5-3-2024

Simulating the Performance of a Single Solid Propellant and Rocket Nozzle Design in a Vertical Trajectory

Kyle Austin Barbour
University of Alabama in Huntsville

Simulating the Performance of a Single Solid Propellant and Rocket Nozzle Design in a Vertical Trajectory

by

Kyle Austin Barbour

An Honors Capstone

submitted in partial fulfillment of the requirements

for the Honors Diploma

to

The Honors College

of

The University of Alabama in Huntsville

3rd May 2024

Honors Capstone Project Director: Dr. Robert Frederick

Kyle Austin Barbour 3rd May 2024

Student Date

5/4/24

Project Director Date

5/5/24

Department Chair Date

Sean Lane Digitally signed by Sean M Lane
Date: 2024.05.07 10:24:07 -05'00'

Honors College Dean Date

Honors College
Frank Franz Hall
+1 (256) 824-6450 (voice)
+1 (256) 824-7339 (fax)
honors@uah.edu

Honors Thesis Copyright Permission

This form must be signed by the student and submitted with the final manuscript.

In presenting this thesis in partial fulfillment of the requirements for Honors Diploma or Certificate from The University of Alabama in Huntsville, I agree that the Library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by my advisor or, in his/her absence, by the Chair of the Department, Director of the Program, or the Dean of the Honors College. It is also understood that due recognition shall be given to me and to The University of Alabama in Huntsville in any scholarly use which may be made of any material in this thesis.

Kyle Austin Barbour

____

Student Name (printed)

Kyle Austin Barbour

____

Student Signature

3rd May 2024

____

Date

Contents

Executive Summary

This project was an analysis of a bonus problem on the MAE 440, Rocket Propulsion I, course during the Fall 2023 Semester. It required an in-depth analysis of three identical rocket tested for three different vertical trajectories. A system for batch processing these rocket designs was created, and it was operated to attempt the discover of a solution. The problem had no known solutions at the start of this project.

After processing hundreds of millions of designs, an analysis was performed to determine where the difficulty of this problem stemmed from. The relationship between how different designs perform on each test was analyzed to recommend potential changes to the problem so more solutions would appear. Overall, this project successfully analyzed the challenges of this problem, and found a relatively minor change that would guarantee at least one solution.

Project Summary

This capstone project was an extension on the final project for MAE 440, Rocket Propulsion I, in during the Fall 2023 Semester. That final project tested each student's mastery of the topics taught during the semester. Students need an understanding of basic mission analysis, trajectory analysis, rocket nozzle performance, solid propellants, and thermochemistry to succeed in that project. I completed that project, but an optional criterion in that project caught my attention. It was a bonus problem. The bonus made minimal modifications to the project guidelines. Yet, nobody knew if it even had a solution. That is one way to get an engineer's attention.

The unknown began to bother me. So, I talked to the professor and setup up an honors capstone project to search for a solution. There were two mutually exclusive goals laid out for this capstone, and the primary goal was to find a solution. If a solution was not found, the goal would be to perform an analysis and determine a small change in the project guidelines that would create a solution. Those were the goals of this capstone.

Original Project Guidelines

The original project guidelines were still a corner stone in this capstone. Again, the bonus problem made minimal changes to these guidelines. So, this section seeks to break down the original guidelines. The first topic is the background of the project. This was a project for a rocket propulsion course; so, it is no surprise that the project revolved around a rocket launch. Specifically, the project focused on the launch of a sounding rocket. Sounding rockets are a type of rocket designed to carry scientific instruments to certain altitudes on a sub-orbital trajectory. Basically, these rockets are used to record data at various altitudes without orbiting the Earth. The guidelines reflected this by stating that the task was to design a sounding rocket that would reach a maximum altitude of five, ten, and fifteen thousand feet.

Now, the three target altitudes could be reached by three different rockets. Students were able to adjust a group of parameters such as the nozzle and ballast mass in-between launches. The rocket casing would even automatically adjust based on the amount of propellant used. However, the geometry of the solid rocket propellant had to remain the same between each launch. The solid propellant's design was based around a propellant grain. Basically, one grain is one piece of solid propellant. The number of grains in a rocket can be changed between launches, but the shape of each individual grain must be identical. For this project, the grain was defined as a cylinder with a central bore.

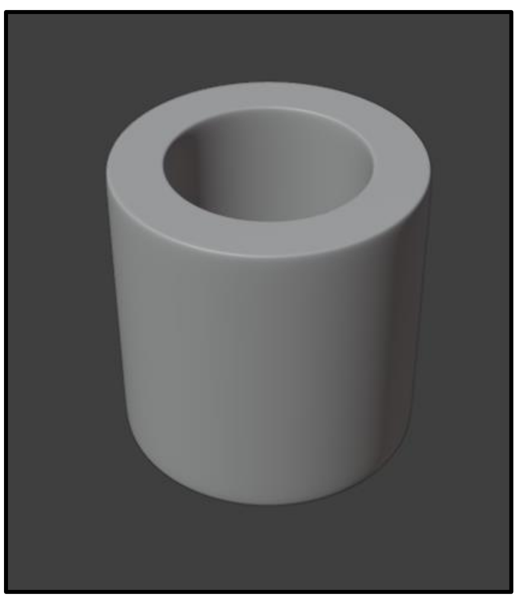

Figure 1: Grain Geometry Angled View

Figure 2: Grain Geometry Horizontal Cross Section

Figure 3: Grain Geometry Vertical Cross Section

So, every grain used had to follow this design. Every grain used had to be identical, and each rocket for a different test can use a different number of grains. Solid rocket propellant will burn on the surface area exposed; this is the burn area. Devices known as inhibiters can be used to change how the propellant burns, but the guidelines stated that no inhibiters were allowed. Inhibiters work by covering portions of the burn area to prevent it from burning. In the end, this propellant grain would burn along the inner radius, or bore radius, and from both ends. The outer radius would be up against the casing and not burn. If the grains are packed closely together, they can act as their own inhibitors. As a result, the guidelines specified that there would be a spacing of 0.125 inches before each grain. The spacing is only before because this put spacing between each grain; it additionally puts a spacing between the first grain and the top of the rocket’s casing. The last grain does not have a spacing requirement because it is placed before the rocket’s combustion chamber and nozzle. That placement leads to the assumption that the required spacing is already there.

Now, the rocket’s in this project used a converging diverging nozzle. This simply meant that the cross-sectional area of the nozzle decreased from the combustion chamber to a location called the throat. The throat is the location of the smallest cross-sectional area in the nozzle, and after it the cross-sectional area would increase.

