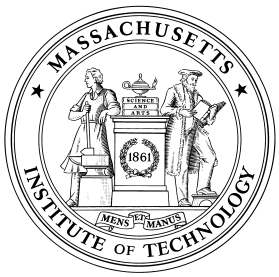
Analysis, Fabrication and Testing of a MEMS-based Micropropulsion System

by

Robert L. Bayt



FDRL TR 99-1

Fluid Dynamics Research Laboratory
Department of Aeronautics and Astronautics
Massachusetts Institue of Technology
Cambridge, MA 02139

June 1999

Abstract

Various trends in the spacecraft industry are driving the development of low-thrust propulsion systems. These may be needed for fine attitude control, or to reduce the mass of the propulsion system through the use of small lightweight components. The nozzle converts the stored energy in a pressurized gas into kinetic energy through an expansion. The nozzle efficiency is characterized by the amount of kinetic energy leaving the nozzle, and is governed by the exit velocity. Because of the increase in viscous losses as scale is reduced, it was feared that high Mach number supersonic flows could not be generated in microdevices. However, a scaling analysis indicates that the reduction in throat area can be offset by an increase in operating pressure to maintain a constant Reynolds number. Therefore, thrust can be decreased by reducing the nozzle scale, and viscous losses mitigated by running at higher chamber pressures.

In order to operate a supersonic nozzle efficiently, the geometry must be contoured to guard against flow separation and reduce the boundary layer thickness at the throat. Deep Reactive Ion Etching enables extruded flow channels of arbitrary in-plane geometry to be created at scales an order of magnitude smaller than conventional machining. These channels are encapsulated by anodically bonding glass to the upper and lower surfaces. Testing indicates that 11.3 milliNewtons of thrust is generated for a nozzle with a 37 micron throat width, 308 microns deep, and a 16.9:1 expansion ratio. The exit velocity was 650 m/s, which corresponds to an exit Mach number of 4.2, and an I_{sp} of 66 seconds. This is 100 m/s higher than previously achieved in a micromachined device and demonstrates that supersonic flows can be generated at this scale.

The performance of the system is increased by electrothermal augmentation. By resistively heating fins present in the chamber, a thruster temperature of $700°C$ has been achieved. This will increase the theoretical I_{sp} to 145 seconds. However, the reduction in Reynolds number with increased chamber temperature causes viscous dissipation to increase and thruster efficiencies to decline. The efficiencies vary with Reynolds number in the same fashion as their unheated counterparts, which confirms that Reynolds number is the governing similarity parameter. The thruster was operated at a temperature of $420°C$, and demonstrated an I_{sp} of 83 seconds. This represents an I_{sp} efficiency of 79% for an 8.25:1 area ratio nozzle. These results suggest that MEMS-based micropropulsion systems offer higher performance at lower mass, when operated at Reynolds numbers above 2500 for both heated and unheated thrusters.

This report is the Ph.D. Thesis of Robert Bayt, Submitted to the Department of Aeronautics and Astronautics, M.I.T. in June, 1999, Supervised by Prof. Kenneth Breuer.

For more information, contact, Prof. K. Breuer: `breuer@mit.edu`

Acknowledgments

It has been remarked that I have set the record for affiliation with the most research groups at MIT. This is primarily due to the diverse nature of Microelectromechanical systems. Without consultation from all of the relevant experts, it would be difficult for microsystems to be developed. This thesis has only been possible, because each of the members of these groups has answered a question of mine over the past four years, and its completion is a tribute to them. These labs are: the Microsystem Technology Lab, particularly the members of the Schmidt Group, the Fluid Dynamics Research Lab, the Space System Lab, the Gas Turbine Lab, and the Active Materials and Structures Lab.

My sincerest thanks go to the members of my committee who were always up to speed on my work, and offered perfect guidance along the way. This is especially true for my advisor, Kenny Breuer. He is a testament to the MIT work ethic, and embodies the inquisitive nature needed in a research environment. I am proud to call him a mentor and a friend. Similarly, Marty Schmidt and Manuel Martinez-Sanchez have inspired me to pursue problems in their respective fields, MEMS and advanced propulsion technologies.

This work was only possible through the financial support of the Graduate Student Researchers Program at Goddard Spaceflight Center. This was originally administered by Dr. Gerald Soffen, who motivated me to attend MIT, and has always provided sound advice for my career. Dennis Asato was the center technical advisor, and has shown nothing but enthusiasm for this work. In addition, the Microdevices Lab at the Jet Propulsion Laboratory provided the initial funding to begin the research into MEMS-based propulsion through contract 960444 monitored by Bill Tang. Finally, the Air Force Office of Scientific Research has provided the necessary funds to allow the completion of this work through under contract #F4920-97-1-0526 monitored by Dr. Thomas Beutner.

This work was an outgrowth of a preliminary study performed by Adam London, and is referenced frequently throughout this thesis. In addition to being a predecessor, Adam has been a friend and confidant that always knew how to look at the big picture a little differently. Newton said he could see farther than others because he stood on the shoulders of giants. Too me, Adam is a giant.

Dr. Arturo Ayon has been the research force behind the deep reactive ion etcher, and has brought this technology to the performance levels we are enjoying today. He has taught me nearly everything I know about microfabrication technology. In addition, he has shown me the true meaning of simpatico (as well as Chianti!). In this same vain, Kurt Broderick has shown me so many tricks and nuances of the Technology Research laboratory, he has probably cut six months from my development time. All of the members of the Schmidt group provided insight and advice in fabrication. Not only that, they are a fun group to know. Thanks to Joel, Samara, Reza, CC, Jo-ey, Becky, Zony, and Christine.

The computational work presented here was based on a prior code written by Ed Piekos. It was such a fine work before I got my hands on it. I turned his '99 'Vette into a '76 Gremlin. But, my thanks to him, Ali Merchant, and Tolulope Okasanya for their insight into CFD, and to really opening my eyes to the computational world.

The experimental work could not have been complete without the thrust results. These were made possible by the father of MEMS-based propulsion, Siegfried Janson and his colleagues at the Aerospace Corporation. In addition, I would not be done with out the help of Vadim Khayms, who was my quals buddy as well as the builder of the MIT test stand. His assistance in making the final measurements was invaluable.

The students of the Gas Turbine lab, particularly Amit Mehra and Luc Frechette, were instrumental in the design of the heater, as well as, the side projects involving FLUENT modeling. Their friendship, as well as Amit's supply of fine Cigars and Cognac, added to the enjoyment of this work.

In addition to the technical support, this work would not have been possible without the enduring love and prayers I have received from home. My wife, Heather, has been my continual source of inspiration. She was genuinely interested in my work, and has always held my goals as her goals. Her strength has grown through these tests, and she is now ready to take on the challenges of motherhood, for which she is ideal.

My family has always believed in me, and it is because of them I achieved my highest goal. It is now because of them I will try to raise the bar one notch higher. My father has been the perfect role model, exemplifying honesty, integrity and strength. My mother has always embodied the spirit and strength to make sense of everyday life. My brother, David, is my source of pride, for he is beginning to spread his wings,

and set his sights toward building a better world. In addition, my In-Laws have always held an interest in my work, and have been very supportive throughout. Thank you to Ron, Suzanne, Debra, Todd and Lynda.

Having been at MIT for so long, I have had many officemates that have enjoyed the excitement and doldrums of each day. Thanks to Tim, Cyrus, Eugene, Karsten, and Chris Protz.

Boston has truly been a better place because of my friends Aaron, Ann, John R., Margee, Alex, Maggie, Kamyar, Mitch, Homero, Jake, John H., Kevin, Suzanne, Paulo, Marcella, James, Tatsuo, Darrel, Karen, Greg^2, Angie, Alex B. and Ashok. My late night ramblings with my friend D.J. Orr have really helped me put a lot of the engineering world into perspective. My lunches and outings with my good friend, Robin Smith, provided me with a reality check not always found in the corridors of MIT.

Last, and above all, I must give my thanks to God, from whom all blessing flow. It has only been through faith that I have known which direction to travel. May my future endeavors always be a testament to His glory.

When I left Indiana, people asked me why I wanted to go to Boston. My response was, "I have always wanted to go to MIT" It has only been because of these people, and not the brick and mortar on this fine piece of real estate, that made this the right choice.

Contents

List of Figures

List of Tables

Nomenclature

Roman

$\bar{C}$	Mean molecular speed
c_d	Coefficient of discharge
c_p	Coefficient of specific heat at constant pressure
D	Local width
E	Total Energy $\rho[e + \frac{1}{2}(u^2 + v^2)]$
F	Inviscid fluxes
F_t	Thrust Force
G	Viscous fluxes
h_o	Etch depth
$\mathrm{I_{sp}}$	Specific impulse
I_{bit}	Impulse bit
Kn	Knudsen number
J	Moment of inertia
k_w	Wall thermal conductivity
$\dot{m}$	Mass flow
N_d	Number density
Pr	Prandtl Number
P	Pressure
q	Rate of heat loss
Re	Reynolds number
T	Temperature
$\dot{w_p}$	Propellant used
u	x-component Gas velocity
v	y-component Gas velocity

Greek

ΔV	Orbital Velocity Change
δ^*	Displacement thickness
ϵ	Expansion Ratio
$\eta_{I_{sp}}$	I_{sp} efficiency= ${I_{sp}}_{\text{measured}}/{I_{sp}}_{\text{Ideal}}$
η_{thrust}	Thrust efficiency= ${Thrust}_{\text{measured}}/{Thrust}_{\text{Ideal}}$
γ	Ratio of specific heats
λ	Mean free path
ρ	Gas density
τ	Shear stress
θ	Momentum Thickness
θ_t	Spacecraft pointing deadband

Subscripts

e, exit	Exit plane of nozzle
edge	Boundary layer edge as defined by $\delta_{99\%}$
eng	Engine
f	fluid
i	inlet
o, ∞	Chamber conditions
s/c	Spacecraft
t, thr	Throat location
w	Wall location

Acroynms

DRIE	Deep Reactive Ion Etching
FEEP	Field Emission Electric Propulsion
JPL	Jet Propulsion Laboratory
MEMS	Microelectromechanical Systmes
PPT	Pulsed Plasma Thruster
STS	Surface Technology Systems Etcher

Chapter 1

Introduction

1.1 The Role of Microspacecraft

In this time of decreasing federal budgets, the focus of NASA, and the aerospace industry at large, is to reduce satellite life-cycle costs while still delivering a spacecraft with the capability of performing useful science or a commercial service. With their New Millennium program, the Jet Propulsion Laboratory (JPL) is developing technology for the next generation of spacecraft that will encompass all of NASA's directives: Faster, Better, Cheaper.

A wide range of resources from program management techniques to advanced technologies are being focused on achieving each of these directives. One of the means to this end is the development of microspacecraft [1]. Such a satellite may contain only one instrument, but this reduction in complexity will lower costs, by facilitating systems integration. In addition, the small sizes allows the selection of a smaller, less expensive launch vehicle, or the integration of multiple satellites per vehicle.

JPL has examined a number of mission scenarios that are conducive to miniaturization, which are described by Mueller [2]. These missions are a rendezvous with the near Earth asteroid, Vesta, a Europa orbiter, an Earth observing cluster, and a deep-space interferometer. In addition to these missions, London [3] performed a similar analysis for a future generation global positioning system, a Mars observer, and a low Earth orbit communication system. This indicates that there are a wealth of opportunities in which microspacecraft can perform useful science, while reducing cost and development time.

The aerospace community envisions Microelectromechanical systems (MEMS) as a means for attaining a high degree of functionality at a much smaller scale. MEMS technologies are sensors and actuators which are fabricated with the use of the micromachining techniques developed by the microelectronics industry. As defined by Mueller [2], first generation microspacecraft will have a mass of 20 kg, a baseline dimension of 30 cm, and consume 20 Watts of power. It will begin to integrate MEMS components over the next 5 years to demonstrate the proof-of-concept. The second generation microspacecraft (1-5 kg), will then begin to integrate large numbers of MEMS subsystems. This ultimately leads to a third generation (< 1 kg), that is a completely micromachined device with all systems integrated during fabrication. The first two generations will illuminate the feasibility issues of the third generation.

1.2 The Role of Micropropulsion

One implication of microspacecraft is that they will need small systems. Due to the limited capabilities of the early launch vehicles, the first satellites were inherently small, and due to the lack of truly miniature systems, possessed limited functionality. Explorer I (1958) was only 15 kg, and Vanguard I (1958) was only 1.5 kg. These were the first true microsatellites. As technology progressed, satellites developed pointing requirements and the need to maintain precise orbits. On-board propulsion systems were added to make-up for aerodynamic drag and solar pressure disturbances, as well as counteract gravity-well distortions due to the oblateness of the Earth. Since the satellites were still small due to the launch vehicle capacity, the first micropropulsion systems were developed. Their function was two-fold: provide slewing capability to adapt

to different pointing requirements, and maintain the attitude within a deadband (angular position limit) to meet pointing tolerances. Sutherland and Maes [4] reviewed the state-of-the-art for different propulsion architectures in 1966. They established system criteria which can be used to judge applicability of a given architecture. These are: Total Impulse, Thrust Level and Control, Life and Reliability, State of Development, Duty Cycle, Environment, Response or Impulse bit, Exhaust Products, and Cost. This lead to an establishment of the domains of applicability of for propulsion architectures, which was later revised and updated by London [3] to include new or improved technologies.

Sutherland and Maes further stress the need to generate not only low thrust, but to minimize the impulse bit (I_{bit}). The I_{bit} is the minimum impulse obtained once the thruster is given the command to fire. It is the integrated thrust over the fastest valve cycle time. Sutherland et al. show that the propellant used per thruster firing for maintaining attitude about a deadband can be computed for the undisturbed rotation of a constant mass vehicle about one axis as

$$\dot{W}_p = \frac{l(\Delta I_{bit})^2}{4JI_{sp}\Theta_t} \quad , \tag{1.1}$$

where J is the moment of inertia, Θ_t is the angular position limit, and l is the moment arm. Thus, an improvement by decreasing I_{bit} is more important than an improvement in increased I_{sp} for the purpose of attitude control.

Using the JPL design reference missions as baseline requirements, this work will focus on the miniaturization of the on-board propulsion and the fabrication of these devices through MEMS technologies. The first generation microspacecraft design requirements are a minimum thrust of 4.65 mN for slewing, and an I_{bit} of 0.013 mNs for maintaining attitude [2].

1.2.1 Propulsion Architectures

With the utility of micropropulsion established, the ideal architecture for meeting the requirements should be assessed.While Sutherland and Maes reviewed the current state-of-the-art for 1966, that has been updated by Mueller and London to include new architectures, such as Field Emission Thrusters (FEEP), and improvements in technology, which include MEMS-fabricated devices. Initial studies by London and Khayms [5] identified the scaling issues with micropropulsion technologies. London laid out the domains of applicability for different propulsion architectures, which is reproduced in figure 1.1. A particular architecture is dominant for a given mission scenario because it results in the lowest mass and power system for the control authority desired. The metrics are ΔV, which indicates the amount of usage a thruster will get, and thrust, which is a factor in how quickly the actuation will occur.

Mueller and London's works establish the capabilities of current technologies, as well as speculating at which technologies are suited for micropropulsion, and the scenarios in which they should be used. London's investigation of scaling issues to the various propulsion architectures does not reveal any significant drawbacks that would preclude the implementation of a particular propulsion concept. However, as the domains of applicability chart implies, the optimal system is a strong function of mission requirements. The goal is to draw from these studies which systems are best suited for microfabrication.

Scaling of bipropellant engines has limited their use to thrust classes greater than 1 N due to loss in combustion efficiency from poor mixing. Monopropellants are promising for increasing the I_{sp}, but there are materials compatibility as well as catalyst integration issues that need to be investigated for implementation in a microfabricated system.

There are current research programs investigating Pulsed Plasma thrusters at NASA Lewis, which might provide the lowest I_{bit} and reasonable thrust for slewing, but according to Mueller, "Current PPT designs are...too large for microspacecraft attitude control." There is much research into the FEEP, but their specific impulse is too high, and efficiency too low for current applications. Scaling to a lower I_{sp} has yet to be demonstrated due to the high extraction voltages required. The power conditioning electronics is a significant portion of the system mass, and as Mueller describes in his design reference mission analysis, "Even neglecting tank masses for the FEEP system, the cold-gas system will still be lighter than the FEEP system by about 0.4 kg (although volume requirements will be higher due to the large tank volume.)" Finally,

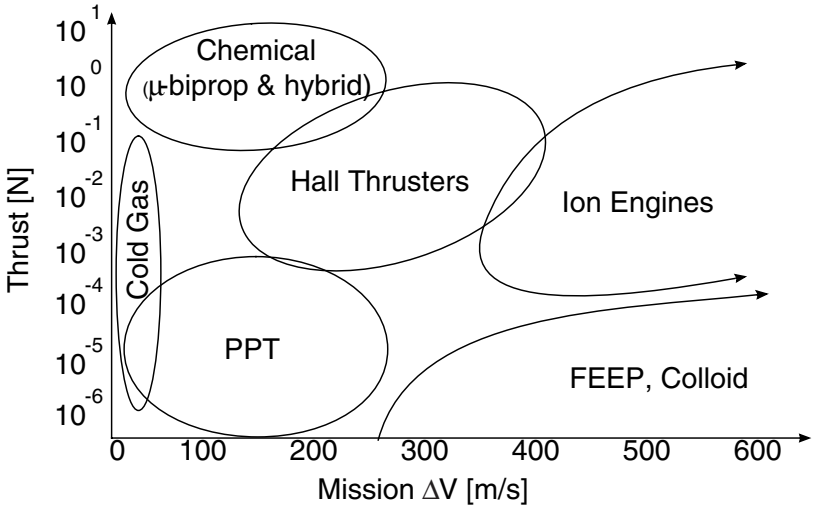


Figure 1.1: Domains of applicability of various propulsion architectures. This shows which architectures minimize mass and power consumption for a given set of mission requirements. Reproduced from London [3]

colloid thrusters are being investigated at MIT as on-board propulsion, and offer many of the advantages of a FEEP, at a lower mass.

1.3 Motivation for a Microfabricated Pressurized Propulsion System

Of all of the possible architectures, it was deemed prudent to begin with the miniaturization of the cold-gas system to demonstrate fabrication and operation. From a review of other research programs, it was apparent that a truly micron-sized thruster, which pushes the limit of the fabrication technology, had yet to be developed. The cold-gas system offers a low I_{bit} in addition to the minimum thrust for slewing requirements. The major drawback has been valve leakage due to contamination, which according to Sutherland is traceable to propellant tank fabrication. But when microfabricated, the system cleanliness may also improve. The cold-gas system will serve as a forerunner to the microresistojet, which could increase the performance with very little additional power due to the low mass flow rates. The micronozzle is an essential component of many propulsion architectures, and its success will lead the way for more advanced chemical systems. Thus, with many of the other technologies currently under investigation, the micronozzle, which is the heart of the cold-gas system, was chosen as the focus of this research. And, its simplicity leads to short development times, and possible implementation into near-term projects.

There are advantages to developing a microfabricated nozzle. Traditional pressurized propulsion systems suffer from high viscous losses due to the low chamber pressure required to obtain low thrust within the limitations of a conventionally-machined system. A majority of current research focuses on rarefied flow due to the low density nature of a low-thrust pressure fed system.

Microfabricated nozzles allow the pressurized propulsion system to run at higher chamber pressures while maintaining low thrust through a reduction in scale. Also, microfabrication offers the capability of on-chip integration of microsystem components such as tanks, lines, valves, and nozzles.

Recent advances in bulk micromachining permit the anisotropic etching of silicon features with high aspect ratios. Deep reactive ion etching (DRIE) offers the nozzle designer the flexibility of a contoured expansion and the advantage of an extruded shape (high aspect ratio nozzles minimize 3-D effects). In

addition, the scale of achievable nozzles are an order of magnitude smaller than previously demonstrated.

The factors that drive the performance of micronozzles is poorly understood. As with all MEMS devices, the operation and performance is coupled with the means by which they are fabricated. With the introduction of a new fabrication technique, such as DRIE, its advantages and drawbacks are to be addressed. This research is driven by the need to evaluate the applicable nozzle scale and operating conditions to maximize nozzle efficiency, and deliver a propulsion system component which can be integrated with a standard propellant delivery system for use on a spacecraft of any scale.

1.4 Previous Micronozzle Investigations

Sutherland and Maes recognized the utility of achieving a small I_{bit}, as well as low thrust, with a cold-gas system. However, for "mechanical and reliability factors", they suggested that nozzles having throat diameters less than 250 microns were undesirable. Thus, to obtain these small thrusts, were forced to run them at chamber pressures of 0.1 psia. This results in throat Reynolds numbers of approximately 30 and I_{sp} efficiencies of 40%. To this point, there have been several investigations of mass flow and thrust efficiency for nozzles at small scales. Notably, Massier et al. [6] investigated it over the range of Reynolds numbers 650 to 350,000, and then a year later Kuluva and Hosack [7] expanded these measurements to Reynolds numbers as low as 20. Both papers investigated the variation in the coefficient of discharge with Reynolds number. Kuluva measured C_D as low as 40% due to the increased blockage from boundary layer displacement at a Reynolds number of 20.

However, the science of nozzle flow was advanced when Rothe [8] made E-beam measurements of the viscous flow within the nozzle. These nozzles had 2.5 and 5 mm throat diameters and were run at chamber pressures of 0.15 psia. This represents a range of Reynolds numbers 55-550 as computed in equation 2.4 of this work. The E-beam measurements allowed Rothe to determine temperature and density profiles along the centerline of the nozzle axis as well as along radii at different cross-sections. The profiles revealed that as the Reynolds number decreased, the effective expansion of the nozzle also decreased. This was realized as an increase in the exit temperature and pressure of the gas, and a decrease in exit Mach number. At Reynolds numbers below 150, the exit temperature was in fact increasing along the channel due to viscous shear forces causing the thermalization of flow energy. At the lowest case (Re=55), the flow has made a shockless transition to subsonic flow by the exit of the nozzle. These results were confirmed numerically by Rae [9], who applied the slender channel approximation to the Navier-Stokes equations, and discovered similar profiles at the nozzle cross-sections. However, each of the nozzles are driven to low Reynolds number operating conditions, as well as highly rarefied flows, due to their inability to fabricate contoured geometries at micron scales.

To reduce the thrust of a nozzle the exit area is to be reduced, but the throat-to-exit area ratio must be maintained to maximize I_{sp}. Grisnik et al. [10] used this to an advantage for a series of four nozzles they tested with throats ranging from 654-711 microns in diameter. These nozzles produced thrusts from 4.59 mN to 45 mN over a range of Reynolds number 1000-9000. The nozzles maintained 85% I_{sp} efficiency at Reynolds numbers above 3000, but dropped quickly to 70% when the Reynolds number was reduced to 500. The same conclusions can be drawn from Rothe's experiment. The effective area ratio is reduced due to the viscous stresses and boundary layer blockage present at the low Reynolds numbers. The results presented by Rothe and Grisnik are for axisymmetric nozzles, which will differ from the current research involving 2-D nozzles. The trends, however, should be similar.

In addition to previous experimental tests, there have been numerical investigations into the flow within a low-Reynolds number nozzle. Grisnik et al. used a two-dimensional kinetic model to predict efficiencies. This code gave very ambiguous results. It predicted an increase in I_{sp} efficiency with decrease in Reynolds number. However, the code was developed in 1984, and had no validation at the operating Reynolds number. They did however provide displacement thicknesses as a function of Reynolds number, that seemed to correspond with that predicted in this work. This allowed an effective area ratio and I_{sp} efficiency to be predicted, which matches well with that found in Chapter 3.

Kim [11] performed a finite volume calculation with similar boundary conditions to that presented in Chapter 3. This major difference was that it was an axisymmetric calculation as compared to the 2-D calculation presented here. He discovered that a 30 degree expansion angle outperformed the 20 degree

angle due to the shorter development length for the same area ratio for his Reynolds number of interest.

Zelesnik [12] performed a DSMC calculation of the nozzles used in the Grisnik experiment. A Navier-Stokes simulation was used to predict the state variables just past the throat, and this served as an inlet condition to the DSMC. Though the Reynolds number was 117, the low chamber pressure allowed the Knudsen number to reach 0.2 at the exit (10 times that found in the present work). Similarly, Ivanov et al. [13] performed DSMC calculations which produced results similar to the N-S simulation presented in this work. While their results overpredict the efficiency by at least 8% compared to the measurements presented here, they noted a performance loss due to the unmodeled plume.

Based on Grisnik's results, smaller scales and higher chamber pressures will provide higher Reynolds number operation for the same low thrust level. This is consistent with the scaling arguments provided in Chapter 2. Though conventional machining such as electro-discharge machining has produced nozzles with throat widths of order 100 microns, the ability to maintain a well defined geometry has yet to be demonstrated. Therefore, microfabrication techniques, which were inspired by the success of inkjet printhead nozzles, were implemented by Janson [14] to demonstrate batch-fabricated microthrusters.

These nozzles, the fabrication of which is detailed in Chapter 4, were anisotropically etched along silicon planes to yield a square nozzle with 32.5° divergence angle. The efficiency of these nozzles is low, due to the limited geometries available through this etch technique. Such devices suffer from separation, high flow divergence losses, and thick boundary layers at the throat. Thrust tests indicated that 56 seconds of I_{sp} were achieved for thrusts of 13 mN. While exhibiting a mediocre performance, it demonstrated a very important proof-of-concept, which needs to be optimized with improved etching techniques.

The utility of these nozzles is demonstrated in their implementation into digital micropropulsion systems as well as JPL's vaporizing liquid microthruster. The digital micropropulsion [15] is based on a thin-walled membrane separating the propellant from the anisotropically etched nozzle. The propellant is ignited and the membrane is burst to yield an impulse of thrust. By batch-fabricating an array of 1000 nozzles, a valveless thruster system can be created for low duty-cycle missions. No leaks other than diffusion will be possible. Currently, only limited arrays of 10 thrusters have been achieved, and the small actuation time causes a large fraction of the propellant to leave the thruster unburned.

Janson [16] has recently introduced a well insulated version of a thin-film heater for fabricating a microresistojet. The polysilicon resistor, deposited on a silicon substrate, is undercut through an anisotropic silicon etch, such as described in Chapter 4. This defines the chamber as well as the nozzles, and leaves the heater suspended in the middle of the cavity such that the gas can flow across the upper and lower surface. This doubles the surface area of a conventional thin-filmed resistor, as well as reduces the thermal gradients across the chamber. However, it will also minimize the conduction to the wall. If the wall remains cool, then it will serve as a heat sink for the flow during thruster operation.

The vaporizing liquid microthruster [17] will utilize ammonia passed through a heater to generate a higher performance, higher density propulsion system. The heater is a thin film metal deposited on a silicon substrate that is resistively heated. The heater is bonded into a plenum through which ammonia flows, and is vaporized. As currently designed, it implements the same nozzle that Janson has demonstrated, which can be significantly improved with the nozzle etch techniques proposed in this research. In addition, the heater effectiveness will suffer at high flow rates due to the limited surface area.

These concepts seek to improve the performance of the thruster through an increase in chamber energy. For the digital micropropulsion, a higher energy propellant is used, and for the vaporizing liquid microthruster, the propellant is resistively heated. The research presented here demonstrates a similar improvement with a resistively heated gas, but the heat transfer is greatly enhanced through the increased surface area of a microfabricated fin array.

1.5 Objectives

The primary objective of this research is to assess the feasibility of fabricating and operating a MEMS-based thruster. Initial studies are performed to analyze the trade-offs with a reduction in scale, and assess the limitations of microfabrication. These limits are used as a guide to determining the performance of the thruster at various operating conditions. Ultimately, nozzles are designed to operate at the highest possible

efficiency for the thrust levels of interest. To meet this goal, intermediate objectives in the operation of the propulsion system and the microfabrication of nozzles are established.

The intermediate propulsion objective is to perform numerical simulations of the flow through 2-D contoured nozzles to benchmark the efficiency at each operating condition. The efficiency is decreased due to the viscous losses inherent to the small scale of this system. This work analyzes flows in the continuum regime due to the high pressure nature of the system. This facilitates modeling by eliminating the need for a direct simulation method. The simulations characterize the viscous losses at each chamber pressure setting. This will be used to determine boundary layer properties for each operating condition.

The performance is to be evaluated through mass flow and thrust tests to determine the scaling effects on I_{sp}, as well as the impact of the limitations of the fabrication. These tests will be used to gauge the exit Mach number and verify supersonic flow has been achieved. Deviations between the model predictions and test results are to be analyzed and quantified. Models are developed to investigate the influence of surface roughness and endwall boundary layers on performance.

Finally, nozzle performance is to be increased by resistively heating the gas in the settling chamber. An integrated microheater, fabricated in parallel with the nozzle and settling chamber, will increase performance and achieve a high efficiency heating of the flow. A one-dimensional heat transfer model is developed to optimize heater geometry.

The microfabrication objectives are established in support of the primary propulsion objectives. The nozzle and plenum are etched into silicon by DRIE. The process parameters, which govern the DRIE, must be optimized to produce the geometry of interest. The etch conditions are established to achieve the highest anisotropy with lowest surface roughness for a high aspect ratio nozzle. Finally, a method for creating fins in the thruster chamber for increased heat transfer, as well as heat dissipation is to be developed.

This work addresses the process undertaken in the achievement of these objectives. Chapter 2 outlines the performance advantages of micropropulsion systems, and establishes the baseline geometry. Chapter 3 presents the numerical analysis of the flow in nozzles of this scale, including trade-offs in design. Chapter 4 details the process development that resulted in the fabrication of the micronozzles. The nozzle thrust and mass flow tests, as well as the performance analysis, is presented in Chapter 5. Chapter 6 details the modeling, fabrication, and testing of an electothermally augmented thruster.

The achievement of these objectives will answer the following fundamental questions about MEMS-based Micropropulsion systems:

- How small can the micropropulsion systems be fabricated?
- What are the fabrication limitations?
- How are they to be packaged and operated? What is the flow condition at each operating point?
- What are the governing similarity parameters? How do they effect efficiency?
- Can the thruster performance be increased substantially through resistive heating?

1.6 Contributions

The analysis and results described herein make the following contributions to the field of propulsion technology:

- Supersonic flow is achieved in flow channels at a scale an order of magnitude smaller than previously documented.
- A higher I_{sp} is achieved for this thrust class cold gas propulsion system than previously attained for a batch-fabricated, or conventionally-machined device.
- 2-D computational modeling combined with experimental testing has confirmed that the Reynolds number is the governing similarity parameter for nozzle flows in MEMS devices.

- Numerical analysis is used to determine the development of endwall boundary layers in an accelerating flow, and their impact on nozzle performance.
- The effects of surface geometry and roughness on nozzle performance is characterized.

The following contributions are made to the field of MEMS technology:

- An effective method for etching features with variable geometry using DRIE is developed and trade-offs highlighted. Surface Roughness and features characteristic of this process are documented.
- A means for high-pressure gas injection into an integrated nozzle plenum system is developed.
- A high efficiency method for heating the plenum and containing the energy in the flow is developed and fabricated in parallel with the nozzle to increase thruster I_{sp}.
- A heater is developed that operates at the silicon intrinsic point to mitigate thermal runaway present in low-Reynolds number gas flows.

1.7 Summary

The utility of the cold-gas system has been recognized since the early sixties due to its reliability and ability to achieve a small I_{bit}. However, to achieve low thrust, low Reynolds numbers ensued. Advances in microfabrication have allowed smaller features to be created which mitigate the low Reynolds number viscous losses through the use of higher pressure. Due to advancements in MEMS technology, the cold-gas system is once again looked at as the near-term solution for on-board micropropulsion. With the first generation MEMS thruster already demonstrated, the stage is set for advanced etching techniques and improved nozzle contours to boost the performance of the next generation MEMS-based propulsion systems.

Chapter 2

Scaling Analysis and the Baseline Microthruster

As mentioned in the introduction, there are several reasons to move toward smaller components in spacecraft systems. They can be broadly grouped into two categories: scale and performance. These represent the direct implications of making an object smaller, it has less mass, and the indirect implications, enhanced properties that become accentuated as scale is reduced. This chapter will explore each of the classes and show the benefits and drawbacks to each.

2.1 Direct Effects of a Reduction in Scale

As the scale of a component is reduced, if it is fabricated from a material of similar density, its mass is reduced. This is stating the obvious, but it accomplishes one of the goals of spacecraft design, mass reduction. This is required to meet launch vehicle constraints, as well as, reduce on-board propulsion system or upper stage mass requirements.

However, smaller mass implies quicker dynamic and thermal response times. An object under loading will accelerate more quickly if it has a smaller mass. This implies fast acting microvalves are possible due to the low mass of the poppet. Heaters can come to temperature more quickly since the thermal mass is less. This reduces the transient of a given architecture, which in the context of a propulsion system will reduce the minimum impulse bit. For a cold-gas thruster this is a function of the valve response time and the thrust of the nozzle. For a resistojet, the heater may be turned on first and allowed to come to temperature, which reduces efficiency, or brought up to temperature while firing, which decreases the I_{sp}. Finer attitude adjustments are achieved by lowering the I_{bit} of the propulsion system.

2.2 Indirect Effects - Performance Variability with Scale

There are several system performance drivers that lead to the selection of smaller scale components. In most cases, these are driven by the "cube-square" law. In other words, properties that are a function of the area of interaction decrease slower than those that are a function of volume. This is formulated as

$$\frac{f(area)}{f(volume)} \propto \frac{L^2}{L^3} \propto \frac{1}{L} \quad . \tag{2.1}$$

Thus, as the length scale (L) is reduced the property ratio increases.

An example is the power to mass ratio of a spacecraft. The power is collected as a function of the area exposed to the solar flux, and yet the mass is a function of the volume of the spacecraft. By applying the principle of equation 2.1, the power to mass ratio increases as the scale decreases, as is shown by

$$\frac{P}{m} \propto \frac{\eta_{array} P_o L^2}{\rho_{s/c} L^3} \propto \frac{1}{L} \quad , \tag{2.2}$$

where η_{array} is the solar array efficiency, $\rho_{s/c}$ is the spacecraft average density, P_o is the solar flux, and L is the scale of the array collection area or spacecraft baseline dimension. An increase in power to mass ratio implies an increase in functionality of the spacecraft. Since the power drawn from system components will, in general, be in proportion to their mass, the increase in power to mass permits either an increase in the power requirements of an instrument or the number of instruments.

Specific to the propulsion system, the thrust-to-mass ratio is of interest. This determines the effectiveness of the system as an actuator. When considering the engine, which includes the nozzle and chamber, the thrust-to-mass ratio can be formulated as

$$\frac{F_t}{m_{eng}} \propto \frac{P_c L^2}{P_c L^3} \propto \frac{1}{L} \quad , \tag{2.3}$$

where P_c is the chamber pressure, and L is a characteristic dimension, such as the throat width. This is the same trend as discovered in equation (2.2), as are all systems which obey the "cube-square" law. However, as London [3] described, the engine mass is about 1% of the total on-board propulsion system mass at these small scales. The dominant mass comes from the propellant and the tanks, due to high pressure storage. London showed that the thrust-to-mass ratio of the tank is independent of scale, and is proportional to the inverse of the tank pressure. Thus, a reduction in scale does not significantly increase the thrust-to-mass ratio of the entire propulsion system as long as the tank pressure and propellant mass remain constant.

The focus now shifts to increasing the performance of the system. If the I_{sp} increases, the amount of propellant required for a given mission decreases, and a direct reduction of system mass occurs. If the nozzle-plenum system scale is reduced, lower thrust (which is required for these applications) is achieved. By shrinking the scale, higher chamber pressures can be used to achieve the same low thrust. The frictional losses are governed by the Reynolds number which is defined as

$$Re = \frac{\rho_t v_t D_t}{\mu_t} = \frac{\dot{m}}{\mu h_o} \quad , \tag{2.4}$$

where ρ is the chamber density (proportional to chamber pressure), v is the velocity, D is the throat width, and μ is the viscosity, all defined at the throat condition. This, in turn, can be defined in terms of the mass flow rate, and the nozzle height, h_o (for an extruded nozzle).

The thrust is proportional to the mass flow rate multiplied by the exit velocity. The exit velocity is ideally set by the expansion ratio of the nozzle. But, as viscous effects become important, it also decreases with Reynolds number. A component of the thrust is the exit pressure, which is neglected since it is also only a function of the nozzle expansion ratio. The thrust, for an extruded nozzle, can then be approximated in terms of the Reynolds number as

$$F_t \propto \dot{m} v_{exit} \propto Re \; h_o \; \mu_t \; v_{exit} \quad . \tag{2.5}$$

Thus, for a constant thrust, the Reynolds number increases as scale decreases. This will result in lower frictional losses, and a higher delivered I_{sp}. Though the tank mass will not directly reduce with scale, the amount of propellant required is less due to the higher I_{sp}, and the engine mass is also less. Thus, the thrust to propulsion system mass can be increased significantly with a reduction in scale.

2.3 Performance Trade-offs with Design

The ability to fabricate small features leads to the enhancements described in the previous section. Just as there are direct benefits to pursuing a reduction in scale, there are direct drawbacks to small features. As the scale is reduced, the residence time of a particle in a flow channel is reduced. This leads to incomplete equilibration of the flow energy. If the residence time is less than the relaxation time, the thermal energy of the stored gas will not fully convert into kinetic energy, and performance will suffer. These are known as frozen losses. Frozen losses may also represent the energy that is used to break molecular bonds, which is not converted into fluid kinetic energy because the molecules do not have time to recombine while in the nozzle. Thus, the flow is a fixed (or frozen) composition at a lower energy state throughout the nozzle. These losses

are a function of Reynolds number since collision frequency is a function of gas density, and residence time or scale.

Another direct drawback of the microscale is the limitation of fabrication technology. This leads to surface roughness that is a larger fraction of the local nozzle width than for macrodevices. This trait must be examined to assess its impact on thruster performance.

Though significant performance advantages can be demonstrated, microscale components present interesting design challenges. In many areas the "cube-square" law can improve performance locally, and at the same time work against the overall design. As described previously, heat transfer to the fluid improves at small scales. However, the hot fluid will just as easily transfer the heat energy back to the environment further downstream if the wall is cold. Since it is difficult to insulate at these small scales, efficiency can suffer when trying to deliver heat energy.

As will be demonstrated throughout this work, many solutions can be arrived at by developing macroscale solutions, and integrating them with the microdevices. In this example, the entire heater chip can run at the heater temperature to prevent fluid energy losses, and then encased in ceramic to reduce the conduction and radiation losses. This way, the microscale geometry can deliver high heater effectiveness, but the larger conduction paths through low conductivity materials can reduce the losses. These issues are all addressed as device packaging, and are solved through system trade studies to optimize the designs.

2.4 Thruster Baseline

The previous analysis directs the low-thrust propulsion system designer to achieve the smallest scales possible, while maintaining high-Reynolds number flows. This section establishes the desired thruster scale, nozzle expansion ratio, and divergence angle, within the constraints of fabrication. Based on the expected geometry, a baseline performance is estimated to show that the requirements can be met.

2.4.1 Thruster Requirements

Using the JPL design reference missions as a requirements driver, the thruster will operate in the thrust range of 1-15 mN. It should be capable of providing at least 56 seconds of I_{sp}, because this has already been demonstrated by previous microsystems [14]. These requirements are to be achieved while reducing the mass and physical dimensions of the system.

2.4.2 Thruster Scale Limitations

The product will be driven by what can be reasonably achieved through microfabrication. The thrusters are limited to 2-D features due to the anisotropic etching techniques employed. 10 micron feature width appears to be the lower limit for high aspect ratio (local nozzle width to depth) etches, which is currently limited to 30:1. In addition, the surface roughness is at least 0.3 microns, and may become as large as 1 micron for deep etches. The minimum feature size should be chosen such that the roughness is no more than 1% of the nozzle width.

High aspect ratios are required so that endwall boundary layer effects are minimized, and the effective expansion ratio of the nozzle is maximized, which maximizes I_{sp}. The aspect ratio limitations of the etcher limits the nozzle expansion ratio. As the geometric expansion ratio grows, the exit plane aspect ratio reduces due to the limitations on nozzle etch depth. This, in turn, increases the endwall boundary layer effects and reduces the overall effective expansion ratio.

Boundary layer growth is a function of the nozzle length, or divergence angle for a given expansion. As divergence angle increases, boundary layer thicknesses decrease, which decreases blockage and increases performance. However, large divergence angles increase the momentum component perpendicular to the thrust axis, which is lost as useful thrust.

2.4.3 Baseline Geometry Specifications and Operating Conditions

The requirements and limitations drive the development of a nozzle that has a 20 micron throat width, 25:1 expansion ratio, 300 micron etch depth, and a nozzle divergence angle of 20°. If this thruster is operated at a chamber pressure of 100 psi, it will ideally produce 7.0 mN of thrust at 75 seconds of I_{sp}, based on a Quasi 1-D evaluation of the nozzle. The thrust reduces to 1.0 mN for a chamber pressure of 1 atmosphere. The expansion ratio is chosen to maintain a greater than 0.5:1 aspect ratio of the nozzle exit plane.

This ideal performance will suffer from significant viscous losses because the baseline operating point is at a Reynolds number of 2040 at 100 psi chamber pressure, and 300 at 14.7 psi chamber pressure. These operating points will serve as inputs to the numerical analysis undertaken in Chapter 3. These simulations are based on the continuum assumption. The continuum assumption is best characterized by the Knudsen number or the ratio of the particle mean free path to that of the scale of interest. Mean free path(λ) and Knudsen number are computed as

$$\lambda = \frac{1}{\sqrt{2}\pi D_c^2 N_d} \tag{2.6}$$

$$Kn = \frac{\lambda}{D_t} \ , \tag{2.7}$$

where D_c is the gas molecular diameter, N_d is the number density, and D_t is the throat width [18]. For the baseline nozzle geometry, the highest throat Knudsen number is 0.003 operating at 1 atm chamber pressure. However, the continuum assumption may break down as the pressure drops in the expansion. For the largest expansion of interest (25:1), the Knudsen number will only increase to 0.036. At this point the nozzle appears to be operating well within the limit of the continuum regime, though wall slip velocity may become significant.

Chapter 3

Numerical Analysis of Micronozzle Performance

3.1 Analysis Objective

Metrics are established for the comparison of the MEMS-based system to conventionally-machined propulsion systems of any architecture. These metrics will characterize the performance of the micronozzle as a propulsion device, and are indicative of the efficiency by which the stored energy of a pressurized gas is converted to kinetic energy exiting the nozzle. The metrics are quantified in terms of momentum performance parameters, such as I_{sp} and thrust, and efficiencies, which indicate the viscous losses. A numerical analysis is used to compute the ideal performance as well as establish performance benchmarks for real operation with viscous losses. The simulations are run for the fabricated nozzle geometries discussed in Chapter 4 at the operating conditions tested, which are discussed in Chapter 5.

This chapter answers fundamental questions concerning the flow behavior in the nozzle geometries at this scale. The computational analysis is used to determine if the flow is subsonic or supersonic, laminar or turbulent at the nozzle exit. A knowledge of the flow field can be used to infer the condition of the boundary layer, and the relative performance of these geometries to those that are of a larger scale and differing geometry. The set of solutions is used to create a performance map which indicates the optimum operating point of the nozzles.

3.2 Assumptions of the Numerical Analysis

The scaling analysis presented in Chapter 2, determined the flow was operating in the continuum regime. With this understanding, the nozzle flow can be simulated with the Navier-Stokes equations, which are based on the continuum assumption. As described in Chapter 1, low-Reynolds number nozzles have been analyzed by both the full viscous Navier-Stokes Equations, and the inviscid Euler equations with boundary layer corrections. The inviscid equations, which are less computationally intensive, could be used to simulate the fluid flow through the nozzle and the geometry could then be modified to account for the boundary layer influence, as long as an inviscid core exists. However, the full Navier-Stokes equations represent a much more complete picture of the flow, involve no iteration of the geometry, and are now easily solvable with today's computational power.

Since the nozzles developed here are of an extruded contoured geometry, the simulation geometry should reflect this. A three-dimensional numerical simulation would provide a complete picture of the fluid dynamics occurring with the nozzle, and an accurate performance prediction. However, this will come at a high computational price. A two-dimensional simulation will provide an accurate performance prediction for cases in which the endwall boundary layers are thin with respect to the nozzle aspect ratio. Due to the anticipated aspect-ratio of the fabricated nozzles, this condition will not be satisfied at the exit for most of the range of Reynolds numbers tested, and a correction for endwall boundary layers will be incorporated.

This correction is presented in section 5.5.1.

This simulation does not account for nonequilibrium effects present in the nozzle. It assumes that all relaxation times are greater than the residence in the nozzle.

3.3 Finite Volume Formulation

The numerical analysis is a two-dimensional Navier-Stokes simulation through a converging-diverging (de Laval) nozzle at various Reynolds numbers. This analysis is based on a computer program, which implements the finite volume formulation of the Navier-Stokes equations with Van Leer's flux-splitting [19]. This computer code was originally developed for flow in closed geometries (microbearings) at the microscale [20], but has been adapted for the current application by the author. The governing equations for a two-dimensional flow can be written in conservative form [21] as

$$\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0 \quad , \tag{3.1}$$

where U represents the state variables for mass, momentum in the x and y directions, and energy. F and G are the fluxes of these variables across a control surface. The state vector is defined as:

$$U = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix} \quad . \tag{3.2}$$

All of the state variables have been non-dimensionalized by the appropriate quantities defined at the throat condition. This non-dimensionalization is carried throughout the remainder of the chapter and in the definition of the fluxes and their component quantities. The non-dimensionalization is detailed in Appendix A. The fluxes in equation (3.1) can be separated as $F = F_i - F_v$ and $G = G_i - G_v$, in the x and y directions respectively, where subscripts i and v represent inviscid and viscous fluxes. These fluxes are defined as:

$$F_i = \begin{bmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ (\rho E + P)u \end{bmatrix} ; \; F_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\tau_{xx} + v\tau_{xy} - q_x \end{bmatrix} \tag{3.3}$$

$$G_i = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + P \\ (\rho E + P)v \end{bmatrix} ; \; G_v = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ u\tau_{xy} + v\tau_{yy} - q_y \end{bmatrix} \quad , \tag{3.4}$$

where the fluid shear stresses and heat conduction are mathematically defined as

$$\tau_{xx} = \frac{2\mu}{3Re}(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}) \tag{3.5}$$

$$\tau_{yy} = \frac{2\mu}{3Re}(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x}) \tag{3.6}$$

$$\tau_{xy} = \frac{\mu}{Re}(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}) \tag{3.7}$$

$$q_x = -\frac{\mu}{(\gamma - 1)Re\ Pr}\frac{\partial T}{\partial x} \tag{3.8}$$

$$q_y = -\frac{\mu}{(\gamma - 1)Re\ Pr}\frac{\partial T}{\partial y} \quad , \tag{3.9}$$

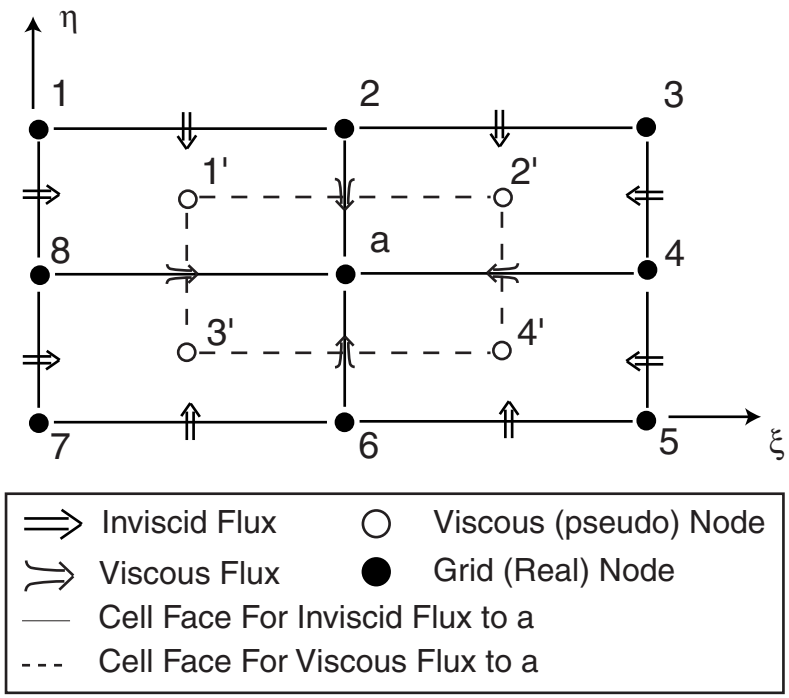


Figure 3.1: Flux model in computational space

where Re and Pr are defined in Appendix A.

3.3.1 Spatial Discretization

The domain is discretized in a structured grid arrangement. The variables x and y represent physical displacements in the nozzle coordinate system, while ξ and η represent cell displacement in an orthogonal grid-centered coordinate system. Figure 3.1 is a sample of the $\xi - \eta$ computational space. There exists a transformation matrix that relates the grid geometry to computational space. All state variable derivatives can be transformed from physical space to grid coordinates by applying the chain rule. An example of this transformation is presented in equation (3.12).

Inviscid fluxes for each node are calculated using the eight nodes which surround it. State information for each of the cells is stored at the vertex locations. Figure 3.1 depicts the contribution of the flux from each cell face to the vertex representing the volume of interest. For example, the contribution to point a from face $1-2$ is

$$F_{ia} = -[\frac{1}{2}(F_{i1} + F_{i2})(y_2 - y_1) - \frac{1}{2}(G_{i1} + G_{i2})(x_2 - x_1)] \quad . \tag{3.10}$$

The convention is that outgoing fluxes are positive, thus the equation is negative for those fluxes shown in the figure. The fluxes are then summed for each of the faces contributing to the cell. This is accomplished in a computationally efficient manner by computing it once for each node, and then distributing the flux to the surrounding nodes, so that the flux from a specific point is not recomputed for each of its adjacent faces.

The viscous fluxes for a given node uses the same surrounding nodes as for the inviscid fluxes. Because the viscous fluxes contain derivative terms, the node itself is also involved and the cell over which these fluxes are calculated is also shown in Figure 3.1. By displacing the pseudo-nodes, the derivative terms can be computed from the four surrounding real nodes. These values are then used to calculate the viscous fluxes into the cell 1'-2'-3'-4'. An example of a partial computed at a pseudo-node such as 1' is:

$$
\begin{aligned}
\frac{\partial u}{\partial x} &= \frac{1}{J}\left(\frac{\partial y}{\partial \eta}\frac{\partial u}{\partial \xi} - \frac{\partial y}{\partial \xi}\frac{\partial u}{\partial \eta}\right) \qquad (3.11)\\
\text{where}& \\
J &= \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}\\
\frac{\partial y}{\partial \eta} &= \frac{1}{2}((y_1 + y_2) - (y_8 + y_a))\\
\frac{\partial u}{\partial \xi} &= \frac{1}{2}((u_2 + y_a) - (u_8 + u_8))\\
\frac{\partial y}{\partial \xi} &= \frac{1}{2}((y_2 + y_a) - (y_1 + y_8))\\
\frac{\partial u}{\partial \eta} &= \frac{1}{2}((u_1 + u_2) - (u_8 + u_a)) \quad .
\end{aligned}
$$

The non-derivative terms for a pseudo-node as well as its x and y location are computed as an average of the values at the four surrounding nodes.

3.3.2 Temporal Discretization

The code implements the four-stage Jameson Runge-Kutta Scheme [22]. Each of the state variables is updated based on the maximum time-step derived from the CFL condition which is modified from the inviscid value due to the presence of viscosity. The fluxes are weighted based on the ratio of the cell area and the maximum time step. The state variable is computed as

$$
\begin{aligned}
u^{(1)} &= u^n - \frac{1}{4}\frac{\Delta t}{A}\sum \text{Fluxes}^{(n)}\\
u^{(2)} &= u^n - \frac{1}{3}\frac{\Delta t}{A}\sum \text{Fluxes}^{(1)} \qquad (3.12)\\
u^{(3)} &= u^n - \frac{1}{2}\frac{\Delta t}{A}\sum \text{Fluxes}^{(2)}\\
u^{(n+1)} &= u^n - \frac{\Delta t}{A}\sum \text{Fluxes}^{(3)}
\end{aligned}
$$

where stage n is the current time step, each of the superscripts 1-3 are the stages of the Runge-Kutta scheme, and $n+1$ is the state vector at the next time step. The viscous fluxes (real and artificial) are computed only at stage 1 to improve computational efficiency.

The timestep is computed as

$$
\Delta t = \frac{S\Delta t_{CFL}}{1 + \frac{2}{Re_\Delta}} \quad , \qquad (3.13)
$$

where S is a safety factor (in this case 1.2) and Re_Δ and Δt_{CFL} are computed as

$$
\begin{aligned}
\Delta t_{CFL} &= \left(\frac{u}{\Delta x} + \frac{v}{\Delta y} + a\sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}}\right)^{-1} \qquad (3.14)\\
Re_\Delta &= min\left(\frac{\rho(u+a)\Delta x}{\mu}, \frac{\rho(v+a)\Delta y}{\mu}\right) \quad . \qquad (3.15)
\end{aligned}
$$

where u and v are the velocity components from before, and a is the speed of sound. The timestep is computed in this manner in order to satisfy the CFL condition for each cell. The minimum timestep for the entire grid is used to permit stability in the solution.

The area enclosed by each group of four contiguous nodes is calculated as the magnitude of one-half the cross product of the diagonal vectors for the quadrilateral they define. For the cell in figure 3.1, this would be

$$A_{12a8} = \frac{1}{2}[(x_2 - x_8)(y_1 - y_a) - (x_1 - x_a)(y_2 - y_8)] \quad . \tag{3.16}$$

The area associated with the inviscid fluxes to a node is the sum of the four surrounding quadrilaterals.

3.3.3 Artificial Viscosity

Since the numerical calculations are accomplished through an explicit scheme, the odd-even mode is allowed to grow. In order to obtain a converged solution, this must be damped out through the use of fourth-order artificial viscosity. It is formulated as an additional term in the viscous flux contribution as

$$\begin{aligned} F_{i,j} &= F_{viscous} + \frac{AV_4 \frac{1}{4} A_{i,j}}{\Delta t}[S_\eta F^4_{\eta_{i,j}} + S_\xi F^4_{\xi_{i,j}}] \qquad (3.17) \\ \text{where} & \\ F^4_{\eta_{i,j}} &= u_{i+2,j} - 4u_{i+1,j} + 6u_{i,j} - 4u_{i-1,j} + u_{i+2,j} \\ F^4_{\xi_{i,j}} &= u_{i,j+2} - 4u_{i,j+1} + 6u_{i,j} - 4u_{i,j-1} + u_{i,j+2} \quad , \end{aligned}$$

where AV_4 is a user defined constant, $A_{i,j}$ is the area of the cell, and Δt is the timestep. It is found that AV_4 can be reduced from 0.02 for the small expansion ratio cases to 0.0005 for the largest expansion ratios and maintain stability.

3.3.4 Boundary Conditions

Since the geometry has been restricted to two dimensions, the domain is bounded by four distinct regions. These are defined as the inlet, outlet, and the contoured nozzle surface (sidewalls). Figure 3.2 depicts the domain of a sample nozzle. The flow performance is based on the pressure ratio across the nozzle. This drives the fluid into the contoured channel where it is accelerated due to pressure forces from the converging-diverging surface. The flow becomes choked when the pressure ratio is sufficient to force sonic flow at the throat. At this point, the flow is hyperbolic in nature, and the downstream region does not influence the upstream state.

Inlet Boundary Condition

The inlet boundary is followed by a section with parallel walls. This allows the inlet velocity to be specified as purely axial. For the operation of the real device, only the inlet pressure will be known from a measurement very near the chamber. Due to the large volume of the chamber, the total pressure is assumed constant across the chamber with very low residual velocity. Since the flow is subsonic, of the three Riemann invariants present in a one dimensional flow, two of them are set by the farfield conditions (∞), and one is defined by the states in the domain (which are unsubscripted). The flow variables in the domain are taken from the grid line at $\xi = 1$. The boundary $\xi = 0$ can then be solved from equations 3.23- 3.26. These are, as found in Hirsch [23]

$$w_1 = J_+ = u_\infty + \frac{2a_\infty}{\gamma - 1} \tag{3.18}$$

$$w_2 = J_- = u - \frac{2a}{\gamma - 1} \tag{3.19}$$

$$w_3 = \frac{p_\infty}{\rho_\infty^\gamma} \tag{3.20}$$

Once the Riemann invariants are computed they can be used to determine the primitive variables at the boundary, as:

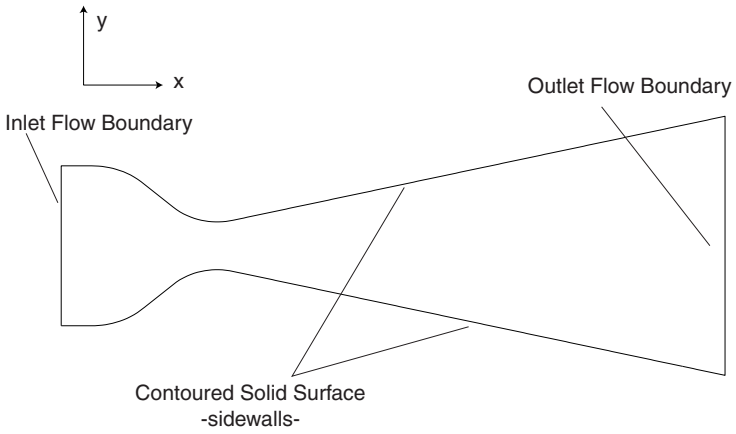


Figure 3.2: Nozzle Domain

$$a = (J_+ - J_-)\frac{\gamma - 1}{4} \tag{3.21}$$

$$u = \frac{1}{2}(J_+ + J_-) \tag{3.22}$$

$$v = 0 \tag{3.23}$$

$$\rho = \rho_\infty (\frac{a}{a_\infty})^{\frac{2}{\gamma - 1}} \tag{3.24}$$

$$p = \rho \left. \frac{p}{\rho^\gamma} \right|_\infty \tag{3.25}$$

$$\rho E = \frac{p}{\gamma - 1} + \frac{1}{2}(\rho u^2 + \rho v^2) \quad . \tag{3.26}$$

Outlet Boundary Condition

As with the inlet condition, the treatment of the outlet boundary depends upon whether the flow is supersonic or subsonic. The subsonic regions should be influenced by the downstream conditions. Under this assumption, the initial computations set the pressure in the subsonic region with an assumed plume pressure. This was 25% of the core flow pressure. However, Kim [11] and Peraire [24] suggested the parabolic nature of the boundary layer allows all state variables to be extrapolated from the interior regardless of Mach Number, as long as the inviscid core flow in the nozzle is supersonic. The equations are parabolic in the thin layer limit, but truly elliptic in the subsonic region. The existence of an inviscid supersonic core ensures a favorable pressure gradient in a nozzle expanding to vacuum and allows the thin layer limit to be applied. Thus, in subsequent versions of the code, the boundary conditions were implemented so that outflow boundary cells only receive fluxes from upstream and streamwise cells. Thus, the flow exiting the domain is not influenced by downstream conditions, in the simulation.

Wall Boundary Condition

For the initial numerical simulations, it is assumed that no-slip will occur at the wall; therefore, the u and v components of velocity are set to zero. This is traditionally accepted as a valid assumption for Knudsen numbers less than 0.03 [18]. Until the Knudsen number is evaluated for test conditions, the no-slip

implementation will provide reasonable results. The numerical results will be used to evaluate the Knudsen number throughout the flow field, and determine if slip flow is required.

With the velocities assumed zero, density and specific energy will complete the state vector. The specific energy is a function of the wall temperature. It is assumed that the fluid particles come to thermal equilibrium with the wall, based on the low Knudsen number assumption. However, at this point the wall equilibration temperature is unknown, but could be determined from an assumption of either an adiabatic or isothermal wall. Once again, the assumption is based upon the nature of the device to be tested. Since the device is going to be made of a large volume of silicon (relative to the volume of the nozzle etched in the die) and the support structure has large mass, it is reasonable to assume that the temperature of the nozzle wall will reduce slowly. The time to equilibrium temperature will be much less than the thruster operation. For these reasons, it is assumed the wall will maintain the supply temperature. The isothermal wall assumption is experimentally verified in section 6.5.3.

