Ramjet Performance Analysis Design Tool

MAE 463: Aircraft Propulsion

Professor: Dr. Dahm

Fall 2023

Eric Weissman

Table of Contents

Nomenclature

A_e, A_e	=	Nozzle Exit Area
C_p	=	Specific Heat at Constant Pressure
f	=	Fuel-air Ratio
I_{sp}	=	Specific Impulse
M	=	Mach Number
$\dot{m}$	=	Mass Flux
$\dot{m}_e$	=	Exit Mass Flux
$\dot{m}_f$	=	Fuel Mass Flux
$\dot{m}_i$	=	Inlet Mass Flux
P	=	Static Pressure
P_t, Pt	=	Total Pressure
q_f, q_f	=	Heating Value
P	=	Static Pressure
R	=	Air Specific Gas Constant
S	=	Entropy
T	=	Static Temperature
T_t, Tt	=	Total Temperature
$TSFC$	=	Thrust Specific Fuel Consumption
V	=	Air Speed
$( )_i$	=	Property of State i
$( )_{ij}$	=	Property from State i to j
η, *eff.*	=	Efficiency
η_d	=	Inlet/Diffuser Isentropic Efficiency
η_n	=	Nozzle Isentropic Efficiency
η_o	=	Overall Efficiency
η_p	=	Propulsive Efficiency
η_{th}	=	Thermal Efficiency
γ	=	Specific Heat Ratio

Introduction

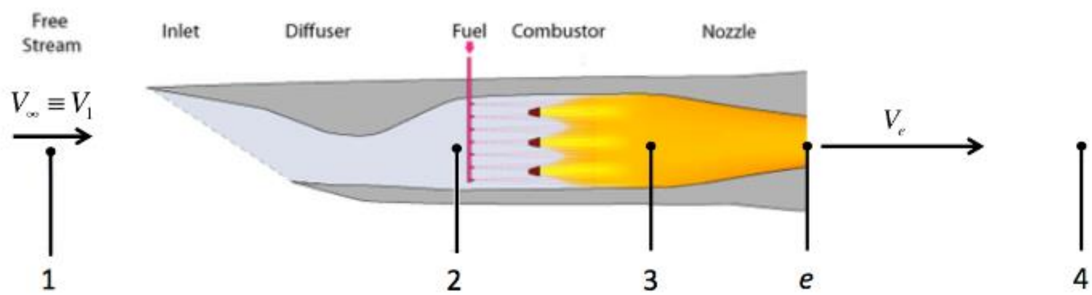


Figure 1: Simplified ramjet/ scramjet with labeled states

Ramjets and Scramjets are a type of air breathing propulsion system which operates on the Brayton cycle, with the compression before the combustion chamber coming exclusively from ram compression (the compression caused by slowing the free stream air down). This permits the engines to operate without a compressor, and therefore without a turbine, meaning the engines contain virtually no moving parts. However, due to the compression required for Brayton cycle gas generators to produce large amounts of usable work, the difference in the Mach number between stations 1 and 2 (see fig 1. for explanation of stations) must be sufficiently high enough to generate an adequate compression ratio. Therefore, ramjets/ scramjets (going forward these two systems will be lumped together and referred to as ramjets exclusively) are particularly attractive in supersonic applications such as supersonic and hypersonic cruise missiles due to the high Mach regime capabilities, low cost due to minimal moving parts, and their abilities to achieve high efficiencies compared to non-airbreathing missiles. However, to properly realize the benefits of this propulsion system, a means to optimize the various input parameters is needed. Therefore, this report seeks to demonstrate a computational design tool built to model the flow through a ramjet engine, and the resulting performance with the following inputs and outputs:

Inputs	Outputs
• Altitude • η_d • η_n • A_e • M_1 • M_2 • T_{t3max} • q_f	• T_i • T_{ti} • P_i • P_{ti} • V_i • M_i • Cp_i • Power • Thrust • I_{sp} • TSFC • f • η_o • $\dot{m}_e$ • $\dot{m}_f$ • $\dot{m}_i$

This set of inputs and outputs will allow for a preliminary optimal design of a ramjet for a given mission profile. In order to demonstrate the validity and uses of this tool, this report will (1) demonstrate that this tool is accurate by comparing the results to known quantities, (2) explain the atmospheric model used, (3) show how the tool can be used to show the relationship between input and output parameters, and (4) use the tool to show how primitive optimizations can be performed.

Analysis and Discussion

(a) Design Tool Validation

The first step to analyzing this design tool is to validate its results against the expected results for multiple cases. To do this I have provided below the results of the tool for two cases: (1) a thermally unchoked, and (2) a thermally choked design point, and have compared the results to the known solutions provided. The results show that my design tool returns solutions accurate to within three significant figures to the expected results for both cases.

Case 1: Thermally Unchoked

Inputs	Flight Altitude (m)	Diffuser Eff.	Nozzle Eff.	A_e (m^2)	M_1	M_2	T_{t3max} (k)	q_f (J/kg)
	4300	0.9200	0.9400	0.0150	2.4000	0.1500	2400	42300000

Model Outputs							
States	**T(k)**	**T_t(k)**	**P (kPa)**	**P_t (kPa)**	**V (m/s)**	**M**	**Cp (J/kg-k)**
1	245.8976	529.1716	58.2607	851.7726	754.2541	2.4000	1.0042e+03
2	526.8010	529.1716	719.3962	730.7906	68.9992	0.1500	1.0042e+03
3	2.3543e+03	2400	719.3962	781.8888	337.0739	0.3597	1.4074e+03
e	2.0870e+03	2400	409.2753	749.9625	882.2540	1	1.3596e+03
4	1.5577e+03	2400	58.2607	379.2094	1.4472e+03	1.8987	1.2648e+03

ΔS_{12} (J/kg-k)	ΔS_{23} (J/kg-k)	ΔS_{3e} (J/kg-k)	ΔS_{e4} (J/kg-k)	ΔS_{13} (J/kg-k)	ΔS_{1e} (J/kg-k)	ΔS_{14} (J/kg-k)
43.9510	2.1085e+03	11.9606	195.6470	2.1525e+03	2.1644e+03	2.3601e+03

Thermal Eff.	Propulsive Eff.	Overall Eff.	Power (W)	Thrust (N)	I_{sp} (1/s)	TSFC (kg/hr-N)	f	q_{23} (J/kg)
0.3627	0.6799	0.2466	5.1139e+06	6.7801e+03	1.4619e+03	0.2513	0.0552	2.335e+06

Inlet Mass Flux (kg/s)	Outlet Mass Flux(kg/s)	Fuel Mass Flux (kg/s)
8.5728	9.0460	0.4732

Expected Outputs							
States	T(k)	T_t(k)	P (kPa)	P_t (kPa)	V (m/s)	M	Cp (J/kg-k)
1	245.9	529.2	58.3	851.8	754.3	2.4	1.004e+03
2	526.8	529.2	719.4	730.8	69.0	0.15	1.004e+03
3	2.354e+03	2400	719.4	781.9	337.0739	0.3597	1.407e+03
e	2.087e+03	2400	409.3	750.	882.3	1	1.360e+03
4	1.557e+03	2400	58.3	379.2	1.447e+03	1.90	1.265e+03

ΔS_{12} (J/kg-k)	ΔS_{23} (J/kg-k)	ΔS_{3e} (J/kg-k)	ΔS_{e4} (J/kg-k)	ΔS_{13} (J/kg-k)	ΔS_{1e} (J/kg-k)	ΔS_{14} (J/kg-k)
43.95	2.108e+03	12.0	195.6	2.152e+03	2.164e+03	2.360e+03

Thermal Eff.	Propulsive Eff.	Overall Eff.	Power (W)	Thrust (N)	I_{sp} (1/s)	TSFC (kg/hr-N)	f	q_{23} (J/kg)
0.362	0.680	0.246	5.11e+06	6.773e+03	1.490e+03	0.247	0.054	2.34e+06

Inlet Mass Flux (kg/s)	Outlet Mass Flux(kg/s)	Fuel Mass Flux (kg/s)
8.6	9.0	0.46

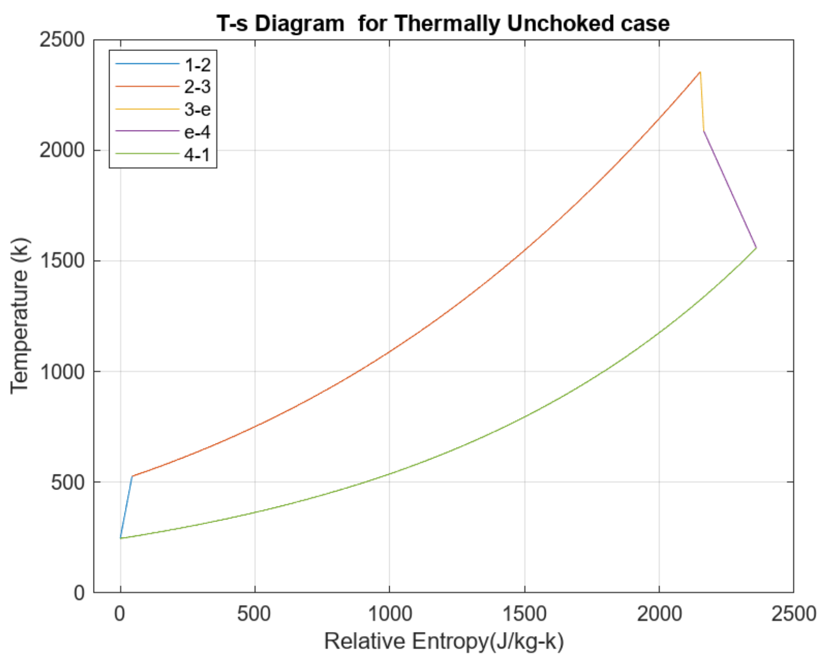


Figure 2: T-S diagram for thermally unchoked case

Case 2: Thermally Choked

Inputs	Flight Altitude (m)	Diffuser Eff.	Nozzle Eff.	A_e (m^2)	M_1	M_2	T_{t3max} (k)	q_f (J/kg)
	4300	0.9200	0.9400	0.0150	2.4000	0.4000	2400	42300000

Model Outputs

States	T(k)	T_t(k)	P (kPa)	P_t (kPa)	V (m/s)	M	Cp (J/kg-k)
1	245.8976	529.1716	58.2607	851.7726	754.2541	2.4000	1.0042e+03
2	512.7632	529.1716	654.5066	730.7906	181.5297	0.4000	1.0042e+03
3	890.9530	1.0246e+03	654.5066	1.1993e+03	576.4536	1	1.1455e+03
e	890.9530	1.0246e+03	627.7816	1.1504e+03	576.4536	1	1.1455e+03
4	634.5804	1.0246e+03	58.2607	464.5100	984.7643	2.0242	1.0996e+03

ΔS_{12} (J/kg-k)	ΔS_{23} (J/kg-k)	ΔS_{3e} (J/kg-k)	ΔS_{e4} (J/kg-k)	ΔS_{13} (J/kg-k)	ΔS_{1e} (J/kg-k)	ΔS_{14} (J/kg-k)
43.9510	614.7385	11.9606	260.1737	658.6895	670.6501	930.8238

Thermal Eff.	Propulsive Eff.	Overall Eff.	Power (W)	Thrust (N)	I_{sp} (1/s)	TSFC (kg/hr-N)	f	q_{23} (J/kg)
0.3603	0.8705	0.3136	3.7526e+06	4.9753e+03	1.8382e+03	0.1998	0.0132	5.573e+05

Inlet Mass Flux (kg/s)	Outlet Mass Flux(kg/s)	Fuel Mass Flux (kg/s)
20.9601	21.2363	0.2762

Expected Outputs

States	T(k)	T_t(k)	P (kPa)	P_t (kPa)	V (m/s)	M	Cp (J/kg-k)
1	245.9	529.2	58.3	851.8	754.3	2.4	1.004e+03
2	512.8	529.2	654.5	730.8	181.5	0.4	1.004e+03
3	891	1.025e+03	654.5	1.199e+03	576.5	1	1.146e+03
e	891	1.025e+03	627.8	1.150e+03	576.5	1	1.146e+03
4	634.6	1.025e+03	58.3	464.5	984.8	2.02	1.100e+03

ΔS_{12} (J/kg-k)	ΔS_{23} (J/kg-k)	ΔS_{3e} (J/kg-k)	ΔS_{e4} (J/kg-k)	ΔS_{13} (J/kg-k)	ΔS_{1e} (J/kg-k)	ΔS_{14} (J/kg-k)
43.95	614.7	12.0	260.	659.	671.	931.

