Minimum Length Nozzle Design Using the Method of Characteristics

Eric Sandall

AE 8803
Georgia Institute of Technology

12/7/2018

Contents

Nomenclature 2
Introduction 2
Method of Characteristics 3
    Determining Characteristic Lines 4
    Determining Compatibility Equations 5
    Solving Characteristics 5
Nozzle Contour Results 8
CFD Verification 10
    Planar CFD 10
    Axisymmetric CFD 11
Summary 14
Appendix A - Tables 15
Appendix B - Planar Script 17
Appendix C - Axisymmetric Script 21
References 28

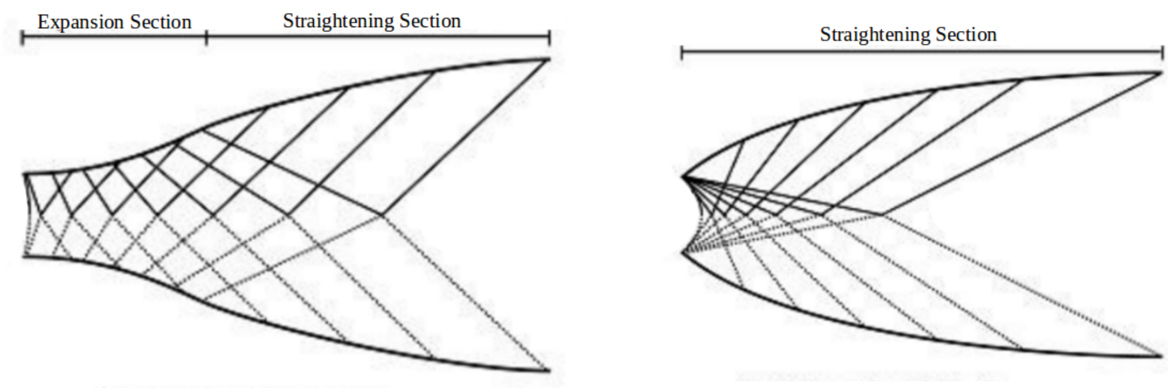
Nomenclature

γ	=	Specific heat ratio
θ	=	Flow angle from horizontal
μ	=	Mach angle
ν	=	Prandtl-Meyer function
ρ	=	Density (mass/volume)
a	=	Local speed of sound
A	=	Cross-sectional area
C_-, C_+	=	Right- and left-running characteristics
e	=	Specific internal energy
$\mathbf{f}$	=	Body forces
K_-, K_+	=	Characteristic constants for right- and left-running characteristics
M	=	Mach number
p	=	Pressure
$\dot{q}$	=	Specific heat flux
t	=	Time
$\mathbf{V}, V$	=	Velocity vector, magnitude

Introduction

The purpose of this work is to design a minimum length supersonic nozzle using the method of characteristics. A minimum length nozzle is often beneficial in rocketry applications in order to reduce weight and to accelerate the flow as quickly as possible. Many nozzles employ a simple conical design, which is shorter than contoured nozzles [Dumitrescu(1975)] and is relatively easier to manufacture. However, these conical nozzles are also accompanied by non-uniform flow and shocks, which result in lower thrust. Maximum thrust is produced when a nozzle perfectly expands the gases with uniform flow at the nozzle exit [Rao(1958)].

Optimum nozzles for producing parallel flow, such as wind tunnel nozzles, typically are very long with reasonably slow expansion [Anderson Jr(1990)]. The diverging section is comprised of two parts, an expansion section and a straightening section as shown in Figure 1a. In order to minimize the nozzle length, the expansion region can be confined to a single point as pictured in Figure 1b. It is also important to note that the sonic line at the throat is curved. The method of characteristics can be applied to obtain this contour.



(a) Smooth Expansion Section

(b) Singular Expansion Section

Figure 1: Typical de Laval Nozzles

Method of Characteristics

[Anderson Jr(1990)] describes the basic theory of the method of characteristics, which will be paraphrased here. First, within a supersonic flow field there are lines on which the fluid properties (pressure, velocity, temperature, etc.) are continuous, but their derivatives are indeterminate. These lines are characteristic lines. Second, by combining the partial differential conservation equations in a certain way, ordinary differential equations can be obtained that only hold along the characteristic lines. The equations are called compatibility equations. Third, solve the compatibility equations along the characteristic lines to obtain a "characteristic net" which maps the entire flow field.

The differential conservation equations for inviscid flows in cartesian coordinates are given as:

$$
\begin{aligned}
&\textbf{Continuity:}\ \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{V}) = 0 \\
&\textbf{Momentum:}\ \begin{cases} \dfrac{\partial(\rho u)}{\partial t} + \nabla \cdot (\rho u \mathbf{V}) = -\dfrac{\partial p}{\partial x} + \rho f_x \\ \dfrac{\partial(\rho v)}{\partial t} + \nabla \cdot (\rho v \mathbf{V}) = -\dfrac{\partial p}{\partial y} + \rho f_y \\ \dfrac{\partial(\rho w)}{\partial t} + \nabla \cdot (\rho w \mathbf{V}) = -\dfrac{\partial p}{\partial z} + \rho f_z \end{cases} \\
&\textbf{Energy:}\ \frac{\partial}{\partial t}\left[\rho\left(e + \frac{V^2}{2}\right)\right] + \nabla \cdot \left[\rho\left(e + \frac{V^2}{2}\right)\mathbf{V}\right] = -\nabla \cdot (p\mathbf{V}) + \rho \dot{q} + \rho(\mathbf{f} \cdot \mathbf{V})
\end{aligned}
\tag{1}
$$

Determining Characteristic Lines

As stated earlier, the first step is to determine the characteristic lines. There are two of these characteristic lines through every point within the flow field, left-running and right-running (see Figure 2). Leveraging the velocity potential equation, the conservation equations and some algebraic manipulation, the characteristic lines for planar flow are given by:

$$\left(\frac{dy}{dx}\right)_{char} = \frac{-uv/a^2 \pm \sqrt{[(u^2+v^2)/a^2]-1}}{[1-(u^2/a^2)]},$$

which after some further manipulation simplifies to

$$\left(\frac{dy}{dx}\right)_{char} = tan(\theta \mp \mu) \tag{2}$$

The subtraction operator is applied for right-running characteristics and the addition operator is applied for left-running characteristics. A similar approach is used to obtain the characteristic lines in an axisymmetric case with, interestingly, the same result.

$$\left(\frac{dr}{dx}\right)_{char} = tan(\theta \mp \mu) \tag{3}$$

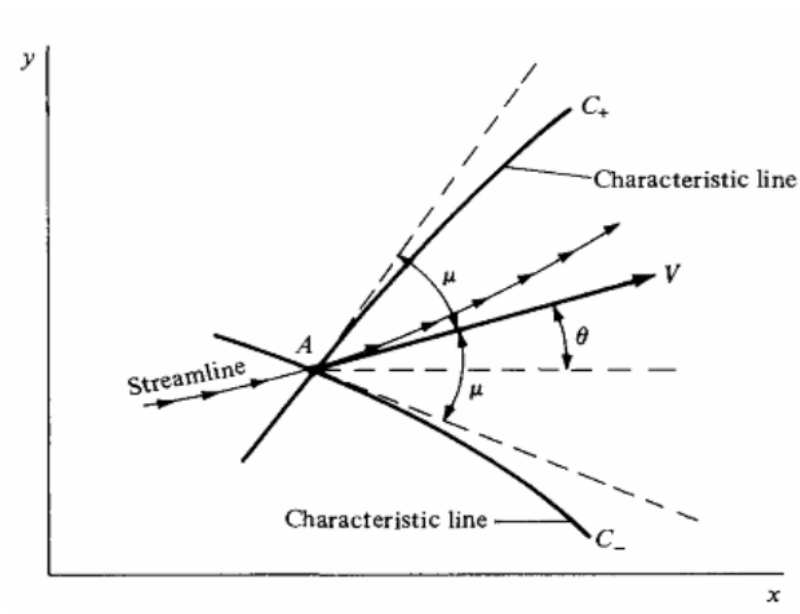


Figure 2: Characteristic lines through a point

Determining Compatibility Equations

The next step is to determine the compatibility equations. A derivation is given by [Anderson Jr(1990)]. For planar flows, the equations reduce to algebraic equations given by:

$$\begin{aligned} \theta + \nu &= constant = K_- \quad \text{(along right-running } C_- \text{ characteristic)} \\ \theta - \nu &= constant = K_+ \quad \text{(along left-running } C_+ \text{ characteristic)} \end{aligned} \tag{4}$$