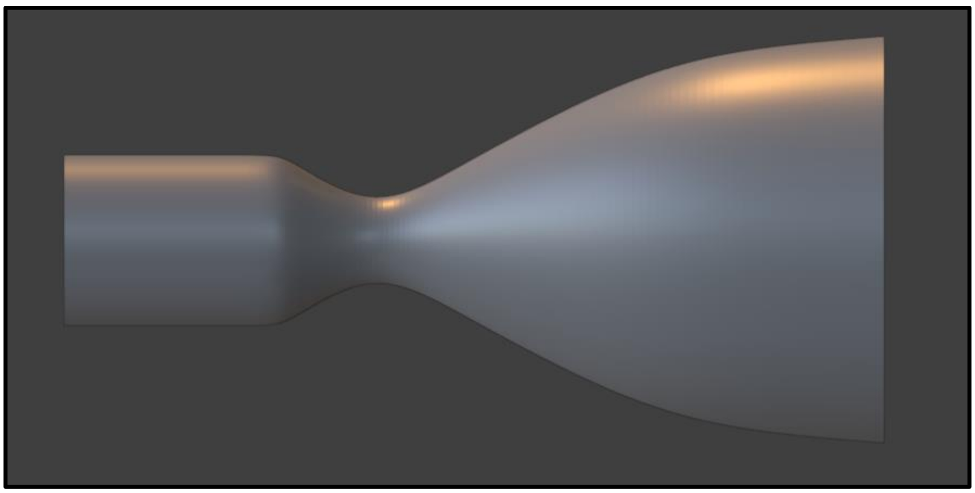

Figure 4: Converging-Diverging Rocket Nozzle

This design follows basic compressible flow principles. The hot gas from the combustion chamber would start moving at a sub-sonic speed: slower than the speed of sound. Sub-sonic flows will increase their velocity as the cross-sectional area they move through decreases. So, the first part of the nozzle is speeding up the flow. The flow would be expected to reach the speed of sound, Mach 1, at the throat of the nozzle. Once the flow reaches Mach 1, it becomes super-sonic. Super-sonic flows will decrease in velocity if the cross-sectional area decreases. The desired effect is to continue increasing the flow’s velocity; so, the nozzle starts expanding in area after the throat. The ratio of the nozzle’s exit area over its throat is known as the area ratio. If the area ratio falls below one, the flow of hot gas exiting the nozzle will be sub-sonic.

What if the flow of hot gas reaches Mach 1 before the nozzle’s throat? It is the same as when the flow reaches Mach 1 at the throat. The condition is referred to as a choked flow. The flow has become super-sonic, and its characteristics have changed. It now has a limited mass flow rate through the nozzle: shown in the following equation.

$$mass\ flow\ rate = density * velocity * area \tag{1}$$

A continuity principle states that the mass flow rate through the nozzle is constant. In the converging section, density and velocity will increase as the area decreases. Then, the diverging section will see an increase in velocity and area with a decrease in density. A choked flow is not bad by itself, but it is bad when the condition is met before the throat of the nozzle. Recall, super-sonic flows will slow down as the cross-section area decreases. A choked flow before the throat introduces the possibility that the flow will become sub-sonic before the throat. If the flow is sub-sonic at the throat, the diverging section will only slow the flow down more. This is bad for a rocket.

This situation is most likely to arise during the ignition of the rocket motor. To avoid this situation, the guidelines had guidance on the relationship between the bore area of the grain and the nozzle's throat area. If the bore area of the grain was not at least twice the throat area the flow would be considered choked before the throat. This effectively limits how low the bore radius of the grain can be based on the nozzle throat area. Now, the lower the bore radius the more propellant a rocket has. The bore is taking away material in the grain; so, this guidance was a limitation on the amount of propellant a single grain can contain.

The grains were not only limited by the bore radius. The outer radius of the grains was also limited, but it was not limited by such a complex condition. Instead, the outer radius of the grain was simply constrained to a radius of 2.375 inches by the guidelines. The rocket was stated as having a radius of 6.19 inches, and the difference between these two values could be considered the thickness of the rocket's casing. The casing had a defined maximum length: which limited the number of grains a design could use. The combustion chamber was rated for 1000 pounds per square inch of pressure. Anything over this value would have caused the chamber to rupture.

The nozzle also had pressure constraints: like the chamber pressure. However, these constraints were not as simply defined. The pressure of the flow through the nozzle will drop in the diverging section. So, there are two pressures present at the exit of the nozzle: the ambient pressure of the atmosphere and the exit pressure of the nozzle's flow. If these pressures are the same, the flow is perfectly expanded.

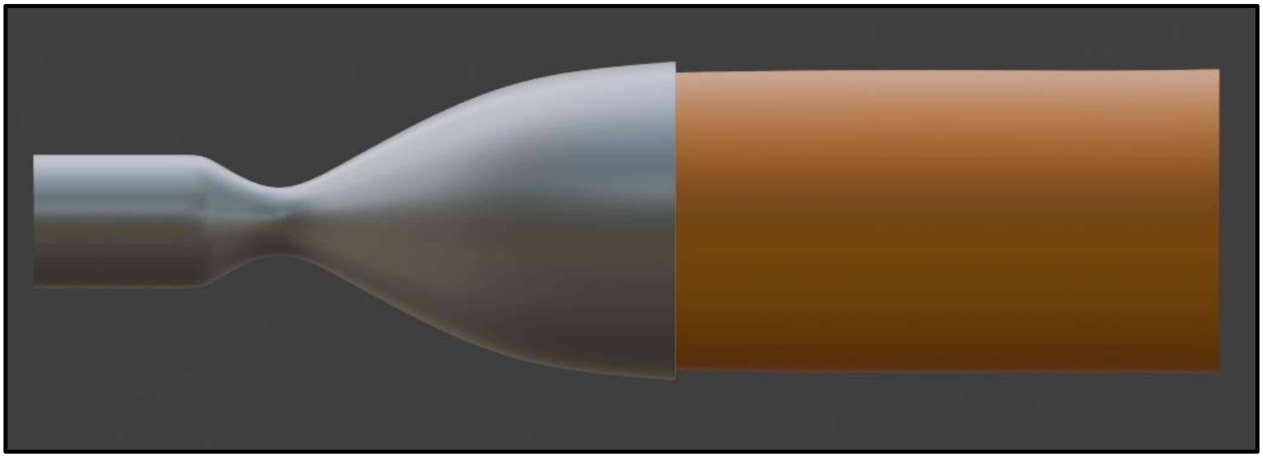

Figure 5: Perfectly Expanded Flow

Flows usually only perfectly expanded for a few moments during the rocket's flight. The exit pressure of the rocket's nozzle is based on the area ratio of the nozzle, and the ambient pressure is based on the altitude. Both values are constantly changing, and the flow is also changing. Typically, the flow will either be over expanded or under expanded. This naming convention can get a bit confusing. Both names are refereeing to the pressure difference between the exit pressure and the ambient pressure, but the names are based on the nozzle's design. The pressure of the super-sonic flow decreases as the diverging section expands. If the exit pressure is less than the ambient pressure, the nozzle decreases the flow's pressure too much by over expanding it.

