ABSTRACT

BHASKAR, SANDESH. A Framework for Multi-physics Modeling, Design Optimization and Uncertainty Quantification of Fast Spectrum Liquid Fuel Molten Salt Reactors. (Under the direction of Dr. Maria Avramova).

In the past, majority of tools for modeling and simulation of nuclear reactors were focused towards simulating light water reactors, due to the large operational experience. Only recently has there been a lot of development with regards to tools for modeling Liquid Fuel Molten Salt Reactors (LFMSRs). The analysis of LFMSRs during steady state, operational transients and accident scenarios require addressing unique reactor multi-physics challenges with coupling between thermal hydraulics, neutronics, inventory control and species distribution phenomena. To further the development of LFMSR concepts, researchers need advanced modeling approaches to obtain results with a good trade-off between accuracy and computational efficiency. The key is to identify and model the multi-physics phenomena that impose the potential challenges to modeling and simulation of these reactors.

This work utilizes the General Nuclear Field Operation and Manipulation (GeN-Foam) and System Analysis Module (SAM) to perform coupled thermal-hydraulics and neutronics calculation of the reactor. A framework is proposed as part of this study to perform modeling, design optimization and uncertainty quantification. The framework aims to establish a protocol for the studies and analyses of LFMSR which can later be expanded to other advanced reactor concepts too. Keeping this in mind, certain code developments followed by verification studies have been done to attain confidence with the codes. A new species transport model has been implemented in GeN-Foam along with the capability to model heat conduction to calculate the associated thermal stresses.

As part of the framework, the uncertainty quantification and sensitivity analysis tool Dakota was successfully coupled with GeN-Foam and SAM to perform uncertainty quantification studies. The uncertainties were propagated through the input design parameters and the output uncertainties were characterized using statistical analysis and Spearman rank correlation coefficients. Three analyses are performed (namely, scalar, functional and three-dimensional analyses) to understand the impact of input uncertainty propagation on temperature and velocity predictions. Preliminary three-dimensional reactor analysis showed that the thermal expansion coefficient, heat transfer coefficient and specific heat of the fuel salt are the crucial input parameters that influence the temperature and velocity predictions inside the LFMSR system.

The main outcome of this work was the development of a systematic approach to the design and analysis of LFMSR systems. To this purpose, a general framework was developed for performing multi-physics modeling, design optimization and uncertainty quantification under one umbrella thereby paving way for further studies to be expanded to account for different physical phenomena inside the reactor while using reduced order models to alleviate the computational cost of running complex multi-physics simulations.

A Framework for Multi-physics Modeling, Design Optimization and Uncertainty Quantification of Fast Spectrum Liquid Fuel Molten Salt Reactors

by
Sandesh Bhaskar

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Nuclear Engineering

Raleigh, North Carolina
2022

APPROVED BY:

Dr. Maria Avramova Committee Chair	Dr. Kostadin Ivanov
Dr. Jason Hou	Dr. Alexei Saveliev

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

GIF	Generation IV International Forum
SFR	Sodium-cooled fast reactor
LFR	Lead-cooled fast reactor
SCWR	Supercritical water-cooled reactor
VHTR	Very high temperature gas-cooled reactor
GCFR	Gas-cooled fast reactor
MSR	Molten salt reactor
LFMSR	Liquid Fuel Molten Salt Reactor
LWR	Light Water Reactor
MOOSE	Multi-physics Object-Oriented Simulation Environment
CFD	Computational Fluid Dynamics
EVOL	Euratom Evaluation and Viability of Liquid Fuel Fast Reactor System
OpenFOAM	Open-Source Field Operations And Manipulation
GeN-Foam	General Nuclear Field Operation and Manipulation
RELAP	Reactor Excursion and Leak Analysis Program
SAM	System Analysis Module
Dakota	Design and Analysis toolKit for Optimization and Terascale Applications
RANS	Reynolds-Averaged Navier-Stokes
FEM	Finite-element method
MSFR	Molten Salt Fast Reactor
VESIL	Versatile Experimental Salt Irradiation Loop
ATR	Advanced Test Reactor

CHAPTER 1

Introduction and Overview

In 2000, the United States Department of Energy identified the need for advanced nuclear technologies and convened the Generation IV International Forum (GIF) to expand the mission of nuclear energy and further the development of future nuclear systems. The six systems chosen for further study include:

- Sodium-cooled fast reactor (SFR),
- Lead-cooled fast reactor (LFR),
- Supercritical water-cooled reactor (SCWR),
- Very high temperature gas-cooled reactor (VHTR),
- Gas-cooled fast reactor (GCFR) and,
- Molten salt reactor (MSR).

The next generation advanced reactors seek to achieve: (1) optimal resource utilization and waste minimization; (2) high proliferation resistance; (3) safety and reliability through passive safety systems that eliminate the likelihood of fission product release during accident scenarios; (4) economic viability and affordability compared to other low-cost energy sources such as natural gas (Sabharwall et. al., 2013).

Although the proposed advanced reactor concepts offer a variety of advantages there are a few challenges that need to be addressed. The primary challenges common to all the advanced concepts include a need for advanced nuclear fuels and materials, and accurate simulation of the different reactor operation scenarios to capture the multi-physics feedbacks. The latter of these

requires coupled multi-physics codes and modules which meet the industry standards. The following set of features are considered crucial to advanced reactor designs (Sabharwall and Kim, 2011):

- Reliable shutdown and decay heat removal mechanisms;
- Fully passive safety systems that guarantee reactor integrity without any operator action;
- Negative temperature coefficient (required for all reactor designs);
- Minimize potential for severe accidents and reduce radioactive exposure.

This study focuses on Liquid Fuel Molten Salt Reactors (LFMSRs) in particular because they have the potential to reduce the cost of generating high-efficiency electricity as the overall cycle efficiency is around 11% higher than the conventional reactors (Altahhan et al., 2020). There is also no need to manufacture fuel pellets and rods as online refueling ensures that the reactor can continue operating. LFMSRs also provide high temperature process heat and minimize nuclear waste. These objectives can be realized due to the flexibility in the design of the LFMSR technology: different salt compositions and fuel forms provide temperature flexibility, neutron spectrum and power density (Sabharwall et al., 2013).

Since this is a liquid fuel system some of the delayed neutrons used in controlling the reactor power are emitted outside of the core. This will make the total effective delayed neutron fraction (β_{eff}) lower with a value of about 0.00446 reducing the reactor maneuverability compared to light water reactors (LWRs) that usually have (β_{eff}) value of about 0.0065, almost one and a half times higher.

Another significant issue from the materials point of view is that components such as pipes, valves and pumps will have to operate in a highly corrosive environment. Also, proper precautions

need to be put in place to regulate the flow of actinide rich fuel outside of the core such as better surveillance and routinely checking the fuel inventory (Altahhan et al., 2020).

To further the development of LFMSR concepts, researchers need advanced modelling approaches to obtain results with a reasonable computational efficiency while maintaining accuracy owing to the unique characteristic of the design such as liquefied fuel, complex core geometry, fast spectrum and moving precursors. The key is to identify and model these multi-physics phenomena that impose the potential challenges to modeling and simulation of the reactors.

Previous Works

In the 1960s, Molten Salt Reactor Experiment (MSRE) concept has been demonstrated by Oak Ridge National Lab (ORNL). MSRE was a thermal-neutron reactor system where the neutrons in the reactor core were moderated by graphite. Currently, both thermal and fast spectrum MSRs are being investigated as the latter proves to be structurally simpler due to the lack of moderating media inside the core. These reactors also feature improved actinide burning and are capable of extending fuel resources (Hu et al., 2017). Liquid-fueled concepts are popular among developers in the private industry due to their potential to improve fuel cycle metrics and passive safety (Betzler et al., 2019). The fluid nature of the fuel also adds to the flexibility in reactor core design, fuel fabrication and cycling. One unique feature of this concept is that there will be no accumulation of thermo-mechanical effects such as stress, creep and swelling as the fuel is liquid. In salt fueled MSRs, the fuel serves as the coolant too and plays a major role in heat removal during normal and accident scenarios.

In the past, majority of modeling tools were focused towards simulating LWRs, due to the large operational experience. In the past five years, there has been a lot of development with

regards to tools for modeling LFMSRs. The analysis of LFMSRs during steady state, operational transients and accident scenarios require addressing unique reactor physics challenges with coupling between thermal hydraulics, neutronics, inventory control and species distribution phenomena.

System thermal-hydraulics (STH) is one of the major disciplines essential for the design and operation of nuclear systems (D'Auria, 2012). System analysis codes have a history of being conservative in its analysis and they are mostly developed for light water reactors. To model advanced reactor concepts such as the molten salt reactors, the system codes need to be updated to accommodate the modeling these new technologies.

The RELAP5 code was originally developed for thermal-hydraulic analysis of LWRs and related experimental systems during loss-of-coolant accidents and operational transients. In the recent years, the code has been under constant development to simulate advanced reactor concepts. The ATHENA code is incorporated as a compile-time option in RELAP5-3D that generalizes the capability of the code to simulate systems that use working fluids other than water. Molten salt properties based on the correlations obtained from ORNL were implemented into RELAP5-3D/ATHENA (Davis, 2005). The simplified equation of state assumes that density is a function of temperature and pressure, based on correlations developed at ORNL and that the liquid heat capacity is constant. The vapor is assumed to have the same composition as the liquid and is assumed to be a perfect gas (Davis, 2005). This allows the code to perform steady-state and transient calculations for FHR. One major limitation of the simplified equation of state is that it cannot accurately represent the entire thermodynamic range of possible pressures and temperatures, which results in discontinuities near the critical point (Davis, 2005). In lieu of multiphysics simulations, point kinetics model considering the effect of fuel flow and thermal-

hydraulics component with heat source is included in RELAP5. Thermal-hydraulic and neutronic models are solved separately in RELAP5 code, while using semi-implicit or nearly-implicit method for pressure–velocity linked equations in thermal-hydraulics model, and Runge–Kutta numerical method for neutronics model. The validity of the extended models and numerical methods were verified against the MSRE experimental benchmark (Chengbin Shi, 2016). A similar study was done using TRACE and there were discrepancies observed with the results. The selected numerical solver for the current TRACE model was the forward Euler method (FEU), which is a first-order explicit method with lower computational load (He, 2016). The difference in the numerical solvers chosen for the study could be the reason for the discrepancies observed. Additionally, Carbajo et al. used RELAP5-3D to depict both the MSRE hydraulic mockup and primary loop (Carbajo, 2017). This work showed reasonable agreement between code and experiments for coolant velocity and pressure drop of the hydraulic mockup. It also showed similar agreement between code and historical calculations for coolant temperatures of the primary loop MSRE (Carbajo, 2017).

It is important to note that, coupled point kinetics and thermal-hydraulics models can yield inaccurate solutions when it comes to asymmetric transients. Transients such as single pump failure can be dealt by spatial kinetics tools in a more efficient way. One of the major limitations of legacy codes, such as TRACE and RELAP-5, is that they still rely on explicit data exchanges and lack robustness. Unfortunately, this type of loose coupling is a less desirable method as it leads to poor accuracy when different physics are strongly coupled, and a fast transient is present. Also, for traditional system codes, it is difficult to accurately model the wave oscillation or the sudden disturbance of the system without any numerical instability and numerical diffusion concerns due to their first-order approximations of the differential equations in both time and space (Rui Hu,

(2017), A Fully-implicit high-order system thermal hydraulics model for advanced non-LWR safety analyses).

RELAP-7 code being developed at Idaho National Laboratory (INL) is set to take advantage of advancements in computer architecture, software design, numerical integration methods, and physical models. The end result being a system code that retains and improves upon RELAP5's capability and extends the analysis capability for all reactor system simulation scenarios. RELAP-7 is written with object-oriented programming language C++ and uses MOOSE (Multi-physics Object-Oriented Simulation Environment) development environment (D. Gaston, 2009). MOOSE is a HPC development and runtime framework for solving computational engineering problems in a well-planned, managed, and coordinated way. By using open-source software packages, such as PETSc (Balay et al., 2017) (a nonlinear solver developed at Argonne National Laboratory) and LibMesh (Kirk et al., 2006) (a Finite Element Analysis package developed at University of Texas), MOOSE reduces the expense and time required to develop new applications. One important advantage to building the code on MOOSE framework is that there are different MOOSE based applications ranging from 3-D transient neutron transport, 3-D transient fuel performance analysis, to long-term material aging which can be integrated to the code. Multiphysics and multi-dimensional analysis capabilities, such as radiation transport and fuel performance, can be obtained by coupling RELAP-7 and other MOOSE-based applications through MOOSE (Hongbin Zhang, 2018).

System analysis module (SAM) is another advanced reactor system analysis code like RELAP-7 that focuses on the modeling of the components and systems that represent typical features of advanced reactor concepts. These advanced concepts are distinguished from light water reactors in their use of single-phase, low-pressure, high-temperature, and low Prandtl number

coolants. This simple yet fundamental change has significant impacts on core and plant design, the types of materials used, component design and operation, fuel behavior, and the significance of the fundamental physics in play during transient plant simulations. SAM uses latest numerical techniques to shorten execution time and increase the accuracy of results. Like RELAP-7, SAM is also built on the MOOSE framework, which interfaces with libMesh and PETSc to provide the underlying geometry (mesh I/O) and numerical capabilities (finite element library and solvers). It is aimed to solve the tightly coupled physical phenomena including fission reaction, heat transfer, fluid dynamics, and thermal-mechanical response in the structures, systems and components in a fully coupled fashion. Reduced-order modeling approaches implemented in SAM also facilitate rapid turn-around for design and safety optimization studies (Hu, 2017). This unique modeling feature distinguishes SAM from its counterpart RELAP-7. A holistic 1-D MSRE system of both hydraulic mockup and molten salt primary loop was modeled in SAM. The results from SAM aligned well with the measured data from the experiment and the RELAP5-3D results (Carbajo, 2017) for the total system head loss and core and heat exchanger axial temperature profiles. Moltres is an open-source tool for simulating MSRs and it solves arbitrary-group neutron diffusion, temperature, and precursor governing equations on a single mesh too. One to three-dimensional geometry can be solved and can be deployed on an arbitrary number of processing units. Moltres is built using the MOOSE (Gaston et al., 2015) framework and it includes physics kernels and boundary conditions for solving for neutron fluxes, temperature, and precursor concentrations. Moltres results also compared favorably with the actual design calculations of the MSRE. However, its application towards transient calculations are still limited due to its simplified modeling of heat exchangers and the lack of Navier-Stokes module to accurately generate a velocity profile.

Advanced system codes can they solve 0D, 1D, and 3D geometries on fairly coarse computational meshes over which the problem variables are averaged. Also, viscous shear stresses or turbulence effects are not included in their formulations. As a result, the codes cannot accurately model flow behavior that are affected by turbulence which includes flow regions that exhibit recirculation zones. However, it is very important to model these turbulence effects as the Reynolds number can reach very high values in the present MSFR systems. One solution to this problem is coupling to CFD codes which is beyond the scope of this work. The development of 1-D system codes have been based on data collected from several experiments over time and generating empirical models which are then plugged into the 1-D codes. Since the fluid flow and temperature profile aren't available, the empirical models cannot be used to design new generation of advanced nuclear reactors. To overcome the limitations of 1-D codes and design new generation of nuclear reactors, high fidelity and reliable numerical tools which can provide detailed 2-D/3-D distribution of thermal-hydraulic and neutronic parameters is required (B. Joshi, 2019). Due to recent advancements in computational power, the detailed 2-D/3-D velocity and temperature patterns (thermal-hydraulic parameters) can be obtained using 3-D Computational Fluid Dynamics (CFD) codes where transient conservation equations of mass, momentum and energy are solved for different species (B. Joshi, 2019).

In an effort to step beyond the use of legacy codes and explicit data exchange interfaces, a new modern multi-physics platform is needed. There are several approaches to couple the neutronics to computational fluid dynamics depending on how the power density and thermophysical data are transferred between the codes used (Gregory et al., 2019). The slowest approach is to write data to a file in a format that the other code can read (Yan et al., 2011), while faster methods involve using message passing protocols controlled by a third program to

communicate the data (Grahn et al., 2017). Mahadevan coupled the neutronics and fluid solvers by exchanging data and converging both physics at each inner time loop (Mahadevan et al., 2014). Alternative approaches implement a deterministic method into the fluid dynamics solver but solve the neutronics and fluid dynamics on separate meshes (Laureau et al., 2015). Aufiero (Aufiero et al., 2014) and Cervi (Cervi et al., 2019) used the same meshes to solve both the fluid dynamics and the neutronics in a Picard iteration.

Most of the recent research efforts have been focused on studying the LFMSR, developing and verifying multi-physics codes for the LFMSR simulation (Dolan, 2017). The European project SAMOFAR coordinates the research efforts on the Molten Salt Fast Reactor (MSFR) concept in Europe. The MSFR is a fast-spectrum breeder reactor with a large negative power coefficient. The fast spectrum allows the reduction of the reprocessing requirements and enables a better reactor breeding ratio (Mathieu et al., 2009). The reference MSFR concept is a 3000 MWth reactor with three different circuits: the fuel circuit, the intermediate circuit, and the power conversion system. Figure 1 illustrates the schematic layout of the MSFR. The components of the primary fuel circuit are the fuel salt, the core cavity, the heat exchangers, the pumps, the inlet and outlet pipes, and the salt-bubble separators. The initial composition of the MSFR is, 77.5 mol% of LiF, and a 22.5% mol mixture of AcF3 and AcF4 (Ac indicates actinides, mostly Th^{232} or U^{233}). The total volume of fuel salt is about 18 m^3 in the fuel loop and the average core salt temperature is approximately 725 °C. The fuel salt upon exiting the core cavity from the top, flows into 16 different loops each containing a pump and heat exchangers located around the core. The fuel salt circulates the loop in about 3-4 seconds (Rubiolo et al., 2017).

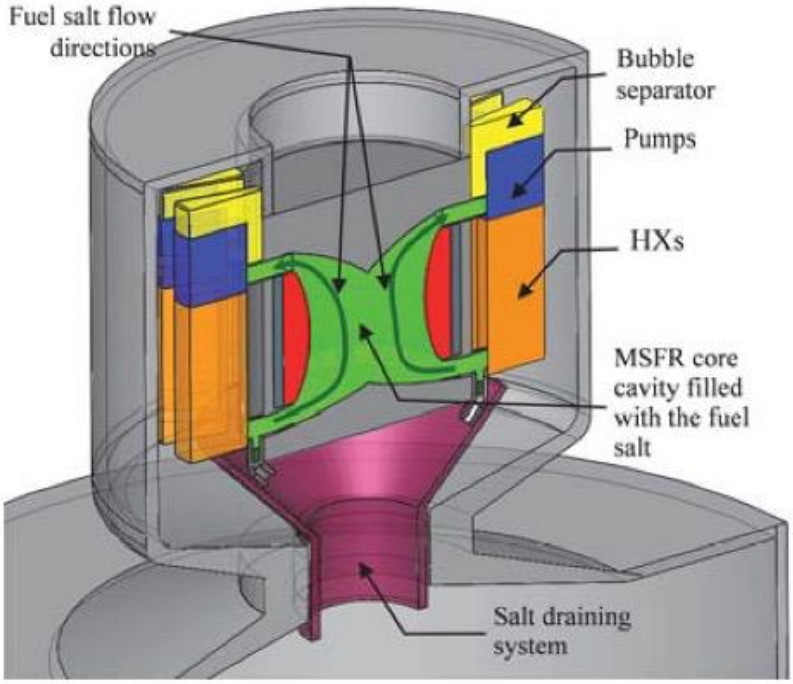


Figure 1. Layout of the Molten Salt Fast Reactor (MSFR) fuel circuit (Rubiolo et al., 2017)

The MSFR design has some potential advantages as discussed but also poses new challenges in terms of reactor design and safety. The presence of several phenomena inherent to molten salts that do not exist in other reactor types requires the development of better modeling tools and new methodologies to perform the safety evaluation. One such research project is the Euratom Evaluation and Viability of Liquid Fuel Fast Reactor System Project (EVOL). In their recent work, a cylindrical core with the diameter equal to the height, surrounded by steel reflectors axially, and a fertile blanket, a boron carbide layer, and a reflector radially is studied. The fuel salt used in their study is a fluoride-based salt and is a mixture of lithium fluoride, thorium fluoride and actinide fluoride salts. Under the EVOL project umbrella, several codes were developed and most of them were based on the Open-Source Field Operations And Manipulation (OpenFOAM) C++ code library (Weller et al., 1998) for solving coupled partial differential equations. Examples of OpenFOAM applications in this regard can be found in (Brovchenko, et al., June 2013) (Cervi,

et al., 2019) (Aufiero, et al., 2014) (Aufiero, et al., 2014). GeN-Foam is one such code that is based on the OpenFOAM C++ library that uses coarse mesh porous media approach instead of standard Reynolds Averaged Navier Stokes (RANS) turbulence model to solve for the various physics involved in a coupled fashion (Fiorina et. al., 2015). The LFMSR can be modeled using GeN-Foam code to avoid a detailed RANS model of components such as heat exchangers and pumps. The flow and heat transfer in these components can be modeled using a porous k-epsilon approach thereby saving a lot of computational expense and time.

Apart from open-source codes, commercial codes have also been used to understand the MSFR system better. A thermal–hydraulics design study of the MSFR was performed using the CFD code FLUENT (general-purpose computational fluid dynamics software package that can be used to model flow, turbulence, heat transfer, and chemical reactions) (H. Rouch, 2014). The meshes were built with the mesher GAMBIT. The MSFR simulations revealed that the flow inside the core cavity mixed well thereby minimizing high temperatures within the salt and the core walls. Flow mixing is important as it leads to improved uniformity in the salt composition, distribution of inert gas bubbles used in fission product extraction and reduced temperature gradient of the salt that helps in decreasing the thermal stress induced on the core wall structures (H. Rouch, 2014). However, the sensitivity of temperature distribution to reactor operation, salt thermophysical properties and heat source are not investigated here. Also, since MSFR is a tightly coupled system, coupling CFD with neutronic simulations might yield more accurate results.

In the current designs of MSR, molten fuel salt needs to be pumped continuously in the primary circuit in order to promote mixing of molten fuel salt inside reactor core. The temperature and velocity profiles followed by their corresponding magnitudes at various locations were studied using FLUENT and it was found that the natural circulation flow problem is transient in nature,

but at certain parameter, there exist condition for steady state after long iteration run (A.M. Pauzi, 2015).. Most of flow region were turbulent in nature and the standard k-ω turbulence model proved to yield the best results in terms of residual convergence (A.M. Pauzi, 2015).

Motivation for Proposed Research

Though there are optimized core geometry designs available, rigorous studies have not yet been performed in this regard. The optimized core geometry available in literature (Rouch et al., 2014) is only based on one factor i.e., hot spots inside the core. These hot spots occur mainly due to the formation of recirculation and stagnation zones which increase the salt temperature considerably. This undesirable phenomenon needs to be avoided or minimized as it leads to higher temperature gradients inside the core. In reality, there are also other factors that need to be considered when optimizing the design. Extensive studies need to be performed to optimize the capital cost and performance of major systems. These analyses must consider the multi-physics design impacts on plant economic performance such as pumping requirements, sizing of heat exchangers, sizing of system pressure boundaries and coolant inventory requirements (Holcomb et al., 2013).

The design adopted for this work is based on the EVOL project in Europe and is discussed in detail in Chapter 4. Once the preliminary reactor design has been chosen, phenomena identification alone will not be sufficient and must be followed with studies exploring their impact on the reactor system during both steady state and transient scenarios. This can be accomplished through CFD and system level modeling. The reactor geometry needs to be studied in detail, meaning detailed studies need to be performed in quantifying the uncertainties related to the design and also perform sensitivity analysis to identify the input variables that have an impact on the output's uncertainty (qualitative). These studies can subsequently help in reducing the dimensions

of the input/output space. Reduced order modeling and building surrogate models is a relatively new concept in nuclear industry. Recent research efforts have focused on building reduced order models for two-dimensional (2-D) cavity case (Alsayyari, et al., 2020) and 2-D LFMSR models (German, et al., 2019) (Ronco, et al., 2020) assuming molten salt properties but there is still a long way to go. It is important to explore the different techniques available in the literature and rank them based on their accuracy with respect to full order models and their corresponding capability to reduce the computational time for solving high fidelity 2-D/3-D problems. This ultimately enables the users to perform high-fidelity uncertainty quantification and sensitivity analysis for steady state and transient scenarios at a significantly lower computational cost.

Objectives

The objective of this work is to develop a framework for multi-physics modeling, design optimization and uncertainty propagation of fast spectrum liquid fuel molten salt reactor. In this study, GeN-Foam is used to model the LFMSR using a coarse mesh porous media approach in places where the detailed interactions can be approximated (such as heat exchanger and pump regions) thereby reducing the computational expense and time significantly. GeN-Foam is based on the OpenFOAM C++ library, capable of performing coupled fine/coarse mesh thermal-hydraulics calculations, multi-group neutron diffusion calculation and thermo-mechanics calculation using both structured and unstructured meshes. The governing equations implemented in GeN-Foam and the theory supporting the statistical analysis is given in Chapter 2. A brief introduction of the codes to be used in this work i.e. SAM and Design and Analysis toolKit for Optimization and Terascale Applications (Dakota) are given in Chapter 2 as well.

To gain confidence with the newly developed code, GeN-Foam is also verified against the CFD code Fluent (ANSYS Fluent Theory Guide, 2013) for both water and salt properties for two

preliminary models: (1) LFMSR core and (2) LFMSR primary loop model respectively and is discussed in detail in Chapter 3 along with some additional code developments that are pointed out. Chapter 3 also discusses the code capabilities of SAM and the verification studies performed which sets the ground for developing system level models of LFMSR. GeN-Foam is coupled to Dakota to perform statistical analysis such as design optimization, uncertainty propagation and sensitivity analysis. The GeN-Foam and Dakota coupling framework is explained in detail in Chapter 3. Preliminary uncertainty propagation and sensitivity analysis results showcasing the successful coupling of Dakota and GeN-Foam is shown in Chapter 3.

The newly developed core design followed by design optimization studies for the reactor core, heat exchanger and pump are discussed in detail in Chapter 4. The modeling approach for the preliminary 2-D and 1-D primary loop models in GeN-Foam and SAM are given in Chapter 4.

Based on the preliminary results available from Chapter 4, the 3-D LFMSR primary loop model is developed, and the uncertainty propagation and sensitivity analysis are performed, and the results are discussed in detail in Chapter 5, followed by conclusions and future perspectives in Chapter 6.

CHAPTER 2

Theory and Methods of Computer Codes Utilized

Performing physics calculations that will provide accurate and efficient results is always a challenge. GeN-Foam is a multi-physics code that can solve for three different physics both individually and in a coupled fashion. In this work, to study the LFMSRs only neutronics and thermal hydraulics are modeled. Section 2.1 goes over the governing equations for each sub-solver and accounts for all necessary source and sink terms along with the coupling mechanism. Since SAM will be part of the framework, a brief introduction of the code is given in Section 2.2 and the governing physics models are discussed as well. As part of physics modeling, it is also important to characterize and reduce the uncertainties associated with the computational model. An uncertainty propagation framework that comprises of different statistical tools that can propagate the input uncertainties through the codes and analyse the quantities of interest using appropriate statistical quantities to model the LFMSR is introduced in Section 2.3. To this purpose, Section 2.4 introduces Dakota, a code that is being developed and maintained by Sandia National Labs. Dakota will be used to perform uncertainty propagation, parameter estimation, and sensitivity analysis studies by coupling with the multi-physics code GeN-Foam.

2.1 Governing Theory and Assumptions in GeN-Foam

GeN-Foam is built on three sub-solvers that account for three different physics to be modeled in LFMSR namely, neutronics, thermal-hydraulics and thermal-mechanics. The thermal-hydraulics sub-solver is based on the OpenFOAM solver, *buoyantPimplefoam* and the code iterates on the conservation equations first. Post that, a thermal-mechanics sub-solver is used to deform and move the mesh if thermal-mechanics is to be solved for. Finally, both the sub-solvers provide

feedback to the neutronics sub-solver which is based on the multi-group neutron diffusion equation or SP3 equations (Fiorina, et al., 2017).

All three sub-solvers use the finite volume method for discretization of the partial differential equations and can iterate on both structured and unstructured meshes thereby adding to the versatility of the code in modeling any shape or domain. A backward Euler scheme is used for time dependent solutions. In the following sub-sections, the different sub-solvers and their governing equations are summarized.

2.1.1 Thermal-hydraulics sub-solver

Reynolds-Averaged Navier-Stokes (RANS) modeling can be computationally expensive when modeling entire primary loop/complex geometries. The thermal-hydraulics sub solver in GeN-Foam can solve for a detailed RANS modeling approach and a porous medium approach thanks to the dedicated models implemented to simulate the interaction between fluid and sub-scale structures. So, in the case of clear fluid regions, the solver only solves for RANS equations. By using the porous media approach in which one set of discretized equations is used, one can alleviate the numerical stability and diffusion problems that can appear if different set of equations were used for different zones. The porosity equations used in this research are given by the modified RANS model as shown in Equations 1-3,

$$\frac{\partial \gamma \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u_D}) = 0 \tag{1}$$

$$\frac{\partial \rho \boldsymbol{u_D}}{\partial t} + \nabla \cdot (\rho \boldsymbol{u_D} \otimes \boldsymbol{u}) = \nabla \cdot (\mu_T \nabla \boldsymbol{u}) - \gamma \nabla p + \gamma \boldsymbol{F_g} + \gamma \boldsymbol{F_{ss}} \tag{2}$$

$$\frac{\partial \gamma \rho e}{\partial t} + \nabla \cdot \left(\boldsymbol{u_D}(\rho e + p)\right) = \gamma \nabla \cdot (k_T \nabla \mathrm{T}) + \boldsymbol{F_{ss}}.\boldsymbol{u_D} + \gamma Q_{ss} + \gamma Q_F \tag{3}$$

Here, (γ) is the porosity of the medium, (ρ) is the medium fluid density, $(\boldsymbol{u_D})$ is the Darcy's velocity of the porous medium (related to the fluid velocity by $\boldsymbol{u_D} = \gamma \boldsymbol{u}$), (μ_T) is the medium turbulent dynamic viscosity, (p) is the fluid pressure, $(\boldsymbol{F_g})$ is the buoyant force exerted on a volume of the fluid in the medium, $(\boldsymbol{F_{ss}})$ is the drag force exerted by the sub-scale structure of the porous medium on the fluid, (e) is the total internal energy of the fluid, (k_T) is the fluid thermal conductivity, (Q_{ss}) is the heat transfer between the fluid and the sub-scale structure, (Q_F) is the heat source in the molten salt due to fission (i.e. volumetric power due to neutrons fission reaction in the fluid). It is evident that the above set of equations represent the conservation equations if not for the porosity term γ. In the case of a clear fluid, $\gamma = 1$, and neglecting the terms containing $(\boldsymbol{F_{ss}})$ and (Q_{ss}) in equations (2-3), one will obtain the RANS equations (Fiorina et al., 2017) (Patankar, 1980). Note that the (Q_F) only appears for LFMSR due to the fluid being a liquid fissionable nuclear fuel. Solution of equations (1-3) is achieved through a merged PISO-SIMPLE (named PIMPLE) pressure-based algorithm for fluid flows (Weller et al., 1998).

The terms $(\boldsymbol{F_{ss}})$ and $(\dot{Q}_{ss})$ are given by Equation 4 and Equation 5,

$$F_{ss,i} = \frac{f_{D,i}\ \rho\ u_{D,i}^2}{2D_h\ \gamma^2} \tag{4}$$

$$Q_{ss} = A_V h (T_{ss} - T) \tag{5}$$

The (i) represents the number of principal directions of a porous medium (for e.g. the directions of the viscous and inertial resistances), $(f_{D,i})$ is the Darcy's friction factor, (D_h) is the Hydraulic diameter, (A_V) is the subscale structure surface area to volume ratio, and (h_{eff}) is the effective heat transfer coefficient, and (T_{ss}) is the sub-scale structure temperature. The friction factor is given by Equation 6,

$$f_{D,i} = A_{f_{D,i}} Re^{B_{f_{D,i}}} \tag{6}$$

The constants (D_h), $\left(A_{f_{D,i}}\right)$ and $\left(B_{f_{D,i}}\right)$ are user defined depending on the problem in hand, while the Reynolds number (Re) is calculated using the velocity magnitude and the hydraulic diameter. The heat transfer coefficient (h) is calculated based on a Nusselt-number (Nu) correlation given by Equation 7,

$$Nu_i = A_{Nu,i} Re^{B_{Nu,i}} \mathrm{Pr}^{C_{Nu,i}} + D_{Nu,i} \tag{7}$$

Here $\left(D_{Nu,i}\right)$, $\left(A_{Nu,i}\right)$, $\left(B_{Nu,i}\right)$, and $\left(C_{Nu,i}\right)$ are user defined on a case by case basis, and (Pr) is the Prandtl number. The sub-scale structure temperature using the porous medium approach is given by the conduction equation (T_{ss}) in Equation 8 (Altahhan et al., 2020) (Fiorina et al., 2015) (with $(\gamma_{ss} = 1 - \gamma)$),

$$\rho_{ss} c_{p,ss} \frac{\partial T_{ss}}{\partial t} = \nabla \cdot (\gamma_{ss} \boldsymbol{k_{ss}} \nabla T) + A_V h (T - T_{ss}) \tag{8}$$

To allow for the possibility of modeling heat exchangers, where the heat is exchanged with an external fluid at fixed temperature, equation (5) is modified as:

$$Q_{hx} = A_{hx} h_{eff} (T_{ext} - T) \tag{9}$$

Where the effective heat transfer coefficient is calculated as the harmonic mean of (h) and a user selected external heat transfer coefficient (h_{ext}):

$$h_{eff} = \frac{2 h h_{ext}}{h + h_{ext}} \tag{10}$$

The terms (k_T) and (μ_T) in Equations (2-3) are modeled using the standard $(k - \epsilon)$ model in the clear fluid zones. In porous zones, the interaction with sub-scale structure typically dominates the pressure drops and heat transfer, so an accurate modeling of (μ_T) and (k_T) becomes inessential, while it becomes relatively important to predict a correct value of (k) and (ϵ) at the outlet of the porous zone i.e., entrance to a clear fluid region where standard $(k - \epsilon)$ model is used

(Launder et al., 1974). Hence, a user defined convergence value (k_0, ϵ_0) is used such that (k), and (ϵ) can converge to it. This is made sure by solving Equation 11 for (ϵ) and (k) respectively,

$$\rho \frac{\partial \epsilon}{\partial t} + \nabla \cdot (\rho \boldsymbol{u_D} \epsilon) = \rho \lambda_\epsilon (\epsilon_0 - \epsilon), \qquad \rho \frac{\partial k}{\partial t} + \nabla \cdot (\rho \boldsymbol{u_D} k) = \rho \lambda_k (k_0 - k) \tag{11}$$

This works provided some information is available from numerical or experimental work to evaluate (k_0) and (ϵ_0). An equilibrium value for (k) and (ϵ) can be assumed or calculated by correlations involving the Reynold's number (Re), turbulence intensity (I) and the turbulence length (l), as it is usually done in a CFD analysis (Altahhan et al., 2020).

2.1.2 Neutronics sub-solver

The neutronics sub-solver is based on the multigroup diffusion equation as shown in equation (12), with the delayed neutrons precursors concentration given by equation (13). The second term in equation (13) is the convection term, to account for precursors drift in the LFMSR. The third term in equation (13) contains the effect of turbulent diffusion on the delayed neutrons precursors concentration is not used in this research due to the lack of LFMSR data for the turbulent diffusion coefficient $\left(\text{i.e. } (\nu_t / Sc_t)\right)$ (Altahhan et al., 2020). The number of energy groups and the precursors groups are arbitrarily chosen and are inputted by the user. The neutronics sub-solver implementation has been verified against several solid-fuel nuclear reactor-based benchmarks and similarly for LFMSR according to (Fiorina et al., 2015) (Fiorina et al., 2017) (Fiorina et al., 2016) (Altahhan et al., 2020). The multigroup-neutron diffusion equation is given by Equation 12.

$$\frac{1}{\mathrm{v_g}} \frac{\partial \phi_\mathrm{g}}{\partial \mathrm{t}} = \nabla \cdot \mathrm{D_g} \nabla \phi_\mathrm{g} - \Sigma_{\mathrm{r_g}} \phi_\mathrm{g} + \sum_{\mathrm{g'} \neq \mathrm{g}} \Sigma_{\mathrm{g'} \to \mathrm{g}} \phi_{\mathrm{g'}} + \chi_{\mathrm{p,g}} \left(1 - \beta_{\mathrm{eff,t}}\right) \sum_{\mathrm{g'}=1} \nu \Sigma_{\mathrm{f,g'}} \phi_{\mathrm{g'}} + \sum_{\mathrm{k}} \chi^{\mathrm{k}}_{\mathrm{d,g}} \lambda_\mathrm{k} \mathrm{C_k} \tag{12}$$

And the delayed neutrons precursors equation is given by Equation 13.

$$\frac{\partial C_k}{\partial t} + \nabla \cdot (u_D C_k) + \nabla \cdot \frac{\nu_t}{Sc_t} \nabla C_k = \frac{\beta_{eff,k} \sum_j \nu \Sigma_{f,j} \phi_j}{k_{eff}} - \lambda_k C_k \quad (13)$$

Here, (v_g) is the neutron velocity in group (g), (ϕ) is the scalar neutron flux, (D_g) is the neutron diffusion coefficient of group (g), (Σ_{r_g}) is the removal cross section for group (g), (χ_{p_g}, χ_{d_g}) are the fission spectra of prompt and delayed neutrons, (β_{eff_t}) is the total effective delayed neutrons fraction, (ν) is the average number of neutrons emitted per fission, (Σ_f) is the fission reaction cross section for a neutron in group (g'), $(\Sigma_{g' \to g})$ is the scattering cross section for a neutron with energy in initial energy group (g'), (λ_k) is the delayed neutrons precursors decay constant of group (k), (C_k) is the delayed neutrons precursors concentration for group (k), (ν_t) is the turbulent kinematic viscosity, and (Sc_t) is the turbulent Schmidt number.

The multi-purpose 3-D continuous-energy Monte Carlo particle transport code Serpent (Leppanen et al., 2015) is used to generate the parameterized cross sections needed for the simulation, and a MATLAB script converts the cross sections into a format readable by GeN-Foam. GeN-Foam uses an interpolation method to calculate the cross sections between different physical states. For this reason, two points per parametrized quantity (i.e. two sets of cross-sections) are needed, the reference point and the perturbed state. To obtain the local cross-sections, the sub-solver interpolates between the nominal and perturbed cross-section using cell-wise information. In the case of fast spectrum neutronics fuel temperature feedback, the interpolation is done using a log curve between the two respective points (Fiorina et al., 2015) (Aufiero et al., 2013). Linear interpolation is used for other physical quantities such as the salt density (Altahhan et al., 2020).

2.1.3 Coupling Mechanism

Figure 2 illustrates the coupling methodology in GeN-Foam as well as the sequence in which the sub-solvers are invoked. The density equation is the mass conservation equation (cf. equation 1), while the momentum equation calculates the velocity predictor and corrector steps (cf. equation 2). The Poisson equation solves for the pressure in the system and is a result of equations (1-2), and this is done within GeN-Foam (Altahhan et al., 2020). The energy equation mentioned in the figure is given by equation 3 and is solved after achieving the velocity field for outer convergence.

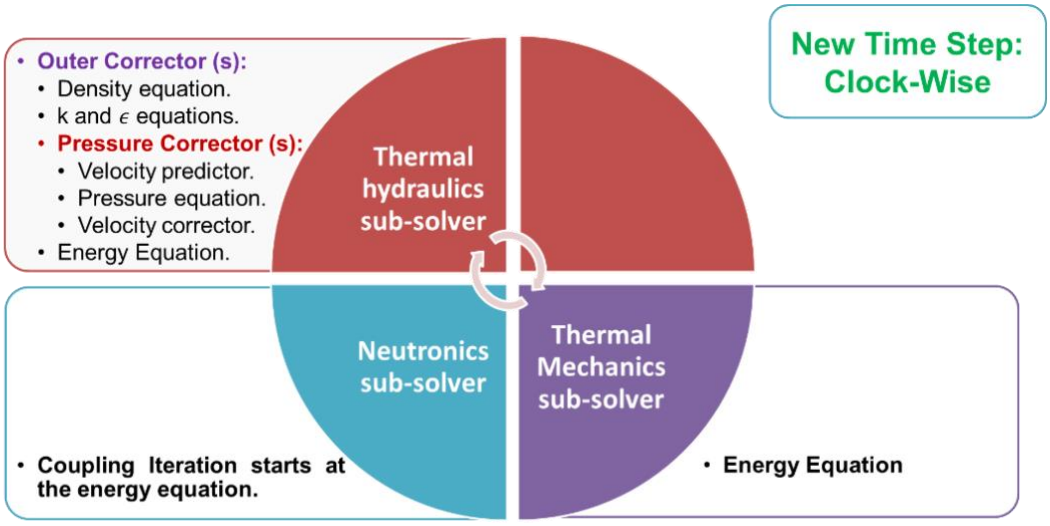


Figure 2. GeN-Foam sub-solvers and coupling scheme (Altahhan et al., 2020)

Once the velocity and temperature fields are calculated, GeN-Foam calls for the thermal mechanics and the neutronics sub-solvers sequentially. Upon attaining a residual tolerance, GeN-Foam moves to the next time step, and is based on the Courant number condition for the fluid and a user selected maximum allowed power variation. If one of the above fails, it solves for the energy equation in the thermal-hydraulics sub-solver. Finally, the user has the flexibility to choose between the sub-solvers and the physics to be solved for while leaving others unaltered. In

summary, the neutronics module computes the power distribution (by using cross sections interpolated based on the fuel temperature/density predicted by the thermal hydraulics module) and provides it to the thermal hydraulics module for updating its results. The thermal hydraulics module transfers the fuel salt velocity field to the neutronics module as it is essential for determining the delayed neutron precursors distribution (Altahhan et al., 2020). The solution variables are mapped between different meshes consistently after each time-step using a cell-volume weighted algorithm found available through OpenFOAM (Fiorina et al., 2015).

2.2 System Analysis Module (SAM)

SAM is a Finite-element method (FEM) based advanced system code that is currently being developed at Argonne National Laboratory (Hu et al., 2015). The code is built on the object oriented application framework MOOSE (Gaston, 2009) and uses open source software packages, such as PETSc (Balay et al., 2017) and LibMesh (Kirk et al., 2006) to significantly reduce computational time and expense required to develop new applications.

The code uses a component-based modeling approach to provide a user-friendly interface and the physics is integrated/associated with each individual component. The fundamental physics models implemented in SAM enables the code to solve for fluid flow, heat transfer (conduction, convection, and radiation), power production in the core, reactivity feedback and decay heat. The fluid flow is governed by the conservation equations and can currently model only 1-D single phase incompressible flows with thermal expansion. The 1-D/2-D heat structure model allows the code to predict temperature distributions in both solid and fluid components. Finally, the point kinetics model implemented in SAM, accounts for the neutronics part of the code. The flexible

coupling interface equipped in the code allows it to be coupled with higher fidelity codes. The code structure is shown in Figure 3 (Hu, 2017).

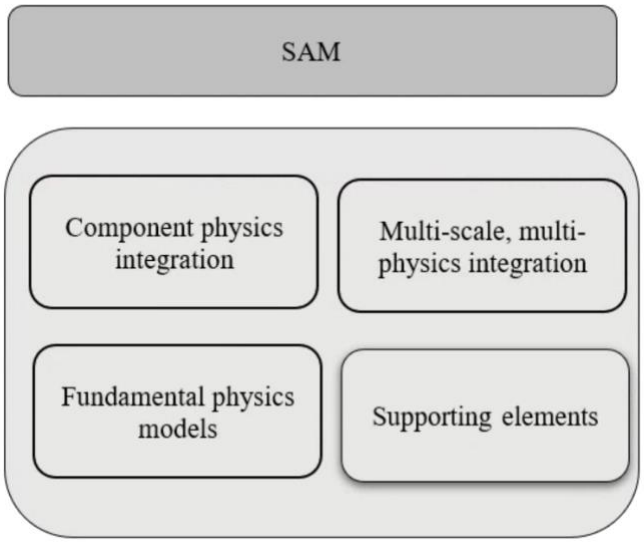


Figure 3. SAM code structure

2.3 Uncertainty Quantification Framework

In the case of LFMSR modeling, strong coupling exists between the different physics, making the uncertainty analysis more challenging. Dependencies and interactions between a large number of input and a large number of non-linear input-output relations add to the complexity of the uncertainty study. In such cases, it is important to establish an uncertainty quantification framework that comprises of different statistical tools that can propagate the input uncertainties through the codes and analyse the quantities of interest using appropriate statistical quantities (Delipei, 2019). Figure 4 presents a general framework for uncertainty quantification.