Finally, the state vector can be completed with either an assumption about the pressure or density. This is easiest to complete with a centripetal force balance at the wall. The normal gradient of pressure at the wall is going to be equal to the head pressure arising from the centripetal force on the fluid. This is formulated as

$$\left.\frac{\partial P}{\partial n}\right|_w = \frac{\rho V^2}{R} \quad , \tag{3.27}$$

where R is the local wall radius of curvature. Since the velocity at the wall is zero, the right hand side is zero. In regions in which the slip velocity might be significant (near the exit), the wall is flat ($R = \infty$), and the right hand side would still be zero. Thus, equation (3.27) can be used to solve for the density at the wall and complete the state vector. In order to evaluate P_n (which is shorthand for $\frac{dP}{dn}$) in an orthogonal grid, a transformation matrix is established as

$$\begin{bmatrix} P_\xi \\ P_\eta \end{bmatrix} = \begin{bmatrix} N_\xi & T_\xi \\ N_\eta & T_\eta \end{bmatrix} \begin{bmatrix} P_n \\ P_t \end{bmatrix} \quad , \tag{3.28}$$

where T and N are the vectors tangential and normal to the wall. Equation (3.28) can be inverted and solved for P_n as

$$P_n = \frac{1}{N_\xi T_\eta - N_\eta T_\xi}[P_\xi T_\eta - P_\eta T_\xi] \quad . \tag{3.29}$$

Since the wall normal pressure gradient is zero from Equation 3.27, the left hand side of 3.29 is also zero. A forward difference can be used to determine P_η and a centered difference to compute P_ξ. The state equation in the form of $p = \rho e(\gamma - 1)$ is substituted for the local pressures and the wall density is computed as

$$\rho_{j,i} = \frac{\frac{T_\eta}{T_\xi(\gamma-1)}P_\xi + 4\rho e_{j+1,i} - \rho e_{j+2,i}}{3e_{j,i}} \quad , \tag{3.30}$$

where i is the column index, and j is the row index. $e_{j,i}$ is known from the assumed wall temperature.

Wall Slip Velocity

As mentioned previously, in the instances in which the Knudsen number increases above 0.03 [18], a wall slip velocity may be imposed in the solution of a flow field. As explained in Rosenhow [25], Maxwell derived a relationship between the tangential velocity of a gas relative to the wall from kinetic theory. Thus, the wall velocity which was previously assumed to be zero in the state vector can be replaced by

$$u = \frac{2-F}{F} Kn \left. \frac{du}{dy} \right|_w \quad , \tag{3.31}$$

where F is the fraction of molecules diffusely reflected from the surface, and $\frac{du}{dy}|_w$ is the wall normal velocity gradient. In this case, F is assumed to be 1; indicating full accommodation; however, this is a function of the surface roughness and gas-surface composition. Arkilic [26] has shown that for an atomically-smooth silicon-air interaction at low Mach numbers, this value is closer to 0.8. The wall roughness in the cases examined here is significantly higher, and would result in a more diffuse reflection and higher accommodation, thus F was assumed to be 1.

Due to the additional computations required to impose this boundary condition, this version of the code runs much slower than the no-slip version. Thus, if the performance results of these two calculations are similar, it is prudent to proceed with the no-slip version. A side-by-side comparison of these computations, presented in the section 3.6.1, demonstrates the no-slip condition is sufficient for performance predictions.

3.4 Running the Code

In order for a solution to be achieved, several steps must be undertaken. 1) A domain must be discretized and arranged in the form of a grid. 2) A solution is converged such that the state variables do not change with each step. 3) The solution must be independent of the grid resolution. Once each of these are accomplished, the converged solution will represent the flow field for the specified Reynolds and Prandtl numbers.

3.4.1 Nozzle Terminology

Figure 3.2 defines the basic nozzle components. The dimensions referred to throughout the text are length, width and height. Length is the x component which defines the primary flow direction of the nozzle. The width is the y direction associated with the streamwise component of the flow. The height is the z direction or the dimension normal to the contoured nozzle geometry. Since the nozzle is created by etching the silicon in the z direction, this is also referred to as the depth. The contoured geometry is the wall feature which makes up the converging-diverging nozzle used to accelerate the flow to supersonic velocities, also referred to as the sidewalls. The endwalls are the flat walls that cap the contoured channel and contain the flow.

The nozzle geometries analyzed in this chapter are based on those fabricated for testing. Each of these nozzles are referred to by their area ratio and operating condition. Table 3.1 lists all area ratios, operating conditions and grid sizes. The operating conditions are set solely by the throat Reynolds numbers, as defined in Appendix A, since all conditions are run for Nitrogen (or a gas of similar Prandtl number).

3.4.2 Grid Development

The choice of grid point placement within the domain will facilitate reaching a grid resolved solution. If the points are well placed, the computed gradients and hence fluxes lead to a faster more efficient solution. Since the code is run on a structured grid, one in which the points are arranged in a logically connected row and column format, the point locations should be smoothly varying with no discontinuities along a row or column. The best solutions will arise from rows which trace out streamlines, and columns which run orthogonal to the rows, and ultimately the grid boundaries. Such a grid is created by an elliptic grid generator, which is based upon solving Laplace's equation for x and y with the domain boundary point locations serving as the boundary conditions. These equations are found in [27] and are repeated here as:

$$\nabla^2 \xi = \xi_{xx} + \xi_{yy} = P(\xi, \eta) \tag{3.32}$$

$$\nabla^2 \eta = \eta_{xx} + \eta_{yy} = Q(\xi, \eta) \tag{3.33}$$

where ξ is the x-component of grid-centered coordinate system and η is the y-component of the same. P and Q represent forcing functions that enforce the interior point orthogonality. Steger and Sorenson [28]

Table 3.1: Nozzle geometry test cases and operating conditions used for analysis

Area Ratio	Throat Size (μm)	Grid Size	Chamber Pressure (psi)	Re_{throat}
5.43:1	19	125x165	96.2	1898
			80	1578
			65	1282
			60	1138
			45	887
			25.2	497
			9	177
7.14:1	34	155x160	96.4	3381
			80	2805
			65	2279
			45	1578
			25	876
			20	350
15.34:1	18	210x220	96.9	1799
			80	1485
			60	1114
			40	742
			20	371
			10	185
16.9:1	37.5	255x195	97.2	3721
			80	3063
			65	2488
			50	1914
			35	1340
			20	765
			10	382

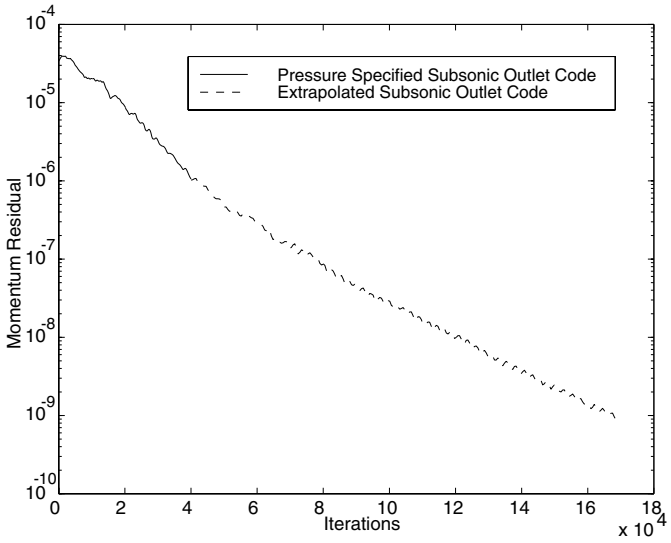


Figure 3.3: Convergence history for a 5.4:1 nozzle with 125x160 points at a Reynolds number of 1284

formulated functions for P and Q which were used in this grid development for clustering points along a boundary, leading to rows very near the streamlines. The number of points can be estimated visually for the early runs, but ultimately is driven by what leads to a grid-resolved solution.

3.4.3 Solution Convergence

The solution varies from one time-step to the next due the flux imbalance on each of the state variables. When the sum of the fluxes into a given cell goes to zero the steady state is reached. Solution convergence is gauged by the reduction in the momentum residual. The local residual is the change in a state variable between timesteps. The average residual is the average of this quantity over all of the grid points. Ultimately, the residual will reach a steady value in which the round-off error continues to vary the solution by a miniscule amount and prevents further convergence. As mentioned previously, there are two modes of running the code. The first converges the solution with a specified outlet pressure in the subsonic region. Though this is not an accurate picture of the flow-field, it is a robust method for achieving a good first approximation of the solution. Once the residual is below 1×10^{-6}, the second mode, which uses extrapolation for the entire exit plane, is run until the residual reaches a small enough value that the solution performance is unaffected by subsequent iterations. There is no change in the first 6 decimal places of the performance for a residual of 1×10^{-10}. Figure 3.3 depicts the convergence history for the 125x160 grid at a Reynolds number of 1284.

3.4.4 Grid Resolution

A converged solution indicates that the solution is not changing significantly with subsequent iterations; however, it may still be dependent upon the number of points specifying the domain. A solution is grid-independent if the flow field does not change for a grid that has twice the number of points as the grid of interest. There is no visible distinction between the test cases run for the 5.4:1 grid, and the performance parameters vary by less than 0.2% indicating the solutions for this mesh are grid resolved.

3.5 Flow Field Analysis

This section examines the steady-state flow fields evaluated for various Reynolds numbers. The primary effect will be a reduction in the effective area ratio due to blockage from the boundary layers. The boundary

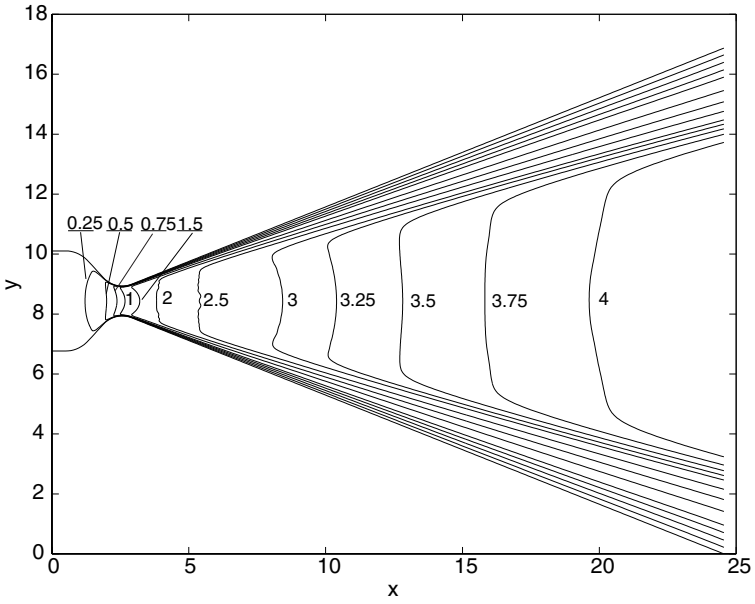


Figure 3.4: Mach Number contours for the 16.9:1 nozzle at a Reynolds number of 1914

layers can be seen in the contour plot depicting the Mach number throughout the nozzle as in Figure 3.4. The rapid decrease in Mach number near the walls is an indication of the viscous interaction occurring there. By inspection, the boundary layer appears to fill half of the exit plane in the 16.9:1 nozzle. The thrust of the nozzle has both a momentum and pressure component. The Mach number is an indication of the momentum component, and the contours can be used to gauge the loss in momentum due to the viscous interaction with the walls.

The pressure contours are shown in Figure 3.5. While the wall-normal pressure gradient is forced to be zero, the contours remain relatively constant at each axial location for the core flow. This graph can also be used to estimate exit pressure to determine if the nozzle is overexpanded, or if a shock should be expected. The simulation predicts the exit pressure will be 300 mTorr for the lowest Reynolds number cases in the largest expansion ratios. As will be discussed in Chapter 5, the nozzles exhausted to 100 mTorr in the tests.

Since thrust is directly related to exit momentum, the fluid density is also necessary for computing this quantity. Figure 3.6 is a contour plot of density, which is nearly orthogonal to pressure contours inside the boundary layer. This is due to the temperature boundary layer which develops due to the wall being at a fixed temperature, and the fluid temperature dropping due to the expansion. Through the equation of state, the density also drops as the temperature drops normal to the wall. Figure 3.7 depicts the temperature contours in the nozzle.

These plots merely show the character of the flow field for one particular operating condition. However, similar results can be found in all of the 27 cases run for the 4 different nozzle geometries. The only variation is the extent to which the boundary layers affect the nozzle performance. To illustrate this, the exit plane Mach number profile is plotted for four Reynolds numbers in the 5.4:1 nozzle in figure 3.8. The viscous effects can be easily discerned as a reduction in the maximum Mach number. In addition, the core flow is reduced to a smaller area due to the momentum being dissipated in the now thicker boundary layers. Figure 3.9 illustrates the pressure profiles at the same Reynolds numbers as in figure 3.8. The reduction in the flow velocity causes the static pressure to rise at the exit.

Figure 3.10 shows the axial variation of Mach number as a function of Reynolds number. The most striking distinction is the slope of these curves near the throat. They are similar for each of the CFD cases, but the slope is considerably less than that of the Quasi 1-D estimate. Thus, the throat contour has greatly reduced the fluid acceleration in the throat region. This does not affect overall performance because the simulations maintain this slope longer than that of the Quasi 1-D solution. Ultimately, the exit Mach number is influenced by the length of the nozzle wall from the throat to the exit, and the boundary layer

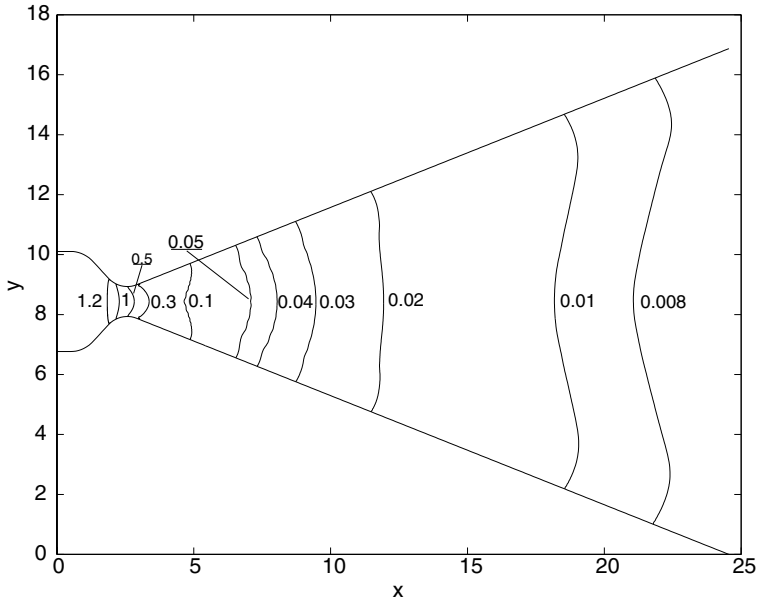


Figure 3.5: Non-Dimensionalized Pressure (P) contours for the 16.9 nozzle at a Reynolds number of 1914

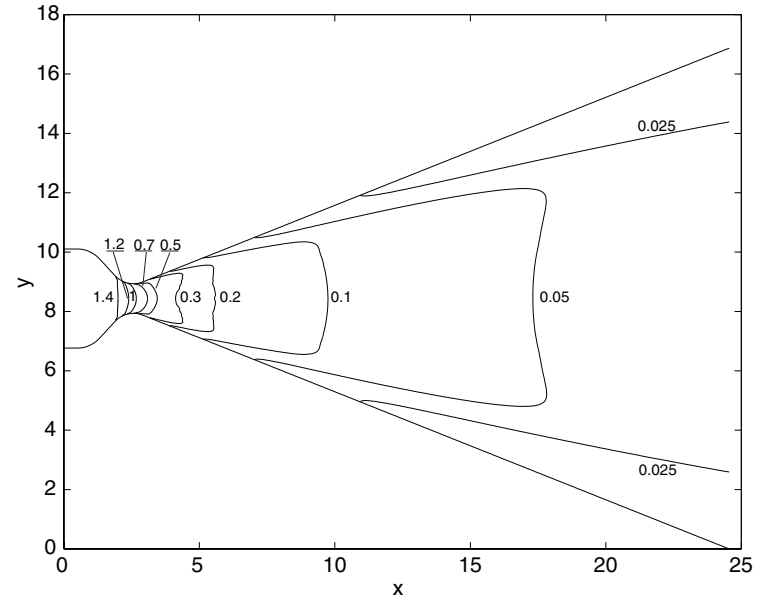


Figure 3.6: Non-Dimensionalized Density (ρ) contours for the 16.9:1 nozzle at a Reynolds number of 1914

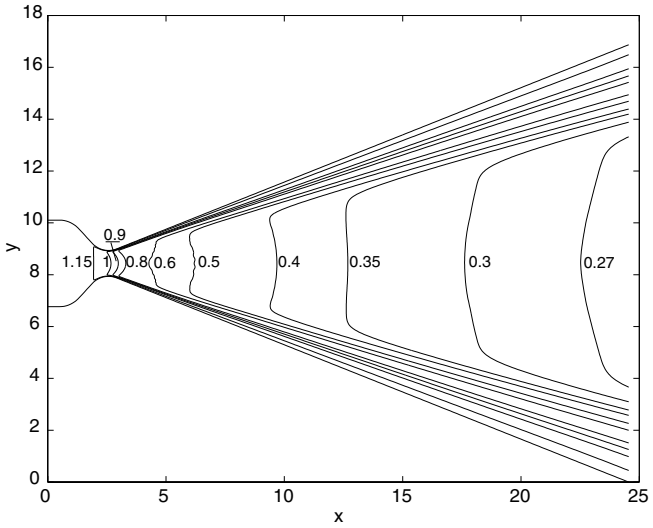


Figure 3.7: Non-Dimensionalized Temperature (T) contours for the 16.9:1 nozzle at a Reynolds number of 1914

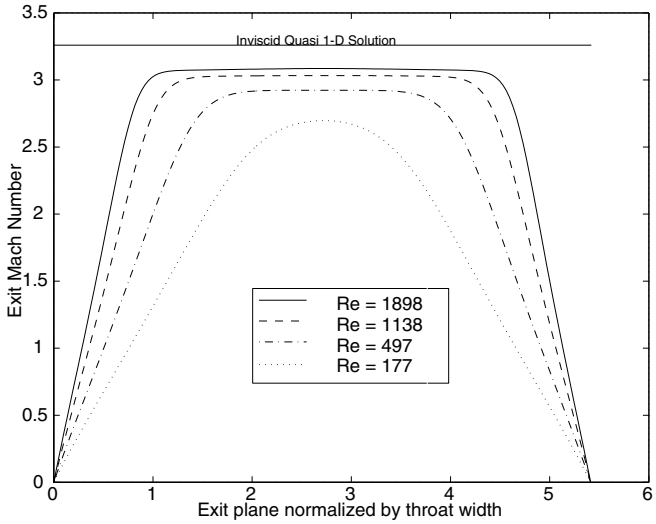


Figure 3.8: Exit Plane Mach Profiles for the 5.4:1 Nozzle operating at various Reynolds numbers.

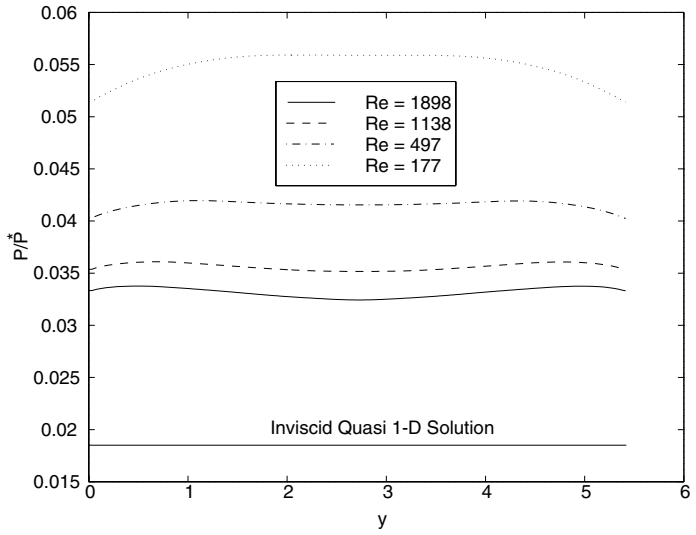


Figure 3.9: Exit Plane Pressure Profiles for the 5.4:1 Nozzle operating at various Reynolds numbers

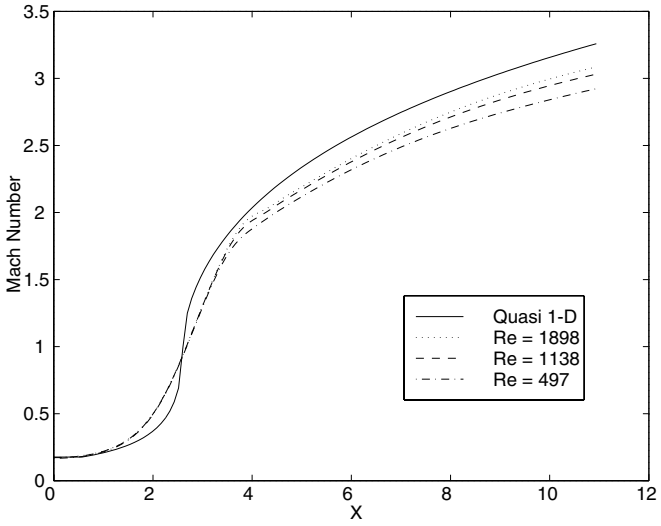


Figure 3.10: Centerline Mach Profiles for the 5.4:1 Nozzle operating at various Reynolds numbers

which develops in this region.

3.6 Performance Parameters

Ultimately, the micromachined nozzles are judged by their effectiveness as a propulsion device. Therefore, of interest, is how their thrust level and I_{sp} varies with chamber pressure. These values determine whether or not a propulsion system is suitable for a particular space mission.

At this stage, the physics of the nozzle flow is of interest, and how well the nozzle performs relative to the ideal. In this case, the ideal is the Quasi 1-D prediction of inviscid nozzle performance. This method assumes that all of the flow is axial; therefore, in order to make a better comparison, the ideal exit momentum is distributed over the nozzle divergence angle, and only the axial component is used to compute thrust. With this value, an efficiency can be computed as the ratio of the computed performance to the ideal for this angle. The viscous effects will cause this value to drop below unity.

The ideal mass flow is computed by applying the continuity equation to the choked conditions at the nozzle throat. The state variables of density and temperature (for computing the speed of sound) are computed from isentropic relations relative to the chamber conditions. The ideal thrust is computed from Quasi 1-D relations to the exit area ratio for both momentum and pressure components corrected for divergence. This is the inviscid or ideal for this operating condition. Ideal I_{sp} is defined as thrust per unit weight flow rate, or mathematically as:

$$I_{sp} = \frac{F_t}{\dot{m} g_o} = \frac{v_{exit}}{g_o} \quad , \tag{3.34}$$

which can be reduced to the effective exit velocity divided by gravity (g_o). I_{sp} is the primary performance indicator of a chemical propulsion system. As I_{sp} increases less propellant is required to achieve a given thrust. The efficiencies are ratios of the numerical result to the ideal. Thus,

$$c_d = \frac{\dot{m}_{\text{numerical}}}{\dot{m}_{\text{Ideal}}} \tag{3.35}$$

$$\eta_{thrust} = \frac{Thrust_{\text{numerical}}}{Thrust_{\text{Ideal}}} \tag{3.36}$$

$$\eta_{I_{sp}} = \frac{I_{sp_{\text{numerical}}}}{I_{sp_{\text{Ideal}}}} \quad . \tag{3.37}$$

The thrust is computed from the CFD result by numerically integrating the following equation over the exit plane as

$$F_t = \int_0^{D_e} (\rho_e u_e^2 + P_e) h_o dy \quad , \tag{3.38}$$

where D_e is the exit width, h_o is the nozzle height, and F_t is the thrust force. The state variables are taken from the CFD solution for a particular operating condition. The continuity equation is integrated at the throat to compute mass flow.

Figure 3.11 depicts the variation of the mass flow efficiency, also known as the coefficient of discharge, with Reynolds number. In general, all of the nozzles behave in roughly the same manner. This is because the mass flow is set at the throat, and all of the nozzles have the same entrance lengths and conditions.

Figure 3.12 shows the Reynolds number variation of I_{sp} efficiency. The I_{sp} is effectively the same as average exit velocity, at high expansion ratios. Since the boundary layer growth governs the effective area ratio, which determines exit velocity, I_{sp} will be a function of nozzle length. Even though the nozzle is longer, the I_{sp} has grown faster than the boundary layer at high Reynolds numbers, resulting in higher efficiencies. However, at low Reynolds numbers, the boundary layer thickness has a greater impact, and the

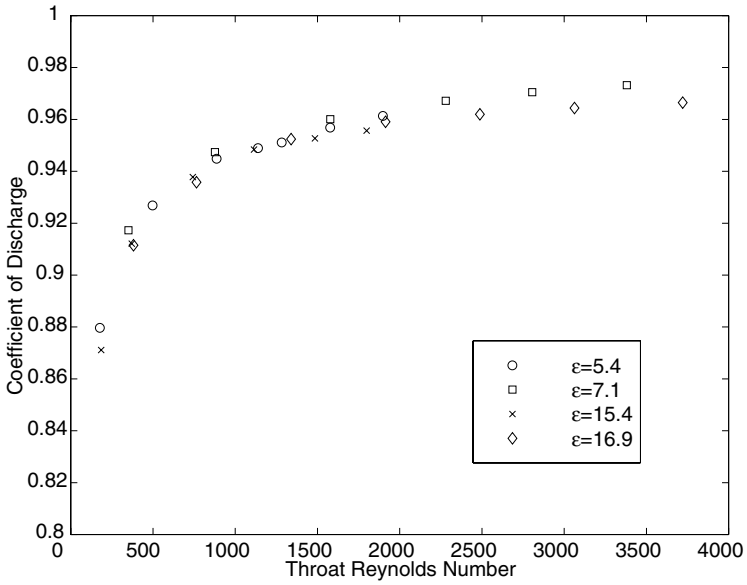


Figure 3.11: Numerical predictions of the variation in the coefficient of discharge with Reynolds number for all geometries tested. Inlet section and throat radius of curvature is identical for cases. Therefor all values are nearly identical.

lower area ratio nozzles have higher efficiencies. These effects are slight, and account for only a 1% variation in performance.

Figure 3.13 depicts the variation of thrust efficiency with Reynolds number. At high Reynolds numbers, the thrust efficiency is highest for high expansion ratio geometries. Thrust efficiency is the product of the mass flow efficiency and the I_{sp} efficiency. (Since thrust is the product of the mass flow and exit velocity, neglecting the pressure term.) Since the C_d is constant between the different cases for a given Reynolds number, the thrust efficiency will only vary as the I_{sp} efficiency. But as is shown in figure 3.12, this variation is slight among the different area ratios.

An additional parameter, which allows the effectiveness of the nozzle to be assessed, is the thrust coefficient (C_f). It is computed as

$$C_f = \frac{F_t}{P_o \; A_t} \quad , \tag{3.39}$$

where A_t is the throat area. This demonstrates the performance added by the nozzle above the chamber pressure acting over an orifice the size of the throat. The C_f variation with Reynolds number is depicted in figure 3.14. The ideal C_f is a function of area ratio and back pressure. However, since these simulations are exhausting to vacuum, they are purely a function of area ratio at high Reynolds numbers. At the lowest Reynolds numbers there is a cross-over between the 15.3:1 and the 16.9:1. This is due to the boundary layer in the longer 16.9:1 nozzle, being thicker, and it reduces the effective area ratio of the nozzle. This lowers the C_f below that of the shorter 15.3:1 nozzle. The concept of effective area ratio is explored further in section 3.7.1

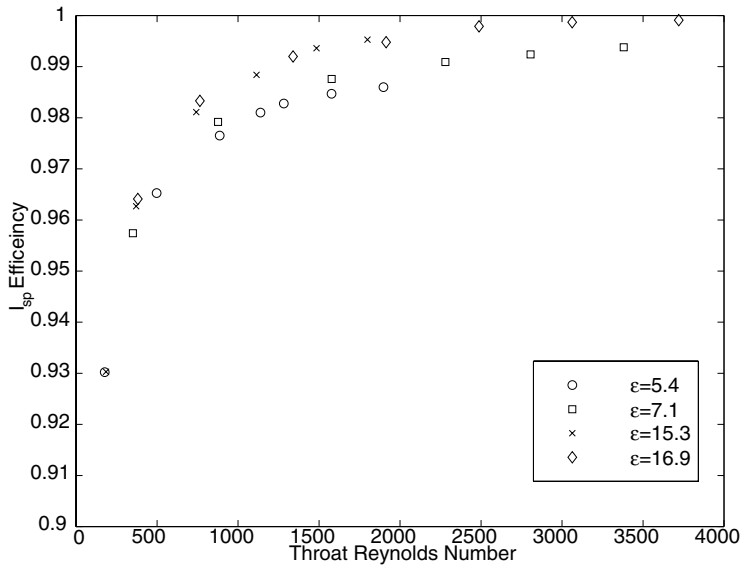


Figure 3.12: Numerical predictions of the variation in the I_{sp} efficiency with Reynolds number for all geometries tested

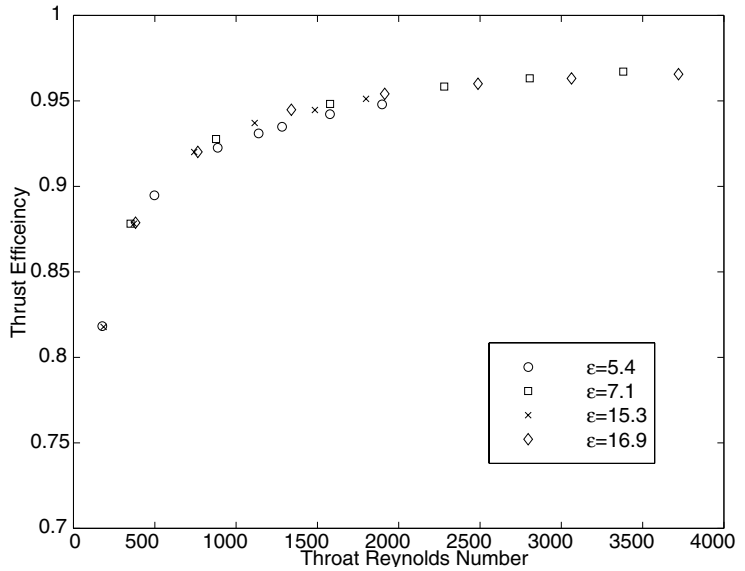


Figure 3.13: Numerical predictions of the variation in the thrust efficiency with Reynolds number for all geometries tested

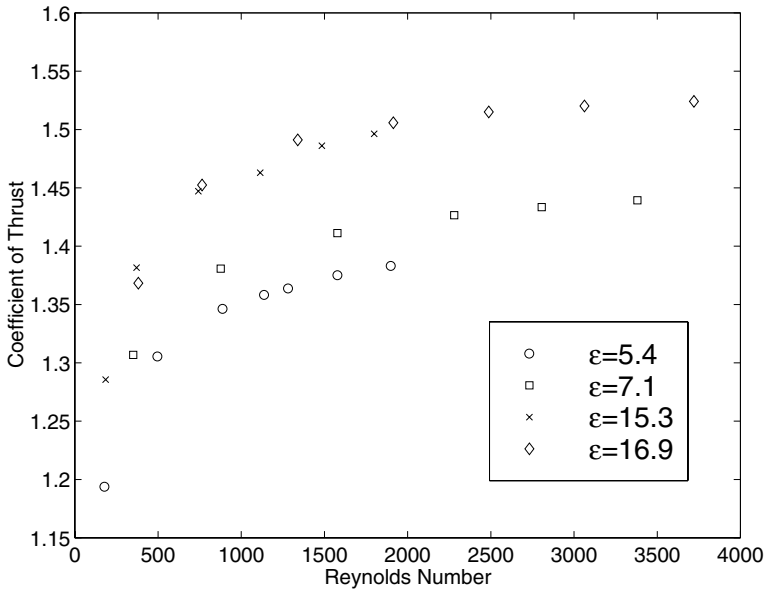


Figure 3.14: Numerical predictions of the variation in the thrust coefficient with Reynolds number for all geometries tested. The highest expansion ratios have the highest C_f, as expected.

3.6.1 Effects of Wall Boundary Conditions on Performance

In order to verify the proper formulation for the wall velocity boundary conditions was chosen, the effect of the wall-slip on performance is assessed. As mentioned previously, this effect becomes important above Knudsen numbers of 0.03. However, its effect on performance has yet to be determined. Figures 3.15 and 3.16 depict the exit plane profile for both the slip and no-slip cases for high and low Reynolds numbers. There is a perceptible difference in the Mach number profiles for the two cases, particularly at the walls, as one would expect. However, its impact on the performance is very slight. Table 3.6.1 records the performance variation for each type of boundary condition.

Table 3.2: Performance property variation with slip velocity incorporated for 5:41 nozzle at $Re = 1578$

Model	C_d	$\eta_{I_{sp}}$	η_{thrust}
No-Slip	0.9607	0.9884	0.9496
Slip	0.9627	0.9857	0.9510
Difference (%)	0.21	0.27	0.14

The influence of the low-density effects will be a function of Knudsen number. Figures 3.17 and 3.18 show the maximum and minimum values of the Knudsen numbers at different operating conditions. The variation over the range of operation is from 0.004 to 0.02 for all nozzle test cases. This results in little variation in slip velocity in the cases of interest. Figure 3.16 shows the effect of slip velocity on a Mach profile for a higher Knudsen number case, which varies little from the no-slip condition. Thus, the overall effect on performance is still negligible, and the use of the no-slip condition is justified.

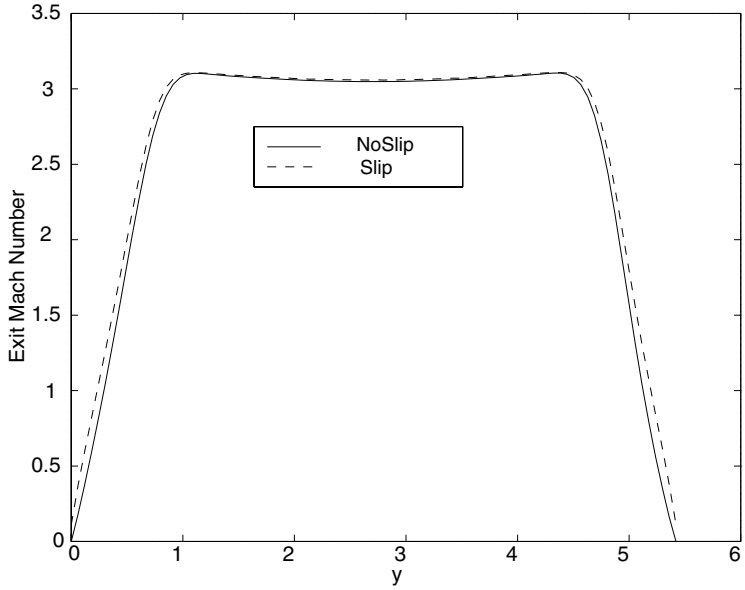


Figure 3.15: Exit plane Mach profiles for the 5.4:1 nozzle at a Reynolds number of 1578. The variation between the solutions is minimal and accounts for less than 0.5% of the thrust.

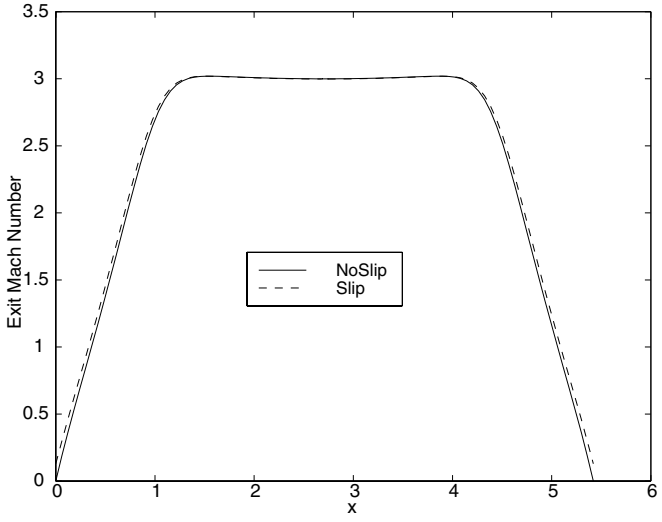


Figure 3.16: Exit plane Mach profiles for the 5.4:1 nozzle at a Reynolds number of 887

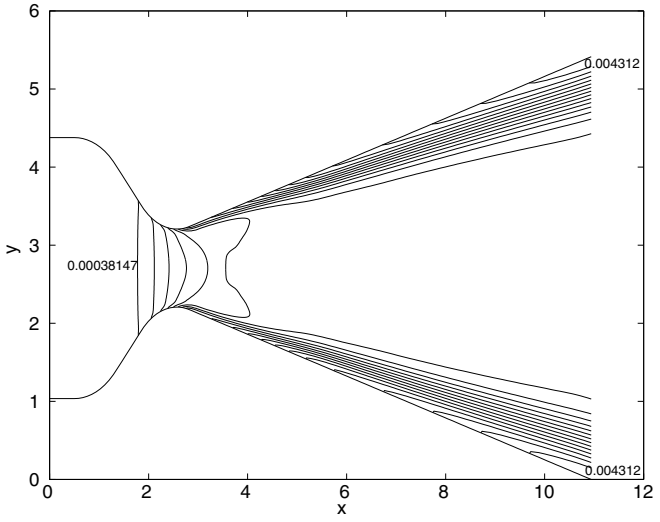


Figure 3.17: Knudsen number contours for the 5.4:1 nozzle at a Reynolds number of 1578

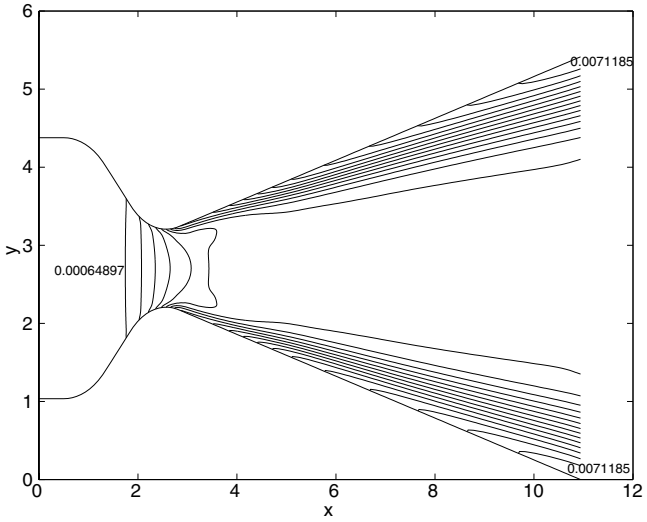


Figure 3.18: Knudsen number contours for the 5.4:1 nozzle at a Reynolds number of 887. The contours verify that slip velocity is not dominant because $Kn < 0.03$.

3.7 Boundary Layer Analysis and Three-Dimensional Effects

The previous section detailed the performance parameters that highlight the effects of viscosity in micronozzle flows. As was shown, there is a profound influence, particularly for throat Reynolds numbers below 1000. In this section, the boundary layers are subject to closer examination. By determining the momentum and displacement thicknesses, the effective area ratio as well as the laminar or turbulent nature of the boundary layer can be inferred.

The displacement thickness or δ^* is the amount that streamlines outside of the boundary layer will deflect due to the loss of mass flow in the boundary layer. It is computed by integrating the continuity equation normal to the wall. Due to the supersonic nature of the flow the definition of displacement thickness for a compressible fluid [29] is used:

$$\delta^* = \int_0^\infty (1 - \frac{\rho u}{\rho_{\text{edge}} u_{\text{edge}}}) dy \quad . \tag{3.40}$$

As stated in equation (3.40), the integration is performed along y, which is normal to the wall. The limits of the integral are the wall ($y = 0$) and the point at which core conditions are reached ($y = \infty$). However, due to the diverging nozzle walls, the core velocity will have a velocity component normal to the wall. Therefore, the boundary layer is integrated to just outside of the shear layer or to $\delta_{99\%}$, where the velocity is 99% of the core, and only the component of velocity parallel to the wall is considered.

The momentum thickness (θ) represents the momentum deficit due to the boundary layer. It is computed in a similar fashion to δ^*, but is formulated as:

$$\theta = \int_0^{\delta_{99\%}} \frac{\rho u}{\rho_{\text{edge}} u_{\text{edge}}} (1 - \frac{u}{u_{\text{edge}}}) dy \quad . \tag{3.41}$$

3.7.1 Test Case Boundary Layers: Determining Effective Area Ratio

Figure 3.19 shows δ^* as it grows along the nozzle wall for a range of Reynolds numbers. The displacement thickness and the distance along the wall are normalized by the throat width. The boundary layers are nearly linear along the nozzle length (outside of the contoured region near the throat), and thicken rapidly with decreasing Reynolds number. At the exit of the nozzle for which δ^* is depicted in figure 3.19, the geometric width is 16.9. The geometric width is the actual value measured on the nozzle. From the figure, the boundary layer thickness is 1.8 at $Re = 3721$, which is 3.6 when both walls are taken into account. Thus, the blockage is 21% of the 2-D area. However, for the lowest operating Reynolds number, the blockage has grown to 38% of the exit width.

The 2-D effective area ratio of the nozzle is the ratio of the effective exit area to the effective throat width. The effective quantities are the geometric width reduced by the amount of δ^*. This quantity is computed for the 15.3:1 and 16.9:1 nozzles in figures 3.20 and 3.21, respectively. As the boundary layer thickness grows downstream, the local cross-section is reduced, and the effective expansion ratio from throat to exit deviates further from the geometric area ratio. The blockage has changed the effective area ratio of nozzle from 16.9 to 14.1 and 10.2 operating at the highest and lowest Reynolds numbers respectively.

If a Quasi 1-D analysis is applied to the effective area ratio, the resultant thrust efficiencies deviate 0.96% and 1.6% from the simulation for the highest and lowest Reynolds numbers respectively. Figure 3.22 compares the performance computed by direct integration of the state variables in the finite volume computation (as in equation 3.38), and those computed by a Quasi 1-D analysis based on effective area ratio. This figure illustrates the utility of the effective area ratio. Once δ^* is known, efficiencies can be predicted to within 2%.

At this point the numerical analysis predicts minimal reduction in thrust and mass flow performance due to the boundary layer blockage over the full range of Reynolds numbers. As will be seen later, the experimental results are markedly different due to a breakdown of the 2-D assumptions. It is important to point out that the change in area ratio only changes the effective exit velocity by a few percent. The major impact of the blockage is the reduction of throat area, which in turn reduces mass flow, and subsequently

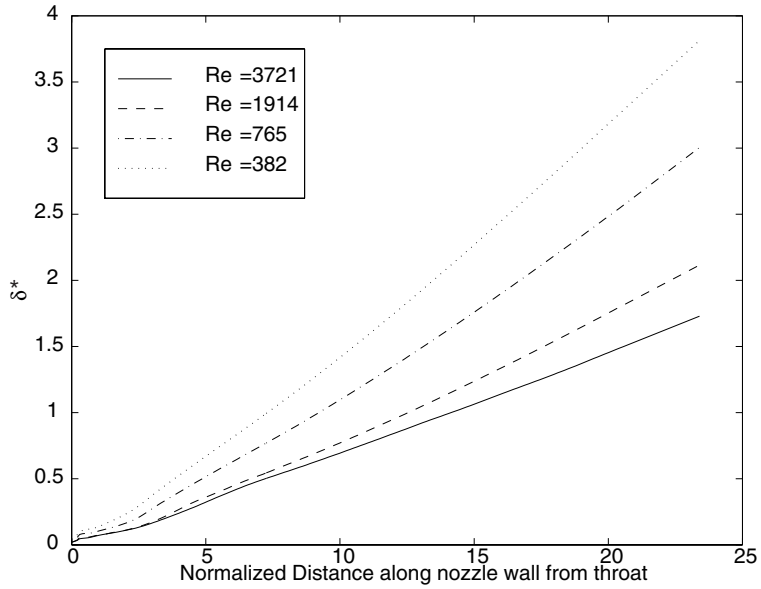


Figure 3.19: Displacement thickness growth from throat to exit along the nozzle wall for the 16.9:1 nozzle. Distance along the nozzle wall and displacement thickness are normalized by the throat width. The boundary layer thickness increases significantly with decreasing Reynolds number. At the lowest Reynolds number, the boundary layer accounts for 38% of the exit plane.

thrust efficiency. At the highest Reynolds number, the increased blockage is 3% of the throat width. This increases to 8% by the lowest Reynolds number. In comparison, the exit velocity decreases only 3% for the lowest Reynolds number, even though the effective area ratio was reduced by 40%. Once the effective area ratio is below 7, the exit velocity, and hence I_{sp} efficiency will substantially decrease.

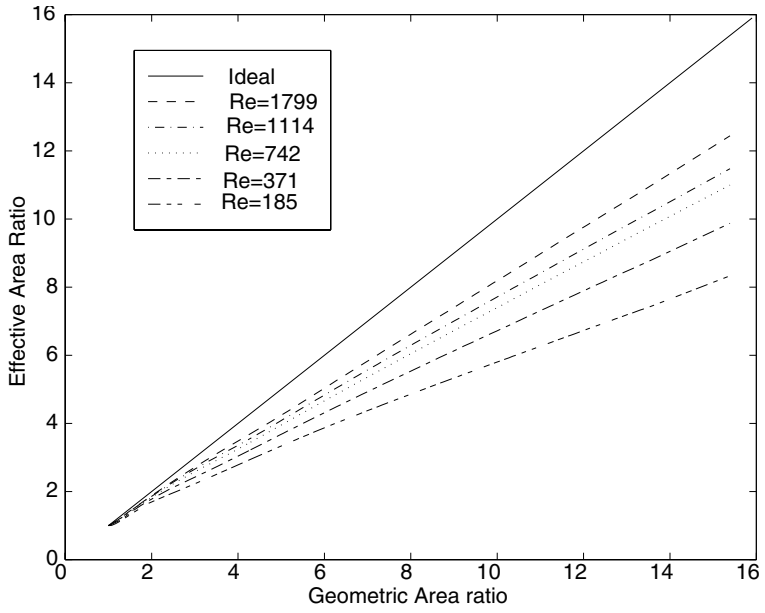


Figure 3.20: Variation of effective area ratio with actual geometric area ratio for the 15.3:1 nozzle. The effective area ratio is calculated by reducing the local width by the amount of the displacement thickness on both walls

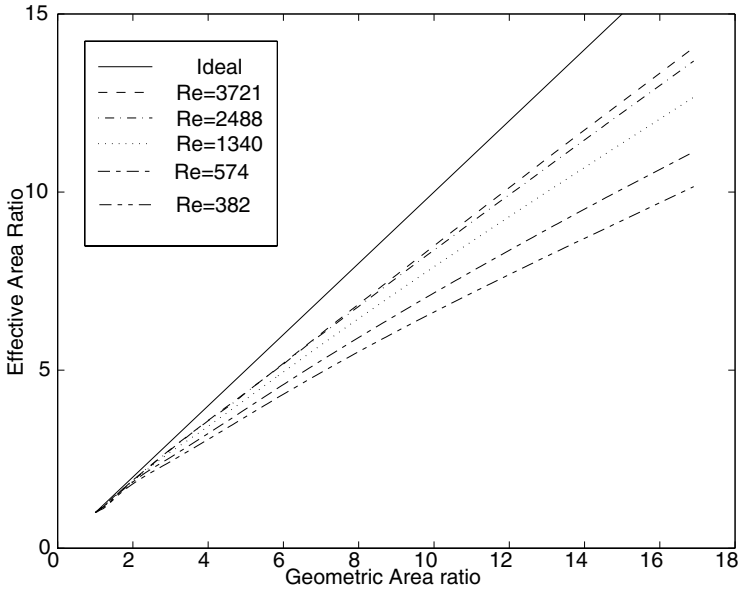


Figure 3.21: Variation of effective area ratio with actual geometric area ratio for the 16.9:1.

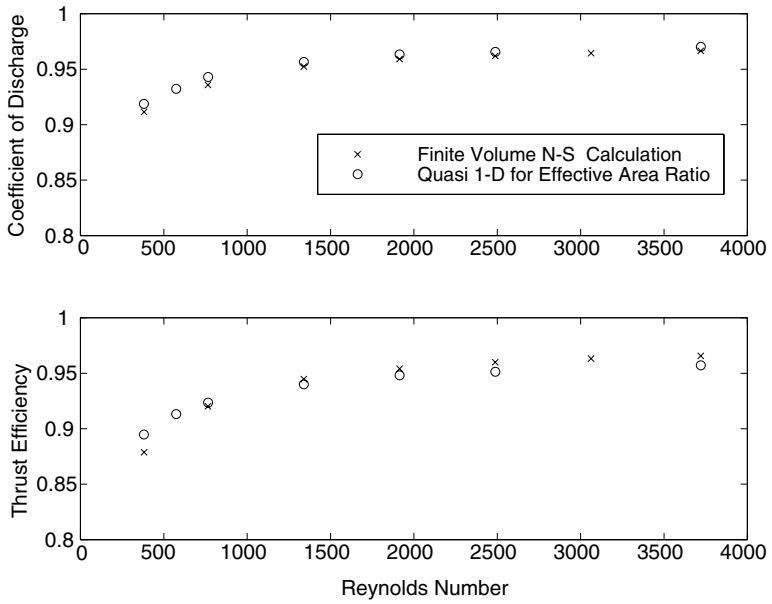


Figure 3.22: Thrust and I_{sp} efficiencies are computed from numerical simulations. These are compared to inviscid calculations of similar nozzles operating at the same effective area ratio as depicted in figure 3.21

3.7.2 Evaluation of Nozzle Separation and the Laminar Flow Assumption

Figures 3.23 depicts the momentum thickness profiles for the largest area ratio nozzle. As is generally found in a compressible flow, this is approximately 10% of the displacement thickness. The ratio of the displacement thickness to momentum thickness, or shape factor, is a good indicator for boundary layer separation in incompressible flows, but the acceptable range is poorly defined for compressible flows. It is sufficient to show that the $\frac{dP}{dx} < 0$ to guard against separation. Since the effective area ratio is increasing along the nozzle, there is a favorable pressure gradient, and this eliminates the possibility of separation.

Finally, the laminar or turbulent state of the flow can be inferred by comparing the flow characteristics to those of a flat-plate analysis. The stability of a flow can only be determined through a compressible stability analysis. However, flat plate cases, can be used to generalize the behavior based on Mach number and Reynolds number based on momentum thickness. Figure 3.24 depicts the variation of Reynolds number with distance along the nozzle throat for the 16.9:1 nozzle. Mack [30] depicts the neutral stability curves for high Mach number flows. For a Mach number of 3.8, the minimum neutral stability point is at $Re_\theta = 300$. The highest value found in the numerical simulations (for the 16.9:1 nozzle operating at a throat Reynolds number of 3721) is $Re_\theta = 121$. Thus, the laminar assumptions that are used to formulate the viscous stresses in simulation, appear validated.

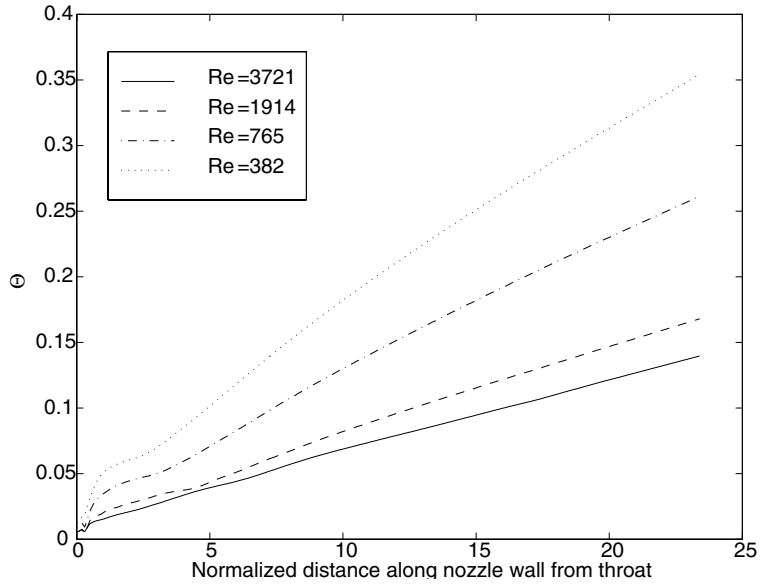


Figure 3.23: Momentum thickness growth from throat to exit along the nozzle wall for the 16.9:1 nozzle. Θ is normalized by the throat width, as is distance along the nozzle wall.

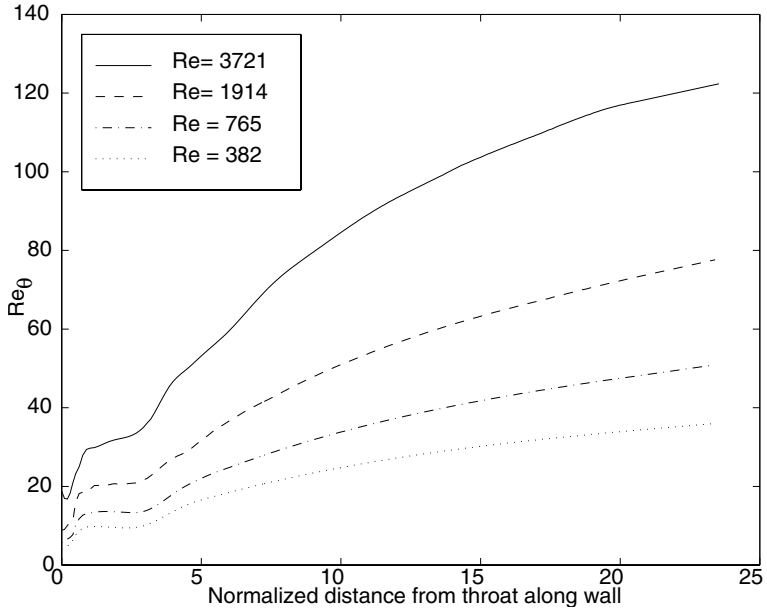


Figure 3.24: Re_Θ variation from throat to exit along the nozzle wall for the 16.9:1 nozzle. This is well below the stability point for a Mach 3.8 flow of $Re_\Theta = 300$ as computed by Mack [30], for all cases.

3.7.3 Wall Roughness Effects

Since the wall roughness is a larger fraction of the local width than in conventional nozzles, its effects on momentum loss should be considered. The greatest loss would occur if the roughness induced shocks. This would result in a pressure rise, and a subsequent momentum loss. The first question that needs to be answered is whether or not the physics of flow at this scale will allow a shock to occur. By looking at the definition of viscosity, a generalized statement can be made about the ability of the flow to shock. As found in White [29], viscosity is defined by:

$$\mu \sim \rho c \lambda \tag{3.42}$$

where c is the molecular speed, and λ is the mean free path. Non-dimensionalizing by the roughness height, l, the following relation can be established:

$$\frac{\mu}{\rho u l} \sim \frac{c}{u}\frac{\lambda}{l} \tag{3.43}$$

$$M \sim ReKn \tag{3.44}$$

where u is the local velocity, and ρ is the local density.

Thus, if the product of the Knudsen number and Reynolds number based on the roughness height is less than one, then the Mach number will be less than one, and a shock will not form.

For the cases of interest, the Reynolds number based on the roughness varies from 0.1 to 10 over the range of Reynolds numbers and area ratios. However, the Knudsen number based on roughness varies from 0.1 to 5. Thus, it is possible to generate a shock at the scale of the roughness. But a shock will only be generated if the roughness height perturbs the supersonic region of the flow. If the roughness is buried within the subsonic boundary layer a shock should not form. Figure 3.25 depicts the distance of the sonic line from the wall along the wall normal for each local area ratio, from simulations. If the roughness is smaller than this value, then a shock will not occur. Only the point at the throat for the smallest nozzle, operating at the highest Reynolds number, violates this condition. In all other cases, they roughness is well inside the subsonic boundary layer.

Since shock formation should not be a problem over nearly the entire range of operation, the impact of wall roughness on friction should be considered. Nikuradse [31] has shown that in the hydrodynamically smooth regime, the friction coefficient is only a function of Reynolds number, and not wall roughness height. This regime is defined by the following relation:

$$\frac{k_s v_*}{\nu} < 5$$

where

$$v_* = \sqrt{\frac{\tau_w}{\rho_w}}$$

This is effectively the Reynolds number based on the roughness height (k_s), where v_* is the friction velocity based on shear stress and fluid density at the wall. The value of this indicator was found for the operating conditions to be 1.98. Due to the low Reynolds number nature of the flow, the nozzles operate in the hydrodynamically smooth regime, and the losses due to friction computed in the Navier-Stokes simulation are sufficient.

3.7.4 Divergence Losses vs. Boundary Layer Blockage: A Trade Study

The divergence angle affects the magnitude of exit momentum aligned in the axial direction, and thus contributes to thrust. Momentum in the streamwise direction is lost as useful thrust. However, as divergence angle decreases, the nozzle length increases, and the boundary layer blockage will also increase. This leads

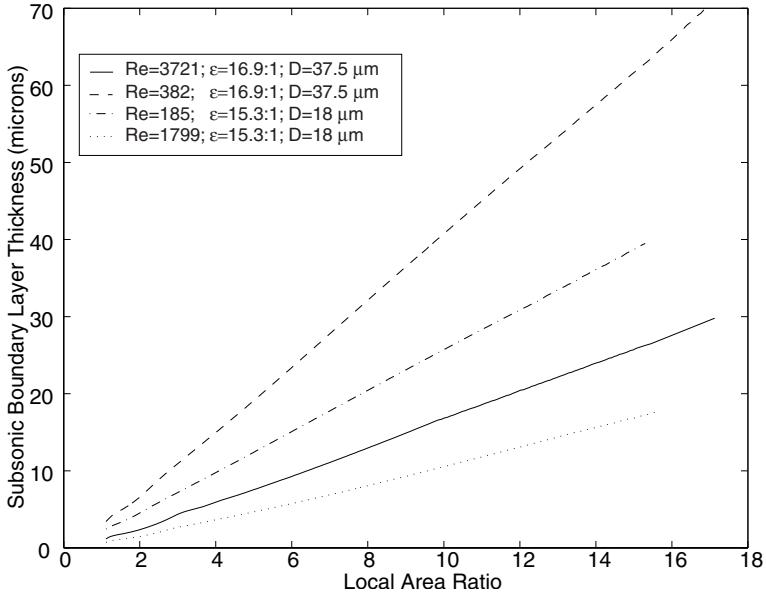


Figure 3.25: Subsonic boundary layer growth from throat to exit for various expansion ratios and operating conditions.

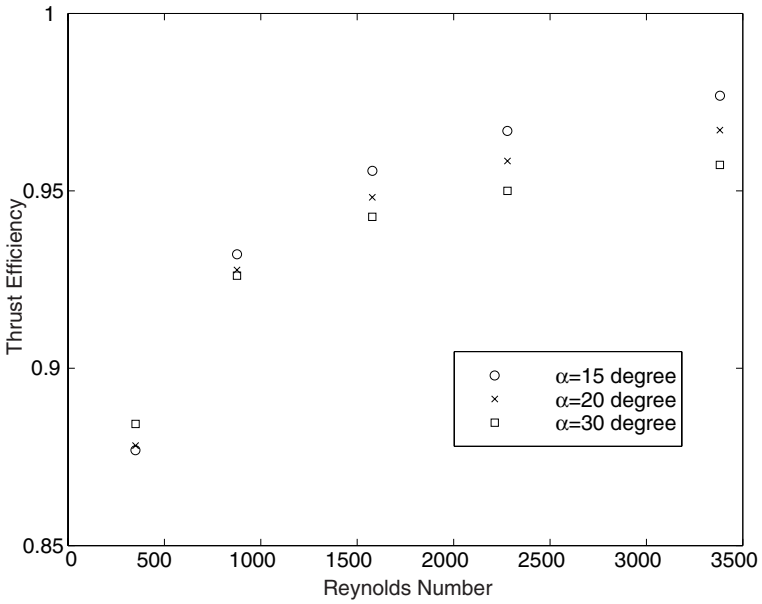


Figure 3.26: Thrust efficiency variation with Reynolds number for different nozzle divergence angles. Notice that the higher angles perform better at low Reynolds numbers.

