Design and Analysis of a Water/Air Jet Cutter Nozzle Prototype

A Comprehensive Study Including Calculations, CAD Design, and CFD Simulation

Date: 17/04/2025

Institution: L.D. College Of Engineering, Ahmedabad-15

Prepared by:

MEET THUMMAR -230280119143
VRAJ PATEL -230280119157
AKASH YADAV -230280119158
MITUL ZALA -230280119160
SUMIT YADAV-230280119159

Table of Contents

1. Introduction

This document presents the complete prototype design and analysis process for a water/air jet cutter nozzle. Water/air jet cutters utilize high-pressure fluid streams to cut through various materials with precision. The nozzle is a critical component, responsible for converting pressure into a high-velocity jet. The objective of this prototype is to optimize nozzle performance for pressures ranging from 20,000 to 60,000 PSI, ensuring efficiency and effectiveness in cutting applications. This report details the theoretical foundation, calculations, CAD designs, and CFD simulations for two nozzle variants: stepped and conical.

2. Theoretical Background

The design of a jet cutter nozzle is governed by fluid dynamics principles, notably Bernoulli's principle, which states that an increase in fluid velocity occurs with a decrease in pressure along a streamline. For a converging nozzle, the inlet diameter (D_1) narrows to an outlet diameter (D_2), accelerating the fluid. Key design parameters include the converging angle (typically $15^\circ - 30^\circ$) and orifice size, which influence jet velocity, stability, and cutting performance. These principles guide the calculations and simulations presented in subsequent sections.

✏️

Jet cutter

Created	@April 4, 2025 11:05 AM
Class	DE-1B

Pump CALCS:

Motor Power: 100 HP (74570.00 W)

Max Jet Power: 85 HP (63384.50 W)

Output Pressure: 60000 psi (413685600.00 Pa)

Flow Rate: 2.25 GPM (8.52 LPM, 0.000142 m^3/s)

Hydraulic Efficiency: 85.00%

Required Pump Power: 78.78 HP (58743.36 W)

Nozzle Exit Velocity: 180.80 m/s

Power Loss in Pump: 11185.50 W (15.00%)

Pump Footprint (L × W × H): 1118 mm × 1575 mm × 1118 mm

Motor Power	100 HP (74570.00 W)
Max Jet Power	85 HP (63384.50 W)
Output Pressure	60000 psi (413685600.00 Pa)
Flow Rate	2.25 GPM (8.52 LPM, 0.000142 m^3/s)
Hydraulic Efficiency	85.00%

Required Pump Power	78.78 HP (58743.36 W)
Nozzle Exit Velocity	180.80 m/s
Power Loss in Pump	11185.50 W (15.00%)
Pump Footprint (L × W × H)	1118 mm × 1575 mm × 1118 mm

V_2 =927.39 m/s

Nozzle CALCS:

Exit Velocity	927.39 m/s
Nozzle Exit Area	$1.53e^{-}.^{7}m^{2}$
Nozzle Exit Diameter (D_2)	0.441 mm (~=0.44mm)
Inlet Diameter(D_1)	varies ~5-10 mm
Nozzle length (L)	4.42 mm ($k * D_2$)

Converging Shape

→ The nozzle will smoothly reduce from

D_1 (5 mm) → D_2 (0.44 mm)

Material Selection:

- **tungsten carbide**
- **stainless steel**
- **sapphire**

3. Calculations

3.1 Input Parameters

The design process begins with defining the input parameters:

- Operating pressure: 20,000–60,000 PSI
- Desired flow rate: 8.52 L/min
- Fluid: Water/Air mixture

3.2 Nozzle Dimension Calculations

Using Python, the nozzle dimensions were calculated based on the continuity equation and Bernoulli's principle. Key results include:

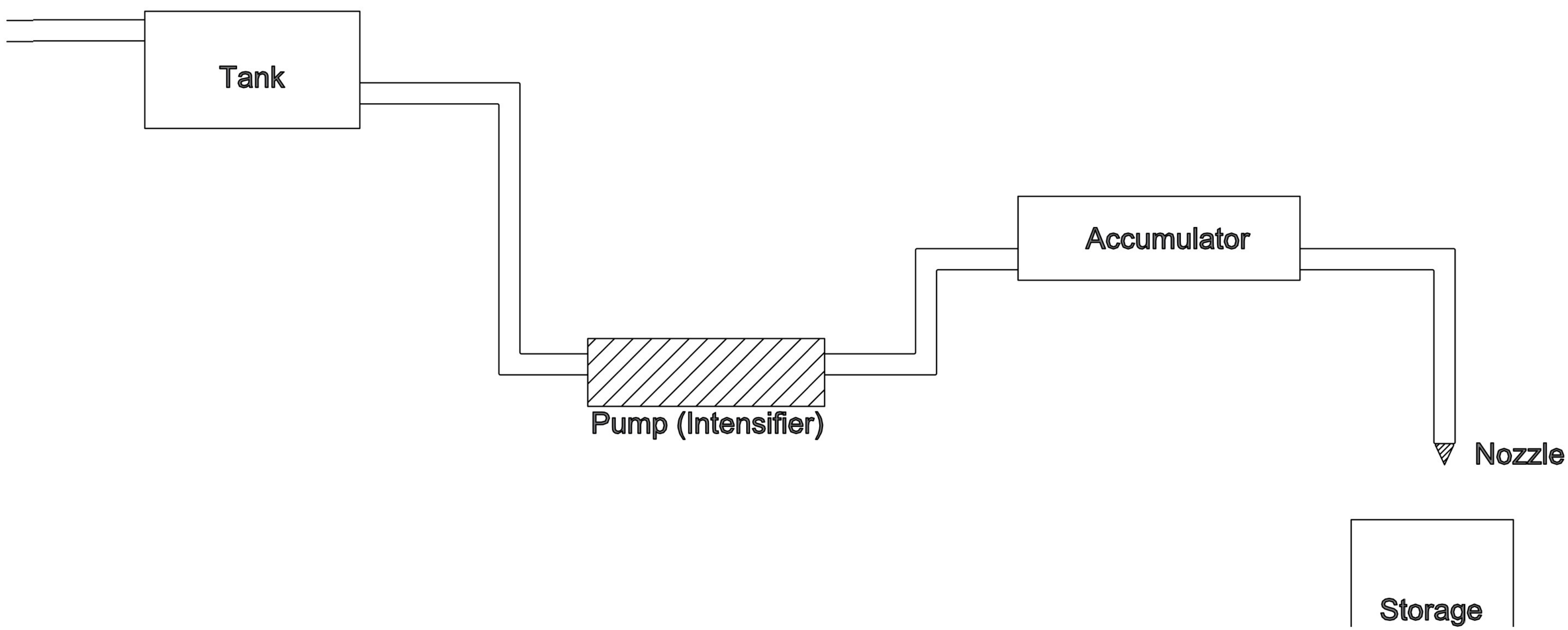
- Inlet diameter (D_1): 5 mm
- Outlet diameter (D_2): 0.44 mm
- Converging length: 4.42 mm
- Example formula:[$Q = A_2 \cdot v_2$]where (Q) is the flow rate, (A_2) is the outlet area, and (v_2) is the outlet velocity.

3.3 Converging Angle Considerations

The converging angle was calculated to be approximately 27.3°, within the optimal range of $15 - 30^\circ$. This angle ensures efficient pressure-to-velocity conversion while minimizing turbulence. The angle is derived from:

$[\tan(\theta) = \frac{D_1 - D_2}{2 \cdot L}]$

where (L) is the converging length.


