

Control System Requirements for a Nuclear Thermal Propulsion System

A. Hasse¹, M. Smith¹, B. Pershke¹, S. Gilliam¹, D. Adams¹, L. Qualls²

¹Dept. of Nuclear Engineering, University of Tennessee, Knoxville

²Reactor and Nuclear Systems Division, Oak Ridge National Laboratory

27 April 2016

NE 472

Abstract

NASA has an interest in manned missions to Mars. However, the safest and most efficient means for getting humans to the Martian surface may require more than traditional chemical rocketry. Chemical rockets, while effective at exiting Earth's atmosphere, still require a one-way transit time to Mars of approximately 8-12 months [1]. During this transit, astronauts are exposed to high doses of radiation from galactic cosmic rays (GCR) and solar energetic particles (SEP). Nuclear thermal propulsion (NTP) is an attractive alternative to traditional chemical rockets as they provide a higher specific impulse and transit velocity, which can shorten the transit time, and thus exposure time, by nearly half. An associated reduction in propellant mass allows for a higher payload per mission. Because of these benefits, the use of NTP systems could greatly benefit the future of manned interplanetary missions both in terms of safety and cost.

For this project, a dynamic simulation model was developed to investigate coolant/propellant pump, control drum, flow performance, and core responses necessary for a successful startup of a moderated NTP system. The system parameters are based on the Small Nuclear Reactor Engine (SNRE) design developed at Los Alamos National Laboratory. Simulink was used as the modeling software to solve a system of coupled differential equations that describe the hydrogen flow (coolant and propellant), control drum actuation, and the core power temperature and response. Multiple Matlab scripts were constructed to interface and control the simulation model, which allowed accelerated iterative testing over a wide range. The results of these simulations were put into time dependent profiles for temperature, temperature differentials, core reactivity and system pressure. Reduced data sets from the collections of these time dependent profiles allowed for the creation of response functions for each tested. Through the analysis and characterization of these response functions (such as the behavior of maximum temperatures, temperature changes, power surges, reactivity insertions, etc.), specific system parameter limits were outlined to ensure the predictability, safety and reliability of the system. Results of these simulations and the resulting control system requirements for a computer model based SNRE design are presented. These initial system requirements serve to inform the next, more detailed, phase of NTP system designs.

Group Information



From left to right:

Adam Hasse | ahasse@vols.utk.edu

Drew Adams | aadams53@vols.utk.edu

Nathan Gilliam | zln179@vols.utk.edu

Mike Smith | msmit312@vols.utk.edu

Bradley Pershke | bpershke@vols.utk.edu

Mentor:

Lou Qualls | quallsal@ornl.gov

Introduction

NASA has an interest in a manned mission to Mars. NTP systems may provide a more effective solution for this initiative than present chemical rocket technology can offer. Most modern rockets follow the simple schematic in *Figure 1*. This system uses a fuel-oxidization system to create combustion, and the heat of this combustion process accelerates the gases. This reaction creates exhaust that is expelled out of the back of a converging-diverging nozzle. Chemical propulsion systems are useful for getting a payload off of the Earth's surface and into orbit, but are inefficient in regards to fuel consumption. The amount of a rocket's useable propellant divided by the rocket's gross mass is a common metric for fuel efficiency. This metric is referred to as the propellant mass fraction, or PMF. For example, the Ares V Earth Departure Stage has a PMF of Approximately 88% leaving only 12% of the total rocket mass for mission payload [2].

The reason for this inefficiency comes from a chemical rocket's inherently low specific impulse. Specific impulse is a measure of the efficiency of a propellant system. It is defined by *Equation 1*, where v_e is the effective exhaust velocity (m/s), g_0 is the acceleration due to gravity on Earth (m/s^2), k is Boltzmann's constant (J/K), T is temperature (K), and m is the molecular mass (kg). High specific impulses are obtained by decreasing the molecular weight of the exhaust and/or increasing the temperature of the exhausting gas. *Equation 2* shows how the effective exhaust velocity (v_e in m/s) is related to thrust created (F_{th} in N), with $\dot{m}$ being the mass flow rate of the gas through the nozzle (kg/s).

$$I_{sp} = \frac{v_e}{g_0} = \frac{1}{g_0} \sqrt{\frac{3kT}{m}}$$

Equation 1

$$F_{th} = v_e \cdot \dot{m}$$

Equation 2

A typical chemical rocket has a specific impulse on the order of 175-400 seconds [3], which is directly related to the propellant's molecular weight of 18 grams per mole. For a nuclear thermal rocket, the exhaust molecule weighs only 2 grams per mole. This allows increases in the specific impulse of the system to values of approximately 900 seconds, which is more than double the specific impulse of traditional chemical rockets.

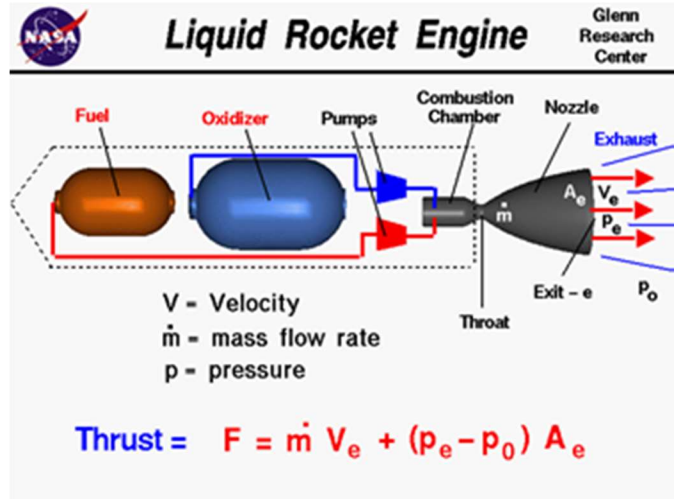


Figure 1: Schematic of chemical liquid rocket engine. Image courtesy of NASA, Glenn Research Center. www.grc.nasa.gov/www/

Furthermore, the expected PMF of NTP systems is approximately 30% which allows for higher payloads per mission and could potentially reduce the number of launches required per mission. These potential savings could potentially reach billions of dollars per mission, thus allowing NTP systems to serve as a more economical and improved propulsion alternative for future manned missions to Mars [4].

History of Nuclear Thermal Propulsion

The idea for using nuclear energy to power a spacecraft was thought of in 1942 [5], soon after Enrico Fermi developed the first fission reactor. In 1944, a joint collaboration of scientists from the University of Chicago and Los Alamos National Laboratory began discussing the finer details of NTP systems, thus leading to the creation of Project Rover in 1955. Project Rover consisted of three main phases (Kiwi, Phoebus, and Pewee), all of which were meant to test the feasibility of a solid-core NTP system. Despite numerous advances in NTP technology this project was cancelled in 1972.

From this project came the next major step in NTP systems known as NERVA, or Nuclear Engine for Rocket Vehicle Application. Beginning in 1961, this project used the basic NTP system outlined in Project Rover and adapted it for space application. NERVA was meant to send a team to Mars, but this never actually happened. However, the NERVA program did develop NTP systems that met all of their specifications needed to send a nuclear rocket into space. This project was cancelled in 1972 with the cessation of Project Rover, but remains the backbone of modern NTP systems [6].

The last design studied during Project Rover was the Small Nuclear Rocket Engine (SNRE), which remains one of the most well-documented designs for NTP systems. The maturity and simplicity of this design allow for the neutronics of the core to be easily modeled, along with providing a benchmark in the lower end of the necessary thrust specifications. A variety of growth options are also available in the literature, thus allowing the SNRE design to be implemented with a higher thrust capacity. Along with these previous design parameters are many simulated and anticipated performance parameters which can serve as reference values for comparative purposes [7].

Project Design Selection

Because of the abundance of past research material on the subject, the SNRE design was chosen as the basis of this research project. The SNRE system is a combination of thermal fluid dynamics and nuclear physics. Starting with the core, the basic construction consists of a sub-critical reactor consisting of hexagonally shaped fuel elements made from a ceramic compound of Uranium-Graphite with Zirconium cladding (U,Zr)C [8]. The core consists of 564, 89 cm long hexagonal fuel elements with 19 Zirconium clad cooling channels (*Figure 2*). The 241 dual purpose tie-tubes serve as both structural support for the core and hydrogen conduit for pre-heating

in the system. Both the fuel elements and the tie-tubes are geometrically arranged throughout the core in a similar pattern to *Figure 2*.

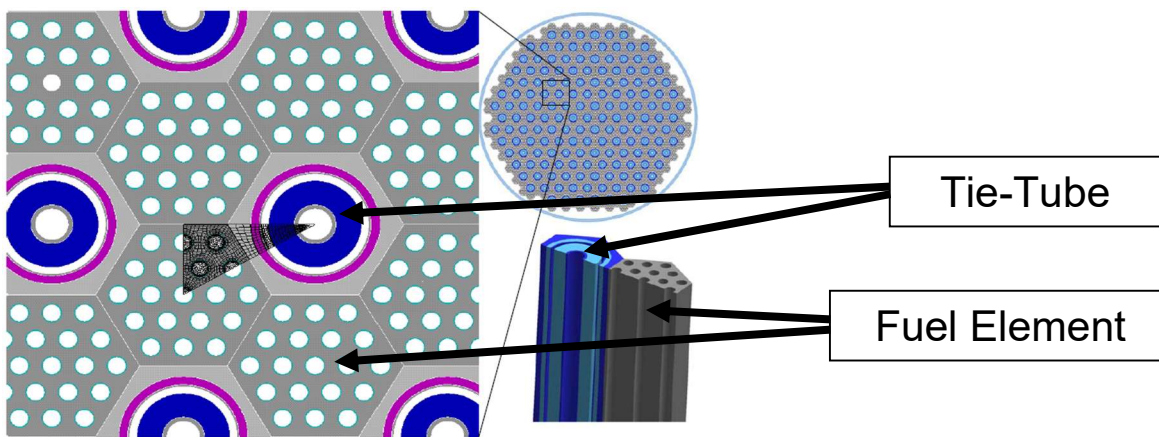


Figure 2: Cross-sectional view of core components and structure.[8]

The fuel enrichment, composition and geometry are designed such that the core is naturally at a sub-critical state. 12 control drums are distributed evenly around the core and may be actuated by either pneumatic or mechanical means (*Figure 3*). Approximately 120 degrees of the face of each drum is plated with a neutron absorbent material while the remaining 240 degrees are a neutron reflective material (beryllium) [8]. To initiate a critical core, the drums are rotated such that the reflective portion of each drum is facing the center of the core. This configuration increases the reactivity of the core by lessening the neutron leakage factor and begins to bring the core critical. The drums are designed to be positioned at 90 degrees at full reactor power and temperature.

Hydrogen serves as both the system coolant and engine propellant for the SNRE system. It is stored at approximately 20 K in liquid form in a pressurized, cryogenic ullage reservoir. A pump is used to pull liquid hydrogen from the reservoir and distribute it through the system's cooling passages. When the rocket is operational and the core is critical, the hydrogen first passes over the exit nozzle where it undergoes a phase change to a gas while cooling the nozzle and preventing it from melting. The coolant is then fed up through the reflector drums towards the top of the core. After accumulating in an upper tie-tube plenum, the hydrogen is passed down through the core through the inner channel of a tie-tube. The conduit passes the hydrogen to a lower tie-tube plenum where the gas is passed back up through the outer layer of the tie-tube. The purpose of this two-fold pass through the core is to preheat the hydrogen for insertion directly into the fuel elements and to heat hydrogen to a point that can drive turbomachinery located at the top of the core. As the hydrogen passes out

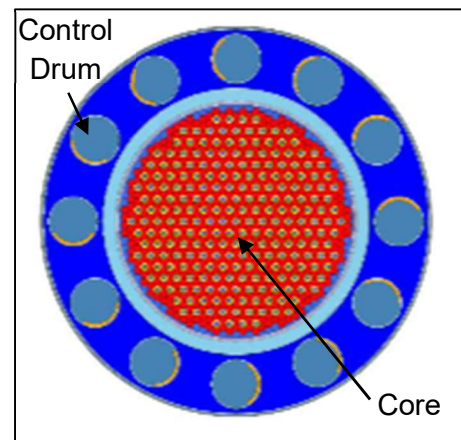


Figure 3: Control drum configuration around reactor core. [8]

of the top of the tie-tube it is used to drive turbomachinery that drives the pump that initiates the entire coolant flow process. After use in the turbomachinery, the hydrogen is passed to the fuel element cooling channels where it is super-heated to approximately 2900 K, accelerated and fed to the rocket nozzle where it is ultimately used as propulsion [9]. A diagram of the system as a whole is shown in *Figure 4*, and some relevant specific parameter values regarding the system are shown below in *Table 1*.

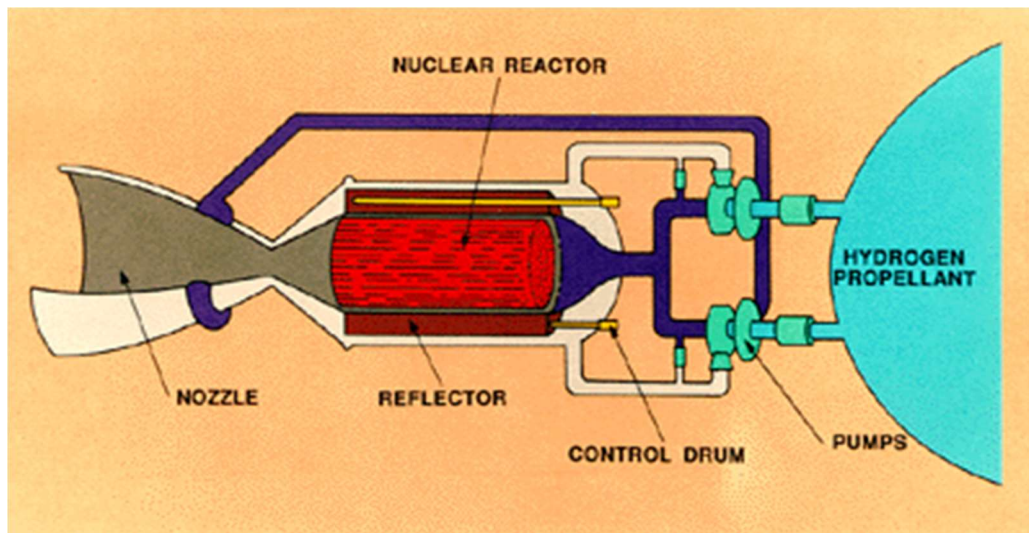


Figure 4: Basic diagram of NTP, core, nozzle and hydrogen flow. Courtesy of Georgia Tech Dept. of Electrical and Computer Engineering.

