DEV LRE Progress Update and Technical Issues

Project (formal) Start: 20-Aug-2024

Date:14-Jan-2025

Table of Contents

Slide 12 → Project Definition, Goal, Scope, Requirements,

Slide 18 → Preliminary Engine Design and high level decisions

Slide 35 → Engine Modeling, Validation Sensitivity Analysis

Slide 60 → Investigating and improving the Injector Model

Slide 85 → Issues Integrating the Injector Model

Slide 96 → Oxidizer Tank Modeling (this is specifically where I present the issue in my model that is the scope of this presentation).

Repository Link:
https://github.com/rwright403/king_hybrid

Disclaimers:

1. Output graphs from my program have bad scaling. Differences are larger than they appear
2. I have not taken the numerical methods course at UVIC yet, my apologies if the error is something extremely basic (although I hope this is the case).

Acronyms

LRE - Liquid Rocket Engine

LC - Launch Canada: https://www.launchcanada.org/

O/F - Oxidizer to Fuel Ratio

SPI - Single Phase Incompressible (injector model type)

HEM - Homogeneous Equilibrium Model (injector model type)

High Level Project Summary

- I am designing a liquid rocket engine. This design process includes
  - Mission Definition and goal, objectives, constraints and requirements
  - Propellant and high level system design decisions
  - System Sizing
  - Analysis and Characterization
- I am currently in the analysis and characterization of my engine. I have been developing my own in-house code based on research; I have hit a roadblock and I am unsure of how to proceed.

Presentation Goal:

- The goal of this slide deck is to present a technical roadblock I have encountered attempting to create a model for use in the design of my LRE personal project. I am hoping that the reader will be able to help me fix the problem, or help me think of next steps to continue towards making this model work.
- To accomplish this, I have created this slide deck to provide context to allow the reader to fully understand:
  - The problem I am facing with the implementation of my engine model (This starts on slide 95).
  - Why it is important to solve this specific problem. (Why a more simple but less accurate model is not feasible under the objectives, constraints and requirements of this project).
- To accomplish this, I have included an overview of some relevant theory, debugging and my entire development path in chronological order.

Presentation Goal:

- As a secondary goal, I would appreciate feedback on the development path I have taken on this engine and anything else that comes to the reader’s mind about my work.
  - Are there more efficient or effective ways I have not thought of to proceed with this project?

Things to keep in mind - what makes a good engine?

From my perspective, a good engine:

- Meets project requirements
- Well characterized
- Produces stable combustion

Key References - in Readme/Readings Package: https://github.com/rwright403/king_hybrid

	Source	Contribution
[1]	NASA Chemical Equilibrium with Applications (CEA)	Used with the rocketcea library for combustion calculations
[2]	Ben Klammer, Hybrid Modelling:	Translated equilibrium model used in this thesis into python and used it as a starting point for analysis
[3]	Benjamin S. Waxman, Jonah E. Zimmerman, Brian J. Cantwell, Mass Flow Rate and Isolation Characteristics of Injectors for Use with Self-Pressurizing Oxidizers in Hybrid Rockets	Used to implement HEM and Dyer injector models, experimental data to validate Emerson + Mohammad injector model and understand nitrous physics.
[4]	Numerical Modeling of Pressurization of a Propellant Tank	Referenced for setting up the pressurized fuel tank model, although my model is much more simple at this stage
[5]	Emerson Vargas Niño, Mohammad Reza H. Razavi, Design of Two-Phase Injectors Using Analytical and Numerical Methods with Application to Hybrid Rockets	Used to implement Emerson + Mohammad injector model
[6]	Tomasz Palacz, Jacek Cieslik. Experimental Study on the Mass Flow Rate of the Self Pressurizing Propellants in the Rocket Injector	Used experimental data to build a better understanding of model feed system
[7]	Zilliac, Gregory & Karabeyoglu, Arif. (2005). Modeling of Propellant Tank Pressurization.	Used to setup non-equilibrium tank model (1/2)
[8]	Review and Evaluation of Models for Self-Pressurizing Propellant Tank Dynamics, Jonah E. Zimmerman, Benjamin S. Waxman, Brian Cantwell and Greg Zilliac	Used to setup non-equilibrium tank model (2/2)

Key References - in Readme/Readings Package: https://github.com/rwright403/king_hybrid

	Source	Contribution
[9]	Michigan Aeronautical Science Association Liquid Bi-Propellant Rocket Team 37 Project Technical Report for the 2018 IREC	Validation Case
[10]	Technical report for Stanford SSI's rocket Project Olympus for the Spaceport America Cup Competition 2022	Validation Case
[11]	Two-phase flow through pressure safety valves. Experimental investigation and model prediction	Used to understand the Emerson + Mohammad model
[12]	Effects of Injector Design and Impingement Techniques on the Atomization of Self-Pressurizing Oxidizers, Benjamin S. Waxman, Brian J. Cantwell, Greg Zilliac	By the same authors and part of the same work as [3] - complimentary

Project Definition

Project Goal / Scope

- Goal: I want to develop a well characterized minimum viable product test bed liquid rocket engine that can be further developed into an engine for a 10k ft apogee competition sounding rocket.
  - This means that the engine should produce thrust sufficient for this mission without having requirements like weight savings, component packaging and flight vehicle integration.
  - Flight engines may have less instrumentation than testbed engines due to flight/vehicle constraints, which may not align with my goal of developing a well characterized engine.
- Scope: The entire engine development cycle from mission definition, analysis, design, manufacture, testing and design iteration.
  - A flight engine and vehicle is outside of the scope of this project. However, I want to document everything well enough that I could leave that door open for myself or one of my friends on the rocketry team to take it in a direction of their choice in the future.

Disclaimer Regarding Testing

It is important to acknowledge that engine testing is extremely dangerous. However, many student groups including the UVic Rocketry Team have demonstrated the ability to **safely and legally test**.

- My decision to build a well characterized engine and follow the testing requirements outlined in the Launch Canada DETG are extremely impactful in mitigating test risk.
- Ultimately, tests and procedures will be designed to account for the worst possible outcome and mitigate risks.
- Additionally, I have built a small network of people with industry experience testing propulsion systems and I plan to review test plans with them.

Project Requirements:

- Design must follow the requirements found in the Launch Canada Design Test and Evaluation Guide (DTEG) which at a high level include:
  - Requirements for selecting non-toxic propellants
  - Sets design standards for pressurized fluid systems
  - Testing
  - CONOPS / Failsafe requirements

Project Constraints:

- Not a club project. Therefore, I do not have UVIC Mech Shop access to machine parts for this project.
- Self-funded.
- Driven by one inexperienced person with part time work hours.
    - I think this is worth mentioning because there are a lot of things I know I have neglected, but justified this with the limited work hours available for this project.

Notes on Project Goals, Requirements and Constraints

- Because the project is self-funded, I want to de-risk as much of the engine as possible through component testing and modelling before building primary engine hardware.
  - This also aligns with my goal of creating a well understood engine.
- Although this isn't a Launch Canada project, the Launch Canada DTEG seemed like a good place to find requirements that prioritized good practices and safety for the scale of engine I am making. Also aligns with the goal of allowing this work to continue into a flight engine in the future.
- The goal of allowing this work to continue into a future flight engine is so I can pass it along to my friends in UVR or leave it on the drive for someone to find and learn from. Unfortunately, a liquid rocket engine does not align with the team's current goals. However, I learned a lot of what I know about propulsion from source [2], which I found on the drive.
- I have reflected on what I want out of this project and I realised I would rather end this project before building hardware if I am not certain that I can build something that I understand well.

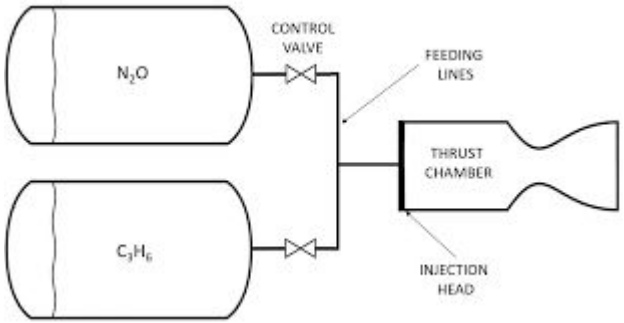
Preliminary Engine Design

Preliminary Engine Design

- The preliminary engine design phase consists of:
  - Trade-off studies between different cycles and propellants to create the overall engine concept.
  - High level calculations to estimate performance and size of the engine.

Determine Engine Cycle

- The first decision to make is the engine's cycle. Under the goals of making a minimum viable product engine, I chose a pressure fed engine cycle. This allows us to avoid turbomachinery for a mechanically simple system.
  - I considered this choice of cycle to dominate all the other ones, so I did not make a formal weighted criteria chart.



Propellant Selection Criteria

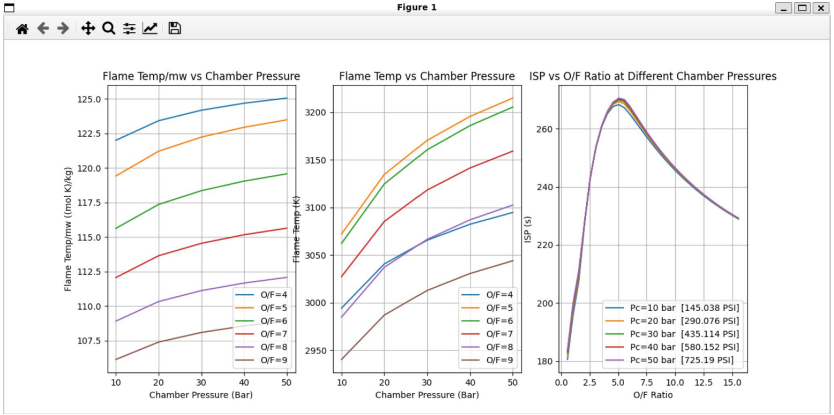
- I created a weighted criteria chart that aligned with project objectives, requirements and constraints to select my propellants.
- Propellant obtainability and non-toxic (defined by Launch Canada DETG requirements) are pass/fail criteria.
- Mechanical simplicity is weighted the highest, with storability next highest, and cost and performance with the lowest rating.

Oxidizer Selection and Nitrous Oxide Physics

- I started by looking at oxidizers first because there are fewer options under the Launch Canada DETG.
- Nitrous Oxide has a high vapor pressure over atmospheric conditions. This allows it to self-pressurize itself, and it has the ability to be pressure fed without an external pressurant gas. This increases the mechanical simplicity of the design.
- The benefit of mechanical simplicity comes at the cost of handling the complicated physics of nitrous oxide. Nitrous oxide has a critical point of 36.37 degrees celsius and 7.245 MPa, which is very close to our operating regime.
- For Non-equilibrium calculations, we also need to work with Nitrous Oxide in metastable states. (This requires working with low level equations of state)

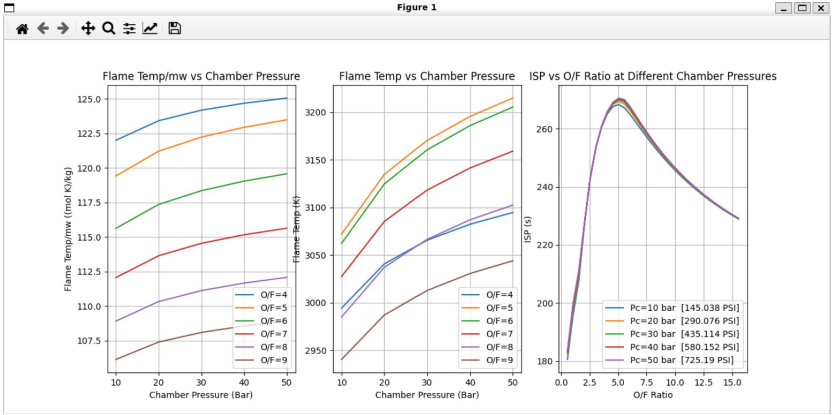
Fuel Selection Method

- I used NASA's CEA [1] to calculate equilibrium combustion of different propellants under different mixture ratios and chamber pressures to compare the theoretical performance of different propellants. I chose ethanol and nitrous oxide as my propellants from the following graph.



Fuel Selection Findings

- I found that the flame temperature of ethanol and nitrous is reasonably insensitive to small changes in chamber pressure and O/F ratio.
  - I believe this will help improve the chances of having a stable engine as small changes in O/F ratio and chamber pressure will not result in large changes to performance.
  - This may make it more challenging to analyse test results and compare models if there are a wide range of inputs or conditions that can produce similar outputs or measurable data.



High Level Design Decisions: Fuel and Oxidizer Injector

- Recommended to use a choked flow injector to isolate potential combustion instabilities from the feed system.
  - Rationale: The cost of the higher pressure drop on performance to eliminate instabilities from propagating up the feed system past the injector at the advantage of improving the certainty of the engine matching the simulation.
  - Since this is my first engine, there is a lot I don’t know about modeling, instrumentation etc and I do not have a good sense of propulsion systems, so I think it makes sense to try to avoid this problem right now.
  - Additionally, I think a lot of problems with combustion instabilities are found and worked out through testing, which has a large cost in time and money and is a challenge with my resources.
- Planning to experimentally test injector orifices to correlate the L/D to pressure drop.
  - I could not find a satisfactory correlation for this so I will use the engine model to find a satisfactory design with a pressure drop, look at atomization and manufacturability to find the diameter and then test to find a length that ensures choked flow throughout the entire liquid phase.

High Level Design Decisions: Feed System

- Feed System: Since nitrous oxide is near the critical point at atmospheric temperatures, small changes in temperature result in large changes in pressure. Therefore, it is common to use active heating devices to heat the tank to increase the pressure. I would like to avoid using these devices if possible to reduce risk because:
  - Uneven heat distribution from electric heating solutions can lead to thermal decomposition. Other solutions like using a heated water bath for uniform heat distribution are logistically impractical to bring to the test site.
  - When nitrous oxide is heated past the critical point, it becomes a supercritical fluid, where it is less stable. Additionally, the fluid is in a different regime where it would behave differently and I have decided that the engine will not be designed for this condition (one example is the loss of self-pressurization).
- The result of this decision is the nitrous oxide feed system being subjected to the temperature of the surrounding environment.

High Level Design Decisions: Fuel Tank Pressurization

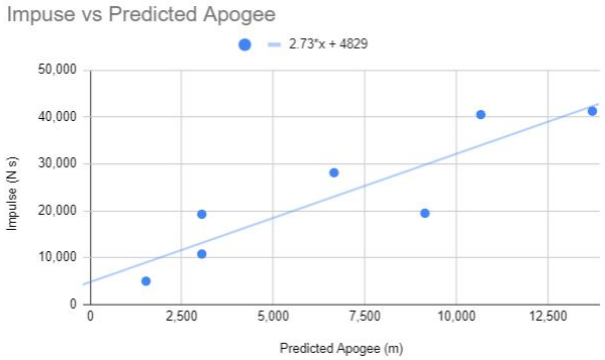
- The fuel tank will be pressurized by nitrogen.
- I have decided to directly pressurize the ethanol tank
- Honestly, I have not put a lot of time into designing this component at this stage. I am confident enough that directly pressurizing the tank is feasible to proceed with the rest of the design. I plan to look into this more at a lower level after I fix the current tank model issue that is the focus of the presentation. I would appreciate feedback on this design decision though.

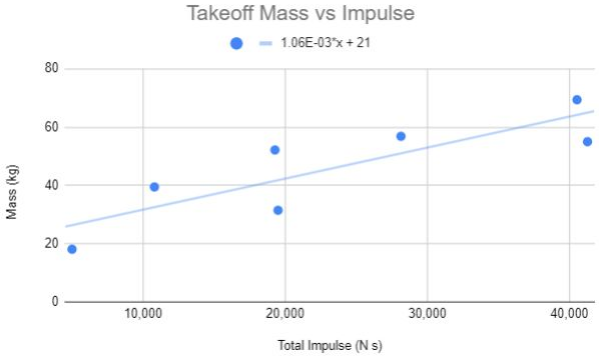
High Level Design Decisions: Cooling

- It is important to recognize that I don’t have much experience with heat transfer. From a literature review of similar sized engines, I think the engine’s short burn time would allow it to be cooled with an ablative material. I plan to explore this in greater detail later in the design cycle when I have a satisfactory thrust curve of my engine.