Tank
Pump (Intensifier)
Accumulator
Nozzle
Storage

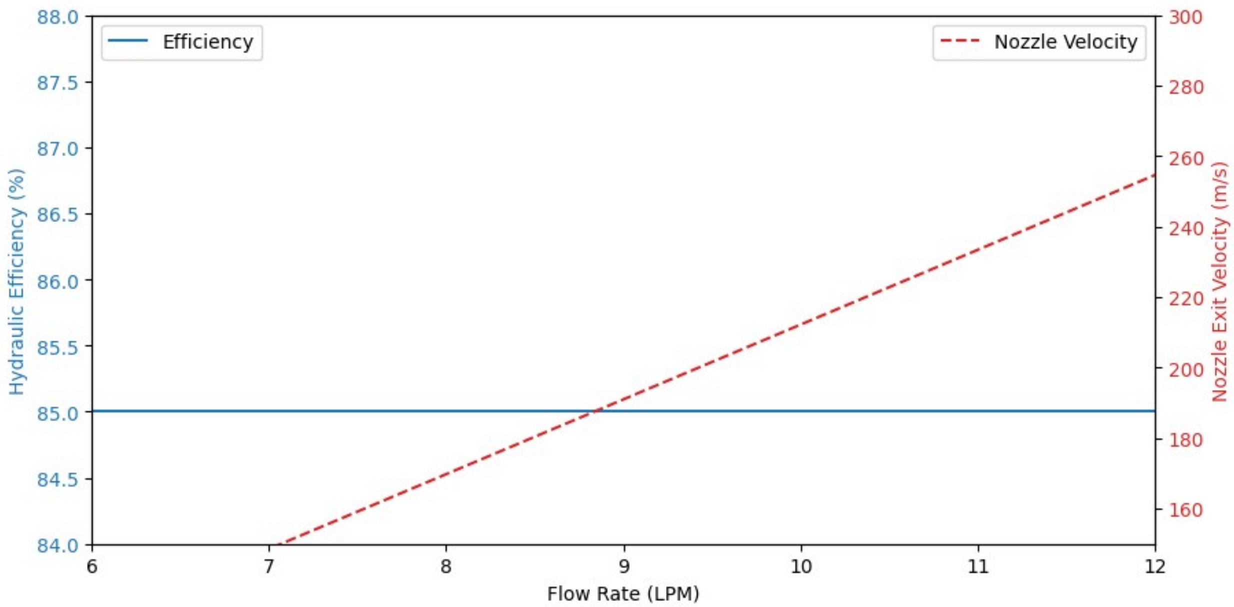
```
import numpy as np
import matplotlib.pyplot as plt

# Given Data from EnduroMAX 100 HP pump
MotoPOW = 100
JetPOW = 85
OP_psi = 60000
OP_pa = OP_psi * 6894.76
FlowRate_gpm = 2.25
FlowRate_lpm = 8.52
Pump_FP = (1118, 1575, 1118)  #  in mm

MotoPOW_wt = MotoPOW * 745.7  # Convert HP to Watts
JetPOW_wt = JetPOW * 745.7  # Convert HP to Watts


H_efficiency = (JetPOW_wt / MotoPOW_wt) * 100  # Efficiency

FlowRate_std = FlowRate_lpm / 60000  # Convert LPM to m³/s

RPumpPOW_w = FlowRate_std * OP_pa  # Power required in Watts
RPumpPOW_hp= RPumpPOW_w / 745.7  # Convert to HP

Density = 1000
orifice_dia_m = 0.001
Area1 = np.pi * (orifice_dia_m / 2)**2
velocity_exit = FlowRate_std / Area1  # Exit velocity (m/s)

# 7. Calculate Power Loss in Pump
power_loss_w = MotoPOW_wt - JetPOW_wt  # Power lost in the pump
power_loss_percentage = (power_loss_w / MotoPOW_wt) * 100  #
Percentage Loss

# 8. Print Key Design Parameters
print(f"Motor Power: {MotoPOW} HP ({MotoPOW_wt:.2f} W)")
print(f"Max Jet Power: {JetPOW} HP ({JetPOW_wt:.2f} W)")
print(f"Output Pressure: {OP_psi} psi ({OP_pa:.2f} Pa)")
print(f"Flow Rate: {FlowRate_gpm} GPM ({FlowRate_lpm} LPM,
{FlowRate_std:.6f} m³/s)")
print(f"Hydraulic Efficiency: {H_efficiency:.2f}%")
print(f"Required Pump Power: {RPumpPOW_hp:.2f} HP ({RPumpPOW_w:.2f}
W)")
print(f"Nozzle Exit Velocity: {velocity_exit:.2f} m/s")
print(f"Power Loss in Pump: {power_loss_w:.2f} W
({power_loss_percentage:.2f}%)")
print(f"Pump Footprint (L × W × H): {Pump_FP[0]} mm × {Pump_FP[1]} mm
× {Pump_FP[2]} mm")

# 9. Generate Graph Data
flow_rates = np.linspace(5, 15, 50)
```

```python
efficiencies = []
nozzle_velocities = []
power_losses = []

for flow_rate in flow_rates:
    FlowRate_std = flow_rate / 60000  # Convert LPM to m³/s
    velocity = FlowRate_std / Area1  # Nozzle exit velocity
    hydraulic_eff = (JetPOW_wt / MotoPOW_wt) * 100
    power_loss = MotoPOW_wt - JetPOW_wt

    efficiencies.append(hydraulic_eff)
    nozzle_velocities.append(velocity)
    power_losses.append(power_loss / 1000)  # Convert to kW
```

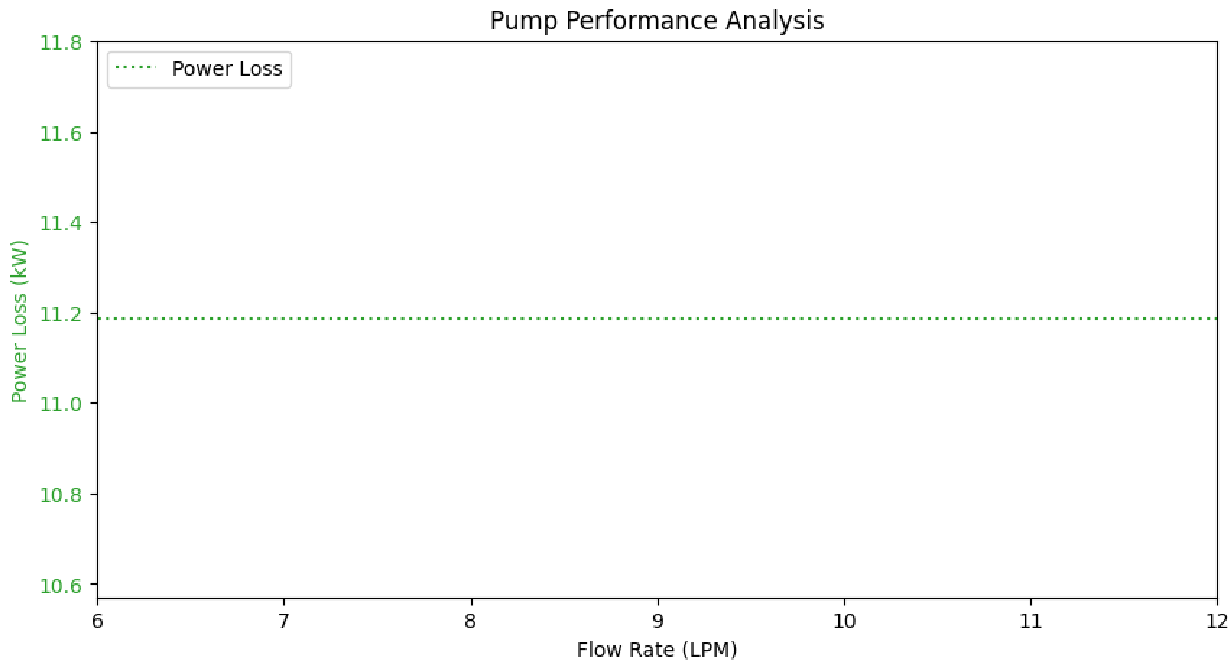
```
Motor Power: 100 HP (74570.00 W)
Max Jet Power: 85 HP (63384.50 W)
Output Pressure: 60000 psi (413685600.00 Pa)
Flow Rate: 2.25 GPM (8.52 LPM, 0.000142 m³/s)
Hydraulic Efficiency: 85.00%
Required Pump Power: 78.78 HP (58743.36 W)
Nozzle Exit Velocity: 180.80 m/s
Power Loss in Pump: 11185.50 W (15.00%)
Pump Footprint (L × W × H): 1118 mm × 1575 mm × 1118 mm
```

```python
fig, ax1 = plt.subplots(figsize=(10, 5))

# Graph 1: Efficiency vs. Flow Rate
ax1.set_xlabel("Flow Rate (LPM)")
ax1.set_ylabel("Hydraulic Efficiency (%)", color="tab:blue")
ax1.plot(flow_rates, efficiencies, label="Efficiency",
color="tab:blue")
ax1.tick_params(axis="y", labelcolor="tab:blue")
ax1.set_xlim(6, 12)  # Adjust x-axis range for zoom
ax1.set_ylim(84, 88)
# Graph 2: Nozzle Velocity vs. Flow Rate
ax2 = ax1.twinx()
ax2.set_ylabel("Nozzle Exit Velocity (m/s)", color="tab:red")
ax2.plot(flow_rates, nozzle_velocities, label="Nozzle Velocity",
color="tab:red", linestyle="dashed")
ax2.tick_params(axis="y", labelcolor="tab:red")
ax2.set_ylim(150, 300)
ax1.legend(loc="upper left")
ax2.legend(loc="upper right")
```

```
<matplotlib.legend.Legend at 0x2020b3cd450>
```