<http://www.propagation.gatech.edu/ECE6390/project/Fall2010/Projects/group3/spaceBusters/propulsion/propulsionRocket.html>

Table 1: Design Specifications for SNRE

<i>Engine System</i>		<i>Reactor System</i>	
Thrust (klbf)	16.4	Active Fuel Length (cm)	89.0
Chamber Inlet Temp (K)	2695	Effective Core Radius (cm)	29.5
Chamber Pressure (psia)	450	Engine Radius (cm)	49.3
Nozzle Expansion Ratio	100:1	Number of Fuel Elements	564
Specific Impulse (s)	875	Number of Tie Tube Elements	241
Engine Thrust-to-Weight	2.92	Max Enrichment (wt% U-235)	93
		Max Fuel Temp (K)	2860
		Margin to Fuel Melt (K)	40

*All data courtesy of Schnitzler, Borowski, and Fittje

Challenges

Although the NTP technologies previously discussed were presented in an overly simplified way, the inherent complexities and technical challenges therein are not trivial. Hydrogen is a coolant a neutron moderator and increases reactivity when put into the core. As the control drums rotate into their critical position, the reactor begins to heat up. This calls for flow

through the coolant and propulsion channels to increase to compensate for the increase in power. The increase in hydrogen pressure in the core inserts additional positive reactivity which causes another increase in core power. This is an inherent stability problem with the system, and reactivity control issues must be addressed in order to complete a safe startup. For the purposes of this project, it is assumed that the constant hydrogen flow can be controlled independently from reactor power so that the problem can be studied in a controlled manner. A later activity would include the coupling of the reactor power to pump power and thus to hydrogen flow and pressure.

Besides the positive reactivity problem, there are a variety of extreme conditions to take into consideration when choosing materials. These include the high variations in temperatures, from 10's of Kelvin to almost ~3000 Kelvin, the high hydrogen concentration, which leads to material degradation, and exposure to large amounts of radiation, which can contribute to material defects and embrittlement. Although many of these problems are necessary to consider for a full-scale design, many generalizations and assumptions were made for the purpose of this project.

The last challenge with NTP systems is that of the control system. Power changes in the core can happen on the order of picoseconds, while certain control systems, like the control drums, respond on the order of seconds. This response lag highlights the necessity of previous knowledge of system dynamics. The control system must be designed and used to prevent core damage. The simulation model allows repeated simulation, which allows designers to anticipate and proactively maintain safe operation.

Purpose

The purpose of this project was to develop a simulation model that, using SNRE baseline values, was able to successfully recreate startup scenarios of the NTP system. With these simulations developed in Matlab and Simulink, a variety of scenarios were tested to outline a set of control systems parameters necessary for the successful operation of the simulated NTP system. The results achieved came in the form of reduced data sets, response functions, and a variety of temperature, reactivity, and flow rate data sets. The results were compared to reference SNRE baseline results to determine the proper functioning of the simulation model. Further testing was performed to determine the specific parameters essential to the predictable, safe and reliable startup of the system.

Methodology

To assess the performance of the NTP design, a computer simulation was chosen as the most practical approach to outlining control system requirements needed for successful startup. This process began by a literature survey of multiple available documents outlining prior research and development of NTP systems. The SNRE design, developed at Los Alamos National

Laboratory (LANL), was chosen as the model system as there was ample parametric data available for the design phase and sufficient performance parameters available for simulation comparisons.

Although the SNRE system was well documented for design and performance parameters, various assumptions were made when translating the original SNRE theory to the analytical models of this project. This was done as a means to create a representative, yet introductory, simulation model and involved sacrificing system fidelity to allot for projected time constraints to complete the project. While a large part of this analysis involved monitoring various performance parameters like temperature stresses on materials, the methods used in this project only serve as a first-order approximation for failure analysis. For example, the values calculated for the maximum allowable rate of temperature change, or thermal shock, for the fuel and tie tubes was determined quantitatively, while if this were a funded R&D project slated for actual mission planning, this particular value would be determined by modeling in CAD and subjecting the model to intensive thermal analysis along with prototyping and field testing to determine more accurate failure parameters [10]. With an understanding of the depth of realistic representation involved in this model, the system layout will be discussed, and relative assumptions will be stated. Some of these assumptions are listed following:

1. Hydrogen flow is uncoupled from system control and maintained at a constant rate
2. No radiative heat-loss to space considered
3. No conductive heat-loss of hydrogen to other components in flow loop
4. All radial conductive heat-transfer between the fuel and the tie tubes is ignored

System Design

The physics of the system model can be compartmentalized into three main subsystems: heat transfer, fluid dynamics, and point kinetics. Inside each of these subsystems, various physics “blocks” representing individual processes controlling reactor physics, heat transfer and fluid dynamics were developed. These processes were individually researched and equations describing each process was determined. The equations were used to make initial hand calculations to confirm the general accuracy of the relationship between the proposed equation and the physics process at hand. If an equation was deemed appropriate for the process it was written into the Simulink model and incorporated into the greater system. These equations will be discussed in detail in the following sections.

Point Kinetics

The values for the variables in the point kinetics equations were based off literature referencing a similar sized reactor core with similar fuel structure [11]. The constant values for the mean neutron lifetime (Λ), delayed neutron fraction (β_i) and individual decay constants (λ_i) are outlined below in *Table 2*.

Table 2: SNRE Point Kinetics Values

<i>Variable</i>	<i>Symbol</i>	<i>Value</i>	<i>Units</i>
Delayed Neutron Fraction	β_i	0.065	N/a
Decay Constant	λ_i	0.077	sec^{-1}
Mean Neutron Lifetime	Λ	0.001	sec

*All data courtesy of M.M. El-Wakil [11]

These values were essential in creating our point kinetics model, shown following in *Figure 5*.

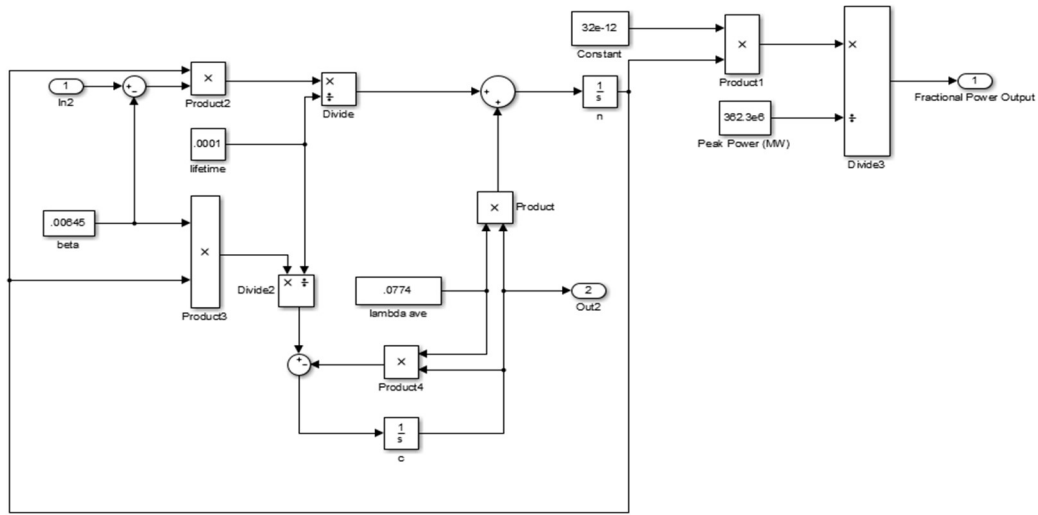


Figure 5: Point Kinetics model in Simulink

This model is derived from the governing point kinetics equation, shown in *Equation 3-4*. For these equations, $\frac{\partial n(t)}{\partial t}$ is the rate of change of neutron density in neutrons/cm³/sec, ρ is the reactivity of the system in dollars, β is the total delayed neutron fraction, Λ is the prompt neutron lifetime in seconds, $n(t)$ is the neutron density in neutrons/cm³, λ_i is the i -th generation radioactive decay constant in seconds⁻¹, and $C_i(t)$ is the average concentration of the i -th generation delayed neutron precursor in #/cm³. By solving the neutron density in the core at any step in time, the energy released may be estimated by assuming each neutron causes a single fission event equal to 200 MeV. The thermal power output is computed by simply converting MeV to Joules at each time step thus providing energy per unit time, or power output. This output is then fractionalized by the SNRE baseline design peak power value (362 MW) and passed to the heat transfer equations in the core and tie-tubes. From the SNRE Monte Carlo analysis of energy deposition to hydrogen in the core it was found that 85% of the total power was deposited in the fuel element coolant channels while the remaining 15% was deposited in the hydrogen in the tie-tubes.

$$\frac{\partial n(t)}{\partial t} = \frac{(\rho - \beta)}{\Lambda} n(t) + \sum_{i=1}^6 \lambda_i C_i(t)$$

Equation 3

$$\frac{\partial}{\partial t} C_i = \lambda_i C_i + \frac{\beta_i}{\Lambda}$$

Equation 4

Heat Transfer

Figure 6 shows how the power generated from point kinetics equations is imparted to the heat transfer model that was used in multiple parts of the simulation, specifically within the core and tie tubes. An iterative approach with six steps for each conductive heat transfer length was used.

For example, the volumetric power deposition to hydrogen in a single coolant channel in a fuel element is found by assuming 85% of total power is deposited into all fuel elements divided by the number of fuel elements in the core (546) and the number of coolant channels per fuel element (19). This solution could be performed over the entire length of each heat transfer section but greater fidelity and accuracy of final bulk temperatures was obtained by the iterative solution process throughout the total length of the heat transfer conduit. Therefore, by dividing the single coolant channel power deposition by the number of iterative step lengths (6) for a single coolant channel in the fuel, the volumetric power deposition was ascertained for the convective heat transfer model. With that said, even greater fidelity could be obtained by dividing each heat transfer length into more divisions, but six divisions were decided to be sufficient for the approximations in this model. This same logic is applied to the tie-tubes except with 15% of total power deposited into all tie-tubes divided by the number of tie tubes (241) and the number of iterative step lengths (6).

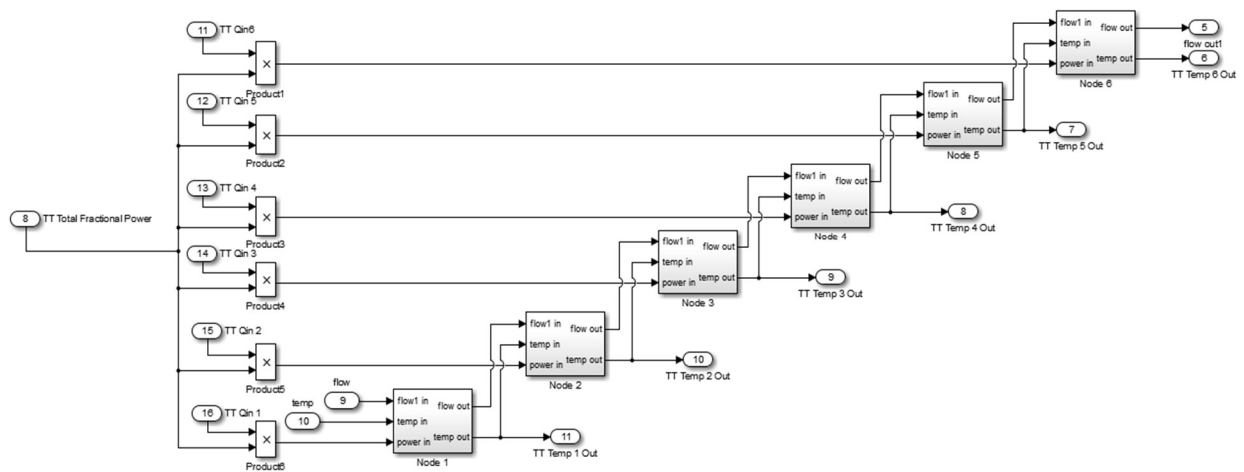


Figure 6: Iterative volumetric power deposition model in Simulink

Equation 5 shows how the volumetric power deposition (Q_{in}) is incorporated into subsequent convection heat transfer processes.

$$T_{out} = \frac{\pi DLh\Delta T + \dot{Q}_{in}}{\dot{m}Cp} \quad \text{Equation 5}$$

Using Equation 5, the specific dimensions of the fuel coolant channels were incorporated into the Simulink model as seen in Figure 7. The circumferential surface area of each channel was described by multiplying the inner-diameter (D) by pi and one sixth of the total length (L). This was multiplied by the heat transfer coefficient of hydrogen (h) and the temperature differential (ΔT) between the coolant channel wall and the mean hydrogen temperature. These values multiplied together equal the total power deposition due to convection in the conduit which is added to the volumetric power deposition (Q_{in}). To convert the total power deposition in a single leg of a heat transfer to an exit temperature (T_{out}), the power is divided by the mass flow rate ($\dot{m}$) and specific heat (Cp) of the hydrogen. This identical solution was performed six subsequent times to find the final exit temperature of a single pass through any given heat transfer conduit. Although a fuel coolant channel was used as the example, the same method was implemented for the tie-tubes only accounting for the different tubing dimensions.

