HIERARCHICAL COMBINED PLANT AND CONTROL DESIGN FOR THERMAL MANAGEMENT SYSTEMS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Austin L. Nash

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

December 2019

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

Prof. Neera Jain
    School of Mechanical Engineering
Prof. James Braun
    School of Mechanical Engineering
Prof. Peter Meckl
    School of Mechanical Engineering
Prof. Bin Yao
    School of Mechanical Engineering
Prof. Stanislaw Żak
    School of Electrical and Computer Engineering

Approved by:

Dr. Nicole Key
Head of the School Graduate Program

To my wife and parents.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, Prof. Neera Jain, for her constant guidance and mentorship throughout my time as a Ph.D. student. I am forever indebted to her for providing such a supportive environment and for helping me navigate research and life during my time at Purdue. Her drive and dedication for both progressing scientific research and education and for helping students reach their potential is something I will aspire to in my own career. I look forward to continuing to learn from her over the coming years. I would also like to thank the members of my advisory committee - James Braun, Peter Meckl, Bin Yao, and Stanislaw Żak - for their valuable insights during the preliminary exam process. I also want to acknowledge Richard Buckius, Beth Hess, and Sameer Naik for their advice and assistance during my year as a Lambert Teaching Fellow. Finally, thank you to Galen Jackson, Kevin McCarthy, Patrick McCarthy, and Stephen Hodson at PCKA for their help along the way.

Thank you to all of my colleagues, past and present, in the Jain Research Lab (JRL) - Rian, Akash, Yeshaswi, Aaron, Ana, Trevor, Karan, and Jianqui (Jack) - for their collaboration and friendship over the last four years. Getting to know so many wonderful people from backgrounds very different from my own has been one of the most enjoyable parts of my experience in graduate school. I look forward to staying in touch with each of you and seeing the great things you accomplish in your careers. Additionally, I would like to acknowledge the entire Herrick community - faculty, staff, and students - for fostering such an incredible collaborative environment that I feel is second to none. Because of you all, I consider myself a Herrick graduate just as much as I consider myself a Purdue graduate.

To my family, I simply wouldn't be where I am today without you. I want to thank my parents - Terry and Janice - for exemplifying the values that have propelled me to this point. My father is the hardest working person I know and my mother is the most kind-hearted person I've ever been around. I couldn't imagine a better set of role models and I hope I make you proud as I pursue my upcoming career. To my sister Courtney, you've always been like a best friend to me and I can't thank you enough for your friendship over the years.

Lastly, but most importantly, I want to thank my wife Caitlin for her encouragement and support during my time at Purdue. While we've been together since high school, it was here that we officially started our life together. I don't deserve the level of patience you have given me the last several years, but I am forever grateful for it. I can't wait to begin the next chapter of our life together and to see what the future has in store for us.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

ABSTRACT

Nash, Austin L. Ph.D., Purdue University, December 2019. Hierarchical Combined Plant and Control Design for Thermal Management Systems. Major Professor: Neera Jain, School of Mechanical Engineering.

Over the last few decades, many factors, including increased electrification, have led to a critical need for fast and efficient transient cooling. Thermal management systems (TMSs) are typically designed using steady-state assumptions and to accommodate the most extreme operating conditions that could be encountered, such as maximum expected heat loads. Unfortunately, by designing systems in this manner, closed-loop transient performance is neglected and often constrained. If not constrained, conventional design approaches result in oversized systems that are less efficient under nominal operation. Therefore, it is imperative that *transient* component modeling and subsystem interactions be considered at the design stage to avoid costly future redesigns. Simply put, as technological advances create the need for rapid transient cooling, a new design paradigm is needed to realize next generation systems to meet these demands.

In this thesis, I develop a new design approach for TMSs called hierarchical control co-design (HCCD). More specifically, I develop a HCCD algorithm aimed at optimizing high-fidelity design and control for a TMS across a system hierarchy. This is accomplished in part by integrating system level (SL) CCD with detailed component level (CL) design optimization. The lower-fidelity SL CCD algorithm incorporates feedback control into the design of a TMS to ensure controllability and robust transient response to exogenous disturbances, and the higher-fidelity CL design optimization algorithms provide a way of designing detailed components to achieve the desired performance needed at the SL. Key specifications are passed back and forth between levels of the hierarchy at each iteration to converge on an optimal design that is responsive to desired objectives at each level. The resulting HCCD algorithm permits the design and control of a TMS that is not only optimized for steady-state efficiency, but that can be designed for robustness to transient disturbances while achieving said disturbance rejection with minimal compromise to system efficiency. Several case studies are used to demonstrate the utility of the algorithm in designing systems with different objectives. Additionally, high-fidelity thermal modeling software is used to validate a solution to the proposed model-based design process.

1. INTRODUCTION

Thermal management systems (TMSs) have historically been designed to perform at steady operating conditions. As a result, these systems are typically over-designed to accommodate the most extreme operating conditions expected. For example, components may be designed to accommodate maximum expected heat loads. In systems which operate at moderately steady loads over long periods of times, this method of design is generally acceptable. However, as technology progresses, transient heat loads are increasingly becoming the norm rather than the exception.

As detailed in [1], recent advances in areas such as microelectronics have enabled the design of microprocessors which can adapt to changes in computing demands on a time scale of microseconds. Unfortunately, slow response of thermal components has negatively impacted the thermal management capabilities of the multicore microprocessors; as a result, device performance is altered to accommodate thermal limitations [2]. By designing systems and components "open-loop" with steady-state operational load assumptions, thermal systems engineers limit *transient* thermodynamic performance. In addition, overdesigning of systems can cause significant increases in system weight or cost. While the example of thermal limitations in microprocessor operation is small-scale, similar dynamic problems occur for thermal systems on all scales.

With both increased system complexity and integration with other dynamic systems, the design of effective thermal systems now requires theoretically sound analyses that address the transient nature of dynamics from a material level all the way to system-level interactions [1]. A critical gap that remains in dynamic thermal management concerns using design optimization strategies that exploit new levers of control in designing these systems. It is imperative that transient closed-loop performance and subsystem interactions be considered at the design stage to avoid costly future redesigns. As the need for high scale cooling increases for many systems [3, 4], the importance of optimal, efficient thermal management increases. *Simply put, as technological advances create the need for rapid transient cooling, a new design paradigm is needed to realize next generation systems to meet these demands.*

1.1 Background

1.1.1 Introduction to Control Co-design

Combined plant and control design, or control co-design (CCD), provides a path toward realizing next generation thermal cooling demands by incorporating the notion of transients early in the system design stage. CCD is a process wherein a system's governing control policy, and thus its closed-loop performance, is considered at the *plant design* stage. A simple illustration of the difference between conventional (sequential) design methods and CCD methods is shown in Fig. 1.1. Typically, a plant (or system) is designed using an offline optimization that defines the open-loop performance of the system. A control systems engineer is then tasked with designing a feedback controller for the plant to yield closed-loop performance which meets a set of desired performance specifications. In CCD, however, both plant design and control design are considered together at the system design stage. CCD essentially enables a broader design space to be considered by leveraging additional degrees of freedom to meet closed-loop performance specifications. The merits of CCD are rooted in the fact that there are fundamental limitations to what can be achieved with the addition of feedback control for a given plant design [5]. Given the opportunity to consider the impact of plant design parameters (e.g., mass) on closed-loop performance, the plant could be designed differently to yield systems with improved performance characteristics such as robustness and efficiency.

CCD algorithms can be grouped into four categories: sequential, iterative, nested, and simultaneous [6]. Using sequential algorithms, the plant is optimized first, followed by an optimization of the controller. In other words, the optimizations occur

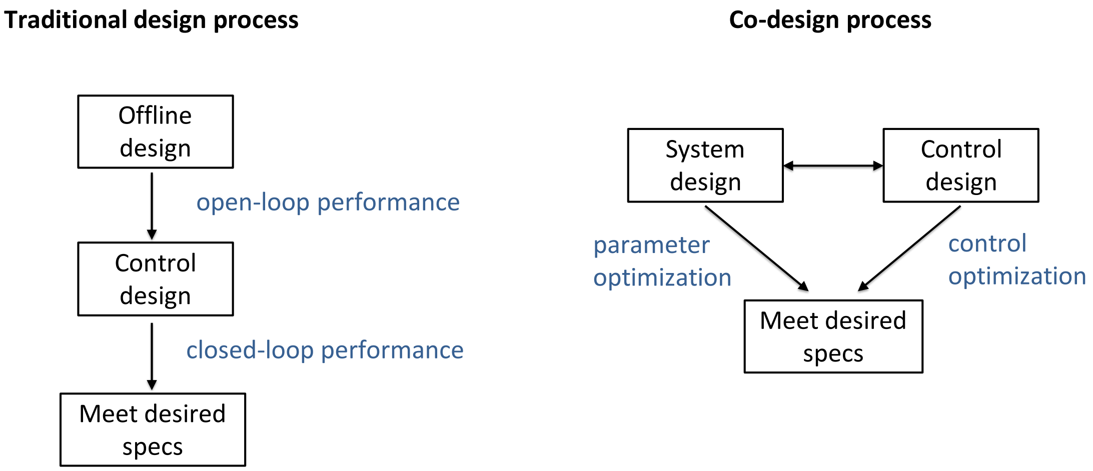


Figure 1.1. : Schematic illustrating fundamental difference between conventional design and CCD.

in series. In iterative algorithms, the sequential process is repeated multiple times until some user-defined convergence is achieved. These algorithms seldom guarantee optimality. Nested CCD algorithms are similar to iterative algorithms but can offer guaranteed optimality. Simultaneous CCD algorithms are the most robust of the four; these algorithms are multidisciplinary and guarantee optimality. However, they are often difficult to solve and are the most computationally complex of the four types.

Applications of existing simultaneous and nested CCD algorithms in the literature have largely been geared toward simple control policies with control objectives such as placing closed-loop poles in the left half plane; furthermore, the use of simplified plant models and quadratic cost functions is prevalent in the CCD literature [7–10]. Authors acknowledge that investigation is needed to understand the application of CCD techniques to more sophisticated systems and applications [10]. Within the literature, the concept of CCD has been largely confined to the aerospace and mechanical engineering communities with applications such as structural design and rotor design [7, 11–15] and has played one of its most critical roles in the design of tactical aircraft [16]. CCD has also been used to explore optimal sizing of micro-grid components [17]. A comprehensive literature review on CCD can be found in [13].

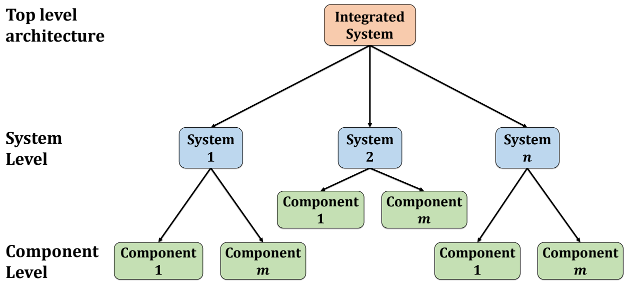
Recently, CCD has been applied to thermodynamic systems such as fuel cell and battery hybrid vehicles [18], micro-combined heat and power (micro-CHP) systems [19], and devices used in process control such as evaporators and mixing tanks [20,21]. Kim and Peng used a pseudo-SDP controller with basis functions to determine control variables that can be leveraged as design variables [18]. Key components of the plant model, however, were static and therefore did not capture some transient behavior of the system. In [19], the authors used a multiobjective optimization framework to design multi-parametric model predictive controllers. They used gray-box LTI state-space models derived with system identification techniques. In [20], the authors used parametric programming to derive an explicit control structure, thereby removing the need for solving an optimization problem online for model predictive control. While these efforts demonstrated initial work in adapting CCD techniques to thermal systems, *modeling simplifications were made that either excluded some dynamics altogether or transformed them such that the model was no longer based on first principles.* This poses a challenge for thermal-fluid systems in which static and dynamic system efficiency, along with other dynamic characteristics such as time delays, are best described by the first and second laws of thermodynamics. In particular, it is well known that the second law can be used to quantify the best theoretical performance attainable for a range of thermodynamic systems. This affects applications ranging from thermal management to thermal energy storage which are facilitated by various energy transport processes. These processes are susceptible to dissipative losses, or irreversibilities, which affect the efficiency of a given system and are quantified in a term defined as *entropy generation*.

Although second law analysis has been used to guide efficient design of both physical systems and control algorithms *separately*, it has not been considered for use in control co-design algorithms. Engineers have long considered the second law of thermodynamics for *steady-state* design decisions [22, 23]. More recently, researchers have leveraged exergy destruction minimization, which is proportional to minimizing entropy generation, in designing controllers for a class of thermal systems such as vapor-compression systems [24–28] as well as other thermal-fluid systems [29, 30]. Second law methods have also been used in passivity-based control and chemical process stabilization [31–36], and for controlling internal combustion engines (ICEs) [37–40].

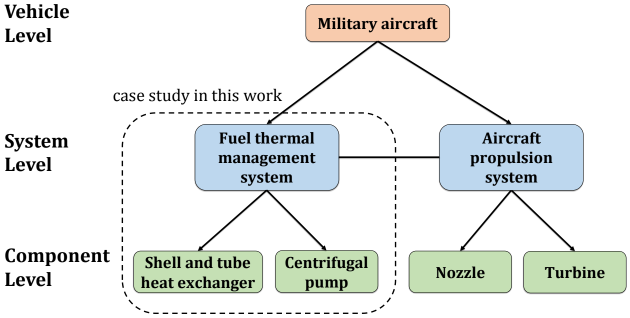
Across various sectors, from fossil-fueled power generation to dynamic thermal management, solving the problem of designing *and* controlling a system to minimize these irreversibilities is becoming increasingly critical [1], particularly because *improved efficiency typically translates to critical performance metrics such as increased thermal endurance in the case of aircraft thermal management.* However, as previously described, *existing research on CCD of thermal systems does not explicitly account for the effect that system design has on efficiency.* **A critical gap concerns developing CCD algorithms that explicitly account for key performance measures and dynamics common to TMS operation, namely static and dynamic system efficiency.**

1.1.2 Hierarchical System Design

A general requirement for model-based control design, and thus CCD algorithms, is that a reduced-order model of the system dynamics must be used. However, this precludes engineers, to an extent, from considering detailed design features in a CCD algorithm. This becomes problematic for thermal management systems (TMSs) which are themselves comprised of several dynamic components which each contribute to the overall dynamics exhibited by the TMS. For example, the component dynamics of a heat exchanger greatly affect the heat dissipation that can be achieved by a TMS. Moreover, the system level dynamics of a TMS affect the capabilities and functionality of other systems on board the aircraft such as the aircraft electronics. However, CCD models are not detailed enough to account for component design features such as the heat exchanger's tube bundle geometry. Therefore, the reduced-order models required for CCD algorithms do not provide a path to selecting detailed component designs. Ideally, design decisions should be coordinated across a system hierarchy, such as the one shown in Figure 1.2, to both ensure optimal transient system operation at a system level *and* to provide a realizable path toward selecting exact component designs to fit the system.



(a) General hierarchy in a complex system.



(b) Example hierarchy in an aircraft fuel thermal management system.

Figure 1.2. : Schematics illustrating hierarchical structure of complex systems.

Integrated (hierarchical) design optimization, or optimizing system design across multiple levels, has been researched previously with applications ranging from structural engine design to electrical circuit design. The authors in [41] leverage a genetic algorithm-based integrated design environment to design and optimize aeronautical piston engine components. In [42] the authors study the optimal design of a chassis integrated system consisting of steering, suspension and brake subsystems with emphasis on ensuring a favorable feel for the driver. In [43], the authors investigate the use of integrated multilevel design in optimizing material structure of porous metals. Martins et al. illustrated how hierarchical multi-objective optimization can be used for design improvements for analog integrated circuit layouts [44].

Recently the notion of integrated design optimization has also been utilized in the mechatronics community to help engineers across different disciplines accomplish complex design tasks [45, 46]. Researchers divide integrated system design into two levels, called macro-level and micro-level [47–50]; the macro-level process represents the high level procedure for the design of mechatronic systems specified according to individual design phases at the micro-level. The micro-level represents specific lower level design phases where individuals can structure design sub-tasks. *While these efforts focus on integrating and coordinating design across multiple levels, they typically do not consider transient device performance.* Rather, the multilevel optimization efforts are design-only and do not consider the implications of integrating feedback control into the optimization.

Within the controls community, hierarchical control is a methodology that has been studied in the context of complex systems, or systems comprised of multiple components (levels) each with distinctive dynamics. Hierarchical control architectures feature multiple controllers across the operation of a larger system architecture, each operating independently to accomplish a subtask on separate time scales. Much of the work seeks to derive analytical guarantees of stability and/or robustness under various assumptions [51–54]. Researchers recently experimentally validated the concept of hierarchical control for a fluid flow power system used for thermal management [55]. While their work successfully demonstrates the use of feedback control across a system level hierarchy, no notions of system (plant) design optimization were considered.

Finally, recent work has investigated combining architectural design optimization with CCD for a given architecture. In [56], the authors presented an algorithm to design and control various architectures for a vehicle suspension system. The higher-level architectural portion of the algorithm was used to determine an overall system setup, such as number of dampers to use, while a lower-level CCD algorithm solved a nested CCD problem for a given architecture. The authors extended a similar framework to the design of a single-split architecture for fluid-based thermal management [57]. However, an open loop control signal was used at the lower level rather than a CCD algorithm by modulating valve operation within the architecture. In both papers, the plant models used were relatively simple. In fact, the authors in [56] acknowledge that further work would be needed to investigate higher fidelity models for architectures deemed favorable by their algorithm. Figure 1.3 illustrates the available literature on the topics at hand. **Absent from the literature is a way to synthesize high-fidelity design optimization with feedback control design to create a larger hierarchical CCD algorithm for optimizing transient closed-loop performance of thermal systems.**

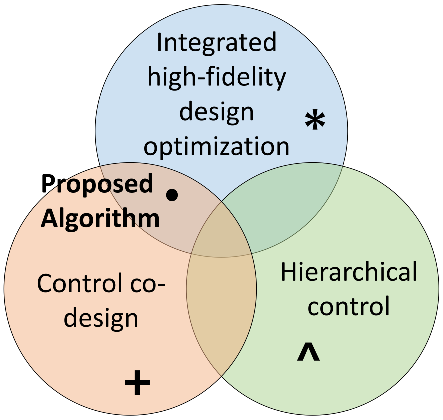


Figure 1.3. : Illustration of the contribution of this research. The symbols in each circle of the Venn diagram represent the following citations: + [7, 10–15, 18–21, 56, 58, 59], * [41–50], ^ [51–55].

1.2 Thesis Objectives

My main contribution in this dissertation is a new integrated design approach for improved transient performance of thermal management systems. More specifically, I develop a hierarchical control co-design (HCCD) algorithm aimed at optimizing the system design and feedback controller for a TMS across a system hierarchy. This is accomplished in part by integrating system level (SL) CCD with high fidelity component level (CL) design optimization. The lower-fidelity SL CCD algorithm incorporates feedback control into the design of a TMS to ensure controllability and robust transient response to exogenous disturbances, and the higher-fidelity CL design optimization algorithms provide a way of designing detailed components to achieve the desired performance needed at the SL. Key specifications are passed back and forth between levels of the hierarchy at each iteration to converge to an optimal design that is responsive to desired objectives at each level.

My system level (SL) CCD algorithm is specifically structured to capture critical elements of TMS performance. Namely, the algorithm explicitly leverages system efficiency, via entropy generation rate, as a design metric for both steady-state and transient performance. This is accomplished in part by structuring the algorithm to utilize a *parameterized first-principles model* that defines entropy generation rate as a state whose dynamics are coupled to the plant and control variables. This is a significant departure from existing CCD algorithms which generally use models that are parameterized only for a small set of variables that are determined *a priori* to be of significance [8, 10, 18, 21]. To facilitate optimization across the hierarchy, I then develop highly-detailed component level (CL) optimization algorithms for multiple components within a TMS architecture. More specifically, I present a new design methodology for transient design optimization of a shell and tube heat exchanger and an optimization algorithm for a set of fluid pumps within the TMS. The component level algorithms provide a method of optimizing detailed physical variables too complex for inclusion in a CCD algorithm while ensuring the designs adhere to detailed component level constraints.

1.3 Outline

The rest of this dissertation is organized as follows. In Chapter 2, I present a system level CCD algorithm along with a case study comparing it against more conventional design strategies for TMSs. In Chapter 3, I develop a component level dynamic optimization algorithm for a shell and tube heat exchanger and present an associated case study. In Chapter 4, I detail the integration of the system level and component level algorithms to create the HCCD algorithm and present a case study to show its advantages. In addition, I present a pump modeling and optimization framework that is included in the hierarchical algorithm. In Chapter 5, the high-fidelity thermal modeling software ATTMO (**A**FRL **T**ransient **T**hermal **M**anagement **O**ptimization) is used to validate the proposed design approach. Finally, conclusions are drawn from this work in Chapter 6 and potential future work is discussed.

2. SYSTEM LEVEL MODELING AND CONTROL CO-DESIGN

The first step in creating the hierarchical control co-design (HCCD) algorithm contributed in this dissertation concerns designing a CCD algorithm explicitly structured for thermal management systems. As stated in Chapter 1, CCD is a design strategy that incorporates feedback control at the *plant design* stage, thereby enabling intelligent design of thermal systems with favorable transient performance characteristics. In this chapter, a new methodology for CCD of thermal management systems is developed that explicitly leverages system efficiency, via entropy generation rate, as a design metric for both steady-state and transient performance. This is accomplished in part by structuring the algorithm to utilize a *parameterized first-principles model* that defines entropy generation rate as a state whose dynamics are coupled to the plant and control variables. This is a significant departure from existing CCD algorithms which generally use simplified models that are parameterized only for a small set of variables that are determined *a priori* to be of significance. The result of the proposed CCD algorithm is a system design that is not only optimized for steady-state efficiency, but that can be designed for robustness to transient disturbances while achieving said disturbance rejection with minimal compromise to system efficiency.

2.1 System Level Dynamic Model Description

In this section, I use mass and energy conservation laws to derive a first principles model of the system. I then augment the state vector with an entropy generation state to facilitate co-design for optimal transient and steady-state efficiency.

2.1.1 Notional Thermal Management System

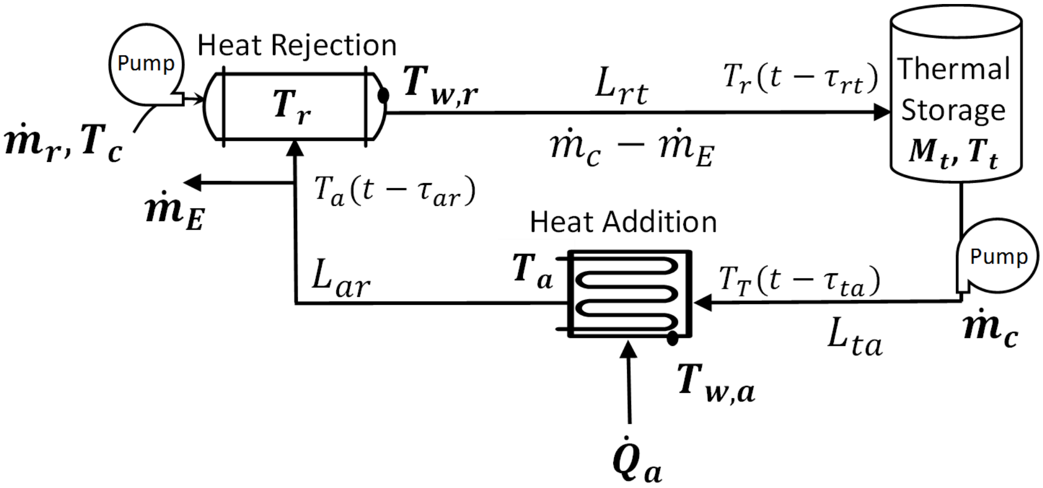


Figure 2.1. : Schematic of notional thermal management system.