Estimate Engine Performance Requirements

- Our mission is to create an engine with the performance to power a 10k ft sounding rocket.
  - I estimated the required performance with a simple linear regression and reported engine specs from similar engines.
  - Additionally, to achieve a minimum off the rail stability, estimated the minimum off the launch rail thrust to weight ratio should be 11

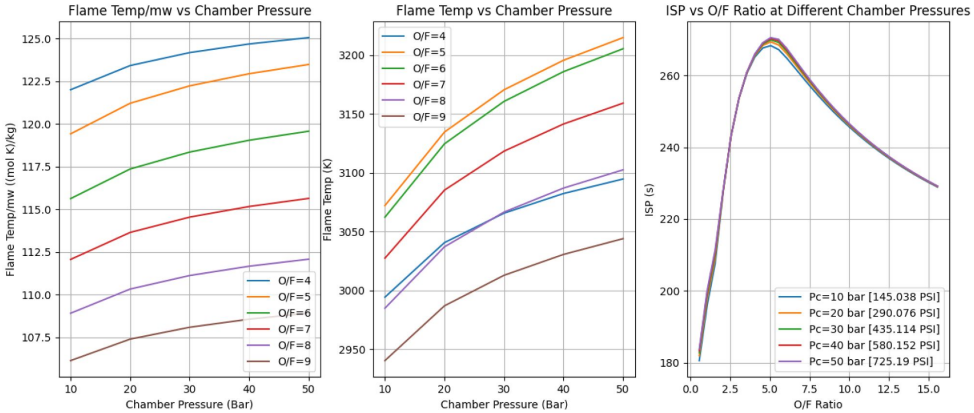




Engine Prelim Performance Estimation Summary:		
Apogee	3048	m
Est Total Impulse	13150.04	N s
Est Vehicle Mass	34.9390424	kg

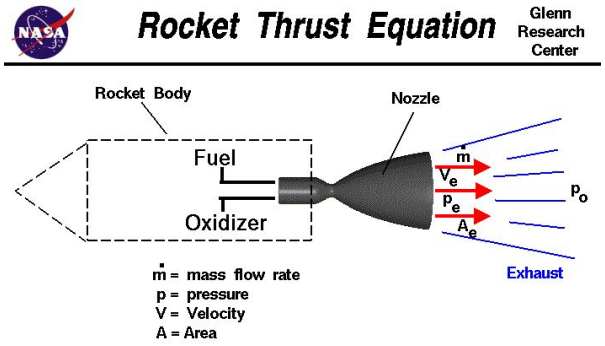
First Pass Engine Sizing

- Simple steady state “spreadsheet/textbook” calculations
- Plotted performance desvars to build an understanding of the system:
  - Highest Flame Temp/Molecular Weight → highest fuel efficiency
  - Stoichiometric O/F ratio → Maximum flame temperature → maximum thrust
    - 5.73 for Nitrous/Ethanol Combustion

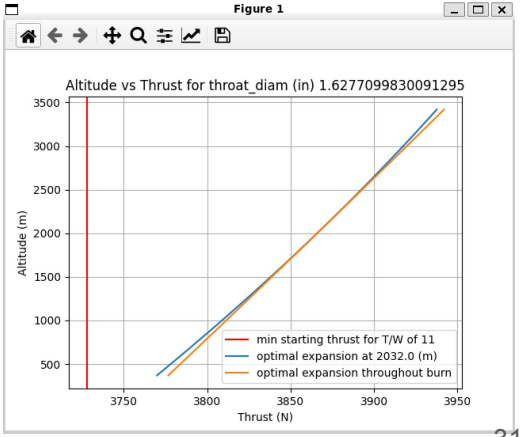


First Pass Nozzle Sizing

- Two terms in thrust equation:
  - Pressure Thrust: (p_e - p_0)*A_e
  - Momentum Thrust: m_dot*v_e
- For a liquid engine fired near sea level, pressure thrust term is negligible compared to the momentum thrust term.
  - Do not want to maximize exit pressure for our this system. A high exit pressure would mean there was energy from the exhaust gas that was not converted into gas exit velocity.
- Plotted altitude vs thrust to find a first pass design point with a starting T/W ratio of over 11. Since we do not change altitude much, we can achieve very a very optimally expanded engine.

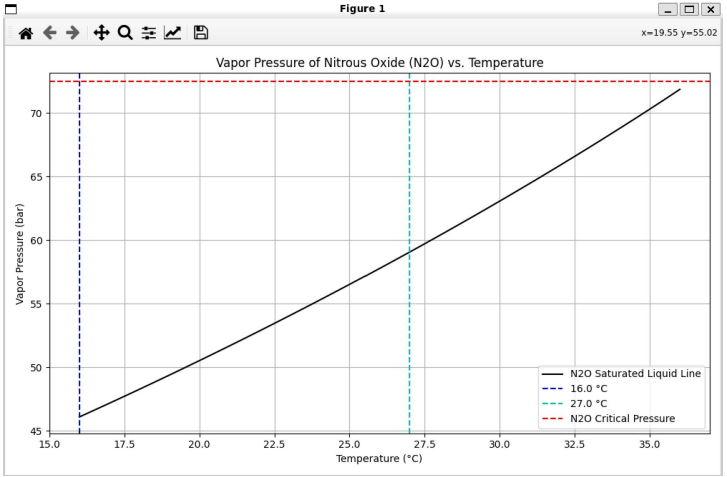


Thrust = $F = \dot{m} V_e + (p_e - p_0) A_e$



Reflecting on the impact of passively heated nitrous oxide

- Nitrous oxide tank pressure is a function of atmospheric temperature.
  - Since atmospheric temperatures are so close to the critical temperature of nitrous oxide, small temperature changes will result in large pressure changes.
    - The upstream pressure has a huge impact on the rest of the engine as a system. (flow choking, combustion, mass flow rate)
- Approach:
  - Looked at weather data to create a preliminary engine operating envelope
  - Will reflect on this envelope during the injector detail design phase.
- Rationale:
  - To design a successful engine, I need to know the conditions where I can successfully operate it. This preliminary envelope will be refined later in the design cycle once the models are setup so I can quantify the performance loss and ensure the engine is capable of safely firing as the tank pressure changes.



First Pass Sizing

- Conservatively chose a low chamber pressure:
  - Minimize pressure loads on combustion chamber.
  - At this stage I don’t have a good understanding of how an increase in pressure will impact heat transfer.
  - We also want to ensure choked flow, but at this stage we do not have a good understanding of the pressure drop that will cause flow choking so we will pick this as our initial guess.
- Acknowledging: I do not have a complete understanding of the tradeoff between the thrust advantages to a high flame temperature and the structural challenges that come with it. At this point, I think the most effective path forward will be to reflect on this choice once I start exploring the design space of the combustion chamber in more detail later in the design cycle.

First Pass Sizing Results:

```
------------
Preliminary Design Summary:
------------
Performance:
------------
Specific Impulse: 261.1889579886951 (s)
Characteristic Velocity: 1576.2621778001946(m/s)
Combustion Chamber:
------------
P_cc: 2000000.0 (Pa)
O/F Ratio: 5.0
Flame Temperature: 3134.737147511508
Ratio of Specific Heats: 1.1416517636190735
Reactant Gas Const. 321.4536775324903 (J/(kg K))
Total Impulse: 13150.039999999999 (N s)
Average Thrust 3937.6615454591415 (N)
Mass Flow Rate 1.700708690430555 (kg/s)
Exit Velocity: 2315.30630002383 (m/s)
Burn Time 3.4 (s)
Expansion Ratio: 4.100299920509235
Throat Diameter 1.6277099830091295 (in)

Feed System:
------------
Estimated Mass Fraction: 0.2657741984
Fuel Mass: 1.5301746244209624 (kg)
Oxidizer Mass: 7.650873122104811 (kg)
Oxidizer Tank Pressure: 5200000.0 (Pa)
Fuel Tank Pressure: 5200000.0 (Pa)
Preliminary Injector Solved by Assuming Fuel and Oxidizer Orifices Follow SPI Model and a Cd guess of 0.6
Oxidizer Injector Discharge Area: 3.090605599220321e-05 (m^2), Fuel Injector Discharge Area 6.566075013100621e-06 (m^2)
------------
```

Engine Modeling

Engine Modeling

- Goal: Simulate the engine's thrust and system pressures over time so we can:
  - Create engine loads with a high degree of accuracy for use in detail component design. (we want this to be accurate to prevent Garbage in Garbage out).
  - Model different engine configurations to find an effective system
  - Model to make decisions about engine operation (timing, operating temperatures)
  - Secondary goal: Create an understanding of the design space.
- Challenges
  - Commercial software like RPA lacked the capability to model this type of pressurized system.
  - Other models like NASA's Generalized Fluid System Simulation Program (GFSSP) are difficult to obtain for students outside of American Universities. (Although papers outlining the theory are publicly available for manual implementation)
  - Later in the process, I looked into using the Flownex commercial code, but I found that it did not model two phase flow well.

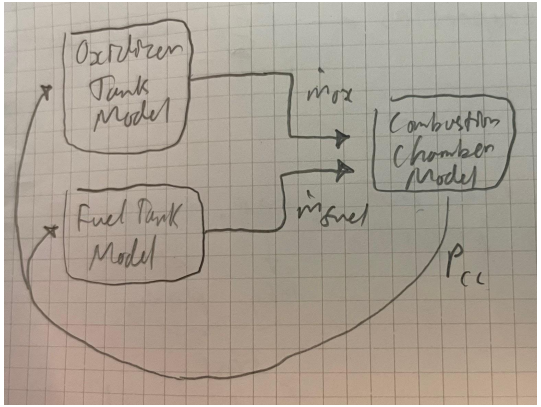
Engine Modeling

I chose to build a python model to characterize my engine because:

- Many open source libraries exist in python so I don’t need to spend as much time building low level features.
- It allows me to reuse previous UVic Rocketry work (oxidizer tank model) and potentially save time. Created a simple fuel tank and combustion chamber model that were quick to make to quickly get results and understand the design space.

Python Model Overview

- The liquid rocket engine is setup as a system of weakly-coupled models:
  - Oxidizer tank and oxidizer injector
  - Fuel tank and fuel injector
  - Combustion chamber and nozzle
- The mass flow rates from the fuel and oxidizer models are fed into the combustion chamber model and used to calculate the chamber pressure, which is fed back into the oxidizer and fuel tank models:
  - Combustion chamber pressure impacts tank mass flow rate, which impacts the O/F ratio of combustion, and thus the resulting combustion chamber pressure



Oxidizer Tank Model Overview

- From Ben Klammer's Mech 498 Thesis [2]
- Quasi-Steady State, Adiabatic Tank
- Liquid and vapor phases are in equilibrium (saturated state)
- Assuming the nitrous oxide in the injector can be modeled as Single Phase Incompressible (SPI model) where 100% of the liquid drains before any vapor drains.
  - ***In reality, there is more nuance to the phase of nitrous oxide!

Nitrous oxide in injector physics

- Source [3]-An investigation of injectors for use with high vapor pressure propellants with applications to hybrid rockets provides an extremely detailed and useful background on nitrous physics.

Nitrous oxide in injector physics

- For nitrous oxide as a self-pressurant, we are operating very close to the vapor pressure. Therefore, it is possible for the pressure of nitrous oxide in the injector to fall below the vapor pressure and cavitate in the orifice and become two-phase flow.

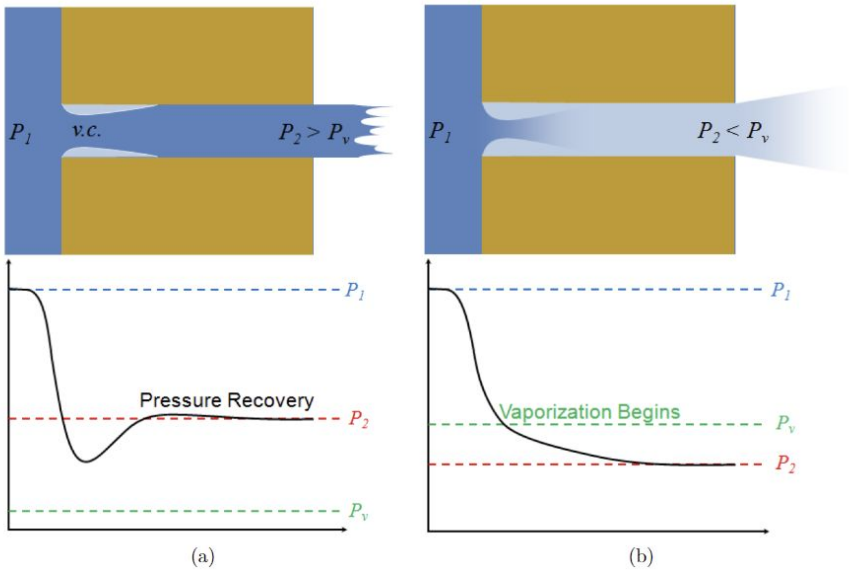


Figure 2.17: Conceptual injector pressure history as adapted from work by Dyer et al. showing (a) low vapor pressure propellant and (b) high vapor pressure propellant with flow from left to right [30]. The minimum area of contraction due to flow separation is denoted by v.c. (vena contracta).

Oxidizer Tank Model - Injector Model Overview

- The SPI model is commonly used with two-phase nitrous oxide through an experimentally determined discharge coefficient Cd.
- This model was already implemented in Ben's thesis, and his investigation found the SPI model produced the best results for his series of hybrid engine test cases.
- Assumes constant density through injector (neglects two phase and compressibility effects)

$$\dot{m}_{SPI} = C_d A \sqrt{2\rho\Delta P}$$

Where:

- Cd - experimental discharge coefficient
- A - total area of injector orifices
- Rho - density
- Delta P - pressure drop across injector

Fuel Tank Model Overview

- Quasi-steady state
- Two fluid control volumes, one for nitrogen and the other for ethanol
- Pressurant behaves as an ideal gas
- Adiabatic Tank
- Incompressible (SPI model) where 100% of the liquid drains without any pressurant.
- Heat transfer from liquid to vapor cv following

Combustion Chamber Model

- Ideal combustion from NASA CEA
- Combustion products are an ideal gas in chamber and nozzle (this is a good assumption - high temperature low pressure). 1D flow.
- Adiabatic casing and nozzle.

Rocket Validation Cases:

- Motivation:
  - Understand the accuracy of my models and ensure they are not producing garbage data.
  - Build an understanding of how model predictions diverge from reality to make informed decisions on how to use model results.
  - Build an understanding of where I can ‘extrapolate’ model’s results to my engine and where I cannot.
- I found my validation cases in student team competition reports and conference papers.
  - Can plug in ‘inputs’ from their systems and compare my predicted model output to their experimental hotfire data.
  - Similar engine scale, publicly available.
- Challenges:
  - Finding reports with similar enough engines (pressure fed bi-propellant with nitrous as the oxidizer and a hydrocarbon fuel)
  - Reports sometimes excluded key information I needed for a full validation. Both inputs into my script (commonly injector discharge coefficients) and experimental engine time histories.
  - (Sometimes) conflicting information in reports.

Rocket Validation Cases:

- An inherent challenge was finding enough validation data. Unfortunately, I was only able to find two team reports with enough data to create a validation case.
- These team reports often lack low level detail about their systems, which makes it hard to understand parameters like which flow regimes (choked vs unchoked) and O/F ratios their engines are burning at, as well as engine efficiencies. This makes it hard to make conclusions or understand which regimes and conditions my model can represent.
- However, I was able to find additional validation cases with a smaller scope (at the single component level) that will be introduced later in this presentation.

