Experimental Frequency Response Methods for the Demonstration of Thermal Hydraulic Similitude Between Molten Fluoride Salt and Surrogate Fluid Scaled Experiments

By

Omar Ashraf Alzaabi

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering – Nuclear Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Per F. Peterson, Chair
Professor Van P. Carey
Assistant Professor Raluca O. Scarlat

Spring 2023

Experimental Frequency Response Methods for the Demonstration of Thermal Hydraulic Similitude Between Molten Fluoride Salt and Surrogate Fluid Scaled Experiments

Abstract

Experimental Frequency Response Methods for the Demonstration of Thermal Hydraulic Similitude Between Molten Fluoride Salt and Surrogate Fluid Scaled Experiments

by

Omar Ashraf Alzaabi

Doctor of Philosophy in Engineering – Nuclear Engineering

University of California, Berkeley

Professor Per F. Peterson, Chair

This dissertation presents the design, development, and experimental results of the Scaled Heat Exchange Frequency Response Analysis (SHEFRA) experiment, which aimed to investigate the use of frequency response methods for measuring quasi-steady Nusselt number values in forced convection of a surrogate fluid for molten fluoride salt in circular ducts. The dynamic response in the flow channel was initiated by a periodically-varying fluid temperature at the inlet. Initial experimental results obtained from the SHEFRA experiment suggest that quasi-steady conditions are achievable for laminar flow conditions, showing good agreement with steady-state analytical predictions. The theoretical modeling and frequency scaling analysis provided insights into the relevant dimensionless parameters and their impact on the system's behavior, allowing for optimal experimental parameters and conditions to be determined. Quasi-steady state heat transfer conditions are best approximated at high or low values of a dimensionless parameter, b^*, defined as the dimensionless frequency multiplied by the ratio of wall thermal capacitance and fluid thermal conductivity. At the limit of high frequency, the quasi-steady regime approximates the analytically-predicted steady state heat transfer with an isothermal wall temperature boundary condition. Meanwhile, at the limit of low frequency, the quasi-steady state conditions approximate steady state predictions for a constant flux boundary condition. In comparison, high frequency tests resulted in a better approximation of quasi-steady state near the inlet of the test section. The most deviation in the instantaneous Nusselt number exists in the middle frequency range. The results suggest that additional experimental data, covering a range of Prandtl and Reynolds numbers of interest for molten fluoride salt reactor operation, can be used to create Nusselt number correlations that can be compared with prototypical molten salt experiment data and hence qualify the use of surrogate fluids in scaled experiments. Additionally, frequency response parameter estimation techniques presented in this dissertation offer promising avenues for future research in improving the accuracy of Nusselt number measurements with the potential of estimating other system parameters such as thermophysical properties. The

SHEFRA experiment demonstrates the potential of using surrogate fluids and frequency response methods to obtain high-fidelity heat transfer data measurements relevant for molten salt reactor development.

To my little ones, Alya and Mayed

Contents

List of Figures

List of Tables

Acknowledgments

I am indebted to the many people who have contributed to this work, directly or indirectly, and have enriched my time at UC Berkeley.

I thank my advisor, Professor Per Peterson, for distilling his passion in thermal-hydraulics into his teaching and mentorship. Thank you for your unwavering confidence, encouragement and patience along the way. Also, for inspiring me to think boldly about the future of this technology. I am also grateful to Professor Van Carey and Professor Raluca Scarlat for their time and support.

It has been an immense privilege and joy to work alongside my fellow Thermal Hydraulics research group members, Clara Alivisatos, Shane Gallagher, Ishak Johnson, James Kendrick, Theodore Ong, Christopher Poresky, and Dane de Wet. I thank Christopher Forsyth, Miles Wilson, Ta-Chun Wang for their significant contributions to the experimental work.

My deepest appreciation goes to my wife, Sara, whose unyielding support has been my source of strength and motivation. To my parents, Asma and Ashraf, thank you for nurturing my curiosity and instilling in me the values that will continue to inspire and shape my life journey.

Chapter 1

Introduction and SHEFRA Experiment Design

1.1 Introduction

Motivation

Experimental use of surrogate fluids for molten fluoride salt reactor development enables faster iteration at lower costs compared to prototypical fluid separate and integral effects tests. This is possible due to operation at reduced scales, including lower temperatures. The ability to retire development risk in stages mitigates the challenges associated with building a first-of-a-kind nuclear reactor, such as the fluoride salt-cooled high temperature reactor [2]. Qualification of surrogate fluids such as Dowtherm A includes the task of experimentally obtaining relevant heat transfer correlations for use in higher fidelity modeling. Dowtherm A can be made to match the Reynolds, Prandtl, and Grashof numbers of Flibe, simultaneously [26]. The present work investigates the use of frequency response methods to measure quasi-steady Nusselt number values for forced convection of Dowtherm A in circular ducts. This is accomplished by time-wise variation of the inlet fluid temperature. Under quasi-steady conditions, the Nusselt number measurements can be compared to steady state forced convection analytical solutions. The experimental and analytical methods used to accomplish this benchmark of thermal hydraulic similitude between quasi-steady surrogate fluid and steady state Flibe conditions may also be extended to match unsteady heat transfer conditions of interest.

Experiment Overview

The test section is a vertical copper tube that is two meters in length, with an outer diameter of 6.35 mm (0.25”), and a wall thickness of 1.24 mm. Unsteady forced convection is initiated in the tube by a periodically-varying fluid inlet temperature entering from the bottom of the test section. This chapter details the test system design process resulting in the described

features. Chapter 2 describes the experiment instrumentation and control platform, focusing on software architecture and programming, that was developed to facilitate arbitrary periodic power inputs to a heater located just upstream of the tube inlet. A mesh screen is placed after the heater to encourage mixing before the fluid enters the test section. At the inlet, a thermocouple is inserted in the flow to obtain an approximate measurement of bulk temperature. Surface thermocouples are placed on the tube wall exterior to measure wall temperature oscillations at different axial coordinate locations. The wall is approximated as a lumped capacitance in the radial direction, i.e. the wall temperature is assumed to be radially uniform.

The system design process prioritized the creation of conditions that would make appropriate this and other approximations as a means to limit the complexity of the selected heat transfer problem. To be more specific, the distortion from the unsteady flow temperature conditions would ordinarily produce an instantaneous spatially-dependent Nusselt number that deviates with respect to time. This distortion however, can be made tractable by identifying limited conditions in which the system is in a quasi-steady state. Reducing the physical complexity of the system facilities the model-based identification of quasi-steady state heat transfer conditions.

The present chapter explains how the test-section was designed utilizing a spatially-independent solution to the unsteady heat transfer problem, i.e. the fluid and wall temperature oscillations are not dampened as they travel through the test section. Chronologically, the design and construction of the SHEFRA experiment precedes the development of a space-dependent transient model described in Chapter 3. A few minor missed opportunities as a result of the incomplete theoretical framework at the design stage are discussed in Section 1.5.

In Chapter 3, the appropriate steady state forced convection Nusselt number analytical predictions and correlations are first identified for both laminar and turbulent flow in the SHEFRA test section. Steady state model assumptions are chosen to match those of both the space-independent and space-independent transient models. Section 3.2 proceeds to derive a the space-dependent quasi-steady model using a solution to the unsteady heat transfer problem as a starting point [22]. This quasi-steady model can be used to obtain optimal experimental parameters and conditions for laminar flow conditions only. A similar treatment would be necessary to apply to the unsteady turbulent flow heat transfer problem for determining quasi-steady conditions in turbulent flow, which is not covered in this dissertation. Chapter 4 describes the general time-domain model used for experimental data reduction that does not assume any imposed fluid inlet temperature boundary condition. The data reduction model is applicable for both laminar and turbulent flow however the analysis focused on the laminar regime. This is because the theoretical framework for determining quasi-steady conditions in the given problem covers only the laminar regime.

1.2 Background

Heat Transfer Problem

The problem of interest is that of forced convection in a circular duct with a periodically-varying inlet fluid temperature. Many experimental and theoretical investigations consider thermal transients can be initiated by timewise variation of wall temperature, wall heat flux, or internal heat generation. Investigations of such problems satisfy the need to understand the transient behaviour of heat exchange components. To a lesser extent, studies can be found in the literature of thermal transients imposed in regenerative heat exchangers by timewise variation of the fluid temperature at the inlet.

1.3 Fields of Application

Thermal Transients in Nuclear Power Plants

In a nuclear power plant, thermal transients can occur as a result of design-basis events, which are conditions of normal operation including anticipated operational occurrences (AOOs) and design-basis accidents. Categorization of events extend to anticipated transients without scram, which are AOOs that are followed by a failure of the reactor trip portion of reactor protection systems, and beyond design-basis events or accidents. The U.S. Nuclear Regulatory Commission (NRC) groups AOOs and postulated accidents in Light Water Reactors into the following seven types [15]:

1. Increase in heat removal by the secondary system
2. Decrease in heat removal by the secondary system
3. Decrease in Reactor Coolant System (RCS) flow rate
4. Reactivity and power distribution anomalies
5. Increase in reactor coolant inventory
6. Decrease in reactor coolant inventory
7. Radioactive release from a subsystem or component

Examples of events that would produce such AOOs include but are not limited to loss of power to recirculation pumps, tripping of the turbine generator, isolation of the main condenser, and loss of all offsite power. While these categorizations and examples are derived from LWR regulatory guidance, they are anticipated to be similar for molten salt reactors.

Thermal hydraulic phenomena that become important to consider in a molten salt reactor include heat transfer interactions between fluids and solid structures due to a similar thermal

inertia between the two. The primary working fluid in the reactor primary loop is of a high Prandlt number compared to water, and is designed to be single-phase during nominal operation and postulated accidents.

A simple channel geometry was chosen as the test-section and subject of the SHEFRA experiment to simplify the development of quantitative analysis methods but also because variations of this geometry can be found in molten salt reactors. A drop in temperature in the hot leg of the primary loop could be an example of thermal transient and AOO in a prototypical system. There is interest in developing accurate predictions of convection heat transfer in twisted tube geometries.

The utility of accurate predictions of thermal transients is to prevent degradation of structural materials, economic optimization of design and control parameters for nominal operation, and to perform safety analyses for design-basis and beyond design-basis accidents.

1.4 SHEFRA Experiment

Scaled separate effects tests (SETs) using surrogate fluids to study molten salt phenomena offer the opportunity to conduct a broad scope of experiments at lower temperatures, reduced height and area, reduced cost and with increased options for instrumentation. The scaled heat exchange frequency response analysis experiment (SHEFRA) conducted at the University of California, Berkeley aims to demonstrate the use of frequency response testing to obtain high fidelity convective heat transfer data for geometries and thermal hydraulic conditions that when scaled, are relevant to the operation of molten salt-cooled reactors. In contrast to steady-state or step-change tests, frequency response or dynamic testing enables the separation of effects due to processes with different time constants. This chapter documents the system design, and provides a description of the experimental platform. The experimental objective is to obtain measured Nusselt number values which will be compared to correlation predictions from literature, as well as existing data from molten salt experiments, and can be used to validate convective heat transfer constitutive models used in system-level codes. To facilitate this analysis, bulk fluid temperature and wall temperature data at five equidistant nodes along the channel are collected. Initial data captured showed wall temperature fluctuations that are progressively dampened as a function of axial distance.

Previous separate effects tests at UC Berkeley (UCB) made use of Drakesol 260AT mineral oil, which when used as a surrogate fluid, can approach the Prandtl number of flibe coolant in prototypical conditions relevant to the UCB Mark 1 Pebble-Bed Fluoride Salt-Cooled Reactor (Mk. 1 PB-FHR) and other fluoride salt-cooled molten salt reactors [3, 2, 11, 9]. An existing experimental facility, the pebble-bed heat transfer experiment (PB-HTX), was modified to deliver sinusoidal power frequency inputs to a low-thermal-inertia nichrome wire heater installed just upstream of the test-section in a forced convection oil loop. Because it is challenging to heat pebble beds uniformly, frequency response methods can use the thermal inertia of pebbles to provide a surrogate heat source. The resulting high-fidelity

data was expected to enable the characterization of convective heat transfer coefficients more accurately compared to data from a steady-state or step-change tests [9]. The experimental facility was restarted and upgraded with improved power control capabilities. Previously, pulse power inputs was manually controlled by turning the power supplies on an off by an operator. After the upgrades, additional pebble bed forced convection data was collected using a sinusoidal frequency response to supplement previous data. Pulsed power inputs were used as an approximation of a sinusoidal inputs as a compromise due to the hardware and software complexity of implementing sinusoidal inputs. A single frequency input provides Nu number parameter estimation opportunities that are explored in Chapter 5. A full description of the versatile analog control platform is presented in Chapter 2.

This experimental platform was then repurposed to study transient convection heat transfer in cylindrical channels using DOWTHERM A surrogate fluid, which between the temperatures of 50°C to 120°C, can be made to simultaneously match the Prandtl, Reynolds, and Grashof numbers of the molten fluoride salt coolant flibe (Li_2BeF_4) at prototypical reactor conditions. See Figure 4.7. The SHEFRA experiment is the focus of this chapter and the dissertation as a whole.

1.5 System Design

SHEFRA was designed before the development of a comprehensive theoretical framework for the dynamics of the physical system . This section documents the design objectives and constraints as they were perceived at the project's inception. This will be supplemented with retrospective commentary as necessary. Chapter 3 also provides an in-depth analysis with the aid of an analytical model for the dynamic system and specifically, the ability to determine under what conditions can the frequency response tests approximate steady state operation.

Surrogate Fluid Selection

Key design decisions such as the selection of loop fluid, the heating method, and the determination of important nondimensional parameters where informed by the experience and outcomes of past scaled experiments conducted at the Thermal Hydraulics Lab. Notable examples are the Compact Integral Effects Test (CIET) and PB-HTX. Dowtherm A, also branded as Therminol VP-1, was chosen for its ability to match the appropriate dimensionless parameters.

Dimensionless Parameters

Table 1.1 summarizes the relevant dimensionless groups for testing and the respective ranges of interest. Empirical correlations for forced convection in ducts typically predict Nusselt number as a function of Prandtl and Reynolds number. This is studied at a more fundamental

Table 1.1: Experimental parameters of interest for SHEFRA experiment testing. The Reynolds number range spans both laminar and turbulent flow regimes. The Prandtl number range is selected to match prototypical Flibe temperatures of interest, as shown in Figure 4.7. Mixed convection scaling, Gr/Re, was not considered at the initial design stage beyond creating the facility of a rotating test section for future investigations in a horizontal orientation. Mixed convection effects were later investigated in Section 3.1. For inlet temperature oscillation frequency, simulations of the space-dependent model derived in this section showed that higher frequencies resulted in greater temperature differences between the fluid and wall as shown in Figure 1.1. In the limit of low frequency, the wall temperature oscillation amplitudes approach that of the fluid. Through simulation, the range of frequencies from 0 to 0.5 Hz were sufficient to initiate the entire range of dynamic response for candidate test-section parameters. In practice, the upper frequency limit was set by power and control hardware specifications and candidate parameters were selected such that the entire range of possible dynamic response could be captured within the frequency limit. Dimensionless frequency was investigated at a later stage after the test-section was built (Chapter 3).

Parameter	**Designed Experimental Range**
Reynolds	500-10,000+
Prandtl	22-9
Grashof	Undefined
Frequency	$0 < f < 0.5$ Hz
Biot	≤ 0.1

level in Chapter 3. The Prandtl numbers of interest in the prototypical system determine the target surrogate fluid temperature ranges as shown in Figure 4.7. The Reynolds number ranges of interest span both laminar and turbulent flow. This creates a design constraint related to the inner diameter of the circular channel test-section. A variable frequency drive enables the collection of data at variable mass flow rates. A channel that can be approximated as lumped capacitance is desirable to achieve quasi-steady operation using the chosen heating method and to simplify the modeling and data reduction process. To ensure the validity of this approximation, a maximum Biot number constraint was imposed. Other relevant dimensionless parameters specific to forced convection with periodically-varying inlet temperatures were discovered after construction.

Signal-To-Noise

After determining the main design constraints, a dynamic model for the heat transfer in the test-section was derived to estimate optimal geometry and materials for high gain in the output. A system response with high gain reduces uncertainties propagated from measurement error. The derived model did not account for spatial-dependence and applies only near the inlet. In other terms, it would model a hypothetical channel where the fluid temperature os-

cillations are not dampened as they travel downstream. There were not a major consequence to this simplification, since the optimal maximum temperature delta between the fluid input and wall output response at the inlet of the test-section would also result in the maximum delta at the outlet. There were however some drawbacks to not having a spatial model. The optimization resulted in a wide range of materials and geometries combinations that were capable of achieving the maximum possible delta at the inlet. Had the model been able to predict the temperature delta at the farthest point downstream of the inlet, there would have been a smaller set or potentially a single optimal combination. Ultimately, a final decision was made from the potential combinations by selecting for a small Biot number, and also based on how close a geometry was to standard tube sizes readily available for order.

Quasi-Steady Approximation

The optimization would have been improved has it considered that quasi-steady forced convection is best approximated at either or low frequency inputs. The most Nusselt number distortion is expected within a middle range of frequencies that is determined by the experimental parameters (Chapter 3 and Appendix 4).

At the limit of very high frequency, the amplitude of the wall temperature oscillation converges to zero. To prevent high error propagation, an updated objective function would ensure that not only is the temperature delta between the fluid and solid temperature is maximized, but that the minimum oscillation amplitudes for each of the bulk fluid and wall temperatures do not fall below a certain value proportional a desired error tolerance.

The conditions for quasi-steady heat transfer were not fully defined at this stage beyond a general intuition. Despite this, an analysis of experimental results in Appendix 4 showed that the as-configured system was ultimately able to reach frequencies high enough for a satisfactory approximation of quasi-steady heat transfer.

Modelling and Optimization Implementation

Using an energy balance,

$$-\dot{E}_{in} = -\dot{E}_{out} \tag{1.1}$$

where $\dot{E}_{in}$ and $\dot{E}_{out}$ is the energy flow into and out of the test-section. Assuming a lumped capacitance for the copper tube wall we get,

$$\frac{dT}{dt} = \frac{hA_s}{\rho V c}\left[T - T_\infty(t)\right] \tag{1.2}$$

where h is the heat transfer coefficient, A_s is the heat transfer area, T is the solid temperature, and T_∞ is the fluid temperature. The input can be described as a function of heater signal frequency, f, a baseline temperature T_b, and scaling parameter K.

$$T_\infty(t) = -K \sin 2\pi f t + T_b \tag{1.3}$$

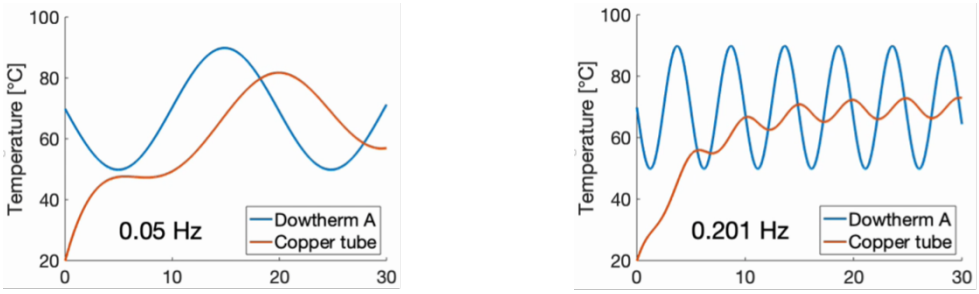


Figure 1.1: Transient heat transfer simulation of Dowtherm A surrogate fluid to a single axial location at the inlet of a cylindrical copper channel at two different sinusoid heater input frequencies.

This is a linear non-homogeneous system which can be solved to obtain a function for the temperature difference distribution as a function of time.

$$\begin{aligned}\theta(t) &\equiv T(t) - T_\infty(t) \\ &= c_1 \cos 2\pi f t + c_2 \sin 2\pi f t + c_3 \\ &\quad + c_0 \exp \frac{-hA_s t}{\rho V c} \\ &\quad - [T_b - K \sin 2\pi f t]\end{aligned} \tag{1.4}$$

The model was used to simulate the heat transfer interaction between the tube wall and fluid at various frequencies. As mentioned, the model is imperfect because it does not account for axial variation in the tube solid temperature and does not show an expected dampening effect of the solid wall temperature along a long tube. Figure 1.1 quantifies the difference of fluid and solid temperatures possible for a candidate combination of surrogate fluid, solid structure material and test-section geometry.

A simplified search algorithm written in MATLAB was used to simulate the model and select an optimal combination of SHEFRA test-section geometric and material parameters to maximize the difference in solid to fluid temperatures. The test-section would also have to pass model constraints that are derived from model assumptions and approximations made, such as a lumped capacitance approximation for the tube wall.

1.6 Test-section Detailed Design

Material Selection and Geometry

The optimization model described in the previous section led to the selection of a two meter long, 0.25” outer diameter, 0.049” thick wall copper tube test-section. Standard copper tube

size was found to be too thin to provide sufficient thermal inertia for our application. A Navy-specification standard copper tube was found to be closer to the target geometry for initial scoping experimental studies. The length of the test-section was chosen to be sufficiently long to acquire a measurable temperature distribution and to study axial dispersion effects. This decision would limit our ability to compare experimental results with equivalent forced convection experiments conducted at Oak Ridge National Lab (ORNL) for laminar flibe flow, as the molten salt results were provided as averaged heat transfer coefficients over an entire entrance region and for fully developed flow. Later calculations, confirmed by experimental results, estimated that laminar flow is still thermally developing at the outlet of the SHEFRA test-section.

Structural Considerations, Plumbing, and Instrumentation

To support the test-section and to facilitate the routing of thermocouple wiring, the loop features a strong-back that rotates along the same axis. The strong-back has a dual-purpose as return plumbing in the form of a rigid pipe. This lends itself to a compact design and a reduction of complexity that was initially introduced due to the rotating test-section feature.

The test-section assembly consists of the copper tube and stainless-steel Swagelok elbow tube fittings on either side. The Swagelok fittings are threaded into a stainless-steel tri-clamp straight adapter. At the test-section outlet, the tri-clamp tube fitting is connected to a 1.7" tube extension, which serves as an instrumentation node. Similarly, other nodes are located at other loop locations as part of a modular "drop-in" instrumentation system. A series of tri-clamp pipe fittings make up the return plumbing, utilizing PTFE for the gasket material.

The wall temperature of the test section is measured at different locations along its length using cemented surface thermocouples. To measure the inlet fluid temperature, a thermocouple is inserted into a modified Swagelok adapter at the test-section inlet. This was done to minimize the parasitic heat loss distortion of a bulky instrumentation node and to minimize transport delays.

1.7 Experimental Setup

Loop Overview

The loop consists of several components. There is a heater placed immediately before the inlet of the test section. An oil-to-water plate heat exchanger is used, an Ameridex Plate Exchanger, Model No. AMDX-10-8 to cool the oil after it exits the test-section.

Flow is provided by centrifugal pump from Price Pump Co., Model no. LT25SS-344-21276R-33-36-3T7, with 0.33 HP with a variable frequency drive. A reservoir tank provides room for thermal expansion and prevents oscillation signals echoing back into the test-section. A Coriolis mass flow meter is used to measure flow (Siemens Coriolis Flow Sensor SITRANS FCS400).

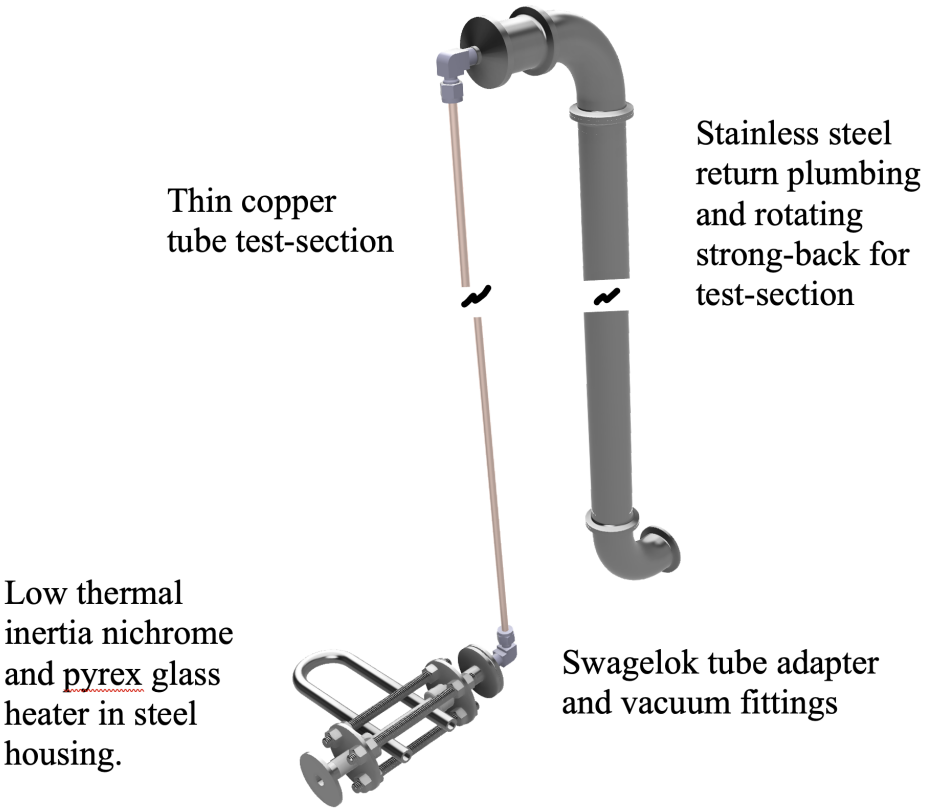


Figure 1.2: This figure shows the CAD models of the glass and Nichrome wire heater, the copper test-section and the return plumbing that has a dual-purpose as a strong-back for structural integrity and the routing of thermocouple wiring.

Specialized fittings and ports are used for thermocouples and pressure gauges. A cover gas system was incorporated into the design to prevent DOWTHERM A/Therminal VP-1 degradation and to control fumes. This system is integrated as part of a gas pressurization system for filling and draining.

Fill/Drain and Cover Gas System

The 3 meter height of the experimental facility necessitated the design and construction of a pressurization system to fill and drain the loop. The fill/drain tank was fabricated out of a 12" nominal 304SS pipe and was fitted with two fill/drain points that connect to the experimental loop and a quick-connect hose fitting for pressurization. Argon gas flows from a pressurized gas cylinder regulated to 20 psi through a series of valves and a rotometer, used to control the direction and flow of gas. During a fill procedure, the pressure exerted by the gas transports the oil out of the tank and into the loop. The oil moves from the

bottom of the tank through vertical dip tubes and out through the fill/drain points. During high temperature operation (above 80F), a constant flow of gas is routed to the reservoir tank to prevent chemical degradation of the oil from exposure to air. A standard operating procedure for filling and draining was drafted. The pressure needed fill the loop to the required height was calculated to be about 5 psi using the relationship for static pressure $P = \rho g h$. A pressure safety valve rated at 10 psi was installed on the drain tank. During the filling, 9 psi of pressure was used to speed up the procedure.

Heating

Sinusoidal heater DC electrical power inputs are used to heat the oil before it entered the copper test-section. A range of mass flow rates is possible with the variable frequency drive-controlled centrifugal pump to collect convective heat transfer data across a range of Reynolds numbers. For frequency response experiments, the DAQ must be able to send and receive signals to A DC power supply at the desired high control frequencies. The heater is a custom-made component made of Pyrex glass with an internal nichrome wire heating element, pictured in Figure 1.4. The heater element was adapted from a 10 kW electric water heater and features a low thermal inertia. This allows the element to respond rapidly to changes in power. With the fast-acting heater design, the sampling and control rates can be increased, hence increasing Nyquist frequency and allowing for the investigation of heat transfer at higher frequencies. Previous experience with digital control of the hardware power supplies achieved control rates that were insufficient for the time scales of interest. Instead, analog controls are used to interface the DAQ and power supply. The control system is discussed in detail in Chapter 2

Multiple iterations of the heater design were fabricated before arriving at the final version, starting with its initial use as an off-the-shelf residential electric instant water heater. In these previous experiments, the heating element was used in its original plastic housing. Researchers noted that the plastic was decomposing when coming into contact with the heat transfer fluids used [10]. The new glass design was chemically compatible but it was also more susceptible to breakage. A metal frame was designed and fabricated to eliminate any potential mechanical stresses. However, the design neglected the possible stress on the glass at the connections between the heating element terminals and the electrical leads from the power supply. In one instance, the heater was broken as the terminal screws were being tightened. The design could be improved extending an insulated rigid structure to the electrical leads, or by eliminating the use of glass completely.

Reservoir Tank

The reservoir is an off-the-shelf stainless-steel tank with tri-clamp fittings. It is located at the highest point in the loop where there is a free surface of oil. A vent to ambient air first travels through Teflon tubing down to an oil overflow drum and then out through a second opening.

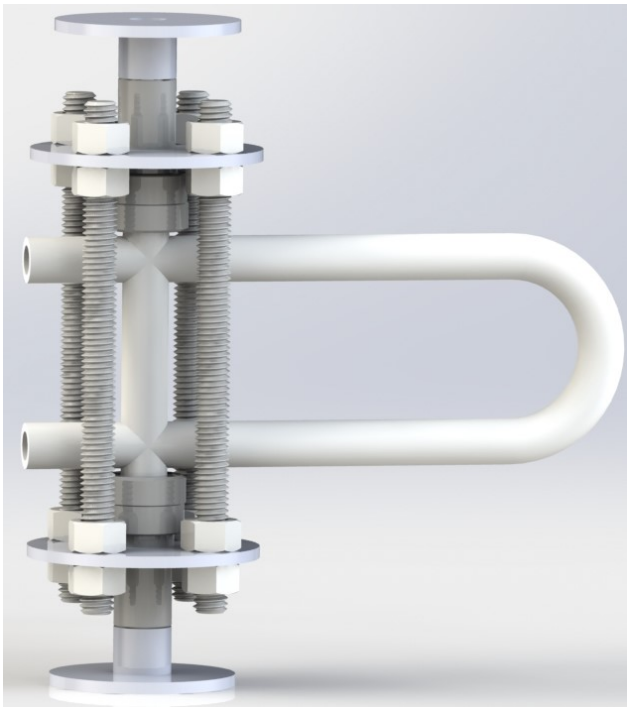


Figure 1.3: Fast-response Pyrex glass heater CAD render

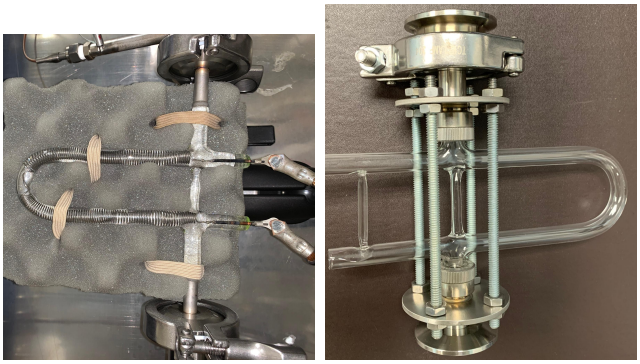


Figure 1.4: Fast-response Pyrex glass heater, (left) improved heater design with vacuum fittings and sightglass frame. (right)

Assembly and Construction

A CAD model was used to plan the layout and positioning of the various loop components. The oil pipe section lengths and sizes were also determined using this method. Further, the fittings selected were tested directly in the model for compatibility. A combination of stainless-steel flexible tubing and tri-clamps fittings were used (sometimes referred to as sanitary quick-clamp fittings) to simplify the assembly process. The frame was constructed out of 80/20 T-slot aluminum extrusions. A parts list for each sub-assembly was prepared as well as a master build document for reference during construction. Engineered safety controls were incorporated into the design to protect operators. Acrylic sheet panels were fitted to the front of the experimental loop to protect personnel from hot oil in the event of a high flow leak. The loop was placed into a welded stainless-steel drip tray of sufficient volume to contain a leak.

1.8 SHEFRA Data Collection

SHEFRA was successfully used to collect transient temperature data using five surface-thermocouples cemented to the copper tube wall. The bulk fluid temperature of the fluid entering and exiting the test section was also recorded. A sample of the collected data is shown in Figure 1.7. While the data shown is for a single frequency, a range of heater sinusoidal input frequencies were collected to capture the frequency-dependent dynamic system behavior.

Flow Meter Measurements

A Coriolis mass flow meter is a highly accurate instrument for measuring flow rate. The unit outputs a 4-20 mA current signal that is sent to the DAQ that is proportional to the measurement value. The signal is read using a DAQ voltage input channel. A known resistance placed in parallel configuration with the two flow meter leads, enabling the current value to be estimated using the DAQ voltage measurement in the LabVIEW project.

The SHEFRA experiment has been designed and constructed to measured heat transfer in a smooth tube with the objective of demonstrating thermal hydraulic similitude using a surrogate fluid for molten fluoride-based salt. A frequency response testing technique is used to characterize the dynamic system in both the time and frequency-domains. Experimental data will be compared with past molten salt experiments and predictions from the literature.

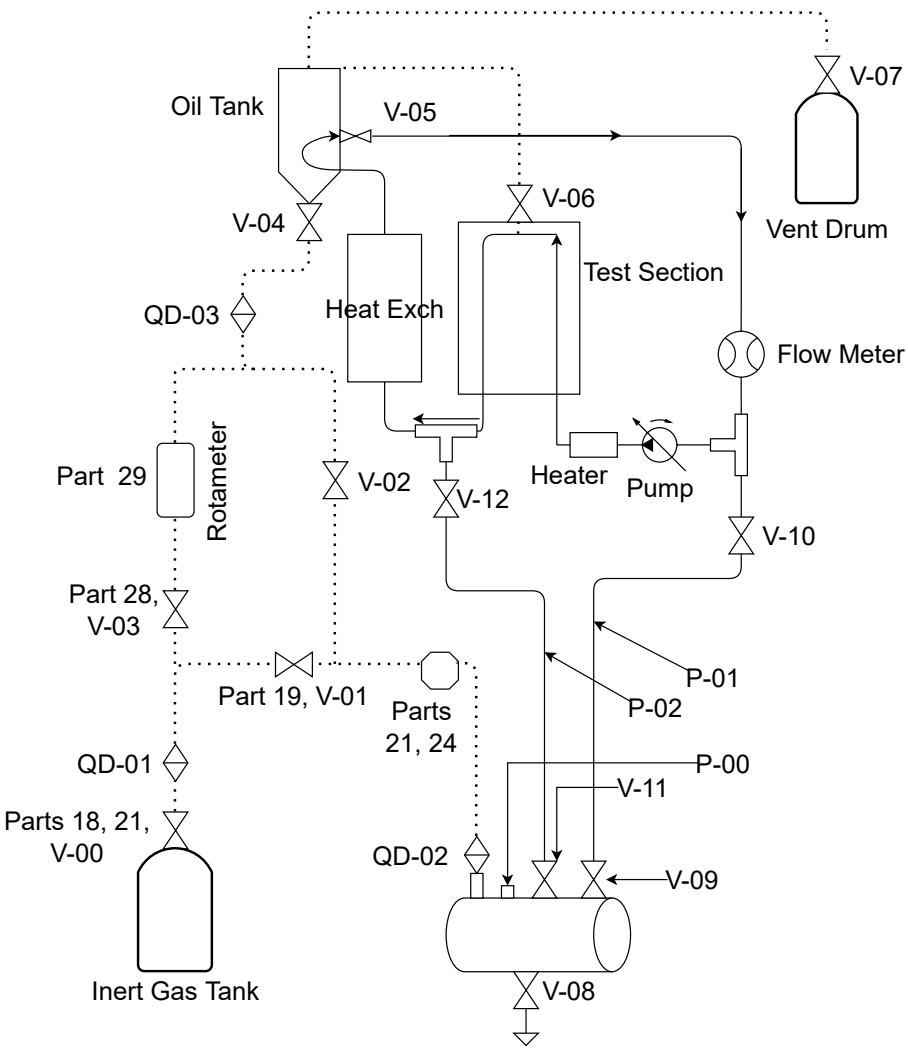


Figure 1.5: Detailed loop diagram that includes a pressurized gas fill and drain system, cover gas manifolds, safety pressure relief mechanisms and the primary oil loop.

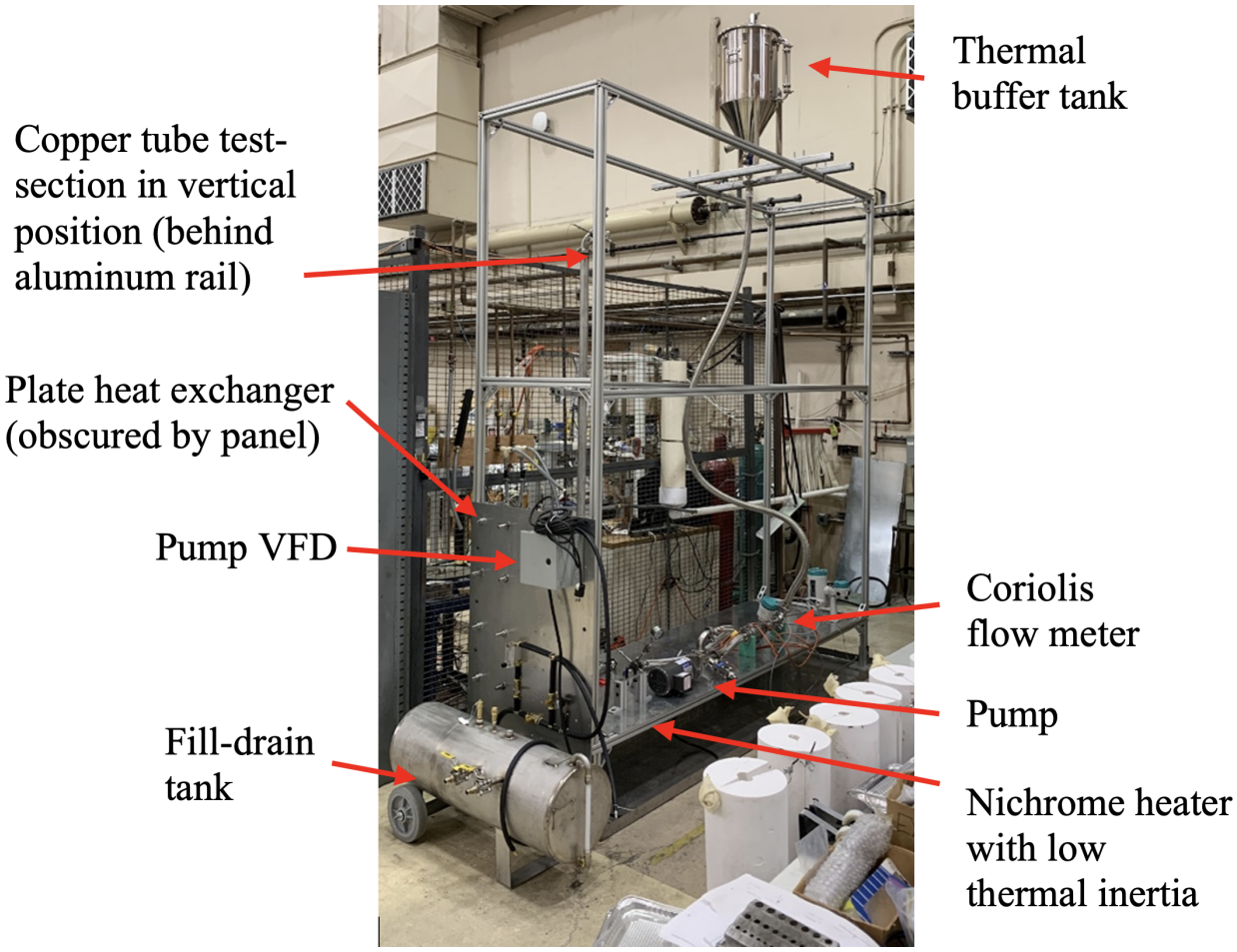


Figure 1.6: The SHEFRA experimental loop after construction.

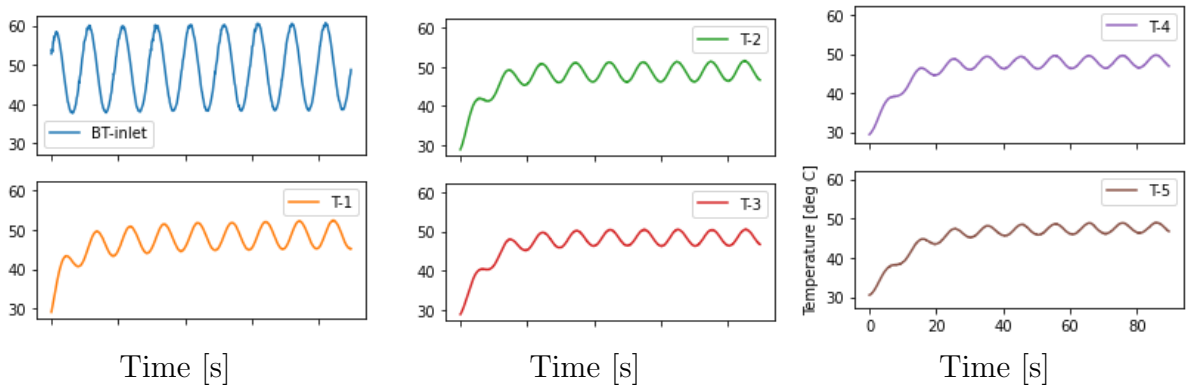


Figure 1.7: Solid temperatures measured at five equidistant thermocouple locations along the tube outer wall, and the bulk fluid temperature at the test-section inlet as a function of time. A mesh screen is placed to encourage mixing of the flow before the bulk temperature measurement. T-1 is the thermocouple closest to the test-section input. It can be observed how the temperature signals are progressively dampened along the tube towards thermocouple T-5.

Chapter 2

Instrumentation and Control: Versatile Test Bay

2.1 Development of a Modular Platform for Scaled Separate Effects Tests

The application of surrogate fluids for scaled Separate Effects Tests (SETs) broadens the scope of experiments that can be conducted in a university setting by allowing for relatively low-temperature operation and increased options for instrumentation and control. An unanticipated benefit has been the ability to conduct quasi-steady heat transfer experiments under periodic forcing that would not be possible to implement using molten salt and graphite without significant distortion. For example, heat transfer data obtained by submerging graphite spheres in Flinak, coupled with an analysis of the characteristic time constants for forced convection flow over a sphere showed that the quasi-steady condition could not be achieved [9]. The unique ability of Dowtherm A surrogate fluid and copper pebbles of a specific geometry to approximate quasi-steady heat transfer under periodic forcing was the foundation for the Pebble-bed Heat Transfer Experiment (PB-HTX). Similarly, we expect to be able to use Dowtherm A and copper to measure quasi-steady heat transfer in a circular channel under the right conditions.

The PB-HTX was chosen for conversion into the Versatile Test Bay (VTB) due to its modular construction. The VTB is a SET facility capable of quick analog control of hardware power supplies for simulating transient conditions and performing frequency response testing in the installed test-section. This test-section can be easily swapped out for future scaled experiments. It is capable of sending a variety of power profile inputs into the system through the heater. Utilizing the VTB, the collection of pebble-bed and circular channel heat transfer data using sinusoidal power inputs was demonstrated. This chapter serves as a detailed technical guide to the Instrumentation and Control in the VTB.

Versatile Test Bay

The Versatile Test Bay (VTB) is separate effects test (SET) facility was previously host to the pebble-bed heat transfer experiment (PB-HTX) at the University of California, Berkeley [9]. The pebble-bed was constructed using a dimpled glass cylinder filled with copper pebbles in which heat transfer to circulating oil was measured using temperature measurements at different radial and axial locations within the test section. A heater and oil-to-water plate heat exchanger were used to vary the temperature of the oil before it entered the pebble-bed. The most basic procedure involves using a periodic function to control the power to the heater to create the required transients. Sinusoidal heater inputs, at different mass flow rates achieved with a variable frequency drive-controlled centrifugal pump, were used to collect packed-bed convective heat transfer data across a range of Prandtl and Reynolds numbers. Although PB-HTX was designed for use with DOWTHERM A mineral oil, the completed experiments were done with Drakesol 260AT mineral oil, which is easier to work with compared to DOWTHERM A. However, it is not capable of reaching the UC Berkeley Mark 1 pebble-bed fluoride salt-cooled high temperature reactor (Mk1 PB-FHR) prototypical ranges for Prandtl and Reynolds numbers. This was not a concern, as the aim of the experiment was simply to demonstrate a scaled pebble-bed heat transfer experiment using a surrogate fluid. The major technical challenges were to develop a robust, fast-responding resistance heater system and to implement robust control of the heater power supply using LabVIEW and data acquisition (DAQ) hardware. In the past, the procedure required an operator to manually turn the power supply on and off at timed intervals for a pulsed test. For frequency response experiments to be possible, the DAQ must be able to send and receive signals to the power supply at high control frequencies. With the fast-acting heater design, the improved sampling and control rate increased the Nyquist frequency, allowing for the investigation of heat transfer at higher frequencies. Previous experience with digital control with these power supplies achieved a maximum control rate of 1 Hz, which would be insufficient for the time scales of interest. Instead, analog controls are used to interface the DAQ and power supply. An overview of the design of the VTB in addition to a detailed description of the hardware and software implementation for analog control for high-frequency dynamic signal input is included in this technical paper.

2.2 Experimental Set-up

The heater is a custom-made component made of Pyrex glass with an internal nichrome wire heating element. The heater element has a very low thermal inertia and responds rapidly to changes in power. The electrodes are connected to a TDK-Lambda GEN 250-40 power supply capable of delivering 40 A at 250 V (10 kW).

Figure 2.1: Photograph of copper pebble-bed test section.

Figure 2.2: Photograph of the original VTB oil loop before it was disassembled to construct the SHEFRA loop.

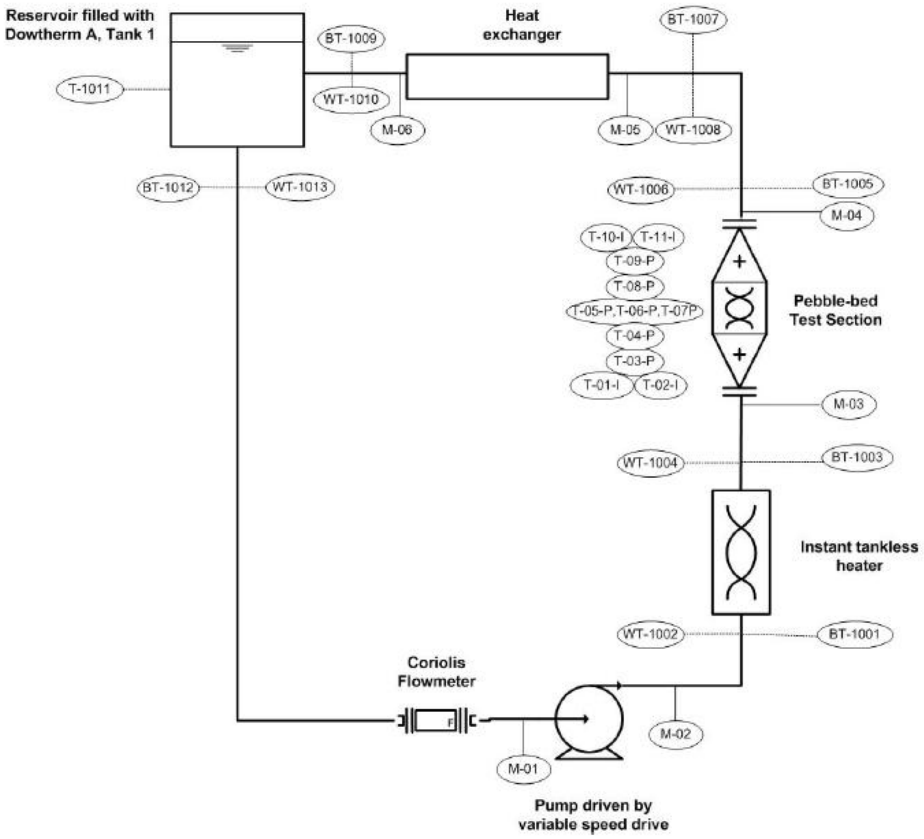


Figure 3-5: Flow schematic of PBHTX, with unidirectional flow in the direction of the arrow

Figure 2.3: Flow schematic of PB-HTX, with unidirectional flow in the direction of the arrow. [11]

2.3 Data Acquisition and Control

Equipment

A Windows PC is connected through USB serial to a National Instruments (NI) SCXI-1000 data acquisition (DAQ) chassis including a NI SCXI-1600 module for the collection of system data. The thermocouples and flow meter are connected to a SCXI-1102 voltage input module via a SCXI-1303 32-channel isothermal terminal block. Analog input and output modules were used for the control of the power supply. SCXI-1124 and SCXI-1102B +/- 10 V analog output and input modules were installed with SCXI-1325 and SXCI-1300 terminal blocks, respectively.

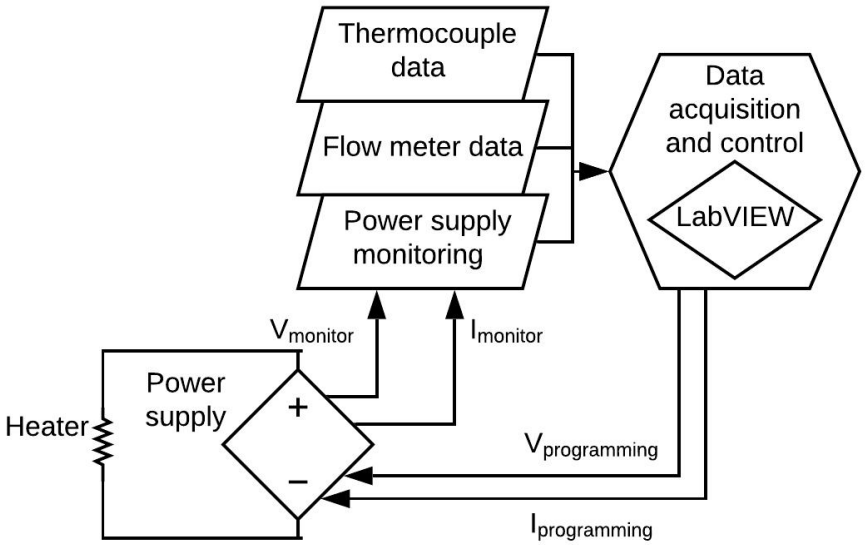


Figure 2.4: Simplified data and signal flow diagram of the VTB.

Resistance Feedback Control

The power is programmed using resistance feedback with four connections between the DAQ and the power supply. Two of these connections are voltage and current monitoring signals from the power supply. The remaining two connections are voltage and current programming signals. Resistance feedback is implemented by first using the voltage and current monitor readings to calculate the estimated resistance in the circuit using Equation 2.1,

$$R_{monitor} = V_{monitor}/I_{monitor} \tag{2.1}$$

then solving for the required voltage output to produce a user-specified desired power using Equation 2.2.

$$V_{out} = \sqrt{P_{desired} \times R_{monitor}} \tag{2.2}$$

The programming is implemented using LabVIEW and NI MAX. See Section IV for more details. The calculated value is scaled to a range of 0-10 V, where 10 V corresponds to the maximum voltage output of the power supply. The $V_{monitor}$ and $I_{monitor}$ analog input signals are scaled similarly. A current limit is set at the power supply maximum of 40 A by assigning this value to I_{out} (which translates to a 10 V signal). The power supply documentation provided instructions on the J1 wiring configuration for remote analog control operation.

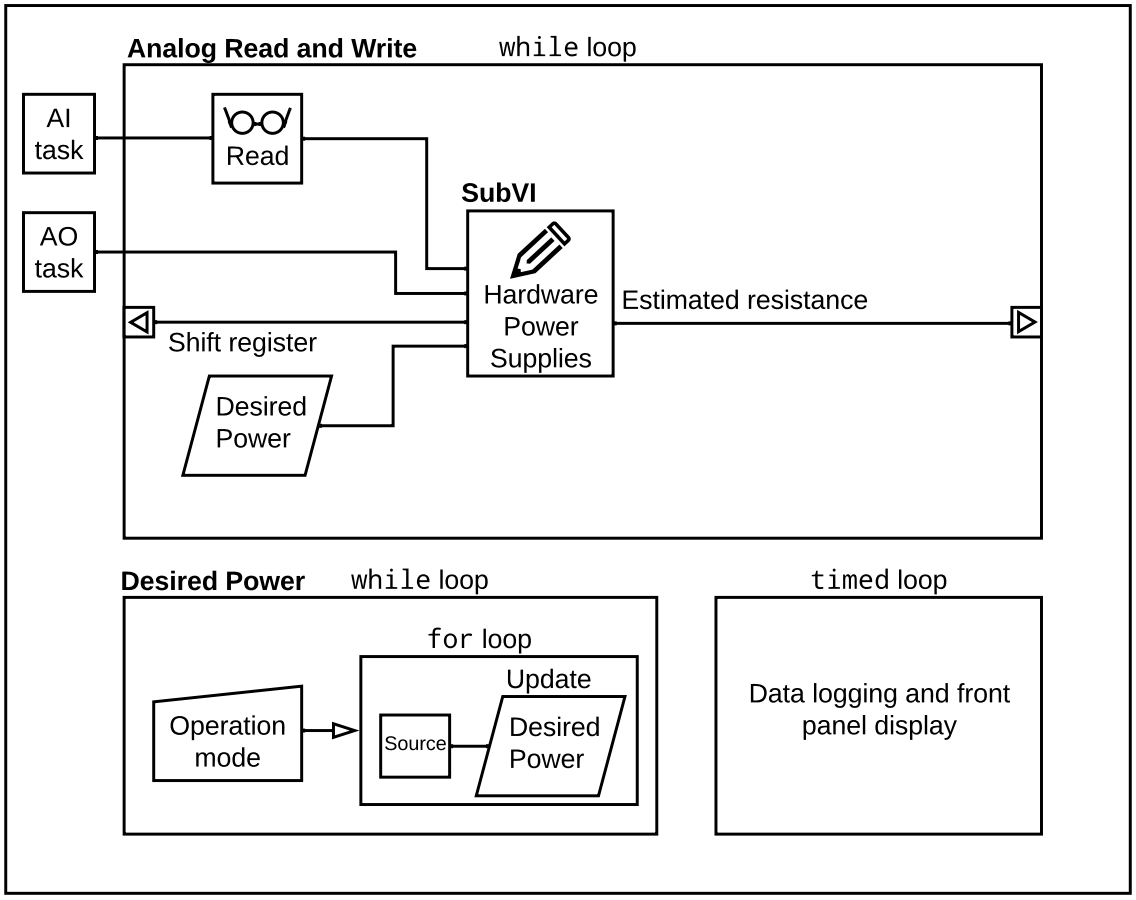


Figure 2.5: Simplified diagram of the main LabVIEW VI.

2.4 LabVIEW

The LabView programming architecture consists of one main graphical program, called a Virtual Instrument (VI), with three main loops running simultaneously. In one loop, the program reads the acquired analog signals and also determines what programming signals to send back to the power supply. The calculations are done in a nested SubVI called Hardware Power Supplies. A second loop continuously updates a local variable called Desired Power. This variable is an input into Hardware Power Supplies. In the third loop, the data is logged in a CSV file and sent to a front panel that is displayed during operation. A simplified representation of the VI can be found in Figure 2.5

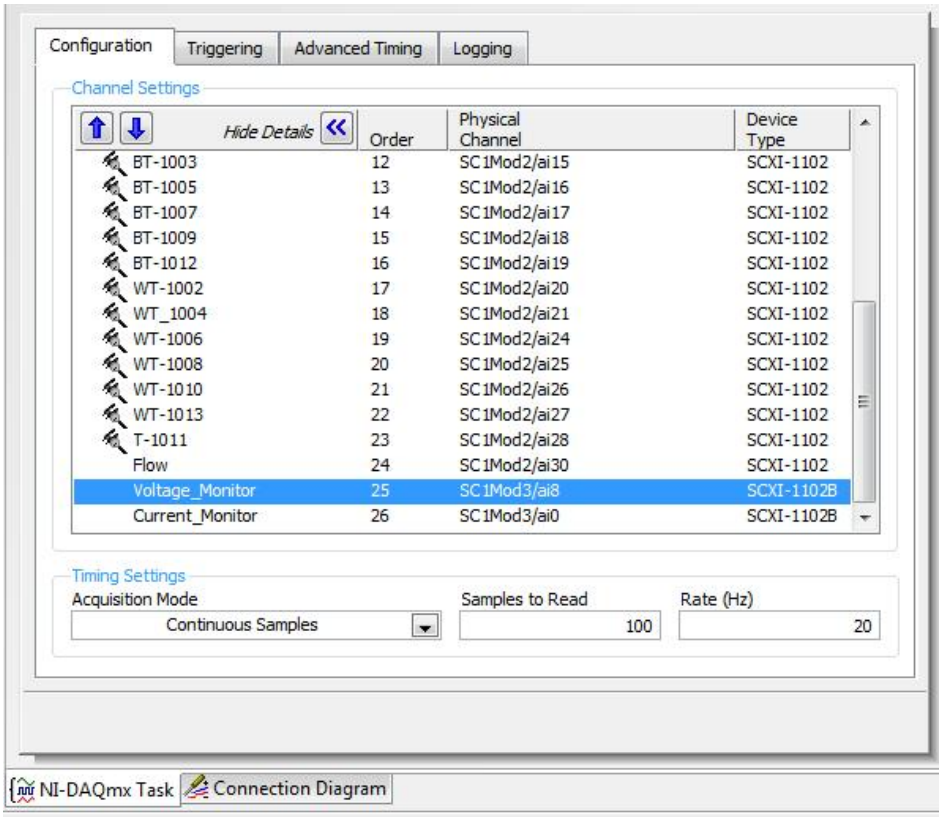


Figure 2.6: NI MAX is used to create and configure LabVIEW Tasks. This figure shows the virtual channels included in the analog input task.

Task Configuration Using NI MAX

LabVIEW tasks, which are "software entities that encapsulate the physical channel along with other channel specific information—range, terminal configuration, and custom scaling—that formats the data," are generated using NI MAX. It is important that the latest NI-DAQmx drivers are installed on the PC for NI MAX to detect the DAQ hardware. The SCXI chassis is found and configured under Devices and Interfaces. NI-DAQmx tasks are created under Data Neighborhood. One task was created to handle all analog inputs including flow, thermocouple, and power supply monitor data. Thermocouples may be calibrated using the NI MAX calibration wizard. Flow and monitor data signals were configured with a 0-10 V input signal range and no scaling. The appropriate timing settings depend on the software or hardware limitations of the set-up. Another task is created for analog write channels V_{out} and I_{out}. The timing settings for these channels are set to "1 Sample (On Demand)".

Desired Power Loop

Three operation modes were implemented: a power profile can be imported, generated using controls on the front panel or the desired power can set to a single value using a manual control. A while loop constantly checks the user-selected operation mode and generates the correct values of Desired Power. In the case of power profile import or generation, a for loop can be used to loop through a comma separated value file or a generated sequence, respectively. A manual time pause is added to the for loop to produce the correct period. The for loop is replaced with a sequence block for manual control operation, to force a pause before assigning the user input to Desired Power.

Analog Read and Write Loop

Outside of the second while loop, the analog input and analog write tasks are initiated and brought into the loop and enter a sequence block at different points. In the first step, an "Analog 1D DBL NChan 1Samp" read block is used to read the analog input task. In the second step a custom SubVI called Hardware Power Supplies is used to generate current and voltage analog output signals using various inputs, including the power monitor signals from the previous step. In a single loop iteration, the SubVI reads $V_{monitor}$ and $I_{monitor}$, calculates an estimated resistance in the circuit, and determines how much voltage the power supply must provide to meet the desired power.

SUBVI: Hardware Power Supplies

Hardware Power Supplies takes in the Desired Power local variable, in addition to an input resistance, maximum power limit, power monitor analog inputs, and the analog write task. The outputs of this SubVI are the completed analog write task, an output resistance local variable, and the output power for front panel display and data logging. The output resistance becomes the input resistance in the next loop iteration by using a shift register.

Inside the SubVI, the resistance feedback control is implemented. The input resistance and the desired power is used to calculate V_{out} which is then scaled to a 0 to 10 V range by dividing by 25. This scaling factor is calculated by dividing the power supply maximum V_{out} by the maximum threshold for analog out voltage. Similarly, for I_{out} 40 A (the current limit) is divided by 4 (the maximum current in Amps divided by 10 V). For safety purposes, coercion blocks may be used to constrain variables to be within appropriate bounds. For example, the desired power input is coerced to be between the maximum power limit (10,000 W) and zero. In addition, analog output signals are coerced to stay within 0 to 10 V before being written out using an analog write block. The SubVI also calculates a new estimated output resistance using $V_{monitor}$ and $I_{monitor}$, after being scaled similarly in reverse. The output resistance is manually bounded using a coercion block to typical operational values to prevent instabilities.

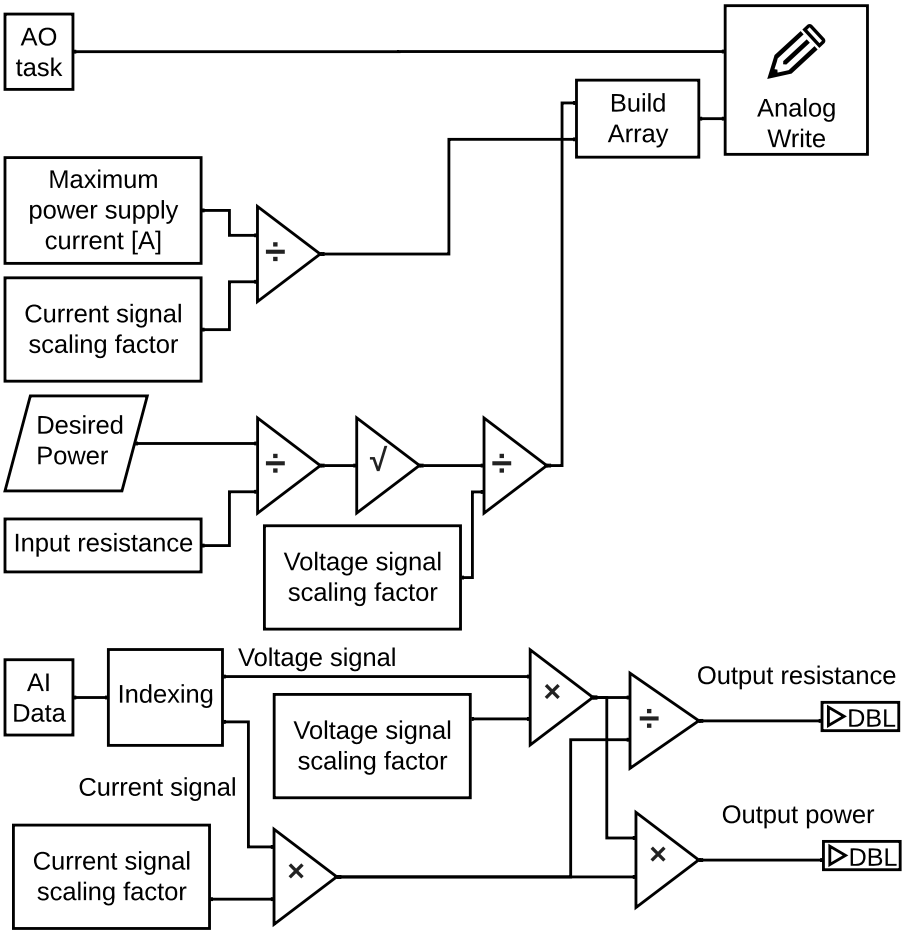


Figure 2.7: Simplified diagram of Hardware Power Supplies, a SubVI implementation of resistance feedback control.

Data Logging

Data logging is implemented in a timed loop that is configured to run in 100ms intervals. The same loop is also used to display different readings on the VI front panel. Data is saved to a comma separated value file after the VI is stopped. It is best practice to save data during operation to prevent any loss of data.