The thermal management system considered here is given in Figure 2.1. The system contains four basic thermodynamic processes: thermal storage, heat addition, heat rejection, and thermal transport. Nearly all thermal-fluid systems are characterized by these thermodynamic processes. Therefore, the governing equations derived here can be adapted to a wide range of more complex system architectures such as systems with multiple storage elements. The single-phase cooling system studied here is representative of a simple aircraft fuel thermal management system.

Working fluid at temperature T_t flows from a storage tank at a mass flow rate $\dot{m}_c$. The mass of working fluid in the tank is denoted by M_t. A pulsed heat load $\dot{Q}_a$ is applied to the walls of the heat addition component. This component is modeled similarly to a cold plate with working fluid temperature T_a and capacitance C_a where I assume tubular channels of working fluid absorb heat applied to the walls of the component, which have a lumped temperature $T_{w,a}$ and mass $m_{w,a}$. Downstream of the heat addition component, the working fluid can exit the cycle at a flow rate $\dot{m}_E$; this mimics aircraft thermal management systems wherein the working fluid is actually fuel for the engine and exits the system as needed for propulsion and power. The remaining working fluid flows into the heat rejection component, modeled as a shell and tube heat exchanger with working fluid temperature T_r, capacitance C_r, wall temperature $T_{w,r}$, and wall mass $m_{w,r}$. The secondary fluid in the heat rejection component has a flow rate $\dot{m}_r$ and lumped temperature T_c. After the heat rejection process, the working fluid is routed back into the storage tank to complete the cycle.

Six dynamic states are used to describe the first law dynamics of the system shown in Figure 2.1: M_t, T_t, T_r, T_a, $T_{w,r}$, and $T_{w,a}$. The mass flow rate leaving the loop ($\dot{m}_E$), the pulsed heat load ($\dot{Q}_a$), and the secondary fluid temperature (T_c) are treated as exogenous disturbances; the mass flow rate exiting the storage tank ($\dot{m}_c$) and the secondary fluid mass flow rate ($\dot{m}_r$) are treated as control inputs in the system.

Using first principles, I define the state equations

$$\frac{dM_t}{dt} = -\dot{m}_E \tag{2.1a}$$
$$M_t\frac{dT_t}{dt} = \left(\dot{m}_c - \dot{m}_E\right)\left(T_r' - T_t\right) \tag{2.1b}$$
$$C_r\frac{dT_r}{dt} = A_{s,r}\alpha_r\left(T_{w,r} - T_r\right) + \left(\dot{m}_c - \dot{m}_E\right)c_p\left(T_a' - T_r\right) \tag{2.1c}$$
$$C_a\frac{dT_a}{dt} = A_{s,a}\alpha_a\left(T_{w,a} - T_a\right) + \dot{m}_c c_p\left(T_t' - T_a\right) \tag{2.1d}$$
$$m_{w,r}c_{p,w}\frac{dT_{w,r}}{dt} = A_{s,r}\alpha_r\left(T_r - T_{w,r}\right) + \dot{Q}_{c\to w} \tag{2.1e}$$
$$m_{w,a}c_{p,w}\frac{dT_{w,a}}{dt} = A_{s,a}\alpha_a\left(T_a - T_{w,a}\right) + \dot{Q}_a \;\;, \tag{2.1f}$$

where A_s represents heat transfer surface area. The α terms represent convective heat transfer coefficients and are functions of the respective fluid flow rates, cross-sectional flow areas, and fluid thermodynamic properties. The term $\dot{Q}_{c\to w}$ is the heat transfer rate from the secondary fluid to the wall of the heat rejection component calculated with an effectiveness-NTU method given as:

$$\dot{Q}_{c\to w} = \dot{m}_r c_{p,c}\left[T_c - \left(T_{w,r} + \left(T_c - T_{w,r}\right)e^{\frac{-\alpha_c A_{s,c}}{\dot{m}_r c_{p,c}}}\right)\right]. \tag{2.2}$$

Finally, I model thermal transport as a function of fluid flow time delays that are directly proportional to the pipe lengths between adjacent components. The time delays are inversely proportional to the mass flow rate for a given component. The term T_i' describes the delayed value $T_i\left(t - \tau_{ab}\right)$, and the time delays are defined by $\tau_{ab} = \frac{\rho A c L_{ab}}{\dot{m}}$ where L_{ab} is the length of pipe between locations a and b.

2.1.2 Dynamic Second Law Model of Entropy Generation Rates

In this section, I use second law principles to derive a dynamic model of the entropy generation rate within the system in Figure 2.1. I then linearize the system about a defined operating point and use simulations to illustrate the accuracy of the linearized model.

Like energy, entropy is an extensive property that can be transferred to and from a given control volume by heat transfer or mass exchange. However, the entropy statement of the second law states that “it is impossible for any system to operate in a way that entropy is destroyed” [60]; in other words, entropy generation can be zero or positive, but never negative. In all real physical systems, entropy is generated through irreversible processes such as mixing or viscous dissipation. For a generic control volume, the entropy rate balance is given by

$$\frac{dS_{cv}}{dt} = \sum_j \frac{\dot{Q}_j}{T_j} + \sum_i \dot{m}_i s_i - \sum_o \dot{m}_o s_o + \dot{S}_{gen,cv} \ , \tag{2.3}$$

where $\frac{dS_{cv}}{dt}$ represents the time rate of change of entropy in the control volume, $\sum_i \dot{m}_i s_i$ and $\sum_o \dot{m}_o s_o$ represent the transfer of entropy into and out of the control volume via mass flow, and $\sum_j \frac{\dot{Q}_j}{T_j}$ represents the rate of entropy transfer across the system boundaries due to the time rate of heat transfer $\dot{Q}_j$ occurring at boundary temperature T_j. The term $\dot{S}_{gen,cv}$ represents the rate of entropy generation due to irreversibilities within the control volume. Minimizing the rate of entropy generation within a given system ($\dot{S}_{gen,cv}$) is analogous to maximizing the amount of available work that can be extracted from the system. In other words, minimizing the rate of entropy generation maximizes the second law efficiency of the system.

Remark 1. In Eqn. (2.3), the term $\dot{S}_{gen,cv}$ is not the same as a typical time derivative. The rate of entropy generation $\dot{S}_{gen,cv}$ is an instantaneous rate and takes on a constant value at steady-state, unlike typical time differentials which become zero at steady-state. In other words, $\dot{S}_{gen,cv}(t)$ refers to the time rate of entropy generation in a control volume at time t.

Remark 2. For any real physical system, *both* the total entropy generation over time S_{gen} *and* the system's instantaneous rate of entropy generation $\dot{S}_{gen}$ must be nonnegative. In other words, $S_{gen}\left(t\right) \geq 0 \ \forall \ t$ and $\dot{S}_{gen}\left(t\right) \geq 0 \ \forall \ t$. For all real systems, entropy generation can never be regulated to zero; instead it will be some positive value due to the fact that purely reversible systems exist only theoretically.

To derive an expression for the total rate of entropy generation ($\dot{S}_{gen}$) in the system, I begin by applying second law principles to the storage tank:

$$\begin{aligned} \frac{dS_{tank}}{dt} &= M_t \frac{ds_t}{dt} + s_t \frac{dM_t}{dt} \\ &= \left(\dot{m}_c - \dot{m}_E\right) s_r' - \dot{m}_c s_t + \dot{S}_{gen,tank} \ . \end{aligned} \tag{2.4}$$

Using the relationships $\frac{ds}{dt} = \frac{c_p}{T}\frac{dT}{dt}$ and $s_2 - s_1 = c_p \ln\left(\frac{T_2}{T_1}\right)$, along with Eqn. (2.1a)-(2.1b), I can solve for an expression for $\dot{S}_{gen,tank}$ as a function of the current system state and the delayed system state:

$$\begin{aligned} \dot{S}_{gen,tank} = & \frac{c_p}{T_t}\left(\dot{m}_c - \dot{m}_E\right)\left(T_r' - T_t\right) \\ & - \left(\dot{m}_c - \dot{m}_E\right) c_p \ln\left(\frac{T_r'}{T_t}\right) . \end{aligned} \tag{2.5}$$

Similar techniques can be used to derive expressions for the entropy generation rates in the heat addition and heat rejection components. To derive the entropy generated in each of these components, I apply a control volume entropy balance to each heat exchanger accounting for entropy transfer due to heat and mass transfer as shown in Eqns. (2.6) - (2.7).

$$\begin{aligned}\frac{dS_{rej}}{dt} &= m_r\frac{ds_r}{dt} + m_{w,r}\frac{ds_{w,r}}{dt} = \frac{C_r}{c_p}\frac{ds_r}{dt} + m_{w,r}\frac{ds_{w,r}}{dt} \\ &= \frac{\dot{Q}_{c\to w}}{T_{b,r}} + \left(\dot{m}_c - \dot{m}_E\right)\left(s_a^{\prime} - s_r\right) + \dot{S}_{gen,rej} \quad ,\end{aligned} \tag{2.6}$$

$$\begin{aligned}\frac{dS_{add}}{dt} &= m_a\frac{ds_a}{dt} + m_{w,a}\frac{ds_{w,a}}{dt} = \frac{C_a}{c_p}\frac{ds_a}{dt} + m_{w,a}\frac{ds_{w,a}}{dt} \\ &= \frac{\dot{Q}_a}{T_{b,a}} + \dot{m}_c\left(s_t^{\prime} - s_a\right) + \dot{S}_{gen,add} \quad .\end{aligned} \tag{2.7}$$

I assume the boundary temperatures to be the dynamic wall temperatures; in other words, $T_{b,r} = T_{w,r}$ and $T_{b,a} = T_{w,a}$. Again making use of $\frac{ds}{dt} = \frac{c_p}{T}\frac{dT}{dt}$ and $s_2 - s_1 = c_p \ln\left(\frac{T_2}{T_1}\right)$, I define expressions for the entropy generation rates of the heat addition and heat rejection components:

$$\begin{aligned}\dot{S}_{gen,rej} = &\frac{C_r}{T_r}\frac{dT_r}{dt} + \frac{m_{w,r}c_{p,w}}{T_{w,r}}\frac{dT_{w,r}}{dt} \\ &-\frac{\dot{Q}_{c\to w}}{T_{b,r}} - \left(\dot{m}_c - \dot{m}_E\right)c_p\ln\left(\frac{T_a^{\prime}}{T_r}\right) \quad ,\end{aligned} \tag{2.8}$$

$$\begin{aligned}\dot{S}_{gen,add} = &\frac{C_a}{T_a}\frac{dT_a}{dt} + \frac{m_{w,a}c_{p,w}}{T_{w,a}}\frac{dT_{w,a}}{dt} \\ &-\frac{\dot{Q}_a}{T_{b,a}} - \dot{m}_c c_p\ln\left(\frac{T_t^{\prime}}{T_a}\right) \quad .\end{aligned} \tag{2.9}$$

Using the state equations given in Eqn. (2.1), I can define a dynamic expression for the entropy generation rate of each component as a function of the current state of the system.

I also account for entropy generation due to pumping losses for both the working fluid $\dot{m}_c$ and the secondary fluid $\dot{m}_r$. The pump losses are calculated from performance pump curve equations such that $\dot{S}_{gen,pump} = f\left(\dot{m}_c, \dot{m}_r\right) = c_1\dot{m}_c^2 + c_2\dot{m}_c + c_3 + c_4\dot{m}_r^2 + c_5\dot{m}_r + c_6$, where each c_i term is a constant coefficient for the given pump. I assume entropy generation in the pipes to be negligible in comparison to the pumping losses. Therefore, I define the total entropy generation rate in the system at time t as the sum of the individual rates of entropy generation for each component: $\dot{S}_{gen} = \dot{S}_{gen,tank} + \dot{S}_{gen,rej} + \dot{S}_{gen,add} + \dot{S}_{gen,pump}$.

To include the entropy generation rate dynamics in the overall system state vector, I introduce a tracking state ζ whose dynamic state derivative is defined by $\frac{d\zeta}{dt} = \dot{S}_{gen} - r$, where r is a desired reference for the entropy generation rate. The tracking state allows the formulation of a linear state feedback controller which can regulate the system entropy generation to an optimized nominal value of entropy generation.

Remark 3. Irreversibilities are often exacerbated during transient operation and can be influenced through real-time decision-making. Although thermodynamic expressions for entropy generation are not typically amenable to control synthesis and design, the utility of this formulation allows the design of model-based state feedback controllers that can impact system efficiency through regulation of entropy generation rate in real-time.

With the addition of the tracking state, I construct a dynamic model with seven states: M_t, T_t, T_r, T_a, $T_{w,r}$, $T_{w,a}$, and ζ. To linearize the model, I append the tracking state dynamics, $\frac{d\zeta}{dt}$, to the nonlinear equation set in Eqn. (2.1) and rewrite the system of equations as

$$Z_1 \begin{bmatrix} \frac{dM_t}{dt} & \frac{dT_t}{dt} & \frac{dT_r}{dt} & \frac{dT_a}{dt} & \frac{dT_{w,r}}{dt} & \frac{dT_{w,a}}{dt} & \frac{d\zeta}{dt} \end{bmatrix}^T = Z_2 \ , \tag{2.10}$$

where

$$Z_1 = \operatorname{diag}\left([1,\ \ M_t,\ \ C_r,\ \ C_a,\ \ m_{w,r}c_{p,w}\ \ m_{w,a}c_{p,w}\ \ 1]\right) \tag{2.11a}$$

$$Z_2 = \left\{ \begin{matrix} -\dot{m}_E \\ \left(\dot{m}_c - \dot{m}_E\right)\left(T_r' - T_t\right) \\ A_{s,r}\alpha_r\left(T_{w,r} - T_r\right) + \left(\dot{m}_c - \dot{m}_E\right)c_p\left(T_a' - T_r\right) \\ A_{s,a}\alpha_a\left(T_{w,a} - T_a\right) + \dot{m}_c c_p\left(T_t' - T_a\right) \\ A_{s,r}\alpha_r\left(T_r - T_{w,r}\right) + \dot{Q}_{c\to w} \\ A_{s,a}\alpha_a\left(T_a - T_{w,a}\right) + \dot{Q}_a \\ \dot{S}_{gen} - r \end{matrix} \right\}. \tag{2.11b}$$

I define the state vector x, time delayed state vector x_d, disturbance input vector d, and control input vector u as

$$x = \begin{bmatrix} M_t & T_t & T_r & T_a & T_{w,r} & T_{w,a} & \zeta \end{bmatrix}^T, \tag{2.12a}$$

$$x_d = \begin{bmatrix} 0 & T_t' & T_r' & T_a' & 0 & 0 & 0 \end{bmatrix}^T, \tag{2.12b}$$

$$u = \begin{bmatrix} \dot{m}_c & \dot{m}_r \end{bmatrix}^T, \quad d = \begin{bmatrix} \dot{m}_E & \dot{Q}_a & T_c & r \end{bmatrix}^T \tag{2.12c}$$

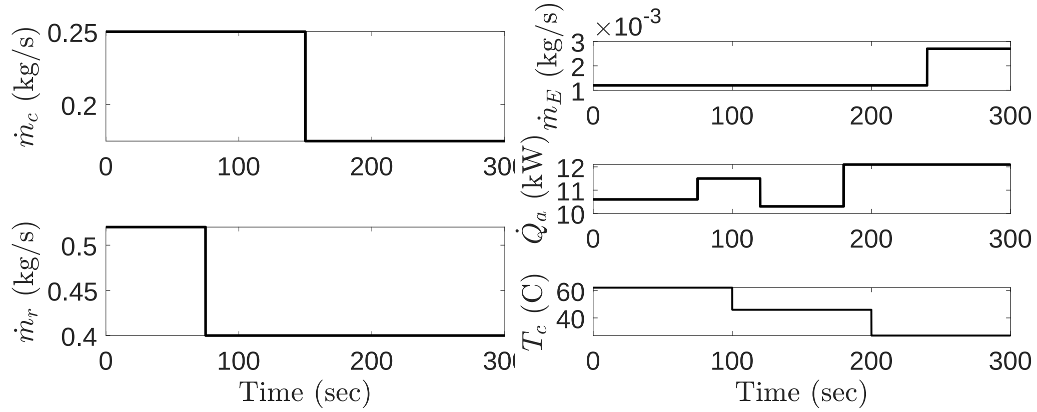
and define the linearized model as

$$
\begin{gathered}
Z_1\dot{x} = \tilde{A}_0\Delta x + \tilde{A}_1\Delta x_d + \tilde{B}\Delta u + \tilde{B}_d\Delta d \ , \\
\tilde{A}_0 = \begin{bmatrix} \frac{\partial(Z_2)_1}{\partial(x)_1} & \cdots & \frac{\partial(Z_2)_1}{\partial(x)_7} \\ \vdots & \ddots & \vdots \\ \frac{\partial(Z_2)_7}{\partial(x)_1} & \cdots & \frac{\partial(Z_2)_7}{\partial(x)_7} \end{bmatrix}, \\
\tilde{A}_1 = \begin{bmatrix} \frac{\partial(Z_2)_1}{\partial(x_d)_1} & \cdots & \frac{\partial(Z_2)_1}{\partial(x_d)_7} \\ \vdots & \ddots & \vdots \\ \frac{\partial(Z_2)_7}{\partial(x_d)_1} & \cdots & \frac{\partial(Z_2)_7}{\partial(x_d)_7} \end{bmatrix} \ , \\
\tilde{B} = \begin{bmatrix} \frac{\partial(Z_2)_1}{\partial(u)_1} & \frac{\partial(Z_2)_1}{\partial(u)_2} \\ \vdots & \vdots \\ \frac{\partial(Z_2)_7}{\partial(u)_1} & \frac{\partial(Z_2)_7}{\partial(u)_2} \end{bmatrix}, \quad \tilde{B}_d = \begin{bmatrix} \frac{\partial(Z_2)_1}{\partial(d)_1} & \cdots & \frac{\partial(Z_2)_1}{\partial(d)_4} \\ \vdots & \ddots & \vdots \\ \frac{\partial(Z_2)_7}{\partial(d)_1} & \cdots & \frac{\partial(Z_2)_7}{\partial(d)_4} \end{bmatrix} \ ,
\end{gathered}
\tag{2.13}
$$

where Δ terms are perturbations from the nominal (equilibrium) operating point (x^e, u^e, d^e). The linear model is obtained by manipulating Eqn. (2.13) to yield Eqn. (2.14).

$$
\begin{gathered}
\dot{x} = A_0\Delta x + A_1\Delta x_d + B\Delta u + B_d\Delta d \\
A_0 = Z_1^{-1}\tilde{A}_0, \quad A_1 = Z_1^{-1}\tilde{A}_1 \\
B = Z_1^{-1}\tilde{B}, \quad B_d = Z_1^{-1}\tilde{B}_d
\end{gathered}
\tag{2.14}
$$

It is worth noting that the tank mass (M_t) is an uncontrollable state in the state-space equation set. For control design, I separate the system into its controllable and uncontrollable systems, where the uncontrollable system is simply the tank mass state.

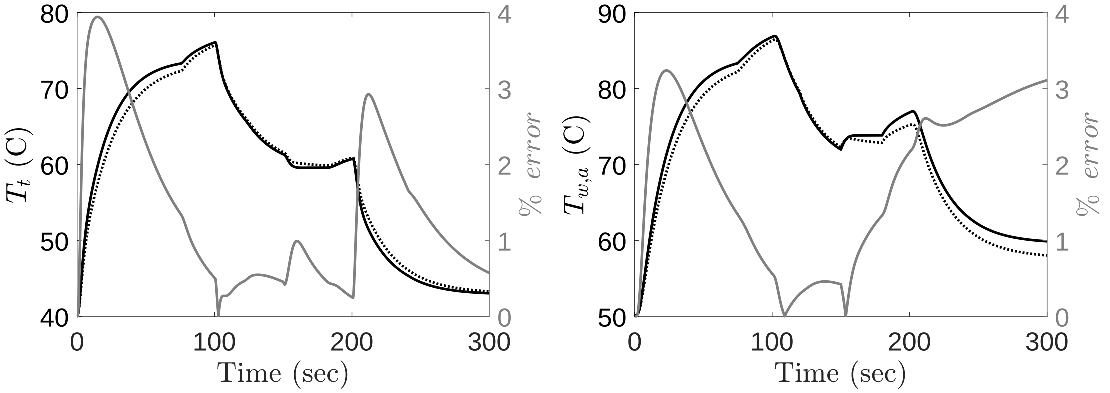


(a) Control input perturbations (b) Disturbance input perturbations

Figure 2.2. : Nonlinear and linear simulation results: exogenous input profiles.

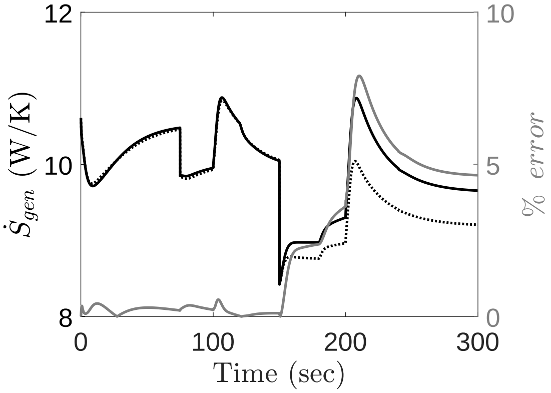
To validate the linearized model formulation, I simulate the nonlinear (Eqn. (2.1)) and linearized (Eqn. (3.23)) models with the following system parameters: $C_a = C_r =$ 2.6 kJ/K, $m_{w,r} = m_{w,a} =$ 7.1 kg, and $A_{s,r} = A_{s,a} = 0.40$ m^2. I assume copper walls for the heat exchanger components, stainless steel pipes connecting each component, and water as both the working fluid and secondary fluid. The pipes connecting each component are 0.91 m (3 ft) long with 0.013 m (1/2 in) diameter. The parameter set was chosen such that the system nominally absorbs 10 kW in the heat addition, or cold plate, component.

