Friedrich-Alexander-Universität Erlangen-Nürnberg
Technische Fakultät

Lehrstuhl für Strömungsmechanik

Prof. Dr.-Ing. habil. Antonio Delgado

Optimization of a Solar Air Heater

Master's Thesis in Computational Engineering

Supervision: apl. Prof. Dr.-Ing. habil. Stefan Becker
Dr. Fabian Wein

Author: Bernhard Schlagbauer
E-Mail: bernhard.schlagbauer@fau.de
Matriculation number: 22732444

Submitted on: April 25, 2022

Statutory Declaration

I confirm that I have written this thesis unaided and without using sources other than those listed and that this thesis has never been submitted to another examination authority and accepted as part of an examination achievement, neither in this form nor in a similar form. All content that was taken from a third party either verbatim or in substance has been acknowledged as such.

Hannover, 22.04.2022

Bernhard Schlagbauer

Abstract

Solar air heaters are not commonly found on your average everyday building or house, but with sustainability and climate change becoming ever present and the focus point of many debates it is important that every solution is considered in our attempt to become more carbon neutral and sustainable.

In Chapter 1 the general background of the problem is stated and the motivation behind it. The state of the art is considered which then leads to setting objectives for the rest of the thesis. In Chapter 2 the base knowledge is discussed and all the physics involved in terms of heat transfer is considered. The basic functionality of the solar air heater is also discussed and different designs are showcase.

In Chapter 3 the simulation theory required to perform the flow and heat transfer simulations are discussed in more detail to give a better understanding of how these fields could be simulated. The Response Surface Method is also discussed and how it could be applied to the problem at hand and used to optimize the system. In Chapter 4 the radiation model that was implemented is discussed in more detail and how the absorber material is incorporated into the flow domain. A brief overview is also given for the algorithm used to bring all the different simulation and modeling techniques together.

The simulation and modeling techniques are validated in Chapter 5, looking at some different aspects of the simulations individually and insuring that the results are realistic and physical. A mesh independence study was also performed to ensure that the mesh used was fine enough for the simulations. In Chapter 6 the different parameters that affect the solar heater performance was investigated and a sensitivity analysis was done with RSM to understand the system and how it works better.

Finally in Chapter 7 after the system and all it's parameters where well understood optimization could be performed via an iterative approach using the RSM and Python's scipy.minimize package together. The final design is then compared to three other designs that where obtained during the course of this thesis and the improvements and results are described in more detail.

Contents

List of Figures

List of Tables

Abbreviations

BGK Bhatnagar-Gross-Kook

CCD Central Composite Design

CFS Coupled Field Simulation

DoE Design of Experiments

EU European Union

GCI Grid Convergence Index

LBM Latice Boltzmann Method

PV Photovoltaics

RSM Response Surface Methods

SUPG Streamline-Upwind Petrov-Galerkin

Chapter 1

Introduction

1.1 Motivation

With sustainability and climate change becoming ever more present and the focus point of many debates it is important that every solution is considered in our attempt to become more carbon neutral and sustainable. In 2019 households represented 26% of final energy consumption in the EU [10]. Of the total energy consumed by households room heating and heating applications have by far the most impact and is the most energy intensive at 71% of the total energy usage[11]. The breakdown of energy consumption per category can be seen in Figure 1.1 below.

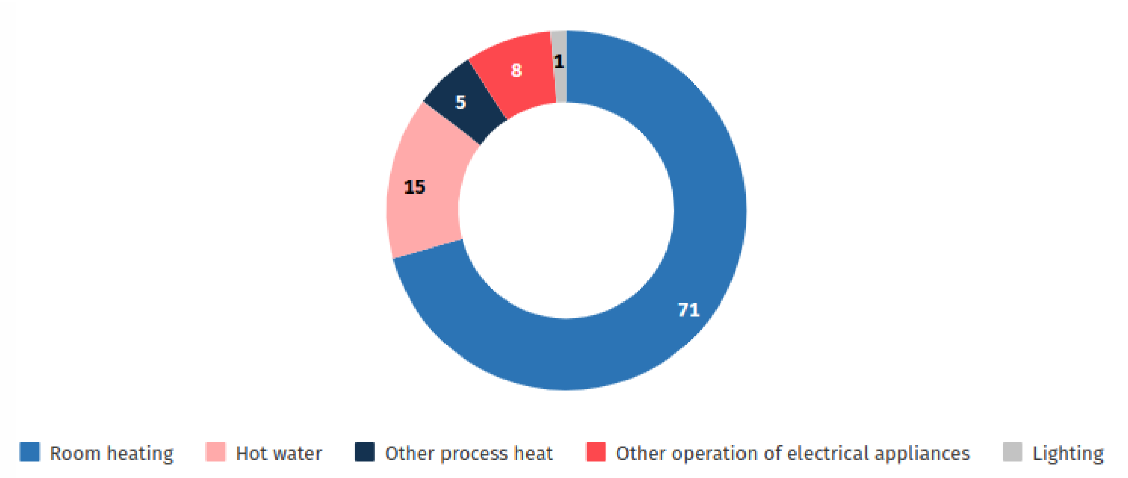

Figure 1.1: Energy Consumption for Housing in 2019. (source: [11])

From Figure 1.1 it is clear to see that heating our homes is the most energy intensive and that it is important to make this as renewable and sustainable as possible. If gas and electricity from non renewable sources are continued to be used for heating our homes and buildings it will be very hard to meet the EU's targets of becoming carbon neutral or reaching net zero by 2050[9].

This means that new an innovative solutions need to be found in order to combat climate change and to reduce the impact that household heating has on sustainability. One solution is to use renewable energy sources to generate electricity and use this in electric heating applications. However the EU has a predominantly gas dependant heating infrastructure and in general electricity is two to three times more expensive than gas so this has major economic and practical considerations to overcome[12].

Another solution is using solar energy directly and instead of converting it to electricity first via Photo-voltaic cells, use the radiation emitted by the sun and capture it in a solar air heater. The purpose of a solar air heater is to capture as much of the sun's solar radiation as possible, absorb the energy and transfer it to the air passing through it going into the building.

This means that in certain times of the year a portion of the heating requirements of a building or house could directly be fore filled by using the sun as a renewable energy source, thus decentralizing heating production to the source where it is required. This could have a dramatic impact on the overall heating requirements which in turn means more sustainable household heating requirements[28].

1.2 State of the Art

Before looking at the objectives of the thesis its worth looking into some previous work and projects to get a better understanding of what has been looked at and achieved within this field. Several designs including corrugated absorber surfaces an wire mesh packing was tested in [6] and at maximum an overall efficiency of 65% was obtained. Temperatures of around 65°C has been achieved by [8] resulting in an efficiency of 50%. Many patents have been published on the subject, many with interesting concepts such as turbulence jets and different glazing arrangements [38]. In [34] the possibility of passing air through double glazing as well was found to ad 10-15% to the solar heater efficiency. Using packing internally in the flow field also found some gains up to a thermal efficiency of 70% by [7].

Much has been done in the field in the past, but recently there is not much activity and active research in this topic. A good source of information can be found on "do it yourself" websites like 'Build it Solar' [3]. It is a community of people actively testing and sharing knowledge on the field of solar energy including solar heaters. It has physical experiments and use-cases and is an excellent source to draw knowledge from and take note of.

There is also very little information about shape optimization in this field as most optimization attempts focuses on materials and other design aspects such as glazing and multi-passes. It is thus interesting to investigate what performance gains could be achieved through a shape optimization approach.

1.3 Objectives

The main goal of this thesis is to come up with an optimized solar heater design. First it is required to investigate and understand different solar heater designs and identify which solar heater is best. Secondly a multi physics simulation need to be perform to asses the performance of the solar heater. The open source simulation tool that will be used in this thesis is openCFS. OpenCFS (Coupled Field Simulation) is a finite element-based multi-physics modelling and simulation tool [30].

The tool has the Lattice Boltzmann Method (LBM) and also the convection-diffusion heat equation implemented in it. The challenge will be to create a model to accurately simulate the radiational heat transfer from the sun to the system and to accurately simulate the heat transferred in the system as a whole.

When the model is created and justified the next objective would be to understand the sensitivity of the system regarding the absorber shape, different layer configurations and airflow through the system. Response Surface Methodologies will be implemented to create a response surface and understand the system as a whole better and which characteristics have the greatest impact.

Finally the response surfaces will be used to find a optimal solution with the given set of inputs, parameters and assumptions. By following this approach the aim in the end is to understand the most important parameters in terms of solar heater performance and efficiency and come up with a better design then the current benchmarks.

1.4 Outline

The second chapter starts with the fundamentals of heat transfer and an overview of solar heaters and the different heat transfer physics that are involved. Then in the third chapter the Lattice Boltzmann Method (LBM) as well as the convection-diffusion-reaction equation will be discussed and finally a theoretical overview will be given on Response Surface Methods (RSM) and how they are used to understand systems better as well as to optimize designs.

The fourth chapter describes the geometry of the solar heater in more detail and the solar radiation model and how it was incorporated into the simulation. Afterwards the algorithm and simulation overview is given to understand the steps taken and the code that was implemented to perform the multi-physics simulation.

In the fifth chapter, multi-physic simulations are performed to simulate the benchmark cases and to ensure that the simulation accurately represents the real world physics and to validate the simulation. After the simulations are validated a parameter study is done in chapter 6 and RSM performed to better understand the different interactions and sensitivity of the solar heater as a whole.

In chapter seven the acquired knowledge from the RSM will be used to then optimize the system within the given parameters and constraints. The results will be discussed and the design will be looked at in further details to compare and assess versus the benchmark case.

The final chapter is for final discussions and conclusions and recommendations, in terms of design as well as future work, is made based on the findings of this work.

Chapter 2

Solar Air Heater Fundamentals

In this chapter, the fundamentals and operation of solar air heaters are discussed theoretically to understand the concept behind them. The different heat transfer mechanisms are discussed and their applicability in terms of solar heaters. After creating a better understanding of the workings of a solar air heater the different solar heater concepts will be discussed and compared to give insight to which designs are best suited for the specific application.

Solar heating is a technology that uses solar thermal energy from the sun together with insulation and an absorbing medium to heat a substance. Since the focus in this thesis is on solar air heaters the substance would be air and any mention of solar heaters in the remainder of this thesis will refer specifically to air heaters.

2.1 Fundamentals of Heat Transfer

Heat is a form of energy that can be transferred from one system to another due to a difference in temperature. Heat energy can be transferred via three different mechanisms. These three mechanisms are conduction, convection and radiation and will be discussed in further detail in this section [44].

2.1.1 Conduction

Conduction is the energy transfer from particles which are more energetic to the adjacent less energetic particles due to their interactions with each-other.

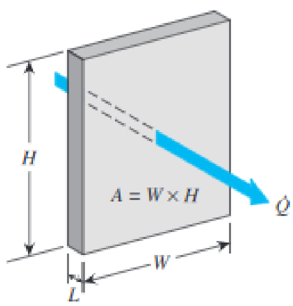

Figure 2.1: Basic Heat Conduction. (source: [44])

Conduction can occur in solids, liquids and gasses. In solids transfer is due to vibration and transport of free electrons. In liquids and gasses it is down to collisions and diffusion. The rate of heat conduction through a medium can be written as

$$\dot{Q}_{\text{cond}} = -kA\frac{dT}{dx}. \qquad [W] \tag{2.1}$$

Here k represents the thermal conductivity of the medium and is a material property of the medium describing it's ability to conduct heat. The temperature gradient is denoted as $\frac{\partial T}{\partial x}$ and this describes simply the change in temperature with the change in distance [44].

2.1.2 Convection

Convectional heat transfer occurs between a solid and the adjacent liquid or gas that is in motion. It is a combination of conduction and fluid motion that leads to convective heat transfer. The more motion and mixing leads to higher convective heat transfer rates.

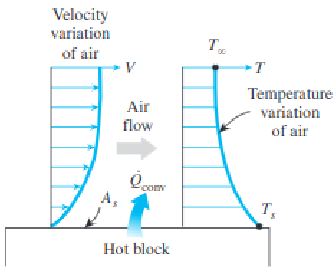

Figure 2.2: Basic Heat Convection. (source: [44])

Convective heat transfer depends on many factors such as the fluid properties, but it also depends on the the roughness and geometry of the surface with which it is in contact with. The rate of heat transfer due to convection can be described as

$$\dot{Q}_{\text{conv}} = hA_s\left(T_s - T_\infty\right). \qquad [W] \tag{2.2}$$

In this equation h is the convectional heat transfer coefficient and is in W/m^2K. A$_s$ is the surface area the fluid is in contact with, T_s the surface temperature and T_∞ the bulk fluid temperature [44].

2.1.3 Radiation

The third and final mechanism is radiation. Radiation is electromagnetic waves that are emitted by all matter. Unlike conduction and convection no medium is required for the energy to be transferred. Radiation is how energy from the sun reaches earth and in the case of a solar air heater, the main focus is on solar radiation and the mechanisms behind it. It is thus important to understand exactly the physics behind solar radiation so that this can later be accurately modelled and implemented into the simulation.

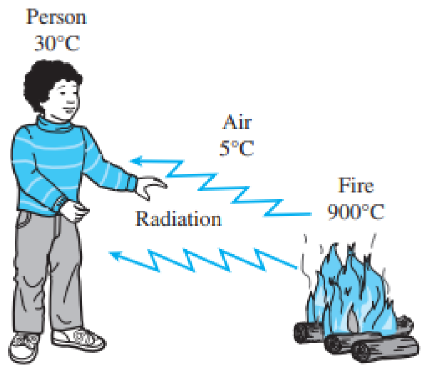

Figure 2.3: Basic Heat Radiation. (source: [44])

Since the sun is the source for radiation here the focus mainly lies with incident radiation. In general all surfaces emit radiation and receive radiation that is reflected or emitted from other surfaces. This can make radiation quite complex, but in general at low temperatures emission and reflection between surfaces are ignored. Bodies only start producing a noticeable amount of radiation at 800k [44]. Since the temperatures of the solar heater is fairly low and compared to the radiation of the sun negligible, there is no need to consider the radiation emitting from the solar heater surfaces itself.

For a surface the radiation flux incident from all directions is **irradiation** G and is given by

$$G = \int_{\phi=0}^{2\pi} \int_{\theta=0}^{\pi/2} I_i(\theta, \phi) \cos\theta \sin\theta \mathrm{d}\theta \mathrm{d}\phi. \qquad [W/m^2] \tag{2.3}$$

Here $I_i(\theta, \phi)$ is the intensity and is defined as the rate at which radiation energy dG is incident from the direction (θ, ϕ) per unit area. With θ the angle between normal of the surface and incident radiation [44].

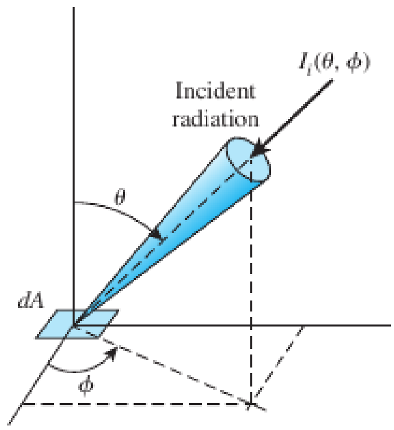

Figure 2.4: Incident Radiation Visualized. (source: [44])

The total solar irradiance G_s is the solar energy that reaches the earth's atmosphere it is also called the solar constant and is given as

$$G_s = 1373 W/m^2. \tag{2.4}$$

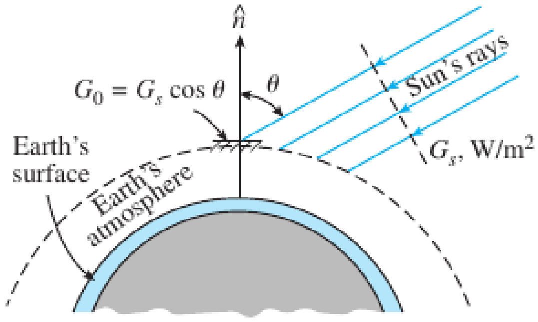

Figure 2.5: Solar Radiation on Earth. (source: [44])

The irradiance on the surface of the earth however differs dramatically according to climate, and position relative to the sun. The energy incident on the earths surface is considered in 2 parts, one the direct radiation and the second diffuse radiation. Direct radiation is the part that is not scattered or deflected/reflected by the earth's atmosphere. Diffuse radiation is the deflected/reflected radiation and is assumed to be uniform from all direction. Thus, the total solar energy incident on a unit surface can be written as

$$G_{\text{solar}} = G_D \cos\theta + G_d. \qquad [W/m^2] \tag{2.5}$$

With G_D the direct radiation and G_d the diffuse radiation. The angle the sun makes with the normal of the surface is θ.

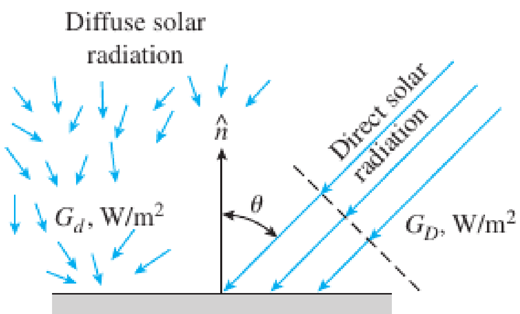

Figure 2.6: Direct and Diffuse Radiation. (source: [44])

The most important properties regarding radiation in this thesis is the radiation properties of the different materials and their effect on how the radiation is modelled. When solar radiation hits a surface there are 3 mechanisms at work. A part of the radiation is absorbed, some of it is reflected and in the case of transparent materials transmitted[44].

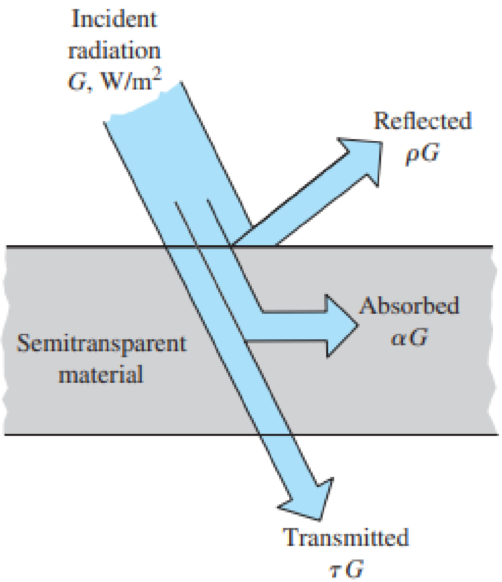

Figure 2.7: Absorptivity, Reflectivity and Transmissivity. (source: [44])

These properties are denoted by a fraction of the total irradiation incident on the material. The fraction that gets absorbed is **absorptivity** α, that gets reflected **reflectivity** ρ and that gets transmitted **transmissivity** τ. These ratio's can be calculated as

$$\begin{aligned}
\alpha &= \frac{\text{Absorbed radiation}}{\text{Incident radiation}} = \frac{G_{\text{abs}}}{G}, \quad 0 \leq \alpha \leq 1 \\
\rho &= \frac{\text{Reflected radiation}}{\text{Incident radiation}} = \frac{G_{\text{ref}}}{G}, \quad 0 \leq \rho \leq 1 \\
\tau &= \frac{\text{Transmitted radiation}}{\text{Incident radiation}} = \frac{G_{\text{tr}}}{G}, \quad 0 \leq \tau \leq 1.
\end{aligned} \tag{2.6}$$

It is important to note that,

$$\alpha + \rho + \tau = 1 \tag{2.7}$$

In some cases such as solid materials there is no **transmissivity** τ and thus by knowing the one property the other can be calculated, since if 1, 2 or all 3 of the mechanisms are involved the total remains = 1. This is very useful when creating the radiation model and trying to predict how much of the energy will be absorbed, reflected and transmitted.

2.2 Solar Air Heater Basics

Now that the basic heat transfer mechanisms are known it is important to look at the solar heater as a whole and understand which heat transfer mechanisms play a role in which part of the solar heater. It is also important to understand how the solar heater is connected to the building or system as a whole and how it can be used to heat the air in a room.

2.2.1 Basic Heat Transfer Concept

As mentioned before in basic terms a solar heater consist of an absorber material and some insulated enclosure. In Figure 2.1 the basic concept behind a solar heater can be seen.

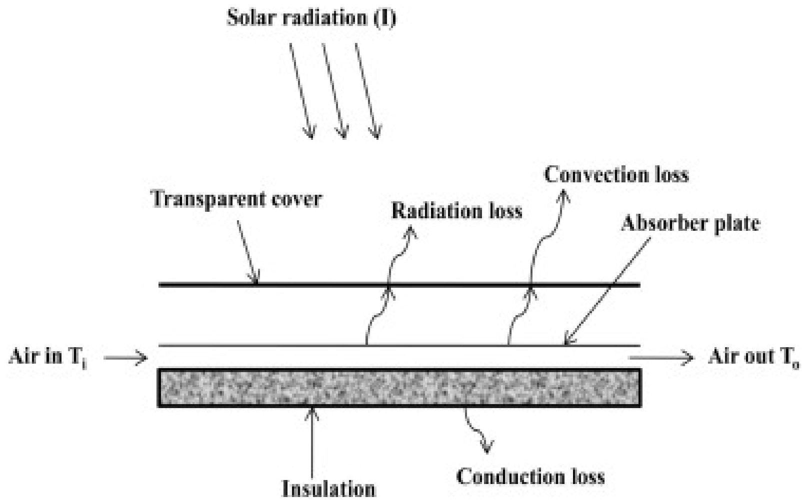

Figure 2.8: Basic Solar Heater Concept. (source: [24])

Solar radiation is allowed to pass through a transparent cover layer. This can be anything from normal glass, perspex to specifically purpose designed glass that allows for less losses and better heat retention. After passing through the transparent cover the solar radiation reaches the absorber material. The purpose of the absorber is to absorb as much energy as possible, thus it needs to have a high absorbtivity coefficient. The closer to a perfect black body one can get the better[28].

