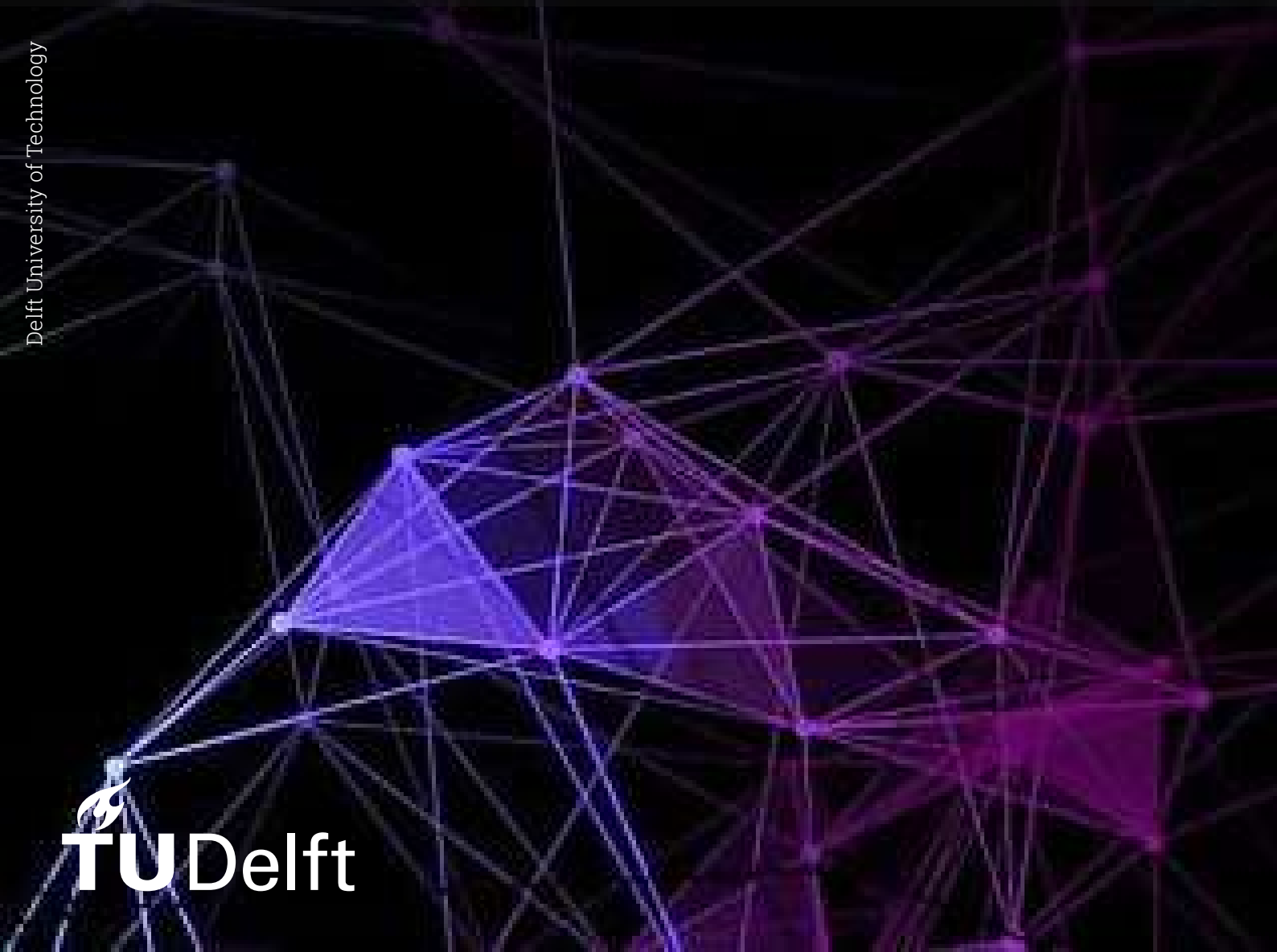
An Integral Approach for Field Reconstruction based on the Helmholtz-Hodge Decomposition

A Proof of Concept for Discrete Data obtained using Flow-Following Temperature Sensors

Medhavi Pandya

Delft University of Technology



TU Delft

An Integral Approach for Field Reconstruction based on the Helmholtz-Hodge Decomposition

A Proof of Concept for Discrete Data obtained using Flow-Following Temperature Sensors

by

Medhavi Pandya

in partial fulfillment of the requirements for the degree of

Master of Science
in Chemical Engineering

at the Delft University of Technology,
to be defended on 18 July 2023 at 14:00.

Student Number:	5459133	
Project duration:	September, 2022 - March, 2023	
Thesis committee:	Dr. Eng. Luis Portela,	Thesis Supervisor, Faculty of Applied Sciences, TU Delft
	Prof. Dr. Ir. Johan Padding,	Faculty of Mechanical, Maritime and Materials Engineering, TU Delft
	Dr. Ir. Cees Haringa,	Faculty of Applied Sciences, TU Delft

Cover: Discrete sensor measurements being reconstructed into a continuous field

TU Delft

Acknowledgements

I want to convey heartfelt gratitude to my thesis supervisor, Dr. Luis Portela, for his constant guidance and support throughout my entire thesis. Working with Dr. Portela has been an incredible learning experience, filled with insightful discussions, light humor, and always a true passion for physics. He has shown a genuine interest during the course of my project and has always been there to help and encourage me as a budding researcher. I deeply appreciate his flexibility and his ability to adapt to his students' needs. It's been a privilege to have him as my supervisor, and I'm truly grateful for this opportunity.

I am thankful and extremely grateful to my beloved parents, their unwavering support and enthusiasm allowed me to pursue my studies at TU Delft. I am also grateful to Nitin and Ninad for their constant care and support during the challenging moments. The past two years have been made all the more memorable and enjoyable by the presence of my friends, I cherish the moments we have shared together. Gratitude to my friends in the TP group, Magda and Ameya, for their critical input on my thesis. I am indebted to Keyur bhai and Krishna who encouraged and helped me immensely while I was trying to balance thesis writing with a new internship.

Lastly, I would like to express my sincere appreciation to Prof. Johan Padding and Dr. Cees Haringa for their valuable contributions as members of my thesis committee. I am grateful for their time, expertise, and thoughtful feedback on my report.

Medhavi Pandya
Delft, July 2023

Summary

Chemical process industries face challenges in monitoring and controlling complex operations. Current 'visualization' techniques require extensive data and computational efforts, which may not always be feasible. Moreover, these techniques often lack detailed insights into the underlying physics. This thesis explores an *a priori* integral approach that can reconstruct scalar or vector fields using sparse sensor measurements. The approach is conceptually built on the mathematical constraints associated with Helmholtz-Hodge Decomposition together with the physical laws. To illustrate this, temperature field reconstructions are considered in steady heat transfer systems, including scenarios with either heat generation or forced convection, using discrete data obtained from flow-following sensors.

A generalized framework is developed using hypothetical heat sources (potentials), with parameters of the heat potentials being determined from the values of the temperature field measured at limited discrete points. Infinitely many reconstructed solutions are possible and the arrangement, population, intensity, and size of the hypothetical heat potentials are the issues of interest that influence the reconstructed solution. Two concrete possibilities are presented for simplification (linearization) by limiting the issues associated with these potentials. The optimal values of unknowns are determined using sparse sensor measurements in a linear system of equations, with the help of 'training'. The framework-assisted reconstructed fields demonstrate accurate predictions of uniform and smooth temperature distribution, while utilizing only a small number of sensor measurements, and minimal computational effort. This validates the effectiveness of an integral approach.

In more complex situations, like locally uneven fields or sharp convective currents, the framework-assisted reconstructions focus primarily on the dominant phenomena and do not capture specific (or, local) characteristics. This highlights the inherent limitations associated with the simplification of a non-linear problem. Potential improvements regarding the treatment of issues associated with heat potentials are suggested for developing a more versatile framework.

The performance of the source framework developed in this work, based on an integral approach, is compared with the Hidden Fluid Mechanics algorithm, a recently developed physics-informed neural network framework, based on a differential approach. This comparison highlights the strength of the source frameworks and the integral approach to 'visualize' simple and smooth domains, in terms of computational expense and accuracy, particularly when dealing with a limited number of measurements. Integrating physics-constrained field functions, developed in this work, into a neural network architecture can present an intriguing avenue for framework optimization. Additionally, gradual enhancements in domain complexity can be explored to expand the applicability of the framework.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

Abbreviation	Definition
BEM	Boundary Element Method
BVP	Boundary Value Problem
CFD	Computational Fluid Dynamics
CNN	Convolutional Neural Network
DNS	Direct Numerical Simulations
DoF	Degrees of Freedom
FEM	Finite Element Method
GSF	Generic Source Framework
HFM	Hidden Fluid Mechanics
HHD	Helmholtz Hodge Decomposition
MAE	Mean Absolute Error
ML	Machine Learning
MRS	Mesh Refinement Study
NS	Navier-Stokes
PINN	Physics-informed Neural Network
POD	Proper Orthogonal Decomposition
PSF	Partition Source Framework
PSF-ES	Partition Source Framework-Exact Solution
PSF-LS	Partition Source Framework-Least Squares
RMSE	Root Mean Square Error
VR	Virtual Reality
VE	Virtual Experiment

Symbols

Symbol	Definition	Unit
T	Temperature	K or $°C$
$\vec{v}$	Velocity	$\frac{m}{s}$
ρ	Fluid density	$\frac{kg}{m^3}$
C	Specific heat capacity	$\frac{J}{kgK}$
k	Thermal conductivity	$\frac{W}{mk}$
$\dot{q}$	Volumetric heat source	$\frac{W}{m^3}$
$\vec{r}$	Position vector	m

Symbol	Definition	Unit
$\lvert\vec{r}\rvert$	The distance between the heat potential and any point P in the domain	m
$D(r)$	Scalar potential function	-
$\overrightarrow{R(r)}$	Vector potential function	-
$H(r)$	Harmonic potential function	-
$\hat{n}$	Surface outward normal	-
R	Radius of Source Potential	m
d	Distance of source plane from the domain	m
m, n	Number of Source Potentials	-
(x, y, z)	Spatial coordinate	m
∇	Nabla operator	-
∇^2	Laplace operator	-
α	Thermal diffusivity	$\frac{m^2}{s}$

1

Introduction to Field Re-Construction

1.1. Introduction

Chemical industries often involve complex processes, with various unit operations and unit processes. Monitoring and controlling such chemical processes is sometimes challenging due to several factors such as non-linearity, fluctuations, field gradients, insufficient data availability, and computational requirements. To overcome these challenges, engineers are leveraging technologies driven by the Fourth Industrial Revolution (Industry 4.0) such as machine learning (ML) based algorithms among other innovations. These technologies are mainly data-driven and rely on large volumes of data to learn/train and make informed decisions. However, when examining intricate physical, biological, or engineering systems, the process of data acquisition is restrictive, leading to a challenge where we must draw conclusions and make decisions based on incomplete information. In such scenarios with limited data, most cutting-edge machine-learning methods lack resilience and cannot ensure convergence [1]. Understanding the behavior of physical fields related to transport processes is a key element in the design and optimization of engineering systems that involve these processes. Therefore, data collection and its relevant treatment for the identification of physical phenomena become a fundamental aspect of research.

The significant revolution in sensor technology has led to the development of flow-following sensors, allowing us to capture spatial information such as pressure, temperature, pH, etc. within the (bio)reactors [2]. These sensors have revolutionized the way data is acquired, providing a new perspective on dynamic processes by enabling variable measurements linked to a particular position and collecting data from multiple locations. Data collected can be used for the identification of the overall flow characteristics, such as circulation time and velocity field, and by analyzing how the process parameters change over time and space [3]. These sensor nodes can retrace their position through mutual communication, without the help of external base stations or beacons [4].

With smart sensor devices capable of generating spatial data, contemporary data-processing techniques are simultaneously under development to maximize the information gained from collected data and transform it into efficacious visualizations. Researchers Garcia *et al.* [5] proposed a technique to

reconstruct the temperature and velocity fields in a natural convection setup via reduced-order approximations. These approximations are expressed in terms of a globally defined basis function using the proper orthogonal decomposition (POD) method, based on the algebraic structure of the finite element method (FEM). While Gong *et al.* [6] suggested a combination of the reduced basis and limited sensor measurements to systematically reconstruct a neutronic field. A convolutional neural network (CNN)-based approach is undertaken by Ponciroli *et al.* [7] and illustrated by Leite *et al.* [8] to reconstruct temperature field at any point in a heated channel driven by an incompressible fluid, from boundary measurements and few measurements within the domain. The researchers formulated their reconstruction problem as a Boundary Value Problem (BVP), for which they implemented Boundary Element Method (BEM) within a CNN framework. Incorporation of BEM into the neural network ensures numerical estimation of Green's function linked to the differential operator, for any arbitrarily shaped domains.

Hidden Fluid Mechanics (HFM), a term coined by Raissi *et al.* [9], is a physics-informed machine learning framework that addresses the challenges associated with the inference of velocity or pressure fields directly from point measurements. This recent approach utilizes a dense point cloud of concentration data from Direct Numerical Simulations (DNS), and Navier-Stokes (NS) equations at once to reconstruct hidden states of the system such as velocity and pressure fields. This method can be viewed as an inverse CFD problem. Notwithstanding, for realistic experimental measurements or numerical simulations, there is limited availability and non-uniform positioning of sensors resulting in an unstructured data grid. A major challenge is the incompatibility of such grids for convolutional neural network (CNN)-based techniques, which are conceptualized using well-organized uniform training data. Further, these methods fail to grasp spatially moving sensors. In response to such issues, Fukami *et al.* [10] recommended using Voronoi tessellation-assisted convolutional neural networks for global field reconstruction. Voronoi tessellation assists in mapping out scattered data onto a structured grid while retaining local sensor information, which then allows utilizing current CNN methods to reconstruct the field without any knowledge of physics. A common feature among the above-presented techniques is the absence of an integral approach to yield global field reconstruction.

1.2. Motivation

In recent research, there is a substantial focus on exploiting machine learning techniques for solving long-standing challenges in the topic of Field Re-Construction. In turn, such techniques rely on the provision of wealthy data or naive ML techniques and lack an integrated approach. Moreover, it is not viable to train these models for all practical situations [8]. In this thesis, the idea is to build a general reconstruction framework with an integrated approach that can utilize a sparse sensor data set measured at random positions within the domain. The reconstruction technique is based on the concepts from Helmholtz-Hodge Decomposition [Chapter 2]. A generalized framework of sources can be developed for which infinitely many reconstructed solutions exist that satisfy the information from sparse sensor data. The situation of choosing the best solution from infinite possibilities is where 'training' will come in the context of our algorithm. It is to be noted that for our technique there is no additional information supplied on boundaries, initial state, or the nature of phenomena occurring within the domain. This global reconstruction technique can be applied to achieve better process control and optimization. At present, it is suitable for real-time analysis of a smooth field but can also be considered an efficient post-processing tool. Such an algorithm is a feasible tool to develop a digital twin model which does not rely solely on ML-based methods. This gives an opportunity to present an interesting case of

performance comparison of our technique with the recent HFM approach [9] in Chapter 7. Figure 1.1 visually represents the general concept behind this thesis. This work is part of ongoing research in the Transport Phenomena group of TU Delft, where a proof of concept is established on a simple steady-state heat conduction problem by [11].

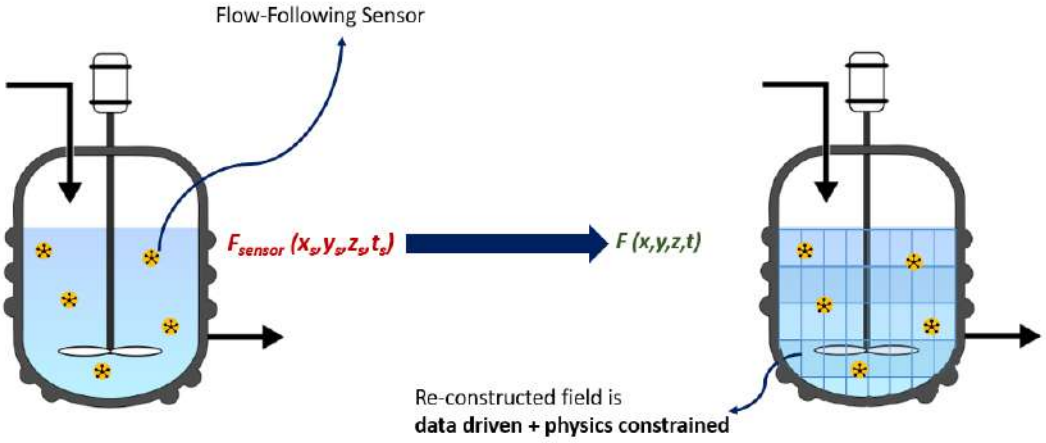


Figure 1.1: General conceptualization of this thesis

1.3. System Simplification and Specific Research Interests

Using the inverse of Helmholtz-Hodge Decomposition for field reconstruction is a yet novel method and while it is under development, taking too many details into account at once can make the research process complicated. For the same reasoning, the domain of study is simplified and assumptions on sensor devices are presented in the next subsections 1.3.1 and 1.3.2, respectively.

1.3.1. Domain of Study

- Field Variable: Temperature considering Scalar Field Reconstruction. Here, temperature is chosen to build upon work done by [11], but any other scalar or vector field can be chosen for reconstruction.
- State of the Study: Steady-state; $\frac{\mathrm{d}T}{\mathrm{d}t} = 0$
- Heat Conduction with
    - **Case1:** Heat Generation within the domain; $\dot{q} \neq 0$
    - **Case2:** Forced Convection with no Heat Generation; $\dot{q} = 0$
- Domain geometry specifications: A 3-D cubical geometry of size $1{\times}1{\times}1m^3$

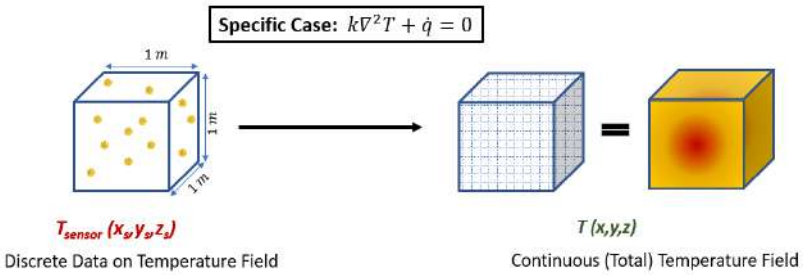


Figure 1.2: Specific objective for this thesis

Figure 1.2 helps to better visualize the domain of study and the objective of this thesis. Specific research questions undertaken for exploration during this thesis are mentioned in subsection 1.3.2.

1.3.2. Assumptions on Sensor Devices

Assumptions on the flow-following sensors are mentioned in [11]. This work continues to follow these assumptions which concern the sensor's operation and measurements. Assumptions are as follows:

1. The sensor provides precise and error-free measurements.
2. The location of the sensor is known.
3. The sensors can move around freely without getting stuck within the area of operation.
4. The sensors are so small that they can be considered neutrally buoyant and treated as a single-point measurement.
5. The presence of the sensor does not have a significant impact on the environment they are measuring.

1.3.3. Thesis Research Questions

This work aims to build a reconstruction framework for the temperature field with either heat generation or forced convection in the domain. Research interests include understanding the effects of localized effects on this framework, estimating the accuracy of the framework with the varying numbers of sensor measurements, and drawing parallels with a recently developed approach. These broad interests can be framed and achieved with help of specific research goals presented as follows:

- Conceptual goals:
    1. Establishing concepts to build a general reconstruction framework and explore different possibilities for the framework development.
- Numerical goals:
    2. Evaluate framework performance across various distribution patterns: smooth variations, locally uneven field, and sharp variations.
    3. Evaluate framework performance under different physical mechanism: heat generation and forced convection.
    4. Determine the minimum requirement of sensor measurements to get a decent field prediction.
    5. Compare the framework-assisted field predictions with the output of the Physics-informed Neural Network [9], using equivalent (sparse) sensor measurements.

These goals are handled systematically in the upcoming chapters. The next section explains the organization of this thesis.

1.4. Thesis Organization

The underlying conceptual background of this thesis is explained thoroughly in Chapter 2, whereas Chapter 3 involves the description of frameworks that are used to optimize the governing field function with a linear approach. Research Goal 1 will be handled in Chapters 2 and 3. Further in Chapter 4, to test the ideas built with previous chapters, sensor and testing data sets are generated by performing virtual experiments. Results pertaining to steady-state heat conduction with heat generation are presented in Chapter 5 while in Chapter 6 results from the case of steady-state heat conduction with forced convection are analyzed. These chapters will handle Research Goals 2 to 4. In Chapter 7, results from the built algorithms are compared with those obtained from the HFM/PINN-based approach [9] using the same sparse sensor data set. This chapter will deal with Research Goal 5. Finally, conclusions drawn from the obtained results are included in Chapter 8 with an outlook for the refinement of the current work and its further development.

2

Conceptual Background

There have been several techniques in use for reconstructing a field as mentioned in the previous chapter, however, all these techniques require some additional input *a posteriori* on the physical phenomena within the domain or at the boundaries which result in the field and variations associated with it. However, the reconstruction ideas applied in this work are postulated with the knowledge of mathematical and physical laws. With this chapter, the aim is to explain these building blocks involved in the development of a field function constrained *a priori*. Such a function would act agnostic to the geometry or any specific domain or boundary conditions.

2.1. Basic Concept behind Field Re-Construction

Field Reconstruction can be viewed as an interpolation technique, wherein a functional value is not directly measured but estimated using the known values. Interpolation can be a useful tool for making predictions or for visualizing data in a more continuous way. Interpolation can be of various types like Linear Interpolation, Spline Interpolation, and Polynomial Interpolation among many other techniques. The nature of the data, the desired level of accuracy, and the specific application determine the correct choice of interpolation method. Considering a variable F varying with distance x. Now, F

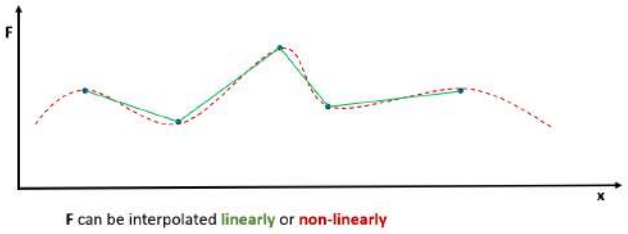


Figure 2.1: Interpolation of a variable F varying with x

can be interpolated linearly or non-linearly with x as shown in Fig2.1. It can be also considered as a combination of basis functions, for example, polynomials of different orders as in Eq. 2.1.

$$F = \sum p^n(x); \tag{2.1}$$

where, $0 \leq n \leq \infty$ and $p^n(x)$ is n^{th} order polynomial in x

While dealing with regular data, these interpolation techniques function as expected but for a data set with underlying physical phenomena, such techniques might not be very sufficient without incorporating insights from physics. For the same reasoning, this work forms the interpolating basis functions based on the theoretical knowledge obtained from physical and mathematical laws. This approach helps in *a priori*-constrained construction of the model field function. Reconsidering the above-presented example in a 1-D space, which can also be visualized as an application of the Fundamental Law of Calculus. The functional value F(x) at intermediate location x is a summation of functional values at the boundaries and the integral of its functional derivatives, as depicted by Eq. 2.2 and illustrated in Fig 2.2.

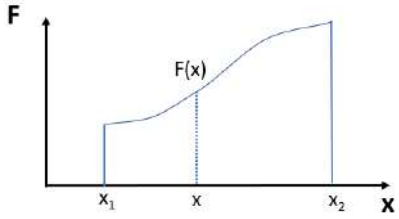


Figure 2.2: Interpolation in 1-D space

$$F(x) = \frac{1}{2}[F(x_1) + F(x_2)] + \frac{1}{2}\left[\int_{x_1}^{x} \frac{dF}{dx}dx + \int_{x}^{x_2} \frac{dF}{dx}dx\right] \tag{2.2}$$

Extending this concept in a 3-D space, a vector $\vec{v}$ is present in a domain of volume V and bounded by a surface S. The vector $\vec{v}$ consists of two components associated with the surface integral of the function and the volume integral of function derivatives. This idea is illustrated in Fig 2.3 and given as Eq. 2.3.

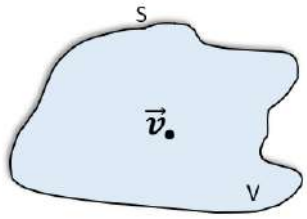


Figure 2.3: Vector in a 3-D space

$$\vec{v} = \int_S Function\ \mathbf{dS} + \int_V Derivatives\ \mathbf{dV} \tag{2.3}$$

To formalize these ideas further in a 3-D space, the concepts presented by Helmholtz and Hodge are referred to in the next section.

2.2. Helmholtz-Hodge Decomposition (HHD)

Helmholtz and Hodge delineated that any vector field ($\vec{v}$), which is continuous and differentiable twice, has domain volume V and is bounded by surface S, can be represented as a three-component field such that:

1. Irrotational or curl-free component (∇D), where D is scalar potential
2. Incompressible or divergence-free component ($\nabla \times \vec{R}$), where $\vec{R}$ is vector potential
3. Harmonic component which is curl-free and divergence-free (∇H), where H is harmonic function

$$\vec{v} = \nabla D + \nabla \times \vec{R} + \nabla H \tag{2.4}$$

Since Component 1 is curl-free, it can be said that

$$\nabla \times \nabla D = 0 \tag{2.5}$$

Similarly, Component 2 is divergence-free and so

$$\nabla \cdot (\nabla \times \vec{R}) = 0 \tag{2.6}$$

While for Component 3 is both curl-free and divergence-free, hence,

$$\nabla \cdot (\nabla H) = 0 \tag{2.7}$$

$$\nabla \times \nabla H = 0 \tag{2.8}$$

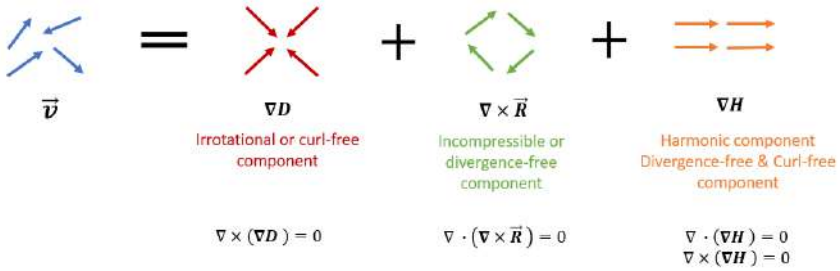


Figure 2.4: Basic concept of Helmholtz-Hodge Decomposition

The scalar potential $D(r)$, vector potential $\vec{R(r)}$, and harmonic function $H(r)$ are expressed with following equations [12]:
Scalar potential D is given as

$$D(r) = -\frac{1}{4\pi} \oint_V \frac{\nabla' \cdot \left(\overrightarrow{v(r')}\right)}{|\mathbf{r} - \mathbf{r}'|} dV' \tag{2.9}$$

Vector potential $\vec{R}$ is given as

$$\overrightarrow{R(r)} = \frac{1}{4\pi} \oint_V \frac{\nabla' \times \left(\overrightarrow{v(r')}\right)}{|\mathbf{r} - \mathbf{r}'|} dV' \tag{2.10}$$

Harmonic function H as

$$H(r) = \frac{1}{4\pi} \oint_S \left(\hat{n}' \cdot \frac{\overrightarrow{v(r')}}{|\mathbf{r} - \mathbf{r}'|} \right) dS' - \frac{1}{4\pi} \oint_S \left(\hat{n}' \times \frac{\overrightarrow{v(r')}}{|\mathbf{r} - \mathbf{r}'|} \right) dS' \tag{2.11}$$

In the above expressions, $\vec{v}$ is the vector field of interest, ∇' is a nabla operator with respect to $\mathbf{r}'$, and $\hat{n}'$ is the surface outward normal. The characteristics that determine these potentials are the surface and volume integrals of the field and field derivatives (curl and divergence), respectively. From Equations 2.9, 2.10 and 2.11, it is also observed that the scalar and vector potentials are congruent with the volume integrals whereas, the harmonic function is associated with the surface integral. Thinking conversely, it can be safely concluded that a distinct vector field can be constructed if its underlying scalar and vector potentials, along with boundary conditions (harmonic potential), are known. If a scalar field needs to be reconstructed, then the scalar (S) is first converted into a vector by taking its gradient. Then, this vector (∇S) can be (re-)constructed using the inverse of the HHD concept.

2.3. Field of Interest

Building upon the work achieved in the previous master's thesis [11], the research is further continued on a scalar field reconstruction, where the scalar field of interest is temperature. Temperature is just an example chosen for developing this conceptual study in a definitive way. However, the theoretical ideas presented in the previous section could be applied to any scalar or vector field. So, moving forward to build a distinct temperature field using the inverse of Helmholtz-Hodge Decomposition, ∇T is considered as the vector. Therefore, the vector field $\vec{v}$ is given as

$$\vec{v} = \nabla T \tag{2.12}$$

Using Eq. 2.12 and substituting in Eq. 2.4,

$$\nabla T = \nabla D + \nabla \times \vec{R} + \nabla H \tag{2.13}$$

Finding curl and divergence of this vector to further estimate its potentials,
Divergence:

$$\nabla \cdot \vec{v} = \nabla \cdot (\nabla T) = \nabla^2 T \tag{2.14}$$

Curl:

$$\nabla \times \vec{v} = \nabla \times (\nabla T) = 0 \tag{2.15}$$

Using Eq 2.12, 2.14 and 2.15 to simplify Eq. 2.9, 2.10 and 2.11 for the field of interest,
Scalar potential D can be given as

$$D(r) = -\frac{1}{4\pi} \oint_V \frac{(\nabla' \cdot \nabla' T)}{|\mathbf{r} - \mathbf{r}'|} dV' = -\frac{1}{4\pi} \oint_V \frac{(\nabla'^2 T(\mathbf{r}'))}{|\mathbf{r} - \mathbf{r}'|} dV' \tag{2.16}$$

Vector potential $\vec{R}$ can be given as

$$\overrightarrow{R(r)} = \frac{1}{4\pi} \oint_V \frac{(\nabla' \times \nabla' T)}{|\mathbf{r} - \mathbf{r}'|} dV' = \mathbf{0} \tag{2.17}$$

Harmonic potential H can be given as

$$H(r) = \frac{1}{4\pi} \oint_S \left(\hat{n}' \cdot \frac{(\nabla' T)}{|\mathbf{r} - \mathbf{r}'|} \right) dS' - \frac{1}{4\pi} \oint_S \left(\hat{n}' \times \frac{(\nabla' T)}{|\mathbf{r} - \mathbf{r}'|} \right) dS' \tag{2.18}$$

Hence, for the chosen field of interest, vector potential is zero which corresponds to the absence of incompressible or rotational part. Eq. 2.14 therefore simplifies to Eq. 2.19

$$\nabla T = \nabla D + \nabla H \tag{2.19}$$

If there is knowledge on potentials D and H, a unique temperature field T can be reconstructed. Further, either of the following two main physical situations can be considered to evaluate these potentials:

1. When $\nabla^2 T = 0$: Steady state with no heat generation or consumption (no change in internal energy) and no flow. In this situation, the scalar potential $D = 0$, Eq. 2.19 then reduces to Eq. 2.20. Temperature T varies solely as a harmonic function and the temperature distribution within the domain is imposed by physics on the boundaries.

$$\nabla T = \nabla H \tag{2.20}$$

2. When $\nabla^2 T \neq 0$:

   i. Steady state with either heat generation/consumption and/or fluid flow.
   ii. Transient state with or without heat generation/consumption and/or fluid flow

   Here, the physical situation corresponds to scalar potential $D \neq 0$, where temperature T varies a combination of scalar and harmonic functions as depicted by Eq 2.19.

Situation 1 ($\nabla^2 T = 0$) was explored by [11]. In this thesis, we extended the research scope and built the concepts further for the case that primarily focuses on Situation 2(i) and the potentials involved in it. The upcoming section illustrates the condition of $\nabla^2 T \neq 0$.

2.4. The System of Study

The physical condition of $\nabla^2 T \neq 0$ is studied as two sub-cases in this work. These sub-cases are 'steady-state heat conduction with heat generation' and 'steady-state heat conduction with forced convection and no heat generation', and are described individually in subsections 2.4.1 and 2.4.2.

2.4.1. Steady-State Heat Conduction with Heat Generation

Starting with the general heat equation assuming no fluid flow in 3D Cartesian coordinates, to interpret the balance between heat being conducted through the fluid/material and any heat sources (heat generation):

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) + \frac{\partial}{\partial z}\left(k\frac{\partial T}{\partial z}\right) + \dot{q} = \rho C\frac{\partial T}{\partial t} \tag{2.21}$$

In vector notation:

$$\nabla \cdot (k\nabla T) + \dot{q} = \rho C\frac{\partial T}{\partial t} \tag{2.22}$$

where,

- T is the temperature [Units: K or °C]
- k is the thermal conductivity [Units: $\frac{W}{mK}$]
- ρ is fluid density [Units: $\frac{kg}{m^3}$]
- C is the specific heat capacity of fluid [Units: $\frac{J}{kgK}$]
- $\dot{q}$ is a heat source term of any form. [Units: $\frac{W}{m^3}$]

Assuming thermal properties remain constant, Eq. 2.22 can also be written as

$$\frac{\partial T}{\partial t} = \alpha\nabla^2 T + \frac{\dot{q}}{\rho C} \tag{2.23}$$

where,

- $\alpha = \frac{k}{\rho C}$ is the thermal diffusivity [Units: $\frac{m^2}{s}$]
- ∇^2 is the Laplace operator given as $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$

For a system in a steady state where the temperature does not change with time ($\frac{\partial T}{\partial t} = 0$), Eq. 2.23 simplifies to

$$k\nabla^2 T + \dot{q} = 0 \iff \nabla^2 T \neq 0 \tag{2.24}$$

2.4.2. Steady-State Heat Conduction with Forced Convection and No Heat Generation

The general energy conservation equation for incompressible fluids, in 3-D Cartesian coordinates (Eq 2.25), describes the conservation of energy in a fluid system by taking into account various transport mechanisms by which energy can be transferred in the system, including conduction, convection, and energy generation while assuming that the fluid and thermal properties remain constant.

$$k(\frac{\partial^2 T}{\partial x^2}) + k(\frac{\partial^2 T}{\partial y^2}) + k(\frac{\partial^2 T}{\partial z^2}) + \dot{q} = \rho C(\frac{\partial T}{\partial t} + v_x\frac{\partial T}{\partial x} + v_y\frac{\partial T}{\partial y} + v_z\frac{\partial T}{\partial z}) \tag{2.25}$$

In vector notation:

$$k(\nabla^2 T) + \dot{q} = \rho C(\frac{\partial T}{\partial t} + \vec{v}\cdot\nabla T) \tag{2.26}$$

where,

- T is the temperature [Units: K or °C]
- k is thermal conductivity [Units: $\frac{W}{mK}$]
- ρ is fluid density [Units: $\frac{kg}{m^3}$]
- C is the specific heat capacity of fluid [Units: $\frac{J}{kgK}$]
- $\dot{q}$ is a heat source term of any form. [Units: $\frac{W}{m^3}$]
- v_x, v_y, v_z are velocity components [Units: $\frac{m}{s}$]

Considering there is no heat generation and the system is in a steady state, after rearrangement, Eq. 2.26 reduces to

$$k\nabla^2 T - \rho C_p(\vec{v}\cdot\nabla T) = 0 \Longleftrightarrow \nabla^2 T \neq 0 \tag{2.27}$$

When the variables in the above equation are non-dimensionalized, Eq. 2.27 transforms to Eq. 2.28:

$$\frac{1}{Pe}(\nabla^2 T^*) - (\vec{v}^*\cdot\nabla T^*) = 0 \tag{2.28}$$

Here, Pe $(= \frac{UL}{\alpha})$ denotes Péclet Number, a non-dimensional quantity that gives a measure of the relative importance of convective transport versus diffusive transport. U denotes flow velocity and L refers to the characteristic length. A high Péclet number means that convective transport is much stronger than diffusive transport, while a low Péclet number means that diffusive transport dominates over convective transport. To sum up, Pe determines whether the heat transfer is dominated by convection or diffusion. Therefore, it can be summarized from the subsections, 2.4.1 and 2.4.2, that $\nabla^2 T \neq 0$ for the physical systems taken into account (Situation 2(i) in Section2.3). The next section brings insights on evaluating the scalar $D(r)$ and harmonic $H(r)$ potentials for the development of a temperature field function.

2.5. Temperature Field Function

A field function refers to a mathematical function that describes the behavior of a physical field related to transport processes, such as heat transfer, mass transfer, or fluid flow. In this thesis, a temperature field function describes the distribution of temperatures within a system. This function can be viewed as a combination of interpolating basis functions which are rooted in the scalar $D(r)$ and harmonic $H(r)$ potentials (Eq. 2.16 and Eq. 2.18).

A temperature field function T can be reconstructed using the Boundary Element Method (BEM), wherein the given boundary conditions are utilized to fit the boundary values into the integral equation. However, in this study, the boundary conditions are unknown. In such cases, a field can be estimated using Convolutional Neural Network as approached in [7], [8]. Alternatively, this study assumes a distribution of hypothetical heat sources (or heat potentials) in and around the domain of interest to evaluate $D(r)$ and $H(r)$. These heat potentials have a uniform size R and unique intensity Q, and their distributed presence influences the conditions within the domain and on its boundaries, resulting in a temperature field. Figure 2.5 illustrates hypothetical heat potentials distributed in and around the domain of interest.

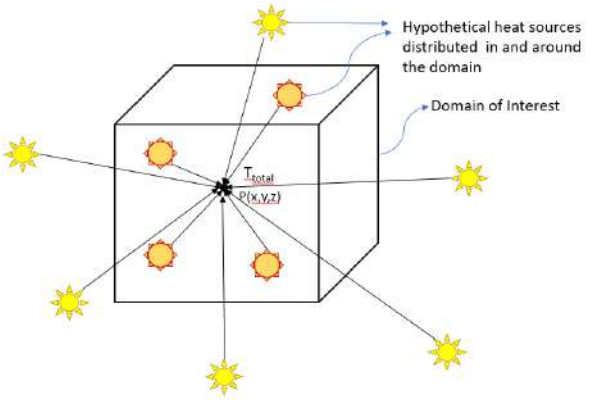


Figure 2.5: Presence of hypothetical heat potentials influencing the temperature distribution within the domain.

The choice of number of heat potentials, associated unique intensity, size and its position determines the basis function. When the distance, $|\vec{r}|$, between a heat potential and any point P in the domain is greater than the size of heat potential R, the heat potential behaves as $H(\vec{r})$. $H(\vec{r})$ determines the boundary conditions that influence the temperature within the domain.

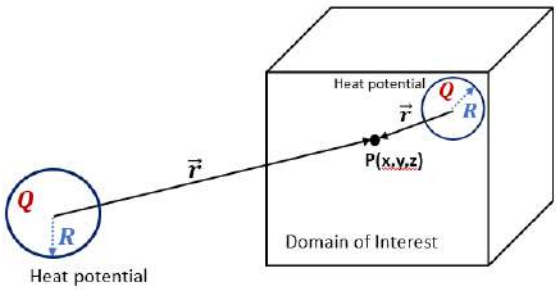


Figure 2.6: Hypothetical heat potential is located such that the distance between the potential and point P in the domain is greater than the radius of the potential.

$$H(\vec{r}) = T(\vec{r}) \propto \frac{1}{|\vec{r}|} = \frac{A}{|\vec{r}|}; \quad |\vec{r}| > R \tag{2.29}$$

where,

- A is a constant denoting the unique characteristics (QR^3) of the heat potential.

- $|\vec{r}|$ is the distance between the heat potential and any point P in the domain.

When the distance, $|\vec{r}|$, between a heat potential and any point P in the domain is less than or equal to the size of heat potential R, the heat potential behaves as $D(\vec{r})$. $D(\vec{r})$ determines the temperature distribution in the domain.

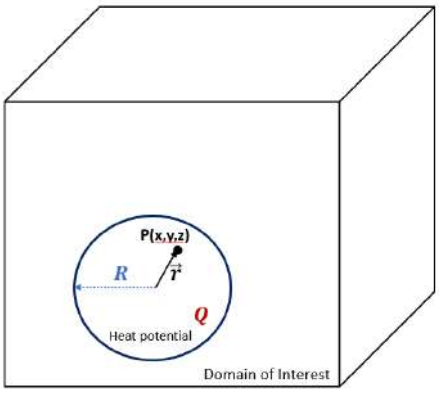


Figure 2.7: Hypothetical heat potential is located such that the distance between the potential and point P in the domain is less than the radius of the potential.

$$D(\vec{r}) = T(\vec{r}) \propto |\vec{r}|^2 = B|\vec{r}|^2; \quad |\vec{r}| \leq R \tag{2.30}$$

where,

- B is a constant which denotes the unique intensity (Q) of the heat potential.
- $|\vec{r}|$ is the distance between the heat potential and any point P in the domain.

There can be infinitely many possibilities on the choice of number of heat potentials, their intensity, size and location which influence the field distribution within the domain. Hence, the temperature at any point in the domain is governed by the presence of multiple heat sources. Eq 2.31 depicts a temperature field function, which is a linear combination of basis functions given in Eq 2.29 and 2.30

$$T(\vec{r}) = \frac{A_1}{|\vec{r}_1|} + \frac{A2}{|\vec{r}_2|} + \cdots + \frac{A_m}{|\vec{r}_m|} + B_1|\vec{r}_1|^2 + B_2|\vec{r}_2|^2 + \cdots + B_n|\vec{r}_n|^2 + T_0 \tag{2.31}$$

where,

- m and n are the number of heat potentials.
- A_i are constants associated with source characteristics $Q_i R_i^3$
- B_i are constants associated with source intensity Q_i
- r_i are the distance between the heat potentials and any point within the domain.
- T_0 is the reference temperature
- T is the temperature at any point in the domain

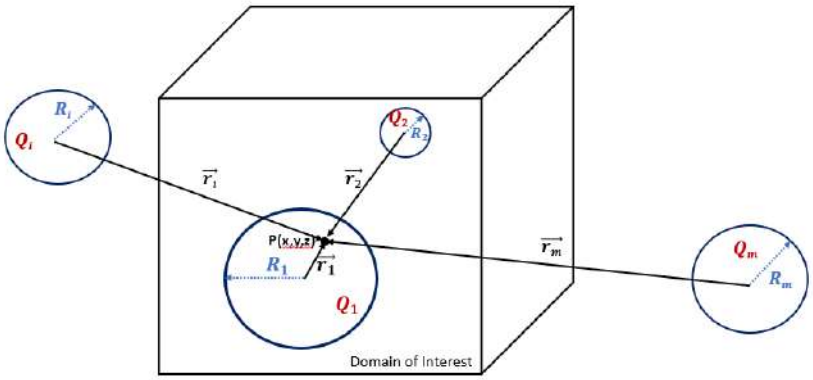


Figure 2.8: 3D visualization of sources in and around the domain of interest.