Validation Case 1: MASA_Laika_2018 - source [9]

- The liquid bipropellant engine that was flown on the University of Michigan's 10k ft apogee Laika Rocket.
- Nitrous Oxide and Ethanol Engine
- Like-unlike propellant impinging injectors
- Overall, this engine is likely the most similar to what I am trying to build (same propellants, same mission, same high level engine design decisions)

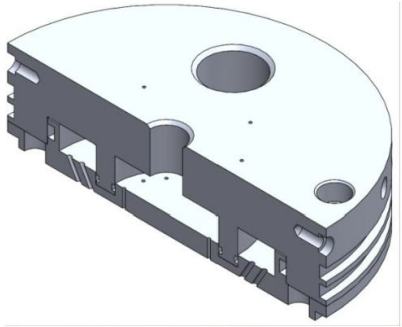


Figure 4. Injector Manifold Assembly. *This cutaway shows the fuel orifices (center) and oxidizer orifices (outer). Two o-rings separate the two flow paths of the manifold. Vent holes can be seen on the top face.*

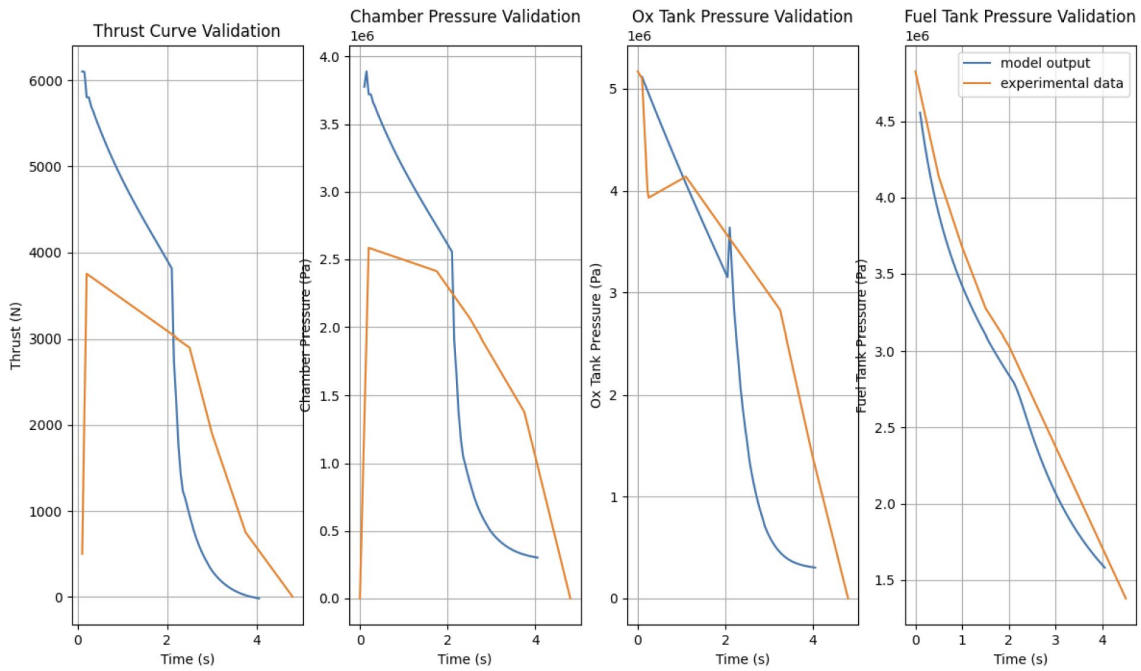
Validation Case 2: Stanford_2022 - source [10]

- Nitrous Oxide and Kerosene Engine (different fuel)
  - From the propellant selection we have previously done, we have seen that ethanol and dodecane (using to approximate kerosene) are very similar in theoretical combustion performance.
- Provides estimated discharge coeffs.
- Unfortunately did not provide chamber pressure data

Failed Validation Cases

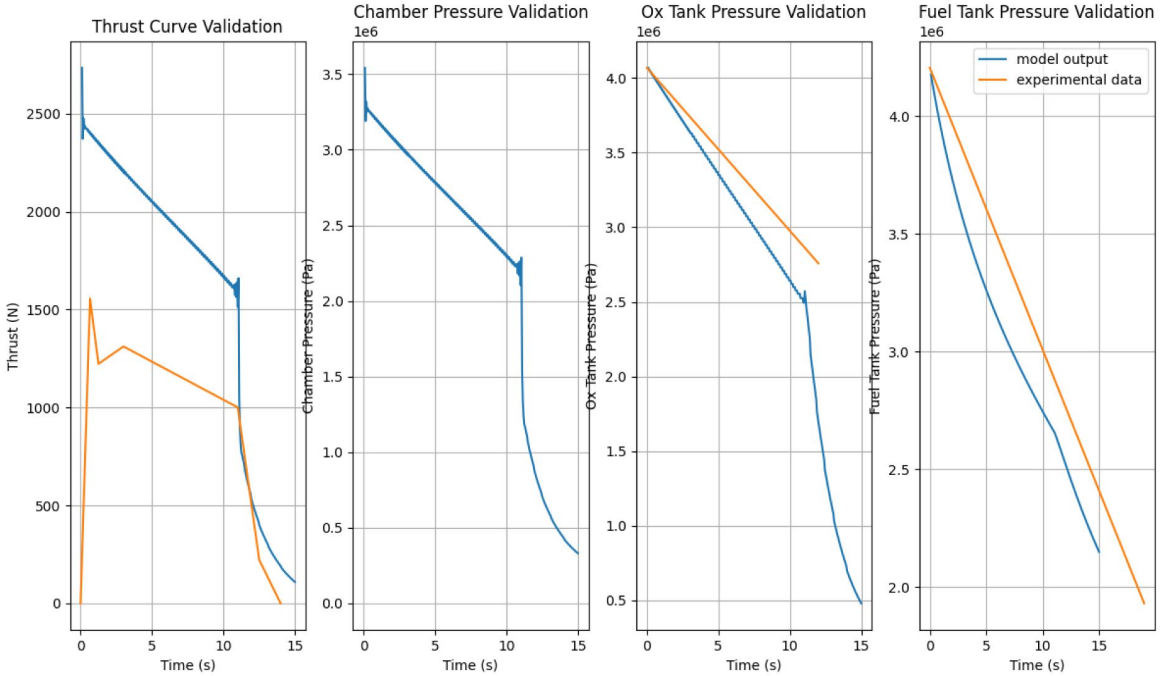
- Waterloo 2024
  - This engine used a different type of injector (Coaxial) - no provided discharge coefficients. I would have to be confident enough in the model that I could “back solve” discharge coeffs through runs - have not built this confidence yet.
  - This was realized later, but I am also not sure if we could predict choking for the outer coaxial geometry (not a round hole).
  - Additionally, their fuel tank uses a piston to feed the gas, which my fuel tank model does not represent. (had to pipe fuel tank pressure into my script)
- Purdue 2019
  - This engine used a different type of injector (pintle) with no published discharge coefficients. Similar problem to Waterloo 2024.
- UCLA 2018
  - Missing discharge coefficients. Only thrust data available (no system pressures).

MASA_Laika_2018 Results



- Significant overprediction of thrust and chamber pressure.
- Significant underprediction of total burn time. Oxidizer tank models the liquid draining at a rate that is too high

Stanford_2022 Results

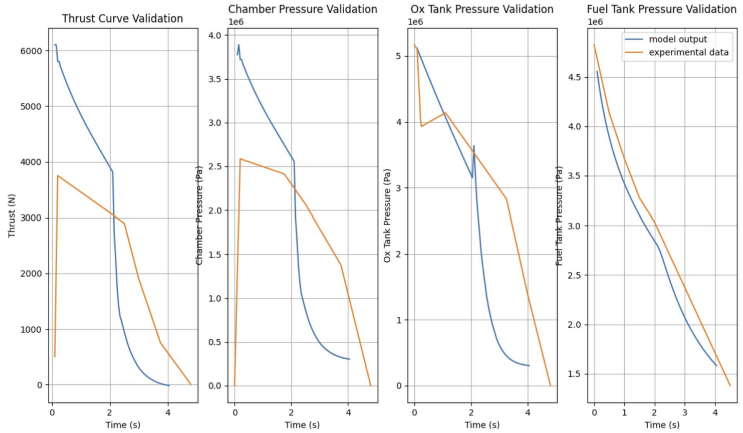


- Note: this paper did not include chamber pressure data.
- Seems like fuel tank drains before oxidizer tank drains.

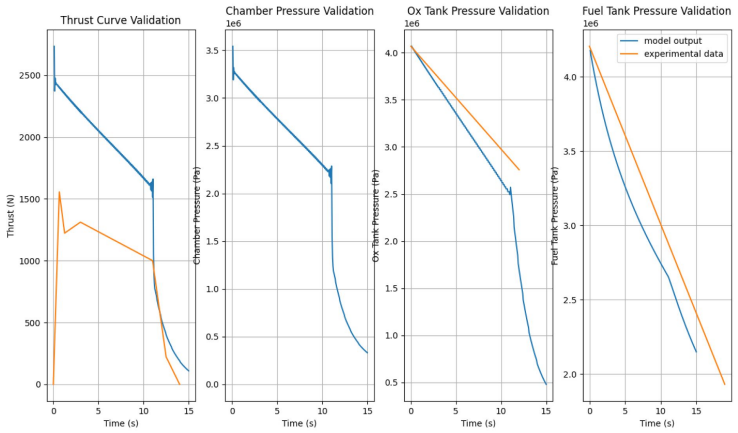
Comparing the Initial Program Runs against each other

- Model significantly overpredicts thrust for both cases.
- We know the MASA_Laika_2018 case has the oxidizer tank drain too fast, but we do not have enough information to make the same conclusion for the Stanford_2022 case

MASA Laika 2018



Stanford 2022



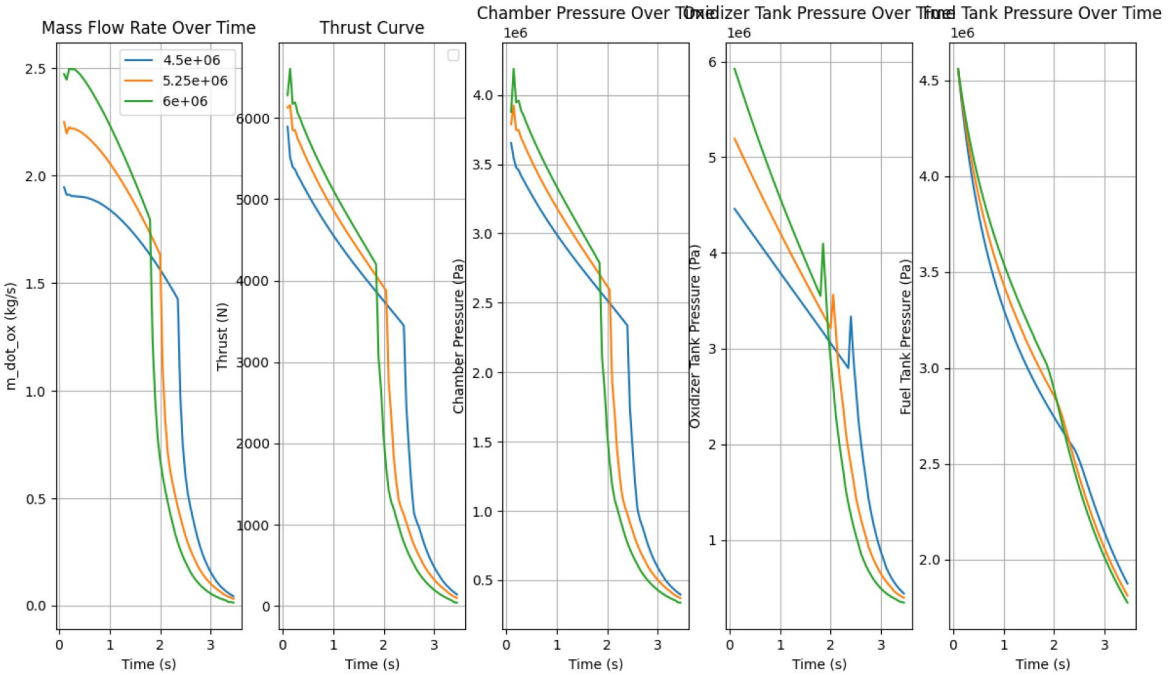
Conclusions from Initial Program Runs

- Hard to draw strong conclusions because of the few test cases we are limited to.
- Overall, we are significantly over predicting thrust.

Sensitivity Analysis

- Motivation: understand how changes in desvars impact the system:
  - Helps us understand trade-offs.
  - Allows us to understand the impact of deviations from predicted conditions on our system .
- Challenge: Properly quantifying the variance in time curves - I have been only visually comparing the graphs at this point. I am not sure how to statistically compare the results. This is an example of something I know should be done better, but I have given it an extremely low priority given my available work hours.
- I have presented two noteworthy examples in the next two slides

Sensitivity Analysis: Changing Oxidizer Tank Pressure of MASA_Laika_2018 Case:



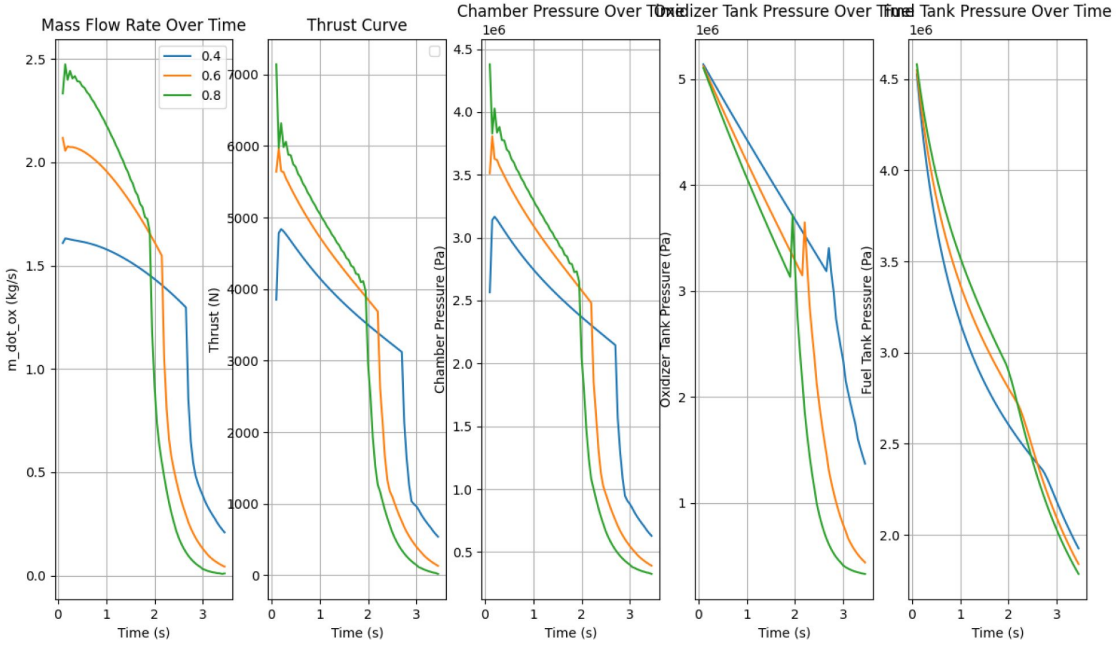
(For reference later in the presentation: simulated with SPI injector model)

```
### Sensitivity Analysis:
test_var_name = "P_tank"
min_bound = 45e5
max_bound = 60e5
num_iterations = 4
```

Overall, system is somewhat insensitive to changes in initial oxidizer tank pressure (Under 1000N of thrust across a 1.5 MPa difference in pressure)

Sensitivity Analysis: Changing Oxidizer Discharge Coefficient of MASA_Laika_2018 Case:

(For reference later in the presentation: simulated with SPI injector model)




```
### Sensitivity Analysis:
test_var_name = "Cd_1"
min_bound = 0.4
max_bound = 0.8
num_iterations = 3
```

Sensitivity Analysis over the expected range of discharge coefficients

A change in 0.4 of the discharge coefficient results in a difference of over 1000N at the peak of the thrust curve.

Learnings from initial runs and building an understanding of the engine as a system:

- Most sensitive desvars are the injector orifice areas and discharge coefficients.
  - This makes sense, the injectors determine the pressure drop, the mass flow rate into the combustion chamber and by extension the O/F ratio of combustion.
- Changing the injector discharge coefficient by 0.4 had more impact on performance than changing the initial oxidizer tank pressure by 1.5 MPa for MASA_Laika_2018.
  - Additionally, Cd changes the shape of the curve more than oxidizer tank pressure.
  - This is extremely concerning. For context, the injector discharge coefficient can significantly change from machined surface finish, and how sharp vs rounded the orifice edges are. These small details can be hard to control, but have a significant impact on performance.
- System is relatively insensitive to change in other parameters like starting oxidizer tank pressure

Estimated Injector Discharge Coefficients

- We found that the injector discharge coefficients are the most sensitive variables in our system.
- Source [5] compared injector orifices of the same diameter with different types of edge (square, rounded, chamfer) and found a 0.1 difference in discharge coefficient. That is huge for a relatively small change in geometry.
- We have seen through our sensitivity analysis that a change of 0.4 in our discharge coefficient has an extreme effect on the model's output.

Figure 19 demonstrate a significant result: while the mass flow rate of an injector hole can be increased dramatically by installing a simple chamfer, there is almost no additional improvement by going through the extra effort to round the edges instead.

[3]

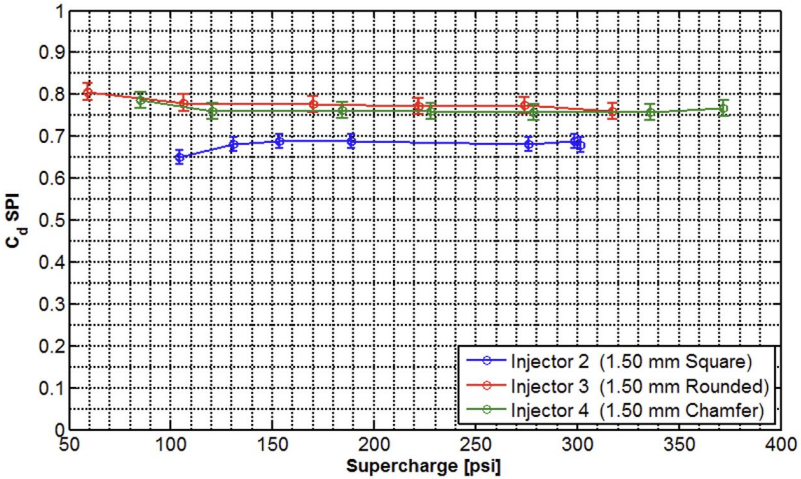
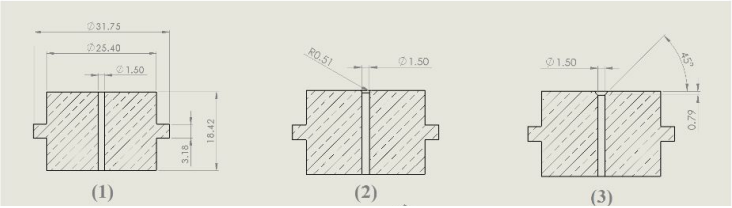


Figure 18. C_d **in the single-phase region vs. supercharge for injectors number 2, 3, and 4 using carbon dioxide.**



[12]

- To some extent we do not have perfect control over the injector orifice edge geometry and surface finish due to the nature of real world manufacturing.
    - This would increase the uncertainty of the discharge coefficient value, which is our most sensitive desvar.

