MEMORANDUM

To:	Frank Jumonville, Department of Mechanical Engineering, Cal Poly SLO fjumonvi@calpoly.edu	
From:	Ari Dennis atdennis@calpoly.edu	Baxter Mercy bmercy@calpoly.edu
Date:	June 5th, 2025	
RE:	ME 443-01 Turbine Project Report	

Introduction:

As part of the ME 443 curriculum, our team participated in Professor Jumonville's 2025 Hydropower Contest, a hands-on engineering challenge focused on the design, construction, and testing of a gravity-powered water turbine. The objective was to build a turbine system capable of pulling a one-Newton weighted sled over a 20-foot stretch of asphalt using only one liter of gravitational water energy. The contest featured a double-elimination bracket and emphasized efficiency, mechanical ingenuity, and safe operation. This memo outlines our design approach, construction process, testing results, and reflections on our turbine's performance throughout the competition.

Design and Testing:

For this project we utilized a Pelton wheel design to achieve the design goal of pulling the one-newton weight. Over the course of the project, the bucket size, pitch diameter, frame, fluid system, and pulley diameter stayed the same. In our initial design that failed to pull the weight the first day of racing had the wheel assembly mounted horizontally and used a plastic, 3-D printed shaft, and cheap, unsealed bearings.

During initial testing of this first iteration, we were able to pull a weight that was similar to the weight to be used during testing, however, was likely a slight bit lighter as weight values for pennies, an empty plastic water bottle, and coat hanger were found on the internet as opposed to actually being weighed due to not having a scale on hand. Additionally, the testing was done with a surface that was likely smoother than the one that the race was conducted on. We also later found out that our method of mounting the nozzles did not result in a repeatable nozzle angle which, upon failing during the first race, we realized to be an integral part to ensuring consistent success.

We also found that the 3-D printed shafts were nearly impossible to make concentric and when mounted between two bearings, required a significant amount of force to rotate. Our unshielded, standard steel bearings also proved to cause issues. After a couple tests they would start to rust and seize up.

***Figure 1**. First design Iteration*

Our second design iteration was motivated by fixing the issues with nozzle angle and the losses with the shaft. We modeled the nozzles into modules that could slide on rails, allowing us to maintain a constant nozzle angle, seen in Figure 2. We performed basic geometry, appended at the bottom, to solve for where the nozzles should be positioned to achieve an angle of 22°. In this second design we designed holes in the bottom of the frame so that it can slide onto screws in the frame, seen in Figure 3. This way the frame in Figure 2 was fixed in place.

To fix the losses associated with the with bad bearings and a non-concentric PLA shaft, we purchased a keyed metal shaft and shielded, high-precision ball bearings. These two optimizations yielded a system that was significantly more efficient and effective. The changes allowed us to consistently pull the weight 20 ft or more. The series of tests are outlined in Table 1.

It is important to note that much of our analysis was done in MATLAB and we used that analysis to design the Pelton wheel, however, these calculations did not take bearing or other mechanical losses into account, much of our design changes were a result of testing as opposed to analytical calculations. Due to time limitations between the first and second rounds of racing, our primary goal was reliable pulling of the weight. There is much more room for improvement regarding pulling speed. The entire system is pictured in Figure 5.

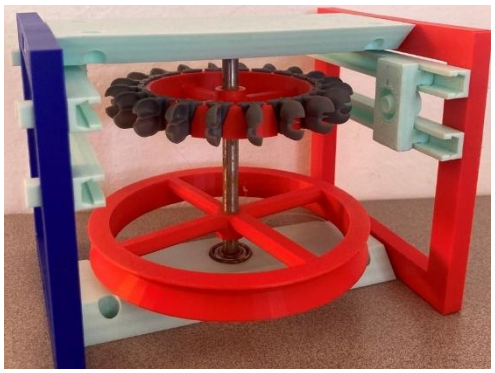


Figure 2. *Frame with sliding rails to allow for nozzle positioning*

Figure 3. *Frame with holes sitting behind screws for mounting*

Figure 4*. Resin printed blades, which helped minimize any losses that would occur from 3-D printer lines effecting flow.*

Table 1. Testing Results

Trial #	Distance Pulled [ft]	Time [seconds]	Item adjusted	Notes
1	15	5	N/A	This was our first run and set a baseline
2	20	6	Nozzle position/angle	This placement yielded 22 ° to the pitch circle
3	20	7	Nozzle position/angle	
4	25	4.7	Nozzle position/angle	We had extra tube length which we cut out
5	23	5	Amount of water in reservoir	We found where 1 liter sat in reservoir
Race day	20	4.3	None	This roughly coincides with our testing results expectations.