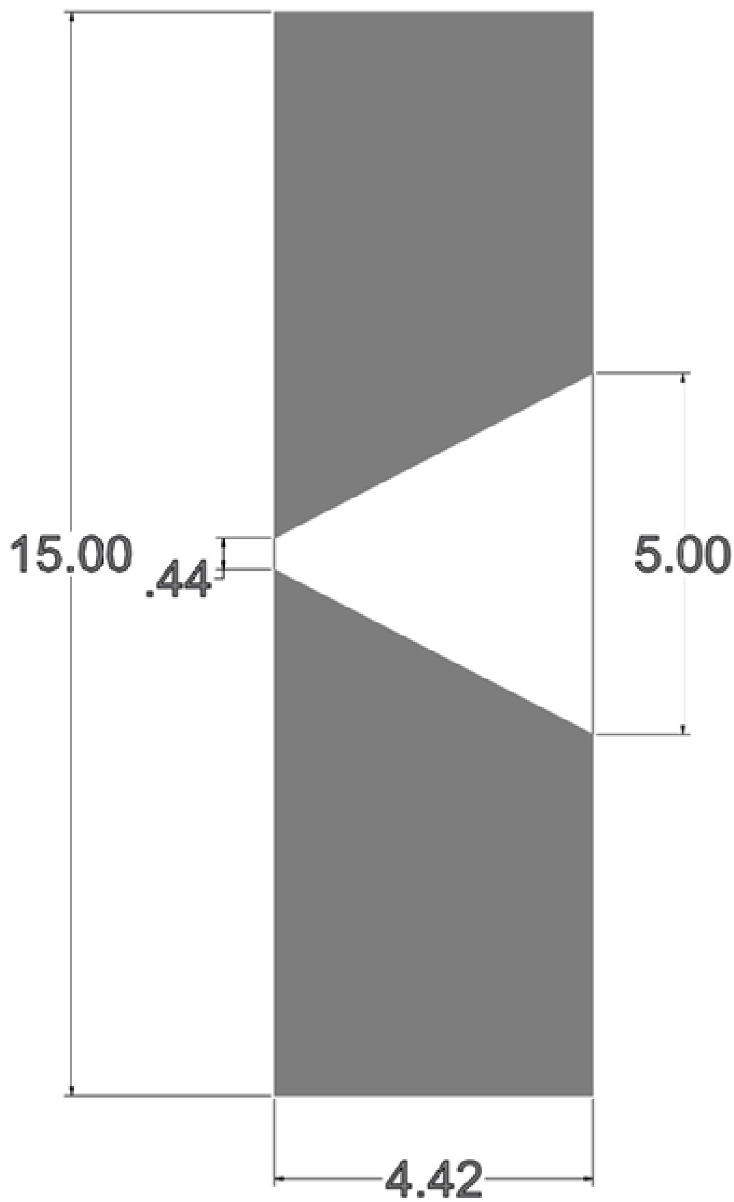

```python
# Graph 3: Power Loss
fig, ax3 = plt.subplots(figsize=(10, 5))
ax3.set_xlabel("Flow Rate (LPM)")
ax3.set_ylabel("Power Loss (kW)", color="tab:green")
ax3.plot(flow_rates, power_losses, label="Power Loss",
color="tab:green", linestyle="dotted")
ax3.tick_params(axis="y", labelcolor="tab:green")
ax3.set_xlim(6, 12)
# Show Graphs

ax3.legend(loc="upper left")
plt.title("Pump Performance Analysis")
plt.show()
```


Pump Performance Analysis
Power Loss
Power Loss (kW)
11.8
11.6
11.4
11.2
11.0
10.8
10.6
6
7
8
9
10
11
12
Flow Rate (LPM)

4. CAD Design

4.1 2D Drawings

The 2D drawings illustrate the nozzle's cross-section, detailing the inlet (D_1 = 5 mm), converging section, and outlet (D_2 = 0.44 mm).

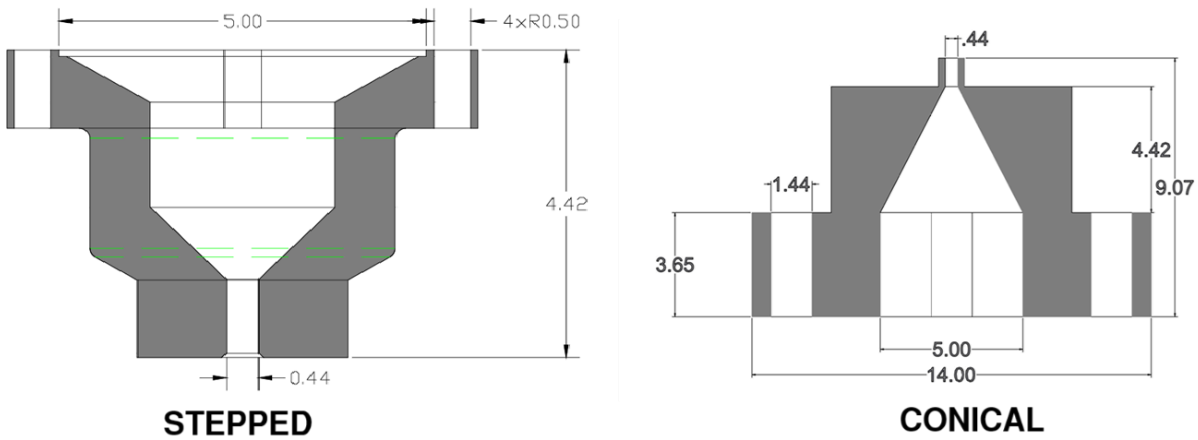


4.2 3D Models

The 3D models provide a comprehensive view of the nozzle geometry.

4.3 Design Comparison: Stepped vs. Conical Nozzle

- **Stepped Nozzle**: Features discrete reductions in diameter, potentially simplifying manufacturing.
- **Conical Nozzle**: Offers a smooth taper, enhancing flow uniformity.