The cooler air from the outside then passes through the solar heater and passes over the heated absorber. This interaction causes heat energy to transfer from the plate to the air by convective heat transfer as discussed previously in 2.1.1. The air thus gets relatively warmer and then exits the solar heater at a higher temperature. There are conductive and convective losses in the system via the window and sides of the solar heater, and thus this needs to mitigated as much as possible through the use of insulation on the back and sides and concepts like double glazed front panels can be used to insulate the system more effectively[38].

2.2.2 System Integration

Now that the fundamentals of the solar heater is clear it is important also to understand how it can be incorporated to heat the air in a room and how it can reduce energy consumption.

In winter when it is cold outside internal heating is essential. Heat losses from a house or building to the environment is inevitable despite insulation of walls and windows, due to the physics behind heat transfer. It is thus necessary to continuously heat an area in order to maintain a comfortable temperature. This is most generally done by warm water radiators or electric heaters[28]. As mentioned before however, it is desirable to use renewable energy instead and in this case the solar energy from the sun.

The way the collector is mounted on a house and how the air flows into the house can be seen in Figure 2.9 below.

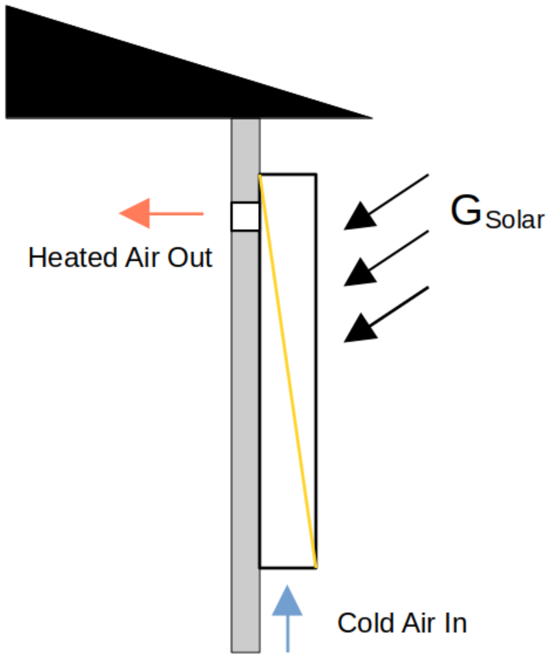

Figure 2.9: Concept of Solar Heater on House Wall.

Here it can be seen how the solar radiation G_{solar} hits the solar collector and warms up the absorber material. Cold air enters through the holes in the bottom and flows through or over the absorber. There it gets heated via convection and ultimately goes through a pipe in the wall and into the house.

There are many different designs and methods that the system can be installed in. In the example depicted fresh outside air is used, but air from inside can also be used to recirculate the already room temperature air and heat it further to maintain the preferred temperature levels inside. The system typically works with a fan to suck the air through the solar collector. This flow rate can also be monitored and changed according to heating requirements[38].

2.3 Different Solar Heater Concepts

As mentioned there are many different designs and this section aims to give a quick and brief overview of some of the designs that people use and comment on them. The designs where investigated on Build It Solar [3] and also experimentally tested to determine which design is the best in terms of performance, cost and ease of build.

Empty Box

As the name suggests this solar heater is purely just an empty box with a screen on the front and the back panel of the box acting as the absorber. The air flows through the box and gets heated via the back panel [17].

Figure 2.10: Empty Box Solar Collector. (source: [17])

Corrugated Plate

This collector uses an aluminium plate with folds and holes in it so that the air can pass over and through it. The plate gets heated by the solar radiation and transfers it to the air flowing over and through it [4].

Figure 2.11: Corrugated Solar Collector. (source: [4])

Screen Collector

The screen collector uses layers of aluminium mesh material. The mesh absorbs the solar radiation the air then passes through and mixes and heats up [5].

Figure 2.12: Screen Solar Collector. (source: [5])

Backpass Collector
This collector uses a sheet of aluminium with some folds in it. The air flows over the front and back of it heating along the way [32].

Figure 2.13: Backpass Collector. (source: [32])

Manifold Collector
The collector uses circular like tubes or manifolds. The tubes are heated by the solar radiation and the air flows through them on the inside and gets heated [15].

Figure 2.14: Manifold Solar Collector. (source: [15])

When comparing the solar collectors the screen solar collector has the best performance of all of them [37]. The screen absorber also has low pressure drop over the system and thus is the preferred solar heater to optimize and the topic of this thesis. The results of their investigations can be seen summarized bellow.

Heater	Thermal Performance	Pressure Drop	Cost
Empty Box	5	1	1
Corrugated Plate	2	3	4
Screen Collector	1	2	2
Backpass Collector	3	5	3
Manifold Collector	4	4	5

Table 2.1: Relative Rankings of Different Solar Heater Designs.

Chapter 3

Simulation Theory

In order to understand the solar heater better as a system and as individual aspects, simulation is required. It is thus important to understand the theory behind the simulation techniques and how they are used to simulate the multi-physics problem. First there will be looked at the Lattice Boltzmann Method in more detail and then the convection-diffusion-response equations. These are important as the flow field will be solved using the LBM method and the heat transfer by the convection diffusion equation. Furthermore theoretical background and insight will be given regarding RSM and how it could be used to understand the system better as well as optimize the system.

3.1 Lattice Boltzmann Method

In this section the purpose is to create a basic understanding of the Lattice Boltzmann Method (LBM) and the underlying physical models for simulating fluid flow. A good understanding of the LBM is required as this method will be used to simulate the airflow through the solar heater.

3.1.1 Fluid Dynamics Characteristic Numbers

Characteristic numbers in fluid dynamics are dimensionless numbers used to describe the character of the flow. For the purpose of LBM it is important to discuss three such numbers in further details. These numbers are important as they form the connection between the simulation and conversion to the real physical world.

Reynolds Number

The Reynolds number (Re) describes the ratio between inertial and viscous forces in a flow. The Re number also gives an indication in terms of the type of flow regime and if the flow is laminar, transitional or fully turbulent [45]. The Re number can be described as follows,

$$Re = \frac{\rho v L}{\mu} = \frac{\text{inertial forces}}{\text{viscous forces}} \tag{3.1}$$

where ρ is the fluid density, v denotes the bulk flow fluid velocity, L is the characteristic length of the domain or geometry and μ is the dynamic viscosity. The Re number is very important in LBM simulation as it needs to be assured that the Re number in terms of lattice units is equal to the Re number in real units[29].

Mach Number

The Mach number (Ma) is an indication of the compressibility of a fluid and describes the relation between inertial and elastic forces. The Ma number is given as,

$$Ma = \frac{U}{c_s} \tag{3.2}$$

here U is the fluid velocity in m/s and c_s is the speed of sound in that specific medium. Generally for Ma numbers lower than 0.3 the flow can be seen as incompressible and for Ma number more than 0.3 the compressibility affects start to have a significant impact on the flow regime[45].

Knudsen Number

The ratio of Ma number to Re number is proportional to the Knudsen number (Kn)

$$Kn \sim \frac{Ma}{Re}. \tag{3.3}$$

The Kn number is the ratio of mean free path length to the characteristic length and is given by,

$$Kn = \frac{\lambda}{L}. \tag{3.4}$$

The Kn number helps in determining if statistical or continuum mechanics need to be used to model a fluid dynamic problem[42].

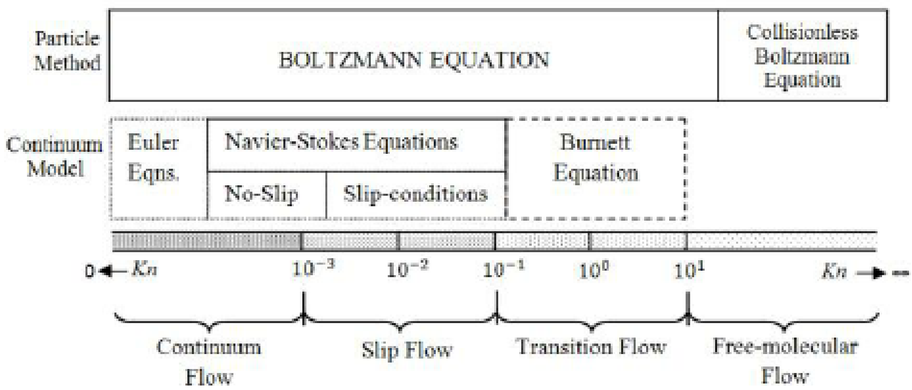

Figure 3.1: Flow simulation range based on Knudsen number. (source:[26])

In figure it can be seen that the LBM has a very wide range in terms of application and that it will also be well suited for the simulation of flow through the solar heater.

3.1.2 Lattice Boltzmann Equation

The Lattice Boltzmann Method originates from Ludwig Boltzmann's kinetic theory of gases. The basic idea behind the LBM is that fluids and gases can be imagined as consisting of small particles moving with random motions [43]. By means of particle streaming and particle collisions the exchange of momentum and energy is achieved. This process of streaming and collisions can be described/modelled by the Boltzmann transport equation given as [29]

$$\frac{\partial f}{\partial t} + \vec{u} \cdot \nabla f = \Omega. \tag{3.5}$$

Here $\frac{\partial f}{\partial t}$ is the function describing the particle distribution, $\vec{u}$ is the velocity of the particle and Ω is the collision operator. The LBM is a simplification of Boltzmann's original gas dynamics idea in the sense that it reduces the amount of particles in the sense of grouping them together and also confining them to a lattice with specific nodes. For example a very common case is to restrict the streaming of the particles in 8 possible direction plus one stationary position. These velocities are commonly referred to as the microscopic velocities[29]. The model just described is known as the D2Q9 model as it is applied in two dimensions and has 9 distinct velocity vectors. These velocities are denoted by $\vec{e_i}$ and is defined as

$$\vec{e_i} = \begin{cases} (0,0) & i = 0 \\ (1,0),(0,1),(-1,0),(0,-1) & i = 1,2,3,4\ . \\ (1,1),(-1,1),(-1,-1),(1,-1) & i = 5,6,7,8 \end{cases} \tag{3.6}$$

These vectors represent the streaming directions and the lattice representation of the D2Q9 model can be seen below[29]:

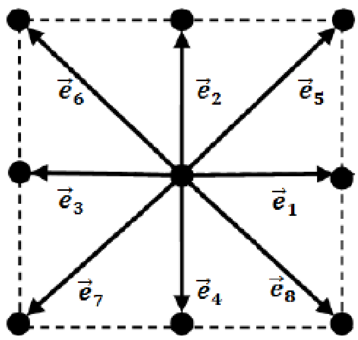

Figure 3.2: Representation of the D2Q9 model lattice. (source:[43])

The density of the macroscopic fluid is defined as the summation of microscopic particle distribution function

$$\rho(\vec{x},t) = \sum_{i=0}^{8} f_i(\vec{x},t). \tag{3.7}$$

Also, the macroscopic fluid velocity $\vec{u}(\vec{x},t)$ is the average of $\vec{e_i}$ weighted by the distribution functions f_i

$$\vec{u}(\vec{x},t) = \frac{1}{\rho}\sum_{i=0}^{8} cf_i\vec{e}_i. \tag{3.8}$$

The steps that are key in LBM are streaming and collision and these processes are given as

$$\underbrace{f_i\left(\vec{x}+c\vec{e}_i\Delta t, t+\Delta t\right) - f_i(\vec{x},t)}_{\text{Streaming}} = -\underbrace{\frac{[f_i(\vec{x},t) - f_i^{eq}(\vec{x},t)]}{\tau}}_{\text{Collision}} \tag{3.9}$$

When considering the actual implementation of the LBM, the streaming and collision are computed separately. Thus special attention needs to be given when considering the nodes on the boundary of the Lattice [43].

The streaming step can be graphically represented for the interior nodes as:

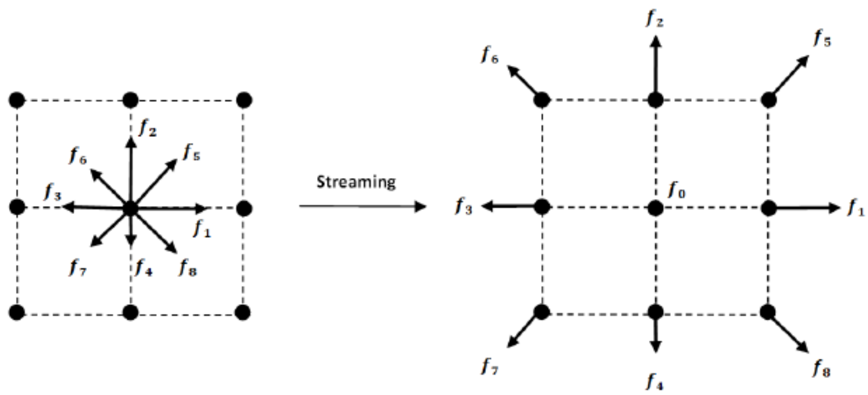

Figure 3.3: Representation of the D2Q9 model lattice. (source:[43])

Regarding the collision term in (3.9), $f_i^{eq}(\vec{x},t)$ represents the equilibrium distribution and τ is the relaxation time towards local equilibrium[21]. The Bhatnagar-Gross-Krook (BGK) collision is sufficient to be used in simulating single phase flows. The equilibrium distribution $f_i^{eq}(\vec{x},t)$ under the BGK method is defined as

$$f_i^{eq}(\vec{x},t) = w_i\rho + \rho s_i(\vec{u}(\vec{x},t)) \tag{3.10}$$

with $s_i(\vec{u})$ defined as

$$s_i(\vec{u}) = w_i\left[3\frac{\vec{e}_i\cdot\vec{u}}{c} + \frac{9}{2}\frac{(\vec{e}_i\cdot\vec{u})^2}{c^2} - \frac{3}{2}\frac{\vec{u}\cdot\vec{u}}{c^2}\right] \tag{3.11}$$

and the weights w_i as

$$w_i = \begin{cases} 4/9 & i = 0 \\ 1/9 & i = 1,2,3,4 \\ 1/36 & i = 5,6,7,8 \end{cases} \tag{3.12}$$

with lattice speed defined as $c = \frac{\Delta x}{\Delta t}$. In the D2Q9 model the kinematic viscosity of the fluid ν is related with the relaxation time τ as follows

$$\nu = \frac{2\tau - 1}{6}\frac{(\Delta x)^2}{\Delta t}. \tag{3.13}$$

To give a better understanding of how all of this could be implemented for solving fluid flow problems the algorithm can be summarized as follows[43]:

1. $\rho, \vec{u}, f_i$ and f_i^{eq} should be initialized
2. Step for Streaming: move $f_i \longrightarrow f_i^*$ in the direction of $\vec{e_i}$
3. Calculate macroscopic ρ and $\vec{u}$ from f_i^* by using (3.7) and (3.8)
4. Calculate $f_i^e q$ by using (3.10)
5. Step for Collision: compute the updated distribution function
   $f_i = f_i^*$ - $\frac{1}{\tau}(f_i^*$ - $f_i^e q)$ using (3.9)
6. Repeat the steps 2 until 5 until convergence.

It is important to note that numerical issues can arise as $\tau \longrightarrow 1/2$. During the steps for streaming and collision, the nodes on the boundaries require special treatment on the distribution functions to satisfy the macroscopic boundary conditions that are imposed and will be discussed in more detail in the next section.

3.1.3 Boundary Conditions for BGK Method

There are many different boundary conditions presented in [29], but for the purpose of this thesis and not to elaborate to much, there will only be focused on the boundary conditions applicable to what is needed to simulate.The boundary conditions that will be described in this section are:

1. *No-Slip* boundary conditions, which are important at the solid surfaces of the wall. No-Slip implies that the flow in contact with the wall has zero velocity.
2. *In- and Outflow* boundary conditions in order to simulate flow coming in and out of the system.

No-Slip BC's

To implement no-slip conditions at the boundaries the particles hitting the wall are reflected 180°, this is formulated by the so called bounce-back rule. This formulation is illustrated in Figure 3.4 can be described as follow. Fluid particles at a boundary are scattered back to the fluid along its incoming direction. The idea of an additional layer is introduced and thus the boundary is centered between these additional nodes and the first layer of fluid nodes[42].

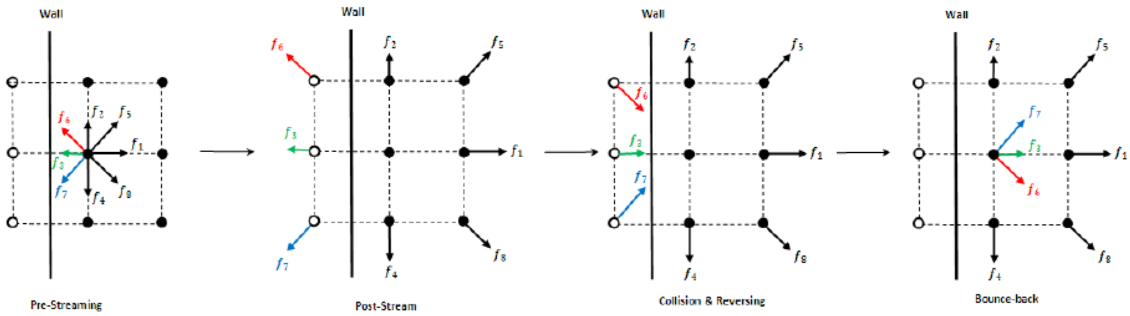

Figure 3.4: Mid-Grid Bounce-Back Illustration. (source:[43])

Pre-Streaming the distribution functions pointing outside the domain would leave it, these are streamed to the additional nodes and stored as seen in the post-streaming. During the collision step the collision is ignored for these nodes and the distribution functions are simply reversed. The values are then propagated back into the domain in the bounce-back step. This approach is more accurate than a node based approach which is only first order accurate. The method is mostly accurate for straight boundaries and is of second order accuracy [43].

Inflow and Outflow BC's

In many physical cases it is preferable to assign velocity or pressure at a boundary. There are many implementations/interpretations of inflow and outflow boundary conditions, but for the purpose of this thesis the description will be brief and based on what has been implemented into openCFS. The inflow and outflow conditions are addressed similarly to [20]. Here a equilibrium distribution velocity inlet condition was used and a non-equilibrium density boundary condition was used. The macroscopic inlet velocity at the inlet nodes $\vec{u_{in}}$ is assigned and the collision step (3.10) is replaced with

$$\tilde{f}_i(\vec{x},t)_{\text{in}} = f_i^{eq}(\rho,\vec{u}_{\text{in}}) = \rho w_i \left[1 + 3(\vec{e}_i \cdot \vec{u}_{\text{in}}) + \frac{9}{2}(\vec{e}_i \cdot \vec{u}_{\text{in}})^2 - \frac{3}{2}(\vec{u}_{\text{in}} \cdot \vec{u}_{\text{in}})\right]. \quad (3.14)$$

The same approach can be followed and the macroscopic outlet density ρ_{out} assigned to the outlet nodes with a modified collision step as

$$\tilde{f}_i(\vec{x},t)_{\text{out}} = f_i^{eq}(\rho_{\text{out}},\vec{u}) = \rho_{\text{out}}\, w_i \left[1 + 3(\vec{e}_i \cdot \vec{u}) + \frac{9}{2}(\vec{e}_i \cdot \vec{u})^2 - \frac{3}{2}(\vec{u} \cdot \vec{u})\right]. \quad (3.15)$$

3.1.4 Porosity Model

For the purpose of this thesis it is important to be able to simulate porous media. The absorber material in the solar heater domain acts as a restriction to the flow and this is simulated via the porous media approach. The porosity model introduced by [27] poses a modification to the standard BGK method in order to model flow in heterogeneous porous media. This is said to be the same as solving the Brinkman equation

$$\mu_e \nabla^2 \langle \boldsymbol{u} \rangle - \mu \boldsymbol{K}^{-1} \langle \boldsymbol{u} \rangle = \nabla \langle p \rangle. \tag{3.16}$$

Here $\langle p \rangle$ and $\langle \boldsymbol{u} \rangle$ represents the volume average pressure and velocity. The effective viscosity inside the porous media is μ_e and $\boldsymbol{K}$ is the permeability tensor. The idea behind it is depending on the porosity at the lattice node to modify the velocity term in the equilibrium distribution function $f_i^{eq}\left(\rho(\vec{x},t), \vec{u}(\vec{x},t)\right)$.
In the case of the model the inverse porosity $d(x)$ is used with $0 \leq d(x) \leq 1$ and a inverse porosity of 1 indicating pure solid and 0 pure fluid. The average velocity of the fluid at the porous node can be computed as

$$\tilde{u} = \left(1 - d(\vec{x})^{\kappa}\right) \vec{u}(\vec{x},t) \tag{3.17}$$

with κ a penalty or so called shaping factor.

3.2 Convection Diffusion

Simulation of the convection-diffusion-reaction is required and useful in many applications. Typical examples include time dependant chemical reactions or steady state heat transport [40]. For the purpose of this thesis there will be focused on convection dominated scalar equations for incompressible flow problems. The Streamline-Upwind Petrov-Galerkin method (SUPG) is also discussed in more detail in this section as this is a major starting point for this type of problems[41].

3.2.1 Scalar Convection-Diffusion-Reaction Equations

General Equation

Let $\Omega \subset \mathbb{R}^d$, $d \in 2, 3$, be a domain and $T > 0$ be the final time. The assumption is made that the boundary is polyhedral and that Ω is a Lipschitz domain. The scalar convection-diffusion-reaction equation can then be written as

$$\partial_t u - \varepsilon \Delta u + \boldsymbol{b} \cdot \nabla u + cu = f \quad \text{in } (0,T] \times \Omega. \tag{3.18}$$

with $\boldsymbol{b}$ as the convective field, ε is the constant diffusion coefficient and c ≥ 0 is a scalar function describing reactions.

Appropriate boundary conditions and initial conditions needs to be assigned in order to obtain a well-posed problem. The equation model the behavior of scalar quantities as previously mentioned like temperature or concentration. These quantities are transported in the flow field with velocity $\boldsymbol{b}$ (convection), they undergo molecular transport(diffusion) and also have some interaction between them (reaction), hence the term convection-diffusion-reaction equation. When incompressible flow is considered it is important to note that $\nabla \cdot \boldsymbol{b} = 0$ [41].