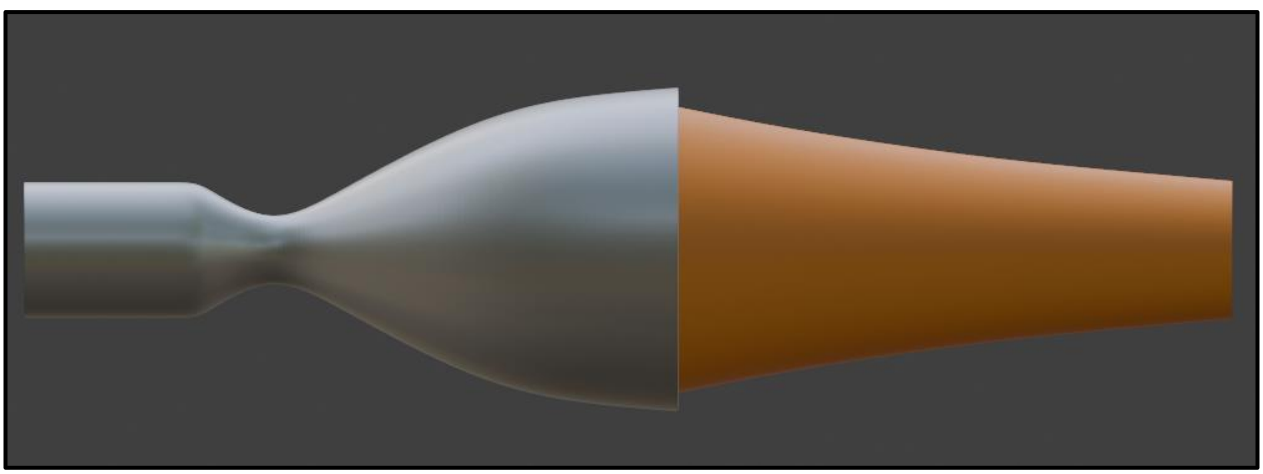

Figure 6: Over Expanded Flow

In contrast, an under expanded flow would actually have an exit pressure greater than the ambient pressure. This is in reference to the fact that the nozzle did not drop the flow's pressure enough to match the ambient pressure. The nozzle did not expand the flow enough.

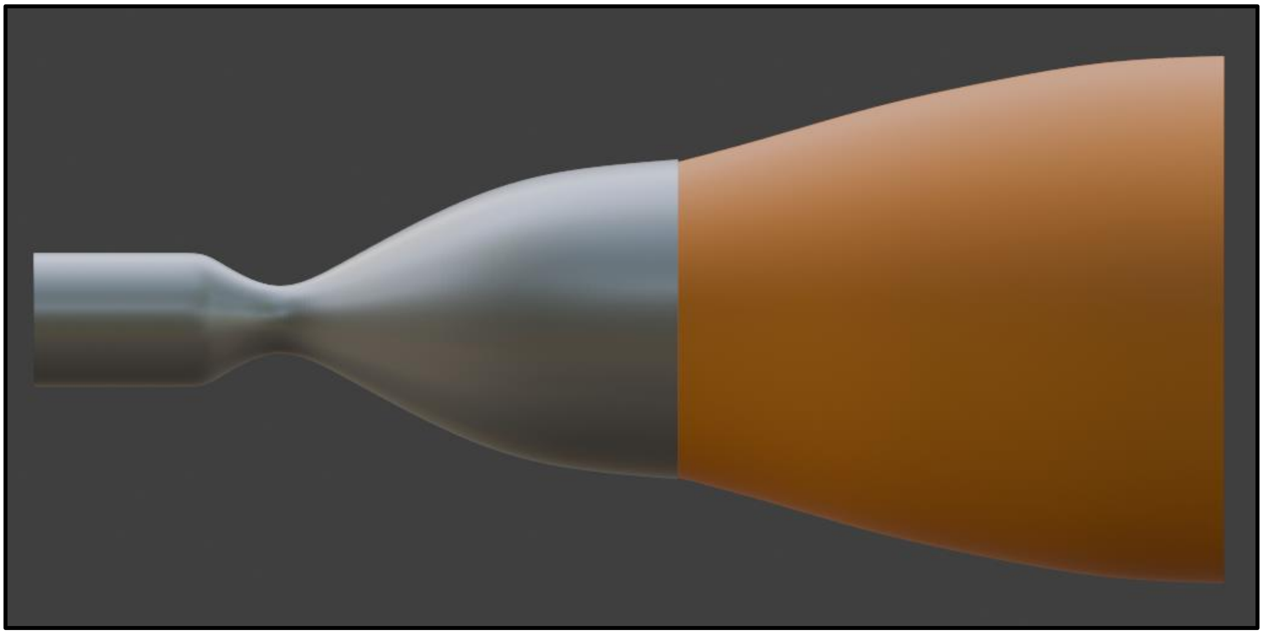

Figure 7: Under Expanded Flow

The primary concern is the over expanded case. If the exit pressure is expanded too far, the ambient pressure can push the flow back into the nozzle. This difference in pressure creates a pressure gradient along the nozzle's wall, and that gradient can lead to the flow in the nozzle separating from the nozzle's wall. This condition is called flow separation, and it was not desired in this project. The guidelines provided a method for detecting flow separation: which will be discussed later. For now, the important thing to remember is that any design that causes flow separation would be considered invalid.

Another constraint defined in the guidelines revolved around a ballast mass. The ballast was essentially extra weight added to the rocket. The ballast was useful to fine tune the maximum altitude of the sounding rocket, but it was limited to a maximum of one-pound mass. This constraint was necessary because almost any design that went past the target could be made to reach the target if enough weight was added. Instead, the ballast was meant to tune the rocket into reach plus or minus one foot from the defined target altitude.

The final constraint each design had to manage was acceleration. Now, these were not crewed flights, but the scientific instruments and the rocket itself could only handle so much force. The guidelines defined an acceleration limit of 15gs: 483 feet per second squared. So, here is a recap of all the constraints.

Table 1: Original Guideline’s Constraints

Constraint Name	Constraint Range
Ballast Mass	$0 \leq$ Ballast Mass $\leq$ 1-lbm
Acceleration	$0 \leq$ Acceleration $\leq$ 483 ft/s^2
Chamber Pressure	$0 \leq$ Pressure $\leq$ 1000 psi
Choked Flow	2*Throat Area $\leq$ Bore Area
Flow Separation	Not Allowed
Nozzle’s Area Ratio	$1 \leq$ Area Ratio
Altitude Tolerance	$-1 \leq$ (Target – Rocket’s Max) $\leq$ 1 ft
Casing Length	$0 \leq$ Casing Length $\leq$ 34 in
Grain Spacing	0.125 inch

Now, here is a list of all the potential inputs and if they can change between flights. The casing length of the rocket was also are variable value between flights, but it was automatically calculated based on the number of grains used.

Table 2: Original Guideline’s Input Variation

Constant Inputs	Variable Inputs
Grain’s Bore Radius	Nozzle’s Throat Area
Grain’s Outer Radius	Nozzle’s Area Ratio
Grain’s Length	Number of Grains Used
	Ballast Mass
	Casing Length (Automatically Calculated)

What did the Bonus Problem Changed?

The bonus problem, the problem of this capstone, made minimal changes to the project guidelines. The target tolerance was raised from plus or minus one foot to plus or minus one hundred feet. Additionally, the only variable input between tests was the number of grains used. The casing length was now also fixed to the length needed to fit the maximum number of grains used. The updated tables are below.

