Optimizing Thermal Protection Systems for Hypersonic Reentry: Design and Simulation Analysis

a project presented to
The Faculty of the Department of Aerospace Engineering
San José State University

in partial fulfillment of the requirements for the degree
Master of Science in Aerospace Engineering

by

Jax A. Williams

May 2025

approved by

Dr. Yawo Ezunkpe
Faculty Advisor

ABSTRACT

Optimizing Thermal Protection Systems for Hypersonic Reentry: Design and Simulation Analysis

Jax A. Williams

This study investigates the optimization for thermal protection systems (TPS) for hypersonic spacecraft reentry through a combination of computational modeling and simulation-based validation. A genetic algorithm (GA) framework was developed in MATLAB to optimize multilayer TPS configurations by minimizing the bottom surface temperatures while balancing mass efficiency and realistic design. Both flat and angled layer geometries were analyzed to explore the potential of thermal refraction through internal material structuring. Results demonstrated strong agreement between methodologies, supporting the effectiveness of angled TPS configurations in improving thermal performance. The findings offer insights into designing more reliable, efficient, and robust TPS structures for future hypersonic reentry vehicles.

Acknowledgements

I would like to extend my sincere gratitude to Dr. Yawo Ezunkpe, Dr. Maria Chierichetti, and Dr. Nikos Mourtos for their invaluable guidance and support throughout my graduate studies. Their expertise in aerospace structures, hypersonics, and aerospace engineering applications have been instrumental in shaping the development of this work. I am particularly thankful for their guidance, patience, and encouragement throughout my years at the university promoting my curiosity, passion, and professional development as a student during my time at San José State University.

Table of Contents

List of Tables

List of Figures

Table of Symbols

Symbol	Definition	Units
T	Temperature	K
q	Heat flux	W/m^2
ρ	Density	kg/m^3
Cp	Specific heat capacity	J/kg·K
K	Thermal conductivity	W/m·K
α	Thermal diffusivity	m^2/s
σ	Stefan-Boltzmann constant	$W/m^2 \cdot K^4$
ε	Emissivity	-
t	Thickness	m
Δx	Grid spacing in x-direction	m
Δy	Grid spacing in y-direction	m
Δt	Time Step	s
L	Domain width	m
x	Horizontal spatial coordinate	m
y	Vertical spatial coordinate	m

1. Introduction

1.1 Motivation

The ever-increasing demand for hypersonic flight and reusable launch vehicles (RLVs) has demonstrated the need for the advancement of thermal protection systems (TPS) capable of withstanding the extremely harsh environments faced upon hypersonic re-entry. These environments consist of intense temperatures upwards of 3000°F (≈1649°C) [1] and large mechanical loads due to the aerodynamic drag these vehicles are exposed to. Current TPS designs often rely on heavy, complex, and expensive materials that have a significant impact on a spacecraft's performance and design.

The primary challenge in TPS optimization is finding the delicate balance between weight and thermal performance. Reducing the weight of the TPS is crucial for optimal spacecraft performance, however, the reduction in material tends to result in a loss of thermal protection. Another key challenge engineers face when designing TPS for hypersonic atmospheric entry vehicles is ensuring that the material can withstand the extreme environment multiple times. Reusability is becoming the ultimate design choice for modern spacecraft for several reasons including economic benefits, environmental benefits, increase in mission frequency, commercial viability, and more. Reusable launch vehicles and other reusable spacecraft require TPS that can maintain their performance over several missions while withstanding repeated exposure to high thermal and mechanical loads.

To develop innovative TPS solutions, computational fluid dynamics (CFD) and finite element analysis (FEA) are increasingly being implemented. These types of programs can simulate complex fluid dynamics, heat transfer phenomena, and the structural response of a spacecraft in hypersonic atmospheric entry. However, simulating an accurate prediction of the physical properties of the TPS as it undergoes reentry can be extremely difficult due to the complexity of heat transfer phenomena and the involved interactions between the TPS, fluid flow, and the spacecraft's internal components. In addition to these programs, data-driven methods are also applied for TPS optimization utilizing various optimization algorithms [10,12,13].

Combining modern methodologies with the power of computational simulations may yield more optimal TPS configurations. This research focuses on exploring the potential of optimizing the internal geometric structuring of multilayer passive TPS using established aerospace materials. Rather than relying on novel materials or active TPS techniques, this study investigates how angled layer configurations can passively improve thermal performance by redirecting heat flow internally. By leveraging computational tools such as computational fluid dynamics simulations, finite difference modeling, and optimization algorithms, this work aims to develop TPS designs that minimize the bottom surface temperatures while maintaining mass efficiency. The results of this research could have significant implications for the future of aerospace engineering, promoting the development of more efficient, sustainable, and affordable hypersonic spacecraft.

1.2 Literature Review

1.2.1 Overview of Thermal Protection Systems (TPS) in Spacecraft

The Thermal Protection System (TPS) is an essential subsystem of spacecraft, designed to protect the internal components from the high temperatures and extreme aerodynamic forces

experienced during various phases of spaceflight, including atmospheric entry and re-entry at hypersonic speeds. The TPS is designed to dissipate the heat generated in these harsh environments, mitigating damage to the spacecraft body. Current technologies for TPS, such as modern ceramics and ablative materials, have proven to be effective but still face several challenges in cost, weight, and overall durability. These design challenges include minimizing the mass while optimizing structural integrity, minimizing manufacturing and materials costs, integrating active system components, maintaining TPS functionality for reusable launch vehicles (RLVs), and possibly more depending on the mission objectives. The typical TPS design consists of high-temperature materials that, when combined create a 'smooth' and 'aerodynamic' surface that can face temperatures up to ≈1700°C (≈3000°F) [1]. Since not all spacecraft missions require an aerocapture maneuver as high temperature as mentioned previously, TPS can be designed with drastic differences when compared to each other.

1.2.2 Types of TPS

Given the natural differences between spacecraft designs and mission objectives, there are several types of TPS designs depending on the final objectives. This project will focus on re-entry vehicles traveling at hypersonic speeds (Ma ≥ 5). With the general design objectives being to minimize mass while maintaining structural and thermal integrity, TPS can be categorized into three distinct categories: passive methods, active methods, and semi-passive methods [2]. Each method is applied to different aerospace applications, which will be described and summarized in the following sub-sections. Additionally, there is a wide range of materials that can be used for practical purposes in TPS. These parameters can vary from material/TPS sizing, type of material (ceramic, metallic, composite, etc.), and the layer count for TPS with multiple layers of protection [3]. To reiterate, these design choices all vary depending on the technology available for the project, the propulsion systems, the geometry of the vehicle, and the mission objectives including time and environment. The technologies that have already been explored will be recounted in this section.

1.2.2.1 Passive TPS

Passive TPS refers to thermal barriers constructed from materials exhibiting favorable thermal properties with no cooling systems present in the design. The passive method of TPS design is constructed to be either thick or robust enough to resist the high temperatures and extreme aerodynamic loads it will face upon hypersonic re-entry. This type of design may also be referred to as a heat sink, hot structure, or insulated structure [2]. Passive methods rely solely on the mechanical and thermal properties of the materials that engineers choose for their mission design. A few select types of materials are primarily chosen for the passive method, those materials include ablative materials, radiative materials, and metallic heat sinks.

Ablative materials, also known as sacrificial materials, are designed to melt or decompose when exposed to extreme temperatures, utilizing the decomposition of the material to absorb excess heat on the vehicle. Although all ablative materials serve the same function, there are 3 main types of ablative materials: charring, melting, and intumescent [4]. Some examples of ablative materials include Phenolic Impregnated Carbon Ablator (PICA), PICA-X, Silicone Impregnated Reusable Ceramic Ablator (SIRCA), Toughed Uni-piece Fibrous Reinforced Oxidation Resistant Composite (TUFROC), as well as many others [2,4].

Radiative materials are materials that reflect or radiate heat energy back into space, reducing the amount of heat that the TPS would absorb with non-radiative materials. Radiative

materials are an essential TPS component, however radiative materials are typically found in RLVs or space shuttles as these materials are not typically required for brief exposure to ultra-high temperatures. Radiative materials can be combined with ablative materials to create a multilayer thermal protection system (MTPS) [3,5]. Some examples of radiative materials used in ultra-high temperature environments include Silicon Dioxide (SiO2), Silicon Carbide (SiC), Boron Carbide (B4C), as well as many others.

The final types of passive TPS design to be discussed are metallic heat sinks. Metallic heat sinks are not designed to melt or reflect heat, rather they are designed to absorb and store the heat energy away from the critical components of the spacecraft. Similar to radiative materials, metallic heat sinks are typically combined with other types of materials to optimize the ratio of heat absorption and heat dissipation, ensuring the critical components of the spacecraft are unharmed through the duration of the flight. Heat sinks are typically only viable for short duration flights or else the system is at risk of overheating [2]. Some examples of metallic heat sink materials are aluminum alloys, copper alloys, and sometimes even silver alloys. When designing a passive TPS, the specific requirements of the mission at hand are what determines how the TPS is designed.

1.2.2.2 Active TPS

Active thermal protection systems are heat shields that utilize active cooling mechanisms to dissipate heat energy more effectively than a passive TPS would. Active TPS tend to rely on external power sources to create a flow of coolant throughout the structure of the TPS. However, there are 3 categorized active TPS methods that are being currently used. These methods include convective cooling, transpiration cooling, and film cooling [2].

Convective cooling pumps a coolant fluid underneath the skin of the TPS. Although this type of active cooling system can be found in many varieties of power consuming technologies, it still comes with many engineering challenges, specifically for aerospace applications. Some of these challenges can include optimizing mass and weight distribution, maintaining the plumbing system for the coolant, allocating enough power for the system to work, and other reliability issues. Types of coolant fluids that are used for these systems include water, glycol-based mixtures, hydrogen fuel [8], and other fluids.

Similar to convective cooling, transpiration cooling also utilizes coolant fluid to improve the thermal properties of the system. However, the difference between convective cooling and transpiration cooling is in the form of how the coolant gets pumped throughout the system. Instead of being run through tubes or channels, transpiration cooling emits the coolant through a porous surface to help cool the outer surface of the TPS. Convective cooling is utilized for long duration flights or maneuvers withstanding extremely high temperatures ($\geq$1649°C) [2]. This is because this kind of cooling system is extremely difficult to create and maintain, however, the cooling properties are superb because it attempts to maintain a steady state on the system for the entire duration of flight. This method of active cooling uses a similar coolant to the convective cooling method.

Lastly, film cooling is an active TPS method that injects the coolant into a tactical place onto the surface of the system. The type of coolant that is used for this type of active TPS is specifically designed to create a thin film on the outer surface of the system to give it an additional layer of protection. This additional layer of protection has been studied extensively, yielding more efficient results for higher speeds upon re-entry. One setback with this method of active cooling is

that it is highly dependent on the geometry of the vehicle and TPS. Despite this setback, it is still well regarded for its thermal properties and light weight in comparison to the other methods.

1.2.2.3 Semi-Passive TPS

A semi-passive TPS is a combination of passive TPS and active TPS, often using a working fluid or heat pipes to help dissipate heat over time. Semi-passive systems offer increased thermal efficiency in comparison to passive cooling systems all without the setbacks of needing an external power supply or moving components when comparing to an active cooling system.

To set them apart from passive TPS configurations, semi-passive TPS does not solely rely on their material properties to absorb and radiate heat. Instead, it incorporates mechanisms that can be controlled or modulated to improve their thermal protection properties. This type of TPS configuration is achieved in various ways, such as controlling the rate of flow of a chosen coolant or by adjusting the orientation of insulated layers. These additional mechanisms allow for more customizability depending on the desired purpose of the TPS. This level of control allows TPS configurations to optimize their performance and minimize the risk of overheating or structural failure.

The adaptability of semi-passive systems is not nearly as controllable as active thermal protection systems; however, this does offer improvements in certain areas where passive TPS lack in performance or reliability. This type of TPS is particularly useful for RLVs which are meant to withstand multiple atmospheric reentries while encountering varying atmospheric conditions. To get the optimal performance out of modern TPS, a semi-passive system is often used in collaboration with passive TPS and active TPS when attempting to withstand extreme temperatures [8]. Overall, semi-passive TPS offers a balance between passive and active systems offering increased thermal resistance compared to passive systems while remaining more lightweight and less energy-intensive than fully active configurations.

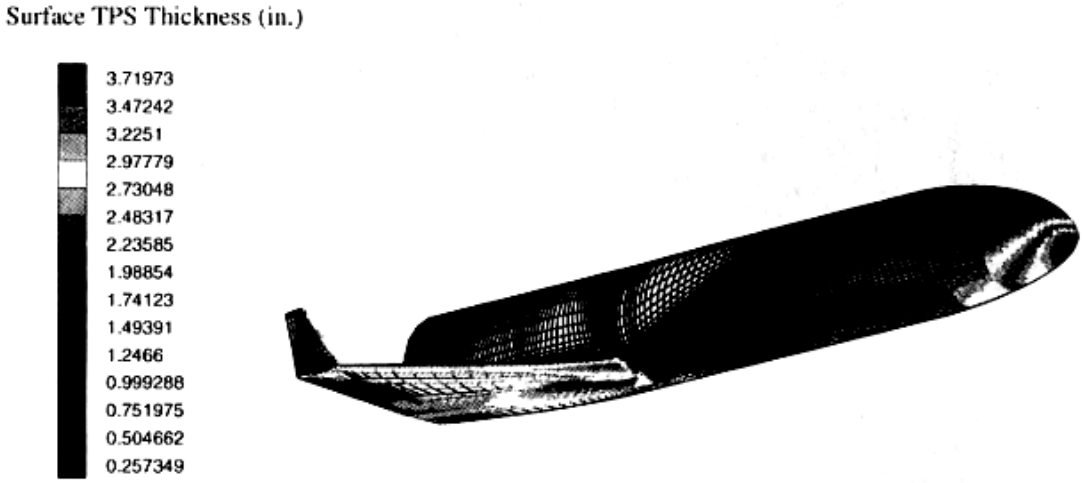
1.2.3 Previous studies

1.2.3.1 TPS Sizing for Access-to-Space Vehicles [9]

A comprehensive study conducted by NASA Ames Research Center focused on optimizing TPS sizing for RLVs, utilizing Navier-Stokes flow simulations combined with thermal analysis. The goal of the study was to identify, create, and validate simulation techniques for optimal material selection and sizing for TPS. Since this study was conducted via computational simulation by NASA, it is crucial to replicate real-life conditions as closely as possible to ensure the validity of the results. Therefore, for the studies conducted in CFD and FEA they chose the physics of the simulations to be fully viscous, chemically reacting (hypersonic), Navier-Stokes flow, and they analyze both laminar and turbulent flow. For this study, the vehicle being analyzed is a fixed wing single-stage-to-orbit (SSTO) shuttle, where they analyze the thermal reaction over the entire body of the vehicle.

The results of this study provide detailed material selections and thickness sizing for the SSTO vehicle. Using a trajectory based hypersonic, Navier-Stokes solution database, TPS designs were obtained for both laminar and turbulent flows. The study uses a base selection of materials such as tailorable advanced blanket insulation (TABI). Utilizing CFD software, the authors generated visual data and figures showing optimal TPS thickness, stagnation heating, and altitude-velocity charts. With a laminar trajectory, the TPS areal mass density was found to be 1.2 lbm/ft^2 while with a turbulent trajectory the areal mass density of the TPS was found to be upwards of 1.3

lbm/ft^2. The conclusion from the study states that the thickness of the TPS sizing for TABI material should be 1.5-2.0 inches, with thickness sizing ranging from 0.25 inches all the way to over 3.5 inches of thickness. The study presents the optimal TPS sizing based on the results of their simulation through a distribution map shown in Figure 1.1.



Top Layer TPS Thickness (in.) for the LaRC Winged Body SSTO Vehicle (Total Heating Time, 6200 sec)

(TURBULENT FLOW SOLUTION)

Figure 1.1: TPS thickness sizing distribution map for turbulent flow [9]

This study used a SSTO RLV, which faces unique challenges during hypersonic reentry as the vehicle configuration is designed for a lifting reentry. A lifting reentry descent exposes the spacecraft to prolonged heat loads and typically does not experience an extreme aerodynamic load due to the strategy at which the vehicle approaches Earth. This varies from a ballistic reentry strategy, where the reentry vehicle enters the atmosphere at hypersonic speeds and attempts to slow down as fast as possible. With a ballistic reentry the vehicle, the vehicle tends to experience higher heat loads with greater amounts of heat flux while also needing to withstand extreme aerodynamic forces up to 12 g's [12]. Figure 1.2 compares the ballistic and lifting reentry strategies for hypersonic atmospheric reentry. The similarities between these strategies are the speed at which they reenter the Earth's atmosphere and the extreme temperatures they face. The key differences between the 2 are the deceleration loads they each face, the amount of heat flux that each type of reentry faces, and how they approach their landing sequences. These differences significantly influence TPS design requirements and are central to the approach of the work demonstrated in this thesis.

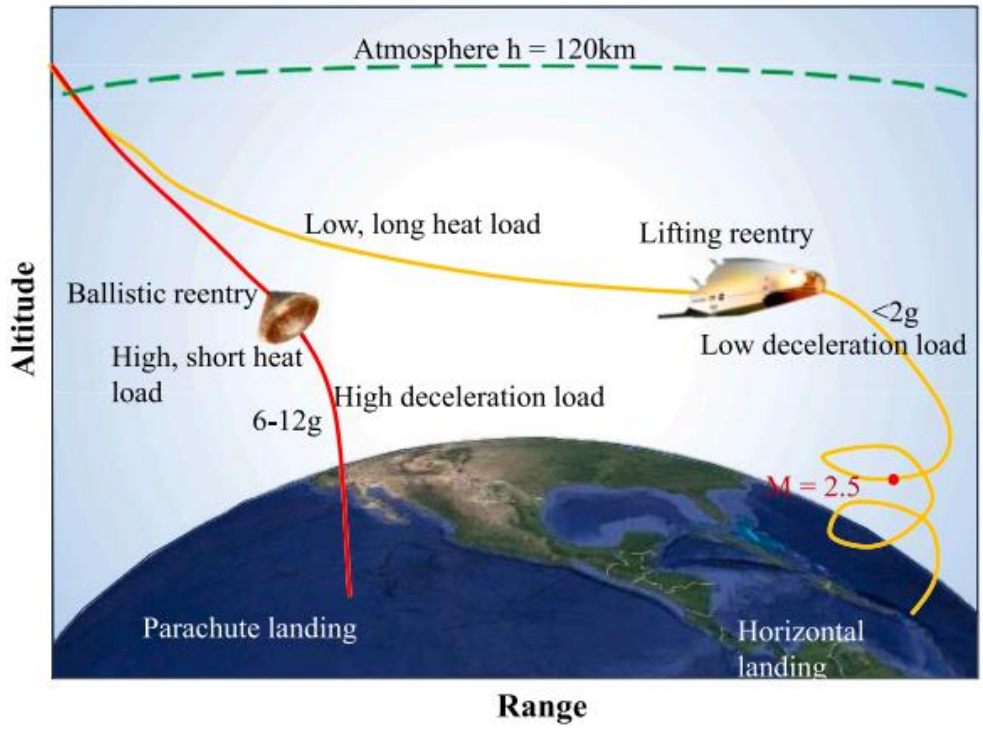


Figure 1.2: Lifting reentry vs ballistic reentry. Reproduced with permission from [12]. © 2024 Elsevier.

1.2.3.2 Thermomechanical Optimization of TPS [10]

This study develops a method to minimize the mass per unit area of an integrated thermal protection system (ITPS) using a sandwich panel design. The research discusses the structural soundness of the TPS as it faces not only extreme temperatures but extreme pressures, which shifts the focus to the structural integrity of the TPS rather than focusing solely on the thermal properties of TPS. In contrast to the aforementioned studies, this paper has the objective of developing an analytical procedure to optimize the mass per unit area of a sandwich panel TPS design. To achieve the objective of their work, they utilize the use of FEM software to analyze heat transfer across the TPS layers.

The study uses a two-step optimization practice where the first step is to perform a transient thermal analysis on the corrugated-core sandwich structure of the ITPS. The next step is to optimize the geometry (excluding the optimized height of the core found in step 1) to optimize the structural sizing, keeping in mind the weight you're adding to the vehicle. The algorithm used in this optimization process is known as the globally convergent method of moving asymptotes (GCMMA). GCMMA has the capability to describe the behavior of non-uniform functions. They used FEM software to model realistic scenarios of TPS reentry with defined initial conditions. Since the TPS geometry strongly affects performance and varies by vehicle, the authors used 2D models to reduce computational cost and time. For a 2D analysis of TPS of various sizes, they implement the following assumptions in their study: there is no temperature variation in the direction along the outer surface, the lower surface on bottom faceplate is assumed to be perfectly insulated, the radiation is applied to the upper surface of the top faceplate with an emissivity of

0.85, the initial temperature of the model is constant, and the radiative and convective heat transfer through insulation is ignored (only conductive heat transfer is taken into account).

Optimizing various material and geometry combinations yielded practical insights into TPS performance trade-offs. The study analyzes materials such as Aluminosilicate/Nextel 720 composites, Saffil® insulation, and Epoxy/carbon fiber laminate. Finding a balance of necessary mechanical load efficiency, thermal load efficiency, and overall mass of the TPS was able to be determined. The study established a design procedure for optimizing a minimum weight TPS designed for RLV's. Using ANSYS Parametric Design Language (ADPL) code with the GCMMA algorithm could easily determine the optimal TPS after several iteration steps. The validity of this approach is constrained by its simplifying assumptions, which may not reflect localized failure modes such as buckling. Overall, the optimized weight of the ITPS was found to be around 18.5 kg/m^2, which is approximately 37% lighter than the initial design of the ITPS which was around 29.4 kg/m^2. Despite the thin thickness of the materials chosen for this ITPS, the maximum mechanical stress applied to the ITPS under FEM simulations was 67.9 MPa. This maximum load is significantly less than the yielding strength of the material chosen for the simulation. Therefore, it was concluded that given a uniform pressure and thermal load the ITPS designed for this optimization study will not fail given the optimal dimensions post-analysis.

1.2.3.3 Optimization of TPS by using Phase Change Material [3]

This paper introduces phase change materials (PCM) into multilayer thermal protection systems (MTPS) to optimize the heat shield in terms of thermal performance and structural integrity using 2D FEM models. This study utilizes PCM which are materials that undergo a phase change under extreme thermal loads and return to normal after cooling. A PCM is a substance that stores and releases thermal energy by changing from one physical state to another, such as solid to liquid or vice versa. Engineers typically use PCM in active and semi-active TPS due to the lower melting point of PCM compared to ablative materials seen in passive TPS designs. However, PCM offers greater reliability than ablative materials because it can be reused under the correct conditions. PCM is also denser than typical TPS materials, making it less favorable in terms of mass efficiency.

Their methodology included 2D models with 3 to 4 sections in the MTPS with constant thickness on the bottom/faceplate as well as a variable thickness in the middle layer(s) containing the PCM. The study compares a purely conductive model to a convective model for in-depth analysis and comparison. The assumptions for their simulation include the following: the initial temperature of the whole system is assumed to be constant and uniform, there is no variation in the lateral direction for the heat flux load and the thermal radiation on the top surface, the two sides and bottom of the system are adiabatic, liquid PCM motions are laminar, Newtonian and incompressible, volume expansion of PCM during phase transition is neglected, and materials used are taken as homogeneous and isotropic. To ensure realistic results from their 2D simulations, they incorporate real flight data to define the boundary conditions and initial conditions.

The TPS design excluding PCM failed under thermal and mechanical loading, while all PCM-enhanced models successfully passed the conducted simulations. However, they found that for each PCM they use they all succeeded though the tests with varying results depending on the

material choice. All PCM enhanced configurations withstood the testing conditions, though performance varied by material.

1.3 Project Objective

The primary objective of this thesis is to investigate and optimize passive multilayer thermal protection systems (MTPS) for hypersonic spacecraft undergoing ballistic reentry. Specifically, the project explores the concept of thermal refraction, in which internal TPS layers are angled relative to the direction of incoming heat flux to passively redirect thermal energy in the lateral direction. This passive control aims to improve thermal performance and minimize temperatures experienced at the spacecraft's interior surfaces. The work in this thesis establishes a theoretical foundation based on existing research in induced anisotropy, thermal metamaterials, and geometric heat manipulation while applying these ideas to the design and analysis of innovative angled MTPS configurations.

The computational framework will be leveraged by utilizing MATLAB's genetic algorithm capabilities, which systemically identifies optimal TPS configurations by minimizing the bottom surface temperatures while maintaining practical constraints related to material selection, layer geometry, mass efficiency, and physical feasibility. These configurations will then be validated using comprehensive CFD simulations to ensure physical stability and reliability of the proposed optimal MTPS designs.

To support and verify the optimization results, a custom finite difference model is developed in MATLAB to offer a more transparent and detailed verification of the heat conduction behavior within these angled configurations. Ultimately, this thesis aims to demonstrate a clear understanding of how geometry can be utilized as a design parameter in passive MTPS systems and to offer insights that may guide the development of more efficient, reusable, and reliable thermal aerospace designs.

1.4 Methodology

This research uses a multi-phase, redundant methodology to optimize multilayer thermal protection systems (MTPS) for hypersonic reentry vehicles. The approach integrates material selection, computational optimization, simulation-based validation, and manual verification to ensure consistency, reliability, and thermal efficiency of the proposed MTPS configurations.

1.4.1 Material Selection

A preliminary literature review investigated a wide range of materials commonly used in passive MTPS designs, including ablatives, insulators, radiative surfaces, and composite structures. While there are many advanced materials that exist within these TPS categories, acquiring reliable thermal property data for modern aerospace materials proved difficult due to proprietary restrictions or insufficient publicly available data. Due to this constraint, this thesis focuses on five materials for which thermal conductivity, specific heat capacity, and density values sourced from verified public sources [20-25].

As a result, the materials chosen primarily fall into two main material categories: composites and insulators. Specifically, the three composites used were carbon-carbon (C/C), carbon-silicon carbide (C/SiC), and silicon carbide-silicon carbide (SiC/SiC), while the two types of insulators chosen are Nextel™ and Saffil®. These are among the most prominent types of materials used in passive TPS design, representing similar material properties to those chosen in real-world practice. While the material selection for this research does not include all traditional TPS materials, the selected materials still reflect realistic design constraints encountered in TPS design.

1.4.2 Genetic Algorithm Optimization

A MATLAB-based genetic algorithm (GA) using the PDE Modeler Toolbox was implemented to simulate two-dimensional transient heat conduction. The GA evaluated MTPS configurations based on material order, layer angle (0°–10°), total thickness (25–50 mm), and layer count (5–11 layers, per geometry). Angled configurations were modeled with angled interior layers (parallelograms) bounded by triangular boundary layers to maintain a normal top/bottom surface, while flat configurations stacked rectangular layers in an orderly fashion.

Each configuration was initialized at 300 K, with a fixed 3000 K boundary applied to the top surface to simulate ballistic reentry conditions. The GA incorporated penalty and reward logic to discourage unrealistic configurations while encouraging thermally efficient, realistic designs. Parallel processing and MATLAB's flexibility enabled rapid simulation and evaluation of a wide range of MTPS configurations.

1.4.3 CFD Simulation-Based Validation

The optimal configurations were reconstructed in ANSYS Fluent using a two-dimensional transient heat conduction model to validate the GA results. A structured quadrilateral mesh (0.5mm resolution) and identical material and boundary conditions ensured comparability. Post-processing in Fluent enabled detailed contour and gradient visualizations to verify lateral heat spread and lower bottom surface temperatures in angled MTPS configurations.

1.4.4 Manual Verification Using 2D Finite Difference Method

For additional physical insight into the proposed angled-layer heat refraction mechanism, a custom two-dimensional finite difference simulation was developed in MATLAB. This model recreated both the flat and angled-layer geometries and solved the transient heat conduction equation using explicit time-stepping and spatial discretization. This verification model provided greater transparency into heat flow behavior and served as a reference for verifying simulation results and deepening physical understanding of the process.

This methodology ensures that the analysis of produced MTPS designs is robust by combining optimization, simulation, and physical modeling. Each part contributes to the results and supports the proposed concept that angled configurations can enhance the thermal performance of a passive MTPS through geometric heat refraction.

2. Theoretical Basis for Angled-Layer Heat Refraction

2.1 Introduction

Traditional TPS designs are typically constructed of multilayered composites arranged in flat, perpendicular stacks, with materials selected to resist heat primarily from normal (perpendicular) flow directions. While this design approach has been successfully implemented in various hypersonic applications, it is constrained by factors such as material cost, mass limitations, and performance trade-offs with increasing TPS thickness.

This study introduces the idea of thermal refraction, in which angling the internal layers of a TPS configuration may redirect the heat laterally, analogous to how light refracts between media of different refractive indices. By leveraging geometric design rather than relying solely on material thermal properties, it may be possible to reduce the temperature at the inner surface of the spacecraft or payload more efficiently, without increasing the TPS thickness.

The goal of this chapter is to establish a theoretical basis for this heat refraction hypothesis. It will explore relevant literature on induced anisotropy, two-dimensional directional heat flow, and studies related to thermal and optical refraction. This theoretical background established the foundation for analyzing the performance of angled-layer TPS configurations in aerospace contexts.

2.2 Literature Support

Various recent studies support the thermal refraction concept that is explored in this thesis. While traditional thermal protection systems focus on material composition and total mass, emerging research in metamaterials, anisotropic conduction, and thermal field manipulation opens a new realm of possibilities to more effectively control heat using geometry. This section highlights the directional behavior of heat in artificially anisotropic structures in theory, practice, and future applications.

2.2.1 Computational Optimization of Directional Heat Conduction [27]

A study by Dede (2010) supports the concept of directional thermal control through geometric and material design by using FEA to simulate and optimize heat flow in anisotropic structures. The main objective of this study was to minimize thermal resistance across a fixed domain by spatially varying the material's thermal conductivity tensor. The optimization approach redirected heat along more efficient conduction paths which enhances passive thermal management without adding material or changing boundary conditions.

The framework for Dede's optimization uses a gradient-based algorithm to assign local conductivity orientations and magnitudes throughout the fixed domain. This allows the algorithm to target optimally efficient heat transfer paths from a defined heat source to a cooling surface. Results demonstrated that directional conduction in two-dimensional space, achieved through anisotropy, could significantly improve the thermal performance of a material compared to its nominally isotropic counterpart. Simulations confirmed that thermal flow could be actively engineered and controlled through manipulative design.

Although the work in this article focuses on continuous anisotropic materials rather than layered composites, the primary principle of controlled thermal conduction redirection by the means of induced anisotropy is directly relevant to the angled-layer configurations explored in the work of this thesis. Where Dede achieved anisotropy through spatially controlled material properties, the approach in this thesis achieves similar effect by using fixed-material layers oriented at various angles. Both of these methods share and demonstrate the goal of controlling heat distribution within an internal structure to optimize thermal efficiency of a fixed configuration.

The technical and computational work provided by Dede supports the broader idea that heat conduction is not constrained to an orthogonal flow through material(s) and directional orientation of material properties can serve as a design tool for fine-tuning heat flow through a material or protection system. The computational foundation of this work supports the idea that angled MTPS geometries can be used to redirect heat flow and enhance thermal efficiency, especially with the use of induced anisotropy.

2.2.2 Heat Flux Through Metamaterials [18]

The article *Guiding conductive heat flux through thermal metamaterials* explores the concept of induced anisotropy to enhance the thermal properties of materials. These engineered thermal metamaterials are specifically designed for thermal protection applications. These metamaterials demonstrate unique directional flow properties that are not typical to materials of the same composition. By inducing anisotropy in nominally isotropic materials, such as copper or steel, engineers can alter the internal structure of a material to control the direction of heat flow throughout a desired configuration.

The authors of this article created induced anisotropic metamaterials by stacking layers of copper and steel on top of each other in increments of thickness ~0.3cm. These layers were arranged in three different types of orientation angles at $\theta = -45^{\circ}$, $\theta = 0^{\circ}$, and $\theta = 45^{\circ}$. The metamaterials were stacked and embedded into a brass plate so they could apply an evenly distributed heat throughout the material in a singular direction, simulating the thermal loads typical of hypersonic ballistic reentry environments. Experimentally testing each orientation angle of the material, they were able to record the heat propagation over time through the metamaterial for each configuration, which can be depicted in Figure 2.1.