Steady State Equation

When considering the performance of the solar heater it is more important to understand its steady state performance rather than the instantaneous differences in temperature. It is thus important to look in to the steady state convection-diffusion-reaction equation in more detail to understand the fundamental theory behind simulating the temperature scalar field.

We have the same assumptions as before, let $\Omega \subset \mathbb{R}^d$, $d \in 2, 3$, be a domain and $T > 0$ be the final time. The assumption is made that the boundary is polyhedral and that Ω is a Lipschitz domain. The steady state scalar convection-diffusion-reaction equation can then be written as

$$-\varepsilon\Delta u + \boldsymbol{b} \cdot \nabla u + cu = f \quad \text{in } (0, T] \times \Omega \tag{3.19}$$

with $-\varepsilon\Delta u$ as the diffusion term, $\boldsymbol{b} \cdot \nabla u$ the convection term and cu the reaction term [13].

With $(\cdot,\cdot)$ the inner product of $L^2(\Omega)$, Consider (3.19) and multiply the equation with an appropriate function $v(x)$ with $v = 0$ on $\partial\Omega$, integrating the resulting equation on Ω and by applying integration by parts the weak solution of (3.19) can be shown as [13]

$$\begin{aligned}
&\int_\Omega (-\varepsilon\Delta u + \mathbf{b} \cdot \nabla u + cu)(\mathbf{x})v(\mathbf{x})d\mathbf{x} \\
&= \int_{\partial\Omega} (-\varepsilon(\nabla u \cdot \mathbf{n})v(\mathbf{s}))d\mathbf{s} + \int_\Omega (\varepsilon\nabla u \cdot \nabla v + (\mathbf{b} \cdot \nabla u + cu)(\mathbf{x})d\mathbf{x} \\
&= \int_\Omega (\varepsilon\nabla u \cdot \nabla v + (\mathbf{b} \cdot \nabla u + cu)v(\mathbf{x})d\mathbf{x} \\
&= \int_\Omega (f(\mathbf{x})v(\mathbf{x}))d\mathbf{x}.
\end{aligned} \tag{3.20}$$

With $\mathbf{n}$ as the outward pointing unit normal vector on $\partial\Omega$ and the highest order derivatives of $u(\mathbf{x})$ has been transferred to $v(\mathbf{x})$. Now if $b, c \in L^\infty(\Omega)$ and $f \in H^{-1}(\Omega)$, the convection-diffusion-reaction equation in (3.19) can be written in weak form as: Find $u \in H_0^1(\Omega)$ such that for all $v \in H_0^1(\Omega)$

$$\varepsilon(\nabla u, \nabla v) + (\mathbf{b} \cdot \nabla u + cu, v) = (f, v). \tag{3.21}$$

The solution of (3.21) is the weak solution, the space where the solution is searched is called the ansatz space and $v(\mathbf{x})$ are called test functions and they come from the spaces called test spaces. the solution space and test space are both $H_0^1(\Omega)$ [13].

Streamline-Upwind Petrov-Galerkin method

The part that is of most interest in terms of this thesis is in applications where convectional affects dominate and are much bigger than diffusion, $||\mathbf{b}||_{L^\infty(\Omega)} \gg \varepsilon$. As described in [41], a characteristic feature of the solutions of (3.18) and (3.19) is that there is an appearance of layers. These layers are regions in the domain where the gradient of the solution is very high and depending if it is exponential or characteristic, the thickness of layers are $\mathcal{O}(\epsilon)$ or $\mathcal{O}(\sqrt{\epsilon})$. Generally it holds that $\sqrt{\epsilon} \ll h$, with h as the mesh size and means that the layers are not possible to resolve on the mesh. This issue causes the failure of standard discretization methods like the Galerkin finite element method.

It is thus important to look at the Streamline-Upwind Petrov-Galerkin (SUPG) method using so called stabilized discretizations. The Streamline-Upwind Petrov-Galerkin method has the form: Find $u^h \in V^h$, such that

$$a^h\left(u^h, v^h\right) = f\left(v^h\right) \forall v^h \in V^h \tag{3.22}$$

with,

$$a^h(v, w) := a(v, w) \tag{3.23}$$

$$+ \sum_{K \in \mathcal{T}^h} \int_K \delta_K(-\varepsilon \Delta v(\mathbf{x}) + \mathbf{b}(\mathbf{x}) \cdot \nabla v(\mathbf{x}) + c(\mathbf{x}) v(\mathbf{x}))(\mathbf{b}(\mathbf{x}) \cdot \nabla w(\mathbf{x})) d\mathbf{x} \tag{3.24}$$

$$f^h(w) := (f, w) + \sum_{K \in \mathcal{T}^h} \int_K \delta_K f(\mathbf{x})(\mathbf{b}(\mathbf{x}) \cdot \nabla w(\mathbf{x})) d\mathbf{x}. \tag{3.25}$$

With δ_K weights chosen by the user and are the so called stabalization parameters or SUPG parameters.

The SUPG method does introduce some artificial diffusion, but only in the streamline direction $\mathbf{b}(\mathbf{x}) \cdot \nabla w(\mathbf{x}$. Also the SUPG is consistent and the stabilization parameter is chosen as a constant function in practice so a good asymptotic choice of it is proposed.

3.3 Response Surface Methods

RSM is very useful for improving, optimizing and developing new processes or products and is based on a statistical and mathematical approach. RSM is most extensively used in the industrial world and is particularly useful when various input variables influences the characteristics or performance measures of a process or product [2]. The characteristic or performance is referred to as the response and is typically on the continuous scale. The input parameters are the independent variables and are mostly chosen by the engineer or just given by some constrains to the physical problem.

3.3.1 Uses for RSM

RSM is very useful as it can be used in such a broad spectrum of problems, generally the problems fall into three categories [2]:

1) **Mapping over particular region:**
From well fitted response surfaces, engineers can predict changes in advance.

2) **Optimization of the Response:**
By minimizing the obtained response surface a design or process can be optimized within the given set of constraints and parameters.

3) **Feature Selection:**
When certain features of the design needs to be changed or is pre-specified by the engineer the selected parameters can be kept constant and the optimal design can still be found by the response surface.

3.3.2 Approximating Response Functions

The idea is to find a continuous so called 'response surface' to accurately predict the outcome of the response given a certain set of parameters. This is achieved by building empirical models which then can be used to model the response surface. We suppose that we have a process or product which has a response y that is dependent on the input variables ξ_1, ξ_2,...ξ_k. The relationship is

$$y = f(\xi_1, \xi_2, ...\xi_k) + \epsilon \tag{3.26}$$

where the shape of the true response function f could be very complicated or unknown and ϵ is the term that represents error and variability that is not accounted for in f and is treated as a statistical error.

In RSM it is easier to work with normalized coded variables x than with the natural variables[2]. The scaling of the variables are done according to the chosen DoE method and is discussed later in more detail. In most cases a first-order or second-order model is used to describe the response surface. In terms of coded variables the first-order model of only two independent variables can be described by:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \tag{3.27}$$

The first-order independent variable model is very basic and struggle to predict very complex co-dependent systems. Thus in the rest of this section emphasis will be put on the dependent second-order model. The second-order model is more flexible and can more accurately represent physical relationships between parameters of the system. The second order model can be described as follow [2]:

$$y = \beta_0 + \sum_{j=1}^{k} \beta_j x_j + \sum_{j=1}^{k} \beta_{jj} x_j^2 + \sum \sum_{i<j=2}^{k} \beta_{ij} x_i x_j \tag{3.28}$$

Where the first term is a constant, the second term the linear relationship, the third term the quadratic relationship and the final term the interaction between input variables [2].

3.3.3 Emperical Models

The second order model above can be written in terms of matrix notation as:

$$y = X\beta + \epsilon \tag{3.29}$$

where,

$$y = \begin{bmatrix} y_0 \\ y_1 \\ \dots \\ y_n \end{bmatrix}, \qquad X = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1k} \\ 1 & x_{21} & x_{22} & \dots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{nk} \end{bmatrix}$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \qquad \epsilon = \begin{bmatrix} \epsilon_0 \\ \epsilon_1 \\ \vdots \\ \epsilon_k \end{bmatrix} \tag{3.30}$$

The least squares method is typically used to estimate regression coefficients in a multi-linear regression model such as in the case of our second-order model. These methods are well described in any mathematical text book and the derivation is outside the scope of this paper. The main idea is to find the values β_j which completes our model and we can simply input our parameters and find the response surface y. By neglecting ϵ and multiplying both sides of (3.29) with transpose of X (X') the least square estimator of β is [2]:

$$X'Xb = X'y \tag{3.31}$$

$$\beta = (X'X)^{-1}X'y \tag{3.32}$$

Now that we know we can estimate the model parameters we need to setup our coded variable matrix X. In the next section the method for constructing this matrix will be discussed in more detail.

3.4 Class of Central Composite Designs

The central composite designs (CCDs) was introduced by Box and Wilson (1951) and much of the CCDs motivation evolves from its sequential experimentation. It is a DoE approach which is well suited for second-order approximations and is one of the most popular choices when it comes to second-order design [1].

3.4.1 CCD Parameters

In order to construct our coded variable matrix we first need to normalise the physical input variables ξ. This can be done as follow:

$$x_{i1} = \frac{(\xi_{i1} - (max(\xi_{i1}) + min(\xi_{i1})/2)}{((max(\xi_{i1}) - min(\xi_{i1}))/2)} \tag{3.33}$$

This scheme is widely used in fitting linear regression models and results in coded variables of the engineers choosing.

With CCD the initial coded variable design point is taken as the zero point.The CCD contains an embedded factorial design with points at coded variable points -1 and 1. It is also augmented with a group of 'star points' relying on parameter α that allow estimation of curvature. The CCD approach always includes twice as many star points as factorials. The star points are representative of extremes (minimums and maximums) in terms of the design factors [2]. The basic CCD for a case of two input variables k= 2 and $\alpha = \sqrt{2}$ can be seen in Figure 3.5.

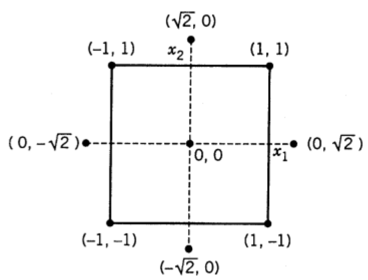

Figure 3.5: CCD coded variable domain.

The precise value of α depends on certain properties desired for the design and on the number of design variables involved. To ensure the correct choice of α and account for curvature and rotatibility of the design α can be calculated by [1]:

$$\alpha = \sqrt[4]{2^k} \tag{3.34}$$

3.4.2 Structure of Coded Variable Matrix

The α parameter together with the factorial points can now be used to construct the design matrix. An example of a design matrix that allows orthogonality and rotatability simultaneously for a case with two design parameters (k=2) is:

$$D = \begin{bmatrix} -1 & 1 \\ 1 & -1 \\ -1 & 1 \\ 1 & 1 \\ 0 & 0 \\ 0 & 0 \\ -\alpha & 0 \\ \alpha & 0 \\ 0 & -\alpha \\ 0 & \alpha \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.35}$$

By looking at Figure 3.5 it is clear to see where the design matrix structure comes from. It include the four factorial points as well as all combinations of the extreme points α. The design matrix also includes four center runs from point (0, 0). The factorial points are the only terms that has a contribution in terms of estimating the interaction terms. The extreme points largely contributes to the estimation of quadratic terms but has no influence on the interaction terms. The center runs are also of importance as it provides an internal estimate of the error and also have some contribution towards the quadratic term estimation [25].

Now that the structure of the design matrix D is know it is possible to construct the coded variable X matrix which is required to determine the least squares estimators β. The X matrix structure is represented by the following:

$$X = \begin{bmatrix} 1 & -1 & -1 & 1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -\alpha & 0 & \alpha^2 & 0 & 0 \\ 1 & \alpha & 0 & \alpha^2 & 0 & 0 \\ 1 & 0 & -\alpha & 0 & \alpha^2 & 0 \\ 1 & 0 & \alpha & 0 & \alpha^2 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.36}$$

The firs column of ones is required for estimating constant β_0. The second and third column is just the design matrix D that was shown in (3.35). The fourth and fifth columns represents the respective quadratic terms of parameter 1 and 2. The final column represents the interaction term and is the product of parameter 1 and 2.

3.4.3 Optimization Procedure

Now that it is clear how we can use the CCD method to setup our coded variable matrix X we can use it together with (3.31) and (3.32) to obtain our second-order model and find a response surface function such as in (3.28). When the response surface is know all kinds of design changes and optimizations could be made.

The optimization of a system is an iterative approach and the initial maximum of the response surface is firstly estimated. The new set of variables are added to the coded variable matrix and the simulation or experiment is then performed with the new input parameters. The new response is then calculated and added to the y vector. The response surface is then calculated again and a new maximum is found. This is done until conversion of the response surface, meaning that the results from the simulation or experiment at a certain design point is the same as what the response surface would estimate. When the response surface is converged it is an accurate representation of the entire physical system and no further simulations are required[25]. The benefit of having a converged response surface is that the engineer can take any parameters within the domain and precisely know the outcome. When the optimum design within the given constraints are required one simply takes the maximum of the converged response surface.

This section concludes the theoretical and mathematical background of the different methods used to calculate and use the RSM.

Chapter 4

Modeling and Algorithm

In Chapter 3 all the different simulation approaches were discussed. However to simulate the radiation a model had to be created and incorporated into the simulations in order to create the full multi-physics approach. Here the geometry considered and how the radiation model works is discussed in more detail. The effects of absorber angle and permeability has an important role as well and how the absorber material is incorporated into the simulation is also looked at.

4.1 Solar Heater Geometry

Before the model that was implemented can be discussed it is important to have a brief overview of the geometry that was considered for the model. There are many factors that could influence the radiation and in effect what needs to be considered for the radiation model, so it is important to create a clear understanding of the system. In Figure 4.1 bellow the basic design and dimensions of the solar heater that was considered can be seen.

The dimensions where chosen to have a similarity with Build It Solar [14] and to be fitted on a wall of a standard small house. Since the simulation will be performed in 2D the width of the solar heater is not important and has been chosen as unit length 1m to make calculations easier.

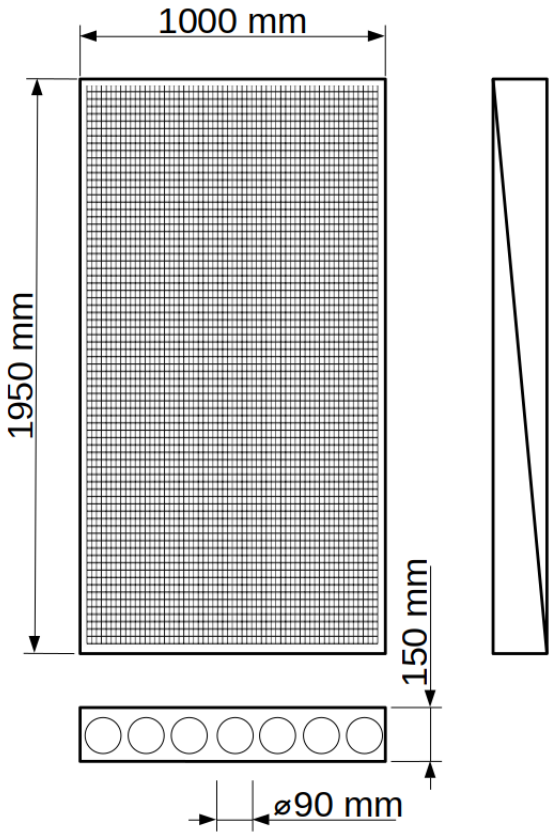

Figure 4.1: Geometry of Solar Heater Considered.

4.2 Radiation Model

4.2.1 Modeling Material Radiation Properties

As mentioned in Section 2.1.3 the parameters required to calculate the radiation heat transfer is **Absorbtivity** α, **Reflectivity** ρ and **Transmisivity** τ. In order to create the model these properties are important to know for each material involved. Thus in the case of the solar heater the properties of the front glass panel, the absorber material and the back panel of the solar heater are of interest. In Figure 4.2 below the radiation properties of standard glass is shown.

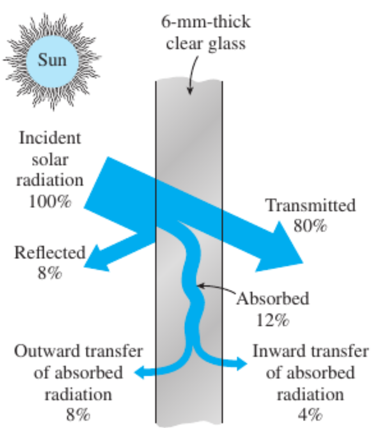

Figure 4.2: Parameters of Typical Glass. (source: [44])

From this we can summarize the radiation properties as following,

Radiation Properties	
Absorbtivity, α	0.12
Reflectivity, ρ	0.08
Transmisivity, τ	0.8

Table 4.1: Radiation Parameters of Glass.

The back panel of the solar heater also has constant radiation parameters and for the back panel black painted aluminium with some insulation behind it will be assumed. The reason for choosing a black back panel is to have the maximum amount of absorption of the radiation that does make its way through the absorber material. The radiation properties of the back panel can be summarized as [39],

Radiation Properties	
Absorbtivity, α	0.95
Reflectivity, ρ	0.05
Transmisivity, τ	0

Table 4.2: Radiation Parameters of Back Panel.

Note that there is no transmisivity as the back panel is solid and thus no light is radiated through it.
The properties of the absorber material material differs with the relative angle of the incident radiation. The total angle depends on the relation between *Incidence Angle*, θ and the *Absorber Angle*, γ as shown in Figure 4.3. It is important to distinguish and to note that in this section permeability (ratio between void and solid) refers to the radiation only and not the effect on the physical flow. For the flow the effects of permeability (porosity) is discussed later.

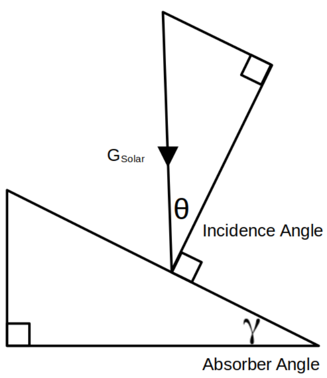

Figure 4.3: Relation between *Incidence Angle*, θ and the *Absorber Angle*.

Due to the nature of the absorber screen and this relation of the angles the radiation properties of the absorber changes.

Figure 4.4 depicts a small single block of the entire absorber screen in Figure 4.1. The absorber material has a pitch of 1mm and the thickness of the strands are 0.1mm. When the absorber is completely straight and has a 0° angle relative to

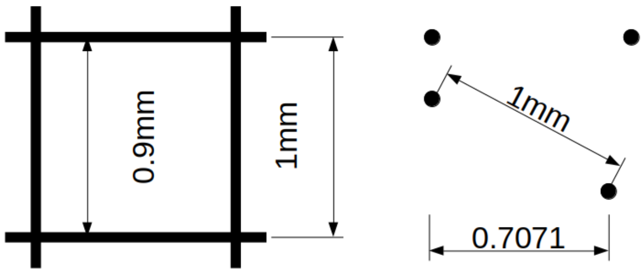

Figure 4.4: Magnified Geometry of Absorber Material.

the incident radiation, the permeability is at its highest. When the relative angle changes, the distance between two strands become smaller when viewed from the incident angle ($\theta = 45°$ used in example). Thus the permeability of the absorber in terms of radiation changes and has an effect on the radiation properties.

We assume that the strands are completely black and solid, thus where there is material there is no transmissivity. The opposite also holds and where there is void there is only transmissivity and no radiation absorbed or reflected. For the solid strands the following assumption is made.

Properties	
Absorbed	90%
Reflected	10%

Table 4.3: Ratio of Absorbed and Reflected Radiation.

This means in effect that with the change in angle the transmissivity decreases and absorbtivity and refelctivity increases since the sum of the components must = 1 as in Equation 2.7. The following equations are used to model this effect,

$$d\cos(\theta + \gamma) = l \tag{4.1}$$

$$A_{total} = dl \tag{4.2}$$

$$l_{inside} = l - s, \quad d_{inside} = d - s \tag{4.3}$$

$$A_{void} = d_{inside}\ l_{inside} \tag{4.4}$$

$$A_{solid} = A_{total} - A_{void} \tag{4.5}$$

$$R_{permeability} = 1 - \frac{A_{solid}}{A_{total}} \tag{4.6}$$

With d as original pitch, l the pitch with angle considered, d_{inside} and l_{inside} the inside dimensions excluding strand material. A_{total}, A_{void} and A_{solid} denoting the specific areas and $R_{permeability}$ the permeability ratio. Now this ratio can be used to calculate the radiation properties as,

By implementing this into the radiation model the relative angle at each discrete position the radiation properties of the absorber material can accurately be modelled.

Properties	
Absorbtivity, α	0.9(1 - $R_{permeability}$)
Reflectivity, ρ	0.1(1 - $R_{permeability}$)
Transmisivity, τ	$R_{permeability}$

Table 4.4: Absorber Radiation Properties.

4.2.2 Radiation Network

Now that the material properties for radiation of the system is know it is useful to look at the solar heater as a whole and understand how radiation for the whole system can be modelled. In Figure 4.5 a very basic single layer solar heater is depicted.