Figure 5. Final system design

Table 2. Final design parameters

Number of nozzles	2
Number of separate reservoirs	2
Reservoir height	63 inches
Number of tubes	2
Length of tubes	56 inches
Nozzle diameter	4.5 mm
Number of buckets	20
Pulley diameter	126 mm

Conclusion:

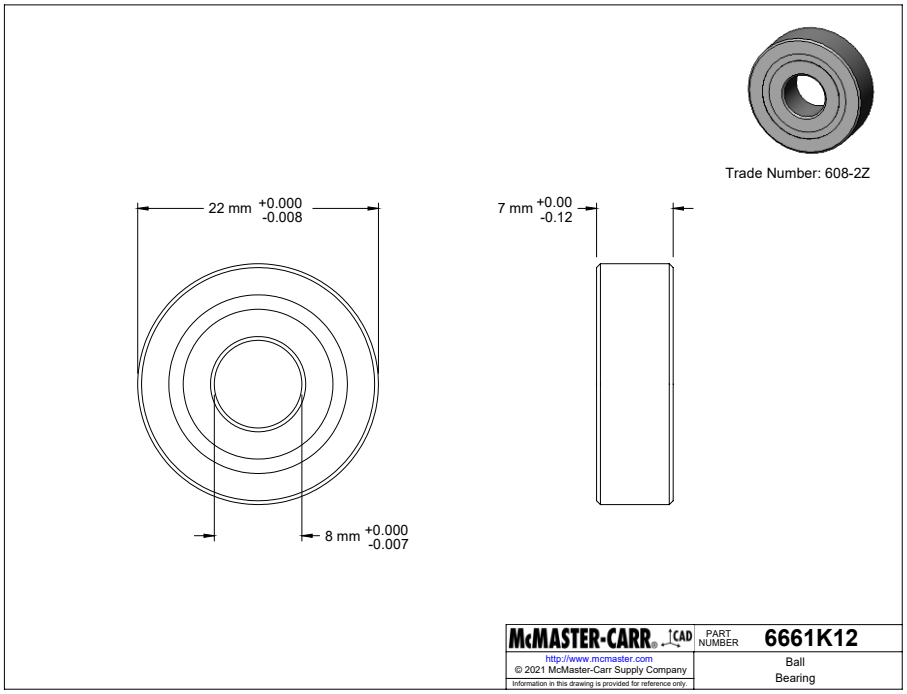
We learned that when optimizing a system it is important to consider all of the potential factors and decide which are most important. As a system becomes more and more optimized there will likely always be a weak link, or something that creates the most system losses when compared to other potential factors.

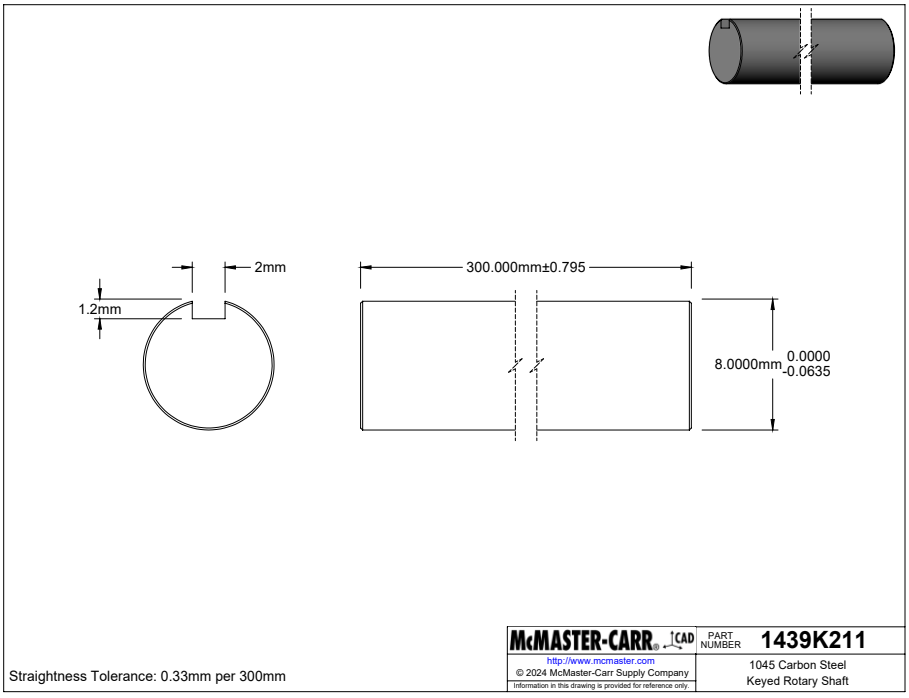
We focused on the optimization of our system around the angles of the nozzles and friction in the bearings. We learned that the pully diameter is very important. With too large of a pulley, the system struggles at the start when the coefficient of static friction needs to be broken. On the other hand, if the pulley wheel is too small, it will do well at the start, but the once the weight is moving it will start to become inefficient. While it is certainly possible to create a pulley that starts at a small diameter and jumps up to a larger diameter, it is a more complicated solution. Alternatively, as the operator we could choose to have the string on the pulley start at no distance from the axis so it has a very low initial torque and then as the pulley wheel rotates the string will simultaneously be moving farther away from the axis of rotation thus creating a larger torque so the system can be optimal for the time the weight is moving. Furthermore, we learned that having just a little bit of extra tubing can add huge amounts of loss to the system due to frictional losses. This was a surprising result, but it does make sense. Moreover, we learned of the importance of optimizing the fluid systems. If we continued this project this is what would drive our future iterations.