2.5 Results

The capability to produce periodic pulsed and sinusoidal heater inputs using analog controls was implemented for the purpose of collecting pebble-bed heat transfer data. Sample inputs

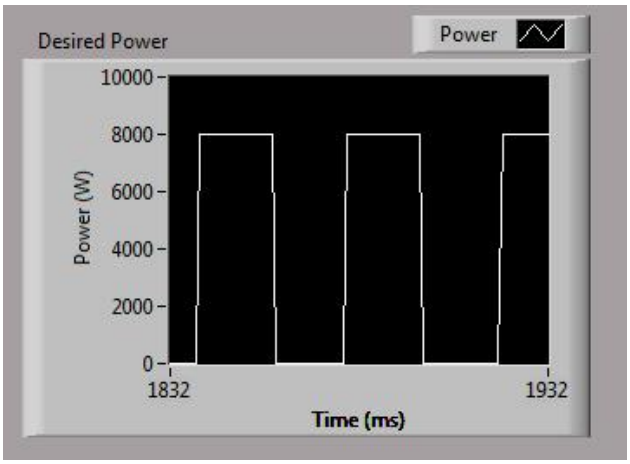


Figure 2.8: Pulsed power input.

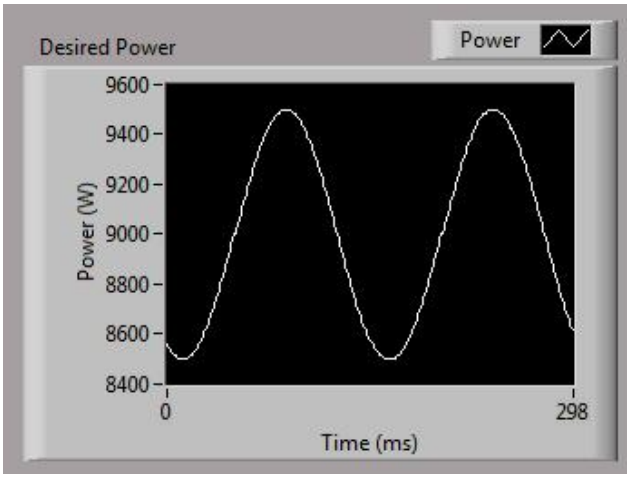


Figure 2.9: Sinusoidal power input.

for power can be seen in Figures 2.8 and 2.9.

The conversion of the PB-HTX facility into VTB was accomplished by the implementation of high frequency dynamic analog controls, and thanks to modular construction that will allow the installation of test sections of various geometries. Resistance feedback control using LabVIEW was used to generate dynamic heater inputs which may be used for collecting heat transfer data and for frequency response experiments.

Additional details are provided on how the desired heater input signals are generated in Appendix B.

Chapter 3

Theoretical Modeling and Frequency Scaling

Initial scoping studies on the SHEFRA experiment and chosen test-section resulted in Nusselt number measurements that needed to be interpreted using an appropriate predictive analytical framework. A review of steady-state analytical models was conducted to obtain a comparable reference depending on the flow development conditions. The rest of the chapter outlines the development of a theoretical framework for the problem of transient forced convection in circular channels with periodically-varying inlet temperatures. Recognizing the potential for analysis of this particular problem in the frequency domain, a solution method from the literature was chosen for its use of the Laplace Transform technique; resulting in

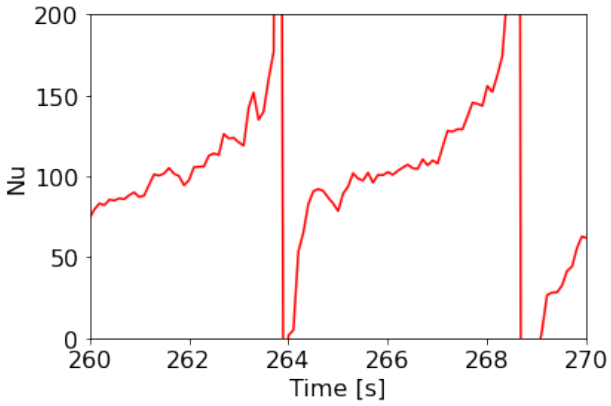


Figure 3.1: Data reduction after initial SHEFRA experimental runs for a transitional-turbulent flow showed Nusselt number measurements that appear to be distorted as shown in the plotted sample data. Similar distortions were observed during PB-HTX experiments, where they were assumed to be due to instrumentation error. The findings presented in the chapter show that the contribution of instrumentation errors was most likely overstated.

nondimensional mathematical expressions for the channel wall temperature, bulk fluid temperature, and wall heat flux in the frequency domain. The solutions were extended by the application of a quasi-steady model to obtain an expression for the Nusselt number as a function of longitudinal distance and other system parameters. Quasi-steady state can be predicted using the transient model by assessing if the Nusselt number for a fixed point is flat over an oscillation cycle within some tolerance. This prediction capability, in coordination with the scaling terms detailed in this chapter, can be used to to design frequency response test-sections and operating parameters tailored for various flow development conditions, wall thermal capacitance effects. The model was applied to the existing SHEFRA test-section and used to select optimal experimental operating conditions for the purposes of model validation. The analysis revealed that the validity of quasi-steady had high sensitivity to natural convection to ambient. This is a likely cause for the distorted Nusselt number measurements made in initial SHEFRA experimental runs that were performed without external insulation, as illustrated in Figure 3.1. Higher inlet fluid temperature oscillation frequencies into the SHEFRA test section result are favorable for adherence to quasi-steady state but only to a certain extent. In the limit of very high frequencies, the wall temperature oscillations amplitudes are small compared to measurement uncertainties. While initial experimental runs were conducted in the transitional flow regime, the described transient model in its current form is only applicable for laminar flow. Model-predicted optimal inlet fluid temperature frequency for laminar flow are when quasi-steady heat transfer is best approximated. This occurs at the limit of either high or low frequencies. At the low range, such as at an oscillation frequency of 0.025 Hz (a 40 sec period), the quasi-steady heat transfer will approximate steady-state heat transfer with a uniform heat flux boundary condition. At the high frequency limit, beyond 0.25 Hz (a 4 second period), the quasi-steady heat transfer approximates steady-state forced convection a with uniform wall temperature boundary condition. The most deviation in the Nu number is expected for frequencies in the mid-range.

The optimal frequency can be reliably found by incrementally increasing or decreasing the relevant scaling term only to the extent that requirements for quasi-steady state are met. At quasi-steady state, minimal deviation in the Nu number is present, depending on the amount of ambient convection distortion. Further, quasi-steady Nu values are expected to approach the steady-state values predicted by the literature.

3.1 Review of Steady-state Models and Relevant Applications

To help assess the validity of the transient analytical solutions presented further in this chapter, steady-state models that are relevant to SHEFRA experimental parameters and operating conditions are reviewed to be used as a reference and compared with transient predictions when appropriate. Recalling that the objective of the experiments is to measure the Nusselt number for the forced convection of high Prandtl number flow, the heat transfer

Table 3.1: Dimensional group ranges for the quasi-steady and space-dependent SHEFRA experiment model. The Reynolds number range of applicability covers only the laminar flow regime. The Prandtl number range is selected to match prototypical Flibe temperatures of interest, as shown in Figure 4.7. Mixed convection scaling, Gr/Re, was neglected in the derived model. Mixed convection effects are expected to be negligible for small values of Gr/Re. This is discussed further in Section 3.1. Quasi-steady state conditions are best approximated at the limit of relatively low and high values of b^*, which is the product of dimensionless frequency and the ratio of wall thermal capacity to fluid thermal conductivity. The quasi-steady condition was determined by finding b^* values at which predicted Nu numbers asymptotically converge for a simulated test-section. The plotted Nu number predictions in 3.11 and 3.12 show no change beyond and below threshold b^* values. This is put in terms of dimensional frequency for the SHEFRA test-section.

Parameter	**Model Applicable Range**
Reynolds	< 2300
Prandtl	$22 - 9$
Gr/Re	$\lesssim 1$
b^*	$b^* \lesssim 10$, $b^* \gtrsim 100$
Frequency	$f < 0.025$ Hz, $f > 0.25$ Hz
Biot	$\lesssim 0.1$

characteristics of the transient problem in a quasi-steady state is expected to be comparable to that of a steady state version of a system where the inlet fluid temperature is not being varied. This section proceeds through simulating the behavior of an equivalent steady-state system to benchmark the SHEFRA quasi-steady model results.

Before applying the steady state models from the literature to the SHEFRA system, simplifying assumptions and approximations must be made and justified. Some of these approximations will be applicable in both the quasi-steady and steady state models. To accomplish this, the section will start from the relevant fundamental governing equations.

Selecting the appropriate steady state analytical model first requires determining how the fluid in the test section is thermally and hydraulically developing along the entrance region. The following analysis shows that the fluid is thermally developing for the entire two meter SHEFRA test section and hydraulically developing for a portion of it during laminar flow. The thermal entrance region is predicted to be three meters. Exact analytical solutions for the Nusselt number in the thermal and combined entrance regions are presented. For turbulent flow, the entrance regions are significantly shorter. Correlations for steady state turbulent flow for developed flow are chosen and presented.

Thermal Development Transition

For fluids with Pr $\gg 1$, scale analysis can be used to show that the flow in the test-section can be assumed to be thermally developing. The following analysis was mostly adapted from chapters of Bejan's *Convection Heat Transfer* [5].

The steady-state conservation of mass, momentum, and energy at every point in a two-dimensional, incompressible, flow field with constant properties can be described by the mathematical equations 3.1, 3.2, 3.3, and 3.4, respectively.

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{3.1}$$

$$u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial P}{\partial x} + v\left(\frac{\partial^2 u}{\partial^2 x} + \frac{\partial^2 u}{\partial^2 y}\right) \tag{3.2}$$

$$u\frac{\partial v}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial P}{\partial y} + v\left(\frac{\partial^2 v}{\partial^2 x} + \frac{\partial^2 v}{\partial^2 y}\right) \tag{3.3}$$

$$u\frac{\partial T}{\partial x} + v\frac{\partial T}{\partial y} = \alpha\left(\frac{\partial^2 T}{\partial^2 x} + \frac{\partial^2 T}{\partial^2 y}\right) \tag{3.4}$$

The boundary layer equation for energy, neglecting thermal diffusion in the x direction, can be written as

$$u\frac{\partial T}{\partial x} + v\frac{\partial T}{\partial y} = \alpha\frac{\partial^2 T}{\partial^2 y} \tag{3.5}$$

Let δ_T be the order of magnitude of the distance in which T changes from T_0 at the wall to T_∞ in the free stream. For the boundary region of height δ_T and length L, we determine the following scales for x, y, and u.

$$x \sim L, \qquad y \sim \delta_T, \qquad u \sim U_\infty \tag{3.6}$$

where U_∞ is the flow velocity in the free stream. We define $\Delta T = T_0 - T_\infty$ to represent the temperature difference in the thermal boundary region as defined. The free stream is the flow region outside of the boundary layer region that is no longer affected by solid obstruction and heating effects.

Substituting the order of magnitude scale terms into the boundary layer equation 3.5, we can show a balance between conduction from the wall into the stream and convection parallel to the wall.

$$u\frac{\Delta T}{L}, v\frac{\Delta T}{\delta_T} \sim \alpha\frac{\Delta T}{\delta_T^2} \tag{3.7}$$

We can estimate δ_t in the limit of a thick thermal boundary layer, where $\delta_t \gg \delta$ and where δ is the velocity boundary layer defined as the order of magnitude distance between

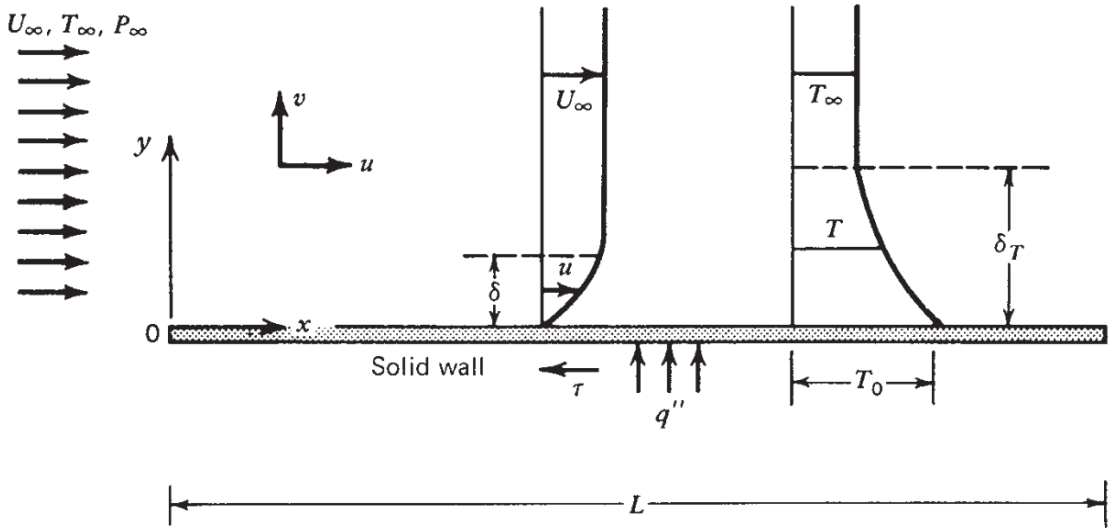


Figure 3.2: Velocity and temeprature boundary layers near a plate parallel to a uniform flow. *Convection Heat Transfer.* [5]

$u = 0$ at the wall and $u = U_\infty$ in the free stream. In this limit, the u scale outside the velocity boundary layer is U_∞. From the mass continuity equation 3.1 we can determine that the v scale goes by

$$v = \delta \frac{U_\infty}{L} \tag{3.8}$$

Using this, we can rewrite the second term of the energy balance 3.5 as

$$v \frac{\Delta T}{\delta_T} \sim U_\infty \frac{\Delta T}{L} \frac{\delta}{\delta_T} \tag{3.9}$$

As δ/δ_T is small, the first term of 3.5 dominates the convection component (the entire left side) of the energy balance.

$$U_\infty \frac{\Delta T}{L} \sim \frac{\alpha \Delta T}{\delta_T^2} \tag{3.10}$$

Rearranging we get

$$\frac{\delta_T}{L} \sim \mathrm{Pe}^{-1/2} \sim \mathrm{Pr}^{-1/2} \mathrm{Re_L}^{-1/2} \tag{3.11}$$

where the Peclét number is defined as $\mathrm{Pe} = U_\infty L/\alpha$. Extending this result to apply for internal duct flow instead of external flow on a flat plate,

$$\delta_T(x) \sim x \mathrm{Pr}^{-1/2} \mathrm{Re_L}^{-1/2} \tag{3.12}$$

At the transition between thermally developing and developed flow, $x \sim X_T$ and $\delta_T \sim D_h$. Substituting into Equation we obtain,

$$X_T \mathrm{Pr}^{-1/2} \mathrm{Re}_{\mathrm{X_T}}^{-1/2} \sim D_h \tag{3.13}$$

From which we get the dimensionless group governing the transition in thermal development:

$$\left(\frac{X_T / D_h}{\mathrm{Pr}\ \mathrm{Re}_{\mathrm{D_h}}} \right)^{1/2} \sim 1 \tag{3.14}$$

Dividing by the dimensionless group that governs the hydrodynamic development transition,

$$\left(\frac{X / D_h}{\mathrm{Re}_{\mathrm{D_h}}} \right)^{1/2} \sim 1 \tag{3.15}$$

we obtain the scaling,

$$\frac{X_T}{X} \sim \mathrm{Pr} \tag{3.16}$$

This scaling can be shown to be valid for all ranges of Pr for internal flow in a duct, which is not the case for external flow on a flat plate.

The Prandtl number range of interest for the lab surrogate fluid and prototypical molten salt is between approximately 22 and 9. This range roughly corresponds to a Flibe temperature between 600 and 800 degrees Celsius. Meaning that the thermal entry region would be 22 to 9 times the hydrodynamic entrance length. This confirms that the flow in the entire test-section is thermally developing.

Mixed Convection

This scaling does not include buoyancy forces. The temperature differences from the wall to the bulk, in some cases, are large enough that mixed convection would occur. The nondimensional group Gr/Re represents the importance of buoyancy-induced flow relative to forced flow in internal mixed convection [12]. Where,

$$\mathrm{Gr} = \frac{g\beta \left(T_w - T_b \right) D_h^3}{\nu^2} \tag{3.17}$$

The buoyancy-aided heat transfer effect scaling is plotted for a range of hypothetical experimental parameters in Figure 3.3. We find that for mixed convection effects become significant at mass flow rates below 30 kg/h with a temperature delta between the bulk fluid and wall of 30°C; using Gr/Re = 1 as a boundary. For context, the complete experimental data set discussed in Chapter 4 is recorded at a mass flow rate of ∼80 kg/h. The chosen boundary is to some degree arbitrary. To predict flow regime boundaries, it is recommended to examine flow regime maps constructed using theory and experimental data. Using a flow regime map, the Re number and the value of $\mathrm{Gr}_4 \mathrm{Pr} D_h / L$ can be used to locate flow regime

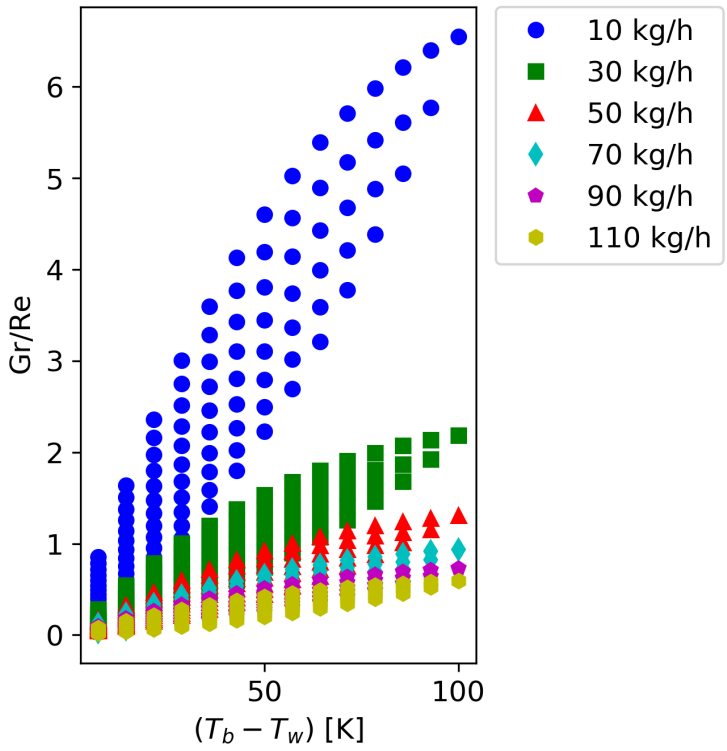


Figure 3.3: Buoyancy-aided heat transfer effect distortion for a range of possible temperature differences between the bulk fluid and channel wall. Mixed convection effects are predicted for very low mass flow rates. Temperature differences greater than 40°C are not anticipated in the experiment. A temperature difference of some value can be obtained by different T_b and T_w combinations, resulting in a range of possible film temperatures that the fluid properties are evaluated at, and hence the range of possible Gr/Re values for any given x-coordinate value. The existing SHEFRA experimental data-set covers a range of temperature differences at a mass flow rate of $\sim$ 80 kg/h.

boundaries where the mixed convection heat transfer was found to not deviate more than 10% from pure forced convection [14, 12].

$$\mathrm{Gr}_4 = \frac{g\beta\left(\bar{T}_w - \bar{T}_b\right) D_h^3}{\nu^2} \tag{3.18}$$

Here, $\bar{T}_w$ is the circumferential average wall temperature, and $\bar{T}_b$ is the fluid bulk mean temperature from $x = 0$ to x. Conservatively using a non-oscillating input, we find that over the entire length of the tube, the averaged Nu values are expected to not deviate from that of a pure convection regime. This leaves open the possibility that local mixed convection effects may be present. Future work will experimentally confirm the presence of any significant buoyancy-aided heat transfer effects by testing high values of Gr/Re in both vertical and horizontal flow configurations.

Hydrodynamic Entry Length

To determine if and where along the test-section the flow is hydrodynamically developed, the hydrodynamic entry length is calculated.

Laminar Flow Velocity Entrance Region

Using Schlichting's solution for the entrance region characteristic length [17],

$$\frac{X/D}{Re_D} \cong 0.04 \tag{3.19}$$

For the existing test-section channel (ID: "channel_1") corresponding to an inner radius of 0.06 inches the characteristic length for selected Reynolds numbers are provided in Table 3.2. A mass flow rate of 80 kg/h was included to compare how much longer the velocity entrance region would be for a hypothetical laminar flow at a Reynolds number of 7036. The motivation to include this mass flow rate is that, at the time of writing, it matched the flow rate of initial experimental data collected from SHEFRA. At a Reynolds number of 7036, the flow might be characterized as in the transitional regime and not fully turbulent. This in part motivated applying laminar flow models and associated analytical solutions for the Nusselt number to check whether the experimentally measured Nusselt numbers were comparable. Hence, determining the velocity entrance region for this flow rate was required to determine whether to compare using a hydraulically developed or developing solution for the Nusselt number. In the end, experimentally measured Nusselt numbers at 80 kg/h were found to be at least an order of magnitude higher than the estimates found in this section. These results are presented in more depth in a subsequent chapter.

The initial set of runs on SHEFRA had a mass flow rate of 80 kg/h corresponding to a Reynolds number of approximately 7000, which indicates that the flow is likely transition-turbulent. Despite this, an imaginary laminar regime is assumed for the experimental data

Re	X [m]	D [mm]	Flow rate [kg/h]	U [m/s]
440	0.05 m	3.05	5	0.05
880	0.11 m	3.05	10	0.38
1759	0.21 m	3.05	20	0.75
7036*	0.85 m	3.05	80	3.01

Table 3.2: The hydrodynamic entry lengths, X, in the SHEFRA test-section for varying mass flow rates are listed. The scale analysis shows that laminar flow is estimated to be hydraulically developed less than or equal to 0.2 m downstream of the channel inlet. *Hypothetical laminar flow for model comparison purposes

Re	X [m]	X_T [m]	D [mm]
Turbulent	0.03 m	0.03 m	3.05
Turbulent	0.06 m	0.06 m	6.10

Table 3.3: The thermal and velocity entrance lengths for a 3 and 6 mm inner diameter flow channels are less than or equal to than 6 cm.

set in order to apply laminar flow heat transfer models. Results will be overlaid with steady-state turbulent flow heat transfer models that are predicted to be within match the experimental Nusselt numbers, for a periodically-varying inlet condition, in order of magnitude.

For the initial runs at 80 kg/h, the hydrodynamic entry length for an imaginary laminar regime is 3.01 m, which is equivalent to the entire length of the SHEFRA channel. Laminar flow regimes can be achieved at lower mass flow rates with entry lengths ranging from 0 to approximately 0.25 m. The practical takeaway from this analysis is that laminar models for both hydraulically-developed and developing flow should be applied where appropriate.

Turbulent Flow Entry Length

The hydrodynamic and thermal entry region is very short compared to that of laminar flow.

$$\frac{X}{D} \cong 10 \cong \frac{X_T}{D} \tag{3.20}$$

Laminar Duct Flow Analytical Solutions

The energy equation for a steady, axisymmetric circular duct is

$$\frac{1}{\alpha}\left(u\frac{\partial T}{\partial x}+v\frac{\partial T}{\partial r}\right)=\frac{\partial^2 T}{\partial r^2}+\frac{1}{r}\frac{\partial T}{\partial r}+\frac{\partial^2 T}{\partial x^2} \tag{3.21}$$

In the case of non-negligible buoyancy forces, the axisymmetric flow assumption would not apply. Having determined the thermal development condition of the flow in the test-section, we proceed to the solve for the steady state solution analytically for the case of hydrodynamically developing and fully developed flow.

Hydrodynamically Fully Developed

For fully developed velocity regions, $v = 0$ and $u = u(r)$. Equation 3.21 becomes

$$\frac{u(r)}{\alpha}\frac{\partial T}{\partial x}=\frac{\partial^2 T}{\partial r^2}+\frac{1}{r}\frac{\partial T}{\partial r}+\frac{\partial^2 T}{\partial x^2} \tag{3.22}$$

The velocity distribution is parabolic, subject to a no-slip condition at the wall, $u(r_0) = 0$.

The convection and conduction energy flow scales can be extracted as follows.

By applying an energy balance over a control volume of length dx in a fully developed cylindrical duct it can be shown that

$$\frac{dT_m}{dx}=\frac{2}{r_0}\frac{q''}{\rho c_p U} \tag{3.23}$$

where U is the average axial velocity. Substituting this relation for the longitudinal temperature variation of the stream into the energy equation we obtain the convection scale.

$$\text{Convection}: \frac{U}{\alpha}\left(\frac{q''}{D\rho c_p U}\right)$$

$$\text{Radial conduction}: \frac{\Delta T}{D^2}, \quad \text{Longitudinal conduction}: \frac{1}{x}\left(\frac{q''}{D\rho c_p U}\right)$$

Multiplying by $D^2/\Delta T$ and applying the definition of the heat transfer coefficient $h = q''/\Delta T$ we arrive at

$$\text{Convection}: \frac{hD}{k}, \quad \text{Radial conduction}: 1, \quad \text{Longitudinal conduction}: \left(\frac{hD}{k}\right)^2\left(\frac{\alpha}{UD}\right)^2$$

The Peclét number appears in the longitudinal conduction scale. For the limit of high Pe_D, longitudinal conduction can be safely omitted to arrive at a further simplified energy equation.

$$Pe_D=\frac{UD}{\alpha}\gg 1 \tag{3.24}$$

A representative Pe_D number of 31,300 for the SHEFRA system is calculated for the experimental variables presented in Table 3.4.

$\mathrm{Pe_D}$	α [m^2/s]	D [mm]	Flow rate [kg/h]	U [m/s]
31,300	7.047×10^{-8}	3.05	20	0.75

Table 3.4: A representative Peclét number of 31,300 was calculated for the SHEFRA test section in the laminar flow regime corresponding to the thermal diffusivity of Dowtherm A at 80°C

Returning to objective of this section, which is to present an analytical model for steady-state thermal developing but hydraulically developed flow, Equation 3.25 is solved for the conditions of a constant and (radially) uniform wall temperature or constant uniform heat flux, imposing symmetry about the centerline, and isothermal fluid at the inlet. The velocity distribution is parabolic, indicating Hagen-Poiseulle flow.

$$\frac{u(r)}{\alpha}\frac{\partial T}{\partial x} = \frac{\partial^2 T}{\partial r^2} + \frac{1}{r}\frac{\partial T}{\partial r} \tag{3.25}$$

This problem was first solved by Graetz, however the Lévêque solution is presented for a simplified alternative [8, 13].

$$\begin{aligned}
\mathrm{Nu}_x &= \begin{cases} 1.077 x_*^{-1/3} - 0.70 & x_* \le 0.01 \\ 3.657 + 6.874\left(10^3 x_*\right)^{-0.488} e^{-57.2 x_*} & x_* > 0.01 \end{cases} \\
\mathrm{Nu}_{0-x} &= \begin{cases} 1.615 x_*^{-1/3} - 0.70 & x_* \le 0.005 \\ 1.615 x_*^{-1/3} - 0.20 & 0.005 < x_* < 0.03 \\ 3.657 + 0.0499/x_* & x_* > 0.03 \end{cases}
\end{aligned} \tag{3.26}$$

For a constant heat flux boundary condition, the solution is,

$$\begin{aligned}
\mathrm{Nu}_{x_*} &= \begin{cases} 1.302 x_*^{-1/3} - 1.00 & x_* \le 0.00005 \\ 1.302 x_*^{-1/3} - 0.50 & 0.00005 < x_* \le 0.0015 \\ 4.364 + 8.68\left(10^3 x_*\right)^{-0.506} e^{-41 x_*} & x_* > 0.001 \end{cases} \\
\mathrm{Nu}_{0-x} &= \begin{cases} 1.953 x_*^{-1/3} & x_* \le 0.03 \\ 4.364 + 0.0722/x_* & x_* > 0.03 \end{cases}
\end{aligned} \tag{3.27}$$

The dimensionless axial position x_* is defined

$$x_* = \frac{x/D}{\mathrm{Re_D} \cdot \mathrm{Pr}} \tag{3.28}$$

At a temperature of 80°C, the Prandtl number of Dowtherm A is 13.86. The analytical Nusselt number values of SHEFRA under a range of laminar flow conditions is plotted in Figure 3.4. The Nusselt plot is accompanied by the set of Figure 3.5 which convert between dimensionless and dimensional longitudinal distance in the SHEFRA test section.

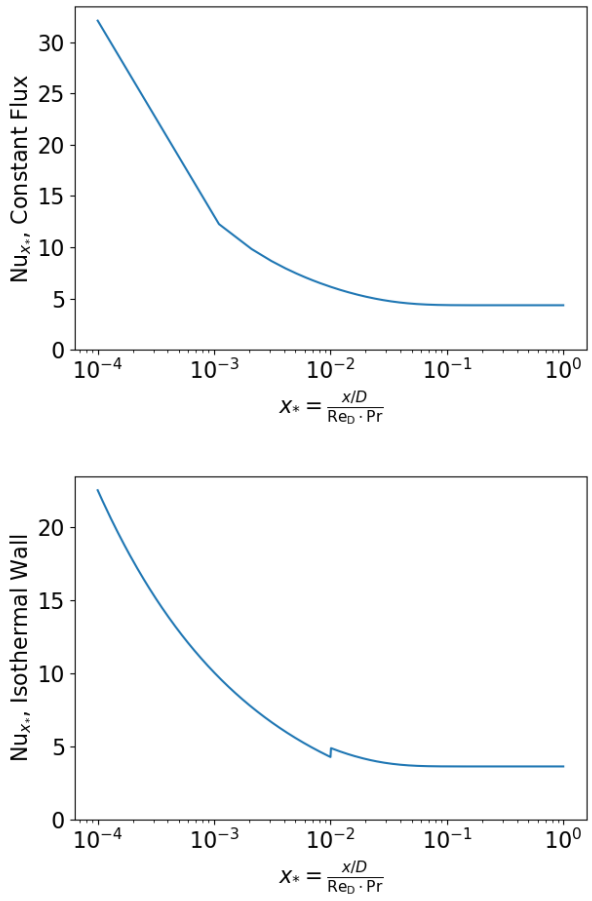


Figure 3.4: Theoretical Nusselt number values for thermally developing, hydraulically developed flow as a function of dimensionless longitudinal distance for constant heat flux and isothermal wall boundary conditions. Using the set of plots in Figure 3.5, a distance of 1 cm downstream of the test section inlet is found to correspond to $x_* = 10^{-4}$. The Nusselt numbers therefore range from around 22.5 to 4.364 and 32.1 to 3.657, from 1 cm downstream of the inlet to the channel outlet, for isothermal wall and constant heat flux boundary conditions, respectively.

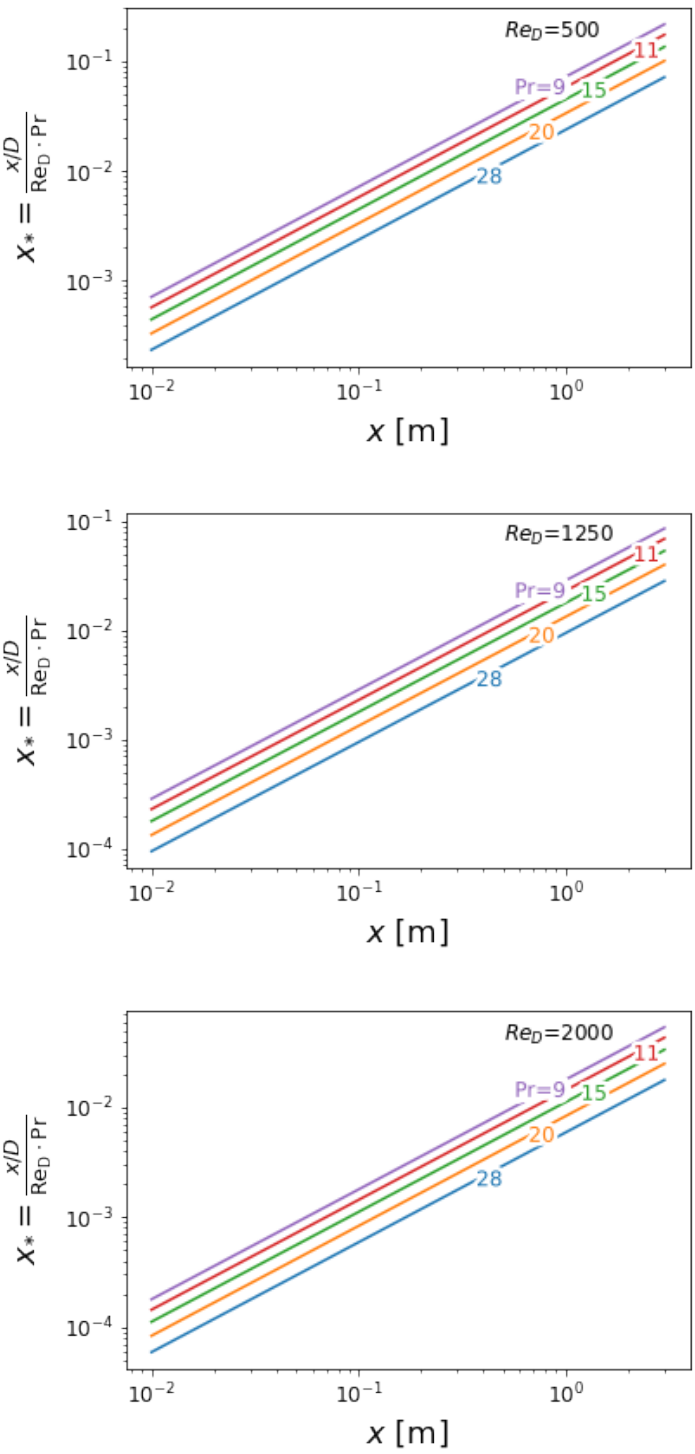


Figure 3.5: The set of figures plot the dimensionless distance along a duct as a function of other dimensional and dimensionless parameters. The top, middle and bottom plots plot x_* for $\mathrm{Re_D} = 500$, 1250, and 2000, respectively. For reference, the SHEFRA channel is 2 meters long.

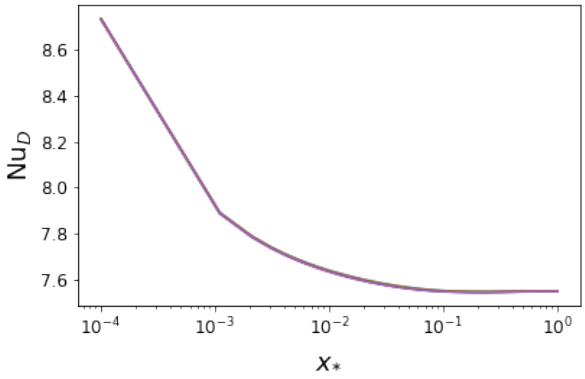


Figure 3.6: Nusselt number as a function of dimensionless length in the combined entrance region (thermally and hydraulically developing). Prandtl number was found to have little effect on the Nusselt number in the range of 27.8 to 9.2, corresponding to the Dowtherm A temperature range of 50 to 140°C.

Thermally and Hydraulically Developing Flow

According to the analysis in this section, laminar flow in the SHEFRA channel is typically both thermally and hydraulically developing. Recommended analytical correlations for the local Nusselt numbers is steady state convection in simultaneously developing flow are used to provide a useful reference point to assess experimental measurements [18]. This is not meant to replacement for a transient analytical model, which will be investigated in Section 3.2.

$$\mathrm{Nu}_x = 7.55 + \frac{0.024 x_*^{-1.14}\left(0.0179\mathrm{Pr}^{0.17}x_*^{-0.64} - 0.14\right)}{\left(1 + 0.0358\mathrm{Pr}^{0.17}x_*^{-0.64}\right)^2} \tag{3.29}$$

Turbulent Duct Flow Correlations

The Nusselt number for steady state conditions in the SHEFRA channel is predicted using the below Gnielinksi correlation [23] and plotted in Figure 3.7.

$$\begin{aligned} &\mathrm{Nu}_D = 0.012\left(\mathrm{Re}_D^{0.87} - 280\right)\mathrm{Pr}^{0.93} \\ &(1.5 \leq \mathrm{Pr} \leq 500, \quad 3 \times 10^3 \leq \mathrm{Re}_D \leq 10^6) \end{aligned} \tag{3.30}$$

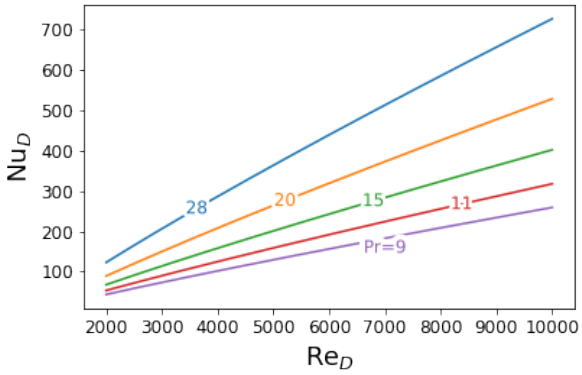


Figure 3.7: Turbulent duct flow Nusselt number predictions for SHEFRA experimental conditions as a function of Reynolds and Prandtl numbers calculated using the Gnielinksi correlation [23].

3.2 Transient Model Derivation

Introduction

Travelho and Santos obtained an analytical solution for unsteady conjugate heat transfer in a duct with a periodically varying inlet temperature by using the Laplace transform. This method circumvents the need to solve the complex eigenvalue problem and also provides a form of the analytical solution in the frequency domain. The solution was extended by the same authors to model the same problem for a circular duct with external convective heat transfer to ambient air.

The solution procedure is replicated and expanded in this section for the purpose of determining the extensibility of the solution method for modified versions of the problem, to add contextualizing commentary and to highlight mathematical approximations made. The first extension of interest is the application of a quasi-steady model for the purpose of calculating the Nusselt number at the fluid-solid boundary. This will be demonstrated in the next subsection and the results of which constitute a primary contribution to the theoretically modeling of this problem. A second objective is to experimentally validate the results as the cited authors have mentioned that this was not done. During literature review, no published work was found for the experimental validation of the Travelho and Santos solution. Thirdly, there is potential value is including this derivation from the literature to show the process of solving the problem in the frequency domain before transforming back into the time domain. Here, the complete temperature solution is then evaluated with the appropriate dimensionless arguments to obtain wall and bulk temperature and heat flux solutions. This approach could be modified such that the formulations for the temperature, heat flux and Nusselt number are in their terminal form in the frequency domain. This would

Symbols	
$A =$	amplitude
$a, a^* =$	duct radius, duct wall heat capacity $= \rho c_p a / \rho_w c_w l$
$b^*, c^* =$	dimensionless parameters defined by Equations 3.46
$c_p, c_w =$	fluid and wall specific heats at constant pressure
$h_o =$	heat transfer coefficient
$k =$	thermal conductivity of fluid
$l =$	wall thickness
$\mathrm{Nu_o} =$	outside Nusselt number $= h_o a / k$
$r, R =$	radial coordinate, dimensional radial coordinate $= r/a$
$s =$	Laplace transform parameter
$t =$	time
$T(r, z, t) =$	fluid temperature
$T_o, T =$	cycle mean temperature, reference temperature
$U =$	mean velocity
$z, Z =$	axial coordinate, dimensionless axial coordinate $\alpha z / U a^2$
$\alpha =$	thermal diffusivity of fluid
$\Delta T_o =$	amplitude of inlet oscillations
$\theta(R, Z, \tau) =$	dimensionless reference temperature $= (T(r, z, t) - T_o)\Delta T_o$
$\theta_\infty =$	dimensionless reference temperature $(T_\infty - T_o)(1 + i)/\Delta T_o$
$\rho, \rho_w =$	fluid density, wall density
$\tau =$	dimensionless time $= \alpha t / a^2$
$\psi(Z) =$	dimensional periodic part of ψ
$\omega =$	frequency of oscillations
$\Omega =$	dimensionless frequency of oscillations $= \omega a^2 / \alpha$
$\sim =$	Laplace transform

Table 3.5: Space-dependent transient model symbols.

allow for opportunities to employ frequency domain analysis methods on the theoretical and experimental systems. For example, parameters of an analytical transfer function that is predictive of the Nusselt number can be fitted to experimental data from the same frequency response tests currently being conducted for SHEFRA. The effect would be that the Nusselt number could be estimated with higher fidelity having utilized both magnitudes and phase changes of the dynamic system of oscillating wall and bulk fluid temperatures. This is in contrast to using only the absolute temperature magnitudes in the case of a conventional time-domain approach. Finally, the solution presented in this chapter applies to laminar fluid flow. Replicating the derivation process will hopefully facilitate future work to extend the solution for turbulent flows.

Problem Definition

The problem solved is for laminar forced convection with unsteady conjugate heat transfer in the thermal entrance region of a circular duct with convection from the ambient and periodically varying inlet temperature. The channel wall heat capacity is accounted for in the analysis and a slug flow model is assumed. At the time of the paper's writing, the Laplace Transform technique was used as means to avoid a complex Sturm-Liouville problem. The byproduct of this approach is the determination of amplitudes and phase lags for the wall and fluid bulk temperatures, and wall heat flux as a function of physical parameters. After the solution is transformed back into the time-domain, the phase lag information is retained and the solutions for temperature, and heat flux are in the form of complex numbers.

Equation 3.32 is a dimensionless form of Equation 3.25. Axial conduction and viscous dissipation effects are assumed to be negligible. Negligible axial conduction for the SHEFRA test-section was determined in Section 3.1. Slug flow is assumed.

$$T_m = \frac{1}{A}\int_A T dA \tag{3.31}$$

$$\frac{\partial\theta(R,Z,\tau)}{\partial\tau} + \frac{\partial\theta(R,Z,\tau)}{\partial Z} = \frac{1}{R}\frac{\partial}{\partial R}\left[R\frac{\partial\theta(R,Z,\tau)}{\partial R}\right] \tag{3.32}$$

The slug flow approximation is introduced by assuming a uniform velocity distribution $u(r) = U$. Eliminating the r dependence simplifies the analytical solution. The velocity term in the fluid energy equation is found in the dimensionless axial coordinate, $Z = \alpha z/Ua^2$. Later in the derivation, the slug flow approximation simplifies the expression for obtaining the bulk fluid temperature from the radially-dependent fluid temperature solution. For constant properties and uniform flow velocity, the bulk temperature is the mean fluid temperature averaged over the flow area, as defined in Equation 3.31. The dimensionless form of this definition appears in Equation 3.60. The approximation refers to the flow of a solid material or a fluid with an extremely small Prandtl number. We can expect that the quasi-steady model will overpredict Nu values compared to the physical system [25]. Despite this distortion, the results and conclusions obtained remain tractable.

The inlet bulk fluid temperature follows a periodic oscillation mathematically represented as a complex exponential.

$$\theta(R,0,\tau) = e^{i\Omega\tau} \tag{3.33}$$

The problem is also subject to the following boundary condition due to symmetry:

$$\left.\frac{\partial\theta(R,Z,\tau)}{\partial R}\right|_{R=0} = 0 \tag{3.34}$$

The convection to ambient air is governed by a constant heat coefficient at the wall boundary. The channel is modeled with a thin wall, and therefore $R = 1$ corresponds to the boundary for both heat flux to the fluid, and to the ambient air on the other side. An energy balance

on the wall produces the following boundary condition.

$$-\left.\frac{\partial\theta(R,Z,\tau)}{\partial R}\right|_{R=1} = \frac{1}{a^*}\frac{\partial\theta(1,Z,\tau)}{\partial\tau} + \mathrm{Nu}_o\left[\theta(1,Z,\tau)-\theta_\infty\right] \tag{3.35}$$

Here a_* represents the ratio of the effect of duct wall capacitance to heat transfer.

Problem Redefined As Sum of Periodic and Permanent Components

Next, the problem is split into a periodic and permanent component.

$$\theta(R,Z,\tau) = \xi(R,Z) + \gamma(R,Z,\tau) \tag{3.36}$$

Permanent Problem Statement

The problem's permanent component $\xi(R,Z)$ is now defined as

$$\frac{\partial\xi(R,Z)}{\partial Z} = \frac{1}{R}\frac{\partial}{\partial R}\left[R\frac{\partial\xi(R,Z)}{\partial R}\right] \tag{3.37}$$

with boundary conditions

$$\xi(R,0) = 0 \quad , \quad \left.\frac{\partial\xi(R,Z)}{\partial R}\right|_{R=0} = 0 \tag{3.38}$$

and

$$\left.\frac{\partial\xi(R,Z)}{\partial R}\right|_{R=1} + \mathrm{Nu}_o\left[\xi(1,Z)-\theta_\infty\right] = 0 \tag{3.39}$$

Periodic Problem Statement

Similarly, the periodic component $\gamma(R,Z,\tau)$ is defined

$$\frac{\partial\gamma(R,Z,\tau)}{\partial\tau} + \frac{\partial\gamma(R,Z,\tau)}{\partial Z} = \frac{1}{R}\frac{\partial}{\partial R}\left[R\frac{\partial\gamma(R,Z,\tau)}{\partial R}\right] \tag{3.40}$$

with boundary conditions

$$\gamma(R,0,\tau) = e^{i\Omega\tau} \quad \text{and} \quad \left.\frac{\partial\gamma(R,Z,\tau)}{\partial R}\right|_{R=0} = 0 \tag{3.41}$$

and

$$\left.\frac{\partial\gamma(R,Z,\tau)}{\partial R}\right|_{R=1} + \frac{1}{a^*}\frac{\partial\gamma(1,Z,\tau)}{\partial\tau} + \mathrm{Nu}_o\gamma(1,Z,\tau) = 0 \tag{3.42}$$

Periodic Problem Redefined

The periodic temperature function $\gamma(R, Z, \tau)$ can be assumed to be of the form $\gamma(R, Z, \tau) = \psi(R, Z)e^{i\Omega(\tau - Z)}$. Replacing this term in Equation 3.40 and its associated boundary conditions we get

$$\frac{\partial \psi(R,Z)}{\partial Z} = \frac{1}{R}\frac{\partial}{\partial R}\left[R\frac{\partial \psi(R,Z)}{\partial R}\right] \quad 0 \leq R \leq 1, Z \geq 0 \tag{3.43}$$

$$\psi(R,0) = 1 \quad \left.\frac{\partial \psi(R,Z)}{\partial R}\right|_{R=0} = 0 \tag{3.44}$$

and

$$\left.\frac{\partial \psi(R,Z)}{\partial R}\right|_{R=1} + c * \psi(1,Z) = 0 \tag{3.45}$$

where the dimensionless parameters are

$$c^* = \mathrm{Nu}_o + ib^*, \quad b^* = \frac{\Omega}{a^*} = \frac{\omega a \rho_w c_w l}{k} \tag{3.46}$$

The dimensionless parameter b^* is the dimensionless frequency multiplied by the ratio wall thermal capacity to fluid thermal conductivity. This becomes an important parameter for the dynamic response of the system, as seen later in this chapter. b^* can be found in c^*, a term that includes the effect of ambient convection, and present in the solutions to the periodic problem solutions presented later in this section.

Permanent Problem Solution Method

Taking the Laplace transform of Equation 3.37 with respect to Z obtains

$$s\tilde{\xi}(R,s) = \frac{1}{R}\frac{\partial}{\partial R}\left[R\frac{\partial \tilde{\xi}(R,s)}{\partial R}\right] \tag{3.47}$$

$$\left.\frac{\partial \tilde{\xi}(R,s)}{\partial R}\right|_{R=0} = 0 \tag{3.48}$$

$$\left.\frac{\partial \tilde{\xi}(R,s)}{\partial R}\right|_{R=1} + \mathrm{Nu}_o\left[\tilde{\xi}(1,s) - \frac{\theta_\infty}{s}\right] = 0 \tag{3.49}$$

where

$$\tilde{\xi}(R,s) = \lim_{Z \to s} \xi(R,Z) = \int_0^\infty \xi(R,Z)e^{-sZ}dZ \tag{3.50}$$

The solution to this problem in the Laplace domain is

$$\tilde{\xi}(R,s) = \frac{\mathrm{Nu}_o \theta_\infty}{s} \frac{I_0(\sqrt{s}R)}{\sqrt{s}I_1(\sqrt{s}) + \mathrm{Nu}_o I_0(\sqrt{s})} \tag{3.51}$$

where I_0 and I_1 are the modified Bessel functions of the first kind.

Permanent Wall Dimensionless Temperature

The permanent wall temperature is defined as

$$\xi_w(Z) \equiv \xi(1, Z) \tag{3.52}$$

where $\xi(1, Z)$ is obtained by the inverse transform of $\tilde{\xi}(1, s)$, per the transform definition in Equation 3.50. The inverse transform solution (Equation 3.55) is found for $R = 1$, and for small Z, and in turn small values of s, which is applicable for the thermal entrance region (Refer to [22], [16] and [1]). Setting $R = 1$ in the Laplace solution to the permanent problem in Equation 3.51 obtains

$$\tilde{\xi}(1, s) = \frac{\mathrm{Nu}_o \theta_\infty}{s} \frac{I_0(\sqrt{s})}{\sqrt{s} I_1(\sqrt{s}) + \mathrm{Nu}_o I_0(\sqrt{s})} \tag{3.53}$$

The asymptotic expansion for large arguments for the modified Bessel functions of the first kind is

$$I_\nu(z) \sim \frac{e^z}{\sqrt{2\pi z}} \left\{ 1 - \frac{\mu - 1}{8z} + \dots \right\} \quad \left(|\arg z| < \frac{\pi}{2} \right) \tag{3.54}$$

where $\mu = 4\nu^2$ Using this expansion, the inverse Laplace solution is

$$\xi_w(Z) \equiv \xi(1, Z) = \theta_\infty \left\{ 1 - \frac{1}{(\eta_1 - \eta_2)} \left[\eta_1 e^{\eta_1^2 Z} \operatorname{erfc}\left(-\eta_1 \sqrt{Z} \right) \ -\eta_2 e^{\eta_2^2 z} \operatorname{erfc}\left(-\eta_2 \sqrt{Z} \right) \right] \right\} \tag{3.55}$$

where η_1 and η_2 are given by $\eta_2^1 = -\mathrm{Nu}_o/2 \pm [\mathrm{Nu}_o/2\,(\mathrm{Nu}_o/\ 2 - 1)]^{1/2}$.

This solution expression is indeterminate for $\mathrm{Nu}_o = 2$ due to the two-term approximation. This is remedied by applying L'Hospital's rule to obtain

$$\xi_w(Z) = \theta_\infty \left[1 + 2\sqrt{\tfrac{Z}{\pi}} - (1 + 2Z) e^Z \operatorname{erfc}(\sqrt{Z}) \right] \quad \text{for } \mathrm{Nu}_o = 2 \tag{3.56}$$

Permanent Wall Dimensionless Heat Flux

The wall heat flux, $\xi_h(Z)$, is given by the following, as per Equation 3.49.

$$\xi_h = -\partial \xi(R, Z) / \left. \partial R \right|_{R=1} \tag{3.57}$$

Applying this definition to the Laplace transform of the permanent solution in Equation 3.51 and obtaining the inverse transform solution results in

$$\xi_h(Z) = -\frac{\mathrm{Nu}_o \theta_\infty}{(\eta_1 - \eta_2)} \left[\eta_1 e^{\eta_1^2 z} \operatorname{erfc}\left(-\eta_1 \sqrt{Z} \right) \ -\eta_2 e^{\eta_2^2 Z} \operatorname{erfc}\left(-\eta_2 \sqrt{Z} \right) \right] \tag{3.58}$$

for $\mathrm{Nu}_o \neq 2$

$$\xi_h(Z) = 2\theta_\infty \left[2\sqrt{\frac{Z}{\pi}} - (1 + 2Z) e^Z \operatorname{erfc}(\sqrt{Z}) \right] \tag{3.59}$$

for $\mathrm{Nu}_o = 2$

Permanent Fluid Bulk Dimensionless Temperature

Starting with the definition of fluid bulk temperature

$$\xi_b(Z) = 2\int_0^1 R\xi(R,Z)dR \tag{3.60}$$

The Laplace transform is applied, and utilizing the previously obtained $\tilde{\xi}(R,Z)$ from Equation 3.51, the expression is integrated to get

$$\tilde{\xi}_b(s) = \frac{2\theta_\infty \mathrm{Nu}_o}{s^{3/2}} \frac{I_1(\sqrt{s})}{\sqrt{s}I_1(\sqrt{s}) + \mathrm{Nu}_o I_o(\sqrt{s})} \tag{3.61}$$

In a similar procedure to the permanent wall temperature solution, the inverse transform of Equation 3.61 is found. The result is, for $\mathrm{Nu}_o \neq 2$

$$\begin{aligned}\xi_b(Z) = 2\theta_\infty \Big\{2 + \tfrac{\mathrm{Nu}_o}{(\eta_1-\eta_2)} \Big[\tfrac{1}{\eta_1} e^{\eta_1^2 z} \operatorname{erfc}\left(-\eta_1\sqrt{Z}\right) \\ -\tfrac{1}{\eta_2} e^{\eta_2^2 z} \operatorname{erfc}\left(-\eta_2\sqrt{Z}\right)\Big]\Big\}\end{aligned} \tag{3.62}$$

and

$$\xi_b(Z) = 4\theta_\infty \left[1 - 2\sqrt{\frac{Z}{\pi}} + 2Ze^Z \operatorname{erfc}(\sqrt{Z})\right] \tag{3.63}$$

for $\mathrm{Nu}_o = 2$

Periodic Problem Solution

The authors describe that a similar method is used to solve the periodic dimensionless fluid bulk temperature, wall temperature and wall heat flux [22]. The dimensionless periodic wall temperature, wall heat flux and and bulk fluid temperature solutions are given by Equations 3.64, 3.65, and 3.66, respectively.

Periodic Wall Dimensionless Temperature

The periodic wall temperature defined at radial coordinate location $R = 1$ in the periodic solution $\psi(R,Z)$.

$$\begin{aligned}\psi_w(Z) \equiv \psi(1,Z) = \frac{1}{\lambda_1 - \lambda_2}\Big(\lambda_1 e^{\lambda_1^2 Z}\operatorname{erfc}(-\lambda_1\sqrt{Z}) \\ -\lambda_1 e^{\lambda_1^2 Z}\operatorname{erfc}(-\lambda_2\sqrt{Z})\Big)\end{aligned} \tag{3.64}$$

Periodic Dimensionless Heat Flux

The periodic dimensionless heat flux is defined at the wall ($R = 1$) by

$$\psi_h(Z) \equiv -\left.\frac{\partial\psi(R,Z)}{\partial R}\right|_{R=1} = c^*\psi(1,Z) \tag{3.65}$$

Periodic Fluid Dimensionless Temperature

$$\psi_b(Z) = -3 - \frac{2c^*}{\lambda_1 - \lambda_2}\Big(\frac{1}{\lambda_1}e^{\lambda_1^2 Z}\mathrm{erfc}(-\lambda_1\sqrt{Z}) - \frac{1}{\lambda_2}e^{\lambda_1^2 Z}\mathrm{erfc}(-\lambda_2\sqrt{Z})\Big) \tag{3.66}$$

where $\lambda_2^1 = -c^*/2 \pm [c^*/2\,(c^*/2-1)]^{1/2}$.

3.3 Quasi-steady Model

Application of Quasi-steady Model

An quasi-steady model to predict the local heat transfer coefficient in the SHEFRA test section is constructed using the Travelho and Santos solution derived in the previous section for unsteady conjugate heat transfer in a circular duct with periodically varying inlet temperature [22]. The approach taken is to work backwards from the permanent and periodic solutions obtained at the end of the previous section to obtain the complete dimensionless fluid bulk temperature, wall temperature, and wall heat flux solutions. These terms are then used to formulate an expression that equals to the Nusselt number.

As described in the previous section, the dimensionless temperature solution composed of a permanent and periodic component.

$$\theta(R, Z, \tau) = \xi(R, Z) + \gamma(R, Z, \tau) \tag{3.67}$$

The solution of dimensionless periodic temperature assumes the form

$$\gamma(R, Z, \tau) = \psi(R, Z)e^{i\Omega(\tau - Z)} \tag{3.68}$$

The dimensionless permanent wall temperature, wall heat flux and and bulk fluid temperature solutions are given by Equations 3.55, 3.58, and 3.62, respectively. The dimensionless periodic wall temperature, wall heat flux and and bulk fluid temperature solutions are given by Equations 3.64, 3.65, and 3.66, respectively. An expression for the heat transfer coefficient at the wall is formulated. The quasi-steady heat transfer coefficient is defined in relation to the wall heat flux as follows

$$q'' = h(T_b(z) - T_w(z)) \tag{3.69}$$

In dimensionless terms this becomes

$$-\left.\frac{\partial\theta(R, Z, \tau)}{\partial R}\right|_{R=1} = \mathrm{Nu}[\theta_b(Z) - \theta_w(Z)] \tag{3.70}$$

The permanent and peridoic contributions to the dimensionless heat flux are defined as

$$-\left.\frac{\partial\theta(R, Z, \tau)}{\partial R}\right|_{R=1} \equiv \eta_h(Z) + \lambda_h(Z) \tag{3.71}$$

Substituting back into Equation X,

$$\eta_h(Z) + \lambda_h(Z) = \mathrm{Nu}[\theta_b(Z) - \theta_w(Z)] \tag{3.72}$$

$$\eta_h(Z) \equiv \xi_h(Z) = \left.\frac{\partial \xi(R,Z)}{\partial R}\right|_{R=1} \tag{3.73}$$

Recalling that the temperature solution was split into the following terms

$$\theta(R,Z,\tau) = \xi(R,Z) + \gamma(R,Z,\tau) \tag{3.74}$$

$$\left.\frac{\partial \theta(R,Z,\tau)}{\partial R}\right|_{R=1} = \left.\frac{\partial \xi(R,Z)}{\partial R}\right|_{R=1} + \left.\frac{\partial \gamma(R,Z,\tau)}{\partial R}\right|_{R=1} \tag{3.75}$$

Recalling that the periodic temperature solution was assumed to have the form in Equation 3.68

$$\left.\frac{\partial \gamma(R,Z,\tau)}{\partial R}\right|_{R=1} = \lambda_h = \left.\frac{\partial \psi(R,Z)}{\partial R}\right|_{R=1} e^{i\Omega(\tau - Z)} \tag{3.76}$$

Substituting these terms into the complete expression for the wall heat flux we obtain

$$-\left.\frac{\partial \theta(R,Z,\tau)}{\partial R}\right|_{R=1} \equiv -\left.\frac{\partial \xi(R,Z)}{\partial R}\right|_{R=1} - \left.\frac{\partial \psi(R,Z)}{\partial R}\right|_{R=1} e^{i\Omega(\tau - Z)} \tag{3.77}$$

The partial derivative on the right side of the equation are replaced with previously defined notation for the respective permanent and periodic wall heat flux solutions.

$$-\left.\frac{\partial \theta(R,Z,\tau)}{\partial R}\right|_{R=1} \equiv \xi_h(Z) + \psi_h(Z)\, e^{i\Omega(\tau - Z)} \tag{3.78}$$

Substituting the Travelho and Santos periodic and permanent wall heat flux solutions we get

$$\begin{aligned} -\left.\frac{\partial \theta(R,Z,\tau)}{\partial R}\right|_{R=1} \equiv\ & -\frac{\mathrm{Nu}_o \theta_\infty}{(\eta_1 - \eta_2)}\Big[\eta_1 e^{\eta_1^2 Z}\mathrm{erfc}(\eta_1\sqrt{Z}) \\ & -\eta_2 e^{\eta_2^2 Z}\mathrm{erfc}(\eta_2\sqrt{Z})\Big] - c^*\psi(1,Z) \end{aligned} \tag{3.79}$$

Utilizing the the periodic dimensionless wall temperature solution for $\psi(1,Z)$

$$\begin{aligned} -\left.\frac{\partial \theta(R,Z,\tau)}{\partial R}\right|_{R=1} \equiv & \\ -\frac{\mathrm{Nu}_o \theta_\infty}{(\eta_1 - \eta_2)}\Big[\eta_1 e^{\eta_1^2 Z}\mathrm{erfc}(\eta_1\sqrt{Z}) - \eta_2 e^{\eta_2^2 Z}\mathrm{erfc}(\eta_2\sqrt{Z}\Big] & \\ +\frac{c^*}{\lambda_1 - \lambda_2}\Big(\lambda_1 e^{\lambda_1^2 Z}\mathrm{erfc}(-\lambda_1\sqrt{Z}) - \lambda_1 e^{\lambda_1^2 Z}\mathrm{erfc}(-\lambda_2\sqrt{Z})\Big) & \end{aligned} \tag{3.80}$$

In addition to the final form of the wall heat flux term, the complete fluid bulk and wall temperature solutions are needed.

$$\theta_w(R, Z, \tau) = \xi_w(R, Z, \tau) + \psi_w(R, Z)\, e^{i\Omega(\tau - Z)} \tag{3.81}$$

$$\theta_b(R, Z, \tau) = \xi_b(R, Z, \tau) + \psi_b(R, Z)\, e^{i\Omega(\tau - Z)} \tag{3.82}$$

Rearranging, we can solve for the Nusselt number in terms of the derived expressions.

$$\mathrm{Nu} = \frac{-\left.\frac{\partial\theta(R,Z,\tau)}{\partial R}\right|_{R=1}}{\theta_b(Z) - \theta_w(Z)} \tag{3.83}$$

Transient Model Evaluation

The transient model provides a general theoretical basis for experiment analysis. Capabilities derived from the analysis in this section include the ability to predict if a system is in a quasi-steady state, a prerequisite for measuring valid (i.e. constant) values of Nu, and model-informed design of frequency response experiments and parameter tuning such that the system is quasi-steady. This lays the groundwork for future work which, as discussed in Section 3.2, would be be to apply the quasi-steady model in the Laplace domain to enable the use of both magnitude and phase change data from frequency response experiments to more reliably estimate constant experimental parameters such as the Nusselt number in a system transfer function.

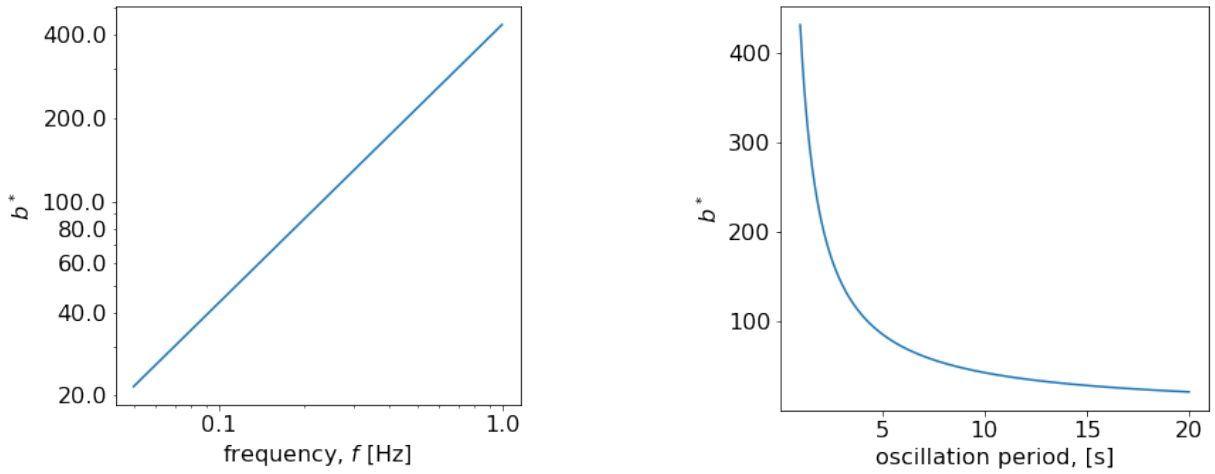
The transient model was evaluated for different combinations and ranges of the characteristic dimensionless parameters to predict the response of a range of hypothetical test-sections and experimental conditions. The effect of different dimensionless parameters on the analytically derived Nusselt number is analysed and compared to similar studies in the literature. Similar trends were observed applying a quasi-steady model on analytical solutions for temperature and heat flux for unsteady transient convection heat transfer between parallel plates [20]. To bring the dimensionless parameters into context, a scaling analysis is presented in the subsequent sections. This is followed by model results that demonstrate key trends and that also reveal potential sources of distortions causing a departure from quasi-steady state and a non-constant Nusselt number at a fixed longitudinal distance. Table 3.6 provides representative experimental parameters specific to the SHEFRA test section to accompany the scaling analysis. Table 1.1 summarizes the dimensionless group ranges that apply to all of the analytical model simulations presented in the results section of this chapter.

