The Big Flying Toad

Authors

Karan Motwani

Nicolas Fransen

Sam Hagner

Saumya Navani

Tyler Battas

Tyler Sarvady

AAE251—Team R16

Fall 2019

Abstract —

In the wake of the zombie apocalypse, it is paramount that the remnants of humanity locate one another, come together, and unite against the threat. To accomplish this task, the Toads have developed a platform to launch a 5 kg probe capable of scanning for survivors of the calamity. The platform had to be reliable, but also simple, safe, and within a limited budget, due to the new world condition. The Toads used these criteria to decide on a multi-stage solid rocket motor powered rocket as the most viable platform for delivering the payload to the necessary height of 100 km, or just past the Karman line. In order to further design the rocket, a simulation function was developed to account for changes in thrust with altitude, drag, mass, and staging. This was called multiple times with different mass parameters to determine the best allocation of fuel between the two stages, ideal throat expansion ratios, as well as the lowest possible mass for the rocket. This made use of historical data for similar rockets, namely the ARCAS sounding rocket, previously used for weather observations. With major parts of the rocket determined by this simulation code, the only remaining decisions had to do with the form factor and minimizing drag while still having room for the payload. A hemispherical nose cone was chosen due to the ease of manufacture, as well as it being the ideal compromise between volumetric aerodynamic efficiency. The result of this development is the BFT, or Big Flying Toad, which is capable of safely sending the 5 kg payload to the designated height of 100 km with a total mass under 400 kg.

Table of Contents

1 Introduction

The Toads' goal with this project proposal is to detail the advantages of our design, as well as justifying the major decisions we made to arrive at our final design. With the stakes as high as they are due to the apocalypse and the resulting lack of otherwise readily available launch vehicles, we recognize the importance of a simple, cheap, and robust design that is guaranteed to work on the first attempt. There are several ways to send an experimental payload to high altitudes, with the major platforms being weather balloons and sounding rockets. Weather balloons, or scientific balloons, do not require thrust, but rely on buoyancy effects to lift payloads. These would be relatively cheap, although sourcing Helium in the current apocalyptic circumstances may prove to be a problem. Even disregarding the Helium issue, scientific balloons can only reach a maximum altitude of 120,000 ft, or about 37 km (Zell, 2017). This is incredibly shy of the target altitude for the payload of 100km. The other option is a sounding rocket. The use of rockets for research purposes has been employed for several decades now. The ARCAS rocket, for example, was developed in the 1960s (Oss, 1966). Sounding rockets are generally solid rocket motors with a research payload. They are "cost effective and time efficient" (Marconi, 2007), and are well suited for the needs of the Toads and our clients. Using our rocket launch system, the BFT, the payload can be lifted safely to the desired altitude of 100km, enabling the probe to scan for nearby living humans and laying the groundwork for nationwide human cooperation.

2 Needs and Requirements Analysis

In this section, we identify the various stakeholders that have interest in the project. We also address the needs and requirements that we deemed necessary to create a platform that can successfully carry out the mission discussed in the project deliverable.

2.1 Stakeholders

Our stakeholders are the people/organizations that are funding the project, lending us launch space and materials to successfully complete the mission.

For this scenario, the stakeholders that we identified are the only surviving humans left in our area.

- **Our Class:** Our class as a group is trying to find other survivors to band together with.
- **Our Team:** We are designing a system to get chosen so that we do not have to put our lives on the line to go scout for survivors.
- **Remaining Survivors:** Without our successful completion of this project, unknown amounts of surviving people could be lost in the apocalypse.
- **Purdue University/Zucrow Lab:** We are using Purdue's facilities and resources as well as Zucrow Lab's launching amenities.

2.2 Needs

1. A launch system capable of putting a thermal imaging probe at the Karman line
2. The system will function properly on the first launch
3. The system will be able to launch at the launching facility built at Purdue
4. The system can be built quickly and easily
5. The system makes use of multiple stages to maximize efficiency
6. The system makes use of a safe fuel system
7. The system is relatively simple in design
8. The system uses modern composites to reduce the inert mass fraction
9. The system can carry the payload safely

2.3 Requirements

1. The system can achieve a height of at least 100km above sea-level to put a satellite at the height of the Karman Line to cover a large area for thermal imaging.
2. The system has more than one stage to maximize efficiency and have a lower margin of error.
3. The launch system can lift a payload weighing at least 5 kilograms, the mass of the probe.
4. The launch system can lift a payload occupying a volume of at least 20 centimeters cubed, the volume of the probe.
5. The system can produce a thrust to weight ratio of 1.4 to ensure the rocket is powerful enough to lift off.
6. The system is within 1 km of the desired height of 100km to prove an accurate launch.
7. Fuel is class c hazardous substance as rated by NASA, to ensure the rocket is as safe and consistent as possible.
8. The system has a max G force of 3 to ensure the payload does not crumple.
9. The system has a drag coefficient less than .4 to ensure efficient travel through the lower stages of the atmosphere.
10. The system weighs no more than 500 kg to ensure the rocket is efficient due to the limited materials in the situation.
11. The system has an inert mass fraction that is less than .35 so it can reach 100 km, its destination.
12. The system splits the propellant mass of each stage to maximize efficiency
13. The rockets fuel cylinder encasing must have a radius of .25 meters to fit underneath the nose cone for aerodynamic efficiency.

2.4 Preliminary Risk Analysis

Risk	**Justification**	**Likelihood (1 Low 100 high)**	**Severity (1 negligible 100 Catastrophic)**	**Mitigation**
1st Stage does not detach	Decoupling System Malfunctions	10	80, The rocket would not be able to reach 100km carrying extra weight	Inspect Decoupling System to ensure it is fully functional
1st or 2nd Stage detaches incorrectly	Decoupling System Malfunctions	10	85, The rocket can be knocked off course	Inspect Decoupling System to ensure it is fully functional
Fuel ignites in an unintended way	Spark Oxidation Corrosion Improper handling	5	100, The rocket explodes destroying the probe	Handle fuel with caution and make sure tanks and igniters are functional
Rocket Falls over on Launch Pad or launches at a poor angle	Launch site error	1	50, rocket does not launch or launches at too large of an angle to reach 100 km	Ensure the rocket is secured and positioned upright

Experiences abnormal atmospheric conditions	High atmospheric pressure Severe weather	20	50, systems could be disrupted, or the flight path could be altered	Observe atmospheric behaviors and weather before launch
Probe breaks during launch	Too much acceleration/vibration Lightning	25	75, No or subpar images are taken	Add padding around the probe that can also protect against electric surges
Structure Breaks	All the forces crack the body Faulty materials Used	10	100, Breaks rocket in half causing an explosion	Inspect the structural pieces of the rocket to ensure quality

3 Concept Generation, Selection, and Development

This section entails the specifics of how we went through the design project. From our initial ideas to our final solution with all the calculations to prove that it will fulfill the mission's requirements.

3.1 Concept Generation and Selection

This section expands on our decision process and how we started with three different propulsion systems and used an expansive decision matrix to narrow the choices down to the one that best suited our requirements.

3.1.1 High-Level Concept Generation

Our three propulsion concepts that we decided to create the decision matrix for were multi-stage solid, liquid, and hybrid fuel systems. The main reason as to why we decided on these three systems is because in the project deliverable, it requested a multistage rocket. Also, our needs state that the system, "can be built quickly, makes use of a safe fuel system, and is relatively simple in its design (2.2)." Historical data itself has proven that solid fuel, liquid fuel, or a combination of both are the most simplistic and reliable types of rocket propulsion that are readily available. Other fuel types like electric propulsion or nuclear propulsion are either too complex or produce very low thrust, which does not satisfy the mission requirements.

3.1.2 Decision Matrix

The Decision Matrix (Figure C-5) weighs the three High-Level concepts in as numerical a fashion could be afforded. The "Inexpensive" criteria are ignored, since no consistent pricing information could be gathered on rocket motors suitable for our mission. However, it was decided to show that this was a consideration.

When determining the weights of the criteria, it was decided that the success of the mission should outweigh criteria such as "Easy to Build" and "Inexpensive," since it was determined that our clients would prioritize results in the case of a zombie apocalypse. Therefore, "Reaches Karman Line," "Does not Rapidly Disassemble Self," and "Does Not Damage Probe"

were weighted more heavily (Figure C-5-2). The benchmarks (Figure C-5-3) were made to be as general as possible as to not suggest a solution in the decision-making process. Furthermore, the preliminary risk analysis took into account the possible obstacles the launch vehicle might face and the likelihood of occurrence of these obstacles to account for all possibilities.

Most information was determined through historical data for sounding rockets, since they most closely resemble the mission: sending small payloads into a suborbital trajectory. Statistics for solid motors came from the ARCAS sounding rocket (Oss, 1966), statistics for the solid-liquid motor combinations is based on the Aerobee 100 sounding rocket (Rogers, 1997). The Aerobee uses a solid first stage and a liquid upper stage. The ARCAS was used as a reference since it almost exactly fulfills the mission, and the Aerobee 100 is one of few liquid sounding rockets in our size range.

The ARCAS rocket comes in 1 and 2 stage variants, has a specific impulse of 256 seconds, has a thrust to weight ratio of 4.75 at launch, has an empty weight fraction of 0.371, used Ammonium Perchlorate + Aluminum grain fuel, and required a tube to launch from.

The Aerobee rocket has 2 stages. The liquid stage (upper stage) has a specific impulse of 200 seconds, has a thrust to weight ratio of 1.8 at ignition, has an empty weight fraction of 0.33, used furfuryl alcohol and aniline as fuel, as well as red fuming nitric acid as an oxidizer, and launched from a launch pad with a launch tower. More current numbers for the specific impulse of liquid fuels were used, which tend to be closer to 350 seconds.

The total Aerobee rocket has a specific impulse of 198 seconds, has a thrust to weight ratio of 12.5 at launch (for roughly 2.5 second from ignition before the first stage was ejected), and had a total empty weight fraction of 0.133.

3.1.3 Concept Selection

Based on the criteria laid out and the corresponding weights assigned to them, the resulting scores for solid motors, liquid motors, and combinations were 1460, 1310, and 1310, respectively. The main discerning factor was ease of construction and tendency of fuel to not

disassemble the vehicle. Hypergolic propellants tend to be used in sounding rockets, since they are simpler than cryogenic fuels and require less infrastructure to use. However, they also tend to be difficult to handle, since leaks can cause ignition on contact, thus destroying the rocket. Furthermore, solid motors satisfied the criterion for simplicity of construction and did not require complex burn-control mechanisms. This suited our requirements as the mission does not require orbit injection or transfers. It also did not require additional pumps/equipment before launch, further increasing the simplicity of the launch vehicle and reducing risk of failure.

Based on these scores, a multi-stage solid propellant rocket was considered the best choice for this mission. It offered safety and simplicity, while ensuring that costs remained low and all mission parameters were met.

3.2 Detailed Concept Development

The Toads chose to use 2 stages since similar rockets, such as the Aerobee and ARCAS, use 1-2 stages. The project deliverable mandates a multistage rocket, thus the Toads decided on a 2-stage rocket.

3.2.1 Simulation Concept Introduction

To assist in the design of the 2-stage solid rocket, a computer-simulation was used. This allowed the Toads to rapidly test the effects of changing parameters such as C_D, fuel performance characteristics, thrust-to-weight ratio, inert mass fraction, and most prominently propellant mass combinations between the 2 stages.

3.2.2 Staging

One of our requirements was to create a multi-stage rocket for this task. It was decided that with the relatively low required ΔV for just reaching the Karman Line and not putting the satellite into orbit as well as the payload mass only being 5 kg, three stages are not needed. Two stages can efficiently perform this task with less complexity than three stages. Along with that, according to table C-3, three stages will have a higher Gross Lift-Off Weight than two stages until ΔV becomes large enough, and with our ΔV being around 4.6km/s, this has

not occurred yet, causing two stages to have a lower weight and be more efficient for our task.

Furthermore, having multiple stages increases the max ΔV the rocket can attain with similar propellant weight. This occurs because when the first stage drops off, the rocket is still travelling at a certain velocity, which is increased by second stage ignition. Thus, the rocket needs less fuel to travel faster. This can actually be seen in the graph for Thrust produced vs. Time (Appendix B-1), where the thrust momentarily drops to zero when the first stage burns out. The thrust subsequently increases as the second stage ignites, increasing the rocket's velocity.

The stages separate after complete fuel burn. There are two enclosures to hold the separation/ejection sections between the first and second stages and between the second stage and the payload bay, each measuring 0.1m in height. These enclosures each house a hot-wired electro-explosive device, also known as an EED, which uses an electric circuit encapsulated by an explosive. Once the circuit is activated, the explosive heats up to certain temperature, leading to an exothermic reaction that leads to stage separation. This technology has been applied to ejection seats and separation mechanisms with an extremely low rate of failure. The first EED would be primed to detonate once the acceleration of the rocket dips below $0\ m/s^2$ for the first time, which leads to first-stage separation. After detonation, there is a 2 second pause to ensure that the separated stage has 'fallen' far enough for the second stage to ignite safely. Once the second stage burns out completely, the acceleration of the rocket falls below 0 for the second time, leading to the detonation of the second EED, which leads to payload deployment above 100 km. Each EED is around 0.1m tall, and is tightly compressed against both stages to ensure a safe and successful stage-separation and payload deployment.

3.2.3 Atmospheric Model

To calculate the density and pressure of air at a specific altitude, we used the code that was written for Homework 1. According to Figure A-6, the function that performs the calculations, the code is split in to three if statements. The first if statement calculates the density and pressure of the standard atmosphere from sea-level to 36,000 meters using the

altitude as a function of temperature slope found in Figure C-2. The else-if statement calculates the same variables at an altitude of 36,000 – 82,000 meters using the same set up and equations. Finally, the else statement calculates the desired outputs at a range of altitudes from 82,000 to roughly 100,000 meters. This final range is an estimate because Figure C-2 does not give a slope for an altitude higher than 105,000 meters. Therefore, we assumed that the relationship would continue to be linear as altitude increases past 100,000 meters. The equations that we used to derive the pressure and thrust were found in the AAE 251 textbook (Anderson, 2016).