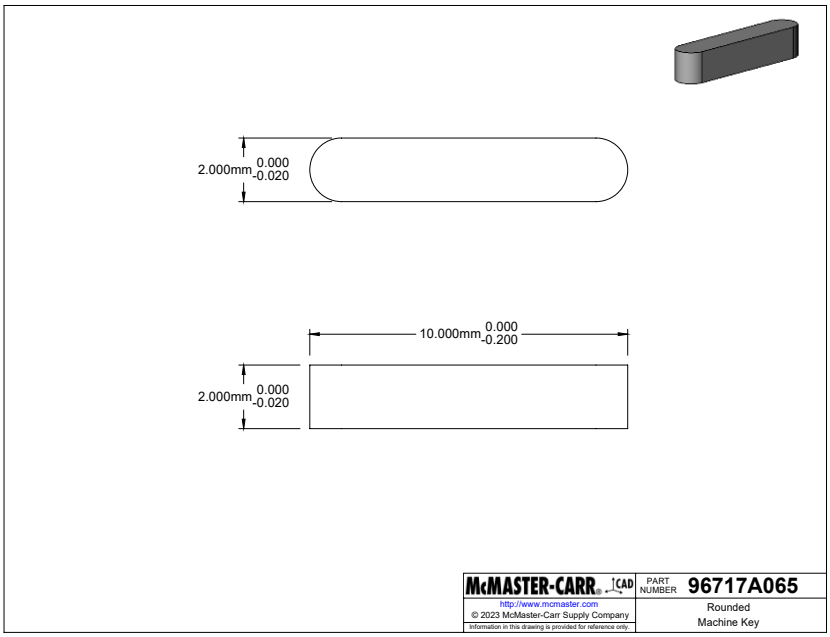
We believe that Professor Jumonville had the optimal fluid system, and the more successful designs of our peers were the ones that utilized a system similar to his. This was accomplished by using multiple tubes to make a larger effective tube diameter, allowing for a bigger nozzle. Also, by maximizing the number of tubes the system can hold the most amount of water. It was great to see how many ways this design problem could be approached. While our primary focus was system reliability, other teams focused on the fluid system. Finally, while testing provided valuable insights, it took ample time. As goes the old saying “measure twice and cut once” we would have likely benefited from more analytical analysis beyond just bucket and nozzle design.