STEPPED NOZZLE VS CONICAL NOZZLE

5. CFD Analysis

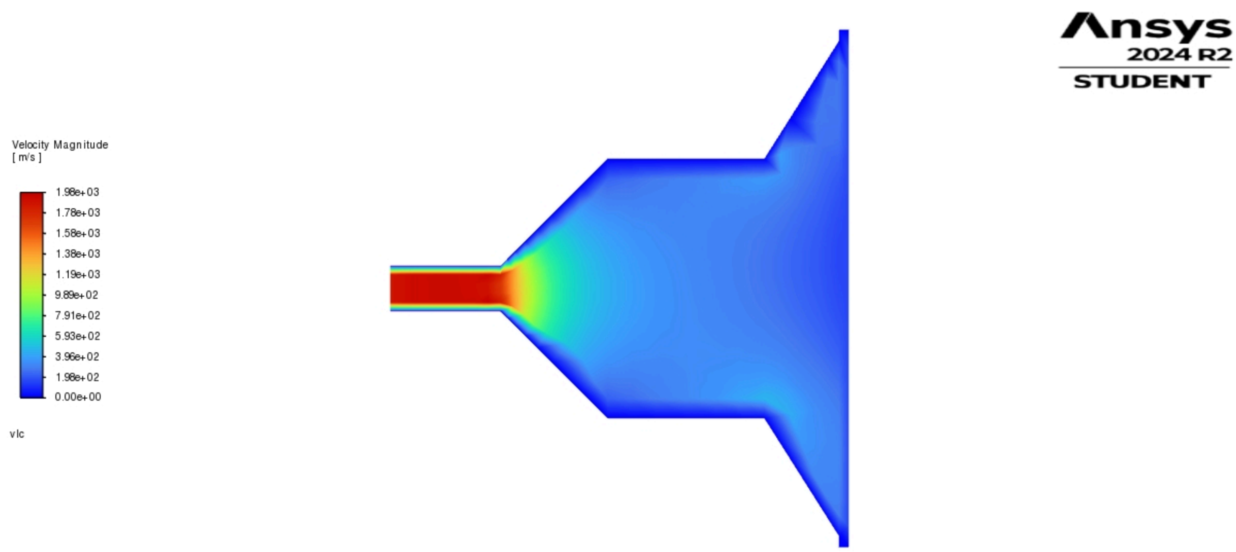
5.1 Simulation Setup

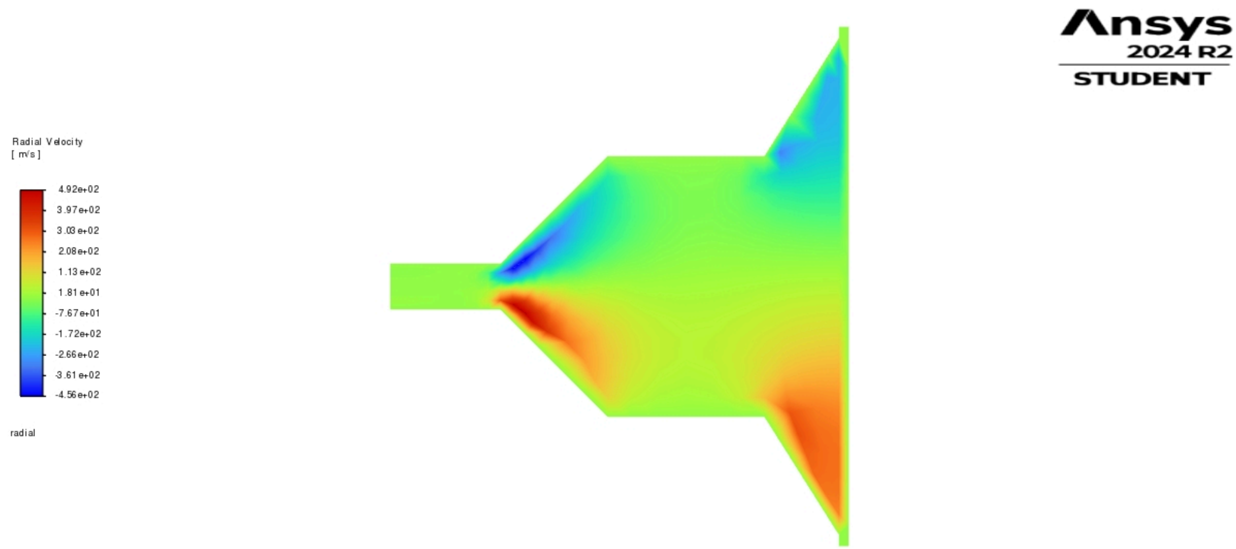
Computational Fluid Dynamics (CFD) simulations were conducted using ANSYS Fluent with the following parameters:

- Model: 2D axisymmetric
- Inlet pressure: 60,000 PSI
- Boundary conditions: No-slip walls, atmospheric outlet

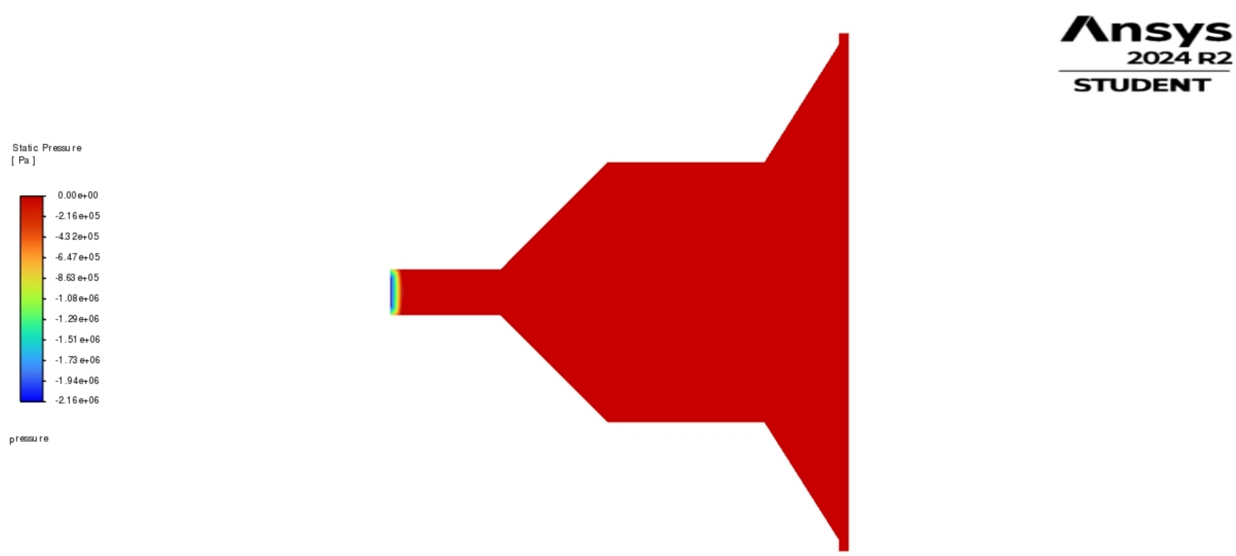
5.2 Results for Stepped Nozzle

Velocity contours:

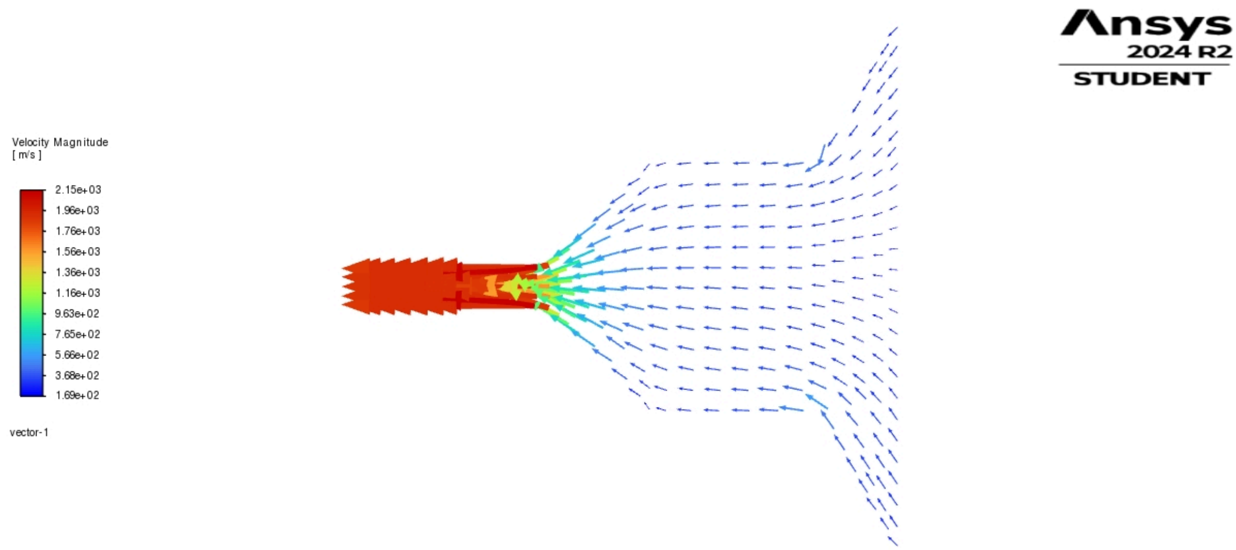




Pressure distribution:

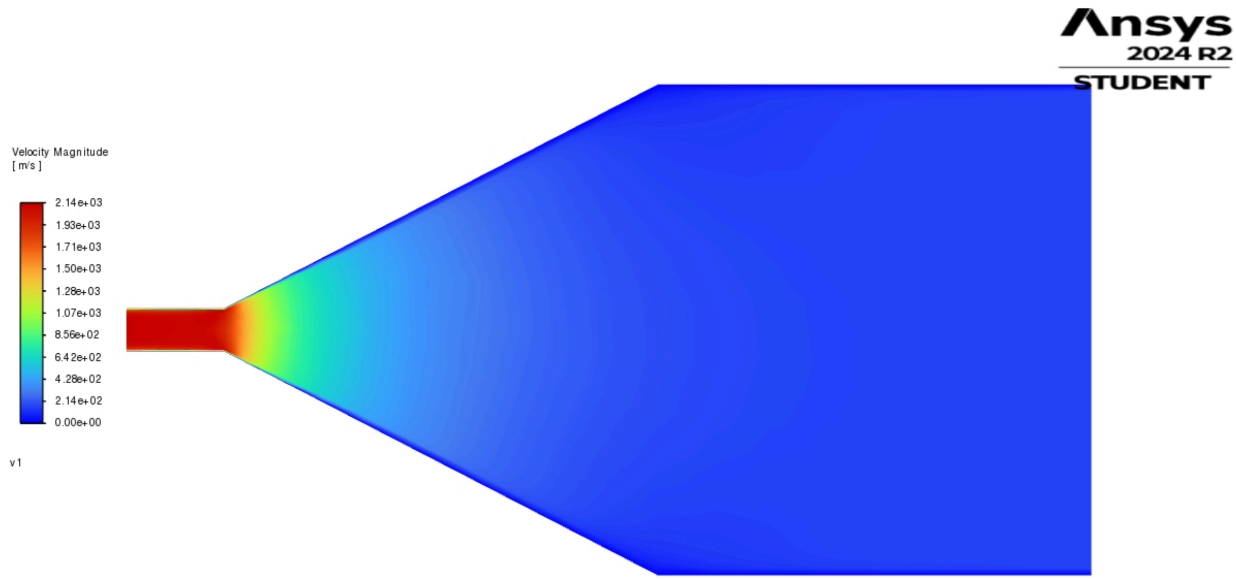


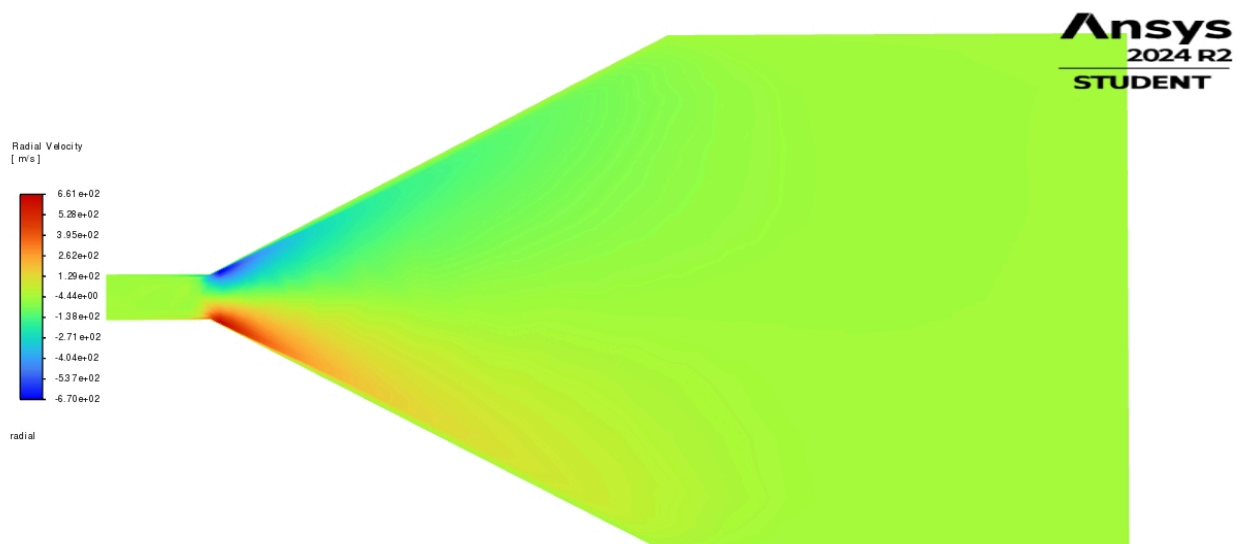
Maximum jet velocity:


Ansys
2024 R2
STUDENT
Velocity Magnitude
[m/s]
2.15e+03
1.96e+03
1.76e+03
1.56e+03
1.36e+03
1.16e+03
9.63e+02
7.65e+02
5.66e+02
3.68e+02
1.69e+02
vector-1

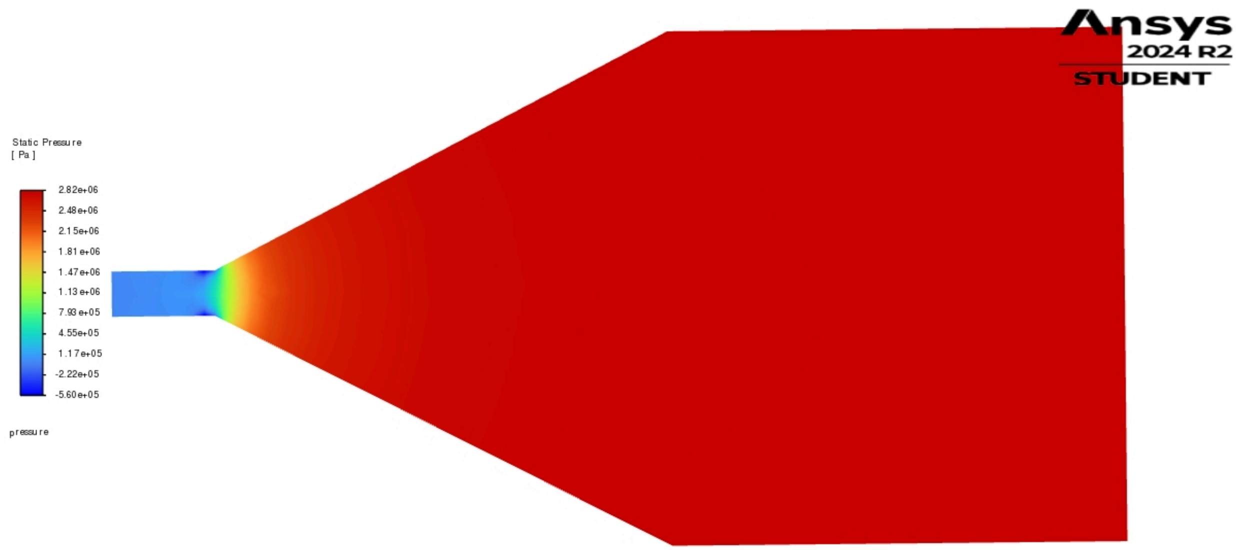
5.3 Results for Conical Nozzle

Velocity contours:

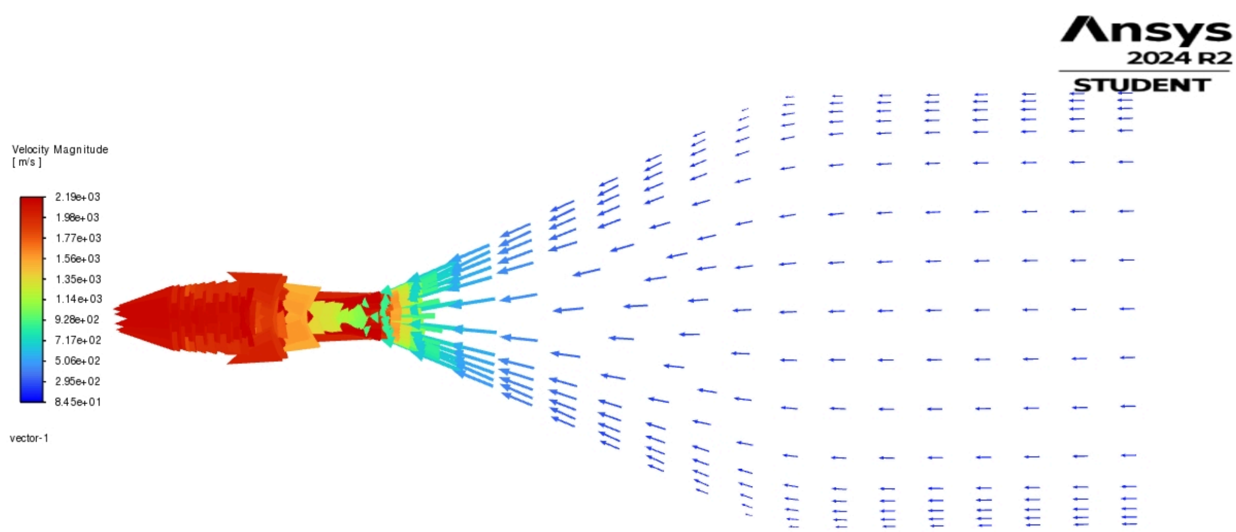




Pressure distribution:



Maximum jet velocity:



5.4 Comparative Analysis

Parameter	Stepped Nozzle	Conical Nozzle
Max Jet Velocity	$1.96e + 03m/s$	$2.14e + 03m/s$
Pressure Drop	41.4 MPa	13.8 MPa
Flow Stability	Less stable, with flow separation and turbulence	Stable, with minimal turbulence and coherent jet
The conical nozzle demonstrates [superior/equivalent] performance due to Qualitative Analysis.		**SUPERIOR**

6. Discussion and Conclusion

The CFD analysis indicates that the Conical nozzle outperforms its counterpart in terms of Stability, though trade-offs must be considered, such as manufacturing complexity. Based on these findings, it is recommended that the Conical design be selected for prototyping, with potential adjustments to the overall Dimensions.