3.2.4 Fuel Selection

From the solid fuels to choose from, ammonium perchlorate as the oxidizer and powdered aluminum was chosen as fuel, with polybutadiene acrylonitrile prepolymer (PBAN) as the binder for the mixture, which is known as *Aluminum Perchlorate Composite Propellant* (APCP). These components are mixed in a ratio of roughly 70% oxidizer, 15% fuel (Aluminum powder), and 15% PBAN/HTPB/binder by mass (Sutton, 2017). This is the fuel used on the ARCAS rocket, the first stage of the Aerobee rocket, and the main boosters on the STS (Dumoulin, 1988). In addition to being a standard fuel for solid rockets, the fuel is stable and does not spontaneously detonate (Orr). The fuel fulfills our requirements: It can achieve the required height (3.2.8), has enough thrust to lift the payload (3.2.8), is not considered hazardous (NASA, 1977), is reputably consistent. The fuel delivers an estimated *Specific impulse* of 242 seconds at sea-level (under perfect conditions for the Space Shuttle Solid Rocket Boosters), while having a specific heat value of $1460 \frac{J}{kg}.K$ (Yang, Thakre, & Cai, 2008), compared to $2600 \frac{J}{kg}.K$ for Kerosene, thus being able to increase temperature at a faster rate. Furthermore, APCP has an elastic, rubbery texture, which means that the fuel is cast into shape instead of being pressed like other solid propellants, minimizing irregularities during the manufacturing phase, easier handling, and better replicability (Yang, Thakre, & Cai, 2008).

3.2.5 Fuel Performance Calculation

In order to calculate the performance of the chosen APAL grain rocket fuel, using historical data or estimates from existing systems was decided to be too inaccurate to create a meaningful simulation. Instead, a software tool called Chemical Equilibrium Analysis (CEA) developed by NASA was used to calculate generalized performance parameters for the combustion of the fuel (Snyder, 2016). Obtaining these parameters for the combustion of the fuel allowed for the critical parameters affecting the flight such as mass flow rate and thrust produced to be customized exactly to the needs of each stage while ensuring that the performance remained meaningful and within the bounds of what the fuel could do. For example, when simulating the performance of an arbitrary mass of propellant in a stage, the throat area of the nozzle was changed to be the right size to produce the thrust needed for that amount of propellant. A more detailed discussion will follow.

First the proper inputs to the CEA program needed to be established. For CEA to run a rocket performance calculation, it needs a description of all the reactants and a relative composition by mass of each reactant in the grain, as well as chamber pressure and expansion ratios of the nozzle. Chamber pressure was taken from the NASA ARCAS design (Oss, 1966), shown in Appendix C-7 as 975 psi. The exact reactant combinations were selected to be 68.8% Ammonium Perchlorate oxidizer, 0.2% Iron Oxide oxidizer, 20% Aluminum fuel, and 10% PBAN/HTPB binder/fuel, with these numbers being based on commonly used mixtures taken from the journal article “Analytical and Experimental Comparisons of HTPB and ABS as Hybrid Rocket Fuels” (Whitmore, Peterson, & Eilers, 2011). Finally, a range of subsonic and supersonic expansion ratios were inputted (5, 10, 20, 50, 100) in order to produce a range of outputs to be chosen from for optimization. The full CEA input file can be seen in Appendix C-8.

Upon running CEA with this input, the output produced looks like Appendix C-9, although the file shown in C-9 is heavily abbreviated to only the output relevant to these calculations. The first section of the abbreviated output shows state information ranging from pressure to sonic velocity and Mach number displayed for the states at chamber, throat, and multiple exit conditions. Each state corresponds to a different expansion ratio. For our purposes, the exit condition at the far right of the table was selected since they gave the most meaningful

results, such as a specific impulse of 257 seconds, which is close to the specific impulse provided by the fuel used in the SRBs for the Space Shuttle (NASA, 2000). The second section named *Performance Parameters* gives the critical information we need to calculate thrust and mass flow rate, including the parameters characteristic velocity c^*, thrust coefficient c_f, expansion ratio A_e/A_t, and specific impulse *ISP*. These values can be seen copied into the code as constants in the fuel parameters calculation function in Appendix A-4.

In order to calculate throat area, the formula

$$F = c_f \cdot P_C \cdot A_t$$

is used where c_f is thrust coefficient, P_C is chamber pressure, and A_t is throat area. In the algorithm, thrust is given and throat area is calculated from thrust. As the propellant mass is being optimized, new guesses for propellant masses are converted into stage weights using the formulae for f_{inert} and gravity, and desired thrust is calculated from the resultant weight and the desired thrust-to-weight ratio of 1.4. Adding the constants c_f and P_C, a value for throat area is obtained and used for the next step. The throat area is also converted to exit area using the A_e/A_t equal to 5. This is implemented on lines 18 and 19 of Appendix A-4.

Next, the mass flow rate is calculated with the formula

$$c^* = \frac{P_C \cdot A_t}{\dot{m}}$$

where c^* is characteristic velocity and $\dot{m}$ is the mass flow rate. The mass flow rate is used to calculate the differential amount of mass *dm* consumed and expelled and each time interval *dt* in the simulation. This is implemented on line 21 of Appendix A-4.

Finally, the thrust is calculated at each interval using the formula

$$F = \dot{m} \cdot V_e + A_e \cdot (P_e - P_\infty)$$

where V_e is exit velocity, A_e is nozzle exit area, P_e is nozzle exit pressure, and P_∞ is the current ambient pressure.

The nozzle chosen is a ring-shaped nozzle illustrated as the 2^{nd} diagram in App. C-6. This nozzle was chosen because of its consistent fuel burn geometry for solid fuels and the ease and accuracy with which it can be manufactured.

3.2.6 Section Mass Estimates

In 1966, NASA presented an ARCAS Meteorological Rocket designed to take a 10.5 lb. payload to an altitude of at least 60 km. The Toads decided to estimate our f_{inert} based off this ARCAS rocket for many reasons. First off, the Toads have a remarkably similar payload mass, with our satellite weighing 5 kg. To go along with that, NASA was launching to an altitude of 60 km with a single stage rocket, while the rockets needs to reach at least 100 km and are using two stages. According to NASA's Table II (C-1), their inert mass was 25.40 lbs., and their weight at lift-off without the payload was 68.40 lbs., which is their inert mass and propellant mass combined.

$$f_{inert} = \frac{m_{inert}}{m_{inert} + m_{prop}}$$

$$f_{inert} = \frac{25.40\ lbs}{68.40\ lbs}$$

$$f_{inert} = 0.371$$

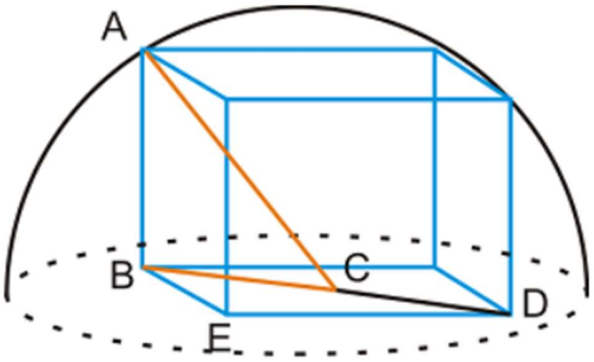
Using this data, we calculated the f_{inert} of this ARCAS rocket to be 0.371. We are estimating our f_{inert} to be 0.3 due to our use of newer, lightweight composites that can handle the same stress with a smaller mass. For example, NASA used fiberglass for their motor case which is the largest portion of the inert mass of their rocket, while we will use carbon fiber (Oss, 1966). This is the inert mass fraction that we used simulate the flight of the launch vehicle and calculate design specifications. The mass of the actual engine is considered to be negligible when compared to the inert mass.

3.2.7 Drag Calculation

$$D = \frac{1}{2} * \rho * v^2 * s * Cd$$

Drag is used to calculate the net force on the rocket. Velocity and Air density are calculated through our code and updated at every iteration. The surface area of the rocket was decided on multiple factors of the rocket. First on NASA's chart with the coefficient of drag for different shapes and team R16 decided the bullet design would be the most effective based

on Aerospace Research Center's data of the coefficient of drag for a hemispherical blunt nose cone being 0.1789 (Nichols & Nierengarten, 1964). The shape of the nose cone is exactly the shape of the rocket's second stage with a hemispherical nose cone on top of a cylindrical stage 2. This stage then tapers off into stage 1 off the rocket, which is thinner to conserve space while allowing the nozzle to be narrow enough to sufficiently expand exhaust flow. Thus, the approximation of the coefficient of drag is reasonable to apply to the rocket. The image below will help show the maximum size cube that can fit in a hemisphere, this will be the calculation used for the probe fitting in the rocket's nosecone.



Given, the radius of the hemi sphere AC = a. Let the side of the cube is x.
From the above diagram,

$$BE^2 + ED^2 = BD^2$$

$$x^2 + x^2 = BD^2$$

$$BD = \sqrt{2} * x$$

$$\mathbf{BC = x/\sqrt{2}}$$

$$AC^2 = AB^2 + BC^2$$

$$a^2 = 3x^2/2$$

$$a = \sqrt{3/2} * x$$

From the probe's dimensions x, the side length of the cube, = .2m.

$$a = \sqrt{3/2} * .20 = .2449\ m$$

The nose cone based on past rocket launch thickness is approximately .05 m for the material of the composite to be strong enough along with the padding necessary to keep the probe safe.

Cross sectional area is a = π* r^2

$$a = \pi * .2949^2 = .2732\ m^2$$

With all these values calculated, the force of drag on the rocket can be accurately calculated.

3.2.8 Simulation Function Integration

Discuss how thrust, drag, weight and atmospheric conditions are calculated and applied to change the state of the rocket at each calculation interval

The MATLAB simulation goes through a loop using a timestep that varies depending on the accuracy of the results desired verse the speed that the code runs. Through multiple iteration the code described below is a simulation of The Big Flying Toad rocket launch based on many of the major factors that affect a launch.

1. The differential quantity of mass is found by using the NASA CEA software which dynamically calculates the differential mass, based on mass flow rate, chamber pressure, and cstar. (App. A-5)
2. Air density and air pressure at different heights are found from the code used in HW1 which takes in altitude and finds the corresponding density and pressure. (App. A-6)
3. The thrust of the rocket is calculated from the mass flow rate, exit velocity, nozzle area and pressure at different heights. (App. A-7)
4. The weight of the rocket is then found which changed every iteration due to the loss of rocket fuel. (App. A-8)
5. Drag is calculated by the velocity and air density which update every loop along with the constant values of the drag coefficient and surface area. (App. A-9)
6. The net force of the rocket is calculated with the drag, thrust, and weight forces. (App. A-10)

7. Acceleration is the last variable found using the net force found in step 6 and dividing by the current mass of the rocket. This acceleration is used to calculate the velocity and the height of the rocket. (App. A-11, A-12, and A-13)
8. The code then updates and prepares for the next iteration until the first stage is completely burned out.
9. The separation phase of the launch occurs after the first stage has expelled all its propellant. The phase lasts two seconds. The weight is constant and thrust is equal to zero, so the rocket is coasting.
10. The second step is when the decoupling system activates release the first stage rocket. There is no thrust and the weight change that occurs is the inert mass of the first stage.
11. The second stage begins burn when the separation is complete and is the same code of the first stage, steps 1-8, with the new values for the second stage.
12. The second stage separates from the payload in the same way as the first stage, step 9-10.
13. The coasting stage is the final part of launch. This is when the payload continues gaining altitude with no thrust, until finally the velocity equals zero due to gravity and the payload reaches the maximum height of 100 km.
14. Then the code plots graphs to visually show the launch.

In conclusion, the code calculates all the reactions the rocket goes through during the launch which will tell us the amount of fuel necessary to carry the probe to 100 km, thus sizing The Big Flying Toad.

3.2.9 Two-Stage Mass Split Optimization

Discuss design of function that finds the most efficient split of mass for stage 1 and 2 which reaches our target height

The Two-Stage Mass Split Optimization function (App. A-2) uses historical data to guess the highest and lowest initial masses of the two stages. The mass is then incremented by a value based on required accuracy. The step gets smaller with increasing accuracy. Then, for each mass, the flight path of the rocket is simulated and the mass of stage 1, the mass of stage 2, and the maximum height reached are stored as part of a single variable named '*Results*'.

Vector indexing is then used to find the rows with the maximum height reached being either more than or equal to the target height, which is 100 km. Then, the initial masses of the two stages in each individual rows are added to find all possible values for the total initial mass of the rocket. These values are sorted in ascending order to find the lowest possible initial mass of the rocket. Thus, this iterative algorithm allows us to run through all possible values of the initial mass of the launch vehicle (given a fairly small iteration step) even after changing values for the other design parameters in an instant.

3.3 Rocket Design & Specifications

2-3 introductory sentences for this section

In this section, we clarify the exact values of specific launch variables.

3.3.1 Specification Table

PROPERTY	VALUE
TOTAL INITIAL MASS	392.4 kg
EMPTY MASS	121.23 kg
TOTAL ΔV	4,767.8 m/s
CHAMBER PRESSURE	975 psi
CHARACTERISTIC VELOCITY	1,695.5 m/s
THRUST COEFFICIENT	1.4851
EXPANSION RATIO	5
SPECIFIC IMPULSE	256.7 s
FIRST STAGE	
PROPELLANT MASS	221.75 kg
INERT MASS	95.04 kg
TOTAL MASS	316.79 kg

VACUUM THRUST	5,961.6 N
INERT MASS FRACTION	0.3
THRUST-WEIGHT RATIO	1.4
BURN TIME	103.7 s
ΔV	2,096.2 m/s
NOZZLE THROAT AREA	5.393 cm^2
NOZZLE EXIT AREA	27.0 cm^2
SECOND STAGE	
PROPELLANT MASS	49.5 kg
INERT MASS	21.2 kg
TOTAL MASS	70.7 kg
VACUUM THRUST	1,138.7 N
INERT MASS FRACTION	0.3
THRUST-WEIGHT RATIO	1.4
BURN TIME	121.1 s
ΔV	2,671.6 m/s
NOZZLE THROAT AREA	0.9620 cm^2
NOZZLE EXIT AREA	4.810 cm^2

3.3.2 Flight Characteristics

The values in this table are found using the plots in Appendix B.