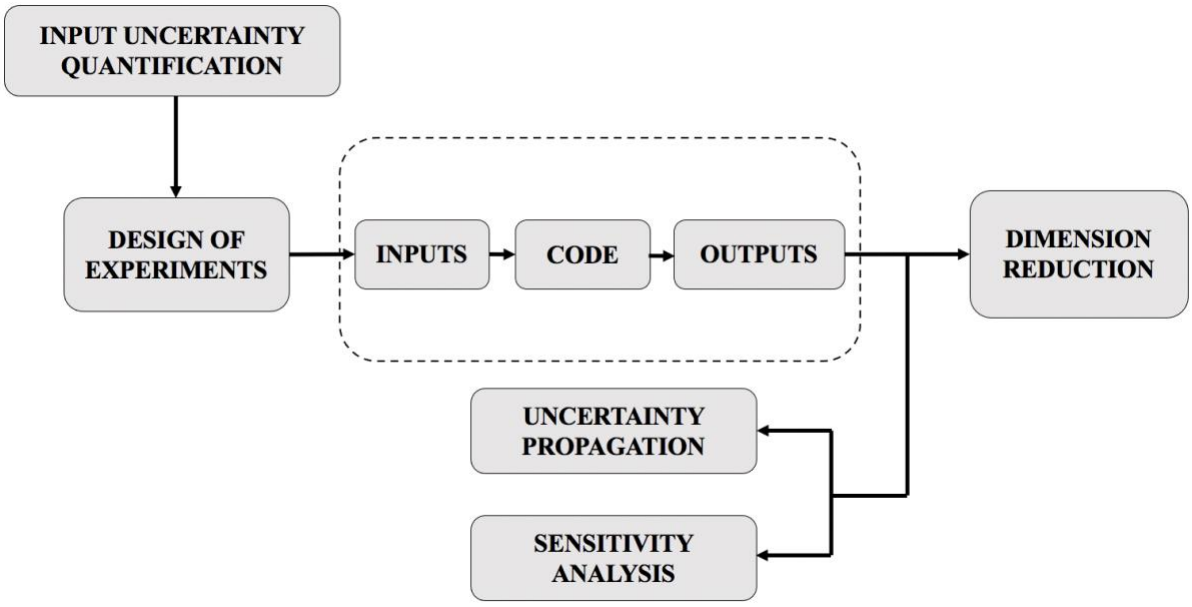


Figure 4. General framework for uncertainty quantification (Delipei, 2019).

As part of the UQ framework, the first step is to identify and quantify the input uncertainties. The input uncertainty is classified into two categories. Aleatoric uncertainty, represents variables that change each time a certain experiment or action is performed under the same testing conditions. For example, variability in weather conditions. Whereas epistemic uncertainty is the uncertainty that stems from lack of knowledge. For example, the lack of experimental data to characterize a phenomenon. The second step is to propagate the uncertainties through the code to quantify the effect on the output/response functions. There are two ways to propagate the uncertainties, deterministic and stochastic based sampling techniques. For this study, stochastic based sampling is used, where random samples are generated from the parameter distributions (normal, lognormal, etc.) and then propagated through the code (Radaideh, 2019).

Latin hypercube sampling (LHS) is a stratified sampling technique where the uncertain domain is divided into N segments and the range of each uncertain variable is N segments of equal

probability as shown in Figure 5. For every uncertain variable, a sample is chosen from the bins randomly.

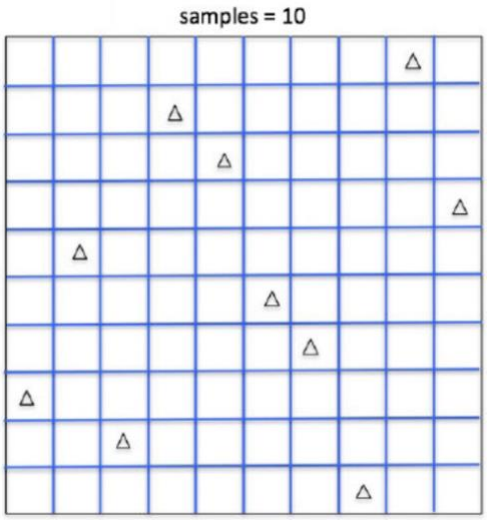


Figure 5. Latin Hypercube Sampling technique supported by Dakota.

Every subgroup of the uncertain variable is randomly assigned to a sample only one time. There is no restriction on the number of bins for each uncertainty, but LHS requires all the uncertain variables to have the same number of bins. The total number of samples equals the number of bins for each variable. LHS also provides a more accurate estimate of the means compared to random sampling and requires fewer samples than MC for the same statistical analysis accuracy (Radaideh, 2019). In this study, LHS technique is used to propagate the uncertainties as it is simple and straightforward.

The code evaluations are performed for each of the samples and the output quantities of interest are postprocessed. As part of the framework, sensitivity analysis is also performed with the aim of identifying whose input variations have the most impact on the output's uncertainty both qualitatively and quantitatively (Radaideh, 2019). Screening methods such as One factor at a time (OAT) and Morris method give qualitative information on the inputs. Correlation based

indices give quantitative information, they explain how the output uncertainty is impacted by each input or by a combination of input variables (Delipei, 2019).

Pearsons coefficient (PC) measures the linear correlation between the input and output parameters. The PC value varies between [-1,1], where +1 is for total positive linear correlation, 0 is when its totally uncorrelated and -1 for total inverse correlation. The covariance of input and output variables divided by their products of the standard deviations will give the PC value as shown in Equation 14.

$$S_{PC}^{i} = \frac{C_X^i Y}{\sigma_X^i \sigma_Y} \tag{14}$$

Spearman coefficient (SC) is used instead of PC, when the model tends to be non-linear but monotonic and measures the strength of monotonic relationship between paired data (Delipei, 2019). The SC value interpretation is similar to PC and it varies between [-1,1], where +1 is for positive monotonic correlation, 0 is when its totally uncorrelated and -1 for strong inverse monotonic correlation. The SC value is calculated as shown in Equation 15 and Equation 16.

$$S_{SC}^{i} = 1 - \frac{6\sum_i d_i^2}{n(n^2-1)} \tag{15}$$

$$d_i = R(X_i) - R(Y_i) \tag{16}$$

Here, (n) is the number of data points between variable, $(\mathrm{d_i})$ is the difference in ranks of the i^{th}element.

Finally, dimension reduction is also proposed as part of the framework. In this method, the statistical analysis is made easy by significantly reducing the number of input parameters thereby reducing the effect of the “curse of dimensionality” (Radaideh, 2019). For the dimension reduction

techniques to be effective, the underlying quantity (input or output) needs to exhibit strong correlations, as usually found in strong functional quantities (Delipei, 2019). There are two techniques to dimension reduction, one is based on screening and sensitivity analysis and the other is feature selection which will be primarily used in this work. Principal component analysis is a feature selection technique that uses information from potential correlations to reduce the dimension of the input/output space.

Principal component analysis (PCA) is a feature selection technique that comes in handy when dealing with large and complex dataset that are difficult to interpret. The technique reduces the dimensionality of the dataset without any major loss in information by introducing new uncorrelated variables that maximize the variance continuously (Jolliffe, 2016). These uncorrelated variables are defined by the dataset that is being processed and the whole PCA method reduces down to a eigenvalue/eigenvector problem to find these variables. To build on this, reduced order modeling techniques can be used to solve high-fidelity, computationally expensive multi-physics problems by using surrogate models. The surrogate model allows for the user to understand the system quantitatively and also interpret the system dynamics at a lower computational cost (Lucia et al., 2004). Especially in the field of CFD, surrogate modeling is gaining popularity as it is easier and faster to run repetitive tasks on the reduced order model and perform UQ studies. The most popular surrogate models are: (1) Generalized linear models; (2) Polynomial chaos expansion (PCE); (3) Kriging; (4) Artificial neural networks (ANN) (Delipei, 2019).

2.4 Dakota

The Design and Analysis toolKit for Optimization and Terascale Applications, Dakota, is a general-purpose software toolkit for performing design optimization, uncertainty quantification,

parameter estimation, design of experiments, and sensitivity analysis on high performance computers.

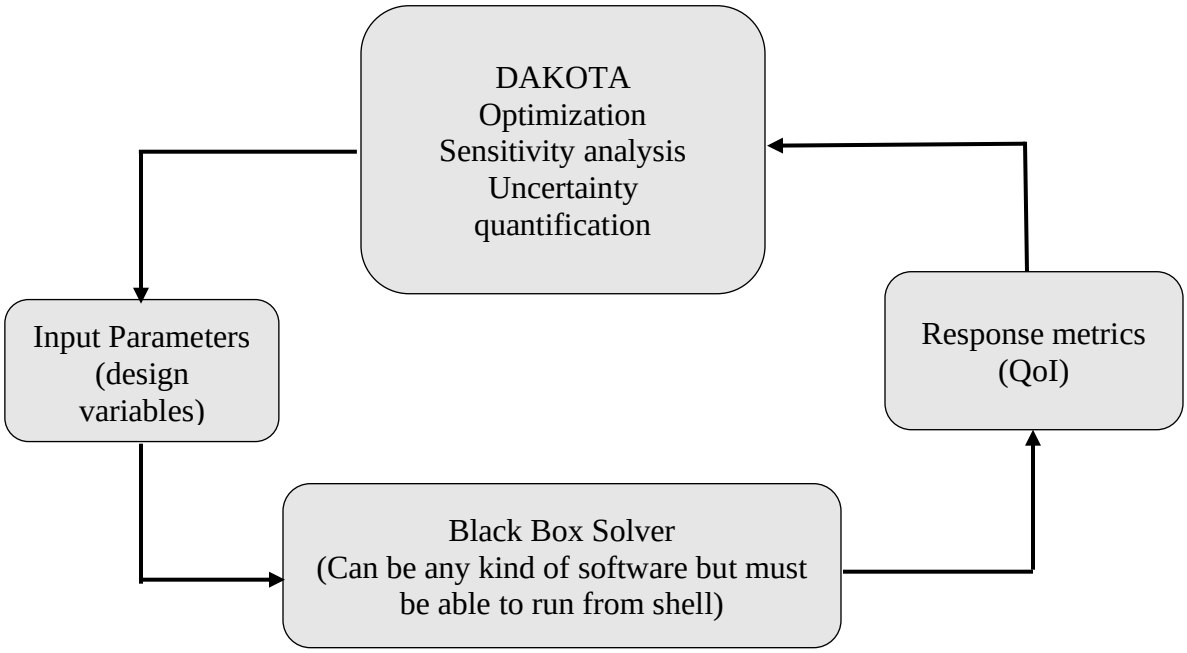


Figure 6. Dakota Framework

It is currently being developed and supported by Sandia National Labs. Figure 6 illustrates the general framework of Dakota. The code also supports several sampling techniques such as Monte Carlo (MC) sampling, Latin Hypercube Sampling (LHS), and Grid Sampling. In this work, LHS technique is used to sample the uncertain domain.

To summarize, the governing theory behind GeN-Foam's sub-solvers and SAM was discussed. The uncertainty quantification framework was established and the different uncertainty quantification methods that will be used as part of this work were discussed.

CHAPTER 3

Code Capabilities and Verification

The analysis of LFMSRs during steady state, operational transients and accident scenarios require addressing unique reactor physics challenges with coupling between thermal hydraulics, neutronics, and species distribution phenomena. Also, from the previous section it was demonstrated that the LFMSR design was prone to recirculation zones that are not desired as they may degrade the structural stability of the reactor vessel.

Keeping these in mind, the following new code capabilities were added to GeN-Foam during the course of this PhD work. A mass transport model is implemented (pending verification) in GeN-Foam and is discussed in detail in Section 3.1.1. Similarly, a thermo-mechanics heat conduction model is implemented and verified as shown in Section 3.1.2. Followed by the verification of thermal-hydraulics sub-solver against Fluent in Section 3.1.3.

Since SAM is part of the proposed framework and will be used in the future to develop simplified models of the LFMSR, the code's capabilities are demonstrated in Section 3.2 by modeling a molten salt system and performing sensitivity studies. The SAM model was verified against a 2-D model developed in STARCCM+ software and is illustrated in Sections 3.2.3 and 3.2.4, respectively. As part of physics modeling, it is also important to characterize and reduce the uncertainties associated with the computational model. In light of this, GeN-Foam is coupled with Dakota in Section 3.3 followed by some preliminary results illustrating the successful coupling in Section 3.4.

3.1 New Capabilities added to GeN-Foam

3.1.1 Mass transport model

Material distribution throughout the system during normal operation constitutes what is in the system, where it is in the system, and how it is evolving. To understand the evolution of liquid fuel salt material over time, it is important to understand the following parameters:

- Inputs from steady-state reactor physics and temperature determining the flux spectrum and material distribution;
- Inputs from reactor physics, heat conduction, heat transfer, and fluid flow quantifying the material being removed from the fuel salt during fuel cycle simulations and fission product disposition;
- Removal, feed and plate-out of materials impacting the chemical and thermophysical properties of salt;
- Mass transport information is required to quantity the movement and accumulation of fuel material within the primary loop;
- Inputs from temperature, fuel salt compositions, thermochemical state and chemical reaction rates are required to characterize the material deposition onto the surfaces and corrosion of material from surfaces of structural alloys.

Implementation of Mass Transport Solver

A mass/species transport tool is necessary for liquid fuel reactors to study the transport of different species and analyze their effect on the reactor design and model. In this work, a framework is established to implement a new solver for species transport modeling inside GeN-Foam. As part of the solver, *speciesTransportClass* is solving the transport equations and a

separate mesh (*speciesMesh*) needs to be developed for the equations to be solved on. Two new field variables are introduced: *C* for the concentration of transported species and D_C for the diffusivity of species. The transport equation shown in Equation 17 solves for *C* and is coupled to the fluid solver to take into account the convection of species. The results from the fluid mesh are passed onto the species mesh using a cell-volume weighted mapping function.

$$\frac{\partial C}{\partial t} + \nabla \cdot (UC) - \nabla^2 (D_C C) = 0 \tag{17}$$

Here, (*C*) is the transported scalar, (*U*) is the velocity of the species and (D_C) is the diffusivity. For the purposes of testing, only one species has been modeled currently with the capability to expand in the future.

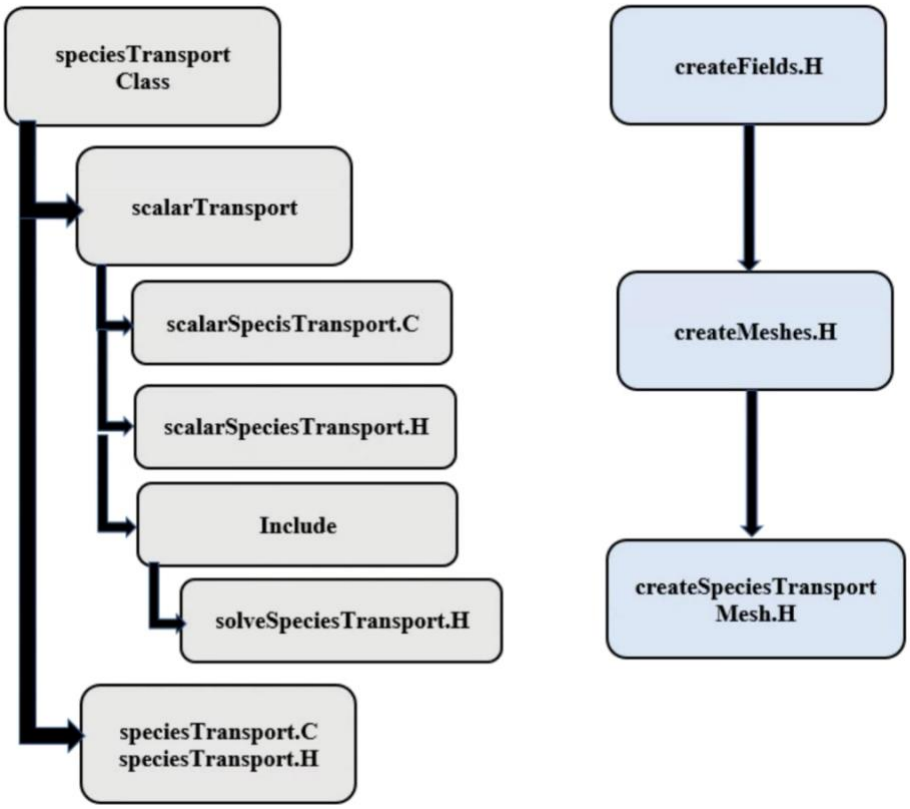


Figure 7. Species transport solver framework.

3.1.2 Thermo-mechanics heat conduction model

In GeN-Foam, the temperature values of the sub-scale structures are calculated by the thermal-hydraulic solver and are passed onto the thermo-mechanics solver which calculates the temperature induced stresses and displacement accordingly. The drawbacks of this approach are the solver's dependence on thermal-hydraulics sub-solver and its inability to take into account the solid-solid heat conduction phenomena thereby underpredicting the deformation caused by the thermal stresses.

Implementation and Verification

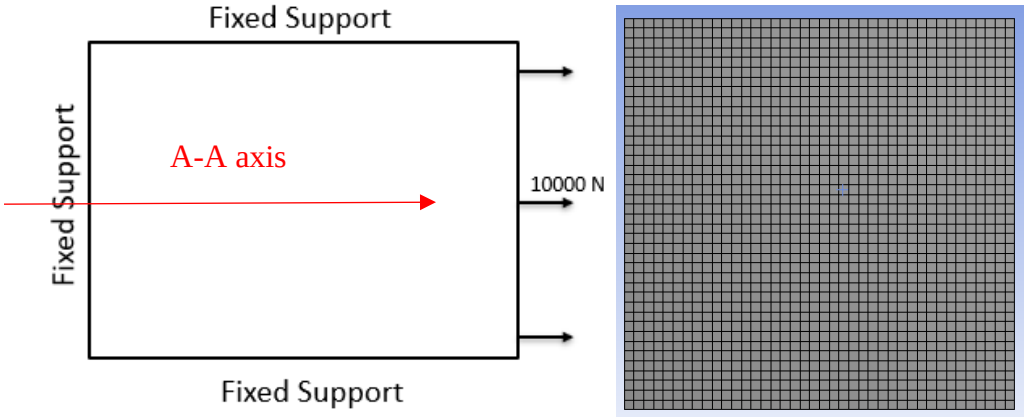
To overcome the above drawbacks, a new variable for solid material temperature (T_s) is introduced followed by a simple heat conduction equation in the *thermalMechanicsClass*. The material temperature field is coupled to the existing thermal-mechanics solver through Equation 19 to calculate the thermal stress. Finally, the thermal stress field is added to the existing solid stress field equations to calculate the total displacement.

$$\frac{\partial T_s}{\partial t} - \nabla \cdot k(\nabla T_s) = 0 \tag{18}$$

$$\nabla \cdot (\alpha * T_s) \tag{19}$$

Two test cases were developed to test/verify the correct implementation of the solver. In the first case, a simple 1×1 2-D cavity was modeled in GeN-Foam, OpenFOAM and Fluent as shown in Figure 8(a). A uniform force of 10,000 N was applied to the right wall while keeping all the other sides of the wall fixed. The solid displacement was studied along the AA axis shown in

Figure 8(b) and was compared between the three available codes. To keep the study consistent, the same number of mesh elements was used across all the three codes and the displacement results are shown in Figure 9.



(a) **(b)**

Figure 8. Geometry with boundary condition (a) and 2-D structured mesh (b).

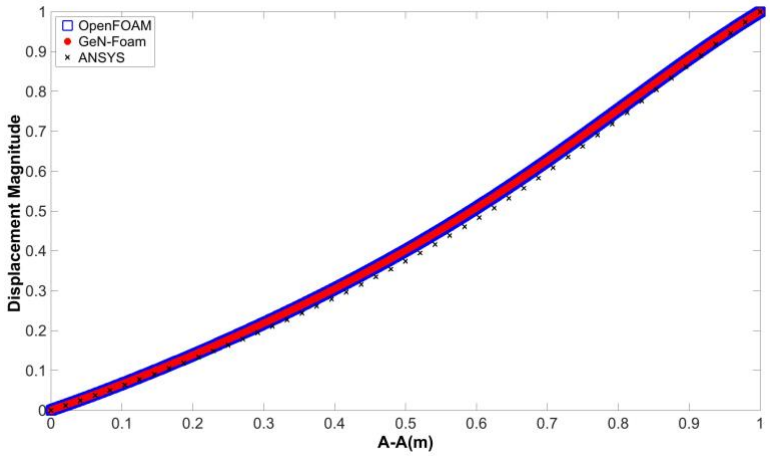


Figure 9. Normalized displacement along the A-A axis.

For the second case, a 2-D heat pipe model was chosen to verify the displacement caused by the thermal stresses. A schematic of the heat pipe model is given in Figure 10 along with the

imposed boundary conditions and the model design parameters are provided for reference in Table 1.

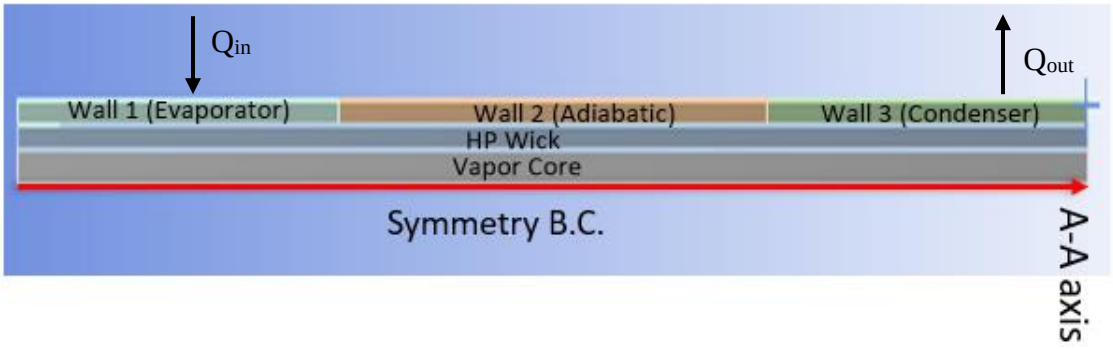


Figure 10. Schematic of the heat pipe.

Table 1. Heat pipe model design specifications

Parameter	Value	Unit
Vapor Core Diameter (d_{vc})	1E-03	m
Wick Thickness (t_{wi})	1E-03	m
Heat Pipe Wall Thickness (t_w)	1E-03	m
Fuel Slug Apothem	2.5E-02	m
Thermal Conductivity of Core ($k_{vaporcore}$)	1E+07	W/(mK)
Thermal Conductivity of Wick (k_{wick})	22.792	W/(mK)
Thermal Conductivity of Heat Pipe Wall (k_w)	24	W/(mK)
Heat Flux (Q'')	8333.3	W/m^2

A uniform heat flux (Q_{in}) of 8333.3 W/m^2 was added through the evaporator section and removed through the condenser section. For brevity, only one half of the heat pipe was modeled assuming symmetry boundary condition at the vapor core. Since this study focuses on modeling

the temperature distribution and its associated stresses, the two-phase nature of the vapor core and the respective flow is not modeled. The results were studied along the AA axis and was verified against the commercial code Fluent. A very good agreement was observed for both temperature distribution and normalized displacement magnitude as shown in Figure 11 and Figure 12 respectively.

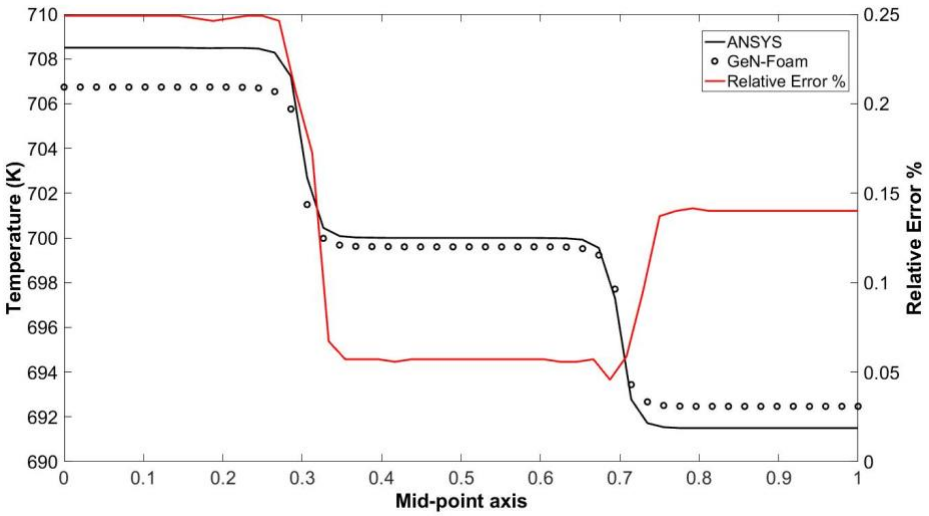


Figure 11. Temperature distribution along the A-A axis.

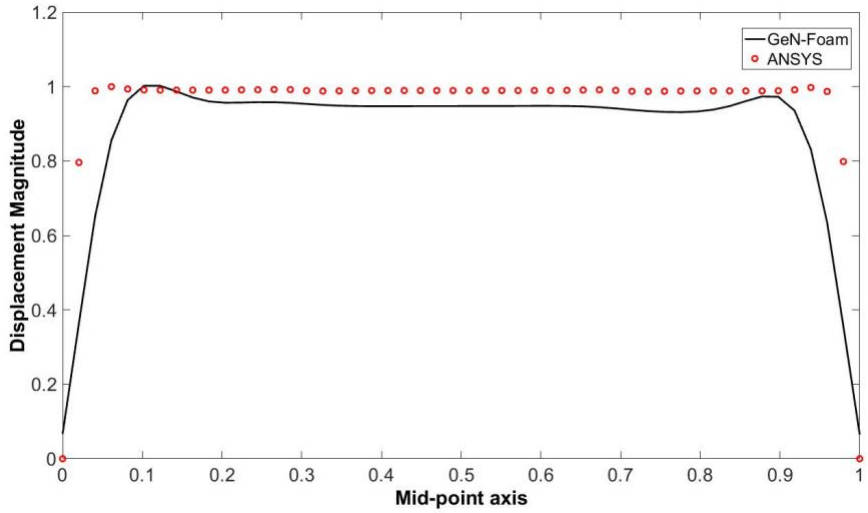


Figure 12. Normalized displacement magnitude along the A-A axis.

3.1.3 Verification of Thermal-Hydraulics Solver

Two models were developed as a part of this verification study. The first model was a simple 2-D right circular cylindrical geometry illustrated in Figure 13. To simplify the verification process, gravity term has been removed, no heat generation or heat transfer is considered, and the kinematic viscosity has been increased to force laminar flow in the geometry.

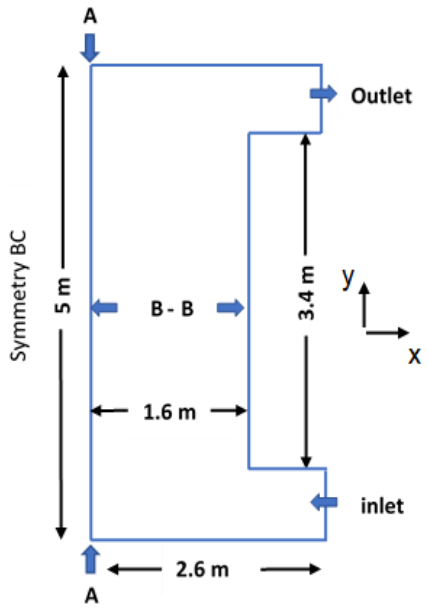


Figure 13. LFMSR core model

The velocity at the inlet was defined to be 1 m/s, the outlet BC as zero gradient and the wall was set to "no-Slip" BC. In the case of pressure boundary conditions, the inlet boundary was set to zero gradient and the outlet boundary condition was set to an arbitrary fixed value. The above boundary conditions ensure that the flow in the geometry is velocity driven and dictated by the magnitude of the velocity at the inlet. The accuracy of the results between the three solvers were measured through absolute error, the Fluent results was the reference solution, and the equation is:

$$\text{Absolute Error} = \left|U_{\text{Fluent}} - U_{\text{GeN-Foam, BPFoam}}\right| \quad (20)$$

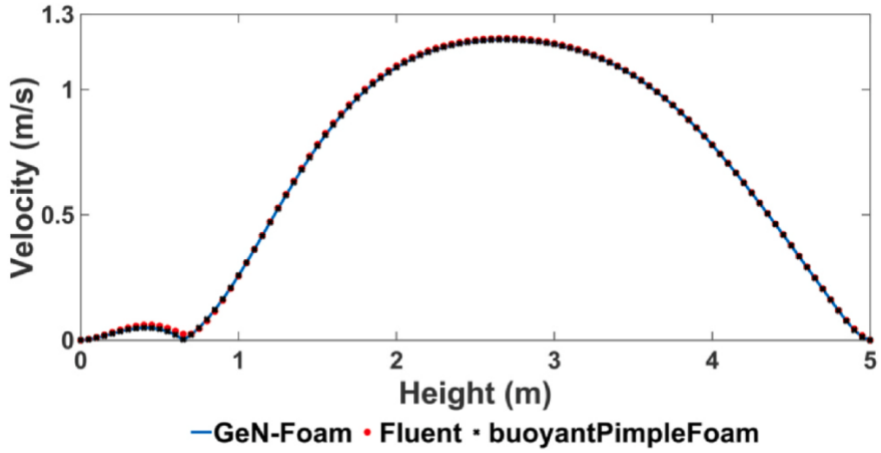


Figure 14. Velocity magnitudes along the (A-A) axis.

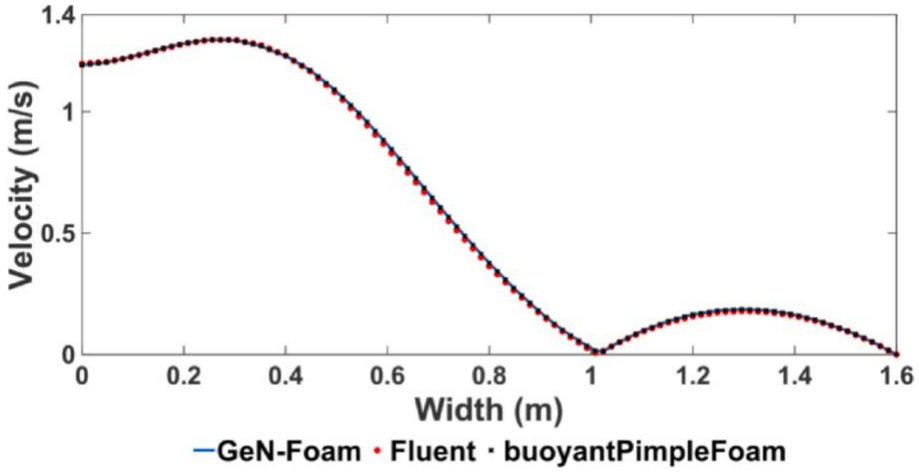


Figure 15. Velocity magnitudes along the (B-B) axis.

The velocity magnitudes along the A-A axis (axial direction) and B-B axis (radial direction) are compared. GeN-Foam and *buoyantPimpleFoam* results are identical as expected and they are in good agreement with the Fluent results. The absolute error between the axial and radial

velocity magnitudes are in the order of 1E-02 m/s. The absolute error is well within the acceptable range for CFD studies. The respective velocity profiles are shown in Figure 14 and Figure 15.

For the next verification study, a 2-D primary loop model was developed with core in the shape of an hourglass. In this study, the energy equation was turned off with the objective of solving only for the fluid flow. Since this was a closed loop model there were no inlet/outlet boundary conditions needed and the flow inside the loop was purely dictated by pump momentum and gravity. The case was run assuming water properties and the turbulence effects were accounted for in both GeN-Foam and Fluent.

It is important to estimate the maximum first cell height near the walls to effectively capture the boundary layer effects. The following set of Equations (21-25) were used to calculate the Δy_{wgp}:

$$C_f = 0.078Re^{-0.25} \tag{21}$$

$$Re = \frac{\rho * U * L}{\mu} \tag{22}$$

$$\tau_w = \frac{1}{2} * C_f * \rho * U^2 \tag{23}$$

$$U_\tau = \sqrt{\frac{\tau_w}{\rho}} \tag{24}$$

$$\Delta y_{wgp} = \frac{y^+ * \mu}{\rho * U_\tau} \tag{25}$$

Here, (ρ) and (μ) is the density and viscosity of the salt, (U) is the velocity of the salt at the inlet and (L) is the characteristic length. (C_f) is the skin friction coefficient, (τ_w) is the wall shear stress, (U_τ) is the frictional velocity. From equation (23) the first cell height was found to be 6E-03 m when y^+ equals 30. Keeping this cell height as reference, a mesh with 1.5 million elements

was developed. A mesh sensitivity analysis was performed to investigate the possibility of using a mesh with fewer elements, while maintaining reasonable accuracy and lower computational cost. The velocities are studied along the axial and radial directions for the meshes. From Table 2 it is clear that a mesh with 490,000 elements was able to predict the velocities within a 2-4% accuracy of the reference values thereby reducing the computation time by almost three times.

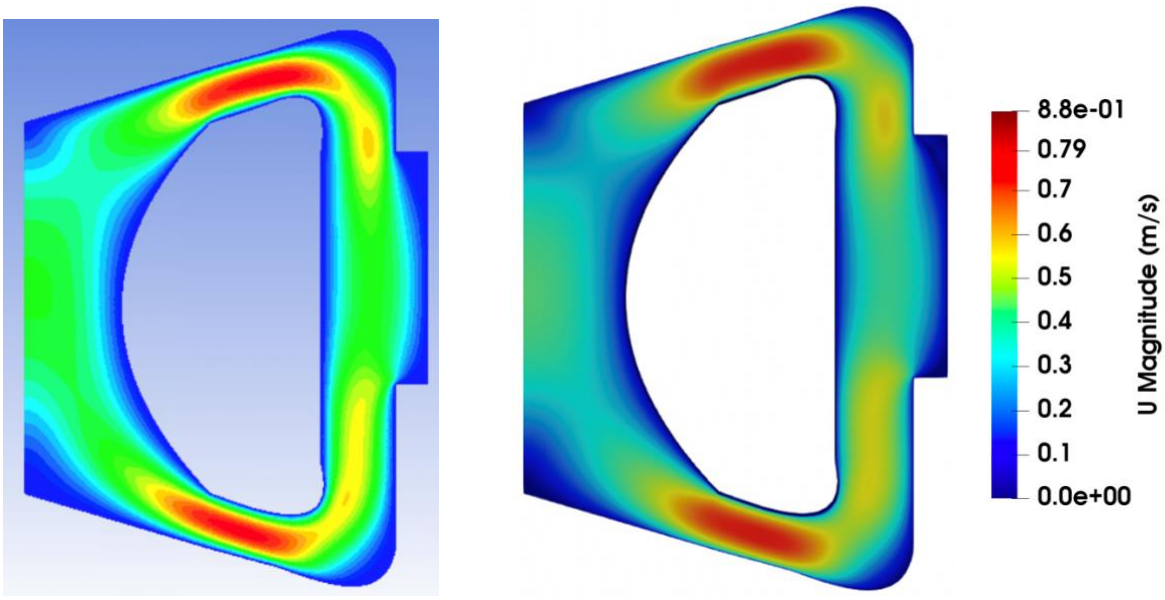
Table 2. Summary of mesh sensitivity analysis.

No. of Elements	Avg. Relative Error (Radial)	Avg. Relative Error (Axial)	Standard Deviation (Radial)	Standard Deviation (Axial)
55,030	19.592	22.68287	19.909	25.42690862
166,481	9.620	11.88799	9.950	15.64432799
493,010	2.492	3.942095	2.825	6.450425698
1,479,990	-	-	-	-

So, for the analysis hereon, a mesh with 490,000 elements was used. The test matrix for the verification study is summarized in Table 3. A porous k-epsilon turbulence model was chosen in GeN-Foam and to be equivalent to the k-epsilon model in Fluent, the porosity terms were turned off. The velocity contours from the two codes match very well and is shown in Figure 16. The velocity magnitudes were studied along the A-A axis (axial direction) and B-B axis (radial direction). From Figure 17 and Figure 18 we can observe that the velocity plots show good agreement between the two codes.

Table 3. Test Matrix

Parameter	Value	Unit
Density (ρ)	1000	kg/m^3
Dynamic Viscosity (μ)	10	kg/(ms)
Cmu	0.09	//
C1	7.00E-01	//
C2	2.9450	//
C3	-0.33	//
sigmaK	1	
sigmaEps	1.3	



(a) **(b)**

Figure 16. Velocity contours: (a) Fluent, and (b) GeN-Foam.

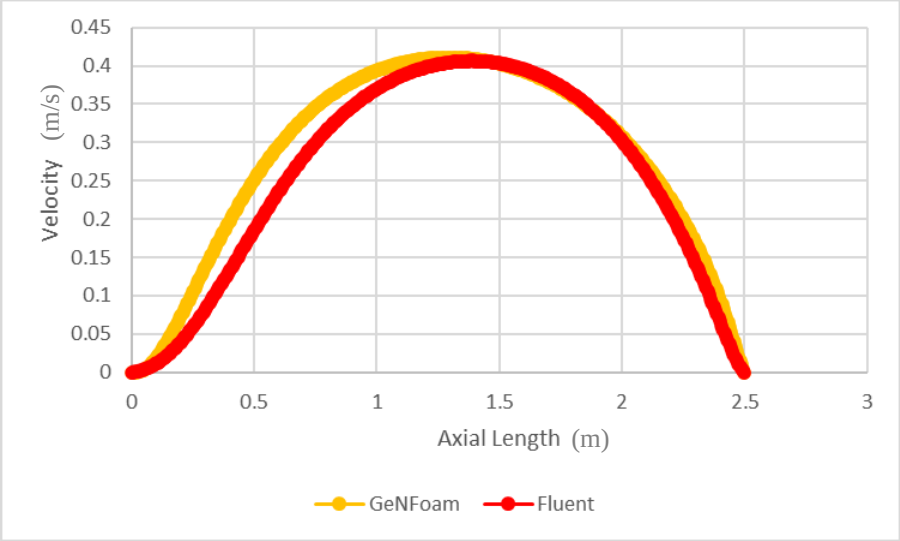


Figure 17. GeN-Foam vs Fluent velocity plots along the axial vector

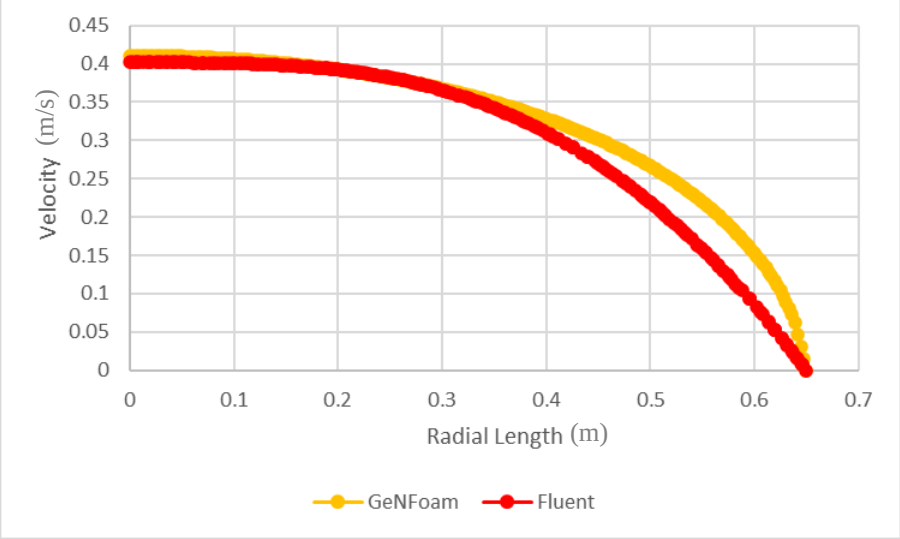


Figure 18. GeN-Foam vs Fluent velocity plots along the radial vector.

3.2 SAM Verification Study

Since SAM will be a part of the proposed framework, the code's capabilities are demonstrated in this section. The Versatile Experimental Salt Irradiation Loop (VESIL) was modeled in SAM and a sensitivity study was performed. In an effort to verify SAM, a 2-D model

of the VESIL system was developed in STARCCM+ (STAR CCM+ Users Manual) and their respective flow and temperature distributions were compared.

3.2.1 Model Design and Specifications

The SAM code was used to model the temperature and flow distribution inside the VESIL. VESIL is a cartridge type loop that is currently under consideration by INL to be deployed in the Advanced Test Reactor (ATR) for performing salt irradiation experiments (Abdalla et al., 2020). A schematic of the natural circulation loop is shown in Figure 19.

SAM is used for the analysis of VESIL to avoid the complexities associated with analytically accounting for axial natural circulation and radial thermal conduction. The tool also provides the flexibility to iterate on the geometrical and thermal-hydraulic parameters in order to estimate the feasibility of a natural-circulation salt irradiation test loop in the ATR while incurring minimal computational costs.

VESIL was modeled using a combination of 1-D fluid-containing “component” and heat structures to represent the various flow paths and heat transfer. The fluid flow in the riser and downcomer is modeled using four pipe components and two other pipe components to connect the riser and downcomer. In reality, the riser and downcomer are separated by a baffle made of the SS-316 material and the temperature at the outer wall is maintained at a constant temperature of 343 K to account for the heat rejected to the ATR coolant.

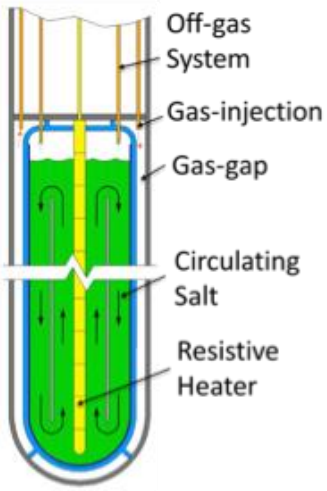


Figure 19. Illustration of VESIL design (Abdalla et al., 2020).

To simplify the SAM model definition, a heat structure is defined in the place of baffle, gap, fins and outer wall. This simplified assumption helps in defining variable thermal conductivities across the structures and adds flexibility to the model. A 0-D heat structure component is used to model the heat conduction across the baffle thereby accurately capturing the physics. Figure 20 describes the 1-D model developed in SAM.

The thermal properties of the salt were based on values found in the open literature for chloride-based salts. The density of the salt is expressed as a linear function of temperature, taken from (Desyatnik et al., 1975). Due to the lack of available literature on UCl_3-NaCl, all other thermal properties were assumed constant and were based on those of another chloride-based salt, bearing the chemical composition 5% $PuCl_3$ – 42% UCl_3 – 52% NaCl (Altahhan et al., 2020). The thermophysical properties are summarized in Table 4 and the material properties in Table 5.