In 3-D Cartesian space, Eq 2.31 transforms to Eq 2.32

$$T_{(x,y,z)} = \frac{A_1}{\sqrt{(x-x_1)^2+(y-y_1)^2+(z-z_1)^2}} + \cdots + \frac{A_m}{\sqrt{(x-x_m)^2+(y-y_m)^2+(z-z_m)^2}} + B_1[(x-x_1)^2+(y-y_1)^2+(z-z_1)^2] + \cdots + B_n[(x-x_n)^2+(y-y_n)^2+(z-z_n)^2] + T_0 \tag{2.32}$$

where,

- m and n are the number of heat potentials.
- A_i and B_i are constants associated with the unique heat source characteristics mentioned previously.
- (x_1, y_1, z_1), .. , $(x_m, y_m, z_m)/(x_n, y_n, z_n)$ are the location of the heat potentials.
- (x, y, z) is any point in the domain of interest
- $T_{(x,y,z)}$ is the temperature evaluated at point (x, y, z) in the domain

Equation 2.32 is a non-linear temperature field function. This function is influenced by many input variables like the number of heat potentials/sources, their intensities, the size of the potential, and their location in the space. Each unique choice of these variables is a reconstructed solution, and so there are infinitely many reconstruction possibilities. Therefore, it is important to build a logical source framework for simplifying the choice of variables. There can be several linear or non-linear optimization approaches to develop such frameworks, out of which two techniques are discussed in the upcoming chapter. The optimized solution for the field function is determined by minimizing the loss or error in the predicted and true temperature values. This discussion will be elaborated in Chapter 5.

3

Source Frameworks

In the previous chapter, an understanding is developed on the hypothetical discrete heat potentials (heat sources) that influence the temperature field in the domain, as shown in Figure 3.1. This concept provided a platform to build a non-linear field function (Eq 2.32) having many degrees of freedom (DoF) such as the number of heat sources, its intensity, size, and location. This indicates that there could be infinitely many possibilities for field reconstruction. Thus, it is important to simplify the field function in a logical manner by setting certain DoF in Eq 2.32. Source Frameworks assist in this regard. Two out of many possible approaches for developing the source frameworks are discussed in this chapter. The two source frameworks are just a form of optimization technique for framing the DoF associated with the number, size, and location of the hypothetical heat potentials. These frameworks ultimately determine the basis function and thus, the temperature field function for the reconstruction.

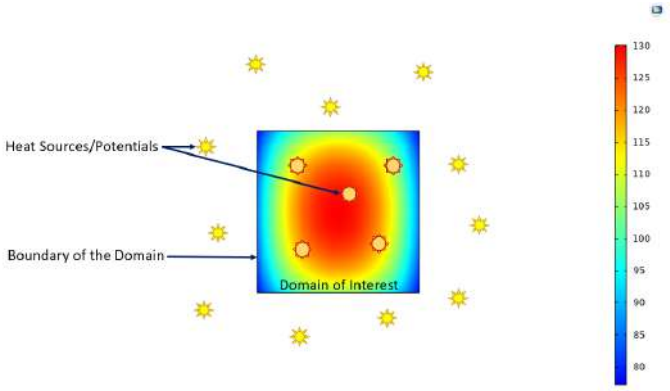


Figure 3.1: Simple representation of heat potentials with respect to the domain of interest.

Henceforth, there are three issues of interest for the source framework development:

- *Issue 1*: The total number of source potentials; $m + n$
- *Issue 2*: The size (radius) of source potential; R
- *Issue 3*: The location of source potential; (x_i, y_i, z_i)

For the issues in interest, two specific approaches are described further for the framework development. These methods distinguish themselves based on their relationship between the size of heat potential and its corresponding position in the space (Link between *Issue 2* and *3*).

1. Generic Source Framework: Heat sources located in the domain have an (infinitely) large R, which is always adequate to contain the domain within the source volume.
2. Partition Source Framework: Heat sources located in the domain have a localized (rigid) R, and may not be always large enough to contain the entire domain within the source volume.

The above-presented frameworks will be explained in detail in the upcoming sections 3.2 and 3.3 with the help of illustrations. These frameworks are tested for their field reconstruction accuracy in Chapter 5, 6, and 7. Prior to explaining different frameworks, it is important to understand the concept of cut-planes and cut-domains, which are discussed in the next section. This concept is shared by both frameworks and deals with *Issues 1* and *3*.

3.1. Cut-Planes and Cut-Domains

Both the Generic Source Framework and the Partition Source Framework share a common approach for Issues 1 and 3 by incorporating the concept of cut-planes and cut-domains illustrated in Figure 3.2. This concept is utilized to define the degrees of freedom (DoF) associated with the number and location (arrangement) of heat sources.

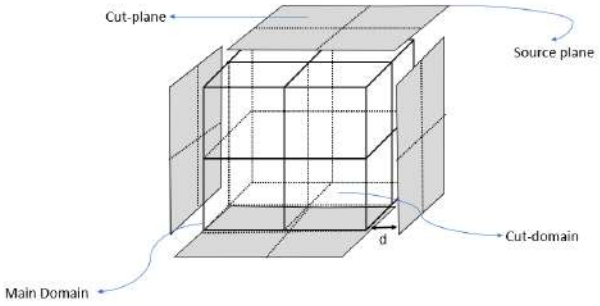


Figure 3.2: Cut-planes and cut-domains.

Cut-Plane: A cut-plane is defined as a plane located parallel to the surface plane of the domain, created by equidistant cuts along each axis of the plane [11]. Cut-planes are collectively known as (external) source planes. Source planes are situated at a distance d perpendicular to the main domain's surface. The heat sources distributed outside the domain are located on the centroid positions of cut-planes. Figure 3.3 visualizes the concept of cuts, cut-planes, and centroid positions. If a $1\times1m^2$ source plane is left uncut (0-cut), it will have only 1 cut-plane (1 centroid), and thus only 1 heat source. When 1-cut is made along each axis of a source plane, 4 cut-planes each of $0.5\times0.5m^2$ will be formed. Therefore, 4 heat sources will be orderly distributed on a source plane. In general, if n cuts are made at equidistant locations on a source plane, then $(n+1)^2$ cut-planes will be formed, each having dimensions of $\frac{1}{(n+1)}\times\frac{1}{(n+1)}m^2$. There will be a total of $(n+1)^2$ heat sources evenly distributed on the source plane lying outside the domain.

Cut-Domain: A cut-domain is a sub-domain created by making equidistant cuts along each surface of the main domain. Figure 3.4 illustrates the concept of cuts and centroid positions on the domain of interest for the arrangement of heat sources that are present within the domain. If a $1{\times}1{\times}1m^3$ domain is left uncut (0-cut), it will have only 1 cut-domain, and thus only 1 heat source. When 1-cut is made along all the planes at their center, 8 cut-domains each of $0.5{\times}0.5{\times}0.5m^3$ will be formed. Therefore, 8 heat sources will be orderly distributed in the domain. In general, if n cuts are made along every plane at equidistant locations, then $(n+1)^3$ cut-domains will be formed, each having dimensions of $\frac{1}{(n+1)} \times \frac{1}{(n+1)} \times \frac{1}{(n+1)} m^3$. There will be a total of $(n+1)^3$ heat sources evenly distributed in the domain.

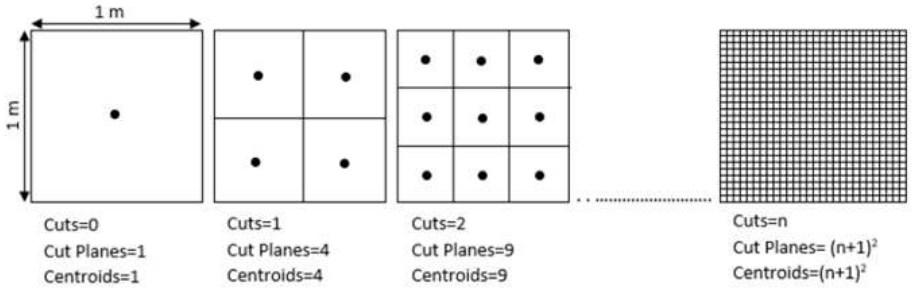


Figure 3.3: Conceptualization of cut-planes with their centroid positions.[11].

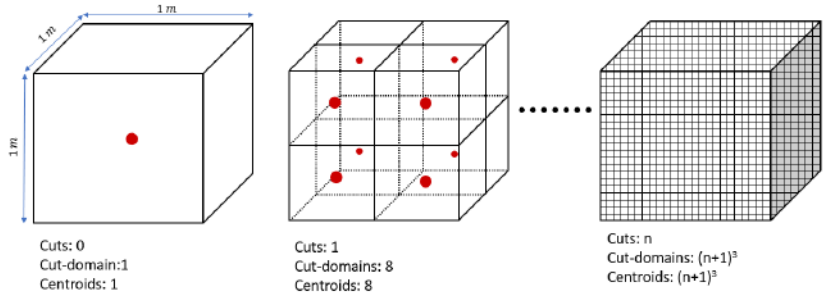


Figure 3.4: Conceptualization of cut-domains and their centroids where the sources will be placed.

Appendix A.3 includes Python scripts that generate centroid coordinates (source locations) on external source planes and in the domain by varying numbers of cuts and the distance of source planes from the domain.

3.2. Generic Source Framework (GSF)

The concept of Generic Source Framework follows from [11], where the overall temperature field was harmonic ($\nabla^2 T = 0$), and discrete hypothetical heat sources were only assumed outside the domain, located on the centroid positions of source planes. The heat sources on the (external) source planes influenced the temperature on boundaries and resulted in a scalar field within the domain. In the current study, the hypothetical heat sources are also assumed inside the domain since $\nabla^2 T \neq 0$. The scalar field is therefore influenced by the boundaries together with the conditions inside the domain. Hence, the concept of cut-domains is introduced in addition to cut-planes. The total number of source potentials (*Issue 1*) and their location (*Issue 3*) can be found by knowing the cuts on the source planes

and domain (0-, 1-, 2-, or 3-cuts), and distance d of the source planes from the domain. All hypothetical heat sources are positioned at the centroids of cut-domains and cut-planes, resulting in temperature distribution within the domain.

Figure 3.5 illustrates the idea of Generic Source Framework. In this figure, we observe external source planes having a 0-cut located at a distance d from the main domain which is also having a 0-cut. To avoid complexity and have a clear understanding, the source planes in the front and back are not included in the schematic but are present in the framework. Each source plane accommodates 1 heat potential. Hence, there are 7 heat potentials in total, out of which 6 are lying on the source planes around the domain and 1 inside the domain.

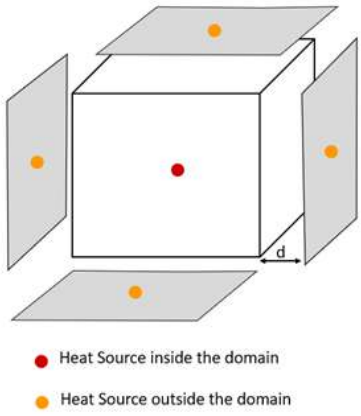


Figure 3.5: Generic Source Framework with 0-cut on external source planes and domain where sources are located at centroids.

While in Figure 3.6, there is 1-cut on the external source planes and on the domain. Each source plane accommodates 4 heat potentials. Hence, there are a total of 32 potentials with $4\times6 = 24$ heat potentials distributed on the external source planes, plus 8 inside the domain. Furthermore, it is possible that the external source plane and domain have dissimilar cuts, i.e., external source planes have a 0-cut but the domain has a 1-cut. In that case, there are 14 heat potentials in total.

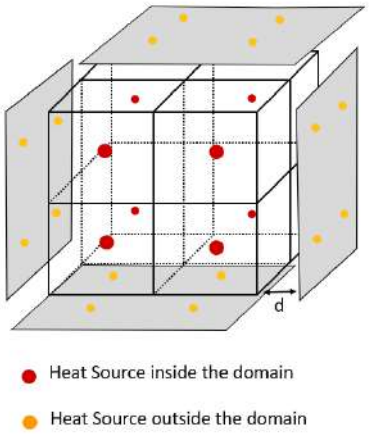


Figure 3.6: Generic Source Framework with 1-cut on external source planes and domain where heat sources located at centroids.

To handle *Issue 2* in this source framework, the size R of every source potential located inside the domain is considered large enough to contain the entire domain within the source volume. While the size R of source potentials located around the domain is considered small enough such that the domain is always outside the source volume. *Issue 2* is linked with *Issue 3* in this regard. This concept is illustrated in Figure 3.7.

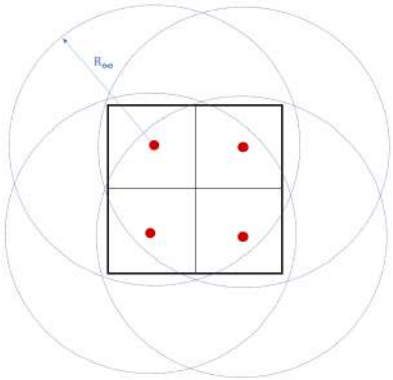


Figure 3.7: Basic idea for *Issue 2* in Generic Source Framework.

Therefore, the source potentials on the external source plane are solved using Eq 2.29 and the sources present in the domain are always treated using Eq 2.30. Together they give Eq 3.1 in Cartesian space for Generic Source Framework. The unknowns in this equation are dependent on the number of cuts on the external source plane, their distance d from the domain, and the cuts on the main domain.

$$T_{(x,y,z)} = \sum_{i=1}^{m} \frac{A_i}{\sqrt{(x-x_i)^2 + (y-y_i)^2 + (z-z_i)^2}} + \sum_{j=1}^{n} B_j[(x-x_j)^2 + (y-y_j)^2 + (z-z_j)^2] + T_0 \tag{3.1}$$

where,

- m and n are the number of sources in cut-planes and cut-domains, respectively.
- A_i denotes the unique characteristics ($Q_i R_i^3$) of the source potential on cut-plane.
- B_i denotes the unique intensity of potential present in cut-domain.
- (x_i, y_i, z_i) and (x_j, y_j, z_j) are the locations of the source potentials.
- T_0 is the reference temperature.
- (x, y, z) is any measurement point in the domain of interest.

For a given number of cuts on source planes and domain, m and n are known. For given m and n, the total number of unknowns will be $m + n + 1$. A global temperature field can be reconstructed once the unknowns (A_i, B_j, and T_0) are determined. In this regard, discrete sensor T measurements made at random locations in the domain will be utilized. To derive an exact solution in a well-determined system, $m + n + 1$ sensor measurements will be required. This will be explained further in section 3.4. Appendix A.4 includes a Python script for solving unknown A_i, B_j, and T_0 with an exact solution for Generic Source Framework.

An inherent disadvantage of this framework could be the 'generic' treatment of *Issue 2*. Due to this, local details in the domain may be disregarded. Hence, there is an opportunity to build a new source framework for the 'specific' treatment of *Issue 2* based on its positioning in space.

3.3. Partition Source Framework (PSF)

Partition Source Framework is similar to Generic Source Framework considering *Issue 1* and *Issue 3*, however, they differ over *Issue 2*. In Partition Source Framework, the size of source potential is rigidly defined. The spherical source potential lying in the domain is assumed to circumscribe around a cubical cut-domain as shown in Figure 3.8. Hence, the size R_0 of the potential is equivalent to the half-length of body-diagonal ($\frac{\sqrt{3}a}{2}$) of the cut-domain having each dimension a. The source potentials in the domain can be partitioned for every measurement point, depending on the source radius R and the distance $|\vec{r}|$ between the source and measurement point. When the distance between a source potential and any point in the domain, $|\vec{r}|$, is greater than the size of the source potential R, the source potential behaves as $H(\vec{r})$ governing the boundaries and is solved using Eq 2.29. The source potential behaves as $D(\vec{r})$ when the distance between a source potential and any point in the domain, $|\vec{r}|$, is less than or equal to the size of the source potential R and is solved using Eq 2.30. While the potentials on the external source planes are considered to be small enough such that the domain is always outside. A configuration with a 0-cut on the domain corresponds to PSF being equivalent to GSF.

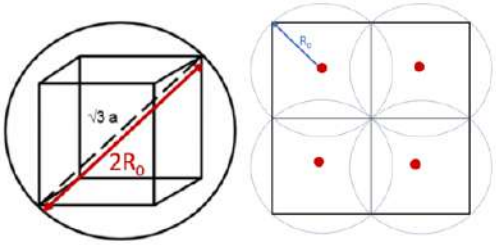


Figure 3.8: Cubical cut-domain of length 'a' is inscribed in a spherical source potential (left) and 2-D representation of the idea for a domain with 1-cut (right).

The 'dynamic' nature of the Partition Source Function is expressed with a general equation (Eq 3.2) for estimating temperature at any point (x, y, z) in the domain.

$$T_{(x,y,z)} = [Q_0 \times F_0((x,y,z),(x_0,y_0,z_0))] + [Q_1 \times F_1((x,y,z),(x_1,y_1,z_1))] + \cdots + [Q_n \times F_n((x,y,z),(x_n,y_n,z_n))] + T_0 \quad (3.2)$$

where,

- (x, y, z) is the any measurement point in the domain
- Q_0, Q_1 ... Q_n are the intensities associated with the source potentials present in and around the domain
- (x_0, y_0, z_0) ... (x_n, y_n, z_n) are location of these source potentials
- $F_i = \eta[(x-x_i)^2 + (y-y_i)^2 + (z-z_i)^2] + \frac{\theta R_i^3}{\sqrt{(x-x_i)^2+(y-y_i)^2+(z-z_i)^2}}$
  - $|\vec{r}| = \sqrt{(x-x_i)^2 + (y-y_i)^2 + (z-z_i)^2}$
  - R is the size of the source potential

i When $|\vec{r}| > R$; $\eta = 0$ and $\theta = 1$
ii When $|\vec{r}| \leq R$; $\eta = 1$ and $\theta = 0$

The concept can be illustrated with simplicity by considering two source potentials, as represented with a 2-D schematic in Figure 3.9. The potentials have sizes R_0 and R_1 and are located at centroid positions. Three measurement points (A, B, and C) are also considered. For source potential of uniform size R_0, point A lies within, and points B, C lie outside the radius of the source. While for source potential with size R_1, point B is within, and points A, C are outside the radius of the source.
Now, knowing

1. Temperature measurements T_A, T_B, and T_C
2. Measurement locations (x_A, y_A, z_A), (x_B, y_B, z_B), and (x_C, y_C, z_C).
3. Size R_0 and R_1, location (x_0, y_0, z_0) and (x_1, y_1, z_1) of source potentials

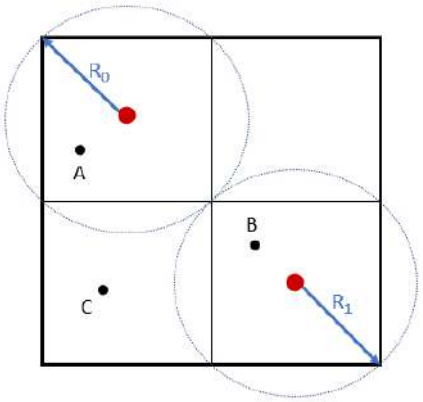


Figure 3.9: Basic idea for *Issue 2* in Partition Source Function.

The temperature at point A can be given by Eq 3.3, at point B by Eq 3.4, and at point C by Eq 3.5.

$$T_A = Q_0[(x_A - x_0)^2 + (y_A - y_0)^2 + (z_A - z_0)^2] + \frac{Q_1 R_1^3}{\sqrt{(x_A - x_1)^2 + (y_A - y_1)^2 + (z_A - z_1)^2}} + T_0 \quad (3.3)$$

$$T_B = \frac{Q_0 R_0^3}{\sqrt{(x_B - x_0)^2 + (y_B - y_0)^2 + (z_B - z_0)^2}} + Q_1[(x_B - x_1)^2 + (y_B - y_1)^2 + (z_B - z_1)^2] + T_0 \quad (3.4)$$

$$T_C = \frac{Q_0 R_0^3}{\sqrt{(x_C - x_0)^2 + (y_C - y_0)^2 + (z_C - z_0)^2}} + \frac{Q_1 R_1^3}{\sqrt{(x_C - x_0)^2 + (y_C - y_0)^2 + (z_C - z_0)^2}} + T_0 \quad (3.5)$$

Hence, the basis function of source potential varies in Partition Source Framework depending on the source radius and location of the measurement point. In the above 3 equations, there are 3 unknowns (Q_0, Q_1, and T_0) which can be solved with an exact solution using matrix inverse multiplication, as given in Python Script A.5. For n heat sources, there will be $n+1$ unknowns, and therefore, $n+1$ sensor measurements will be required to derive an exact solution in a well-determined system. However, an exact solution may not be very robust to the noise built-up (or oscillations) during the interpolation of

larger datasets, when the number of heat potentials is higher in the domain. In such cases, it is rather beneficial to find the best fit than the exact solution. A least-squares solution can be implemented to suppress the noise and find the best fit (approximate solution) by solving under-, well- or over-determined systems. This solution approach can be seen as equivalent to having an additional constraint while solving the system of equations. Refer to Appendix A.5.3 for the Python Script solving unknowns with a least-squares approach.

Partition Source Framework is a modified version of Generic Source Framework that entails a 'specific or rigid' configuration of *Issue 2* in link to *Issue 3*. PSF tailors the heat potentials in the domain to avoid generalization, in an attempt that the local effects within the domain are not neglected. Section 3.2 and 3.3 are aimed to simplify the temperature field function (Eq 2.32) by setting the number, size, and location of the heat sources. This leaves heat source intensity as the only DoF to vary. Again, GSF and PSF are just two of many optimization possibilities for solving the reconstruction problem considered in this study.

3.4. Sensor and True Datasets

3.4.1. Sensor Dataset (Sensor Measurements)

The temperature field at any location in the domain can be estimated once the unknowns in Eq 3.1 (A_i, B_i, and T_0) or in Eq 3.2 (Q_i and T_0) are solved for a given number of cuts and 'd'. Flow-following sensors provide sparse temperature measurements at arbitrary locations within the domain to solve for unknowns. This helps in forming a linear system of equations. The required number of sensor measurements depends on the type of system (under-determined, well-determined, or over-determined) and solution technique used: Exact or Least-Squares.

Following the assumptions in subsection 1.3.2, the measurements from the sensor devices are considered precise, and error-free, and the sensor location is always known. Random sensor measurement positions are generated such that sensor locations remain within the domain of interest. A Python script is given in Appendix A.2 for generating random sensor positions as shown in Figure 3.10. Using these arbitrary positions, the temperature measurements will be drawn out from virtual experiments (presented in the next chapter) and can be viewed as a sensor dataset. Utilizing different sensor datasets for the same task would have a minimal impact on the final results due to the random distribution of the sensors.

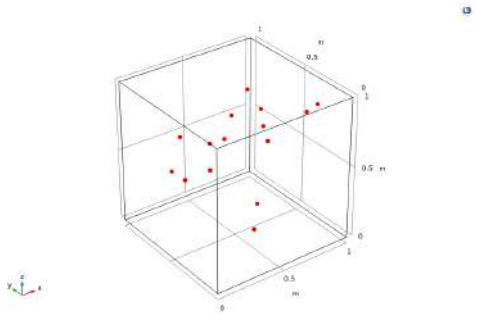


Figure 3.10: 15 Random Sensor Positions.

3.4.2. True-Mapping Dataset (Actual Temperature Values)

The optimal values of unknowns are determined with the help of 'training'. The number of cuts on the domain and source plane and the distance 'd' are varied to find the best combination of basis functions and minimize the loss between predicted temperature and true (actual) temperature values. This is known as the 'training' of the source frameworks. Training a framework fixes three issues involved: the number, size, and location of heat potentials. For this purpose, true temperatures must be known to calculate the error/loss. Actual temperature values are also extracted from the virtual experiments (described in Chapter 4). These values are known as 'True-Mapping Dataset'. The error metrics are presented in Chapter 5.

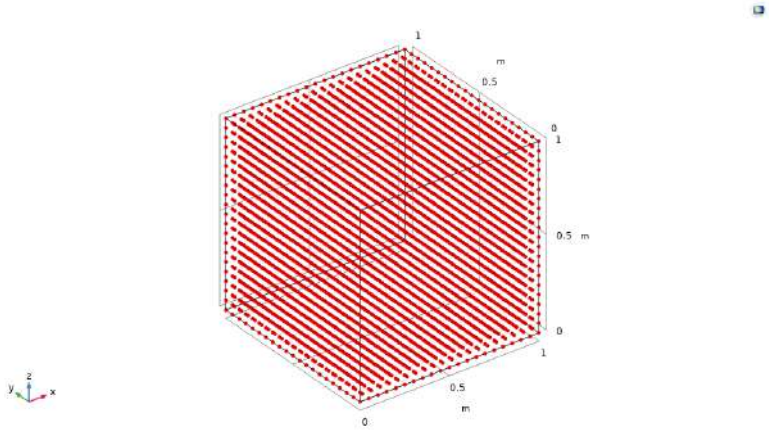


Figure 3.11: Representation of 3-D True Datagrid

For error minimization, the domain is assumed to be evenly spaced with a 3-D array of grid points, similar to [11]. The true and predicted temperature values are assessed on a 21×21×21 grid evenly spaced within the range of [0,1]. While higher data points could be used to represent a finer grid, this was not done to avoid computational complexities. Figure 3.11 shows a 3-D grid where true values and predicted temperature values will be compared. Appendix A.6 includes a Python script for generating 21×21×21 grid points and True-Mapping Dataset. The entire process of the field reconstruction in this thesis is illustrated with the help of Figure 3.12.

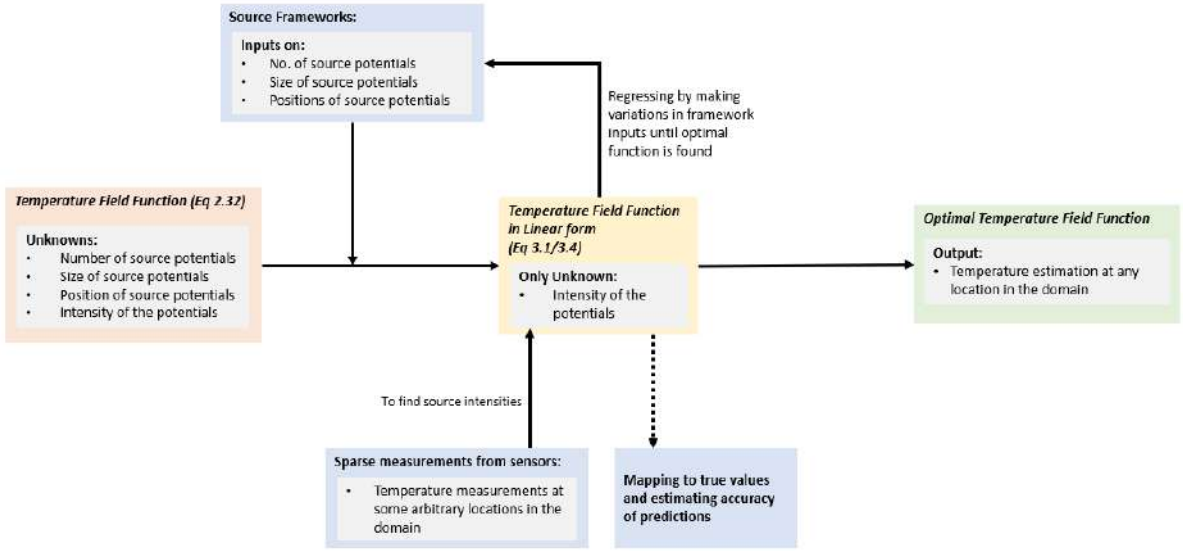


Figure 3.12: Flow diagram illustrating the process of field reconstruction applied in this work.

4

Virtual Experiments

The concepts built in the previous chapters are tested with the help of virtual experiments, which give an expectation of a true (actual) temperature field for comparison. In this chapter, various types of virtual experiments are described which enable gathering random sensor datasets and true datasets. Sensor data is utilized to compute the unknown source intensities. While the true dataset is used to compare with the source framework-predicted values, to train the framework for optimal prediction, and assess the performance of source frameworks.

In principle, the sensor measurements could be collected using physical experiments. However, since the concepts are under development, for the sake of simplicity virtual experiments are conducted. These virtual experiments are performed using COMSOL Multiphysics 5.6 as a numerical solver. Further, any commercial numerical solver could be used. To be consistent and be able to compare situations from previous work, COMSOL was selected for this study. Moreover, the base setup of current virtual experiments is kept the same as Virtual Reality-3 (VR-3) in [11]. The results from VR-3 act as a reference for this thesis to have a fair assessment of the ideas on heat sources built further in the current study. The details on Virtual Reality-3 set-up are given in Appendix B.1.

In this thesis, three cases are considered for setting up the virtual experiments. Virtual Experiment 1 (Section 4.1) deals with smooth temperature variations. It includes two sub-cases, one with a constant heat source (Subsection 4.1.1) and one with a Gaussian source located in the center of the domain (Subsection 4.1.2). Virtual Experiment 2 (Section 4.2) involves localized temperature variations and has a combination of Gaussian heat sources where one of the sources is dominant, generating localized effects towards one edge of the domain. Virtual Experiment 3 (Section 4.3) describes forced convection setups with two fluids, air, and water. Sub-case 3.1 (Subsection 4.3.1) with air has a low Péclet number, having a similar temperature profile to Virtual Reality-3 in [11]. Sub-case 3.2 (Subsection 4.3.2) with water has a high Péclet number, generating sharp temperature variations in the domain.

4.1. Virtual Experiment 1: Smooth Temperature Variations

In this virtual experiment, smooth field variations are achieved with a heat source $\dot{q}$ present in the domain of interest. Heat source $\dot{q}$ is either a constant or varies as a Gaussian function. The surfaces of

the extended domain are given constant temperature values, such that surfaces lying in the same plane have the same temperature, similar to the situation depicted in VR-3 (Refer to Figure B.1). Figure 4.1 encapsulates the set-up configuration with a 3-D schematic.

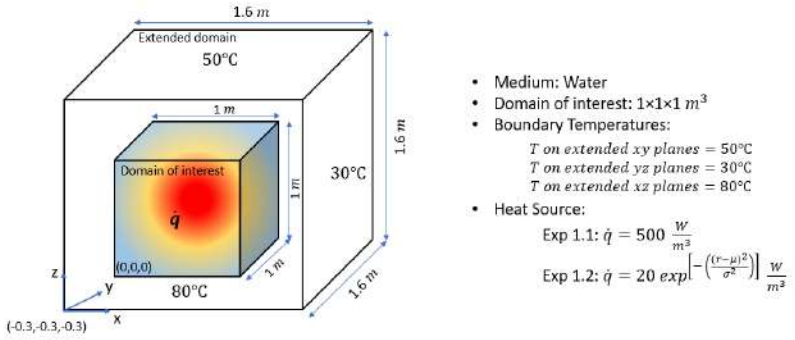
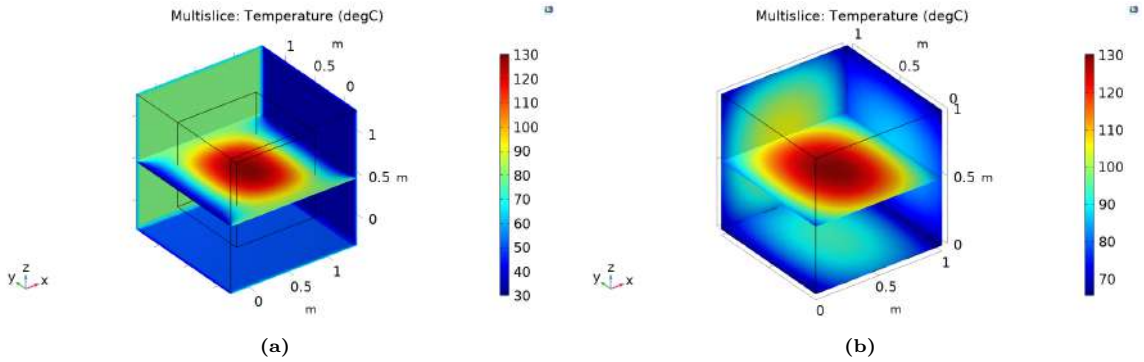


Figure 4.1: 3-D representation of Virtual Experiment 1.1 and 1.2 configurations.

4.1.1. Virtual Experiment 1.1: Constant Heat Source

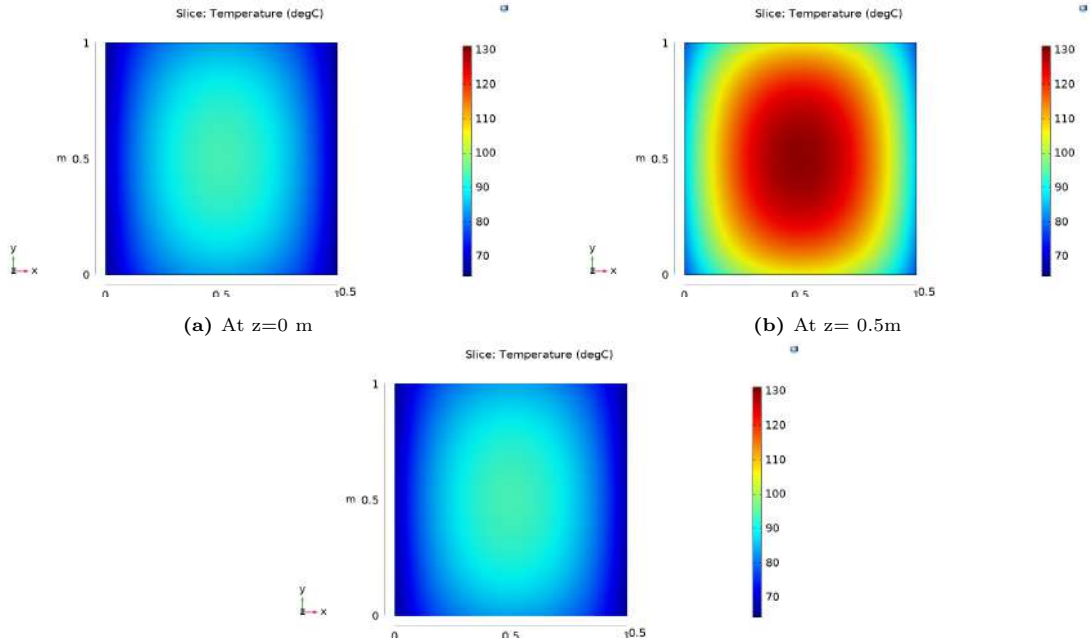
Smooth field variations are attained with the presence of a constant heat source of $\dot{q} = 500\frac{W}{m^3}$ in the domain, and the surfaces of the extended domain are given constant temperature values in a range of 30 to 80°C, such that surfaces lying in the same plane have the same temperature. Figure 4.2 is a 3-D multi-slice temperature plot from COMSOL to summarize the region of interest and region of extended boundary in the virtual environment. Information on COMSOL setup is given in Appendix B.2.



(a) (b)

Figure 4.2: 3-D multi-slice temperature plot of Virtual Experiment-1.1 showcasing (a) domain of interest and region with extended boundaries (b) cropped to the domain of interest.

Source Frameworks-predicted temperature values will be compared to the XY temperature planes cropped to the domain of interest in Virtual Experiment 1.1. The 2-D temperature profiles are presented below in Figure 4.3 and are considered the 'reality' of our domain of interest. The temperature across these planes varies in the range of 70-130°C. The YZ and XZ planes are given in Appendix B.2.3.



(a) At z=0 m

(b) At z= 0.5m

(c) At z=1 m

Figure 4.3: True XY Temperature Plots of Virtual Experiment 1.1

Figure 4.4 depicts a temperature along a line passing through the center of the domain, on the x-axis, at y= 0.5m and z=0.5m, confirming that the temperature variations are indeed smooth where the heat source is present.

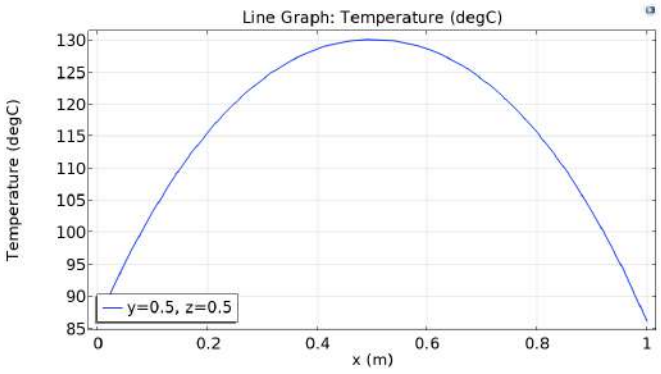


Figure 4.4: Temperature Line Graph for Virtual Experiment 1.1 from x=0 to 1m, at y= 0.5m and z=0.5m.

4.1.2. Virtual Experiment 1.2: Gaussian Heat Source

Temperature variations are kept smooth but now with the presence of a 3D Gaussian heat source varying as $\dot{q} = c \cdot exp[-\frac{(x-x_s)^2+(y-y_s)^2+(z-z_s)^2}{\sigma^2}]\frac{W}{m^3}$. The source is located at the center of the domain with $(x_s, y_s, z_s) = (0.5, 0.5, 0.5)$. Similar to the previous experiment, the surfaces of the extended domain are given constant temperature values in a range of 30 to 80$°C$, such that surfaces lying in the same plane have the same temperature. The characteristic values of the Gaussian function in Virtual Experiment 1.2 are chosen such that the source spans across the domain and temperature values in the domain lie in a similar range as in Virtual Experiment 1.1, Hence, the values are set as follows:

$$c = 20\frac{W}{m^3} \qquad (x_s, y_s, z_s) = (0.5, 0.5, 0.5) \qquad \sigma^2 = 2m^2$$

Figure 4.5 is a 3-D multi-slice temperature plot from COMSOL to give an outline of the domain of interest in the virtual environment. Information on geometry, mesh, and physics setup is given in Appendix B.3.

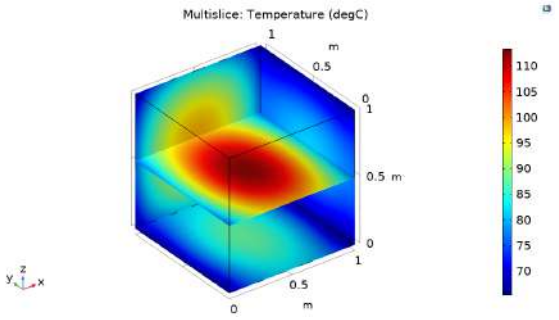
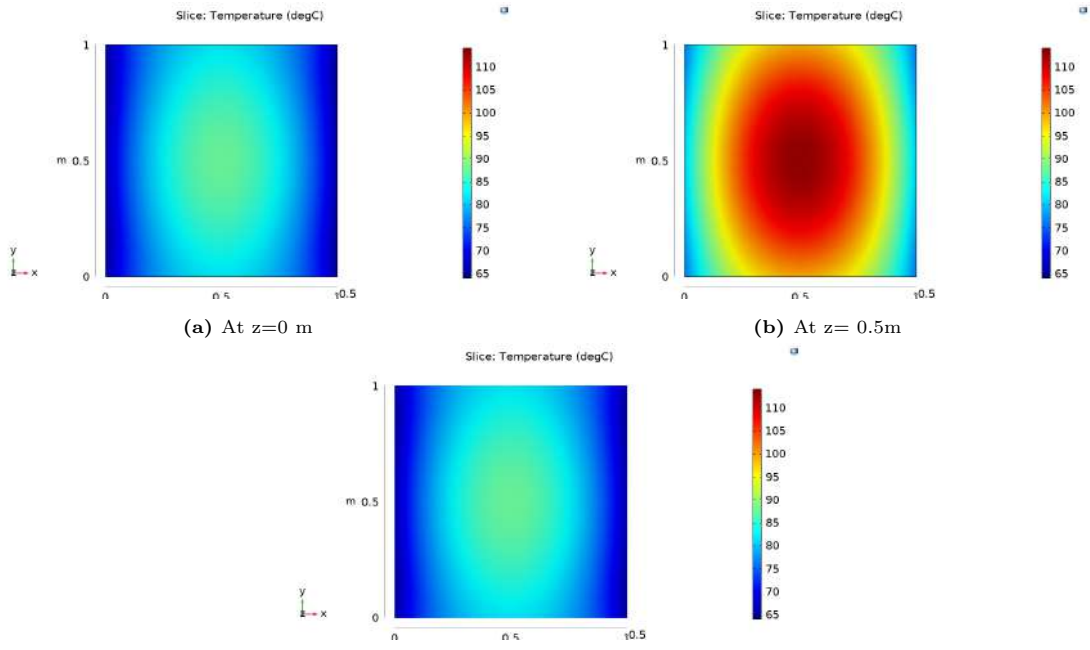


Figure 4.5: 3-D multi-slice temperature plot for the domain of interest in Virtual Experiment 1.2.

Figure 4.6 represents the reality as XY temperature profiles cropped to the domain of interest for Virtual Experiment 1.2. The temperature values vary in the range of 65-110°C. These profiles are utilized for comparison with source framework-predicted temperature values. The YZ and XZ planes are given in Appendix B.3.3.



(a) At z=0 m

(b) At z= 0.5m

(c) At z=1 m

Figure 4.6: True XY Temperature Plots of Virtual Experiment 1.2.

Figure 4.7 represents a temperature line graph along the x-axis, at y=0.5m and z=0.5m. This confirms that a smooth temperature field can exist in the domain even when the heat source is not constant and varies as a Gaussian function that spans across the domain.

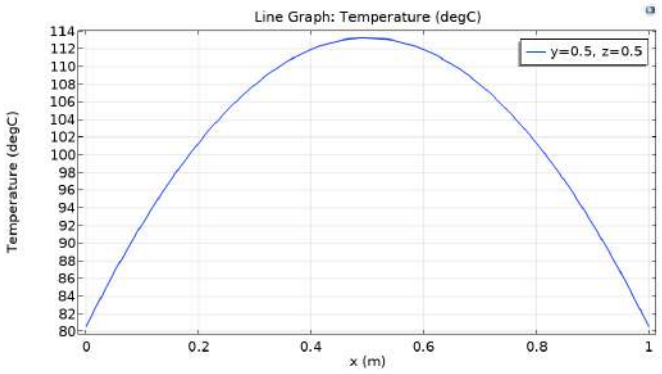


Figure 4.7: Temperature Line Graph for Virtual Experiment 1.2 from x=0 to 1m, at y= 0.5m and z=0.5m.

4.2. Virtual Experiment 2: Localized Temperature Variations

The motive behind this virtual experiment is to test the framework performance in the presence of sudden spikes (discontinuities) in an otherwise smooth temperature distribution. The discontinuities are aroused by an arbitrary combination of two Gaussian heat sources. This combination is additive, such that one of the sources is dominant and generates localized effects towards one of the domain edges. Each heat source in the combination varies as $\dot{q} = c \cdot exp[-\frac{(x-x_s)^2+(y-y_s)^2+(z-z_s)^2}{\sigma^2}]\frac{W}{m^3}$. Figure 4.8 helps to visualize the configuration with a schematic. It is to be noted that this schematic does not correspond to the actual situation in the virtual experiment 2 and is just for illustration purposes.

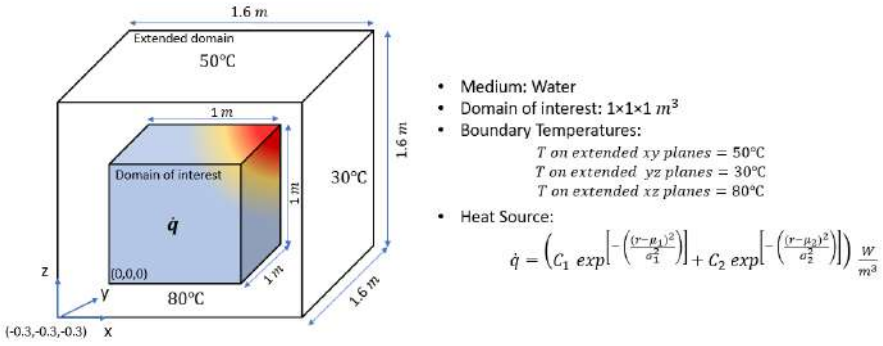


Figure 4.8: 3-D schematic of Virtual Experiment 2 configuration.