The control input signals, shown in Fig. 2.2a, are step commands away from the nominal operating point with magnitudes $\dot{m}_c^e = 0.25$ kg/s and $\dot{m}_r^e = 0.40$ kg/s. The disturbance input profiles are given in Figure 2.2b and the nominal values for each disturbance are $\dot{m}_E^e = 0$ kg/s, $\dot{Q}_a^e = 10$ kW, and $T_c^e = 27$ °C. I begin the simulation with each dynamic state at its nominal operating value. Simulation results for the storage tank working fluid temperature, the heat addition component wall temperature, and the total system entropy generation rate are shown in Fig. 2.3; results for the heat addition component working fluid temperature, heat rejection component working fluid temperature, and heat rejection component wall temperature are shown in Fig. 2.4. The time history of the entropy generation rate is examined by post-processing



(a) Temperature of working fluid in storage tank

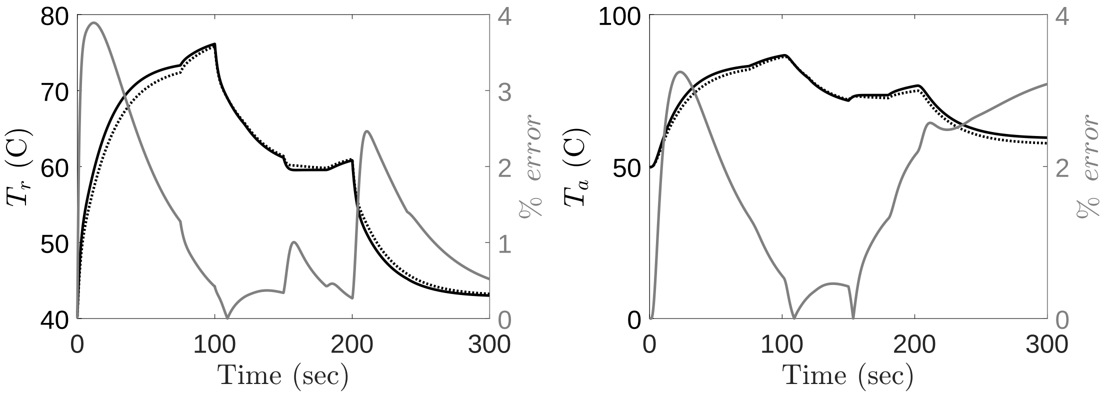
(b) Temperature of walls in heat addition component



(c) Total entropy generation rate

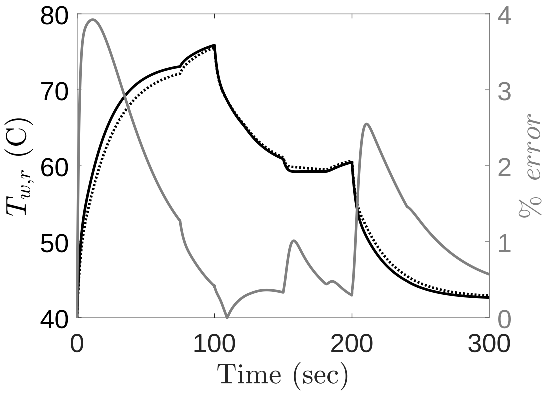
Figure 2.3. : Comparison of nonlinear and linear model states, as well as percent error between the two. The nonlinear model states are plotted using a solid black line, and the linearized model states are plotted using a dotted black line.

state temperatures and exogenous inputs to compute $\dot{S}_{gen} = f(x, u, d)$ in accordance with Eqns. (2.5)-(2.9). Performing these calculations for both models shows that the linearized model predicts the trends of the system's entropy generation rate (Figure 2.3c) quite well. Additionally, the right hand axes in Figure 2.3 show the percent error of the linear model with respect to the full nonlinear model as a function of time. Through simulation, I found the linearized model states match the full nonlinear model states to within 10% error in the presence of perturbations to any control or disturbance input up to 30% about the nominal value.



(a) Temperature of working fluid in heat rejection component

(b) Temperature of working fluid in heat addition component



(c) Temperature of walls in heat rejection component

Figure 2.4. : Further comparison of nonlinear and linear model states, as well as percent error between the two. The nonlinear model states are plotted using a solid black line, and the linearized model states are plotted using a dotted black line.

2.2 Nested CCD Algorithm

While simultaneous CCD algorithms generally guarantee optimal solutions [6], they require that only a single objective function be used to represent both plant and control design objectives. This may be suitable for classes of mechanical systems, where design goals and control objectives might both involve designing a dynamically stable system [8, 13]. However, when designing thermal management systems, the performance aspects of interest from a plant design perspective are often unrelated to, or different from, the transient performance aspects dictated by control design. For example, when designing an aircraft fuel thermal management system, less mass is desirable because of the significant impact weight has on the fuel efficiency of the aircraft. From a controls perspective, it is crucial that a thermal management system is robust to exogenous disturbances, such as pulsed heat loads, which occur during system operation. By designing the algorithm with separate plant and controller objective functions, one can more easily consider each of these performance aspects.

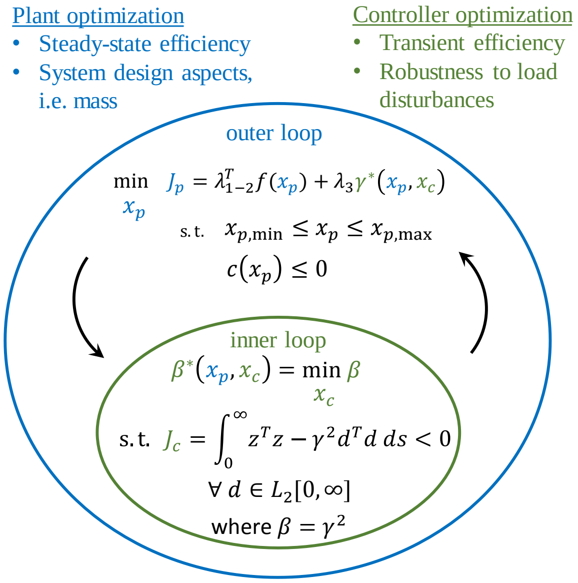

Plant optimization
• Steady-state efficiency
• System design aspects, i.e. mass
Controller optimization
• Transient efficiency
• Robustness to load disturbances
outer loop
$\min_{x_p} J_p = \lambda_{1-2}^T f(x_p) + \lambda_3 \gamma^*(x_p, x_c)$
s.t. $x_{p,\min} \leq x_p \leq x_{p,\max}$
$c(x_p) \leq 0$
inner loop
$\beta^*(x_p, x_c) = \min_{x_c} \beta$
s.t. $J_c = \int_0^\infty z^T z - \gamma^2 d^T d \, ds < 0$
$\forall\, d \in L_2[0,\infty]$
where $\beta = \gamma^2$

Figure 2.5. : Overview of proposed nested CCD algorithm.

These requirements lend themselves towards a nested CCD architecture in which the inner optimization problem is solved at each iteration of the outer loop. The two objective functions are then linked via the optimal decision variable(s) from the inner loop. I propose the nested CCD algorithm shown in Figure 2.5. One feature that differentiates the proposed approach from other nested CCD algorithms is that I solve a nonlinear optimization problem in the outer loop and re-linearize the full first principles model at each iteration in the inner loop to solve the optimal control problem. Therefore, the model-based controller is based upon a fully parameterized version of the nonlinear dynamic model such that the controller behaves accurately for perturbations in any input channel.

The outer loop of the nested algorithm, discussed in Sec. 2.2.1, is the plant design optimization algorithm. The plant objective function, J_p, contains terms for optimizing system mass and steady-state entropy generation rate. In addition, the solution of the inner loop (control optimization), γ^*, is appended to the plant objective function to penalize plant designs with poor potential for transient closed-loop performance. The specific algorithm used to determine γ^* will be discussed in Section 2.2.2. I note that the plant objective function described here is constructed to reflect critical design decisions typically considered for aircraft; however, the objective function could be parameterized otherwise to reflect design considerations relevant to other applications such as economic cost in ground-based applications.

2.2.1 Outer Loop Plant Optimization

As an initial set of plant design variables, I define the outer plant optimization loop with nine decision variables given by

$$x_p = [L_{ta},\ L_{ar},\ L_{rt},\ L_{ref,r},\ L_{ref,a},\ f_r,\ \dot{m}_c^e,\ \dot{m}_r^e,\ M_T^e]^T \ ,$$

where L_{ta}, L_{ar}, and L_{rt} represent the lengths between each of the components, $\dot{m}_c^e$ is the commanded mass flow rate of primary working fluid at the nominal operating point, $\dot{m}_r^e$ is the commanded mass flow rate of secondary coolant in the heat rejection component at the nominal operating point, and M_T^e is the nominal amount of mass in the storage tank. The terms $L_{ref,r}$ and $L_{ref,a}$ represent the total length of working fluid tubing in the heat rejection component and heat addition component, respectively. Additionally, the term f_r represents a scaling factor due to the effect of fins on the heat transfer surface area over which the secondary coolant flows in the heat rejection component and for the mass of the walls in the heat rejection component. From this set of decision variables, the necessary geometric parameters and system properties of the optimal heat exchangers are derived as given in Eqn. (2.15).

$$A_{s,a} = \pi L_{ref,a} D_i \tag{2.15a}$$

$$A_{s,r} = \pi L_{ref,r} D_i \tag{2.15b}$$

$$C_r = \left(A_{c,r} L_{ref,r}\right) \rho c_p \tag{2.15c}$$

$$C_a = \left(A_{c,a} L_{ref,a}\right) \rho c_p \tag{2.15d}$$

$$m_{w,a} = \left(\frac{\pi}{4} L_{ref,a} \left(D_o^2 - D_i^2\right)\right) \rho_w \tag{2.15e}$$

$$m_{w,r} = \left(\frac{\pi}{4} L_{ref,r} \left(D_o^2 - D_i^2\right)\right) \rho_w f_r \tag{2.15f}$$

$$A_{s,c} = f_r \pi D_o L_{ref,r} \tag{2.15g}$$

The inner and outer diameters of the heat exchanger tubes are assumed fixed, along with the diameter of the pipes connecting each component. In addition, I assume a fixed set of dimensions for the footprint of the heat rejection heat exchanger. Given the fixed allowable length L_{hx}, width w_{hx}, and height h_{hx}, the number of tubes needed in the heat rejection component is calculated as $N_r = \frac{L_{ref,r}}{L_{hx}}$. The cross-sectional area for the secondary coolant flow is then computed as $A_{c,c} = h_{hx} w_{hx} - N_r A_{c,r}$, where $A_{c,r}$ is the cross-sectional area of a single tube.

Remark 4. By using a control-oriented modeling approach for the heat exchanger components, very few restrictions are placed on the physical design of the components themselves. Rather, the optimization is designed to ensure that the interactions between the components at a system level enable the targeted control performance to be achieved. Component level design optimization will be used to optimize individual heat exchanger geometries in Chapter 3.

Outer plant optimization loop

$$\begin{aligned}
\min_{x_p} \quad & J_p\left(x_p, x_c\right) = \lambda_1 \cdot \left[M_{sys}\left(x_p\right)\right] + \lambda_2 \cdot \left[\dot{S}^e_{gen}\left(x_p\right)\right] + \lambda_3 \cdot \left[\gamma^*\left(x_p, x_c\right)\right] \\
\text{s.t.} \quad & x_{p,min} \leq x_p \leq x_{p,max} \\
& T_{c,low} \leq T^e_c\left(x_p\right) \leq T_{c,high} \\
& \dot{S}^e_{gen,total}\left(x_p\right) \geq 0 \\
& \dot{S}^e_{gen,tank}\left(x_p\right), \dot{S}^e_{gen,rej}\left(x_p\right), \dot{S}^e_{gen,add}\left(x_p\right) \geq 0 \\
& \text{rank}\left[\text{ctrb}\left(A_0 + A_1, B\right)\right] = n - 1
\end{aligned} \tag{2.16}$$

Inner control optimization loop

$$\begin{aligned}
\beta^*\left(x_p, x_c\right) = \min_{x_c} \quad & \beta \\
\text{s.t.} \quad & \text{LMI}_1 < 0 \\
& Q_1 > 0 \\
& \beta > 0
\end{aligned} \tag{2.17}$$

$$\text{where } \gamma^* = \sqrt{\beta^*},\ K = Y Q_1^{-1}$$

and

$$x_c = U_{1-2}, Q_{1-3}, R_{1-2}, Y, \beta$$

For the outer plant optimization loop, the plant objective function is given in Eqn. (2.16) and the total system mass is defined as

$$\begin{aligned}
M_{sys} = M^e_T + \left(\rho A_c\right)_{pipe}\left(L_{ta} + L_{ar} + L_{rt}\right) \\
+ m_{w,a} + m_{w,r} + \rho A_c\left(L_{ta} + L_{ar} + L_{rt}\right) + \\
\frac{1}{c_p}\left(C_a + C_r\right),
\end{aligned} \tag{2.18}$$

where M^e_T represents the nominal amount of working fluid in the storage tank. Physical masses for the pipes connecting each component and the wall masses of each heat exchanger are accounted for, as well as the mass of working fluid in each heat exchanger component and each connecting pipe. In other words, I assume the entire system is filled with working fluid at all times. Note that masses of the physical pumps are not yet accounted for. This will be taken into consideration in Chapter 4.

The term $\dot{S}^e_{gen}$ in Eqn. (2.16) is the nominal (steady-state) entropy generation rate of a given plant design; recall that $\dot{S}^e_{gen}$ takes on a constant positive value at steady-state. The term γ^* is supplied to the outer loop from the inner control optimization loop and is an optimal value for robust transient performance *for a given plant design.* Therefore, a plant design with poor transient performance potential is penalized in the outer optimization loop by the term $\lambda_3\gamma^*$. The weighting terms λ_{1-3} can be used to place relative weights on the different components of the objective function, thereby allowing the user to minimize a weighted combination of various elements of static and dynamic system performance.

There are several key constraints in Eqn. (2.16) that potential plant designs must adhere to. Bounds are placed on the nominal value of the secondary fluid temperature T^e_c. Additionally, the overall system and each component must not violate the second law of thermodynamics. Finally, the rank constraint on the linearized model ensures the plant design permits a fully controllable linearization such that all states of the system dynamics can be controlled except for the uncontrollable state associated with the mass of working fluid in the storage tank. I note that controllability for the system is defined according to the non-delayed system. In other words, I constrain the pair $(A_0 + A_1, B_2)$ to be fully controllable. The control synthesis presented in Section 2.2.2 then ensures the closed loop time delayed system to be asymptotically stable for time delays up to a maximum user-defined value.

2.2.2 Inner Control Loop Optimization

Here I describe the inner loop of the nested CCD algorithm which utilizes an H_∞-based control algorithm to synthesize a feedback controller for a given plant design. Note that other optimal control algorithms can be accommodated in the proposed algorithm, depending on the primary control objectives for a given problem. For the algorithm considered here, the primary decision variable in the inner loop is $\beta = \gamma^2$, where γ connects the inner and outer optimization loops of the nested CCD algorithm. An H_∞ control synthesis typically entails regulating output signals to desired reference values in the presence of exogenous disturbances, while minimizing control effort. In the context of thermal management, a general H_∞ controller might maintain the temperature of the walls in the heat addition component ($T_{w,a}$) to a desired threshold in the presence of a pulsed power load ($\dot{Q}_a$) while minimizing other performance objectives such as the commanded flow rate of primary working fluid $\dot{m}_c$. The emphasis on robustness, however, tends to overlook system efficiency during operation. Therefore, I seek to instead derive a control law that will not only regulate performance outputs such as state temperatures, but do so while *ensuring that control actuation does not exacerbate adverse effects on transient system efficiency.* The following control synthesis is adapted from the work of Fridman, et al. [61].

$$\begin{bmatrix}
Q_2+Q_2^T & \Sigma & 0 & 0 & 0 & Q_1C^T & Y^TD^T & 0 & Q_2^T & 0 & Q_2^T & h_1Q_2A_1^T & 0 \\
* & -Q_3-Q_3^T & B_d & h_1R_1 & 0 & 0 & 0 & 0 & Q_3^T & 0 & Q_3^T & h_1Q_3A_1^T & 0 \\
* & * & -\gamma^2 I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & * & * & -h_1R_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & * & * & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & * & * & * & * & -I & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & * & * & * & * & * & -I & 0 & 0 & 0 & 0 & 0 & 0 \\
* & * & * & * & * & * & * & -U_1 & 0 & 0 & 0 & 0 & 0 \\
* & * & * & * & * & * & * & * & -U_1 & 0 & 0 & 0 & 0 \\
* & * & * & * & * & * & * & * & * & -U_2 & 0 & 0 & 0 \\
* & * & * & * & * & * & * & * & * & * & -U_2 & 0 & 0 \\
* & * & * & * & * & * & * & * & * & * & * & -h_1R_1 & 0 \\
* & * & * & * & * & * & * & * & * & * & * & * & -h_2R_2
\end{bmatrix} \leq 0 \tag{2.19}$$

To begin, I define a vector $z(t)$ of performance outputs to be regulated and cost function $J_c(d)$ as

$$
\begin{aligned}
J_c(d) &= \int_0^\infty z^T z - \gamma^2 d^T d \ \ ds, \\
z(t) &= \begin{Bmatrix} Cx(t) \\ Du(t) \end{Bmatrix}.
\end{aligned}
\tag{2.20}
$$

The state feedback control law $u(t) = Kx(t) = \left[YQ_1^{-1}\right] x(t)$ is synthesized by finding the smallest value of $\gamma > 0$ that satisfies $J_c(d) < 0$ for all nonzero $d \in L_2^q[0, \infty)$. This is attainable if there exist matrices $Q_1 > 0,\ U_1,\ U_2,\ Q_2,\ Q_3,\ R_1,\ R_2$, and Y satisfying the LMI in Eqn. (2.19). Within Eqn. (2.19), the variable h_1 represents the maximum time delay that would be encountered during operation. The control synthesis here aims to guarantee asymptotic stability for the closed loop time delayed system for all time delays less than this maximum expected value.

The expression $J_c\left(d\right) < 0$ can be equivalently written as

$$
\int_0^\infty z^T z \ \ ds < \int_0^\infty \gamma^2 d^T d \ \ ds \tag{2.21}
$$

for all nonzero $d \in L_2^q[0, \infty)$. My primary control objectives are to 1) regulate the heat addition component wall temperature and 2) regulate the system's transient entropy generation rate to its nominal value $\dot{S}^e_{gen}$. Within the inner loop, a plant design that is inherently less robust to transient disturbances will result in a larger minimum value of γ in Eqn. (2.21). In other words, for a higher value of γ, the upper bound on the magnitude of the transient performance objectives $z^T z$ increases. This implies a lower threshold magnitude of disturbances for which I can satisfy my control objectives.

By introducing a variable change defined as $\beta = \gamma^2$ into Eqns. (2.20)-(2.21), I formulate the optimization problem shown in Eqn. (2.17) to minimize β, and thus the performance output vector z, in the presence of disturbance vector d. By further defining the minimum value of β from Eqn. (2.17) to be β^*, I can compute the optimal (minimum) achievable value of $\gamma > 0$ from the original cost function in Eqn. (2.20) as $\gamma^* = \sqrt{\beta^*}$.

Remark 5. The control synthesis presented here is valid for all disturbances d in the L_2 space defined by $d \in L_2^q[0, \infty)$, where q is the dimension of the disturbance vector. The L_2 space is the set of square-integrable L_2 functions for which the integral $\int_{-\infty}^{\infty} |f(x)|^2 \ dx$ has a finite value. This includes, for example, sets of rectangular heat pulses $\dot{Q}_a$ perturbed from a nominal heat load value $\dot{Q}_a^e$ such as those in Figure 2.2b. For other types of disturbance signals, such as sinusoids, the integral would not have a finite value and an alternative control method would be needed to guarantee stability.

2.3 Merits of CCD for Thermal Management Systems: Comparison to Conventional Design Methods

In this section I compare the transient performance of two separate system design strategies: conventional system design and the proposed nested co-design approach. Most importantly, I highlight the improvement in robustness and transient system efficiency that is achieved using the proposed approach. For the Conventional Design case, I minimize two static performance metrics— the mass and steady-state entropy generation rate of the plant— prior to designing the feedback controller. In other words, the $\lambda_3\gamma^*$ term from the outer loop cost function in Eqn. (2.16) is omitted and the weighting penalties on system efficiency (λ_2) and mass (λ_1) are set equal to one. Once the plant is optimized, a subsequent H_∞ state feedback controller is designed in accordance with Eqn. (2.17).

I use the proposed nested CCD algorithm to design two separate systems: Co-design A and Co-design B. In the Co-design A case, I set $\lambda_1 = 1$, $\lambda_2 = 1$, and $\lambda_3 = 100$; in other words, this case places a high penalty on closed-loop performance and robustness. For the Co-design B case, I set $\lambda_1 = 1$, $\lambda_2 = 1$, and $\lambda_3 = 1$ to normalize the relative contributions of system mass, static entropy generation rate, and transient performance to the overall objective function J_p given the differences in magnitude expected between the various contributions. This case is used to achieve a near-equal balance on static system design and transient performance. In each case, the controller is designed with performance output matrices, defined in Eqn. (2.20), given by

$$C = \begin{bmatrix} 0.02 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 \end{bmatrix}, \qquad D = \begin{bmatrix} 0 & 0 \end{bmatrix}, \tag{2.22}$$

in order to regulate three states to their nominal values: heat addition component surface (wall) temperature ($T_{w,a}$), system entropy generation rate ($\dot{S}_{gen}$), and storage tank temperature (T_t). For both cases, I place a higher penalty on regulating the system efficiency than surface temperature and storage tank temperature as evidenced by the relative weights of the terms in the C matrix in Eqn. (2.22), where the 2 entry weights the regulation of $T_{w,a}$, the 0.02 entry weights the storage tank temperature, and the 5 entry weights $\dot{S}_{gen}$ to the respective nominal value. The penalty on regulating the storage tank temperature helps to avoid temperature 'drift' in the system, thus making the system more robust to heat input disturbances.

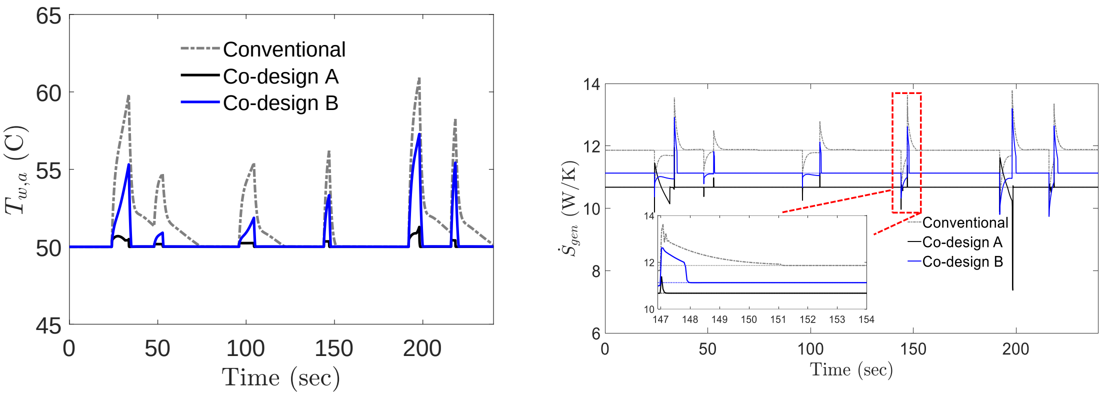
I leverage a multistart approach in all cases to avoid suboptimal solutions. The outer loop optimization problems are solved using the NOMAD global nonlinear optimization algorithm and the toolbox OPTI [62], and the inner loop optimization problems are solved using Yalmip [63]. The resulting optimal values for the decision variables in both the conventionally designed and co-designed cases are shown in Table 2.1. To regulate $T_{w,a}$ to its nominal value, a temperature difference is required between the working fluid temperature T_a and the wall temperature $T_{w,a}$. This temperature difference can be maintained largely by rejecting heat via the heat rejection component. This heat rejection is facilitated by either adding heat transfer area (more tubing) in the heat rejection component or by leveraging higher mass flow rates for

Table 2.1. : Optimal Values of plant and control decision variables, and total system mass.

