Water Bottle Rocket Performance and Prediction

Colin Harman, Brian Freeburg, Joe Hendrickson
ME495 W14 Section 3, Team 4
4/21/2015

FOREWORD

Rockets made from water bottles and propelled by water and compressed air demonstrate fundamental fluids principles with low cost and minimal safety hazard. We used these water bottle rockets as a mechanism to test and develop our design skills and our ability to apply mechanical engineering theory to practical situations. You have requested that we design and build a water bottle rocket and develop a mathematical model of its performance for use in a class competition for accuracy. You have also asked us to document our methods for design and modeling. Finally, to spread the knowledge and enthusiasm that we gained and ensure that more individuals may one day stand in our place, you requested that we create outreach materials targeted at K-12 students. We have completed these tasks. The purpose of this report is to provide our rocket design, mathematical model, methods, and outreach materials.

SUMMARY

We have created a physical water bottle rocket and developed a model to predict its behavior. After we determined rocket characteristics to plug into the model we used it to win the first round of a class competition to hit a target with our physical rocket. Our physical rocket performed well due primarily due to its stability, and the model performed well due to appropriate assumptions and accurate rocket characteristics. Through experimentation we determined that our model can predict the distance traveled by a rocket launch within 8.5±2.0%. Throughout the process we used a number of advanced analyses and techniques including pendulum inertia analysis and MATLAB ODE solving, and we gained insight into rocket design. We created a lesson plan for middle school students to spread our knowledge to a younger generation.

METHODS

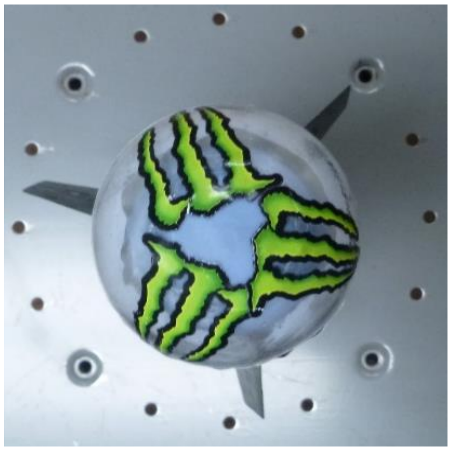
To produce the highest quality rocket and mathematical model we carefully designed and built the rocket and then empirically determined its physical characteristics, which would later be used in the mathematical model. The model was developed from a set of governing equations, and then validated using experimental data. The model was then tuned and reduced to yield the simplest, most effective model possible.

Creation of the Rocket

We had limited knowledge of rocket design but still set several goals for our design. We wanted our rocket to be stable and its motion to be simple and predictable to facilitate making an effective model. To ensure stability we chose a long, narrow bottle, added weight at the very front, and added fins at the back. This helps keep the tail of the rocket behind the nose. We used three fins at an angle of 9.6 degrees each, relative to the rocket's longitudinal axis, shown in Figure 1, to induce a reasonable amount of stabilizing spin. We formed our nose cone out of a bottle neck and a half ping pong ball, with a parabolic shape in mind. The full rocket assembly is shown in Figure 2 on page 2.

Figure 1, Right: Head-on view of rocket shows angle of fins

Figure 2, Below: Isometric view of rocket shows overall shape and fin positioning

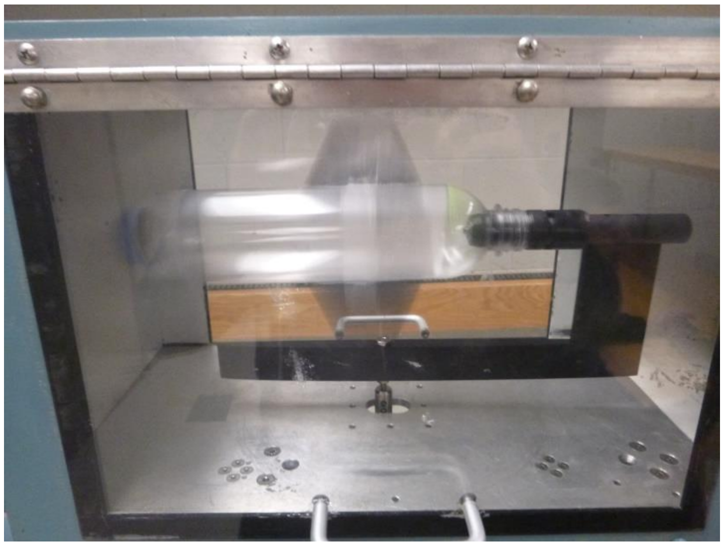


Rocket Characterization
For later use in the model we needed a number of physical characteristics of the rocket. Many of these, such as mass or dimensions, were trivial to obtain. This section discusses how the non-trivial characteristics coefficient of drag and moment of inertia were experimentally found.

Coefficients of Drag and Lift: Lift and drag coefficients are critically important to predicting the rocket's projectile motion phase. Before conducting our experiment we first conducted pre-test measurements and setup and calibration procedures, which are elaborated upon in Appendix A. Next, we installed the fixture that will be used to mount the water bottle rocket and orientated it parallel to the wind tunnel. We mounted the water bottle as shown in Figure 3, page 4. We then took data on the lift and drag forces at three different speeds with the rocket rotating and not rotating. Using the calibration curves and the lift and drag forces the coefficients of lift and drag were calculated to be $C_{D,static}$ = 0.25 and $C_{L,static}$ = -0.1 when static (not spinning), and $C_{D,ss}$ = 0.22 and $C_{L,ss}$ = 0.02 when spinning at steady state. These coefficients are related to Reynolds number in Appendix A.

A problem that arises from using a fixture for wind tunnel testing is that the drag force created by the fixture itself is not factored into the data output from LabView. In order to eliminate the force of drag due to the sting, we took drag force data for only the sting at each speed and subtracted these drag values from the corresponding data for the spinning and non-spinning rocket tests. Similarly, the sting also creates a lift force in the wind tunnel. This comes from the fixture being imprecisely aligned in the wind tunnel. Although the drag and lift forces are not additive, subtracting the forces due to the fixture brings the value nearer to reality than taking no action.

Figure 3: The rocket was mounted on a fixture inside the wind tunnel that allowed it to spin, as shown, or be locked in place.

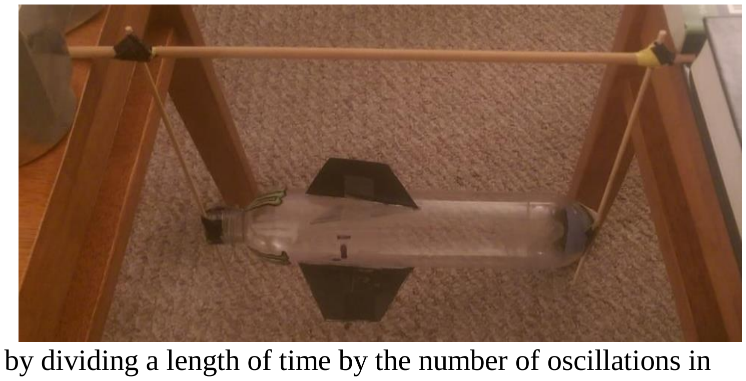


Moment of Inertia: The moment of inertia about the rocket's longitudinal axis was needed for inclusion in the model because the rocket was designed to spin during flight, and how quickly it "spins up" after launch depends partly on the moment of inertia. We devised a dynamic experiment to determine the moment of inertia, rather than calculating it from static measurements of the rocket, since the mass distribution in the calculation would have to be estimated.
We created a fixture that allowed the rocket to oscillate about an axis parallel to the axis of interest, shown in Figure 4, since the moment of inertia of a pendulum can be determined from its frequency of oscillation by solving the small-angle torque balance shown in Equation 1, yielding Equation 2.

$$I\ddot{\theta} = lmg\theta \qquad \text{Eq. 1} \qquad\qquad \omega_n = \sqrt{\frac{lmg}{I}} \qquad \text{Eq. 2}$$

Where I is the moment of inertia about the axis of rotation, θ is the angle with vertical of the pendulum, l is the length from the axis of rotation to the center of gravity of the pendulum, g is the acceleration due to gravity, and ω_n is the natural frequency of oscillation. Air drag is assumed to be negligible due to the low speed of the apparatus during the experiment.

Figure 4: The rocket was suspended from wooden dowels, forming a pivot on the top dowel. The assembly was perturbed and the assembly's moment of inertia was calculated from the resulting period of oscillation.



We slightly perturbed the assembly, allowing it to oscillate about the pivot dowel. The period of oscillation was found by dividing a length of time by the number of oscillations in that time period. Natural frequency was found by dividing two pi by the period. The combined moment of inertia about the pivot was solved for using Equation 3. This process was repeated for the dowel fixture with the rocket removed. The fixture's moment of inertia was subtracted from the entire assembly's, yielding the rocket's moment of inertia about the pivot dowel. The parallel axis theorem, shown in Equation 3, was used to determine the rocket's moment of inertia about its longitudinal axis.

$$I_C = I_P - mr^2 \quad \text{Eq. 3}$$

Where I_C is the moment of inertia around the center of the body, I_P is the moment of inertia about a parallel axis, m is the mass of the body, and r is the distance between the center of the body and the parallel axis. We found the moment of inertia about the longitudinal axis of our rocket to be $2.7\text{x}10^{-5} \pm 1.0\text{x}10^{-5}$ kg m^2, with the error resulting from variation between multiple period measurements.

Model Creation

To predict the rocket's behavior we developed a set of equations that described the physical phenomena present. We organized these equations as a combination of first-order differential equations and simple algebraic equations and solved them using MATLAB's ODE45. These equations fell into the categories of conservation of momentum, external forces, gas expansion, and nozzle velocity. We first executed this process in a single dimension and then converted it to three dimensions. Finally, we implemented certain conditions that occur during launch. The model was constructed as a continuous whole, rather than as separate functions for the launch and projectile motion.

Defining Variables: All variables used in this section are defined in Appendix B.

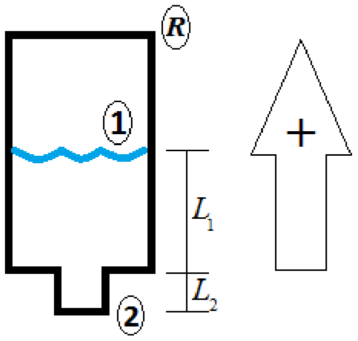


Figure 5: The bottle control volume. Point 1 is located at the top of the water inside the bottle. Point 2is located at the bottom of the water at the bottle nozzle. The rocket, R, is the control volume containing points 1 and 2. Air occupies the volume above point 1. Positive velocities are defined as moving upward.