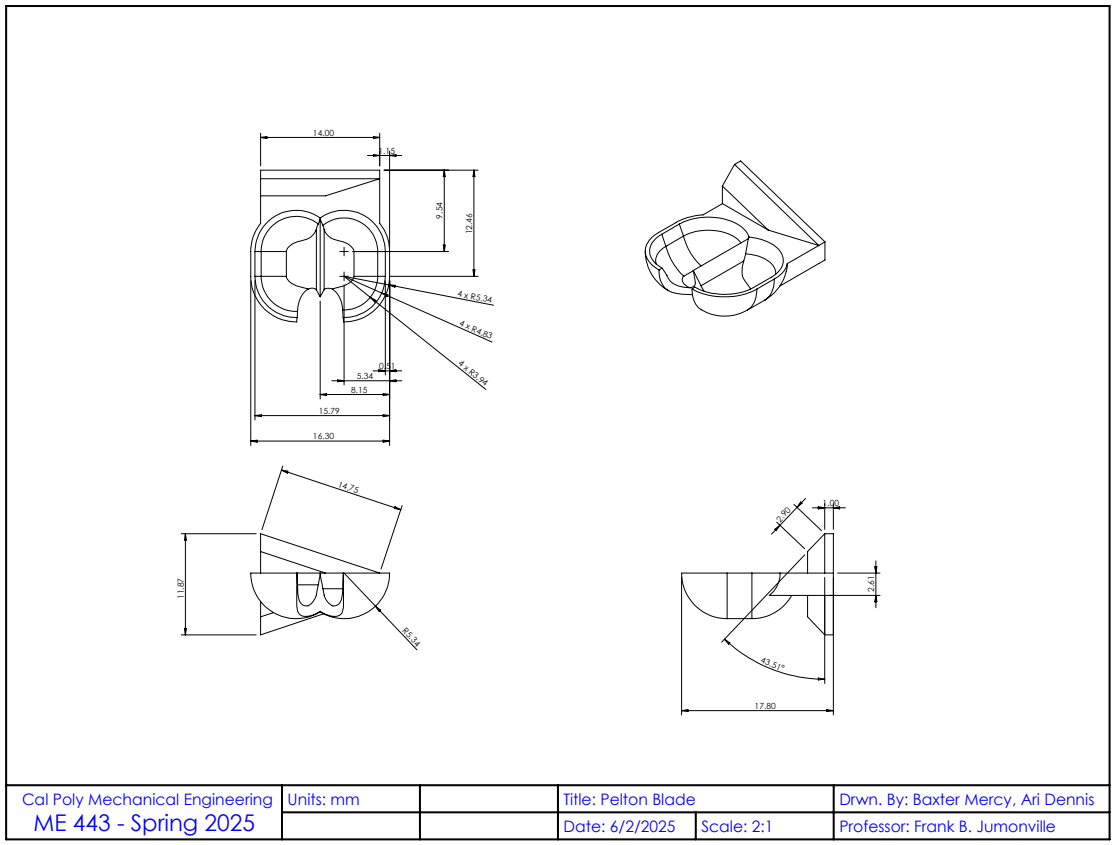
Appendix

Part Drawings

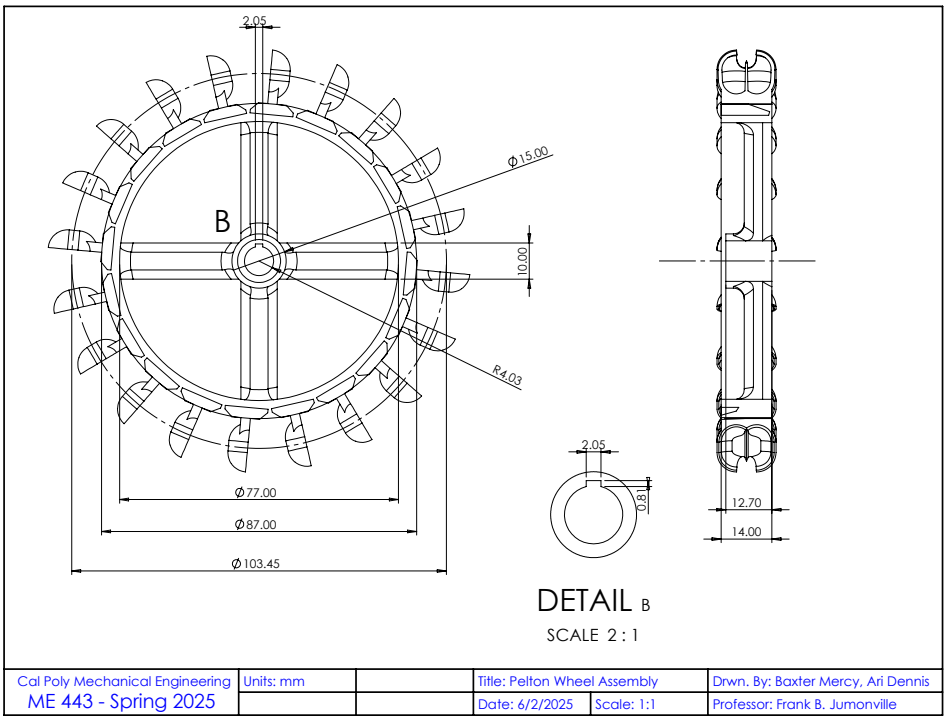








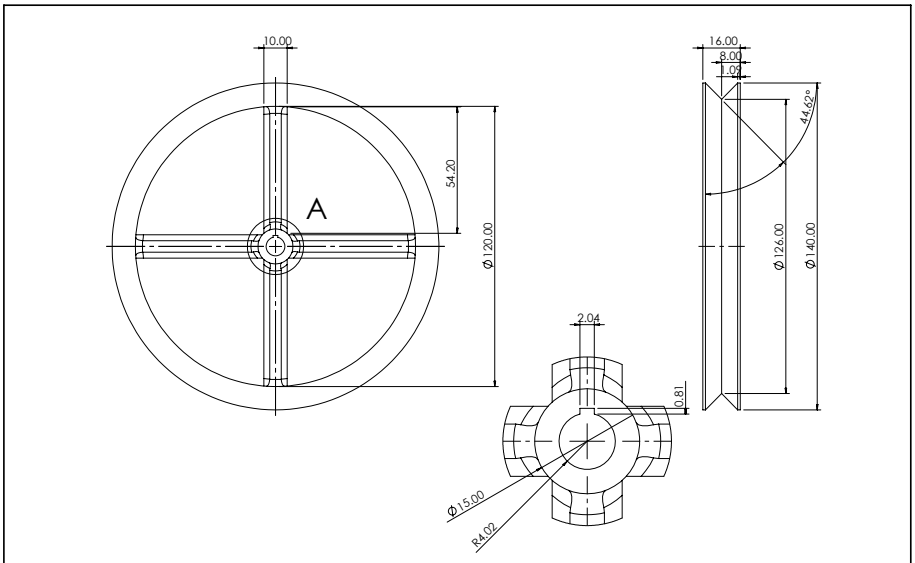
Cal Poly Mechanical Engineering ME 443 - Spring 2025	Units: mm		Title: Pelton Blade		Drwn. By: Baxter Mercy, Ari Dennis
			Date: 6/2/2025	Scale: 2:1	Professor: Frank B. Jumonville