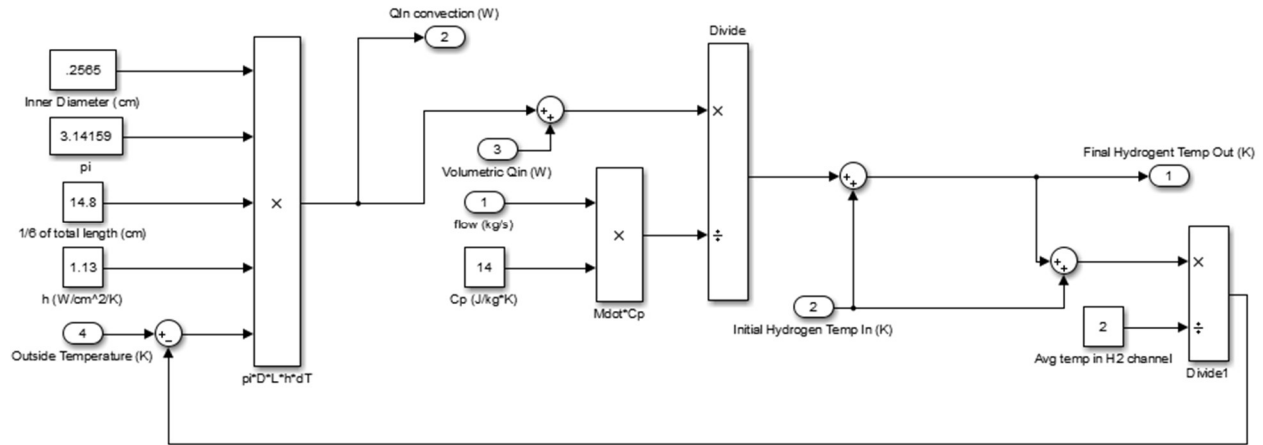


Figure 7: Heat transfer model for each iteration in a single core coolant channel

Fluid Dynamics

In a real hydrogen cooled NTP system, the hydrogen is in liquid, gaseous and combined two-phase states at various times and places during the heat transfer process. Modeling of these phase transitions and combined two-phase fluid dynamics is complex, and many assumptions were made for the state of hydrogen throughout the system. For the simulations, hydrogen was modeled

as an ideal gas. While not exact, this assumption gives a reasonable estimate of how the hydrogen gas will behave based on temperature, pressure, and volume changes from external perturbations. *Figure 8* shows the Simulink model of the Ideal Gas Law given by *Equation 6*.

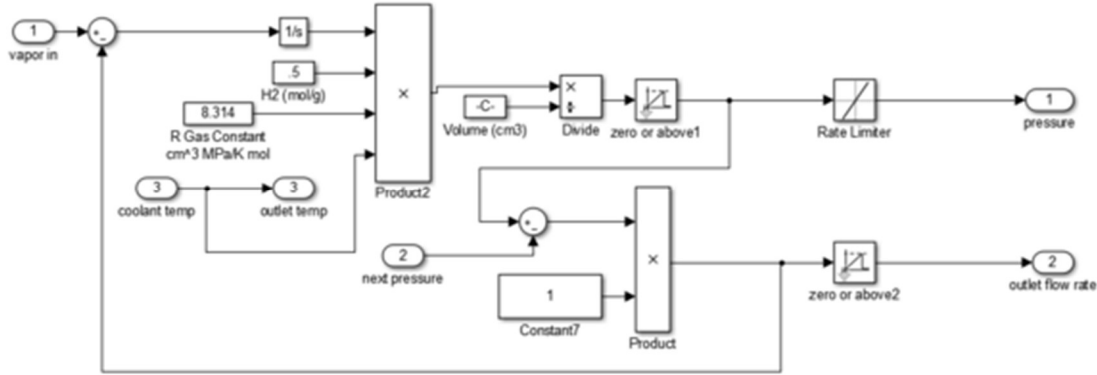


Figure 8: Simulink model of the Ideal Gas Law (Eq. 5)

$$PV = nRT \quad \text{Equation 6}$$

P is the pressure of the gas in a certain plenum (MPa), V is the volume of the plenum (cm³), n is the number of moles of hydrogen gas, R is the universal gas constant (cm³-MPa/mol-K), and T is the temperature of the gas (K). The model shows how the mass flow rate of the hydrogen gas (vapor in) is converted to a molar flow rate and integrated over time, thus giving the n value for the function. With known volumes and know inlet temperatures, the pressure in each plenum was determined.

Testing

After completing the Simulink model, the next step was to develop a novel way to efficiently run multiple simulations while iterating through multiple variables. In addition, each series of simulations was designed to automatically generate 25 post processing data sets outlining various parameters of interest within the system. These rapid testing and data processing methods allowed for efficient analysis of system performance while troubleshooting the model and ultimately supplied convenient tools for producing the final results. Three main parameters were determined as high priority for system performance:

1. Determine the proper sequence for a successful startup
2. Understand the importance of hydrogen flow and articulate a range of allowable flow rates under certain conditions
3. Describe a range of acceptable drum control rotation speeds and final drum positions

The following parameters were determined as high priority factors to be monitored during simulations to serve as a metric for startup diagnostics.

1. Temperature of fuel and tie tubes
2. Rate of change of fuel and tie tube temperature
3. Thermal power output of core
4. Total reactivity of system
5. Pressures in tie-tube, core and pump plenums

The reason that these are high priority factors is because of material stresses and potential material failures. Melting temperatures are of great concern but the rate of temperature changes, or thermal shock, were found to be a highly limiting factor on system performance as well. Along with the material concerns, general system power performance (or lack thereof) was of great interest too. Minimum thermal output from the core is approximately 362 MW_{th} and specific limits for various materials can be seen in *Table 3*.

Table 3: Material Properties for SNRE System

Component	Material	Melting Temperature (K)	Marginal Operating Temperature (K)	Thermal Shock Maximum (K/sec)
Fuel Element	(UZr)C	2900	2860	1000
Tie-Tube	Inconel-718	1609	1569	2000

*All data courtesy of M.M. El-Wakil [11]

Matlab served as a scripting interface to the Simulink model and a control script was developed to allow for iterations through different variables with user-specified ranges. Invoking this script provided graphical user interfaces (GUI) to permit users to easily select the variable that the user wanted to observe performance changes in. Once the variable was chosen, a second GUI would appear supplying options for testing a range of values for the variable of choice.

The examples in *Figures 9* and *10* show the initial GUI (*Figure 9*) where the user has chosen to test the effects of control drum velocity and the second GUI (*Figure 10*) shows an example of the range of values that the user may control to run a series of simulations. For example, the series of simulations chosen in *Figure 10* will run 10 separate simulations for 1000 seconds each with the drum velocity ranging from 0.1 degrees/second to 1.0 degrees/second in 0.1 degrees/second steps. The complete script is located in *Appendix B*.

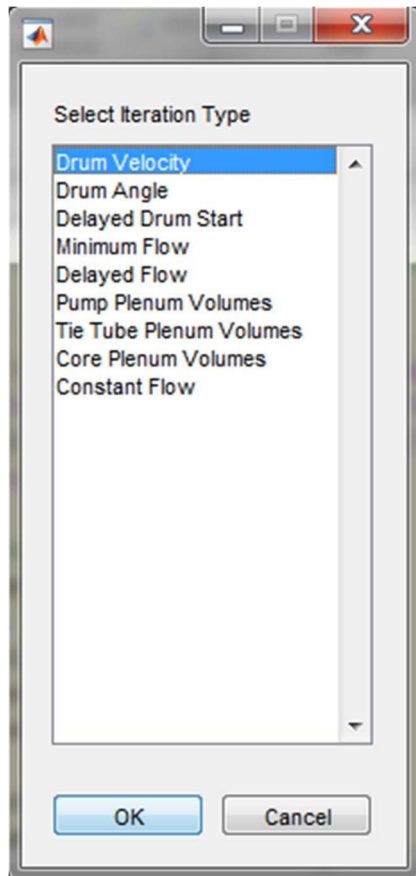


Figure 9: GUI for Simulink variable choice.

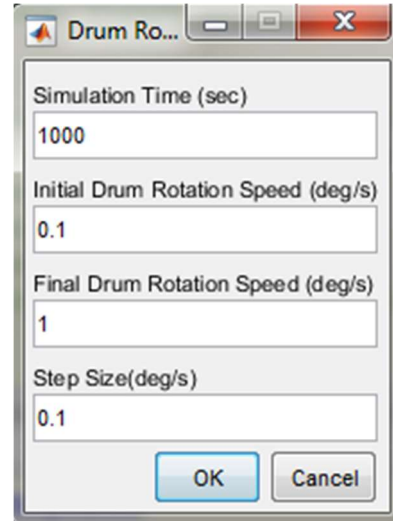


Figure 10: GUI to select specific range of values for the previously selected variable.

After the user selects the values, the Matlab script will initialize all of the appropriate input variables for the Simulink model to run. The model then uses a variable time-step solution process to solve the entire system of equations. For each simulation, multiple outputs are generated which are then fed back into Matlab for automatic post processing. Every time the simulation is run, twenty-five different plots will appear, showing a variety of trends in the model for the predetermined parameters of importance. An example showing all data plots for a single test is shown in *Appendix A*.

Before generating final results, it was essential to test the model for accuracy against accepted values of previously researched SNRE computational models. *Table 4* shows some values this simulation achieved along with their accompanying values from the previous SNRE models.

Table 4: Comparison of Results to SNRE Published Data

<i>Parameter</i>	<i>Published SNRE Data</i>	<i>Simulation Results</i>
Core Steady State Temp (K)	2728	2740 $\pm$ 20
Tie Tube Steady State Temp (K)	429	540 $\pm$ 10
Steady State Power (MW _{th})	362	370 $\pm$ 5

*All data courtesy of Schnitzler, Borowski, and Fittje

Results

After showing that the simulation was solving the time-differential equations correctly, multiple series of simulations were developed to illustrate how the range of the parameter of interest affects the performance of the system. For example, a parameter, such as hydrogen flow, would run an initial simulation that would illustrate the effects of an extremely low flow. The model would predict that materials will exceed their melting temperatures. The next simulation would incrementally increase the hydrogen flow and then run the full simulation again. In this example, the parameter's range of effect on the system could be observed from low-end failure (melt down) to high-end failure (inefficient use of hydrogen) to outline the optimum performance range between. This logic and process was applied to the following sequences and parameters of interest:

1. Startup sequence (hydrogen flow before drum rotation and drum rotation before hydrogen flow)
2. Hydrogen flow magnitude
3. Drum rotational velocity.

The results of interest for each simulation series are presented in two forms: time dependent profiles and the reduced data sets for each time dependent profile.

In the time dependent profiles, the value of the parameter of interest (temperature, power, etc.) are displayed on the ordinate axis while the time of the profile will be displayed on the abscissa axis. All time dependent profiles will display multiple simulations in a single graph for each iteration of the parameter of interest. Each full simulation for each iterative step of the parameter will be represented by a separately colored line so the change of the system's performance can be observed as the parameter is changed over a series of values.

The reduced data sets will display a particular point of interest from a time dependent profile, like the maximum or minimum value of the entire simulation. The graphs for the reduced data sets will display this single value of interest from the previous time dependent profile on the ordinate axis while the iterated parameter's value for each simulation will be displayed on the abscissa axis. This allows rapid observation of critical values from the time dependent profiles and makes trends of the system performance more identifiable.

Startup Sequence

To test the startup sequence, two possible scenarios were tested. First, the drums were initiated without any hydrogen flow in the system. This was achieved by having the drum start time iterate from 0-1,000 seconds in 100 second increments. The hydrogen flow start time was always at 500 seconds. This means that of the 10 simulations ran, the first five showed the effects of drum rotation without hydrogen in the system and the second five showed the effects of starting the drums with hydrogen in the system. *Figure 11* shows the temperature profiles in the core (left)

and tie tubes (right) for each simulation. The solid line represents the melting temperature of each material and the dashed line indicates the 40K operational margin from melting temperatures. All relevant parameters and their values can be found in *Table 5*.

Table 5: Parameter Values to Test Startup Sequence

<i>Parameter</i>	<i>Variable or Constant</i>	<i>Value or Range</i>
Hydrogen Start Time (sec)	Variable	0-1000
Drum Start Time (sec)	Constant	500
Hydrogen Flow (kg/s)	Constant	8
Drum Rotational Velocity (degrees/sec)	Constant	1

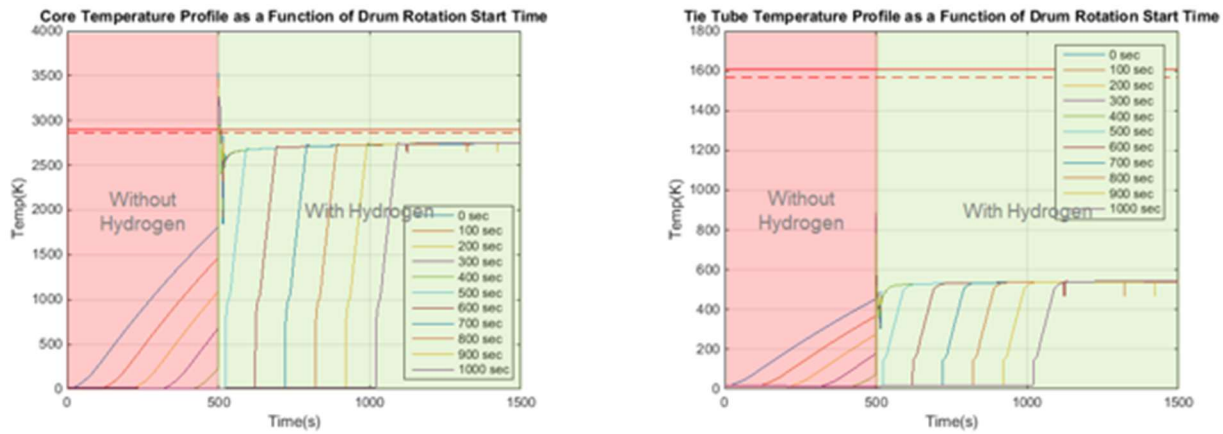


Figure 11: Core and tie tube temperature profiles due to varying drum rotation start times.