The characteristics of two Gaussian functions in Virtual Experiment 2 are as follows:

- **Heat Source 1:** This heat source exhibits a higher intensity and less spread compared to Heat Source 2 such that it generates acute local effects in the system.

$$c_1 = 125\frac{W}{m^3} \qquad \text{Location}(x_{s1}, y_{s1}, z_{s1}) = (0.5, 0.25, 0.75) \qquad \text{Spread}(\sigma_1^2) = 0.0034m^2$$

- **Heat Source 2:** This heat source has lower intensity with higher spread, such that the region is influenced more by the surrounding boundaries rather than its internal characteristics. It results in a comparatively lower impact on the system.

$$c_2 = 25\frac{W}{m^3} \qquad \text{Location}(x_{s2}, y_{s2}, z_{s2}) = (0, 0, 0) \qquad \text{Spread}(\sigma_2^2) = 0.0208m^2$$

Figure 4.9 includes a 3-D multi-slice temperature plot from COMSOL to summarize virtual experiment 2. Figure 4.10 represents the reality as XY temperature profiles for this experiment. These 2-D temperature profiles are cropped to the region of interest. Details on COMSOL setup are given in Appendix B.4. True YZ and XZ temperature planes for this experiment can also be referred to in Appendix B.4.3.

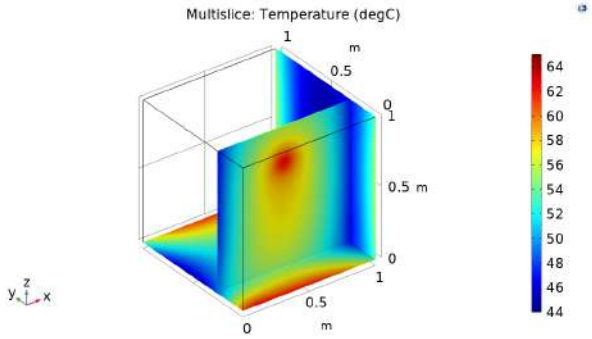
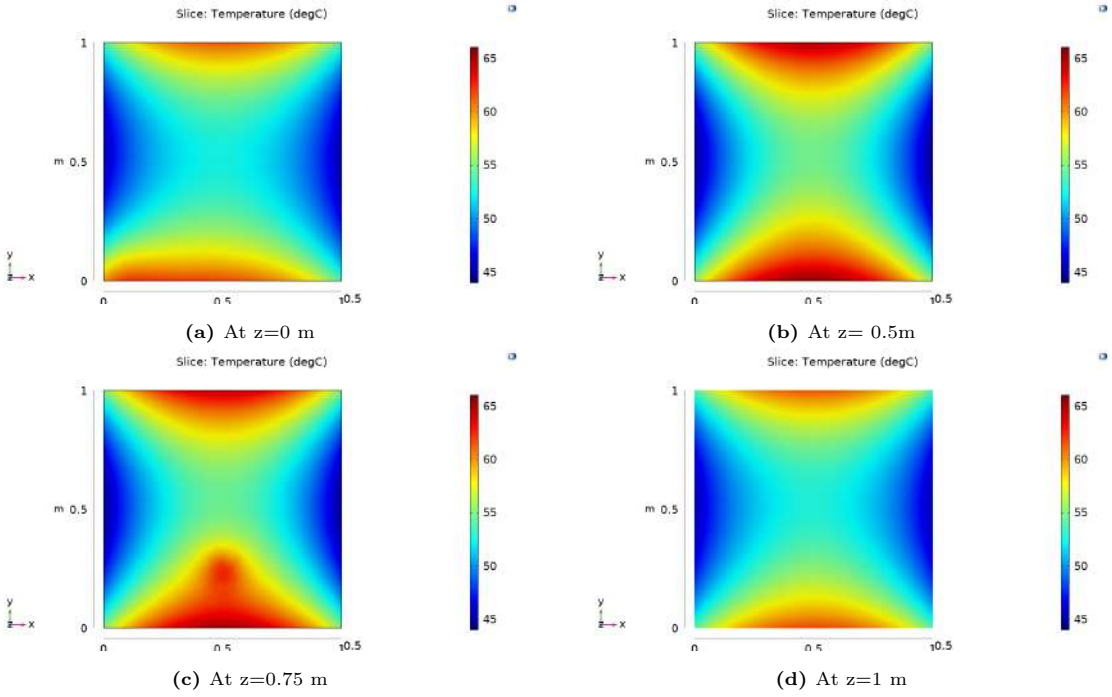


Figure 4.9: 3-D multi-slice temperature plot for the domain of interest in Virtual Experiment 2.



(a) At z=0 m

(b) At z= 0.5m

(c) At z=0.75 m

(d) At z=1 m

Figure 4.10: True XY Temperature Plots of Virtual Experiment 2.

Figures 4.10a, 4.10b, and 4.10d showcase temperature profiles similar to VR-3 from [11] (Appendix B.1). However, when Figure 4.10c is taken into careful consideration, the local effects can be observed. This can be further supplemented by YZ and XZ planes in Appendix B.4.3. A line is considered passing along the z-axis near the center of dominating source, at x= 0.5m and y=0.25m. Temperature variation along this line is plotted as a line graph in Figure 4.11 which confirms the presence of localized effects. Source framework-predicted temperature values will be interesting to compare for this experiment and the results will be elaborated on in Chapter 5.

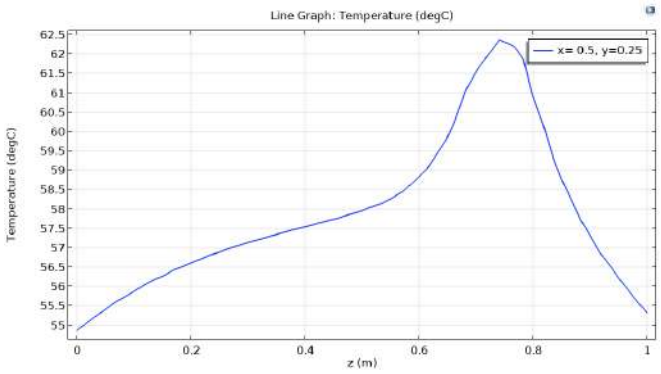


Figure 4.11: Temperature Line Graph for Virtual Experiment 2 from z=0 to 1m, at x=0.5m, y= 0.25m.

4.3. Virtual Experiment 3: Temperature Variations in a Forced Convection Setup

Virtual experiments 3.1 and 3.2 depict the physical setup for forced convection phenomena. These experiments are built for evaluating the concept of "$\rho C(\vec{v} \cdot \nabla T)$ acting as a heat source $\dot{q}_{conv}$" presented in Section 2.5. Two extreme cases based on the Péclet number are considered for the study. Péclet number measures the relative significance of advection versus diffusion; with higher values indicating domination by advection and lower values indicating a conductive/diffusive flow. The velocity is a constant and equal in both cases. This velocity value is chosen such that the flow is laminar. Figure 4.12 encapsulates the setup configuration with a 3-D schematic for visualization purposes.

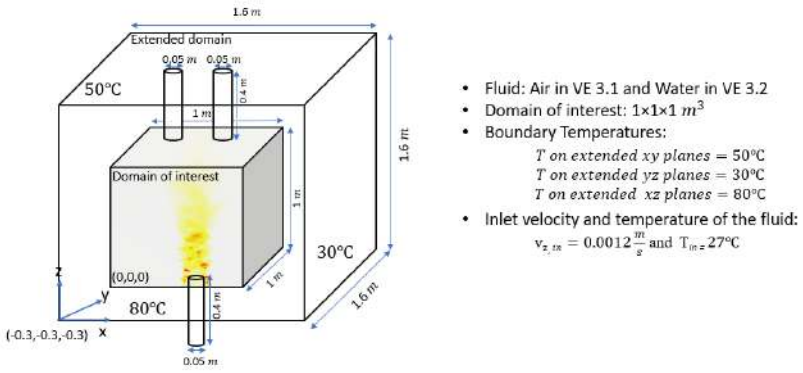


Figure 4.12: 3-D representation of Virtual Experiment 3.1 and 3.2 configurations.

4.3.1. Virtual Experiment 3.1: Forced Convection Setup with Air

Virtual experiment 3.1 setup with air as fluid signifies low Péclet number (Pe $\approx$ 57). This Péclet number is based on the characteristic length of the domain of interest and fluid injection velocity. The velocity value is selected such that the flow is laminar in the domain. This can be supplemented by the XZ velocity plane of the extended domain and the domain of interest in Appendix B.5.3. Furthermore, Appendix B.5.4 includes a plot of $\dot{q}_{conv,air}$ $(= \rho_{air} C_{air}(\vec{v} \cdot \nabla T))$ which shows that most of the region has nearly negligible source intensity effect $(0.008 \frac{W}{m^3})$. The highest (convective) heat source intensity in the domain locally exists near the inlet and is only about $35 \frac{W}{m^3}$. A small negative intensity is observed near the outlets, indicating that as the fluid flows out, it carries away heat from the system. This

negative value signifies the outflow of heat along with the fluid. A system with a low Péclet number indicates that the convective effects are not significant. As a consequence, it can be noticed from Figure 4.14 that the temperature distribution in this experiment is alike to VR-3 from [11] where only heat conduction was considered. Minor field differences observed are at the place of the inlet in the domain. The temperature varies in the range of 45-65°C which is identical to the temperature variations in VR-3. Therefore, the heat transfer in the current domain of interest can be observed to be dominated by the conductive flow.

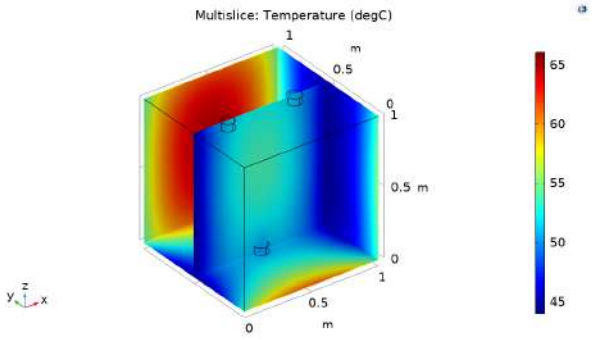
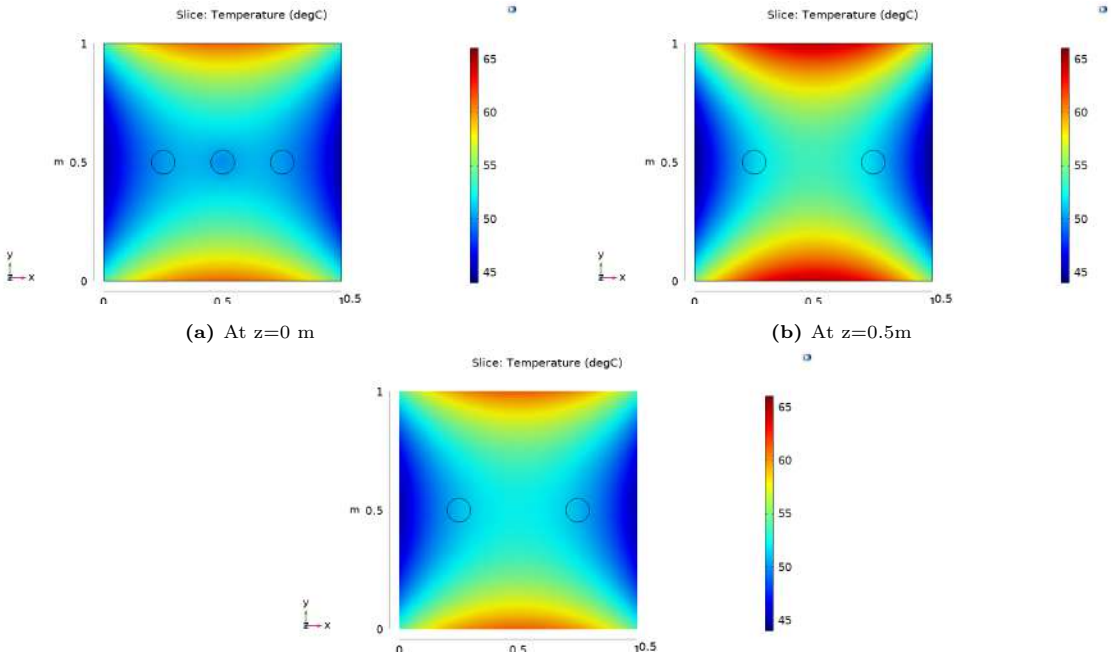


Figure 4.13: 3-D multi-slice temperature plot for the domain of interest in Virtual Experiment 3.1.

Fig 4.13 illustrates a 3D multi-slice temperature plot to summarize the experimental situation. At z=0 in Fig 4.14a, there is a minor temperature variation near the inlet which differentiates Experiment 3.1 from VR-3. Details for setting up Experiment 3.1 in COMSOL are given in Appendix B.5. YZ and XZ temperature planes are presented in Appendix B.5.5. Figure 4.15 showcases the temperature variations in Experiment 3.1 through a line graph along the x-axis just above the inlet, at y=0.5m and z=0.05 m.



(a) At z=0 m

(b) At z=0.5m

(c) At z=1 m

Figure 4.14: True XY Temperature Plots of Virtual Experiment 3.1.

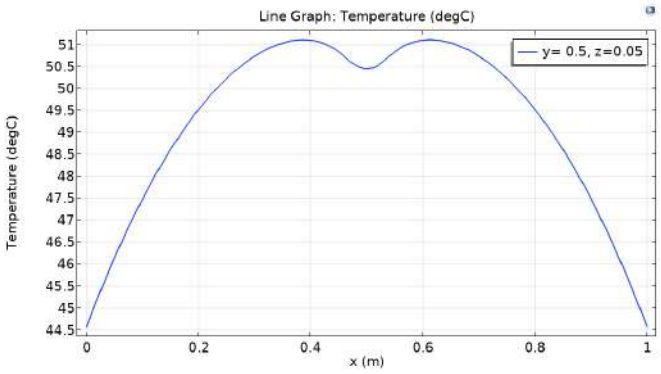


Figure 4.15: Temperature Line Graph for Virtual Experiment 3.1 along the x-axis, at y=0.5m and z=0.05m.

4.3.2. Virtual Experiment 3.2: Forced Convection Setup with Water

Virtual experiment 3.2 setup with water as fluid signifies a high Péclet number (Pe $\approx$ 8200). Similar to the previous experiment, this Péclet number is based on the characteristic length of the domain of interest and fluid injection velocity. The flow is laminar in the domain, which is supplemented by the XZ velocity plot of the domain of interest in Appendix B.6.3. The difference between the Péclet number in the two forced convection cases is observed in their velocity plots.

Similar to VE 3.1, there is a plot of $\dot{q}_{conv,water}$ $(= \rho_{water} C_{water}(\vec{v} \cdot \nabla T))$ in Appendix B.6.4 which shows that the highest (convective) heat source intensity in the domain is in the order of $10^5 \frac{W}{m^3}$ which is found near the inlet. This source intensity is substantially higher than in the previous experiments. Moreover, there is a negative value of the same order signifying the outflow of heat along with the exit of the fluid from the outlet of the domain. The average convective source intensity in the domain is about $45.5 \frac{W}{m^3}$. A high Péclet number indicates the higher order of incoming and outgoing intensity ($\dot{q}_{conv}$) explaining the dominance of convection over conduction and the resulting sharper variations observed in Fig 4.16.

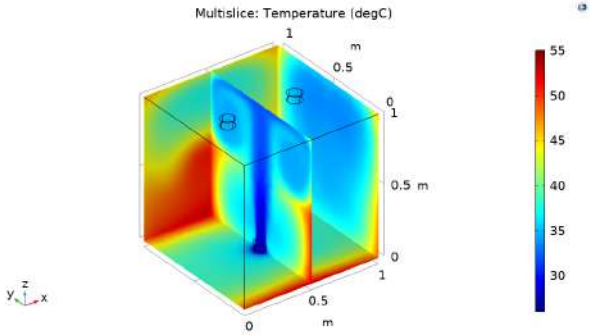
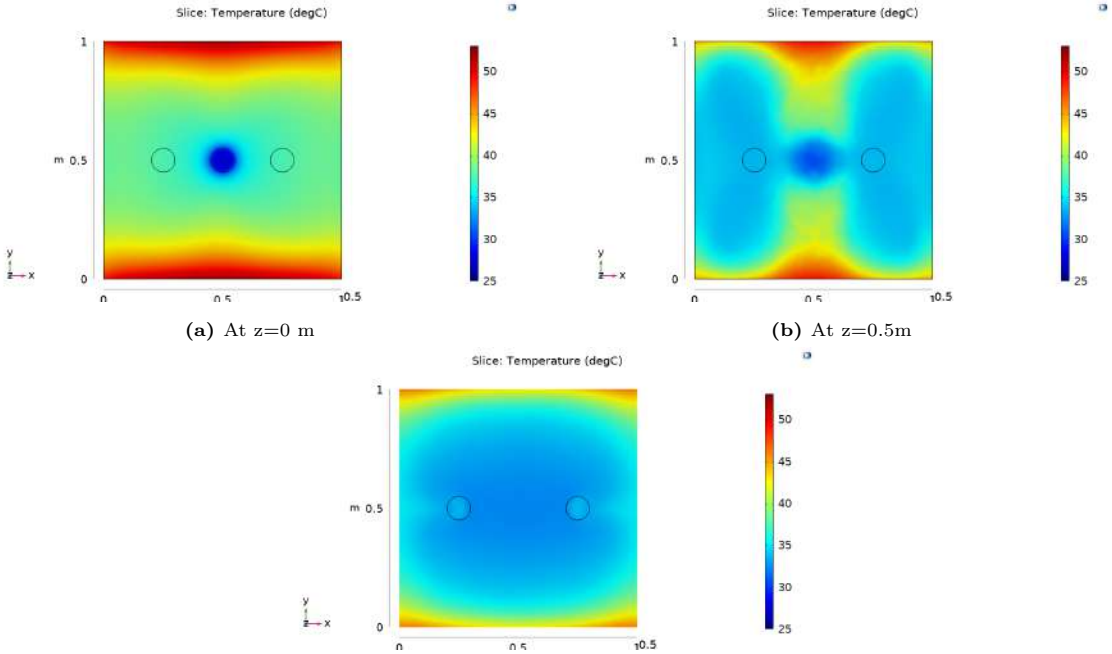


Figure 4.16: 3-D multi-slice temperature plot for the domain of interest in Virtual Experiment 3.2

The XY temperature planes are presented in Fig 4.17, where the temperature varies in the range of 25-55$°C$ but rather intensely. Separate YZ and XZ temperature planes for this experiment can be referred to in Appendix B.6.5. The temperature planes are cropped to the region of interest. Furthermore, Fig 4.18 illustrates the temperature variations along the y-axis near the outlets, at x=0.5m and z= 0.9m. It can be observed from this plot that temperature changes are far from being smooth, moreover, there is a sudden jump of nearly 3.5$°C$ as we move toward the boundaries. Details for setting up Experiment

3.2 in COMSOL are given in Appendix B.6.



(a) At z=0 m

(b) At z=0.5m

(c) At z=1 m

Figure 4.17: True XY Temperature Plots of Virtual Experiment 3.2.

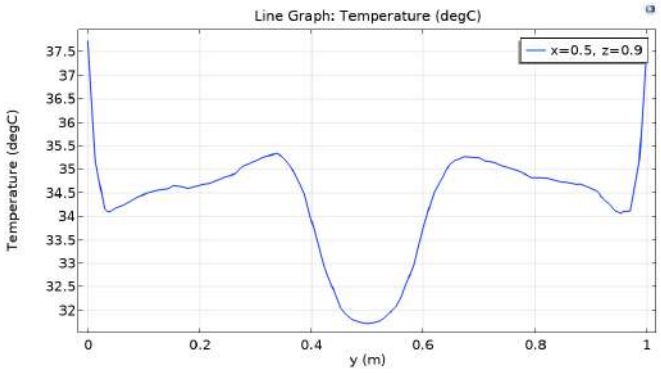


Figure 4.18: Temperature Line Graph for Virtual Experiment 3.2 along the y-axis, at x=0.5m and z=0.9m.

5

Results: Virtual Experiment 1 and 2

The aim of this chapter is to test and evaluate the performance of source frameworks in predicting temperature. The frameworks are based on Equations 3.1/3.2, which involve the selection of three issues of interest. These include the total number of source potentials, their size (radius), and their location in the space. The issues can be adjusted via a variety of cuts on the domain and on the source planes arranged around the domain. The only 'unknowns' in the equations are the source intensities, which are evaluated with the help of random sensor data in an exact or a least-squares solution technique. Once the unknowns are solved, the equations can make temperature predictions at any point lying in the domain. These resulting predictions are then compared to the actual values obtained from Virtual Experiments using COMSOL. In this chapter, we assess the source framework performance for Virtual Experiments 1 and 2. To provide context, Virtual Experiments 1 and 2 both involved the presence of heat source(s), with the nature of the source being either constant or Gaussian. Experiments 1.1 and 1.2 showed that the resulting temperature field variations were smooth and uniform throughout the domain. However, in Virtual Experiment 2, a sudden temperature spike occurred in an otherwise smooth field due to the dominant effects of a local heat source. To ensure a fair comparison of performance among all cases, both the actual (true) and predicted temperature values are normalized in the following manner:

$$T_{nd} = \frac{(T - T_{min})}{(T_{max} - T_{min})} \tag{5.1}$$

The performance of the source frameworks can be evaluated using conventional error metrics like Mean Absolute Error (MAE) or Root Mean Square Error (RMSE). Mean Absolute Error (MAE) represents the average absolute difference between the actual values and the model-predicted values, while RMSE represents the square root of the average of the squared differences between the predicted and actual values. However, when the data is normalized, MAE is generally considered a superior error metric compared to RMSE. The reason is that RMSE is more sensitive to outliers than MAE. Since the data is normalized, the range of the data is limited and outliers are less likely to occur. Therefore, the added sensitivity of RMSE to outliers is not necessary and can lead to over-penalization of the model's performance. For this work, Mean Absolute Error (MAE) is calculated using Eq 5.2. It will provide a measure of how far the predictions deviate from the actual output. A low MAE value is desired for overall good predictive capabilities. However, it is important to note that MAE, being an average measure, may mask localized errors within the domain. Although the overall MAE value may be low,

specific regions may exhibit higher errors than the average. To ensure comprehensive analysis in this study, absolute error plots are employed alongside MAE values to identify and assess these localized errors accurately.

$$MAE = \frac{1}{N}\sum_{i=0}^{N} |T_{r,i} - T_{p,i}| \tag{5.2}$$

where,

- $T_{r,i}$ is the true or actual value at a point i
- $T_{p,i}$ is the model-predicted value at a point i
- N is the total number of points where the absolute error is calculated, which corresponds to the true grid points mentioned in subsection 3.4.2.

5.1. Results: Virtual Experiment 1.1 - Constant Heat Source

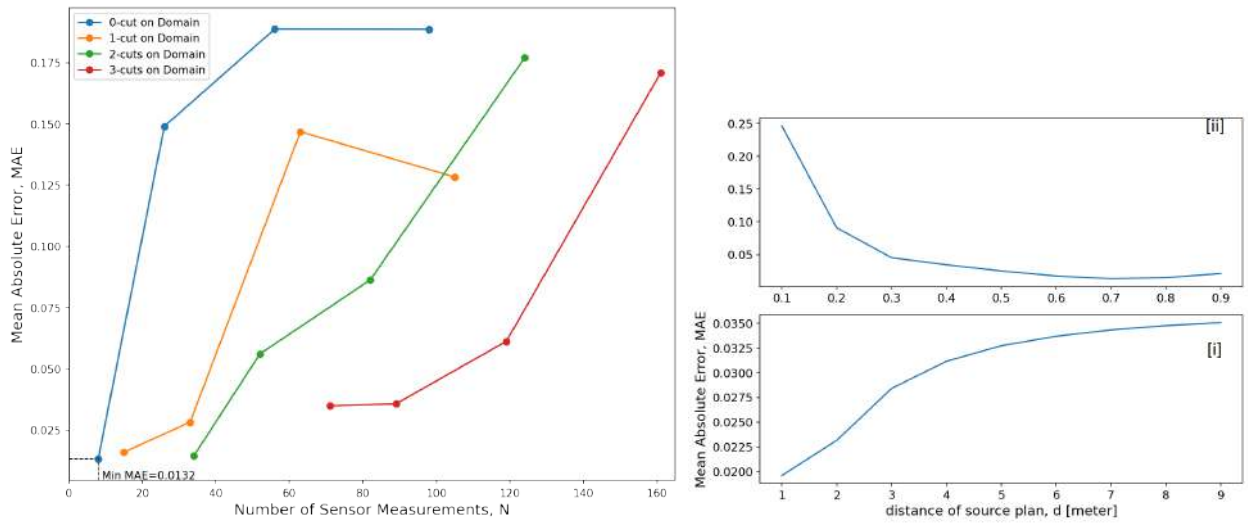
In this experiment, smooth field variations are attained with the presence of a constant heat source of $\dot{q} = 500\frac{W}{m^3}$ in the domain, the surfaces of the extended domain are given constant temperature values in the range of 30 to 80°C, such that surfaces lying in the same plane have the same temperature.

5.1.1. Field Predictions using Generic Source Framework

The optimal values of source intensities in Eq 3.1 of Generic Framework are achieved with the help of 'training', as described in subsection 3.4.2. The Framework is trained by varying its characteristics: the number of cuts on the domain, the cuts on external source planes, and the distance 'd' of these source planes. Every combination of these values specifies a distinct set of the source potentials present in the framework and the basis function associated with them. This results in a unique temperature field function giving a distinct reconstructed solution. It might not be the finest reconstruction, but it is still a solution that satisfies a particular sensor dataset. The optimum characteristic values (giving optimum reconstruction) are chosen where the MAE value across the domain is the lowest.

For an exact solution, the required number of sensor measurements increases with every increase in a cut. For a systematic increase, first, a cut on domain surfaces is fixed, and then cuts on source planes and their distance from the domain are varied. The changes in the MAE of predicted values are observed across the entire domain (MAE), as shown in Figure 5.1a. The source plane distance 'd' is incrementally adjusted, between 1 to 9m with a step size of 1m (Fig 5.1b[i]), to assess the optimal plane location. Fig 5.1b[ii] zooms in between 0.1 to 1m with a step size of 0.1m, similar to the approach followed by [11].

In Figure 5.1a, each colored line indicates the number of cuts on the domain. Whereas, round marking on a line indicates cuts on the external plane in increasing order from bottom to top (as in, 0, 1, 2, and 3 cuts). It is evident from this figure that with the least number of sensor measurements, it is possible to achieve minimum MAE across the domain. For this case, as observed from Fig 5.1, with just 8 sensor measurements (i.e., a combination of 0-cut on the domain surfaces and 0-cut on source planes, and d=0.7m), a minimum MAE of 0.013 is achieved. We can also observe that for a given cut on the domain, increasing cuts on the external source planes increases the MAE. The possible reason for this increase could be the over-fitting of the data [11]. Further, we can notice that increasing the number of cuts on the domain also does not improve the error.

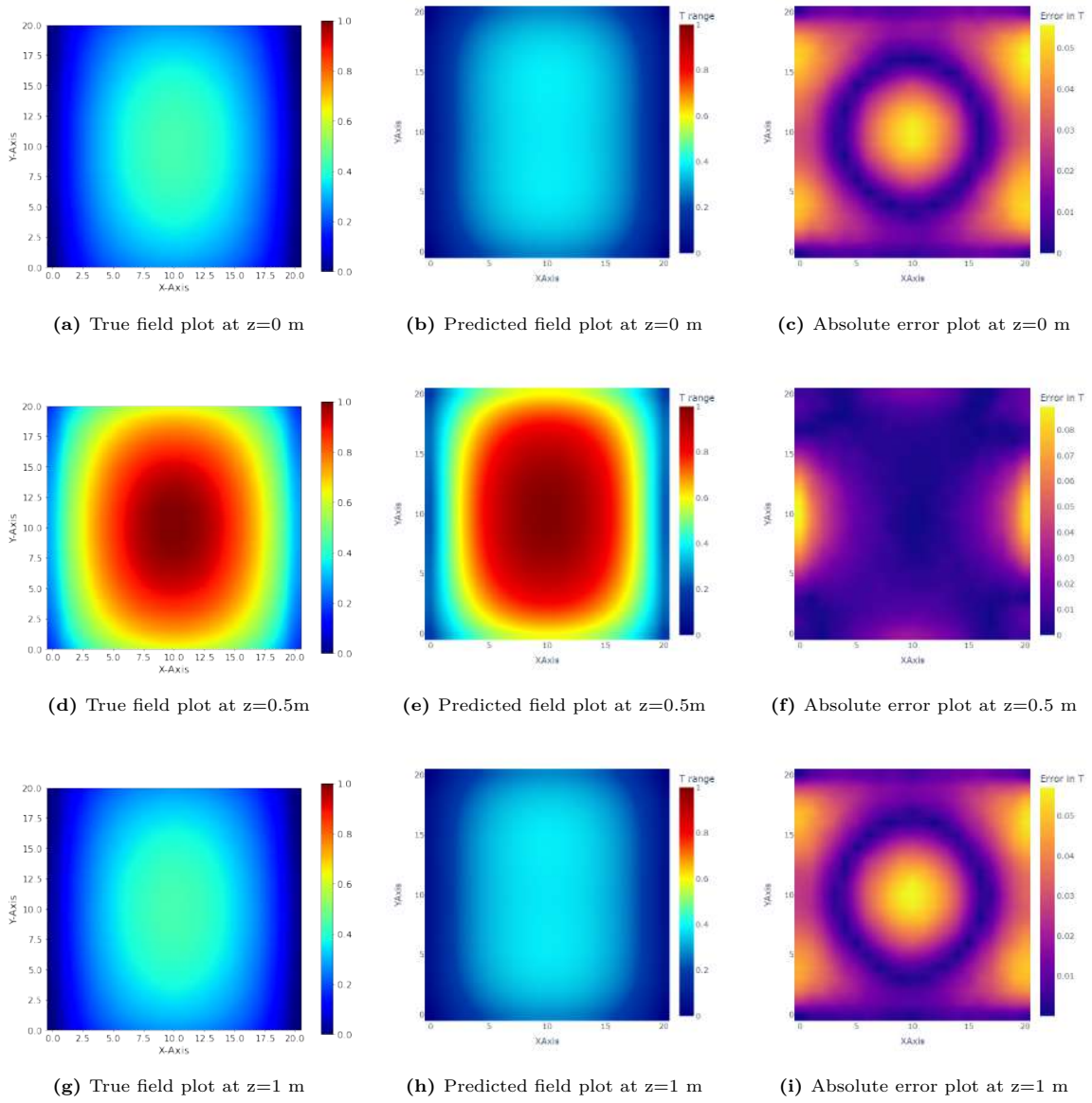


(a) (b)

Figure 5.1: Optimal GSF settings for Virtual Experiment 1.1 (a)A plot of MAE with the varying number of cuts on the domain and source planes (equivalently varying Sensor Measurements). (b)A plot of MAE with varying source-planes distances 'd' for 0-cut on the source-planes and domain; [i] between 1 to 10m [ii] zoomed in between 0.1 to 1m.

Figure 5.2 represents the true and reconstructed normalized temperature plots. Alongside, respective absolute error plots $|T_{actual} - T_{predicted}|$ on an XY plane are demonstrated. True values from the virtual experiment are extracted as a grid of (21 x 21 x 21) points, described in subsection 3.4.2, to give an expectation for the reconstructed field. The length of 1 m in COMSOL plots is now discretized with a grid size of 0.05 m, meaning that x=20.0 in the true plot represents the temperature at x=1m in reality. The predictions are also evaluated on the same grid points. Due to normalization (Eq 5.1), the temperature range [65, 130]°C observed in Figure 4.3 is now transformed to the range [0, 1]. A Python script written for generating these plots can be referred to in Appendix A.6.3, A.7.3, and A.8.5.

As we can observe from these plots, Generic Source Framework is able to re-construct the field with a minimum requirement of sensor measurements. It is capable of identifying smooth temperature trends sufficiently well. This general observation is in line with the VR-3 results from [11], where smooth temperature variations were considered in the absence of a heat source. In Figure 5.2f, at z=0.5m we can also observe the error is almost negligible where the heat source strength is strongest. However, while going towards the domain surface (boundaries), predictions suffer a higher error in the range of [0.05-0.08], also supported by Figures 5.2c and 5.2i at z=0 and 1m. The error near the boundaries is relatively higher in the presence of a heat source compared to the case without a heat source in VR-3 (Refer to Appendix C.1). This observation suggests a potential conflict arising from the imposition of boundary conditions from external source planes and the sources within the domain. These source potentials yield many different solutions depending on the basis function (Eq 2.29 or 2.30), their population, size, and location. However, these solutions may not be strongly interconnected, leading to a higher error near the boundaries.



(a) True field plot at z=0 m **(b)** Predicted field plot at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field plot at z=0.5m **(e)** Predicted field plot at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field plot at z=1 m **(h)** Predicted field plot at z=1 m **(i)** Absolute error plot at z=1 m

Figure 5.2: True (Fig a,d,g) and GSF-reconstructed (Fig b,e,h) normalized temperature XY planes with respective temperature error plots (Fig c,f,i) for Virtual Experiment 1.1. The optimal global field prediction requires only 8 sensor measurements.

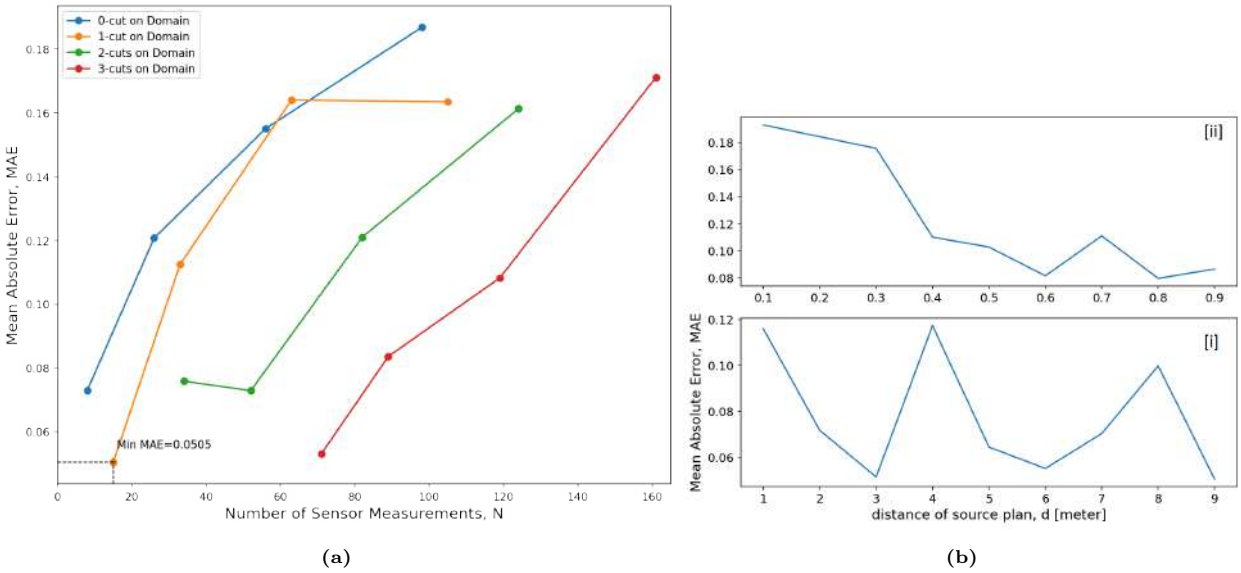
5.2. Results: Virtual Experiments 1.2 – Gaussian Heat Source

In Virtual Experiment 1.2, smooth field variations are attained with a Gaussian heat source of the form: $\dot{q} = 20 \cdot exp[-\frac{(x-0.5)^2+(y-0.5)^2+(z-0.5)^2}{2}]$ in the domain; the surfaces of the extended domain are again given constant temperature values in the range of 30 to 80°C, such that surfaces lying in the same plane have the same temperature. The strength and spread of the source are chosen such that there are smooth temperature variations spanning across the domain.

5.2.1. Field Predictions using Generic Source Framework

The heat source varies as a Gaussian function in the virtual experiment, increasing DoF associated with it. Such heat sources exhibit a characteristic where their strength gradually decreases with increasing

distance from the center. However, Generic Framework will continue to use the same functional form (Eq 3.1) and a 'generic' approach to address the *Issue 2* for the field reconstruction. Similar to the previous case, the training entails a systematic increase in characteristic values of the framework. First, a cut on domain surfaces is fixed and then cuts and distance of the source plane are varied. The required number of sensor measurements increases with increasing cuts for an exact solution. The changes in MAE are observed with varying cuts as shown in Figure 5.3a. While Figure 5.3b shows the incremental adjustment of the source plane distance 'd'.

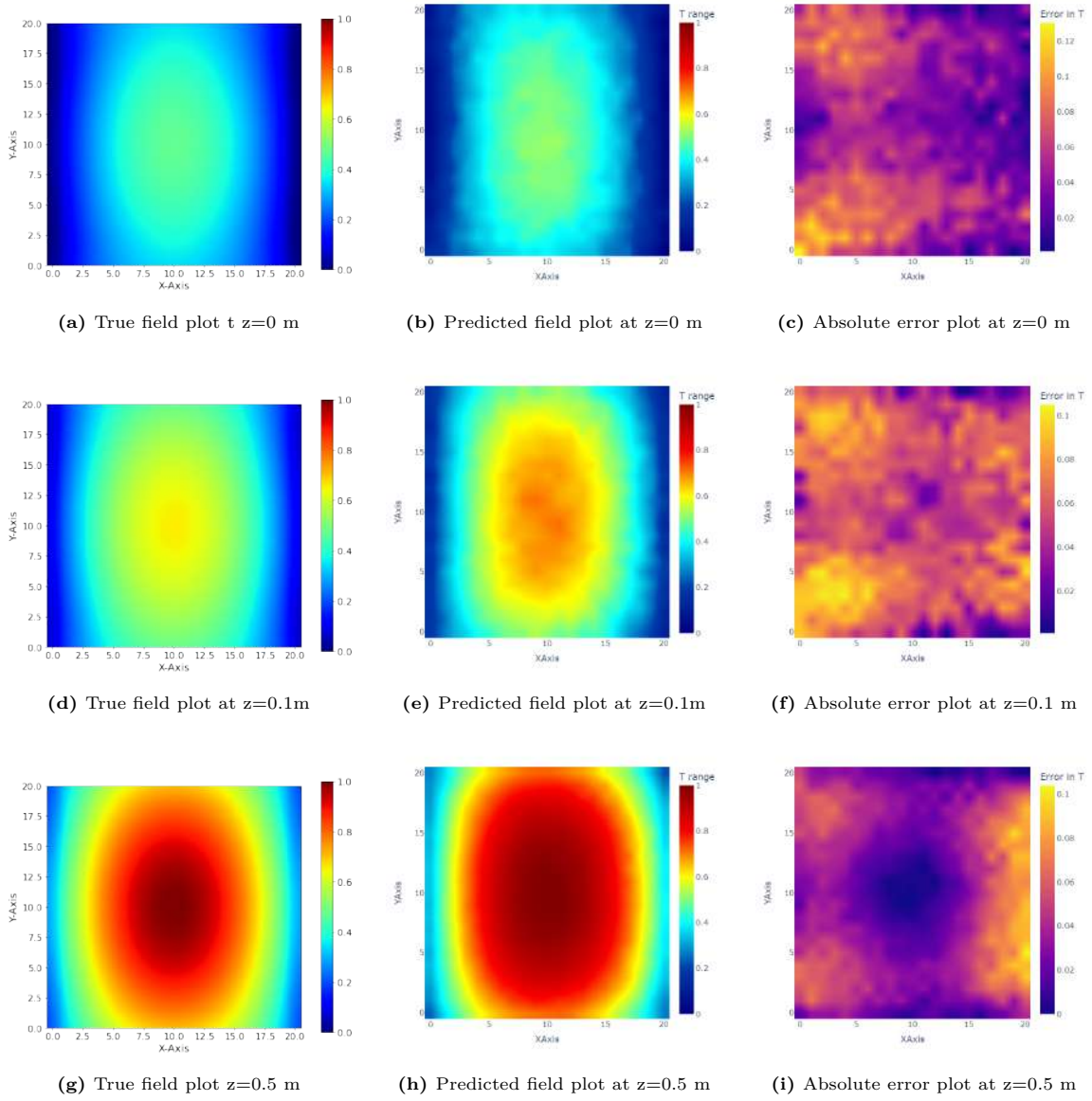


(a) **(b)**

Figure 5.3: Optimal GSF settings for Virtual Experiment 1.2 (a)A plot of MAE with the varying number of cuts on the domain and source planes (equivalently varying Sensor Measurements). (b)A plot of MAE with varying source-planes distances 'd' for 0-cut on the source-planes and domain; [i] between 1 to 10m [ii] zoomed in between 0.1 to 1m.

Analogous to the observations made for the previous experiment, for a given cut on the domain, increasing cuts on the external source planes generally increases the MAE. And, also increasing the number of cuts on the domain doesn't improve the error. For this case, with 15 sensor measurements (1-cut on the domain, 0-cut on source planes with d=9m), an optimal global reconstruction is achieved with a minimum MAE of 0.050. Figure 5.4 showcases the true and reconstructed normalized temperature field on XY plane. Alongside, respective absolute error plots $|T_{actual} - T_{predicted}|$ on an XY plane are depicted.

The predictions by Generic Framework are fair at the center XY plane (z=0.5m), where the heat source is located. However, towards the surface at z=0m, the noise built-up increases. This noise is not observed when the heat source is constant (VE 1.1). The error distribution in Fig 5.4c, 5.4f and 5.4i suggest that the error tends to be higher (0.10-0.12) towards the boundaries of the domain. While near the location of Gaussian source, the error is considerably low. These findings provide further support for the conclusions drawn in the previous case. The probable hypothesis described in the previous section regarding the infinite solution possibilities is reinforced. Many solution possibilities arising from outside and inside source potentials lead to conflicting boundary conditions which causes a higher error towards the boundaries.



(a) True field plot t z=0 m **(b)** Predicted field plot at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field plot at z=0.1m **(e)** Predicted field plot at z=0.1m **(f)** Absolute error plot at z=0.1 m

(g) True field plot z=0.5 m **(h)** Predicted field plot at z=0.5 m **(i)** Absolute error plot at z=0.5 m

Figure 5.4: True (Fig a,d,g) and GSF-reconstructed (Fig b,e,h) normalized temperature XY planes with respective temperature error plots (Fig c,f,i) for Virtual Experiment 1.2. The optimal global field prediction requires just 15 sensor measurements

Moreover, the error near the boundaries is greater compared to the case of a constant heat source (VE 1.1), despite both cases exhibiting similar temperature ranges. This remark, combined with the presence of noise built-up, can be attributed to the 'generic' approach employed to address *Issue 2* when the heat source varies as a Gaussian function.