DETAIL B

SCALE 2 : 1

Cal Poly Mechanical Engineering ME 443 - Spring 2025	Units: mm		Title: Pelton Wheel Assembly		Drwn. By: Baxter Mercy, Ari Dennis
			Date: 6/2/2025	Scale: 1:1	Professor: Frank B. Jumonville

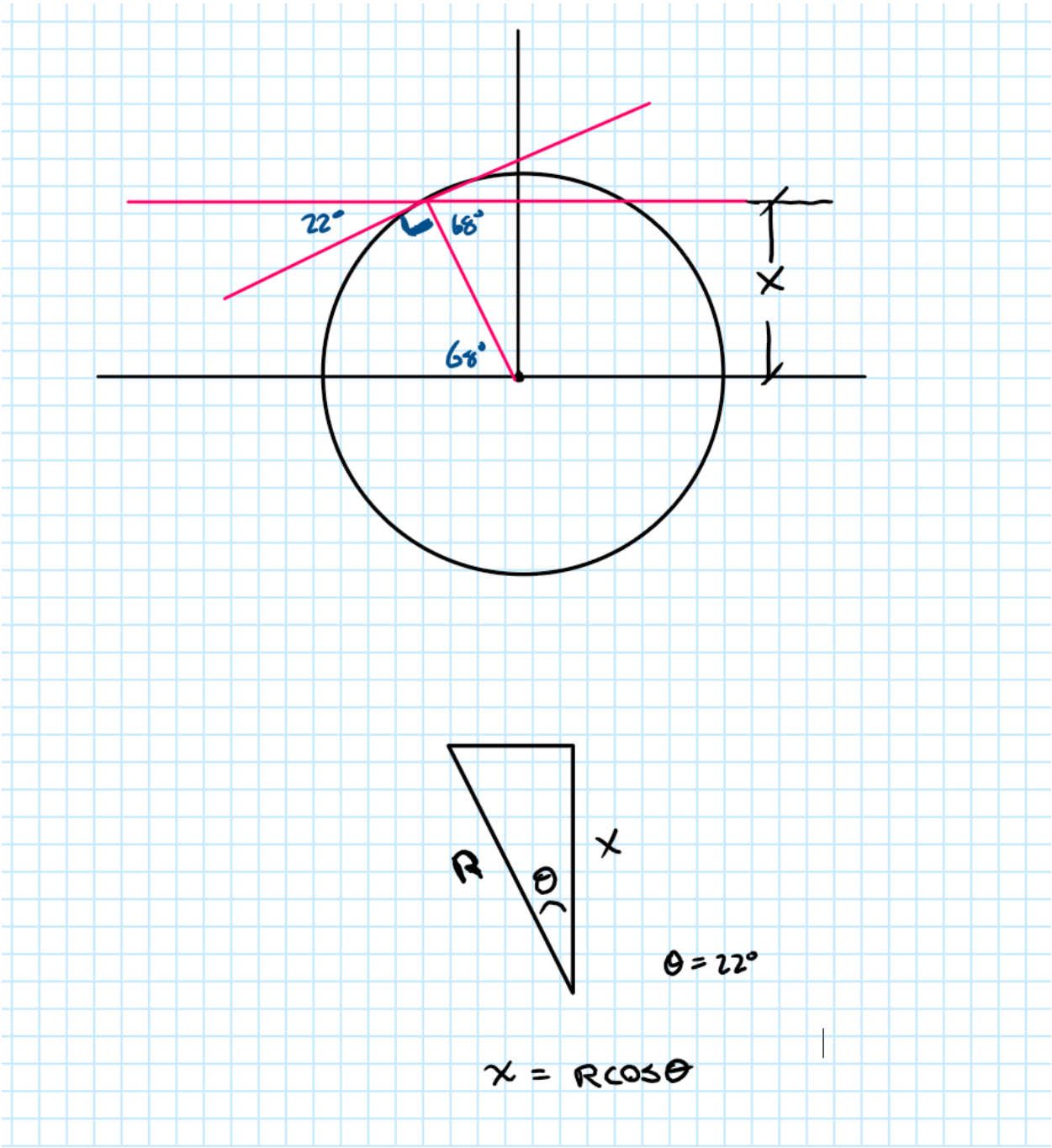


DETAIL A

SCALE 2 : 1

Cal Poly Mechanical Engineering ME 443 - Spring 2025	Units: mm		Title: Pulley Wheel		Drwn. By: Baxter Mercy, Ari Dennis
			Date: 6/2/2025	Scale: 1:1.5	Professor: Frank B. Jumonville

Analysis for Nozzle Position.