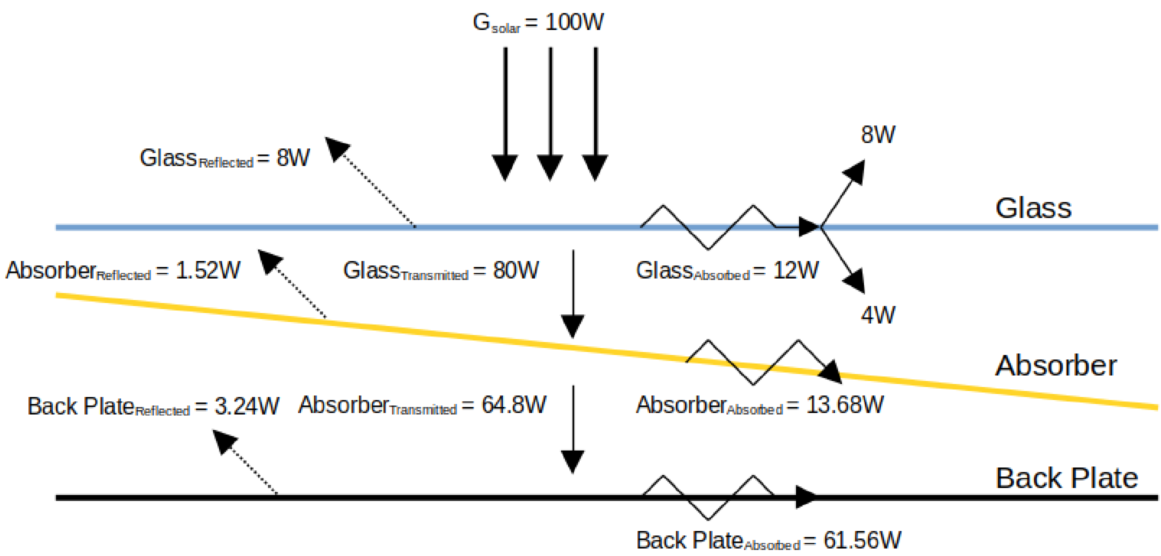

Figure 4.5: Radiation Network of Solar Heater.

Here a value of 100W for G_{solar} was used to get a better understanding of what portion of the heat energy is going where in the system. Calculating these values are fairly simple, for the glass the properties in Table 4.1 was used and as can be seen about 80% of the energy is transitted, 16% is reflected or lost to the outside air, the remaining 4% is transferred via convection inside. As discussed previously the parameters of the absorber varies depending on the relative angles, thus no definitive values can be given. It was clear however that a lot of energy is transitted to the back panel which is not ideal and thus in [14], 3 layers is used to absorb the solar energy resulting in less radiation going to the back panel and more energy being absorbed in the area where the air is flowing. The number of layers will be part of the simulation discussions later, but what is important to note that for each layer, the radiation energy that it receives is dependent on the transmissivity of the section directly above it.

Since the irradiance is measured in W/m^2 it is fairly easy to compute the energy per unit area absorbed in the absorber. This heat is then directly added to the simulation in the form of a heat source. Thus for each discrete element in the domain, belonging to the aborber, the angle is know and in turn the solar irradiance according to angle can be calculated and effectively then the energy absorbed per unit area can also be calculated.

This effectively means adding a source term to the convection-diffusion-reaction equation discussed in section 3.2.1 which is then applied inside the domain. The other two parts, the glass and the back panel is handled in a similar way, but in this case since they are on the boundaries are implemented as boundary conditions. Since the glass looses heat there is a negative heat flux over the boundary and in the case of the back panel, some of the remaining radiation is absorbed and heat source is added to the boundary accordingly.

4.3 Absorber Material

It is important to incorporate the absorber layers into the flow simulation correctly. This is done by the use of the inverse porosity as discussed in section 3.1.4. The density field needs to be created and a density is applied in the elements where the absorber material is located. The porosity was calculated with a similar method as the permeability in section 4.2.1 and the density that was assigned to the field was 0.2 for the porosity model.

It is required to simulate different shapes for the absorber material and in order to assign the density field and shape changes a parametric shape optimization feature of openCFS also used in [19] was used to do so. It is important to note that this method was not used for optimization, but only for creating the density fields and to execute shape changes as the method aligns well with what is required for the absorber layers. In Figure 4.6 a example of how the density field is created for 3 layers can be seen.

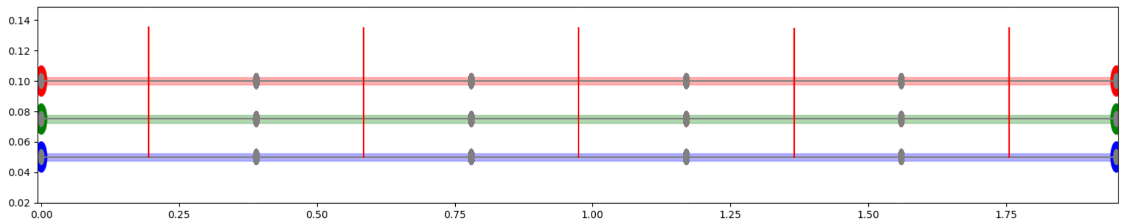

Figure 4.6: Parametric Shape Optimization Feature of openCFS.

Here the red, green and blue sections represent the different layers. Each layer has a beginning and an end point and an use assigned amount of points in between. The layer thickness can be assigned as well as the relative positions of the points in between. In this example the relative positions are all zero but the points can be assigned to move up or down according to the input. Each part between 2 points will for the purpose of this thesis be referred to as a section and the thin red lines represent the normal's of these sections. These sections will later be used to calculate their individual angles and hence the permeability and heat energy due to radiation for each section can also be computed.

These structures are then projected to a pseudo density field that is correlated with a given fixed mesh via differential mapping 4.6. This resultant density field can be seen in Figure 4.7.

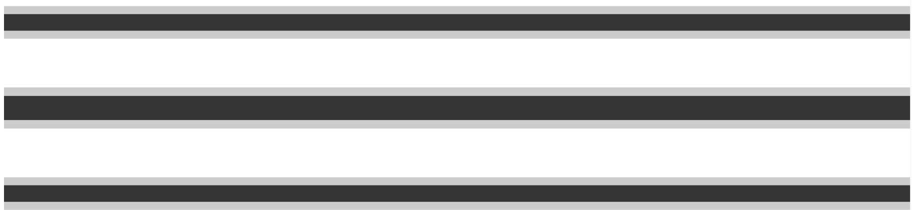

Figure 4.7: Resultant Density Field.

The appropriate density according to the porosity as discussed can then be applied and the density field can be loaded into the LBM simulation.

This approach is very useful as it makes it possible to automatically assign density fields in the shape of the layers and also different designs. This is then used to perform parameter studies, RSM and ultimately understand the sensitivity and performance of the system better.

4.4 Simulation Algorithm

Since the modeling and simulation of this problem is quite complex and involves multi-physics it was required to write an algorithm in python to automate the entire simulation and bring the different simulation and modeling steps together. The following steps where followed and implemented in the code to execute the simulation:

Algorithm Step's:

1. Create required mesh's 1) Mesh for thermal simulation 2) Mesh for LBM.
2. Set the parameters of the individual absorber layers.
3. Create the layer density field.
4. Filter density to the correct porosity.
5. Use LBM mesh + density field to run flow simulation.
6. Extract velocity and pressure data and interpolate velocity field for later use in the thermal simulation.
7. Extract element numbers and Calculate coordinates of nodes associated with the absorber material.
8. Extract coordinates of section end points.

9. Calculate radiation of each section as per radiation model.

10. Assign calculated nodes (Step: 7) to sections and add heat source to nodes accordingly.

11. Run thermal simulation with appropriate mesh and imported velocity field from (Step: 6).

12. Extract thermal results.

13. Calculate energy difference between inlet and outlet.

14. Store Results.

These results could then be stored for post-processing and to do further investigations. The basic algorithm was also later enhanced and adapted in order to perform the CCD and RSM and ultimately it was used to perform the automated optimization procedure.

Chapter 5

Simulation Application

The purpose of this chapter is to validate the simulation techniques and modelling approaches discussed in the previous chapter. An overview is given on the simulation setup and boundary conditions as well as the mesh. The flow together with heat transfer is simulated and the pressure drop over the absorber material is validated. A mesh independent study is also conducted to ensure that the mesh used is good enough to represent the physics accurately.

5.1 Basic Simulation Setup

It is important to first create a clear understanding of how the physical problem is transformed to be able to simulate it. As mentioned before simulations will be performed in 2D, thus it is important that it is set up in such a way that it captures the physics correctly.

5.1.1 Domain Geometry

The simulation will be done in 2D in order to simplify the simulation, reduce run time, since the problem lends itself well to it. Thus a 2D plane view of Figure 4.1 can be considered as seen in Figure 5.1

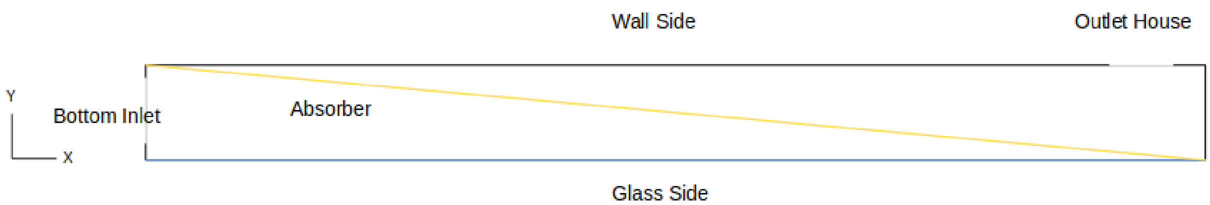

Figure 5.1: 2D Side View Geometry of Solar Heater

This orientation will be kept for the remainder of all the simulations, which makes visualization and comparisons easier. Even though the solar heater is flipped 90°, the physics of flow and heat transfer remains unaffected, but the radiation model was created with this in mind and implemented accordingly.

5.1.2 Meshing and Boundary Conditions

In terms of meshing two different meshes where used for performing simulations. The first mesh the one used for LBM simulation and the second for the thermal simulation. Both the meshes where created using the python meshing tool that comes with openCFS. For the LBM mesh some changes to the code had to be made. The standard pipe mesh was used and modified to create the so called "solar" mesh required to run the LBM simulations. For the thermal simulations the existing "bulk2d" mesh was used to create it. Both the meshes had the same resolution for the respective simulations and are Cartesian meshes. The mesh at the inlet can be seen in Figure 5.2

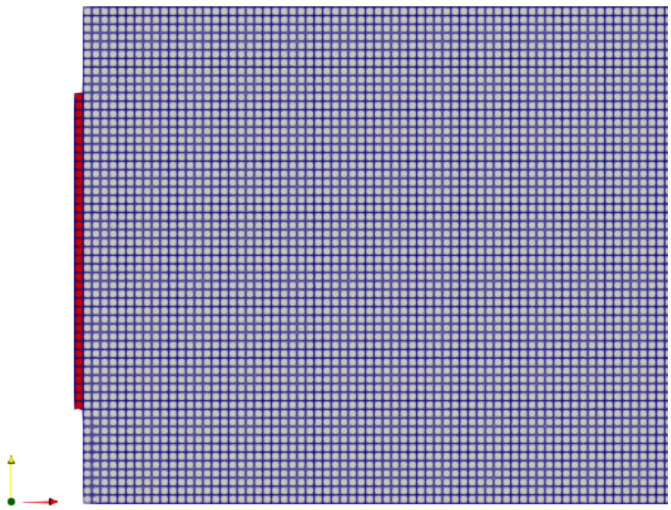

Figure 5.2: Cartesian Mesh at Inlet

The boundary conditions for the LBM simulations is visualized in Figure 5.3

Figure 5.3: BC's of LBM Simulation

Here the red elements refers to the inlet and green elements to the outlet. The blue is where the glass screen is and the rest is the walls of the solar heater, both these areas are treated as a no-slip wall. The grey section is the internal flow field of the solar heater.

The boundary conditions for the thermal simulations can be seen in Figure 5.4

Figure 5.4: BC's of Thermal Simulation

The light-blue inlet on the left is where the inlet temperature is specified. The orange at the outlet is the nodes where the exit temperature is taken to calculate the temperature difference. The red section is the screen where heat-loss is incurred and the white section is the back panel of the solar heater, that in this case absorbs the the solar radiation.

The boundary conditions are summarized in Table 5.1.

Boundary	LBM Simulation	Thermal Simulation
Inlet	Velocity Inlet	Temperature
Outlet	Outlet	n.a
Glass Screen	Wall	Heat Sink
Back Panel	Wall	Heat Source
Box	Wall	Insulated

Table 5.1: BC's Summary.

These boundary conditions where then set appropriately in the simulation files.

5.1.3 Simulation Setup

For the simulation it is important to first have the correct parameters for the LBM simulation. This means scaling needs to be done from the physical units into the LBM units. The way to do this is via the *Re* number discussed in section 3.1.1. The idea is that the *Re* number of the scaled LBM units, Re_l should match that of the real physical problem. The procedure that was followed is according to [29].

$$Re = Re_l = \frac{U_l N}{\nu_l} \tag{5.1}$$

With U_l, N, ν_l the LBM velocity, lattice height at inlet and lattice viscosity respectively. It is preferable to have the U_l more or less unit, $U_l = 1$ is chosen and depending on the grid resolution N is set. The lattice viscosity is determined from,

$$\nu_l = \frac{1}{3}(\tau - 0.5), \quad \tau = \frac{1}{\omega} \tag{5.2}$$

Here τ is the relaxation time that is dependent on parameter ω. Note that ω needs to be chosen cautiously as it affects stability and if $\omega = 2$ division by zero is incurred. The physical parameters of the system was chosen to be in line with [17] so that the results can be compared to experimental findings.

Parameter	Dimensions	Value
Flow Rate	m^3/h	48
A_{inlet}	m^2	0.0445
$T_{outside}$	°C	15
Solar Radiation	$\mathrm{W/m}^2$	933

Table 5.2: Physical Parameters.

With the physical inlet velocity = 0.299 m/s, height 0.15m and kinematic viscosity of air ν = 1.48e-5, the physical Re = 3036.7. The LBM parameters can now be scaled as,

Parameter	Dimensions
Re_l	3036.7
U_l	1
N_{inlet}	36
ω	$1.867 \approx 1.9$

Table 5.3: Scaled Parameters.

These values can now be used to setup the LBM simulation. With openCFS the simulation parameters are given in a '.xml' file. An example of the LBM parameter setup can be seen in Figure 5.5 below.

```xml
<pdeList>
  <LatticeBoltzmann>
    <regionList>
      <region name="design" boundary="false" />
      <region name="boundary" boundary="true" />
    </regionList>
    <bcsAndLoads>
      <inlet name="inlet" dof="x" value="1"/>
      <inlet name="inlet" dof="y" value="0.0"/>
      <outlet name="outlet"/>
    </bcsAndLoads>

      <LBM omega="1.9" maxWallTime="1e6"  maxIter="1000000" convergence="1e-7" solver="internal"/>
    <storeResults>
      <elemResult type="LBMVelocity">
        <allRegions/>
      </elemResult>
      <elemResult type="LBMPressure">
        <allRegions/>
      </elemResult>
      <elemResult type="mechPseudoDensity">
        <allRegions/>
      </elemResult>
      <elemResult type="LBMPhysicalPseudoDensity">
        <allRegions/>
      </elemResult>
    </storeResults>
  </LatticeBoltzmann>
</pdeList>
```

Figure 5.5: Setup of LBM in .xml file

Here the .xml file format can be seen and the values that where entered as calculated in Table 5.2. Since the number of inlet nodes is calculated from the mesh the only inputs required is ω and U_l and are set in the 'LBM' and 'bcsAndLoads' sections respectively.
For the thermal simulations, setting the boundary conditions and inlet temperature can be done in a similar way in the .xml file. The interpolated velocity field from the LBM simulation is however required and is read in via the 'scatteredData' element as seen in Figure 5.6.

```
<scatteredData>
        <csv fileName="node_extractedData.csv" id="myCSV">
            <coordinates>
                <comp col="2" dof="x" />
                <comp col="3" dof="y" />
            </coordinates>
            <quantity id="jlp" knnLib="flann" name="meanFluidMechVelocity" numNeighbors="1">
                <comp col="0" dof="x" />
                <comp col="1" dof="y" />
            </quantity>
        </csv>

</scatteredData>
```

Figure 5.6: Reading of LBM Velocities

The final file that is important to run the simulation is the 'mat.xml' file and this file contains the properties of the materials. For the case of the solar heater air is the material of choice and the applicable regions are set to air in the LBM and thermal simulation files.

5.2 Box Collector Reference

In order to ensure that the simulation and model represents the physics accurately, the parameters from section 5.1.3 where taken and will be compared to the findings in [17] as mentioned before. The values recorded in the experiment was as follows,

Parameter	Dimensions	Value
Solar Radiation	W/m^2	933
Inlet Temperature	°C	15.16
Outlet Temperature	°C	32.67

Table 5.4: Experimental Values.

These values together with the values in Table 5.3 where used to setup and run the simulation.

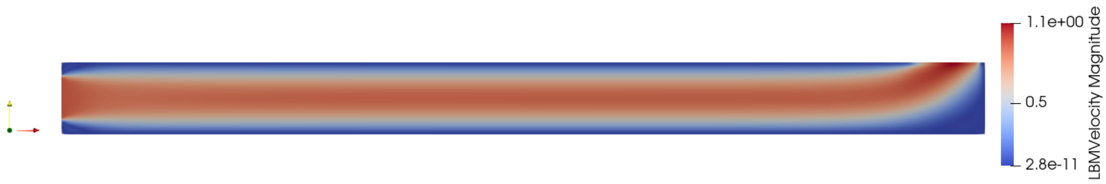

Figure 5.7: Velocity Field of Empty Box collector

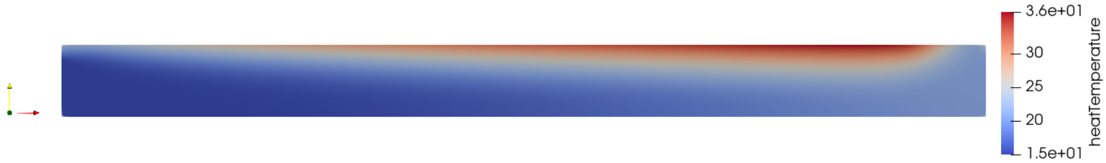

Figure 5.8: Temperature Field of Empty Box collector

In Figure 5.8 and 5.7 the velocity field and steady state temperature field of the results can be seen. There is no velocity data for the internal field from the experimental data, but regarding the solar heater in 2D is similar to internal pipe flow. Thus, the no-slip boundary layer as well as an increase in velocity in the center due to the boundary layer effect seems in line with what is physically expected.

The temperature also can be seen rising along the length of the solar heater with the warmest part being close to the back plate as expected. Since there is not much mixing and the flow has a laminar nature the conduction / convection of heat though air is very low and thus the heat is fairly localized towards where the radiation is absorbed. The temperature at the outlet ranges between 34.6 to 25.3 with an average of ≈ 30°C. The difference in temperature in terms of performance could be down to the fact that the solar heater is a bit smaller than the one used in the experiments although the same flow$/m^2$ ratio was used to determine the flow rate. The simulations and model seem to perform well and give realistic results in line with what is physically expected.

5.3 Wire Mesh Investigation

Now that the flow and thermal simulations have been verified it is important to investigate the density based approach to simulate the porosity of the wire mesh. It is important to simulate the pressure drop over the screen correctly to determine the overall efficiency of the solar heater. It is also important to determine the effects that multiple layers have on the flow field and the heat transfer.

A simulation was set up in such a way to test the pressure drop and this density approach. The porosity was calculated in the same way as in section 4.2.1, but unlike the change in permeability with incidence angle the porosity remains constant. From those calculations the density field was set to 0.2 in the elements where the wire mesh is situated. In Figure 5.9 the density field with the wire mesh perpendicular to the flow can be seen in white.

Figure 5.9: Density Field of a Single Wire Mesh

The purpose of this setup is to investigate if the pressure drop over the wire mesh is in line with what is found in literature. The same LBM parameters where used as for the box collector, however in this simulation a density field was incorporated in the simulation via a .density.xml. The results of the simulation are in the figures below.

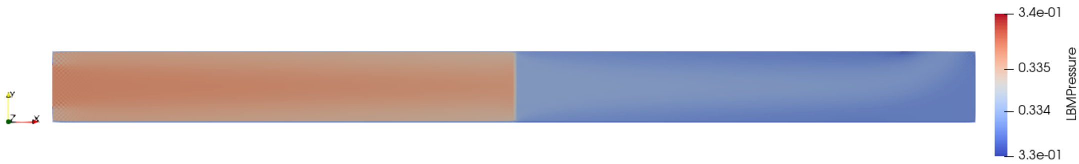

Figure 5.10: Pressure over Single Wire Mesh

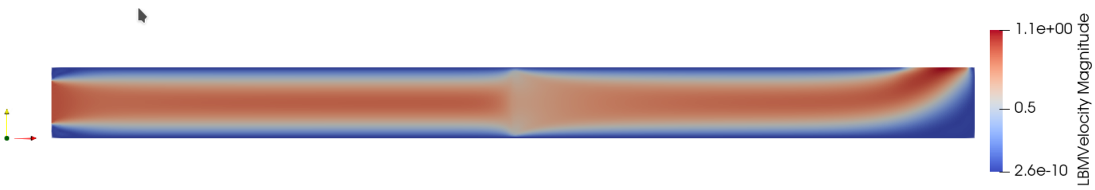

Figure 5.11: Velocity Field of Wire Mesh

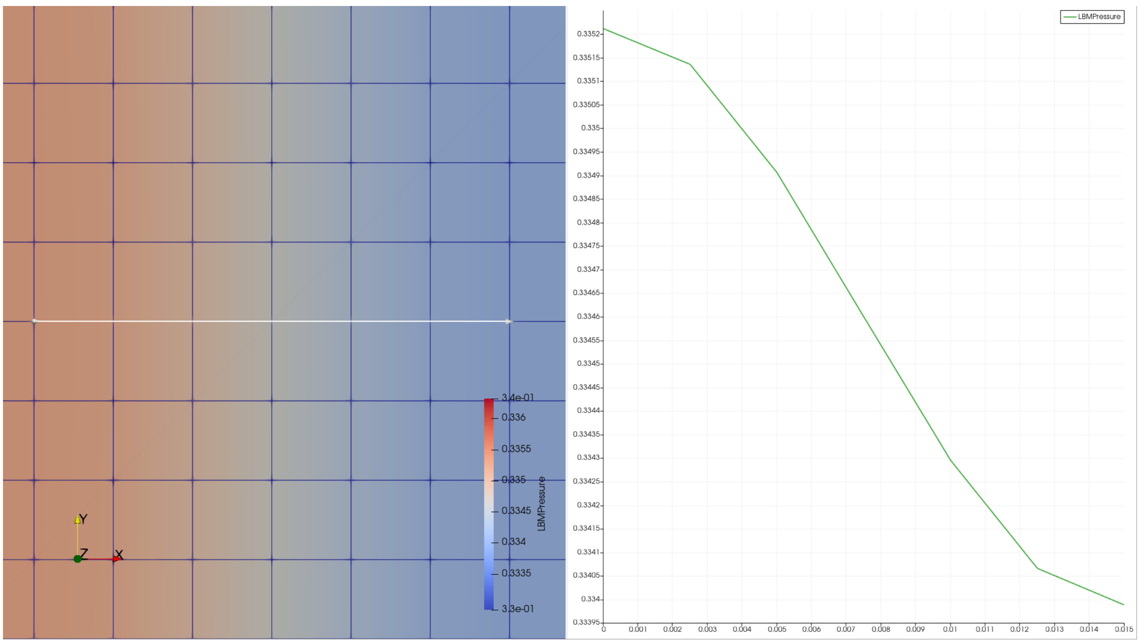

Figure 5.12: Pressure over Single Wire Mesh Close View

Here a clear pressure drop can be seen over the wire mesh as well as a change in the velocity field in the zone where the wire mesh density was incorporated. In Figure 5.12 the mesh can also be seen close up and the pressure drop over the mesh (white line on mesh represented by green graph) has been plotted to understand which values where considered. The resulting LBM pressures was then converted in accordance with [33] and the results can be seen in the table below.