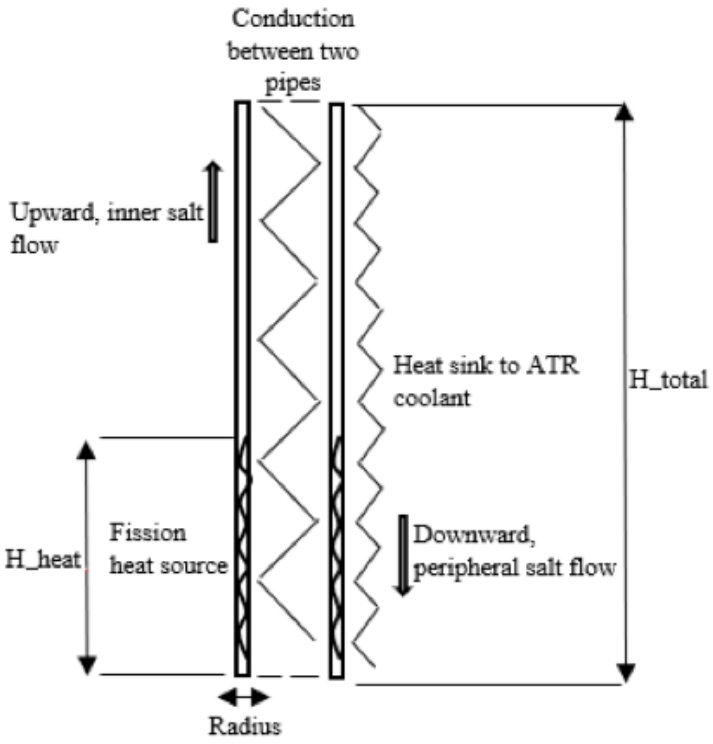


Figure 20. 1-D VESIL model in SAM.

Table 4. Summary of thermophysical molten salt properties

Parameter	Value	Unit
Density (ρ)	3860.40-0.84T (K)	kg/m^3
Specific Heat Capacity (C_p)	9.50E+02	J/(kgK)
Dynamic Viscosity (μ)	2.17E-03	kg/(ms)
Thermal Expansion Coefficient (β)	4.60E-4	K^{-1}
Thermal Conductivity (k_T)	7.00E-01	W/(mK)
Prandtl Number (Pr)	2.9450	//

Table 5. Material Properties

Material	Parameter	Value	Unit
SS-316	Density (ρ)	7.9E+03	kg/m^3
He-Ar (Gas-Gap)	Density (ρ)	4.66E-01	kg/m^3
SS-316	Specific Heat Capacity (C_p)	6.38E+02	J/(kgK)
He-Ar (Gas-Gap)	Specific Heat Capacity (C_p)	5.2E+02	J/(kgK)
He-Ar (Gas-Gap)	Thermal Conductivity (k_T)	0.08	W/(mK)
Cu (Fins)	Thermal Conductivity (k_T)	14.00	W/(mK)
SS-316 (Baffle)	Thermal Conductivity (k_T)	24.00	W/(mK)

3.2.2 Sensitivity Study

In this preliminary analysis, a uniform fission power density of 100 W/cm^3 is applied to the core region. To achieve reasonable natural circulation velocities, the effective resistance across the fins and the gap are calculated using a MATLAB code and is used as a fixed boundary condition in SAM input deck. The maximum and minimum thermal conductivities are limited by the gas-gap mixture and the fin properties.

The buoyancy driven velocity is a function of temperature drop, height, and the diameter of VESIL. The temperature drop within the loop, ΔT, will be a design constraint dictated by material limits and operational targets of the tested concept. The analysis therefore focuses on

dimensional parameters: the total height H, and the salt radius r. In order to independently elucidate their impact on salt velocity, DT is held constant during the parametric evaluation. This is achieved by fine-tuning the conductivity to the ATR coolant.

The resulting evolution of salt velocity with VESIL dimensions is plotted in Figure 21 and Figure 22. The dependency can now be inferred to the dimensions of VESIL. A 1% increase in height leads to a 0.18% increase in velocity, while a 1% increase in radius is responsible for around a 0.68% increase in the salt velocity.

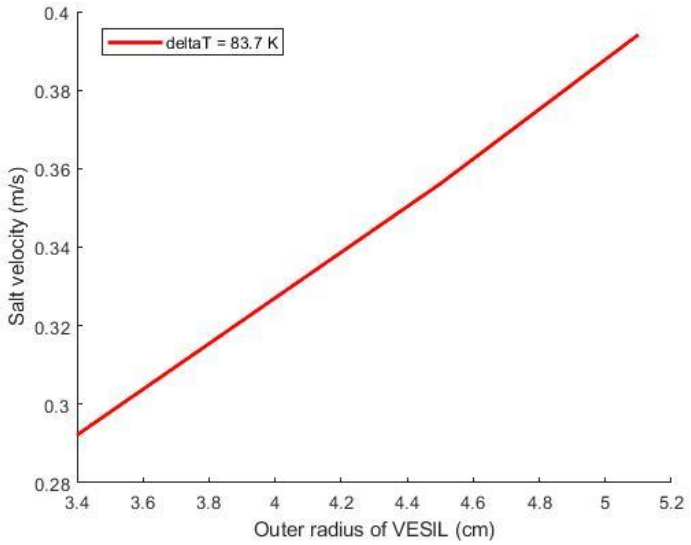


Figure 21. Evolution of salt velocity with VESIL height. Radius was held constant at 3.4 cm.

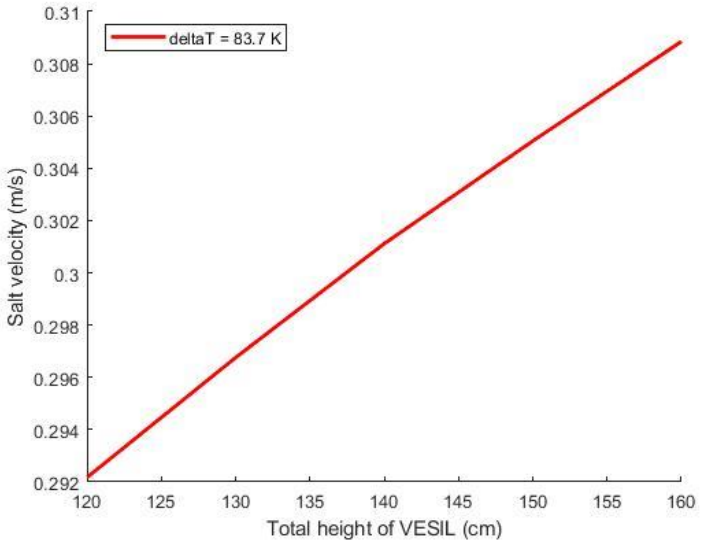


Figure 22. Evolution of salt velocity with VESIL radius. Total height was held constant at 120 cm.

3.2.3 STARCCM+ Model

The primary objective is to study the velocity and temperature distribution of the molten salt inside the VESIL and a 2-D loop model is developed in STARCCM+. The Boussinesq approximation and a linear dependence of the salt density on the salt temperature was also assumed, and the thermal expansion coefficient β is used to save computational time. Figure 23 describes the 2-D axisymmetric model developed in STARCCM. Since only one half of the VESIL loop is modeled, symmetry boundary condition is used for the physics of molten salt on the lateral surface.

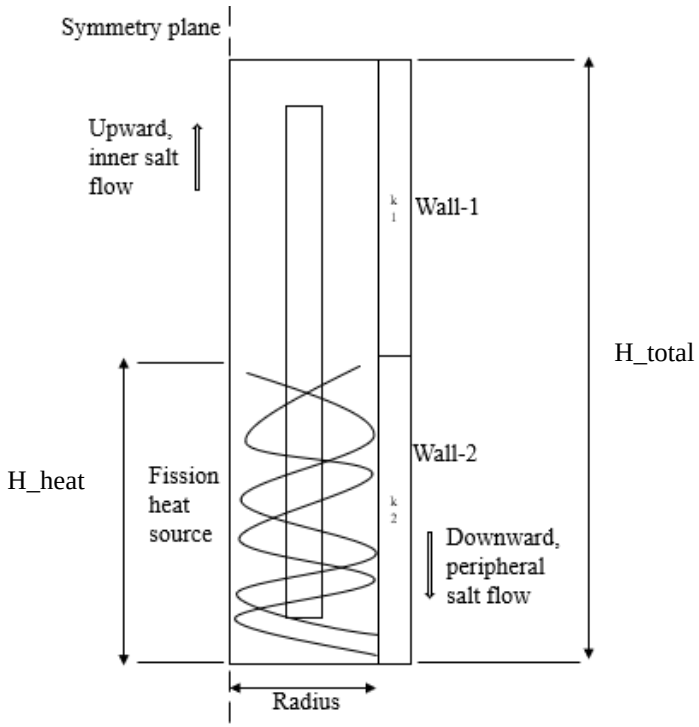


Figure 23. 2-D axisymmetric VESIL loop STARCCM+ model.

For all other components such as baffle, wall-1, gap and wall-2 are modeled as solids and the wall-boundary condition is used. Interfaces between the fluid and solid components are also defined to model the conjugate heat transfer and to accurately capture thermal conduction. A constant volumetric heat source is applied to the core fluid to model the fission power. The surface of wall-1 and wall-2 are given a constant temperature boundary condition thereby modeling the coolant flow. This assumption holds valid as the mass flux of the coolant is high, the change in bulk temperature becomes negligible and hence a constant wall temperature boundary condition can be assumed for the structures directly in contact with the coolant.

3.2.4 SAM to STARCCM+ comparison

A benchmark comparison between the SAM and CFD results is done to verify the validity of the model developed in SAM. As SAM is a 1-D system code, the correlations used for fluid

flow and heat transfer modeling can vary the results significantly. It is important to develop 2-D model in STARCCM+ and use the results as a benchmark to use the correct correlation for SAM and see if the results are within the acceptable error limits for a benchmark comparison.

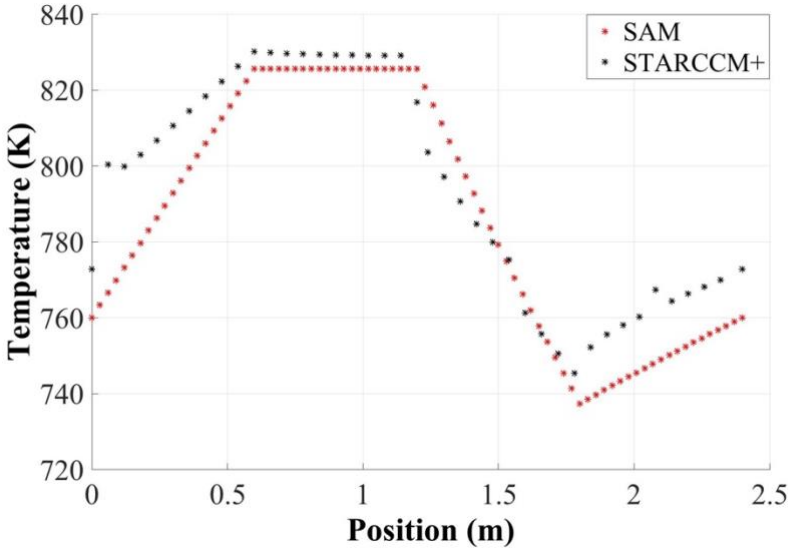


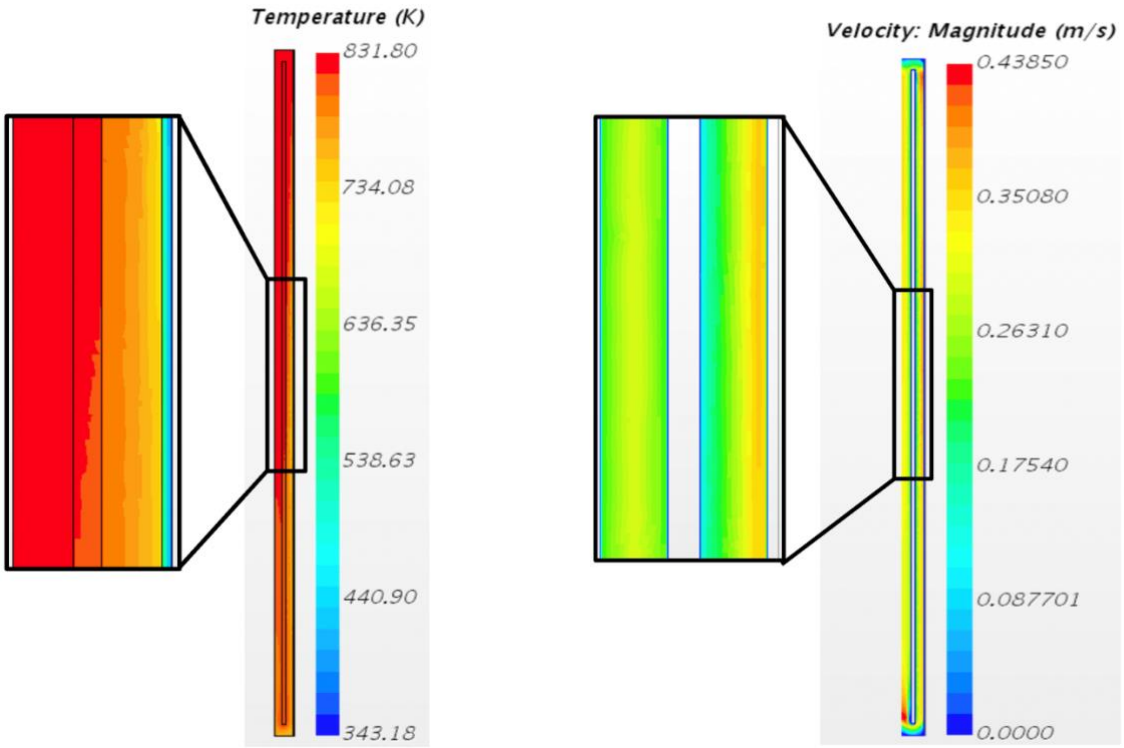
Figure 24. SAM vs STARCCM+ (Temperature profile).

The temperature profiles obtained from SAM and STARCCM+ are shown in Figure 24. The trend of the curves seems to agree well and Table 6 shows the relative error percentages for this benchmark comparison.

Table 6. Summary of the materials considered for fin modeling

Parameter	SAM	STARCCM+	Relative Error %
Maximum Temperature (K)	825.74	830.22	0.5 %
Minimum Temperature (K)	737.42	745.49	1.0 %
Salt Velocity (m/s)	0.276	0.28	1.4 %

The temperature profile of STARCCM+ from Figure 24 shows a jump in temperatures at the beginning and at the very end. This jump in temperature is attributed to the geometry of the VESIL by itself. As the fluid flows from the downcomer to the riser, there is accumulation of salt at the bottom due to the formation of dead zones. These dead zones greatly reduce the salt velocity and this reduction in velocity leads to the salt spending more time closer to the baffles and thereby increasing the temperature of the salt as outlined in Figure 24. This phenomenon is not observed in SAM as the baffle is being modelled as a heat structure just to account for conjugate heat transfer. The temperature and velocity distributions are shown in Figure 25 to better understand the salt flow and heat transfer inside the VESIL system.



(a) **(b)**

Figure 25. STARCCM+ predictions for (a) temperature distribution, and (b) velocity distribution.

The results showcase the SAM code's capability to model liquid fuel molten salt systems and hence can be used in the future to build simpler 1-D models to perform uncertainty studies. The nuclear reactions are not modeled in this study, and they may require different modeling tools. Thanks to the coupling interface established in SAM, the code can be coupled to other high-fidelity codes such as GeN-Foam in our case, to perform a full plant system analysis.

3.3 GeN-Foam and Dakota Coupling Framework

Coupling GeN-Foam and Dakota will enable the user to perform high fidelity uncertainty studies. Dakota is similar to GeN-Foam as they both use a control file for their simulations. Dakota initially reads the input file specified by the user where the method, variables interface and responses for the uncertainty quantification are defined. The sampling method is identified in Chapter 2. Uncertain variables, probability distributions, means and standard deviations are specified in the variable section. The interface section defines the driver file name, the parameter file for the random values generated by Dakota, the GeN-Foam input template, and the response file for saving the GeN-Foam simulation results. Upon reading the input file, the simulator script is run which invokes the *dprepro* utility. This utility will create the *params.in* file and takes a *x.template* file and finds all occurrences of the variable names specified in the control file of Dakota and replace them with the, current values for the iteration. The script will then move the file that *dprepro* just created, into the GeN-Foam directory where the simulation will start. Figure 26 illustrates this coupling scheme. A similar approach was used to couple the system code SAM to Dakota as well.

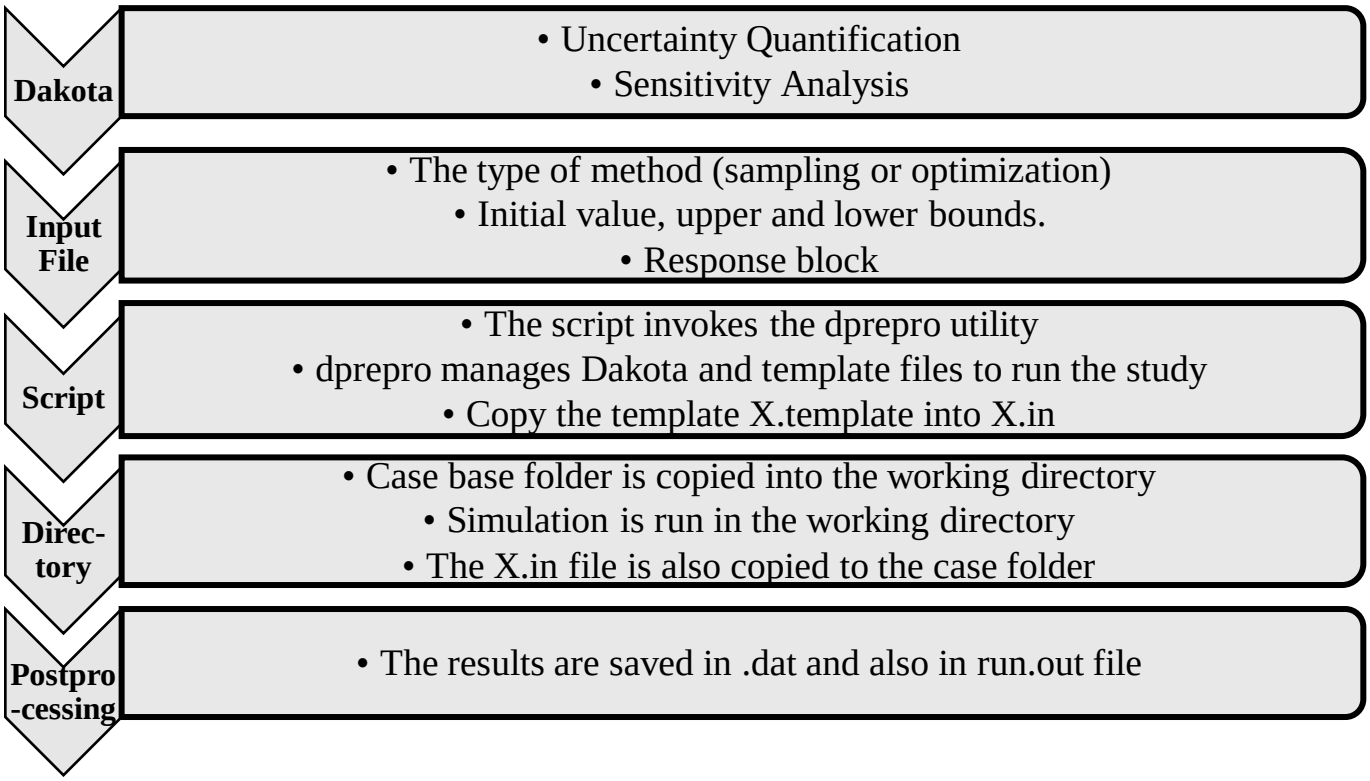


Figure 26. Dakota and GeN-Foam coupling framework.

3.4 Preliminary results showcasing successful coupling of GeN-Foam and Dakota

To verify the successful implementation of the coupling between GeN-Foam and Dakota, uncertainties are propagated through input thermophysical properties, and the output parameter of interest is the temperature at different locations of the reactor core. To demonstrate the GeN-Foam Dakota coupled code capabilities, statistical analysis is performed to quantify the impact of the input uncertainties followed by sensitivity analysis using the Pearson indices to identify the input variables that have an impact on the output's uncertainty.

3.4.1 Uncertainty Propagation

Nuclear and material data uncertainties need to be considered when modelling a reactor core. The fuel thermophysical properties are not well known and hence their uncertainties vary

between 2%-20%. In this work, to characterize the effects of thermophysical properties uncertainties, sensitivity analysis and uncertainty propagation analysis using statistical sampling was performed.

After successfully coupling Dakota and GeN-Foam, the statistical sampling incorporated was used to perform uncertainty quantification for the LFMSR system. Each of the thermophysical properties were sampled using LHS (Latin Hypercube Sampling) from a normal distribution of the property and were held constant for the entirety of the calculation. The thermophysical property data considered and the respective 95% confidence level uncertainties propagated are shown in Table 7.

Table 7. Summary of the thermophysical salt properties

Parameter	Value	Unit	Uncertainty
Density (ρ)	4.20E+03	kg/m^3	2%
Specific Heat Capacity (C_p)	9.50E+02	J/(kgK)	20%
Dynamic Viscosity (μ)	2.17E-03	kg/(ms)	10%
Thermal Expansion Coefficient (β)	4.60E-4	K^{-1}	15%
Thermal Conductivity (k_T)	7.00E-01	W/(mK)	15%

3.4.2 Postprocessing in GeN-Foam

Passing on information from GeN-Foam to Dakota in order to perform statistical analysis on the response functions is an integral part to this study. Since GeN-Foam is a high-fidelity CFD code, the results are in the form of contours which are hard to postprocess in Dakota.

To simplify the results and to better understand the impact of uncertainty propagation, the analysis was reduced to two parameters, temperature (in Kelvin) and pressure difference ΔP (in Pa). Preliminary analysis of the thermal-hydraulics model revealed the flow streamlines for the core among which one representative streamline was chosen for this study as shown in Figure 27. Several postprocessing utilities such as probes and pressure difference patches are added to GeN-Foam, enabling the code to output the necessary information. This simplifies the results and makes it easier for Dakota to perform statistical analysis on the output.

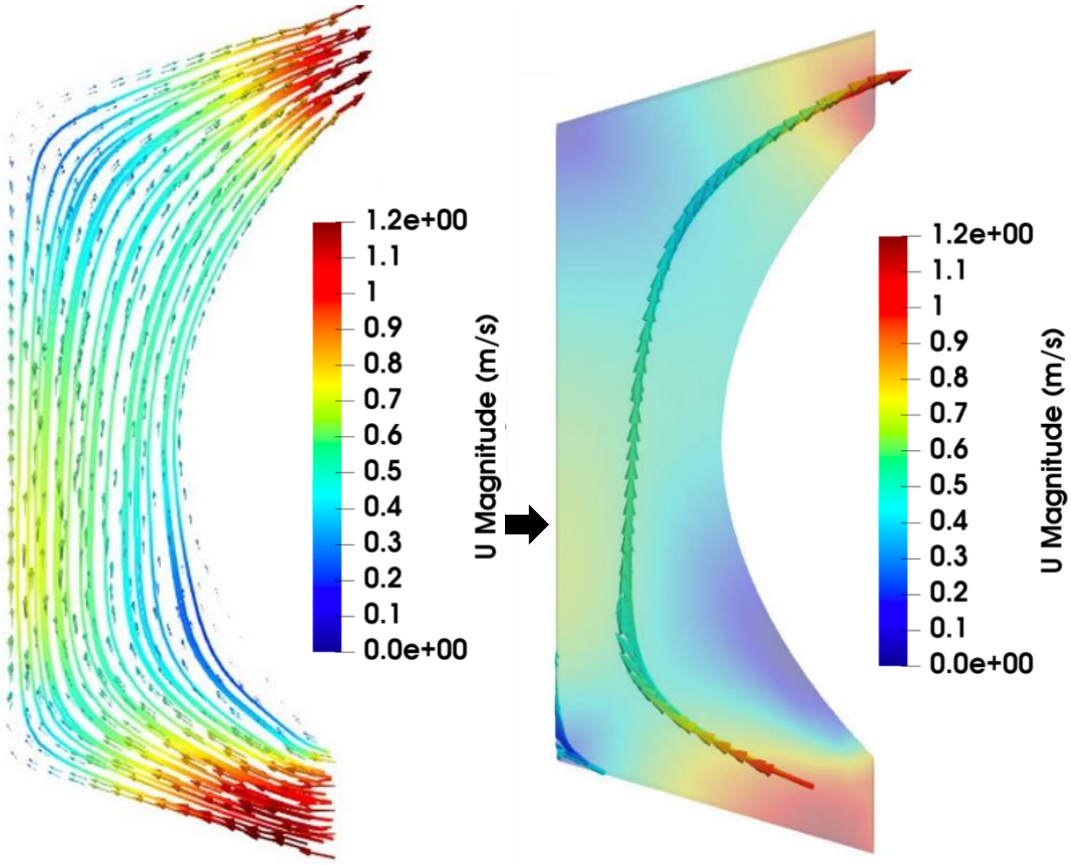


Figure 27. Flow velocity streamlines (a) and representative centerline streamline (b)

The first set of probes are chosen carefully along the representative streamline. Similarly, probes are also chosen at the symmetry, top, bottom and curved wall respectively to better understand the effects on recirculation zones. The different locations of the probes are shown in Figure 28 respectively and their respective coordinates are given in Appendix A and Appendix B for reference.

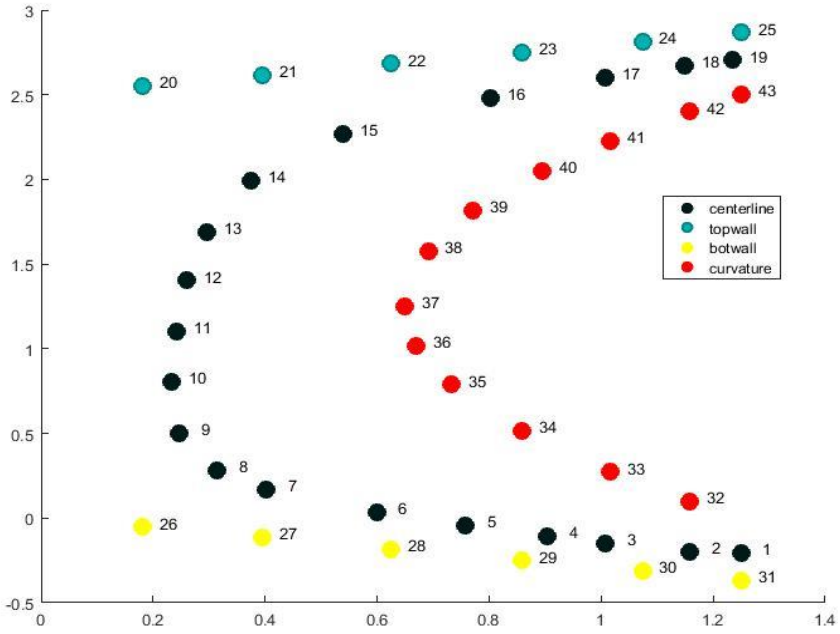


Figure 28. Summary of the data points and their respective ID numbers.

3.4.3 Uncertainty Quantification

In this study, 100 samples are chosen for the run. This will provide the statistical results important for the analysis, mean, and standard deviation. This quantification helps in identifying the hot spots inside the reactor core when thermophysical property uncertainties are propagated. Uncertainty quantification of centreline temperature is shown in Figure 29. The standard deviation values for the centreline probes near the outlet are higher by two orders of magnitude compared to

the probes near the inlet. As we move along the curved wall, a sudden spike in temperature is observed as shown in Figure 30. This reflects the recirculation zones that exist within the LFMSR core.

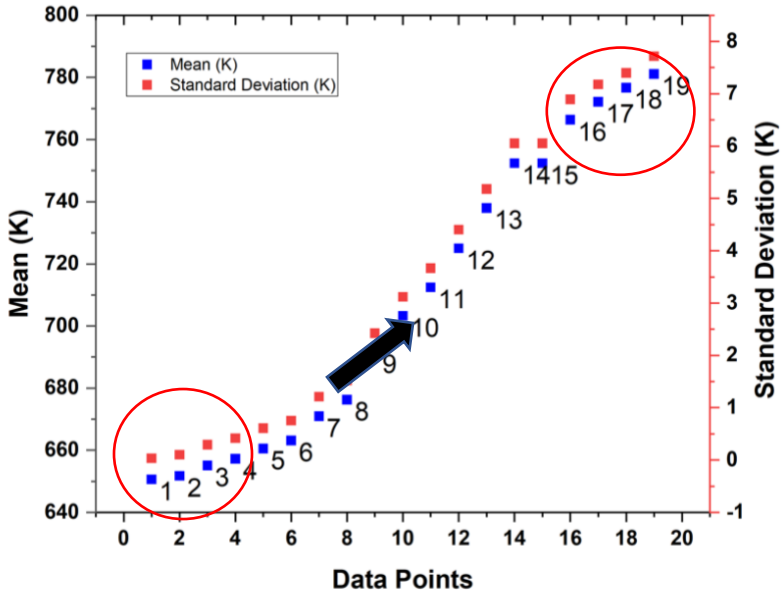


Figure 29. Uncertainty quantification of centreline temperature.

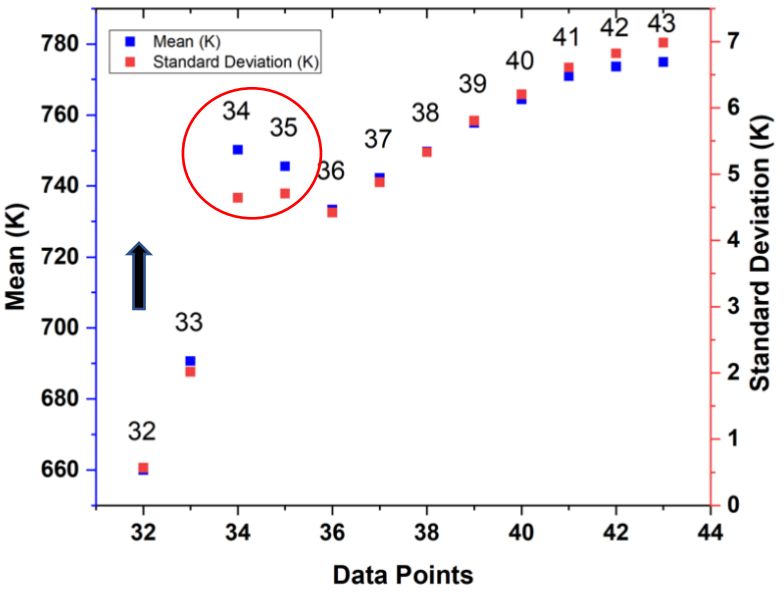


Figure 30. Uncertainty quantification of temperature at curved wall.

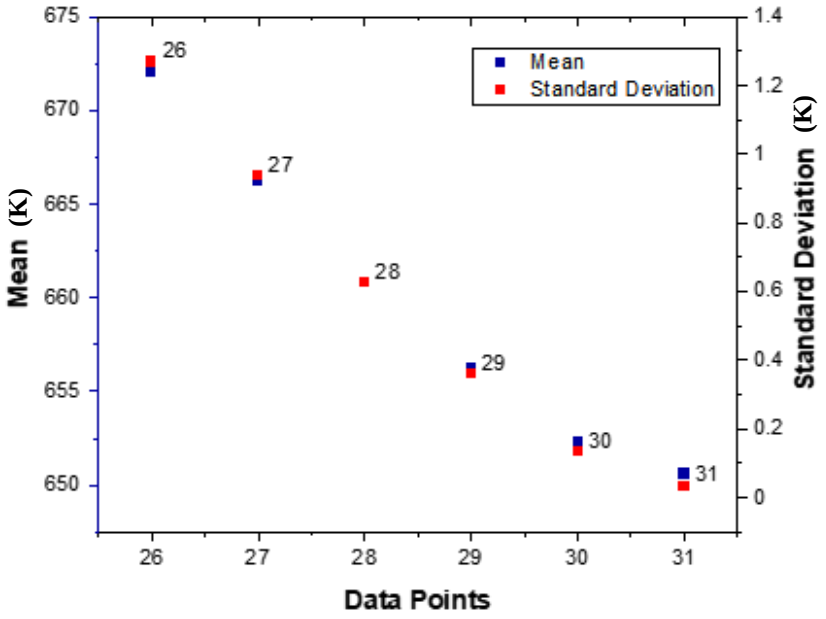


Figure 31. Uncertainty quantification of temperature at top wall.

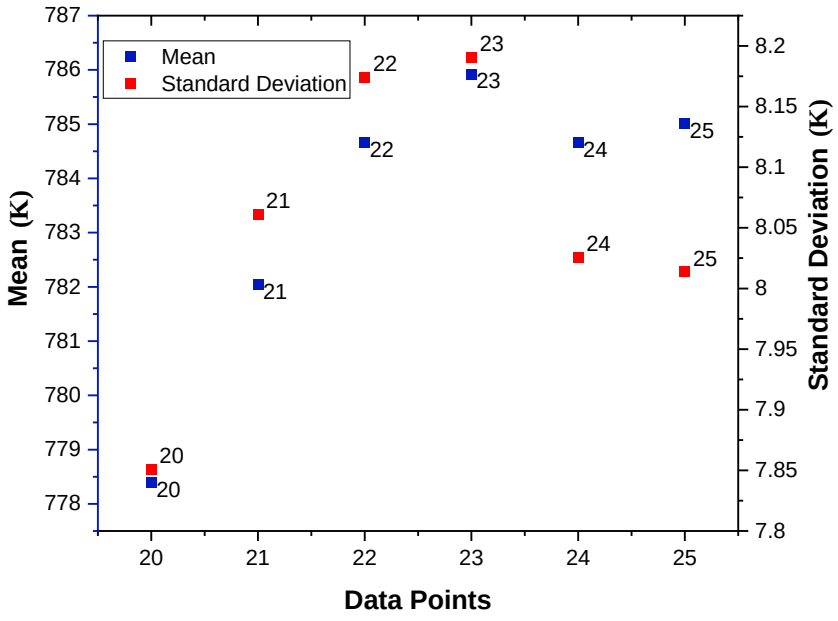


Figure 32. Uncertainty quantification of temperature at bottom wall.

It is clear from the above figures that the impact of the uncertainties on temperature is felt more towards the outlet of the core. Hence for future modelling purposes, the number of probes chosen can be reduced and be more concentrated towards the outlet.

3.4.4 Sensitivity Analysis

A sensitivity analysis using Pearson correlations is done to highlight the necessity of using sensitivity indices to capture the interactions of inputs on the outputs. The impact of the uncertainties on the ΔP and temperature are quantified by the Pearson Coefficients and are shown in the following figures, respectively.

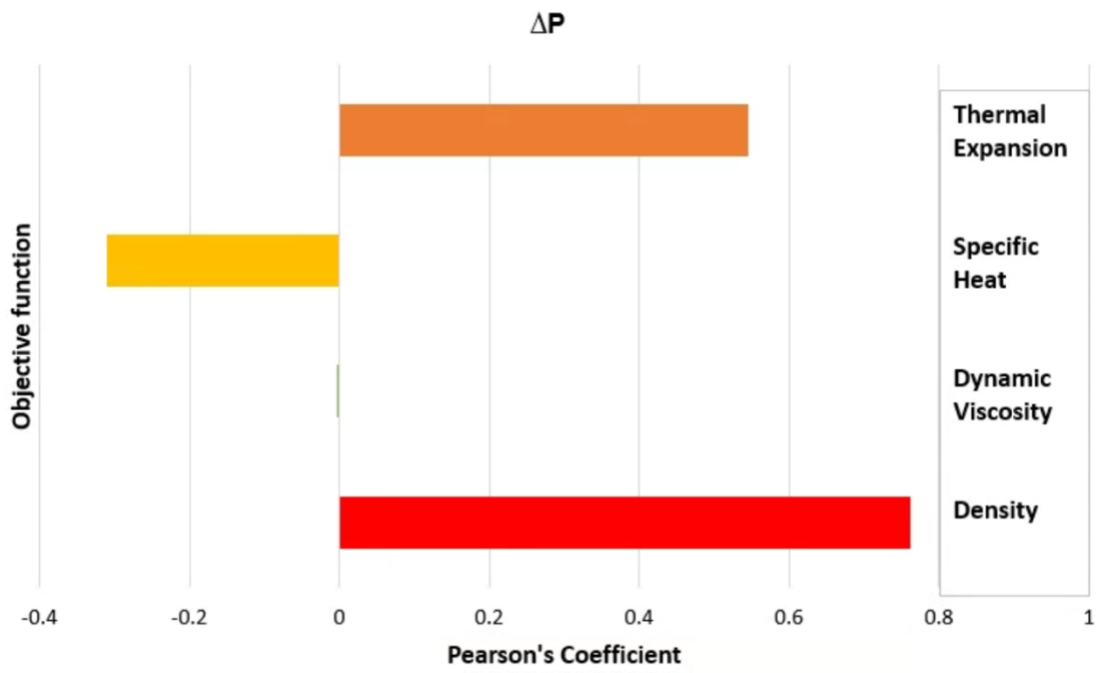


Figure 33. Sensitivity coefficients for pressure difference.

Thermal expansion coefficient and density of the fuel salt are important input parameters that influence the pressure difference inside the LFMSR system as seen from Figure 33. We know that as the fuel salt becomes denser and thermally expands, more pressure is needed to pump the

fuel across the core. This phenomenon is verified in Figure 33 as we see a positive linear correlation between density, thermal expansion coefficient and pressure difference.

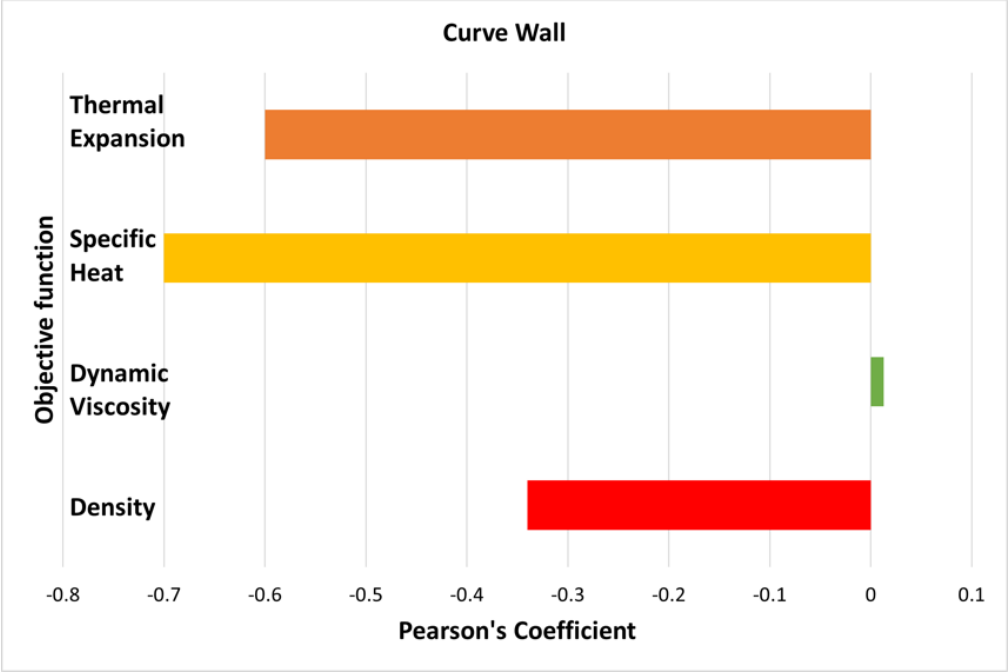


Figure 34. Sensitivity coefficients for curved wall temperature.

It is important to note that dynamic viscosity does not have a larger impact on the response functions as seen from Figure 34, Figure 35 and Figure 36 and is highly uncorrelated as the simulation was run under the assumption of incompressibility. Another important behavior to note is at (0.858, 0.513) of the curved wall where the thermal expansion coefficient is negatively correlated compared to other temperature probes. This is due to the recirculation zones that exist near the curved walls.

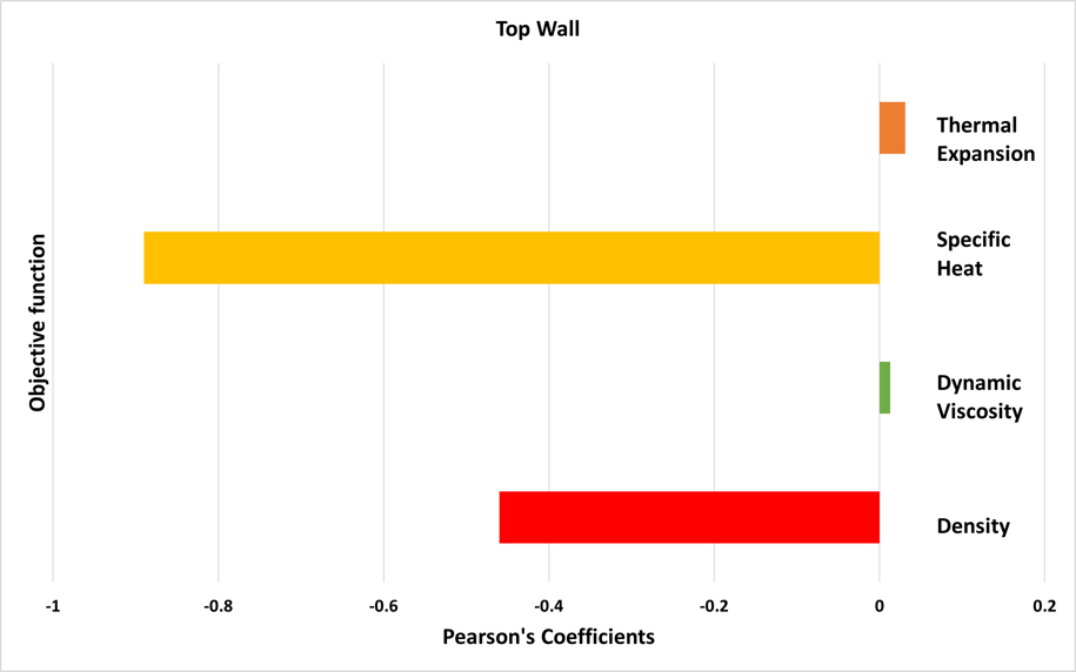


Figure 35. Sensitivity coefficients for top wall temperature.

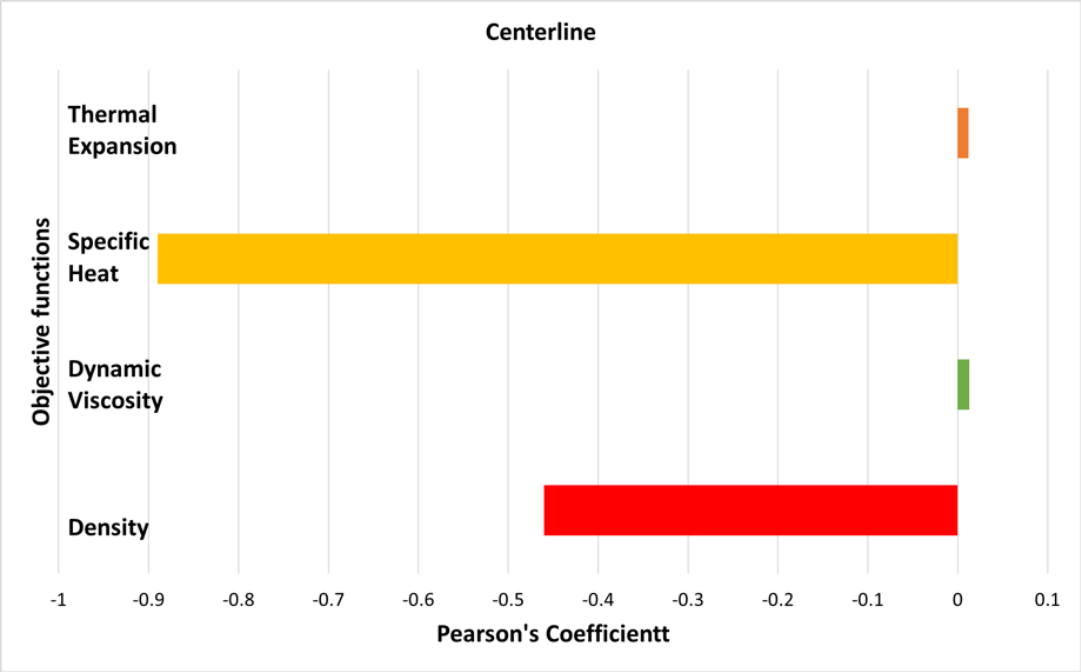


Figure 36. Sensitivity coefficients for centerline temperature.

