Studying the Effect of Horsepower on Acceleration of a Toyota Camry

William Cusato

December 16, 2024

Contents

1: Description

This program solves the distance traveled by a Toyota Camry (my own car) during the acceleration process. The acceleration process is defined as the time between the start of the study, where no acceleration is present, and the time corresponding to when a velocity threshold is reached. At this time, data collection ends, and analysis ensues. The acceleration of the car is found through Newton's second law of motion, where the forces acting on the car are from the engine, drag, and friction between the wheels and the road. The force from the engine is found from the car's horsepower, and a range of horsepowers are tested for data analysis. The program plots the functions of velocity vs. time, and distance traveled vs. horsepower of the engine.

2: Inputs

2.1 Manually Entered Parameters

These parameters are provided by the user of the program. The program will ask the user for these parameters before the analysis begins.

- Maximum velocity to be used as the threshold to end data collection.
- The maximum horsepower that the user wants to analyze.

2.2 Force Balance Parameters

- Mass of the car (2007 Toyota Camry)
- Density of air at 25°C
- Frontal diameter and area of the car
- Drag coefficient in turbulent flow
- Acceleration due to gravity
- Rolling resistance between the tires and the road
- Initial velocity of the car (to avoid division by zero error)

3: Outputs

The program performs the following during operation:

- Sets up a Newton's second law of motion force balance and solves for the acceleration as a function of the Camry's velocity.
- Calculates the velocity using the expression of acceleration derived from Newton's second law of motion and the 4th-order Runge-Kutta ODE solver. The program then outputs a plot of velocity vs. time.
- Uses Simpson's $\frac{3}{8}$ rule to calculate the distance traveled by the car during acceleration.
- The previous two steps are repeated four times with varying levels of engine power. The results are then compiled into a graph that displays the distance traveled during acceleration compared to the horsepower of the engine. The graph includes a spline curve of best fit.

4: Numerical Methods

- ODEs - Runge-Kutta fourth-order ODE solver
- Numerical Integration - Simpson's $\frac{3}{8}$ rule
- Curve Fitting - Spline curve fitting

5: Validation of Results

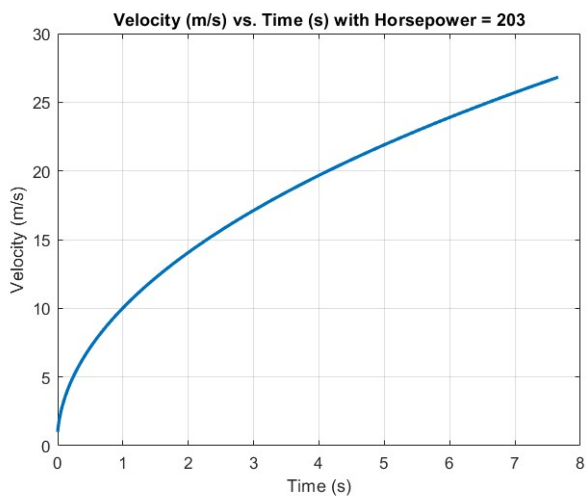
There is not readily available data for the distance traveled during the acceleration of a car, but there exists data closely tied to the problem being solved by this program. In particular, the time it takes a car to travel from 0 to 60 mph is perhaps the most used method of quantifying the engine performance of a car, and while solving the acceleration ODE, the program keeps track of the velocity of the car and the current time that has passed since the beginning of the study.

To draw the easiest comparison between the problem being solved by the program and the experiments conducted in reality, the time it takes for a Toyota Camry to accelerate to 60 mph was found online. The time it takes for a car to accelerate is heavily dependent on certain aspects of the car, such as the mass. Therefore, being consistent with the car used between both experiments is paramount. It should also be noted that there are many different models of a Toyota Camry; therefore, the LE model of the Camry was used for comparison. The experimental data for a Camry's acceleration can be found below:

LE	2.5-L 4-Cylinder	203 hp	135 mph (estimated)	7.6 seconds (estimated)

To best mirror the conditions tested by Toyota, the program was run with a threshold velocity of 26.82 m/s, which is equivalent to 60 mph, and a horsepower of 203. The results of the program simulation can be seen below:

From the graph to the right, it can be seen that the time corresponding to a velocity of 26.82 m/s (or 60 mph) is between seven and eight seconds. In the console output of the program (which is too large to put here), it can be seen that this time is 7.66 seconds, which matches the data provided by Toyota (shockingly) well.