The objective of using various angled orientations was to create an effective thermal medium (EMT) through the means of creating metamaterials rather than using an abundance of material for proper thermal protection. After obtaining and analyzing the results of the experimental studies, it was concluded that orienting the materials at certain angles can prove to be a more effective thermal medium than a non-angled material would. Figure 2.1 represents the experimental results achieved for each orientation, demonstrating much greater thermal properties in the angled materials compared to the non-angled material. These results demonstrate the value of induced anisotropy for the development of metamaterials in thermal applications.

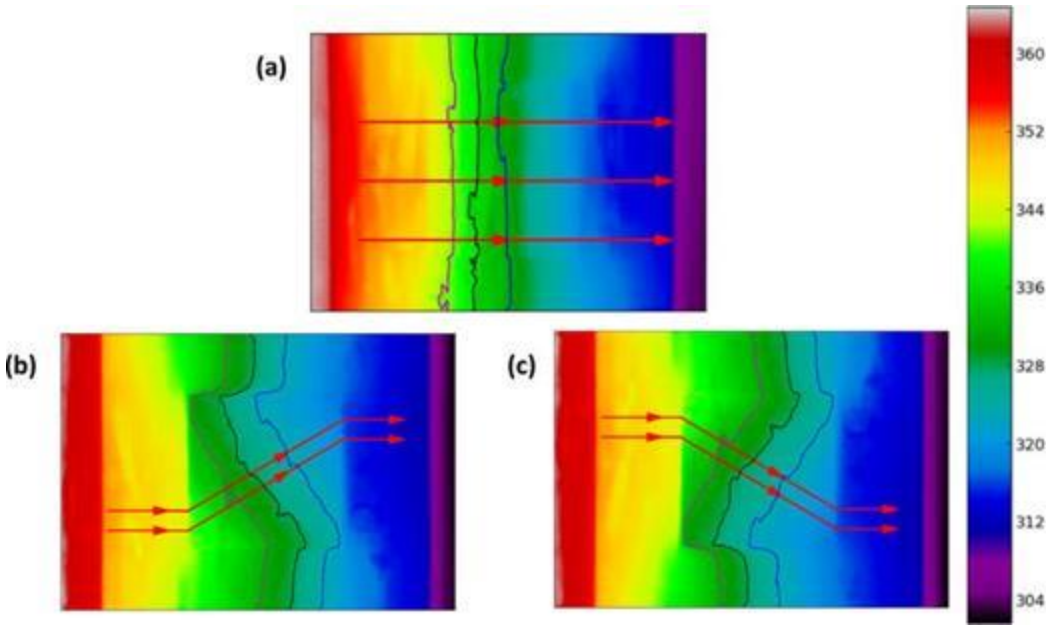


Figure 2.1: Experimental results for (a) θ = 0°, (b) θ = 45°, and (c) θ = −45°. Reproduced with permission from [18].

Furthermore, the article expands on the reaction of the materials over time with exposure to heat transfer. The authors emphasize the analogy of thermal conduction behaving similar to how light behaves in a medium, where it can refract through different materials and can potentially be redirected in a controlled manner. This idea is expanded upon by suggesting the implementation of thermal concentrators and thermal cloaks. The article suggests that with these devices applied to a metamaterial, the material could be designed to have a temperature gradient of 0 in a cloaked region while directing the heat elsewhere with thermally concentrated regions.

This article provides unique and important information on the forefront of metamaterials being used in thermal protection applications. While the experimental temperatures (~100°C) are significantly lower when compared to a hypersonic atmospheric reentry (~2000°C), the study established foundational principles for directing heat through engineering anisotropy, potentially enabling enhanced aerospace thermal designs.

2.3 Refraction Analysis

Geometrically controlled heat conduction is supported by the well-documented physical behavior of heat in fixed, multi-material, and multilayer domains. This thesis investigates how angled multilayer configurations use internal structural geometry to actively influence the directional path of thermal energy. This concept draws inspiration from preexisting physical principles of optical refraction, particularly Snell's Law as shown in Equation 2.1.

$$n_1 \sin\theta_1 = n_2 \sin\theta_2 \tag{2.1}$$

As discussed in 2.2.1 [27], heat behaves similarly to light by following the path of least resistance. This suggests that heat, like light, can be directed and controlled through structured configurations, enabling thermal pathways to be tailored through geometric and material design. This is especially true when the configuration uses multiple materials with different thermal media.

In a solid conduction domain, heat flows along temperature gradients which are governed by both the material's thermal properties and domain's geometrical layout. Fourier's Law, (Equation 2.2) defines the relationship between heat flux, material conductivity, and temperature gradient. In this form, $\vec{q}$ represents the heat flux, **K** is the conductivity tensor, and ∇T is the temperature gradient. This equation allows us to analyze the flow of heat conduction through a fixed domain over time.

$$\vec{q} = -\boldsymbol{K} \cdot \nabla T \tag{2.2}$$

In a multilayer configuration with flat, aligned layers of isotropic materials, the conduction is primarily one-dimensional with no notable lateral variation throughout the configuration. However, when layers are angled and composed of various materials, the interaction between conductivity and temperature gradient induces a lateral heat flux, resulting in multidirectional heat flow. This behavior is described by the two-dimensional form of Fourier's Law, which is demonstrated in Equation 2.3.

$$\vec{q}\,(x, y) = -k\left(\frac{\partial T}{\partial x}\hat{\imath} + \frac{\partial T}{\partial y}\hat{\jmath}\right) \tag{2.3}$$

This equation captures the heat flux vector of both the lateral and the normal directions, demonstrating the inherently two-dimensional nature of heat conduction in angled, multi-material configurations of different thermal conductivities.

When heat encounters a material boundary at an angle, especially within the domains of varying conductivity, the path of heat flux bends in the direction of the induced angle of the internal geometry. While this is not true refraction in the same way as optics, as there is no wave propagation, the resulting thermal behavior of the domains resembles geometric flux control. This resemblance to optical refraction offers string potential for designing thermal structures that precisely guide heat in aerospace and other high-performance systems.

The effectiveness of this mechanism is demonstrated with computational simulations using both MATLAB and ANSYS Fluent. Even with computational models, directional control of internal heat flux is clear when altering internal geometry. The effects of layer angle and thickness on directional conduction are analyzed in detail in later chapters of the thesis.

2.4 Aerospace Implications

The practical implications of the thermal refraction mechanism explored in this thesis have the potential to be applied to several aerospace applications, specifically for ballistic reentry conditions. In those missions, a reentry vehicle focuses extreme thermal loads concentrated on one side of the vehicle. Since this thermal concentration is inherent to ballistic reentry, TPS designs must prioritize thermal efficiency. Additionally, spacecraft are inherently mass-sensitive systems, meaning minimizing weight and material usage can result in savings on fuel, cost, and performance parameters.

By implementing internal geometric variance that redirects heat laterally, angled MTPS configurations may allow engineers to guide thermal energy toward less sensitive regions of the vehicle, such as edges of the spacecraft or radiative surfaces. In more advanced applications this approach may enable semi-active thermal management systems, where redirected thermal energy is routed toward systems designed to absorb, store, or reuse thermal energy.

Although the manufacturing of angled metamaterials in aerospace-grade composite materials presents a significant challenge but is within the realm of feasibility. Additive manufacturing and advanced composite layup techniques [1,4] are continuously evolving, driving the possibility of fabricating anisotropic and angled thermal configurations. Additionally, certain isotropic materials can exhibit directional heat flow when anisotropy is induced through controlled structural processing [see 2.2.2]. These developments may help improve the reliability and efficiency of thermal protection and redirection systems.

Conventional TPS design systems primarily focus on optimizing materials and thickness, with geometric considerations often simply conforming to specific spacecraft geometry. While material selection is always fundamental for TPS configuration and performance, the work in this thesis suggests that the geometry of the TPS configuration can be used as a design parameter to improve thermal performance, reduce overall material usage, and create a new axis of optimization in the design process.

Finally, the principles behind this heat refraction mechanism can be extended to a wide range of thermal systems in aerospace and beyond. These directional conduction strategies can be integrated into launch vehicles, orbital platforms, hypersonic vehicles, and other thermal applications. By integrating these technologies into future TPS designs, it may be possible to achieve greater thermal control, better structural efficiency, and reduced reliance on material abundance.

3. MATLAB Optimization Framework and Results

3.1 Introduction

The optimization process of TPS for hypersonic reentry has several unique challenges due to the complexity of TPS design, which involves interactions between materials, thermal gradients, and geometric configurations. To address these challenges, computational tools and analytical methods are used to efficiently explore design and parameter choices. Specifically, MATLAB is an efficient computational tool that allows for flexible and robust optimization capabilities. MATLAB is applied in this study to effectively design and analyze multilayer TPS configurations.

This chapter outlines the procedures and applications of the MATLAB framework implemented to identify an optimal multilayer TPS design. Varying parameters such as layer sizing, total thickness, layer count, and material choice allow the MATLAB framework to minimize the heat transfer to the spacecraft's interior while satisfying design constraints of the given system. The structure of the MATLAB work integrates thermal analysis models based on Fourier's law of heat conduction. Combining the structure of the MATLAB work with a partial differential equations modeler enables the potential of rapidly designing and evaluating various TPS configurations for multiple layers.

The following sections explain the initial set up of the optimization problem, the algorithm implementation, and the results achieved from the MATLAB script. The optimized designs developed within this chapter will be tested and refined by utilizing detailed CFD simulations based on the results obtained from the script. Prior to the CFD validation, this chapter outlines the setup, execution, and results of the genetic algorithm (GA) optimization process.

3.2 Optimization Framework

The objective of the optimization framework is to minimize the heat flux and temperature through the TPS to unshielded spacecraft. Five high-performance materials were selected as the material database based on an initial set of 19 candidate materials initially identified during preliminary review. The materials in this database include carbon-carbon composite (C/C), carbon-silicon carbide composite (C/SiC), silicon carbide-silicon carbide composite (SiC/SiC), Saffil®, and Nextel™ 312. Materials were ranked by using an initial efficiency metric (Equation 3.1), defined as the ratio of thermal diffusivity and density.

$$E_{MAT} = \frac{\alpha}{\rho} \tag{3.1}$$

In the above efficiency metric, it favors materials that combine a high thermal resistance with a lower mass. However, the optimization process itself does not rely directly on this efficiency metric during evaluation. Instead, material arrangements are evaluated dynamically during each generation of the GA based on thermal performance and realistic design constraints. The primary

fitness metric is the average bottom surface temperature at the end of the simulation duration. Penalties are applied to discourage physically unrealistic configurations, including:

- Placing insulation materials near the hot side
- Selecting bottom materials with poor insulating properties
- Excessive system mass
- Unfavorable thermal conductivity gradients

This approach allows the GA to freely explore combinations while guiding the selection process toward practical and effective TPS designs.

In parallel, a reward system was implemented to reinforce top-performing MTPS configurations with exceptionally low bottom surface temperatures. This dynamic evaluation allows the optimization to balance thermal protection effectiveness with mass efficiency and engineering conventions, producing practical MTPS designs aligning with aerospace design principles. The properties of the selected material database are summarized in Table 3.1.

Table 3.1 – Finalized material list for MATLAB implementation

Material Name	**Density (kg/m³)**	**Specific Heat Capacity (J/kg·K)**	**Thermal Conductivity (W/m·K)**	**Thermal Diffusivity (m²/s)**	**Efficiency Metric (E_{MAT})**
C/C	1750	850	20	1.68E-05	9.60E-09
C/SiC	2100	800	30	1.88E-05	8.95E-09
SiC/SiC	2980	760	180	7.87E-05	2.64E-08
Saffil	100	1000	25.6	2.56E-04	2.56E-06
Nextel	96	1050	0.112	1.12E-06	1.17E-08

Once the choice of materials is determined and the properties of each material are clearly defined, design variables are the next parameter to be decided for the framework of the optimization process. The parameters of the TPS chosen to be fixed included the layer sizing, total thickness of the TPS, and number of layers. The layer sizing was defined as a constant 5 mm per layer. The value of 5 mm was chosen for several reasons, the first of which was that the TPS design focused on realistic ballistic reentry profiles into Earth's atmosphere. The Apollo Command Module served as inspiration for the TPS sizing decision in this optimization study. This capsule

consisted of a TPS that ranged in total thickness from approximately 0.5 inches to 2 inches (~13 to 51 mm) [19] with several layers of material, varying depending on the location of the TPS on the spacecraft. To ensure the simulation is as realistic as possible, a small layer thickness similar to that of the Apollo mission was chosen. In addition, the algorithm permits repeated selection of the same material across layers to optimize effective thickness. This allows for the algorithm to try the same material multiple times in a row to attempt to figure out the best thickness of the chosen material.

The number of layers is another case that is being analyzed for optimization, however, the framework of the optimization is designed to define a set number of layers before attempting to optimize the TPS design. Due to the nature of the framing of the optimization, the number of layers was defined to be 5 through 10 layers. This means that 5 simulations were optimized based on the constant layer thickness of 5 mm. The total TPS thickness was computed based on the number of layers and the layer thickness in the TPS design. The total thickness was calculated as the product of the number of layers and the layer thickness (5 mm). Since layer thickness is defined as a constant, the total thickness of the TPS is 25 mm, 30 mm, 35 mm, 40 mm, 45 mm, and 50 mm for layer counts 5 through 10, respectively.

A two-dimensional heat conduction model based on Fourier's Law was applied to compute thermal resistance throughout the TPS. This was defined in the MATLAB script using built-in PDE modeling tools and applications which will be elaborated upon in chapter section 3.3. Although the physics models used the aid of additional programs, the objective function of the optimization was defined explicitly to minimize the total heat flux through the MTPS.

Due to the fundamentally different geometric structures of flat and angled TPS configurations, two separate but equivalent MATLAB scripts were developed. While the generation of the geometry varied between the flat and angled cases, the underlying logic of the optimization, material selection, and fitness evaluation methods remained the same. This approach allowed consistent results between configurations while sparing computational expense, preserving geometric accuracy across simulations.

To begin the optimization process based on these initial conditions, the optimization algorithm used for this particular setup was a genetic algorithm (GA). This algorithm is a built-in function of MATLAB and is well-suited for an optimization problem for heat transfer. This is because the GA is able to solve complex, nonlinear, and multidimensional optimization problems. The complexity of heat flux behavior makes it difficult to solve using simple analytical equations, making a GA better suited for exploring a large design space. Similarly, this algorithm was chosen due to its ability to handle a wide variety of variables and constraints. The GA can penalize solutions that are not realistic which help deduce an optimal configuration design with a given material list. The overall effectiveness of the GA is the ultimate reason for it being the primary choice of optimization given the objective function and constraints. With the optimization framework defined, the implementation and execution of the GA are described next.

3.3 MATLAB Implementation

MATLAB was selected for the optimization process due to its built-in optimization and PDE modeling tools, robust visualization capabilities, and overall flexibility. MATLAB allowed efficient implementation of the optimization process, enabling modeling of conductive heat transfer through MTPS configurations.

The optimization process for this algorithm used a fitness metric primarily focused on minimizing the average bottom surface temperature throughout a designated duration. In addition to minimizing bottom temperatures, the optimization framework also applied penalties to discourage physically unrealistic designs. Configurations were penalized for inefficient material placement, such as placing insulative materials in high temperature regions, which would realistically lead to material degradation and a potential system failure. Additional penalties were assigned for designs with excessive total mass, encouraging the algorithm to favor lightweight and more ideal MTPS configurations to be used practically. Similarly, a reward system was also introduced, granting bonuses to designs that achieved lower bottom surface temperatures, encouraging efficient solutions that remained physically feasible.

The genetic algorithm was configured with the following parameters:

- A population size of 300 individuals
- 30 total generations
- A mutation rate of 0.1
- An early stopping criterion triggered after 7 generations with no improvement

Each individual in the population is a fully configured TPS design, defined as a list of material assignments corresponding to each assigned layer. Every configuration is run through a two-dimensional heat conduction simulation, applying a boundary temperature of 3000 K on the exposed top surface and an initial condition of 300 K across the domain. These boundary temperatures were chosen to simulate realistic physical conditions of a ballistic reentry vehicle, experiencing temperatures of ~3000 K for up to 240 seconds while attempting to maintain a safe temperature of ~300 K. After each simulation, the resulting configurations and average bottom surface temperatures are extracted, evaluated with a fitness metric, and adjusted through the rewards and penalty system. Following the execution of this optimization process, the resulting configurations and figures are presented in the following section.

3.4 Results and Discussion

To maintain consistency, the mesh was parametrized to include four elements per 5mm layer thickness. This ensured uniform mesh density across simulations, regardless of the layer count. However, higher layer counts increased computational demand due to finer meshing requirements. Figure 3.1 shows a representative mesh for the 10-layer flat configuration, illustrating at least four triangular cells per layer. Since the framework of this optimization relied on the built-in tool of the PDE Modeler in MATLAB, there was no choice to create a structured

mesh consisting of quadrilaterals as a triangular mesh was the only choice available. Despite this drawback, a mesh sizing of 4 cells in between each layer is considered sufficient when attempting to balance computational cost and producing accurate results.

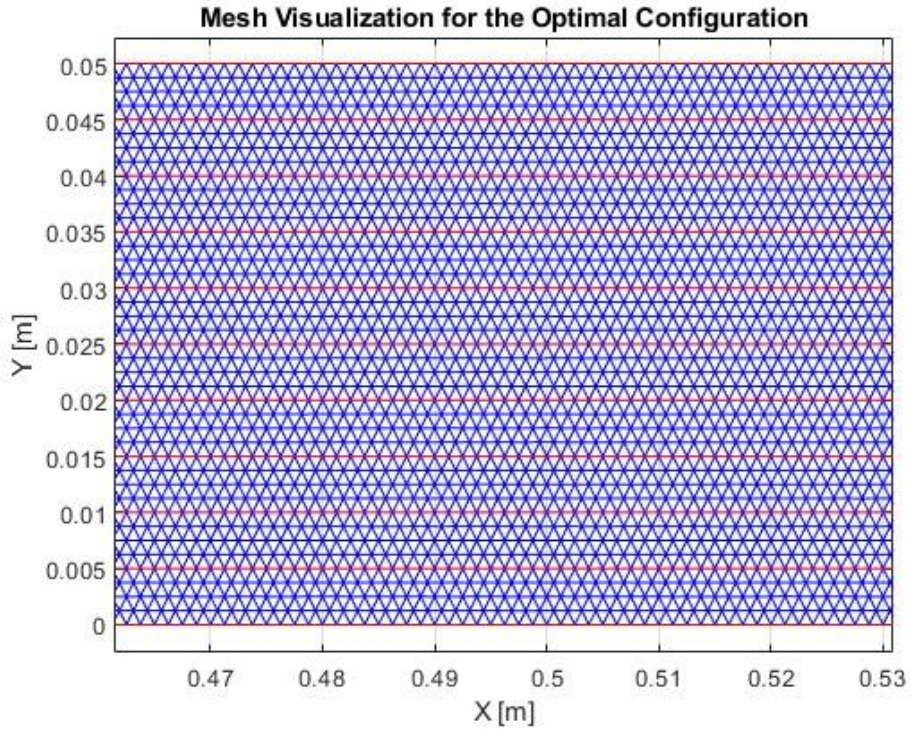
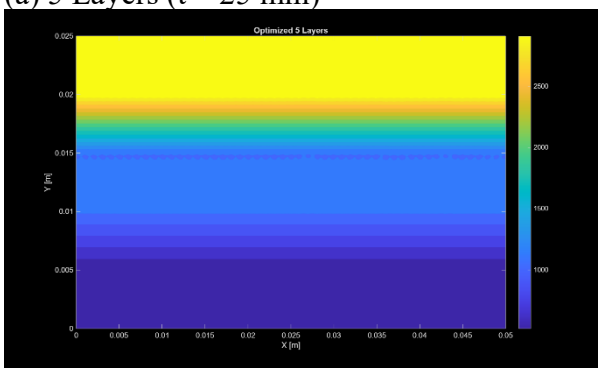


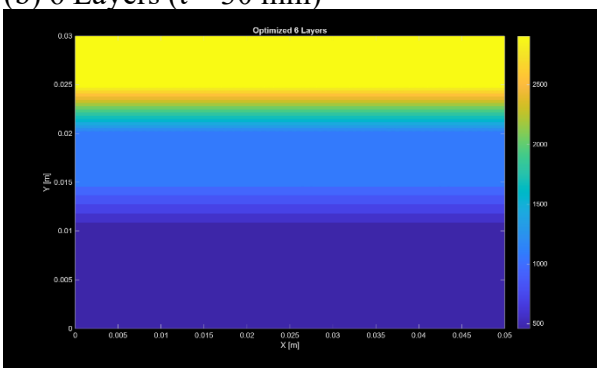
Figure 3.1 – Mesh visualization

Figure 3.2 presents the temperature distribution for the optimal configuration at each layer count. As layer count increases, bottom surface temperatures decrease, indicating improved thermal resistance. This is expected, as additional layers increase thermal impedance across the TPS configuration. Notably, Figures 3.2(e) and 3.2(f) show smoother temperature gradients, suggesting more effective heat dissipation.

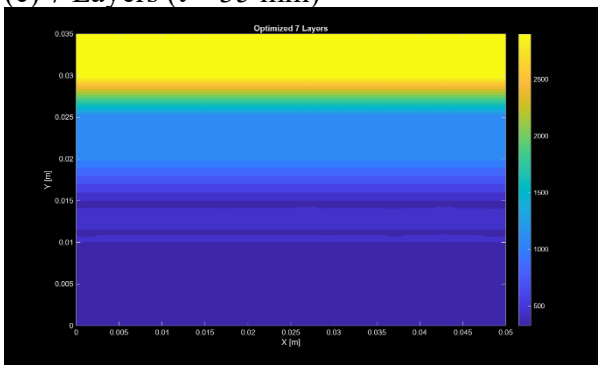
(a) 5 Layers (t = 25 mm)



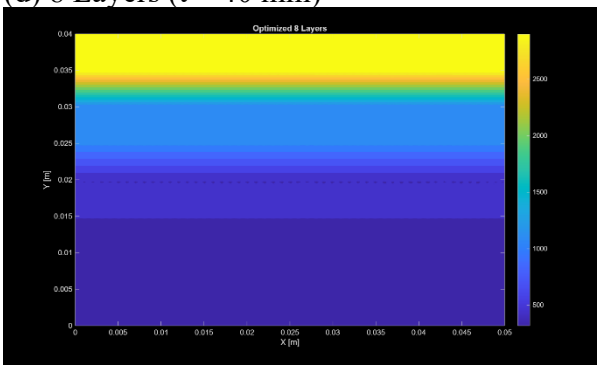
(b) 6 Layers (t = 30 mm)



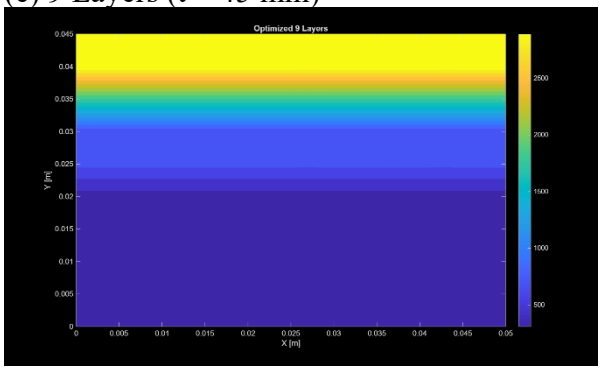
(c) 7 Layers (t = 35 mm)



(d) 8 Layers (t = 40 mm)



(e) 9 Layers (t = 45 mm)



(f) 10 Layers (t = 50 mm)

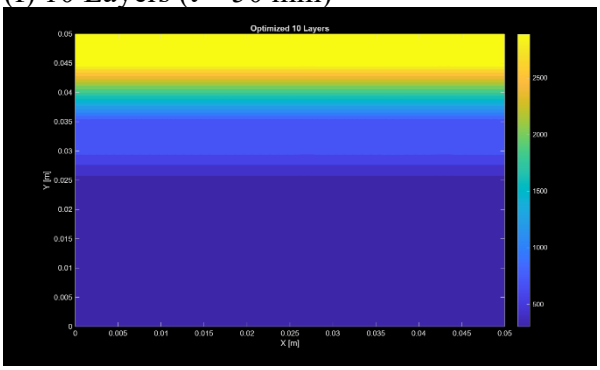
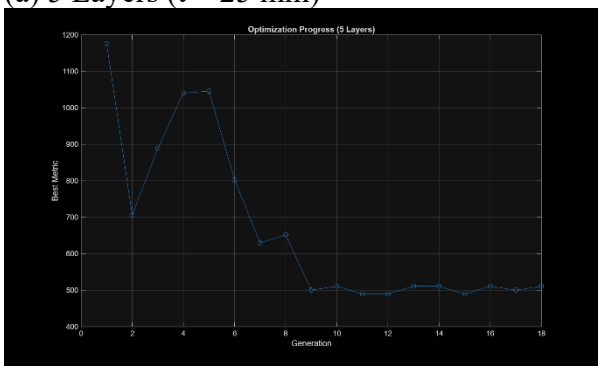


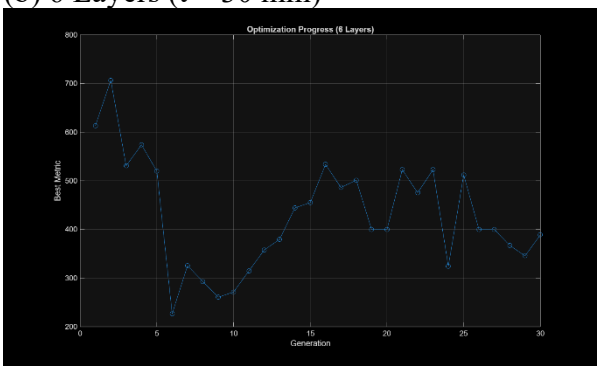
Figure 3.2 – Final temperature distribution plot for optimized configurations (5-10 Layers, 0°)

Figure 3.3 demonstrates the convergence behavior of the fitness metric over successive generations for each optimized configuration. As expected, performance improves over the generations until convergence is achieved or the limit of 30 generations is exceeded. Convergence occurs more rapidly for configurations with higher layer counts, reflecting the additional thermal resistance introduced by thicker MTPS designs.

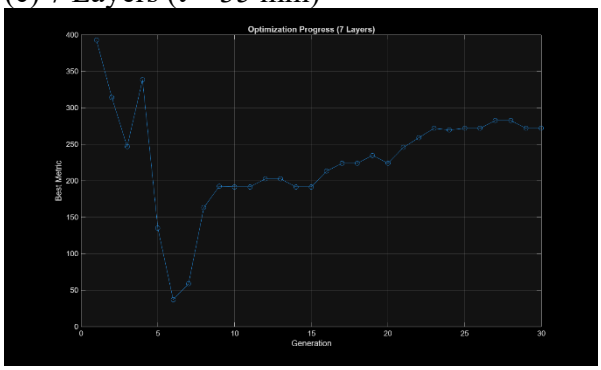
(a) 5 Layers (t = 25 mm)



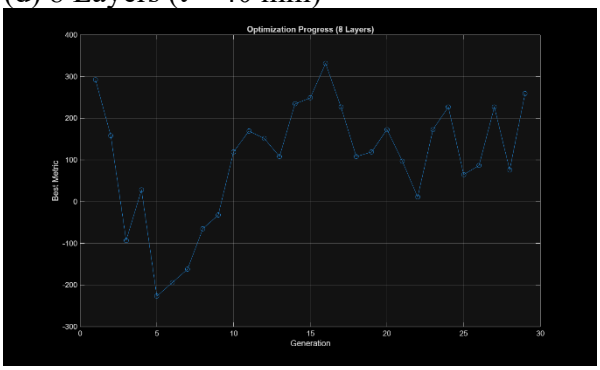
(b) 6 Layers (t = 30 mm)



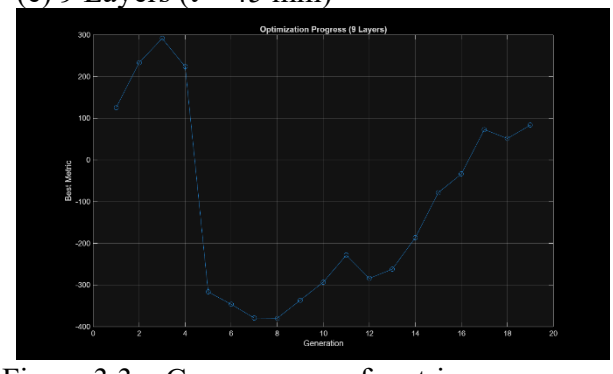
(c) 7 Layers (t = 35 mm)



(d) 8 Layers (t = 40 mm)



(e) 9 Layers (t = 45 mm)



(f) 10 Layers (t = 50 mm)

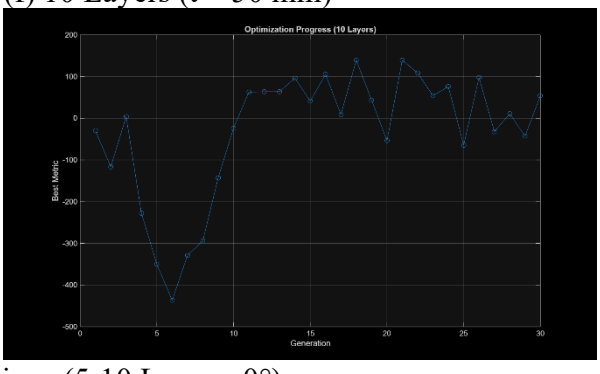


Figure 3.3 – Convergence of metrics over generations (5-10 Layers, 0°)

Table 3.2 is a chart visualization of the optimal configurations for each layer count as determined by the optimization algorithm. Each layer count was run through multiple generations to determine an optimal configuration. As each layer count converged with time, a noticeable trend was determined as demonstrated in Figures 3.2 and 3.3. This trend continues with the configuration patterns. Layer 1 is always a composite layer, which remains consistent with real life, and the material of choice is a silicon carbide composite (SiC/SiC). Although SiC/SiC is the densest material on the list included in the optimization, it also demonstrates the most ideal thermal diffusivity of the list. Due to its high thermal diffusivity, SiC/SiC is preferred for top-layer applications where incoming heat reaches ~3000 K, as it quickly spreads thermal energy away from the exposed surface. The trend of carbon composites followed by insulation material also follows the trend of reality and expectations, determining Nextel as the fiber of choice due to its lower density and thermal conductivity.

Overall, the results produced in Table 3.2 follow all expectations when compared to previous results and real previous TPS configurations. Note that Layer 1 corresponds to the bottom surface, adjacent to the interior of the spacecraft, while the highest number layer corresponds to the exposed top surface. The following tables summarize the optimal material configuration and best achieved results for each layer count and configuration angle.

To ensure a consistent comparison between flat and angled TPS configurations, an additional set of flat geometries were created based on matching the layer counts used in the angled configurations. Since the angled geometries with equivalent total thicknesses naturally require an additional layer to ensure uniformity, the corresponding flat “equal layer count” (ELC) cases were simulated with 6-11 layers of variable thickness instead of 5-10 layers with constant thickness. This allowed for direct comparison between flat and angled configurations of equivalent layer counts, ensuring a fair evaluation of MTPS improvements provided by angled layering.