To summarize, these new capabilities added to GeN-Foam enable the code to solve for species distribution phenomena and temperature induced stresses and displacement accordingly. The thermo-mechanic solver implementation was verified by modeling a heat pipe in GeN-Foam and comparing the displacement results with the Fluent solver. Whereas the thermal-hydraulic solver was verified against Fluent for two test cases and both showed reasonably good agreement. Since SAM will be part of the proposed framework, the code's capabilities were also demonstrated by modeling VESIL. The code was also verified against the CFD software STARCCM+. As part of the framework, GeN-Foam was coupled to Dakota and their coupling mechanism has been explained in detail in this chapter.

CHAPTER 4

LFMSR Core Design, Specifications and Modeling

MSRs are one among the six systems chosen for further study by the GIF. In this chapter, a preliminary core model, which was developed based on the reference MSFR concept, is discussed in detail. Initially, the core model was a simplified right circular cylindrical geometry as demonstrated in Section 4.1 and thermal-hydraulic and neutronics studies were performed. These studies revealed the formation of recirculation zones, and as a result, a new core design was proposed that could alleviate the issues arising from the previous geometric design.

4.1 Initial LFMSR Core Model and Specifications

The initially proposed core design was a simplified right circular cylindrical geometry. The fuel salt composition is given by Table 8 with a chemical form of $1 \cdot PuCl_3$ $8 \cdot UCl_3$ $10 \cdot NaCl$, yielding a molar weight of 3660 g/mol. The dimensions of the rector are given in Figure 37, while the properties used for this case are reported in Table 9.

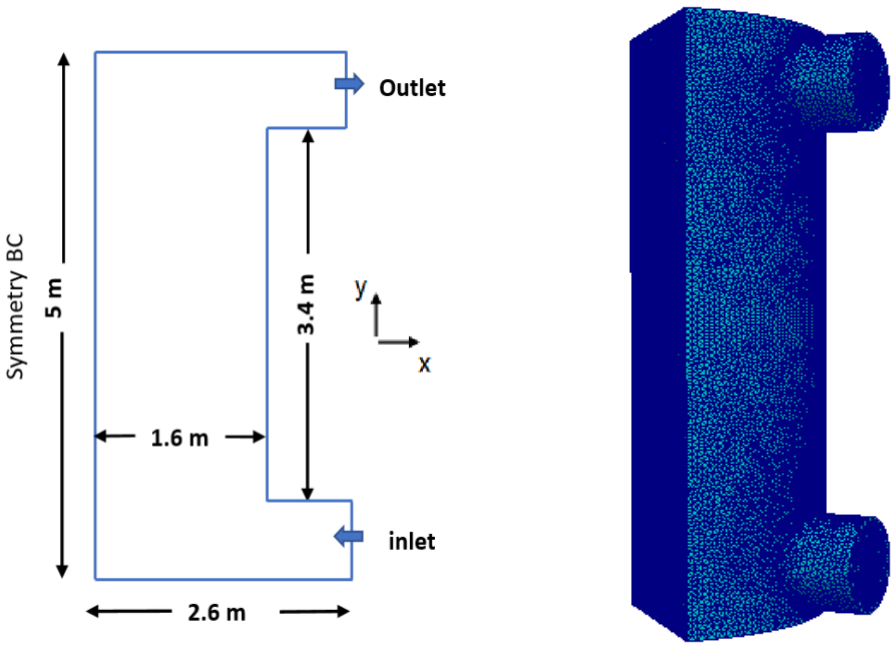
Table 8. Initial salt composition

Isotope	Atomic Fraction	Isotope	Atomic Fraction
U-235	1.02E-03	Pu-241	8.92E-04
U-238	1.41E-01	Pu-242	1.78E-03
Pu-239	1.33E-02	Na-23	1.81E-01
Pu-240	1.78E-03	Cl-37	6.59E-01

In order to save on the computation time only a quarter of the core was modeled in GeN-Foam as shown in Figure 37.

Table 9. Molten salt properties at 750 K

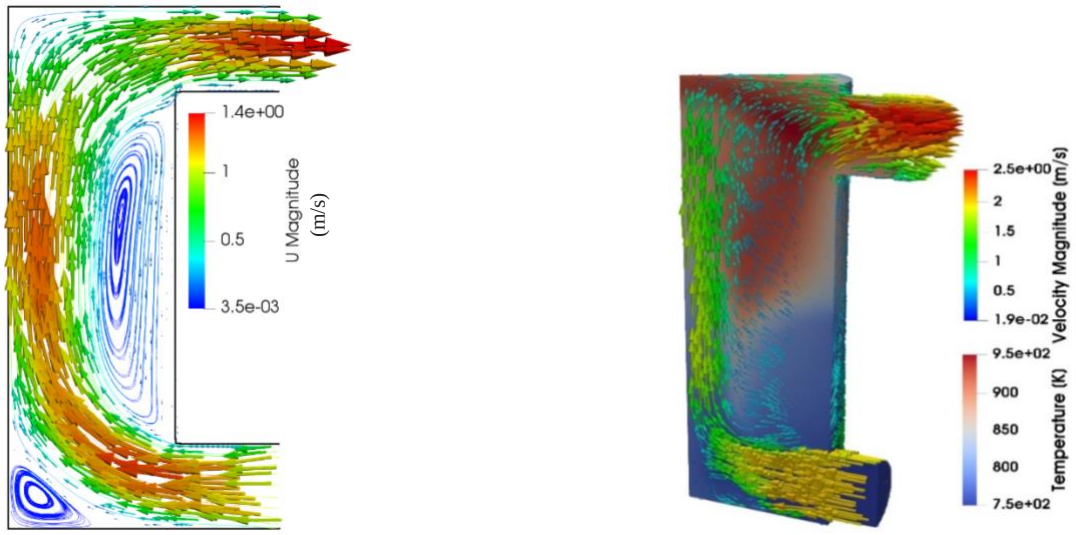
Parameter	Value	Unit
Density (ρ)	4.20E+03	kg/m^3
Specific Heat Capacity (C_p)	9.50E+02	J/(kgK)
Dynamic Viscosity (μ)	2.17E-03	kg/(ms)
Thermal Expansion Coefficient (β)	4.60E-4	K^{-1}
Thermal Conductivity (k_T)	7.00E-01	W/(mK)
Prandtl Number (Pr)	2.9450	//



(a) **(b)**

Figure 37. Side view of LFMSR core (a) and 3D spatial mesh (b)

The salt velocity at the inlet was 2 m/s with an inlet temperature of 750 K. The outlet boundary conditions are zero gradient for both the temperature and velocity. The pressure boundary conditions are atmospheric pressure, and the calculation was initialized using an average temperature of 800 K. The k-epsilon turbulence model is used with a turbulence intensity of 2% and mixing length of 0.056 m. For neutronics, zero flux boundary conditions were used on the system walls with zero gradient boundary conditions for the inlet and outlet fuel. Symmetric boundary conditions were used at the core center and a power of 625 MW was used for this simulation. The preliminary 2-D/3-D thermal-hydraulics results are shown in Figure 38. The spatial distribution of the neutron flux for six energy groups in the reactor is shown in Figure 39. The flux spatial distribution is higher at the core center with the maximum occurring in energy group 3. The calculated k_{eff} for the 3-D geometry is 1.060375. Figure 40 shows the spatial distribution of the precursors' concentration in the whole reactor for the first delayed neutrons group ($T_{1/2}$=278 s) and the eighth delayed neutrons group ($T_{1/2}$=0.39 sec). The short-lived precursor (Group 8) decays almost in its place of production while the longer-lived delayed neutron precursor (Group 1) accumulates near the outlet. This phenomenon depends on the neutron flux distribution and hence has a similar spatial distribution to that of the neutron flux (Altahhan et al., 2020).



(a) **(b)**

Figure 38. Flow velocity vectors for 2-D cylindrical geometry (a); and flow velocity vectors and temperature distribution for 3-D cylindrical geometry (b) (Altahan et al., 2020).

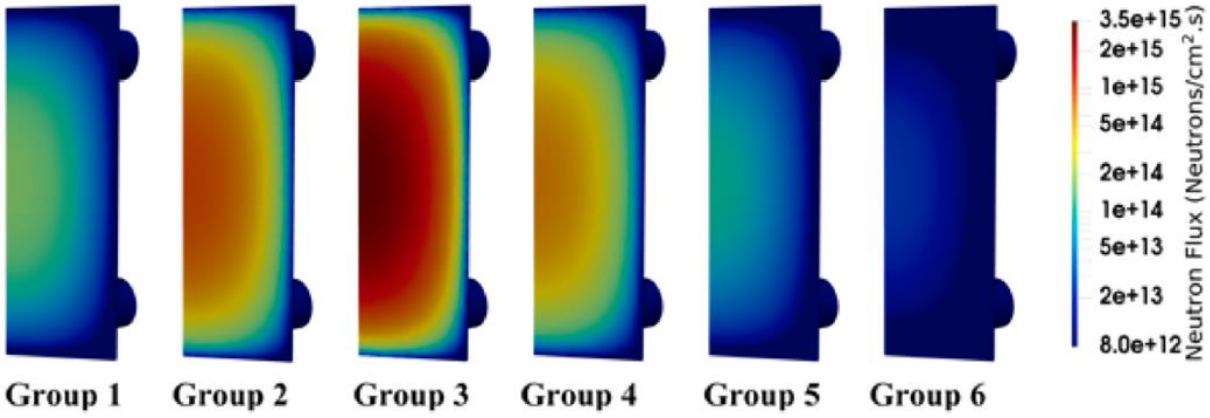
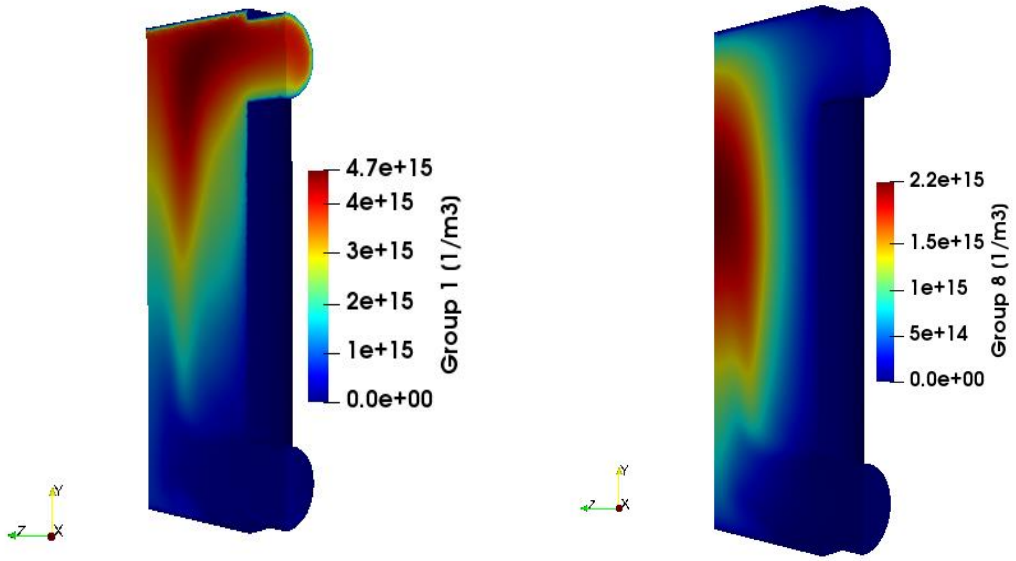


Figure 39. Neutron flux spatial distribution (Altahan et al., 2020).



(a) **(b)**

Figure 40. Delayed neutron precursor spatial distribution for group 1 (a) and group 8 (b) (Altahan et al., 2020).

The right circular design is relatively simple to model, but its flow paths cause recirculation zones. These are not desired because they allow coolant to retain heat energy and may degrade the reactor vessel with thermal striping.

4.2 Newly Proposed Core Model and Specifications

A new core geometry is proposed that can minimize the presence of recirculation zones. The new geometry is based on an hourglass shape and is as shown in Figure 41. The LFMSR core design consists of sixteen loops, with each loop consisting of three main components: the core, the pump, and the heat exchanger. In order to save on computational cost only one-sixteenth of the core was modeled as shown in Figure 41. A similar curved wall design was also proposed and evaluated in the EVOL project for fluoride-based MSRs (Brovchenko et al., 2019; Li et al., 2015). The design was successful in minimizing the recirculation zones (Li et al., 2015). The fuel salt

used for this study is same as the previous study and its composition is given in Table 8. The dimensions of the reactor core are given in Figure 41 and the thermophysical properties used to set up the reference case are shown in Table 9.

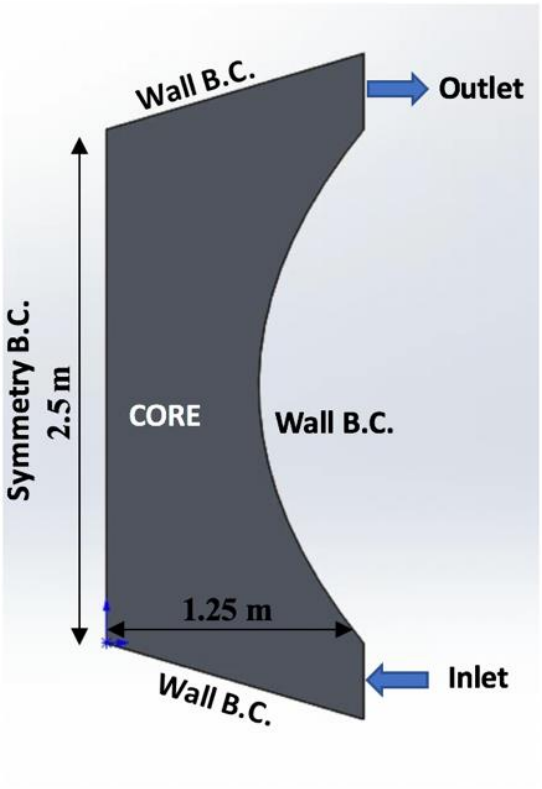
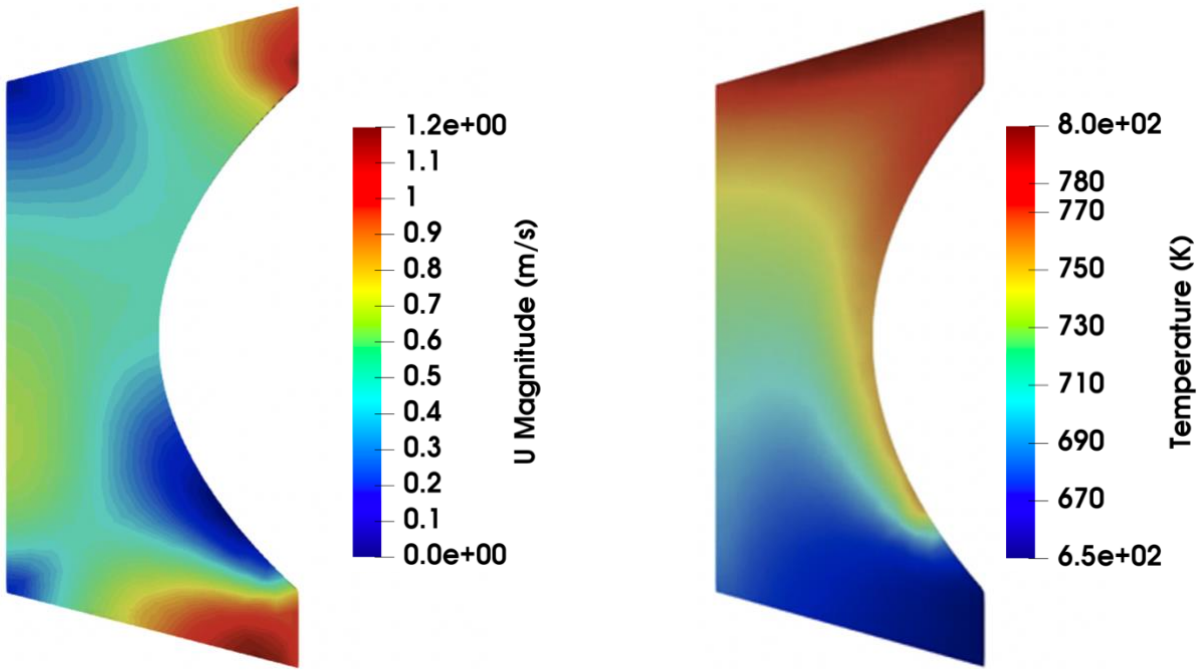


Figure 41. LFMSR core geometry

In the updated LFMSR model, the salt velocity at the inlet was taken to be 1 m/s with an inlet temperature of 650 K. The outlet boundary conditions are zero gradient for both temperature and velocity. The pressure boundary conditions are atmospheric pressure, and the simulation is started by initializing an average temperature of 800 K throughout the core. The k-epsilon turbulence model is used for modeling the flow alongside the RANS equations.

The preliminary 2-D results are shown in Figure 42. The velocity is higher where the cross-sectional area is smaller as expected and is also low at the bottom center due to the deviation

created by the design of the core. Also, the temperature is higher towards the outlet of the core as the fluid in this position would have absorbed more energy with respect to fluid in the other positions of the core and because of the constant inlet temperature boundary condition.



(a) **(b)**

Figure 42. Thermal hydraulics parameters: flow velocity (a) and temperature (b).

4.3 Design Optimization

The preliminary design does limit recirculation zones inside the reactor core thereby avoiding any hot spot formation inside the core. In reality, there are also other factors that need to be considered when optimizing the design. Further studies need to be performed to optimize the reactor core geometry and primary loop from a neutronics standpoint and performance of major systems. These analyses must consider the multi-physics design impacts on plant economic

performance such as pumping requirements, sizing of heat exchangers, sizing of system pressure boundaries (Holcomb et al., 2013).

4.3.1 Reactor Core Geometry

To perform the design optimization study, a non-dimensional parameter (σ) is introduced as shown in equation 26.

$$\sigma = \frac{H}{R} \tag{26}$$

It is defined as the ratio of the height of the reactor to the radius of the vessel. The goal of this study is to find an optimum design based on the multiplication factor ($\boldsymbol{k}_{\infty}$) inside the core. The height and the radius of the reactor core are varied, and the respective eigenvalues are calculated as shown in Table III. References indicated that the eigenvalue needs to be in the range of [1.006, 1.247] for molten salt reactor operation and an optimum reactor core design is chosen based on that constraint. The second important constraint is that for a fixed power, larger the reactor dimension smaller the power density and from literature it is understood that for a reactor with large dimensions, the thermal power reduction during transients isn't very pronounced either (Altahan et. al, 2020). Based on the above constraints, a reactor design with height 4.1 m and radius 3 m was chosen for further studies. The design optimization parameter (σ) acts as a reference parameter to inform the community and make any design dependent decisions faster and easier.

Table 10. Reactor core optimization based on multiplication factor

H	R	σ	k_∞
3.24 m	1.25 m	2.592	0.68416
4.3 m	1.35 m	3.185	0.79034
6.1 m	4 m	1.525	1.15839
5.4 m	3.5 m	1.542	1.12970
4.7 m	3.5 m	1.342	1.12188
4.1 m	3 m	1.366	1.08925

4.3.2 Sizing of Heat Exchanger

Primary heat exchanger plays a significant role in the design of the MSFR system as it is responsible for extracting the heat from the primary side and transferring it to the secondary side. The size of the primary heat exchanger depends on four main parameters, reactor power, reactor core dimensions and thermophysical properties of primary and secondary fluid (Ariu, 2014). In our study, the reactor power, core dimensions and thermophysical properties are fixed and are given in Table III. When it comes to sizing heat exchangers for MSFR system there are other design constraints that need to be accounted for such as compactness of the design to limit the residence time of fuel salt in non-active regions, low salt inventory, lower pressure drops and resistance to high temperature (973-1073 K) (Ronco et. al, 2020).

Considering the design constraints imposed by the MSFR system, a conventional shell and tube heat exchanger might not be sufficient to achieve high heat transfer rate as there would be significant losses in pressure drop which in turn would increase the pumping requirements. To keep the pumping requirements low for this MSFR system, an advanced heat exchanger, Printed

Circuit Heat Exchanger (PCHE) that can accommodate the requirements of a MSFR system is chosen.

Printed circuit heat exchanger is manufactured by a company called Heatric that uses chemical etching and diffusion bonding process to create a heat exchanger core with no joints, welds or points of failure (https://www.heatric.com) (Yoon et. al, 2014). PCHE uses micro-channel technology combined with diffusion bonding enabling the design to be 85% more compact and lighter than conventionally available technologies such as the shell and tube heat exchanger (https://www.heatric.com).

With regards to the sizing of PCHE there is only one main constraint, reactor core dimensions. Taking this into consideration, several design iterations were performed by building 3-D CAD models in SOLIDWORKS for both single loop and sixteen loop models to tentatively pick the PCHE dimensions which are illustrated in Figure 43. The PCHE is modelled using the following assumptions (Ronco et. al, 2020): (1) Heat conduction along axial direction is neglected (2) Negligible heat transfer from heat exchanger walls (3) Counter flow heat transfer. The layout of tube arrangement in PCHE for this study is shown in Figure 44.

In this study, thermophysical properties of the salt on the primary and secondary side are known and the temperature at the inlet and outlet of the PCHE is fixed. The secondary inlet temperature is fixed as well, and the secondary outlet temperature is varied to determine the total number of channels needed. This is to verify that the proposed dimensions of the PCHE system is well within the design limits imposed by the number of channels.

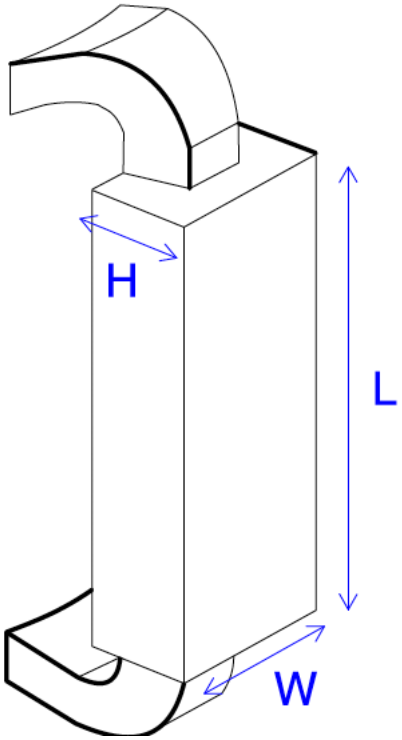


Figure 43. Dimensions of PCHE (Ronco et. al, 2020)

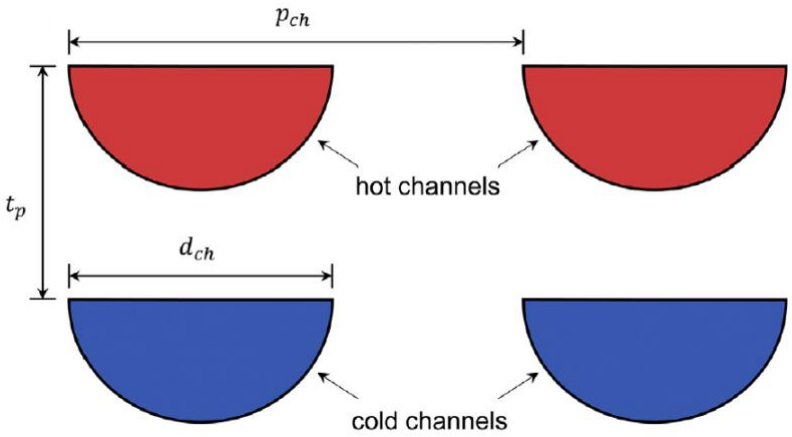


Figure 44. Tube layout in PCHE (Ronco et. al, 2020)

Table 11. PCHE design parameters

Parameter	Value	Unit
Primary inlet temperature ($T_{in,p}$)	900	K
Primary outlet temperature ($T_{out,p}$)	835	K
Secondary inlet temperature ($T_{in,s}$)	800	K
Diameter of micro-channel (d_{ch})	0.0021	m
Length of micro-channel (L_{ch})	1.79	m
Power	2400	MW

The overall heat transfer coefficient is calculated as shown in Equation 27,

$$U = \left(\frac{1}{h_{hot}} + \frac{1}{h_{cold}} + \frac{t_p}{k_{wall}}\right)^{-1} \quad (27)$$

Where, (h_{hot}) is the hot side heat transfer coefficient, (h_{cold}) is the cold side heat transfer coefficient, (k_{wall}) is the thermal conductivity of the wall and (t_p) is the thickness of wall $(0.625d_{ch})$. The heat transfer coefficient for flows in semi-circular channels is defined as shown in Equation 28 (Ebadian et. al, 1998):

$$Nu = \frac{h_{hot/cold} * d_h}{k_{hot/cold}} \quad (28)$$

Where, (d_h) is the hydraulic diameter of the channel calculated as shown in Equation 29.

$$d_h = \frac{d_{ch}}{(1+\frac{2}{\pi})} \quad (29)$$

The pressure drop across the primary and secondary side of the PCHE is calculated using Equation 20 respectively.

$$\Delta P_i = 2 * f_i * \rho_i * u_i^2 * \frac{L_{ch}}{d_h}, \qquad i = hot, cold \tag{30}$$

Where, (f_i) is the fanning friction factor $(15.767/Re)$ (Ebadian et. al, 1998), (ρ_i) is the density of the fluid, (u_i) is the fluid velocity, $(d_h)$is the hydraulic diameter and (L_{ch}) is the length of the channel. For a fixed secondary outlet temperature (870 K), the overall heat transfer coefficient and the pressure losses are calculated and is summarized in Table 12.

Table 12. Summary of PCHE design

Parameter	Value	Unit
Pressure drop primary side (ΔP_p)	1.24E+05	Pa
Pressure drop secondary side (ΔP_c)	1.65E+06	Pa
Overall heat transfer coefficient (U)	2584.6	W/m^{2}K

4.3.3 Pump Position

Position of the pump in the reactor system plays a significant role as it dictates the flow characteristics in the reactor system. As part of design optimization, the impact of pump's position in the primary loop is studied. Two pump positions are chosen for this study, one before the heat exchanger and one after the heat exchanger. When the pump is placed after the heat exchanger, it copes for the pressure lost during the heat transfer to the secondary side and if the pressure lost isn't too high then the pump can be placed before the heat exchanger too. The governing metric for the pump position will be flow distribution, average and maximum velocity of the fuel salt in the system.

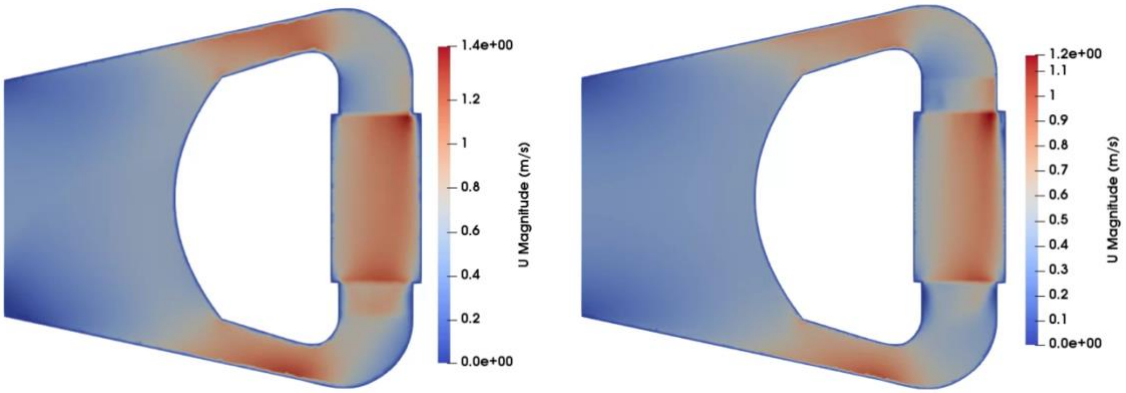


Figure 45. Flow distribution in the reactor system (a) Pump position after the HEX (b) Pump position before HEX.

The flow distribution in the reactor system for different pump positions are shown in Figure 45. The velocity contours are similar for both the pump positions whereas the velocity magnitudes are quite different. From Table 13 it is clear that the average velocity and maximum velocity of the salt is higher in the reactor system when the pump is placed after the HEX. Achieving higher velocities inside the reactor system for the same pump momentum source is ideal as it minimizes the risk of formation of recirculation zones leading to better thermal efficiency and performance during steady state and accident scenarios.

Table 13. Summary of velocity parameter for different pump positions

Position	Average Velocity	Maximum Velocity	Units
Pump before HEX	4.80E-01	1.21E+00	m/s
Pump after HEX	6.21E-01	1.47E+00	m/s

4.4 Optimized Primary Loop Model and Specifications

The reactor core geometry was optimized based on the constraints imposed from the neutronics standpoint. Whereas for the primary loop design, sizing of heat exchanger and the position of pump in the primary loop was done. Based on neutronics and performance of primary

systems, an optimized primary loop model is proposed with the reactor core height as 4.1 m and the radius of the core as 3 m. A 2-D slice along with the respective dimensions of the LFMSR primary loop model is shown in Figure 46. The resulting LFMSR design is a 2400 MW_{th} reactor with sixteen primary loops, each containing a HEX and a pump. The length of the HEX is decided based on the optimization study done earlier and the connecting pipes are added to ensure a continuous flow of fuel salt from the core into the primary loop components and back.

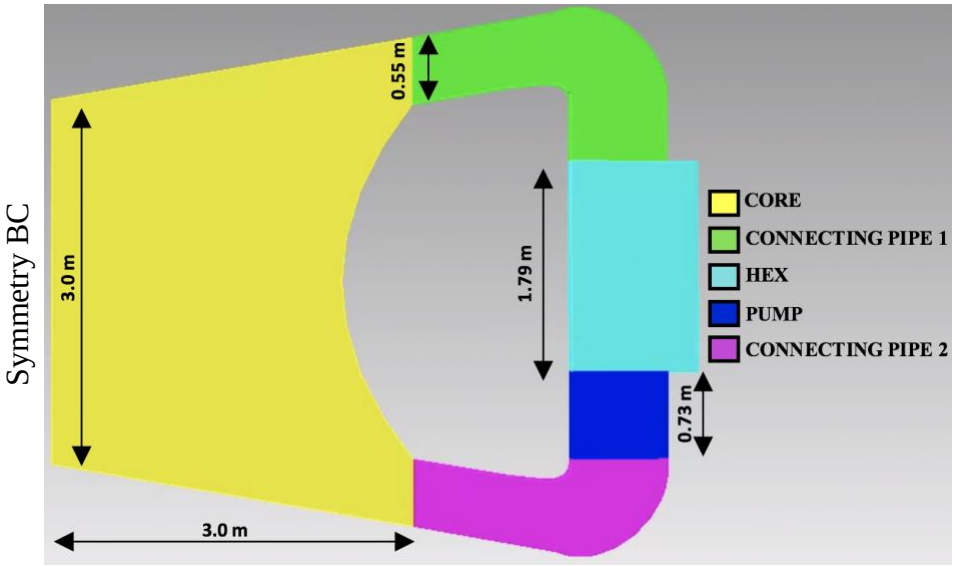


Figure 46. LFMSR Primary Loop Geometry

4.4.1 2-D GeN-Foam LFMSR Model

A simplified 2-D LFMSR model is built in SOLIDWORKS and divided into different cell zones as shown in Figure 46 using Trelis (https://www.coreform.com). The idea behind defining cell zones is that it enables the user to define different initial and boundary condition for each cell zone. The boundary conditions used for the simulation are given in Table 14. To simplify the modeling process and reduce computational time only one-sixteenth of the core and a single primary loop is modelled. The geometry is defined in an x-z plane with the gravity term being

accounted for along the z direction. Since all problems are generally set up as 3-D in GeN-Foam, a special empty boundary condition is used along the third dimension which in our case is along the y plane where no solution is required thereby reducing the problem to a 2-D study. A *symmetryPlane* boundary condition is used along the axial plane of the geometry as shown in Figure 46 and wall boundary condition is used for the rest of the edges of the geometry. Vacuum boundary condition is used to model the neutron flux at the system walls as it is more practical than zero value boundary condition. The fuel salt composition used for the analysis is same as the one in Table 8 and the thermophysical properties chosen are same as given in Table 9.

Table 14. Boundary conditions for the simulation

Parameter	Boundary Condition	Units
Neutron Flux	Vacuum	$m^{-2}s^{-1}$
Pressure	Zero gradient	Pa
Temperature	Zero gradient	K
Velocity	No slip	m/s
Turbulence Dissipation	Wall function	m^2/s^3
Turbulence Kinetic Energy	Wall function	m^2/s^2

Since it is not computationally time efficient to solve a detailed primary loop system, porosity is turned on to simplify the behaviours of HEX and pump. In this analysis, porosity of the HEX is assumed to be 50% of the total volume and the pump porosity to be 20% of the total volume. The pump is simulated as an additional momentum source driving the fuel salt flow and the parameters used to set up the primary loop system are given in Table 15 for reference. The $k-\in$ turbulence model boundary conditions are wall functions defined in OpenFOAM libraries to

approximate and bridge the gradients at the walls. Linear dependence of the salt density on temperature is also assumed.

The Monte Carlo continuous energy code Serpent is used to generate the six-energy groups cross-sections for the neutron diffusion sub-solver. The cross-sections are generated based on the fuel salt composition given in Table 8 using JEFF-3.1.1 nuclear data library. This particular library is chosen as it has updated entries for Pu-239 and Cl-37 isotopes and it includes 8 groups of delayed neutron precursors that is validated for molten salt applications (Brovchenko et al., 2019, 2013; Aufiero et al., 2014; Li et al., 2015). The energy group boundaries are given in Table 16 and are based on the study in (Fiorina et al., 2014). The total delayed neutron fraction $\beta_{eff_{tot}}$ calculated by Serpent assuming static fuel salt is 4.46E-03.

Table 15. Design parameters for the primary loop

Parameter	Value	Units
Power (Q)	2.4E+09	W
Overall heat transfer coefficient (U)	2.5E+03	W/m^2K
Heat exchanger porosity (γ_{hex})	5.0E-01	-
Pump porosity (γ_{hex})	2.0E-01	-
Secondary side coolant temperature (T_c)	8.00E+02	K
Pump momentum source	2.00E+04	Pa

The parameterized cross-sections are generated using a Serpent infinite medium model and postprocessed using a MATLAB script to prepare the cross-sections for GeN-Foam. The spatial temperature distribution for the 2-D model is shown in Figure 47. A more detailed study of the neutronic parameters is done for the 3-D geometry in Section 5.1.

Table 16. Energy groups boundaries

Energy group #	Upper energy boundary (MeV)
1	1.500E+01
2	2.231E+00
3	4.979E-01
4	2.479E-02
5	5.531E-03
6	7.485E-04

The parameterized cross-sections are generated using a Serpent infinite medium model and postprocessed using a MATLAB script to prepare the cross-sections for GeN-Foam. The spatial temperature distribution for the 2-D model is shown in Figure 47. A more detailed study of the neutronic parameters is done for the 3-D geometry in Section 5.1.

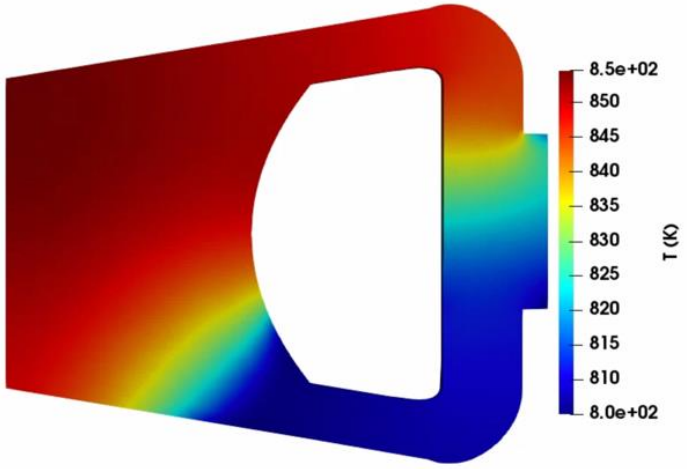


Figure 47. Temperature distribution in the primary loop

The temperature is higher towards the core outlet and upper core wall as the fuel salt in that location has generated and absorbed more energy on its way to the outlet whereas the

temperature of the fuel salt is lower at the core inlet as it is entering after heat transfer in the heat exchanger. The velocity of the fuel salt is higher at the core inlet and outlet due to the change in cross-sectional area as the fuel salt flows in and out of the core.

4.4.2 SAM LFMSR Model

A simplified LFMSR model is built in SAM based on the results available from the 2-D GeN-Foam simulation. The LFMSR model in SAM is a combination of zero-dimensional (0-D) and 1-D components that construct the various flow paths. The model primarily consists of three main components, reactor core, heat exchanger and pump which are connected by pipes as shown in Figure 48. A constant temperature boundary condition is used to model the secondary side of the intermediate heat exchanger (IHX) in both the GeN-Foam model and the SAM model. Also, an additional pipe component (Pipe 5) is introduced in the SAM model as recommended by the code developers to help towards faster convergence of results. A pressure boundary condition is given at the outlet.

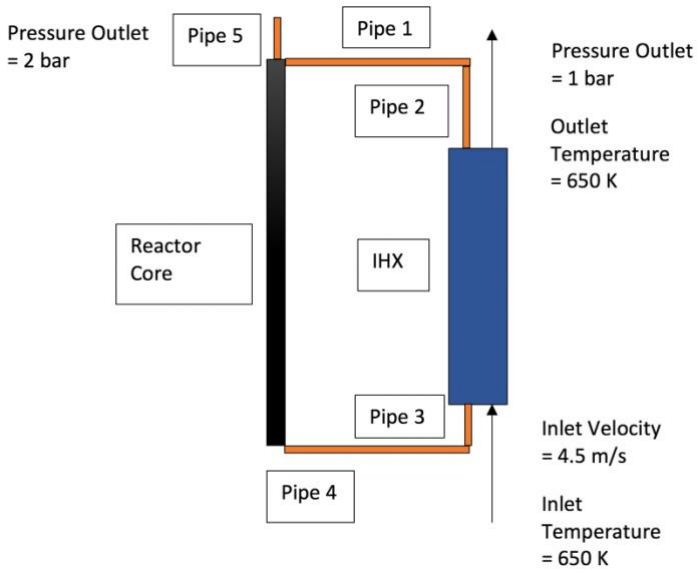


Figure 48. The 1-D LFMSR SAM model

The point kinetics module implemented in SAM simplifies the core power calculation during transients. In this work, the point kinetics parameters available from the Serpent run for static fuel was used. The only feedback available in SAM is the fuel/coolant density reactivity feedback and is calculated with Equation 31. During a transient, SAM calculates the change in reactivity with Equation 32.

$$R_{CD}(t) = \sum_{n}^{N} * \alpha_c^n * \rho_c^n(t) * V_c^n \tag{31}$$

$$\Delta R_{CD}(t) = R_{CD}(0) - R_{CD}(t) \tag{32}$$

Where,

R_{CD} is the integrated coolant reactivity at time t in $\frac{\Delta k}{k}$

α_c^n is the fuel/coolant density reactivity coefficient in $\frac{\Delta k}{k}$ per kg of node n

ρ_c^n is the fuel/coolant density in $\frac{kg}{m^3}$ of node n

V_c^n is the coolant volume in m^3 of node n

N is the total node number in the flow channel.

The SAM model is verified by comparing the steady state velocity and temperature values at the inlet and outlet of the reactor core and heat exchanger with the GeN-Foam steady state results. Table 17 and Table 18 outline the comparison between the GeN-Foam model and SAM model at steady state conditions.

Small differences in velocity and temperature fields can be attributed to the difference in modeling techniques employed between the two codes. In SAM, the symmetry effects are not modeled, and the small discrepancies can be attributed to the higher modeling fidelity (2-D) of the CFD code.

Table 17. SAM model vs GeN-Foam model – Inlet/Outlet Temperature in Kelvin

Part	GeN-Foam Inlet	SAM Inlet	ΔT	GeN-Foam Inlet	SAM Inlet	ΔT
Reactor Core	807.56	808.30	0.74	850.45	851.37	0.92
IHX	850.45	851.37	0.92	807.56	808.30	0.74

Table 18. SAM model vs GeN-Foam model – Inlet/Outlet Temperature in Kelvin

Part	GeN-Foam Inlet	SAM Inlet	ΔT	GeN-Foam Inlet	SAM Inlet	ΔT
Reactor Core	1.102	1.024	0.078	1.097	1.024	0.073
IHX	0.689	0.624	0.065	0.693	0.611	0.080

Also, GeN-Foam can accurately model the viscous shear stresses and turbulence through appropriate turbulence models whereas in SAM they are approximated through the use of wall friction coefficients. Hence, SAM doesn't capture the boundary layer effects while GeN-Foam does.

4.4.2.1 Unprotected Loss of Flow Transient Results

An Unprotected Loss of Flow (ULOF) transient is chosen to evaluate the LFMSR design. This particular transient is chosen as it can provide information about the safety of the LFMSR design without modeling in detail the reactor vessel or the secondary loop. The LFMSR ULOF transient is simulated by maintaining the reactor at full power and tripping the primary pump. This is modeled in SAM using a pump coast down curve as a function of time and is shown in Figure 49.

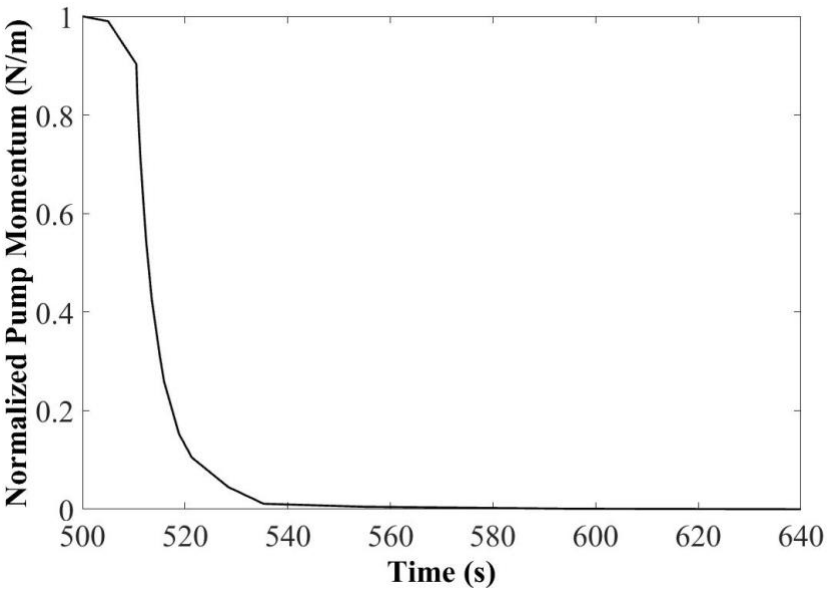


Figure 49. Pump coast down curve for ULOF transient scenario

The two parameters of importance in this study are temperature and velocity. It is crucial that the maximum temperatures do not exceed the boiling temperature (1673 K) of the salt and that the system attains a reasonable natural circulation velocity upon initiating the transient. Figure 50 shows the temperature and velocity response to the pump coast down. As the flow rate reduces, there is a significant increase in temperature which decreases the density of the fuel. The fuel density reactivity feedback is triggered immediately and counteracts this decrease in density by reducing the reactor power and thereby stabilizing the temperature. Similarly, Figure 51 shows the reactor system's capability to obtain reasonable non-zero natural circulation velocity inside the primary loop to manage any severe accidents.