Parameter	Value
LBM Pressure Before	0.335217
LBM Pressure After	0.333988
Δ LBM Pressure	0.001229
Δ Real Pressure [Pa]	367

Table 5.5: Experimental Values.

This pressure drop is dependent on flow velocity as well and when comparing to the results and findings of [16] the pressure drop from this simulation is representing the physics accurately.

Now that the simulation of the wire mesh is also clear and the density based porosity approach verified, the layers can be incorporated into the flow field with confidence that the physics will be accurately represented.

5.4 Grid Independence

To ensure that the physics that are simulated are correctly captured by the mesh and that the mesh is fine enough Grid Convergence Index (GCI) has been performed following the approach in [18], [35] and similar to [36].

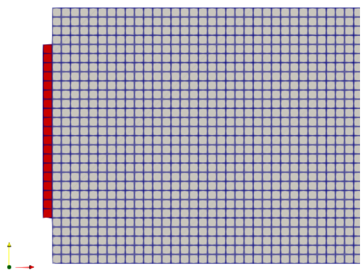

Figure 5.13: Coarse Mesh

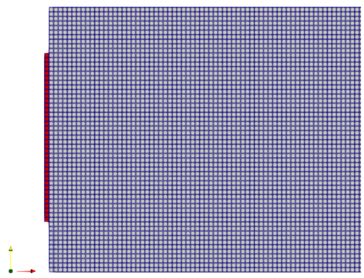

Figure 5.14: Medium Mesh

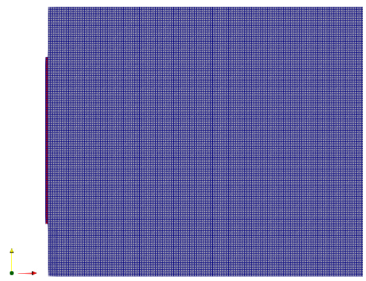

Figure 5.15: Fine Mesh

In Figure 5.13, 5.14 and 5.15 the three different meshes that where considered is shown.

The mesh resolution doubles for each refinement and the meshes that have been considered are as follows:

Mesh	Dimensions	Size [mm]
Coarse	390 x 30	5
Medium	780 x 60	2.5
Fine	1560 x 120	1.25

Table 5.6: Different meshes considered.

The simulation used for GCI is a 2 layer, straight layer configuration. The solar radiation is assumed to have a 0° incidence angle and the rest of the BC's where left the same as per the empty box case. In Figure 5.16 the density field generated can be seen with the absorber section in very light-blue and the walls in red.

Figure 5.16: Setup Used for GCI Simulation

The algorithm in section 4.4 was used to perform the three different simulations. Here different flow fields for the different meshes can be seen in Figure 5.17.

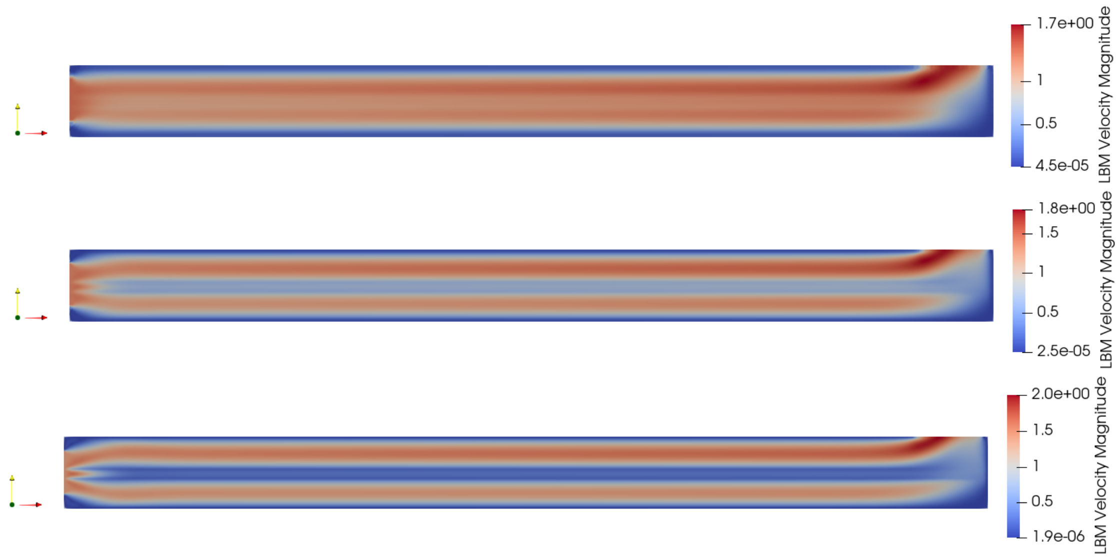

Figure 5.17: LBM Velocity Comparison

Here the mesh is getting finer from top to bottom and the flow field could be used for the thermal simulations. It can be seen that the course mesh does not resolve the flow field as accurately as the two finer ones and thus it is important to understand what is an acceptable mesh resolution.

The thermal simulations where performed next and the different temperature fields for the different meshes can be seen in Figure 5.18.

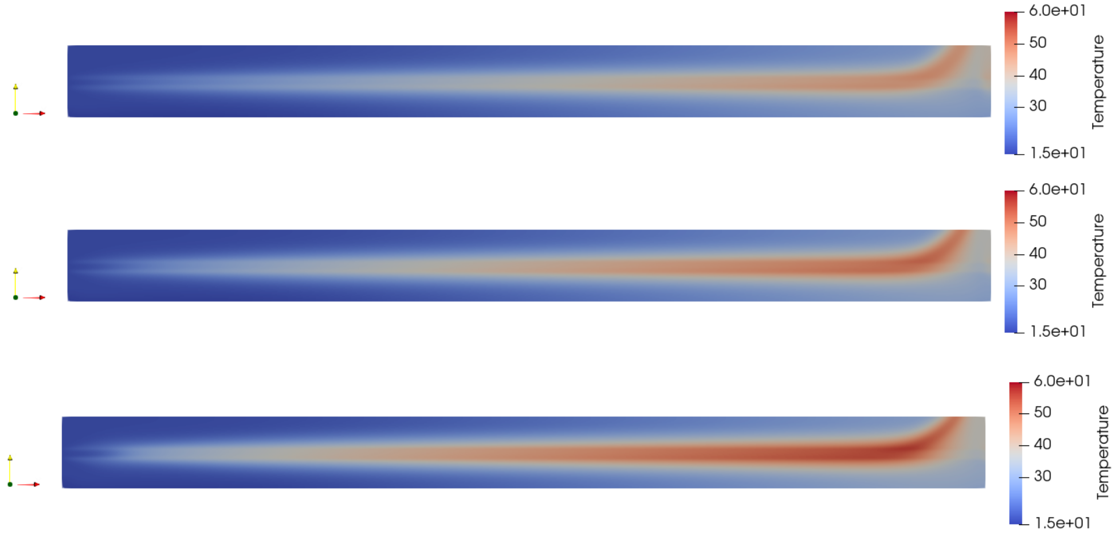

Figure 5.18: Temperature Comparison

After the simulation for all three different meshes where performed the heat energy gained at the outlet of the solar heater was chosen as the parameter of interest to perform the GCI. The heat energies are calculated in accordance with [31] and the results of the heat energy gained are:

Mesh	ΔHeat Energy [W]
Coarse	301.04
Medium	309.67
Fine	312.53

Table 5.7: GCI Heat Energy.

Now that the parameters of interest are known they can be used in the following formulas to calculate the GCI and comment on if the mesh is sufficient or not. In the formulas below the foot-script 1 refers to the fine mesh, 2 to the medium and 3 to the coarse mesh [18].

$$p = \ln\left(\frac{f_3 - f_2}{f_2 - f_1}\right) / \ln(r) \tag{5.3}$$

Where f_3, f_2, f_1 are the parameters of interest, and r the ratio of mesh size $h_3/h_2 = h_2/h_1 = 2$ for these meshes. The relative error can then be computed as,

$$e_{21} = \left|\frac{f_2 - f_1}{f_1}\right| \tag{5.4}$$

and the index calculated accordingly.

$$GCI_{21} = F_s \frac{e_{21}}{r_{21}^p - 1} \tag{5.5}$$

Here F_s is a safety factor chosen as 1.25. The steps are also computed for index GCI_{32} and then is checked if convergence is asymptotic with ratio:

$$\frac{GCI_{32}}{r^p GCI_{21}} \tag{5.6}$$

The result of these calculations is a ratio of 1.00925 which is ≈ 1 and thus it can be concluded that there is convergence. Thus the medium mesh will be used to do any of the further simulations as it gives a good balance in terms of accuracy, but also speed.

Chapter 6

Simulation Investigation

The simulations, radiation model and algorithm are all working and showing good realistic representation of the physics. An investigation needs to be done to understand the different design parameters and what effect they have on the system as a whole. There are many things that can influence how the radiation energy is absorbed as well as transferred to the airflow. Parameters like the absorber angle, layer count, how close the layers are to each other and ultimately the layer position in the domain are of interest. In this chapter the focus will lie on the first 3 aspects (angle, layer count and layer distance). The positioning of the layers and the optimization thereof can be done once a parameter study has been done and the systems sensitivity analysis is completed.

6.1 Simulation Assumptions

Before any comparisons can be made it is important to state all assumptions made and that these assumptions apply to all the simulations in order to make a direct and fair comparison.

Assumptions Made:

1. For comparison the sun is always at the same angle with 0° incident angle.
2. The solar radiation energy is 400W$/m^2$, in line with a typical average for a sunny spring or autumn day in Germany [22].
3. The same outside temperature holds for simulations that are compared.
4. The same flow rate holds for simulations that are compared.
5. The same boundary conditions hold for simulations that are compared

6.2 Layer Investigation

The first parameter to understand is the effect of the number of layers in the system. This is important as the layers act as the aborber and will be the primary heat source in the solar heater. The layers impact the system in terms of pressures also and it is important to understand the losses that result from the layers.

6.2.1 Layer Considerations

With zero layers there is not anything except the back panel to absorb the solar radiation. This means that the heat energy is very localized and outside of the main airflow, resulting in pour performance in terms of convection heat transfer and overall heat energy transfer efficiency.

The one layer approach also seems a bit impractical as the permeability of the absorber material means up to about 80% of the total solar radiation could pass through the absorber and strike the back panel, which as discussed before is not very favourable. This can also be seen in Figure 4.5 in section 4.2.2 of the radiation model.

One might think then adding as much layers as possible would be the ideal approach, but this also has its drawbacks. The more layers that are added the less solar radiation is reaching each consecutive layer. This means the layer captures very little solar radiation in the end, but is adding resistance in the flow field and increasing the pressure head. This means the fan needs to work harder and consume more energy which can also be inefficient. It is thus important to understand the impact of the layers in the system.

It was shown in section 5.3 that the pressure drop over a single layer is 367 Pa, thus the power required to overcome this can be calculated as

$$P = \Delta p\, Q. \tag{6.1}$$

With P being power in [W], Δp the pressure head in Pa and Q the flow rate in m^3/s. Using $48m^3$/h, the total power required is P ≈ 5W. However the efficiencies of the fan needs to be considered to calculate the true power required. Based on [23] 15% has been assumed and thus total power required is 33.3W.

The next parameter to consider is how much radiation/heat energy the absorber obtains for each layer. This needs to be higher than the per layer power consumed by the fan otherwise losses are incurred. Considering the 400W/m^2, a transmissivity of 0.81 and absorbtivity of 0.171 as from section 4.2.1, this becomes fairly easy to calculate as

$$E_{absorber} = G_{solar}\tau_w\tau_l\alpha_l n. \tag{6.2}$$

Here τ_w and τ_l are window and layer transmissivities. The α_l is the absorbtivity of the absorber and n is the number of layers.

This results in

Layer	$E_{absorber}[W/m^2]$
1	54.72
2	44.32
3	35.90
4	29.08

Table 6.1: Heat energy absorbed per layer.

From the table it can be concluded that anything more then 3 layers could start showing some potential losses depending on the system size and many other factors such as solar angle and irradiance. For the purpose of this thesis there will thus only be looked at 2 and 3 layers for further investigation.

6.2.2 Layer Comparison

The simulations for the two and three layer comparison was set up in the same way as the previous simulations and all the assumptions hold. The result can be seen in Figures 6.1 and 6.2.

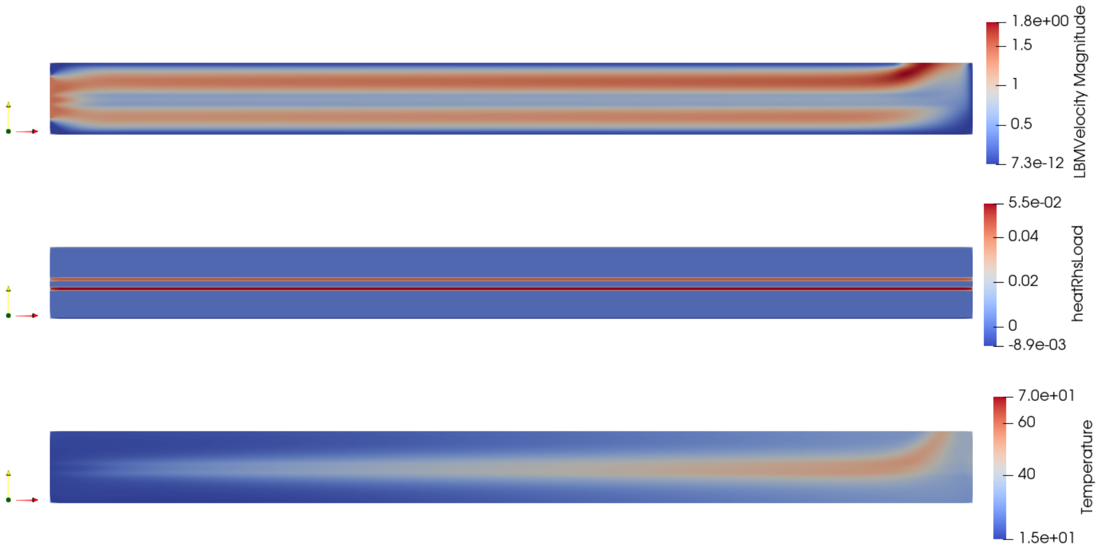

Figure 6.1: Two Absorber Layers Comparison. Velocity Field, Heat Loads and Temperature

In terms of the flow field the air flow and velocities seem fairly similar for both systems. The part of most interest is the heat load and temperature fields. For the heat loads the different layers and their respective heats can be seen. The first 2 layers of both heaters are exactly the same which is expected. For the temperature field it can be seen that the 3 layer absorber has quite a bit higher temperature and that the heat is a bit more distributed in the flow field. The parameter of interest however is the thermal energy gained and the absorber efficiency.

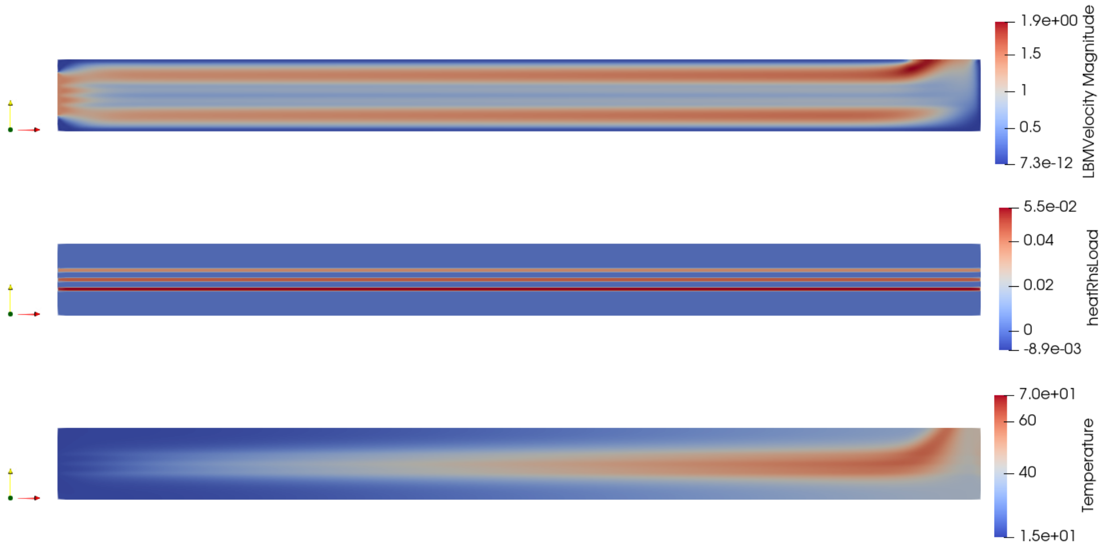

Figure 6.2: Three Absorber Layers Comparison. Velocity Field, Heat Loads and Temperature

In Table 6.2 the outcome of the respective heats can be seen.

Layers	$E_{Gained}[W]$	Efficiency [%]
2	323.92	34.60
3	444.13	47.47

Table 6.2: Heat energy absorbed per layer.

The efficiencies where calculated based on [31] and is the total energy output compared with the total amount of radiation energy hitting the surface. In this case the direct irradiation is 400W$/m^2$ and the diffuse radiation is assumed at 20% of that so 80W$/m^2$ giving a total of 480W$/m^2$ and 936W for the entire solar heater. The energy obtained by having 3 layers is significantly higher than 2 layers and much more than the per layer pressure drop discussed previously. Although this is just considering one single angle and shape it proves that the 3 layer design would be the one to consider for further simulations and improvement.

6.3 Layer Angle

Since the heat energy is so dependent on angle it is easy to make the assumption that straight layers would have the highest heat absorption and thus heat the air the most. The heat gained by the air is however not only dependant on the heat energy of the absorber, but also how effectively the heat can be transferred. This means that with a straight configuration there is not much interaction and much of the air does not come in contact with the absorber for a long time if any. When the absorber is skew or at an angle however all the air has to pass through it and come in to contact with the absorber, this might lead to a more homogeneous heat distribution and better performance overall.

In order to understand these interactions better RSM as discussed in section 3.3 was performed. The method was to use the right and left hand positions on the

(a) Design 1

(b) Design 2

(c) Design 3

(d) Design 4

(e) Design 5

(f) Design 6

(g) Design 7

(h) Design 8

(i) Design 9

Figure 6.3: Different Designs for Comparing End Positions

y-axis i.e. [0,0.15] and use these heights as the design variables. In total 9 different designs where created according to the CCD approach via a python script and the simulations ran with the same input and boundary condition parameters as before. In Figure 6.3 the different designs can be seen.

For these designs the layer distances relative to each-other was kept constant and the left and right hand side end of the absorber was free to move up and down. The heights where constrained as mentioned previously not to go outside the domain. The resulting coefficients of the response function are seen in Table 6.3.

Coefficient	Value
β_0	441.18942067
β_1	27.55463975
β_2	-76.65339558
β_3	-7.86503713
β_4	-40.20929701
β_5	-76.51990213

Table 6.3: Coefficients of End Position Response Function.

These coefficients are then used to create the surface plot for the response surface. This is useful to visualize the effect the parameters have on the system and the output and give an indication of the sensitivity of the system.

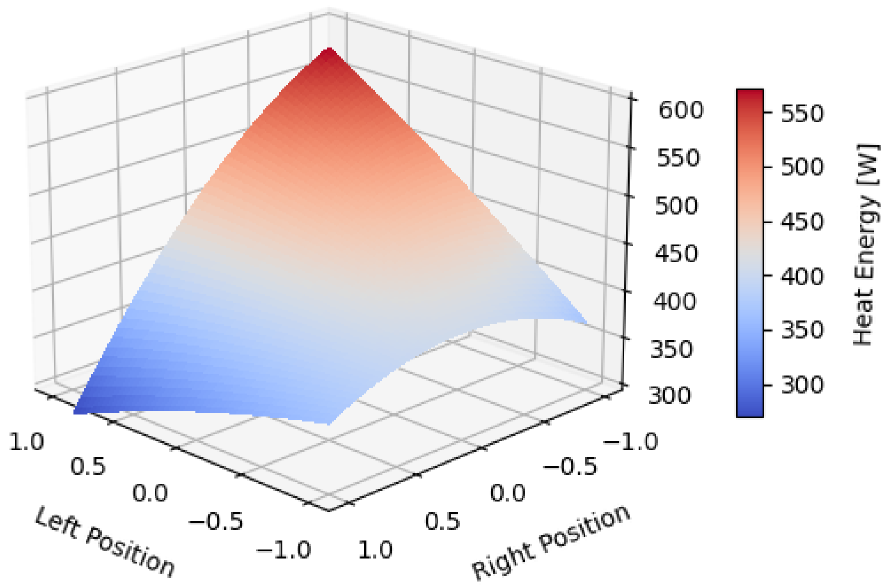

Figure 6.4: Response Surface of Parameter Study of End Positions

Here in Figure 6.4 the response surface can be seen. The position of the absorber at x = 0 is the left position and x = 1.95 the right position. These values are given in their scaled terms where 0 is the mid point at 0.075 m, 1 at the maximum 0.15 m and -1 corresponding to 0.0 m. It can be seen by looking at the response surface that the best design is when the left position is at a maximum = 0.15 m and the right position at a minimum = 0.0 m. This indicates that a diagonal design would be the best, this also coincides with the remark that was made earlier in this section that a skew/angled absorber would come into more contact with all of the airflow and potentially have better heat transfer.