Conserving Momentum: We used Newton's second law, shown in Equation 4, as the starting point for describing the rocket's motion, with no external forces. This means that at any point in time, the sum of the changes in momentum of each part of the system must equal zero, shown in Equation 5.

$$\frac{dp_{sys}}{dt} = 0 \quad \text{Eq. 4} \qquad \frac{dp_i}{dt} + \frac{dp_{i+1}}{dt} + \cdots + \frac{dp_n}{dt} = 0 \quad \text{Eq. 5}$$

We applied this equation to our rocket by lumping the momentum into four parts: the momentum of water inside the nozzle (at point 2), the momentum of water inside the rocket (at point 1), the momentum of the rocket structure, and the momentum of the water exiting the rocket (past point 2). The momentum of the air inside the rocket is negligible so we ignored it. Together, these four lumps account for the momentum of the entire system. The result is shown in Equation 6. Note that the fluid velocities must take into account the velocity of the rocket as the velocity of the control volume, shown in Equation 7.

$$\frac{d(m_1U_1)}{dt} + \frac{d(m_2U_2)}{dt} + \frac{d(m_Rv_R)}{dt} + U_2\frac{dm_{CV}}{dt} = 0 \quad \text{Eq. 6} \qquad U_i = v_R + v_i \quad \text{Eq. 7}$$

Next we plugged in Eq. 7 and evaluated the derivatives in Equation 6 to get Equation 8. We took m_2 to be constant since the length of the nozzle is unchanging and water was assumed to be incompressible and took m_R to be constant since the rocket structure's mass does not change. The chain rule was used in evaluating the term at point 1 since m_1 is not constant.

$$m_1(\dot{v}_R + \dot{v}_1) + \dot{m}_1(v_R + v_1) + m_2(\dot{v}_R + \dot{v}_2) + m_R\dot{v}_R + \dot{m}_{CV}(v_R + v_2) = 0 \quad \text{Eq. 8}$$

The mass flow out of the control volume is described by Equation 9.

$$\dot{m}_{CV} = \rho_{H20} A_2 v_2 \qquad \text{Eq. 9}$$

Relating Flow Velocities: We assumed a unidirectional, rectilinear flow, through sections of constant area, which allowed us to use the equation of constant flow rate, Equation 10, and its derivative, Equation 11.

$$A_1 v_1 = A_2 v_2 \quad \text{Eq. 10} \qquad\qquad A_1 \frac{dv_1}{dt} = A_2 \frac{dv_2}{dt} \qquad \text{Eq. 11}$$

Adding External Forces: When external forces are applied to a system they contribute to a change in momentum. This can be expressed through a modification of Newton's second law shown in Equation 12.

$$\frac{dp_{sys}}{dt} + F_{ext} = 0 \qquad \text{Eq. 12}$$

This means that any external forces can be directly added on to the left hand side of Equation 8. The dominating external forces that act on a water bottle rocket are thrust, gravity, and air drag. For the one-dimensional solution we assumed that the rocket would travel vertically, parallel to the force of gravity. The gravitational force and air drag force are shown in Equations 13 and 14, respectively. The force of lift was assumed to be zero for the entire problem.

$$F_g = -(m_R + m_1 + m_2)g \qquad \text{Eq. 13} \qquad\qquad F_D = -\frac{1}{2} C_D \rho_{air} A_R v_R |v_R| \qquad \text{Eq. 14}$$

The gravitational force is negative since gravity points down. The drag force is negative because it points opposite v_R, and the absolute value sign rather than a simple squared term allows the sign to change with the sign of v_R. The modified momentum equation is shown in Equation 15.

$$m_1(\dot{v}_R + \dot{v}_1) + \dot{m}_1(v_R + v_1) + m_2(\dot{v}_R + \dot{v}_2) + m_R \dot{v}_R + \dot{m}_{CV}(v_R + v_2) + F_D + F_G = 0 \qquad \text{Eq. 15}$$

Describing Gas Expansion: As water leaves the rocket, the volume occupied by the air above point 1 increases. We assumed that an adiabatic process occurs during expansion, yielding Equation 16.

$$PV^{\gamma} = constant \qquad \text{Eq. 16}$$

Describing Flow Velocity: With flow velocities related, pressure behavior described, and rocket behavior described in terms of flow velocities, we still had to describe the flow velocities. We did this by starting with the unsteady Bernoulli equation, shown in Equation 17 [1]. It describes the flow between two points in a control volume with incompressible, frictionless flow along a streamline.

$$\frac{U_1^2 - U_2^2}{2} + \frac{(P_1 - P_2)}{\rho} + (gz_1 - gz_2) - \int_1^2 \frac{dU}{dt} ds = 0 \qquad \text{Eq. 17}$$

Points 1 and 2 are applied to the water in a water bottle control volume in Figure 5 on page 4. Because pressure only appears as a difference and P_2 represents atmospheric pressure, gauge pressure P was used for $P_1 - P_2$. The gravity term was ignored since the maximum height difference in height is less than 0.1 m and during launch acceleration is far greater than 9.8 m/s^2 (it was observed to be on the order of 500 m/s^2). We evaluated the integral term, assuming that the fluid flows through two discrete sections of constant area, and combined the result with Equation 17 to get Equation 18

.

$$\frac{U_1^2-U_2^2}{2}+\frac{(P_1-P_2)}{\rho}+(\frac{dv_1}{dt}L_1+\frac{dv_2}{dt}L_2)=0 \qquad \text{Eq. 18}$$

Converting to Three Dimensions: Once the basic, one-dimensional model was completed we converted it to a simplified three-dimensional model by vectorizing it while making several assumptions. The axes convention is that *z* is positive height, *x* is positive horizontal distance in the same direction as launch, and *y* is *x* cross *z*. We treated the rocket's behavior in the *y*-direction as independent of its behavior in the *x*- and *z*-direction. We assumed that the rocket always points into the oncoming air in the *x*-*z* plane, shown in Equation 19. This is reasonable based on our observations because the spin of the rocket prevents it from rotating observably about the *z* axis, and the rearward location of the fins ensures that they are kept behind the nose of the rocket by the wind.

$$\theta=\arctan\left(\frac{v_{asz}}{v_{asx}}\right) \qquad \text{Eq. 19}$$

We used this angle to convert the momentum equation, Equation 8, from one dimension to two by splitting it into *x*- and *z*- parts, shown in Equations 20 and 21.

Eq. 21, 22:

$$\cos(\theta)[m_1(\dot{v}_R+\dot{v}_1)+\dot{m}_1(v_R+v_1)+m_2(\dot{v}_R+\dot{v}_2)+m_R\dot{v}_R+\dot{m}_{CV}(v_R+v_2)+F_D]=0$$
$$\sin(\theta)\,[m_1(\dot{v}_R+\dot{v}_1)+\dot{m}_1(v_R+v_1)+m_2(\dot{v}_R+\dot{v}_2)+m_R\dot{v}_R+\dot{m}_{CV}(v_R+v_2)+F_D]+F_G=0$$

We described the rocket's motion *y*-direction using Newton's second law with air drag acting as an external force, shown in Equation 22. C_{Dy} was defined as 1.17 [2].

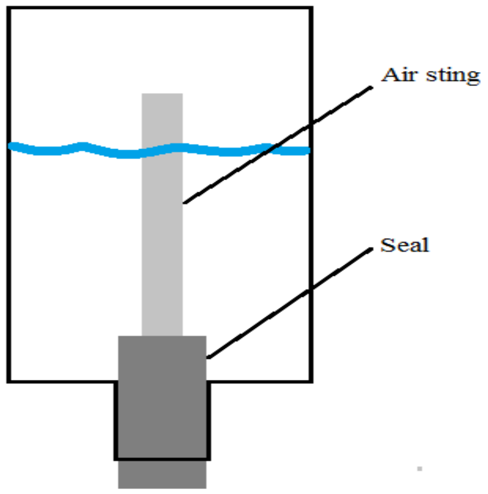
$$F_{Dy}=m_{tot}\frac{dv_{asy}}{dt} \qquad \text{Eq. 22}$$

Adding Launch Conditions and Advanced Factors: After we created a working three-dimensional model we began to include special conditions to bring the model's predictions closer to reality. We first turned our attention to the launch environment. Figure 6 gives an overview of how the rocket interacts with the launcher prior to takeoff. It is immediately apparent that, as long as the nozzle is over the seal, no water can escape the bottle. Therefore, the momentum of the water leaving the bottle is zero and the system can be described by *F=ma*, where the force comes from the pressure on the seal. The single-dimensional version of this relation is shown in Equation 23.

$$PA_2=m_{tot}\frac{dv_R}{dt} \qquad \text{Eq. 23}$$

We applied this condition to the model while the distance traveled was less than the length of the seal. Along with the altered motion equation, we stated that the flow velocity inside the rocket was zero. We also stipulated that while the rocket was on the seal its angle θ could not change since the seal constrains the angle of the rocket.

Figure 6: Prior to launch the bottle nozzle area is completely occupied by the seal. As the rocket accelerates it leaves the seal and moves along the air sting, which occupies part of the nozzle area.



After leaving the seal instructed the model to return to the previously stated equations. However, we added another condition for the period after leaving the seal but before leaving the air sting. Because the sting occupies part of the nozzle area, we stated that the effective nozzle area equals the original nozzle area minus the area of the sting for this period.

With the launch physics accounted for we set out to include wind in the model, modeled as having a constant velocity. We assumed that the wind speed in the *z* direction was zero and implemented wind speed in the *x* and *y* directions by defining a new variable, airspeed: $v_{as} = v - v_{wind}$ where *v* is the absolute velocity. We used airspeed in the drag equations in *x* and *y* and the rocket angle θ equation.

Next we integrated the spin of the rocket into the model. This is important because as the rocket "spins up" its coefficient of drag changes from $C_{D,\ static}$ to $C_{D,\ ss}$, which affects the drag force. To describe how the rocket spins up we used the torque equation, Equation 24, and found the steady state rotational speed of the rocket, Equation 25.

$$I\frac{d\omega}{dt} = n_{fins} r_R F_L \qquad \text{Eq. 24} \qquad\qquad \omega_{ss} = \frac{v_R \tan(\theta_{fin})}{r_R} \qquad \text{Eq. 25}$$

This assumes that the lift force of each fin acts at its base, which is not completely accurate. Because we had two drag coefficients and knew the angular velocities associated with each, we decided to linearly interpolate in between to simplify the problem. We created Equation 26 to do this.

$$C_D = C_{D,static} - \frac{\omega}{\omega_{ss}}(C_{D,static} - C_{D,ss}) \qquad \text{Eq. 26}$$

We stipulated that the fins did not begin to rotate the bottle until the water was gone in an effort to avoid the complexity of evaluating the moment of inertia with water partially included due to viscosity.

