow-solver solutions in the process of accomplishing a least-squares optimization. The program EFFCPG.FOR (Appendix H) computes an effective thermally and calorically perfect gas model given equilibrium nozzle exit flow conditions. The program NODELOCS.FOR (Appendix I) is used to locate nodes along the contour according to an appropriate clustering scheme. Finally, the program NOZCOR.FOR (Appendix J) computes a sinusoidal contour perturbation with a single maximum point at a specified location along the contour.

3.2 DESIGN PROCEDURE

3.2.1 Contour Initialization

The first step in the design procedure is selection of an initial contour based on the design requirements. The design requirements usually include specification of flow stagnation conditions, nozzle exit conditions (usually Mach number), and geometric constraints, including the nozzle exit dimension (radius or half-height) and often a length limitation. The final step of the contour initialization procedure is generation of a contour using an MOC code that assumes a TCPG. Since this assumption is discrepant relative to flows of primary interest, additional effort is involved in the initialization.

Under the assumptions of a TCPG, the specific heat ratio ($\gamma = C_p/C_v$) and the gas constant (R , as in $P = \rho RT$) are constant. To employ the MOC code CONTUR, it is necessary to relate these two parameters to the actual gas with which the nozzle must operate. The present procedure is to find an *effective* TCPG model that accurately represents certain important aspects of the nozzle flow. To determine the TCPG model, the NASA Glenn Chemical Equilibrium with Applications code (CEA96, Ref. 31) is used. This code computes stagnation properties and quasi-1D flow, assuming chemical equilibrium for a wide range of substance mixtures. It is important to note that CEA96 assumes a thermally perfect gas and does not contain a correction for high-pressure effects. For present purposes, CEA96 is used to compute an isentropic expansion from the specified stagnation conditions to the target Mach number. It is necessary to iteratively enter the exit area ratio until the desired exit Mach number is obtained. This calculation then establishes the inviscid area ratio of the nozzle and the exit static conditions. The effective γ is computed from Ref. 32, Eq. (80), as follows:

$$\frac{A}{A^*} = \frac{1}{M} \left[\frac{2}{\gamma + 1} \left(1 + \frac{\gamma - 1}{2} M^2 \right) \right]^{\frac{\gamma + 1}{2(\gamma - 1)}} \quad (13)$$

using the target Mach number M and the area ratio A/A^* found with CEA96. The effective gas constant is obtained from the relation

$$\frac{A^*}{A} = \left(\frac{\gamma+1}{2}\right)^{\frac{\gamma+1}{2(\gamma-1)}} \left(\frac{V}{a_0}\right) \left[1 - \frac{\gamma-1}{2} \left(\frac{V}{a_0}\right)^2\right]^{\frac{1}{\gamma-1}} \quad (14)$$

from Ref. 32, Eq. (82) using the area ratio, exit velocity V from CEA96, and the effective γ , noting that $a_0^2 = \gamma RT$. These effective γ and R values are later used in the MOC code to ensure that the initial inviscid contour produces the target Mach number at the equilibrium area ratio. If the flow is not in equilibrium, further adjustment of the nozzle exit diameter might be required during the optimization. A small computer code EFFCPG.FOR (Appendix H) was written to accomplish the effective TCPG calculation, which is an iterative solution of Eqs. (13) and (14).

The importance of the gas model assumption is illustrated in Figs. 9 through 12. Shown in the plots are the static pressure, temperature, and density, along with velocity versus Mach number for three thermodynamic models. The models include 1) thermally and calorically perfect air with specific heat ratio $\gamma = 1.4$ and gas constant $R = 287$ J/kg-K; 2) an effective TCPG model for Mach number 4 at a specific area ratio based on equilibrium air ($\gamma = 1.190$, $R = 311$ J/kg-K); and 3) equilibrium air (Ref. 31) for stagnation conditions of 100 atm and 5000 K. In Fig. 9, the effective TCPG model agrees well with equilibrium air while the $\gamma = 1.4$ increasingly disagrees at higher Mach numbers. In Fig. 10, the static temperature for the effective TCPG model does not agree as well as the static pressure but is closer at the exit. In Fig. 11, the density discrepancy for $\gamma = 1.4$ is larger than for the static pressure. The results are similar for velocity (Fig. 12). This calculation illustrates that if a TCPG model must be used outside of its preferred range of applicability, its accuracy can be improved with judicious choice of values for specific heat ratio and gas constant.

The next step in the contour initialization procedure is to design an inviscid contour via the method of characteristics with the code CONTUR. This code is described in some detail in Ref. 1. The input to CONTUR includes the effective γ and R values given above and a Sutherland fit to the viscosity of the test medium.* After the inviscid contour is computed, the grid established by the characteristic net is revised to ensure reasonably uniform variation of the characteristic spacing on the wall. Once this grid refinement is accomplished, the boundary-layer correction in the MOC code can be activated to obtain the initial viscous contour. Manual iteration is necessary if the final viscous contour must have a specified exit diameter. Normally, a quality check is then accomplished to ensure that the coordinates, slopes, and second derivatives along the contour are smooth and monotonic where required. If problems are found in the contour, the control parameters for the inviscid contour may need further adjustment.

* Private communication, E. S. Powell, 5 Mar 2003.

Sivells' method offers several design options in terms of how the centerline Mach number distribution is represented. The centerline distribution can be 1) in two segments separated by a radial flow region; 2) a single segment from the throat to the start of the test rhombus; or 3) read in from a user-prepared data file. Experience has shown that the highest flow quality (lowest nonuniformity) is obtained from two-segment nozzles with a radial flow region. However, such designs tend to be relatively long. With the two-segment option, it may be difficult to satisfy length constraints when the nozzle must be used in an existing facility. For high-enthalpy facilities such as combustion-heated or arc-heated facilities, a common design philosophy is that the nozzle should be as short as possible to minimize the loss through wall heat transfer of the elaborately achieved high enthalpy. Although experience has also shown that relatively short nozzles tend to have poorer flow quality than longer nozzles, the selected trade-off usually favors reduced enthalpy loss and facility length constraints. The single-segment Mach number distribution is used in the demonstration design presented in this report, but this decision is not critical to the design procedure. Within the single-segment option, there still are alternatives available. The centerline distribution may be represented by a cubic, quartic, or quintic polynomial. As the polynomial degree increases, additional end conditions are needed to define the polynomial coefficients, and these are provided by matching higher order derivatives with the transonic solution. In some cases, use of a quartic or quintic polynomial will allow a shorter nozzle than will the cubic. However, particularly with the quintic, there is no guarantee that the Mach number will remain monotonically increasing, moving downstream.

To complete the design, a contraction contour upstream of the throat must be developed. To date, contractions have traditionally been designed more on the basis of subjective selection of geometry than on the basis of quantitative fluid flow physics. The contraction is usually an analytical function that is piecewise continuous through second derivative. The function is chosen to match the coordinate, slope (zero), and curvature of the viscous contour at the throat. The dimensions of the contraction inlet are usually given as design requirements.

3.2.2 Contour Representation

To be optimized, a contour must be represented mathematically in terms of design parameters that can be perturbed to assess their effect on flow quality. The number of design parameters must be kept to a minimum because each one will require multiple flow solutions to compute the sensitivities. Also, a large number of parameters lessens the stability of the optimization iteration.

One favored method of representing a contour initially defined only as a table of numbers is the well-known cubic spline. In this method, a cubic polynomial is placed between each pair of adjacent coordinates; slope and second derivative are required to be continuous from cubic segment to cubic segment. Splines are used to avoid the large oscillations that can occur when a single higher degree polynomial is placed through many points. For the design optimization problem, the coefficients in the spline would then be the design parameters. If cubic splines are to

be suitable for the present application, they must be able to represent a contour with acceptable accuracy and without nonmonotonicity in slope and second derivative. Furthermore, if the spline coefficients are to be design parameters, they must be limited in number. Since an MOC contour may typically have hundreds of contour points, clearly only about 1/10 of this number may be used to spline the contour.

To evaluate the ability of splines to accurately represent an entire contour, investigators fitted a specific nozzle contour designed with MOC. The contour chosen for the fitting was that for a relatively long and gradual contour designed for excellent aerodynamic flow quality (see inset in Fig. 13). This contour was chosen (from four contours discussed in Section 4.1, below) because it has the least severe gradients in slope and second derivative of the given alternatives and should be the easiest to fit accurately. The MOC contour had 224 points from the throat to the exit. About 1/10 (23) of these were used to compute a spline fit. The resulting fit was then used to compute (interpolate) the coordinates at all 224 points, and the differences from the original coordinates were computed. These differences are shown in Fig. 13. The finding is that the spline fit represents the original contour to within ± 0.001 in. However, on AEDC machine drawings, coordinates are usually tabulated to 0.0001 in. While the tabulated accuracy is probably better than the actual machining accuracy, the machining accuracy is considerably better than ± 0.001 in. Thus it appears that splines introduce contour oscillations larger than machining accuracy. Further, if the coordinates acquire oscillations, the derivatives will also suffer accordingly. As a result, the decision here was not to use cubic splines to represent the contour in the design optimization.

An alternate representation would be to use MOC itself as the function generator. Experience has demonstrated that MOC contours, if carefully computed, will be smooth through second derivative and monotonic where desired. Furthermore, the present analysis of candidate APTU contours from MOC indicates that the flow quality of these contours can be very good, if not quite good enough. If MOC should be chosen as the function generator, then selected input variables would be chosen as the design parameters. However, much has been made here of the inadequacy of the equations of state and viscous corrections available with MOC. A design optimization limited to pure MOC contours could not transcend these deficiencies. Accordingly, some additional correction of the MOC contour would be necessary. The correction then must be parameterized and represented as a generating function. This is the approach taken here.

A fundamental aspect of the present design philosophy is to alter a candidate MOC contour with a splined correction. The basic contour is represented with a table of coordinates that is slightly altered with a correction distribution represented with cubic splines through a limited number of nodes. The radial corrections at these nodes are the design parameters that are themselves perturbed to improve the flow uniformity at the nozzle exit. This approach breaks with past practice, where the entire contour was represented with cubic splines and the spline coefficients were the design parameters. This previous practice faced a dichotomy of opposing requirements: the need to minimize the number of design parameters and, therefore, the number

of nodes (since each requires a separate flow-solver evaluation) and the need to accurately represent the contour.

Thus, the need for two different Navier-Stokes-based optimization steps emerges. The first step optimizes the MOC contour using the MOC code as the function generator for the contour. The accuracy of this process is, of course, limited by the physics embodied in the MOC code, which, by itself, has been argued to be inadequate for hypersonic nozzles. The second step of optimization finds a splined correction to the MOC contour that yields minimum flow nonuniformity.

3.2.3 MOC Contour Optimization

Step 1 of the contour design is optimizing the MOC design itself. The procedure is to vary the contour, selecting only candidate contours that are generated by the MOC design code. Contour variations are generated by varying the values of selected input variables to the MOC code. The contour must depend continuously on the input parameters because the variation of the nozzle exit flow properties will be divided by the variation of the design parameters to obtain the finite-difference derivatives that make up the Jacobian matrix. If the contour dependence on the input parameters were not continuous, the Jacobian formally would not exist.

Sivells' CONTUR program has many potential input parameters that might serve as design parameters. Following is a list of potential design parameters:

- GAM = specific heat ratio
- AR = gas constant in ideal gas equation
- SFOA = location of radial flow region
- ETAD = inflection angle
- RC = ratio of wall radius of curvature at throat to throat radius
- FMACH = centerline Mach number at Sivells' point F (see Fig. 1)
- BMACH = centerline Mach number at Sivells' point B, the end of the radial flow region (see Fig. 1)
- CMC = nozzle exit Mach number
- SF = inviscid nozzle exit radius
- XC = nozzle length control parameter (actual definition depends on other options selected)
- PPQ = stagnation pressure
- TO = stagnation temperature

See Ref. 1 for a more detailed description. The set of variables available to the designer depends directly upon the other program options chosen. For example, if a single centerline velocity distribution (no radial flow region) is chosen (ETAD = 60), then SFOA, FMACH, and BMACH are not available for use as design parameters. It might seem inadvisable to consider such variables as the exit Mach number, stagnation pressure, or stagnation temperature as design parameters since they are normally firmly specified as design requirements. However, the

stagnation conditions are independently input to the Navier-Stokes flow solver, and the exit Mach number (or perhaps flow speed) is a component of the objective function returned by the flow solver. Thus, there is the possibility that small perturbations in the MOC input would have a useful effect on the contour design. The stagnation conditions (particularly the pressure) would offer some limited control over the boundary-layer correction computed by CONTUR.

The designer should be aware that, when the geometry at the throat is altered during the design process, the contraction design may need to be updated to maintain continuity of radius and curvature at the throat.

The design optimization of the MOC contour proceeds as illustrated in Fig. 5, Step 1. After the initial MOC contour is selected, a DPLR flow-solver solution should be computed to establish the initial value of the objective function and to qualitatively assess the exit flow uniformity. Next, for each CONTUR input variable selected as a design parameter, a perturbed value of that variable should be selected and a new MOC contour generated. DPLR flow-solver solutions should then be computed for each design parameter perturbation. The collection of flow profiles from these solutions is then used in an LSO computation to obtain corrections to the initial design parameters. It is probable that the corrections computed by Iso.f will need to be reduced (i.e., the relaxation factor f in Eq. (12) must be chosen to be less than one). When new values of the design parameters are selected, a new MOC contour should be designed with CONTUR and a DPLR flow solution computed. If the objective function appears to be at a minimum, the design iteration can be terminated. It may be necessary to carry the design process one iteration beyond the minimum point to verify graphically or numerically that the minimum point has indeed been passed.

3.2.4 Optimization of Splined Corrections

After optimization of the MOC contour has been completed, Step 2 of the design may commence. The process is illustrated in Step 2 of Fig. 5. First, the MOC contour is to be altered with a distribution of small corrections (radius perturbations) at a limited number of stations (nodes) along the contour (Fig. 14). Correction values are slightly perturbed, one at a time, to generate a sequence of contours. A flow-solver solution is to be computed for the initial correction, followed by one for each perturbation of the corrections. To proceed, the designer must make the following decisions:

- Number of nodes
- Range of nodes
- Location and distribution of nodes
- Size of corrections
- Size of perturbations of corrections

The designer should attempt to limit the number of nodes since each will require a flow-solver solution for each successive set of perturbations. The range of the nodes can be the full nozzle length including the contraction or any subset. Note that the first and last nodes each must

be located at a point on the initial tabular contour and that the first and last corrections must be zero. There is no restriction on node spacing, and the designer may wish to cluster nodes in some region of the contour such as the high-curvature region between the throat and the inflection point. A clustering program is described in Appendix I. The magnitude of the corrections should be small compared to the local radius but larger than the expected machining accuracy. The corrections may be positive, negative, or zero (actually, all may be zero) and may change signs from node to node. The perturbations of the corrections should be small compared to the corrections, and the signs may be positive or negative. Again, the perturbations at the first and last nodes should be zero. Once the correction distribution and the perturbations are selected, the designer may proceed with the LSO process. This portion of the design procedure is automated as described in Section 3.1.4 and Appendix A. The automated process accomplishes a single LSO step after which the designer must evaluate the predicted update of the corrections and decide how much of the update to accept. The designer must then decide whether to continue or terminate the LSO process, depending on whether the objective function has been minimized. When the LSO for one set of corrections is finished, the designer may wish to apply additional corrections, perhaps concentrated in some other region of the contour. Not a great deal of experience has been accumulated yet on applying the present procedure. The demonstration design presented in Section 4.0 summarizes experience to date.

4.0 RESULTS

This section presents the results of applying the above procedure to designing a Mach number 6 nozzle for the AEDC APTU facility. The demonstration nozzle design addresses the following design requirements:

Nominal exit Mach number	6
Stagnation pressure	4.595 MPa (666.5 psia)
Stagnation temperature	1592 K (2865°R)
Exit diameter	1.067 m (42 in.)
Maximum nozzle length (inlet to exit)	4.29 m (169 in.)
Test medium	isobutane, LOX, air
Flow nonuniformity (any parameter)	±2 percent

Since the flow leaving the combustion burners is fairly nonuniform, the flow nonuniformity requirement is interpreted to mean that the nozzle must add no more than ±2 percent to the existing nonuniformity. The function of the nozzle is to expand a uniform subsonic flow into a uniform hypersonic flow, but it may not be possible to design a contour to remove significant nonuniformity already present in the subsonic flow.

4.1 CONTOUR INITIALIZATION

To begin the contour design, investigators followed the procedure outlined in Section 3.2.1. In preparation for designing a contour using the MOC code CONTUR, an effective TCPG model was constructed using CEA96 and EFFCPG (Appendix H). Results obtained for the APTU Mach 6 design are shown below.

Inviscid exit area ratio (CEA96)	84.341
Exit Mach number (CEA96)	6.053
Exit speed (CEA96)	1798.8 m/s
Exit static pressure (CEA96)	1898 Pa
Exit static density (CEA96)	0.029828 kg/m ³
Exit static temperature (CEA96)	223.47 K
Effective γ (EFFCPG)	1.3437
Effective R (EFFCPG)	301.24 J/kg-K

It is worthwhile to reiterate that the effective TCPG model is used only in the MOC and that full nonequilibrium thermodynamics are accounted for by the DPLR flow solver.

The next step in initializing the contour is to design a contour with CONTUR using the effective gas model developed above. For the APTU Mach 6 application, four MOC designs were developed as candidates for the initial contour. Each design was analyzed with DPLR to assess the initial flow quality. The designs differ in the selection of the centerline Mach number distribution used by CONTUR and in the amount of theoretical nozzle length truncated to meet the length constraint of 169 in. (159 in. from throat to exit). The notation “DesignX” was adopted for identification of the various designs, where X=1,2,...,8,9,A,B, The first four designs using SIVELLS were those shown below.

Design1	Single quintic centerline Mach distribution; small throat radius of curvature; truncated 26 in.
Design2	Single cubic centerline Mach distribution; large throat radius of curvature; truncated 26 in.
Design3	Dual quartic centerline Mach distribution up- and downstream of a (conical) radial flow region; not truncated.
Design4	Single quintic centerline Mach distribution; intermediate throat radius of curvature; truncated 1 in.

Completion of the design requires a contraction contour to be developed upstream of the throat. The contraction functions used here were either a monotonic supercubic (Design1, 2, 4, 5, ...) or a quartic-cone-quartic with continuous curvature (Design3). The contraction contours are not optimized here.

The four initial viscous-corrected MOC contours for the APTU application are shown in Fig. 15. Design1 uses a relatively large throat wall radius of curvature, which is sometimes said to be associated with good flow quality. Design2 uses a smaller radius of curvature to place maximum curvature at the throat, which also is recommended for good flow quality. Both Design1 and Design2 are truncated significantly from their theoretical length. Design3 is a traditionally good aerodynamic contour that uses a radial flow region and ignores the length constraint (i.e., it represents a full theoretical length). Design4 ignores the maximum curvature recommendation and shortens the theoretical length as much as possible to minimize truncation length (1 in.). The characteristics of the four initial contours are summarized below.

	Design1	Design2	Design3	Design4	Design9
RC	18	4	12	4	4.45
XC	30	-1	0	31	18.86
M(x)	5 deg	3 deg	4/4 deg	5 deg	5 deg
ETAD, deg	19.2	21.4	9	27.0	26.4
Theoretical length, in.	185.9	185.5	281.7	160.8	160.8
Truncation length, in.	159	159	281.7	159	159

The flow in each contour was computed using DPLR as adapted for the APTU chemistry model to assess the exit flow quality. The computed flow profiles at the nozzle exit, including the boundary layer, are shown in Figs. 16 through 22. Figure 16 shows the axial velocity profile for all four contours along with the equilibrium exit velocity from CEA (black symbols). Since CEA96 is an inviscid, quasi-1D computation, its single predicted exit value for velocity is plotted as a vertical column of symbols. Clearly, all four contours yield very good uniformity in terms of axial velocity. Note that the boundary layer on the Design3 contour is slightly thicker than that on the three shorter nozzles. This simply reflects the greater length of Design3. Figure 17 shows the exit profile of the axis-normal or radial velocity component. The maximum normal velocity here (Design1) corresponds to a flow angle (Fig. 18) of about 0.9 deg, the computation of which is based on the equilibrium exit velocity. Clearly, the long nozzle Design3 has the lowest flow angularity and is within ± 0.25 deg. Second lowest is Design4, which has a flow angularity of easily less than ± 0.5 deg. Figure 19 shows the exit static pressure profiles. The nonuniformity exhibited here is significant and is unacceptable for Designs1, 2, and 4, though Design4 is the best of the three short nozzles. A similar situation is seen for the static density profiles in Fig. 20. For both pressure and density, there is a significant difference between the equilibrium values and

those from the flow solver. The differences here are large relative to the mean static pressure but small relative to the stilling chamber stagnation pressure. Figures 21 and 22 show the static translational/rotational temperature and the vibrational temperature profiles, respectively. The temperature differences between Figs. 21 and 22 are directly attributable to vibrational nonequilibrium effects, which is to say, to the energy stored in the relative vibration of the individual atoms within the molecules (and thus they are analogous to vibrating masses connected by springs).

Figures 23 through 27 show the streamwise flow variation along the nozzle centerline. In each of these five figures, it is important to observe that the property distributions are not monotonic for Designs 1, 2, and 4, while Design3 (the long nozzle) is monotonic. The nonmonotonicity, which is generally regarded as undesirable, does not degrade the exit flow quality in Design4 as much as it does in the other two short designs. Reacting flow effects are illustrated by Figs. 28 through 31. These four figures show the species distributions in terms of mass fraction for each design. Note the clear grouping of the four most prevalent species (N_2 , O_2 , CO_2 , and H_2O) followed by NO and then a further grouping of the least prevalent species (OH , O , CO , H , and H_2). The five most prominent species show no variation from the scale plotted, whereas the five least prominent species show variation in the vicinity of the throat.

4.2 MOC CONTOUR OPTIMIZATION

In preparation for the design optimization, the objective function components chosen were the axial velocity component, u , the axis-normal velocity component, v , the static pressure, P , and the static density, ρ . These flow properties were differenced from the target values and scaled as follows:

$$Q = \sum_{grid} \left(\frac{u - u_{equil}}{u_{equil}} \right)^2 + \sum_{grid} \left(\frac{v}{u_{equil}} \right)^2 + \sum_{grid} \left(\frac{P - P_{mean}}{P_{mean}} \right)^2 + \sum_{grid} \left(\frac{\rho - \rho_{mean}}{\rho_{mean}} \right)^2 \quad (15)$$

where the target value u_{equil} is the equilibrium exit speed obtained from the CEA96 code and has the value 1798.8 m/s. Note that the target value of v is zero and that scaling by the axial velocity converts the quantity to the tangent of a flow angle. The mean values are the values of the Design4 DPLR solution and are held fixed during the optimization. The mean pressure and density were chosen because of the large difference between the equilibrium values and because uniformity is more important than the specific values. The summations are taken over 56 grid points at the nozzle exit along an axis-normal line starting on the centerline and moving toward the wall. The grid points in the boundary layer were excluded from the objective function.

Figures 32 through 37 show nozzle exit flow profiles exclusive of the boundary-layer region. Figures 32 through 35 represent the components of the objective function. These plots show the difference between the flow properties from DPLR and the equilibrium values from the CEA96 code. The equilibrium values represent the ideal nozzle, which would be sufficiently long to

allow all reactions and vibrational energy transfers to reach equilibrium. The differences between DPLR and CEA96 thus would be indicative of thermodynamic processes that have not gone to completion. Figure 32 shows the difference between the DPLR streamwise velocity component and the equilibrium value from the CEA96 code. All designs are already within the $\pm 1/4$ -percent requirement, with Design3 being the best and Design4 the next best. The axis-normal velocity in Fig. 33 is within the ± 2 -percent tolerance (for propulsion testing) with Design3 and Design4 again being best and second best, respectively. The static pressure profiles are shown in Fig. 34. Design3 and Design4 both meet the ± 2 -percent criterion, but the other two nozzles show much larger variations, probably because of 1) decisions regarding the MOC design that affect the downstream throat region, and 2) their significant truncation from the theoretical length. Figure 35 shows the static density profiles. While Design3 still meets the ± 2 -percent criterion, Design4 does not. The density also shows significant differences from equilibrium. Figures 36 and 37 show the static temperature and vibrational temperature. Clearly, Design3 is the best of the designs, and Design4 is superior in most respects to Design1 or 2, one exception (for Design4) being the vibrational temperature shown in Fig. 37. Accordingly, Design4 was chosen as the best length-compliant alternative for the starting point of the design optimization.

Step 1 of the design optimization began with perturbing the design parameters, one at a time, to radius ratio (RC) = 5 and length ratio (XC) = 18.02. The first optimization step yielded RC = 4.245 and XC = 20.10. The XC value exceeded a maximum limit associated with the assumed quintic centerline Mach number distribution. Accordingly, XC = 18.86 was accepted as the final value, and the optimization proceeded with one design parameter, RC. RC = 4.35 was the next perturbation. The optimization step yielded RC = 4.45, and here the iteration was terminated, since this appeared to be a minimum point. Following is a summary of the optimization and perturbation steps and the values of the objective function.

Design Step	Radius Ratio (RC)	Length Ratio (XC)	Objective Function (Q)	Reduction of Objective Function, Percent of Design4
Baseline (Design4):	4	18	0.01922	0
Perturbation	5	18	0.11049	
Perturbation	4	18.02	0.02088	
Optimization	4.245	18.86	0.01396	27
Perturbation	4.35	18.86	0.01025	
Perturbation (Design9)	4.45	18.86	0.009595	50

As shown above, the objective function was reduced by 50 percent over the three contours. Figure 38 shows the objective function plotted versus RC. It appears that the objective function is near a minimum point. The effect of the optimization process on flow quality (the objective function

components) is shown in Figs. 39 through 43. The variation of axial velocity (Fig. 39) from uniform flow is already less than ± 0.25 percent and does not improve with iteration. The overall value does increase slightly, reflecting a higher value of the equilibrium velocity. The axis-normal velocity component (Fig. 40) shows a larger spread but is almost within ± 0.25 percent. However, the uniformity is improved observably by the iteration. The static pressure (Fig. 41) shows a much larger variation of about ± 2 percent but is improved to nearly ± 1 percent over the three contours. The static density (Fig. 42) shows a variation that is initially greater than ± 2 percent but is within the tolerance at the end of the iteration. These four results are replotted to the same horizontal scale in Fig. 43 to permit comparison of the relative variation. Clearly, the velocity components show much smaller variation than the static pressure and density. The improvement in flow quality here is observable but not dramatic. This indicates that the flow quality of the initial contour was actually fairly good and did not allow for great improvement. However, the improvement obtained here was fortuitously sufficient to bring all four flow variables to within the ± 2 percent requirement for propulsion testing.

4.3 OPTIMIZATION OF SPLINED CORRECTIONS

Once the optimization of the MOC contour was completed, the design process proceeded to Step 2, which has been termed “optimization of splined corrections.” Before the splined correction capability was implemented, a less elaborate correction function based on $\sin^n x$ was tested to evaluate the feasibility of the correction approach. The argument “x” was mapped to allow positioning of the maximum correction point anywhere on the contour. (Details are given in Appendix J.) The initial correction was applied so that the maximum point occurred at the inflection point. The design parameters were the maximum correction and the station of the maximum correction. The optimized result was DesignG (see listing below), which yielded a small but observable reduction in the objective function. The optimized correction value was -0.004882 in. at station 2.99082 in. from throat (inflection point is at 3.30157 in.). Progress on the objective function is illustrated in Fig. 44. The plotted value of the objective function in Fig. 44 is a percent of its value for the unoptimized MOC contour (Design4).

Design Identified	Description
Design4	Initial MOC contour before optimization
Design9	Optimized MOC contour
DesignG	Two-parameter, single node correction about inflection point
DesignM	Three-parameter correction over full length of nozzle
DesignU	Three-parameter correction from throat to inflection point
DesignAI	Nine-parameter correction over full length of nozzle

Next, a three-parameter, splined correction was applied over the full nozzle length. The final set of applied corrections (units are in inches) resulting in DesignM is given below.

<u>Station</u>	<u>Correction</u>	
0	0.000	THROAT
39.75	-0.005	node1
79.5	0.001	node2
119.25	0.005	node3
1.5900000E+02	0.000	EXIT

This design iteration yielded a significant further reduction in objective function (see Fig. 44), now at about 31 percent of the unoptimized MOC design.

Next, a second three-node spline correction was applied where the node locations were chosen on the basis of results from tracing streamlines with the program RCFROMLC (Appendix G). For this iteration, both the corrections and the node locations were first perturbed. However, this approach was unsuccessful in reducing the objective function. Perturbing only the corrections was not much more successful in that it produced only a small reduction in objective function. This final result of this iteration was DesignU (not called out in Fig. 44).

At this point, the *automated* nozzle design optimization procedure (ANDO) was developed and immediately applied to a nine-parameter correction over the full length of the nozzle. The node locations were chosen using the program NODELOCS.FOR (Appendix I). Figure 14 illustrates the perturbations with the correction magnitude exaggerated for purposes of illustration. Two nine-perturbation LSO computations were performed (ten DPLR jobs for each). In general, it was found that perturbation of any individual node correction would yield at least a small reduction of the objective function. However, when all design parameter corrections were applied simultaneously, the objective function was found to increase, even when only a small fraction of the LSO-predicted correction was applied. Accordingly, the approach used was to apply corrections only to the nodes with the three largest predicted improvements. This led to the final DesignAI with the following node corrections (in.) applied to the DesignU contour:

<u>Station</u>	<u>Correction</u>	
0.0000000E+00	0.0000	node00
1.5900000E+00	0.0000	node01
5.2871610E+00	0.0000	node02
1.1467187E+01	0.0000	node03
2.0599835E+01	0.0015	node04
3.3266834E+01	0.0000	node05
5.0155869E+01	-0.0015	node06
7.1946970E+01	0.0015	node07
9.8794221E+01	0.0000	node08
1.2878993E+02	0.0000	node09
1.5900000E+02	0.0000	node10

The degree of improvement in flow quality from the unoptimized contour to DesignAI is illustrated in Figs. 45 through 48. These figures show the deviation from target values of the four flow properties selected as components of the objective function. In Fig. 45, the axial velocity deviation ratioed to the equilibrium exit velocity is shown for four critical points in the design process. Note first that the axial velocity deviation is considerably less than ± 0.25 percent. This ameliorates the fact that the axial velocity nonuniformity actually increases as the design proceeds. In Fig. 46 the scaled axis-normal velocity (effectively a flow angle) improves observably but was already better than ± 0.5 percent. The static pressure deviation (Fig. 47), however, shows dramatic improvement over the initial MOC contour, having decreased from about ± 1.8 percent to about ± 0.6 percent. Finally, the static density deviation (Fig. 48) improved less dramatically, from about ± 2.5 to ± 1.4 percent.