Thermal Eff.	Propulsive Eff.	Overall Eff.	Power (W)	Thrust (N)	I_{sp} (1/s)	TSFC (kg/hr-N)	f	q_{23} (J/kg)
0.360	0.870	0.312	3.75e+06	4.971e+03	1.875e+03	0.196	0.013	5.57e+05

Inlet Mass Flux (kg/s)	Outlet Mass Flux(kg/s)	Fuel Mass Flux (kg/s)
21.0	21.2	0.27

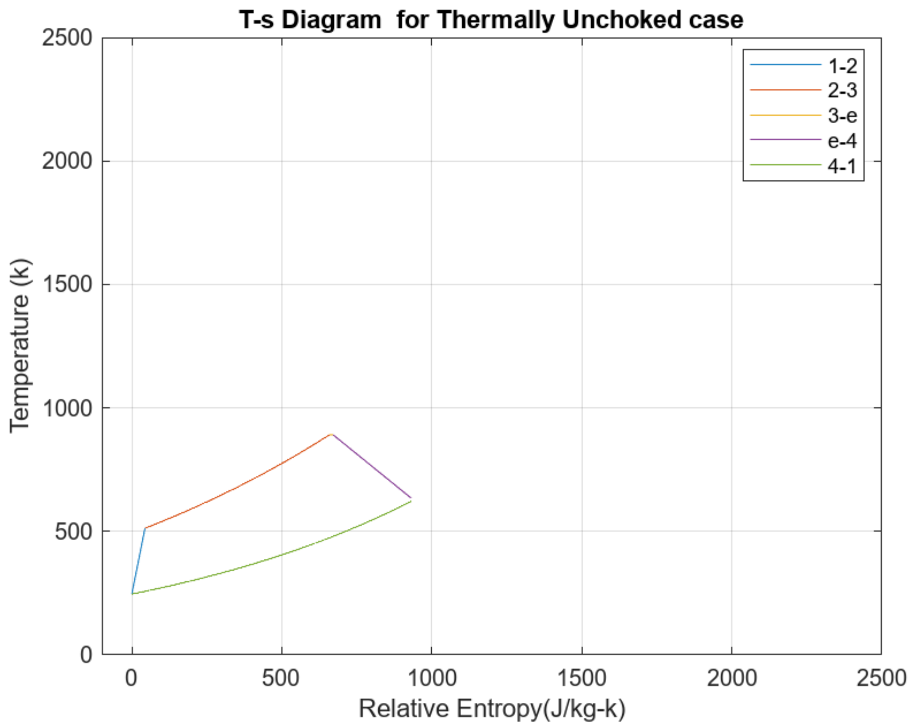


Figure 3: T-S diagram for thermally choked case

The only difference between case 1 and 2 was the Mach number of the air entering the combustor (M_2). The higher M_2 in case 2 meant that, due to Rayleigh flow, the flow is thermally choked, and no further heat can be added to the gas, which appears to have reduced the net work produced by the engine as shown by comparing the power output and the area enclosed in the T-s diagrams (figures 2 and 3). This result follows intuition as the net work must be equal to net heat into and out of the cycle, so if the nozzle is thermally choked, and therefore unable to accept more heat in, then the net heat into the system will be smaller in case 2, ergo the net work in case 1 must be greater than the work in case 2. Nevertheless, having shown that the design tool is able to accurately model a ramjet and that the results from the tool match the expected results, we can begin to employ the tool to analyze the relationship between some of the input and output parameters and demonstrating the features of the tool.

(b) Validation of Atmospheric Model

The ramjet model relies on a two-part model of the atmosphere to predict the static temperature and pressures in state 1 for a given altitude. The first part of the model applied up to 7958m and relies on the concept of a fluid element moving vertically up through the atmosphere at a rate slow enough for the process to be reversible and assumed adiabatic (i.e., isentropic). This results in the following relationships:

$$\frac{p(z)}{p_s} = \left[1 - \frac{\gamma - 1}{\gamma}(z / z^*)\right]^{\frac{\gamma}{\gamma-1}}$$

$$\frac{T(z)}{T_s} = \left[1 - \frac{\gamma - 1}{\gamma}(z / z^*)\right]$$

Where P_s, T_s, and Z^* are all constants. However, after approximately 7958m the assumption that the atmosphere is adiabatic no longer holds due to solar radiation transferring heat into the atmosphere. Therefore, a second empiric model is used to account for the change in physics:

$$T(z) = 210K$$

$$P(z) = 33.6e^{-\frac{z-7958}{6605}}$$

Figures 4 and 5 show the resulting atmospheric models along with a comparison to the "U.S. Standard Atmospheric" model, which is another empiric model of the atmosphere. While the two models are close, the temperature model can vary as much as 25K at points. That said, the standard atmosphere model itself is imperfect as it does not account for the natural variability of the troposphere that can occur due to geographical changes or changes in weather. Therefore, for the purposes of this demonstration, this atmospheric model works well enough, however this does illustrate an issue with this tool: All the input parameters fail to account for uncertainties, which would be required for a more accurate design tool. As will be discussed below, future iterations of this tool would need to account for these uncertainties, potentially by using a Monte-Carlo approach.

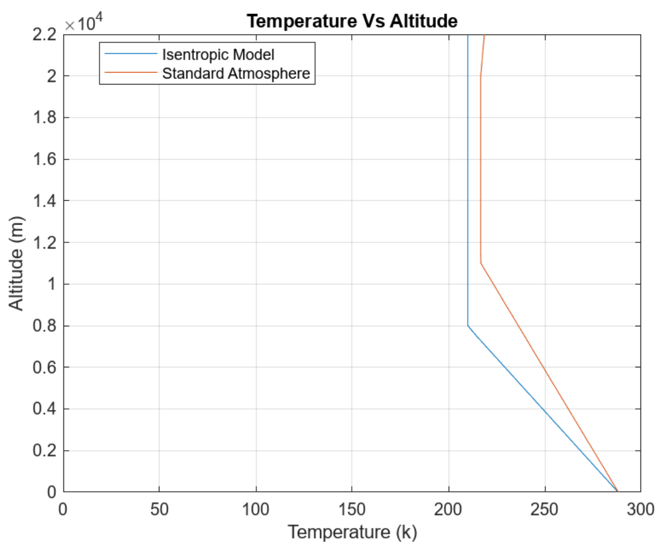


Figure 4: Comparison of isentropic and "U.S. Standard Atmosphere" atmospheric temperature models

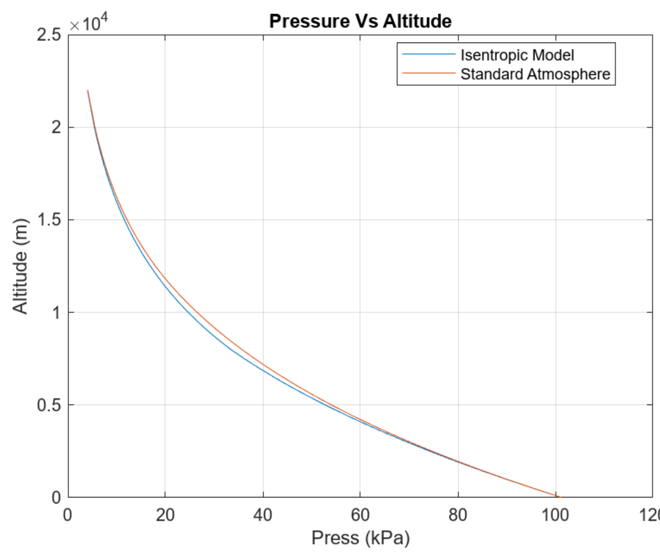


Figure 5: Comparison of isentropic and "U.S. Standard Atmosphere" atmospheric Pressure models

(c) Flight Mach Number Effects

Having demonstrated the validity of this design tool, we can now begin to use it to show something interesting: how the performance of a ramjet engine varies as a function of the flight Mach number. To achieve this, the inputs from case 1 (the thermally unchoked case) were used and M_1 was varied from 0.8 to 5 in 0.1 increments and the resulting thrust, overall eff., and TSFC (hereafter referred to as the "performance indicators") were recorded (see figures 6, 7, and 8).