We instructed the simulation to stop when the rocket reaches a height of -0.5 m to account for the starting height of the launcher. We implemented head losses, which are included in the working model and discussed in Appendix C.

Model Summary
We combined and arranged the basic equations in motion described above into the Euler method-/MATLAB ODE-friendly first-order differential equations and algebraic equations below. The launch conditions and special factors were included in the model but are not listed here. A sample simulation is show in Figure 7 on page 8, and the code for the model is found in Appendix E.

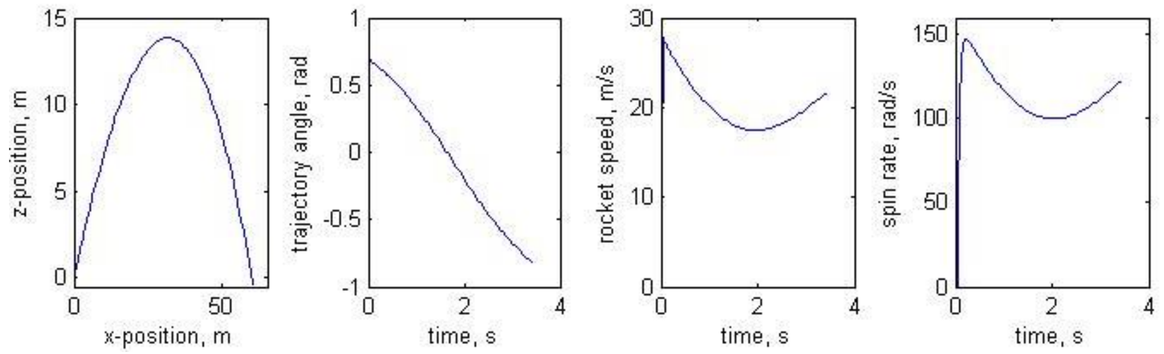
$P = P_0 \left(\frac{V_0}{V}\right)^{1.4}$ Eq. 27 $v_2 = \frac{A_1}{A_2} v_1$ Eq. 28 $\frac{dv_2}{dt} = \frac{A_1}{A_2} \frac{dv_1}{dt}$ Eq. 29

$\frac{dv_1}{dt} = -\frac{\left[P+\frac{1}{2}\rho_{H2O} v_1^2\left(1-\left(\frac{A_1}{A_2}\right)^2\right)\right]}{\rho_{H2O} L_1+\rho_{H2O}\frac{A_1}{A_2}L_2}$ Eq. 30 $\frac{d\theta}{dt} = \frac{\left(v_{asx}\frac{dv_{asz}}{dt}-v_{asz}\frac{dv_{asx}}{dt}\right)}{v_{asx}^2+v_{asz}^2}$ Eq. 31 $\frac{d\dot{L}_1}{dt} = v_1$ Eq. 32

$$\frac{dv_{Rx}}{dt} = \frac{\cos(\theta)\left[-\rho_{H2O}A_2L_2\frac{dv_2}{dt}\left(\frac{dL_1}{dt}(v_R+v_1)-L_1\frac{dv_1}{dt}\right)-\rho_{H2O}A_2L_2\frac{dv_2}{dt}-\rho_{H2O}A_2(v_R+v_2)v_2-\frac{1}{2}C_D\rho_{air}A_2v_{as}|v_{as}|\right]}{m_R+\rho_{H2O}A_1L_1+\rho_{H2O}A_2L_2} \quad \text{Eq. 33}$$

$$\frac{dv_{Ry}}{dt} = \frac{\sin(\theta)\left[-\rho_{H2O}A_2L_2\frac{dv_2}{dt}\left(\frac{dL_1}{dt}(v_R+v_1)-L_1\frac{dv_1}{dt}\right)-\rho_{H2O}A_2L_2\frac{dv_2}{dt}-\rho_{H2O}A_2(v_R+v_2)v_2-\frac{1}{2}C_D\rho_{air}A_2v_{as}|v_{as}|\right]}{m_R+\rho_{H2O}A_1L_1+\rho_{H2O}A_2L_2} - g \quad \text{Eq. 34}$$

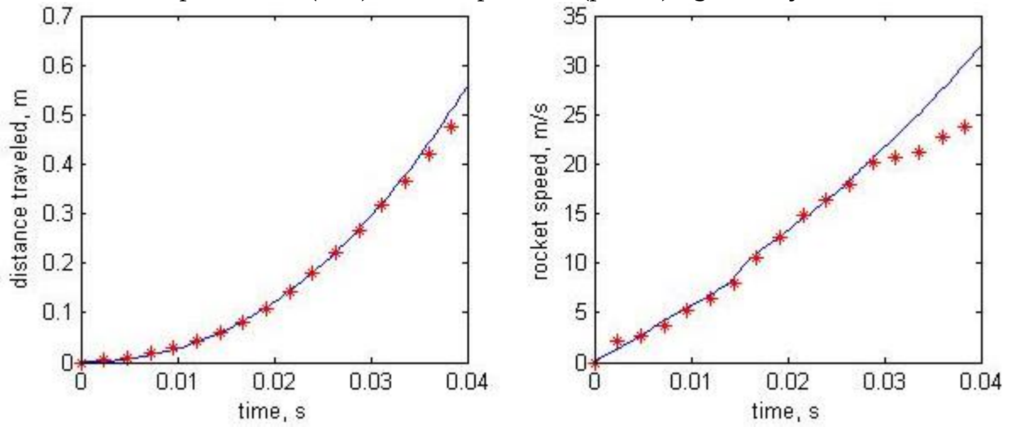
Figure 7: Results of a simulation with initial pressure of 40 psi and launch angle of 40 degrees



Model Validation and Tuning
To evaluate the accuracy of our model we compared it to test data from two sources: high-speed camera launch data provided by course staff, and flight distance data from our own tests [3]. The launch data allows validation of the launch phase, while the distance data allows validation of the combined launch and projectile motion phases.

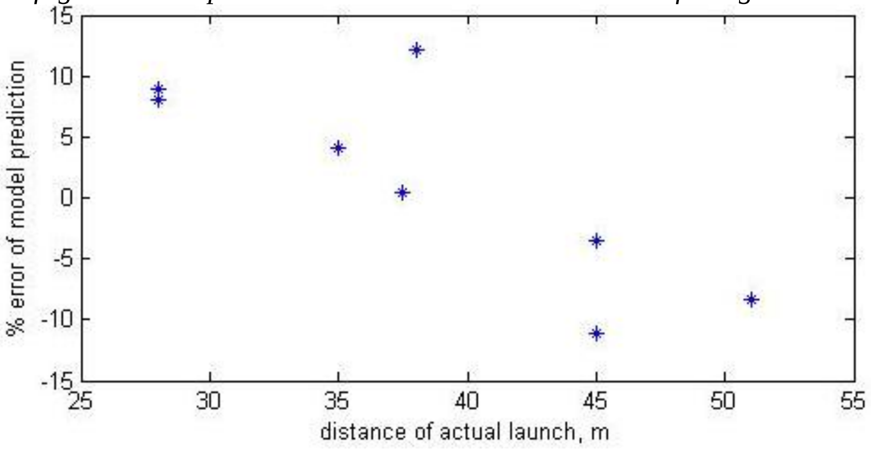
As shown in Figure 8, the model is very close to the sample data for time < 0.03 s. At time = 0.03 s the sample data velocity corners and separates from the model's prediction. Sample data was not available for time > 0.04 s so the behavior of the rocket afterward is not known. We attributed the model's velocity error to abnormal behavior occurring when the rocket left the air sting because that is when error begins to accumulate.

Figure 8: Launch model prediction (line) and sample data (points) agree very well until t >0.03s.



We decided to first evaluate the error in total distance prediction before attempting to improve the launch performance of the model because any changes made in the launch phase could negatively impact the distance prediction. We launched the rocket at varying initial angles and pressures, recording the distance traveled by each launch. We were not able to measure the wind velocity so it was assumed to be zero in the model. This accounts for some noise in the data. The ability of the algorithm to predict distances is shown in Figure 9, and we calculated it to be accurate to within 8.5±2.0% of the launch distance. There seems to be a negative trend in the error with increasing distance, but this was not explored in depth.

Figure 9: Error in flight distance prediction was between -15% and 15% for eight test launches.



Model Reduction

We conducted tests to determine whether all of the equations used in the working model were necessary and removed terms based on this analysis. This process is described in Appendix D and the resulting model is contained in Appendix E.

RESULTS

We have created a water bottle rocket, shown in Figure 2, page 2. In practice it fulfilled our desire for it to be stable and predictable. We mathematically characterized our rocket's drag coefficient and moment of inertia and created a mathematical model of the rocket's behavior, summarized in Equations 27-34 on page 8 and contained in MATLAB code Appendix E. A sample of simulation data is shown in Figure 7 on page 8. We tuned and validated this model using data from high-speed camera footage of a previous test and test launches of our rocket. The model proved to be accurate to within 8.5±2.0% of a target distance. This helped us predict the necessary initial conditions to apply to win our class section's hit-the-target competition. In the entire class competition our predictions were perfect at first, hitting the target in our practice launch. However, in the next launch a fin broke off and, due to insufficient repair materials, we were unable to fully repair it. The rocket became unstable and the model less accurately predicted its new, less-predictable behavior. We created a simplified version of our model, available in Appendix E and described in Appendix D, in an effort to determine what was truly important in the model. Finally, we created a lesson plan allowing middle school students to embark on the journey that we have just completed, albeit at a far lower technical level.

RECOMMENDATIONS

If we were to repeat this project or have more time to spend on it, there are several areas we would improve or focus on. With respect to rocket design, we would increase the mass to make it more resistant to gusts of wind. The biggest area of improvement for the model is the launch stage. We would like to gather additional high-speed camera footage, showing the position of the rocket until the water runs out. This would allow us to definitely evaluate the performance of the model in the launch phase. Additional investigation of what happens as the rocket passes the end of the sting could significantly improve the quality of the model. Also, we would like to gather additional flight distance data with wind velocity being recorded, which would provide better validation data. We estimate that with the information and changes above our model could be made twice as accurate, accurate to within 4%.