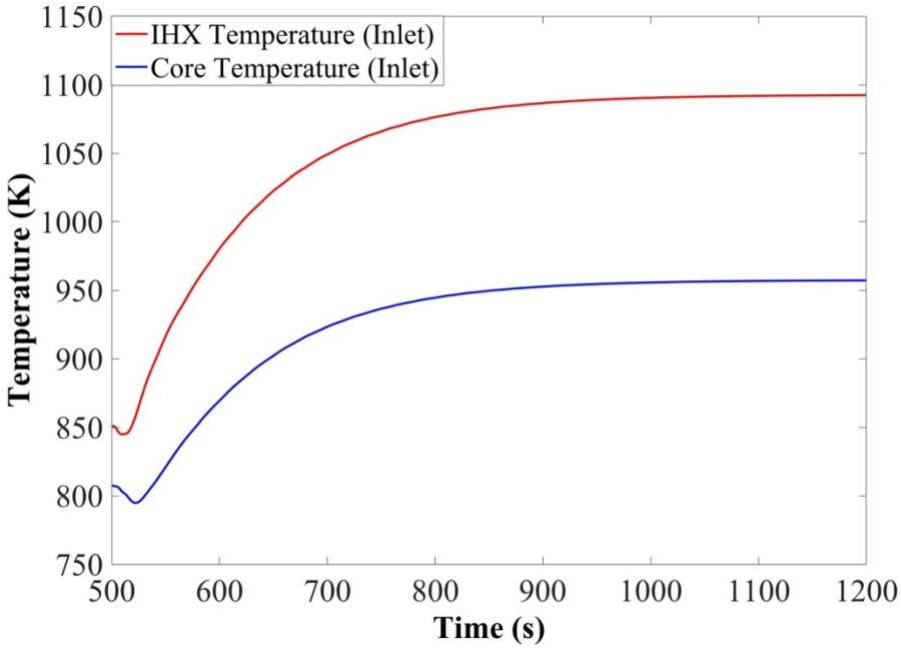


Figure 50. Temperature evolution during ULOF transient

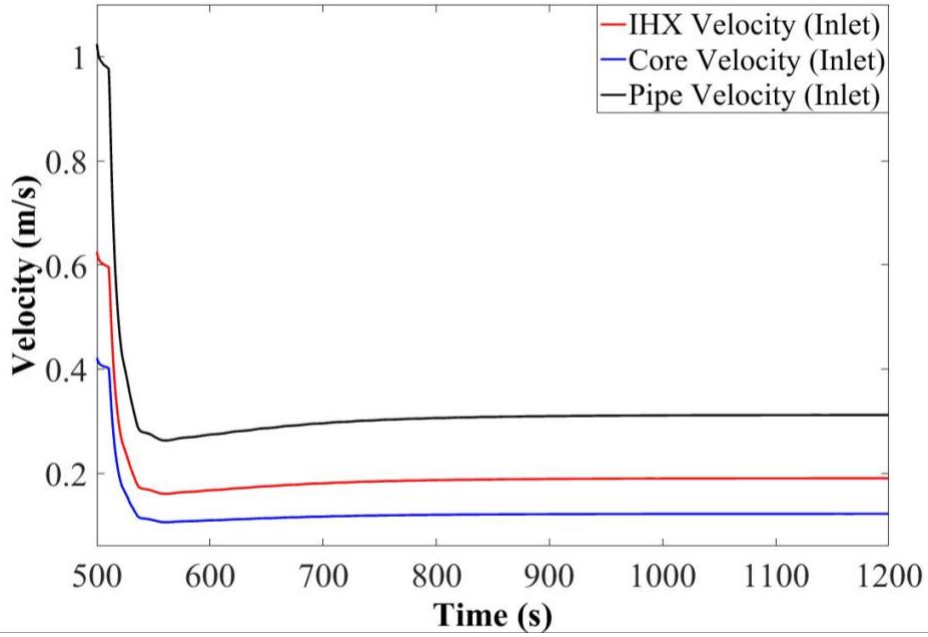


Figure 51. Velocity evolution during ULOF transient

To summarize, the right circular design was relatively simple to model, but its flow paths caused recirculation zones. As a result, a new core geometry design was proposed that can

minimize the presence of recirculation zones and was based on an hourglass shape. Design optimization studies were performed for the newly proposed core design and simplified models of the LFMSR system were built in GeN-Foam and SAM.

CHAPTER 5

Results and Analysis

Once the reactor design has been optimized, a multi-physics 3-D coupled simulation of the LFMSR system is performed using GeN-Foam to obtain steady state results and is shown in Section 5.1. An ULOF transient is also modeled in GeN-Foam to better understand the reactor behavior during transients. From a safety perspective, the design of the reactor system needs to be studied in detail, meaning detailed studies need to be performed in understanding the uncertainties related to the design and also perform sensitivity analysis. In order to perform these studies, uncertainties are propagated through input parameters and statistical analysis is used to quantify the impact of input uncertainties as shown in Section 5.3. Sensitivity analysis was performed in Section 5.4 using the Spearman rank correlations to identify the input variables that have an impact on the output quantities of interest. In Section 5.5, the output dimension space is reduced to enable easier and faster processing of data for future modeling purposes.

5.1 3-D LFMSR Steady State Results

Although 2-D models do present a vast amount of information about the LFMSR model it is merely an approximation of the 3-D model, developed to save computational time. The 2-D model might not always be sufficient to understand the reactor's characteristics completely. To bridge this gap, 3-D models need to be developed. The 3-D LFMSR model is based on the optimized core design and primary loop components from Section 4.4. The fuel salt composition is same as the one given in Table I and the thermophysical property of the salt used for this simulation is given in Table II. Similar to the 2-D model study, only one-sixteenth of the core along with a primary loop is modeled. The 3-D LFMSR primary loop model is built in

SOLIDWORKS and is shown in Figure 52. The geometry is then imported into Trelis to generate a mesh to solve for the different physics. Upon meshing, a mesh convergence study is performed to investigate the possibility of using a mesh with fewer elements, while maintaining reasonable accuracy and lower computational cost. The velocity is studied along the A-A axis for the different meshes and is shown in Figure 53.

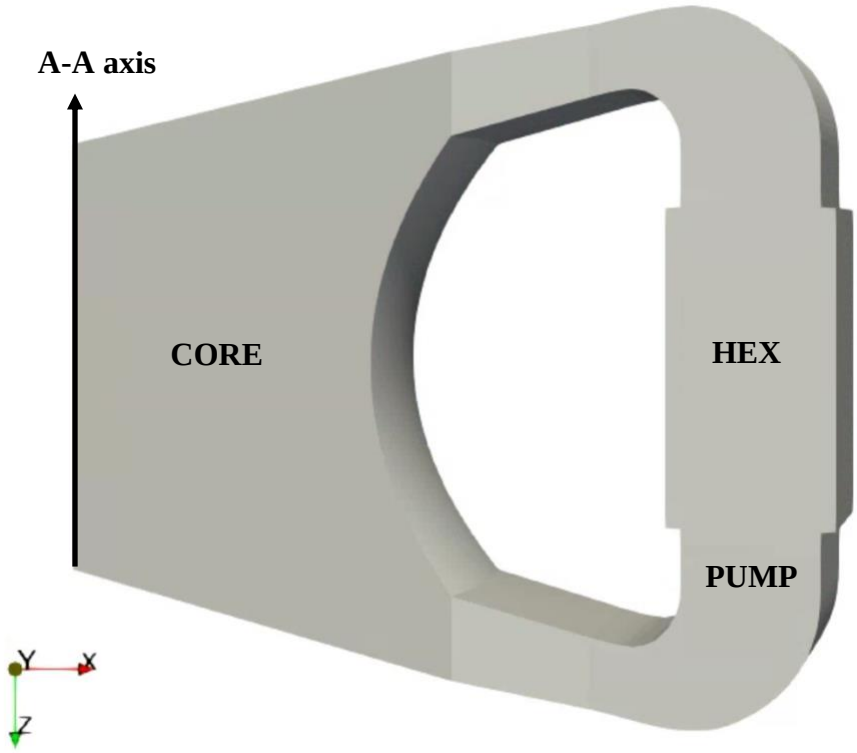


Figure 52. 3-D CAD model of LFMSR

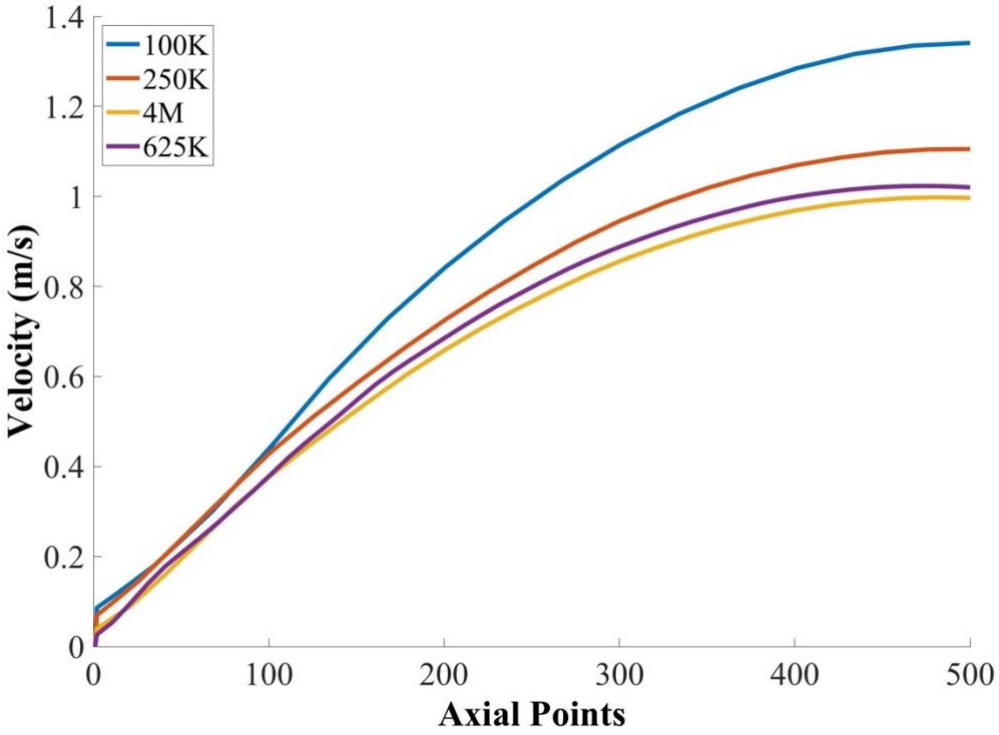


Figure 53. Mesh sensitivity analysis along the axial direction

Table 19. Summary of mesh sensitivity analysis.

No. of Elements	Avg. Relative Error % (Axial)	Standard Deviation % (Axial)
100,000	24.74	5.30
250,000	7.94	2.72
625,000	3.06	1.07
4,000,000	-	-

From Table 19 it is clear that a mesh with 625,000 elements is able to predict the velocities within a 3% accuracy of a finer mesh with 4,000,000 elements thereby reducing the computation

time by almost three times. So, for the analysis hereon, a mesh with 625,000 elements is used. The cells of the mesh are tetrahedral, and the generated mesh is shown in Figure 54 for reference.

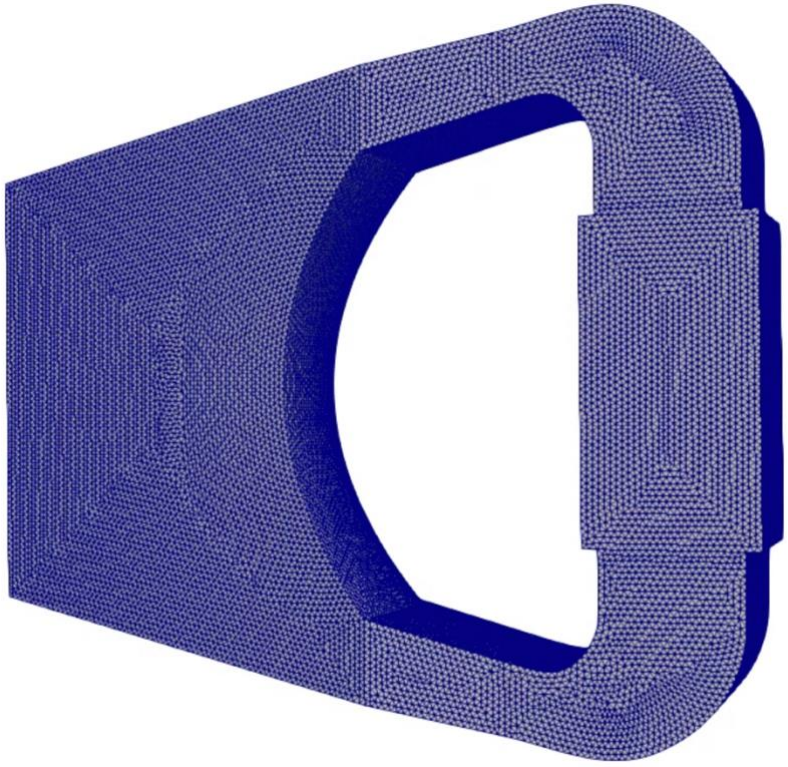


Figure 54. 3-D Spatial Mesh used for LFMSR simulation

The boundary and initial conditions used for the steady state simulation are similar to the 2-D study except for the empty boundary condition and are given in Table 14. Since only one-sixteenth of the core is modelled, symmetry boundary conditions are used for the core lateral surfaces and wall boundary condition for the rest of the cell zones to model all the physics. The wall boundary condition makes sures that the primary loop is treated independently and avoids crossflows. Similar to the 2-D model, vacuum boundary condition is used to model the neutron flux at the system walls as it is more practical than zero value boundary condition. The fuel salt composition used for the analysis is same as the one in Table 8 and the thermophysical properties chosen are same as given in Table 9.

The porous $k-\in$ turbulence model is turned on to approximate the heat exchanger and pump behaviours and a similar procedure is used to generate cross-sections.

The steady state simulation of the LFMSR model is performed in two steps where in the first step, a steady state case with only fluid-dynamics is solved. In the second step, a second steady state simulation is launched solving for neutronics and energy equations starting from the results of the first steady state run. For the first steady state case, the velocity residuals and the minimum, maximum and average velocity magnitudes are tracked to check for convergence. From Figure 55 it is clear that there is no change in the maximum and average velocity values after 35s and hence the simulation has converged, and steady state is achieved.

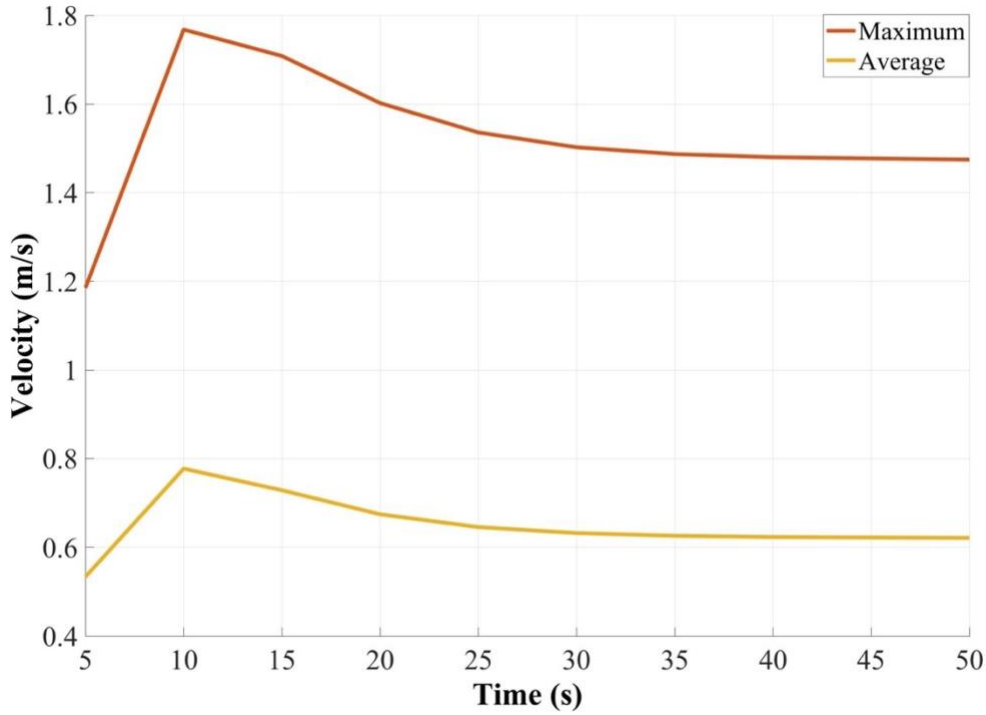


Figure 55. Velocity during steady state calculation

Similarly, the minimum, maximum and average temperature and total neutron flux values are tracked for the second steady state case. From Figure 56 it is observed that that there is no

change in the values after 100s, hence the simulation has achieved steady state. The spatial velocity and pressure distribution for the 3-D reactor system is shown in Figure 57 (a) and (b). The pressure profile is higher at the bottom because of the salt static pressure and the pump momentum head. A noticeable difference in the velocity contours can be observed between a clear flow region and a porous flow region. The spatial temperature distribution is shown in Figure 58 (a) along with the density profile in Figure 58 (b). Linear dependence of density on temperature can be observed. Similar to the 2-D results, the temperature is higher near the core outlet and the upper wall and lower at the core inlet.

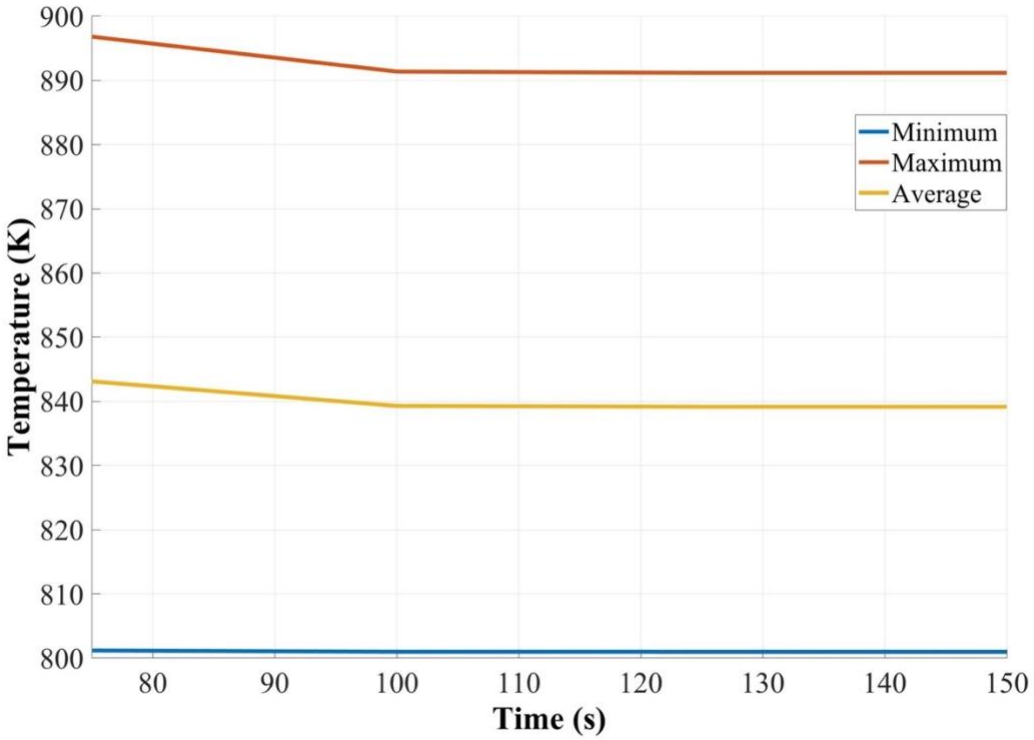
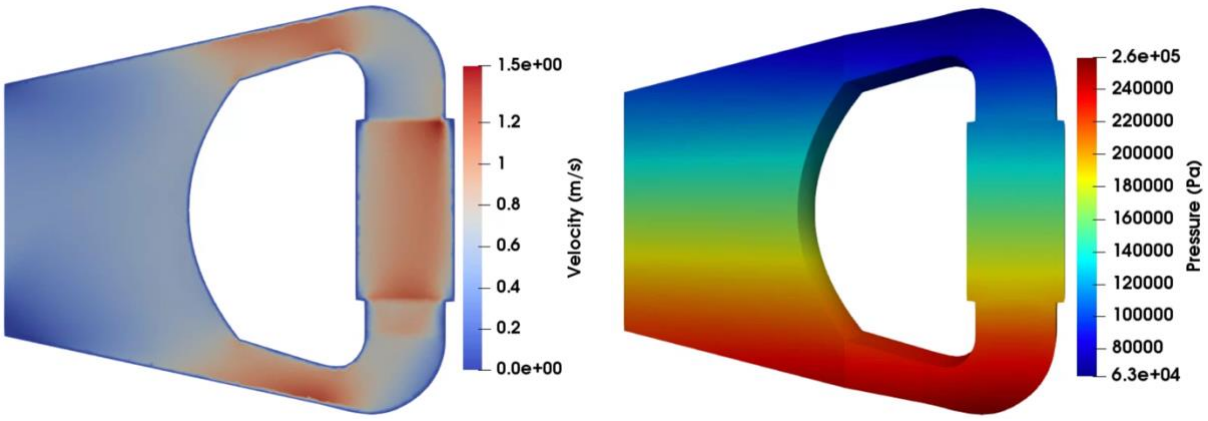
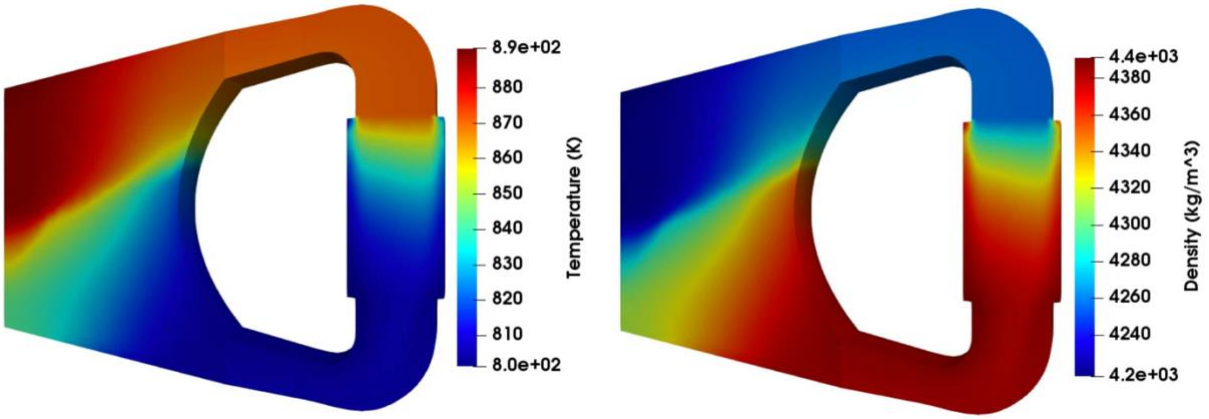


Figure 56. Temperature during steady state calculation



(a) (b)

Figure 57. Thermal hydraulic parameters: (a) Velocity (b) Pressure



(a) (b)

Figure 58. Thermal hydraulic parameters: (a) Temperature (b) Density

The neutron flux in the six groups is shown in Figure 59. The primary loop components have lower significance for the neutron kinetics calculation and hence the neutron flux is smaller by several orders of magnitude compared to the core region. The neutron flux is concentrated in the energy groups 2, 3 and 4, with the maximum in energy group 3. The flux concentration depends on the salt composition and the energy boundaries used. Majority of prompt fission neutrons of U-235, U-238 and Pu-239 isotopes appear in the 0.7 MeV energy range, while the average prompt

neutron energy is around 2.2 MeV (Neudecker et al., 2018; Desai et al., 2015). Also, for the molten salt mixture in the energy range given by three energy groups, lesser absorptions occur as a result more neutron population (Altahan et al., 2020). The precursors distributions for both group 1 and group 8 are shown in Figure 60. Since group 1 has the highest half-life (T1/2 = 55.6 s), it does not decay immediately and is convected by the fuel salt from the core into the heat exchanger, pump, and then back into the core again where it accumulates near the outlet in the upper region. The group 1 precursors can live long enough to move through the whole system many times and their fraction is low (β_1 = 6.186E−05). Therefore, their concentration does not depend anymore on where the precursors are generated, and it becomes relatively homogeneous compared to, e.g., the group 8 precursors concentration. The concentration of group 1 is higher at the core upper region because of the vicinity to the generation area, and the direction of flow. Group 8 having a relatively smaller half-life compared group 1 (T1/2 = 0.2 s) will mostly decay in the place where it is generated and as a result its concentration is proportional to the neutron flux shown in Figure 59. It is important to note that, GeN-Foam lacks a mathematical steady state modeling feature and all the results obtained for the steady simulation are with a time-dependent solver and a system that is not perturbed.

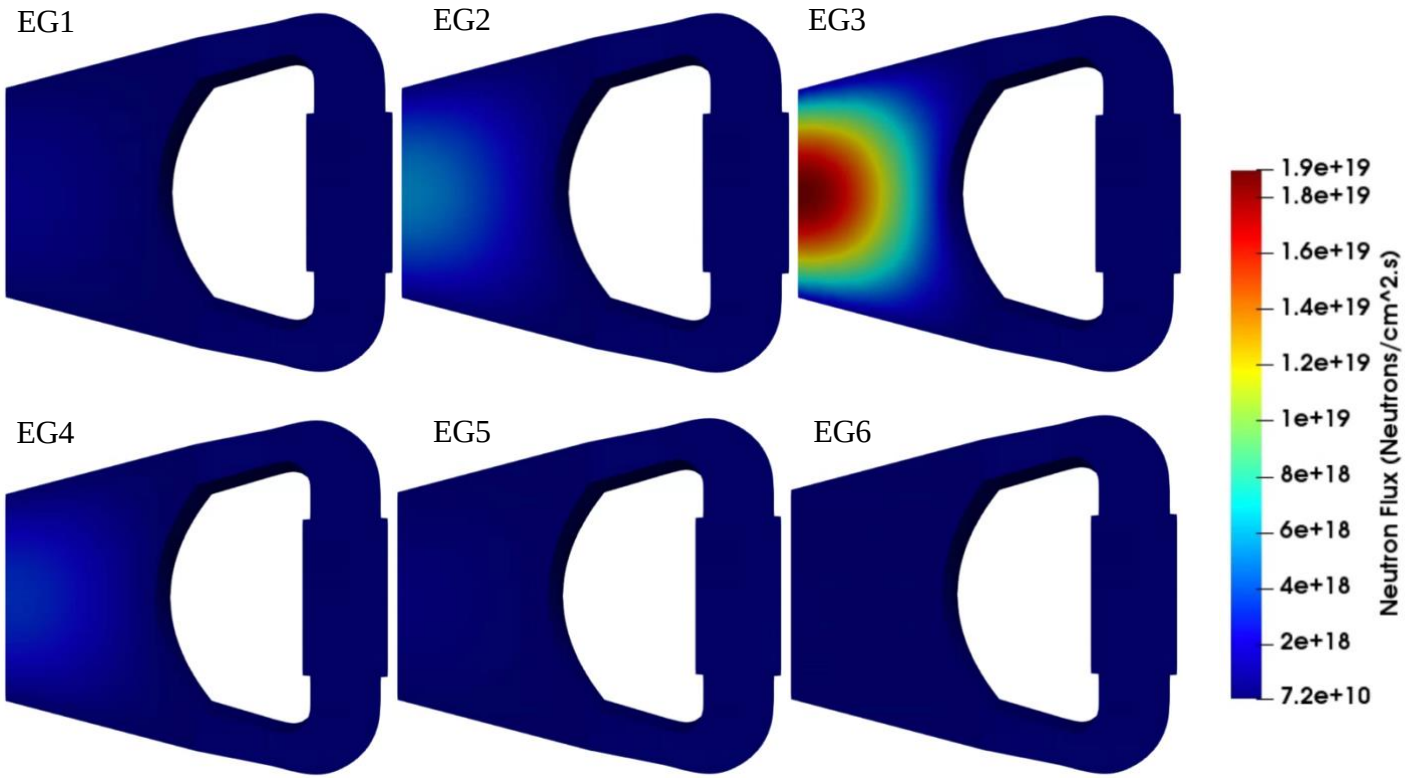
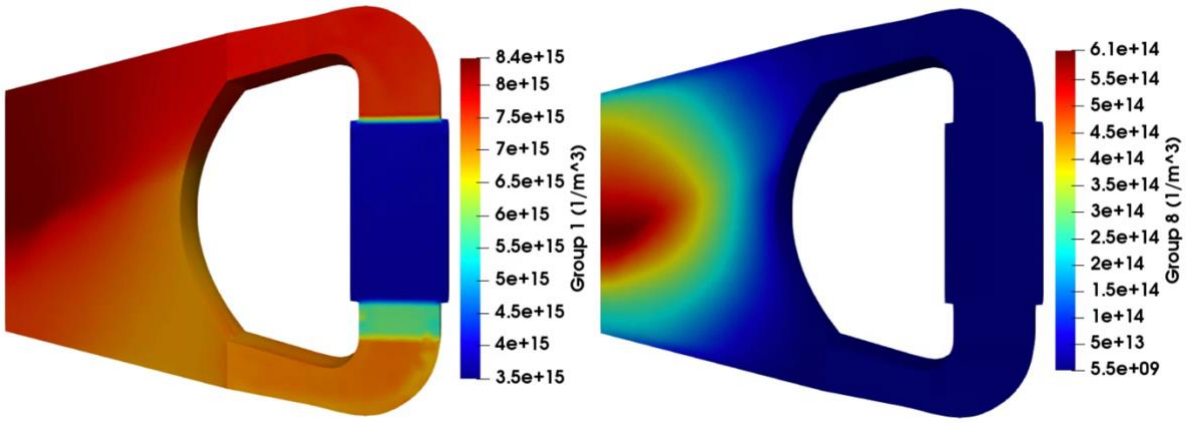


Figure 59. Neutron flux distribution in the primary loop



(a) (b)

Figure 60. Precursor spatial distribution in the primary loop (a) Group 1 (b) Group 8

5.2 Unprotected Loss of Flow Transient Accident Simulation

The ULOF accident scenario may happen when there is a sudden power outage, and the back-up generators fail to start as well. The ULOF accident scenario is modeled as a null transient since GeN-Foam lacks actual transient capabilities. To simulate the following transient in GeN-Foam, the pump momentum source is immediately set to zero after the reactor system reaches steady state. At 150 s, the transient solvers are turned on and both thermal-hydraulic and neutronics solvers are solved in a coupled fashion to study the reactor transient behavior.

The two parameters of importance during the ULOF transient are velocity and temperature. It is crucial that the maximum temperatures do not exceed the boiling temperature (1673 K) of the fuel salt and that the system attains a reasonable natural circulation velocity. Upon initiating the transient, salt velocity decreases rapidly in the first 25 s, after which it stabilizes due to the establishment of a natural circulation regime. The maximum and average fuel salt velocity magnitude evolution is shown in Figure 61.

A constant heat transfer coefficient has been used for modeling the heat exchanger. As the flow rate reduces, there is a significant increase in temperature which decreases the density of the fuel. The temperature feedback is triggered immediately, and a negative reactivity is induced to counteract the increase in temperature thereby reducing the reactor power and stabilizing the fuel salt temperature. The maximum and average fuel salt temperature evolution is shown in Figure 62. The spatial distribution of the velocity and the temperature at the end of the simulated transient (after 300 s) are shown in Figure 63 (a) and (b), respectively.

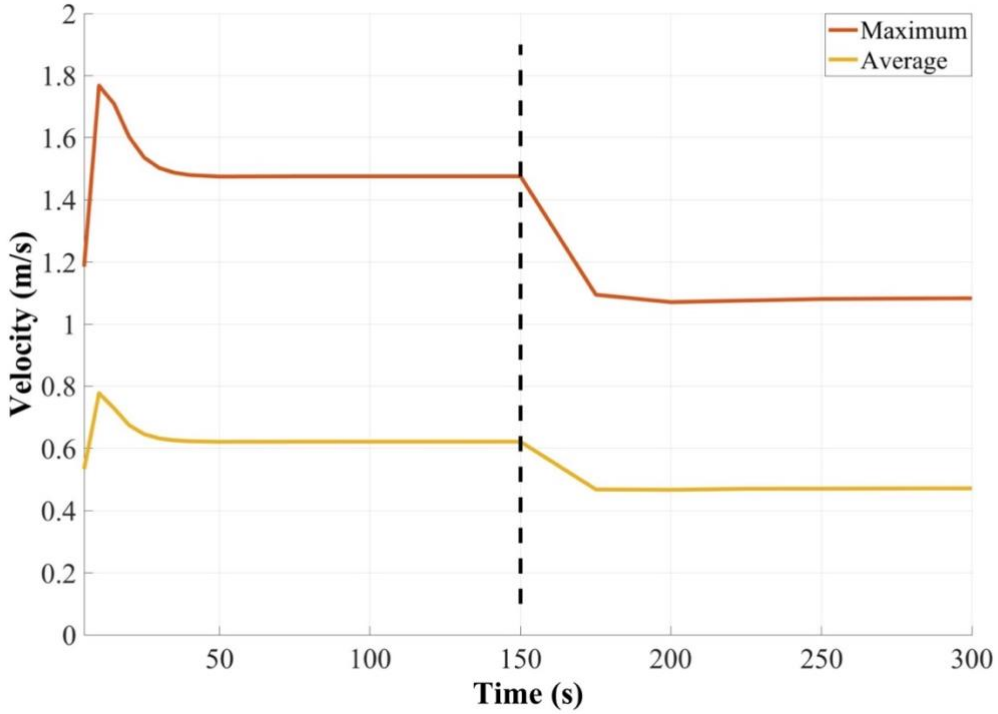


Figure 61. Velocity evolution during ULOF transient

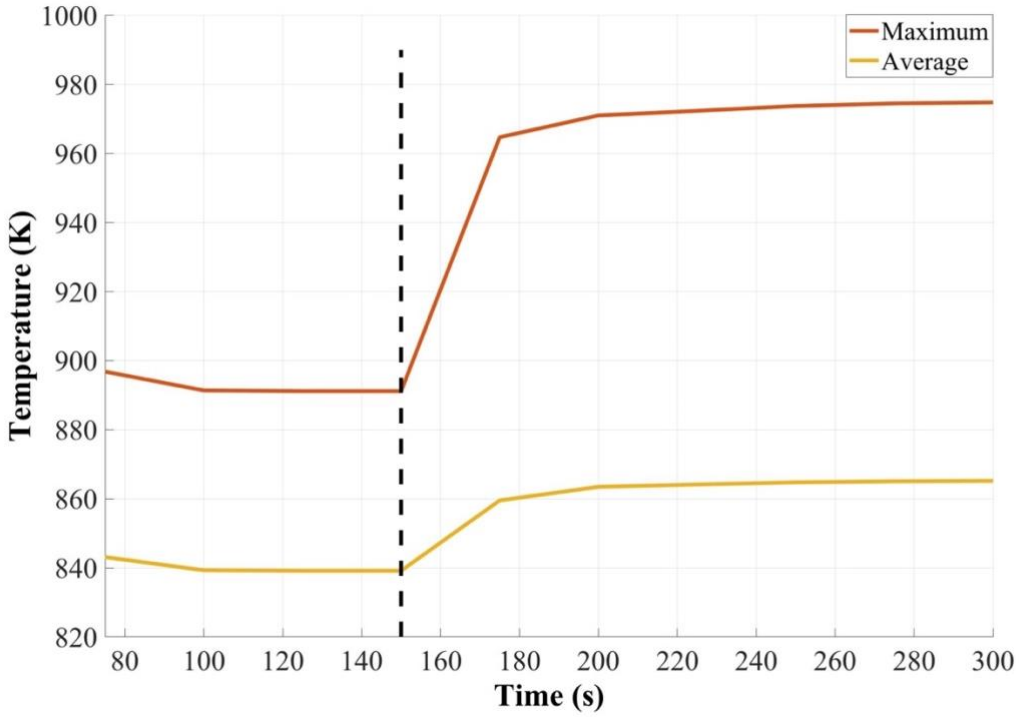
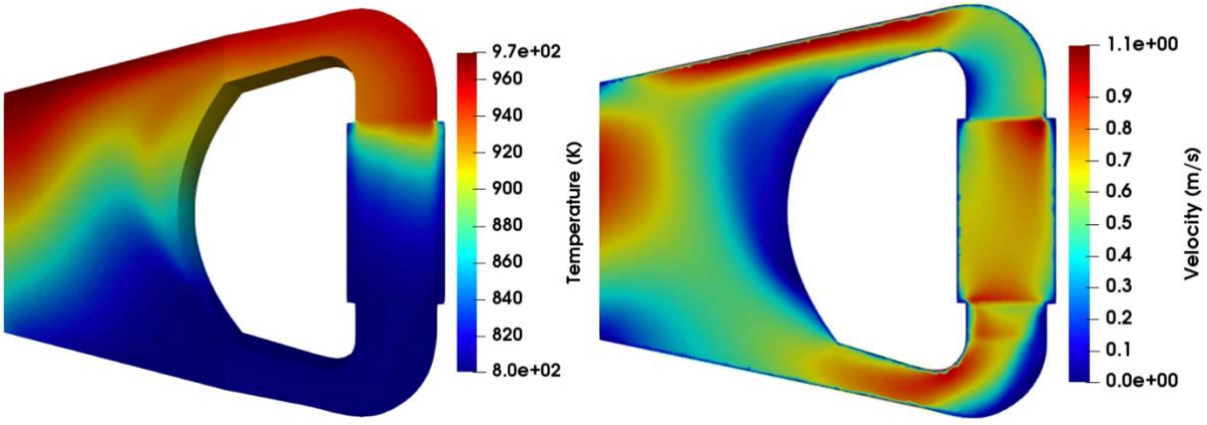


Figure 62. Temperature evolution during ULOF transient



(a) (b)

Figure 63. Thermal hydraulic parameters: (a) Temperature (b) Velocity at the end of ULOF transient

Recirculation zones did not appear during the transient, a reasonable natural circulation velocity is attained and the maximum temperature in the reactor system is below the boiling temperature of the fuel; hence the optimized reactor core design is a good option to eliminate this problem for both steady state and transient simulation.

5.3 Uncertainty Propagation Analysis for Optimized Reactor Design

Detailed studies need to be performed in analyzing the uncertainties related to the design and also perform sensitivity analysis to identify the input variables that have an impact on the output's uncertainty. These studies can subsequently help in reducing the dimensions of the input/output space. Input/design uncertainties are propagated for a ULOF transient simulation through two codes in this study, SAM and GeN-Foam. The ULOF transient in SAM is modeled in the same way as shown in Section 4.4.2.1. Whereas the ULOF transient in GeN-Foam is modeled as shown in Section 5.2. The motivation behind using two codes is that SAM being a system code, the uncertainty propagation studies are by many orders of magnitude faster compared to the GeN-

Foam runs. Any discrepancies observed during the SAM run can be analyzed, corrected, and implemented for a more detailed higher fidelity GeN-Foam uncertainty propagation study.

5.3.1 Input Design Parameters Uncertainties

Nuclear and material data uncertainties need to be considered when modelling a reactor core. The uncertainties in cross-sections are not considered as part of this study due to the lack of information available in the literature for fast spectrum molten salt reactors. The only nuclear data uncertainties considered in this study are for neutron kinetics parameters, delayed neutron fraction (β_i), delayed neutron precursor decay constant (λ_i) for GeN-Foam and the density reactivity feedback (α_c) for SAM in addition to the other two. Sensitivity and uncertainty analysis of neutron kinetic parameters have been investigated in the past. Zukeran et. al performed uncertainty analysis for fast reactors considering three input uncertainty sources, namely, basic delayed neutron data (DND), energy dependence of delayed neutron yield, and fission cross-section uncertainties. The uncertainty in β_{eff} was found to be about 4–5% (Radaideh et al., 2019) (Zukeran, 1999). Similarly, D'Angelo estimated the uncertainties in β_{eff} to be around 5% by accounting for uncertainties in delayed neutron yield and delayed fission spectra. Also, Kodeli et. al employed first-order perturbation theory using SUSD3D to estimate sensitivity coefficients and found that the total uncertainty in the β_{eff} for U-235 benchmarks was around 3%-7% (Radaideh et al., 2019) (Kodeli, 2013). Considering the lack of accurate data available, Santanoceto et. al assumed a 5% uncertainty in their work (Santanoceto et al., 2021). Upon reviewing the literature, a 5% uncertainty is assumed for the neutron kinetic parameters as a starting point in this work. The fuel salt thermophysical properties uncertainties vary between 2%-20% (Romatoski et al., 2019). Statistical sampling is used to perform uncertainty propagation for the reactor system. Each of the thermophysical properties and nuclear data properties are sampled using LHS assuming a normal distribution. The

mean value available from literature is used as part of the normal distribution calculation and the respective 95% confidence level uncertainties propagated are shown in Table 20 (Bhaskar et. al, Romatoski et. al).

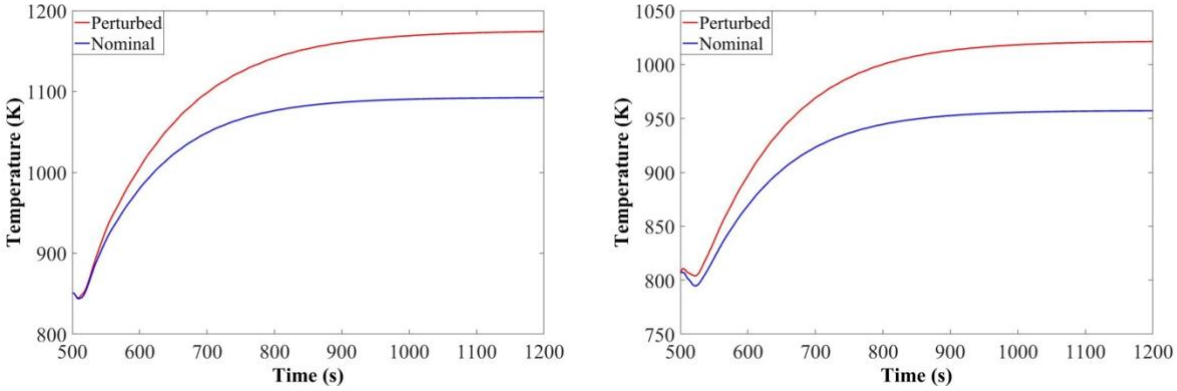
Table 20. Summary of the input design parameters and uncertainties

Parameter	Value	Unit	Distribution	Uncertainty
Density (ρ)	4.20E+03	kg/m^3	Normal	2%
Specific Heat Capacity (C_p)	9.50E+02	J/(kgK)	Normal	20%
Dynamic Viscosity (μ)	2.17E-03	kg/(ms)	Normal	10%
Thermal Expansion Coefficient (β_{th})	4.60E-04	K^{-1}	Normal	15%
Heat Transfer Coefficient	2.5E+03	W/m^2K	Normal	20%
Pump Momentum	20.0E+03	N/m	Normal	20%
Decay Constant (λ)	total		Normal	5%
Delayed Neutron Precursor (β)	total		Normal	5%
Density Reactivity Feedback			Normal	5%

5.3.2 SAM Model Uncertainty Propagation Results

The impact of thermophysical property and nuclear data uncertainties on the optimized reactor system is assessed by randomly sampling the selected input parameters given in Table 20 during ULOF transient. A total of 250 samples are generated using LHS and propagated using Dakota. The quantities of interest (QoI) for this study, salt temperature and velocity, are compared against the safety limit criteria. The reactor design is deemed safe for operation if the nominal and perturbed maximum temperature during both the transients stay below the boiling temperature of the salt (1673 K). Among the 250 samples, the maximum temperature values at the core and heat

exchanger for the perturbed cases are chosen and plotted with the nominal case to understand the temperature range for the uncertainties propagated.



(a) (b)

Figure 64. ULOF transient (a) Nominal and perturbed maximum core temperature (b) Nominal and perturbed maximum heat exchanger temperature.

It is clear from Figure 64 (a) and (b) that the maximum temperatures at both the core and heat exchanger of the perturbed case remain well within the boiling temperature of the salt.