Property	Time of Occurrence	Value
Max Acceleration	226.8 s	17.15 m/s^2

Max Velocity	226.8 s	984.3 m/s
Max Thrust	103.6 s	5,884 N
Max Drag	103.6 s	2,715 N

The Thrust graph has two different almost constant burns. The first burn slowly increases due to the pressure at the exit of the nozzle decreasing as altitude increases. The 2nd burn is constant as atmosphere as all but diminished at the higher altitudes. The max thrust being at the end of the first stage about 100 seconds into flight.

The drag graph increases as the rocket flies during the first stage as it is low in the atmosphere, so the air density is high and the rocket gains velocity also increasing the drag. The drag maxes out at the end of the first stage about 100 seconds into launch. During the second stage the drag is constant as density is decreasing proportionally to the square of the velocity increasing and is much lower than during the first stage as the atmosphere is extremely thin at the high altitudes.

The velocity graph increases linearly for both stages. After the separation of the first stage the velocity continues to decrease even after the 2nd stage ignites, until the thrust surpasses the weight and drag forces. The max velocity is reached at the end of the second stage when the last of the propellant is expelled. The rocket then coasts until velocity equals zero.

The acceleration graph has two different slowly increasing portions, one for each stage. This is due to less drag and weight on the rocket as each stage continues the burn. The max acceleration occurs at the end of the second rocket burn when the rocket is the lightest and the drag the smallest. Then when the rocket is coasting gravity is the constant 9.81 of Earth.

3.4 Rocket Physical Sizing

The nozzle is roughly shaped based on the B-200 rocket motor. This motor has a ratio from nozzle diameter to height of roughly 2.5, which was used to determine the heights of the nozzles (Nakka, 2002). The density of the fuel is approximately 1500kg per cubic meter (Amir Aziz, 2015).

Lower Stage Nozzle: $\sqrt{\frac{27.0cm^2}{\pi}} * 2 * 2.5 = 14.66cm$

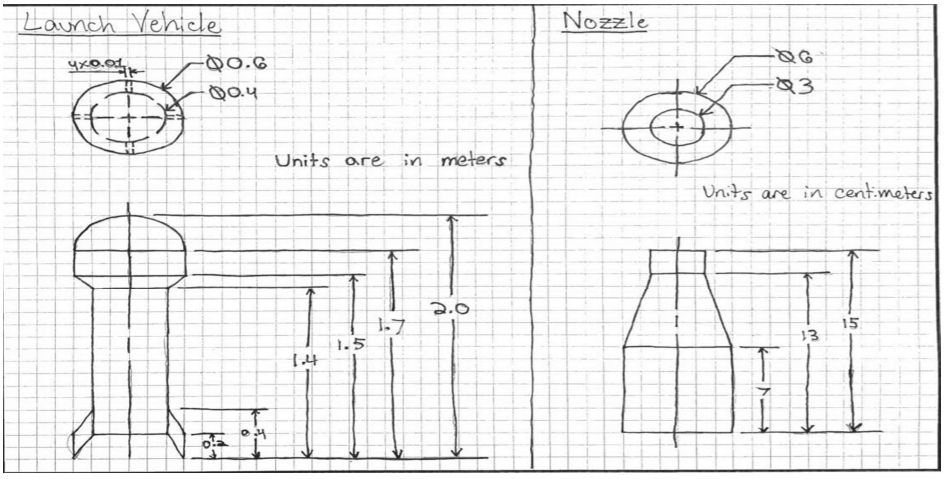
Upper Stage Nozzle: $\sqrt{\frac{4.81cm^2}{\pi}} * 2 * 2.5 = 6.18cm$

Staging Mechanism: 7cm (CITE THIS SOURCE)

Lower Fuel Tank Height: $\frac{221.75kg}{\left(1500^{kg}/_{m^3}\right) * \pi * (0.2m)^2} = 1.176m$

Upper Fuel Tank Height: $\frac{49.5kg}{\left(1500^{kg}/_{m^3}\right) * \pi * (0.2949m)^2} = 0.121m$

Nose Cone Height: 0.2949m



3.5 Infrastructure Requirements

The goal for the infrastructure requirements was to be as simple and easy to construct as possible. Due to the choice of a solid rocket motor, the infrastructure required is minimal; no

launch umbilical is required, and no loading infrastructure is needed. All fuel is pre-loaded into the launch vehicle before launch countdown and no external pumps/cooling systems are required before launch. The only requirement for launch infrastructure is that it aims the rocket vertically and can light the first stage solid motor. To accomplish this, a launch system similar to the ARCAS rocket's was devised (Oss, 1966). A tube launcher that is the width of the rocket and slots for the fins will hold the rocket in place. Additionally, it will have an electro-explosive device, or EED (Douglass, 1971). This will require an electric potential of between 500V and 2500V. EEDs provide a "highly reproducible ignition" for stage separation and payload deployment.

4 Conclusions

In this section, we discuss how our final design will perform, steps we might have taken in order to further enhance the project, and some of the lessons that we learned along the way.

4.1 Design Evaluation

We believe that our final concept will successfully complete the mission. This is because the simulation that we have created takes into account all of the important aspects of the rocket launch. All of the important launch variables are calculated with a really low time step in order to have extreme accuracy. The risks that have been considered cover almost every situation in which the rocket can fail. These risks can mostly be mitigated with the careful inspection of every aspect of the launch vehicle. As with all rockets though, there are safety hazards due to the materials being used, and with that a chance for catastrophe. However, the simplicity of our design decreases these risks noticeably. Therefore, we believe that the final design has a high chance of completing the mission at hand.

4.2 Next Steps

If the Toads were to continue working on this project for an additional semester, the Toads would refine our simulation to account for additional variables, such as real weather data, subsequent changes in direction, and effects of the fins on performance. This would add a new layer of complexity to the project, but also take it one step further to being a real

solution. We would also rely less on historical data and create or model parts ourselves using CAD software, such as CATIA or Solidworks. Furthermore, the design of the launch vehicle could be improved to insert a payload into LEO (Low Earth Orbit) or for orbit transfer injections by adding an additional stage. Given additional time, we could conduct more research into liquid/hybrid fuels to use and design the launch vehicle for possible reusability and re-landing launch. We could also look into how varying the chamber pressure inside the engine affects exhaust expansion, leading to more efficient thrust production.

4.3 Lessons Learned

This project was no easy feat, but nevertheless it brought technical and professional growth for the Toads. From a technical side, The Toads learned a great deal about rocket design, specifically about using existing tools to fill in the blanks for when information on a rocket is incomplete. When developing the thrust function for the simulation, several important parameters were missing from the Toads' main source for historical information: the ARCAS rocket technical documentation. Initially, the toads used a long string of equations found online to relate the known information and find specific impulse and exit pressure. The Toads later discovered a NASA tool that provides more information with much more reliability. The Toad's learned the importance of using available resources to solve problems when possible, and that developing aerospace hardware is an iterative process that does not necessitate reinventing the wheel.

The most significant technical revelation came with the choice to program a holistic simulation for the rocket flight to quickly optimize rocket parameters, such as starting mass and final height. Initially, calculations were done by hand, thus promoting heavy use of assumptions. Using the simulation allowed the Toad's to effectively justify design decisions and make changes to the design quickly without the need for extended periods of crunching numbers.

More important than the technical lessons learned are the professional lessons. This project is not straight forward and requires several minds working together to complete; No single Toad could have accomplished this task. The Toads had to become accustomed to delegation, as well as collaborating with sources outside of the group, such as TA's and professors.

Embracing a culture of collaboration lowered the individual stresses on the Toads and improved collective productivity.

Bibliography

Amir Aziz, R. M. (2015). REVIEW ON TYPICAL INGREDIENTS FOR AMMONIUM PERCHLORATE BASED SOLID PROPELLANT . *ARPN Journal of Engineering and Applied Sciences* .

Anderson, J. D. (2016). *Introduction to Flight Eighth Edition.* New York: McGraw-Hill Education.

Ashish Narayan, S. N. (2019, July). *arc.aiaa.org*. Retrieved from Control of Aerodynamic Drag and Heating of Nosecones Through Taper Spikes: https://arc.aiaa.org/doi/pdf/10.2514/1.A34250

Campusgate. (n.d.). Retrieved from Mensuration II: https://www.campusgate.co.in/2012/10/mensuration-ii-important-results.html

Douglass, H. W. (1971). *Solid Rocket Motor Igniters.* Lewis Research Center.

Dumoulin, J. (1988). Solid Rocket Boosters. KSC: NASA.

Marconi, E. M. (2007, November 22). What is a Sounding Rocket? Kennedy Space Center, Florida, United States of America.

McBride, B. J., & Gordon, S. (1996). *Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications.* Cleveland, Ohio: National Aeronautics and Space Administration.

Mora, N. (2012). On the Electromagnetic Susceptibility of Hot Wire-Based Electroexplosive Devices to RF Sources. *IEEE Transactions on Electromagnetic Compatibility*, 1-10.

Nakka, R. (2002, March 24). *B-200 Motor.* Retrieved from Nakka Rocketry: https://www.nakka-rocketry.net/engine1.html

NASA. (1977, May). *NASA.gov*. Retrieved from Rocket Propulsion Hazard Summary: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19770018296.pdf

NASA. (2000, August 31). *Solid Rocket Boosters*. Retrieved from https://science.ksc.nasa.gov/shuttle/technology/sts-newsref/srb.html

NASA. (n.d.). *NASA.gov*. Retrieved from Space Shuttle Ascent: https://www.nasa.gov/pdf/466711main_AP_ST_ShuttleAscent.pdf

Nichols, J. O., & Nierengarten, E. A. (1964). *Aerodunamic Characteristics of Blunt Bodies.* Pasadena: Jet Propulsion Laboratory.

Orr, G. (n.d.). *ESTIMATION AND ANALYSIS OF QUASI-1D SOLID ROCKET MOTOR.* Retrieved from Harvey Mudd College Archive: http://www.eng.hmc.edu/NewE80/PDFs/SolidPropulsionGO.pdf

Oss, G. K. (1966). *Development of the Frangible ARCAS Meteorological Rocket Vehicle.* Alexandria, Virginia: Atlantic Research Corporation.

Rogers, C. E. (1997, July). The Aerobee 100, 150, and 300 Series Sounding Rocket. *Tech Series*.

Snyder, C. A. (2016). *Chemical Equilibrium with Applications - Glenn Research Center*. Retrieved November 25, 2019, from https://www.grc.nasa.gov/WWW/CEAWeb/

Sprik, D. R. (2010). Solid propellant grain geometry design, a model for the evolution of star shaped interfaces. *Universteit Van Amsterdam*.

Sutton, G. P. (2017). *Rocket Propulsion Elements.*

Whitmore, S. A., Peterson, Z. W., & Eilers, S. D. (2011). Analytical and Experimental Comparisons of HTPB and ABS as Hybrid Rocket Fuels. *AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit.* Logan, Utah: American Institute of Aeronautics and Astronautics.

Yang, V., Thakre, P., & Cai, W. (2008). A Model of AP/HTPB Composite Propellant Combustion in Rocket-Motor Environments. *Combust. Sci. and Tech.* .

Zell, H. (2017, August 3). Scientific Balloons. Kennedy Space Center, Florida, United States of America.