For the axisymmetric case, things become a little more difficult as the compatibility equations reduce to differential equations rather than algebraic equations. They are defined by:

$$\begin{aligned} d(\theta + \nu) &= \frac{1}{\sqrt{M^2 - 1} - cot\theta} \frac{dr}{r} \quad \text{(along right-running } C_- \text{ characteristic)} \\ d(\theta - \nu) &= -\frac{1}{\sqrt{M^2 - 1} + cot\theta} \frac{dr}{r} \quad \text{(along left-running } C_+ \text{ characteristic)} \end{aligned} \tag{5}$$

Solving Characteristics

A few assumptions and simplifications are made before solving the characteristics. First, although the sonic line is curved at the throat, a straight-line is assumed. Also, as the term minimum length nozzle implies, the expansion section of the nozzle is assumed to exist at a singular point. The expansion angle is given by half the Prandtl-Meyer function for the designed Mach number of the nozzle. It is also important to note that isentropic, irrotational flow is assumed in all of the derivations and equations regarding the characteristics.

Along a given characteristic line, we have two equations, but four unknowns. Thus the system of characteristics can only be solved at the intersection of the right- and left-running characteristic lines. This results in four equations for four unknowns (x,y or r, θ, and ν). The other terms (μ and M) can be determined algebraically using the Prandtl-Meyer and Mach angle equations. Thus, knowing the initial data at two preceding points, a solution can be determined by marching downstream. A sample characteristic web is shown in Figure 3.

Planar Case

For the planar case, the compatibility equations are already algebraic, but the characteristic lines are given as differential equations. Thus, to simplify the problem, a finite difference method is applied to the differential equations and the result is four algebraic equations. At the expansion corner, the expansion angle is divided into parts (the number of characteristics used to solve the nozzle will determine the number of divisions). Knowing the initial conditions at the throat, the system of equations can be solved moving through the flow field. At the symmetry line, the flow angle, θ, and y-coordinate are predefined as 0, and these equations can be used as substitutes for the left-running characteristic equations. At

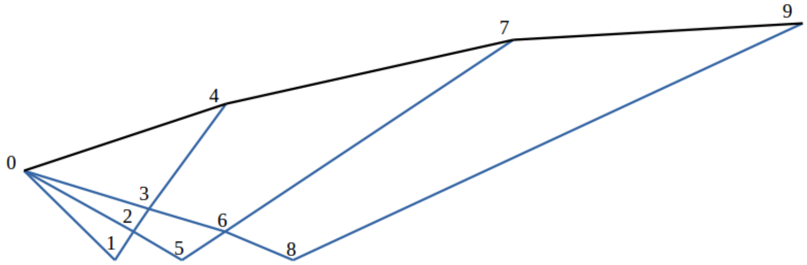


Figure 3: Characteristic lines through a point

the wall, the Mach number (and thus Mach angle and Prandl-Meyer function) is the same as the previous Mach number along the left-running characteristic and the wall angle is equal to the average flow angle of the previous wall point and the current wall point being solved.

The system of equations for an interior point (point 6 in Figure 3 for this example) is shown. The subscript defines the point on the characteristic net being evaluated. For example, y_6 is the y-coordinate of point 6 in the characteristic net. Although the system can be solved simultaneously using a linear solver, it is effective to explicitly solve for each unknown algebraically. This way, a matrix solver does not need to be implemented and the solution can be found with four explicit equations. The explicit equations are straightforward, yet somewhat tedious to derive and are not shown.

$$\begin{cases} \dfrac{y_6 - y_3}{x_6 - x_3} = tan(\theta_3 - \mu_3) & (C_- \text{ characteristic}) \\ \dfrac{y_6 - y_5}{x_6 - x_5} = tan(\theta_5 + \mu_5) & (C_+ \text{ characteristic}) \\ \theta_6 + \nu_6 = K_- = \theta_3 + \nu_3 & (C_- \text{ characteristic}) \\ \theta_6 - \nu_6 = K_+ = \theta_5 - \nu_5 & (C_+ \text{ characteristic}) \end{cases} \tag{6}$$

The system for a point on the symmetry line (point 5):

$$\begin{cases} \dfrac{y_5 - y_2}{x_5 - x_2} = tan(\theta_2 - \mu_2) & (C_- \text{ characteristic}) \\ y_5 = 0 & \\ \theta_5 + \nu_5 = K_- = \theta_2 + \nu_2 & (C_- \text{ characteristic}) \\ \theta_5 = 0 & \end{cases} \tag{7}$$

The system for a point on the wall (point7):

$$
\begin{cases}
\dfrac{y_7 - y_4}{x_7 - x_4} = tan\Big(\dfrac{\theta_4 + \theta_7}{2}\Big) \\
\dfrac{y_7 - y_6}{x_7 - x_6} = tan(\theta_6 + \mu_6) \quad (C_+ \text{ characteristic}) \\
\nu_7 = \nu_6 \\
\theta_7 - \nu_7 = K_+ = \theta_6 - \nu_6 \quad (C_+ \text{ characteristic})
\end{cases}
\tag{8}
$$

Axisymmetric Case

The axisymmetric characteristic equations are all differential equations, so the finite difference approach is applied to all four equations. Some approaches employ a predictor-corrector method, but for the present work only a single step finite difference method is used. The same approach described previously for the planar case is also applied for the axisymmetric case.

The system for an interior point (point 6):

$$
\begin{cases}
\dfrac{r_6 - r_3}{x_6 - x_3} = tan(\theta_3 - \mu_3) \quad (C_- \text{ characteristic}) \\
\dfrac{r_6 - r_5}{x_6 - x_5} = tan(\theta_5 + \mu_5) \quad (C_+ \text{ characteristic}) \\
\dfrac{(\theta_6 + \nu_6) - (\theta_3 + \nu_3)}{r_6 - r_3} = \dfrac{1}{\sqrt{M_3^2 - 1} - cot\theta_3} \dfrac{1}{r_3} \quad (C_- \text{ characteristic}) \\
\dfrac{(\theta_6 - \nu_6) - (\theta_5 - \nu_5)}{r_6 - r_5} = -\dfrac{1}{\sqrt{M_5^2 - 1} + cot\theta_5} \dfrac{1}{r_5} \quad (C_+ \text{ characteristic})
\end{cases}
\tag{9}
$$

The system for a point on the symmetry line (point 5):