Decision Variable	**Lower Bound**	**Upper Bound**	**Conv. Design**	**Co-des. A**	**Co-des. B**
L_{ta} (m)	0.61	6.1	0.61	0.61	0.61
L_{ar} (m)	0.61	6.1	0.61	0.61	0.61
L_{rt} (m)	0.61	6.1	0.61	0.61	0.61
$L_{ref,r}$ (m)	5.0	30	9.95	21.3	19.4
$L_{ref,a}$ (m)	5.0	30	5.0	5.0	5.0
f_r	1.01	2.0	1.42	1.76	1.01
M_t^e (kg)	2.0	15.0	2.0	2.0	2.0
$\dot{m}_c^e$ (kg/s)	0.095	0.155	0.150	0.095	0.119
$\dot{m}_r^e$ (kg/s)	0.095	0.295	0.295	0.243	0.256
γ^*			**7.64**	**4.57**	**6.42**
Total Mass (kg)			**10.5**	**19.1**	**12.7**
$\mathbf{S_{gen}^e}$ **(W/K)**			**11.9**	**10.7**	**11.1**

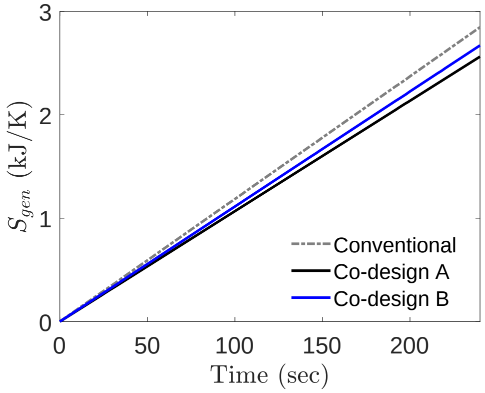
the secondary fluid. The heat rejection in turn helps to regulate the storage tank temperature to avoid temperature drift in the system.

In the Conventional Design case, the optimization minimizes a combination of mass and steady-state efficiency with no notion of how it may affect actuation. As a result, the conventional design opts for a smaller total tube length (9.95 m) in the heat rejection component than does Co-design A. In turn, it relies more heavily on using the secondary fluid flow rate to achieve the required heat transfer; in this case, the nominal secondary fluid flow rate is $\dot{m}_r^e = 0.295$ kg/s. *In other words, the Conventional Design case opts for high mass flow rates for both primary and secondary working fluid.* While the high mass flow rates yield a system design with a lower total mass, the steady-state entropy generation is adversely affected due to pumping power. Conversely, Co-design A features a lower nominal mass flow rate for the primary fluid ($\dot{m}_c^e = 0.095$ kg/s) and the secondary fluid ($\dot{m}_r^e = 0.243$ kg/s) but requires a longer length of total tubing ($L_{ref,r} = 21.3$ m) to help increase the rejection side heat transfer surface area and ensure the system is more robust to the pulsed heat load disturbances. As a result, Co-design A features a γ^* value 40% smaller than that of the Conventional Design, indicating that the optimal co-design system is more robust to the disturbance signals. The improved robustness of Co-design A can be seen in Figs. 2.6a-2.6b.



(a) Heat addition wall temperature

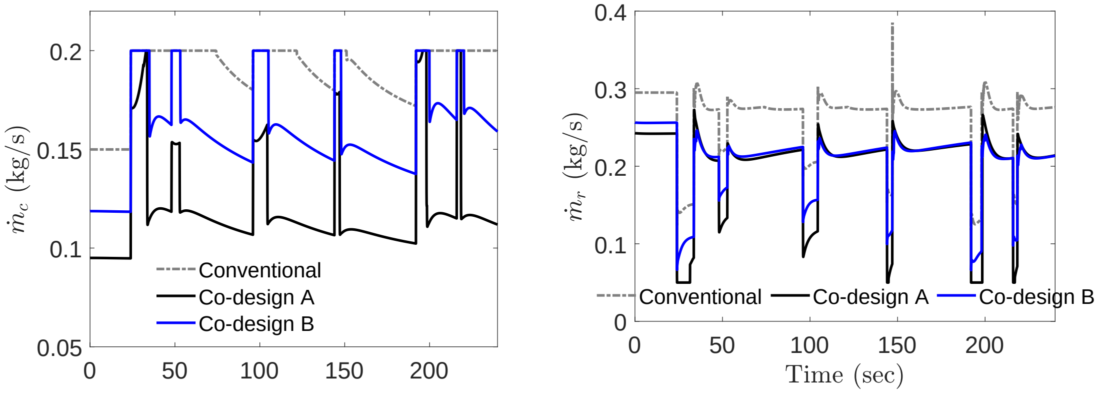
(b) Entropy generation rate



(c) Total entropy generated

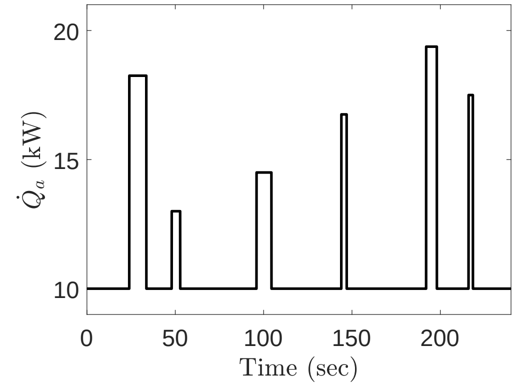
Figure 2.6. : Conventional design vs. co-design case study: selected performance outputs.

The wall temperature of the heat addition component is shown in Fig. 2.6a, while the system entropy generation rate is shown in Fig. 2.6b. The heat input disturbance profile for this case study is shown in Fig. 2.7c and is meant to resemble the types



(a) Commanded control input: primary fluid mass flow rate

(b) Commanded control input: secondary fluid mass flow rate



(c) Pulsed load disturbances

Figure 2.7. : Conventional design vs. CCD case study: system inputs.

of pulsed heat loads that are encountered in applications such as aircraft thermal management. The nominal heat pulse represents some steady-state amount of power to be dissipated due to electronics cooling, and the higher heat pulses are meant to represent fast transients that may result, for example, from directed weapon firings. The commanded control signals for each system are shown in Fig. 2.7. Recall that $\dot{m}_c$ is the mass flow rate of primary working fluid through the system and $\dot{m}_r$ is the mass flow rate of secondary fluid through the heat rejection component. For these simulations, constant values of $\dot{m}_E = 0$ kg/s and $T_c = 10$ °C were used for the exit flow rate and secondary coolant temperature, respectively.

As shown in Fig. 2.6a, the wall temperature of the heat addition component in Co-design A remains within approximately two degrees of the reference in the presence of the heat pulses. Conversely, the Conventional Design is less successful in regulating the wall temperature to its 50 °C reference as the surface temperature rises to a maximum of approximately 61 °C during the penultimate heat pulse. Moreover, the Conventional Design is less successful in regulating the entropy generation rate to its nominal value (represented by a dashed line in Fig. 2.6b). During the transient heat pulses, the controller in Co-design A is able to tightly regulate the system's entropy generation rate to its steady-state (nominal) operating point as evidenced by the plot enclosed in the bottom left corner of Fig. 2.6b showing the entropy generation rates from 147-154 seconds. This regulation, along with the lower steady-state entropy generation rate in the Co-Design A case, results in 10% less total entropy generated during the process as shown by Fig. 2.6c. This is critical in the context of dynamic thermal management. The larger spikes in the entropy generation rate of the Conventional Design case could lead to a lower thermal endurance for an integrated thermal management system. In mission critical applications, such as aircraft fuel thermal management, the increased thermal endurance in the co-designed system directly translates to tangible benefits such as decreased fuel consumption or increased military capability with respect to weapon firings.

The differences in transient performance between the Conventional Design and Co-design A can be explained by examining the commanded control inputs for each design (Fig. 2.7). Because the conventionally designed case relies much more heavily on the use of mass flow for providing its cooling, it is significantly less robust to the heat pulses. During each transient heat pulse, the conventionally designed system attempts to increase the primary working fluid flow rate $\dot{m}_c$ further above nominal. This causes input saturation, and the system is unable to regulate the wall temperature $T_{w,a}$ and entropy generation rate $\dot{S}_{gen}$ as desired. On the other hand, Co-design A requires lower working fluid flow rates to achieve the necessary heat transfer and is able to modulate the control inputs to maintain a surface temperature much closer to the

50 °C nominal value. The maximum surface temperature $T_{w,a}$ in the co-designed system is almost 10 °C lower than in the conventionally designed system. Moreover, the flow rates during the transients for Co-Design A result in smaller transient spikes in entropy generation and facilitate better regulation of transient entropy generation.

While Co-design A featured improved transient performance capabilities for both entropy generation and temperature regulation, the design resulted in a system with 82% more mass than the conventional case. Therefore, I use the nested CCD algorithm for Co-design B to achieve an intermediate design with *both* desirable transient performance *and* lower system mass. Recall that in Co-design B, I used weights of $\gamma_1 = 1$, $\gamma_2 = 1$, and $\gamma_3 = 1$ in the outer loop optimization problem. The resulting system features a total system mass of 12.7 kg, which is only a 21% increase from the Conventional Design. Moreover, Co-design B has $\gamma^* = 6.42$, a value still 16% lower than the Conventional Design. Figure 2.6a shows the maximum wall temperature in Co-design B is 57 °C, a temperature rise roughly 4 degrees less than the Conventional Design case. More importantly, Co-design B still regulates the entropy generation rate of the system closer to nominal than the conventional design and results in 6% less total entropy generation over the 4 minute mission than the Conventional Design.

2.4 Chapter Summary

In this chapter, I laid the foundation for a new approach to designing thermal management systems by designing a CCD algorithm for TMSs with an explicit focus on transient robustness and efficiency. Key takeaways are summarized as follows:

- The algorithm utilizes a first-principles thermal management system model that incorporates entropy generation rate in the state vector so that system efficiency, both steady-state and transient, can be accounted for in the design optimization.
- The use of parameterized nonlinear and linearized first principles models, as opposed to empirical models, enables optimization of meaningful, albeit lumped, design variables.

- The results suggest that using the plant design to optimize nominal efficiency and the feedback controller to maintain operation close to the optimized nominal value can provide a path toward minimizing the adverse effect of transient performance on efficiency.
- The optimization case study illustrates the co-designed systems' improved robustness to a set of pulsed heat load disturbances which has the potential to increase aircraft thermal endurance.

The main limitation of the CCD algorithm itself concerns the fidelity of several of the optimized plant variables, specifically with respect to the heat exchangers. Some of the decision variables are defined as lumped parameters, such as total lengths of tubing in the heat exchanger components, which map to high-level characteristics such as component mass. However, these lumped parameter variables fail to provide insight on how to design high-fidelity features of a given component. As discussed in Chapter 1, the specific design decisions made for a given component affect not only how that component interacts with the rest of the TMS, but also the achievable dynamics of the system level TMS and other integrated dynamic systems in a larger vehicle architecture. Therefore, component design optimization algorithms are needed to enable optimization of high-fidelity physical component geometries to meet desired system level performance.

3. COMPONENT LEVEL HEAT EXCHANGER DESIGN OPTIMIZATION

The next step in creating the proposed hierarchical CCD algorithm involves designing high-fidelity component level design optimization algorithms for components within a TMS. Here, the concept is illustrated by creating and demonstrating the utility of a dynamic component optimization algorithm for a shell and tube heat exchanger.

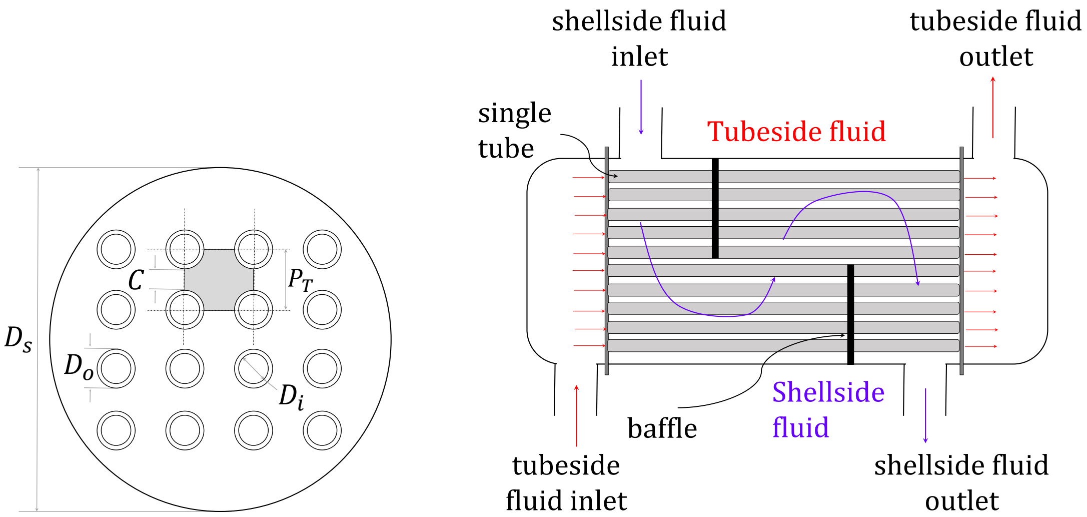
In the literature, design optimization of heat exchangers is often considered only in a static sense [64–73]. In other words, steady-state assumptions are used and the transient dynamics of a heat exchanger design, which ultimately dictate the speed of heat removal and thus the thermal capabilities of a larger integrated system, aren't typically included in the context of the optimization algorithm. In most cases, researchers derive an overall heat transfer coefficient for a given heat exchanger and use steady-state assumptions to impose a desired heat transfer rate as an operational constraint. Existing works usually consider optimizing one aspect of heat exchanger design/performance such as total component mass, operational cost [65, 71], and/or component efficiency [64,67,68]. In [66], the authors examine optimizing the total annual cost of heat exchangers based on different optimization algorithms. Their study compares the use of three separate optimization algorithms in the design of optimal economic characteristics of shell and tube mixed material heat exchangers. Tam et al. [65] derive a new optimization algorithm similar to a genetic algorithm, called the algorithm of changes, to optimize component sizing and power consumption. Li et al. [67] use the concept of entropy in their analysis to intelligently design aspects of heat exchangers for use in aircraft environmental control systems.

While the cited literature considers optimizing heat exchangers in some manner, it does not consider their transient behavior which dictates system level performance. Moreover, research that *does* consider transient modeling of heat exchangers does so largely in the context of synthesizing feedback controllers to guide the operation of already-designed heat exchangers [74–77]; in other words, no "transient" design optimization is considered.

In this chapter, I present a new design methodology for designing a heat exchanger that explicitly considers both static *and* transient performance characteristics. I specifically consider a shell and tube geometry and propose an optimization algorithm that utilizes a highly detailed static model, coupled with a reduced-order dynamic model, to achieve desired performance specifications. This combination of models with differing fidelity can be used to move beyond the conventional static approach to component design optimization. I use a static design model from the literature to embed design requirements and constraints on nominal heat removal rates in a global nonlinear optimization algorithm. I derive a dynamic model of single-pass shell and tube heat exchanger specifically to capture key transient characteristics of a given design. I linearize the dynamic model and leverage the optimization algorithm in part to push a heat exchanger design's eigenvalues further into the left half plane, which in turn helps improve the transient performance of a given design. I demonstrate the algorithm using a simulated case study.

3.1 Static Heat Exchanger Modeling

In this section I describe the high-fidelity, albeit static, model of a shell and tube heat exchanger which captures the key relationships between the heat exchanger design and resulting heat transfer coefficients. Nonetheless, the modeling concepts and analysis presented here can be generalized to other types of heat exchangers such as double pipe or plate-frame heat exchangers. Shell and tube heat exchangers are characterized by two separate paths of fluids, called tubeside fluid (also referred to as primary fluid) and shellside fluid (also referred to as secondary fluid). Figure 3.1 illustrates the functionality. In operation, the primary fluid enters the heat exchanger through an inlet plenum and is directed into numerous parallel tubes, and then ultimately exits through an outlet plenum. The secondary fluid enters the heat exchanger



Tube bundle end view

HEX frontal view

Figure 3.1. : Schematic of shell and tube heat exchanger.

through its inlet and flows inside the heat exchanger “shell” but external to the tubes carrying the primary fluid. The secondary fluid exchanges heat with the primary fluid in the tubes and is routed through the shell by equally-spaced baffles before exiting through the shellside fluid outlet. Here, I consider the hot fluid to be the tubeside fluid and the cold fluid to be the shellside fluid.

Several variables are of interest in designing and analyzing shell and tube heat exchangers. Figure 3.1 gives an end view of a heat exchanger tube bundle with a square pitch layout. For this geometry, the tube pitch P_T is defined as the distance between two adjacent tube centers, and the tube clearance C is defined as the distance between the outer edges of two adjacent tubes. Each individual tube has an inner diameter D_i and an outer diameter D_o. For a tube wall thickness w_t, tube outer diameter is related to tube inner diameter by $D_o = D_i + 2 \cdot w_t$. The inner diameter of the shell, which houses the entire tube bundle with N_T number of tubes, is denoted by D_s. The baffle spacing, or distance between adjacent baffles (see Figure 3.1 for reference), is denoted by B. The set of variables defined here fixes the geometry of a shell and tube heat exchanger. Using these variables, I can derive heat transfer coefficients for both the primary and secondary fluids. An overall heat transfer coefficient can then be calculated and used to approximate a steady-state rate of heat transfer for a given design. The following section details the process of computing the steady-state rate of heat transfer and outlet temperatures for each fluid.

The modeling concepts in this section are adapted from the work of Janna [78] and the American Society of Mechanical Engineers (ASME) design course number PD673 Design and Selection of Heat Exchangers. At steady-state, the rate of heat transfer achieved in a shell and tube heat exchanger is given by

$$\begin{aligned} \dot{Q}_{hx} &= (UA)_{hx} \Delta T_{lmtd} && (3.1a) \\ &= (\dot{m}c_p)_t (T_{in} - T_{out})_t && (3.1b) \\ &= (\dot{m}c_p)_s (T_{out} - T_{in})_s \quad , && (3.1c) \end{aligned}$$

where U_{hx} is an overall heat transfer coefficient for the heat exchanger, A_{hx} is the total heat transfer area, T is temperature, c_p is the specific heat capacitance of the respective fluid, subscripts *in* and *out* denote inlet and outlet, and subscripts s and t denote shellside fluid and tubeside fluid, respectively. The term ΔT_{lmtd} is a log mean temperature difference, which can be calculated for various types of flow. Here, I study parallel flow in which the tubeside and shellside fluids are flowing in the same direction through the heat exchanger; the log mean temperature difference is computed as

$$\Delta T_{lmtd} = \frac{(T_{in,t} - T_{in,s}) - (T_{out,t} - T_{out,s})}{\ln\left(\frac{T_{in,t} - T_{in,s}}{T_{out,t} - T_{out,s}}\right)} \quad . \tag{3.2}$$

Given values of mass flow rate and inlet temperature for both fluids, there are three unknowns in Eqn. (3.1): tubeside outlet temperature ($T_{t,out}$), shellside outlet temperature ($T_{s,out}$), and heat transfer rate ($\dot{Q}_{hx}$). Therefore, by computing the overall heat transfer coefficient U_{hx} and heat transfer area A_{hx}, I can solve for the unknowns.

The parameters defined in Figure 3.1 are used to compute the heat transfer coefficient and heat transfer area for a given design. The overall heat transfer coefficient is calculated using

$$\frac{1}{U_{hx}} = \frac{D_o}{h_t D_i} + \frac{1}{h_s} \ , \tag{3.3}$$

where h_t is the convective heat transfer coefficient for the primary fluid and h_s is the convective heat transfer coefficient for the secondary fluid. To evaluate h_s for the shell, a characteristic dimension, called an equivalent diameter D_e, is defined for the shell as

$$D_e = \frac{4P_T^2}{\pi D_o} - D_o. \tag{3.4}$$

Physically, this dimension is represented by the gray shaded region in Figure 3.1. The characteristic cross-sectional area for the shell is defined as

$$A_{c,s} = \frac{D_s C B}{P_T}. \tag{3.5}$$

The shell fluid velocity can then be computed as

$$V_s = \frac{\dot{m}_s}{\rho_s A_{c,s}}, \tag{3.6}$$

where ρ is the fluid density. From these computations, the shellside fluid Reynolds and Nusselt numbers are computed using

$$Re_s = \frac{V_s D_s}{\nu} \tag{3.7a}$$

$$Nu_s = \frac{h_s D_e}{k_s} = 0.36 Re^{0.55} \left(\frac{\nu_s}{\alpha_s}\right)^{1/3} \quad , \tag{3.7b}$$

where ν_s, α_s, and k_s are the shell fluid properties kinematic viscosity, thermal diffusivity, and thermal conductivity, respectively. Equation (3.7b) can then be solved for h_s.

Similar methods are used to compute a value for the primary fluid convection coefficient h_t. The characteristic diameter of the tube fluid is simply the tube inner diameter, D_i, and the characteristic area is given by

$$A_{c,t} = N_T \cdot \frac{\pi}{4} D_i^2, \tag{3.8}$$

where N_T represents the number of parallel tubes in the heat exchanger tube bundle. The tube fluid velocity is calculated according to

$$V_t = \dot{m}_t / \left(\rho_t A_{c,t}\right) \tag{3.9}$$

and the Reynolds is computed as

$$Re_t = V_t D_i / \nu_t. \tag{3.10}$$

For the tubeside Nusselt number, the modified Sieder-Tate correlation is used:

$$Nu_t = \begin{cases} 1.86\left(\frac{D_i Re_t \nu_t}{\alpha_t L_t}\right) & Re_t < 2200, 0.48 < \frac{\nu_t}{\alpha_t} < 16,700 \\ \frac{Re_t^{4/5}(\nu_t/\alpha_t)^{0.3}}{43.478} & Re_t > 10^4, 0.7 < \frac{\nu_t}{\alpha_t} < 160, \frac{L_t}{D_i} > 60 \end{cases}. \tag{3.11}$$

The tubeside convection coefficient is then computed via

$$h_t = Nu_t k_t / D_i. \tag{3.12}$$

The convection coefficients are used to find U_{hx} and the heat transfer area is taken as

$$A_{hx} = N_T \pi D_o L_t. \tag{3.13}$$

Equation (3.1) can then be used to solve for $\dot{Q}_{hx}$ and the heat exchanger outlet temperatures $T_{out,t}$ and $T_{out,s}$. To solve for these variables, an iterative process is needed. At the initial iteration, fluid properties ρ, c_p, ν, α, and k are computed according to the heat exchanger inlet temperatures. The process is repeated iteratively, with fluid properties evaluated each time at the average of the inlet temperatures and the previous iteration's outlet temperatures, until each outlet temperature converges to within a defined tolerance.