Appendix A: MATLAB Code

A-1 Master Function

```
%function [h] = Simulate(mProp1, mProp2)
%% Constants
dt = 0.001;
hLaunchPad = 181; % initial height in meters
finert = 0.3; % inert mass estimate
tToW = 1.4; % Thrust to weight ratio of each stage
mPay = 5; % payload mass, kg

storeData = true;
mProp1 = 221.75; % mass of stage 1 propellant
mProp2 = 49.45; % mass of stage 2 propellant

stage1Mass = getstagemass(finert, mProp1);
stage2Mass = getstagemass(finert, mProp2);
mInitial = stage1Mass + stage2Mass + mPay;

% Get the initial masses at the beginning of each stage burn
stage1InitialMass = mInitial;
stage2InitialMass = stage2Mass + mPay;

m = mInitial; % initialize mass to mInitial
h = hLaunchPad; % initialize height to launch pad
v = 0; % initialize rocket velocity to zero

time = 0;

if storeData
    % Preallocate Storage Variables
    % use mdot and dt to estimate size
    mdot1 = getfuelparams(stage1InitialMass, hLaunchPad, tToW);
    mdot2 = getfuelparams(stage2InitialMass, 6e4, tToW);
    burnTime1 = mProp1 / mdot1;
    burnTime2Estimate = mProp2 / mdot2;
    arraySize = round((burnTime1 + burnTime2Estimate) / dt);
```

```
35
36 -     thrustArray = zeros(arraySize, 2);
37 -     weightArray = zeros(arraySize, 2);
38 -     dragArray = zeros(arraySize, 2);
39 -     accelArray = zeros(arraySize, 2);
40 -     velocityArray = zeros(arraySize, 2);
41 -     heightArray = zeros(arraySize, 2);
42 -     massArray = zeros(arraySize, 2);
43
44 -     idx = 1;
45 -   end
46
47    % Simulate the portion of the flight for stage 1
48 - while m > mInitial - mProp1
49        % Calculate Current State
50 -     dm = getdm(dt, stage1InitialMass, hLaunchPad, tToW);
51 -     [rho, pAmbient] = getpp(h);
52 -     thrust = getthrust(pAmbient, stage1InitialMass, hLaunchPad, tToW);
53 -     weight = getweight(m, h);
54 -     drag = getdrag(rho, v);
55 -     fnet = getfnet(thrust, weight, drag);
56 -     accel = getaccel(fnet, m);
57
58 -     if storeData
59            % Save Current State
60 -         thrustArray(idx, :) = [time, thrust];
61 -         weightArray(idx, :) = [time, weight];
62 -         dragArray(idx, :) = [time, drag];
63 -         accelArray(idx, :) = [time, accel];
64 -         velocityArray(idx, :) = [time, v];
65 -         heightArray(idx, :) = [time, h];
66 -         massArray(idx, :) = [time, m];
67
68 -         idx = idx + 1;
```

```
    end

    % Prepare Next State
    vnew = getvel(v, accel, dt);
    hnew = gethigh(h, vnew, dt);
    mnew = getmnew(m, dm);

    % Update Next State
    time = time + dt;
    v = vnew;
    h = hnew;
    m = mnew;
end

tEndStage1 = time;

% Pre Stage 1 Separation 1-second Delay
while time < tEndStage1 + 1
    % Calculate Current State
    dm = 0;
    [rho, ~] = getpp(h);
    thrust = 0;
    weight = getweight(m, h);
    drag = getdrag(rho, v);
    fnet = getfnet(thrust, weight, drag);
    accel = getaccel(fnet, m);

    if storeData
        % Save Current State
        thrustArray(idx, :) = [time, thrust];
        weightArray(idx, :) = [time, weight];
        dragArray(idx, :) = [time, drag];
        accelArray(idx, :) = [time, accel];
        velocityArray(idx, :) = [time, v];
```

```
103 -           heightArray(idx, :) = [time, h];
104 -           massArray(idx, :) = [time, m];
105
106 -           idx = idx + 1;
107 -       end
108
109         % Prepare Next State
110 -       vnew = getvel(v, accel, dt);
111 -       hnew = gethigh(h, vnew, dt);
112 -       mnew = getmnew(m, dm);
113
114         % Update Next State
115 -       time = time + dt;
116 -       v = vnew;
117 -       h = hnew;
118 -       m = mnew;
119 -   end
120
121     % Drop Stage 1
122 -   m = mInitial - stage1Mass;
123
124     % Post Stage 1 Separation 1-second Delay
125 -   while time < tEndStage1 + 2
126         % Calculate Current State
127 -       dm = 0;
128 -       [rho, ~] = getpp(h);
129 -       thrust = 0;
130 -       weight = getweight(m, h);
131 -       drag = getdrag(rho, v);
132 -       fnet = getfnet(thrust, weight, drag);
133 -       accel = getaccel(fnet, m);
134
135 -       if storeData
136             % Save Current State
```

```
137 -           thrustArray(idx, :) = [time, thrust];
138 -           weightArray(idx, :) = [time, weight];
139 -           dragArray(idx, :) = [time, drag];
140 -           accelArray(idx, :) = [time, accel];
141 -           velocityArray(idx, :) = [time, v];
142 -           heightArray(idx, :) = [time, h];
143 -           massArray(idx, :) = [time, m];
144
145 -           idx = idx + 1;
146 -       end
147
148         % Prepare Next State
149 -       vnew = getvel(v, accel, dt);
150 -       hnew = gethigh(h, vnew, dt);
151 -       mnew = getmnew(m, dm);
152
153         % Update Next State
154 -       time = time + dt;
155 -       v = vnew;
156 -       h = hnew;
157 -       m = mnew;
158 - end
159
160     % Capture the height before stage 2 starts
161 -   hStartStage2 = h;
162
163     % Simulate the portion of the flight for stage 2
164 - while m > mInitial - stage1Mass - mProp2
165         % Calculate Current State
166 -       dm = getdm(dt, stage2InitialMass, hStartStage2, tToW);
167 -       [rho, pAmbient] = getpp(h);
168 -       thrust = getthrust(pAmbient, stage2InitialMass, hStartStage2, tToW);
169 -       weight = getweight(m, h);
170 -       drag = getdrag(rho, v);
```

```
171 -       fnet = getfnet(thrust, weight, drag);
172 -       accel = getaccel(fnet, m);
173
174 -       if storeData
175             % Save Current State
176 -           thrustArray(idx, :) = [time, thrust];
177 -           weightArray(idx, :) = [time, weight];
178 -           dragArray(idx, :) = [time, drag];
179 -           accelArray(idx, :) = [time, accel];
180 -           velocityArray(idx, :) = [time, v];
181 -           heightArray(idx, :) = [time, h];
182 -           massArray(idx, :) = [time, m];
183
184 -           idx = idx + 1;
185 -       end
186
187         % Prepare Next State
188 -       vnew = getvel(v, accel, dt);
189 -       hnew = gethigh(h, vnew, dt);
190 -       mnew = getmnew(m, dm);
191
192         % Update Next State
193 -       time = time + dt;
194 -       v = vnew;
195 -       h = hnew;
196 -       m = mnew;
197 -   end
198
199 -   tEndStage2 = time;
200
201     % Pre Stage 2 Separation 1-second Delay
202 -   while time < tEndStage2 + 1
203         % Calculate Current State
204 -       dm = 0;
```

```
205 -       [rho, ~] = getpp(h);
206 -       thrust = 0;
207 -       weight = getweight(m, h);
208 -       drag = getdrag(rho, v);
209 -       fnet = getfnet(thrust, weight, drag);
210 -       accel = getaccel(fnet, m);
211
212 -       if storeData
213             % Save Current State
214 -           thrustArray(idx, :) = [time, thrust];
215 -           weightArray(idx, :) = [time, weight];
216 -           dragArray(idx, :) = [time, drag];
217 -           accelArray(idx, :) = [time, accel];
218 -           velocityArray(idx, :) = [time, v];
219 -           heightArray(idx, :) = [time, h];
220 -           massArray(idx, :) = [time, m];
221
222 -           idx = idx + 1;
223 -       end
224
225         % Prepare Next State
226 -       vnew = getvel(v, accel, dt);
227 -       hnew = gethigh(h, vnew, dt);
228 -       mnew = getmnew(m, dm);
229
230         % Update Next State
231 -       time = time + dt;
232 -       v = vnew;
233 -       h = hnew;
234 -       m = mnew;
235 -   end
236
237     % Drop the rest of the weight
238 -   m = mPay;
```

```
239
240     % Coast Phase
241 -   while v > 0
242         % Calculate Current State
243 -       dm = 0;
244 -       [rho, ~] = getpp(h);
245 -       thrust = 0;
246 -       weight = getweight(m, h);
247 -       drag = getdrag(rho, v);
248 -       fnet = getfnet(thrust, weight, drag);
249 -       accel = getaccel(fnet, m);
250
251 -       if storeData
252             % Save Current State
253 -           thrustArray(idx, :) = [time, thrust];
254 -           weightArray(idx, :) = [time, weight];
255 -           dragArray(idx, :) = [time, drag];
256 -           accelArray(idx, :) = [time, accel];
257 -           velocityArray(idx, :) = [time, v];
258 -           heightArray(idx, :) = [time, h];
259 -           massArray(idx, :) = [time, m];
260
261 -           idx = idx + 1;
262 -       end
263
264         % Prepare Next State
265 -       vnew = getvel(v, accel, dt);
266 -       hnew = gethigh(h, vnew, dt);
267 -       mnew = getmnew(m, dm);
268
269         % Update Next State
270 -       time = time + dt;
271 -       v = vnew;
272 -       h = hnew;
```

```
273 -         m = mnew;
274 -     end
275
276       % Plot Results
277
278 -     if storeData
279           % Save Current State
280 -         time = thrustArray(:,1);
281
282 -         figure('Renderer', 'painters', 'Position', [10 10 900 600]);
283 -         plot(time, thrustArray(:,2), 'r', 'LineWidth', 1.5);
284 -         title('Thrust vs Time') %creates a title
285 -         xlabel('Time (s)') %labels x axis
286 -         ylabel('Thrust (N)') %labels y axis
287 -         grid on
288
289 -         figure('Renderer', 'painters', 'Position', [10 10 900 600]);
290 -         plot(time, dragArray(:,2), 'g', 'LineWidth', 1.5);
291 -         title('Drag vs Time') %creates a title
292 -         xlabel('Time (s)') %labels x axis
293 -         ylabel('Drag (N)') %labels y axis
294 -         grid on
295
296 -         figure('Renderer', 'painters', 'Position', [10 10 900 600]);
297 -         plot(time, weightArray(:,2), 'b', 'LineWidth', 1.5);
298 -         title('Weight vs Time') %creates a title
299 -         xlabel('Time (s)') %labels x axis
300 -         ylabel('Weight (N)') %labels y axis
301 -         grid on
302
303 -         figure('Renderer', 'painters', 'Position', [10 10 900 600]);
304 -         plot1 = [thrustArray(:,2), weightArray(:,2), dragArray(:,2)];
305 -         plot(time, plot1, 'LineWidth', 1.5);
306 -         title('Thrust, Drag, and Weight vs Time') %creates a title
```

```
307 -       xlabel('Time (s)') %labels x axis
308 -       ylabel('Newtons') %labels y axis
309 -       grid on
310 -       legend('Thrust', 'Weight', 'Drag', 'Location', "Best");
311
312 -       figure('Renderer', 'painters', 'Position', [100 100 900 600]);
313 -       subplot(2,2,1);
314 -       plot(time, heightArray(:,2), 'b');
315 -       title('Height vs Time') %creates a title
316 -       xlabel('Time (s)') %labels x axis
317 -       ylabel('Height (m)') %labels y axis
318 -       grid on
319
320 -       subplot(2,2,3);
321 -       plot(time, velocityArray(:,2), 'r');
322 -       title('Velocity vs Time') %creates a title
323 -       xlabel('Time (s)') %labels x axis
324 -       ylabel('Velocity (m/s)') %labels y axis
325 -       grid on
326
327 -       subplot(2,2,2);
328 -       plot(time, accelArray(:,2), 'g');
329 -       title('Acceleration vs Time') %creates a title
330 -       xlabel('Time (s)') %labels x axis
331 -       ylabel('Acceleration (m/s^2)') %labels y axis
332 -       grid on
333 -   end
```

A-2 Staging Optimization Function

```
%% CONSTANTS
targetHeight = 1.05e5; % Target altitude in meters, just above the karman line
m1Low = 220;
m1High = 222;
m1Int = 0.05;

m2Low = 49;
m2High = 52;
m2Int = 0.05;

ispan = round((m1High - m1Low) / m1Int);
jspan = round((m2High - m2Low) / m2Int);

Result = zeros(ispan * jspan, 3);

for m1 = m1Low:m1Int:m1High
    i = round((m1 - m1Low) / m1Int);
    for m2 = m2Low:m2Int:m2High
        j = round((m2 - m2Low) / m2Int);

        row = i * jspan + j;
        h = Simulate(m1, m2);
        Result(row + 1, :) = [m1, m2, h];
    end
end

Result = Result(Result(:,3) >= targetHeight, :);

% Add Total Mass Column
Result(:,4) = Result(:,1) + Result(:,2);

% Sort by smallest total mass
Result = sortrows(Result, 4);
```

A-3 Stage Mass Calculation

```
function [m_stage] = getstagemass(finert, mprop)

%% CALCULATIONS
m_stage = finert * mprop / (1 - finert) + mprop; %calculates stage mass (kg)

end
```

A-4 Fuel Parameter Calculations

```
function [mdot, ve, ae, pe] = getfuelparams(stageMass, stageStartHeight, tToW)
%% CONSTANTS
% Taken from NASA CEA program
SonicVelocityExit = 906.0; % from CEA
MachNumExit = 2.779; % from CEA
pe = 214640; % Pascals, from CEA
chamberPressure = 6.722e6; % pascals, from CEA
cstar = 1695.5; % m/s, from CEA
cf = 1.4851; % unitless, from CEA
aeToAt = 5; % exit area to throat area, from CEA

%% CALCULATIONS
ve = SonicVelocityExit * MachNumExit;

stageInitialWeight = getweight(stageMass, stageStartHeight);
neededThrust = stageInitialWeight * tToW;

throatArea = neededThrust / cf / chamberPressure;
ae = throatArea * aeToAt;

mdot = chamberPressure * throatArea / cstar;

end
```

A-5 Change in Mass Calculation

```
function [dm] = getdm(dt, stageMass, stageStartHeight, tToW)

%% CALCULATIONS
mdot = getfuelparams(stageMass, stageStartHeight, tToW);

% calculates the change in mass of the fuel per iteration, kg
dm = dt * mdot;

end
```

A-6 Pressure and Density Calculation Function

```
function [rho_si, p_si] = getpp(alt)

%% Function Description
% This function calculates the density and pressure of the air at the
% altitude designated for each time iteration.

%% Initialization

hg = 0:500:100000; %geometric altitude vector in feet
h_si = hg * 0.3048 / 1000; %altitude vector in kilometers
R = 287; %the gas constant in joules per kilogram Kelvin
a1 = -6.5 * 10 ^ -3; %slope for section one in Kelvin per meter
a2 = 3 * 10 ^ -3; %slope for section two in Kelvin per meter
p_init = 101324.02081; %the initial pressure in Newtons per meter squared
g = 9.81; %the gravitational acceleration
rho_init = 515.378818 * 2.3769 * 10 ^ -3; %the initial density in kilograms per meters cubed
gamma = 1.4; %speed of sound constant that is dimensionless

%% calculations

if (alt >= 0 && alt <= 36000) % if statement that creates the temperature, pressure, density, and speed for the first section of altitude in SI units
    t_K = 288.16 + a1 * alt * 0.3048; %creates the first section of the temperature vector in Kelvin
    p_si = p_init * exp(-(g / (R * t_K)) * (alt - hg(1)) * 0.3048); %creates the first section of the pressure vector in Newtons per meters squared
    rho_si = rho_init * exp(-(g / (R * t_K)) * (alt - hg(1)) * 0.3048); %creates the first section of the density vector in kilograms per meters cubed

elseif (alt > 36000 && alt <= 82000) %else if statement that creates the temperature, pressure, density, and speed for the second section of altitude in SI units
    t_K = 216.66; %creates the second section of the temperature vector in Kelvin
    p_si = p_init * exp(-(g / (R * t_K)) * (alt - hg(1)) * 0.3048); %creates the second section of the pressure vector in Newtons per meters squared
    rho_si = rho_init * exp(-(g / (R * t_K)) * (alt - hg(1)) * 0.3048); %creates the second section of the density vector in kilograms per meters cubed

else %else statement that creates the temperature, pressure, densit, and speed for the third section of altitude in SI units
    t_K = 216.66 + a2 * (alt * 0.3048 - 25000); %creates the third section of the temperature vector in Kelvin
    p_si = p_init * exp(-(g / (R * t_K)) * (alt - hg(1)) * 0.3048); %creates the third section of the pressure vector in Newtons per meters squared
    rho_si = rho_init * exp(-(g / (R * t_K)) * (alt - hg(1)) * 0.3048); %creates the third section of the density vector in kilograms per meters cubed
end