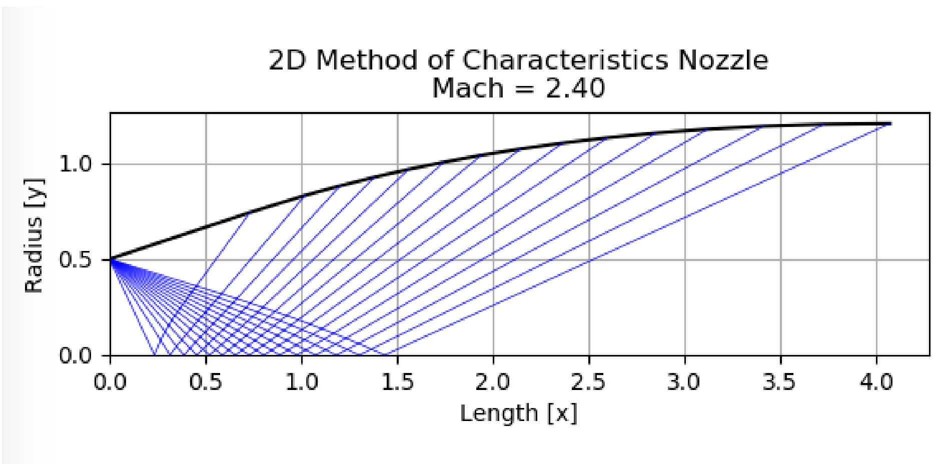
$$
\begin{cases}
\dfrac{r_5 - r_2}{x_5 - x_2} = tan(\theta_2 - \mu_2) \quad (C_- \text{ characteristic}) \\
r_5 = 0 \\
\dfrac{(\theta_5 + \nu_5) - (\theta_2 + \nu_2)}{r_5 - r_2} = \dfrac{1}{\sqrt{M_2^2 - 1} - cot\theta_2} \dfrac{1}{r_2} \quad (C_- \text{ characteristic}) \\
\theta_5 = 0
\end{cases}
\tag{10}
$$

The system for a point on the wall (point7):

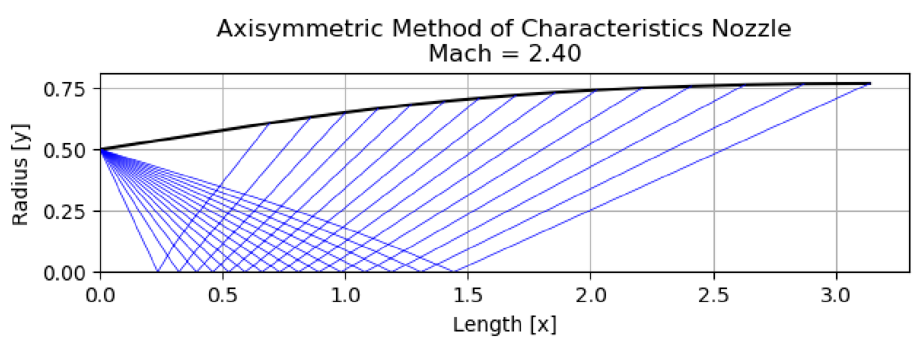
$$\begin{cases} \dfrac{r_7 - r_4}{x_7 - x_4} = tan\Big(\dfrac{\theta_4 + \theta_7}{4}\Big) \\ \dfrac{r_7 - r_6}{x_7 - x_6} = tan(\theta_6 + \mu_6) \quad (C_+ \text{ characteristic}) \\ \nu_7 = \nu_6 \\ \dfrac{(\theta_7 - \nu_7) - (\theta_6 - \nu_6)}{r_7 - r_6} = -\dfrac{1}{\sqrt{M_6^2 - 1} + cot\theta_6}\dfrac{1}{r_6} \quad (C_+ \text{ characteristic}) \end{cases} \tag{11}$$

Nozzle Contour Results

Figure 4 shows a sample output of the code for both planar (2D) and axisymmetric nozzles.



(a) Planar Nozzle



(b) Axisymmetric Nozzle

Figure 4: Code Output: Nozzle Contour

As more characteristic lines are used in the solver, the wall contour converges to the optimal shape. [Anderson Jr(1990)] gives a table illustrating the theoretical area ratio, A/A^*, for isentropic flow with $\gamma = 1.4$ for various Mach numbers. The area ratio predicted by the code is then compared to these values for multiple Mach numbers and number of characteristic lines. Tabulated results are given in Tables 1 and 2 and graphical results are shown in Figures 5 and 6.

Comparing the area ratio errors, it can be seen that, in general, the planar nozzle is about an order of magnitude closer to the theoretical value than the axisymmetric nozzle. This is understandable since the finite difference method was applied to four differential equations versus two for the planar case. While the finite difference method is effective, the error is carried and compounded as the simulation marches through the flow field.

It is also evident that for the planar nozzle, the error scales by $\mathcal{O}\left(\frac{1}{N^2}\right)$, where N is the number of characteristics. This is analogous to a step size for other numerical analyses. When N is increased by a factor of two, the error decreases by a factor of about four. This differs from the axisymmetric case, which appears to scale by $\mathcal{O}\left(\frac{1}{N}\right)$. Furthermore, as the number of characteristics increases, the rate that the error decreases diminishes. This may be due to the fact that the final Mach number is slowly increasing away from the design Mach number as the N increases. So, while increasing N reduces the error in general, it is countered by the Mach number error.

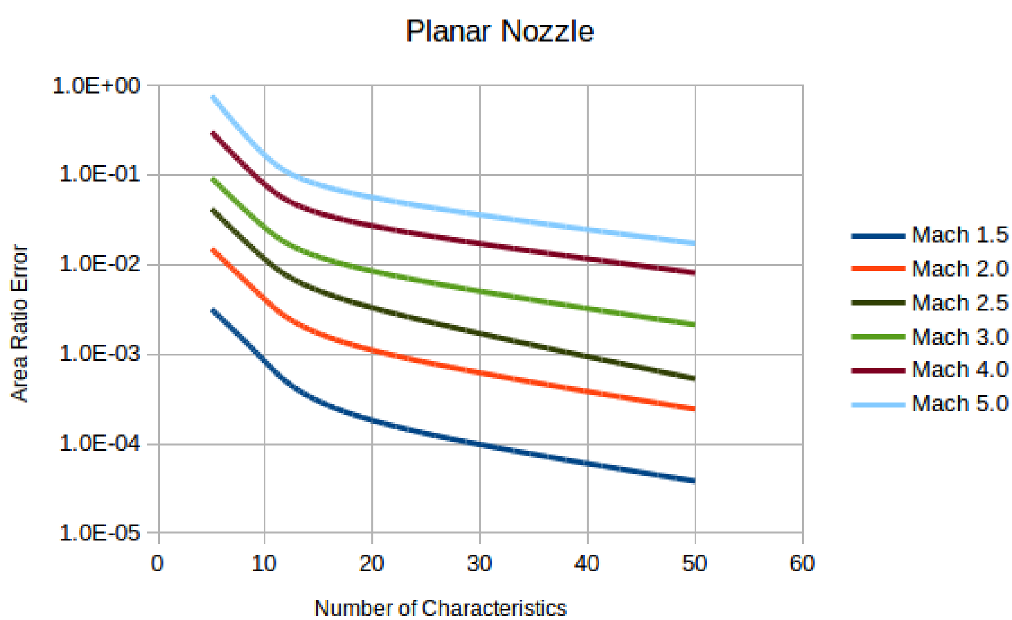


Figure 5: Planar Nozzle Error

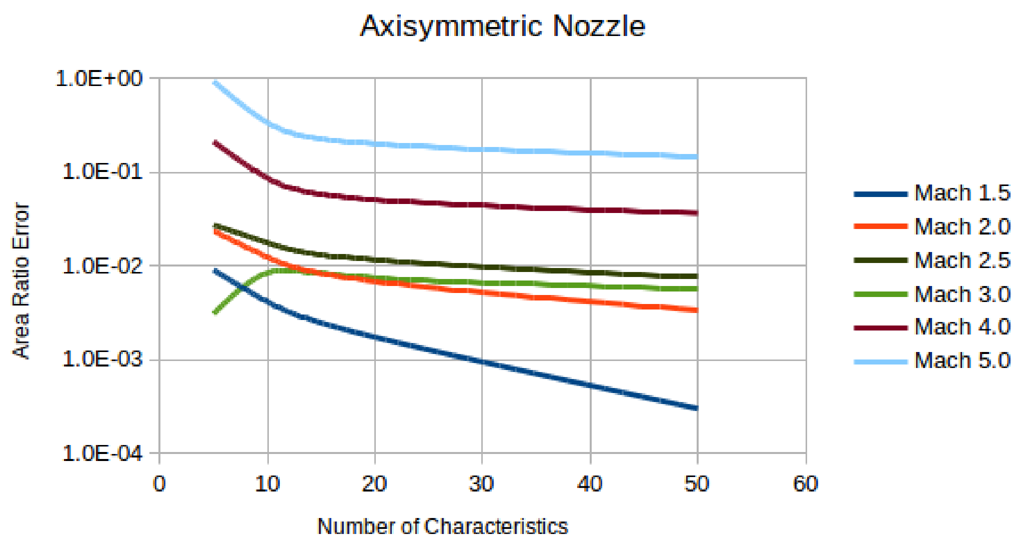


Figure 6: Axisymmetric Nozzle Error

CFD Verification

The Computational Combustion Lab (CCL) at Georgia Tech has an in-house computational fluid dynamics (CFD) software package, called LESLIE, capable of running heavily parallelized simulations. In hopes of verifying the nozzle contour code, simulations are run in 2D and 3D for the planar and axisymmetric cases respectively. The dimensions are similar for both simulations:

1. combustor length = 0.03 m
2. combustor half-height (or radius) = 0.01 m
3. throat half-height (or radius) = 0.005 m
4. Mach number = 2.4
5. N = 20 characteristics

The inviscid simulations should produce shock-free uniform flow at the designated Mach number.

Planar CFD

The planar nozzle performed nearly to design specifications according to the CFD results. Figure 7 shows the Mach numbers throughout the flow field. It can be seen that the Mach number approaches the desired value of 2.4 without the presence of any shocks within the nozzle. Figure 8a is a line chart along the symmetry line of the domain from inlet to outlet. The final Mach number at the center of the exit plane is approximately 2.445, which is slightly high, but within 2% of the desirved value. Figure 8b shows the directional Mach numbers at the exit plane. It can be seen that the y-directional flow speed is very small compared to the x-direction (the z-directional speed is zero since the simulation is 2D), indicative of near uniform flow at the exit. Thus the design parameters have been met, and the nozzle contour is verified.

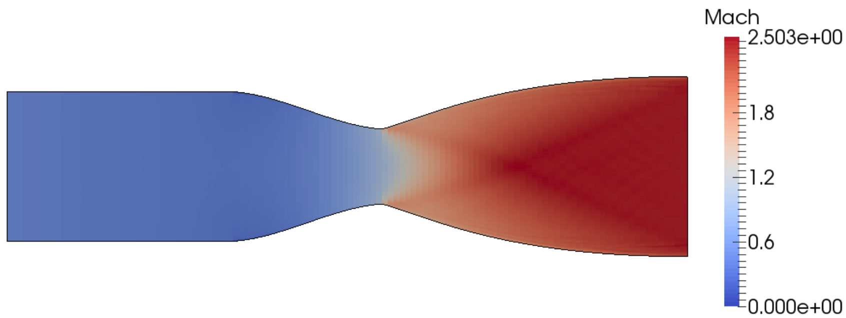
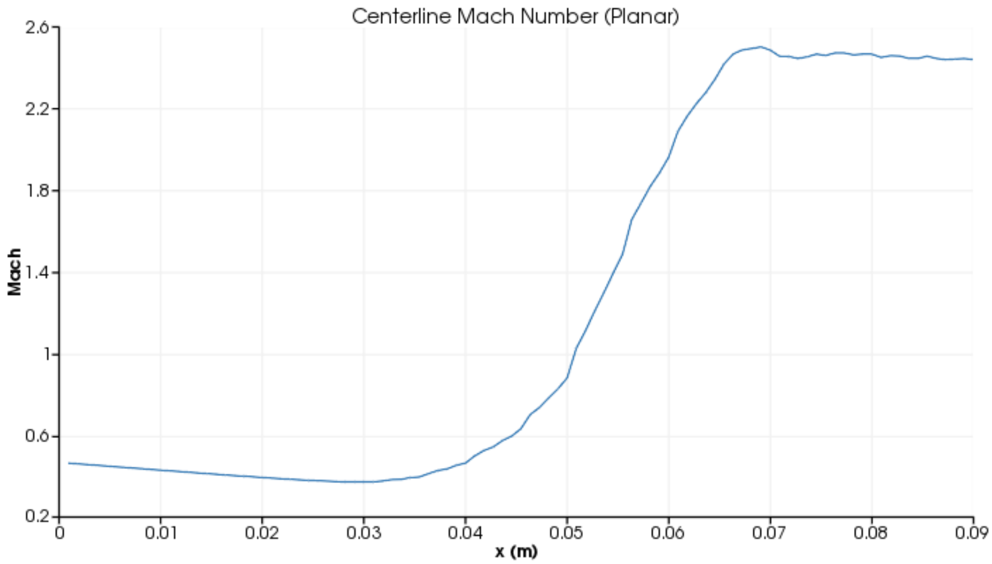


Figure 7: Planar Mach Number

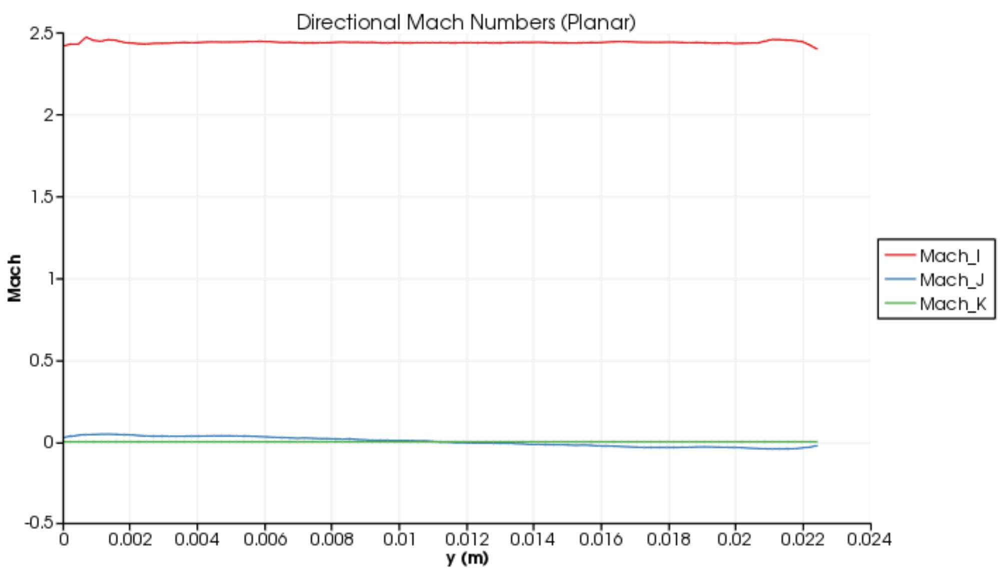
Axisymmetric CFD

The axisymmetric nozzle performed well, but didn't meet all of the design criteria. Figure 9 is a vertical slice through the center length of the nozzle and shows the acceleration of the flow to around the desired Mach number. However, it is clear that shocks are present by the presence of "Mach diamonds" shortly behind the throat. A dramatic jump in the Mach number can be seen in Figure 10 as a result of the shock. This is a result of over-expansion [Loth et al.(1992)Loth, Baum, and Löhner], indicating that the initial expansion angle at the throat may be too high.

Figure 11 shows the directional Mach numbers at the exit plane of the nozzle. Notice the difference in scales of the plots. From these plots, it is evident that most of the flow is traveling in the x-direction between Mach 2.3 and Mach 2.6. In either transverse direction, the flow is generally within Mach ± 0.1. These results show that the final Mach number is roughly the desired value, with nearly uniform flow at the exit.