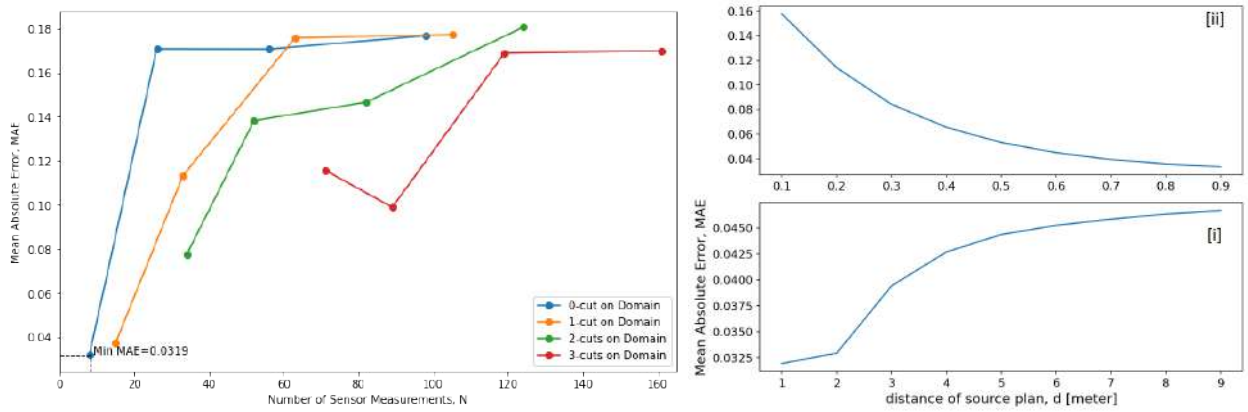
The Generic Framework works adequately for reconstructing a smooth field. This provokes an interest to unravel the framework's performance in reconstructing a locally uneven field. The upcoming section touches upon this discussion.

5.3. Results: Virtual Experiment 2 – Localized Heat Source

To recall, this experiment had a sudden spike in otherwise smooth field variations. This discontinuity was attained with a combination of Gaussian heat sources. The heat sources in the combination varied as $\dot{q}_1 = 125 \cdot exp[-\frac{(x-0.5)^2+(y-0.25)^2+(z-0.75)^2}{0.0034}]$ and $\dot{q}_2 = 25 \cdot exp[-\frac{(x-0)^2+(y-0)^2+(z-0)^2}{0.0208}]$ in the domain. q_1 has greater intensity and less spread compared to q_2. The surfaces of the extended domain are given constant temperature values in the range of 30 to 80°C, such that surfaces lying in the same plane have the same temperature. The resulting temperature variations are generally smooth everywhere, similar to VR-3 [11], except at the locations of heat sources. The presence of a dominant heat source manifests as a localized, abrupt increase in temperature within the field.

5.3.1. Field Predictions using Generic Source Framework

The heat sources ($\dot{q}_1$ and $\dot{q}_2$) in the experiment vary as a Gaussian function, where the strength decreases with increasing distance from the center. The decrease in strength is gradual or abrupt depending on the spread of the source. However, the Generic Framework adopts a generalized approach to tackle *Issue 2* and relies on Eq. 3.1 to capture the exponential attribute involved with these sources and continues to based on Eq 3.1.

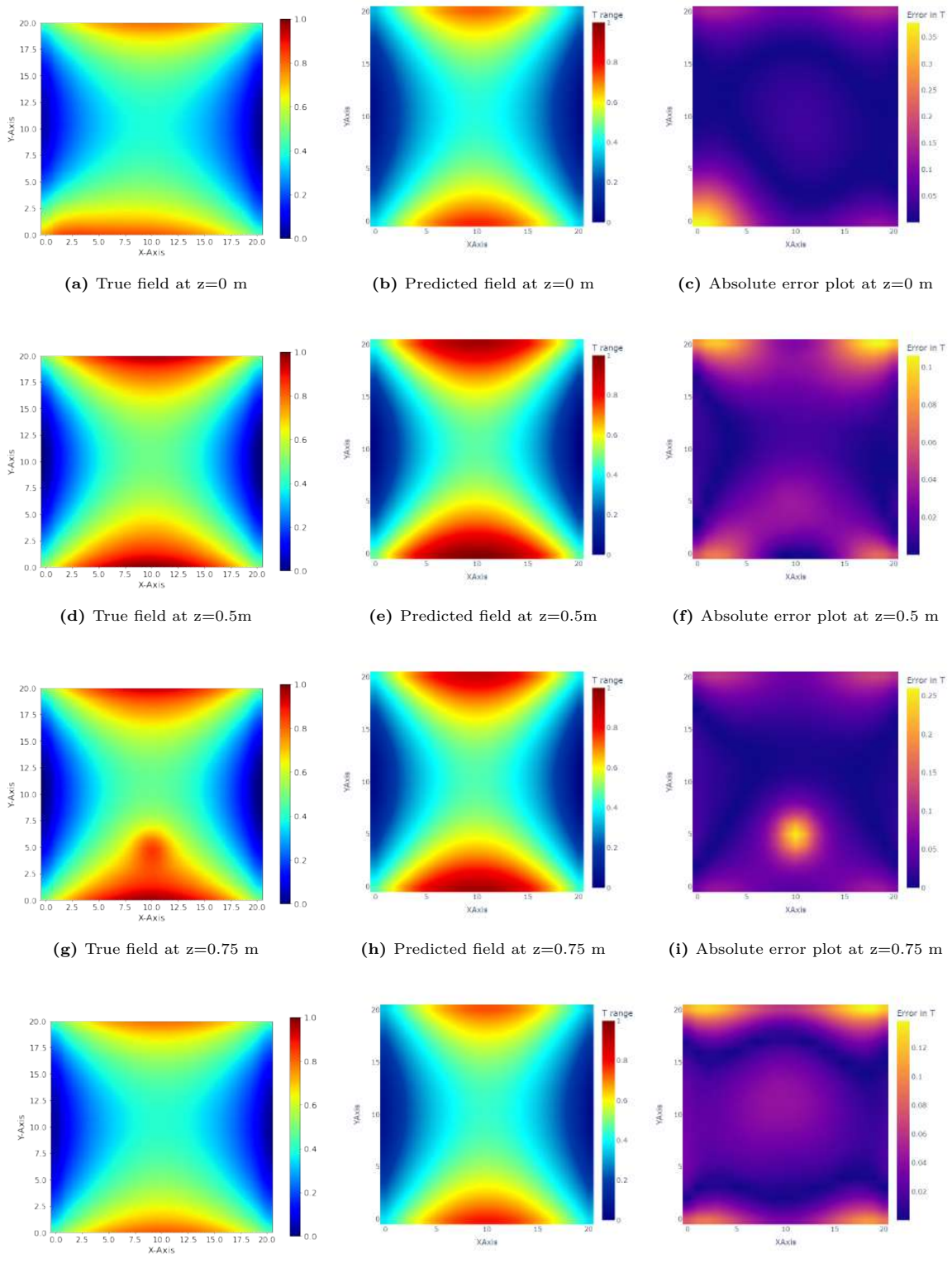


(a) **(b)**

Figure 5.5: Optimal GSF settings for Virtual Experiment 2 (a)A plot of MAE with the varying number of cuts on the domain and source planes (equivalently varying Sensor Measurements). (b)A plot of MAE with varying source-planes distances 'd' for 0-cut on the source-planes and domain; [i] between 1 to 10m [ii] zoomed in between 0.1 to 1m.

The framework is trained with a systematic variation in its characteristic values to find the best combination of the number and location of the hypothetical heat potentials. Figure 5.5 represents a plot of MAE versus the number of sensor measurements. Observations gained from Figure 5.5 are consistent with those discussed in the previous cases, in terms of increasing cuts and changes in MAE over the domain. In general, increasing the number of measurements does not improve the predictions and rather suffers an increase in MAE over the domain. Generic Framework-assisted reconstructed XY planes are depicted in Figure 5.6, with 0-cut on the domain and on external source planes (optimally located at d= 1m from the domain). With 8 sensor measurements, an MAE of 0.032 is achieved over the entire domain. Figure 5.6 showcases the true and reconstructed normalized temperature field on the XY plane. Alongside, respective absolute error plots $|T_{actual} - T_{predicted}|$ on an XY plane are depicted. On observing Fig 5.6d and 5.6j, the temperature profiles at z = 0.5 and 1m looks equivalent to that in VR-3 in [11]. However, the XY planes at z = 0 and 0.75m in Fig 5.6a and 5.6g reveal the local spikes

from the heat sources involved in this experiment.



(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field at z=0.5m **(e)** Predicted field at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field at z=0.75 m **(h)** Predicted field at z=0.75 m **(i)** Absolute error plot at z=0.75 m

(j) True field at z=1 m **(k)** Predicted field at z=1 m **(l)** Absolute error plot at z=1m

Figure 5.6: True (Fig a,d,g) and GSF-reconstructed (Fig b,e,h) normalized temperature XY planes with respective temperature error plots (Fig c,f,i) for Virtual Experiment 2. The optimal global field prediction requires 8 sensor measurements.

The reconstructed XY plane in Fig 5.6e suggests that GSF is able to retain smooth and symmetric variations. However, it fails to recognize the local temperature spike as observed in Fig 5.6h. Instead, the framework homogenizes the local pattern while making predictions. This detail again refers back to the 'generic' treatment and oversimplified approach to deal with *Issue 2* in the Generic Framework. An oversimplified model may underestimate the role of local factors. In addition, such a model can conceal important interactions or relationships in the data as noticed in this example. This can lead to inaccurate or incomplete conclusions.

Even though the heat sources vary as a Gaussian function similar to VE 1.2, the predictions do not exhibit noisy behaviour. This difference can be linked to the dissimilar characteristics of the Gaussian functions in both experiments. Although the strength of $\dot{q}_1$ and $\dot{q}_2$ is higher than the source in VE 1.2, their spread is much narrower in comparison. Hence, the field variations are primarily governed by heat conduction. For this reason, the Generic Framework gives predictions much like VR-3 in [11], without any noise. Furthermore, it is worth noting that the average error is comparatively lower compared to VE 1.2, attributed to the same underlying reason. Absolute error plots for the XY planes (Fig 5.6c, 5.6f, 5.6i, 5.6l) reveal that error is higher near the boundaries and near the local heat sources, in the range of [0.10-0.35]. It is noticed that even though the average absolute error across the entire domain was low (≈ 0.03), the local error came out significantly high (≈ 0.25 to 0.35). A local high error was expected as the framework did not identify the sudden field variations at all. YZ and XZ reconstructions are supplemented in C.2.1 for reference.

The limitations arising from the Generic Framework-assisted predictions can be pointed towards the non-specific treatment of *Issue 2*. A hypothetical source potential distributed in the Generic Framework is either infinitely large or small, depending on its position in space. This leads to the oversimplification of the model. Eventually, the local effects are not identified by the framework-assisted field predictions. This paved the way for testing the Partition Source Framework, where the *Issue 2* is addressed more specifically. The predictions using PSF for Virtual Experiment 2 are presented in the next subsection.

5.3.2. Field Predictions using Partition Source Framework

A PSF configuration set-up with just a 0-cut on the domain will be similar to a 0-cut Generic Framework since only one source would be present in the center of the domain. And so, the training does not include the 0-cut on the domain. Training with 3-cuts and higher was avoided since the computational complexities were increasing considerably. Moreover, increasing the number of cuts on external source planes may not be beneficial or effective based on the foregoing experience. Therefore, certain decisions are made regarding the training of the Partition Framework. The training process involves either implementing a 1-cut or 2-cut on the domain, accompanied by the variation of the source plane distance 'd,' while maintaining 0-cuts on the source planes.

Partition Source Framework (Exact Solution Approach)

Table 5.1: PSF Exact Solution for VE 2: MAE values and optimal distance of external source planes for 1-cut and 2-cuts on the domain.

Cuts on the Domain	Cuts on the Source Plane	Distance 'd' of the Source Plane from the domain	MAE
1-cut	0-cut	2 m	0.053
2-cut	0-cut	1 m	0.116

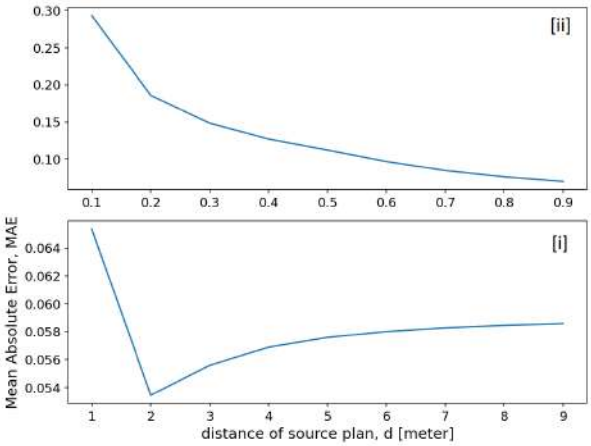
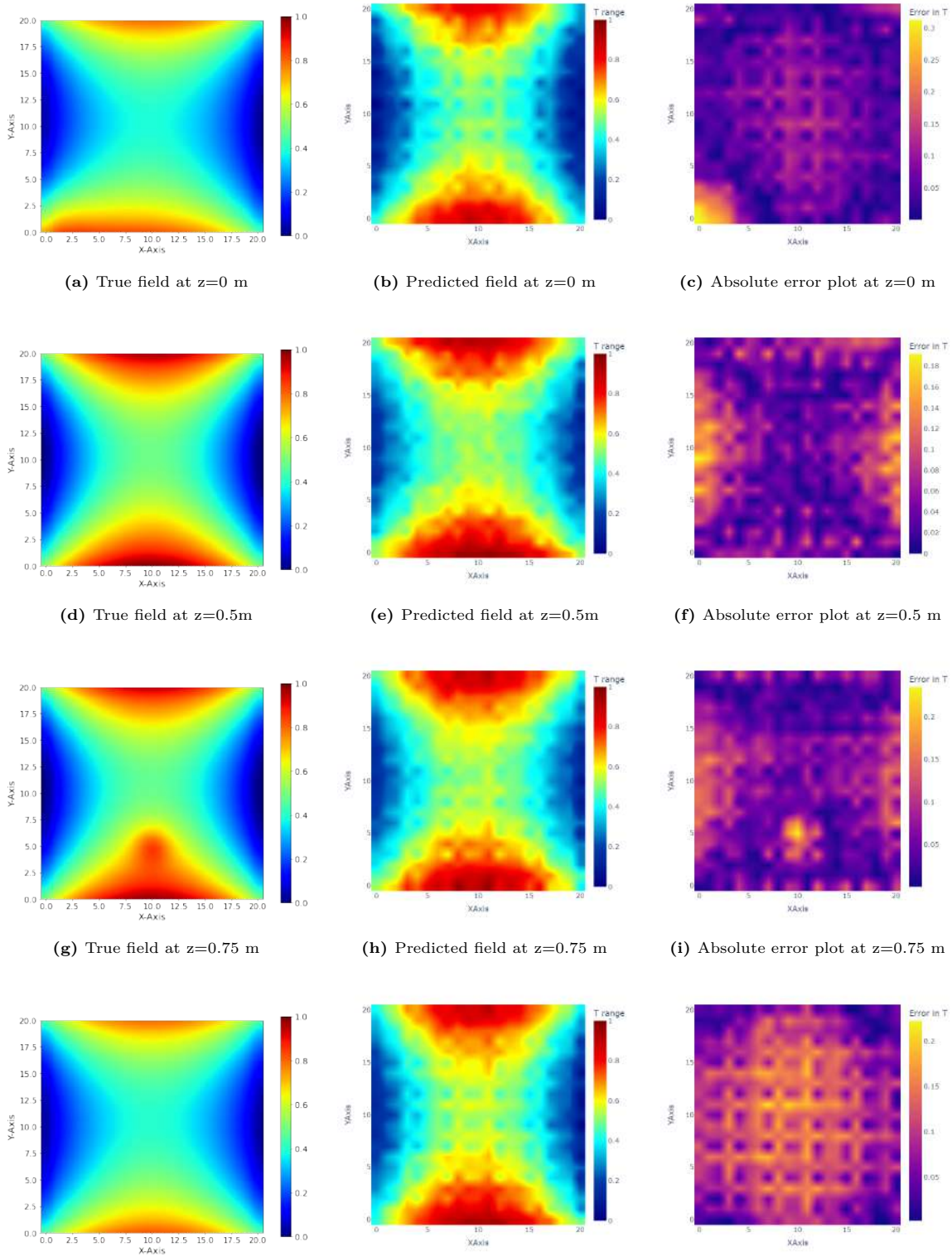


Figure 5.7: PSF-Exact Solution for VE 2: A plot of MAE with varying source-planes distances 'd' for 0-cut on the source-planes and 1-cut on the domain; [i] between 1 to 10m [ii] zoomed in between 0.1 to 1m.

The optimal framework configuration is selected as 1-cut on the domain and 0-cut on the external source planes (located at d= 2m) around the domain (Table 5.1). A mean absolute error (MAE) of 0.053 is achieved for the overall domain. Figure 5.8 presents the true and reconstructed normalized temperature field on the XY plane using the Partition Framework.

The reconstruction plots in Figure 5.8 reveal a conspicuous cross-like pattern, which is also observed in the error plots and could be interpreted as noise. However, the noise present in the GSF predictions did not exhibit similar cross patterns. Hence, this occurrence of the pattern in PSF results can be attributed to the rigid (and complex) formulation of *Issue 2* in link with *Issue 3* within the (Partition) Framework. Additionally, the utilization of an exact solution technique on a complex model can lead to overfitting. Overfitting occurs when a model memorizes the training data instead of learning the underlying patterns, it may produce noisy predictions when presented with new data.

The error plots at z=0m and 1m, in Fig 5.8b and 5.8l, suggest that the absolute error near the boundaries (domain surfaces) is higher; in the range of [0.18-0.30]. This observation supports the 'conflict of boundary conditions' hypothesis. The issue of local error continued to persist as depicted in Figure 5.8i. Although, the framework at least attempts to identify the local effect by bringing down the error to [0.20-0.30] (from [0.25-0.35] in GSF). In an attempt to enhance the predictions, a combination of different cuts on the domain and source planes (e.g., 0-cut + 2-cut) was implemented. However, this approach did not yield any improvement in the predictions. Supplementary reconstructed YZ planes can be referred to in Appendix C.2.2 to support all the observations made above.



(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field at z=0.5m **(e)** Predicted field at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field at z=0.75 m **(h)** Predicted field at z=0.75 m **(i)** Absolute error plot at z=0.75 m

(j) True field at z=1 m **(k)** Predicted field at z=1 m **(l)** Absolute error plot at z=1 m

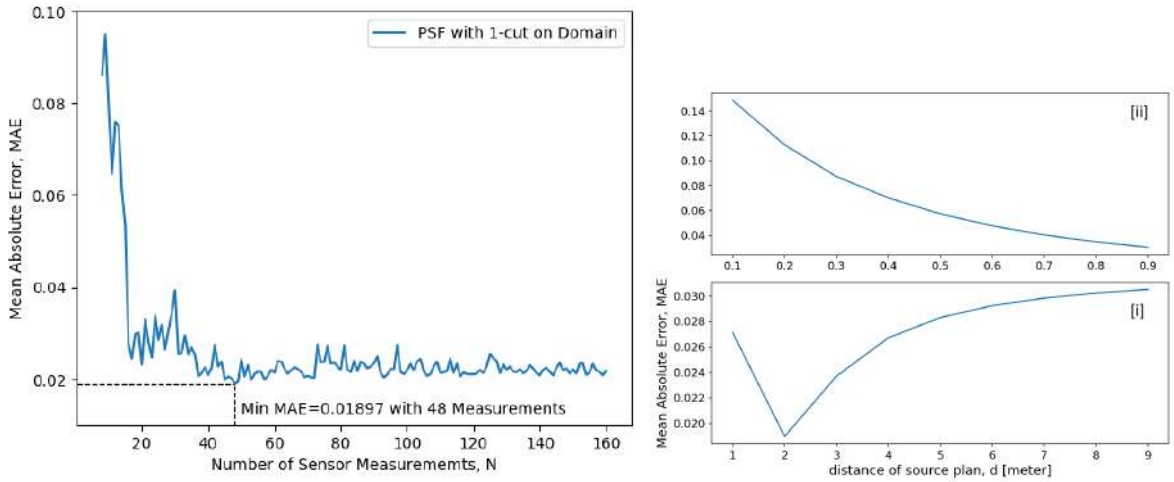
Figure 5.8: True and Partition Framework-Exact Solution assisted predictions: XY Temperature Plots of Virtual Experiment 2 and respective error plots.

The implementation of a 'rigid' approach to tackle *Issue 2* in Partition Framework introduces complexity to the model (Eq 3.3). To prevent overfitting, field reconstruction with a least-squares solution approach is attempted. The results from this approach are presented below.

Partition Source Framework (Least-Squares Solution Approach)

The domain configuration was fixed with 1-cut on the domain and 0-cut on external source planes. For obtaining optimal least-squares solution,

1. The number of sensor measurements varied from 8 to 160. This range was chosen to match the range of sensor measurements involved in the training for Generic Framework.
2. For each number of measurements used, the source-plane distance was varied to find the optimal location 'd' from the domain.



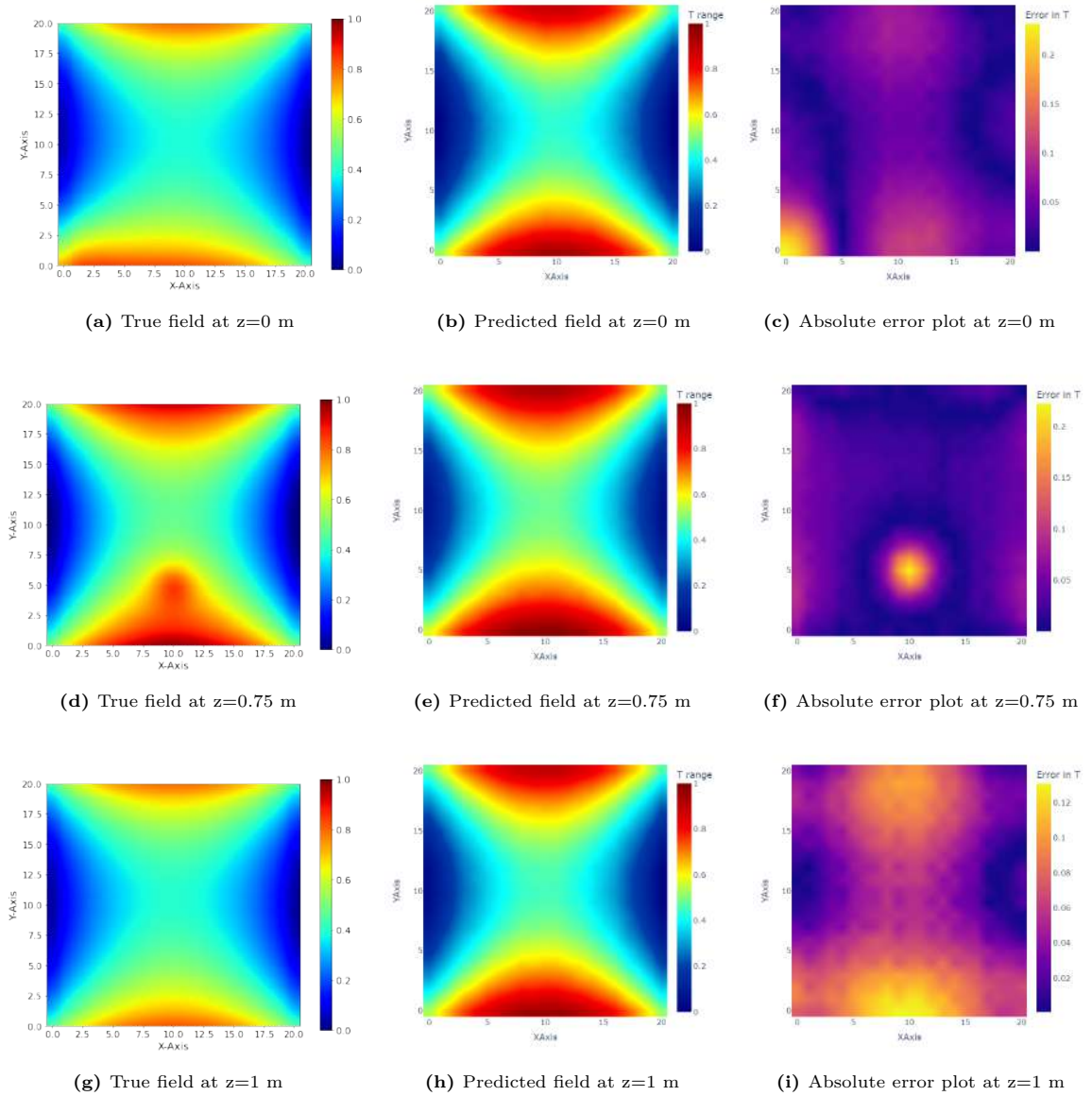
(a) (b)

Figure 5.9: Optimal framework settings for Virtual Experiment 2 using PSF: Least Squares Approach (a)A plot of MAE with the varying sensor measurements. (b)A plot of MAE with varying source-planes distances 'd' for 0-cut on the source-planes and 1-cut on the domain; [i] between 1 to 10m [ii] zoomed in between 0.1 to 1m.

Figure 5.9a shows a graph where for every number of sensor measurements, the corresponding optimal MAE is plotted. It can be concluded that after reaching a certain number of measurements, the optimal MAE remains almost the same. Figure 5.9b shows the incremental adjustment of the source plane distance 'd'. For the particular case of 1-cut on the domain and 0-cut on the source planes, the optimal MAE of 0.018 is achieved with 48 measurements, and the external planes are located at d = 2m. While transitioning from an exact solution to a least-squares solution, the minimum MAE is reduced from 0.053 to 0.018 by using additional 33 measurements. Figure 5.10 includes the reconstructed temperature field on XY planes for which the least-squares approach is implemented with 48 sensor measurements.

The conspicuous cross-like pattern noticed in Fig 5.8 is no longer observed in Fig 5.10. The least-squares technique, therefore, assists in the reduction of noise by providing an approximate solution and preventing overfitting. It gives flexibility to a complex model for not having an exact solution. In addition, it also assists in the error reduction to [0.06-0.20] near the boundaries, compared to [0.10-0.35] in GSF and PSF-Exact Solution. However, the local error does not diminish entirely, and yet the sudden

field spikes are not clearly detected. The technique was also applied to a configuration with 2-cuts on the domain, but it did not achieve any better results (0.020 with 89 measurements) than the 1-cut configuration. Consequently, the 'rigid' formulation of *Issue 2* does not effectively address the challenge of 'homogenized predictions' for local effects. This highlights the inherent limitation of linearizing a non-linear problem with confined choices (in this case, source frameworks). In the present context, it may be more appropriate to retain the field function (Eq 2.32) in its non-linear form, allowing *Issues* to be resolved by a non-linear solver. In future studies, incorporating *a priori* knowledge of heat sources or heat generation into the framework development could aid in enhancing the accuracy of predictions for local effects.



(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field at z=0.75 m **(e)** Predicted field at z=0.75 m **(f)** Absolute error plot at z=0.75 m

(g) True field at z=1 m **(h)** Predicted field at z=1 m **(i)** Absolute error plot at z=1 m

Figure 5.10: True and Partition Framework-Least Squares Solution assisted predictions: XY Temperature Plots of Virtual Experiment 2 and respective temperature error plots.

5.4. Key Conclusions

Overall, the Generic Source Framework appears to be a promising approach for reconstructing smooth temperature fields with a relatively small number of sensor measurements, while still maintaining a reasonable level of accuracy. The average error is about 0.013-0.050 on a scale of 0-1. For all the cases considered, the error near the boundaries is higher than in the remaining regions. This can be attributed to the existence of multiple (uncoupled) solutions from different hypothetical source potentials in the framework and their internal conflict while delivering boundary predictions.

In the Generic Framework, the radius of heat sources (*Issue 2*) is chosen generally and can have only two extremes. It can be either infinitely large or small (with respect to their position). This general treatment of *Issue 2* is incapable to identify the sudden spikes present in reality (Virtual Experiment 2). This feature instead smoothens out the local variation resulting in a high local error. Therefore, the Generic framework selectively captures and retains information solely from the dominant physical mechanism that occurs within the domain.

On the other hand, Partition Source Framework has a specific treatment of *Issue 2* with respect to the number of cuts on the domain. It adds stiffness to the framework making it difficult to fit the data with an exact solution. Therefore, the framework delivers noisy predictions with cross-like patterns while using an exact solution approach. The difficulty is solved by finding an approximate solution using a least-squares solution. This delivers satisfactory results. The error values in the critical regions reduce relatively. However, similar to Generic Source Framework, the Partition Source Framework captures the dominant phenomena in its predictions, but it is insufficient for accurately identifying localized effects. Primarily, these frameworks only serve as one of many tools for linearizing the non-linear field function (Eq 2.32), allowing temperature predictions to be made with ease throughout the domain. However, it is important to recognize that this elementary approach has its limitations in accurately predicting local effects. Therefore, in order to improve the prediction of such localized effects, it may be necessary for the current frameworks to withhold from fully linearizing the field function and instead leave certain *Issues* to be evaluated by a non-linear solver. For instance, instead of imposing pre-determined *Issue 2* or *Issue 3*, allowing them to remain 'unknown' in the framework.

The following research objective is to investigate the framework performance specifically in the case of a combined forced convection and heat conduction mechanism. This evaluation is crucial as it assesses how well the frameworks can handle situations where the convective term behaves like a heat source ($\dot{q}_{conv}$). The intensity of the convective term can lead to temperature variations that can be either gradual or sharp. Therefore, in the upcoming chapter, the performance of these frameworks will be further tested on the forced convection setups.

6

Results: Virtual Experiment 3

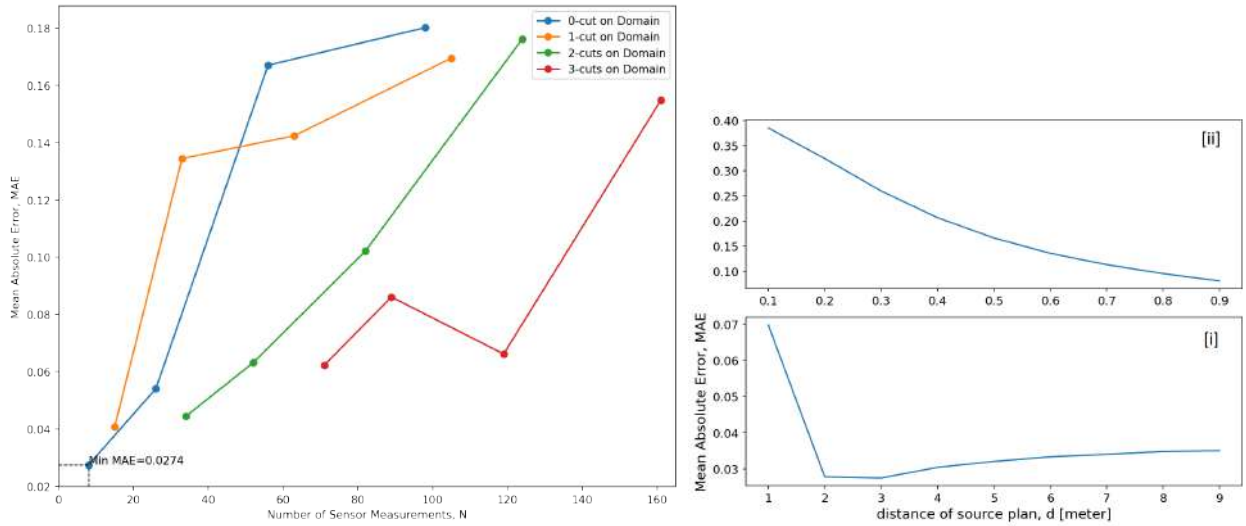
The purpose of this chapter is to assess the effectiveness of elementary source frameworks in predicting temperature field in a forced convection setup, as given in Virtual Experiment 3, and ascertain the concept of "$\rho C_p(\vec{v} \cdot \nabla T)$ acting equivalent to heat source $\dot{q}_{conv}$". To recollect, Virtual Experiment 3 involved a forced convection setup with two sub-cases, air (VE 3.1) and water (VE 3.2). The geometry of the domain is modified to include fluid flow in the experiments as shown in Fig B.13. The sub-case with air was smooth in terms of temperature field variations and was comparable to the case of VR-3 from [11] where only heat conduction was involved. While in the sub-case with water, due to dominating convective phenomena, sharp variations were observed in the field. Again, the true and predicted temperature values presented in this chapter have been normalized to ensure a fair comparison of performance among all cases.

6.1. Virtual Experiment 3.1: Forced Convection Setup with Air

In Virtual Experiment 3.1, the fluid medium is air with a low Péclet number, indicating that the convective effects were not significant. As a result, temperature variations in the field were primarily governed by conductive phenomena and the trends were similar to those observed in VR-3 from [11]. Minute field differences observed were near the inlet of the domain.

6.1.1. Field Predictions with Generic Source Framework

A similar approach to the previous cases considered in Chapter 5 is followed for the training of the Generic Framework. First, a cut on domain surfaces is fixed, and then cuts on the source planes and their distance from the domain are varied. Changes in the mean absolute error (MAE) can be observed for the field predictions over the entire domain as shown in Figure 6.1. Increasing cuts on the source planes does not relieve the average error across the domain. This outcome is consistent with the earlier observations.



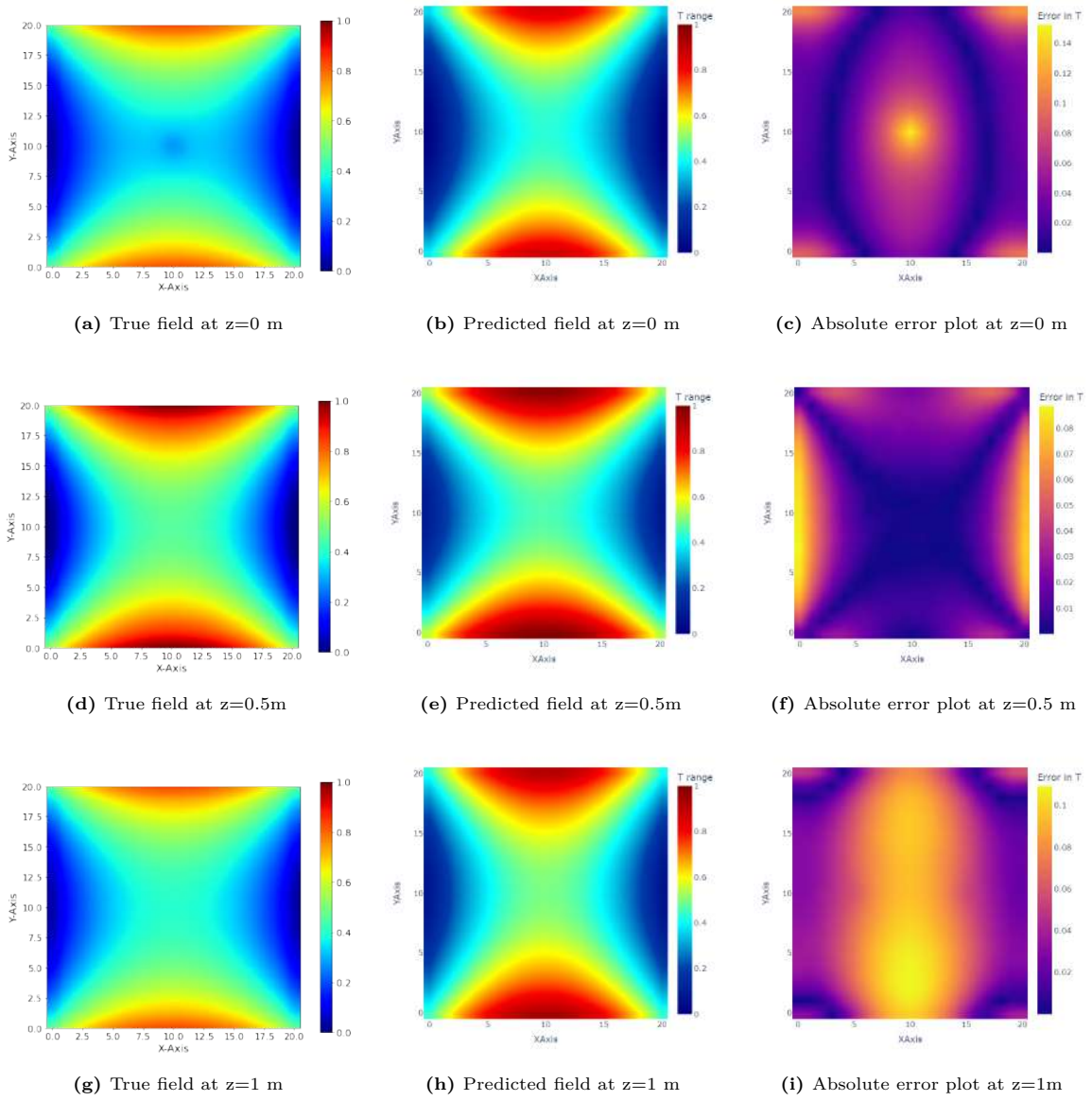
(a) (b)

Figure 6.1: Optimal GSF settings for Virtual Experiment 3.1 (a)A plot of MAE with the varying number of cuts on the domain and source planes (equivalently varying Sensor Measurements). (b)A plot of MAE with varying source-planes distances 'd' for 0-cut on the source-planes and domain; [i] between 1 to 10m [ii] zoomed in between 0.1 to 1m.

With 8 sensor measurements, a minimum MAE of 0.027 is obtained. Hence, for reconstruction, this corresponds to a 0-cut on the domain and 0-cut on the external source planes (located at d=3 m). The true and reconstructed normalized temperature field on XY planes are plotted in Figure 6.2, along with respective Temperature Error Plots. Additionally, true and reconstructed YZ and XZ planes can be referred to in Appendix C.3 for comparison. On close observation of the XY plane at z=0m (Fig 6.2a), we can notice the inlet is present in the center.

The predictions exhibit a reasonable level of accuracy; however, there are some notable observations. At the inlet (z=0m), the reconstructed plane does not clearly distinguish the inlet region, which is supported by the error plot (in Fig 6.2c) where the local error shoots up to 0.14. Additionally, higher errors are observed near the outlet (z=1m), approximately reaching up to 0.10 (Fig 6.2i). These locations correspond to the areas where the local heat effects ($\dot{q}_{conv}$) are present, as depicted in Fig B.15. Yet again, GSF tends to homogenize the local temperature differences, recognizing only the dominant phenomena in the domain.

The error plots also indicate higher errors towards the domain's edges and surfaces, ranging in [0.08-0.11]. These findings support the hypothesis of a 'solution conflict' at the boundaries from the hypothetical source potentials. The temperature variations in the current predictions can be compared with those in VE 2, as both cases are primarily governed by the heat conduction mechanism. The average error values in both cases (with GSF) are comparable. However, the local error values in the current case are lower than the GSF-predictions for VE 2. This difference can be attributed to the lower intensity of the heat source in the present experiment. All the results align with the observations made in the previous chapter. Consequently, in the next step, the study proceeds to test the predictions of the Partition Source Framework for this virtual experiment.



(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field at z=0.5m **(e)** Predicted field at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field at z=1 m **(h)** Predicted field at z=1 m **(i)** Absolute error plot at z=1m

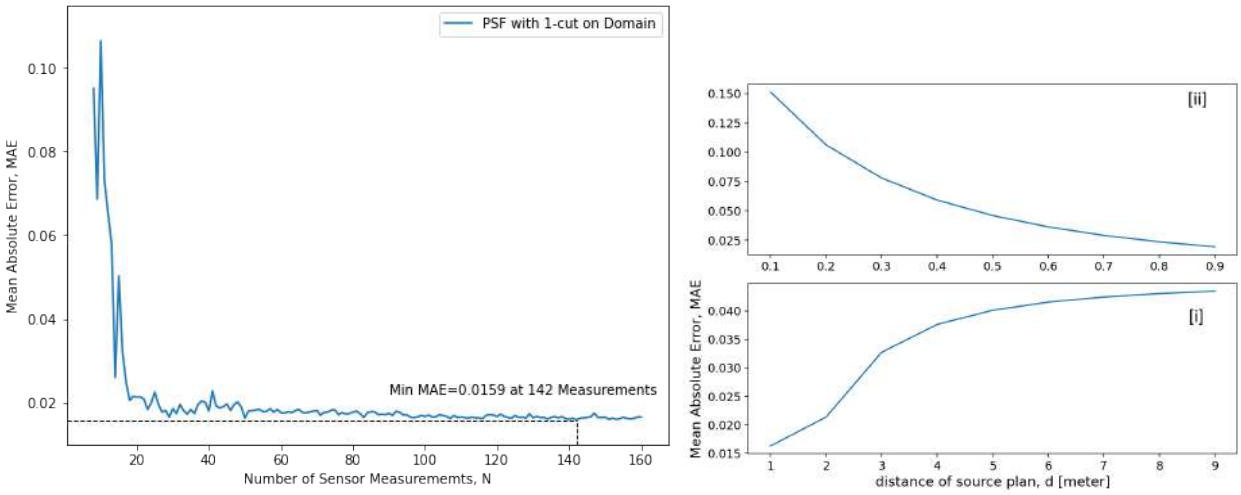
Figure 6.2: True (Fig a,d,g) and GSF-reconstructed (Fig b,e,h) normalized temperature XY planes with respective temperature error plots (Fig c,f,i) for Virtual Experiment 3.1. The optimal global field prediction requires 8 sensor measurements.

6.1.2. Field Predictions with Partition Source Framework

We concluded in Chapter 5 that Partition Source Framework is too rigid to apply the exact solution approach. It delivers the output with some noise build-up. In order to avoid noise build-up, Partition Source Framework will only utilize the least-squares approach to solve for 'unknown intensities'. To obtain an optimal least-squares solution for VE 3.1, the adjustments are kept similar to VE 2. The domain configuration is kept consistent with 1-cut on the domain and 0-cuts on external source planes, and

- The number of sensor measurements was adjusted between 8 to 160 to match the range of sensor measurements used in the training process for the Generic Framework.
- For each number of measurements used, the source-plane distance was varied to identify the

optimal location 'd' from the domain.