```

A-7 Thrust Calculation

```
function thrust = getthrust(pAmbient, stageMass, stageStartHeight, tToW)

% Retrieve fuel parameters
[mdot, ve, ae, pe] = getfuelparams(stageMass, stageStartHeight, tToW);

%% CALCULATIONS
thrust = mdot * ve + ae * (pe - pAmbient); %calculate thrust

end
```

A-8 Weight Calculation

```
function [w] = getweight(m, h)

%% INITIALIZATION
m_e = 5.98 * 10 ^ 24; %mass of the earth (kg)
G_e = 6.67 * 10 ^ -11; %gravitational constant on earth (N*m^2/kg^2)
r = 6.38 * 10 ^ 6; %radius of the earth (m)

%% CALCULATIONS
g = G_e * m_e / (r + h) ^ 2; %calculates gravity at the given height (m/s^2)
w = m * g; %finds weight of object with calculated gravity (N)

end
```

A-9 Drag Calculation

```
function [drag] = getdrag(rho,v)

%% INITIALIZATION
% rho = density of the air depending on elevation of the rocket
% v = velocity of the rocket depending on time
s = .2733; % cross sectional area of the rocket m^2
cd = .295;% drag coefficient for a cone

%% CALCULATIONS
drag = (1/2) * rho * s * cd * v^2; % calculates the force of drag on the rocket

end
```

A-10 Net Force Calculation

```
function [fnet] = getfnet(thrust,weight,drag)
% thrust of the rocket engine depending on time
% weight of the rocket depending on time
% drag on the rocket dependign on altitude

%% CALCULATIONS
fnet = thrust - weight - drag; % sums all the forces acting on the rocket including thrust weight and drag

end
```

A-11 Acceleration Calculation

```
function [a] = getaccel(fnet, m)

%% CALCULATIONS
a = fnet / m; %calculate acceleration based off of fnet and mass (m/s^2)

end
```

A-12 Velocity Calculation

```
function vnew = getvel(velocity, acceleration, dt)

%% CALCULATIONS
vnew = velocity + dt * acceleration; %Update velocity

end
```

A-13 Height Calculation

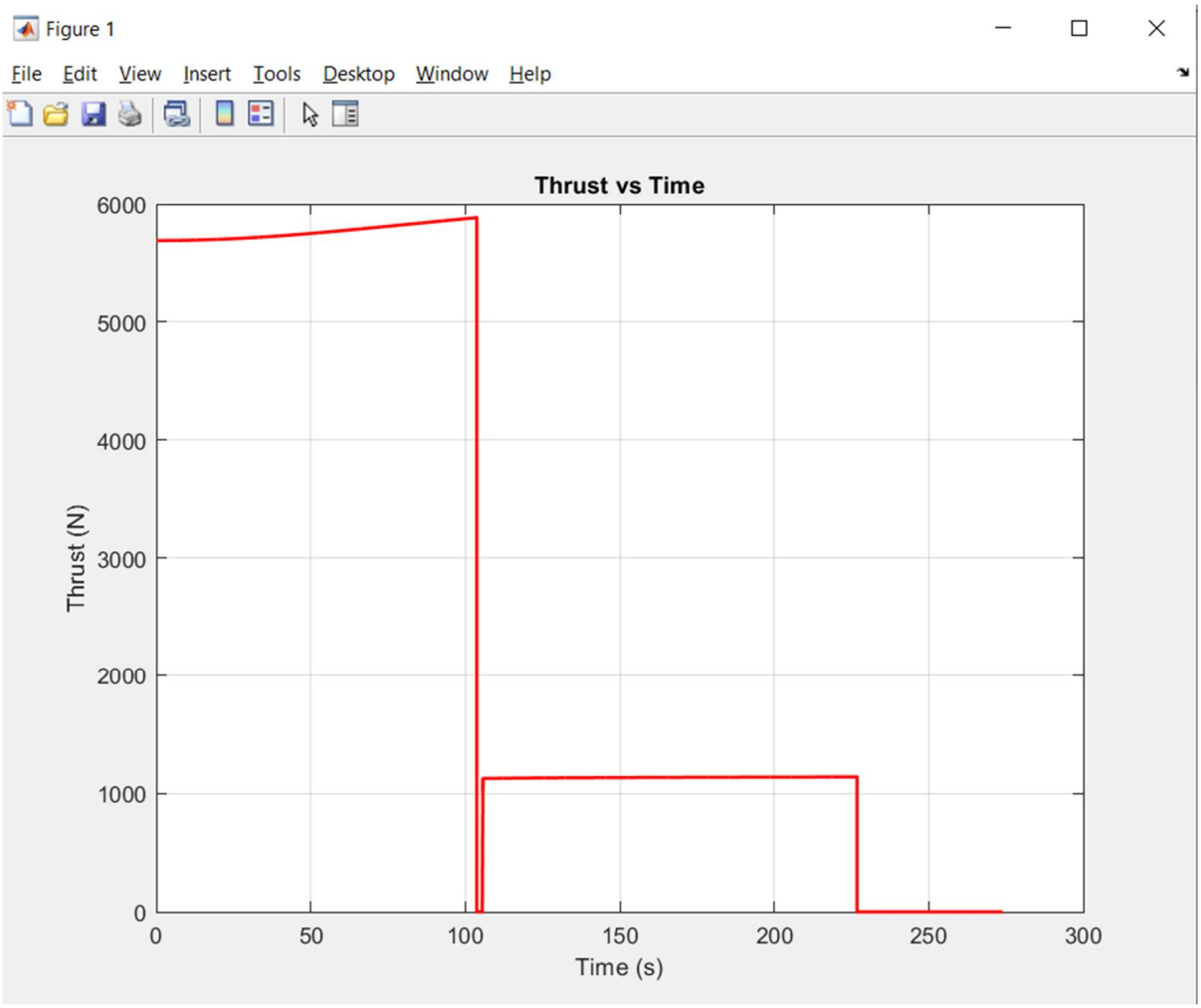
```
function heightnew = gethigh(height, velocity, dt)

%% CALCULATIONS
heightnew = height + dt * velocity; %Update height

end
```

A-14 New Mass Calculation

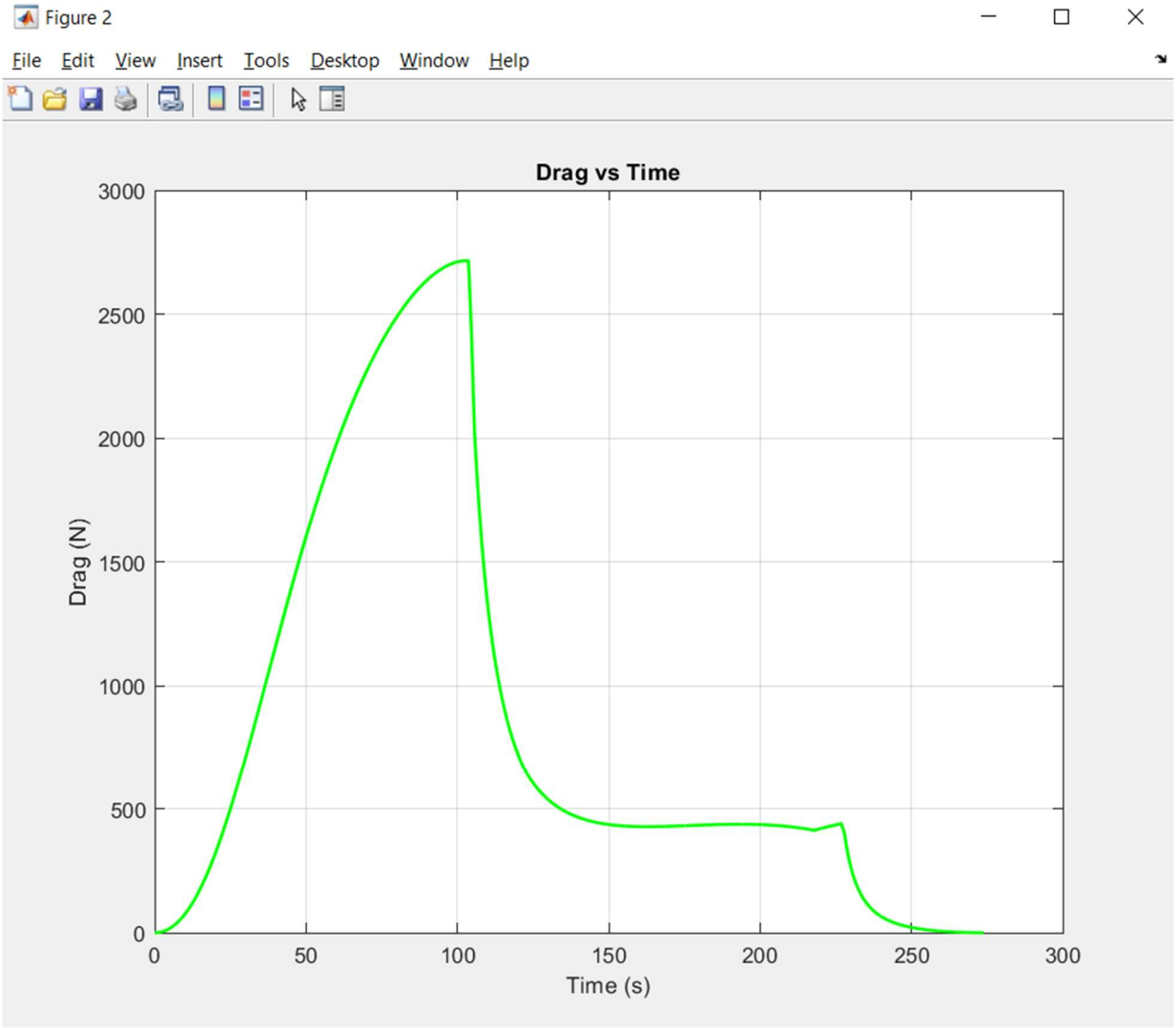
```
function [mnew] = getmnew(m,dm)

%% INITIALIZATION
% m = initial mass of the rocket
% dm = rate of change of mass
% t = time

%% CALCULATIONS
mnew = m - dm; % calculates the mass of the rocket after a certain time
               % after launch depending on the rate of mass being expelled

end
```