In the next section, I present a generalized *dynamic* model of a shell and tube heat exchanger. Heat transfer coefficients are computed via the same methods described in this subsection; however, the dynamic model accounts for transients ignored in the static calculations presented here.

3.2 Dynamic Heat Exchanger Modeling

Here I derive a reduced-order dynamic shell and tube heat exchanger model and present a case study to illustrate the utility of the formulation in identifying trade offs between the static design of the component and its transient performance.

3.2.1 Model Derivation

I begin by considering a heat exchanger with n equal-length control volumes and $n-1$ baffles. Each volume is separated by adjacent baffles or by a baffle and a tube bundle end plate as shown in Figure 3.2a.

To model the heat exchanger dynamically [74, 76], I treat each control volume as a single tube of length

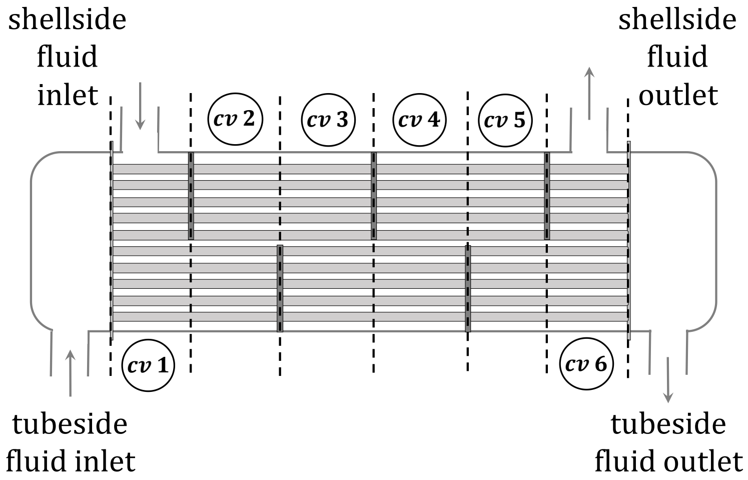
$$L_{cv} = \frac{L_t N_t}{n}, \tag{3.14}$$

where $n = 6$ is the number of control volumes. In other words, each control volume has $1/n^{th}$ of the total length of tubing in the entire heat exchanger. Primary (tubeside) fluid flows internal to each control volume and secondary (shellside) fluid flow is routed over the tubes by the baffles as depicted in Figure 3.1.

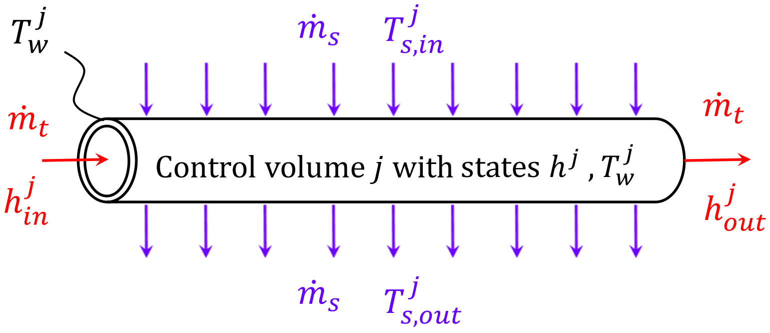
Each control volume j has two dynamic states: primary fluid enthalpy (h^j) and tube wall temperature (T_w^j). Writing conservation of energy on the primary fluid inside the tube yields

$$\frac{d}{dt}\left(mu^j\right) = \dot{m}_t\left(h_{in}^j - h_{out}^j\right) + \alpha_t^j A_{s,t}\left(T_w^j - T_t^j\right) \quad , \tag{3.15}$$

where α_t is the tubeside heat transfer coefficient between the tube wall and the primary working fluid, T_w^j is the wall temperature of control volume j, and $A_{s,t}$ is the heat transfer surface area between the wall and the primary working fluid. The primary fluid temperature T_t^j in each control volume is easily computed as a function



(a) Control volume discretization.



(b) Tube control volume.

Figure 3.2. : Shell and tube heat exchanger dynamic modeling schematics.

of the control volume's primary fluid enthalpy and pressure. I can write the mass as $m = \rho V$ and use the thermodynamic relationship $u = h - P/\rho$ to write

$$\frac{d}{dt}\left(mu^j\right) = \rho V \frac{dh^j}{dt} - V \frac{dP^j}{dt}. \tag{3.16}$$

Neglecting pressure dynamics and assuming a known value of fluid pressure, a common assumption for incompressible substances such as water, allows us to write Eqn. (3.15) as

$$\rho V \frac{dh^j}{dt} = \dot{m}_t \left(h_{in}^j - h_{out}^j \right) + \alpha_t^j A_s \left(T_w^j - T_t^j \right) \quad . \tag{3.17}$$

The volume of primary fluid in each control volume is calculated as $V = \frac{\pi}{4} D_i^2 L_{cv}$. Writing conservation of energy for the tube wall in each control volume gives

$$\left(mc_p \right)_w \frac{dT_w^j}{dt} = \alpha_t^j A_s \left(T_t^j - T_w^j \right) + \dot{Q}_{s \to w}^j \quad , \tag{3.18}$$

where $\dot{Q}_{s \to w}^j$ is the rate of heat transfer between the tube wall of control volume j and the secondary fluid (shellside) flowing in volume j. This rate of heat transfer is calculated using an effectiveness-NTU method as

$$\dot{Q}_{s \to w}^j = \dot{m}_s c_{p,s} \left[T_{s,in}^j - \left(T_w^j + \left(T_{s,in}^j - T_w^j \right) e^{\frac{-\alpha_s^j A_{s,s}}{\dot{m}_s c_{p,s}}} \right) \right] \quad . \tag{3.19}$$

To ensure continuity between adjacent control volumes within the heat exchanger, I take the secondary fluid inlet temperature for volume j ($T_{s,in}^j$) to be equal to the secondary fluid outlet temperature for volume $j-1$ ($T_{s,out}^{j-1}$). This captures the dynamics of the shell fluid being routed through the shell by the baffles while absorbing heat as it flows over the tubes. In Eqn. (3.18), the mass of the tube wall is computed according to

$$m_w = \rho_w \frac{\pi}{4} \left(D_o^2 - D_i^2 \right) L_{cv}. \tag{3.20}$$

The heat transfer coefficients α_t and α_s are computed via the same methods presented in Section II for the steady-state shell and tube heat exchanger calculations.

Finally, I can write Eqn. (3.17) and Eqn. (3.18) for each of the n volumes, yielding n state equations and a dynamic state vector

$$x = [h_1,\ h_2,\ \dots\ h_n,\ T_{w,1},\ T_{w,2},\ \dots\ T_{w,n}]. \tag{3.21}$$

The input vector for the model is $u = [\dot{m}_t,\ h_{in},\ T_{s,in},\ \dot{m}_s]^T$. The nonlinear model derived here allows us to capture the relevant dynamics of a shell and tube heat exchanger in a computationally-inexpensive manner.

In order to use this model to impose constraints on the speed of heat rejection in an optimization algorithm, I can linearize the model and apply constraints to the eigenvalues of the linearized model. To linearize, I first write the state equations in matrix-vector form as $Z_1 \frac{dx}{dt} = Z_2$. With the model's state and input vectors defined above, I can express the linearized model as

$$Z_1 \dot{x} = \tilde{A}\Delta x + \tilde{B}\Delta u \ ,$$

$$\tilde{A} = \begin{bmatrix} \frac{\partial (Z_2)_1}{\partial (x)_1} & \dots & \frac{\partial (Z_2)_1}{\partial (x)_{12}} \\ \vdots & \ddots & \vdots \\ \frac{\partial (Z_2)_{12}}{\partial (x)_1} & \dots & \frac{\partial (Z_2)_{12}}{\partial (x)_{12}} \end{bmatrix}, \quad \tilde{B} = \begin{bmatrix} \frac{\partial (Z_2)_1}{\partial (u)_1} & \dots & \frac{\partial (Z_2)_1}{\partial (u)_4} \\ \vdots & \ddots & \vdots \\ \frac{\partial (Z_2)_{12}}{\partial (u)_1} & \dots & \frac{\partial (Z_2)_{12}}{\partial (u)_4} \end{bmatrix} , \tag{3.22}$$

where Δ terms are perturbations from the nominal (equilibrium) operating point (x^e, u^e). The final linear model is obtained by manipulating Eqn. (3.22) to yield Eqn. (3.23):

$$\dot{x} = \bar{A}\Delta x + \bar{B}\Delta u,\ \ \bar{A} = Z_1^{-1}\tilde{A},\ \ \bar{B} = Z_1^{-1}\tilde{B} \ . \tag{3.23}$$

3.2.2 Model Simulations

In this section, I present simulation results and an eigenvalue analysis to show how different design parameters, such as number of tubes or tube wall thickness, affect the transient characteristics of a given system design. First, however, I wish to illustrate that the linearized model dynamics closely match those of the full nonlinear model.

To test the accuracy of the linear model, I apply step pulses to each input channel 10% above the nominal input values. Figure 3.3 gives a comparison of the linear and nonlinear model dynamics for control volumes 2, 4, and 6 of each model and Figure 3.4 gives the input signals applied for both cases. The results indicate that, while there are slight offsets in the steady-state values between the nonlinear and linear models after a given step input, the linear model captures the transients of the full nonlinear model quite well. Most notably, the time constants of each state are captured accurately by the linear model. This implies the linear model can be used to improve notions of transient performance within a heat exchanger optimization algorithm.

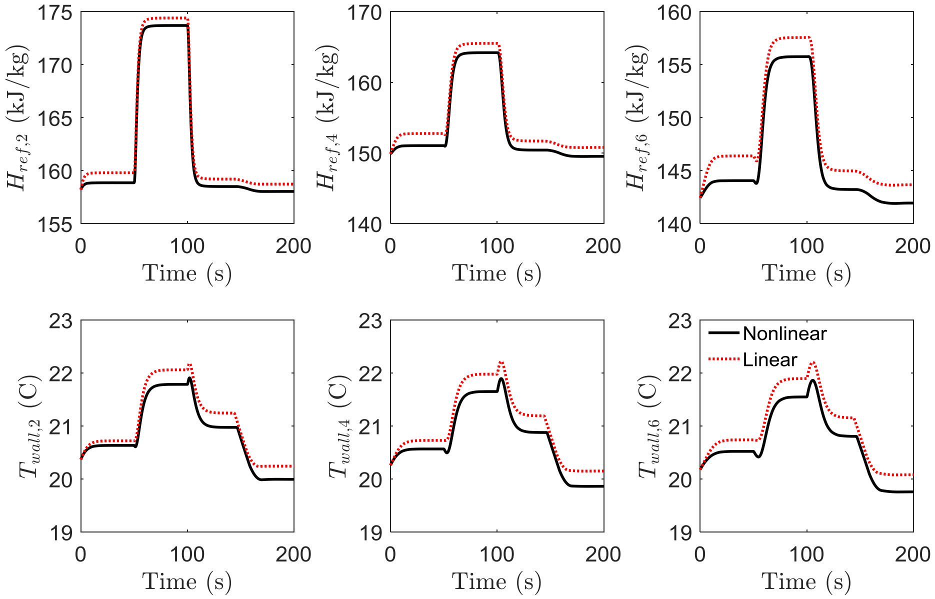


Figure 3.3. : A comparison of nonlinear and linear simulation results for the states of the dynamic model: responses for control volumes 2, 4, and 6.

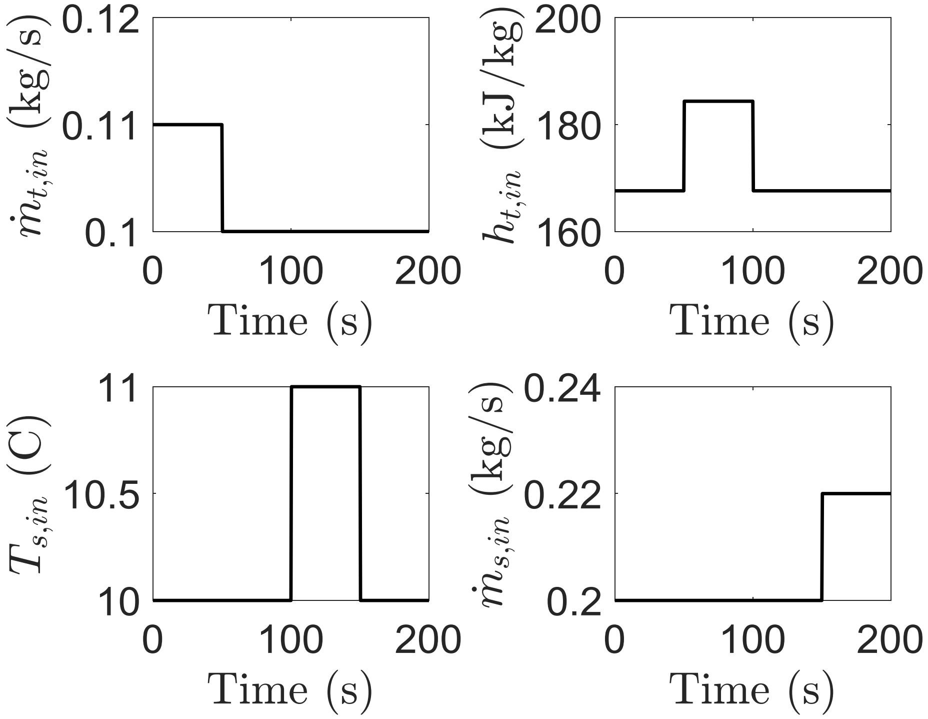


Figure 3.4. : A comparison of nonlinear and linear simulation results for the states of the dynamic model: step inputs to each input channel.

To illustrate how different design parameters affect the transient characteristics of a given system design, I compare the dynamics of a baseline plant design with the dynamics of an augmented plant design. As the baseline case, I simulate the dynamics of a shell and tube heat exchanger with $N_T = 24$ tubes, each of length $L_{tube} = 1.2192$ m, and a wall thickness $w_{t,tube} = 0.0008$ m. As a comparison case, I hold all other design parameters constant, but use $N_T = 12$ tubes, $L_{tube} = 0.6096$ m, and $w_{t,tube} = 0.0016$ m. The parameters for each case are summarized in Table 3.1. For each case, I apply a step input in the primary fluid flow rate $\dot{m}_t = 0.12$ kg/s (increased from the steady-state value of 0.10 kg/s), while using the steady-state values of the other input variables: $\dot{m}_s = 0.2$ kg/s, $T_{s,in} = 10°\text{C}$, and $h_{in} = 188.5$ kJ/kg.

Table 3.1. : Parameters for dynamic model simulation.

Case	$\mathbf{N_T}$	$\mathbf{L_{tube}}$ **(m)**	$\mathbf{w_t}$ **(m)**
Baseline	24	1.2192	0.0008
Augmented	12	0.6096	0.0016

Figure 3.5 shows a comparison of the step responses of six of the dynamic states for each design simulated with the nonlinear dynamic model - the states shown are enthalpy and wall temperature for control volumes two, four, and six. In the figure, δ on the vertical axis labels signifies that each response is a change from its nominal, or steady-state, value. Additionally, dashed vertical lines represent the equivalent first-order time constant associated with each dynamic mode. Figure 3.6 shows a comparison of the pole plots for the linearization of each respective heat exchanger design.

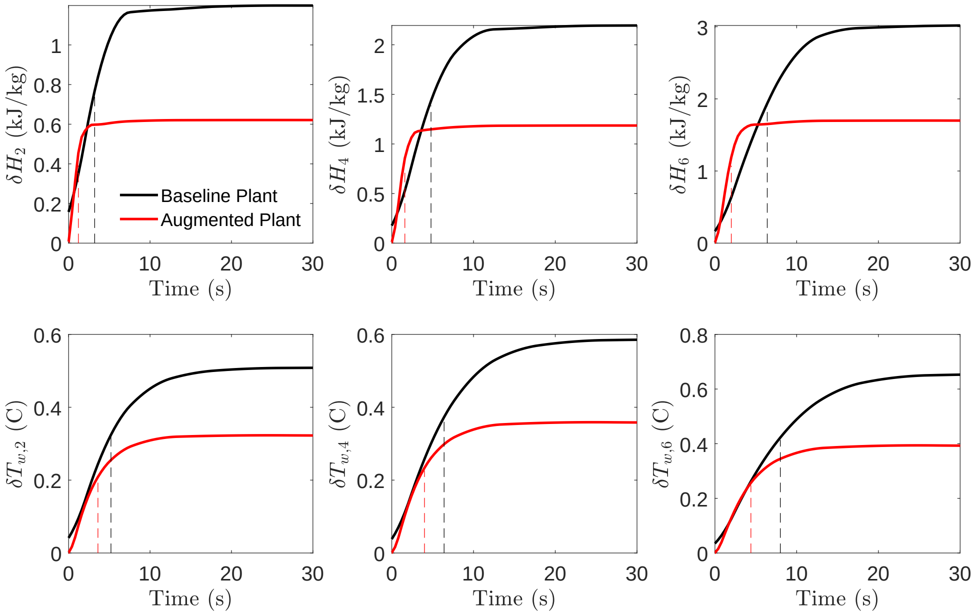


Figure 3.5. : Step responses for 6 of 12 dynamic states of each heat exchanger design.

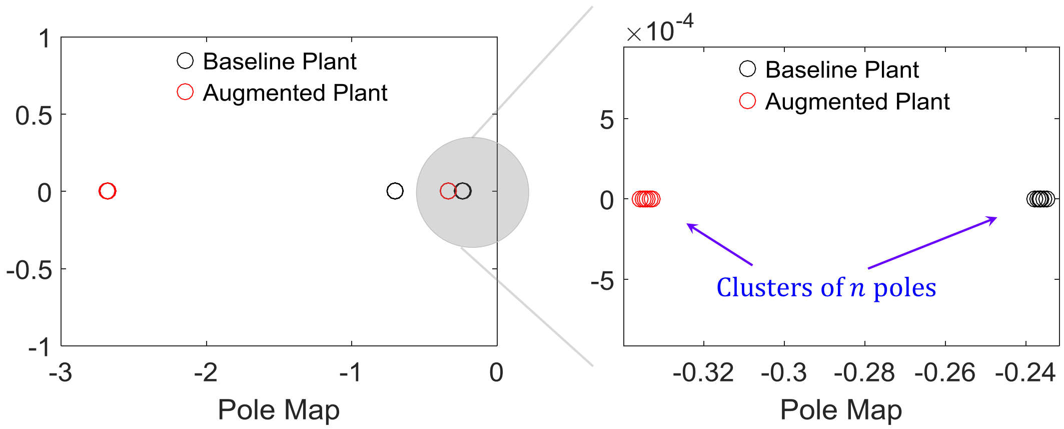


Figure 3.6. : Pole plots for each heat exchanger design.

From Figure 3.5 one can see that decreasing the length and number of tubes while increasing the wall thickness of the tubes speeds up the dynamics of each state. Intuitively, this makes sense, given that both the overall mass of the tubes and the volume of primary fluid decrease. These dynamic trends are also evident from the pole plot shown in Figure 3.6 in which there are two distinct pole clusters for each heat exchanger design. The faster cluster of poles moves further into the left half plane in the case of the augmented plant due to the decrease in tube length and the number of tubes; moreover, the slower cluster of poles also moves further into the left half plane due the overall decrease in tube bundle mass. The simulation results verify that the proposed dynamic model does capture the effect that changes in the design parameters have on the eigenvalues of the linearized model.

Remark 6. While the practice of evaluating pole locations for a linear dynamic model is not difficult, it does have important meaning in the context of heat exchanger design. By incorporating the dynamic model into an optimization algorithm for a shell and tube heat exchanger, I can effectively design the heat exchanger to move slower poles to the left in the complex plane, thereby improving the transient performance of the component and permitting greater thermal capabilities for an integrated thermal management system.

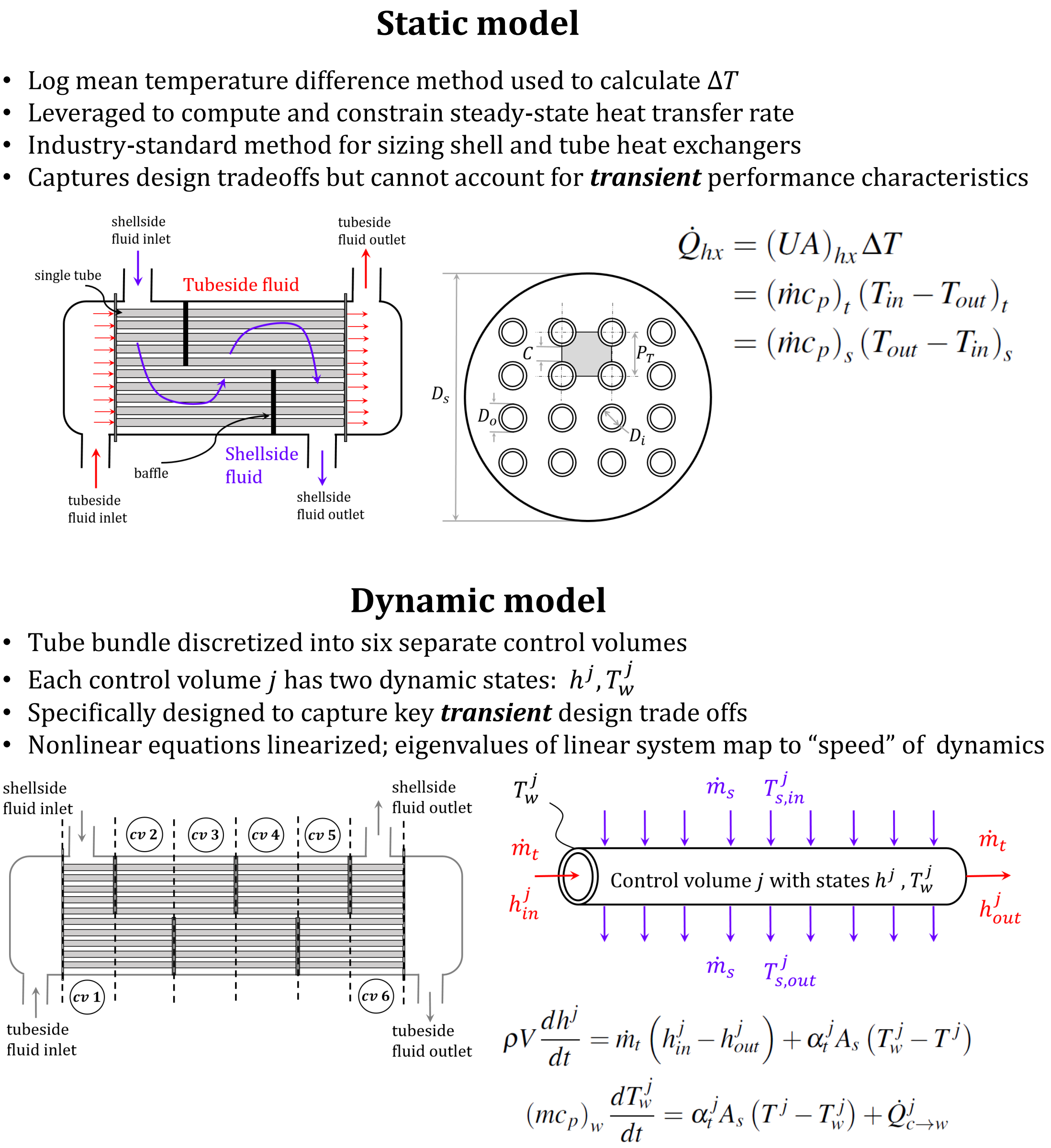
Until this point, I have presented both a static shell and tube heat exchanger model (used as an industry-standard for component sizing) and a reduced-order dynamic model for predicting the heat exchanger's transient behavior. The characteristics of both models are summarized in Figure 3.7. The static model presented earlier facilitates the design of a heat exchanger with desired static performance specifications such as a target steady-state rate of heat removal. It is not, however, useful for designing a system to meet transient performance specifications. In the next section I demonstrate how I integrate these models in a single optimization problem for achieving dynamic design optimization.