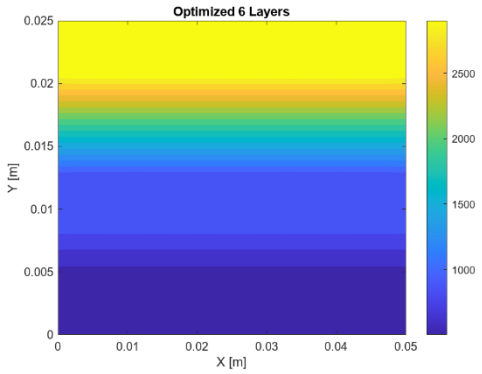
Table 3.2 – List of optimal configurations for each layer count (0°)

	5 Layers	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**
Layer 1	C/SiC	Nextel	C/SiC	Nextel	Nextel	Nextel
Layer 2	Nextel	SiC/SiC	Nextel	SiC/SiC	SiC/SiC	SiC/SiC
Layer 3	SiC/SiC	Nextel	SiC/SiC	Nextel	Nextel	Nextel
Layer 4	Nextel	SiC/SiC	Nextel	SiC/SiC	SiC/SiC	Nextel
Layer 5	SiC/SiC	Nextel	SiC/SiC	Nextel	Nextel	SiC/SiC
Layer 6	-	SiC/SiC	Nextel	SiC/SiC	SiC/SiC	Nextel
Layer 7	-	-	SiC/SiC	Nextel	Nextel	SiC/SiC
Layer 8	-	-	-	SiC/SiC	Nextel	Nextel
Layer 9	-	-	-	-	SiC/SiC	Nextel
Layer 10	-	-	-	-	-	SiC/SiC

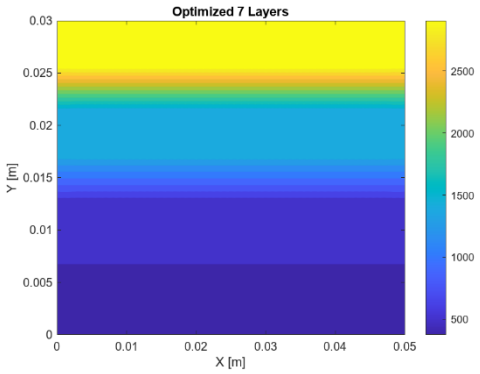
Table 3.3 – Best temperature metric for each layer count (0°)

Layer Count	**5 Layers**	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**
Best Temp. Metric	528.2573K	459.5984K	329.5659K	319.5182K	308.8937K	304.129K

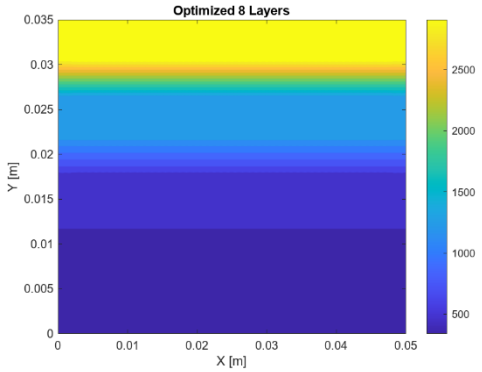
(a) 6 Layers (t = 25 mm)



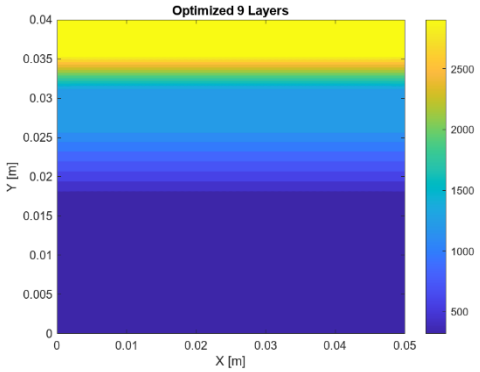
(b) 7 Layers (t = 30 mm)



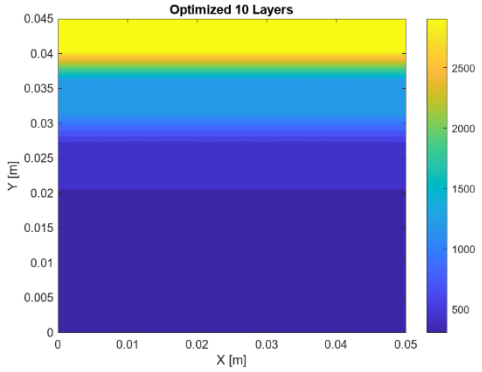
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

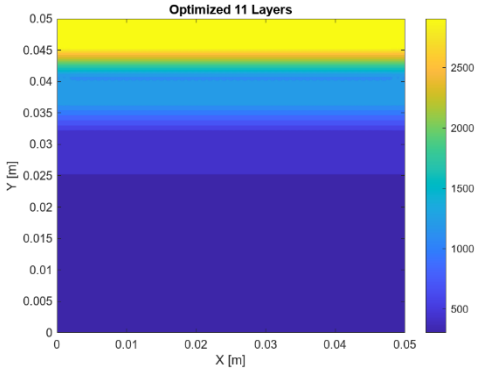
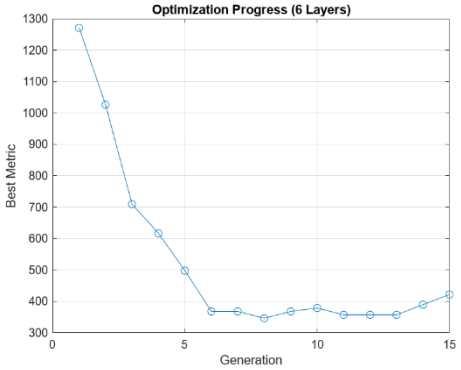


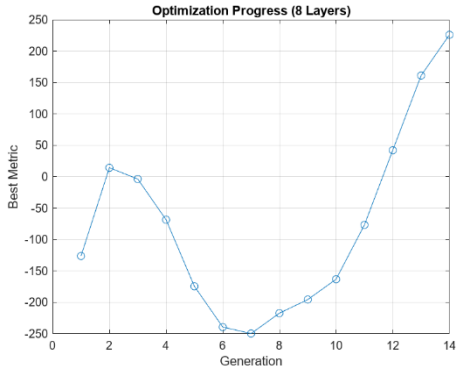
Figure 3.4 – Final temperature distribution plot for optimized configurations (6-11 Layers, ELC)

(a) 6 Layers (t = 25 mm)

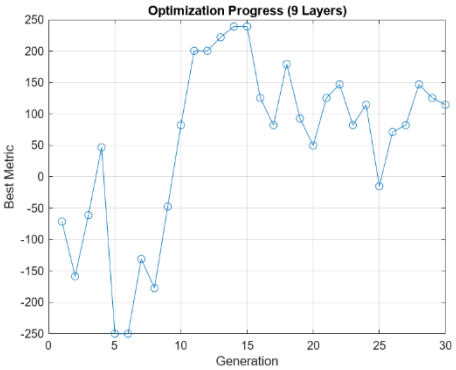


(b) 7 Layers (t = 30 mm)

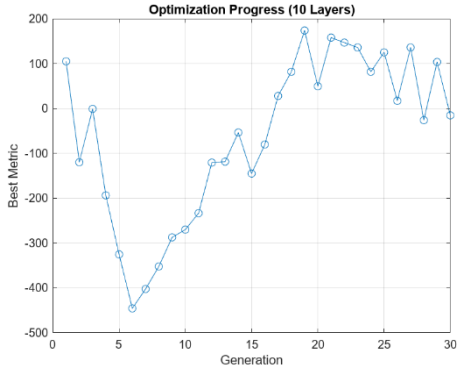
(c) 8 Layer (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

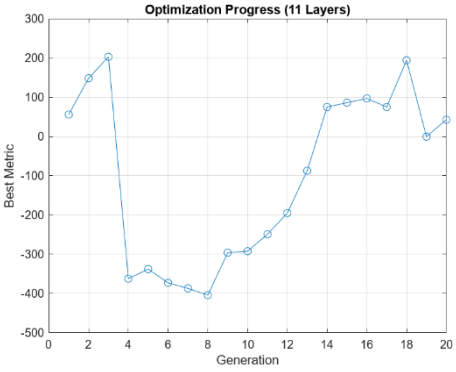


Figure 3.5 – Convergence of metrics over generations (6-11 Layers, ELC)

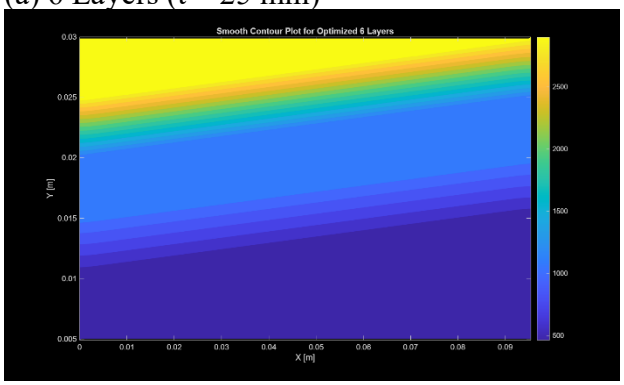
Table 3.4 – List of optimal configurations for each layer count (ELC)

	6 Layers	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Layer 1	C/SiC	C/SiC	Nextel	Nextel	Nextel	Nextel
Layer 2	Nextel	Nextel	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC
Layer 3	SiC/SiC	SiC/SiC	Nextel	Nextel	Nextel	Nextel
Layer 4	Nextel	Nextel	SiC/SiC	SiC/SiC	SiC/SiC	Nextel
Layer 5	Nextel	C/SiC	Nextel	Nextel	Nextel	C/SiC
Layer 6	SiC/SiC	Nextel	SiC/SiC	Nextel	SiC/SiC	Nextel
Layer 7	-	SiC/SiC	Nextel	SiC/SiC	Nextel	SiC/SiC
Layer 8	-	-	SiC/SiC	Nextel	SiC/SiC	Nextel
Layer 9	-	-	-	SiC/SiC	Nextel	SiC/SiC
Layer 10	-	-	-	-	SiC/SiC	Nextel
Layer 11	-	-	-	-	-	SiC/SiC

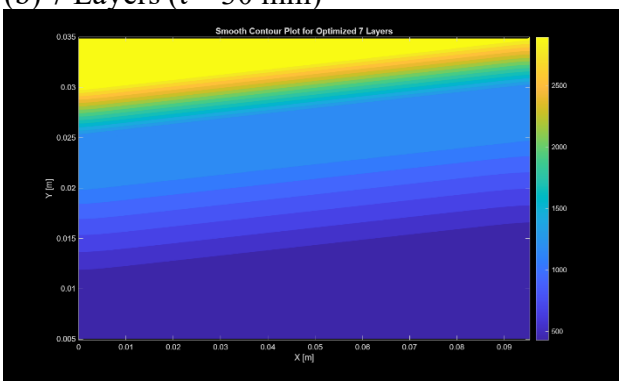
Table 3.5 – Best temperature metric for each layer count (ELC)

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Best Temp. Metric	508.4609K	378.1151K	339.9675K	318.3218K	303.995K	302.1878K

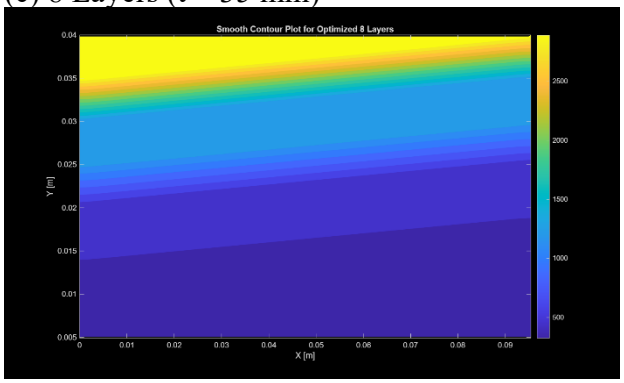
(a) 6 Layers (t = 25 mm)



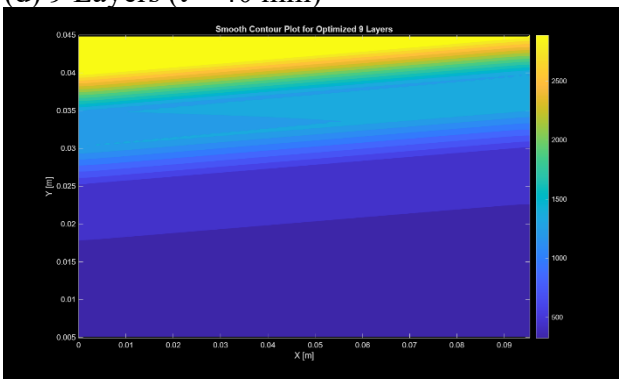
(b) 7 Layers (t = 30 mm)



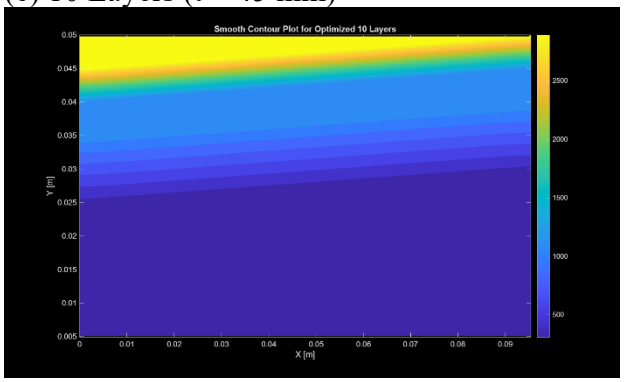
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

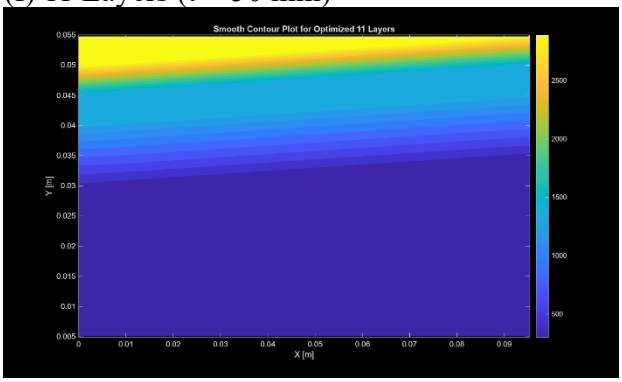
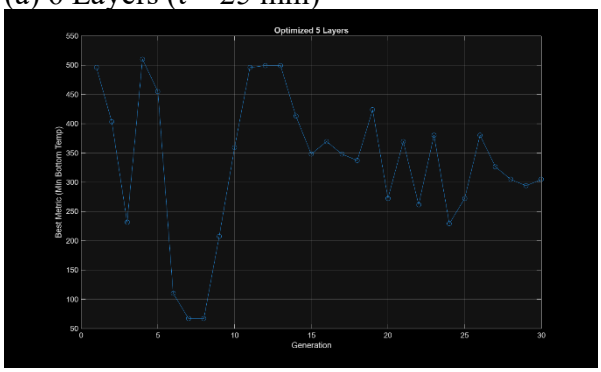
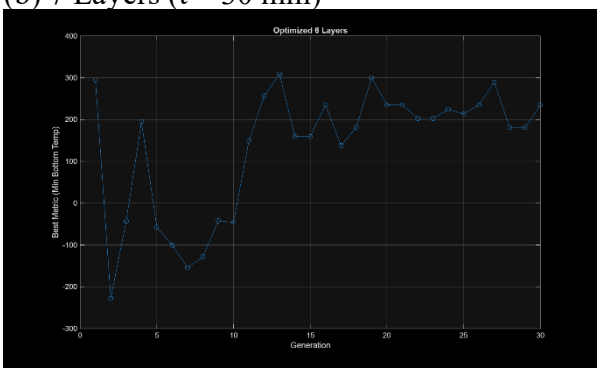


Figure 3.6 – Final temperature distribution plot for optimized configurations (6-11 Layers, 3°)

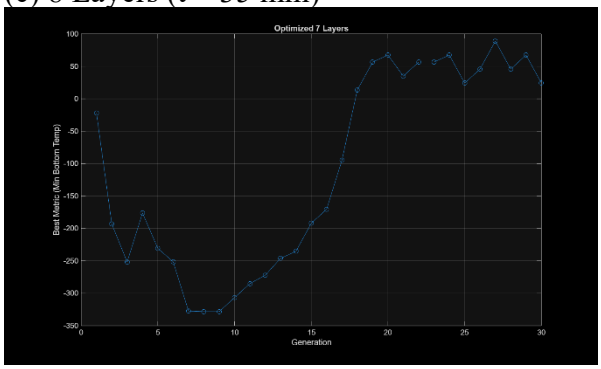
(a) 6 Layers (t = 25 mm)



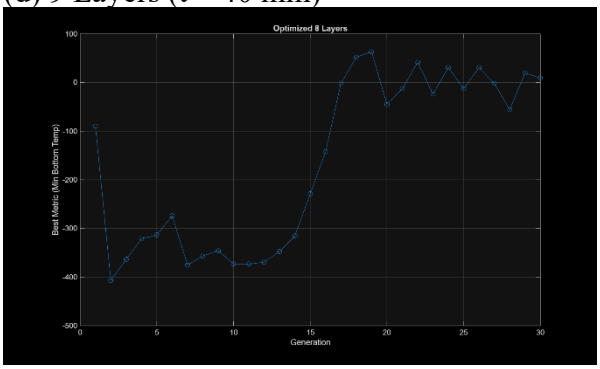
(b) 7 Layers (t = 30 mm)



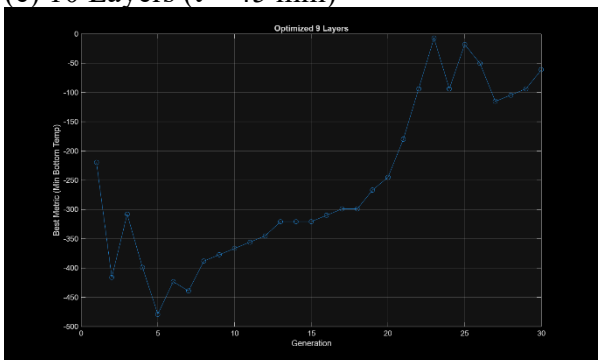
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

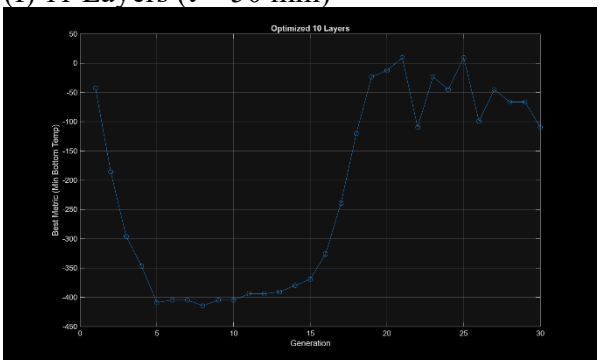


Figure 3.7 – Convergence of metrics over generations (6-11 Layers, 3°)

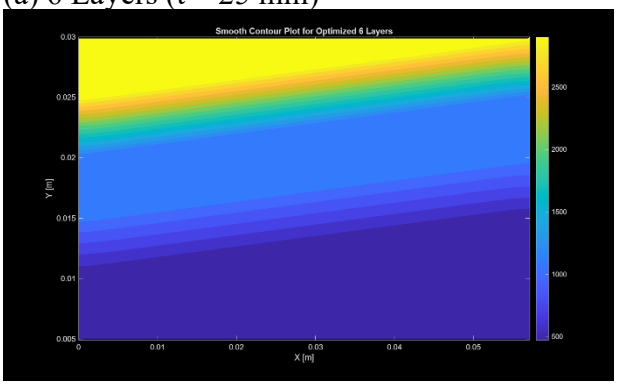
Table 3.6 – List of optimal configurations for each layer count (3°)

	5 Layers	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**
Layer 1	Nextel	Nextel	Nextel	Nextel	Nextel	Nextel
Layer 2	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC
Layer 3	Nextel	Nextel	Nextel	Nextel	Nextel	Nextel
Layer 4	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC	C/SiC	C/SiC
Layer 5	Nextel	Nextel	Nextel	Nextel	Nextel	Nextel
Layer 6	SiC/SiC	Nextel	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC
Layer 7	-	SiC/SiC	Nextel	Nextel	Nextel	Nextel
Layer 8	-	-	SiC/SiC	Nextel	SiC/SiC	SiC/SiC
Layer 9	-	-	-	SiC/SiC	Nextel	Nextel
Layer 10	-	-	-	-	SiC/SiC	Nextel
Layer 11	-	-	-	-	-	SiC/SiC

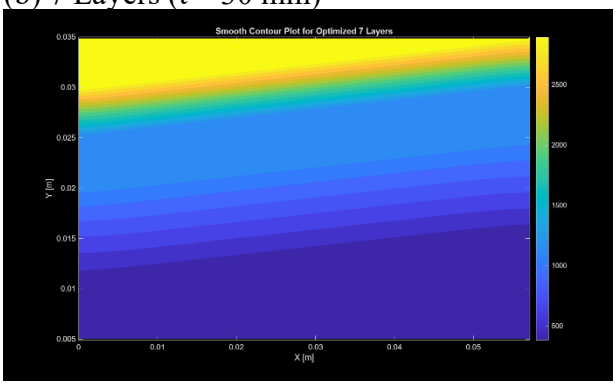
Table 3.7 – Best temperature metric for each layer count (3°)

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Best Temp. Metric	472.9306K	385.1348K	321.8892K	310.0455K	302.6516K	301.1661K

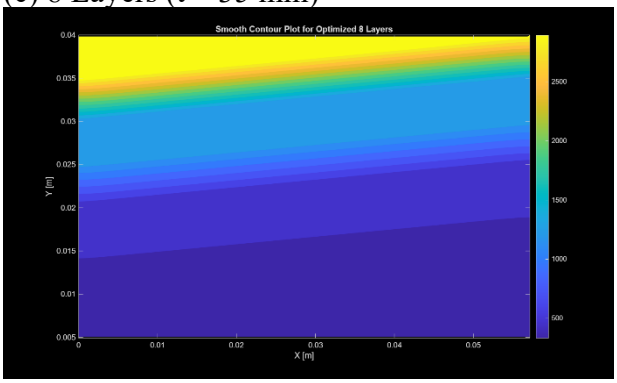
(a) 6 Layers (t = 25 mm)



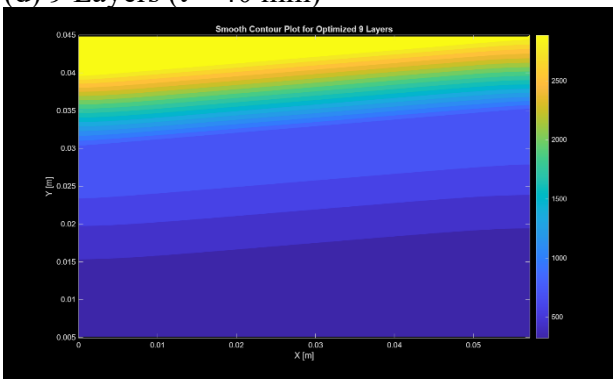
(b) 7 Layers (t = 30 mm)



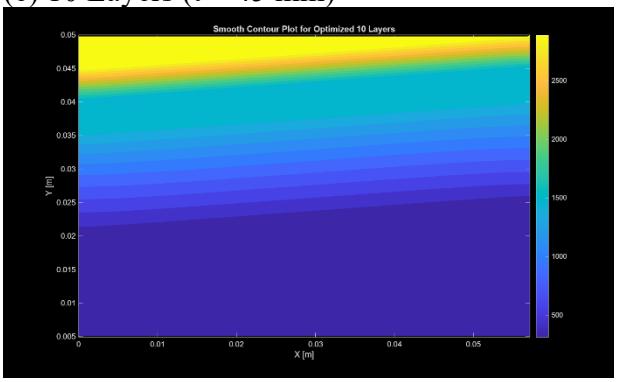
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

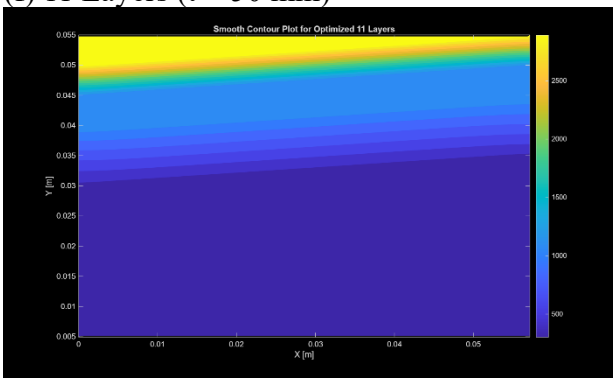
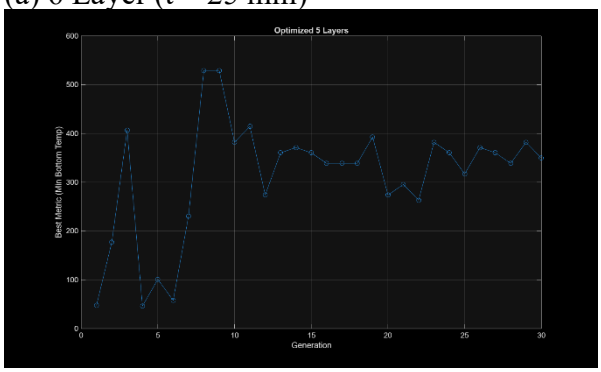
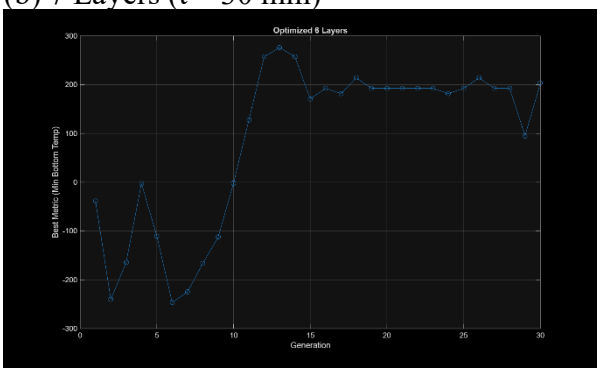


Figure 3.8 – Final temperature distribution plot for optimized configurations (6-11 Layers, 5°)

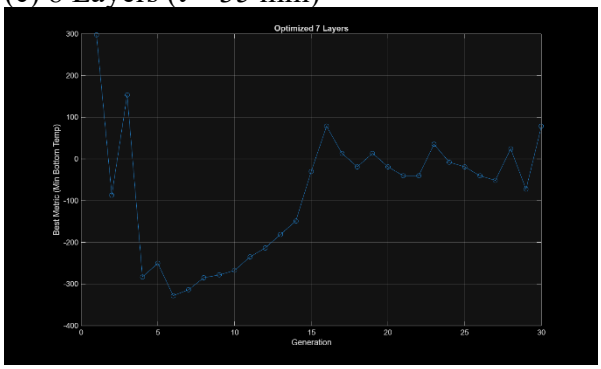
(a) 6 Layer (t = 25 mm)



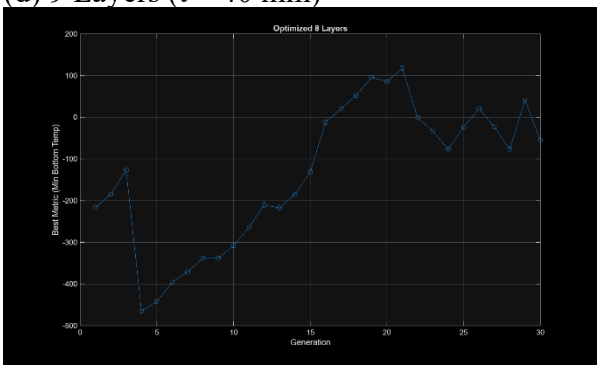
(b) 7 Layers (t = 30 mm)



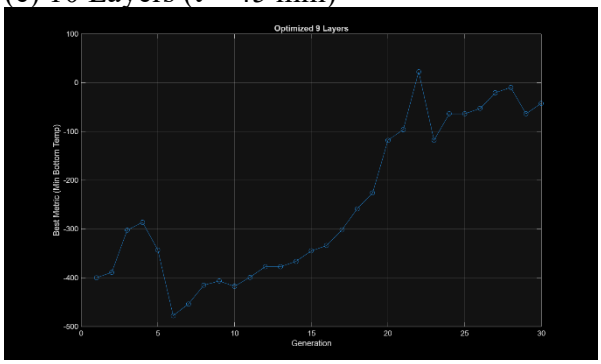
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

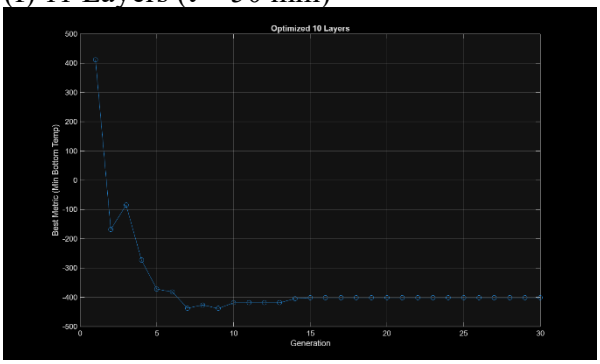


Figure 3.9 – Convergence of metrics over generations (6-11 Layers, 5°)

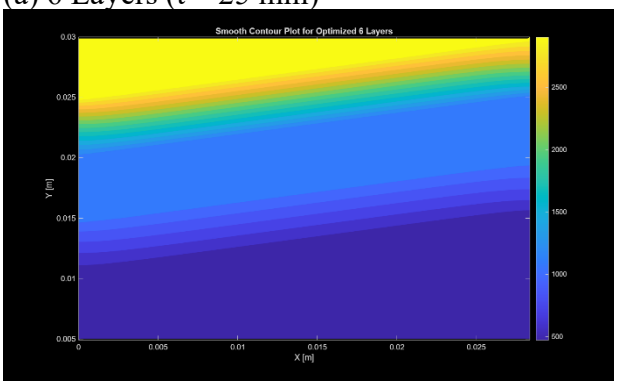
Table 3.8 – List of optimal configurations for each layer count (5°)

	5 Layers	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**
Layer 1	Nextel	Nextel	Nextel	Nextel	Nextel	Nextel
Layer 2	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC
Layer 3	Nextel	Nextel	Nextel	Nextel	Nextel	Nextel
Layer 4	SiC/SiC	SiC/SiC	SiC/SiC	Nextel	SiC/SiC	C/SiC
Layer 5	Nextel	Nextel	Nextel	SiC/SiC	Nextel	Nextel
Layer 6	SiC/SiC	Nextel	SiC/SiC	Nextel	C/C	SiC/SiC
Layer 7	-	SiC/SiC	Nextel	SiC/SiC	Nextel	Nextel
Layer 8	-	-	SiC/SiC	Nextel	C/SiC	Nextel
Layer 9	-	-	-	SiC/SiC	Nextel	SiC/SiC
Layer 10	-	-	-	-	SiC/SiC	Nextel
Layer 11	-	-	-	-	-	SiC/SiC

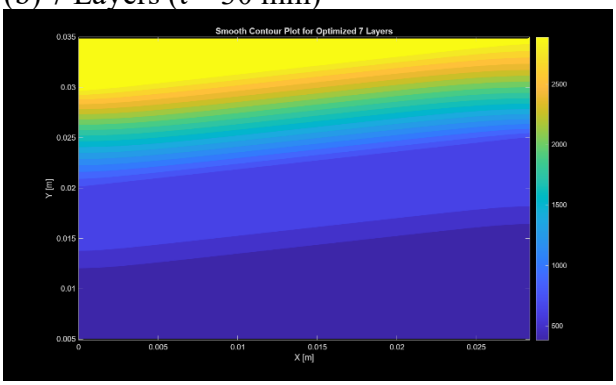
Table 3.9 – Best temperature metric for each layer count (5°)

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Best Temp. Metric	474.4259K	384.8418K	322.1719K	310.5389K	303.7311K	301.2182K

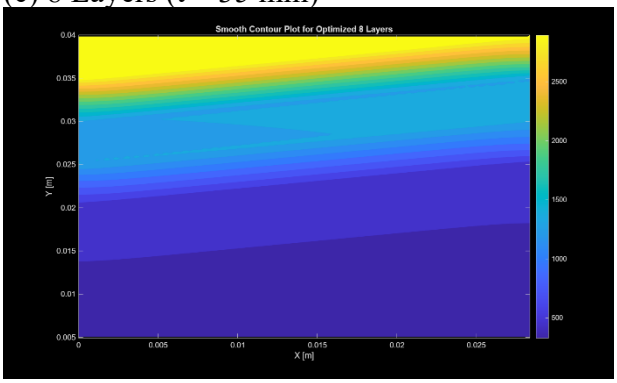
(a) 6 Layers (t = 25 mm)



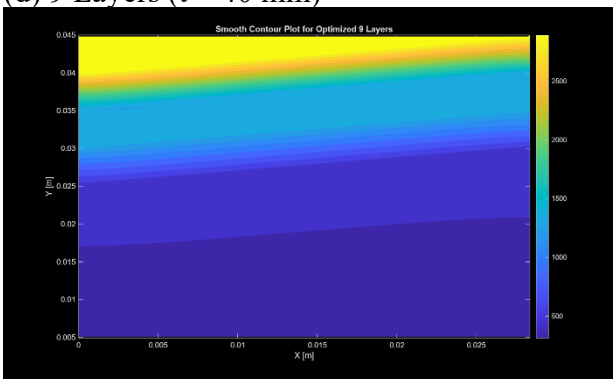
(b) 7 Layers (t = 30 mm)