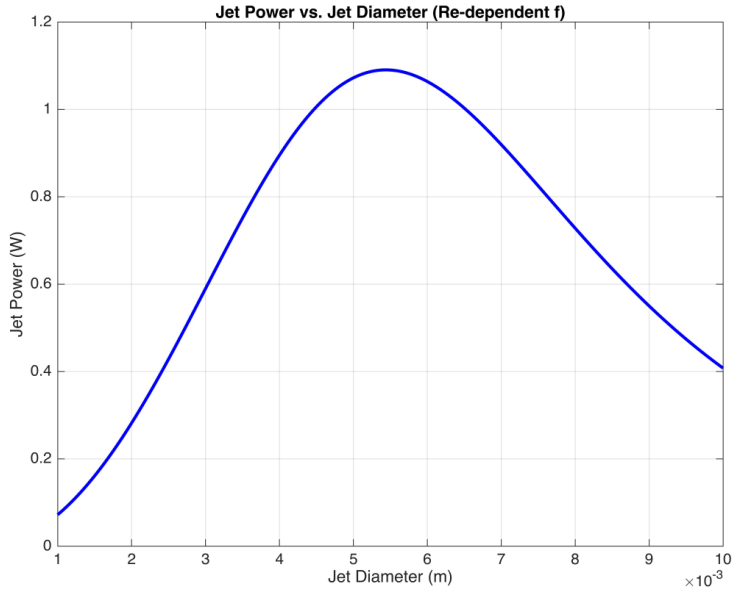


MATLAB Analysis:

```
% Jet Power Optimization and Pelton Wheel Design
clc;
clear;

% ME 443 – Jet Power with friction factor dependent on f (Haaland)
% Baxter Mercy

%% System and Fluid Constants
rho = 1000;       % kg/m^3
mu = 0.001;       % Pa·s
g = 9.81;         % m/s^2
h = 1.65;         % m (height of water column)
D_p = 0.00953;    % m (pipe inner diameter)
L = 1.64;         % m (pipe length)
K = 0.5;          % minor loss coefficient
eps = 0;          % smooth pipe roughness

%% Jet Diameter Optimization
D_j_array = linspace(0.001, 0.01, 400); % Range of jet diameters

% Preallocate arrays
P_array = zeros(size(D_j_array));
Q_array = zeros(size(D_j_array));
V_j_array = zeros(size(D_j_array));
f_array = zeros(size(D_j_array));

for i = 1:length(D_j_array)
    D_j = D_j_array(i);
    A_j = pi * D_j^2 / 4;

    % Initial guess for friction factor and jet velocity
    f = 0.021;
    V_j = sqrt((2 * g * h) / (1 + (D_j^4 / D_p^4) * (f * (L / D_p))));

    for iter = 1:10
        V_p = V_j * (D_j / D_p)^2;  % Pipe velocity
        Re = (rho * V_p * D_p) / mu;

        % Haaland friction factor (custom adjustment)
        if Re > 0
            f = (-1.8 * log10((eps / (3.7 * D_p))^1.11 + (6.9 / Re)))^-2;
        else
            f = 0.021;
        end

        % Update V_j with new f
        loss_term = (D_j^4 / D_p^4) * (f * (L / D_p) + K);
        V_j = sqrt((2 * g * h) / (1 + loss_term));
    end
```