Table 3: New Guideline’s Constraints

Constraint Name	Constraint Range
Ballast Mass	0 ≤ Ballast Mass ≤ 1-lbm
Acceleration	0 ≤ Acceleration ≤ 483 ft/s^2
Chamber Pressure	0 ≤ Pressure ≤ 1000 psi
Choked Flow	2*Throat Area ≤ Bore Area
Flow Separation	Not Allowed
Nozzle’s Area Ratio	1 ≤ Area Ratio
Altitude Tolerance	-100 ≤ (Target – Rocket’s Max) ≤ 100 ft
Casing Length	0 ≤ Casing Length ≤ 34 in
Grain Spacing	0.125 inch

Table 4: New Guideline’s Input Variation

Constant Inputs	Variable Inputs
Grain’s Bore Radius	Number of Grains Used
Grain’s Outer Radius	
Grain’s Length	
Nozzle’s Throat Area	
Nozzle’s Area Ratio	
Ballast Mass	
Casing Length (Fixed)	

Simulation Design

My original simulation for the original final project was in MATLAB, but I decided to write this version in C. Why? I figured C would provide the performance boost necessary for batch processing, and I wanted to learn C. Yes, this was my first project in C. C has become my favorite language, but my application of it would be considered rudimentary. The program created still performed its task, and it performed better than anything I could have written in MATLAB. However, I was not able to capitalize on its full potential since I started learning the language for this project.

The highest-level view of the simulation program is a collection of structures and functions. Each structure's purpose was to either store or link data stored in variables, and each function was meant to manipulate the data stored in those structures. Linking structures were structures that indirectly held data. They were responsible for connecting either data structures or other linking structures. The best example of these was the `Simulation` structure which linked all data relevant to a single simulation: shown below.

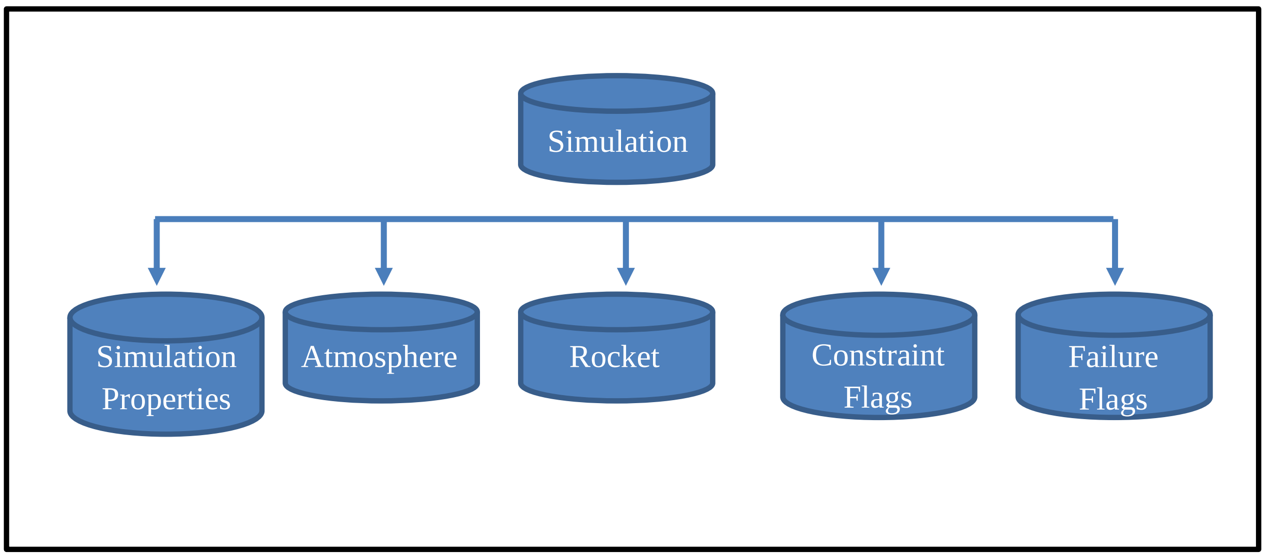

Figure 8: Simulation Linking Structure Example

The `Simulation` could be passed into the `RunSimulation()` function, and the function would perform all calculations necessary too complete the simulation. This configuration allowed for multiple instances of a simulation structure to be created, and each instance could be tested and scored on its performance. Essentially, this design made batch processing easier. The `Atmosphere` structure has been showed below to provide and example of a data storing structure.

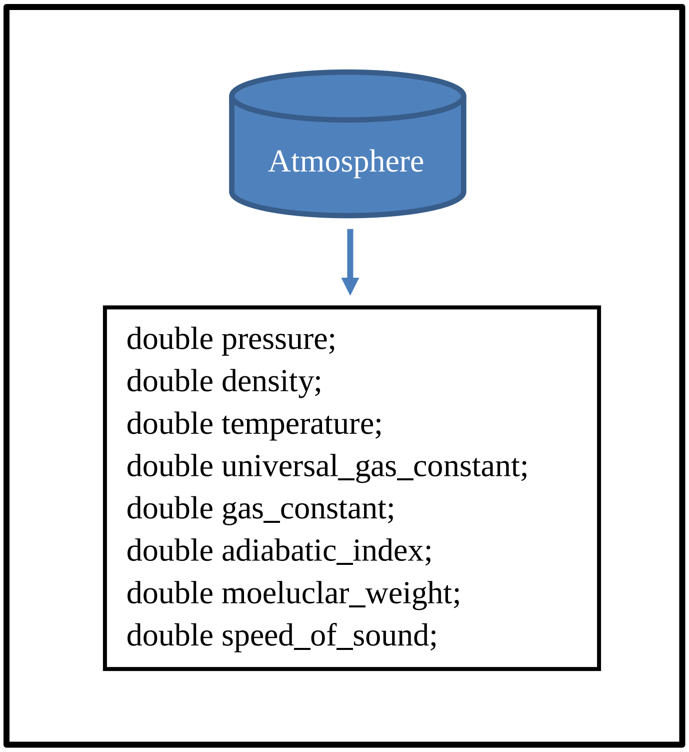

Figure 9: Atmosphere Data Structure Example

Both the linking and data carrying structures work together to pass all relevant information between functions. The structures only store the necessary data to continue the simulation; therefore, the history of data being calculated is not saved.

Atmosphere Model

The first model to be discussed is the simulations atmosphere model. The model was relatively simple, and it only required a few variables to track. The biggest contribution of the atmosphere model were the ambient pressure, density, and temperature values. The following equations were used to determine these values at any given time step.

h -- Rocket's Altitude in ft
P -- Ambient Pressure in psi
R -- Ambient Density in $\frac{lb_m}{ft^3}$
T -- Ambient Temperature in R

If h ≤ 83,000

$$\mathrm{P} = -4.272981\mathrm{E} - 14 * h^3 + 8.060081\mathrm{E} - 9 * h^2 - 5\mathrm{x}482655\mathrm{E} - 4 * \mathrm{h} + 14.69241 \tag{2A}$$

Else

$$P = 0 \tag{2B}$$

If h ≤ 82,000

$$R = 1.255E - 11 * h^2 - 1.9453E - 6 * h + 0.7579 \tag{3A}$$

Else

$$R = 0 \tag{3B}$$

If h ≤ 32,809

$$T = -0.0036 * h + 518 \tag{4A}$$

Else

$$T = 399 \tag{4B}$$

In addition to these variables, the atmosphere model calculated the gas constant of the air for calculations revolving around the speed of sound: which this model also held.