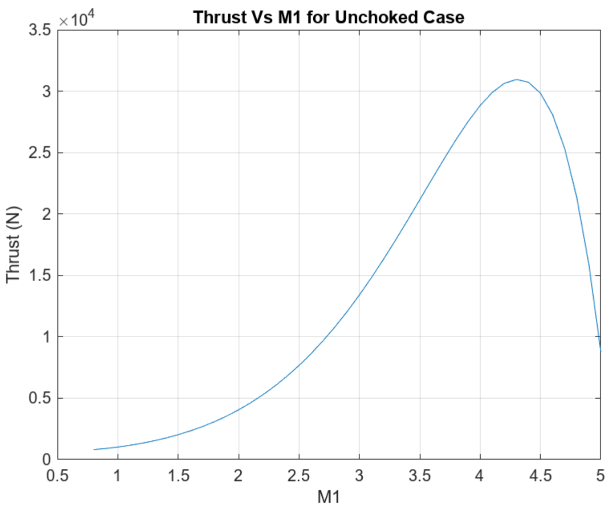


Figure 6: Thrust vs M1 for case 1

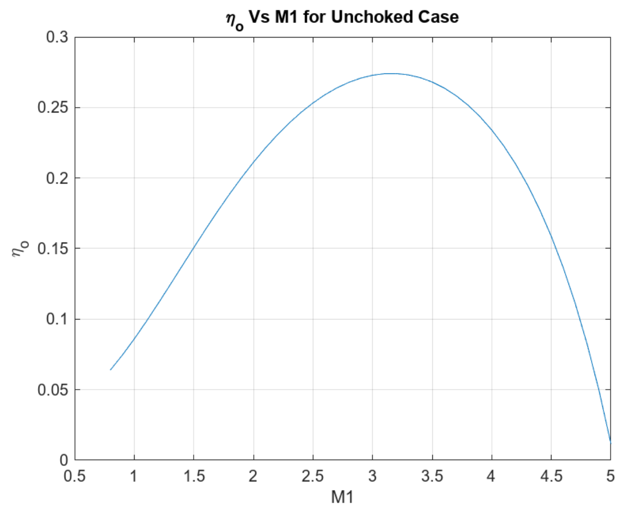


Figure 7: Overall efficiency vs M1 for case 1

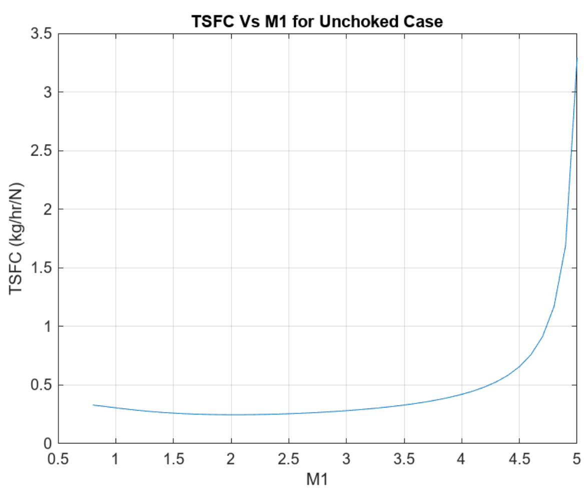


Figure 8: TSFC vs M1 for case 1

It is interesting to note how each of these performance statistics initially improve as the Mach number increases, and then dramatically "worsen" around Mach 4 to Mach 5. This makes sense as the compression of the air prior to the combustion chamber depends on the flight Mach number. The compression can loosely be compared to the OPR in turbojets, which strongly controls the thermal efficiency and performance of the engine, so it makes sense that as our Mach number increases and subsequently the compression increases, so too does the performance of the engine. Furthermore, the drop in performance can be explained by V_e remaining relatively constant across the range of Mach numbers (the nozzle is choked, thus $M_e = 1$) while V_1 is linear to M_1 because the atmospheric temperature is held constant. While the effects of this are drowned out at lower Mach numbers due to the dramatic improvement in performance due to the compression, at higher flight speeds the compression begins to have marginal returns, while the difference between V_e and V_1 becomes negative around Mach 3. This velocity difference contributes to the jet thrust and can be found in the equations for all three of these performance indicators in one way or another.

(d) Flight Altitude Effects

Likewise, we can analyze the effects that flight altitude has on the same performance statistics as discussed in the previous section. The results will be analyzed for the same input parameters and the altitude will be varied from 2 to 30 km in 1 km increments.

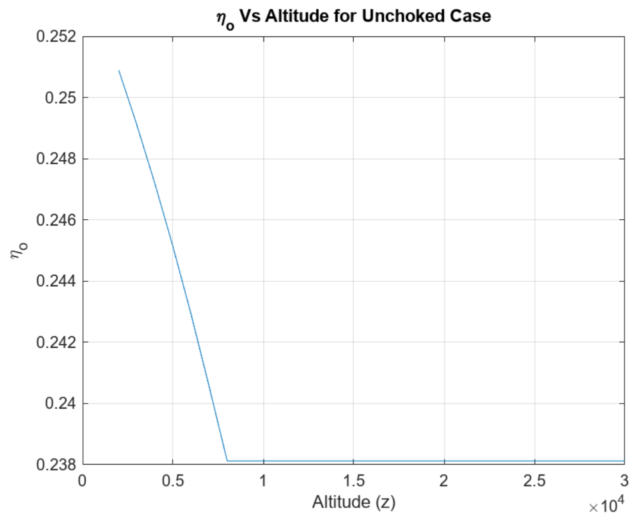


Figure 9: Overall efficiency vs altitude for case 1

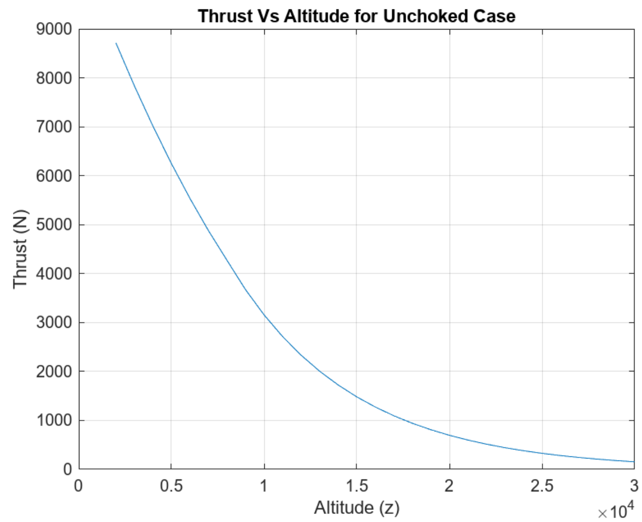


Figure 10: Thrust vs altitude for case 1

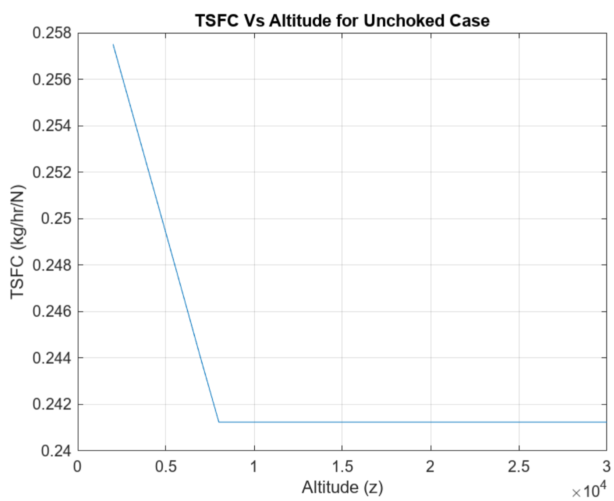


Figure 11: TSFC vs altitude for case 1

As shown in figures 4 and 5, the atmospheric temperature varies linearly with altitude until approximately 8 km, while the pressure varies exponentially. Noting this, figures 9 and 11, depicting the overall eff. and the TSFC as a function of flight altitude respectively, appear to decrease linearly until 8km, after which both curves are constant. This suggests that both performance indicators are linear (with a negative slope) to the external temperature and are independent of P_1. Likewise, the thrust curve visually appears identical in form to the pressure vs altitude curve in fig. 5. This should come as no surprise as the mass flow rate through the engine will decrease proportionally to the external pressure, which can be seen in the mass flow equation:

$$\dot{m}_e = \rho_e V_e A_e = \frac{p_e}{RT_e} V_e A_e$$

It is important to note that P_e and P_1 are proportional to one another, so as P_1 decreases with altitude, so does P_e.

(e) Optimizing Overall Efficiency and TSFC

The next use for the design tool is to optimize performance parameters. For this simple demonstration we will perform single variable optimizations for both maximizing the overall eff. and minimizing TSFC as functions of M_1 ($\in [0.8, 5]$) and altitude ($\in [2km, 20km]$). All other inputs are the same as the previous analysis.

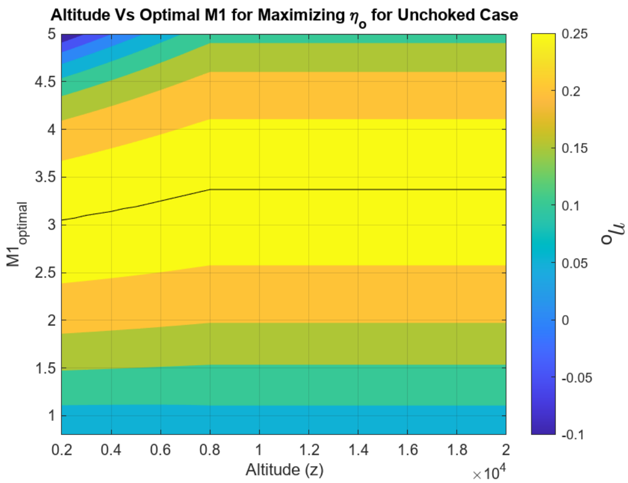


Figure 12: Altitude vs optimal M1 to maximize overall efficiency for case 1. Black line follows maximum values of η_o for each altitude

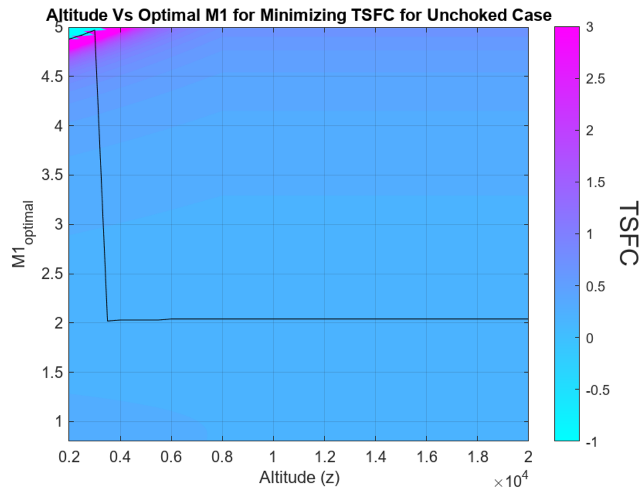


Figure 13: Altitude vs optimal M1 to minimize TSFC for case 1. Black line follows minimum values of TSFC for each altitude. Color-bar limits artificially forced between [-1, 3] for clarity.

Like the previous analysis, it appears that the form of fig. 12 is close to the shape of fig. 4, with the optimal M_1 changing linearly with altitude, until 8km is reached, at which point M_1 stops varying with altitude. This indicates that the optimal M_1 for overall eff. is linearly proportional to external temperature (again with a negative slope). Conversely, fig. 13, showing the optimal M_1 for each altitude which minimizes TSFC, has a less trend, with the ideal Mach number being nearly hypersonic while low in the atmosphere, and then dramatically dropping to near Mach 2 around 3500m, at which point it remains relatively constant throughout the rest of the altitudes. Looking deeper into the results, it appears that the nozzle becomes choked at the aforementioned "dramatic drop" in M_1, which is because an unchoked nozzle at lower altitude is going to be able to allow a significantly higher mass flux, thereby improving the performance of the engine. This result, I suspect, is unrealistic as this tool assumes that the inlet efficiency is constant, however, it would be difficult to fly near hypersonic speeds close to sea level without forming normal shocks in the inlet. This means there is some relationship between the inlet/ diffuser isentropic efficiency, the inlet/ diffuser geometry, and the flight Mach number which this tool has neglected to include.

(f) Inlet/Diffuser Isentropic Efficiency Effects

Likewise, we can analyze the effects that inlet/ diffuser isentropic eff. has on the same performance indicators as discussed in the previous sections, again using the same input parameters from the thermally unchoked scenario. For this analysis the η_d was varied from 0.5 to 1 in 0.05 increments.