5.3.3 GeN-Foam Model Uncertainty Propagation Results

LFMSR 3-D simulation in GeN-Foam is computationally expensive and requires 128 processors to solve for the physics in a coupled manner. Though Dakota has parallel processing capability, the code failed to efficiently assign processors and parallelly run multiple 3-D LFMSR simulations. As a result, a tool is built using PYTHON to automate the preprocessing, simulation run and postprocessing of every case data. The tool is similar to Dakota in its functionality and more emphasis is put on its postprocessing capability. Since GeN-Foam outputs the results in one particular format, the tool is developed to read through every case file and extract quantities of interest to the user at every single nodal point. One main advantage of this tool is that its applicability to postprocess data efficiently can be extended to other simulations/cases that use GeN-Foam. More information about the tool is given in Appendix C and Appendix D. To

understand the impact of input uncertainty propagation, three analyses are performed namely, scalar analysis, functional analysis and three-dimensional analysis, on the quantities of interest. The quantities of interest are represented by a variable P which is defined in Equation 33.

$$P(m,t,n)\ where\ P\ is\ the\ Quantity\ of\ Interest\ (QoI) \tag{33}$$

$$n = 1,2\ldots N\ where\ N = 100\ and\ n\ is\ the\ case\ number \tag{34}$$

$$m = 1,2\ldots M\ where\ M = 511568\ and\ m\ is\ the\ node\ number \tag{35}$$

$$t = 0, t_1, t_2, t_3 \ldots t_f, where\ t_f\ indicates\ the\ final\ time\ step \tag{36}$$

To post-process the quantities of interest *P(m,t,n)*, it is important to understand a few general definitions such as spatial average and ensemble average. Spatial average is defined as the average of the quantity of interest over the spatial domain. The spatial domain is defined in Equation 35 and the spatial average definition is given in Equation 37. The ensemble average has two definitions, where one is a function of time and the other a function of both time and node number. The first definition is given in Equation 38 and is the average of the spatially averaged quantity of interest with respect to time. Whereas the second definition is the average of the quantities of interest for different cases as a function of node number and time as shown in Equation 39. The standard deviation is calculated as shown in Equation 40. The standard deviation formula is used to calculate all types of averages and P-values, for brevity only one example is shown here.

Spatial Average:

$$\overline{P}(t,n) = \frac{1}{M}\Sigma_{m=1}^{M} P(m,t,n), m\ is\ node\ number \tag{37}$$

Ensemble Average:

$$\hat{\overline{P}}(t) = \frac{1}{N}\Sigma_{n=1}^{N}\overline{P}(t,n) \tag{38}$$

$$\hat{P}(m,t) = \frac{1}{N}\Sigma_{n=1}^{N} P(m,t,n) \tag{39}$$

Standard Deviation:

$$\hat{\sigma}_P = \sqrt{\frac{\left(P - \hat{P}\right)^2}{N-1}} \tag{40}$$

5.3.3.1 Scalar Analysis Results

In the scalar analysis, an Empirical Cumulative Distribution Function (ECDF) plot is used to display results in the order of lowest to highest against their percentiles. The ECDF plot is a step function that helps assess,

- range of the data,
- identify where most values occur and
- percentiles and proportions for data ranges.

To perform a scalar analysis and understand the reactor system behavior better it is important to calculate the ensemble average/statistical mean of the system/spatial average and system/spatial maximum of the quantities of interest over different cases at the final time step (t_f). The first step is to calculate the spatial average and since in this study we are interested only in the final timestep $t = t_f$, the above definition $P(m,t,n)$ is modified accordingly,

$$P_f(m,n) = P\left(m, t = t_f, n\right) \; where \; subscript \; f \; denotes \; the \; final \; timestep$$

Spatial average is calculated as follows,

$$\overline{P}_f(n) = \overline{P}\left(t = t_f, n\right) \tag{41}$$

Ensemble average of spatial average is calculated as follows,

$$\hat{\bar{P}}_f = \frac{1}{N} \Sigma_{n=1}^{N} \overline{P}_f(n) \tag{42}$$

Similarly, the standard deviation is calculated as follows,

$$\hat{\sigma}_{P_f} = \sqrt{\frac{\left(P_f - \hat{\bar{P}}_f\right)^2}{N-1}} \tag{43}$$

The second step is to calculate the spatial maximum and the definition of $P(m,t,n)$ is modified accordingly,

$$P_{f_{max}}(n) = \max\left(P\left(t = t_f, n\right)\right) \tag{44}$$

Ensemble average for spatial maximum is calculated as follows,

$$\hat{P}_{f_{max}} = \frac{1}{N} \Sigma_{n=1}^{N} P_{f_{max}}(n) \tag{45}$$

The standard deviation for the spatial maximum is calculated as follows,

$$\hat{\sigma}_{P_{f_{max}}} = \sqrt{\frac{\left(P_{f_{max}} - \hat{P}_{f_{max}}\right)^2}{N-1}} \tag{46}$$

The two quantities of interest in this scalar analysis are velocity and temperature. The ensemble average of the spatial maximum and spatial average are calculated as shown in Equations 42 and 45. It is clear from Figure 65 and Figure 66 that 99% of the data for the maximum and average temperature lies below 1000 K and 875 K. The temperatures of the system are much below the boiling temperature of the salt thereby not imposing any design safety risks. The respective standard deviation are 10.73 K and 5.025 K.

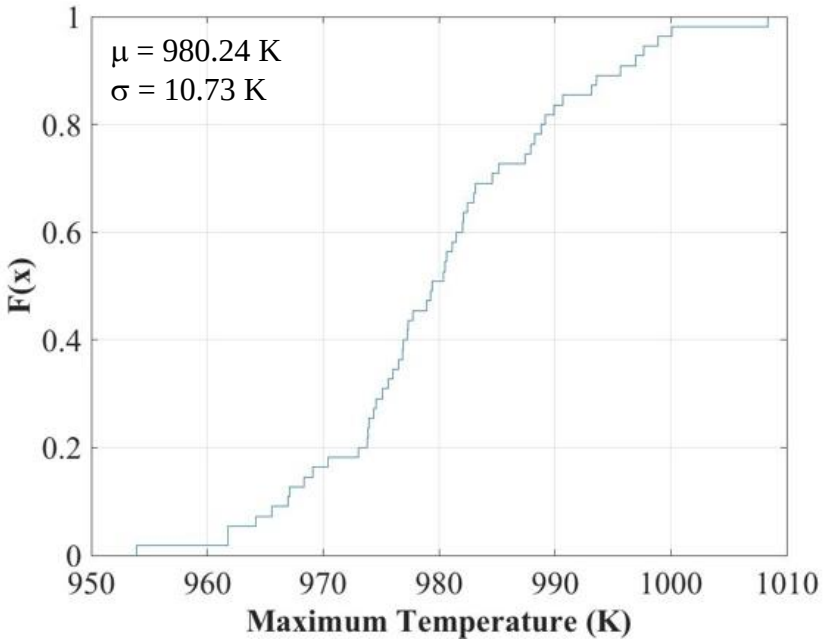


Figure 65. ECDF plot for maximum temperature

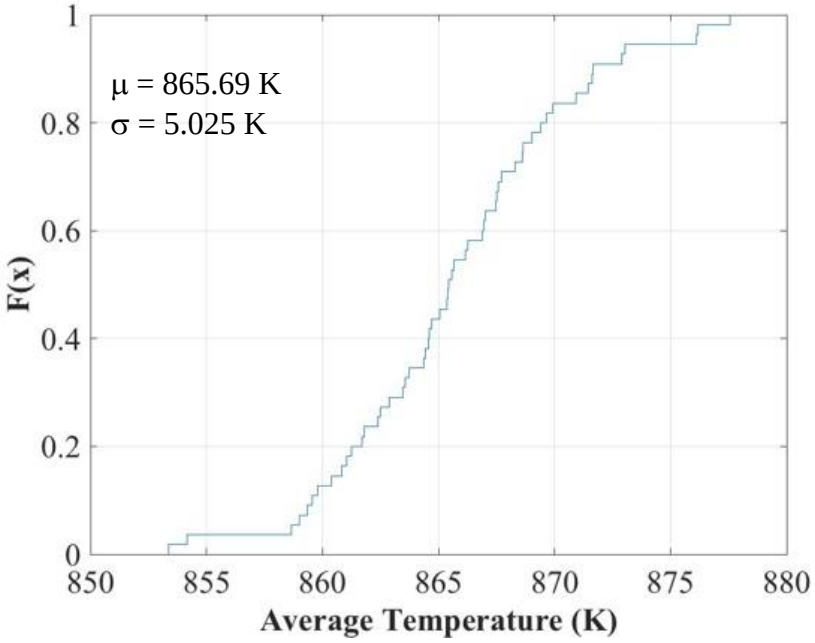


Figure 66. ECDF plot for average temperature

Similarly, Figure 67 and Figure 68 and 37 show that the 99% of data for maximum and average velocity lies below 1.15 m/s and 0.5 m/s. These velocities attained for a natural circulation driven system can manage reactor system during accident scenarios by keeping the salt flowing.

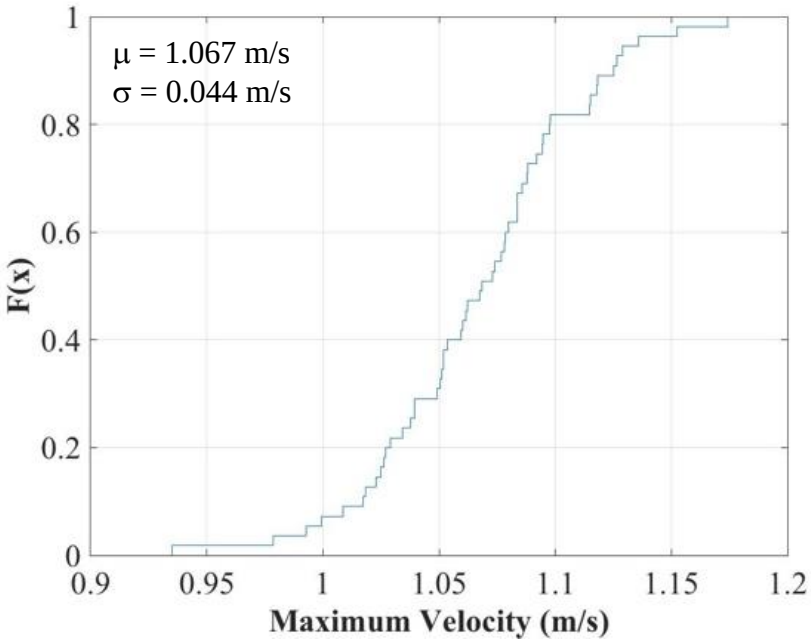


Figure 67. ECDF plot for maximum velocity

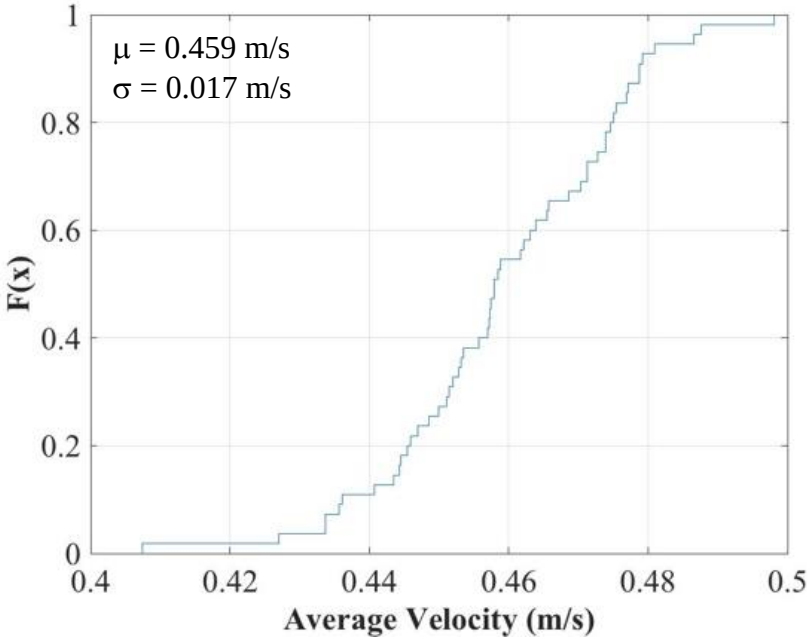


Figure 68. ECDF plot for average velocity

5.3.3.2 Functional Analysis Results

As for any transient it is important to study the evolution of quantities of interest over time. In the functional analysis ensemble average of the spatial average and spatial maximum is plotted

as a function of time. The quantities of interest in this study are temperature and velocity. The time function plot helps assess,

- evolution of the data,
- identify reactor system behavior

Functional analysis is similar to the scalar analysis, the only difference is that the values are calculated at different time intervals rather than just at the last time step. Hence the quantities of interest are a function of time.

Spatial average is calculated as follows,

$$\overline{P}(t,n) = \frac{1}{M} \Sigma_{m=1}^{M} P(m,t,n) \tag{47}$$

Ensemble average for spatial/system average is calculated as follows,

$$\hat{\overline{P}}(t) = \frac{1}{N} \Sigma_{n=1}^{N} \overline{P}(t,n) \tag{48}$$

Standard deviation for the ensemble averaged quantity is calculated as follows,

$$\hat{\overline{\sigma}}_P(t) = \sqrt{\frac{\left(P(t) - \hat{\overline{P}}(t)\right)^2}{N-1}} \tag{49}$$

System maximum is calculated as follows,

$$P_{max}(t,n) = \max\left(P(m,t,n)\right) \tag{50}$$

Ensemble average for system maximum is calculated as follows,

$$\hat{P}_{max}(t) = \frac{1}{N} \Sigma_{n=1}^{N} P_{max}(t,n) \tag{51}$$

Standard deviation for the ensemble averaged system maximum is calculated as follows,

$$\hat{\sigma}_{P_{max}}(t) = \sqrt{\frac{\left(P_{max}(t) - \hat{P}_{max}(t)\right)^2}{N-1}} \tag{52}$$

The two quantities of interest in this functional analysis are velocity and temperature. The evolution of statistical mean maximum and average temperature/velocity in the reactor system is shown in Figure 69 and Figure 71. Initially, before the transient begins, the standard deviation values are low for both temperature and velocity. This is because the input uncertainties propagated don't impact the output parameters by much in a steady state calculation as the flow in the reactor system is dominated by the pump momentum. As soon as the transient is initiated and the pump momentum is brought to zero, other parameters such as density, thermal expansion coefficient, heat transfer coefficient and specific heat start to take control of the flow within the system resulting in larger uncertainties. As a result, the standard deviation of temperature and velocity both increases right after the transient is initiated, when the reactor system is most unstable. As the system attains natural circulation and the velocity stabilizes, the standard deviations come down as well as shown in Figure 70 and Figure 72. In the case of velocity, it is important to note that the relative standard deviation is much higher for the average velocity of the system, and this is attributed to the loss of pump momentum source thereby leading to very low velocities at some regions of the core.

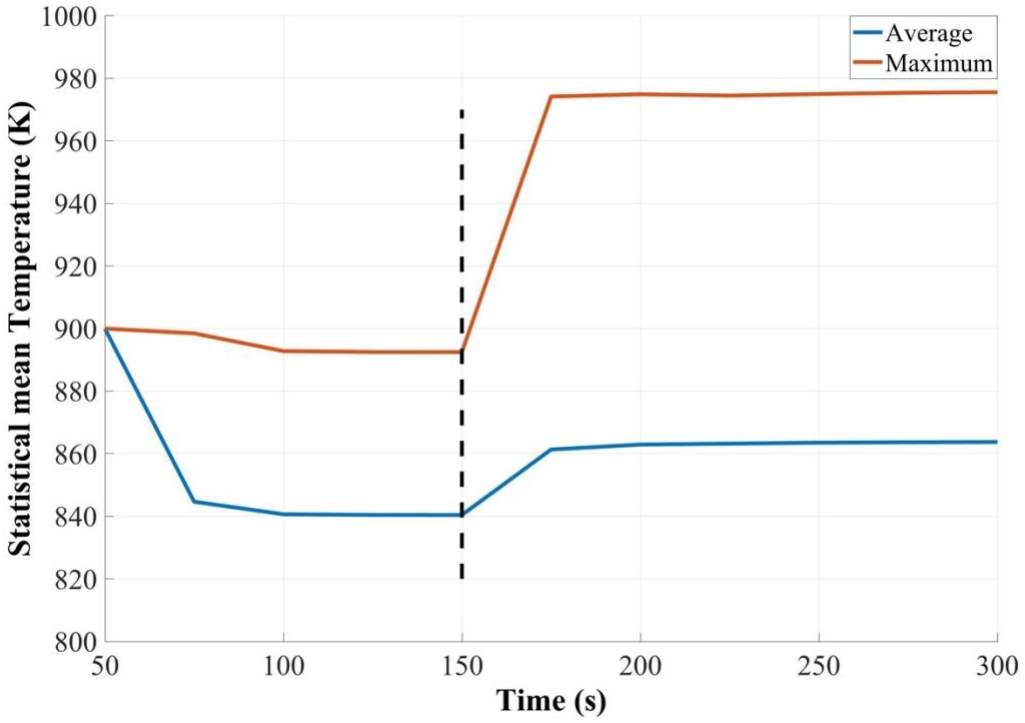


Figure 69. Statistical mean temperature evolution

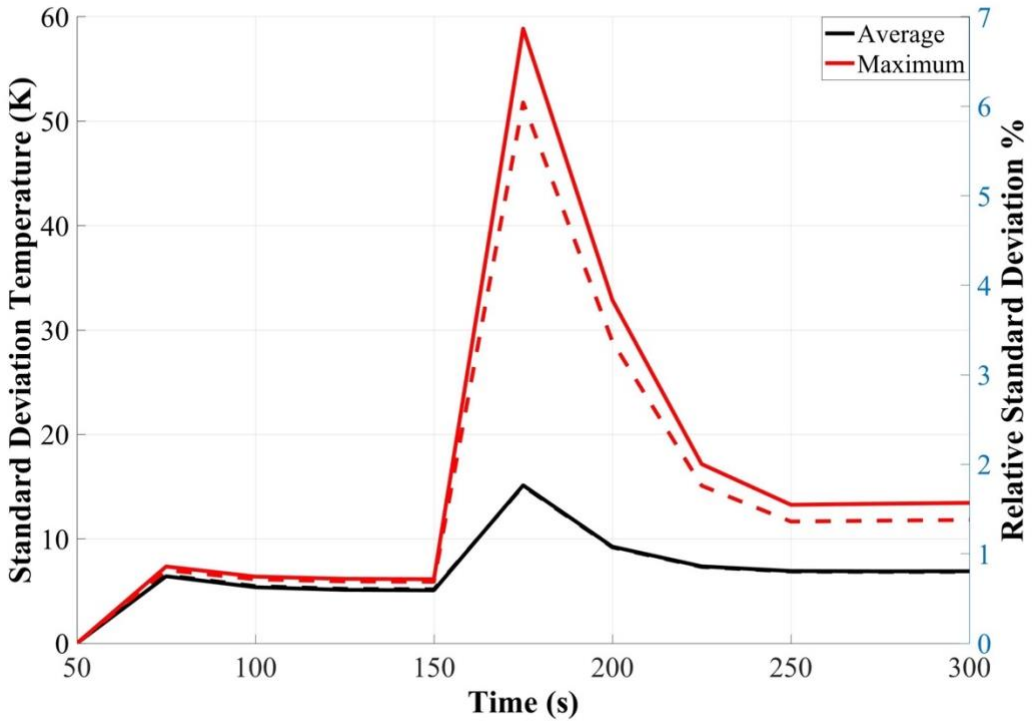


Figure 70. Standard deviation of temperature over time

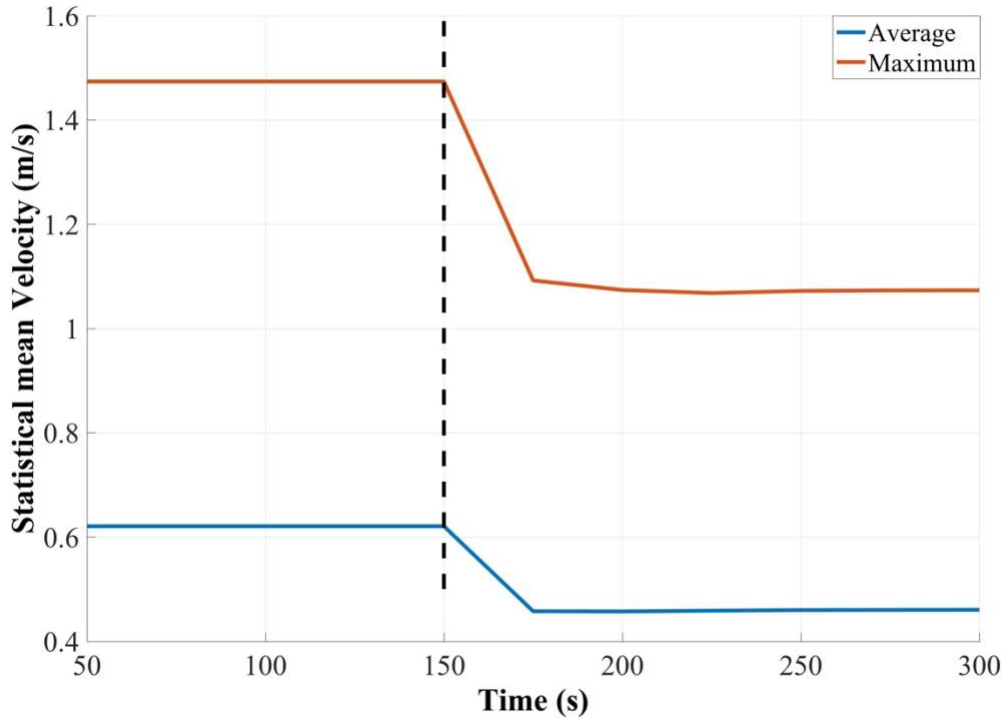


Figure 71. Statistical mean velocity evolution

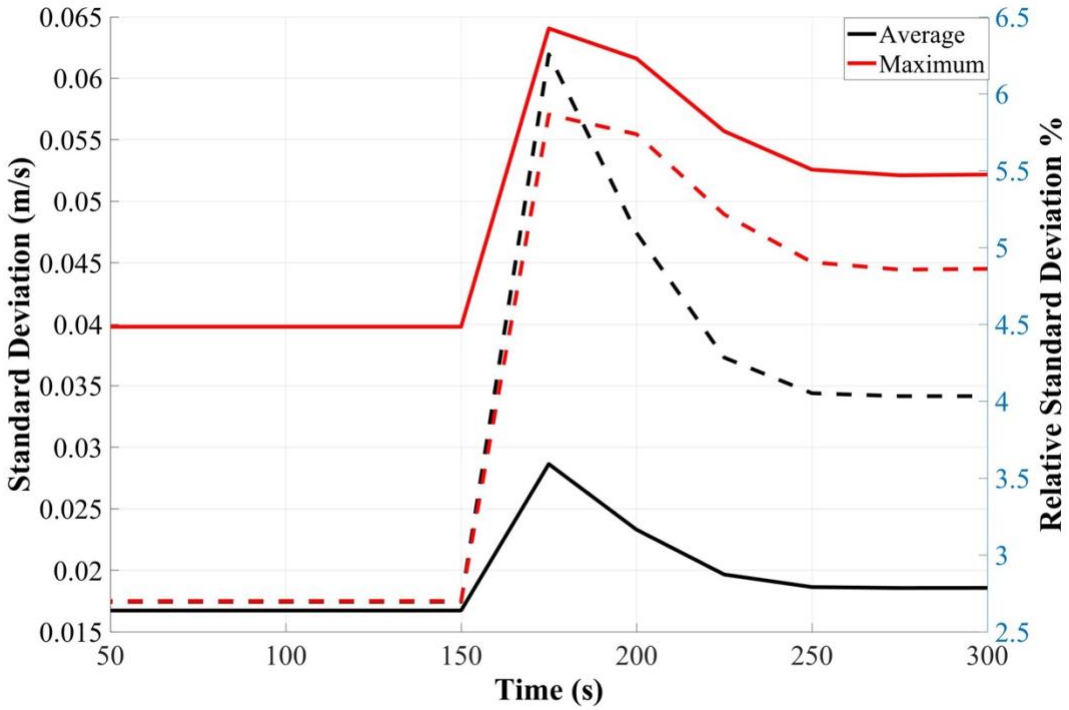


Figure 72. Standard deviation of velocity over time

The relative standard deviations come down as the reactor system stabilizes upon establishing a natural circulation regime. Also, there is a possibility that the rate of convergence of output parameters vary based on the input perturbation and for a fixed time interval there could be large variances observed between the outputs resulting in higher standard deviations.

5.3.3.3 Three-Dimensional Analysis Results

It is clear from the evolution of quantities of interest with respect to time that the maximum/minimum occurs towards the end of transient. Since not all details are captured through the functional analysis over time, the study is extended to every single nodal point in the spatial domain of the reactor system but limited to the last timestep. The 3-D contour helps assess,

- range of the data,
- location of maximum/minimum
- percentile and proportions spatially

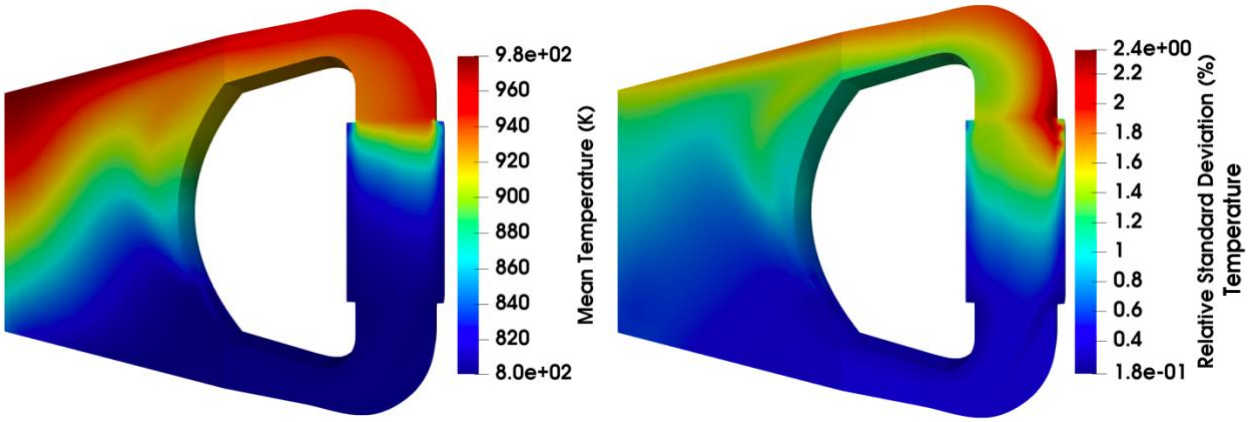
To perform a 3-D analysis and understand the reactor system behavior spatially it is important to calculate the ensemble average/statistical mean of the quantities of interest at every nodal point m, over different cases at the final time step (t_f). Equation 53 shows the ensemble average calculation in a functional form but since only the last timestep is important the equation reduces down to Equation 54.

$$\hat{P}(m,t) = \frac{1}{N} \Sigma_{n=1}^{N} P(m,t,n) \tag{53}$$

$$\hat{P}_f(m) = \frac{1}{N} \Sigma_{n=1}^{N} P_f(m,n) \tag{54}$$

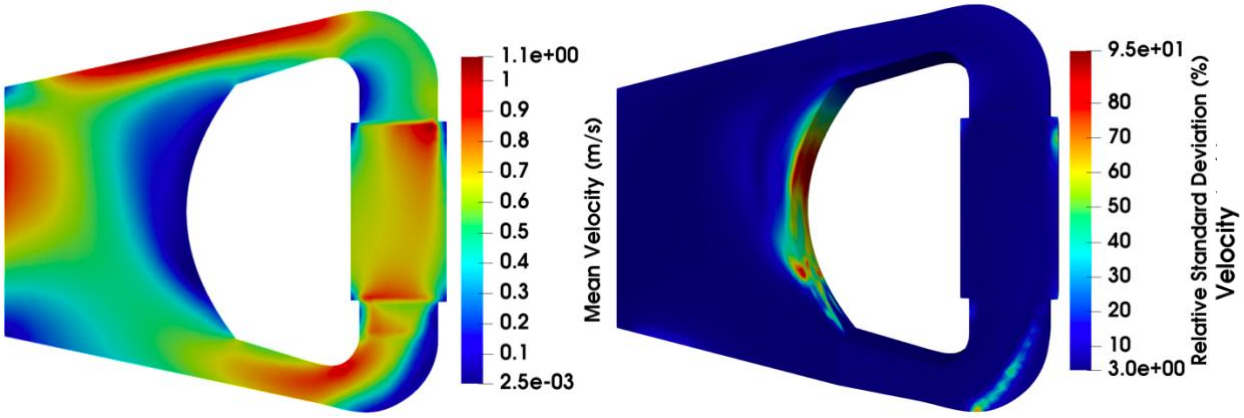
$$\hat{\sigma}_{P_f}(m) = \sqrt{\frac{\left(P_f(m) - \hat{P}_f(m)\right)^2}{N-1}} \tag{55}$$

The two quantities of interest in this 3-D analysis are velocity and temperature. The spatial distribution of the statistical means temperature and velocity in the reactor system is shown in Figure 73 (a) and Figure 74 (a). The standard deviations of temperature and velocity are shown in Figure 73 (b) and Figure 74 (b) respectively.



(a) (b)

Figure 73. Spatial distribution of (a) Temperature (b) Standard deviation



(a) (b)

Figure 74. Spatial distribution of (a) Velocity (b) Standard deviation

As the hot fuel salt enters the HEX, the temperature of the fuel salt depends primarily on the heat transfer coefficient input parameter but as the fuel salt moves through the core other input parameters such as thermal expansion coefficient, specific heat and density come into play and the uncertainties propagated through these parameters start to add up thereby leading to higher uncertainties just before the entrance to the HEX. With regards to velocity, the maximum uncertainty is observed near the curved wall of the reactor where the velocities are very close to zero whereas in other regions the standard deviation are relatively lower by many orders of magnitude.

5.4 Sensitivity Analysis for Optimized Reactor Design

A sensitivity analysis using Spearman Rank Correlation Coefficient (SRCC) is done to nonparametrically measure the strength and direction of association between the two ranked variables i.e., input and output data. The input data considered for this study are the parameters through which uncertainties are propagated and they are, delayed neutron precursor (DNP), decay constant (λ), density (ρ), dynamic viscosity (μ), heat transfer coefficient (HTC), pump momentum (PM), specific heat (C_p) and thermal expansion (β_{th}). The output parameters of interest for this study are temperature and velocity. The sensitivity analysis study is done at three different levels, scalar, functional and three-dimensional level to understand how the correlations change with time and space. At the scalar level, the correlations are computed between the input data and the ensemble averaged system average/maximum of the output quantities of interest computed at the final timestep. The functional sensitivity analysis focuses on how the correlations between the input and output data evolves with time. This functional study is important as it gives more information about the significance of parameters before and after the transient.

In the scalar sensitivity study, the spearman rank correlation coefficients are computed for the ensemble average of system maximum and system average temperature and velocity. The rank correlations for maximum temperature and average temperature are shown in Figure 75 and Figure 76 respectively. Similarly, the rank correlations for maximum and average velocity are shown in Figure 77 and Figure 78. From the scalar SRCC analysis it is clear that, kinetic parameters do not have much of an impact on both the temperature and velocity. Whereas specific heat, thermal expansion and heat transfer coefficient are the most important at the end of transient. As heat is transferred to and from the system, the temperature of the fuel salt changes and this temperature change depends on two factors, number of molecules in the system and the heat capacity of the system. Since the volume of the system is fixed, fuel salt temperature strongly depends on the specific heat capacity of the salt. As specific heat capacity increases/decreases the fuel salt temperature decreases/increases and this strong negative correlation is also verified here. Upon initiating the ULOF transient, as the temperature of the system increases, the thermal expansion of fluid causes an increase in the density difference between the cold and hot sections which leads to the increase in driving head thereby increasing fuel salt velocity. Since it is now a density- driven system it impacts the maximum and average temperature of the core as shown in Figure 75 and Figure 76. The SRCC of maximum temperature and average velocity are very similar to the average temperature and maximum velocity and for brevity they aren’t discussed in detail.

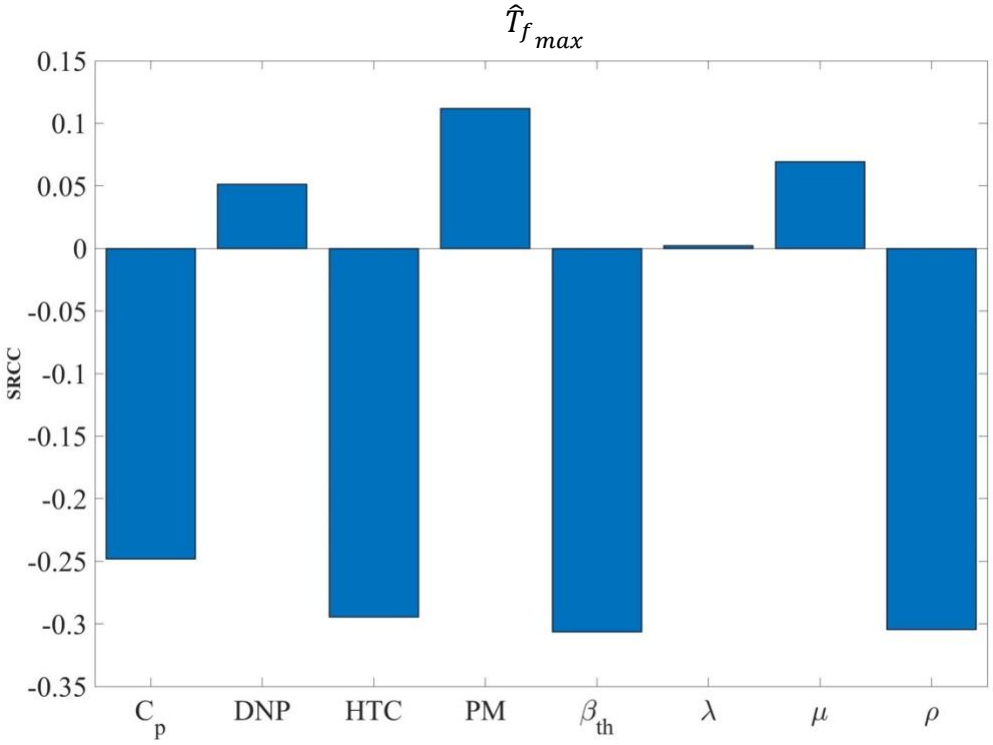


Figure 75. SRCC for system maximum temperature

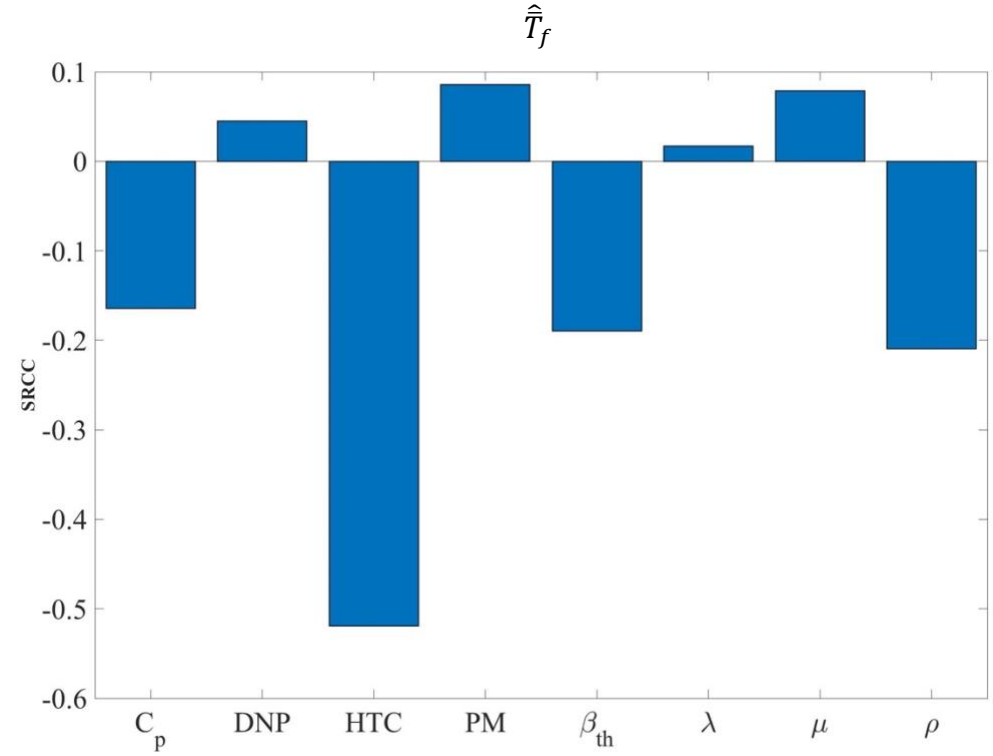


Figure 76. SRCC for system average temperature

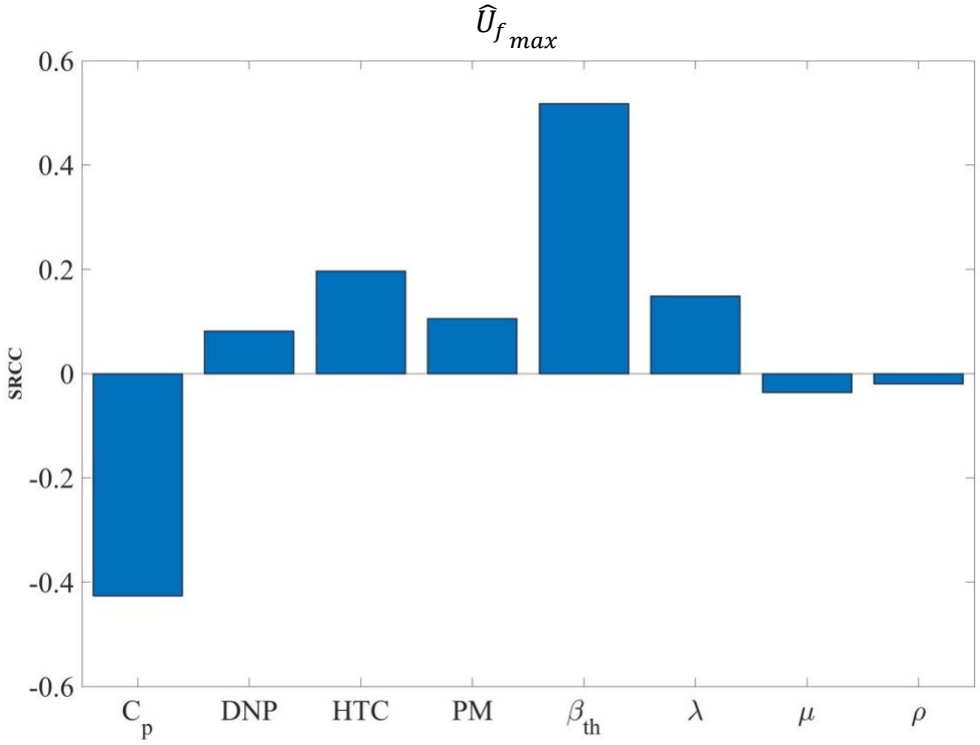


Figure 77. SRCC for system maximum velocity

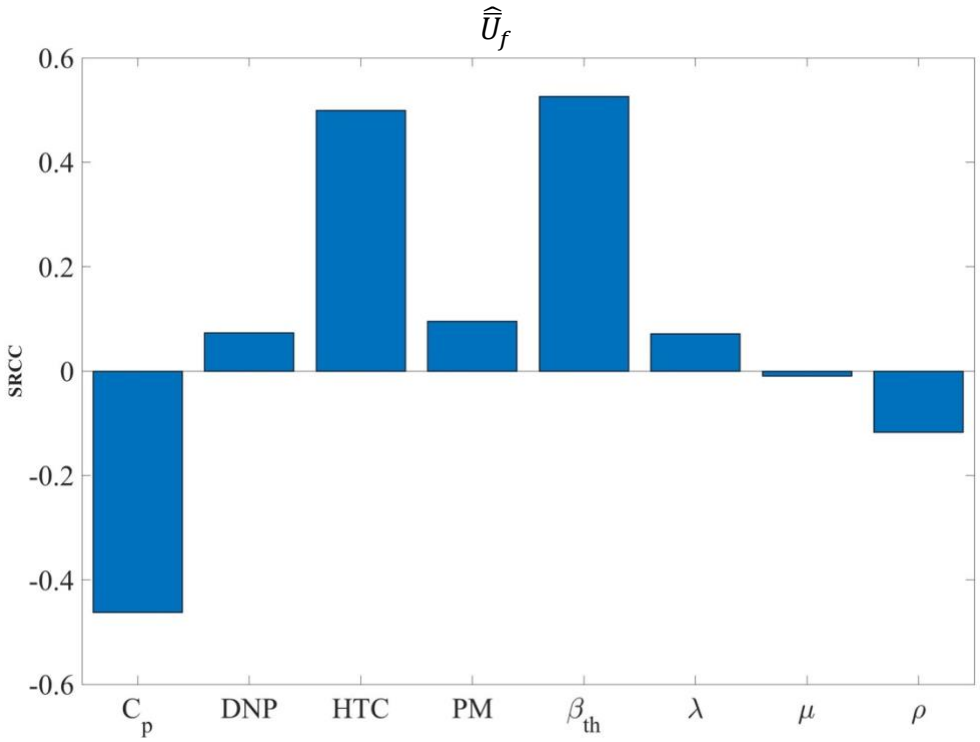


Figure 78. SRCC for system average velocity

The functional sensitivity analysis is similar to the scalar analysis with only one exception, the SRCCs are computed not just at end of the transient but at every timestep. The rank correlations for maximum temperature and average temperature are shown in Figure 79 and Figure 80 respectively. Similarly, the rank correlations for maximum and average velocity are shown in Figure 81 and Figure 82.