Symbol	Parameter	Value	Units
ρ_w	Wall density	1358	kg/m^3
k	Thermal conductivity of fluid	$f(T)$, Dowtherm A	W/(m K)
k_s	Thermal conductivity of copper	8933, Ref. [6]	W/(m K)
c_w	Specific density of wall	8933	J/(kg K)
c_p	Specific density of fluid	$f(T)$, Dowtherm A	J/(kg K)
a	Channel inner radius	0.06	inches
l	Wall thickness	0.065	in
ΔT_o	Fluid temperature amplitude	50	K
T_o	Cycle mean temperature	400	K
T_∞	Ambient air temperature	293.15	K

Table 3.6: Representative experimental parameters for the SHEFRA test section used for analytical analysis. The thermal conductivity of the copper wall was used to impose an upper limit of 0.1 to the candidate test section Biot number. The Biot analysis is shown for the actual test section in Chapter 4.

Symbol	Parameter	Value
Pr	Prandtl number	$7 < \mathrm{Pr} < 19$
Re	Reynolds number	$1253 < \mathrm{Re} < 4522$
θ_∞	Dimensionless reference temperature	$-2.33(1+i)$
a^*	Dimensionless parameter (Table 3.5)	0.48
b^*	Dimensionless parameter (Eq. 3.46)	Variable
c^*	Dimensionless parameter (Eq. 3.46)	Variable
$\mathrm{Nu_o}$	Nusselt number, ambient convection	Variable

Table 3.7: Dimensionless experimental parameters for the SHEFRA test section analytical model simulation. The Reynolds number exceeds the range of applicability for laminar flow regime due to temperature dependent fluid properties that fluctuate over a heating cycle. The simulation Re upper limit is sufficiently close to 2300 such that no significant distortion is expected to result from assuming a hypothetical laminar regime at a Reynolds number of 4522. However, future work should repeat this analysis with a mass flow rate lower than 30 kg/h for confirmation.



(a) (b)

Figure 3.8: b^* dependence on inlet oscillation frequency on a log-log scale (a) and in terms of oscillation period on a linear scale (b). b^* becomes infinitely large as the oscillation period approaches zero.

3.4 Scaling Analysis

Frequency Scaling

Recalling the definition of dimensionless parameter b^* from Equation 3.46,

$$b^* = \frac{\Omega}{a^*} = \frac{\omega a \rho_w c_w l}{k}$$

we can observe its dependence on the inlet temperature oscillation frequency in Figure 3.8. The b^* parameter represents the nondimensional frequency multiplied by the ratio between the thermal capacity of the wall and the heat transfer by conduction across the fluid. When $b^* = 0$, the wall heat capacitance can be neglected, which results in the equivalent problem for a non-participating wall.

Parameter b^* exits the linear regime and becomes exponentially large as the oscillation period drops below 5 seconds. The remaining variables of b^* are evaluated using the representative values in Table 3.6 at a fluid temperature of 100°C. The inlet temperature oscillation frequency is converted from Hz to rad/s before evaluating b^*.

Distance Scaling

The dimensionless distance Z is given by

$$Z = \alpha z / U a^2 \tag{3.84}$$

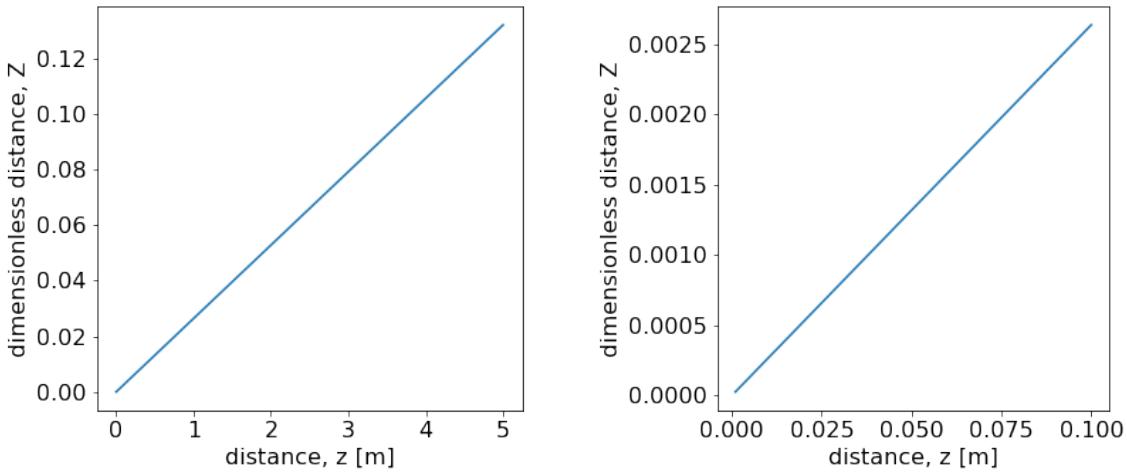


Figure 3.9: Dimensionless distance as a function of dimensional distance downstream of the SHEFRA test section inlet. A distance of 1 cm, and 3 m downstream of the inlet corresponds to Z values of 10^{-4} and 0.079, respectively. The plot on the right covers a section close to the inlet while the left plot covers a distance that goes beyond the physical channel.

The values of Z are small in the thermal entrance region of a channel, which is the case for the entire 2 meter length of the SHEFRA test section as shown in Figure 3.9.

Reference Temperature Scaling

The dimensionless reference temperature θ_∞ is the reference temperature scale for the SHEFRA system as defined in Table 3.5 as,

$$\theta_\infty = (T_\infty - T_o)(1 + i)/\Delta T_o \tag{3.85}$$

Figure 1.9 shows the effect of the mean bulk fluid temperature, T_o in addition to the inlet bulk fluid temperature amplitude, ΔT_o. The ranges for each variable were chosen to cover and extend beyond desired Prandtl scales for Dowtherm A as a surrogate fluid for Flibe and the amplitudes possible to achieve experimentally based on operational experience with SHEFRA hardware.

3.5 Analytical Solutions and Results

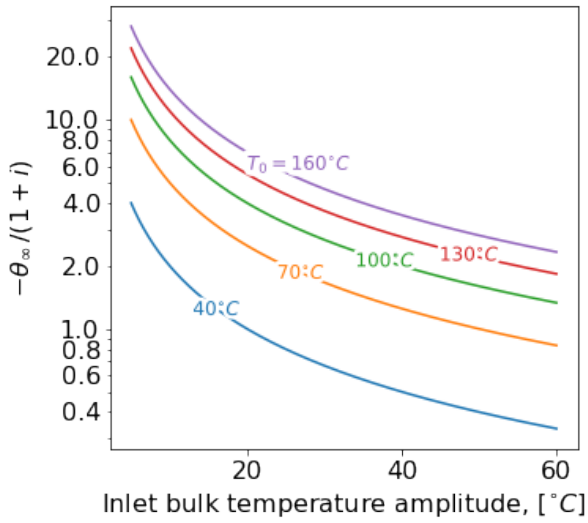


Figure 3.10: Effect of inlet bulk temperature amplitude and mean cycle bulk fluid temperature on θ_∞ on a log y-axis. A typical amplitude of $30°C$ and mean temperature of $90°C$ result in a $\theta_\infty = -2.33(1+i)$.

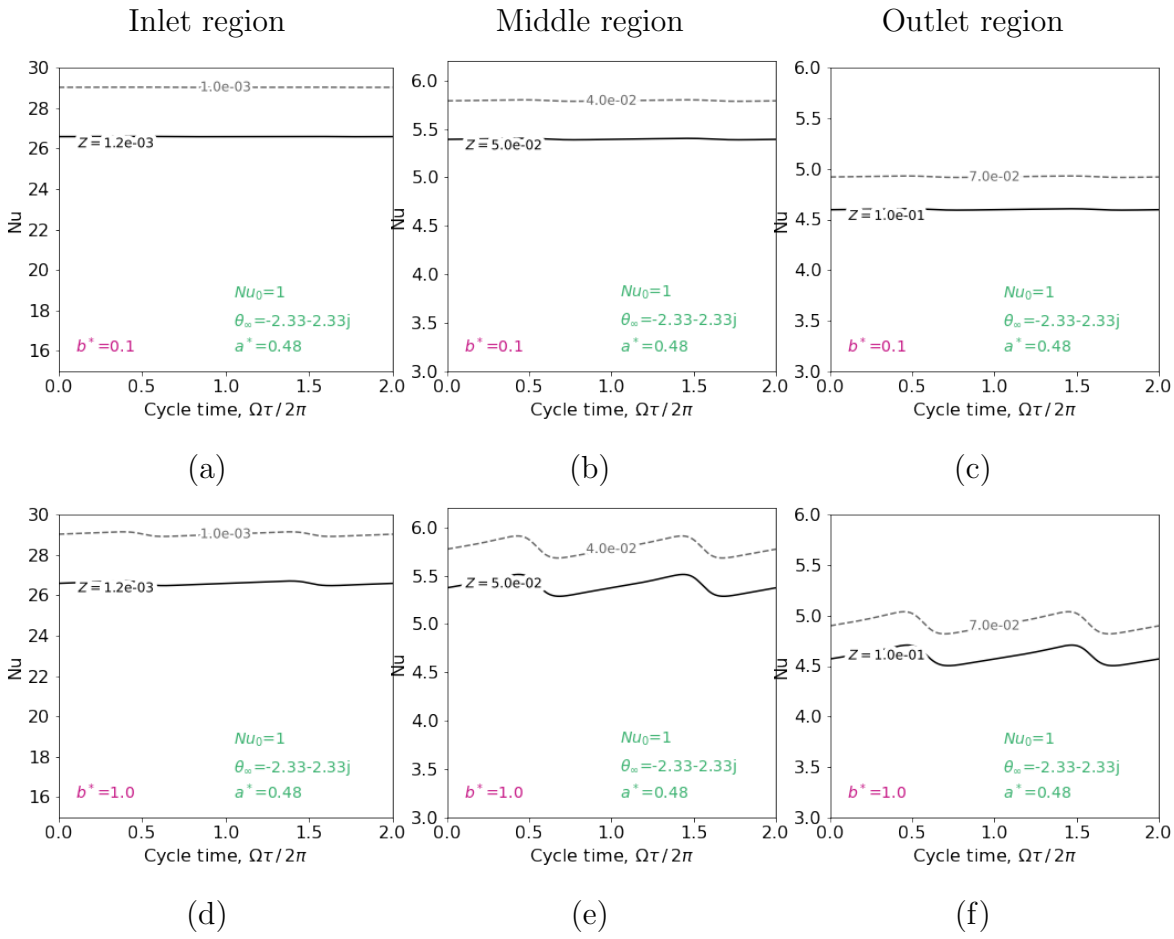


Figure 3.11: The quasi-steady model predictions for Nu(Z) for $b^* = 0.1$ are plotted in the top row for two different positions within each of the inlet (a), middle (b) and outlet (c) regions, respectively. The bottom row shows the same for $b^* = 1.0$. Nu values in the thermally developed outlet region are elevated compared to steady-state predictions, suggesting that while distances further down the channel are better approximated by the quasi-steady model, low values of b^* produce worse approximations compared to much higher values. The dimensionless parameters that are specific problem are printed on the plot area. To create this simulation, representative experimental conditions were chosen for which the Prandtl number range is between 24 and 11. The Reynolds number ranges between 1871 and 4724 for a fixed mass flow rate of 30 kg/h. In practice, the test condition that is being changed from one b^*

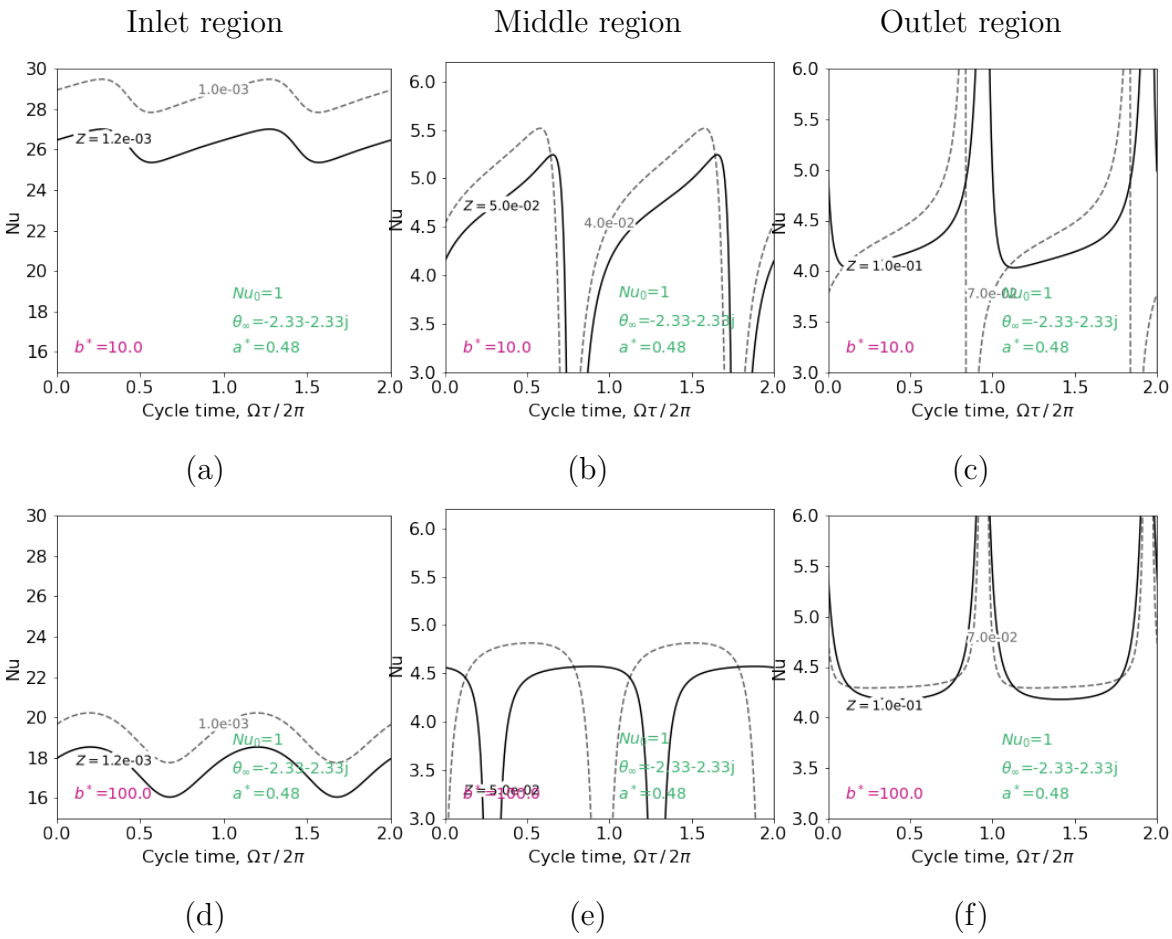


Figure 3.12: The quasi-steady model predictions for Nu(Z) for $b^* = 10$ are plotted in the top row for two different positions within each of the inlet (a), middle (b) and outlet (c) regions, respectively. The bottom row shows the same for $b^* = 100$. The highest deviations in Nu are found for intermediate values of b^*. Quasi-steady operation would be represented by a horizontal line, which can be found at higher values of b^* for portions of the cycle time, at vertical positions that more closely matches steady-state thermally developed conditions in the outlet region, despite a large distortion added from ambient convection ($\mathrm{Nu_o} = 1$.)

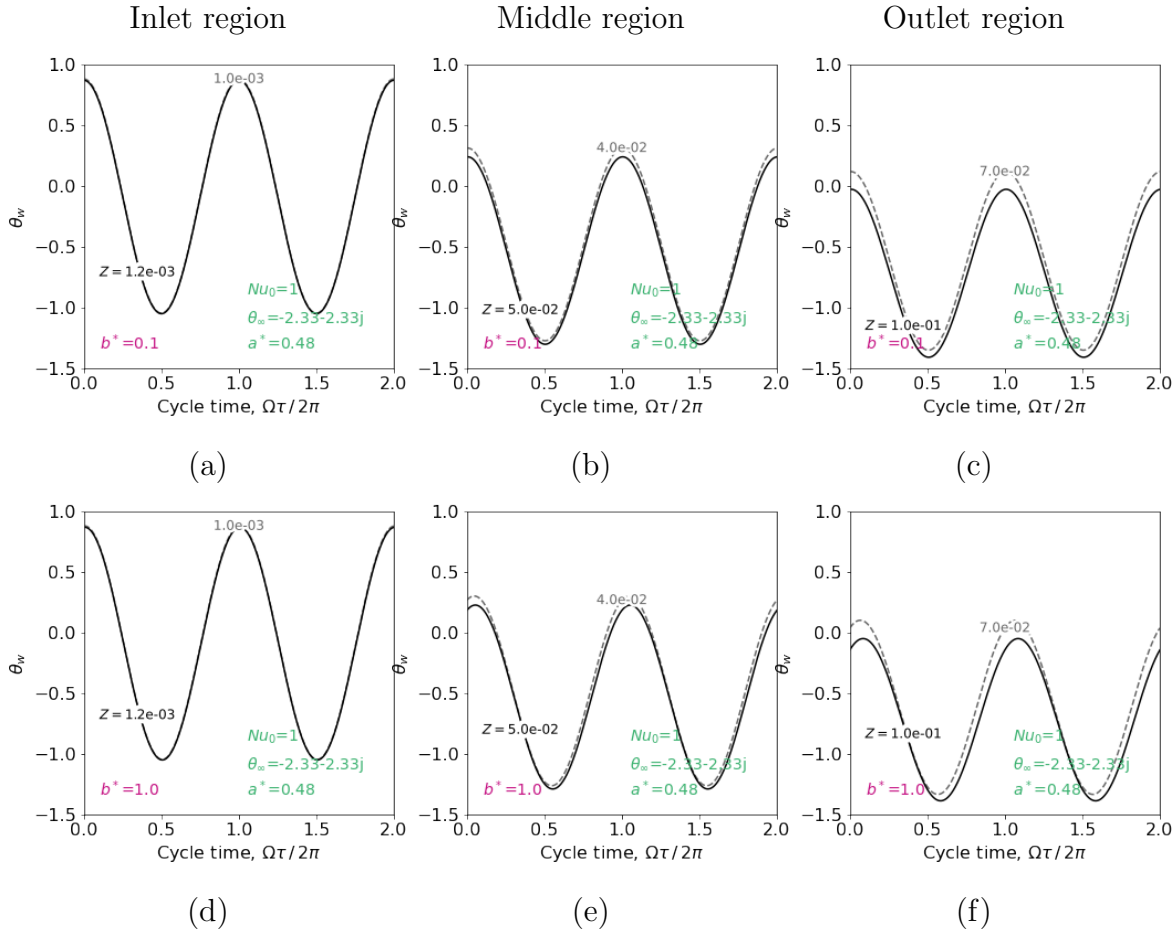


Figure 3.13: Wall temperature, $\theta_{\mathrm{w}}(Z)$ for $b^* = 0.1$ is plotted in the top row for two different positions within each of the inlet (a), middle (b) and outlet (c) test-section regions, respectively. The bottom row shows the same for $b^* = 1.0$.

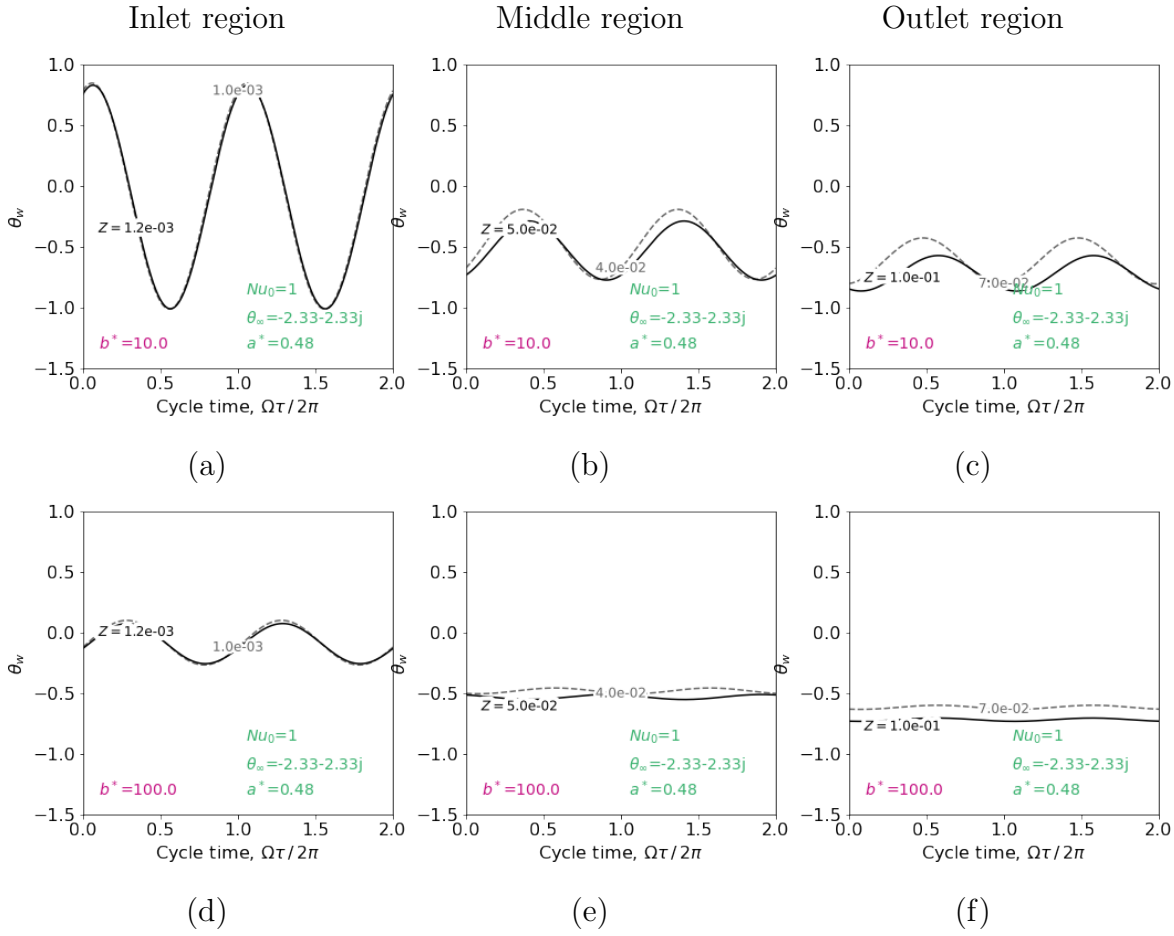


Figure 3.14: Wall temperature, $\theta_{\mathrm{w}}(Z)$ for $b^* = 10$ is plotted in the top row for two different positions within each of the inlet (a), middle (b) and outlet (c) test-section regions, respectively. The bottom row shows the same for $b^* = 100$.

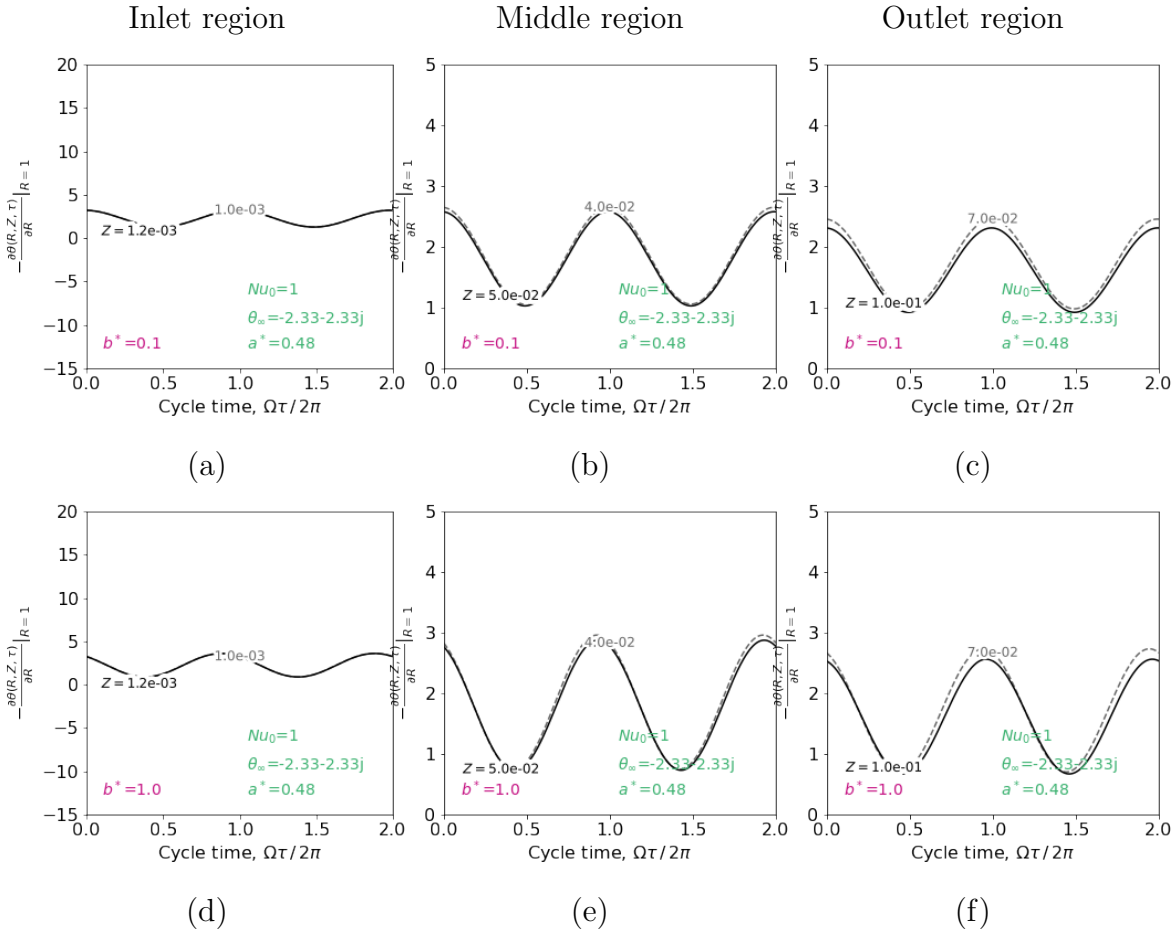


Figure 3.15: Wall heat flux for $b^* = 0.1$ is plotted in the top row for two different positions within each of the inlet (a), middle (b) and outlet (c) test-section regions, respectively. The bottom row shows the same for $b^* = 1.0$.

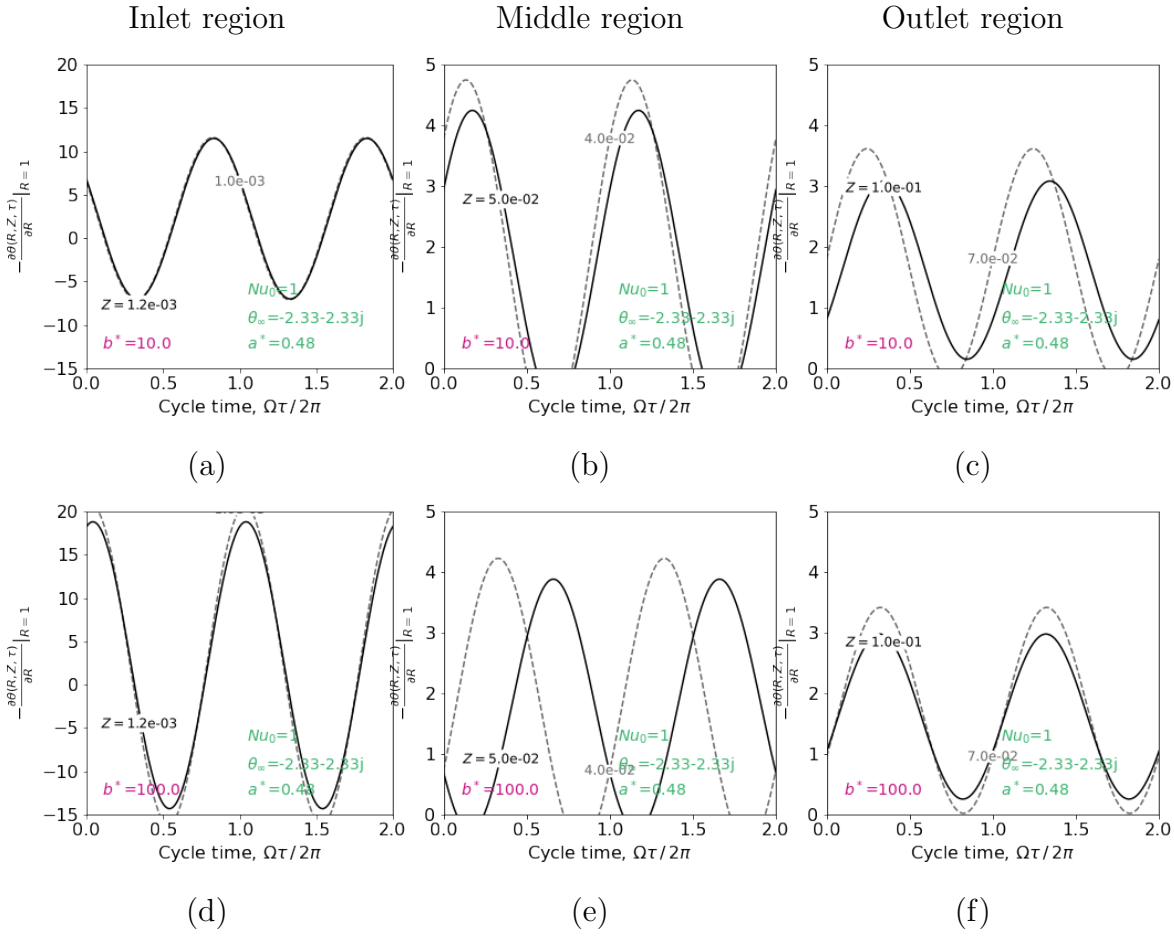


Figure 3.16: Wall heat flux for $b^* = 10$ is plotted in the top row for two different positions within each of the inlet (a), middle (b) and outlet (c) test-section regions, respectively. The bottom row shows the same for $b^* = 100$.

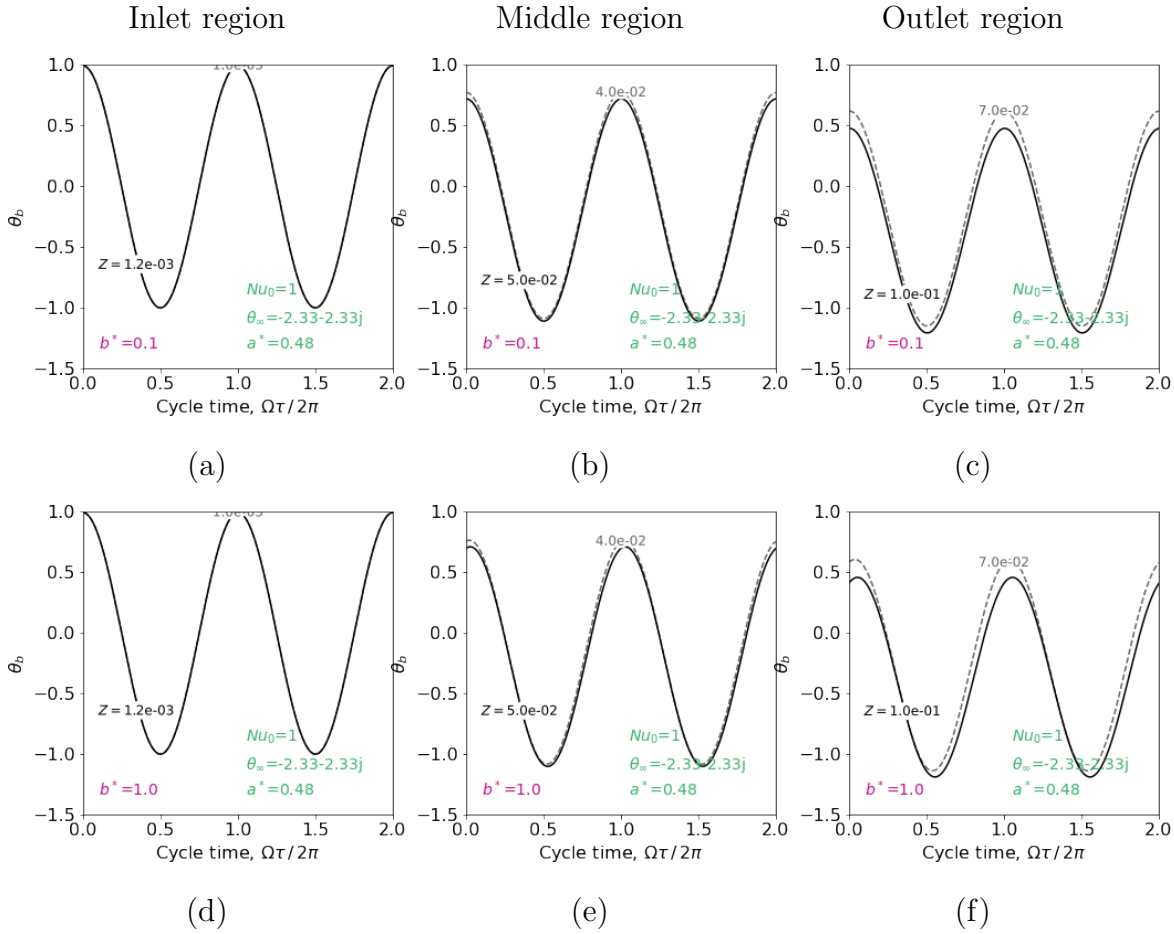


Figure 3.17: Bulk fluid temperature, $\theta_b(Z)$ for $b^* = 0.1$ is plotted in the top row for two different positions within each of the inlet (a), middle (b) and outlet (c) test-section regions, respectively. The bottom row shows the same for $b^* = 1.0$.

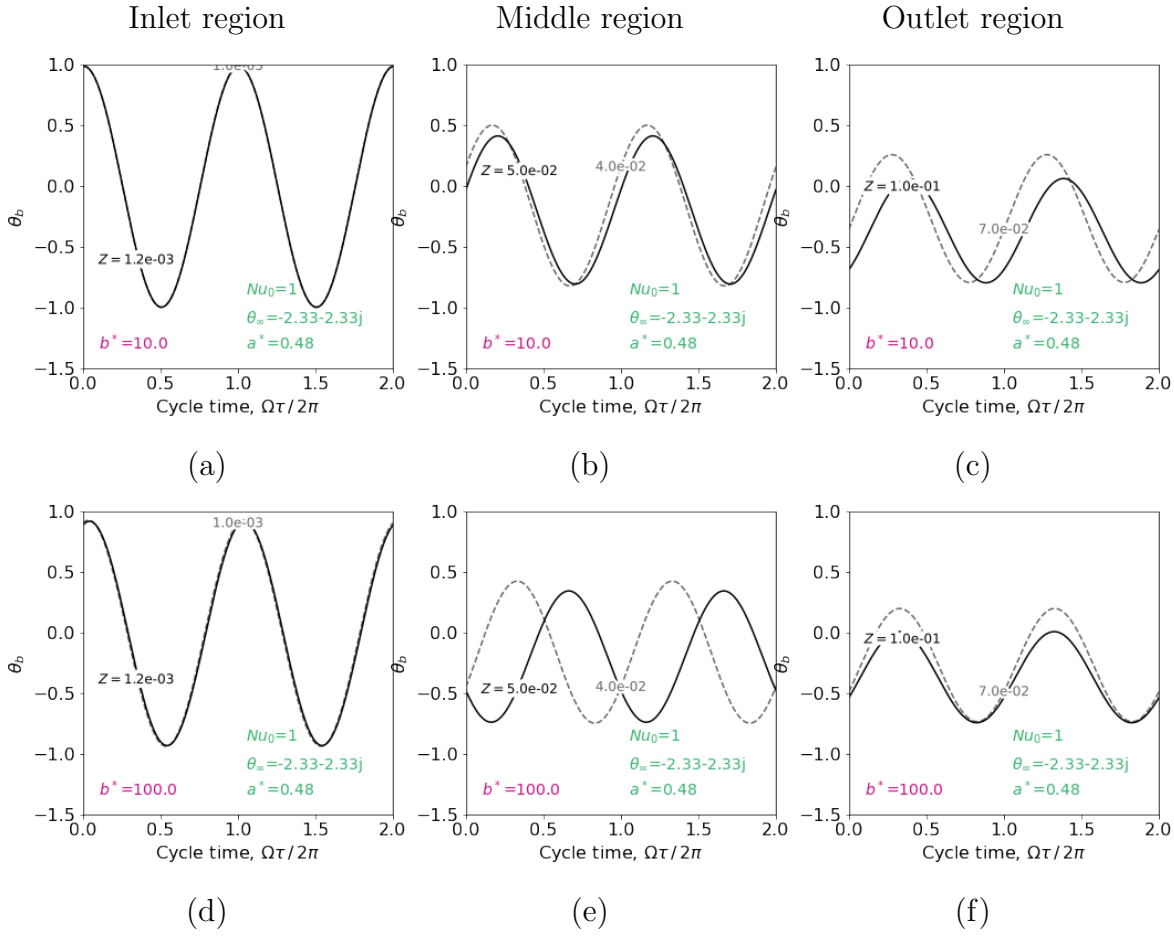


Figure 3.18: Bulk fluid temperature, $\theta_b(Z)$ for $b^* = 10$ is plotted in the top row for two different positions within each of the inlet (a), middle (b) and outlet (c) test-section regions, respectively. The bottom row shows the same for $b^* = 100$.

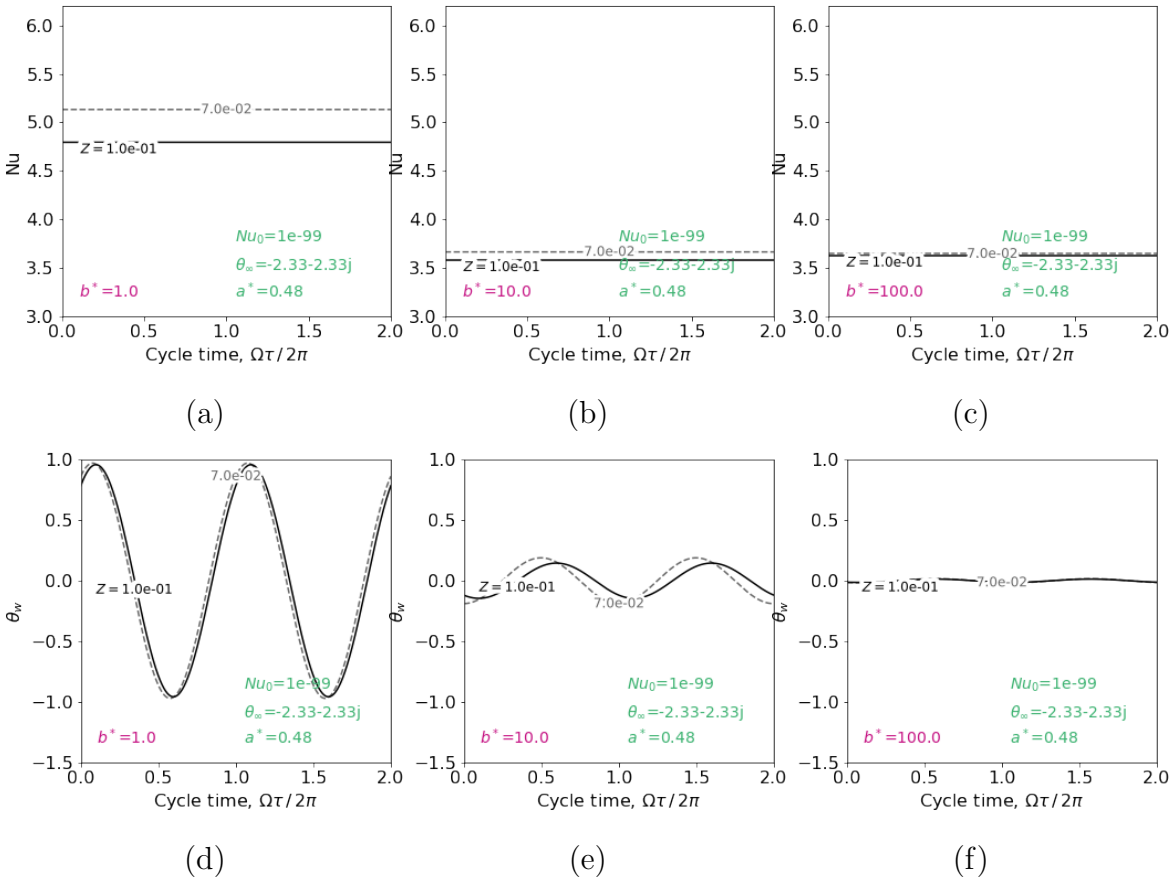


Figure 3.19: Nu and wall temperature, θ_w, for $b^* = 1$ (a, d) $b^* = 10$ (b, e) and $b^* = 100$ (c, f) for the case of zero external natural convection to ambient. Distortions are effectively eliminated by removing the effect of natural convection to ambient surroundings. It remains that the quasi-steady condition is better approximated at higher values of b^*. However the trade-off of increasing b^* can clearly be seen comparing plots (d), (e) and (f) for the wall temperature. Depending on the accuracy of thermocouples used and absolute dimensional temperatures, increasing b^* beyond 10 provides rapidly diminishing returns in terms of satisfying the quasi-steady condition and will lead to higher measurement uncertainty and in turn, a lower signal-to-noise ratio.

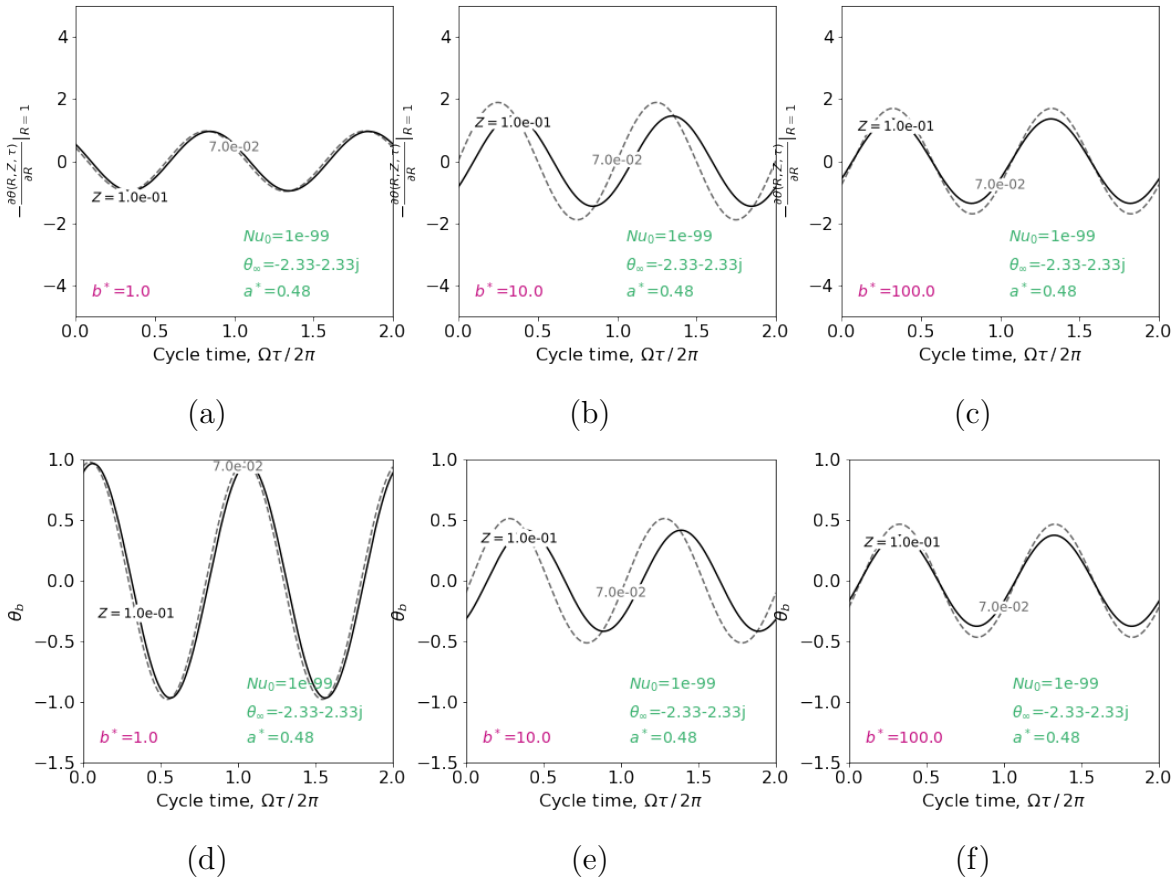
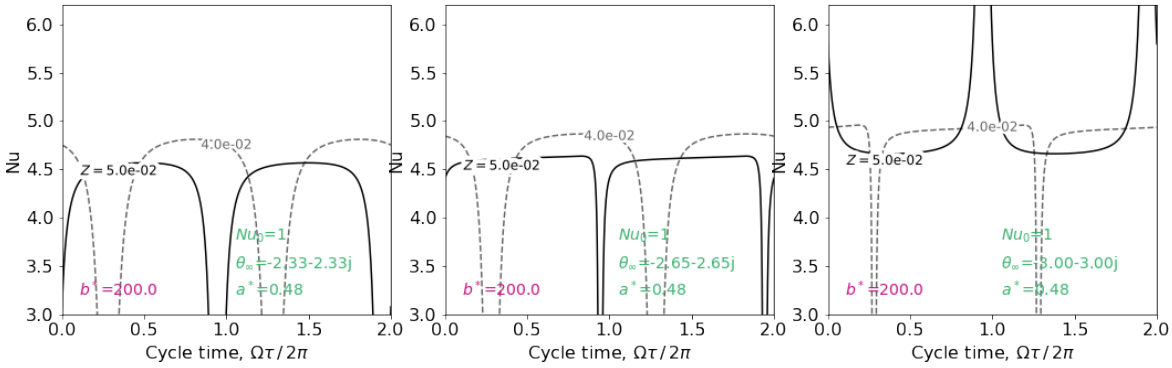


Figure 3.20: Wall heat flux and bulk fluid temperature, θ_b, for $b^* = 1$ (a, d) $b^* = 10$ (b, e) and $b^* = 100$ (c, f) with zero external natural convection to ambient.



(a) (b) (c)

Figure 3.21: Tuning θ_∞ can be used to produce a system response that is better approximated by the quasi-steady condition as shown in shown plots. The tuning is however highly sensitive to the longitudinal distance being measured and therefore eliminating ambient convection distortions would be a more effective strategy if possible

Chapter 4

SHEFRA Experimental Results

4.1 Introduction

SHEFRA is an experimental facility used to investigate the use of frequency response testing methods and frequency domain analysis to extract high fidelity heat transfer measurements while minimizing error and uncertainty. These methods have been applied towards the demonstration of thermal hydraulic similitude between Dowtherm A and Flibe experimental and prototypical forced convection applications. The governing equations for laminar flow in circular channel with a periodically-varying fluid inlet temperatures boundary condition are nondimensionalized and analyzed in the frequency domain. The analytical solutions and scaling terms can be used to obtain the optimal experiment parameters in terms of error and the ability to approximate steady state response even as the temperature in the system are fluctuating.

The first stage of demonstrating thermal hydraulic similitude between a molten salt and surrogate fluid pair, such as Flibe and Dowtherm A, is to confirm that the respective steady or quasi-steady state measurements for Nu as a function of Pr and Re are in agreement. This first phase can be referred to as steady state similitude. Dynamic similitude at higher dimensions is also theoretically possible, where the quasi-steady model is no longer valid and the Nu predictions would appear to deviate. This can be referred to as transient or dynamic thermal hydraulic similitude which would be measured in terms of local heat flux and temperatures. Instead of a Nu number, a basket of other dimensionless parameters would be needed for the scaling of the dynamic system from the surrogate fluid lab scale to prototypical Flibe. An example of this is the group of dimensionless parameters derived in Chapter 3 for a periodically-varying fluid inlet temperature in a cylindrical channel. At first impression, the modelling approach and scaling terms appear specialized to a particular problem. However, there is wide applicability and utility in the modeling of systems with a periodic forcing function. This is because periodic functions can be expanded as a sum of sines and cosines. The Fourier series approximation is ideally suited for the continuous and smooth functions often encountered in physical systems like solutions for heat transfer.

Consequently, the frequency domain-based scaling and modeling approach could be adapted to serve as a versatile tool for scaling and modeling dynamic interactions between molten salt and solid structures for a wide range of arbitrary periodic forcing functions.

The experimental studies presented in this chapter are concerned with confirming thermal hydraulic similitude for steady state operation or for a transient with a quasi-steady heat transfer coefficient. We expect measured Nusselt numbers in the SHEFRA test-section under quasi-steady conditions to be in agreement with steady state analytical solutions and correlations from the literature. This verifies that the equations used for the data reduction procedure are in alignment with the theoretical and physical system. Using the same data, we can determine if the transient model can be validated for the quasi-steady case. The results obtained demonstrate that the transient model has been preliminarily validated for quasi-steady heat transfer conditions. This indicates that for steady laminar flow, Dowtherm A surrogate fluid can be used in a scaled experiment as a surrogate for Flibe, for both thermally developing and developed flow.

Forced convection experiments using Flibe would be needed to confirm this similitude. Laminar and transitional flow experiments were done with Flibe at ORNL in addition to turbulent flow testing. However, the heat transfer coefficients were calculated by integrating over the thermal entrance region. The local heat transfer measurements are not listed in the report [7]. The SHEFRA test section comes about a meter short of allowing flow to become fully developed for the case of laminar flow, and therefore a direct comparison is not possible using the ORNL salt data.

Seeing as the transient model is also capable of resolving heat flux and temperature solutions for when the heat transfer is unsteady, another level of model validation could lead to an advanced capability to simulate unsteady molten salt thermal hydraulics with surrogate fluids.

4.2 Data Reduction

To obtain an expression for the local heat transfer coefficient as a function of experimental parameters and measured values, an energy balance on a channel wall element is used. Approximations made in Section 3.1 for thermally developing laminar flow and turbulent flow can be applied here, with the relevant dimensionless groups summarized in Table 4.2.

Lumped Capacitance Approximation

A channel wall element is assumed to be appropriate to approximate as a lumped capacitance. The following analysis explores the validity of this assumption.

$$\mathrm{Bi} = \frac{hl}{k_s} \tag{4.1}$$

$$\mathrm{Nu} = \frac{hD}{k_f} \tag{4.2}$$

Symbol	Parameter	Value	Units
k_s	Thermal conductivity of copper	385, Ref. [6]	W/(m K)
k_f	Thermal conductivity of fluid	$f(T)$, Dowtherm A	W/(m K)
c_w	Specific heat of wall	8933, Ref. [6]	J/(kg K)
c_p	Specific heat of fluid	$f(T)$, Dowtherm A	J/(kg K)
l	Wall thickness	0.049	in
D	Channel diameter	0.12	in
ρ_w	Wall density	1358	kg/m^3
ρ_f	Fluid density	$f(T)$, Dowtherm A	kg/m^3

Table 4.1: Actual experimental parameters for the SHEFRA test section. The model accounts for the wall thermal capacitance. The wall specific heat and density are represented in a channel element internal energy term in Equation 4.4. The fluid thermal conductivity is used to nondimensionalize the local heat transfer coefficient, to calculate the measured Nusselt number. The solid thermal conductivity is not used in the solid energy model, and in a Biot number analysis to justify the lumped capacitance approximation.

The Biot number is calculated for the test-section using Equation 4.1. An estimated local heat transfer coefficient must be used for the calculation. An upper limit for Nu of 500 is chosen which exceeds that of steady-state turbulent convection heat transfer at Re=10,000 and Pr=15. The conservatively high estimate for the local heat transfer coefficient is calculated using Equation 4.2 to be 20,653 $\mathrm{W\,m^{-2}\,K^{-1}}$. Using this, and parameters found in Table 4.1 such as the characteristic length, the maximum Biot number is expected to be no more than 0.064, which satisfies the condition, Bi < 0.1. In other terms, the maximum Nu for which this approximation is valid is around Nu=800. Table 4.2 summarizes the dimensionless groups for which the data reduction model applies.

Solid Energy Equation

The energy balance in 4.3 includes the net flow of thermal energy from forced convective heat transfer and the time-dependent internal energy of a channel element, U.

$$hA\left(T_b - T_w\right) = U \tag{4.3}$$

The internal energy term is expanded according to the lumped thermal capacitance approximation.

$$hA\left(T_b - T_w\right) = \left(\rho V c_p\right)_s \frac{dT_w}{dt} \tag{4.4}$$

Dividing by unit volume, V, the equation is formulated to use a specific surface area term, a_v.

$$ha_v\left(T_b - T_w\right) = \left(\rho c_p\right)_s \frac{dT_w}{dt} \tag{4.5}$$

Table 4.2: Dimensionless group ranges of applicability for SHEFRA experiment data reduction model. The Reynolds number range spans both laminar and turbulent flow regimes. The Prandtl number range is selected to match prototypical Flibe temperatures of interest, as shown in Figure 4.7. Mixed convection scaling, Gr/Re, was not explicitly represented in the data reduction model. In the case of buoyancy-aided heat transfer, an increased measured Nu number would be detected when comparing between operation in vertical and horizontal tube orientations. The model assumes a large Péclet number that allows longitudinal conduction in the fluid to be neglected, which is the case for the SHEFRA experiment in both laminar and turbulent conditions according to the relevant analysis in Section 3.1.

Parameter	**Applicable Experimental Range**
Reynolds	Laminar and turbulent
Prandtl	$0 - 22$
Gr/Re	$\lesssim 1$
Frequency	$0 - 0.5$ Hz
Biot	$\lesssim 0.1$
Péclet	Large, or $\gtrsim 10^4$

a_v for a cylindrical channel is obtained using the unit volume and surface area of a channel wall element.

$$\begin{aligned} a_v = \frac{A_{wall\,element}}{V_{wall\,element}} &= \frac{2\pi a\, dz}{\left[\pi(l+a)^2 - \pi(a)^2\right]\, dz} \\ &= \frac{2a}{l^2 + 2al} \end{aligned} \tag{4.6}$$

Solving for the local heat transfer coefficient obtains

$$h = \frac{\left(\rho c_p\right)_s \frac{dT_w}{dt}}{a_v \left(T_b - T_w\right)} \tag{4.7}$$

However, the bulk fluid temperature is not known along the length of the test-section and for this reason the fluid energy equation must be used.

Fluid Energy Equation

A differential approach is used to create an energy balance model of a channel fluid segment. The first law of thermodynamics requires that the heat transferred from the wall to the stream equals the sum of the enthalpy gain of the stream and any change in the stored energy for a control volume ∂x.

$$mc_p \frac{\partial T_m}{\partial t} + \dot{m}(h_{x+\partial x} - h_x) = q'' \, 2\pi r_0 \, \partial x \tag{4.8}$$

Modeling the fluid as an incompressible liquid with negligible pressure changes yields

$$\partial h \simeq c_p \partial T_m \tag{4.9}$$

$$mc_p \frac{\partial T_m}{\partial t} + \dot{m}\,\partial h = q''\,2\pi r_0\,\partial x \tag{4.10}$$

Dimensional analysis:

$$\mathrm{kg} \cdot \frac{\mathrm{J}}{\mathrm{kgK}} \cdot \frac{\mathrm{K}}{\mathrm{s}} + \frac{\mathrm{kg}}{\mathrm{s}} \cdot \frac{\mathrm{J}}{\mathrm{kgK}} \cdot \mathrm{K} = \frac{\mathrm{J}}{\mathrm{m^2 s}} \cdot \mathrm{m} \cdot \mathrm{m} \tag{4.11}$$

$$\rho V c_p \frac{\partial T_m}{\partial t} + \dot{m} c_p \partial T_m = q''\,2\pi r_0\,\partial x \tag{4.12}$$

$$V = \pi r_0^2\,\partial x \tag{4.13}$$

$$\dot{m} = \rho c_p U A \tag{4.14}$$

$$A = \pi r_0^2 \tag{4.15}$$

$$\dot{m} = \rho c_p U \cdot \pi r_0^2 \tag{4.16}$$

$$\rho c_p \frac{\partial T_m}{\partial t} \cdot \pi r_0^2\,\partial x + \rho c_p U\,\partial T_m \cdot \pi r_0^2 = q''\,2\pi r_0\,\partial x \tag{4.17}$$

$$\rho c_p \frac{\partial T_m}{\partial t} \cdot \pi r_0^2\,\partial x + \rho c_p U\,\partial T_m \cdot \pi r_0^2 = q''\,2\pi r_0\,\partial x \tag{4.18}$$

Divide by ∂x

$$\rho c_p \frac{\partial T_m}{\partial t} \cdot \pi r_0^2 + \rho c_p U\,\frac{\partial T_m}{\partial x} \cdot \pi r_0^2 = q''\,2\pi r_0 \tag{4.19}$$

Divide by $2\pi r_0$ to isolate q''.

$$\rho c_p \frac{r_0}{2} \left(\frac{\partial T_m}{\partial t} + U \frac{\partial T_m}{\partial x} \right) = q'' \tag{4.20}$$

Dimensional analysis:

$$\begin{aligned} \frac{\mathrm{kg}}{\mathrm{m^3}} \cdot \frac{\mathrm{J}}{\mathrm{kgK}} \cdot \mathrm{m} \left(\frac{\mathrm{K}}{\mathrm{s}} + \frac{\mathrm{m}}{\mathrm{s}} \frac{\mathrm{K}}{\mathrm{m}} \right) &= \frac{\mathrm{W}}{\mathrm{m^2}} \\ \frac{\mathrm{W}}{\mathrm{m^2}} &= \frac{\mathrm{W}}{\mathrm{m^2}} \end{aligned} \tag{4.21}$$

At this stage, a formula for T_m can be obtained. For a bundle of small streams $\rho u\,\partial A$, the first law in Equation 4.18 can be rewritten as

$$q''\,2\pi r_0\,\partial x = \rho c_p \frac{\partial T_m}{\partial t} \cdot \pi r_0^2\,\partial x + \partial \iint_A \rho u c_P T \partial A \tag{4.22}$$

Combining Equation 4.22 with Equation 4.20, we obtain

$$\rho c_p \frac{r_0}{2}\left(\frac{\partial T_m}{\partial t}+U\frac{\partial T_m}{\partial x}\right)2\pi r_0\,\partial x=\rho c_p\frac{\partial T_m}{\partial t}\cdot\pi r_0^2\,\partial x+\partial\iint_A \rho u c_P T\partial A \tag{4.23}$$

$$\rho c_p A\frac{\partial T_m}{\partial t}\,\partial x+\partial T_m\rho c_p UA=\rho c_p\frac{\partial T_m}{\partial t}\cdot\pi r_0^2\,\partial x+\partial\iint_A \rho u c_p T\partial A \tag{4.24}$$

The time-dependent terms cancel, and after assuming constant-property tube flow, this expression reduces to the mean temperature definition.

$$T_m=\frac{\iint_A uTdA}{UA} \tag{4.25}$$

Our attention now returns to the fluid energy balance. Recognizing that $q''=-h(T_b-T_w)$ from Equation 4.3, and substituting into Equation 4.20, we get

$$\rho c_p\frac{r_o}{2}\left(\frac{\partial T_m}{\partial t}+U\frac{\partial T_m}{\partial x}\right)=-h(T_b-T_w) \tag{4.26}$$

Finite Difference Model

Using finite differences, a local bulk fluid temperature can be estimated using a reference bulk fluid temperature as shown in Equation 4.27.

$$\rho c_p\frac{r_o}{2}\left(\frac{\partial T_m}{\partial t}+U\frac{\Delta T_m}{\Delta x}\right)=-h(T_b-T_w) \tag{4.27}$$

$$\rho c_p\frac{r_o}{2}\left(\frac{\partial T_m}{\partial t}\right)_{\text{Node }i}+\rho c_p\frac{r_o}{2}\left(U\frac{\Delta T_m}{\Delta x}\right)_{\text{Node }i\text{ to }j}=-h\left(T_b-T_w\right)_{\text{Node }i} \tag{4.28}$$

Solving for the local mean or bulk fluid temperature

$$\rho c_p\frac{r_o}{2}\left(U\frac{\Delta T_b}{\Delta x}\right)_{\text{Node }i\text{ to }j}=-h\left(T_b-T_w\right)_{\text{Node }i}-\rho c_p\frac{r_o}{2}\left(\frac{\partial T_b}{\partial t}\right)_{\text{Node }i} \tag{4.29}$$

$$\rho c_p\frac{r_o}{2}U\frac{T_{b,j}-T_{b,i}}{\Delta x}=-h\left(T_{b,i}-T_w\right)-\rho c_p\frac{r_o}{2}\frac{\partial T_{b,i}}{\partial t} \tag{4.30}$$

$$T_{b,j}=\frac{2(x_j-x_i)}{\rho c_p U r_o}\left[-h\left(T_{b,i}-T_w\right)-\rho c_p\frac{r_o}{2}\frac{\partial T_{b,i}}{\partial t}\right]+T_{b,i} \tag{4.31}$$

The scheme implemented is illustrated in the nodal diagram in 4.1. To calculate the measured heat transfer at some distance downstream of the inlet represented by node n, it is first calculated at a node that represents the inlet using Equation 4.7. The code must then calculate the heat transfer coefficient at each node between the inlet and the desired node. The heat transfer coefficient from the previous node, is used to estimate the local fluid bulk

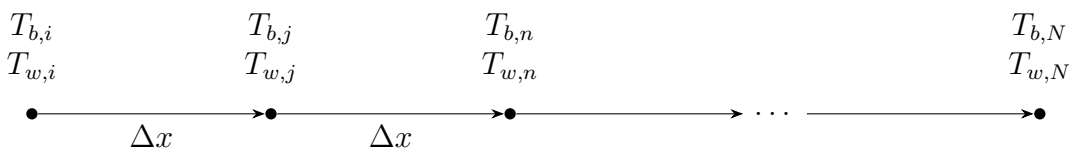


Figure 4.1: Nodal diagram of the finite difference model used to estimate downstream bulk fluid temperatures

temperature at a node using Equation 4.31. Since there are five thermocouples distributed along the test section length, there are also five corresponding nodes. The wall temperature measurements are the anchors to the finite difference model. The inlet bulk temperature is used rather than an outlet bulk temperature because the temperatures are higher and therefore the uncertainty in that measurement becomes relatively smaller. This model neglects any potential mixed convection effects which may be important for high enough temperature gradients. Additional model development and testing in a horizontal orientation will be conducted in the future for further investigation. The code implementation of the described data reduction is provided in Appendix C.

4.3 Code Development Methodology

Lessons learned from past experiments

Debugging was a major challenge during the development of data reduction codes for the Pebble-bed Heat Transfer Experiment due to the absence of physical test cases needed to assess the correctness of the code implementation. The approach taken was to carefully review and manually test individual units of code, a tedious and error-prone activity. Future work on the estimation of convective heat transfer coefficients in pebble-beds in the Reynolds and Prandlt number ranges of interest for PB-FHRs would benefit from using physical test cases derived from analytical solutions. The lack of physical test cases led to an increased time burden of manual testing and increased risks of breaking established code behavior during ongoing development.

SHEFRA code development approach

Object-oriented programming

Anticipating SHEFRA, The PBHTX code base was written with abstractions to allow the analysis of other transient convection heat transfer experiments. Despite this, when it came time to extract results from SHEFRA experimental data, the analysis code was restructured to make the code easier to understand and debug. The rewrite aims to improve on

the application of the object-oriented programming principles of abstraction, inheritance, encapsulation, and polymorphism.

To illustrate this in practice, consider the calculation of the flow Reynolds number. Reynolds numbers were initially evaluated within fluid class instances called fluid objects. This however required information on the dimensions of the flow channel. To facilitate this, an interface was made between the fluid and a test-section class. The two classes would then interface again before the convective heat transfer calculation. The result is a web of interfaces that are poorly encapsulated and difficult to test and debug. A more logical grouping would result if the Reynolds number was to be calculated within a separate node class instance that would inherit fluid and solid object information. Encapsulation is achieved as the fluid class is now limited to only provide information on the fluid thermophysical properties. Multiple inheritance is also achieved with this change, meaning that the node class is logically derived from the fluid and solid base classes. When the derived information is combined with channel-specific attributes such as the physical dimensions, it provides an appropriate and logical place to evaluate the Reynolds number.

Testing and Test-driven development

The simple circular channel geometry of the SHEFRA test-section and the availability of analytical solutions presented an opportunity to develop physical test cases. Assertions on the expected results of code units are continually tested. Tests are conventionally written after a unit of code is completed however an emerging best practice is the test-driven development (TDD). TDD can be summarized in the following steps below [4].