Ultimately, with the large amount of error between our model's predictions and the experimental data, as well as the uncertainty of what our engine's actual injector discharge coefficient may be, I am not confident enough to use the model to generate the loads for the detail design phase.

Injector Modeling

Injector Modelling

- From seeing the results of the script, I decided to see if implementing different injector models would improve the accuracy of the model.
- I chose the Homogeneous equilibrium Model (HEM) and Dyer (weighted average of SPI and HEM) models because they are commonly used for nos injectors (theory presented in later slides)
- the SPI is known to overpredict flow rate and the HEM model is known to underpredict. Therefore, the Dyer model might be very accurate.
- If the accuracy of the model's results are still poor, I want to see if we can use the different models to find an upper and lower bound of mass flow rate for injector design. If the bounds are narrow enough, we could potentially proceed with a design.
- These models are also very simple to implement.

Homogeneous Equilibrium Model (injector model) [3]

- Assumptions:
  - Isentropic flow
  - Flow is not necessarily a single phase, and if not, nitrous may exist as a saturated liquid vapor. More specifically, this means the liquid and vapor states of nitrous oxide are in thermal equilibrium.

$$\dot{m}_{HEM} = C_d A \rho_2 \sqrt{2\left(h_1 - h_2\right)}$$

Where:
- h_1 = upstream enthalpy
- h_2 = downstream enthalpy
- rho_2 = downstream density
- Cd = HEM discharge coefficient
- A = orifice area

Dyer injector model

- In reality, there will likely be non-equilibrium effects, which would cause the HEM model to fall short. Therefore, I would also like to try implementing the Dyer model because it is simple (based off of HEM and SPI model) and it accounts for some non-equilibrium behaviour.

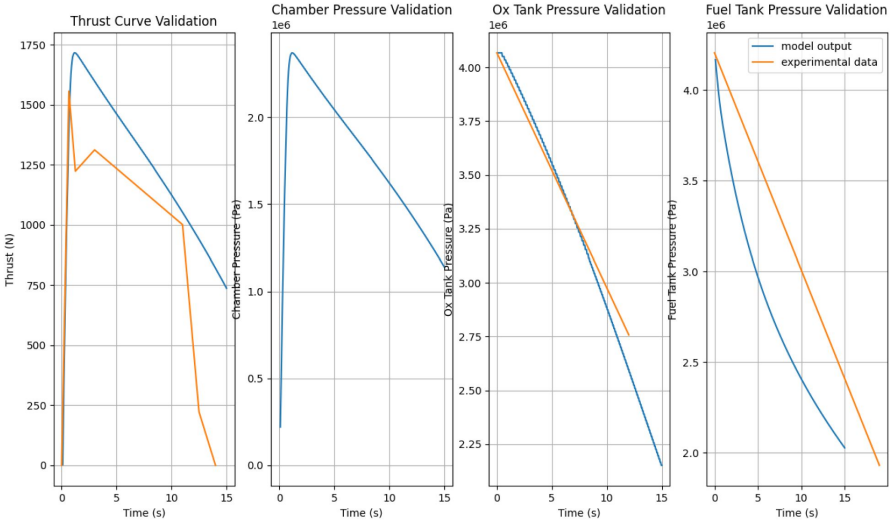
Dyer injector model

$$\dot{m}_{DYER} = A\left(\frac{\kappa}{1+\kappa}\dot{m}_{SPI} + \frac{1}{1+\kappa}\dot{m}_{HEM}\right)$$

- A non-homogenous, non-equilibrium model which uses a weighted average between the SPI and HEM mass flow rates:
  - These mass flow rates are weighted with a non equilibrium term κ, which compares the bubble growth time with the fluid residence time:

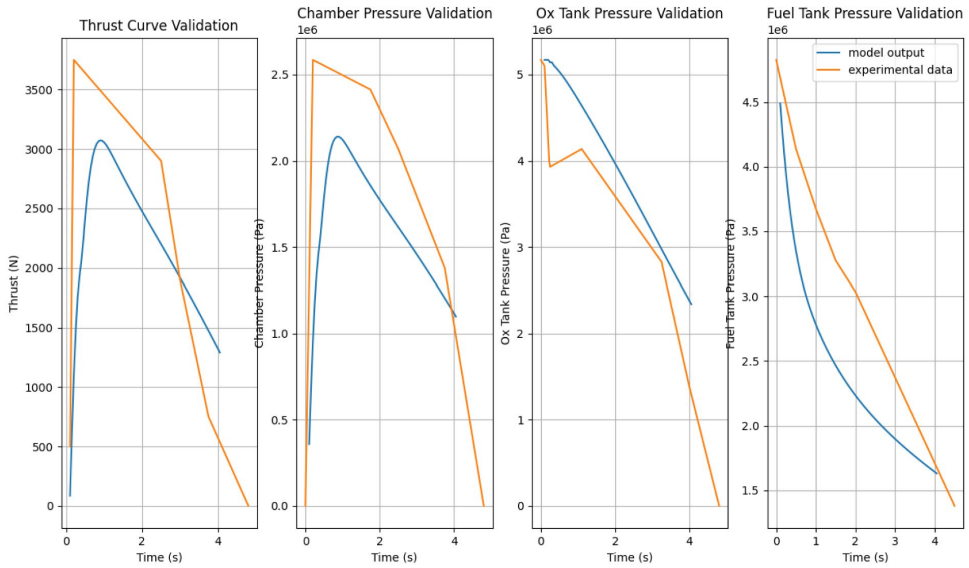
$$\kappa = \sqrt{\frac{P_1 - P_2}{P_v - P_2}} \propto \frac{\tau_b}{\tau_r}$$

Stanford_2022 HEM injector model results



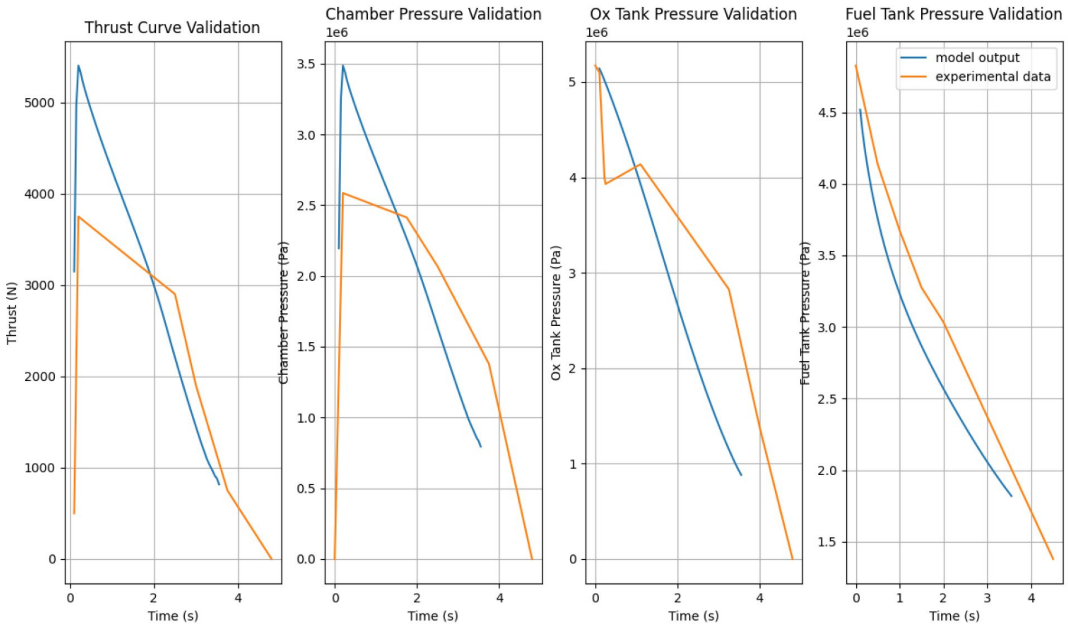
- Hem model significantly overpredicts thrust in this case.
- Oxidizer tank pressure matches extremely well.
- Fuel tank pressure does not match well. It is hard to make a conclusion without seeing the chamber pressure data, although there is likely an overprediction of thrust because the model is predicting an O/F ratio of combustion closer to stoichiometric than occurred during the experiment.

MASA_Laika_2018 HEM injector model results



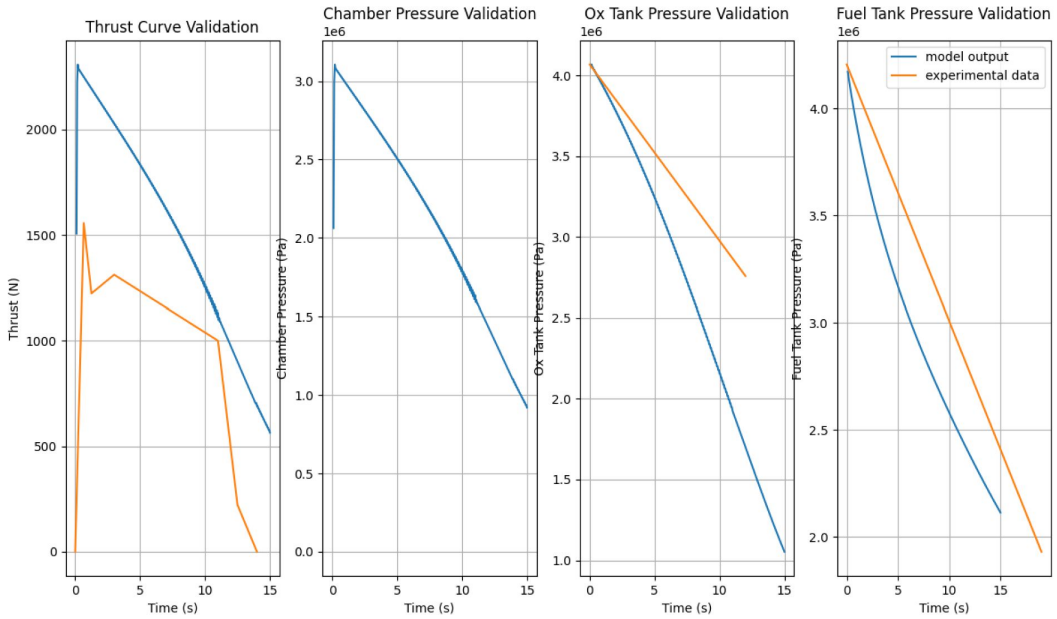
- Significant underprediction of thrust and chamber pressure
- See the next slide for a comment on the fuel tank performance

MASA_Laika_2018 Dyer injector model results



- Note: here we see the fuel tank model dramatically change as only the oxidizer injector model used in the model changes (through chamber pressure changing). This is why I decided to look into the oxidizer injector model.

Stanford_2022 Dyer injector model results



- Significantly overpredicts thrust while underpredicting both oxidizer and fuel pressure.

Injector Modelling Conclusions

- Problem: Seeing an extreme variance in results of SPI (overprediction) and HEM (significant underprediction).
- Dyer model does not appear to be a good average.

Injector Model Variance Solution

Conclusion:

- Realized model could not predict if flow was choking, which was causing unreasonable high mass flow rates and garbage results.
  - This explains why the SPI MASA case overpredicted thrust and underpredicted burn time: If the model was not predicting choking, it would overpredict the mass flow rate of nitrous and cause the tank to drain too quickly.
  - It also seems like using the SPI discharge coefficient to account for flow choking is not a reliable method (Also mentioned in [3]).

Reaction:

- Implement “modified” models to account for flow choking. It is easy to predict if liquid ethanol chokes (coolprop library has this capability), but it is much harder to predict when the choking point of two-phase nitrous oxide is.

A closer look at the HEM model:

- This explains why the HEM model would significantly underpredict: we were in the choked regime long past the maximum and getting an artificially low mass flow rate.
- Because the magnitude of the HEM mass flow was so small and the weight calculated by the Dyer model, the resulting Dyer mass flow was also extremely small.

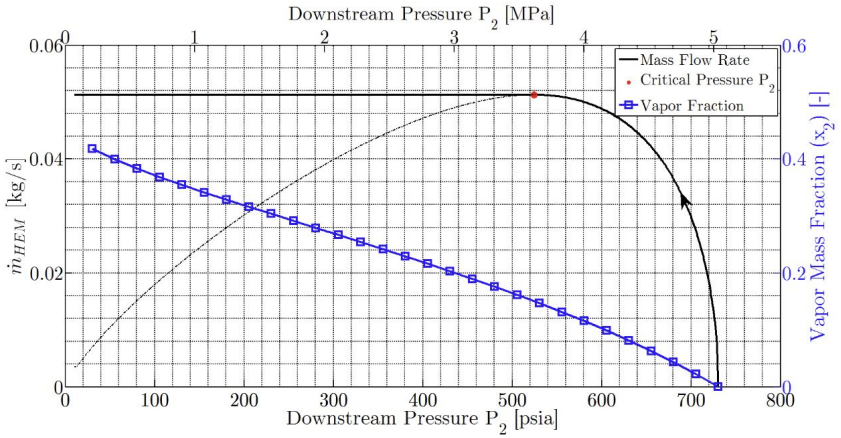


Figure 2.19: Critical mass flow rate prediction using the Homogeneous Equilibrium Model for initially saturated liquid nitrous oxide at 730 psia (5.03 MPa) corresponding to room temperature of approximately 20 ^oC (293 K). The data used to produce this plot was calculated using REFPROP [14, 15].

[3]

“Modified” HEM model to predict choking

- Modified HEM model → the approach is to take the maximum predicted mass flow as the choked mass flow rate, and to use this number past the pressure ratio it is found at.
- Unfortunately, this method does not predict the point of choking or the magnitude of choked flow very accurately

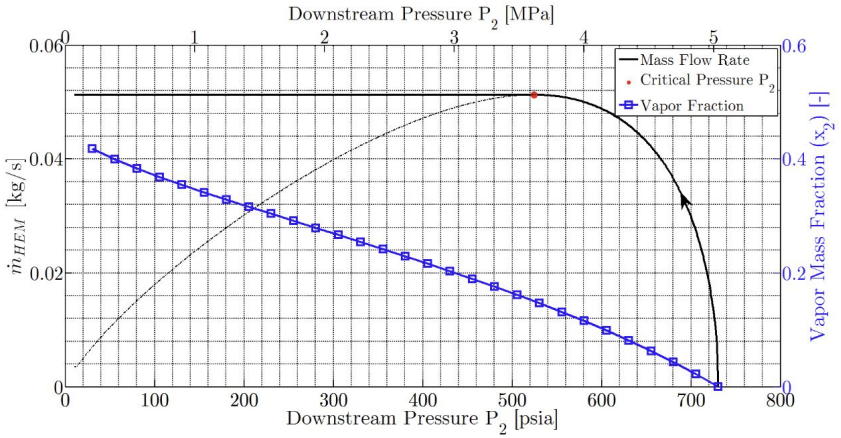


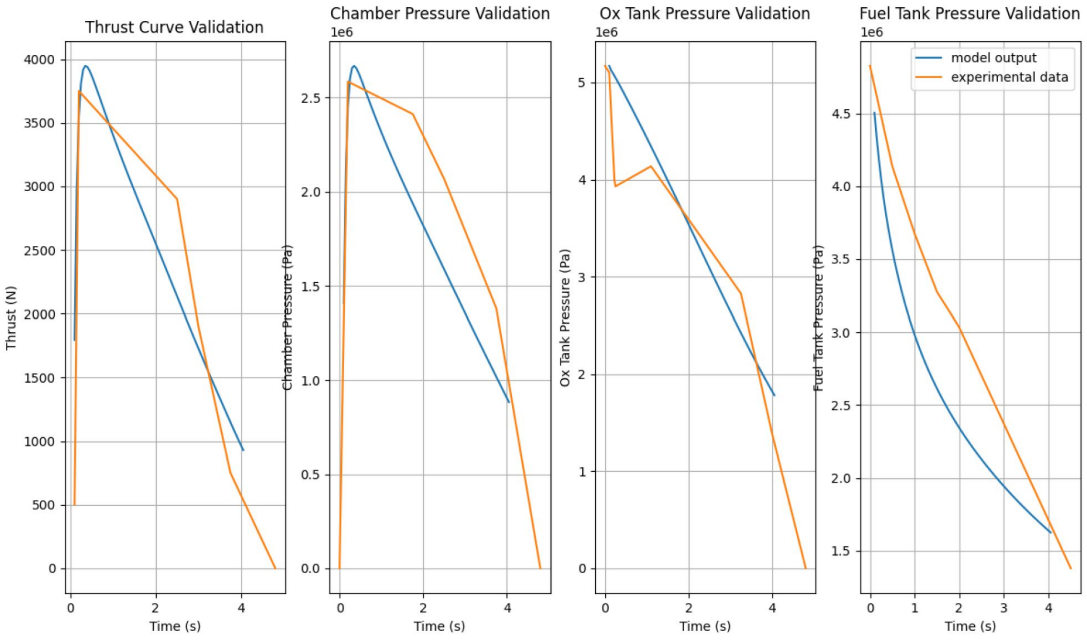
Figure 2.19: Critical mass flow rate prediction using the Homogeneous Equilibrium Model for initially saturated liquid nitrous oxide at 730 psia (5.03 MPa) corresponding to room temperature of approximately 20 ^oC (293 K). The data used to produce this plot was calculated using REFPROP [14, 15].