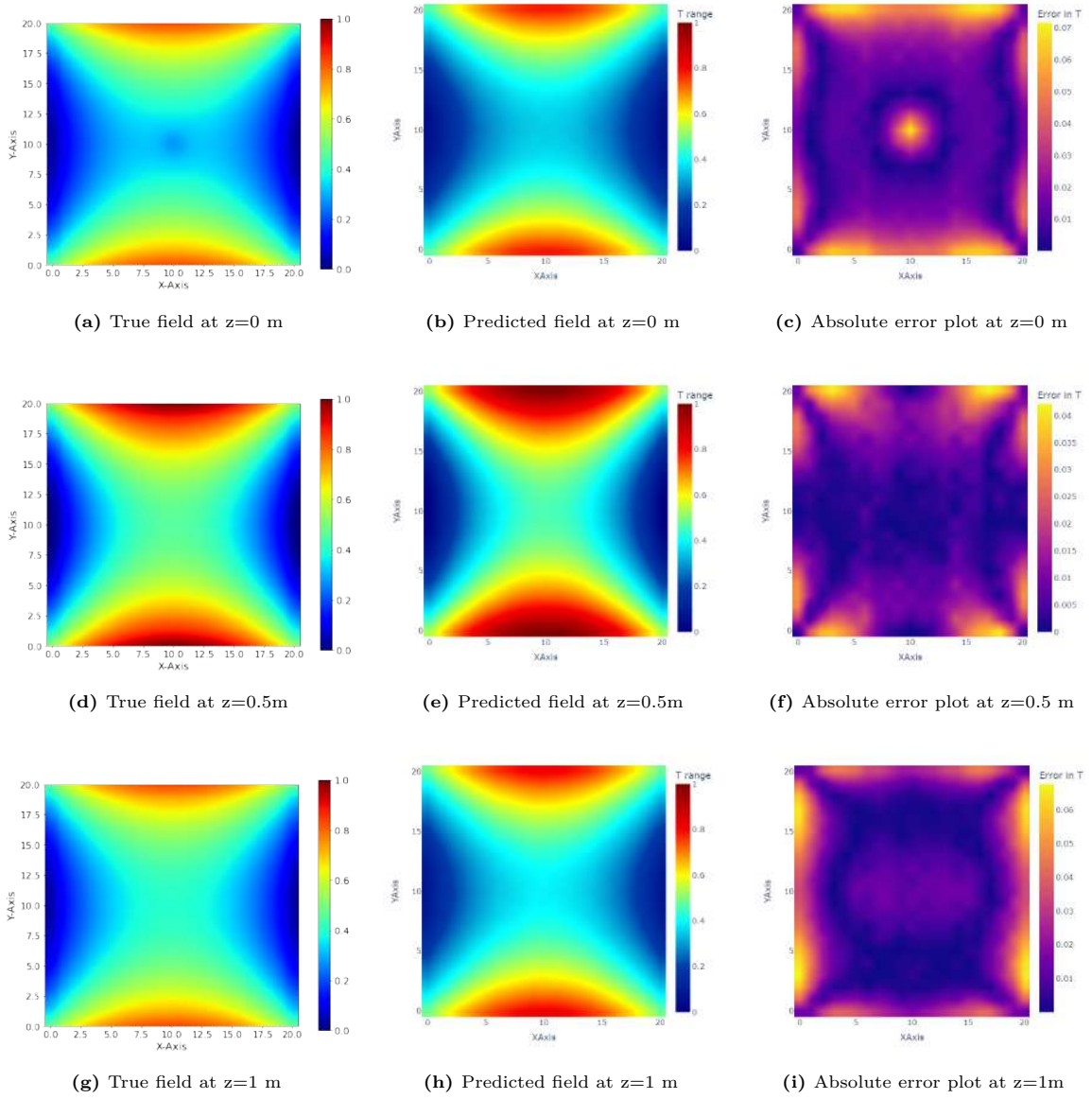
(a) **(b)**

Figure 6.3: Optimal PSF settings for Virtual Experiment 3.1 (a)A plot of MAE with the varying Sensor Measurements. (b)A plot of MAE with varying source-planes distances 'd' for 0-cut on the source-planes and 1-cut on the domain; [i] between 1 to 10m [ii] zoomed in between 0.1 to 1m.

Figure 6.3a depicts a graph displaying the optimal mean absolute error (MAE) for each number of sensor measurements. It can be inferred that after reaching a certain number of measurements, the optimal MAE remains nearly constant without significant changes. This observation is in agreement with the result observed in Figure 5.9a for Virtual Experiment 2. For this particular case, a minimum MAE of 0.015 is obtained using 142 sensor measurements, and source planes are located at d= 1 m. Alternatively, only 50 sensor measurements could also be utilized to achieve an MAE of 0.016 with source planes located at the same distance from the domain. The average error comes down from 0.027 in GSF-assisted predictions to 0.016 using the PSF-Least Squares technique.

Figure 6.4 represents the true and reconstructed normalized temperature field on XY planes for which the PSF-Least Squares approach is implemented with 50 sensor measurements. Appendix C.3.2 shows the reconstructed YZ planes. The application of the least-squares technique proves effective in mitigating noise accumulation and leads to a reduction in error values near the boundaries. In the GSF results, the error range near the boundaries was observed to be [0.08-0.14]. With the utilization of the PSF-LS approach, it reduces to half, i.e. [0.04-0.07].

The error comes down to 0.07 near the inlet and only about 0.01 near the outlets. Although the local error does not completely disappear, the framework approximately identifies the majority of the physical effects in the domain. These characteristics of the GSF and PSF predictions are consistent with the observations made in the predictions for VE 2, where the local heat source effects are present.



(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field at z=0.5m **(e)** Predicted field at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field at z=1 m **(h)** Predicted field at z=1 m **(i)** Absolute error plot at z=1m

Figure 6.4: True (Fig a,d,g) and PSF-reconstructed (Fig b,e,h) normalized temperature XY planes with respective temperature error plots (Fig c,f,i) for Virtual Experiment 3.1. The optimal global field is reconstructed with 50 sensor measurements.

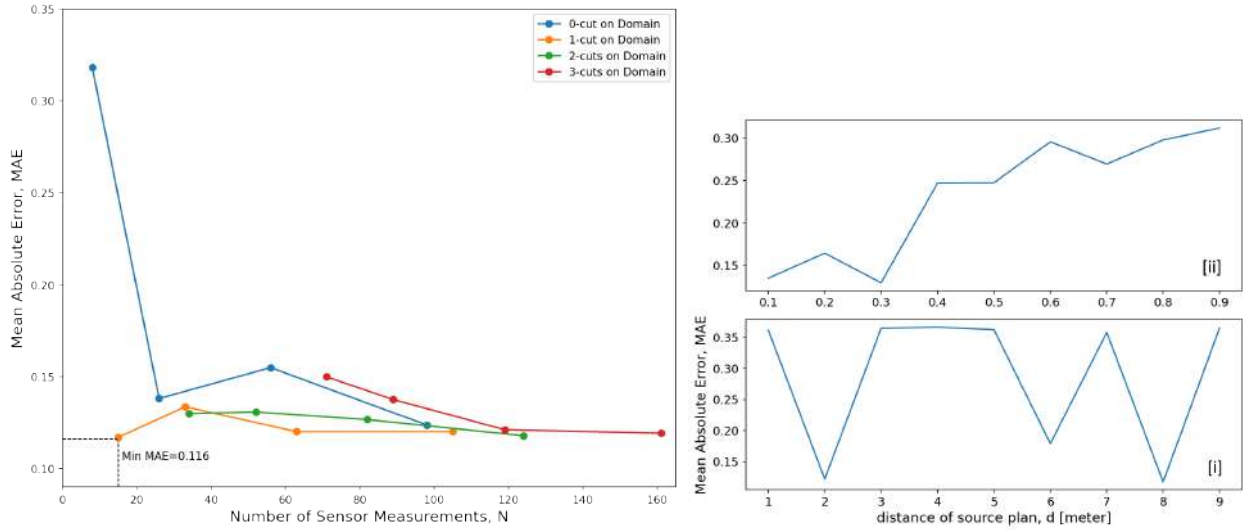
6.2. Virtual Experiment 3.2: Forced Convection Setup with Water

In Virtual Experiment 3.2, the fluid medium used was water and the remaining parameters were kept the same as in VE 3.1. This resulted in a high Péclet number indicating that the convective effects were now significant. As a result, temperature variations in the field were primarily governed by strong convective currents.

6.2.1. Field Predictions with Generic Source Framework

In GSF training, for every cut on the domain, the cut on the source planes is varied to train the framework algorithm with the corresponding number of sensor measurements. The resulting MAE measured over the entire domain is then plotted against the number of sensor measurements as represented in

Figure 6.5a.

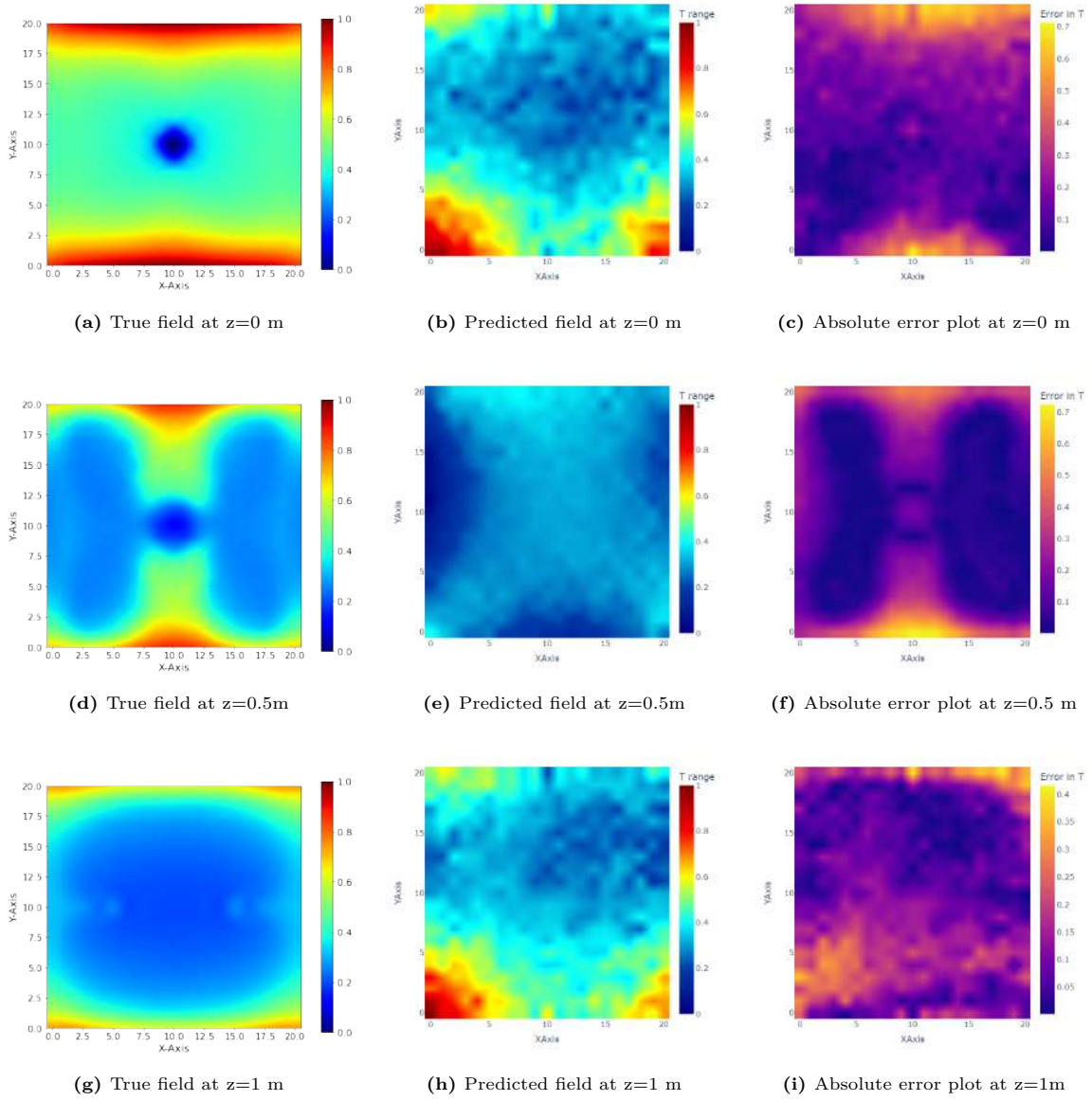


(a) **(b)**

Figure 6.5: Optimal GSF settings for Virtual Experiment 3.2 (a)A plot of MAE with the varying number of cuts on the domain and source planes (equivalently varying Sensor Measurements). (b)A plot of MAE with varying source-planes distances 'd' for 0-cut on the source-planes and 1-cut on the domain; [i] between 1 to 10m [ii] zoomed in between 0.1 to 1m.

As we can observe from this plot in Fig 6.5a, the mean error consistently varies in the range of [0.11, 0.15] for almost all the cut combinations, except for 0-cut on both domain and source planes. The minimum mean absolute error (MAE) of 0.116 is obtained with 15 sensor measurements (1-cut domain and 0-cut on source planes, located at d=8 m). Moreover, Fig 6.5b has oscillatory behaviour which suggests overfitting. Such a high average error reveals that the source framework is insufficient to capture substantial convective effects and that the reconstruction would not be satisfactory. The error near critical regions is expected to be worse than this average value. To validate our prognosis, reconstructed temperature XY planes and corresponding error plots are depicted in Figure 6.6. This figure also includes the true temperature XY planes to give an expectation for the temperature field which reflects the existing sharp variations in the temperature profile. Supplementary true and reconstructed YZ and XZ planes can be referred to in Appendix C.4.

Generic Framework does not attempt to identify the temperature trends in a forced convection setup. The error plots in Figure 6.6 support this observation. The error rises as high as 0.7 near the domain surfaces (boundaries). Additionally, predictions suffer from noisy response. This noise could be attributed to the convective currents present in the domain, causing the input data to be too complex for fitting with our elementary source frameworks. Appendix C.5 showcases a table with MAE of Generic Framework results for hypothetical fluids varying in Péclet range between extremes of air and water. The velocity is maintained constant in all cases. These values indicate that the resulting acute temperature variations, with increasing Péclet number, are not predicted well using Generic Framework. In the upcoming subsection, we test the Partition Source Framework-Least Squares technique's performance for a convection-dominated domain.



(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field at z=0.5m **(e)** Predicted field at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field at z=1 m **(h)** Predicted field at z=1 m **(i)** Absolute error plot at z=1m

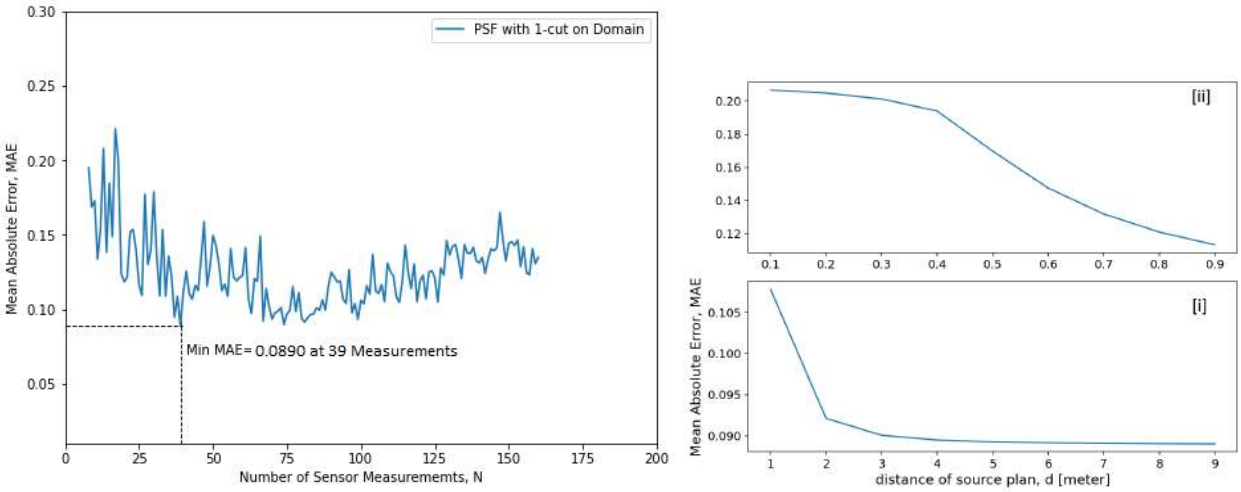
Figure 6.6: True and GSF-reconstructed normalized temperature XY planes with respective temperature error plots for Virtual Experiment 3.2.

6.2.2. Field Predictions with Partition Source Framework

The configuration for testing this framework with the least-squares approach remains similar to the prior setup. The domain configuration is kept consistent with 1-cut on the domain and 0-cuts on external source planes. The number of sensor measurements varied between 8 to 160. For each number of measurements used, the source-plane distance was altered to identify the optimal location 'd' from the domain.

Figure 6.7a shows the mean absolute error (MAE) varying with each number of sensor measurements. Even though the optimum is achieved with 39 sensor measurements (source plane distance d = 9m), more oscillatory behaviour is observed. In the earlier cases, the error became nearly constant but it is not the same in this case. In essence, this oscillatory behaviour hints that the implementation of this

framework is not suited to achieve a satisfactory reconstruction of a strongly convective field.

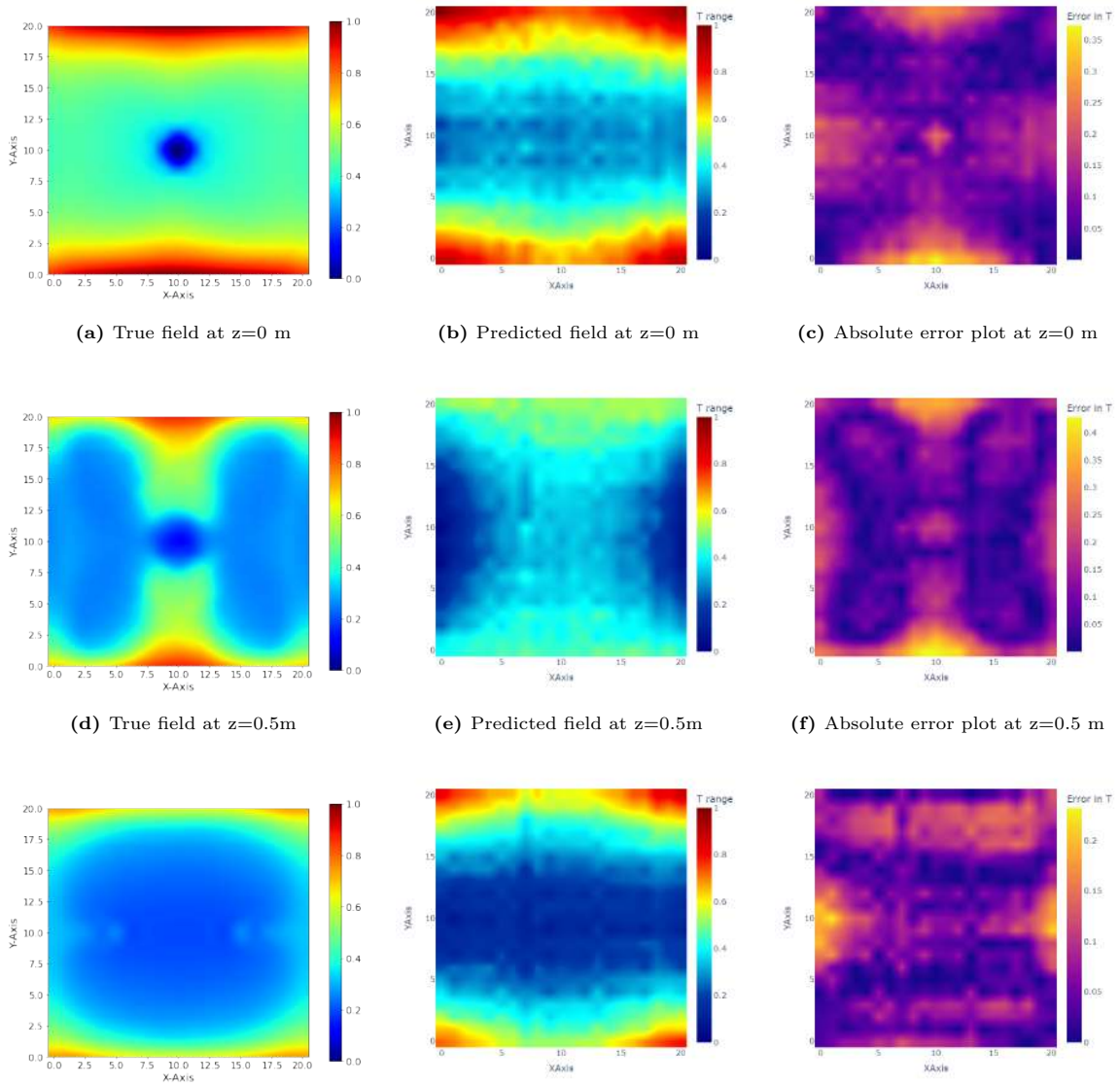


(a) **(b)**

Figure 6.7: Optimal PSF settings for Virtual Experiment 3.2 (a)A plot of MAE with the varying Sensor Measurements. (b)A plot of MAE with varying source-planes distances 'd' for 0-cut on the source-planes and 1-cut on the domain; [i] between 1 to 10m [ii] zoomed in between 0.1 to 1m.

Figure 6.7b displays the variations in MAE as the source plane distance from the domain changes. Contrary to previous observations, the MAE continues to decrease until a distance of 9m from the domain. It is possible that the MAE could potentially decrease further beyond 9m, but this range was not tested in order to maintain uniformity in the configuration of the source plane distance across different examples. True and reconstructed temperature fields on XY planes with their corresponding temperature error plots are presented in Figure 6.8.

While PSF shows some improvement compared to GSF, its predictions are still inadequate in capturing the present temperature trends accurately. The error near the domain surfaces continues to remain high, in the range of [0.35-0.4]. Despite applying the least-squares technique, the output showcases significant noise build-up. Least-squares solution with 2-cuts on the domain did not improve the results (MAE=0.1020 with 123 measurements), rather more oscillatory behaviour was observed and substantially high computational effort was required. Additional true and reconstructed YZ planes may be referred to in Appendix C.4.



(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field at z=0.5m **(e)** Predicted field at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field at z=1 m **(h)** Predicted field at z=1 m **(i)** Absolute error plot at z=1m

Figure 6.8: True and PSF-reconstructed normalized temperature XY planes with respective temperature error plots for Virtual Experiment 3.2. 39 sensor measurements are used for reconstructing the field.

6.3. Key Conclusions

The Generic and Partition Source Frameworks exhibit a reasonable level of efficiency in predicting smooth temperature fields in the low Péclet range. Framework-assisted predictions capture the dominant variations within the domain. Among the two frameworks, the Partition Source Framework, particularly when complemented with the least-squares solution technique, surpasses the Generic Source Framework in reducing noise, achieving lower average error values, and improving accuracy near the boundaries. However, both frameworks face limitations when it comes to accurately reproducing local effects or sharp temperature variations, especially in regions dominated by convective effects.

It is important to acknowledge that each framework-assisted reconstruction is still a solution satisfying a random sensor dataset. These frameworks primarily serve as a way to linearize the non-linear field

function (Eq 2.32). Frameworks address certain *Issues* involved in this equation, enabling convenient temperature predictions throughout the domain. Nevertheless, it is essential to recognize the inherent limitations of this simplistic approach in accurately predicting localized or acute effects. Future advancements in these frameworks could involve leaving the hypothetical source potential positions and/or sizes open to a non-linear solver. Established optimization techniques can be implemented to enhance prediction accuracy. Supplementing with *a priori* information on the nature of the heat source (for example, heat generation in an exothermic reaction) or on the velocity field in a forced convection case can provide further sophistication.

7

Comparison with Hidden Fluid Mechanics Algorithm

This chapter covers an investigation conducted into the effectiveness of Physics-Informed Neural Networks, and their performance was compared to the results obtained from our source frameworks. In recent years, there has been growing interest in using physics-based knowledge to enhance machine learning algorithms. This led to the emergence of the field of physics-informed machine learning. One of the key approaches in this field is the use of physics-informed neural networks (PINNs). PINNs integrate physical laws into the neural network architecture to improve their predictive capabilities. Such physics-informed neural networks are developed into an algorithm known as "Hidden Fluid Mechanics (HFM)" [9] and can be utilized to extract information from different fields from a point cloud of passive scalar data.

7.1. Introduction to the HFM Algorithm

While experimental fluid mechanics has made significant progress, accurately determining fluid velocity and pressure or stress fields through measurements remains a complex and challenging task. Applications of HFM have been demonstrated by Raissi *et al.* [9] for the extraction of quantitative information where direct measurement techniques are not feasible. The HFM algorithm consists of a physics-*uninformed* deep neural network to approximate $(t, x, y, z) \rightarrow (c, u, v, w, p)$, which is followed by a physics-*informed* neural network (t,x,y,z) $\rightarrow (e_1, e_2, e_3, e_4, e_5)$, where the Navier-Stokes equations are incorporated into outputs as residuals e_i. e_1 represents the residual of the transport equation for the passive scalar c, while e_2, e_3, and e_4 pertain to the momentum equations in the x, y, and z directions. Lastly, e_5 corresponds to the residual of the continuity equation. u, v, w represent velocity components in the x, y, and z directions, and p stands for pressure. Figure 7.1 illustrates the concept behind this neural network architecture.

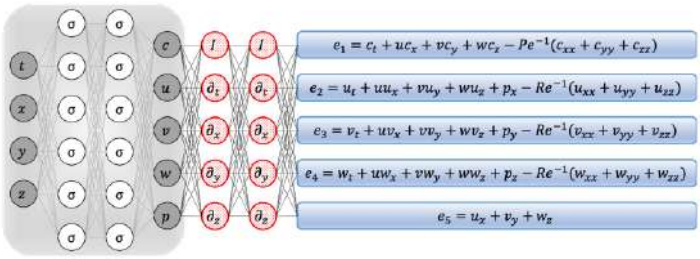


Figure 7.1: Schematic of Physics-Informed Neural Network retrieved from [9]

To minimize the prediction errors, the residuals (e_i) are also included into the loss function while learning. The loss function is given as Mean Squared Error (MSE). This technique is also agnostic of geometry, providing freedom to apply this concept to any domain of interest. Figure 7.2 shows an example of an intracranial aneurysm sac from [9], where reference and regressed fields for the concentration variable c are presented as contours on XY and YZ planes. The HFM algorithm is also said to have robustness to low-resolution input data. The authors examined the spatial-temporal resolution of the training data in detail for which spatial resolutions ranging from 250 to 15,000 data points, and time resolutions ranging from 3 to 201 time frames were explored. The results indicated that the proposed algorithm is highly resilient to variations in the spatial-temporal resolution of the data point cloud.

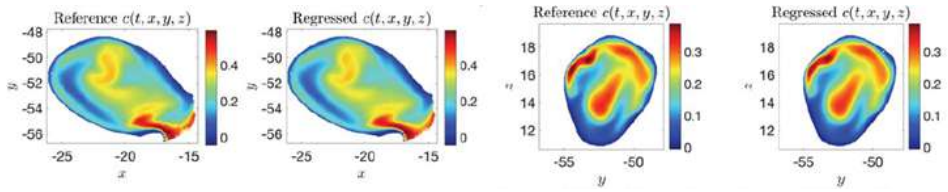


Figure 7.2: An example of aneurysm sac where reference and regressed concentration fields are shown as contours on XY and YZ planes (Retrieved from [9]).

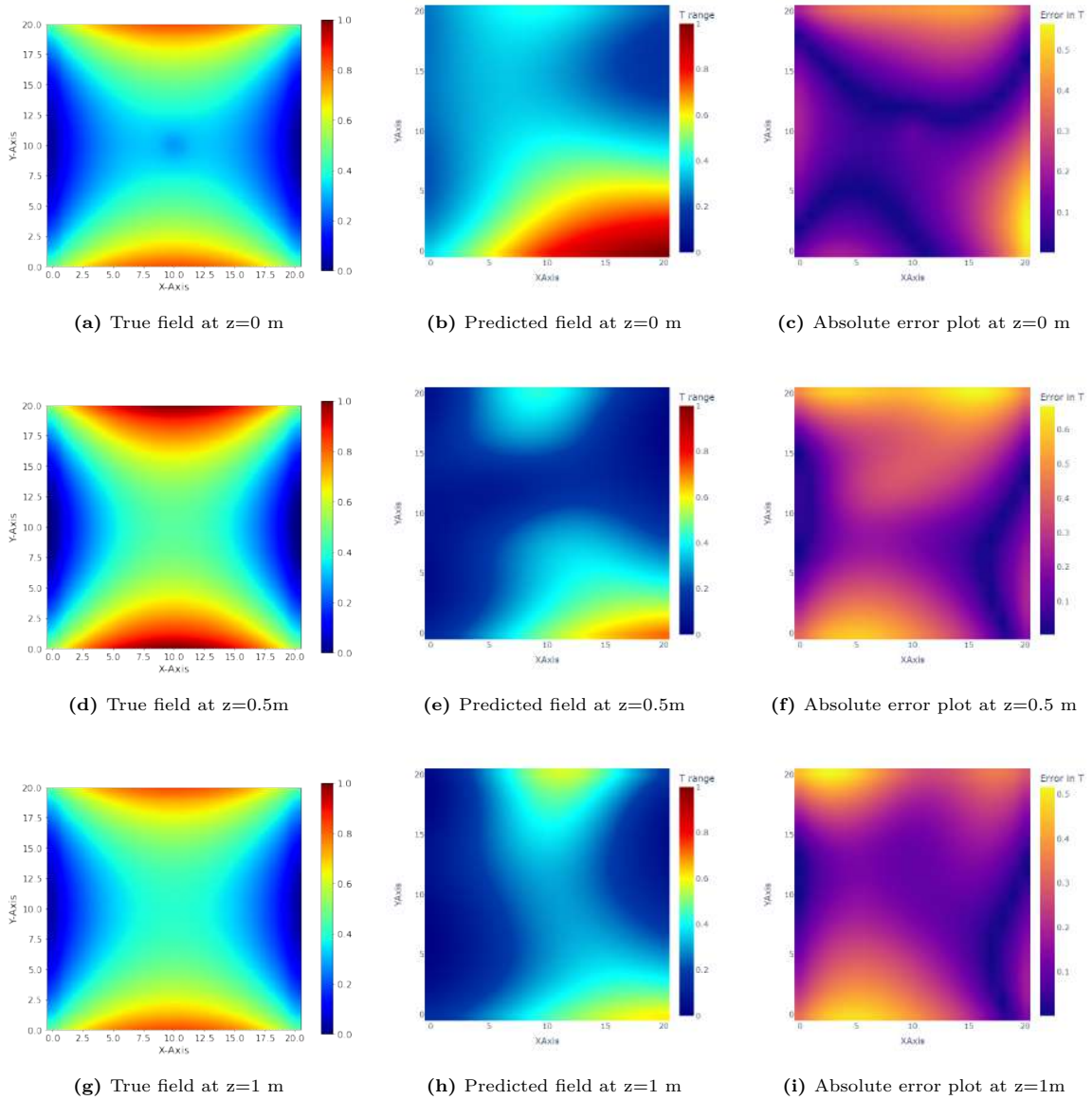
Albeit, the above-described approach is completely different (differential approach), the end motive of flow visualization has similarities to this thesis. Hence, a comparative study was pursued to evaluate HFM capabilities against source frameworks for temperature field predictions; using the same limited sensor dataset. For this performance comparison, the HFM algorithm is retrieved from [13] and modified for a steady-state situation to test on Virtual Experiments 3.1 and 3.2. Results are then compared to source framework results presented in Chapter 6. It is worth noting that for this study, we solely focussed on temperature and did not assess the velocity or pressure prediction capabilities of the algorithm. Additionally, only the time component is excluded, and the training data is constrained to a range of 8-15 measurements to check algorithm resilience beyond its tested range. However, the remaining aspects, such as the algorithm training methodology, remain unchanged from Raissi *et al.*'s original work in [9],[13].

7.2. Virtual Experiment 3.1: HFM Predictions

To recall briefly, the temperature variations for Virtual Experiment 3.1 are nearly smooth everywhere except near the inlet. Generic and Partition Frameworks identified the patterns satisfactorily except for the inlet position, where the local error shot up. The HFM algorithm is trained by inputting sensor measurements in a similar range as provided to train the Generic and Partition Source Framework for a fair comparison. For VE 3.1, initially, 8 and 119 sensor measurements were utilized to train the HFM

algorithm for temperature predictions. The predictions were tested on the grid presented in subsection 3.4.2. The predicted temperature values were normalized to aid the comparison process.

The algorithm was unable to capture the field with just 8 measurements, as shown in Figure 7.3, where the regressed temperature field and corresponding absolute error plots are depicted on various XY planes. The mean absolute error is measured to be 0.228 throughout the domain, which is high. The absolute error also rises to 0.5 locally in certain regions within the domain. In this scenario, even with just 8 sensor measurements, both the Generic and Partition Frameworks are at least capable of capturing the fundamental features of the domain, except for the inlet.



(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

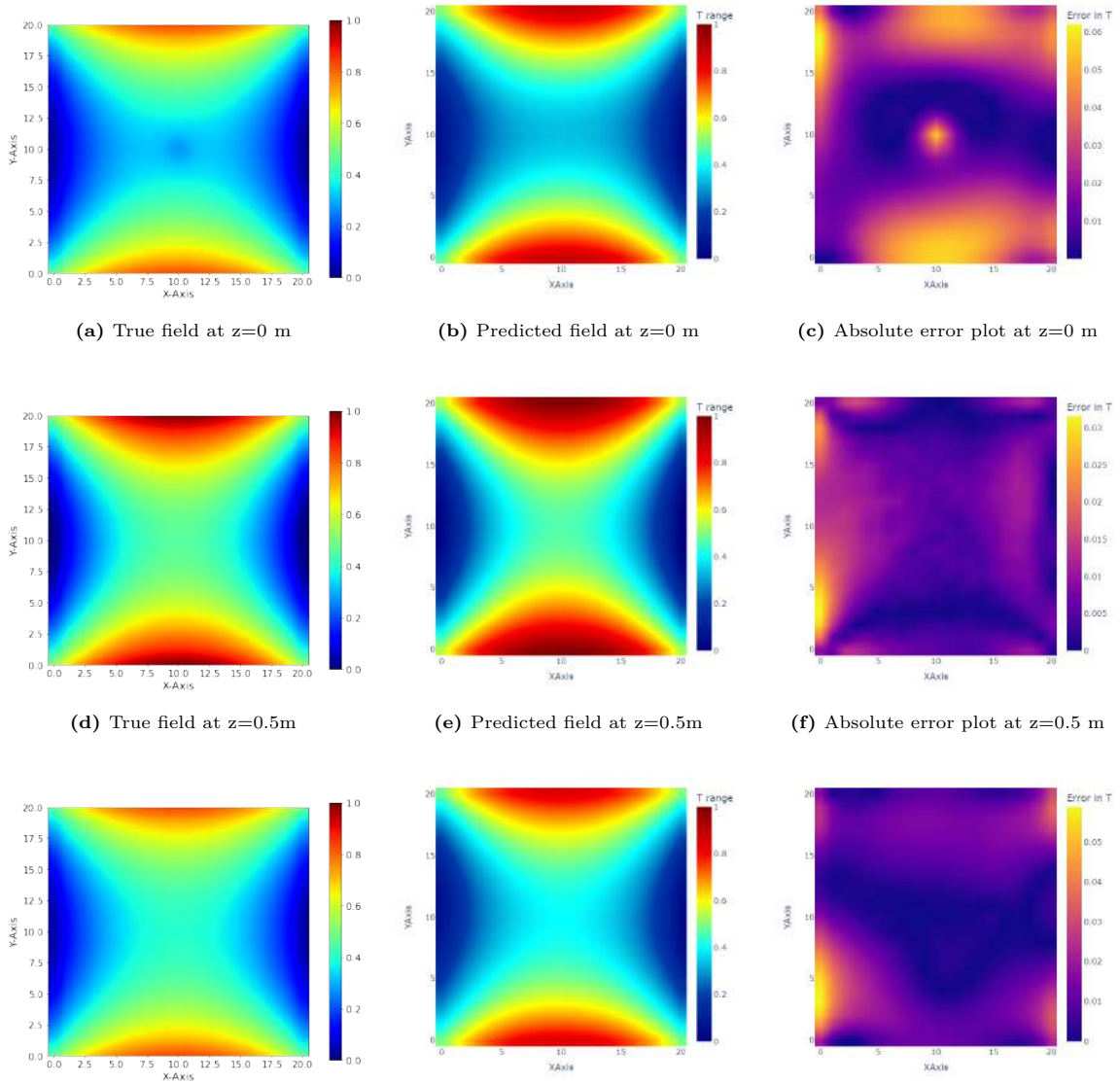
(d) True field at z=0.5m **(e)** Predicted field at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field at z=1 m **(h)** Predicted field at z=1 m **(i)** Absolute error plot at z=1m

Figure 7.3: HFM Algorithm assisted predictions: XY Temperature Plots of Virtual Experiment 3.1 and respective temperature error plots. 8 sensor measurements were used for the training.

Subsequently, the input training data was increased to 119 measurements to evaluate whether this would lead to improved predictions. At this stage, the HFM predictions are equivalent to the results

obtained using source frameworks. However, it does not clearly identify the inlet position in the domain. This can be realized with the help of XY planes presented in Figure 7.4. The mean absolute error falls down to 0.009. Even, locally the error is not higher than 0.06. Although, when the computational efforts are compared, HFM is considerably more expensive than Source Frameworks for achieving equivalent quality results.



(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

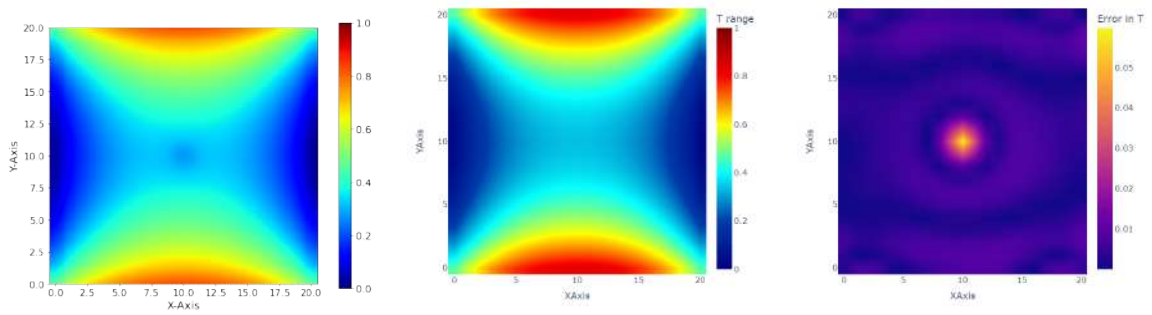
(d) True field at z=0.5m **(e)** Predicted field at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field at z=1 m **(h)** Predicted field at z=1 m **(i)** Absolute error plot at z=1m

Figure 7.4: HFM Algorithm assisted predictions: XY Temperature Plots of Virtual Experiment 3.1 and respective temperature error plots. 119 sensor measurements were used for the training.

Later, a larger training data set was utilized, and HFM delivers the output with accurate values and even lower MAE. However, even with nearly 9000 training data input, it could not specifically predict the location of the inlet as shown in Figure 7.5. Additionally, generating a larger training set in a short period requires many sensor devices. It is not desirable to have many sensor devices in a flow process since it could hinder the physical phenomena. It also increases the computational time to make sufficient iterations required for training. In this situation, the source frameworks are a better

alternative for making decent predictions.



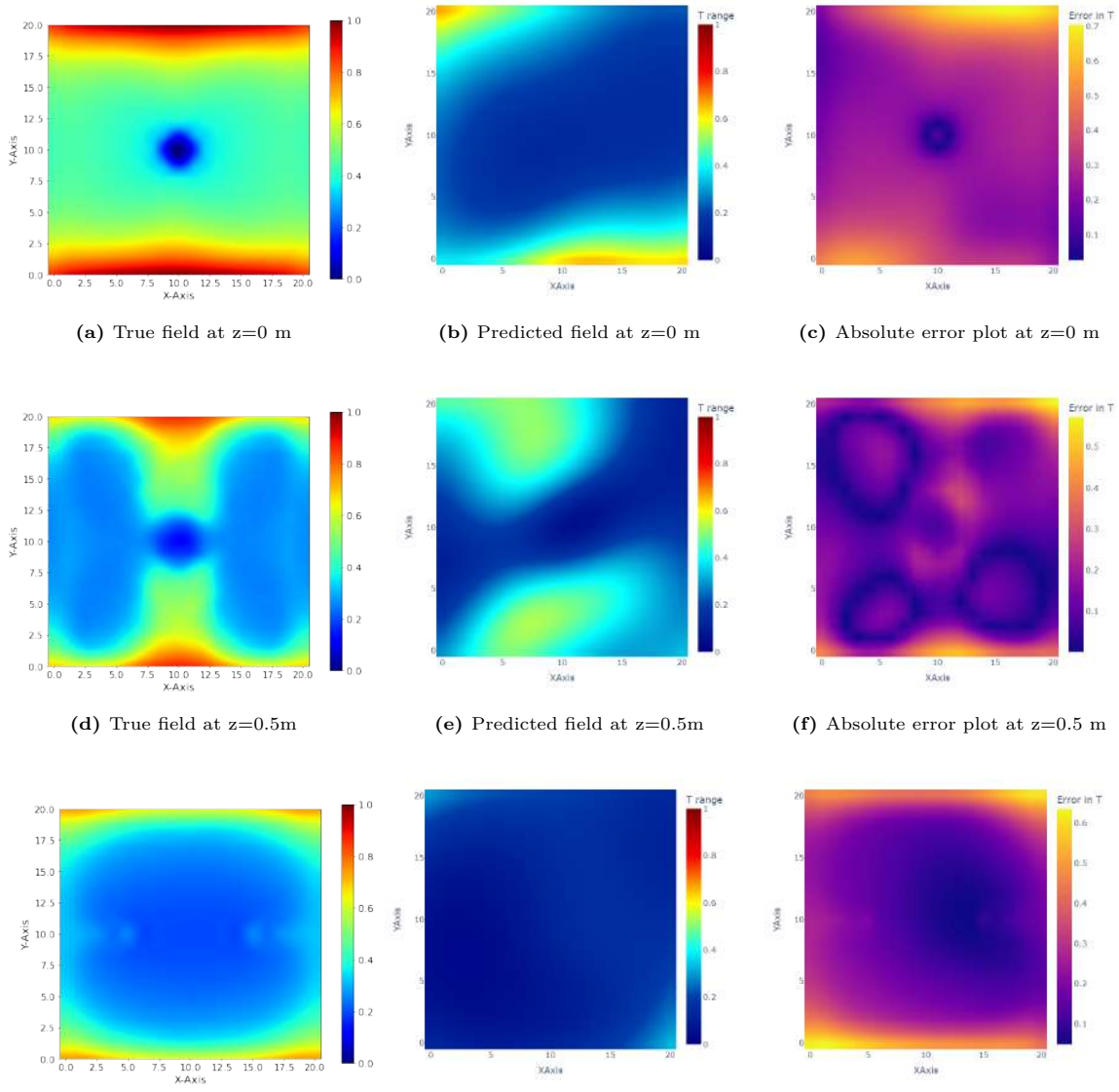
(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

Figure 7.5: HFM Algorithm assisted prediction: XY Temperature Plot of Virtual Experiment 3.1 and respective temperature error plot at z=0 m. Approximately, 9000 sensor measurements were used for the training.

7.3. Virtual Experiment 3.2: HFM Predictions

In this experiment, the convective currents are prominent. Generic and Partition Framework were too primitive to identify such acute variations in the field. It is interesting to assess HFM's predictive capabilities for this experiment with the same sensor dataset and check if it delivers satisfactory output. Here, 15 and 161 sensor measurements were used to train the HFM algorithm.

The algorithm fails to predict the field with 15 measurements training, as represented in Figure 7.6, where the regressed temperature field and corresponding temperature error plots are depicted on various XY planes. The mean absolute error is measured high throughout the domain and is equal to 0.1614. There is no specific trend observed in the error plots to identify the exact regions in the domain for poor field prediction. With 161 measurements training, the MAE reduces to 0.113, as expected with an increase in input data. However, it continues to remain in the higher error range. At this error level, the algorithm generates distorted predictions on the field. This can be noticed in Figure 7.7. One difference between HFM and Source Frameworks is that the HFM predictions are noise-free even at such high error values. As observed earlier, GSF and PSF predictions show considerable noise build-up in the predicted field plots at the same error range.