1. Write a test
2. Run all tests. The new test should fail
3. Write the simplest code that passes the new test
4. Verify that all tests are passing
5. Refactor code as necessary, and running tests in between and after.

This sequence is repeated until the required functionality is achieved. Refactoring can include restructuring, removing hard code, removing duplicate code, etc..

Selection of physical test cases

The data reduction code processes the raw measurement data and provides the inputs needed for model calculations. Code functionality can be segmented for testing in a modular fashion. The output of a code module can be validated using a true solution from a known physical model, or through other assertions. This section will provide examples of code functions and corresponding test cases.

Heat transfer coefficient at the inlet

The simplest version of the calculation of the heat transfer coefficient is at the test section inlet where the bulk fluid temperature is directly measured.

Physical test case 1: steady inlet temperature

The most basic physical test case is for a steady fluid inlet temperature and a wall temperature initial condition. To create the inputs for the test case, model parameters including the heat transfer coefficient and mean temperature are chosen and the solution is obtained for the tube wall temperature. The test will read the solution temperature data, run the code module, and then check whether the output value for the heat transfer coefficient matches the value used to generate the test data. The model assumes that the tube wall can be approximated as a lumped capacitance.

Starting with the tube wall energy model in Equation 4.32

$$\frac{(\rho c_p)_w}{a_v}\frac{\partial T_w(t,z)}{\partial t} = h(T_b(t,z) - T_w(z)) \tag{4.32}$$

$$a_v = \frac{2a}{l^2 + 2la} \tag{4.33}$$

where, a_v is the specific surface area, a is the channel inner radius, and l is the wall thickness.

We impose an initial condition

$$T_w(0) = T_{w,i} \tag{4.34}$$

Introducing the temperature difference,

$$\theta(t,z) \equiv T_w(t,z) - T_b(t,z) \tag{4.35}$$

Evaluating the time derivative of the temperature difference given a constant mean fluid temperature, we get

$$\frac{\partial \theta}{\partial t} = \frac{\partial T_w}{\partial t} \tag{4.36}$$

Substituting the temperature difference and its derivative back into the energy balance, we obtain

$$\frac{(\rho c_p)_w}{h a_v}\frac{\partial \theta}{\partial t} = -\theta \tag{4.37}$$

Separating variables and integrating over time from the initial condition

$$\frac{(\rho c_p) w}{h a_v}\int \theta_i{}^{\theta}\frac{\partial \theta}{\theta} = -\int_0^t \partial(t) \tag{4.38}$$

where

$$\theta_i \equiv T_{w,i} - T_b \tag{4.39}$$

After evaluating the integral we get

$$\frac{(\rho c_p)_w}{h a_v} \ln \frac{\theta_i}{\theta} = t \tag{4.40}$$

Which can be rearranged as follows

$$\frac{\theta_i}{\theta} = \frac{T_w - T_b}{T_{w,i} - T_b} = \exp\left[\left(-\frac{h a_v}{(\rho c_p)_w}\right) t\right] \tag{4.41}$$

Solving for the tube wall temperature we arrive at

$$T_w = (T_{w,i} - T_b) \exp\left[\left(-\frac{h a_v}{(\rho c_p)_w}\right) t\right] + T_b \tag{4.42}$$

We also require the time derivative of the tube wall temperature which is obtained by rearranging the original energy balance equation.

$$\frac{\partial T_w(t,z)}{\partial t} = \frac{h a_v}{(\rho c_p)_w}(T_b(t,z) - T_w(z)) \tag{4.43}$$

The following parameters were selected for the test case.

Table 4.3: Parameter values for heat transfer analysis

Symbol	Parameter	Value (SI units)	Units (SI)
h	Heat transfer coefficient	2000	W/(m^2 K)
T_b	Bulk fluid temperature	400	K
$T_{w,0}$	Initial wall temperature	300	K
ρ	Wall density	8933	kg/m^3
c_p	Specific heat capacity	385	J/(kg K)
a	Channel inner radius	0.02	m
l	Wall thickness	1E−3	m

The resulting temperature distribution for $t = 0$ to $t = 10$s is plotted in Figure 4.2, and is saved in a .csv formatted table.

Designing a test in practice is often non-trivial, and care must be taken into ensuring that the functionality of the code unit is actually being tested. For illustrative purposes and as a practical example for the reader, the first test case as coded in Python is included in this section below. The input data is first packaged into a form that the WallEnergyBalance class expects. The test will only pass if the correct heat transfer coefficient is calculated. In

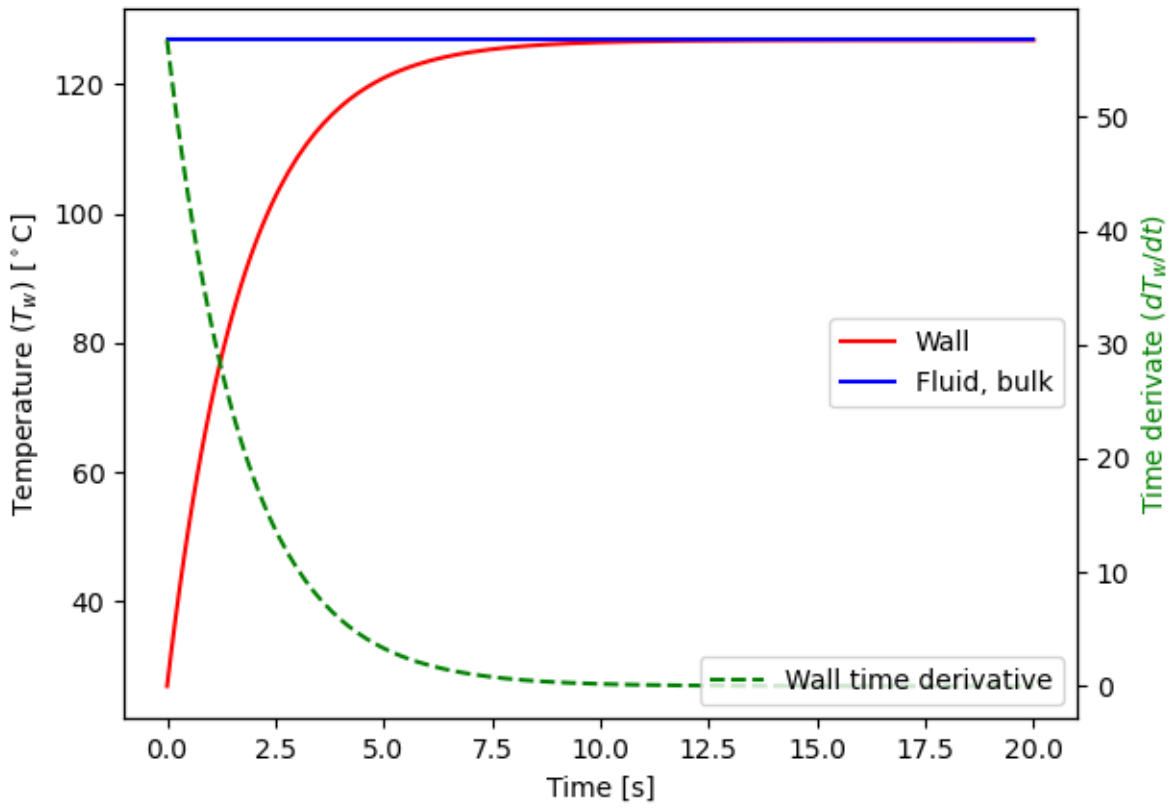


Figure 4.2: Steady fluid inlet temperature physical test data

following the TDD methodology, this test was written before the energy balance code was written. Development of the code unit is complete once the test passes. The Pytest package was used to run the test files.

```
from src.shefra.model import WallEnergyBalance
import numpy as np
import pandas as pd

class Test_WallEnergyBalance:
    def test_WallEnergyBalanceSteadyInlet(self):

        #Prepare test input 'node' object

        path=("testInputs/WallEnergyBalanceSteadyInlet.csv")
```

```
        testdata=pd.read_csv(path)
        time=testdata['time'].to_numpy()
        solidTemp=testdata['solidTemp'].to_numpy()
        solidTempDerivative=testdata['solidTempDerivative'].to_numpy
            ()

        #expected result

        h=1500

        #test inputs

        density=8933
        specificHeat=385
        radiusInner=0.02
        thickness=1e-3

        # mock a Node instance with input data and parameters

        class Node:
            def __init__(self,density,specificHeat,radiusInner,
                thickness,
            solidTemp,solidTempDerivative,time):
                self.density=density
                self.specificHeat=specificHeat
                self.dim={
                    'radiusInner':radiusInner,
                    'thickness': thickness
                    }
                self.solidTemp=solidTemp
                self.solidTempDerivative=solidTempDerivative

        node = Node(density,specificHeat,radiusInner,thickness,
            solidTemp,solidTempDerivative,time)


        solidModel = WallEnergyBalance(node)

        assert (solidModel.h == np.ones(len(time))*h).all
```

Listing 4.1: Heat transfer coefficient code unit test in Python.

Physical test case 2: periodically-varying inlet temperature

An additional test case for the tube wall energy balance is designed to ensure the code can accept non-steady fluid temperature data. This will more closely resemble actual experimental data. This model was also used in the system design phase of the physical experiment.

The energy balance equation is the same as in the previous test.

$$\frac{\partial T_w(t,z)}{\partial t} = \frac{ha_v}{(\rho c_p)_w}(T_w(z) - T_b(t,z)) \tag{4.44}$$

However, now the bulk fluid temperature at the inlet boundary is a periodic function defined as

$$T_b(t) = -K\sin(2\pi ft) + T_0 \tag{4.45}$$

where K is an amplitude factor, f is the frequency of the periodic function, t is the time, and T_0 is the time-averaged or baseline bulk fluid temperature. The i subscript is henceforth dropped from $T_{b,i}$.

This is a linear non-homogeneous differential equation. The solution is the sum of a homogeneous and a particular solution. The homogenous part is solved as follows.

$$\frac{\partial T_h}{\partial t} = \frac{ha_v}{(\rho c_p)_w}(-T_h) \tag{4.46}$$

$$T_h = c_0 \exp\frac{-ha_v t}{(\rho c_p)_w} \tag{4.47}$$

Next, the particular solution is solved using the method of undetermined coefficients,

$$T_w(t) = c_0 \exp\frac{-ha_v t}{(\rho c_p)_w} + c_1\cos(2\pi ft) + c_2\sin(2\pi ft) + c_3 \tag{4.48}$$

where

$$c_1 = \frac{ha_v}{(\rho c_p)_w 2\pi f}\left(K - \frac{ha_v}{(\rho c_p)_w 2\pi f}\right) \tag{4.49}$$

$$c_2 = \frac{-ha_v}{(\rho c_p)_w 2\pi f} \tag{4.50}$$

$$c_3 = T_0 \tag{4.51}$$

Applying an initial condition

$$T_w(0) = T_{w,0} \tag{4.52}$$

The last constant is determined to be

$$c_0 = T_{w,0} - c_1 - c_3 \tag{4.53}$$

Taking the time derivative of the wall temperature solution we obtain

$$T_w(t) = -\frac{Ka_v^2h^2\sin(2\pi ft)}{a_v^2h^2+4\pi^2c_p^2f^2\rho^2} + \frac{2\pi Ka_vc_pfh\rho e^{\frac{a_vht}{c_p\rho}}}{a_v^2h^2+4\pi^2c_p^2f^2\rho^2} - \frac{2\pi Ka_vc_pfh\rho\cos(2\pi ft)}{a_v^2h^2+4\pi^2c_p^2f^2\rho^2} + T_0 \quad (4.54)$$

$$\frac{\partial}{\partial t}T_w(t) = -\frac{Ka_v^2h^2\sin(2\pi ft)}{a_v^2h^2+4\pi^2c_p^2f^2\rho^2} + \frac{2\pi Ka_vc_pfh\rho e^{\frac{a_vht}{c_p\rho}}}{a_v^2h^2+4\pi^2c_p^2f^2\rho^2} - \frac{2\pi Ka_vc_pfh\rho\cos(2\pi ft)}{a_v^2h^2+4\pi^2c_p^2f^2\rho^2} + T_0 \quad (4.55)$$

The wall temperature solutions can now be used to generate test case data given a set of test parameters.

Table 4.4: Parameter values for heat transfer analysis

Symbol	Parameter	Value (SI units)	Units (SI)
h	Heat transfer coefficient	2000	W/(m^2 K)
$T_{w,0}$	Initial wall temperature	300	K
ρ	Wall density	8933	kg/m^3
c_p	Specific heat capacity	385	J/(kg K)
a	Channel inner radius	0.02	m
l	Wall thickness	1E−3	m
K	Fluid temperature amplitude	50	K
f	Frequency	0.1	Hz
T_0	Baseline fluid temperature	400	K

The resulting test data set is plotted in Figure 4.3.

The test data generated simulates experimental measurements of the change in wall temperature in response to a periodic variation in the local bulk fluid temperature. This model is applicable at any node along the channel length, as long as T_b represents the local fluid temperature. In the case of SHEFRA, the bulk fluid temperature is only directly measured at the inlet and must be inferred for node locations along the length of the tube using a finite difference approximation.

Node construction

The channel test-section is organized into nodes; one for each thermocouple location on the tube wall. The heat transfer coefficient is measured at each axial node location. The code test will verify that the fluid and solid class instances are combined to construct a channel node class that is packaged in a form that the WallEnergyBalance class functions will expect. Flow parameters of the Channel Node class include flow parameters such as the Reynolds, and Prandtl numbers. Inputs to the node constructor includes attributes from a Channel object. These are properties that are common to all nodes along the channel, such as the channel dimensions.

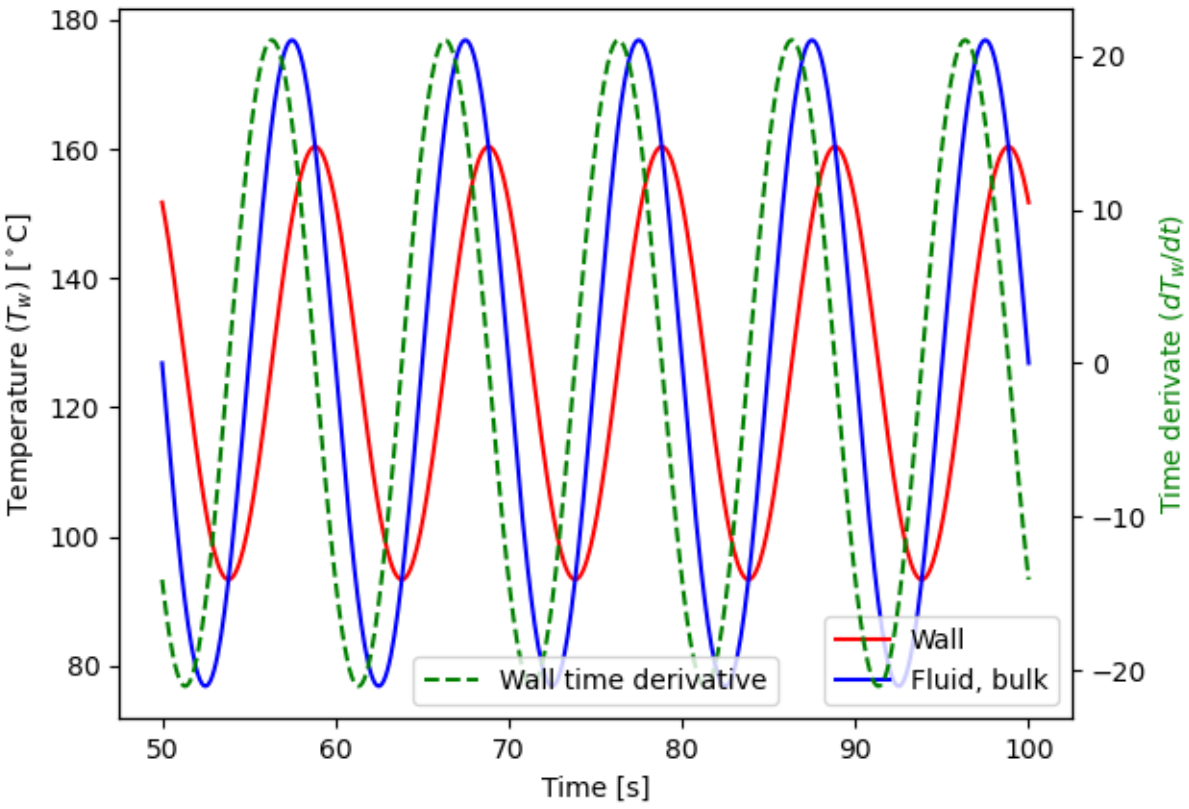


Figure 4.3: Space-independent model simulation that shows the dynamic wall temperature response to a sinusoidally-varying fluid inlet temperature in a circular channel.

For the Node construction test case, the test data from one of the previously described tests is used to construct mock Channel, Fluid and Solid inputs. The test verifies that the input attributes can be accessed from the Node output in the specified format.

This test case also introduces the getter and setter design pattern to protect class attributes from inadvertent changes. Class attributes are configured as private, read-only properties and getter and setter methods expose public interfaces in a controlled and safe manner. To illustrate the utility of this design pattern, the getter methods for obtaining material properties is shown here to require a temperature input before the attribute is retrieved. The retrieval happens through a predefined function call.

```
1 class Solid:
2
3     # initialize solid object using constructor method
```

```
4     def __init__(self,density,specificHeat,solidTemp,
          solidTempDerivative,time):
5         self.__density=density
6
7     # getter method
8     def get_density(self, temp):
9         self.__density=props.solidDensity(temp)
10        return self.__density
11
12    # example of how the getter method is called
13 class Node:
14    def __init__(self,channel,fluid,solid):
15        self.solidDensity=solid.get_density(solid.temp)
```

Listing 4.2: Getter and setter methods help to prevent unintended changes to object attributes

If multiple solid materials were present in the properties tables, the above function would also require an additional argument to identify the material.

Noisy derivatives

The wall energy balance modeled in Equation 4.3 contains the time derivative of the solid wall temperature. Due to the the noise associated with making physical measurements, the solid wall temperature is not perfectly smooth. The finite difference method for taking the derivative by using adjacent time step results in large fluctuations due to error propagation. A common method is a Savitsky-Golay filter which is applied to increase the precision of the data without distorting the signal tendency. In a convolution process, successive sub-sets of adjacent data points are fitted with a low-degree polynomial by the method of linear least squares. Other options include convolution with a Gaussian in the Fourier space or multi-pass moving-average filtering.

A study was done to compare different methods for computing the temperature derivative with respect to time. Solid temperature data from the periodic inlet temperature physical test case was used and random noise was added to simulate noisy measurement samples. The differentiation methods were evaluated by comparing the RMS error compared to the true derivative of the solid wall temperature.

As expected, the Fourier spectral derivative had the least error for computing noisy derivatives for the SHEFRA system with a sinusoidal forcing function. However, the Fourier method proved to be inconsistent and the physical test used was not comprehensive to the types of errors introduced due to the basic cut-off filter used, i.e. all frequencies above a threshold are zeroed. More sophisticated frequency filters are likely to mitigate this problem. For the purposes of the experiment, the Savitz-Golay method was chosen due to its acceptable error, and consistency.

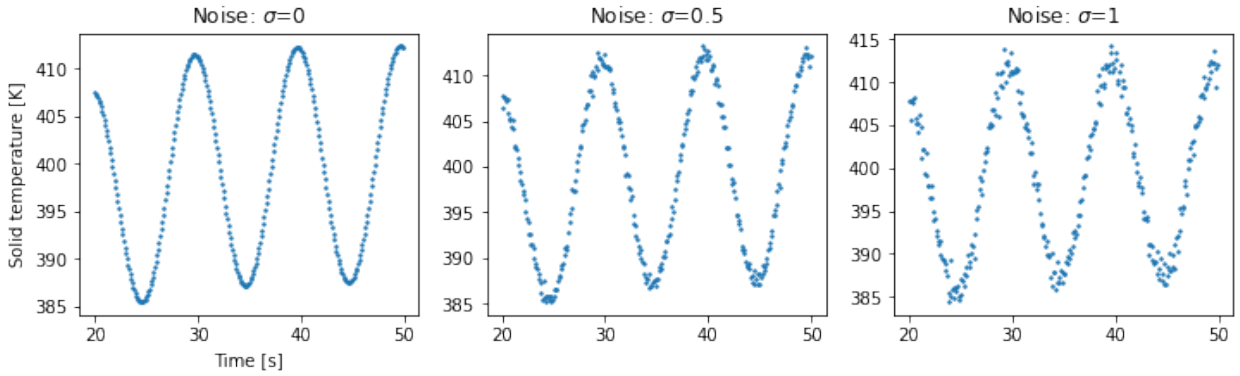


Figure 4.4: Simulated wall temperature data with random noise added.

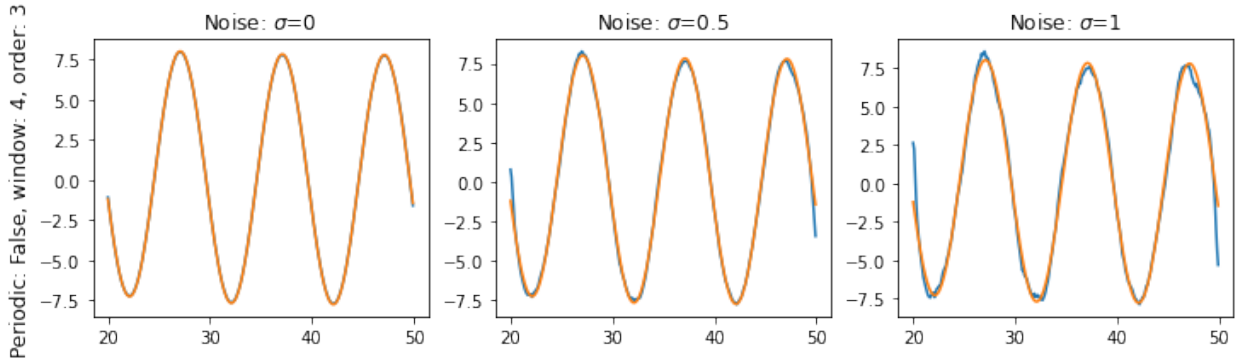


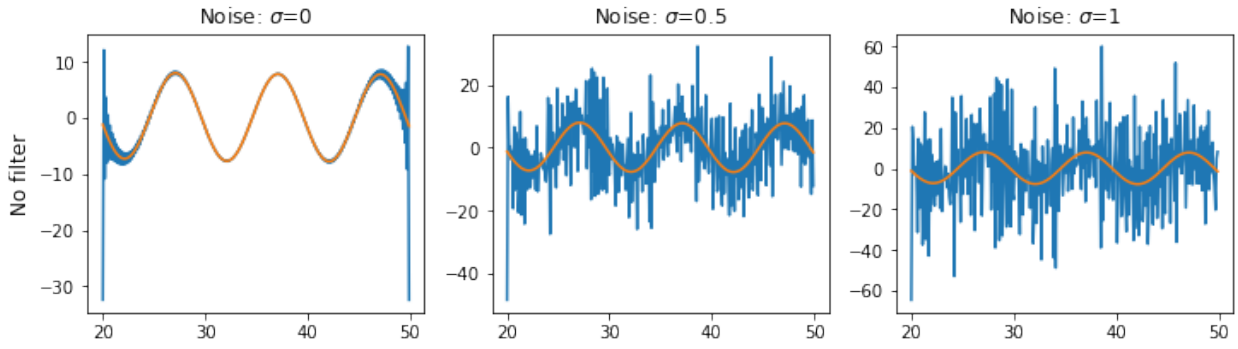
Figure 4.5: Savitz-Golay filter result plotted with the true derivative.

Table 4.5: RMS values for different time derivative methods

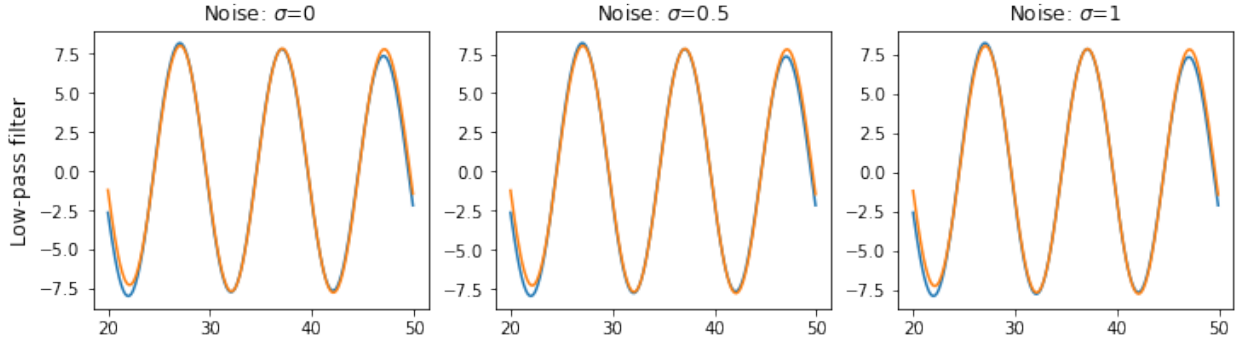
Method	$\sigma = 0$	$\sigma = 0.5$	$\sigma = 1$
Savitz-Golay filter	0.0345	0.3124	0.6094
Fourier spectral derivative (no filter)	3.0426	10.0525	19.4022
Fourier spectral derivative (low pass filter)	0.4052	0.3944	0.3850
Fourier spectral derivative (low pass filter, detrended)	0.1791	0.1796	0.1829

Integration testing

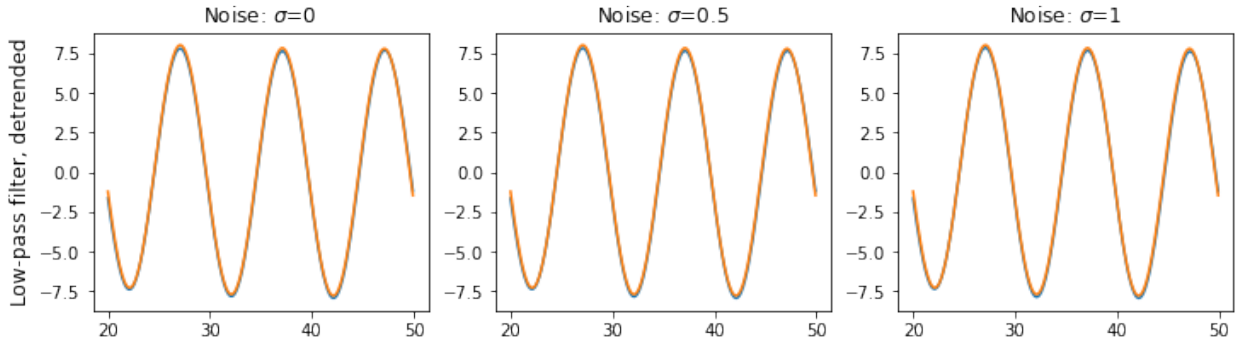
Once the unit tests were written, an end-to-end integration test was written to simulate the entire data reduction process from file import to the calculation of the final parameter. To pass, the code must reproduce the actual heat transfer coefficient from the periodically-varying fluid inlet temperature problem. The input to the integration test is a .csv file that



(a) Fourier spectral derivative with no filter



(b) Fourier spectral derivative with a low pass filter



(c) Fourier spectral derivative with low pass filter applied after removing the linear trend or drift

Figure 4.6: Fourier spectral derivative result plotted with the true derivative.

includes the the inlet fluid temperature, solid wall temperature, and mass flow rate and is formatted similarly to the experiment data files. For clarity, the wallEnergyBalance test mentioned previously only tests the final equation solving step of the data reduction process, while the integration test follows the stream of data through the entire process.

4.4 Experimental Apparatus

A comprehensive overview of the experiment design and apparatus is detailed in Chapter 1, however an abbreviated version is provided in this chapter. The SHEFRA test section is a 2 m long copper tube with an outer diameter dimension of 0.25" and a thickness of 0.049". The tube conforms to US Navy specification MIL-T-24107B, which features thicker walls compared to standard specifications. The test section is instrumented with five Type-T Omega CO3-T "cement-on" surface thermocouples. The thermocouple pads were cemented using Omega 400 and further secured with Teflon tape. Bulk fluid temperature is measured at the inlet of the SHEFRA test section by inserting a Type-T ungrounded, sheathed thermocouple with a bead diameter of 0.02" to provide a fast response. A Coriolis mass flowmeter (Siemens SITRANS FC430) is used to measure the flow rate of the heat transfer fluid through the loop. A National Instruments DAQ was used to deploy the analog power supply controller which worked by estimating the resistance in the heater Nichrome coiled wire and proportionally adjusting the voltage to follow the desired power output. The DAQ was also used to read and store the thermocouple and flow meter measurement data in the control computer. The fluid used is Therminol VP-1, a heat transfer fluid that can be scaled to match the Prandtl of Flibe for temperatures expected in a prototypical Mk1 PB-FHR.

4.5 Procedure

Standard operating procedure documents for SHEFRA loop loading, draining and operation were drafted and approved after a safety review. This section summarizes relevant procedure steps for the data collected and analyzed in this chapter. After the facility has been prepared for testing, the variable frequency drive (VFD) is used to gradually increase the centrifugal pump's speed until the Coriolis mass flow meter measured a rate of 80 kg/h. Cooling water is then flowed through the water-to-oil plate heat exchangers. The desired sinusoidal power profile is loaded into the LabVIEW project to control the power sent to a Nichrome-wire heater. This heater heats the Dowtherm A before it enters the copper test-section. Each run consists of a different combination of heater power cycle frequencies, mean and amplitude values, and pump speed. These operational conditions translate to system parameters that describe the periodically-varying inlet bulk temperature summarized in Table 4.6.

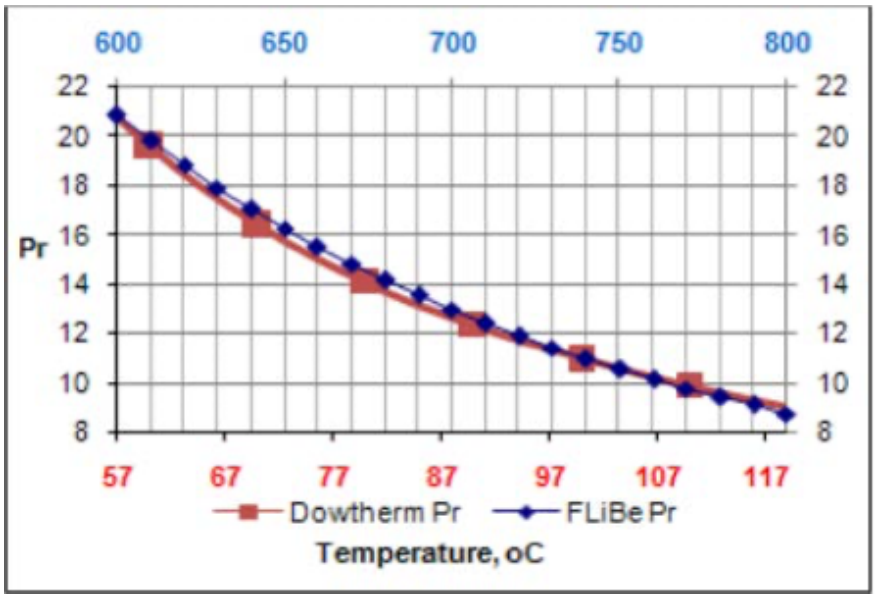


Figure 4.7: Prandtl number scaling between Dowtherm A and Flibe. [26, 24, 19, 21].

4.6 Data collection

At the time the experiment was conducted, the analytical model and relevant scaling terms had not yet been developed. A simple model for the fluid and solid interaction at the inlet was used to design the test-section, however the model could not predict downstream system behavior. For this reason, it was not fully understood which system parameters would enable an optimal response. Nonetheless, a variety of power settings were used to cover the entire Prandtl range of interest and different inlet oscillations frequencies were used. Table 4.6 lists the experimental runs in terms of the the important system parameters. Mass flow rate was steadily increased until the maximum reading on the pressure gauge situated near the pyrex glass heater was reached. The mass flow rate was then kept constant for all experimental runs. At the time, the intent was to collect turbulent flow data, however the mass flow rate was not sufficient to produce consistently turbulent flow. A new pressure gauge was ordered in order to safely increase the mass flow rate for future turbulent flow measurements. Despite a constant mass flow rate, the collected data covered flow in the Reynolds range of 3000 to 10,000, and in many cases the estimated Re value shifted by thousands within the same cycle, creating unpredictable results. A few runs contained data where the fluid remained in a single flow regime, providing useful data. Ultimately, the SHEFRA theoretical research shifted towards the modeling of laminar flow, which meant that a sample of the collected data was sufficient for preliminary assessment of model validity and measuring quasi-steady Nusselt values.

With the benefit of the scaling terms from Chapter 3, we can nondimensionalize the experimental run system parameters as seen in Table 4.8.

Run	Mean Cycle Temperature (°C)	Amplitude (°C)	Inlet Oscillation Frequency (Hz)	Mass Flow Rate (kg/h)	Time Stamp
1	50.5	22.5	0.0833	75.5	0508P-111921
2	51.1	11	0.1	75	0344P-111921
3	51.1	22	0.1	75	0405P-111921
4	77.2	40.8	0.1	78.7	0419P-111921
5	100.9	38.3	0.1	80	0435P-111921
6	50.4	20.7	0.125	77.2	0519P-111921
7	50.4	20.8	0.125	77.2	0528P-111921
8	50.6	18.5	0.1667	76	0452P-111921
9	50.3	15	0.25	75.9	0458P-111921

Table 4.6: Experimental runs and cycle parameters.

We can now retroactively apply the guidelines from Chapter 3 for selecting the optimal set of dimensionless parameters. For this experiment, the system must be in a quasi-steady state. Larger values of b^* will converge the system to quasi-steady state. This is discussed further in Section 4.8. The trade-off is that as b^* increases, the wall temperature amplitudes become small. This becomes an issue as the amplitude approaches the measurement uncertainty of the thermocouple used. Reviewing Table 4.8, the run with the highest inlet oscillation frequency of 0.25 Hz has the highest b^* value. This run also happened to produce results that are in close agreement with known steady state Nu correlations, which cannot be said for most of the experimental runs which feature much smaller values of b^*. The wall temperature fluctuations for Run #9 were sub-optimally small and on the same order magnitude as the thermocouple uncertainty. Despite this shortcoming, the run produced the most significant results, and the signal-to-noise ratio can be significantly improved in future experiments by increasing the magnitude of power fluctuations. The power output sinusoid was set at a 1000 W baseline value with an amplitude of 1000 W. As noted previously, the maximum power output of the heater is 9 kW.

Measurements, Thermophysical Properties and Uncertainty

The data reduction procedure accounted for error propagation due to measurement uncertainties, including a ±1°C for T-type thermocouples, 1% of the Coriolis Mass Flowmeter reading, and 10% uncertainty for Dowtherm A fluid property correlations listed in Table 4.9. The relative temperature error for the Prandtl ranges covered is, at most, 3%. All fluid properties are evaluated at the film temperature, defined as the average of the local bulk and wall temperatures. The thermocouple limit of error is assumed to be an absolute error. The standard deviation of the thermocouple error is calculated assuming a normal distribution. We estimate that the thermocouple absolute error of ±1 overestimates the

Run	Reynolds Range	Prandtl Range	Time Stamp
1	(2000, 4700)	(20, 41)	0508P-111921
2	(2500, 3800)	(23, 33)	0344P-111921
3	(2000, 4600)	(20, 40)	0405P-111921
4	(2400, 9500)	(11, 35)	0419P-111921
5	(4100, 13000)	(9, 23)	0435P-111921
6	(2100, 4600)	(20, 40)	0519P-111921
7	(2100, 4600)	(20, 40)	0528P-111921
8	(2100, 4400)	(21, 38)	0452P-111921
9	(2300, 4100)	(22, 36)	0458P-111921

Table 4.7: Reynolds and Prandtl ranges for each run. The ranges shown in this table are slightly exaggerated because the thermophysical properties were evaluated at the bulk fluid temperature rather than the film temperature. Accurate experimental Prandtl and Reynolds numbers, using thermophysical properties evaluated at the film temperature, are provided in Figure 4.15 and plots corresponding to the remaining runs in Appendix A. Additional data will be collected and analyzed as none of the initial experimental runs satisfy the condition that the Reynolds number range must not exceed the laminar flow regime (Re$<$ 2300). Nonetheless, Run #9 was found to be an adequate approximation of quasi-steady state heat transfer in terms of the oscillation frequency used and the experimentally measured Nu numbers were found to be a good match to steady state forced convection correlations for laminar flow despite the test moderately exceeding the applicable Reynold number range. The Prandtl number range for Run #9 is higher than the Prandtl range of interest for prototypical molten salt. The present work demonstrates the potential for periodic forcing under quasi-steady state conditions to match steady state predictions. This provides the motivation to collect additional experimental data to cover the entire Prandtl and Reynolds ranges of interest.

error in the context of the SHEFRA experiment. The error in the absolute temperature measurement is of lesser importance to the measurement of the change in temperature over each cycle. This depends on the DAQ's ability to measure the change in voltage coming from the thermocouple, not the absolute voltage. All thermocouples were calibrated using the LabVIEW calibration wizard. The calibration procedure required the use of a calibrated reference RTD probe and consisted of a zero-point calibration step in an ice bath, followed by a heated Duratherm G oil bath covering the temperatures of operation. After calibration, negligible deviations in temperature were observed between the thermocouples during initial isothermal testing at various temperatures. SHEFRA uses Therminol VP-1 which the same composition as Dowtherm A, and it is assumed that there are negligible differences in properties for the covered range of temperatures. Copper specific heat and density were assumed constant with negligible uncertainty. The axial locations of the wall temperature

Run	a^*	b^*	c^*	θ_∞	Time Stamp
1	0.78	32.32	(0+32.32j)	(-1.36-1.36j)	0508P-111921
2	0.78	38.82	(0+38.82j)	(-2.83-2.83j)	0344P-111921
3	0.78	38.82	(0+38.82j)	(-1.41-1.41j)	0405P-111921
4	0.79	40.08	(0+40.08j)	(-1.40-1.40j)	0419P-111921
5	0.81	41.28	(0+41.28j)	(-2.11-2.11j)	0435P-111921
6	0.78	48.49	(0+48.49j)	(-1.47-1.47j)	0519P-111921
7	0.78	48.49	(0+48.49j)	(-1.46-1.46j)	0528P-111921
8	0.78	64.68	(0+64.68j)	(-1.65-1.65j)	0452P-111921
9	0.78	96.97	(0+96.97j)	(-2.02-2.02j)	0458P-111921

Table 4.8: Experimental runs and dimensionless parameters.

thermocouples which correspond to the five nodes are assumed have an uncertainty of of 1/4". The CO3-T variant of the Omega Fast-Response Thermocouples with "Cement-On" Pads features a 0.010" bead diameter with a 300 ms response time and conforms to Standard Limits of Error. The response time or time constant is the time required to reach 63.2% of an instantaneous temperature. An adjustment of 200 ms was made to the collected data to correct for this. The data acquisition sampling rate was set to 10 Hz. Uncertainties associated with the time derivative calculation were not accounted for.

The Python `uncertainties` package is used to calculate the standard deviation of mathematical expressions through the linear approximation of error propagation theory.

Property	Temperature-dependent correlation [K]	Units
Density	$\rho = -8.91977 \times 10^{-1} \cdot T + 1.3261 \times 10^{3}$	kg/m^3
Specific Heat	$c_p = 2.79813 \cdot T + 7.54676 \times 10^{2}$	J/(kg K)
Thermal Conductivity	$k = 1.85606 \times 10^{-1} - 1.60002 \times 10^{-4} \cdot T$	W/(m K)
Dynamic Viscosity	$\mu = 4.31224 \times 10^{-6} \cdot \exp\left(2021.208061/T\right)$	Pa s

Table 4.9: Dowtherm A fluid temperature-dependent property correlations based on data in the 298 and 500 K range. [21, 26]

4.7 Results

The dimensionless parameters that describe Run # 9 in Tables 4.6 and 4.8 indicate that the system is converging to quasi-steady state. To provide context to interpret results, the steady state values for thermally developing laminar flow from the correlations in Chapter 3 are shown in Table 4.14. The steady state predictions are evaluated at five Re and Pr combinations that track the experimental Pr and Re domain ranging from a low-end Re and

Property	Temperature-dependent correlation [°C]	Units
Density	$\rho = 2279.92 - 0.488 \cdot T$	kg/m^3
Specific Heat	$c_p = 2415.78$	J/(kg ° C)
Thermal Conductivity	$k = 0.7662 + 0.0005 \cdot T$	W/(m °C)
Dynamic Viscosity	$\mu = \frac{4.638 \times 10^5}{T^{2.79}}$	Pa s

Table 4.10: Flibe temperature-dependent property correlations based on data in 600 to 800 $°C$ range. [21, 26]

Table 4.11: Copper thermophysical properties at 300 K. [6]

Property	Value	Units
Density	ρ =8933	kg/m^3
Specific Heat	$c_p = 385$	J/(kg K)

high-end Pr flow to a high-end Re and low-end Pr. The measured Nu values for the node at the inlet for are plotted as a function of Re and Pr in Figure 4.9a. The measurements with the lowest standard deviation are in agreement with the steady state prediction of Nu=24, with a measured value of 25±5. Ignoring measurements with uncertainties higher than 11, the measured Nu is bounded by an upper value of 33±11 which corresponds to the prediction for thermally developing flow with a constant flux boundary condition. The Nu number variation for the T-1 and T-2 nodes in Run #9 appear to be correlated with Pr however the variation is actually correlated to when the measurement is taken on the oscillation cycle. The Nu number is seen to increase as the difference between the wall and bulk temperature approaches zero. This can be observed in a plot of the Nu as a function of time in Figure 4.14b. Relying on the analysis in Chapter 3 we hypothesize that this is a distortion to quasi-steady response from natural convection to ambient air causing a non-zero heat flux when the temperate gradient is zero. Other external factors that may have contributed to the deviations in Nu include parasitic heat loss due to the metal fitting and connected structures that cradle the test-section and that are co-located with the inlet node. Errors are also introduced due to the data logging sampling rate, and due to the wall temperature thermocouple time constants. The uncertainty quantification procedure does not account for these errors. Amending the uncertainty quantification will increase the number of data points omitted near the intersection point in the bulk and wall temperatures. Mitigation strategies to dramatically reduce the relative magnitude of these errors are outlined in the discussion (Section 4.8).

Similar conclusions can be made for nodes T-2 to T-5, shown in Figures 4.9a though Figure 4.13b. Acceptable agreement is observed between Nu measurements on the lowest-end of uncertainty error and steady state predictions for thermally developing flow. Table 4.12 is a selection of experimental Nu measurements with the lowest uncertainty compared

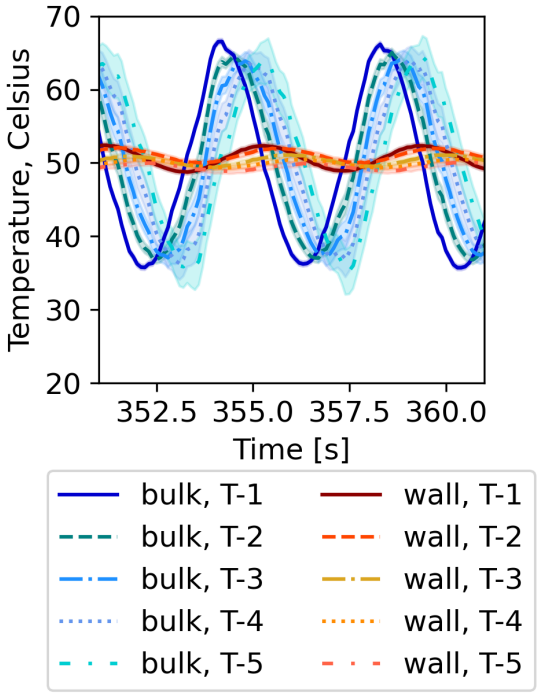
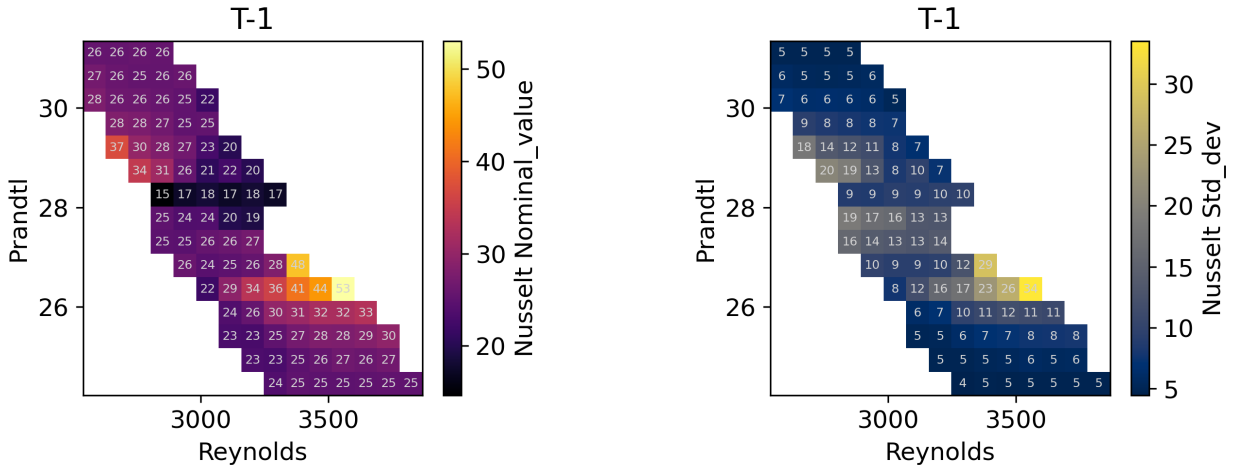


Figure 4.8: Experimental temperature measurements for channel nodes with uncertainty represented as a shaded area. T-1 is the node located at the duct inlet, and T-5 is situated at the outlet. Reynolds number range: (2300, 4100). Prandtl number range: (22, 36).

with steady state predictions. Due to filtering of Nu measurements of a standard deviation greater than 80% of the nominal value, Nu measurements of nodes further downstream of the inlet become progressively sparse in the the grid representations shown. This propagation of uncertainty is due to the finite difference approximation for estimating the local fluid temperature at a given node. To calculate the Nu number at T-5, the data reduction code first calculates the heat transfer coefficient at the inlet, and uses that result to solve at the adjacent node, and so on until it arrives at T-5. The standard deviation of uncertainty error must be referenced when assessing the validity of a nominal value in the grid figures. These higher uncertainty values were not omitted to demonstrate this trend.

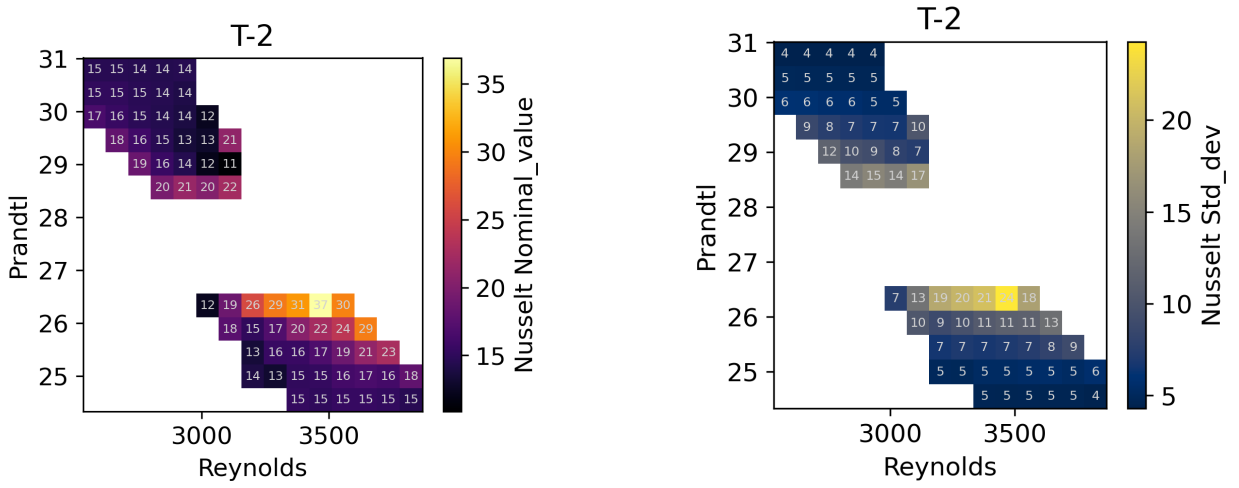
The distribution of uncertainty error of the Nu, Re and Pr numbers as a function of their nominal values are presented in Figures 4.16, 4.17a and 4.17b at each node.

Averaged Nu measurements from all 9 experimental runs as a function of Pr and Re at each node can be found in Figures 4.18a to 4.19d. A general trend observed is higher measured Nu numbers for lower values of Pr and higher values of Re. Since the experimental runs included are not strictly in quasi-steady state, and the flow regimes often cycled between laminar, transitional and turbulent flow regimes no significant conclusions can be derived from the overall data for the objective at hand.



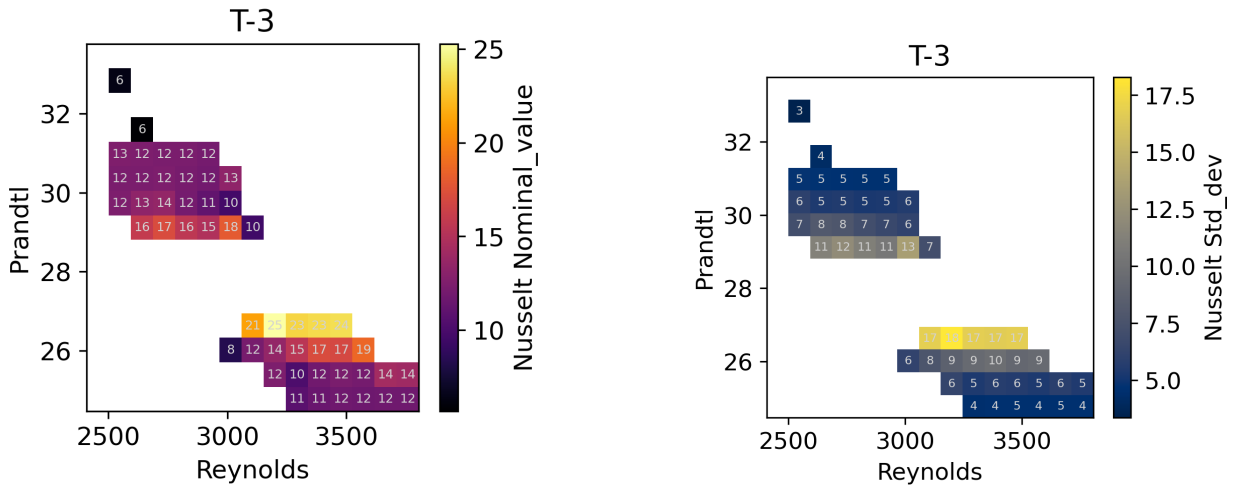
(a) (b)

Figure 4.9: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-1. (f =0.25Hz, T_o =50°C, ΔT_o =15°C, 76kg/h, timestamp label: 0458-111921)



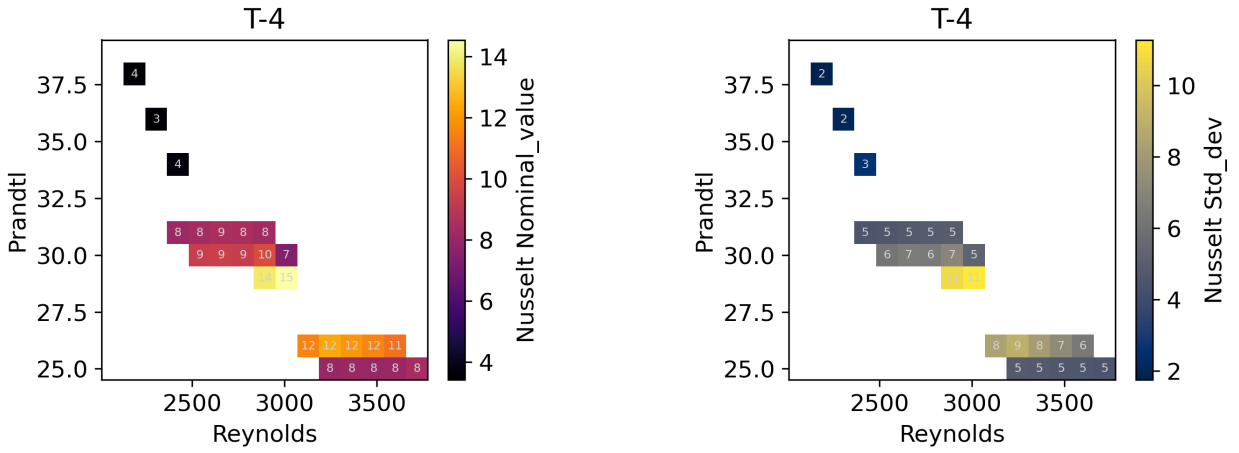
(a) (b)

Figure 4.10: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-2. (f =0.25Hz, T_o =50°C, ΔT_o =15°C, 76kg/h, timestamp label: 0458-111921). Node location T-2 is downstream of T-1. The wall thermocouple measurement nodes are equidistant along the 2 meter vertical copper tube, with one thermocouple on each of the two ends.



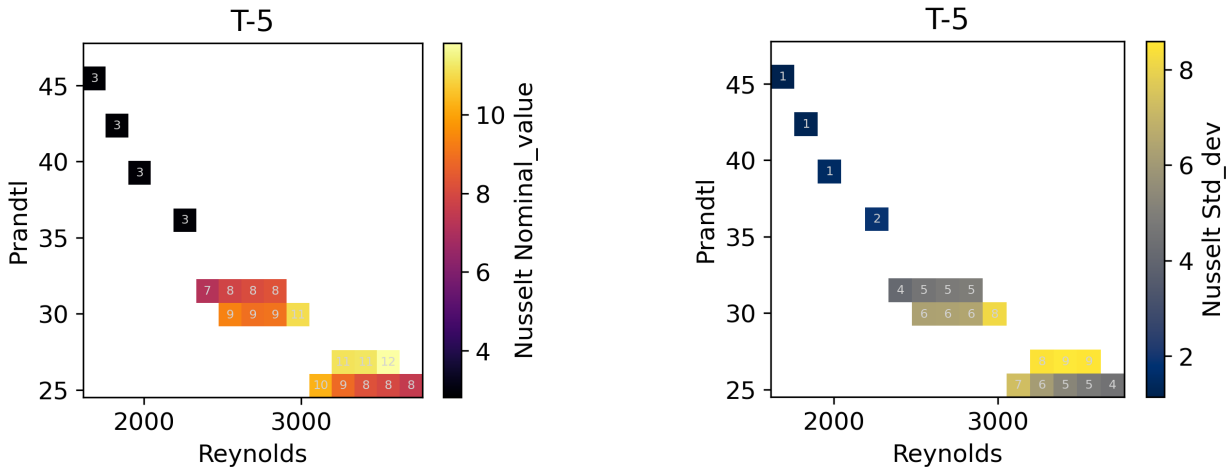
(a) (b)

Figure 4.11: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-3. (f =0.25Hz, T_o =50°C, ΔT_o =15°C, 76kg/h, timestamp label: 0458-111921)



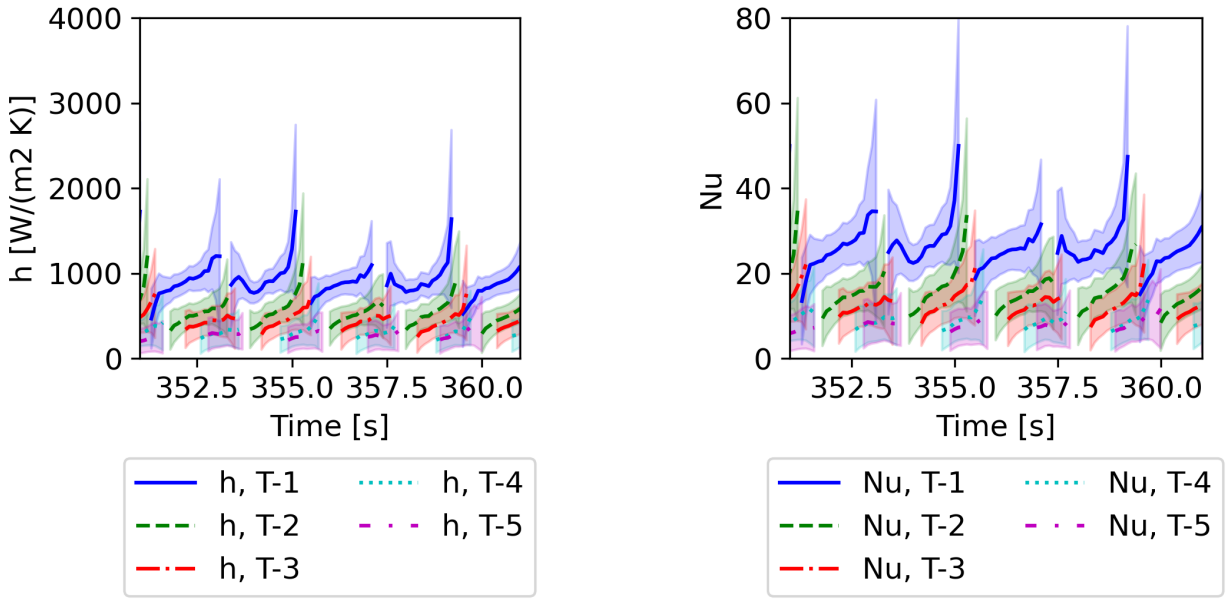
(a) (b)

Figure 4.12: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-4. (f =0.25Hz, T_o =50°C, ΔT_o =15°C, 76kg/h, timestamp label: 0458-111921)



(a) (b)

Figure 4.13: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-5. (f =0.25Hz, T_o =50°C, ΔT_o =15°C, 76kg/h, timestamp label: 0458-111921)



(a) (b)

Figure 4.14: Heat transfer coefficient and Nusselt measurements for experimental run (f =0.25Hz, T_o =50°C, ΔT_o =15°C, 76kg/h, timestamp label: 0458-111921). Reynolds number range: (2300, 4100). Prandtl number range: (22, 36).

Table 4.12: Low uncertainty experimental quasi-steady Nu numbers compared to steady state Nusselt number predictions for various flow regimes. The distances correspond to the axial locations of the measurement nodes, T-1 to T-5, relative to the inlet. Upper and lower bound experimental Nu values with low relative uncertainty are selected. The values are an average of test data within a limited Pr and Re range. This approach provides more resolution compared to taking an overall average that will be skewed towards outlier data. The trade-off is that the selected measured values represent a small fraction of the entire data set. The experimental parameters and instrumentation can be optimized to increase the percentage of usable data.

Re	Pr	Nu					Flow Type
		0.029 m	0.528 m	0.988 m	1.484 m	1.960 m	
3600	30	24	9	7	6	5	T.D., Iso. Wall
		34	11	9	8	7	T.D., Const. Flux
2500-4000	25-31	25±5	11±7	6±3	4±2	3±1	Exp. (lower-end)
		33±11	21±8	12±5	8±5	8±4	Exp. (upper-end)

4.8 Discussion

The experimental measurements are in agreement with the transient model predictions that quasi-steady state is achieved at high frequencies, corresponding to higher values of b^*. However, quasi-steady state can also be approximated in both limits as b^* approaches zero and infinity. Therefore low frequencies should also be able approach quasi-steady heat transfer, with the largest deviations in Nu number results are expected in between high and low values of b^*. Sparrow and Farias arrived at this conclusion after applying a quasi-steady model to a solution for periodically varying inlet temperature between participating parallel plates. This can be reasoned by imagining what is happening to the thermal boundary layer. At the limit of low frequencies, the thermal boundary is not changing much over time, satisfying quasi-steady state conditions. At the limit of a very high frequency, the thermal boundary layer is unable to shift substantially, due to the rapid cycling, and on average the system converges to a quasi-steady state.

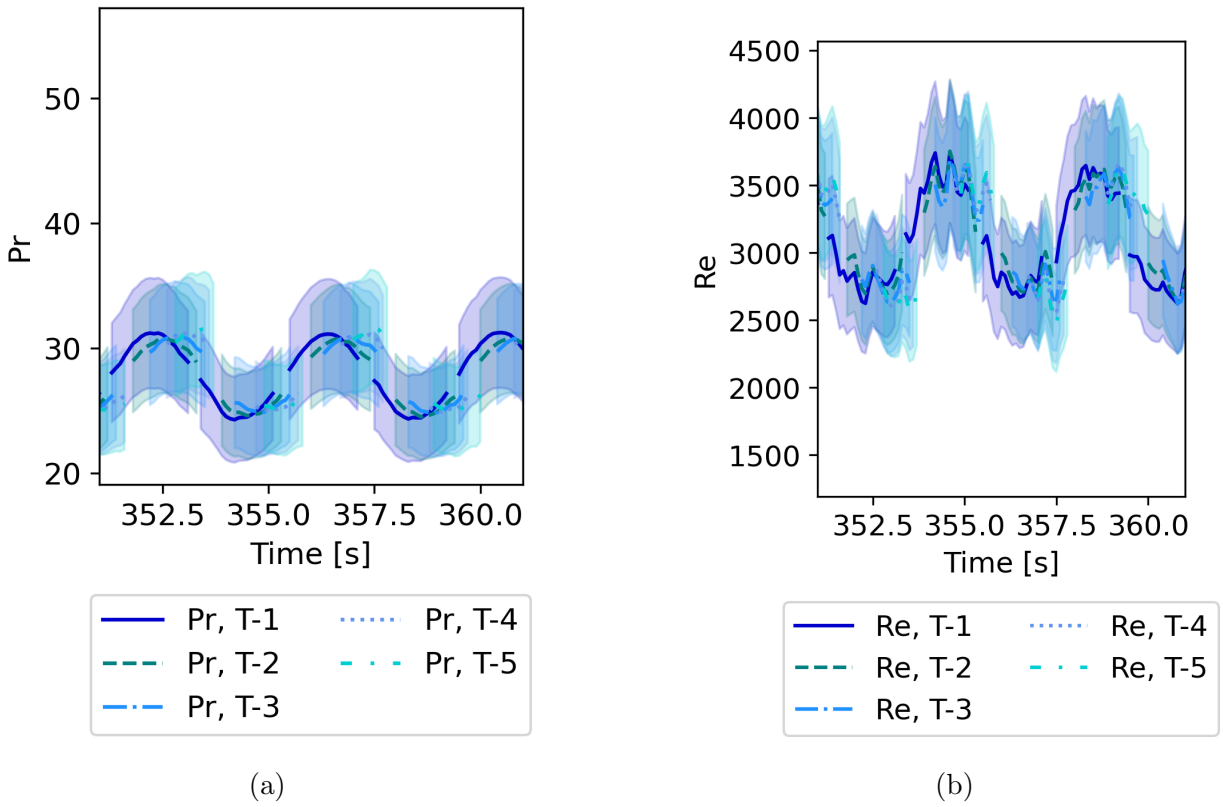
An experiment run at the limit of a very low frequency provides a smaller delta between fluid and solid temperatures and the potentially for more noise. The lowest frequency run for SHEFRA was at 0.0833 Hz, and Figure A.23b shows that at least at the inlet node, the Nu number is in close agreement with steady state predictions for constant flux. This is in contrast to the high frequency run (0.25 Hz) that more closely match steady state predictions for uniform wall temperature. Using the SHEFRA transient model to analyze a variant of Run #1 (0.0833 Hz) in the limit of a very low frequency such as 0.01 Hz, we find that the predicted Nu numbers at each node are 49, 12, 9, 8, and 7, respectively. Each

Table 4.13: Comparison of percent errors for low uncertainty experimental quasi-steady Nu numbers compared to steady state Nusselt number predictions for the covered flow regimes. The lower-end and higher-end of experimental measurements at each node for Run #9 are abbreviated as L.E. and H.E., respectively. The steady-state predictions for thermally developing laminar flow with isothermal wall and constant heat flux boundary conditions are abbreviated as I.W. and C.F., respectively. Referring to Figures 4.9a though Figure 4.13b, we see a distribution of experimental Nu numbers that is concentrated at the upper and lower bounds of the measurement data. The pattern that emerges is that the high end measurements more closely match the steady state thermally developing Nu predictions with a constant flux boundary condition, and low end measurements generally more closely match the predictions for an isothermal wall temperature boundary condition. The experimental measurements overlap most in the range between the constant flux and isothermal wall steady state predictions, indicating general quasi-steady state heat transfer. The uncertainties are likely exaggerated here due to the conservative temperature measurement error uncertainty.