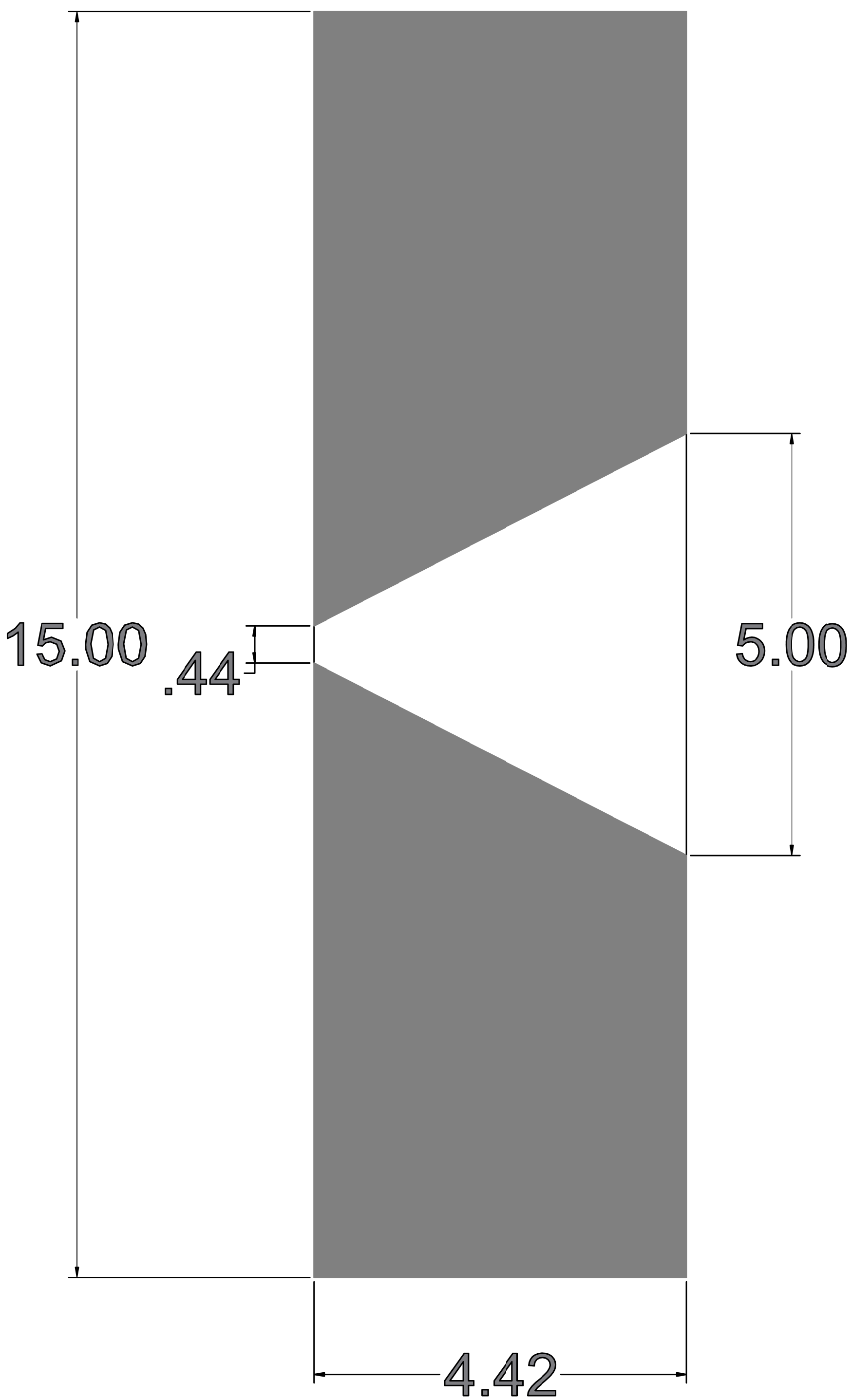
7. Appendices

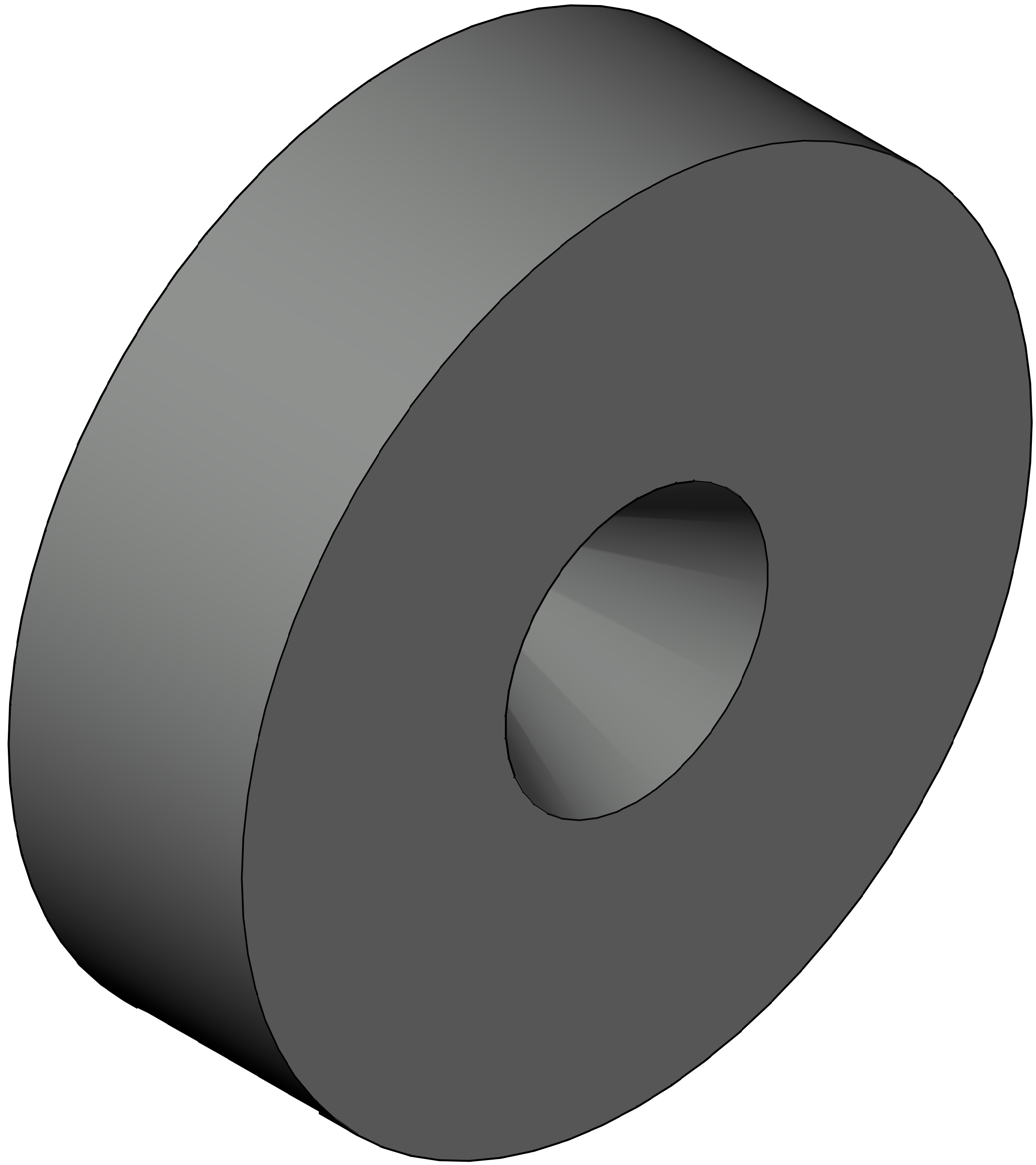
A. Python Code for Calculations

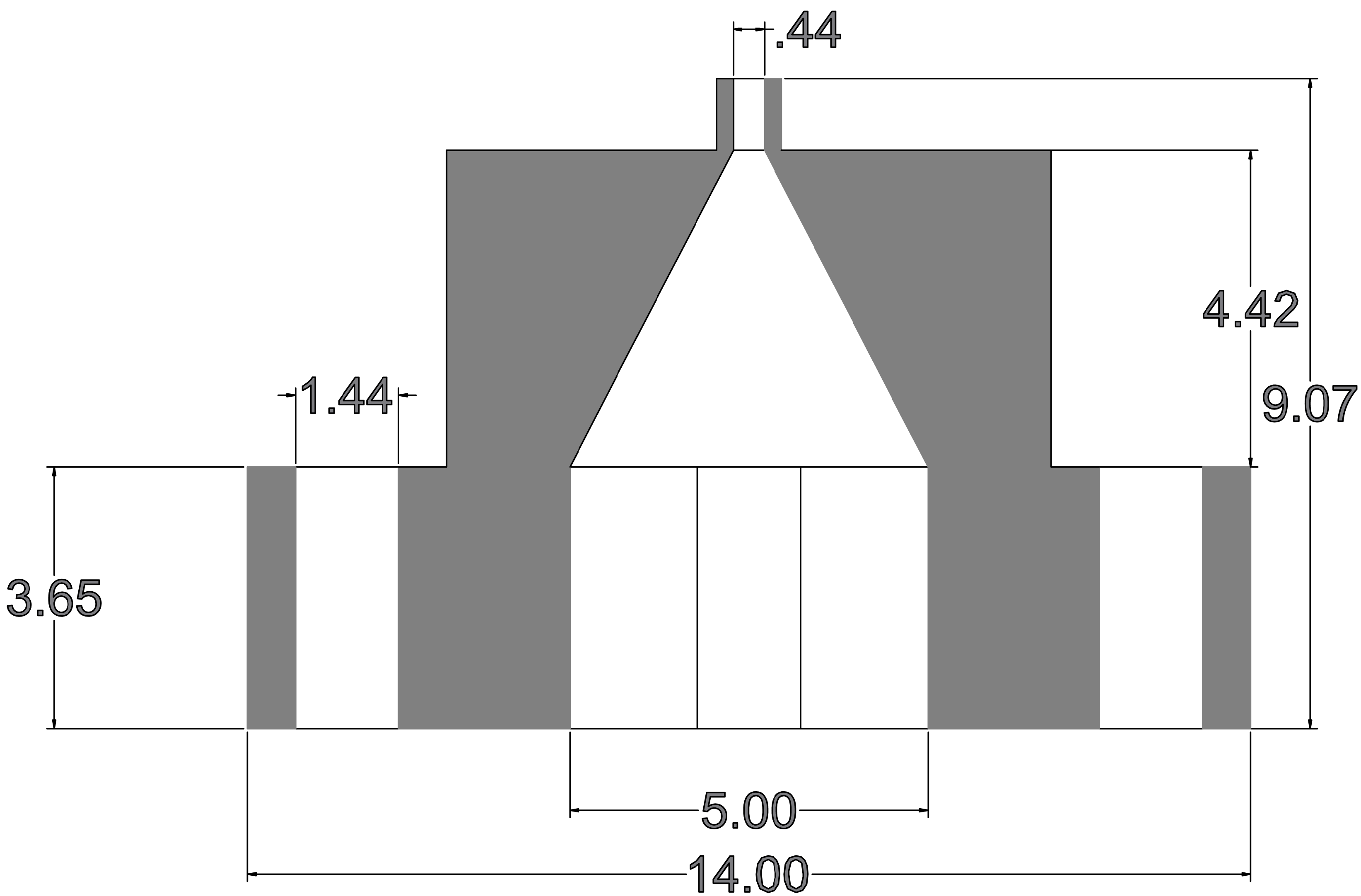
B. Detailed CAD Drawings