At this point, the design iteration was terminated. However, all possible perturbation alternatives have not nearly been exhausted. Accordingly, it may be expected that there remains potential for further reduction of the objective function. However, the process has been carried far enough to demonstrate that the procedure advocated here can yield significant improvement in flow quality over the unoptimized MOC contour.

4.4 CRITIQUE OF METHOD AND RESULTS

It is a not uncommon opinion among designers that the field of design optimization is as much art as science. Consistent with that view, it is not claimed that the present method represents an end product in nozzle contour design. Rather, the present method is simply one of several available alternate approaches that has been shown to have potential worthy of further examination and engineering application.

Accordingly, there are several caveats in the present approach that must be explicitly stated as such. The objective function used here contained four flow variables: pressure, density, and two components of velocity. However, since a chemically reacting flow is, thermodynamically speaking, not a simple substance for which all thermodynamic properties may be mathematically determined if any two are specified (Ref. 33, p. 79), it can be argued that additional thermodynamic variables should be included in the objective function. The present choice of four was based on a philosophy of simplicity of approach during feasibility investigation and learning. Future designs should consider including additional flow variables, such as species profiles. A second caveat is the observed nonconvergence of the LSO procedure. It is suspected that nonconvergence is inevitable based on the derivation given in Section 3.1.2. That is, seeking the zeroes of a plane through data assumes that zeroes actually exist in the data, while the probable reality is that only nonzero minima exist. Finding such minima would require retention of second-degree terms in the least-squares fit and proportionately many more flow-solver evaluations. The present observed behavior of the objective function seems to support the expectation of nonconvergence, but there may be another, undiscerned explanation. A third caveat concerns representation of the contour and corrections thereto. In applying a method of splined corrections,

the present work examined only a few alternatives, including partial-length corrections and full-length corrections, some with node clustering. Within even these limited bounds, there are many more and probably better choices yet to be tested. A final caveat is the presentation of the present method as a reacting flow design procedure. That, of course, is the specific objective here. However, the connection between the optimization procedure and the range of physics accounted for in the design is tenuous, and almost any competent flow solver with the appropriate physics would suffice. The present design procedure could have been demonstrated almost equally well for an inviscid flow of a TCPG.

5.0 SUMMARY AND CONCLUSIONS

A procedure has been presented for the optimization of test facility nozzle contours when reacting flow effects are important. The procedure builds on the already considerable design capability offered by the classical method of characteristics (MOC). A two-step procedure is developed: Step 1 optimizes contours generated by an MOC code using a reacting Navier-Stokes flow solver; Step 2 optimizes a splined correction distribution applied to the optimized MOC contour to account for reacting flow effects not admitted by a typical MOC program. Optimization is accomplished with the well-know least-squares optimization (LSO) process. Software is developed to accomplish the optimization and to automate a portion of the process. The procedure was demonstrated by design of a Mach number 6 nozzle for a combustion-heated test facility.

The following conclusions are advanced on the basis of the present work.

- The MOC design procedure remains quite viable for designing a hypersonic nozzle, even with reacting flow effects not fully treated. Some of this viability retention depends on using a carefully developed model for an effective thermally and calorically perfect gas (TCPG). However, there remains considerable potential for improvement of flow quality beyond MOC capabilities through LSO.
- The concept of optimizing a splined correction distribution applied to an existing contour has been demonstrated and shown to be a viable approach to including thermodynamics effects not accounted for by the initial contour.
- The numerical iteration of the LSO procedure for the nozzle problem has not been found to be nicely convergent, and some user intervention is necessary. However, a direct search based on the sensitivities is an effective approach.

REFERENCES

1. Sivells, J. C. "A Computer Program for the Aerodynamic Design of Axisymmetric and Planar Nozzles for Supersonic and Hypersonic Wind Tunnels." AEDC-TR-78-63 (AD-A062944), December 1978.

2. Sivells, J. C. "Aerodynamic Design of Axisymmetric Hypersonic Wind Tunnel Nozzles." *AIAA Journal of Spacecraft and Rockets*, Vol. 7, No. 11, November 1970, pp.1292-1299.
3. Wright, M. J. and Candler, G.V. "Data-Parallel Line Relaxation Method for the Navier-Stokes Equations." *AIAA Journal*, Vol. 36, No. 9, September 1998, pp. 1603-1609.
4. Glowacki, W. J. "FORTRAN IV (IBM 7090) Program for the Design of Contoured Axisymmetric Nozzles for High Temperature Air." NOLTR-64-219, February 1965.
5. Nelms, L. T. "Design of Minimum Length Supersonic Nozzle Contour and Associated Subcontours." AEDC-TR-68-212 (AD-697769), November 1968.
6. Johnson, C. B. and Boney, L. R. "A Method for Calculating a Real-Gas Two-Dimensional Nozzle Contour Including the Effects of Gamma." NASA TM X-3243, September 1975.
7. Simeonides, G. "The Aerodynamic Design of Hypersonic Contoured Axisymmetric Nozzles Including Real Gas Effects." VKI Technical Memorandum 43, March 1987.
8. McCabe, A. "Design of a Supersonic Nozzle." British Ministry of Aviation, Aeronautical Research Council, R. & M. No. 3440, 1967.
9. Vanco, M. R. and Goldman, L. J. "Computer Program for Design of Two-Dimensional Supersonic Nozzle with Sharp-Edged Throat." NASA TM X-1502, January 1968.
10. Argrow, B. M. and Emanuel, G. "Comparison of Minimum Length Nozzles." *Journal of Fluids Engineering* (ASME), Vol. 110, September 1988, pp. 283-288.
11. Beckwith, I. E., Ridyard, H. W., and Cromer, N. "The Aerodynamic Design of High Mach Number Nozzles Utilizing Axisymmetric Flow with Application to a Nozzle of Square Test Section." NACA TN 2711, 1952.
12. Haddad, A. and Moss, J. B. "Aerodynamic Design for Supersonic Nozzles of Arbitrary Cross Section." *Journal of Propulsion* (AIAA), November-December 1990, Vol. 6, No. 4, pp. 740-746.
13. Varner, M. O. "Application of Optimization Principles to Flexible Wall Nozzle Design - A Case Study." AIAA-86-0775, 1986.
14. Brodsky, S. L. "Supersonic Nozzle Design." Naval Ordnance Laboratory, NOLTR 70-131, September 1970.
15. Korte, J. J., Kumar, A., Singh, D. J., and White, J. A. "CAN-DO, CFD-Based Aerodynamic Nozzle Design Optimization Program for Supersonic/Hypersonic Wind Tunnels." AIAA-92-4009, AIAA 17th Aerospace Ground Testing Conference, Nashville TN, July 1992 (A92-56832).

16. Keeling, S. L. "A Strategy for the Optimal Design of Nozzle Contours." AIAA 93-2720, 28th AIAA Thermophysics Conference, Orlando, FL, July 6-9, 1993 (A93-46476).
17. Barger, R. L. and Moitra, A. "On Minimizing the Number of Calculations in Design-By-Analysis Codes." NASA Technical Paper 2706, June 1987.
18. Molvik, G. A. and Merkle, C. L. "A Set of Strongly Coupled, Upwind Algorithms for Computing Flows in Chemical Nonequilibrium." AIAA-89-0199, January 1989.
19. Tolle, R. "A New Optimum Design Code for Hypersonic Nozzles, Utilizing Response Surface Methodology." AIAA-97-0519, AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, 1997.
20. Borggaard, J. and Burns, J. "A Sensitivity Equation Approach to Optimal Design of Nozzles." AIAA-94-4274-C, 1994.
21. Borggaard, J. and Burns, J. "A Sensitivity Equation Approach to Shape Optimization in Fluid Flows." NASA Contractor Report 191598, ICASE Report No. 94-8, January 1994.
22. Nadarajah, S. and Jameson, A. "A Comparison of the Continuous and Discrete Adjoint Approach to Automatic Aerodynamic Optimization." AIAA-2000-0667, 38th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 10-13, 2000.
23. Xie, L. "Gradient-Based Optimum Aerodynamic Design Using Adjoint Methods." Ph.D. dissertation, Virginia Polytechnic Institute and State University, Blacksburg, VA, April 12, 2002.
24. Korte, J. J. and Hodge, J. S. "Flow Quality of Hypersonic Wind-Tunnel Nozzles Designed Using Computational Fluid Dynamics." *Journal of Spacecraft and Rockets*, Vol. 32, No. 4, July-August 1995, pp. 569-580.
25. Korte, J. J., Redlund, E., and Anandakrishnan, S. "A Comparison of Experimental Data with CFD for the NSWC Hypervelocity Wind Tunnel #9 Mach 14 Nozzle." AIAA-92-4010, AIAA 17th Aerospace Ground Testing Conference, Nashville TN, July 6-8, 1992.
26. Candler, G. V. "Hypersonic Nozzle Analysis Using an Excluded Volume Equation of State." AIAA-2005-5202, 35th AIAA Fluid Dynamics Conference, Toronto, Ontario, Canada, 6-9 June 2005.
27. Candler, G. V. "APTU Nozzle Code Manual, Version 3.0." 30 September 2004 (included with DPLR software distribution).
28. Scales, L. S. *Introduction to Non-Linear Optimization*. Springer-Verlag New York, Inc., New York, 1985, Chapter 4.

29. Conte, S. D. and de Boor, C. *Elementary Numerical Analysis*. McGraw-Hill Book Company, New York, Second Edition, 1972.
30. Welch, B. W., Jones, K., and Hobbs, J. *Practical Programming in Tcl and Tk*. Prentice Hall PTR, Upper Saddle River, NJ, Fourth Edition, 2003.
31. Gordon, Sanford and McBride, Bonnie J. "Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications, I. Analysis, II. Users Manual and Program Description." NASA Reference Publication 1311, June 1996
32. Ames Research Staff, "Equations, Tables, and Charts for Compressible Flow." NACA Report 1135, 1953.
33. Reynolds, W. C. and Perkins, H. C. *Engineering Thermodynamics*, McGraw-Hill Book Company, New York, Second Edition, 1977.

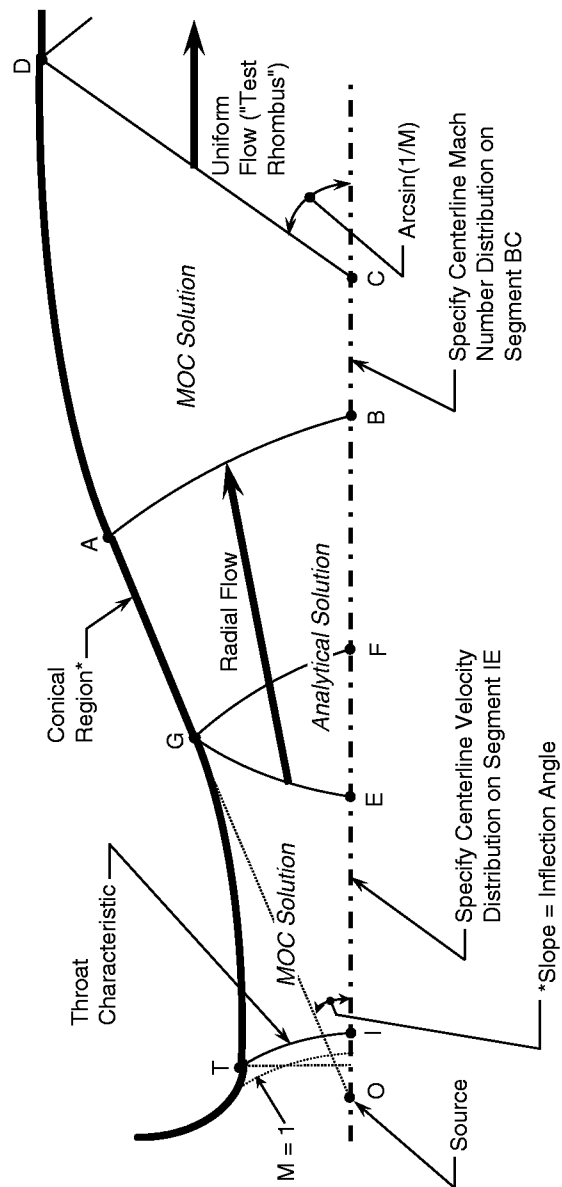


Figure 1. Notation and Design Procedure Used by Sivells

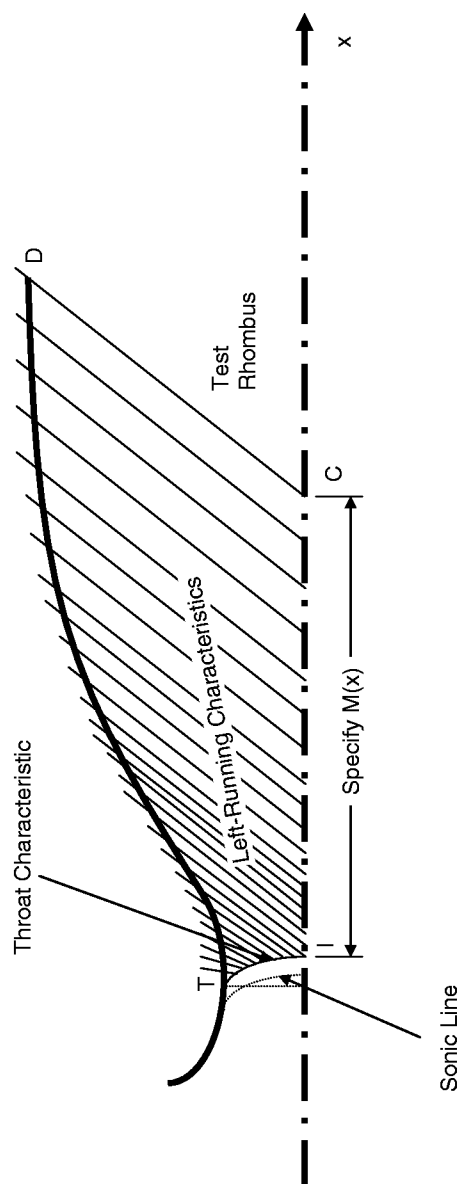


Figure 2. Sivells' Option Chosen for Present Contour Design Problem

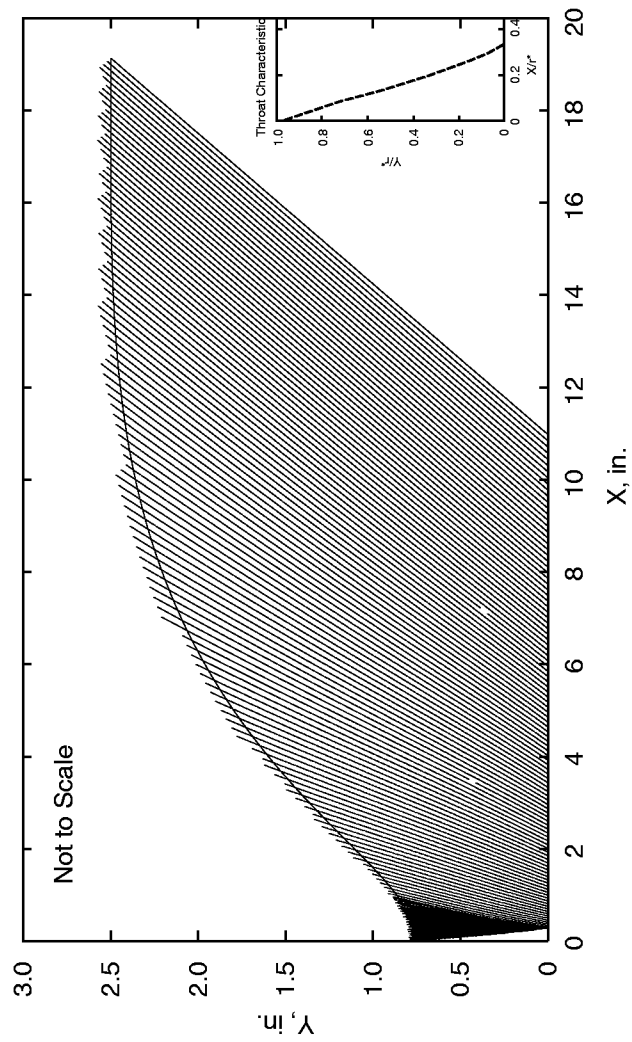


Figure 3. Left-Running Characteristics for an AEDC Arc-Heater Nozzle

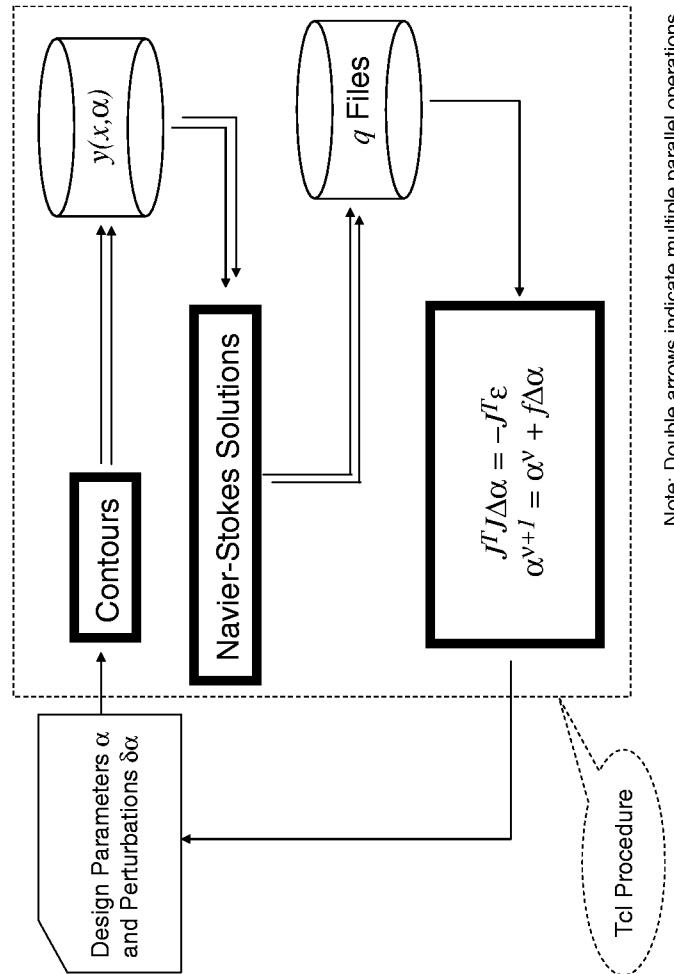
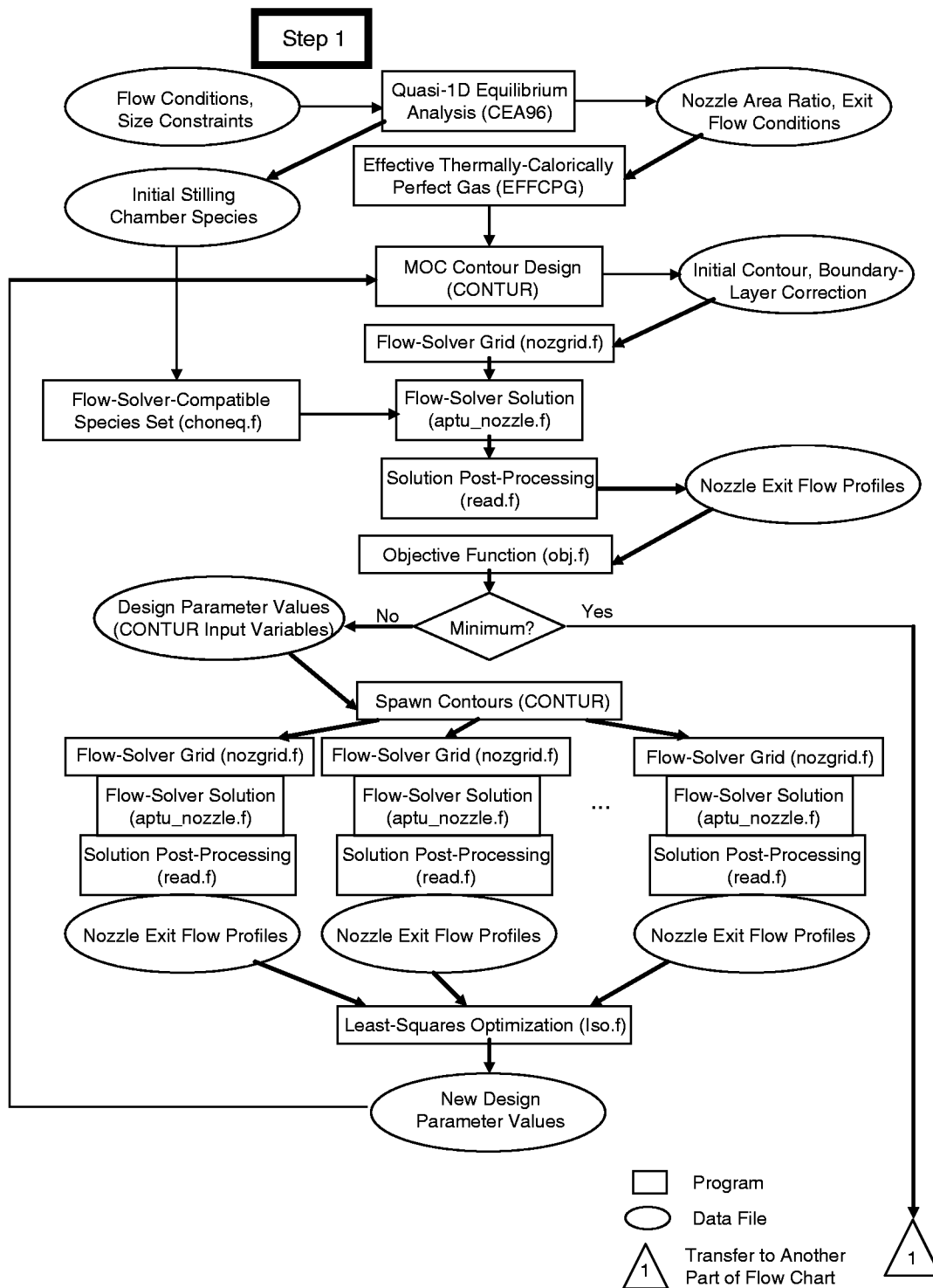
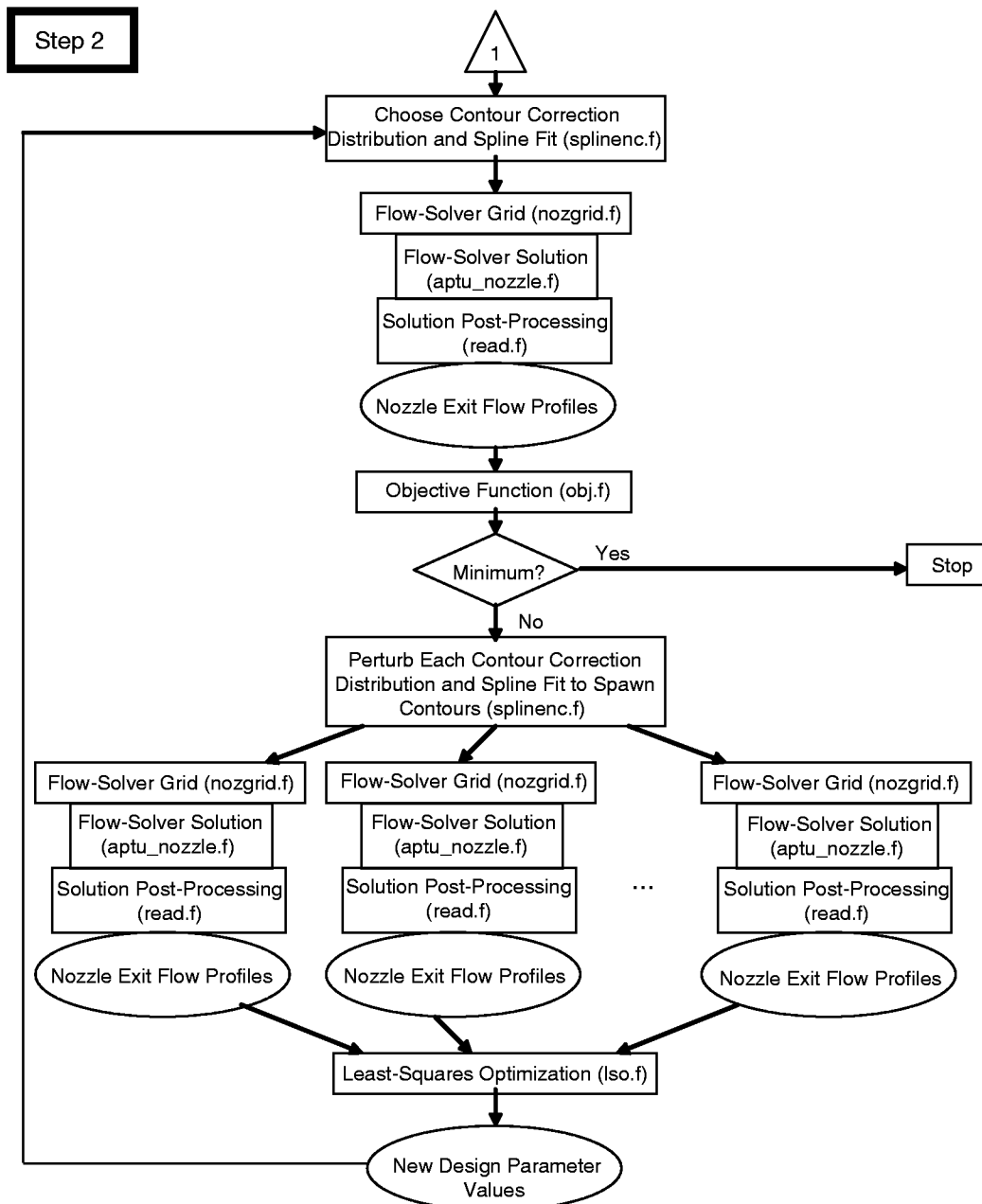


Figure 4. Design Procedure Based on Least-Squares Optimization



a. Step One, Optimization of MOC Contour
Figure 5. Flowchart of Nozzle Contour Design Procedure



b. Step Two, Optimization of Splined Corrections
Figure 5. Concluded

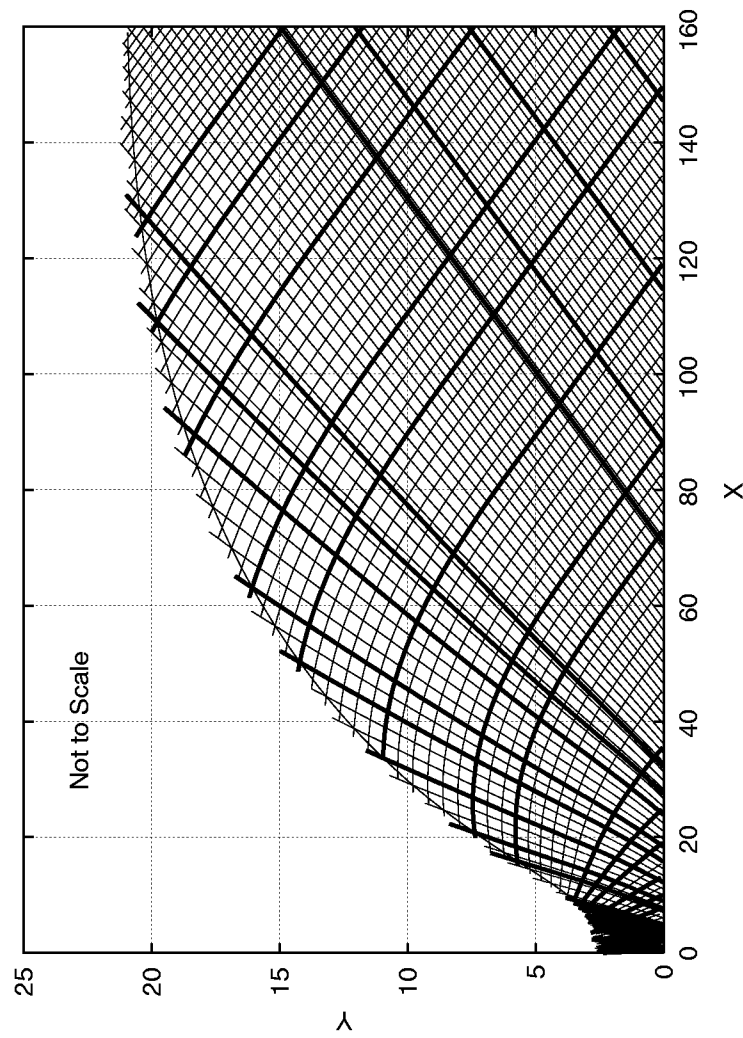


Figure 6. Characteristic Net and Characteristics Traced Upstream from Nozzle Exit