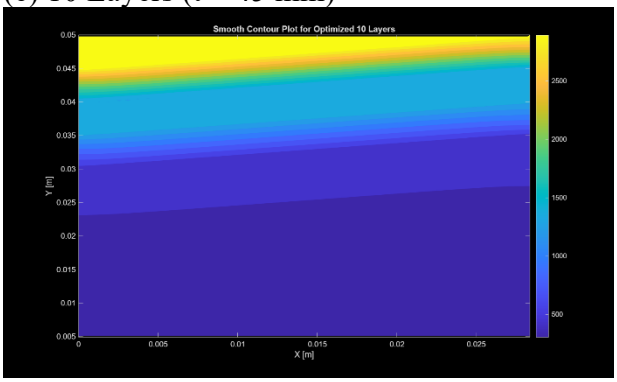
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

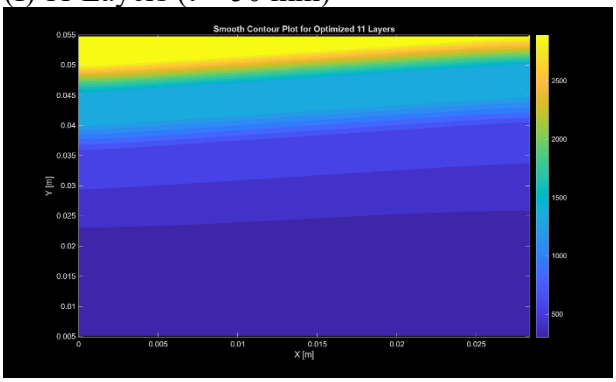
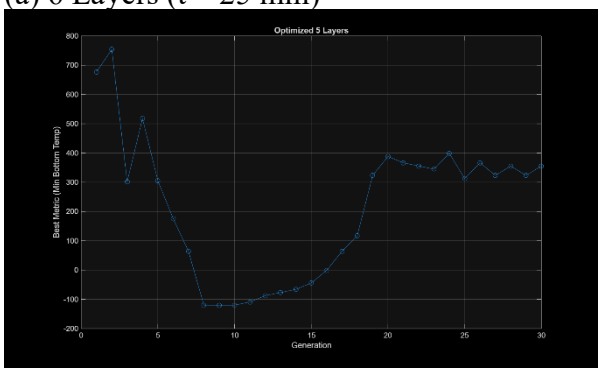
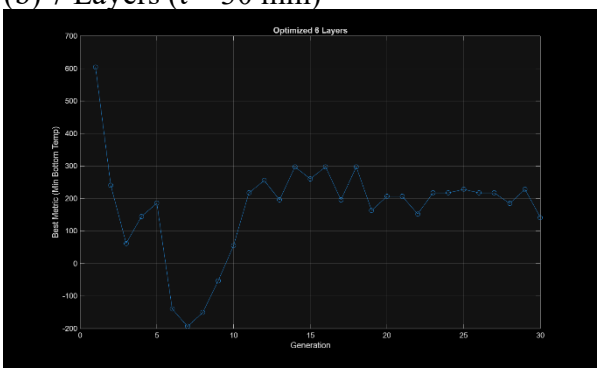


Figure 3.10 – Final temperature distribution plot for optimized configurations (6-11 Layers, 10°)

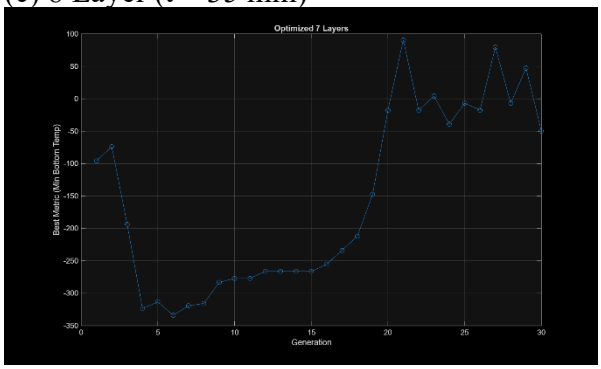
(a) 6 Layers (t = 25 mm)



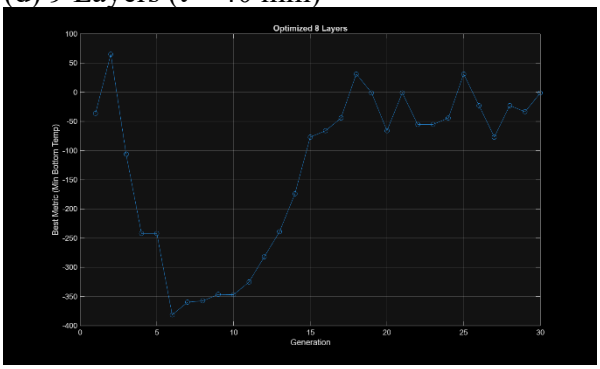
(b) 7 Layers (t = 30 mm)



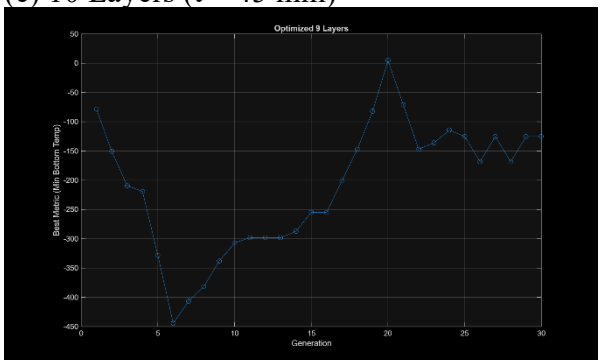
(c) 8 Layer (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

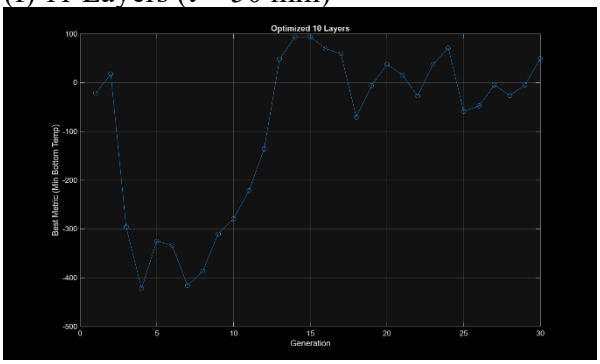


Figure 3.11 – Convergence of metrics over generations (6-11 Layers, 10°)

Table 3.10 – List of optimal configurations for each layer count (10°)

	5 Layers	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**
Layer 1	Nextel	Nextel	Nextel	Nextel	Nextel	Nextel
Layer 2	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC
Layer 3	Nextel	Nextel	Nextel	Nextel	Nextel	Nextel
Layer 4	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC	C/C
Layer 5	Nextel	Nextel	Nextel	Nextel	Nextel	Nextel
Layer 6	SiC/SiC	Nextel	SiC/SiC	SiC/SiC	SiC/SiC	SiC/SiC
Layer 7	-	SiC/SiC	Nextel	Nextel	Nextel	Nextel
Layer 8	-	-	SiC/SiC	Nextel	SiC/SiC	Nextel
Layer 9	-	-	-	SiC/SiC	Nextel	SiC/SiC
Layer 10	-	-	-	-	SiC/SiC	Nextel
Layer 11	-	-	-	-	-	SiC/SiC

Table 3.11 – Best temperature metric for each layer count (10°)

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Best Temp. Metric	480.4215K	387.9507K	323.4294K	310.7742K	302.327K	301.442K

Across both flat and angled optimization runs, the algorithm consistently favored placing composite materials before insulators materials in a sandwich pattern arrangement. This material arrangement aligns with the established aerospace TPS design practices, reinforcing physical validity of the optimization results. Across all configurations, the GA consistently favored a composite-insulator stacking pattern. This aligns with known aerospace TPS strategies and validates the model's physical realism. Furthermore, angled configurations consistently outperformed their flat counterparts in terms of bottom surface temperature, supporting the effectiveness of thermal refraction through internal geometry manipulation.

4. CFD Validation and Results

4.1 Introduction

Chapter 3 details the MATLAB-based optimization framework and the resulting MTPS configurations. While the MATLAB optimization provided a strong foundation for MTPS design, additional validation steps are necessary to confirm the physical accuracy of the results. The complex and nonlinear nature of hypersonic reentry and thermal transfer properties mean that validation is a key consideration when attempting to optimize any configuration. ANSYS Fluent was chosen as the primary tool for the validation of the MATLAB results for its advanced CFD capabilities for thermal analysis, hypersonic flow, and a large variety of built-in tools used for analysis and processing.

This chapter provides the CFD setup, execution, and results of the simulations used to evaluate the optimized TPS configurations obtained in the MATLAB optimization process. Simulating the heat transfer in a 2D environment similar to the MATLAB environment will allow for controllability and precision when it comes to validating and processing the results from both the MATLAB optimization and CFD results. The CFD simulation will illustrate the heat flux experienced by the TPS, the temperature distribution across the layers, and the overall effectiveness of each TPS configuration.

These CFD simulations provide a critical step in verifying the results of the optimization process. This step ensures the proposed TPS configurations meet the thermal performance requirements for hypersonic atmospheric reentry. All results, findings, and discrepancies between the simulations will be discussed and analyzed for refinement feedback.

4.2 CFD Setup

The MATLAB optimization used a PDE modeler to introduce a 2D geometry as the TPS design. To simulate this in the CFD setup the geometry can be created in ANSYS Discovery with an analysis type of 2D. This allows the different simulations to function in the same dimension, behaving in a similar manner to each other. 6 Different geometries were created for the 6 different optimal configurations produced from the MATLAB optimization for each layer count. Therefore, the same dimensions were used to create 5 mm equally spaced layers for layer count 5 through 10.

The geometries were then refined through a mesh equal to that of the MATLAB simulation. The mesh for the PDE Modeler was defined as approximately 1.25 mm. For clearer results, the mesh in the CFD simulations were refined further to a resolution of 0.5 mm. Additionally, there is a discrepancy in the meshing software of ANSYS Fluent and MATLAB's PDE Modeler. The PDE Modeler in MATLAB only produces triangular mesh, while with the meshing software in Fluent you are free to structure the mesh with quadrilateral meshing. Since the geometry of the TPS is designed to be rectangular, a structured quadrilateral mesh aligned with the boundaries offers more accurate results during post-processing, especially for heat conduction across the rectangular

geometries. Although more refined than the MATLAB mesh, the CFD simulations all use a consistent element size of 0.5 mm to ensure comparability across geometries.

The physics setup involves activating the energy equation in Fluent to enable conductive properties in solid materials. Since this is a thermal analysis problem, viscosity and other physics settings go unchanged as the conductivity of the materials is the primary concern. Similarly, the setup uses a density-based solver with transient time steps to simulate the flow of heat in a similar way to how the PDE Modeler did. The transient time steps allow the simulation to define a time step, final time, and iterations per time step. This is ideal for the setup of this simulation since the time step from the MATLAB optimization can be replicated in the CFD setup. Additionally, the density-based solver is selected over the alternative pressure-based solver since no pressure is applied to the inside of the TPS layers and density is a driving factor in the efficiency of TPS.

The material list from the MATLAB optimization is replicated in the material definition of the CFD setup. The material name, thermal conductivity, density, and specific heat capacity are all necessary to simulate the heat transfer between the TPS layers. Once materials are defined, boundary conditions are also defined. To ensure that the simulations are simulating the same type of heat transfer analysis, there is a fixed temperature to the top surface of the TPS of 3000K as well as a fixed temperature to the bottom surface of the TPS at 300K. Once the materials and boundary conditions are defined, the solution is then initialized.

Post initialization, the calculation and time steps should be defined to match the simulation in the MATLAB optimization. Since there is a time step of 1 and final time of 100 in the MATLAB script, the time step and final time will be mirrored in the CFD setup to maintain consistency. The maximum iterations per time step was defined to be 30 to ensure that the iterations converged before moving to the next time step. After setting the calculation activities up according to the MATLAB simulations, graphics and plots may be defined for results and analysis.

An X-Y plot for position and temperature are defined and set up to display at the end of the simulation to produce a plot similar to that produced in the MATLAB simulations. Similarly, a contour of static temperature is created to obtain a temperature distribution graph of the simulation for each layer count. With these results, it is possible to visualize the data and compare it directly to the results produced by the MATLAB simulations.

The below figure shows a visualization of the mesh in ANSYS Fluent. This figure demonstrates the difference between the mesh produced in the previous chapter (as shown in Figure 3.1). The structured meshing in comparison to the triangular meshing from the previous chapter will produce more accurate results although it uses the same face meshing size. Similar to the previous chapter, the increase in layer count will increase the number of cells of the total simulation but remain constant for each layer which ensures equal analysis over the entire TPS configuration. Figure 4.1 shows the structured mesh generated in ANSYS Fluent. Compared to the triangular mesh used in the PDE solve of MATLAB, this setup provides improved alignment along layer boundaries with more resolved thermal gradients.

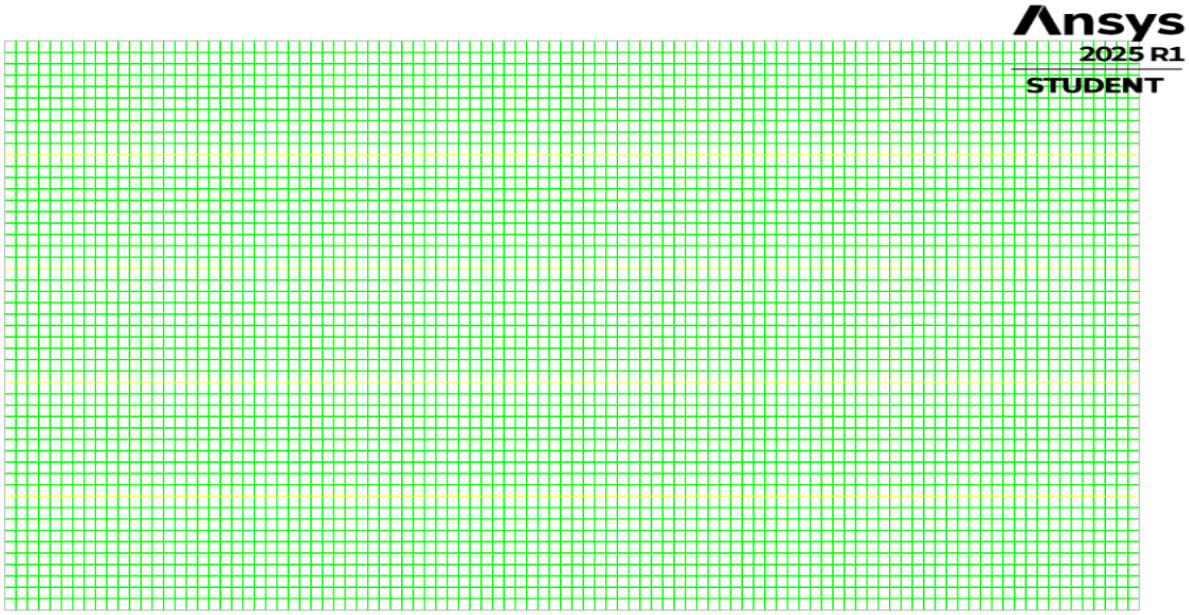
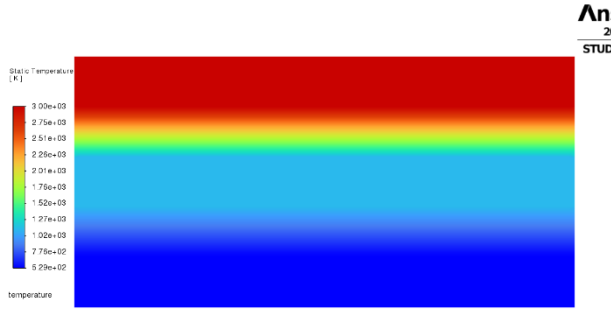


Figure 4.1 – Mesh visualization in ANSYS Fluent (0.5 mm cell size)

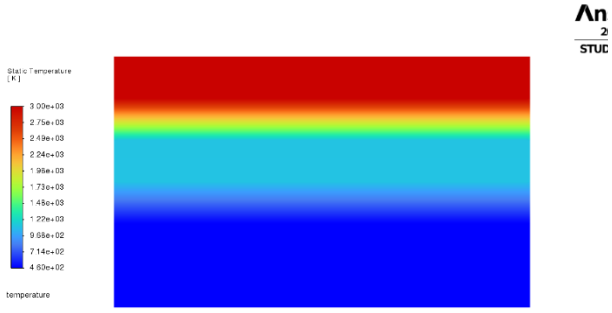
4.3 Results and Discussion

Figure 4.2 presents the final temperature distributions generated by the CFD simulations for each optimized flat MTPS configuration. These are directly compared to the MATLAB results shown in Figure 3.2 to verify performance consistency. Each configuration uses the same material arrangement as its MATLAB counterpart, allowing direct comparison between the two environments. These results confirm the expected trend of higher layer counts resulting in increased thermal resistance and lower bottom surface temperatures.

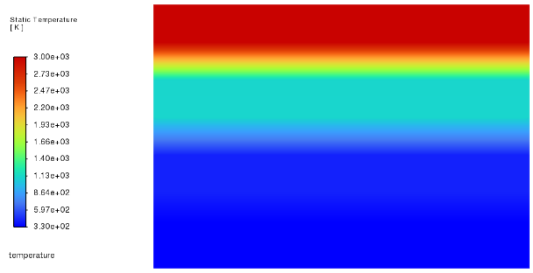
(a) 5 Layers (t = 25 mm)



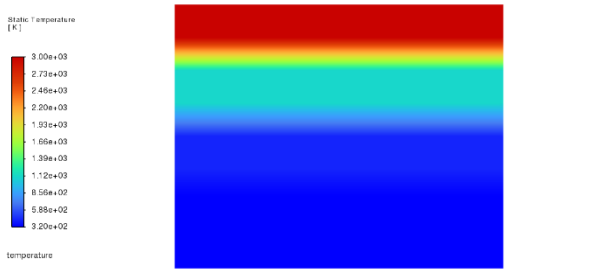
(b) 6 Layers (t = 30 mm)



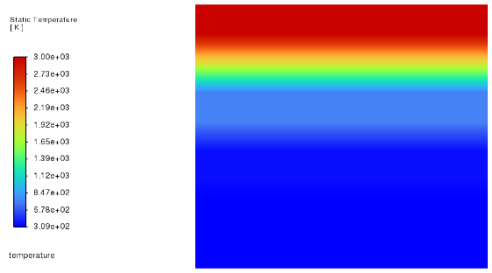
(c) 7 Layers (t = 35 mm)



(d) 8 Layers (t = 40 mm)



(e) 9 Layers (t = 45 mm)



(f) 10 Layers (t = 50 mm)

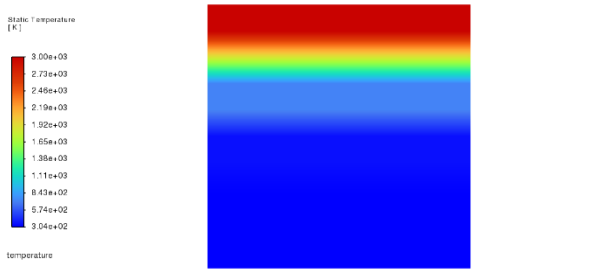
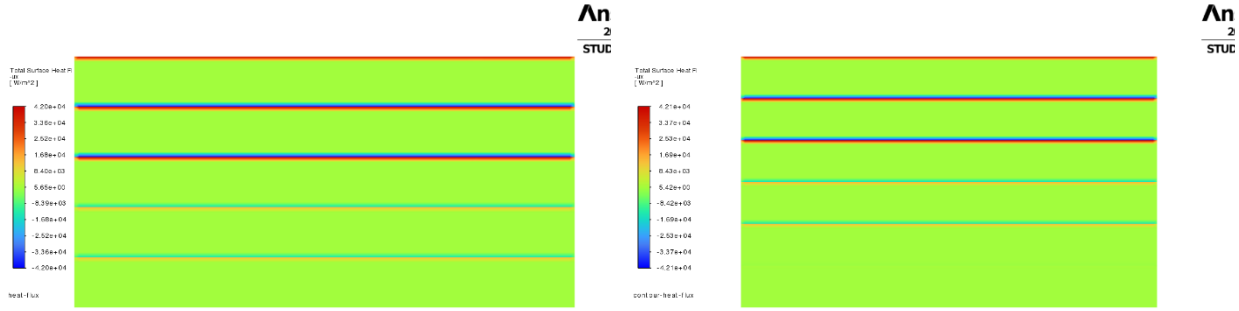


Figure 4.2 – Temperature distribution plot for optimal configurations (5-10 Layers, 0°)

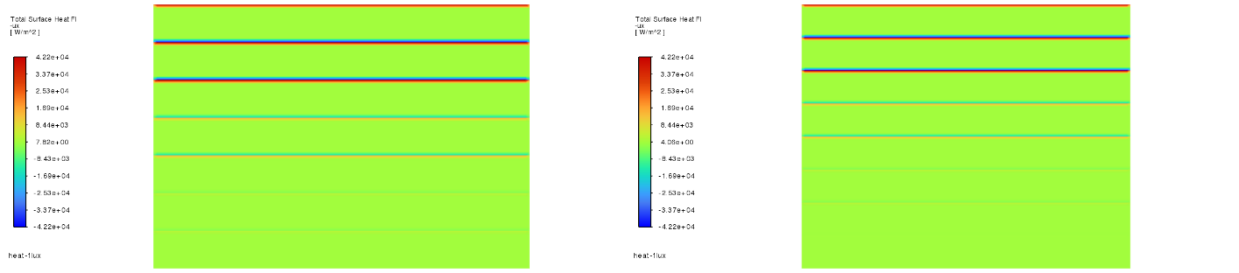
(a) 5 Layers (t = 25 mm)

(b) 6 Layers (t = 30 mm)



(c) 7 Layers (t = 35 mm)

(d) 8 Layers (t = 40 mm)



(e) 9 Layers (t = 45mm)

(f) 10 Layers (t = 50mm)

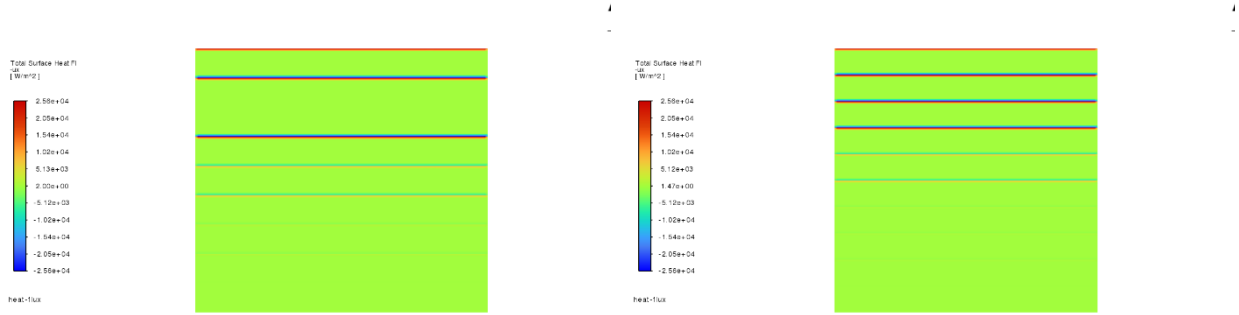
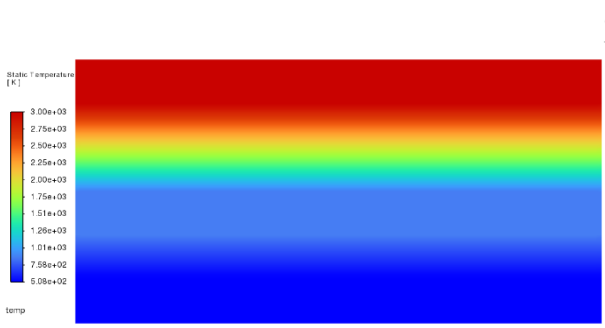


Figure 4.3 – Heat flux plot for optimal configurations (5-10 Layers, 0°)

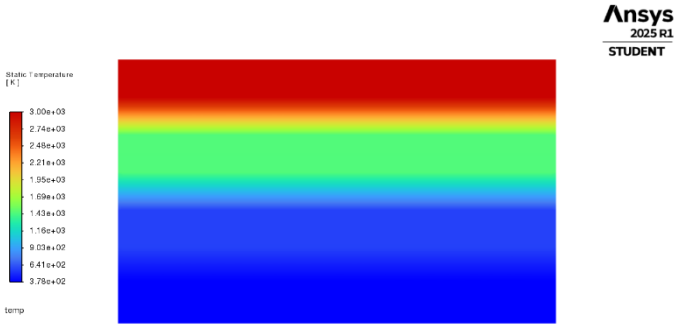
Table 4.1 – Best temperature metric for each layer count (0°)

Layer Count	5 Layers	6 Layers	7 Layers	8 Layers	9 Layers	10 Layers
Best Temp. Metric	529.32 K	459.78 K	329.61 K	319.53 K	308.89 K	304.13 K

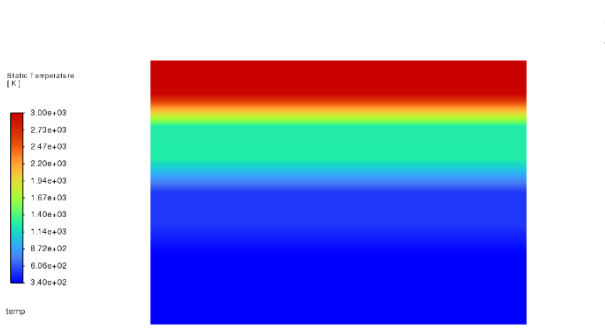
(a) 6 Layers (t = 25 mm)



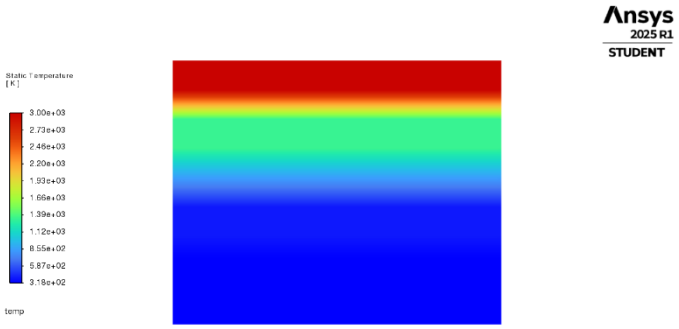
(b) 7 Layers (t = 30 mm)



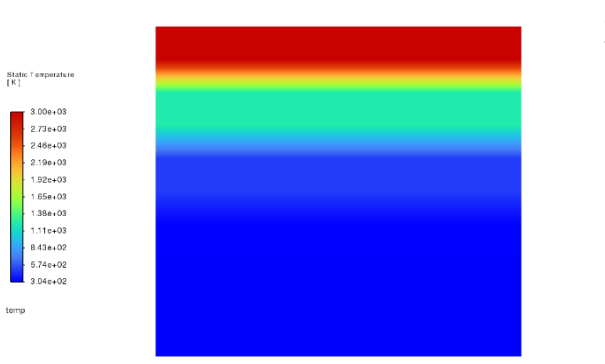
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

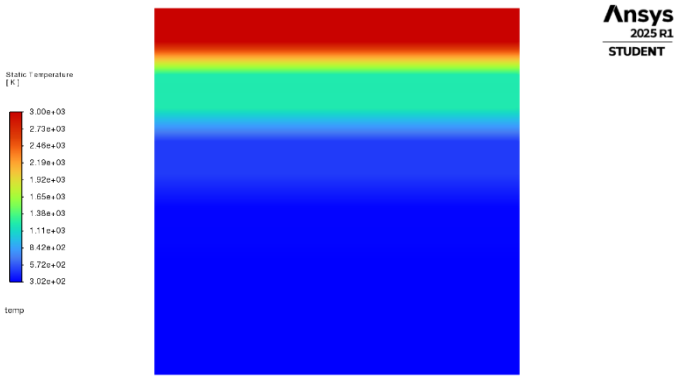
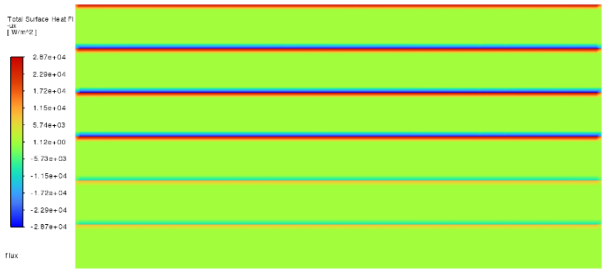
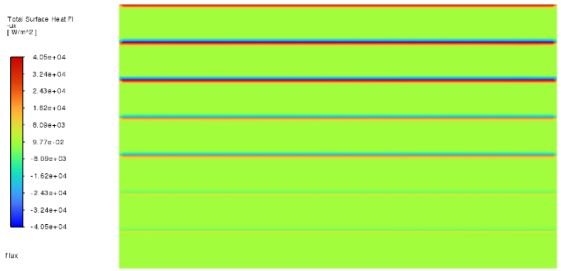


Figure 4.4 – Temperature distribution plot for optimal configurations (6-11 Layers, ELC)

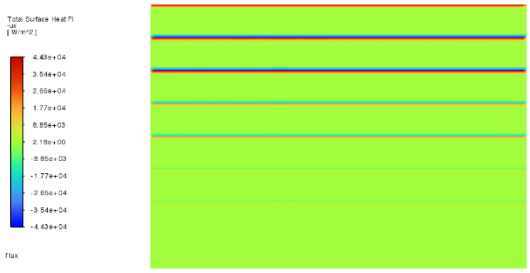
(a) 6 Layers (t = 25 mm)



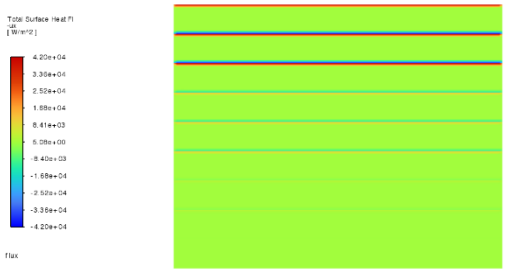
(b) 7 Layers (t = 30 mm)



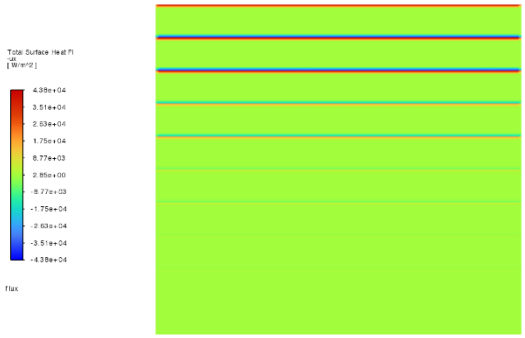
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

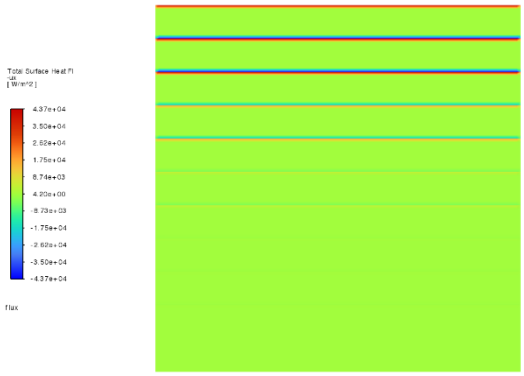
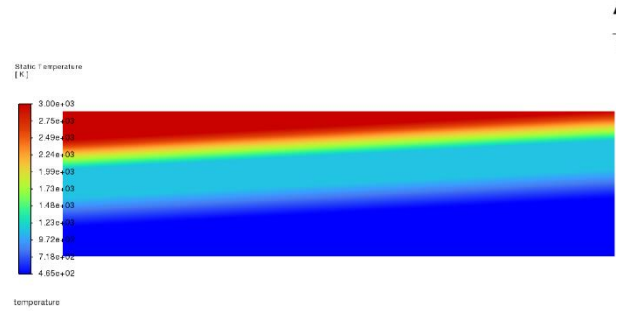