The pink-shaded region shows the area when the drums begin rotating before the hydrogen flow. At $t=500$ seconds, the hydrogen begins flowing. From there, the green-shaded region shows the area where the hydrogen flow starts before the drums begin to rotate. The graph on the left shows that when the drums begin rotating before the hydrogen flow begins, the core temperature exceeds the fuel melting temperature. However, on the right, the tie tube temperature does not exceed its melting point. This is further illustrated below in *Figure 12*, which shows the extracted maximum temperature values for the core and tie tube for the same simulation.

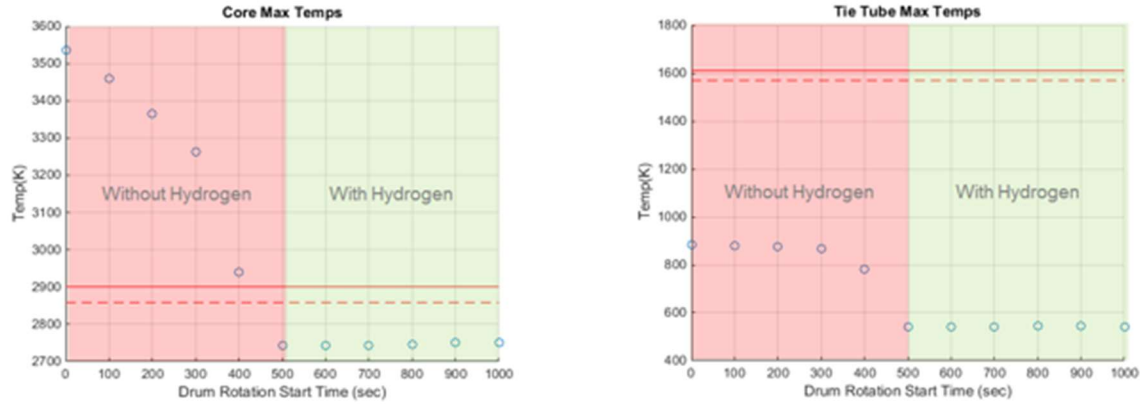


Figure 12: Reduced data of control drum rotation start time intervals showing maximum core and tie tube temperatures.

This reduced data version gives the same conclusion as before, only highlighting extreme points of interest (maximum temperature of profile). Another variable of interest when determining the feasibility of the sequencing and, the other two variables of interest, is the time derivative of temperature or the rate of change of temperature. Since rapidly heating up or cooling down a material can cause thermal shock, it is essential to keep this rate of change below the prescribed limit (see Table 3). The results of the maximum temperature derivatives from each simulation are shown below in Figure 13.

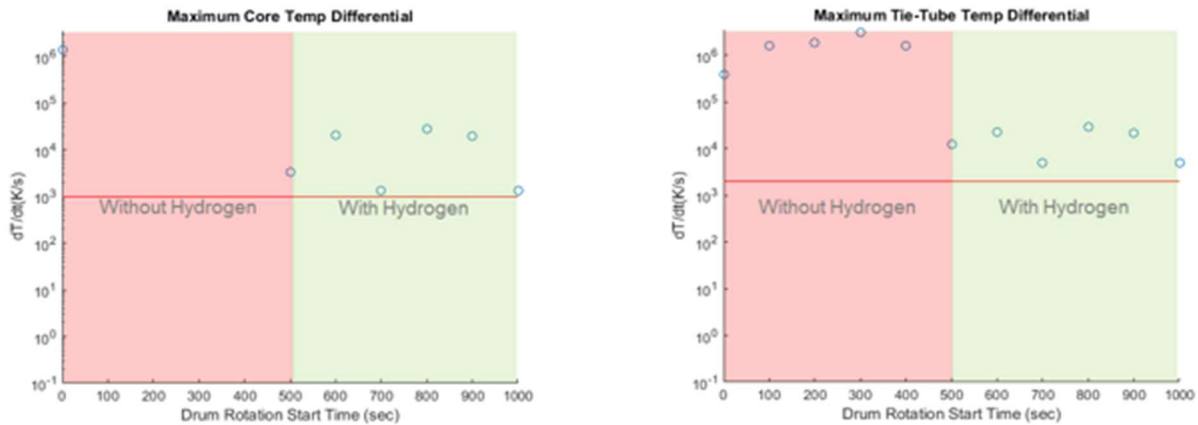


Figure 13: Temperature time derivatives for varying drum rotation start times for the core and tie tubes

From these graphs, it is clear that neither starting the drum rotation before nor after would lead to an acceptable temperature differential in the core or the tie tubes. That being said, having the drum rotation start with or after the hydrogen flow is orders of magnitude better in regards to temperature derivatives.

From these data, it is a necessary condition for the control drums to begin rotating at the same time or after the hydrogen flow begins to avoid meltdown. The subject of thermal stress due

to excessive temperature differentials will be discussed further when looking at hydrogen flow rates and control drum rotation speeds.

Hydrogen Flow

The next variable of interest that was tested was the hydrogen flow rate. Hydrogen acts as a coolant, moderator, and propellant for a thermal NTP system, Therefore, a high degree of understanding of the system response caused by hydrogen flow is crucial for ascertaining optimum performance of the system. To identify the effects of hydrogen flow on system performance the magnitude of flow was incrementally simulated from 1 - 20 kg/second in increments of 1 kg/second (twenty individual simulations). *Figure 14* show the core and tie tube temperature profiles as a function of time for all twenty simulations. All relevant variables and variable ranges for the hydrogen flow simulations may be found in *Table 6*.

Table 6: Parameter Values to Test Hydrogen Flow Rate

Parameter	Variable or Constant	Value or Range
Hydrogen Start Time (sec)	Constant	0
Drum Start Time (sec)	Constant	0
Hydrogen Flow (kg/s)	Constant	1-20
Drum Rotational Velocity (degrees/sec)	Constant	1

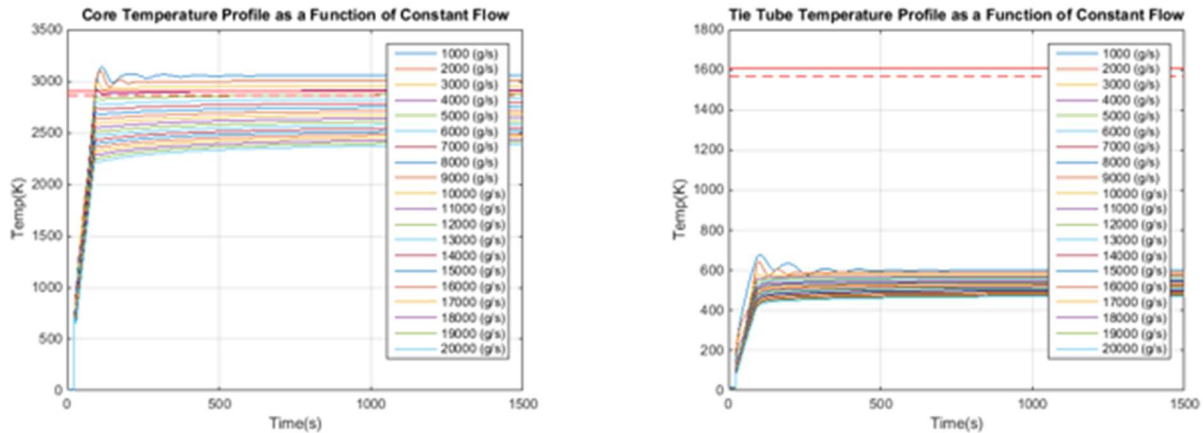


Figure 14: Core and tie tube temperature profiles as a function of time with varying hydrogen flow rates

From these graphs, a reduced data set of maximum temperatures can be generated and graphed (*Figure 15*).

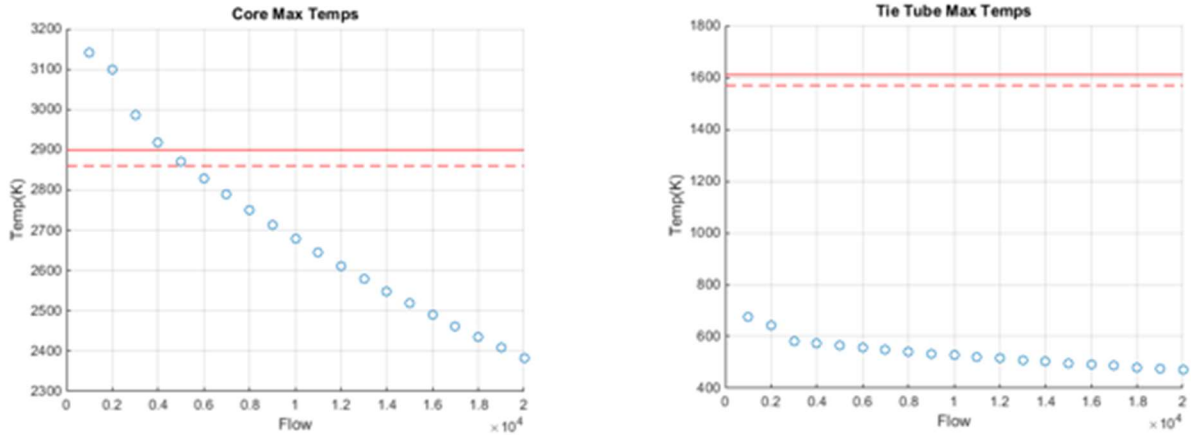


Figure 15: Reduced data sets of maximum core and tie tube temperatures as a function of flow rate

These graphs show that the flow must be above a value of ~6 kilograms per second to ensure that the core does not exceed its maximum temperature. Much like with the sequencing, the differential temperature is also an important value to consider, shown following in *Figure 16*. These graphs, again, show that the maximum temperature differential in the core is exceeded, although it is no longer exceeded in the tie tubes.

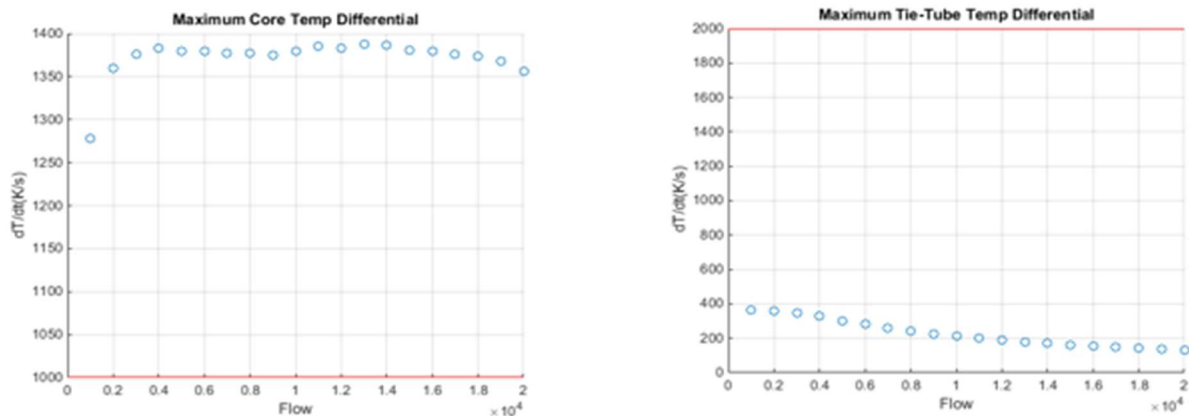


Figure 16: Core and tie tube temperature differentials as a function of hydrogen flow rate

From all three sets of graphs, it can be determined that the hydrogen flow rate must be equivalent to or exceed 6 kilograms/second in order to prevent the core from melting down. This ensures that the core and tie tubes are well below melting temperature, although the issue of the temperature differentials must be resolved by control drum rotation speed.

Control Drum Rotation Speed

The final series of simulations that were performed involved varying the control drum rotation speed. Since neither the sequencing nor the hydrogen flow rate could control the temperature differentials, it was hoped that the control drum rotation speed would. Much like

before, the first set of graphs (*Figure 17*) shows the temperature profiles for the core and tie tubes as a function of time, while the next set (*Figure 18*) show the maximum temperature values for the core and tie tubes as a function of drum rotation speed. Both figures show that the control drum speed does not significantly alter the maximum temperatures in either plenum. However, that being said, the control drum rotational velocity does affect the startup transient duration. The SNRE is designed to accelerate for only 20 minutes, therefore it is exceedingly important that the startup transient be no more than 10% of the total time, or 2 minutes. Slower ramp-up speeds, while conservative with temperature differentials, are extremely wasteful of the limited propellant volume so an optimum nominal speed fast enough to be efficient yet slow enough to not cause thermal shock is desired.