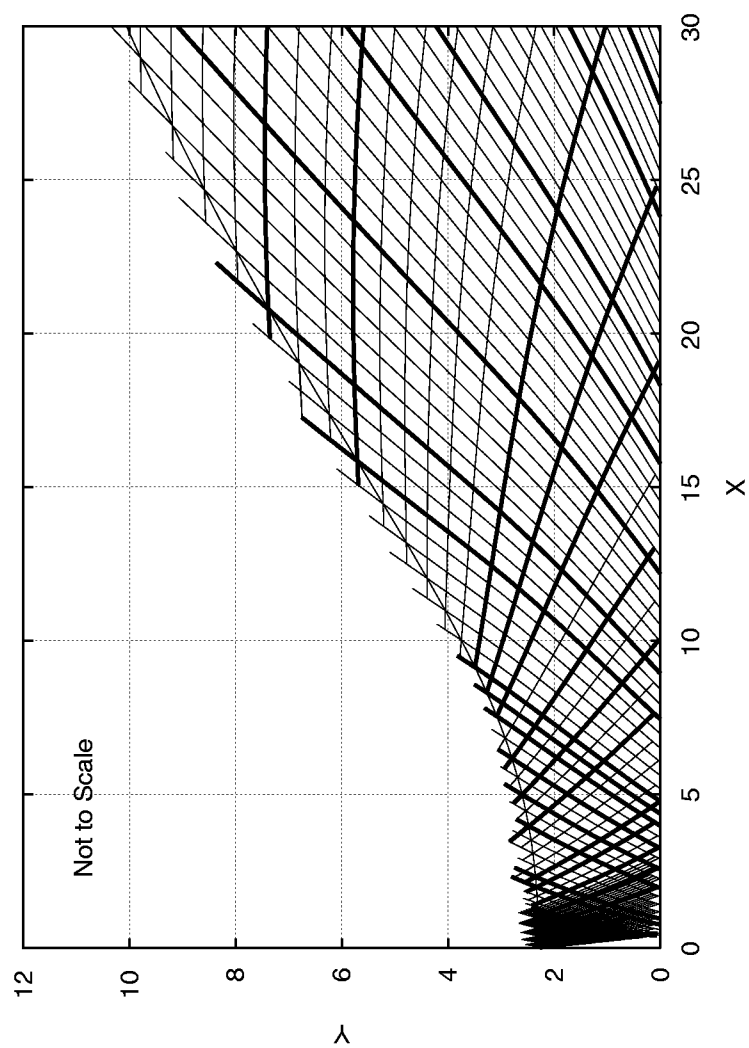


Figure 7. Traced Characteristics in the Inflection Region of Nozzle

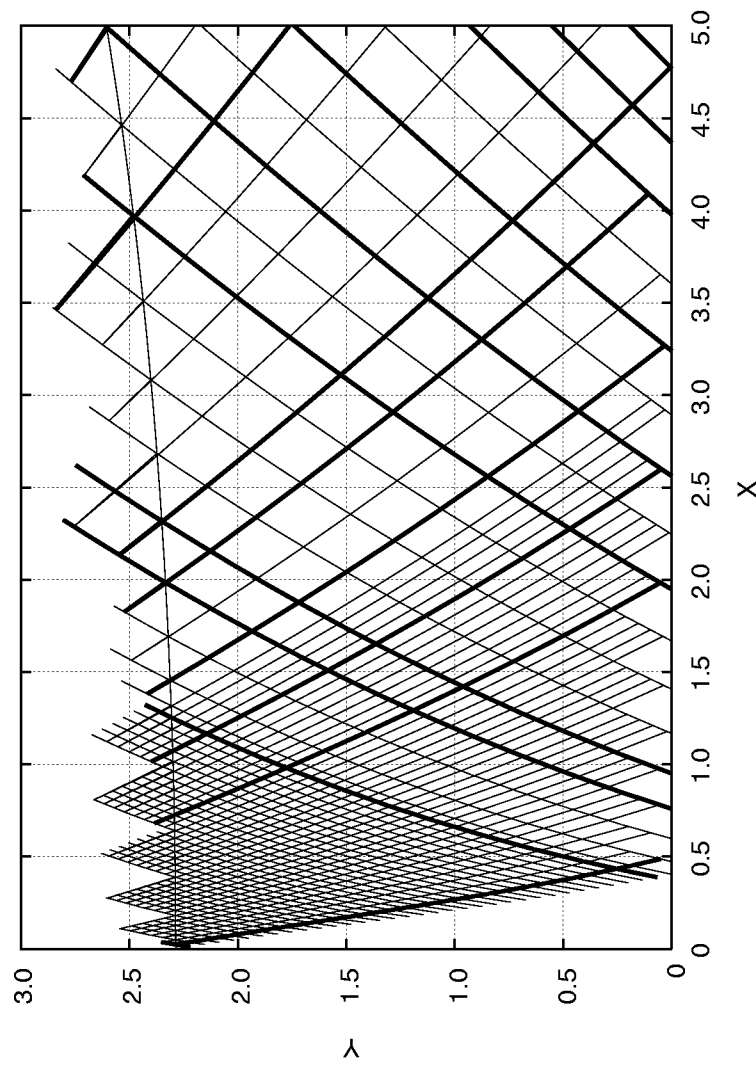


Figure 8. Traced Characteristics in Throat Region

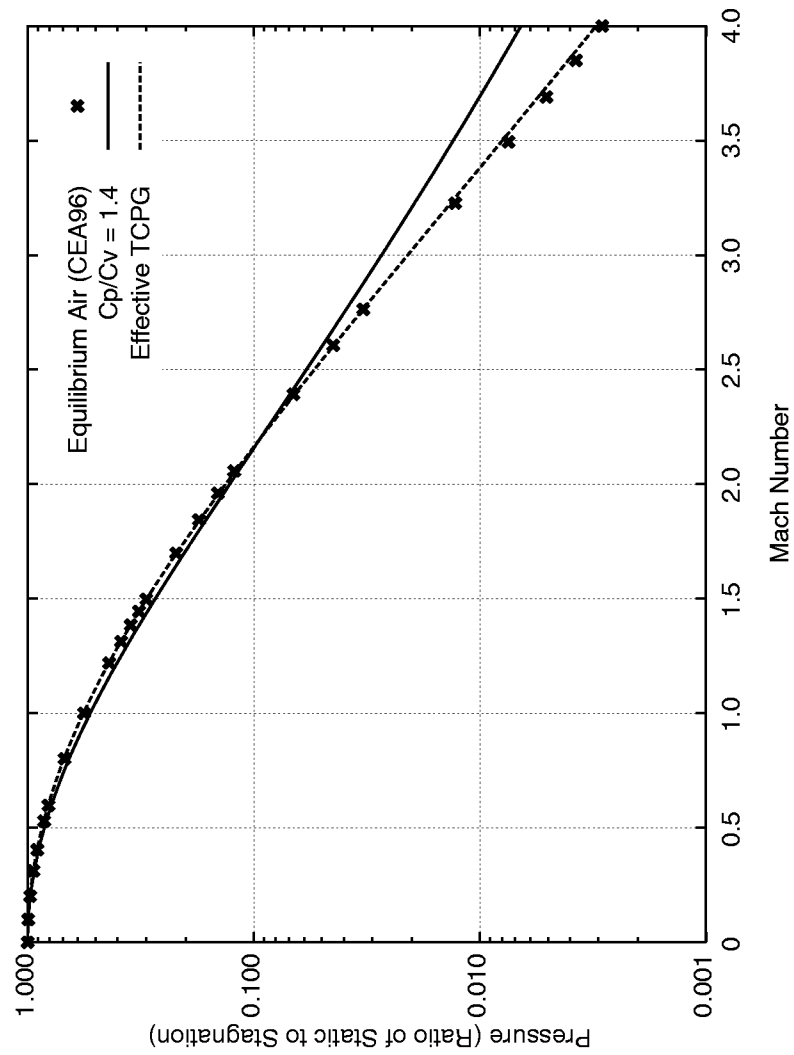


Figure 9. Pressure Distribution for Three Gas Models

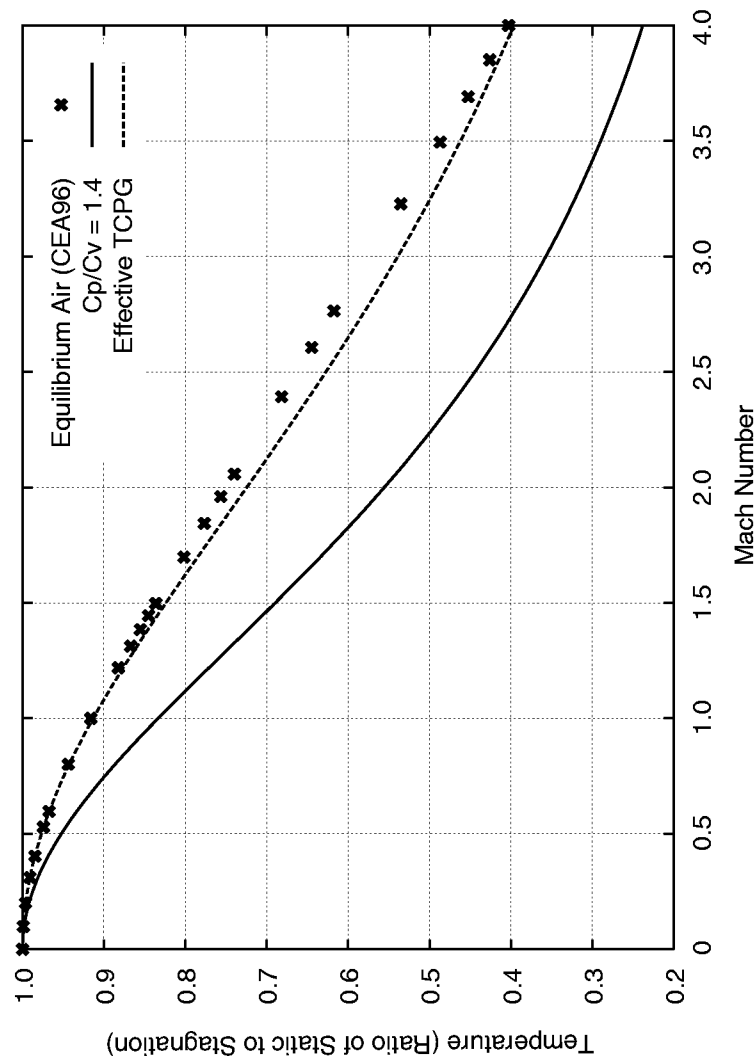


Figure 10. Temperature Distribution for Three Gas Models

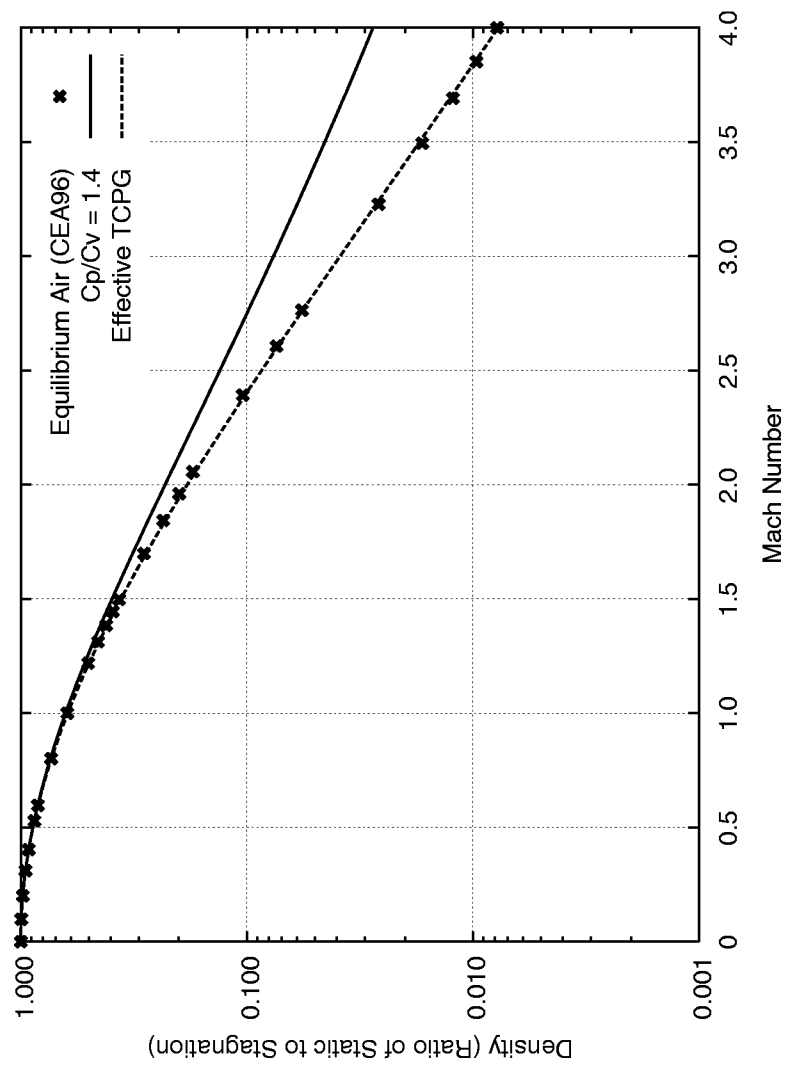


Figure 11. Density Distribution for Three Gas Models

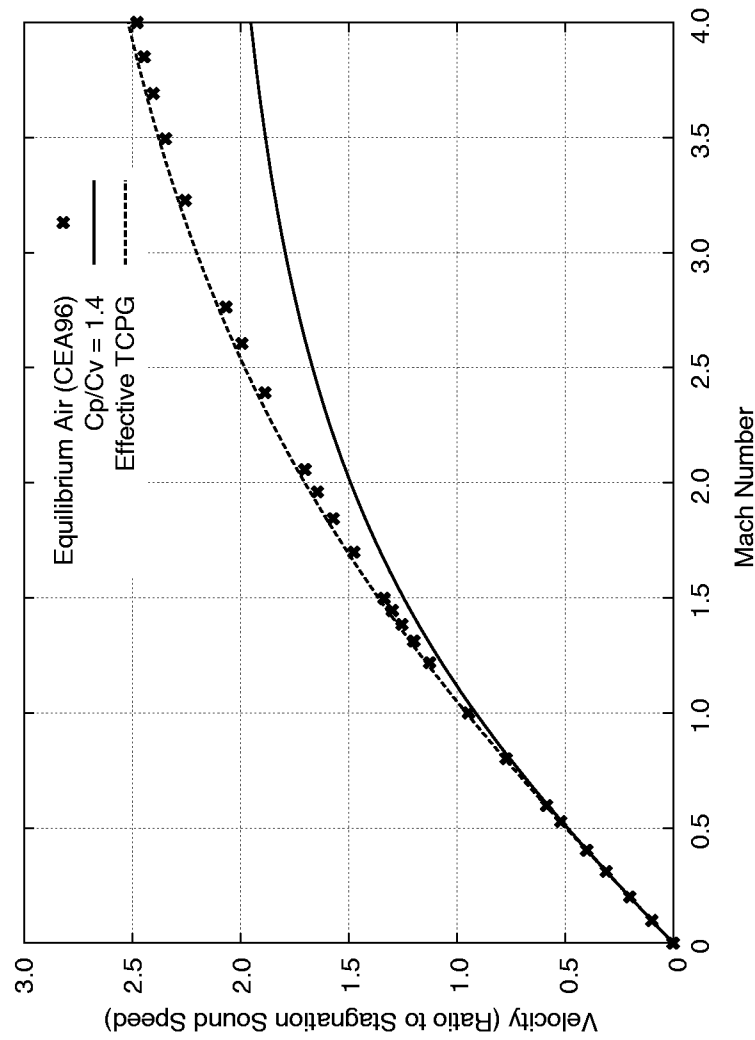


Figure 12. Velocity Distribution for Three Gas Models

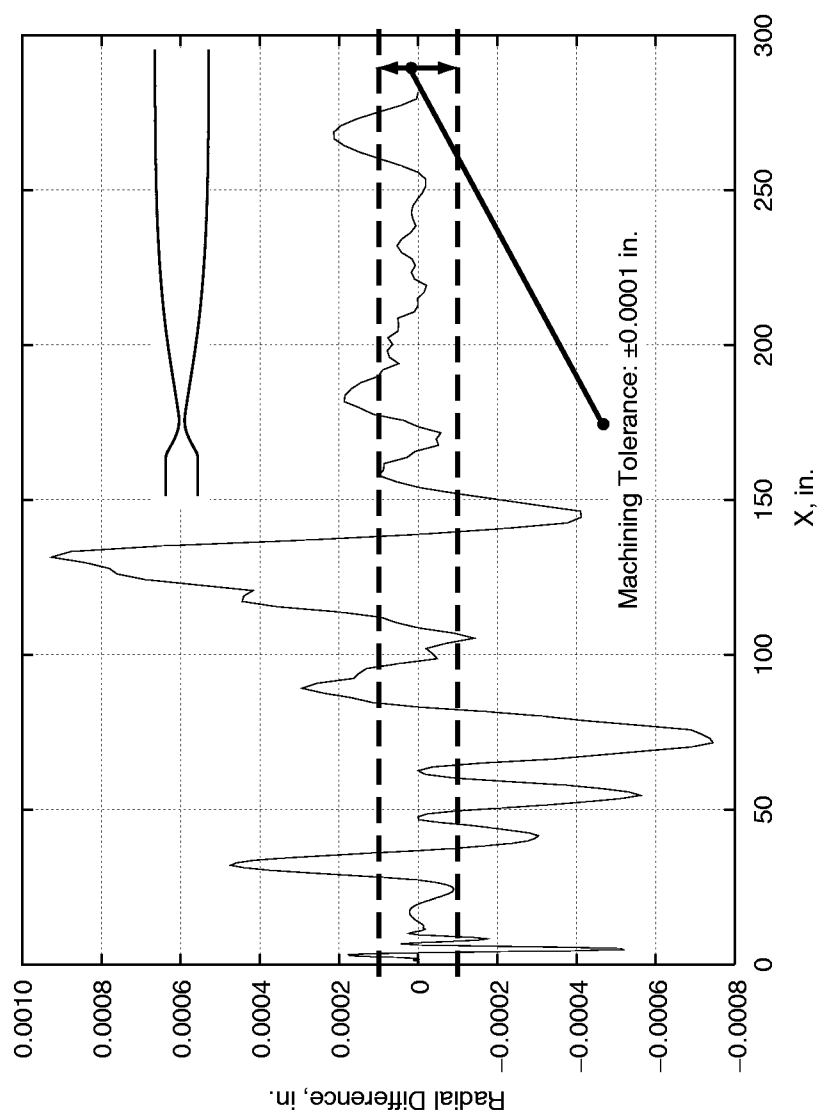


Figure 13. Difference Between Cubic Spline Representation and Original MOC Contour

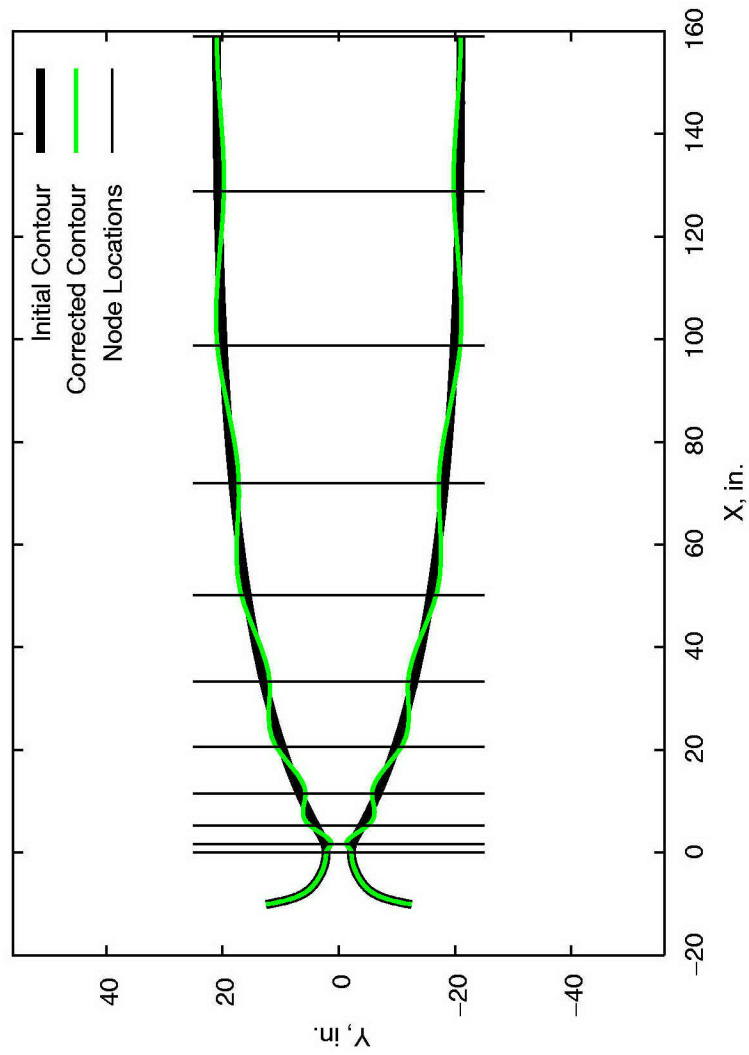


Figure 14. Nozzle Contour with Exaggerated Correction Distribution

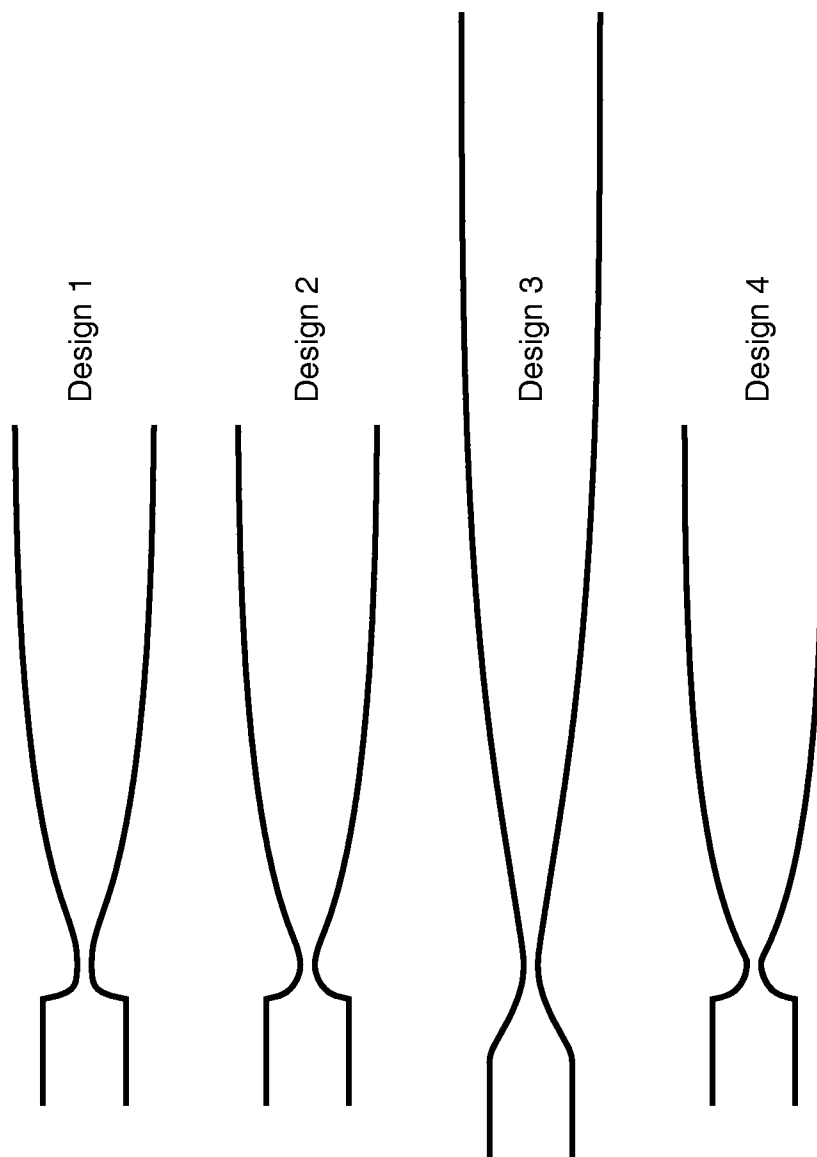


Figure 15. Candidate Viscous MOC Contours for APTU Mach 6

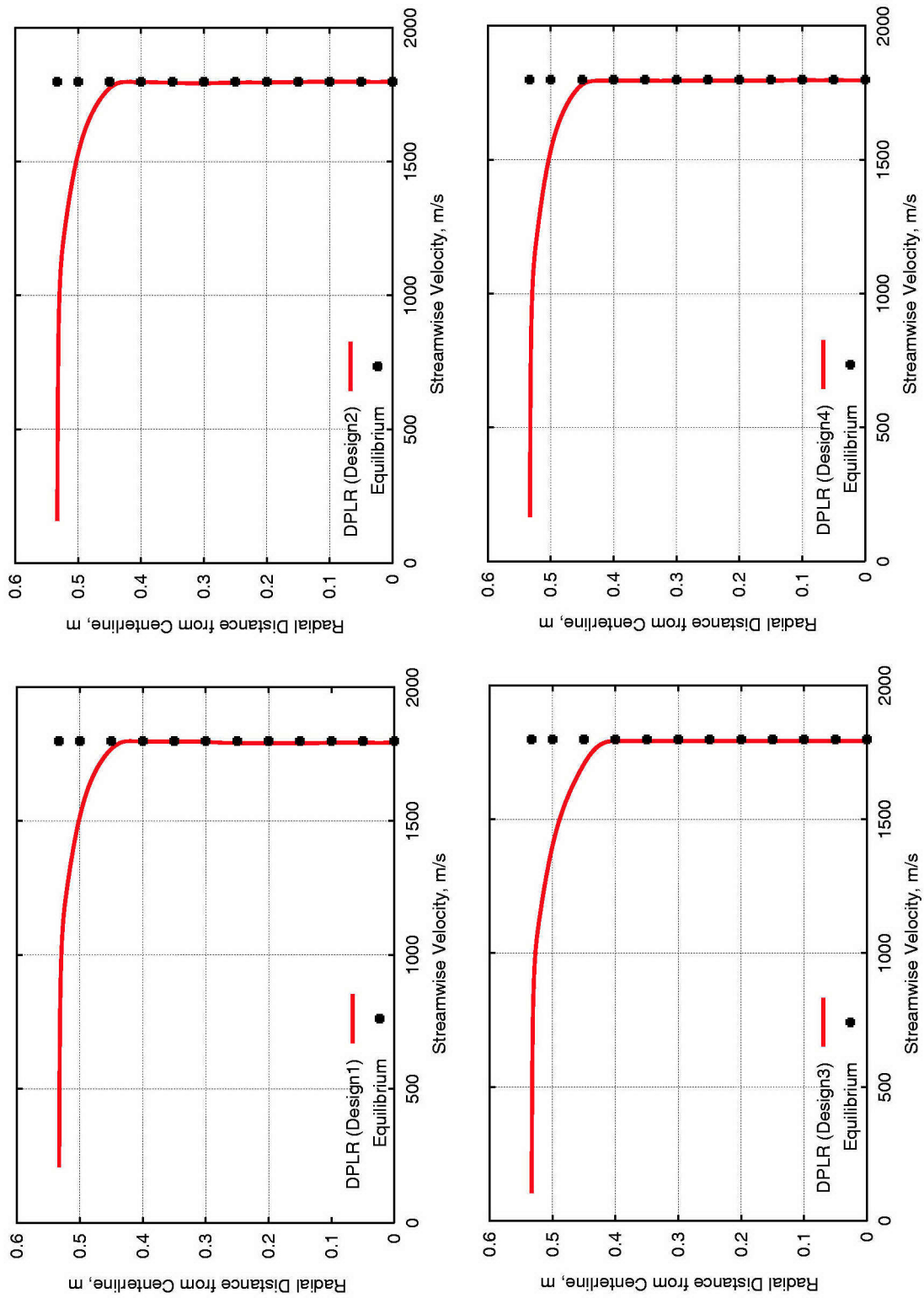


Figure 16. DPLR Exit Axial Velocity Profiles Compared to Equilibrium Values for APTU Mach 6 Candidate Contours

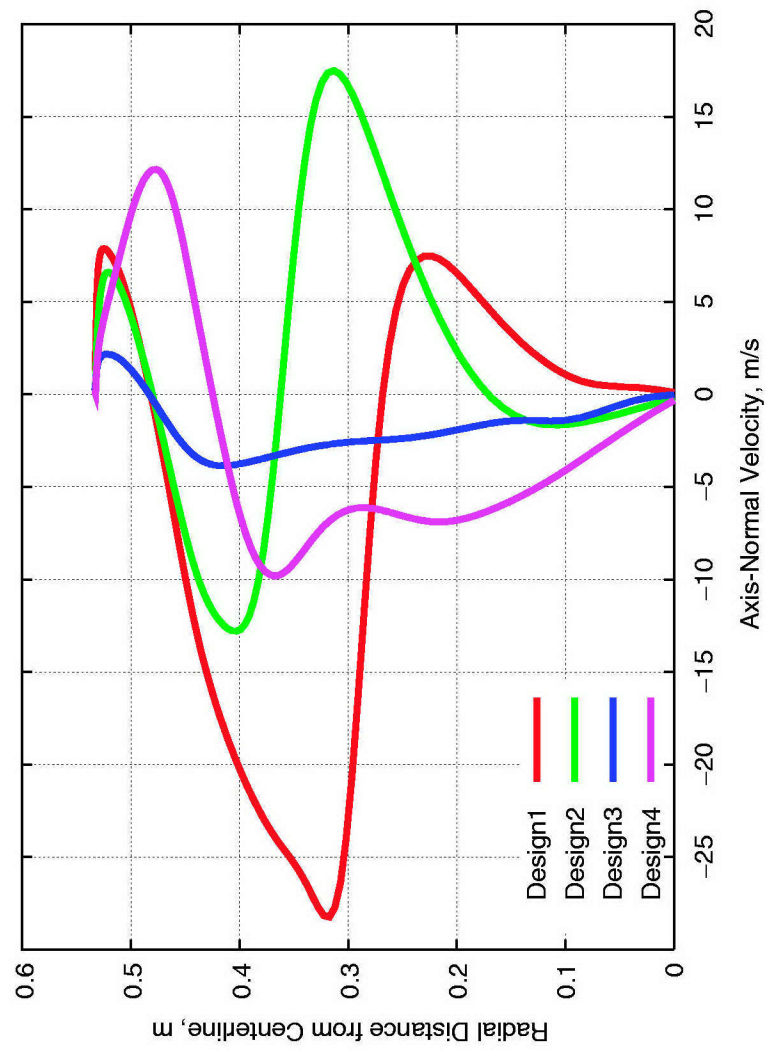


Figure 17. DPLR Exit Axis-Normal Velocity Profiles for APTU Mach 6 Candidate Contours