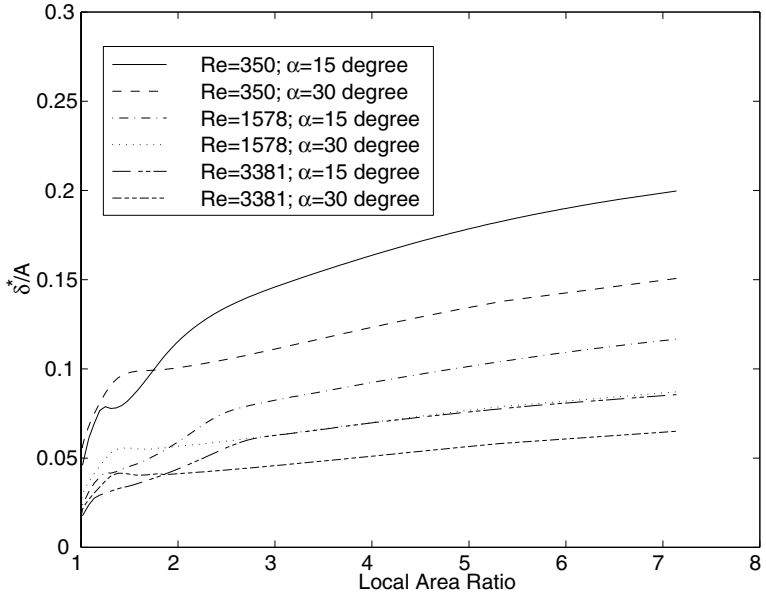


Figure 3.27: Displacement thickness as a fraction of the nozzle local area along the nozzle.

to a trade-off between blockage and divergence loss. To investigate this, computational cases were run for 15, 20, and 30 degree divergence nozzles, while keeping the expansion ratio set at 7.1:1.

Figure 3.26 depicts the performance variation with Reynolds number for the different angles. At the highest Reynolds numbers, the highest divergence angle has a 2% lower thrust efficiency than the lowest angle. The boundary layers in this region are relatively thin, and divergence loss is dominant. Below Reynolds numbers of 1000, the boundary layers are so thick that the dominant loss is from the blockage, which is reflected in the improved performance of the 30 degree nozzle.

This trend is most evident in the examination of the displacement thickness profiles, which are depicted in Figure 3.27. At the low Reynolds numbers, there is a 33% difference between the highest and lowest divergence angle displacement thickness. Since this is a 2-D calculation, there will be a reduction in device performance when the displacement thickness of the endwalls are considered. When the blockage is applied to the upper and lower surface, the effective area ratio of the 15 degree case will drop faster than the 30 degree case due to the larger displacement thickness. Thus, the difference between the curves at high Reynolds numbers will be less (possibly within 1%), and the higher angles will perform much better at low Reynolds numbers. Ultimately, the 30 degree nozzle will perform the best over the full range, but if operation is restricted to high Reynolds numbers, the lower angles are best. The limiting factor will be the operating Reynolds number, which determines the boundary layer development and ultimately the effective area ratio.

3.8 Summary

This chapter describes the numerical analysis implemented to investigate the performance of micronozzles over a range of operating conditions. A two-dimensional finite-volume discretization of the Navier-Stokes equations was used to determine the steady-state flow field for specified boundary conditions. The governing parameter is the throat Reynolds number, which is a function of the mass flow through the nozzle, and in turn is set by the plenum conditions. Nozzle thrust and mass flow are determined from the state variables computed in the simulations.

The results of the simulations can be summarized as follows:

- The micronozzle flows are laminar, and do not separate for all Reynolds numbers (177-3800) and Area Ratios (5.4, 7.1, 15.3, and 16.9) investigated in the 2-D simulation.
- The performance is not affected by slip boundary conditions for the cases considered.
- 2-D nozzle performance exhibits thrust, I_{sp}, and mass flow efficiencies above 90% for Reynolds number above 1000.
- Boundary layer induced blockage accounts for the reduction in efficiency with Reynolds number.
- The reduction in mass flow efficiency is dominant because it is a linear function of the blockage. I_{sp} efficiency is less affected because the exit velocity is a weak function of the effective area ratio until it is reduced below 7.
- A 20 degree expansion angle offers the best trade-off between divergence losses and viscous losses over the operating range tested. As the operating Reynolds number increases, the optimum angle will decrease.

Chapter 4

Cold-Gas Thruster Fabrication

4.1 Microfabrication Description

Microfabrication is the prime tool used in the development of Microelectromechanical systems. It is the geometrical or chemical alteration of a base material through etching or diffusion to create structures that enable a desired electrical or mechanical behavior. These processes were initially developed by the microelectronics industry to facilitate the batch-fabrication of integrated circuits on the most popular semiconductor, silicon. Silicon is an excellent working material in its single crystal form. It has a very high fracture strength and high thermal conductivity.

The scale of these devices are typically limited between the size of the silicon wafer used and the resolution of the lithography used to pattern features of interest. At MIT, 100 mm diameter wafers (soon to be converted to 150 mm) ranging anywhere from 300 microns to 1mm thick (500 microns is standard thickness) are used for processing. The lithography step, described later, can resolve features as small as 2 microns when operated with a standard ultraviolet source with a contact mask. These limits are guidelines for the fabrication of devices that are primarily created through the techniques of surface micromachining, bulk micromachining and wafer bonding.

4.1.1 Surface Micromachining

Surface micromachining can be generalized as the build-up of structures through the deposition and selective removal of thin films. These films can be a chemically altered form of the silicon substrate such as oxide or nitride, a deposited layer of metal, such as gold or nickel, or a deposited layer of silicon in its polycrystalline form. Surface micromachining is primarily used for thin films and the creation of CMOS devices in integrated circuits, but has found application in the MEMS community for creating suspended beams and membranes. Additional applications, and a more detailed account of surface micromachining can be found in Guckel and Burns [32].

4.1.2 Bulk Micromachining

Bulk micromachining refers to the selective removal of the silicon substrate to create three dimensional structures. These structures are generally of much higher aspect ratio than fabricated using surface micromachining. This material can be removed through a physical means such as sputtering or ion milling, which relies upon the momentum of an inert particle to induce surface material removal [33]. Bulk micromachining can also refer to a chemical process, such as a wet etch, to remove the base material. The effectiveness can be generally categorized by the anisotropy of the etch, that is, how preferential the etch is in the direction of interest relative to the etching in the other directions.

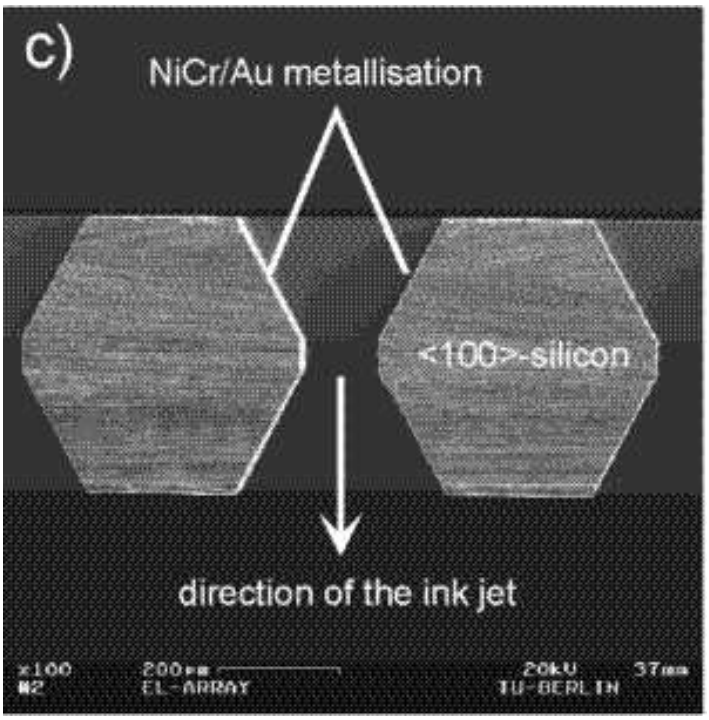


Figure 4.1: Example of anisotropic etching of silicon with KOH using oxide as a masking material. This results in square nozzles with at 32.5 degree divergence. Notice the sharp corners at the throat. [34]

Anisotropic Chemical Etching

As described in Chapter 1, the first batch-fabricated micronozzles were created by the anisotropic etching of silicon with KOH [14]. The silicon can be masked by oxide against the KOH etch. This etch is anisotropic because the silicon crystalline planes in the $< 100 >$ direction are etched preferentially to those in the $< 111 >$ direction at a ratio of 400:1 at an etch rate of 1.4 microns/minute. Thus, the resultant feature will lie almost exactly along the $< 111 >$ crystalline plane, and will be at an included angle of 57.4° to the surface. Figure 4.1 depicts the results of such an etch on a square feature etched into oxide with silicon as the underlying substrate.

This technique has been used to manufacture inkjet printheads due to the ability to maintain tight tolerance on such small features. If this feature is etched from both sides, in a well-aligned manner, a converging-diverging nozzle can be created which will allow an acceleration of gases to supersonic velocities. The drawbacks are that the etch is limited to the 32.5° expansion and the thickness of the wafers available. Neither of these are optimal in creating a nozzle specifically tailored to achieving supersonic velocities.

Deep Reactive Ion Etching

Reactive ion etching combines physical and chemical etching techniques by accelerating an ion which reacts with silicon on the surface. This etching provides high etch rate, anisotropy, and selectivity of etching silicon relative to the masking material. An alternative is now available for significantly improving the performance of a conventional RIE. The time-multiplexed deep etcher (TMDE), patented by Robert Bosch, Gmbh [35], utilizes a cycle alternating between the etchant SF_6 and the passivation material C_4F_8. Surface Technology Systems (STS) has employed this technique to create an etcher that can achieve trench aspect ratios of 30:1. The STS etcher consists of a 1000 Watt RF power source which is a single coil around the chamber used to create the high density plasma. In addition, a 300 Watt supply connected to the wafer electrode can be used to bias the potential of the wafer with respect to the plasma. Thus, the energy of the ions reaching the

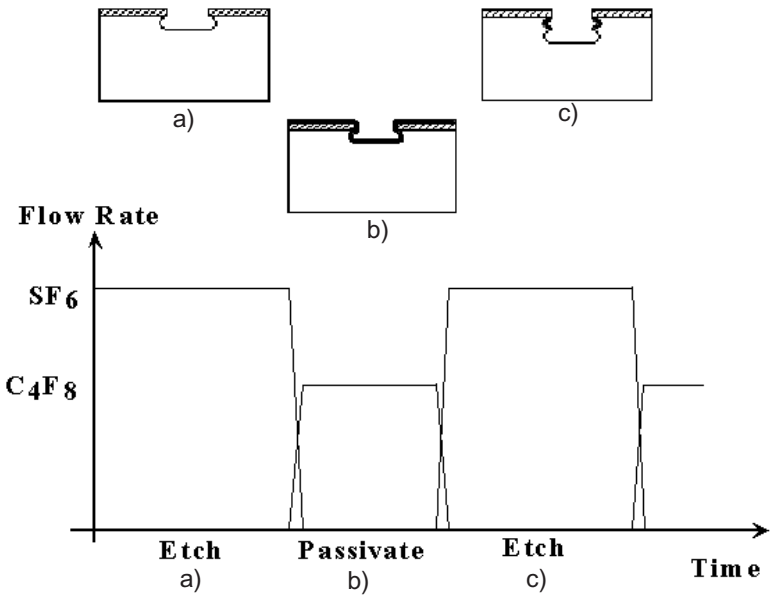


Figure 4.2: Gas flow cycles during plasma etching. Overlaps occur due to finite response times of flow controllers [36]

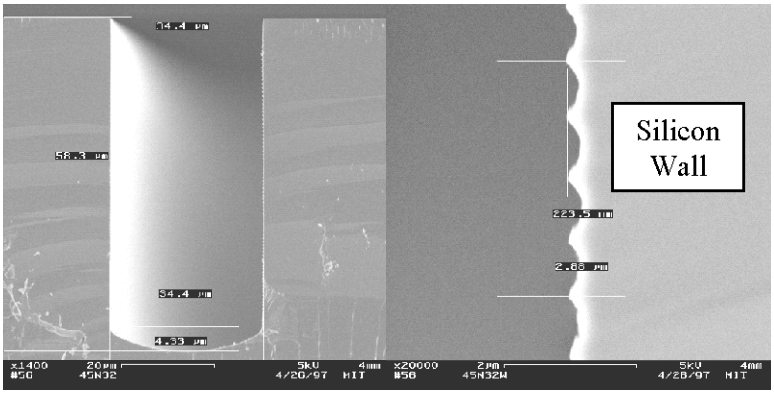


Figure 4.3: Micrograph view of an anisotropic etch (left) and the scalloping observed on the walls due to the periodic etch/deposition[36]

substrate, as well as ion flux, can be controlled.

The etching cycle, shown in figure 4.2 [36], illustrates the TMDE scheme. The etching occurs during the SF_6 flow, which occurs laterally as well as vertically. The polymer passivation occurs during the C_4F_8 flow. When the cycle reverts back to the SF_6, the passivating film is preferentially removed at the bottom of the trenches due to ion bombardment, while preventing the etch of the sidewalls. Thus, the lateral penetration depends upon the etch recipe, but is on the order of 300 nm. The etch rate of silicon depends upon feature size as well as aspect ratio. The tailoring of this recipe to the features of interest are described in a future section and in greater depth by Ayon et al. [36]. The results are shown in figure 4.3. This figure also depicts the scalloping that arises due to the cyclic etch and passivation scheme.

4.1.3 Wafer Bonding

Surface and bulk micromachining can be augmented with the ability to bond wafers together in order to convert etched trenches into sealed cavities. Borosilicate glass can be bonded to silicon with an anodic bond, and two silicon wafers can be joined through fusion bonding. Both processes are described in the following

subsections.

Anodic Bonding

With flow channels etched into the silicon substrate, a capping wafer must be bonded so a cavity is formed and the flow is contained. In some instances, optical access to these channels is desired. In addition, glass offers a lower thermal and electrical conductivity than silicon. Pyrex is generally used because it maintains a similar coefficient of thermal expansion, reducing residual stress in the bond. Recently, other types of borosilicate glass have been developed which permit a matched coefficient of thermal expansion to higher temperatures.

This process was first described by Wallis and Pomerantz [37]. The bond is formed by placing an RCA cleaned ($NH_4OH : H_2O_2 : H_2O$: (1:1:3) followed by $HCl : H_2O_2 : H_2O$ (1:1:3)) silicon wafer in contact with a piranha ($H_2SO_4 : H_2O_2$ (3:1)) cleaned glass wafer. Both of these cleans will remove organic contaminants and particulates from the surfaces. The mated pair is heated to $400 - 500$°C, 3 bars of pressure is applied, and an electric potential of 1000 V is placed across the stack. The borosilicate glass contains mobile sodium ions which migrate toward the negative electrode away from the silicon-glass interface. The field, created by the negative charges along the interface, assists in the chemical bonding of the surfaces. Though bond strengths are not explicitly reported by Wallis and Pomerantz, they suggest bond strengths above 2000 psi have been achieved. Rogers [38] investigates the requirements and constraints for anodic bonding to a greater depth.

Fusion Bonding

If the capping wafer is silicon and not glass, a fusion bond is used to join the surfaces. This process is similar to the anodic bond in that the contacting wafers must be specularly cleaned prior to contact and possess a surface roughness ~ 1 nm, thus a mirror polished wafer is required. Particulate in either process will prevent bonding locally.

The fusion bond relies upon the RCA clean due to its ability to create a hydrophilic surface. Grunder [39] examined silicon surfaces after this clean and found a high concentration of OH groups attached to the dangling bonds on the wafer surface. The attractive force, albeit weak, between the two wafer surfaces begins with hydrogen bonds between the OH groups. A permanent bond can only be formed with a high temperature annealing step. The anneal serves to dehydrate the interface. At about 200°C, this forms a siloxane or ($Si - O - Si$) bond and water as a byproduct. As the temperature increases, the oxygen diffuses into the substrate as a trapped atom, and the siloxane is reduced to covalently bonded $Si - Si$ atoms. The strength of the bond is a function of the annealing temperature. Ristic [40] quotes this in terms of the surface energy to be 500 erg/cm^2 for low temperature bonds ($< 600°$ anneal) and 1000 erg/cm^2 for high temperature bonds ($> 1000°$ anneal). The applicability of bonded sensors is explored in depth by Schmidt [41].

4.2 Process Flow for Micronozzle Fabrication

The micronozzle process flow is straightforward in principle. The challenge lies in attaining a highly variable geometry that maintains anisotropy over the full height of the structure. Large features etch faster than smaller features, which causes the smaller features to distort when performing a deep etch. If small features of a constant geometry are etched, wall roughness becomes accentuated. These trade-offs are examined in the following sections. This section presents the process which is the backbone for the micronozzle fabrication. The fabrication goals are:

- Create a micronozzle with a throat less than 20 microns and an expansion ratio of 20:1
- Perform the nozzle etch to a depth of 500 microns
- Maintain less than 1 micron variation in feature size over the etch depth
- Maintain wall roughness less than 1 micron to prevent shocks

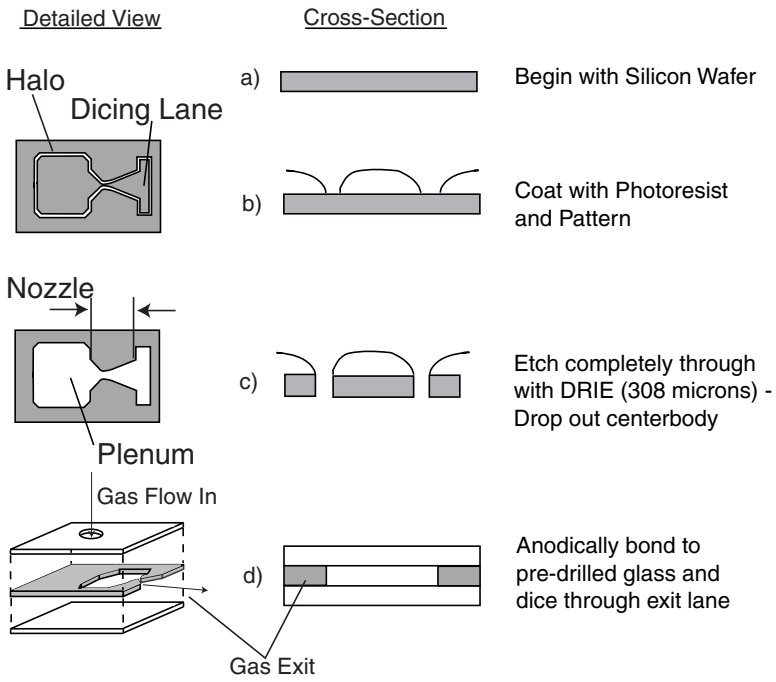


Figure 4.4: Cross-section of the silicon device wafer at key points in the fabrication process. a) begin with a clean $< 100 >$ silicon wafer b) coat in photoresist and pattern through exposure through features in a mask c) DRIE through nozzle feature d) Strip resist and anodically bond to glass

The process sequence is shown in Figure 4.4, in cross section. The following steps detail this process:

- A clean double-polished $< 100 >$ wafer is protected with photoresist that can be applied with thicknesses up to 12 microns. This is done by spin coating the resist at 1000 rpm.
- The resist is patterned by an exposure of 370 seconds for a power density of 6 mW/cm^2 at a wavelength of $460nm$ and developed in MIF-440 developer for 6 minutes. The pattern is a trench of constant width (known as a halo), which defines the geometry of interest.
- With the pattern transferred through lithography, the resulting features are etched through the wafer using DRIE. Typical etch rates are measured between 2 to 3 microns/min depending on feature size and recipe. The silicon wafers are mounted to quartz wafers with photoresist to prevent the backside helium coolant from leaking when the features etch through the wafer. The quartz allows the cleared features to be detected by inspection since it is optically clear.
- The resist is stripped and the handle wafer separated through a piranha clean.
- The upper surface is anodically bonded to a Pyrex wafer at 500° C with 1000 Volts and 2400 mbars of pressure applied across the stack. This task is accomplished with an Electronic Visions/Aligner Bonder. The aligner permits the 1 mm gas injection holes, which are pre-drilled into the glass, to line-up with the chambers. The glass wafer has an auxiliary hole drilled in it so that electrical contact can be made with the silicon through a brass spring, when the second glass wafer is bonded to the stack. This contact allows a voltage to be applied only relative to the second glass wafer and the silicon.
- Finally, the wafer is diced along lanes which intersect the nozzle, but prevent the nozzle geometry from being distorted by inaccurate blade alignment. The recesses are 1 mm wide as to not affect the expansion of gases from the critical geometries. The dicing exposes the flow channel permitting the gas to discharge from the nozzle through the edge of the die. The exit cavity of nozzle is shown in figure 4.5.

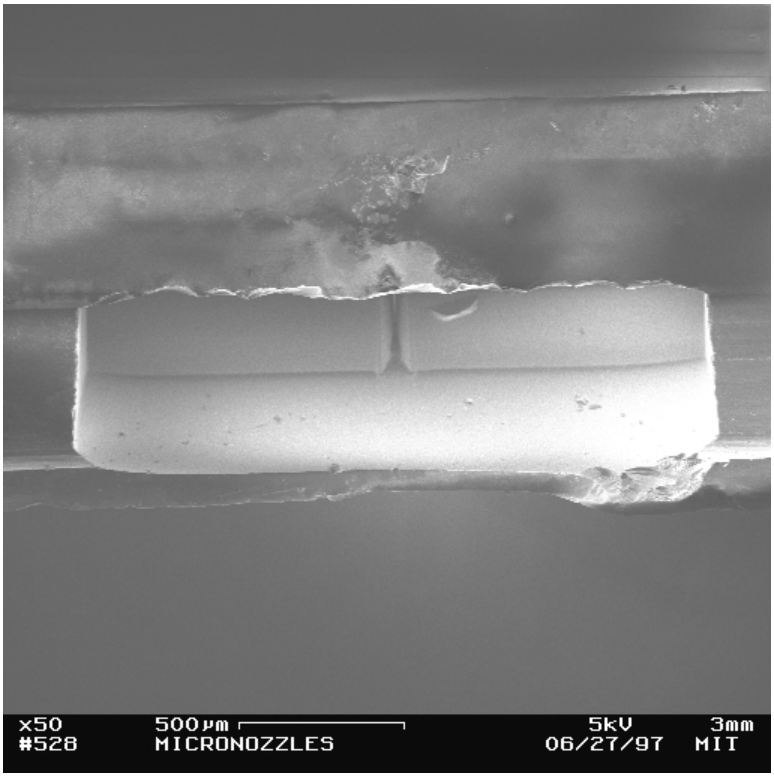


Figure 4.5: SEM of the nozzle exhaust cavity. Nozzle exit plane is small opening centered in the cavity.

The die layout was chosen so that its package could be integrated with the macroscopic gas-delivery system. If the bonded silicon wafer was diced into individual nozzles, the die would be less than 2 mm^2 apiece and difficult to handle. Thus, there are 9 nozzles per die as in figure 4.6. Also, if the gas were to be injected into the chamber along the nozzle axis, the interface would be a 300x1200 micron rectangular orifice. Outside of a very difficult bond to small diameter tubing, there is not a reasonable packaging solution for axial injection to a single nozzle, and thus the injection hole is perpendicular to the nozzle chamber, and multiple nozzles are on a die.

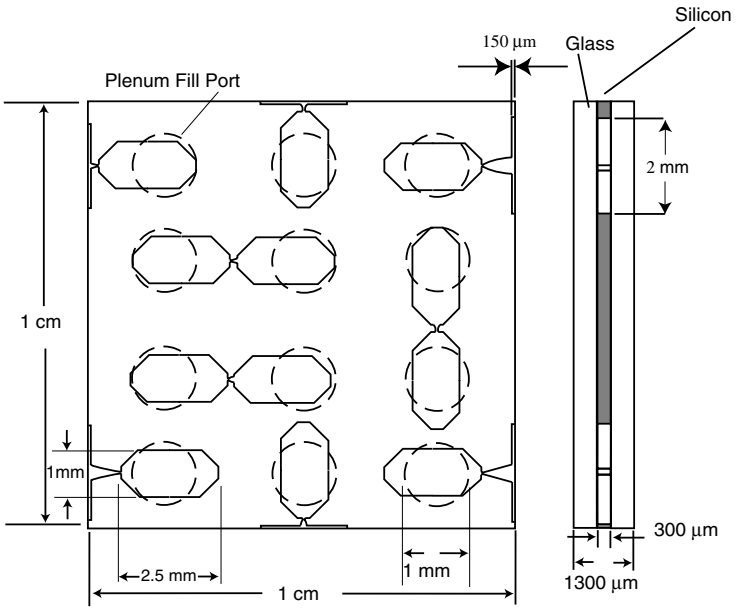


Figure 4.6: Arrangement of individual nozzles in a die. Six discharge to the environment, and the three in the center discharge into an on-chip cavity for benchtop testing.

4.3 Process Analysis and Enhancement

This section reviews each of the improvements made in the fabrication of the nozzles, and the resulting effects. The largest effort was focused toward overcoming the aspect ratio dependent etching characteristics of plasma etchers. As mentioned previously, this causes larger features to etch faster than smaller features. A test of 5 different recipes (variable coil power, flow rates, and electrode power) shows that almost every option results in this loading effect. Figure 4.7, from Ayon et al. [36] illustrates this trend.

4.3.1 Full Geometry Etch

The first trial attempted to etch the full nozzle-plenum geometry simultaneously, the results are shown in figure 4.8. The DRIE performs well for about 200 microns depth with the full geometry. There is noticeable loading effect at the throat, which is designed to have a width of 8 microns. The etch rate is 0.3 microns/minute less than the larger features. Ultimately, the throat region becomes distorted through the overetching that is necessary to clear it. This distortion is obvious near the bottom of the throat in Figure 4.9. This loading effect can be tailored through optimization of the etch parameters to provide a degree of three-dimensionality to the nozzles. By lowering the inlet and outlet divergence angles, the nozzle floor will gradually slope from the plenum to a minimum at the throat, and increase to a maximum at the exit. This would allow even further increase in the Reynolds number for a low-thrust propulsion system, as per the argument of equation (2.4).

Figure 4.10 illustrates an alternative nozzle geometry, the bell nozzle, etched to 100 microns. This shows that a well defined curvature can be maintained to this depth as well as highlighting the wall smoothness. This process results in functioning nozzles, but the geometry is distorted at large depths and prevents adequate comparison to the fluid model.

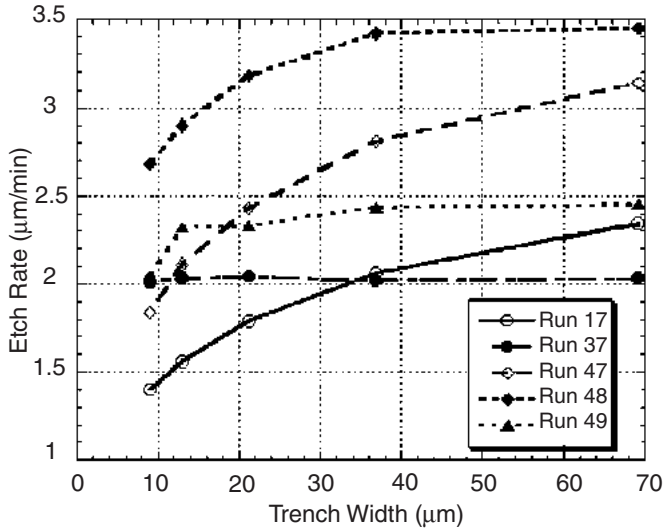


Figure 4.7: Variability of etch rate with feature size for different etch recipes [36]. This reduction in etch rate is termed the loading-effect. The recipe number corresponds to a proprietary combination of etch parameters documented by MIT.

4.3.2 Nested Masking

One attempt at reducing the loading effect was to use a nested mask set that allows the nozzles to be etched initially to a depth of about 40 microns before the larger plenum features are exposed. An oxide layer is patterned with the nozzles, and then coated with photoresist, which is patterned with the settling chamber. The STS etcher begins the etch in the nozzle region, and then the oxide is stripped in the settling chamber region with a buffered oxide etch. The wafer then goes back into the STS to complete the etch. The success of this scheme is too constrained to timing the etches to finish simultaneously, and once again leads to overetching and feature distortion. Figure 4.11 depicts the overetching which occurs due to the nozzle completing the etch prior to the settling chamber. Figure 4.12 shows the interface issues of etching a feature around a previously etched trench. Since there is polymer coating the sidewalls during the nozzle etch, it must be stripped in an O_2 plasma prior to the plenum etch. The remnants of the polymer cause the nozzle to be very rough where it interfaces the settling chamber. It is apparent from these results that success would only be possible through etching a constant feature size.

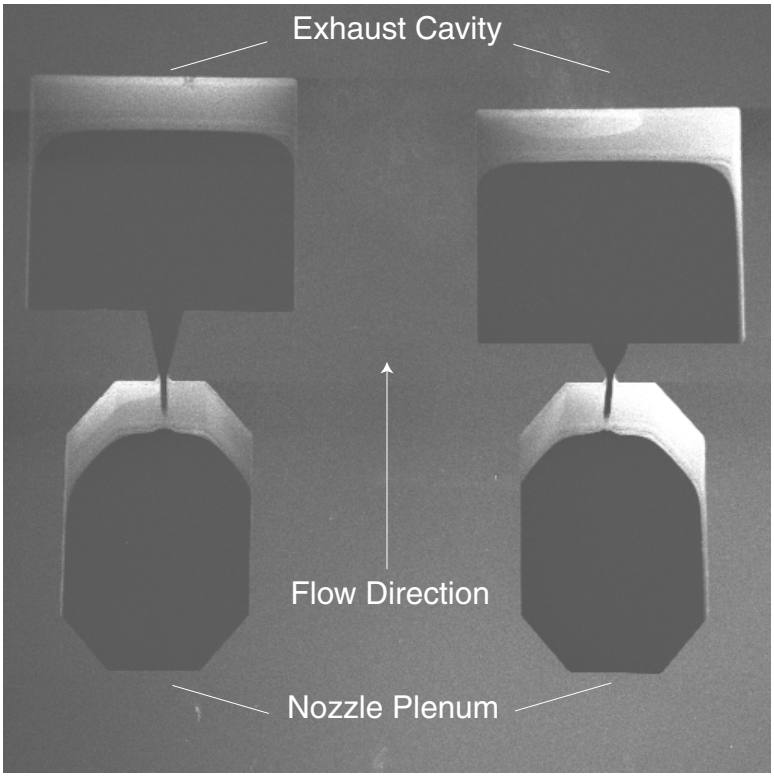


Figure 4.8: SEM of the nozzle-plenum system etched simultaneously for two different nozzle geometries

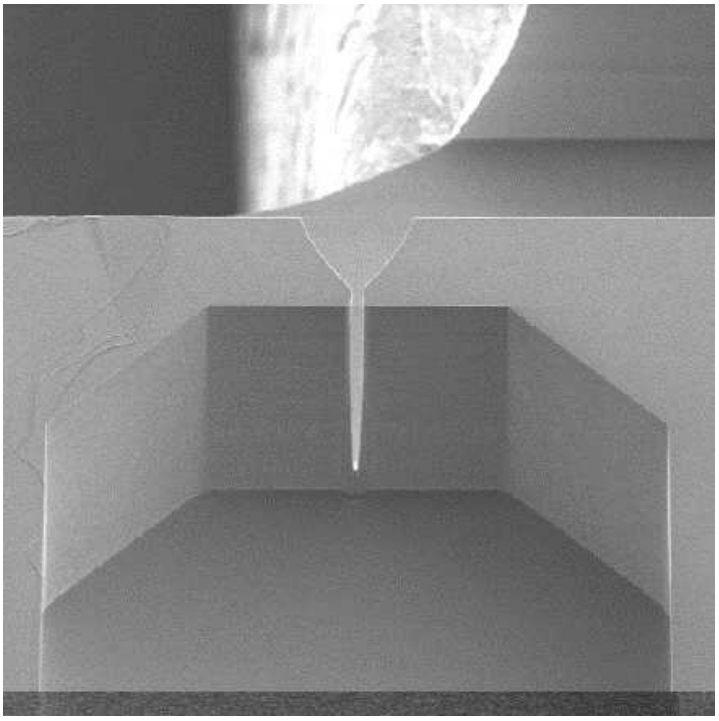


Figure 4.9: Close-up of a similar nozzle throat to that in figure 4.8

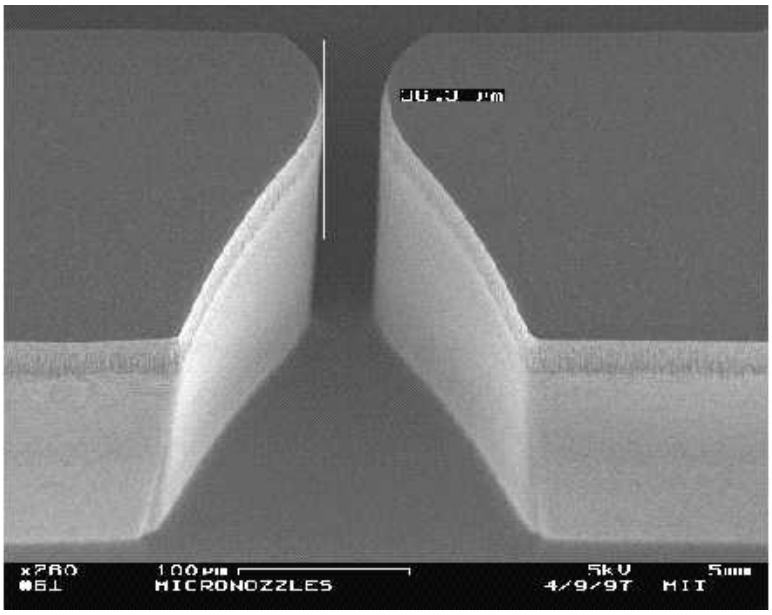


Figure 4.10: SEM of a bell nozzle exit plane, etched to 250 microns. Notice the smooth walls as well as the anisotropy.

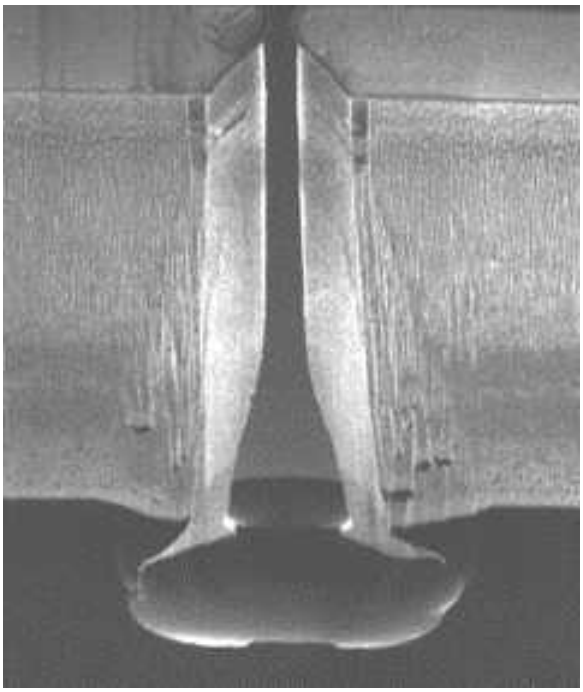


Figure 4.11: An example of overetching which occurs when a nested mask is not properly timed

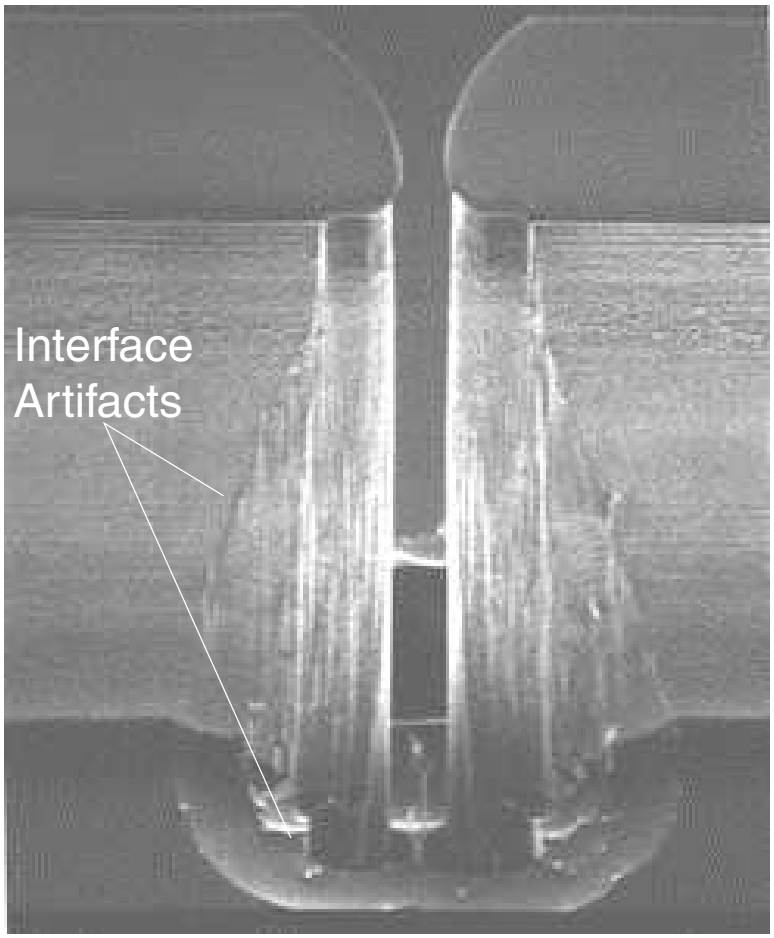


Figure 4.12: Poor feature interface due to nested feature not clearing properly

Table 4.1: Comparison of design nozzle geometries and the result after fabrication

Design Throat Width (μm)	Design Expansion Ratio	Fabricated Throat Width	Fabricated Expansion Ratio
10	10:1	19	5.4:1
10	25:1	18	15.3:1
25	10:1	34	7.1:1
25	25:1	37.5	16.9:1

4.3.3 Halo Masking

The optimal geometry was eventually achieved by etching a feature of constant size to maintain an equal loading. A halo mask accomplishes this. The nozzle outline is defined by a 10-micron trench outlining the feature of interest, as illustrated in figure 4.4b. If the etch is completely through the wafer, the centerbody will fall out during the resist strip and rinse cycle. Because of small feature distortion, best results are obtained for etches 300 microns in depth. Figure 4.13 is a micrograph of a nozzle for which this process was used. The 2-D nature of this nozzle is striking. There is only a 1-micron variation in feature size from the top to the bottom of the wafer.

The etch recipe was optimized for this particular geometry. As shown by Ayon [36] et al., higher coil power improves etch anisotropy and etch rate, at the cost of selectivity. Thus, the etches will appear more uniform, but the protecting photoresist is consumed much faster, and the achievable depth is reduced. The full geometry, in section 4.3.1, was etched with a coil power of 600 Watts, but the halo geometry achieved its best results for 800 Watts. The electrode power was 12 Watts during the etch cycle with the etchant flowing at 105 sccms and 6 Watts during the passivation, with the polymer flowing at 40 sccms. This results in an etch rate of about 1.6 microns per minute, greatly reduced from the 3 microns per minute that could be achieved for a 50 micron feature.

The wall roughness is larger for this type of mask due to the small halo trench. The roughness is on the order of 800 nm, where it had been about 300 nm with the previous processes. Since the resist tapers from its full thickness to zero thickness at the feature edge, as illustrated in figure 4.4b, the tapered region clears faster than the full thickness resulting in the increased roughness and feature enlargement during very long etches. The features are 10 microns on the mask and become 18 microns by the completion of the etch. It may be possible to mitigate this effect with an oxide mask. Other nozzles which were tested are shown in figure 4.14. All fabricated nozzles and their design values are listed in table 4.1.

Most of the design goals were met, with the exception of etch depth. Since coil power was increased to obtain high anisotropy, etch selectivity suffered, and depth was reduced. This was sacrificed to achieve the highest anisotropy possible.

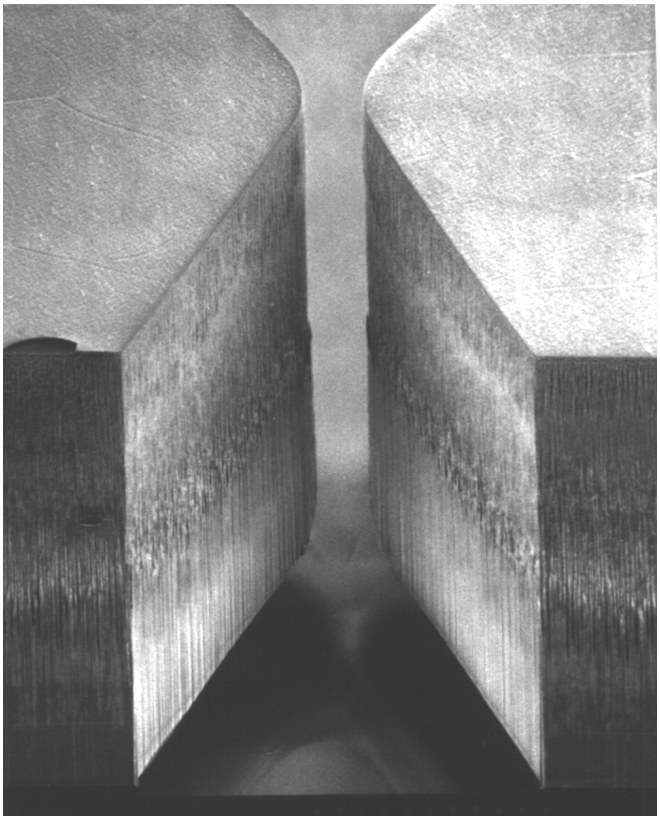


Figure 4.13: 5.4:1 nozzle with a 19 micron throat created using a halo mask. The depth of all nozzles in this section was 308 microns.

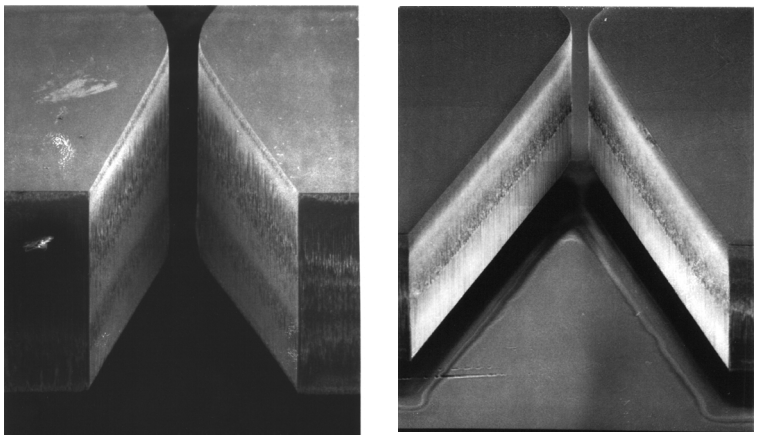


Figure 4.14: a) 7:1 nozzle with a 34 micron throat b) 15.3:1 nozzle with a 18 micron throat. Both were created using a halo mask

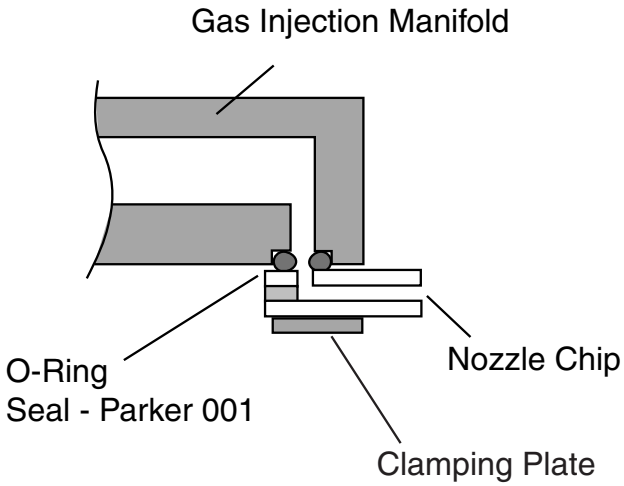


Figure 4.15: Packaging solution for the micronozzle study. Pre-drilled hole in die is clamped to an o-ring in a gas-injection manifold.

4.4 Packaging

Figure 4.15 depicts the gas injection manifold that allows the die to interface with a standard high-pressure source. The manifold consists of an o-ring gland that runs to a $\frac{1}{4}$" pipe fitting that is interconnected with stainless steel tubing. The die is clamped against the o-ring to prevent fluid leakage. A valve and regulator are present upstream to allow flow control to the nozzle. The o-ring is a standard Parker-001, of 40 durometer hardness. The 50 and 60 durometer were tried initially, but they cracked the glass at clamping pressures required for sealing.

4.4.1 Entrance Effects

In order to evaluate the microfabricated nozzle as a propulsion device, the entire on-chip system should be examined for potential losses which would threaten its performance. The fluid delivery system leading up to the chip consists of large diameter valves and lines, which at these low flow rates represent very little resistance to the flow. The concern that the gas injected into the nozzle's settling chamber through the equivalent of a 90° miter bend would have a large total pressure loss, and a large z-velocity component.

A 2-D simulation using the commercial code FLUENT is used to determine whether the pressure drop across the settling chamber or residual z-velocity prior to gas injection poses an inconsistency with assumptions on the nozzle boundary conditions employed in the CFD. Figure 4.16 illustrates the entrance flow for the $Re = 1450$ flow. As is expected, a separation region forms due to the imbalance in the inertia forces on the fluid and the centripetal forces around the bend. This separation region grows with increasing throat Reynolds number. The reattachment length is the x-distance from the bend at which the flow becomes fully axial. In this analysis, fully axial flow is defined as the z-velocity being less than 1% of the x-velocity. Figure 4.17 depicts the variation of the reattachment length with the throat Reynolds number. The normalized channel length of all nozzles tested is 8.1, which is greater than the reattachment length for all Reynolds numbers tested. Therefore, the flow can be considered axial. The reattachment length will be even shorter in the device than in the simulation due to the 3-D nature of inlet, which will allow the separation region to expand in the y-direction.

The pressure drop across the plenum is a function of the dynamic pressure. Since the mass flow is choked, the mean chamber velocity is constant over the range of throat Reynolds numbers. This value is 1.8 m/s and results in a chamber Reynolds number from 18 to 370 for throat Reynolds numbers of 175 to 3721. The FLUENT simulation predicts that the pressure drop across the plenum for the range of Reynolds numbers is negligible. It never exceeds 0.01% of the total pressure, which is of the same order as the dynamic pressure in the chamber.

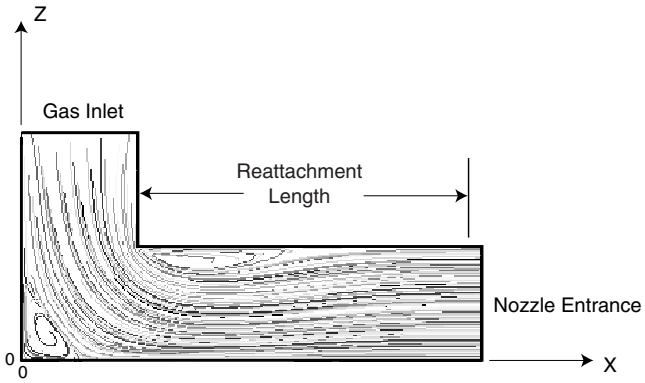


Figure 4.16: Streamlines in xz-plane of nozzle chamber at a $Re = 1450$. A 2-D simulation with $FLUENT$ is used to compute the flow field.

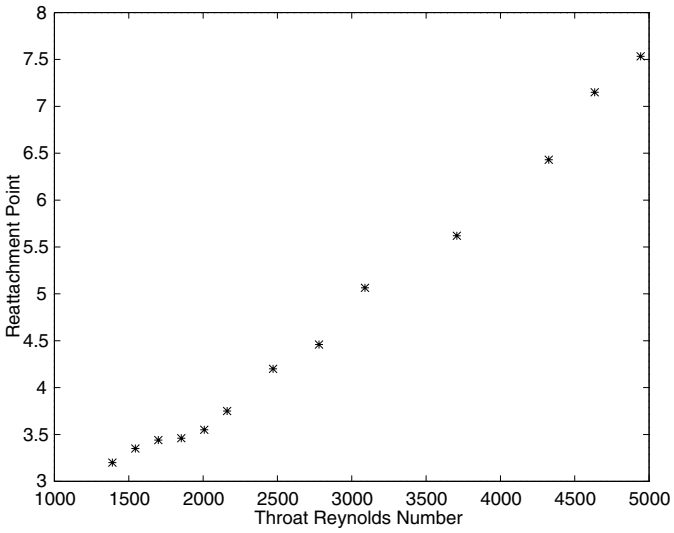


Figure 4.17: Reattachment point of separation bubble downstream of injection inlet (distance normalized by channel height). This is the point the flow returns to nearly axial.

4.5 Summary

Advancements in microfabrication have lead to the creation of high aspect ratio structures. Deep reactive ion etching was used to create 2-dimensional contoured nozzles. These nozzles have throat widths 5 times smaller than previously achieved and yet maintain anisotropy as well as permitting a smoothly varying geometry. The fabrication results are summarized as:

- Anisotropic nozzles are best fabricated with a halo mask when used in conjunction with high coil power during the etch cycle. This minimizes loading for geometries of variable feature sizes and greatly improves anisotropy.
- Wall Roughness is less than 1 micron, and feature variation is less than 1 micron from the top to the bottom of the etch.
- Nozzle feature sizes enlarge due to resist taper near feature. This may be prevented with an oxide mask.
- Nozzle etch depth for a 10 micron trench is limited to 300 microns due to poor selectivity at high coil power etches. Oxide masks can be used to reach greater etch depth, though anisotropy will suffer due to aspect ratio dependent etching.
- Two-sided anodic bonding can be accomplished as long as electrical contact can be made with silicon during the second bond.
- Gas injection into the nozzle is best accomplished through an elastomer seal to a port pre-drilled into the glass prior to anodic bonding.
- Gas injection perpendicular to the nozzle inlet has little affect on pressure drop or axial velocity, as long as the chamber is sufficiently long to allow the flow to recover.

Chapter 5

Cold-Gas Thruster Testing

This chapter presents the experimental testing of the cold-gas nozzles. This testing is grouped into mass flow tests performed at MIT and thrust tests performed at the Aerospace Corporation in Redondo Beach, California. The mass flow tests help to quantify the viscous losses. But, only by quantifying the thrust can the propellant efficiency of the nozzle be assessed.

5.1 Test Set-up

5.1.1 Mass Flow Testing

A schematic of the test stand assembled at the MIT Fluid Dynamics Research Laboratory is shown in figure 5.1. Grade 5 nitrogen is chosen as the gas source because it is relatively inert. Helium, a noble gas which is inert by nature, is not used because it is very difficult to contain due to its small atomic size, which has a tendency to leak. Grade 5 nitrogen is a purer form of dry nitrogen with less than 0.1% of trace gases. It is important to reduce the propellant to its purest component since performance is a function of the average molecular weight of the gas.

The system consists of $\frac{1}{8}$" copper tubing connecting the gas supply to the nozzle. This is connected through a series of Swagelok fittings. The flow rate is measured with a Hastings-Teledyne HFM-200 0-1000 sccms flow meter (0-10 mg/s). This flow meter is inherently linear with an accuracy of $\pm 1\%$ of the full scale reading (i.e. $\pm$10sccms). The pressure is measured at the source by an analog gauge and at the manifold location by an Omega PX303-3005V digital pressure transducer. This is accurate to 0.3% of full scale (i.e. ± 1 psi).

A 0.5 micron filter is placed in line to prevent contamination from clogging the nozzle. In addition a solenoid valve is placed in line to activate the flow for thruster testing. Since the flow rate in the system is so low, the pressure drop is measured at less than 1 psi from supply to nozzle manifold at the maximum operating flow rate of 1000 sccm.

Initial tests were performed exhausting to atmospheric pressure. Subsequently, a 5 liter bell jar was modified by placing a feed line through the bulkhead, and allowing the manifold to be mounted inside the jar. This jar was then pumped down to 0.5 torr and maintained below 30 torr over the full range of operation. The flow, for most test cases, is in the underexpanded regime and choked for all cases.

5.1.2 Thrust Stand

The thrust stand located at the Aerospace Corporation is identical in functionality to the mass flow stand, except that the bell jar is replaced by a vacuum chamber and the nozzle is mounted to a thrust plate.

The vacuum chamber is approximately 10000 liters and is evacuated by roughing pumps and a mechanical blower. This configuration can achieve a back pressure between 100-200 mTorr. Outside of the chamber, pressure and mass flow transducers are in the supply line. Thus, all of the data that was taken at MIT, can

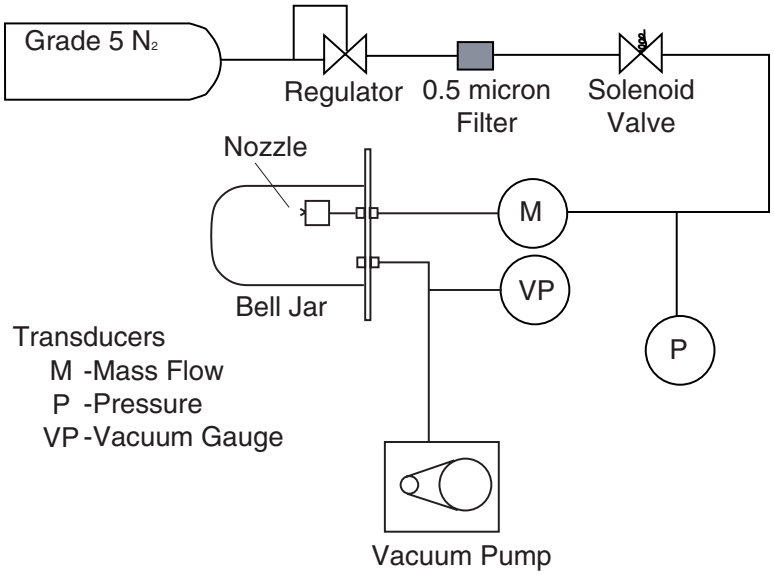


Figure 5.1: Mass Flow Test Assembly. The gas flow through the nozzle is discharged into a bell jar evacuated to 1 torr. This allows mass flow through the overexpanded nozzle to be measured.

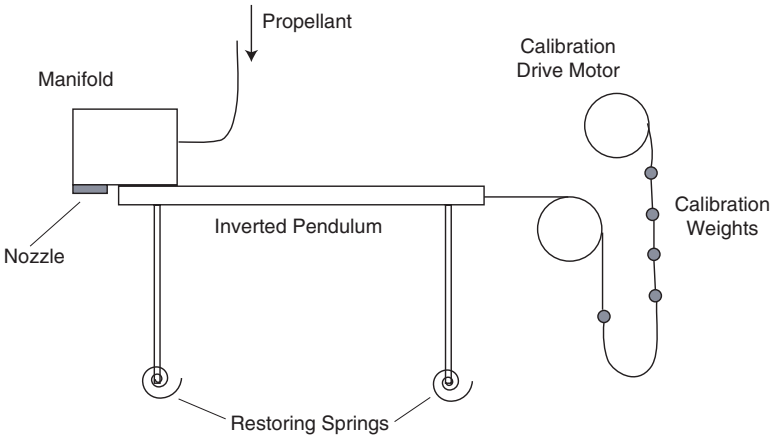


Figure 5.2: Thrust Stand Schematic. The thrust against a restoring spring allows the displacement to be measured, and the subsequent thrust to be evaluated. Weights of known mass hanging from the pulley provide a calibration thrust to the stand.

also be duplicated at the Aerospace Corporation. The MKS 1000 sccm mass flow meter with an accuracy of 1% of full scale is in-line.

The gas injection manifold with attached nozzle is mounted to the thrust plate atop the inverted pendulum, as in figure 5.2. The plate thrusts against restoring springs, and displacement is measured to determine the thrust level. The accuracy of this system is ± 0.2 mN and ± 0.2 psia [14] for thrust and pressure respectively. Unfortunately, there is no damping on the thrust plate, thus, the accuracy diminishes to 1 mN due to the oscillations and the inability to maintain a constant reading.

The stand is calibrated by applying a known force to the plate and measuring the voltage output. The force is applied by hanging known weights from a wire attached to the plate. The wire runs over a low friction pulley to redirect the gravity force vector from vertical to parallel to the plate. By performing the calibration, the thrust measurement can be corrected for the resistance of the propellant supply line. The smallest calibration weight is equivalent to 9.21 mN of thrust. This is the only data point used to resolve the thrust-voltage output curve around the testing point. Any nonlinearities that might exist at low thrusts will be smoothed over in the calibration. However, for additional weights used in calibration, the thrust stand remains linear over a large range of thrusts and should be reliable at the low thrusts, also.

5.2 Uncertainty Analysis

As explained in section 5.1, there is an inherent uncertainty in the measurements. This is quantified for each measured quantity (pressure, flow rate, thrust). In addition, there will be uncertainty introduced by the way the measurements are taken. For example, the ambient is measured to be at 22.5° C, and this is assumed to be the same as the supply temperature. In addition, pressure is measured upstream of the settling chamber, but at these low flow rates, the pressure drops are very low (less than 1 psi). Both of these measurements entail some uncertainty since they are not located directly in the settling chamber.

The uncertainties in the feature geometry, while having no affect on the accuracy of the thrust and mass flow, directly affect the derived quantities of thrust, mass flow, and I_{sp} efficiency. Without exact knowledge of the throat area, the ideal mass flow rate cannot be predicted. This leads to uncertainty in the C_d. Since Reynolds number is also a function of throat width, there will be uncertainty when assessing performance as a function of this parameter.

The throat is a critical point in the geometry, because it sets the mass flow rate in the choked condition. The nozzle is measured using an optical microscope with a calibrated stage. By translating the stage in x and y, a distance between two points can be determined. This method was used to measure the throat, exit, and nozzle length to an accuracy of 0.5 microns. This is a 2.5% uncertainty in the throat width for the smallest nozzles tested. In addition, it sets the area ratio of the nozzle. Such a variation can result in an uncertainty of ± 1 in the expansion ratio for the smallest features. The depth of the nozzles is measured by the observation of fringes from an interferometer on the focal plane. These fringes are focused on the top surface, and, through a calibrated adjustment of the focal plane they are focused on the bottom surface. This is accurate to ± 1 micron, although the uncertainty is a smaller percentage of the feature size ($\sim 0.3\%$).

The largest source of uncertainty is in the thrust measurement, and at low Reynolds numbers, the uncertainty is larger than the value being measured. In addition, this uncertainty also occurs in the calibration of the thrust stand, which could alter the nature of the efficiency variation with Reynolds number. Variation of the slope of the calibration curve can displace the efficiency versus Reynolds number curve horizontally, whereas the uncertainty in the measurement shifts the curve vertically.

To summarize, the uncertainty bars presented with the experimental data come from three sources: uncertainty in the geometry, the inherent uncertainty of the instruments, and the assumptions of the measurements which increase the uncertainty. The latter of the three is the most difficult to quantify, but, with the reasonable assumptions made, it is shown to be small.