Nu Percent Error					Comparison
0.03 m	0.53 m	0.99 m	1.48 m	1.96 m	
4%±21%	22%±78%	-14%±43%	-33%±33%	-40%±20%	L.E. to I.W.
-26%±32%	0%±64%	-33%±33%	-50%±25%	-57%±14%	L.E. to C.F.
38%±46%	133%±89%	71%±71%	33%±83%	60%±80%	H.E. to I.W.
-3%±32%	91%±73%	33%±56%	0%±63%	14%±57%	H.E. to C.F.

of those Nu number predictions except for that of the inlet node is a very close match to the steady state predictions for uniform wall temperature. This is consistent with Sparrow and Farias' conclusion that in all cases, quasi-steady state by a transient model of a similar construction is better approximated at locations downstream of the inlet. Looking at the SHEFRA experimental results, it appears that higher frequencies are able to better approximate quasi-steady near the inlet which is an intuitive result. The measured Nu numbers for the downstream nodes during Run #1 (0.0833 Hz) persist at higher values compared to steady state predictions, which could be due to a transition from laminar flow as the Re approaches 5000.

While the Nu data collected during the single 0.25 Hz run was observed to match the steady state analytical solutions for laminar flow, the test only covered Pr values from 22 to 34. The preliminary agreement observed in the results shown provide an incentive to collect additional data for Pr numbers 8 to 22, and Re numbers 100 to 10,000. After determining the optimal baseline and power settings for sections of Pr, the experiment can be repeated for multiple mass flow rate to sweep through the entire domain of interest. Special care must be taken to avoid a flow regime transition within a single oscillation cycle to ensure predictable results. Experiments for forced convection of Flibe in a smooth tube at ORNL



(a) (b)

Figure 4.15: Prandtl and Reynolds measurements for experimental run (f =0.25Hz, T_o =50°C, ΔT_o =15°C, 76kg/h, timestamp label: 0458-111921)

determined that Flibe behaved like normal fluids. For this reason, we anticipated that additional Dowtherm A experiments will show steady state thermal hydraulic similitude between the surrogate fluid and Flibe. This conclusion does not upend any existing expectations, and attention should be instead be drawn to how the frequency domain derived analytical model has provided the tools to intelligently select experiment parameters that will produce data with optimally high signal-to-noise ratio. To be more specific, the analytical model solves the problem of determining which combination of channel material, inner radius, wall thickness, length, mass flow rate, mean cycle temperature, cycle amplitude, and oscillation frequency, produce data with the lowest uncertainty in conjunction with Dowtherm A between a fixed range of temperatures. Further, the model can be used to determine if the system is in quasi-steady state. The analytical model predictions of Run #9 can be used to demonstrate this. Although the run was conducted before the model was developed, the steps required to increase the accuracy of experimental measurements are easily apparent.

It is useful to separate the application of the analytical model to the objective of collecting Nu measurements into two categories. The model can be used to determine if the

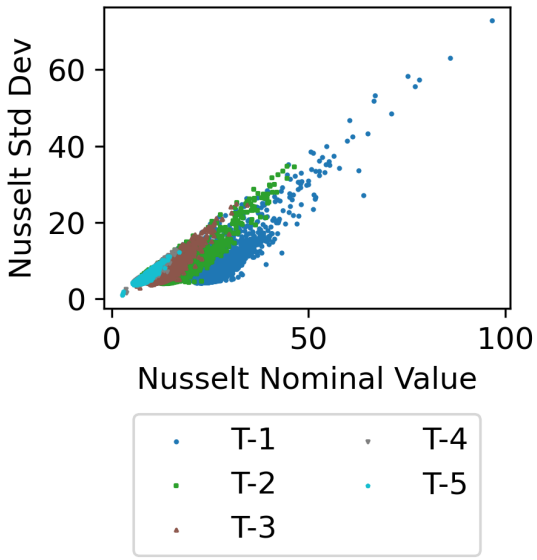
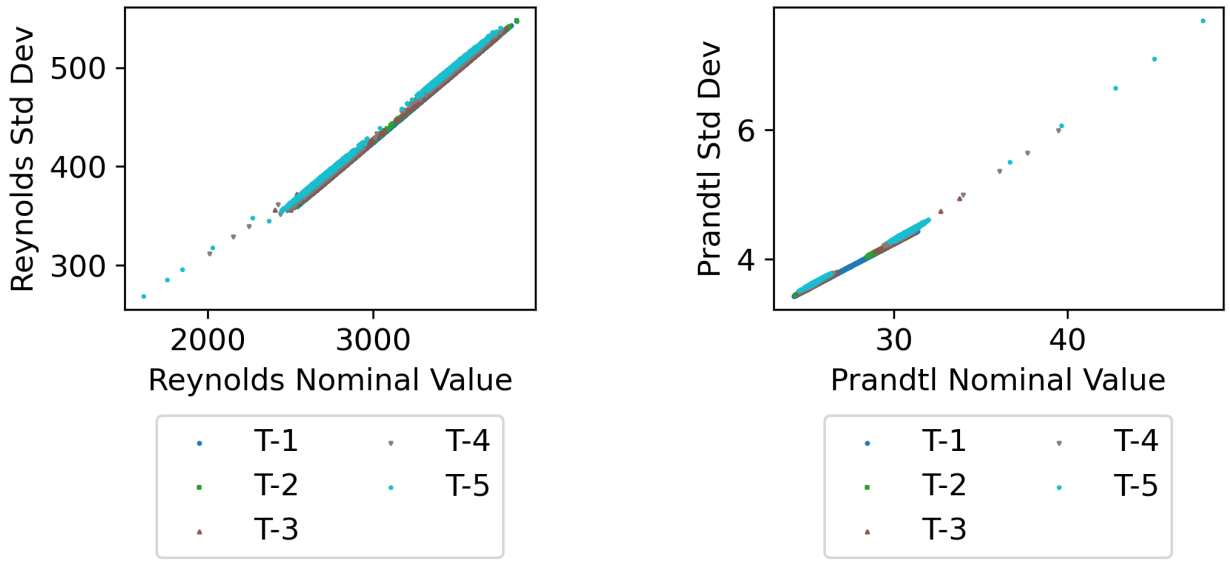


Figure 4.16: Nusselt uncertainty distribution for range of nominal values. (f =0.25Hz, T_o =50°C, ΔT_o =15°C, 76kg/h, timestamp label: 0458-111921)



(a) (b)

Figure 4.17: Reynolds and Prandtl uncertainty distributions for range of nominal values. (f =0.25Hz, T_o =50°C, ΔT_o =15°C, 76kg/h, timestamp label: 0458-111921)

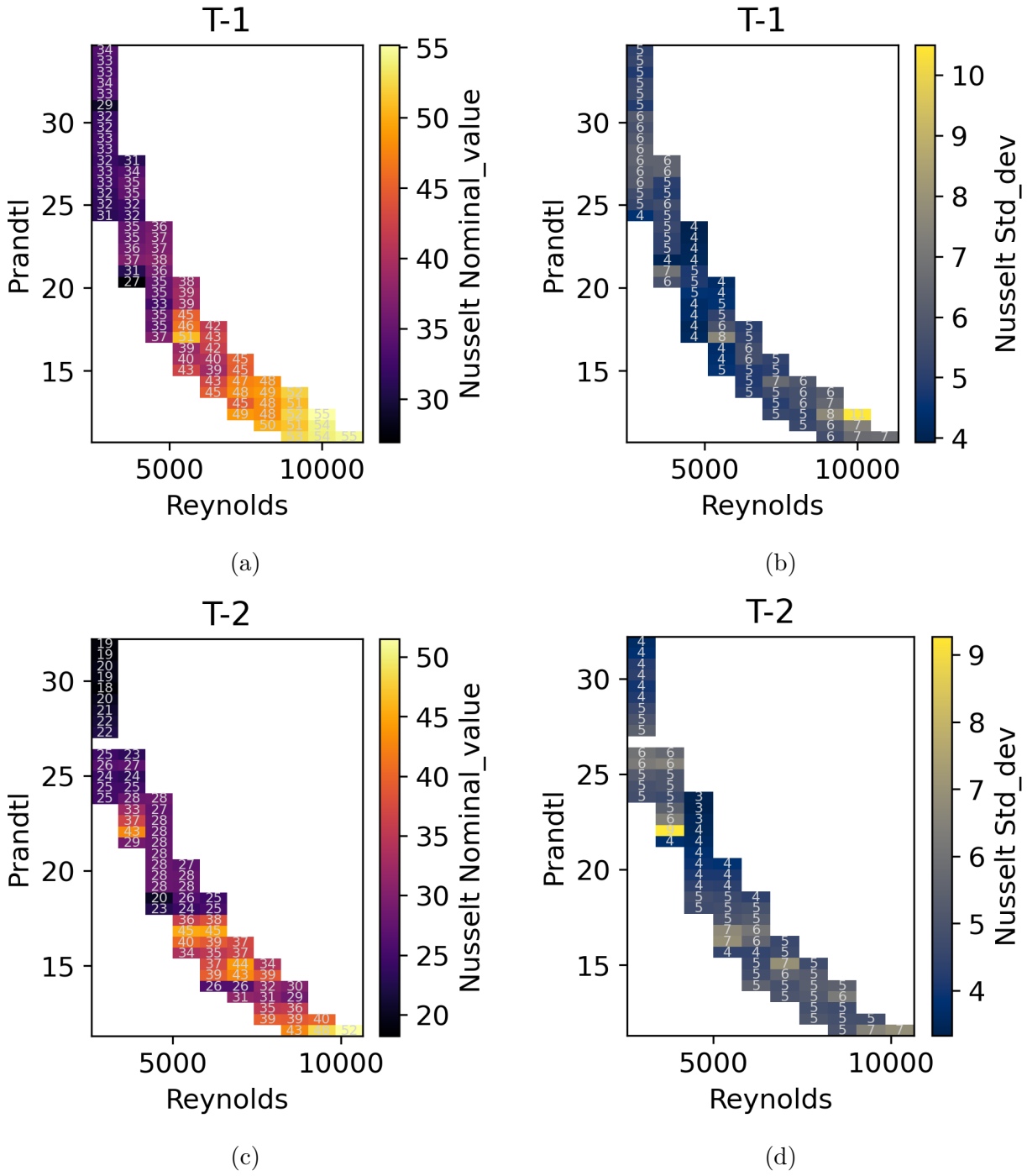
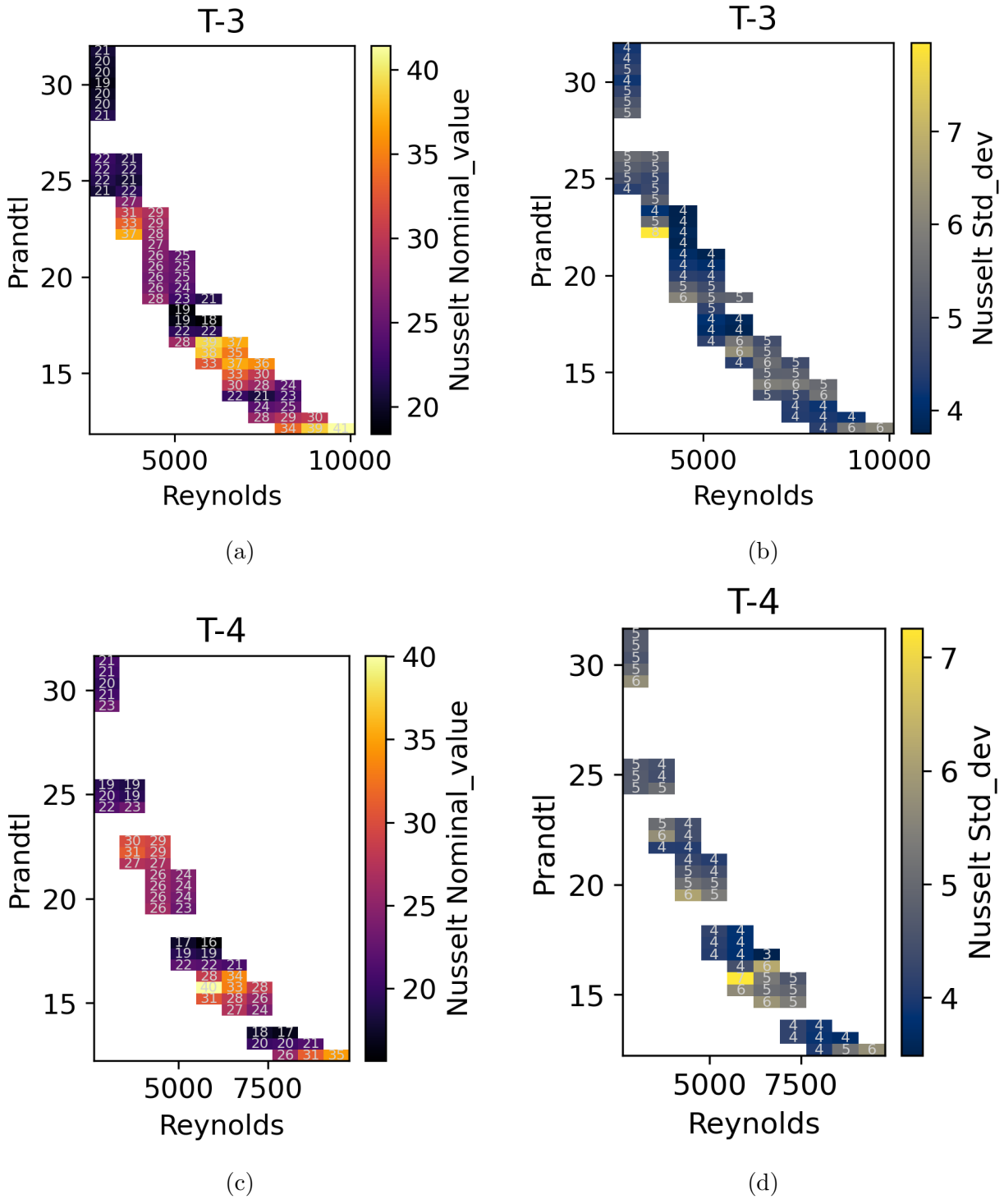


Figure 4.18: Nusselt number measurements and associated uncertainty for the series of experimental runs summarized in Table 4.6 at node location T-1 (inlet) and T-2. Measurements with an uncertainty greater than 25% were omitted from these plots.



(a) (b) (c) (d)

Figure 4.19: Nusselt number measurements and associated uncertainty for the series of experimental runs summarized in Table 4.6 at node location T-3 and T-4. Measurements with an uncertainty greater than 25% were omitted from these plots.

Table 4.14: Steady state Nusselt number predictions for various flow regimes. The combined entrance region prediction is unlikely to be observed in experiment due to a mesh screen inserted upstream of the inlet for mixing. The distances correspond to the axial locations of the measurement nodes relative to the inlet.

Re	Pr	Nu					Flow Type
		0.029 m	0.528 m	0.988 m	1.484 m	1.960 m	
3600	30	24	9	7	6	5	T.D., Iso. Wall
		34	11	9	8	7	T.D., Const. Flux
		9	8	8	8	8	Combined
				273			Turbulent
6000	20	24	9	7	6	5	T.D.,Iso. Wall
		36	11	9	8	7	T.D., Const. Flux
		9	8	8	8	8	Combined
				322			Turbulent
9000	14	25	9	7	6	6	T.D.,Iso. Wall
		36	11	9	8	7	T.D., Const. Flux
		9	8	8	8	8	Combined
				346			Turbulent
10000	13	25	9	7	6	6	T.D.,Iso. Wall
		37	11	9	8	8	T.D., Const. Flux
		9	8	8	8	8	Combined
				357			Turbulent
14000	11	27	10	8	7	6	T.D.,Iso. Wall
		40	12	10	9	8	T.D., Const. Flux
		9	8	8	8	8	Combined
				420			Turbulent

hypothetical or physical experiment will be in a quasi-steady state. A system in quasi-steady state has a thermal boundary layer than does not change with time, and therefore the Nu value is expected to remain constant. Secondly, the experimental parameters can be tuned to produce smaller uncertainties.

Quasi-steady state predictions

The model-predicted Nu is equal to 36 for when the dimensionless parameter $b^* = 97$, which is the case for a frequency of 0.25 Hz. As we increase b^* we find the Nu converges to 31 as the inlet frequency is increased. This value is about midway between steady state analytical

solutions' predictions of 34 for constant flux and 25 for isothermal wall temperature found in Table 4.14. Given that the wall temperature is oscillating it would be reasonable to assume that the quasi-steady prediction and the experimental values would match the constant heat flux steady state analytical solution prediction. However, the wall temperature oscillations in this case are much smaller in magnitude compared to the fluid temperature oscillations. This leads to the likely conclusion that the quasi-steady state prediction will vary between the constant heat flux and isothermal wall steady state predictions depending on the ratio of oscillation amplitudes between the fluid and wall. The model assumes constant properties, and the predictions evaluated in the table are for a fixed Re and Pr value that is representative of the experimental run; this is expected to introduce a limited amount of error. This factor aside, increasing the inlet frequency would make the system a better approximation of quasi-steady state. For the SHEFRA test-section, increasing the inlet oscillation frequency provides rapidly diminishing returns in terms of reaching model-predicted Nu convergence. Doubling the frequency to 0.5 Hz, and again to 1 Hz would lead to a Nu of 31.5 and 31.1, respectively. While the current setup's maximum frequency of 0.25 Hz provides an acceptable approximation of quasi-steady state, this could be improved by modifying the tube geometry to increase b^*. Increasing the b^* parameter without increasing the inlet frequency avoids over dampening the wall temperature oscillations.

Model disrepancies

The Sparrow and Farias model showed deviations in the Nu number results and discontinuities when the wall and bulk temperatures are equal, but the heat flux was not zero. This trend was observed in Chapter 3, but only in the presence of heat transfer to ambient air, dictated by $\mathrm{Nu_o}$.

Reduction of uncertainties

If the wall temperature oscillations are dampened too far, thermocouple measurement error will lead to large propagated uncertainties. Without significantly modifying the test-section, this is best mitigated by increasing power. Alternatively, the geometry of the test-section may be modified to achieve the same effect. Since b^* is proportional to r/l, increasing the radius and decreasing the wall thickness slightly can be a substitute of higher inlet frequencies.

In general, error and uncertainty associated with thermocouple measurements can be minimized by meeting the following criteria.

1. Ensure a large delta between wall and fluid temperatures, $\theta_b - \theta_w$), relative to the measurement uncertainty. See Figure 4.21.

2. Ensure that the amplitudes for both the cycle fluid and wall temperatures are large compared to the measurement uncertainty.

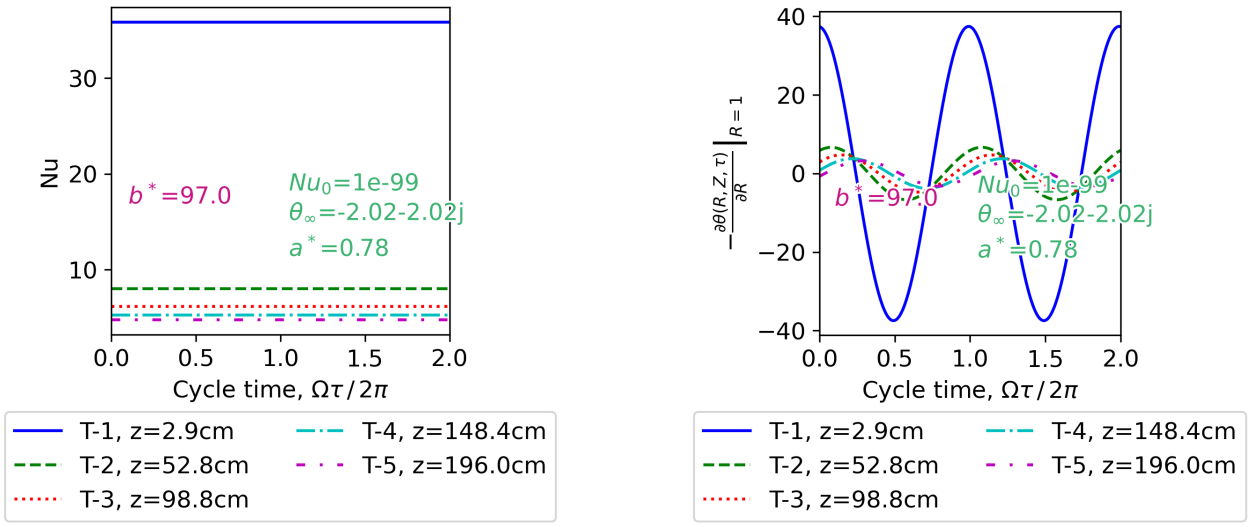
As noted above, the uncertainty in measurement of the change in temperature is lower than the uncertainty in the measurement of the absolute temperature.

Depending on the DAQ hardware and control cycle speed, a lower inlet frequency can lead to smaller time sampling errors and less error from thermocouple time constants. Instrumentation improvements that would reduce error and uncertainty include the use of thermocouples with time constants on the order of 1 to 10 ms. T-type thermocouples that conform to the Special Limits of Error are also available, which have an error of ± 0.5. Installing more wall thermocouples, especially near the inlet would reduce the propagation of error due to the finite difference method. If possible, it would be desirable to thermally insulate the inlet of the test-section from the metal structures that support it to mitigate parasitic heat loss distortion. Error introduced due to parasitic heat loss is suspected in the high power (lower frequency) runs in Appendix A especially at the first node. It is possible that the low power used for the experimental run #9 presented in this chapter is the reason that the parasitic heat loss has not created a large distortion. Higher power is needed to amplify wall temperature oscillation, and to reach lower Pr numbers, but this should not come at the cost of distorted wall temperature near the inlet. Such an error at the inlet is propagated through to other nodes. Additional experimental and theoretical investigations are suggested to confirm this and to rule out model error. With a long channel, it would be advantageous to be able to non-intrusively measure the bulk fluid temperature along the channel. This would anchor any finite difference approximations made to the physical system.

4.9 Future work

Using the recommendations outlined in the Discussion section, additional data will be collected to cover the entire Pr and Re range of interest. For laminar flow, a direct comparison with ORNL salt data would ideally require the SHEFRA channel to be a little over 1 m longer in order to obtain the heat transfer coefficient integrated over the entire entrance region. Alternatively, a curve can be fitted to the available measurements and axially extrapolated beyond the physical channel without incurring too high of an uncertainty penalty. The uncertainty analysis can be improved by accounting for time sampling errors. This can be done by using the samples before and after a measurement to scale the timing uncertainty that is inherent to the experimental set-up.

Turbulent flow is of significant interest in the application of molten salt reactors. The frequency scaled analytical model for laminar flow can be adapted for turbulent flow. The model would be used to provide guidance on the optimal set of experimental parameters for quasi-steady operation and for low uncertainty. The results of these experiments could then be compared to steady state turbulent flow empirical correlations and to the salt forced convection experiments such as the mentioned ORNL study. Turbulent flow data collected without the appropriate analytical framework is not guaranteed to be in quasi-steady state. At a minimum, a simplified, low fidelity criteria for turbulent flow quasi-steady state can be



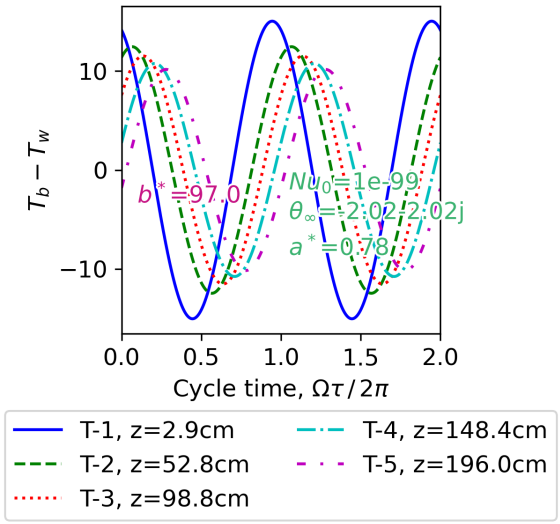
(a) (b)

Figure 4.20: Transient laminar model-predicted Nusselt numbers and dimensionless heat flux for experimental run (f =0.25Hz, T_o =50°C, ΔT_o =15°C, 76kg/h, timestamp label: 0458-111921)

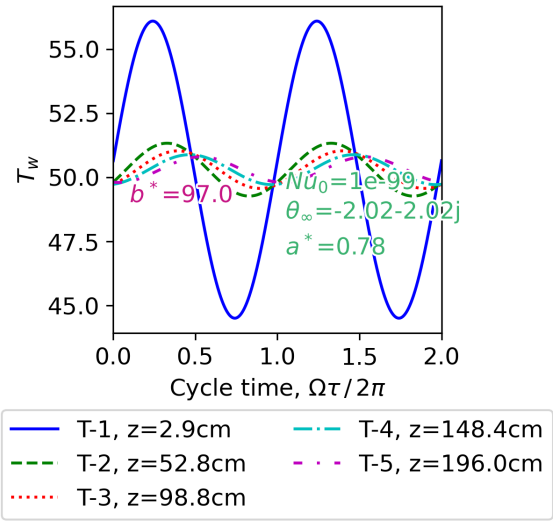
derived, or trends from the laminar model can be extrapolated.

The above steps would provide the steps to conclusively demonstrate steady state thermal hydraulic similitude between the surrogate heat transfer oil and Flibe. Theoretically, the similitude should extended to unsteady or non-quasi steady flows. If proving unsteady similitude is deemed to be important for design purposes or safety-related assessments, this could be proved experimentally with unsteady forced convection Flibe experiments using the same frequency response methods.

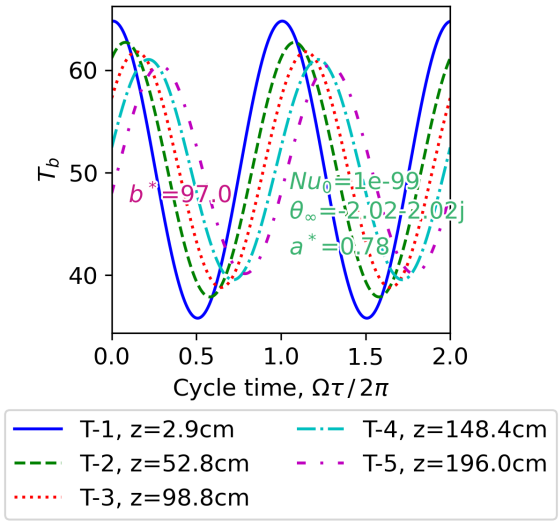
Finally, there is a pathway for extending the experimental methods used for more complex inlet boundary conditions that can be approximated as a Fourier series.



(a) Transient model-predicted delta between wall and local bulk temperature.



(b) Transient laminar model-predicted wall temperature.



(c) Transient laminar model-predicted bulk temperature.

Figure 4.21: Bulk temperature and wall temperature measurements and transient laminar model predictions for experimental run (f =0.25Hz, T_o =50°C, ΔT_o =15°C, 76kg/h, timestamp label: 0458-111921)

Chapter 5

Frequency Response Parameter Estimation

A simplified model of the SHEFRA experiment is constructed in the form of a transfer function that describes the system response to a forcing function input. Additional layers of model complexity are added incrementally, including nonlinearity due to temperature dependent thermophysical properties. This approach provides the ability to verify more complex, higher order transfer function models. For example, the response at the inlet of the channel in a model with axial spatial dependence should match the response for the spatially-independent model.

5.1 Spatially-Independent Bulk Temperature

To determine the transient response in a channel wall to a periodically-varying bulk fluid temperature, we perform an overall energy balance on the solid. In this case, the balance relates the rate of heat loss at the wall to the rate of change of the internal energy. We assume that the channel wall can be approximated as a lumped capacitance, physical properties are constant, and the bulk fluid temperature is the same everywhere along the channel. In the physical experiment, the input signal is dampened as it travels downstream.

$$-\dot{E}_{out} = \dot{E}_{st} \tag{5.1}$$

$$-hA_s(T_w - T_b(t)) = (\rho c_p)_w V \frac{dT_w}{dt} \tag{5.2}$$

Rearranging, and employing a specific volume term, a_v, we get

$$\frac{\partial T_w}{\partial t} = -\frac{ha_v}{(\rho c_p)_w}(T_w - T_b(t)) \tag{5.3}$$

a_v for a cylindrical channel is obtained using the per unit length volume and surface area of a channel wall element.

$$\begin{aligned} a_v = \frac{A_s}{V} &= \frac{2\pi a\, dz}{\left[\pi(l+a)^2 - \pi(a)^2\right]\, dz} \\ &= \frac{2a}{l^2 + 2al} \end{aligned} \tag{5.4}$$

The periodic time-varying fluid temperature is represented by a sinusoidal function.

$$T_b(t) = T_o + \Delta T_o \sin(\Omega\tau) \tag{5.5}$$

The problem is now nondimensionalized to produce a solution that can be scaled between different systems. Conveniently, this also simplifies the the problem to be solved. The dimensionless wall temperature is put in terms of the fluid temperature cycle parameters.

$$\theta_w(\tau) = \left(T_w(t) - T_o\right)/\Delta T_o \tag{5.6}$$

$$\mathrm{Nu} = \frac{h(2a)}{k_f} \tag{5.7}$$

$$\tau = \alpha t/a^2 \tag{5.8}$$

$$\alpha = \frac{k_f}{\left(\rho c_p\right)_f} \tag{5.9}$$

$$\Omega = \omega a^2/\alpha \tag{5.10}$$

Replace h with Nu

$$\frac{\partial T_w}{\partial t} = -\frac{\mathrm{Nu}\, k_f a_v}{\left(\rho c_p\right)_w (2a)}\left(T_w - T_b(t)\right) \tag{5.11}$$

Expand bulk fluid temperature

$$\frac{\partial T_w}{\partial t} = -\frac{\mathrm{Nu}\, k_f a_v}{\left(\rho c_p\right)_w (2a)}\left(T_w - \left(\Delta T_o \sin(\omega t) + T_o\right)\right) \tag{5.12}$$

Non-dimensionalize time and frequency

$$\frac{\partial T_w}{\partial(\tau a^2/\alpha)} = -\frac{\mathrm{Nu}\, k_f a_v}{\left(\rho c_p\right)_w (2a)}\left(T_w - \Delta T_o \sin(\Omega\tau) - T_o\right) \tag{5.13}$$

$$\frac{\alpha}{a^2}\frac{\partial T_w}{\partial \tau} = -\frac{\mathrm{Nu}\, k_f a_v}{\left(\rho c_p\right)_w (2a)}\left(T_w - \Delta T_o \sin(\Omega\tau) - T_o\right) \tag{5.14}$$

Expand thermal diffusivity, α.

$$\frac{k_f}{\left(\rho c_p\right)_f a^2}\frac{\partial T_w}{\partial \tau} = -\frac{\mathrm{Nu}\, k_f a_v}{\left(\rho c_p\right)_w (2a)}\left(T_w - \Delta T_o \sin(\Omega\tau) - T_o\right) \tag{5.15}$$

Divide both sides by k_f.

$$\frac{1}{(\rho c_p)_f a^2}\frac{\partial T_w}{\partial \tau} = -\frac{\mathrm{Nu}\, a_v}{(\rho c_p)_w (2a)}(T_w - \Delta T_o \sin(\Omega\tau) - T_o) \tag{5.16}$$

Rearrange

$$\frac{\partial T_w}{\partial \tau} = -\frac{\mathrm{Nu}\,(\rho c_p)_f a\, a_v}{(\rho c_p)_w (2)}(T_w - \Delta T_o \sin(\Omega\tau) - T_o) \tag{5.17}$$

Non-dimensionalize wall temperature using Equation 5.6.

$$\frac{\partial(\Delta T_o \theta_w(\tau) + T_o)}{\partial \tau} = -\frac{\mathrm{Nu}\,(\rho c_p)_f a\, a_v}{(\rho c_p)_w (2)}((\Delta T_o \theta_w(\tau) + T_o) - \Delta T_o \sin(\Omega\tau) - T_o) \tag{5.18}$$

Simplify the right-side of the equation.

$$\frac{\partial(\Delta T_o \theta_w(\tau) + T_o)}{\partial \tau} = -\frac{\mathrm{Nu}\,(\rho c_p)_f a\, a_v}{(\rho c_p)_w (2)}(\Delta T_o \theta_w(\tau) - \Delta T_o \sin(\Omega\tau)) \tag{5.19}$$

Utilize the chain rule to simplify the left-side partial derivative.

$$\Delta T_o \frac{\partial \theta_w(\tau)}{\partial \tau} = -\frac{\mathrm{Nu}\,(\rho c_p)_f a\, a_v}{(\rho c_p)_w (2)}(\Delta T_o \theta_w(\tau) - \Delta T_o \sin(\Omega\tau)) \tag{5.20}$$

Divide both sides by ΔT_o

$$\frac{\partial \theta_w(\tau)}{\partial \tau} = -\frac{\mathrm{Nu}\,(\rho c_p)_f a\, a_v}{(\rho c_p)_w (2)}(\theta_w(\tau) - \sin(\Omega\tau)) \tag{5.21}$$

Define a new nondimensional parameter, where a is the characteristic length of the flow and a_v is the specific volume.

$$\alpha^* \equiv \frac{(\rho c_p)_f}{(\rho c_p)_w}\frac{a_v a}{2} \tag{5.22}$$

The final non-dimensionlized form of the equation becomes

$$\frac{\partial \theta_w(\tau)}{\partial \tau} = -\mathrm{Nu}\,\alpha^*(\theta_w(\tau) - \sin(\Omega\tau)) \tag{5.23}$$

with the initial condition,

$$\theta_w(0) = 0 \tag{5.24}$$

Take the Laplace transform of the equation. Starting with the left-side.

$$\mathcal{L}\left\{\frac{\partial \theta_w(\tau)}{\partial \tau}\right\} = s\tilde{\theta}_w(s) - \theta_w(0) \tag{5.25}$$

On the right side we have

$$\mathcal{L}\left\{-\mathrm{Nu}\,\alpha^*(\theta_w(\tau)-\sin(\Omega\tau))\right\} = -\mathrm{Nu}\,\alpha^*\left(\tilde{\theta}_w(s)-\frac{\Omega}{s^2+\Omega^2}\right) \tag{5.26}$$

The transformed equation becomes

$$s\tilde{\theta}_w(s) = \mathrm{Nu}\,\alpha^*\left(-\tilde{\theta}_w(s)+\frac{\Omega}{s^2+\Omega^2}\right) \tag{5.27}$$

This can be simplified further by expanding the right side of the equation.

$$s\tilde{\theta}_w(s) = -\mathrm{Nu}\,\alpha^*\tilde{\theta}_w(s)+\mathrm{Nu}\,\alpha^*\frac{\Omega}{s^2+\Omega^2} \tag{5.28}$$

Solving for the dimensionless wall temperature we get the final response function in the Laplace domain.

$$\tilde{\theta}_w(s) = \frac{\mathrm{Nu}\,\alpha^*}{(\mathrm{Nu}\,\alpha^*+s)}\frac{\Omega}{s^2+\Omega^2} \tag{5.29}$$

To obtain an expression for the forcing function input, we must nondimensionalize and transform the bulk fluid temperature separately.

$$\theta_b(\tau) = \left(T_b(t)-T_o\right)/\Delta T_o \tag{5.30}$$

This dimensionless bulk temperature becomes

$$\theta_b(\tau) = \sin(\Omega\tau) \tag{5.31}$$

We take the Laplace transform to obtain the input function.

$$\mathcal{L}\left\{\theta_b(\tau)\right\} = \mathcal{L}\left\{\sin(\Omega\tau)\right\} \tag{5.32}$$

$$\tilde{\theta}_b = \frac{\Omega}{s^2+\Omega^2} \tag{5.33}$$

A transfer function for the system is defined, where the bulk temperature forcing function is the input $\tilde{\theta}_b(s) = X(s)$, and the wall temperature response is the output, $\tilde{\theta}_w(s) = Y(s)$.

$$G(s) = \frac{Y(s)}{X(s)} \tag{5.34}$$

The system transfer function simplifies to

$$G(s) = \frac{\mathrm{Nu}\,\alpha^*}{s+\mathrm{Nu}\,\alpha^*} \tag{5.35}$$

Variable	Value
Nu α^*	2
Ω	10
Sample time ($\Delta\tau$)	0.01
Total time (τ_{run})	6

Table 5.1: Initialization parameters for spatially independent, constant properties frequency domain model

Parameter Estimation

One of the potential applications of frequency domain analysis is parameter estimation. A primary question for this work is to explore whether it is possible to leverage frequency response testing as a tool to more accurately measure thermal hydraulic system parameters such as the Nusselt number or potentially thermophysical properties such as the thermal conductivity of Flibe.

To test this hypothesis, discrete time series or frequency domain data for the the experimental input and output response can be inserted into parameter estimation algorithms. The MATLAB System Identification Toolbox was used to accomplish this task.

The parameter estimation starts with performing a baseline estimation of simulated time-series data. Time series data is generated using the derived transfer function in Equation 5.35. Code Listing 5.1 shows how this is set up in MATLAB. In this case, Nu and α^* are combined into one coefficient, a. Table 5.1 shows the initialization parameters used for example system.

```
a=2
Omega=10
sampleTime=0.01
totalTime=6
system=tf([a],[1, a])
timeVector=[0:sampleTime:totalTime]';
initialState=[0];
inputSine=sin(Omega*timeVector)
[simulatedOutputSine, timeVectorSimulatedSine]=lsim(system,inputSine
   ,timeVector,initialState);
```

Listing 5.1: Simulate transfer function response.

To get a baseline accuracy for the parameter estimation process, a transfer function is estimated using the simulated discrete time series data as shown in Listing 5.2. Here, the initial condition variable provides the model structure, which consists of the number of zeros and poles, initial guesses, and the specification of which zeros and poles are constrained. It is advantageous to fix coefficients if there is prior knowledge of their values. The constrained

parameter estimation reduces the degrees of freedom and typically leads to more accurate results. In this example, the higher order pole is fixed to be equal to one, however the other coefficients must be allowed to freely float.

```
%% perform parameter estimation by using the sine response to test
   the basic accuracy

% prepare the parameter estimation data
parameterEstimationDataSine = iddata(simulatedOutputSine, inputSine,
    sampleTime);

%% set the model structure and constrained parameters
init_sys = idtf(a, [1 a]);
init_sys.Structure.Denominator.Free(1) = false; % Constrain the
   first denominator coefficient

estimatedSystemSine = tfest(parameterEstimationDataSine, init_sys)

% Get list of model parameters and their uncertainties
[pvecSine, dpvecSine] = getpvec(estimatedSystemSine, 'free');
covSine = getcov(estimatedSystemSine, 'value', 'free');
```

Listing 5.2: Parameter estimation from simulated true transfer function response.

This baseline accuracy for the estimated system was calculated to be 97.39% relative to the estimation data. This accuracy metric is slightly misleading, as one of the estimated coefficients in the continuous-time transfer function, seen in the denominator of Equation 5.36 has a relative error of 18% compared to the true value of $a = 2$. Further, the algorithm converges on two different values for what should be the same coefficient in the numerator and the denominator. This happens because the estimation algorithm is inadequately constrained. This is an apparent limitation with the MATLAB transfer function estimation tool, `tfest`. This issue was resolved by exploring other parameter estimation methods later in this Chapter.

$$G_{\text{true}}(s) = \frac{2}{s+2}, \qquad G_{\text{baseline}}(s) = \frac{2.015}{s+2.355} \tag{5.36}$$

The next step is to add normally distributed random error to simulate noisy physical measurements. The noisy response data is then used to estimate the transfer function given the same model structure as before. The estimated transfer function was calculated to be an 85% fit compared to the true system simulation. The estimates for a sample run are show in Equation 5.37

```
% test the sine response results with noise
simulatedOutputSineNoise = simulatedOutputSine + 0.02 * randn(size(
   simulatedOutputSine));
```

```
3
4 parameterEstimationDataSineNoise = iddata(simulatedOutputSineNoise,
     inputSine, sampleTime);
5
6 estimatedSystemSineNoise = tfest(parameterEstimationDataSineNoise,
     init_sys)
7
8 % Get list of model parameters and their uncertainties
9 [pvecSineNoise, dpvectNoise] = getpvec(estimatedSystemSineNoise, '
     free');
10 covSineNoise = getcov(estimatedSystemSineNoise, 'value', 'free')
11
12 % simulate the estimated model and compare it with the noisy data
13 [simulatedOutputSineEstimated, timeVectorSimulatedSineEstimated] =
     lsim(estimatedSystemSineNoise, inputSine, timeVector,
     initialState);
```

Listing 5.3: Parameter estimation from simulated noisy response.

$$G_{\text{noisy}}(s) = \frac{2.008}{s + 2.36} \tag{5.37}$$

We can attempt to unpackage the estimated coefficients by using a fixed value for α^*. Given a hypothetical α^* value of 0.85, the Nusselt number is estimated to be either 2.36 or 2.78. This uncertainty may still be within acceptable bounds given that the true value is 2.35. However, this can be improved significantly if an equality constraint can be set between the two coefficients or if the parameters that make up each coefficient in the transfer function can be fixed independently. The latter approach was achieved and demonstrated later in the next section. The model order can also influence the error between the measured data and the predicted model. A higher order model has more flexibility, allowing it to fit the data with increasing accuracy with the trade-off of higher uncertainty and a risk of over fitting. However, if the model order is too low, it could lead to systematic errors due to measurement noise. The estimated and true transfer function simulated time domain responses are plotted with respect to dimensionless time in Figure 5.1.

5.2 Temperature-Dependent Spatially-Independent

Linearization

The assumption of constant properties in the above example might be an oversimplification depending on the system. This can be remedied by introducing temperature dependent properties, although this now results in a nonlinear problem. This is incompatible with transfer function models which must be linear and time-invariant. To work around this,

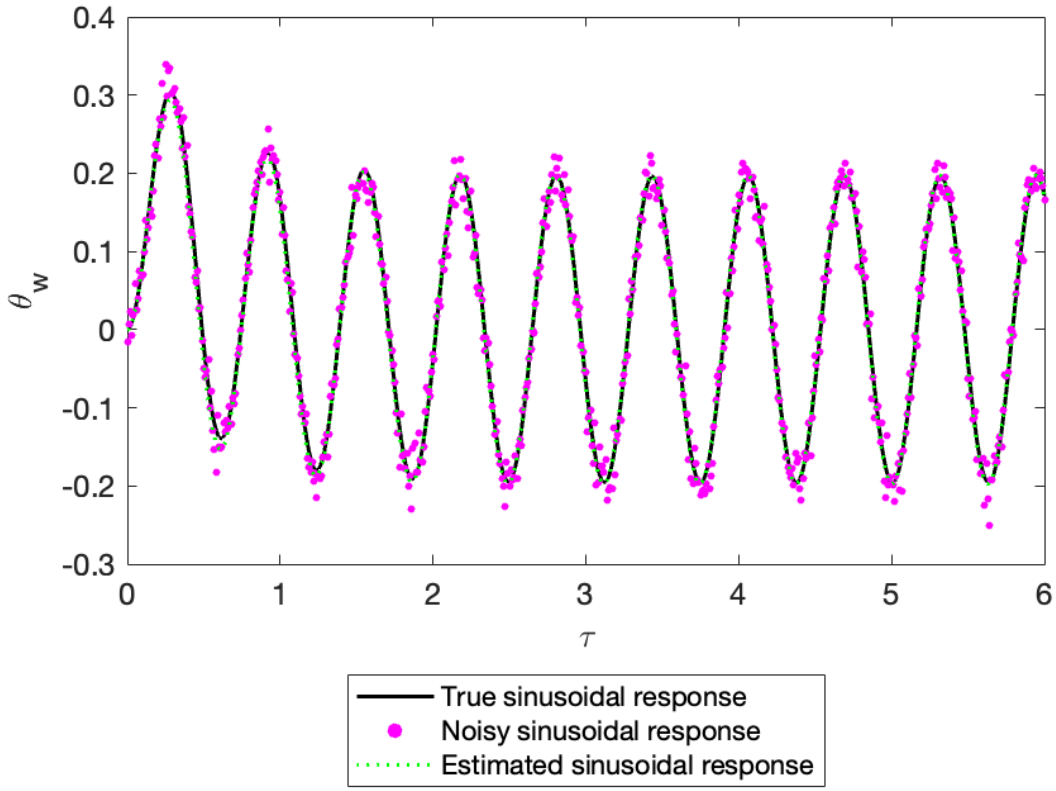


Figure 5.1: Spatially-independent transfer function parameter estimation. Simulated experimental data was generated using the following dimensionless groups: $\mathrm{Nu}\alpha^* = 2$, and $\Omega = 10$

the equation will be linearized using a Taylor series approximation before being transformed into the Laplace domain. The nondimensional parameter α^* temperature dependence can be fitted to a combined correlation using interpolation and linear regression analysis. Assuming a first order polynomial is a sufficient approximation, the parameter takes the form of the function in Equation 5.38. Depending on the temperature dependence trend, a second order polynomial or exponential function might be more appropriate. Since the thermophysical properties must be evaluated at the film temperature, the average of the bulk and fluid temperatures is used.

$$\alpha^*(\theta_{\mathrm{film}}) = p_1\left(\frac{\theta_b + \theta_w}{2}\right) + p_2 \tag{5.38}$$

Substituting in the nondimensional bulk fluid temperature we have.

$$\alpha^*(\theta_{\mathrm{film}}) = p_1\left(\frac{\sin(\Omega\tau) + \theta_w}{2}\right) + p_2 \tag{5.39}$$

This temperature dependent form of α^* can now be replaced into the energy equation.

$$\frac{\partial\theta_w(\tau)}{\partial\tau} = -\mathrm{Nu}\left[\frac{p_1}{2}(\sin(\Omega\tau)+\theta_w)+p_2\right](\theta_w(\tau)-\sin(\Omega\tau)) \tag{5.40}$$

The expression is expanded to consolidate the dependent variable terms.

$$\begin{aligned}\frac{\partial\theta_w(\tau)}{\partial\tau} = -\mathrm{Nu}\Big[&\frac{p_1}{2}\left(\sin(\Omega\tau)\theta_w-\sin^2(\Omega\tau)+\theta_w^2-\sin(\Omega\tau)\theta_w\right)\\ &+p_2(\theta_w-\sin(\Omega\tau))\Big]\end{aligned} \tag{5.41}$$

Simplifying we get

$$\frac{\partial\theta_w(\tau)}{\partial\tau} = -\mathrm{Nu}\Big[\frac{p_1}{2}\left(\theta_w^2-\sin^2(\Omega\tau)\right)+p_2(\theta_w-\sin(\Omega\tau))\Big] \tag{5.42}$$

The equilibrium point must be found and this is done by setting $\partial\theta_w(\tau)/\partial\tau = 0$ and by ignoring all explicit functions of time. The equation is then solved for the dependent variable.

$$0 = \frac{p_1}{2}\theta_w^2+p_2\theta_w \tag{5.43}$$

$$\bar{\theta}_w = \left(0,\,-\frac{2p_2}{p_1}\right) \tag{5.44}$$

While a negative dimensionless temperature is valid, it is reasonable to center the expansion around zero due to the sinusoidal periodicity. Linearizing we obtain

$$\theta_w^2 \approx 2\bar{\theta}_w(\theta_w-\bar{\theta}_w) = 0 \tag{5.45}$$

The linearized equation is put in terms of the incremental deviation

$$\theta_w = \bar{\theta}_w+\hat{\theta}_w(\tau) \tag{5.46}$$

Substituting into the nonlinear energy equation we get

$$\frac{\partial\hat{\theta}_w(\tau)}{\partial\tau} = -\mathrm{Nu}\Big[-\frac{p_1}{2}\sin^2(\Omega\tau)+p_2(\hat{\theta}_w-\sin(\Omega\tau))\Big] \tag{5.47}$$

The linearized energy balance can now be transformed to the Laplace domain. It can be determined that

$$\mathcal{L}\left\{\sin^2(\Omega\tau)\right\} = \frac{2\Omega^2}{s(s^2+4\Omega^2)} \tag{5.48}$$

Using this, and referencing a table of Laplace transforms we obtain

$$s\tilde{\theta}_w(s)-\theta_w(0) = -\mathrm{Nu}\Big[-\frac{p_1}{2}\frac{2\Omega^2}{s(s^2+4\Omega^2)}+p_2\left(\tilde{\theta}_w(s)-\frac{\Omega}{s^2+\Omega^2}\right)\Big] \tag{5.49}$$

Variable	Value
Nu	2
$\alpha^* = p_1\theta_{\text{film}} + p_2$	-
p_1	0.2
p_2	0.85
Ω	10
Sample time ($\Delta\tau$)	0.01
Total time (τ_{run})	6

Table 5.2: Initialization parameters for spatially independent, temperature dependent properties frequency domain model

The dependent variable is consolidated as follows

$$s\tilde{\theta}_w(s) + \text{Nu}\, p_2\tilde{\theta}_w(s) = \text{Nu}\left[\frac{p_1}{2}\frac{2\Omega^2}{s(s^2+4\Omega^2)} + p_2\frac{\Omega}{s^2+\Omega^2}\right] \tag{5.50}$$

$$\tilde{\theta}_w(s) = \frac{\text{Nu}}{s+\text{Nu}\, p_2}\left[\frac{p_1}{2}\frac{2\Omega^2}{s(s^2+4\Omega^2)} + p_2\frac{\Omega}{s^2+\Omega^2}\right] \tag{5.51}$$

Similarly to before, the transfer function with linearized temperature dependent thermophysical properties is obtained by dividing the system output by the input forcing function.

$$G(s) = \frac{\frac{\text{Nu}}{s+\text{Nu}\, p_2}\left[p_1\frac{\Omega^2}{s(s^2+4\Omega^2)} + p_2\frac{\Omega}{s^2+\Omega^2}\right]}{\frac{\Omega}{s^2+\Omega^2}} \tag{5.52}$$

This was further simplified using the Python `sympy` package.

$$G(s) = \frac{\text{Nu}\,(p_2s^3 + \Omega p_1 s^2 + 4\Omega^2 p_2 s + \Omega^3 p_1)}{s^4 + \text{Nu}p_2s^3 + 4\Omega^2s^2 + 4\text{Nu}\Omega^2 p_2 s} \tag{5.53}$$

The transfer function system has four poles and three zeros, which also means in this case is that there are more coefficients that can be solved for the Nu number. The same testing procedure was conducted wherein the parameters of the transfer function were estimated using a noisy measurements simulation of the true model. Table 5.2 shows the initialization parameters and variables.

$$G_{\text{true}}(s) = \frac{1.7s^3 + 4s^2 + 680s + 400}{s^4 + 1.7s^3 + 400s^2 + 680s} \tag{5.54}$$

$$G_{\text{baseline}}(s) = \frac{1.71s^3 + 6.55s^2 + 680s + 399.97}{s^4 + 1.7s^3 + 400s^2 + 680s} \tag{5.55}$$

$$G_{\text{noisy}}(s) = \frac{1.79s^3 + 7.45s^2 + 680s + 399.97}{s^4 + 1.68s^3 + 400s^2 + 638.74s} \tag{5.56}$$

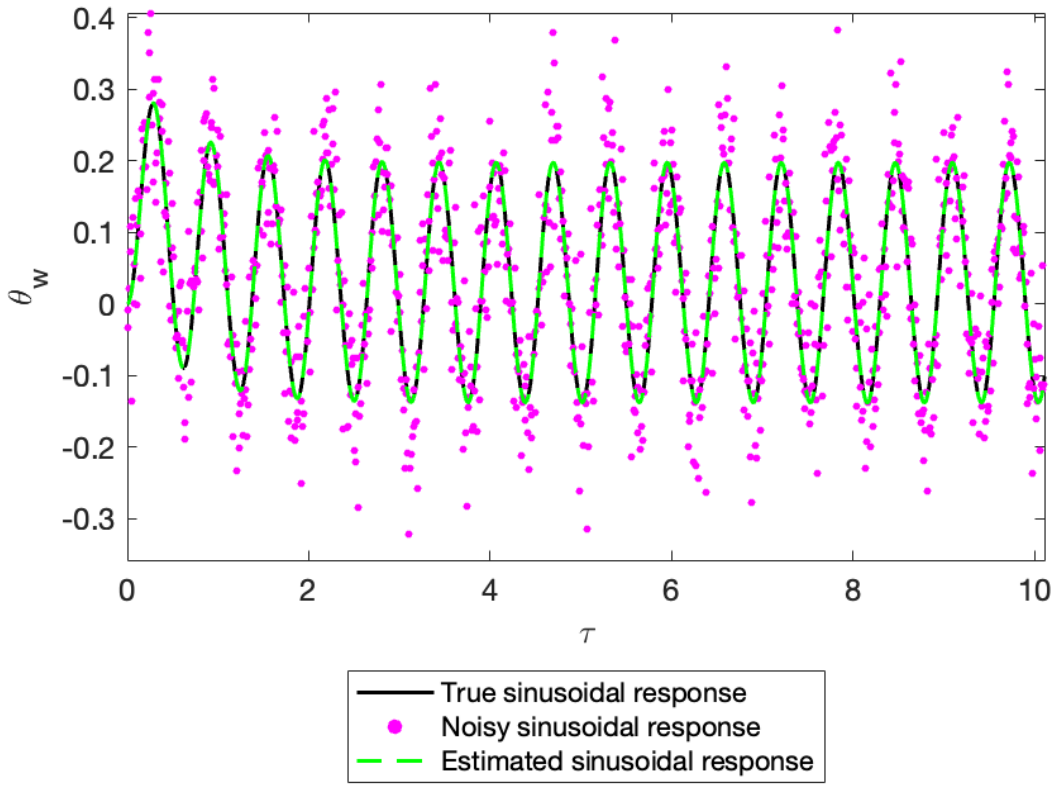


Figure 5.2: Transfer function parameter estimation for temperature dependent thermophysical properties case. Simulated experimental data was generated using the following dimensionless parameters: Nu = 2, $\alpha^* = p_1\theta_{\text{film}} + p_2$, where $p_1 = 0.2$ and $p_2 = 0.85$, and $\Omega = 10$

A sample result for the estimated transfer function is provided in Equation 5.56. Comparing the true, baseline and noisy estimated transfer function, it is clear that coefficients that should be equal do not always match. As a result, and depending on the simulated experimental run, the estimated Nu number can deviate as much as 18% from the true value. The MATLAB function `tfest` does not provide the option to fix individual parameters or to set equality constraints on the model coefficients. A different numerical method is needed to improve this result. The Newton-Raphson method was implemented with a custom objective function that minimizes the error between the noisy measured data and a simulation of the estimated transfer function. The Newton-Raphson method is an iterative root-finding algorithm. Given an initial guess, the method calculates successive approximations or estimates of the root by linearizing the function at the current estimate. The mathematical expression

for the method is shown in Equation 5.57.

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{5.57}$$

where x_n is the current estimate, $f(x_n)$ is the output at the current estimate, and $f'(x_n)$ is the derivative of the output at the current estimate. The gradient is computed numerically using the central difference method defined in Equation 5.58. The step size, h, of 1E-6 was used.

$$f'(x_n) \approx \frac{f(x_n + h) - f(x_n - h)}{2h} \tag{5.58}$$

The algorithm was provided an initial guess for the Nu number value (Nu = 0) and during each iteration, the estimated output is simulated using the transfer function model. The transfer function output is evaluated using the current estimate for Nu and the known parameters Ω, p_1 and p_2, as per Table 5.2. This method takes advantage of both model structure and parameters known a priori.

A Monte Carlo simulation of $n = 1000$ experiments was conducted to obtain an estimate of the method's uncertainty.

Parameter Estimation Results

The Monte Carlo simulaton produced an estimated Nu number of 1.9940 with a standard deviation of 0.0199. This corresponds to a relative error of 0.3%±1%. This level of accuracy appears to be much higher compared to more conventional data reduction procedures implemented for the SHEFRA experiment and also compared to typical heat transfer experiments. The magnitude of random noise in the simulations is scaled to match the uncertainty from SHEFRA wall thermocouples when the amplitude of temperature fluctuations is equal to 5. Lower amplitudes scale to higher measurement noise. The simulations do not account for any other source of measurement uncertainty such as for the bulk fluid temperature input. To compensate for this, an additional Monte Carlo simulation was run with three times the wall temperature uncertainty. The higher noise simulation produced an estimated Nu with a relative error of 0.04%±2.8%. This result suggests that higher Gaussian noise might lead to smaller relative error in the nominal value, but with more uncertainty. The MATLAB code is provided in the Appendix. Figure 5.2 shows the simulated transfer function estimated from the noisy linearized temperature dependent response. An additional benefit of the optimization algorithm used, is that it is capable of solving nonlinear systems without resorting to linearization. The MALTAB code implementation of the parameter method is provided in Appendix D.

5.3 Spatially-Dependent Channel

In the physical SHEFRA experiment, the bulk fluid temperature oscillations are dampened as the flow travels downstream of the channel inlet. This effect is important to capture

in order to measure the Nu number as a function of axial distance. This section will use a dynamic model for unsteady conjugate heat transfer in a circular duct with convection from ambient and periodically varying inlet temperature. The problem was solved using the Laplace Transform by Travelho and Santos [22]. In the present work, in Chapter 3, a quasi-steady model was applied to approximate the Nu number using the exact wall and bulk fluid temperature, and heat flux solutions. This will be the starting point for a transfer function model for parameter estimation.

$$\mathrm{Nu}(Z) = \frac{-\left.\frac{\partial\theta(R,Z,\tau)}{\partial R}\right|_{R=1}}{\theta_b(Z,\tau) - \theta_w(Z,\tau)} \tag{5.59}$$

The expression for the quasi-steady Nu number in Equation 5.59 is rearranged to solve for the dimensionless wall temperature. Equation 5.61 expresses the output as a function of the bulk fluid temperature input, the solution to the heat flux, and the Nu number.

$$\mathrm{Nu}\,\theta_b(Z,\tau) - \mathrm{Nu}\,\theta_w(Z,\tau) = -\left.\frac{\partial\theta(R,Z,\tau)}{\partial R}\right|_{R=1} \tag{5.60}$$

$$\theta_w(Z,\tau) = \frac{1}{\mathrm{Nu}}\left.\frac{\partial\theta(R,Z,\tau)}{\partial R}\right|_{R=1} + \theta_b(Z,\tau) \tag{5.61}$$

Taking the Laplace transform of Equation 5.61 with respect to τ provides an expression for the quasi-steady model in the Laplace domain.

$$\tilde{\theta}_w(Z,s) = \frac{1}{\mathrm{Nu}}\left(s\tilde{\theta}(R=1,Z,s) - \left.\frac{\partial\theta(R,Z,\tau=0)}{\partial R}\right|_{R=1}\right) + \tilde{\theta}_b(Z,s) \tag{5.62}$$

The terms in Equation 5.62 are simplified and put in terms of the dependent variable or as functions of R, Z and the Laplace variable s. The problem solution, θ when evaluated at $R = 1$ is the dimensionless wall temperature solution. When transformed into the Laplace domain, we recognize that this is our dependent variable.

$$\tilde{\theta}(R=1,Z,s) = \tilde{\theta}_w(Z,s) \tag{5.63}$$

Next, the solution to the dimensionless heat flux is evaluated at the initial condition. The solutions of the dimensionless heat flux, wall and bulk temperature are each a sum of a permanent and periodic solution, $\xi(R,Z)$ and $\gamma(R,Z,\tau)$, respectively.

$$\theta(R,Z,\tau) = \xi(R,Z) + \gamma(R,Z,\tau) \tag{5.64}$$

The periodic solution assumes the form

$$\gamma(R,Z,s) = \psi(R,Z)e^{i\Omega(\tau-Z)} \tag{5.65}$$

Recognizing that the exponential term is not a function of R, taking the partial derivative produces this expression for the dimensionless heat flux.

$$\left.\frac{\partial\theta(R,Z,\tau)}{\partial R}\right|_{R=1} = \xi_h(R,Z) + \psi_h(R,Z)e^{i\Omega(\tau-Z)} \tag{5.66}$$

where the permanent and periodic dimensionless heat flux are defined as

$$\psi_h(Z) \equiv -\left.\frac{\partial\psi(R,Z)}{\partial R}\right|_{R=1} = c^*\psi(1,Z) \tag{5.67}$$

$$\begin{aligned}\xi_h(Z) = -\frac{\mathrm{Nu_o}\theta_\infty}{(\eta_1-\eta_2)}\Big[&\eta_1 e^{\eta_1^2 Z}\mathrm{erfc}(\eta_1\sqrt{Z}) \\ &-\eta_2 e^{\eta_2^2 Z}\mathrm{erfc}(\eta_2\sqrt{Z})\Big]\end{aligned} \tag{5.68}$$

By assuming no convection to ambient, the permanent solution is eliminated. Evaluating at the initial condition we get

$$\left.\frac{\partial\theta(R,Z,\tau=0)}{\partial R}\right|_{R=1} = c^*\psi(1,Z)e^{i\Omega(0-Z)} \tag{5.69}$$

where $\psi(1,Z)$ is the periodic dimensionless periodic wall temperature.

$$\begin{aligned}\psi_w(Z) \equiv \psi(1,Z) = \frac{1}{\lambda_1-\lambda_2}\Big(&\lambda_1 e^{\lambda_1^2 Z}\mathrm{erfc}(-\lambda_1\sqrt{Z}) \\ &-\lambda_2 e^{\lambda_2^2 Z}\mathrm{erfc}(-\lambda_2\sqrt{Z})\Big)\end{aligned} \tag{5.70}$$

Instead of using the provided solution, the model can be reduced by representing $\psi(1,Z)$ in terms of the dependent variable. To obtain this relationship, the time-domain form of the dependent variable is expanded into it's periodic and permanent components.

$$\theta_w(Z,\tau) = \xi_w(R,Z) + \psi_w(Z)e^{i\Omega(\tau-Z)} \tag{5.71}$$

Similarly as before, the permanent dimensionless wall temperature is eliminated. The Laplace transform with respect to τ is then applied.

$$\begin{aligned}\tilde{\theta}_w(Z,s) &= \mathcal{L}\left\{\psi_w(Z)e^{i\Omega(\tau-Z)}\right\} \\ &= \mathcal{L}\left\{\psi_w(Z)\frac{e^{i\Omega\tau}}{e^{i\Omega Z}}\right\} \\ &= \frac{\psi_w(Z)}{e^{i\Omega Z}}\frac{1}{s-i\Omega}\end{aligned} \tag{5.72}$$

After rearranging, we obtain an expression that can be subsituted into the heat flux initial condition term in Equation 5.69.

$$\psi_w(Z) = \tilde{\theta}_w(Z,s)e^{i\Omega Z}(s-i\Omega) \tag{5.73}$$

After substitution, the dimensionless heat flux initial condition becomes

$$\left.\frac{\partial\theta(R,Z,\tau=0)}{\partial R}\right|_{R=1} = c^*\tilde{\theta}_w(Z,s)e^{i\Omega Z}(s-i\Omega)e^{-i\Omega Z} \tag{5.74}$$

The exponential terms cancel and we are left with

$$\left.\frac{\partial\theta(R,Z,\tau=0)}{\partial R}\right|_{R=1} = c^*\tilde{\theta}_w(Z,s)(s-i\Omega) \tag{5.75}$$

The last term needed to construct the transfer function ouput in Equation 5.62, is the dimensionless bulk fluid temperature in the Laplace domain.

$$\begin{aligned}\tilde{\theta}_b(Z,s) &= \mathcal{L}\left\{\psi_b(Z)e^{i\Omega(\tau-Z)}\right\} \\ &= \frac{\psi_b(Z)}{e^{i\Omega Z}}\frac{1}{s-i\Omega}\end{aligned} \tag{5.76}$$

The solution to periodic dimensionless bulk temperature is provided.

$$\begin{aligned}\psi_b(Z) = -3 - \frac{2c^*}{\lambda_1-\lambda_2}\Big(&\frac{1}{\lambda_1}e^{\lambda_1^2 Z}\mathrm{erfc}(-\lambda_1\sqrt{Z}) \\ &-\frac{1}{\lambda_2}e^{\lambda_2^2 Z}\mathrm{erfc}(-\lambda_2\sqrt{Z})\Big)\end{aligned} \tag{5.77}$$

where $\lambda_2^1 = -c^*/2 \pm \left[c^*/2\,(c^*/2-1)\right]^{1/2}$. Substituting Equations 5.63, 5.76 and 5.75 into Equation 5.62 we obtain

$$\tilde{\theta}_w(Z,s) = \frac{1}{\mathrm{Nu}}\left(s\tilde{\theta}_w(Z,s) - c^*\tilde{\theta}_w(Z,s)(s-i\Omega)\right) + \frac{\psi_b(Z)}{e^{i\Omega Z}}\frac{1}{s-i\Omega} \tag{5.78}$$

The expression is solved for $\theta_w(Z,s)$.

$$\tilde{\theta}_w(Z,s) - \frac{1}{\mathrm{Nu}}\left(s\tilde{\theta}_w(Z,s) - c^*\tilde{\theta}_w(Z,s)(s-i\Omega)\right) = \frac{\psi_b(Z)}{e^{i\Omega Z}}\frac{1}{s-i\Omega} \tag{5.79}$$

$$\tilde{\theta}_w(Z,s)\left(1 - \frac{s}{\mathrm{Nu}} + \frac{c^*}{\mathrm{Nu}}(s-i\Omega)\right) = \frac{\psi_b(Z)}{e^{i\Omega Z}}\frac{1}{s-i\Omega} \tag{5.80}$$

$$\tilde{\theta}_w(Z,s) = \frac{\psi_b(Z)}{e^{i\Omega Z}}\frac{1}{(s-i\Omega)\left(1-\frac{s}{\mathrm{Nu}}+\frac{c^*}{\mathrm{Nu}}(s-i\Omega)\right)} \tag{5.81}$$

$$\tilde{\theta}_w(Z,s) = \frac{\psi_b(Z)}{e^{i\Omega Z}}\frac{\mathrm{Nu}}{(s-i\Omega)\left(\mathrm{Nu}-s+c^*(s-i\Omega)\right)} \tag{5.82}$$

The input to the system is inlet bulk fluid temperature boundary condition represented as a complex exponential.

$$\theta(R,0,\tau) = e^{i\Omega\tau} \tag{5.83}$$

Transforming into the Laplace domain, we get the transfer function input function.

$$\mathcal{L}\left\{e^{i\Omega\tau}\right\} = \frac{1}{s - i\Omega} \tag{5.84}$$

$$\tilde{\theta}(R, 0, s) = \frac{1}{s - i\Omega} \tag{5.85}$$

The spatially dependent transfer function can now be constructed using the output and input functions.

$$G(Z, s) = \frac{\tilde{\theta}_w(Z, s)}{\tilde{\theta}(R, 0, s)} \tag{5.86}$$

Using the Python `sympy` symbolic math package the expression for the following transfer function is obtained.

$$G(s) = -\frac{\mathrm{Nu}\psi_b(Z)e^{-i\Omega Z}}{-\mathrm{Nu} + c^*\left(i\Omega - s\right) + s} \tag{5.87}$$

The dimensionless parameters are originally defined in Equation 3.46.

$$c^* = \mathrm{Nu}_o + ib^*, \quad b^* = \frac{\Omega}{a^*} = \frac{\omega a \rho_w c_w l}{k} \tag{5.88}$$

Neglecting convection to ambient, c^* simplifies to ib^*. This is substituted into the transfer function to obtain its final form.

$$G(s) = \frac{\mathrm{Nu}\psi_b(Z)e^{-i\Omega Z}}{\mathrm{Nu} + \Omega b^* + s\left(ib^* - 1\right)} \tag{5.89}$$

We can also update the solution for periodic dimensionless bulk temperature by replacing c^*.

$$\begin{aligned}\psi_b(Z) = -3 - \frac{2ib^*}{\lambda_1 - \lambda_2}\Big(&\frac{1}{\lambda_1}e^{\lambda_1^2 Z}\mathrm{erfc}(-\lambda_1\sqrt{Z}) \\ &-\frac{1}{\lambda_2}e^{\lambda_1^2 Z}\mathrm{erfc}(-\lambda_2\sqrt{Z})\Big)\end{aligned} \tag{5.90}$$

where, $\lambda_2^1 = -ib^*/2 \pm \left[ib^*/2\left(ib^*/2 - 1\right)\right]^{1/2}$.