It should be noted that if the program is to be used for other cars, all parameters at the beginning of the script must be changed to those of the car that is to be tested. The script is highly dependent on all parameters, and these parameters are different for every car (i.e., the frontal area of the car).

6: Thermal Fluid and Mathematical Details

6.1 Mathematical and Physics Details

Newton's second law states that the net force acting on an object is equal to the object's mass multiplied by its acceleration. In the case of the Toyota Camry, there are three forces being experienced: the force provided by the engine, the force associated with drag, and the force created by friction between the tires and the road. Both drag and friction oppose motion, while the engine force creates motion. If the positive x-direction is pointing in the direction of movement, then the net force in the x-direction can be written as:

$$F_{net} = m \cdot a = F_{engine} - F_{drag} - F_{friction}$$

The force coming from the engine is a result of the power that the engine creates (expressed in horsepower). Power is related to force by a factor of velocity, therefore, we can find the force of the engine as follows:

$$P = F \cdot v \Longrightarrow F = \frac{P}{v}$$

Drag force is the accumulation of the force in the x-direction that results from an object impeding fluid flow, or an object moving through a fluid flow. It is commonly represented as:

$$F_{drag} = \frac{1}{2} \cdot C_D \cdot \rho \cdot v^2 \cdot A$$

Where ρ is the density of the fluid, v is the velocity of the fluid (in our case, the car), and A is the frontal area of the object. More information on the drag coefficient C_D and how the frontal area is calculated can be found in subsection 6.3 Thermal Fluid Details.

Finally, the force resulting from friction is calculated as a friction factor multiplied by the normal force acting on the object. In the case of the car, because there are no external y-direction forces acting on the car, the normal force is equal to the weight of the car, so the friction force can be written as:

$$F_{friction} = \mu \cdot m \cdot g$$

Where μ is the friction coefficient. Because the wheels are rolling, μ will be somewhat insignificant. In the case of stock tires, $\mu = .02$.

One final note is the relationship between acceleration, velocity, and time. It can be seen that:

$$a = \frac{dv}{dt} = \frac{d^2x}{dt^2}$$

Therefore, it is valid to take the integral of velocity to obtain distance traveled. Likewise, it is valid to solve an ODE of acceleration to obtain a function for velocity.

6.2 Car Assumptions

The main assumption used in the program revolves around the change of horsepower as the car accelerates. Horsepower is typically shown as a function of the RPM of the engine, which can be seen below:

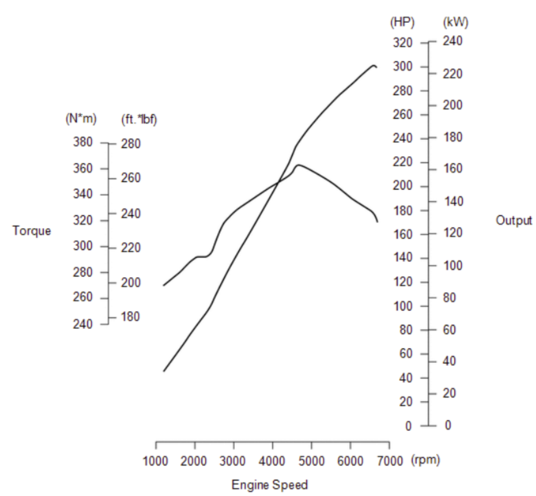
$$P = \tau \cdot \text{RPM}$$

τ is the torque created by the engine, which is dependent on the gear ratio among other factors. It is also important to note that several conversion factors were left out of the above equation, as they are dependent on the unit system (which confused me during this project).

Most importantly, the RPM of the motor is linearly related to the speed at which the car moves. Therefore, the power output by the engine is (roughly) proportional to the speed of the car.

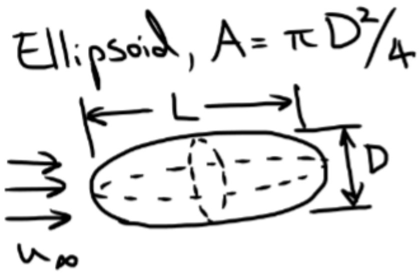
$$P \propto v$$

Therefore, using this relation, we can make a key assumption (that may not always hold valid), and that is that the average power that the car outputs is equal to half of its maximum power output.