REFERENCES

1. Fox, Robert W., Robert W. Fox, Philip J. Pritchard, and Alan T. McDonald. *Fox and McDonald's Introduction to Fluid Mechanics*. Hoboken, NJ: John Wiley & Sons, 2011. Print.
2. Aerodynamics for Students, (c) Aerospace, Mechanical & Mechatronic Engg. 2005, University of Sydney. www-mdp.eng.cam.ac.uk/web/library/enginfo/aerothermal_dvd_only/aero/fprops/introvisc/
3. Personal communication, ME495 Instructional Staff. 100psi sample data & video. ctools.umich.edu

Appendix

Appendix A – Coefficients of drag and lift

Pre-Test Measurements

Before performing any tests we took pre-test measurements in order to calculate information for future reference. Pre-test measurements included recording ambient temperature, pressure, the dimensions of the wind tunnel, voltage of the power supply, and frequency of the current. The ambient temperature was found to be 69 ± 1 °F and the ambient pressure was 29.4 ±0.1 inHg. We used these two values to find the density of air to be 1.15 ± 0.11 kg/m^3. We also verified that the overload rods were seated correctly in the load cell apparatus for accurate readings. We obtained the dynamic viscosity of air at 70°F to be 3.82 Nm/s^2 from interpolations of table A.10 on page 796 [1]. Before performing any tests we first inspected the wind tunnel to make sure everything was plugged in, turned on, and that nothing was obstructing the tunnel.

Calibration

Before we could begin testing we first had to make calibration curves for the drag force, lift force, and the relationship between the speed setting and the actual air speed. The LabView program outputs units of mV/V (millivolts per volt) for forces of lift and drag. These units however are not useful for our analysis. The mV/V values need to be converted into Newtons. In order to convert from units of mV/V for lift and drag forces we created calibration curves using known weights and a Correx force gauge to do so. For drag forces we used the force gauge to apply a small force on the sting. We then recorded the value of mV/V that LabView outputs for that force. This was done four times to create a calibration curve (Figure A.1). For lift forces we placed our unmodified block inside the wind tunnel and nulled the LabView program. Then we placed weights on the block and obtained lift values in mV/V for the known weight. This was done with 6 weights to create the calibration curve for lift (Figure A.2) Next, by taking measurements of inches of H_2O from the manometer at various airspeed settings of the FINCOR 2300 MKII we were able to create a calibration curve for the air speed (Figure A.3). The air speed curve was used to verify the validity of the equation given in the manometer specification sheet in comparison to our system, so that it could be used for further data tabulation.

Figure A.1: Load cell calibration of recorded lift forces at specified weights.

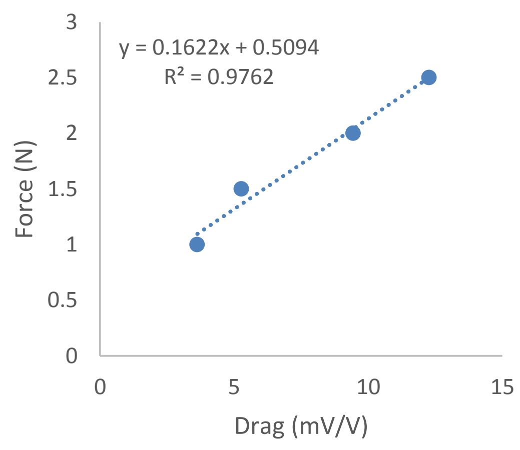


Figure A.2: Load cell calibration of recorded drag forces at specified loads.

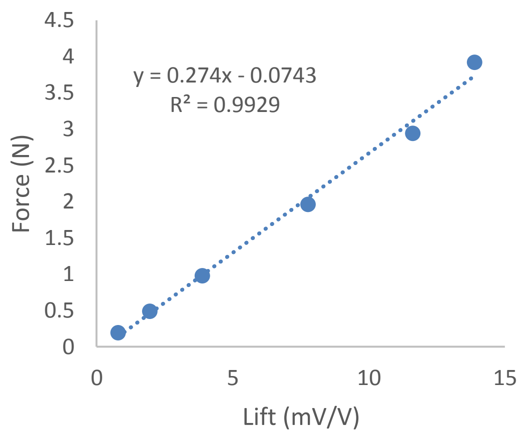


Figure A.3: Calibration of the inches of water reading shown on the manometer and the air speed readout on the FINCOR 2300 MKII.

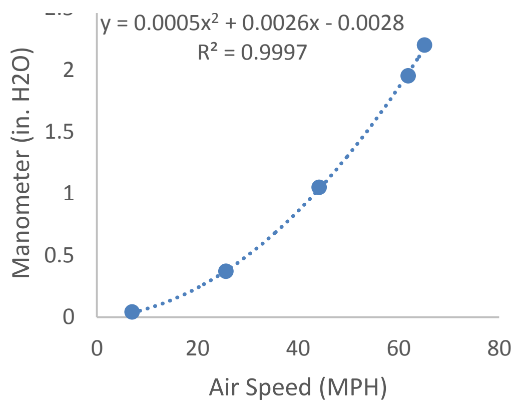
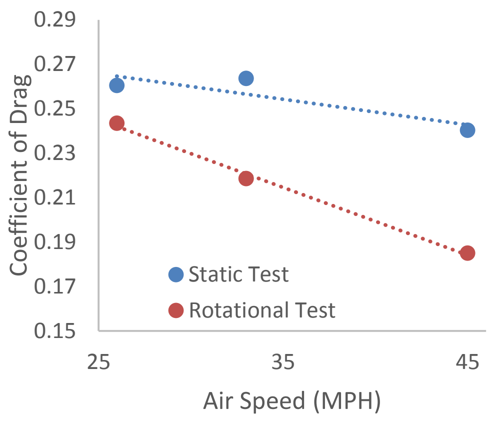


Figure A.4: Drag force of the static and rotational test with fit lines show a downward trend.



$$C_D = \frac{F_D}{\frac{1}{2}\rho_{air}V^2 A_{cs}} \quad \text{(Eq. A.1)}$$

$$C_L = \frac{F_L}{\frac{1}{2}\rho_{air}V^2 A_{pl}} \quad \text{(Eq. A.2)}$$

$$\frac{\rho_{air}VD}{\mu} = Re \quad \text{(Eq. A.3)}$$

$$P_1 + \frac{1}{2}\rho v_1^2 + \rho g\, h_1 = P_2 + \frac{1}{2}\rho v_2^2 + \rho g h_2 \quad \text{(Eq. A.4)}$$

Figure C.1: Reynolds Numbers vs. Coefficients of Drag for both the static and rotational test with linear trend lines shown

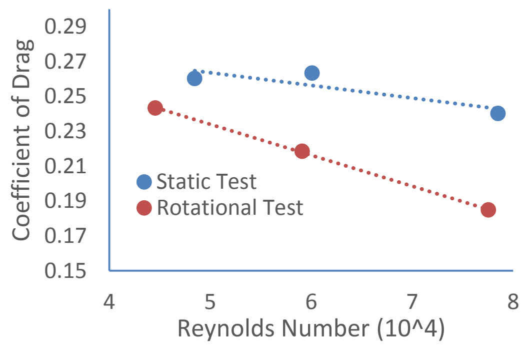


Table C.2: Coefficients of Lift and Drag and /Reynolds number for the Static and Rotational tests.

	Static Test			Rotational Test		
Airspeed (mph)	**Coefficient of Lift**	**Coefficient of Drag**	**Reynolds Number (10^4)**	**Coefficient of Lift**	**Coefficient of Drag**	**Reynolds Number (10^4)**
26	-0.09	0.26	2.05	0.01	0.24	1.89
33	-0.09	0.26	2.54	0.01	0.22	2.50
45	-0.11	0.24	3.32	0.06	0.18	3.27
Average	**-0.10**	**0.25**	**2.63**	**0.02**	**0.22**	**2.55**

Appendix B –Definition of variables in model creation

Variables

m	mass
t	time
p	momentum
P	pressure
A	area
L	length (shown in Figure 5, page 4)
F	force
V	volume
v	velocity
U	fluid velocity
θ	angle of rocket relative to horizontal
ω	angular velocity of the rocket about its longitudinal axis
γ	adiabatic index, 1.4 for air
ρ	density
z	height
g	acceleration due to gravity
C_D	coefficient of drag
I	moment of inertia of the rocket about its longitudinal axis

Subscripts

x	in the x-direction
y	in the y-direction
z	in the y-direction
as	airspeed, or relative to wind speed
0	initial
1	referring to point 1 in Figure 5, page 4
2	referring to point 2 in Figure 5, page 4
R	referring to the rocket structure
tot	referring to the total *something* of points 1, 2, and R
sys	an entire system
$H2O$	referring to water
air	referring to air
$static$	not spinning
ss	spinning at steady state (ss)
D	due to drag
G	due to gravity

Appendix C –Head loss analysis

We decided to try including head losses to improve the accuracy of the launch phase and reduce overestimation of total distance. To do this we started with Equation C.1

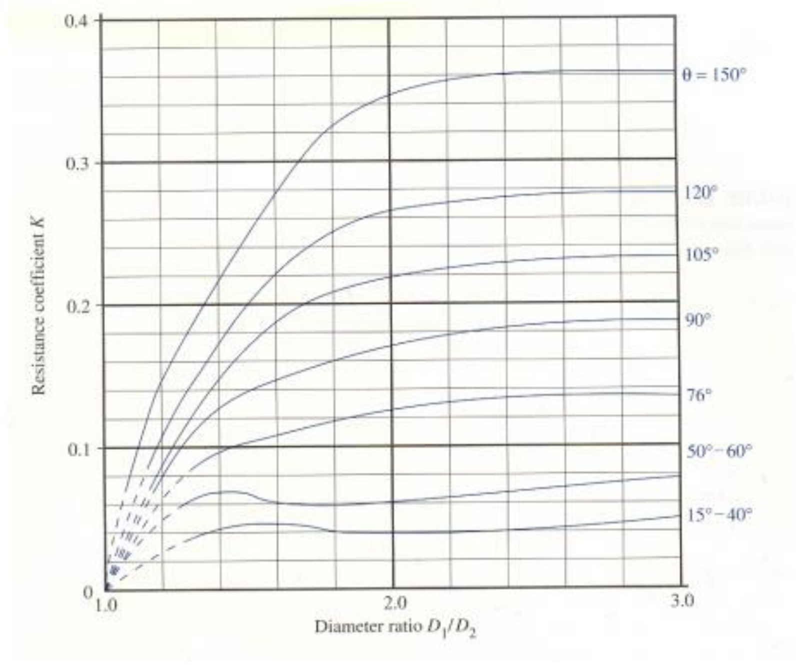
$$\frac{P_1}{\rho g}+\frac{V_1^2}{2g}+z_1=\frac{P_2}{\rho g}+\frac{V_2^2}{2g}+z_2+h_f \qquad \text{Eq. C.1}$$

Where h_f is the sum of the major and minor head losses of the system. We assumed laminar flow (which is dubious, as the system's flow is not steady and may be turbulent). For these conditions, major head loss is given by Equations C.2 and C.3, and minor head loss is given by Equation C.4 [C1][C2].