Appendix B: Plots

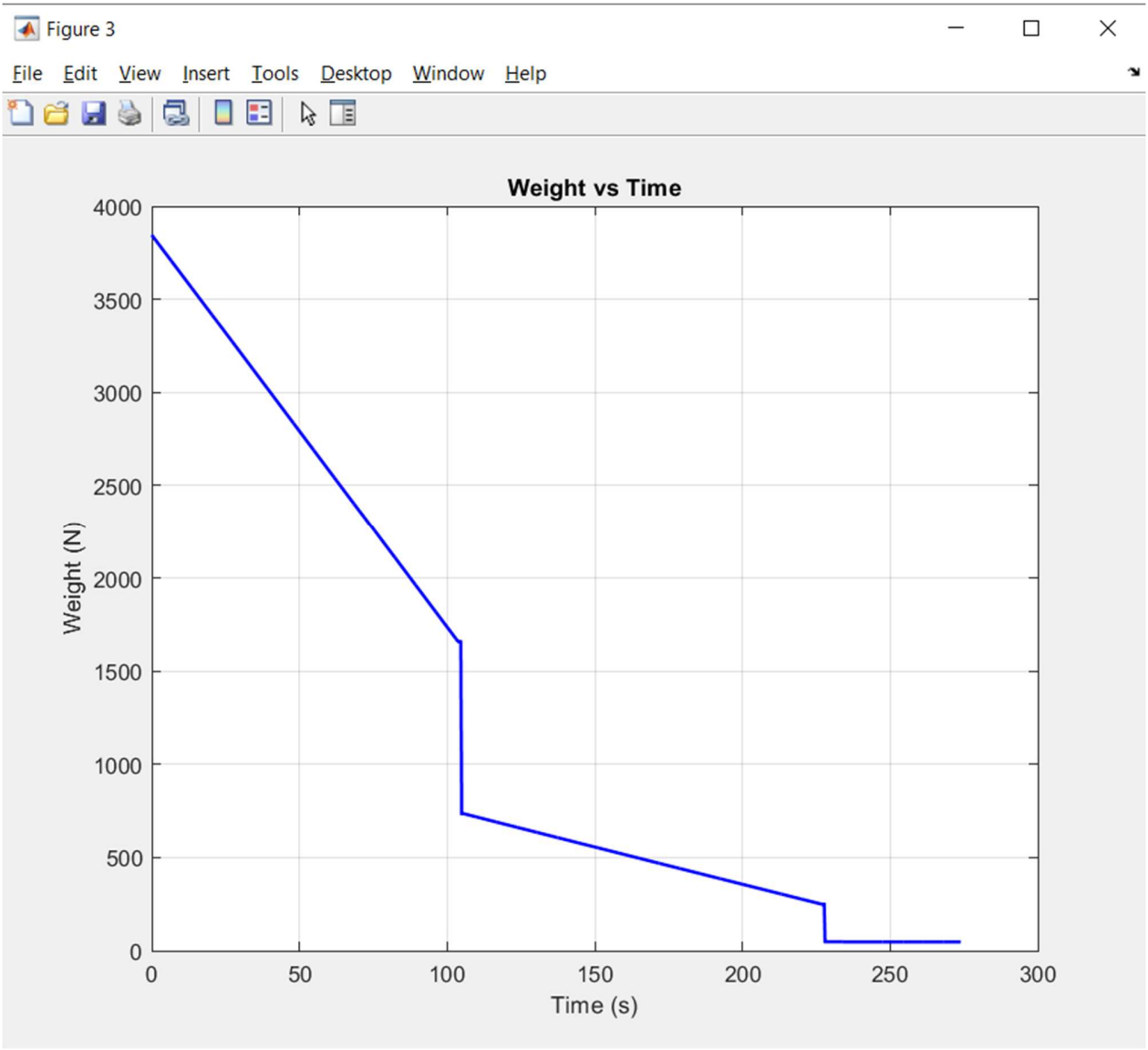
B-1 Thrust vs Time



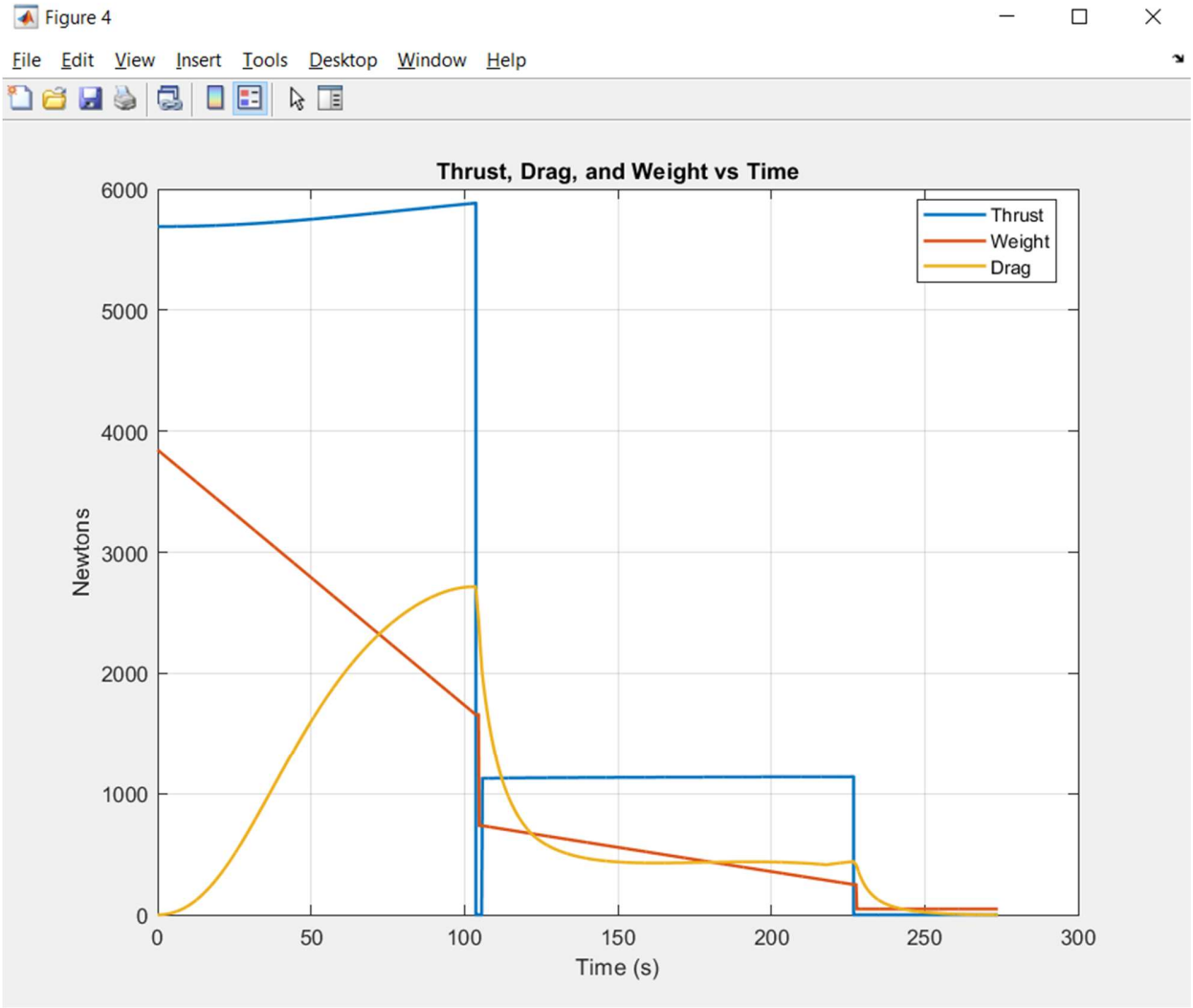
B-2 Drag vs Time



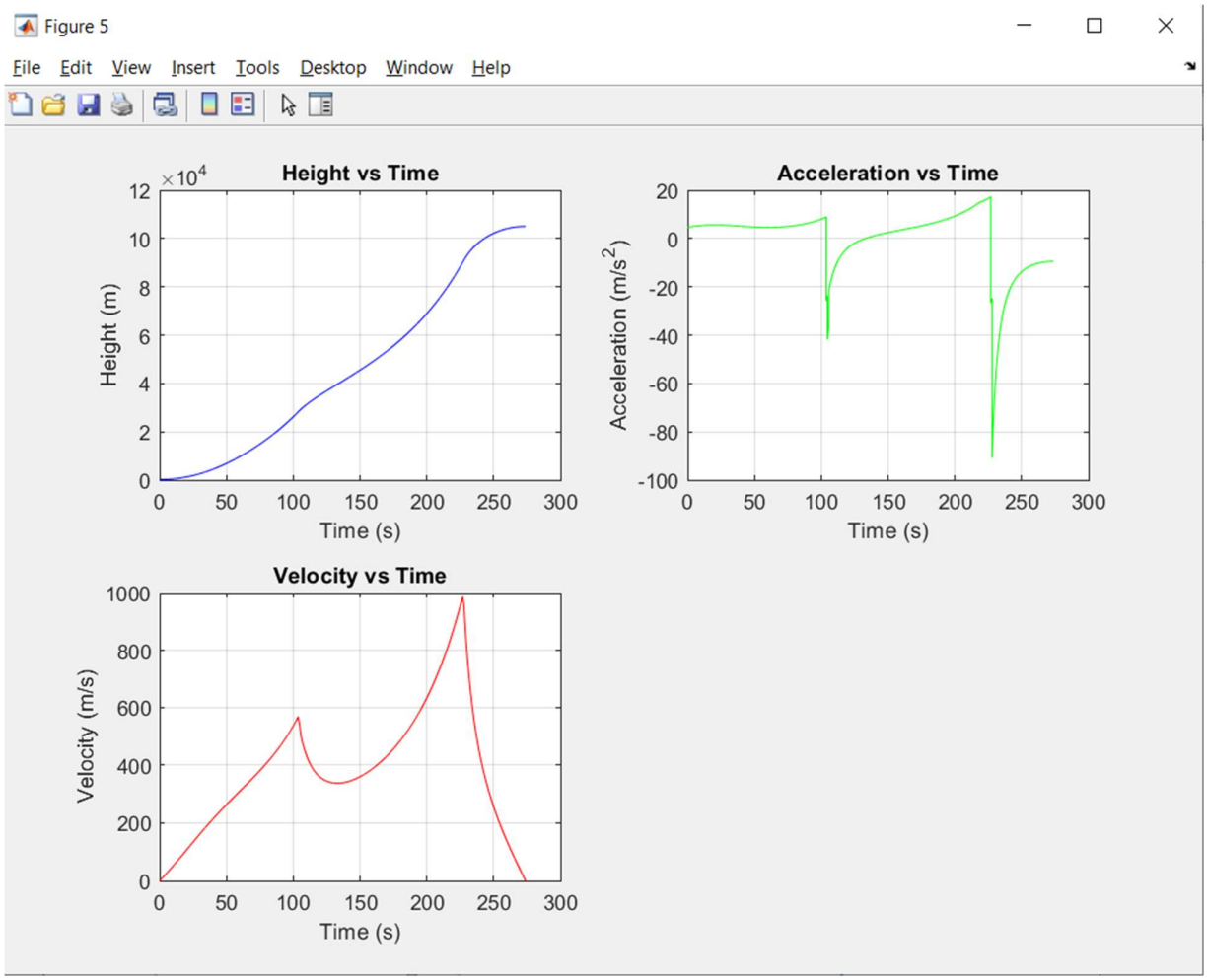
B-3 Weight vs Time



B-4 Thrust, Drag, Weight vs Time



B-5 Height, Acceleration, Velocity vs Time



Appendix C: Other Figures & Tables

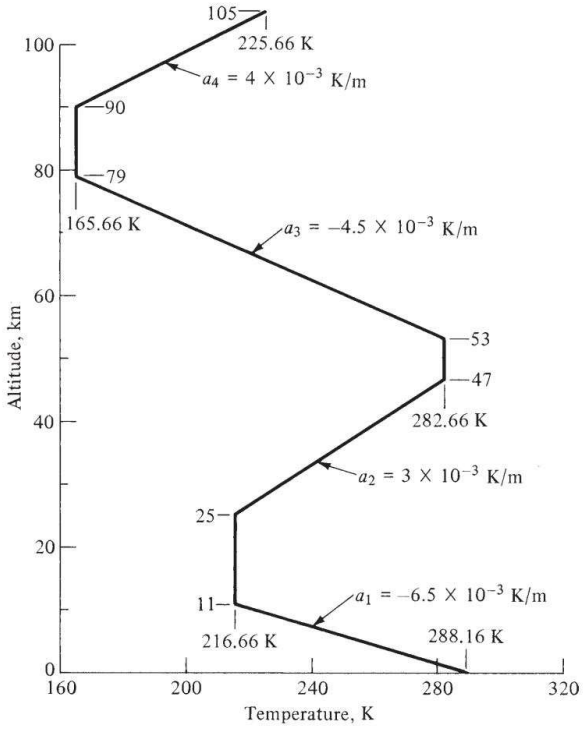
C-1 NASA ARCAS Mass Table

Table II

Detailed Weight Breakdown of the Frangible ARCAS Vehicle (Less Payload)

Component	Nominal Weight at Lift-Off (lbs)	Nominal Weight at Burnout (lbs)
Motor Case Assembly	16.50	15.20
Fin Assembly	1.69	1.69
Fin Screws	0.03	0.03
Propellant Assembly*	42.70	1.00
Retaining Sleeve	1.49	1.49
Explosive Module Fwd. Plate	0.32	0.32
Explosive Module Aft Plate	0.29	0.29
Push Rod	0.02	0.02
Mechanical Timer Assembly	0.70	0.70
Redundant Initiator	0.30	0.30
Primary Explosive Charge	2.06	2.06
Sheet Explosive Charge & Overwrap	2.30	2.30
	68.40 lbs.	25.40 lbs.

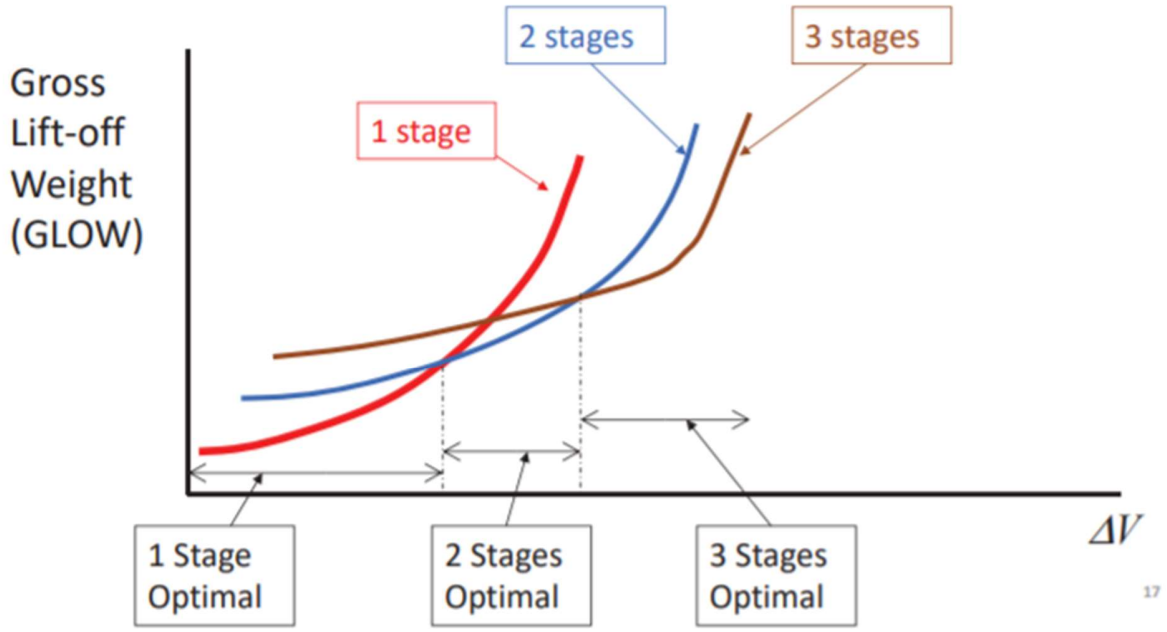
C-2 Standard Atmosphere Temperature Model