R_u -- The universal gas constant in $\frac{ft*lb_f}{lb_m mol*R}$

M -- The molecular weight of air in $\frac{lb_m}{lb_m mol}$

g -- The conversion from lb_f to lb_m for R_u

R -- The specific gas constant for air $\frac{ft^2*lb_m}{s^2*R}$

$$R = \frac{R_u}{M * g} \tag{5}$$

The speed of sound was calculated by the following equation.

R -- The specific gas constant for air in $\frac{ft^2*lb_m}{s^2*R}$

T -- Ambient Temperature in R

γ -- The adiabatic index of air

a -- The speed of sound in $\frac{ft}{s}$

$$a = \sqrt{R * T * \gamma} \tag{6}$$

Thrust Model

The thrust model used for the rocket contained the most steps. The first information needed was the parameters that describe the propellant that made up the grains used in the rocket. The following table shows these parameters.

Table 5: Propellant Parameters

Name	Symbol	Value
Characteristic Exhaust Velocity	c^*	$5210 \frac{ft}{s}$
Adiabatic Index	γ	1.25
Reference Burn Rate Constant	a_0	$0.030 \frac{in}{s} * \left(\frac{lb_f}{in^2}\right)^{-n}$
Burn Rate Exponent	n	0.35
Temperature Sensitivity	σ_p	$0.001 \frac{}{F}$
Initial Burn Temperature	T_b	$70\ F$
Reference Initial Burn Temperature	$T_{b,0}$	$70\ F$
Density	ρ_p	$0.065 \frac{lb_m}{in^3}$

These values were used to calculate the actual burn rate constant, a, through the following equation.

$$a = a_0 * \exp[\sigma_p * (T_{b,0} - T_b)] \tag{7}$$

Now, the first iterative calculation can begin: the burn area calculation. First, the grain's web needs to be discussed. The web is the distance of material that has been removed from the grain since the ignition. So, the web would be the distance the inner radius of the grain has expanded while burning, and the length of the grain would also be shrinking from either side during this process.

A_b -- The burn area of all propellant grains in in^2
N -- The number of grains
w -- The web in in
L -- The initial length of each grain in in
R_o -- The initial outer radius of each grain in in
R_i -- The initial inner radius of each grain in in

$$A_b = 2\pi * N * [(R_i + w)(L - 2 * w) + (R_o^2 - (R_i + w)^2)] \tag{8}$$

Next, the chamber pressure was calculated.

a -- The burn rate constant of the propellant in $\frac{in}{s} * \left(\frac{lb_f}{in^2}\right)^{-n}$
n -- The burn rate exponent of the propellant
A_b -- The burn area of all propellant grains in in^2
A_t -- The throat area of the nozzle in in^2
c^* -- The characteristic exhaust velocity of the propellant in $\frac{ft}{s}$
ρ_p -- The density of the propellant in $\frac{lb_m}{in^3}$
p_c -- The chamber pressure of the rocket in psi

$$p_c = \left[a * c^* * \rho_p * \frac{A_b}{A_t}\right]^{\frac{1}{1-n}} \tag{9}$$

The next calculation had to be done numerically. The goal was to find the exit pressure of the rocket's nozzle, and that required the following equation.

p_c -- The chamber pressure of the rocket in psi
γ -- The adiabatic index of the rocket's propellant
M -- The mach number of the flow at the exit of the nozzle
p_e -- The nozzle's exit pressure in psi

$$p_e = p_c * \left(1 + \frac{\gamma - 1}{2} * M^2\right)^{\frac{-\gamma}{\gamma-1}} \tag{10}$$

The issue is finding the Mach number at the exit of the nozzle. That Mach number has a relationship with the nozzle area ratio, but there is no direct way to solve the following equation for the exit Mach number.

M -- The mach number of the flow at the exit of the nozzle
γ -- The adiabatic index of the rocket's propellant
ε -- The nozzle's area ratio

$$\varepsilon = \frac{1}{M}\left[\left(1 + \frac{\gamma - 1}{2} * M^2\right)\left(\frac{\gamma + 1}{2}\right)\right]^{\frac{\gamma+1}{2(\gamma-1)}} \tag{11}$$

As a result, the Newton-Raphson Algorithm was applied to numerically solve for the Mach number based on the relative different between iterations. Once the change in the evaluation of the exit Mach number fell below a relative value of 0.0001, the algorithm would stop. With the exit Mach number, the exit pressure could be found.

The exit pressure was the final piece necessary to find the thrust coefficient of the rocket, and this is the final piece before calculating the thrust.

c_f -- The thrust coefficient
γ -- The adiabatic index of the propellant
p_e -- The exit pressure of the rocket's nozzle in psi
p_c -- The chamber pressure of the rocket in psi
p_a -- The ambient pressure of the atmosphere in psi
ε -- The nozzle's area ratio

$$c_f = \left[\left(\frac{2 * \gamma^2}{\gamma - 1}\right) * \left(\frac{2}{\gamma + 1}\right)^{\frac{\gamma+1}{\gamma-1}} * \left(1 - \left(\frac{p_e}{p_c}\right)^{\frac{\gamma-1}{\gamma}}\right)\right]^{\frac{1}{2}} + \left(\frac{p_e}{p_c} - \frac{p_a}{p_c}\right)\varepsilon \tag{12}$$

Before the thrust can be calculated, the nozzle needs to be checked for flow separation. If the thrust coefficient is less than the value calculated by the following equation, flow separation has occurred.

$c_{f,min}$ -- The minimum thrust coefficient
ε -- The nozzle's area ratio

$$c_{f,min} = -0.0445 * \ln^2(\varepsilon) + 0.5324 * \ln(\varepsilon) + 0.1843 \tag{13}$$

Now the thrust is calculated by the following equation.

T -- Rocket's thrust in lb_f
c_f -- The thrust coefficient
A_t -- The throat area of the nozzle in in^2
p_c -- The chamber pressure of the rocket in psi

$$T = c_f * p_c * A_t \tag{14}$$

Drag Model

The drag model was used to calculated the drag force acting on the rocket at every time step during its flight. The first step was to find the Mach number of the rocket.

M -- The Mach number of the rocket
a -- The speed of sound at a given altitude in $\frac{ft}{s}$
v -- The velocity of the rocket in $\frac{ft}{s}$

$$M = \frac{v}{a}$$

(15)

The Mach number was then used to estimate the drag coefficient, c_d, from the following table.

Table 6: Drag Coefficient Estimation

Mach Number Range	Drag Coefficient Equation
$0 \leq M < 0.6$	$c_d = 1.5$
$0.6 \leq M < 1.2$	$c_d = -0.12 + 0.45 * M$
$1.2 \leq M < 1.8$	$c_d = 0.76 - 0.283 * M$
$1.8 \leq M < 4.0$	$c_d = 0.311 - 0.034 * M$
$4.0 \leq M$	$c_d = 0.175$

(16)

The reference area of the rocket was found from its diameter, and that area was converted to match the units of the ambient density.

A_r -- The reference area of the rocket
R_r -- The radius of the rocket

$$A_r = \frac{1}{4} * \frac{1}{144} * \pi * R_r^2$$

(17)

The final drag calculation was as follows.