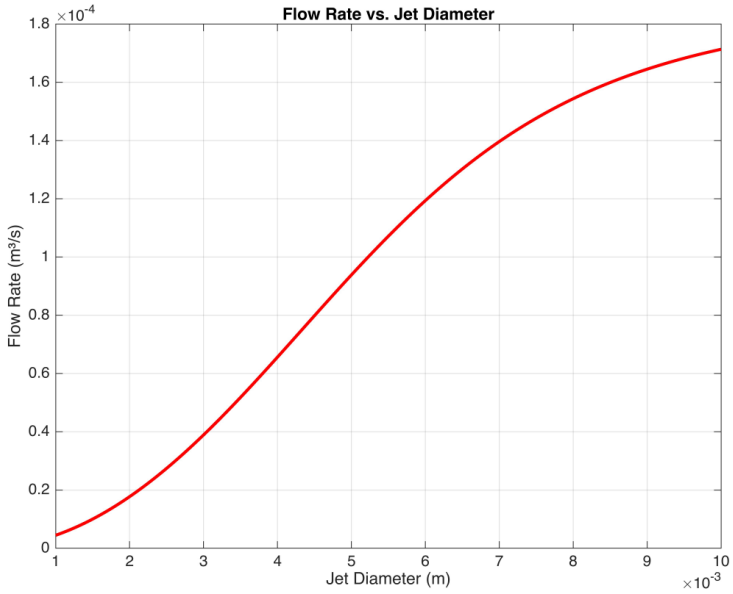
1

```
    % Final power and flow rate calculations
    Q = A_j * V_j;
    P = 0.5 * rho * Q * V_j^2;

    % Store results
    P_array(i) = P;
    Q_array(i) = Q;
    V_j_array(i) = V_j;
    f_array(i) = f;
end

%% Plot Jet Power Results
figure;
plot(D_j_array, P_array, 'b-', 'LineWidth', 2);
xlabel('Jet Diameter (m)');
ylabel('Jet Power (W)');
title('Jet Power vs. Jet Diameter (Re-dependent f)');
grid on;
```

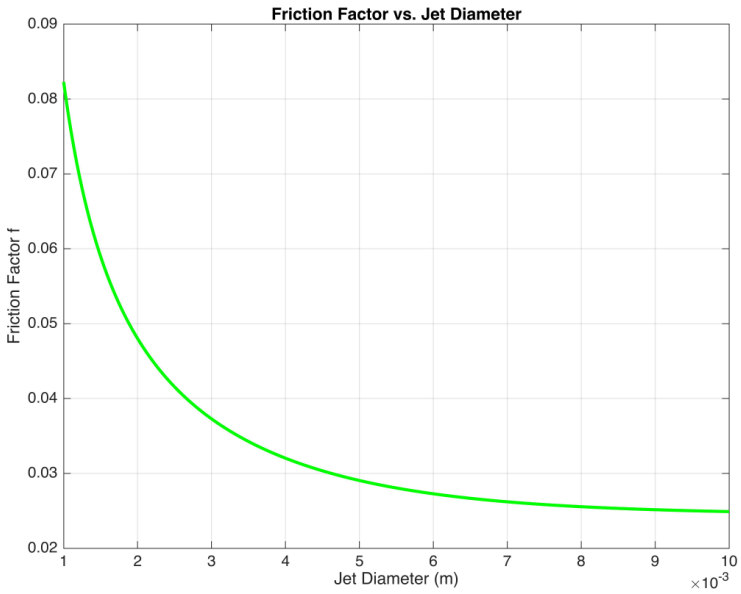



```
figure;
plot(D_j_array, Q_array, 'r-', 'LineWidth', 2);
xlabel('Jet Diameter (m)');
ylabel('Flow Rate (m³/s)');
title('Flow Rate vs. Jet Diameter');
grid on;
```