Figure 4.5 – Heat flux plot for optimal configurations (6-11 Layers, ELC)

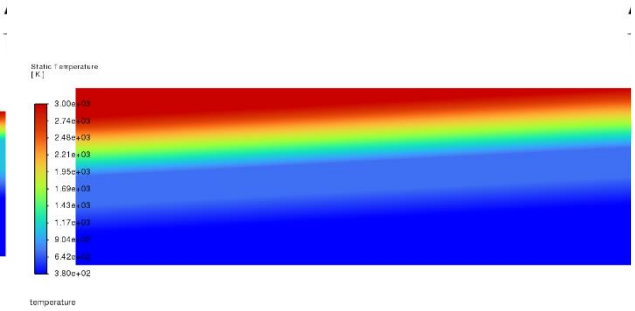
Table 4.2 – Best temperature metric for each layer count (ELC)

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Best Temp. Metric	508.42 K	378.4 K	340.18 K	318.43 K	304.01 K	302.19 K

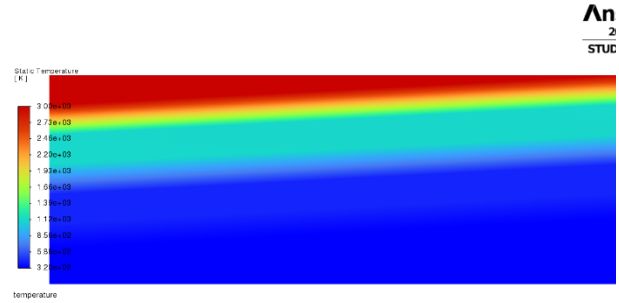
(a) 6 Layers (t = 25 mm)



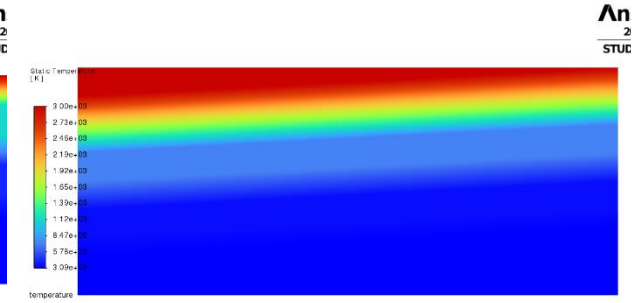
(b) 7 Layers (t = 30 mm)



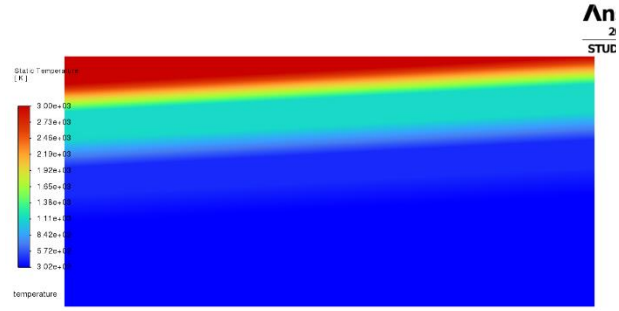
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

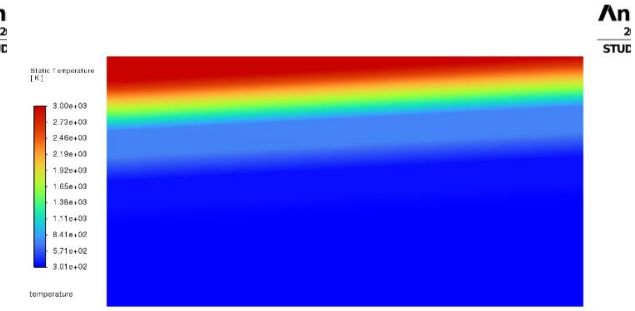
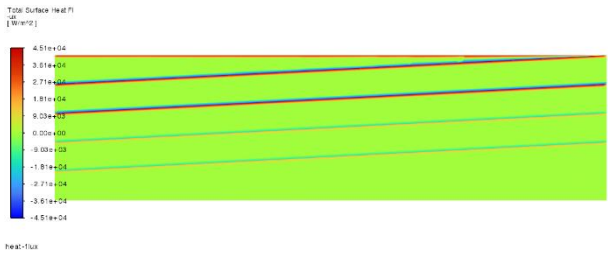
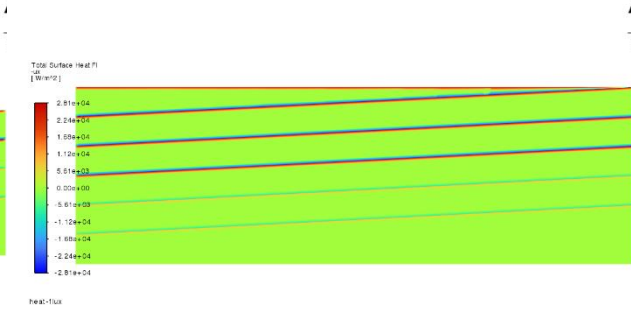


Figure 4.6 – Temperature distribution plot for optimal configurations (6-11 Layers, 3°)

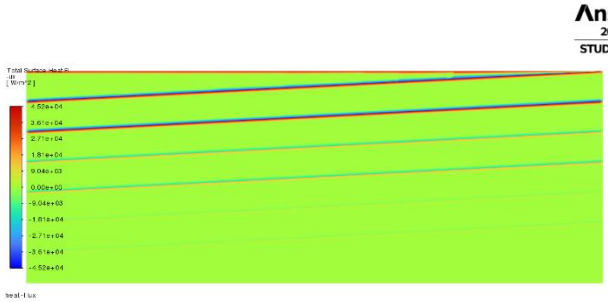
(a) 6 Layers (t = 25 mm)



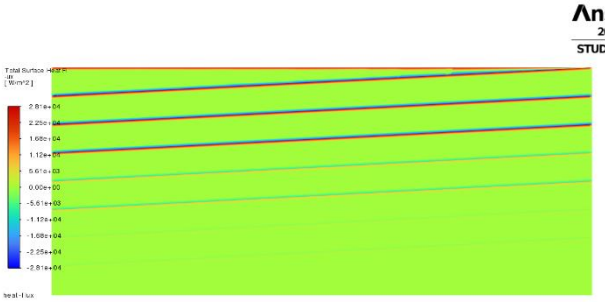
(b) 7 Layers (t = 30 mm)



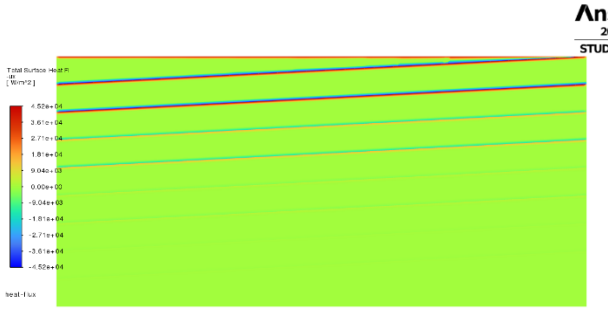
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

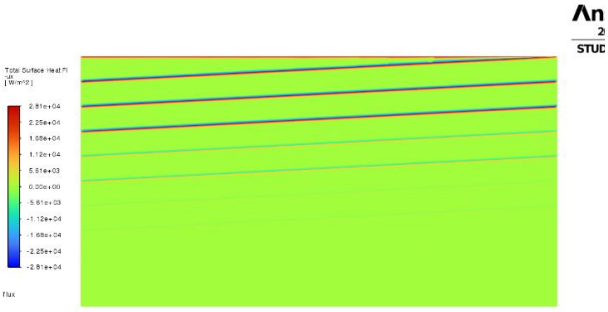
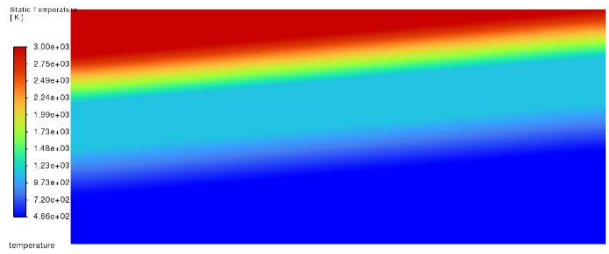


Figure 4.7 – Heat flux plot for optimal configurations (6-11 Layers, 3°)

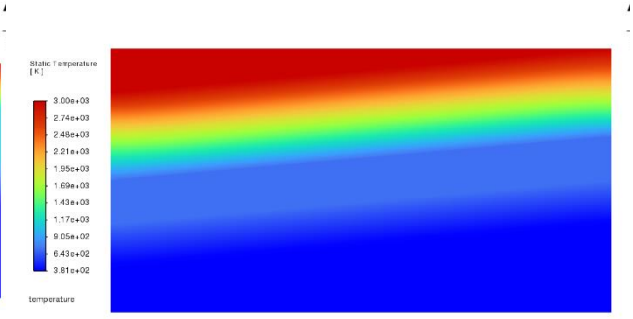
Table 4.3 – Best Temperature Metric for Each Layer Count (3°)

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Best Temp. Metric	472.75 K	384.07 K	321.9 K	310.03 K	302.66 K	301.15 K

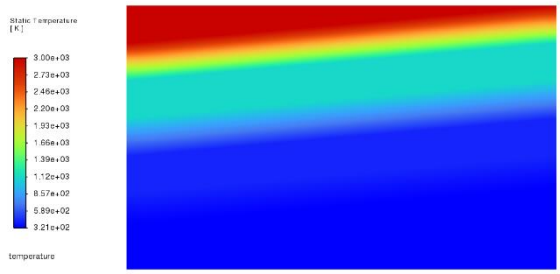
(a) 6 Layers (t = 25 mm)



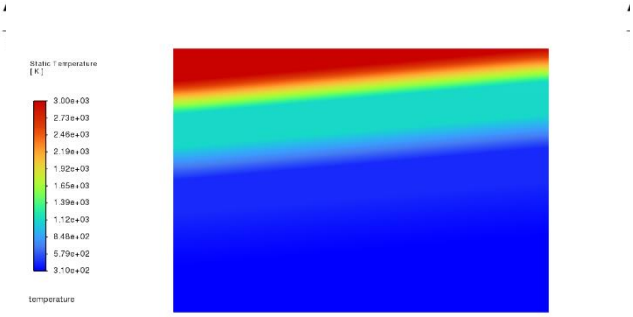
(b) 7 Layers (t = 30 mm)



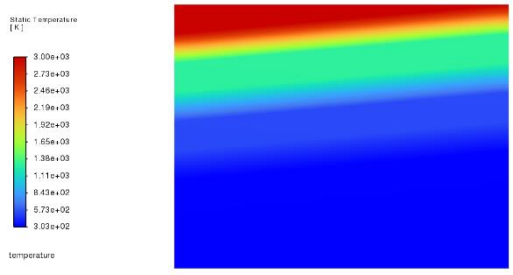
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

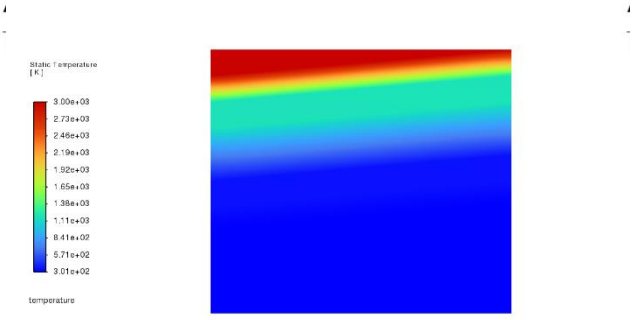
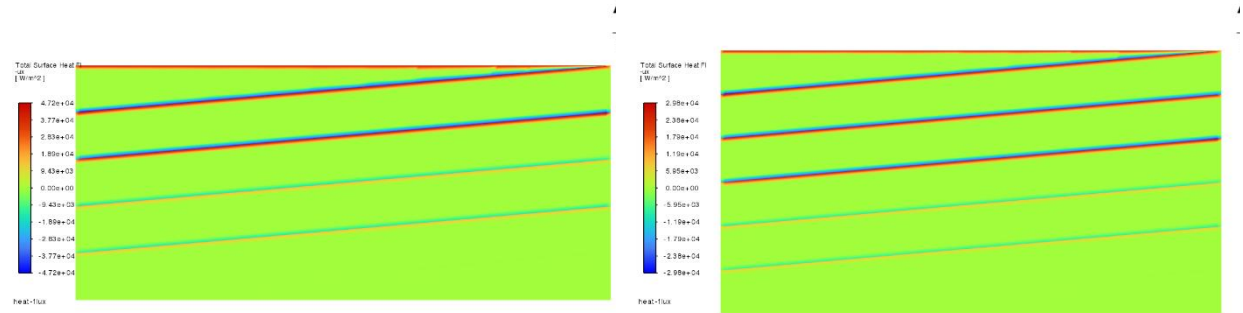


Figure 4.8 – Temperature distribution plot for optimal configurations (6-11 Layers, 5°)

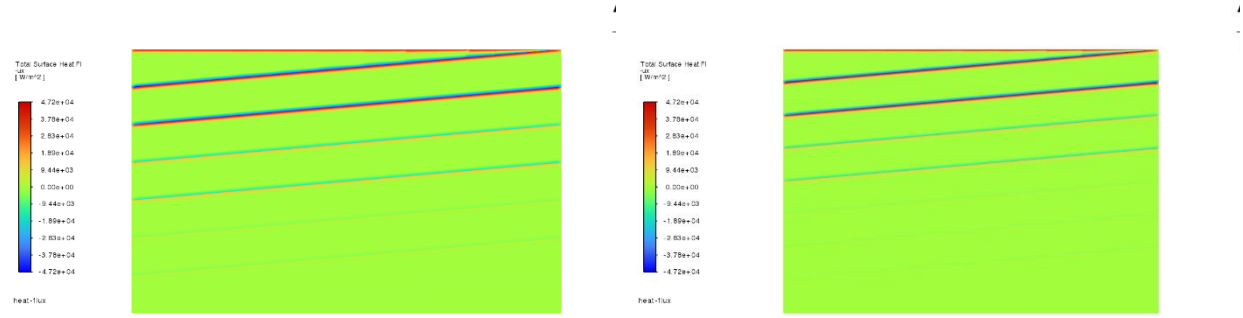
(a) 6 Layers (t = 25 mm)

(b) 7 Layers (t = 30 mm)



(c) 8 Layers (t = 35 mm)

(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)

(f) 11 Layers (t = 50 mm)

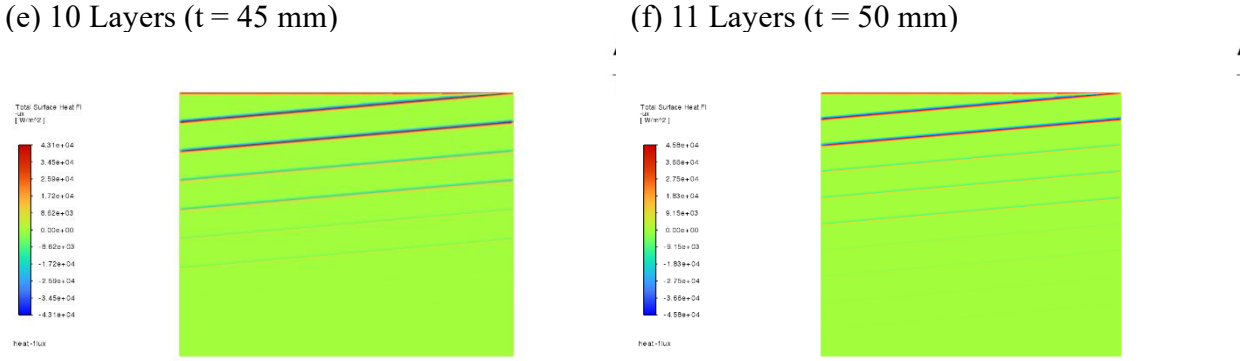
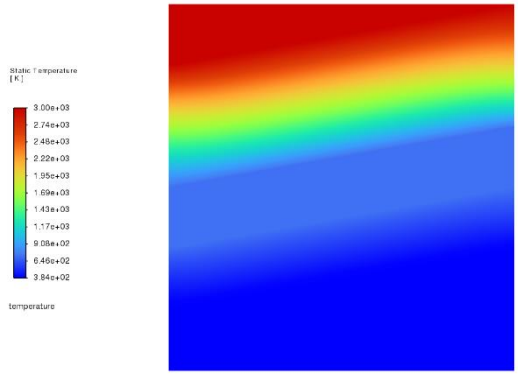


Figure 4.9 – Heat flux plot or optimal configurations (6-11 Layers, 5°)

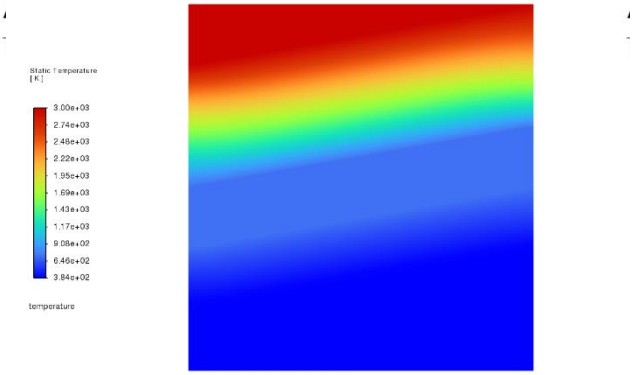
Table 4.4 – Best temperature metric for each layer count (5°)

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Best Temp. Metric	474.01 K	386.49 K	322.11 K	310.5 K	303.71 K	301.2 K

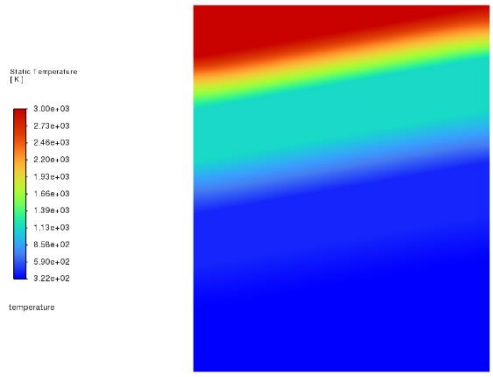
(a) 6 Layers (t = 25 mm)



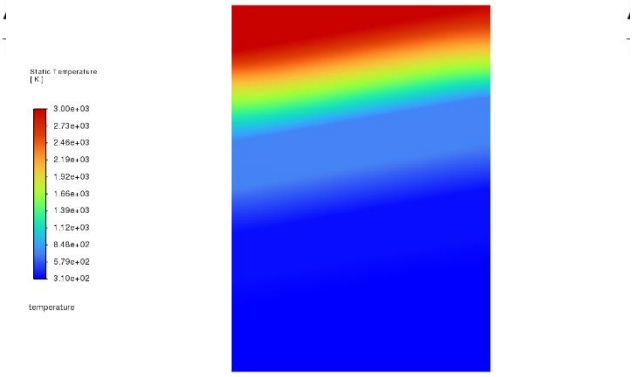
(b) 7 Layers (t = 30 mm)



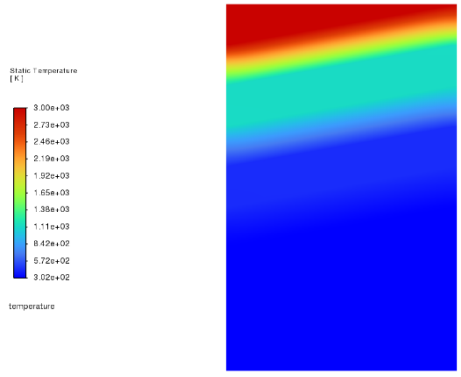
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

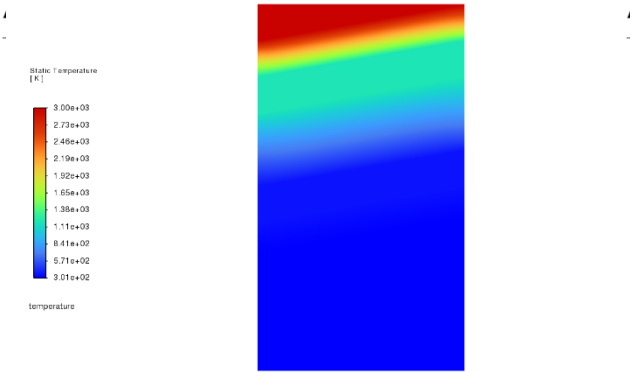
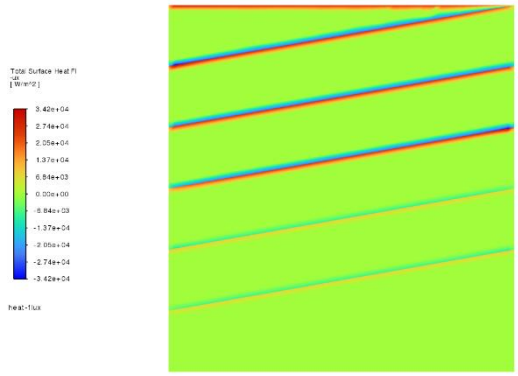
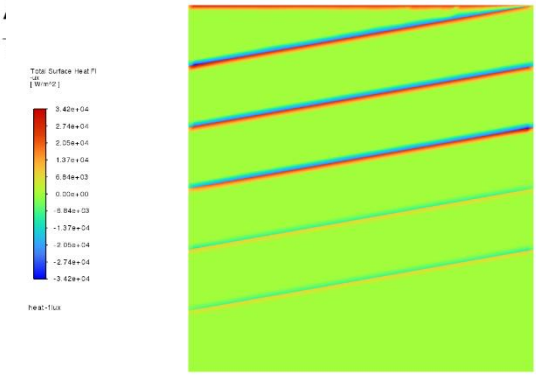


Figure 4.10 – Temperature distribution plot for optimal configurations (6-11 Layers, 10°)

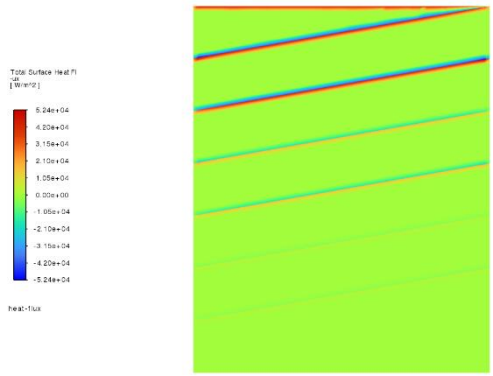
(a) 5 Layers (t = 25 mm)



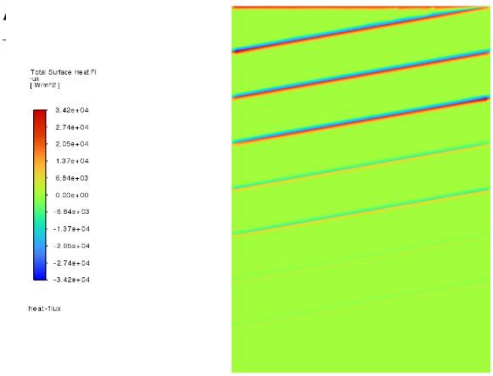
(b) 6 Layers (t = 30 mm)



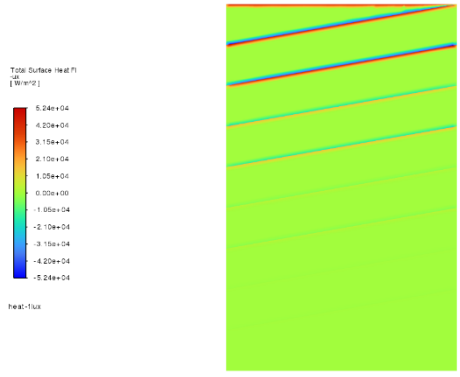
(c) 7 Layers (t = 35 mm)



(d) 8 Layers (t = 40 mm)



(e) 9 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

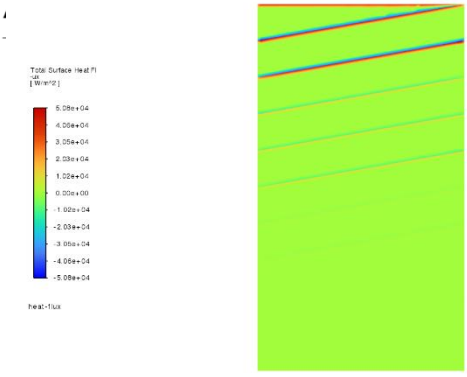


Figure 4.11 – Heat flux plot for optimal configurations (6-11 Layers, 10°)

Table 4.5 – Best temperature metric for each layer count (10°)

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Best Temp. Metric	480.51 K	387.9 K	323.42 K	310.76 K	302.31 K	301.43 K

The Fluent simulations confirm the reliability of the MATLAB-optimized configurations by producing consistent thermal performance trends. To deepen understanding of the physical mechanisms behind angled TPS effectiveness, the next chapter introduces a manual verification approach to analyze heat conduction behavior across flat and angled domains.

5. Manual Verification Using 2D Finite Difference Method

5.1 Introduction

In addition to the GA optimization and ANSYS Fluent CFD validation steps presented in earlier chapters, a manual verification approach was developed using a finite difference model implemented in MATLAB. This method uses a fully manual 2D transient heat conduction solver, independent of built-in solvers or toolboxes, to further validate and understand the physical behavior of the angled MTPS configurations. By implementing this numerical model method from scratch, the script offers transparent insight into how geometric orientation affects heat conduction through passive MTPS.

The goal of the work discussed in this chapter is to document the utilized approach, explain the logic behind the simulations, and present data that supports the main objective of this thesis. The script developed in this chapter functions as a hand-calculation-style model that recreates identical simulation conditions to those used in the MATLAB GA optimization and the Fluent simulations. This methodological consistency helps establish confidence in the heat refraction principle and the thesis findings overall.

5.2 Numerical Approach and Implementation

This verification script solves this two-dimensional, transient heat conduction problem by using the finite difference method (FDM). Since solving the heat equation analytically over a 240-second period analytically becomes impractical for complex geometries, the domain is discretized into a grid and assigned physical properties to approximate how heat moves between cells over time.

The governing heat transfer equation is the two-dimensional transient conduction form of Fourier's Law, as shown in Equation (5.1). The equation represents T as temperature, $\alpha = \frac{k}{\rho C_p}$ as thermal diffusivity, and the second partial derivative terms describe the change in temperature across the grid in two-dimensional space.

$$\frac{\partial T}{\partial t} = \alpha \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right) \tag{5.1}$$

This equation is approximated numerically using an explicit time-stepping method, where the temperature at each node is updated based solely on the current values of its neighboring cells. Each grid point updates its temperature according to the temperatures of its four immediate neighbors. The use of a rectangular mesh introduces some limitations near domain boundaries, which may reduce real-world accuracy, particularly at the edges. However, increased mesh resolution improves accuracy, allowing the simulation to closely approximate the user-defined geometry. The finite difference update rule used in the code is shown in Equation (5.2).

$$T_{new}(i,j) = T(i,j) + \alpha(i,j) \cdot \Delta t \cdot \left(\frac{T(i+1,j) - 2T(i,j) + T(i-1,j)}{\Delta x^2} + \frac{T(i,j+1) - 2T(i,j) + T(i,j-1)}{\Delta y^2} \right) \tag{5.2}$$

Each layer in the TPS has different thermal properties, such as density, heat capacity, and conductivity. The thermal diffusivity (α) of each material is computed and assigned to every grid cell based on the layer's assigned material. The layers are chosen based on a user's input for layer thickness, angle magnitude, mesh sizing, and layer count. Similarly, the material selection is chosen by a user from bottom to top, assigning each cell in the mesh their material according to their position on the domain and which layer count the cell falls in.

At every time step, the fixed boundary conditions are applied to maintain consistency with prior simulations. These boundary conditions follow the identical conditions as seen in the previous simulations to maintain consistency, which is an initial 300K interior with insulated edges and a fixed 3000K top surface temperature to represent ballistic reentry heating. To simplify user input, this script utilizes custom MATLAB functions that allow control over geometry, materials, and other parameters. This allows users control over angler variation, layer count, horizontal domain, layer thickness, mesh sizing, material choice, initial conditions, boundary conditions, and simulation time. This design choice was implemented to support future research and facilitate user-friendly development; further discussion is provided in Chapter 7.

The simulation runs using a user-defined time-step, which must satisfy stability conditions to ensure accurate and stable results. Specifically, it must satisfy the Courant-Friedrichs-Lewy (CFL) condition for thermal diffusion, as shown in Equation (5.3). In this equation, Δt is the time step, α is the thermal diffusivity, and Δx and Δy are the spatial discretization intervals in the x and y directions, respectively. This ensures that the heat does not propagate unrealistically across the domain within a single time step during the simulation. After running the simulation, the script then outputs the final temperature field contour plot, average bottom surface temperature, and midline temperature slice to verify a two-dimensional conduction path.

$$\Delta t \leq \frac{1}{2\alpha\left(\frac{1}{(\Delta x)^2}+\frac{1}{(\Delta y)^2}\right)} \tag{5.3}$$

5.3 Verification Cases and Procedure

Verification was performed on both flat and angled multilayer configurations to ensure that the developed numerical models accurately portrayed heat conduction behavior consistent with previous MATLAB and CFD results. As described previously, the geometries were generated from user-defined input using user-defined MATLAB functions based on the desired configuration. Both the geometry and the mesh are generated using these helper functions, creating a 2D rectangular domain subdivided into layers with constant layer thickness. The angled configurations were constructed using a trapezoidal stacking logic with each layer sloped to the normal heat flow direction, while flat configurations were generated using a separate helper function that stacked rectangular layers vertically in a uniform pattern.

Each mesh cell was assigned with a material index corresponding to the one of five materials in the predefined database. Property maps are then defined in each cell for conductivity,

density, and specific heat which are then used to calculate the thermal diffusivity map. This thermal diffusivity map defines how heat behaves and the speed at which it propagates through each material section.

To verify the work of this thesis, angled configurations with varying layer counts (6-11 layers), total thicknesses (25mm-50mm), and angles 3°, 5°, and 10° were explicitly modeled and compared directly to the MATLAB optimization and Fluent CFD results presented in earlier chapters. Similarly, the flat configurations with varying layer counts and thicknesses were modeled to match the optimal TPS configurations found from previous simulations. Each of these simulations were assigned to identical material properties using the same database values as previously. With matching geometry and material properties as shown in Figure 5.1, the script then performed explicit FDM calculations over the user-defined simulation duration.

Figure 5.1 includes a legend bar indicating the material index (1-5), corresponding to C/C, C/SiC, SiC/SiC, Nextel™, and Saffil®, respectively. The views demonstrated in Figure 5.1 also demonstrate the mesh sizing implemented into these simulations, 0.5mm rectangular meshing. This design choice was implemented into each simulation to maintain consistency between simulations and to ensure that the results can be used as verification for the optimal MTPS designs.

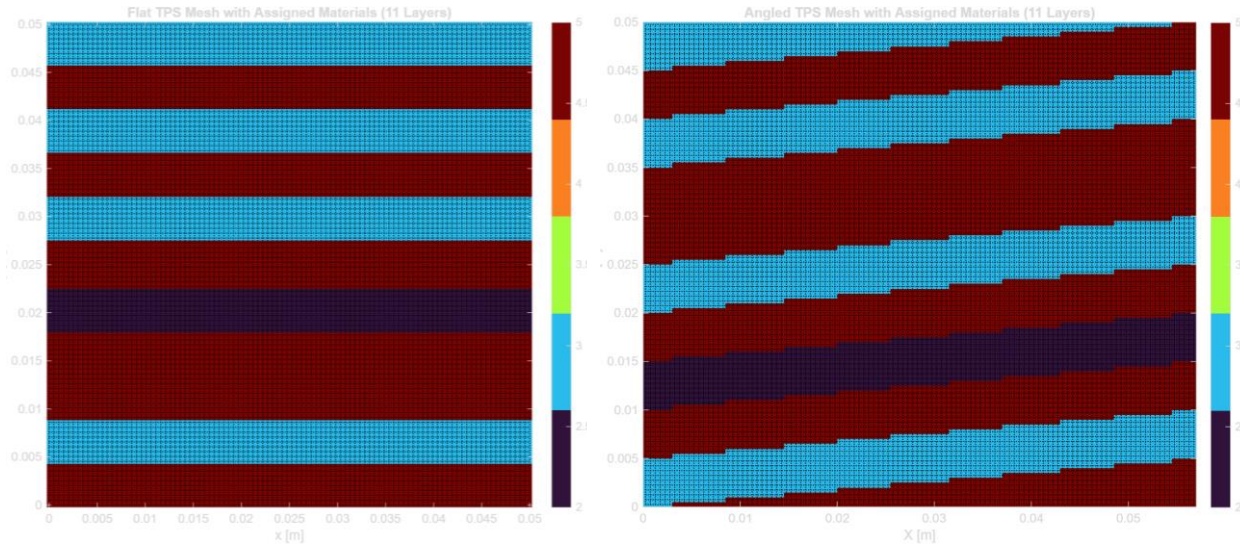


Figure 5.1 – Geometrical mesh view with material index, flat (left) and 5° (right)

Each verification simulation replicated the corresponding geometry, material index, and boundary conditions representative of Figure 5.1. These visuals validate the geometrical configuration, material assignments, and mesh resolution used throughout the domain simulation. They then produce the following temperature distribution plots, lateral variations, and average bottom surface temperatures.