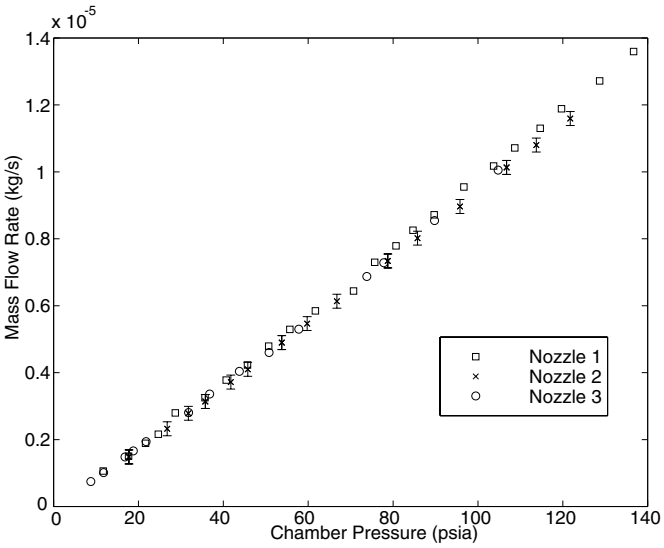


Figure 5.3: Mass flow measurements for 3 nozzles of 5.4:1 area ratio and 20 micron throats. Variation of results is minimal because the geometric variation in manufactured nozzles is minimal.

5.3 Test Results

5.3.1 Mass Flow Test Results

Tests were run on all of the nozzle geometries to assess their performance. Figure 5.3 compares 3 nozzles with throat widths of 21 ± 1 microns, and area ratios of 5.4:1. This figure shows the similarity in the nozzle responses to chamber pressure. This is due to the small variations in nozzle geometry from sample to sample. Though this plot appears linear, the mass flow is decreasing faster than linear at low pressures due to low Reynolds number viscous effects. Figure 5.4 depicts the percent deviation of flow rate from the linear slope connecting the highest flow rate tested and the origin. This is as high as 12% at the lowest pressure tested. This curve has a maximum since it must also pass through zero flow rate at zero applied pressure. This maximum cannot be resolved due to limitations on regulated pressure.

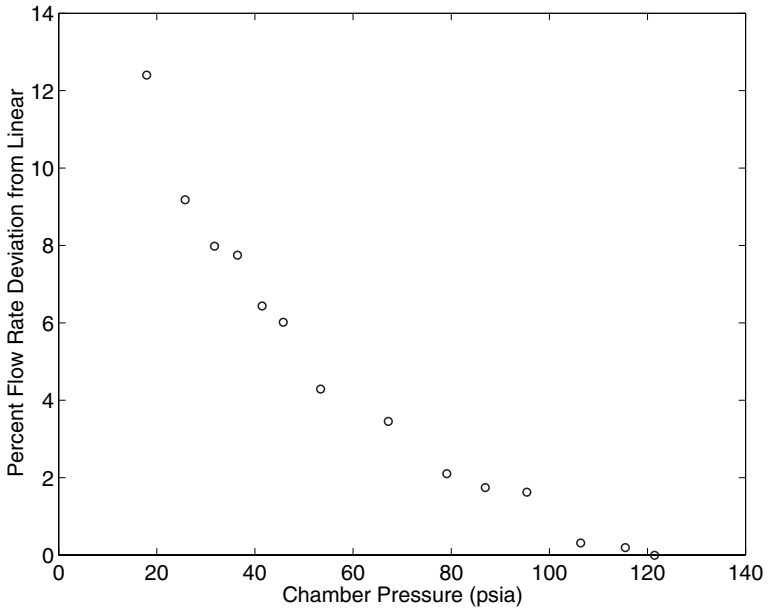


Figure 5.4: Deviation of mass flow in figure 5.3 from linear slope connecting origin and the maximum flow rate data point. The deviation is due to increased viscous effects at low Reynolds numbers.

5.3.2 Thrust Test Results

The mass flow test procedures were repeated at the Aerospace Corporation, with the manifold mounted on the thrust stand in the vacuum chamber. Thus, thrust was measured in addition to mass flow, chamber pressure, and ambient pressure. The experiments were run on three conical nozzle geometries of 7.1:1, 15.3:1 and 16.9:1 expansion ratios. Their respective throat widths were 34, 18, and 37.5 microns. The mass flow results are shown in figure 5.5. Whereas the previous section highlighted the repeatability of nozzle performance for similar nozzles within the same batch, figure 5.5 compares different nozzle geometries. As expected, the mass flow decreases with throat size for a given pressure. The thrust results as a function of chamber pressure are shown in figure 5.6. Just as with the mass flow, the nozzle thrust for a given pressure decreases with throat area. This is due to thrust being a product of the mass flow and exit velocity.

The I_{sp} is a derived quantity indicating the thrust per unit weight flow rate and defined in equation (3.34). Of interest to propulsion system designers is the variation of I_{sp} with thrust, as depicted in figure 5.7. This clearly indicates which is the best nozzle for each thrust level. The largest geometric expansion ratio nozzle has the highest performance above 2 mN. At 2 mN, there is a crossover with the 15.3:1 nozzle, which has a smaller throat diameter, but a larger exit-plane aspect ratio. As will be demonstrated shortly, this causes the exit velocity to decrease rapidly at low Reynolds numbers, due to the low effective area ratio.

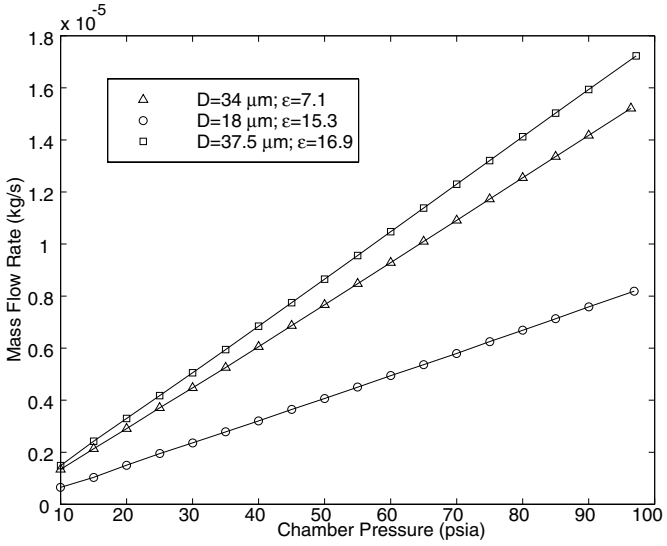


Figure 5.5: Mass Flow variation with chamber pressure for 3 nozzle geometries. Larger throats result in larger flow rates for the same pressure.

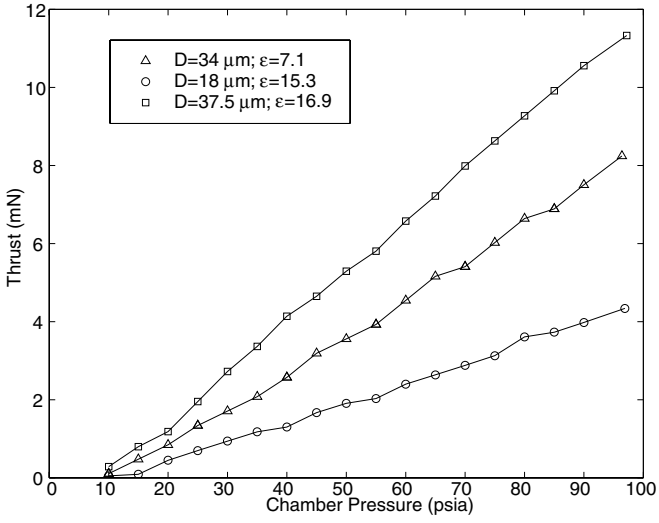


Figure 5.6: Thrust variation with chamber pressure for 3 nozzle geometries. Larger exit areas result in larger thrusts.

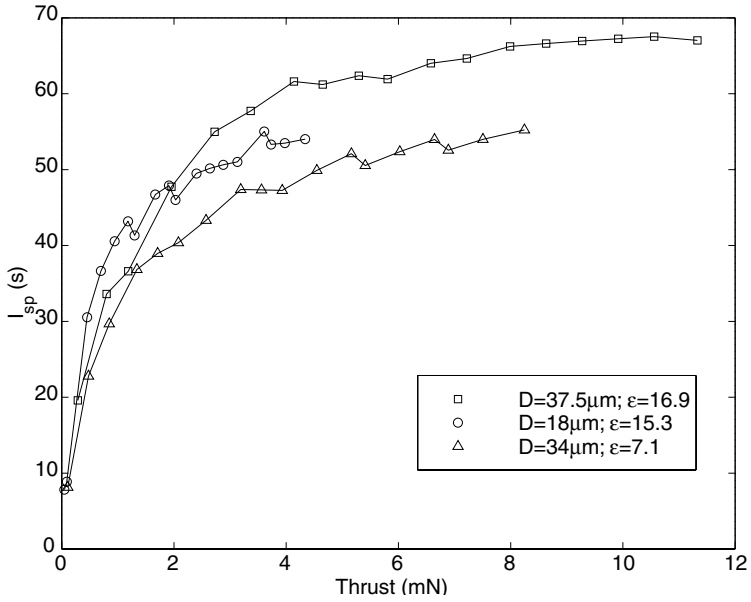


Figure 5.7: I_{sp} variation with thrust. The dramatic reduction at low thrusts is due to the increase in viscous effects present at low chamber pressures, and hence low thrusts.

5.4 Comparisons between Numerical and Experimental Data

The two-dimensional computational analysis described in Chapter 3 serves as the benchmark to which the experiments are compared. Figures 5.8-5.10 depict the coefficient of discharge variation with Reynolds number for each of the three nozzle geometries tested. Within the uncertainty of the measurement, the C_d agrees with the numerical computation for all but the lowest Reynolds numbers, for all cases. The deviation of the results from the numerical solution increases with decreasing Reynolds number. There are several factors influencing the deviation. Endwall boundary layers, which are neglected in the 2-D simulation, account for a portion of the loss. At the lowest Reynolds numbers, the endwall boundary layers account for additional 10% loss in throat area. Nonequilibrium effects may also account for a portion of the deviation.

The thrust and I_{sp} efficiencies decrease with Reynolds number in a much more dramatic fashion than the C_d. Figures 5.11 - 5.13 depict this for thrust efficiency, and Figures 5.14 - 5.16 for the I_{sp} efficiency. This is much larger than the C_d because the mass flow rate is set at the throat, by the choked condition, and thrust and I_{sp} are determined at the exit-plane. Since the flow is continuing to evolve downstream of the throat, viscous losses continue to decrease the performance much beyond that seen in the C_d. It should stand out, however, that the decrease is far beyond that predicted by the two-dimensional model. Once again the endwall boundary layers further reduce the effective area ratio of the nozzle, and subsequently the exit velocity achieved by the flow. This, in turn, reduces the thrust and I_{sp}. The losses are quantified in the next section.

The uncertainty bars in the figures are based on the analysis presented in section 5.2. All of the uncertainties are quantified as a fixed number. They all grow relative to the measured value as Reynolds number decreases, because the fixed uncertainty becomes a larger fraction of the measurement as the measurement decreases. In the case of the thrust measurements, the uncertainty for the smallest thrust is the size of the measurements. The uncertainty is a smaller fraction of the total thrust for larger nozzles.

Figure 5.17 depicts the variation of thrust coefficient with Reynolds number for the 16.9:1 nozzle. This is a good performance indicator because it illustrates the effectiveness of the nozzle. At the highest Reynolds number cases, the nozzle performance is as expected. Once the C_f drops below 1.2, it has a lower performance than that of isentropic flow through a choked orifice of the same scale as the throat. This highlights the useful regime of the nozzle, and the substantial viscous effects.

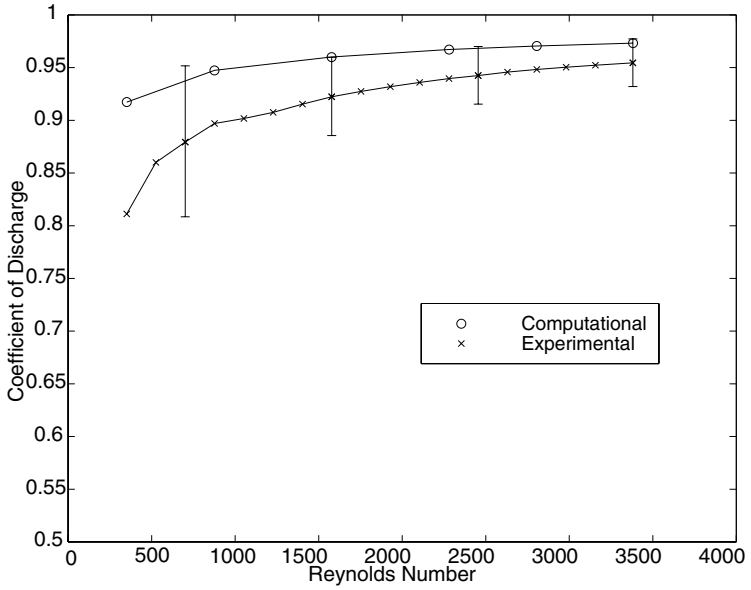


Figure 5.8: Mass flow efficiency (C_d) variation with Reynolds number for the 7.1:1 geometry. The C_d compares well with the experiment for all but the lowest Reynolds numbers. This is where the boundary layers are the thickest.

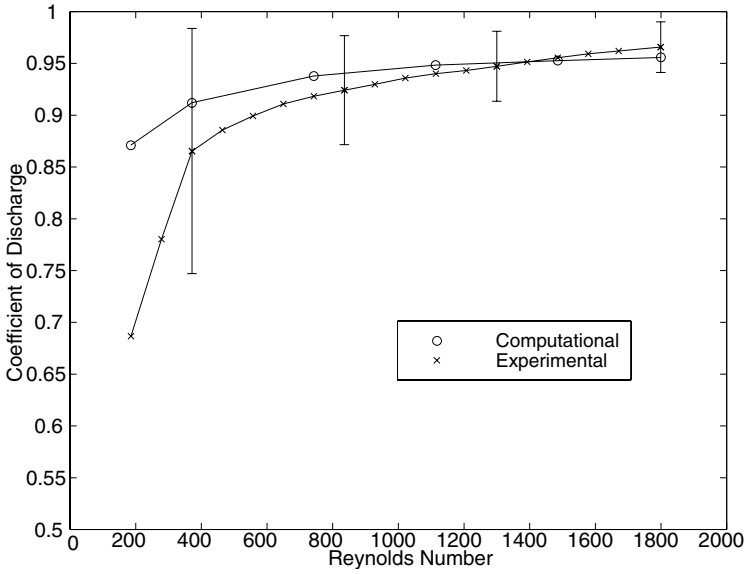


Figure 5.9: Mass flow efficiency (C_d) variation with Reynolds number for the 15.3:1 geometry

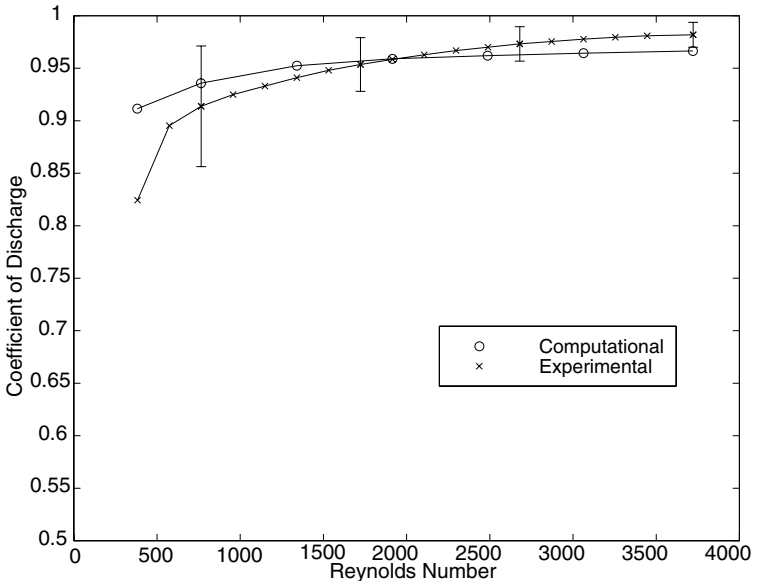


Figure 5.10: Mass flow efficiency (C_d) variation with Reynolds number for the 16.9:1 geometry

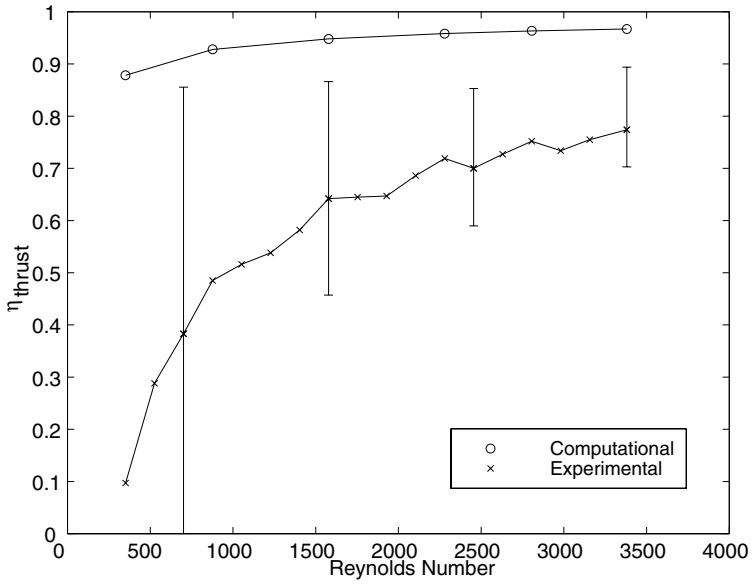


Figure 5.11: Thrust efficiency variation with Reynolds Number for the 7.1:1 nozzle geometry

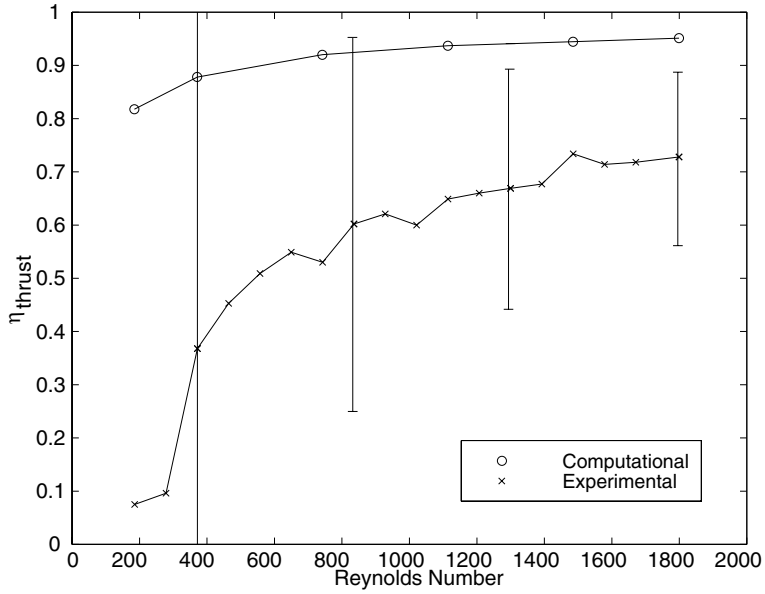


Figure 5.12: Thrust efficiency variation with Reynolds Number for the 15.3:1 nozzle geometry

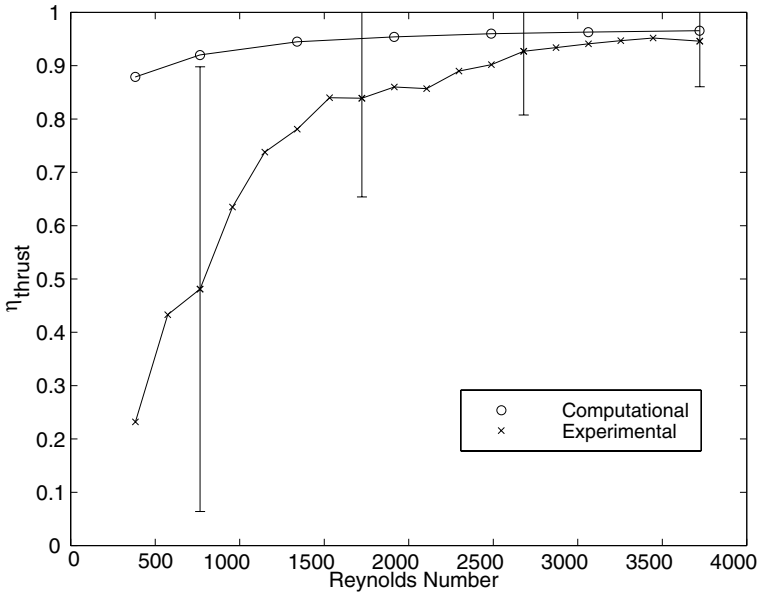


Figure 5.13: Thrust efficiency variation with Reynolds Number for the 16.9:1 nozzle geometry

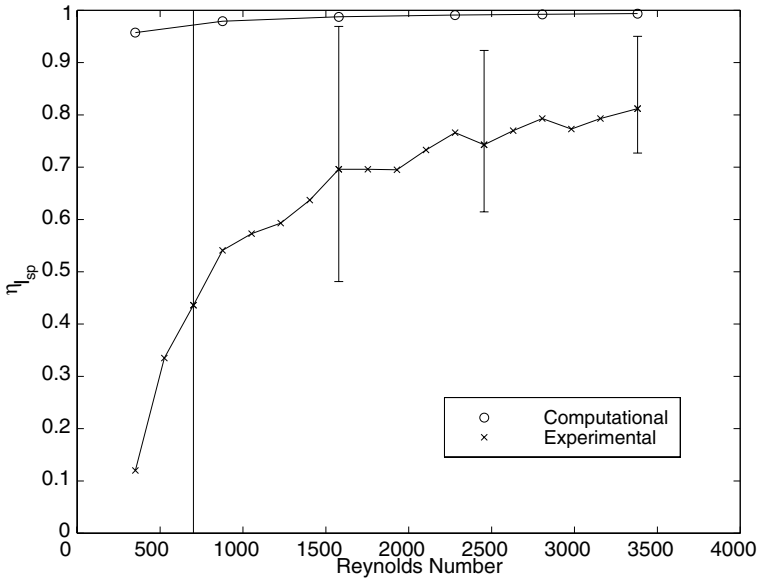


Figure 5.14: I_{sp} efficiency variation with Reynolds Number for the 7.1:1 nozzle geometry

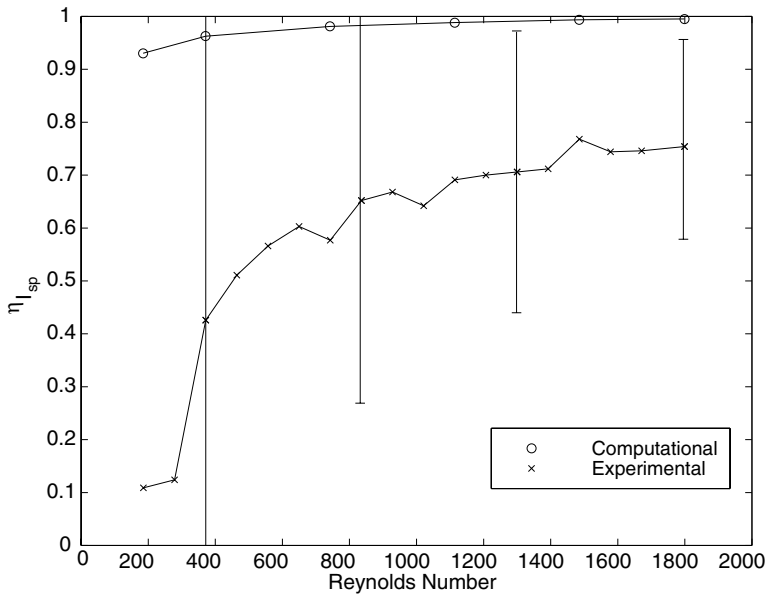


Figure 5.15: I_{sp} efficiency variation with Reynolds Number for the 15.3:1 nozzle geometry

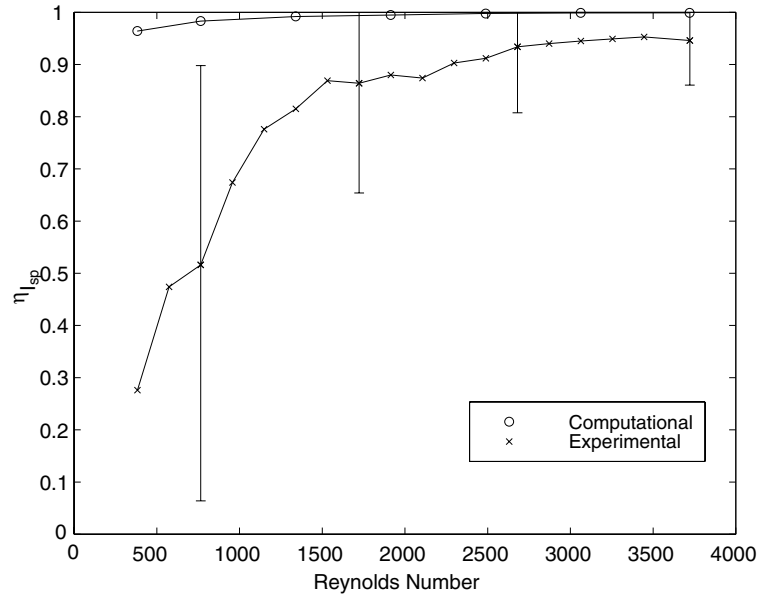


Figure 5.16: I_{sp} efficiency variation with Reynolds Number for the 16.9:1 nozzle geometry

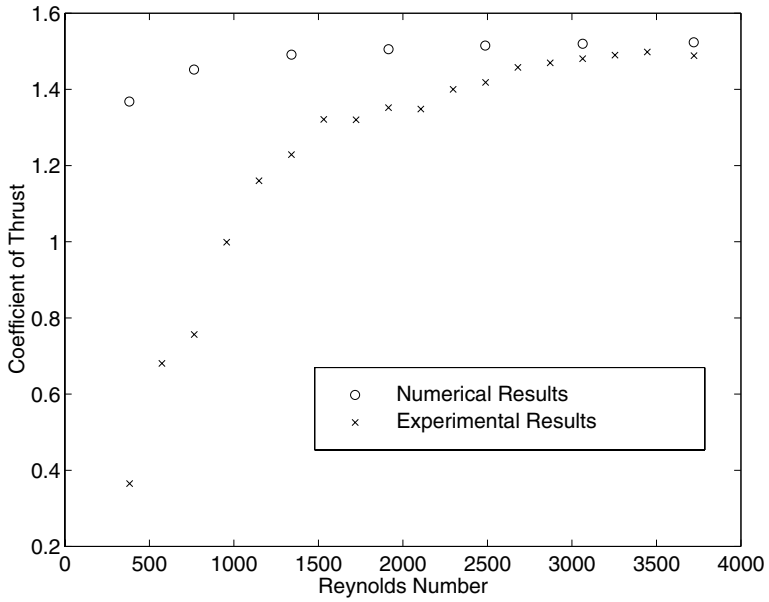


Figure 5.17: Comparison of the numerical results and the experimental results of the coefficient of thrust as it varies with Reynolds number for the 16.9:1 nozzle

5.5 Experimental Deviation from the Numerical Calculations

This section examines the impact of several sources of performance loss, which could account for the deviation of the test data from the predictions of the numerical analysis. The uncertainty bars indicated one source of the discrepancy and was explained in a previous section. This section looks at the flow field, and the effect of unmodeled flow characteristics.

5.5.1 Endwall Boundary Layer Blockage

Section 3.7.1 concluded that a majority of the performance variation with Reynolds number is associated with the blockage created by the boundary layers. Additional performance losses will be incurred due to boundary layer interactions at the corners, as well as a breakdown in the thin boundary layer assumption which allows the outflow state variables to be extrapolated from the nozzle interior. To first order, the correction for blockage should provide a guide for optimum nozzle operation. However, if the computational analysis reported in Chapter 3 is to be used to estimate the nozzle performance, the blockage generated in the full geometry must be considered. The best solution would be to pursue a three-dimensional computation. However, the 2-D simulation of the flow in the largest of these nozzles takes 4 days to run on a Silicon Graphics workstation, and an additional dimension of 50 grid points would take 50 times as long for each operating condition. With the computational resources at hand, this is unacceptable. The 2-D simulations, however, provide insight into the boundary layer thickness at a given cross-section.

Just as the geometry was altered in section 3.7.1 to produce a 2-D effective area ratio, the full 3-D geometry can be approximated by subtracting the displacement thickness from all four walls that make up the nozzle cross-section. Since the boundary layer that is developing on the endwalls is driven by the same core velocity as the sidewalls, the displacement thickness should be the same as that computed in the 2-D simulation. In reality, the endwall boundary layers will be thinner due to the flow channel diverging, but this will be counteracted by a thickening that will occur due to the slower flow found in a reduced area-ratio nozzle. Thus, this will serve as a first approximation to the performance loss in the full 3-D nozzle, as long as the core flow is unaltered by the displacement thickness of the endwall, which is true for large exit aspect ratios (high h_o/D).

The variation of the effective area ratio with Reynolds number for nozzles of different width-to-depth aspect ratios are shown in figures 5.18-5.19. The exit plane of the nozzle and the effective exit area at the lowest Reynolds number are shown as insets in these figures. Both nozzles are 308 microns deep. Since the 15.3:1 nozzle has an 18 micron throat and only a 275 micron exit, it has a larger exit plane aspect ratio for a similar expansion ratio. For the nozzles tested, the aspect ratios are 1.14 for the 15.3:1 nozzle and 0.48 for the 16.9:1 nozzle.

As the Reynolds number decreases, the displacement thickness increases, and it reduces the effective area ratio at a given cross-sectional location (defined as the geometric or actual area ratio). For the 16.9:1 nozzle, the exit aspect ratio is small enough to cause the effective area ratio to reach a maximum within the nozzle. For these operating conditions, the additional length of nozzle drives performance down because of the increase in viscous losses from additional boundary layer growth. These plots seem to indicate that a dramatic performance drop should accompany these large adjustments in effective area ratio. However, the performance variation with expansion ratio is minimal. Figure 5.20 depicts the variation of exit Mach number with expansion ratio using a quasi-1-D analysis. Since the exit velocity is a function of the exit speed of sound, the chamber temperature sets the maximum exit velocity that can be achieved for a nozzle of infinite expansion ratio. This is 790 m/s for the 300 K chamber temperature present in these runs. The corresponding exit velocity is depicted on the right side of figure 5.20. Therefore, increasing the expansion ratio from 4 to 16 only contributes 10% to the I_{sp}. As stated in section 3.7.1, the primary factor in performance reduction is the boundary layer thickness at the throat, which drives down mass flow efficiency and ultimately the thrust efficiency, and not the reduction in effective area ratio.

The flow variables can be computed from a quasi-1-D analysis of the effective area ratio depicted in figures 5.18-5.19. These flow variables can be used to compute thrust and I_{sp} for the quasi-3-D geometry. The corrected thrust and I_{sp} efficiencies for the 16.9:1 nozzle are shown in figures 5.21- 5.22, respectively. Though this rudimentary model does not match the experimental data exactly, it does indicate that the

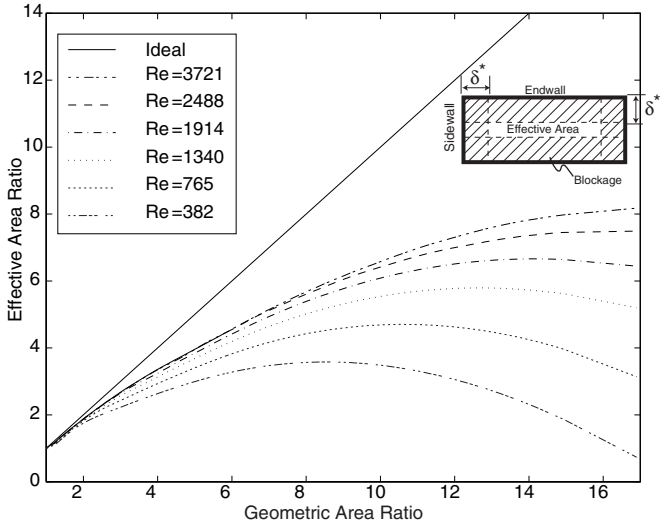


Figure 5.18: Variation of the quasi-3-D effective area ratio with actual geometric area ratio. This is for a nozzle with an exit height to width aspect ratio of 0.48.

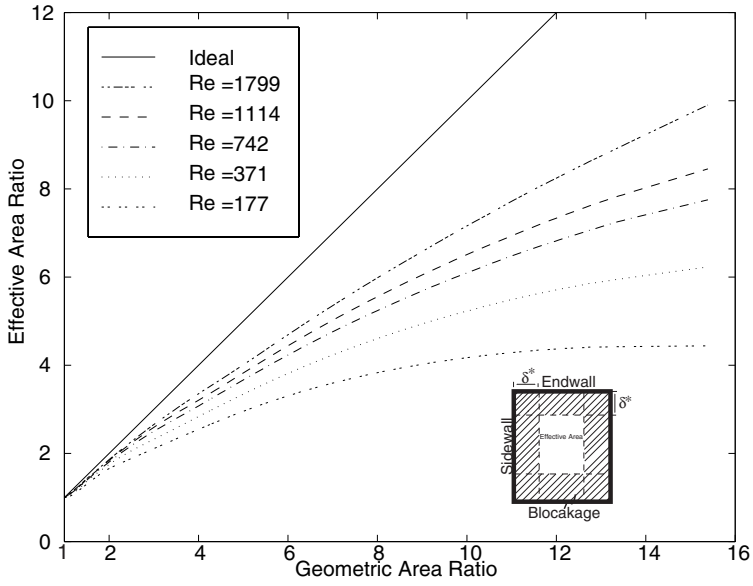


Figure 5.19: Variation of the quasi-3-D effective area ratio with the actual geometric area ratio. This is for a nozzle with an exit height to width aspect ratio of 1.14.

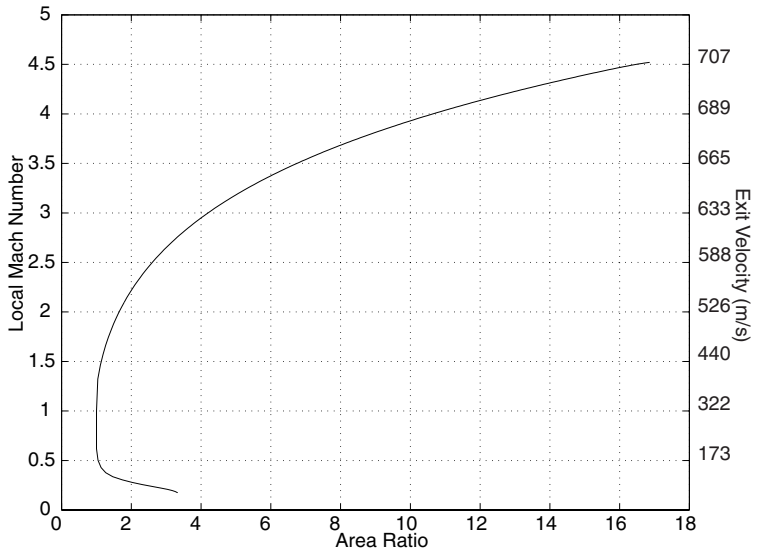


Figure 5.20: Variation of exit Mach number with area ratio. If operated at 300K chamber temperature, the exit velocity on the right applies

endwall blockage accounts for a large portion of the performance loss. Furthermore, it is indicative of the rapid drop in the performance at Reynolds numbers below 1000.

Figure 5.23 depicts the endwall corrections in thrust efficiency for the 15.3:1 nozzle. Since the exit plane aspect ratio is nearly double, the endwall corrections are minor. The same trend is noted in the 16.9:1 case, but there is a dramatic reduction in measured thrust performance for each operating condition that is not explained by the endwall correction. This quasi-1-D analysis suggests that the 15.3:1 nozzle should out-perform the 16.9:1 nozzle because it has double the aspect ratio in the exit plane. This should result in a reduction of endwall effects, and an increase in I_{sp} over that delivered by the 16.9:1 nozzle. And, it suggests the performance should not deviate much from the 2-D simulation. Test results in figure 5.7 do show this higher performance, but only at the lowest Reynolds numbers. At high Reynolds number, the 16.9:1 has the highest performance. Once again, this trend is difficult to discern due to the uncertainty of the thrust measurements.

Additional thrust and mass flow data were taken for the portion of this research detailed in Chapter 6. This data was taken on a thrust stand with improved resolution and damping. The Reynolds number in this case was lowered by increasing the chamber temperature (lowering chamber density). A preview of this data is shown in figure 5.24. This improved data does resolve that the variation of thrust efficiency with Reynolds number is much less than shown in the previous figures, but the corrected numerical results are still different from those experimentally measured. In order for the deviation to be an endwall effect, the boundary layers would have to be twice as thick as predicted by the simulation. This is not the case, since they should at best be the same size as the sidewalls.

Though the endwalls can incur blockage that reduces the performance by 40% in the lowest Reynolds number cases, in most cases it accounts for $3 - 10\%$ variation in performance due to a reduction in effective area ratio. Thus, there are additional effects not modeled that are reducing the performance.

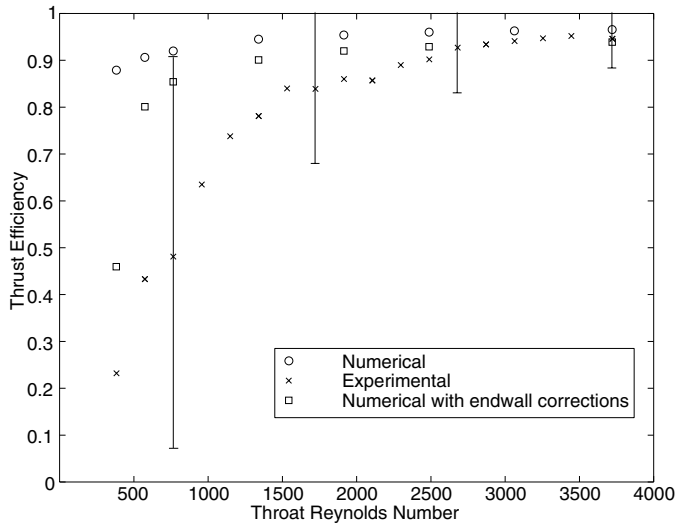


Figure 5.21: Thrust efficiency variation with Reynolds Number for the 16.9:1 nozzle geometry. This is for a nozzle with an exit height to width aspect ratio of 0.48.

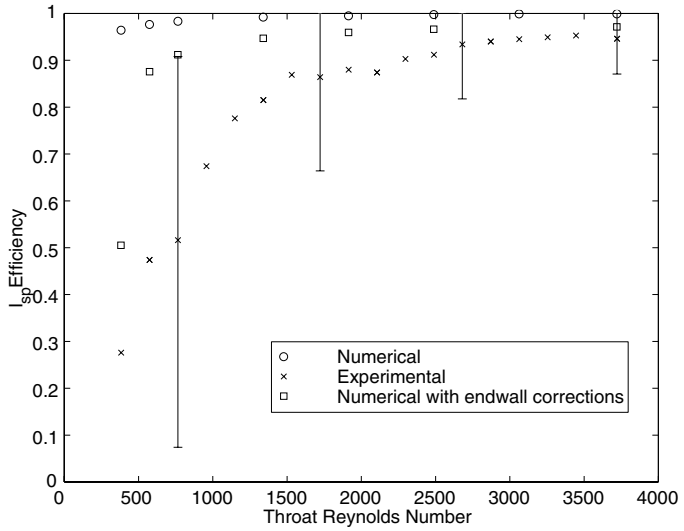


Figure 5.22: I_{sp} efficiency variation with Reynolds Number for the 16.9:1 nozzle geometry. This is for a nozzle with an exit height to width aspect ratio of 0.48.

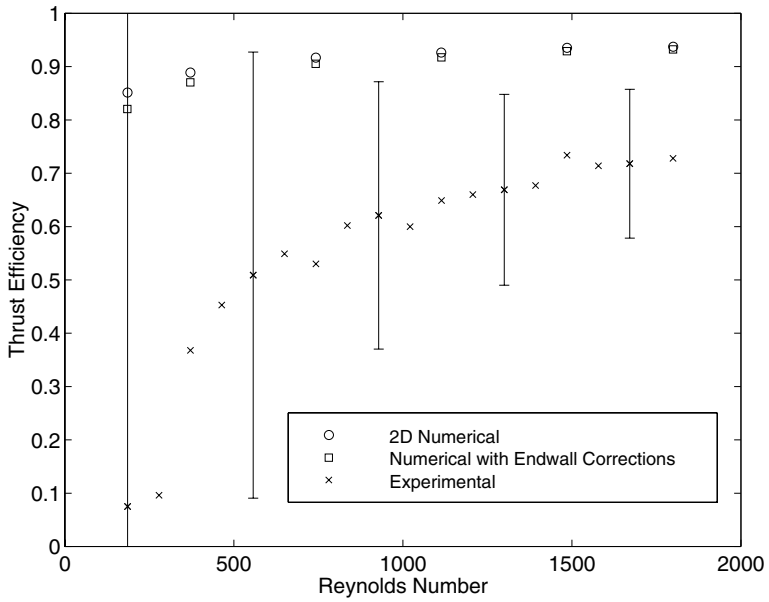


Figure 5.23: Thrust efficiency variation with Reynolds Number for the 15.3:1 nozzle geometry. This is for a nozzle with an exit height to width aspect ratio of 1.14.

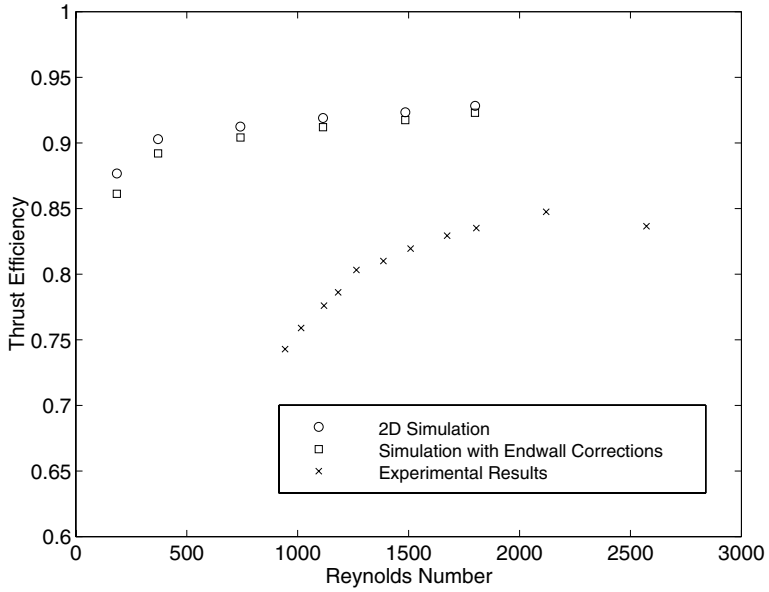


Figure 5.24: Thrust efficiency variation with Reynolds Number for the 8.26:1 nozzle geometry. This is compared to a 2-D CFD simulation with endwall corrections. Maximum overprediction is 13% This is for a nozzle with an exit height to width aspect ratio of 1.03.

5.5.2 Nonequilibrium Effects

The energy in the chamber is a function of the temperature, which characterizes the intermolecular collisions of the gas. As the gas expands, the thermal energy of the molecules is converted into directed translational energy (kinetic energy). The time it takes for gas to reach local equilibrium is referred to as the relaxation time. The relaxation time is a function of the intermolecular collisions and can be computed as

$$t = \frac{1}{\nu} = N A_c \bar{c} \quad , \tag{5.1}$$

where ν is the collision frequency, N is the number density of the gas, A_c is the collisional cross-sectional area, and $\bar{c}$ is the mean molecular speed that can be computed as $\sqrt{3RT}$. In order for the gas to be considered in equilibrium, the residence time in the flow chamber must be much greater than the relaxation time. As seen in equation (5.1), the relaxation is a function of the number density; therefore, this effect increases with decreasing Reynolds number. Thus, as the gas is expanding in the nozzle, the relaxation time is increasing.

At standard conditions, the relaxation time is on the order of 1 nanosecond. The flow time from the throat to the exit of the 16.9:1 nozzle is 2 microseconds. Therefore, the flow is well within equilibrium at this condition. However, as the pressure decreases in the nozzle, the relaxation time increases; meanwhile, the residence time also decreases due to the increased velocity, and shorter distance left to travel. In the boundary layer, the temperature is higher than the core due to the thermal gradient created by the relatively warm wall. This, also, increases the relaxation time. Once a fluid particle reaches the location of the 3:1 area ratio in the supersonic portion of the nozzle, the relaxation time is 1% of the residence time in the core and less in the boundary layer due to the slower velocity. This will represent a loss in the conversion of thermal energy into kinetic energy. In order to gauge the magnitude of this loss, an equilibrium gas model must be introduced into the CFD to resolve the true velocity. But, to first order, the reduction in exit energy could be considered to be 1% of the gain achieved past the 3:1 area ratio.

A more difficult timescale to assess is the rotational relaxation rate. Whereas translational relaxation can be accurately determined from the hard sphere model as used in equation (5.1), the rotational collision cross-section is dependent upon angle and energy of collision. A model presented by Belikov and described by Gochberg [42] can be used to determine the relaxation time of nitrogen with temperature, and is valid over the range of 6-320 K. A plot of the ratio of rotational relaxation time to residence time at the location of the 3:1 area ratio is shown in figure 5.25. This is the ratio at the wall, and decreases toward the centerline due to the thermal gradient in the boundary layer. For the operating pressures investigated, this model reveals that the rotational relaxation is three times that of the translation. This is still only 3% of the residence time of the fluid particles at the highest Reynolds numbers, but increases to 9% in the boundary layer at the lowest Reynolds number cases. If a diatomic gas flow is rotationally frozen, it will behave as a monatomic gas due to the loss of two degrees of freedom. A rotationally frozen flow would have 15% less kinetic energy because it is bound in the rotational energy. Therefore, a 10% nonequilibrium would result in a 1.5% deviation from the predicted efficiency. Thus, it appears that nonequilibrium effects, while significant in some regions of the boundary layer at low Reynolds numbers, are not the sole source of discrepancy between the simulation and experiment.

Though this calculation is not presented here, a DSMC calculation by Zelesnik et al. [12] showed that the I_{sp} efficiency for an axisymmetric conical nozzle operating at a Reynolds number 117 was 79%. (The thrust coefficient in this simulation was 1.10.) This was for an identical nozzle to that tested by Grisnik et al. [10], and the DSMC numerical result was very close to their experimental results. Since the DSMC calculation looks at discrete particles, and the model transfers energy between internal and translational energy modes through the Larson-Borgnakke model, this method captures the intermolecular reduction in momentum leaving the nozzle that the current CFD does not.

5.5.3 Roughness Induced Transition

As discussed in section 3.7.2, the flow should not transition due to Tolmein-Schlichting instabilities due to the low operating Reynolds numbers. However, that does not preclude roughness-induced transition to turbulence. The effect on performance would be very difficult to quantify without a simulation that includes a turbulence model.

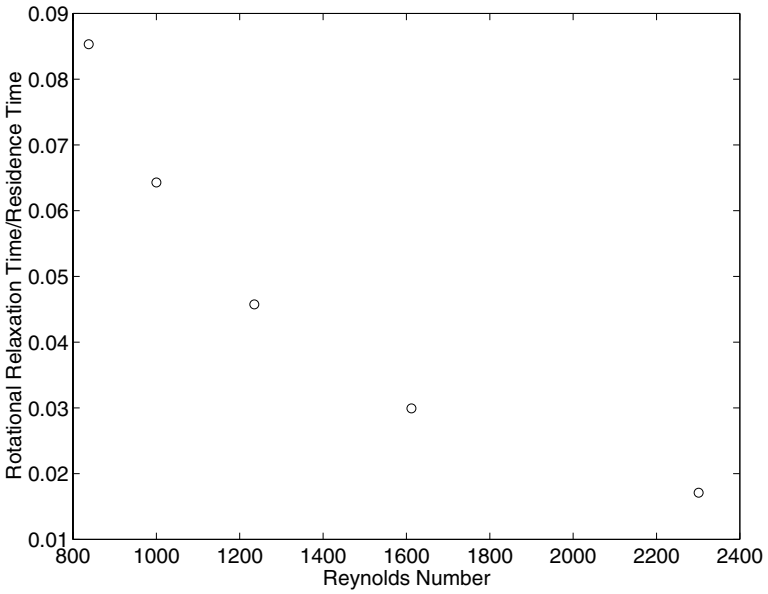


Figure 5.25: Ratio of the rotational relaxation time to the residence time at the wall of the nozzle. This computation was performed at the 3:1 area ratio in the supersonic portion of the nozzle for various Reynolds numbers based on the Belikov model described in Gochber [42]

The dissipation in a turbulent flow can increase 2 to 10,000 times that of a laminar flow. This implies a large deviation from the numerical results from the 2-D laminar flow model. However, only the highest Reynolds numbers tested would be affected by this transition, and these are, once again, the very cases that are in agreement with the model prediction. Therefore, it seems unlikely that transition is occurring in the nozzle flow.

5.5.4 Effect of Simulation Outlet Boundary Conditions

Ivanov et al. [13] performed finite volume and DSMC computations on geometries similar to silicon etched nozzles. Though their DSMC calculations produced results similar to the N-S simulation presented in this work, they claim there is a significant over prediction when extrapolated outflow conditions are used. In their studies, the extrapolated outflow conditions resulted in less of an expansion than the DSMC solution with a portion of the plume modeled. This leads to a much lower pressure along the wall, and less thrust. They show the difference in efficiency as 3% at Reynolds numbers over 1000, and 8% at 120. Zelesnik et al. also modeled a portion of the plume in their DSMC calculation, and noted a similar loss in efficiency. This would account for the improved accuracy of their calculation.

5.6 Performance Comparison with Other Experiments and Models

The following table summarizes the results of other Cold-Gas thrusters of similar thrust levels. The nozzles tested by Grisnik [10] were conventionally machined axisymmetric nozzles of various geometries (bell, conical, and trumpet), with the smallest throat diameter of 640 microns. Janson [14] implemented an anisotropically etched nozzle, as describe in Chapter 4, with a 250 $\mu m \times 250\ \mu m$ square throat. A side-by-side comparison is presented in table 5.6. The KOH nozzles are operating at a higher Reynolds number, but the lower performance suggests flow losses beyond Reynolds number effects (such as separation).

I_{sp} efficiencies of the different computational analyses described in Chapter 1 are compared in figure 5.26. The similarity in the results between this work and Grisnik et al. lends credence to the measurements

Table 5.1: Comparison of the performance of DRIE nozzles with conventionally machined nozzles, and KOH etched nozzles

	Chamber Pressure (psia)	Reynolds Number	Thrust (mN)	I_{sp} (seconds)
DRIE Nozzles	85	3492	9.91	66.63
Conventional Nozzles	3.32	3000	9.65	61.96
KOH Nozzles	25	3258	9	< 56

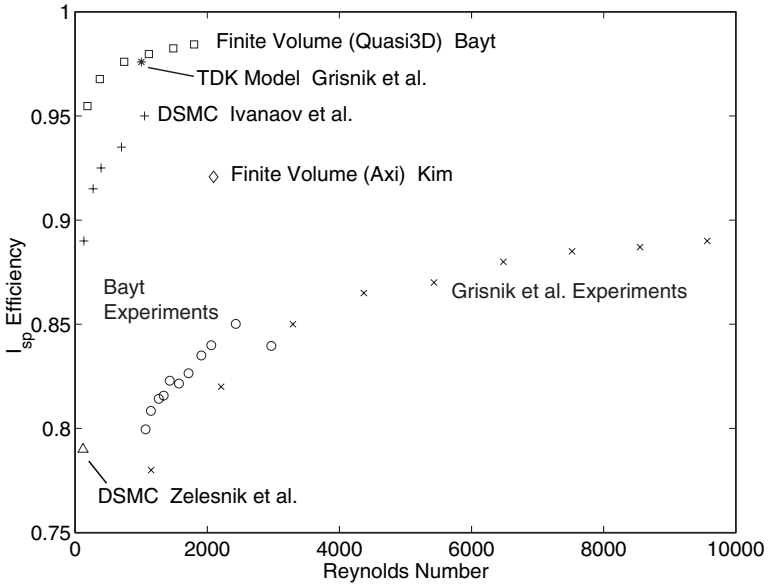


Figure 5.26: I_{sp} efficiency variation with Reynolds Number for models and experiments of other investigators.

made here. However, in no other case except the Zelesnik DSMC calculation did the experiment match the simulation to within the uncertainty of the measurement.

A list of possible deviations due to unmodeled effects has been presented in section 5.5. However, it is likely that there is some element of the physics missing from the analysis in Chapter 3 that is preventing a better correlation with the experimental data.

5.7 Summary

The experimental test results indicate that the DRIE fabricated nozzle performs better than other cold-gas nozzles in the 2-12 mN thrust range. Nozzle performance characteristics can be summarized as follows:

- The thrust and I_{sp} efficiencies for the largest nozzle are 90%, or within 2% of that predicted by numerical simulations at large Reynolds numbers.
- As the Reynolds number decreases, the performance deviates from that of the numerical analysis.
- There are several reasons the experimental results deviate from the model predictions:

  a) Uncertainty in the measurements at low thrusts is of the order of the magnitude of the measurementitself .

  b) Blockage that occurs due to the boundary layer formation on the endwalls is not modeled.

  c) Translational and Rotational Relaxation times beyond a 3:1 area ratio are a measurable fraction of the residence time.

  d) Flow divergence is increased substantially by the additional expansion present at the nozzle exit, and is not modeled with extrapolated boundary conditions.
- The endwall blockage can be accounted for by correcting the effective area ratio with a similar displacement thickness as computed for the 2-D case. This causes dramatic variation in the simulation data, and matches the trend found in the experiments for low exit aspect ratio nozzles. This is not the case for nozzles of unity aspect ratio.
- The laminar flow and absence of shocks in the nozzle is validated due to the similarity of the test results to the simulation data for operating conditions in which turbulence or shocks may occur.

Chapter 6

Electrothermal Augmentation

As discussed in previous chapters, the microthruster performance is primarily gauged by the I_{sp}, which is an indication of the exit velocity or momentum of the propellant. This is a strong function of the chamber energy or temperature. By transferring stored electrical energy into the settling chamber, which increases the fluid enthalpy, the exit velocity will increase. In the isentropic nozzle, the exit velocity can be computed by a simple energy balance between the internal energy and the momentum of the gas in the chamber with that present at the exit. As can be found in Sutton [43], this result is:

$$u_{exit} = \sqrt{\frac{2\gamma RT_o}{\gamma - 1}(1 - \frac{P_e}{P_o})^{\frac{\gamma-1}{\gamma}}} \quad , \tag{6.1}$$

where u_e is the exit velocity, T_o is the chamber temperature, R is the propellant gas constant, and γ is the ratio of specific heats. The pressure ratio is from the exit plane to the chamber. This derivation assumes the nozzle injection velocity is very small as well as constant specific heats. This equation shows that as chamber temperature increases, fluid exit velocity increases.

This concept has been used for many years to increase thruster performance, and represents a trade-off of one consumable for another (power for propellant). While on-orbit, the power can be replenished from the solar arrays, whereas the gas can not. This is a trade-off between the extra weight needed in the power system for heater operation and that of the propellant savings. This trade-off is left for a systems level study.

6.1 Background

A variety of MEMS applications employ a heater. Microreactors heat reactants for chemical production [44]. Valves create sealing by thermal expansion of the poppet such as those developed by Redwood Microsystems [45]. Even fiber-optic switches have employed evaporating fluid as an optical switch. In general, each of these heaters has employed a thin-film metal or polysilicon as a resistor. Current passing through the resistor dissipates heat. The resistors can be winding to increase the surface area without decreasing the resistance, as is depicted in figure 6.1.

The concept is to get the heat from the source into the fluid through convective heat transfer. The goal is to maximize the heat transfer efficiency while minimizing the pressure drop across the heater. The heat transfer can be modeled by Newton's Law of Cooling which states

$$q = hA(T_w - T_f) \quad , \tag{6.2}$$

where h is the convection coefficient, A is the surface area, q is the heat transfer rate, and the subscripts f and w refer to temperatures of the fluid and wall respectively. The convection coefficient is set by flow parameters, such as Reynolds number, which will be described in the section on heater modeling. At, this point the important driver is the wall surface area. By increasing the convective area, the power transferred

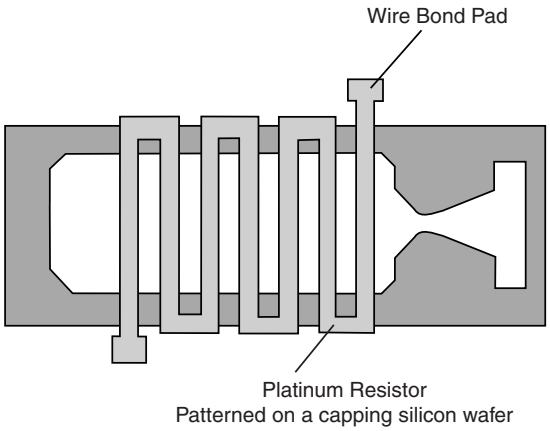


Figure 6.1: Metal or polysilicon resistor defined by photolithography techniques for resistojet applications

increases. This is very important when making a design choice between possible micromachined heater architectures.

6.2 Resistojet Architecture

The thruster is designed to operate with similar packaging to the cold-gas system, with slight modification to minimize conduction losses into the supports. The benchmark thruster is to operate with a chamber temperature of 700°C at a pressure of 2.5 atmospheres and a flow rate of 1000 sccms through a 50 micron nozzle of 25:1 area ratio. Without losses, this would result in an I_{sp} of 159 seconds.

5.8 Watts of dissipated power would be required for the flow to reach the goal temperature of 700°C. Using equation (6.2) to solve for area and a convection coefficient of 280 W/mK computed from equation (6.15), a chamber 1 cm long and 1 mm wide would be required.

In order to improve the heat exchanger, the convective area must be increased over that available with a flat-plate heater. For this reason, an array of fins is fabricated in the chamber in parallel with the nozzle. Figure 6.2 is an illustration of the design. A copper electrode will supply current to the silicon. The fins serve to increase surface area, and as resistors for dissipating the heat.

The conductive properties of silicon offer several advantages to the system level design. Most obvious, it allows the fins to serve as the heat source. Second, the highly doped silicon moves the intrinsic operating point to very high temperatures ($\geq$ 700°C). This offers a very stable operating point while the heater is operated in a constant current fashion. For temperatures lower than the intrinsic point, increases in temperature increase resistivity, which increases power dissipation and temperature. If the temperature increases above the intrinsic point, the reverse effect occurs. Fluctuations that incur higher temperatures result in a resistance drop and consequently a power drop. This cooling occurs until the intrinsic point is again reached. Thus, the intrinsic point is a naturally stable operating condition. This behavior is described in detail in section 6.3.2.

At this juncture, the effectiveness of on-chip insulation is investigated. The advantage of micromachining is that nearly all materials are selected for the ability to insulate or conduct electricity. However, due to the thermophysical properties of matter, good electrical conductors are generally good thermal conductors, and the same is true for insulators. This might allow the fins to be fabricated on an insulator, and prevent the heat from escaping into the substrate. The alternative is to allow the entire resistojet chip to run at the chamber temperature (due to the high thermal conductivity of silicon), and insulate at the die level. However, since the insulated thruster will take longer to heat up, it will suffer in performance during a short

duration firing, and the isothermal thruster architecture will be implemented.

Several compromises were made in order to demonstrate the utility of the first generation resistojet. Better design in future generations will minimize thermal losses as well as the thermal transient. These compromises were:

- The nozzle throat width was chosen to be identical to the fin spacing so that the loading effect would minimized. The nozzle throat can be smaller in future iterations to increase Reynolds number for the same thrust.
- Large voids are place on either side of the chamber to decrease heat dissipation in a region not interacting with the gas. The voids were kept smaller than desired from an electrical standpoint to increase the strength of the die, which needs to withstand the clamping force of the electrode. Future generations should be made from a honeycomb pattern to increase strength and minimize conduction, as well as, thermal mass.
- A large majority of the heat loss could be recovered by making the gas supply line the conductive lead. This would rely on a more advanced bond to the lead than was used here.
- The device was designed with a larger plenum than necessary to facilitate the attachment of gas lead, and to leave room for the clamp for the electrical leads.