In terms of sensitivity the heat energy varies with quite a considerable amount. The best design has a gain of 558.9 W and the worst design 344.4 W this is a 25,8% increase and a 22,4% decrease when compared to the straight base case in section 6.2.2. The two designs have a thermal efficiency of 59,7% and 36.8% respectively. This would indicate that the system is fairly sensitive to the absorber shape and that further investigation into the internal shape change should be done to see what performance gains could be achieved further.

6.4 Layer Distance

Another parameter of importance is how close the layers should be from each-other and how this affects the airflow and ultimately the heat transfer. In the previous section it was seen that the position and angle of the absorbers have quite a big impact on the efficiency of the absorber. A similar parameter study has thus been done as in the previous section. In this case the distance between the individual layers where the parameters of interest and used as parameters for the RSM. The original design for the layer distance can be seen below and this is the base design.

Figure 6.5: Original Design for Layer Distance Parameter Study.

The points at the edges of the 1st and 3rd layer where varied in the y-direction in order to move the layers closer or further away from the middle 2nd layer. The different designs can be seen in Figure 6.6.

After running the simulation the resulting response surface could be calculated and the coefficients can be seen in Table 7.2 below.

Coefficient	Value
β_0	434.15737114
β_1	11.22278892
β_2	-17.97430515
β_3	-11.81498067
β_4	1.52523444
β_5	2.11512992

Table 6.4: Coefficients of End Position Response Function.

The response surface could then again be plotted to visualize the effect of the parameters on the system and see how sensitive the system is to changes in the parameters.

In Figure 6.7 the response surface for the layer distance can be seen. It is important to note that for layer 1 the positive values refers to moving up in the positive y-direction closer to the middle second layer. For the third layer the negative values refer to the negative y-direction closer to the second layer. From the response surface it can be seen that it is desirable to have the layers closer to each other.

In terms of sensitivity the heat energy varies much less then with the position parameters. The best design has a gain of 453.17 W and the worst design 412.93 W this is a 2.03% increase and a 7.02% decrease when compared to the straight base case in section 6.2.2. The two designs have a thermal efficiency of 48.42% and 44.12% respectively. This would indicate that the system is much less sensitive to the absorber layer distance when compared to the layer positions. Thus the shape of the absorber should remain the priority in further optimization attempts with the layers kept close together to ensure the best efficiency in terms of layer distance is achieved.

(a) Design 1

(b) Design 2

(c) Design 3

(d) Design 4

(e) Design 5

(f) Design 6

(g) Design 7

(h) Design 8

(i) Design 9

Figure 6.6: Different Designs for Comparing End Positions

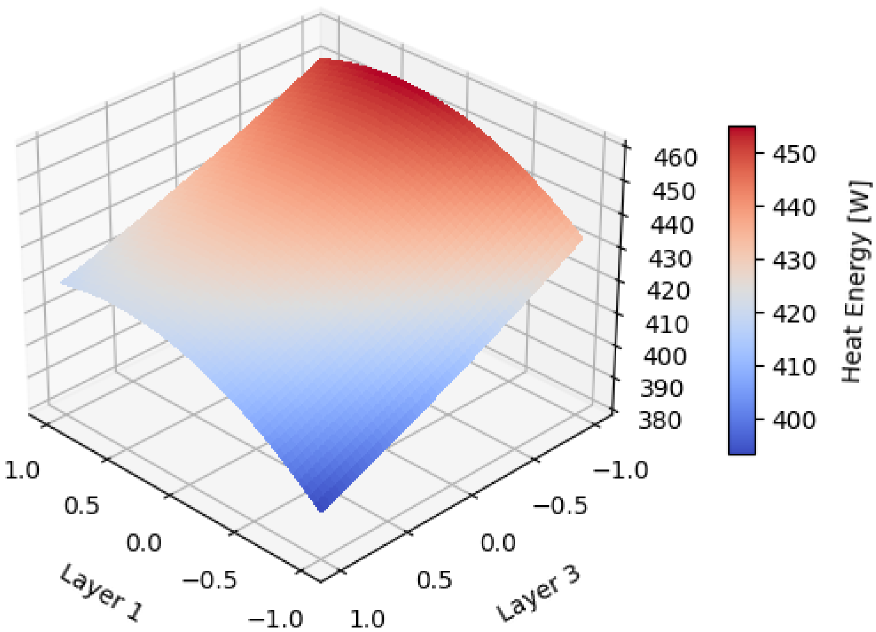

Figure 6.7: Response Surface of Parameter Study of End Positions

Chapter 7

Solar Heater Optimization Results

Up until this chapter the main focus was to create a simulation and model that accurately simulates the physics and also to understand the design parameters better. The different parameters had different affects on the solar heaters performance, but the shape of the absorber was proven to be the most important aspect. It is thus desirable to try and optimize the system even further and try and gain even more performance by changing the internal shape of the absorber. The aim is to improve even more on the resulting best design (Diagonal) from chapter 6 and to see if a little bit more performance can be achieved.

7.1 Optimization Assumptions

Before optimization can start it is important to set a few assumptions and constraints. Many of the assumptions from Chapter 6 also hold in order to compare designs and make direct comparisons.

Assumptions Made:

1. For comparison the sun is always at the same angle with 0° incident angle.
2. The solar radiation energy is 400W$/m^2$, in line with a typical average for a sunny spring or autumn day in Germany [22].
3. The same outside temperature holds for all simulations and are same as in Chapter 6.
4. The same flow rate holds for all simulations and are same as in Chapter 6.
5. The same boundary conditions hold for all simulations and are same as in Chapter 6.
6. External components and factors such as double glazing and material type-/thicknesses are not part of the scope, the focus is purely on the shape optimization of the absorber material.
7. The absorber material is constrained to the inside of the solar heater.

7.2 Parameter Setup and Shape Change

As in Chapter 6 the RSM was used together with DoE to create the response function of the system. This time however instead of just investigating the sensitivity of the system and try to obtain a better understanding which parameters are important, the RSM will be used together with optimization as discussed in section 3.4.3. It was decided for the purpose of optimizing the interior shape of the system that 6 design points will be used for the CCD. The visualization of the 6 points and the layers can be seen in Figure 7.1 below.

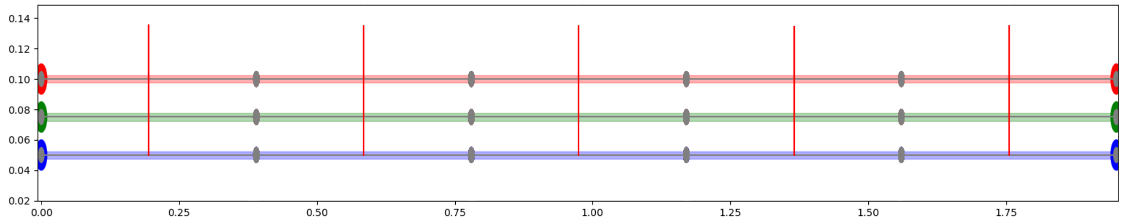

Figure 7.1: Shape Optimization 6 Variable Parameters.

In the figure the grey points shown are the design parameters that will be varied in the CCD. The two end points and 4 interior points where chosen dividing the layers into 5 sections. The layer distance was kept constant between the 3 layers meaning that all three sections that are on top of each other would move in the same orientation. The layers are only permitted to move in the normal direction indicated by the red lines.

The next step was to determine the distances that the sections could be displaced in order not to go outside the domain. This was factored into the CCD method and in the end, to create an accurate second order model, 77 different designs needed to be considered. This means 77 different simulations had to be performed in order to create the response surface. This however is much less then the 720 simulations that would have to be performed if all random combinations of the point positions where to be considered.

Function 3.33 was used to scale the variables and to create the design matrix. The design matrix was then incorporated into the python algorithm and the different parameter positions could be calculated and used directly in the python script to run the designs. This means that the y-coordinates of each design point changes for each simulation run and in turn it changes the density field according to the new shape parameters. The visualization of some examples can be seen in Figure 7.2 below where the base design can be seen as compared to designs 10, 20, 30, 40, 50 and 60 in order.

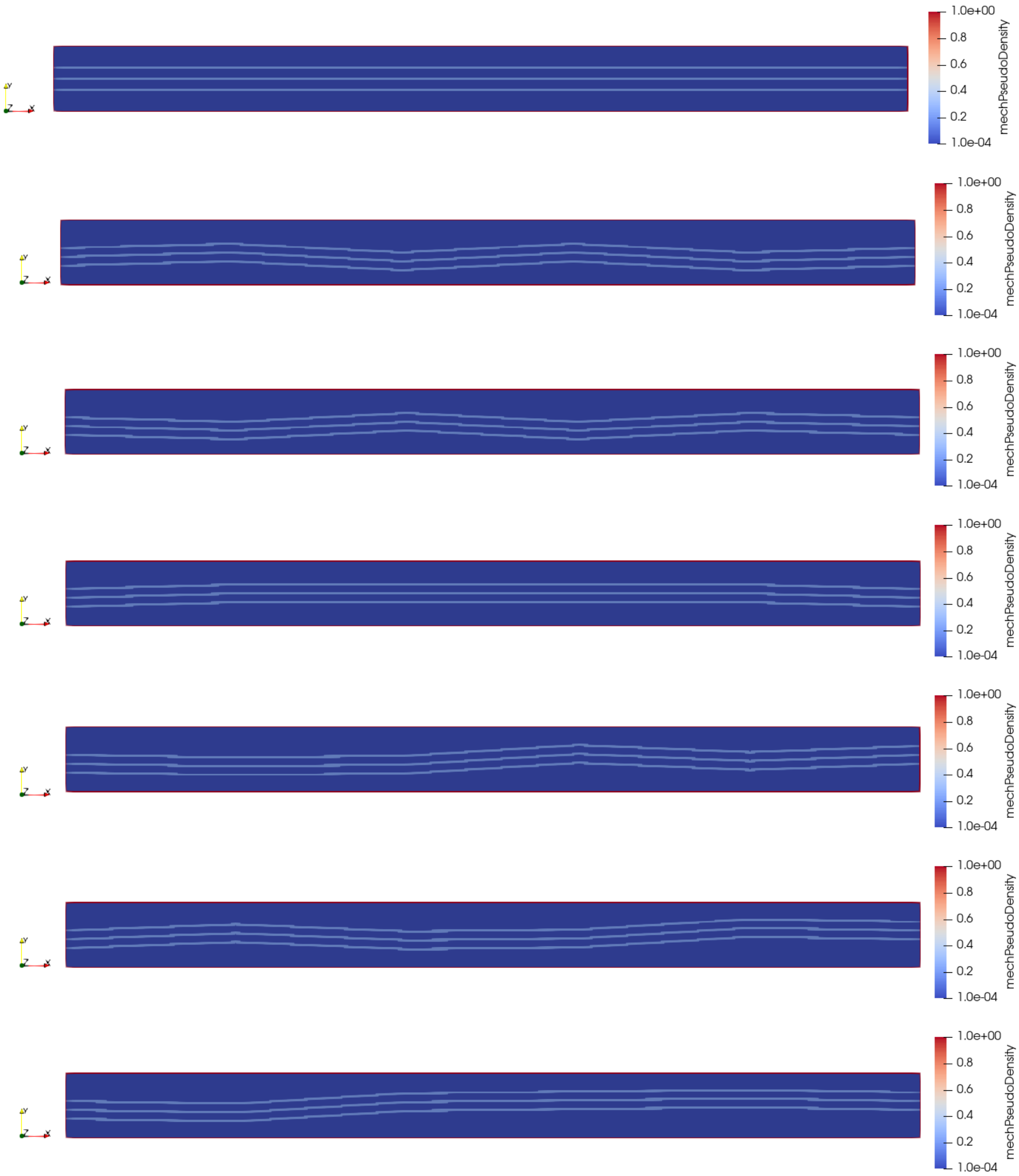

Figure 7.2: Design Density Field Comparison

Here the shape changes can clearly be seen and is dictated by the design matrix of the CCD. These different density fields could then be used to load into the LBM simulations and solve the flow field and then after temperature simulations could be done.

7.3 RSM Results

After running all the simulations the respective outlet energies could be stored in a vector and by following the approach described in 3.3.3 and using equation 3.32 the second order model could be determined and all the β values required.

Coefficient	Value	Coefficient	Value
β_0	438.12261131	β_{14}	-1.58957099
β_1	19.84692061	β_{15}	-3.01998554
β_2	15.18956919	β_{16}	-18.14278877
β_3	15.4938452	β_{17}	-27.61310954
β_4	6.30502974	β_{18}	-2.23904566
β_5	-61.88026071	β_{19}	-0.86447866
β_6	-79.25272433	β_{20}	-8.48411229
β_7	-1.68286203	β_{21}	-12.13842782
β_8	2.0077459	β_{22}	-3.00096324
β_9	2.4658156	β_{23}	-9.6933826
β_{10}	2.55094015	β_{24}	-13.04157462
β_{11}	12.10647252	β_{25}	-10.98883878
β_{12}	-3.44426763	β_{26}	-9.03457365
β_{13}	-2.35892147	β_{27}	15.52341957

Table 7.1: Coefficients of Full 6 Design Points Response Function.

The resulting function has 5 unknowns and thus it is very hard to plot or visualize the response surface as a surface plot. What does help however is to look at the coefficients of the equation and see which ones has the most impact on the resulting response. The values that stand out the most are β_1, β_5, β_6 and β_{17} and are worth a further inspection.

Here β_1 is the starting point on the left hand side and is positive. This would indicate that it is desirable to have the left hand side as high as possible similar to the findings in Chapter 6. The β_5 and β_6 values refer to the second last and final point on the right hand side. These points seem to have a very big impact and carry the most weight. It is thus desirable to have the absorber as low as possible in this region toward scaled design variable = -1, which would in turn change this into a positive value for the response. This also coincides with the findings in Chapter 6. The last point of noticeable interest is the β_{17}. This point corresponds to the relation between design points 1 and 6 and the interaction of these two terms. This seems to suggest that the end points of the absorber and its position is the most important and that most emphasis should be put on these two points when creating a design.

The rest of the points and their interactions are fairly mixed and it becomes hard to describe what factors influence the response when looking at the design points in the middle. It is also hard to predict what would be the best positions, but by looking just at the first 6 linear terms it seems that it is desirable to have the first 4 points as high as possible (scaled design variable = 1) and the last 2 points low as possible (scaled design variable = -1). The reaction of this and the influence of the non-linear and interaction terms is hard to predict by inspection and thus optimization together with a design change algorithm will be used to get more insight into the final design.

7.4 Optimization Results

Now that the final parameters and the response function is known it is possible to optimize this function. A python script has been used to take the resulting response function and find the maximum value. The maximization problem has been turned into a minimization problem by multiplying it with -1 and reads as

$$\begin{aligned} &\underset{x}{\text{minimize}} && -f(x_i) \\ &\text{subject to} && -1 \leq x_i \leq 1, \quad i = 0, \ldots, m. \end{aligned} \tag{7.1}$$

Where $f(x_i)$ is the response function and x_i the different design variables. In this case m = 5 since there are 6 design variables considered. As before - 1 corresponds to the lowest y-position and 1 to the highest y-position.

The scipy.optimize python package was used for the minimization and the steps that where followed during the optimization is as follows:

Optimization Procedure:

1. Import energy resultant vector and the design matrix .
2. From design matrix create the X matrix and use energy resultant vector to calculate β coefficients.
3. Create the corresponding response surface function using calculated coefficients.
4. Set upper and lower bound constraints.
5. Run minimization via scipy.optimize package.
6. Obtain design points.
7. Use points to change shape and create new density field.
8. Run simulation with new density field.
9. Update energy resultant vector with one extra new entry.
10. Update X matrix with design points obtained in Step 6.
11. Repeat Steps 1 to 10 and check difference in design change, if sufficient end.

In the final step when the design does not change significantly anymore the procedure can be stopped and the final design is obtained. The algorithm ran for 10 iterations and the resulting design variables added to the design matrix can be seen below the full design matrix can be found in Appendix A.

1.000E+00	1.000E+00	1.000E+00	1.000E+00	-1.000E+00	-1.000E+00
1.000E+00	4.488E-01	5.286E-01	2.026E-01	-1.000E+00	-1.000E+00
5.190E-01	6.804E-01	6.149E-01	7.757E-01	-1.000E+00	-1.000E+00
7.327E-01	6.206E-01	6.748E-01	5.330E-01	-1.000E+00	-1.000E+00
7.531E-01	6.333E-01	6.277E-01	5.936E-01	-1.000E+00	-1.000E+00
7.514E-01	6.370E-01	6.450E-01	5.908E-01	-1.000E+00	-1.000E+00
7.522E-01	6.395E-01	6.490E-01	5.927E-01	-1.000E+00	-1.000E+00
7.530E-01	6.410E-01	6.508E-01	5.942E-01	-1.000E+00	-1.000E+00
7.534E-01	6.419E-01	6.517E-01	5.951E-01	-1.000E+00	-1.000E+00
7.538E-01	6.424E-01	6.523E-01	5.957E-01	-1.000E+00	-1.000E+00

Figure 7.3: Parameters added to the design matrix at each step.

In Figure 7.3 the resulting optimized scaled design variables can be seen. As predicted in section 7.3 the first optimization corresponded to a [1 1 1 1 -1 -1] design. After about 8 iterations there was almost no more significant changes and at the 10th iteration the procedure was stopped. The change in design between iteration 9 and 10 corresponded to a 0.025mm physical change, this is to small for the mesh to capture or to practically implement in a design of this scale. The change in design over the 10 iterations can be seen in Figure 7.4. The resulting energies did also not change significantly anymore either and can be seen below.

Iteration	Heat Energy Value
1	583.497
2	556.598
3	557.915
4	558.459
5	563.529
6	563.529
7	563.529
8	564.645
9	564.645
10	564.645

Table 7.2: Heat Energy for each Design Iteration.

It is important to note also that the response function is just a second order model of the system and thus as more design points are added it becomes more and more accurate. The first design thus shows a higher gain then what the actual amount in the simulation is. Thus the simulation is performed with those design parameters and the output then added to the response surface. The new optimum based on the old data plus the new point is then determined and this procedure continues until termination. This results in a converged response surface that can accurately predict the outcome of the system without having to do a simulation.

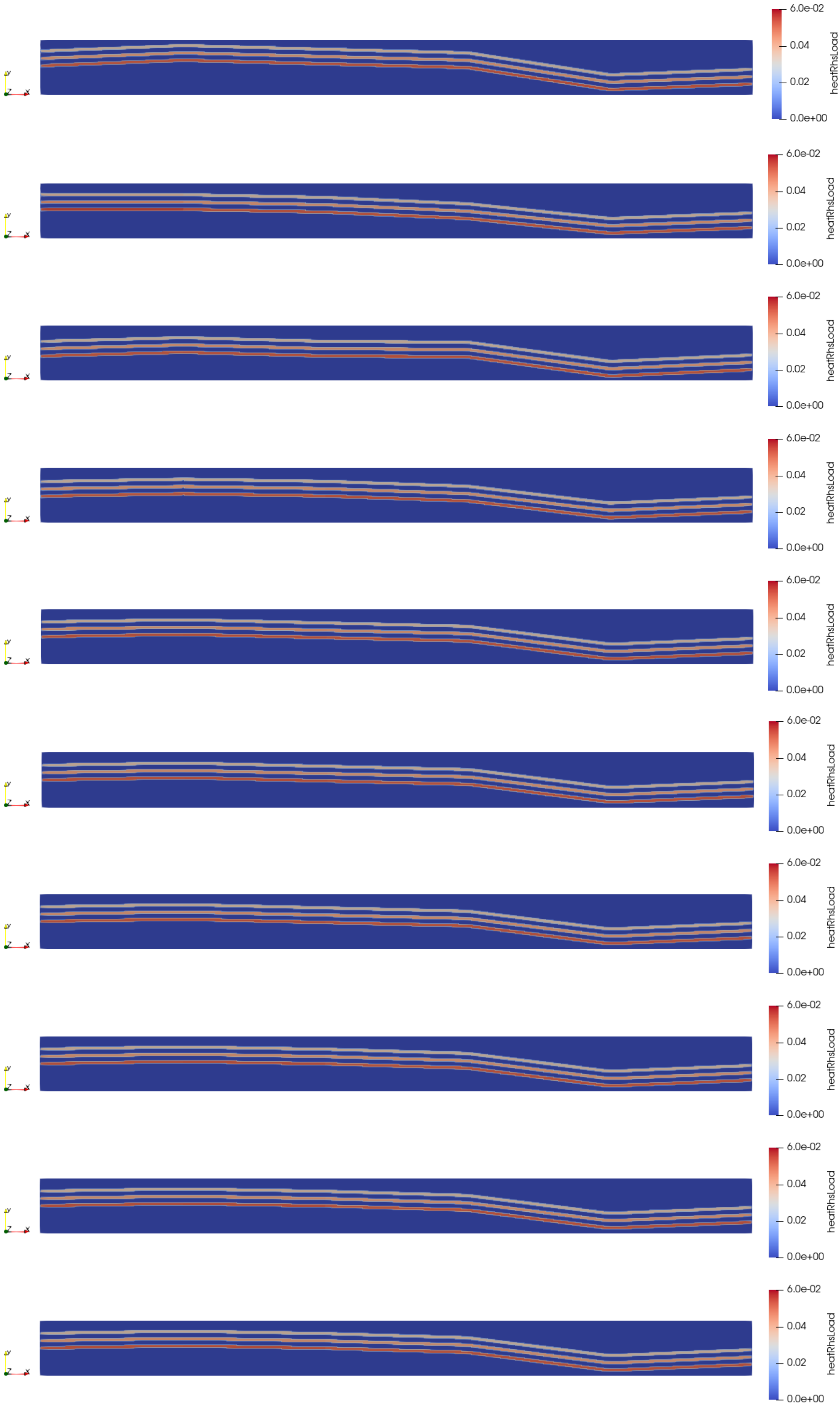

Figure 7.4: Design Differences for each Optimization step.