$h_{major} = \lambda\ (l / d_h)\ (v^2 / 2\ g)$ Eq. C.2 $\lambda = 64 / Re$ Eq. C.3 $h_{minor} = K(v_2^2/(2g))$ Eq. C.4

Where λ is the friction coefficient, v is the flow velocity, l is the length of the flow, d_h is the hydraulic diameter (equal to the diameter for a round flow), g is the acceleration due to gravity and Re is Reynold's number. K is chosen using Figure C.1, resulting in about 0.8 for our rocket. These equations were added into the Bernoulli equation in the model, Equation 30.

Figure C.1: Choose K based on contraction angle and diameter ratio [C2].



References
[C1] http://www.engineeringtoolbox.com/major-loss-ducts-tubes-d_459.html
[C2] http://udel.edu/~inamdar/EGTE215/Minor_loss.pdf

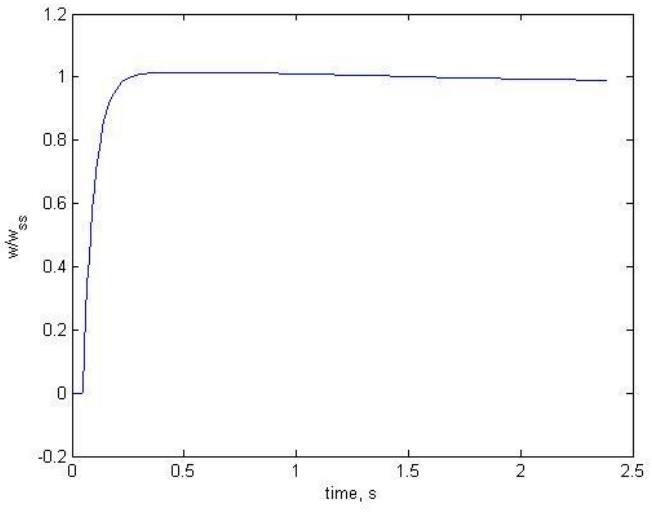
Appendix D – Model reduction

In an attempt to find Occam's razor with respect to this problem – the simplest functional solution – we conducted some tests to determine whether all model parameters and stipulations were entirely necessary. We would run a simulation, remove a term, rerun using the same initial conditions, and compare the total distance traveled. Unnecessary terms were removed from the model, yielding the MATLAB function simplefun.m in Appendix E. The following terms were found to be unnecessary to the function of the model:

Major head loss
The removal of this term resulted in a <1% change in launch distance. We believe that this is because the pipe lengths involved in the problem are small. Minor head loss, however, had a significant effect due to the high velocity and relatively large contraction angle.

Spin and drag coefficient interpolation
As seen in Figure 7, page 8, the rocket spins up very quickly, meaning that the drag coefficient described by Equation 26 is close to $C_{D,\,ss}$ for the vast majority of the flight. When this was set as the drag coefficient for the entire simulation, the difference in final distance was <1%. Setting the drag coefficient as a constant allowed the removal of the differential equations to describe the spin of the rocket, considerably simplifying the model. Again, the reason that the spinning drag coefficient worked well as the constant drag coefficient is because the rocket reaches steady-state angular velocity very quickly (within 0.5 seconds). This is shown again in Figure D.1.

Figure D.1: ω/ω_{ss} rapidly reaches and stays around 1. This allows us approximate the drag coefficient throughout the flight with a constant.



Pressure drop after leaving the air sting
This condition was implemented in the function-running scripts to improve the launch behavior with respect to the sample data. However, in order for it to have a significant positive effect on the launch the pressure drop had to so large that it negatively affected the flight distance. With the pressure drop at 2%, this was easily removed with <1% effect on flight distance.

Appendix E – Model MATLAB code and instructions

The following files are contained in text form here:

• model_sim.m	*script runs main function modelfun. Requires eventLA.m and eventZ0.m*
• modelfun.m	*main function for simulation*
• launch_val.m	*script runs launch validation function valfun. Requires eventLA.m and eventZ0.m*
• valfun.m	*main function for launch validation*
• simplefun.m	*simplified version of modelfun.m according to Appendix D*
• eventLA.m	*event function for rocket leaving the sting*
• eventZ0.m	*event function for rocket hitting the ground*

Alternatively, these files can be downloaded at https://github.com/clharman/rocket-ode. Downloading them there would almost certainly be easier than copying the text in this document.

Instructions to run a simulation:

1) Put model_sim.m, modelfun.m, eventLA.m, and eventZ0.m in the same folder
2) Open model_sim.m and enter desired initial conditions
3) (Some constant conditions, such as wind speed, are in modelfun.m)
4) Un-comment desired plot outputs
5) Run model_sim.m
6) MATLAB will output final distance and any uncommented plots

Instructions to run a launch validation:

1) Put launch_val.m, valfun.m, eventLA.m, and eventZ0.m in the same folder
2) Run launch_val.m
3) MATLAB will plot the velocity vs time and position vs time of the model and the sample data

model_sim.m

```
%This script runs modelfun with a set of initial conditions and
%  displays selected simulation results.
%Created for MECHENG 495 W14 Lab 4 by:
%  Section 003 Team 4: Colin Harman, Brian Freeburg, Joe Hendrickson
%CHANGE THESE-----------------------------------------------
L_0 = 0.064;            %m, initial height of water in bottle
theta_0 = 30;           %degrees, launch angle
psi = 30;
%ROCKET-SPECIFIC CONSTANTS----------------------------------
D_b = 0.0592;           %m, diameter of bottle
A_b = pi*D_b^2/4;       %cm^2, area of bottle
L_b = 0.196;            %m, length of bottle
V_0 = A_b*(L_b-L_0);    %m^3, initial volume of air in bottle
IC = [0, 0.064, 0, 0, 0, 0, psi*6894, theta_0*pi/180, V_0, 0, 0, 0];
%RUNS UNTIL ROCKET LEAVES STING
%pressure drop when leaving sting
options = odeset('Events',@eventLA);
[t1,y1] = ode45(@modelfun,[0, 6], IC,options);
y1(end,7) = y1(end,7) * 0.98;
%RUNS UNTIL ROCKET HITS GROUND
options = odeset('Events',@eventZ0,'InitialStep',t1(end)-t1(end-1));
[t2,y2] = ode45(@modelfun,[t1(end), t1(end)+30], y1(end,:), options);
```

```
%Concatenate arrays
t = cat(1, t1, t2);
y = cat(1, y1, y2);
linear_pos = sqrt(y(:,5).^2 + y(:,6).^2);
linear_vel = sqrt(y(:,3).^2 + y(:,4).^2);

%figure(1); plot(t,y(:,8));              title('angle vs t');
%figure(2); plot(t,linear_vel);          title('speed vs t');
%figure(3); plot(t,y(:,6));              title('height vs t');
%figure(3); plot(t,y(:,12));             title('y vs t');
%figure(4); plot(t,y(:,5));              title('horizontal distance vs t');
%figure(5); plot(y(:,5), y(:,6));        title('z vs x');
%figure(6); plot(t,y(:,3));              title('xvel vs time');
%figure(7); plot(t,y(:,4));              title('zvel vs time');
%figure(13); plot(t,y(:,11));            title('yvel vs time');
%figure(8); plot(t, y(:,8));             title('trajectory angle');
%figure(10); plot(t, y(:,10));           title('spin rate');

%creates multiplot of x-z position, speed, angle, spin rate:
%{
figure
subplot(1,4,1) % first subplot
plot(y(:,5), y(:,6))
%title('First subplot')
xlabel('x-position, m')
ylabel('z-position, m')
xlim([0 65])
ylim([-0.5 15])
subplot(1,4,2)
plot(t, y(:,8))
%title('First subplot')
ylabel('trajectory angle, rad')
xlabel('time, s')
%ylim([0 160])
subplot(1,4,3)
plot(t, linear_vel)
%title('First subplot')
ylabel('rocket speed, m/s')
xlabel('time, s')
subplot(1,4,4)
plot(t, y(:,10))
%title('First subplot')
ylabel('spin rate, rad/s')
xlabel('time, s')
ylim([0 160])
%}
y(end,5) * 3.28
```