5.4 Results and Discussion

This section presents the results from the manual verification simulations for the optimized MTPS configurations, including final temperature distributions, lateral temperature profiles, and

average bottom surface temperatures across all previously tested configurations. Both flat and angled geometries were evaluated to compare the thermal performance and to further assess how geometric orientation influences heat conduction behavior.

For the flat configurations, lateral temperature variation was specifically investigated to verify the expectation of purely vertical heat conduction. As expected, each flat configuration (both 0° and ELC) exhibited negligible lateral variation across the domain. Therefore, only a single figure representing the lateral variation across the flat domain is presented in Figure 5.2. However, this is not the case for angled configurations, which have unique lateral variations depending on the specific configuration parameters. Note that while the angled configurations shown in the temperature distribution plots may appear to vary in angle, they all share the same internal slope. This visual distortion is a natural result of aspect-ratio stretching applied in post processing for a clearer comparison.

Figure 5.3 presents the final temperature distribution for the optimal flat configurations, providing a visual baseline for conduction behavior in the absence of angled layering. These results are directly related to earlier results discussed in Chapters 3 and 4.

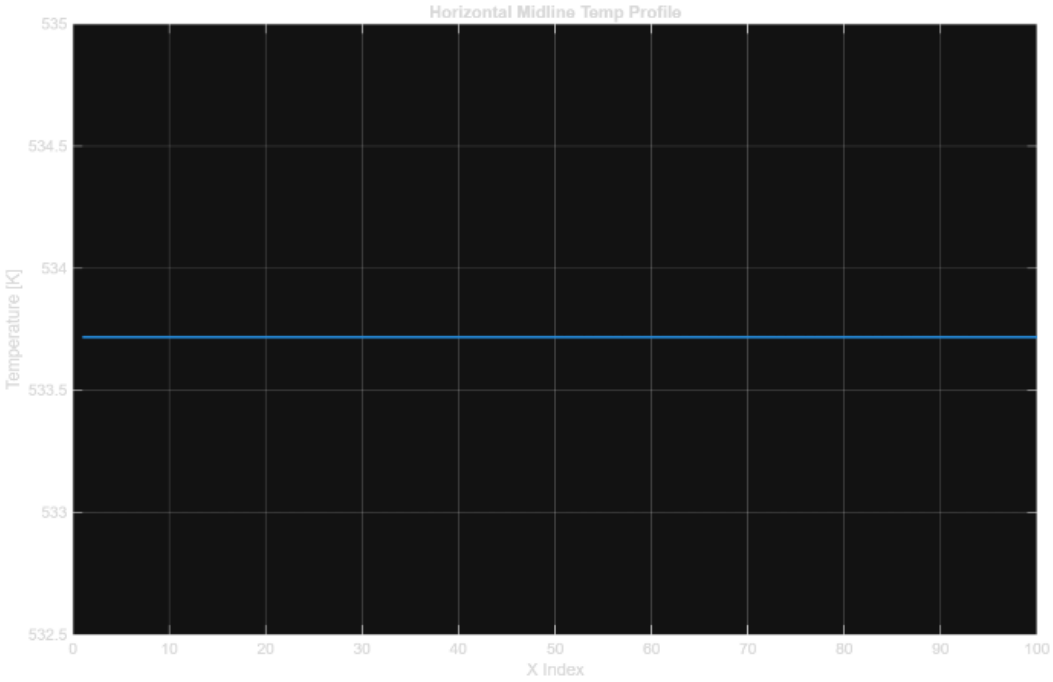
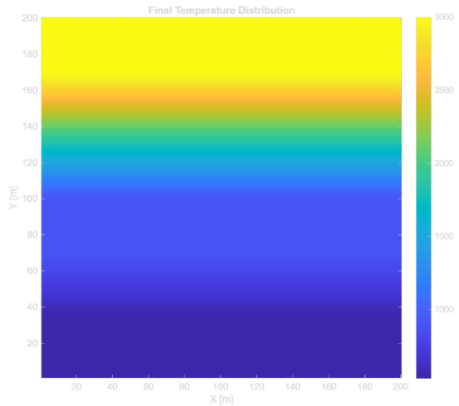
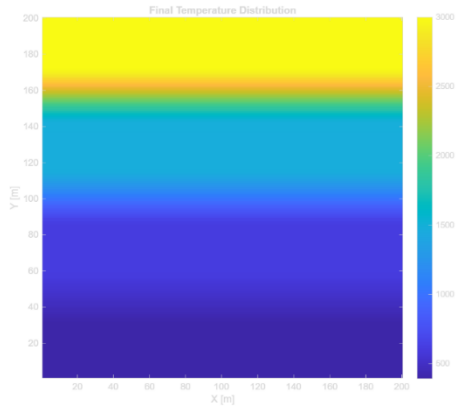


Figure 5.2 – Representative lateral variation across horizontal axis for flat configurations (8 Layers, ELC)

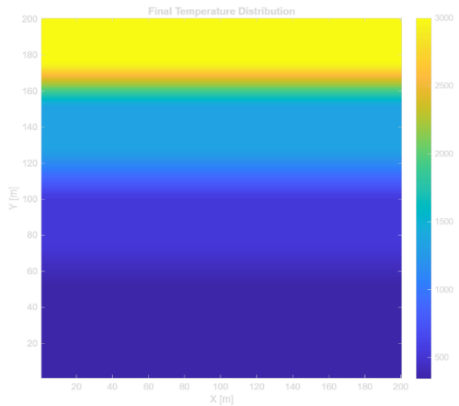
(a) 6 Layers (t = 25 mm)



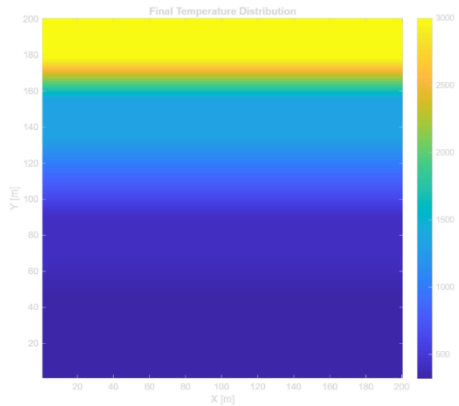
(b) 7 Layers (t = 30 mm)



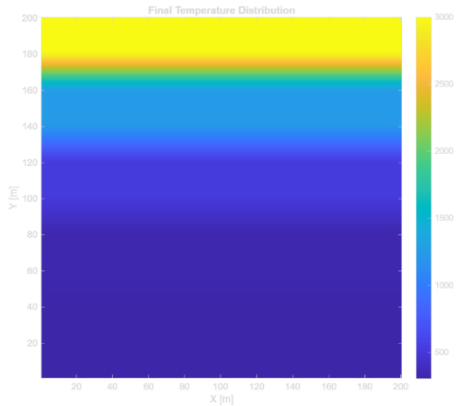
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

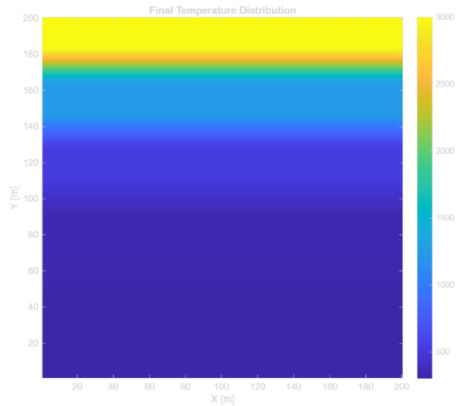
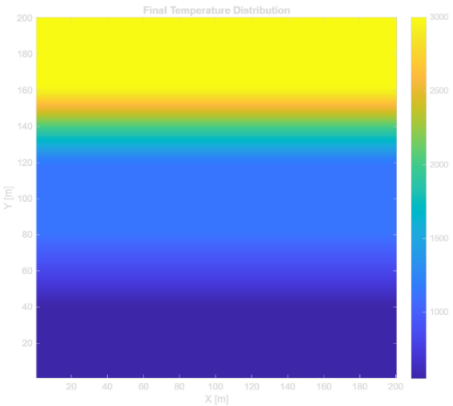


Figure 5.3 – Final temperature distribution (6-11 Layers), ELC

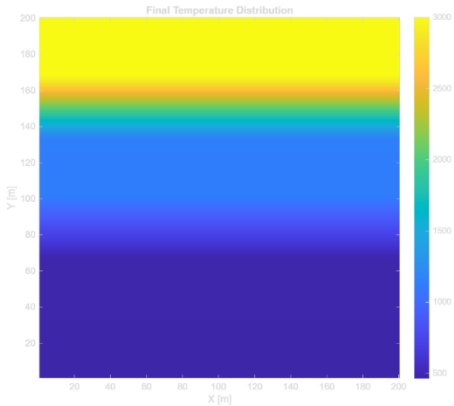
Table 5.1 – Average bottom temperature for each layer count, ELC

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Avg Bottom Temp.	527.14 K	391.19 K	343.56 K	318.73 K	304.21 K	302.21 K

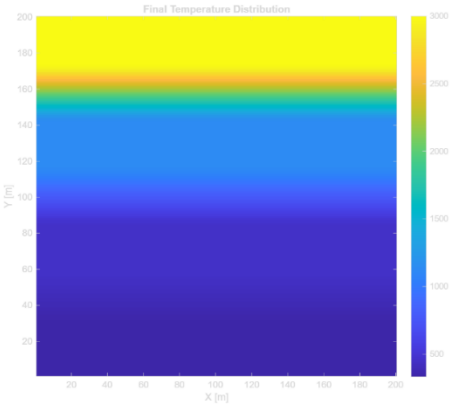
(a) 5 Layers (t = 25 mm)



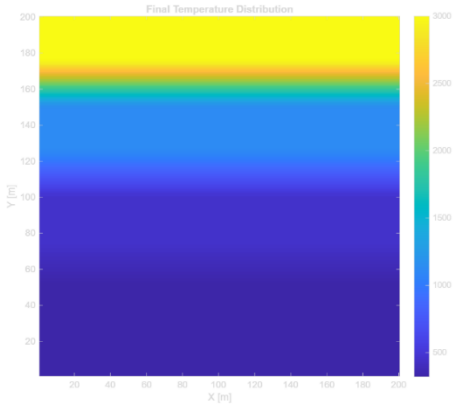
(b) 6 Layers (t = 30 mm)



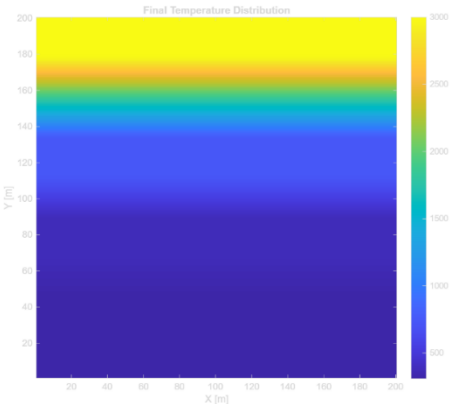
(c) 7 Layers (t = 35 mm)



(d) 8 Layers (t = 40 mm)



(e) 9 Layers (t = 45 mm)



(f) 10 Layers (t = 50 mm)

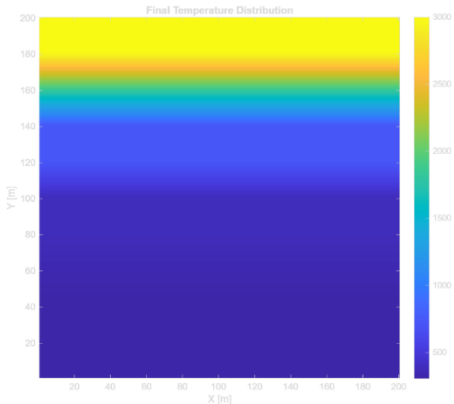
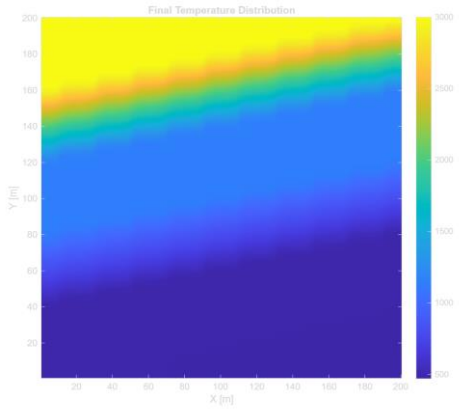


Figure 5.4 – Final temperature distribution (5-10 Layers), 0°

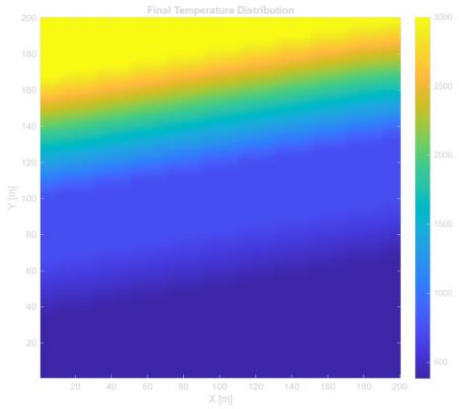
Table 5.2 – Average bottom temperature for each layer count, 0°

Layer Count	**5 Layers**	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**
Avg Bottom Temp.	549.46 K	462.80 K	332.32 K	320.05 K	309.14 K	304.25 K

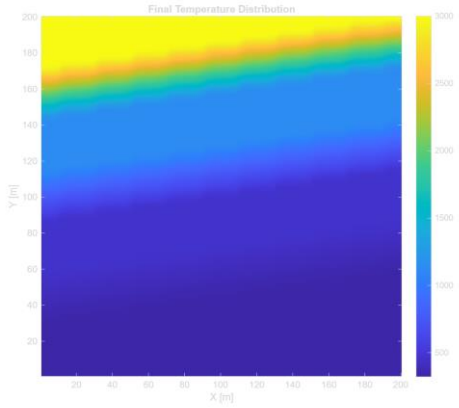
(a) 6 Layers (t = 25 mm)



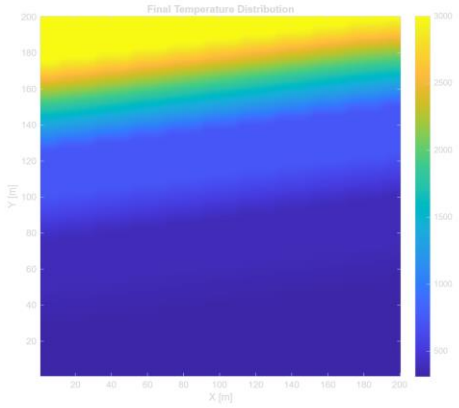
(b) 7 Layers (t = 30 mm)



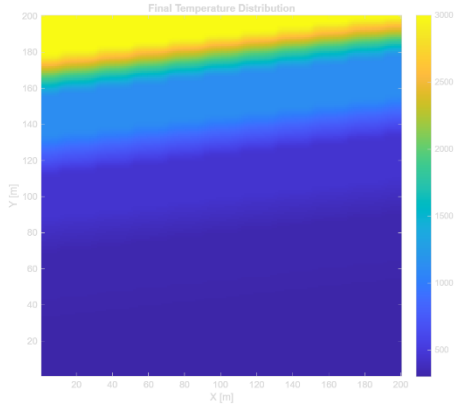
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

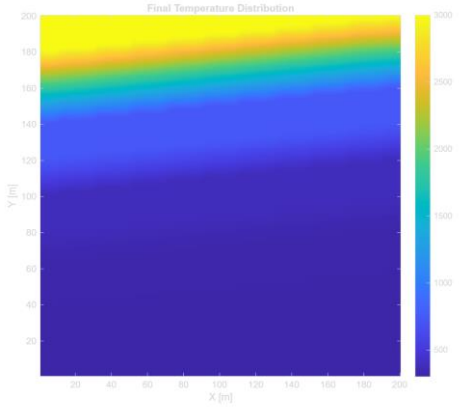
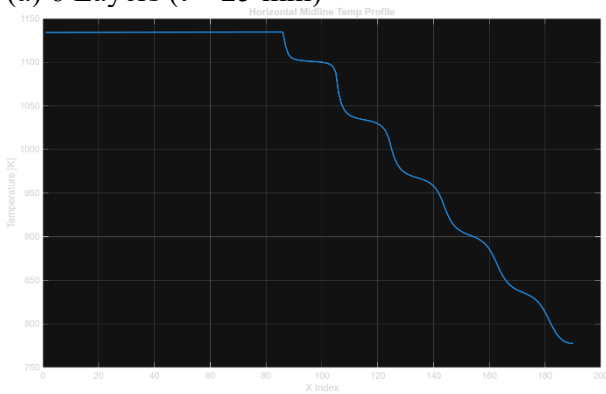
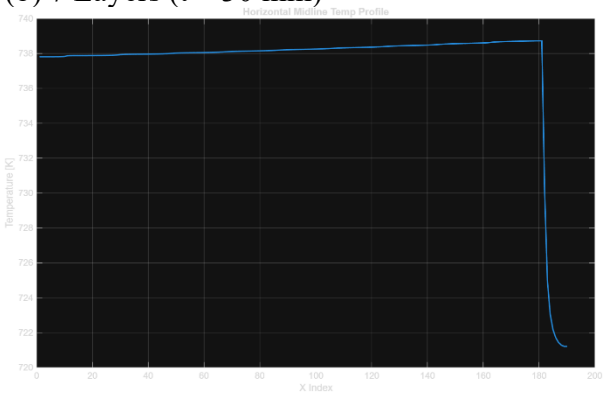


Figure 5.5 – Final temperature distribution (6-11 Layers), 3°

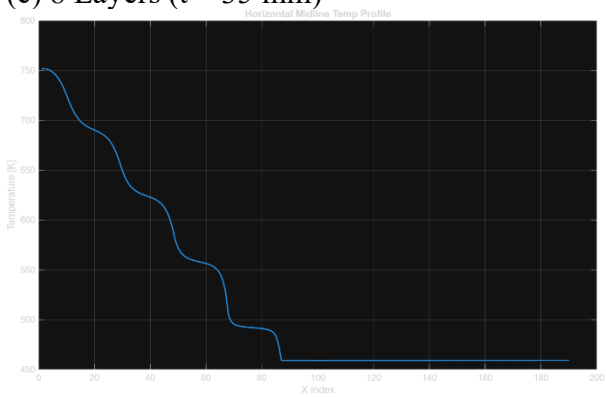
(a) 6 Layers (t = 25 mm)



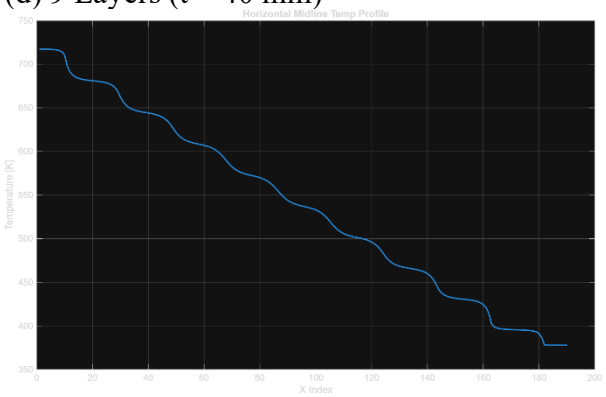
(b) 7 Layers (t = 30 mm)



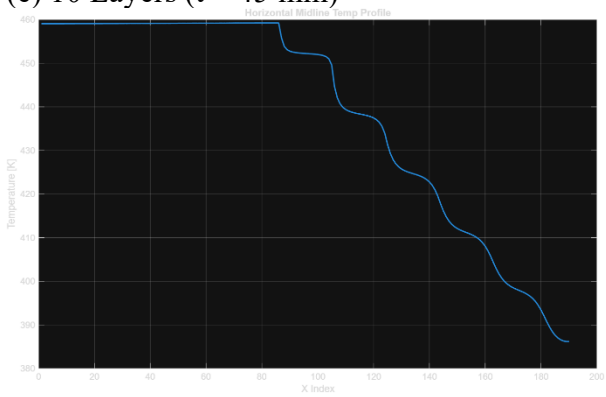
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

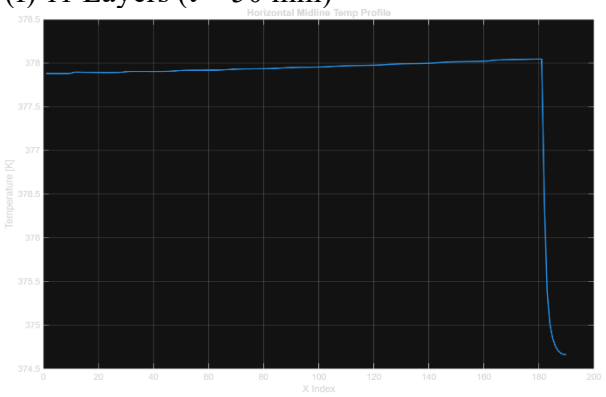
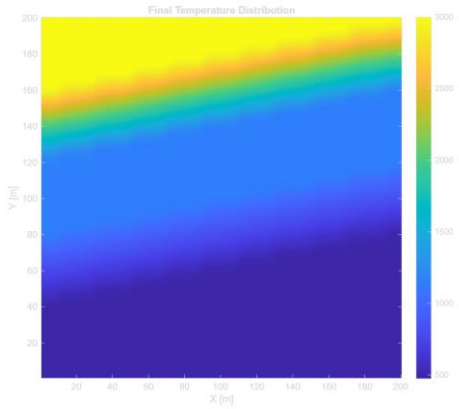


Figure 5.6 – Lateral variation across horizontal axis (6-11 Layers), 3°

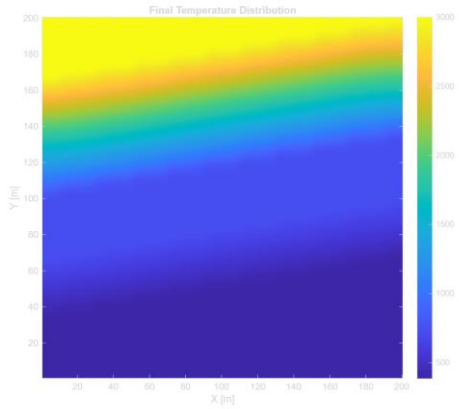
Table 5.3 – Average bottom temperature for each layer count, 3°

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Avg Bottom Temp.	477.92 K	386.32 K	322.75 K	310.40 K	302.78 K	301.24 K

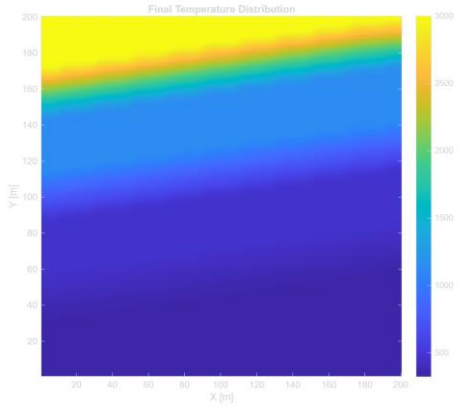
(a) 6 Layers (t = 25 mm)



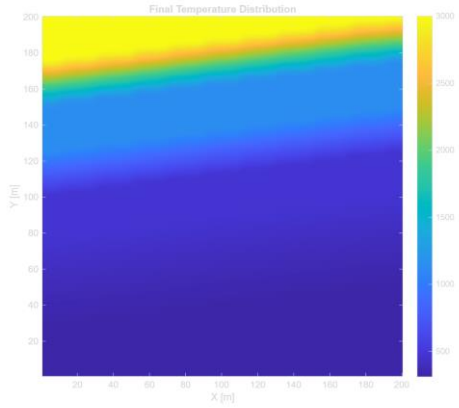
(b) 7 Layers (t = 30 mm)



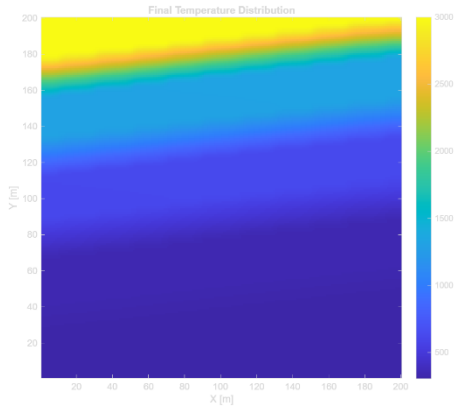
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

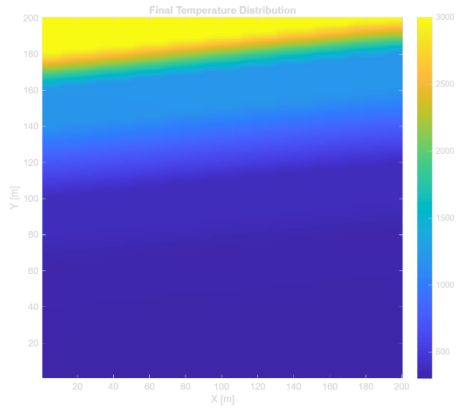
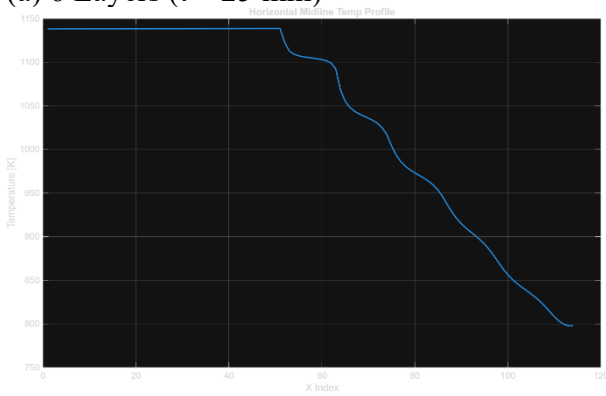
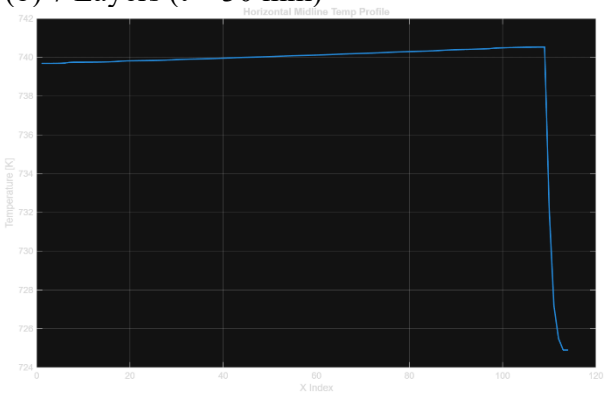


Figure 5.7 – Final temperature distribution (6-11 Layers), 5°

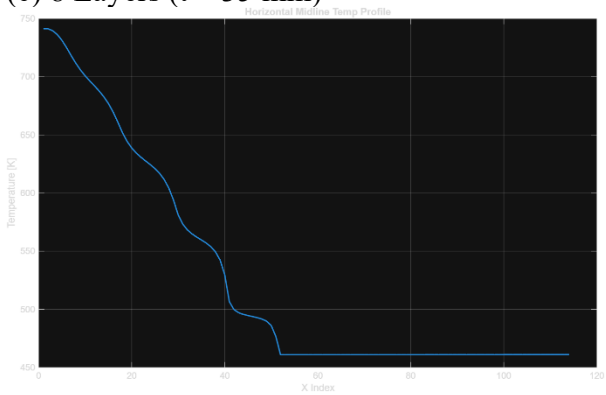
(a) 6 Layers (t = 25 mm)



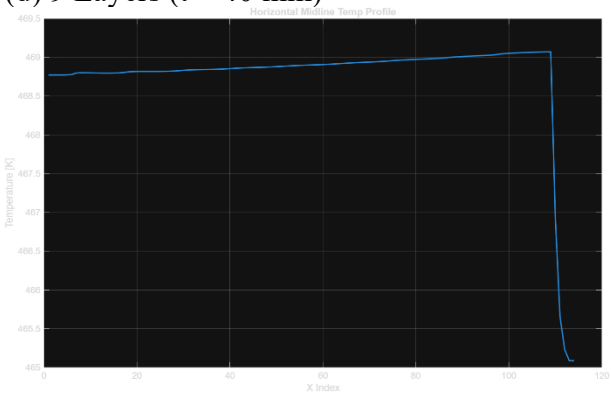
(b) 7 Layers (t = 30 mm)



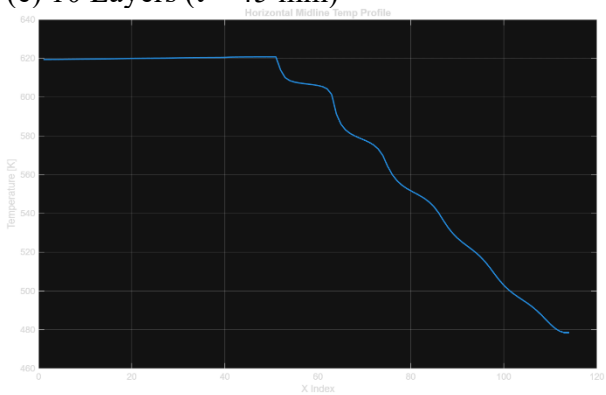
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

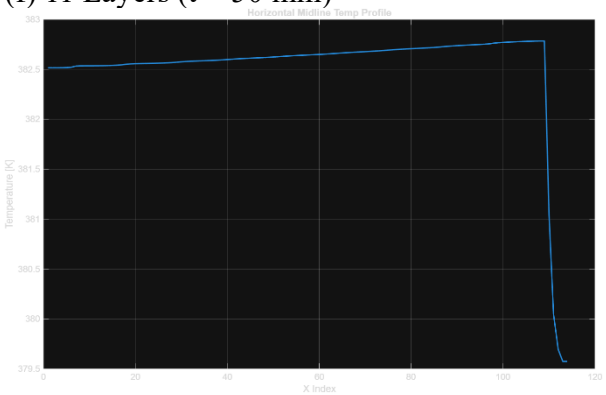
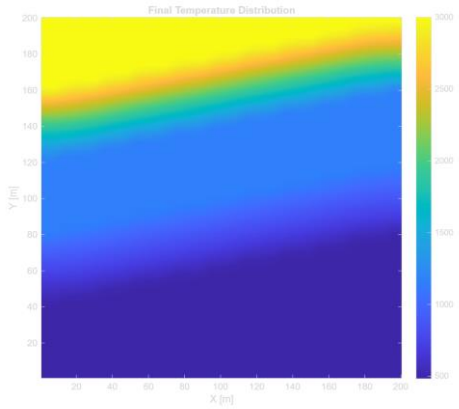


Figure 5.8 – Lateral variation across horizontal axis (6-11 Layers), 5°

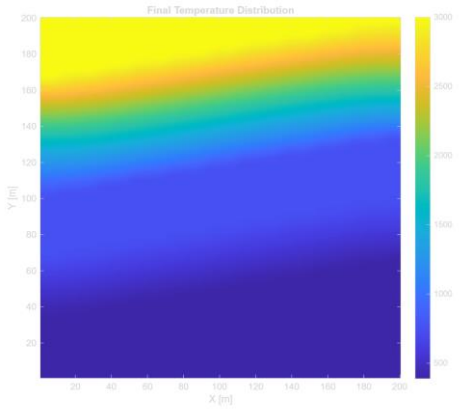
Table 5.4 – Average bottom temperature for each layer count, 5°