[3]

“Modified” SPI model to predict choking

- No true modified SPI model → the approach is to use a separate discharge coefficient Cd_SPI to account for flow choking.
  - This does not work well (ultimately just multiplying a scalar to hold a function at a maximum value). I am also trying to create a engine model that can be extrapolated past validated results to design a new engine, and this approach with the Cd_SPI seems to be mostly used to characterize an already existing engine.
  - I also tried to use the HEM choking criteria to find a pressure ratio to consider my SPI model choked, but there was nothing in literature about this method and my results were extremely poor.

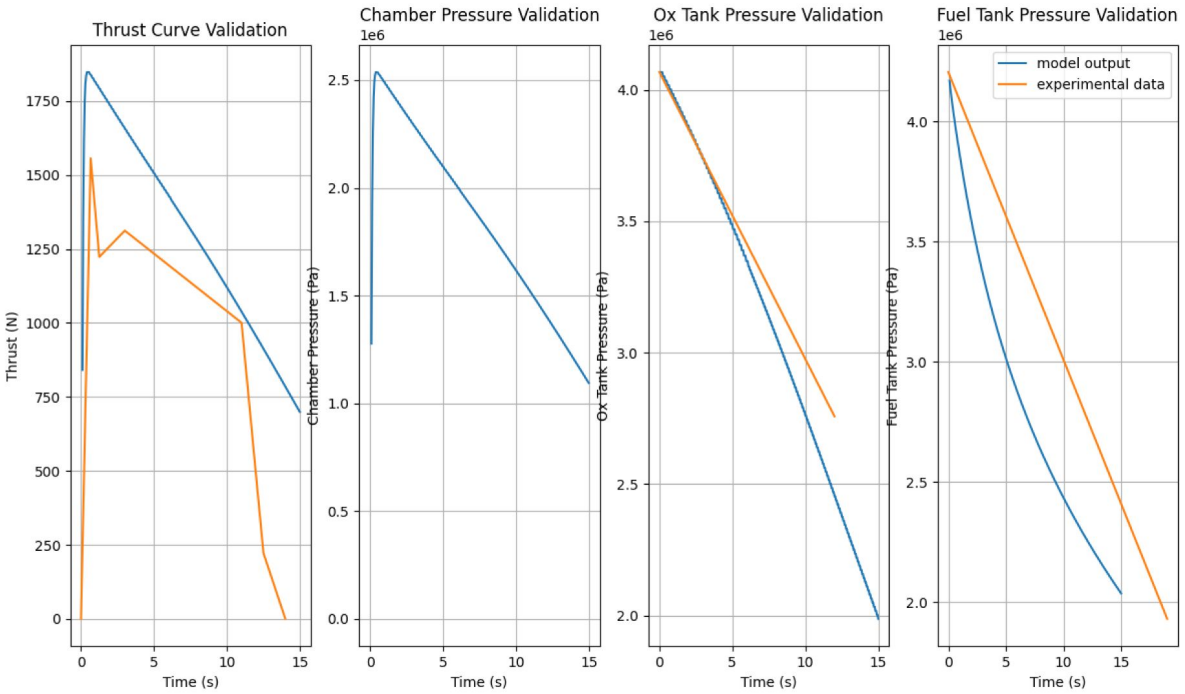
MASA_Laika_2018 test case with modified HEM Model



(Note: bad graph scaling)

- Much closer, both underpredicts and overpredicts thrust and chamber pressure at different intervals.
- At its worst point, we underpredict thrust by 1kN and underpredict chamber pressure by 0.5MPa

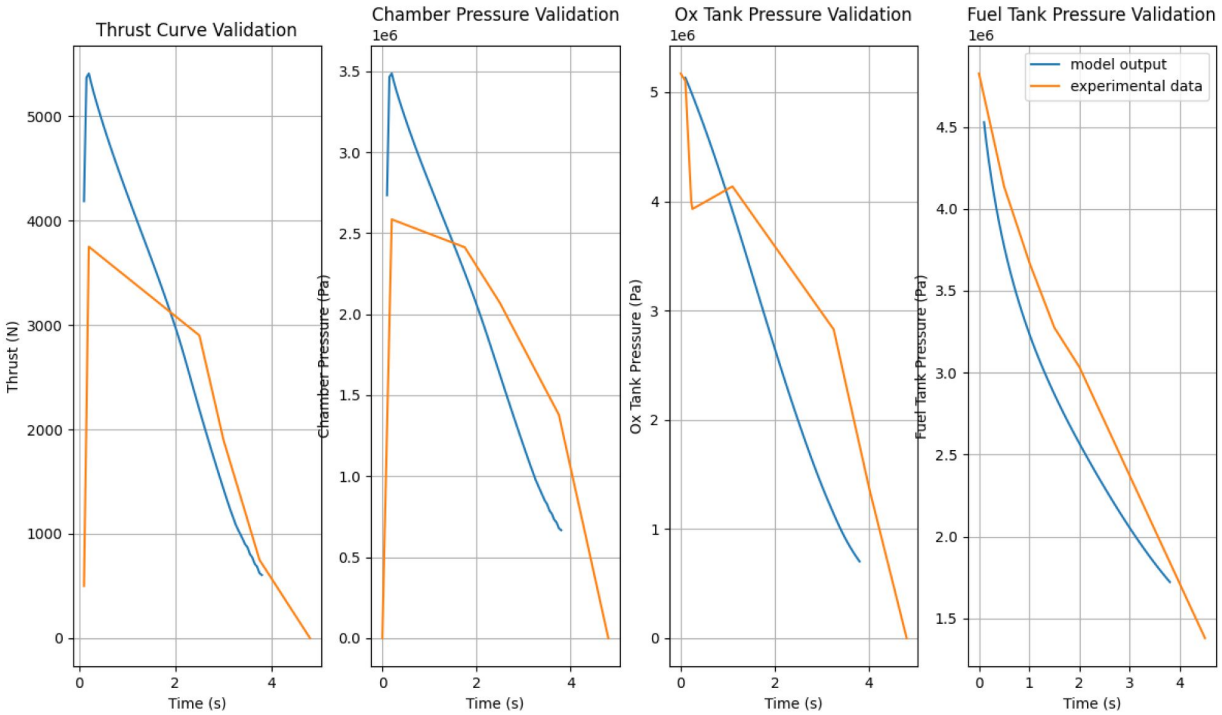
Stanford_2022 test case with modified HEM Model



(Note: bad graph scaling)

- The oxidizer tank pressure correlates extremely well.
- The error on the thrust curve can most likely be explained by error in the fuel tank model. Less fuel would likely mean we are burning with more oxygen closer to the stoichiometric ratio, which would produce higher temperatures than being fuel rich. (Not enough information to prove with certainty because no chamber pressures).
- Although the curve is similar in shape, it deviates differently than the MASA_Laika_2018 case

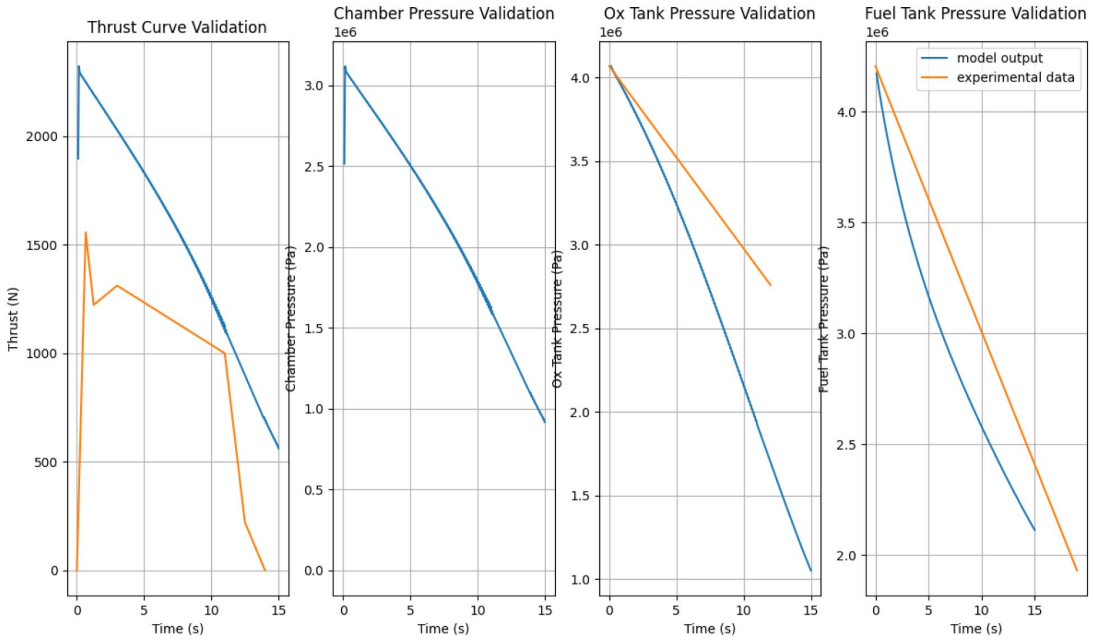
MASA_Laika_2018 test case with Dyer Model (updated HEM)



(Note: bad graph scaling)

- Really bad curve fit, both underpredicts and overpredicts thrust and chamber pressure. (+/-1500N, +/-1MPa), which I believe is not accurate enough to proceed with detail design

Stanford_2022 test case with Dyer Model (Updated HEM)



(Note: bad graph scaling)

- Does not change from the simulation without modeling choked flow. This shows that this engine is not modelled as choking Much worse than the HEM model alone.

Modified model results

- Still gives results that are not acceptably accurate.
- Cannot apply a correction factor because the results are not very uniformly out, and we do not have enough test cases to make conclusions.
- Another inherent issue is being unsure of exactly when and why the engines deviate from these models.
- I think it is extremely important that given my project goal, I can have the ability to confidently extrapolate the results of my model to a test case that isn’t an already defined engine. These results do not give the confidence needed to do this.

Possible solution: Pick a bunch of injector orifice sizes and test the discharge coefficients

- Given how sensitive the system is to the injector properties, we need to build a good understanding of this system to have confidence in the results of our analysis.
- I have considered the above solution, but deemed it not ideal given the objectives and constraints of this project:
  - By definition, choked flow is when the mass flow rate cannot decrease past a certain downstream pressure. Since our engine will experience a range of upstream pressures, we would need to perform a large amount of testing to measure the discharge coefficients under choked flow for the range of our upstream conditions.
  - The system is too sensitive to the injector properties to proceed without testing numerous injector orifice diameters. The discharge coefficient will also change with the L/D ratio of the orifices as well. This method quickly becomes overwhelming for a single person if we have to test multiple samples at multiple upstream pressures.
  - Additionally, this will limit our design space if we cannot consider injector orifices outside of the ones we have tested. This may be an issue if we find that the tested geometry does not yield a configuration that can meet our mission objectives (slight increase of project risk).

Chosen Solution: Implement a new injector model:

- Through research, I found a new injector model: Emerson + Mohammad's injector model
- This model has the ability to very accurately predict the point of flow choking and magnitude
- Provides mass flow rate as a function of only known upstream thermodynamic properties and back pressure. This means it integrates extremely well with the current model.

Chosen Solution: Implementing the Emerson + Mohammad Model

- Because nitrous oxide is a two phase fluid, it is extremely difficult to predict when the flow chokes. However, this model succeeds in accurately predicting the point of choking:

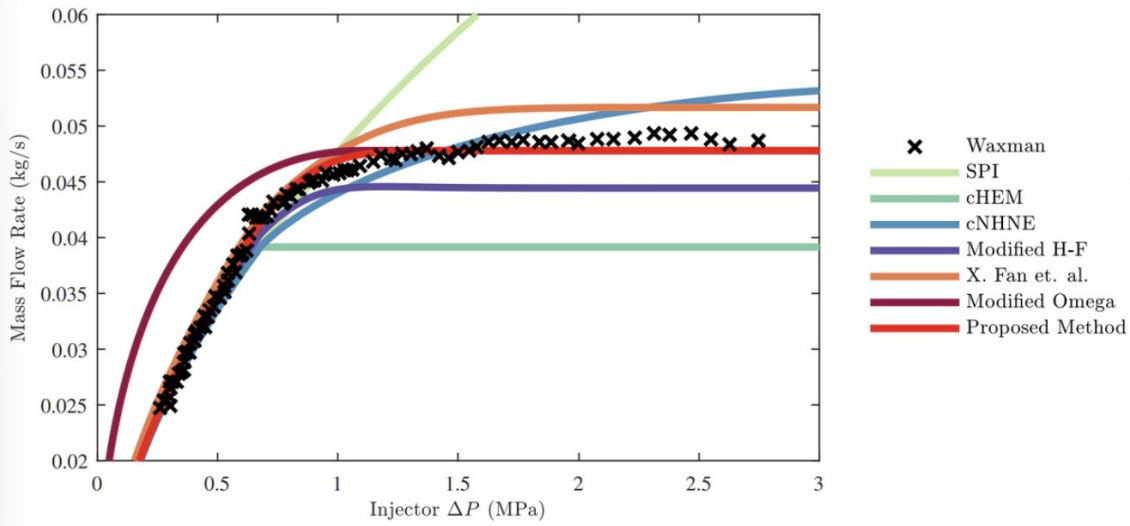


FIGURE 2: Pressure drop and corresponding mass flow rate predictions of the models for a 1.5 mm, N_2O injector. Experimental data from identical Waxman test case is also shown.

[5] - note:
black ‘x’ represents experimental data,
Red “Proposed Model” is by Emerson + Mohammad

Emerson + Mohammad Injector Model Overview [5]

- The authors identify 3 possible nitrous oxide states, low supercharged liquid, high supercharge liquid and two-phase inlet. (Note: supercharge = subcooled)
- This model is composed of 3 models:
  - If the nitrous oxide is in high supercharge, the Omega model is used.
  - For low supercharge, a Modified version of the Omega model is used.
  - The results from the Dyer model are blended with the choked regime and used for non-choked flow for both high and low supercharge.
  - Saturated cases are calculated according to the Omega model
- Another significant advantage is that the model uses more general flow discharge coefficients for choked regimes. This will significantly reduce the chances of using a bad input discharge coefficient for the choked regime (our primary engine operating regime)

Omega Model Overview [11]

- Very General: designed for two-phase, subcooled or saturated inlet conditions, caviating or non-cavitating, choked or non-choked flow.
- A type of Homogeneous Equilibrium Model (assuming inviscid flow and adiabatic orifice), where the difference between the previous HEM model is the thermodynamic path the fluid undergoes (given by eqn (2)).
    - Does not predict non-equilibrium behavior between liquid and vapor phases.
- Eqn (2) assumes that the initial upstream pressure is the saturation pressure found at the temperature of the inlet.