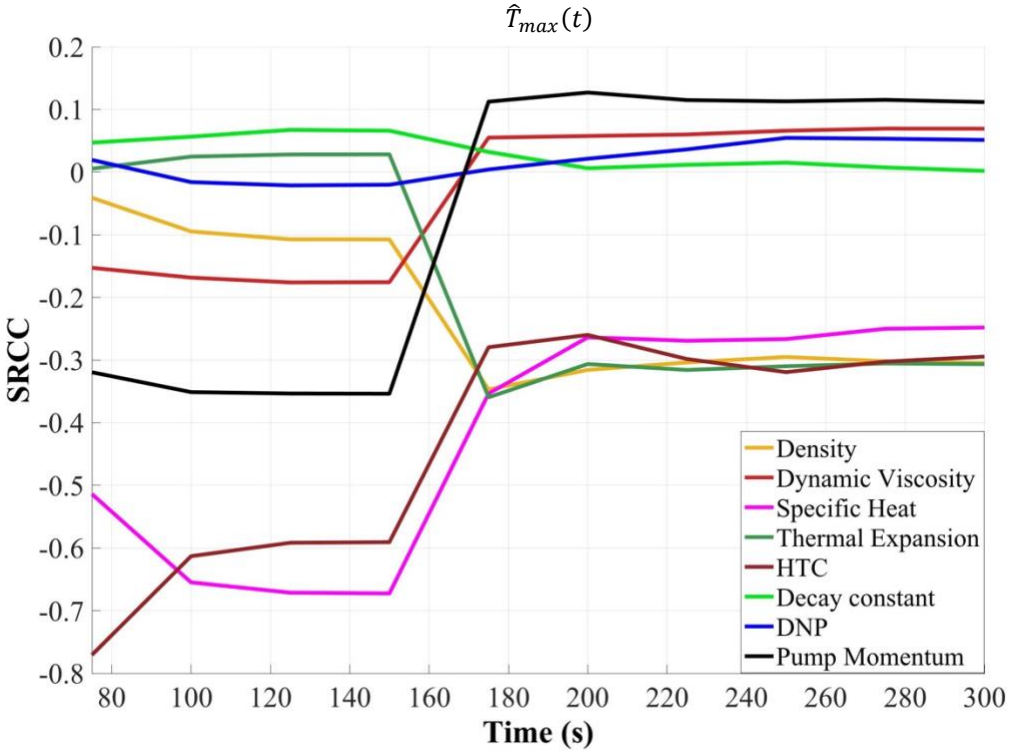


Figure 79. Time dependent SRCC for system maximum temperature

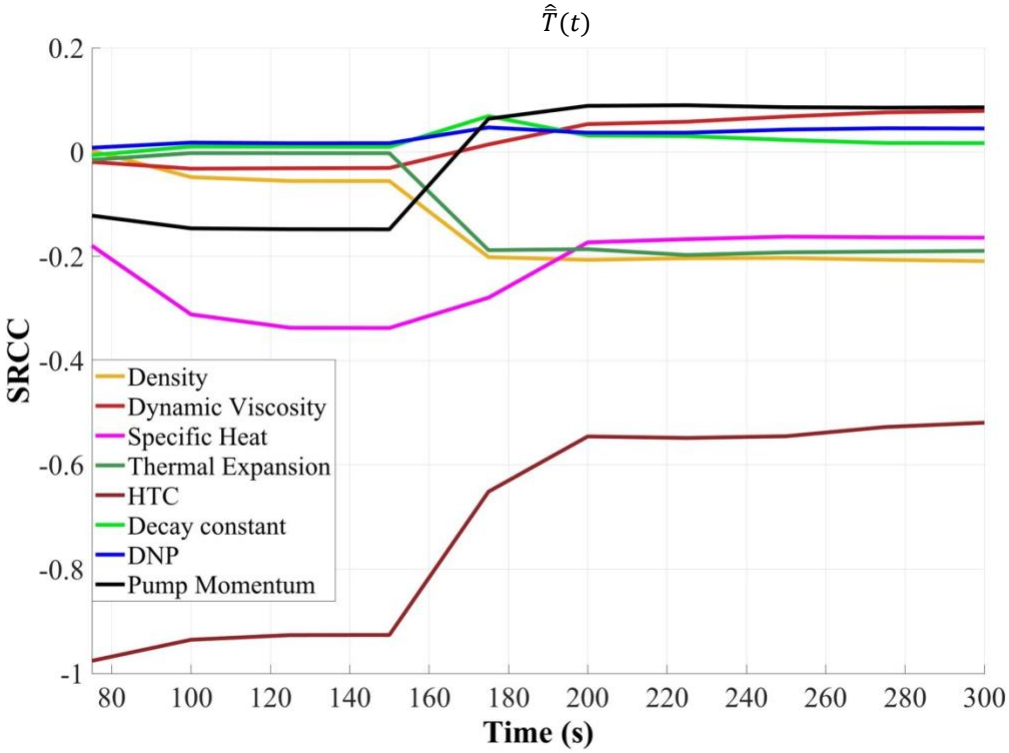


Figure 80. Time dependent SRCC for system average temperature

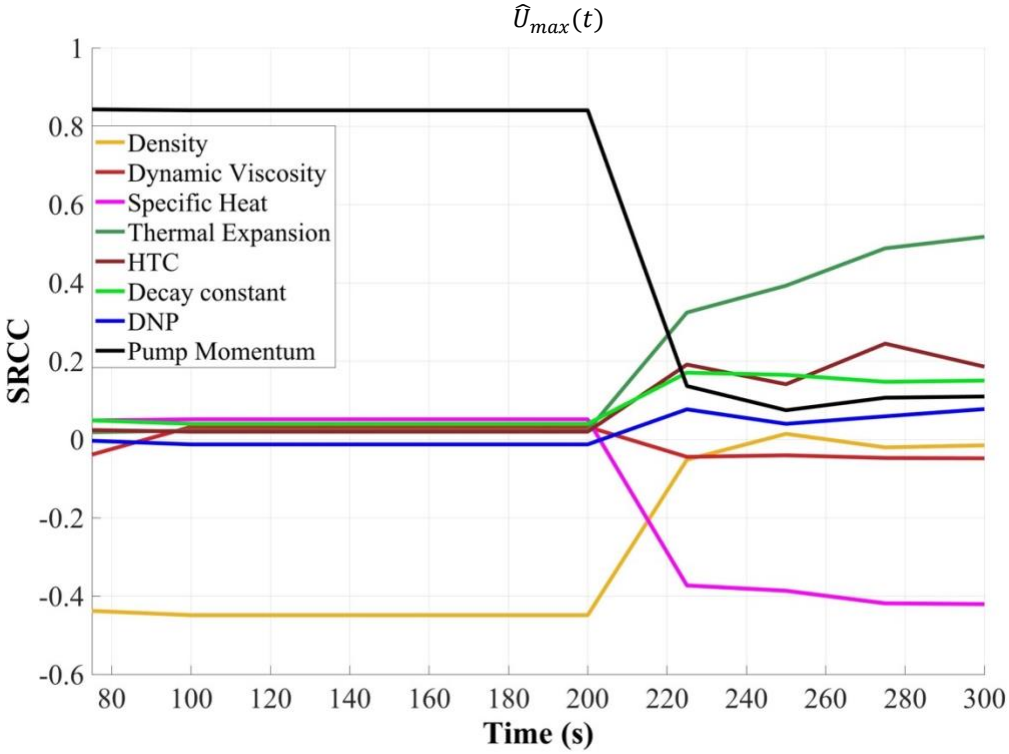


Figure 81. Time dependent SRCC for system maximum velocity

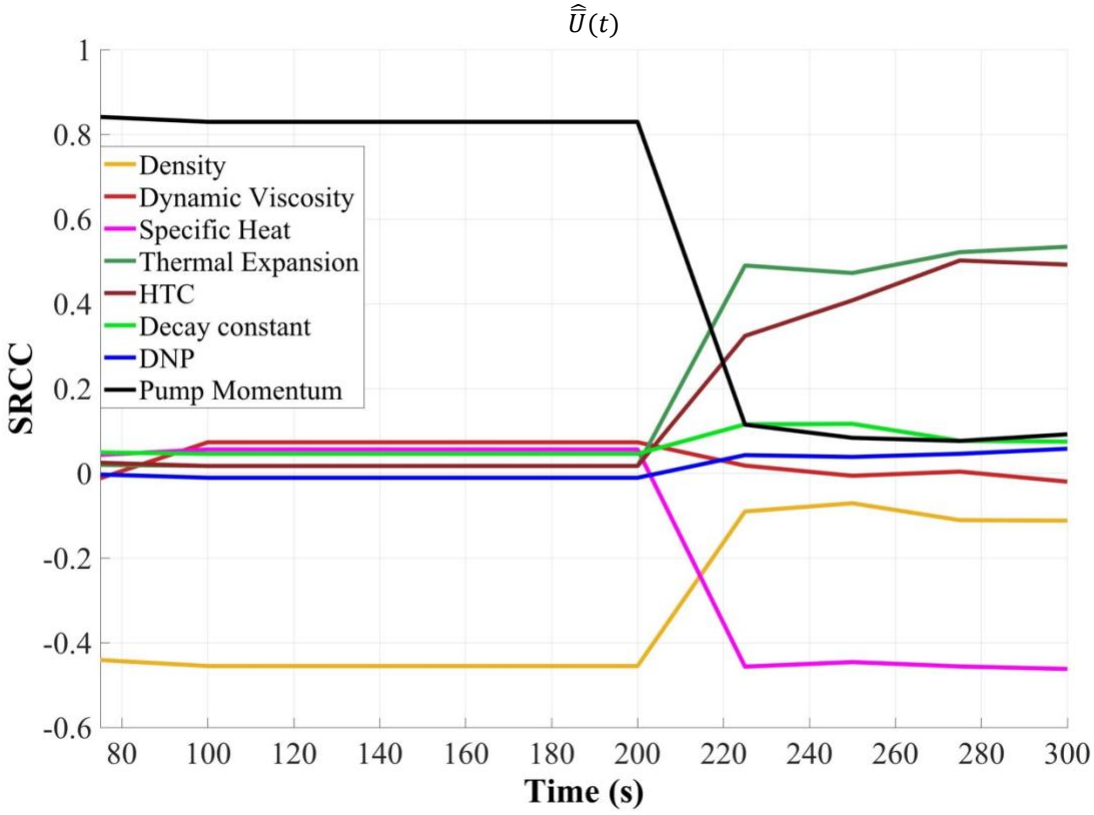


Figure 82. Time dependent SRCC for system average velocity

From the time dependent analysis, it can be seen that the thermal expansion coefficient correlation gains more importance as the transient begins. At 150s, when the pump is turned off completely, temperature of the system changes and β_{th} determines the magnitude by which the density change occurs as shown in Equation 56.

$$\rho = 1 - \beta_{th}\left(T - T_{ref}\right) \tag{56}$$

In the reactor system after the pump is turned off, the difference in density is what drives the flow of fuel salt and helps the system attain natural circulation thereby dictating the temperature distribution. When the flow is driven by pump momentum, the effect of β_{th} is suppressed as pump momentum is the dominant factor that influences velocity and temperature of the fuel salt. To understand the significance of specific heat capacity correlation with input parameters it is important to understand the following equations,

$$Q_s = mC_p dT \tag{57}$$

$$Q_{hex} = UAdT \tag{58}$$

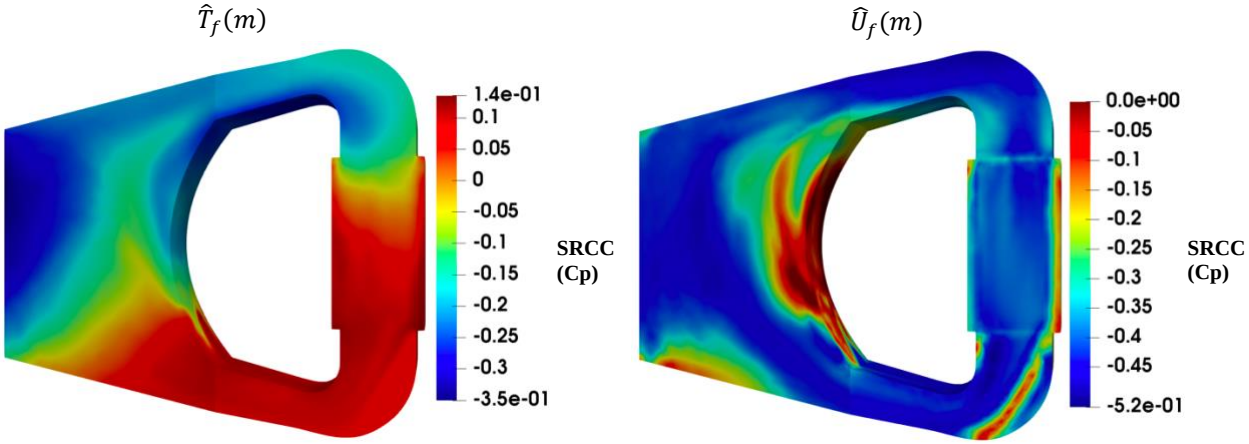
Equation 57 is the single side heat transfer equation. It accounts for heat capacity of fluid and enthalpy change that is occurring. Equation 58 is the heat exchanger design and performance equation. It is a function of thermal performance, Log Mean Temperature Difference (LMTD) and exchanger area. The heat transferred from primary fuel salt to the secondary fluid (coolant) is governed by Equation 58 and the total cooling achieved in the primary fuel salt is governed by Equation 57. The heat gained or lost in the fuel salt depends on the inherent characteristics of the salt such as specific heat, as a result, it would affect the outlet temperature of the primary fuel salt exiting the heat exchanger and in general the overall temperature of the salt. After the transient is initiated, specific heat ceases to be a dominating factor and its effect on temperature is suppressed as soon as density and thermal expansion take over as they start dictating the temperature of the fuel salt in the reactor system. Since specific heat still has an impact on temperature, it's just not very pronounced, this impacts the velocity as density/temperature difference drive the flow upon attaining natural circulation. Hence specific heat capacity correlation with velocity increases after initiating the transient as seen from Figure 81 and Figure 82.

With regards to heat transfer coefficient, the correlation with temperature decreases upon initiating the transient. This is because as soon as the transient is initiated, it becomes a density driven flow and specific heat capacity and thermal expansion coefficient start dictating the temperature and velocity of the system. Whereas the velocity correlation with heat transfer coefficient increases after initiating the transient. The two behaviors are interlinked, the amount of cooling (single side heat transfer) achieved in the primary fuel salt depends on the velocity of the system as pointed out in Equation 57 and the heat transferred from the primary fuel salt to the

secondary coolant depends on the heat transfer coefficient. The velocity of the fuel salt determines the residence time of the salt in the heat exchanger and as a result the heat transfer coefficient correlation with velocity of the system increases as seen in Figure 81 and Figure 82.

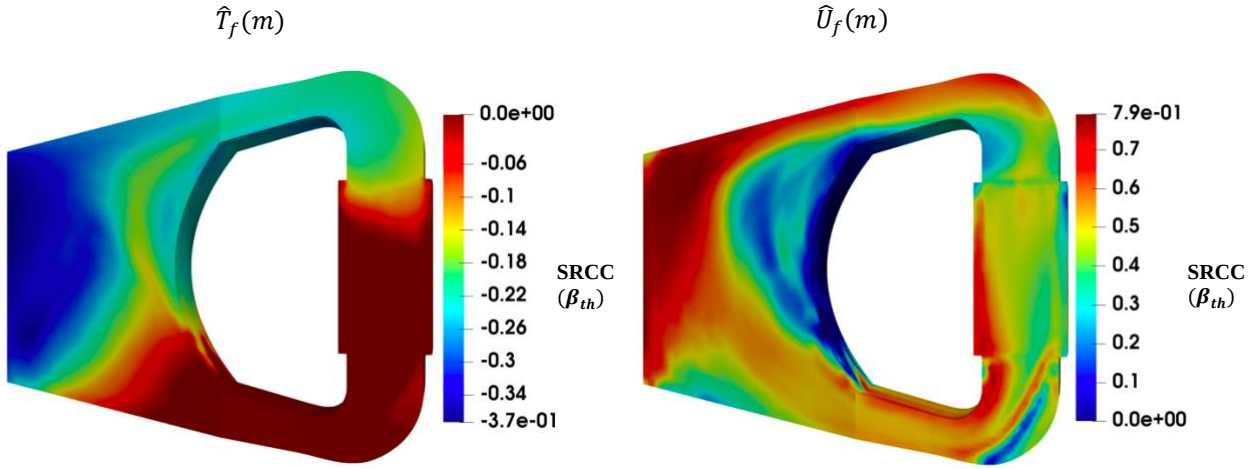
In the 3-D sensitivity study, the SRCCs are computed for the ensemble average of temperature and velocity at every nodal point in space at the end of the transient. From the scalar and time dependent SRCC analysis it is clear that the kinetic parameters do not have much of an impact on both the temperature and velocity. The advantage of a 3-D SRCC study is that it helps in understanding input parameter correlation with the output quantity of interest at every point in space. As pointed out from scalar and functional analysis, specific heat, thermal expansion, and heat transfer coefficient have the biggest impact on temperature and velocity of the system and only these parameters are taken into consideration in this study. From Figure 73 it is clear that the minimum and maximum temperatures inside the core occurs along the centerline and these temperatures are impacted by the specific heat of the salt as shown in Equation 57. As a result, there is a strong correlation between temperature and specific heat observed along the centerline of the core as shown in Figure 83 (a). Since this is a natural circulation driven flow, the temperature differences inside the core drive the fuel salt flow and specific heat contributes towards these temperature differences within the core, as a result a strong correlation between specific heat and velocity of the fuel salt can be observed. The inverse relation between specific heat and velocity can also be verified through Figure 83 (b). Figure 84 (a) shows that the thermal expansion coefficient has a similar effect on the temperature of the salt as specific heat and is strongly correlated near the centerline of the core. As soon as the pump is turned off, it becomes a density driven flow and β_{th} determines the magnitude by which the density change occurs thereby

impacting the fuel salt velocity as shown in Figure 84 (b) which in turn governs the temperature distribution inside the core.



(a) (b)

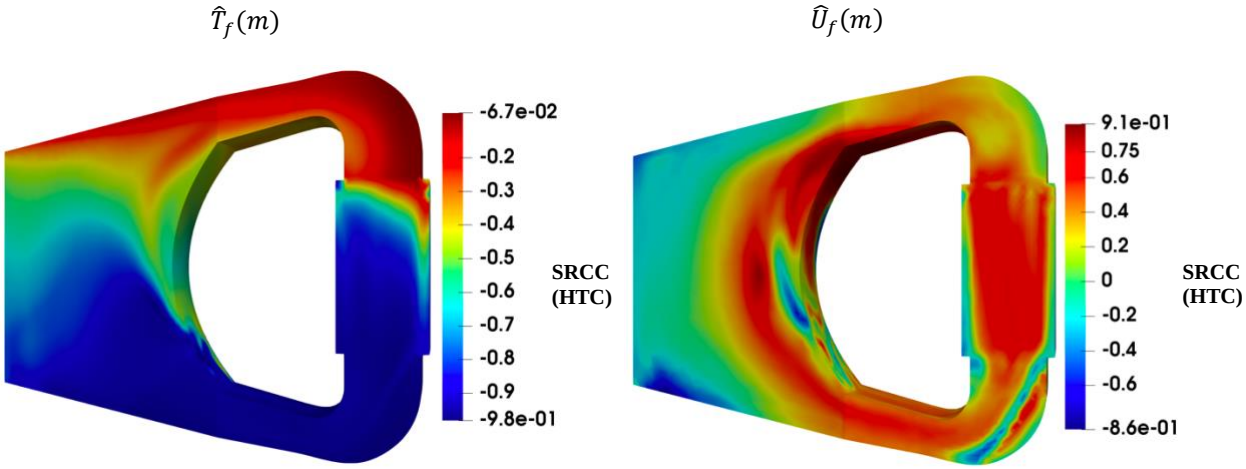
Figure 83. Three-dimensional SRCC of specific heat with (a) Temperature (b) Velocity



(a) (b)

Figure 84. Three-dimensional SRCC of thermal expansion coefficient with (a) Temperature (b) Velocity

From Equation 58 it is understood that the heat transfer coefficient plays a significant role in determining the temperature of the fuel salt at the exit of the heat exchanger and subsequently at the inlet of the core. This strong correlation between heat transfer coefficient and temperature is verified through Figure 85 (a). As soon as the transient is initiated, it becomes a density-driven flow and specific heat capacity and thermal expansion coefficient take over and dictate the temperature and velocity of the system. So, the amount of cooling (single side heat transfer) achieved in the primary fuel salt depends on the velocity of the system as pointed out in Equation 57 and the heat transferred from the primary fuel salt to the secondary coolant depends on the heat transfer coefficient as shown in Equation 58. The velocity of the fuel salt determines the residence time of the salt in the heat exchanger and as a result a strong correlation between heat transfer coefficient and velocity can be observed in Figure 85 (b).



(a) (b)

Figure 85. Three-dimensional SRCC of heat transfer coefficient with (a) Temperature (b) Velocity

5.5 3-D LFMSR Output Space Dimension Reduction

In this study so far, both temporal and spatial variation of output quantities of interest are considered important to understand the reactor system behavior during transients better. In light of this, the dimension of the output space is relatively huge thereby adding to the complexity of post-processing the data. Potentially, Principal Component Analysis (PCA) method can be used to reduce the dimension of the output space and with only a few principal components be able to represent the entire output dimension space. The dimension reduction can be very useful for more complex functional sensitivity analyses or for creating reduced order models for the GeN-Foam outputs.

To calculate the principal components a simple python tool is used, and the following results are obtained. For the temporal analysis, with just three principal components, 99% of variance in all the outputs at different time intervals can be explained.

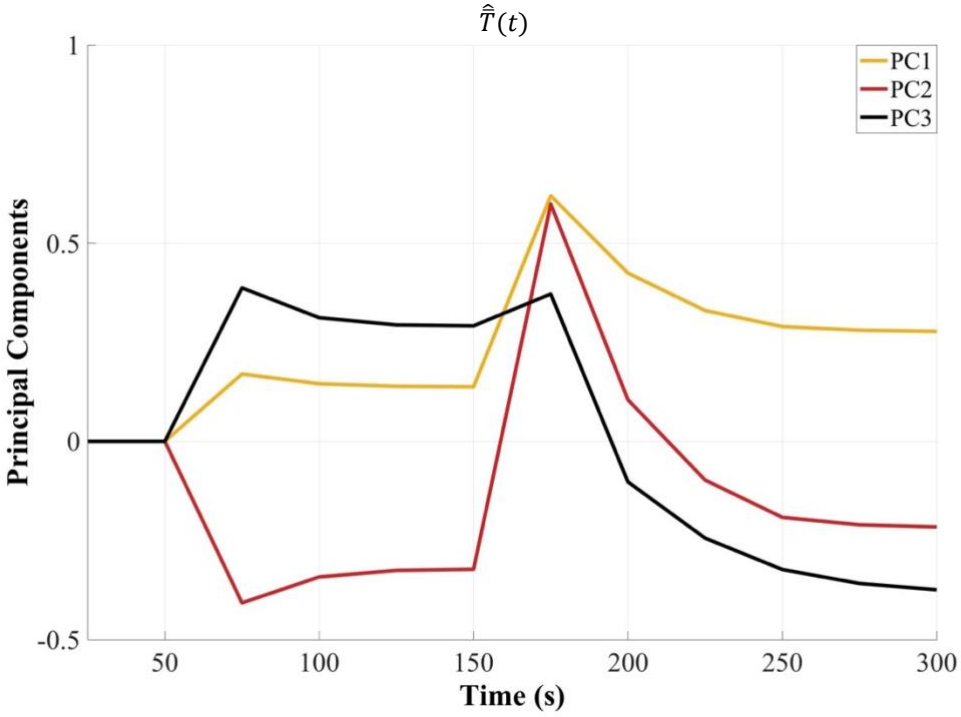


Figure 86. Principal components for average temperature of salt

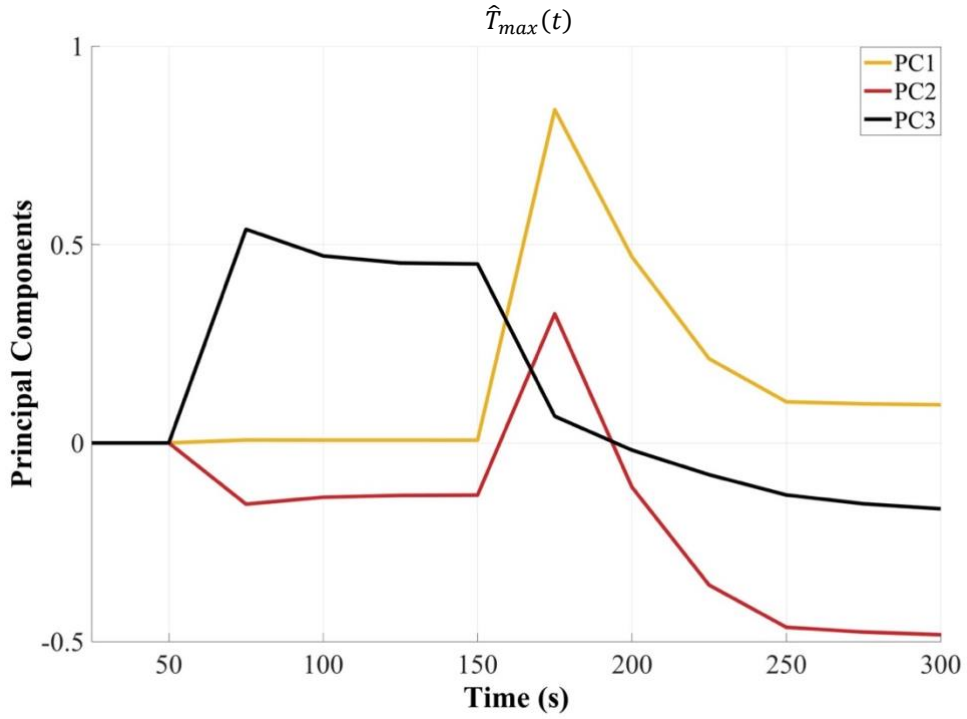


Figure 87. Principal components for maximum temperature of salt

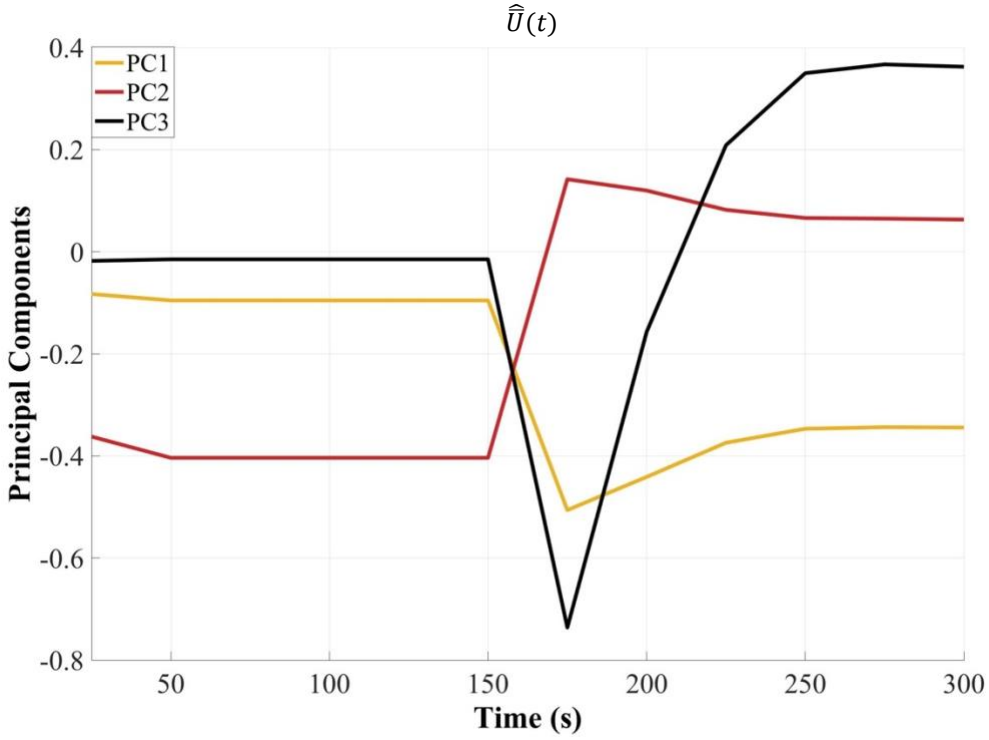


Figure 88. Principal components for average velocity of salt

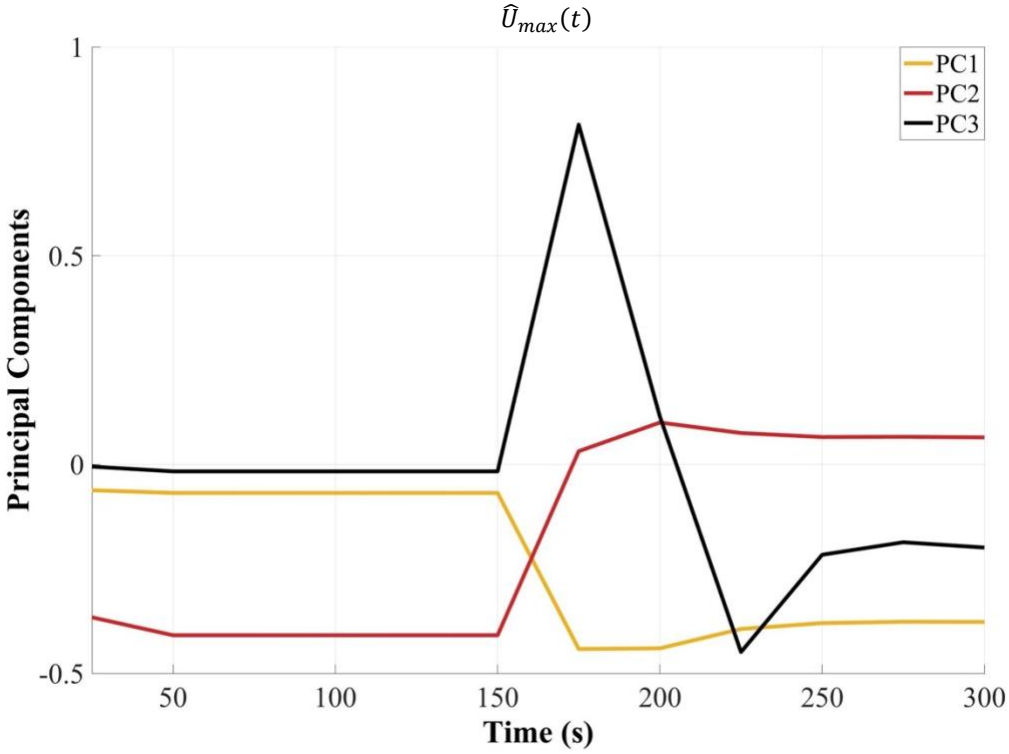


Figure 89. Principal components for maximum velocity of salt

Comparing Figure 69 and Figure 70 with Figure 86 and Figure 87, it is clear that principal component one follows a very similar trend with the mean and standard deviation for both average and maximum temperature thereby capturing most of the uncertainties arising after the initiation of transient. Meanwhile principal components two and three focus on capturing the smaller uncertainties observed before the transient. Similar trend is observed with the principal components for average and maximum velocity as shown in Figure 88 and Figure 89 and is not discussed here for brevity. Similarly, for the 3-D analysis the output dimensions are reduced from 511568 to just 4 using the PCA method.

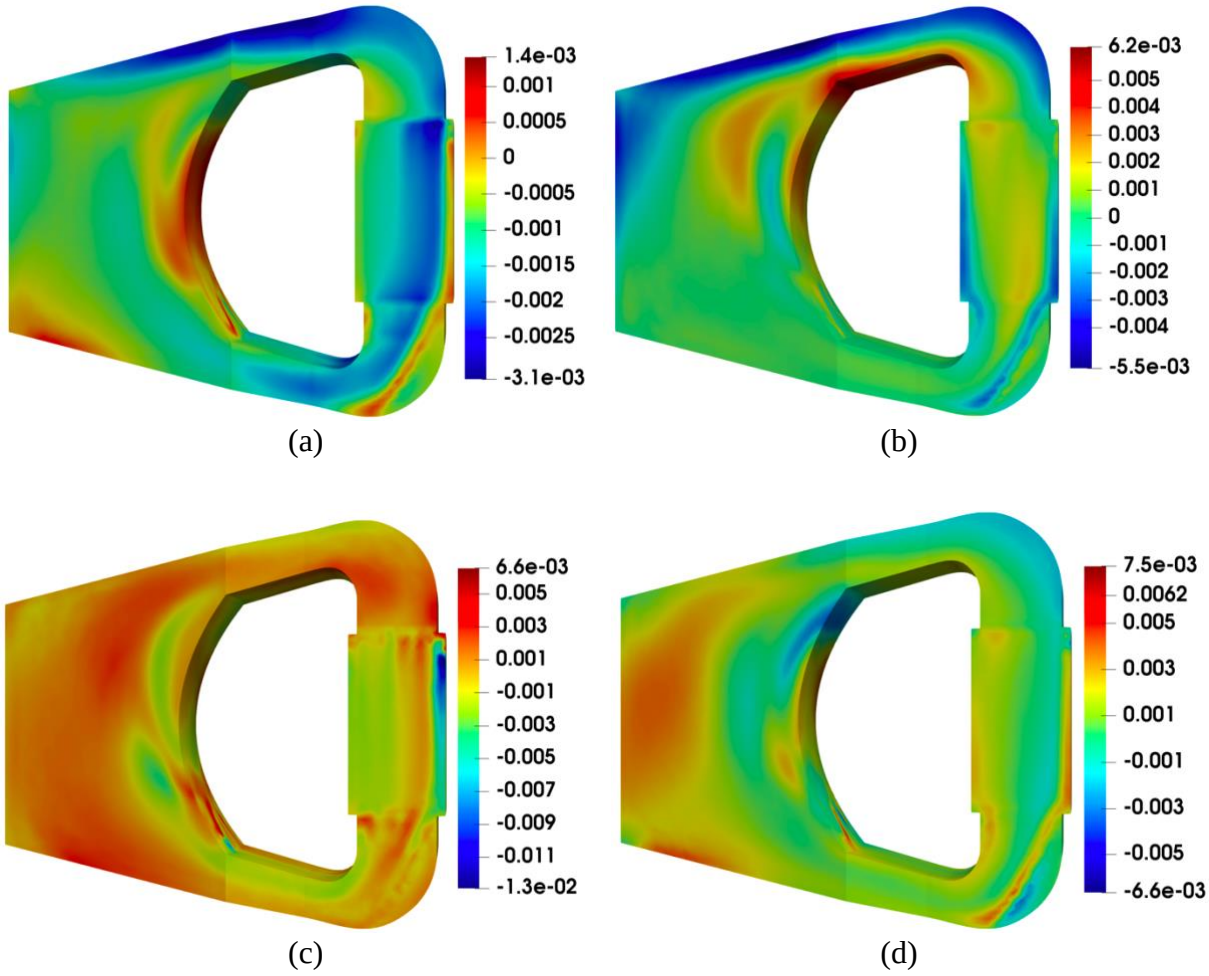


Figure 90. Principal components for velocity (a) PC1 (b) PC2 (c) PC3 (d) PC4

The first principal component captures most of the uncertainties around the upper region of the core where the mean and standard deviation values of velocity are higher. This can be seen in Figure 90 (a) where more weights are placed near the curved wall and the upper regions of the core. The second principal component focuses on capturing the local sub-region uncertainties associated with the upper part of the core as shown in Figure 90 (b). Whereas third and fourth principal component focus on scaling all the other regions that were left out. The first two principal components capture about 85% of the uncertainties and the four principal components together are able to represent about 95% of the variance in the output space for velocity.

With regards to temperature, the first principal component is able to capture the global uncertainties within the reactor system as shown in Figure 91 (a). While the second principal component focuses on the uncertainties associated with the heat exchanger and upper part of the core as seen in Figure 91 (b). Whereas third and fourth principal component focuses on different aspects of the reactor system.

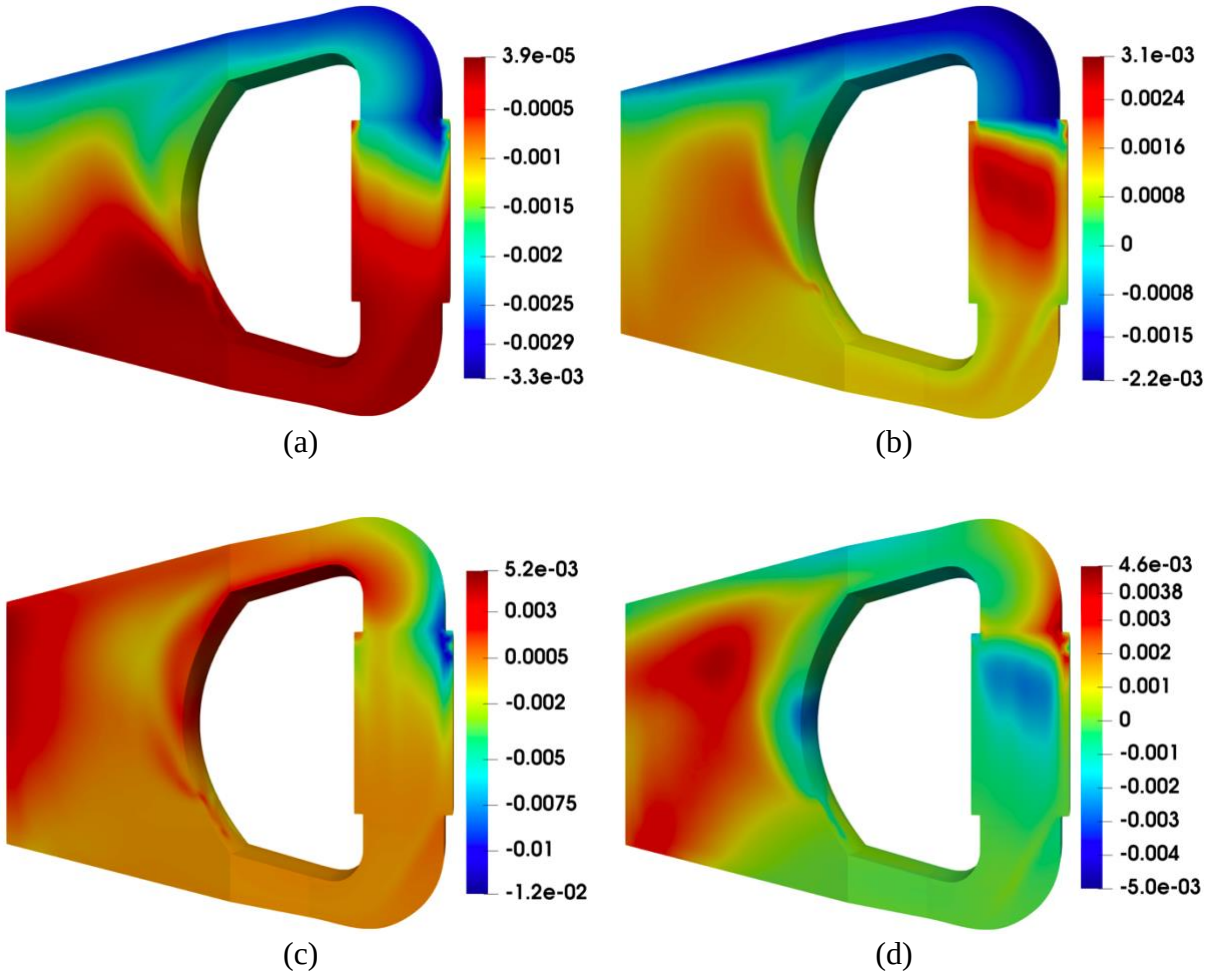


Figure 91. Principal components for temperature (a) PC1 (b) PC2 (c) PC3 (d) PC4

To summarize, the 3-D LFMSR system was modeled in GeN-Foam and uncertainties were propagated through the input design parameters. The output uncertainties were characterized using statistical analysis and spearman coefficients.

CHAPTER 6

Conclusions and Perspectives

6.1 Conclusions

The LFMSR concept has been around since the 1960s, but only recently has it gained traction among the nuclear community. From literature it is understood that there are several software developments addressing the LFMSR analysis and aiming to model the multi-physics aspect of LFMSRs accurately; however, at the moment lacking a definitive complete approach to the design and analysis of such reactors. The primary objective of this thesis was to develop a framework to perform multi-physics modeling, design optimization and uncertainty quantification under one umbrella.

To this purpose, codes such as GeN-Foam, SAM and Dakota are used as part of the framework. GeN-Foam to perform high-fidelity coupled fine/coarse mesh thermal-hydraulics calculations and multi-group neutrons diffusion calculation using unstructured meshes of the reactor; and SAM to perform time efficient system level coupled thermal-hydraulics and point-kinetics calculations of the LFMSR design. While Dakota is used to perform the uncertainty propagation and sensitivity analysis studies. Several code developments were done as part of this work: a mass transport model was implemented in GeN-Foam to study the transport of different species and analyze their effect on the reactor design and model, along with a solid-solid heat conduction model to take into account the corresponding thermal stresses that were implemented in the thermal-mechanics solver of GeN-Foam. The code capabilities have been demonstrated in Chapter 3 along with the respective verification studies to attain confidence with codes. For the

verification studies performed, CFD codes, Fluent and STARCCM+ were used to benchmark our predictions via code-to-code comparative analyses.

As part of the framework, GeN-Foam and SAM were coupled with Dakota for propagation of uncertainties and post-processing the results in a more efficient way. The successful coupling of GeN-Foam/Dakota framework was verified by propagating the uncertainties and performing sensitivity analysis. Preliminary results revealed that the impact of input uncertainties was felt more towards the outlet of the core design. This is because the temperature is higher towards the outlet of the core as the fuel salt near the outlet would have absorbed more energy with respect to other positions of the core. For future modeling purposes involving input uncertainty propagation and quantification of uncertainties in the outputs, the number of probes used for the study can be reduced significantly and be more concentrated towards the outlet of the core where the impact on temperature due to input uncertainties was more prominent.

Pearson coefficients calculated as part of the sensitivity analysis, revealed that the density and thermal expansion coefficient play the most important roles when it comes to the pressure difference across the modeled system. Dynamic viscosity had the least impact of all parameters in an incompressible case study and the heat capacity of the salt had a strong correlation with temperature as expected.

Considering the lack of rigorous studies done on the reactor geometry and primary loop components, it was important to perform detailed analysis by optimizing the design and propagating uncertainties to understand the reactor system behavior during normal and accident scenarios. Initially, a right-circular cylindrical core model was proposed based on the reference MSFR concept as shown in Chapter 4. Preliminary studies revealed that the design was simple to model but its flow paths cause recirculation zones which are not desirable as they allow the coolant

to retain heat energy which may degrade integrity of the reactor vessel. As a result, a new core geometry in the shape of an hourglass was proposed that can minimize the presence of recirculation zones. This particular design was further optimized based on the neutronics and the primary loop components (heat exchanger and pump) performance thereby finalizing the reactor core and primary loop components dimensions along with their respective positions. Preliminary 2-D multi-physics models were built in both GeN-Foam and SAM and an unprotected loss of flow transient was simulated in SAM. The results revealed that the optimized reactor design was safe as the temperatures stayed well below the boiling temperature of the fuel salt and a reasonable natural circulation velocity was attained during the transient.

Upon gaining confidence with the optimized reactor design, a 3-D model was built in GeN-Foam to do a more thorough analysis. The 3-D LFMSR primary loop model was built in SOLIDWORKS and then meshed using Trelis to solve for thermal-hydraulics and neutronics in a coupled fashion. The steady state simulation of the LFMSR model was performed in two steps, first obtaining a steady state with only fluid-dynamics equations solved and then launching a second steady state simulation from the results of the first steady state run solving for both neutronics and energy equations. The peak temperature occurred near the core outlet and the upper wall as expected and was lower at the core inlet after exiting the heat exchanger. The neutron flux concentration/density depended on the salt composition and the energy boundaries used and as a result energy group 3 had the maximum concentration/density. With regards to precursor concentration, group 1 was higher near the core upper region because of its vicinity to the generation area, and the direction of flow whereas Group 8 had a relatively smaller half-life compared group 1 ($T1/2 = 0.2$ s) decayed in the place of generation. To study the reactor transient behavior, an ULOF accident scenario was simulated by setting the pump momentum source to

zero instantaneously after the reactor system reached steady state and turning on both thermal-hydraulic and neutronics solvers. A reasonable natural circulation velocity was attained and the maximum temperature in the reactor system was below the boiling temperature of the fuel salt.

To further analyse the design, an uncertainty quantification and propagation study was done where uncertainties of the input design parameters were propagated through a ULOF transient simulation using the two codes, SAM and GeN-Foam. The motivation behind using two codes was that SAM being a system code, the uncertainty propagation studies would be many orders of magnitude faster compared to the GeN-Foam runs. There were no discrepancies observed during the SAM runs and hence a more detailed higher fidelity GeN-Foam uncertainty propagation study was done. Three analyses were performed namely, scalar, functional and three-dimensional analysis to understand the impact of input uncertainty propagation on temperature and velocity predictions. Each of these studies provided unique information about the reactor design. Through the scalar analysis it was observed that 99% of the data for the maximum and average temperature was below 1000 K and 875 K and the respective standard deviations were 10.73 K and 5.025 K thereby not imposing any design safety risks. Since this was a transient scenario that was being evaluated it was important to ensure that reasonable temperature and velocity was maintained during the entire duration of the transient. As a result, maximum and average temperature and velocity was studied over time. Functional analysis results revealed that the standard deviations of both temperature and velocity increased right after the transient was initiated i.e.., exactly when the reactor system was most unstable and as the system attained natural circulation regime, the standard deviations decreased. The rate of convergence of output parameters can also play a role in the large standard deviations observed as they could vary depending on the input perturbations. In the three-dimensional study, each single nodal point in the spatial domain of the reactor system

was taken into account and the results revealed that the relative standard deviation values were higher near the inlet of the heat exchanger for temperature predictions. This is the result of accumulation of uncertainties contributed by the input design parameters such as thermal expansion coefficient and specific heat that take effect as the fuel salt moves through the core. With regards to velocity, the maximum uncertainty is observed near the curved wall of the reactor where the velocities are very close to zero whereas in other regions the standard deviation are relatively lower by many orders of magnitude. Sensitivity analysis was performed using SRCCs to measure the correlation between the input and output data. The sensitivity analysis study was done at three different levels (scalar, functional and three-dimensional level) to understand how the correlations changed with time and space. Overall, it was found that the dynamic viscosity and kinetic parameters did not have much of an impact on temperature and velocity whereas the thermal expansion coefficient, heat transfer coefficient and specific heat had a stronger impact. As soon as the transient is initiated, the reactor system switches from a pump momentum driven flow to a density/temperature difference driven flow thereby attaining natural circulation. The specific heat determines the temperature at which the primary fuel salt exits the heat exchanger, the thermal expansion coefficient determines the magnitude of the density change with respect to temperature, and finally the heat transfer coefficient impacts the amount of heat transferred from the primary to the secondary fluid. All the three parameters impact the temperature in one way or other thereby impacting the velocity of the fuel salt in the system. In the future, more focus needs to be given to specific heat, thermal expansion coefficient and heat transfer coefficient calculations while performing experiments to reduce the associated uncertainties as they have the biggest impact of all input design parameters on the reactor system.