Table 7: Parameter Values to Rest Control Drum Rotation Speed

Parameter	Variable or Constant	Value or Range
Hydrogen Start Time (sec)	Constant	0
Drum Start Time (sec)	Constant	0
Hydrogen Flow (kg/s)	Constant	8
Drum Rotational Velocity (degrees/sec)	Constant	0.1-2.1

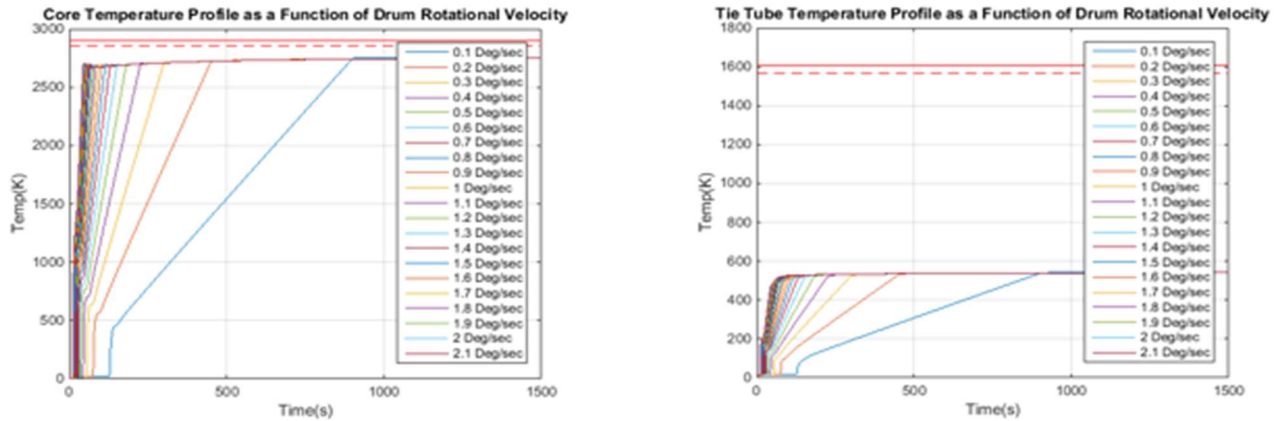


Figure 17: Core and tie tube temperature profiles as a function of time with varying drum rotational velocities

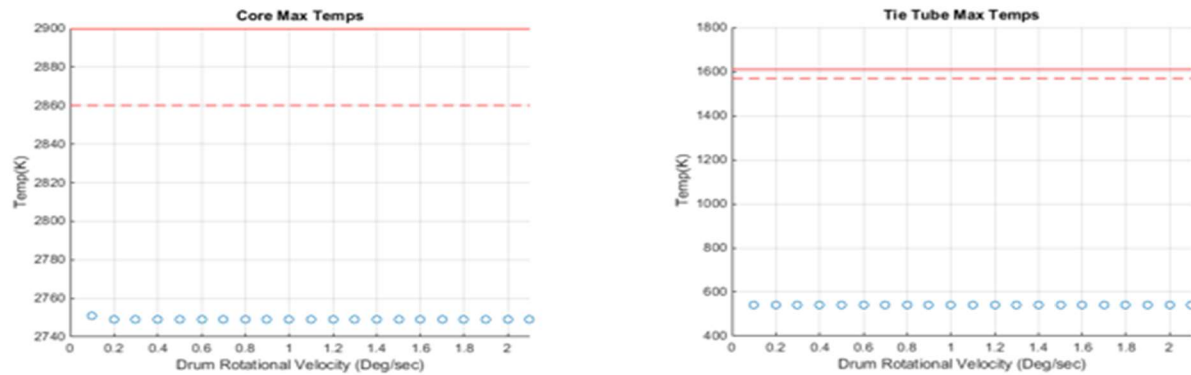


Figure 18: Core and tie tube maximum temperatures as a function of drum rotational velocity

Figure 19 illustrates that the control drum speed does have a significant impact on the temperature differentials, as a velocity of <0.8 degrees/second removes any previous problems with exceeding the rate of thermal shock limits. Therefore, it is necessary that the control drum rotational velocity to be less than or equivalent to 0.7 degrees/second to avoid any thermal shock issues that can arise from extreme temperature differentials.

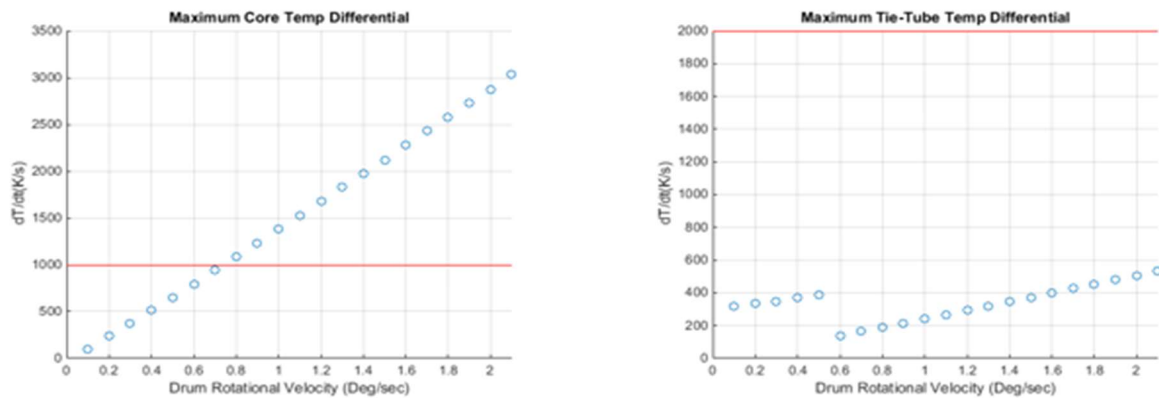


Figure 19: Maximum temperature differentials in the core and tie tubes as a function of drum rotational velocity

Final Results

From the previous simulations, necessary control requirements to ensure a safe start-up of an NTP system are as follows:

1. Hydrogen flow must precede or begin at the same time as control drum rotation.
2. The flow rate of hydrogen through the system must be in excess of 6 kilograms/second.
3. Drum rotational velocity cannot exceed 0.7 degrees/second, but cannot be so low as to induce extended start-up transients.

Using those requirements, a set of recommendations can be made for acceptable ranges with which to consider when engineering control systems for the NTP system.

1. It is recommended to start the hydrogen flow and the control drum rotation at the same time during the startup transient. This ensures that the core remains below the accepted limit of its marginal melting temperature.
2. It is recommended that the flow rate of hydrogen through the system be between 8 and 9 kilograms/second. This is what SNRE was engineered to have, and it is what the Simulink model agreed was an acceptable flow rate.
3. It is recommended that the rotational velocity of the control drums be 0.7 degrees/second. This ensures that the temperature differentials are within an acceptable range and that the start-up transient takes up no more than 10% of the total burn time.

From those recommendations, *Figures 20-21* show the temperature profiles and temperature differentials. In all following simulations notice that no high priority factor exceeds its limit, and the system is running at a steady-state temperature and power after just 2 minutes.

Table 8: Parameter Values for Final Results

<i>Parameter</i>	<i>Variable or Constant</i>	<i>Value or Range</i>
Hydrogen Start Time (sec)	Constant	0
Drum Start Time (sec)	Constant	0
Hydrogen Flow (kg/s)	Constant	8 and 9
Drum Rotational Velocity (degrees/sec)	Constant	0.7

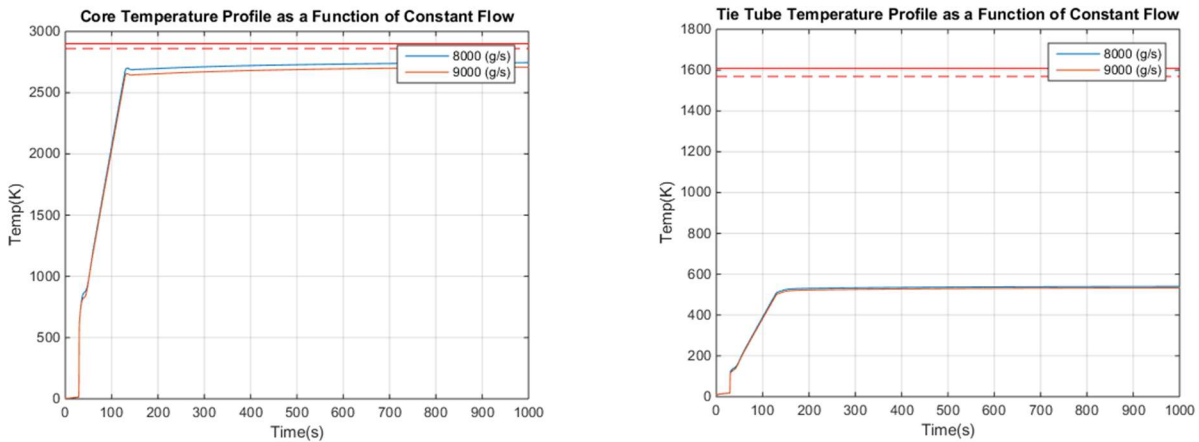


Figure 20: Core and tie tube temperature profiles as a function of time for flows of 8 and 9 kg/s with a drum rotational velocity of 0.7 degrees/s and the same start time for the control drums and hydrogen flow.

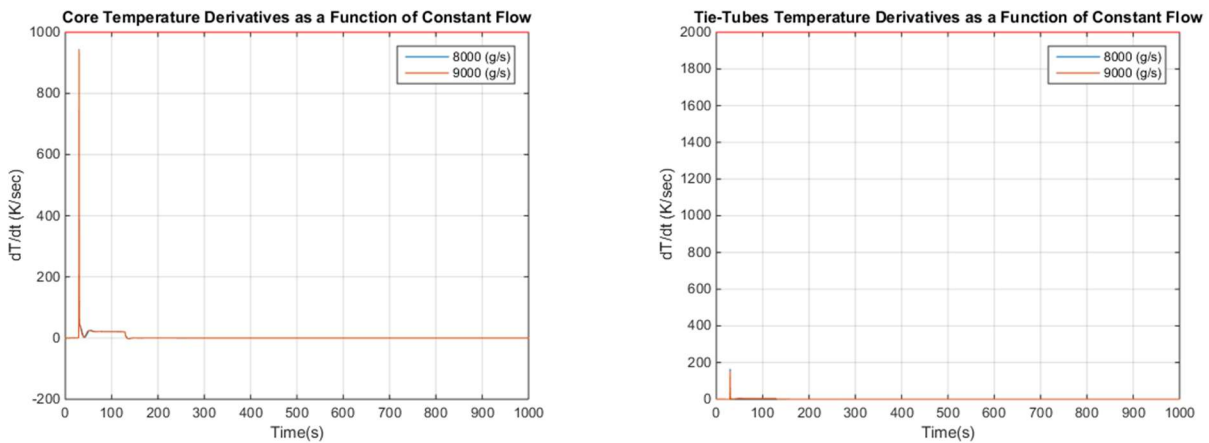


Figure 21: Temperature differentials as a function of time for flows of 8 and 9 kg/s with a drum rotational velocity of 0.7 degrees/s and the same start time for the control drums and hydrogen flow.

Discussion

The Simulink model developed is a preliminary first order approximation of a complex process. The values presented in this paper serve as recommendations for future models, which can be significantly improved. An in-depth Monte Carlo analysis of the core with its desired geometry must be completed to look for variations in reactivities, more precise heat generation maps, and experimentally determined point kinetics values. There also needs to be a more accurate fluid dynamics models using computational fluid dynamics techniques. Furthermore, a computational and physical analysis of structural material must be developed to assess the effects of extreme temperatures, high hydrogen concentrations, thermal shock, vibrations, and material degradation due to extreme radiation exposure.

Conclusion

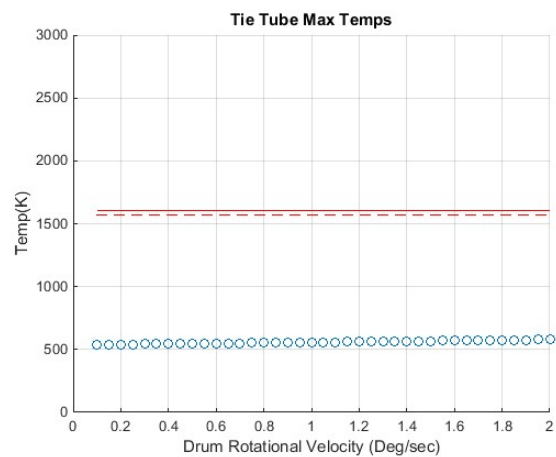
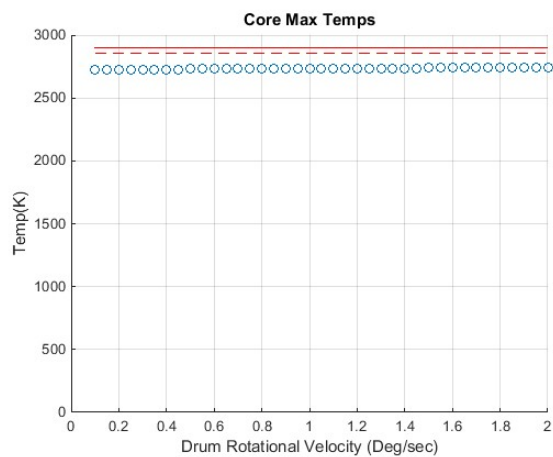
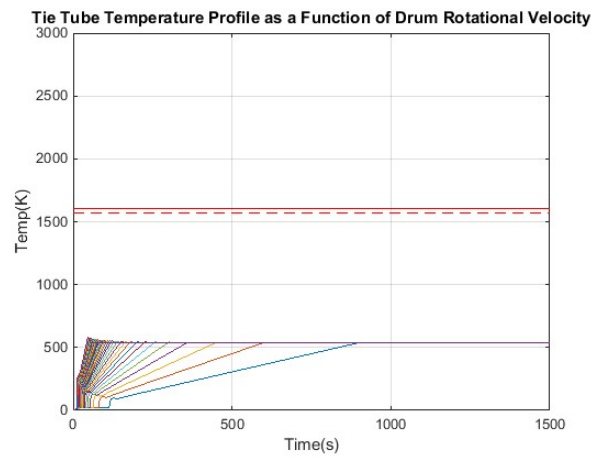
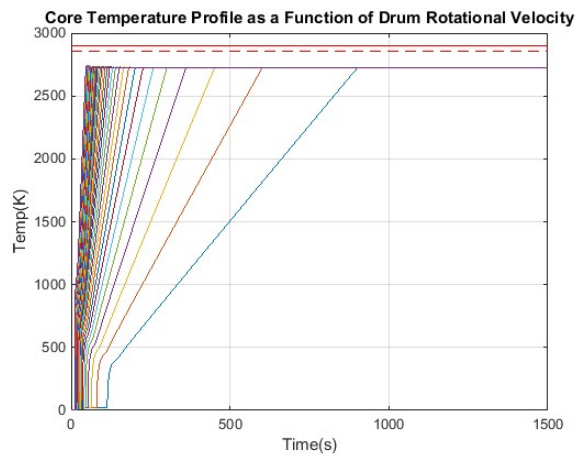
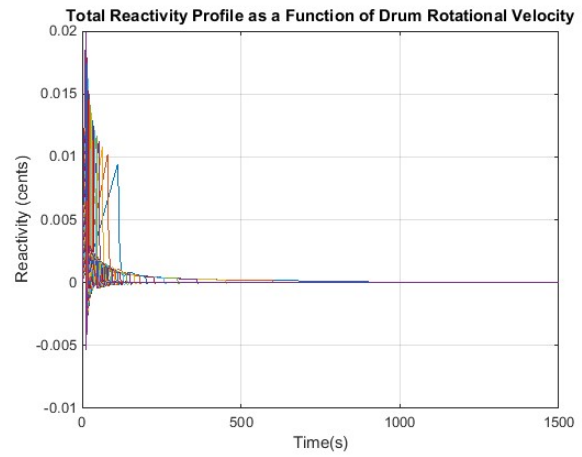
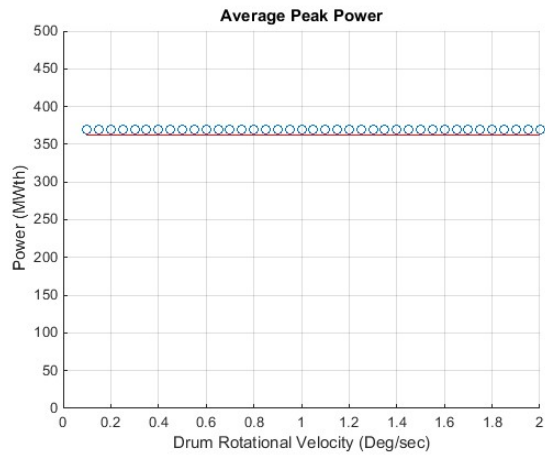
It was found that Simulink is a viable software package to model the point kinetics equations and simplified thermal hydraulic behavior of a dynamic SNRE system. The model and corresponding Matlab code offers a simplified way to quickly iterate through numerous variables in order to determine specific control requirements. From this model, a variety of simulations were run to determine the starting sequence, the hydrogen flow rate, and the control drum rotational velocity necessary to produce a safe and effective start-up of a moderated NTP system. Using data from these simulations along with known system requirements from past literature, a set of specific recommendations were developed for the SNRE system to explicitly demonstrate the viability of a safe and efficient NTP start-up. These recommendations include:

1. It is recommended to start the hydrogen flow and the control drum rotation at the same time during the startup transient. This ensures that the core remains below the accepted limit of its marginal melting temperature.
2. It is recommended that the flow rate of hydrogen through the system be between 8 and 9 kilograms/second. This is what SNRE was engineered to have, and it is what the Simulink model agreed was an acceptable flow rate.
3. It is recommended that the rotational velocity of the control drums be 0.7 degrees/second. This ensures that the temperature differentials are within an acceptable range and that the start-up transient takes up no more than 10% of the total burn time.

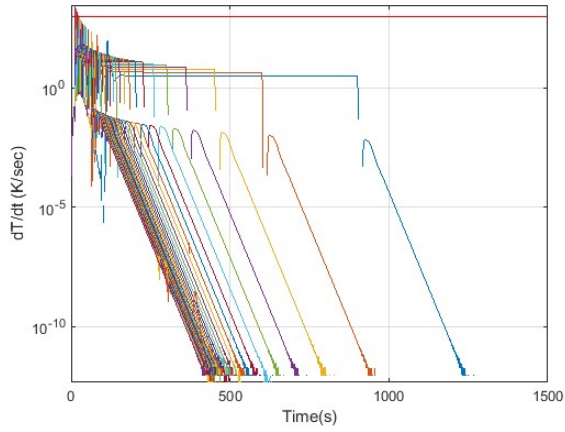
References

- [1] S. K. Borowski, “Nuclear Thermal Propulsion (NTP): A Proven, Growth Technology for “Fast Transit” Human Mission to Mars”, NASA, Glenn Research Center, Cleveland OH, 2014.
- [2] J. B. Holt, T. S. Monk, “Propellant Mass Fraction Calculation Methodology for Launch Vehicles and Application to Ares Vehicles”, NASA Marshall Space Flight Center, Huntsville AL, 2009
- [3] H. F. Crouch, Nuclear Space Propulsion, Astronuclear Press, 1965.
- [4] H. S. Nam, et al., “Innovative concept for an ultra-small nuclear thermal rocket utilizing a new moderated reactor”, Department of Nuclear and Quantum Engineering, Korea Advanced Institute of Science and Technology and Handong Global University, Republic of Korea, 2015
- [5] Lt Col T.J. Lawrence, “Nuclear Thermal Rocket Propulsion Systems”, U.S. Air Force Academy Department of Astronautics, USAFA, Colorado, 2005
- [6] J.L. Finseth, “Overview of Rover Engine Tests: Final Report”, NASA Marshall Space Flight Center, Huntsville, AL, 1991.
- [7] B. Schnitzler, S. Borowski and J. Fittje, "25,000-lbf Thrust Engine Options Based on the Small Nuclear Rocket Engine Design," in *45th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, Denver, 2009.
- [8] B. Schnitzler, “Neutronics Models and Analysis of the Small Nuclear Rocket Engine (SNRE)”, 43rd AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit, 8-11 July 2007, Cincinnati, OH.
- [9] M. E. M. Stewart and B. G. Schnitzler, "Multidisciplinary Simulation of Graphite-Composite and Cermet Fuel Elements for NTP Point of Departure Designs," in *AIAA SPACE 2015 Conference and Exposition, SPACE Conferences and Exposition*, 2015.
- [10] W. D. Kingery, H. K. Bowen, and D. R. Uhlmann, *Introduction to ceramics*, 2nd ed. New York: Wiley, 1976.
- [11] M. M. El-Wakil, “Nuclear Heat Transport,” International Textbook Company, Intext Publisher, Scranton, 1971.

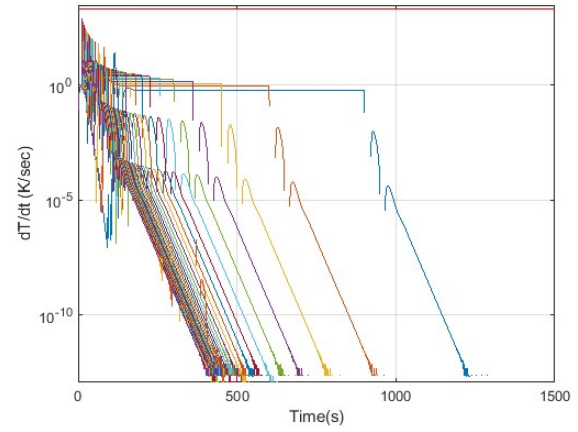
Appendix A: Example of full data set from a single iterative simulation.



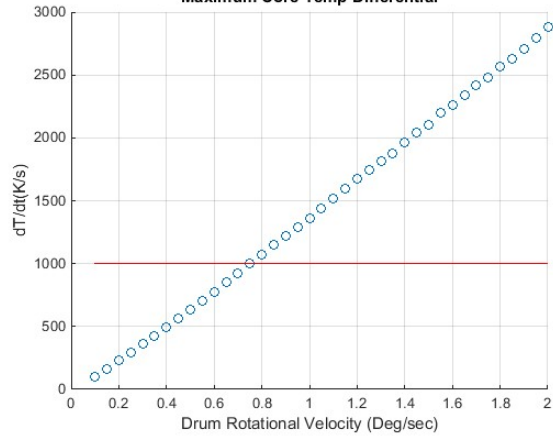
Core Temperature Derivatives as a Function of Drum Rotational Velocity



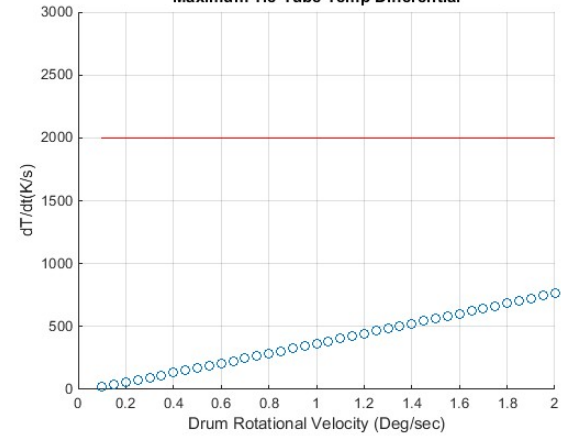
Tie-Tubes Temperature Derivatives as a Function of Drum Rotational Velocity



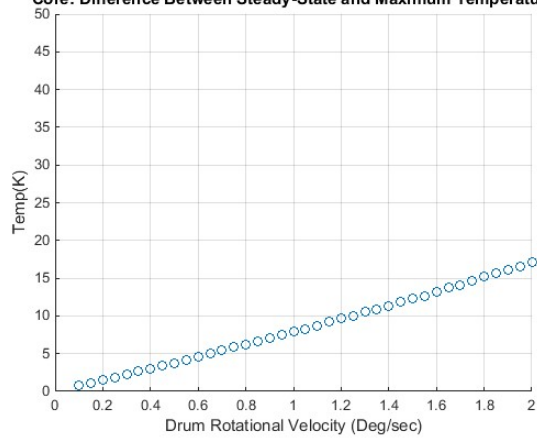
Maximum Core Temp Differential



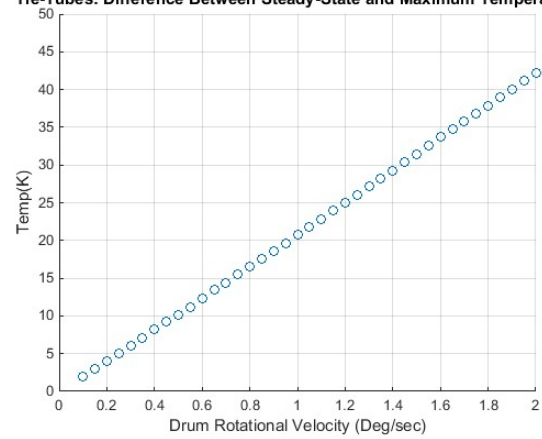
Maximum Tie-Tube Temp Differential

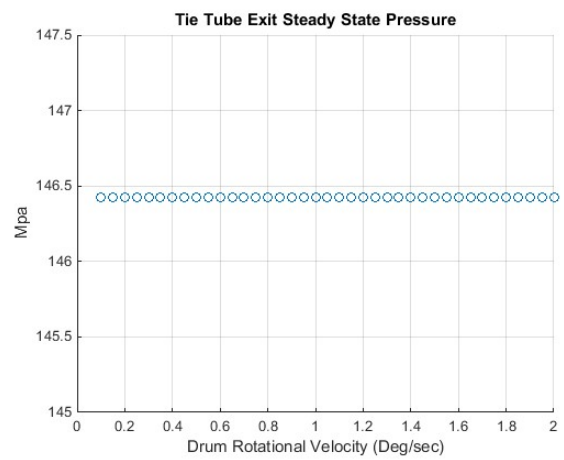
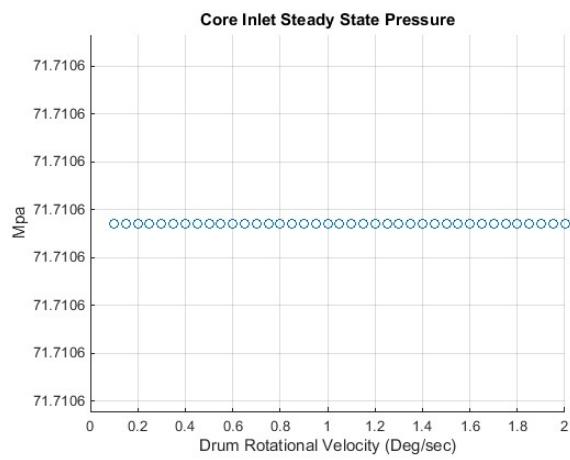
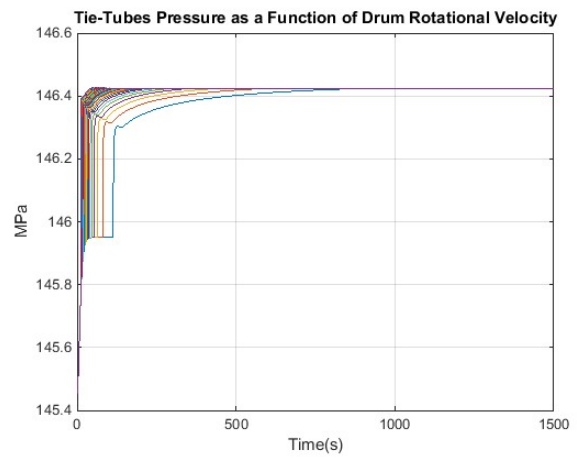
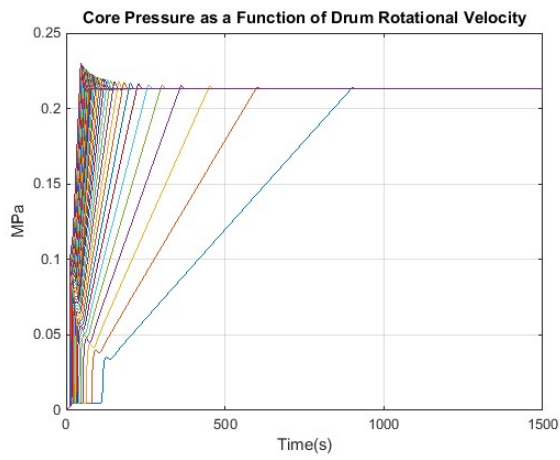
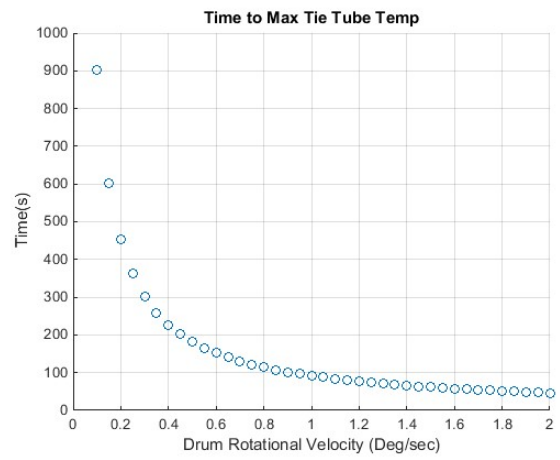
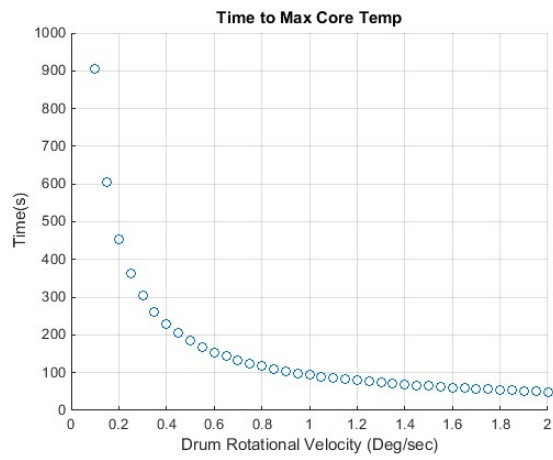