3.3 Dynamic Shell and Tube Heat Exchanger Optimization

3.3.1 Algorithm Design

In this section, I present an optimization algorithm designed to leverage heat exchanger design parameters to minimize a combination of heat exchanger footprint, mass, and system speed while guaranteeing a desired rate of steady-state heat removal. The footprint and mass of a component are common design metrics for a range of applications, including aerospace thermal management wherein mass, in particular, has a significant effect on fuel consumption.

The proposed shell and tube heat exchanger optimization algorithm is defined in Eqn. (3.25). In the objective function, γ_1, γ_2, and γ_3 are slack variables that enable minimization of a weighted combination of footprint, mass, and transient performance. These values can be chosen to prioritize design goals as needed. I take the term λ_{max} to be the maximum, or slowest, eigenvalue of the linearized model given a set of design parameters. Therefore the algorithm can be leveraged to place a penalty on pushing the slowest eigenvalue further left in the complex plane, thereby improving the transient performance of the heat exchanger. Here, I take the footprint of the heat exchanger to be $FP = L_{tube} \cdot D_s$. I define the total mass of the heat exchanger as a summation of the tube bundle mass and the shell mass, computed as



Static model

- Log mean temperature difference method used to calculate ΔT
- Leveraged to compute and constrain steady-state heat transfer rate
- Industry-standard method for sizing shell and tube heat exchangers
- Captures design tradeoffs but cannot account for ***transient*** performance characteristics

$$\dot{Q}_{hx} = (UA)_{hx}\Delta T$$
$$= (\dot{m}c_p)_t (T_{in} - T_{out})_t$$
$$= (\dot{m}c_p)_s (T_{out} - T_{in})_s$$

Dynamic model

- Tube bundle discretized into six separate control volumes
- Each control volume j has two dynamic states: h^j, T_w^j
- Specifically designed to capture key ***transient*** design trade offs
- Nonlinear equations linearized; eigenvalues of linear system map to “speed” of dynamics

$$\rho V \frac{dh^j}{dt} = \dot{m}_t \left(h_{in}^j - h_{out}^j\right) + \alpha_t^j A_s \left(T_w^j - T^j\right)$$

$$(mc_p)_w \frac{dT_w^j}{dt} = \alpha_t^j A_s \left(T^j - T_w^j\right) + \dot{Q}_{c\to w}^j$$

Figure 3.7. : Summary of static versus dynamic model.

$$m_{tot} = m_{tubes} + m_{shell} = \frac{\pi}{4} L_{tube}\rho \left[N_T \left(D_o^2 - D_i^2\right) + \left(D_{o,s}^2 - D_s^2\right)\right] \quad , \tag{3.24}$$

where the shell outer diameter is computed as $D_{o,s} = D_s + 2w_s$ and the shell wall thickness w_s is assumed to be twice the tube wall thickness.

$$\begin{aligned} \underset{x_d}{\text{minimize}} \quad & J\left(x_d\right) = \gamma_1 \cdot FP + \gamma_2 m_{tot} + \gamma_3 \cdot \left(\frac{1}{\lambda_{max}}\right)^2 \\ \text{subject to} \quad & x_{d,min} \leq x_d \leq x_{d,max}, \\ & \dot{Q}_{hx} \geq \dot{Q}_{des}, \\ & B \leq D_s \leq 5B, \\ & T_{tube,out} \geq T_{shell,out}, \\ & A_{c,tubes} \leq A_{c,shell}, \\ & C, D_e \geq \epsilon \end{aligned} \tag{3.25}$$

The decision variable vector for the algorithm is

$$x_d = \left[P_T,\ \ D_o,\ \ L_{tube},\ \ N_T,\ \ D_s,\ \ \dot{m}_t,\ \ \dot{m}_s\right]^T \ , \tag{3.26}$$

where P_T, D_o, L_{tube}, N_T, and D_s are previously defined design parameters and the mass flow rates $\dot{m}_t$ and $\dot{m}_s$ are operational input variables. Note that when integrating the system level CCD algorithm and the component level algorithms (as in Chapter 4), the mass flow rates will be inputs to the component level optimization. In this section, however, they will be treated as decision variables to illustrate the utility of this formulation.

I define several constraints related to the geometry of the heat exchanger design. First, each design variable is both lower bounded ($x_{d,min}$) and upper bounded ($x_{d,max}$). At steady-state, the heat removal rate $\dot{Q}_{hx}$, as calculated with the steady-state methods described in Section 2.2, is required to be greater than a desired rate $\dot{Q}_{des}$. In accordance with typical design practices, the shell diameter (D_s) should be larger than the baffle spacing but no larger than five times the baffle spacing. The shell diameter is further constrained to ensure it is large enough to accommodate all N_T tubes. Additionally, the tube clearance C and shell equivalent diameter D_e, both functions of various decision variables, are required to be positive values. I also constrain the tubeside fluid outlet temperature to be greater than the shellside fluid outlet temperature to prevent crossover during system operation.

In the following subsection, I demonstrate the optimization algorithm with a simulated case study.

3.3.2 Case Study: Static Optimization vs. Proposed Dynamic Optimization

Here I present a case study illustrating the utility of the optimization algorithm detailed above. As a baseline case, hereon referred to as the Static Case, I optimize the design of a shell and tube heat exchanger considering only static design metrics. In other words, I set γ_3 in Eqn. (3.25) to zero and use values of $\gamma_1 = 100$ and $\gamma_2 = 0.1$ to weight footprint and mass, respectively. The full constraint list is applied and the decision variable vector is the same as in Eqn. (3.26). Fundamentally, the Static Case represents a conventional method of optimizing heat exchanger performance in the literature. The device is optimized with respect to sizing metrics using static design principles while being constrained to achieve a desired steady-state rate of heat removal.

For my proposed algorithm, referred to as the Dynamic Case, I solve the optimization problem presented in Eqn. (3.25) with an added penalty on dynamic (transient) performance. In the Dynamic Case, I use the same slack variable weightings $\gamma_1 = 100$ and $\gamma_2 = 0.1$ for footprint and mass in the objective function. Additionally, I use a penalty $\gamma_3 = 100$ to weight the transient performance contribution to the overall objective function. In both cases, I employ a multistart global nonlinear optimization algorithm using the OPTI toolbox [62]; the optimal designs for each case, along with upper and lower bounds on the decision variables, are summarized in Table 3.2.

Table 3.2. : Results for optimization case study.

			Optimal Design	
Variable	**Low Bound**	**High Bound**	**Static Case**	**Dynamic Case**
P_T (m)	0.0127	0.0381	0.013	0.016
D_o (m)	0.0063	0.0254	0.010	0.016
L_{tube} (m)	0.1524	3.658	0.155	0.746
N_T (–)	1	100	75	1
D_s (m)	0.0257	3.018	0.107	0.124
$\dot{m}_t$ (kg/s)	0.1125	0.188	0.188	0.188
$\dot{m}_s$ (kg/s)	0.225	0.375	0.375	0.318
Footprint			**0.017 m^2**	**0.093 m^2**
Total mass			**4.10 kg**	**9.01 kg**
$\tau_{\mathbf{t,out}}$			**1.03 s**	**0.276 s**

The results in Table 3.2 highlight a few important trade offs. In the static case, it can be seen that the optimal design features tubes with a tube length of $L_{tube} = 0.155$ meters, compared to a much longer tube length of 0.746 meters in the dynamic case. Moreover, the static case features 75 tubes as opposed to a single tube in the dynamic case. Physically, the length dimension dominates the 2D footprint of the heat exchanger. Therefore, the static case seeks to use a higher number of short-length tubes to get the requisite heat transfer area to meet the 5kW heat load requirement. Additionally, because the tubes are short in the Static Case, the shell diameter is permitted to be smaller and the mass of the shell is much lighter than in the Dynamic Case.

On the other hand, using the higher number of tubes decreases the mass flux (flow rate divided by cross-sectional area) for the primary fluid in the Static Case;

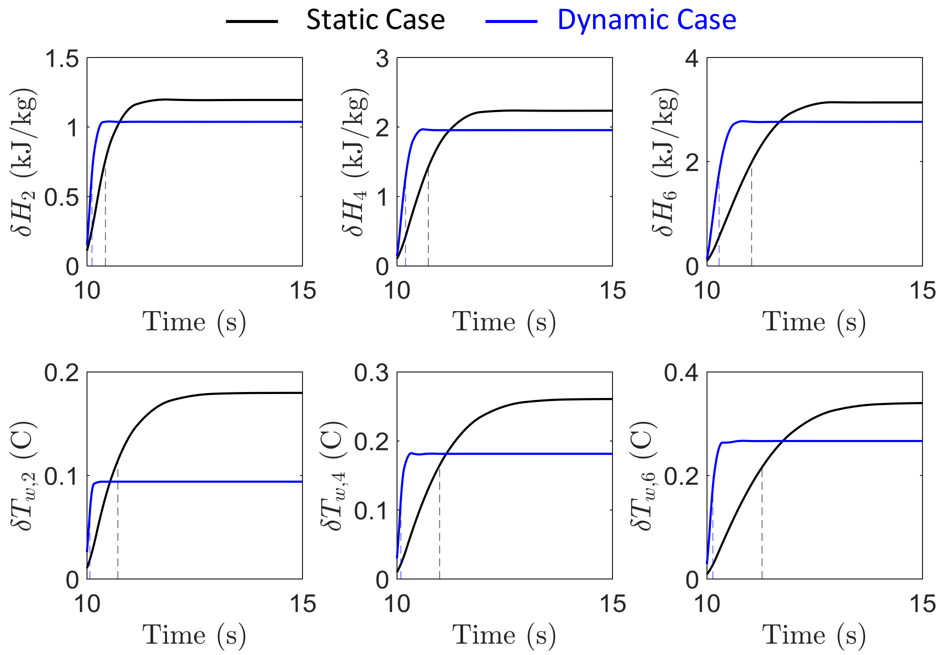
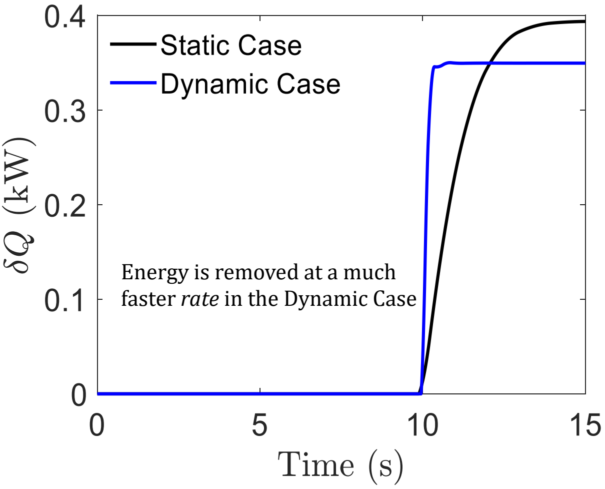


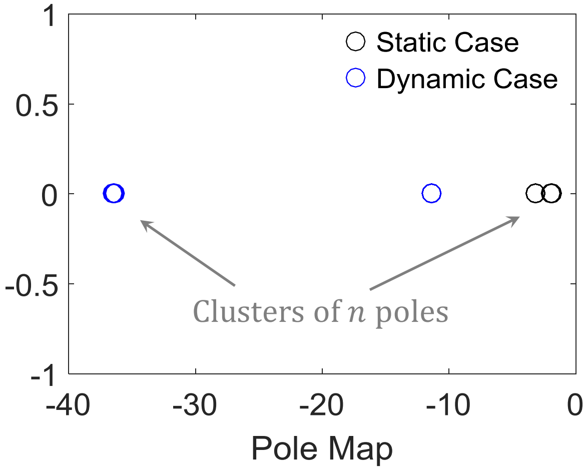
Figure 3.8. : Step responses of states for control volumes 2, 4, and 6.

this, in turn, decreases the speed of heat exchanger's heat transfer dynamics. In the Dynamic Case, the optimal tubeside mass flow rate is the same as in the static case, but because the optimal number of tubes is only 1, this results in a much higher mass flux and ultimately a faster dynamic response. In the dynamic case, the time constant for the primary (tubeside) outlet fluid is just $\tau_{t,out} = 0.276$ seconds compared to a value of $\tau_{t,out} = 1.03$ seconds in the Static Case.

To further illustrate the difference in transient performance between the two cases, I simulate the step responses of each optimal component design. For each case, I apply a step input at $t = 10$ seconds of 30% from the steady-state value in the primary fluid flow rate $\dot{m}_t$ while using the steady-state values of the other input variables. Figure 3.8 shows a comparison of the step responses for six of the dynamic states for each design (states for control volumes two, four, and six in Figure 3.2a). Figure 3.9a shows the step response of the rate of heat transferred from the tube walls to the secondary fluid. Figure 3.9b shows a comparison of the pole plots for the linearized dynamic



(a) Heat transfer rate



(b) Pole plot.

Figure 3.9. : Dynamic comparison of optimal designs: (a) shows the heat transfer rate, with respect to nominal, out of the primary fluid and (b) gives the corresponding pole plots.

model corresponding to each optimal design. In Figure 3.8-3.9a, δ on the vertical axis labels signifies that each response is a change from its nominal, or steady-state, value. Additionally, dashed vertical lines in Figure 3.8 represent the time constant of each response.

The results in Table 3.2 and Figure 3.9 illustrate the utility of including dynamic performance considerations into the design of thermal management components. Figure 3.9b illustrates the algorithm's ability to move the poles of the system further left in the complex plane, which maps to faster dynamics (Figure 3.8). In the Dynamic Case, the primary fluid absorbs the requisite amount of heat in 0.276 seconds, almost four times as fast as in the Static Case; Figure 3.9a illustrates the superior speed of the heat removal dynamics in the Dynamic Case. Fundamentally, this indicates energy is being pulled from the primary fluid much faster. This has important implications in dynamic thermal management, where increased electrification now requires dissipation of highly transient heat loads. Component mass and footprint were higher in the Dynamic Case, however, because of the weights used in the objective function. By further tuning the weights γ_1, γ_2, and γ_3 in the objective function, an intermediate balance could be achieved between static design goals and dynamic performance.

Remark *7.* The proposed approach further demonstrates how a combination of models with differing fidelity can be used to move beyond the conventional static approach to component design optimization. The static model was effective in capturing the complex nonlinear relationship between heat exchanger geometry and heat transfer coefficient, whereas the dynamic model used simplifying assumptions to predict transient performance in a computationally-tractable manner.

3.4 Chapter Summary

In this chapter, I presented a new methodology for designing a heat exchanger that explicitly considers both static *and* transient performance characteristics. I specifically considered a shell and tube heat exchanger geometry and proposed an optimization algorithm that utilizes both a highly detailed and nonlinear static model, coupled with a reduced-order dynamic model, to optimize both static and dynamic performance metrics. The proposed optimization problem relied on pole location as a proxy for dynamic response of the component, and the proposed cost function was structured to penalize poles closer to the origin of the complex plane. Through a simulated case study, I demonstrated how the proposed algorithm exploits the trade off between static design metrics, including mass and footprint, against the rate at which heat is removed from the system.

In the next chapter, I combine the CCD algorithm at the system level with component level algorithms such as the one presented here to create the first hierarchical control co-design (HCCD) architecture for fully optimizing the design and control of a transient TMS.

4. HIERCHICAL CCD (HCCD) ALGORITHM

In this chapter, I present the main contribution of this dissertation. Namely, I integrate the system level CCD algorithm from Chapter 2 and the heat exchanger component optimization from Chapter 3 to create a novel hierarchical control co-design (HCCD) optimization algorithm aimed at optimizing the design and control of a transient TMS. The decision variable vectors for each algorithm are amended to facilitate integration of the two levels. Therefore, the slightly modified optimization algorithms used at both the system and component levels in the HCCD architecture are presented in this chapter. Additionally, a static pump model and optimization algorithm are integrated at the component level to capture tradeoffs between pump mass and characteristics such as mass flow rate range and pump entropy generation rate.

4.1 Hierarchical CCD Algorithm Architecture

Here I describe the operational framework and architecture of the proposed hierarchical control co-design (HCCD) optimization algorithm. I note that while I am specifically optimizing TMSs in this dissertation, the proposed algorithm could be adapted to any complex integrated system. Therefore, the discussion in this section will cover the general functionality of the algorithm before the following sections discuss its application to dynamic thermal management.

My proposed algorithm consists of a system level (SL) and component level (CL) and runs as depicted in Fig. 4.1. First, various SL setpoints and parameters must be declared in order to execute initial component optimizations. Once an initial set of optimal component designs has been generated, the optimal component designs and target variables are passed back up to the SL optimization algorithm where a CCD algorithm is used to (1) optimize a full system design to connect the optimal components and (2) design an optimal feedback controller to guide the system's transient performance in a robust manner. The resulting set of optimal SL setpoints is then passed back down to the CL where a new set of optimal components is designed in parallel. The process repeats iteratively until all objective functions in the hierarchy converge to within a user-defined sensitivity tolerance $\Delta J^k \leq \epsilon$ where ΔJ^k is defined as in Eqn. (4.1)

$$\Delta J^k(x^k) = \left|\frac{\partial J^k}{\partial x_1^k}\right| \cdot |\Delta x_1^k| + \left|\frac{\partial J^k}{\partial x_2^k}\right| \cdot |\Delta x_2^k| + \ldots + \left|\frac{\partial J^k}{\partial x_n^k}\right| \cdot |\Delta x_n^k| \ , \tag{4.1}$$

and ΔJ^k represents a sensitivity in the k^{th} objective function due to the latest changes in the optimal design represented by the k^{th} set of decision variables. This criterion is used to (1) ensure convergence for variables which dominate changes in the objective function of a given algorithm and (2) avoid unnecessary iteration due to changes in design variables which do not largely affect a given objective function.

Figure 4.2 provides a more detailed graphic of the structure of each individual algorithm. The system level (SL) algorithm represents the top level of my hierarchical CCD algorithm. Here the system model is intentionally low-fidelity, or reduced-order, to promote computationally efficient modeling for use in synthesizing feedback controllers in the CCD algorithm. The SL CCD algorithm is itself a nested optimization algorithm as described in Chapter 2 in which the outer loop optimization is used to optimize a set of plant (system) design variables to minimize a weighted combination of system design properties, such as weight or mass, and steady-state performance objectives. Within each iteration of the outer loop plant optimization, an inner loop control optimization is solved to generate an optimal controller for the given plant design. Generally speaking, the inner loop can be structured to synthesize various types of state feedback controllers; here I specifically consider the H_∞ robust controller discussed in Chapter 2.

The SL outer plant loop is linked to the inner control loop via the inner loop's optimal cost. In Fig. 4.2, the term $\lambda_{n+1} g^* \left(x_p, x_u\right)$ is the contribution of the inner loop's optimal solution to the outer loop cost. Any plant design with poor potential for robust transient performance is then penalized at the plant design stage. The SL outer loop is similarly linked to each individual component level optimization by the term $h\left(f^*_{c_i}\right)$ for the i^{th} component being optimized. Therefore, components with poor designs are also captured at each iteration of the SL optimization. The optimized plant design variables are generally variables that are passed to the CL as setpoints or variables that are involved in connecting components with one another. For the thermal management system, these will include plant design variables such as mass flow rates of fluid pumped through the various components or lengths of pipe connecting each component.

At the component level (CL), the models are of higher fidelity to enable optimization of a detailed set of design variables that together will result in a component set capable of meeting specific setpoints optimized at the SL. For example, CL models in mechanical system design may take into account complex nonlinear effects such as Coulomb friction and viscous damping whereas the SL model might ignore these phenomena or lump them into a single term. The component level optimizations are independent of one another and can be run in parallel. Each has its own specific cost function and set of design variables. Once optimal components are determined, the component designs and relevant parameters are passed back up to the SL for the next iteration of the hierarchical optimization.

Model tuning is a key step in linking the two levels of the hierarchy. In other words, I must ensure that the reduced-order model at the SL accurately represents the characteristics of the higher-fidelity models at the CL. This is done by performing a model tuning analysis between an iteration of the CL optimizations and the subsequent iteration of the SL CCD optimization. Tuning parameters within the SL model are updated to ensure that improvements in component designs are accurately reflected by the reduced-order model at the SL. This process will be described in more detail in subsequent sections. By utilizing the model tuning within the hierarchical optimization, the CL provides a means of selecting high-fidelity physical component designs to meet system design and control objectives and the SL ensures that the optimal components can be connected and controlled in an optimal manner.

In the following sections, I describe the models used at each level of the hierarchy and introduce a model for the pumps at the component level.

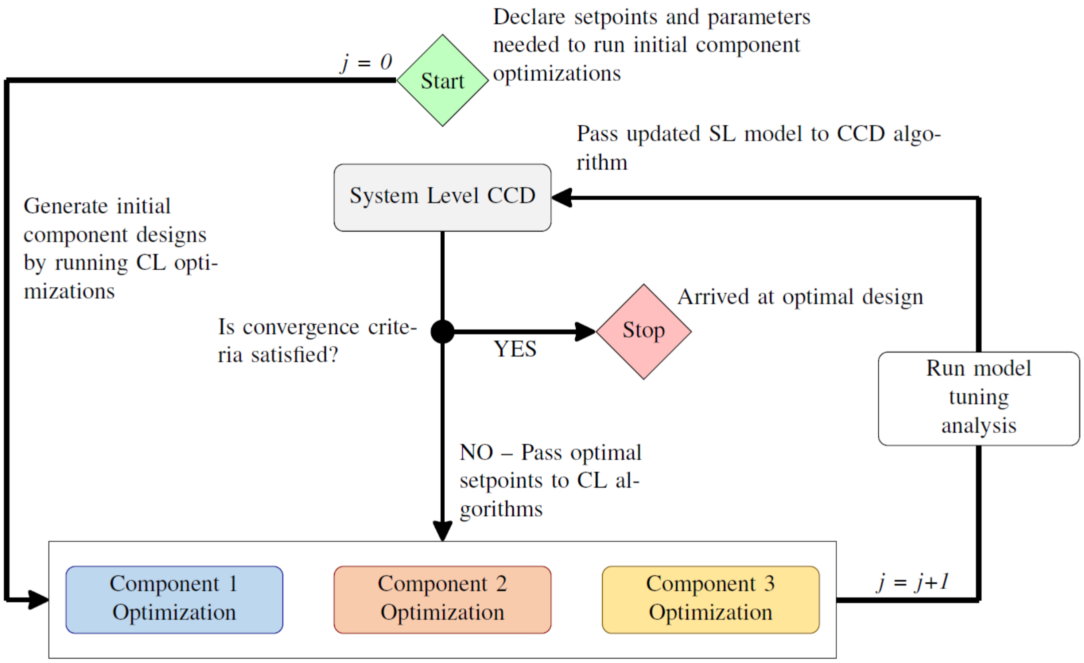

Declare setpoints and parameters needed to run initial component optimizations
j = 0
Start
Pass updated SL model to CCD algorithm
System Level CCD
Generate initial component designs by running CL optimizations
Arrived at optimal design
Is convergence criteria satisfied?
YES
Stop
Run model tuning analysis
NO – Pass optimal setpoints to CL algorithms
j = j+1
Component 1 Optimization
Component 2 Optimization
Component 3 Optimization

Figure 4.1. : Flowchart depicting operation of hierarchical co-design algorithm.