(a) Centerline



(b) Nozzle Exit

Figure 8: Planar Mach Number Line Plots

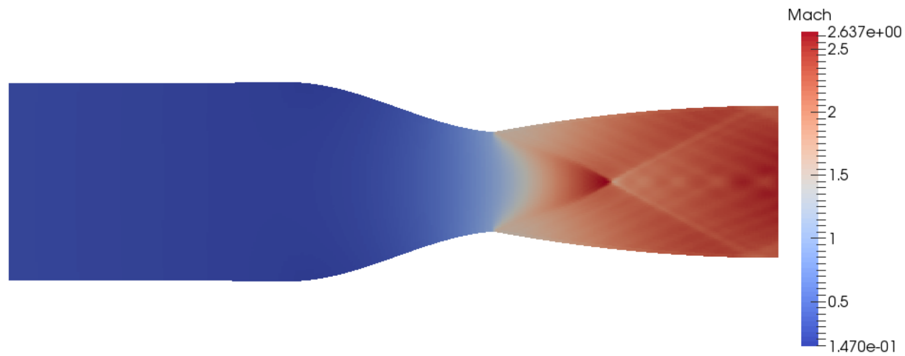


Figure 9: Axisymmetric Mach Number

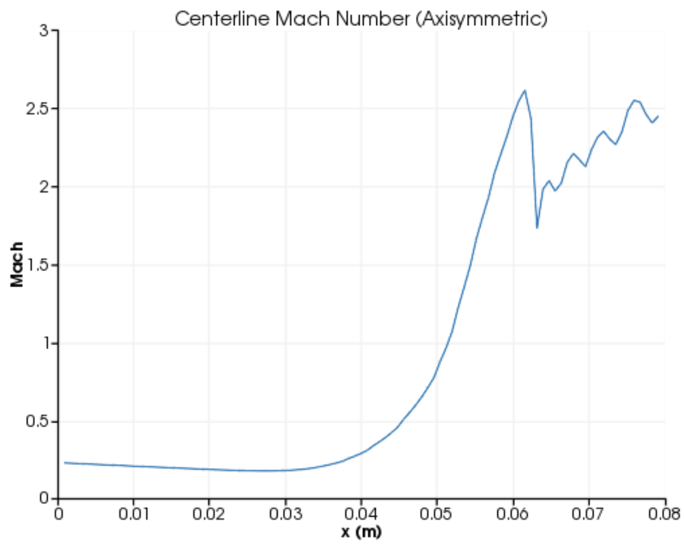
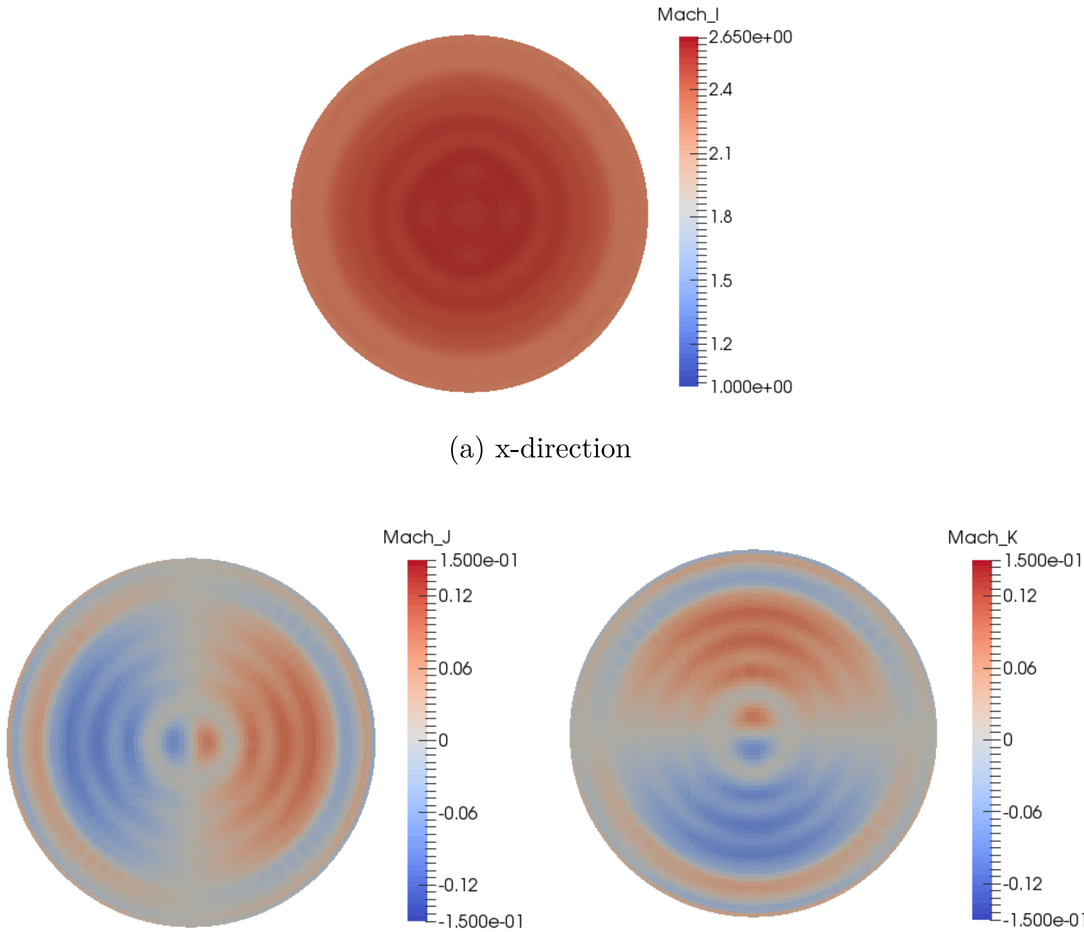


Figure 10: Axisymmetric Centerline Mach Number



(a) x-direction

(b) y-direction

(c) z-direction

Figure 11: Directional Mach Numbers at Exit Cross-Section

Summary

A method of characteristics code was developed to design the contour of a supersonic nozzle. The accuracy of the code was determined by comparing area ratios (A/A^*) to theoretical values as well as running inviscid CFD simulations with a code developed by CCL. The correct area ratios were met with reasonable accuracy. The error scaled by $\mathcal{O}\left(\frac{1}{N^2}\right)$ and $\mathcal{O}\left(\frac{1}{N}\right)$ for the planar and axisymmetric cases respectively. The planar nozzle successfully reached the design Mach number with uniform flow without any shock formation. The axisymmetric nozzle also reached the design Mach number (with some spacial variation throughout the exit plane) with nearly uniform flow, but shock waves were present. This is most likely due to over-expansion at the initial corner of the throat.

Appendix A - Tables

Table 1: Planar Nozzle Results

Theory		Number of Characteristics							
		5		**10**		**20**		**50**	
		Value	**Error**	**Value**	**Error**	**Value**	**Error**	**Value**	**Error**
M	1.5	1.5	0.0%	1.5	0.0%	1.5	0.0%	1.5	0.0%
A/A^*	1.176	1.180	0.317%	1.177	0.082%	1.176	0.018%	1.176	0.004%
M	2.0	2.0	0.0%	2.0	0.0%	2.0	0.0%	2.0	0.0%
A/A^*	1.687	1.712	1.498%	1.694	0.403%	1.689	0.110%	1.687	0.024%
M	2.5	2.5	0.0%	2.5	0.0%	2.5	0.0%	2.5	0.0%
A/A^*	2.637	2.747	4.166%	2.667	1.133%	2.646	0.328%	2.638	0.053%
M	3.0	3.0	0.0%	3.0	0.0%	3.0	0.0%	3.0	0.0%
A/A^*	4.235	4.626	9.222%	4.343	2.554%	4.271	0.840%	4.244	0.212%
M	4.0	4.0	0.0%	4.0	0.0%	4.0	0.0%	4.0	0.0%
A/A^*	10.72	13.954	30.168%	11.548	7.725%	11.008	2.689%	10.806	0.805%
M	5.0	5.0	0.0%	5.0	0.0%	5.0	0.0%	5.0	0.0%
A/A^*	25.00	44.135	76.540%	29.090	16.358%	26.395	5.578%	25.429	1.717%