6.3 Thermal Fluid Details

The drag coefficient is an experimentally defined value and is also highly dependent on the geometry of the object being exposed to flow. Therefore, the drag coefficient is often assumed using simplified geometry. In the case of this project, the drag coefficient for an ellipsoid was used.



L/D	C_D laminar	C_D turbulent
0.75	0.5	0.2
1	0.5	0.2
2	0.3	0.1
4	0.3	0.1

Figure 1: Table showcasing the drag coefficient for an ellipsoid with varying dimensions and different flow regimes. (From Prof. Kemmerling's Lecture Slides)

For the Camry, the length of the car was used for L (4.92 m), and the average value of the height and width of the car was used for d, ($\frac{1.44+1.83}{2} = 1.525$ m), which yields a ratio of $\frac{4.92}{1.525} = 3.22$.

Finally, it was assumed that the flow would be turbulent due to the highly chaotic nature of a car driving through air. This yields a drag coefficient of 0.1.

7: Sample Output

The input into this code was:

- Max Velocity: 50 m/s (111.85 mph)
- Minimum Power: 100 HP

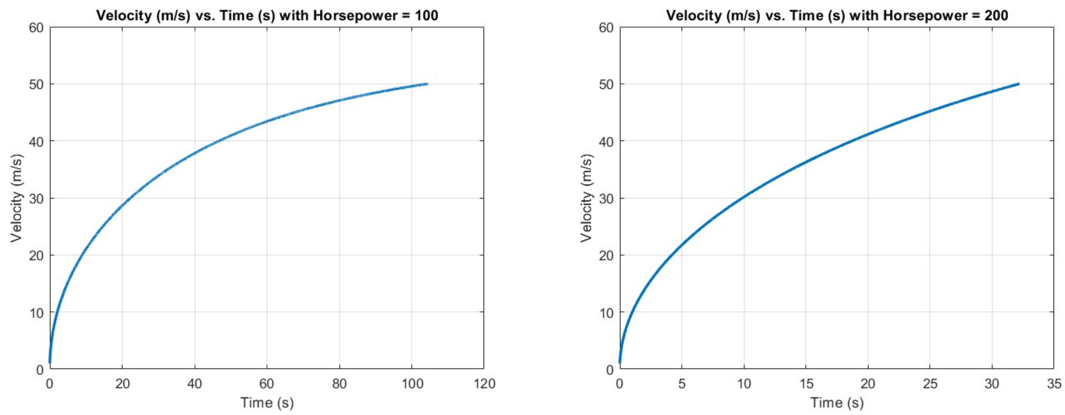


Figure 2: Graphs for Velocity vs. Time for HP = 100 and 200 respectively

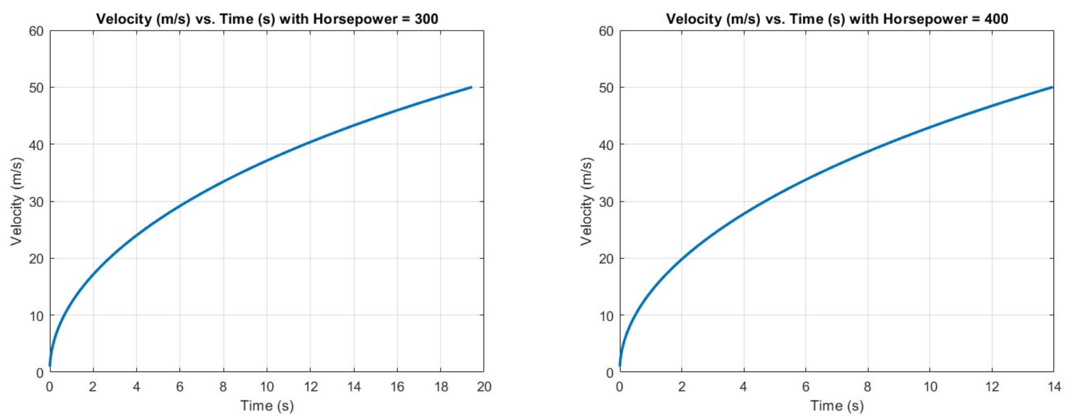


Figure 3: Graphs for Velocity vs. Time for HP = 300 and 400 respectively

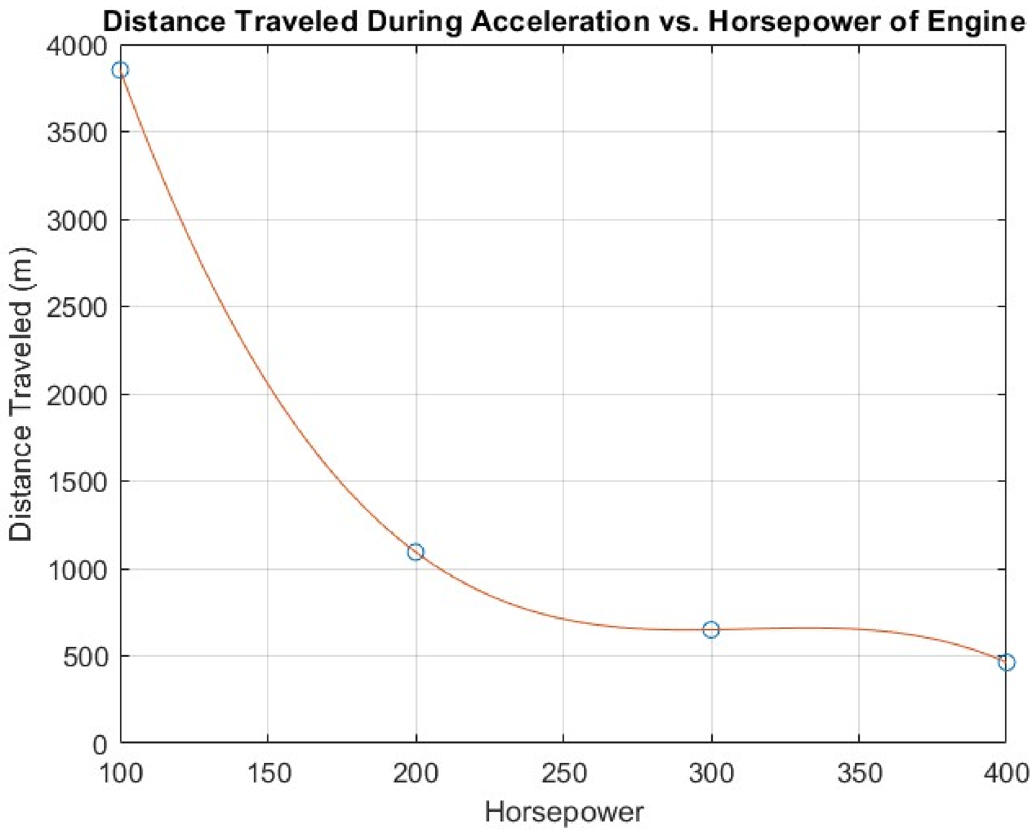


Figure 4: Graph for Distance traveled during acceleration vs. HP

8: Code

```
clc; clear; clear all;

%% Variable Decleration
m = 1480; % Mass of 2007 Toyota Camry in kg
rho = 1.184; % Desnity of Air at 25 degrees celsius
d = 1.65; % Estimated frontal diamter of 2007 Toyota Camry
A = pi/4*d^2; % Frontal surface area of 2007 Toyota Camry
Cd = .1; % Drag Coefficient for Elipsoid with L/d = 3 in turbulent flow
g = 9.81; % Value for gravity on earth
mu = .02; % Value of rolling resistance between tire and dry asphalt
v = 1; % Initial velocity of the car

%% Matlab Functions
%{
  4th Order Runge-Kutta Method for Solving Linear ODE
  Takes Two functions, their boundary conditions, a step size and a