C-3 Rocket Staging GLOW vs ΔV

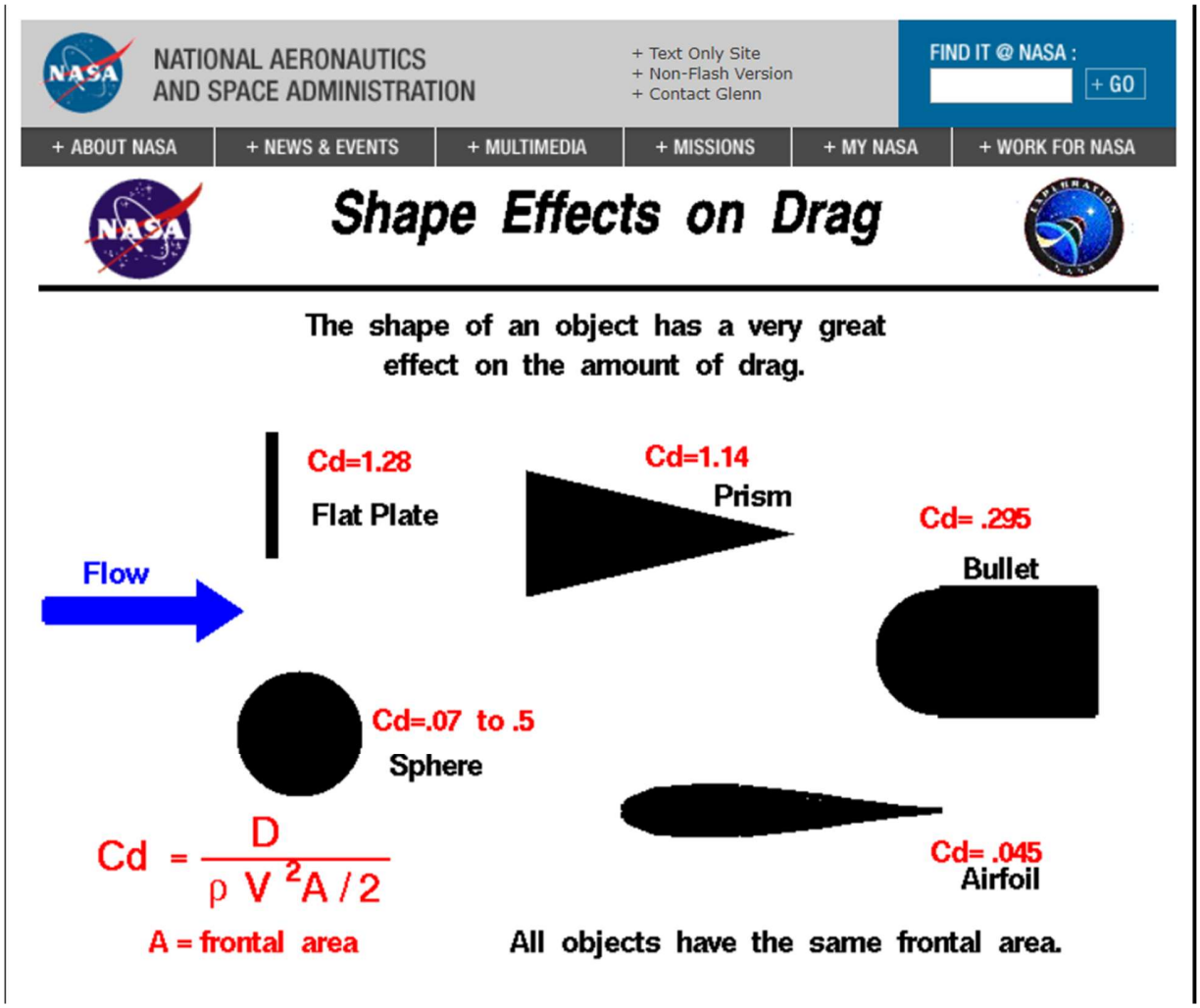
Multi-Stage Rockets

Multi-staged rockets are used to achieve a **higher *ΔV*** for the vehicle, and reduce inert mass, by "dropping" stages when they are no longer useful.



17

C-4 NASA Shape Effects on Drag



C-5 Decision Matrix

Figure C5-0-1

Criteria	Weight	Cost	Stages Required	Isp	Thrust to Weight Ratio	Empty Weight Fraction	Toxicity	Infrastructure Required	Fuel Stability
Inexpensive	20	X							
Easy to build	30		X				X	X	
Does not rapidly disassemble self	50						X		X
Reaches Karman Line	50			X	X	X			
Does not damage probe during the launch	50								X
	Units	$	#	s	ratio	ratio	relative	relative	

Figure C5-0-2

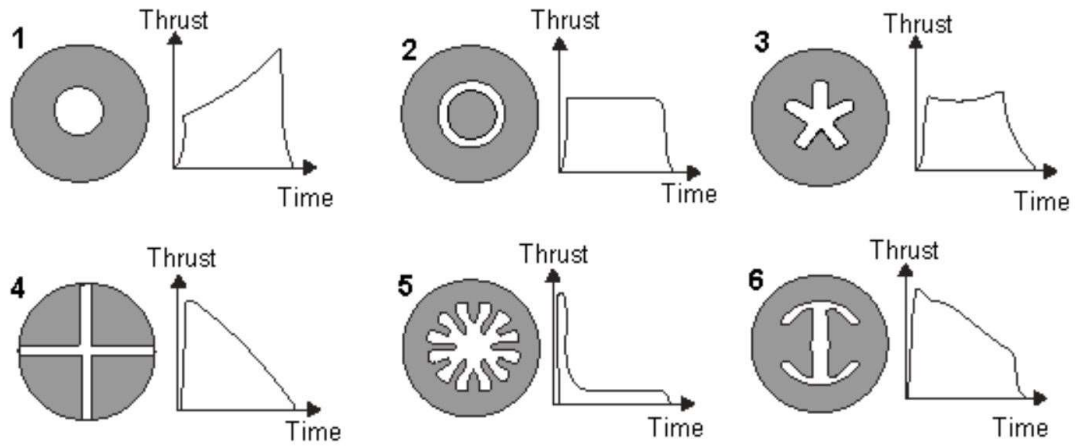
Unweighted Totals			
	Multi Stage Solid	Multi Stage Liquid	Mulit Stage Combination
Inexpensive	?	?	?
Easy to build	12	7	7
Does not rapidly disassemble self	9	8	8
Reaches Karman Line	8	9	9
Does not damage probe during the launch	5	5	5
Weighted Totals			
Inexpensive (20)	?	?	?
Easy to build (30)	360	210	210
Does not rapidly disassemble self (50)	450	400	400
Reaches Karman Line (50)	400	450	450
Does not damage probe during the launch (50)	250	250	250
	1460	1310	1310

Figure C5-3

Benchmarking

Cost	Dollars	Score	
	80000000	1	
	50000000	2	
	30000000	3	
	10000000	4	
	4	5	
Stages Required	#	Score	
	3	1	
		2	
	2	3	
		4	
	1	5	
ISP	Seconds	Score	
	<250	1	
	250-325	2	
	325-350	3	
	350-375	4	
	>375	5	
Thrust to Weight Ratio	Ratio	Score	
	1.4	1	
	1.5	2	
	1.6	3	
	1.7	4	
	>1.7	5	
Empty Weight Fraction	Ratio	Score	
	0.1	1	
		2	
	0.5	3	
		4	
	0.8	5	
Toxicity	How fast it kills	Score	Description
	Kills	1	Affected persons is killed by fuel or exhaust
		2	
	Somewhat Kills	3	Affected persons may be killed by fuel contact,
		4	but not terminally effected by exhuast
	Does not kill	5	Affected persons is not affected by fuel or exhaust
Infrastructure Required	Relative	Score	Description
		1	Full tower with ambilicals for cryogenic fuels
		2	
		3	Launchpad and external ignition necessary
		4	
		5	Hardly any infrastructure, pretty much just the airport runway as is
Fuel Stability	Oxidation Rate	Score	Description
		1	Similar to Florine: Reactions with fuel systems itself
		2	
		3	
		4	
		5	Similar to Kerosene: Relatively inert, does not spontaneously react

C-6 Solid Fuel Burn Geometry Comparisons



C-7 NASA ARCAS Rocket Motor Performance

Nozzle Throat Area, in^2	0.197
Nozzle Exit Area, in^2	2.550
Expansion Ratio	13
Average Chamber Pressure, psia	975
Average Thrust, lb	325
Maximum Chamber Pressure, psia	1080
Maximum Thrust, lb	360
Total Impulse, lb-sec	9400
Burning Time, sec	30

C-8 NASA CEA Input File

CEA_Analysis_Toads.inp.txt

```
prob rocket p,psia=975 sub=5,10,20,50 sup=5,10,20,100

reac
 name = HTPB C 10 H 15.4 O 0.07 h,kj=-51.9 t,k=298.15 wt%=10
 name = AL t,k=298.15 wt%=20
 name = Fe2O3(cr) t,k=298.15 wt%=0.2
 name = NH4CLO4(I) t,k=298.15 wt%=68.8

output siunits
end
```

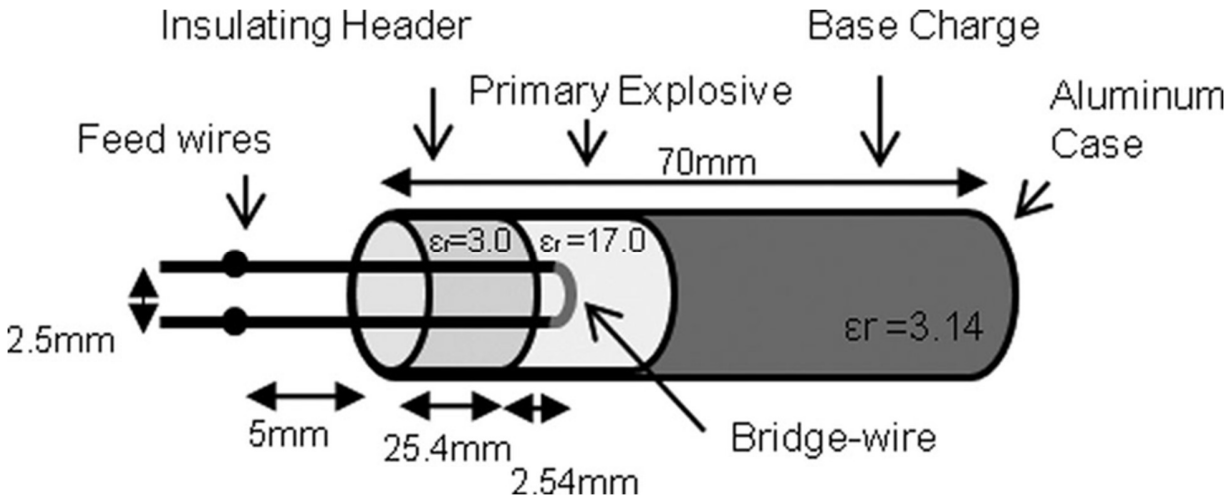
C-9 NASA CEA Output File (Abbreviated)

CEA_Analysis_Toads.out

```
 1
 2    ******************************************************************************
 3
 4              NASA-GLENN CHEMICAL EQUILIBRIUM PROGRAM CEA2, MAY 21, 2004
 5                            BY  BONNIE MCBRIDE AND SANFORD GORDON
 6          REFS: NASA RP-1311, PART I, 1994 AND NASA RP-1311, PART II, 1996
 7
 8    ******************************************************************************
 9
10                                  CHAMBER   THROAT     EXIT     EXIT     EXIT     EXIT     EXIT
11    Pinf/P                         1.0000   1.7682   1.0085   1.0021   1.0005   1.0000   31.319
12    P, BAR                         67.224   38.019   66.654   67.082   67.188   67.221   2.1464
13    T, K                          4061.96  3703.56  4056.39  4060.58  4061.62  4061.93  2298.72
14    RHO, KG/CU M    5.5813 0 3.4620 0 5.5417 0 5.5715 0 5.5789 0 5.5811 0 3.1491-1
15    H, KJ/KG                       672.60   16.846   662.36   670.06   671.96   672.54 -2497.67
16    U, KJ/KG                      -531.84 -1081.32  -540.43  -533.97  -532.37  -531.89 -3179.28
17    G, KJ/KG                     -39260.7 -36393.0 -39216.1 -39249.7 -39258.0 -39260.5 -25096.5
18    S, KJ/(KG)(K)                  9.8310   9.8310   9.8310   9.8310   9.8310   9.8310   9.8310
19
20    M, (1/n)                       28.041   28.041   28.041   28.041   28.041   28.041   28.041
21    MW, MOL WT                     26.251   26.251   26.251   26.251   26.251   26.251   26.251
22    Cp, KJ/(KG)(K)                 1.8362   1.8229   1.8360   1.8362   1.8362   1.8362   1.7479
23    GAMMAs                         1.1926   1.1943   1.1926   1.1926   1.1926   1.1926   1.2043
24    SON VEL,M/SEC                  1198.5   1145.2   1197.7   1198.3   1198.4   1198.5    906.0
25    MACH NUMBER                     0.000    1.000    0.119    0.059    0.030    0.008    2.779
26
27    PERFORMANCE PARAMETERS
28
29    Ae/At                                  1.0000   5.0000   10.000   20.000   50.000   5.0000
30    CSTAR, M/SEC                           1695.5   1695.5   1695.5   1695.5   1695.5   1695.5
31    CF                                     0.6754   0.0844   0.0420   0.0210   0.0059   1.4851
32    Ivac, M/SEC                            2104.1   8549.0  16991.0  33931.3 119492.4   2788.7
33    Isp, M/SEC                             1145.2    143.1     71.2     35.5     10.1   2518.0
34
35
```

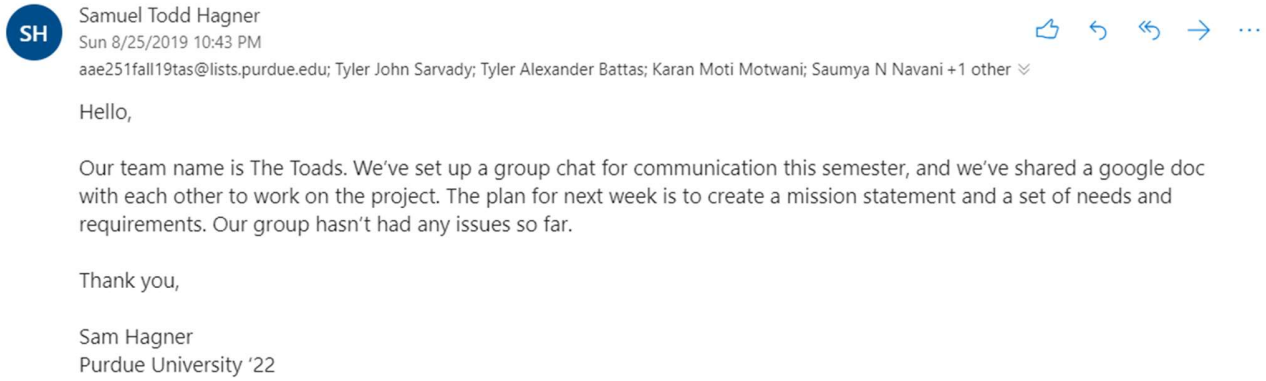
C-10 EED Diagram

(Mora, 2012)