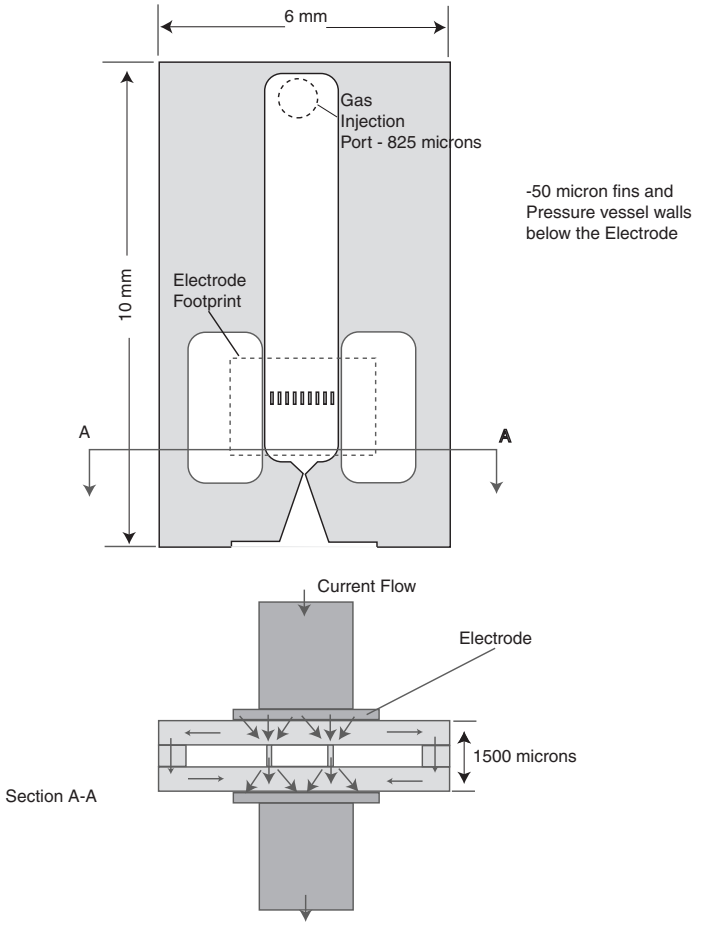


Figure 6.2: a) Detail of completed resistojet etch b) bonded resistojet with packaged electrode. Current is passed through chip, and preferentially passes through fins due to large resistance around the voids.

- A copper film could be deposited directly to the silicon through E-beam deposition. The copper lead could then be brazed directly to the device to eliminate the need for a clamp.

6.3 Heater Thermal Model

With a heater design selected, the geometry is to be optimized to yield the highest heater effectiveness while maintaining the lowest pressure drop. The heater effectiveness is defined as

$$\epsilon = \frac{T_{\text{fluid, exit}} - T_{\text{fluid, inlet}}}{T_{\text{wall, exit}} - T_{\text{fluid, inlet}}} \quad , \tag{6.3}$$

This is the ratio of the temperature rise of the fluid to the maximum possible temperature rise that could occur for this wall temperature. An infinitely long heater would bring the fluid up to the wall temperature.

The heater design parameters can be divided into two groups: those that govern heat transfer and those that govern power dissipation. Heat transfer in this instance is a convective transport problem and will be governed by the bulk motion of the fluid. Thus, fin spacing or gap (w_{gap}), the height of the channel (h_o), the length of the channel (L), and the number of channels form the heater design space. The power dissipation is governed by the fin width, and the resistivity of the silicon.

The operational parameters of the device are the mass flow rate ($\dot{m}$), the current supplied to the device (i), and the gas supply temperature. As will be explained later, the heater is operated in a constant current mode. With these parameters specified, a thermal model is implemented that will determine for the steady-state operating condition: the power that is dissipated, the pressure drop across the fins, and the exit temperature, which will allow effectiveness to be computed.

6.3.1 Governing Equations and Correlations

A 1-D model is selected so that variations at each local cross-section can be neglected, and only variations along the length of the fin are examined. As the fin spacing grows, there will be a larger thermal variation at each cross-section. But for the spacing examined in this analysis, the 1-D model can be justified by applying the classic problem of temperature distribution in fully developed duct flow, as derived in White [29]. The two-dimensional energy equation, which is written as:

$$\rho c_p u \frac{\partial T}{\partial x} = k \frac{\partial^2 T}{\partial y^2} \quad , \tag{6.4}$$

where k is the thermal conductivity of the fluid, c_p is the specific heat of the fluid, and u is the velocity. The x direction is the direction of the flow, and the y direction is normal to the wall, and are defined in figure 6.3. The aspect ratio of these channels are 10 to 1, so the z-components are neglected. The wall temperature is known because of the nature of the heating. The flow is symmetric, which implies a Neumann condition at the center. The temperature variation along the fluid ($\frac{\partial T}{\partial x}$) is assumed constant. Therefore, the temperature profile at a given x location can be found by integrating the energy equation, which reduces to

$$T_w - T = w_{fin}^2 \frac{\rho c_p u}{k} \frac{\partial T}{\partial x} \left(\frac{y}{w_{gap}} - \frac{y^2}{w_{gap}^2}\right) \quad . \tag{6.5}$$

where T_w is the wall temperature, w_{fin} is the width of the fin, w_{gap} is the fin spacing, and k is the fluid conductivity. The temperature variation across the channel is 20°C at the exit for a 10 m/s flow through a 50 micron channel of 700°C wall temperature. This is small compared to the magnitude of the temperature, and will be neglected for design purposes.

The model assumes that heat is being generated at a rate set by the electrical model, which is detailed later. The dissipation is treated as a volumetric heating source within the fin. The heat transfer is computed by performing an energy balance on differential fin and fluid elements as shown in figure 6.3. The heat is

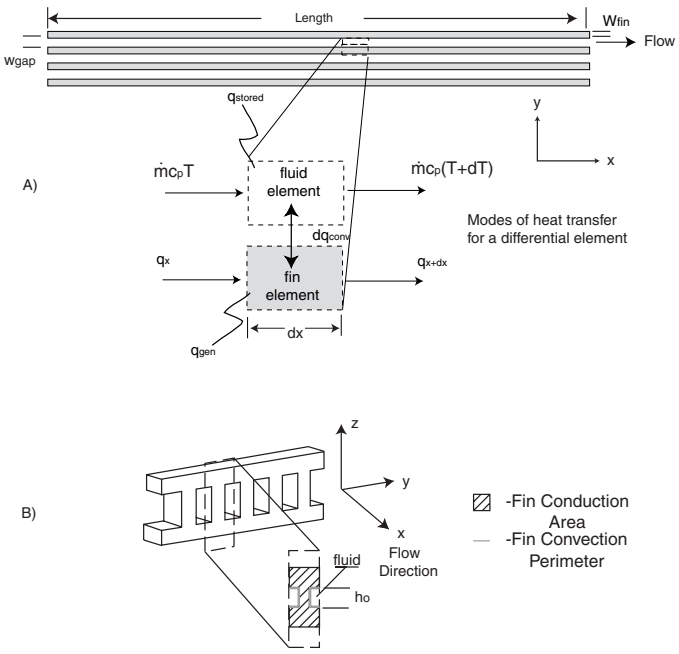


Figure 6.3: A) An energy balance is performed on differential elements of the fin and fluid to determine equilibrium temperatures. Arrows represent heat transfer paths B) Effective Heater cross-sectional area and convective area of the full geometry are used despite this being a 1-D analysis.

transferred to the fluid through Newton's Law of cooling from equation (6.2) and into the up and downstream portions of the fin through conduction.

In order to enhance the fidelity of the model, and take advantage of the heater symmetry, the heat transfer analysis is performed for the fin segment shown in figure 6.3b. The channel symmetry plane is the boundary on both sides of the fin. The conduction area is the full cross-section, and the convection area is the wetted area outlined. By taking into account the upper and lower surfaces, a closer approximation to the heater geometry is made.

The energy inputs for both the fin and fluid control volumes obey the following energy balance.

$$\dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen} = \dot{E}_{stored} \quad . \tag{6.6}$$

where $\dot{E}$ is the heat power. Applying this equation to each control volume, the following system results.

$$-k_w A_c \frac{d^2 T_w}{dx^2} dx + hPdx(T_w - T_f) - \dot{q} A_c dx = 0 \tag{6.7}$$

$$\dot{m} c_p dT_f = Ph(T_w - T_f)dx \quad , \tag{6.8}$$

where equation (6.7) is defined for the fin, and equation (6.8) is defined for the fluid. The first term in equation (6.7) is the conduction along the fin. The second term is the convective transfer to the fluid. The final term is the volumetric heat dissipation. Equation (6.8) is the balance between the energy convected downstream, on the left-hand side of the equation, and the heat transfer to the wall, on the right hand side. k_w is the fin thermal conductivity. A_c is the fin cross-sectional area that governs conduction through the fin. P is the perimeter of the channel cross-section that governs the convective heat transfer. h is the convection coefficient. $\dot{q}$ is the volumetric heat generated. c_p is the coefficient of specific heat. By dividing each equation through by the differential dx, and cancelling terms, the following system of governing equations is established:

$$\frac{d^2T_w}{dx^2} + \frac{hP}{k_w A_c}(T_w - T_f) - \frac{\dot{q}}{k_w} = 0 \tag{6.9}$$

$$\frac{dT_f}{dx} = \frac{Ph}{\dot{m}c_p}(T_w - T_f) \quad . \tag{6.10}$$

This coupled system requires three boundary conditions. The fins are assumed to be insulated at each end, and the inlet fluid temperature is specified. These are formulated as

$$\left.\frac{dT_w}{dx}\right|_{x=o} = 0; \;\; \left.\frac{dT_w}{dx}\right|_{x=L} = 0; \;\; T_i = 300K; \tag{6.11}$$

The equations and boundary conditions can be non-dimensionalized by the inlet temperature (T_i) and channel height (h_o) to simplify the equations to three parameters.

$$\frac{d^2T_w}{dx^2} + Bi(T_w - T_f) - s^* = 0 \tag{6.12}$$

$$\frac{dT_f}{dx} = St(T_w - T_f) \tag{6.13}$$

$$\text{where} \quad Bi = \frac{hPh_o^2}{k_w A_c}; \;\; s^* = \frac{\dot{q}h_o^2}{T_i k_w}; \;\; St = \frac{hh_o P}{\dot{m}c_p}$$

As denoted by their symbols, the coefficient of the wall-fluid temperature difference in the fin equation is a form of the Biot (Bi) number, and in the fluid equation the coefficient of the wall-fluid temperature difference is a form of the Stanton (St) number. The non-dimensional source term is represented by s^*.

It is important to observe the relevance of each of the non-dimensional parameters. The Biot number governs the ratio of the conduction along the fin to the convective heat transfer. High Biot numbers combined with high fin length-to-width aspect ratios result in large temperature differences between the entrance region and exit region of the fins. This is because the heat transfer in the entrance region is so high that all of the heat generated locally goes into the fluid, and relies on heat generated downstream to be conducted along the fin, so it can enter the fluid in the entrance region. The Stanton number plays a similar role, as it governs the ratio between the convection rate and the advection rate along the fluid. Thus, when Biot number is low and Stanton number is high, the fins are nearly isothermal. Finally, the source term merely scales the volumetric heat dissipation, relative to that conducted along the wall. When the ratio of Stanton number to the source term is high, then the heater effectiveness will be high. This is due to all of the dissipated heat being carried away by the fluid, which is indicative of a high Stanton number flow.

As mentioned previously, the heat transfer is governed by the fluid mechanics occurring within the channel. Thus, the convection parameter h is a function of Reynolds number as well as the distance along the fin the fluid has traveled. The correlation for computing the heat transfer coefficient was developed by Stephan and is reported by Kakač et al. [46] for developing flow along parallel duct walls. This is computed from average Nusselt number based on constant temperature walls and Reynolds number based on inlet velocity:

$$Nu_T = 7.55 + \frac{0.024(Pr/X^*)^{1.14}}{1 + 0.0358\frac{(Pr)^{0.81}}{(X^*)^{0.64}}} \tag{6.14}$$

where

$$Pr = \frac{\mu c_p}{k_w}; \;\; X^* = \frac{\frac{x}{D/2}}{RePr}; \;\; Re = \frac{\rho u_i w_{gap}/2}{\mu}; \;\; Nu_T = \frac{hw_{gap}/2}{k_f} \quad .$$

6.3.2 Electrical Model

The source term in the heater model is a function of the volumetric heating occurring within the fin. The heating occurs because of ohmic dissipation from the current passing through the silicon. The resistivity of the silicon, however, varies with temperature.

At low temperatures, a pure silicon lattice is a perfect insulator because all of the available electrons are covalently bonded to the neighboring silicon atoms in the lattice. As the temperature increases, the energy of the thermal vibrations is high enough to break some of these bonds, creating electron-hole pairs, which are free to move within the lattice. A hole is the positive charge imbalance left behind once an electron leaves the valence band. Holes are also charge carriers, and can be conducted through the lattice just as electrons. These thermally-generated electrons and holes are referred to as intrinsic carriers. The number of intrinsic carriers (n_i) are a function of temperature and can be found from statistical mechanics. This is formulated in Ashcroft and Mermin [47] as

$$n_i = 4.83 \times 10^{21} T^{3/2} e^{-\frac{E_G}{2kT}} \quad , \tag{6.15}$$

where k is the Boltzman constant of $8.69 \times 10^{-5} eV/K$, T is the temperature, and E_G is the band gap energy, which for silicon is $1.12\ eV$. The electrical conductivity (σ) can be computed as a function of carrier mobility, and the charge of each carriers by

$$\sigma = q[\mu_n n + \mu_p p] = q[\mu_n + \mu_p] N_i = \frac{1}{\rho} \quad , \tag{6.16}$$

where n and p are the concentration of electrons and holes and μ_n and μ_p are their mobilities. The conductivity is equivalent to the inverse of the resistivity Since the electrons and holes (of charge q) are created simultaneously, $N_p = N_n = N_i$ for an intrinsic conductor . The carrier mobility (μ) quantifies the ease with which the carriers move through the lattice and decreases with increasing temperature. This, in turn, decreases the conductivity, and is the basis for the temperature coefficient of resistivity.

Silicon, in the lattice, can be substituted by a dopant atom. Typical dopants are boron (p-type) and phosphorous (n-type). Boron has one less valence electron than silicon, and thus contributes a hole to the lattice without a corresponding electron being created. This is termed an acceptor, and the concentration of acceptor carriers is N_A. Phosphorous has one more valence electron that it contributes to the lattice without a corresponding hole being created. This is termed a donor, and the concentration of donor carriers is N_D. In an extrinsic semiconductor, the carriers are almost entirely from the dopant atoms. Not enough silicon to silicon bonds have been broken for the intrinsic carrier concentration to be significant. The conduction is still governed by equation (6.16); however, in a p-type wafer $p \approx N_A \gg n_i$, and the equation can be reformulated as

$$\sigma = q \mu_A N_A \quad . \tag{6.17}$$

Therefore, when the temperature of the silicon is low, the conductivity is extrinsic, and is governed by equation (6.17). As the lattice is thermally excited, and $N_i \sim N_A$, the semiconductor becomes intrinsic, and the conductivity is governed by equation (6.16). Figure 6.4 illustrates the variation in resistivity (the inverse of conductivity) with temperature as measured by Pearson and Bardeen [48]. As the temperature increases, the resistivity gradually increases due to the mobility reduction. Once the intrinsic carriers become significant, there is a dramatic reduction in the resistivity. In this model, a p-type doping of $1x10^{19}$ atoms/cc is used because it is the highest doping of commercially available silicon wafers. This results in an intrinsic temperature of $\sim 700°$C.

Heater Stability

The previously mentioned stability will prevent thermal runaway from occurring in the heater. For gases, the viscosity increases with temperature. Thus, if a local hotspot occurs, the pressure drop across a particular

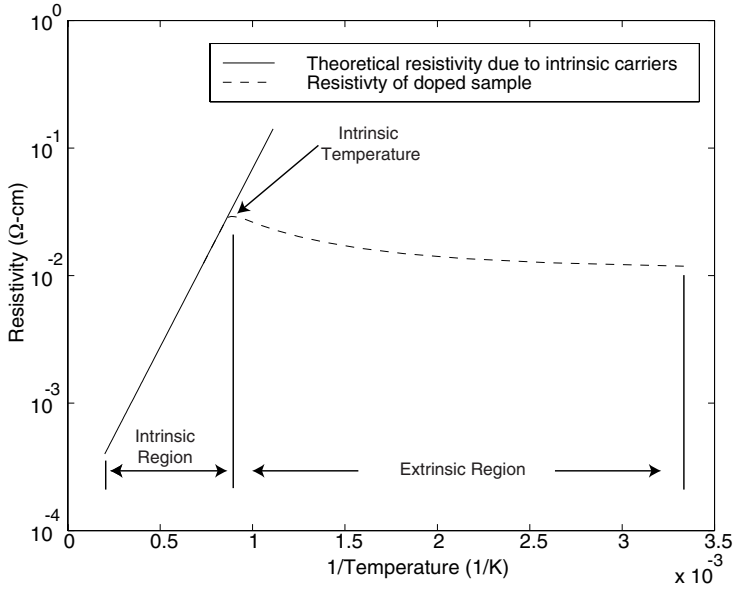


Figure 6.4: Resistance variation with temperature for boron-doped silicon with a doping of 1×10^{19} carriers/cc [48]. The intrinsic point occurs when the silicon carriers are dominant over the dopant acceptor carriers. This is the stable point during constant current operation.

channel could increase dramatically thus lowering the heat transfer due to low flow, and ultimately leading to failure. Such a failure was investigated by Kalamas [49] for particle bed nuclear reactors which is a similar heat transfer problem to that of the bank of fins, operating at similar Reynolds numbers. He found through a stability analysis that for a similar temperature rise, the reactor became unstable at Reynolds numbers based on a particle size of 10. The Reynolds number based on fin diameter (similar to the particle diameter established by Kalamas) is on the order of 100, and assures the stability of the system. However, the nature of the fin electrical conductivity due to availability of intrinsic carriers will reduce the power output and mitigate any instability.

Heat Dissipation

The electrical model would not be complete without an understanding of how current density distribution affects heat dissipating in the cross-section shown in figure 6.3b. In this figure, the electrodes are located along the top and bottom branches of the "I" and current passes vertically. To determine the current density, and hence power dissipation, Laplace's equation must be solved inside this domain. The governing equation is

$$\nabla^2 V = 0 \quad , \tag{6.18}$$

where V is the voltage with respect to ground. The current density (j) is then computed from

$$j = \sigma E \quad , \tag{6.19}$$

where σ is the conductivity of the material, and E is the E-field, which is the spatial derivative of voltage ($\frac{\partial V}{\partial X}$). Ultimately, the volumetric heating is of interest, and can be computed from

$$\frac{P}{V} = E\dot{j} \tag{6.20}$$

$$= \sigma E^2 \quad . \tag{6.21}$$

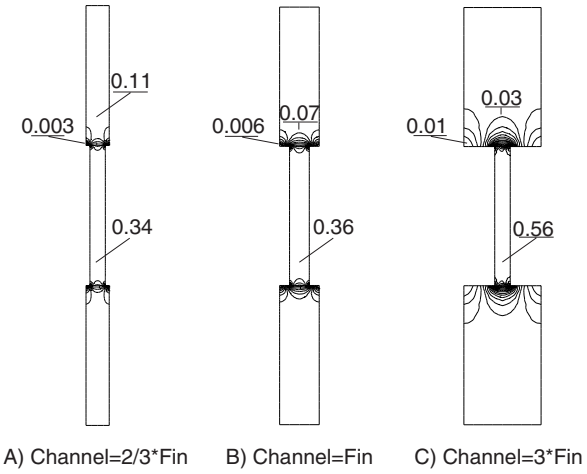


Figure 6.5: Contours of power dissipation throughout the domain. The values are symmetric about the centerline of the fin. Non-dimensional quantities can be converted to dimensional values with equations (6.22) and 6.21.

In order to understand the relative potential throughout the domain, the governing system is non-dimensionalized by the applied voltage. This allows the dissipation to computed in the capping wafer, and the fins. The following relations are a result of the non-dimensionalization.

$$V^* = \frac{V}{Vo};\ E^* = \frac{H_o}{V_o}E;\ j^* = \frac{\sigma V_o}{H_o}E^*;\ X^* = \frac{X}{H_o}\quad , \tag{6.22}$$

where H_o is the height of the fin. Once again, symmetry can be applied to the problem, both along the horizontal and vertical. The planes of symmetry that define the fin become insulated boundary conditions, and are formulated as

$$\frac{\partial V^*}{\partial X^*} = 0\quad , \tag{6.23}$$

whereas, the boundaries that are in contact with the electrodes are Dirchlet conditions set to the value of the applied voltage. In the non-dimensionalized system, the boundary conditions are

$$V^* = 1 \text{ on the upper surface} \tag{6.24}$$

$$V^* = 0 \quad \text{on the lower surface}\ . \tag{6.25}$$

A simulation using the Matlab PDE toolbox was used to compute the E-field throughout the domain. Figure 6.5 shows the volumetric dissipation in the cross-sections of different geometries. The governing parameter is the relative width of the fin to the channel. This influences the amount of dissipation which occurs in the capping wafers relative to the fin. In the limiting case, of an infinitely wide channel, the dissipation would occur entirely in the fin. Conversely, if there were no channel, the dissipation would be uniform.

For Case A, the dissipation in the capping wafer is 32% of that within the fin. (The E-field in the capping wafer is 56% of fin; however, the dissipation goes like the E-field squared.) Case B has 20% and case C has 5%. These values are combined to compute the total dissipation within the fin, and are fed directly into the thermal model as the source term.

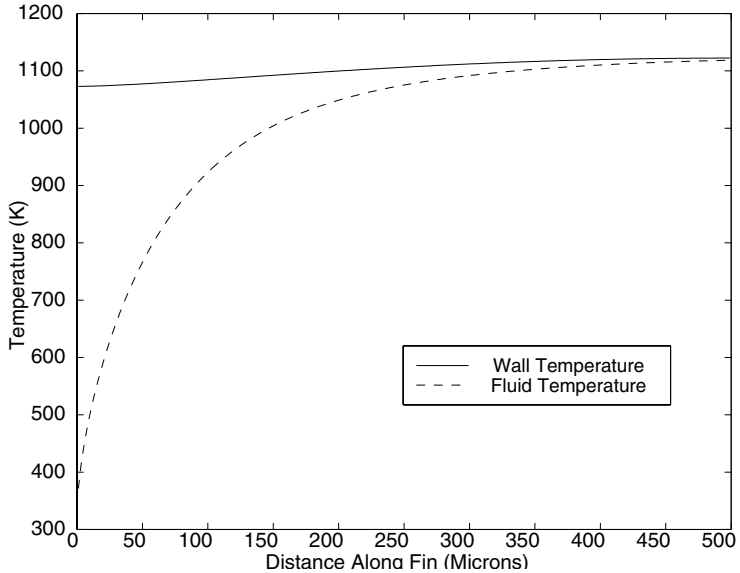


Figure 6.6: Temperature Profiles for a heater with 5 fins 500 microns long, 100 microns wide, and a gap of 50 microns between fins. The governing parameters are $Bi = 1.09$, $S^* = 0.46$, and $St = 6.51$

6.3.3 Design Studies

Returning to the goal of finding the optimum length that minimizes pressure drop and maximizes effectiveness, a number of model test cases were run. To best understand the effect of the governing non-dimensional parameters, let us consider four cases and determine how they drive the solution to the optimum. Figures 6.6-6.8 depict these temperature profiles for various operating parameters. All cases are operating at a mass flow rate of 7.87×10^{-6} or 378 sccms. This was chosen to keep the operating power below 10 watts, and at the same time maintain as high a throat Reynolds number as possible (approximately 550 at this operating condition).

Figure 6.6 depicts the wall and fluid temperature profiles for a heater with 5 fins 500 microns long, 100 microns wide, and a gap of 50 microns between fins. This results in a heater effectiveness of 99.5% and a pressure drop of 0.2135 atm. This would be a very reasonable design. The pressure drop is small, and nearly all of the possible temperature rise is achieved. The governing parameters for this arrangement are $Bi = 1.09$, $S^* = 0.46$, and $St = 6.51$.

Figure 6.7 depicts the wall and fluid temperature profiles for a heater with 10 fins 125 microns long, 50 microns wide, and a gap of 50 microns between fins. In this case, the non-dimensional parameters are $Bi = 1.76$, $S^* = 1.47$, and $St = 12.96$. This results in a heater effectiveness of 97.02% and a pressure drop of 0.026 atm. The ratio of the Stanton number to source term has dencreased from the previous case, and as expected, the effectiveness has dropped, but only by 3%. However, the pressure drop has decreased by a factor of 10. Thus, this is deemed a much better design due to the higher injected pressure. The fins remain almost isothermal, and the low cross-section of the fins will increase the electrical resistance, thus driving the current down. For both cases, the dissipated power is 7.67 Watts, which is set by the flow rate and desired temperature rise.

The fin gap is kept constant between the two cases for fabrication reasons. The gap should match the throat diameter, so that a halo mask can be used as described in chapter 4. This will allow a constant loading to be maintained during etching, and prevent features from distorting.

Figure 6.8 depicts the wall and fluid temperature profiles for a heater with 5 fins 125 microns long, 50 microns wide, and a gap of 50 microns between fins. In this case, the non-dimensional parameters are $Bi = 1.80$, $S^* = 2.52$, and $St = 6.61$. This results in a heater effectiveness of 85.1% and a pressure drop of 0.047 atm. By halving the number of fins, the mass flow through a given channel is doubled. This reduces the temperature rise, as well as, increases the pressure drop. This is realized in the lowest Stanton number to source term ratio, and the lowest effectiveness, yet. This is clearly not the direction in which the design

should move.

Finally, the case represented by the surface micromachined resistor shown in figure 6.1 is analyzed. This would be equivalent to a case with 1 fin. Even with a chamber 7 mm long, the heater effectiveness is only 90.4% with a pressure drop of 0.014 atm. Thus, using the design with fins, a much more compact arrangement can be attained with a similar pressure drop.

Table 6.1 details the design space studied and results obtained with the thermal model. The first entry is the design selected as optimal for the given fabrication and operation.

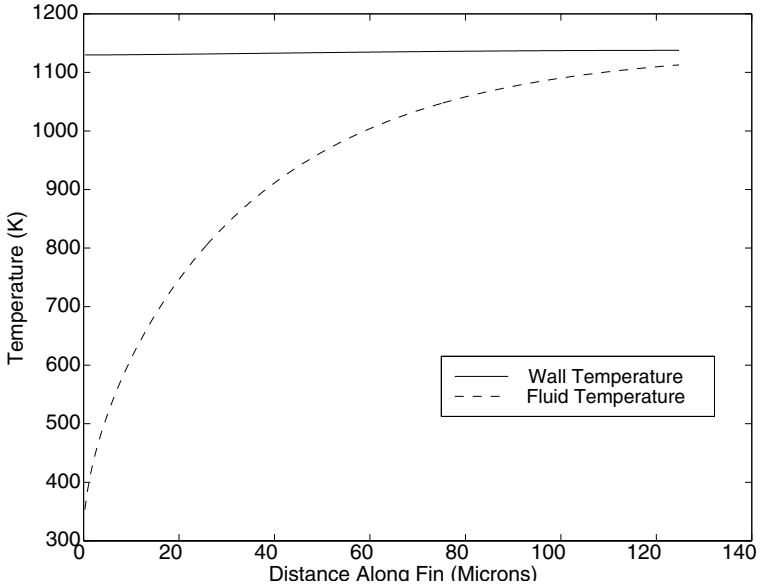


Figure 6.7: Temperature Profiles for a heater with 10 fins 125 microns long, 50 microns wide, and a gap of 50 microns between fins. The governing parameters are $Bi = 1.09$, $S^* = 0.46$, and $St = 6.51$

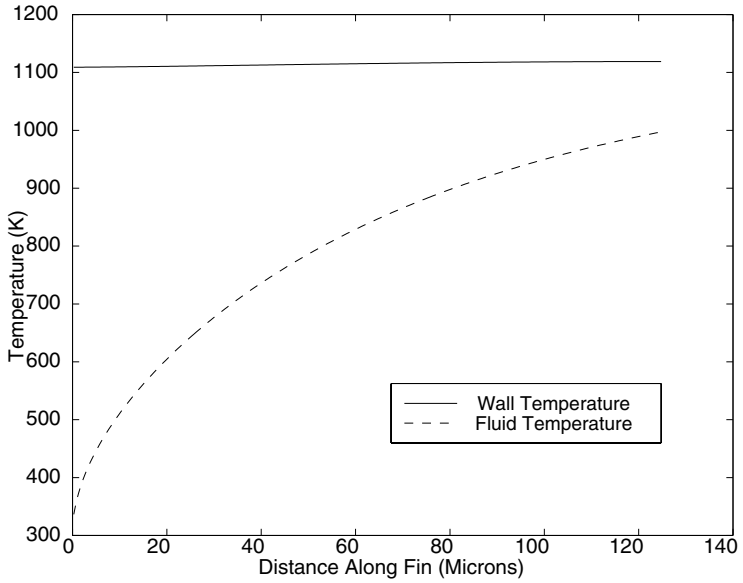


Figure 6.8: Temperature Profiles for a heater with 5 fins 125 microns long, 50 microns wide, and a gap of 50 microns between fins. The governing parameters are $Bi = 1.80$, $S^* = 2.52$, and $St = 6.61$

$w_{gap}(\mu m)$	Length (μm)	Number of Fins	$w_{fin}(\mu m)$	Current	Power In (Watts)	$T_{fluidexit}$	Pressure Drop (atm)	Effectiveness (%)
50	125	10	50	1.28	7.62	1088	0.0265	97.0
50	2540	10	50	6	7.7416	1125	0.546	99.99
50	2000	10	50	5.27	7.665	1117	0.4265	99.99
50	1500	10	50	4.53	7.68	1119	0.3204	99.98
50	1000	10	50	3.67	7.67	1118	0.2135	99.96
50	500	10	50	2.59	7.72	1123	0.1073	99.92
50	250	10	50	1.82	7.67	1118	0.0534	99.7
50	125	5	50	0.84	6.53	1118	0.0471	85.14
50	125	25	50	2.23	7.69	1120	0.0107	99.96
100	125	25	50	2.23	7.71	1122	0.0013	99.13
50	500	5	100	2.39	7.67	1122	0.2135	99.54
25	1000	10	50	2.72	7.84	1137	1.7384	99.98
1000	5000	1	50	3.35	6.94	1040	0.0099	86.68
1000	7000	1	50	4.12	7.09	1056	0.014	90.44

Table 6.1: Fin geometry test cases and operating conditions used for analysis

6.3.4 Parasitic losses

While the model presented in the previous section is effective for determining the steady-state temperatures, it does not consider the losses of heat energy through conduction through the gas supply line, electrical leads, or the radiation to the environment. Since the heater is to be run at the intrinsic temperature, the losses can be estimated from Fourier's Law for the conduction and the Stefan-Boltzman Law for radiation, which is stated as

$$q = \epsilon\sigma A(T_s^4 - T_{environment}^4) \quad , \tag{6.26}$$

where A is the surface area, ϵ is the surface emissivity, σ is the Stefan-Boltzman constant which is $5.67 \times 10^{-8}\frac{W}{m^2K^4}$. The surface emissivity of a highly polished surface, such a silicon, is about 0.1. The surface area of the device is 1.67 cm^2. The radiation losses alone could account for 4.1 Watts. The conduction through the gas injection tube which has a cross-sectional area of $2.55 \times 10^{-7}m^2$ and a thermal conductivity of $17.3\frac{W}{mK}$ is 0.1 Watt.

The conduction losses through the leads is a bit more complex since they are carrying current, which generates heat within the lead. A simple analysis of ohmic heat generated in a differential element, with known boundary temperatures, can be used to compute the heat loss. This was presented by Martinez-Sanchez [50]. The heat loss is formulated as

$$q = Ak_{lead}\beta\frac{T_{hot} - T_{cold}\cos(\beta L)}{sin(\beta L)} \tag{6.27}$$

where

$$\beta = \frac{I}{A}\sqrt{\frac{\alpha}{k_{lead}}} \quad , \tag{6.28}$$

where α is the temperature coefficient of resistivity, which is $6.69 \times 10^{-11}\frac{\Omega - m}{K}$ for copper, I is the current, k is the thermal conductivity which is averaged to $360\frac{W}{m^2K^4}$. Over the range of currents 0.5-10 Amps, the heat loss is 3.2-7.0 per lead. The length of the lead was not optimized since it was constrained by the requirements of the vacuum chamber and thrust stand. If this lead length is optimized, this could be reduced to 0.2 Watts per lead, a substantial savings.

The total heat loss for the current required is ~ 25 watts. This is in addition to the 5.8 Watts required to operate at the benchmark mass flow rate.

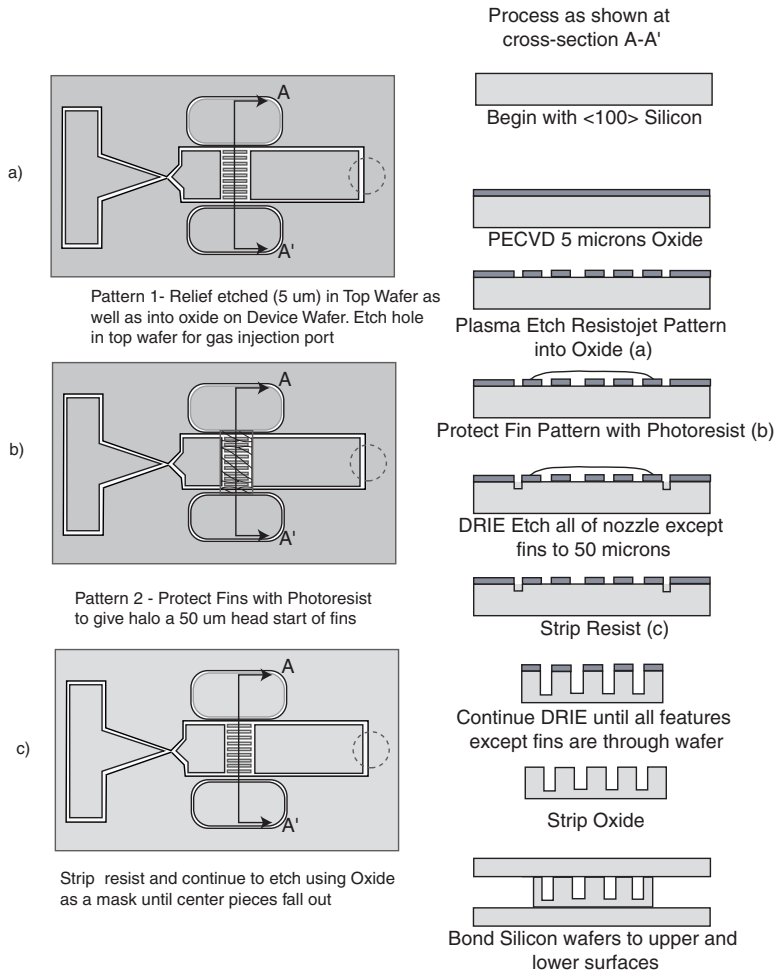


Figure 6.9: Fabrication sequence of microresistojet.

6.4 Resistojet Fabrication

With the design selected from the trade study performed on heater geometries, it is fabricated in a similar fashion to the cold-gas thrusters described in chapter 4. The same etching rules apply in order to achieve the highest anisotropy possible. However, a new technique must be implemented to achieve the suspended fins in the settling chamber.

Figure 6.9 illustrates the fabrication sequence required to maintain the integrity of the fins after the through etch of the nozzle. A halo mask is implemented as before, however, it must be nested so that the thruster outline is completed before the fins are completely etched through. A nested mask is achieved by etching the nozzle fin pattern in oxide that is deposited using PECVD. This can be deposited to thickness up to 40 microns, however, the residual stress results in substantial wafer bowing. Since the selectivity of the STS etcher of silicon to oxide is about 150:1, 4 microns is sufficient to mask an etch through the wafer. An anisotropic etch of the oxide with photoresist as the mask can be performed in the MTL using the AME 5000 etcher, and a CHF_3 chemistry.

With the pattern now etched in the oxide, the fins can be masked with resist, while the rest of the pattern is being etched. This etch is performed to a depth between 50 and 100 microns. When the resist is subsequently stripped and the full pattern etched. A bus bar supporting the fins will remain once the feature etch is complete. The rest of the lanes will etch completely through and the nozzle and plenum center bodies will drop out. The final device is illustrated in figure 6.2. The capping wafers are attached through a fusion

bond to silicon wafers of the same doping as the thruster wafer. This way, all of the silicon will go intrinsic at the same temperature, and the resistances will be similar between the upper, middle, and lower surfaces.

The structure is packaged by clamping two copper electrodes to the thruster as shown in figure 6.2b. The potential difference in the two electrodes will cause current to flow through the fins, and outer chamber wall. The void on either side of the fins is to create a large resistance so that the current flows preferentially through the fins and not the surrounding material. As mentioned earlier, this void could be made larger, but is minimized for the first generation thruster to support clamping. The upper surface has been pre-etched with a hole, which is aligned with the chamber, so that gas can be injected through the upper surface. The gas is injected into the hole through a 5 cm length of Kovar tubing which is attached to silicon through the use of a glass frit. The opposite end of the tubing is brazed to a stainless steel plate. The plate is then clamped with an O-ring to the same manifold as described in chapter 4. This allows a high temperature gas interconnect to be achieved.

The greatest difficulty in the fabrication sequence is the nested masking step. As illustrated in chapter 4, figure 4.2, the passivation step leaves a residual polymer on the sidewalls. When the photoresist mask is removed and the space between the fins is to be etched, there is residual polymer on the wall intersecting the chamber etch. This polymer causes a very small sheet of silicon to remain intact and possibly block the flow between the fins. Figure 6.10 is a scanning electron micrograph of the first attempt at the nested mask. There is silicon "grass", which is approximately one-half of the fin height, obstructing the flow path.

It is known that the polymer can burned off with an oxygen plasma, but it requires direct impingement of the ions. Since the polymer is on the sidewalls, it is difficult to obtain much reduction in the grass with an intermediate oxygen plasma alone. The silicon etch parameters must be optimized. By increasing the plasma density and ion energy by increasing the coil power on the etch step, the resulting grass can be considerably diminished. Figure 6.11 shows the bank of fins etched with the same recipe as before, but with a coil power of 800 Watts instead of 750 Watts. The grass is nearly eradicated. However, there appears to be considerable over etching of the fins. Their profile is tapered from top to bottom, and the width is approximately 15 microns smaller than the design. Figure 6.12 is a SEM of 775 Watt etch, which is half way between the previous two attempts. This etch yields very small grass height, and yet maintains the geometry of the features. Such a compromise in power is the result of tailoring the etch recipe to the mask. The oxygen plasma is still used after the resist strip and prior to the full feature etch for it is still credited with somewhat improving etch performance.

The etches performed during this fabrication sequence are 491 microns deep as opposed to the cold-gas nozzles, which were only 308 microns deep. Since the nozzles have larger features (50 micron throats), the etch must be deeper to maintain a similar exit plane aspect ratio. After fabrication, the nozzle specifications are: Fin length: 112.5 microns, Throat Width: 65 microns, Expansion Ratio: 8.26:1, Fin width: 41 microns, Fin Gap: 60.5 microns. The exit plane aspect ratio was 1.01.

The final concern is whether the capping wafer will properly bond to a fin that is only 50 microns wide. Flexibility in the bus bar as well as the weakness of the thin support could lead to the features not bonding, or in the worst case, fracture. However, the initial test shows that both 100 micron wide fins, as well as the 50 micron fins, are bonded to the upper surface, as shown in figure 6.13. If the bond did not occur, this would preclude current from flowing into the fins, which would be realized as a higher than expected operating voltage.

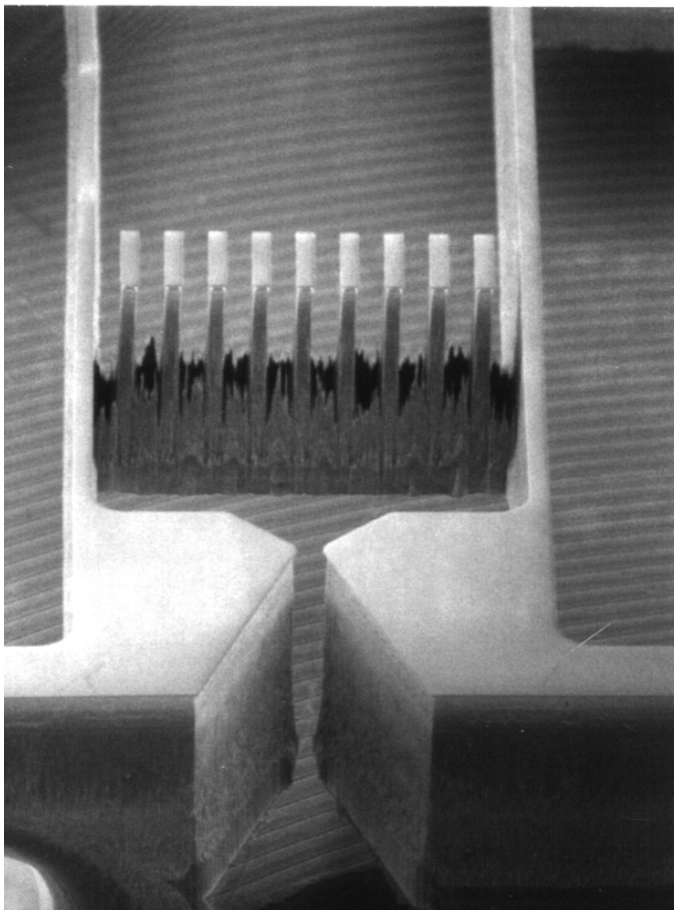


Figure 6.10: First attempt at the resistojet etch. Grass in between the fins will obstruct flow, and is a result of the nested mask

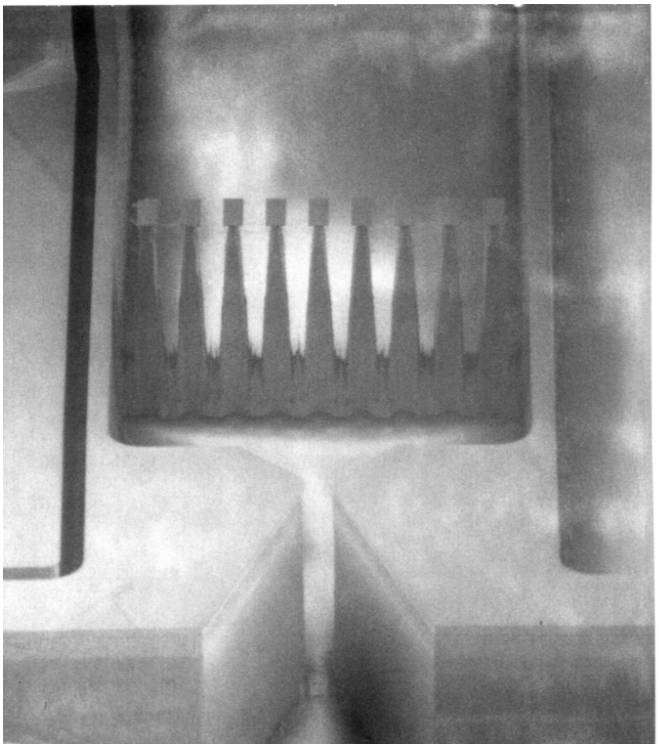


Figure 6.11: Increased coil power during the etch, following an oxygen plasma, has eradicated the grass at the expense of feature anisotropy

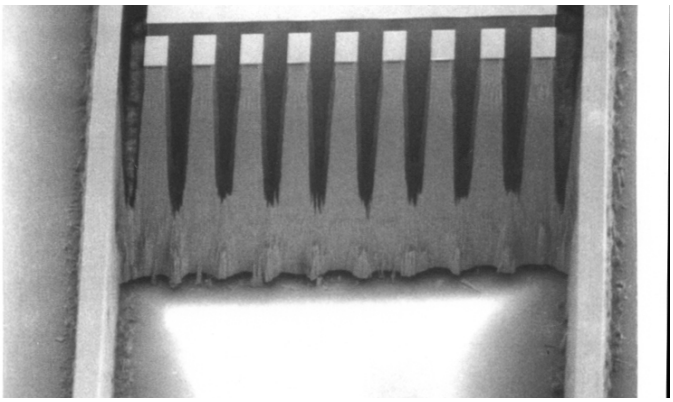
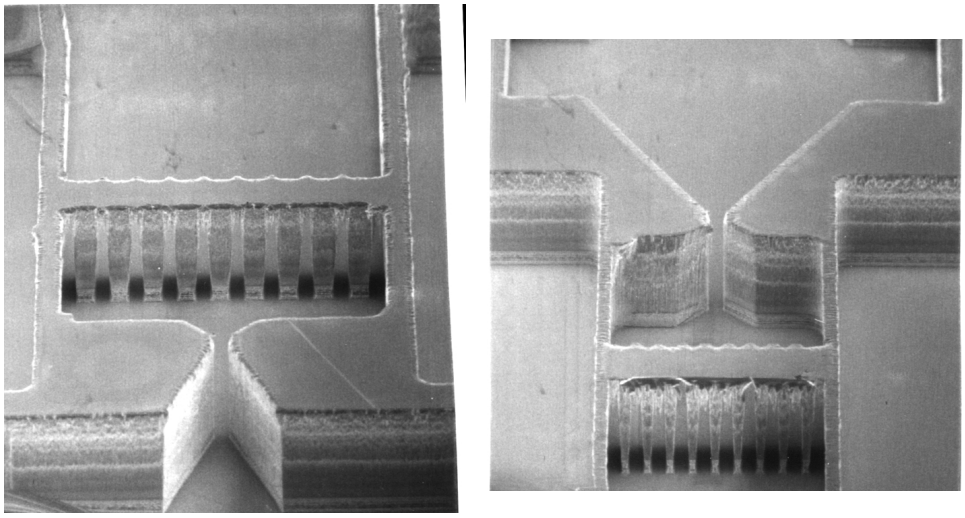


Figure 6.12: Intermediate coil power eliminates the grass and maintains etch anisotropy



a) 100 micron thick fins

b) 50 micron thick fins

Figure 6.13: SEMs of a) 100 micron fins with 50 micron gaps bonded to thesilicon wafer b) 50 micron fins bonded to the silicon wafer. The bond was successful for both dimensions on all fins. No fractures were incurred.

6.5 Resistojet Testing

With the silicon micromachined into the geometry of interest, the thruster is to be packaged to interface with testing equipment. One of the difficulties in working with silicon is making electrical contact. In general, a p-n junction occurs between the metal lead and silicon, which results in behavior similar to a diode. In a diode, current does not flow below a threshold voltage. In addition, there can be a large contact resistance between the atomically smooth silicon surface and the copper electrodes, which will be used to create the potential difference across the fins. In order to facilitate the copper-silicon interface, a metallic paste was used to make contact. SilverGoop^{TM}, a product of Swagelok, is designed as a thread lubricant for high temperature applications (up to 1500° F). Its silver base allows current to flow and since it is a liquid, it fills in the gaps between electrode and silicon. The SilverGoop is applied to the electrodes, which are clamped onto the die using a ceramic package. The package provides the necessary contact force, and thermally insulates the thruster from excessive heat losses. The copper leads are held inside ceramic sleeves to thermally, and electrically isolate them from the surrounding structure. The integrated set-up is depicted in figure 6.14. The Kovar tubing, which is attached to the silicon via the glass frit is brazed to a stainless steel backing. The backing is clamped via an O-ring to the manifold previously fabricated for the cold-flow experiments.

Temperature data is taken with a type-K thermocouple mounted directly to the silicon chip with SilverGoop as the adhesive. Since they are operated in series with an identical thermocouple at a known reference temperature, these thermocouples are accurate to ±4°C,. The current source is an HP-403a 0-20 VDC 0-120 A Power Supply. The readout on the supply is accurate to 10 milliamps, and 1 millivolt. Mass flow, pressure, temperature and power settings, as well as thrust, are read by a Keithly multi-channel voltmeter, and post-processed using Matlab.

6.5.1 Electrical Tests

The first goal was to demonstrate the capabilities of the heater prior to flowing gas through the device. The design criteria specified in table 6.1 are based on ideal conditions with no heat losses. Based on the required power, and the fin dimensions, the electrical resistance of the device should be 4.65 Ω. However, since this is a benchtop test, convective cooling will be significant in addition to those losses analyzed in section 6.3.4. Figure 6.15 illustrates in a strip-chart format the temperature and power histories of the heater at various

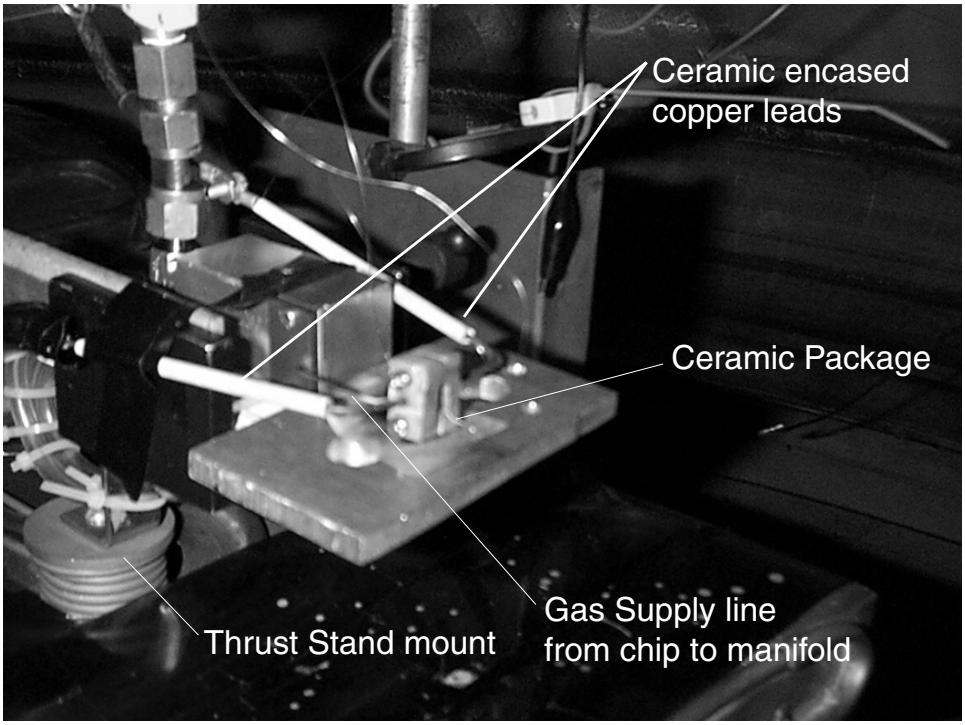


Figure 6.14: Integrated thruster testing in vacuum chamber. Copper leads are clamped to nozzle by means of a ceramic package. Gas is injected through a Kovar tube brazed to the manifold on one end, and attached to the silicon with a glass frit on the other.

current settings.

The temperature increases at about 25°C/sec during the initial rise. The temperature then rises to a steady value over the next 300 seconds. This is due to the increasing power dissipation with temperature. Once the current was set to 6.06 Amps, sufficient power was dissipated to reach the intrinsic temperature, and the temperature rose above that point, and subsequently plummeted with the drop in dissipation. It then increased, and dropped again, repeating the cycle until the steady state was reached. As can be seen from the figure, even while the current was continually increased, the power dissipation remained constant, as did the temperature. This is better illustrated in figure 6.16, which is the same data but plotted as temperature versus current. The current is set, and the temperature increases through increased dissipation from the change in resistivity of the silicon with temperature. The temperature is flat beyond 6 Amps due to the constant dissipation. There is still a long period oscillation occurring, which in time will stabilize, but its amplitude is less than 20 degrees. These figures allow us to experimentally derive the intrinsic temperature as being between $690 - 710$°C, and demonstrate the ability to stably operate at the intrinsic temperature. The benchmark assessments of a 900°C intrinsic temperature were based on the range of resistivities ($0.01 - 0.02\ \Omega - cm$) quoted for these wafers. The lowest value was selected for the analysis in the first half of the chapter. As it turns out, the higher resistivity was present, and a 700°C intrinsic temperature is realized. This coincides to the predicted value for that temperature.

In addition to the uncertainty in the resistivity, there was a deviation from the predicted resistance of the thruster. It was originally calculated to be 4.65 Ω at temperature, but is experimental determined to be 1.2 Ω. This is partially due to the resistivity being twice the assumed value. But it is also due to current leakage through other portions of the thruster, other than the fins. If the rest of the current path between the electrodes is modeled as the shortest distance through the silicon not passing through the fins, we find that the rest of the thruster has a resistance of one-half of the path through the fins. This increases the required current by a factor of 5 to achieve the required power dissipation in the fins, which is the same as the current measured during the heater operation.

One of the first trials was operated in constant voltage mode due to a malfunction in the power sup-

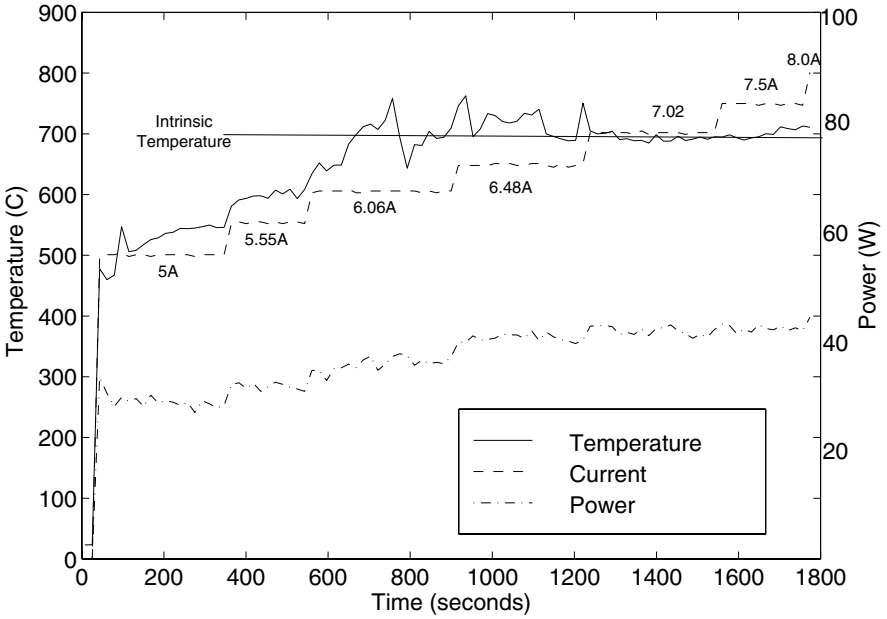


Figure 6.15: Heater Temperature during the test at various current and dissipated power settings. The time history plot illustrates transient behavior as well as power and current levels needed to achieve a given temperatures.

ply. Once the intrinsic point was reached, the resistance decreases exponentially, and dissipated power increases. This melted the heater and electrodes, which implies a minimum temperature of 1600 K. Thus, the importance of appropriate power control hardware is emphasized.

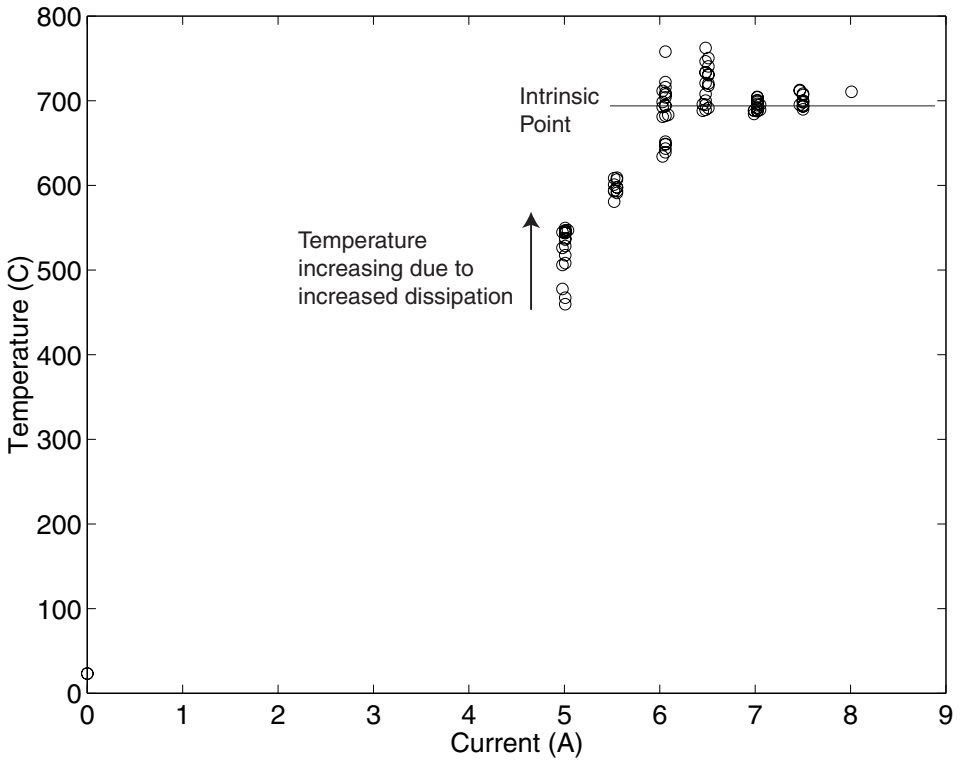


Figure 6.16: Heater temperature as a function of applied current. Once temperature has reached stability, increases in current do not increase temperature.

6.5.2 Vacuum Chamber Thrust Tests

The newly built thrust stand at MIT is identical to that used in the previous test with the added feature of velocity damping, which greatly improves the accuracy of the measurements. Though it is sensitive to 0.1 mN, drift in the zero point after calibration or during firing, results in an uncertainty of ± 0.5 mN. In addition, the calibration weights are in 3.8 mN increments as compared to the testing at Aerospace corporation, which was in 9.1 mN increments. This offers a better resolution of any nonlinearities in the calibration.

Mass flow and pressure are read from the same meters used in the benchtop testing in Chapter 5, and are located in the fluid line outside of the vacuum chamber. In addition, thermocouples are mounted on the chip, which is assumed isothermal, the electrical lead (approximately 10 cm from the chip), and the gas injection tubing. This instrumentation will allow the fluid temperature and the thermal losses to be estimated.

Figure 6.17 shows the thruster nozzle exit during heater operation. The manifold is integrated with a solenoid valve such that the flexible supply line can be charged to a given pressure right up to injection into the manifold. This allows the change in line flexibility with pressure to be accounted for in the calibration.

The vacuum chamber is approximately 30 m^3 and is pumped down with roughing pumps to 50 mTorr. Cryo pumps are available for achieving 10^{-6} Torr, but the roughing pumps are sufficient for the thruster testing.

The testing begins by calibrating the thrust stand with the thruster off, but the line charged to the specified pressure. The calibration samples ten points at each weight setting, and averages them. The thruster is then run while it is cold and the thrust response is averaged over 5 minutes. With the damping and the pressure control available from the regulator, the steady state is reached by the first data point after actuation (< 5 seconds).

Once the cold flow is run, power is supplied by a current source in 0.25 Amp increments. Each current setting is run for 7 minutes to obtain an average and allow transients to settle. The chip temperature, in general, over shoots the steady-state for a given current setting, but equilibrates within 30 seconds.