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Avg Bottom Temp.	480.20 K	387.38 K	323.20 K	311.01 K	305.17 K	301.27 K

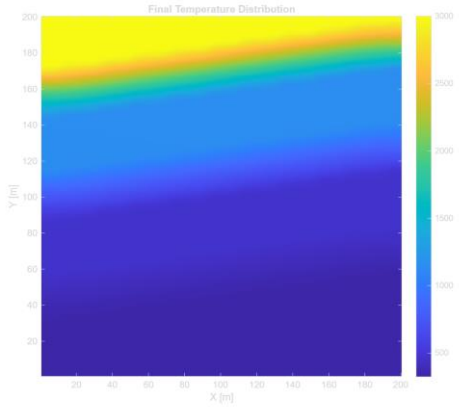
(a) 6 Layers (t = 25 mm)



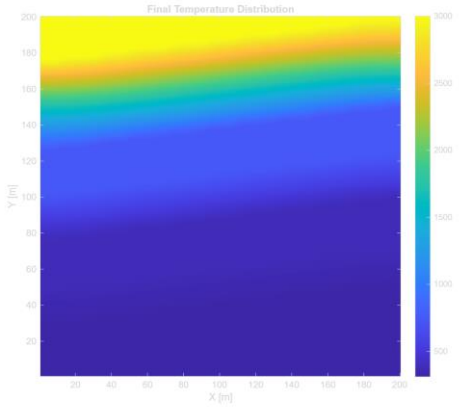
(b) 7 Layers (t = 30 mm)



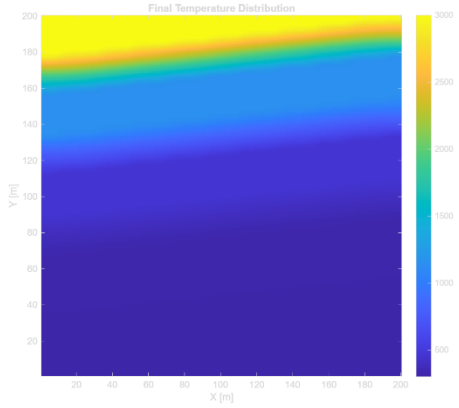
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

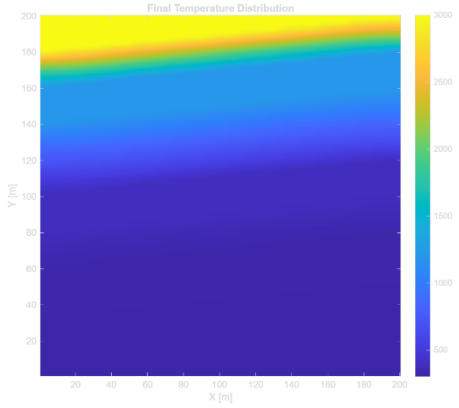
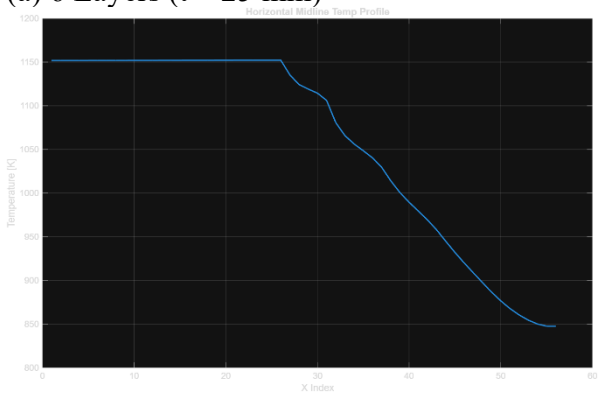
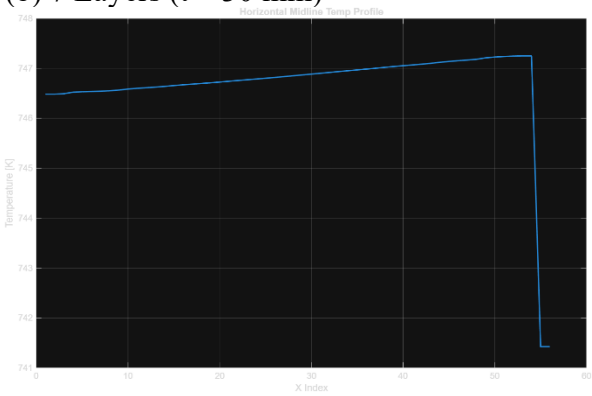


Figure 5.9 – Final temperature distribution (6-11 Layers), 10°

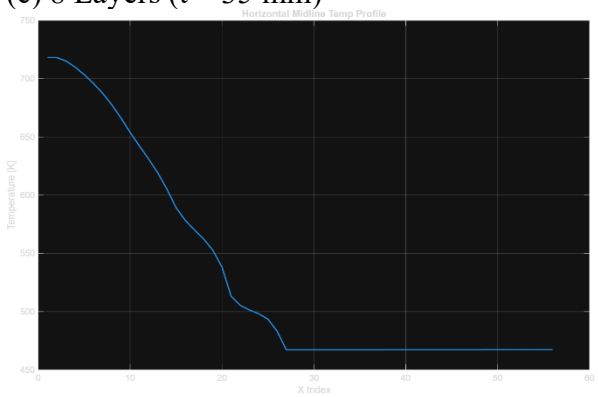
(a) 6 Layers (t = 25 mm)



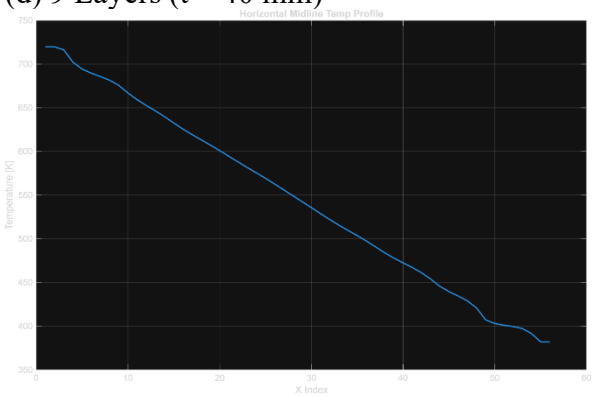
(b) 7 Layers (t = 30 mm)



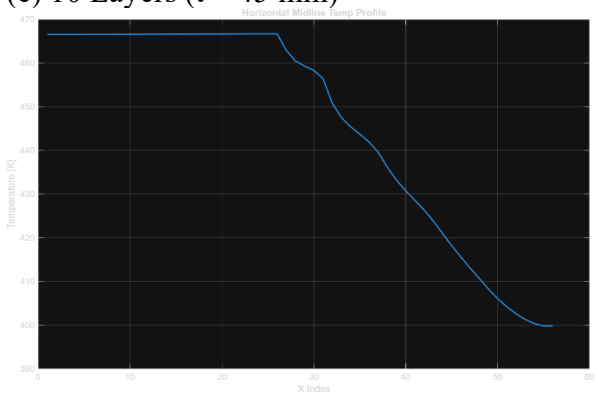
(c) 8 Layers (t = 35 mm)



(d) 9 Layers (t = 40 mm)



(e) 10 Layers (t = 45 mm)



(f) 11 Layers (t = 50 mm)

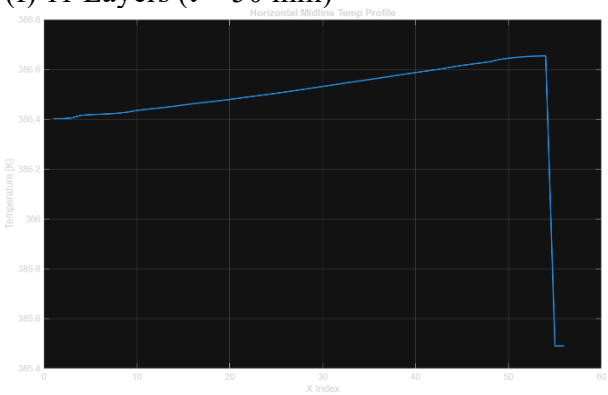


Figure 5.10 – Lateral variation across horizontal axis (6-11 Layers), 10°

Table 5.5 – Average bottom temperature for each layer count, 10°

Layer Count	**6 Layers**	**7 Layers**	**8 Layers**	**9 Layers**	**10 Layers**	**11 Layers**
Avg Bottom Temp.	487.90 K	391.09 K	324.83 K	311.35 K	302.49 K	302.01 K

The manually interpreted results in the above figures show strong agreement between both initially optimized GA configurations and the CFD simulations. All flat configurations, ELC and 0°, demonstrated no lateral heat spreading. In contrast, each angled configuration exhibited lateral heat spreading at varying rates depending on the angle. Notably, the lateral temperature

distributions observed in the angled configurations match the expectations based on the geometry and specific angle. In particular, the 3° configurations demonstrate a clear diagonal temperature gradient from the hotter left side (down-slope) to the cooler right side (up-slope), indicating expected thermal refraction through the angled layers. This directional heat variation begins to diminish at greater angles, such as 10°, where the steeper angle creates more direct heat paths that reduce lateral dissipation. These results are consistent with the thermal redirection hypothesis in Chapter 2 and support the idea that angled geometries can redirect heat flow laterally.

The results demonstrated above reinforce the findings from the optimization framework and support the underlying physics that angled internal geometries can significantly enhance thermal protection performance through passive means.

6. Comprehensive Discussion

Throughout this thesis, the concept of thermal refraction using angled MTPS configurations was thoroughly investigated with multiple computational and analytical methods. The main hypothesis explored in this work, that angled internal layers passively redirect heat laterally to significantly improve thermal protection performance, has been repeatedly supported by the results from the GA optimization, CFD simulations, and custom FDM verification. The consistency of these results strongly validates both the theoretical foundation and practical viability of angled MTPS designs.

Beginning with the MATLAB GA optimization framework, results showed that optimized angled MTPS configurations consistently yielded lower bottom surface temperatures compared to equivalent flat configurations under identical boundary conditions. Increasing the layer count and, consequently, the overall thickness of the configurations naturally improved thermal resistance for all configurations, agreeing with established engineering principles. However, notable improvements emerged when internal layers were angled. More specifically, angled configurations between 3° and 5° consistently achieved lower bottom surface temperatures compared to both flat configurations and steeper configurations, demonstrating an optimal angle range for thermal redirection efficiency. These improvements became evidently distinct once the genetic algorithm converged on an optimal configuration.

During the optimization, the GA highly favored silicon-carbide based composites and Nextel™ as the preferred insulating material. These choices likely stemmed from the remarkable thermal diffusivity of SiC-based composites, effectively redirecting heat flux laterally, while the low thermal conductivity and density of Nextel™ made it ideal for internal insulation. Notably, the GA consistently yielded configurations resembling well-known sandwich-layer patterns [14], with composites on the hot exterior surface and insulating materials towards the interior, despite the algorithm not being explicitly programmed to generate such patterns. This displays evolutionary behavior, observed consistently across generations, which validates the GA's ability to identify realistic and efficient MTPS designs within practical engineering constraints.

Subsequent CFD validation was performed using ANSYS Fluent to replicate and further analyze the optimized MTPS configurations previously produced. Some inherent differences became apparent when translating simulation parameters into Fluent, notably the meshing strategies and solver methods. Fluent employed structured quadrilateral meshing compared to the triangular mesh used in MATLAB's PDE Modeler Toolbox. Moreover, it employed a density-based energy solver as opposed to MATLAB's finite element method (FEM) to solve heat transfer problems. Despite these methodological variations, both methods generated remarkably consistent temperature distributions and bottom surface temperature trends. In fact, due to Fluent's superior mesh resolution and parallel processing capabilities, the results provided clearer visual confirmation of thermal refraction within the angled-layer geometries. These refined CFD results further reinforced confidence in the GA optimization findings.

Between simulations, minor numerical discrepancies were observed in average bottom surface temperatures as detailed in Table 6.1. These minor variations are primarily due to

differences in numerical discretization, mesh resolution, and solver methodologies inherent to the software utilized. However, these variations did not undermine the primary conclusions or the validity of the angled MTPS hypothesis. Instead, the consistency of these trends across methodologies highlights the reliability and robustness of the GA optimization framework and further supports the efficiency of angled MTPS configurations.

Table 6.1 – Average percentage error between practiced methodologies across all MTPS configurations

Methodology Comparison	**Average Temperature Difference**
GA Optimization vs Fluent CFD	0.189 K
GA Optimization vs Manual Verification	24.16 K
Fluent CFD vs Manual Verification	24.27 K

The final verification step involved a custom MATLAB script using the finite difference method (FDM). This method transparently demonstrated the fundamental physics behind thermal refraction by explicitly calculating transient heat conduction across each cell within the geometry. Importantly, the verification script was developed independently of built-in PDE solvers or external numerical libraries, using a basic explicit numerical approach. The results obtained through this manual verification closely matched the results obtained by both the GA optimization and Fluent simulations. The agreement of these results strongly confirms that the observed lateral heat flux redirection and reduced bottom-surface temperatures are not computational discrepancies, but rather a genuine physical phenomenon firmly established in fundamental heat transfer principles.

The combined results from the GA optimization, CFD validation, and manual FDM verification undoubtedly support the hypothesis that angled MTPS geometries significantly improve passive thermal protection performance with thermal refraction. This development has immediate and potentially valuable implications for aerospace applications, especially for hypersonic spacecraft undergoing ballistic reentry conditions. Practical benefits of this approach include the potential to reduce overall mass and material usage without compromising thermal performance, potentially improving durability, reusability, and decreasing overall mass of the vehicle. Notably, these improvements could consequentially enhance vehicle durability, reliability, mission safety, and financial feasibility.

Additionally, the demonstrated consistency and robustness across the independent computational methodologies suggest that these angled-layer designs could feasibly be pursued or perhaps enhanced through future experimental or real-world manufacturing efforts. Advances in composite engineering and additive manufacturing techniques further support the practicality and potential integration of angled MTPS configurations into improved aerospace designs and space exploration technologies.

Furthermore, the computational frameworks developed for this thesis are modular and adaptable, allowing researchers and industry professionals with access to more detailed property

data to potentially explore further MTPS optimization possibilities based on material selection. Future advancements in anisotropic material engineering and manufacturing precision could further expand upon the results and enhance the thermal management capabilities demonstrated in this thesis.

7. Conclusion and Future Work

Ultimately, the results of this thesis collectively and conclusively demonstrate the thermal efficiency of angled MTPS configurations and their promising viability for future aerospace applications. The concept of thermal refraction, supported by both manual and computational solving methods, offers unique and meaningful advantages to MTPS design compared to traditionally flat configurations. Through precise material selection, geometric arrangement, and the exploitation of directional heat conduction behavior, this research not only validates a promising physical concept but also establishes a strong foundation for its future applications in thermal management and aerospace system design.

The robustness and modularity of the computational frameworks provided (as shown in the appendices) demonstrate complete and reliable results while leaving room for future expansion. The verification script was intentionally designed to be user-friendly and modifiable, allowing future researchers to adjust parameters such as angle variation, layer count, geometry sizing, and simulation conditions.

Areas of improvement that could further improve the presented findings include expanding the GA optimization framework to test a wider range of internal angles for a fixed thickness and layer count to better determine an optimal refraction angle range for minimizing bottom surface temperatures. Additional improvements could involve optimizing individual layer thicknesses, exploring broader variations in total TPS thickness, and introducing angle differences across different layers rather than using a constant angle value. Combined with an expanded materials database, these future advancements could potentially yield even better-performing MTPS designs than those presented in this study.

Regardless of these future opportunities, the work presented and discussed in this thesis provides a detailed, thorough, and well-rounded exploration of thermal refraction principles, computational and analytical heat transfer methodologies, and practical material and aerospace engineering techniques. The consistent results across multiple independent verification methods, combined with the transparency and flexibility of the developed tools, establish this research as a meaningful step forward in the design of more efficient and effective TPS systems for extreme aerospace environments.

References

[1] Al-Jothery, H. K. M. “A review of ultra-high temperature materials for thermal protection system,” *IOP Conference Series: Materials Science and Engineering*, 863, 012003, Oct. 2019. https://doi.org/10.1088/1757-899X/863/1/012003

[2] Uyanna, O. and Najafi, H. “Thermal protection systems for space vehicles: a review on technology development, current challenges, and future prospects,” *Acta Astronautica*, 176, 341–356, Nov. 2020. https://doi.org/10.1016/j.actaastro.2020.06.047

[3] Ren, X. M., Zhang, M., and Nie, H. (2021). “Optimization of multilayer thermal protection system by using phase change material under aerodynamic heating,” *Applied Thermal Engineering*, *191*, 11667, 2021. https://doi.org/10.1016/j.applthermaleng.2021.116677

[4] Natali, M., Kenny, J. M., and Torre, L. “Science and technology of polymeric ablative materials for thermal protection systems and propulsion devices: a review,” Progress in Materials Science, 84, 192–275., 2016. https://doi.org/10.1016/j.pmatsci.2016.08.003

[5] Li, W., Huang, J., Zhang, Z., Wang, L., Huang, H., and Liang, J. “A model for thermal protection ablative material with local thermal non-equilibrium and thermal radiation mechanisms,” *Acta Astronautica*, 183, 101–111., 2021 https://doi.org/10.1016/j.actaastro.2021.03.001

[6] Xie, G., Qi, W., Zhang, W., Sunden, B., and Lorenzini, G. “Optimization design and analysis of multilayer lightweight thermal protection structures under aerodynamic heating conditions,” *Journal of Thermal Science and Engineering Applications*, *5*(1)., 2013 https://doi.org/10.1115/1.4007919

[7] Blosser, M. L. “Investigation of fundamental modeling and thermal performance issues for a metallic thermal protection system design,” Reston, VA; *AIAA*., Jan. 2002. https://doi.org/10.2514/6.2002-503

[8] Marley, C. D., and Driscoll, J. F. “Optimization of active and passive thermal protection systems for a hypersonic vehicle,” *Journal of Aircraft*, *59*(1), 173–183., 2022. https://doi.org/10.2514/1.c036411

[9] Henline, W., Olynick, D., and Palmer, G. “TPS sizing for access-to-space vehicles,” Moffett Field; *NASA Ames Research Center*., 1995.

[10] Xie, G., Wang, Q., Sunden, B., and Zhang, W. “Thermomechanical optimization of lightweight thermal protection system under aerodynamic heating,” *Applied Thermal Engineering*, *59*(1–2), 425–434., 2013. https://doi.org/10.1016/j.applthermaleng.2013.06.002

[11] Pichon, T., Lacoste, M., and Barreteau, R. “Integrated thermal protection systems and heat resistant structures,” *57th International Astronautical Congress*., 2006. https://doi.org/10.2514/6.iac-06-d2.5.09

[12] Le, V. T., Ha, N. S., and Goo, N. S. “Advanced sandwich structures for thermal protection systems in hypersonic vehicles: a review,” *Composites Part B: Engineering*, *226*, 109301., 2021. https://doi.org/10.1016/j.compositesb.2021.109301

[13] Qiao, Y., Liu, P., Liu, W., and Liu, Z. “Analysis and optimization of flow and heat transfer performance of active thermal protection channel for hypersonic aircraft,” *SSRN Electronic Journal*., 2022 https://doi.org/10.2139/ssrn.41419306

[14] Steeves, C. A., and Evans, A. G. “Optimization of thermal protection systems utilizing sandwich structures with low coefficient of thermal expansion lattice hot faces,” *Journal of the American Ceramic Society*, *94*(s1)., 2011. https://doi.org/10.1111/j.1551-2916.2011.04447.x

[15] Huang, A. “Thermal protection materials branch - design and analysis,” NASA., July 25, 2023.

[16] Haddad, N., McWilliams, H., and Wagoner, P. “NASA engineering design challenges thermal protection systems,” Washington, D.C.; NASA., 2023.

[17] Kolodziej, P. “Strategies and approaches to TPS design,” *Moffett Field; NATO STO.,* May 2024.

[18] Vemuri, K. P., Canbazoglu, F. M., and Bandaru, P. R. “Guiding conductive heat flux through thermal metamaterials,” *Applied Physics Letters*, 105(19)., 2014. https://doi.org/10.1063/1.4901885

[19] Pavlosky, James E., and Leslie G. St. Leger. “Apollo experience report – thermal protection subsystem.” NTRS - NASA Technical Reports, NASA, NASA-TN-D-6725, Jan. 1974

[20] 3M, “3M™ Nextel™ ceramic fibers and textiles,” 3M Materials Catalog, 2021. [Online]. Available: https://www.3m.com

[21] MatWeb, “Carlisle 201HL carbon-carbon composite,” MatWeb Material Property Data, [Online]. Available: https://www.matweb.com. [Accessed: Dec, 2024].

[22] MatWeb, “COI Ceramics Sylramic™ S400 silicon carbide ceramic matrix composite, carbon,” MatWeb Material Property Data, [Online]. Available: https://www.matweb.com. [Accessed: Dec, 2024].

[23] AZoM, “Properties: Supplier Data - SAFFIL (96% Alumina/4% Silica) Fibre (Goodfellow),” AZoM Materials Data Sheet, [Online]. Available: https://www.azom.com. [Accessed: Dec, 2024].

[24] MatWeb, “Saffil® Alumina/Silica SiO_2 4%/Al_2O_3 96%,” MatWeb Material Property Data, [Online]. Available: https://www.matweb.com. [Accessed: Dec, 2024].

[25] MatWeb, “Thermal Transfer Composites PRIME™ MCX-693™ aluminum silicon carbide composite, reinforced,” MatWeb Material Property Data, [Online]. Available: https://www.matweb.com. [Accessed: Dec, 2024].

[26] Turner, M. J., and Clough, R. W. "Effective stiffness of irregular grids by finite element techniques," Computers and Structures, 1(1), 3–12, 1971. https://doi.org/10.1016/S1364-8152(96)00030-8

[27] Wang, L., Zhang, W., and Sunden, B. "Numerical investigation of thermal protection system performance considering material degradation," Computational Materials Science, 50(2), 497–503, 2010. https://doi.org/10.1016/j.commatsci.2010.09.012

Appendix A: GA Optimization Script - Flat Geometries

```
%% GA Optimization Script - Flat Geometries

clc;
close all;
clear all;

%% Material properties
materials = {'C/C', 'C/SiC', 'SiC/SiC', 'Saffil', 'Nextel'};  % List of Materials
dens = [1750, 2100, 2980, 10, 96];  % Density of Materials (kg/m^3)
k = [20, 30, 180, 25.6, 0.112];  % Thermal conuctivity (W/m-K)
Cp = [850, 800, 760, 1000, 1050];  % Specific Heat (K/kg-K)

% Define material categories
composites = [1,2,3];
insulators = [4,5];

%% GA Parameters
popSize = 300; % Population size
numGenerations = 30; % Number of generations per run
mutationRate = 0.1; % Mutation rate
generationsNoImprovement = 7; % Generations to wait before stopping if no improvement
tlist = 0:1:240; % Time range for solving the PDE
layerthickness = 0.005; % Fixed layer thickness [m]
layerRange = 4; % Range of layer counts to be tested
domainWidth = 0.05;  % [m]  (same width as Fluent/MATLAB geometry)

% Prepare cell arrays to store results for each layer count
allBestConfigs = cell(length(layerRange), 1);
allBestMetrics = cell(length(layerRange), 1);
allGenPlots = cell(length(layerRange), 1);
allPastAttempts = cell(length(layerRange), 1);
allBestResults = cell(length(layerRange), 1); % Store best simulation results

if isempty(gcp('nocreate'))
    parpool; % Starts parallel processing if not already active
end

scriptDir = pwd; % Get current script directory
masterFolder = fullfile(scriptDir, 'flatTPS_penalty_results');
if ~exist(masterFolder, 'dir')
    mkdir(masterFolder);
end

%% Outer loop: Run GA for each layer count
for lrIdx = 1:length(layerRange)
    numlayers = layerRange(lrIdx);
    total_thickness = numlayers * layerthickness;

    fprintf('\n***** Running GA for %d layers (Total thickness = %.3f m) *****\n',
numlayers, total_thickness);

    % Create folder to store figures
        folderName = fullfile(masterFolder, sprintf('Results_Layers_%d', numlayers));
```

```
        if ~exist(folderName, 'dir')
            mkdir(folderName);
        end

    % Initialize GA population
    population = cell(popSize, 1);
    for i = 1:popSize
        population{i}.Config = materials(randi(length(materials), 1, numlayers));
    end

    % Results variables
    bestConfig = [];
    bestMetric = Inf;
    bestResult = [];
    metricsOverGenerations = zeros(numGenerations, 1);
    noImprovementCounter = 0;
    noMetricImprovementCounter = 0;
    noTempImprovementCounter   = 0;
    bestBottomTemp = Inf;
    pastAttempts = [];
    generationPlots = cell(numGenerations, 1);
    bottomTempsOverGenerations = zeros(1, numGenerations);

    %% Main GA loop
    for gen = 1:numGenerations
        disp(['Generation: ', num2str(gen)]);
        tempMetrics = zeros(popSize, 1);
        bottomTemps = zeros(1, popSize);
        tempConfigs = cell(popSize, 1);
        workerResults = repmat(struct('Mesh', [], 'Temperature', []), popSize, 1); %
Stores sim results

        parfor p = 1:popSize
            currentConfig = population{p}.Config;
            workerThermalModel = createpde('thermal', 'transient'); % Create PDE
model
            gd = []; % Geometry intialization
            currentheight = 0; % Origin

            % Define & Create Geometry
            for i = 1:numlayers
                gd = [gd,
[3;4;0;0.05;0.05;0;currentheight;currentheight;currentheight+layerthickness;currenthe
ight+layerthickness]];
                currentheight = currentheight + layerthickness;
            end

            names = char(arrayfun(@(i) ['R' num2str(i)], 1:numlayers,
'UniformOutput', false));
            namedgeo = decsg(gd, strjoin(cellstr(names), '+'), names');
            geometryFromEdges(workerThermalModel, namedgeo);

            for i = 1:numlayers
                materialIdx = find(strcmp(materials, currentConfig{i}));
```

```
                thermalProperties(workerThermalModel, 'ThermalConductivity',
k(materialIdx),...
                    'MassDensity', dens(materialIdx), 'SpecificHeat',
Cp(materialIdx), 'Face', i);
            end

            numEdges = workerThermalModel.Geometry.NumEdges;

            % Calculate the sidewall edges dynamically
            sidewalls = (3 + numlayers - 1):(3 + 2*(numlayers - 1));

            % Boundary Conditions
            thermalBC(workerThermalModel, 'Edge', 2, 'Temperature', 3000); % Top
layer constant temperature
            thermalBC(workerThermalModel, 'Edge', sidewalls, 'HeatFlux', 0); %
Adiabatic sidewalls

            % Initial Conditions
            thermalIC(workerThermalModel, 300);

            % Generate the Mesh for PDE model
            generateMesh(workerThermalModel, 'Hmax', 0.005/2.5);
            resultsWorker = solve(workerThermalModel, tlist);
            bottomboundarynodes = find(resultsWorker.Mesh.Nodes(2, :) == 0);
            bottomTemp = mean(resultsWorker.Temperature(bottomboundarynodes, end));

            layerMaterialIdx = zeros(1, numlayers);
            for l = 1:numlayers
                layerMaterialIdx(l) = find(strcmp(materials, currentConfig{l}));
            end

            %% Penalty System Start
            penalty = 0;  % Initialize total penalty for this individual

            % Get top and bottom material *names*
            topMatName    = currentConfig{end};  % Top layer
            bottomMatName = currentConfig{1};    % Bottom layer

            % Convert to index
            topMatIdx    = find(strcmp(materials, topMatName));
            bottomMatIdx = find(strcmp(materials, bottomMatName));

            % Penalty A: Material-type convention
            if ismember(topMatIdx, insulators)
                penalty = penalty + 500;
            end
            if ismember(bottomMatIdx, composites)
                penalty = penalty + 500;
            end

            % Penalty B: Thermal conductivity thresholds
            k_top = k(topMatIdx);
            k_bot = k(bottomMatIdx);
            k_max = max(k);
```

```
            if k_top < 0.25 * k_max
                penalty = penalty + 1000;
            end
            if k_bot > 0.25 * k_max
                penalty = penalty + 1000;
            end

            % Penalty C: Adaptive penalty based on gradient flip
            adaptivePenalty = 1500 * (max(0, (k_bot - k_top)) / k_max)^2;
            penalty = penalty + adaptivePenalty;

            % Penalty D: Mass-based penalty
            mass = sum(dens(layerMaterialIdx)) * domainWidth * layerthickness;
            maxMass = max(dens) * numlayers * domainWidth * layerthickness;
            massRatio = mass / maxMass;

            penaltyD = 1500 * massRatio^2;
            penalty = penalty + penaltyD;

            metric = bottomTemp + penalty;

            tempMetrics(p) = metric;
            bottomTemps(p) = bottomTemp;
            tempConfigs{p} = currentConfig;
            workerResults(p).Mesh = resultsWorker.Mesh;
            workerResults(p).Temperature = resultsWorker.Temperature;
        end

        %% Reward system start
        [~, sortedIdx] = sort(bottomTemps);  % Sort by bottom temperature
        numTopConfigs = round(0.25 * popSize);  % Top 25%
        rewardValues = linspace(-1000, -200, numTopConfigs);  % Decreasing reward

        for r = 1:numTopConfigs
            idx = sortedIdx(r);  % Index of one of the best configs
            reward = rewardValues(r);

            tempMetrics(idx) = tempMetrics(idx) + reward;

        end

        %% Process results
        for p = 1:popSize
            pastAttempts = [pastAttempts; struct('Config', {tempConfigs{p}},
'Metric', tempMetrics(p),...
                'BottomTemps', bottomTemps(p), 'TotalThickness', total_thickness)];
        end

        [genBestMetric, genBestIdx] = min(tempMetrics);
        genBestConfig = tempConfigs{genBestIdx};

        % Check if current generation improved either metric OR bottom temp
        if genBestMetric < bestMetric || bottomTemps(genBestIdx) < bestBottomTemps
            bestMetric = genBestMetric;
            bestBottomTemps = bottomTemps(genBestIdx);
```

```
            bestConfig = tempConfigs{genBestIdx};
            bestResult = workerResults(genBestIdx);
            noImprovementCounter = 0;
            bottomTempsOverGenerations(gen) = bottomTemps(genBestIdx);
        else
            noImprovementCounter = noImprovementCounter + 1;
            bottomTempsOverGenerations(gen) = bottomTempsOverGenerations(gen - 1);
        end

        metricsOverGenerations(gen) = genBestMetric;
        bottomTempsOverGenerations(gen) = bestBottomTemps;

        %% Visualization/Current Results
        disp('--- Optimal Configuration for this Layer Count ---');
        for i = 1:length(bestConfig)
            disp(['Layer ', num2str(i), ': ', bestConfig{i}]);
        end
        disp(['Best (Optimal) Metric: ', num2str(bestMetric)]);
        disp(['Average Bottom Temp: ', num2str(bestBottomTemps)]);  % bottom
temperature only

        if noImprovementCounter >= generationsNoImprovement
            disp('Convergence achieved. Stopping early for this layer count.');
            metricsOverGenerations = metricsOverGenerations(1:gen);
            generationPlots = generationPlots(1:gen);
            break;
        end

        figGen = figure;
        x = workerResults(genBestIdx).Mesh.Nodes(1, :);
        y = workerResults(genBestIdx).Mesh.Nodes(2, :);
        T_final = workerResults(genBestIdx).Temperature(:, end);
        [X, Y] = meshgrid(linspace(min(x), max(x), 200), linspace(min(y), max(y),
200));
        contourf(X, Y, griddata(x, y, T_final, X, Y, 'cubic'), 20, 'LineColor',
'none');
        colorbar;
        xlabel('X [m]'); ylabel('Y [m]');
        title(sprintf('Temp Distribution, %d Layers, Generation %d', numlayers,
gen));

        genFileName = fullfile(folderName, sprintf('Layer%d_Generation%d.png',
numlayers, gen));
        saveas(figGen, genFileName);
        generationPlots{gen} = genFileName;
        drawnow;

        %% GA Operators
        [~, sortedIdx] = sort(tempMetrics);
        population = population(sortedIdx);
        newPopulation = population(1:ceil(popSize/2));

        for i = 1:ceil(popSize/2)
            parent1 = population{randi(ceil(popSize/2))};
            parent2 = population{randi(ceil(popSize/2))};
```

```
            crossoverPoint = randi([1, numlayers-1]);
            offspring.Config = [parent1.Config(1:crossoverPoint),
parent2.Config(crossoverPoint+1:end)];
            newPopulation{end+1} = offspring;
        end

        for i = 1:length(newPopulation)
            if rand < mutationRate
                mutationLayer = randi(numlayers);
                newPopulation{i}.Config{mutationLayer} =
materials{randi(length(materials))};
            end
        end
        population = newPopulation;
    end

    %% Store and display final results
    allBestConfigs{lrIdx} = bestConfig;
    allBestMetrics{lrIdx} = bestMetric;
    allGenPlots{lrIdx} = generationPlots;
    allPastAttempts{lrIdx} = pastAttempts;
    allBestResults{lrIdx} = bestResult;

    %% Final visualizations
    figMesh = figure;
    pdeplot(bestResult.Mesh, 'Mesh', 'on'); % Use bestResult.Mesh safely
    title(sprintf('Mesh Visualization for %d Layers', numlayers));
    xlabel('X [m]');
    ylabel('Y [m]');
    saveas(figMesh, fullfile(folderName, sprintf('Mesh_Layer%d.png', numlayers))); %
Save mesh figure

    figure;
    x = bestResult.Mesh.Nodes(1,:);
    y = bestResult.Mesh.Nodes(2,:);
    T_final = bestResult.Temperature(:, end);
    [X, Y] = meshgrid(linspace(min(x), max(x), 200), linspace(min(y), max(y), 200));
    contourf(X, Y, griddata(x, y, T_final, X, Y, 'cubic'), 20, 'LineColor', 'none');
    colorbar;
    xlabel('X [m]'); ylabel('Y [m]');
    title(sprintf('Optimized %d Layers', numlayers));
    saveas(gcf, fullfile(folderName, sprintf('Optimized_%dLayers_Contour.png',
numlayers)));

    figure;
    plot(1:length(metricsOverGenerations), metricsOverGenerations, '-o');
    xlabel('Generation'); ylabel('Best Metric');
    title(sprintf('Optimization Progress (%d Layers)', numlayers));
    grid on;
    saveas(gcf, fullfile(folderName, sprintf('Optimized_%dLayers_Metrics.png',
numlayers)));

    figure;
    plot(1:length(bottomTempsOverGenerations), bottomTempsOverGenerations, '-o',
'LineWidth', 1.5);
```

```
    xlabel('Generation');
    ylabel('Best Bottom Temperature [K]');
    title(sprintf('Bottom Temp Convergence (%d Layers)', numlayers));
    grid on;
    saveas(gcf, sprintf('Bottom_Temp_Convergence_%d_Layers.png', numlayers));
end

% Save results locally
save(fullfile(masterFolder, 'GA_TPS_Results.mat'), 'allBestConfigs',
'allBestMetrics', 'allGenPlots', 'allPastAttempts', 'allBestResults');
```