D -- The drag force in lb_m
c_d -- The drag coefficient
v -- The velocity of the rocket in $\frac{ft}{s}$
A_r -- The reference area of the rocket in ft^2
ρ -- The ambient density of the atmosphere in $\frac{lb_m}{ft^3}$

$$D = c_d * \rho * v * abs(v) * \frac{A_r}{2} \tag{18}$$

The absolute value of the rocket's velocity is used in the drag equation because it allows for the direction of the drag force to be consistent every time it is calculated. The drag force returned will always be pointed in the same direction as the rocket's velocity. Therefore, the force can be flipped every time its effects on the rocket are considered regardless of the rocket's direction.

Updating States

The states of the rocket need to be updated every iteration of the simulation's main loop. The first state considered was the web used in the thrust model. The web is supposed to update by a value of 0.01 inches every iteration by default, but it can update by less if this default value would push the web over the maximum web. The maximum web was the maximum value the web could reach before either the length of the grain became zero or the bore radius became equal to the outer radius.

$$maximum\ web = \min\left(outer\ radius - inner\ radius, \quad \frac{1}{2} * grain\ length\right) \tag{19}$$

The propellants burn rate would also be updated with each time step.

r_b -- The burn rate of the propellant in $\frac{in}{s}$
a -- The burn rate constant of the propellant in $\frac{in}{s} * \left(\frac{lb_f}{in^2}\right)^{-n}$
n -- The burn rate exponent of the propellant
p_c -- The chamber pressure of the rocket in psi

$$r_b = a * p_c^n \tag{20}$$

The time step did have a default value of 0.1, but this default value was only used after burn out of the rocket's motor. During the boost phase, the time step was found using the following equation.

r_b -- The burn rate of the propellant in $\frac{in}{s}$
Δw -- The difference between the web at the next time and the current time in in
Δt -- The difference between the next time value and the current time value in s

$$\Delta t = \frac{\Delta w}{r_b} \tag{21}$$

The mass of the rocket was then calculated. While most components of the rocket's mass remain constant, the propellant mass changed as fuel was used. The mass of the propellant was found with the following equation.

m_p -- The mass of the propellant in lb_m
N -- The number of grains
w -- The web of the propellant grain in in
ρ_p -- The density of the propellant in $\frac{lb_m}{in^3}$
R_i -- The inner radius of the propellant grain in in
R_o -- The outer radius of the propellant grain in in
L -- The length of the propellant grain in in

$$m_p = N * \pi * [R_o^2 - (R_i + w)^2] * \rho_p * (L - 2 * w) \tag{22}$$

The rocket's casing mass was found based on the length of the casing, and a constant provided by the guidelines.

$$casing\ mass = casing\ length * 0.25 \tag{23}$$

The total mass of the rocket was found with the following equation.

m_e -- The empty mass of the rocket in lb_m
m_c -- The mass of the casing in lb_m
m_b -- The mass of the ballast in lb_m
m_p -- The mass of the propellant in lb_m
m_t -- The total mass of the rocket in lb_m

$$m_t = \ m_p + m_e + m_c + m_b \tag{24}$$

Now, the three primary states of the rocket can be updated: acceleration, velocity, and position. These states were updated using Euler Integration. The side effect this integration had was an offset in indices. The acceleration state vector always lagged one iteration behind both the velocity and position states. This was because the acceleration state was defined in the first iteration, but the velocity and position states are needed during the first iteration. Therefore, the velocity and position states were initialized to zero. The maximum velocity achieved by the rocket would be at the time index right after burn out. Burn out was the time step were the maximum web was reach, and the simulation did iterate through the boost model at the maximum web. The following is the first equation: acceleration.

a -- The acceleration of the rocket at the current time in $\frac{ft}{s^2}$
T -- The thrust of the rocket at the current time in lb_f
D -- The drag force acting on the rocket at the current time in lb_m
m_t -- The total mass of the rocket at the current time in lb_m
c -- The conversion from lb_f to lb_m in $\frac{lb_m * ft}{s^2 * lb_f}$
g -- The acceleration due to gravity in $\frac{ft}{s^2}$

$$a = \frac{T * c - D}{m_t} - g \tag{25}$$

Next, velocity was found.
v_t -- The rocket's velocity at the current time in $\frac{ft}{s}$
v_{t+1} -- The rocket's velocity at the next time in $\frac{ft}{s}$
a -- The rocket's acceleration at the current time in $\frac{ft}{s^2}$
Δt -- The change in time

$$v_{t+1} = v_t + a * \Delta t \tag{26}$$

Finally, the position state was updated.

h_t -- The rocket's altitude at the current time in ft
h_{t+1} -- The rocket's altitude at the next time in ft
v_t -- The rocket's velocity at the current time in $\frac{ft}{s}$
v_{t+1} -- The rocket's velocity at the next time in $\frac{ft}{s}$
Δt -- The change in time

$$h_{t+1} = h_t + \frac{v_t + v_{t+1}}{2} * \Delta t \tag{27}$$

Batch Processing Methods

Batch processing is simply a method of testing designs in groups to find the desired result. This project relied heavily on this concept. Three different methods were used to perform batch processing: each at different levels. Together these methods provided this project with the results it obtained. Designs were not considered failures if constrain values were exceeded. Instead, designs were only considered failures if they encountered flow separated, choked flow, or their nozzle's area ratio fell below one. Any designs that did not fail were saved to the disk for easy management.

The highest-level batch processing method was simply a human guessing and checking. This method was not elaborate, but it used to human's intuition and knowledge of the subject to make massive movements across the solution space to find results. The solution space for this problem is massive, and this method was not used to narrow down a design. Instead, this method was used to get an idea on a good range of values to have the computer test.

Each design was defined by nine parameters: grain inner radius, grain outer radius, grain length, ballast mass, nozzle throat area, nozzle area ratio, first test's grain number, second test's grain number, and third test's grain number. Multiply the number of values tested for each parameter, and the result is the number of combinations that will be tested. For example, I want to test all the designs with each input having to possible values. There are nine parameters; so, that simplifies to $2^9 = 256$. That is not many designs. A human could run through those in a semester. However, if each input was increase to just ten possible values: $10^9 = 10{,}000{,}000{,}000$. This gets out of hand fast; the computer can deal with it.

The second method for batch processing was a brute force method. Simply, the computer was given a range of values for each parameter and told to run a test with every combination. This method was still not the icon of efficiency, but it never missed a valid design. If there was a valid design in the range provided, the computer would find it. The efficiency of this method varied based on how many valid designs were found in its range. The simulation was designed to return early if any of the three failure modes were encountered. This prevented wasting time on calculations that would provide an invalid result anyways. Either way, this method could run through about 2,000,000 to 10,000,000 designs every hour.

The final method was to treat the solution space like a group of nodes. Each node was a design, and it was connected to eighteen other nodes. That is one connection per input variation: nine inputs in two directions each. The process was provided one valid node to start, and it would expand that node. The demonstrations for this only use four connections per node for simplicity. In the following figures, green nodes are designs that were tested and worked, red nodes are designs that failed, and blue nodes are known but not tested designs.