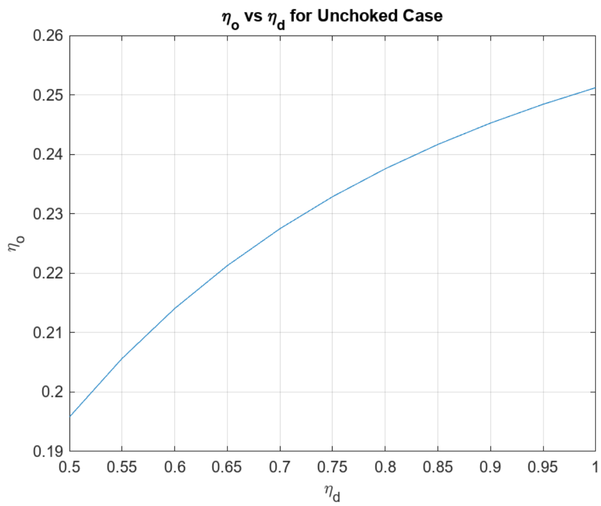


Figure 14: Overall eff. vs diffuser eff. for case 1

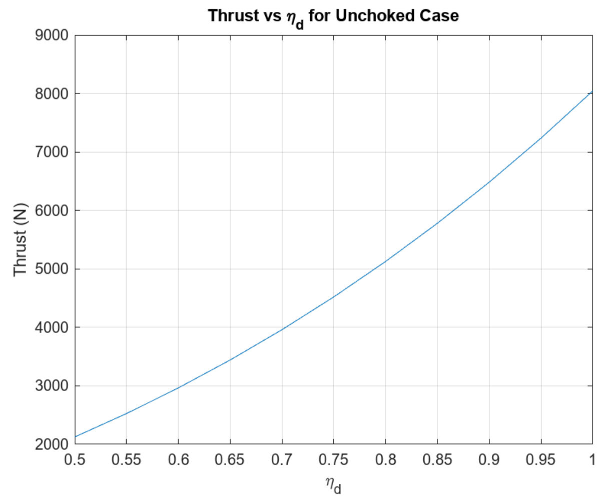


Figure 15: Thrust vs diffuser eff. for case 1

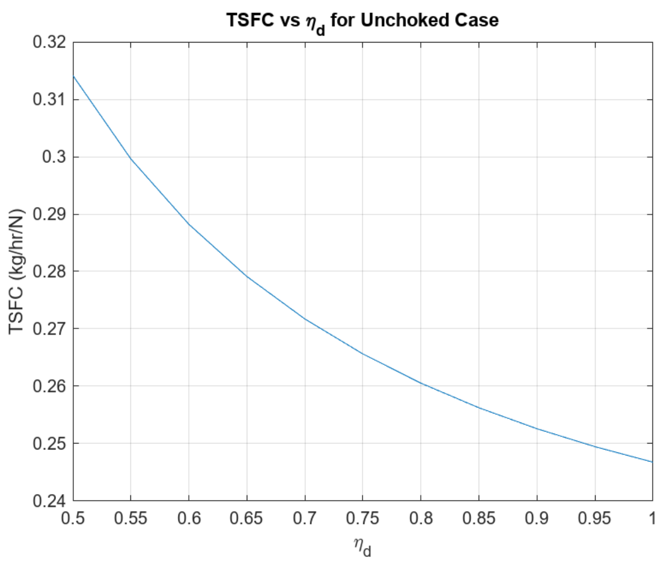


Figure 16: TSFC vs diffuser eff. for case 1

The results largely match expectations, with higher eff. resulting in better performance, however the nonlinearity of the relationships is surprising. It appears that an incremental improvement of the inlet isentropic eff. has a much stronger effect on the overall eff. and on the TSFC when the inlet isentropic eff. is low. Conversely, an incremental improvement of the inlet isentropic eff. has a much stronger effect on the thrust when the inlet eff. is already high. Furthermore, it was surprising to see the huge impact that the inlet isentropic eff. had on the thrust, with a near 300% increase in thrust from improving the inlet isentropic eff. from 0.5 to 1. However, this does make sense as we have seen that the pressure ratio from state 1 to state 2 is a major contributor to performance, so decreasing pressure loss from state 1 to 2 is expected to have a large impact on performance.

(g) Nozzle Isentropic Efficiency Effects

Similar to the previous section, we can analyze the effects that nozzle isentropic eff. has on the same performance, again using the same input parameters from the thermally unchoked scenario. For this analysis the η_n was varied from 0.5 to 1 in 0.05 increments.

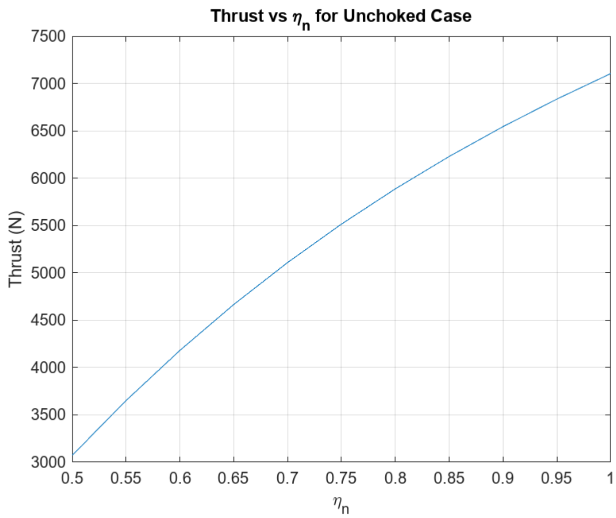


Figure 17: Thrust vs nozzle eff. for case 1

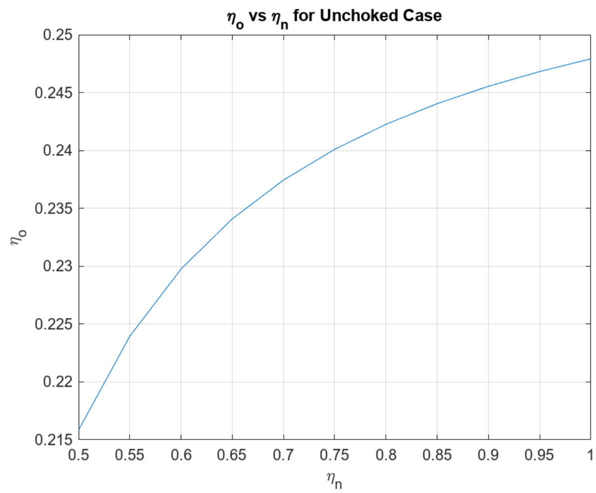


Figure 18: Overall eff. vs nozzle eff. for case 1

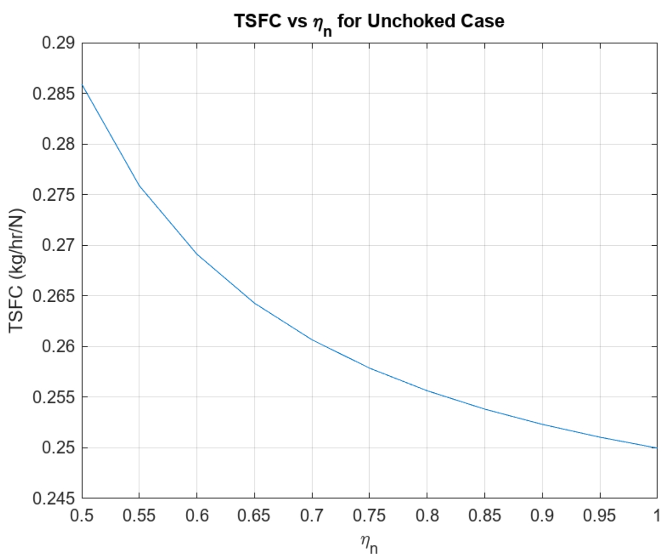


Figure 19: TSFC vs nozzle eff. for case 1

Like the previous section, we can see that a higher isentropic eff. can significantly improve the performance of the propulsion system, however the trends are slightly different for the nozzle isentropic eff., as compared to the inlet/ diffuser isentropic eff. For all three performance indicators, as the nozzle isentropic eff. increases, the marginal improvement in our indicators decreases. Additionally, the nozzle isentropic eff. appears to have a slightly less pronounced influence as compared to the inlet/ diffuser isentropic eff. (e.g., the change in thrust is only about 150% across the range of η_n values).

(h) Effects of Mach Number Entering Combustion Chamber

This next analysis looks at the effects of varying the Mach number of the flow entering the combustor over a range of M_2 value from 0.1 to 2.5 in 0.1 increments. Again, this analysis used the same input parameters as the thermally unchoked case.

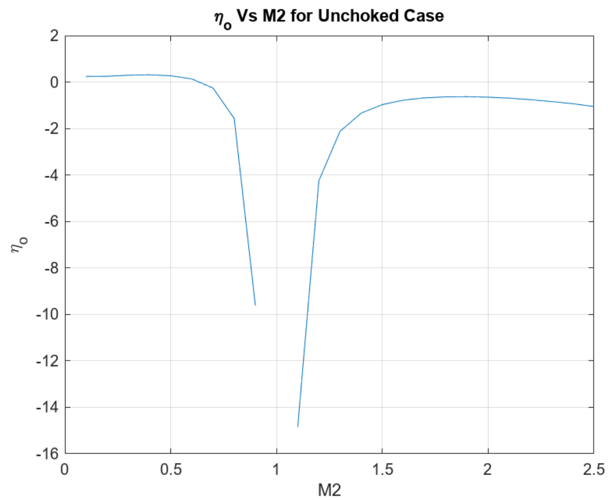


Figure 20: Overall eff. vs M2 for case 1

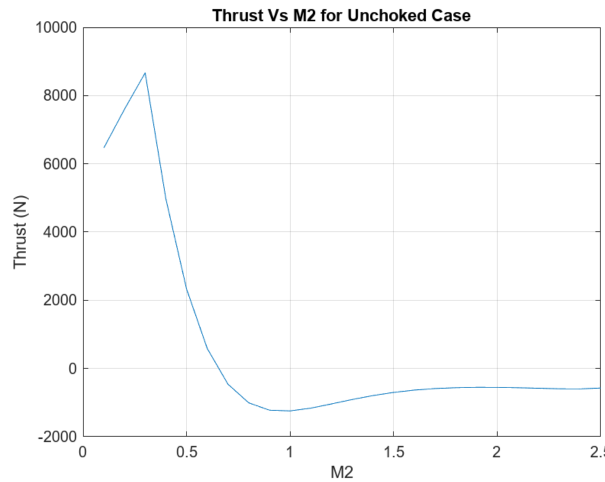


Figure 21: Thrust vs M2 for case 1

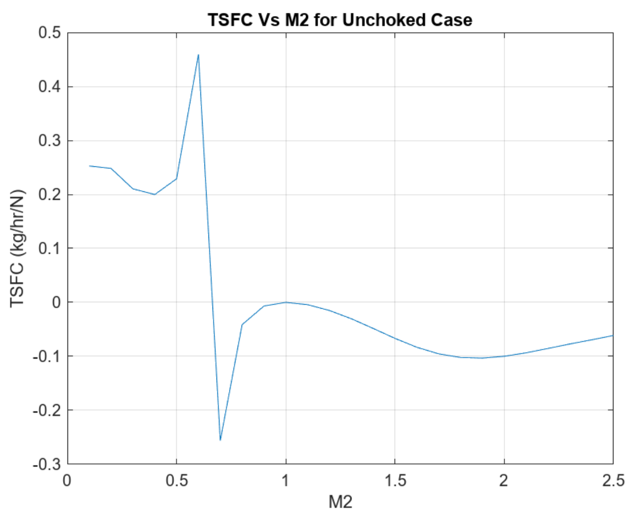


Figure 22: TSFC vs M2 for case 1

Clearly from figures 20, 21 and 22, the effect of M_2 on the performance of the propulsion system is relatively complicated, with all three of the analyzed performance indicators demonstrating asymptotic behavior at different M_2 values. That said, the thrust, TSFC and overall eff. perform better for subsonic M_2 values and are negative for supersonic M_2 values. This result makes sense as the flight Mach number is only 2.4, so the compression from state 1 to state 2 becomes very small or even below 1 for the higher M_2 values. Likewise, the higher M_2 values mean that the engine becomes thermally choked at much lower T_{t3} values, indicating that less heat can be transferred into the air, again significantly reducing the performance of the system.