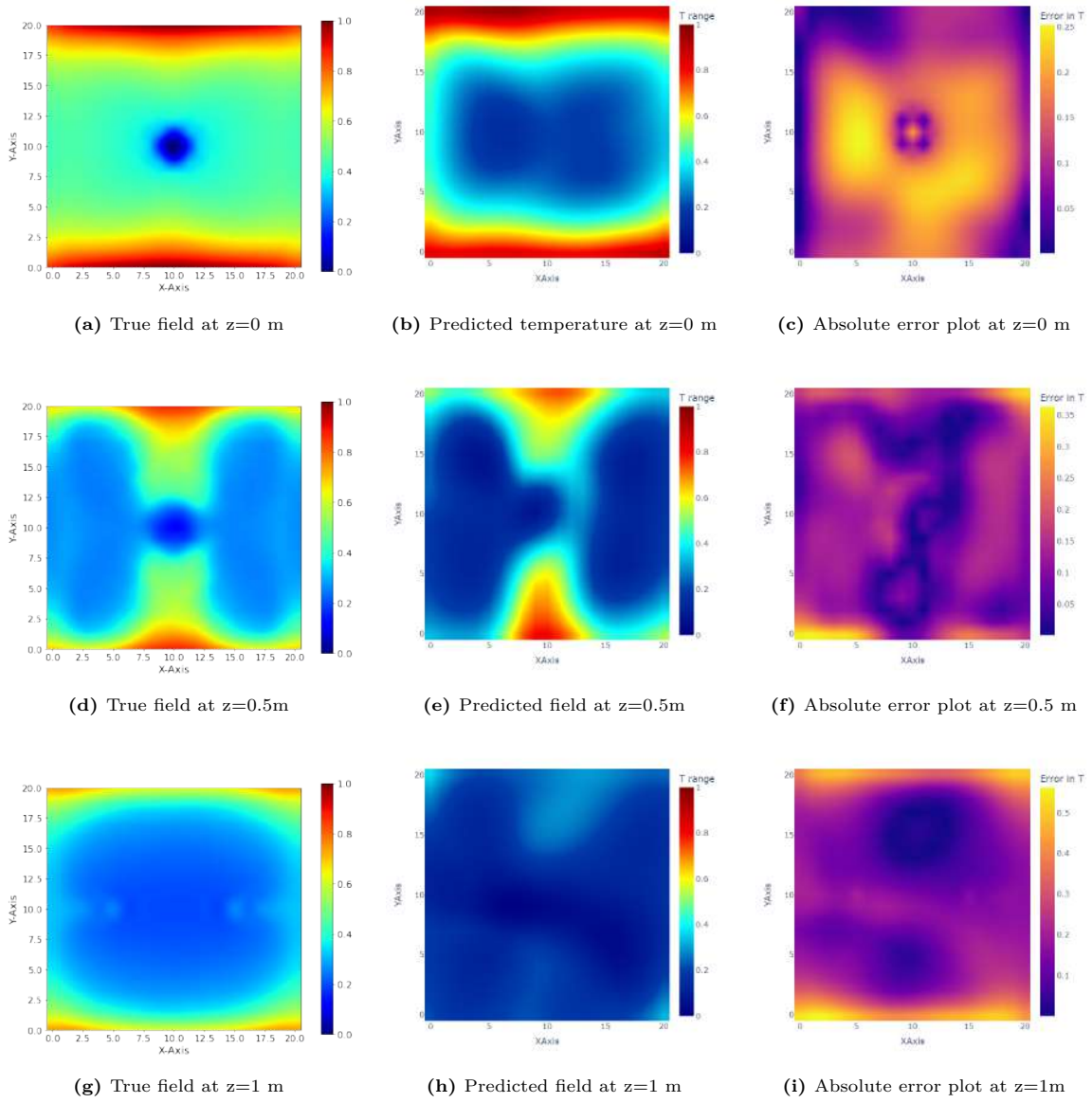


(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field at z=0.5m **(e)** Predicted field at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field at z=1 m **(h)** Predicted field at z=1 m **(i)** Absolute error plot at z=1m

Figure 7.6: HFM Algorithm assisted predictions: XY Temperature Plots of Virtual Experiment 3.2 and respective temperature error plots. 15 sensor measurements were used for the training.



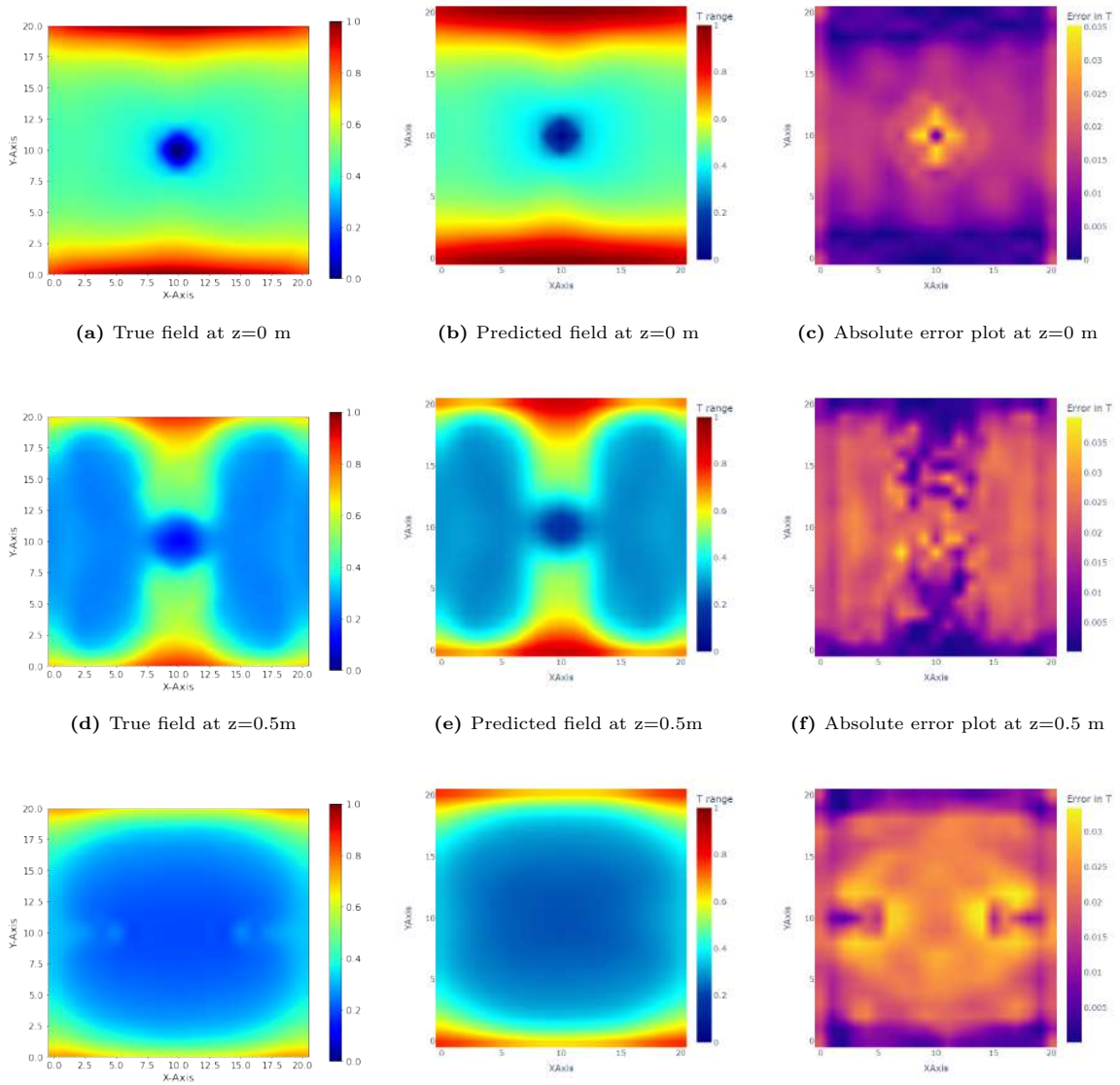
(a) True field at z=0 m **(b)** Predicted temperature at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field at z=0.5m **(e)** Predicted field at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field at z=1 m **(h)** Predicted field at z=1 m **(i)** Absolute error plot at z=1m

Figure 7.7: HFM Algorithm assisted predictions: XY Temperature Plots of Virtual Experiment 3.2 and respective temperature error plots. 161 sensor measurements were used for the training.

After training the algorithm with a smaller dataset, a larger training input of approximately 9000 data points was used to assess its predictive performance. The results are presented in Figure 7.8, which shows the predicted XY temperature planes and the respective temperature error plots. The predictions capture the convective currents well. The mean absolute error (MAE) across the domain is as low as 0.0183, indicating that the algorithm performs well in predicting the temperature field. However, as mentioned earlier, generating a large training set in a short time span requires many sensor devices. This may not always be desirable in a flow process, as it may interfere with the physical phenomena being studied and also increase the computational time required for sufficient training iterations required for the HFM technique.



(a) True field at z=0 m **(b)** Predicted field at z=0 m **(c)** Absolute error plot at z=0 m

(d) True field at z=0.5m **(e)** Predicted field at z=0.5m **(f)** Absolute error plot at z=0.5 m

(g) True field at z=1 m **(h)** Predicted field at z=1 m **(i)** Absolute error plot at z=1m

Figure 7.8: HFM Algorithm assisted predictions: XY Temperature Plots of Virtual Experiment 3.2 and respective temperature error plots. Approximately, 9000 sensor measurements were used for the training.

7.4. Comparison: HFM versus Source Frameworks Predictions

The study by Raissi *et al.* [9] showed that their algorithm can provide accurate predictions when trained on large datasets. The lowest spatial resolution tested by the researchers was 250 data points. The present study challenged the HFM algorithm's resilience by reducing the input data below the tested spatial-temporal resolution. It is found that the algorithm breaks down at such reduced sensor measurements, which is supported by Figures 7.3, 7.4, 7.6, and 7.7. In contrast, source frameworks require minimal data input for delivering decent predictions with as low as 8 sensor measurements for smooth fields. However, as the problem complexity increases, source frameworks tend to have noisy output, while the HFM algorithm provides broken but noise-free outputs.

The HFM algorithm has not been implemented for predicting the systems that involve sources, consumption, or reactions in the governing equations. While the source frameworks involve the presence of source potentials in the domain to predict heat effects. The HFM technique is developed to handle conduction-convection systems and source frameworks are not yet sufficiently developed for convective domains. Furthermore, the source frameworks can utilize physical experimental data using flow-following sensors, whereas HFM is not guaranteed the same. Even if it may, a higher number of sensor devices are required to generate sufficient datasets in a given time frame compared to the sensors required for generating source framework training datasets.

A TU Delft Desktop, equipped with an Intel Core i5 (64-bit) processor and 8GB RAM, required approximately 40 hours to complete nearly 31,500 iterations for the HFM algorithm. Utilizing Google Colab with a T4 GPU as a hardware accelerator completed an equivalent number of HFM training iterations in just 4 hours. In contrast, the GSF-assisted predictions could be generated within a maximum of 15 minutes, while the PSF-LS predictions took up to 2 hours on the TU Delft Desktop. Hence, HFM is considerably more computationally expensive than the source frameworks, including additional computational efforts required for the generation of training data.

The HFM technique and its neural network architecture are promising, with an ingeniously structured loss function. However, it is not possible to make predictions outside the training domain, and it is also not viable to train the algorithm for all practical situations. Moreover, it does not seem to have added benefits in terms of gaining knowledge on a problem for which state-of-the-art 'forward' solutions exist (i.e., CFD). Utilizing HFM for a problem may therefore be a futile exercise when the existing 'forward' solution set is fed as training data to the algorithm for generating an 'inverse' solution. From our perspective, it is an addition to the computational expense. Alternatively, neural networks can be useful in the optimization of the source frameworks. In place of Navier-Stokes Equations, source frameworks can be incorporated into the network architecture. Such networks can then utilize sensor data to determine the size (*Issue 2*) or location (*Issue 3*) of the hypothetical source potentials. This would then be our version of 'Physics-informed Neural Networks'.

8

Conclusion and Outlook

Chemical industries often deal with complicated processes involving various unit operations and processes. Monitoring and controlling these processes can be challenging at times, which provided the motivation for this thesis. This study aimed to visualize scalar (or, vector) fields in physical systems using an integral approach for field reconstruction. As an example, temperature field visualizations were studied in heat-generative and forced convection setups with the help of sparse sensor measurements. To realize this broad objective, certain conceptual and numerical goals had been identified and discussed in Section 1.3.3. This chapter concludes with an overview of the findings and future prospects for physics-constrained and data-driven field reconstruction.

8.1. Conceptual and Numerical Goals

An *a priori* integral approach is implemented to build a temperature field function. The field function comprises a combination of basis functions. These basis functions are rooted in the Helmholtz-Hodge Decomposition (HHD) and its inverse. Certain assumptions are considered for the system studied as mentioned in Section 1.3.1. With these system specifications, the temperature field is deduced to vary as a harmonic-scalar potential function. In such a system, conditions inside the domain and on the boundaries influence the overall temperature distribution in the domain.

To examine the harmonic-divergent temperature field, a distribution of hypothetical heat potentials is assumed. The basis functions are associated with the nature of the heat potentials. As a harmonic potential, the temperature is inversely proportional to the distance from the source (Eq 2.29). While as a scalar potential, the temperature is directly proportional to the square of the distance from the source (2.30). The resulting temperature field function is a non-linear equation that is simplified into a linear equation with the help of Source Frameworks. Source Frameworks tackle *Issues* relating to the number, size, and location of these hypothetical potentials. Sparse sensor measurements are used to evaluate the remaining 'unknowns' in a linear function.

Finally, framework-assisted temperature predictions are carried out for a few virtual experiments. Results indicate that uniform smooth temperature fields, without any sudden spikes or acute variations, are well predicted with minimal error and computational efforts. Fair predictions can be achieved with

as low as 8 sensor measurements. This brings confidence in the *a priori* integral methodology applied to build this technique.

However, as the domain complexity increases, such as in the case of a locally uneven field in VE2, this simplistic approach proves inadequate in capturing the specific characteristics of the heat source and only recognizes the dominant phenomena within the domain. The frameworks homogenize the local effects. When dealing with a weak convective field, the predictions are similar to those of a local heat source, where in both situations heat conduction becomes the dominant mechanism. While in strong convective systems, the temperature variations become sudden and sharper, falling beyond the framework's ability to identify them accurately.

In the end, an interesting case on framework performance comparison with the HFM algorithm (a recent development based on a differential approach) is included. Utilizing HFM for a simple problem with a limited sensor dataset is concluded to be a futile exercise in comparison to the source framework capabilities. With a considerably large dataset and higher computational expense, HFM is capable to deliver lower error values and better predictions.

8.2. Potential Improvements and Outlook

While the source framework assisted in achieving good predictions of the smooth well-distributed temperature field, it is not sufficient to identify local effects or sharp variations. Framework possibilities need to be looked at for the sophistication of this generalized reconstruction technique. There is a potential of developing these frameworks more versatile by not pre-fixing *Issues*, for e.g. the radius or location of the hypothetical sources. These *Issues* will be then solved by a non-linear optimization technique using sensor measurements. Additionally, some *a priori* information can provide more intelligence to hypothetical heat sources. For e.g., exothermic chemical reactions in the heat source cases or velocity field in forced convection cases. Complexities can be added on gradually, like introducing transient state or chemical reactions in the domain of interest. Another compelling pursuit for framework optimization could be developing a smart variety of Physics-Informed Neural Networks combined with concepts from this work. This can be achieved by incorporating our physics-constrained field function into the network's architecture and (or) loss function.

References

[1] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational physics* 378 (2019), pp. 686–707.

[2] Jonas Bisgaard et al. "Flow-following sensor devices: A tool for bridging data and model predictions in large-scale fermentations". In: *Computational and Structural Biotechnology Journal* 18 (2020), pp. 2908–2919.

[3] Sebastian Felix Reinecke et al. "Process characterization in industrial vessels by flow-following sensor particles". In: *Measurement Science and Technology* 33.9 (2022), p. 095106.

[4] Erik HA Duisterwinkel et al. "Go-with-the-flow swarm sensing in inaccessible viscous media". In: *IEEE sensors journal* 20.8 (2019), pp. 4442–4452.

[5] Míriam R García et al. "Optimal field reconstruction of distributed process systems from partial measurements". In: *Industrial & engineering chemistry research* 46.2 (2007), pp. 530–539.

[6] Helin Gong et al. "Optimal and fast field reconstruction with reduced basis and limited observations: Application to reactor core online monitoring". In: *Nuclear Engineering and Design* 377 (2021), p. 111113.

[7] Roberto Ponciroli, Andrea Rovinelli, and Lander Ibarra. "A Convolutional Neural Network-based Approach to Field Reconstruction". In: *arXiv preprint arXiv:2108.13517* (2021).

[8] Victor Coppo Leite et al. "A Study on Convolution Neural Network for Reconstructing the Temperature Field of Wall-Bounded Flows". In: *arXiv preprint arXiv:2202.00435* (2022).

[9] Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. "Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations". In: *Science* 367.6481 (2020), pp. 1026–1030.

[10] Kai Fukami et al. "Global field reconstruction from sparse sensors with Voronoi tessellation-assisted deep learning". In: *Nature Machine Intelligence* 3.11 (2021), pp. 945–951.

[11] R.S. Devarasetty. "Field Re-construction with Flow-Following Sensors: A Proof of Concept for a Scalar Field". In: *MSc Thesis* (2022).

[12] H.B. Phillips. *Vector Analysis*. 1948.

[13] Maziar Raissi. *maziarraissi/HFM: Hidden Fluid Mechanics*. Version v1.0. Dec. 2019. DOI: `10.5281/zenodo.3566161`. URL: `https://doi.org/10.5281/zenodo.3566161`.

A

Python Codes

A.1. Python Libraries

The list of Python libraries utilized for developing algorithms in this study is as follows:

```
import numpy as np
print("Successfully imported %s -- Version: %s"%(np.__name__ ,np.__version__))
import scipy
print("Successfully imported %s -- Version: %s"%(scipy.__name__ ,scipy.__version__))
import matplotlib.pyplot as plt
print("Successfully imported %s"%plt.__name__)
import pandas as pd
print("Successfully imported %s -- Version: %s"%(pd.__name__ ,pd.__version__))
import sympy as sym
print("Successfully imported %s -- Version: %s"%(sym.__name__ ,sym.__version__))
from scipy import optimize
print("Successfully imported %s"%optimize.__name__)
from numpy import linalg as LA
print("Successfully imported %s"%LA.__name__)
from mpl_toolkits import mplot3d
print("Successfully imported %s"%mplot3d.__name__)
import scipy.interpolate
import math
import statistics
print("Successfully imported %s"%stat.__name__)
import plotly.graph_objects as go
import plotly.io as pio
import random
from itertools import product
import itertools as it
from tqdm import tqdm
```

A.2. Random Sensor Locations and Sensor Dataset

A.2.1. Generating Random Sensor Locations in the Domain of Interest

Using Python, random sensor measurement positions are generated such that sensor movement remains within the domain of interest. These locations are written into a .txt file which can be imported into

COMSOL for finding temperature measurements associated with these positions.

```
#Generating Random Sensor Positions
N = 15  # Number of sensors positions to be generated
length = 1

sensor_positions = []
while len(sensor_positions) < N:
    x = round(2*length*random.uniform(0.05, 0.95)/ 2, 1)
    y = round(2*length*random.uniform(0.05, 0.95)/ 2, 1)
    z = round(2*length*random.uniform(0.05, 0.95)/ 2, 1)
    position = [x, y, z]
    if position not in sensor_positions:
        sensor_positions.append(position)

n = len(sensor_positions)

if n == N:
    print('No duplicate sensor positions')
    print('No. of Sensors =', N)
else:
    print('Duplicate sensor positions')

#Writing random positions into a text file for importing into COMSOL
with open("SensorLocation_15Sensors.txt", "w") as o:
    for line in sensor_positions:
        print ("{} {} {}". format(line[0], line[1], line[2]), file=o)
```

A.2.2. Generating Sensor Dataset

The .txt file containing unique sensor positions is loaded in COMSOL and the temperature values are evaluated at these positions, which are then exported to an Excel file. This Excel file contains Sensor Dataset (Discrete temperature measurements at random locations in the domain), which is read in Python using the pandas library.

```
# Reading Data from COMSOL
data = pd.read_excel('Sensordata_HS12_Q-2.xlsx', sheet_name='Sheet1-Q1+Q2')
data = np.round(data, 4)

NSensors = 15

Temp = data.iloc[:, 0].tolist()
x = data.iloc[:, 1].tolist()
y = data.iloc[:, 2].tolist()
z = data.iloc[:, 3].tolist()

S1 = Temp[:NSensors]
sensor_position = list(zip(x[:NSensors], y[:NSensors], z[:NSensors]))

print(S1)
print(sensor_position)
```

A.3. Hypothetical Heat Source Locations

A.3.1. Defining Source Locations on External Source Planes

This script is an optimized version of the version in [11] and is used to generate external plane source locations.

```
#Generating the Source Positions on External Source Plane for Generic and Partition Framework

def cutplane_centroids(length, cuts):
    l = (2*length)+1
    n = cuts  # number of cuts in each axis
    N = n+2  # number of points in each axis
    X = -length
    Y = 1+length
    P = (X, Y)
    p = np.linspace(X, Y, n+2)
    Area = (l*l)/((n+1) ** 2)

    # Defining all the surfaces
    s = []
    for k in range(2):
        A = list(product(p, repeat=2))
        A = [[p[i], p[j], P[k]] for j in range(len(p)) for i in range(len(p))]

        for i in range(N-1):
            i = N*i
            for j in range(N-1):
                q = i + j
                s.append([A[q], A[q+1], A[q+N], A[q+1+N]])

    for i in range(2):
        A = list(product(p, repeat=2))
        A = [[P[i], p[j], p[k]] for k in range(len(p)) for j in range(len(p))]

        for i in range(N-1):
            i = N*i
            for j in range(N-1):
                q = i+j
                s.append([A[q], A[q+1], A[q+N], A[q+1+N]])

    for j in range(2):
        A = list(product(p, repeat=2))
        A = [[p[i], P[j], p[k]] for k in range(len(p)) for i in range(len(p))]

        for i in range(N-1):
            i = N * i
            for j in range(N-1):
                q = i+j
                s.append([A[q], A[q+1], A[q+N], A[q+1+N]])

    NS = len(s)  # number of surfaces

    # Centroid of each surface
    cs = []
    for i in range(NS):
        coords = np.array(s[i])
        centroid = np.round(np.mean(coords, axis=0), 4)
```

```
        cs.append(centroid)

    return np.asarray(cs)


def extframework(length, cuts):
    cent = cutplane_centroids(0, cuts)
    n = cent.shape[0]
    for i in range(n):
        if i<(n//6):
            cent[i][2] -= length
        elif i>=(n//6) and i<(2*n//6):
            cent[i][2] += length
        elif i>=(2*n//6) and i<(3*n//6):
            cent[i][0] -= length
        elif i>=(3*n//6) and i<(4*n//6):
            cent[i][0] += length
        elif i>=(4*n//6) and i<(5*n//6):
            cent[i][1] -= length
        elif i>=(5*n//6) and i<n:
            cent[i][1] += length

    return np.round(cent, 2)

extsource_position = extframework(1, 1)
#print(extsource_position)
```

A.3.2. Defining Source Locations in the Domain

Using this Python Script, source locations inside the domain at centroid positions can be generated. This script is used by both source frameworks (GSF and PSF) to define source locations within the domain.

```
#Defining centroid positions in the domain with varying cuts for Generic and Partition Source
     Framework

def intframework(n):

    p0=np.array([0,0,0])
    p1=np.array([1,1,1])
    l=1
    cut0=l/((2*n)+2)
    cut=[]
    for i in range(n+1):
        cut1=(2*i+1)*cut0
        cut.append(cut1)

    points=it.product(cut,repeat=3)
    pc=[]
    for j in list(points):
        pc.append(j)

    ncubes=np.power((n+1),3)
    print('Coordinates of Centroids for {d} sub-cubes with {e}-cuts along each axis:'.format(
        d=ncubes, e=n), pc)
    return pc
```

```
22
23 intsource_position=intframework(1)
24 #print(intsource_position)
```

A.4. Generic Source Framework: Exact Solution

Sensor datasets can be used to solve for the unknown source intensity terms by generating a system of linear equations. These equations are formulated by the Generic Framework (Eq. 3.1) and can be solved with an exact solution technique (matrix inverse multiplication) as follows:

$$T_{sensor} = Dist \cdot C \tag{A.1}$$

$$C = Dist^{-1} \cdot T_{sensor} \tag{A.2}$$

where,

- T_{sensor} contains sensor temperature measurements; matrix size: ($NSensors$ x 1)
- $Dist$ contains computed $\frac{1}{r}$ and r^2 values between sensor measurement point and source locations; matrix size: ($NSensors$ x $NSensors$)
- C contains all the unknowns (A_i, B_i, and $T0$) to be solved; matrix size: ($NSensors$ x 1)

```
1  def exactsolution(NSensors, S1, extsource_position, intsource_position):
2      dista=[]
3      for j in range(NSensors):
4          for n in range(np.shape(extsource_position)[0]):
5              r_inv= 1/math.dist(sensor_position[j],extsource_position[n])
6              dista.append(r_inv)
7
8      dista= np.resize(dista,(NSensors,np.shape(extsource_position)[0]))
9      #print(dista)
10     #print(np.shape(dista))
11
12     distb=[]
13     for k in range(NSensors):
14         for m in range(np.shape(intsource_position)[0]):
15             r_2=np.square(math.dist(sensor_position[k],intsource_position[m]))
16             distb.append(r_2)
17
18     distb= np.resize(distb, (NSensors, np.shape(intsource_position)[0]))
19     #print(distb)
20     #print(np.shape(distb))
21
22     distc= np.ones((NSensors,1))
23     #print(distc)
24     #print(np.shape(distc))
25
26     dist= np.concatenate((dista,distb,distc), axis=1)
27     #print(np.shape(dist))
28     T_Sensor = np.array(S1)
29     T_Sensor = np.resize(T_Sensor,(NSensors,1))
30     #print(T_Sensor)
31     C= np.linalg.solve(dist, T_Sensor) #C contains intensities' values
32
33     A=C[0:np.shape(extsource_position)[0]] #A contains all outside sources' intensity values
```

```
    B=C[np.shape(extsource_position)[0]:(NSensors-1)] #B contains all inside sources' intensity values
    T0=C[-1] #T0 Reference Temperature
    return C, A, B, T0

A= exactsolution(NSensors,S1, extsource_position, intsource_position)[1]
B= exactsolution(NSensors,S1, extsource_position, intsource_position)[2]
T0= exactsolution(NSensors,S1, extsource_position, intsource_position)[3]
```

A.5. Partition Source Framework

A.5.1. Defining Inside and Outside Source-Sensor Distance for Partition Source Framework

```
#Defining Inside and Outside Source-Sensor Distance for Partition Source Framework
def psf(sensor_position, intsource_position, extsource_position, int_cuts):
    #int_cuts are cuts on the domain
    l=1
    l= (1/(int_cuts+1))
    R0 = (np.sqrt(3)/2)*l

    dist_int=[]

    for i in range(np.shape(intsource_position)[0]):
        for j in range(np.shape(sensor_position)[0]):
            r=math.dist(sensor_position[j], intsource_position[i])
            if (r <= R0):
                r2= np.power(r,2)
                dist_int.append(r2)
            else:
                r_inv= 1/r
                R0_rinv1= (np.power(R0,3))*(r_inv)
                dist_int.append(R0_rinv1)

    dist_int =np.reshape(dist_int, ((np.shape(sensor_position)[0]), (np.shape(intsource_position)[0])))


    dist_ext=[]
    for j in range(np.shape(sensor_position)[0]):
        for n in range(np.shape(extsource_position)[0]):
            r_inv= 1/math.dist(sensor_position[j],extsource_position[n])
            #R0_rinv2= (np.power(R0,3))*(r_inv)
            dist_ext.append(r_inv)

    dist_ext= np.reshape(dist_ext, ((np.shape(sensor_position)[0]), (np.shape(extsource_position)[0])))

    d = np.concatenate((dist_int, dist_ext), axis=1)
    I= np.ones((np.shape(sensor_position)[0],1))
    dist= np.concatenate((d, I), axis=1)


    return dist_int, dist_ext, dist

```

```
di=psf(sensor_position, intsource_position, extsource_position, 2)[0]
de=psf(sensor_position, intsource_position, extsource_position, 2)[1]
d= psf(sensor_position, intsource_position, extsource_position, 2)[2]
```

A.5.2. PSF: Exact Solution

```
def psfexactsolution(NSensors, S1, d):
    T_Sensor = np.array(S1)
    T_Sensor = np.resize(T_Sensor,(NSensors,1))
    ES= np.linalg.solve(d, T_Sensor) #ES contains unknown intensities' values
    return ES

ES= psfexactsolution(NSensors, S1, d)
A= ES[0:(np.shape(intsource_position)[0])]
B= ES[(np.shape(intsource_position)[0]):(-1)]
T0= ES[-1]
```

A.5.3. PSF: Least Squares Solution

For finding the least-squares solution, *np.linalg.lstsq* from numpy library is utilized. It solves the equation $Dist \cdot C = T_{sensor}$ by computing a vector C that minimizes the Euclidean 2-norm (or, L2 norm) $||T_{sensor} - Dist \cdot C||$. The equation may be under-determined or over-determined (i.e., the number of linearly independent rows of a can be less than or greater than its number of linearly independent columns). In case, $Dist$ is a full-rank square matrix, then C (excluding round-off error) is the 'exact' solution of the equation. If there are multiple minimizing solutions, the one with the smallest 2-norm $||C||$ is returned.

```
def lstsqsolution(NSensors, S1, d):
    T_Sensor = np.array(S1)
    T_Sensor = np.resize(T_Sensor,(NSensors,1))
    ES, *_= np.linalg.lstsq(d, T_Sensor) #ES contains intensities' values
    return ES

ES = lstsqsolution(NSensors, S1, d)
A= ES[0:(np.shape(intsource_position)[0])]
B= ES[(np.shape(intsource_position)[0]):(-1)]
C= ES[-1]
```

A.6. True Data-grid and True-Mapping Dataset

A.6.1. True Data-grid

For generating a true data grid of temperature values (mentioned in Section 3.4.2), the following code is implemented to form a (21 x 21 x 21) grid of data points, this will be later written into a .txt file. The .txt file is loaded onto COMSOL for evaluating the temperature values at the grid points. The evaluated values are then exported as an Excel file which contains our 'True Mapping Dataset' for comparison.

```
#Code for True Temperature Data-Grid for Comparision
def ComparisionGrid(cuts):
    gridpoints=np.linspace(0,1,cuts)
    E=[]
    for i in range(np.shape(gridpoints)[0]):
        for j in range(np.shape(gridpoints)[0]):
            for k in range(np.shape(gridpoints)[0]):
```

```
                p=[gridpoints[k],gridpoints[j],gridpoints[i]]
                E.append(p)
    return E

grid= ComparisionGrid(21)

#Writing into a text file for COMSOL- RUN ONLY ONCE
with open("XYZGrid_ComparisionPoints.txt", "w") as o:
    for line in ComparisionGrid(21):
        print ("{} {} {}". format(line[0], line[1], line[2]), file=o)
```

A.6.2. True-Mapping Dataset

The Excel file containing True-Mapping Dataset is read with the help of the pandas library. The true values are normalized between 0 to 1 so that it is easier to compare results across different virtual experiments.

```
#Actual Temperatures (Expected Temperature Field) for Comparision
data2=pd.read_excel("Sensordata_HS12_Q-2.xlsx",sheet_name ='Sheet2-Q1+Q2')
TR= data2.values.tolist()
#print(np.shape(TR))

TRmin= min(TR)[0]
TRmax= max(TR)[0]
trnd= TRmax- TRmin
ndTR = []

for i in range(len(TR)):
    TR[i] = TR[i][0] - TRmin
    nondimTR = TR[i]/trnd
    ndTR.append(nondimTR)

ndTR = np.resize(ndTR, (9261,1))
#print(np.shape(ndTR))
#print(max(ndTR))
```

A.6.3. True (Actual) Normalized Temperature Field Distribution on an XY Plane

```
def true_plot(Z):
    x = np.linspace(0.0, 1.0, 21, endpoint=True)
    y = np.linspace(0.0, 1.0, 21, endpoint=True)
    z = Z

    g = np.linspace(0.0, 1.0, 21, endpoint=True)
    I = g.tolist().index(z)

    TR1 = ndTR[441 * I:441 * (I + 1)]
    data = TR1

    data = [data[n:n+21] for n in range(0, len(data), 21)]

    fig, ax = plt.subplots(figsize=(10, 8))
    im = ax.imshow(data, cmap="jet", interpolation='lanczos')
    ax.set_ylim(0, 20)
    ax.set_xlabel("X-Axis", fontsize=18)
```

```
    ax.set_ylabel("Y-Axis", fontsize=18)
    ax.tick_params(axis='both', which='major', labelsize=16)
    cbar = plt.colorbar(im)
    cbar.ax.tick_params(labelsize=16)
    plt.show()
    return I, plt.show()

tp = true_plot(0.75)
```

A.7. Source Framework-predicted Temperature Field Distribution

Once the unknowns in the framework-formulated temperature field function are solved and known, the temperature at any position within the domain can be evaluated. Such evaluations are known as 'Predicted Temperature' or 'Temperature Predictions'. Similar to the true dataset, the predicted dataset is also normalized.

A.7.1. Generic Framework: Predicted XY Temperature Field

```
#Generic framework-predicted temperature
def predictedT(grid, A, B, T0):
    T_pred1=[]
    for y in tqdm(range(np.shape(grid)[0])):
        position=grid[y]
        def dis(source):
            return 1/math.dist(position, extsource_position[source])

        def dis1(pcp):
            return np.square(math.dist(position, intsource_position[pcp]))

        T=T0
        for k in range(np.shape(extsource_position)[0]):
            T=T+(A[:][k]*dis(k))
            T=np.round(T,2)

        for h in range(np.shape(intsource_position)[0]):
            T=T+ (B[:][h]*dis1(h))
            T=np.round(T,2)

        T_pred1.append(T)

    Tpred_min= min(T_pred1)
    Tpred_max= max(T_pred1)

    tprednd= Tpred_max - Tpred_min

    T_pred = (T_pred1 - Tpred_min)/tprednd
    return T_pred1, T_pred

Tpred=predictedT(grid,A,B,T0)[1]
#print(max(Tpred))
#print(np.shape(Tpred))
```

A.7.2. Partition Framework: Predicted XY Temperature Field

```
#Partition framework-predicted temperature
def predictedT(grid, intsource_position, extsource_position, A, B, T0, ES):
    disr= psf(grid, intsource_position, extsource_position, 2)[2]
    di=psf(grid, intsource_position, extsource_position, 2)[0]
    de=psf(grid, intsource_position, extsource_position, 2)[1]
    T_pred1= np.dot(di, A) + np.dot(de, B) + C
    #Alternatively, the temperature can be predicted with the below equation also
    #T_pred1 = np.inner(disr, ES.T)

    Tpred_min= min(T_pred1)
    Tpred_max= max(T_pred1)

    tprednd= Tpred_max - Tpred_min

    T_pred = (T_pred1 - Tpred_min)/tprednd
    return disr, T_pred1, T_pred

Tpred=predictedT(grid, intsource_position, extsource_position, A, B, T0, ES)[2]
```

A.7.3. Predicted Temperature Field Distribution on an XY Plane

```
#(Predicted) Plot at any XY Plane
x = np.linspace(0.0, 1.0, 21, endpoint=True)
y = np.linspace(0.0, 1.0, 21, endpoint=True)
z = 0.5

g = np.linspace(0.0, 1.0, 21, endpoint=True)
T = predictedT(grid, A, B, T0)[1] #When using GSF
#T = predictedT(grid, intsource_position, extsource_position, A, B, T0, ES)[2] #When using
    PSF
I = g.tolist().index(z)

data = T[441 * I:441 * (I + 1)]
data1 = [item for sublist in data for item in sublist]

print("Mean Temp:", np.round(statistics.mean(data1), 2))
print(np.round(min(data1), 2))
print(np.round(max(data1), 2))
data = [data1[n:n+21] for n in range(0, len(data1), 21)]
trace = go.Heatmap(z=data, colorscale='jet', zsmooth='best', colorbar=dict(title='T range'))
data2 = [trace]

layout = go.Layout(
    xaxis=go.layout.XAxis(title=go.layout.xaxis.Title(text='XAxis', font=dict(size=18))),
    yaxis=go.layout.YAxis(title=go.layout.yaxis.Title(text='YAxis', font=dict(size=18))),
    height=700,
    width=700
)

fig = go.Figure(data2, layout=layout)
fig.update_layout(font=dict(size=16))

pio.show(fig)
```

A.8. Error Metrics and Absolute Error Distribution

A.8.1. Mean Absolute Error (MAE)

Mean Absolute Error (MAE) across the domain is calculated using the following Python script.

```
#Error Analysis: MEAN ABSOLUTE ERROR
def error(Tpred, ndTR):

    ER=[]
    for k in range(np.shape(ndTR)[0]):
        err=0
        err=err+np.abs(Tpred[k]-ndTR[k])
        ER.append(err)

    sumerr=0
    for k in range(np.shape(ER)[0]):
        sumerr=sumerr+ER[k]

    mae=(sumerr/np.shape(ER)[0])
    return mae

mae= error(Tpred, ndTR)
print(mae)
```

A.8.2. Root Mean Square Error (RMSE)

```
#Error Analysis: ROOT MEAN SQUARE ERROR
def rmse(Tpred, ndTR):
    SE=[]
    for k in range(np.shape(ndTR)[0]):
        err=0
        err=err+np.square(Tpred[k]-ndTR[k])
        SE.append(err)

    MSE= np.mean(SE)
    RMSE= np.sqrt(MSE)

    return RMSE

rmse0 = rmse(Tpred, ndTR)
print(rmse0)
```

A.8.3. MAE with varying source plane distance `d' from the domain

For a given cut on the domain and source planes, the distance ‘d’ can be varied to find the optimum positioning of the source planes where the MAE is the least. The example considered here includes a 1-cut on the domain and a 0-cut on the source planes.

```
me = []
me1 = []
intsource_position=intframework(1)
d1= np.linspace(0.1, 0.9, num=9)
d2= np.linspace(1, 9, num=9)
d = np.concatenate((d1, d2), axis=0)

for i in d:
```

```
    extsource_position = extframework(i, 0)
    A, B, T0 = exactsolution(NSensors, S1, extsource_position, intsource_position)[1:]

    grid = ComparisionGrid(21)
    Tpred = predictedT(grid, A, B, T0)[1] #When using GSF
    #T = predictedT(grid, intsource_position, extsource_position, A, B, T0, ES)[2] #When
        using PSF

    mae1 = error(Tpred, ndTR)
    me.append(mae1)

    rmse1 = rmse(Tpred, ndTR)
    me1.append(rmse1)
#print(me)
#print(me1)


print('Minimum MAE=', min(me))
print('Minimum RMSE=', min(me1))
print('Minimum MAE and RMSE at d=', d[me.index(min(me))], 'm')

#Plotting MAE versus source plane distance 'd'

fig, (ax1, ax2) = plt.subplots(2, figsize=(8, 6), dpi=150)
ax1.plot(d[0:9], me[0:9])
ax2.plot(d[9:18], me[9:18])
ax2.set_xlabel('distance of source plan, d [meter]', fontsize=12)
ax2.set_ylabel('Mean Absolute Error, MAE', fontsize=12)
ax1.tick_params(axis='both', which='major', labelsize=10)
ax2.tick_params(axis='both', which='major', labelsize=10)

plt.tight_layout()
plt.show()
```

A.8.4. Optimum Number of Sensor Measurements for Least-Squares Solution

It is important to find the optimal number of sensor measurements which give the best approximate solution using the least-squares technique. Hence, this code varies the number of sensor measurements from 8 to 160, determining the corresponding MAE. The optimum number of measurements is determined when the MAE across the domain is minimized. Furthermore, the distance of source planes 'd' is varied, however, the cuts are fixed; 1-cut on the domain and 0-cut on source planes.

```
data3= pd.read_excel('Sensordata_HS12_Q-2.xlsx', sheet_name='Sheet1-Q1+Q2')

d1= np.linspace(0.1, 0.9, num=9)
d2= np.linspace(1, 9, num=9)
ext_d = np.concatenate((d1, d2), axis=0) #Distance 'd' of source planes from the domain

n = np.arange(8,161,1) #No. of Sensor Measurements

min_mae=[]
min_mae_d=[]

for i in tqdm(range(np.shape(n)[0])):
    NSensors, S1, sensor_position1 = sensordata(n[i], data3)
    me=[]
```

```
    for j in ext_d:
        extsource_position= extframework(j,0)
        d= csf(sensor_position1, intsource_position, extsource_position, 1)[2]
        disr= csf(grid, intsource_position, extsource_position, 1)[2]
        di=csf(grid, intsource_position, extsource_position, 1)[0]
        de=csf(grid, intsource_position, extsource_position, 1)[1]
        ES= lstsqsolution(NSensors, S1, d)
        A= ES[0:(np.shape(intsource_position)[0])]
        B= ES[(np.shape(intsource_position)[0]):(-1)]
        T0= ES[-1]
        grid= ComparisionGrid(21)
        Tpred=predictedT(grid, intsource_position, extsource_position, A, B, T0, ES)[2]
        mae1= error(Tpred, ndTR)
        me.append(mae1)
    minmae= min(me)
    ext_d_min= ext_d[me.index(min(me))]
    min_mae.append(minmae)
    min_mae_d.append(ext_d_min)


Least_MAE=min(min_mae)
NS_optimum=n[min_mae.index(min(min_mae))]
NSensors_optimum = sensordata(NS,data3)[0]
print(NSensors_optimum)

plt.xlabel('Number of Sensor Measurememts, N')
plt.ylabel("Mean Absolute Error, MAE")
plt.plot(n, min_mae, label='PSF with 1-cut on Domain')
plt.ylim(0.01, 0.10)
#plt.axvline(x=48, ymin=0, ymax=0.10, ls='--', linewidth=0.85, color='k')
#plt.axhline(y=0.01897501, xmin=0, xmax=0.285, ls='--', linewidth=0.9, color='k')
#plt.annotate('Min MAE=0.01897 with 48 Measurements', xy=(48,0.01897),xytext=(50,0.0125))
plt.legend()
plt.show()
```

A.8.5. Absolute Error Distribution on an XY Plane

Absolute Error Plot aids in the identification of local regions with poor predictions.

```
i#Absolute Error Plot at any XY Slice
x = np.linspace(0.0, 1.0, 21, endpoint=True)
y = np.linspace(0.0, 1.0, 21, endpoint=True)
z = 0.75

g = np.linspace(0.0, 1.0, 21, endpoint=True)
I = g.tolist().index(z)

T_R1 = ndTR[441 * I:441 * (I + 1)]
T_11 = predictedT(grid, A, B, T0)[1] #When using GSF
#T_11 = predictedT(grid, intsource_position, extsource_position, A, B, T0, ES)[2] #When using
    PSF
T_1 = T_11[441 * I:441 * (I + 1)]
errorxy = np.abs(T_R1 - T_1)

data_xy = [errorxy[n:n+21] for n in range(0, len(errorxy), 21)]
dataxy = np.concatenate(data_xy).tolist()
dataxy = [item for sublist in dataxy for item in sublist]
```

```
data_xy1 = [dataxy[n:n+21] for n in range(0, len(dataxy), 21)]

print("Mean Temp:", np.round(statistics.mean(dataxy), 2))
print(np.round(min(dataxy), 2))
print(np.round(max(dataxy), 2))

trace = go.Heatmap(z=data_xy1, zsmooth='best', colorbar=dict(title='Error in T'))
data1xy = [trace]

layout = go.Layout(
    xaxis=go.layout.XAxis(title=go.layout.xaxis.Title(text='XAxis', font=dict(size=18))),
    yaxis=go.layout.YAxis(title=go.layout.yaxis.Title(text='YAxis', font=dict(size=18))),
    height=700,
    width=700
)

fig = go.Figure(data1xy, layout=layout)
fig.update_layout(font=dict(size=16))

pio.show(fig)
```

B

Virtual Experimental Setups

B.1. Virtual Reality-3 (VR-3)

A virtual reality environment is created by constructing an extended domain with dimensions of $1.6{\times}1.6{\times}1.6m^3$ containing the domain of interest. The origin of this extended domain is positioned at $(-0.3, -0.3, -0.3)$. The domain of interest is represented by a cube with dimensions $1{\times}1{\times}1m^3$ such that it is centered at $(0, 0, 0)$ as represented in Fig B.1.