The density, flow field and thermal results of the final optimized solar heater can be seen below.

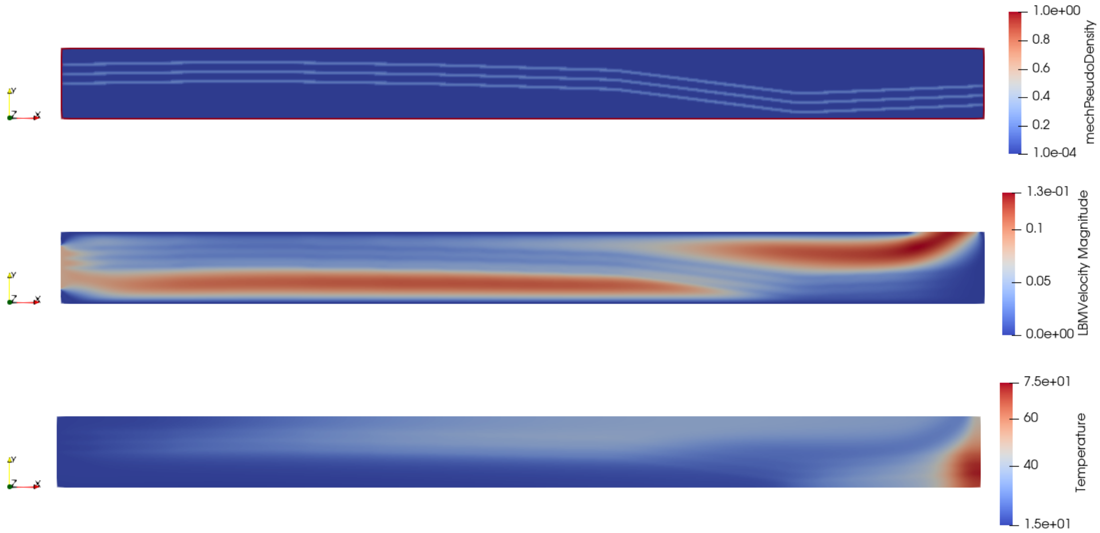

Figure 7.5: Final Design Results.

The design seems to have a more homogeneous heat distribution and some type of heat spot at the right hand bottom side of the domain. This heat spot occurs as a result of the final two points of the layers being very low down in the flow field. This slows the flow down and as can be seen in the flow field there is very low velocity in this section. This increases the residence time of the air in the heater at this section spending more time in contact with the absorber and gaining more heat energy creating a hot spot. This is normally undesirable as an even heat distribution is normally the aim when designing a solar heater. The mechanism behind it is not fully clear and further investigations should be done experimentally to validate this type of phenomena and see if it is in actual fact beneficial to the heat energy gained. Now the final design parameters are known and comparisons can be made with the designs of previous sections to see how well the optimized design performs.

7.5 Results Comparison

In this section the four different designs from the start to final optimized solar heater will be compared. The first design is the standard empty box design from section 5.2, the second design the three layer straight design from section 6.2, the third design the diagonal 3 layer design from section 6.3 and finally the optimized design from the above results. The parameters of interest was summarized and can be seen in Table 7.3 below.

Parameter	Box Design	Str 3 Layer	Diag 3 Layer	Opt 3 Layer
Thermal Energy [W]	241.2	444.13	558.9	564.65
Thermal Efficiency [%]	25.77	47.47	59,7	60.33
ΔP [Pa]	109	1770	1942	1844
Work [W]	1,2	23.54	25.82	24.53
Fan Theoretical Work [W]	8	156.93	172.13	163.53
Overall Efficiency [%]	24.91	30.68	41.32	42.85

Table 7.3: Summary of Resulting Parameters.

Here 'Thermal Energy [W]' is the total energy gain in terms of heat energy within the solar heater. 'Thermal Efficiency [%]' refers to the ratio of thermal energy gained divided by the overall radiation energy hitting the solar heater. 'ΔP [Pa]' is taken as the difference between the inlet and outlet pressures with the outlet pressure being at standard room pressure 101.3kPa and the variables scaled accordingly. 'Work [W]' is the value obtained by multiplying the pressure difference with the volume flow rate and 'Fan Theoretical Work [W]' is taking into a fan efficiency of 15% as used in section 6.2.1. Finally 'Overall Efficiency [%]' is obtained by subtracting the fan work from the thermal energy gained then calculating the efficiency.

Here it can be seen by adding the 3 layers into the solar heater it drastically improves the performance as discussed in previous sections. The most important factor seems to ensure that all the flow goes through the absorber material and thus the diagonal and optimized designs have the best performance. It can be seen that the optimized design has slightly better performance then the normal diagonal design with an increase of 5.75 W or 1.03% in thermal energy, still being a decent amount of extra energy to squeeze out of an already fairly optimal system. Also considering the pressure difference the optimized solution has slightly better pressure drop values meaning that the overall efficiency is 1.53% better. These numbers might not sound to significant but over the long term it adds up.

The most important fact is that the final two designs are significantly better then the more basic cases and that the overall system sensitivity is now well understood and optimized given the constraints. This means that the parameters defining the system performance from a absorber shape point of view is much better understood.

Chapter 8

Conclusion

The main goal of the thesis was to come up with an optimized solar heater design. It was first important to create a better understanding of how the solar heater works and the theory behind it. Based on this information the screen absorber solar heater was chosen. When it was clear what type of geometry needed to be optimized it was important to create a strategy in terms of how the multi-physics problem could be solved. The LBM method together with a convection-diffusion-reaction equation and a radiation model created specifically for this thesis was used. The simulations where validated using references from experiments as well as theoretical calculations.

After the model was proven to work for all the physics involved and the grid independence completed a parameter study was done. The parameter study focused on different aspects of the solar heater. The first parameter that was investigated was to see the effect of different layers on the system. It was proven that a 3 layer approach was best as adding more layers could lead to a bigger pressure drop and in turn make the system over all less efficient. By choosing less then 3 layers it is also not found to be a good solution as much of the solar radiation still passes through the wire mesh absorber material. This meant although the pressure drop was less the thermal efficiency was a lot worse meaning that it also performed worse then the 3 layer system overall.

After the layer count was investigated a parameter study could be done on the left and right hand side y-positions of the absorber. RSM was performed together with CCD to create a response surface. It was found that in terms of sensitivity the heat energy varies with quite a considerable amount. The best design had a gain of 558.9 W and the worst design 344.4 W this was a 25,8% increase and a 22,4% decrease when comparing to the standard straight case. This meant that the end positions of the absorber was very important to consider and impacts the performance of the screen absorber the most. The sensitivity analysis of layer distance was also conducted and it was found that The best design had a gain of 453.17 W and the worst design 412.93W this is a 2.03% increase and a 7.02% decrease when compared to the straight base case. This was significantly less impactful then the position of the absorber and the layers where kept close together for further optimizations.

The final step was to do RSM with 6 input parameters. These parameters were the two end points as well as 4 central points creating 5 different sections that could move within the domain. The optimization procedure that was followed was to calculate the response surface and find the parameters that give the highest heat energy output accordingly. This was turned into a minimization problem and solved using python's scipy.optimize package. The parameters could then be taken and fed into the simulation and the results returned to the RSM and the procedure repeated until there was convergence and the absorber did not move anymore. The final result was found to have 1.03% higher thermal efficiency and 1.53% higher overall efficiency than the diagonal design that was found to be the best from the previous parameter studies.

The final shape is fairly similar to the diagonal design and the most important parameters seem to be the position of the first node and the final two nodes. Its desirable to have the first one on the left as high as possible and the two on the right as low as possible. It was seen however that the design does form a heat spot on the lower right hand side which is normally undesirable in the solar heater system. It is unclear if this effect is contributing to the better performance of the system and further investigation would be required.

In terms of limitation the model that was created and the code is only specifically for this type of solar heater and this very specific approach, thus it can not easily be extended to other geometries. The flow rate is also limited by the LBM and higher or turbulent flows can not be considered with this approach. The suspicion based on theory and lessons learned is that more mixing and interaction with the absorber would yield even greater gains and thus turbulence would be desired.

In terms of future work it would be desirable to have an even more dynamic system in the sense that all the layers can move completely independent of each other. With the approached used in this thesis it would have meant too many design points and to much simulation would be required which was just not practical. Perhaps it is even possible to use an adjoint approach and create an adjoint based optimization of the mesh shape. Further experimental investigations could also be performed to validate or disprove the results and to have better insight into the physical effects of the system.

Appendix A

Final Design Matrix

-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01
3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01
-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01
3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01
-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01
3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01
-3.92E-01	3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01
3.92E-01	3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01
-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01
3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01
-3.92E-01	3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01
3.92E-01	3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01
-3.92E-01	-3.92E-01	3.92E-01	3.92E-01	-3.92E-01	-3.92E-01
3.92E-01	-3.92E-01	3.92E-01	3.92E-01	-3.92E-01	-3.92E-01
-3.92E-01	3.92E-01	3.92E-01	3.92E-01	-3.92E-01	-3.92E-01
3.92E-01	3.92E-01	3.92E-01	3.92E-01	-3.92E-01	-3.92E-01
-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01
3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01
-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01
3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01
-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	3.92E-01	-3.92E-01
3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	3.92E-01	-3.92E-01
-3.92E-01	3.92E-01	3.92E-01	-3.92E-01	3.92E-01	-3.92E-01
3.92E-01	3.92E-01	3.92E-01	-3.92E-01	3.92E-01	-3.92E-01
-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	3.92E-01	-3.92E-01
3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	3.92E-01	-3.92E-01
-3.92E-01	3.92E-01	-3.92E-01	3.92E-01	3.92E-01	-3.92E-01
3.92E-01	3.92E-01	-3.92E-01	3.92E-01	3.92E-01	-3.92E-01
-3.92E-01	-3.92E-01	3.92E-01	3.92E-01	3.92E-01	-3.92E-01
3.92E-01	-3.92E-01	3.92E-01	3.92E-01	3.92E-01	-3.92E-01
-3.92E-01	3.92E-01	3.92E-01	3.92E-01	3.92E-01	-3.92E-01
3.92E-01	3.92E-01	3.92E-01	3.92E-01	3.92E-01	-3.92E-01
-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01
3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01
-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01
3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01
-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	3.92E-01
3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	3.92E-01
-3.92E-01	3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	3.92E-01
3.92E-01	3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	3.92E-01
-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	3.92E-01
3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	3.92E-01
-3.92E-01	3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	3.92E-01
3.92E-01	3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	3.92E-01
-3.92E-01	-3.92E-01	3.92E-01	3.92E-01	-3.92E-01	3.92E-01
3.92E-01	-3.92E-01	3.92E-01	3.92E-01	-3.92E-01	3.92E-01
-3.92E-01	3.92E-01	3.92E-01	3.92E-01	-3.92E-01	3.92E-01
3.92E-01	3.92E-01	3.92E-01	3.92E-01	-3.92E-01	3.92E-01
-3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	3.92E-01

Figure A.1

3.92E-01	-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	3.92E-01
-3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	3.92E-01
3.92E-01	3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	3.92E-01
-3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	3.92E-01	3.92E-01
3.92E-01	-3.92E-01	3.92E-01	-3.92E-01	3.92E-01	3.92E-01
-3.92E-01	3.92E-01	3.92E-01	-3.92E-01	3.92E-01	3.92E-01
3.92E-01	3.92E-01	3.92E-01	-3.92E-01	3.92E-01	3.92E-01
-3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	3.92E-01	3.92E-01
3.92E-01	-3.92E-01	-3.92E-01	3.92E-01	3.92E-01	3.92E-01
-3.92E-01	3.92E-01	-3.92E-01	3.92E-01	3.92E-01	3.92E-01
3.92E-01	3.92E-01	-3.92E-01	3.92E-01	3.92E-01	3.92E-01
-3.92E-01	-3.92E-01	3.92E-01	3.92E-01	3.92E-01	3.92E-01
3.92E-01	-3.92E-01	3.92E-01	3.92E-01	3.92E-01	3.92E-01
-3.92E-01	3.92E-01	3.92E-01	3.92E-01	3.92E-01	3.92E-01
3.92E-01	3.92E-01	3.92E-01	3.92E-01	3.92E-01	3.92E-01
-1.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
1.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
0.00E+00	-1.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
0.00E+00	1.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
0.00E+00	0.00E+00	-1.00E+00	0.00E+00	0.00E+00	0.00E+00
0.00E+00	0.00E+00	1.00E+00	0.00E+00	0.00E+00	0.00E+00
0.00E+00	0.00E+00	0.00E+00	-1.00E+00	0.00E+00	0.00E+00
0.00E+00	0.00E+00	0.00E+00	1.00E+00	0.00E+00	0.00E+00
0.00E+00	0.00E+00	0.00E+00	0.00E+00	-1.00E+00	0.00E+00
0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00	0.00E+00
0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-1.00E+00
0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
1.00E+00	1.00E+00	1.00E+00	1.00E+00	-1.00E+00	-1.00E+00
1.00E+00	4.49E-01	5.29E-01	2.03E-01	-1.00E+00	-1.00E+00
5.19E-01	6.80E-01	6.15E-01	7.76E-01	-1.00E+00	-1.00E+00
7.33E-01	6.21E-01	6.75E-01	5.33E-01	-1.00E+00	-1.00E+00
7.53E-01	6.33E-01	6.28E-01	5.94E-01	-1.00E+00	-1.00E+00
7.51E-01	6.37E-01	6.45E-01	5.91E-01	-1.00E+00	-1.00E+00
7.52E-01	6.39E-01	6.49E-01	5.93E-01	-1.00E+00	-1.00E+00
7.53E-01	6.41E-01	6.51E-01	5.94E-01	-1.00E+00	-1.00E+00
7.53E-01	6.42E-01	6.52E-01	5.95E-01	-1.00E+00	-1.00E+00
7.54E-01	6.42E-01	6.52E-01	5.96E-01	-1.00E+00	-1.00E+00