(i) Example of Design Tool use in Optimization

The final demonstration is the ability for the tool to be modified for different types of optimization objectives. In this scenario, the tool will be used to optimize the values of M_2 and the T_{t3} (closely related to f) for two different objectives: (1) the maximization of overall efficiency, and (2) the maximization of thrust. The design space and scenario parameters are found below:

$$M_2 \in [0.1, 2.4]$$

$$T_{t3} \in [1500, 2400]$$

Inputs:

Flight Altitude (m)	Diffuser Eff.	Nozzle Eff.	A_e (m^2)	M_1	q_f (j/kg)
27400	0.9200	0.9400	0.0150	5	42300000

To perform this optimization, some adjustments were made to the tool which took T_{t3} as an input in place of $T_{t3,\,max}$. Then after the combustor was checked for thermal choking, if the nozzle was not thermally choked, then the T_{t3} temperature was used instead of $T_{t3,\,max}$, as was done in the original design tool. This analysis can be thought of as a circuitous manner to investigate the effects of changing the fuel mass flux without having to significantly overhaul the design tool.

Overall Efficiency Optimization

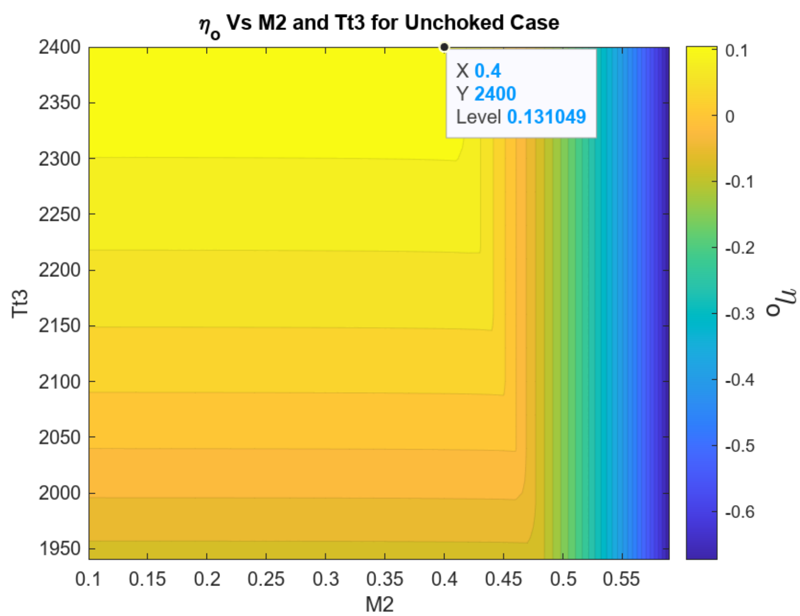


Figure 23: Contour plot showing overall eff. as a function of M2 and Tt3 (zoomed in for clarity)

Max Efficiency	Optimal M_2	Optimal T_{t3} (k)
0.1310	0.4000	2400

In the first optimization (performed using a full factorial sampling), the optimal M_2 and T_{t3} was found to be at 0.4 and 2400K respectively, which gave a max overall eff. of 0.131. A higher T_{t3} will give a higher thermal efficiency, so it is no surprise that the optimal value was found on the edge of the design space. Likewise, the optimal M_2 value was found just before the asymptote described in the previous section, meaning that this is the highest M_2 value where the both the mass flow through the engine is high and the engine is not thermally choked.

Thrust Optimization

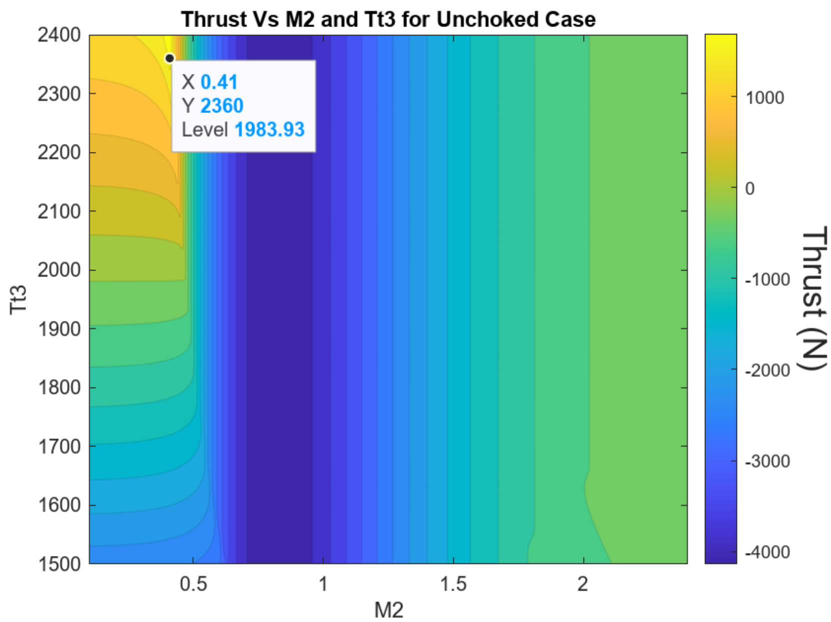


Figure 24: Contour plot showing thrust as a function of M2 and Tt3

Max Thrust (N)	Optimal M_2	Optimal T_{t3} (K)
1.9839e+03	0.4100	2360

In this optimization, the optimal M_2 and T_{t3} was found to be at 0.41 and 2360K respectively, which gave a max thrust of 1984N. It is interesting to note the large gradients near to optimal solution. This would indicate that small perturbations in T_{t3} or in M_2 could dramatically affect the thrust output of the ramjet and noting the imperfection of manufacturing, turbulence, and all the other small sources of error, this might indicate that this solution is in fact not a usable case. Therefore, to include the effects of uncertainties in the optimization, in place of this analytical optimization (where the model used is deterministic), instead a more sophisticated algorithm could be used that relies on a statistical approach. Depending on the level of sophistication desired, this could mean a simple Bayesian optimization, or a more complex Monte-Carlo Bayesian Optimization. Of course, to implement these algorithms would require a more nuanced description of the problem to be optimized that is beyond the scope of this report.

Conclusion

While this report has demonstrated some of the interesting and useful functions this ramjet design tool could have, it is important to note the assumptions and short coming of this model, namely:

- The combustor was assumed to have no irreversibility's.
- It was assumed the flow through sections had a constant γ, R, and C_p.
- The model neglects the larger design problem of how these different inputs would affect the vehicle. For example, drag increases dramatically in higher Mach regimes, however this has been largely ignored in this analysis. Therefore, while this tool might suggest that a certain flight Mach number is the most efficient or has a higher thrust, that is only for the propulsive system and potentially not for the overall design.

Therefore, the tool clearly has some space for future growth, including addressing the above issues. Additionally future work could include more sophisticated optimization methods which could be considered which would allow for a large design space to be explored. In particular, because the observed discontinuities (the asymptotic behavior described above) appear to occur far from the optimal solutions, a method such as sequential quadratic programming is an attractive choice for a first pass at optimization due to the algorithm's ability to handle both equality and inequality constraints while remaining computationally cheap. Likewise, a Monte-Carlo Bayesian optimization could be similarly useful due to its ability to account for statistical uncertainties and for its ability to globally converge on a solution. Accounting for statistical uncertainty would also be of particular interest for future work due, as shown by the thrust optimization shown in section (i). The found optimal solution lies near some steep gradients, especially as M2 varies which may render this design solution unusable due to small perturbations in M2 causing large variations in thrust output.

Appendix

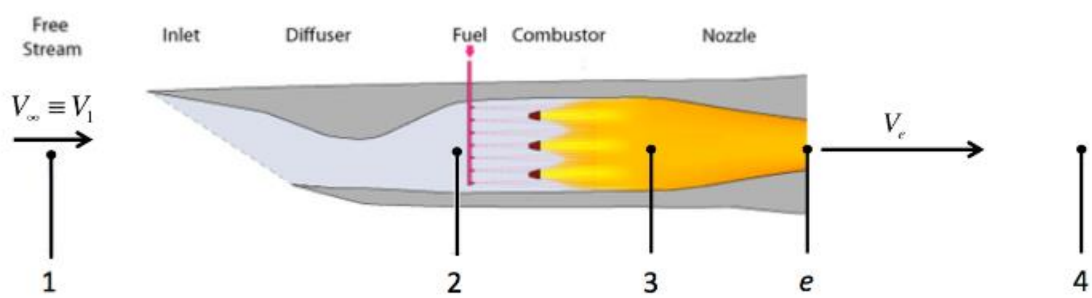
U.S. Standard Atmosphere Model

z [m]	T(z) [K]	p(z) [kPa]
0	288.15	101.325
500	284.9	95.46
1000	281.7	89.88
1500	278.4	84.56
2000	275.2	79.5
2500	271.9	74.69
3000	268.7	70.12
3500	265.4	65.78
4000	262.2	61.66
4500	258.9	57.75
5000	255.7	54.05
5500	252.4	50.54
6000	249.2	47.22
6500	245.9	44.08
7000	242.7	41.11
7500	239.5	38.3
8000	236.2	35.65
8500	233	33.15
9000	229.7	30.8
9500	226.5	28.58
10000	223.3	26.5
10500	220	24.54
11000	216.8	22.7
11500	216.7	20.98
12000	216.7	19.4
12500	216.7	17.93
13000	216.7	16.58
13500	216.7	15.33
14000	216.7	14.17
14500	216.7	13.1
15000	216.7	12.11
15500	216.7	11.2
16000	216.7	10.35
16500	216.7	9.572
17000	216.7	8.85
17500	216.7	8.182

18000	216.7	7.565
18500	216.7	6.995
19000	216.7	6.467
19500	216.7	5.98
20000	216.7	5.529
22000	218.6	4.047