modelfun.m

```
function dq = modelfun(t, q) %This function contains differential
 %  equations that describe the behavior of a water bottle rocket.
 %Created for MECHENG 495 W14 Lab 4 by:
 %  Section 003 Team 4: Colin Harman, Brian Freeburg, Joe Hendrickson
 %Written by Colin Harman with guidance from GSI Colin Jiang, 4/21/2015

%--------------------------------------------------------------------------
%-------------CONSTANTS----------------------------------------------------
%Rocket dimensions
L_b = 0.196;                 %m, length of bottle
L_2 = 0.0203;                %m, length of nozzle
D_b = 0.0592;                %m, diameter of bottle
D_n = 0.0218;                %m, diameter of nozzle
A_b = pi*D_b^2/4;            %m^2, area of bottle (calculated from D_b)
A_n0 = pi*D_n^2/4;           %m^2, area of nozzle (calculated from D_n)
A_fin = 0.0023;              %m^2, area of each fin

%Other rocket characteristics
m_r = 0.06478;               %kg, mass of rocket without water
aoa_fin = 9.6*pi/180;        %rad, angle of attack of fins
n_fins = 3;                  %number of fins
Cd_lock = 0.28;              %-, drag coefficient of rocket when still
Cd_spin = 0.21;              %-, drag coefficient of rocket when spinning
Cd_y = 1.17;                 %-, drag coefficient for side of rocket
I_r = 0.000027;              %kg m^2, moment of inertia of empty rocket about
long axis
K_minor = 0*0.05;              %-, resistance coefficient of bottle-nozzle
contraction

%Launcher dimensions
D_a = 0.0103;                %m, diameter of air sting
L_a = 0.209;                 %m, length of air sting
L_s = 0.057;                 %m, length of seal
A_s = pi*D_a^2/4;            %cm^2, area of air sting (calculated from D_a)

%Physical constants
rho_h2o = 997;              %kg/m^3, density of water
rho_air = 1.225;             %kg/m^3, density of air (uncompressed)
g = 9.807;                   %m/s^2, acceleration due to gravity
mu = 0.00089;                %Pa s, dynamic viscosity of water at 25 C
%eps_pl = 0.0000025;         %mm, roughness of plastic
%rpr_1 = eps_pl/D_b;         %-, relative pipe roughness, bottle
%rpr_2 = eps_pl/D_n;         %-, relative pipe roughness, nozzle

%Environmental variables
vy_wind = 0;                 %m/s, wind speed in the y direction (2.23 m/s = 5
mph)
vx_wind = 0;                 %m/s, wind speed in the x direction

%--------------------------------------------------------------------------
%-------------STATE VARIABLES----------------------------------------------
v_1 = q(1);                  %velocity of flow in the bottle
```

```
L_1 = q(2);                  %height of water in the bottle
vrx = q(3);                  %velocity of rocket in x
vrz = q(4);                  %velocity of rocket in z
x = q(5);                    %rocket position in x
z = q(6);                    %rocket position in z
P_0 = q(7);                  %initial gage pressure in pa (so that a pressure
drop can be affected externally)
theta = q(8);                %pointing angle of the rocket in radians
V_0 = q(9);                  %initial volume of air (so that it does not need
to be changed inside the function)
omega = q(10);               %spin speed of the rocket in rad/s
vry = q(11);                 %velocity of rocket in y
y = q(12);                   %rocket position in y


%--------------------------------------------------------------------------
%-------------CONSTANT ALGEBRAIC EXPRESSIONS-(never change)----------------
lin = sqrt(x^2+z^2);         %magnitude of distance from launch point
vlin = sqrt(vrx^2+vrz^2);    %magnitude of x-z velocity
V = (L_b-L_1)*A_b;           %air volume
P = P_0*V_0^1.4/V^1.4;       %air pressure
asx = vrx - vx_wind;         %airspeed in x direction
asy = vry - vy_wind;         %airspeed in y direction
aslin = sqrt(asx^2+vrz^2);   %magnitude of x-z windspeed

%--------------------------------------------------------------------------
%-------------LAUNCH CONDITIONS--------------------------------------------
if lin < L_s                 %rocket is on the launcher seal
    on_seal = true;          %   switch
else
    on_seal = false;
end

if lin < L_a                 %rocket is on the air sting
    A_n = A_n0-A_s;          %   effective nozzle area is smaller
else
    A_n = A_n0;
end

if(L_1 <= 0)                 %all fuel is gone
    empty = true;            %   switch
else
    empty = false;
end

%--------------------------------------------------------------------------
%-------------VARIABLE ALGEBRAIC EXPRESSIONS-(depend on launch conditions)-
%Constant flow rate
v_2 = A_b/A_n * v_1;
Re_1 = rho_h2o*v_1*D_b/mu;
Re_2 = rho_h2o*v_2*D_n/mu;
f_d_1 = 0*64/Re_1;
f_d_2 = 0*64/Re_2;
%Spin equations
phi_fin = aoa_fin - atan(omega*D_b/2/vlin);
Cl = 2*pi*phi_fin;
```

```
omega_ss = 2*vlin*tan(aoa_fin)/D_b;
Cd = Cd_lock - omega/omega_ss*(Cd_lock-Cd_spin);

%---------------------------------------------------------------------------
%-------------DERIVATIVE DEFINITIONS----------------------------------------
if empty
    dv_1 = 0;
    dL_1 = 0;
    L_2 = 0;
    v_2 = 0;
    domega = n_fins*D_b/2*0.5*rho_air*vlin^2*A_fin*Cl/I_r;
else
    dL_1 = -v_1;
    dv_1 = ((P)+rho_h2o/2*v_1^2*(1-(A_b/A_n)^2) -
rho_h2o*f_d_1*L_1/D_b*(v_1^2)/2 - rho_h2o*f_d_2*L_2/D_n*(v_2^2)/2 -
rho_h2o*K_minor*(v_2^2)/2 ) / (rho_h2o*L_1 + rho_h2o*A_b/A_n*L_2);
    domega = 0;
end
if on_seal
    dtheta = 0;
    dvrx = cos(theta)*P*A_n0 / (m_r+rho_h2o*A_b*L_1 + rho_h2o*A_n*L_2);
    dvrz = sin(theta)*P*A_n0 / (m_r+rho_h2o*A_b*L_1 + rho_h2o*A_n*L_2);
    dv_1 = A_n / A_b * sqrt(dvrx^2 + dvrz^2);
    dvry = 0;
else
    dv2 = A_b/A_n * dv_1;
    dvrx = (cos(theta)*-rho_h2o*A_b*(dL_1*(vlin-v_1)-L_1*dv_1) +
cos(theta)*rho_h2o*A_n*L_2*dv2 - cos(theta)*rho_h2o*A_n*(vlin-v_2)*v_2 -
cos(theta)*(Cd/2)*rho_air*A_b*aslin^2)/(m_r+rho_h2o*A_b*L_1 +
rho_h2o*A_n*L_2);
    dvrz = (sin(theta)*-rho_h2o*A_b*(dL_1*(vlin-v_1)-L_1*dv_1) +
sin(theta)*rho_h2o*A_n*L_2*dv2 - sin(theta)*rho_h2o*A_n*(vlin-v_2)*v_2 -
(m_r+rho_h2o*A_b*L_1)*g -
sin(theta)*(Cd/2)*rho_air*A_b*aslin^2)/(m_r+rho_h2o*A_b*L_1 +
rho_h2o*A_n*L_2);
    dvry = 0.5*rho_air*Cd_y*(asy)^2*D_b*L_b/(m_r+rho_h2o*A_b*L_1 +
rho_h2o*A_n*L_2);
    dtheta = (asx*dvrz - vrz*dvrx)/(asx^2 + vrz^2);
end
dx = vrx;
dz = vrz;
dy = vry;
%---------------------------------------------------------------------------
%-------------STATE DERIVATIVES---------------------------------------------
dq = zeros(length(q),1);
dq(1) = dv_1;
dq(2) = dL_1;
dq(3) = dvrx;
dq(4) = dvrz;
dq(5) = dx;
dq(6) = dz;
dq(7) = 0;              %initial pressure does not change internally
dq(8) = dtheta;
dq(9) = 0;              %initial volume does not change internally
dq(10) = domega;
dq(11) = dvry;
dq(12) = dy;
```

launch_val.m

```
psi = 100;
L_0 = 0.0793;
theta_0 = 32;
D_b = 0.0762;       %m, diameter of bottle
A_b = pi*D_b^2/4;   %cm^2, area of bottle
L_b = 0.2;          %m, length of bottle
V_0 = A_b*(L_b-L_0);
IC = [0, L_0, 0, 0, 0, 0, psi*6894, theta_0*pi/180, V_0, 0, 0, 0];

options = odeset('Events',@eventLA);
[t1,y1] = ode45(@valfun,[0, 0.05], IC, options);
y1(end,7) = y1(end,7) * 0.95;
%y1(end,1) = y1(end,1) * 0.1;
[t2,y2] = ode45(@valfun,[t1(end), 0.04], y1(end,:), options);

y = cat(1, y1, y2);
t = cat(1, t1, t2);

% time (s)
sampleT = [ 0.0000 0.0024 0.0048 0.0072 0.0096 0.0120 0.0144 ...
          0.0168 0.0192 0.0216 0.0240 0.0264 0.0288 0.0312 ...
          0.0336 0.0360 0.0384 0.0408 ];
% distance (m)
sampleX = [ 0.0000 0.0051 0.0102 0.0178 0.0279 0.0432 0.0584 ...
          0.0813 0.1092 0.1422 0.1803 0.2210 0.2667 0.3175 ...
          0.3658 0.4191 0.4750 0.5334 ];
% time step (s)
% fps = 1250; 3 is a magic number, taken from the spreadsheet
dT = 3 / 1250;
% calculate velocity
sampleV = zeros(1, length(sampleX)-1);
for i = 2 : length(sampleX) - 1
    % using central difference
    sampleV(i) = ( sampleX(i+1) - sampleX(i-1) ) / ( 2*dT );
end

linear_pos = sqrt(y(:,5).^2 + y(:,6).^2);
linear_vel = sqrt(y(:,3).^2 + y(:,4).^2);

figure(1);
subplot(1,2,2)
plot(t,linear_vel, sampleT(1:end-1),sampleV, '*r');
ylabel('rocket speed, m/s')
xlabel('time, s')
subplot(1,2,1)
plot(t,linear_pos, sampleT(1:end),sampleX, '*r');
xlim([0 0.04])
ylabel('distance traveled, m')
xlabel('time, s')
```