  number of iterations and solves a system of ODEs
%}
function [rk1, steps] = RKMethodProj(df1, maxV, h, start1)
    % Array Decleration so results can be printed in tabular form
    steps = [];
    rk1 = [];
    steps(1) = 0;

    % Boundary Conditions
    rk1(1) = start1;
    i = 1;

    % Iterate and Solve ODEs
    while rk1(i) < maxV
        % Initialize values for current iteration
        t = steps(i);
        x1 = rk1(i);

        % Solve for all values of k
        k1_1 = df1(t, x1);
        k2_1 = df1(t + h/2, x1 + h/2 * k1_1);
        k3_1 = df1(t + h/2, x1 + h/2 * k2_1);
        k4_1 = df1(t + h, x1 + h * k3_1);

        % Solve linear set of ODE and increment
        rk1(i + 1) = (x1 + h/6 * (k1_1 + 2*k2_1 + 2*k3_1 + k4_1));
        steps(i + 1) = t + h;
        i = i+1;
    end

    % Print in tabular form
    T = table(steps.', rk1.', 'VariableNames', {'Time', 'Velocity'});
    disp(T)

end


%{
  Simpson's 3/8 method take a function, the lower bound, the upper bound
  and outputs a numerically derived derivative of the function
```

```
%}
function I = simpsons38(f, a, b)
% Create step size, generate x-values, and solve for x values
a1 = a*100+1; % Change a to be useable for vectors
b1 = b*100+1; % Change b to be useable for vectors
h = (b - a) / 3;
x = [floor(a1), floor(a1 + (h*100)), floor(a1 + 2*(h*100)), floor(b1)];

% Simpson's 3/8 Rule
I = (3*h/8) * (f(x(1)) + 3*f(x(2)) + 3*f(x(3)) + f(x(4)));
end


%% Main Driver Code

% Ask the user for the desired maximum speed
maxSpeed = input('Enter your desired max speed of the car (m/s): ');

% Make sure the input makes sense
if maxSpeed <= 0
    disp('The speed must be a positive number.\n\n');
elseif maxSpeed > 150
    disp('This is a 2007 Toyota Camry, not a Lamborghini!')
else

    % Prompt user for desired maximum velcoity and horsepower range
    fprintf('The car will be modeled to reach a maximum speed of %.2f m/s.\n\n', maxSpeed);
    hp = input('What is the minimum horsepower would you like the car to study: ');
    if hp <= maxSpeed