Figure A.2

Appendix B

Main Final Code

```
import numpy as np
from optimization_tools import*
from xml.dom import minidom
import xml.etree.ElementTree as ET
from operator import itemgetter
import fileinput
import sys
import matplotlib.pyplot as plt
import h5py
import hdf5_tools as hdf5T
from scipy.interpolate import griddata
from scipy import interpolate
import os
import csv
import pandas as pd
from cfs_utils import*
ET.register_namespace("", "http://www.cfs++.org/simulation")
from pyDOE2 import*


def round_nearest(x, a):
    return round(x / a) * a
#Starting Variables
param = 10
allPressures = np.zeros((param, 2))
allEnergy = []
sections = 5
layers = 3
nmrElements = 780
#Total Loop for each itteration
for iterationNum in range(param):
    #RSM CCD Setup
    designPoints = 6
    maxDistance = 0.025
    midDistance = 0.075
```

```
dMatrix = np.loadtxt("optResultsNew.txt", comments="#",\
delimiter=",", unpack=False)
newdMatrix = dMatrix * maxDistance + midDistance
print(dMatrix[2])
print(np.size(newdMatrix))

#Make copy of mesh from backup
command = "cp bulk2d_780_60-w_1_95-h_0_15BackUp.mesh\
bulk2d_780_60-w_1_95-h_0_15.mesh"
os.system(command)
#Specify Domain
domainX = 1.95
domainY = 0.15
contingency = 0.01
contingencyX = 0.0025
#Set Parameters for density field
paramXML = "spaghetti_bending.density.xml"
xml = open_xml(paramXML)

layer1Points = newdMatrix - 0.02
layer2Points = newdMatrix
layer3Points = newdMatrix + 0.02

replace(xml, '//shapeParamElement[@nr="0"]/@design',\
str(x1test));
replace(xml, '//shapeParamElement[@nr="2"]/@design',\
str(x2test));

replace(xml, '//shapeParamElement[@nr="1"]/@design',\
str(round_nearest(layer1Points[0], 0.0025)));
replace(xml, '//shapeParamElement[@nr="3"]/@design',\
str(round_nearest(layer1Points[5], 0.0025)));

replace(xml, '//shapeParamElement[@nr="5"]/@design',\
str(round_nearest(layer1Points[1] - midDistance + 0.02,\
0.0025)));
replace(xml, '//shapeParamElement[@nr="6"]/@design',\
str(round_nearest(layer1Points[2] - midDistance + 0.02,\
0.0025)));
replace(xml, '//shapeParamElement[@nr="7"]/@design',\
str(round_nearest(layer1Points[3] - midDistance + 0.02,\
0.0025)));
replace(xml, '//shapeParamElement[@nr="8"]/@design',\
str(round_nearest(layer1Points[4] - midDistance + 0.02,\
0.0025)));

replace(xml, '//shapeParamElement[@nr="9"]/@design',\
str(x1test));
```

```
replace(xml, '//shapeParamElement[@nr="11"]/@design',\
str(x2test));

replace(xml, '//shapeParamElement[@nr="10"]/@design',\
str(round_nearest(layer2Points[0], 0.0025)));
replace(xml, '//shapeParamElement[@nr="12"]/@design',\
str(round_nearest(layer2Points[5], 0.0025)));

replace(xml, '//shapeParamElement[@nr="14"]/@design',\
str(round_nearest(layer2Points[1] - midDistance, 0.0025)));
replace(xml, '//shapeParamElement[@nr="15"]/@design',\
str(round_nearest(layer2Points[2] - midDistance, 0.0025)));
replace(xml, '//shapeParamElement[@nr="16"]/@design',\
str(round_nearest(layer2Points[3] - midDistance, 0.0025)));
replace(xml, '//shapeParamElement[@nr="17"]/@design',\
str(round_nearest(layer2Points[4] - midDistance, 0.0025)));

replace(xml, '//shapeParamElement[@nr="18"]/@design',\
str(x1test));
replace(xml, '//shapeParamElement[@nr="20"]/@design',\
str(x2test));

replace(xml, '//shapeParamElement[@nr="19"]/@design',\
str(round_nearest(layer3Points[0], 0.0025)));
replace(xml, '//shapeParamElement[@nr="21"]/@design',\
str(round_nearest(layer3Points[5], 0.0025)));

replace(xml, '//shapeParamElement[@nr="23"]/@design',\
str(round_nearest(layer3Points[1] - midDistance - 0.02,\
0.0025)));
replace(xml, '//shapeParamElement[@nr="24"]/@design',\
str(round_nearest(layer3Points[2] - midDistance - 0.02,\
0.0025)));
replace(xml, '//shapeParamElement[@nr="25"]/@design',\
str(round_nearest(layer3Points[3] - midDistance - 0.02,\
0.0025)));
replace(xml, '//shapeParamElement[@nr="26"]/@design',\
str(round_nearest(layer3Points[4] - midDistance - 0.02,\
0.0025)));

xml.write('spaghetti_bending.density.xml')

#Run command for creating density field
command = "cfs_rel -m bulk2d_780_60-w_1_95-h_0_15.mesh\
-x spaghetti_bending.density.xml spaghetti_bending"
os.system(command)

#Filter density field to Porosity
```

```
originalDensity = read_density('spaghetti_bending.density.xml')
originalDensity[originalDensity < 0.45] = 0.0001
originalDensity[originalDensity >= 0.45] = 0.2
write_density_file('spaghetti_bending_new.density.xml'\
, originalDensity)

#Run LBM
command = "cfs_rel -m solar-nx780-ny_60.mesh -x \
spaghetti_bending_new.density.xml solarLBM"
os.system(command)

#Extract LBM Data
lbmf = h5py.File("solarLBM.cfs")
e = hdf5T.get_result(lbmf, 'LBMVelocity', region='design',\
step='last', multistep=1)
c = hdf5T.get_centroids(lbmf, region='design')
n = hdf5T.get_coordinates(lbmf, region='design')

centerX = c[:, 0]
centerY = c[:, 1]
nodeX = n[:, 0]
nodeY = n[:, 1]
u = e[:, 0]
v = e[:, 1]

points = np.transpose(np.vstack((centerX, centerY)))
newPoints = np.transpose(np.vstack((nodeX, nodeY)))
#Interpolate Data
u_interp = interpolate.griddata(points, u, newPoints,\
method='linear')
v_interp = interpolate.griddata(points, v, newPoints,\
method='linear')
u_interp_Boundary = interpolate.griddata(points, u,\
newPoints, method='nearest')
v_interp_Boundary = interpolate.griddata(points, v,\
newPoints, method='nearest')

for k in range(np.size(u_interp)):
    if np.isnan(u_interp[k]):
        u_interp[k] = u_interp_Boundary[k]
    if np.isnan(v_interp[k]):
        v_interp[k] = v_interp_Boundary[k]

df = pd.DataFrame(list(zip(*[u_interp, v_interp,\
newPoints[:, 0], newPoints[:, 1]])))
df.to_csv('node_extractedData.csv', header=False, index=False)
lbmf.close()
#Read density field and obtain element numbers that have density
```

```
mydoc = minidom.parse('spaghetti_bending_new.density.xml')
elements = mydoc.getElementsByTagName('element')
solids = []
f = open("solids.txt", "w")
for elem in elements:
    if float(elem.attributes['design'].value) > 0.15:
        solids.append(float(elem.attributes['nr'].value))
        f.write(elem.attributes['nr'].value + "\n")
f.close()
#Calculate element node coordinates
nmrElements = 780
discDistance = 1.95 / nmrElements

halfDisc = discDistance / 2
solidsCoord = np.array(solids)

y1Coord = np.round((np.floor(solidsCoord / nmrElements)\
* discDistance), 5)
x1Coord = np.round((((((solidsCoord / nmrElements) % 1)\
* nmrElements) - 1) * discDistance), 5)

y2Coord = np.round((np.floor(solidsCoord / nmrElements)\
* discDistance), 5)
x2Coord = np.round((((((solidsCoord / nmrElements) % 1)\
* nmrElements) - 1) * discDistance + discDistance), 5)

y3Coord = np.round((np.floor(solidsCoord / nmrElements)\
* discDistance + discDistance), 5)
x3Coord = np.round((((((solidsCoord / nmrElements) % 1)\
* nmrElements) - 1) * discDistance + discDistance), 5)

y4Coord = np.round((np.floor(solidsCoord / nmrElements)\
* discDistance) + discDistance, 5)
x4Coord = np.round((((((solidsCoord / nmrElements) % 1)\
* nmrElements) - 1) * discDistance), 5)

xx1Coord = np.append(x1Coord, x2Coord)
xx2Coord = np.append(x3Coord, x4Coord)
yy1Coord = np.append(y1Coord, y2Coord)
yy2Coord = np.append(y3Coord, y4Coord)

xCoord = np.append(xx1Coord, xx2Coord)
yCoord = np.append(yy1Coord, yy2Coord)

allCoords = np.zeros((np.size(xCoord), 2))
allCoords = allCoords.tolist()
#Assemble into all coordinates matrix
allCoords[0][0] = xCoord[0]
```

```
allCoords[0][1] = yCoord[0]
nmrOfNodes = 0
#Ensure there are no duplicates of coordinates
for i in range(1, np.size(xCoord)):

    if [float(xCoord[i]), float(yCoord[i])] in allCoords:
        continue
    else:
        allCoords[i][0] = xCoord[i]
        allCoords[i][1] = yCoord[i]
        nmrOfNodes = nmrOfNodes + 1

allCoords = sorted(allCoords, key=itemgetter(0))

#Load absorber section data and coordinates
absorberAdjust = layers * (sections + 1)
absorberX = np.loadtxt("spaghetti_Points.txt")\
[absorberAdjust:, 0]
absorberY = np.loadtxt("spaghetti_Points.txt")\
[absorberAdjust:, 1]
command = "mv spaghetti_Points.txt spaghetti_Points_Previous.txt"
os.system(command)
absorberArea = []
absorberAngle = []
#Calculate length and angle of section
for j in range(np.size(absorberY) - 1):
    if abs(absorberX[j] - absorberX[j + 1]) > 0.51:
        continue
    distanceY = absorberY[j + 1] - absorberY[j]
    distanceX = absorberX[j + 1] - absorberX[j]

    totalLength = np.sqrt(distanceX ** 2 + distanceY ** 2)
    absorberArea.append(totalLength)
    absorberAngle.append(np.arctan(distanceY / distanceX))

# SOLAR RADIATION FROM THE SUN - MAXIMUM RADIATION\
THAT REACHES THE COLLECTOR #
solarRadiation = 400
solarAngle = 0
solarAngleRad = solarAngle * (math.pi / 180)
diffuseRadiation = 0.2 * solarRadiation
solarG = (solarRadiation * math.cos(solarAngleRad))\
+ diffuseRadiation
print(solarG)
# MATERIAL PROPERTIES #
# GLASS #
glassAbsorptivity = 0.12
glassReflectivity = 0.08
```

```
glassTransmissivity = 0.8
# ABSORBER MATERIAL #
absorberAbsorptivity = 0.171
absorberReflectivity = 0.019
absorberTransmissivity = 0.81
# GLASS LOSSES #
glassReflected = glassReflectivity * solarG
glassAbsorbed = glassAbsorptivity * solarG
glassTransmitted = glassTransmissivity * solarG
#Check which coordinates belong to which section
counterSectionNodes = []
for j in range(np.size(absorberX) - 1):
    print(j)
    # print(abs(absorberXL1[j] - absorberXL1[j+1]))
    if abs(absorberX[j] - absorberX[j + 1]) > 1.5:
        continue
    m = ((absorberY[j + 1] - absorberY[j]) / (absorberX[j + 1]\
    - absorberX[j]))
    extraRegion = 0.005
    nodesCounter = 0
    for i in range(np.size(xCoord)):
        if allCoords[i][0] >= absorberX[j] and allCoords[i][0]\
        < absorberX[j + 1] and allCoords[i][1] <= m * (
                allCoords[i][0] - absorberX[j]) + absorberY[j]\
                + extraRegion and allCoords[i][1] > m * (
                allCoords[i][0] - absorberX[j]) + absorberY[j]\
                - extraRegion:

                nodesCounter = nodesCounter + 1

    counterSectionNodes.append(nodesCounter)

print(counterSectionNodes)

#Start writing to thermal setup file
tree = ET.parse('thermalBasic.xml')
root = tree.getroot()
heatCounter = 0
totalNodeCount = 0
for j in range(np.size(absorberX) - 1):
    print(j)
    # print(abs(absorberXL1[j] - absorberXL1[j+1]))
    if abs(absorberX[j] - absorberX[j + 1]) > 1.5:
        continue
    m = ((absorberY[j + 1] - absorberY[j]) / (absorberX[j + 1]\
    - absorberX[j]))
    print(m)
    #Assign solar heat according to model and permeability \
```

```
and layer depth
sectionAngle = solarAngleRad + absorberAngle[heatCounter]

if heatCounter < sections:
    totalSectionRadiation = glassTransmitted
elif heatCounter < 2 * sections:
    totalSectionRadiation = glassTransmitted * (0.9 *\
    np.cos(solarAngleRad + absorberAngle[heatCounter\
    -sections]) * 0.9)
elif heatCounter < 3 * sections:
    totalSectionRadiation = glassTransmitted * (0.9 \
    * np.cos(solarAngleRad + absorberAngle[heatCounter\
    -2*sections]) * 0.9) * (0.9 * np.cos(solarAngleRad\
    + absorberAngle[heatCounter-sections]) * 0.9)

sectionG = (totalSectionRadiation * np.cos(sectionAngle) *\
absorberArea[heatCounter])  # [W]

#Assign nodal point heats to each node
extraRegion = 0.005
counterNodes = 0
for i in range(np.size(xCoord)):
    if allCoords[i][0] >= absorberX[j] and allCoords[i][0] \
    < absorberX[j + 1] and allCoords[i][1] <= m * (
            allCoords[i][0] - absorberX[j]) + absorberY[j] \
            + extraRegion and allCoords[i][1] > m * (
            allCoords[i][0] - absorberX[j]) + absorberY[j] \
            - extraRegion:

        nodes = ET.Element("nodes", {"name": 'section ' +\
        str(j) + '_' + str(totalNodeCount) + '_'\
        + str(counterNodes)})
        coord = ET.SubElement(nodes, "coord",\
        {"x": str(allCoords[i][0]), "y": str(allCoords[i][1])})
        root[1][1].append(nodes)
        heatSource = ET.Element("heatSource",\
        {"name": 'section ' + str(j) + '_' +\
        str(totalNodeCount) + '_' + str(counterNodes),\
        "value": str((sectionG * 0.9 * (1 - (0.9 * \
        np.cos(sectionAngle) * 0.9)))/\
        counterSectionNodes[heatCounter])})
        root[2][1][0][2].append(heatSource)

        counterNodes = counterNodes + 1
        totalNodeCount = totalNodeCount + 1

heatCounter = heatCounter + 1
```

```
    #Write final thermal setup final with all parameters set
    tree.write('thermalAbsorber.xml')
    #Run thermal simulation
    command = "cfs_rel -m bulk2d_780_60-w_1_95-h_0_15.mesh\
    thermalAbsorber"
    os.system(command)
    #Store files per itteration number for post processing
    lbmString = "solarLBM_" + str(iterationNum)  + ".cfs"
    command = "cp solarLBM.cfs " + lbmString
    os.system(command)

    thermalString = "thermalAbsorber_" + str(iterationNum)
+ ".cfs"
    command = "cp thermalAbsorber.cfs " + thermalString
    os.system(command)

    spaghettiString = "spaghetti_bending_" + str(iterationNum)
+ ".cfs"
    command = "cp spaghetti_bending.cfs " + spaghettiString
    os.system(command)
    #Calculate outlet parameters
    pressuref = h5py.File("solarLBM.cfs")
    totalPressure = hdf5T.get_result(pressuref, 'LBMPressure',\
    region='design', step='last', multistep=1)

    outletNodeCount = 0.15 / discDistance

    initialNodeP = np.size(totalPressure) - (5)
    finalNodeP = int(initialNodeP - outletNodeCount)

    outletPressure = totalPressure[finalNodeP:initialNodeP]
    inletPoints = range((11 * 778 + 1),(47 * 778 + 1), 778)
    inletPressure = totalPressure[inletPoints]

    allPressures[iterationNum][0] = np.average(inletPressure)
    allPressures[iterationNum][1] = np.average(outletPressure)

    #Calculate Thermal parameters
    thermalf = h5py.File("thermalAbsorber.cfs")
    totalTemp = hdf5T.get_result(thermalf, 'heatTemperature',\
    region='mech', step='last', multistep=1)

    initialNode = np.size(totalTemp) - (nmrElements + 5)
    finalNode = int(initialNode - outletNodeCount)
    outletTemp = totalTemp[finalNode:initialNode]

    outletVelX = np.array(u_interp[-64:-4])
    outletVelY = np.array(v_interp[-64:-4])
```

```
    outletVelTot = np.sqrt(outletVelX ** 2 + outletVelY ** 2)

    Energy = []
    rho = 1.24
    cp = 1005
    inletTemp = 5
    #Calculate energy per element on outlet
    for count in range(np.size(outletTemp)):
        Energy.append((outletVelTot[count] * rho * discDistance)\
        * cp * (outletTemp[count] - inletTemp))

    #Calculate final energy
    Energy = np.array(Energy)
    totEnergy = np.sum(Energy)

    allEnergy.append(totEnergy)
    thermalf.close()
    pressuref.close()
    #Save the results
    energyf = open('energyResultsNew.txt', 'a')
    energyf.write(str(totEnergy))
    energyf.write("\n")
    energyf.close()

    allpressuref = open('pressurePerDResults.txt', 'a')
    allpressuref.write(str(allPressures))
    allpressuref.write("\n")
    allpressuref.close()

    allpressuredf = open('pressureDiffPerDResults.txt', 'a')
    allpressuredf.write(str(allPressures[iterationNum][0] -\
    allPressures[iterationNum][1]))
    allpressuredf.write("\n")
    allpressuredf.close()
    #Run Optimization Code
    exec(open("rsmAbsorber.py").read())

#Write final total results
totenergyf = open('energyResults.txt', 'w')

totenergyf.write(str(allEnergy))

totenergyf.close()

totallpressuref = open('pressureResults.txt', 'w')

totallpressuref.write(str(allPressures))
```

```
totallpressuref.close()

totallpressuredf = open('pressureDiffPerDResults.txt', 'w')

totallpressuredf.write(str(allPressures[:][0] - allPressures[:][1]))

totallpressuredf.close()
```

Appendix C

Main Final Optimization Code

```
import numpy as np
import sys
import matplotlib.pyplot as plt
import matplotlib as mpl
from mpl_toolkits import mplot3d
from scipy.interpolate import griddata
from scipy import interpolate
import os
import csv
import pandas as pd
from cfs_utils import*
from pyDOE2 import*
from scipy.optimize import minimize
from scipy.optimize import Bounds
from math import *

energy = np.loadtxt('energyResultsNew.txt')
print(energy)

designPoints = 6
runs = np.size(energy)
combination = 2

maxDistance = 0.025
midDistance = 0.075
dMatrix = np.loadtxt("newdMatrix.txt")

newdValues = np.loadtxt("optResultsNew.txt")
stackdMatrix = np.vstack((dMatrix,newdValues))

#newdMatrix = dMatrix * maxDistance + midDistance


xMatrixCombinations = int(np.math.factorial(designPoints)\
/(np.math.factorial(combination) * np.math.factorial\
```

```
(designPoints - combination)))


xMatrixColums = int(1 + 2*designPoints + xMatrixCombinations)
xMatrixRows = int(runs)


xMatrix = np.zeros((xMatrixRows, xMatrixColums))
print(xMatrix)


for i in range(xMatrixRows):
    xMatrix[i][0] = 1

    for j in range(designPoints):
        xMatrix[i][1+j] = stackdMatrix[i][j]
        xMatrix[i][1+designPoints+j] = stackdMatrix[i][j]**2

    count = 0
    for m in range(designPoints):
        for n in range(m + 1 , designPoints, 1):
            xMatrix[i][2 * designPoints + 1 + count] =\
            stackdMatrix[i][m] * stackdMatrix[i][n]
            count = count + 1



xMatrixTranspose = np.transpose(xMatrix)

xMatrixTransposexMatrix = xMatrixTranspose @ xMatrix

#xMatrixTransposexMatrix = xMatrixTransposexMatrix -\
(np.ones(77 )) * 0.00001

invMatrix = np.linalg.inv(xMatrixTransposexMatrix)

xMatrixTransposeY = xMatrixTranspose @ energy

betaMatrix = invMatrix @ xMatrixTransposeY



def func(X1X2X3X4X5X6):
    X1, X2, X3, X4, X5, X6 = X1X2X3X4X5X6
    return -(betaMatrix[0] + betaMatrix[1]*X1 +\
    betaMatrix[2]*X2 + betaMatrix[3]*X3 + betaMatrix[4]*X4\
    + betaMatrix[5]*X5 + betaMatrix[6]*X6 \
    + betaMatrix[7]*X1**2 + betaMatrix[8]*X2**2 +\
```

```
    betaMatrix[9]*X3**2 + betaMatrix[10]*X4**2 +\
    betaMatrix[11]*X5**2 + betaMatrix[12]*X6**2 \
    + betaMatrix[13]*X1*X2 + betaMatrix[14]*X1*X3\
    + betaMatrix[15]*X1*X4 + betaMatrix[16]*X1*X5\
    + betaMatrix[17]*X1*X6 + betaMatrix[18]*X2*X3\
    + betaMatrix[19]*X2*X4 + betaMatrix[20]*X2*X5\
    + betaMatrix[21]*X2*X6 + betaMatrix[22]*X3*X4\
    + betaMatrix[23]*X3*X5 + betaMatrix[24]*X3*X6 \
    + betaMatrix[25]*X4*X5 + betaMatrix[26]*X4*X6\
    + betaMatrix[27]*X5*X6)



bounds = Bounds([-1, -1, -1, -1, -1, -1], [1, 1, 1, 1, 1, 1])
x0 = np.array([0, 0, 0, 0, 0, 0])

res = minimize(func, x0, bounds=bounds)

file = open("optResultsNew.txt","w").close()

for j in range(np.size(res.x)):
    f = open('optResultsNew.txt', 'a')
    f.write(str(res.x[j]))
    f.write("\n")
    f.close()

ff = open('newdMatrix.txt', 'w')
np.savetxt(ff,stackdMatrix)
ff.close()
```

Bibliography

[1] *5.3.3.6.1. Central Composite Designs (CCD)*. URL: https://www.itl.nist.gov/div898/handbook/pri/section3/pri3361.htm.

[2] Anderson-Cook, R. H. M. D. C. M. C. M. *RESPONSE SURFACE METHODOLOGY Process and Product Optimization Using Designed Experiments*. 2009.

[3] *Build It Solar*. URL: https://www.builditsolar.com/.

[4] *Building the Aluminum Soffit Solar Air Heating Collector*. URL: https://www.builditsolar.com/Experimental/AirColTesting/SoffitCollector/Building.htm.

[5] *Building the Aluminum Window Screen Solar Air Heating Collector*. URL: https://www.builditsolar.com/Experimental/AirColTesting/ScreenCollector/Building.htm.

[6] C.L.GuptaH.P.Garg. "Performance studies on solar air heaters". In: *Solar Energy* (1967).

[7] Choudhury, C. and Garg, H. P. "Performance of air-heating collectors with packed airflow passage". In: *Solar Energy* (1993).

[8] Close, D. J. "Solar air heaters for low and moderate temperature applications". In: *Solar Energy* (1963).

[9] Commission, E. *2050 long-term strategy*. URL: https://www.destatis.de/EN/Press/2021/08/PE21_383_85.html.

[10] Commission, E. *Energy consumption in households*. URL: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Energy_consumption_in_households#Energy_products_used_in_the_residential_sector.

[11] Commission, E. *Household energy consumption for housing continued to increase in 2019*. URL: https://ec.europa.eu/clima/eu-action/climate-strategies-targets/2050-long-term-trategy_en.

[12] Commission, E. *In focus: Reducing the EU's dependence on imported fossil fuels*. URL: https://ec.europa.eu/info/news/focus-reducing-eus-dependence-imported-fossil-fuels-2022-apr-20_en.

[13] Ding, Y. "Adaptive Methods for Convection-Diffusion Equations". In: (2019).

[14] *DIY Solar Air Heating Collectors: Pop Can vs Screen Absorbers*. URL: https://www.builditsolar.com/Experimental/PopCanVsScreen/PopCanVsScreen.htm.

[15] *Downspout Test Collector Construction.* URL: https://www.builditsolar.com/Experimental/AirColTesting/DownSpoutCol/DownSpoutColProto.htm.

[16] E. Glatt S. Rief, e. a. “Structure and pressure drop of real and virtual metal wire meshes”. In: (2009).

[17] *Empty Box Solar Air Heating Collector Performance.* URL: https://www.builditsolar.com/Experimental/AirColTesting/EmptyBoxCol/101217EmptyBox.htm.

[18] *Examining Spatial (Grid) Convergence.* URL: https://www.grc.nasa.gov/www/wind/valid/tutorial/spatconv.html.

[19] Fabian Wein, M. S. “A combined parametric shape optimization and ersatz material approach”. In: (2017).

[20] Georg Pingen, A. E. and Maute, K. “Topology optimization of flow domains using the lattice boltzmann method”. In: *Structural and Multidisciplinary Optimization* (2007), pp. 507–524.

[21] Heubes, D., Bartel, A., and Ehrhardt, M. “An introduction to Lattice-Boltzmann methods”. In: *Novel Trends in Lattice-Boltzmann Methods - Reactive Flow, Physicochemical Transport and Fluid-Structure Interaction* 3 (Jan. 2013), pp. 3–30. DOI: 10.2174/9781608057160113030004.

[22] *Historical and TMY.* URL: https://toolkit.solcast.com.au/historical.

[23] Jose J Corona Jr Osama Mesalhy, e. a. “The best efficiency point of an axial fan at low-pressure conditions”. In: (2021).

[24] K.Rajarajeswari, A. “Matrix solar air heaters – A review”. In: *Renewable and Sustainable Energy Reviews* (2016), pp. 704–712. URL: https://www.sciencedirect.com/science/article/abs/pii/S1364032115015105.

[25] Khuri, A. I. and Mukhopadhyay, S. *Response surface methodology.* Mar. 2010. DOI: 10.1002/wics.73.

[26] M. Ayub, A. C. Z. “A Review of Acoustic Absorption Mechanisms of Nanoscopic Fibres”. In: (2011).

[27] Michael A. A. Spaid, F. R. P. “Lattice boltzmann methods for modeling microscale flow in fibrous porous media”. In: *Physics of Fluids* (1997), pp. 2468–2474.

[28] MINISTRY OF AGRICULTURE, F. and AFFAIRS, R. *Solar Air Heating Systems.* URL: http://www.omafra.gov.on.ca/english/engineer/facts/sol_air.htm.

[29] Mohamad, A. A. *Lattice Boltzmann Method Fundamentals and Engineering Applications with Computer Codes.* Springer-Verlag London Ltd, 2019. ISBN: 978-1-4471-7422-6.

[30] openCFS. *Multiphysics and more based on the finite element method.* URL: https://www.opencfs.org/.

[31] Ratinun Luampon, B. K. “A Study Thermal Efficiency of Solar Air Heater with Wire Mesh Stainless Installation: Using Solar Simulator”. In: (2018).

[32] *Reference Collector Construction.* URL: `https://www.builditsolar.com/Experimental/AirColTesting/BackpassCol/TestingFirstGo.htm`.

[33] Saleh S. Baakeem Saleh A. Bawazeer, A. A. M. "A Novel Approach of Unit Conversion in the Lattice Boltzmann Method". In: (2021).

[34] Satcunanathan, S. and Deonarine, S. "A two-pass solar air heater". In: *Solar Energy* (1973).

[35] Schwer, L. E. "IS YOUR MESH REFINED ENOUGH? Estimating Discretization Error using GCI". In: (2008).

[36] Shams-Ul-Islam Hamid Rahman, W. S. A. "Grid Independence Study of Flow Past a Square Cylinder Using the Multi-Relaxation-Time Lattice Boltzmann Method". In: *International Journal of Mathematical and Computational Sciences* (2014), pp. 1052–1062.

[37] *Solar Air Heating Collector Testing – Which DIY Solar Collectors Perform the Best?* URL: `https://www.builditsolar.com/Experimental/AirColTesting/Index.htm`.

[38] Summers, E. K. "High Efficiency Solar Air Heaters with Novel Built-in Heat Storage for Use in a Humidification-Dehumidification Desalination Cycle". In: (2008).

[39] *Surface Radiation Absorptivity.* URL: `https://www.engineeringtoolbox.com/radiation-surface-absorptivity-d_1805.html`.

[40] Volker John, E. S. "Finite element methods for time-dependent convection–diffusion–reaction equations with small diffusion". In: (2008).

[41] Volker John Petr Knobloch, J. N. "Finite Elements for Scalar Convection-Dominated Equations and Incompressible Flow Problems – a Never Ending Story?" In: ().

[42] Vu, B. N. "Generalization of a Topology Optimization Method Using Lattice Boltzmann Flow Simulation Applicable to Three Spatial Dimensions". In: (2015).

[43] Yuanxun Bill Bao, J. M. "Lattice Boltzmann Method for Fluid Simulations". In: (2011).

[44] YUNUS A.ÇENGEL, A. J. G. *Heat and Mass Transfer Fundamentals Applications.* McGraw-Hill Education, 2015. ISBN: 978-0-07-339818-1.

[45] YUNUS A.ÇENGEL, J. M. *FLUID MECHANICS Fundamentals Applications.* McGraw-Hill, 2014. ISBN: 978-0-07-338032-2.