valfun.m

```
function dq = fcn1(t, q)
%--------------------------------------------------------------------------
%-------------CONSTANTS----------------------------------------------------
%Rocket dimensions
L_b = 0.2;                  %m, length of bottle
L_2 = 0.0203;                %m, length of nozzle
D_b = 0.0762;                %m, diameter of bottle
D_n = 0.0218;                %m, diameter of nozzle
A_b = pi*D_b^2/4;            %m^2, area of bottle (calculated from D_b)
A_n0 = pi*D_n^2/4;           %m^2, area of nozzle (calculated from D_n)
%Other rocket characteristics
m_r = 0.084;                 %kg, mass of rocket without water
Cd = 0.2;                    %-, drag coefficient of rocket when spinning
K_minor = 0.1;              %-, resistance coefficient of bottle-nozzle
contraction
%Launcher dimensions
D_a = 0.0103;                %m, diameter of air sting
L_a = 0.209;                 %m, length of air sting
L_s = 0.057;                 %m, length of seal
A_s = pi*D_a^2/4;            %cm^2, area of air sting (calculated from D_a)
%Physical constants
rho_h2o = 997;              %kg/m^3, density of water
rho_air = 1.225;             %kg/m^3, density of air (uncompressed)
g = 9.807;                   %m/s^2, acceleration due to gravity
mu = 0.00089;                %Pa s, dynamic viscosity of water at 25 C
%Environmental variables
%--------------------------------------------------------------------------
%-------------STATE VARIABLES----------------------------------------------
v_1 = q(1);                  %velocity of flow in the bottle
L_1 = q(2);                  %height of water in the bottle
vrx = q(3);                  %velocity of rocket in x
vrz = q(4);                  %velocity of rocket in z
x = q(5);                    %rocket position in x
z = q(6);                    %rocket position in z
P_0 = q(7);                  %initial gage pressure in pa (so that a pressure
drop can be affected externally)
theta = q(8);                %pointing angle of the rocket in radians
V_0 = q(9);                  %initial volume of air (so that it does not need
to be changed inside the function)
omega = q(10);               %spin speed of the rocket in rad/s
vry = q(11);                 %velocity of rocket in y
y = q(12);                   %rocket position in y
%--------------------------------------------------------------------------
%-------------CONSTANT ALGEBRAIC EXPRESSIONS-(never change)----------------
lin = sqrt(x^2+z^2);         %magnitude of distance from launch point
vlin = sqrt(vrx^2+vrz^2);    %magnitude of x-z velocity
V = (L_b-L_1)*A_b;           %air volume
P = P_0*V_0^1.4/V^1.4;       %air pressure
asx = vrx - 0;               %airspeed in x direction
asy = vry - 0;               %airspeed in y direction
aslin = sqrt(asx^2+vrz^2);   %magnitude of x-z windspeed
%--------------------------------------------------------------------------
%-------------LAUNCH CONDITIONS--------------------------------------------
```

```
if lin < L_s                 %rocket is on the launcher seal
    on_seal = true;          %   switch
else
    on_seal = false;
end
if lin < L_a                 %rocket is on the air sting
    A_n = A_n0-A_s;          %   effective nozzle area is smaller
else
    A_n = A_n0;
end

if(L_1 <= 0)                 %all fuel is gone
    empty = true;            %   switch
else
    empty = false;
end
%--------------------------------------------------------------------------
%-------------VARIABLE ALGEBRAIC EXPRESSIONS-(depend on launch conditions)-
%Constant flow rate
v_2 = A_b/A_n * v_1;
Re_1 = rho_h2o*v_1*D_b/mu;
Re_2 = rho_h2o*v_2*D_n/mu;
f_d_1 = 64/Re_1;
f_d_2 = 64/Re_2;
%--------------------------------------------------------------------------
%-------------DERIVATIVE DEFINITIONS---------------------------------------
if empty
    dv_1 = 0;
    dL_1 = 0;
    L_2 = 0;
    v_2 = 0;
else
    dL_1 = -v_1;
    dv_1 = (P+rho_h2o/2*v_1^2*(1-(A_b/A_n)^2) -
rho_h2o*f_d_1*L_1/D_b*(v_1^2)/2 - rho_h2o*f_d_2*L_2/D_n*(v_2^2)/2 -
rho_h2o*K_minor*(v_2^2)/2 ) / (rho_h2o*L_1 + rho_h2o*A_b/A_n*L_2);
end

if on_seal
    dtheta = 0;
    dvrx = cos(theta)*P*A_n0 / (m_r+rho_h2o*A_b*L_1 + rho_h2o*A_n*L_2);
    dvrz = sin(theta)*P*A_n0 / (m_r+rho_h2o*A_b*L_1 + rho_h2o*A_n*L_2);
    dv_1 = A_n / A_b * sqrt(dvrx^2 + dvrz^2);
else
    dv2 = A_b/A_n * dv_1;
    dvrx = (cos(theta)*-rho_h2o*A_b*(dL_1*(vlin-v_1)-L_1*dv_1) +
cos(theta)*rho_h2o*A_n*L_2*dv2 - cos(theta)*rho_h2o*A_n*(vlin-v_2)*v_2 -
cos(theta)*(Cd/2)*rho_air*A_b*aslin^2)/(m_r+rho_h2o*A_b*L_1 +
rho_h2o*A_n*L_2);
    dvrz = (sin(theta)*-rho_h2o*A_b*(dL_1*(vlin-v_1)-L_1*dv_1) +
sin(theta)*rho_h2o*A_n*L_2*dv2 - sin(theta)*rho_h2o*A_n*(vlin-v_2)*v_2 -
(m_r+rho_h2o*A_b*L_1)*g -
sin(theta)*(Cd/2)*rho_air*A_b*aslin^2)/(m_r+rho_h2o*A_b*L_1 +
rho_h2o*A_n*L_2);
    dtheta = (asx*dvrz - vrz*dvrx)/(asx^2 + vrz^2);
end
```

```
dx = vrx;
dz = vrz;
%---------------------------------------------------------------------------
%-------------STATE DERIVATIVES--------------------------------------------
dq = zeros(length(q),1);
dq(1) = dv_1;
dq(2) = dL_1;
dq(3) = dvrx;
dq(4) = dvrz;
dq(5) = dx;
dq(6) = dz;
dq(7) = 0;              %initial pressure does not change internally
dq(8) = dtheta;
dq(9) = 0;              %initial volume does not change internally
dq(10) = 0;             %no spin
dq(11) = 0;             %no y-acceleration
dq(12) = 0;             %no y-velocity
```

simplefun.m

```
function dq = simplefun(t, q) %This function contains differential
 %  equations that describe the behavior of a water bottle rocket.
 %This is a simplified version of modelfun.m
 %Created for MECHENG 495 W14 Lab 4 by:
 %  Section 003 Team 4: Colin Harman, Brian Freeburg, Joe Hendrickson
 %Written by Colin Harman with guidance from GSI Colin Jiang, 4/21/2015
%---------------------------------------------------------------------------
%-------------CONSTANTS----------------------------------------------------
%Rocket dimensions
L_b = 0.196;                %m, length of bottle
L_2 = 0.0203;               %m, length of nozzle
D_b = 0.0592;               %m, diameter of bottle
D_n = 0.0218;               %m, diameter of nozzle
A_b = pi*D_b^2/4;           %m^2, area of bottle (calculated from D_b)
A_n0 = pi*D_n^2/4;          %m^2, area of nozzle (calculated from D_n)
%Other rocket characteristics
m_r = 0.06478;              %kg, mass of rocket without water
Cd_spin = 0.21;             %-, drag coefficient of rocket when spinning
Cd_y = 1.17;                %-, drag coefficient for side of rocket
%Launcher dimensions
D_a = 0.0103;               %m, diameter of air sting
L_a = 0.209;                %m, length of air sting
L_s = 0.057;                %m, length of seal
A_s = pi*D_a^2/4;           %cm^2, area of air sting (calculated from D_a)
%Physical constants
rho_h2o = 997;             %kg/m^3, density of water
rho_air = 1.225;            %kg/m^3, density of air (uncompressed)
g = 9.807;                  %m/s^2, acceleration due to gravity
%Environmental variables
vy_wind = 0;             %m/s, wind speed in the y direction (2.23 m/s = 5
mph)
vx_wind = 0;                %m/s, wind speed in the x direction
%---------------------------------------------------------------------------
%-------------STATE VARIABLES----------------------------------------------
v_1 = q(1);                 %velocity of flow in the bottle
L_1 = q(2);                 %height of water in the bottle
```

```
vrx = q(3);                 %velocity of rocket in x
vrz = q(4);                 %velocity of rocket in z
x = q(5);                   %rocket position in x
z = q(6);                   %rocket position in z
P_0 = q(7);                 %initial gage pressure in pa (so that a pressure
drop can be affected externally)
theta = q(8);               %pointing angle of the rocket in radians
V_0 = q(9);                 %initial volume of air (so that it does not need
to be changed inside the function)
omega = q(10);              %spin speed of the rocket in rad/s
vry = q(11);                %velocity of rocket in y
y = q(12);                  %rocket position in y
%---------------------------------------------------------------------------
%-------------CONSTANT ALGEBRAIC EXPRESSIONS-(never change)-----------------
lin = sqrt(x^2+z^2);        %magnitude of distance from launch point
vlin = sqrt(vrx^2+vrz^2);   %magnitude of x-z velocity
V = (L_b-L_1)*A_b;          %air volume
P = P_0*V_0^1.4/V^1.4;      %air pressure
asx = vrx - vx_wind;        %airspeed in x direction
asy = vry - vy_wind;        %airspeed in y direction
aslin = sqrt(asx^2+vrz^2);  %magnitude of x-z windspeed
Cd = Cd_spin;
%---------------------------------------------------------------------------
%-------------LAUNCH CONDITIONS---------------------------------------------
if lin < L_s                %rocket is on the launcher seal
    on_seal = true;         %   switch
else
    on_seal = false;
end

if lin < L_a                %rocket is on the air sting
    A_n = A_n0-A_s;         %   effective nozzle area is smaller
else
    A_n = A_n0;
end

if(L_1 <= 0)                %all fuel is gone
    empty = true;           %   switch
else
    empty = false;
end
%---------------------------------------------------------------------------
%-------------VARIABLE ALGEBRAIC EXPRESSIONS-(depend on launch conditions)-
%Constant flow rate
v_2 = A_b/A_n * v_1;
%---------------------------------------------------------------------------
%-------------DERIVATIVE DEFINITIONS----------------------------------------
if empty
    dv_1 = 0;
    dL_1 = 0;
    L_2 = 0;
    v_2 = 0;
else
    dL_1 = -v_1;
    dv_1 = (P+rho_h2o/2*v_1^2*(1-(A_b/A_n)^2)) / (rho_h2o*L_1 +
rho_h2o*A_b/A_n*L_2);
```

```
end

if on_seal
    dtheta = 0;
    dvrx = cos(theta)*P*A_n0 / (m_r+rho_h2o*A_b*L_1 + rho_h2o*A_n*L_2);
    dvrz = sin(theta)*P*A_n0 / (m_r+rho_h2o*A_b*L_1 + rho_h2o*A_n*L_2);
    dv_1 = A_n / A_b * sqrt(dvrx^2 + dvrz^2);
    dvry = 0;
else
    dv2 = A_b/A_n * dv_1;
    dvrx = (cos(theta)*-rho_h2o*A_b*(dL_1*(vlin-v_1)-L_1*dv_1) +
cos(theta)*rho_h2o*A_n*L_2*dv2 - cos(theta)*rho_h2o*A_n*(vlin-v_2)*v_2 -
cos(theta)*(Cd/2)*rho_air*A_b*aslin^2)/(m_r+rho_h2o*A_b*L_1 +
rho_h2o*A_n*L_2);
    dvrz = (sin(theta)*-rho_h2o*A_b*(dL_1*(vlin-v_1)-L_1*dv_1) +
sin(theta)*rho_h2o*A_n*L_2*dv2 - sin(theta)*rho_h2o*A_n*(vlin-v_2)*v_2 -
(m_r+rho_h2o*A_b*L_1)*g -
sin(theta)*(Cd/2)*rho_air*A_b*aslin^2)/(m_r+rho_h2o*A_b*L_1 +
rho_h2o*A_n*L_2);
    dvry = 0.5*rho_air*Cd_y*(asy)^2*D_b*L_b/(m_r+rho_h2o*A_b*L_1 +
rho_h2o*A_n*L_2);
    dtheta = (asx*dvrz - vrz*dvrx)/(asx^2 + vrz^2);
end
dx = vrx;
dz = vrz;
dy = vry;
%---------------------------------------------------------------------------
%-------------STATE DERIVATIVES--------------------------------------------
dq = zeros(length(q),1);
dq(1) = dv_1;
dq(2) = dL_1;
dq(3) = dvrx;
dq(4) = dvrz;
dq(5) = dx;
dq(6) = dz;
dq(7) = 0;              %initial pressure does not change internally
dq(8) = dtheta;
dq(9) = 0;              %initial volume does not change internally
dq(10) = domega;
dq(11) = dvry;
dq(12) = dy;
```

eventLA.m

```
function [value,isterminal,direction] = eventLA(t,y)
% Locate the time when the rocket passes the sting
value = sqrt(y(5)^2+y(6)^2)-0.197;     % Detect sting passed
isterminal = 1;   % Stop the integration
direction = 0;
```

eventZ0.m

```
function [value,isterminal,direction] = eventZ0(t,y)
% Locate the time when the rocket passes the sting
value = y(6)+0.4;     % launch zero is greater than zero
isterminal = 1;   % Stop the integration
direction = 0;
```