        disp('The car may not make it to the max speed. Try a higher horse power')
    else

        % Create range of horse-powers to study
        incriment = hp;
        hpVector = hp:incriment:4*hp;

        % Create arrays and cellular arrays for use in plots
        distanceVector = [0;0;0;0];
        velos = {[],[],[],[]};
        times = {[],[],[],[]};

        % Iterate four times to create distance data
        for i = 1:4
            % Function for acceleration and velocity
            dv = @(t, v) ((hpVector(i)*745.7)/(2*v) - 0.5*Cd*rho*A*v^2 - mu*m*g)/m;
            disp('Horespower = ', hpVector(i))
            [velos{i}, times{i}] = RKMethodProj(dv, maxSpeed, .01, 1);

            % Make plot of Acceleration with time
            figure();
            plot(times{i}, velos{i}, 'LineWidth', 2);
            grid on
            xlabel('Time (s)')
            ylabel('Velocity (m/s)')
            title(sprintf('Velocity (m/s) vs. Time (s) with Horsepower = %d', floor(hpVector(i)))

            % Determine distance traveled using numeric integration
            distanceVector(i) = simpsons38(velos{i}, 0, times{i}(end));
        end
```

```
        % Make best fit curve (cubic spline) and plot it
        x = linspace(min(hpVector), max(hpVector), 100);
        y = spline(hpVector, distanceVector, x);
        figure();
        plot(hpVector, distanceVector, 'o', x, y, '-');
        title('Cubic Spline Fit');
        grid on
        xlabel('Horsepower')
        ylabel('Distance Traveled (m)')
        title('Distance Traveled During Acceleration vs. Horsepower of Engine')
    end
end
```

9: Bibliography

9.1 Toyota data:

- https://www.toyotanation.com/threads/torque-curves-for-new-engines-vs-previous-gen.1623010/
- https://www.bobhowardtoyota.com/2021-toyota-camry-0-60/
- https://www.greentoyota.com/research/new-toyota-camry-weight.htm

9.2 Other Parameters:

- https://www.engineeringtoolbox.com/rolling-friction-resistance-d_1303.html
- https://www.engineeringtoolbox.com/air-density-specific-weight-d_600.html