Table 2: Axisymmetric Nozzle Results

Theory		**Number of Characteristics**							
		5		**10**		**20**		**50**	
		Value	**Error**	**Value**	**Error**	**Value**	**Error**	**Value**	**Error**
M	1.5	1.506	0.392%	1.506	0.427%	1.507	0.451%	1.507	0.472%
A/A^*	1.176	1.165	0.906%	1.171	0.414%	1.174	0.173%	1.176	0.030%
M	2.0	2.013	0.651%	2.014	0.693%	2.014	0.723%	2.015	0.752%
A/A^*	1.687	1.647	2.359%	1.666	1.247%	1.676	0.681%	1.681	0.336%
M	2.5	2.521	0.851%	2.522	0.889%	2.523	0.921%	2.524	0.956%
A/A^*	2.637	2.564	2.754%	2.591	1.755%	2.606	1.159%	2.617	0.760%
M	3.0	3.030	1.001%	3.031	1.032%	3.032	1.067%	3.033	1.112%
A/A^*	4.235	4.222	0.307%	4.199	0.840%	4.203	0.750%	4.211	0.566%
M	4.0	4.044	1.111%	4.046	1.147%	4.049	1.213%	4.052	1.302%
A/A^*	10.72	13.002	21.290%	11.651	8.680%	11.265	5.087%	11.113	3.663%
M	5.0	5.045	0.890%	5.050	0.995%	5.057	1.148%	5.066	1.329%
A/A^*	25.00	48.520	94.081%	33.505	34.019%	30.028	20.112%	28.669	14.674%

Appendix B - Planar Script

```
# Design a supersonic nozzle using 2D Method of Characteristics
# for isentropic, non-rotational flow
# Default values are M = 2, gamma = 1.4
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.axes as ax
from Math_Fxns import *
import sys
import csv

###########################
#        MoC Solver       #
###########################

def MOCsolver(Me,yL,Nwaves,G,iplot):
    xL = 0.0 #x-coordinate for start of nozzle expansion
    yL = (yL/2.0); #y-coordinate for start of nozzle expansion

    #Initialize variables
    x=np.zeros([Nwaves,Nwaves]) #x coordinates
    y=np.zeros([Nwaves,Nwaves]) #y coordinates
    M=np.zeros([Nwaves,Nwaves]) #mach number
    Mu=np.zeros([Nwaves,Nwaves]) #mach angle
    theta=np.zeros([Nwaves,Nwaves]) #angle relative to horizontal
    Nu = np.zeros([Nwaves,Nwaves]) #Prandtl-Meyer angle
    Km = np.zeros([Nwaves,Nwaves]) #K- values
        #(Constant along right running characteristic lines)
    Kp = np.zeros([Nwaves,Nwaves]) #K+ values
        #(Constant along left running characteristic lines)

    nuMax=M2nu(G,Me)/2.0 #Prandtl-Meyer function
                         #for design mach number
    thetaMax=nuMax; #Maximum angle at expansion
                    #corner of nozzle

    #Flow data for characteristic lines (1st iteration)
    theta[:,0] = (np.arange(thetaMax/Nwaves,thetaMax+
                  0.5*thetaMax/Nwaves,thetaMax/Nwaves))
    Nu[:,0] = (np.arange(thetaMax/Nwaves,thetaMax+
               0.5*thetaMax/Nwaves,thetaMax/Nwaves))
```

```python
for i in range(Nwaves):
    M[i,0]=nu2M(G,theta[i,0])
    Km[i,0]=theta[i,0]+Nu[i,0]
    Kp[i,0]=theta[i,0]-Nu[i,0]
    Mu[i,0]=np.arcsin(1.0/M[i,0])*180.0/np.pi

#Flow data for characteristic lines (2:end iterations)
for j in range(1,Nwaves):
    for i in range(Nwaves-j):
        if i==0:
            theta[i,j]=0.0
            Km[i,j]=Km[i+1,j-1]
            Kp[i,j]=2.0*theta[i,j]-Km[i,j]
            Nu[i,j]=0.5*(Km[i,j]-Kp[i,j])
            Mu[i,j]=nu2mu(G,Nu[i,j])
        else:
            Km[i,j]=Km[i+1,j-1]
            Kp[i,j]=Kp[i-1,j]
            theta[i,j]=0.5*(Km[i,j]+Kp[i,j])
            Nu[i,j]=0.5*(Km[i,j]-Kp[i,j])
            Mu[i,j]=nu2mu(G,Nu[i,j])

#Characteristic line coordinates (first C+ line)
y[0,0] = 0.0
x[0,0] = xL-yL/(np.tan((theta[0,0]-Mu[0,0])*np.pi/180.0))
for i in range(1,Nwaves):
    mp = np.tan((theta[i-1,0]+Mu[i-1,0])*np.pi/180.0)
    mm = np.tan((theta[i,0]-Mu[i,0])*np.pi/180.0)
    yi=(y[i-1,0]-mp*(yL/mm-xL+x[i-1,0]))/(1-mp/mm)
    xi=(yi-yL)/mm+xL
    x[i,0]=xi
    y[i,0]=yi

#Characteristic line coordinates
for j in range(1,Nwaves+1):
    for i in range(0,Nwaves-j):
        if i==0: #point is on symmetry axis (y=0)
            yi=0
            mm=np.tan((theta[i+1,j-1]-Mu[i+1,j-1])*np.pi/180.0)
            xi=(yi-y[i+1,j-1]+mm*x[i+1,j-1])/mm
            x[i,j]=xi
            y[i,j]=yi
```

```
        else:
            mp=np.tan((theta[i-1,j]+Mu[i-1,j])*np.pi/180.0)
            mm=np.tan((theta[i+1,j-1]-Mu[i+1,j-1])*np.pi/180)
            yi=((mm*(-y[i-1,j]/mp+x[i-1,j]-x[i+1,j-1])+
                 y[i+1,j-1])/(1-mm/mp))
            xi=(yi-y[i-1,j])/mp+x[i-1,j]
            x[i,j]=xi
            y[i,j]=yi

#Wall point data
xwall = np.zeros([Nwaves+1])
ywall = np.zeros([Nwaves+1])
thetaw = np.zeros([Nwaves+1])
Nuw = np.zeros([Nwaves+1])

xwall[0] = xL
ywall[0] = yL
thetaw[0] = thetaMax
Nuw[0] = nuMax

#Wall angles
for j in range(Nwaves):
    thetaw[j+1] = theta[Nwaves-j-1,j]
    Nuw[j+1] = Nu[Nwaves-j-1,j]

#Wall points
for j in range(1,Nwaves+1):
    mw=((np.tan((thetaw[j-1])*np.pi/180.0)+
         np.tan((thetaw[j])*np.pi/180.0))/2.0);
         #average wall slope between two points
    mp=(np.tan((theta[Nwaves-j,j-1]+
        Mu[Nwaves-j,j-1])*np.pi/180.0))
    yj=((mp*(-ywall[j-1]/mw+xwall[j-1]-x[Nwaves-j,j-1])+
         y[Nwaves-j,j-1])/(1-mp/mw))
    xj=(yj-ywall[j-1])/mw+xwall[j-1]
    xwall[j]=xj
    ywall[j]=yj

#Organize data
outlet_geom = [xwall,ywall]
outlet_geom = np.transpose(outlet_geom)
```

```
######################
#Plot nozzle geometry#
######################
if iplot == 1:

    #plot wall geometry
    plt.plot(xwall,ywall,'k')

    #plot first characteristics from nozzle throat
    for i in range(Nwaves):
        plt.plot([xL,x[i,0]],[yL, y[i,0]],'b',linewidth=0.5)

    #plot characteristics from wall
    for j in range(1,Nwaves+1):
        plt.plot([x[Nwaves-j,j-1],xwall[j]],[y[Nwaves-j,j-1],
                 ywall[j]],'b',linewidth=0.5)

    #plot inner characteristics
    for i in range(Nwaves-1):
        for j in range(Nwaves-i-1):
            plt.plot([x[i,j],x[i+1,j]],[y[i,j],
                     y[i+1,j]],'b',linewidth=0.5)
            plt.plot([x[i+1,j],x[i,j+1]],[y[i+1,j],
                     y[i,j+1]],'b',linewidth=0.5)
    #Plot settings
    plt.xlabel('Length␣[x]')
    plt.ylabel('Radius␣[y]')
    pltTitle = ('2D␣Method␣of␣Characteristics␣Nozzle\nMach␣=␣%.2f'
               % Me)
    plt.title(pltTitle)
    plt.axis('scaled')
    plt.xlim(xmin=xwall[0], xmax=1.05*xwall[-1])
    if np.max(ywall) >= np.max(yi):
        ymax = np.max(ywall)
    else:
        ymax = np.max(yi)
    plt.ylim(ymin=0, ymax = 1.05*ymax)
    plt.grid(True)
    plt.savefig('Output/2D_MoC_Nozzle.png')
    plt.show()

return outlet_geom
```