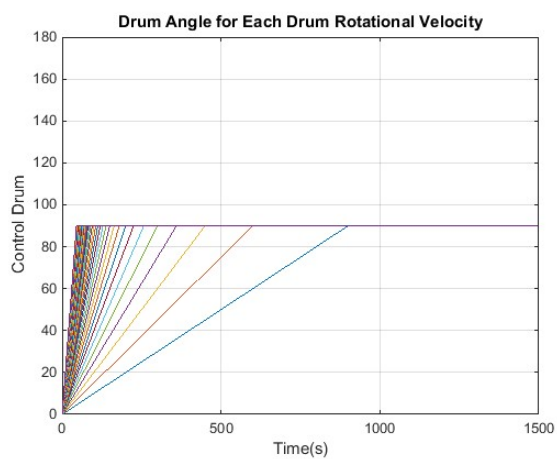
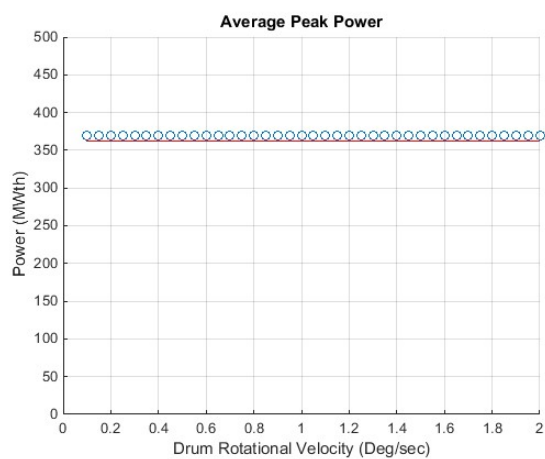
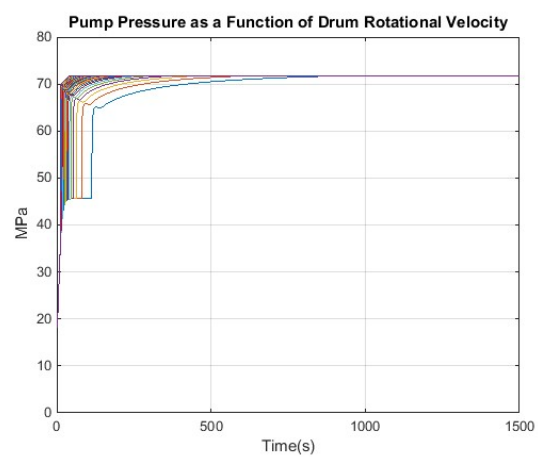
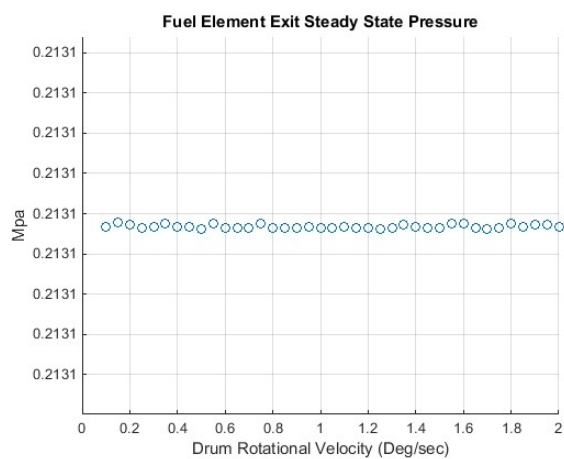
Core: Difference Between Steady-State and Maximum Temperature



Tie-Tubes: Difference Between Steady-State and Maximum Temperature







Appendix B: Matlab Code

```
clc

str = {'Drum Velocity','Drum Angle','Delayed Drum Start','Minimum Flow','Delayed Flow','Pump Plenum Volumes',...
      'Tie Tube Plenum Volumes','Core Plenum Volumes','Constant Flow'};
prompt1 = listdlg('PromptString','Select Iteration Type', 'SelectionMode','single',...
      'ListString',str);

%Simulation Time
SimTime = 1000;
%Drums
DrumVelocity= 1; %Degrees/s
DrumPosition = 90; %Degrees
DrumStartTime = 0; %Seconds
%Flow
FracMinFlow = 1;
FlowTime=0; %Sec
ConstFlowStatus=0;
FlowIn=8050;
%Tie Tubes
TTPlenumVolume = 4500; %cm^3
%Pump
PumpPlenumVolume = 100; %cm^3
%Core
CorePlenumVolume = 746000; %cm^3
NeutronLifeTime=.001;
Beta=.0075;
Lambda=.076;
%Temperatures
Core_Temp=0;
TT_Temp=0;
CoreTempDeriv = 0;
TTTempDeriv = 0;

%Concatinating Pressure
TT_Pressure=0;
Pump_Pressure=0;
Core_Pressure=0;
% Prompt user for input
if prompt1==1
    prompt = {'Simulation Time (sec)','Initial Drum Rotation Speed (deg/s)','Final Drum Rotation Speed (deg/s)','Step
Size(deg/s)'};
    dlg_title = 'Drum Rotational Velocity';
    num_lines = 1;
    defaultans = {'1000','0.1','1','0.1'};
    user_input = inputdlg(prompt,dlg_title,num_lines,defaultans);

    %Plotting
    reduced_Xlabel='Drum Rotational Velocity (Deg/sec)';
    profile_units = ' Deg/sec';
    profile_title= ' Drum Rotational Velocity';

elseif prompt1==2
```

```

    prompt = {'Simulation Time (sec)', 'Initial Drum Position (Degrees)', 'Final Drum Position (Degrees)', 'Step Size (Degrees)'};
    dlg_title = 'Drum Position';
    num_lines = 1;
    defaultans = {'1000', '0', '90', '10'};
    user_input = inputdlg(prompt, dlg_title, num_lines, defaultans);

    %Plotting
    reduced_Xlabel = 'Final Drum Angle (Deg)';
    profile_units = ' Deg';
    profile_title = ' Final Drum Angle';
elseif prompt1 == 3
    prompt = {'Simulation Time (sec)', 'Initial Drum Start Time (Sec)', ...
        'Final Drum Start Time (Sec)', 'Step Size (Sec)', 'Drum Velocity (deg/s)'};
    dlg_title = 'Drum Start Time';
    num_lines = 1;
    defaultans = {'1000', '0', '100', '10', '1'};
    user_input = inputdlg(prompt, dlg_title, num_lines, defaultans);

    DrumVelocity = str2num(user_input{5});

    %Plotting
    reduced_Xlabel = 'Drum Rotation Start Time (sec)';
    profile_units = ' sec';
    profile_title = ' Drum Rotation Start Time';
elseif prompt1 == 4
    prompt = {'Simulation Time (sec)', 'Initial Minimum Flow Threshold (Fractional)', 'Final Minimum Flow Threshold (Fractional)', ...
        'Step Size (Fractional)', 'Mass Flow Rate (g/s)'};
    dlg_title = 'Minimum Flow Threshold';
    num_lines = 1;
    defaultans = {'1000', '0.1', '1', '0.1', '8050'};
    user_input = inputdlg(prompt, dlg_title, num_lines, defaultans);

    FlowIn = str2num(user_input{5});

    %Plotting
    reduced_Xlabel = 'Minimum Flow Threshold';
    profile_units = ' Fractional';
    profile_title = ' Minimum Flow Threshold';
elseif prompt1 == 5
    prompt = {'Simulation Time (sec)', 'Initial Flow Start Time (Sec)', 'Final Flow Start Time (Sec)', ...
        'Step Size (Sec)', 'Mass Flow Rate (g/s)'};
    dlg_title = 'Flow Start Time';
    num_lines = 1;
    defaultans = {'1000', '0', '100', '10', '8050'};
    user_input = inputdlg(prompt, dlg_title, num_lines, defaultans);

    FlowIn = str2num(user_input{5});

    %Plotting
    reduced_Xlabel = 'Flow Start Time (sec)';
    profile_units = ' (sec)';

```

```

        profile_title= 'Flow Start Time';
elseif prompt1==6
    prompt = {'Simulation Time (sec)','Initial Plenum Volume (cm^3)','Final Plenum Volume (cm^3)','Step Size (cm^3)'};
    dlg_title = 'Pump Plenum Volume';
    num_lines = 1;
    defaultans = {'1000','90','110','5'};
    user_input = inputdlg(prompt,dlg_title,num_lines,defaultans);

    %Plotting
    reduced_Xlabel='Pump Plenum Volume (cm^3)';
    profile_units = ' (cm^3)';
    profile_title= ' Pump Plenum Volume';
elseif prompt1==7
    prompt = {'Simulation Time (sec)','Initial Plenum Volume (cm^3)','Final Plenum Volume (cm^3)','Step Size (cm^3)'};
    dlg_title = 'Tie Tubes Plenum Volume';
    num_lines = 1;
    defaultans = {'1000','40500','49500','1800'};
    user_input = inputdlg(prompt,dlg_title,num_lines,defaultans);

    %Plotting
    reduced_Xlabel='Tie Tube Plenum Volume (cm^3)';
    profile_units = ' (cm^3)';
    profile_title= ' Tie Tube Plenum Volume';
elseif prompt1==8
    prompt = {'Simulation Time (sec)','Initial Plenum Volume (cm^3)','Final Plenum Volume (cm^3)','Step Size (cm^3)'};
    dlg_title = 'Core Plenum Volume';
    num_lines = 1;
    defaultans = {'1000','671400','820600','29840'};
    user_input = inputdlg(prompt,dlg_title,num_lines,defaultans);

    %Plotting
    reduced_Xlabel='Core Plenum Volume (cm^3)';
    profile_units = ' (cm^3)';
    profile_title= ' Core Plenum Volume';
elseif prompt1==9
    prompt = {'Simulation Time (sec)','Initial Flow (g/s)','Final Flow (g/s)',...
        'Step Size (g/s)','Fraction of max Flow'};
    dlg_title = 'Constant Flow';
    num_lines = 1;
    defaultans = {'1000','8000','9000','100','1'};
    user_input = inputdlg(prompt,dlg_title,num_lines,defaultans);

    FracMinFlow=str2num(user_input{5});
    ConstFlowStatus=1;

    %Plotting
    reduced_Xlabel='Flow';
    profile_units = ' (g/s)';
    profile_title= ' Constant Flow';
end

% Simulation Time
SimTime = user_input{1};

```

% Variable Time

```
variable_start = str2num(user_input{2});  
variable_stop = str2num(user_input{3});  
variable_step = str2num(user_input{4});
```

% Iteration Array

```
variable_array = [variable_start:variable_step:variable_stop];
```