Appendix D: Weekly Email Updates

D-1 Update 1



Samuel Todd Hagner
Sun 8/25/2019 10:43 PM
aae251fall19tas@lists.purdue.edu; Tyler John Sarvady; Tyler Alexander Battas; Karan Moti Motwani; Saumya N Navani +1 other

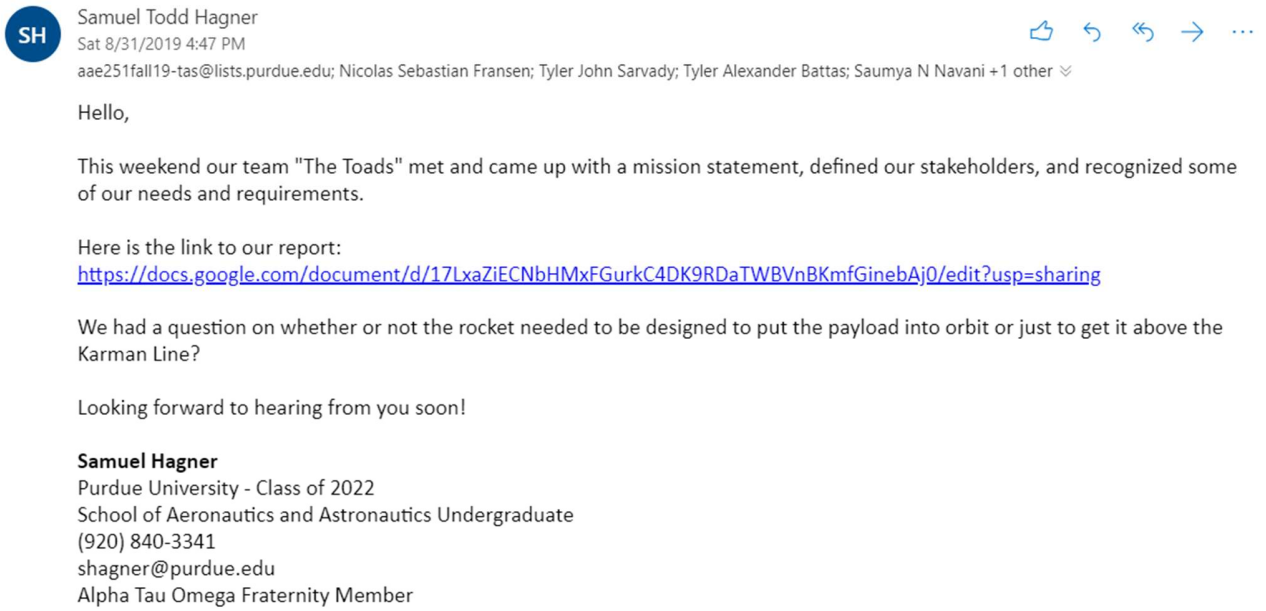
Hello,

Our team name is The Toads. We've set up a group chat for communication this semester, and we've shared a google doc with each other to work on the project. The plan for next week is to create a mission statement and a set of needs and requirements. Our group hasn't had any issues so far.

Thank you,

Sam Hagner
Purdue University '22

D-2 Update 2



Samuel Todd Hagner
Sat 8/31/2019 4:47 PM
aae251fall19-tas@lists.purdue.edu; Nicolas Sebastian Fransen; Tyler John Sarvady; Tyler Alexander Battas; Saumya N Navani +1 other

Hello,

This weekend our team "The Toads" met and came up with a mission statement, defined our stakeholders, and recognized some of our needs and requirements.

Here is the link to our report:
https://docs.google.com/document/d/17LxaZiECNbHMxFGurkC4DK9RDaTWBVnBKmfGinebAj0/edit?usp=sharing

We had a question on whether or not the rocket needed to be designed to put the payload into orbit or just to get it above the Karman Line?

Looking forward to hearing from you soon!

Samuel Hagner
Purdue University - Class of 2022
School of Aeronautics and Astronautics Undergraduate
(920) 840-3341
shagner@purdue.edu
Alpha Tau Omega Fraternity Member

D-3 Update 3

SH

Samuel Todd Hagner
Sun 9/8/2019 3:46 PM

aae251fall19-tas@lists.purdue.edu; Tyler Alexander Battas; Tyler John Sarvady; Nicolas Sebastian Fransen +2 others

Hello,

This week our team met on Sunday at 3 pm and touched up on our mission statement, needs and requirements, and stakeholders. After this we felt confident in our setup so we submitted the first project update.

Thank you,

The Toads

Samuel Hagner
Purdue University - Class of 2022
School of Aeronautics and Astronautics Undergraduate
(920) 840-3341
shagner@purdue.edu
Alpha Tau Omega Fraternity Member

D-4 Update 4

NF

Nicolas Sebastian Fransen
Mon 9/16/2019 12:00 AM

aae251fall19-tas@lists.purdue.edu; Tyler Alexander Battas; Tyler John Sarvady; Karan Moti Motwani; Samuel Todd Hagner +1 other

Hello,

This weekend our team held a discussion about potential orbits for our probe and the considerations surrounding orbital parameters such as latitude, inclination, and azimuth, based upon what we learned in class, and the mission requirements of our project. The Probe must be able to cover as many key areas as possible in order to detect other humans.

Thanks,

The Toads

D-5 Update 5

SN **Saumya Naresh Navani <navani@purdue.edu>**
Sun 9/22/2019 11:54 PM
Islam Aly Sadek Nazmy; Nicolas Sebastian Fransen; aae251fall19-tas@lists.purdue.edu; Tyler John Sarvady +3 others

Here is the link to the document
https://docs.google.com/document/d/17LxaZiECNbHMxFGurkC4DK9RDaTWBVnBKmfGinebAj0/edit?usp=drivesdk

Get Outlook for Android

...

SN **Saumya Naresh Navani <navani@purdue.edu>**
Sun 9/22/2019 11:47 PM
Islam Aly Sadek Nazmy; Nicolas Sebastian Fransen; aae251fall19-tas@lists.purdue.edu; Tyler John Sarvady +3 others

Hello,
This week our team discussed potential launch sites based on whether we decide to launch to either a polar orbit, a geostationary orbit, or to LEO (given that we can launch from places other than Purdue). Assuming that the only launch site available is Purdue's airport, we worked on calculating the azimuth, inclination and velocity values for the particular orbits we had in mind.

Thanks,
The Toads.

D-6 Update 6

SH Samuel Todd Hagner
Sun 9/29/2019 10:06 PM
aae251fall19-tas@lists.purdue.edu; Nicolas Sebastian Fransen; Tyler Alexander Battas; Tyler John Sarvady +2 others

Hello,

Our team met tonight and added multiple needs and requirements to our report document. We also tried to get on the same page about the mission. We have been somewhat confused as to what the end goal of the mission is due to some differences between the project report template, the design project deliverable document, and what the TA's have been telling us. The project deliverable document makes no mention of putting the probe into orbit: "We just have to get the probe to an altitude of 100km". Looking at the FAQ document, it also says that the end goal is to reach the Karman Line.
Also, the initial template says that we have to come up with requirements for the probe and the rocket, but the FAQ document says that we just need requirements for the launch device. We're hoping that you could make some type of announcement in class on Tuesday so that all of the students are completely sure of the task at hand.

Thank you for your hard work and understanding,

The Toads

Samuel Hagner
Purdue University - Class of 2022
School of Aeronautics and Astronautics Undergraduate
(920) 840-3341
shagner@purdue.edu
Alpha Tau Omega Fraternity Member

D-7 Update 7

NF

Nicolas Sebastian Fransen
Sun 10/13/2019 11:52 PM
aae251fall19-tas@lists.purdue.edu; Tyler Alexander Battas; Tyler John Sarvady; Karan Moti Motwani; Saumya Naresh Navani +1 other

Hello,

This week our team performed rough calculations to begin estimating the design parameters of our rocket, using the rocket equation. Since we have not learned staging yet, we assumed it would be a single-stage rocket for now and we will compare our results with those obtained from a multi-stage rocket once staging is covered in class.

https://docs.google.com/document/d/17LxaZiECNbHMxFGurkC4DK9RDaTWBVnBKmfGinebAj0/edit?usp=sharing

AAE251 Report Template

The BFT I (The Big Flying Toad) Authors\u000BSam Hagner\u000BTyler Sarvady\u000BTyler Battas\u000BKaran Motwani\u000BSaumya Navani\u000BNicolas Fransen AAE251\u000B\u000B\u000B\u000B\u000BFall 2019 Abstract—The abstract is a summary of your entire report. It provides a discussion of the problem, your approach to solving it, and your findings. Find ...

docs.google.com

Thanks,

The Toads
AAE 251 Team R16 Afternoon section

D-8 Update 8

SH

Samuel Todd Hagner
Sun 10/27/2019 11:02 PM
aae251fall19-tas@lists.purdue.edu; Tyler John Sarvady; Tyler Alexander Battas; Nicolas Sebastian Fransen +2 others

Hello,
This week our team met and discussed our criteria and metrics for a decision matrix. The three different rocket we chose to compare are multi stage solid, multi stage mixed solid and liquid, Space Shuttle configuration solid with liquid. We are going to meet up multiple times in the upcoming week to choose our final choice rocket. We are going to do research on our own to find all the values for metrics based off historical data.

Thank you

Samuel Hagner
Purdue University - Class of 2022
School of Aeronautics and Astronautics Undergraduate
(920) 840-3341
shagner@purdue.edu

D-9 Update 9

KM

Karan Moti Motwani
Sun 11/3/2019 11:56 PM
aae251fall19-tas@lists.purdue.edu; Tyler Alexander Battas; Tyler John Sarvady; Saumya Naresh Navani; Nicolas Sebastian Fransen

Hello,

This week our team continued work on the decision matrix. We found the historical data for the 3 configurations we had in consideration: Multi-stage solid, multi-stage liquid, and the multi-stage solid - liquid configuration. However, we found that we needed to adjust our metrics to more closely match our requirements, so a final decision was not reached. Also, based on the work we did in HW-5, we planned a MATLAB script that would help us determine how many stages we would need, although we would like to expand this to also calculate parameters such as delta-V allocation as well.

Thanks,
The Toads
AAE 251 Team R16 Afternoon section

https://docs.google.com/document/d/17LxaZiECNbHMxFGurkC4DK9RDaTWBVnBKmfGinebAj0/edit?usp=sharing

AAE251 Report Template

The BFT I (The Big Flying Toad) Authors\u000BSam Hagner\u000BTyler Sarvady\u000BTyler Battas\u000BKaran Motwani\u000BSaumya Navani\u000BNicolas Fransen AAE251\u000B\u000B\u000B\u000B\u000BFall 2019 Abstract—The abstract is a summary of your entire report. It provides a discussion of the problem, your approach to solving it, and your findings. Find ...

docs.google.com

D-10Update 10

SH

Samuel Todd Hagner
Mon 11/18/2019 1:05 AM
aae251fall19-tas@lists.purdue.edu; Tyler Alexander Battas; Tyler John Sarvady; Karan Moti Motwani +2 others

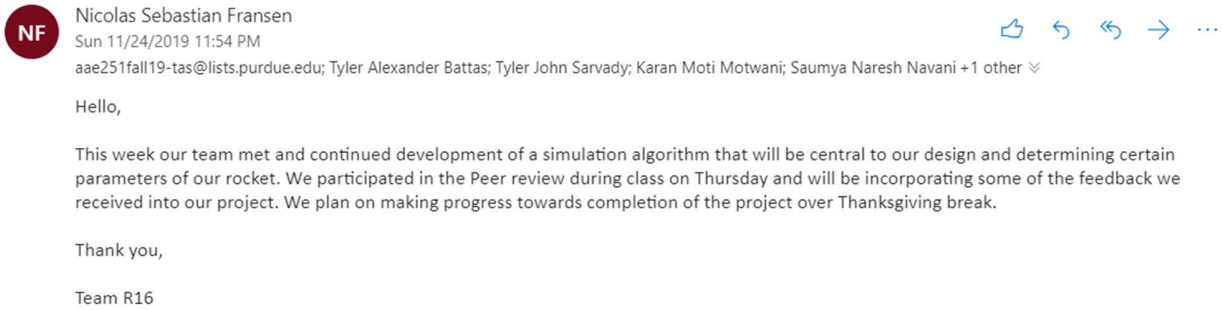
Hello,

Our team did not get much done on our project this week due to the amount of time we put in to preparing for our vehicle of the week. We plan on meeting multiple times this week in order to finish off the final project and hammer out the finishing touches.

Thank you,

Samuel Hagner
Purdue University - Class of 2022
School of Aeronautics and Astronautics Undergraduate
(920) 840-3341
shagner@purdue.edu

D-11Update 11



Nicolas Sebastian Fransen
Sun 11/24/2019 11:54 PM
aae251fall19-tas@lists.purdue.edu; Tyler Alexander Battas; Tyler John Sarvady; Karan Moti Motwani; Saumya Naresh Navani +1 other

Hello,

This week our team met and continued development of a simulation algorithm that will be central to our design and determining certain parameters of our rocket. We participated in the Peer review during class on Thursday and will be incorporating some of the feedback we received into our project. We plan on making progress towards completion of the project over Thanksgiving break.

Thank you,

Team R16

D-12Update 12

Samuel Todd Hagner
Sun 12/1/2019 10:25 PM
aae251fall19-tas@lists.purdue.edu

Hello,

This week we finished gathering all the data we need for the project. All we have left to do is complete final project report. We have a thorough outline done and will meet twice this week to finish the document.

Thank you,

Samuel Hagner
Purdue University - Class of 2022
School of Aeronautics and Astronautics Undergraduate
(920) 840-3341
shagner@purdue.edu