4.2 HCCD: System and Component Models

4.2.1 System Level TMS and Shell and Tube HX Models

The models used for the system level TMS and the component level shell and tube heat exchanger are taken from Chapters 2-3. At the system level, I use the low-fidelity model derived for the TMS in Chapter 2. For reference, a schematic

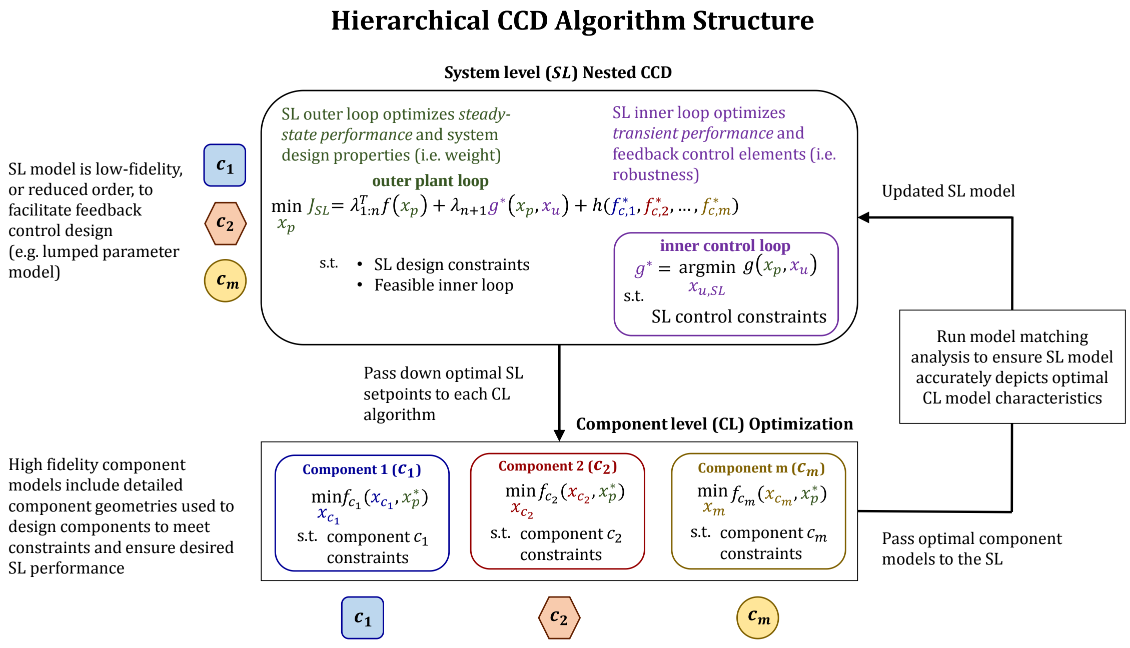

Hierarchical CCD Algorithm Structure
System level (SL) Nested CCD
SL outer loop optimizes steady-state performance and system design properties (i.e. weight)
SL inner loop optimizes transient performance and feedback control elements (i.e. robustness)
outer plant loop
$\min_{x_p} J_{SL} = \lambda_{1:n}^T f(x_p) + \lambda_{n+1} g^*(x_p, x_u) + h(f_{c,1}^*, f_{c,2}^*, \dots, f_{c,m}^*)$
s.t. • SL design constraints • Feasible inner loop
inner control loop
$g^* = \operatorname{argmin}_{x_{u,SL}} g(x_p, x_u)$
s.t. SL control constraints
SL model is low-fidelity, or reduced order, to facilitate feedback control design (e.g. lumped parameter model)
c_1
c_2
c_m
Updated SL model
Run model matching analysis to ensure SL model accurately depicts optimal CL model characteristics
Pass down optimal SL setpoints to each CL algorithm
Component level (CL) Optimization
High fidelity component models include detailed component geometries used to design components to meet constraints and ensure desired SL performance
Component 1 (c_1)
$\min_{x_{c_1}} f_{c_1}(x_{c_1}, x_p^*)$
s.t. component c_1 constraints
Component 2 (c_2)
$\min_{x_{c_2}} f_{c_2}(x_{c_2}, x_p^*)$
s.t. component c_2 constraints
Component m (c_m)
$\min_{x_m} f_{c_m}(x_{c_m}, x_p^*)$
s.t. component c_m constraints
Pass optimal component models to the SL
c_1
c_2
c_m

Figure 4.2. : Schematic of the optimization structure of each level in hierarchical control co-design.

of the model is given in Figure 4.3. Recall that this model captures the dominant behavior and transients of the system in a computationally-efficient manner. More specifically, the model contains six dynamic states corresponding to the first law dynamics of the TMS: M_t, T_t, T_r, T_a, $T_{w,r}$, and $T_{w,a}$. Additionally, I use the second law of thermodynamics to develop an expression for the entropy generation rate of the system and then append the tracking state ζ to the state vector for use in the CCD algorithm.

The SL model, albeit physics-based, was intentionally designed with lumped-parameter techniques. For example, I intentionally treated the heat rejection component (shell and tube heat exchanger) in Fig. 4.3 as a single long tube with two lumped states to describe its dynamics. By doing so, the SL model can more easily be used in the HCCD algorithm for designing feedback controllers in the inner loop of the SL CCD algorithm. In practice, however, as discussed in Chapter 3, shell and tube heat exchanger dynamics are extremely complex. Therefore, at the component

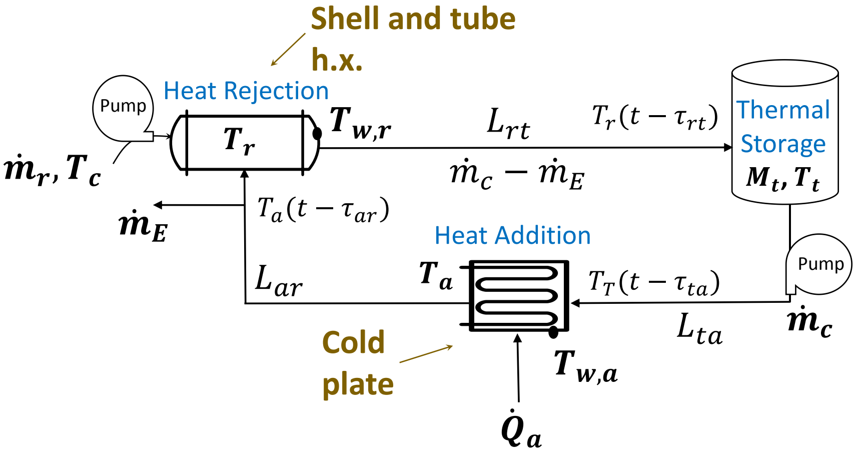


Figure 4.3. : System diagram for the notional thermal management system. Note that the shell and tube heat exchanger is equivalent to the heat rejection component.

level, I can use the shell and tube heat exchanger models from Chapter 3 to optimize the detailed physical design of the shell and tube heat exchanger. More specifically, I use the highly-detailed static model of the heat exchanger, based on industry standard modeling concepts, to capture the key relationships between the heat exchanger geometry and resulting heat transfer coefficients. I then have my dynamic model of the heat exchanger's tubeside dynamics to capture the tradeoffs between various design decisions and the *transient performance* of the heat exchanger.

4.2.2 Component Level Pump Modeling

I now also integrate component models for both pumps in the TMS into the hierarchy. To model the pumps in the TMS, I use a static model that maps a key design variable, impeller diameter, to relevant specifications such as overall pump mass, achievable mass flow rate, and pump entropy generation rate.

Given a baseline, or reference, pump design with an impeller diameter D_{ref} and maximum mass flow rate $\bar{\dot{m}}_{ref}$, I can compute a maximum achievable mass flow rate associated with any candidate pump ($\bar{\dot{m}}_{cand}$) from the pump affinity law [79] in Eqn.

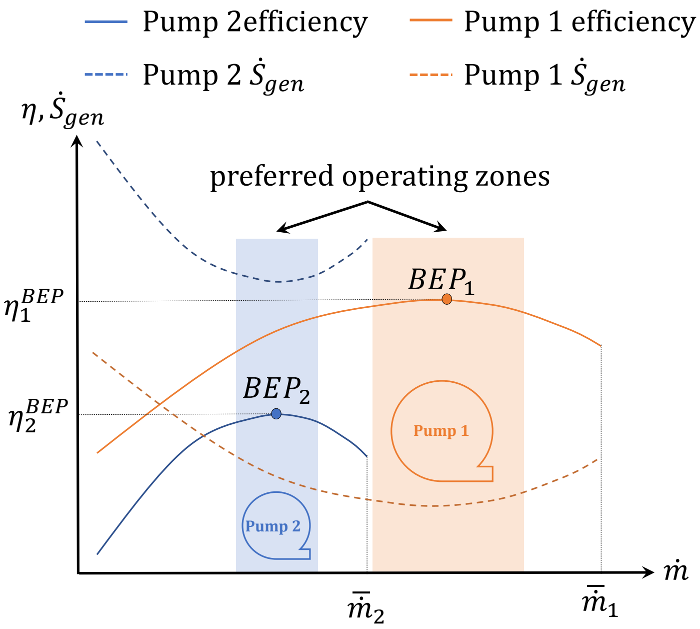


Figure 4.4. : Illustration of pump characteristics with varying impeller diameters. A larger pump can achieve higher mass flow rates with better efficiency, but comes with a higher physical mass.

(4.2a) where D_{cand} is the impeller diameter required to generate $\bar{\dot{m}}_{cand}$. Assuming a disc-shaped impeller, the mass of the candidate pump can then be computed as shown in Eqn. (4.2b) where w_{imp} is the width of the impeller and ρ is the density of the material from which the impeller is constructed. Finally, I also model efficiency and entropy generation curves for each candidate pump. Figure 4.4 illustrates the effect of mass flow rate on the efficiency curve for two separate size pumps.

$$\frac{\bar{\dot{m}}_{cand}}{\bar{\dot{m}}_{ref}} = \left(\frac{D_{cand}}{D_{ref}}\right)^3 \quad , \tag{4.2a}$$

$$m_{cand} = \rho \frac{\pi}{4} w_{imp} D_{cand}^2 \quad . \tag{4.2b}$$

In Fig. 4.4, the larger pump (Pump 1) can command a higher mass flow rate $(\bar{\dot{m}}_1)$ than the smaller pump (Pump 2). Pump 1 also has a higher efficiency, η_1, throughout its operating range. However, Pump 1 requires a larger impeller diameter to achieve the higher flow rates and thus comes with additional mass when compared to Pump 2. I note that the entropy generation curves for each pump are inversely proportional to the efficiency curves. In general, as efficiency nears unity, entropy generation approaches zero. Figure 4.4 also illustrates the effect of mass flow rate on a pump's efficiency. The mass flow rate corresponding to the highest efficiency of a given pump is referred to as the "best efficiency point," or BEP. The shaded regions around each BEP are preferred operating zones. Operating a pump at flow rates far from its BEP can cause wear and fatigue over time, thereby shortening the life span of the pump. Therefore when selecting a pump, it is preferable to choose a design for which the BEP is near the expected nominal operating point for a given application.

In the following section, I present the augmented SL CCD and CL design optimization algorithms for leveraging the HCCD algorithm to optimize the design and control of a thermal management system. I note that each algorithm is a specific form of the generalized structure for a hierarchical CCD algorithm presented in Section 4.1.

4.3 Hierarchical CCD: Optimization Problem at Each Level

The complete structure of the hierarchical CCD optimization algorithm for thermal management systems is shown in Fig. 4.5. To initialize the algorithm, I first declare initial setpoint mass flow rates of primary fluid $(\dot{m}_c)$ and secondary fluid $(\dot{m}_r)$, as well as initial values for the inlet temperatures and desired rate of heat rejection for the shell and tube heat exchanger (HX). Parallel component level (CL) algorithms then optimize the pumps and a shell and tube heat exchanger. These optimal component designs are passed up to the SL where the CCD algorithm optimizes the system for static and transient performance elements such as mass and robustness by selecting optimal mass flow rates along with other variables such as

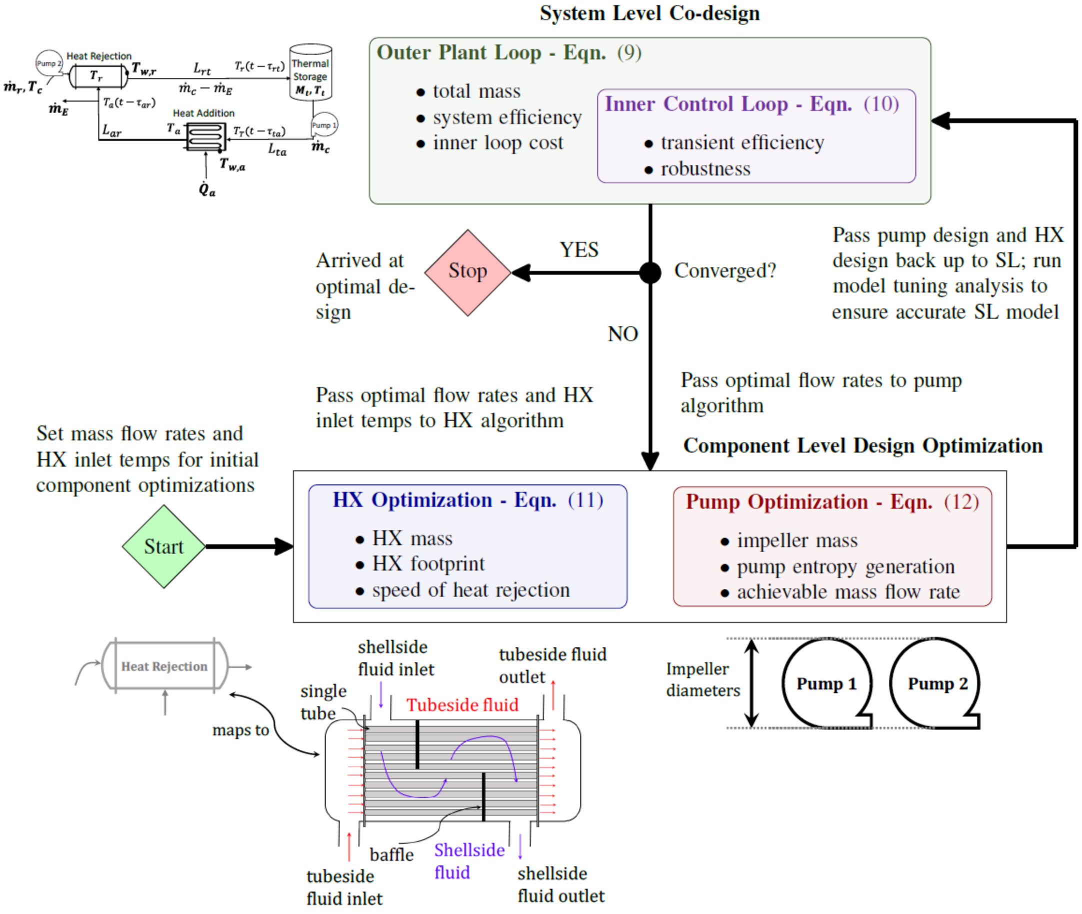

System Level Co-design
Outer Plant Loop - Eqn. (9)
• total mass
• system efficiency
• inner loop cost
Inner Control Loop - Eqn. (10)
• transient efficiency
• robustness
Arrived at optimal design
Stop
YES
Converged?
NO
Pass pump design and HX design back up to SL; run model tuning analysis to ensure accurate SL model
Pass optimal flow rates and HX inlet temps to HX algorithm
Pass optimal flow rates to pump algorithm
Component Level Design Optimization
Set mass flow rates and HX inlet temps for initial component optimizations
Start
HX Optimization - Eqn. (11)
• HX mass
• HX footprint
• speed of heat rejection
Pump Optimization - Eqn. (12)
• impeller mass
• pump entropy generation
• achievable mass flow rate
Heat Rejection
Thermal Storage
Heat Addition
Pump 1
Pump 2
maps to
shellside fluid inlet
tubeside fluid outlet
single tube
Tubeside fluid
baffle
Shellside fluid
tubeside fluid inlet
shellside fluid outlet
Impeller diameters
Pump 1
Pump 2

Figure 4.5. : Flowchart depicting operation of hierarchical co-design algorithm.

lumped parameters in the cold plate. The hierarchical optimization scheme iterates between the SL and CL until all levels achieve convergence to $\Delta J \leq \epsilon$ (Eqn. (4.1)).

The rest of this section details the structure of each individual algorithm and describes the manner in which the levels of the hierarchy are linked together. Note that the decision variables for the SL CCD algorithm and the CL heat exchanger algorithm are augmented from the algorithms outlined in Chapters 2-3 to facilitate integration of the two levels. Therefore, the slightly updated optimization problems for each level are restated here for clarity.

SL CCD outer loop plant optimization

$$\begin{aligned}
\min_{x_p} \quad & J_p\left(x_p, x_u, x_{hx}^*, x_{cp}^*\right) = \lambda_1 \cdot M\left(x_p, x_{hx}^*, x_{cp}^*\right) \\
& + \quad \lambda_2 \cdot \dot{S}_{gen}^e\left(x_p, x_{hx}^*, x_{cp}^*\right) \\
& + \lambda_3 \cdot \gamma^*\left(x_p, x_u, x_{hx}^*, x_{cp}^*\right) \\
\text{s.t.} \quad & x_{p,min} \le x_p \le x_{p,max} \\
& T_{c,low} \le T_c^e\left(x_p, x_{hx}^*, x_{cp}^*\right) \le T_{c,high} \\
& \dot{S}_{gen}^e\left(x_p, x_{hx}^*, x_{cp}^*\right) \ge 0 \\
& \dot{m}_c^e \le \epsilon \cdot \bar{\dot{m}}_{p,1} \\
& \dot{m}_r^e \le \epsilon \cdot \bar{\dot{m}}_{p,2} \\
& \gamma^*\left(x_p, x_u, x_{hx}^*, x_{cp}^*\right) \in (0, \infty)
\end{aligned} \tag{4.3}$$

SL CCD inner loop control optimization

$$\begin{aligned}
\beta^*\left(x_p, x_u, x_{hx}^*, x_{cp}^*\right) = \arg\min_{x_u} \quad & \beta \\
\text{s.t.} \quad & \text{LMI}_1 < 0 \\
& Q_1 > 0 \\
& \beta > 0 \\
\text{where } \gamma^* = \sqrt{\beta^*} &
\end{aligned} \tag{4.4}$$

CL Shell and Tube HX Dynamic Optimization

$$\begin{aligned}
x_{hx}^* = \arg\min_{x_{hx}} \quad & J_{hx}\left(x_{hx}, x_p^*\right) = \gamma_1 \cdot FP\left(x_{hx}, x_p^*\right) \\
& + \gamma_2 \cdot m_{tot}\left(x_{hx}, x_p^*\right) + \gamma_3 \cdot \left(\frac{1}{\bar{\lambda}\left(x_{hx}, x_p^*\right)}\right)^2 \\
\text{s.t.} \quad & x_{hx,min} \le x_{hx} \le x_{hx,max} \\
& \dot{Q}_{hx}\left(x_{hx}, x_p^*\right) \ge \dot{Q}_{des} \\
& B \le D_s\left(x_{hx}, x_p^*\right) \le 5B \\
& T_{tube,out}\left(x_{hx}, x_p^*\right) \ge T_{shell,out}\left(x_{hx}, x_p^*\right) \\
& A_{c,tubes}\left(x_{hx}, x_p^*\right) \le \pi/4 \cdot D_{s,shell}^2 \\
& C\left(x_{hx}, x_p^*\right), D_e\left(x_{hx}, x_p^*\right) \ge \epsilon \\
& \Delta P_{tubes}\left(x_{hx}, x_p^*\right), \ \Delta P_{shell}\left(x_{hx}, x_p^*\right) \le \epsilon
\end{aligned} \tag{4.5}$$

CL Pump Optimization

$$\begin{aligned}
x_{cp}^* = \arg\min_{x_{cp}} \quad & J_{cp}\left(x_{cp}, x_p^*\right) = \tau_1 \cdot m_{pumps}\left(x_{cp}, x_p^*\right) \\
& + \tau_2 \cdot \dot{S}_{gen,pumps}\left(x_{cp}, x_p^*\right) \\
& \eta_{p,1}^e\left(x_{cp}, x_p^*\right) \ge s_1 \cdot \eta_2^{BEP} \\
& \eta_{p,2}^e\left(x_{cp}, x_p^*\right) \ge s_2 \cdot \eta_2^{BEP} \\
& \eta_{p,1}^e\left(x_{cp}, x_p^*\right) \ge \eta_{min} \\
& \eta_{p,2}^e\left(x_{cp}, x_p^*\right) \ge \eta_{min}
\end{aligned} \tag{4.6}$$

4.3.1 CL Heat Exchanger Optimization

As detailed in Chapter 3, the CL shell and tube heat exchanger optimization is itself a dynamic optimization algorithm. Its objective is to use both the static and dynamics models derived in Chapter 3 to minimize a weighted combination of static and transient design characteristics. The full statement of the optimization problem is given in Eqn. (3.25). Recall that γ_1, γ_2, and γ_3 are user-defined weighting parameters that penalize footprint (FP), mass (m_{tot}), and transient performance ($\bar{\lambda}$), respectively. Also recall that $\bar{\lambda} = \lambda_{max}$ is the maximum, or slowest, eigenvalue of the linearized dynamic heat exchanger model for a given set of design parameters. As the slowest eigenvalue becomes faster, the magnitude of $\bar{\lambda}$ increases and the corresponding term in the objective function decreases.

Table 4.1. : Decision variables for S&T HX optimization in HCCD algorithm.