%This preallocates the SimOut arrays

```
CoreMaxTemp=zeros(1,((variable_stop-variable_start)/variable_step));  
TTMaxTemp=zeros(1,((variable_stop-variable_start)/variable_step));  
CoreMaxTempTime=zeros(1,((variable_stop-variable_start)/variable_step));  
TTMaxTempTime=zeros(1,((variable_stop-variable_start)/variable_step));  
Peak_Power=zeros(1,((variable_stop-variable_start)/variable_step));  
Tot_Reactivity=zeros(1,((variable_stop-variable_start)/variable_step));  
TT_Pump_Flow=zeros(1,((variable_stop-variable_start)/variable_step));  
MaxPower=zeros(1,((variable_stop-variable_start)/variable_step));  
MaxCoreDerivs=zeros(1,((variable_stop-variable_start)/variable_step));  
MaxTTDerivs=zeros(1,((variable_stop-variable_start)/variable_step));  
TT_SteadyStateTemp=zeros(1,((variable_stop-variable_start)/variable_step));  
TT_DiffTemps=zeros(1,((variable_stop-variable_start)/variable_step));  
Core_SteadyStateTemp=zeros(1,((variable_stop-variable_start)/variable_step));  
Core_DiffTemps=zeros(1,((variable_stop-variable_start)/variable_step));  
Core_In_SteadyStatePress=zeros(1,((variable_stop-variable_start)/variable_step));  
Core_Out_SteadyStatePress=zeros(1,((variable_stop-variable_start)/variable_step));  
TT_SteadyStatePress=zeros(1,((variable_stop-variable_start)/variable_step));
```

```
counter=0;
```

```
for i=variable_array;
```

```
    if prompt1==1  
        DrumVelocity = i;  
    elseif prompt1==2  
        DrumPosition=i;  
    elseif prompt1==3  
        DrumStartTime=i;  
    elseif prompt1==4  
        FracMinFlow=i;  
    elseif prompt1==5  
        FlowTime=i;  
    elseif prompt1==6  
        PumpPlenumVolume=i;  
    elseif prompt1==7  
        TTPlenumVolume=i;  
    elseif prompt1==8  
        CorePlenumVolume=i;  
    elseif prompt1==9  
        FlowIn=i;  
    end
```

```
    counter=counter+1;
```

```
simout=sim('NTP_Master_Model','SaveTime','on','SaveOutput','on','StopTime',SimTime);
```

```

%Clears readouts from Simulink on each iteration
clc

%Pulling timeseries from "simout" class
HydrogenFlows = simout.get('TT_Pump_Core_Flows');
time=simout.get('tout');
Core_Temp1=simout.get('Core_Temp_Out');
Core_Temp_Deriv=simout.get('Core_Temperature_Deriv');
TT_Temp1=simout.get('TT_Temp_Out');
TT_Temp_Deriv=simout.get('TT_Temperature_Deriv');
Frac_Peak_Power=simout.get('PercentPeakPower');
PeakPower=simout.get('PeakPower');
TTPumpFlow=simout.get('Flow');
H2_Temp_and_Density_Reactivity=simout.get('Temp_Density_Reactivity');
Total_Reactivity=simout.get('Total_Reactivity');
TT_Pressure1=simout.get('TT_Pressure');
Pump_Pressure1=simout.get('Pump_Pressure');
Core_Pressure1=simout.get('Core_Pressure');

%Take max of derivative array
MaxCoreDerivs(counter) = max(Core_Temp_Deriv.data);
MaxTTDerivs(counter) = max(TT_Temp_Deriv.data);
%Populating Max FracPeakPower
MaxPower(counter)=mean(PeakPower.data);

%Populating Max Temp Arrays, Steady States and Differences
CoreMaxTemp(counter)=max(Core_Temp1.data);
Core_SteadyStateTemp(counter)= mean(Core_Temp1.data([round(length(Core_Temp1.data)*.9):end]));%This takes
the last 10% of the time series data and averages it
Core_DiffTemps(counter) = max(Core_Temp1.data)-
mean(Core_Temp1.data([round(length(Core_Temp1.data)*.9):end]));
TTMaxTemp(counter)=max(TT_Temp1.data);
TT_SteadyStateTemp(counter) = mean(TT_Temp1.data([round(length(TT_Temp1.data)*.9):end]));%This takes the
last 10% of the time series data and averages it
TT_DiffTemps(counter) = max(TT_Temp1.data)- mean(TT_Temp1.data([round(length(TT_Temp1.data)*.9):end]));
Core_Out_SteadyStatePress(counter)= mean(Core_Pressure1.data([round(length(Core_Pressure1.data)*.9):end]));
TT_SteadyStatePress(counter)= mean(TT_Pressure1.data([round(length(TT_Pressure1.data)*.9):end]));
Core_In_SteadyStatePress(counter)= mean(Pump_Pressure1.data([round(length(Pump_Pressure1.data)*.9):end]));

%Populating Max Temp Time Arrays
CoreMaxTempTime(counter)=Core_Temp1.time(max(find(Core_Temp1.data==max(Core_Temp1.data))));
TTMaxTempTime(counter)=TT_Temp1.time(max(find(TT_Temp1.data==max(TT_Temp1.data))));

%Concatinating Temp Timeseries
Core_Temp=vertcat(Core_Temp,Core_Temp1);
TT_Temp=vertcat(TT_Temp,TT_Temp1);

%Concatinating Temp Derivatives for Core
CoreTempDeriv=vertcat(CoreTempDeriv,Core_Temp_Deriv);
TTTempDeriv=vertcat(TTTempDeriv,TT_Temp_Deriv);

```

```

%Concatinating Percent Peak Power Timeseries
Peak_Power = vertcat(Peak_Power,PeakPower);

%Concatinating Reactivities
Tot_Reactivity = vertcat(Tot_Reactivity,Total_Reactivity);

%Concatinating Flows
TT_Pump_Flow = vertcat(TT_Pump_Flow,TTPumpFlow);

%Concatinating Pressure
TT_Pressure=vertcat(TT_Pressure,TT_Pressure1);
Pump_Pressure=vertcat(Pump_Pressure,Pump_Pressure1);
Core_Pressure=vertcat(Core_Pressure,Core_Pressure1);
%Printing out iteration
printout = ['***** JUST FINISHED Variable: ',num2str(i)];
disp(printout);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Plots only the Max temperatures for each iteration
h(1) = figure();
scatter(variable_array,CoreMaxTemp); title('Core Max Temps'); xlabel(reduced_Xlabel); ylabel('Temp(K)');
grid on;
line([0 variable_stop],[2900 2900],'color','r');
line([0 variable_stop],[2860 2860],'color','r','LineStyle','--');
h(2) = figure();
scatter(variable_array,TTMaxTemp); title('Tie Tube Max Temps'); xlabel(reduced_Xlabel); ylabel('Temp(K)');
grid on;
line([0 variable_stop],[1609 1609],'color','r');
line([0 variable_stop],[1569 1569],'color','r','LineStyle','--');

%Plots the Steady State Temps for each iteration
h(3) = figure();
scatter(variable_array,Core_SteadyStateTemp); title('Core Steady State Temps'); xlabel(reduced_Xlabel); ylabel('Temp(K)');
grid on;
line([0 variable_stop],[2900 2900],'color','r');
line([0 variable_stop],[2860 2860],'color','r','LineStyle','--');
h(4) = figure();
scatter(variable_array,TT_SteadyStateTemp); title('Tie Tube Steady State Temps'); xlabel(reduced_Xlabel); ylabel('Temp(K)');
grid on;
line([0 variable_stop],[1609 1609],'color','r');
line([0 variable_stop],[1569 1569],'color','r','LineStyle','--');

%Plots the difference between Steady-State and Max for each iteration
h(5) = figure();
scatter(variable_array,Core_DiffTemps); title('Core: Difference Between Steady-State and Maximum Temperature');
xlabel(reduced_Xlabel); ylabel('Temp(K)');
grid on;
h(6) = figure();
scatter(variable_array,TT_DiffTemps); title('Tie-Tubes: Difference Between Steady-State and Maximum Temperature');
xlabel(reduced_Xlabel); ylabel('Temp(K)');

```

```

grid on;

%Plots Max Power for each drum iteration
h(7) = figure();
scatter(variable_array,MaxPower); title('Average Peak Power'); xlabel(reduced_Xlabel); ylabel('Power (MWth)');
grid on;

%Plots the time that the maximum temperature was reached in the core & TT's
h(8) = figure();
scatter(variable_array,CoreMaxTempTime); title('Time to Max Core Temp'); xlabel(reduced_Xlabel); ylabel('Time(s)');
grid on;
h(9) = figure();
scatter(variable_array,TTMaxTempTime); title('Time to Max Tie Tube Temp'); xlabel(reduced_Xlabel); ylabel('Time(s)');
grid on;

%Max Core derivatives
h(10) = figure();
scatter(variable_array,MaxCoreDerivs); title('Maximum Core Temp Differential'); xlabel(reduced_Xlabel); ylabel('dT/dt(K/s)');
grid on;
line([0 variable_stop],[1000 1000],'color','r');

%Max Core derivatives
h(11) = figure();
scatter(variable_array,MaxTTDerivs); title('Maximum Tie-Tube Temp Differential'); xlabel(reduced_Xlabel);
ylabel('dT/dt(K/s)');
grid on;
line([0 variable_stop],[2000 2000],'color','r');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Plots all the temperature profile transients for each drum iteration
h(12) = figure();
CoreTempLegend=cell(1,((variable_stop-variable_start)/variable_step)+1); %empty cell array to be filled with DrumPos values
for plot legend
    for v=2:counter+1
        plot(Core_Temp(v));
        Core_Temp(v);
        CoreTempLegend{v-1} = strcat(num2str(variable_array(v-1)),profile_units);
    end
end
grid on;
title(strcat('Core Temperature Profile as a Function of',profile_title)); xlabel('Time(s)'); ylabel('Temp(K)');
legend(CoreTempLegend);
line([0 max(time)],[2900 2900],'color','r');
line([0 max(time)],[2860 2860],'color','r','LineStyle','--');

%Plots the temperature profiles for the core and the tie-tubes
h(13) = figure();
TTTempLegend=cell(1,((variable_stop-variable_start)/variable_step)+1);
for v=2:counter+1
    plot(TT_Temp(v));
    TTTempLegend{v-1} = strcat(num2str(variable_array(v-1)),profile_units);
end

```

```

        hold on
    end
    grid on;
    title(strcat('Tie Tube Temperature Profile as a Function of ',profile_title)); xlabel('Time(s)'); ylabel('Temp(K)');
    legend(TTTempLegend);
    line([0 max(time)],[1609 1609],'color','r');
    line([0 max(time)],[1569 1569],'color','r','LineStyle','--');

    %Plots the Percent Peak Power profiles
    h(14) = figure();
    PeakPowerLegend=cell(1,((variable_stop-variable_start)/variable_step)+1);
    for v=2:counter+1
        plot(Peak_Power(v));
        PeakPowerLegend{v-1} = strcat(num2str(variable_array(v-1)),profile_units);
        hold on
    end
    grid on;
    title(strcat('Peak Power Profile as a Function of ',profile_title)); xlabel('Time(s)'); ylabel('Power(MWth)');
    legend(PeakPowerLegend);

    %Plots the Total Reactivity Profile for each drum position
    h(15) = figure();
    TotalReactivityLegend=cell(1,((variable_stop-variable_start)/variable_step)+1);
    for v=2:counter+1
        plot(Tot_Reactivity(v));
        TotalReactivityLegend{v-1} = strcat(num2str(variable_array(v-1)),profile_units);
        hold on
    end
    grid on;
    title(strcat('Total Reactivity Profile as a Function of ',profile_title)); xlabel('Time(s)'); ylabel('Reactivity (cents)');
    legend(TotalReactivityLegend);

    %Plots the Pump Flow Profile for each drum position
    h(16) = figure();
    TTPumpFlowLegend=cell(1,((variable_stop-variable_start)/variable_step)+1);
    for v=2:counter+1
        plot(TT_Pump_Flow(v));
        TTPumpFlowLegend{v-1} = strcat(num2str(variable_array(v-1)),profile_units);
        hold on
    end
    grid on;
    title(strcat('Pump Flow Profiles as a Function of ',profile_title)); xlabel('Time(s)'); ylabel('Flow (kg/sec)');
    legend(TTPumpFlowLegend);

    %Plots the dT/dt for Core
    h(17) = figure();
    CoreTempDerivLegend=cell(1,((variable_stop-variable_start)/variable_step)+1);
    for v=2:counter+1
        plot(CoreTempDeriv(v));
        CoreTempDerivLegend{v-1} = strcat(num2str(variable_array(v-1)),profile_units);
        hold on;
    end
    grid on;

```

```

title(strcat('Core Temperature Derivatives as a Function of ',profile_title)); xlabel('Time(s)'); ylabel('dT/dt (K/sec)');
legend(CoreTempDerivLegend);
line([0 max(time)],[1000 1000],'color','r');

%Plots the dT/dt for Tie-Tubes
h(18) = figure();
TTTempDerivLegend=cell(1,((variable_stop-variable_start)/variable_step)+1);
for v=2:counter+1
    plot(TTTempDeriv(v));
    TTTempDerivLegend{v-1} = strcat(num2str(variable_array(v-1)),profile_units);
    hold on;
end
grid on;
title(strcat('Tie-Tubes Temperature Derivatives as a Function of ',profile_title)); xlabel('Time(s)'); ylabel('dT/dt (K/sec)');
legend(TTTempDerivLegend);
line([0 max(time)],[2000 2000],'color','r');

%Plots of Pressure
h(19)=figure();
TT_PressureLegend=cell(1,((variable_stop-variable_start)/variable_step)+1);
for v=2:counter+1
    plot(TT_Pressure(v));
    TT_PressureLegend{v-1} = strcat(num2str(variable_array(v-1)),profile_units);
    hold on;
end
grid on;
title(strcat('Tie-Tubes Pressure as a Function of ',profile_title)); xlabel('Time(s)'); ylabel('MPa');
legend(TT_PressureLegend);

h(20)=figure();
Pump_PressureLegend=cell(1,((variable_stop-variable_start)/variable_step)+1);
for v=2:counter+1
    plot(Pump_Pressure(v));
    Pump_PressureLegend{v-1} = strcat(num2str(variable_array(v-1)),profile_units);
    hold on;
end
grid on;
title(strcat('Pump Pressure as a Function of ',profile_title)); xlabel('Time(s)'); ylabel('MPa');
legend(Pump_PressureLegend);

h(20)=figure();
Core_PressureLegend=cell(1,((variable_stop-variable_start)/variable_step)+1);
for v=2:counter+1
    plot(Core_Pressure(v));
    Core_PressureLegend{v-1} = strcat(num2str(variable_array(v-1)),profile_units);
    hold on;
end
grid on;
title(strcat('Core Pressure as a Function of ',profile_title)); xlabel('Time(s)'); ylabel('MPa');
legend(Core_PressureLegend);

%Plots the Steady State Pressure for each iteration
h(21) = figure();

```

```

scatter(variable_array,Core_In_SteadyStatePress); title('Core Inlet Steady State Pressure'); xlabel(reduced_Xlabel); ylabel('Mpa');
grid on;
h(22) = figure();
scatter(variable_array,Core_Out_SteadyStatePress); title('Fuel Element Exit Steady State Pressure'); xlabel(reduced_Xlabel);
ylabel('Mpa');
grid on;
h(23) = figure();
scatter(variable_array,TT_SteadyStatePress); title('Tie Tube Exit Steady State Pressure'); xlabel(reduced_Xlabel); ylabel('Mpa');
grid on;

%prompts user to input path to folder to save files from simulation
%fileSave = inputdlg('Please review your plots. Would you like to save them? (Y=1, N=0)');

%if fileSave == 1
%     user_file_path = inputdlg('Please write the absolute path to the folder where you will save the data:');
%     filename = strcat(user_file_path,'Pump Plenum Volumes Simulation');
%     save(filename);
%     h = get(0,'children');
%     for i=1:length(h)
%         saveas(h(i), ['figure' num2str(lenth(h)+1-i)], 'fig');
%     end
%end

```