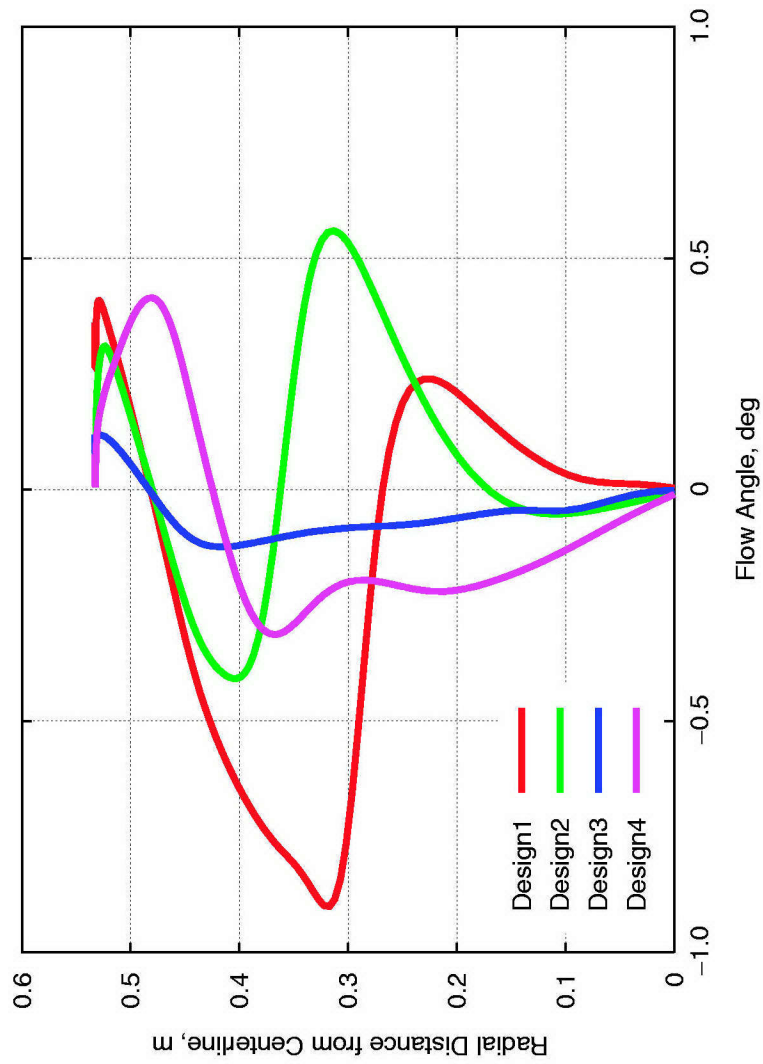


Figure 18. DPLR Exit Flow Angle Profiles for APTU Mach 6 Candidate Contours

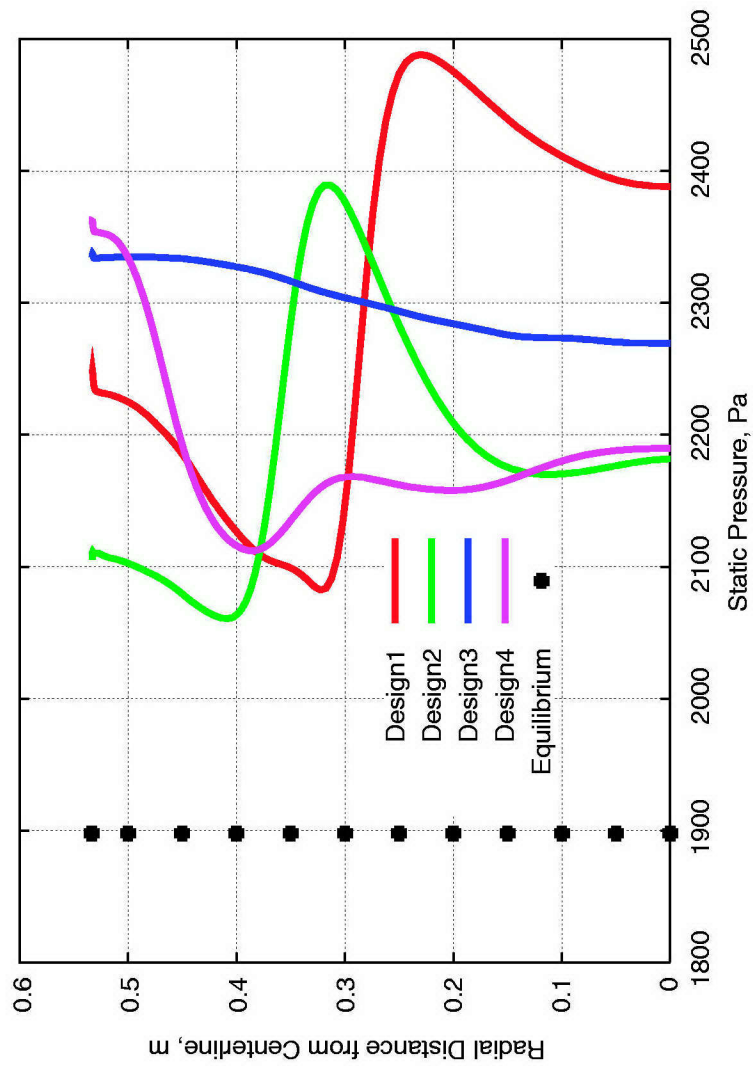


Figure 19. DPLR Exit Static Pressure Profiles Compared to Equilibrium Values for APTU Mach 6 Candidate Contours

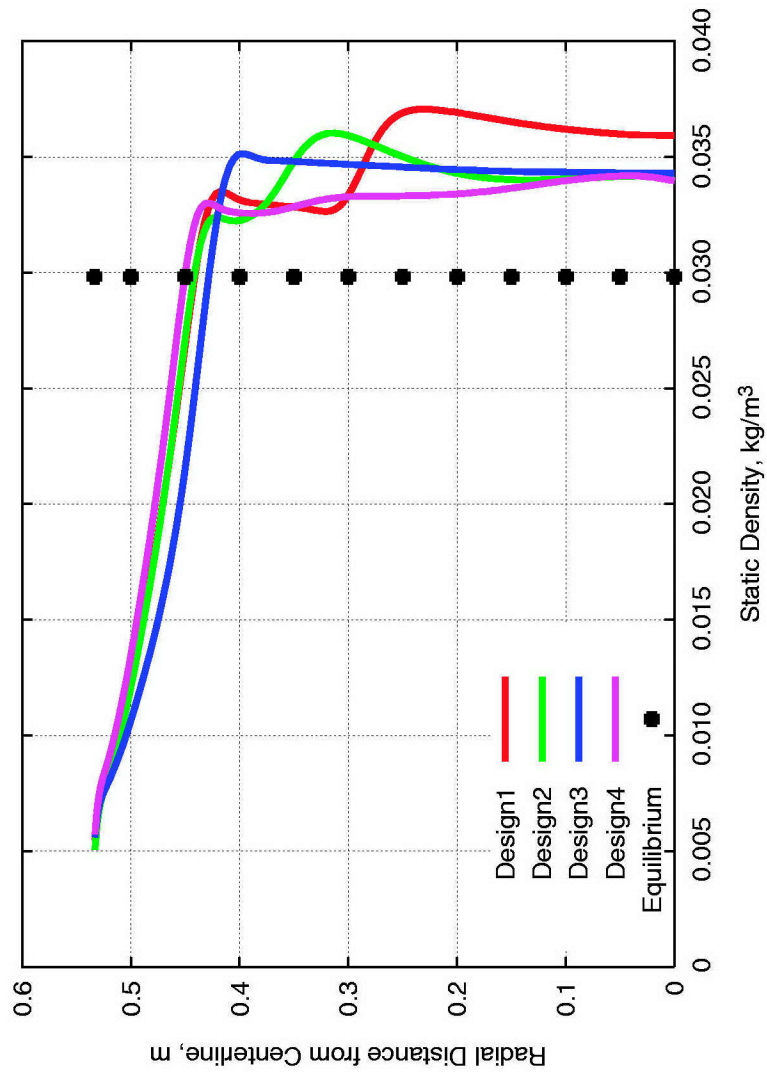


Figure 20. DPLR Exit Static Density Profiles Compared to Equilibrium Values for APTU
Mach 6 Candidate Contours

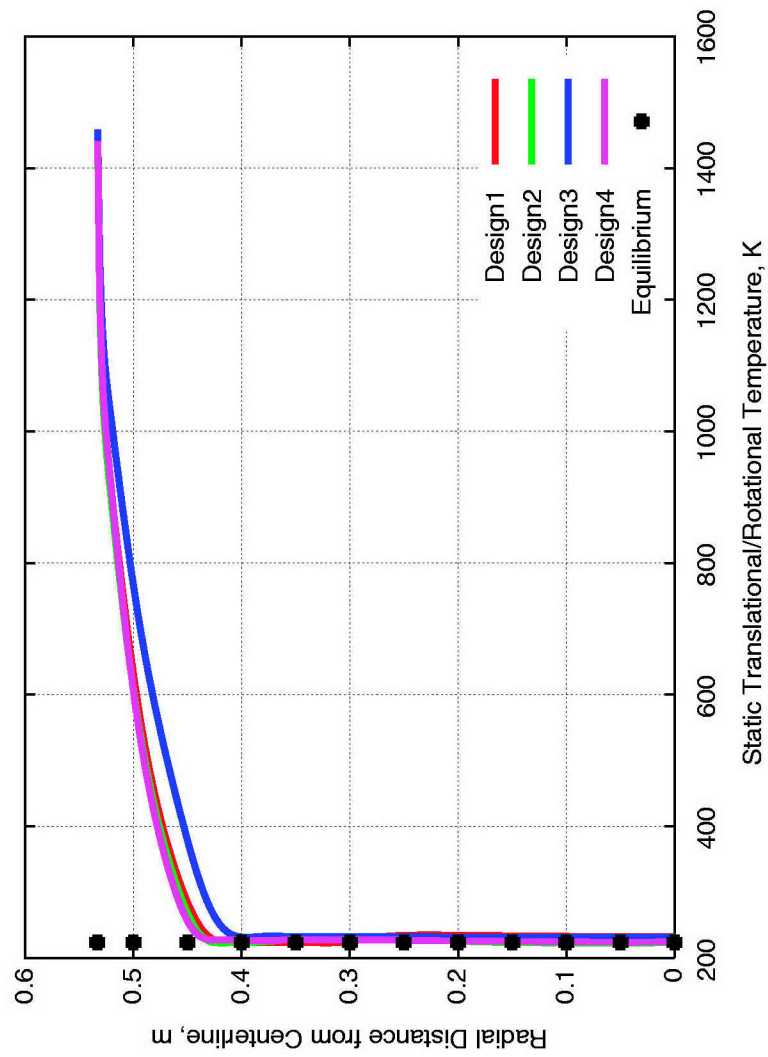


Figure 21. DPLR Exit Static Temperature Profiles Compared to Equilibrium Values for APTU Mach 6 Candidate Contours

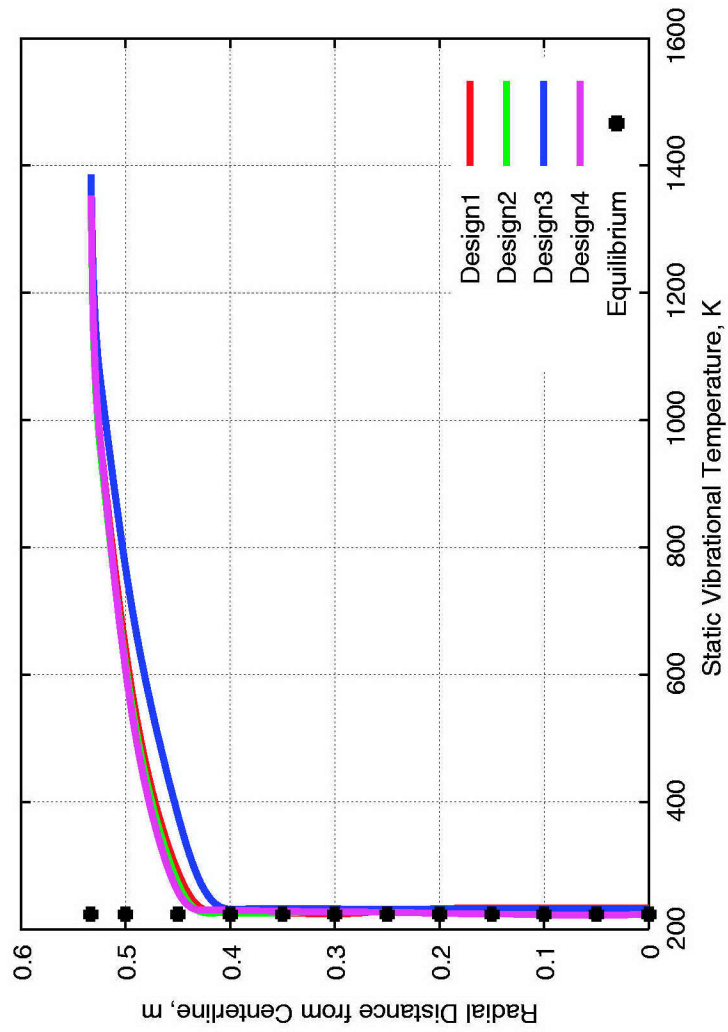


Figure 22. DPLR Exit Static Vibrational Temperature Profiles Compared to Equilibrium Values for APTU Mach 6 Candidate Contours

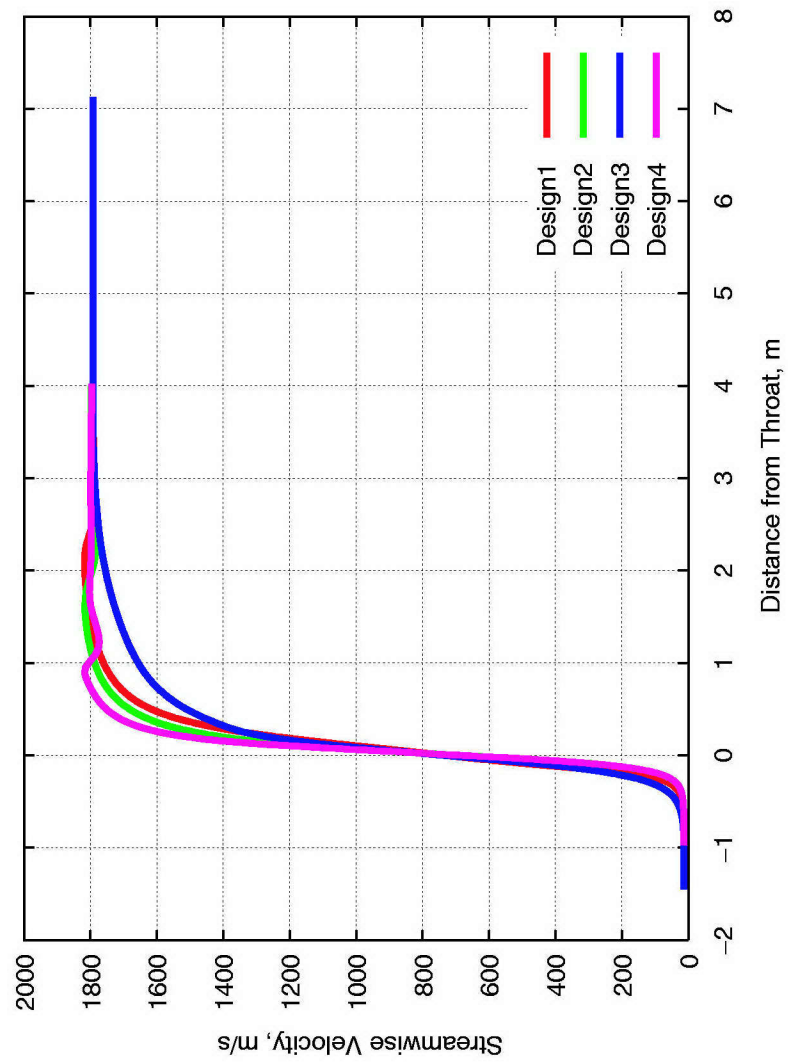


Figure 23. Centerline Streamwise Velocity for APTU Mach 6 Candidate Contours

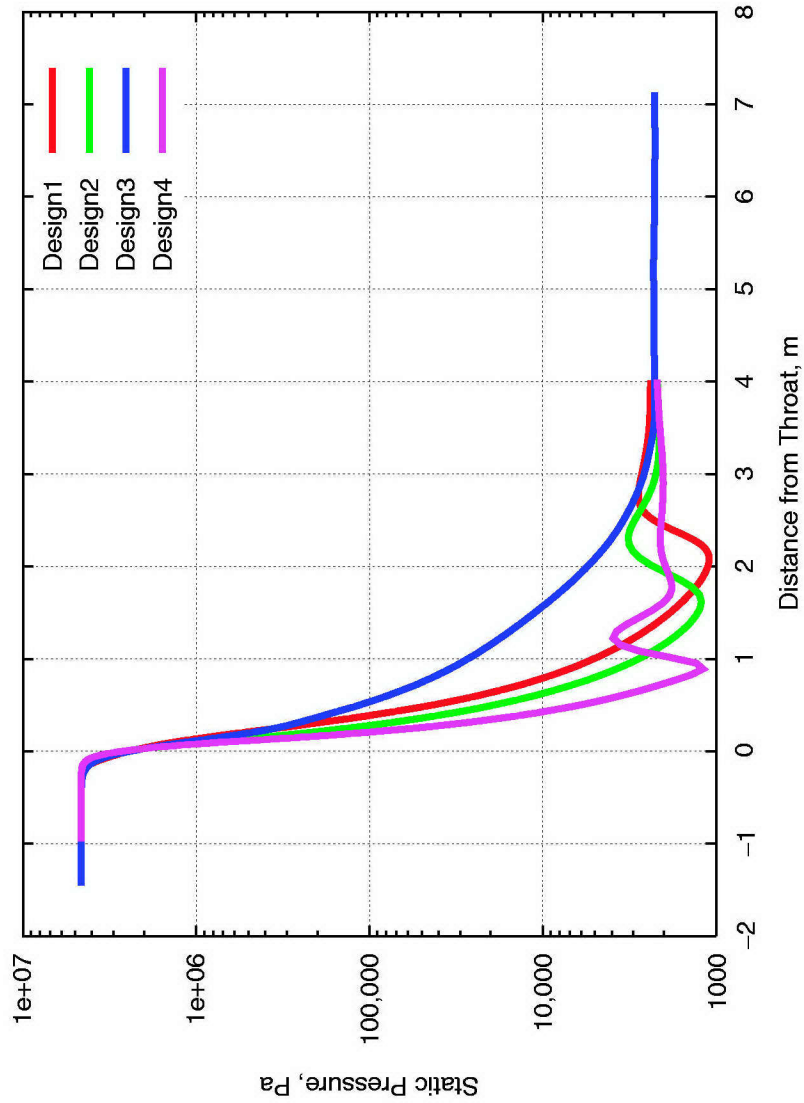


Figure 24. Centerline Static Pressure for APTU Mach 6 Candidate Contours

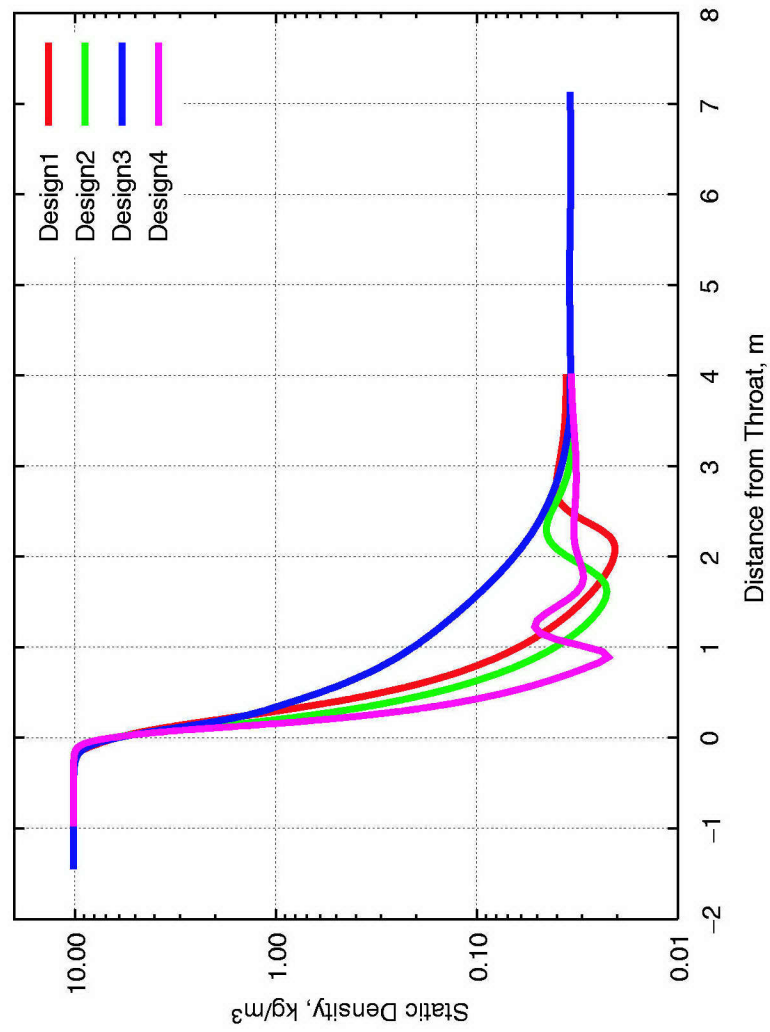


Figure 25. Centerline Static Density for APTU Mach 6 Candidate Contours

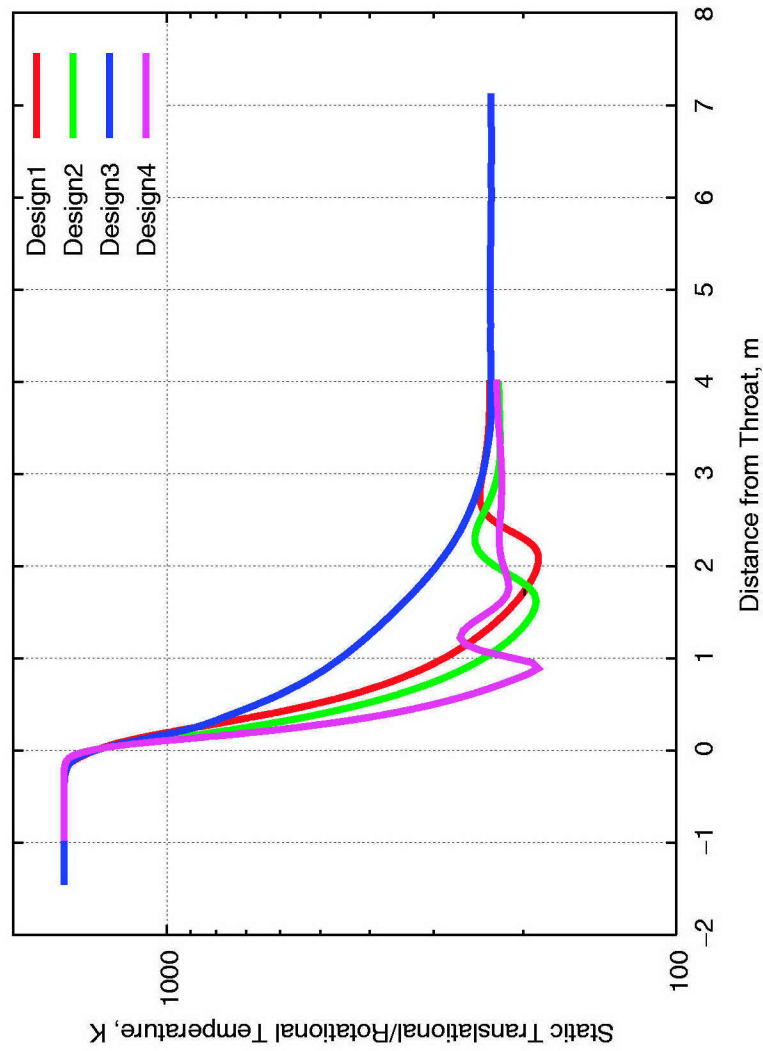


Figure 26. Centerline Static Translational/Rotational Temperature for APTU Mach 6 Candidate Contours

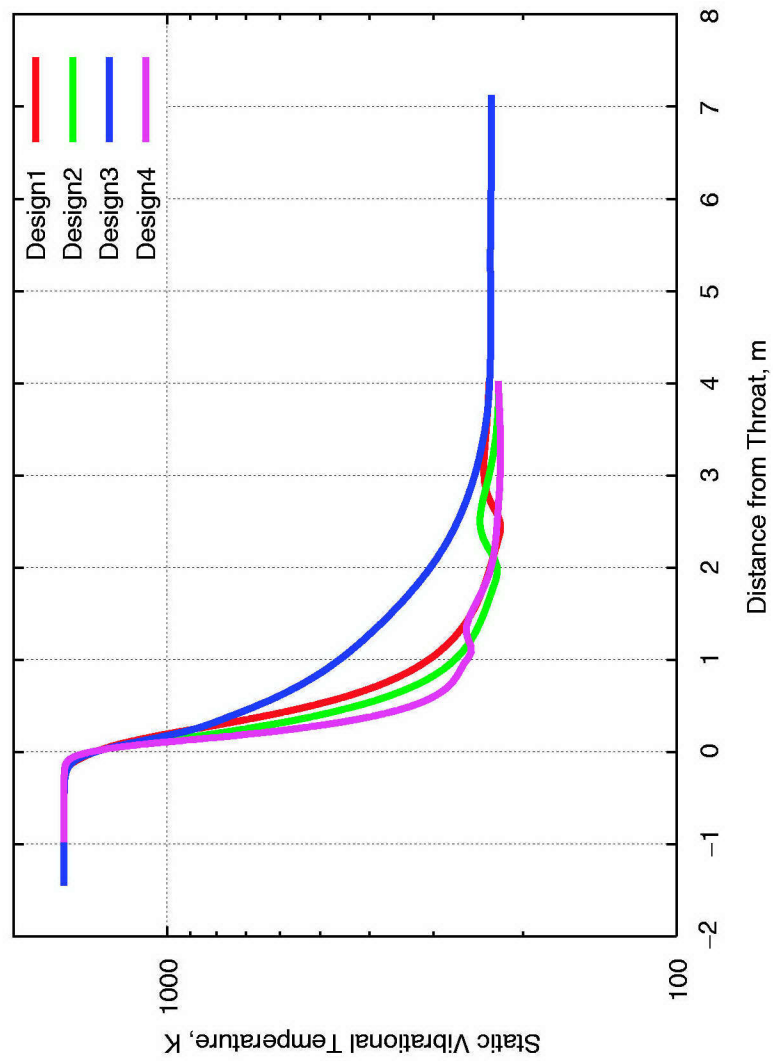


Figure 27. Centerline Static Vibrational Temperature for APTU Mach 6 Candidate Contours