Appendix B: GA Optimization Script – Angled Geometries

```
%% GA Optimization Script - Angled Geometries

clc;
close all;
clear all;

%% Material properties
materials = {'C/C', 'C/SiC', 'SiC/SiC', 'Saffil', 'Nextel'};  % List of Materials
dens = [1750, 2100, 2980, 10, 96];  % Density of Materials (kg/m^3)
k = [20, 30, 180, 25.6, 0.112];  % Thermal conuctivity (W/m-K)
Cp = [850, 800, 760, 1000, 1050];  % Specific Heat (K/kg-K)

%% GA Parameters
popSize = 300; % Population size
numGenerations = 30; % Number of generations per run
mutationRate = 0.1; % Mutation rate
generationsNoImprovement = 7; % Generations to wait before stopping if no improvement
tlist = 0:1:240; % Time range for solving the PDE
layerthickness = 0.005; % Fixed layer thickness [m]
layerRange = 6:11; % Range of layer counts to be tested
angle = 10; % Pre-define angle to save files accordingly
displayedLayerCount = layerRange - 1; % Shift the naming convention to keep total
thickness consistent

% Prepare cell arrays to store results for each layer count
allBestConfigs = cell(length(layerRange), 1);
allBestMetrics = cell(length(layerRange), 1);
allGenPlots = cell(length(layerRange), 1);
allPastAttempts = cell(length(layerRange), 1);
allBestResults = cell(length(layerRange), 1); % Store best simulation results

if isempty(gcp('nocreate'))
    parpool; % Starts parallel processing if not already active
end

scriptDir = fileparts(mfilename('fullpath'));
masterFolder = sprintf('angled_TPS_penalty_results_%ddeg', angle);
if ~exist(masterFolder, 'dir')
    mkdir(masterFolder);
end

%% Outer loop: Run the GA for each layer count
for lrIdx = 1:length(layerRange)
    numlayers = layerRange(lrIdx);
    total_thickness = numlayers * layerthickness;

    fprintf('\n***** Running GA for %d layers (Total thickness = %.3f m) *****\n',
displayedLayerCount(lrIdx), total_thickness-0.005);

    % Create a folder to store figures for the current layer count
    folderName = fullfile(masterFolder, sprintf('Results_Layers_%d',
displayedLayerCount(lrIdx)));
    if ~exist(folderName, 'dir')
```

```
        mkdir(folderName);
    end

    % Initialize GA population (each individual is a cell array of material names)
    population = cell(popSize, 1);
    for i = 1:popSize
        population{i}.Config = materials(randi(length(materials), numlayers, 1))';
    end

    % Results variables
    bestConfig = [];
    bestMetric = Inf;
    bestResult = [];
    metricsOverGenerations = zeros(numGenerations, 1);
    bottomTempsOverGenerations = zeros(1, numGenerations);
    noMetricImprovementCounter = 0;
    noTempImprovementCounter   = 0;
    bestBottomTemp = Inf;
    noImprovementCounter = 0;
    noImprovementCounter_metric = 0;
    noImprovementCounter_temp = 0;
    pastAttempts = [];
    generationPlots = cell(numGenerations, 1);

    %% Main GA loop
    for gen = 1:numGenerations
        disp(['Generation: ', num2str(gen)]);
        tempMetrics = zeros(popSize, 1);
        tempConfigs = cell(popSize, 1);
        bottomTemps = zeros(1, popSize);  % Store bottom temps for reward system

        % Define angle parameters
        angle = 10; % Set desired angle (in degrees)
        L = layerthickness/tand(angle); % Domain length
        epsilon = 0.0001; % small #

        % For storing results from each worker in parfor
        parfor p = 1:popSize
            currentConfig = population{p}.Config;
            nl = length(currentConfig);

            % Create a thermal PDE model for this worker
            workerThermalModel = createpde('thermal', 'transient');
            gd = [];
            currentheight = 0;
            total_shift = L * tand(angle);       % Total shift applied at the right
side
            for i = 1:numlayers
                layerheight = layerthickness;

                % Left side remains unchanged
                xLeft = 0;
                yBottomLeft = currentheight;
                yTopLeft = yBottomLeft + layerheight;
```

```
                % Right side shifts uniformly for all layers
                xRight = L;
                yBottomRight = yBottomLeft + total_shift; % Shift applied once for
the entire domain
                yTopRight = yBottomRight + layerheight;   % Maintain layer thickness

                if i == 1
                    % Bottommost layer: "Triangle"
                    poly = [3; 4; ...
                        xLeft; xRight; xRight; xLeft; ... % x-coordinates
                        yTopLeft-epsilon; yBottomRight; yTopRight; yTopLeft]; % y-
coordinates
                elseif i < numlayers
                    % Middle layers: Parallelograms
                    poly = [3; 4; ...
                        xLeft; xRight; xRight; xLeft; ... % x-coordinates
                        yBottomLeft; yBottomRight; yTopRight; yTopLeft]; % y-
coordinates
                else
                    % Topmost layer: "Triangle"
                    poly = [3; 4; ...
                        xLeft; xRight; xRight; xLeft; ... % x-coordinates (Top edge
is horizontal)
                        yBottomLeft; yBottomRight; ... % Bottom follows slant
                        yBottomRight+0.00001; yTopLeft]; % Flat top
                end

                % Append to geometry matrix
                gd = [gd, poly];
                currentheight = currentheight + layerthickness;
            end

            %% Define Geometry
            names = char(arrayfun(@(i) ['R' num2str(i)], 1:numlayers,
'UniformOutput', false));
            nameset = strjoin(cellstr(names), '+');
            namedgeo = decsg(gd, nameset, names');
            geometryFromEdges(workerThermalModel, namedgeo);

            % Assign material properties for each layer
            for i = 1:nl
                materialIdx = find(strcmp(materials, currentConfig{i}));
                thermalProperties(workerThermalModel, ...
                    'ThermalConductivity', k(materialIdx), ...
                    'MassDensity', dens(materialIdx), ...
                    'SpecificHeat', Cp(materialIdx), ...
                    'Face', i);
            end

            % Calculate the sidewall edges dynamically
            sidewall = 3 + numlayers - 1;

            % Boundary Conditions
            thermalBC(workerThermalModel, 'Edge', 2, 'Temperature', 3000);
            thermalBC(workerThermalModel, 'Edge', sidewall, 'HeatFlux', 0);
```

```
            % Initial conditions
            thermalIC(workerThermalModel, 300);

            % Generate Mesh
            generateMesh(workerThermalModel, 'Hmax', (0.005/cosd(angle))/4);
            resultsWorker = solve(workerThermalModel, tlist);
            bottomboundarynodes = find(abs(resultsWorker.Mesh.Nodes(2, :) -
min(resultsWorker.Mesh.Nodes(2, :))) < 1e-4);
            T_final = resultsWorker.Temperature(:, end);
            bottomTemp = mean(T_final(bottomboundarynodes));

            %% Penalty system start
            penalty = 0;

            % Material indices
            layerMaterialIdx = zeros(1, numlayers);
            for l = 1:numlayers
                layerMaterialIdx(l) = find(strcmp(materials, currentConfig{l}));
            end
            topMatIdx = layerMaterialIdx(end);
            bottomMatIdx = layerMaterialIdx(1);

            % Penalty A: Material-type convention
            if ismember(topMatIdx, [4,5])  % Insulators
                penalty = penalty + 500;
            end
            if ismember(bottomMatIdx, [1,2,3])  % Composites
                penalty = penalty + 500;
            end

            % Penalty B: Thermal conductivity thresholds
            k_top = k(topMatIdx);
            k_bot = k(bottomMatIdx);
            k_max = max(k);
            if k_top < 0.25 * k_max
                penalty = penalty + 1000;
            end
            if k_bot > 0.25 * k_max
                penalty = penalty + 1000;
            end

            % Penalty C: Adaptive k-gradient
            adaptivePenalty = 1500 * (max(0, (k_bot - k_top)) / k_max)^2;
            penalty = penalty + adaptivePenalty;

            % Penalty D: Mass-based
            mass = sum(dens(layerMaterialIdx)) * (layerthickness * (0.05)); % Area =
width * thickness
            maxMass = max(dens) * numlayers * (layerthickness * (0.05));
            massRatio = mass / maxMass;
            penalty = penalty + 1500 * massRatio^2;

            metric = bottomTemp + penalty;
```

```
    tempMetrics(p) = metric;
    bottomTemps(p) = bottomTemp;
    tempConfigs{p} = currentConfig;
    workerResults(p).Mesh = resultsWorker.Mesh;
    workerResults(p).Temperature = resultsWorker.Temperature;
end

% Consolidate generation results
for p = 1:popSize
    attemptStruct.Config = tempConfigs{p};
    attemptStruct.Metric = tempMetrics(p);
    attemptStruct.TotalThickness = total_thickness;
    pastAttempts = [pastAttempts; attemptStruct];
end

%% Reward system start
[~, sortedIdx] = sort(bottomTemps);  % Sort by bottom temp (ascending)
numTopConfigs = round(0.25 * popSize);  % Top 25%
rewardValues = linspace(-1000, -200, numTopConfigs);  % Rewards scaled

for r = 1:numTopConfigs
    idx = sortedIdx(r);
    reward = rewardValues(r);
    tempMetrics(idx) = tempMetrics(idx) + reward;

end

[genBestMetric, genBestIdx] = min(tempMetrics);
genBestConfig = tempConfigs{genBestIdx};

disp(['Current generation: ', num2str(gen)]);
disp(['Best metric in this generation: ', num2str(genBestMetric)]);
disp(['Average Bottom Temp: ', num2str(bottomTemps(genBestIdx))]);
disp(['Index of best config: ', num2str(genBestIdx)]);
disp(['Current best metric before update: ', num2str(bestMetric)]);

% Update best config if metric improves
if genBestMetric < bestMetric
    bestMetric = genBestMetric;
    bestConfig = genBestConfig;
    bestResult = workerResults(genBestIdx);
    bestBottomTemp = bottomTemps(genBestIdx);
    noMetricImprovementCounter = 0;
else
    noMetricImprovementCounter = noMetricImprovementCounter + 1;
end

% Update best bottom temp if it improves
if bottomTemps(genBestIdx) < bestBottomTemp
    bestBottomTemp = bottomTemps(genBestIdx);
    noTempImprovementCounter = 0;
else
    noTempImprovementCounter = noTempImprovementCounter + 1;
end
```

```
        metricsOverGenerations(gen) = genBestMetric;
        bottomTempsOverGenerations(gen) = bottomTemps(genBestIdx);

        fprintf('--- Best Configuration for Generation %d ---\n', gen);
        for i = 1:length(genBestConfig)
            fprintf('Layer %d: %s\n', i, genBestConfig{i});
        end

        %% Generate a smooth contour plot for this generation using the best worker
result
        figGen = figure;
        x = workerResults(genBestIdx).Mesh.Nodes(1, :);
        y = workerResults(genBestIdx).Mesh.Nodes(2, :);
        T_final = workerResults(genBestIdx).Temperature(:, end);
        [X, Y] = meshgrid(linspace(min(x), max(x), 200), linspace(min(y), max(y),
200));
        T_interp = griddata(x, y, T_final, X, Y, 'cubic');
        contourf(X, Y, T_interp, 20, 'LineColor', 'none');
        colorbar;
        xlabel('X [m]');
        ylabel('Y [m]');
        title(sprintf('Temp Distribution, %d Layers, Generation %d',
displayedLayerCount(lrIdx), gen));

        % Save the generation plot image and keep the figure open
        genFileName = fullfile(folderName, sprintf('Layer%d_Generation%d.png',
displayedLayerCount(lrIdx), gen));
        saveas(figGen, genFileName);
        generationPlots{gen} = genFileName;
        drawnow;  % update display

       if noImprovementCounter_metric >= generationsNoImprovement && ...
           noImprovementCounter_temp >= generationsNoImprovement

            disp('Convergence achieved (Metric + Temp). Stopping early for this layer
count.');
            metricsOverGenerations = metricsOverGenerations(1:gen);
            bottomTempsOverGenerations = bottomTempsOverGenerations(1:gen);
            generationPlots = generationPlots(1:gen);
            break;
        end


        %% --- GA Operators ---
        [~, sortedIdx] = sort(tempMetrics);
        population = population(sortedIdx);

        newPopulation = population(1:ceil(popSize/2));
        for i = 1:ceil(popSize/2)
            parent1 = population{randi(ceil(popSize/2))};
            parent2 = population{randi(ceil(popSize/2))};
            if iscolumn(parent1.Config)
                parent1.Config = parent1.Config';
            end
            if iscolumn(parent2.Config)
```

```
                parent2.Config = parent2.Config';
            end
            crossoverPoint = randi([1, numlayers-1]);
            offspring.Config = [parent1.Config(1:crossoverPoint),
parent2.Config(crossoverPoint+1:end)];
            newPopulation{end+1} = offspring;
        end
        % Mutation: randomly change one layer's material in each individual
        for i = 1:length(population)
            if rand < mutationRate
                mutationLayer = randi(numlayers);                  % Choose a random
layer
                newMaterial = materials{randi(length(materials))};   % Choose a
random material
                population{i}.Config{mutationLayer} = newMaterial;   % Apply mutation
            end
        end
        population = newPopulation;
    end % end GA loop

    %% Store best results for current layer count
    allBestConfigs{lrIdx} = bestConfig;
    allBestMetrics{lrIdx} = bestMetric;
    allGenPlots{lrIdx} = generationPlots;
    allPastAttempts{lrIdx} = pastAttempts;
    allBestResults{lrIdx} = bestResult;  % Save best simulation result for post-
analysis

    disp('--- Optimal Configuration for this Layer Count ---');
    for i = 1:length(bestConfig)
        disp(['Layer ', num2str(i), ': ', bestConfig{i}]);
    end
    disp(['Best (Optimal) Metric: ', num2str(bestMetric)]);

    %% Visualize the Mesh for the Layer Count
    figMesh = figure;
    pdeplot(bestResult.Mesh, 'Mesh', 'on'); % Use bestResult.Mesh safely
    title(sprintf('Mesh Visualization for %d Layers', displayedLayerCount(lrIdx)));
    xlabel('X [m]');
    ylabel('Y [m]');
    saveas(figMesh, fullfile(folderName, sprintf('Mesh_Layer%d.png',
displayedLayerCount(lrIdx)))); % Save mesh figure

    %% Visualize Temperature Distribution for Best Configuration using stored
simulation result
    figure;
    x = bestResult.Mesh.Nodes(1,:);
    y = bestResult.Mesh.Nodes(2,:);
    T_final = bestResult.Temperature(:, end);
    [X, Y] = meshgrid(linspace(min(x), max(x), 200), linspace(min(y), max(y), 200));
    T_interp = griddata(x, y, T_final, X, Y, 'cubic');
    contourf(X, Y, T_interp, 20, 'LineColor', 'none');
    colorbar;
    xlabel('X [m]');
    ylabel('Y [m]');
```

```
    title(sprintf('Smooth Contour Plot for Optimized %d Layers', numlayers));
    contourFileName = fullfile(folderName, sprintf('Optimized_%dLayers_Contour.png',
displayedLayerCount(lrIdx)));
    saveas(gcf, contourFileName);
    drawnow;

    %% Metric Progression Plot for this Layer Count
    figure;
    plot(1:length(metricsOverGenerations), metricsOverGenerations, '-o');
    xlabel('Generation');
    ylabel('Best Metric (Min Bottom Temp)');
    title(sprintf('Optimized %d Layers', displayedLayerCount(lrIdx)));
    grid on;
    metricFileName = fullfile(folderName, sprintf('Optimized_%dLayers_Metrics.png',
displayedLayerCount(lrIdx)));
    saveas(gcf, metricFileName);
    drawnow;

    % Plot Bottom Temp Convergence
    figBottomTemp = figure;
    plot(1:gen, bottomTempsOverGenerations(1:gen), '-o');
    xlabel('Generation');
    ylabel('Best Bottom Temperature [K]');
    title(sprintf('Bottom Temp Convergence (%d Layers)', numlayers));
    grid on;
    saveas(figBottomTemp, fullfile(folderName,
sprintf('BottomTemp_Convergence_%d_Layers.png', numlayers)));

end % end layerRange loop

%% Save all results to file for later analysis
save(fullfile(masterFolder, 'GA_TPS_Results.mat'), 'allBestConfigs',
'allBestMetrics', 'allGenPlots', 'allPastAttempts', 'allBestResults');
```

Appendix C: Manual 2D FDM Framework

```
%% Hand Calculation Verification Script

clc;
clear;
close all;

%  User-Defined Geometry
angle_deg = input('Enter desired angle [degrees]: ');

%% Material Properties
materials = {'C/C', 'C/SiC', 'SiC/SiC', 'Saffil', 'Nextel'};
dens = [1750, 2100, 2980, 96, 96]; % [kg/m^3]
k = [20, 30, 180, 25.6, 0.112]; % [W/mK]
Cp = [580, 800, 760, 1000, 1050]; % [J/kgK]

%% Geometry + Mesh Generation
if angle_deg == 0
    TPS = generate_flat_TPS();
else
    TPS = generate_angled_TPS(angle_deg);
end

[Ny, Nx] = size(TPS.X);

%% Assign Material Properties
TPS.k = k(TPS.material_index);
TPS.rho = dens(TPS.material_index);
TPS.Cp = Cp(TPS.material_index);

%% Boundary & Initial Conditions
T_init = input('Initial temperature inside domain [K]: ');
T_top = input('Top surface temperature [K]: ');
bottom_type = input('Bottom surface condition ("fixed" or "insulated"): ', 's');
if strcmpi(bottom_type, 'fixed')
    T_bottom = input('Bottom surface temperature [K]: ');
else
    T_bottom = NaN;
end
t_final = input('Total simulation time [s]: ');

dx = TPS.dx;
dy = TPS.dy;
alpha_tmp = TPS.k ./ (TPS.rho .* TPS.Cp);
alpha_max = max(alpha_tmp(:));
dt_max = (dx^2 * dy^2) / (2 * alpha_max * (dx^2 + dy^2));
fprintf('Suggested maximum stable dt: %.6f s\n', dt_max);

dt = input('Time step size [s]: ');
if dt > dt_max
    warning('Time step may be unstable. Suggested dt ≤ %.6f\n', dt_max);
end

% Initialize Temperature Field
```

```
T = T_init * ones(Ny, Nx);
T_new = T;  % Preallocate update buffer

%% Time Simulation
num_steps = round(t_final / dt);
fprintf('\nRunning simulation: %d time steps...\n', num_steps);

for step = 1:num_steps
    for j = 2:Ny-1
        for i = 2:Nx-1
            T_C = T(j,i);
            rhoCp = TPS.rho(j,i) * TPS.Cp(j,i);

            T_E = T(j,i+1); T_W = T(j,i-1);
            T_N = T(j+1,i); T_S = T(j-1,i);

            k_C = TPS.k(j,i);
            k_E = TPS.k(j,i+1); k_W = TPS.k(j,i-1);
            k_N = TPS.k(j+1,i); k_S = TPS.k(j-1,i);

            kx_e = 2 * k_C * k_E / (k_C + k_E);
            kx_w = 2 * k_C * k_W / (k_C + k_W);
            ky_n = 2 * k_C * k_N / (k_C + k_N);
            ky_s = 2 * k_C * k_S / (k_C + k_S);

            qx = (kx_e * (T_E - T_C) - kx_w * (T_C - T_W)) / dx^2;
            qy = (ky_n * (T_N - T_C) - ky_s * (T_C - T_S)) / dy^2;

            T_new(j,i) = T_C + dt * (qx + qy) / rhoCp;
        end
    end

    % Boundary Conditions
    T_new(end,:) = T_top;

    if strcmpi(bottom_type, 'fixed')
        T_new(1,:) = T_bottom;
    else
        T_new(1,:) = T_new(2,:);
    end

    T_new(:,1) = T_new(:,2);
    T_new(:,end) = T_new(:,end-1);

    T = T_new;

    if mod(step, 1000) == 0
        fprintf('Step %d: Center = %.2f K | Bottom = %.2f K\n', ...
            step, T(round(Ny/2), round(Nx/2)), mean(T(1,:)));
    end
end

%% Plot Results
figure('Name','Final Temperature Field','Color','w');
imagesc(TPS.X(1,:), TPS.Y(:,1), T);
```

```
axis xy equal tight;
colormap(parula); colorbar;
xlabel('X [m]'); ylabel('Y [m]');
title('Final Temperature Distribution');

mid_row = round(Ny/2);
figure('Name','Midline Temperature Profile','Color','w');
plot(1:Nx, T(mid_row,:), 'LineWidth', 1.5);
xlabel('X Index'); ylabel('Temperature [K]');
title('Horizontal Midline Temp Profile');
grid on;

avg_bottom_temp = mean(T(1,:));
fprintf('\nFinal Average Bottom Surface Temperature: %.2f K\n', avg_bottom_temp);
```

Appendix D: Flat Geometry Generator Function

```
function TPS = generate_flat_TPS()

clc;
clear;
close all;

fprintf("Flat Layered TPS Geometry Creator \n");

% User Input
num_layers = input('Enter number of layers: ');
Lx = input('Enter horizontal length [m]: ');
layer_height = input('Enter layer thickness [m]: ');
mesh_size = input('Enter mesh resolution [m]: ');

% Derived Quantities
Ly = num_layers * layer_height;
Nx = floor(Lx / mesh_size);
Ny = floor(Ly / mesh_size);
dx = Lx / Nx;
dy = Ly / Ny;

% Material Database
materials = {'C/C', 'C/SiC', 'SiC/SiC', 'Saffil', 'Nextel'};
dens = [1750, 2100, 2980, 10, 96];
k = [20, 30, 180, 25.6, 0.112];
Cp = [850, 800, 760, 1000, 1050];

fprintf('\nSelect a material for each layer (bottom to top):\n');
for i = 1:length(materials)
    fprintf('  %d: %s\n', i, materials{i});
end

material_selection = zeros(num_layers,1);
for i = 1:num_layers
    valid = false;
    while ~valid
        idx = input(sprintf('Layer %d material (1-%d): ', i, length(materials)));
        if idx >= 1 && idx <= length(materials)
            material_selection(i) = idx;
            valid = true;
        else
            fprintf('  Invalid selection. Try again.\n');
        end
    end
end

% Mesh and Material Map
x = linspace(0, Lx, Nx);
y = linspace(0, Ly, Ny);
[X, Y] = meshgrid(x, y);

material_id = zeros(Ny, Nx);
k_map = zeros(Ny, Nx);
```

```
Cp_map = zeros(Ny, Nx);
rho_map = zeros(Ny, Nx);

for i = 1:num_layers
    y_start = round((i - 1) * layer_height / dy) + 1;
    y_end = round(i * layer_height / dy);
    y_start = max(1, min(y_start, Ny));
    y_end = max(1, min(y_end, Ny));

    mat_idx = material_selection(i);
    material_id(y_start:y_end, :) = mat_idx;
    k_map(y_start:y_end, :) = k(mat_idx);
    Cp_map(y_start:y_end, :) = Cp(mat_idx);
    rho_map(y_start:y_end, :) = dens(mat_idx);
end

% CLEAN FLAT GEOMETRY
figure('Name', 'Clean Flat Geometry', 'Color', 'w');
hold on;
skyblue = [0.53, 0.81, 0.98];
for i = 1:num_layers
    y0 = (i - 1) * layer_height;
    y1 = i * layer_height;

    fill([0, Lx, Lx, 0], ...
         [y0, y0, y1, y1], ...
         skyblue, 'EdgeColor', 'k');
end
axis equal tight;
xlabel('X [m]');
ylabel('Y [m]');
title(sprintf('Flat TPS Geometry (Clean) — %d Layers', num_layers));

% MESH + MATERIAL INDEX
figure('Name', 'Flat Mesh with Material Index', 'Color', 'w');
imagesc(x, y, material_id);
axis xy equal tight;
colormap(turbo(length(materials)));
colorbar;
xlabel('x [m]'); ylabel('y [m]');
title(sprintf('Flat TPS Mesh with Assigned Materials (%d Layers)', num_layers));

% Draw visible mesh grid
hold on;
for i = 1:length(x)
    plot([x(i), x(i)], [y(1), y(end)], 'k:', 'LineWidth', 0.25);
end
for j = 1:length(y)
    plot([x(1), x(end)], [y(j), y(j)], 'k:', 'LineWidth', 0.25);
end

% Output Struct
TPS.k = k_map;
TPS.Cp = Cp_map;
TPS.rho = rho_map;
```

```
TPS.material_index = material_id;
TPS.X = X;
TPS.Y = Y;
TPS.dx = dx;
TPS.dy = dy;
TPS.num_layers = num_layers;
TPS.layer_thickness = layer_height;
TPS.material_names = materials(material_selection);

end
```

Appendix E: Angled Geometry Generator Function

```
function TPS = generate_angled_TPS(angle_deg)

clc;
clear;
close all;

% User input
num_layers = input('Enter number of layers: ');
layer_thickness = input('Enter layer thickness [m]: ');
mesh_size = input('Enter desired mesh resolution [m]: ');

% Material database
materials = {'C/C', 'C/SiC', 'SiC/SiC', 'Saffil', 'Nextel'};
dens = [1750, 2100, 2980, 10, 96];     % [kg/m^3]
k = [20, 30, 180, 25.6, 0.112];        % [W/m-K]
Cp = [850, 800, 760, 1000, 1050];      % [J/kg-K]

fprintf('\nSelect a material for each layer (bottom to top):\n');
for i = 1:length(materials)
    fprintf('  %d: %s\n', i, materials{i});
end

material_selection = zeros(num_layers,1);
for i = 1:num_layers
    valid = false;
    while ~valid
        idx = input(sprintf('Layer %d material (1-%d): ', i, length(materials)));
        if idx >= 1 && idx <= length(materials)
            material_selection(i) = idx;
            valid = true;
        else
            fprintf('  Invalid selection. Try again.\n');
        end
    end
end

% DERIVED GEOMETRY
angle_rad = deg2rad(angle_deg);
Lx = layer_thickness / tan(angle_rad);   % Domain width
Ly = num_layers * layer_thickness;       % Total height
skyblue = [0.53, 0.81, 0.98];

% CLEAN GEOMETRY
figure('Name', 'Clean Geometry', 'Color', 'w');
hold on;

% Bottom triangle
fill([0, Lx, Lx], ...
     [0, 0, layer_thickness], ...
     skyblue, 'EdgeColor', 'k');

% Middle layers
for i = 2:(num_layers - 1)
```

```
    y0 = (i - 2) * layer_thickness;
    y1 = y0 + layer_thickness;

    fill([0, Lx, Lx, 0], ...
         [y0, y0 + layer_thickness, y1 + layer_thickness, y1], ...
         skyblue, 'EdgeColor', 'k');
end

% Top triangle
y_top = (num_layers - 1) * layer_thickness;
fill([0, 0, Lx], ...
     [y_top, y_top - layer_thickness, y_top], ...
     skyblue, 'EdgeColor', 'k');

axis equal tight;
xlabel('X [m]');
ylabel('Y [m]');
title(sprintf('Angled TPS Geometry (Clean) — %d Layers @ %.2f°', num_layers,
angle_deg));

% MESH + MATERIAL ASSIGNMENT
dx = mesh_size; dy = mesh_size;
x = dx/2 : dx : Lx - dx/2;
y = dy/2 : dy : Ly - layer_thickness - dy/2;
[Xc, Yc] = meshgrid(x, y);  % cell centers

% Initialize maps
mat_id_map = zeros(size(Xc));
k_map = zeros(size(Xc));
Cp_map = zeros(size(Xc));
rho_map = zeros(size(Xc));

% Assign each cell to correct layer
for row = 1:size(Yc,1)
    for col = 1:size(Xc,2)
        xc = Xc(row, col);
        yc = Yc(row, col);

        sloped_offset = (xc / Lx) * layer_thickness;
        adjusted_y = yc - sloped_offset;

        layer_idx = ceil(adjusted_y / layer_thickness) + 1;

        if layer_idx >= 1 && layer_idx <= num_layers
            mat_idx = material_selection(layer_idx);
            mat_id_map(row, col) = mat_idx;
            k_map(row, col)   = k(mat_idx);
            Cp_map(row, col)  = Cp(mat_idx);
            rho_map(row, col) = dens(mat_idx);
        end
    end
end

% Display Mesh View
figure('Name', 'Mesh View with Material Index', 'Color', 'w');
```

```
imagesc(x, y, mat_id_map);
axis xy equal tight;
colormap(turbo(length(materials)));
colorbar;
hold on;

for i = 1:length(x)
    plot([x(i), x(i)], [y(1), y(end)], 'k:', 'LineWidth', 0.25);
end
for j = 1:length(y)
    plot([x(1), x(end)], [y(j), y(j)], 'k:', 'LineWidth', 0.25);
end

xlabel('X [m]');
ylabel('Y [m]');
title(sprintf('Angled TPS Mesh with Assigned Materials (%d Layers)', num_layers));

% OUTPUT STRUCT
TPS.k = k_map;
TPS.Cp = Cp_map;
TPS.rho = rho_map;
TPS.material_index = mat_id_map;
TPS.X = Xc;
TPS.Y = Yc;
TPS.dx = dx;
TPS.dy = dy;
TPS.num_layers = num_layers;
TPS.layer_thickness = layer_thickness;
TPS.angle_deg = angle_deg;
TPS.material_names = materials(material_selection);

end
```