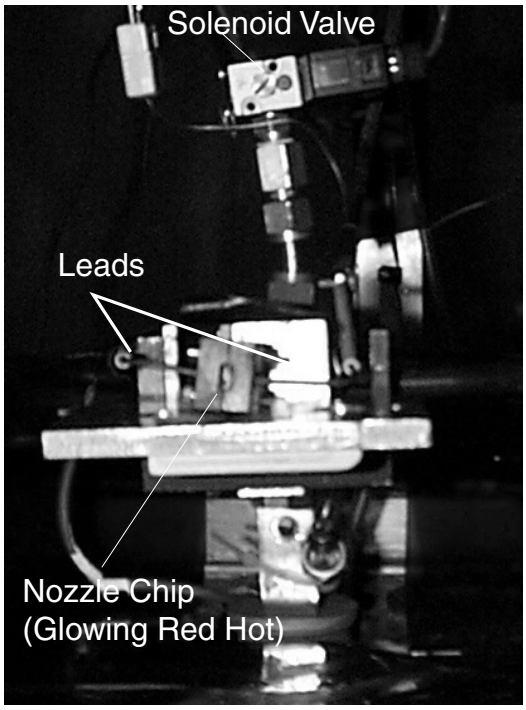


Figure 6.17: View of the nozzle opening during heater operation. The thermal deformation of the leads has caused the ceramic package to rise off of the support. This could potentially lead to frit failure.

The greatest difficulty with the gas flow test was the delicate nature of the Kovar-silicon bond. Thus, it became readily apparent that the ceramic clamp described earlier was not ideal. Since the gas injection tube is clamped to the manifold, and the thruster is suspended by the bond to the cantalevered tubing, the ceramic package must be adequately supported as to not stress the bond. However, the thermal expansion of the electrical leads, ceramic package, and gas tubing in some cases lead to cracking of the seal, but in most cases worked properly. For this reason, the thruster was only run to 420°C for the test results presented here. An earlier run was taken to the full 700°C, but began leaking after two thermal cycles. In order to get multiple data sets, this was limited to the lower temperatures to capture the performance data, and reduce thermal fatigue.

6.5.3 Thrust Test Results

A steady state average is established for each operating condition, and is depicted in dimensional form in figure 6.18. The tests are run on an 8.25:1 expansion ratio nozzle with a throat width of 65 microns, and a nozzle depth of 491 microns. The plenum contains 10 fins of 125 microns in length and 50 micron spacing. For the cold results, the thrust is reduced by reducing the operating pressure. The results at specified pressures are achieved by increasing the power dissipation, and subsequently the heater temperature. Ideally, the thrust is only a function of pressure. Ideal thrust is computed as:

$$T = \rho u_e^2 A_e + P_e A_e = \frac{P}{RT}\gamma R T M_e^2 A_e + P_e A_e \quad . \tag{6.29}$$

Since temperature cancels out of the momentum component, the ideal thrust should not vary with chamber temperature. However, as temperature increase, the Reynolds number decreases, and the exit Mach number decreases due to the boundary layer blockage effects described in Chapter 5. For this reason,

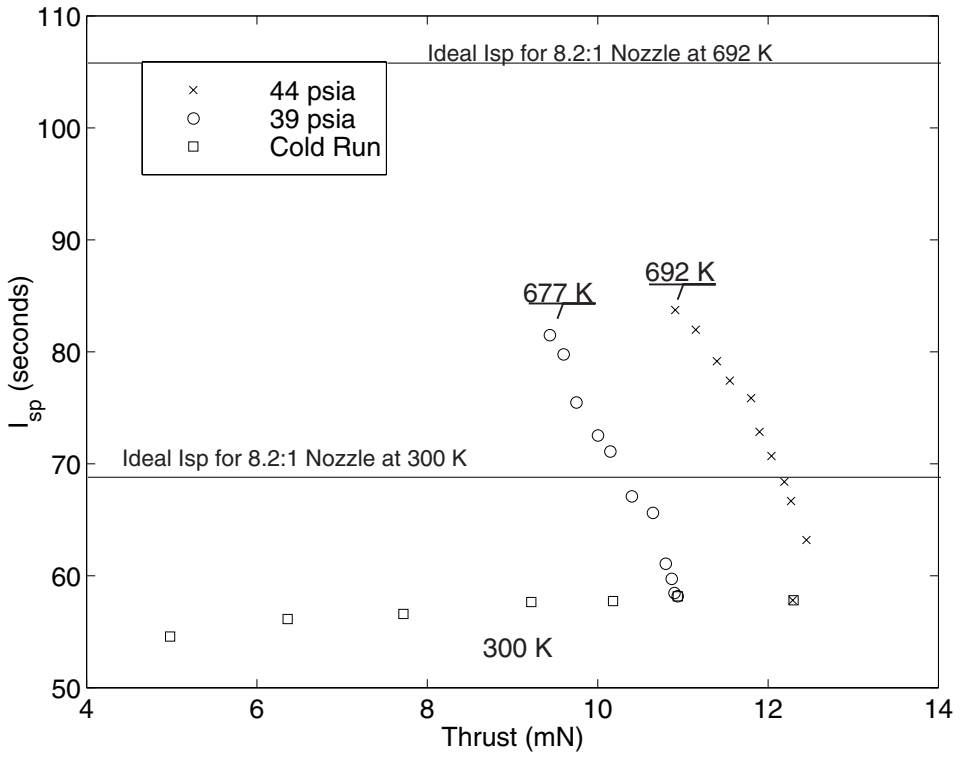


Figure 6.18: Performance of the thruster at various thrust levels for both heated and unheated devices. As temperature increases, thrust should remain constant and I_{sp} will increase. However, increases in temperature reduce the Reynolds number and cause the thrust to decrease.

the thrust reduces with temperature, and even more importantly, the I_{sp} relative to the theoretical also decreases significantly. This causes the deviation from vertical, which would be the ideal for this case.

The cold thrust results reduce with chamber pressure at a much slower rate than previously demonstrated, due to the improved thrust results, and increased exit-plane aspect ratio of 1.01.

The temperature data from the cold runs answers an outstanding question from the numerical analysis. That is, the silicon remains at the supply temperature during the thruster operation. This justifies the isothermal wall assumption. As the gas expands through the nozzle, the temperature drops and heat is transferred from the wall to the flow. This will cause the chip to lose energy and lower the wall temperature. However, the amount of energy lost from the chip is small compared to the heat capacity of the silicon and support package. Therefore nozzle is receiving enough heat from the external structure to maintain the chip temperature constant for the duration of the testing.

Figure 6.18 is evidence that viscous effects are causing the thrust and I_{sp} performance to deviate from the theoretical. Therefore, if we normalize the data by the Reynolds number as previously shown in Chapter 5, the viscous effects become apparent. Figure 6.19 shows the mass flow efficiency for the Reynolds numbers at the two operating pressures tested. The Reynolds number is still defined as before in equation (2.4), but the properties (such as viscosity) are allowed to vary with temperature. Since there is similarity occurring within the nozzles, the efficiency should be consistent with Reynolds number for different pressures as well as temperature. Figure 6.19 shows that this is, in fact, the case for a comparison with the previous cold runs and the current hot runs. Since the boundary layers develop at the same rate for the same Reynolds numbers, the efficiency decreases with Reynolds in the same fashion. There is some deviation at the lowest Reynolds numbers between the cold run and the hot runs. This is on the order of 5%. The accuracy of the measurements is 2% at this flow rate. Also, the accuracy of the temperature information would create a 100 point uncertainty in the Reynolds number as well as an additional 2% change in the C_d. Therefore, the difference in the hot and cold data is within the uncertainty of the measurements, and the different nozzles

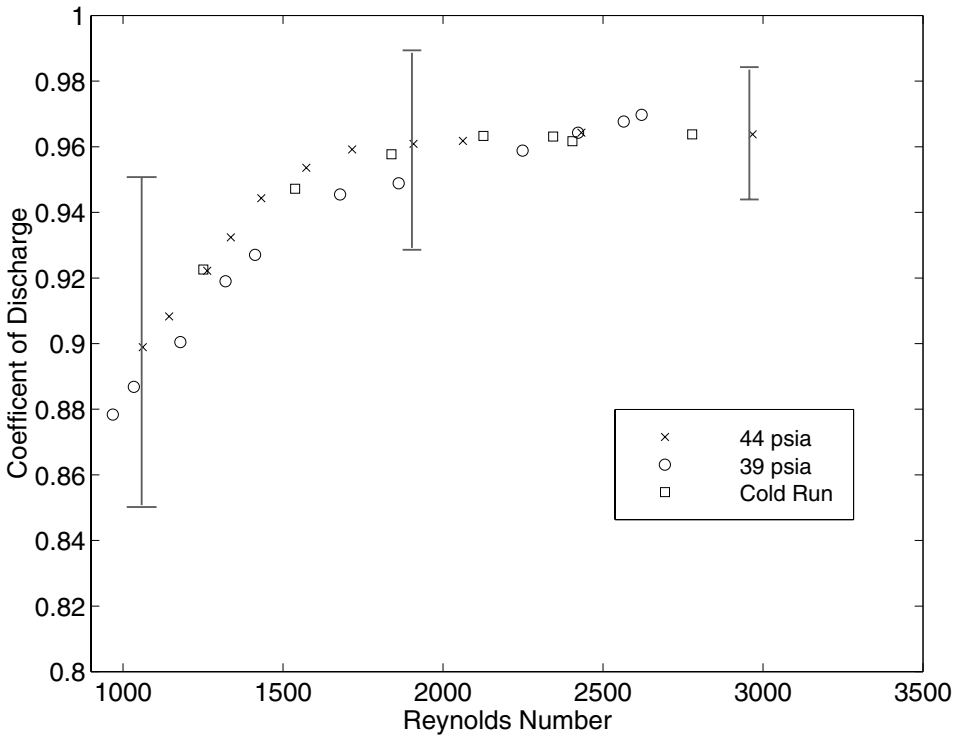


Figure 6.19: Coefficient of Discharge variation with Reynolds number for heated cases in comparison to previous experimental runs of the old nozzle design which are unheated.

compare well when normalized by the similarity parameter.

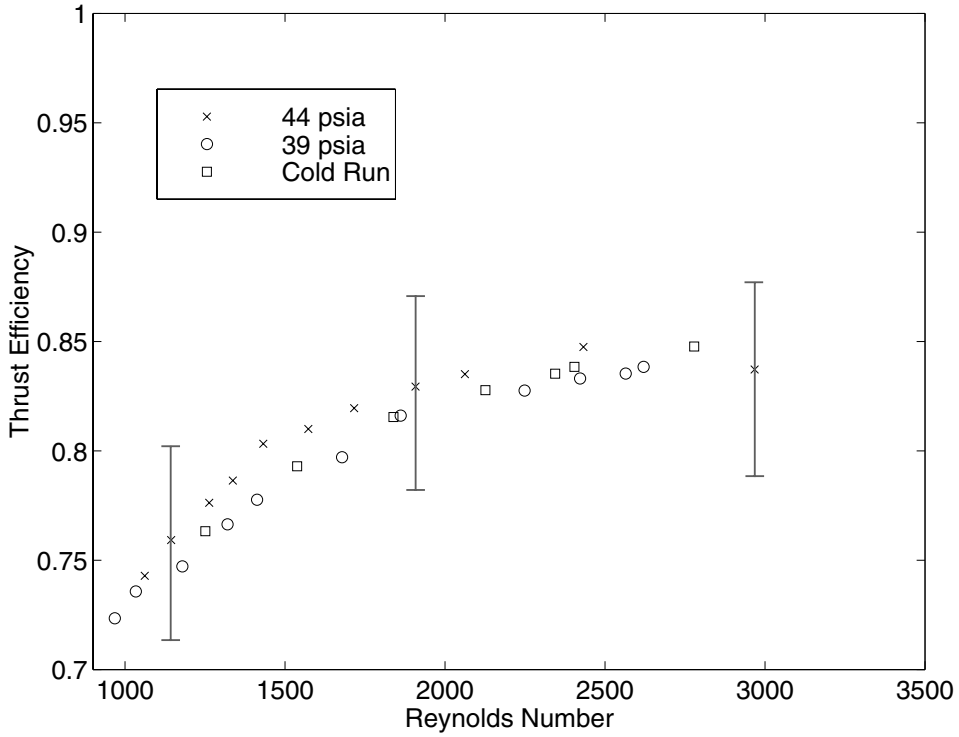


Figure 6.20: Thrust Efficiency variation with Reynolds number for heated cases in comparison to previous experimental runs of the old nozzle design which are unheated

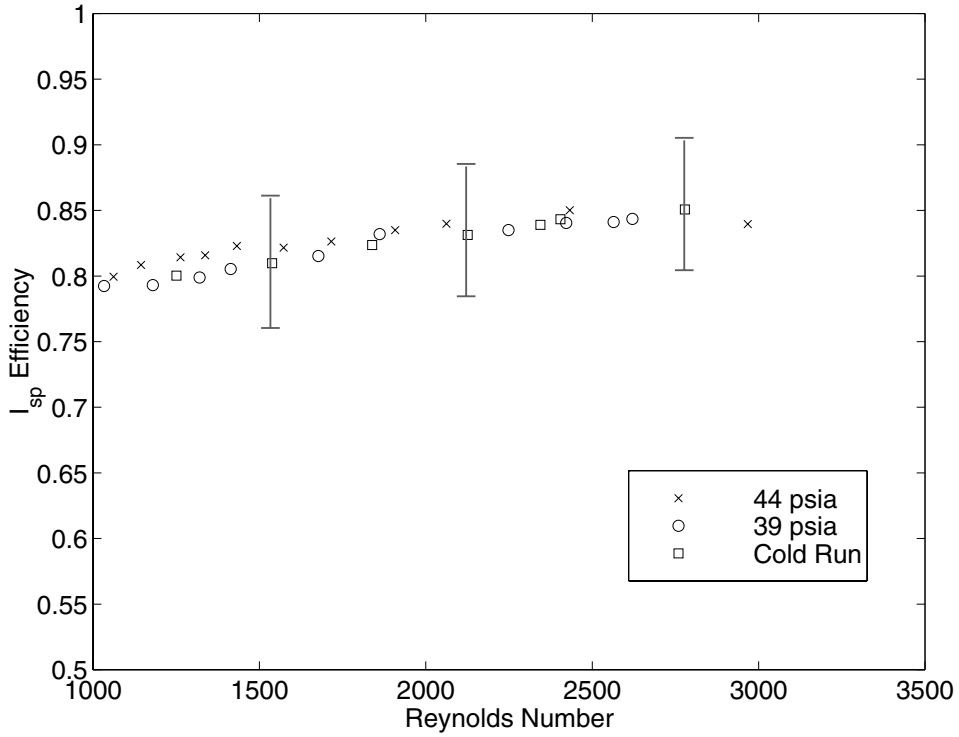


Figure 6.21: I_{sp} Efficiency variation with Reynolds number for heated cases in comparison to previous experimental runs of the old nozzle design which are unheated. This, along with figures 6.19 and 6.20, verify that Reynolds number is the governing similarity parameter.

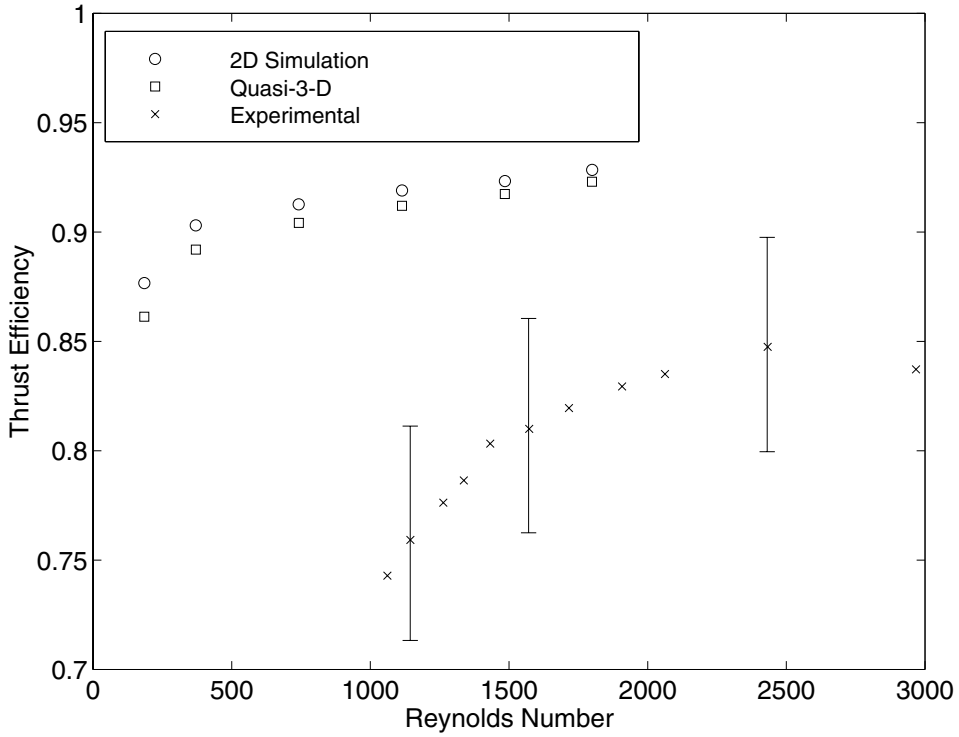


Figure 6.22: Thrust Efficiency variation with Reynolds Number for the 8.26:1 nozzle geometry. This is compared to a 2-D CFD simulation with endwall corrections. Maximum overprediction is 13%

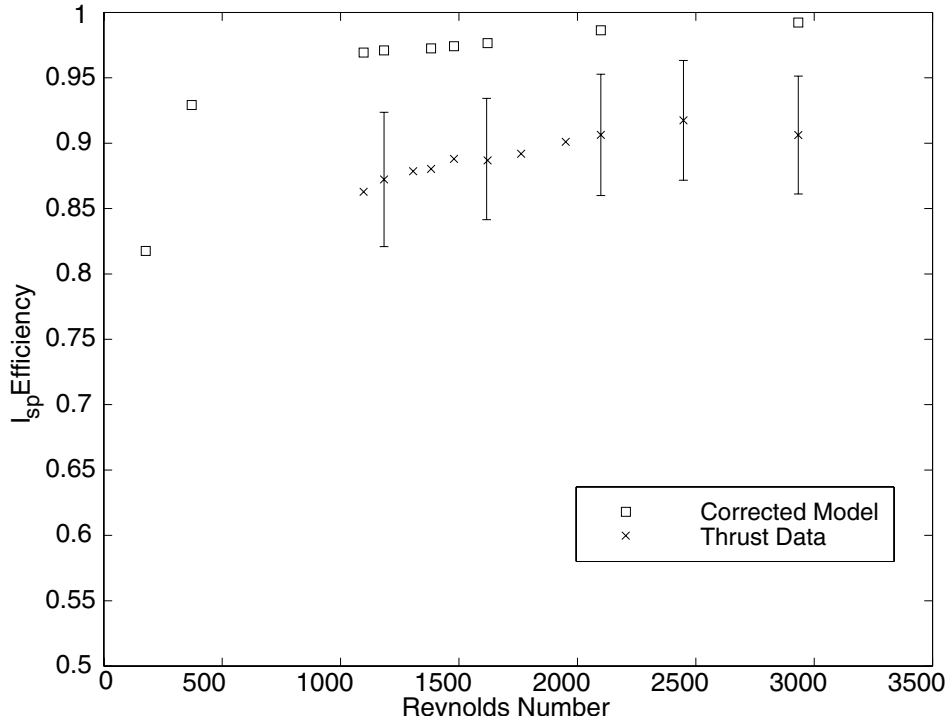


Figure 6.23: I_{sp} Efficiency variation with Reynolds Number for the 8.26:1 nozzle geometry. This is compared to a 2-D CFD simulation with endwall corrections. The maximum overprediction is 13%

The consistency between hot and cold performance parameters is again evident in figure 6.20, which depicts the thrust efficiency variation with Reynolds number. The uncertainty in the measurement is ±5%. Once again, the uncertainty in the temperature leads to an uncertainty in the Reynolds number that could account for the variation in the data. Similarly, this trend is reflected in the I_{sp} efficiency found in figure 6.21. Both of these figures are also evidence that the efficiencies do not drop off nearly as fast as found in the first set of runs of the cold nozzle in Chapter 5, due to the improved thrust measurements and larger exit-plane aspect ratio. The thrust and I_{sp} efficiencies for the 8.26:1 area ratio nozzle are compared to the 2-D CFD simulations with endwall corrections in figures 6.22-6.23. The maximum overprediction of the thrust and I_{sp} efficiencies is 13%. Though the predictions are outside the uncertainty of the measurement, it is understood that the corrections merely provide a first approximation of the endwall boundary layers. In addition, there are breakdowns in the assumptions of equilibrium flow and exit-plane extrapolation that can lead to significant frozen and flow divergence losses.

6.5.4 Overall Heated Thruster Assessment

The fluid flow in the heated thruster performs as expected. The efficiencies scale with Reynolds number as they did with the cold-gas tests. In dimensional numbers, the I_{sp} was continually increasing with chamber temperature despite the reduction in I_{sp} efficiency.

Though the propulsion system operates as expected at a given temperature, the amount of power required to reach that temperature is significantly higher than predicted. This is to be expected since the initial modeling was performed without thermal losses. Initial estimates for parasitic losses in section 6.3.4 assumed that the heat lost by radiation and conduction would occur on a surface area the size of the die. As it turns out, the ceramic package becomes much hotter than initially expected and loses heat due to conduction to the thrust stand, as well as increased radiation losses. Figure 6.24 depicts the variation of power input to achieve thruster temperatures for both runs. The amount of energy required to heat the fluid to the operating temperature is also indicated on the graph. The difference between this calculation and the actual power is the loss at a given temperature. These losses are within 1.8 Watts of the predicted losses at the highest power setting, which is within the uncertainty of the measurements. This indicates that all of the heat loss is accounted for, and is dominated by the conduction through the power leads (13.6 Watts at the highest power).

The propulsive efficiency is used to evaluate the overall performance of the device. It is computed as

$$\eta_{prop} = \frac{\frac{1}{2}\frac{T^2}{\dot{m}}}{\frac{1}{2}\frac{T_{ideal}^2}{\dot{m}_{ideal}} + P_{input}} \tag{6.30}$$

where the thrust squared divided by $\dot{m}$ represents the thrust power. The propulsive efficiency, therefore, represents the efficiency of the conversion of electrical power and stored energy as pressure to thrust power. Figure 6.25 illustrates the variation of this efficiency with Reynolds number. Since Reynolds number is reduced by increasing temperature in this case, the thermal gradients and losses are low at high Reynolds numbers. In this region, the efficiency is dominated by the fluid losses due to viscosity. As temperature is increased, and subsequently the Reynolds number decreases, the thermal losses become larger, thus lowering the overall efficiency. There is also an increase in the viscous losses at the lower Reynolds number which accounts for about 40% of the inefficiency, whereas the remainder is due to thermal losses.

A majority of the thermal loss is into the leads. There is a lead length which offers the optimal trade-off between power dissipated in the lead, and heat conducted along the lead, and is derived in the notes by Martinez-Sanchez [50]. It was not possible to operate at this length because of the constraints of the vacuum chamber, and thrust stand. If the length is optimized, an improved performance, plotted as least losses in figure 6.25, can be achieved.

Based on the raw performance numbers, as well as the efficiencies, it is possible to assess whether or not it is worth implementing electrothermal augmentation at this scale. From the performance numbers, the I_{sp} was continually increasing with thruster temperature. In addition, the thrust was well within the desirable range for microsatellite applications. However, the power consumption is still high. Without losses, 5.8 Watts is required to run at the intrinsic temperature, which is near the total power produced by a microspacecraft.

However, the thrusters are run in short bursts, and the nozzles come up to temperature quickly. Thus, this power is used for only a limited amount of time, and would allow this power requirement to be met on an intermittent basis. With a proper operations plan, the thrusters could easily be incorporated into a microsatellite for station keeping, but probably not for attitude control unless it is a low duty cycle. One such relationship often used by mission planners is the variation in efficiency with I_{sp} as show in figure 6.26. This shows the price in power, in terms of efficiency, for operating at a higher I_{sp}.

The losses can be considerably reduced by fabricating the device with the capability of regeneratively cooling the thruster with the propellant, and bringing the outer wall temperature down. However, the losses are dominated by the conduction through the leads, which can be significantly reduced by implementing the optimal lead length. Both of these oculd be accomplished if the gas lead was the electrical lead. The conduction losses could drop by 98% for the optimal length.

With a proper package and electrical interfaces, the efficiency at the lowest Reynolds numbers could be increased to above 40%. This would require the device be fabricated with a smaller throat to lower the mass flow (and power) for a given thrust. In addition, the optimal lead length as well as smaller surface area for the nozzle should be used. A device with 40% efficiency with an I_{sp} of over 100 seconds would offer a significant advantage over the cold microthruster or a conventionally machined device of similar thrust.

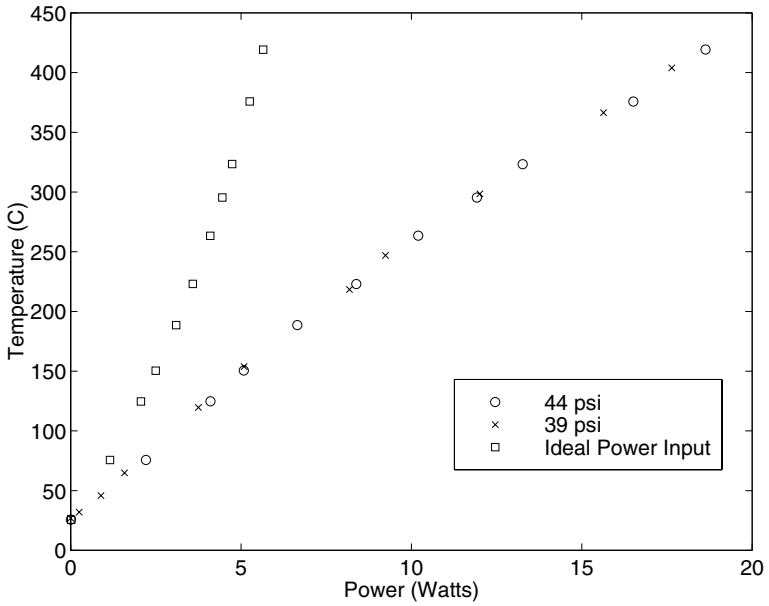


Figure 6.24: Dissipated power for a range of thruster temperatures. The square points depict the ideal power input for the flow to reach a given temperature

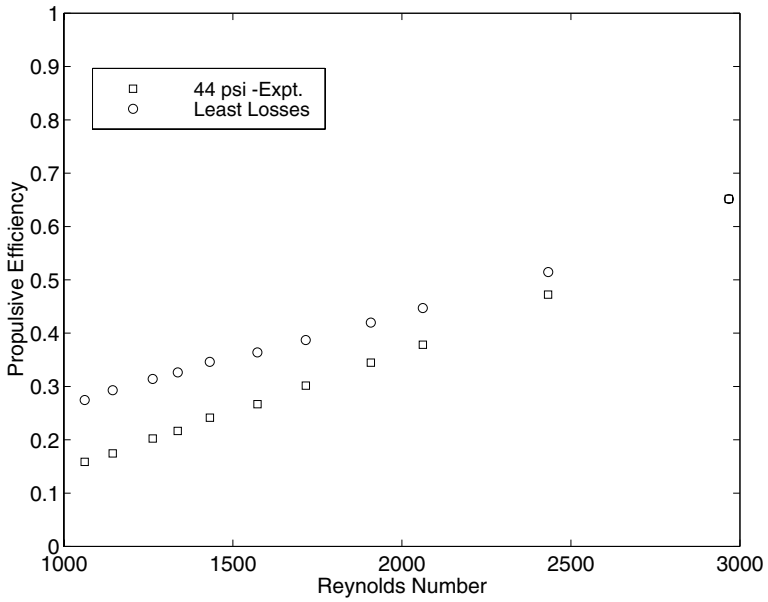


Figure 6.25: Variation of propulsive efficiency with Reynolds for the 44 psia chamber pressure. If the lead length is optimized for least power losses, as well the conduction into the support, the least losses curve could be achieved.

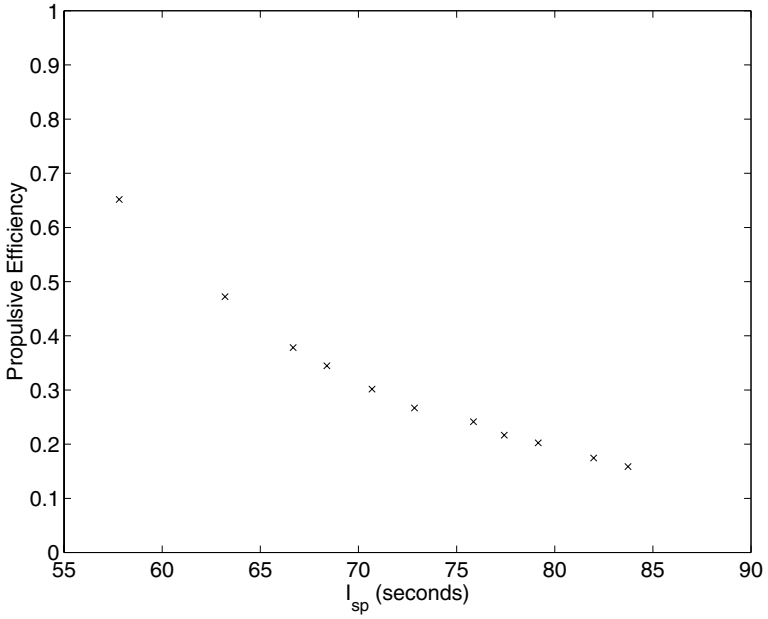


Figure 6.26: Variation of propulsive efficiency with I_{sp}. This provides mission designers with an indication of the power versus propellant trade-off of the thruster.

6.6 Summary

By increasing the chamber energy, the amount of mass flow required to achieve a thrust level is reduced. This decreases the amount of propellant required during a mission, and subsequently reduces the satellite mass. The following is a summary of the analysis and testing reported in this chapter:

- By increasing the heat transfer to the fluid through fins present in the chamber, device length and pressure drop in the chamber can be minimized.
- By running the entire chip at the operating temperature, fabrication is simplified with very little loss in performance.
- The optimal nozzle geometry of interest is a bank of 10 fins spaced 50 microns apart, and 125 microns long for the benchmarked flow rate of 350 sccms.
- Heater test demonstrate stable operation at the intrinsic point when powered in constant current mode.
- Thrust results demonstrate an I_{sp} of 93 seconds for a wall temperature of 419°C. The 18.6 watts of power is required to reach these conditions. This results in a propulsive efficiency of 17%.
- Thrust and I_{sp} efficiency exhibit the same variation with Reynolds number as the computational simulations. The simulations, however, underpredict boundary layer thicknesses, and subsequently overpredict the efficiencies by 13%.

Chapter 7

Conclusions and Recommendations

From the outset of this research, the task has been to demonstrate the miniaturization of the propulsion system by incorporating Microelectromechanical systems. Components that were amenable to microfabrication were identified. This was accomplished by establishing the capabilities of microfabrication technologies, and assessing the operating parameters required to achieve the optimum performance. Since the performance of a propulsion system is a function of the momentum exchange of the pressurized gas, the nozzle was chosen as the focus of this work, in order to assess the capabilities of a truly micropropulsion system.

This work has confirmed the performance variation of the nozzles is strictly a function of the Reynolds number. As shown analytically, the operating Reynolds number for a given thrust can be increased by fabricating the throat as small as possible, and operating at a higher pressure. This lowers thrust, while reducing viscous losses. The Reynolds number can be used to gauge the blockage created by the boundary layers, and the subsequent reduction of effective area ratio of the nozzle due this blockage. This work has established that by operating the nozzles at Reynolds numbers above 1500, thrust and I_{sp} efficiencies above 80% can be achieved. Below this number, viscous effects rapidly increase, driving the performance down. In addition, the low operating pressure results in high relaxation times preventing the gas from obtaining full equilibrium in the flow.

By optimizing the operating point for the deep reactive ion etching of silicon, nozzles of variable geometry and minimum features of 18 microns were created. Flexibility in the definition of the nozzle geometry allows an arbitrary expansion ratio to be set, and the fabrication of contoured features to guard against separation. The first generation nozzles were limited to exit-plane aspect ratios of less than 1. This greatly increased the influence of the endwall boundary layers on performance, due to the blockage they create. At the lowest Reynolds numbers, the effective area ratio was reduced to less than one. The most recent nozzle designs have larger aspect ratios, and the influence of endwall blockage is greatly reduced.

The testing has demonstrated a cold gas I_{sp} of 66 seconds at a thrust level of 10 mN. This is higher than previous microdevices, or conventionally-machined cold-gas propulsion systems of this thrust level. By simultaneously etching fins into the plenum during fabrication, which serve as resistors to dissipate heat, an I_{sp} of 83 seconds at a thrust level of 12 mN was demonstrated for 420°C chamber temperature. Though the thermal losses were substantial, the fluid performance is as expected, and can be improved with better insulation methods, most notably, the reduction in lead length. The MEMS community is currently lacking fast-acting microvalves capable of high pressure sealing. However, standard small valves have actuation times of a few milliseconds [2]. This would create an I_{bit} of the order of $30\mu Ns$. Initial studies at JPL indicate that $I_{bit} < 15\mu Ns$ for control authority of less than $10^{-5} rad/sec$.

7.1 Impact on Propulsion Technologies

With the demonstrated performance enhancements over current technologies, this work serves as a driver for further microcomponent development. It shows that the nozzle-plenum system can be fabricated and operated in a device less than 10 grams. By operating the devices for hours without failure, we have dispelled the concerns of failure due to contamination. All of this should serve as a motivation for the development

of microvalves to create a truly integrated propulsion system. In addition, benchtop tests demonstrated 700°C temperatures are achievable, and would permit an I_{sp} of 110 seconds after viscous losses. This should encourage further investigation into on-chip insulation to reduce thermal losses.

7.2 Impact on MEMS

This work significantly contributed to the improvement of DRIE performance through recipe optimization. A variable geometry flow feature was fabricated with high anisotropy. It also established a method for creating a bulk-micromachined heater for enhancing heat transfer to a flow, and simultaneously providing a stable operating temperature at the intrinsic point. This will prevent thermal runaway from occurring in gas systems, where the viscosity increases with temperature.

7.3 Recommendations for Future Work

Since it has been established that the performance of these systems is dominated by viscous effects, and that they can be mitigated by operating at a higher Reynolds number, the nozzle throat area should be reduced further to achieve higher performance at the lower thrust levels tested. This can be accomplished with the recent improvements of DRIE and photolithography, which maintain a 10 micron trench to 400 micron etch depth. This would double the operating Reynolds number.

In addition, the boundary layer development can be reduced by decreasing the nozzle length. By relying upon the loading effect of the etcher, nozzles could be created with shallower throats, and diverging exit sections, lending an element of three dimensionality to the etch. This will serve to reduce the throat area (hence driving up the Reynolds number for a given thrust), and increasing the expansion ratio for a given length. Both of these schemes will decrease the viscous effects significantly, and allow a cold I_{sp} of greater than 60 seconds for thrusts of 1 mN and possibly less.

It is realized that the devices developed in this work can be further optimized to improve performance. Many suggestions have been mentioned throughout this work, and are summarized here.

- As mentioned above, the nozzle throat can be reduced through a proper implementation of the loading effect to achieve a 3-dimensional nozzle expansion. This will also reduce the boundary layer development length for a given area ratio.
- The microresistojet could use the gas lead as the electrical lead to minimize conduction losses as well as regeneratively heat the propellant.
- If the thruster is run a pulsed fashion, on chip insulation may be beneficial. This would reduce heat losses locally.
- Larger amounts silicon in the thruster plane can be removed to lower the thermal transient. This could be replaced with a honeycomb pattern to improve integrity while minimizing mass and forcing the electrical current path through the fins.
- A lower molecular weight propellant could be implemented to improve specific impulse. A higher density liquid propellant could be used to make this a vaporizing liquid microthruster.

Appendix A

Non-Dimensionalization

The Finite-Volume Navier-Stokes simulation utilizes non-dimensional parameters for its computations. All parameters are non-dimensionalized by the the throat (subscript t) conditions. These variables are defined by:

$$x = \frac{x^*}{D_t^*};\ y = \frac{y^*}{D_t^*};\ u = \frac{u^*}{a_t^*};\ v = \frac{v^*}{a_t^*};\ t = \frac{t^*}{\frac{D_t^*}{a_t^*}} \tag{A.1}$$

$$\rho = \frac{\rho^*}{\rho_t^*};\ p = \frac{p^*}{\rho^* a_t^{2^*}};\ e = \frac{e*}{a_t^{2^*}};\ Re = \frac{\rho^* a_t^* D_t^*}{\mu_t^*};\ Pr = \frac{{c_{p_t}}^* \mu_t^*}{k_t^*} \tag{A.2}$$

where D_t is the throat width and a_t is the speed of sound. All properties with an asterisk are dimensional quantities. All state variable values are based on the isentropic definition at Mach 1. The following are relations of total conditions in the plenum to the sonic condition:

$$\frac{P}{P_o} = 0.5283;\ \frac{\rho}{\rho_o} = 0.6339;\ \frac{T}{T_o} = 0.8333; \tag{A.3}$$

Appendix B

Code Listing

The following pages are a listing of ns.c, a C-program used to simulate the viscous flow through a converging-diverging nozzle. This version uses extrapolation to determine the boundary values throughout the entire exit plane, though the hooks are still in place to specify exit pressure in the subsonic portion of the boundary layer.

In order to run the code from an initial guess, an input file containing the grid x and y dimensions, number of iterations, the operating Reynolds number, chamber pressure, temperature, and throat diameter needs to be created. These values specify the operating conditions and the dimensional parameters that are used to vary the gas properties with temperature and pressure. In addition to the input file, two grid files of the x and y grid point locations for each point in row i and column j.

The code can be run from the initial condition of Quasi 1-D flow with a parabolic patch to wall conditions, or from a previous solution by RESTART. All constants are specified in the initial define commands, such as artificial viscosity of both first and second order (AV4 and AV2), gas constants such as R, Molecular Weight (MW), and Prandtl Number (PR), and code operation parameters, like the time step safety factor.

The code is controlled through Main, and following subroutines perform the specified operations:

Init Vars: Initializes the state vector for each grid point
Calc I step: Computes initial time step for the artificial viscosity for first iteration
IFlux: Calculates the invisicid fluxes based on the state vector
VFlux: Calculates the viscous fluxes based on the state vector, including artificial viscosity
Cell Areas: Computes the area of each cell. This is used for weighting the residual for each timestep.
IFLUX: Computes intermediate value of state vector for each stage of the Runge-Kutta timestepping.
FUpdate: Corrects the state vector based on the final stage of the time step.

```
/*************************************************************************/
/*   Navier-Stokes Solver for 2-D Flows (MicroNozzles) - v1.5            */
/*   E.S. Piekos - K.S. Breuer- R.L. Bayt Spring 1996                    */
/*************************************************************************/

#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <malloc.h>

#define GMA      1.4    /* Ratio of specific heats */
#define GMA1     0.4    /* Gamma - 1 */
#define PR       0.72 /* Prandtl Number */
/*#define MU       1.0*/    /* Dimensionless viscosity coef */
#define TS_SF    1.5   /* Time step safety factor */
#define R_MAX    1e-9  /* Maximum residual at 'convergence' */
/*#define POUT     0.03*/  /* Exit Pressure */
#define R        8314.0/* Universal Gas  Constant */
#define MW       28.00 /* Molecular Weight */
#define AV4_COEF 0.001  /* Coefficient for 4th order artif. visc */
#define AV2_COEF -0.01 /* Coefficient for residual-dependent 2nd order AV */
#define RESTART  /* Restart code from a previous solution */
#define pi       3.14159265
#define thrustcalc
/*#define DOUBGRID */  /* Double number of radial and circumferential nodes*/
/*#define DOUBCNDS */     /* Double number of circumfential nodes only */
/*#define HALFGRID */  /* Halve number of radial and circumferential nodes */
/*#define HALFCNDS */   /* Halve number of circumferential nodes only */
/*#define PLANAR*/      /* Do a bearing with infinite radius */
/*#define ADIABATIC*/   /* Impose adiabatic walls instead of isothermal */
/*#define TEST*/
#define UNSTEADY
/*#define MOVING_GRID*/
#define ANG_OFST 0

#define float double

#define MAX(A,B) ((A)>(B)?(A):(B))
#define MIN(A,B) ((A)<(B)?(A):(B))
#define ABS(A)   ((A)>0?(A):(-1*(A)))

#define FDEBUG(A)  printf(#A": %f \n",A); fflush(NULL);
#define IDEBUG(A)  printf(#A": %d ",A); fflush(NULL);
#define FDEBUGQ(A) printf("%f ",A); fflush(NULL);
#define IDEBUGQ(A)/*printf("%d ",A); fflush(NULL);*/
#define SDEBUG(A) /*printf(#A); fflush(NULL);*/

struct node{
  float w[4],p;
};
struct flux{
  double w[4];
};
struct point{
  double x,y;
};
struct locale{
  double u,l;
};
struct a_cell{
  float visc,invs;
};

#ifndef MOVING_GRID
struct flxface{
  double dxe,dye,dxw,dyw,dxn,dyn,dxs,dys;
}**iflxface;
#endif

float jrad,bpress,residual=0,th=0;
int squit_flag=0;

void Load_Grid(float **x, float **y, int *n_cpts, int *n_rpts);
void Load_Vars(int n_r, int n_c, struct node **state, struct flux **iflux,
               struct flux **vflux);
void Dgrid_Interpolate(int n_r, int n_c, struct node **state,
                       struct flux **iflux, struct flux **vflux);
void Dump_Grid(int n_r, int n_c, struct point **grid);
float Init_Vars(int n_r, int n_c, double theta_step,  struct point
               **grid, struct node **state, struct flux **iflux,
               struct flux **vflux, float brad, float totp, float tott,
               float reyno, struct node *ff, struct locale *te,
               struct locale *ts, struct locale *ne, struct locale *ns);
void Calc_I_TStep(int n_r, int n_c, struct point **grid, struct node **state,
                  float **tstep);
void Calc_IFlux(int n_r, int n_c, struct point **grid, struct node **input,
                struct flux **sumflx, double theta_step,int wfact);
void Calc_VFlux(int n_r, int n_c, struct point **grid, struct node **input,
                struct flux **vflux, float Re, struct a_cell **area, float
                **tstep, int wfact, float tott);
void Calc_Areas(int n_r, int n_c, struct point **grid, struct a_cell **area);
void IUpdate(int n_r, int n_c, struct node **istate, struct node **state,
             struct flux **iflux, struct flux **vflux, struct a_cell **area,
             float **tstep, struct point **grid, double theta_step, int wfact,
             float reyno, struct node *ff, struct locale *te,
             struct locale *ts, struct locale *ne, struct locale *ns);
float FUpdate(int n_r, int n_c, struct node **state, struct flux **iflux,
              struct flux **vflux, struct a_cell **area, float **tstep, double
              theta_step, struct point **grid, float reyno, struct node *ff,
              struct locale *te, struct locale *ts, struct locale *ne,
              struct locale *ns, float resm);
void Dump_Data(int n_r, int n_c, struct node **state, float totp, float tott,
               float thrD, struct point **grid);
void Dump_Data2(int n_c, int n_r, struct node **state, float t);
float arat(float M, float eps);
float thrust(int n_r, int n_c, struct node **state, struct point **grid,
             float tott, float totp, float thrtD, float invthr, float Mie);
float mdot(int n_r, int n_c, struct node **state, struct point **grid,
           int pnt);
void thick(int n_c, int n_r, struct node **state, struct point **grid);


void Backdoor()
{
  squit_flag=1;
  signal(SIGQUIT,Backdoor);
}

/*********************    MAIN   ******************************/
main()
{
  int n_cnds, n_rnds, ir, it=0,ij,ii, tsteps,ic,iv,pnt;
  float b_rad=0.0,clrnc,reyno,fresx,fresy,oldresidual=8888,time=0,
  **tstep,cd1,cd2;
  float cd3,cd4,cd5,cd6,cd7,cd8,At,invthr, Mie;
  float tott,totp,thrtD, resm=0.0;
  double nd_anglr_sep=0.0;
  struct node **state, **istate, *ff;
  struct flux **iflux, **vflux;
```

```
struct point ecctry, **grid;
struct a_cell **area;
struct locale *te, *ts, *ne, *ns;
FILE *fpo, *fpt, *fps;

signal(SIGQUIT,Backdoor);

/**** Get grid density parameters ****/
fprintf(stderr," Number of Nodes in the x-direction: ");
scanf ("%d",&n_cnds);
fprintf(stderr," Number of Nodes in the y-direction: ");
scanf ("%d",&n_rnds);

/**** Get flow parameters ****/
fprintf(stderr," Timesteps: ");
scanf ("%d",&tsteps);
fprintf(stderr," Reynolds Number Based on Throat: ");
scanf ("%lf",&reyno);
fprintf(stderr," Plenum Pressure: ");
scanf ("%lg",&totp);
fprintf(stderr," Plenum Temperature: ");
scanf ("%lf",&tott);
fprintf(stderr," Throat D: ");
scanf ("%lg",&thrtD);
fprintf(stderr,"\n");


/**** Allocate memory for working arrays ****/


grid = (struct point **)malloc((size_t)(n_rnds*sizeof(struct point *)));
grid[0] = (struct point *)malloc((size_t)(n_rnds*n_cnds
                                  *sizeof(struct point)));
state = (struct node **)malloc((size_t)(n_rnds*sizeof(struct node *)));
state[0] = (struct node *)malloc((size_t)(n_rnds*n_cnds
                                  *sizeof(struct node)));
ff = (struct node *)malloc((size_t)(n_rnds*sizeof(struct node )));

istate = (struct node **)malloc((size_t)(n_rnds*sizeof(struct node *)));
istate[0] = (struct node *)malloc((size_t)(n_rnds*n_cnds
                                  *sizeof(struct node)));
iflux = (struct flux **)malloc((size_t)(n_rnds*sizeof(struct flux *)));
iflux[0] = (struct flux *)malloc((size_t)(n_rnds*n_cnds
                                  *sizeof(struct flux)));
vflux = (struct flux **)malloc((size_t)(n_rnds*sizeof(struct flux *)));
vflux[0] = (struct flux *)malloc((size_t)(n_rnds*n_cnds
                                  *sizeof(struct flux)));
area = (struct a_cell **)malloc((size_t)(n_rnds*sizeof(struct a_cell *)));
area[0] = (struct a_cell *)malloc((size_t)(n_rnds*n_cnds
                                  *sizeof(struct a_cell)));
tstep = (float **)malloc((size_t)(n_rnds*sizeof(float *)));
tstep[0] = (float *)malloc((size_t)(n_rnds*n_cnds*sizeof(float)));

te = (struct locale *)malloc((size_t)(n_cnds*sizeof(struct locale)));
ts = (struct locale *)malloc((size_t)(n_cnds*sizeof(struct locale)));

ne = (struct locale *)malloc((size_t)(n_cnds*sizeof(struct locale)));
ns = (struct locale *)malloc((size_t)(n_cnds*sizeof(struct locale)));

for (ir=1; ir<n_rnds; ir++){
  grid[ir] = grid[ir-1] + n_cnds;
  state[ir] = state[ir-1] + n_cnds;
  istate[ir]= istate[ir-1]+ n_cnds;
  iflux[ir] = iflux[ir-1] + n_cnds;
  vflux[ir] = vflux[ir-1] + n_cnds;
    area[ir]  = area[ir-1]  + n_cnds;
    tstep[ir] = tstep[ir-1] + n_cnds;
  }

#ifndef MOVING_GRID
  iflxface = (struct flxface **)malloc((size_t)(n_rnds*sizeof
                                                (struct flxface *)));
  iflxface[0] = (struct flxface *)malloc((size_t)(n_rnds*n_cnds
    *sizeof(struct flxface)));
  for (ir=1; ir<n_rnds; ir++)
    iflxface[ir] = iflxface[ir-1] + n_cnds;
#endif

  fpo  = fopen("x.dat","r");

  for(ij=0;ij< n_rnds; ij++){
    for(ii=0; ii< n_cnds; ii++) {
      fscanf(fpo, "%lf", &grid[ij][ii].x);
    }
  }

  fclose(fpo);


  fpt=fopen("y.dat","r");
for(ij=0;ij<n_rnds;ij++) {
  for(ii=0;ii<n_cnds;ii++) {
    fscanf(fpt, "%lf", &grid[ij][ii].y);

    }

}

fclose(fpt);

  printf("\n Grid Read Complete %i %i\n",n_cnds,n_rnds);

#ifdef RESTART
  fps  = fopen("time.dat","r");

  do{

#ifdef UNSTEADY
    if(!(feof(fps))){
      fscanf(fps, "%i %lg %lg %lg %lg %lg",&it,&tstep[0][0],&time,
             &residual, &cd1, &cd2);
    }

#else
    if(!(feof(fps))){
      fscanf(fps, "%i",&it);
      fscanf(fps, "%e",&residual);
    }
#endif
  }while(pnt=getc(fps) !=EOF);


  fclose(fps);
  tsteps =it+tsteps;
  fps  = fopen("time.dat","a");
  Mie=Init_Vars(n_rnds,n_cnds,nd_anglr_sep,grid,state,iflux, vflux, b_rad,
                totp, tott, reyno, ff, te, ts, ne, ns);
  At=(grid[n_rnds-1][n_cnds-1].y-grid[0][n_cnds-1].y);
  invthr=pow(state[n_rnds/2][n_cnds-1].w[1],2)/
    state[n_rnds/2][n_cnds-1].w[0]*At;
```

```
  Load_Vars(n_rnds,n_cnds,state,iflux,vflux);
#if defined DOUBGRID || defined DOUBCNDS
  Dgrid_Interpolate(n_rnds, n_cnds, state,iflux,vflux);
#endif

#else
 /* printf("here 1\n"); */
  fps  = fopen("time.dat","w");
  /*printf("here 2\n");*/
  Mie=Init_Vars(n_rnds,n_cnds,nd_anglr_sep,grid,state,iflux, vflux, b_rad,
                totp, tott, reyno, ff, te, ts, ne, ns);
  At=(grid[n_rnds-1][n_cnds-1].y-grid[0][n_cnds-1].y);
  invthr=pow(state[n_rnds/2][n_cnds-1].w[1],2)/
    state[n_rnds/2][n_cnds-1].w[0]*At;

#endif

Dump_Grid(n_rnds,n_cnds,grid);

if (tsteps == 0){
  Dump_Data(n_rnds,n_cnds,state,totp,tott,thrtD,grid);
  th=thrust(n_rnds,n_cnds,state,grid, tott, totp, thrtD, invthr, Mie);
  printf("The total thrust is %f \n",th);
  exit(0);
}



/**** Get an initial time step (needed for artificial viscosity ****/
Calc_I_TStep(n_rnds,n_cnds,grid,state,tstep);

/**** Calculate cell areas ****/
Calc_Areas(n_rnds,n_cnds,grid,area);

Dump_Data(n_rnds,n_cnds,state,totp,tott,thrtD,grid);
do{
  Calc_IFlux(n_rnds,n_cnds,grid,state,iflux,nd_anglr_sep, 4);

  Calc_VFlux(n_rnds,n_cnds,grid,state,vflux,reyno,area,tstep, 4, tott);


  IUpdate(n_rnds,n_cnds,istate,state,iflux,vflux,area,tstep,grid,
          nd_anglr_sep,4, reyno, ff, te, ts, ne, ns);

  Calc_IFlux(n_rnds,n_cnds,grid,istate,iflux,nd_anglr_sep, 3);

  IUpdate(n_rnds,n_cnds,istate,state,iflux,vflux,area,tstep,grid,
          nd_anglr_sep,3, reyno, ff, te, ts, ne, ns);

  Calc_IFlux(n_rnds,n_cnds,grid,istate,iflux,nd_anglr_sep, 2);

  IUpdate(n_rnds,n_cnds,istate,state,iflux,vflux,area,tstep,grid,
            nd_anglr_sep,2, reyno, ff, te, ts, ne, ns);

  Calc_IFlux(n_rnds,n_cnds,grid,istate,iflux,nd_anglr_sep, 1);

  residual=FUpdate(n_rnds,n_cnds,state,iflux,vflux,area,tstep,
                   nd_anglr_sep,grid, reyno, ff, te, ts, ne, ns, resm);
  it++;

#ifdef UNSTEADY
    time += tstep[0][0];
    if(!(it%500)){
        cd1=mdot(n_rnds,n_cnds,state,grid,10);
        cd2=mdot(n_rnds,n_cnds,state,grid,30);
        cd3=mdot(n_rnds,n_cnds,state,grid,55);
        cd4=mdot(n_rnds,n_cnds,state,grid,75);
        cd5=mdot(n_rnds,n_cnds,state,grid,100);
        cd6=mdot(n_rnds,n_cnds,state,grid,155);
        cd7=mdot(n_rnds,n_cnds,state,grid,225);
        cd8=mdot(n_rnds,n_cnds,state,grid,n_cnds-1);

       /* cd1=mdot(n_rnds,n_cnds,state,grid,56);
        cd2=mdot(n_rnds,n_cnds,state,grid,n_cnds-1);
        cd3=mdot(n_rnds,n_cnds,state,grid,40);
        cd4=mdot(n_rnds,n_cnds,state,grid,60);
        cd5=mdot(n_rnds,n_cnds,state,grid,100);
        cd6=mdot(n_rnds,n_cnds,state,grid,120);*/


        printf("%e %f %f %f %f %f %f %f \n ",residual,cd2,cd3,cd4,cd5,cd6,cd7,cd8);

        fprintf(fps,"%d %e %e %e %e %e\n",it,tstep[0][0],time,residual,cd1,cd2);
        fflush(stdout);
      }

     if(!(it%1000))
       Dump_Data(n_rnds,n_cnds,state,totp,tott,thrtD,grid);
#else

       if(!(it%10))
       fprintf(fps,"%d\t%e\n",it,residual);


#endif

    /**** Dump data if a SIGQUIT (ctrl-\) is sent ****/
       if (squit_flag){
         Dump_Data(n_rnds,n_cnds,state,totp,tott,thrtD,grid);
         close(fps);
         exit(0);
#ifdef UNSTEADY
         printf("%d\t%f\t%f\t%f\n",it,tstep[0][0],time,residual);
#else
         printf("%d\t%f\n",it,residual);
         fflush(stdout);
#endif
       squit_flag = 0;
       }


  } while(it < tsteps);
/*while(residual==residual && residual>R_MAX); */
 printf("%d\t%e\n",it,residual);
if(residual==residual)
  Dump_Data(n_rnds,n_cnds,state,totp,tott,thrtD,grid);

   th=thrust(n_rnds,n_cnds,state,grid,tott,totp,thrtD,invthr,Mie);
   thick(n_cnds,n_rnds,state,grid);
  /* printf("The total axial thrust is %f and the mass residual is %e \n"
          ,th,resm); */
   printf("Reynolds number: %f\n",reyno);
   fclose(fps);

}
```

```
/*********************   END MAIN ***************************/


/********************    Initialize Variables   ***************************/

float Init_Vars(int n_r, int n_c, double theta_step,
                struct point **grid, struct node **state, struct flux **iflux,
                struct flux **vflux, float brad, float totp, float tott,
                float Re, struct node *ff, struct locale *te, struct locale *ts,
                 struct locale *ne, struct locale *ns)
{
  int ir, ic, iv,bl1,bl2, icp1,icm1, thra,bub;
  float e_inf, *st,At,anew,Ain,Ae,area,Tstat[260][260],p0,rho0,cv,M1,M2,M,
    savem,a,b,c,d,c1,c2,c3,c4,c5,c6,c7,c8[260],c9,c10,p1,p2,g,eps,qin;
  float error,T0,a0,u0,ptp1,TtT1,rtr1,ewall,pt1,pt2,pr1,pr2,xt,xin,Mout,entpnt;
  float ds,dtm,dtp,sumdt,xn,yt,yn,xs,ys,xe,ye,nt,ijac1,Dp,invthr,Min, alph;
  double *iflx, *vflx,Mie;

  /* Initialize throat ratios */
     /* printf("here 3\n"); */

     ptp1=pow(2.0/(GMA+1),GMA/(GMA1));
     TtT1=2.0/(GMA+1.0);
     rtr1=1.0/pow((GMA+1)/2.0,1/(GMA1));
     ewall=1.0/(GMA*GMA1*TtT1);

  /*  Find the throat in xi-eta */

  /* Nozzle Flow B.C.'s */
 /*printf("here 4\n");*/

  Dp=grid[n_r-1][0].y-grid[0][0].y;
  At=9999.0;
  xt=0.0;
  for(ic=0; ic<n_c; ic++){
    anew=grid[n_r-1][ic].y-grid[0][ic].y;

    if( anew < At) {
      At=anew;
      xt=grid[0][ic].x/At;
      thra = ic ;
    }
  }
 /*printf("here 5 %f %f %i\n",At,xt,thra);*/
/* Boundary layer limits */
  bl1=n_r*0.2;
  bl2=n_r*0.8;

  bl1=(int)bl1;
  bl2=(int)bl2;
  /* Non-dimensionalize the Grid
     Throat D= At*/
    entpnt=grid[0][0].x;
  for(ir=0; ir<n_r; ir++)
    for(ic=0; ic<n_c; ic++){
          /*printf("%i %i %f %f ",ir,ic,grid[ir][ic].x,grid[ir][ic].y);*/
            grid[ir][ic].x=(grid[ir][ic].x-entpnt)/At;
            grid[ir][ic].y=grid[ir][ic].y/At;
     /* printf("%i %i %f %f\n",ir,ic,grid[ir][ic].x,grid[ir][ic].y);*/
    }
  /*exit(0);*/
 /* At=grid[bl2][thra].y-grid[bl1][thra].y;*/
  /*At=2.0; */
  xin=grid[n_r/2][11].x;
  Ain=grid[n_r-1][0].y-grid[0][0].y;
    Ae=grid[n_r-1][n_c-1].y-grid[0][n_c-1].y;
    qin=Ain;
    /*printf("%f %f %f %f %i\n",Ain,Ae,At,xt,thra);   */
    /* Nozzle Flow B.C.'s */
   /*  Initialize Interior Nodes  */
   /*printf("here 6 %f %f %f\n",xin,Ain,Ae);*/
    for(ic=0; ic<n_c; ic++){
      for(ir=0; ir<n_r; ir++){
        bub=0;
        if(grid[ir][ic].x < xt+0.001){
          At=1.0;
          if(ic < 11){
            area=Ain;
            bub=1;
    /*printf("%f ",qin);*/
          }else{
            area=Ain-(Ain-At)/(xt-xin)*(grid[ir][ic].x-xin);
            bub=2;
          }
            eps=area/At;

  /*printf("%i %i %f %f %f %f\n",ir,ic,M,area,grid[ir][ic].x,grid[ir][ic].y);*/
          /*printf("%i %i area %f \n",ir,ic,area);*/

          if(grid[ir][ic].x < 2.0 && grid[ir][ic].x > 1.5){
            M1=0.3;
            M2=0.5;
          }else if (grid[ir][ic].x > 2.0) {
            M1=0.5;
            M2=0.99;
          }else {
            M1=0.3;
            M2=0.05;
          }


          M=0.0;
          /*if(ic > 10){*/
          do{
            savem=M;

            a=eps*eps-1/(M1*M1)*pow(2/(GMA+1)*(1+GMA1/2.*M1*M1),
                                    ((GMA+1)/GMA1));
            b=eps*eps-1/(M2*M2)*pow(2/(GMA+1)*(1+GMA1/2.*M2*M2),
                                    ((GMA+1)/GMA1));
            g=(a-b)/(M1-M2);
            M=M2-b/g;

            error=ABS(M-savem);
            M1=M2;
            M2=M;


          }while (error > 0.00001);
        /*}*/
        }else {

          At=1.0;

          area=At+(Ae-At)/(grid[0][n_c-1].x-xt)*(grid[ir][ic].x-xt);
          eps=area/At;
          M1=1.01;
          M2=4.0;
          do{
            a=eps*eps-1/(M1*M1)*pow(2/(GMA+1)*(1+GMA1/2.*M1*M1),
```

```
                                ((GMA+1)/GMA1));
        b=eps*eps-1/(M2*M2)*pow(2/(GMA+1)*(1+GMA1/2.*M2*M2),
                                ((GMA+1)/GMA1));
        g=(a-b)/(M1-M2);
        M=M2-b/g;
        error=ABS(M-savem);
        M1=M2;
        M2=M;

        savem=M;

      }while (error > 0.00001);
    }

    /*if(ir == 20){*/
/*printf("%i %i %f %f %f %f %i\n",ir,ic,M,area,grid[ir][ic].x,grid[ir][ic].y,bub);*/
    /*}*/

    if(ic == 11)
      Min=M;

    state[ir][ic].w[0]=1.0/pow(1.+GMA1/2.*M*M,1./GMA1)/rtr1;
    /*state[ir][ic].p=1./pow(1.0+GMA1/2.*M*M,GMA/GMA1)/ptp1;*/
      /* state[ir][ic].w[0]; */
    /* Tstat is NonD temp and a is NonD speed of sound */
    Tstat[ir][ic]=1./(1.+GMA1/2.*M*M)/TtT1;
    state[ir][ic].p=state[ir][ic].w[0]*Tstat[ir][ic]/GMA;
    a=sqrt(GMA*state[ir][ic].p/state[ir][ic].w[0]);
    state[ir][ic].w[1]=state[ir][ic].w[0]*M*a;
    state[ir][ic].w[2]=0.0;
    state[ir][ic].w[3]=state[ir][ic].w[0]*(1.0/GMA/GMA1*Tstat[ir][ic])
      +0.5*(state[ir][ic].w[1]*state[ir][ic].w[1]
            +state[ir][ic].w[2]*state[ir][ic].w[2])/state[ir][ic].w[0];

    /*if(ic = n_c-1){
      printf(" %i Mach: %f Rho: %f rho*u %f a: %f p: %f\n",ir,M,
             state[ir][ic].w[0], state[ir][ic].w[1],a,state[ir][ic].p);
    }*/
  }
    c8[ic]=Tstat[n_r/2][ic];
 /* printf("%i Tstat: %f %f\n",ic,Tstat[16][ic],c8[ic]);*/
}
Mie=M;
    /*  Initial Boundary Conditions */
     /*printf("here 7 \n");*/

    for(ic=0; ic<n_c; ic++){
      state[0][ic].p=state[1][ic].p;
      state[n_r-1][ic].p=state[n_r-2][ic].p;
      state[0][ic].w[3]=state[0][ic].w[0]*ewall;
      state[n_r-1][ic].w[3]=state[n_r-1][ic].w[0]*ewall;
      state[0][ic].w[1]=0.0;
      state[n_r-1][ic].w[1]=0.0;
      state[0][ic].w[2]=0.0;
      state[n_r-1][ic].w[2]=0.0;
    }

    /* BL patch */




    for(ic=0;ic<n_c;ic++){
```

```
    c1=0.0;
    c2=state[bl1+1][ic].w[1]/state[bl1+1][ic].w[0];

    c3=grid[0][ic].y;
    c4=grid[bl1][ic].y;

    c5=grid[n_r-1][ic].y;
    c6=grid[bl2][ic].y;

    c7=1.0/TtT1;
    /*c8=Tstat[n_r/2][ic];*/
    /*printf("%i Tstat: %f %f\n",ic,c7,c8[ic]);*/

    c9=state[0][ic].w[0];
    c10=state[bl1+1][ic].w[0];

    p1=(c3-c4)*(c3-c4)/-4.0/c2;
    p2=(c5-c6)*(c5-c6)/-4.0/c2;

    pt1=(c3-c4)*(c3-c4)/4.0/(c7-c8[ic]);
    pt2=(c5-c6)*(c5-c6)/4.0/(c7-c8[ic]);

    pr1=(c3-c4)*(c3-c4)/4.0/(c9-c10);
    pr2=(c5-c6)*(c5-c6)/4.0/(c9-c10);

    for(ir=0;ir<bl1;ir++){

      Tstat[ir][ic]=pow(grid[ir][ic].y-c4,2)/4.0/pt1+c8[ic];
      state[ir][ic].w[0]=state[ir][ic].p*GMA/Tstat[ir][ic];
      state[ir][ic].w[1]=(pow(grid[ir][ic].y-c4,2)/4.0/p1+c2)
        *state[ir][ic].w[0];
      /*if(ic < n_c-1){ */
      state[ir][ic].w[3]=state[ir][ic].w[0]/GMA/GMA1*Tstat[ir][ic]
        +0.5*(state[ir][ic].w[1]*state[ir][ic].w[1]
              +state[ir][ic].w[2]*state[ir][ic].w[2])/state[ir][ic].w[0];
    /*}else{
       state[ir][ic].w[3]=POUT/GMA1
        +0.5*(state[ir][ic].w[1]*state[ir][ic].w[1]
              +state[ir][ic].w[2]*state[ir][ic].w[2])/state[ir][ic].w[0];
     }*/
    }

    for(ir=bl2;ir<n_r;ir++){



      Tstat[ir][ic]=pow(grid[ir][ic].y-c6,2)/4.0/pt2+c8[ic];
      state[ir][ic].w[0]=state[ir][ic].p*GMA/Tstat[ir][ic];
       state[ir][ic].w[1]=(pow(grid[ir][ic].y-c6,2)/4.0/p2+c2)
        *state[ir][ic].w[0];
    /*  if(ic < n_c-1){*/
      state[ir][ic].w[3]=state[ir][ic].w[0]/GMA/GMA1*Tstat[ir][ic]
        +0.5*(state[ir][ic].w[1]*state[ir][ic].w[1]
              +state[ir][ic].w[2]*state[ir][ic].w[2])/state[ir][ic].w[0];
      /*}else{
        state[ir][ic].w[3]=POUT/GMA1
          +0.5*(state[ir][ic].w[1]*state[ir][ic].w[1]
          +state[ir][ic].w[2]*state[ir][ic].w[2])/state[ir][ic].w[0];
      }*/

    }

  }

  for(ir=0; ir<n_r; ir++){
```

```
/* Stagnation Conditions */
        ff[ir].w[0]=pow(((GMA+1.0)/2.0),1.0/GMA1);
        ff[ir].w[1]=0.0;
        ff[ir].w[2]=0.0;
        ff[ir].w[3]=(GMA+1)/2.0;
        ff[ir].p=ff[ir].w[0]*ff[ir].w[3]/GMA;

/* Farfield Conditions Based on inlet Mach Number */
        ff[ir].w[0]=ff[ir].w[0]/pow(1.0+GMA1/2.0*Min*Min,1.0/GMA1);
        ff[ir].p=ff[ir].p/pow(1+GMA1/2.0*Min*Min,GMA/GMA1);
        ff[ir].w[1]=Min*sqrt(GMA*ff[ir].p/ff[ir].w[0]);
        ff[ir].w[2]=0.0;
        ff[ir].w[3]=1.2;


        /*printf("%i %f %f %f %f %f %f\n",ir,state[ir][0].w[0],
               state[ir][0].w[1],state[ir][0].w[2],state[ir][0].w[3],
               state[ir][0].p,GMA*state[ir][0].p/state[ir][0].w[0]); */

      }

      /* Grid Metrics: Gradient of Wall Normals and Tangentials */

  ds=sqrt((grid[1][0].x-grid[0][0].x)*(grid[1][0].x-grid[0][0].x)
          +(grid[1][0].y-grid[0][0].y)*(grid[1][0].y-grid[0][0].y));
  dtp=sqrt((grid[0][1].x-grid[0][0].x)*(grid[0][1].x-grid[0][0].x)
           +(grid[0][1].y-grid[0][0].y)*(grid[0][1].y-grid[0][0].y));

  xt=(grid[0][1].x-grid[0][0].x)/dtp;
  yt=(grid[0][1].y-grid[0][0].y)/dtp;
  /*xn=(grid[1][0].x-grid[0][0].x)/ds;
  yn=(grid[1][0].y-grid[0][0].y)/ds;*/
  alph=atan((grid[0][1].y-grid[0][0].y)/(grid[0][1].x-grid[0][0].x));
  if(alph < 0){
    eps=pi/2+alph;
  }else{
    eps=pi/2+alph;
  }
   xn=cos(eps);
   yn=sin(eps);
  ijac1= 1/(xt*yn-yt*xn);
  xs =  (grid[0][1].x-grid[0][0].x);
  ys =  (grid[0][1].y-grid[0][0].y);
  xe =  0.5*(-3*grid[0][0].x+4*grid[1][0].x-grid[2][0].x);
  ye =  0.5*(-3*grid[0][0].y+4*grid[1][0].y-grid[2][0].y);
  ts[0].l = ijac1*(yn*xs-xn*ys);
  ns[0].l = ijac1*(-yt*xs+xt*ys);
  te[0].l = ijac1*(yn*xe-xn*ye);
  ne[0].l = ijac1*(-yt*xe+xt*ye);


  ds=sqrt((grid[n_r-1][0].x-grid[n_r-2][0].x)*
          (grid[n_r-1][0].x-grid[n_r-2][0].x)+
          (grid[n_r-1][0].y-grid[n_r-2][0].y)*
          (grid[n_r-1][0].y-grid[n_r-2][0].y));
  dtp=sqrt((grid[n_r-1][1].x-grid[n_r-1][0].x)*
           (grid[n_r-1][1].x-grid[n_r-1][0].x)+
           (grid[n_r-1][1].y-grid[n_r-1][0].y)*
           (grid[n_r-1][1].y-grid[n_r-1][0].y));
  xt=(grid[n_r-1][1].x-grid[n_r-1][0].x)/dtp;
  yt=(grid[n_r-1][1].y-grid[n_r-1][0].y)/dtp;
  /*xn=(grid[n_r-1][0].x-grid[n_r-2][0].x)/ds;
  yn=(grid[n_r-1][0].y-grid[n_r-2][0].y)/ds;*/
  alph=atan((grid[n_r-1][1].y-grid[n_r-1][0].y)
     /(grid[n_r-1][1].x-grid[n_r-1][0].x));
  if(alph < 0){
    eps=3*pi/2+alph;
  }else{
    eps=3*pi/2+alph;
  }
   xn=cos(eps);
   yn=sin(eps);
  ijac1= 1/(xt*yn-yt*xn);
  xs =  (grid[n_r-1][1].x-grid[n_r-1][0].x);
  ys =  (grid[n_r-1][1].y-grid[n_r-1][0].y);
  xe =  0.5*(3*grid[n_r-1][0].x-4*grid[n_r-2][0].x+grid[n_r-3][0].x);
  ye =  0.5*(3*grid[n_r-1][0].y-4*grid[n_r-2][0].y+grid[n_r-3][0].y);
  ts[0].u = ijac1*(yn*xs-xn*ys);
  ns[0].u = ijac1*(-yt*xs+xt*ys);
  te[0].u = ijac1*(yn*xe-xn*ye);
  ne[0].u = ijac1*(-yt*xe+xt*ye);

 for(ic=1;ic<n_c-1;ic++){
    icp1=ic+1;
    icm1=ic-1;

    ds=sqrt((grid[1][ic].x-grid[0][ic].x)*(grid[1][ic].x-grid[0][ic].x)
            +(grid[1][ic].y-grid[0][ic].y)*(grid[1][ic].y-grid[0][ic].y));
    dtp=sqrt((grid[0][icp1].x-grid[0][ic].x)*(grid[0][icp1].x-grid[0][ic].x)
             +(grid[0][icp1].y-grid[0][ic].y)*(grid[0][icp1].y-grid[0][ic].y));
    dtm=sqrt((grid[0][ic].x-grid[0][icm1].x)*(grid[0][ic].x-grid[0][icm1].x)
             +(grid[0][ic].y-grid[0][icm1].y)*(grid[0][ic].y-grid[0][icm1].y));
    sumdt=dtp+dtm;

    xt=1/sumdt*(dtm/dtp*(grid[0][icp1].x-grid[0][ic].x)
                +dtp/dtm*(grid[0][ic].x-grid[0][icm1].x));
    yt=1/sumdt*(dtm/dtp*(grid[0][icp1].y-grid[0][ic].y)
                +dtp/dtm*(grid[0][ic].y-grid[0][icm1].y));
   /* xn=(grid[1][ic].x-grid[0][ic].x)/ds;
    yn=(grid[1][ic].y-grid[0][ic].y)/ds;*/
  alph=atan((grid[0][icp1].y-grid[0][ic].y)/(grid[0][icp1].x-grid[0][ic].x));
  if(alph < 0){
    eps=pi/2+alph;
  }else{
    eps=pi/2+alph;
  }
   xn=cos(eps);
   yn=sin(eps);
    ijac1= 1/(xt*yn-yt*xn);
    xs = 0.5*(grid[0][icp1].x-grid[0][icm1].x);
    ys = 0.5*(grid[0][icp1].y-grid[0][icm1].y);
    xe = 0.5*(-3*grid[0][ic].x+4*grid[1][ic].x-grid[2][ic].x);
    ye = 0.5*(-3*grid[0][ic].y+4*grid[1][ic].y-grid[2][ic].y);
    ts[ic].l = ijac1*(yn*xs-xn*ys);
    ns[ic].l = ijac1*(-yt*xs+xt*ys);
    te[ic].l = ijac1*(yn*xe-xn*ye);
    ne[ic].l = ijac1*(-yt*xe+xt*ye);


    ds=sqrt((grid[n_r-1][ic].x-grid[n_r-2][ic].x)*
            (grid[n_r-1][ic].x-grid[n_r-2][ic].x)+
            (grid[n_r-1][ic].y-grid[n_r-2][ic].y)*
            (grid[n_r-1][ic].y-grid[n_r-2][ic].y));
    dtp=sqrt((grid[n_r-1][icp1].x-grid[n_r-1][ic].x)*
             (grid[n_r-1][icp1].x-grid[n_r-1][ic].x)+
             (grid[n_r-1][icp1].y-grid[n_r-1][ic].y)*
             (grid[n_r-1][icp1].y-grid[n_r-1][ic].y));
    dtm=sqrt((grid[n_r-1][ic].x-grid[n_r-1][icm1].x)*
             (grid[n_r-1][ic].x-grid[n_r-1][icm1].x)+
```

```
            (grid[n_r-1][ic].y-grid[n_r-1][icm1].y)*
            (grid[n_r-1][ic].y-grid[n_r-1][icm1].y));
    sumdt=dtp+dtm;
    xt=1/sumdt*(dtm/dtp*(grid[n_r-1][icp1].x-grid[n_r-1][ic].x)
                +dtp/dtm*(grid[n_r-1][ic].x-grid[n_r-1][icm1].x));
    yt=1/sumdt*(dtm/dtp*(grid[n_r-1][icp1].y-grid[n_r-1][ic].y)
                +dtp/dtm*(grid[n_r-1][ic].y-grid[n_r-1][icm1].y));
   /* xn=(grid[n_r-1][ic].x-grid[n_r-2][ic].x)/ds;
    yn=(grid[n_r-1][ic].y-grid[n_r-2][ic].y)/ds;*/
  alph=atan((grid[n_r-1][icp1].y-grid[n_r-1][ic].y)
     /(grid[n_r-1][icp1].x-grid[n_r-1][ic].x));
  if(alph < 0){
    eps=3*pi/2+alph;
  }else{
    eps=3*pi/2+alph;
  }
   xn=cos(eps);
   yn=sin(eps);

    ijac1= 1/(xt*yn-yt*xn);
    xs = 0.5*(grid[n_r-1][icp1].x-grid[n_r-1][icm1].x);
    ys = 0.5*(grid[n_r-1][icp1].y-grid[n_r-1][icm1].y);
    xe = 0.5*(3*grid[n_r-1][ic].x-4*grid[n_r-2][ic].x+grid[n_r-3][ic].x);
    ye = 0.5*(3*grid[n_r-1][ic].y-4*grid[n_r-2][ic].y+grid[n_r-3][ic].y);
    ts[ic].u = ijac1*(yn*xs-xn*ys);
    ns[ic].u = ijac1*(-yt*xs+xt*ys);
    te[ic].u = ijac1*(yn*xe-xn*ye);
    ne[ic].u = ijac1*(-yt*xe+xt*ye);
    /*printf("%i %f %f %f %f %f %f %f %f %f \n",ic,alph*180/pi,eps*180/pi,
          xt,yt,xn,yn,ts[ic].u,te[ic].u,ns[ic].u,ne[ic].u);*/
  /*  printf("%f %f %f %f \n", te[ic].l, te[ic].u, ts[ic].l, ts[ic].u);*/
  }

/* Grid metrics for Outflow corners */

  ds=sqrt((grid[1][n_c-1].x-grid[0][n_c-1].x)
          *(grid[1][n_c-1].x-grid[0][n_c-1].x)
          +(grid[1][n_c-1].y-grid[0][n_c-1].y)
          *(grid[1][n_c-1].y-grid[0][n_c-1].y));
  dtp=sqrt((grid[0][n_c-1].x-grid[0][n_c-2].x)
           *(grid[0][n_c-1].x-grid[0][n_c-2].x)
           +(grid[0][n_c-1].y-grid[0][n_c-2].y)
           *(grid[0][n_c-1].y-grid[0][n_c-2].y));

  xt=(grid[0][n_c-1].x-grid[0][n_c-2].x)/dtp;
  yt=(grid[0][n_c-1].y-grid[0][n_c-2].y)/dtp;
  /*xn=(grid[1][n_c-1].x-grid[0][n_c-1].x)/ds;
  yn=(grid[1][n_c-1].y-grid[0][n_c-1].y)/ds;*/
  alph=atan((grid[0][n_c-1].y-grid[0][n_c-2].y)
       /(grid[0][n_c-1].x-grid[0][n_c-2].x));
  if(alph < 0){
    eps=pi/2+alph;
  }else{
    eps=pi/2+alph;
  }
   xn=cos(eps);
   yn=sin(eps);
  ijac1= 1/(xt*yn-yt*xn);
  xs = (grid[0][n_c-1].x-grid[0][n_c-2].x);
  ys = (grid[0][n_c-1].y-grid[0][n_c-2].y);
  xe = 0.5*(-3*grid[0][n_c-1].x+4*grid[1][n_c-1].x-grid[2][n_c-1].x);
  ye = 0.5*(-3*grid[0][n_c-1].y+4*grid[1][n_c-1].y-grid[2][n_c-1].y);
  ts[n_c-1].l = ijac1*(yn*xs-xn*ys);
  ns[n_c-1].l = ijac1*(-yt*xs+xt*ys);
  te[n_c-1].l = ijac1*(yn*xe-xn*ye);
  ne[n_c-1].l = ijac1*(-yt*xe+xt*ye);


  ds=sqrt((grid[n_r-1][n_c-1].x-grid[n_r-2][n_c-1].x)*
          (grid[n_r-1][n_c-1].x-grid[n_r-2][n_c-1].x)+
          (grid[n_r-1][n_c-1].y-grid[n_r-2][n_c-1].y)*
          (grid[n_r-1][n_c-1].y-grid[n_r-2][n_c-1].y));
  dtp=sqrt((grid[n_r-1][n_c-1].x-grid[n_r-1][n_c-2].x)*
           (grid[n_r-1][n_c-1].x-grid[n_r-1][n_c-2].x)+
           (grid[n_r-1][n_c-1].y-grid[n_r-1][n_c-2].y)*
           (grid[n_r-1][n_c-1].y-grid[n_r-1][n_c-2].y));
  xt=(grid[n_r-1][n_c-1].x-grid[n_r-1][n_c-2].x)/dtp;
  yt=(grid[n_r-1][n_c-1].y-grid[n_r-1][n_c-2].y)/dtp;
  /*xn=(grid[n_r-1][n_c-1].x-grid[n_r-2][n_c-1].x)/ds;
  yn=(grid[n_r-1][n_c-1].y-grid[n_r-2][n_c-1].y)/ds;*/
  alph=atan((grid[n_r-1][n_c-1].y-grid[n_r-1][n_c-2].y)
     /(grid[n_r-1][n_c-1].x-grid[n_r-1][n_c-2].x));
  if(alph < 0){
    eps=3*pi/2+alph;
  }else{
    eps=3*pi/2+alph;
  }
   xn=cos(eps);
   yn=sin(eps);
  ijac1= 1/(xt*yn-yt*xn);
  xs = (grid[n_r-1][n_c-1].x-grid[n_r-1][n_c-2].x);
  ys = (grid[n_r-1][n_c-1].y-grid[n_r-1][n_c-2].y);
  xe = 0.5*(3*grid[n_r-1][n_c-1].x-4*grid[n_r-2][n_c-1].x
            +grid[n_r-3][n_c-1].x);
  ye = 0.5*(3*grid[n_r-1][n_c-1].y-4*grid[n_r-2][n_c-1].y
            +grid[n_r-3][n_c-1].y);
  ts[n_c-1].u = ijac1*(yn*xs-xn*ys);
  ns[n_c-1].u = ijac1*(-yt*xs+xt*ys);
  te[n_c-1].u = ijac1*(yn*xe-xn*ye);
  ne[n_c-1].u = ijac1*(-yt*xe+xt*ye);



      return Mie;
}

void Load_Vars(int n_r, int n_c, struct node **state, struct flux **iflux,
               struct flux **vflux)
{
  int ir, ic, iv, rstep, cstep;
  float *st;
  double p,u,v,rho;
  double *iflx, *vflx;
  FILE *fp_rho, *fp_p, *fp_u, *fp_v;

  fp_rho = fopen("r.dat","r");
  fp_p   = fopen("p.dat","r");
  fp_u   = fopen("u.dat","r");
  fp_v   = fopen("v.dat","r");

#ifdef DOUBGRID
  rstep = cstep = 2;
#elif defined DOUBCNDS
  rstep=1;
  cstep=2;
#else
  rstep = cstep = 1;
#endif
```

```
  for(ir=0; ir<n_r; ir+=rstep){
    for(ic=0; ic<n_c; ic+=cstep){
      st = state[ir][ic].w;
      fscanf(fp_rho,"%lf",&rho);
      fscanf(fp_u,  "%lf",&u);
      fscanf(fp_v,  "%lf",&v);
      fscanf(fp_p,  "%lf",&p);

      st[0] = rho;
      st[1] = rho*u;
      st[2] = rho*v;
      st[3] = p/GMA1+0.5*rho*(u*u+v*v);

     /**** Skip a column in data files if we are coarsening circumf. dir. ****/
#if defined HALFCNDS || defined HALFGRID
      fscanf(fp_rho,"%lf",&rho);
      fscanf(fp_u,  "%lf",&u);
      fscanf(fp_v,  "%lf",&v);
      fscanf(fp_p,  "%lf",&p);
#endif

      /**** Zero fluxes for first iteration ****/
      iflx = iflux[ir][ic].w;
      vflx = vflux[ir][ic].w;
      for (iv=0;iv<4;iv++)
        iflx[iv] = vflx[iv] = 0;
    }
    /**** Skip a row in data files if we are coarsening radial direction ****/
#ifdef HALFGRID
    if(ir<n_r-1){
      for(ic=0; ic<n_c; ic+=cstep){
        fscanf(fp_rho,"%lf %lf",&rho,&rho);
        fscanf(fp_u,  "%lf %lf",&u,&u);
        fscanf(fp_v,  "%lf %lf",&v,&v);
        fscanf(fp_p,  "%lf %lf",&p,&p);
      }
    }
#endif
  }
  fclose(fp_rho);
  fclose(fp_p);
  fclose(fp_u);
  fclose(fp_v);
}


/*~~~~ This function interpolates data from a coarse grid onto a fine grid ~~*/
void Dgrid_Interpolate(int n_r, int n_c, struct node **state,
                       struct flux **iflux, struct flux **vflux)
{
  int ir,ic,iv;
  double *iflx,*vflx;

  /**** If we are doubling both radial and circumferential nodes, first fill
    gaps in radial direction on partially filled columns by interpolating
   from rows above and below empty position ****/
#ifdef DOUBGRID
  for(ir=1;ir<n_r-1;ir+=2)
    for(ic=0;ic<n_c;ic+=2)
      for(iv=0;iv<4;iv++)
        state[ir][ic].w[iv]=0.5*(state[ir-1][ic].w[iv]+state[ir+1][ic].w[iv]);
#endif

  /**** Fill in missing portions of rows by interpolating from columns
    to the left and right of empty position ****/
  for(ir=0;ir<n_r;ir++){
    for(ic=1;ic<n_c-1;ic+=2)
      for(iv=0;iv<4;iv++)
        state[ir][ic].w[iv]=0.5*(state[ir][ic-1].w[iv]+state[ir][ic+1].w[iv]);
    for(iv=0;iv<4;iv++)
      state[ir][ic].w[iv]=0.5*(state[ir][ic-2].w[iv]+state[ir][0].w[iv]);
  }

  /**** Initialize fluxes for first iteration ****/
  for(ir=0;ir<n_r;ir++)
    for(ic=0;ic<n_c;ic++){
      iflx = iflux[ir][ic].w;
      vflx = vflux[ir][ic].w;
      for (iv=0;iv<4;iv++)
        iflx[iv] = vflx[iv] = 0;
    }
}


/*********************    Initial Timestep   ****************************/

/*~~~~ This function calculates an initial time step (needed by the artificial
  viscosity terms) for the first iteration ~~~~*/
void Calc_I_TStep(int n_r, int n_c, struct point **grid, struct node **state,
                  float **tstep)
{
  int ir,ic,icp1,icm1,icm2,icp2;
  float u,v,u_hat,v_hat,rho,p,a,dxds,dxdn,dyds,dydn,
    Re_x,Re_y,t,*w,jac,a_temp;
  struct point *grp1, *gr, *grp2, *grm1, *grm2;

  tstep[0][0] = 8888;

  for(ir=0;ir<n_r-1;ir++){

    grp1 = grid[ir+1];
    grp2 = grid[ir+2];
    gr   = grid[ir];
    grm1 = grid[ir-1];
    grm2 = grid[ir-2];

    for(ic=0;ic<n_c;ic++){

      w = state[ir][ic].w;

      icp2 = ic+2;
      icp1 = ic+1;
      icm1 = ic-1;
      icm2 = ic-2;

      if(ic<1){
        dxds=0.5*(-3*gr[ic].x+4*gr[icp1].x-gr[icp2].x);
        dyds=0.5*(-3*gr[ic].y+4*gr[icp1].y-gr[icp2].y);

      }else if(ic>n_c-2){

        dxds=0.5*(3*gr[ic].x-4*gr[icm1].x+gr[icm2].x);
        dyds=0.5*(3*gr[ic].y-4*gr[icm1].y+gr[icm2].y);

      }else{
        dxds =0.5*(gr[icp1].x-gr[icm1].x);
      dyds =0.5*(gr[icp1].y-gr[icm1].y);
      }

      if(ir<1){
```

```
        dxdn=0.5*(-3*gr[ic].x+4*grp1[ic].x-grp2[ic].x);
        dydn=0.5*(-3*gr[ic].y+4*grp1[ic].y-grp2[ic].y);

      }else if(ir>n_c-2){
        dxdn=0.5*(3*gr[ic].x-4*grm1[ic].x+grm2[ic].x);
        dydn=0.5*(3*gr[ic].y-4*grm1[ic].y+grm2[ic].y);

      }else{
        dxdn =0.5*(grp1[ic].x-grm1[ic].x);
        dydn =0.5*(grp1[ic].y-grm1[ic].y);

      }

      rho=w[0];
      u = w[1]/rho;
      v = w[2]/rho;
      jac = dxds*dydn - dxdn*dyds;
      u_hat=u*dydn-v*dxdn;
      v_hat=v*dxds-u*dyds;
      u_hat = ABS(u_hat);
      v_hat = ABS(v_hat);
      p = GMA1*(w[3] - 0.5*(w[1]*w[1]/w[0]+ w[2]*w[2]/w[0]));
      a = sqrt(GMA*p/rho);
      Re_x = rho*(u_hat+a);
      Re_y = rho*(v_hat+a);
      a_temp = dxds*dxdn+dydn*dyds;
      t = 2.8284*ABS(jac)/(TS_SF*(1+2/MIN(Re_x,Re_y))*(u_hat+
      v_hat+a*sqrt(dxds*dxds+dyds*dyds+dxdn*dxdn+dydn*dydn+2*ABS(a_temp))));

      tstep[0][0] = MIN(tstep[0][0],t);

    }
  }
}


/*********************    Dump Grid   ******************************/
/*~~~~ This function dumps a copy of the current grid to disk ~~~~*/
void Dump_Grid(int n_r, int n_c, struct point **grid)
{
  int ir, ic;
  FILE *outx, *outy;

  outx = fopen("grid_x","w");
  outy = fopen("grid_y","w");

  for (ir=0; ir<n_r; ir++){
    for(ic=0; ic<n_c; ic++){
      fprintf(outx,"%f ",grid[ir][ic].x);
      fprintf(outy,"%f ",grid[ir][ic].y);
    }
    fprintf(outx,"\n");
    fprintf(outy,"\n");
  }
  fclose(outx);
  fclose(outy);
}


/*********************    IFLUX   ******************************/
/*~~~~ This function calculates inviscid (Euler equation) fluxes from
  the state vector supplied as 'input' and stores them in 'iflux' ~~~~*/
void Calc_IFlux(int n_r, int n_c, struct point **grid, struct node **input,
                struct flux **sumflx, double theta_step, int wfact)
{
  int ir, ic, icp1, icm1, iv;
  float u, v, p, f[4], g[4], *in, pb, brad, dxw1, dyw1, dxw2, dyw2,
    dxe1, dye1, dxe2, f2[4],g2[4];
  float dye2;
  double dxn, dyn, dxs, dys, dxe, dye, dxw, dyw;
#ifndef MOVING_GRID
  static int c_flag=0;
  struct flxface *iffc;
#endif

  /* Compute Fluxes from Walls */

  /* Lower Left Corner*/

  /* Some terms for f=x+y are left in comments, this is to allow an easy
check in the flux calculations. The flux of the grid should identically equal
 to 2.  Thus if f=x+y, the state variables should all be 2*/

  u = input[0][0].w[1] / input[0][0].w[0];
  v = input[0][0].w[2] / input[0][0].w[0];

  /**** Calculate the pressure at this node ****/
  p = GMA1*(input[0][0].w[3] - 0.5*(input[0][0].w[1]*u
                                    + input[0][0].w[2]*v));

  /**** Calculate flux vectors ****/


  f[0] = input[0][0].w[1];
  /*f[0] = grid[0][0].x+grid[0][0].y;*/
  f[1] = input[0][0].w[1]*u+p;
  f[2] = g[1] = input[0][0].w[1]*v;
  f[3] = (input[0][0].w[3]+p)*u;
  g[0] = input[0][0].w[2];
  /*g[0] = grid[0][0].x + grid[0][0].y;*/
  g[2] = input[0][0].w[2]*v+p;
  g[3] = (input[0][0].w[3]+p)*v;

  dxw1=0;
  dyw1=0;
  dxe1 = 0.5*(grid[0][1].x - grid[0][0].x);
  dye1 = 0.5*(grid[0][1].y - grid[0][0].y);
  dxn = 0.5*(grid[1][0].x - grid[0][0].x);
  dyn = 0.5*(grid[1][0].y - grid[0][0].y);

  for(iv=0;iv<4;iv++){
    sumflx[1][1].w[iv] += (f[iv]*dye1-g[iv]*dxe1) -
      (f[iv]*dyn-g[iv]*dxn);
    sumflx[1][0].w[iv] += (dye1*f[iv]-dxe1*g[iv])-
       (dyn*f[iv]-dxn*g[iv]);
    sumflx[0][1].w[iv] += (dye1*f[iv]-dxe1*g[iv]) -
      (dyn*f[iv]-dxn*g[iv]);
    sumflx[0][0].w[iv] += (dye1*f[iv]-dxe1*g[iv]) -
      (dyn*f[iv]-dxn*g[iv]);

  }

  /**** Store pressure for later use ****/
  input[0][0].p=p;

  /* repeat for upper left corner*/

  u = input[n_r-1][0].w[1] / input[n_r-1][0].w[0];
  v = input[n_r-1][0].w[2] / input[n_r-1][0].w[0];
```

```
/**** Calculate the pressure at this node ****/
p = GMA1*(input[n_r-1][0].w[3] - 0.5*(input[n_r-1][0].w[1]*u
                                    + input[n_r-1][0].w[2]*v));


/**** Calculate flux vectors ****/
f2[0] = input[n_r-1][0].w[1];
/*f2[0] = grid[n_r-1][0].x+grid[n_r-1][0].y;*/
f2[1] = input[n_r-1][0].w[1]*u+p;
f2[2] = g2[1] = input[n_r-1][0].w[1]*v;
f2[3] = (input[n_r-1][0].w[3]+p)*u;
g2[0] = input[n_r-1][0].w[2];
/*g2[0] = grid[n_r-1][0].x+grid[n_r-1][0].y;*/
g2[2] = input[n_r-1][0].w[2]*v+p;
g2[3] = (input[n_r-1][0].w[3]+p)*v;

input[n_r-1][0].p=p;


dxw2=0;
dyw2=0;

dxe2 = 0.5*(grid[n_r-1][1].x - grid[n_r-1][0].x);
dye2 = 0.5*(grid[n_r-1][1].y - grid[n_r-1][0].y);

dxs = 0.5*(grid[n_r-2][0].x - grid[n_r-1][0].x);
dys = 0.5*(grid[n_r-2][0].y - grid[n_r-1][0].y);

for(iv=0;iv<4;iv++){
  sumflx[n_r-2][1].w[iv] += (-f2[iv]*dye2+g2[iv]*dxe2) +
    (f2[iv]*dys-g2[iv]*dxs);
  sumflx[n_r-2][0].w[iv] += (-f2[iv]*dye2+g2[iv]*dxe2) +
    (f2[iv]*dys-g2[iv]*dxs);
  sumflx[n_r-1][1].w[iv] += (-f2[iv]*dye2+g2[iv]*dxe2) +
    (f2[iv]*dys-g2[iv]*dxs);
  sumflx[n_r-1][0].w[iv] += (-f2[iv]*dye2+g2[iv]*dxe2) +
    (f2[iv]*dys-g2[iv]*dxs);

}

/* Lower Surface*/

for(ic=1; ic<n_c-1; ic++){
  in = input[0][ic].w;
  u = in[1] / in[0];
  v = in[2] / in[0];

  /**** Calculate the pressure at this node ****/
  p = GMA1*(in[3] - 0.5*(in[1]*u + in[2]*v));

  /**** Calculate flux vectors ****/
  f[0] = in[1];
  /*f[0] = grid[0][ic].x+grid[0][ic].y;*/
  f[1] = in[1]*u+p;
  f[2] = g[1] = in[1]*v;
  f[3] = (in[3]+p)*u;
  g[0] = in[2];
  /*g[0] = grid[0][ic].x+grid[0][ic].y;*/
  g[2] = in[2]*v+p;
  g[3] = (in[3]+p)*v;


  /**** Store pressure for later use ****/
  input[0][ic].p = p;
```

```
/* repeat for upper surface */
in = input[n_r-1][ic].w;
u = in[1] / in[0];
v = in[2] / in[0];

/**** Calculate the pressure at this node ****/
p = GMA1*(in[3] - 0.5*(in[1]*u + in[2]*v));


/**** Calculate flux vectors ****/
f2[0] = in[1];
/*f2[0] = grid[n_r-1][ic].x+grid[n_r-1][ic].y;*/
f2[1] = in[1]*u+p;
f2[2] = g2[1] = in[1]*v;
f2[3] = (in[3]+p)*u;
g2[0] = in[2];
/*g2[0] = grid[n_r-1][ic].x + grid[n_r-1][ic].y;*/
g2[2] = in[2]*v+p;
g2[3] = (in[3]+p)*v;

/**** Store pressure for later use ****/
input[n_r-1][ic].p = p;

/* printf(" %f %f %f",u,v,p);
   printf(" %i %i f2 and g2 %f  %f \n", n_r-1, ic,  f2[1],g2[2]);*/

icm1=ic-1;
icp1=ic+1;

dxw1 = 0.5*(grid[0][ic].x - grid[0][icm1].x);
dyw1 = 0.5*(grid[0][ic].y - grid[0][icm1].y);
dxw2 = 0.5*(grid[n_r-1][ic].x - grid[n_r-1][icm1].x);
dyw2 = 0.5*(grid[n_r-1][ic].y - grid[n_r-1][icm1].y);


dxe1 = 0.5*(grid[0][icp1].x - grid[0][ic].x);
dye1 = 0.5*(grid[0][icp1].y - grid[0][ic].y);
dxe2 = 0.5*(grid[n_r-1][icp1].x - grid[n_r-1][ic].x);
dye2 = 0.5*(grid[n_r-1][icp1].y - grid[n_r-1][ic].y);

dxn = 0.5*(grid[1][ic].x - grid[0][ic].x);
dyn = 0.5*(grid[1][ic].y - grid[0][ic].y);
dxs = 0.5*(grid[n_r-2][ic].x - grid[n_r-1][ic].x);
dys = 0.5*(grid[n_r-2][ic].y - grid[n_r-1][ic].y);

for(iv=0;iv<4;iv++){

  sumflx[1][icm1].w[iv] += f[iv]*dyn-g[iv]*dxn +
    (f[iv]*dyw1-g[iv]*dxw1);
  sumflx[n_r-2][icm1].w[iv] += -f2[iv]*dys+g2[iv]*dxs -
    (f2[iv]*dyw2-g2[iv]*dxw2);
  sumflx[0][icm1].w[iv] += (f[iv]*dyw1-g[iv]*dxw1) +
    (f[iv]*dyn-g[iv]*dxn);
  sumflx[0][ic].w[iv] += (f[iv]*dyw1-g[iv]*dxw1) +
    (f[iv]*dye1-g[iv]*dxe1);
  sumflx[1][icp1].w[iv] += (f[iv]*dye1-g[iv]*dxe1) -
    (f[iv]*dyn-g[iv]*dxn);
  sumflx[n_r-2][icp1].w[iv] += (-f2[iv]*dye2+g2[iv]*dxe2) +
      (f2[iv]*dys-g2[iv]*dxs);
  sumflx[1][ic].w[iv] += (f[iv]*dye1-g[iv]*dxe1 +
                          (f[iv]*dyw1-g[iv]*dxw1));
  sumflx[n_r-2][ic].w[iv] += (-f2[iv]*dye2+g2[iv]*dxe2 -
                              (f2[iv]*dyw2-g2[iv]*dxw2));
  sumflx[0][icp1].w[iv] += (f[iv]*dye1-g[iv]*dxe1) -
```

```
      (f[iv]*dyn-g[iv]*dxn);
  }
}

/* Lower Right Corner */

in = input[0][n_c-1].w;
u = in[1] / in[0];
v = in[2] / in[0];

/**** Calculate the pressure at this node ****/
p = GMA1*(in[3] - 0.5*(in[1]*u + in[2]*v));

/**** Calculate flux vectors ****/
f[0] = in[1];
/*f[0] = grid[0][n_c-1].x+grid[0][n_c-1].y;*/
f[1] = in[1]*u+p;
f[2] = g[1] = in[1]*v;
f[3] = (in[3]+p)*u;
g[0] = in[2];
/*g[0] = grid[0][n_c-1].x+grid[0][n_c-1].y;*/
g[2] = in[2]*v+p;
g[3] = (in[3]+p)*v;


/**** Store pressure for later use ****/

input[0][n_c-1].p = p;

  /* repeat for upper right corner */
in = input[n_r-1][n_c-1].w;
u = in[1] / in[0];
v = in[2] / in[0];
p = input[n_r-1][n_c-1].p;

  /**** Calculate the pressure at this node ****/
p = GMA1*(in[3] - 0.5*(in[1]*u + in[2]*v));


/**** Calculate flux vectors ****/
f2[0] = in[1];
/*f2[0] = grid[n_r-1][n_c-1].x+grid[n_r-1][n_c-1].y;*/
f2[1] = in[1]*u+p;
f2[2] = g2[1] = in[1]*v;
f2[3] = (in[3]+p)*u;
g2[0] = in[2];
/*g2[0] = grid[n_r-1][n_c-1].x+grid[n_r-1][n_c-1].y;*/
g2[2] = in[2]*v+p;
g2[3] = (in[3]+p)*v;

/**** Store pressure for later use ****/
input[n_r-1][n_c-1].p = p;

ic = n_c-1;
icm1= ic -1;

dxe1=0;
dye1=0;
dxe2=0;
dye2=0;

dxw1 = 0.5*(grid[0][ic].x - grid[0][icm1].x);
dyw1 = 0.5*(grid[0][ic].y - grid[0][icm1].y);
dxw2 = 0.5*(grid[n_r-1][ic].x - grid[n_r-1][icm1].x);
dyw2 = 0.5*(grid[n_r-1][ic].y - grid[n_r-1][icm1].y);

dxn = 0.5*(grid[1][ic].x - grid[0][ic].x);
dyn = 0.5*(grid[1][ic].y - grid[0][ic].y);
dxs = 0.5*(grid[n_r-2][ic].x - grid[n_r-1][ic].x);
dys = 0.5*(grid[n_r-2][ic].y - grid[n_r-1][ic].y);

for(iv=0;iv<4;iv++){
  sumflx[1][icm1].w[iv] += f[iv]*dyn-g[iv]*dxn +
    (f[iv]*dyw1-g[iv]*dxw1);
  sumflx[n_r-2][icm1].w[iv] += -f2[iv]*dys+g2[iv]*dxs -
    (f2[iv]*dyw2-g2[iv]*dxw2);

  sumflx[1][ic].w[iv] += (f[iv]*dyn-g[iv]*dxn
                          +(f[iv]*dyw1-g[iv]*dxw1));

  sumflx[n_r-2][ic].w[iv] += (-f2[iv]*dys+g2[iv]*dxs
                              -(f2[iv]*dyw2-g2[iv]*dxw2));
  sumflx[0][icm1].w[iv] += (f[iv]*dyn-g[iv]*dxn) +
    (f[iv]*dyw1-g[iv]*dxw1);
  sumflx[n_r-1][icm1].w[iv] += -(f2[iv]*dyw2-g2[iv]*dxw2) -
    (f2[iv]*dys-g2[iv]*dxs);
  sumflx[0][ic].w[iv] += (f[iv]*dyn-g[iv]*dxn) +
    (f[iv]*dyw1-g[iv]*dxw1);
  sumflx[n_r-1][ic].w[iv] += -(f2[iv]*dyw2-g2[iv]*dxw2) -
    (f2[iv]*dys-g2[iv]*dxs);

}

/* Left Boundary (Inlet) */
/* This is commented out if the inlet conditions do not require knowledge of
   the inlet fluxes */

for(ir=1; ir<n_r-1; ir++){
  ic=0;
  icp1=1;

  in = input[ir][ic].w;
  u = in[1] / in[0];
  v = in[2] / in[0];

  /**** Calculate the pressure at this node ****/
  p = GMA1*(in[3] - 0.5*(in[1]*u + in[2]*v));

  /**** Calculate flux vectors ****/
  f[0] = in[1];
  /*f[0] = grid[ir][0].x + grid[ir][0].y;*/
  f[1] = in[1]*u+p;
  f[2] = g[1] = in[1]*v;
  f[3] = (in[3]+p)*u;
  g[0] = in[2];
  /*g[0] = grid[ir][0].x + grid[ir][0].y;*/
  g[2] = in[2]*v+p;
  g[3] = (in[3]+p)*v;


  /**** Store pressure for later use ****/
  input[ir][ic].p = p;


  dxw = 0;
  dyw = 0;

  dxn = 0.5*(grid[ir+1][ic].x - grid[ir][ic].x);
  dyn = 0.5*(grid[ir+1][ic].y - grid[ir][ic].y);
```

```
  dxs = 0.5*(grid[ir-1][ic].x - grid[ir][ic].x);
  dys = 0.5*(grid[ir-1][ic].y - grid[ir][ic].y);

  dxe = 0.5*(grid[ir][icp1].x - grid[ir][ic].x);
  dye = 0.5*(grid[ir][icp1].y - grid[ir][ic].y);

  for(iv=0;iv<4;iv++){

    sumflx[ir-1][ic].w[iv] +=
      f[iv]*dys-g[iv]*dxs - (f[iv]*dye-g[iv]*dxe);

    sumflx[ir+1][ic].w[iv] +=
      -f[iv]*dyn+g[iv]*dxn + (f[iv]*dye-g[iv]*dxe);

    sumflx[ir][ic].w[iv] +=
      -f[iv]*dyn + g[iv]*dxn +(f[iv]*dys-g[iv]*dxs);

    sumflx[ir-1][icp1].w[iv] +=
      f[iv]*dys-g[iv]*dxs - (f[iv]*dye-g[iv]*dxe);
    sumflx[ir+1][icp1].w[iv] +=
      f[iv]*dye-g[iv]*dxe - (f[iv]*dyn-g[iv]*dxn);

    sumflx[ir][icp1].w[iv] +=
      f[iv]*dys-g[iv]*dxs - (f[iv]*dyn-g[iv]*dxn);
  }
}

/* Right Boundary (Outlet) */
for(ir=1; ir<n_r-1; ir++){
  ic=n_c-1;
  icm1=n_c-2;

  in = input[ir][ic].w;
  u = in[1] / in[0];
  v = in[2] / in[0];

  /**** Calculate the pressure at this node ****/
  p = GMA1*(in[3] - 0.5*(in[1]*u + in[2]*v));

  /**** Calculate flux vectors ****/
  f[0] = in[1];
  /* f[0] = grid[ir][ic].x+grid[ir][ic].y;*/
  f[1] = in[1]*u+p;
  f[2] = g[1] = in[1]*v;
  f[3] = (in[3]+p)*u;
  g[0] = in[2];
  /*g[0] = grid[ir][ic].x+grid[ir][ic].y;*/
  g[2] = in[2]*v+p;
  g[3] = (in[3]+p)*v;


  /**** Store pressure for later use ****/
  input[ir][ic].p = p;
  dxe = 0;
  dye = 0;

  dxw =0.5*( grid[ir][ic].x-grid[ir][icm1].x);
  dyw =0.5*( grid[ir][ic].y-grid[ir][icm1].y);

  dxn = 0.5*(grid[ir+1][ic].x - grid[ir][ic].x);
  dyn = 0.5*(grid[ir+1][ic].y - grid[ir][ic].y);
  dxs = 0.5*(grid[ir-1][ic].x - grid[ir][ic].x);
  dys = 0.5*(grid[ir-1][ic].y - grid[ir][ic].y);

  for(iv=0;iv<4;iv++){

    sumflx[ir-1][ic].w[iv] +=
      -f[iv]*dys+g[iv]*dxs - (f[iv]*dyw-g[iv]*dxw);
    sumflx[ir+1][ic].w[iv] +=
      f[iv]*dyn-g[iv]*dxn + (f[iv]*dyw-g[iv]*dxw);
    sumflx[ir][ic].w[iv] +=
      f[iv]*dyn - g[iv]*dxn - (f[iv]*dys-g[iv]*dxs);
    sumflx[ir-1][icm1].w[iv] +=
      -f[iv]*dyw+g[iv]*dxw -(f[iv]*dys-g[iv]*dxs);
    sumflx[ir+1][icm1].w[iv] +=
      f[iv]*dyn-g[iv]*dxn + (f[iv]*dyw-g[iv]*dxw);
    sumflx[ir][icm1].w[iv] +=
      f[iv]*dyn-g[iv]*dxn -(f[iv]*dys-g[iv]*dxs);
  }
}


  /* Interior Nodes */

for(ir=1; ir<n_r-1; ir++)
  for(ic=1; ic<n_c-1; ic++){
    in = input[ir][ic].w;
    u = in[1] / in[0];
    v = in[2] / in[0];

    /**** Calculate the pressure at this node ****/
      p = GMA1*(in[3] - 0.5*(in[1]*u + in[2]*v));

    /**** Calculate flux vectors ****/
    f[0] = in[1];
    /*f[0] = grid[ir][ic].x+grid[ir][ic].y;*/
    f[1] = in[1]*u+p;
    f[2] = g[1] = in[1]*v;
    f[3] = (in[3]+p)*u;
    g[0] = in[2];
    /*g[0] = grid[ir][ic].x+grid[ir][ic].y; */
    g[2] = in[2]*v+p;
    g[3] = (in[3]+p)*v;


    /**** Store pressure for later use ****/
    input[ir][ic].p = p;

    /* Calculate length of faces for flux distribution.  Note that
       differences are always taken with current node second (ie. on a
       uniform grid dxe = -dxw and dxs = -dxn). Also note that flux across
       each face is average of its endpoints, so a half face length is
       associated with each */

    icm1=ic-1;
    icp1=ic+1;



    dxw =0.5*( grid[ir][ic].x-grid[ir][icm1].x);
    dyw =0.5*( grid[ir][ic].y-grid[ir][icm1].y);

    dxe = 0.5*(grid[ir][icp1].x - grid[ir][ic].x);
    dye = 0.5*(grid[ir][icp1].y - grid[ir][ic].y);

    dxn = 0.5*(grid[ir+1][ic].x - grid[ir][ic].x);
    dyn = 0.5*(grid[ir+1][ic].y - grid[ir][ic].y);
    dxs = 0.5*(grid[ir-1][ic].x - grid[ir][ic].x);
    dys = 0.5*(grid[ir-1][ic].y - grid[ir][ic].y);
```

```
      /**** Distribute each member of flux vector associated with this node
        to its neighbors ****/


      for(iv=0;iv<4;iv++){

          sumflx[ir-1][icm1].w[iv] +=
            -f[iv]*dyw+g[iv]*dxw -(f[iv]*dys-g[iv]*dxs);
          sumflx[ir+1][icm1].w[iv] +=
            f[iv]*dyn-g[iv]*dxn + (f[iv]*dyw-g[iv]*dxw);
          sumflx[ir][icm1].w[iv] +=
            f[iv]*dyn-g[iv]*dxn -(f[iv]*dys-g[iv]*dxs);
          sumflx[ir-1][icp1].w[iv] +=
            f[iv]*dys-g[iv]*dxs - (f[iv]*dye-g[iv]*dxe);
          sumflx[ir+1][icp1].w[iv] +=
            f[iv]*dye-g[iv]*dxe - (f[iv]*dyn-g[iv]*dxn);
          sumflx[ir][icp1].w[iv] +=
            f[iv]*dys-g[iv]*dxs - (f[iv]*dyn-g[iv]*dxn);
          sumflx[ir-1][ic].w[iv] +=
            -f[iv]*dyw+g[iv]*dxw -(f[iv]*dye-g[iv]*dxe);
          sumflx[ir+1][ic].w[iv] +=
            f[iv]*dye-g[iv]*dxe + (f[iv]*dyw-g[iv]*dxw);
      }

    }


#ifndef MOVING_GRID
  c_flag = 1;
#endif



}

/**********************    VFLUX   *******************************/
/*~~~~ This function calculates viscous fluxes associated with a given
 *cell* and distributes them to nodes on its corners ~~~~*/
void Calc_VFlux(int n_r, int n_c, struct point **grid, struct node **input,
                struct flux **vflux, float Re, struct a_cell **area,
                float **tstep, int wfact, float tott)
{
  int ir, ic, icp1, icp2, iv, ivp1, icm1, icm2;
  double
    dxds,dxdn,dyds,dydn,ijac,lwrxc,lwryc,dxw,dyw,dxn,dyn,dxe,dye,dxs,dys,
    lwrm1xc,lwrm1yc,jac,lwrp1xc,lwrp1yc,p;
  double
    duds,dudn,dvds,dvdn,dtds,dtdn,dudx,dudy,dvdx,dvdy,dtdx,dtdy,u,v,rho,txx,
    txy,tyy,f[3],g[3],a,Re_x,Re_y,a_temp,u_hat,v_hat,t,TtT1,mut,mugp,T0,S,mu0;
#ifdef AV4_COEF
  float av4_const,pderiv;
#endif
#ifdef AV2_COEF
  float av2_const;
#endif
#ifndef MU
  float MU;
#endif

  struct point *gr, *grp1;
  struct node  *nd, *nd1, *nd2;
  struct temp{
    float u,v,t;
    double xc,yc;
  }*bot, *top;

static float qcoeff,tcoeff_d,tcoeff_od;
static struct temp *etemp,*otemp;
static int c_flag=0;

 TtT1=2.0/(GMA+1.0);
/* Properties of Nitrogen */
 mu0=1.663e-5;
 T0=273;
 S=107;

if(!c_flag){
  /**** Allocate space for working vectors if this is first iteration ****/
  etemp = (struct temp *)malloc((size_t)(n_c*sizeof(struct temp)));
  otemp = (struct temp *)malloc((size_t)(n_c*sizeof(struct temp)));


  /**** Set a flag to signify that static variables are set ****/
  c_flag = 1;
}

/**** Calcluate bottom working vector for first iteration ****/
nd = input[0];
gr=grid[0];
grp1= grid[1];

for(ic=0; ic<n_c-1; ic++){
  etemp[ic].u = nd[ic].w[1]/nd[ic].w[0];
  etemp[ic].v = nd[ic].w[2]/nd[ic].w[0];
  etemp[ic].t = GMA*nd[ic].p/nd[ic].w[0];
  etemp[ic].xc = 0.25*(gr[ic].x+gr[ic+1].x+grp1[ic].x+grp1[ic+1].x);
  etemp[ic].yc = 0.25*(gr[ic].y+gr[ic+1].y+grp1[ic].y+grp1[ic+1].y);
}
etemp[n_c-1].u = nd[n_c-1].w[1]/nd[n_c-1].w[0];
etemp[n_c-1].v = nd[n_c-1].w[2]/nd[n_c-1].w[0];
etemp[n_c-1].t = GMA*nd[n_c-1].p/nd[n_c-1].w[0];
etemp[n_c-1].xc = 0.5*(gr[n_c-1].x+grp1[n_c-1].x);
etemp[n_c-1].yc = 0.5*(gr[n_c-1].y+grp1[n_c-1].y);

/**** Save minimum time step from last iteration for AV terms ****/
tstep[0][1] = tstep[0][0];

/***** Initialize minimum time step for this iteration ****/
tstep[0][0] = 8888;

for(ir=0; ir<n_r-1; ir++){

  /*** Bottom working vector of previous row becomes top wv for this row***/
  if(!(ir%2)){
    bot = etemp;
    top = otemp;
  }else{
    bot = otemp;
    top = etemp;
  }

  gr  =grid[ir];
  grp1=grid[ir+1];
  nd=input[ir];
  nd1=input[ir+1];
  nd2=input[ir+2];

  top[0].u=nd1[0].w[1]/nd1[0].w[0];
  top[0].v=nd1[0].w[2]/nd1[0].w[0];
  top[0].t=nd1[0].p/nd1[0].w[0];
```

```
for(ic=0;ic<n_c;ic++){

  icp1=ic+1;
  icm1=ic-1;

  if(ic <n_c-1) {
    top[icp1].u = nd1[icp1].w[1]/nd1[icp1].w[0];
    top[icp1].v = nd1[icp1].w[2]/nd1[icp1].w[0];
    top[icp1].t = GMA*nd1[icp1].p/nd1[icp1].w[0];
  }

  /**** Save centroid location of cell below current cell before it is
    overwritten by centroid location of cell above current cell ****/
  if(ir>0){
    if(ic>0){
      lwrm1xc = lwrxc;
      lwrm1yc = lwryc;
    }else{
      lwrm1xc =0;
      lwrm1yc =0;
    }

    lwrxc = top[ic].xc;
    lwryc = top[ic].yc;

  }


    if(ir<n_r-2 && ic < n_c-1) {
      top[ic].xc = 0.25*(grid[ir+2][ic].x+grid[ir+2][icp1].x+
                         grp1[ic].x+grp1[icp1].x);
      top[ic].yc = 0.25*(grid[ir+2][ic].y+grid[ir+2][icp1].y+
                         grp1[ic].y+grp1[icp1].y);

    } else if(ic > n_c-2 && ir<n_r-2) {
      top[ic].xc = 0.5*(grid[ir+2][ic].x+grp1[ic].x);
      top[ic].yc = 0.5*(grid[ir+2][ic].y+grp1[ic].y);
    }


    /**** Calculate the geometric parameters for the derivatives.  Note
      that 0.5 coefs cancel in final products, so they are dropped ****/

    /* Calculate the geometric derivatives at the pseudonodes*/


    if(ic < n_c-1) {

      dxds = grp1[icp1].x+gr[icp1].x - (gr[ic].x+grp1[ic].x);
      dxdn = grp1[ic].x+grp1[icp1].x - (gr[ic].x+gr[icp1].x);
      dyds = grp1[icp1].y+gr[icp1].y - (gr[ic].y+grp1[ic].y);
      dydn = grp1[ic].y+grp1[icp1].y - (gr[ic].y+gr[icp1].y);

    } else {
      dxds=(grp1[ic].x+gr[ic].x)-(grp1[icm1].x+gr[icm1].x);
      dyds=(grp1[ic].y+gr[ic].y)-(grp1[icm1].y+gr[icm1].y);
      dxdn=(grp1[ic].x+grp1[icm1].x)-(gr[ic].x+gr[icm1].x);
      dydn=(grp1[ic].y+grp1[icm1].y)-(gr[ic].y+gr[icm1].y);
    }

    /*  printf("%i %i Distribute Flux \n",ic,ir); */
    /**** Calculate Jacobian ****/
    jac = dxds*dydn - dxdn*dyds;

       if(jac<0) printf("Negative Jacobian! %i %i\n",ir,ic);

    /* Calculate length of faces for flux distribution */

    if(ir<n_r-1){
      dxn = 0.5*(top[ic].xc - bot[ic].xc);
      dyn = 0.5*(top[ic].yc - bot[ic].yc);
    }
    else{
      dxn=0;
      dyn=0;
    }

    if(ic>0){
      dxw =0.5*(bot[ic].xc-bot[icm1].xc) ;
      dyw =0.5*(bot[ic].yc-bot[icm1].yc) ;
    }else {
      dxw =0.0;
      dyw =0.0;
    }

    if(ir>0){
      dxs = 0.5*(lwrxc - bot[ic].xc);
      dys = 0.5*(lwryc - bot[ic].yc);
    }else
      dxs=dys=0;

    if(ic< n_c-1){
      dxe = 0.5*(bot[icp1].xc - bot[ic].xc);
      dye = 0.5*(bot[icp1].yc - bot[ic].yc);
    }else{
      dxe=0.0;
      dye=0.0;
    }

    /**** Invert Jacobian for later convenience ****/
    ijac = 1/jac;

    /**** Calculate flow derivatives for this cell */

    if(ic < n_c-1){

      duds = nd1[icp1].w[1]/nd1[icp1].w[0]+nd[icp1].w[1]/nd[icp1].w[0]
        - (nd1[ic].w[1]/nd1[ic].w[0]+nd[ic].w[1]/nd[ic].w[0]);
      dudn = nd1[ic].w[1]/nd1[ic].w[0]+nd1[icp1].w[1]/nd1[icp1].w[0]
        - (nd[ic].w[1]/nd[ic].w[0]+nd[icp1].w[1]/nd[icp1].w[0]);
      dvds = nd1[icp1].w[2]/nd1[icp1].w[0]+nd[icp1].w[2]/nd[icp1].w[0]
        - (nd1[ic].w[2]/nd1[ic].w[0]+nd[ic].w[2]/nd[ic].w[0]);
      dvdn = nd1[ic].w[2]/nd1[ic].w[0]+nd1[icp1].w[2]/nd1[icp1].w[0]
        - (nd[ic].w[2]/nd[ic].w[0]+nd[icp1].w[2]/nd[icp1].w[0]);
      dtds = nd1[icp1].p/nd1[icp1].w[0]+nd[icp1].p/nd[icp1].w[0]
        - (nd1[ic].p/nd1[ic].w[0]+nd[ic].p/nd[ic].w[0]);
      dtdn = nd1[ic].p/nd1[ic].w[0]+nd1[icp1].p/nd1[icp1].w[0]
        - (nd[ic].p/nd[ic].w[0]+nd[icp1].p/nd[icp1].w[0]);
      dudx = ijac*(dydn*duds-dyds*dudn);
      dudy = ijac*(dxds*dudn-dxdn*duds);
      dvdx = ijac*(dydn*dvds-dyds*dvdn);
      dvdy = ijac*(dxds*dvdn-dxdn*dvds);
      dtdx = ijac*(dydn*dtds-dyds*dtdn);
      dtdy = ijac*(dxds*dtdn-dxdn*dtds);

      /**** Take flow speed as average over four cell nodes ****/
      u  = 0.25*(nd1[icp1].w[1]/nd1[icp1].w[0]+nd[icp1].w[1]/nd[icp1].w[0]
                 + nd1[ic].w[1]/nd1[ic].w[0]+nd[ic].w[1]/nd[ic].w[0]);
      v  = 0.25*(nd1[icp1].w[2]/nd1[icp1].w[0]+nd[icp1].w[2]/nd[icp1].w[0]
                 + nd1[ic].w[2]/nd1[ic].w[0]+nd[ic].w[2]/nd[ic].w[0]);
```

```
        } else{

          duds = nd1[ic].w[1]/nd1[ic].w[0]+nd[ic].w[1]/nd[ic].w[0]
            - (nd1[icm1].w[1]/nd1[icm1].w[0]+nd[icm1].w[1]/nd[icm1].w[0]);
          dudn = nd1[ic].w[1]/nd1[ic].w[0]+nd1[icm1].w[1]/nd1[icm1].w[0]
            - (nd[ic].w[1]/nd[ic].w[0]+nd[icm1].w[1]/nd[icm1].w[0]);
          dvds = nd1[ic].w[2]/nd1[ic].w[0]+nd[ic].w[2]/nd[ic].w[0]
            - (nd1[icm1].w[2]/nd1[icm1].w[0]+nd[icm1].w[2]/nd[icm1].w[0]);
          dvdn = nd1[ic].w[2]/nd1[ic].w[0]+nd1[icm1].w[2]/nd1[icm1].w[0]
            - (nd[ic].w[2]/nd[ic].w[0]+nd[icm1].w[2]/nd[icm1].w[0]);
          dtds = nd1[ic].p/nd1[ic].w[0]+nd[ic].p/nd[ic].w[0]
            - (nd1[icm1].p/nd1[icm1].w[0]+nd[icm1].p/nd[icm1].w[0]);
          dtdn = nd1[ic].p/nd1[ic].w[0]+nd1[icm1].p/nd1[icm1].w[0]
            - (nd[ic].p/nd[ic].w[0]+nd[icm1].p/nd[icm1].w[0]);

          dudx = ijac*(dydn*duds-dyds*dudn);
          dudy = ijac*(dxds*dudn-dxdn*duds);
          dvdx = ijac*(dydn*dvds-dyds*dvdn);
          dvdy = ijac*(dxds*dvdn-dxdn*dvds);
          dtdx = ijac*(dydn*dtds-dyds*dtdn);
          dtdy = ijac*(dxds*dtdn-dxdn*dtds);

          /**** Take flow speed as average over two cell nodes ****/
          u  = 0.5*(nd1[ic].w[1]/nd1[ic].w[0]+nd[ic].w[1]/nd[ic].w[0]);
          v  = 0.5*(nd1[ic].w[2]/nd1[ic].w[0]+nd[ic].w[2]/nd[ic].w[0]);
        }

      /**** Pre-calculate coefficients of shear and heat flux terms ****/
#ifndef MU


      rho=input[ir][ic].w[0];
      u=input[ir][ic].w[1]/rho;
      v=input[ir][ic].w[2]/rho;
      p = GMA1*(input[ir][ic].w[3] - 0.5*rho*(u*u+v*v));
      t = GMA*p/rho;
      mugp=mu0*pow((tott*TtT1*t/T0),1.5)*(T0+S)/(TtT1*t*tott+S);
      mut=mu0*pow((tott*TtT1/T0),1.5)*(T0+S)/(TtT1*tott+S);

        MU=mugp/mut;

#endif
        qcoeff = -MU/(GMA1*Re*PR);
        tcoeff_d  = -2*MU/(3*Re);
        tcoeff_od = -MU/Re;

        /**** Calculate stresses and flux vectors at pseudonodes ****/
        txx = tcoeff_d*(2*dudx-dvdy);
        txy = tcoeff_od*(dudy+dvdx);
        tyy = tcoeff_d*(2*dvdy-dudx);

        f[0] = txx;
        /*f[0] = bot[ic].xc+bot[ic].yc;*/
        f[1] = txy;
        f[2] = u*txx+v*txy+qcoeff*dtdx;
        g[0] = txy;
        /*g[0] = bot[ic].xc+bot[ic].yc;*/
        g[1] = tyy;
        g[2] = u*txy+v*tyy+qcoeff*dtdy;



        /**** Calculate time step according to CFL condition ****/
        if(ir>0){
```

```
          if(ic>0){
            dxdn = 0.5*(bot[icm1].xc+bot[ic].xc-(lwrm1xc+lwrxc));
            dxds = 0.5*(bot[ic].xc+lwrxc-(bot[icm1].xc+lwrm1xc));
            dydn = 0.5*(bot[icm1].yc+bot[ic].yc-(lwrm1yc+lwryc));
            dyds = 0.5*(bot[ic].yc+lwryc-(bot[icm1].yc+lwrm1yc));
            if(ic > n_c-2){
              dxds=dxds*0.5;
              dyds=dyds*0.5;
            }


            jac = dxds*dydn - dxdn*dyds;
            rho=input[ir][ic].w[0];
            u = input[ir][ic].w[1]/rho;
            v = input[ir][ic].w[2]/rho;
            p=GMA1*(input[ir][ic].w[3]
                    -0.5*(input[ir][ic].w[1]*u+input[ir][ic].w[2]*v));
            u_hat=u*dydn-v*dxdn;
            v_hat=v*dxds-u*dyds;
            u_hat = ABS(u_hat);
            v_hat = ABS(v_hat);
            a = sqrt(GMA*p/rho);
            Re_x = rho*(u_hat+a);
            Re_y = rho*(v_hat+a);
            a_temp = dxds*dxdn+dydn*dyds;
            tstep[ir][ic] = 2.8284*ABS(jac)/
              (TS_SF*(1+2/MIN(Re_x,Re_y))*
               (u_hat+v_hat+a*sqrt(dxds*dxds+dyds*dyds+dxdn*dxdn+dydn*dydn
                                   +2*ABS(a_temp))));

            tstep[0][0] = MIN(tstep[0][0],tstep[ir][ic]);
          }
        }


#ifdef AV4_COEF
pderiv=0.01;
#ifdef UNSTEADY
      av4_const = AV4_COEF*area[ir][ic].visc/tstep[0][1];
#else
      av4_const = AV4_COEF*area[ir][ic].visc/tstep[ir][ic];
#endif
#endif



        icm2=ic-2;
        icp2=ic+2;

#ifdef AV2_COEF
#ifdef UNSTEADY
        av2_const = residual*AV2_COEF*area[ir][ic].visc/tstep[0][1];
#else
        av2_const = residual*AV2_COEF*area[ir][ic].visc/tstep[ir][ic];
#endif
#endif
        /**** Distribute each term in flux vector to appropriate nodes.  Note
          that viscous flux for first term in state vector is zero, so vectors
          are offset by one ****/

        for(iv=0;iv<3;iv++){
          ivp1 = iv+1;

          vflux[ir][ic].w[ivp1] += -f[iv]*dyw+g[iv]*dxw -
            (f[iv]*dys-g[iv]*dxs);
```

```
          vflux[ir+1][ic].w[ivp1] += f[iv]*dyn-g[iv]*dxn +
            (f[iv]*dyw-g[iv]*dxw);

          if(ic < n_c-1) {
            vflux[ir][icp1].w[ivp1] += f[iv]*dys-g[iv]*dxs -
              (f[iv]*dye-g[iv]*dxe);
            vflux[ir+1][icp1].w[ivp1] += f[iv]*dye-g[iv]*dxe -
              (f[iv]*dyn-g[iv]*dxn);
          }


#ifdef AV4_COEF
          /**** Apply fourth-order artificial viscosity ****/
          if(ir>0 && !(iv%2)){
            if(ir>1 && ir<n_r-2){

              vflux[ir][ic].w[iv] += pderiv*av4_const*(input[ir+2][ic].w[iv]-
                           4*input[ir+1][ic].w[iv]+6*input[ir][ic].w[iv]-
                           4*input[ir-1][ic].w[iv]+input[ir-2][ic].w[iv]);
              vflux[ir][ic].w[ivp1] +=pderiv*av4_const*
                 (input[ir+2][ic].w[ivp1]-4*input[ir+1][ic].w[ivp1]
                  +6*input[ir][ic].w[ivp1]-4*input[ir-1][ic].w[ivp1]
                  +input[ir-2][ic].w[ivp1]);
            }else{
              if(ir==1){

                vflux[ir][ic].w[iv]+=pderiv*av4_const*
                   (input[3][ic].w[iv]-4*input[2][ic].w[iv]
                    +6*input[1][ic].w[iv]-3*input[0][ic].w[iv]);
                vflux[ir][ic].w[ivp1]+=pderiv*av4_const*
                   (input[3][ic].w[ivp1] - 4*input[2][ic].w[ivp1]
                    +6*input[1][ic].w[ivp1]-3*input[0][ic].w[ivp1]);
              }
              else if(ir==n_r-2){

                vflux[ir][ic].w[iv]+=pderiv*av4_const*
                   (6*input[ir][ic].w[iv]-3*input[ir+1][ic].w[iv]
                    -4*input[ir-1][ic].w[iv]+input[ir-2][ic].w[iv]);
                vflux[ir][ic].w[ivp1]+=pderiv*av4_const*
                   (6*input[ir][ic].w[ivp1]-3*input[ir+1][ic].w[ivp1]
                    -4*input[ir-1][ic].w[ivp1]+input[ir-2][ic].w[ivp1]);
              }
            }

            if (ic>1 && ic<n_c-2) {

              vflux[ir][ic].w[iv] += pderiv*av4_const*
                (input[ir][icp2].w[iv]-4*input[ir][icp1].w[iv]
                 +6*input[ir][ic].w[iv]-4*input[ir][icm1].w[iv]
                 +input[ir][icm2].w[iv]);
              vflux[ir][ic].w[ivp1] += pderiv*av4_const*
                (input[ir][icp2].w[ivp1]-4*input[ir][icp1].w[ivp1]
                 +6*input[ir][ic].w[ivp1]-4*input[ir][icm1].w[ivp1]
                 +input[ir][icm2].w[ivp1]);
            } else if(ic == 1){

              vflux[ir][ic].w[iv]+=pderiv*av4_const*
                (input[ir][3].w[iv]-
                 4*input[ir][2].w[iv]+6*input[ir][1].w[iv]-
                 3*input[ir][0].w[iv]);
              vflux[ir][ic].w[ivp1]+=pderiv*av4_const*
                (input[ir][3].w[ivp1]-
                 4*input[ir][2].w[ivp1]+6*input[ir][1].w[ivp1]-
                 3*input[ir][0].w[ivp1]);
```

```
            } else if(ic == n_c-2){

              vflux[ir][ic].w[iv]+=pderiv*av4_const*
                (6*input[ir][ic].w[iv]
                 -3*input[ir][ic+1].w[iv]-4*input[ir][ic-1].w[iv]+
                                          input[ir][ic-2].w[iv]);
              vflux[ir][ic].w[ivp1]+=pderiv*av4_const*
                (6*input[ir][ic].w[ivp1]
                 -3*input[ir][ic+1].w[ivp1]-4*input[ir][ic-1].w[ivp1]+
                 input[ir][ic-2].w[ivp1]);


            } else if(ic == n_c-1){

/*
              vflux[ir][ic].w[iv]+=pderiv*av4_const*
                (-2*input[ir][ic].w[iv]-12*input[ir][ic-1].w[iv]
                 +36*input[ir][ic-2].w[iv]
                 -28*input[ir][ic-3].w[iv]+6*input[ir][ic-4].w[iv]);
              vflux[ir][ic].w[ivp1]+=pderiv*av4_const*
                (-2*input[ir][ic].w[ivp1]
                 -12*input[ir][ic-1].w[ivp1]+36*input[ir][ic-2].w[ivp1]
                 -28*input[ir][ic-3].w[ivp1]+6*input[ir][ic-4].w[ivp1]);
*/
              /*  vflux[ir][ic].w[iv]+=pderiv*av4_const*
                (input[ir][ic].w[iv]-4*input[ir][ic-1].w[iv]
                 +6*input[ir][ic-2].w[iv]+
                 -4*input[ir][ic-3].w[iv]+4*input[ir][ic-4].w[iv]);
              vflux[ir][ic].w[ivp1]+=pderiv*av4_const*
                (input[ir][ic].w[ivp1]
                 -4*input[ir][ic-1].w[ivp1]+6*input[ir][ic-2].w[ivp1]+
                 -4*input[ir][ic-3].w[ivp1]+4*input[ir][ic-4].w[ivp1]);*/
            }

#ifdef AV2_COEF
            vflux[ir][ic].w[iv] += av2_const*
              (input[ir-1][ic].w[iv]-
               4*input[ir][ic].w[iv]+input[ir+1][ic].w[iv]
               +input[ir][icm1].w[iv]+input[ir][icp1].w[iv]);
            vflux[ir][ic].w[ivp1] += av2_const*
              (input[ir-1][ic].w[ivp1]-
               4*input[ir][ic].w[ivp1]+input[ir+1][ic].w[ivp1]+
               input[ir][icm1].w[ivp1]+input[ir][icp1].w[ivp1]);
#endif

          }

#endif

        }
    }
  }


}



/*********************    Cell Areas    ******************************/
/*~~~~ This function calculates the cell area associated with each node.
  This area includes all four cells which include each node ~~~~*/
void Calc_Areas(int n_r, int n_c, struct point **grid, struct a_cell **area)
{
```

```
int ir, ic, icm1, icp1;
double a,av;
struct a_cell *ar, *arp1;
struct point *gr, *grp1, *etemp, *otemp, *curr, *prev;

etemp = (struct point *)malloc((size_t)(n_c*sizeof(struct point)));
otemp = (struct point *)malloc((size_t)(n_c*sizeof(struct point)));

for(ir=0;ir<n_r;ir++)
  for(ic=0;ic<n_c;ic++)
    area[ir][ic].invs = 0;

for(ir=0;ir<n_r-1;ir++){

  if(ir%2){
    prev = etemp;
    curr = otemp;
  }else{
    prev = otemp;
    curr = etemp;
  }

  gr = grid[ir];
  grp1 = grid[ir+1];
  ar   = area[ir];
  arp1 = area[ir+1];

  for(ic=0;ic<n_c;ic++){

    icm1 = ic-1;
    icp1 = ic+1;

   /**** Calculate area of cell which this node is lower left corner ****/

    if(ic< n_c-1){
      a = 0.5*((grp1[ic+1].x-gr[ic].x)*(grp1[ic].y-gr[ic+1].y)-
               (grp1[ic].x-gr[ic+1].x)*(grp1[ic+1].y-gr[ic].y));
    } else {
      a=0.0;
    }

    a=ABS(a);

    /**** Add this area to all nodes which make up cell ****/

    ar[ic].invs += a;
    arp1[ic].invs += a;

    if(ic < n_c-1){
      ar[icp1].invs += a;
      arp1[icp1].invs += a;
    }


    if(ic<n_c-1){
      curr[ic].x = 0.25*(gr[ic].x+gr[ic+1].x+grp1[ic].x+grp1[ic+1].x);
      curr[ic].y = 0.25*(gr[ic].y+gr[ic+1].y+grp1[ic].y+grp1[ic+1].y);
    } else {
      curr[ic].x = 0.5*(gr[ic].x+grp1[ic].x);
      curr[ic].y = 0.5*(gr[ic].y+grp1[ic].y);
    }

    if( ic > 0 ){
      if(ir>0 && ir<n_r-1){
        av = 0.5*((curr[ic].x-prev[icm1].x)*(curr[icm1].y-prev[ic].y)-
                  (curr[icm1].x-prev[ic].x)*(curr[ic].y-prev[icm1].y));
        ar[ic].visc = ABS(av);

      }
      }else{
        ar[ic].visc =0.0;
      }
    }
  }

}


/**********************    Intermediate State Update **********************/
/*~~~~ This function calculates a new value for intermediate state vector
  based on fluxes, actual state vector, and a weight factor for RK scheme~~~~*/
void IUpdate(int n_r, int n_c, struct node **istate, struct node **state,
             struct flux **iflux,  struct flux **vflux, struct a_cell **area,
             float **tstep, struct point **grid, double theta_step, int wfact,
             float Re, struct node *ff, struct locale *te, struct locale *ts,
             struct locale *ne, struct locale *ns)
{
  int ir, ic, iv,icp1,icm1;
  double jr_rat, icoef, vcoef, rstep;
  float theta,xt,yt,ijac1,xs,ys,xe,ye,jac2,*ist,dtp,dtm,ds,sumdt;
  float Jp,Jm,cv,ra,a,a0,xn,yn,rho0,u0,a1,a2, uout,q,ptp1,TtT1,rtr1,v0;
  float ewall,uff,u,v,re1,re2,rho;

  ptp1=pow(2.0/(GMA+1),GMA/(GMA-1.0));
  TtT1=2.0/(GMA+1.0);
  rtr1=1.0/pow((GMA+1)/2.0,1/(GMA-1));


#ifdef ADIABATIC
  float oldjrho,oldbrho;
  oldjrho = state[0][n_c-1].w[0];
  oldbrho = state[n_r-1][n_c-1].w[0];
#else
  ewall= 1/(GMA*GMA1)/TtT1;
#endif
  rstep = 1.0/(n_r-1);



 if(wfact==4 ) {
    for(ir=0;ir<n_r;ir++)
      for(ic=0;ic<n_c;ic++){
        for(iv=0;iv<4;iv++){
          istate[ir][ic].w[iv]=state[ir][ic].w[iv];

        }
        istate[ir][ic].p=(istate[ir][ic].w[3]
                          -0.5*(state[ir][ic].w[1]*state[ir][ic].w[1]+
                                state[ir][ic].w[2]*state[ir][ic].w[2])
                          /state[ir][ic].w[0])*GMA1;
      }
  }

  /**** Calculate state vector for interior nodes ****/
  for(ir=1;ir<n_r-1;ir++)
    for(ic=1;ic<n_c;ic++){

#ifdef UNSTEADY
      icoef = -tstep[0][0]/wfact/area[ir][ic].invs;
      vcoef = -tstep[0][0]/wfact/area[ir][ic].visc;
```

```
#else
      icoef = -tstep[ir][ic]/wfact/area[ir][ic].invs;
      vcoef = -tstep[ir][ic]/wfact/area[ir][ic].visc;
#endif
      ist=istate[ir][ic].w;
      for(iv=0;iv<4;iv++){



        ist[iv]=state[ir][ic].w[iv]+icoef*iflux[ir][ic].w[iv] +
          vcoef*vflux[ir][ic].w[iv];


        /**** Intialize fluxes for next iteration ****/
        iflux[ir][ic].w[iv] = 0;

        u0=state[ir][ic].w[1]/state[ir][ic].w[0];
        v0=state[ir][ic].w[2]/state[ir][ic].w[0];
        istate[ir][ic].p=(istate[ir][ic].w[3]
                          -0.5*(state[ir][ic].w[1]*u0+
                                state[ir][ic].w[2]*v0))*GMA1;
      }
    }

  for(ir=1;ir<n_r-1;ir++){

    /* Left (Inflow) BC */

    /* Nozzle Flow */
    a0=sqrt(GMA*ff[ir].p/ff[ir].w[0]);
    a=sqrt(GMA*state[ir][1].p/state[ir][1].w[0]);
    Jm=state[ir][1].w[1]/state[ir][1].w[0]-2/GMA1*a;
    Jp=2/GMA1*a0+ff[ir].w[1];


    u0=0.5*(Jm+Jp);
    v0=0.0;

    a=(Jp-Jm)*GMA1/4.0;
    istate[ir][0].w[0]=ff[ir].w[0]*pow((a/a0),2./GMA1);
    istate[ir][0].p=ff[ir].p/pow(ff[ir].w[0],GMA)
      *pow(istate[ir][0].w[0],GMA);
    istate[ir][0].w[1]=u0*istate[ir][0].w[0];
    istate[ir][0].w[2]=v0*istate[ir][0].w[0];
    istate[ir][0].w[3]=istate[ir][0].p/GMA1+0.5*(istate[ir][0].w[1]*u0+
                                                 istate[ir][0].w[2]*v0);

    /* Right BC */
    /* Nozzle Flow */


  }

  /* Compute new density on solid surfaces and use it to correct state
     vector there */

  /* Inflow Lower Wall */
  re1=istate[1][0].p/GMA1;
  re2=istate[2][0].p/GMA1;
  istate[0][0].w[0]=(te[0].l/ts[0].l/GMA1
                     *(3*istate[0][0].p-4*istate[0][1].p+istate[0][2].p)
                     +4*re1-re2)/3/ewall;
  istate[0][0].w[3]=istate[0][0].w[0]*ewall;
  istate[0][0].w[1]=0.0;
  istate[0][0].w[2]=0.0;

  /*Inflow Upper Wall */
  re1=istate[n_r-2][0].p/GMA1;
  re2=istate[n_r-3][0].p/GMA1;
  istate[n_r-1][0].w[0]=(te[0].u/ts[0].u/GMA1
                         *(3*istate[n_r-1][0].p-4*istate[n_r-1][1].p
                           +istate[n_r-1][2].p)+4*re1-re2)/3/ewall;
  istate[n_r-1][0].w[3]=istate[n_r-1][0].w[0]*ewall;
  istate[n_r-1][0].w[1]=0.0;
  istate[n_r-1][0].w[2]=0.0;

  /*Outflow Lower Wall */
  re1=istate[1][n_c-1].p/GMA1;
  re2=istate[2][n_c-1].p/GMA1;
  istate[0][n_c-1].w[0]=(te[n_c-1].l/ts[n_c-1].l/GMA1
                         *(3*istate[0][n_c-1].p-4*istate[0][n_c-2].p
                           +istate[0][n_c-3].p)+4*re1-re2)/3/ewall;
  istate[0][n_c-1].w[3]=istate[0][n_c-1].w[0]*ewall;
  istate[0][n_c-1].w[1]=0.0;
  istate[0][n_c-1].w[2]=0.0;

  /* Outflow Upper Wall */
  re1=istate[n_r-2][n_c-1].p/GMA1;
  re2=istate[n_r-3][n_c-1].p/GMA1;
  istate[n_r-1][n_c-1].w[0]=(te[n_c-1].u/ts[n_c-1].u/GMA1
                         *(3*istate[n_r-1][n_c-1].p-4*istate[n_r-1][n_c-2].p
                           +istate[n_r-1][n_c-3].p)+4*re1-re2)/3/ewall;
  istate[n_r-1][n_c-1].w[3]=istate[n_r-1][n_c-1].w[0]*ewall;
  istate[n_r-1][n_c-1].w[1]=0.0;
  istate[n_r-1][n_c-1].w[2]=0.0;



  for(ic=1;ic<n_c-1;ic++){

    icm1=ic-1;
    icp1=ic+1;


    /*   Bottom  Wall BC */
    rho=state[1][ic].w[0];
    u=state[1][ic].w[1]/rho;
    v=state[1][ic].w[2]/rho;
    re1=state[1][ic].w[3]-0.5*rho*(u*u+v*v);
    rho=state[2][ic].w[0];
    u=state[2][ic].w[1]/rho;
    v=state[2][ic].w[2]/rho;
    re2=state[2][ic].w[3]-0.5*rho*(u*u+v*v);
    istate[0][ic].w[0]=(te[ic].l/ts[ic].l/GMA1
                        *(state[0][icp1].p-state[0][icm1].p)
                        +4*re1-re2)/3/ewall;
    istate[0][ic].w[3]=state[0][ic].w[0]*ewall;
    istate[0][ic].w[1]=0.0;
    istate[0][ic].w[2]=0.0;
    istate[0][ic].p=state[1][ic].w[3]*GMA1;


    /*  Top Wall BC */

    rho=state[n_r-2][ic].w[0];
    u=state[n_r-2][ic].w[1]/rho;
    v=state[n_r-2][ic].w[2]/rho;
    re1=state[n_r-2][ic].w[3]-0.5*rho*(u*u+v*v);
    rho=state[n_r-3][ic].w[0];
    u=state[n_r-3][ic].w[1]/rho;
```

```
    v=state[n_r-3][ic].w[2]/rho;
    re2=state[n_r-3][ic].w[3]-0.5*rho*(u*u+v*v);

    istate[n_r-1][ic].w[0]=(te[ic].u/ts[ic].u/GMA1
                            *(state[n_r-1][icp1].p-state[n_r-1][icm1].p)
                            +4*re1-re2)
                            /3/ewall;
    istate[n_r-1][ic].w[3]=state[n_r-1][ic].w[0]*ewall;
    istate[n_r-1][ic].w[1]=0.0;
    istate[n_r-1][ic].w[2]=0.0;
    istate[n_r-1][ic].p=state[n_r-1][ic].w[3]*GMA1;

  }

}

/**********************    Final Update   **********************/
/*~~~~ This function corrects the state vector based on the fluxes, time step,
  and cell areas ~~~~*/
  float FUpdate(int n_r, int n_c, struct node **state, struct flux **iflux,
                struct flux **vflux, struct a_cell **area, float **tstep,
                double theta_step, struct point **grid, float Re,
                struct node *ff, struct locale *te, struct locale *ts,
                struct locale *ne, struct locale *ns, float resm)
{
  int ir, ic, iv,icp1,icm1;
  double icoef, vcoef, e_j, e_b, rstep;
  float res=0,oldxmom,theta,xt,yt,ijac1,xs,ys,xe,ye,jac2,oldxmass=0.0;
  float ra,Jp,Jm,a,a0,dtp,dtm,ds,sumdt,cv,xn,yn,u0,a1,uout,a2,q,resmax=0;
  float rtr1,TtT1,ptp1,ewall,count,v0,uff,u,v,p,re1,re2,rho;
  struct node *jsurf, *bsurf;

  ptp1=pow(2.0/(GMA+1),GMA/(GMA-1.0));
  TtT1=2.0/(GMA+1);
  rtr1=1.0/pow((GMA+1)/2.0,1/(GMA-1));

#ifdef ADIABATIC
  float oldjrho,oldbrho;
  oldjrho = state[0][n_c-1].w[0];
  oldbrho = state[n_r-1][n_c-1].w[0];
#else
   ewall= 1.0/(GMA*GMA1)/TtT1;
#endif

  rstep = 1.0/(n_r-1);
  res=0.0;
  resm=0.0;
  count=0.0;


  /**** Update interior nodes ****/
  for(ir=1;ir<n_r-1;ir++){
    for(ic=1;ic<n_c;ic++){

#ifdef UNSTEADY
      icoef = -tstep[0][0]/area[ir][ic].invs;
      vcoef = -tstep[0][0]/area[ir][ic].visc;
#else
      icoef = -tstep[ir][ic]/area[ir][ic].invs;
      vcoef = -tstep[ir][ic]/area[ir][ic].visc;
#endif
      oldxmom = state[ir][ic].w[1];
      oldxmass = state[ir][ic].w[0];
      for(iv=0;iv<4;iv++){
        state[ir][ic].w[iv] += icoef*iflux[ir][ic].w[iv] +
            vcoef*vflux[ir][ic].w[iv];



        /**** Intitialize fluxes for next iteration ****/
        iflux[ir][ic].w[iv] = 0;
        vflux[ir][ic].w[iv] = 0;
      }

      u0=state[ir][ic].w[1]/state[ir][ic].w[0];
      v0=state[ir][ic].w[2]/state[ir][ic].w[0];
      state[ir][ic].p=(state[ir][ic].w[3]
                       -0.5*(state[ir][ic].w[1]*u0+
                             state[ir][ic].w[2]*v0))*GMA1;

      /**** Calculate sum of changes to x-momentum to monitor convergence **/
      count=count+1.0;
      res = ABS(oldxmom-state[ir][ic].w[1])+res;
      resm = ABS(oldxmass-state[ir][ic].w[0])+resm;


#ifdef UNSTEADY_AREAS
      area[ir][ic].invs = 0;
#endif
    }
  }

      res=res/count;
      resm=resm/count;
      /*printf("Avg Mass Residual: %e \n",resm);*/

  /* Correct density at solid surfaces and use it to adjust other members
     of the state vector */

      for(ir=1;ir<n_r-1;ir++){

        for(iv=0;iv<4;iv++){
          iflux[ir][0].w[iv] = 0.0;
          vflux[ir][0].w[iv] = 0.0;
          iflux[ir][n_c-1].w[iv] = 0.0;
          vflux[ir][n_c-1].w[iv] = 0.0;
        }

        /* Left BC */

        /* Nozzle Flow */
        a0=sqrt(GMA*ff[ir].p/ff[ir].w[0]);
        a=sqrt(GMA*state[ir][1].p/state[ir][1].w[0]);
        Jm=state[ir][1].w[1]/state[ir][1].w[0]-2/GMA1*a;
        Jp=2/GMA1*a0+ff[ir].w[1];

        u0=0.5*(Jm+Jp);
        v0=0.0;

        a=(Jp-Jm)*GMA1/4.0;



        state[ir][0].w[0]=ff[ir].w[0]*pow((a/a0),2./GMA1);
        state[ir][0].p=ff[ir].p/pow(ff[ir].w[0],GMA)
          *pow(state[ir][0].w[0],GMA);
        state[ir][0].w[1]=u0*state[ir][0].w[0];
        state[ir][0].w[2]=v0*state[ir][0].w[0];
        state[ir][0].w[3]=state[ir][0].p/GMA1+0.5*(state[ir][0].w[1]*u0+
                                                   state[ir][0].w[2]*v0);
```

```
      /* Right BC */
      /* Outflow is treated as supersonic, and governved by parabolic
         equations, therefore the values on the outlet plane are extrapolated
         from the interior */
    }

/* Inflow Lower Wall */
re1=state[1][0].p/GMA1;
re2=state[2][0].p/GMA1;
state[0][0].w[0]=(te[0].l/ts[0].l/GMA1
                  *(3*state[0][0].p-4*state[0][1].p+state[0][2].p)
                  +4*re1-re2)/3/ewall;
state[0][0].w[3]=state[0][0].w[0]*ewall;
state[0][0].w[1]=0.0;
state[0][0].w[2]=0.0;

/*Inflow Upper Wall */
re1=state[n_r-2][0].p/GMA1;
re2=state[n_r-3][0].p/GMA1;
state[n_r-1][0].w[0]=(te[0].u/ts[0].u/GMA1
                      *(3*state[n_r-1][0].p-4*state[n_r-1][1].p
                        +state[n_r-1][2].p)+4*re1-re2)/3/ewall;
state[n_r-1][0].w[3]=state[n_r-1][0].w[0]*ewall;
state[n_r-1][0].w[1]=0.0;
state[n_r-1][0].w[2]=0.0;

/*Outflow Lower Wall */
re1=state[1][n_c-1].p/GMA1;
re2=state[2][n_c-1].p/GMA1;
state[0][n_c-1].w[0]=(te[n_c-1].l/ts[n_c-1].l/GMA1
                      *(3*state[0][n_c-1].p-4*state[0][n_c-2].p
                     +state[0][n_c-3].p)+4*re1-re2)/3/ewall;
state[0][n_c-1].w[3]=state[0][n_c-1].w[0]*ewall;
state[0][n_c-1].w[1]=0.0;
state[0][n_c-1].w[2]=0.0;

/* Outflow Upper Wall */
re1=state[n_r-2][n_c-1].p/GMA1;
re2=state[n_r-3][n_c-1].p/GMA1;
state[n_r-1][n_c-1].w[0]=(te[n_c-1].u/ts[n_c-1].u/GMA1
                          *(3*state[n_r-1][n_c-1].p-4*state[n_r-1][n_c-2].p
                            +state[n_r-1][n_c-3].p)+4*re1-re2)/3/ewall;
state[n_r-1][n_c-1].w[3]=state[n_r-1][n_c-1].w[0]*ewall;
state[n_r-1][n_c-1].w[1]=0.0;
state[n_r-1][n_c-1].w[2]=0.0;


for(ic=1;ic<n_c-1;ic++){

  for(iv=0;iv<4;iv++){
    iflux[0][ic].w[iv] = 0.0;
    vflux[0][ic].w[iv] = 0.0;
        iflux[n_r-1][ic].w[iv] = 0.0;
    vflux[n_r-1][ic].w[iv] = 0.0;
  }

  icm1=ic-1;
  icp1=ic+1;

  rho=state[1][ic].w[0];
  u=state[1][ic].w[1]/rho;
  v=state[1][ic].w[2]/rho;
  re1=state[1][ic].w[3]-0.5*rho*(u*u+v*v);
  rho=state[2][ic].w[0];
  u=state[2][ic].w[1]/rho;
  v=state[2][ic].w[2]/rho;
      re2=state[2][ic].w[3]-0.5*rho*(u*u+v*v);
  state[0][ic].w[0]=(te[ic].l/ts[ic].l/GMA1
                     *(state[0][icp1].p-state[0][icm1].p)
                     +4*re1-re2)/3/ewall;
  state[0][ic].w[3]=state[0][ic].w[0]*ewall;
  state[0][ic].w[1]=0.0;
  state[0][ic].w[2]=0.0;
  state[0][ic].p=state[1][ic].w[3]*GMA1;

  /*  Top Wall BC */
  rho=state[n_r-2][ic].w[0];
  u=state[n_r-2][ic].w[1]/rho;
  v=state[n_r-2][ic].w[2]/rho;
  re1=state[n_r-2][ic].w[3]-0.5*rho*(u*u+v*v);
  rho=state[n_r-3][ic].w[0];
  u=state[n_r-3][ic].w[1]/rho;
  v=state[n_r-3][ic].w[2]/rho;
  re2=state[n_r-3][ic].w[3]-0.5*rho*(u*u+v*v);

  state[n_r-1][ic].w[0]=(te[ic].u/ts[ic].u/GMA1
                         *(state[n_r-1][icp1].p-state[n_r-1][icm1].p)
                         +4*re1-re2)
    /3/ewall;
  state[n_r-1][ic].w[3]=state[n_r-1][ic].w[0]*ewall;
  state[n_r-1][ic].w[1]=0.0;
  state[n_r-1][ic].w[2]=0.0;
  state[n_r-1][ic].p=state[n_r-1][ic].w[3]*GMA1;
 }

 return res;
}

#ifdef thrustcalc
float thrust(int n_r, int n_c, struct node **state, struct point **grid,
             float tott, float totp, float thrtD, float invthr, float Mie)
{
int ic,ir;
float dath, ath=0.0, symth=0.0, dsymth, totth=0.0, dtotth, ds, rho, u, v,
vmag;
float dsl, dsu, mdot=0.0, dmdot=0.0,ptp1,TtT1,rtr1,tref,pref,rref, dimthr,
dimdot;
float isp, nisp, cd, nthr;
float dath2, ath2=0.0, symth2=0.0, dsymth2, totth2=0.0, dtotth2, ds2, rho2,
u2, v2, vmag2,cfideal,etap,p2p1;
float dsl2, dsu2, mdot2=0.0, dmdot2,dstot=0.0,dstot2=0.0, f[260],videal,cf;

ptp1=pow(2.0/(GMA+1),GMA/(GMA1));
TtT1=2.0/(GMA+1.0);
rtr1=1.0/pow((GMA+1)/2.0,1/(GMA1));

ds=sqrt(pow((grid[1][n_c-1].x-grid[0][n_c-1].x),2)
        +pow((grid[1][n_c-1].y-grid[0][n_c-1].y),2));

rho=state[0][n_c-1].w[0];
u=state[0][n_c-1].w[1]/rho;
v=state[0][n_c-1].w[2]/rho;
vmag=sqrt(u*u+v*v);

dath=state[0][n_c-1].w[1]*u*ds*0.5;
dsymth=state[0][n_c-1].w[2]*v*ds*0.5;
dtotth=rho*vmag*vmag*ds*0.5;
dmdot=rho*u*ds*0.5;
```

```
ath += dath;
symth += ABS(dsymth);
totth += dtotth;
mdot += dmdot;
dstot += ds;

  for(ir=1;ir<n_r-1;ir++){

    dsl=sqrt(pow((grid[ir][n_c-1].x-grid[ir-1][n_c-1].x),2)
           +pow((grid[ir][n_c-1].y-grid[ir-1][n_c-1].y),2));
    dsu=sqrt(pow((grid[ir][n_c-1].x-grid[ir+1][n_c-1].x),2)
           +pow((grid[ir][n_c-1].y-grid[ir+1][n_c-1].y),2));

    rho=state[ir][n_c-1].w[0];
    u=state[ir][n_c-1].w[1]/rho;
    v=state[ir][n_c-1].w[2]/rho;
    vmag=sqrt(u*u+v*v);

    dath=state[ir][n_c-1].w[1]*u*(dsu+dsl)*0.5;
    dsymth=state[ir][n_c-1].w[2]*v*(dsu+dsl)*0.5;
    dtotth=rho*vmag*vmag*(dsu+dsl)*0.5;
    dmdot =rho*u*(dsu+dsl)*0.5;

    ath += dath;
    symth += ABS(dsymth);
    totth += dtotth;
    mdot += dmdot;
    dstot += 0.5*(dsl+dsu);



  }

ds=sqrt(pow((grid[n_r-1][n_c-1].x-grid[n_r-2][n_c-1].x),2)
       +pow((grid[n_r-1][n_c-1].y-grid[n_r-2][n_c-1].y),2));

rho=state[n_r-1][n_c-1].w[0];
u=state[n_r-1][n_c-1].w[1]/rho;
v=state[n_r-1][n_c-1].w[2]/rho;
vmag=sqrt(u*u+v*v);

dath=state[n_r-1][n_c-1].w[1]*u*ds*0.5;
dsymth=state[n_r-1][n_c-1].w[2]*v*ds*0.5;
dtotth=rho*vmag*vmag*ds*0.5;
dmdot =rho*u*ds*0.5;

dstot += 0.5*(dsl+dsu);
ath += dath;
symth += ABS(dsymth);
totth += dtotth;
mdot += dmdot;

printf("Axial thrust: %f   Symmetric thrust: %f \n",ath,symth);
printf("Non-D mass flow Rate: %f \n",mdot);

p2p1=pow((1+GMA1/2*Mie*Mie),-GMA/GMA1);
etap=(1-1.0/(1+GMA1/2*Mie*Mie));
videal=sqrt(2*GMA/GMA1*R/MW*tott*etap);
tref=TtT1*tott;
rref=rtr1*totp/R*MW/tott;
dimthr=ath*rref*GMA*R/MW*tref*thrtD;
dimdot=mdot*rref*sqrt(GMA*R/MW*tref)*thrtD;
isp=dimthr/dimdot/9.81;
nisp=isp/videal*9.81;
```

```
cf=dimthr/totp/thrtD;
cfideal=sqrt(2*GMA*GMA/GMA1*pow(2/(GMA+1),(GMA+1)/GMA1)*etap)
  +p2p1*(grid[n_r-1][n_c-1].y-grid[0][n_c-1].y);

printf("Axial Thrust per depth [mN/micron]: %f \n",dimthr/1000);
printf("Total thrust [mN/micron]: %f   Symmetric thrust [mN/micron]: %f \n"
       ,totth*rref*GMA*R/MW*tref*thrtD/1000,
       symth*rref*GMA*R/MW*tref*thrtD/1000);
printf("Actual Mass Flow Rate: %f (mg/micron*s)\n",
       dimdot);
printf("Ideal Exit Mach number: %f  \n",Mie);
printf("Isp: %f  Ideal Isp: %f  Nisp %f \n",isp,videal/9.81,nisp);
printf("Cd: %f   Cf: %f   Cf ideal:  %f  Nthr: %f \n"
       ,dimdot/rref/sqrt(GMA*R/MW*tref)/thrtD, cf,cfideal,ath/invthr);


return totth;
}

#endif

float mdot(int n_r, int n_c, struct node **state, struct point **grid,
           int pnt)
{

float dmdot,mdot, f[260], f1[260], ds, nx, ny;
int ir;

dmdot=0.0;
mdot=0.0;
f[0]=state[0][pnt].w[1];
f1[0]=state[0][pnt].w[2];

  for(ir=0;ir<n_r-1;ir++){
    ds=sqrt(pow((grid[ir+1][pnt].x-grid[ir][pnt].x),2)
        +pow((grid[ir+1][pnt].y-grid[ir][pnt].y),2));
    nx=(grid[ir+1][pnt].y-grid[ir][pnt].y)/ds;
    ny=-(grid[ir+1][pnt].x-grid[ir][pnt].x)/ds;
    f[ir+1]=state[ir+1][pnt].w[1];
    f1[ir+1]=state[ir+1][pnt].w[2];
    dmdot=(0.5*nx*(f[ir]+f[ir+1])+0.5*ny*(f1[ir]+f1[ir+1]))*ds;
    mdot += dmdot;
  }
  return mdot;
}

void Dump_Data(int n_r, int n_c, struct node **state, float totp, float tott,
               float thrD, struct point **grid)
{
  int ir, ic;
  double u, v, t, p , rho, vmag, m,a,ptp1,TtT1,rtr1,pref,rref,tref;
  float nd,cd,k,lambda,kn;
  FILE *fp_m, *fp_u, *fp_v, *fp_t,*fp_p, *fp_rho, *fp_kn;


  fp_m=fopen("m.dat","w");
  fp_u=fopen("u.dat","w");
  fp_v=fopen("v.dat","w");
  fp_t=fopen("t.dat","w");
  fp_p=fopen("p.dat","w");
  fp_rho=fopen("r.dat","w");
  fp_kn=fopen("kn.dat","w");

  ptp1=pow(2.0/(GMA+1),GMA/(GMA1));
  TtT1=2.0/(GMA+1.0);
```

```
  rtr1=1.0/pow((GMA+1)/2.0,1/(GMA1));
  k=1.38e-23;
  cd=3.798e-10;

  tref=TtT1*tott;
  rref=rtr1*totp/tott/R*MW;
  pref=GMA*R/MW*tref*rref;

  for(ir=0;ir<n_r;ir++){
    for(ic=0;ic<n_c;ic++){
      rho=state[ir][ic].w[0];
      u = state[ir][ic].w[1] / rho;
      v = state[ir][ic].w[2] / rho;
      p = GMA1*(state[ir][ic].w[3] - 0.5*(state[ir][ic].w[1]*u +
                                          state[ir][ic].w[2]*v));

      t = GMA*p/rho;
      vmag=sqrt(u*u+v*v);
      a = sqrt(GMA*p/rho);
      m=vmag/a;
      nd=p*pref/k/(t*tref);
      lambda=1/sqrt(2)/pi/cd/cd/nd;
      kn=lambda/(grid[n_r-1][ic].y-grid[1][ic].y)/thrD;
      /*printf("%i %i %e %e %e %e %f %f\n",ir,ic,nd,lambda,kn,p*pref,
            t*tref,thrD);*/

      fprintf(fp_m,"%.10lf ",m);
      fprintf(fp_u,"%.10lf ",u);
      fprintf(fp_v,"%.10lf ",v);
      fprintf(fp_t,"%.10lf ",t);
      fprintf(fp_p,"%.10lf ",p);
      fprintf(fp_rho,"%.10lf ",rho);
      fprintf(fp_kn,"%.10lf ",kn);
    }
    fprintf(fp_m,"\n");
    fprintf(fp_u,"\n");
    fprintf(fp_v,"\n");
    fprintf(fp_t,"\n");
    fprintf(fp_p,"\n");
    fprintf(fp_rho,"\n");
    fprintf(fp_kn,"\n");
  }
  fclose(fp_m);
  fclose(fp_u);
  fclose(fp_v);
  fclose(fp_t);
  fclose(fp_p);
  fclose(fp_rho);
  fclose(fp_kn);

}

void thick(int n_c, int n_r, struct node **state, struct point **grid)
{
int ir,ic,mid,is;
float disp[260],mom[260],mome,ds,f[260],g[260],shape[260],ue,u,v;
float gmome,gume,u1,v1;
FILE *fpr;

fpr=fopen("thick.dat","w");
mid=(int)(n_r/2);


for(ic=0;ic<n_c;ic++){
  gmome=0;
  gume=0;
  for(is=0;is<mid;is++){
    u=state[is][ic].w[1]/state[is][ic].w[0];
    v=state[is][ic].w[2]/state[is][ic].w[0];
    ue=sqrt(u*u+v*v);
    mome=state[is][ic].w[0]*sqrt(u*u+v*v);

    if(mome > gmome)
      gmome=mome;

    if(ue > gume)
      gume=ue;
  }

  ue=gume;
  mome=gmome;
  ir=0;


  disp[ic]=0.0;
  mom[ic]=0.0;
  shape[ic]=0.0;
  u=state[ir][ic].w[1]/state[ir][ic].w[0];
  v=state[ir][ic].w[2]/state[ir][ic].w[0];
  f[0]=(1.0-(state[ir][ic].w[0]*sqrt(u*u+v*v))/mome);
  g[0]=(1.0-sqrt(u*u+v*v)/ue)*(state[ir][ic].w[0]*sqrt(u*u+v*v))/mome;

  do{
    u1=state[ir+1][ic].w[1]/state[ir+1][ic].w[0];
    v1=state[ir+1][ic].w[2]/state[ir+1][ic].w[0];

    g[ir+1]=(1.0-sqrt(u1*u1+v1*v1)/ue)
      *state[ir+1][ic].w[0]*sqrt(u1*u1+v1*v1)/mome;
    f[ir+1]=(1.0-state[ir+1][ic].w[0]*sqrt(u1*u1+v1*v1)/mome);
    ds=grid[ir+1][ic].y-grid[ir][ic].y;
    disp[ic] = 0.5*ds*(f[ir]+f[ir+1])+disp[ic];
    mom[ic] = 0.5*ds*(g[ir]+g[ir+1])+mom[ic];
    ir++;

  }while (ir < mid);

  shape[ic]=disp[ic]/mom[ic];
  fprintf(fpr,"%i %f %f %f %f \n",ic,grid[0][ic].x,disp[ic],mom[ic],shape[ic]);

}
  fclose(fpr);
}


/*~~~~ This function writes the current flow variables to disk in separate
  files ~~~~*/
void Dump_Data2(int n_c, int n_r, struct node **state, float t)
{
  int ir;
  int ic;
  double u, v, temp, p,rho;
  static int f_flag=0;
  static FILE *fp;
  char name[50];

  sprintf(name,"p%03.dat",t);
  fp = fopen(name,"w");

  for(ir=0;ir<n_r;ir++)
    for(ic=0;ic<n_c;ic++){
```

```
      rho=state[ir][ic].w[0];
      u = state[ir][ic].w[1] / rho;
      v = state[ir][ic].w[2] / rho;
      p = GMA1*(state[ir][ic].w[3] - 0.5*(state[ir][ic].w[1]*u +
                                          state[ir][ic].w[2]*v));
      fprintf(fp,"%.10lf ",p);

    }
  fflush(NULL);
}
```