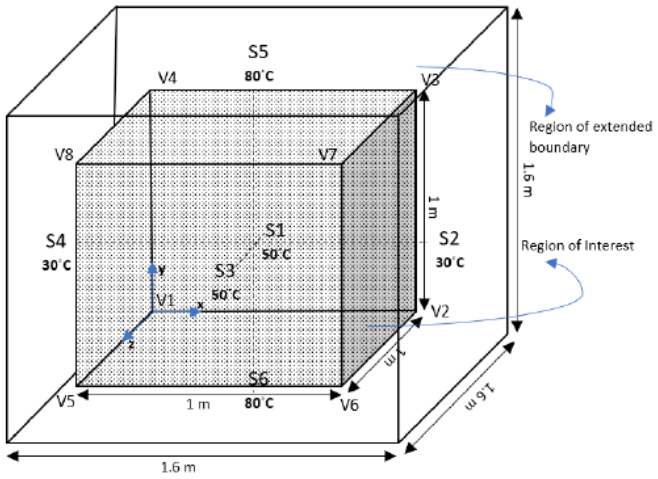


Figure B.1: 3-D representation of Virtual Reality-3 from [11].

The specifications for extended-domain and domain of interest based on arbitrary choices for Virtual Reality-3 are as follows:

- **Shape:** Cube
- **Dimensions: Domain of Interest** $L \times W \times H$: $1 \times 1 \times 1$ m^3
- **Dimensions: Extended Domain** $L \times W \times H$: $1.6 \times 1.6 \times 1.6$ m^3
- **Medium:** Water initially at $0°C(273.15K)$
- **Temperature at Surfaces of Extended Domain:**

- $S1, S3$: Temperature of $50°C (323.15K)$
- $S2, S4$: Temperature of $30°C (303.15K)$
- $S5, S6$: Temperature of $80°C (353.15K)$

- $V1, V2, V3, V4, V5, V6, V7, V8$ are the 8 vertices of the domain of interest, with vertex $V1$ positioned at coordinates $(0, 0, 0)$
- There are no heat sources within the domain.

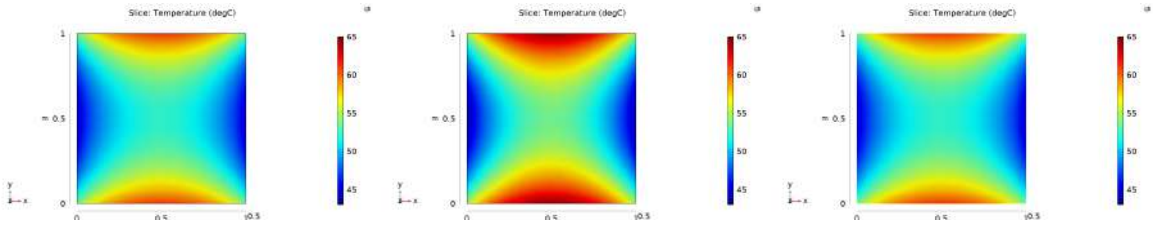
Through the incorporation of an extended domain, the boundary temperatures are shifted away from the region of interest by a distance of $0.3m$. This adjustment aims to achieve a more uniform distribution of temperature, eliminating sudden variations near the edges. The purpose of this Virtual Setup is to create a domain of interest with smoother temperature profiles along its boundaries, as opposed to the previous versions (VR-1 and VR-2) [11].

B.1.1. Physics, Study, and Mesh

Under the *Model Wizard* option, 3D is selected as the *Space Dimension*. Then under the *Heat Transfer* module, *Heat Transfer in Fluids* interface is added as *Physics*. In the next step, *Stationary Study* is selected to solve for steady-state condition explained in subsection 1.3.1.

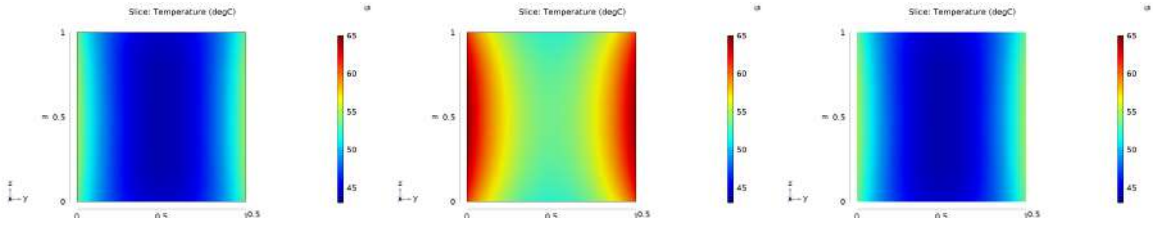
Once the *Physics* and *Study* are chosen, the Geometry and Material for the medium, boundary conditions are incorporated as per the specifications listed above. A Mesh Refinement Study (MRS) was also performed by [11] on this domain to identify the meshing refinement required based on the element size. MRS results suggested that with *Finer* element size the mesh converged, indicating that the temperature values are not affected by further refinement of mesh.

B.1.2. Results: XY, YZ and XZ Temperature Planes



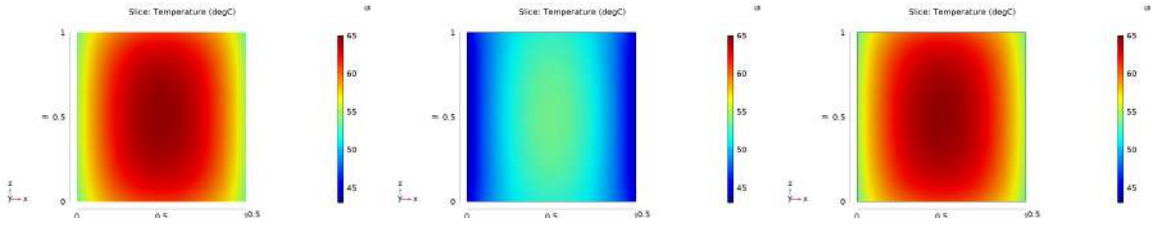
(a) At z=0 m **(b)** At z=0.5m **(c)** At z=1 m

Figure B.2: True Temperature XY Planes of Virtual Reality-3



(a) At x=0 m **(b)** At x=0.5m **(c)** At x=1 m

Figure B.3: True Temperature YZ Planes of Virtual Reality-3



(a) At y=0 m **(b)** At y=0.5m **(c)** At y=1 m

Figure B.4: True Temperature XZ Planes of Virtual Reality-3

B.2. Virtual Experiment 1.1: COMSOL Set-up

The domain specifications remain the same as Virtual Reality-3 (B.1), except that there is a constant heat source ($Q = 500\frac{W}{m^3}$) present within the domain.

B.2.1. Physics and Study

The settings for *Physics* and *Study* remain the same as mentioned for Virtual Reality-3. Additionally, the heat source is added as a user-defined General Source in the *Heat Source* settings available in *Domain* drop-down menu under the *Physics* tab.

B.2.2. Mesh Refinement Study

A Mesh Refinement Study is conducted for this virtual experiment to determine the level of mesh refinement (in terms of element size) needed to ensure that the results are not affected by further refining the size of the mesh. MRS is executed using the *Parametric Sweep* function in COMSOL. The sizes of the mesh elements for different refinement types (e.g., coarse, fine, finer) are defined in Table B.1, and these values are provided as input to COMSOL through a text file using the *Load from File* option in Settings for *Parametric Sweep*.

Mesh	No. of elements	Max. element size	Min. element size	Max. element growth rate	Curvature factor	Resolution of Narrow Regions
Coarser	3257	0.280	0.060	1.7	0.8	0.3
Coarse	4168	0.240	0.042	1.6	0.7	0.4
Normal	16142	0.160	0.032	1.5	0.6	0.5
Fine	38634	0.120	0.014	1.45	0.5	0.6
Finer	103940	0.086	0.0065	1.4	0.4	0.7
Extra fine	870584	0.043	0.0018	1.35	0.3	0.85
Extremely fine	2142064	0.032	0.00042	1.3	0.2	1.0

Table B.1: Different mesh properties for Mesh Refinement Study on Virtual Experiment 1.1

The steady-state or stationary solution obtained from the *Parametric Sweep* study is presented as the Mesh Refinement Study (MRS) results in Table B.2. Temperature values were measured along points on a body diagonal extending to the center of the domain, as depicted in Fig B.5. This measurement approach ensures coverage of both the boundary region and the core region, while considering the symmetric temperature distribution within the domain. The temperature values have been rounded to one decimal place for clarity. The results indicate that the mesh converges with an *Extremely fine* element size.

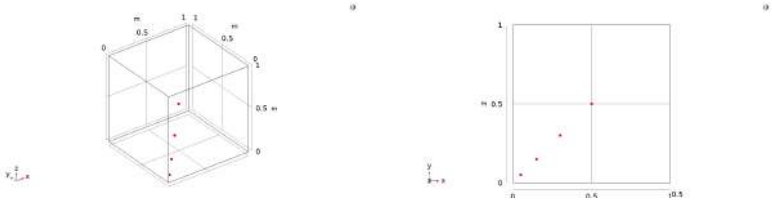
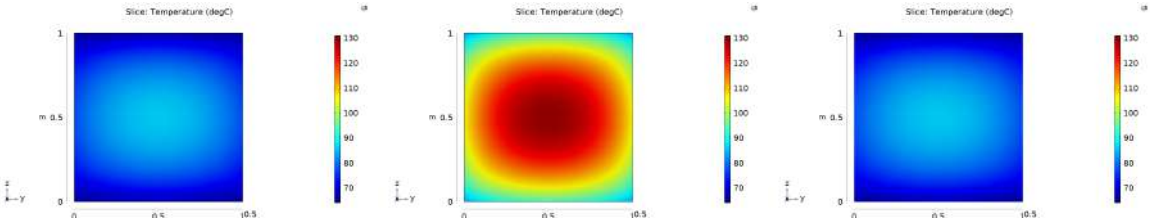


Figure B.5: Temperature measurement points in the domain for Mesh Refinement Study (a) 3D view (b) 2D view

Mesh	Temperature (degC) at (0.05, 0.05,0.05)	Temperature (degC) at (0.15,0.15,0.15)	Temperature (degC) at (0.3,0.3,0.3)	Temperature (degC) at (0.5,0.5,0.5)
Coarser	72.4	89.6	114.0	126.4
Coarse	72.4	88.2	113.1	126.8
Normal	72.6	91.2	115.1	128.4
Fine	72.9	91.8	115.8	129.3
Finer	73.5	92.0	116.1	129.7
Extra fine	73.7	92.2	116.5	130.2
Extremely fine	73.7	92.2	116.5	130.3

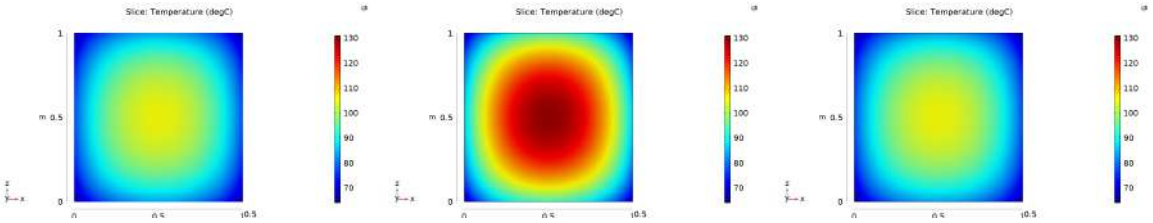
Table B.2: Mesh Refinement Study Results for Virtual Experiment 1.1

B.2.3. Additional Results: YZ and XZ temperature planes



(a) At x=0 m **(b)** At x=0.5m **(c)** At x=1 m

Figure B.6: True Temperature YZ Planes of Virtual Experiment 1.1



(a) At y=0 m **(b)** At y=0.5m **(c)** At y=1 m

Figure B.7: True Temperature XZ Planes of Virtual Experiment 1.1

B.3. Virtual Experiment 1.2: COMSOL Set-up

The domain specifications continue to remain the same as Virtual Reality-3 (B.1), additionally there is a Gaussian heat source ($\dot{q} = C \cdot exp[-\frac{(x-0.5)^2+(y-0.5)^2+(z-0.5)^2}{2}]$) present within the domain.

B.3.1. Physics and Study

The settings for *Physics* and *Study* remain the same as mentioned previously. To add a Gaussian heat source, first, a Gaussian function is added as a *Variable* (Q) under the option *Definitions*. The heat

source is added with the same variable name in the user-defined General Source option in the *Heat Source* settings available in *Domain* drop-down menu under the *Physics* tab.

B.3.2. Mesh Refinement Study

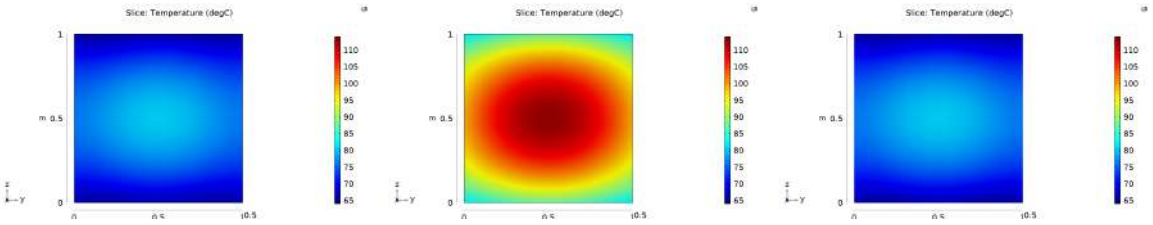
The sizes of the mesh elements for different refinement types (e.g., coarse, fine, finer) remain the same as defined in Table B.1, and these values are provided as input to COMSOL through a text file using the *Load from File* option in Settings for *Parametric Sweep*.

Temperature values are measured as the same points as illustrated in B.5. MRS Results are presented in Table B.3. The results suggest that the mesh converges with an *Extremely fine* element size.

Mesh	Temperature (degC) at (0.05, 0.05,0.05)	Temperature (degC) at (0.15,0.15,0.15)	Temperature (degC) at (0.30,0.30,0.30)	Temperature (degC) at (0.5,0.5,0.5)
Coarser	71.0	83.0	100.6	111.80
Coarse	70.9	81.7	100.0	111.57
Normal	71.1	83.3	101.3	112.6
Fine	71.1	83.6	101.6	112.9
Finer	71.4	83.7	101.8	113.1
Extra fine	71.4	83.8	102.0	113.3
Extremely fine	71.4	83.8	102.0	113.4

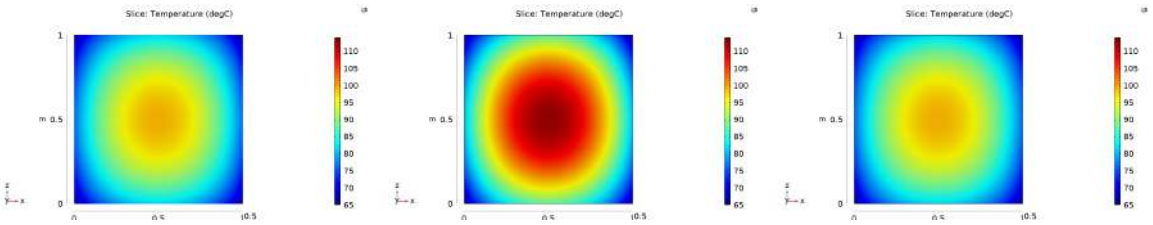
Table B.3: Mesh Refinement Study Results for Virtual Experiment 1.2

B.3.3. Additional Results: YZ and XZ temperature planes



(a) At x=0 m **(b)** At x=0.5m **(c)** At x=1 m

Figure B.8: True Temperature YZ Planes of Virtual Experiment 1.2



(a) At y=0 m **(b)** At y=0.5m **(c)** At y=1 m

Figure B.9: True Temperature XZ Planes of Virtual Experiment 1.2

B.4. Virtual Experiment 2: COMSOL Set-up

The domain specifications remain the same as Virtual Reality-3 (B.1) generating smooth temperature variations in most of the domain. In this experiment, there are two Gaussian heat sources ($\dot{q}_1 = 125 \cdot exp[-\frac{(x-0.5)^2+(y-0.25)^2+(z-0.75)^2}{0.0034}]\frac{W}{m^3}$ and $\dot{q}_2 = 25 \cdot exp[-\frac{(x-0)^2+(y-0)^2+(z-0)^2}{0.0208}]\frac{W}{m^3}$) present within the domain that result in sudden spikes in an otherwise smooth temperature distribution.

B.4.1. Physics and Study

The settings for *Physics* and *Study* remain the same as mentioned previously. Gaussian heat sources are added as *Variable* with a unique variable (Q_1 and Q_2) name under the option *Definitions*. The heat sources are then inputted with their unique variable name in the user-defined General Source option in the *Heat Source* settings available in *Domain* drop-down menu under the *Physics* tab.

B.4.2. Mesh Refinement Study

The sizes of the mesh elements for different refinement types (e.g., coarse, fine, finer) continue to remain the same as defined in Table B.1, and these values are provided as input to COMSOL through a text file using the *Load from File* option in Settings for *Parametric Sweep*.

Temperature values are measured as the points near the boundaries and heat source locations as showcased in Fig B.10. MRS Results are presented in Table B.4. The results suggest that the mesh converges everywhere with an *Extremely fine* element size, except near the heat source locations. Further refinement will increase computational expense, and so the mesh refinement is not pursued beyond this stage for achieving grid independence. Hence, *Extremely fine* element size is chosen for this experiment.

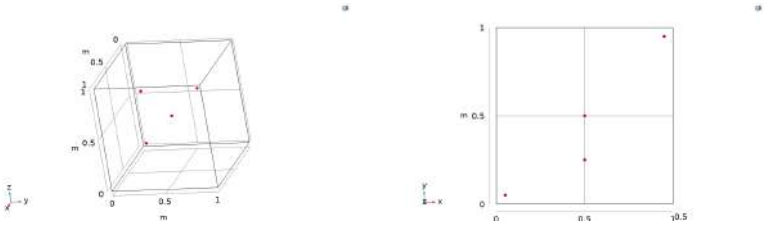
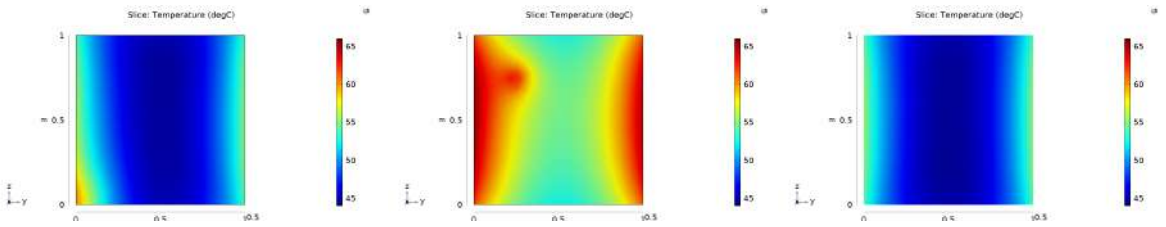


Figure B.10: Temperature measurement points in the domain for Mesh Refinement Study of Virtual Experiment 2 (a) 3D view (b) 2D view

Mesh	Temperature (degC) at (0.05,0.05,0.05)	Temperature (degC) at (0.5,0.25,0.75)	Temperature (degC) at (0.5,0.5,0.5)	Temperature (degC) at (0.95,0.95,0.95)
Coarser	58.3	59.8	54.9	54.1
Coarse	58.2	59.2	54.8	54.1
Normal	58.8	60.0	54.8	53.9
Fine	59.0	61.3	54.7	53.9
Finer	59.3	61.7	54.7	54.0
Extra fine	59.4	62.5	54.7	54.0
Extremely fine	59.5	62.8	54.7	54.0

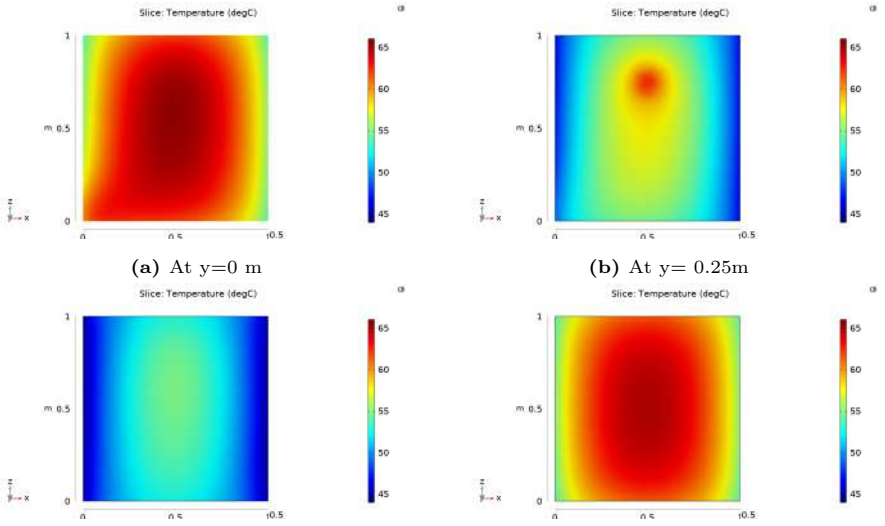
Table B.4: MRS Results for Virtual Experiment 2

B.4.3. Additional Results: YZ and XZ temperature planes



(a) At x=0 m **(b)** At x= 0.5m **(c)** At x=1 m

Figure B.11: YZ Temperature Plots of Virtual Experiment 2.



(a) At y=0 m **(b)** At y= 0.25m

(c) At y=0.5 m **(d)** At y=1 m

Figure B.12: XZ Temperature Plots of Virtual Experiment 2.

B.5. Virtual Experiment 3.1: COMSOL Set-up

In this experiment, there is laminar fluid flow involved and there is no separate heat generation. The domain specifications remain the same as Virtual Reality-3 (B.1), except the medium of fluid. In this experiment,

- **Medium:** Air at $20°C$
- **Velocity:** 0.0012 $\frac{m}{s}$

The geometry is modified from VR-3 to include fluid inlet and outlet as shown in Fig B.13. The setup is kept simple as the concepts are still exploratory. The Inlet and Outlets are built as *Cylinders* in COMSOL. Their dimension and location specifications are as follows:

- **Inlet:** Radius: 0.05m and Height: 0.4m (along the z-axis) positioned at coordinates (0.5, 0.5, -0.35).
- **Outlet 1:** Radius: 0.05m and Height: 0.4m (along the z-axis) positioned at coordinates (0.25, 0.5, 0.95).
- **Outlet 2:** Radius: 0.05m and Height: 0.4m (along the z-axis) positioned at coordinates (0.75, 0.5, 0.95).

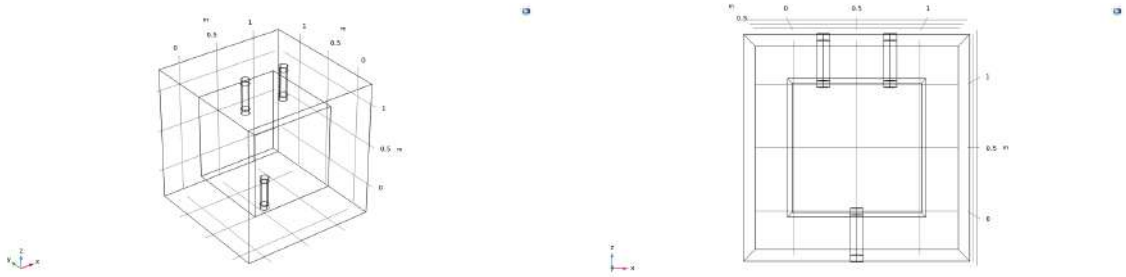


Figure B.13: The (extended-) domain geometry in Virtual Experiments 3.1 and 3.2 to include fluid flow.

B.5.1. Physics and Study

The settings for *Study* remain the same as mentioned previously. *Laminar Flow* interface is added along with *Heat Transfer in Fluids* interface in *Physics*. The walls along with the interior walls should be supplied with *No-slip* boundary condition. Velocity is inputted in *Inlet* settings. Pressure is given as a boundary condition in *Outlet* settings.

B.5.2. Mesh Refinement Study

The sizes of the mesh elements for different refinement types (e.g., coarse, fine, finer) are defined in Table B.5, and these values are provided as input to COMSOL through a text file under the *Global Parameters* section.

Mesh	Max. element size	Min. element size	Max. element growth rate	Curvature factor	Resolution of Narrow Regions
Coarser	0.224	0.048	1.70	0.80	0.30
Coarse	0.164	0.030	1.60	0.70	0.40
Normal	0.124	0.022	1.50	0.60	0.50
Fine	0.10	0.012	1.45	0.50	0.60
Finer	0.082	0.0054	1.40	0.40	0.70
Extra fine	0.042	0.0018	1.35	0.30	0.85
Extremely fine	0.020	0.00022	1.30	0.20	1.00

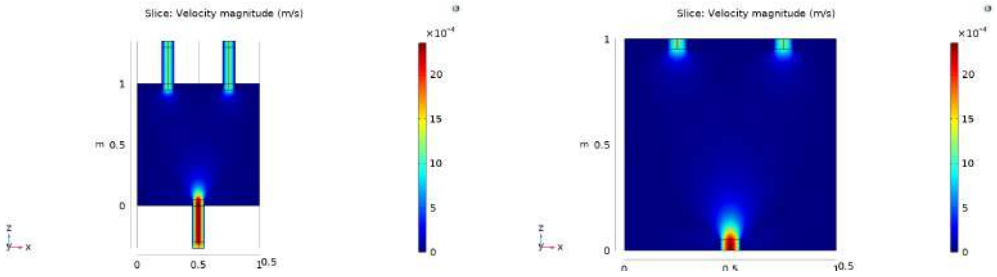
Table B.5: Different mesh input values for MRS for Virtual Experiment 3.1 and 3.2

MRS Results are presented in Table B.6. The results suggest that the mesh converges with a *Finer* element size.

Mesh	Temperature (degC) at (0.5,0.5,0.1)	Temperature (degC) at (0.25,0.5,0.9)	Temperature (degC) at (0.75,0.5,0.9)	Temperature (degC) at (0.5,0.5,0.5)	Temperature (degC) at (0.5,0.1,0.5)	Temperature (degC) at (0.5,0.9,0.5)
Coarser	51.1	51.1	51.1	53.5	60.6	60.5
Coarse	51.1	51.1	51.1	53.6	60.7	60.7
Normal	51.1	51.1	51.1	53.7	60.8	60.8
Fine	51.1	51.1	51.1	53.6	60.8	60.8
Finer	51.0	51.1	51.1	53.6	60.8	60.8
Extra fine	51.0	51.1	51.1	53.6	60.8	60.8
Extremely fine	51.0	51.1	51.1	53.6	60.8	60.8

Table B.6: Mesh Refinement Study Results for Virtual Experiment 3.1

B.5.3. Additional Results 1: XZ velocity planes



(a) **(b)**

Figure B.14: XZ Velocity Plot of Virtual Experiment 3.1 at y=0.5m (a) extended domain (b) cropped to the main domain of interest

B.5.4. Additional Results 2: XZ $\dot{q}_{conv}$ plane

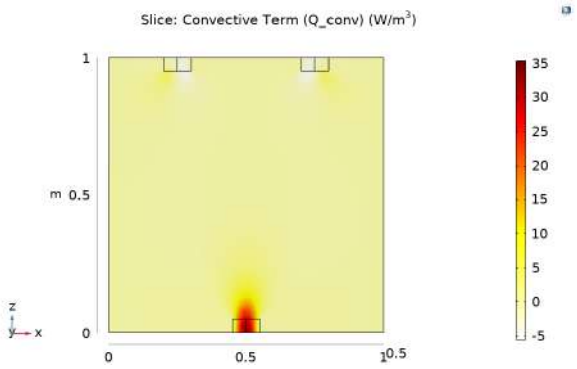
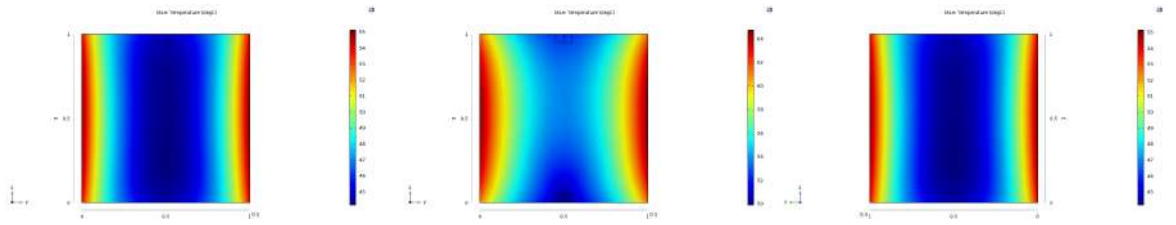


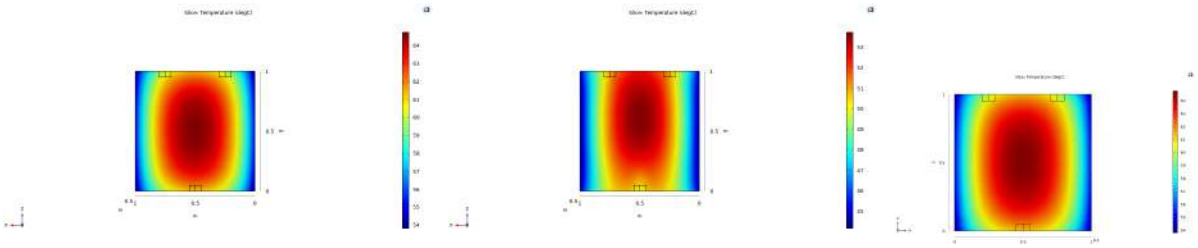
Figure B.15: XZ Plot of Convective Heat Source Term ($Q_{conv} = \rho_{air} C_{air}(\vec{v} \cdot \nabla T)$) at y=0.5m for Virtual Experiment 3.1

B.5.5. Additional Results 3: YZ and XZ temperature planes



(a) At x=0 m **(b)** At x=0.5m **(c)** At x=1 m

Figure B.16: YZ Temperature Plots of Virtual Experiment 3.1.



(a) At y=0 m **(b)** At y=0.5m **(c)** At y=1 m

Figure B.17: XZ Temperature Plots of Virtual Experiment 3.1.

B.6. Virtual Experiment 3.2: COMSOL Set-up

The domain geometry and specifications are the same as in Virtual Experiment 3.1, except for the medium of fluid. In this experiment, the **Medium** is Water at $20°C$.

B.6.1. Physics and Study

The settings for *Study*, *Physics* interfaces along with the boundary conditions remain the same as in Virtual Experiment 3.1.

B.6.2. Mesh Refinement Study

The sizes of the mesh elements for different refinement types (e.g., coarse, fine, finer) are defined in Table B.5, and these values are provided as input to COMSOL through a text file under the *Global Parameters* section.

MRS Results are presented in Table B.7. The results suggest that the mesh converges with an *Extremely fine* element size. Further refinement is not pursued to avoid computational expenses.

Mesh	Temperature (degC) at (0.5,0.5,0.1)	Temperature (degC) at (0.25,0.5,0.9)	Temperature (degC) at (0.75,0.5,0.9)	Temperature (degC) at (0.5,0.5,0.5)	Temperature (degC) at (0.5,0.1,0.5)	Temperature (degC) at (0.5,0.9,0.5)
Coarser	28.5	33.6	33.6	33.4	36.0	35.9
Coarse	28.3	33.5	33.6	32.8	37.0	36.3
Normal	28.0	33.6	33.6	32.8	37.8	37.6
Fine	28.0	33.7	33.7	31.8	40.6	39.4
Finer	27.8	33.7	33.7	31.4	40.7	41.3
Extra fine	27.5	33.6	33.6	29.6	46.3	46.3
Extremely fine	27.2	33.6	33.7	27.8	47.0	47.0

Table B.7: Mesh Refinement Study Results for Virtual Experiment 3.2

B.6.3. Additional Results 1: XZ velocity planes

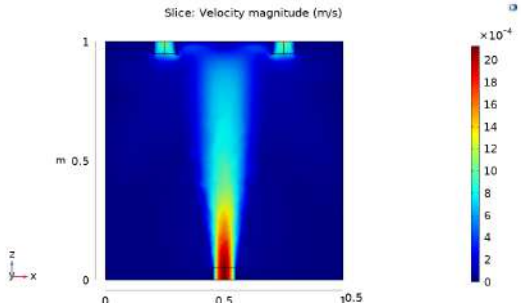


Figure B.18: XZ Plane Velocity Plot at y=0.5m for Virtual Experiment 3.2

B.6.4. Additional Results 2: XZ $\dot{q}_{conv}$ plane

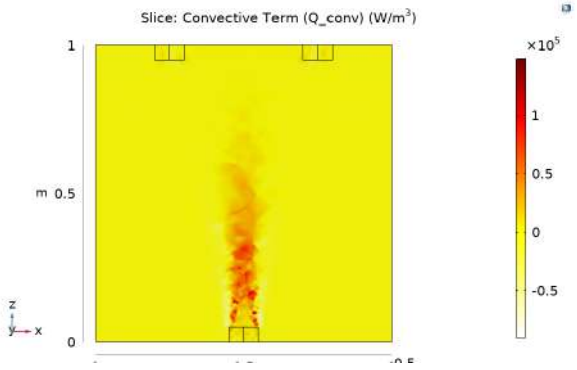
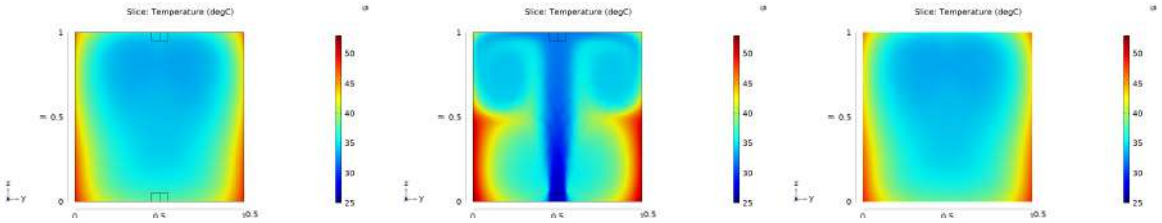


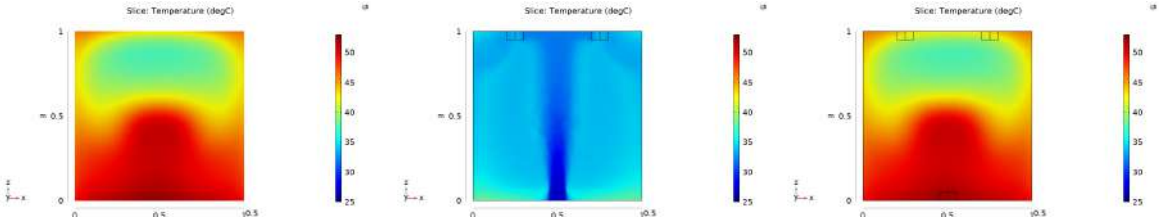
Figure B.19: XZ Plot of Convective Heat Source Term ($Q_{conv} = \rho_{water} C_{water}(\vec{v} \cdot \nabla T)$) at y=0.5m for Virtual Experiment 3.1

B.6.5. Additional Results 3: YZ and XZ temperature planes



(a) At x=0 m **(b)** At x=0.5m **(c)** At x=1 m

Figure B.20: YZ Temperature Plots of Virtual Experiment 3.2



(a) At y=0 m **(b)** At y=0.5m **(c)** At y=1 m

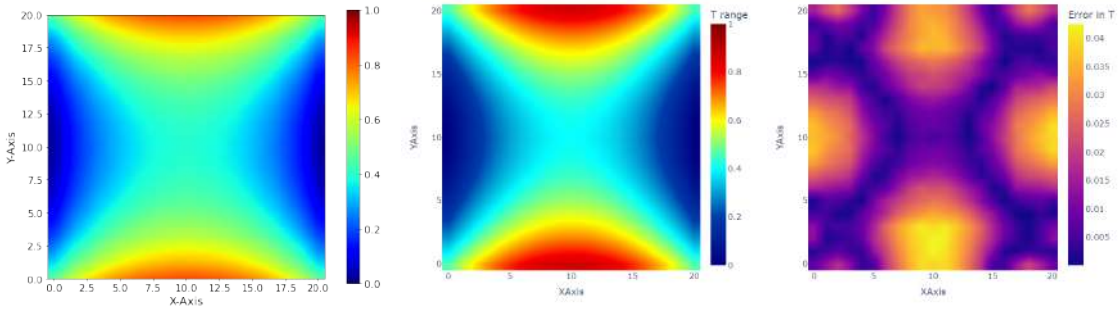
Figure B.21: XZ Temperature Plots of Virtual Experiment 3.2

C

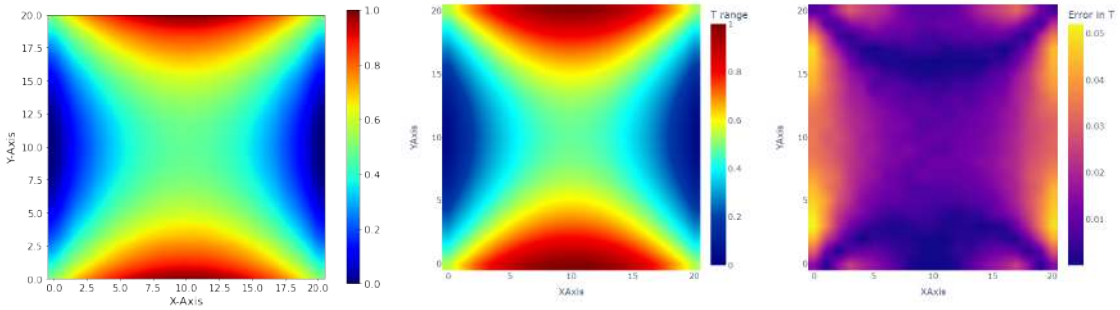
Supplementary Results

C.1. Virtual Reality-3 (VR-3)

VR-3 Results in [11] were not presented in non-dimensional form. Hence, for an accurate comparison, here we have included non-dimensionalized results of VR-3. The results were achieved with the same cut-configuration (0-cut on the source planes located at d=2m) as mentioned in [11].



(a) True Plot at z=0 m **(b)** Predicted Temperature at z=0m **(c)** Absolute Error Plot at z=0m



(d) True Plot at z=0.5 m **(e)** Predicted Temperature at z=0.5m **(f)** Absolute Error Plot at z=0.5m

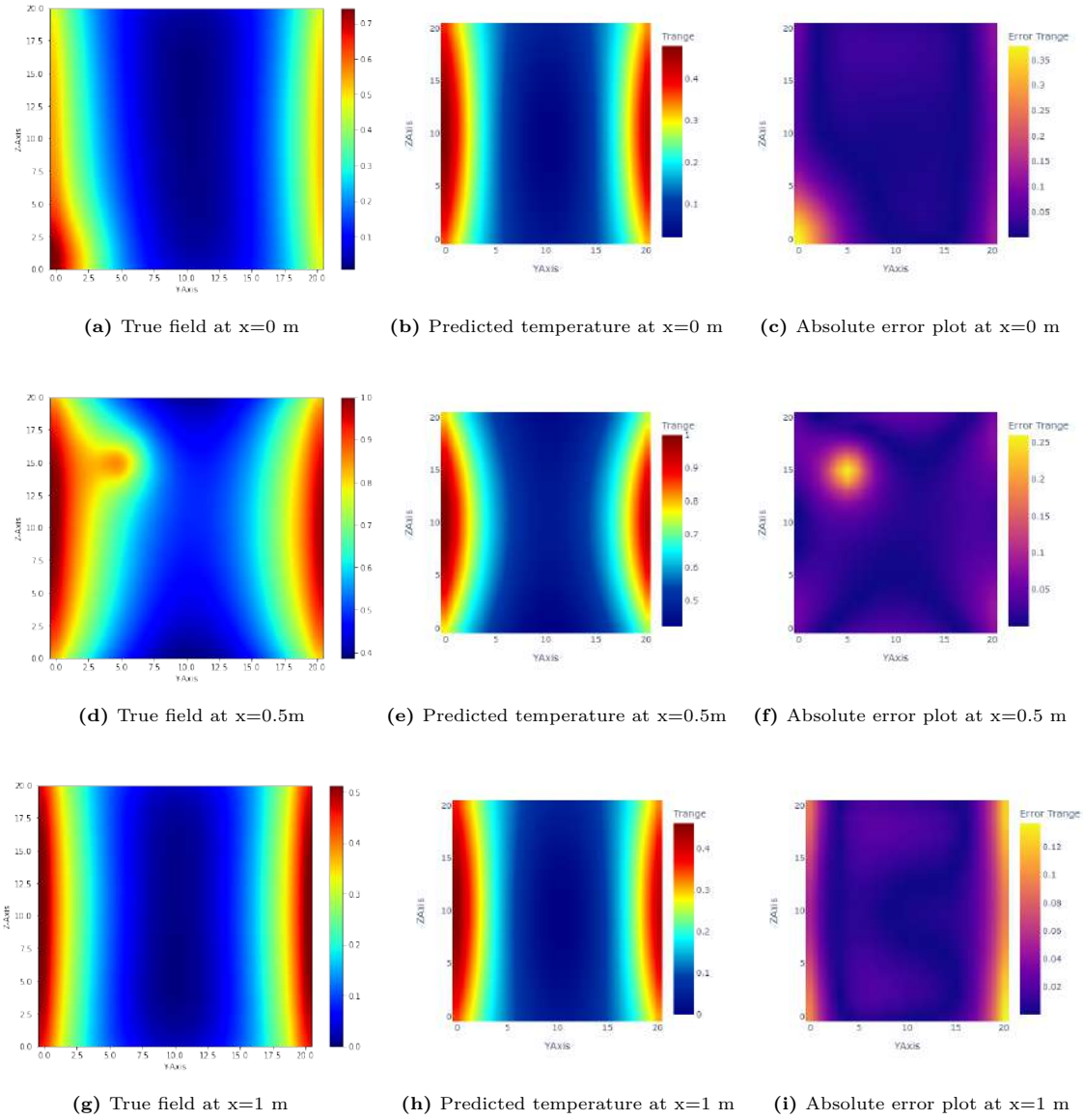
Figure C.1: True and Predicted Normalized Temperature XY Planes along with their Absolute Error plots for Virtual Reality 3 (VR3)

The minimum MAE obtained was 0.015. The absolute error was noted to be higher towards the

boundaries of the domain, in the range of [0.03-0.05].