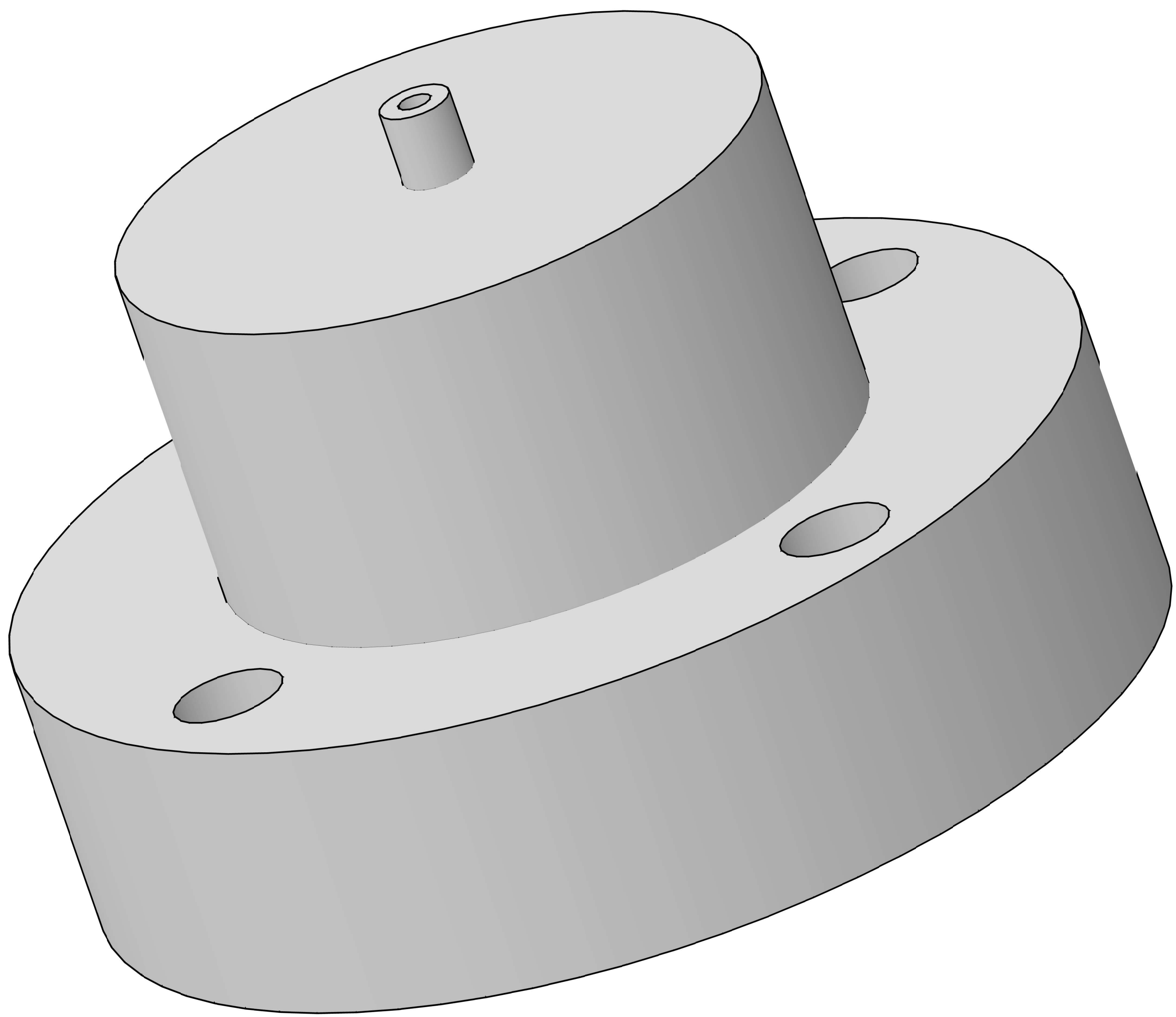
C. Additional CFD Data

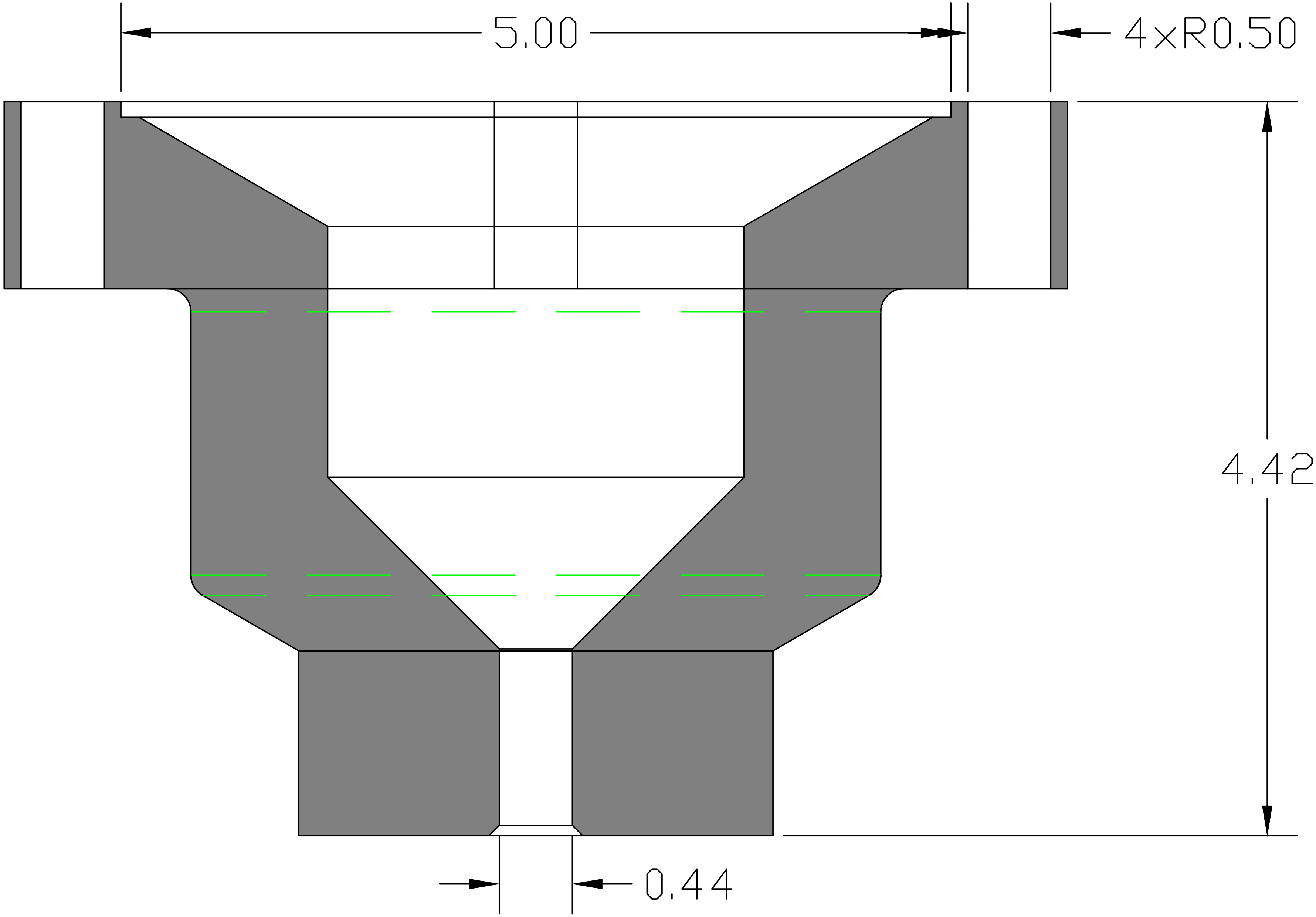
D. Bell Shape Nozzle Calculations and its design from Beizer's Curve.