Results and Conclusion

The derived transfer function was found to be unstable, as evident in the simulation plotted in Figure 5.3. This is determined by seeing that the system pole, or the root of the denominator, has a positive real component. The unstable transfer function is most likely due to a derivation error as the solutions have been proven to be stable in Chapter 3.

This result prompted the search of simpler or more versatile approaches to estimate the spatially-dependent Nu number using the same error-minimizing optimization method. The

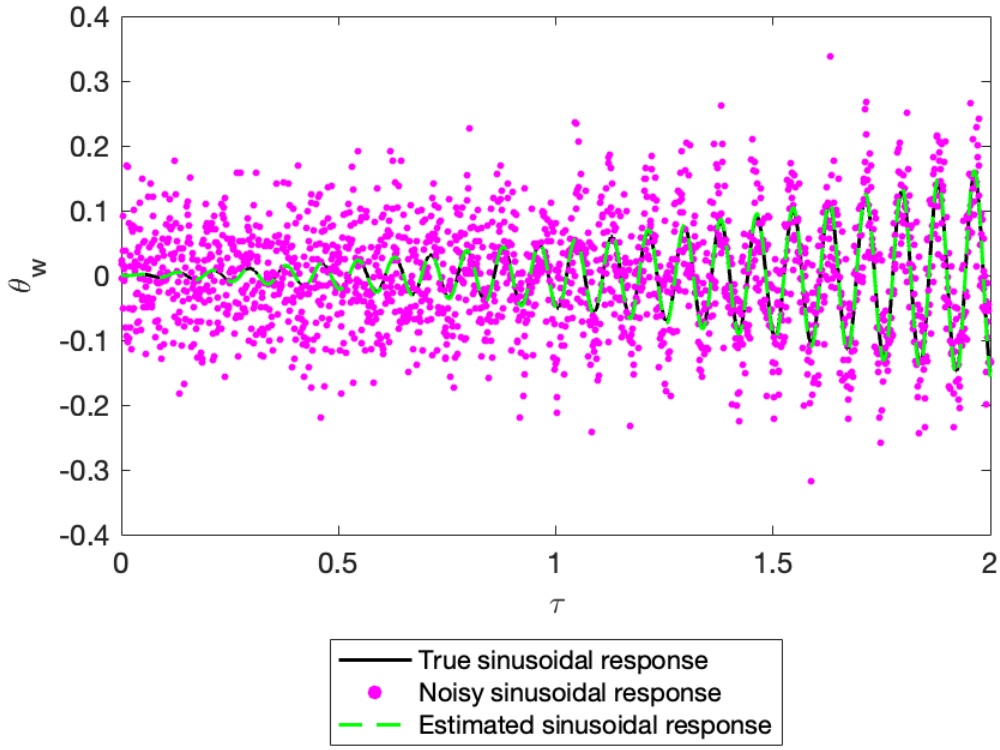


Figure 5.3: Spatially-dependent transfer function model parameter estimation. Simulated experimental data was generated using the following dimensionless parameters: Nu = 4.8, $b^* = 97$ ($f = 0.25$ Hz), $Z = 0.02355$ ($z = 1.96$ meters), $\Omega = 75$

first possibility is to shift the problem boundary so that the forcing input is the local bulk fluid temperature at some axial location Z. The new input function could be obtained by using the dimensionless bulk temperature solution, $\psi_b(Z)$. The dimensionless inlet bulk temperature function is simply multiplied by $\psi_b(Z)$ to result in the local bulk temperature function at the appropriate scale. We can now reuse the linearized transfer function derived for the temperature dependent case in Equation 5.53. The estimated Nu using this method should have a similar accuracy as was seen in the previous example. Additional testing is needed to demonstrate the validity of this method.

Yet another approach would be to represent the time-domain partial differential equations in state space form and to then employ a similar grey-box parameter estimation scheme as implemented for the temperature dependent model. The accuracy improvement in estimating the Nusselt number using the current parameter estimation method is not reliant on the model being a transfer function. The transfer function serves two main purposes: (1)

simulating the system in the time-domain during each iteration of the Newton-Raphson technique, and (2) providing the model structure in the objective function. System parameters such as b^* and Ω can then be fixed to reduce the flexibility of the model, hence improving the accuracy. A time-domain model representation should fulfill both of these requirements. State-of-the-art nonlinear solvers for PDEs such as those available in MATLAB's System Identification Toolbox would also avoid the need to linearize temperature dependent parameters. These functions have the ability to ingest time or frequency domain data from multiple experiments simultaneously.

The error minimization arises from estimating the Nusselt (Nu) number value over the entire data set instead of calculating it at each time step. Additionally, the accuracy of the estimation procedure is contingent on the intrinsic characteristics of the response, such as high gain, low distortion, and minimal bias. High fidelity data can be represented in both frequency and time-domains interchangeably. This highlights the importance of selecting the optimal experimental parameters and conditions. This can be achieved by examining the system's response data in the frequency domain through the use of empirical Bode plots. With SHEFRA, the strategy implemented involved integrating periodic forcing into the physical model, as detailed in Chapter 3. This offers a potent means for identifying optimal frequencies and experimental conditions. When possible, this method holds benefits over the empirical frequency domain analysis technique, as it enables the theoretical prediction of a specific parameter's influence on the system response.

After gathering the data, the performance of the parameter estimation technique is anticipated to be comparable, irrespective of whether the data is in the time or frequency domain. However, acquiring the frequency response through a single known frequency sinusoidal input is the critical factor enabling highly accurate estimates. In more tangible terms, the optimization algorithm is well-conditioned to converge to the true Nu value by knowing or being constrained to the system's characteristic frequency. This as a key advantage derived from application of the frequency response method using sinusoidal inputs.

As seen in the temperature-dependent case, providing model structure and constraining system parameters that are known a priori, such as the input frequency, results in Nu number estimates with low variance despite high measurement noise.

Chapter 6

Conclusions

The design, development, and experimental results of the Scaled Heat Exchange Frequency Response Analysis (SHEFRA) experiment were presented in this dissertation, focusing on the investigation of frequency response methods for measuring quasi-steady Nusselt number values in forced convection of a surrogate fluid for molten fluoride salt in circular ducts. The study's primary motivation is to provide Nu correlations that can be compared with prototypical molten salt experiment data to qualify the use of surrogate fluids in scaled experiments.

The theoretical modeling and frequency scaling analysis provided insights into the relevant dimensionless parameters and their impact on the system's behavior, allowing for optimal experimental parameters and conditions to be determined. The study revealed that quasi-steady heat transfer conditions could be achieved for the laminar flow regime, enabling the accurate prediction of Nusselt numbers using the frequency response technique.

The initial experimental results obtained from the SHEFRA experiment suggest that quasi-steady conditions were validated for laminar flow conditions, showing good agreement with steady-state analytical predictions. Based on this result, additional data will be collected in the laminar regime to cover a range of operating conditions that go beyond the narrow band of Prandlt and Reynolds numbers presented in this work. Further work is needed to extend the analysis and experimental data collection to cover the turbulent flow regime. Additionally, the frequency response parameter estimation techniques presented in this dissertation offer promising avenues for future research in improving the accuracy of Nusselt number estimation and potentially estimating other model parameters including thermophysical properties. Instead of conventionally estimating the Nusselt number at each time step, as done in Chapter 4, a "grey-box" optimization utilizing a frequency domain model of the test section is used to find an estimate time-independent Nusselt number that minimizes the overall error between experimentally measured values and the grey-model simulation.

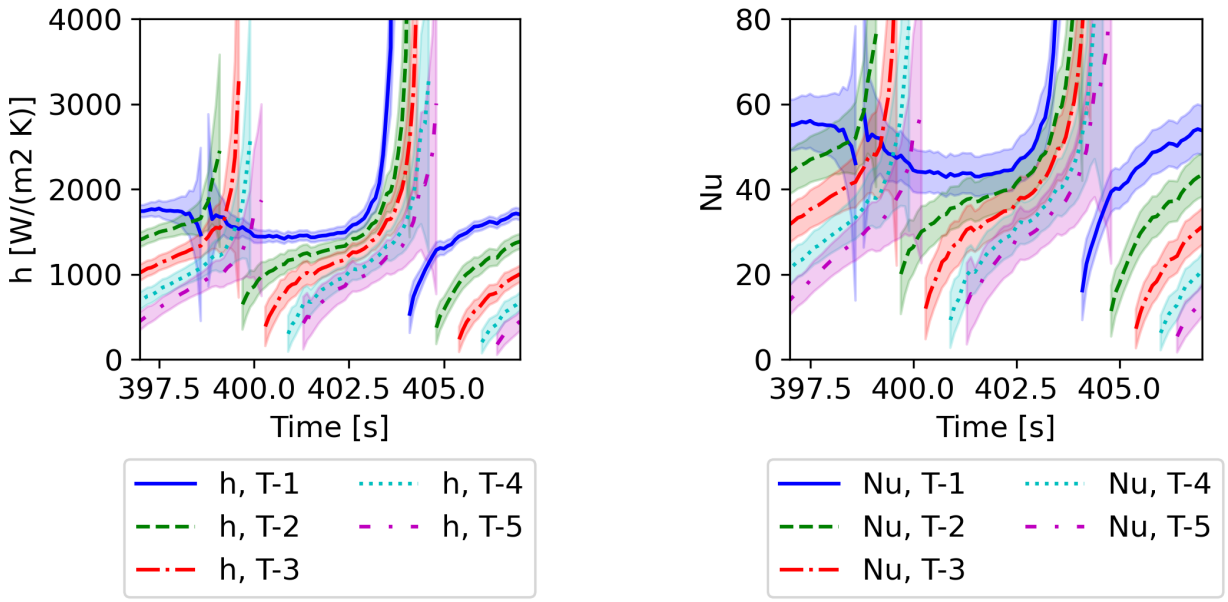
The SHEFRA experiment demonstrates the potential of using surrogate fluids and frequency response methods to obtain high-fidelity heat transfer data measurements relevant for molten salt reactor development.

Appendix A

SHEFRA Experimental Data and Results

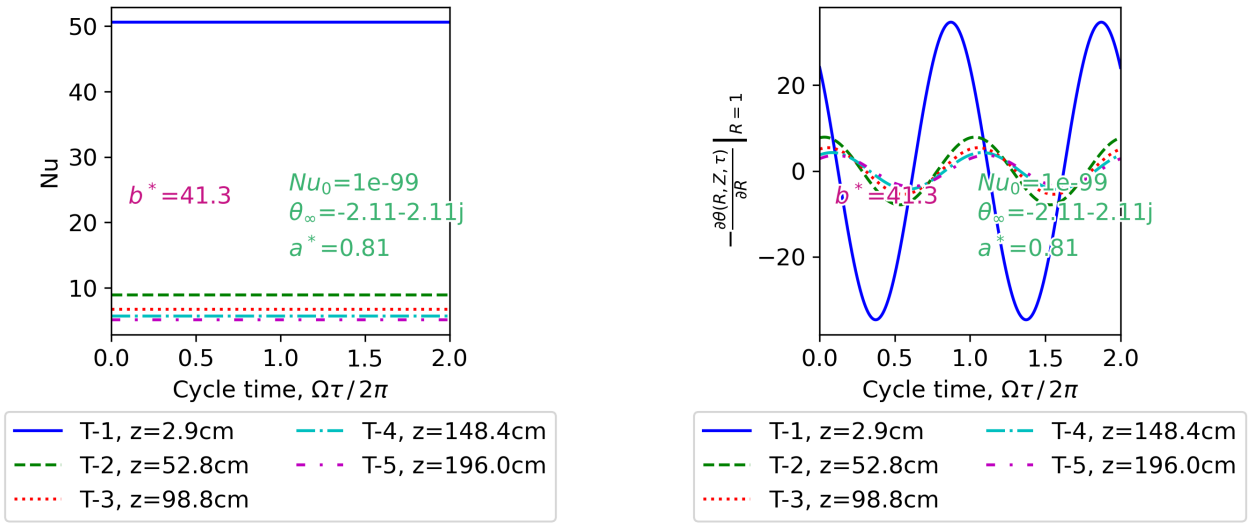
A.1 Series 0

Run parameters:
$f = 0.1\,\text{Hz}$, $T_o = 101°\text{C}$, $\Delta T_o = 38°\text{C}$, $80\,\text{kg/h}$, timestamp : 0435P − 111921



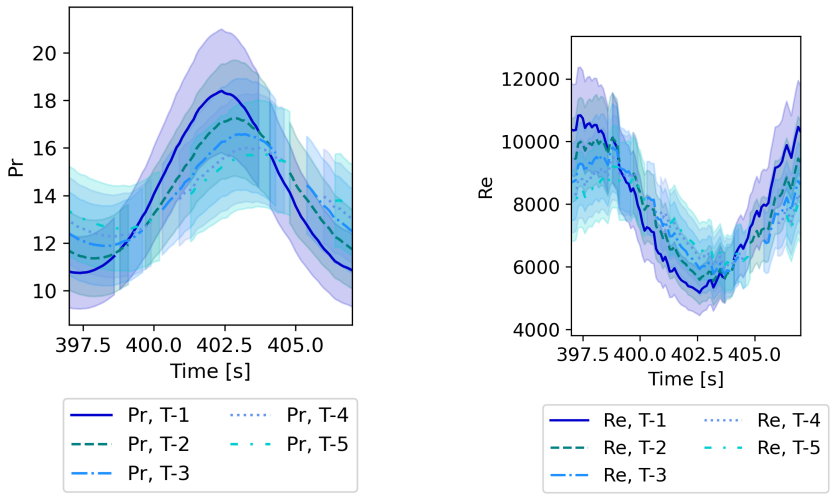
(a) (b)

Figure A.1: Heat transfer coefficient and Nusselt measurements for experimental run (f =0.1Hz, T_o =101°C, ΔT_o =38°C, 80kg/h, timestamp label: 0435P-111921)



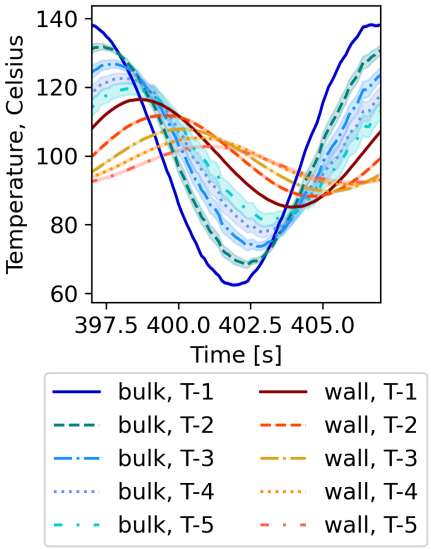
(a) (b)

Figure A.2: Transient laminar model-predicted Nusselt numbers and dimensionless heat flux for experimental run (f =0.1Hz, T_o =101°C, ΔT_o =38°C, 80kg/h, timestamp label: 0435P-111921)

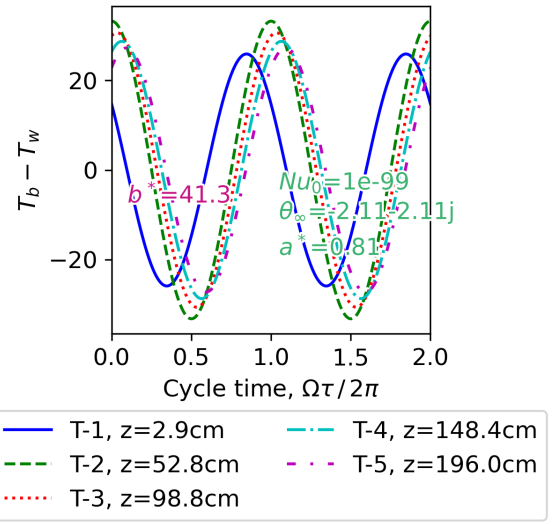


(a) (b)

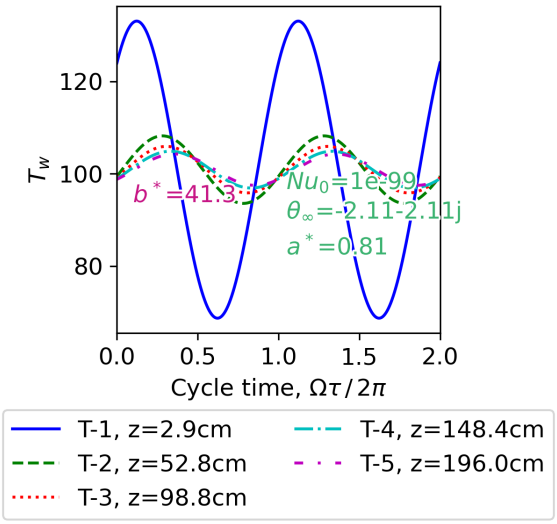
Figure A.3: Prandtl and Reynolds measurements for experimental run (f =0.1Hz, T_o =101°C, ΔT_o =38°C, 80kg/h, timestamp label: 0435P-111921)



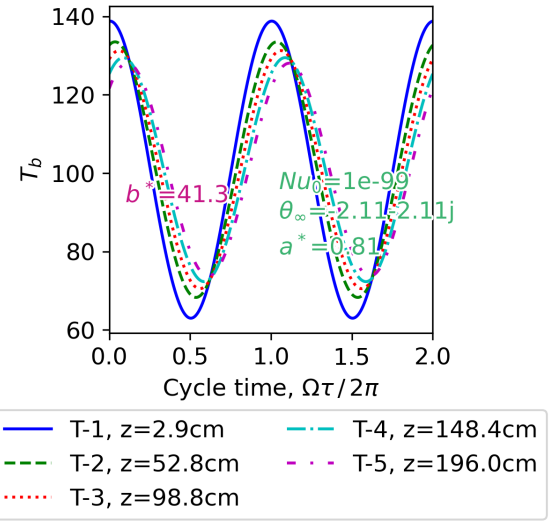
(a) Experimental measurements with uncertainty.



(b) Transient model-predicted delta between wall and local bulk temperature.

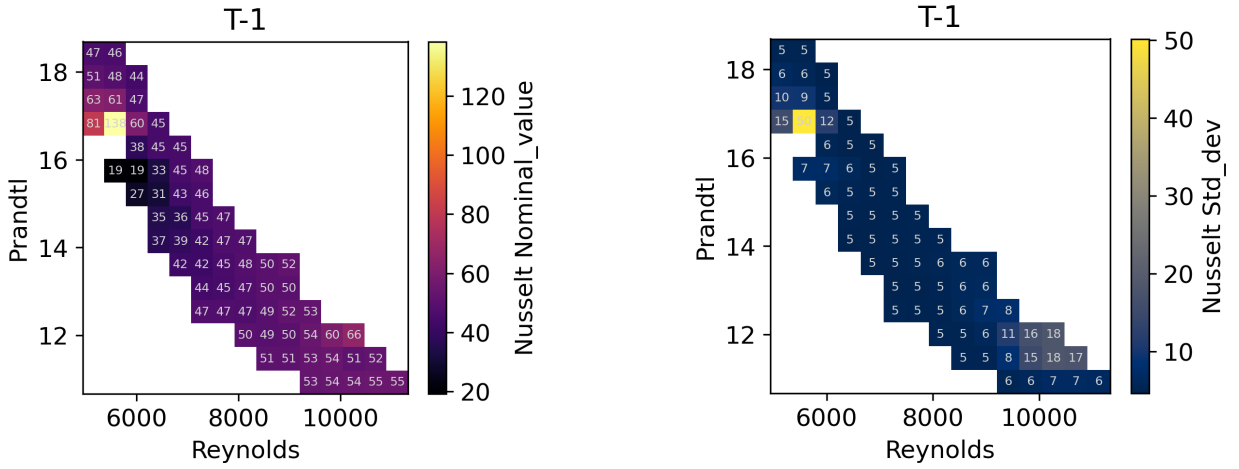


(c) Transient laminar model-predicted wall temperature.



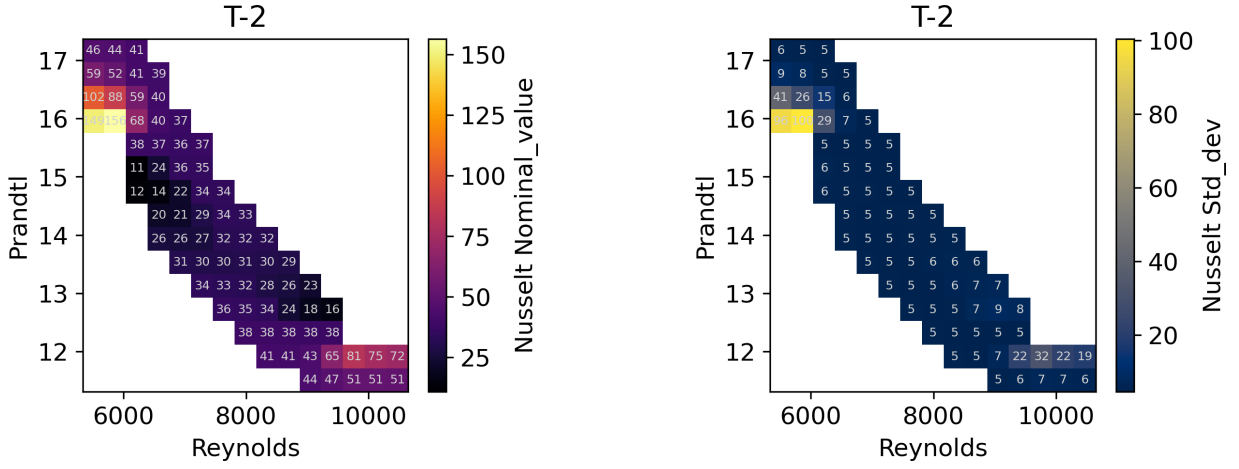
(d) Transient laminar model-predicted bulk temperature.

Figure A.4: Bulk temperature and wall temperature measurements and transient laminar model predictions for experimental run (f =0.1Hz, T_o =101°C, ΔT_o =38°C, 80kg/h, timestamp label: 0435P-111921)



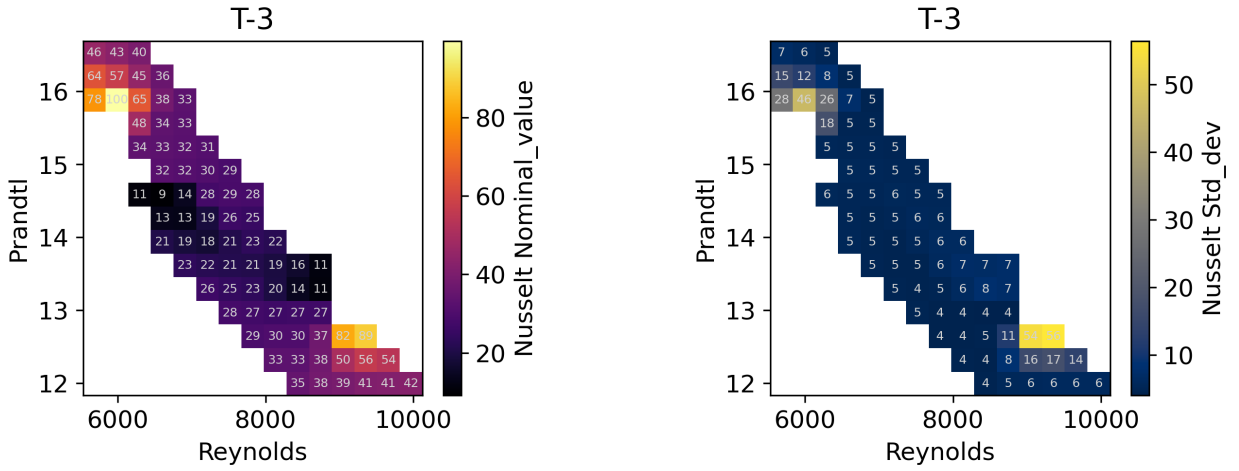
(a) (b)

Figure A.5: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-1. (f =0.1Hz, T_o =101°C, ΔT_o =38°C, 80kg/h, timestamp label: 0435P-111921)



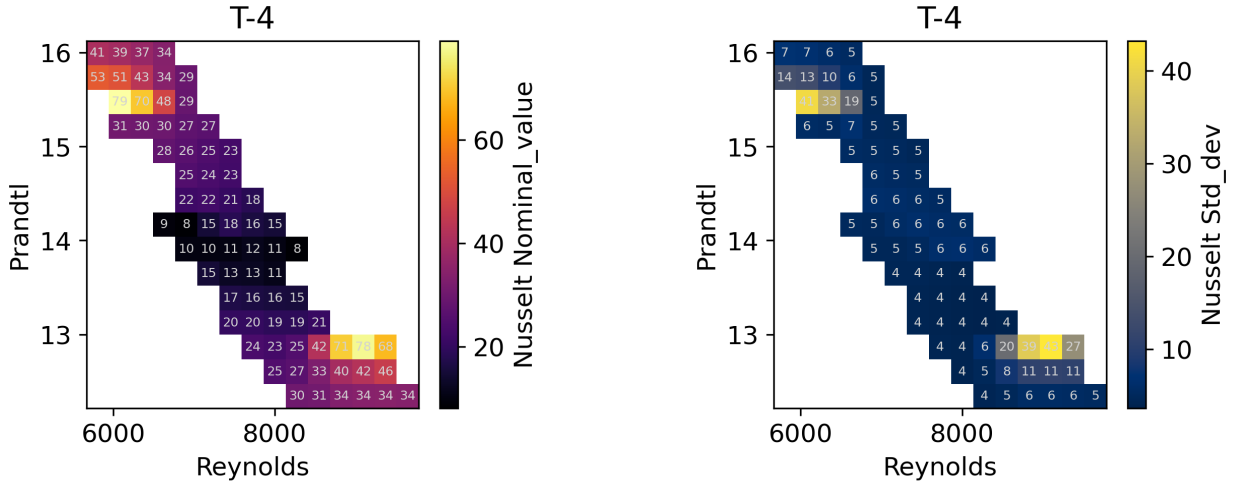
(a) (b)

Figure A.6: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-2. (f =0.1Hz, T_o =101°C, ΔT_o =38°C, 80kg/h, timestamp label: 0435P-111921)



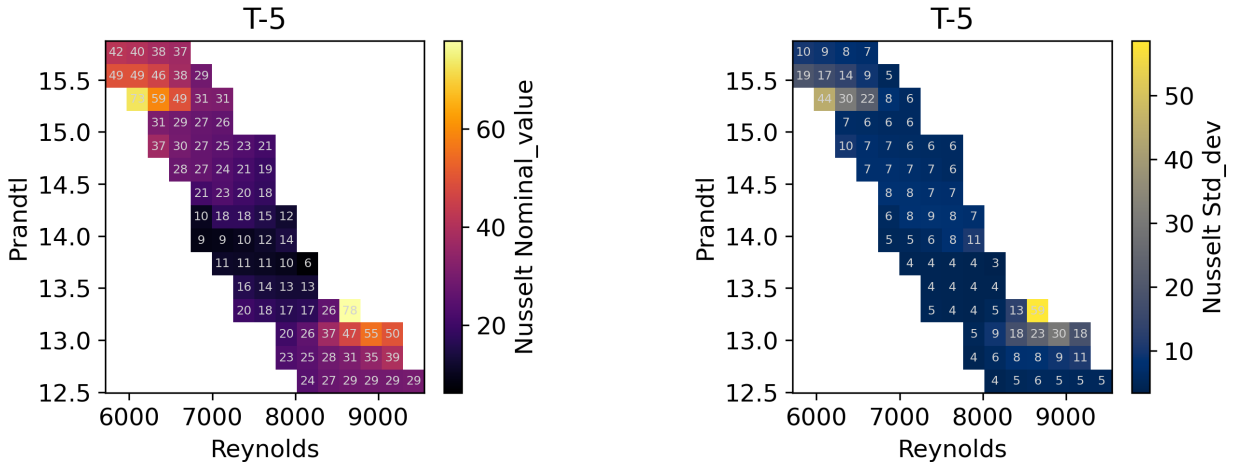
(a) (b)

Figure A.7: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-3. (f =0.1Hz, T_o =101°C, ΔT_o =38°C, 80kg/h, timestamp label: 0435P-111921)



(a) (b)

Figure A.8: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-4. (f =0.1Hz, T_o =101°C, ΔT_o =38°C, 80kg/h, timestamp label: 0435P-111921)



(a) (b)

Figure A.9: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-5. (f =0.1Hz, T_o =101°C, ΔT_o =38°C, 80kg/h, timestamp label: 0435P-111921)

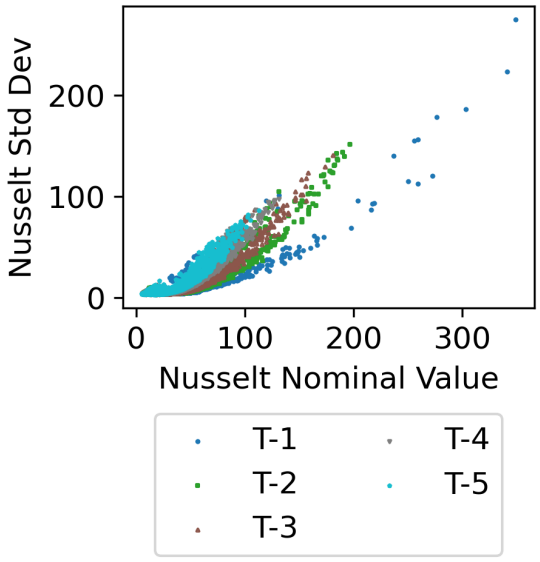
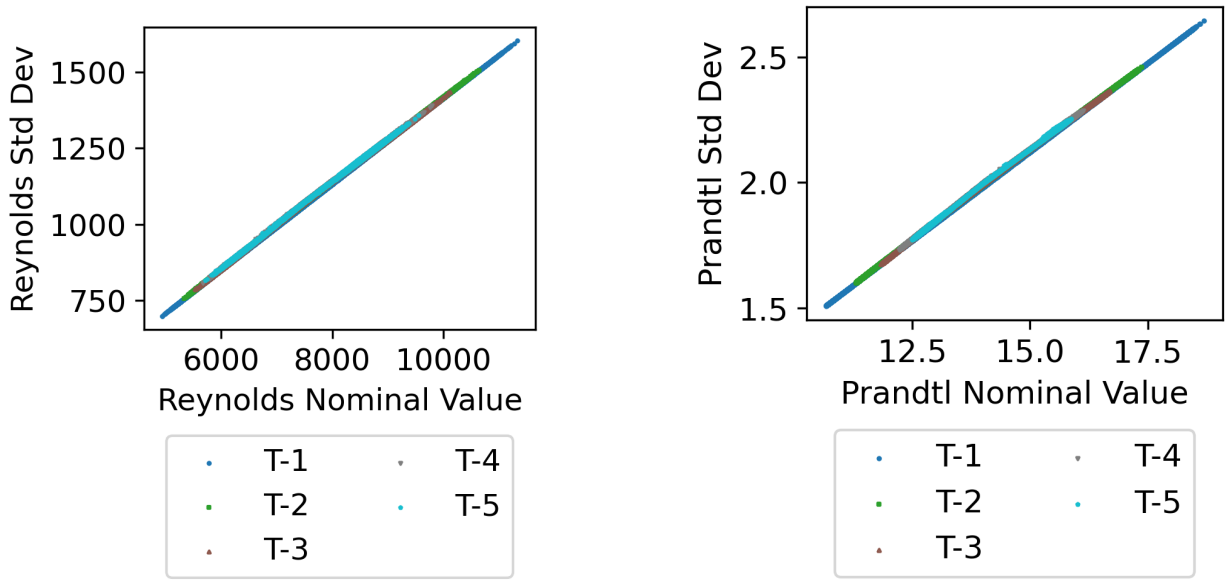


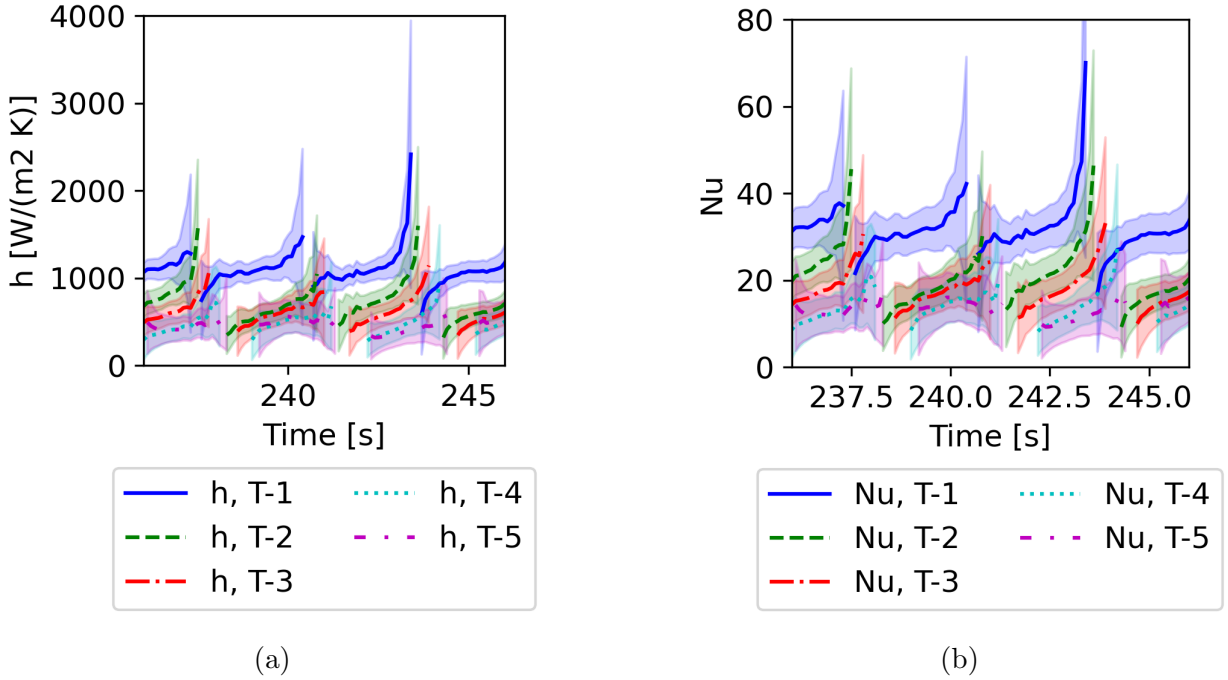
Figure A.10: Nusselt uncertainty distribution for range of nominal values. (f =0.1Hz, T_o =101°C, ΔT_o =38°C, 80kg/h, timestamp label: 0435P-111921)



(a) (b)

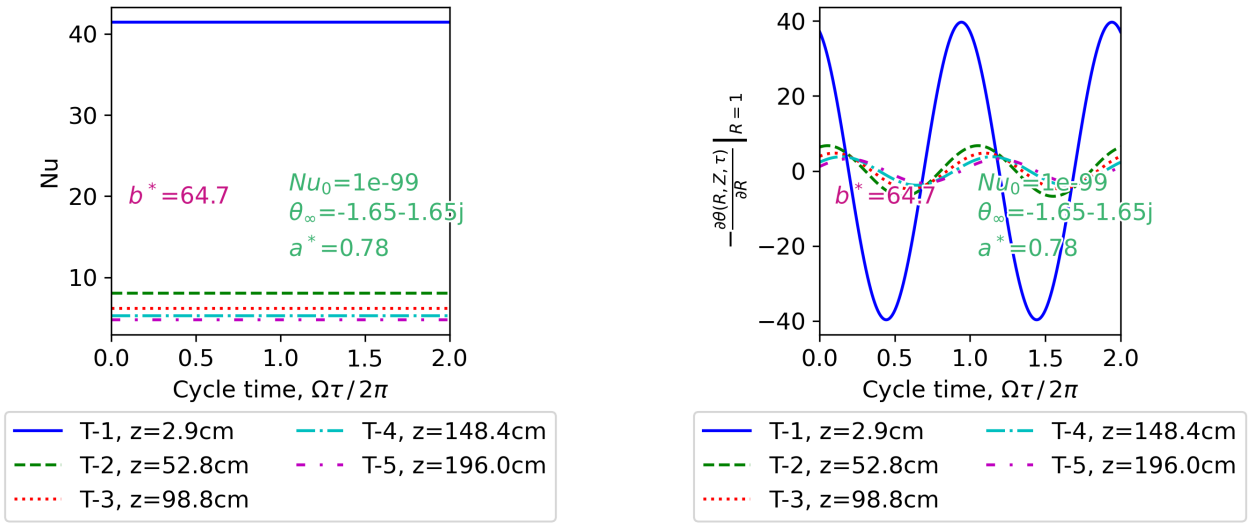
Figure A.11: Reynolds and Prandtl uncertainty distributions for range of nominal values. (f =0.1Hz, T_o =101°C, ΔT_o =38°C, 80kg/h, timestamp label: 0435P-111921)

Run parameters:
$f = 0.167\,\text{Hz}$, $T_o = 51°\text{C}$, $\Delta T_o = 19°\text{C}$, $76\,\text{kg/h}$, timestamp : 0452P − 111921



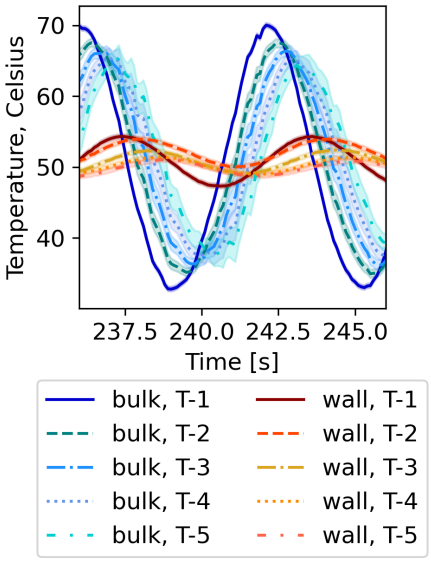
(a) (b)

Figure A.12: Heat transfer coefficient and Nusselt measurements for experimental run (f =0.167Hz, T_o =51°C, ΔT_o =19°C, 76kg/h, timestamp label: 0452P-111921)

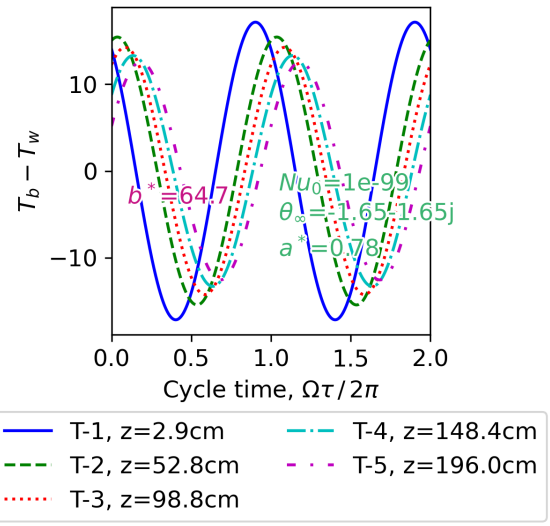


(a) (b)

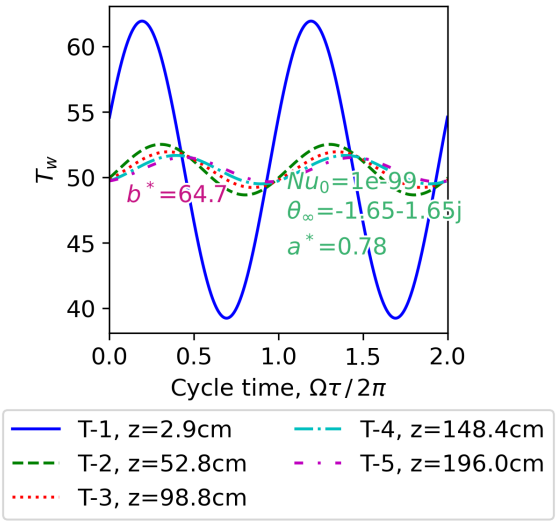
Figure A.13: Transient laminar model-predicted Nusselt numbers and dimensionless heat flux for experimental run (f =0.167Hz, T_o =51°C, ΔT_o =19°C, 76kg/h, timestamp label: 0452P-111921)



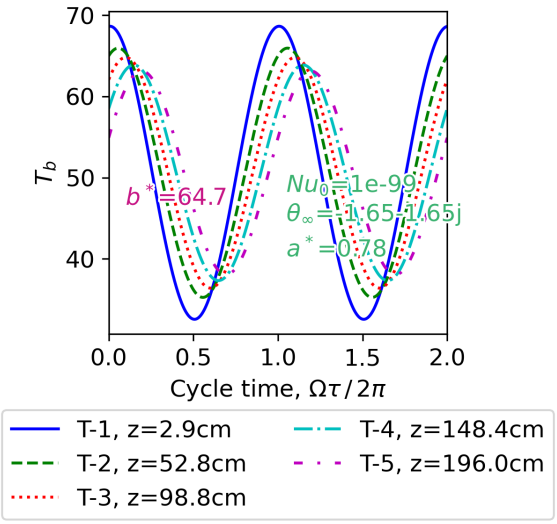
(a) Experimental measurements with uncertainty.



(b) Transient model-predicted delta between wall and local bulk temperature.

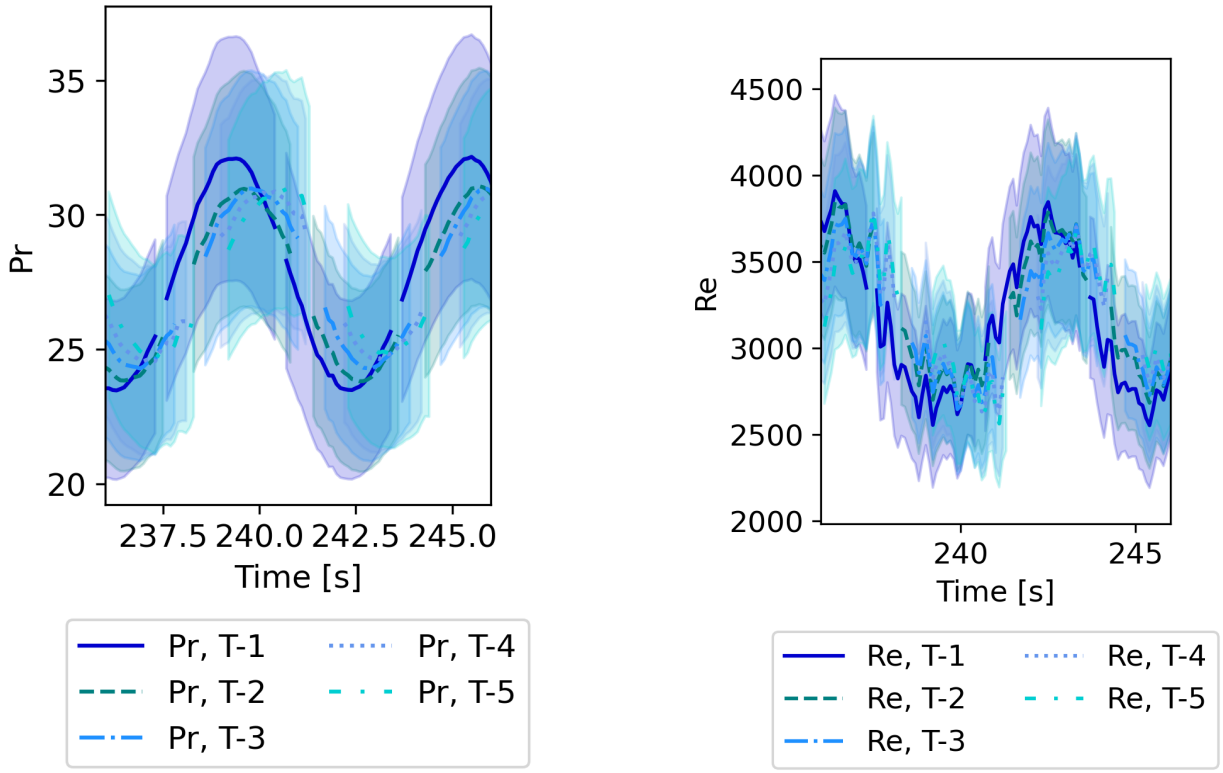


(c) Transient laminar model-predicted wall temperature.



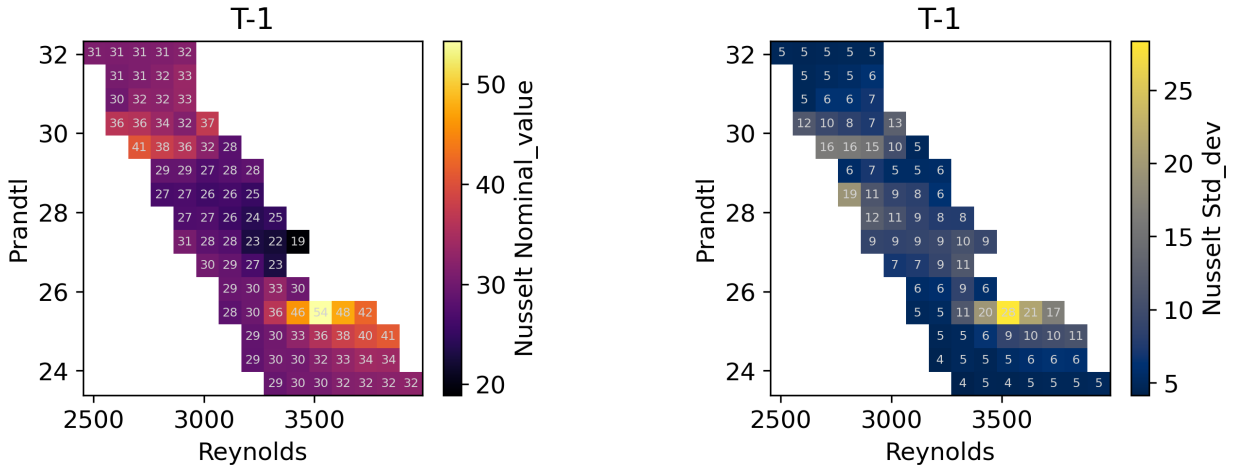
(d) Transient laminar model-predicted bulk temperature.

Figure A.14: Bulk temperature and wall temperature measurements and transient laminar model predictions for experimental run (f =0.167Hz, T_o =51°C, ΔT_o =19°C, 76kg/h, timestamp label: 0452P-111921)



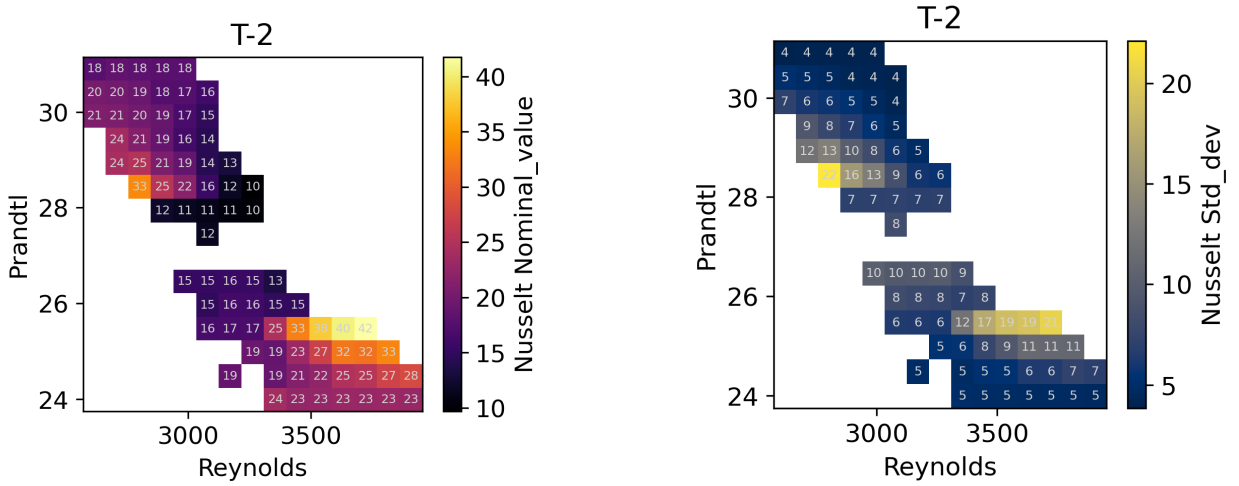
(a) (b)

Figure A.15: Prandtl and Reynolds measurements for experimental run (f =0.167Hz, T_o =51°C, ΔT_o =19°C, 76kg/h, timestamp label: 0452P-111921)



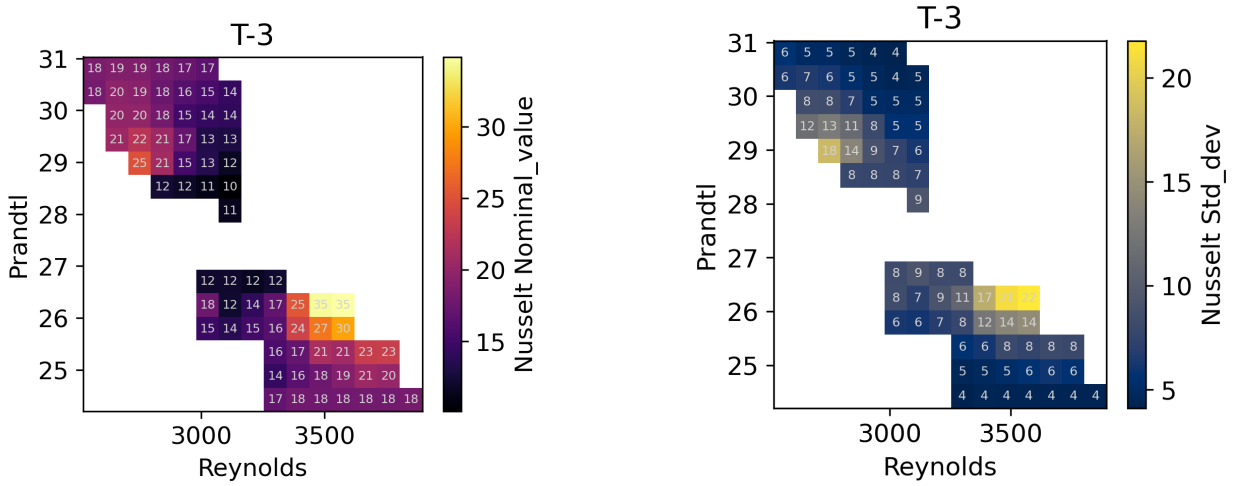
(a) (b)

Figure A.16: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-1. (f =0.167Hz, T_o =51°C, ΔT_o =19°C, 76kg/h, timestamp label: 0452P-111921)



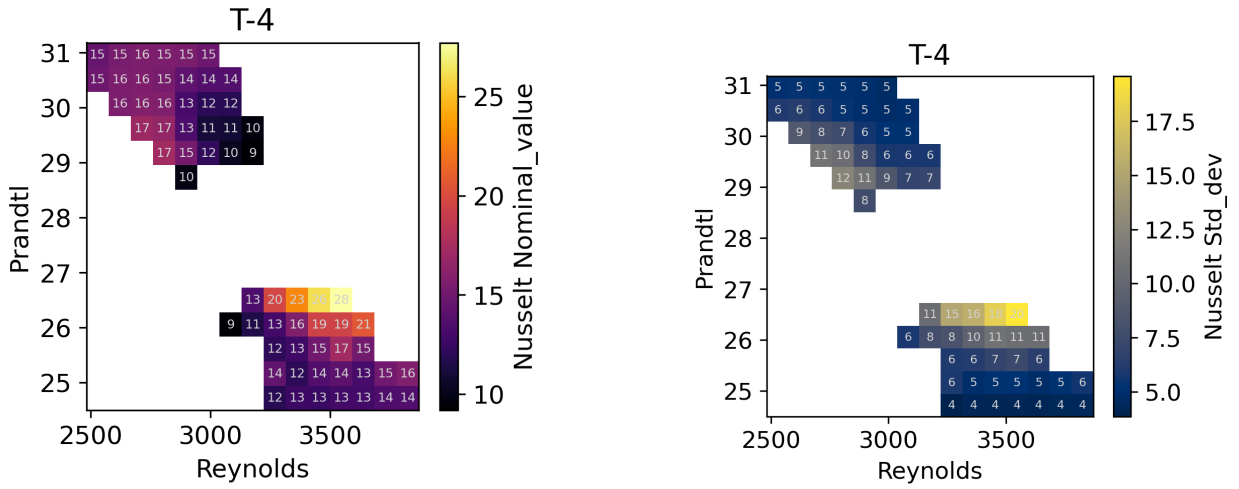
(a) (b)

Figure A.17: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-2. (f =0.167Hz, T_o =51°C, ΔT_o =19°C, 76kg/h, timestamp label: 0452P-111921)



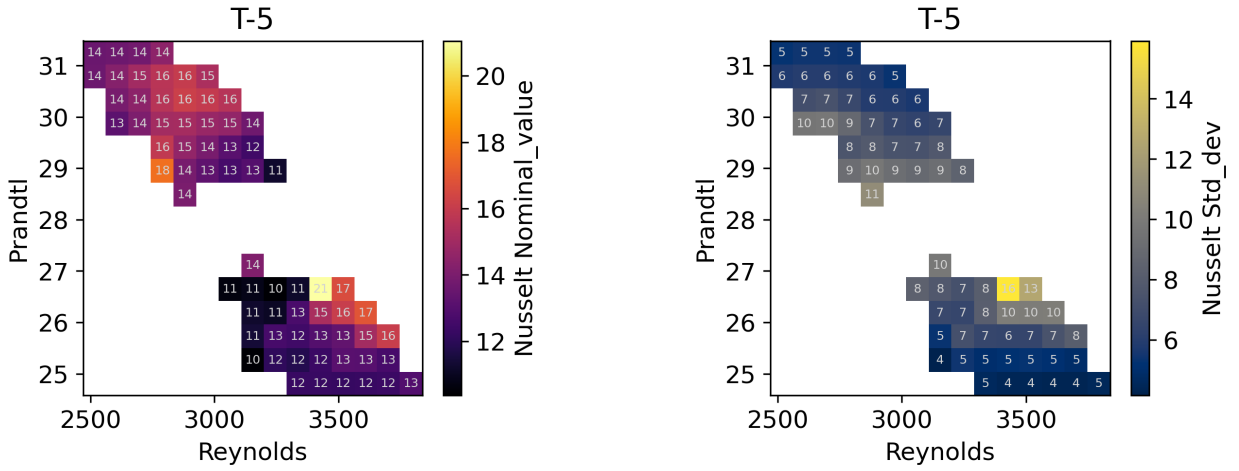
(a) (b)

Figure A.18: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-3. (f =0.167Hz, T_o =51°C, ΔT_o =19°C, 76kg/h, timestamp label: 0452P-111921)



(a) (b)

Figure A.19: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-4. (f =0.167Hz, T_o =51°C, ΔT_o =19°C, 76kg/h, timestamp label: 0452P-111921)



(a) (b)

Figure A.20: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-5. (f =0.167Hz, T_o =51°C, ΔT_o =19°C, 76kg/h, timestamp label: 0452P-111921)

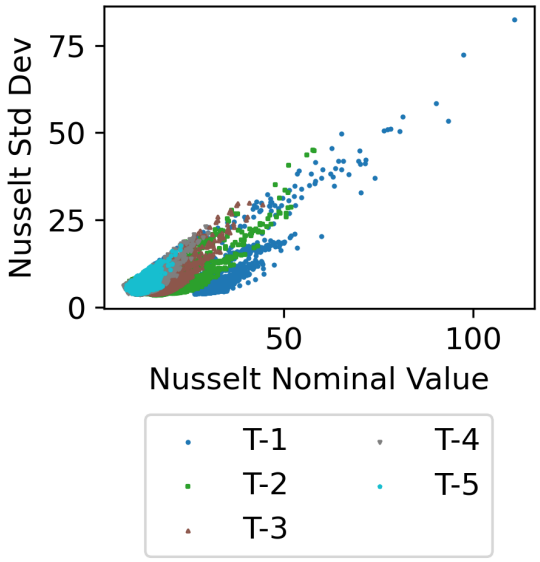
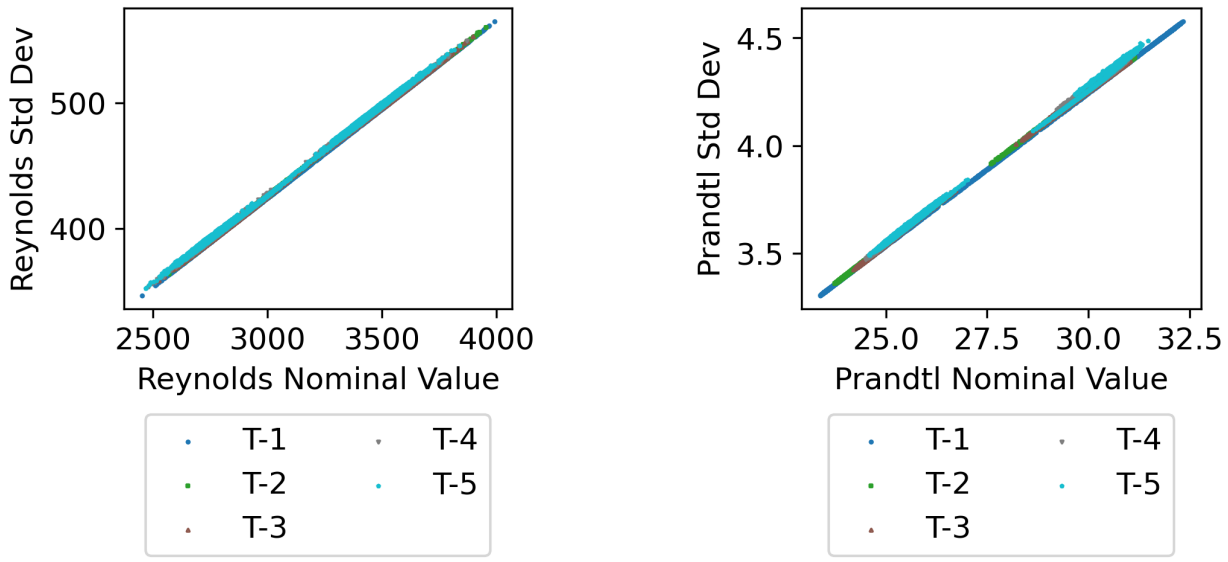


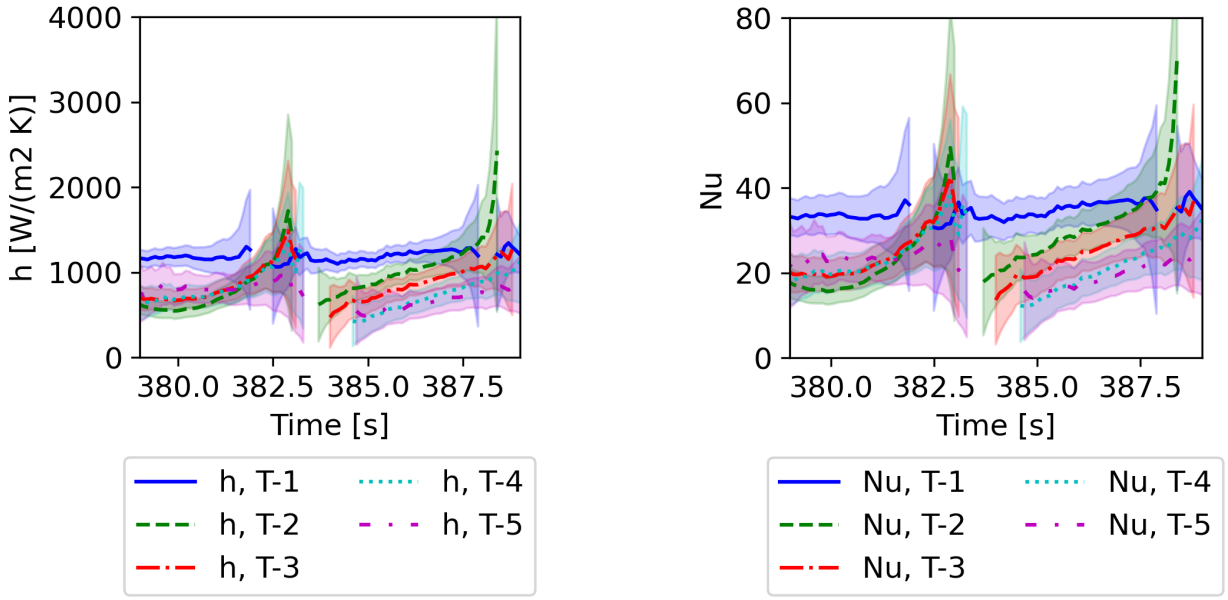
Figure A.21: Nusselt uncertainty distribution for range of nominal values. (f =0.167Hz, T_o =51°C, ΔT_o =19°C, 76kg/h, timestamp label: 0452P-111921)



(a) (b)

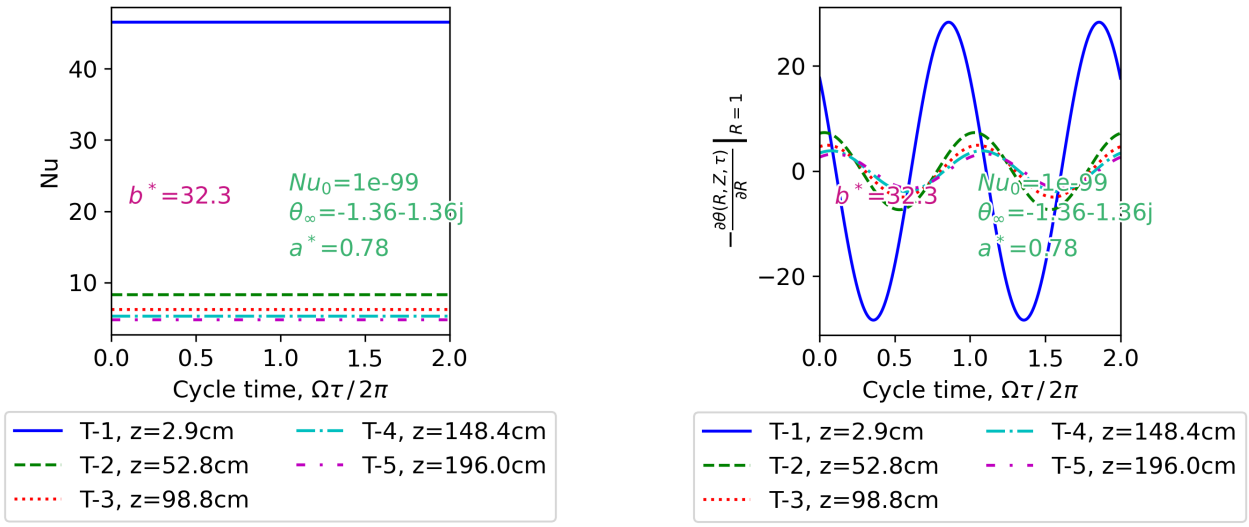
Figure A.22: Reynolds and Prandtl uncertainty distributions for range of nominal values. (f =0.167Hz, T_o =51°C, ΔT_o =19°C, 76kg/h, timestamp label: 0452P-111921)

Run parameters: $f = 0.0833\,\text{Hz}$, $T_o = 51°\text{C}$, $\Delta T_o = 23°\text{C}$, $76\,\text{kg/h}$, timestamp : 0508P − 111921



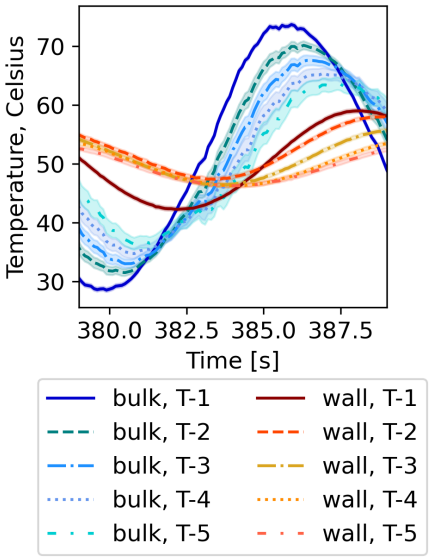
(a) (b)

Figure A.23: Heat transfer coefficient and Nusselt measurements for experimental run (f =0.0833Hz, T_o =51°C, ΔT_o =23°C, 76kg/h, timestamp label: 0508P-111921)

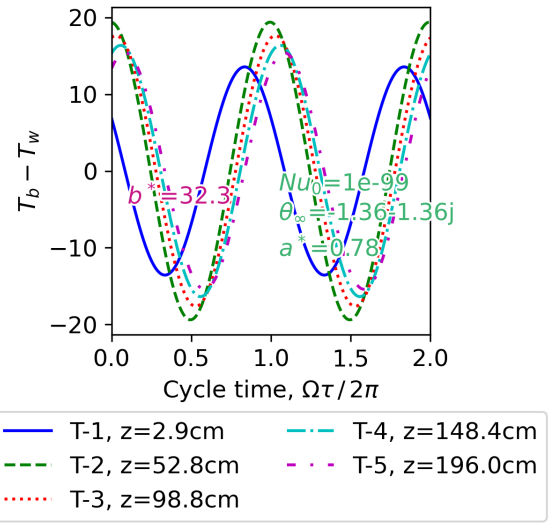


(a) (b)

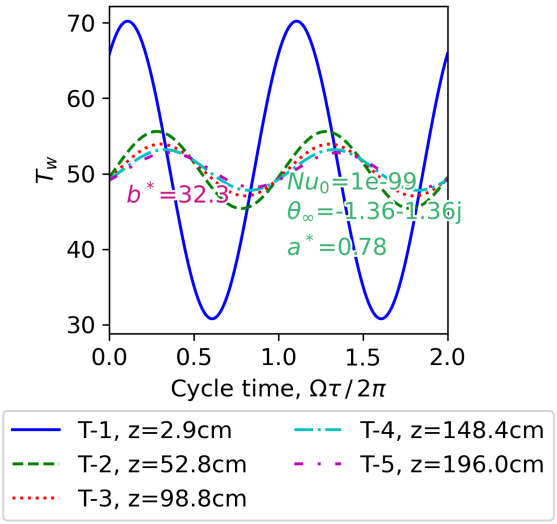
Figure A.24: Transient laminar model-predicted Nusselt numbers and dimensionless heat flux for experimental run (f =0.0833Hz, T_o =51°C, ΔT_o =23°C, 76kg/h, timestamp label: 0508P-111921)



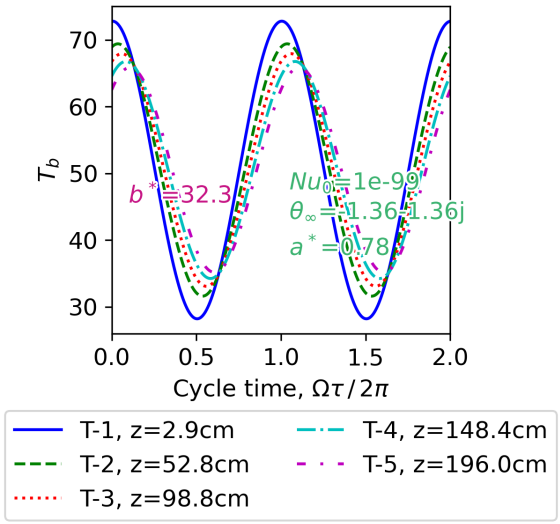
(a) Experimental measurements with uncertainty.