Code

Non-Isentropic Ramjet Calculator



Module 7- Thermally Unchoked Check

```
clear
alt =4300; %flight alt-- m
eta_d = .92; %inlet eff.
eta_n = .94; %nozzle eff.
A_e = .015; %m^2
M1 = 2.4;
M2= .15;
Tt3_max = 2400; %k
qf = 42.3*10^6; %j/kg
[T, Tt, P, Pt, V, M, cp, deltaS, Thermal_efficiency, Propulsive_efficiency,
Overall_efficiency, Power,Thrust, Isp, TSFC, MassFlux, f, q23] = RamjetCalculator(alt,
eta_d, eta_n, A_e, M1, M2, Tt3_max, qf);
states = ['1', '2', '3', 'e', '4'];


%results
```

```
InputsTable = table(alt, eta_d, eta_n, A_e,M1, M2, Tt3_max,qf,
'VariableNames',{'Flight Altitude','Diffuser Eff.', 'Nozzle Eff.', 'A_e','M1',
'M2','Tt3_max', 'q_f' }  )
StatesTable =table(states',T', Tt', P', Pt', V', M', cp',
'VariableNames',{'States','T','Tt','P','Pt','V','M', 'Cp'})
PerformanceStatistics = table(Thermal_efficiency, Propulsive_efficiency,
Overall_efficiency, Power,Thrust, Isp, TSFC, f, q23, 'VariableNames',{'Thermal
efficiency', 'Propulsive efficiency', 'Overall efficiency', 'Power', 'Thrust', 'Isp',
'TSFC', 'f', 'q23'})
EntropyResults
=table(deltaS(1),deltaS(2),deltaS(3),deltaS(4),deltaS(5),deltaS(6),deltaS(7),
'VariableNames',{'deltaS_12', 'deltaS_23', 'deltaS_3e', 'deltaS_e4', 'deltaS_13',
'deltaS_1e', 'deltaS_14'})
MassFluxTable =table(MassFlux(1),MassFlux(2),MassFlux(3), 'VariableNames',{'Inlet Mass
Flux','Outlet Mass Flux','Fuel Mass Flux'})
```

Plotting T-S Diagram for Thermally Unchoked Check (a)

```
ds_23 = (deltaS(5)-deltaS(1))/10000;
s_23= deltaS(1):ds_23:deltaS(5);
T_23 = [];
T_23(1) = T(2);


for i = 1:length(s_23)-1
  T_23(i+1) = T_23(i) + (T_23(i)/(986 + .179*T_23(i)))*ds_23;
  if (T_23(i) >=T(3))
      break
  end
end
s_23 = linspace(deltaS(1), deltaS(5), length(T_23));


ds_41 = (deltaS(7))/10000;
s_41= 0:ds_41:deltaS(7);
T_41 = [];
T_41(1) = T(1);


for i = 1:length(s_41)-1
  T_41(i+1) = T_41(i) + (T_41(i)/1004)*ds_41;
    if (T_41(i) >=T(5))
      break
  end
end
```

```
s_41 = linspace(0, deltaS(7), length(T_41));


plot([0, deltaS(1)], [T(1), T(2)]) %1-2
hold on
plot(s_23,T_23) %2-3
plot([deltaS(5), deltaS(6)], [T(3), T(4)]) %3-e
plot([deltaS(6), deltaS(7)], [T(4), T(5)]) %e-4
plot(s_41,T_41) %4-1
legend({'1-2','2-3','3-e','e-4','4-1',}, 'Location','best')
xlabel('Relative Entropy(J/kg-k)')
ylabel('Temperature (k)')
title('T-s Diagram  for Thermally Unchoked case')
grid on
axis([-100 2500 0 2500])
hold off
```

Module 8- Thermally Choked Check

```
alt =4300; %flight alt-- m
eta_d = .92; %inlet eff.
eta_n = .94; %nozzle eff.
A_e = .015; %m^2
M1 = 2.4;
M2= .4;
Tt3_max = 2400; %k
qf = 42.3*10^6; %j/kg
[T, Tt, P, Pt, V, M, cp, deltaS, Thermal_efficiency, Propulsive_efficiency,
Overall_efficiency, Power,Thrust, Isp, TSFC, MassFlux, f, q23] = RamjetCalculator(alt,
eta_d, eta_n, A_e, M1, M2, Tt3_max, qf);
states = ['1', '2', '3', 'e', '4'];


%results
InputsTable = table(alt, eta_d, eta_n, A_e,M1, M2, Tt3_max,qf,
'VariableNames',{'Flight Altitude','Diffuser Eff.', 'Nozzle Eff.', 'A_e','M1',
'M2','Tt3_max', 'q_f' }  )
StatesTable =table(states',T', Tt', P', Pt', V', M', cp',
'VariableNames',{'States','T','Tt','P','Pt','V','M', 'Cp'})
PerformanceStatistics = table(Thermal_efficiency, Propulsive_efficiency,
Overall_efficiency, Power,Thrust, Isp, TSFC, f,q23, 'VariableNames',{'Thermal
efficiency', 'Propulsive efficiency', 'Overall efficiency', 'Power', 'Thrust', 'Isp',
'TSFC', 'f', 'q23'})
EntropyResults
=table(deltaS(1),deltaS(2),deltaS(3),deltaS(4),deltaS(5),deltaS(6),deltaS(7),
'VariableNames',{'deltaS_12', 'deltaS_23', 'deltaS_3e', 'deltaS_e4', 'deltaS_13',
'deltaS_1e', 'deltaS_14'})
```

```
MassFluxTable =table(MassFlux(1),MassFlux(2),MassFlux(3), 'VariableNames',{'Inlet Mass
Flux','Outlet Mass Flux','Fuel Mass Flux'})
```

Plotting T-S Diagram for Thermally Choked Check (a)

```
ds_23 = (deltaS(5)-deltaS(1))/10000;
s_23= deltaS(1):ds_23:deltaS(5);
T_23 = [];
T_23(1) = T(2);


for i = 1:length(s_23)-1
  T_23(i+1) = T_23(i) + (T_23(i)/(986 + .179*T_23(i)))*ds_23;
  if (T_23(i) >=T(3))
      break
  end
end
s_23 = linspace(deltaS(1), deltaS(5), length(T_23));


ds_41 = (deltaS(7))/10000;
s_41= 0:ds_41:deltaS(7);
T_41 = [];
T_41(1) = T(1);


for i = 1:length(s_41)-1
  T_41(i+1) = T_41(i) + (T_41(i)/1004)*ds_41;
    if (T_41(i) >=T(5))
      break
  end
end


s_41 = linspace(0, deltaS(7), length(T_41));


plot([0, deltaS(1)], [T(1), T(2)]) %1-2
hold on
plot(s_23,T_23) %2-3
plot([deltaS(5), deltaS(6)], [T(3), T(4)]) %3-e
plot([deltaS(6), deltaS(7)], [T(4), T(5)]) %e-4
plot(s_41,T_41) %4-1
legend({'1-2','2-3','3-e','e-4','4-1',}, 'Location','best')
xlabel('Relative Entropy(J/kg-k)')
ylabel('Temperature (k)')
title('T-s Diagram  for Thermally Unchoked case')
grid on
axis([-100 2500 0 2500])
```

```
hold off
```

(b) Validation of Atmospheric Model

```
clear
StdAtmModelData = importdata('Validation of Atmospheric Model.xlsx');
z = StdAtmModelData.data(13:54, 5);
T_stdAtm = StdAtmModelData.data(13:54, 6);
P_stdAtm = StdAtmModelData.data(13:54, 7);


T_isen= [];
P_isen = [];


for i = 1:length(z)
[T_isen(i), P_isen(i),~,~] = IsenAltModel(z(i));
end


%T plot
plot(T_isen, z)
hold on
plot(T_stdAtm,z)
title('Temperature Vs Altitude')
xlabel('Temperature (k)')
ylabel('Altitude (m)')
grid on
axis([0 300 0 z(end)])
legend({'Isentropic Model', 'Standard Atmosphere'}, 'Location','best')
hold off


%P plot
plot(P_isen/1000, z)
hold on
plot(P_stdAtm,z)
title('Pressure Vs Altitude')
xlabel('Press (kPa)')
ylabel('Altitude (m)')
grid on
legend({'Isentropic Model', 'Standard Atmosphere'}, 'Location','best')
hold off
```

(c) Flight mach number effects in unchoked case

```
clear
alt =4300; %flight alt-- m
eta_d = .92; %inlet eff.
```

```
eta_n = .94; %nozzle eff.
A_e = .015; %m^2
M2= .15;
Tt3_max = 2400; %k
qf = 42.3*10^6; %j/kg


M1 = 0.8:.1:5;


Overall_efficiency_array = [];
Thrust_array = [];
TSFC_array = [];


for i = 1:length(M1)
    [~, ~,~,~, ~, ~, ~, ~, ~, ~, Overall_efficiency_array(i), ~,Thrust_array(i), ~,
TSFC_array(i), ~, ~, ~] = RamjetCalculator(alt, eta_d, eta_n, A_e, M1(i), M2, Tt3_max,
qf);
end


plot(M1,Overall_efficiency_array)
title('\eta_o Vs M1 for Unchoked Case')
xlabel('M1')
ylabel('\eta_o')
grid on


plot(M1,Thrust_array)
title('Thrust Vs M1 for Unchoked Case')
xlabel('M1')
ylabel('Thrust (N)')
grid on


plot(M1,TSFC_array)
title('TSFC Vs M1 for Unchoked Case')
xlabel('M1')
ylabel('TSFC (kg/hr/N)')
grid on
```

(d) Altitude effects in unchoked case

```
clear


eta_d = .92; %inlet eff.
eta_n = .94; %nozzle eff.
```

```
A_e = .015; %m^2
M1 = 2.4;
M2= .15;
Tt3_max = 2400; %k
qf = 42.3*10^6; %j/kg


alt = 2000:1000:30000;


Overall_efficiency_array = [];
Thrust_array = [];
TSFC_array = [];


for i = 1:length(alt)
    [~, ~,~,~, ~, ~, ~, ~, ~, ~, Overall_efficiency_array(i), ~,Thrust_array(i), ~,
TSFC_array(i), ~, ~, ~] = RamjetCalculator(alt(i), eta_d, eta_n, A_e, M1, M2, Tt3_max,
qf);
end


plot(alt,Overall_efficiency_array)
title('\eta_o Vs Altitude for Unchoked Case')
xlabel('Altitude (z)')
ylabel('\eta_o')
grid on


plot(alt,Thrust_array)
title('Thrust Vs Altitude for Unchoked Case')
xlabel('Altitude (z)')
ylabel('Thrust (N)')
grid on


plot(alt,TSFC_array)
title('TSFC Vs Altitude for Unchoked Case')
xlabel('Altitude (z)')
ylabel('TSFC (kg/hr/N)')
grid on
```

(e) Optimizing efficiency and TSFC

```
Clear

eta_d = .92; %inlet eff.
eta_n = .94; %nozzle eff.
A_e = .015; %m^2
```

```
M2= .15;
Tt3_max = 2400; %k
qf = 42.3*10^6; %j/kg


alt = 2000:500:20000;
M1 = 0.8:.01:5;


Overall_efficiency_array = [];
TSFC_array = [];
OptimalM1_eta_o = [];
OptimalM1_TSFC = [];




for i = 1:length(alt)
    for j = 1:length(M1)
    [~, ~,~,~, ~, ~, ~, ~, ~, ~, Overall_efficiency_array(j), ~,~, ~, TSFC_array(j),
~, ~, ~] = RamjetCalculator(alt(i), eta_d, eta_n, A_e, M1(j), M2, Tt3_max, qf);
    end
    [~, indx] = max(Overall_efficiency_array);
    OptimalM1_eta_o(i) = M1(indx);


    [~, indx] = min(TSFC_array);
    OptimalM1_TSFC(i) = M1(indx);
end


plot(alt,OptimalM1_eta_o)
title('Altitude Vs Optimal M1 for Maximizing \eta_o for Unchoked Case')
xlabel('Altitude (z)')
ylabel('M1_{optimal}')
grid on


plot(alt,OptimalM1_TSFC)
title('Altitude Vs Optimal M1 for Minimizing TSFC for Unchoked Case')
xlabel('Altitude (z)')
ylabel('M1_{optimal}')
grid on
```