Appendix C

Process Traveler

The following are process steps undertaken to fabricate the cold-gas microthrusters. The processes were performed in the Integrated Circuits Laboratory (ICL) and Technology Research Laboratory (TRL). The raw materials are:

Nozzle wafer (N) - $< 100 >$ Silicon 500 microns thick
Handle wafer (H) - Quartz Wafer
Injection Plate (P1) - Pyrex wafer pre-drilled with gas injection holes
Back Plate (P2) - Plain Pyrex wafer

Location	Wafer	Machine	Process Specifications
TRL	N	HMDS	Standard HMDS cycle
TRL	N	Coater	Coat wafer with AZ 4620 spin
			at 1750/1000/7000 rpm for 10/60/9 seconds
TRL	N	Pre-bake	30 min at $90°C$
TRL	N	Karl Suss 2	Expose halo mask for 360 seconds
TRL	N	Photo-Wet-L	Develop for 5:30 in Mif-440
TRL	H	Coater	Spin on thick resist at 3000 rpm; Mount to device wafer
TRL	N,H	Post-bake	$90°C$ for 30 minutes
TRL	N,H	STS	Etch using Recipe STS-56 for approximately 200 minutes
			for a halo trench of 50 microns
TRL	N,H	Acid Hood	Strip resist and separate handle wafer in piranha clean
TRL	N,P1,P2	Acid Hood	Piranha clean Pyrex and silicon in clean labware
TRL	N,P1	EV Bonder	Align glass and silicon holes in the aligner
			Bond at $500°C$ with 1000 Volts
TRL	N/P1,P2	EV Bonder	Place Brass Spring in auxiliary hole repeat previous bond step
ICL	N/P1/P2	Diesaw	Cut along dicing lanes with glass blade to permit operation

The process was enhanced to produce the microresistojets developed in chapter 7. They are fabricated using the following steps and raw material:

Thruster wafer (T) - $< 100 >$ Silicon 500 microns thick of at least 1×10^{19} p-type doping

Handle wafer (H) - Quartz Wafer

Injection Plate (IP) - Silicon Wafer identical to T wafer

Back Plate (BP) - Unprocessed Silicon wafer identical to IP and T

Location	Wafer	Machine	Process Specifications
ICL	IP,T	RCA Station	Standard RCA clean
ICL	IP,T	Concept 1	PECVD oxide 4 microns thick
ICL	IP,T	Tube B5	Densify Oxide for 1 hour using recipe-180
TRL	IP,T	HMDS	Standard HMDS cycle
TRL	IP,T	Coater	Spin on thick resist at 3000 rpm
			to protect back side from AME damage
TRL	IP,T	Pre-bake	30 min at $90°C$
TRL	IP,T	Coater	Coat side w/ oxide with AZ 4620
			at 1750/1000/7000 rpm for 10/60/9 seconds
TRL	IP,T	Pre-bake	30 min at $90°C$
TRL	IP,T	Karl Suss 2	Expose Injection hole mask on IP for 360 seconds
			and nozzle-fin mask on T
TRL	IP,T	Photo-Wet-L	Develop for 5:30 in Mif-440
TRL	IP,T	Post-bake	$90°C$ for 30 minutes
ICL	IP,T	AME 5000	Etch Oxide using Recipe: Nagle CHF3
TRL	IP,T	Acid Hood	Strip resist in piranha clean
TRL	H	Coater	Spin on thick resist at 3000 rpm; Mount to IP wafer
TRL	H	Post-bake	$90°C$ for 30 minutes
TRL	IP	STS	Etch using Recipe STS-56 for $\sim$ 200 minutes
			for a halo trench of 50 microns
TRL	IP	Acid Hood	Strip resist and separate hand wafer in piranha clean
TRL	IP ,T	HMDS	Standard HMDS cycle
TRL	T	Coater	Coat side w/ oxide with OCG 825-35 spin
			at 200/700/2000 rpm for 10/6/30 seconds
TRL	T	Pre-bake	30 min at $90°C$
TRL	T	Karl Suss 2	Expose using fin protection mask for 60 seconds
TRL	T	Photo-Wet-L	Develop for 1:30 in OCG 934 1:1
TRL	T	Post-bake	$120°C$ for 30 minutes
TRL	T	STS	Etch using Recipe STS-56 for approximately 25 minutes
TRL	T	Acid Hood	Strip resist in piranha clean
ICL	T	Asher	Place in asher for 5 minutes to strip side wall passivation
TRL	H	Coater	Spin on thick resist at 3000 rpm; Mount to T wafer
TRL	H	Post-bake	$90°C$ for 30 minutes
TRL	T	STS	Continue Etch using Recipe STS-56 for $\sim$ 175 minutes
			Stop when all trenches are clear but those between fins
TRL	T	Acid Hood	Strip resist in piranha clean and separate handle
TRL	T,BP,IP	RCA Station	Perform RCA clean on silicon wafers prior to bonding
TRL	T,BP,IP	EV Bonder	Align and bond IP, BP and T simultaneously
TRL	T/BP/IP	Tube B3	Anneal wafer stack at $1100°C$ for 1 hour
			ramp up at $20°C/min$
ICL	T/BP/IP	Diesaw	Cut along dicing lanes with glass blade to permit operation

Bibliography

[1] Collins D. Kukkonen K. and Venneri S. Miniature, low-cost, highly Autonomous Spacecraft-A Focus for the New Millennium. In *46th International Astronautical Congress*. IAF-96-U.2.06, 1996.

[2] J. Mueller. Thruster Options for Microspacecraft: a Review and Evaluation of Existing Hardware and Emerging Technologies. In *33rd AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*. AIAA 97-3058, 1997.

[3] London A. P. A Systems Study of Propulsion Technologies for Orbit and Attitude Control of Microspacecraft. Master's thesis, Massachusetts Institute of Technology, 1996.

[4] G. Sutherland and M. Maes. A Review of Microrocket Technology: 10^{-6} to 1 lbf Thrust. *Journal of Spacecraft and Rockets*, 3(8):1153–1163, 1966.

[5] V. Khayms and M. Martinez-Sanchez. Design of a Miniaturized Hall Thruster for Microsatellites. In *32rd AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*. AIAA 96-3291, 1996.

[6] Massier P. Back L. Noel M. and Saheli F. Viscous Effects on the Flow Coefficient for a Supersonic Nozzle. *AIAA Journal*, 8(3):605–607, 1970.

[7] G. A. Kuluva N. M. Hosack. Supersonic Nozzle Discharge Coefficients at Low Reynolds Numbers. *AIAA Journal*, 9(9):1876–1877, 1971.

[8] D. Rothe. Electron-beam Studies of Viscous Flow in Supersonic Nozzles. *AIAA Journal*, 9(5):804–811, 1971.

[9] W. Rae. Some Numerical Results on Viscous Low-Density Nozzle Flows in the Slender-Channel Approximation. *AIAA Journal*, 9(5):811–821, 1971.

[10] T. A. Grisnik, S. P. Smith and Saltz L.E. Experimental Study of Low Reynolds Number Nozzles. In *19th AIAA/DGLR/JSASS Internation Electric Propulsion Conference*. AIAA 87-0992, 1987.

[11] S. C. Kim. Calculations of Low-Reynolds-Number Resistojet Nozzles. *Journal of Spacecraft and Rockets*, 31(4):259–264, March-April 1994.

[12] M. Zelesnik, D. Micci and L. Long. Direct Simulation Monte Carlo Model of Low Reynolds Number Nozzle Flows. *Journal of Propulsion and Power*, 10(4):546–553, 1994.

[13] Ivanov M.S. Markelov G.N. Ketsdver A.D. and Wadworth D.C. Numerical Study of Cold Gas Micronozzle Flows. In *37th Aerospace Sciences Meeting and Exhibit*. AIAA 99-0166, 1999.

[14] S.W. Janson and H. Helvajian. Batch-Fabricated Microthrusters: Initial Results. In *32nd AIAA/ASME/SAE/ASEE Joint Propulsion Conference*. AIAA-96-2988, 1996.

[15] Lewis D.H. Janson S.W. Cohen R.B. Antonsson E.K. Digital Micropropulsion. In *MEMS '99 Orlando, FL*. IEEE, 1999.

[16] S. W. Janson. Batch-Fabricated Resistojets: Initial Results. In *International Electric Propulsion Conference*. Electri Rocket Propulsion Society IEPC-97-070, 1997.

[17] Mueller J. Tang W. Wallace A. Li W. Bame D. Chakraborty I. and Lawton R. Design, Analysis and Fabrication of a Vaporizing Liquid Micro-Thruster. In *33rd AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*. AIAA 97-3054, 1997.

[18] J.D. Anderson. *Hypersonic and High Temperature Gas Dynamics*. McGraw-Hill, 1st edition, 1989.

[19] B. Van Leer. Flux-vector Splitting for the Euler Equations. *Lecture Notes in Physics*, 170:507–512, 1982.

[20] E.S. Piekos. Personal communication and documentation. 1996.

[21] Anderson D.A. Tannehill J.C. and Pletcher R.H. *Computational Fluid Mechanics and Heat Transfer*. Taylor and Francis, 1st edition, 1984.

[22] W. Jameson, A. Schmidt and E. Turkel. Numerical Solution of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time Stepping Schemes. AIAA 81-1259, 1981.

[23] C. Hirsch. *Numerical Computation of Interal and External Flows*. John Wiley and Sons, 2nd edition, 1990.

[24] J. Peraire. Personal communication. 1996.

[25] W. M. Rohsenow. *Heat, Mass, and Momentum Transfer*. Prentice-Hall, 1st edition, 1961.

[26] E. B. Arkilic. *Measurement of the Mass Flow and Tangential Momentum Accomodation Coefficient in Silicon Micromachined Channels*. PhD thesis, Massachusetts Institute of Technology, 1997.

[27] J.D. Hoffman. *Numerical Methods for Engineers and Scientists*. McGraw-Hill, 1st edition, 1992.

[28] J. Steger and R. Sorenson. Automatic Mesh-Point Clustering in Grid Generation with Elliptic Partial Differential Equations. *Journal of Computational Physics*, 33:405–410, 1979.

[29] F. M. White. *Viscous Fluid Flow*. McGraw-Hill, 2nd edition, 1991.

[30] L.M. Mack. Boundary-Layer Linear Stability Theory. In *Special Course on Stability and Transition in Laminar Flow*, volume 709. AGARD, 1984.

[31] Schlichting H. *Boundary Layer Theory*. Artech House, 4th edition, 1979.

[32] D. Guckel, H. Burns. A technology for integrated transducers. In *Technical Digest, International Conference on Solid-State Sensors and Actuators (Transducer '85)*. IEEE, 1985.

[33] Senturia S.D. Schmidt M.A. Harrison D.J. *Microsystems: Mechanical, Chemical, Optical*. MIT, 1.2 edition, 1996.

[34] T. Diepold. A Micromachined Continuous Ink Jet Print Head for High Resolution Printing. In *Microsensor and Actuator Technology*. TU-Berlin, 1998.

[35] Gmbh Robert Bosch. Method for Anisotropically Etching Silicon. Patents 4855017 and 4784720(USA), and 4241045C1(Germany).

[36] Ayon A.A Lin C.C. Braff R.A. Schmidt M.A. Bayt R.L. Sawin H.H. Etching Characteristics and Profile Control in a Time Multiplexed Inductively Coupled Plasma Etcher. In *Solid-State Sensor and Actuator Workshop*. Transducers Research Foundation, 1998.

[37] G. Wallis and D. I. Pomerantz. Field Assisted Glass-Metal Sealing. *Journal of Applied Physics*, 40(10):3946–3949, Septemeber 1969.

[38] T. Rogers and J. Kowal. Selection of glass, anodic bonding conditions and material compatibility for silicon-glass capacitive sensors. *Sensors and Actuators A*, 46(113):113–120, Septemeber 1995.

[39] M. Grundner and H. Jacob. Investigation on hydrophilic and hydrophobic silicon $< 100 >$ wafer surfaces by x-ray photoelectron and high-rsolution electron energy-loss spectroscopy. *Journal of Applied Physics*, A39:73, 1986.

[40] L. Ristic. *Sensor Technology and Devices*. Artech House, 1st edition, 1994.

[41] M.A. Schmidt. Silicon wafer-bonding for micromechanical devices. *Technical Digest, Solid State Sensor and Actuator Workshop*, pages 127–131, 1994.

[42] L.A. Gochberg and B.L. Haas. Evaluation of Rotational Relaxation Rate Models in Low-Density Expanding Flows of Nitrogen. In *30th AIAA Thermophysics Conference*. AIAA 95-2070, 1995.

[43] G. P. Sutton. *Rocket Propulsion Elements*. Wiley-InterScience, 6th edition, 1992.

[44] K.F. (with others) Jensen. Micromachined Chemical Reactors for Heterogeneously Catalyzed Partial Oxidation Reaction. *AlChe Journal*, 143:3059–3069, 1997.

[45] Fitch J.S. Henning A.K. Arkilic E.B. and Harris J.M. Pressure-Based Mass-Flow Control Using Thermopneumatically-Actuated Microvalves. In *Solid-State Sensor and Actuator Workshop*. Transducers Research Foundation, 1998.

[46] R. K. Kakac, S. Shah and A. E. Bergles. *Low Reynolds Number Flow Heat Exchangers*. Hemisphere Publishing, 1st edition, 1983.

[47] D. N. Ashcroft, N.W. Mermin. *Solid State Physics*. Holt, Rinehart and Winston, 1st edition, 1976.

[48] G. L. Pearson and J. Bardeen. Electrical properties of Pure Silicon and Silicon Alloys Containing Boron and Phosphorous. *Physical Review*, 75(1):865–886, 1949.

[49] J. Kalamas. *A Three-Dimensional Flow Stability Analysis of the Particle Bed Nuclear Reactor*. PhD thesis, Massachusetts Institute of Technology, 1993.

[50] M. Martinez-Sanchez. Electrical Leads with Least Thermal Losses. In *16.522 course notes on Electric Propulsion*, 1996.