Appendix C - Axisymmetric Script

```
# Design a supersonic nozzle using 2D Method of Characteristics
# for isentropic, non-rotational flow
# Default values are M = 2, gamma = 1.4
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.axes as ax
from math import *
import scipy.optimize as opt
from Math_Fxns import *
import sys
import csv
import warnings
import copy

###########################
#        MoC Solver       #
###########################

def MOCsolverAxi(Me,r_star,Nwaves,G,iplot):
    xL = 0.0 #x-coordinate for start of nozzle expansion
    yL = r_star; #y-coordinate for start of nozzle expansion
    tol = 1.0e-4
    dnu = 1.0e-3

#    #Sauer solution at throat
#    r = np.linspace(0,yL,Nwaves)
#    in_curve = 2.0*r_star
#    alpha = sqrt(2.0/((G+1.0)*in_curve*yL))
#    eta = yL/8.0*sqrt(2.0*(G+1.0)*yL/in_curve)
#    x_star = -eta
#    u = alpha*x_star+(G+1)/4.0*alpha**2.0*yL**2.0
#    v = ((G+1.0)/2.0*(alpha**2.0*x_star*yL)+
#          alpha**3.0*yL**3.0*(G+1.0)**2.0/16.0)
#    mach = sqrt((1+u)**2.0+v**2.0)
#    x = []
#    r = np.linspace(0,yL,Nwaves)
#    for R in r:
#        x.append(-(G+1)/4.0*alpha*R**2.0)
#    x = x - np.min(x)
#    plt.scatter(x,r)
```

```
#    plt.axis('equal')
#    plt.show()

    #Initialize variables
    x = np.zeros([Nwaves,Nwaves]) #x coordinates
    y = np.zeros([Nwaves,Nwaves]) #y coordinates
    M = np.zeros([Nwaves,Nwaves]) #mach number
    Mu = np.zeros([Nwaves,Nwaves]) #mach angle
    theta = np.zeros([Nwaves,Nwaves]) #angle relative to horizontal
    Nu = np.zeros([Nwaves,Nwaves]) #Prandtl-Meyer angle
    Y = np.zeros([Nwaves,Nwaves]) #theta + nu
    Z = np.zeros([Nwaves,Nwaves]) #theta - nu

    nuMax=M2nu(G,Me)/2.0 #Prandtl-Meyer function for
                         #design mach number
    thetaMax=nuMax; #Maximum angle at expansion
                    #corner of nozzle

    #Initial data at entrance point of throat
    thetaL = np.zeros(Nwaves)
    #thetaL[:] = np.linspace(thetaMax*1.0e-2,thetaMax,Nwaves)
    thetaL[:] = np.arange(thetaMax/Nwaves,
          thetaMax+thetaMax/Nwaves/2.0,thetaMax/Nwaves)
    #thetaL[0] = 0.375
    #thetaL[1:] = np.linspace(3.375,nuMax,Nwaves-1)
    ML = np.ones(Nwaves)
    nuL = np.zeros(Nwaves)
    muL = np.zeros(Nwaves)
    YL = np.zeros(Nwaves)
    ZL = np.zeros(Nwaves)
    for i in range(Nwaves):
        nuL[i] = thetaL[i]
        muL[i] = nu2mu(G,nuL[i])
        YL[i] = thetaL[i] + nuL[i]
        ZL[i] = thetaL[i] - nuL[i]
        #ML[i] = nu2M(G,nuL[i])

    #Flow data for characteristic lines (1st iteration)
    for i in range(Nwaves):
        if i == 0:
            theta[0,i] = 0.0
            y[0,i] = 0.0
```

```
            x[0,i] = ((y[0,i]-yL)/tan((thetaL[i]-
                        muL[i])*np.pi/180.0) + xL)
            Y[0,i] = (1.0/(sqrt(ML[i]**2.0 - 1.0) -
                       1.0/tan(thetaL[i]*pi/180.0))/yL*(y[0,i]-yL)+
                       YL[i])
            Nu[0,i] = Y[0,i] - theta[0,i]
            Z[0,i] = theta[0,i] - Nu[0,i]
            Mu[0,i] = nu2mu(G,Nu[0,i])
            M[0,i] = nu2M(G,Nu[0,i])
        else:
            x[0,i] = ((tan((thetaL[i]-muL[i])*pi/180.0)*xL -
                 tan((theta[0,i-1]+Mu[0,i-1])*pi/180.0)*x[0,i-1]+
                 y[0,i-1] - yL)/(tan((thetaL[i]-muL[i])*pi/180.0)-
                 tan((theta[0,i-1]+Mu[0,i-1])*pi/180.0)))
            y[0,i] = (tan((thetaL[i]-
                 muL[i])*pi/180.0)*(x[0,i]-xL) + yL)
            if i == 1:
                Nu[0,i] = ((1.0/(sqrt(ML[i]**2.0-1.0)-
                 1.0/tan(thetaL[i]*pi/180.0))*1.0/yL*(y[0,i]-
                 yL)+(thetaL[i]+nuL[i])-(theta[0,i-1]-
                 Nu[0,i-1]))/2.0)
                theta[0,i] = (theta[0,i-1]-Nu[0,i-1])+Nu[0,i]
            else:
                Nu[0,i] = ((1.0/(sqrt(ML[i]**2.0-1.0)-
                 1.0/tan(thetaL[i]*pi/180.0))*1.0/yL*(y[0,i]-
                 yL)+(thetaL[i]+nuL[i])+1.0/(sqrt(M[0,i-1]**2.0-
                 1.0)+1.0/tan(theta[0,i-1]*pi/180.0))*1.0/y[0,
                 i-1]*(y[0,i]-y[0,i-1])-(theta[0,i-1]-
                 Nu[0,i-1]))/2.0)
                theta[0,i] = (-1.0/(sqrt(M[0,i-1]**2.0-1.0)+
                 1.0/tan(theta[0,i-1]*pi/180.0))*1.0/y[0,i-
                 1]*(y[0,i]-y[0,i-1])+(theta[0,i-1]-
                 Nu[0,i-1])+Nu[0,i])
            Y[0,i] = theta[0,i] + Nu[0,i]
            Z[0,i] = theta[0,i] - Nu[0,i]
            Mu[0,i] = nu2mu(G,Nu[0,i])
            M[0,i] = nu2M(G,Nu[0,i])

    #Flow data for characteristic lines (2:end iterations)
    for i in range(1,Nwaves):
        for j in range(Nwaves-i):
            if j == 0: #Center line (symmetry line)
```

```
        theta[i,j] = 0.0
        y[i,j] = 0.0
        x[i,j] = ((y[i,j]-y[i-1,j+1])/tan((theta[i-1,
         j+1]-Mu[i-1,j+1])*np.pi/180.0) + x[i-1,j+1])