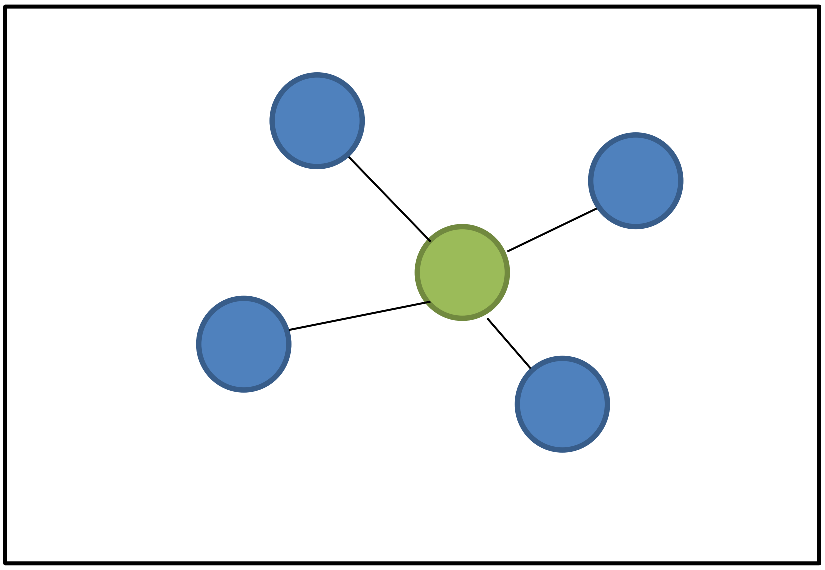

Figure 10: Expansion of a Single Design Node

As designs were tested, they were also scored. This allowed the algorithm to sort each design based on its score, and the algorithm would expand the highest scoring designs first. The designs score was based on the standard percent different equation: applied to the constraint and observed values.

$$score\ weight = \frac{(constraint - observation)}{constraint} * 100 \tag{28}$$

The percent difference acted as weight: adjusting the maximum score values for each criterion. The most preferred of these was the maximum altitude; it was weighted the highest and saw the algorithm heavily pursue designs that made it inside the altitude tolerance for all three tests. After more expansion, the batch of designs would look something like the following figure.

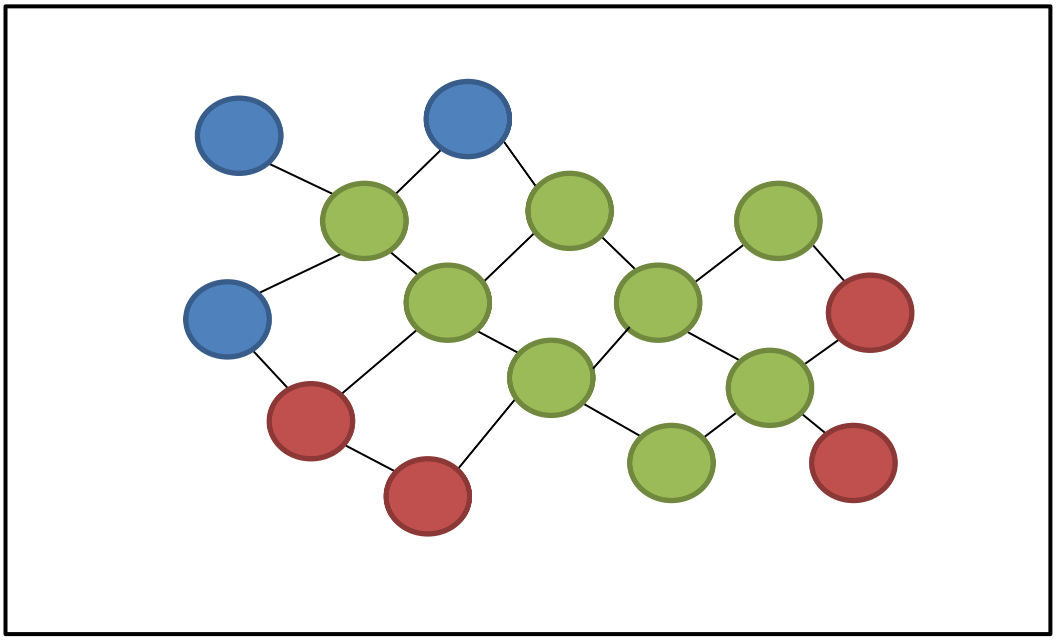

Figure 11: Expanded Solution Space Example

The above figure shows how the algorithm would expand the solution space in the direction that it believed a solution existed. However, the problem with this method also begins to show. What if all the designs the algorithm can expand to are invalid? Well, the algorithm would simply stop; it would state it ran out of designs. The problem was not that the algorithm ran out of designs. The problem was that this proved that solutions existed in groups. This method could only find a solution if that solution was in the same groups as the starting design. This was why three methods were used in this project. A human would quickly identify a region of the solution space that showed promise. The brute force method would identify potential starting nodes within a portion of that region. Then, the pathfinding method would work its way from that starting design towards a potential solution. These were the methods used when trying to find a solution.

Why is This Problem so Difficult?

Why was this problem so difficult? The difficulty arises from lack of adaptability. Each of the three tests, five, ten, and fifteen thousand feet, have ever so slightly different preferences. The 5,000-foot test prefers a small throat area; it allows that design to reach higher with fewer grains and less propellent. However, the 15,000-foot test prefers a larger throat area; this allows the design to reduce the maximum chamber pressure and acceleration experienced during the 15,000-foot test. There are multiple balancing acts that would be need to be found for a design to reach within 100 feet of each target altitude.

Consider the following base design.

Table 7: Analysis Base Design

Input	Value
Grain Inner Radius	0.800
Grain Outer Radius	1.800
Grain Length	4.300
Ballast Mass	0.000
Nozzle Throat Area	0.800
Nozzle Area Ratio	4.000
15,000 Test Grain Number	4
10,000 Test Grain Number	3
5,000 Test Grain Number	2

Here is how this design performed.

Table 8: Base Design Results

Test	15,000	10,000	5,000
Maximum Altitude (ft)	15,991.086	11,045.831	4091.983
Maximum Acceleration (ft/s^2)	362.253	222.397	96.493
Maximum Chamber Pressure (psi)	542.199	348.294	186,668

The table above displays a common issue with this problem; two designs are on one side of the target and one is on the opposite side. In this example, the 5,000-foot test needs to raise its maximum altitude, but the 15,000- and 10,000-foot tests need to lower theirs. What happens when the area ratio is lowered to 3?

Table 9: Base Design with Area Ratio = 3

Test	15,000	10,000	5,000
Maximum Altitude (ft)	15,781.579	11,025.837	4389.770
Maximum Acceleration (ft/s^2)	356.206	221.036	99.358
Maximum Chamber Pressure (psi)	542.199	348.294	186,668

The results were desired. Each test moved in the desired direction, maximum acceleration slightly rose, and the chamber pressure remained constant. What about an area ratio of 2?