Appendix F – Middle school water bottle rocket lesson plan

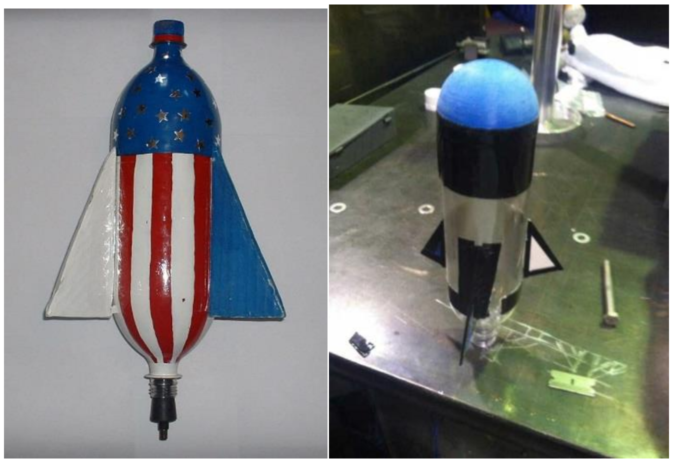
Purpose:
The purpose of this lab is to increase interest in science and engineering through an interactive lab on water bottle rockets. Students will learn about the importance of rocket mass, rocket spin, nose cone design, and nozzle shape on rocket dynamics. Students will experiment with different rocket designs in order to accurately land their rocket closest to a target.

Materials:
1) 20 fluid oz. water, juice, or soda bottles for rocket body, nozzle, and nose cone
2) Ping pong balls for nose tip
3) Plastic sheet, cardboard, or balsa wood for fins
4) Metal washers or modeling clay for rocket ballast
5) Super glue, super glue, or 5 minute epoxy for assembly of components
6) Tape for assembly and to reduce drag
7) Utility or hobby knife to cut and trim components
8) Stickers to decorate rocket
9) Water bottle launcher

Safety
1) Safety glasses must be worn at all times during manufacturing of rocket.
2) Gloves must be worn when handling super glue and 5 minute epoxy.
3) When launching rocket, classmates not directly participating in the prepping and launch of a rocket should stand at a distance behind the launcher
4) Safety glasses must be worn at all times during testing and launch of rocket

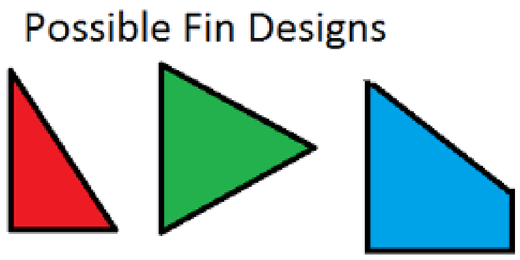
Lesson Plan
Take a 20 oz. bottle to be used as the main rocket body and nozzle. Another bottle can be cut to create the nose cone. The top end of a soda bottle can be used to create a conical nose cone. A ping pong ball may also be used to create a rounded tip shape. Both of these designs can be seen below. It is up to the students to decide on which method and to experiment with it.

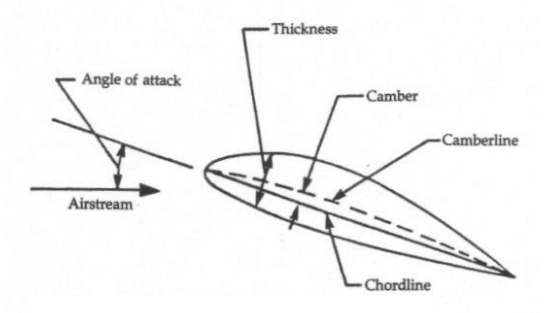
Figure F.1: Conical and round nose designs on water bottle rockets



Card board, plastic, or balsa wood can be used to create the fin design. Students will have to decide on how many fins as well as the fin shape. Some design ideas are given below. Students will also decide on the angle of attack of the fins. Fin angle of attack that is will induce rocket spin during flight.

Figure F.2: Some fin shape ideas as well as an explanation of fin parameters





Fins can be secured onto the rocket body using super glue, tape, or 5 minute epoxy. Tape is not the most ideal choice since it does not prevent fins from falling off during rocket landing. Super glue and 5 minute epoxy create a much more secure connection. Gloves must be worn when handling super glue and 5 minute epoxy to prevent chemicals from contacting skin. 5 minute epoxy should be mixed on scrap card board before applying. The figure below shows where the adhesive should be applied.

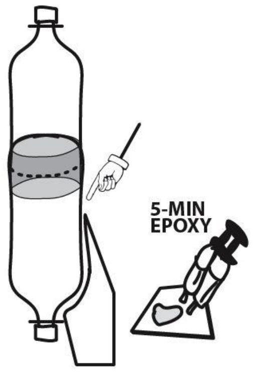
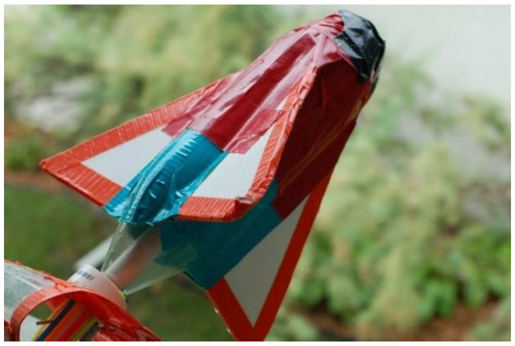


Figure 3: Location of adhesive to secure fins

Rocket ballast should be added underneath the nose cone as metal washers or modeling clay. Students must decide on how much ballast required. Mass should be placed as evenly around the rocket central axis for balance during flight. The nose cone can then be placed onto the rocket body by either tape or adhesive. Students should remember to keep their rocket as adjustable as possible in order to experiment with rocket parameters. Gluing the nose cone to the rocket body is not ideal because rocket ballast cannot be adjusted. Tape can be used to smooth the gap between nose cone and rocket body. This will decrease drag. Tape can also be used to decrease drag from the adhesive connections between the rocket body and fins. Once finished, rockets should look similar to Figure 4.

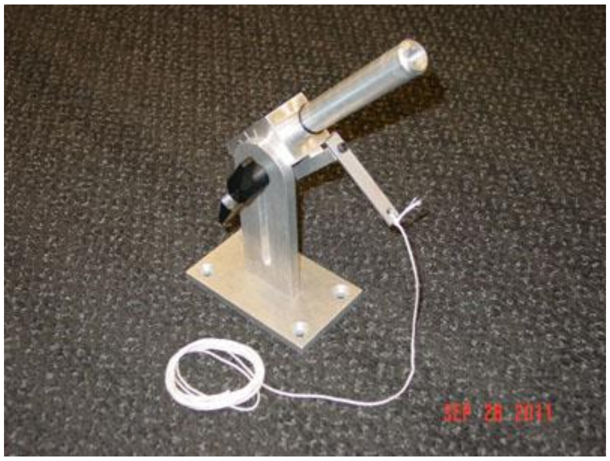
Figure 4: Finished water bottle rocket with drag reducing tape



Launching

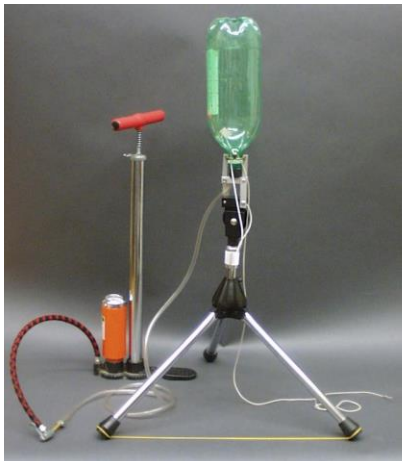
When launching water bottle rockets, a launcher supplied by the instructor or purchased online must be used. Figure 5 shows one possible launcher that can be used. The purpose of this lab is to create a rocket that land accurately on a target. A launcher that shoots at angles from horizontal to vertical would be the most ideal so that launch angle can be used as a varied parameter.

Figure 5: One possible rocket launcher with adjustable launch angle.



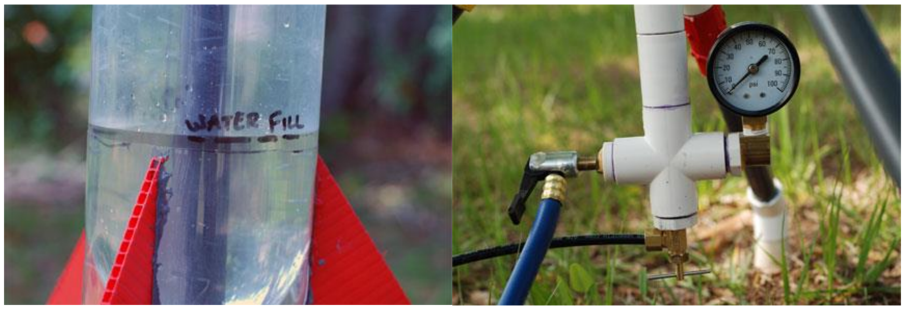
The water bottle rocket should be loaded onto the launcher and secured for water filling and pressurizing. The rocket should be placed in a vertical position for water filling. Figure 6 below shows the initial water bottle and launcher set up.

Figure 6: Initial water bottle and launcher set up for water filling and pressurizing



Two of the rocket parameters that play an important role in rocket dynamics are water fill level and air pressure within the rocket. Students should pick a couple water levels and mark on the level on their rockets to record and experiment with the amount of water. Figure 7 shows how to mark the water level. Water level should not exceed 70% of the rocket body internal volume. Air pressure should also be recorded and experimented with. Figure 7 shows air pressure measurement. Air pressure should not exceed 70psi. It is up to the students to experiment and decide on what water levels and air pressure to test and to determine which of these parameters are more important to the dynamics of the rocket. Launch angle should also be adjusted along with water level and air pressure.

Figure 7: Water level mark and air pressure measurement. Both are important parameters for rocket dynamics



When launching the rocket, all students not directly participating in the launch of a rocket should stand at least 20ft away and behind the launcher. The distance between the target and the landing position of the rocket should be measured and recorded to determine the optimal water level, air pressure, and launch angle for a given rocket design. It is recommended to pick a water level and vary only the air pressure and launch angle. Picking a water level decreases the amount of trials required to fully characterize the dynamics of a rocket's motion based upon water amount, air pressure, and launch angle.