         Y[i,j] = (1.0/(sqrt(M[i-1,j+1]**2.0 - 1.0) -
         1.0/tan(theta[i-1,j+1]*pi/180.0))/y[i-1,
         j+1]*(y[i,j]-y[i-1,j+1]) + Y[i-1,j+1])
        Nu[i,j] = Y[i,j] - theta[i,j]
        Z[i,j] = theta[i,j] - Nu[i,j]
        Mu[i,j] = nu2mu(G,Nu[i,j])
        M[i,j] = nu2M(G,Nu[i,j])
    else:
        x[i,j] = ((tan((theta[i-1,j+1]-Mu[i-1,
         j+1])*pi/180.0)*x[i-1,j+1]-tan((theta[i,
         j-1]+Mu[i,j-1])*pi/180.0)*x[i,j-1]+
         y[i,j-1]-y[i-1,j+1])/(tan((theta[i-1,j+1]-
         Mu[i-1,j+1])*pi/180.0)-tan((theta[i,j-1]+
         Mu[i,j-1])*pi/180.0)))
        y[i,j] = (tan((theta[i-1,j+1]-Mu[i-1,
         j+1])*pi/180.0)*(x[i,j]-x[i-1,j+1])+
         y[i-1,j+1])
        if j == 1:
            Nu[i,j] = ((1.0/(sqrt(M[i-1,j+1]**2.0-
             1.0)-1.0/tan(theta[i-1,
             j+1]*pi/180.0))*1.0/y[i-1,j+1]*(y[i,j]-
             y[i-1,j+1])+(theta[i-1,j+1]+Nu[i-1,j+1])-
             (theta[i,j-1]-Nu[i,j-1]))/2.0)
            theta[i,j] = (theta[i,j-1]-Nu[i,j-1])+Nu[i,j]
        else:
            Nu[i,j] = ((1.0/(sqrt(M[i-1,j+1]**2.0-
             1.0)-1.0/tan(theta[i-1,
             j+1]*pi/180.0))*1.0/y[i-1,
             j+1]*(y[i,j]-y[i-1,j+1])+
             (theta[i-1,j+1]+Nu[i-1,j+1])+
             1.0/(sqrt(M[i,j-1]**2.0-1.0)+
             1.0/tan(theta[i,
             j-1]*pi/180.0))*1.0/y[i,j-1]*(y[i,j]-
             y[i,j-1])-(theta[i,j-1]-Nu[i,j-1]))/2.0)
            theta[i,j] = (-1.0/(sqrt(M[i,j-1]**2.0-1.0)+
             1.0/tan(theta[i,j-1]*pi/180.0))*1.0/y[i,
             j-1]*(y[i,j]-y[i,j-1])+(theta[i,j-1]-
             Nu[i,j-1])+Nu[i,j])
```

```
            Y[i,j] = theta[i,j] + Nu[i,j]
            Z[i,j] = theta[i,j] - Nu[i,j]
            Mu[i,j] = nu2mu(G,Nu[i,j])
            M[i,j] = nu2M(G,Nu[i,j])

print np.transpose(theta)
print np.transpose(Nu)
print np.transpose(Mu)
print np.transpose(M)
print np.transpose(x)
print np.transpose(y)


#Wall points
xwall = np.zeros(Nwaves+1)
ywall = np.zeros(Nwaves+1)
thetaw = np.zeros(Nwaves+1)
Nuw = np.zeros(Nwaves+1)
Muw = np.zeros(Nwaves+1)
Mw = np.zeros(Nwaves+1)

xwall[0] = xL
ywall[0] = yL
thetaw[0] = thetaMax
Nuw[0] = nuMax
Muw[0] = nu2mu(G,Nuw[0])
Mw[0] = nu2M(G,Nuw[0])

for j in range(1,Nwaves+1):
    def myFunctions(z):
        [X,Y] = z
        eq = np.empty((2))
        eq[0] = (tan((thetaw[j-1]+
         thetaw[j])/4.0*np.pi/180.0)*(X-xwall[j-1])-
         (Y-ywall[j-1]))
        eq[1] = (tan((theta[j-1,Nwaves-j]+
         Mu[j-1,Nwaves-j])*np.pi/180.0)*(X-
         x[j-1,Nwaves-j])-(Y-y[j-1,Nwaves-j]))
        return eq

    thetaw[j] = theta[j-1,Nwaves-j]
    Mw[j] = M[j-1,Nwaves-j]
```

```
    Nuw[j] = Nu[j-1,Nwaves-j]
    Muw[j] = Mu[j-1,Nwaves-j]

    guess = np.array([xwall[j-1],ywall[j-1]])
    sol = opt.root(myFunctions, guess, method='hybr')
    xwall[j] = sol.x[0]
    ywall[j] = sol.x[1]

print xwall
print ywall
print thetaw
print Nuw
print Mw
print Muw

#Organize data
outlet_geom = [xwall,ywall]
outlet_geom = np.transpose(outlet_geom)

######################
#Plot nozzle geometry#
######################

#plot wall geometry
plt.plot(xwall,ywall,'k')

#plot first characteristics from nozzle throat
for i in range(Nwaves):
    plt.plot([xL, x[0,i]], [yL, y[0,i]], 'b', linewidth=0.5)

#plot characteristics from wall
for j in range(1,Nwaves+1):
    plt.plot([xwall[j],x[j-1,Nwaves-j]],
    [ywall[j],y[j-1,Nwaves-j]],'b', linewidth=0.5)

#plot inner characteristics
for i in range(Nwaves-1):
    for j in range(Nwaves-i-1):
        plt.plot([x[i,j], x[i,j+1]], [y[i,j],
                 y[i,j+1]], 'b', linewidth=0.5)
        plt.plot([x[i,j+1], x[i+1,j]], [y[i,j+1],
                  y[i+1,j]], 'b', linewidth=0.5)
```

```
    #Plot settings
    plt.xlabel('Length␣[x]')
    plt.ylabel('Radius␣[y]')
    pltTitle = ('Axisymmetric␣Method␣of␣Characteristics
Nozzle\nMach␣=␣%.2f' % Me)
    plt.title(pltTitle)
    plt.axis('scaled')
    plt.xlim(xmin=xwall[0], xmax=1.05*np.max(xwall))
    ymax = np.max(ywall)
    plt.ylim(ymin=0, ymax = 1.05*ymax)
    plt.grid(True)
    plt.savefig('Output/2D_MoC_Nozzle.png')
    if iplot == 1:
        plt.show()

    return outlet_geom
```

References

[Dumitrescu(1975)] Dumitrescu, L. Z., "Minimum Length Axisymmetric Laval Nozzles," *AIAA Journal*, Vol. 13, No. 4, 1975, p. 521. doi:10.2514/3.49743.

[Rao(1958)] Rao, G. V. R., "Exhaust Nozzle Contour for Optimum Thrust," *Journal of Jet Propulsion*, Vol. 28, No. 6, 1958, pp. 377–382. doi:10.2514/8.7324.

[Anderson Jr(1990)] Anderson Jr, J. D., *Modern Compressible Flow*, 2^{nd} ed., McGraw-Hill, New York, 1990, Chaps. 6,11, pp. 186–189,307–359.

[Loth et al.(1992)Loth, Baum, and Löhner] Loth, E., Baum, J., and Löhner, R., "Formation of Shocks Within Axisymmetric Nozzles," *AIAA Journal*, Vol. 30, No. 1, 1992, pp. 268–270. doi:10.2514/3.10909.