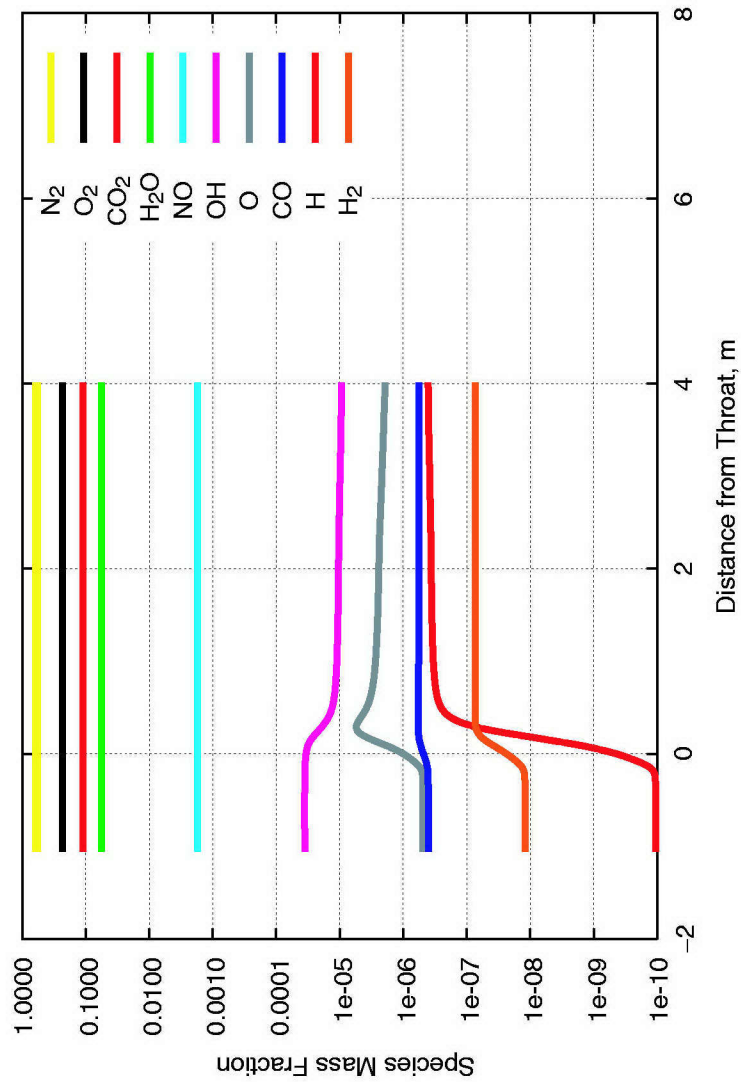


Figure 28. Centerline Species Mass Fraction Distribution for Design I

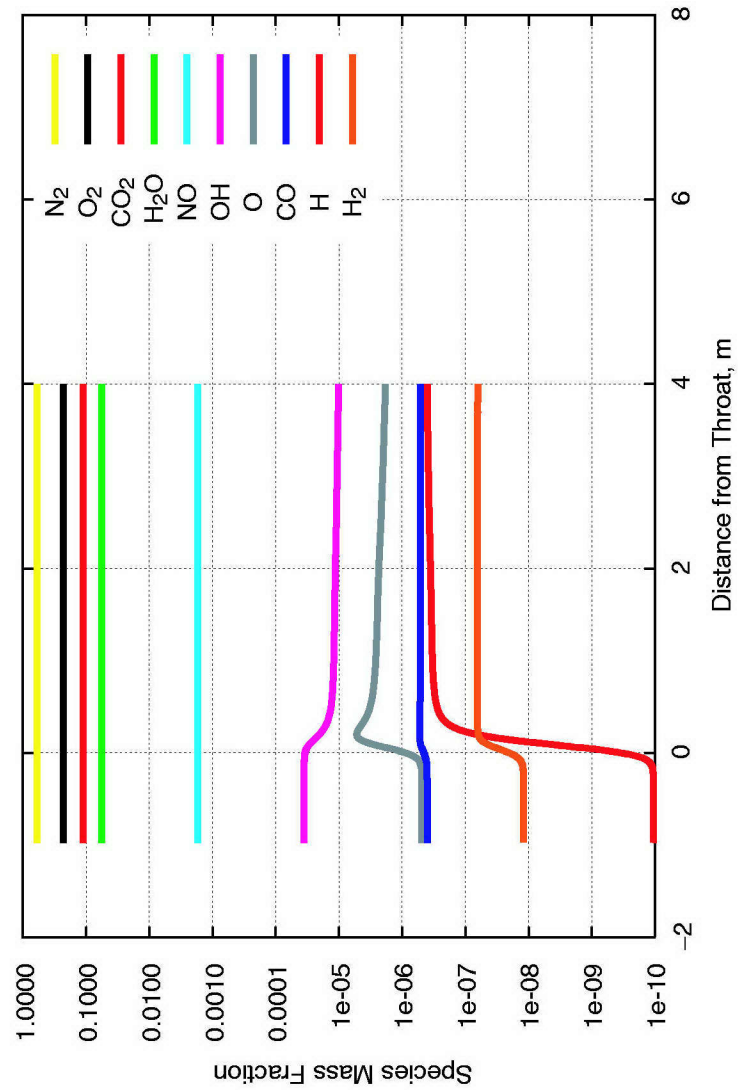


Figure 29. Center line Species Mass Fraction Distribution for Design2

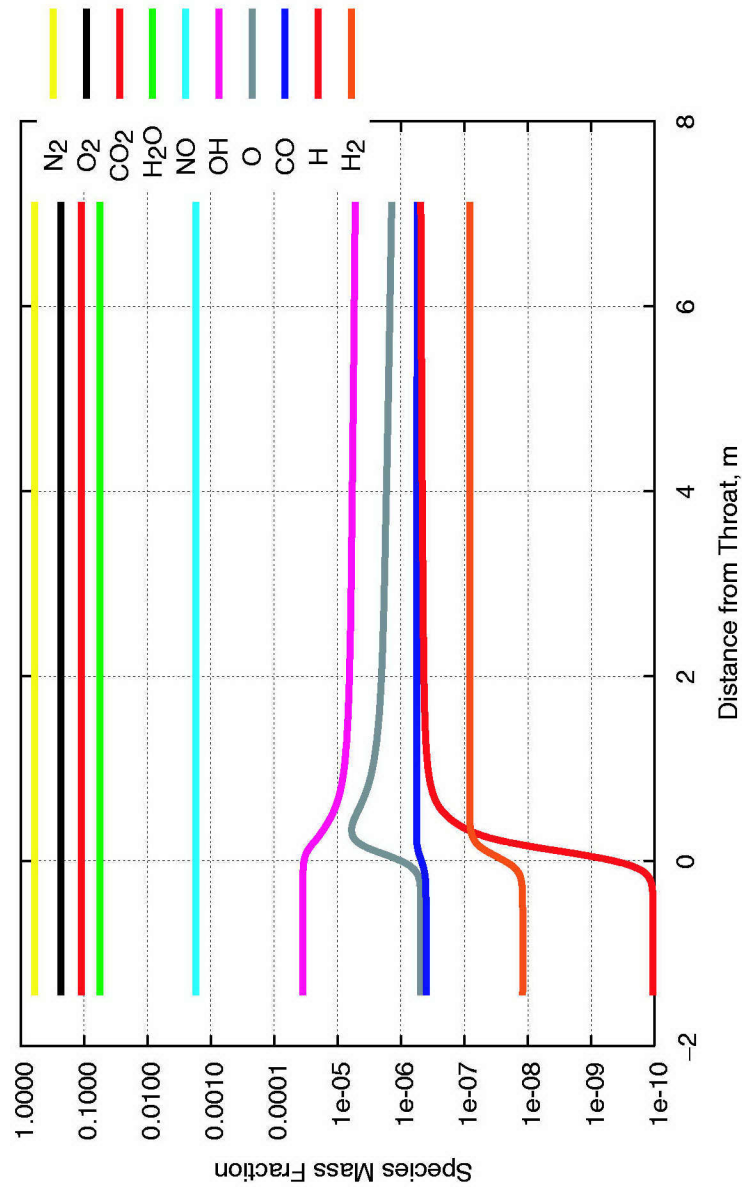


Figure 30. Centerline Species Mass Fraction Distribution for Design 3

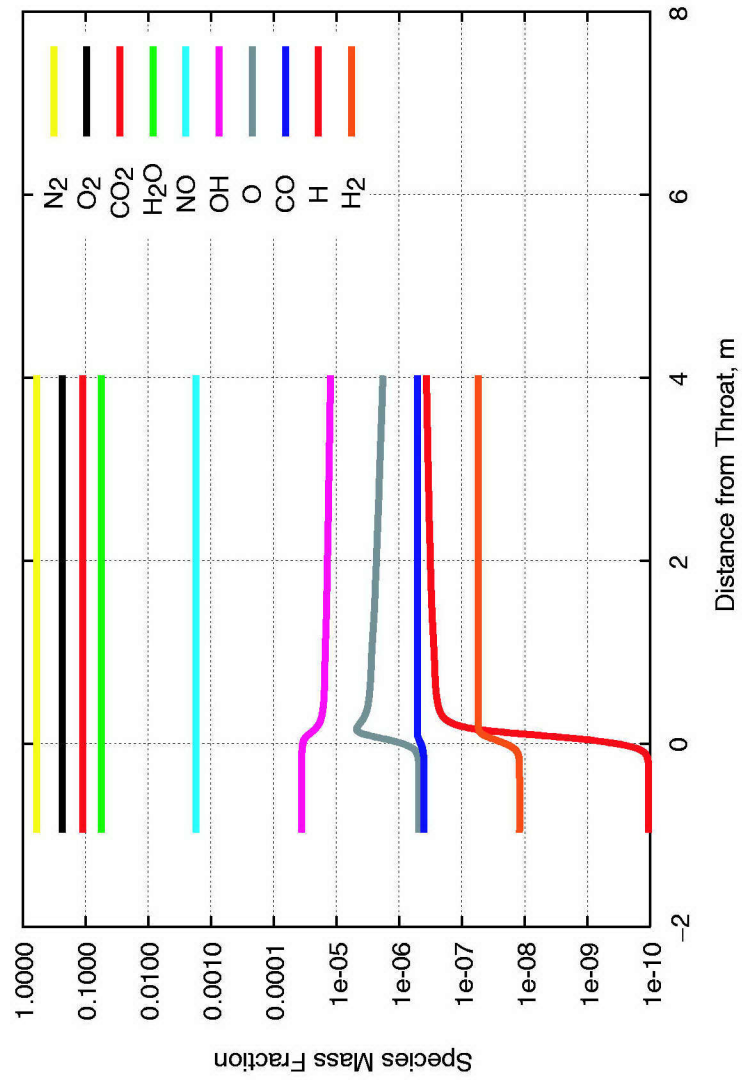


Figure 31. Centerline Species Mass Fraction Distribution for Design 4

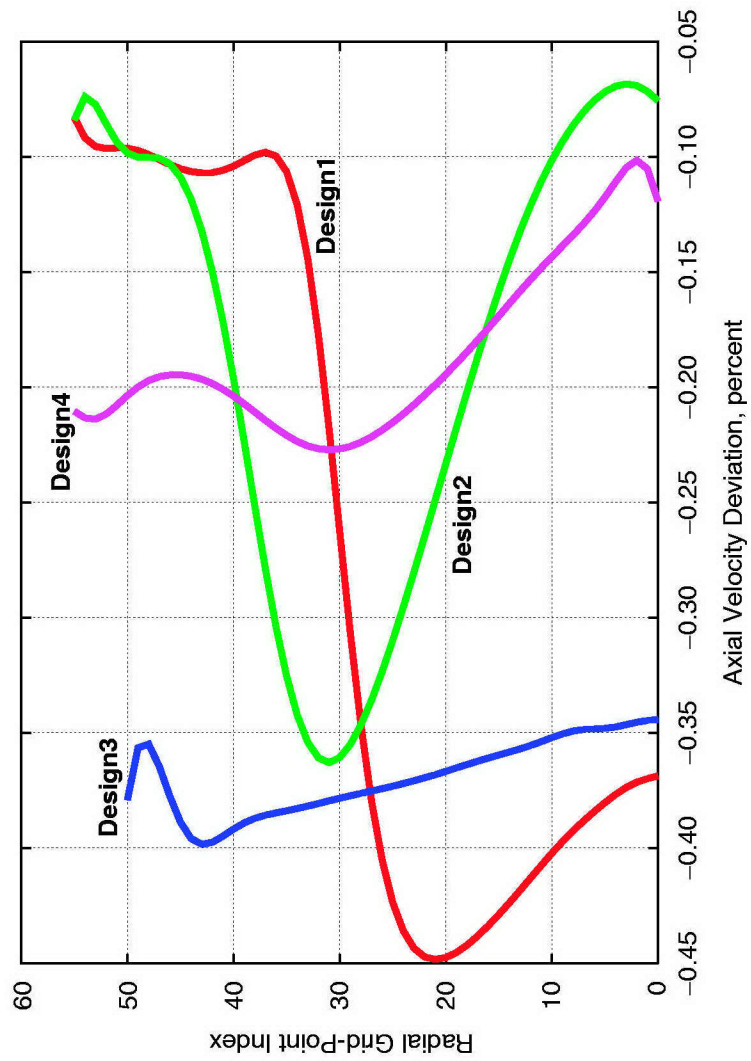


Figure 32. Deviation of Exit Axial Velocity from Equilibrium as a Percent of CEA96 Equilibrium Velocity, Designs 1-4

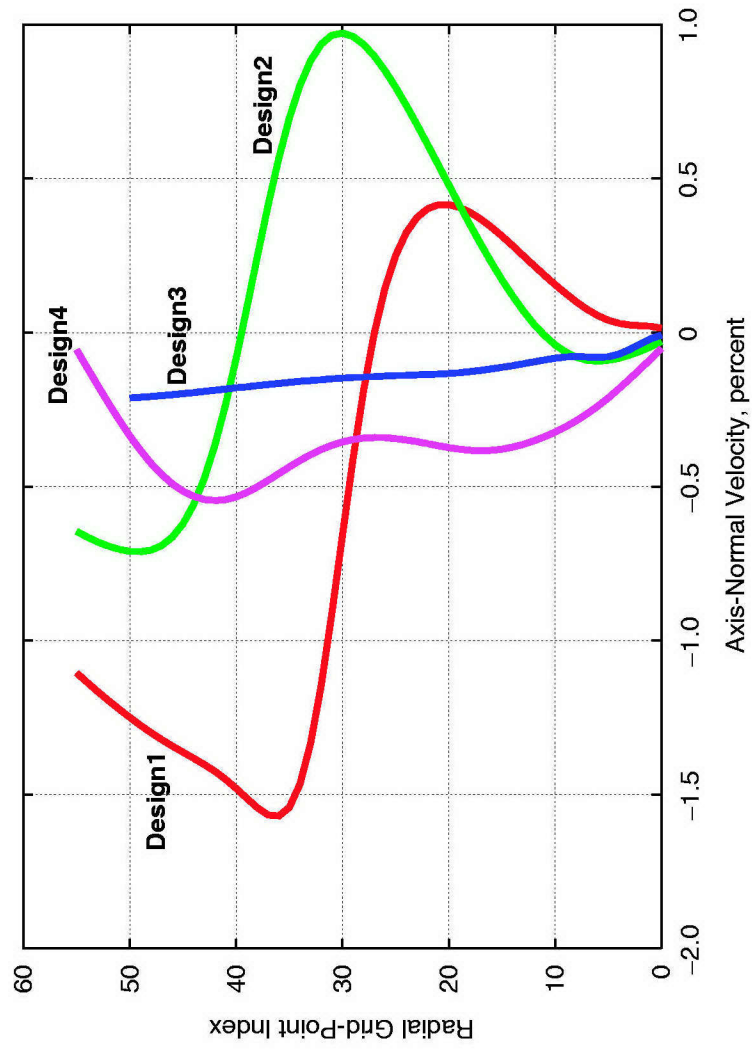


Figure 33. Exit Axis-Normal Velocity as a Percent of CEA96 Equilibrium Velocity, Designs 1-4

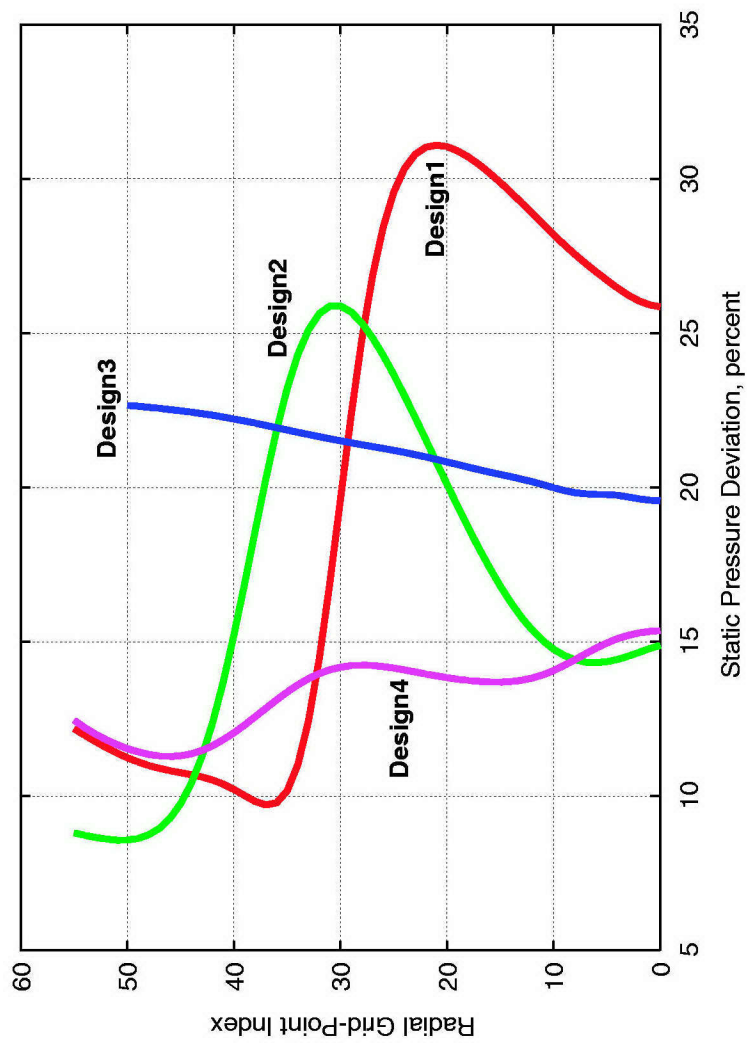


Figure 34. Deviation of Exit Static Pressure from Equilibrium as a Percent of CEA96 Equilibrium Static Pressure, Designs 1-4

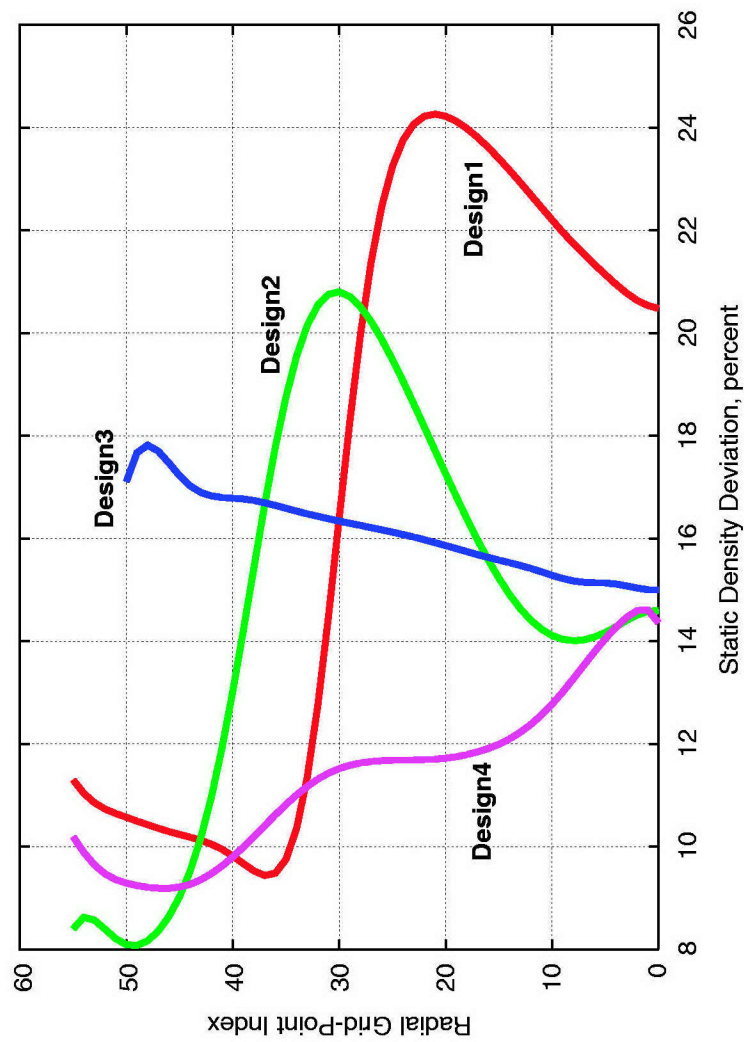


Figure 35. Deviation of Exit Static Density from Equilibrium as a Percent of CEA96 Equilibrium Static Density, Designs1-4

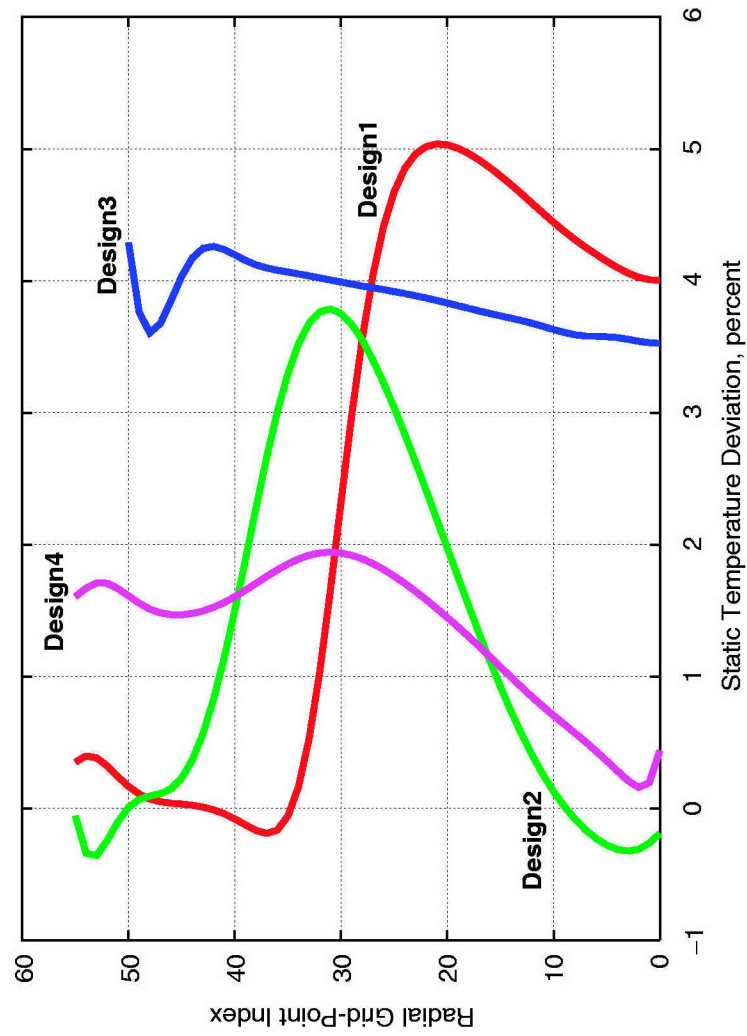


Figure 36. Deviation of Exit Static Temperature from Equilibrium as a Percent of CEA96 Equilibrium Static Temperature, Designs1-4

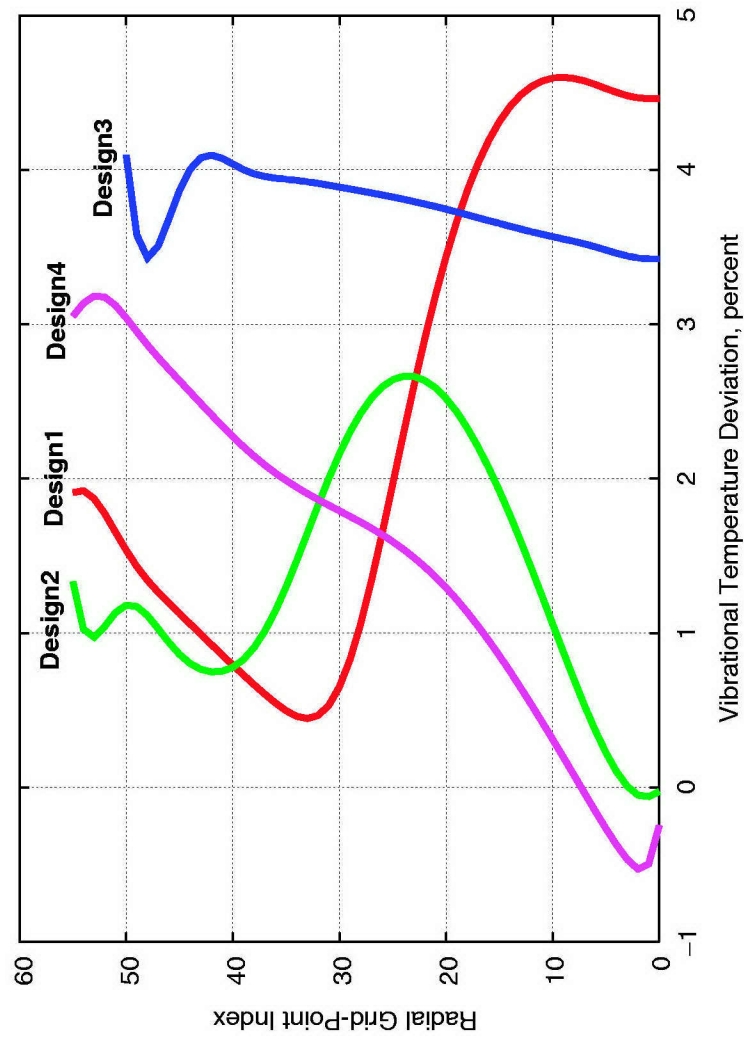


Figure 37. Deviation of Exit Vibrational Temperature from Equilibrium as a Percent of CEA96 Equilibrium Static Temperature, Designs 1-4

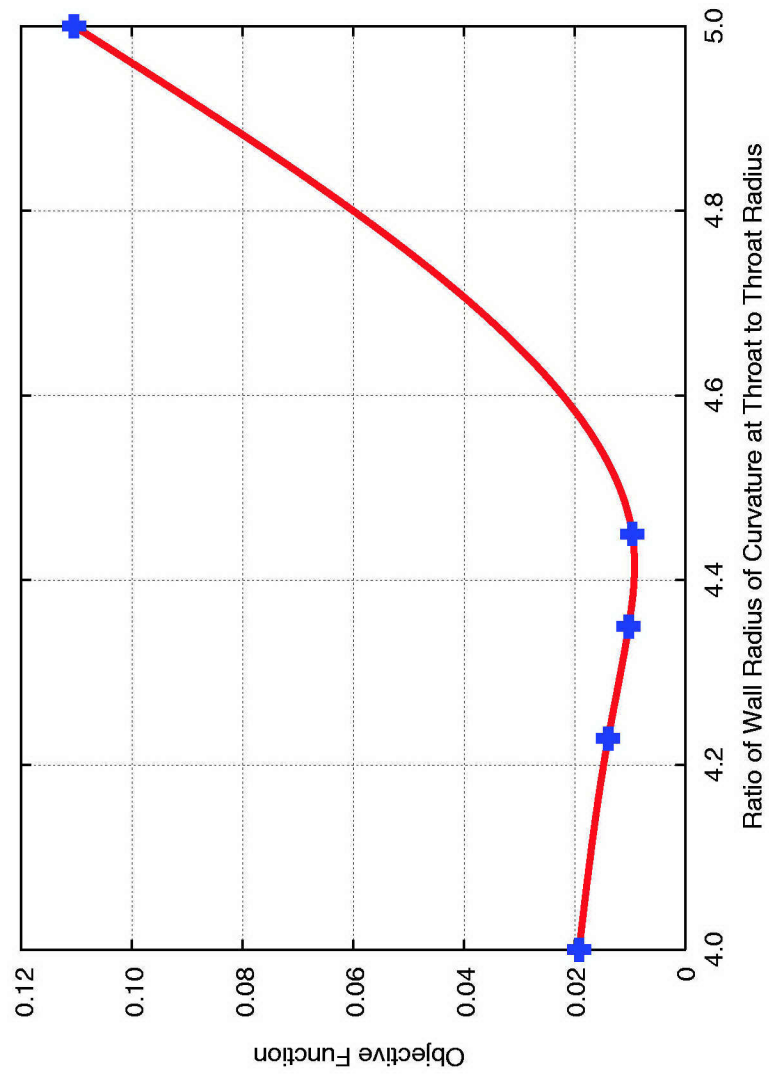


Figure 38. Objective Function vs Design Parameter RC

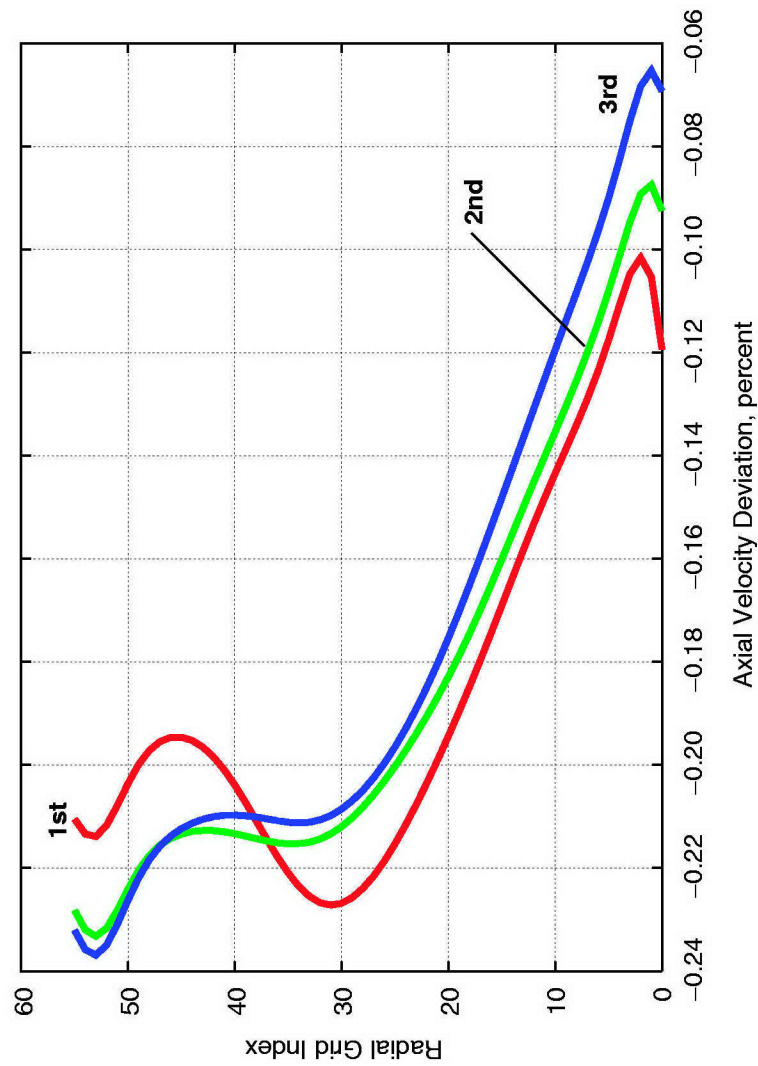


Figure 39. Axial Velocity Component of Objective Function for Three Iterations of MOC Contour Optimization

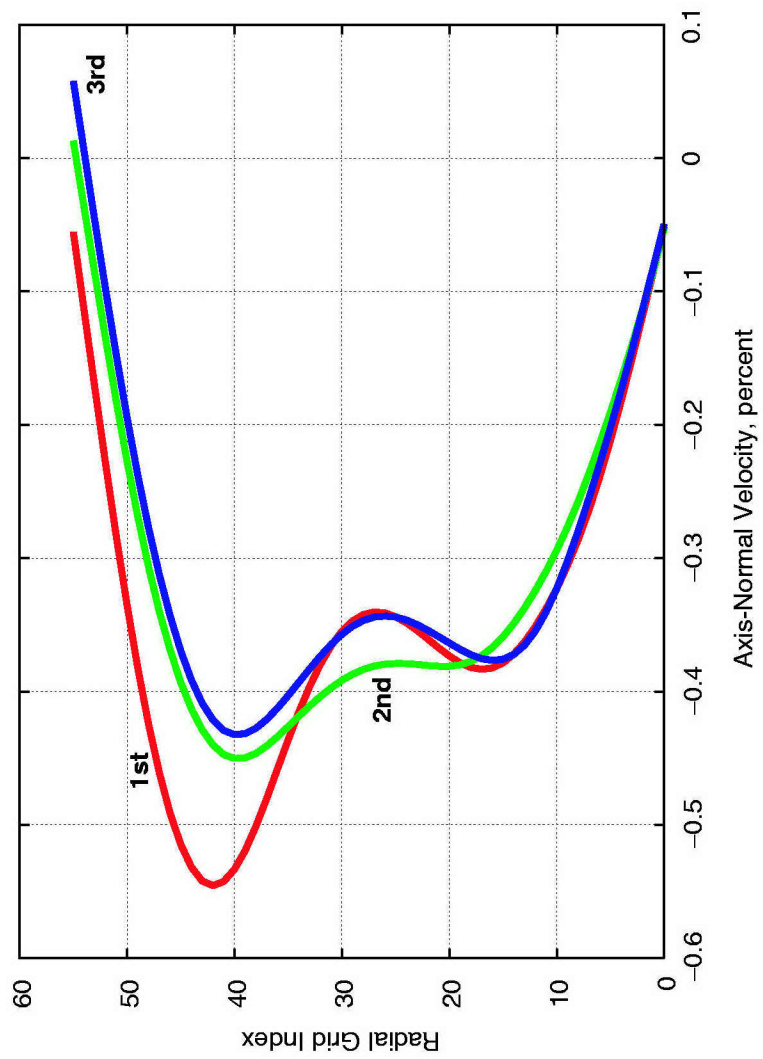


Figure 40. Axis-Normal Velocity Component of Objective Function for Three Iterations of MOC Contour Optimization

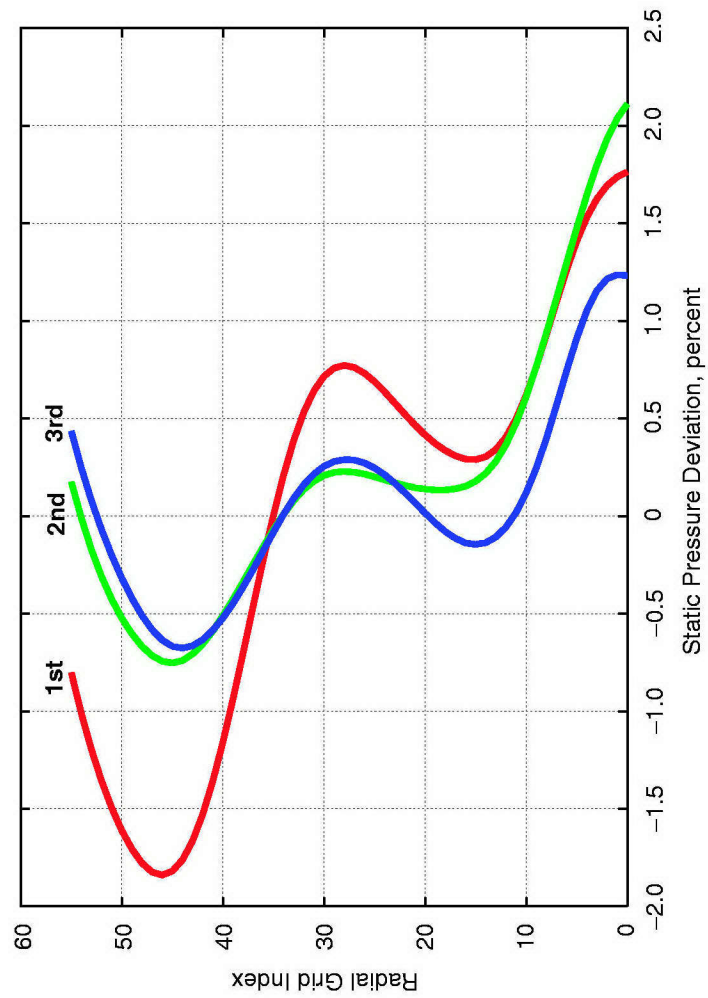


Figure 41. Static Pressure Component of Objective Function for Three Iterations of MOC Contour Optimization