2




```
figure;
plot(D_j_array, f_array, 'g-', 'LineWidth', 2);
xlabel('Jet Diameter (m)');
ylabel('Friction Factor f');
title('Friction Factor vs. Jet Diameter');
grid on;
```




```
%% Find Optimal Jet Diameter
[P_max, idx] = max(P_array);
D_opt = D_j_array(idx);
Q_opt = Q_array(idx);
V_opt = V_j_array(idx);
f_opt = f_array(idx);

fprintf('\n  Jet Optimization Results:\n');
```

```
  Jet Optimization Results:
```

```
fprintf('Optimal Jet Diameter = %.4f m\n', D_opt);
```

```
Optimal Jet Diameter = 0.0054 m
```

```
fprintf('Max Power = %.2f W\n', P_max);
```

```
Max Power = 1.09 W
```

```
fprintf('Flow Rate at Max Power = %.6f m³/s\n', Q_opt);
```

```
Flow Rate at Max Power = 0.000106 m³/s
```

```
fprintf('Jet Velocity at Max Power = %.2f m/s\n', V_opt);
```

```
Jet Velocity at Max Power = 4.54 m/s
```

```
fprintf('Friction Factor at Max Power = %.4f\n', f_opt);
```

```
Friction Factor at Max Power = 0.0282
```

```
%% Pelton Wheel Sizing Based on Jet Diameter

D_j = D_opt;  % Optimal jet diameter for design
D_j_mm = D_j * 1000;  % mm

% Limits
D_wheel_max = 0.150;  % Maximum allowed wheel diameter [m]
D_wheel_min = 0.080;  % Minimum target pitch diameter [m]
wheel_width_max = 0.050; % Maximum wheel width [m]

% Basic sizing rules
bucket_width_factor = 3;    % Bucket width ≈ 3× jet diameter
bucket_depth_factor = 1.3;  % Bucket depth ≈ 1.3× jet diameter
spacing_factor = .5;
bucket_length_factor = 2.5;
% Spacing factor between buckets

% Bucket dimensions
bucket_width = bucket_width_factor * D_j;
bucket_depth = bucket_depth_factor * D_j;
bucket_length = bucket_length_factor * D_j;

% Wheel pitch diameter
D_pitch_estimate = max(16 * D_j, D_wheel_min);
D_pitch = min(D_pitch_estimate, D_wheel_max); % Final pitch diameter
r_pitch = D_pitch / 2;
C_pitch = pi * D_pitch;

% Buckets
bucket_pitch = spacing_factor * bucket_width;
num_buckets = floor(C_pitch / bucket_pitch);
actual_bucket_pitch = C_pitch / num_buckets;

% Ideal tip speed and RPM
v_tip = 0.46 * V_opt;
omega = v_tip / r_pitch;
RPM = omega * 60 / (2*pi);

% Torque and Mechanical Power
mass_flow_rate = rho * Q_opt;
Force_jet = 2 * mass_flow_rate * V_opt; % Full flow reversal
Torque = Force_jet * r_pitch;
MechanicalPower = Torque * omega;

% Stem length from center to bucket center
stem_length = r_pitch;
```

```
%% Print Pelton Wheel Design Results
fprintf('\n--- Pelton Wheel Design ---\n');
```

```
--- Pelton Wheel Design ---
```

```
fprintf('Jet Diameter = %.2f mm\n', D_j_mm);
```

```
Jet Diameter = 5.44 mm
```

```
fprintf('Wheel Diameter (Pitch Diameter) = %.1f mm (%.2f m)\n',
D_pitch*1000, D_pitch);
```

```
Wheel Diameter (Pitch Diameter) = 87.1 mm (0.09 m)
```

```
fprintf('Bucket Width = %.1f mm\n', bucket_width*1000);
```

```
Bucket Width = 16.3 mm
```

```
fprintf('Bucket Depth = %.1f mm\n', bucket_depth*1000);
```

```
Bucket Depth = 7.1 mm
```

```
fprintf('Bucket length = %.1f mm\n', bucket_length*1000);
```

```
Bucket length = 13.6 mm
```

```
fprintf('Number of Buckets = %d\n', num_buckets);
```

```
Number of Buckets = 33
```

```
fprintf('Actual Bucket Pitch = %.1f mm\n', actual_bucket_pitch*1000);
```

```
Actual Bucket Pitch = 8.3 mm
```

```
fprintf('Jet Velocity = %.2f m/s\n', V_opt);
```

```
Jet Velocity = 4.54 m/s
```

```
fprintf('Ideal Tip Speed = %.2f m/s\n', v_tip);
```

```
Ideal Tip Speed = 2.09 m/s
```

```
fprintf('Ideal Wheel Speed = %.1f RPM\n', RPM);
```

```
Ideal Wheel Speed = 458.2 RPM
```

```
fprintf('Estimated Torque Output = %.3f N·m\n', Torque);
```

```
Estimated Torque Output = 0.042 N·m
```

```
fprintf('Estimated Mechanical Shaft Power = %.2f W\n', MechanicalPower);
```

```
Estimated Mechanical Shaft Power = 2.01 W
```

```
fprintf('Stem Length (from Center to Bucket Center) = %.1f mm\n',
stem_length * 1000);
```

```
Stem Length (from Center to Bucket Center) = 43.5 mm
```

```
%% Pelton Wheel Layout Plot
theta_buckets = linspace(0, 2*pi, num_buckets+1);
theta_buckets(end) = []; % Remove last point (2*pi overlaps)

%% Pulley Diameter Sizing (Based on Sled Friction)

% Given sled parameters
W_sled = 1;           % N (weight of sled)
mu_sled = 0.6;        % friction coefficient
F_required = mu_sled * W_sled;  % Force needed to overcome friction [N]

% Calculate maximum pulley diameter that still delivers enough force
D_pulley_max = (2 * Torque) / F_required;  % [m]
v_belt = RPM * pi * D_pulley_max / 60;     % belt linear speed [m/s]

fprintf('\n--- Pulley Sizing ---\n');
```

```
--- Pulley Sizing ---
```

```
fprintf('Sled Pulling Force Required = %.2f N\n', F_required);
```

```
Sled Pulling Force Required = 0.60 N
```

```
fprintf('Maximum Pulley Diameter = %.1f mm\n', D_pulley_max * 1000);
```

```
Maximum Pulley Diameter = 139.4 mm
```

```
fprintf('Belt Speed at Pulley Rim = %.2f m/s\n', v_belt);
```

```
Belt Speed at Pulley Rim = 3.34 m/s
```