(f) Inlet/diffuser eff. effects in unchoked case

```
clear
alt =4300; %flight alt-- m
eta_n = .94; %nozzle eff.
```

```
A_e = .015; %m^2
M1 = 2.4;
M2= .15;
Tt3_max = 2400; %k
qf = 42.3*10^6; %j/kg


eta_d = .5:.05:1;


Overall_efficiency_array = [];
Thrust_array = [];
TSFC_array = [];


for i = 1:length(eta_d)
    [~, ~,~,~, ~, ~, ~, ~, ~, ~, Overall_efficiency_array(i), ~,Thrust_array(i), ~,
TSFC_array(i), ~, ~, ~] = RamjetCalculator(alt, eta_d(i), eta_n, A_e, M1, M2, Tt3_max,
qf);
end


plot(eta_d,Overall_efficiency_array)
title('\eta_o vs \eta_d for Unchoked Case')
xlabel('\eta_d')
ylabel('\eta_o')
grid on


plot(eta_d,Thrust_array)
title('Thrust vs \eta_d for Unchoked Case')
xlabel('\eta_d')
ylabel('Thrust (N)')
grid on


plot(eta_d,TSFC_array)
title('TSFC vs \eta_d for Unchoked Case')
xlabel('\eta_d')
ylabel('TSFC (kg/hr/N)')
grid on
```

(g) Nozzle eff. effects in unchoked case

```
clear
alt =4300; %flight alt-- m
eta_d = .92;
A_e = .015; %m^2
M1 = 2.4;
```

```
M2= .15;
Tt3_max = 2400; %k
qf = 42.3*10^6; %j/kg


eta_n = .5:.05:1;


Overall_efficiency_array = [];
Thrust_array = [];
TSFC_array = [];


for i = 1:length(eta_n)
    [~, ~,~,~, ~, ~, ~, ~, ~, ~, Overall_efficiency_array(i), ~,Thrust_array(i), ~, TSFC_array(i), ~, ~, ~] = RamjetCalculator(alt, eta_d, eta_n(i), A_e, M1, M2, Tt3_max, qf);
end


plot(eta_n,Overall_efficiency_array)
title('\eta_o vs \eta_n for Unchoked Case')
xlabel('\eta_n')
ylabel('\eta_o')
grid on


plot(eta_n,Thrust_array)
title('Thrust vs \eta_n for Unchoked Case')
xlabel('\eta_n')
ylabel('Thrust (N)')
grid on


plot(eta_n,TSFC_array)
title('TSFC vs \eta_n for Unchoked Case')
xlabel('\eta_n')
ylabel('TSFC (kg/hr/N)')
grid on
```

(h) M2 effects in unchoked case

```
clear
alt =4300; %flight alt-- m
eta_d = .92; %inlet eff.
eta_n = .94; %nozzle eff.
A_e = .015; %m^2
M1 = 2.4;
Tt3_max = 2400; %k
```

```
qf = 42.3*10^6; %j/kg


M2= .1:.1:2.5;


Overall_efficiency_array = [];
Thrust_array = [];
TSFC_array = [];


for i = 1:length(M2)
    [~, ~,~,~, ~, ~, ~, ~, ~, ~, Overall_efficiency_array(i), ~,Thrust_array(i), ~,
TSFC_array(i), ~, ~, ~] = RamjetCalculator(alt, eta_d, eta_n, A_e, M1, M2(i), Tt3_max,
qf);
end


plot(M2,Overall_efficiency_array)
title('\eta_o Vs M2 for Unchoked Case')
xlabel('M2')
ylabel('\eta_o')
grid on


plot(M2,Thrust_array)
title('Thrust Vs M2 for Unchoked Case')
xlabel('M2')
ylabel('Thrust (N)')
grid on


plot(M2,TSFC_array)
title('TSFC Vs M2 for Unchoked Case')
xlabel('M2')
ylabel('TSFC (kg/hr/N)')
grid on
```

(i) Design Case

```
clear
M1 = 5;
alt = 27400;
qf = 42.3*10^6; %j/kg
eta_d = .92; %inlet eff.
eta_n = .94; %nozzle eff.
A_e = .015;
```

```
InputsTable = table(alt, eta_d, eta_n, A_e,M1,qf, 'VariableNames',{'Flight
Altitude','Diffuser Eff.', 'Nozzle Eff.', 'A_e','M1', 'q_f' }  )


M2 = .1:.01:2.4;
Tt3 = 1500:10:2400;


[X,Y] = meshgrid(M2,Tt3);


for i = 1:length(Tt3)
    for j = 1:length(M2)
        [Overall_efficiency(i,j), Thrust(i,j)] = RamjetTt3DesignCalculator(alt, eta_d,
eta_n, A_e, M1, X(i,j), qf, Y(i,j));
        if(Overall_efficiency(i,j)<0)
        end
    end
end


[MaxEfficiency, tempIndx]= max(Overall_efficiency,[],'all');
OptimizedEffTable = table(MaxEfficiency,X(tempIndx), Y(tempIndx),
'VariableNames',{'Max Efficiency', 'Optimal M2', 'Optimal Tt3'} )
contourf(X(45:end, 1:50), Y(45:end, 1:50),Overall_efficiency(45:end,
1:50),30,'EdgeAlpha',0.1)
title('\eta_o Vs M2 and Tt3 for Unchoked Case')
h = colorbar;
xlabel('M2')
ylabel('Tt3')
h.Label.FontSize = 16;
h.Label.String = '\eta_o';
h.Label.Rotation = 270;
h.Label.VerticalAlignment = "bottom";
ax = gca;
chart = ax.Children(1);
datatip(chart,X(tempIndx),Y(tempIndx));




[MaxThrust, tempIndx]= max(Thrust,[],'all');
OptimizedThrustTable = table(MaxThrust,X(tempIndx), Y(tempIndx), 'VariableNames',{'Max
Thrust', 'Optimal M2', 'Optimal Tt3'} )


contourf(X, Y,Thrust,20, 'EdgeAlpha',0.1)
title('Thrust Vs M2 and Tt3 for Unchoked Case')
```

```
h = colorbar;
xlabel('M2')
ylabel('Tt3')
h.Label.FontSize = 16;
h.Label.String = 'Thrust (N)';
h.Label.Rotation = 270;
h.Label.VerticalAlignment = "bottom";
ax = gca;
chart = ax.Children(1);
datatip(chart,X(tempIndx),Y(tempIndx));
```

Non-Isentropic Ramjet Calculator

Input

- alt: in m
- eta_d
- eta_n
- A_e
- M1
- M2
- Tt3_max: in k
- qf: j/kg

Output

- T - Static Temperatures in form [T1, T2, T3, Te, T4]
- Tt - Total Temperatures in form [Tt1, Tt2, Tt3, Tte, Tt4]
- P - Static Pressure (kPa) in form [P1, P2, P3, Pe, P4]
- Pt - Total Pressure (kPa) in form [Pt1, Pt2, Pt3, Pte, Pt4]
- V- air speed in form [V1, V2, V3, Ve, V4]
- M- mach numbers in form [M1, M2, M3, Me, M4]
- Cp- in form [cp_1, cp_2, cp_3, cp_e, cp_4]
- deltaS = in form [deltaS_12, deltaS_23, deltaS_3e, deltaS_e4, deltaS_13, deltaS_1e, deltaS_14]
- Thermal_efficiency
- Propulsive_efficiency
- Overall_efficiency
- Power
- Thrust
- Isp
- TSFC (kg/hr/N)
- MassFlux- in form [mi,me, mf]

- f

```
function [T, Tt, P, Pt, V, M, cp, deltaS, Thermal_efficiency, Propulsive_efficiency,
Overall_efficiency, Power,Thrust, Isp, TSFC, MassFlux, f, q23] = RamjetCalculator(alt,
eta_d, eta_n, A_e, M1, M2, Tt3_max, qf)
```

Constants

```
R = 286.9;
gamma_12= 1.4;
gamma_23= 1.3;
gamma_3e= 1.3;
gamma_e4= 1.3;
```

State Calculations

Returns T, Tt, P, Pt, M ,v

State 1 (module 1)

```
[T1,P1,~,~]=IsenAltModel(alt);
a1= sqrt(gamma_12*R*T1);
Pt1=TotalToStaticPressureRel([],P1,M1,gamma_12);
Tt1=TotalToStaticTemperatureRel([],T1,M1,gamma_12);
V1 = M1*a1;
cp_1 = gamma_12*R / (gamma_12-1);
```

State 2 (module 2)

```
Tt2 = Tt1; %no w or q
T2 = TotalToStaticTemperatureRel(Tt2,[],M2,gamma_12);


Pt2 = P1*(1+eta_d*((gamma_12-1.)/2).*M1.^2).^(gamma_12./(gamma_12-1));
P2 = TotalToStaticPressureRel(Pt2,[],M2,gamma_12);


a2 = sqrt(gamma_12*T2*R);
V2= a2*M2;


cp_2= cp_1;
deltaS_12 = cp_2*log(Tt2/Tt1) - R*log(Pt2/Pt1);
```

State 3 (module 3)

Check if we are thermally choked:

```
Tt3_choked = Tt2*( (1/(2*(gamma_23+1))) * (1/M2^2) * (1+gamma_23*M2^2)^2 *
(1+((gamma_23-1)./2).*M2.^2)^-1);
```

```
if(Tt3_choked<Tt3_max)
    Tt3 = Tt3_choked;
    M3 = 1;
    %disp('Combustor Thermally Choked')
else
    Tt3=Tt3_max;
    fun = @(M3)M3Solver(M3, Tt3, Tt2,M2, gamma_23);
    M3_0 = .5;
    options = optimoptions('fsolve', 'MaxFunctionEvaluations',5000,
'MaxIterations',5000, Display='none');
    M3 = fsolve(fun,M3_0,options);
    %disp('Combustor is not Thermally Choked: Tt3 Materially Limited')
end
```

```
q23 = 986*(Tt3-Tt2)+0.5*0.179*(Tt3^2-Tt2^2);
P3 = P2; %const. p heat addition
T3 =  TotalToStaticTemperatureRel(Tt3,[],M3,gamma_23);
Pt3 =  TotalToStaticPressureRel([],P3,M3,gamma_23);
a3 = sqrt(gamma_23*T3*R);
V3 = a3*M3;
cp_3 = 986+0.179*T3;
deltaS_23 = cp_3*log(Tt3/Tt2) - R*log(Pt3/Pt2);


deltaS_13 = deltaS_23+ deltaS_12;
```

Nozzle (module 4)