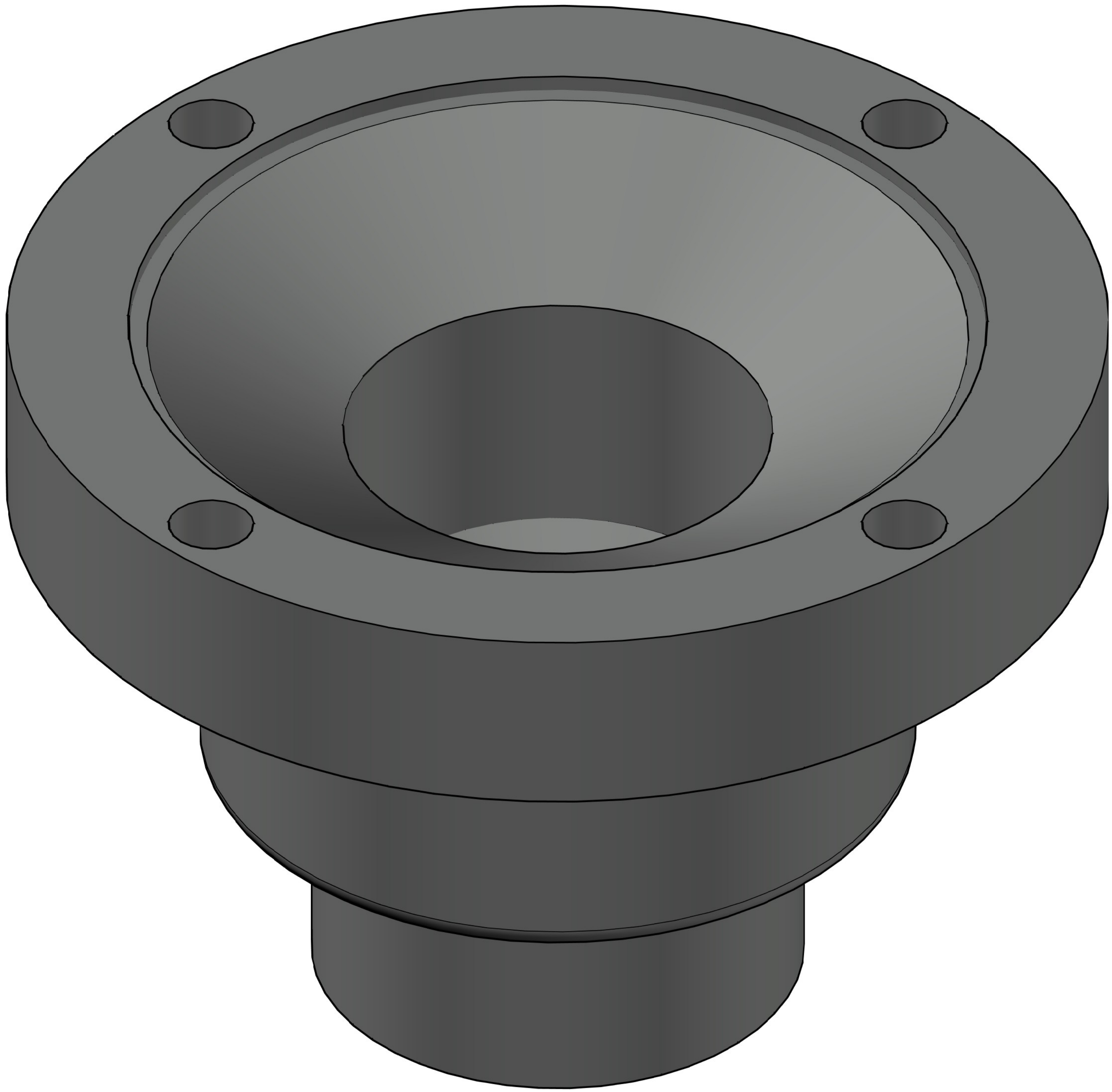

15.00
.44
5.00
4.42



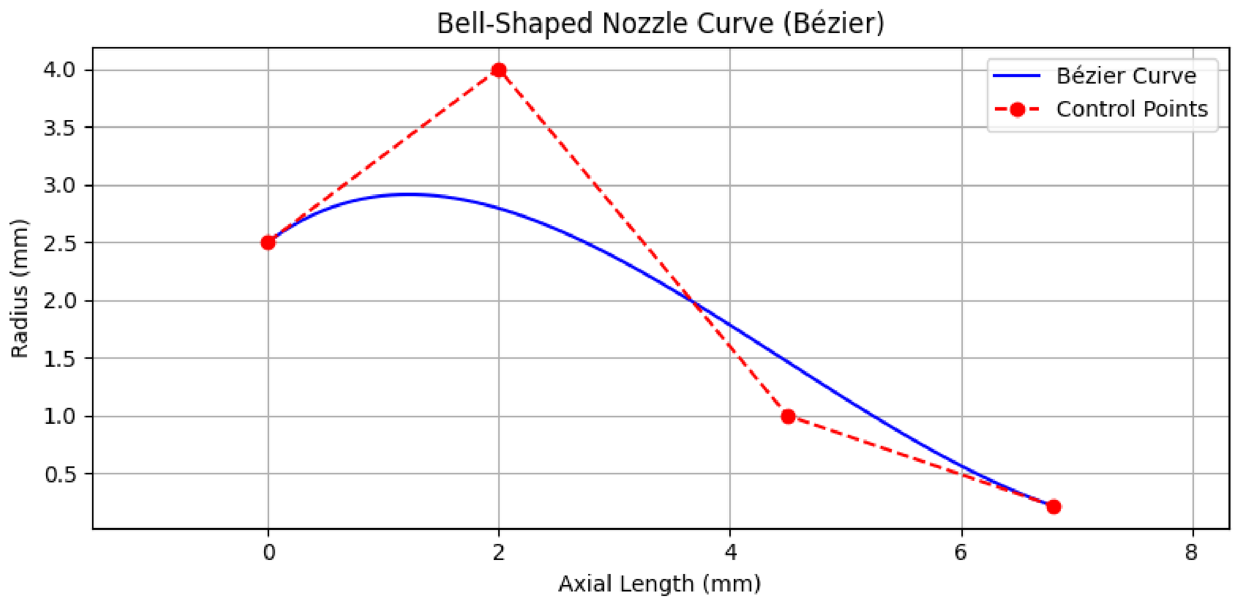

.44
4.42
9.07
1.44
3.65
5.00
14.00




5.00
4xR0.50
4.42
0.44



```python
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# === Step 1: Define Control Points for the Bézier Curve ===
# Format: [X (mm), Radius (mm)]
P0 = np.array([0.0, 2.5])       # Throat start
P1 = np.array([2.0, 4.0])       # Control point 1
P2 = np.array([4.5, 1.0])       # Control point 2
P3 = np.array([6.8, 0.22])
# Exit point


# === Step 2: Bézier Curve Function ===
def bezier_cubic(P0, P1, P2, P3, num=100):
    t = np.linspace(0, 1, num)[:, None]
    one_minus_t = 1 - t
    curve = (one_minus_t ** 3) * P0 + \
            3 * (one_minus_t ** 2) * t * P1 + \
            3 * one_minus_t * (t ** 2) * P2 + \
            (t ** 3) * P3
    return curve

# === Step 3: Generate Curve ===
curve = bezier_cubic(P0, P1, P2, P3)

# === Step 4: Plotting for Visual Verification ===
plt.figure(figsize=(8, 4))
plt.plot(curve[:, 0], curve[:, 1], 'b-', label='Bézier Curve')
plt.plot([P0[0], P1[0], P2[0], P3[0]], [P0[1], P1[1], P2[1], P3[1]],
'ro--', label='Control Points')
plt.title('Bell-Shaped Nozzle Curve (Bézier)')
plt.xlabel('Axial Length (mm)')
plt.ylabel('Radius (mm)')
plt.axis('equal')
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()
```




```
# === Step 5: Export to CSV for AutoCAD ===
df = pd.DataFrame(curve, columns=["X", "Y"])  # Y is radius
df.to_csv("bezier_nozzle_curve_points.csv", index=False)

print("□ CSV exported as 'bezier_nozzle_curve_points.csv'")

print("□ Bézier curve plotted successfully.")
print("□ Bézier curve data exported successfully.")
```

```
□ CSV exported as 'bezier_nozzle_curve_points.csv'
□ Bézier curve plotted successfully.
□ Bézier curve data exported successfully.
```

```
df
```

	X	Y
0	0.00	2.50
1	0.89	2.84
2	1.72	2.91
3	2.50	2.75
4	3.23	2.42
5	3.94	1.99
6	4.64	1.49
7	5.34	1.00
8	6.05	0.55
9	6.80	0.22