The two-phase equation of state (EOS) adopted to analytically solve Eq. (1) is

$$\frac{\rho_0}{\rho} = \omega\left(\frac{P_0}{P} - 1\right) + 1, \tag{2}$$

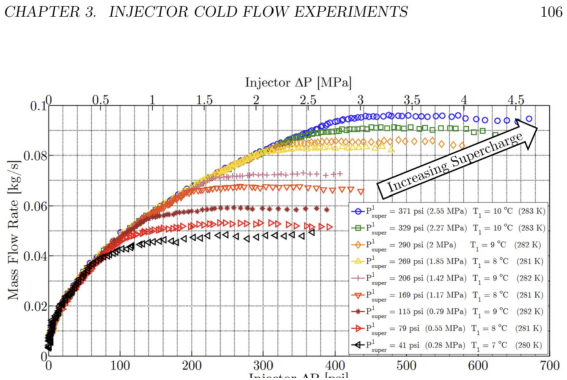
where the subscript 0 refers to the inlet conditions. The parameter ω (so-called "compressible flow parameter") is a function of the fluid properties at the inlet conditions only and, for saturated or two-phase inlet conditions, it can be written as

$$\omega = \frac{x_0 v_{LG0}}{v_0} + \frac{C_{L0} T_0 P_0}{v_0}\left(\frac{v_{LG0}}{h_{LG0}}\right)^2 \tag{3}$$

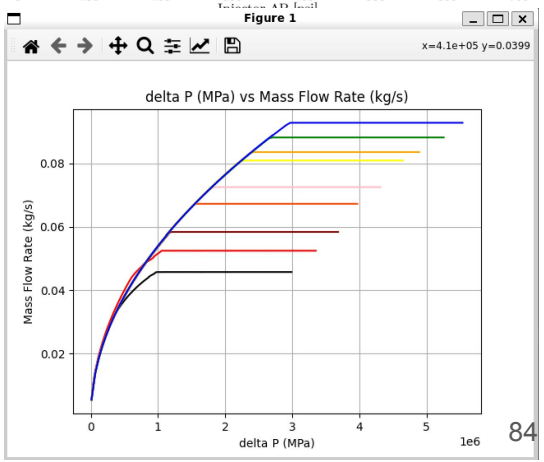
Validating Emerson and Mohammad Model

- I validated the injector model against experimental data from a different paper.
- This dataset was chosen to isolate the injector model during testing
- Result: Successfully matched the test data.

Published Data: [3]



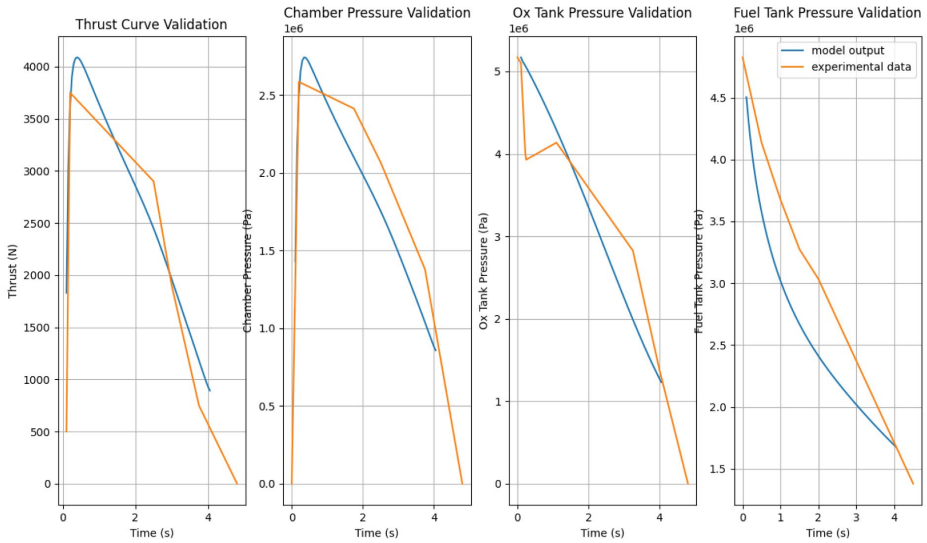
My Implementation Output:



Oxidizer Tank Modeling

The next step is to integrate the new injector model into the equilibrium oxidizer tank model:

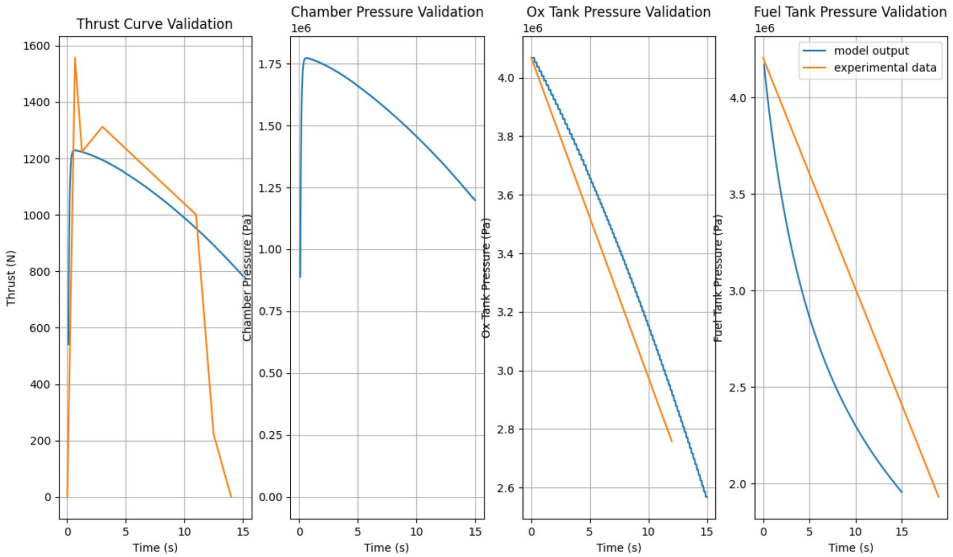
- MASA_Laika_2018 test case with Emerson + Mohammad Model:



- We are seeing a much better curve fit, but the curve is still not that accurate:
  - Predicted oxidizer tank pressure is 1 MPa less than measured at the worst point.
  - Predicted fuel tank pressure is at 0.5 MPa less than measured at the worst point. (likely a separate issue with the fuel tank's simple assumptions)

The next steps are to integrate the new injector model into the equilibrium oxidizer tank model:

- Stanford 2022 test case with Emerson + Mohammad Model:



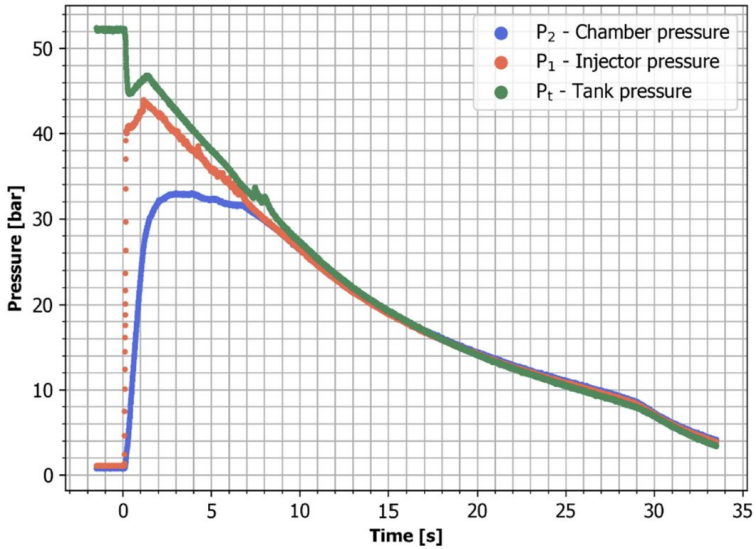
- This case has a thrust curve and oxidizer tank that match extremely well.

Work Completed so far: Injector Modelling (2)

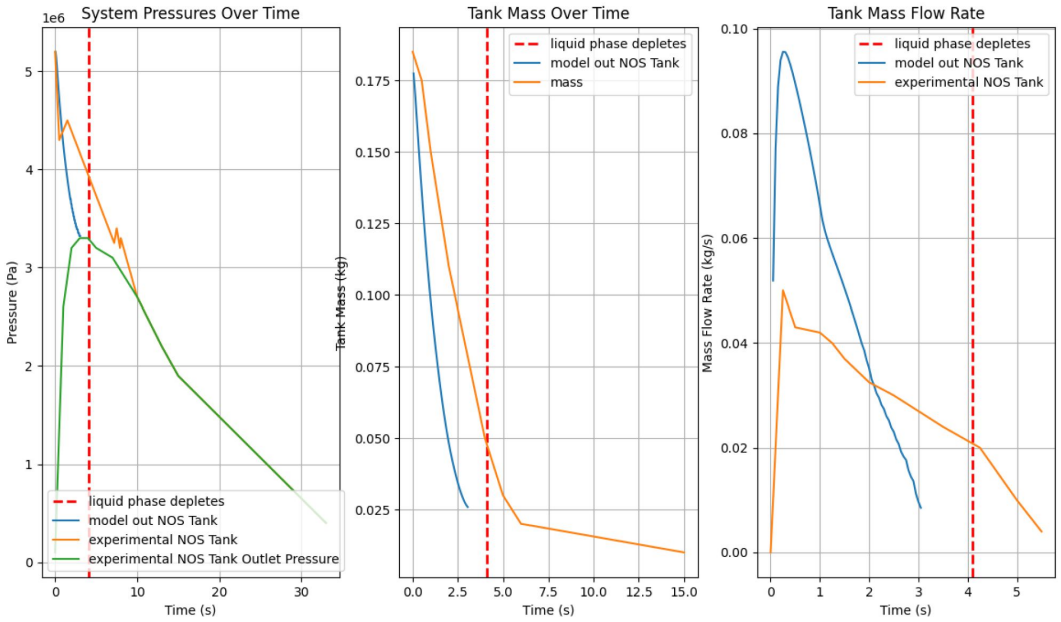
- Results: when I tested the new injector model with the tank model, I was getting extraneous results. Found that the density of nitrous oxide fed into the injector was too low. This can be explained by the equilibrium assumption in the tank.
- Approach:
  - Created a new test case with just a tank and an injector to isolate other effects on the results. Found another paper with experimental data from draining a nitrous tank and injector system (Tomasz test case).
  - Ran the test case and discovered error in mass flow rate and tank pressure. Note:Injector flow proportional to upstream pressure and nitrous density at injector. Compared the Emerson + Mohammad model to the SPI model, which showed poor agreement between both upstream pressure and tank mass over time. Makes sense since SPI is not modelling choking.
- Result: Emerson + Mohammad model shows a strong improvement with tank mass over time (modelling choking), but still shows significant error in mass flow rate from upstream tank pressure

Reference [6] Tomasz-test-case overview:

- This paper contains published nitrous oxide run-tank data where the run tank is vented to the atmosphere. (isolated from a combustion chamber), as well as enough information to input into the tank model.
- Using the test case to learn more about the injector model issue



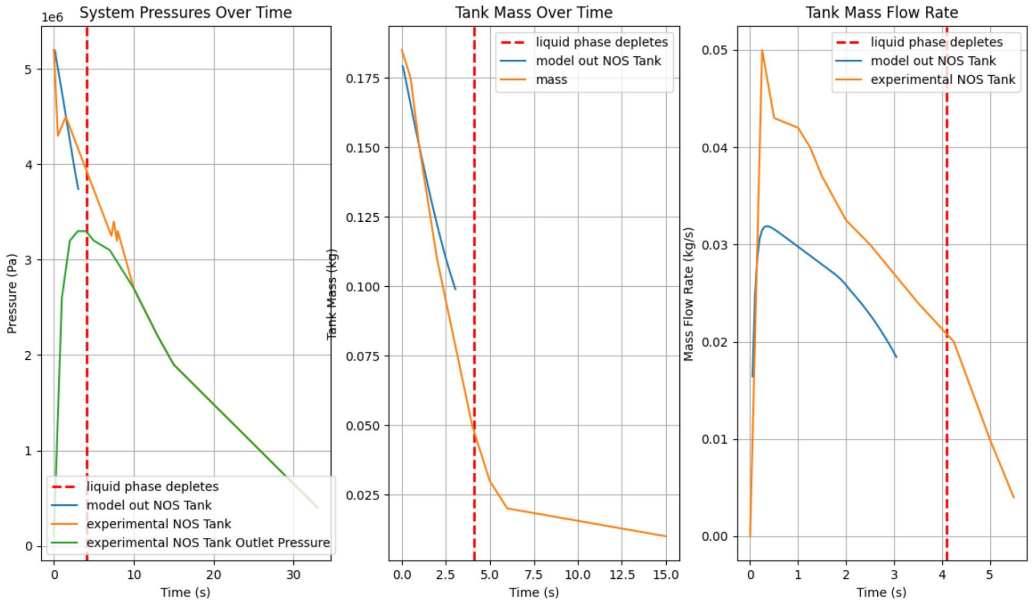
Tomasz-test-case simulated with SPI Injector model



Tank mass flow rate is too high (flow is not being modeled as choked when it is in reality)

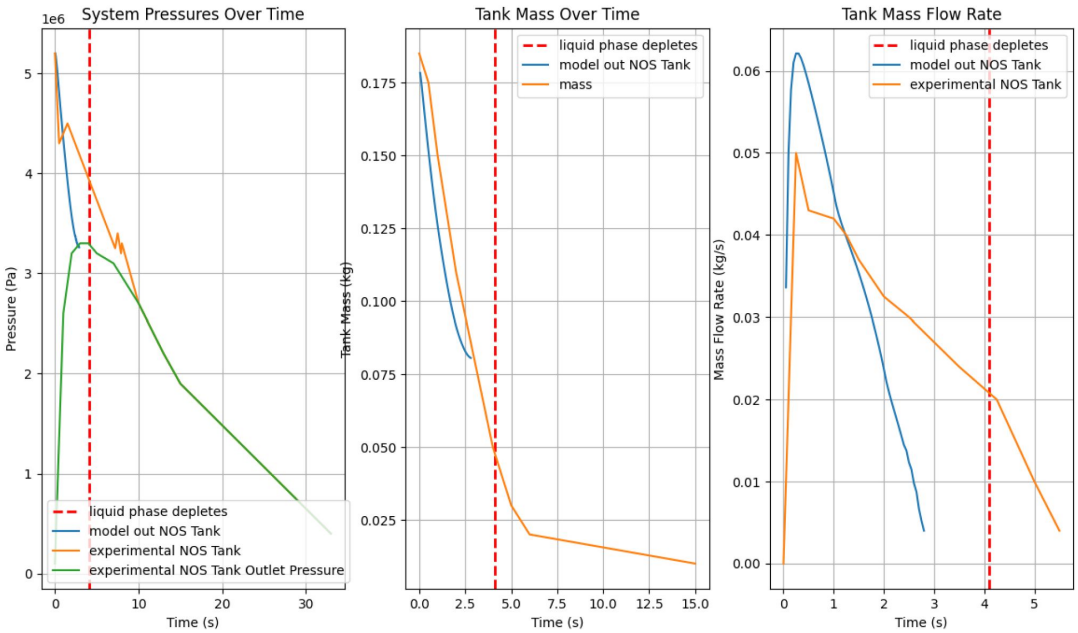
- Because the SPI model is so out, the DYER model will also not be accurate

Tomasz-test-case simulated with modified HEM Injector model (choking)



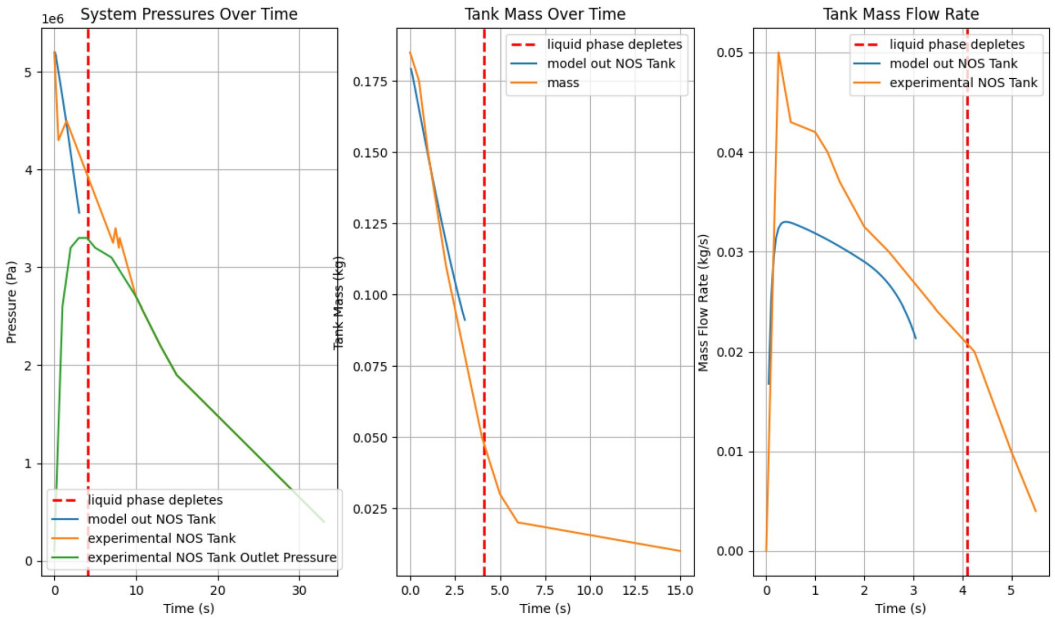
- A much better representation:
  - Tank mass over time matches reasonably well
  - Tank mass flow rate and system pressures show significant error
- This shows a potential issue with the density and pressure input into the model from the tank

Tomasz-test-case simulated with Dyer Injector model



- Tank mass flow rate is too high because the flow is not being modeled as choked in the SPI when it is in reality.