Is the Nozzle Choked?

```
Test_Mach_num = sqrt((2/(gamma_3e-1)) * ((eta_n*(1-(P1/Pt3)^((gamma_3e-
1)/gamma_3e)))/(1-eta_n*(1-(P1/Pt3)^((gamma_3e-1)/gamma_3e)))));


if (Test_Mach_num<1)
    %disp('Nozzle is not Choked')
    Me = Test_Mach_num;
    Pe =P1;


else
    %disp('Nozzle is Choked')
    Me = 1;
    Pe = Pt3*(1-(1/eta_n)*((gamma_3e-1)/(gamma_3e+1)))^(gamma_3e/(gamma_3e-1));
end
```

Nozzle Exit

```
Tte = Tt3;


Pte = TotalToStaticPressureRel([],Pe,Me,gamma_3e);
Te =  TotalToStaticTemperatureRel(Tte,[],Me,gamma_3e);


ae = sqrt(gamma_3e*Te*R);
Ve = ae*Me;


cp_e = 986+0.179*Te;
deltaS_3e = cp_e*log(Tte/Tt3) - R*log(Pte/Pt3);
deltaS_1e = deltaS_23+ deltaS_12+deltaS_3e;


mass_flux_exit = (Pe/(R*Te))*Ve*A_e;
```

State 4 (module 5)

```
if (Test_Mach_num<1)
    eta_n_ext = 1;
else
    eta_n_ext = Test_Mach_num^-.3;
end


P4 = P1;
Tt4= Tte;


T4 = Tt4*(1-eta_n_ext*(1-(P1/Pte)^((gamma_e4-1)/gamma_e4)));


M4 = TotalToStaticTemperatureRel(Tt4,T4,[],gamma_e4);
Pt4 = TotalToStaticPressureRel([],P4,M4,gamma_e4);


a4 = sqrt(gamma_e4*T4*R);
V4 = a4*M4;


cp_4 = 986+0.179*T4;
deltaS_e4 = cp_4*log(Tt4/Tte) - R*log(Pt4/Pte);
deltaS_14 = deltaS_23+ deltaS_12+deltaS_3e+deltaS_e4;
```

Engine Performance (module 6)

Returns Thrust, prop. power, eta_th, eta_p, overall eta, fuel mass flux, TSFC, Isp

```
mass_flux_inlet = mass_flux_exit/(1+(q23/qf));
mass_flux_fuel = mass_flux_inlet*(q23/qf);
f = mass_flux_fuel/mass_flux_inlet;
Thrust = mass_flux_inlet*((1+f)*Ve-V1) + (Pe-P1)*A_e;
TSFC = 3600*mass_flux_fuel/Thrust;
Isp = Thrust/(9.8*mass_flux_fuel);
Veq = Ve + (Pe-P1)*(A_e/mass_flux_exit);
Propulsive_efficiency = 2/(1+Veq/V1);
Thermal_efficiency = ((mass_flux_exit*.5*Veq^2)-
(mass_flux_inlet*.5*V1^2))/(mass_flux_inlet*q23);
Overall_efficiency = Propulsive_efficiency*Thermal_efficiency;
Power = Thrust*V1;


T= [T1, T2, T3, Te, T4];
Tt = [Tt1, Tt2, Tt3, Tte, Tt4];
P= [P1, P2, P3, Pe, P4]/1000;
Pt=[Pt1, Pt2, Pt3, Pte, Pt4]/1000;
V=[V1, V2, V3, Ve, V4];
M= [M1, M2, M3, Me, M4];
cp=[cp_1, cp_2, cp_3, cp_e, cp_4];
deltaS= [deltaS_12, deltaS_23, deltaS_3e, deltaS_e4, deltaS_13, deltaS_1e, deltaS_14];
MassFlux = [mass_flux_inlet, mass_flux_exit, mass_flux_fuel];
end
```

```
function F = M3Solver(M3, Tt3, Tt2,M2, gamma_23)
F = ((M3/M2)^2 * ((1+gamma_23*M2^2)/(1+gamma_23*M3^2))^2 * ((1+((gamma_23-
1)./2).*M3.^2)/(1+((gamma_23-1)./2).*M2.^2))) - Tt3/Tt2;
end
```

Non-Isentropic Ramjet Tt3 Design Calculator

Input

- alt: in m
- eta_d
- eta_n
- A_e
- M1
- M2
- Tt3
- qf: j/kg

Output

- Overall_efficiency
- Thrust

```
function [Overall_efficiency, Thrust] = RamjetTt3DesignCalculator(alt, eta_d, eta_n,
A_e, M1, M2, qf, Tt3)
```

Constants

```
R = 286.9;
gamma_12= 1.4;
gamma_23= 1.3;
gamma_3e= 1.3;
```

State Calculations

State 1

```
[T1,P1,~,~]=IsenAltModel(alt);
a1= sqrt(gamma_12*R*T1);
Tt1=TotalToStaticTemperatureRel([],T1,M1,gamma_12);
V1 = M1*a1;
```

State 2

```
Tt2 = Tt1; %no w or q


Pt2 = P1*(1+eta_d*((gamma_12-1.)/2).*M1.^2).^(gamma_12./(gamma_12-1));
P2 = TotalToStaticPressureRel(Pt2,[],M2,gamma_12);
```

State 3

```
Tt3_choked = Tt2*( (1/(2*(gamma_23+1))) * (1/M2^2) * (1+gamma_23*M2^2)^2 *
(1+((gamma_23-1)./2).*M2.^2)^-1);


if(Tt3_choked<Tt3)
    Tt3 = Tt3_choked;
    M3 = 1;
    %disp('Combustor Thermally Choked')
else
    fun = @(M3)M3Solver(M3, Tt3, Tt2,M2, gamma_23);
    M3_0 = .5;
    options = optimoptions('fsolve', 'MaxFunctionEvaluations',5000,
'MaxIterations',5000, Display='none');
    M3 = fsolve(fun,M3_0,options);
    %disp('Combustor is not Thermally Choked: Tt3 Materially Limited')
```

```
end


q23 = 986*(Tt3-Tt1)+0.5*0.179*(Tt3^2-Tt1^2);
P3 = P2; %const. p heat addition
Pt3 =  TotalToStaticPressureRel([],P3,M3,gamma_23);
```

Nozzle

Is the Nozzle Choked?

```
Test_Mach_num = sqrt((2/(gamma_3e-1)) * ((eta_n*(1-(P1/Pt3)^((gamma_3e-
1)/gamma_3e)))/(1-eta_n*(1-(P1/Pt3)^((gamma_3e-1)/gamma_3e)))));


if (Test_Mach_num<1)
    %disp('Nozzle is not Choked')
    Me = Test_Mach_num;
    Pe =P1;
else
    %disp('Nozzle is Choked')
    Me = 1;
    Pe = Pt3*(1-(1/eta_n)*((gamma_3e-1)/(gamma_3e+1)))^(gamma_3e/(gamma_3e-1));
end
```

Nozzle Exit

```
Tte = Tt3;
Te =  TotalToStaticTemperatureRel(Tte,[],Me,gamma_3e);
ae = sqrt(gamma_3e*Te*R);
Ve = ae*Me;
mass_flux_exit = (Pe/(R*Te))*Ve*A_e;
```

Engine Performance

```
mass_flux_inlet = mass_flux_exit/(1+(q23/qf));
mass_flux_fuel = mass_flux_exit-mass_flux_inlet;
f = mass_flux_fuel/mass_flux_inlet;
Thrust = mass_flux_inlet*((1+f)*Ve-V1) + (Pe-P1)*A_e;


Veq = Ve + (Pe-P1)*(A_e/mass_flux_exit);
Propulsive_efficiency = 2/(1+Veq/V1);
Thermal_efficiency = ((mass_flux_exit*.5*Veq^2)-
(mass_flux_inlet*.5*V1^2))/(mass_flux_inlet*q23);
Overall_efficiency = Propulsive_efficiency*Thermal_efficiency;
end


function F = M3Solver(M3, Tt3, Tt2,M2, gamma_23)
```

```
F = ((M3/M2)^2 * ((1+gamma_23*M2^2)/(1+gamma_23*M3^2))^2 * ((1+((gamma_23-
1)./2).*M3.^2)/(1+((gamma_23-1)./2).*M2.^2))) - Tt3/Tt2;
end
```

Isentropic Atmosphere Model

Inputs:

- Altitude (in meters)

Output:

- static Temperature (k)
- Static Pressure (pa)
- Air Density (kg/m^3)
- Speed of Sound (m/s)

```
function [T_static,P_static,a,rho] = IsenAltModel(alt)


    if(alt>7958)
        T_static = 210;
        P_static = (33.6.*exp(-(alt-7958)./6605)).*1000;
        a= nan;
        rho= nan;

    else
        T_s = 288;
        P_s = 101300;
        rho_s = 1.23;
        z_star = 8404;

        gamma =1.4;
        SpecificGasConst_air = 286.9;

        T_static = T_s.*(1-((gamma-1)./gamma).*(alt./z_star));
        P_static = P_s.*(1-((gamma-1)./gamma)*(alt./z_star)).^(gamma./(gamma-1));
        rho = rho_s.*(1-((gamma-1)./gamma).*(alt/z_star)).^(1./(gamma-1));
        a = sqrt(T_static.*gamma.*SpecificGasConst_air);
    end
end
```

Total to Static Temperature Relations

This function will solve for a specified unknown given the other 2 knowns and gamma

Inputs:

- T_total
- T_static
- M

The above statics or totals or M can be left as [] and the funtion will know to return the blanks

- Gamma

Outputs:

whichever input is left as []

```
function SpecifiedUnknown = TotalToStaticTemperatureRel(T_total, T_static, M, gamma)
    if(isempty(T_total) && isempty(T_static))
        SpecifiedUnknown = (1+((gamma-1)./2).*M.^2);
    elseif(isempty(T_total))
        SpecifiedUnknown = T_static.*(1+((gamma-1)./2).*M.^2);
    elseif (isempty(T_static))
        SpecifiedUnknown = T_total./(1+((gamma-1)./2).*M^2);
    elseif(isempty(M))
        SpecifiedUnknown = sqrt((2.*(T_total./T_static-1))./(gamma-1));
    else
        error('Please leave an input blank')
    end
end
```

Total to Static Pressure Relations

This function will solve for a specified unknown given the other 2 knowns and gamma

Inputs:

- P_total
- P_static
- M

The above statics or totals or M can be left as [] and the funtion will know to return the blanks

- Gamma

Outputs:

whichever input is left as []

```
function SpecifiedUnknown = TotalToStaticPressureRel(P_total, P_static, M, gamma)
```

```
    if(isempty(P_total) && isempty(P_static))
        SpecifiedUnknown = (1+((gamma-1.)/2).*M.^2).^(gamma./(gamma-1));
    elseif(isempty(P_total))
        SpecifiedUnknown = P_static.*(1+((gamma-1.)/2).*M.^2).^(gamma./(gamma-1));
    elseif (isempty(P_static))
        SpecifiedUnknown = P_total./((1+((gamma-1)./2).*M.^2).^(gamma./(gamma-1)));
    elseif(isempty(M))
        SpecifiedUnknown = sqrt((2.*((P_total./P_static).^((gamma-1)./gamma)-
1))./(gamma-1));
    else
        error('Please leave an input blank')
    end
end
```