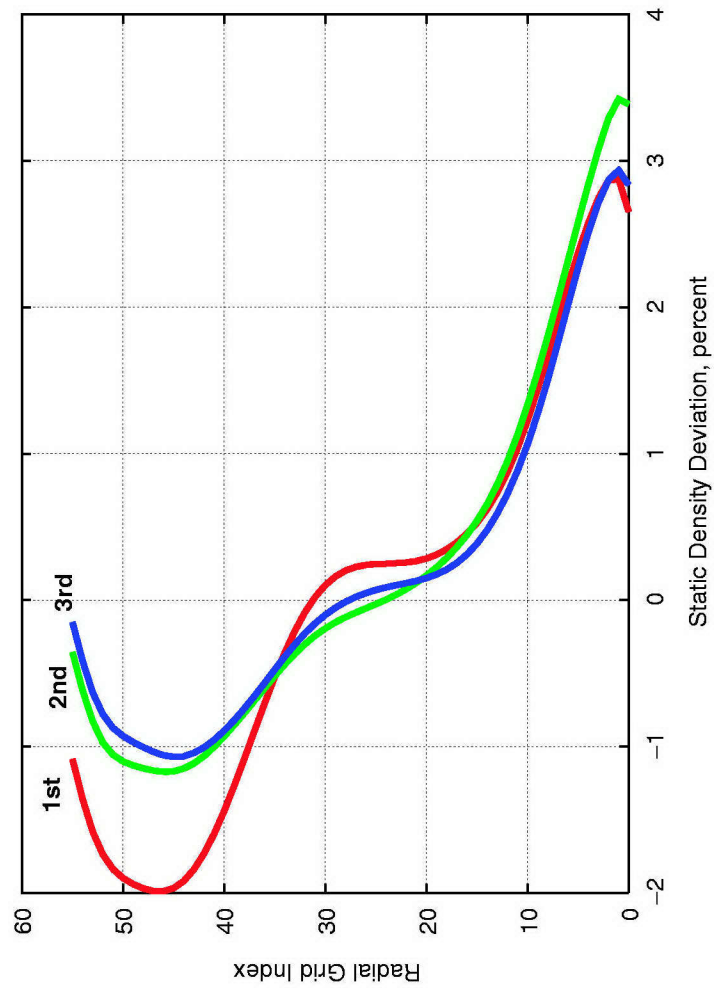


Figure 42. Static Density Component of Objective Function for Three Iterations of MOC Contour Optimization

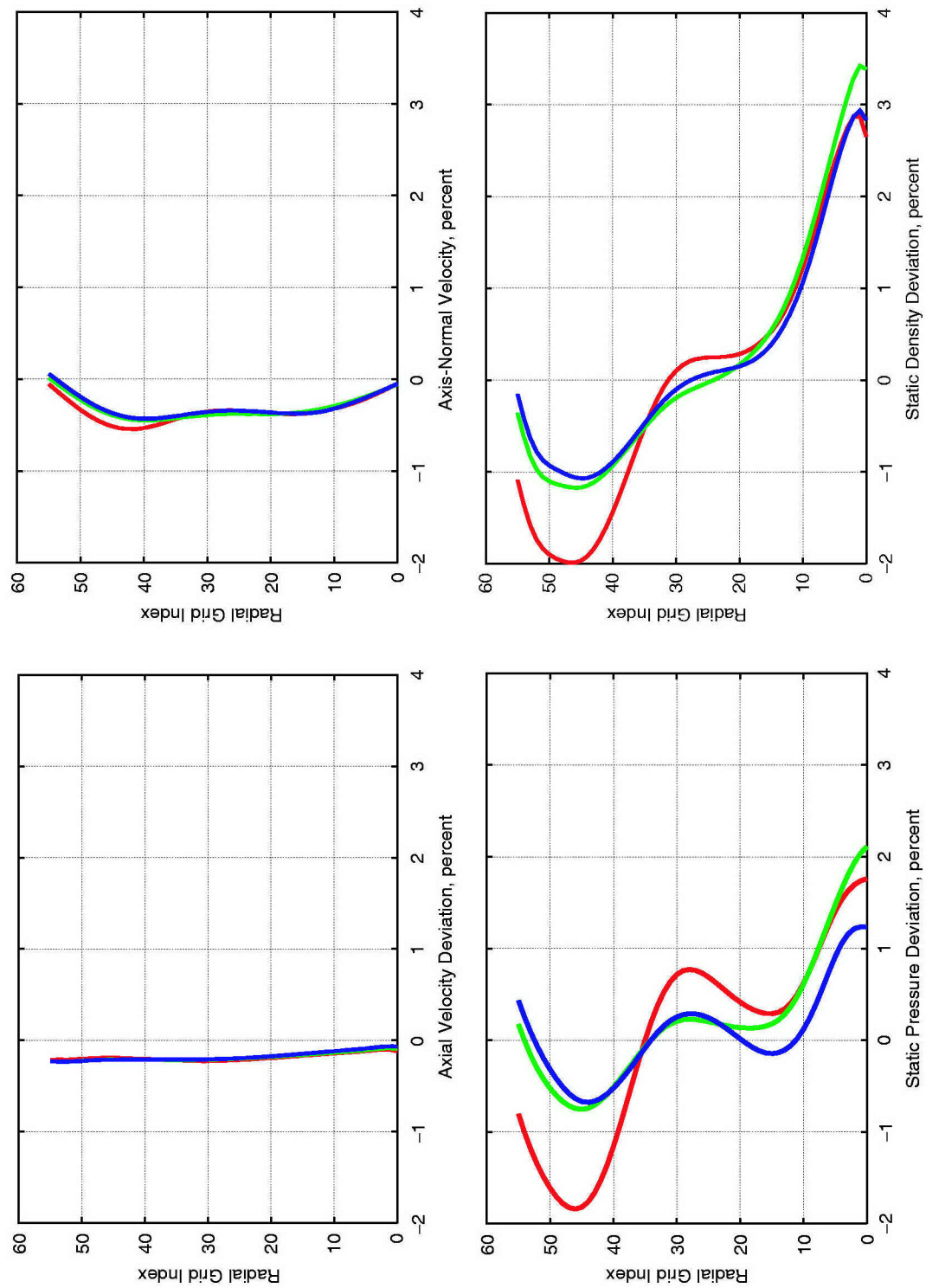


Figure 43. Comparison of All Four Components of Objective Function for Three Iterations of MOC Contour Optimization

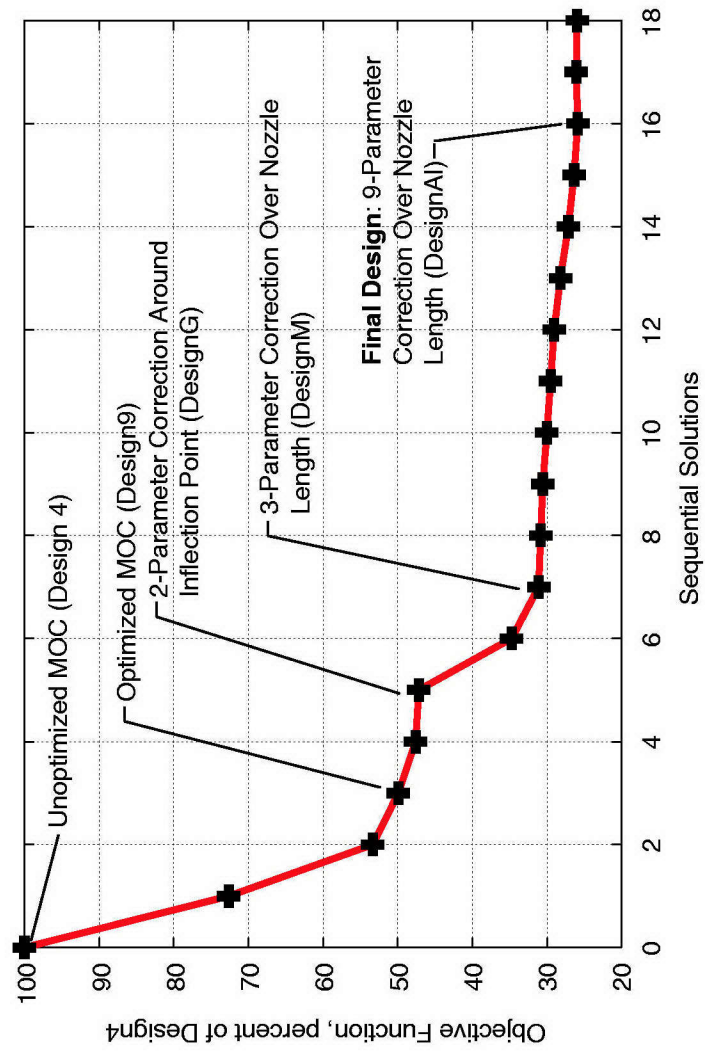


Figure 44. Objective Function History for LSO

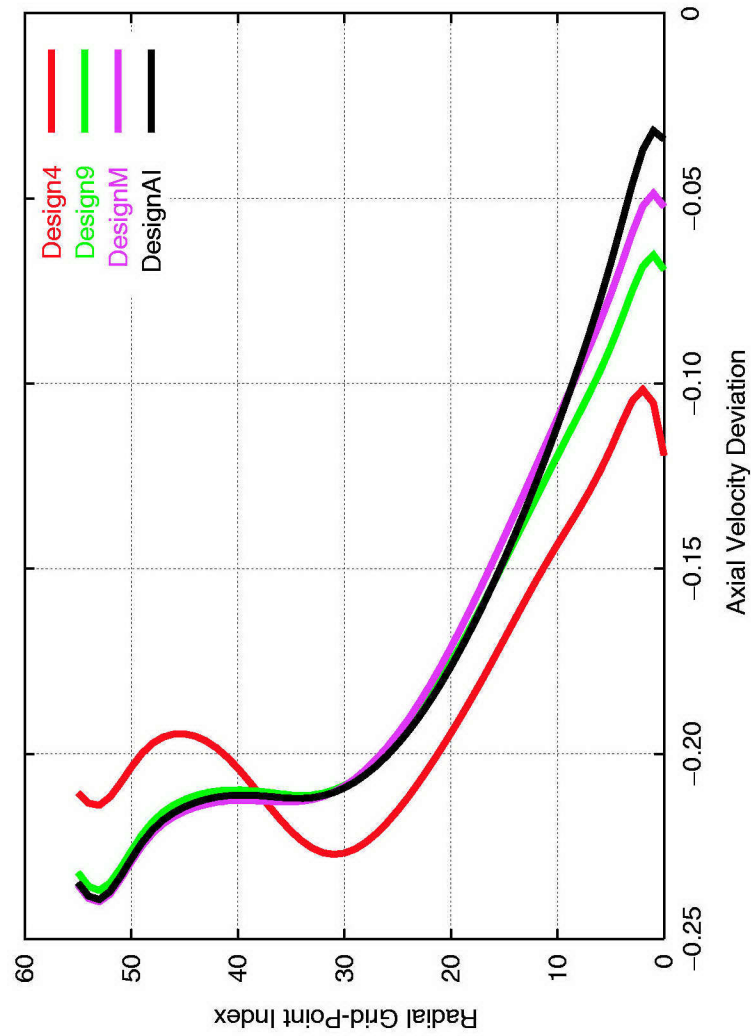


Figure 45. Axial Velocity Component of Objective Function for LSO

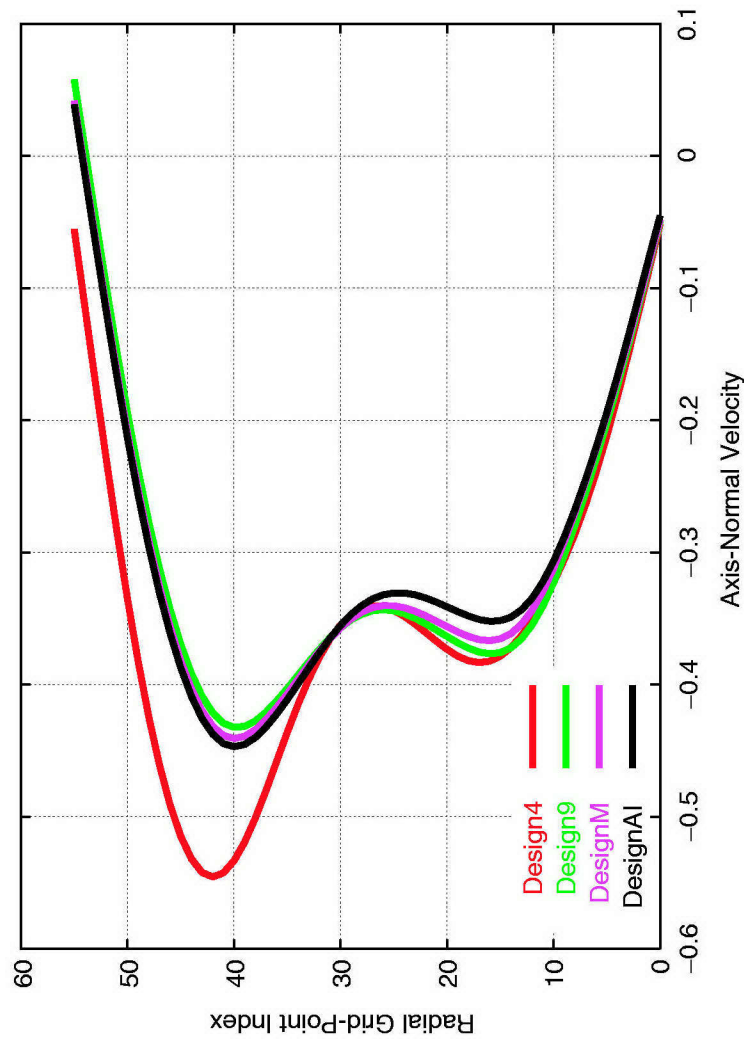


Figure 46. Axis-Normal Velocity Component of Objective Function for LSO

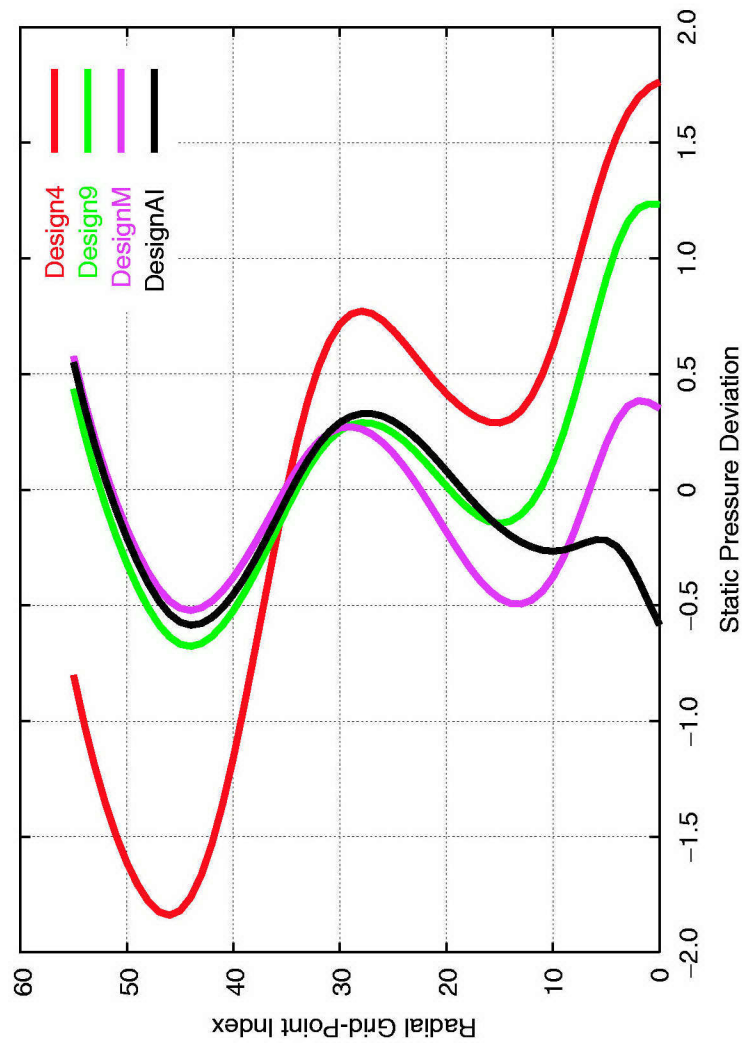


Figure 47. Static Pressure Component of Objective Function for LSO

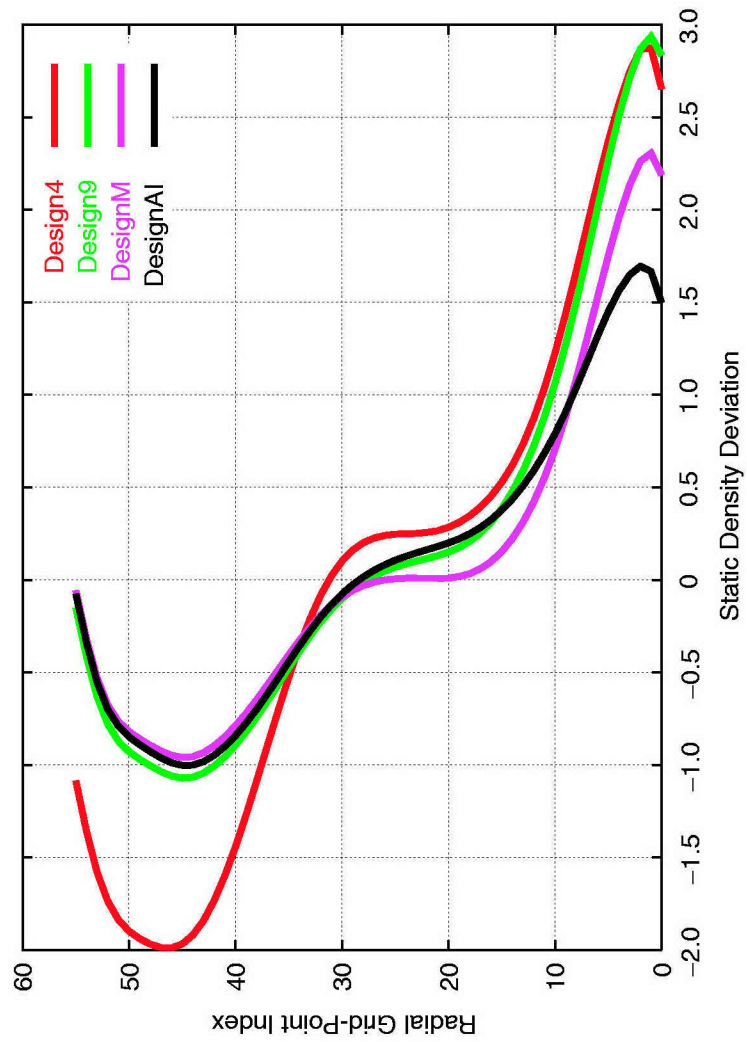


Figure 48. Static Density Component of Objective Function for LSO

APPENDIXES

The appendixes to follow describe the various components of the present nozzle design software. Attention is focused on the programs developed by the author; descriptions of the flow solver are less complete since they are documented elsewhere.[†] The reader should understand that this software is of restricted distribution and that access will require certain formal certification of eligibility.

The demonstration design accomplished herein used the 2002 version of the DPLR flow solver. A 2003 and a 2004 version (formally version 3.0) have been delivered to AEDC. The following material is not expected to change with use of the 2004 code, but as of this writing, the 2004 version has not been implemented in the design procedure.

In reading the following appendixes and while examining the software directories, the user may find useful the following explanation of the file and directory name conventions. In the text, computer-related names are in boldface type.

File Name Extension	Description
.c	Unix script file containing the compiler command
.dat	Data file, input or output
.dir	Directory name
.f	Fortran source file under Unix
.FOR	Fortran source file under Microsoft Windows [®]
.g	Batch-processing control file
.gnu	Gnuplot command file
.inp	DPLR input data file
.ps	Postscript file, often displayed with Ghostview
.s	Batch processing output file containing operating system messages and console output, often including unit 06 output
.t	Tcl source code
.x	Program executable generated by Fortran compiler
ftnXX	Default file name created during program execution without an explicit Fortran OPEN statement (e.g., ftn08)
(no extension)	Unix script file

[†] Candler, G.V. "APTU Nozzle Code Manual, Version 3.0." 30 September 2004 (included with DPLR software distribution).

A collection of computer files corresponding to those documented herein is located in an AEDC mass storage system under the following directory name:

ReactingNavierStokesNozzleDesignProcedure.dir

As of this writing the files may be found on the system umassy at the following location:

/pool1/users/b09452/ReactingNavierStokesNozzleDesignProcedure.dir

Although this directory may well be migrated to other systems as time passes, the bottom-level directory name will be maintained as given above.

APPENDIX A

AUTOMATED NOZZLE DESIGN OPTIMIZATION (ANDO.t and ANDO2.t)

An automated procedure is provided to allow use of Candler's DPLR code (Ref. A-1) to design nozzle contours. The automated procedure (Step 2) is intended to be used after completion of a manual procedure (Step 1) that finds an optimum contour that is limited to contours generated by method of characteristics (MOC). The automated procedure of present focus applies a spline-fitted correction to the optimized MOC. The design parameters are the radius corrections of the contour at the spline nodes. The optimization procedure perturbs these corrections and generates DPLR batch jobs to compute the effect of the baseline correction and each perturbation on the exit flow profiles. The flow profiles from each batch job are used in a least-squares optimization (LSO) step that yields corrections to the initial corrections.

The present procedure consists of two Tool Control Language (Tcl) programs (Ref. A-2), ANDO.t and ANDO2.t. ANDO.t generates the directories and batch jobs for a single LSO step. ANDO2.t collects the output files from each batch job and runs the LSO program. Subsequent LSO steps each require a manual execution of ANDO.t and ANDO2.t. The manual control is used because, at present, convergence of a fully automated LSO iteration seems unlikely. Following execution of ANDO.t and after completion of all batch jobs, it is expected that the user will then execute ANDO2.t to finish the LSO step.

This appendix is intended to document what ANDO.t and ANDO2.t are doing, some of the important assumptions behind their operation, and the primary steps the user must accomplish. ANDO.t and ANDO2.t are discussed sequentially below.

ANDO.t and ANDO2.t are written in Tcl and are presently being interactively executed on an HP Superdome (uhp32) under Tcl 8.3. The codes are less than 150 lines, including comments and print statements. Execution of ANDO.t takes on the order of minutes, with most of the time consumed by nozgrid.x. ANDO2.t runs in about one minute, and most of the time is accounted for by the execution of read.f.

A.1 ANDO.t

ANDO.t creates one directory for each batch job. The directory names are numbered 11, 12, ... 99, and this number is part of the directory name. Only two-digit numbers greater than 10 and less than 100 are permitted. A limit of 89 batch jobs may be generated. A flow chart of ANDO.t is shown in Fig. A-1.

The user must first create directories called start.dir and commonfiles.dir. The directory start.dir must contain the following files:

```

AND0.inp (created by user)
AND0.t (altered by user)
clean.x
go
nodepe03.dat (created by user)
nodepe05.dat (created by user)
nodepe07.dat
nodepert.f
nodepert.x
spline02.dat (created by user)

```

The user must create AND0.inp, which must contain a list of directory names that will be created by AND0.t; for example,

```

DesignV11.dir splin0211.dat
DesignV12.dir splin0212.dat
DesignV13.dir splin0213.dat
DesignV14.dir splin0214.dat
...
DesignV99.dir splin0299.dat

```

AND0.t creates the directories DesignVxx.dir. The files nodepe03.dat, nodepe05.dat, and spline02.dat must be created by the user according to the input description in Appendix D for nodepert.f. The file nodepe03.dat contains the perturbations to the corrections, which, for a five-node spline, might be

```

0.0000
0.0002
0.0002
0.0002

```

Note that the first value must always be zero and the last value (not input) is assumed to be zero. Thus, in this example, only three nodes are perturbed. A total of four batch jobs and four directory sets are created for this example: directory DesignV11.dir (for the baseline correction without perturbations) and DesignV12.dir, DesignV13.dir, and DesignV14.dir (for the three perturbations).

The file nodepe05.dat contains only a user-defined line of identification information and may be arbitrary.

The user must create the file spline02.dat, which contains the initial corrections of the MOC contour. Each record contains two values, the x-station and the radial correction. The following is

a sample file read by nodepert.f. Note that this is for a five-node spline correction and that the first and last corrections must be zero.

```
5 correction file DesignV
5.04469800E-01    0.0000
1.08166800E+00    0.0010
1.60148600E+00    0.0030
2.23784800E+00    0.0008
2.74787300E+00    0.0000
```

The first name on each record of ANDO.inp is the directory name to be created for each batch job. The second name is the name of the file to contain the perturbed corrections. The characters "DesignV" may be chosen by the user but should be the same for all file names. The characters "spline02" and ".dat" must not be altered. The characters "11", "12", must be entered as shown and must increase sequentially. The splin02xx.dat files (xx = 11, 12, ...) will be created by program nodepert.f.

The directory commonfiles.dir must contain the following files:

aptu_nozzle.c	plotg
aptu_nozzle.f	q
aptu_nozzle.o	ql
aptu_nozzle.x	read.c
chon_aptu.inp	read.f (may require editing and compiling by user)
coords.dat (created by user)	read.gnu
nozgri05.dat (created by user)	read.o
nozgrid.c	read.out
nozgrid.f	read.x
nozgrid.gnu	read02.dat (created by user)
nozgrid.x	readme.txt
nozzle.comm	run
nozzle.inp (edited by user)	spline01.dat
nozzle.update	splinenc.c
obj.c	splinenc.f
obj.f	splinenc.gnu
obj.o	splinenc.o
obj.x	splinenc.ps
obj05.dat	splinenc.x
plot	

The files in this directory are copied by ANDO.t to each of the newly created directories. It is recommended that the user copy these files from a recent DPLR directory to commonfiles.dir.

Not all of these files are necessarily used by the LSO procedure. The file `coords.dat` contains the coordinates of the nozzle contour prior to the present LSO step and prior to addition of the corrections or perturbations. The file `nozgrid05.dat` is input to the grid generator (`nozgrid.f`). The file `nozzle.inp` is the primary user-input file to the DPLR flow solver (`aptu_nozzle.f`). See user manual accompanying DPLR (Ref. A-1) and comments embedded in the file. The file `read02.dat` is the user input to the postprocessor (`read.f`), which generates the flow profile files for the LSO objective function. The user may need to edit and recompile `read.f`, for example, to set the number of grid points selected for the flow profile in the objective function.

This directory should specifically exclude a file by the name `aptu_nozzle.s` because this file is created by the batch job, and its presence is an indicator that the batch job has run and finished. In addition, a file by the name `aptu_nozzle.g` should not be present because this is created by `ANDO.t`. If `aptu_nozzle.s` and `aptu_nozzle.g` are present, they should be manually deleted before execution of `ANDO.t`.

`ANDO.t` may be executed from the directory `start.dir` with the command `go`, which executes `ANDO.t` and displays the results with `nedit`.

`ANDO.t` first executes the program `nodepert.f` (executable is `nodepert.x`), which reads `nodepe05.dat`, `nodepe03.dat`, and `spline02.dat`. Program `nodepert.f` generates files named `splin0211.dat`, `splin0212.dat`, etc., which are input files to `splinenc.f`. These files are placed in the directory `start.dir`. Next `ANDO.t` reads the directory and filenames from `ANDO.inp` as illustrated above. It creates new directories `DesignV11.dir`, etc., on the same level as `start.dir`. Accordingly, these directories should not already exist, or the procedure will fail. The procedure `clean.x` in `start.dir` removes these directories but may need alteration if other directory names are to be used. `Clean.x` assumes the directory names to be removed begin with "D." Next, `ANDO.t` sequentially changes directory (`cd`) to each of the new `DesignV11.dir`, etc., and accomplishes processing in each directory. First, it copies all of the files in `commonfiles.dir` to the current directory. Next, it copies `splin0211.dat` to `spline02.dat` and `coords.dat` to `spline01.dat`. Then it executes `splinenc.f`, which spline fits the correction distribution, including any perturbations, generating `spline03.dat` in the process. The file `spline03.dat` contains the nozzle coordinates, with the correction and perturbations. The original file `coords.dat` is removed, and `spline03.dat` is copied to `coords.dat` (n.b.: the file `coords.dat` in `commonfiles.dir` is not altered). Next, `ANDO.t` creates the file `aptu_nozzle.g`, which controls the batch job. The contents of `aptu_nozzle.g` are contained in Tcl puts statements in `ANDO.t` and may need to be altered by the user for a specific computer system. Next, `ANDO.t` executes the grid generator `nozgrid.f`. Finally, `ANDO.t` submits the batch job for the flow solver with the `bsub` command. The flow-solver job proceeds as illustrated in Fig. A-2. When all of this is finished for one directory, `ANDO.t` changes to the next directory and repeats the process until all directories given in `ANDO.inp` have been created, filled, and processed.

A.2 ANDO2.t

ANDO2.t is executed manually by the user after all DPLR batch jobs have finished. It is executed from the directory named start.dir. If ANDO2.t is being rerun for some reason, clean2.x should be executed first to remove directories, which ANDO2.t will create (again). A flow chart of the process controlled by ANDO2.t is shown in Fig. A-3.

ANDO2.t begins by creating the directory DLSO.dir on the same level as start.dir. DLSO.dir is where the flow profile files are to be copied and the LSO program lso.f is to be executed. The directory name starts with "D" so that clean.x and clean2.x will delete it. After directory creation, the following files are copied in from start.dir:

```
lso.x
targets.dat
nodepe04.dat
```

The file lso.x is the executable for lso.f.