C.2. Virtual Experiment 2 (VE 2)

C.2.1. Predictions: Generic Framework

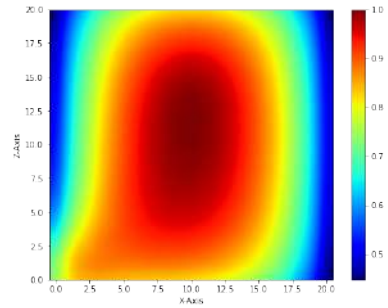


(a) True field at x=0 m **(b)** Predicted temperature at x=0 m **(c)** Absolute error plot at x=0 m

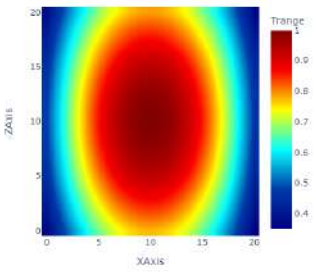
(d) True field at x=0.5m **(e)** Predicted temperature at x=0.5m **(f)** Absolute error plot at x=0.5 m

(g) True field at x=1 m **(h)** Predicted temperature at x=1 m **(i)** Absolute error plot at x=1 m

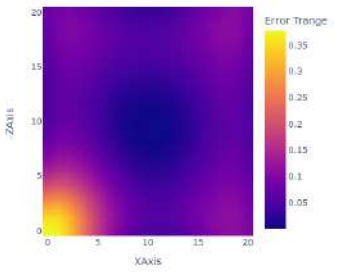
Figure C.2: Generic Framework assisted predictions: YZ Temperature Plots of Virtual Experiment 2 and respective error plots.



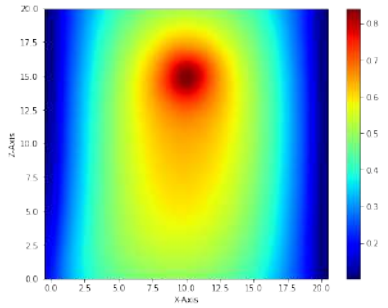
(a) True field at y=0 m



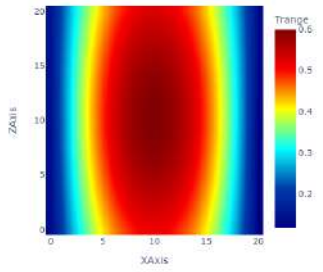
(b) Predicted temperature at y=0 m



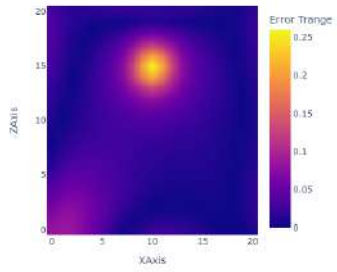
(c) Absolute error plot at y=0 m



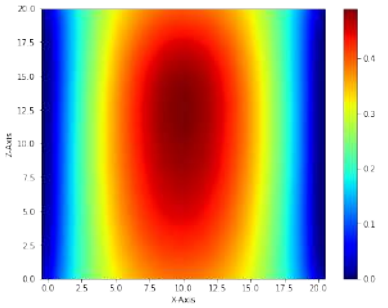
(d) True field at y=0.25m



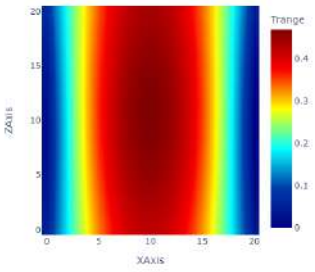
(e) Predicted temperature at y=0.25m



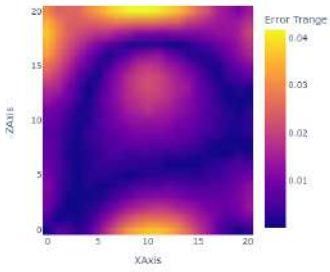
(f) Absolute error plot at y=0.25 m



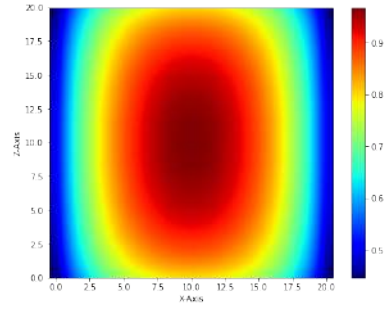
(g) True field at y=0.5 m



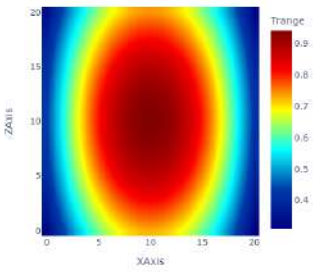
(h) Predicted temperature at y=0.5 m



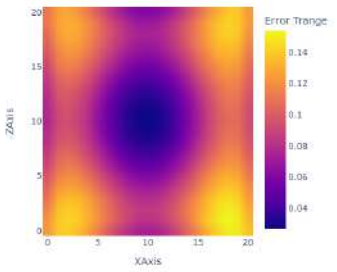
(i) Absolute error plot at y=0.5 m



(j) True field at y=1 m



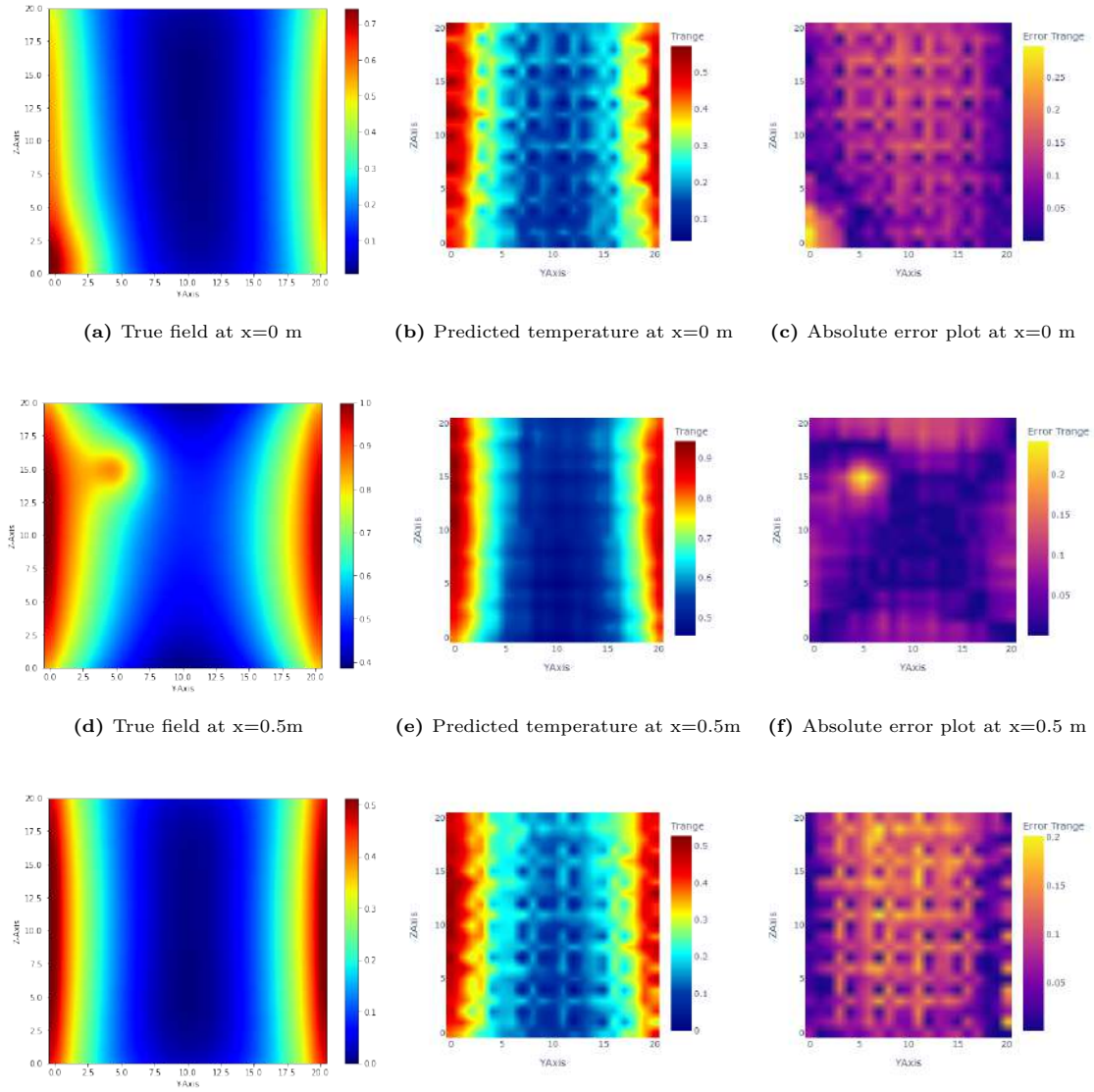
(k) Predicted temperature at y=1 m



(l) Absolute error plot at y=1 m

Figure C.3: Generic Framework assisted predictions: XZ Temperature Plots of Virtual Experiment 2 and respective error plots.

C.2.2. Predictions: Partition Framework (Exact Solution)



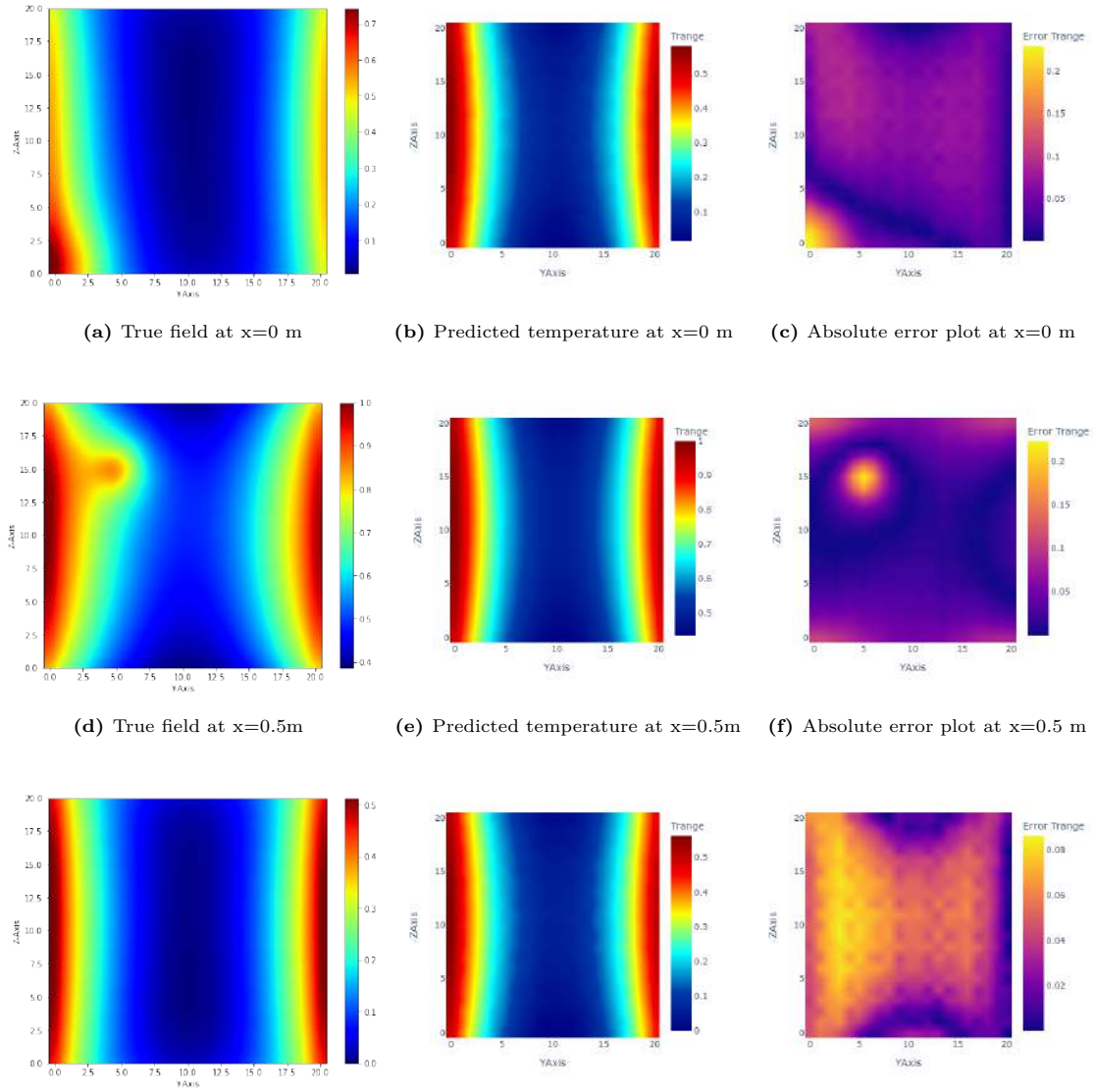
(a) True field at x=0 m **(b)** Predicted temperature at x=0 m **(c)** Absolute error plot at x=0 m

(d) True field at x=0.5m **(e)** Predicted temperature at x=0.5m **(f)** Absolute error plot at x=0.5 m

(g) True field at x=1 m **(h)** Predicted temperature at x=1 m **(i)** Absolute error plot at x=1 m

Figure C.4: Partition Framework-Exact Solution assisted predictions: YZ Temperature Plots of Virtual Experiment 2 and respective error plots.

C.2.3. Predictions: Partition Framework (Least Squares Solution)



(a) True field at x=0 m **(b)** Predicted temperature at x=0 m **(c)** Absolute error plot at x=0 m

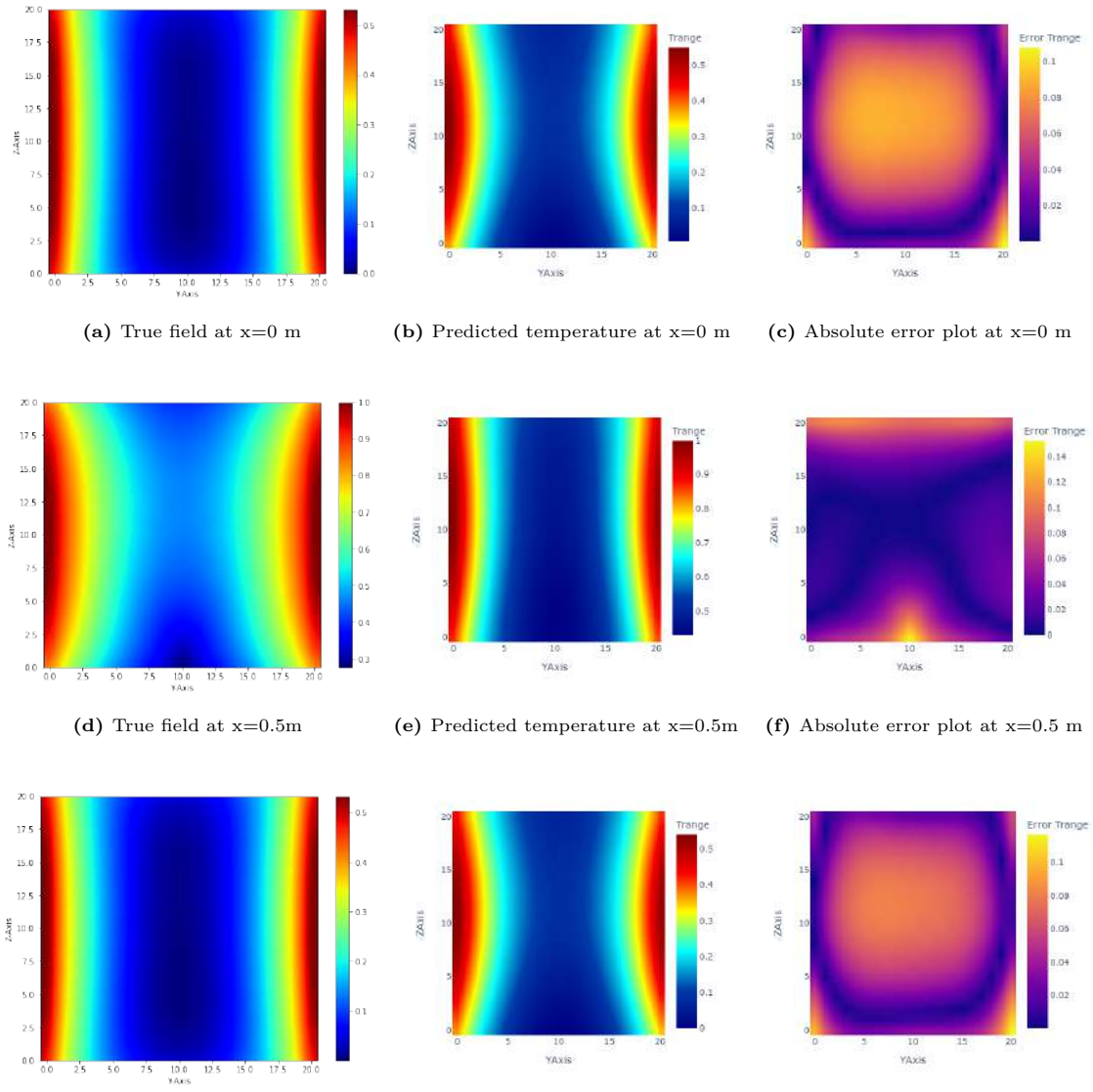
(d) True field at x=0.5m **(e)** Predicted temperature at x=0.5m **(f)** Absolute error plot at x=0.5 m

(g) True field at x=1 m **(h)** Predicted temperature at x=1 m **(i)** Absolute error plot at x=1 m

Figure C.5: Partition Framework-Least Squares Solution assisted predictions: YZ Temperature Plots of Virtual Experiment 2 and respective temperature error plots.

C.3. Virtual Experiment 3.1

C.3.1. Predictions: Generic Framework

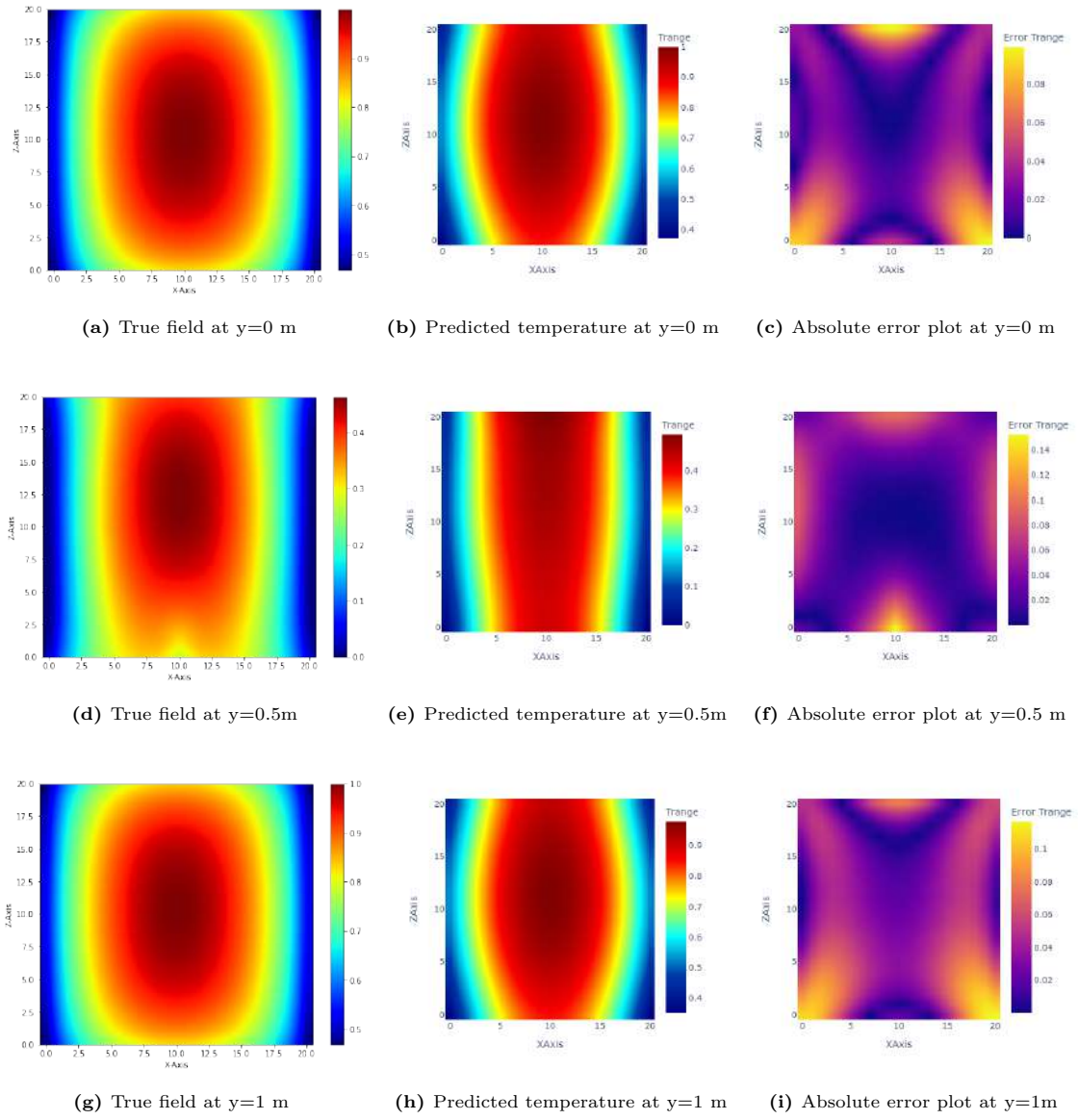


(a) True field at x=0 m **(b)** Predicted temperature at x=0 m **(c)** Absolute error plot at x=0 m

(d) True field at x=0.5m **(e)** Predicted temperature at x=0.5m **(f)** Absolute error plot at x=0.5 m

(g) True field at x=1 m **(h)** Predicted temperature at x=1 m **(i)** Absolute error plot at x=1m

Figure C.6: Generic Framework assisted predictions: YZ Temperature Plots of Virtual Experiment 3.1 and respective temperature error plots.



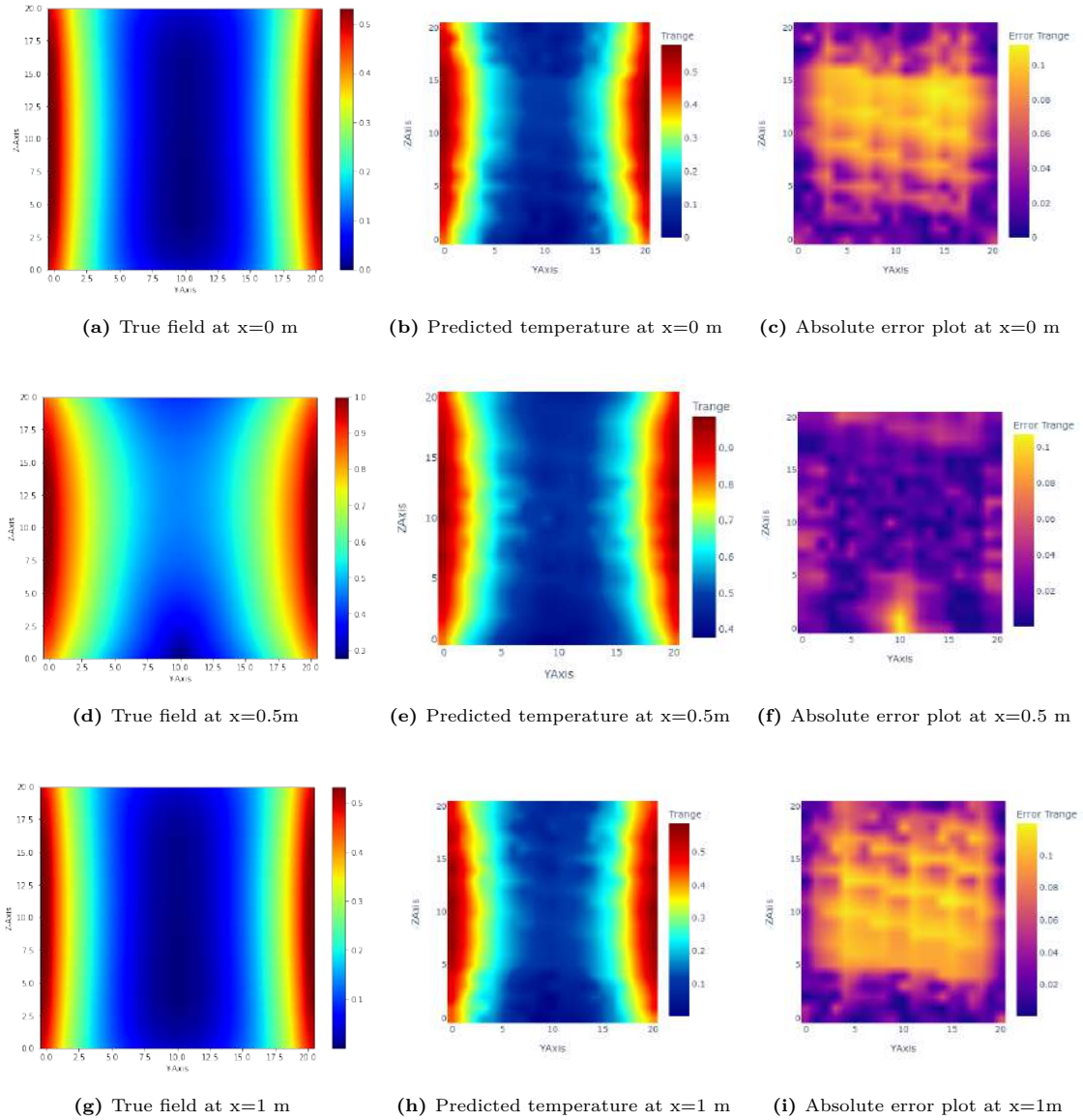
(a) True field at y=0 m **(b)** Predicted temperature at y=0 m **(c)** Absolute error plot at y=0 m

(d) True field at y=0.5m **(e)** Predicted temperature at y=0.5m **(f)** Absolute error plot at y=0.5 m

(g) True field at y=1 m **(h)** Predicted temperature at y=1 m **(i)** Absolute error plot at y=1m

Figure C.7: Generic Framework assisted predictions: XZ Temperature Plots of Virtual Experiment 3.1 and respective temperature error plots.

C.3.2. Predictions: Partition Framework (Exact Solution)



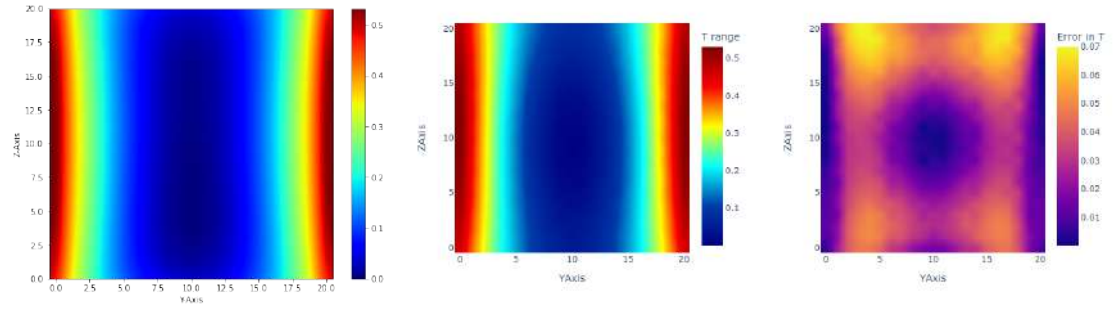
(a) True field at x=0 m **(b)** Predicted temperature at x=0 m **(c)** Absolute error plot at x=0 m

(d) True field at x=0.5m **(e)** Predicted temperature at x=0.5m **(f)** Absolute error plot at x=0.5 m

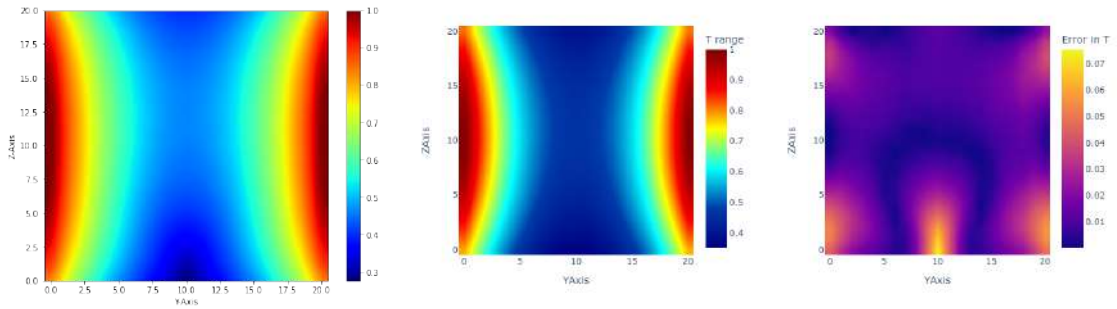
(g) True field at x=1 m **(h)** Predicted temperature at x=1 m **(i)** Absolute error plot at x=1m

Figure C.8: Partition Framework-Exact Solution assisted predictions: YZ Temperature Plots of Virtual Experiment 3.1 and respective temperature error plots.

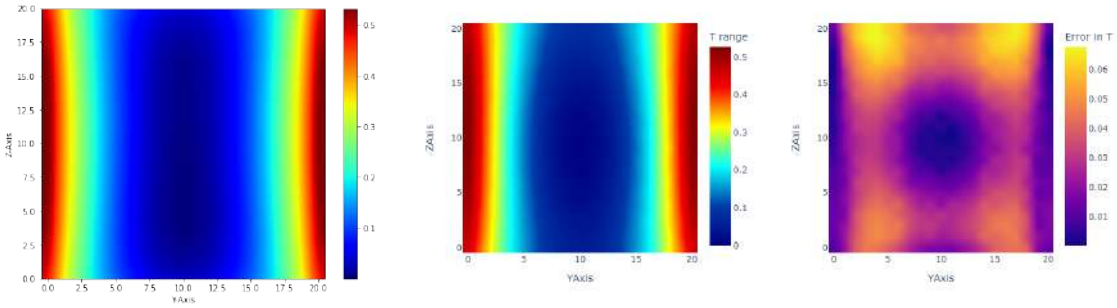
C.3.3. Predictions: PSF (Least Squares Solution)



(a) True field at x=0 m **(b)** Predicted temperature at x=0 m **(c)** Absolute error plot at x=0 m



(d) True field at x=0.5m **(e)** Predicted temperature at x=0.5m **(f)** Absolute error plot at x=0.5 m

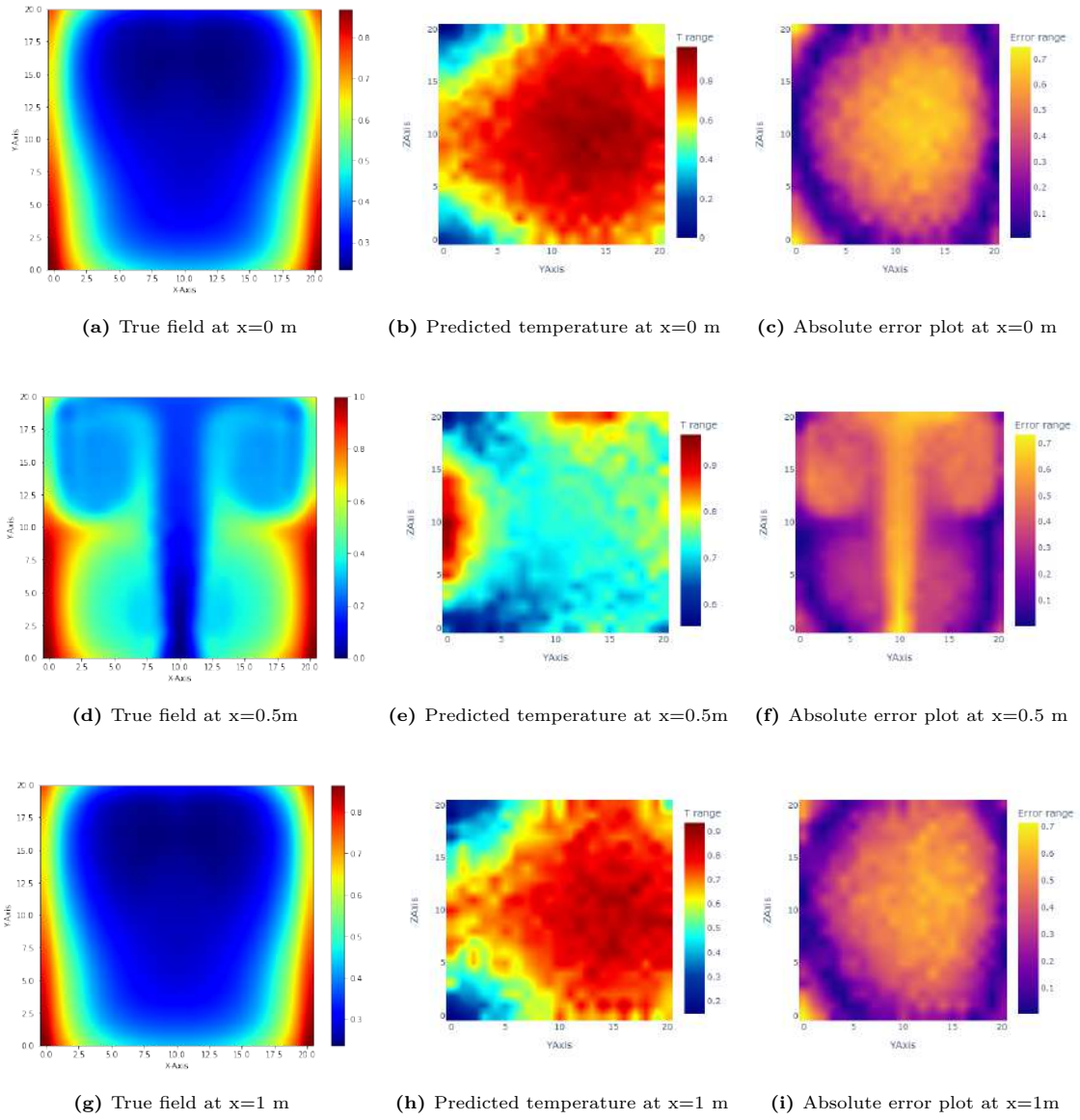


(g) True field at x=1 m **(h)** Predicted temperature at x=1 m **(i)** Absolute error plot at x=1m

Figure C.9: Partition Framework-Least Squares Solution assisted predictions: YZ Temperature Plots of Virtual Experiment 3.1 and respective temperature error plots.

C.4. Virtual Experiment 3.2

C.4.1. Predictions: Generic Framework

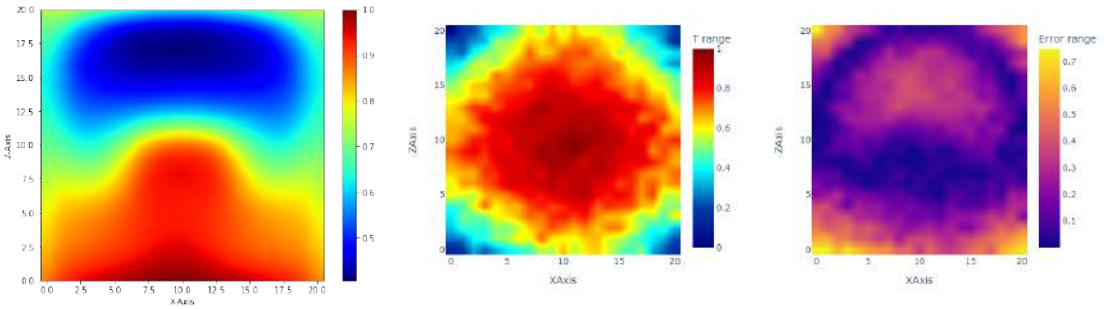


(a) True field at x=0 m **(b)** Predicted temperature at x=0 m **(c)** Absolute error plot at x=0 m

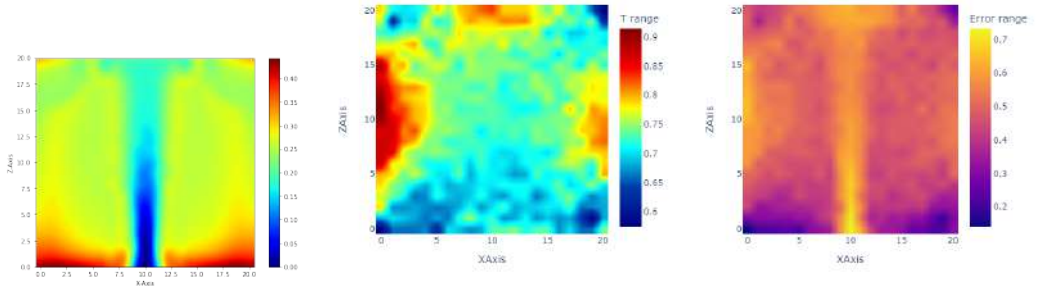
(d) True field at x=0.5m **(e)** Predicted temperature at x=0.5m **(f)** Absolute error plot at x=0.5 m

(g) True field at x=1 m **(h)** Predicted temperature at x=1 m **(i)** Absolute error plot at x=1m

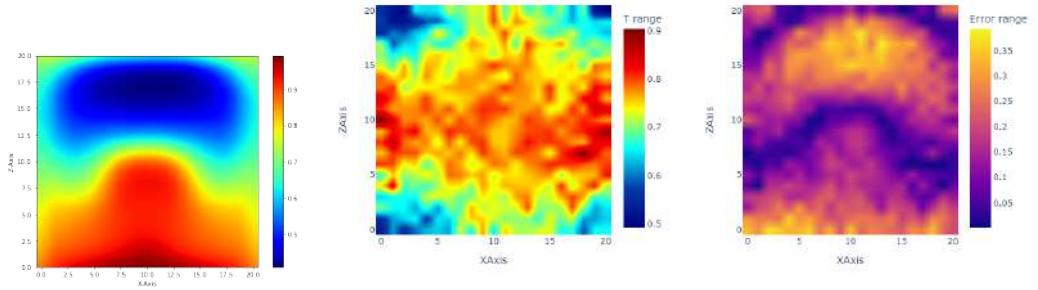
Figure C.10: Generic Framework assisted predictions: YZ Temperature Plots of Virtual Experiment 3.2 and respective temperature error plots.



(a) True field at y=0 m **(b)** Predicted temperature at y=0 m **(c)** Absolute error plot at y=0 m



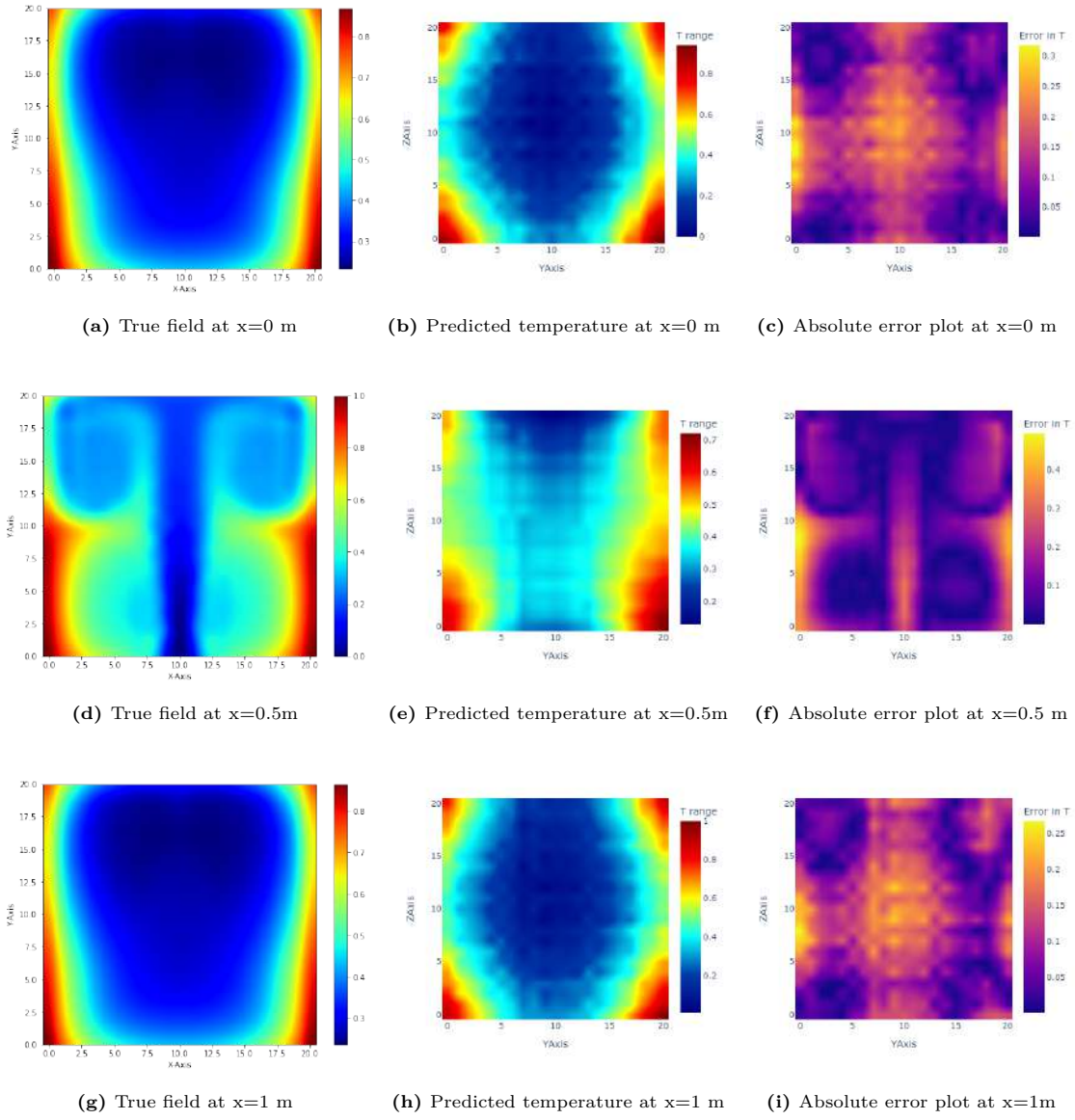
(d) True field at y=0.5m **(e)** Predicted temperature at y=0.5m **(f)** Absolute error plot at y=0.5 m



(g) True field at y=1 m **(h)** Predicted temperature at y=1 m **(i)** Absolute error plot at y=1m

Figure C.11: Generic Framework assisted predictions: XZ Temperature Plots of Virtual Experiment 3.2 and respective temperature error plots.

C.4.2. Predictions: Partition Framework (Least Squares Solution)



(a) True field at x=0 m **(b)** Predicted temperature at x=0 m **(c)** Absolute error plot at x=0 m

(d) True field at x=0.5m **(e)** Predicted temperature at x=0.5m **(f)** Absolute error plot at x=0.5 m

(g) True field at x=1 m **(h)** Predicted temperature at x=1 m **(i)** Absolute error plot at x=1m

Figure C.12: Partition Framework assisted predictions: YZ Temperature Plots of Virtual Experiment 3.2 and respective temperature error plots.

C.5. Generic Framework Results for Hypothetical Fluids

To investigate the influence of convective heat source intensity ($\dot{q}_{conv}$) on GSF predictions, various hypothetical fluids were generated with different Péclet Numbers while maintaining a constant velocity across all cases.

Fluid	**Péclet Number (Pe)**	**MAE**
Air	57	0.027
Hypothetical Fluid 1	665	0.070
Hypothetical Fluid 2	1330	0.080
Hypothetical Fluid 3	1560	0.081
Hypothetical Fluid 4	2400	0.091
Water	8200	0.116

Table C.1: Generic Framework: varying MAE with changing Péclet Number in the domain of interest.