Finally, PCA method was used to reduce the dimension of the output space thereby making it easy to post-process data efficiently. For the temporal analysis, with just three principal components, 99% of the variance in all the outputs at different time intervals were explained. While for the three-dimensional analysis, with four components the output dimensions were reduced from 511568 to just 4. In the future, these principal components can be used to build reduced order models to efficiently predict the temperature and velocity of the system under different conditions.

6.2 Perspectives

Although this PhD research successfully developed a framework for multi-physics modeling, design optimization and uncertainty quantification of LFMSRs, considering the several assumptions made to simplify the multi-physics modeling and uncertainty propagation, the following future improvements are recommended:

- Currently, the macroscopic cross-sections are generated assuming fuel is static but in reality, that is not the case as the fuel circulates around the entire reactor system. This needs to be taken into account while generating the macroscopic cross-sections for the neutronics diffusion solver.
- A constant heat transfer coefficient was used for GeN-Foam models, but it is more practical to use a coefficient that correlates with the Nusselt number. Although GeN-Foam has the capability to calculate the heat transfer coefficient based on different Nusselt number correlations (e.g., Dittus-Boelter), it was not used in this research since a dedicated correlation for fuel molten salts aren't still widely available.
- In this study so far, uncertainties in the macroscopic cross-sections are not considered and for a more comprehensive study, SCALE code can be used to generate the cross-sections

along with their covariances. This will allow for propagating nuclear data uncertainties through the model.

- Extending the study to model the entire 3-D sixteen loop LFMSR model. The important transients to simulate for the 3-D model would be:
    - Single-loop transient – to study the change in salt composition, pump failure or increase, natural circulation, increasing heat transfer coefficient, tube blockage and salt freezing.
    - Four-loop/Eight-loop transient - to analyze the asymmetric behavior of pump failure, increasing heat transfer coefficient, tube blockage, salt freezing and other symmetric phenomena.
- Modeling the entire 3-D primary loop and performing uncertainty studies can be expensive and impractical. The computational cost can be reduced by developing reduced order models and they can also be used as part of the design optimization studies.

REFERENCES

1. Alsayyari, F., Tiberga, M., Perko, Z., Lathouwers, D., & Kloosterman, J. L. (2020). A nonintrusive adaptive reduced order modeling approach for a molten salt reactor system. *Annals of Nuclear Energy 141*.
2. Altahhan, M. R., Avramova, M., & Ivanov, K. (2018). Reactor Dynamics Model for The Molten Salt Reactor Based on P1 Approximation. *PHYSOR 2018, Cancun, Mexico, April 22-26.*
3. Altahhan, M. R., Bhaskar, S., Ziyad, D., Balestra, P., Fiorina, C., Hou, J., . . . Avramova, M. (2020). Preliminary design and analysis of Liquid Fuel Molten Salt Reactor using multi-physics code GeN-Foam. *Nuclear Engineering and Design 369*.
4. Altahhan, M., Bhaskar, S., Balestra, P., & Hou, J. (December 13, 2018). *NCSU Progress Report for the MSR design project.* Raleigh, NC: Reactor Dynamics and Fuel Modeling Group.
5. Altahhan, M., Bhaskar, S., Balestra, P., Hou, J., Avramova, M., & Smith, N. (2018). Advanced Liquid Fuel Molten Salt Reactor Core Simulation Using GeN-Foam. *American Nuclear Society Winter meeting.* Florida.
6. Altahhan, M., Bhaskar, S., Ziad, D., Balestra, P., & Hou, J. (April 8, 2018). *NCSU technical report for the MSR design project.* Raleigh, NC: Reactor Dynamics and Fuel Modeling Group.
7. Ariu, V. (2014). Heat Exchanger Analysis for Innovative Molten Salt Fast Reactor. *ETH Zurich, Master Thesis*.
8. Aufiero, M. (2014). *Development of Advanced Simulation Tools For Circulating-Fuel Nuclear Reactors. PhD. Thesis.* Milan, Italy: Politecnico Di Milano.

9. Aufiero, M., & Rubiolo, P. (2018). Testing and Verification of Multiphysics Tools for Fast-Spectrum MSRs: The CNRS Benchmark. *Transactions of the American Nuclear Society*.

10. Aufiero, M., Brovchenko, M., Cammi, A., Clifford, I., Geoffroy, O., Heuer, D., . . . Rouch, H. (2014). Calculating the effective delayed neutron fraction in the Molten Salt Fast Reactor: Analytical, deterministic and Monte Carlo approaches. *Annals of Nuclear Energy 65*, 78-90.

11. Aufiero, M., Cammi, A., Fiorina, C., Luzzi, L., & Sartori, A. (2013). A multi-physics time-dependent model for the Lead Fast Reactor single-channel analysis. *Nuclear Engineering and Design 256*, 14-27.

12. Aufiero, M., Cammi, A., Geoffroy, O., Losa, M., Luzzi, L., Ricotti, M. E., & Rouch, H. (2014). Development of an OpenFOAM model for the Molten Salt Fast Reactor transient analysis. *Chemical Engineering Science*(111), 390–401.

13. Betzler, B. R., Heidet, F., Feng, B., Rabit, C., Sofu, T., & Brown, N. R. (2019). Modeling and simulation functional needs for molten salt reactor licensing. *Nuclear Engineering and Design*, 355.

14. Bhaskar, S., Altahhan, M., Ziyad, D., Balestra, P., Hou, J., Avramova, M., & Smith, N. (2018). Modelling and Simulation of a Liquid Fuel Molten Salt Reactor Core Using GeN-Foam. *PHYTRA4 - The Fourth International Conference on Physics and Technology of Reactors and Applications.* Marrakech, Morocco.

15. Brovchenko M., Merle-Lucotte E., Heuer D., Rineiski A. (June 2013). Molten Salt Fast Reactor transient analyses with the COUPLE code. *ANS Annual Meeting.* Atlanta, USA.

16. Brovchenko, M., Merle-Lucotte,E.,Rouch,H.,Alcaro,F.,Aufiero,M.,Cammi,A.,Dulla,S.,. (2013). *Optimization of the Pre-Conceptual Design of the MSFR—EVOL Project.*

17. Carbajo, J. D. (2017). Modeling the Molten Salt Reactor Experiment with the RELAP5-3D Code. ANS Annual Meeting. San Francisco.

18. Cervi, E. L. (2019). Development of a multiphysics model for the study of fuel compressibility effects in the molten salt fast reactor. *Chemical Engineering Science*, 379-393.

19. Cervi, E., Lorenzi, S., Cammi, A., & Luzzi, L. (2019). Development of a multiphysics model for the study of fuel compressibility effects in Molten Salt Fast Reactor. *Chemical Engineering Science 193*, 379-393.

20. Cetin, B. (2016). Computational Modeling of Vehicle Radiators Using Porous Medium Approach. *INTECH*.

21. Chengbin Shi, M. C. (2016). Development and application of a system analysis code for liquid fueled molten salt reactors based on RELAP5 code. Nuclear Engineering Design, 378-388.

22. Collins, B., Graham, A., Dawn, A., Salko, R., Smith, N., & Gentry, C. (2018). Coupled Simulations of Fast-Spectrum Molten Salt Reactors. *PHYSOR 2018.* Cancun, Mexico.

23. Darcy, H. (1856). *Les Fontaines Publiques de la Ville de Dijon: Exposition et Application des Principes a Suivre et des Formulesa Employer dans les Questions de Distribution d'Eau.* Paris.

24. David E. Holcomb, G. F. (2013). *Fluoride Salt-Cooled High-Temperature Reactor Technology Development and Demonstration Roadmap.* Tennessee: ORNL.

25. Davis, C. B. (2005). *Implementation of Molten Salt Properties into RELAP5-3D/ATHENA.* Idaho Falls: Idaho National Engineering and Environmental Laboratory.

26. D'Auria, F. (2012). *Perspectives in System Thermal-Hydraulics*. Nuclear Engineering and Technology, 855-870.

27. Delipei, G. (2019). *Development of an uncertainty quantification methodology for Multi-Physics Best Estimate analysos and application to the Rod Ejection Accident in a Pressurised Water Reactor.* Paris: Universite Paris-Saclay.

28. Desyatnik, V. N., Katyshev, S. F., Raspopin, S. P., & Chervinskii, Y. F. (1975). Density, Surface Tension, and Viscosity of Uranium Trichloride-Sodium Chloride Melts. *Atommaya Energiya,* Vol 39, 70-72.

29. Dolan, T. J. (2017). *Molten Salt Reactors and Thorium Energy.* Cambridge, USA: Woodhead Publishing.

30. Engel, J. R. (1980). *Conceptual Design Characteristics of a Denatured Molten-Salt Reactor with Once-Through Fueling.* Tennessee: Technical Report ORNL-TM-7207. Oak Ridge National Lab.

31. Fiorina, C. (2013). The Molten Salt Reactor as a Fast-Spectrum Candidate for Thorium Implementation. *Politecnico di Milano, Doctoral Thesis*.

32. Fiorina, C., & Mikityuk, K. (2015). Application of the new GeN-Foam multi-physics solver to the European Sodium Fast Reactor and verification against available codes. *ICAPP 2015.* Nice, France.

33. Fiorina, C., Clifford, I., Aufiero, M., & Mikityuk, K. (2015). GeN-Foam: a novel OpenFOAM based multi-physics solver for2D/3D transient analysis of nuclear reactors. *Nuclear Engineering and Design*(294), 24–37.

34. Fiorina, C., Hursin, M., & Pautz, A. (2017). Extension of the GeN-Foam neutronic solver to SP3 analysis and application to the CROCUS experimental reactor. *Annals of Nuclear Energy*(101), 419-428.

35. Fiorina, C., Hursin, M., & Pautz, A. (2017). Extension of the GeN-Foam neutronic solver to SP3 analysis and application to the CROCUS experimental reactor. *Annals of Nuclear Energy 101*, 419-428.

36. Fiorina, C., Kerkar, N., Mikityuk, K., Rubiolo, P., & Pautz, A. (2016). Development and verification of the neutron diffusion solver for the Gen-Foam multi-physics platform. *Annals of Nuclear Energy 96*, 212-222.

37. Fiorina, C., Kerkar, N., Mikityuk, K., Rubiolo, P., & Pautz, A. (2016). Development and verification of the neutron diffusion solver for the GeN-Foam multi-physics platform. *Annals of Nuclear Energy*(96), 212-222.

38. Fiorina, C., Lathouwers, D., Aufiero, M., Cammi, A., Guerrieri, C., Kloosterman, J. L., Ricotti, M. E. (2014). Modelling and analysis of the MSFR transient behavior. *Annals of Nuclear Energy*(64), 485–498.

39. Gentry, C. A. (2016). *Development of a Reactor Physics Analysis Procedure for the Plank-Based and Liquid Salt-Cooled Advanced High Temperature Reactor.* Knoxville, Tennessee: PhD Thesis.

40. German, P., Ragusa, J. C., & Fiorina, C. (2019). Application of multiphysics model order reduction to doppler/neutronic feedback. *EPJ Nuclear Sciences and Technologies 5*, 17.

41. Grahn, A. G. (2017). Simulation of an MSLB scenario using the 3D neutron kinetic core model DYN3D coupled with the CFD software Trio_U. *Nuclear Engineering Design*, 117-127.

42. Gregory, A. S. (2019). On the numerical modelling of frozen walls in a molten salt fast reactor. *Nuclear Engineering and Design*.

43. Haubenreich, P. N., & Engel, J. R. (1970). Experience with the Molten-Salt Reactor Experiment. *Nuclear Applications and Technology*, 118-136.

44. He, X. (2016). Validation of the TRACE Code for the System Dynamic Simulations of the Molten Salt Reactor Experiment and the Preliminary Study on the Dual Fluid Molten Salt Reactor. Munich: Technischen Universität München.

45. Hongbin Zhang, D. A. (2018). RELAP-7 User's Guide. Idaho Falls: Idaho National Laboratory.

46. Hu, R. (2017). *SAM Theory Manual.* Nuclear Engineering Division, Argonne National Laboratory.

47. Hu, R., Fanning, T. H., Sumner, T., & Yu, Y. (2015). *Status Report on NEAMS System Analysis Module Development.* ANL/NE-15/4, Argonne National Laboratory.

48. Hu, T., Cao, L., Wu, H., Du, X., & He, M. (2017). Coupled neutronics and thermal-hydraulics simulation of molten salt reactors based on OpenMC/TANSY. *Annals of Nuclear Energy*, 260-276.

49. Laureau, A. A.-L. (2015). Coupled neutronics and thermal-hydraulics transient calculations based on a fission matrix approach: application to the molten salt fast reactor. *Joint International Conference on Mathematics and Computation (M&C), Supercomputing in Nuclear Applications (SNA) and the Monte Carlo (MC) Method*, (pp. 1-12).

50. Leppanen, J., Pusa, M., Viitanen, T., Valtavirta, V., & Kaltiaisenaho, T. (2015). The Serpent Monte Carlo code: Status, development and applications in 2013. *Annals of Nuclear Energy 82*, 142-150.

51. Leppänen, J., Pusa, M., Viitanen, T., Valtavirta, V., & Kaltiaisenaho, T. (2015). The Serpent Monte Carlo code: Status, development and applications in 2013. *Annals of Nuclear Energy, 82*, 142-150.

52. Lucia, D. J., Beran, P. S., & Silva, W. A. (2004). Reduced-order modeling: new approaches for computational physics . *Progress in Aerospace Sciences 40*, 51-117.

53. Luzzi, L., Di Marcello, V., Cammi, A., & Kloosterman. (2012). *Multi-Physics Approach to The Modeling and Analysis of Molten Salt Reactors.* New York, USA: Nova Science Publishers.

54. Mahadevan, V. M. (2014). High-resolution coupled physics solvers for analysing fine-scale nuclear reactor design problems. *Philos. Trans. R. Soc. A: Math. Phys. Eng. Sci.*

55. Mathieu, L., Heuer, D., Merle-Lucotte, E., Brissot, R., Le-Brun, C., Liatard, E., . . . Lecarpentier, D. (2009). Possible configurations for the thorium molten salt reactor and advantages of the fast nonmoderated version. *Nuclear Science and Engineering*, 161(1):78–89.

56. Ott, K. O., & Neuhold, R. J. (1985). *Introductory Nuclear Reactor Dynamics, Ch.11.* Illinois, USA: American Nuclear Society.

57. Patankar, S. V. (1980). *Numerical Heat Transfer and Fluid Flow.* USA: Hemishpere Publishing Corporation.

58. Pauzi, A. M.(2015). Parametric Study of Natural Circulation Flow in Molten Salt Fuel in Molten Salt Reactor. AIP Conference Proceedings 1659.

59. Project, E. (2018). *Evaluation and Viability of Liquid Fuel Fast Reactor System (EVOL).* France: CORDIS.

60. Radaideh, M. (2019). *A novel framework for data-driven modeling, uncertainty quantification, and deep learning of nuclear reactor simulations.* Urbana: University of Illinois at Urbana Champaign.

61. Robertson, R. C. (1965). *MSRE Design and Operations Report. ORNL-TM-728.* ORNL.

62. Ronco, A. D., Introini, C., Cervi, E., Lorenzi, S., Jeong, Y. S., Seo, S. B., Cammi, A. (2020). Dynamic mode decomposition for the stability analysis of the Molten Salt Fast Reactor core. *Nuclear Engineering and Design 362*.

63. Rosenthal, W., Haubenreich, P. N., & Briggs, R. B. (1972). *The Development Status of Molten Salt Breeder Reactor Report. ORNL-4812.* ORNL.

64. Rouch, e. a. (Feburary 2014). Preliminary thermal-hydraulic core design of the Molten Salt Fast Reactor. *Annals of Nuclear Energy*.

65. R.R. Romatoski, L.W. Hu, "Fluoride-Salt-Cooled High-Temperature Test Reactor Thermal Hydraulic Licensing and Uncertainty Propagation Analysis," *Nuclear Technology*, 205:11, 1495-1512.

66. Rubiolo, P. R., Retamales, M. T., Ghetta, V., & Giraud, J. (2017). High Temperature Thermal Hydraulics Modeling of a Molten Salt: Application To a Molten Salt Fast Reactor (MSFR). *ESAIM: PROCEEDINGS AND SURVEYS*, (pp. Vol. 58, p. 98-117).

67. Sabharwall, P., & Kim, E. S. (2011). High temperature gas cooled reactors. *Nova Science Publications Inc.*

68. Sabharwall, P., Bragg-Sitton, S. M., & Stoots, C. (2013). Challenges in the development of high temperature reactors. *Energy conversion and management*, 574-581.

69. Taube, M., & Heer, W. (1980). *Reactor with very low fission product inventory EIR-Bericht Nr.411.* EIR. (January 2014). *Technology Roadmap Update for Generation IV Nuclear Energy Systems.* OECD Nuclear Energy Agency for the Generation IV International Forum.

70. Tianliang Hu, L. C. (2017). Coupled neutronics and thermal-hydraulics simulation of molten salt reactors based on OpenMC/TANSY. *Annals of Nuclear Energy*, 260-276.

71. Vafai, K. (2005). *Handbook of porous media.* New York: CRC Press, Taylor & Francis.

72. Weller, H. G., Tabor, G., Jasak, H., & Fureby, C. (1998). A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in Physics, 12*(6), 620-631.

73. Whitaker, S. (1986). Flow in porous media I: A theoretical derivation of Darcy's law. *Transport in Porous Media, 1*, 3-25.

74. Yan, J. K. (2011). Coupled computational fluid dynamics and MOC neutronic simulations of westinghouse PWR fuel assemblies with grid spacers. *14th International Topical Meeting on Nuclear Reactor Thermal Hydraulics, NURETH-14*, (pp. 1-15).

75. Ariu, V. (2014). *Heat Exchanger Analysis for Innovative Molten Salt Fast Reactor*. Laboratory for Reactor Physics and System Behavior: Paul Scherrer Institute.

76. Ronco, A. D. (2020). Preliminary analysis and design of heat exchangers for Molten Salt Fast Reactor. *Nuclear Engineering and Technology*, 51-58.

77. M.A. Ebadian, Z.F. Dong, Forced convection, internal flow in ducts, in: Y.I.C.W.M. Rohsenow, J.P. Hartnett (Eds.), Handb. Heat Transf., 1998, pp. 5.1e5.137.

78. Yoon, S. H. (2014). Assesment of straignt, zig-zag, S-shape, and airfoil PCHEs for intermediate heat exchangers of HTGRs and SFRs. *Nuclear Engineering and Design*, 334-343.

79. Jolliffe, I. T., Cadima, J., (2016). Principal component analysis: A review and recent developments. *Phil. Trans. R. Soc. A.* 374: 20150202.

80. Launder, B.E., Sharma, B., 1974. Application of the energy-dissipation model of turbulence to the calculation of flow near a spinning disc. Letters in Heat and Mass Transfer 1, 131–137.

APPENDICES

APPENDIX A

A probe is a postprocessing utility available in GeN-Foam that can be used to output the necessary information. Summary of the probes used along with their coordinate locations are given here.

Data Points	X coordinate	Y coordinate	Data Points	X coordinate	Y coordinate
1	0.31488	0.28046	25	0	1.75000
2	0.24702	0.50060	26	0	2.00000
3	0.23232	0.80537	27	0	2.25000
4	0.24126	1.09980	28	0	2.50000
5	0.26059	1.40270	29	1.15800	0.09900
6	0.29635	1.68920	30	1.01700	0.27200
7	0.37467	1.99110	31	0.85800	0.51300
8	0.53853	2.26660	32	0.73200	0.78700
9	0.80249	2.48180	33	0.67000	1.02000
10	1.00660	2.60300	34	0.65000	1.25000
11	0.40229	0.17143	35	0.69100	1.57700
12	0.59897	0.03612	36	0.77200	1.81400
13	0.90340	-0.10770	37	0.89500	2.04900
14	1.00780	-0.14950	38	1.01700	2.22700
15	1.15870	-0.19656	39	1.15800	2.40000
16	1.24920	-0.20635	40	1.25000	2.50000
17	1.14970	2.67050	41	0.18100	2.55300
18	1.23460	2.70370	42	0.39500	2.61500
19	0.75772	-0.04435	43	0.62500	2.68300
20	0	0.25000	44	0.85800	2.75100
21	0	0.50000	45	1.07400	2.81400
22	0	1.00000	46	1.25000	2.86600
23	0	1.25000	47	0.18100	-0.05300
24	0	1.50000	48	0.39500	-0.11500

APPENDIX B

Summary of mean and standard deviation values at different points

Mean and standard deviation values for centreline data points

Samples (100)	Mean	Standard Deviation
ΔP	1.415692e+04	3.981e+02
(1.2492 -0.20635 0)	6.506000e+02	3.500e-02
(1.1587 -0.19656 0)	6.517779e+02	1.038e-01
(1.00780 -0.1495 0)	6.551092e+02	2.985e-01
(0.90340 -0.1077 0)	6.572463e+02	4.235e-01
(0.75772 -0.04435 0)	6.605763e+02	6.191e-01
(0.59897 0.03612 0)	6.631425e+02	7.704e-01
(0.40229 0.17143 0)	6.710075e+02	1.236e+00
(0.31488 0.28046 0)	6.763816e+02	1.556e+00
(0.24702 0.50060 0)	6.820858e+02	1.901e+00
(0.23232 0.80537 0)	6.916585e+02	2.478e+00
(0.24126 1.09980 0)	7.034357e+02	3.190e+00
(0.26059 1.40270 0)	7.126542e+02	3.753e+00
(0.29635 1.68920 0)	7.252429e+02	4.509e+00
(0.37467 1.99110 0)	7.382810e+02	5.301e+00
(0.53853 2.26660 0)	7.527676e+02	6.201e+00
(0.80249 2.48180 0)	7.667572e+02	7.065e+00
(1.00660 2.60300 0)	7.725769e+02	7.348e+00
(1.1497 2.67050 0)	7.771482e+02	7.568e+00
(1.2346 2.70370 0)	7.815895e+02	7.898e+00

Mean and standard deviation for data points at symmetry wall

Samples (100)	Mean	Standard Deviation
(0 0.25 0)	6.890384e+02	2.278e+00
(0 0.50 0)	6.936764e+02	2.558e+00
(0 1.00 0)	7.042168e+02	3.205e+00
(0 1.50 0)	7.161763e+02	3.959e+00
(0 2.00 0)	7.367834e+02	5.299e+00
(0 2.50 0)	7.680797e+02	7.373e+00

Mean and standard deviation for data points at curved wall

Samples (100)	Mean	Standard Deviation
(1.158 0.099 0)	6.599942e+02	5.794e-01
(1.017 0.272 0)	6.907749e+02	1.993e+00
(0.858 0.513 0)	7.502994e+02	4.351e+00
(0.732 0.787 0)	7.457370e+02	4.642e+00
(0.670 1.020 0)	7.336410e+02	4.448e+00
(0.650 1.250 0)	7.425342e+02	4.906e+00
(0.691 1.577 0)	7.499974e+02	5.380e+00
(0.772 1.814 0)	7.581428e+02	5.867e+00
(0.895 2.049 0)	7.647287e+02	6.278e+00
(1.017 2.227 0)	7.713319e+02	6.701e+00
(1.158 2.400 0)	7.740242e+02	6.928e+00
(1.250 2.500 0)	7.753141e+02	7.108e+00

Mean and standard deviation for data points at top wall

Samples (100)	**Mean**	**Standard Deviation**
(0.181 2.553 0)	7.789526e+02	8.058e+00
(0.395 2.615 0)	7.826112e+02	8.272e+00
(0.625 2.683 0)	7.852473e+02	8.380e+00
(0.858 2.751 0)	7.865134e+02	8.388e+00
(1.074 2.814 0)	7.852376e+02	8.207e+00

APPENDIX C

Python tool developed for extracting fluidRegion and neutroRegion data from every nodal point in GeN-Foam

```
import os
file_list = os.listdir(".")
fluid_region_files = ["U", "T", "Re", "powerDensity", "p", "p_rgh", "nut", "k",
                                                    "epsilon"]
neutro_region_files = ["flux0", "flux1", "flux2", "flux3", "flux4", "flux5",
"fluxStar0", "fluxStar1", "fluxStar2", "fluxStar3", "fluxStar4", "fluxStar5",
"prec0", "prec1", "prec2", "prec3", "prec4", "prec5", "prec6", "prec7",
"precStar0", "precStar1", "precStar2", "precStar3", "precStar4", "precStar5",
"precStar6", "precStar7", "oneGroupFlux", "rhoCool", "TClad", "TCool", "TFuel",
"volFuelPower"]

fluid_region_files = ["U", "T", "powerDensity", "p"]
neutro_region_files = ["flux3", "prec0", "prec7", "oneGroupFlux", "rhoCool"]

types = ["fluidRegion", "neutroRegion"]

#fluid_timeslices = ["25", "50", "75", "100", "125", "150", "175", "200", "225", "250",
#"275", "300"]
fluid_timeslices = ["300"]
#neutro_timeslices = ["75", "100", "125", "150", "175", "200", "225", "250", "275",
#"300"]
neutro_timeslices = ["300"]
```

```
def process_file (file_name):
    # TODO: ignore if file not present
    values = []
    values_1 = []
    values_2 = []
    values_3 = []
    f = None
    try:
        f = open(file_name, "r")
    except:
        if file_name.split("/")[-1] == "U":
            return ([], [], [])
        else:
            return []
    read_file = f.read().split("\n")
    if file_name.split("/")[-1] == "U":
        # This is to handle velocity
        extract = False
        for line in read_file:
            l = None
            if extract:
                line = line.replace("(", "")
                line = line.replace(")", "")
```

```
        if line == "(":
            extract = True
        elif line == ")":
            extract = False
            break
        elif extract and len(line.split(" ")) > 2:
            split = line.split(" ")
            values_1.append(split[0])
            values_2.append(split[1])
            values_3.append(split[2])
    return (values_1, values_2, values_3)
else:
    extract = False
    for line in read_file:
        if line == "(":
            extract = True
        elif line == ")":
            extract = False
            break
        elif extract:
            values.append(line)
print(file_name, len(values))
return values
```

```
def max_dict(dictionary):
    lengths = []
    for key, value in dictionary.items():
        lengths.append(len(value))
    return max(lengths)


def write_csv_file (dictionary, file_name, fluid_or_neutro):
    import csv
    from itertools import zip_longest
    max_length = max_dict(dictionary)
    if fluid_or_neutro == "fluid":
        csv_array = list(zip_longest(dictionary["ux"],
                    dictionary["uy"],
                    dictionary["uz"],
                    dictionary["T"],


                    dictionary["powerDensity"],
                    dictionary["p"]))
        try:
            with open(file_name, "w", newline='', encoding='utf-8') as f:
                writer = csv.writer(f)
                writer.writerow(["Ux", "Uy", "Uz"] + fluid_region_files[1:])
```

```
            for row in csv_array:
                writer.writerow(row)
        except Exception as e:
            import traceback
            traceback.print_exc();
            print(e)
            return
    else:
neutro_region_files = ["flux0", "flux1", "flux2", "flux3", "flux4", "flux5",
"fluxstar0", "fluxstar1", "fluxstar2", "fluxstar3", "fluxstar4", "fluxstar5",
"prec0", "prec1", "prec2", "prec3", "prec4", "prec5", "prec6", "prec7",
"precStar0", "precStar1", "precStar2", "precStar3", "precStar4", "precStar5",
"precStar6", "precStar7", "oneGroupFlux", "rhoCool", "TClad", "TCool", "TFuel",
"volFuelPower"]
        #csv_array = list(zip_longest(*[ for i in neutro_region_files dictionary[i]]
        csv_array = list(zip_longest( dictionary["prec0"],
                    dictionary["prec7"], dictionary["oneGroupFlux"],
                    dictionary["rhoCool"]))
        try:
            with open(file_name, "w", newline='', encoding='utf-8') as f:
                writer = csv.writer(f)
                writer.writerow(neutro_region_files)
                for row in csv_array:
```

```
                writer.writerow(row)
        except Exception as e:
            import traceback
            traceback.print_exc();
            print(e)
            return


for f in file_list:
    if "Case" in f:
        case_files = os.listdir("./" + f)
        print(case_files)
        # fluid region first
        for time in fluid_timeslices:
            fluid_dict = {}
            neutro_dict = {}
            for fluid in fluid_region_files:
                if fluid == "U":
                    (ux, uy, uz) = process_file("./" + f + "/" + time + "/fluidRegion/" + fluid)
                    fluid_dict["ux"] = ux
                    fluid_dict["uy"] = uy
                    fluid_dict["uz"] = uz
                else:
                    values = process_file("./" + f + "/" + time + "/fluidRegion/" + fluid)
```

```
            fluid_dict[fluid] = values


    write_csv_file(fluid_dict, "./" + f + "/" + time + "/fluidRegion_out.csv", "fluid")
# neutro region next
for time in neutro_timeslices:
    for neutro in neutro_region_files:
        values = process_file("./" + f + "/" + time + "/neutroRegion/" + neutro)
        neutro_dict[neutro] = values
    write_csv_file(neutro_dict, "./" + f + "/" + time + "/neutroRegion_out.csv", "neutro")
```

APPENDIX D

Python tool developed for extracting data from GeN-Foam at different time intervals specified by the user

```
import os
import csv

file_list = os.listdir(".")

#Time0 = min(T) CN=78, max(t) CN=82, avg(T) CN=83, max(U (Magnitude)) CN=110, avg(U (Magnitude)) CN=111, (filename = respective variable)


# 4 files - fluid, neutro, time0, time4
# Case#{}
minimum_temperature = [["TimeStep", "25", "50", "75", "100", "125", "150", "175", "200", "225", "250", "275", "300"]]
maximum_temperature = [["TimeStep", "25", "50", "75", "100", "125", "150", "175", "200", "225", "250", "275", "300"]]
average_temperature = [["TimeStep", "25", "50", "75", "100", "125", "150", "175", "200", "225", "250", "275", "300"]]
maximum_onegroup_flux = [["TimeStep", "25", "50", "75", "100", "125", "150", "175", "200", "225", "250", "275", "300"]]
average_onegroup_flux = [["TimeStep", "25", "50", "75", "100", "125", "150", "175", "200", "225", "250", "275", "300"]]
minimum_onegroup_flux = [["TimeStep", "25", "50", "75", "100", "125", "150", "175", "200", "225", "250", "275", "300"]]
```

```
maximum_velocity = [["TimeStep", "25", "50", "75", "100", "125", "150", "175", "200", "225",
"250", "275", "300"]]
average_velocity = [["TimeStep", "25", "50", "75", "100", "125", "150", "175", "200", "225",
"250", "275", "300"]]
average_temperature = [["TimeStep", "25", "50", "75", "100", "125", "150", "175", "200", "225",
"250", "275", "300"]]



u_mag = []
temperature_3d = []
power_density = []
pressure = []
density = []
prec0 = []
prec7 = []



for f in file_list:
    if f == "Case42" or f == "Case43" or f == "Case44" \
    or f == "Case45" or f == "Case46" \
    or f == "Case47" or f == "Case48" or f == "Case49" \
    or f == "Case50" or f == "Case51" \
    or f == "Case53" or f == "Case55" or f == "Case56" \
```

```
or f == "Case57" or f == "Case58" or f == "Case59" \
or f == "Case60":

if f == "Case1" or f == "Case2" or f == "Case3" \
or f == "Case5" or f == "Case6" or f == "Case4" \
or f == "Case7" or f == "Case8" or f == "Case9" \
or f == "Case10" or f == "Case11" or f == "Case12" \
or f == "Case13" or f == "Case15" or f == "Case16" \
or f == "Case14" or f == "Case18" or f == "Case19" \
or f == "Case17" or f == "Case20":

if f == "Case21" or f == "Case22" or f == "Case23" \
or f == "Case25" or f == "Case26" \
or f == "Case27" or f == "Case28" or f == "Case29" \
or f == "Case30" or f == "Case31" or f == "Case32" \
or f == "Case33" or f == "Case35" or f == "Case36" \
or f == "Case34" or f == "Case38" or f == "Case39" \
or f == "Case40":

if f == "Case41" or f == "Case52" or f == "Case61" \
or f == "Case62" or f == "Case63" or f == "Case66" \
or f == "Case67" or f == "Case73" or f == "Case74" \
or f == "Case75" or f == "Case76" or f == "Case77" \
```

```
or f == "Case80" or f == "Case81" or f == "Case82" \
or f == "Case83" or f == "Case84" or f == "Case86" \
or f == "Case90" or f == "Case92" or f == "Case95" \
or f == "Case96" or f == "Case97" or f == "Case100" \
or f == "Case87" or f == "Case89":

if f == "Case24" or f == "Case37" or f == "Case54" \
or f == "Case64" or f == "Case65" or f == "Case68" \
or f == "Case69" or f == "Case71" or f == "Case72" \
or f == "Case78" or f == "Case79" or f == "Case85" \
or f == "Case88" or f == "Case91" or f == "Case93" \
or f == "Case98" or f == "Case99" or f == "Case70" \
or f == "Case94":



   with open(f + "/time0.csv", mode ='r') as file:

      csvFile = csv.reader(file)
      test = list(zip(*list(csvFile)))
      min_tmp_column = [f]
      min_tmp_column += list(test[78][1:])
      minimum_temperature.append(min_tmp_column)
```

```
        max_tmp_column = [f]
        max_tmp_column += list(test[82][1:])
        maximum_temperature.append(max_tmp_column)

        avg_tmp_column = [f]
        avg_tmp_column += list(test[83][1:])
        average_temperature.append(avg_tmp_column)

        max_vel_column = [f]
        max_vel_column += list(test[110][1:])
        maximum_velocity.append(max_vel_column)

        avg_vel_column = [f]
        avg_vel_column += list(test[111][1:])
        average_velocity.append(avg_vel_column)

    with open(f + "/time4.csv", mode ='r') as file:
        csvFile = csv.reader(file)
        test = list(zip(*list(csvFile)))

        #Time4 = max(oneGroupFlux) CN= 180, avg(oneGroupFlux) CN=181,
min(oneGroupFlux) CN=176 (filename = respective variable)
```

```
        max_onegroup_flux = [f]
        max_onegroup_flux = list(test[180][1:])
        maximum_onegroup_flux.append(max_onegroup_flux)

        avg_onegroup_flux = [f]
        avg_onegroup_flux = list(test[181][1:])
        average_onegroup_flux.append(avg_onegroup_flux)

        min_onegroup_flux = [f]
        min_onegroup_flux = list(test[176][1:])
        minimum_onegroup_flux.append(min_onegroup_flux)

    #
  #fluidRegion_out = (Ux,Uy,Uz)=Umag CN=0,1,2, T CN=3, powerDensity CN=4, p CN=5
(filename = 3D_respective variable)
    with open(f + "/300/fluidRegion_out.csv", mode ='r') as file:
        csvFile = csv.reader(file)
        # print(test)
        rows = []
        for line in list(csvFile)[1:]:
            cur_row = []
            #print(line)
```

```
            if line[0] == '':
                break
            import math
            sum = math.pow(float(line[0]), 2) + math.pow(float(line[1]), 2) +
math.pow(float(line[2]), 2)

            cur_row.append(math.sqrt(sum))
            cur_row += line[3:]
            rows.append(cur_row)

        test = list(zip(*list(rows)))

    # u_mag, temperature_3d, power_density, pressure
        tmp_umag = [f]
        tmp_umag = list(test[0])
        u_mag.append(tmp_umag)

        tmp_temperature = [f]
        tmp_temperature = list(test[1])
        temperature_3d.append(tmp_temperature)

        tmp_power = [f]
        tmp_power = list(test[2])
```

```
    power_density.append(tmp_power)

    tmp_pressure = [f]
    tmp_pressure = list(test[3])
    pressure.append(tmp_pressure)

#neutroRegion_out = prec0 CN=1, prec7 CN=2
with open(f + "/300/neutroRegion_out.csv", mode ='r') as file:
    csvFile = csv.reader(file)
    test = list(zip(*list(csvFile)))

    tmp_prec0 = [f]
    tmp_prec0 = list(test[1][1:])
    prec0.append(tmp_prec0)

    tmp_prec7 = [f]
    tmp_prec7 = list(test[2][1:])
    prec7.append(tmp_prec7)

    tmp_density = [f]
    tmp_density = list(test[3][1:])
    density.append(tmp_density)
```

```
min_tmp_zipped = list(zip(*minimum_temperature))

max_tmp_zipped = list(zip(*maximum_temperature))

avg_tmp_zipped = list(zip(*average_temperature))

max_vel_zipped = list(zip(*maximum_velocity))

avg_vel_zipped = list(zip(*average_velocity))

min_flux_zipped = list(zip(*minimum_onegroup_flux))

max_flux_zipped = list(zip(*maximum_onegroup_flux))

avg_flux_zipped = list(zip(*average_onegroup_flux))

prec0_zipped = list(zip(*prec0))

prec7_zipped = list(zip(*prec7))

density_zipped = list(zip(*density))

u_mag_zipped = list(zip(*u_mag))

temperature_3d_zipped = list(zip(*temperature_3d))

power_density_zipped = list(zip(*power_density))

pressure_zipped = list(zip(*pressure))



#------1to20---------


with open('minimum_temperature_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in min_tmp_zipped:
        wr.writerow(word)
```

```
with open('maximum_temperature_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_tmp_zipped:
        wr.writerow(word)
with open('average_temperature_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_tmp_zipped:
        wr.writerow(word)
with open('maximum_velocity_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_vel_zipped:
        wr.writerow(word)
with open('average_velocity_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_vel_zipped:
        wr.writerow(word)


with open('minimum_onegroup_flux_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in min_flux_zipped:
        wr.writerow(word)
with open('maximum_onegroup_flux_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
```

```
    for word in max_flux_zipped:
        wr.writerow(word)
with open('average_onegroup_flux_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_flux_zipped:
        wr.writerow(word)

with open('3d_precursor1_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in prec0_zipped:
        wr.writerow(word)

with open('3d_precursor8_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in prec7_zipped:
        wr.writerow(word)

with open('3d_density_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in density_zipped:
        wr.writerow(word)
with open('u_mag_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
```

```
    for word in u_mag_zipped:
        wr.writerow(word)


with open('temperature_3d_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in temperature_3d_zipped:
        wr.writerow(word)
with open('power_density_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in power_density_zipped:
        wr.writerow(word)
with open('pressure_1to20.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in pressure_zipped:
        wr.writerow(word)


#------21to40--------


with open('minimum_temperature_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in min_tmp_zipped:
        wr.writerow(word)
with open('maximum_temperature_21to40.csv', 'w', ) as myfile:
```

```
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_tmp_zipped:
        wr.writerow(word)
with open('average_temperature_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_tmp_zipped:
        wr.writerow(word)
with open('maximum_velocity_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_vel_zipped:
        wr.writerow(word)
with open('average_velocity_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_vel_zipped:
        wr.writerow(word)


with open('minimum_onegroup_flux_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in min_flux_zipped:
        wr.writerow(word)
with open('maximum_onegroup_flux_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_flux_zipped:
```

```
        wr.writerow(word)
with open('average_onegroup_flux_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_flux_zipped:
        wr.writerow(word)

with open('3d_precursor1_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in prec0_zipped:
        wr.writerow(word)

with open('3d_precursor8_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in prec7_zipped:
        wr.writerow(word)

with open('3d_density_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in density_zipped:
        wr.writerow(word)
with open('u_mag_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in u_mag_zipped:
```

```
        wr.writerow(word)


with open('temperature_3d_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in temperature_3d_zipped:
        wr.writerow(word)
with open('power_density_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in power_density_zipped:
        wr.writerow(word)
with open('pressure_21to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in pressure_zipped:
        wr.writerow(word)


----------41to60----------


with open('minimum_temperature_41to40.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in min_tmp_zipped:
        wr.writerow(word)
with open('maximum_temperature_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
```

```
    for word in max_tmp_zipped:
        wr.writerow(word)
with open('average_temperature_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_tmp_zipped:
        wr.writerow(word)
with open('maximum_velocity_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_vel_zipped:
        wr.writerow(word)
with open('average_velocity_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_vel_zipped:
        wr.writerow(word)


with open('minimum_onegroup_flux_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in min_flux_zipped:
        wr.writerow(word)
with open('maximum_onegroup_flux_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_flux_zipped:
        wr.writerow(word)
```

```
with open('average_onegroup_flux_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_flux_zipped:
        wr.writerow(word)

with open('3d_precursor1_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in prec0_zipped:
        wr.writerow(word)

with open('3d_precursor8_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in prec7_zipped:
        wr.writerow(word)

with open('3d_density_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in density_zipped:
        wr.writerow(word)

with open('u_mag_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in u_mag_zipped:
```

```
        wr.writerow(word)


with open('temperature_3d_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in temperature_3d_zipped:
        wr.writerow(word)
with open('power_density_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in power_density_zipped:
        wr.writerow(word)
with open('pressure_41to60.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in pressure_zipped:
        wr.writerow(word)


# ------------61to100----------------


with open('minimum_temperature_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in min_tmp_zipped:
        wr.writerow(word)
with open('maximum_temperature_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
```

```
    for word in max_tmp_zipped:
        wr.writerow(word)
with open('average_temperature_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_tmp_zipped:
        wr.writerow(word)
with open('maximum_velocity_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_vel_zipped:
        wr.writerow(word)
with open('average_velocity_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_vel_zipped:
        wr.writerow(word)


with open('minimum_onegroup_flux_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in min_flux_zipped:
        wr.writerow(word)
with open('maximum_onegroup_flux_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_flux_zipped:
        wr.writerow(word)
```

```
with open('average_onegroup_flux_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_flux_zipped:
        wr.writerow(word)

with open('3d_precursor1_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in prec0_zipped:
        wr.writerow(word)

with open('3d_precursor8_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in prec7_zipped:
        wr.writerow(word)

with open('3d_density_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in density_zipped:
        wr.writerow(word)
with open('u_mag_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in u_mag_zipped:
        wr.writerow(word)
```

```
with open('temperature_3d_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in temperature_3d_zipped:
        wr.writerow(word)
with open('power_density_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in power_density_zipped:
        wr.writerow(word)
with open('pressure_61to100.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in pressure_zipped:
        wr.writerow(word)

# ------------RestofCases----------------

with open('minimum_temperature_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in min_tmp_zipped:
        wr.writerow(word)
with open('maximum_temperature_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_tmp_zipped:
```

```
        wr.writerow(word)
with open('average_temperature_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_tmp_zipped:
        wr.writerow(word)
with open('maximum_velocity_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_vel_zipped:
        wr.writerow(word)
with open('average_velocity_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_vel_zipped:
        wr.writerow(word)


with open('minimum_onegroup_flux_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in min_flux_zipped:
        wr.writerow(word)
with open('maximum_onegroup_flux_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in max_flux_zipped:
        wr.writerow(word)
with open('average_onegroup_flux_restofcases.csv', 'w', ) as myfile:
```

```
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in avg_flux_zipped:
        wr.writerow(word)

with open('3d_precursor1_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in prec0_zipped:
        wr.writerow(word)

with open('3d_precursor8_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in prec7_zipped:
        wr.writerow(word)

with open('3d_density_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in density_zipped:
        wr.writerow(word)
with open('u_mag_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in u_mag_zipped:
        wr.writerow(word)
```

```
with open('temperature_3d_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in temperature_3d_zipped:
        wr.writerow(word)
with open('power_density_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in power_density_zipped:
        wr.writerow(word)
with open('pressure_restofcases.csv', 'w', ) as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    for word in pressure_zipped:
        wr.writerow(word)
```