The file targets.dat contains the numerical values of the target variables that form the objective function being minimized by lso.f. That is, the difference between the DPLR flow profiles (at each selected grid point) and the target value, after squaring and summing, yields the objective function value for a single DPLR solution. Following is a sample targets.dat file:

```
1798.8d0          CEA U [m/s]
0.d0             V = 0
2151.765910198162d0 Design4 mean P [Pa]
3.322589131034744d-02 Design4 mean RHO [kg/m3]
end              end of file indicator
```

The above file contents imply that four flow variables make up the objective function. Unless the value is zero, these values are also used to scale the objective function. The user must create targets.dat, which remains unchanged for the entire nozzle design process. The character data in the second column is for information only and is not read by lso.f.

The file nodepe04.dat contains the following data (as an example):

```
1.0000E-03  MOC CORRECTION
3.0000E-03  MOC CORRECTION
8.0000E-04  MOC CORRECTION
2.0000E-04  PERTURBATIONS
2.0000E-04  PERTURBATIONS
2.0000E-04  PERTURBATIONS
```

The first three lines are the corrections made to the optimized MOC contour or contour from a previous NDO step. The second set of three values is the perturbations to the corrections. This file is generated by nodepert.f on the basis of user input described above. Again, the right column is for identification information only.

The files targets.dat and nodepe04.dat are later used to assemble the input file lso05.dat.

Next, ANDO2.t reads ANDO.inp (the same file used by ANDO.t), which might contain

```
DesignV11.dir splin0211.dat
DesignV12.dir splin0212.dat
DesignV13.dir splin0213.dat
DesignV14.dir splin0214.dat
end
```

ANDO2.t uses this information to define the names of the flow profile files to be read by lso.f. Each of these files is called read09.f in the directories containing the DPLR solutions, but they are all needed in DLSO.dir and must have unique names. To define unique names, ANDO2.t uses only the first column of ANDO.inp, strips off the characters ".dir," and replaces them with ".dat." Thus, the flow profile file names will be the same as the batch directory names (except for the .dir and .dat). The profile file names are stored in a Tcl array called profilefilename. In the present example, there are four such file names. The first is for the baseline solution corresponding to the original contour modified by the correction. The next three correspond to each of the three perturbations (see nodepe04.dat above).

Next, ANDO2.t executes a for-loop, with one iteration occurring for each batch directory. In the loop, the postprocessor code read.f is executed to generate the file read09.dat, which contains the flow profile data. Program read.f sends output to the standard error channel because of the Fortran "stop 'message'" command even though no execution error has occurred. For this reason, the Tcl catch command is used. After read.x generates read09.dat, ANDO2.t copies read09.dat to the file name stored in profilefilename in directory DLSO.dir.

Next, ANDO2.t changes its current directory to DLSO.dir. It then creates the lso.f input file lso05.dat. First, it writes the flow profile file names to lso05.dat. Then it copies the contents of targets.dat to lso05.dat, following that with the contents of nodepe04.dat. Finally, it writes a record to lso05.dat containing the user-selected relaxation factor. The user sets this value by editing ANDO2.t before execution. At present, a default value of 0.1 is being used, which implies that the Newton iteration on which LSO is based is being used only to obtain a search direction. Even so, convergence is not assured.

After lso05.dat is complete, the file is closed by Tcl. This has the effect of rewinding the file (positioning the read pointer at the first record). If the close is not executed, the file pointer is at

the end of file, and the Fortran Iso.f will fail when it tries to read the first record. Finally, Iso.x is executed. Its primary output is on Iso07.dat.

The following is a list of things to do to run ANDO.t:

- Create design directory name (e.g., Design V.dir)
- Create start.dir and copy in contents
- Create commonfiles.dir and copy in contents
- Create ANDO.inp with batch run directory names, perturbation file names
- Edit ANDO.t for maximum number of directories, uhp32 batch job control (bsub commands)
- Create nodepe03.dat containing y-perturbations
- Create nodepe05.dat containing case identification information
- Create spline02.dat containing node stations and y-corrections
- Create coords.dat containing the nozzle coordinates
- Create nozgri05.dat containing input to nozgrid.f
- Edit nozzle.inp
- Edit read.f, if necessary, and recompile
- Create read02.dat
- Execute ANDO.t (type "go")
- Review results in ANDO.o via nedit (started automatically)
- Determine when all batch jobs have finished (e.g., run qstat)
- Create targets.dat
- Edit ANDO2.t for relaxation factor value, if necessary
- Execute ANDO2.t (type "go2")

Below is a formal procedural check list for the DPLR optimization procedure. Both ANDO.t and ANDO2.t steps are included.

Design ID																	

Create design.dir																	
create start.dir																	
create commonfiles.dir																	
create ANDO.inp																	
edit ANDO.t																	
create nodepe03.dat																	
create nodepe05.dat																	
create spline02.dat																	
create coords.dat																	
create nozgri05.dat																	
edit nozzle.inp																	
edit read.f																	
create read02.dat																	
execute ANDO.t																	
review results																	
verify batch jobs done																	
create targets.dat																	
edit ANDO2.t																	
execute ANDO2.t																	

REFERENCES

- A-1. Candler, G. V. "APTU Nozzle Code Manual, Version 3.0." 30 September 2004 (included with DPLR software distribution).
- A-2. Welch, B. W., Jones, K., and Hobbs, J. *Practical Programming in Tcl and Tk*. Prentice Hall PTR, Upper Saddle River, NJ, Fourth Edition, 2003.

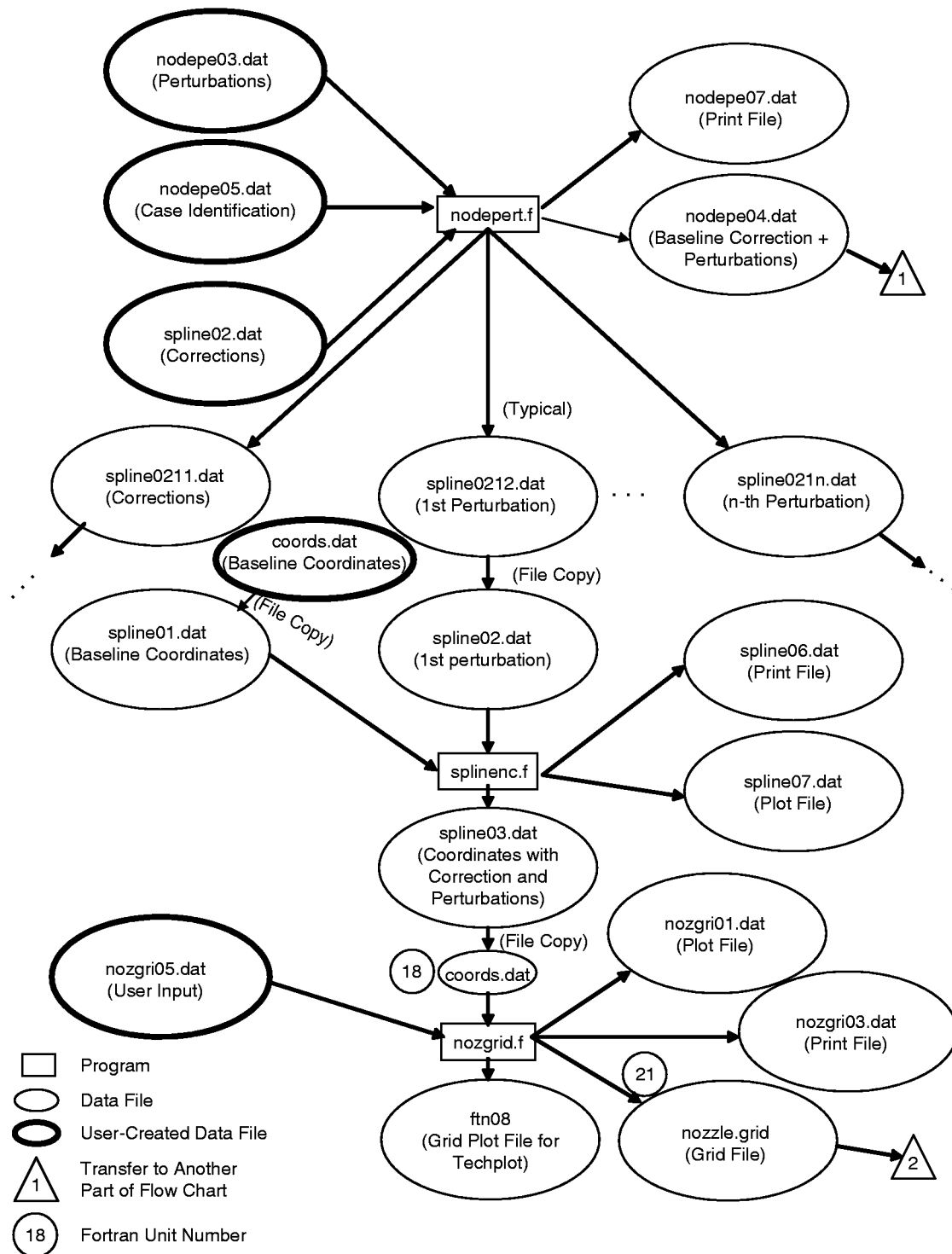


Figure A-1. ANDO.t Flowchart

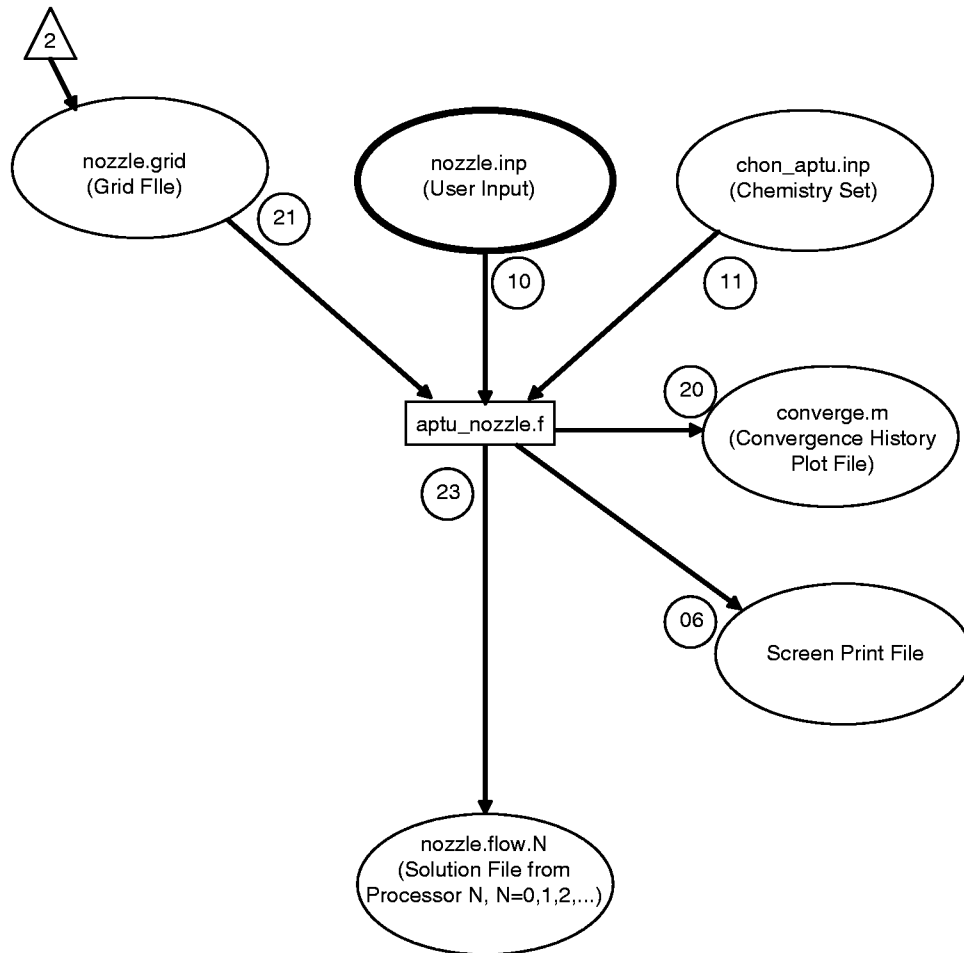


Figure A-2. Flowchart of DPLR Batch Job as Submitted by ANDO.t

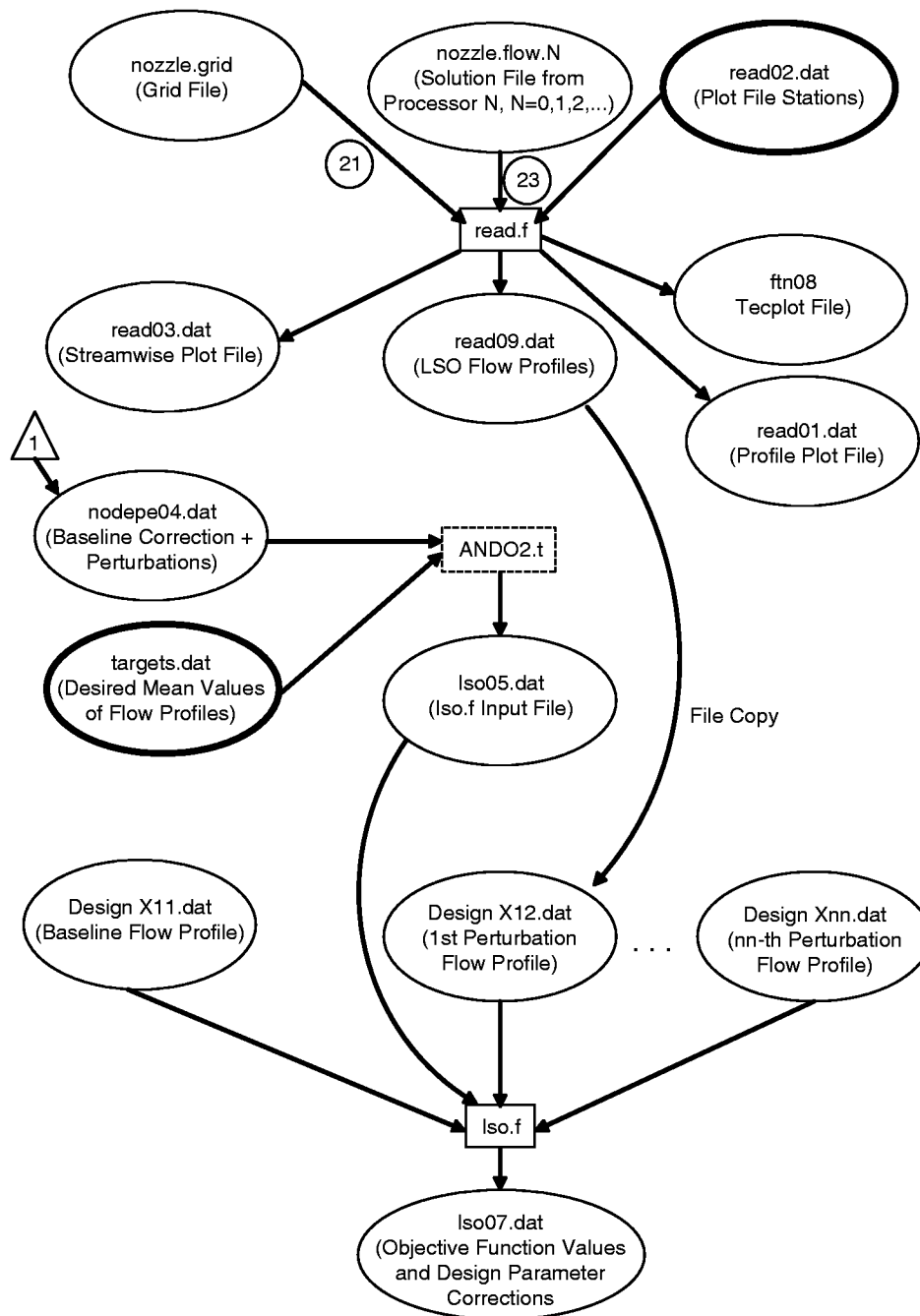


Figure A-3. ANDO2.t Flowchart

APPENDIX B

LEAST-SQUARES OPTIMIZATION PROGRAM (Iso.f)

The least-squares optimization program (Iso.f) computes corrections to a set of design variables with a goal of reducing an objective function to zero as described in Section 3.1.2 above. The source code consists of a main program and three small subroutines that solve a nonsparse linear system. When Iso.f is executed, it is assumed that flow-solver solutions have been computed and that each has provided a nozzle exit profile of the flow variables chosen for the objective function. Each profile is on a separate file, and each column of data on the file is one flow variable. Each row of the file corresponds to one grid point on an axis-normal line across the nozzle exit. The main program first reads the names of these files from a user-prepared file named Iso05.dat (Fortran unit 5, see Table B-1, below, for input description). The first profile file name is for the baseline solution (i.e., the original contour plus any correction) before the design parameters have been perturbed. The remaining filenames each correspond to the perturbation of one design parameter, with the others having the baseline value. Next, the main program reads a target value corresponding to each column of the profile files. For example, the target values might be a pressure value, density value, and two velocity-component values. The entire profile is to be driven toward the target values by varying the design parameters. Next, the main program reads the values of the design parameters before and after being perturbed. The initial unperturbed values are referred to as the baseline in the source code comment lines. The perturbed values of the design parameters are what were used for each of the previously computed flow-solver solutions. These perturbations are to be used in Iso.f to compute the finite-difference derivatives that comprise the Jacobian matrix. Finally, MAIN reads the relaxation factor, and this completes the input from Iso05.dat.

Next, MAIN reads the profile file defined on Iso05.dat for the baseline solution, which corresponds to the unperturbed baseline values of the design parameters. Next, MAIN computes the difference between the baseline profile property value and the target value for each point and each property on the baseline profile. The difference is scaled by the target value, unless the target value is zero (as it is for the axis-normal component of velocity, in which case it is scaled by the target value for the axis-parallel velocity and is thus the tangent of the flow angle). Note that the special case of zero target value for axis-normal velocity is hardcoded in Iso.f. Next, the objective function itself is computed as the sum of square-scaled differences over all grid points with a separate value stored for each of the four properties. This separation is retained to allow assessment of the importance of the individual properties in contributing to the final objective function value. Also, the total objective function is computed as a sum over all grid points and property values (resulting in a single number for the baseline solution).

Next, MAIN reads a profile file for each design parameter perturbation (i.e., one for each flow-solver solution). Then MAIN accomplishes the same differencing, scaling, and objective function computation for each profile file as described for the baseline.

Next, MAIN computes the Jacobian matrix. The matrix has one column for each design parameter and one row for each profile file value (i.e., one row per grid point per flow property). For example, if there are 100 grid points on the nozzle exit profile and four property values, the Jacobian matrix would have 400 rows. Then, MAIN forms the transpose of the Jacobian and computes the matrix product of the transposition and the Jacobian, producing a square matrix. MAIN now computes the right-hand side of the linear system as the product of the transposed Jacobian matrix and the negative baseline scaled difference vector. Finally, MAIN solves the now-complete linear equation system

$$J^T J \Delta \alpha = -J^T \epsilon$$

to obtain the vector of corrections to the baseline design parameters.

To complete the processing, a series of verification tests is computed on the linear equation solution. Then the final results are written on file Iso07.dat, which contains many interim results and is printer compatible. There is no output file whose contents are fed directly into a next-LSO iteration. The primary outputs are the values of the corrected design parameters and are printed on unit 7.

Processing information: There are three code dimensions that must be tailored to the specific problem. These dimensions are set in a parameter statement. They are the variables NA, NV, and NL, defined as follows:

NA = number of design parameters.

NV = number of flow variables that form the differences (i.e., if nonuniformity of pressure, density, and two velocity components were to be minimized, then NV = 4).

NL = number of geometric locations (grid points) where flow variables are taken.

After setting these values, the user must compile Iso.f for 64-bit arithmetic. Fortran 90 has been used here, but Fortran 77 should be adequate.

Table B-1. Input Description for File Iso05.dat

Variable Name	Format	Definition
FNAME(IA)	A15	File names containing the flow solution. These files are the output of read.f. names of form "XXXXXXXXXXXXX.XXX," i.e., 15 characters or less (one file name per record). IA = 0 is the baseline solution (unit 10). IA = 1, NA are the solutions for the perturbed design parameters. NA = number of design parameters. (unit = IA + 10, i.e., 11, 12, ...)
QT(IV)	READ(*)	Target values of flow variables, IV = 1, NV (one value per record)
ALPHA0(IA)	READ(*)	Design parameter values for baseline solution (one value per record)
ALPHA(IA)	READ(*)	Design parameter values for perturbed solutions, IA = 1, NA (one value per record)
F	READ(*)	Relaxation factor for computing corrected design parameters, used for printing only, $0 < F \leq 1$

APPENDIX C

SPLINED CORRECTION PROGRAM (splinenc.f)

This program computes a spline fit through a set of small corrections to a nozzle contour, where the contour is defined as a table of coordinates, and then computes a new contour as the original table plus the spline-interpolated corrections. The important point here is that the spline fit is not to the coordinate table but to the correction distribution. The reason for such a program is to allow a sufficient number of tabular coordinates for contour accuracy while using only a few nodes (design parameters) for the corrections. A feature of this program is that the range of the spline may be varied from the entire length of the tabular contour to any contiguous subset of the contour. A restriction is that the first and last points of the spline correction must be coincident with points on the coordinate table. The program file splinenc.f consists of a main program plus subroutines for spline-based interpolation (CSINTERP), computation of the spline coefficients (SPLINE and CALCCF), evaluation of the splines (PCUBIC), and a second-degree interpolation procedure for setting boundary conditions needed by the splining process (PSLOPE).

The program processing proceeds as follows. First, the original contour is read from spline01.dat. The number of points on the table is read followed by the coordinates themselves with an x-y couple on each record. Next, the number of points on the correction distribution is read from spline02.dat followed by the x-value of the node and the radial correction at that x-value. The first and last x-values read here must be within a tolerance of 1.E-4 of points on the input contour. The program is dimensioned for 1000 points. See Table C-1 for more information.

After input is finished, the boundary conditions for the spline fit of the correction distribution are set to zero slope at both ends. That is, the correction distribution must not change the slope of the input contour at the correction-range end points. Next, the end-point slopes of the input coordinates are set, with zero at the throat and the slope of a parabola through the last three points at the nozzle exit. These slopes are boundary conditions used in the spline-fitting computation. The input coordinates are splined only to generate derivatives for graphical comparison with the corrected output contour. Otherwise, the splined input contour is unused.

Next, the process of spline-fitting the correction distribution commences. First, the x-coordinates of the corrected output table are set equal to those for the input table. Then, the array indices of the two x-coordinates from the input contour that match the first and last x-coordinates of the end points of the correction distribution are determined. Execution is halted abnormally if matches are not found within the tolerance of 1.E-4. Next, the spline fit to the correction distribution is computed and interpolation of correction values at the input coordinates is computed.

Finally, the corrected contour is computed as a sum of the input coordinates and the splined, interpolated corrections.

For postprocessing analysis, spline fits are next computed for both the input and corrected contours. The results are used to compute maximum differences between the two contours in terms of coordinate, slope, and second derivative. Finally, a plot file is written on spline07.dat, and the corrected contour is written to spline03.dat. Execution is then terminated normally.

Table C-1. Input Description for splinenc.f

File Name	Variable Name	Format	Description
spline01.dat	NNI	READ(1,*)	Number of points on original input contour
spline01.dat	XNI(I), YNI(I)	READ(1,*)	x-y coordinate couples, two values per record, I=1,NNI ; $NNI \leq 1000$
spline02.dat	NCI	READ(2,*)	Number of points on correction table
spline02.dat	XC(I), YC(I)	READ(2,*)	x-coordinate of correction and radial correction value, two values per record, I=1,NCI; $NCI \leq 1000$. XC(1) and XC(NCI) must each coincide with some value of XNI. The range [XC(1),XC(NC)] need not span the entire nozzle length.

APPENDIX D

NODE PERTURBATION PROGRAM (nodepert.f)

To clarify the purpose of the program nodepert.f, some specialized terminology has been established. During one iteration of a least-squares optimization process, three types of nozzle contours are defined. The first contour type is referred to as the *baseline* contour and is stored on file coords.dat. The baseline contour is that contour before any changes have been made. The second contour type is the *corrected* contour. The corrected contour is the baseline plus a small spline-fitted correction in the radial direction. The spline passes through user-specified correction values at a few nodes along the nozzle contour. The corrections at these nodes are the design parameters. To accomplish the least-squares optimization, these corrections are each perturbed one at a time, and a Navier-Stokes solution is to be computed for each perturbation. Accordingly, for each correction (or each design parameter), a *perturbed* contour is computed as the baseline contour plus a spline-fitted perturbation, where the correction at a single node has been perturbed before the spline fit. The contours are illustrated in Fig. D-1. The program splinenc.f computes the spline fits for both the corrected contour and the perturbation contours. Table D-1 summarizes this terminology.

On the basis of the same terminology, the description of program nodepert.f is as follows. First, a case identification record is read from nodepe05.dat. This is the only record on unit 05. Next, the correction distribution is read from nodepe02.dat: the number of nodes is read first and is followed by the node stations and the correction values. Next, the perturbations of the corrections are read from nodepe03.dat. The number of perturbations is equal to the number of nodes. The perturbation of the first and last nodes must be zero and must be input as such. In general, the corrections and perturbations are expected to be small compared to the local nozzle radius.

After input is complete, the data are subjected to several checks. The number of nodes must be less than or equal to 89. In addition, both the corrections and the perturbations at the first and last nodes must be zero. Otherwise, processing is abnormally terminated with an error message.

Next, file names are created to contain the perturbed corrections. There is one such file for each node except the last. Each file name has the form splin02XX.dat where XX=11,12,...,NN-1+10, where $NN \leq 89$ and $XX \leq 98$. File splin0211.dat is the set of corrections without perturbation, and the remaining files splin0212, splin0213, ..., contain the perturbed corrections, one perturbation per file. The last perturbation is always zero and does not require a file or flow-solver evaluation. After the file names are created, the node stations and perturbation values are written to them. Each file has two data elements per record: the node station and the perturbation value. Table D-2 lists the input descriptions.

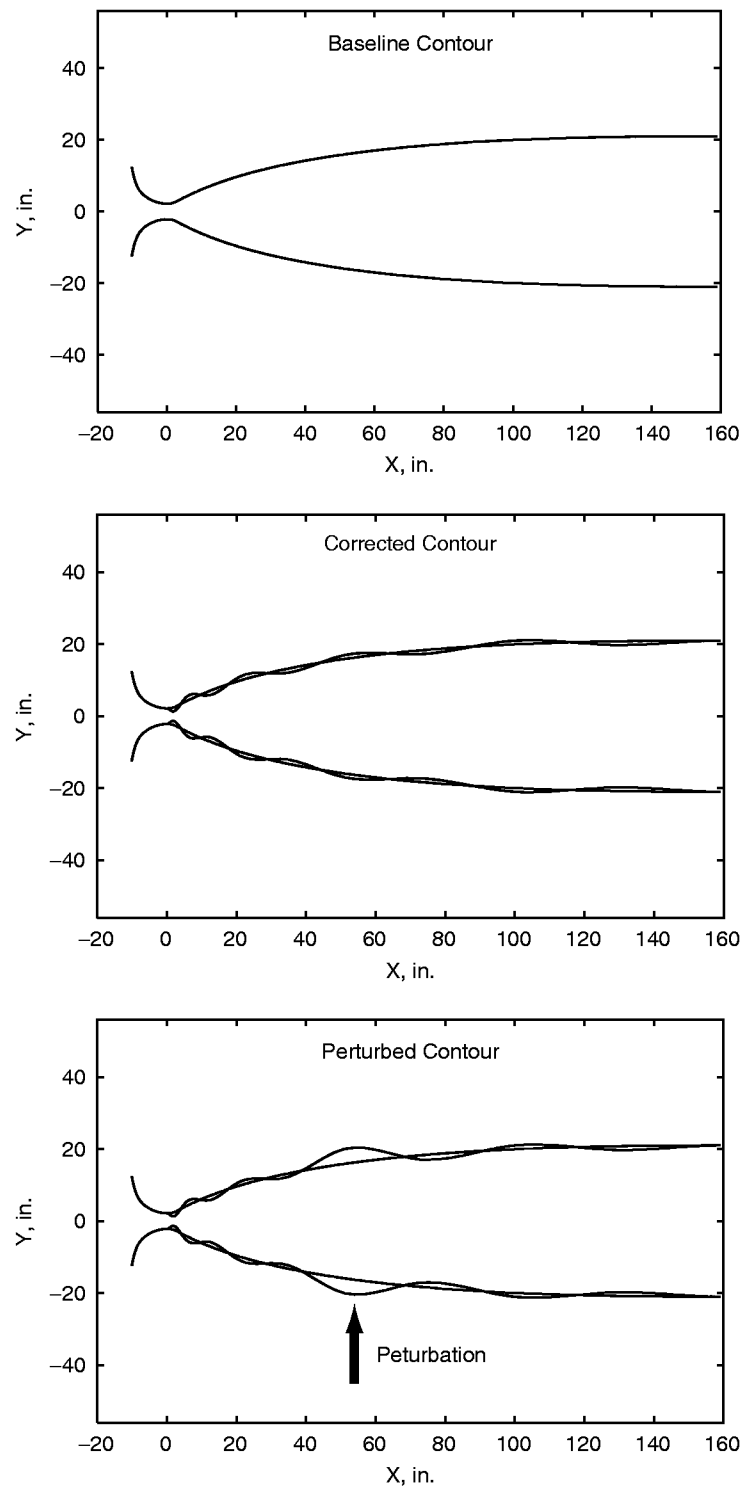


Figure D-1. Baseline, Corrected, and Perturbed Contours (Exaggerated)

Table D-1. Terminology for Three Types of Contours

<i>Baseline</i> Contour	Nozzle contour before any alterations are made
<i>Corrected</i> Contour	Baseline contour plus a spline-fitted correction where the correction values are specified at a few nodes along the contour and are the design parameters
<i>Perturbed</i> Contour	Baseline contour plus a spline-fitted perturbation, where the perturbation distribution is the result of perturbing a single correction value (at one node)

Table D-2. Description of Input to nodepert.f

File Name	Variable Name	Format	Description
nodepe05.dat	TITLE	A80	User-specified case identification information
spline02.dat	NN	READ(,*)	Number of nodes for correction spline
spline02.dat	XN(I),YN(I)	READ(,*)	x-value of correction node and value of correction, two values per record; I=1,NN. YN(1) and YN(NN) must be zero. NN≤89.
nodepe03.dat	DY(I)	READ(,*)	Perturbations of corrections, one per record; I=1,NN. DY(1) and DY(NN) must be zero.