(b) Transient model-predicted delta between wall and local bulk temperature.

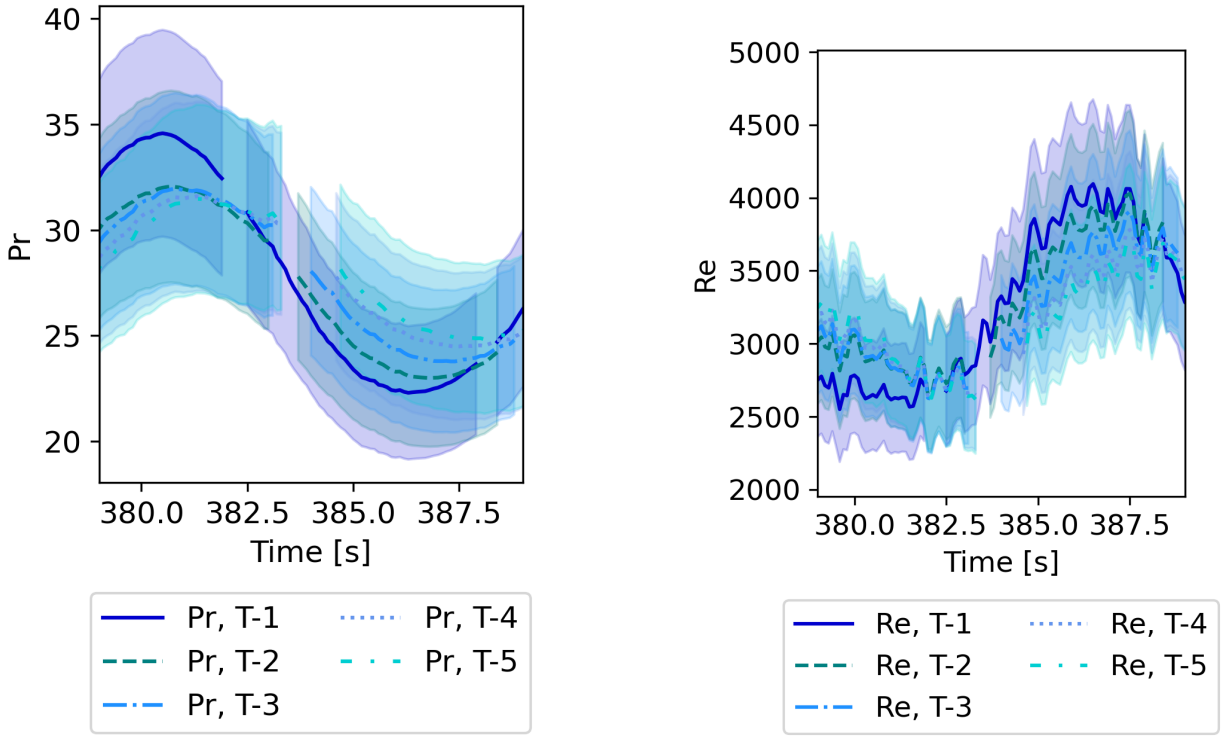


(c) Transient laminar model-predicted wall temperature.



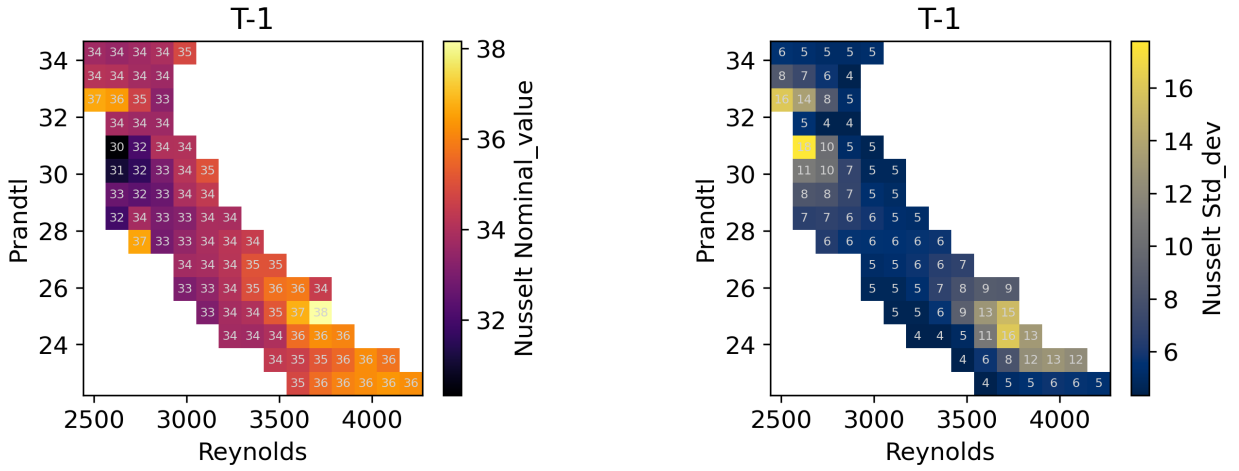
(d) Transient laminar model-predicted bulk temperature.

Figure A.25: Bulk temperature and wall temperature measurements and transient laminar model predictions for experimental run (f =0.0833Hz, T_o =51°C, ΔT_o =23°C, 76kg/h, timestamp label: 0508P-111921)



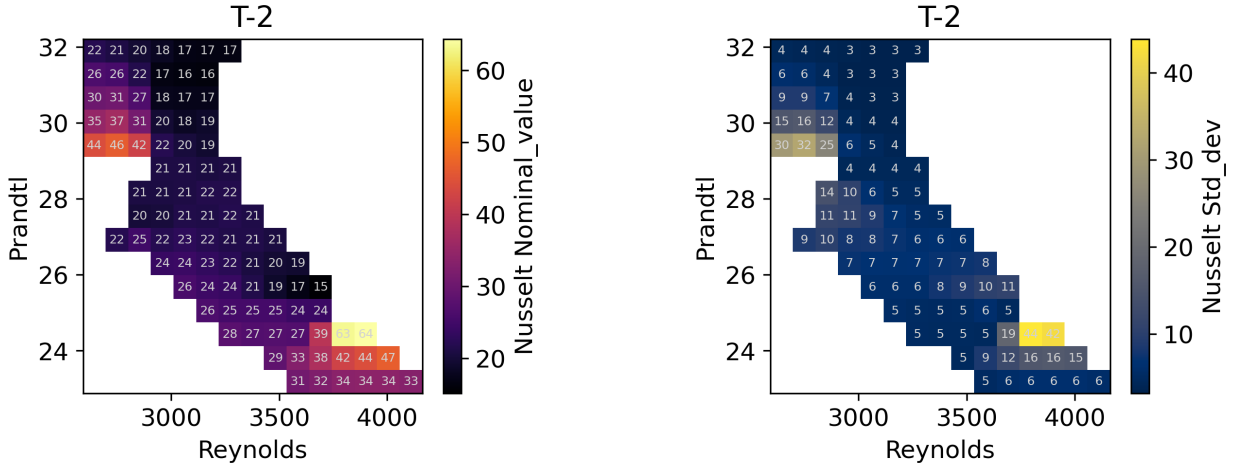
(a) (b)

Figure A.26: Prandtl and Reynolds measurements for experimental run (f =0.0833Hz, T_o =51°C, ΔT_o =23°C, 76kg/h, timestamp label: 0508P-111921)



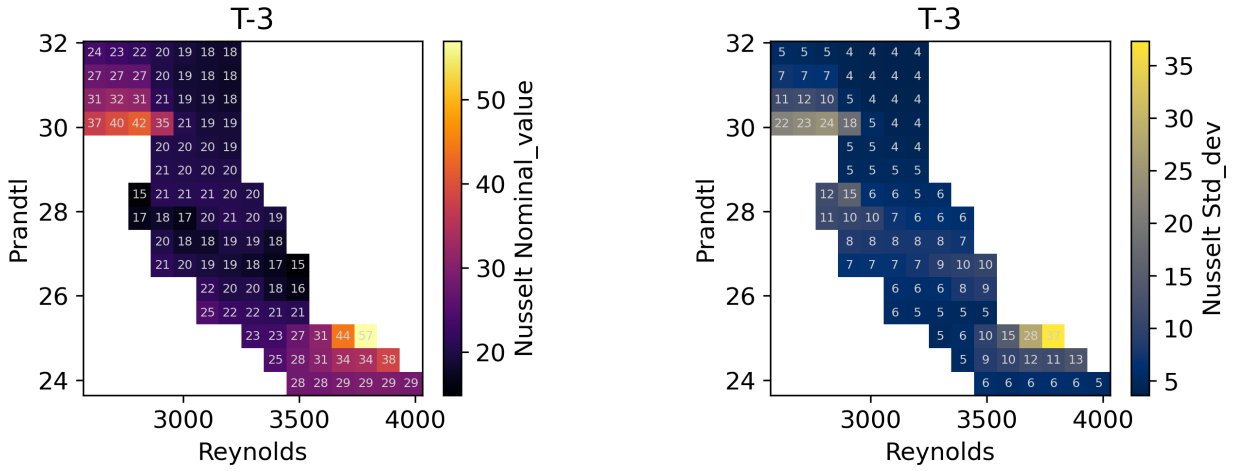
(a) (b)

Figure A.27: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-1. (f =0.0833Hz, T_o =51°C, ΔT_o =23°C, 76kg/h, timestamp label: 0508P-111921)



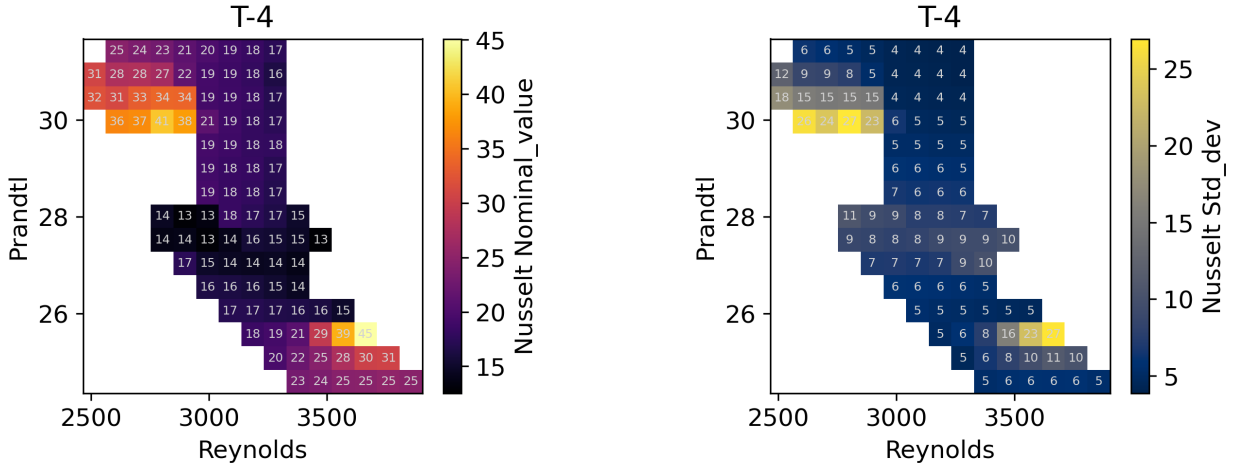
(a) (b)

Figure A.28: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-2. (f =0.0833Hz, T_o =51°C, ΔT_o =23°C, 76kg/h, timestamp label: 0508P-111921)



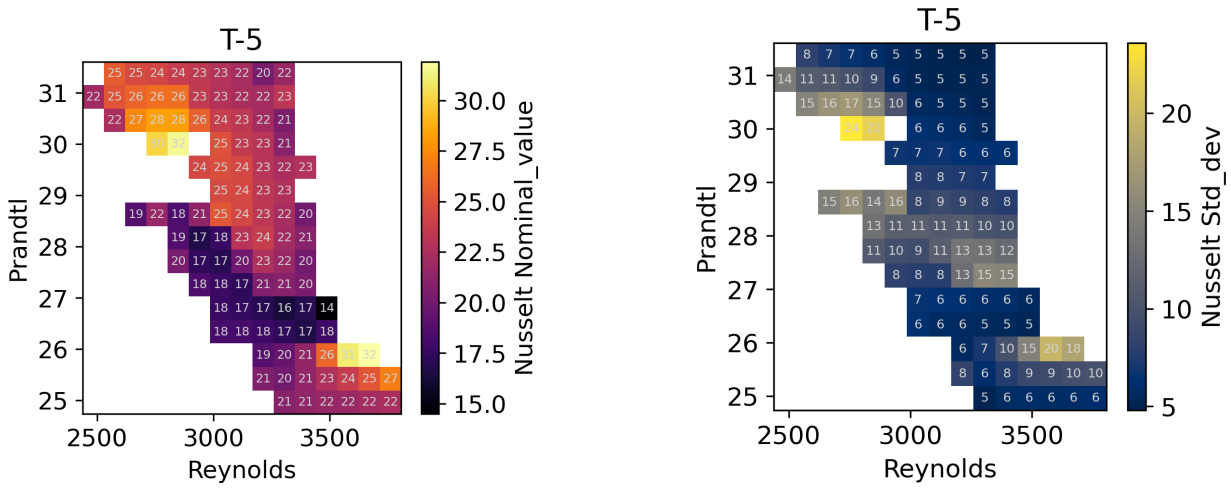
(a) (b)

Figure A.29: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-3. (f =0.0833Hz, T_o =51°C, ΔT_o =23°C, 76kg/h, timestamp label: 0508P-111921)



(a) (b)

Figure A.30: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-4. (f =0.0833Hz, T_o =51°C, ΔT_o =23°C, 76kg/h, timestamp label: 0508P-111921)



(a) (b)

Figure A.31: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-5. (f =0.0833Hz, T_o =51°C, ΔT_o =23°C, 76kg/h, timestamp label: 0508P-111921)

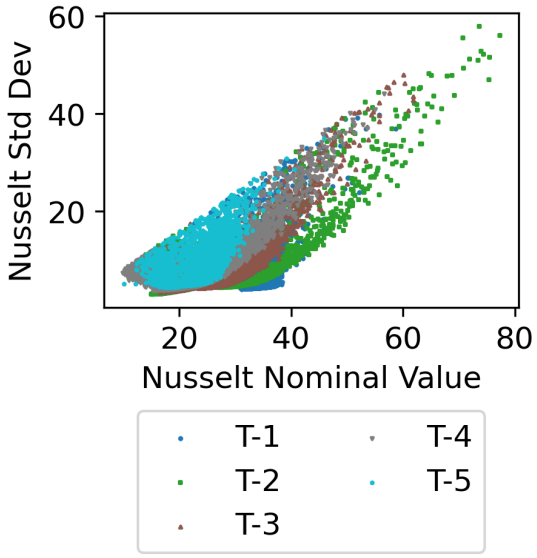
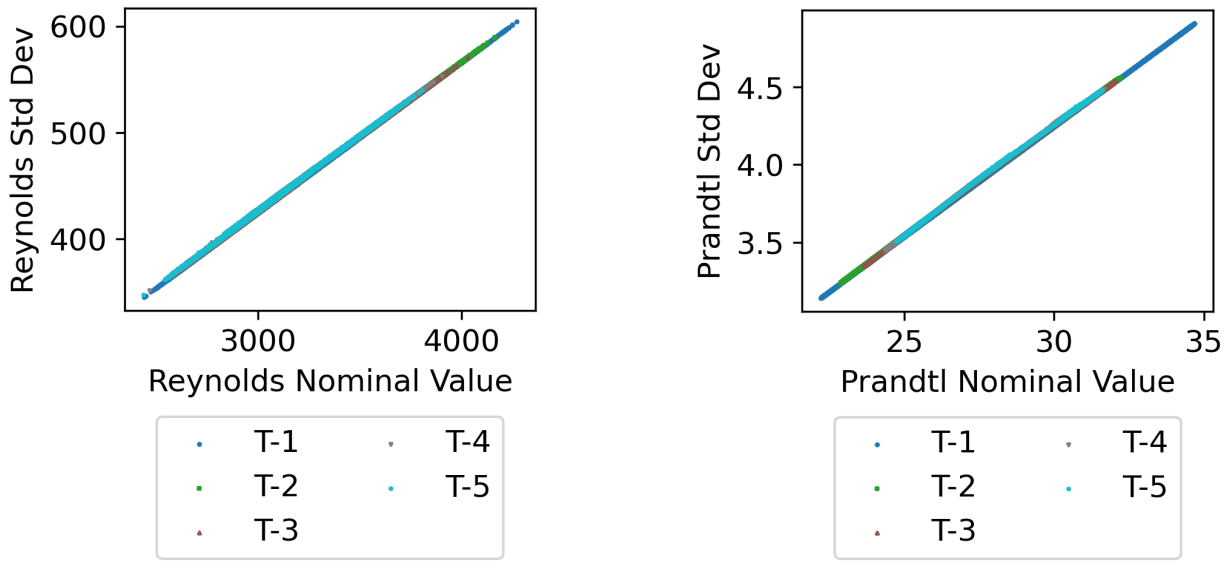


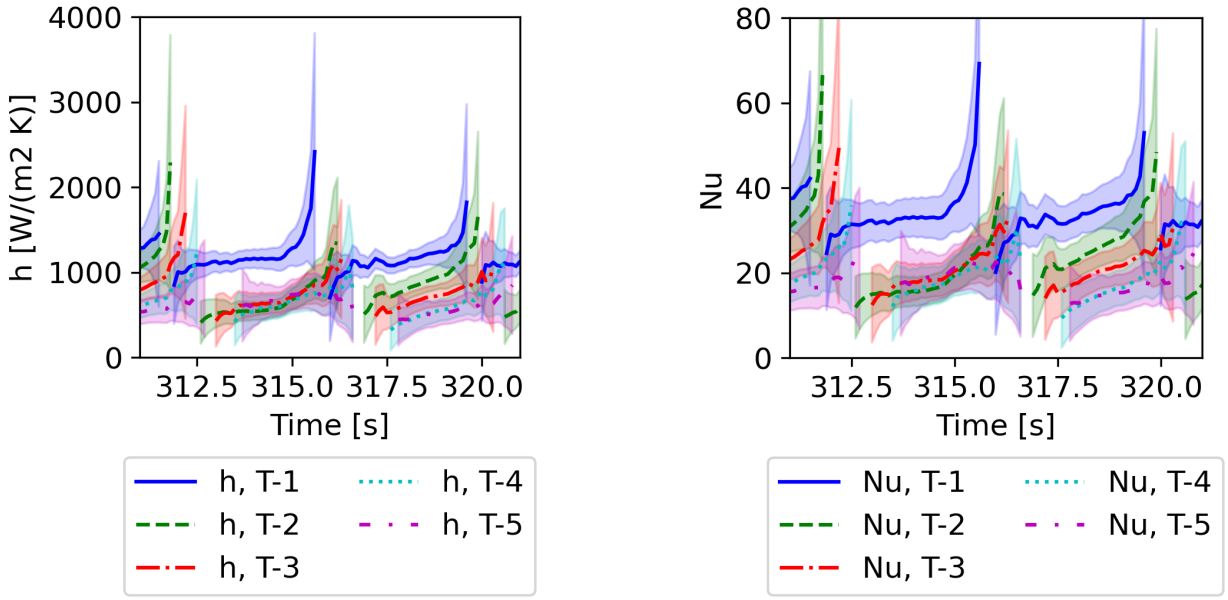
Figure A.32: Nusselt uncertainty distribution for range of nominal values. (f =0.0833Hz, T_o =51°C, ΔT_o =23°C, 76kg/h, timestamp label: 0508P-111921)



(a) (b)

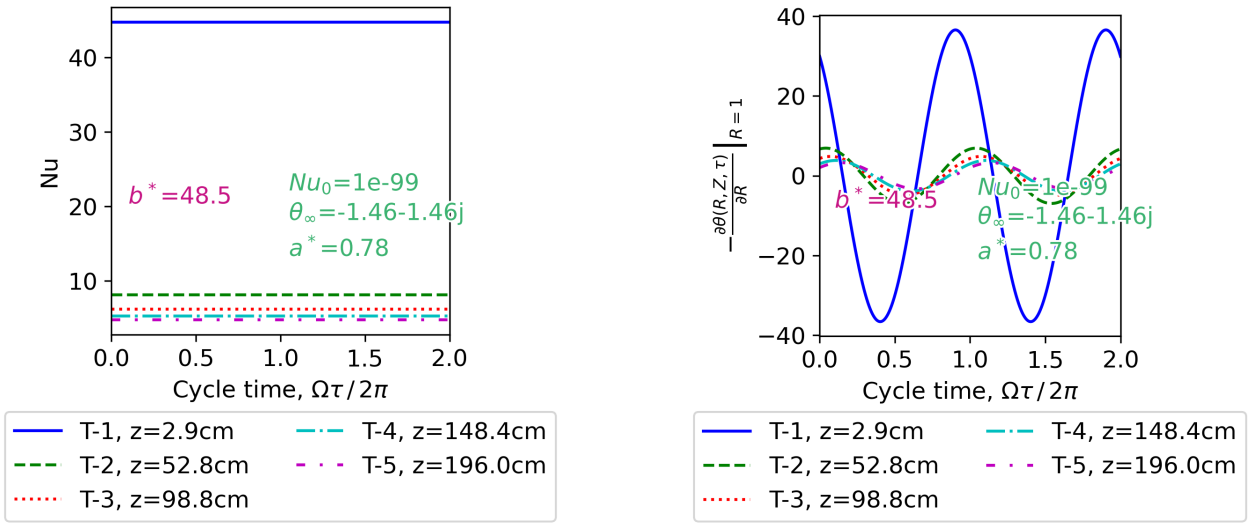
Figure A.33: Reynolds and Prandtl uncertainty distributions for range of nominal values. (f =0.0833Hz, T_o =51°C, ΔT_o =23°C, 76kg/h, timestamp label: 0508P-111921)

Run parameters:
$f = 0.125\,\text{Hz}$, $T_o = 50°\text{C}$, $\Delta T_o = 21°\text{C}$, $77\,\text{kg/h}$, timestamp : 0528P − 111921



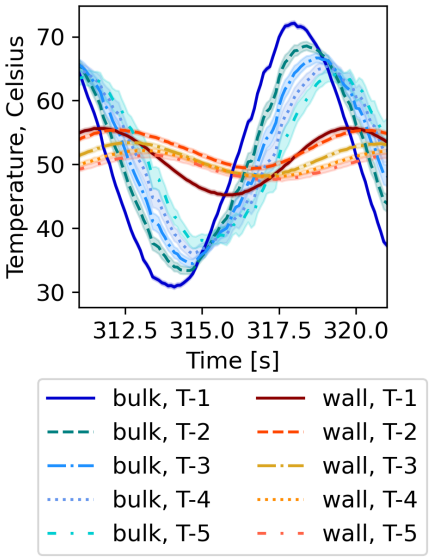
(a) (b)

Figure A.34: Heat transfer coefficient and Nusselt measurements for experimental run (f =0.125Hz, T_o =50°C, ΔT_o =21°C, 77kg/h, timestamp label: 0528P-111921)

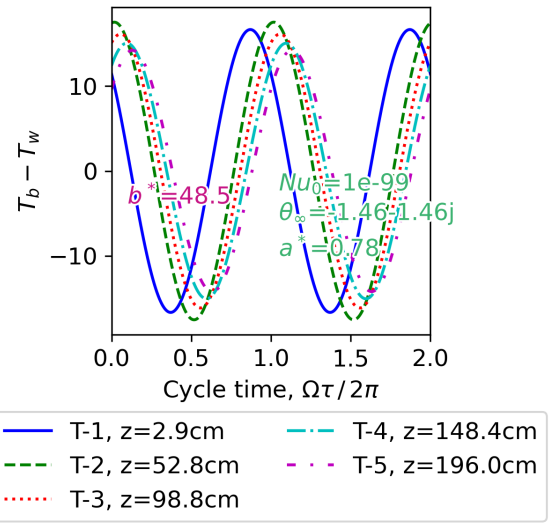


(a) (b)

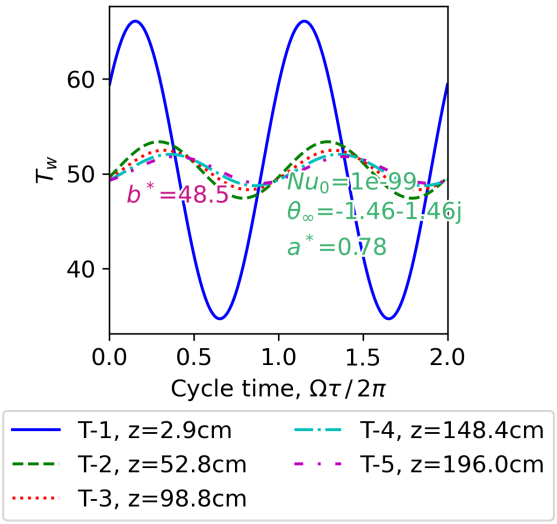
Figure A.35: Transient laminar model-predicted Nusselt numbers and dimensionless heat flux for experimental run (f =0.125Hz, T_o =50°C, ΔT_o =21°C, 77kg/h, timestamp label: 0528P-111921)



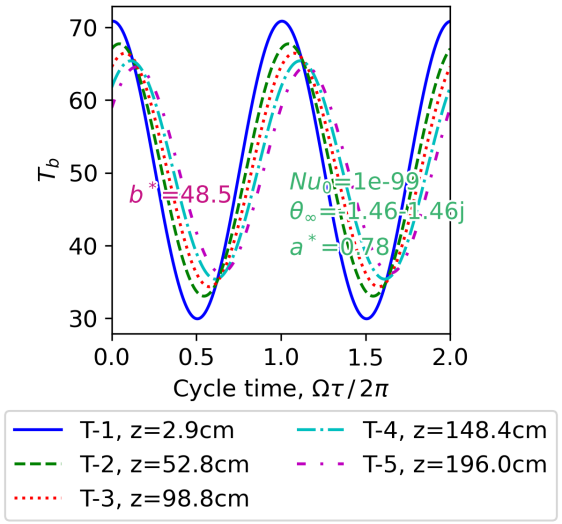
(a) Experimental measurements with uncertainty.



(b) Transient model-predicted delta between wall and local bulk temperature.

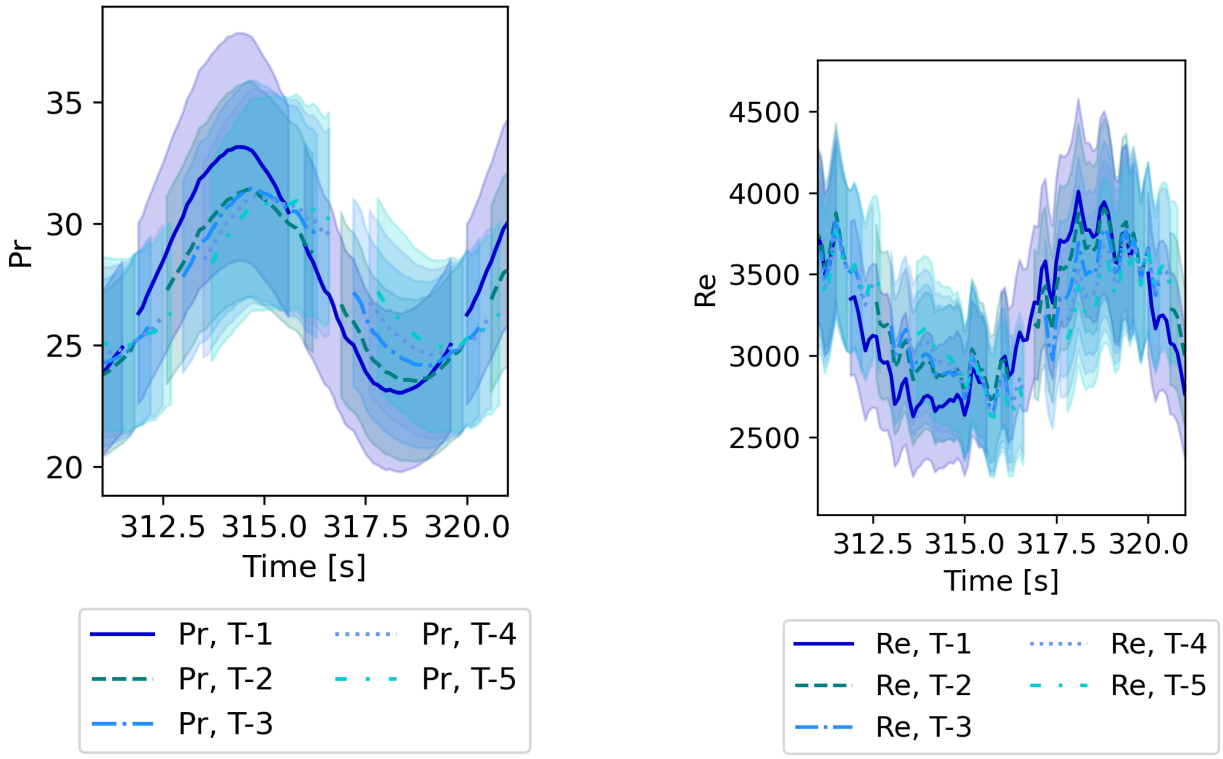


(c) Transient laminar model-predicted wall temperature.



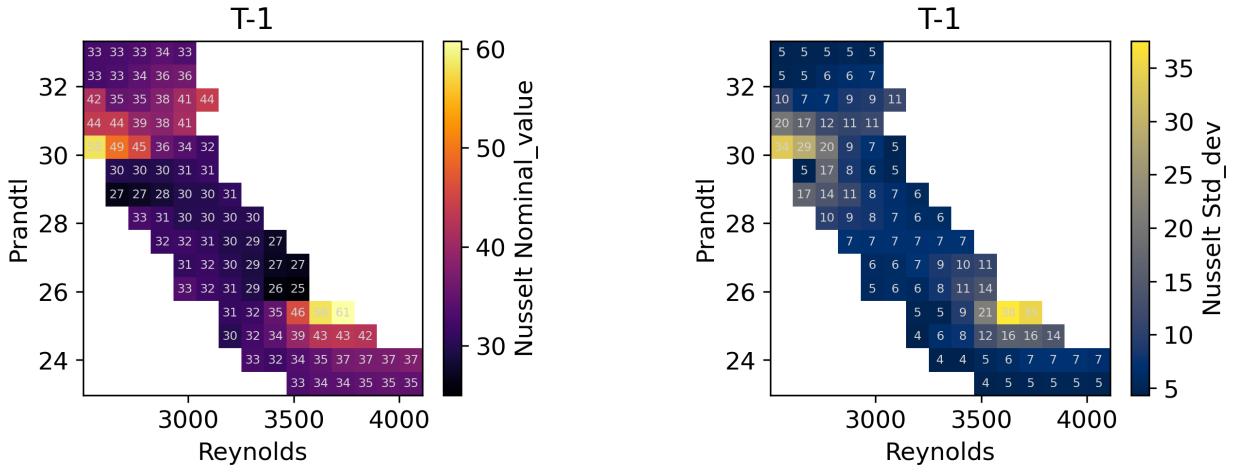
(d) Transient laminar model-predicted bulk temperature.

Figure A.36: Bulk temperature and wall temperature measurements and transient laminar model predictions for experimental run (f =0.125Hz, T_o =50°C, ΔT_o =21°C, 77kg/h, timestamp label: 0528P-111921)



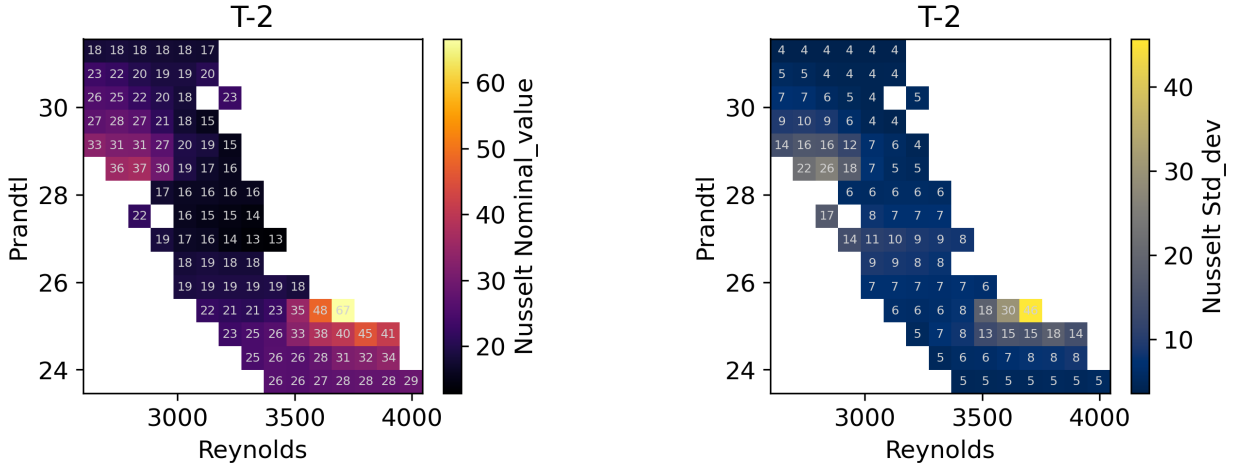
(a) (b)

Figure A.37: Prandtl and Reynolds measurements for experimental run (f =0.125Hz, T_o =50°C, ΔT_o =21°C, 77kg/h, timestamp label: 0528P-111921)



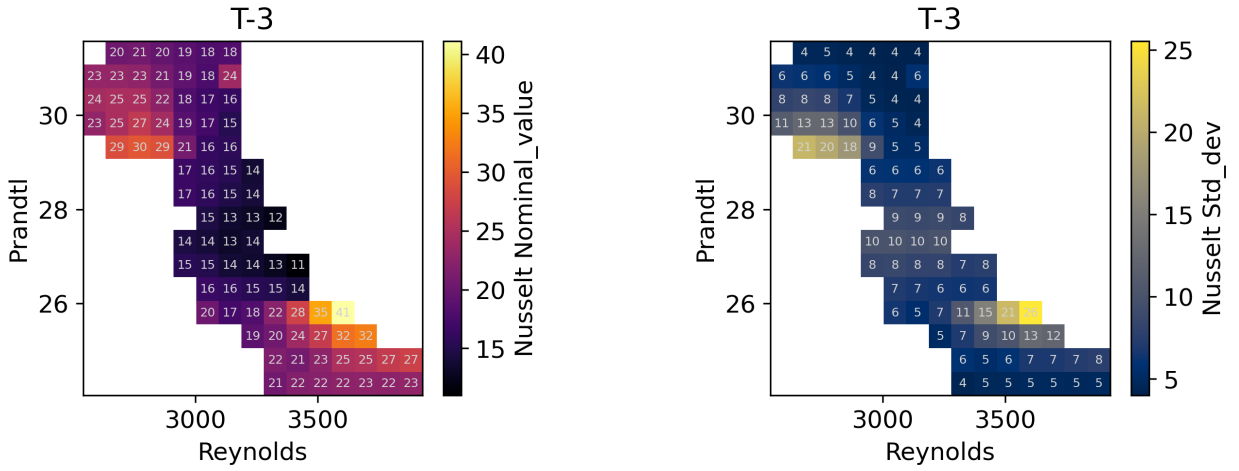
(a) (b)

Figure A.38: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-1. (f =0.125Hz, T_o =50°C, ΔT_o =21°C, 77kg/h, timestamp label: 0528P-111921)



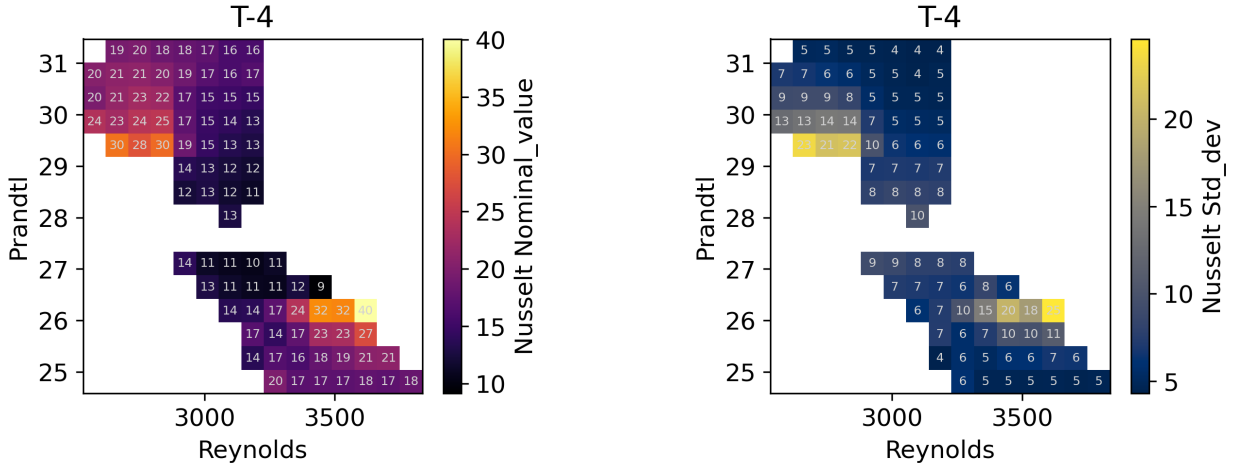
(a) (b)

Figure A.39: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-2. (f =0.125Hz, T_o =50°C, ΔT_o =21°C, 77kg/h, timestamp label: 0528P-111921)



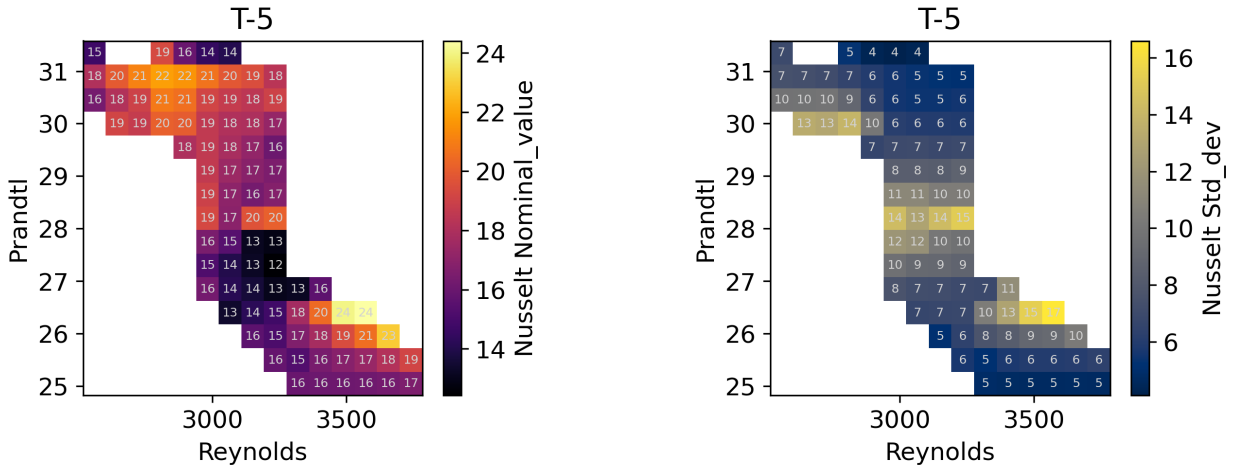
(a) (b)

Figure A.40: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-3. (f =0.125Hz, T_o =50°C, ΔT_o =21°C, 77kg/h, timestamp label: 0528P-111921)



(a) (b)

Figure A.41: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-4. (f =0.125Hz, T_o =50°C, ΔT_o =21°C, 77kg/h, timestamp label: 0528P-111921)



(a) (b)

Figure A.42: Nusselt measurements and uncertainty over the range of the experimental run Reynolds and Prandtl values at node location T-5. (f =0.125Hz, T_o =50°C, ΔT_o =21°C, 77kg/h, timestamp label: 0528P-111921)

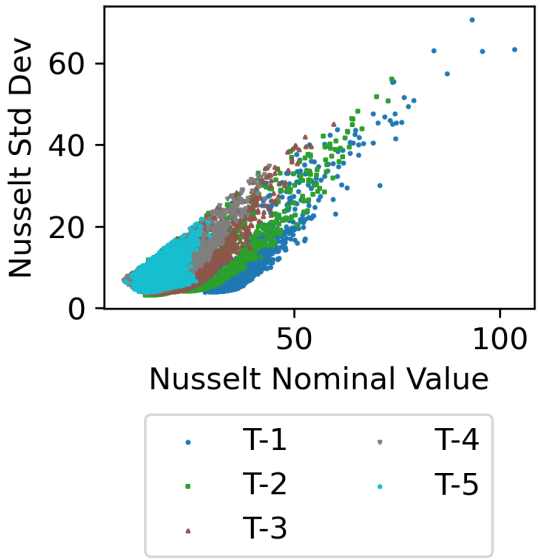
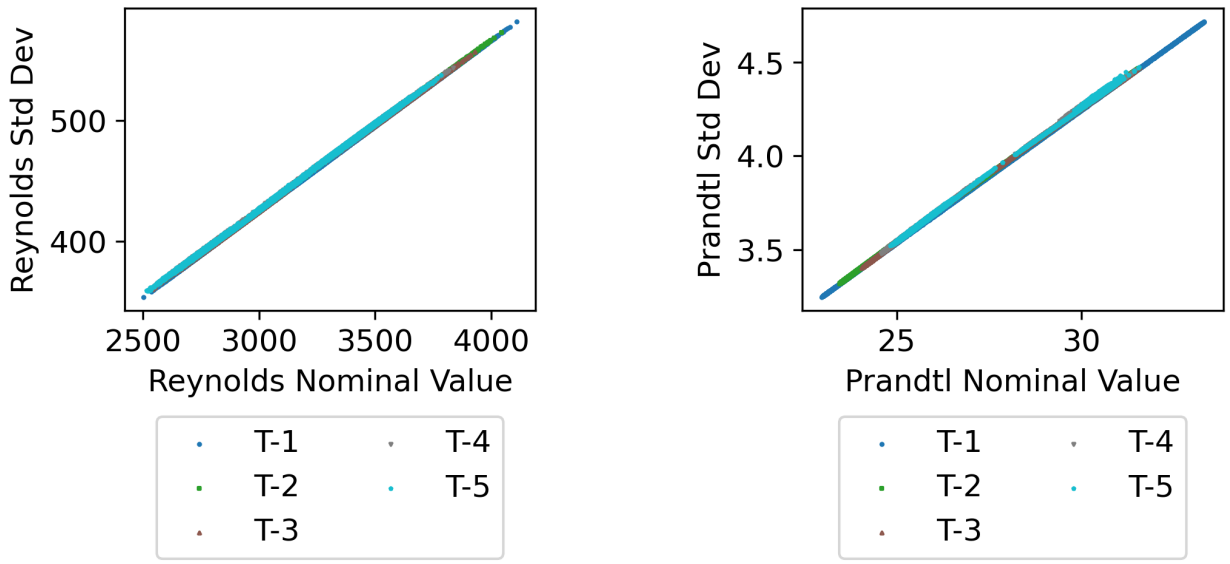


Figure A.43: Nusselt uncertainty distribution for range of nominal values. (f =0.125Hz, T_o =50°C, ΔT_o =21°C, 77kg/h, timestamp label: 0528P-111921)



(a) (b)

Figure A.44: Reynolds and Prandtl uncertainty distributions for range of nominal values. (f =0.125Hz, T_o =50°C, ΔT_o =21°C, 77kg/h, timestamp label: 0528P-111921)

Appendix B

Analog Power Control Programming in LabVIEW

In this section, further details are provided on how the desired heater input signals are generated. On the front panel of the VI, there are three options for controlling heater power. Manual control is implemented by setting the heater power value in Watts, by typing a value between 0 and 10,000 under "Manual Power Control". "Power Profile" allows the user to input a pre-made CSV file of their choice, which dictates the heater power values over time. "Pulse Control" is similar to "Power Profile" mode, except users are able to create and edit a custom periodic power profile "on the fly" (i.e., during actual operation), instead of needing to create an Excel file ahead of time.

By switching between the three options on the front panel, the state of the LabVIEW case structure changes to produce the desired functionality.

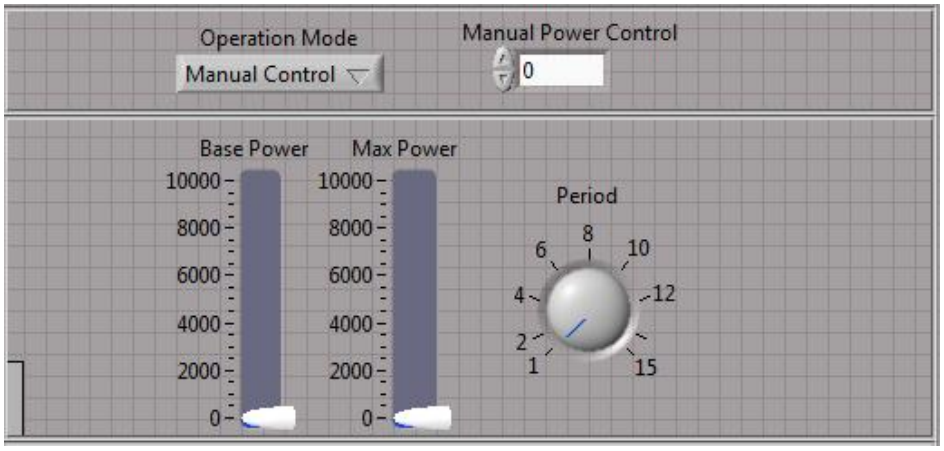


Figure B.1: Desired Power controls on the front panel in LabVIEW.

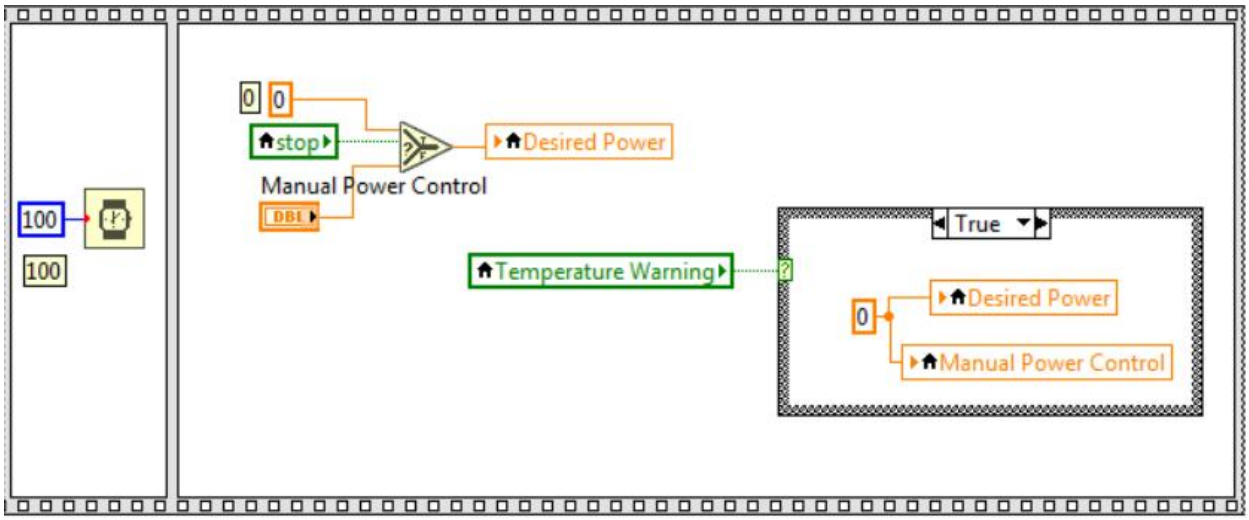


Figure B.2: Manual Control Mode in LabVIEW.

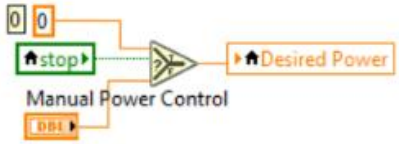


Figure B.3: Component 1 of Manual Control Mode in LabVIEW.

Manual Control

This operation mode is straightforward. Inside the case structure is a sequence structure. In the first panel of the sequence structure, a 100-millisecond wait statement ensures that there is a brief pause between signals sent to the heater, so as not to bog down the VI and to keep consistent with the 100-millisecond measurement intervals. This wait statement could be changed as desired to achieve higher resolutions. Inside the second panel of the sequence structure are two components. Figure X. shows the first component.

Component 1 feeds the user-typed value of "Manual Power Control" into the local variable Desired Power, so it can be sent to the "Analog Read-and-Write Loop" and fed into the PB-HTX Analog Sub-VI. However, if the stop button has been pressed and a "stop" signal was sent out, this component will instead feed a value of 0 to Desired Power in order to shut the heater off.

Figure B.3 shows Component 2 of Manual Control Mode which only activates if there is a temperature warning message sent out by the Data Logging Loop. (The data logging loop includes programming that reads the thermocouple measurements every iteration to ensure that no temperature values have exceeded a set maximum, and if any have exceeded

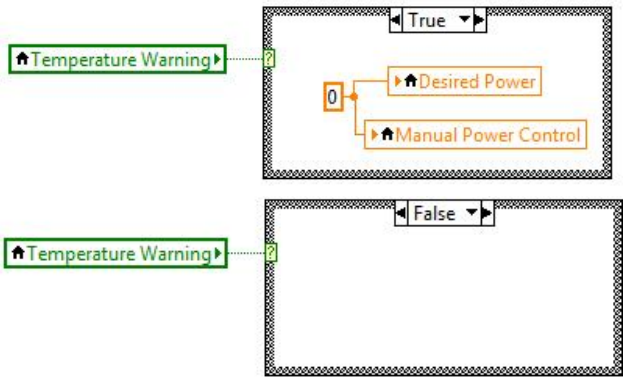


Figure B.4: Component 2 of Manual Control Mode in LabVIEW.

the maximum, it sends out the warning message.) Component 2 is a case structure which is empty if the Boolean "Temperature Warning" is false, but if it is true, code appears inside the case structure which feeds a value of 0 to Desired Power as well as automatically changing the value of the user-controlled "Manual Power Control" field to 0.

Import Power Profile

Unlike Manual Control, the Power Profile functionality makes use of a for loop. This for loop is used to iterate over each element in an array of heater power values from an Excel or CSV file imported by the user, with a wait statement used to set the time step. Component 1 of Power Profile mode, shown in Figure B.6, is very similar to Component 1 from Manual Control mode.

The main difference is that the value being fed to "Desired Power" is taken from the Excel file chosen by the user, and this value is updated continuously by the for loop as it iterates. Once the for loop has run through all of the values in the Excel file, it terminates the for loop, which has the effect of restarting the process all over again from the first index.

Component 2 of Power Profile mode, shown in Figure B.7, is also similar to Component 2 from Manual Control with one key difference. Instead of sending a "0" to Desired Power when Temperature Warning is true, it changes the Operation Mode to Manual Control, so that Manual Control can handle the temperature shutdown. This is because if the user had been alternating between Manual Control mode and the other two modes, and a nonzero value typed into Manual Control when the Temperature Warning was issued, it is useful for the program to set the value under Manual Control to 0 before shutting down so that it is clear that the power being sent to the heater is actually 0 after shutdown. Since Manual Control mode already has this functionality, it makes sense for the Power Profile to pass control of the high temperature safety shutdown to Manual Control.

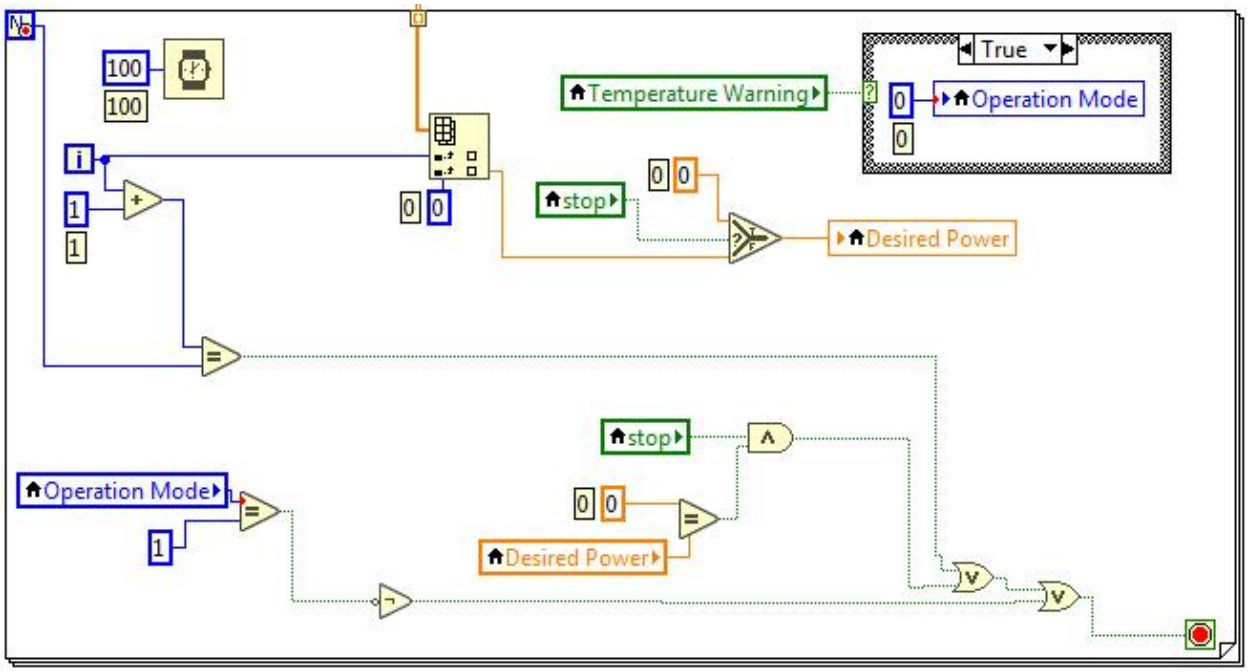


Figure B.5: Power Profile Mode in LabVIEW.

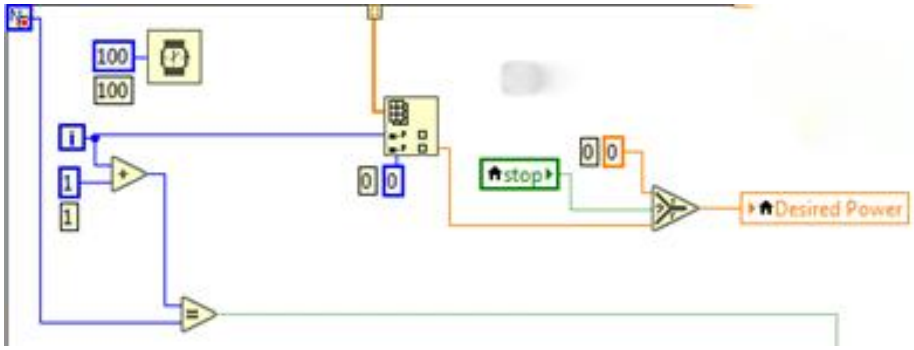


Figure B.6: Component 1 of Power Profile Mode in LabVIEW.

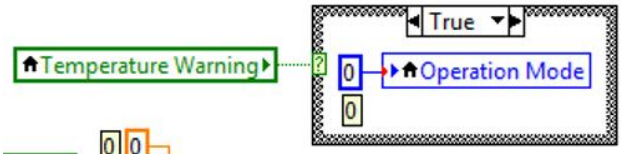


Figure B.7: Component 2 of Power Profile Mode in LabVIEW.

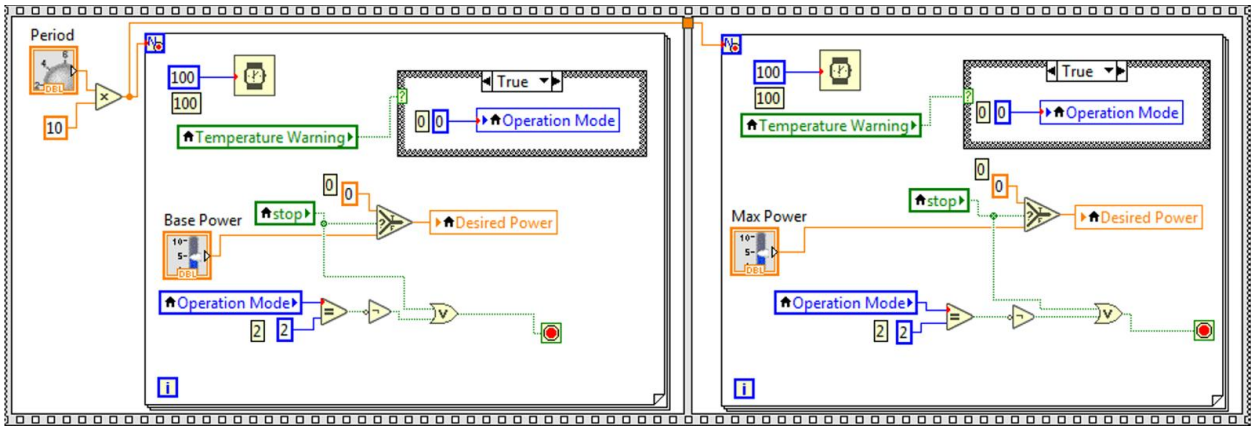


Figure B.8: Pulse Control Mode in LabVIEW.

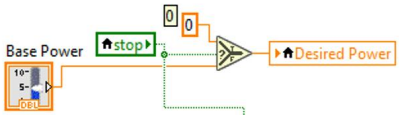


Figure B.9: Component 1 of Pulse Control Mode in LabVIEW.