Tomasz-test-case simulated with Emerson + Mohammad Injector model



- Overall, the shape of the curves look very similar to HEM case, but this model is slightly closer to the experimental data curves.
  - (this makes sense considering both are HEM models)
- For reference, the nitrous oxide was a saturated liquid vapor for the whole duration through the injector

Understanding the results of Tomasz test case

- Approach: Research to find possible causes for the tank mass over time to be correct but the tank pressure to be wrong in the Tomasz test case.
- Result:
  - Found research that identified the problem with tank pressure: The equilibrium model does not model the evaporation of nitrous liquid since it is assumed the nitrous oxide is a saturated liquid vapor.

Tank Evaporation

- This implies that for systems that have a low enough amount of mass flow rate relative to the size of the tank where the liquid and vapor can maintain equilibrium or the evaporation is negligible, the equilibrium tank model is ok.
  - Unfortunately, we don’t have enough validation cases to try to find a relationship (potentially with non dimensional numbers between heat transfer rate and evaporation) to try to understand where I can apply the equilibrium tank model.
- Because the error is still unacceptable and there is ambiguity where the tank model applies, I have found research that presents a new tank model that models the evaporation.

Oxidizer Tank Modeling

Non-equilibrium tank model review (source 7,8)

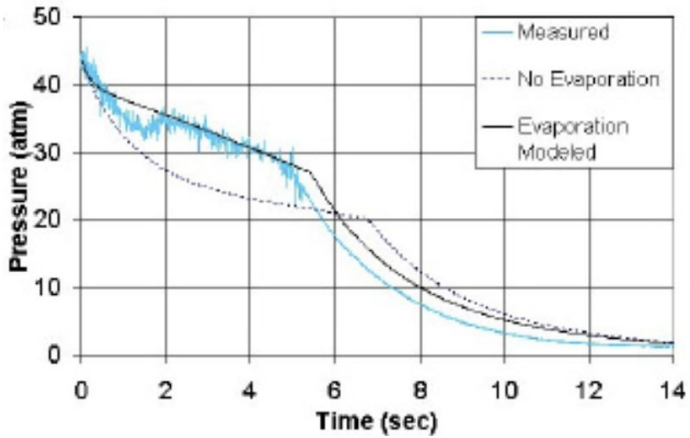
- I have provided a summary here, but I would strongly recommend reading the thesis.
- From this point, it is important to differentiate the model described by the thesis and my implementation, because the results do not match right now.
- Convention:
  - The words “My Implementation”,

Non-equilibrium tank model review (source 7,8)

- Developed by Stanford, Koç University and NASA for the Peregrine Sounding Rocket Program.
  - This program uses a hybrid rocket engine with paraffin wax as fuel and nitrous oxide as the oxidizer. (Model was developed for the same application).
  - Additionally, the authors performed a cold flow and published the data to validate the model.

Non-equilibrium tank model review (source 7,8)

- The key difference between this non-equilibrium model and the previously used equilibrium model is the inclusion of liquid-gas evaporation of nitrous oxide.
- The authors of this paper compared an equilibrium model (no evaporation) and a non-equilibrium model (evaporation) against experimental data:



The authors found that the equilibrium model predicted a much faster drop in tank pressure during the liquid phase, which is exactly what we are seeing from the results of simulating the Tomasz test case with our equilibrium model. Therefore, it is reasonable to believe that modelling the evaporation will significantly improve the model's accuracy

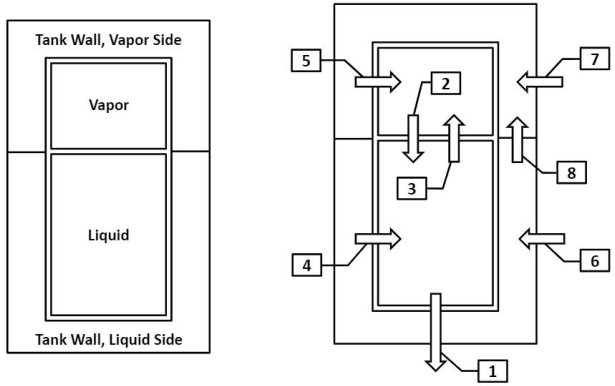
Non-equilibrium tank model overview:

- Node based solver:
  - Liquid node, gas node, liquid adjacent wall, gas adjacent wall.

[8]

A. Fundamental Theory

The models all begin by dividing the tank into a small number of regions and representing each as a single node with averaged properties. Figure 2 shows the four nodes used by the models discussed in this paper: the liquid nitrous oxide, the vapor nitrous oxide, the portion of the tank wall in contact with the liquid, and the portion of the tank wall in contact with the vapor.



(a) Nodes (b) Heat and Mass Transfer

Figure 2. Diagrams showing the nodes and the heat and mass transfer processes between them.

The heat and mass transfer processes between the nodes are numbered in figure 2 and described here:

1. Mass flow of liquid nitrous oxide out of the tank
2. Heat and mass transfer from the vapor to the liquid via condensation, diffusion, and convection
3. Heat and mass transfer from the liquid to the vapor via boiling, evaporation, diffusion, and convection
4. Heat transfer from the liquid side of the tank wall to the liquid
5. Heat transfer from the vapor side of the tank wall to the vapor
6. Heat transfer from the atmosphere to the liquid side of the tank wall
7. Heat transfer from the atmosphere to the vapor side of the tank wall

Non-equilibrium tank model assumptions: (source 7,8)

- Temperature at the liquid-vapor boundary is assumed to be the saturation temperature of the vapor.
- Thermal equilibrium exists in the liquid control volume, the vapor control volume, and the surface layer, but these control volumes are not treated as being in thermal equilibrium with each other (except on the first instant at the start).
- No longer assuming an adiabatic tank, heat transfer between liquid and vapor occurs, as well as between the wall and environment.

Non-equilibrium tank model overview:

System of differential equations for the liquid and gas nodes: [8]

Mathematically, the model contains two non-saturated nodes, which equation (4) dictates requires four equations. Therefore we apply conservation of mass and energy separately to each phase:

$$\frac{dm_{vap}}{dt} = \dot{m}_{evap} - \dot{m}_{cond} \tag{24}$$

$$\frac{dm_{liq}}{dt} = -\dot{m}_{evap} + \dot{m}_{cond} - \dot{m}_{outlet} \tag{25}$$

$$\frac{dU_{vap}}{dt} = \dot{m}_{evap}h_{evap} - \dot{m}_{cond}h_{cond} - P\frac{dV_{vap}}{dt} + \dot{Q}_{in,vap} \tag{26}$$

$$\frac{dU_{liq}}{dt} = -\dot{m}_{outlet}h_{outlet} - \dot{m}_{evap}h_{evap} + \dot{m}_{cond}h_{cond} - \boxed{P\frac{dV_{liq}}{dt}} + \dot{Q}_{in,liq} \tag{27}$$

"The work term"

Non-equilibrium tank model overview:

*** The issue with my implementation is found in this step.

volume, entropy, etc. There are two additional constraints on the system. The first is from assuming that the pressure is uniform within the tank, yielding the intensive constraint

$$P_{node\ i} = P_{node\ j} \tag{2}$$

Which is actually $n-1$ constraints, where n is the number of nodes. The second constraint is extensive in nature and arises from the fixed volume of a propellant tank:

$$\sum_i V_{node\ i} = V_{tank} \tag{3}$$

The work term $(P\frac{dV}{dt})$ presents a problem because $\frac{dV}{dt}$ is not explicitly known. One option is to compute it via backwards differences, but this can introduce numerical instability in some situations and therefore a different procedure is desirable. One alternative is to use the pressure and volume constraints (equations (2) and (3)) to derive an equation for $\frac{dV}{dt}$:

$$P_{vap} = P_{liq} \tag{30}$$

$$\frac{dP_{vap}}{dt} = \frac{dP_{liq}}{dt} \tag{31}$$

$$\left[\left(\frac{\partial P}{\partial T}\right)_\rho \frac{dT}{dt} + \left(\frac{\partial P}{\partial \rho}\right)_T \frac{d\rho}{dt}\right]_{vap} = \left[\left(\frac{\partial P}{\partial T}\right)_\rho \frac{dT}{dt} + \left(\frac{\partial P}{\partial \rho}\right)_T \frac{d\rho}{dt}\right]_{liq} \tag{32}$$

[8]

Non-equilibrium tank model overview (continued):

The thermodynamic partial derivatives $\frac{\partial P}{\partial T}$ and $\frac{\partial P}{\partial \rho}$ can be calculated from an equation of state. The time derivative of temperature can be calculated from:

$$\frac{dU}{dt} = u\frac{dm}{dt} + m\frac{du}{dt} \tag{33}$$

$$= u\frac{dm}{dt} + m\left[\left(\frac{\partial u}{\partial T}\right)_\rho \frac{dT}{dt} + \left(\frac{\partial u}{\partial \rho}\right)_T \frac{d\rho}{dt}\right] \tag{34}$$

$$\frac{dT}{dt} = \frac{1}{c_V}\left[\frac{1}{m}\left(\frac{dU}{dt} - u\frac{dm}{dt}\right) - \left(\frac{\partial u}{\partial \rho}\right)_T \frac{d\rho}{dt}\right] \tag{35}$$

And finally $\frac{d\rho}{dt}$ is found from

$$\frac{d\rho}{dt} = \frac{1}{V}\frac{dm}{dt} - \frac{m}{V^2}\frac{dV}{dt} \tag{36}$$

$$\frac{dV_{liq}}{dt} = -\frac{dV_{vap}}{dt} \tag{37}$$

A value of $\frac{dV}{dt}$ must be chosen that solves (32), entering the equation via $\frac{d\rho}{dt}$. In order to compute various properties of the two phases, the thermodynamic state of each must be specified. Given that (35) must be computed to find $\frac{dV}{dt}$, it is convenient to integrate (35) for T_{vap} and T_{liq} rather than integrating (26) and (27) for U_{vap} and U_{liq}. The internal pressure of the tank is readily chosen as the second thermodynamic state variable and can be found by using another version of the tank volume constraint,

$$V_{tank} = \frac{m_{liq}}{\rho_{liq}} + \frac{m_{vap}}{\rho_{vap}} \tag{38}$$

where $\rho_{liq} = \rho(T_{liq}, P)$ and $\rho_{vap} = \rho(T_{vap}, P)$.

[8]

*** The issue with my implementation is found in this step.

Non-equilibrium tank model overview:

The Zilliac & Karabeyoglu model is then a set of six ordinary differential equations that must be integrated: (15) for $T_{w,vap}$, and (15) for $T_{w,liq}$, (24) for m_{vap}, (25) for m_{liq}, (35) for T_{liq}, (35) for T_{vap}.

This model directly calculates the rates of heat and mass transfer between the phases and as a result at any given point in time one or both of the phases may be in a metastable state. Therefore, care is needed in calculating thermodynamic and transport properties.

To handle the metastable states, the Peng Robinson equation of state was used according to [7]

Non-equilibrium tank model overview: Numerical Methods

- The thesis solves this system of 6 differential equations with the Runge Kutta algorithm:
- (Note: instead of solving directly for the time derivative of internal energy dU/dt, the time derivative of temperature (dT/dt) is used instead since it is used elsewhere.

$$\frac{dm_{vap}}{dt} = \dot{m}_{evap} - \dot{m}_{cond} \tag{24}$$

$$\frac{dm_{liq}}{dt} = -\dot{m}_{evap} + \dot{m}_{cond} - \dot{m}_{outlet} \tag{25}$$

$$\frac{dU_{vap}}{dt} = \dot{m}_{evap}h_{evap} - \dot{m}_{cond}h_{cond} - P\frac{dV_{vap}}{dt} + \dot{Q}_{in,vap} \tag{26}$$

$$\frac{dU_{liq}}{dt} = -\dot{m}_{outlet}h_{outlet} - \dot{m}_{evap}h_{evap} + \dot{m}_{cond}h_{cond} - P\frac{dV_{liq}}{dt} + \dot{Q}_{in,liq} \tag{27}$$

$$\frac{dT_w}{dt} = \frac{\dot{Q}_{w,in} - \dot{Q}_{w,out} + \dot{Q}_{w,cond} + \dot{m}_{w,in}c_w(T_{w,in} - T_w)}{m_w c_w} \tag{15}$$

The temperatures of the wall adjacent to the liquid and the wall adjacent to the gas are solved with (15)

Non-equilibrium tank model challenges:

- I found it hard to understand the nuances of the research paper’s language.
- Most difficult challenge: Solving work term / thermodynamic properties of liquid and vapor node, as they must be iteratively solved.

Non-equilibrium tank model: my implementation

I have written this model entirely in one python file. Here is the github repo link: https://github.com/rwright403/king_hybrid/blob/main/src/models/Karabeyoglu-Zilliac-Zimmerman-nos-tank.py

I have isolated the error:

- The secant method used to iteratively solve the work term is always finding the solution to the change in volume with respect to time to be equal to 0.

Therefore, I will focus specifically on this part of the script.

Non-equilibrium tank model numerical methods: my implementation

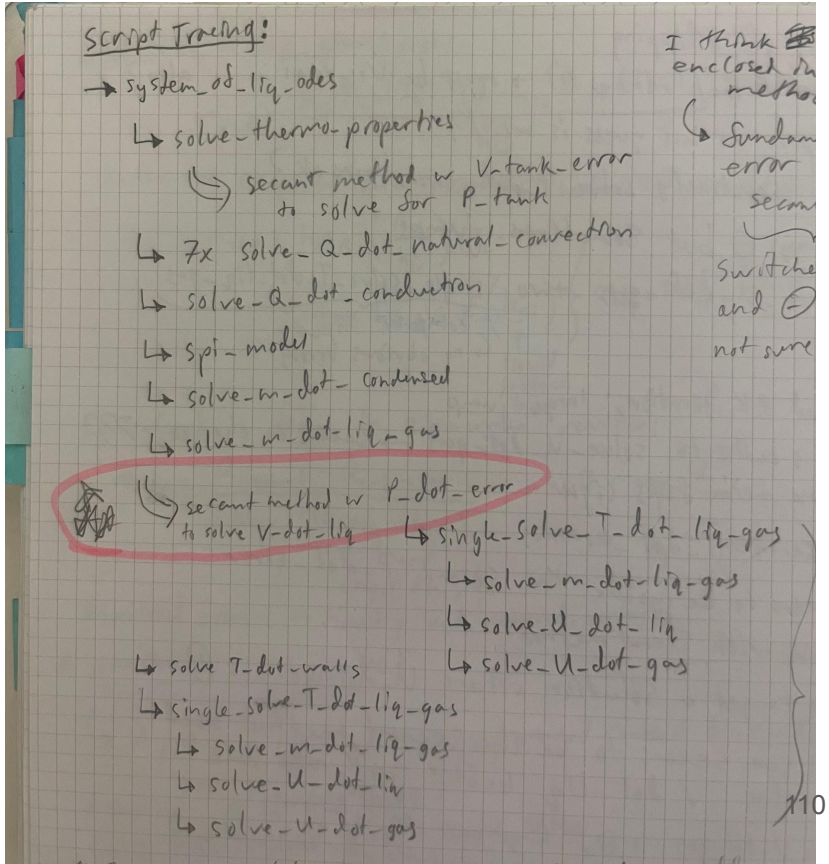
- Explicit Runge-Kutta 45 algorithm from scipy: (https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.RK45.html)
- https://en.wikipedia.org/wiki/Runge%E2%80%93Kutta%E2%80%93Fehlberg_method
- I chose to use this version of the Runge-Kutta algorithm because it was implemented in scipy and the standard version used by the authors was not. This algorithm is more accurate than the original, and similar enough to implement.
- At each timestep, I solve this algorithm for the duration of the timestep so I can couple the tank to the rest of the engine model:

```
#solve with scipy
solution = solve_ivp(debug_system_of_odes, [0, self.timestep], y0, args=(constants, self.iteration), method='RK45')
```

Explicit Model Algorithm

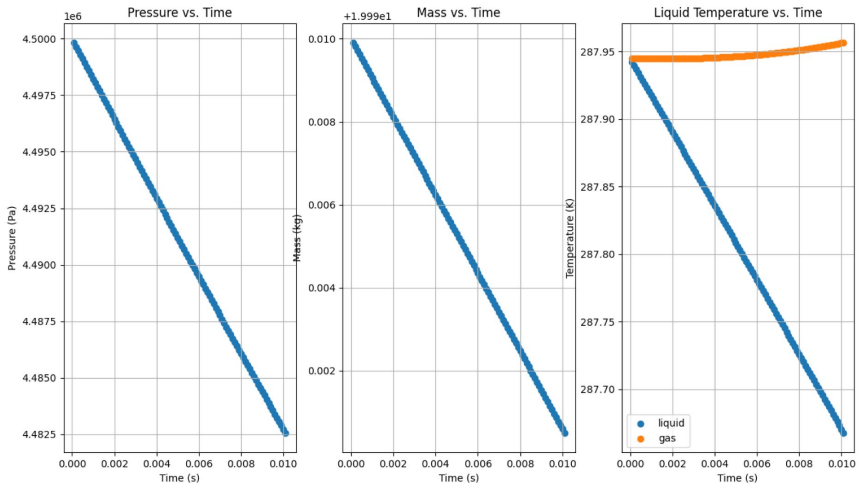
Here is an order of all the function calls in my script in one runge kutta step.