APPENDIX E

OBJECTIVE FUNCTION EVALUATOR PROGRAM (obj.f)

It is frequently necessary to compute a single flow-solver solution and to evaluate the objective function resulting from the exit flow. Such is necessary after completion of a least-squares optimization (LSO) step when it is desirable to evaluate the effect of the chosen corrections on the design parameters without committing to a new LSO step. The program lso.f performs the computations needed to evaluate the objective function, but lso.f expects at least two input profiles to be read in. Accordingly, the program obj.f was created as a subset of lso.f computations. Thus, the objective function evaluation in obj.f is exactly as described in Appendix B for lso.f. The only input to obj.f is the target values of the profile properties (see Table E-1) and the profile file itself, obj01.dat, which is copied from read09.dat, obtained as the output of read.f.

Table E-1. Input Description for File obj01.dat

Variable Name	Format	Definition
QT(IV)	READ(,*)	Target values of flow variables, IV = 1, NV (one value per record)

Caveat: If the designer chooses to run a single solution to check the resulting objective function and later decides to proceed to another LSO iteration with the full complement of flow-solver solutions, it should be unnecessary to rerun the baseline solution. However, ANDO.t as coded does not contain an option to skip this rerun.

APPENDIX F DPLR CODE PACKAGE

Four Fortran 90 programs comprise the DPLR code package that constitutes the flow solver used herein. This software is documented in a separate user manual (Ref. F-1) that accompanies the software package and will not be fully documented here. The present documentation presents an overview with focus on adaptations pertinent to the nozzle design problem. The four computer programs are `choneq.f`, which computes the equilibrium stagnation conditions; `nozgrid.f`, the grid generator; `aptu_nozzle.f`, the flow solver; and `read.f`, the postprocessor. A high-level flow chart of the job processing is shown in Fig. F-1, which illustrates the input and output files for each of the four programs. Program modifications made for the nozzle design problem are superficial in that they only add input and output files and do not alter the computational procedure. Each program is discussed in the following sections.

F.1 PROGRAM `choneq.f`

This program computes the stagnation properties and composition for input to the flow solver. The program is executed once during initial problem setup but is not part of the iterative design process. The program `choneq.f` is tailored to the products of burning isobutane, air, and make-up oxygen. In the 2004 version of DPLR, there are additional programs for air and nitrogen, but these have not been demonstrated in the design procedure. Program `choneq.f` was modified to read the stagnation density and temperature and initial species concentrations from `choneq01.dat` and to write the primary output to `choneq02.dat`. The initial concentrations may be taken from any chemical equilibrium program that treats the needed species. The present work uses the NASA Glenn CEA96 program (Ref. F-2). The concentrations and other data on `choneq02.dat` are used to prepare input to `aptu_nozzle.f`. Table F-1 shows the input description for `choneq01.dat`:

F.2 PROGRAM `nozgrid.f`

This program is the grid generator for `aptu_nozzle.f`. Other grid generators may be used, but `nozgrid.f` will run in noninteractive mode (without user intervention), which is essential for the automated nozzle design procedure (ANDOT). The user must prepare two input files to `nozgrid.f`. The file `coords.dat` contains the nozzle coordinates. These data may be taken from a method-of-characteristics (MOC) design or from a previous iteration of the design procedure. The data are in the form of an x-coordinate and a y-coordinate, with two values per file record. Note that `nozgrid.f` skips the first record on the file; therefore, it may contain any information the user wishes. The coordinates should be in inches, and the throat is $x = 0$. Note that the last record must be -999. -999., which serves as an end of file.

A second user-prepared input file is `nozgri05.dat`. This was added to provide control over certain parameters without recompiling the source code. The contents of `nozgri05.dat` are defined in Table F-2.

The program modifies the geometry on `coords.dat` by adding a cylindrical section upstream of the input contour. The length of this cylindrical section can be controlled with the variables `inoz` and `fdd`. The program generates two grid plot files: `ftn08`, a Tecplot-compatible file, and `nozgr01.dat`, a gnuplot-compatible file. The primary output file is `nozzle.grid`, which is read by `aptu_nozzle.f`.

F.3 Program `aptu_nozzle.f`

No significant changes have been made herein to `aptu_nozzle.f`. The primary user task is to prepare `nozzle.inp`. Notice that the structure of this file has been carefully created to assist the user in preparing the file. Accordingly, the user should avoid adding or deleting lines in the file, except for step-size control. Information listed toward the end of the file is critical descriptive text for the various input variables. Once `nozzle.inp` is prepared, it probably will remain unchanged for the duration of the design procedure. The program will normally be run with parallel processing. Each process generates a solution file with file name `nozzle.flow.N`, where $N=0,1,2,\dots$. This form of the file name is defined on `nozzle.inp`, but it is recommended that this notation not be changed because `read.f` has this name format hardcoded. The user must set appropriate dimensions for the specific problem being computed, and the code must be recompiled. The user can also select the number of processors. The present demonstration used four processors. For further information, the user is referred to the DPLR user manual (Ref. F-1).

F.4 PROGRAM `read.f`

The program `read.f` combines the contents of the `nozzle.flow.N` files into a single array and generates a Tecplot-compatible plot file for the flow field. The program was modified to generate various additional files for plotting with gnuplot and for use with the least-squares optimization code, `lso.f`. Also, several simple statistical computations on the exit profile were added to evaluate means and variations of the profile. The mean values could be used for target values in the `lso.f` program. To obtain profile plot files, the user must prepare `read02.dat`, which is simply a list of plotting stations where profiles are to be plotted. The stations are in units of meters measured from the throat. The primary output file is `read09.dat`, which is in the format expected by `lso.f` and `obj.f` for the flow profiles. The program contains one internal variable that the user will need to set. This is the variable `jopt2`, which sets the index of the outermost grid point to be included in the flow profile used for optimization. The purpose of this option is to permit exclusion of the boundary layer from the profile to be optimized, since the contour design can do little to eliminate the nonuniformity of the boundary layer. Accordingly, `jopt2` should be set to the index of the last outer radial grid point that is not in the boundary layer. Once set, this should not need to be changed during the design optimization. The variable `jopt2` could have been placed on the input file, but `read.f` must be recompiled for each new design problem because of possibly changing dimensions.

REFERENCES

- F-1. Candler, G. V. "APTU Nozzle Code Manual, Version 3.0." 30 September 2004 (included with DPLR software distribution).
- F-2. Gordon, Sanford, and McBride, Bonnie J. "Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications, I. Analysis, II. Users Manual and Program Description." NASA Reference Publication 1311, June 1996

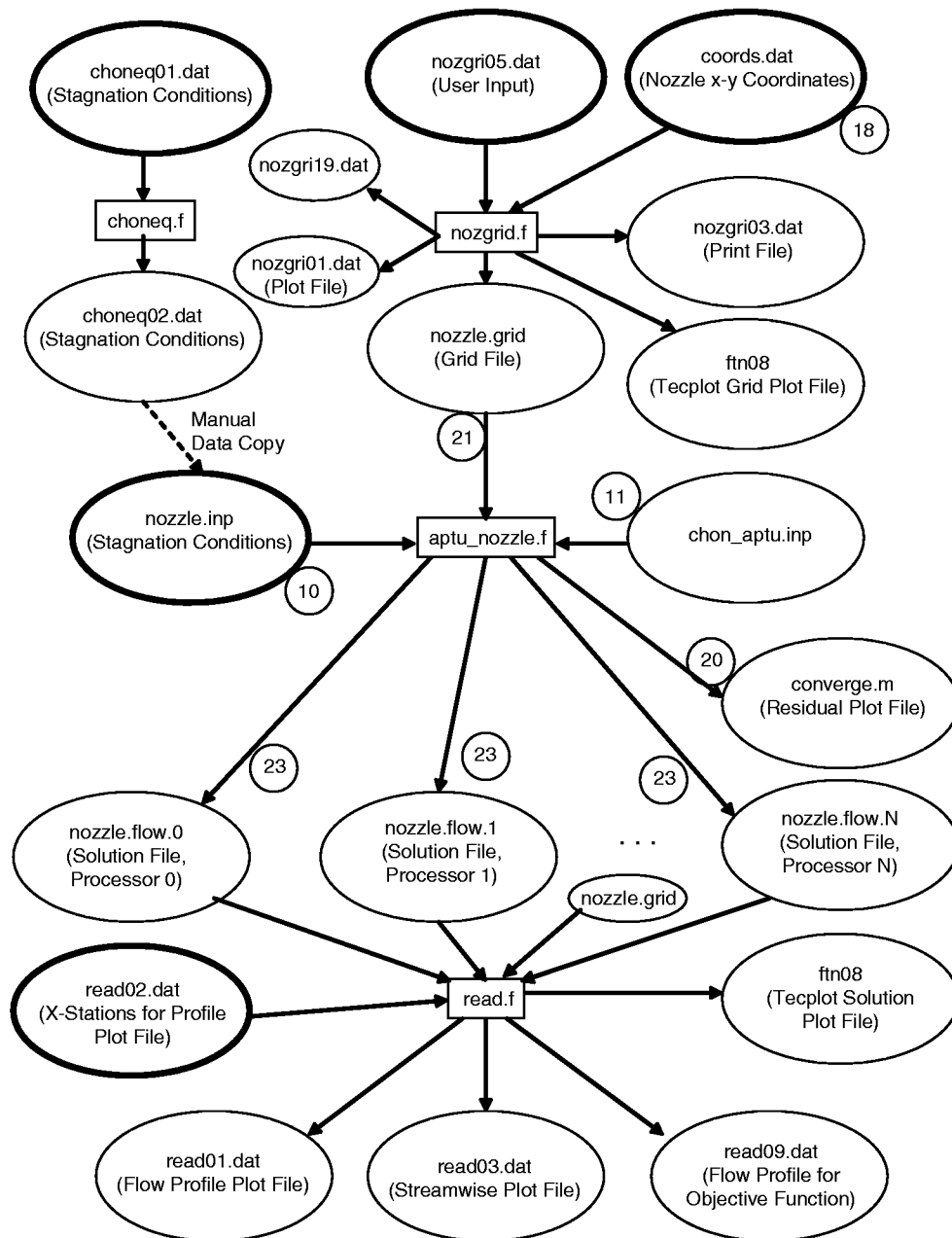


Figure F-1. DPLR Processing as Modified for Design Optimization Procedure

Table F-1. Input to choneq.f on File choneq01.dat

Variable Name	Description
title	Case identification information (format A80)
t	Stagnation temperature, K
rho	Stagnation density, kg/m ³
p	Stagnation pressure, N/m ²
xx1	CO ₂ mole fraction
xx2	H ₂ O mole fraction
xx3	CO mole fraction
xx4	N ₂ mole fraction
xx5	O ₂ mole fraction
xx6	H ₂ mole fraction
xx7	OH mole fraction
xx8	O mole fraction
xx9	H mole fraction
xx10	NO mole fraction
xx11	N mole fraction

Table F-2. Input Description for File nozgrid05.dat

Variable Name	Default Value	Description
il	500	Number of x-grid points
jl	128	Number of y- grid points
inoz	20	Number of grid points added upstream of inlet
nmax	300	Number of passes through grid generator
rk	1.05	Wall stretching factor
fdd	1.	Fractional multiplier for upstream x-step

APPENDIX G

CHARACTERISTIC TRACING PROGRAM (RCFROMMLC.FOR)

To assist in deciding where and how to alter a nozzle contour to reduce a specific anomaly in the exit flow profile, a computer program was deemed necessary to trace characteristics back upstream from the nozzle exit to possible wall points at which the anomaly might have originated. Because the method-of-characteristics (MOC) code of Sivells (CONTUR, Refs. G-1) naturally generates a set of characteristics for the inviscid flow, a new code, RCFROMMLC.FOR, was written to trace characteristics through the characteristic net generated by Sivells' code. The new code turned out to be logically complex. Tracing characteristics requires a complete set of both right- and left-running characteristics (see Fig. G-1 for "right" and "left" notation), but Sivells' code makes only the left-running set available. The program RCFROMMLC.FOR accomplishes the creation of the right characteristics from the left characteristics; hence, the program name. With both sets available, it becomes possible to select points along the radius of the nozzle exit and to trace right and left characteristics back upstream until the tracing process exhausts all of the characteristics. The primary output of the program is a plot file for the characteristic net and the traced characteristics for visual examination.

RCFROMMLC.FOR was developed on a PC under Microsoft Fortran PowerStation 4 and Windows 2000 but should compile easily under Fortran 77 or 90. The user must set the program dimensions as indicated in Table G-1. Execution time is less than one minute on a 1-GHz PC. The program is intended for use with gnuplot. However, memory limitations were encountered using gnuplot on a 250-MB RAM PC. Accordingly, plots were generated on an SGI Octane with 1.4 GB of main memory.

The program first reads a user-prepared data set, RCFROM01.DAT. This file contains the nozzle exit Mach number, a tolerance for locating characteristics intersections, a list of radii at the nozzle exit from which characteristics traces are to originate, and the number of tracings to be accomplished. The recommended tolerance for APTU-sized nozzles is 1 in. This value may depend on nozzle size, but it has not been widely tested. The tracing number is the number of segments of a single characteristic, where each segment would be the result of a reflection at the wall or a virtual reflection at the nozzle centerline. A typical number is three to five. The program also reads the nozzle coordinates from RCFROM02.DAT. The first record must be the number of points on the contour. RCFROM02.DAT could be a copy of the file coords.dat used elsewhere. The final input is a file of left-running characteristics on RCFROM03.DAT. The order and structure of this file is critical. The first characteristic on the file must be the nozzle exit characteristic, and the characteristics must be ordered from downstream to upstream. The first point read on each characteristic must be either on the nozzle centerline or on the right-running characteristic at the throat. The points thereafter must proceed from the centerline to the wall, and the final point should be on or outside of the wall. The specific data read on each record are the x- and y-coordinates of the characteristic point, the local Mach number of the velocity magnitude, and the flow angle, in degrees. The characteristics may come from any valid solution, but the file structure is that from the AEDC working version of Sivells' nozzle design code CONTUR. The specific output file from CONTUR is SIVEL52.DAT. Table G-1 summarizes the input to RCFROMMLC.FOR.

Program operation is described as follows: After the above input process is complete, the input left characteristics are reordered from downstream first to upstream first. The reordering process is complicated by the desire to avoid storing two copies of the characteristics. Next, since the characteristics stop with the beginning of the test rhombus, the triangular region downstream of the last characteristic is filled with left characteristics. The purpose of this filling-in is to allow specification of points along an axis-normal radial line at the nozzle exit, from which characteristic tracings can begin. Next, the points on the contour are located where the left characteristics intersect the nozzle wall. Then, these wall intersection points are used as starting points to construct the right characteristics. The right characteristics are propagated through the flow field (as defined by the Mach number and flow angle along the left characteristics) in the direction defined by the local Mach angle. The propagation process uses a two-step predictor-corrector computation to locate the next point on the right characteristic using the Mach angle at the midpoint. The construction process terminates at the nozzle centerline. Finally, the actual tracing process is accomplished as follows: Beginning with the list of points across the nozzle exit from which characteristics are to be traced (XFIN and YFIN), the left and right characteristics that pass most closely to these points are found. The logic for the identification process is contained in subroutine CHARFIND. The characteristics so identified are referred to in the code comments as "found characteristics." Next, the process is repeated for the upstream ends of the found characteristics, which results then define reflection points either at the wall or at the centerline. The collection of points comprising the traced characteristics is stored in variables XCF and YCF. Finally, the various arrays created by RCFROMLC.FOR are written to plot files intended for use with gnuplot. Command files for gnuplot are included in the software package.

REFERENCES

- G-1. Sivells, J. C. "A Computer Program for the Aerodynamic Design of Axisymmetric and Planar Nozzles for Supersonic and Hypersonic Wind Tunnels." AEDC-TR-78-63 (AD-A062944), December 1978.

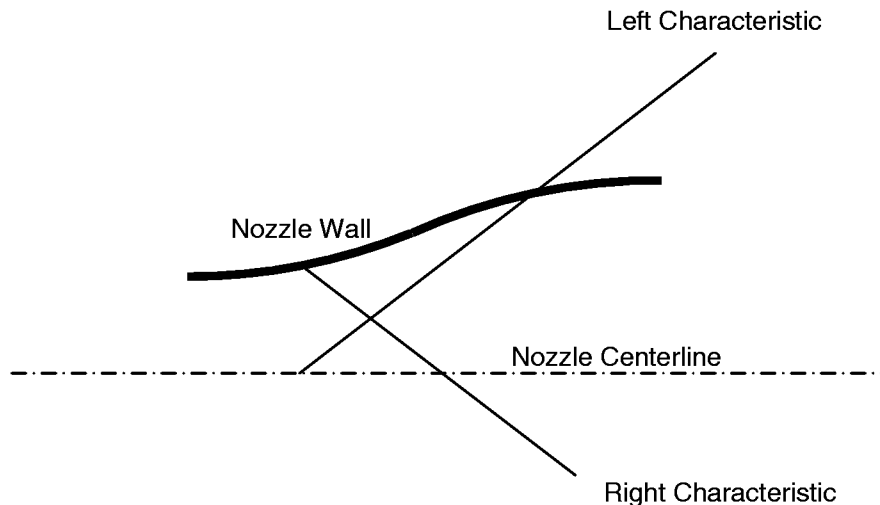


Figure G-1. Definition of Left- and Right-Running Characteristics

Table G-1. Input Description for Program RCFROMLC.FOR

File Name	Variable Name	Typical Value	Description
(PARAMETER statement in source code)	NPMX	1500	Maximum number of points on one characteristic
	NCMX	1500	Maximum number of characteristics in entire flow field
	NCOMX	1500	Maximum number of points on input contour
	NCIMX	1500	Maximum number of points on interpolated contour
	NFMX	100	Maximum number of "found" characteristics (from tracing)
RCFROM01.DAT	TITLE		Case identification (A80)
	MACHCD	> 1	Nozzle exit Mach number
	TOLF	1 in.	Numerical tolerance for locating intersection of left and right characteristics.
	XFIN(I), YFIN(I), I≤NFMX		The x- and y-coordinates on nozzle exit from which characteristics are to be traced upstream (two values per record, terminate table with XFIN < 0)
	NTRACE	3 to 5	Number of characteristic tracings (i.e., number of wall or centerline reflections + 1)
RCFROM02.DAT	NCO		Number of points on contour (first record)
	XCO(I),YCO(I), I≤NCOMX		The x- and y-coordinates of contour (one pair per record)
RCFROM03.DAT			For table of left characteristics, number of characteristics ≤ NCMX, five values per record as follows:
	I, I≤NPMX		Point number on characteristic. I = 0 indicates first record for a new characteristic (last four values are dummy values). I = 1 is nozzle centerline or a point on the right-running throat characteristic.
	XIN		The x-coordinate of point (x = 0 is throat)
	YIN		The y-coordinate of point (y = 0 is nozzle centerline)
	MIN		Mach number at XIN, YIN
	AIN		Flow angle (deg) at XIN, YIN, deg

APPENDIX H

PROGRAM FOR EFFECTIVE CALORICALLY PERFECT GAS MODEL (EFFCPG.FOR)

When it is necessary to use a program that assumes a thermally and calorically perfect gas (TCPG) for a high-speed flow where the actual thermodynamics might violate the assumption, the resulting error can be ameliorated somewhat by the use of an *effective* TCPG. Such a model is nothing more than carefully chosen values of the specific-heat ratio $\gamma = C_p/C_v$ and the gas constant R (as in $P = \rho RT$) that reduce the error. The program EFFCPG.FOR was developed to facilitate such a choice based on the following equations [Eqs. (12) and (14) in AEDC-TR-04-2].

$$\frac{A}{A^*} = \frac{1}{M} \left[\frac{2}{\gamma + 1} \left(1 + \frac{\gamma - 1}{2} M^2 \right) \right]^{\frac{\gamma + 1}{2(\gamma - 1)}}$$

$$\frac{A^*}{A} = \left(\frac{\gamma + 1}{2} \right)^{\frac{\gamma + 1}{2(\gamma - 1)}} \left(\frac{V}{a_0} \right) \left[1 - \frac{\gamma - 1}{2} \left(\frac{V}{a_0} \right)^2 \right]^{\frac{1}{\gamma - 1}}$$

Since these equations are transcendental, they cannot be solved analytically for the desired variables. In the program, they are solved with a direct-search iteration procedure.

To make use of this program, investigators must have available a more appropriate thermodynamic model. Typically, for a high-pressure and high-temperature wind tunnel flow, a thermodynamic model based on equilibrium chemistry (e.g., CEA96, Ref. H-1) will be more accurate than the thermally and calorically perfect assumption. An early step in any nozzle design is to find the approximate area ratio for the nozzle with a specified exit Mach number. If this is done using an equilibrium chemistry program, that program should generate the input information needed for EFFCPG.FOR. Further, the flow at the throat and nozzle exit is, to a good approximation, uniform and parallel. Hence, a program such as CEA96 should give a very accurate estimate of the inviscid area ratio. Use of the specific-heat ratio and gas constant from EFFCPG.FOR in Sivells' design code will nearly guarantee that the inviscid contour will yield the equilibrium speed and Mach number at the nozzle exit. Program input is described in Table H-1.

REFERENCE

- H-1. Gordon, Sanford and McBride, Bonnie J. "Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications, I. Analysis, II. Users Manual and Program Description." NASA Reference Publication 1311, June 1996

Table H-1. Input Description for Program EFFCPG.FOR

Variable Name	Description	Units
AR	Area ratio (i.e., ratio of nozzle exit area to throat area)	(--)
ME	Exit Mach number from equilibrium model	(--)
VE	Exit velocity from equilibrium model	m/s
TT	Reservoir stagnation temperature	K

APPENDIX I NODE LOCATOR PROGRAM (NODELOCS.FOR)

This program locates points along a line and clusters them at either end. The program may be used to distribute correction nodes along a nozzle contour and to cluster nodes at either end (but not both simultaneously) of the distribution range. The clustering is exponential and depends on user-selected increments at the ends of the interval. The program may be used for selecting the location of correction nodes along the nozzle contour.

The program is based on the following: points along a line of length 1 have locations ξ_j , $j = 1, 2, \dots, J + 1$, where $\xi_1 = 0$ and $\xi_{J+1} = 1$. The increment $\Delta\xi_1$ between points 1 and 2 is specified, as is the increment $\Delta\xi_J$ between points J and $J + 1$. The increment variation is to be defined by

$$\Delta\xi_j = \Delta\xi_1 + (\Delta\xi_J - \Delta\xi_1) \left(1 - e^{-a \frac{j-1}{J-j}} \right)$$

with the additional condition that the $\Delta\xi_j$ sum to 1. The program thus numerically solves the following equation for variable a :

$$J\Delta\xi_1 - (\Delta\xi_J - \Delta\xi_1) \sum_{j=1}^J e^{-a \frac{j-1}{J-j}} = 1$$

Note that the last term in the summation is zero but is computationally singular, and therefore its evaluation is avoided in the computer program. The numerical solution is accomplished with a direct-search algorithm.

To apply NODELOCS.FOR, the user prepares an input file as shown in Table I-1, below.

Table I-1. Input Description for Program NODELOCS.FOR

Variable Name	Description
N	Number of node points
DKSI1	Point spacing at lower end of point range (first increment)
DKSIJJ	Point spacing at upper end of point range (last increment)
X1	Coordinate value of first node point
XJ	Coordinate value of last node point

Note the slight notational inconsistency of the variable XJ, which is, in fact, the location of the last node point, not the second to last.

APPENDIX J

SINGLE-NODE PERTURBATION PROGRAM (NOZCOR.FOR)

This program computes a smooth contour perturbation with a single peak at an arbitrary point along the contour. The perturbation has zero value and zero slope at its end points, the ends of the nozzle contour. A parameter is included to control the effective range over which the perturbation is significant. The basic perturbation function y is given by

$$y = A \sin^n \pi x_1$$

The peak location is controlled with the following mapping:

$$x_1 = b(1 - e^{-a\xi})$$

where

$$\xi \equiv x/x_e$$

x_e is the location of the nozzle exit. The mapping function is determined by solving numerically the following relation for a .

$$2e^{-a\xi_t} - e^a - 1 = 0$$

where $\xi_t \equiv x_t/x_e$ and x_t is the location in physical space of the maximum correction.. Once a is determined, the constant b can be computed from

$$b = 1 / (1 - e^{-a})$$

To use the program, the designer prepares the input as described in Table J-1.

Input variable AA is the maximum correction to the contour and is to be applied at station XT. AA may be positive or negative. The value of AA should be small compared to the nozzle radius at the application station. NX, if greater than zero, is the number of equally spaced points on the interval [0,XE], at which correction values are to be computed. If NX is negative, the actual nozzle coordinates are read from another file as described below. NP is the power on the sine function and must be even. As NP is increased, the correction magnitude becomes smaller at locations away from XT. The numerical solution of the mapping, a Newton iteration, is controlled by the variables ERRMAX, ITMAX, and FDX. The user should accept the default values unless convergence fails, in which case $0 < \text{FDX} < 1$ and larger ITMAX values should be tried. If $\text{FDX} < 1$ is used, it may be necessary to increase ITMAX to reach convergence.

For $\text{NX} < 0$, the nozzle coordinates given in Table J-2 are input to NOZCOR.FOR.

If the user chooses to use NOZCOR01.DAT, the correction is added to the input y-coordinate. Otherwise, only the correction itself is computed. A sample correction distribution is shown in Fig. J-1.

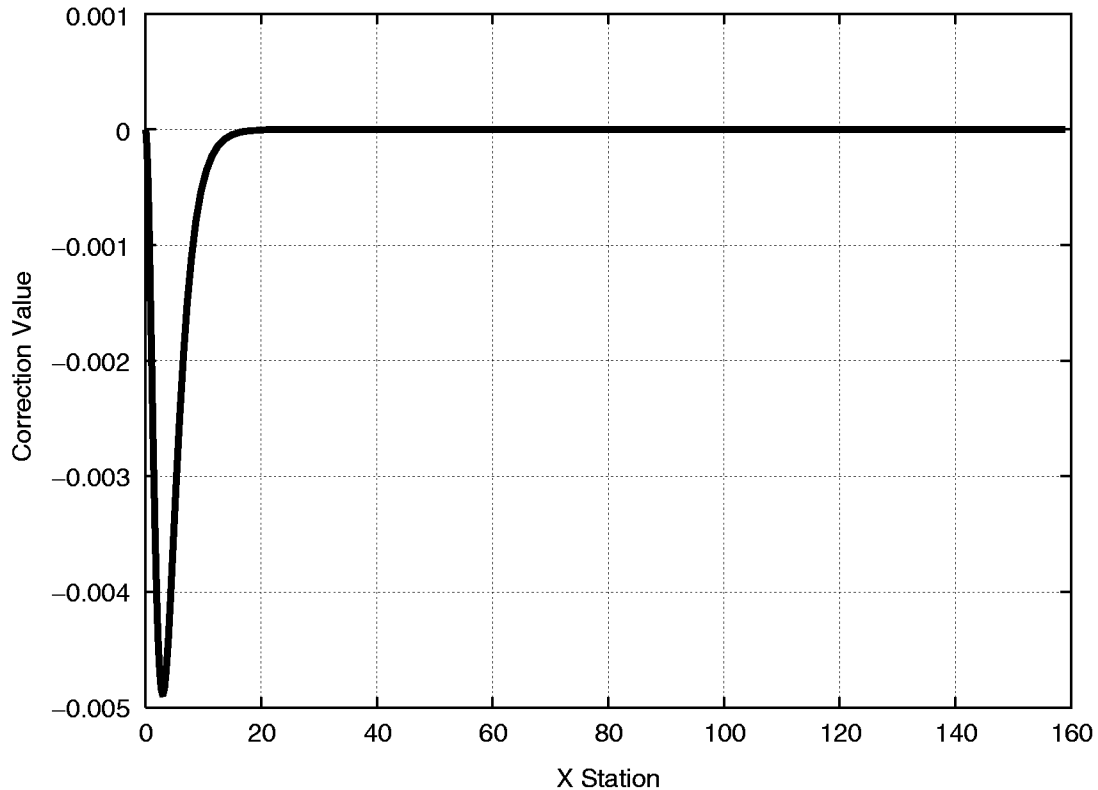


Figure J-1. Example of Contour Correction Based on $\sin^n$ Function

Table J-1. Program Input on File NOZCOR05.DAT

AA	Maximum correction amplitude at point XT (e.g., 0.01)
A	Initial guess at variable a , $\text{EXP}(A \cdot X)$ (e.g., 20)
XE	Distance from throat to nozzle exit
XT	Distance from throat to location of maximum correction
NX	Number of points on plot file (enter < 0 to read coordinate table from unit 1)
ERRMAX	Maximum allowable error for Newton iteration (enter 0 for default, 1.D-6)
ITMAX	Iteration limit for Newton iteration (enter 0 for default, 20)
FDX	Fraction of Newton correction accepted (0,1) (enter 0 for default, 1)
NP	Power on sine function, i.e., $\text{SIN}()^{**NP}$ ($NP \geq 2$ and must be even)

Table J-2. Program Input on File NOZCOR01.DAT:

NX	Number of points on contour
XTBL(I), YTBL(I)	Nozzle coordinates, two values per record, I = 1, NX. XTBL(NX) = XE.

NOMENCLATURE

Symbols

A	Flow area
A^*	Throat flow area
a_0	Stagnation speed of sound
C_p, C_v	Specific heats at constant pressure and volume, respectively
E, F, G	Flux vectors in Navier-Stokes equations
E', F', G'	Flux vector derivatives with respect to design parameters $\partial E/\partial \alpha, \partial F/\partial \alpha, \partial G/\partial \alpha$
f	Relaxation factor for Newton iteration
J	Jacobian matrix, $\partial \epsilon / \partial \alpha = \partial q / \partial \alpha$
M	Mach number
P	Static pressure
Q	Objective function
q	Vector of flow properties used to form objective function or solution vector in Navier-Stokes equations
q^*	Target values of flow properties in objective function
q'	Sensitivity derivative $\partial q / \partial \alpha$
R	Ideal gas constant as in $P = \rho RT$
T	Static temperature
u, v	Axial and axis-normal flow velocity, respectively
V	Flow velocity magnitude
x, y, z	Cartesian coordinates
α	Design parameter vector
γ	Specific heat ratio, C_p/C_v
ϵ	Deviation vector of flow properties from target values
ρ	Static density

Subscripts

<i>equil</i>	Equilibrium value
<i>mean</i>	Mean value

Acronyms

APTU	Aerodynamic and Propulsion Test Unit
CONTUR	Sivells' name for his MOC-based nozzle contour design program
DBA	Design by analysis
DPLR	Data Parallel Line Relaxation
HEAT	High-Enthalpy Ablation Test
LSO	Least-squares optimization
MOC	Method of characteristics
MPI	Message-passing interface
PDE	Partial differential equation
PNS	Parabolized Navier-Stokes
RAM	Random access memory
RSM	Response Surface Methodology
Tcl	Tool Control Language
TCPG	Thermally and calorically perfect gas