Pulse Control

At this time, the Pulse Control functionality is only able to support square waves. For this reason, it makes use of a sequence structure with two panels, where the first panel represents the "valleys" of the square waves and the second panel represents the "crests" of the square waves. This is why the two panels look similar, and the only difference between the two is the use of the variable "Base Power" in the first panel and "Max Power" in the second.

The user can control three key variables in Pulse Control mode: Base Power (the power level of the square wave "valleys"), Max Power (the power level of the square wave "crests"), and Period, the period of the wave.

This Operation Mode works almost exactly the same as Power Profile mode. Component 1 of Pulse Control looks almost the same as Component 1 of Power Profile except that instead of values for Desired Power being fed in from an Excel spreadsheet, they are instead fed in from the Base Power variable, which can be changed at any time by the user during operation.

In the first panel of the sequence structure, the value of "Base Power" is fed to Desired Power in 100-millisecond intervals by the for loop, for i = 1 to N, where N is the period specified by the user. It is worth noting that the user specifies the period in seconds, and the loop executes in 100-millisecond intervals, so Period is actually multiplied by 10 before being assigned to N. Once this process ends, control passes to the second panel of the sequence structure, where the process is repeated using "Max Power" instead of "Base Power". This

forms the crests of the square waves.

Appendix C

SHEFRA Experiment Data Reduction Code

```
'''
data reduction

objective:
calculate local heat transfer coefficient along test-section spatial dimension

'''
import matplotlib
from matplotlib import pyplot as plt
import numpy as np
import pandas as pd
import derivative
from uncertainties import ufloat, unumpy, UFloat
from scipy import signal
from copy import deepcopy
import os
import logging
import re

root_logger = logging.getLogger()
root_logger.setLevel(logging.INFO)

# Remove existing handlers and add a new StreamHandler with the desired format
for handler in root_logger.handlers[:]:
    root_logger.removeHandler(handler)

```

```
27 formatter = logging.Formatter('%(asctime)s - %(levelname)s - %(name)
      s - %(message)s')
28 stream_handler = logging.StreamHandler()
29 stream_handler.setFormatter(formatter)
30 root_logger.addHandler(stream_handler)
31
32
33 class Experiment:
34     def __init__(self, testsectionID):
35         self.logger = logging.getLogger(__name__)
36         self.logger.info("Experiment initialized")
37
38         match testsectionID:
39             case 'test_channel':
40                 radiusInner = 0.02
41                 thickness = 1e-3
42             case 'channel_1':
43                 # radiusInner = (0.12/2)*0.0254
44                 radiusInner=(0.25-(0.049*2))*0.0254/2
45                 # thickness = 0.065*0.0254
46                 thickness = 0.049*0.0254
47             case _:
48                 raise Exception("Test section ID not found")
49         self.__channel = Channel(radiusInner, thickness)
50
51     @property
52     def channel(self):
53         return self.__channel
54
55     def getMeasurements(self, path, frequency_in_Hz, startTime,
          endTime):
56         '''extracts measurement data from csv file given path, and
              start and end times in seconds'''
57         raw = pd.read_csv(path)
58         rawSlice = raw[(raw["Time"] >= startTime * 1000) & (raw["
              Time"] <= endTime * 1000)].copy()
59         rawSlice['Time'] = (rawSlice['Time']) / 1000
60         self.logger.debug("shifting wall TC measurements 200 ms
              ahead in time due to 300 ms response time")
61         # Shift the T-1, T-2, T-3, T-4, and T-5 columns 3 rows
              earlier
62         columns_to_shift = ["T-1", "T-2", "T-3", "T-4", "T-5"]
63         for col in columns_to_shift:
64             rawSlice[col] = rawSlice[col].shift(-2)
```

```
65
66          # Drop the last 3 rows to truncate the total number of time
               steps
67          rawSlice = rawSlice.iloc[:-2]
68
69          f = frequency_in_Hz
70          self.logger.debug('time range multiple of period?')
71          # mod=(time[-1]-time[0])%(1/f)
72          mod=(rawSlice['Time'].iloc[-1]-rawSlice['Time'].iloc[0])%(1/
               f)
73          if mod!=0:
74              self.logger.debug(False)
75              self.logger.debug('remainder = {}'.format(mod))
76              self.logger.debug('truncate {} seconds'.format(mod))
77              rows_to_truncate=round(mod/timeStep)
78              self.logger.debug('truncating data by {} rows, given a
                   time step of {} seconds'.format(rows_to_truncate,
                   timeStep))
79              rawSlice = rawSlice.iloc[:-rows_to_truncate]
80          else:
81              self.logger.debug(True)
82              self.logger.debug('time range {} is a multiple of the
                   period, {}'.format(rawSlice['Time'][-1]-rawSlice['
                   Time'][0], (1/f)))
83
84
85          self.__data = rawSlice[["Time", "T-1", "T-2", "T-3", "T-4",
               "T-5", "BT-inlet", "Flow rate"]]
86          self.logger.debug("measurements extracted from csv")
87          return self.__data
88
89      def plotTemperatureData(self, data):
90          data.plot(x="Time", y=["BT-inlet","T-1","T-2","T-3","T-4","T
               -5"], figsize=(4, 12), subplots=True, sharey=True)
91          plt.ylabel('Temperature [deg C]')
92          plt.xlabel('Time [s]')
93          data.plot(x="Time", y=["BT-inlet","T-1","T-2","T-3","T-4","T
               -5"], figsize=(16, 8))
94      def buildNodeFiniteDiff(self, data, nodeLabel):
95          # CO3-T uncertainty of Omega T-type thermocouples
96          utemp=1*0.341
97
98          # find index of requested node
99          nodeList=['T-1','T-2','T-3','T-4', 'T-5']
```

```
100         for i, n in enumerate(nodeList):
101             if nodeLabel==n:
102                 endNodeIndex=i
103         stepIndex=0
104         nodes=[]
105         heats=[]
106         while stepIndex<=endNodeIndex:
107             if nodeList[stepIndex]=='T-1':
108                 # extract thermocouple data with assoc. uncertainty
109                 fluidData = np.array([ufloat(t, utemp) for t in data
                        ['BT-inlet'].to_numpy()])
110                 fluid = Fluid(fluidData, data['Time'].to_numpy())
111             else:
112                 prevNode=nodes[stepIndex-1]
113                 prevHeat=heats[stepIndex-1]
114                 fluidEnergyBalance = FluidEnergyBalanceFiniteDiff(
                        prevNode, prevHeat, nodeList[stepIndex])
115                 fluid = Fluid(fluidEnergyBalance.localFluidTemp,
                        data['Time'].to_numpy())
116             solidData = np.array([ufloat(t, utemp) for t in data[
                    nodeList[stepIndex]].to_numpy()])
117             solid = Solid(solidData, data['Time'].to_numpy())
118             film=Fluid((fluid.temp+solid.temp)/2, fluid.time)
119             uflowRateFactor=0.01
120             flowRateHour=np.array([ufloat(t, uflowRateFactor) for t
                    in data['Flow rate'].to_numpy()])
121             node=Node(self.channel, fluid, solid, film, flowRateHour
                    , nodeList[stepIndex])
122             heat=self.calcHeat(node)
123             nodes.append(node)
124             heats.append(heat)
125             self.logger.info("{} calculated".format(node.label))
126             self.logger.debug('stepIndex = {}'.format(stepIndex))
127             stepIndex+=1
128
129         return nodes[-1], nodes, heats
130
131     def calcHeat(self,node):
132         return WallEnergyBalance(node)
133
134 class FluidEnergyBalanceFiniteDiff:
135     def __init__(self, prevNode, prevHeat, nodeLabel):
136         self.logger = logging.getLogger(__name__)
137         a=prevNode.dim['radiusInner']
```

```
138         xInlet=self.axialPos(prevNode.label)
139         xLocal=self.axialPos(nodeLabel)
140         self.logger.debug('xLocal = {}'.format(xLocal))
141 
142         self.__localFluidTemp=((2*(xLocal-xInlet))/(
143             prevNode.film.get_density(prevNode.film.temp)*prevNode.
                film.get_specificHeat(prevNode.film.temp)*prevNode.
                velocity*a
144         ))*(
145             (-1*prevHeat.h*(prevNode.fluid.temp-prevNode.solid.temp)
                )
146             - (prevNode.film.get_density(prevNode.film.temp)*
147             prevNode.film.get_specificHeat(prevNode.film.temp)*
148             a*(1/2)*
149             prevNode.fluid.get_tempDerivative(prevNode.fluid.time))
150         )+prevNode.fluid.temp
151 
152     def debug(self, prevNode, prevHeat, nodeLabel):
153         a=prevNode.dim['radiusInner']
154         xInlet=self.axialPos(prevNode.label)
155         xLocal=self.axialPos(nodeLabel)
156         self.logger.debug('local fluid temperature calculation
                components for {}'.format(nodeLabel))
157         A=(2*(xLocal-xInlet))
158         B=Util.nominal(prevNode.fluidDensity*prevNode.
                fluidSpecificHeat*prevNode.velocity*a) # this also
                shouldn't change
159         C=Util.nominal((-1*prevHeat.h*(prevNode.fluidTemp-prevNode.
                solidTemp))) #this won't change
160         D=Util.nominal((prevNode.fluidDensity*
161             prevNode.fluidSpecificHeat*
162             a*(1/2)*
163             prevNode.fluidTempDerivative)) #this shoudn't change
164         Z=Util.nominal(((2*(1))/(
165             prevNode.fluidDensity*prevNode.fluidSpecificHeat*
                prevNode.velocity*a
166         ))*(
167             (-1*prevHeat.h*(prevNode.fluidTemp-prevNode.solidTemp))
168             - (prevNode.fluidDensity*
169             prevNode.fluidSpecificHeat*
170             a*(1/2)*
171             prevNode.fluidTempDerivative)
172         ))
173         tt=prevNode.time
```

```
174             self.logger.debug("A = {}".format(A))
175             self.logger.debug("xLocal-xInlet = {}".format(xLocal-xInlet)
                    )
176             fig, ax = plt.subplots()
177             ax.plot(tt, B, label='B')
178             ax.plot(tt, C, label='C')
179             ax.plot(tt, D, label='D')
180             plt.legend()
181             plt.ylim([0,20000])
182             fig, ax = plt.subplots()
183             ax.plot(tt, Z, label='Z')
184             ax.plot(tt, Z*(1)*(xLocal-xInlet), label='Z*{}'.format((
                    xLocal-xInlet)))
185             plt.legend()
186             self.logger.debug("A = {}".format(A))
187             fig, ax = plt.subplots()
188             ax.plot(tt, Util.nominal(prevNode.fluidTemp), label='T-1
                    fluidTemp')
189             ax.plot(tt, Util.nominal(self.__localFluidTemp), label=
                    nodeLabel)
190             plt.legend()
191
192         @property
193         def localFluidTemp(self):
194             return self.__localFluidTemp
195
196         def axialPos(self, nodeLabel):
197             axialPosDictInchesFromOutletExposedSection={
198                 'T-1': (76+1/2),
199                 'T-2': (56+7/8),
200                 'T-3': (38+3/4),
201                 'T-4': (19+1/4),
202                 'T-5': (1/2)}
203             channelLengthInchesExposed=77
204             NonExposedOutlet=0.0138
205             NonExposedInlet=0.01675
206             axialPosDict={}
207             for key, value in axialPosDictInchesFromOutletExposedSection
                    .items():
208                 positionInchesFromInlet = channelLengthInchesExposed-
                        value
209                 #convert from inches to meters
210                 positionFromInlet=positionInchesFromInlet*0.0254
```

```
211             positionFromInletNonexposed=positionFromInlet+
                    NonExposedInlet
212             axialPosDict[key]= round(positionFromInletNonexposed,3)
213         position = axialPosDict[nodeLabel]
214         position_with_uncertainty = ufloat(position, 0.25*0.0254)
215         return position_with_uncertainty
216         # return axialPosDict[nodeLabel]
217
218 class Fluid:
219         def __init__(self,fluidTemp, time):
220             self.logger = logging.getLogger(__name__)
221             self.__temp=fluidTemp
222             self.__time=time
223
224         def get_tempDerivative(self, time):
225             # tempDerivativeAlt=Util.get_tempDerivativeFourier(self,
                    time)
226             tempDerivative = Util.get_tempDerivative(self, time)
227             return tempDerivative
228
229         @property
230         def temp(self):
231             return self.__temp
232
233         @temp.setter
234         def temp(self, new_value):
235             self.__temp = new_value
236
237
238         @property
239         def time(self):
240             return self.__time
241
242         def get_density(self, temp):
243             tempKelvin=temp+273.15
244             return (-8.91977e-1*tempKelvin + 1.3261e3)*ufloat(1,
                    0.1)
245
246         def get_specificHeat(self, temp):
247             tempKelvin=temp+273.15
248             return (2.79813*tempKelvin + 7.54676E2)*ufloat(1, 0.1)
249
250         def get_thermalCond(self, temp):
251             tempKelvin=temp+273.15
```

```
252             return (1.85606E-1 - 1.60002E-4 * tempKelvin)*ufloat(1,
                    0.1)
253
254         def get_dynamicVis(self,temp):
255             tempKelvin=temp+273.15
256             return 4.31224E-6*unumpy.exp(2021.208061/tempKelvin)*
                    ufloat(1, 0.1)
257
258         def get_prandtl(self, temp):
259             return self.get_dynamicVis(temp)*self.get_specificHeat(
                    temp)/self.get_thermalCond(temp)
260
261 class Solid:
262         def __init__(self,solidTemp,time):
263             self.logger = logging.getLogger(__name__)
264             self.__time=time
265             self.__temp=solidTemp
266
267         def get_tempDerivative(self, time):
268             # tempDerivativeAlt=Util.get_tempDerivativeFourier(self,
                     time)
269             tempDerivative = Util.get_tempDerivative(self, time)
270             return tempDerivative
271         @property
272         def time(self):
273             return self.__time
274
275         @property
276         def temp(self):
277             return self.__temp
278
279         @temp.setter
280         def temp(self, new_value):
281             self.__temp = new_value
282
283         def get_density(self, temp=None):
284             # Density @300 K, kg/m^3, Incropera Appendix A
285             return 8933
286
287         def get_specificHeat(self, temp=None):
288             # Specific heat capacity @300 K,J/kg K, Incropera
                    Appendix A
289             return 385
290
```

```
291 class Node:
292     '''
293     defines model state at specified axial locations
294     composed of a fluid and solid component
295     instantiates nodes and nodal data from Experiment.getMeasurments
296     inherits Channel class
297     '''
298     def __init__(self,channel, fluid, solid, film, flowRateHour,
            label):
299         self.label=label
300         self.fluid=fluid
301         self.film=film
302         self.solid=solid
303         if (solid.time == fluid.time).all:
304             self.time=solid.time
305         else:
306             raise Exception("Mismatch between solid and fluid time
                    arrays within node")
307         self.dim=channel.dim
308 
309         kgh_to_kgs=1/3600
310         self.__prandtl=self.film.get_prandtl(self.film.temp)
311         self.__flowRate=flowRateHour*kgh_to_kgs
312         self.__reynolds = (self.film.get_density(self.film.temp)*
               self.velocity*(self.dim['radiusInner']*2))/self.film.
               get_dynamicVis(self.film.temp)
313 
314     @property
315     def velocity(self):
316         flowArea=np.pi*self.dim['radiusInner']**2
317         self.__velocity = self.flowRate/(self.film.get_density(self.
               film.temp)*flowArea)
318         return self.__velocity
319 
320     @property
321     def flowRate(self):
322         return self.__flowRate
323 
324     @property
325     def reynolds(self):
326         return self.__reynolds
327 
328     @reynolds.setter
329     def reynolds(self, new_value):
```

```
330         self.__reynolds = new_value
331
332     @property
333     def prandtl(self):
334         return self.__prandtl
335
336     @prandtl.setter
337     def prandtl(self, new_value):
338         self.__prandtl = new_value
339
340 class WallEnergyBalance:
341     def __init__(self, node):
342         self.__label=node.label
343
344         a=node.dim['radiusInner']
345         l=node.dim['thickness']
346
347         #specific surface area
348         a_v=(2*a)/(l**2+(2*a*l))
349         self.a_v=a_v
350         self.__h = (node.solid.get_density(node.solid.temp)*node.
               solid.get_specificHeat(node.solid.temp))*node.solid.
               get_tempDerivative(node.solid.time)/a_v/(node.fluid.temp-
               node.solid.temp)
351         self.__Nu = self.__h * node.dim['radiusInner'] * 2 / node.
               film.get_thermalCond(node.film.temp)
352
353
354     @property
355     def h(self):
356         return self.__h
357
358     @h.setter
359     def h(self, new_value):
360         self.__h = new_value
361
362     @property
363     def Nu(self):
364         return self.__Nu
365
366     @Nu.setter
367     def Nu(self, new_value):
368         self.__Nu = new_value
369
```

```
370     @property
371     def label(self):
372         return self.__label
373 
374 class Channel:
375     '''
376     define model behaviour that is consistent throughout the channel
            component
377     copper and fluid materials will be inherited by node objects
           from the component object
378     channel geometry
379     ambient conditions if recorded
380     inherits Experiment class
381     '''
382     def __init__(self,radiusInner,thickness):
383         self.__dim={
384             'radiusInner':radiusInner,
385             'thickness': thickness
386             }
387 
388     @property
389     def dim(self):
390         return self.__dim
391 
392 import numpy as np
393 import matplotlib.pyplot as plt
394 import itertools
395 
396 class Util:
397     def __init__(self, filename, runLabel, frequencyHz,
           tempInletMean, tempInletAmplitude, meanMassFlow):
398         self.foldername=Util.get_prefix(filename)
399         self.runLabel = runLabel
400         self.tempInletMean = tempInletMean
401         self.tempInletAmplitude = tempInletAmplitude
402         self.meanMassFlow = meanMassFlow
403         self.frequencyHz = frequencyHz
404         self.logger = logging.getLogger(__name__)
405         self.logger.info("Starting run {}".format(filename))
406     class Plotter:
407         def __init__(self, parent_util, data):
408             self.logger = logging.getLogger(__name__)
409             self.parent_util = parent_util
410             self.data = data
```

```
411             self.foldername = parent_util.foldername
412             self.runLabel = parent_util.runLabel
413             self.tempInletMean = parent_util.tempInletMean
414             self.tempInletAmplitude = parent_util.tempInletAmplitude
415             self.meanMassFlow = parent_util.meanMassFlow
416             self.frequencyHz = parent_util.frequencyHz
417
418         def unique_color_linestyle(self):
419             combinations = [
420                 ('b', '-'),
421                 ('g', '--'),
422                 ('r', '-.'),
423                 ('c', ':'),
424                 ('m', (0, (3, 5, 1, 5))),
425                 ('y', (0, (5, 5))),
426                 ('k', (0, (1, 1))),
427                 ('orange', (0, (3, 10, 1, 10))),
428                 ('purple', (0, (3, 5, 1, 5, 1, 5))),
429                 ('brown', (0, (3, 1, 1, 1))),
430             ]
431             return itertools.cycle(combinations)
432
433         def warm_color_linestyle(self):
434             combinations = [
435                 ('darkred', '-'),
436                 ('orangered', '--'),
437                 ('goldenrod', '-.'),
438                 ('darkorange', ':'),
439                 ('tomato', (0, (3, 5, 1, 5))),
440             ]
441             return itertools.cycle(combinations)
442
443         def cold_color_linestyle(self):
444             combinations = [
445                 ('mediumblue', '-'),
446                 ('teal', '--'),
447                 ('dodgerblue', '-.'),
448                 ('cornflowerblue', ':'),
449                 ('darkturquoise', (0, (3, 5, 1, 5))),
450             ]
451             return itertools.cycle(combinations)
452
453         def plot_with_uncertainty(self, property_name, xlabel,
                ylabel, xlim, ylim):
```

```
454             unique_labels = self.data['label'].unique()
455             color_linestyle_gen = self.unique_color_linestyle()
456             fig, ax = plt.subplots(figsize=(3.25, 4), dpi=300)
457 
458             for label in unique_labels:
459                 node_data = self.data[self.data['label'] == label]
460                 x = node_data['fluid_time']  # Accessing fluid_time
                        directly since it doesn't have subcolumns
461                 y = node_data[property_name]['nominal_value']
462                 yerr = node_data[property_name]['std_dev']
463 
464                 color, linestyle = next(color_linestyle_gen)
465                 plt.plot(x, y, label=f'{self.formatStringsShort([
                        property_name])}, {label}', color=color,
                        linestyle=linestyle)
466                 plt.fill_between(x, y - yerr, y + yerr, color=color,
                         alpha=0.2)
467 
468             ax.set_xlabel(xlabel)
469             ax.set_ylabel(ylabel)
470             ax.set_xlim(xlim)
471             ax.set_ylim(ylim)
472 
473             ax.legend(loc='upper center', bbox_to_anchor=(0.5, -
                    0.25), ncol=2, frameon=True)
474             plt.subplots_adjust(bottom=0.1)
475             # rect=[0, 0.1, 1, 1]
476             rect=None
477 
478             # plt.title(f'{property_name} vs. {xlabel}')
479             # plt.tight_layout(rect=[0, 0, 0.75, 1])  # Adjust
                    layout to fit the legend
480             # Util.saveFig('{} time series'.format(property_name), '
                    results', plt, self.foldername))
481             measurements_string = self.formatStrings([property_name
                    ])
482             plot_name='{} time series'.format(measurements_string)
483             plotLabel='{} - {} time series for {}'.format(self.
                    runLabel, measurements_string, label)
484             caption="{} measurements for experimental run with a
                    mean cycle bulk temperature at the inlet of {}$^{{\
                    circ}}$C, with amplitude equal to {}$^{{\circ}}$C,
                    and mass flow rate of {} kg/h. The shaded area
                    visualizes the estimated propagated uncertainty. (run
```

```
                timestamp label: {})".format(measurements_string.
                capitalize(), self.tempInletMean, self.
                tempInletAmplitude, self.meanMassFlow, self.runLabel)
485             Util.saveFig(plot_name, 'results', plt, self.foldername,
                    rect)
486             Util.append_latex_code(plot_name, caption, plotLabel,
                    self.foldername, width=0.45, tex_file='plots.tex')
487
488             self.logger.debug('executed save fig')
489             # plt.show()
490             plt.close()
491
492         def formatStrings(self, property_names):
493             # Mapping dictionary
494             name_mapping = {
495                 'fluid_temp': 'bulk temperature',
496                 'h': 'Heat transfer coefficient',
497                 'solid_temp': 'wall temperature'
498             }
499
500             # Replace property names if they exist in the mapping
                    dictionary and check if the list has more than one
                    item
501             if len(property_names) > 1:
502                 formatted_string = "{} and {}".format(name_mapping.
                        get(property_names[0], property_names[0]),
                        name_mapping.get(property_names[1],
                        property_names[1]))
503             else:
504                 formatted_string = name_mapping.get(property_names
                        [0], property_names[0])
505             return formatted_string
506
507         def formatStringsShort(self, property_names):
508             # Mapping dictionary
509             name_mapping = {
510                 'fluid_temp': 'bulk',
511                 'solid_temp': 'wall',
512                 'prandtl': 'Pr',
513                 'reynolds': 'Re',
514                 'nusselt': 'Nu'
515             }
516
```

```
517             # Replace property names if they exist in the mapping
                    dictionary and check if the list has more than one
                    item
518             if len(property_names) > 1:
519                 formatted_string = "{} and {}".format(name_mapping.
                        get(property_names[0], property_names[0]),
                        name_mapping.get(property_names[1],
                        property_names[1]))
520             else:
521                 formatted_string = name_mapping.get(property_names
                        [0], property_names[0])
522             return formatted_string
523
524
525         def plot_multiple(self, property_names, xlabel, ylabel, xlim
                , ylim):
526             fig, ax = plt.subplots(figsize=(3.25, 4.5),dpi=300)
527
528             linestyle_gens = [self.cold_color_linestyle(), self.
                    warm_color_linestyle()]
529
530             for p, property_name in enumerate(property_names):
531                 unique_labels = self.data['label'].unique()
532
533                 for label in unique_labels:
534                     node_data = self.data[self.data['label'] ==
                            label]
535                     x = node_data['fluid_time']  # Accessing
                            fluid_time directly since it doesn't have
                            subcolumns
536                     y = node_data[property_name]['nominal_value']
537                     yerr = node_data[property_name]['std_dev']
538
539                     color, linestyle = next(linestyle_gens[p])
540                     ax.plot(x, y, label=f'{self.formatStringsShort([
                            property_name])}, {label}', color=color,
                            linestyle=linestyle)
541                     ax.fill_between(x, y - yerr, y + yerr, color=
                            color, alpha=0.2)
542
543             ax.set_xlabel(xlabel)
544             ax.set_ylabel(ylabel)
545             ax.set_xlim(xlim)
546             ax.set_ylim(ylim)
```

```
547
548             ax.legend(loc='upper center', bbox_to_anchor=(0.5, -
                    0.20), ncol=2, frameon=True)
549             plt.subplots_adjust(bottom=0.2)
550
551             measurements_string = self.formatStrings(property_names)
552             plot_name='{} time series'.format(measurements_string)
553             plotLabel='{} - {} time series for {}'.format(self.
                    runLabel, measurements_string, label)
554             caption="{} measurements for experimental run with a
                    mean cycle bulk temperature at the inlet of {}$^{{\
                    circ}}$C, with amplitude equal to {}$^{{\circ}}$C,
                    and mass flow rate of {} kg/h. The shaded area
                    visualizes the estimated propagated uncertainty. (run
                     timestamp label: {})".format(measurements_string.
                    capitalize(), self.tempInletMean, self.
                    tempInletAmplitude, self.meanMassFlow, self.runLabel)
555             Util.saveFig(plot_name, 'results', plt, self.foldername)
556             Util.append_latex_code(plot_name, caption, plotLabel,
                    self.foldername, width=0.45, tex_file='plots.tex')
557
558             #plt.show()
559             plt.close()
560
561     def nominal(objectAttribute):
562         return np.array([t.nominal_value for t in objectAttribute])
563
564     def uncertainty(objectAttribute):
565         return np.array([t.std_dev for t in objectAttribute])
566
567     def rmse(predictions, true_values):
568         squared_errors = (predictions - true_values) ** 2
569         mean_squared_error = np.mean(squared_errors)
570         root_mean_squared_error = np.sqrt(mean_squared_error)
571         return root_mean_squared_error
572
573     def get_nested_attr(obj, attr):
574         for level in attr.split('.'):
575             obj = getattr(obj, level)
576         return obj
577
578     def get_tempDerivative(self, time):
579         self.logger.debug('using SG derivatives')
580         method='SG'
```

```
581         sg = derivative.SavitzkyGolay(left=2, right=2, order=3,
               periodic=False)
582         self.logger.debug('NEGLECTING UNCERTAINTY FROM TEMP TIME
               DERIVATIVE')
583         temp_nominal = np.array([t.nominal_value for t in self.temp
               ])
584         temp_uncertainty = np.array([t.std_dev for t in self.temp])
585         temp_nominal_detrended=signal.detrend(temp_nominal)
586         tempDerivative_nominal = sg.d(temp_nominal_detrended, time)
587         tempDerivative =  np.array([ufloat(t, temp_uncertainty[0])
               for t in tempDerivative_nominal])
588         # if self.__class__.__name__=='Solid':
589         #     ri=600
590         #     re=800
591         #     plt.plot(time[ri:re], tempDerivative_nominal[ri:re],
               label='{}, {}, {}'.format(round(np.max(temp_nominal)),
               method, np.max(tempDerivative_nominal[ri:re])))
592         #     plt.legend(loc='upper left', bbox_to_anchor=(1.05, 1))
593         return tempDerivative
594     def get_tempDerivativeFourier(self, time):
595         method='Fourier'
596         self.logger.info('using fourier domain derivatives')
597         yes_filter = derivative.Spectral(filter=np.vectorize(lambda
               f: 1 if abs(f) < 0.11 else 0))
598         self.logger.info('NEGLECTING UNCERTAINTY FROM TEMP TIME
               DERIVATIVE')
599         temp_nominal = np.array([t.nominal_value for t in self.temp
               ])
600         temp_uncertainty = np.array([t.std_dev for t in self.temp])
601         temp_nominal_detrended=signal.detrend(temp_nominal)
602         tempDerivative_nominal = yes_filter.d(temp_nominal_detrended
               , time)
603         tempDerivative =  np.array([ufloat(t, temp_uncertainty[0])
               for t in tempDerivative_nominal])
604         return tempDerivative
605
606     def create_multilevel_dataframe(Nodes, Heats):
607         data = []
608         for node, heat in zip(Nodes, Heats):
609             fluid_temp = node.fluid.temp
610             film_temp = node.film.temp
611             solid_temp = node.solid.temp
612             nodeLength=len(node.time)
613
```

```
614                 row_data = {
615                     ('label', ''): np.repeat(node.label,nodeLength),
616                     ('dim', ''): np.repeat(node.dim,nodeLength),
617                     ('velocity', 'nominal_value'): unumpy.nominal_values
                           (node.velocity),
618                     ('velocity', 'std_dev'): unumpy.std_devs(node.
                           velocity),
619                     ('flowRate', 'nominal_value'): unumpy.nominal_values
                           (node.flowRate),
620                     ('flowRate', 'std_dev'): unumpy.std_devs(node.
                           flowRate),
621                     ('reynolds', 'nominal_value'): unumpy.nominal_values
                           (node.reynolds),
622                     ('reynolds', 'std_dev'): unumpy.std_devs(node.
                           reynolds),
623                     ('prandtl', 'nominal_value'): unumpy.nominal_values(
                           node.prandtl),
624                     ('prandtl', 'std_dev'): unumpy.std_devs(node.prandtl
                           ),
625                     ('nusselt', 'nominal_value'): unumpy.nominal_values(
                           heat.Nu),
626                     ('nusselt', 'std_dev'): unumpy.std_devs(heat.Nu),
627                     ('h', 'nominal_value'): unumpy.nominal_values(heat.h
                           ),
628                     ('h', 'std_dev'): unumpy.std_devs(heat.h),
629                     ('fluid_temp', 'nominal_value'): unumpy.
                           nominal_values(fluid_temp),
630                     ('fluid_temp', 'std_dev'): unumpy.std_devs(
                           fluid_temp),
631                     ('fluid_time', ''): node.fluid.time,
632                     ('fluid_density', 'nominal_value'): unumpy.
                           nominal_values(node.fluid.get_density(fluid_temp)
                           ),
633                     ('fluid_density', 'std_dev'): unumpy.std_devs(node.
                           fluid.get_density(fluid_temp)),
634                     ('fluid_specificHeat', 'nominal_value'): unumpy.
                           nominal_values(node.fluid.get_specificHeat(
                           fluid_temp)),
635                     ('fluid_specificHeat', 'std_dev'): unumpy.std_devs(
                           node.fluid.get_specificHeat(fluid_temp)),
636                     ('fluid_thermalCond', 'nominal_value'): unumpy.
                           nominal_values(node.fluid.get_thermalCond(
                           fluid_temp)),
```

```
637                 ('fluid_thermalCond', 'std_dev'): unumpy.std_devs(
                        node.fluid.get_thermalCond(fluid_temp)),
638                 ('fluid_dynamicVis', 'nominal_value'): unumpy.
                        nominal_values(node.fluid.get_dynamicVis(
                        fluid_temp)),
639                 ('fluid_dynamicVis', 'std_dev'): unumpy.std_devs(
                        node.fluid.get_dynamicVis(fluid_temp)),
640                 ('film_temp', 'nominal_value'): unumpy.
                        nominal_values(film_temp),
641                 ('film_temp', 'std_dev'): unumpy.std_devs(film_temp)
                        ,
642                 ('film_time', ''): node.film.time,
643                 ('film_density', 'nominal_value'): unumpy.
                        nominal_values(node.film.get_density(film_temp)),
644                 ('film_density', 'std_dev'): unumpy.std_devs(node.
                        film.get_density(film_temp)),
645                 ('film_specificHeat', 'nominal_value'): unumpy.
                        nominal_values(node.film.get_specificHeat(
                        film_temp)),
646                 ('film_specificHeat', 'std_dev'): unumpy.std_devs(
                        node.film.get_specificHeat(film_temp)),
647                 ('film_thermalCond', 'nominal_value'): unumpy.
                        nominal_values(node.film.get_thermalCond(
                        film_temp)),
648                 ('film_thermalCond', 'std_dev'): unumpy.std_devs(
                        node.film.get_thermalCond(film_temp)),
649                 ('film_dynamicVis', 'nominal_value'): unumpy.
                        nominal_values(node.film.get_dynamicVis(film_temp
                        )),
650                 ('film_dynamicVis', 'std_dev'): unumpy.std_devs(node
                        .film.get_dynamicVis(film_temp)),
651                 ('solid_temp', 'nominal_value'): unumpy.
                        nominal_values(solid_temp),
652                 ('solid_temp', 'std_dev'): unumpy.std_devs(
                        solid_temp),
653                 ('solid_time', ''): node.solid.time,
654                 ('solid_density', 'nominal_value'): unumpy.
                        nominal_values(node.solid.get_density(solid_temp)
                        ),
655                 ('solid_density', 'std_dev'): unumpy.std_devs(node.
                        solid.get_density(solid_temp)),
656                 ('solid_specificHeat', 'nominal_value'): unumpy.
                        nominal_values(node.solid.get_specificHeat(
                        solid_temp)),
```

```
657                 ('solid_specificHeat', 'std_dev'): unumpy.std_devs(
                        node.solid.get_specificHeat(solid_temp))
658             }
659
660             row_df = pd.DataFrame(row_data)
661             data.append(row_df)
662
663         df = pd.concat(data, ignore_index=True)
664         df.columns = pd.MultiIndex.from_tuples(df.columns)
665         return df
666
667     def drop_uncertain(df: pd.DataFrame, percentage: float) -> pd.
        DataFrame:
668         """
669         Drop rows in which the nusselt value standard deviation is
                above a specified percentage of the nominal value.
670
671         :param df: The input multilevel dataframe
672         :param percentage: The specified percentage (e.g., 0.1 for
                10%)
673         :return: The filtered dataframe
674         """
675         # Assuming the nusselt nominal value and standard deviation
                are located at 'nusselt', 'nominal_value' and 'nusselt',
                'std_dev' respectively.
676         nusselt_nominal = df[('nusselt', 'nominal_value')]
677         nusselt_std = df[('nusselt', 'std_dev')]
678
679         # Calculate the threshold value for the standard deviation
680         threshold = nusselt_nominal * percentage
681
682         # Filter the rows based on the threshold
683         filtered_df = df[nusselt_std <= threshold]
684
685         return filtered_df
686
687     def filter_uncertain(df: pd.DataFrame, percentage: float) -> pd.
        DataFrame:
688         """
689         Replace rows in which the nusselt value standard deviation
                is above a specified percentage of the nominal value with
                 NaN values.
690
691         :param df: The input multilevel dataframe
```

```
692         :param percentage: The specified percentage (e.g., 0.1 for
               10%)
693         :return: The filtered dataframe
694         """
695         # Assuming the nusselt nominal value and standard deviation
               are located at 'nusselt', 'nominal_value' and 'nusselt',
               'std_dev' respectively.
696         nusselt_nominal = df[('nusselt', 'nominal_value')]
697         nusselt_std = df[('nusselt', 'std_dev')]
698 
699         # Calculate the threshold value for the standard deviation
700         threshold = nusselt_nominal * percentage
701 
702         # excluded columns
703         # excluded_cols=['fluid_time', 'solid_time', 'film_time', '
               label']
704         excluded_cols = [
705         ('fluid_time', ''),
706         ('solid_time', ''),
707         ('film_time', ''),
708         ('label', ''),
709         ('solid_temp', 'nominal_value'),
710         ('solid_temp', 'std_dev'),
711         ('fluid_temp', 'nominal_value'),
712         ('fluid_temp', 'std_dev'),
713     ]
714 
715         # Create a boolean mask that filters rows where `nusselt_std
               ` > `threshold`
716         mask = nusselt_std > threshold
717 
718         # Select the columns to update using `.loc`, and exclude the
                columns in `excluded_cols`
719         # update_cols = ~df.columns.get_level_values(0).isin(
               excluded_cols)
720         update_cols = ~df.columns.isin(excluded_cols)
721         df.loc[mask, update_cols] = np.nan
722         return df
723 
724     def plot_2D_color_grid(self, dataframe, label, x_grid_length,
           y_grid_length, value_type='nominal_value', figsize=(4, 3.5)):
725         # Filter the DataFrame by the given label
726         df_label = dataframe[dataframe['label'] == label]
727 
```

```
728         # Create x and y grid edges using 'nominal_value' for the x
                and y axes
729         x_edges = np.linspace(df_label[('reynolds', 'nominal_value')
                ].min(), df_label[('reynolds', 'nominal_value')].max(),
                x_grid_length + 1)
730         y_edges = np.linspace(df_label[('prandtl', 'nominal_value')
                ].min(), df_label[('prandtl', 'nominal_value')].max(),
                y_grid_length + 1)
731
732         # Create an empty grid for the average Nusselt values
733         nusselt_grid = np.zeros((y_grid_length, x_grid_length))
734
735         fig, ax = plt.subplots(figsize=figsize, dpi=300)
736         # Iterate through the grid and compute the average Nusselt
                value for each grid cell
737         for i in range(y_grid_length):
738             for j in range(x_grid_length):
739                 cell_data = df_label[(df_label[('reynolds', '
                        nominal_value')] >= x_edges[j]) &
740                                      (df_label[('reynolds', '
                                         nominal_value')] < x_edges[j
                                         +1]) &
741                                      (df_label[('prandtl', '
                                         nominal_value')] >= y_edges[i
                                         ]) &
742                                      (df_label[('prandtl', '
                                         nominal_value')] < y_edges[i
                                         +1])]
743                 nusselt_value = cell_data[('nusselt', value_type)].
                        mean()
744                 nusselt_grid[i, j] = nusselt_value
745                 if not np.isnan(nusselt_value):  # This line has
                        been added
746                     ax.text((x_edges[j] + x_edges[j+1])/2, (y_edges[
                            i] + y_edges[i+1])/2, f'{nusselt_value:.0f}',
                             color='lightgrey', fontsize=6, ha='center',
                            va='center')  # Modified color to 'lightgrey'
747
748                 # ax.text((x_edges[j] + x_edges[j+1])/2, (y_edges[i]
                        + y_edges[i+1])/2, f'{nusselt_value:.0f}', color
                        ='white', fontsize=6, ha='center', va='center')
749
750         # Check if there are any finite values in the nusselt_grid
751         if not np.any(np.isfinite(nusselt_grid)):
```

```
752             logging.warning('No finite Nusselt values found in the
                    grid for label: %s', label)
753             return
754
755         # Choose the color palette based on the value_type
756         colormap = 'inferno' if value_type == 'nominal_value' else '
                cividis'
757
758         # Plot the 2D color grid
759         pcm = ax.pcolormesh(x_edges, y_edges, nusselt_grid, shading=
                'auto', cmap=colormap, vmin=np.nanmin(nusselt_grid), vmax
                =np.nanmax(nusselt_grid))
760         cbar = fig.colorbar(pcm, ax=ax, label=f'Nusselt {value_type.
                capitalize()}')
761         ax.set_xlabel('Reynolds')
762         ax.set_ylabel('Prandtl')
763         ax.set_title(f'{label}')
764
765         plot_name='nusselt grid for {} {}'.format(label, value_type)
766         plotLabel='{} - nusselt grid for {} {}'.format(self.runLabel
                , label, value_type)
767         if value_type=="nominal_value":
768             caption="Nusselt measurements over the range of the
                    experimental run Reynolds and Prandtl values. Values
                    are averaged within each grid."
769         elif value_type=="std_dev":
770             caption="Uncertainty in Nusselt measurements over the
                    range of the experimental run Reynolds and Prandtl
                    values."
771         Util.saveFig(plot_name, 'results', plt, self.foldername)
772         Util.append_latex_code(plot_name, caption, plotLabel, self.
                foldername, width=0.35, tex_file='plots.tex')
773         plt.close()
774
775
776
777     def plot_2D_color_grid_depr(self, dataframe, label,
            x_grid_length, y_grid_length, value_type='nominal_value',
            figsize=(4, 3.5)):
778         # Filter the DataFrame by the given label
779         df_label = dataframe[dataframe['label'] == label]
780
781         # Create x and y grid edges using 'nominal_value' for the x
                and y axes
```

```
782         x_edges = np.linspace(df_label[('reynolds', 'nominal_value')
               ].min(), df_label[('reynolds', 'nominal_value')].max(),
               x_grid_length + 1)
783         y_edges = np.linspace(df_label[('prandtl', 'nominal_value')
               ].min(), df_label[('prandtl', 'nominal_value')].max(),
               y_grid_length + 1)
784
785         # Create an empty grid for the average Nusselt values
786         nusselt_grid = np.zeros((y_grid_length, x_grid_length))
787
788         # Iterate through the grid and compute the average Nusselt
               value for each grid cell
789         for i in range(y_grid_length):
790             for j in range(x_grid_length):
791                 cell_data = df_label[(df_label[('reynolds', '
                       nominal_value')] >= x_edges[j]) &
792                                      (df_label[('reynolds', '
                                         nominal_value')] < x_edges[j
                                         +1]) &
793                                      (df_label[('prandtl', '
                                         nominal_value')] >= y_edges[i
                                         ]) &
794                                      (df_label[('prandtl', '
                                         nominal_value')] < y_edges[i
                                         +1])]
795                 nusselt_grid[i, j] = cell_data[('nusselt',
                       value_type)].mean()
796
797         # Check if there are any finite values in the nusselt_grid
798         if not np.any(np.isfinite(nusselt_grid)):
799             logging.warning('No finite Nusselt values found in the
                    grid for label: %s', label)
800             return
801
802         # Choose the color palette based on the value_type
803         colormap = 'inferno' if value_type == 'nominal_value' else '
               cividis'
804
805         # Plot the 2D color grid
806
807         fig, ax = plt.subplots(figsize=figsize, dpi=300)
808         pcm = ax.pcolormesh(x_edges, y_edges, nusselt_grid, shading=
               'auto', cmap=colormap, vmin=np.nanmin(nusselt_grid), vmax
               =np.nanmax(nusselt_grid))
```

```
809         cbar = fig.colorbar(pcm, ax=ax, label=f'Nusselt {value_type.
                capitalize()}')
810         ax.set_xlabel('Reynolds')
811         ax.set_ylabel('Prandtl')
812         ax.set_title(f'{label}')
813 
814         plot_name='nusselt grid for {} {}'.format(label, value_type)
815         plotLabel='{} - nusselt grid for {} {}'.format(self.runLabel
                , label, value_type)
816         if value_type=="nominal_value":
817             caption="Nusselt measurements over the range of the
                    experimental run Reynolds and Prandtl values. Values
                    are averaged within each grid."
818         elif value_type=="std_dev":
819             caption="Uncertainty in Nusselt measurements over the
                    range of the experimental run Reynolds and Prandtl
                    values."
820         Util.saveFig(plot_name, 'results', plt, self.foldername)
821         Util.append_latex_code(plot_name, caption, plotLabel, self.
                foldername, width=0.35, tex_file='plots.tex')
822         plt.close()
823 
824 
825     def plot_nominal_vs_std_dev_single_node(self, dataframe, label,
            variable_name):
826         # Filter the DataFrame by the given label
827         df_label = dataframe[dataframe['label'] == label]
828 
829         # Extract the nominal values and standard deviations for the
                given variable
830         nominal_values = df_label[(variable_name, 'nominal_value')]
831         std_dev_values = df_label[(variable_name, 'std_dev')]
832 
833         # Plot the nominal values against the standard deviations
834         fig, ax = plt.subplots(figsize=(3.25, 3.25), dpi=300)
835         xlabel=f'{variable_name.capitalize()} Nominal Value'
836         ylabel=f'{variable_name.capitalize()} Std Dev'
837         title=f'{label}'
838         ax.set_xlabel(xlabel)
839         ax.set_ylabel(ylabel)
840         ax.set_title(title)
841 
842         ax.scatter(nominal_values, std_dev_values, s=1, color='k')
```

```
843         plot_name='nom vs std dev {} for {}'.format(variable_name,
                label)
844         plotLabel='{} - nom vs std dev {} for {}'.format(self.
                runLabel, variable_name, label)
845         caption='{} uncertainty distribution for range of nominal
                values in node location {}'.format(variable_name.
                capitalize(), label)
846         Util.saveFig(plot_name, 'results', plt, self.foldername)
847         Util.append_latex_code(plot_name, caption, plotLabel, self.
                foldername, width=0.45, tex_file='plots.tex')
848
849         #plt.show()
850         plt.close()
851
852     def plot_nominal_vs_std_dev(self, dataframe, labels,
            variable_name):
853         markers = ['o', 's', '^', 'v', 'p', '*', 'X', 'D', 'h', '8']
854         colors = plt.cm.tab10(np.linspace(0, 1, len(labels)))
855
856         fig, ax = plt.subplots(figsize=(3.25, 3.5), dpi=300)
857         for idx, label in enumerate(labels):
858             # Filter the DataFrame by the given label
859             df_label = dataframe[dataframe['label'] == label]
860
861             # Extract the nominal values and standard deviations for
                    the given variable
862             nominal_values = df_label[(variable_name, 'nominal_value
                    ')]
863             std_dev_values = df_label[(variable_name, 'std_dev')]
864
865             # Plot the nominal values against the standard
                    deviations
866             ax.scatter(nominal_values, std_dev_values, s=1, color=
                    colors[idx], marker=markers[idx % len(markers)],
                    label=label)
867             xlabel=f'{variable_name.capitalize()} Nominal Value'
868             ylabel=f'{variable_name.capitalize()} Std Dev'
869             title=f'{label}'
870             ax.set_xlabel(xlabel)
871             ax.set_ylabel(ylabel)
872
873
874         # ax.legend()
```

```
875         ax.legend(loc='upper center', bbox_to_anchor=(0.5, -0.3),
               ncol=2, frameon=True)
876         plt.subplots_adjust(bottom=0.3)
877         # ax.legend(fontsize='x-small', ncol=2, loc='lower center',
               bbox_to_anchor=(0.5, -0.6))
878         plot_name = 'nom vs std dev {}'.format(variable_name)
879         plotLabel = '{} - nom vs std dev {}'.format(self.runLabel,
               variable_name)
880         caption = '{} uncertainty distribution for range of nominal
               values'.format(variable_name.capitalize())
881         Util.saveFig(plot_name, 'results', plt, self.foldername)
882         Util.append_latex_code(plot_name, caption, plotLabel, self.
               foldername, width=0.45, tex_file='plots.tex')
883
884         #plt.show()
885         plt.close()
886
887
888     def plot_variables_with_error_bars(self, dataframe, label, x_var
           , y_var, third_var, third_var_range):
889         # Filter the DataFrame by the given label
890         df_label = dataframe[dataframe['label'] == label]
891
892         # Filter the DataFrame by the specified range for the third
               variable
893         df_filtered = df_label[(df_label[(third_var, 'nominal_value'
               )] >= third_var_range[0]) &
894                                (df_label[(third_var, 'nominal_value')]
                                   <= third_var_range[1])]
895
896         # Extract the nominal values and standard deviations for the
                x and y variables
897         x_nominal_values = df_filtered[(x_var, 'nominal_value')]
898         x_std_dev_values = df_filtered[(x_var, 'std_dev')]
899         y_nominal_values = df_filtered[(y_var, 'nominal_value')]
900         y_std_dev_values = df_filtered[(y_var, 'std_dev')]
901
902         # Plot the nominal values with error bars
903         plt.figure(figsize=(4, 4))
904         plt.errorbar(x_nominal_values, y_nominal_values, xerr=
               x_std_dev_values, yerr=y_std_dev_values, fmt='ok',
               markersize=1, alpha=0.5, ecolor='grey')
905         plt.xlabel(f'{x_var.capitalize()} Nominal Value')
906         plt.ylabel(f'{y_var.capitalize()} Nominal Value')
```

```
907         # plt.title(f'{x_var.capitalize()} vs {y_var.capitalize()}
                with Error Bars for {label}')
908         plt.title(f'{label}')
909         Util.saveFig('{} vs {} with error'.format(x_var, y_var), '
                results', plt, self.foldername)
910         #plt.show()
911         plt.close()
912
913     def saveFig(figName, subfolderName, fig, subsubfolderName=None,
            legendBox=None):
914         fname = "{}.png".format(figName)
915         current_directory_path = os.getcwd()
916         subfolder_path = os.path.join(current_directory_path,
                subfolderName)
917
918         if subsubfolderName:
919             subfolder_path = os.path.join(subfolder_path,
                    subsubfolderName)
920
921         if not os.path.exists(subfolder_path):
922             os.makedirs(subfolder_path)
923
924         path = os.path.join(subfolder_path, fname)
925         if legendBox != None:
926             fig.tight_layout(rect=legendBox)
927         else:
928             fig.tight_layout()
929         # fig.show()
930         fig.savefig(path, bbox_inches='tight')
931
932     def append_latex_code(plot_name, caption, label, subfolder,
            width=0.45, tex_file='plots.tex'):
933         latex_code = f"""\t\\begin{{subfigure}}[b]{{{width}\\
                textwidth}}
934         \\centering
935         \t\\includegraphics[width=\\textwidth]{{chap-shefra-exp/
                results/{subfolder}/{plot_name}.png}}
936         \t\\caption{{{caption}}}
937         \t\\label{{fig:{label}}}
938         \\end{{subfigure}}
939         \\hfill
940     """
941
942         with open(tex_file, 'a') as file:
```

```
943             file.write(latex_code)
944
945     def get_dataPath(subfolderName, fname):
946         # obtains a csv file given a subfolder and file name
947         current_directory_path = os.getcwd()
948         subfolder_path = os.path.join(current_directory_path,
                subfolderName)
949         path = os.path.join(subfolder_path, fname)
950         return path
951     def get_prefix(filename):
952         return filename.split('.')[0]
953
954     def extract_filenames(folder_name):
955         # Get the current directory where the script is located
956         current_directory = os.path.dirname(os.path.abspath(__file__
                ))
957
958         # Build the full path of the target folder
959         folder_path = os.path.join(current_directory, folder_name)
960
961         # Check if the folder exists
962         if os.path.exists(folder_path) and os.path.isdir(folder_path
                ):
963             # Get the list of filenames in the folder
964             all_filenames = os.listdir(folder_path)
965             # Filter out filenames starting with a period
966             filenames = [filename for filename in all_filenames if
                    not filename.startswith('.')]
967             return filenames
968         else:
969             print(f"Folder '{folder_name}' not found in the current
                    directory.")
970             return []
971
972     #     return temp_inlet_mean, temp_inlet_amplitude,
            freq_inlet_hz, mass_flow_kgh
973     def extract_variables_from_filename(filename):
974         temp_inlet_mean_regex = r"mean_([\d]+-[\d]+)"
975         temp_inlet_amplitude_regex = r"amp_([\d]+(?:-[\d]+)?)"
976         freq_inlet_hz_regex = r"f_([\d]+(?:-[\d]+)?)"
977         mass_flow_kgh_regex = r"flow_([\d]+(?:-[\d]+)?)"
978
979         temp_inlet_mean = float(re.search(temp_inlet_mean_regex,
                filename).group(1).replace('-', '.'))
```

```
980         temp_inlet_amplitude = float(re.search(
               temp_inlet_amplitude_regex, filename).group(1).replace('-
               ', '.'))
981         freq_inlet_hz = float(re.search(freq_inlet_hz_regex,
               filename).group(1).replace('-', '.'))
982         mass_flow_kgh = float(re.search(mass_flow_kgh_regex,
               filename).group(1).replace('-', '.'))
983
984         return temp_inlet_mean, temp_inlet_amplitude, freq_inlet_hz,
                mass_flow_kgh
985
986     def dataInfo(fname):
987         # return experiment time in seconds for clean splices of
               periodically-steady state data and the frequency of inlet
                temperature
988         cases = {
989             'mean_51-1_amp_11_flow_75_f_0-1_range_783012-to-1218713
                   _datetime_11_19_21_03_44_04_PM.csv': ([783, 1218],
                   0.1, '0344P-111921'),
990             'mean_51-1_amp_22_flow_75_f_0-1_range_181700-to-5054
                   _datetime_11_19_21_04_05_14_PM.csv': ([181, 505],
                   0.1, '0405P-111921'),
991             'mean_77-2_amp_40-8_flow_78-7_f_0-1_range_172200-to-
                   879602_datetime_11_19_21_04_19_57_PM.csv': ([172,
                   879], 0.1, '0419P-111921'),
992             'mean_100-9_amp_38-3_flow_80_f_0-1_range_103800-to-
                   690101_datetime_11_19_21_04_35_25_PM.csv': ([103,
                   690], 0.1, '0435P-111921'),
993             'mean_50-6_amp_18-5_flow_76_f_0-1667_range_114200-to-
                   358200_datetime_11_19_21_04_52_02_PM.csv': ([114,
                   358], 0.16666666667, '0452P-111921'),
994             'mean_50-3_amp_15_flow_75-9_f_0-25_range_173800-to-
                   530701_datetime_11_19_21_04_58_32_PM.csv': ([173,
                   530], 0.25, '0458P-111921'),
995             'mean_50-5_amp_22-5_flow_75-5_f_0-0833_range_119500-to-
                   640001_datetime_11_19_21_05_08_24_PM.csv': ([119,
                   640], 0.83333333333, '0508P-111921'),
996             'mean_50-4_amp_20-7_flow_77-2_f_0-125_range_115300-to-
                   552801_datetime_11_19_21_05_19_25_PM.csv': ([115,
                   552], 0.125, '0519P-111921'),
997             'mean_50-4_amp_20-8_flow_77-2_f_0-125_range_71000-to-
                   552801_datetime_11_19_21_05_28_55_PM.csv': ([71,
                   552], 0.125, '0528P-111921'),
998         }
```

```
999
1000        if fname in cases:
1001            time_range, freq_inlet_hz, timestamp = cases[fname]
1002            temp_inlet_mean, temp_inlet_amplitude, _, mass_flow_kgh
                    = Util.extract_variables_from_filename(fname)
1003            return time_range, freq_inlet_hz, timestamp,
                    temp_inlet_mean, temp_inlet_amplitude, mass_flow_kgh
```

Listing C.1: SHEFRA experiment data reduction and plotting Code written in Python

Appendix D

Parameter Estimation Code

The following code was used for parameter estimation for the space-independent model of SHEFRA described in Chapter 5.

```
'''%% initialize
clear, clc, close all
set(groot, 'DefaultAxesFontSize', 12);

Nu = 2;
Nu0=0;
p1 = 0.1; % Define p1 value
p2 = 0.85; % Define p2 value
Omega = 10;

sampleTime = 0.01;
totalTime = 40;

% Transfer function coefficients
numerator_coeffs = [Nu*p2, Nu*Omega*p1, 4*Nu*Omega^2*p2, Nu*Omega^3*
   p1];
denominator_coeffs = [1, Nu*p2, 4*Omega^2, 4*Nu*Omega^2*p2, 0];
system = tf(numerator_coeffs, denominator_coeffs);

timeVector = [0:sampleTime:totalTime]';

input_sine = sin(Omega * timeVector);
[simulatedOutputSine, timeVectorSimulatedSine] = lsim(system,
   input_sine, timeVector);

% Define initial parameter values
par0 = [Nu0, p1, p2, Omega];
fixed_p1 = par0(2);
```

```
fixed_p2 = par0(3);
fixed_Omega = par0(4);

num_runs = 1000;
Nu_est_all = zeros(num_runs, 1);

for i = 1:num_runs
    % Add noise to the simulated output
    thermocoupleUncertainty = 3; % absolute error
    lowestAmplitude = 5; % celsius
    dimensionlessUncertainty = thermocoupleUncertainty /
       lowestAmplitude;
    noise = dimensionlessUncertainty * 0.341 * (randn(size(
       simulatedOutputSine)));
    simulatedOutputSineNoise = simulatedOutputSine + noise;

    % prepare the system identification data
    systemIdentificationDataSine = iddata(simulatedOutputSineNoise,
       input_sine, sampleTime);

    % Estimate Nu using Newton-Raphson method
    Nu_est = fsolve(@(Nu) grad_obj_fun(Nu, fixed_p1, fixed_p2,
       fixed_Omega, systemIdentificationDataSine), par0(1));
    Nu_est_all(i) = Nu_est;
end

% Calculate mean and standard deviation of Nu_est
mean_Nu_est = mean(Nu_est_all);
std_Nu_est = std(Nu_est_all);

disp('Mean of Estimated Parameters:');
disp(mean_Nu_est);
disp('Standard Deviation of Estimated Parameters:');
disp(std_Nu_est);

%% Gradient of the objective function
function grad = grad_obj_fun(Nu, p1, p2, Omega, data)
    % Compute the gradient of the objective function with respect to
        Nu
    delta = 1e-6;
    err_plus = obj_fun(Nu + delta, p1, p2, Omega, data);
    err_minus = obj_fun(Nu - delta, p1, p2, Omega, data);
    grad = (err_plus - err_minus) / (2 * delta);
end
```

```
66
67 %% Objective function
68 function err = obj_fun(Nu, p1, p2, Omega, data)
69     sys = tf_with_constraints(Nu, p1, p2, Omega);
70     y_sim = lsim(sys, data.InputData, data.SamplingInstants);
71     err = mean((y_sim - data.OutputData).^2); % Calculate mean
          squared error
72 end
73
74 %% Custom model function
75 function sys = tf_with_constraints(Nu, p1, p2, Omega)
76     numerator_coeffs = [Nu*p2, Nu*Omega*p1, 4*Nu*Omega^2*p2, Nu*
          Omega^3*p1];
77     denominator_coeffs = [1, Nu*p2, 4*Omega^2, 4*Nu*Omega^2*p2, 0];
78     sys = tf(numerator_coeffs, denominator_coeffs);
79 end
```

Listing D.1: Parameter estimation for space-independent SHEFRA model written in MATLAB using the System Identification Toolbox.

Bibliography

[1] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions*. en. Google-Books-ID: TzMZRQAACAAJ. Dover, 1965. ISBN: 978-0-88307-589-0.

[2] Charalampos Andreades. *Technical Description of the'Mark1'Pebble-bed Fluoride-salt-cooled High-temperature Reactor (PB-FHR) Power Plant*. Department of Nuclear Engineering, University of California, Berkeley, 2014.

[3] Philippe M. Bardet and Per F. Peterson. "Options for Scaled Experiments for High Temperature Liquid Salt and Helium Fluid Mechanics and Convective Heat Transfer". en. In: *Nuclear Technology* 163.3 (Sept. 2008), pp. 344–357. ISSN: 0029-5450, 1943-7471. DOI: 10.13182/NT163-344. URL: https://www.tandfonline.com/doi/full/10.13182/NT163-344 (visited on 04/18/2023).

[4] Kent Beck. *Test-driven Development: By Example*. en. Addison-Wesley Professional, 2003. ISBN: 978-0-321-14653-3.

[5] Adrian Bejan. *Convection Heat Transfer*. en. John Wiley & Sons, Mar. 2013. ISBN: 978-1-118-33008-1.

[6] Theodore L. Bergman et al. *Incropera's Principle of Heat and Mass Transfer*. en. John Wiley & Sons Canada, Limited, Sept. 2019. ISBN: 978-1-119-64241-1.

[7] J. W. Cooke and B. Cox. *Forced-convection heat-transfer measurements with a molten fluoride salt mixture flowing in a smooth tube*. en. Tech. rep. ORNL-TM–4079, 4486196. Mar. 1973, ORNL–TM–4079, 4486196. DOI: 10.2172/4486196. URL: http://www.osti.gov/servlets/purl/4486196/ (visited on 04/02/2023).

[8] von L. Graetz. "Über die wärmeleitungsfähigkeit von flüssigkeiten". In: *Annalen der Physik* 261.7 (1885). Publisher: WILEY-VCH Verlag Leipzig, pp. 337–357.

[9] L Huddar et al. "Application of frequency response methods in separate and integral effects tests for molten salt cooled and fueled reactors". en. In: *Nucl. Eng. Des.* (2017). DOI: 10.1016/j.nucengdes.2017.11.045. (Visited on 04/18/2023).

[10] L Huddar et al. "Experimental Strategy for the Determination of Heat Transfer Coefficients in Pebble-Beds Cooled by Fluoride Salts". en. In: (2015), p. 17.

[11] Lakshana Ravindranath Huddar. "Heat Transfer in Pebble-Bed Nuclear Reactor Cores Cooled by Fluoride Salts". en. PhD thesis. University of California, Berkeley, 2016.

[12] S. Kakac, R. K. Shah, and W. Aung. "Handbook of single-phase convective heat transfer". English. In: (Jan. 1987). Publisher: John Wiley and Sons Inc.,New York, NY. URL: https://www.osti.gov/biblio/6291382 (visited on 04/25/2023).

[13] M. A. Leveque. "Les lois de la tranmission de chaleur par convection". In: *Ann Mines Rec Mem L'Explotas Mines* 13 (1928), p. 210.

[14] B. Metais and E. R. G. Eckert. "Forced, Mixed, and Free Convection Regimes". In: *Journal of Heat Transfer* 86.2 (May 1964), pp. 295–296. ISSN: 0022-1481. DOI: 10.1115/1.3687128. URL: https://doi.org/10.1115/1.3687128 (visited on 04/25/2023).

[15] NRC. *Standard Review Plan for the Review of SafetyAnalysis Reports for Nuclear Power Plants: LWR Edition (NUREG-0800, Formerly issued as NUREG-75/087), Chapter 15 Transient and Accident Analyses*. 2007.

[16] Wilson F. N. Santos and Jerônhimo S. Travelho. "Transient conjugated forced convection in a parallel plate duct with convection from the ambient and periodic variation of inlet temperature". en. In: *Applied Scientific Research* 51.3 (Oct. 1993), pp. 625–638. ISSN: 1573-1987. DOI: 10.1007/BF00868004. URL: https://doi.org/10.1007/BF00868004 (visited on 10/18/2022).

[17] Hermann Schlichting and Klaus Gersten. *Boundary-Layer Theory*. en. Springer Berlin Heidelberg, June 2018. ISBN: 978-3-662-57095-1.

[18] R. K. Shah and M. S. Bhatti. "Laminar convective heat transfer in ducts". In: *Handbook of single-phase convective heat transfer* 3 (1987). Publisher: Wiley New York, pp. 1–137.

[19] Manohar S. Sohal et al. *Engineering database of liquid salt thermophysical and thermochemical properties*. Tech. rep. Idaho National Lab.(INL), Idaho Falls, ID (United States), 2010.

[20] E.M. Sparrow and F.N. De Farias. "Unsteady heat transfer in ducts with time-varying inlet temperature and participating walls". en. In: *International Journal of Heat and Mass Transfer* 11.5 (May 1968), pp. 837–853. ISSN: 00179310. DOI: 10.1016/0017-9310(68)90128-2. URL: https://linkinghub.elsevier.com/retrieve/pii/0017931068901282 (visited on 10/02/2022).

[21] The Dow Chemical Company. *Dowtherm A Heat Transfer Fluid Product Technical Data*. 1997.

[22] J. S. Travelho and W. F. N. Santos. "Unsteady Conjugate Heat Transfer in a Circular Duct With Convection From the Ambient and Periodically Varying Inlet Temperature". In: *Journal of Heat Transfer* 120.2 (May 1998), pp. 506–510. ISSN: 0022-1481. DOI: 10.1115/1.2824277. URL: https://doi.org/10.1115/1.2824277 (visited on 10/19/2022).

[23] Gnielinski V. "New equations for heat and mass transfer in turbulent pipe and channel flow". In: *Int. Chem. Eng.* 16.2 (1976), pp. 359–368. URL: `https://cir.nii.ac.jp/crid/1574231875472963072` (visited on 01/05/2023).

[24] David F. Williams, Louis M. Toth, and Kevin T. Clarno. *Assessment of Candidate Molten Salt Coolants for the Advanced High Temperature Reactor (AHTR).* United States. Department of Energy, 2006.

[25] Asako Yutaka, Nakamur Hiroshi, and Mohammad Faghri. "Developing laminar flow and heat transfer in the entrance region of regular polygonal ducts". In: *International journal of heat and mass transfer* 31.12 (1988). Publisher: Elsevier, pp. 2590–2593.

[26] Nicolas Zweibaum. "Experimental Validation of Passive Safety System Models: Application to Design and Optimization of Fluoride-Salt-Cooled, High-Temperature Reactors". en. PhD thesis. UC Berkeley, 2015. URL: `https://escholarship.org/uc/item/744882bt` (visited on 05/02/2023).