The issue (highlighted) is where I use the secant method to iteratively solve the work term.



Current Non-equilibrium tank model results

Gas node temperature increases as gas expands - this does not make sense, as the Joule Thomson Coefficient of Nitrous Oxide is Positive (the gas cools as it expands under an isenthalpic process)



My implementation output

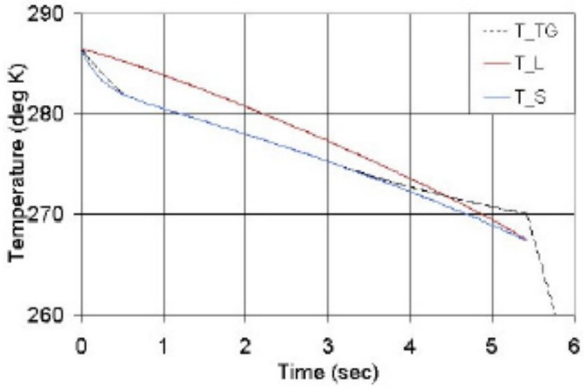


Figure 5. Modeled temperature time history for test 1.

Model output for the same case provided by [7]

Debugging Non-equilibrium tank model

- This is caused by temperature derivatives having incorrect solutions. The gas temperature derivative is positive, while the liquid temperature derivative is extremely large.
- Investigated other possible issues and made fixes to the following before getting the output on the previous page:
  - Heat transfer sign convention
  - Enthalpy reference state
  - Wall node setup

Secant Method loop overview

```
while np.abs(P_dot_error(V_dot_liq, m_liq, m_gas, T_liq, T_gas, P_tank, m_dot_inj, m_dot_evap, m_dot_cond, Q_dot_liq, Q_d
    V_dot_liq = secant((lambda V_dot: P_dot_error(V_dot, m_liq, m_gas, T_liq, T_gas, P_tank, m_dot_inj, m_dot_evap, m_dot
```

I chose the secant method to iteratively solve for dV/dt. Honestly, I do not have a good reason to pick this method other than I have previously successfully used it and I am solving a root:

```
def P_dot_error(V_dot_guess, m_liq, m_gas, T_liq, T_gas, P_tank, m_dot_inj, m_dot_evap, m_dot_cond, Q_dot_liq, Q_dot_gas):
    #print(V_dot_guess)
    V_dot_gas = -V_dot_guess

    T_dot_liq, T_dot_gas = single_solve_T_dot_liq_gas(V_dot_guess, m_liq, m_gas, T_liq, T_gas, P_tank, m_dot_inj, m_dot_evap,

    preos_l = PR(Tc=TC, Pc=PC, omega=OMEGA, T=T_liq, P=P_tank)
    partial_dP_dT_const_v_liq = preos_l.dP_dT_l
    partial_dP_dv_const_T_liq = preos_l.dP_dV_l

    P_dot_liq = partial_dP_dT_const_v_liq*T_dot_liq + partial_dP_dv_const_T_liq*V_dot_guess

    preos_g = PR(Tc=TC, Pc=PC, omega=OMEGA, T=T_gas, P=P_tank)
    partial_dP_dT_const_v_gas = preos_g.dP_dT_g
    partial_dP_dv_const_T_gas = preos_g.dP_dV_g

    P_dot_gas = partial_dP_dT_const_v_gas*T_dot_gas + partial_dP_dv_const_T_gas*V_dot_gas

    return P_dot_liq - P_dot_gas
```

From the thesis, we know that the liquid and gas nodes are at the same pressure. Therefore, dP/dt should be the same for liquid and gas

Secant Method loop overview

Temperature derivatives are solved in the following function according to [7],[8]

```
def single_solve_T_dot_liq_gas(V_dot_liq, m_liq, m_gas, T_liq, T_gas, P_tank, m_dot_inj, m_dot_evap, m_dot_cond, Q_dot_liq, Q

    m_dot_liq, m_dot_gas = solve_m_dot_liq_gas(m_dot_evap, m_dot_cond, m_dot_inj)
    #print("m dot: ", m_dot_liq, m_dot_gas)

    V_dot_gas = -V_dot_liq
    #print("in single solve, ",V_dot_gas)

    U_dot_liq = solve_U_dot_liq(T_liq, P_tank, m_dot_inj, m_dot_evap, m_dot_cond, V_dot_liq , Q_dot_liq)
    U_dot_gas = solve_U_dot_gas(T_gas, P_tank, (-1)*m_dot_evap, (-1)*m_dot_cond, V_dot_gas, Q_dot_gas) #NOTE: convert evap an
    #print("U_dot: ", U_dot_liq, U_dot_gas)

    preos_g = PR(Tc=TC, Pc=PC, omega=OMEGA, T=T_gas, P=P_tank)
    partial_du_dv_const_T_gas = T_gas*preos_g.dP_dT_g - P_tank #NOTE: derivative is partial at constant volume
    n2o_ig_g = Chemical('N2O', T=T_gas)
    Cv_gas = preos_g.Cv_dep_g/MW + n2o_ig_g.Cvg

    u_gas = preos_g.U_dep_g/MW + (n2o_ig_g.H - (R_U*T_liq/MW) )

    preos_l = PR(Tc=TC, Pc=PC, omega=OMEGA, T=T_liq, P=P_tank)
    partial_du_dv_const_T_liq = T_liq*preos_l.dP_dT_l - P_tank #NOTE: derivative is partial at constant volume
    n2o_ig_l = Chemical('N2O', T=T_liq)

    Cv_liq = preos_l.Cv_dep_l/MW + n2o_ig_l.Cvg
    u_liq = preos_l.U_dep_l/MW + (n2o_ig_l.H - (R_U*T_liq/MW) )#NOTE: same warning as ^ but also for this line


    T_dot_liq = (1/Cv_liq)*( (1/m_liq) * (U_dot_liq - u_liq*m_dot_liq) - (partial_du_dv_const_T_liq* V_dot_liq) )
    T_dot_gas = (1/Cv_gas)*( (1/m_gas) * (U_dot_gas - u_gas*m_dot_gas) - (partial_du_dv_const_T_gas* V_dot_gas) )

    return T_dot_liq, T_dot_gas #TODO: TRACE THROUGH T_DOT SOLVING
```

Secant method implementation

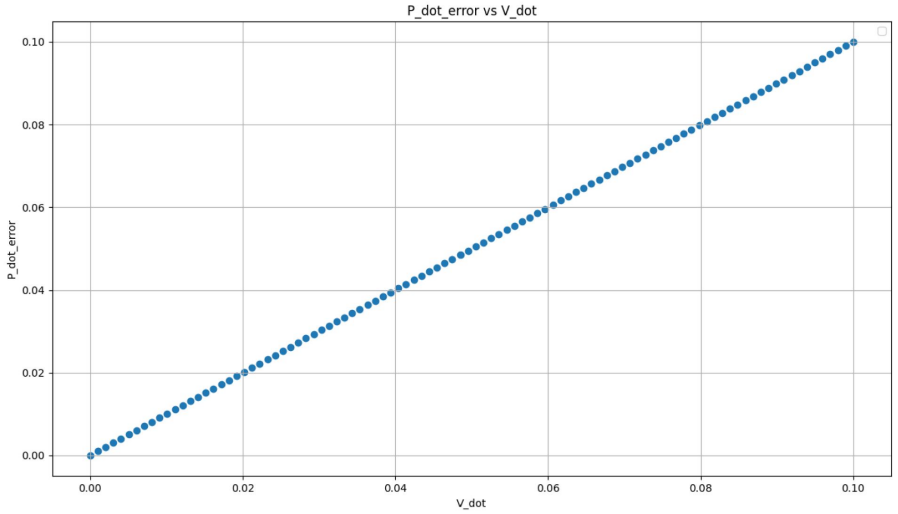
- Exactly from [2], validated

```
def secant(func, x1):
    x_eps = x1 * 0.005  # Set the tolerance to be 0.5% of init guess
    x2 = x1 -x1 * 0.01  # Set a second point 1% away from the original guess
    #print("eval func 1 at start")
    F1 = func(x1)  # Evaluate function at x1
    #print("eval func 2 at start")
    F2 = func(x2)  # Evaluate function at x2
    kk = 1  # Set up counter
    kk_max = 1000

    while np.abs(x2 - x1) >= (x_eps) and kk < kk_max:  # While error is too large and counter is less than max
        x3 = x2 - (F2 * (x2 - x1) / (F2 - F1))
        x1 = x2  # Move everything forward
        x2 = x3
        F1 = F2
        #print("  inside while loop evaluating at", kk)
        F2 = func(x2)
        kk = kk + 1
    x = x2
    return x
```

Diagnosing the secant method issue

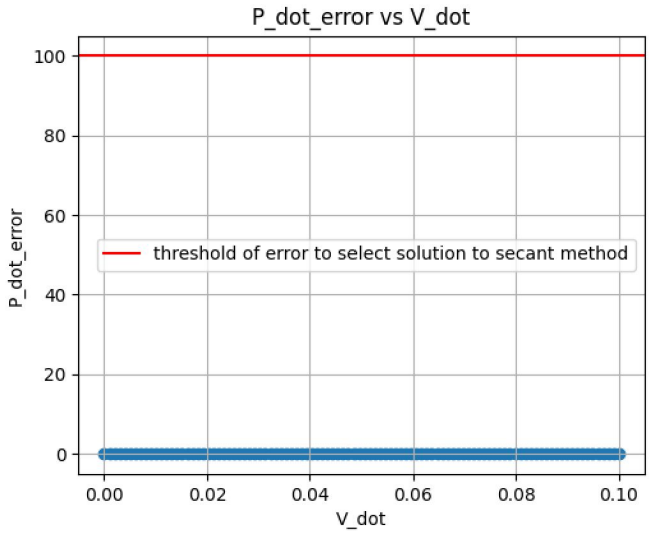
- After triple checking everything inside the secant method loop, i decided to check the solution given by the secant method. I did this by plotting the P_dot_error function against a range of input V_dot values.
- For the other necessary inputs to the secant function, I printed their values as I ran the oxidizer tank script. I looked at a variety of time and runge kutta points, but the solution was the same for all cases.



- A solution of 0 does not make sense for our problem. We expect the volume of the control volumes to change as liquid drains and gas evaporates from the liquid control volume.

Diagnosing the secant method issue

- Interestingly enough, we are not getting a 0 solution to V_dot in our model, so I plotted the same graph with an additional horizontal line to represent the maximum tolerable error.
- The non-zero solution can be explained because the secant method finds the first satisfactory point underneath the threshold of error and continues with the program.
- (This just explains why we are seeing what we are seeing. We are not expecting a solution of 0 for this value so it is wrong regardless)



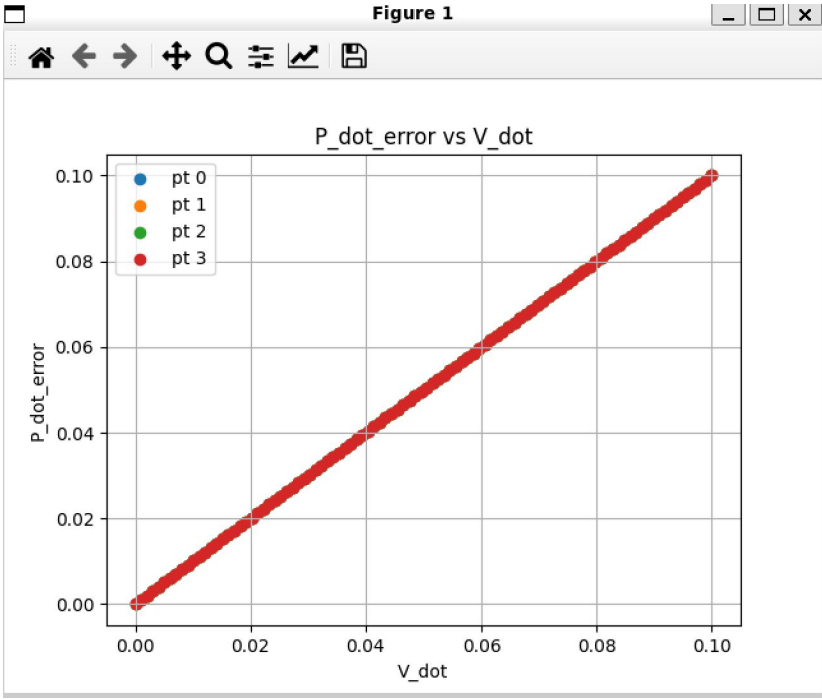
Looking at a variety of input points:

```
if pt == 0:
    m_liq = 18.406207917835648
    m_gas = 1.5937920821643532
    T_liq = 287.94493237008703
    T_gas = 287.94493237008703
    P_tank = 4500000.0
    m_dot_inj = -0.9405973525424474
    m_dot_evap = 0
    m_dot_cond = 0
    Q_dot_liq = 0
    Q_dot_gas = 0

elif pt == 1:
    m_liq = 18.191676519439664
    m_gas = 1.5937920821643532
    T_liq = 281.7425130192451
    T_gas = 287.88275327956984
    P_tank = 4159380.7114885226
    m_dot_inj = -0.9263192433082763
    m_dot_evap = -2.2016418957922097
    m_dot_cond = 0
    Q_dot_liq = -69.43294058099202
    Q_dot_gas = -0.01075874380289937
```

```
elif pt == 2:
    m_liq =  18.40611385540957
    m_gas = 1.5937920852192562
    T_liq = 287.942212869688
    T_gas = 287.9449053987172
    P_tank = 4499821.562928544
    m_dot_inj = -0.940590071802295
    m_dot_evap = -7.118075104972902e-05
    m_dot_cond = 0
    Q_dot_liq = -0.0016943844066170343
    Q_dot_gas = -2.8522725141935474e-07

elif pt == 3:
    m_liq =   18.405266698348967
    m_gas = 1.5937927405909187
    T_liq = 287.91772397175913
    T_gas = 287.9447219652313
    P_tank = 4498218.581200586
    m_dot_inj =  -0.9405249281617616
    m_dot_evap =  -0.0015383701510544964
    m_dot_cond = 0
    Q_dot_liq = -0.04036248059563283
    Q_dot_gas = -5.052822554686893e-06
```



Result: not only same solution but EXACTLY the same output

Potential Problems:

1. Problem is thermodynamically not well posed.
   1.1. I want to say this is not likely because of how closely I followed the thesis, and the amount of checking I have performed on the thermodynamic functions inside the secant method loop.
2. Would it be possible for there to be a problem with using the secant method for this application?
   2.1. I am using the secant method to explicitly solve a time derivative inside each Runge-Kutta step.

- Did I miss a hint/keyword in the thesis on how to approach solving the thermodynamic constraints?
- At this point, I know that my implementation is wrong, but I am not sure exactly why and what to try next.

Non-equilibrium tank model questions:

- Is the secant method being inappropriately applied here? What would some other potential numerical methods to try for this problem?
- Am I making a mistake with how I setup the thermodynamic constraint equations in the secant method?
- Are there any other potential issues with my implementation? Anything else?

Closing

- I firmly believe that the tank model is worth completing and I believe this will reduce the overall model error to something acceptable for proceeding with the detail component design.

- I am excited to enter the detail component phase and learn about engine heat transfer and material selection, and see how to reconcile the model assumptions with reality.

Noteworthy Lessons Learned

A lot of these were learned from other people’s advice, but I feel like I really learned the importance through applying the advice to my work:

- Isolate as much as possible when testing
- Modelling can be extremely time intensive.
- Very wrong things can look very correct on the surface - it is important to check everything and learn to think of creative ways things can be wrong.
- How to understand complex research papers.
- Organization is extremely important when modelling.

Additional Questions:

1) As a unique trait of this engine as a personal project, I do not have any serious project deadlines to meet and investors judging my progress unlike industry. As a student, I have the opportunity to spend time where I need to so I can learn about the theory and system to the level I need. How would a complicated project that required a lot of research and development be kept on track in terms of time and money in industry?

2) What are your thoughts on my development process and the conclusions/actions that I have made up until this point?

3) When the tank model is done, I have two options to proceed. I can use the model to arbitrarily find an optimal design, or I can look into setting up a formal optimization case. My problem would involve a black-box/poorly understood non-linear system with around 20 desvars. Although this is a skill I would like to learn, I am not sure if it would align with the goal of an MVP engine. Is this a good place to start learning this skill? What potential optimization methods should I look into? General thoughts on how to proceed?