Table 10: Base Design with Area Ratio = 2

Test	15,000	10,000	5,000
Maximum Altitude (ft)	15,158.129	10,579.599	4428.255
Maximum Acceleration (ft/s^2)	341.007	213.560	98.813
Maximum Chamber Pressure (psi)	542.199	348.294	186,668

The altitudes are still moving in the right direction, and the chamber pressure remains constant. The chamber pressure is determined by the grain geomtery and area ratio; so, it will not change unless those variables are changed. However, the acceleration has started to fall. This would be good, but it is also concerning. Is the problem going to flip? What about an area ratio of 1.5?

Table 11: Base Design with Area Ratio = 1.5

Test	15,000	10,000	5,000
Maximum Altitude (ft)	14,409.657	9,932.432	4,199.813
Maximum Acceleration (ft/s^2)	324.828	204.100	95.353
Maximum Chamber Pressure (psi)	542.199	348.294	186,668

Now all three designs are below their target is there a solution in this space. This would be the desired position. Each test is linked. A change that alters one test will alter each of the other test in the same direction but with a different magnitude. The last statement is not always true; raising the area ratio too high can cause the 5,000-foot test to plummet while the other two rise. However, that statement is a reasonable approximation of the problem as a solution is approached. What about 1.6 for that area ratio?

Table 12: Base Design with Area Ratio = 1.6

Test	15,000	10,000	5,000
Maximum Altitude (ft)	14,607.431	10,106.562	4270.093
Maximum Acceleration (ft/s^2)	328.922	206.563	96.364
Maximum Chamber Pressure (psi)	542.199	348.294	186,668

The original problem has returned. Two designs are on one side of the target, and one design is on the opposite side. This pattern encompassed this project, and it was present when varying each input. More variation in the grain numbers were required to solve this issue. The grain number in each test was the only way to effectively move two designs in different directions in this scenario. However, that control does not outpace the gain in acceleration. By adding more grains, designs that only go 4,000 feet up but have over 700 ft/s^2 for maximum acceleration and 1150 psi for maximum chamber pressure; both of these values are already over their respective constraints. Additionally, designs like this become common with more grain numbers. More viable designs were found with less grain.

At this point, over 300 hundred million designs have been tested for this project. There are many designs that have gotten extremely close, but the before mentioned issue remains. Two designs end up on one side of the target, and the other is on the opposite. This challenge is extremely difficult because there was no apparent way to adjust the height of a single test without affecting the other tests in the same direction. Every combination of two successes and one failure were found during this project. Below is a table showing the common patterns encountered.

Table 13: Common Outcomes

15,000 Test	10,000 Test	5,000 Test
Too Low	Passed	Passed
Passed	Passed	Too Low
Passed	Too High	Passed

Here is an example of the first combination. The S stands for success and the C is for a constraint violation, altitude in all these cases, and the test are placed in descending order from left to right.

Table 14: CSS Design

Input	Value
Grain Inner Radius	1.100
Grain Outer Radius	2.100
Grain Length	2.900
Ballast Mass	1.000
Nozzle Throat Area	1.590
Nozzle Area Ratio	1.480
15,000 Test Grain Number	5
10,000 Test Grain Number	4
5,000 Test Grain Number	3

Table 15: CSS Results

Test	15,000	10,000	5,000
Maximum Altitude (ft)	14210.294	10083.912	4999.403
Maximum Acceleration (ft/s^2)	326.583	225.364	131.575
Maximum Chamber Pressure (psi)	289.703	205.523	132.022

The second combination is next.

Table 16: SSC Design

Input	Value
Grain Inner Radius	1.139
Grain Outer Radius	2.211
Grain Length	3.440
Ballast Mass	0.562
Nozzle Throat Area	1.355
Nozzle Area Ratio	1.201
15,000 Test Grain Number	4
10,000 Test Grain Number	3
5,000 Test Grain Number	2

Table 17: SSC Results

Test	15,000	10,000	5,000
Maximum Altitude (ft)	14980.102	9964.229	3644.712
Maximum Acceleration (ft/s^2)	311.155	193.855	87.929
Maximum Chamber Pressure (psi)	336.225	215.981	115.746

The third combination is listed below.

Table 18: SCS Design

Input	Value
Grain Inner Radius	1.113
Grain Outer Radius	2.061
Grain Length	3.196
Ballast Mass	1.000
Nozzle Throat Area	1.800
Nozzle Area Ratio	1.900
15,000 Test Grain Number	5
10,000 Test Grain Number	4
5,000 Test Grain Number	3

Table 19: SCS Results

Test	15,000	10,000	5,000
Maximum Altitude (ft)	14999.067	10622.281	4999.528
Maximum Acceleration (ft/s^2)	320.313	218.277	123.553
Maximum Chamber Pressure (psi)	249.593	177.068	113.743

The results above show the frustration that surrounded this project every time a batch finished processing. The SCS result even has two tests within the original one-foot tolerance, but the 10,000-foot test was two high. Out of all these combinations the SCS results should the greatest promise of finding a solution. The biggest problem appears to be the spacing of the altitudes. To fix this more grains are needed to more variation can be applied between tests, but this idea quickly changes the issue to the acceleration and pressure values. These values grow too fast when grains are added.

Was a Solution Found?

Was a solution found? No. No design was found that could pass all three tests, but one design did come close. How close? One design got within 500 feet of the 100-foot tolerance and passing all three tests.

Table 20: Closest Design

Input	Value
Grain Inner Radius	1.120
Grain Outer Radius	2.070
Grain Length	3.170
Ballast Mass	1.000
Nozzle Throat Area	1.800
Nozzle Area Ratio	1.810
15,000 Test Grain Number	5
10,000 Test Grain Number	4
5,000 Test Grain Number	3

Table 21: Closest Result

Test	15,000	10,000	5,000
Maximum Altitude (ft)	14930.579	10590.830	5032.263
Maximum Acceleration (ft/s^2)	320.259	218.649	124.328
Maximum Chamber Pressure (psi)	250.459	177.682	114.138

Even with this result, the problem still shows itself. Not having all the maximum altitudes on the same side of their target makes this problem extremely difficult. Decreasing one design will decrease all the design. Trying to avoid drop two designs by less than 100 feet and one design by almost 500 felt near impossible. Of course, there is likely a solution created after adding an unreasonable amount of grains, but the acceleration and pressure values make it more practical to fire these rockets out of a railgun. There may be a solution out there, but this was my best attempt.

Conclusion

Overall, this project was extremely fun. It helped me further my understanding of rocket propulsion and programming. I still wonder if there is a solution, but I am ready to pass that challenge onto the next in line. Until then, these are my recommendations for adjusting the project guidelines to create at least one solution.

My analysis shows that the biggest issue was the spacing of the target altitudes. The are nicely spaced at a consistent five thousand feet, but that was the problem. The current altitudes appeared to be a bit outside a single design is able to cover by only adjusting grains. If the goal is the guarantee a solution, changing the 10,000-foot target to 10,500 would be viable. This guarantees at least one, in fact the only one so far, solution. However, too make the bonus still challenging but more possible one of two solutions are recommended. Either raise the 10,000-foot target to 11,000 feet, or lower the 15,000-foot target to 14,000 feet. Both of these solutions open a sizable amount of solutions that are still challenging to find. The 5,000-foot target should remain were it is because the solutions that would require it to move would see this target lowered by over 1,000 feet. These are my recommendations for adjusting the project guidelines to guarantee solutions to the bonus problem: without making it too easy.