COMPONENT LEVEL SHELL AND TUBE HX	
Variable	**Description**
P_T	Tube pitch
D_o	Tube outer diameter
L_T	Length of tubes in tube bundle
N_T	Number of tubes in tube bundle
D_s	Shell inner diameter
B	Spacing between adjacent baffles

The decision variables for the heat exchanger algorithm are geometric quantities that dictate the heat transfer characteristics of the component; more specifically, the decision variable vector x_{hx} consists of tube pitch, tube outer diameter, length of a single tube, number of tubes, inner shell diameter, and baffle spacing. Table 4.1 provides a description of each quantity. I note that each heat exchanger design is also a function of optimal SL setpoint variables such as primary (tubeside) and secondary (shellside) mass flow rates. For a given CL iteration, the inlet mass flow rates and temperatures for both the primary (tubeside) and secondary (shellside) fluids are passed down from the previous iteration of the SL CCD algorithm. Each HX design must adhere to several constraints. First, each design variable is both lower bounded ($x_{hx,min}$) and upper bounded ($x_{hx,max}$). Second, I constrain the heat exchanger design to reject a minimum rate of heat transfer $\dot{Q}_{des}$. This ensures the CL HX design provides a physical component capable of achieving the heat removal rate, and thus the temperature characteristics, desired by the optimal TMS at the SL. Additionally, in accordance with typical design practices, the shell diameter (D_s) is constrained to be larger than the baffle spacing but no larger than five times the baffle spacing. The shell diameter is further constrained to ensure it is large enough to accommodate all N_T tubes. The tube clearance C and shell equivalent diameter

D_e, both functions of various decision variables, are required to be positive values. I constrain the tubeside fluid outlet temperature to be greater than the shellside fluid outlet temperature to prevent heat transfer crossover during system operation. Finally, I constrain each design such that the pressure drops for the tubeside and shellside fluids are below an allowable tolerance.

4.3.2 CL Pump Optimization

The CL pump optimization, stated formally in Eqn. (4.6), is a static algorithm designed to minimize a weighted combination of pump mass and pump entropy generation rate. In Eqn. (4.6), τ_1 and τ_2 are user-defined weights that penalize pump mass (m_{pumps}) and pump entropy generation rate ($\dot{S}_{gen,pumps}$), respectively. The decision variable vector for the pump algorithm (Table 4.2), x_{cp}, consists of two variables: the impeller diameter of the primary fluid pump ($D_{p,1}$) and the impeller diameter of the secondary fluid pump ($D_{p,2}$). Each pump design is likewise a function of the optimal SL design; the optimal flow rates passed down from the SL place restrictions on the impeller diameters that will meet the relevant pump design constraints. In addition to being both upper and lower bounded, the pump decision variables are constrained to ensure the pumps meet constraints related to achievable mass flow rate and efficiency. In Eqn. (4.6), each pump's efficiency at its nominal operating point ($\eta_{p,1}$, $\eta_{p,2}$) must be greater than a user-defined efficiency threshold η_{min}. In addition, each pump must nominally operate within a user-defined range of its best efficiency point (Fig. 4.4) to avoid unnecessary wear on the component.

Table 4.2. : Decision variables for pump optimization in HCCD algorithm.

COMPONENT LEVEL PUMPS

Variable	**Description**
$D_{p,1}$	Impeller diameter of Pump 1 in TMS
$D_{p,2}$	Impeller diameter of Pump 2 in TMS

4.3.3 SL CCD Algorithm

Recall that the SL CCD algorithm is a nested optimization wherein an outer plant loop optimizes a set of plant (system) design properties, including steady-state performance, and an inner control loop synthesizes a controller to optimize transient performance and robustness for the given plant. Here I use the outer loop to minimize a combination of system mass (M), steady-state entropy generation rate ($\dot{S}^e_{gen}$), and the inner loop's optimal cost (γ^*) while maintaining a heat addition component surface temperature of $T_{w,a} = 40$ °C in the presence of a nominal heat load $\dot{Q}^e_a = 5$ kW. Within the inner loop, I design a time-delayed H_∞ controller to optimize the fuel TMS's *transient* performance; more specifically, I place a penalty on regulating the system's entropy generation rate and the heat addition component's (cold plate heat sink) surface temperature ($T_{w,a}$). The full statement of the SL CCD algorithm is given in Eqns. (4.3)-(4.4).

Table 4.3. : Decision variables for SL CCD optimization in HCCD algorithm.

System Level TMS

Variable	**Description**
L_{ta}	Length of pipe between tank and heat sink
L_{ar}	Length of pipe between cold plate and S&T HX
L_{rt}	Length of pipe between S&T HX and tank
$L_{tot,a}$	Length of tube channeling in cold plate
M_T	Nominal amount of working fluid in tank
$\dot{m}_c$	Primary fluid mass flow rate
$\dot{m}_r$	Secondary fluid mass flow rate in S&T HX

The outer loop decision variables, described briefly in Table 4.3, are variables that connect the components of the integrated TMS, such as lengths of connecting pipe, as well as operational variables that dictate how the components operate together in an integrated system, such as mass flow rates. I also treat the total length of channel tubing in the cold plate, $L_{tot,a}$, and the initial (nominal) amount of fluid in the storage tank, M_T, as plant loop decision variables. The inner loop controller decision variables, described in detail in [59], are non-physical variables used to synthesize a state-feedback H_∞ controller to meet the control objectives for the given plant. The SL plant design must adhere to a number of constraints. First, each design variable is both upper and lower bounded. More specifically, the mass flow rates are bounded to be within safe operating ranges with respect to each pump's efficiency curve (see Fig. 4.4). Additionally, the entropy generation rates of each component must be non-negative. Bounds are placed on the nominal (steady-state) value of the secondary fluid temperature T_c^e of the shell and tube heat exchanger. In aircraft fuel thermal management, the secondary fluid in the heat rejection component is often ambient air; therefore, this temperature cannot be controlled and is typically within a known expected range. I also constrain each plant to ensure that a realizable controller exists for that plant. In other words, all feasible plant designs must have $\gamma \in (0, \infty)$. Finally, I constrain the optimal mass flow rates $\dot{m}_c$ and $\dot{m}_r$ to be less than the maximum achievable flow rates by a factor of safety to ensure the controller can increase each mass flow rate as needed during transients.

Equations (4.3)-(4.4) also illustrate that the SL objective function is not only a function of the outer loop decision variables (x_p) and inner loop decision variables (x_u), but also of the optimal solutions from the previous iteration of the shell and tube heat exchanger (x_{hx}^*) and pump (x_{cp}^*) algorithms. For example, the total mass of the optimal heat exchanger from Eqn. (3.25), m_{tot}^*, and the masses of each optimal pump from Eqn. (4.6), m_{pumps}^*, are included in the SL outer loop mass calculation (M) in Eqn. (4.3). Therefore component designs with poor performance characteristics are identified and penalized in the SL algorithm. Once an optimal plant and controller are designed at the SL, the mass flow rates and HX inlet temperatures for that system design are passed down to the CL which updates the component designs to accommodate the required setpoints from the SL CCD.

Recall that the SL model is intentionally low-fidelity whereas the CL models are of higher fidelity. The shell and tube heat exchanger from the CL maps to the heat rejection component at the SL, where the heat exchanger is treated as a single tube with lumped fluid temperature T_r and lumped wall temperature $T_{w,r}$. The difference in model fidelity can cause discrepancy in the predicted dynamics. To avoid this issue, I run a model tuning analysis preceding each iteration of the SL CCD algorithm. The nominal heat transfer coefficients in the SL model (α_a, α_r, α_c in Eqn. (2.1)) are tuned to ensure that the nominal temperatures in the SL model accurately reflect the temperature differences across the optimal HX resulting from the CL HX optimization for the given mass flow rates. The SL CCD algorithm can then be used to select new optimal flow rates based on the updated component geometry. The model matching allows the hierarchy to (1) select *physically tractable* component designs at the CL and (2) ensure the component(s) can be integrated at the SL to accomplish the desired tasks in an optimal manner. Therefore at each iteration of the hierarchy, the optimal heat exchanger geometry and pump characteristics are reflected by the SL design model; the subsequent iteration of the SL CCD algorithm then selects updated optimal variables to connect and control the optimal components.

4.4 Hierarchical Co-design Case Study: Results

In this section I present a case study to demonstrate the hierarchical CCD algorithm. I consider two cases: Case M and Case S. In Case M, the primary goal at each level in the hierarchy is to minimize system mass. In Case S, the goal at the system level is to minimize a weighted combination of system mass and entropy generation rate. The goal for the CL HX optimization in Case S is to minimize a combination of mass and footprint, and the goal for the CL pump optimization is to minimize a combination of the pump masses and their nominal rates of entropy generation. In both cases, an additional penalty is placed on the inner loop contribution γ^* in the

SL CCD algorithm to ensure the resulting system is robust to transient disturbances. The weightings for each case are summarized in Table 4.4.

Table 4.4. : Objective function weightings for CCD case study on thermal management.

$$J_p = \lambda_1 M + \lambda_2 \dot{S}_{gen} + \lambda_3 \gamma^*,\ J_{hx} = \gamma_1 FP + \gamma_2 m_{hx} + \gamma_3 \left(1/\bar{\lambda}\right)^2,\ J_{cp} = \tau_1 m_p + \tau_2 \dot{S}_{gen}$$

Case	λ_1	λ_2	λ_3	γ_1	γ_2	γ_3	τ_1	τ_2
M	1	0	1	0	1	0	1	0
S	1	5	1	500	1	0	1	3

In both cases, the SL inner loop is structured to place a penalty on regulating (1) the surface temperature of the heat addition component and (2) the system's transient entropy generation rate. I assume water to be both the primary fluid for the entire TMS loop and the secondary fluid in the heat rejection component and the heat exchangers are made of copper. For the initial component optimizations, I declare initial primary and secondary fluid mass flow rates to be $\dot{m}_c = 0.20$ kg/s and $\dot{m}_r = 0.20$ kg/s, respectively. Additionally, I declare the initial tubeside inlet temperature to be $T_{t,in} = 36$ °C and the initial shellside inlet temperature to be $T_{s,in} = 15$ °C along with the desired 5 kW rate of heat rejection.

Each individual algorithm uses the NOMAD nonlinear optimization algorithm which interfaces with MATLAB through the program OPTI [62]. In addition, the SL inner loop uses the software package Yalmip [63] to solve the linear matrix inequality (LMI) H_∞ control problem. Both component level algorithms use a multistart technique to avoid convergence to local optima. In both cases, the hierarchical CCD algorithm iterates back and forth between the SL and CL until all decision variables converge to within $\delta J^k \leq \epsilon^k$. Case M required 3 iterations across each level to converge, whereas Case S required 5 iterations. The final optimal system and components for each case are shown in Table 4.5. Several key differences in the design of the systems and components are discussed below.

Table 4.5. : Final optimal designs for hierarchical CCD case study.

SYSTEM LEVEL TMS

Case	L_{ta} (m)	L_{ar} (m)	L_{rt} (m)	$L_{tot,a}$ (m)	M_T (kg)	$\dot{m}_c$ (kg/s)	$\dot{m}_r$ (kg/s)	γ^* (–)	$\dot{S}_{gen}$ (W/K)	Mass (kg)
M	0.61	0.61	0.61	1.00	2.0	0.150	0.235	0.536	19.5	21.6
S	0.61	0.61	0.61	1.30	2.0	0.243	0.204	0.382	9.62	28.9

COMPONENT LEVEL

Shell and tube HX

Case	P_T (m)	D_o (m)	L_T (m)	N_T (–)	D_s (m)	B (m)	FP_{hx} (m^2)	m_{hx} (kg)
M	0.00559	0.00480	0.226	85	0.100	0.02	0.023	4.84
S	0.00603	0.00509	0.152	179	0.100	0.032	0.015	5.66

Pumps

Case	$D_{p,1}$ (m)	$D_{p,2}$ (m)	$m_{p,1}$ (kg)	$m_{p,2}$ (kg)	$\bar{\dot{m}}_{p,1}$ (kg/s)	$\bar{\dot{m}}_{p,2}$ (kg/s)	$\dot{S}_{gen,p,1}$ (W/K)	$\dot{S}_{gen,p,2}$ (W/K)
M	0.273	0.310	5.71	7.36	0.179	0.262	9.22	8.25
S	0.363	0.348	10.1	9.3	0.424	0.373	4.21	4.62

In Case M, the primary goal was to minimize the overall mass of the system, including the mass of the shell and tube HX and each pump. Because the masses of the pumps are the largest contributor to the overall mass of the system, the SL CCD algorithm opts for mass flow rates ($\dot{m}_c = 0.150$ kg/s, $\dot{m}_r = 0.235$ kg/s) that minimize the overall contribution of pump mass. In fact, the total mass contributed by the pumps in Case M is 13.07 kg, compared to a larger contribution of 19.4 kg in Case S. The CL pump algorithm then designed a set of pumps to ensure the system can be nominally operated within a safe range on each pump's efficiency curve (see Fig.

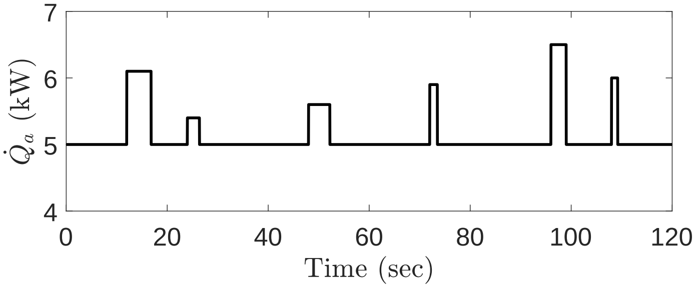
4.4) for the optimal flow rates determined at the SL. The optimal HX in Case M was designed to minimize mass for the desired optimal mass flow rates at the SL while meeting the inlet temperature setpoints determined at the SL for both the tubeside and shellside fluids. Recall from Eqn. (3.1) that the nominal rate of heat rejection for the shell and tube HX is proportional to UA_{hx}, the HX's overall heat transfer coefficient (HTC). Because the optimal flow rates in Case M were chosen to be small to minimize pump mass, the optimal HX is designed to enable a higher overall HTC. The optimal number of tubes (N_T) in Case M is 85 compared to 179 in Case S and the tubeside fluid is split amongst a lesser number of tubes in Case M; this leads to a higher mass flux and a higher overall HTC. Therefore, the optimal HX design in Case M is chosen such that it overcomes the smaller pump flow rates. By combining the SL CCD and CL optimizations, the total mass of the thermal management system in Case M is 21.6 kg, or 7.3 kg less than in Case S, a reduction of 34%.

In Case S, the goal for the SL CCD algorithm was to minimize a weighted combination of total system mass and nominal entropy generation rate ($\dot{S}_{gen}$). The CCD algorithm ultimately selects flow rates ($\dot{m}_c = 0.243$ kg/s, $\dot{m}_r = 0.204$ kg/s) to operate within an allowable range of each pump's efficiency curve which are in turn chosen to be larger in Case S to lower entropy generation due to pumping. Specifically, the pump algorithm chooses pump designs with impeller diameters $D_{p,1} = 0.363$ m and $D_{p,2} = 0.348$ m; in this case, the optimal pumps each have a nominal rate of entropy generation less than 5 W/K, whereas each pump in Case M had a nominal entropy generation rate greater than 8 W/K. The optimal HX in Case S was designed to minimize a combination of mass and footprint given optimal mass flow rates and inlet temperature setpoints determined through the previous iteration of the SL CCD algorithm. In this case, because there are higher nominal mass flow rates, the overall HTC (and tubeside mass flux) of the HX is not required to be as high as in Case M. Therefore, a higher number of short length tubes ($N_T = 179$, $L_T = 0.152$) is used in the optimal design. This allows a HX design with a much smaller footprint of 0.015 m^2, or a 30% reduction compared to Case M. In total, the nominal rate of entropy generation in Case S is $\dot{S}_{gen} = 9.62$ W/K, less than half of the $\dot{S}_{gen} = 19.5$ W/K in Case M. However, this increase in nominal efficiency does come at the expense of system mass; the total system mass in Case S is 28.9 kg compared to 21.6 kg in Case M.

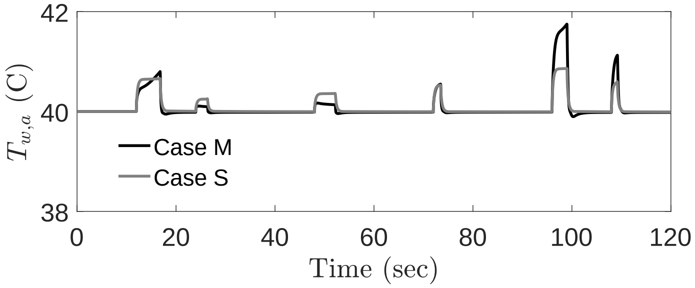
The results of these two cases illustrate how the HCCD algorithm can be used to design TMSs to meet different design objectives, namely mass and nominal efficiency. However, another key element of the algorithm concerns designing the system for robustness to transient disturbances. As described in Section 4.3, this is accomplished through the SL CCD's inner loop. In both Case S and Case M, I placed a penalty of 0.01 on regulating the cold plate surface temperature and a penalty of 10^{-5} on regulating the system's transient entropy generation rate. In each Case, the CCD algorithm generates an optimal H_∞ controller to ensure the regulation can be accomplished. Figures 4.6-4.7 help to illustrate the robustness of each design.

Figure 4.6b shows that, in each case, the optimal controllers are able to regulate the cold plate surface temperature to within two degrees in the presence of heat pulses 20% above nominal (Fig. 4.6a). Figure 4.7a illustrates how each controller modulates the mass flow rates, $\dot{m}_c$ and $\dot{m}_r$, during heat load pulses to achieve the desired temperature regulation. To regulate the system entropy generation rate (Fig. 4.6c), each controller modulates the secondary fluid mass flow rates ($\dot{m}_r$) in concert with $\dot{m}_c$ to limit the adverse affect of the disturbances on system efficiency during transients. While the entropy generation rate regulation is less effective than the surface temperature regulation, this is due to the heavier penalty placed on regulating the surface temperatures. In practice, the controller(s) can be tuned to achieve a desired balance between competing regulation objectives.

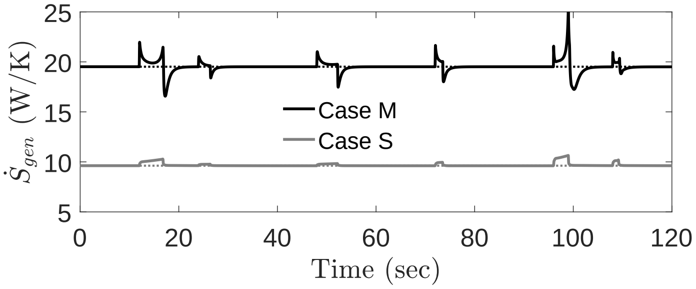
Iterating between the two levels is a key aspect of the proposed CCD hierarchy. By continually iterating between each level until the convergence criterion in Eqn. (4.1) is satisfied, the algorithm selects system and component designs otherwise unachievable by using a conventional approach. Without leveraging the hierarchy, a controls engineer might first use a set of general system specifications to optimize a



(a) Heat load disturbance



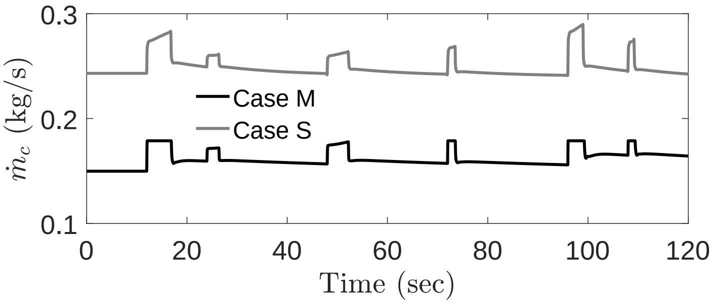
(b) Heat addition wall temperature



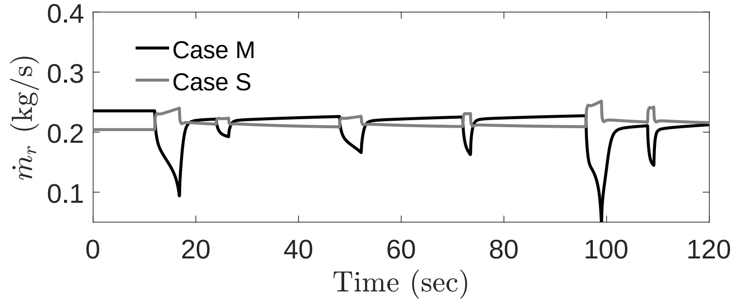
(c) Entropy generation rate

Figure 4.6. : CCD case study: disturbance rejection for cold plate surface temperature and system entropy generation rate.

set of components, followed by a CCD optimization to set the full plant design and controller for the given plant. This would correspond to selecting the components represented by COMPONENT LEVEL Iteration 1 and the plant/controller design from SYSTEM LEVEL TMS Iteration 1 in Table 4.6. By using this method of designing the TMS and its components, the total mass would be 24.1 kg. However, by iterating until Eqn. (4.1) is satisfied for all levels, the total system mass is reduced by over 10% to 21.6 kg at Iteration 3. This is done largely by (1) choosing a lower secondary (shellside) fluid inlet temperature and (2) lowering the mass flow rate of



(a) Primary fluid mass flow rate



(b) Secondary fluid mass flow rate

Figure 4.7. : CCD case study: modulated control inputs.

primary fluid in the TMS ($\dot{m}_c$). I note that the TMS primary fluid is the tubeside fluid in the shell and tube HX. Designing the system to operate nominally at a lower T_c value helps increase the natural temperature difference across the shell and tube heat exchanger which in turn means lower flow rates can meet the heat rejection requirements. Therefore, the SL CCD algorithm selects a lower $\dot{m}_c$ at the second iteration to lower the pump mass. Figures 4.8-4.9 illustrate the performance of the optimal TMS in Case M at both Iteration 1 and Iteration 3. By iterating between levels of the hierarchy, I am able to achieve equal levels of disturbance rejection for the cold plate surface temperature (Fig. 4.8b) while reducing the mass of the system by 10%. In both cases, the primary fluid mass flow rates are modulated to absorb the extra heat during the transient pulses as shown in Fig. 4.9a. Additionally, the secondary fluid mass flow rate is modulated as shown in Fig. 4.9b to balance out losses due to pumping such that the entropy generation rate can be regulated as desired.

Table 4.6. : CCD Case M: Optimal solutions at each iteration of the hierarchy.

SYSTEM LEVEL TMS

Iteration	L_{ta} (m)	L_{ar} (m)	L_{rt} (m)	$L_{tot,a}$ (m)	M_T (kg)	$\dot{m}_c$ (kg/s)	$\dot{m}_r$ (kg/s)	γ^* (–)	$\dot{S}_{gen}$ (W/K)	Mass (kg)
1	0.610	0.610	0.610	1.0	2.0	0.156	0.211	0.415	17.0	24.1
2	0.610	0.610	0.610	1.0	2.0	0.153	0.223	0.533	19.6	21.3
3	0.610	0.610	0.61	1.0	2.0	0.150	0.235	0.536	19.5	21.6

COMPONENT LEVEL

Shell and tube HX

Iteration	P_T (m)	D_o (m)	L_T (m)	N_T (–)	D_s (m)	B (m)	FP_{hx} (m^2)	m_{hx} (kg)
1	0.00585	0.00505	0.280	96	0.100	0.020	0.028	6.75
2	0.00559	0.00479	0.220	87	0.100	0.020	0.022	4.78
3	0.00559	0.00480	0.226	85	0.100	0.020	0.023	4.84

Pumps

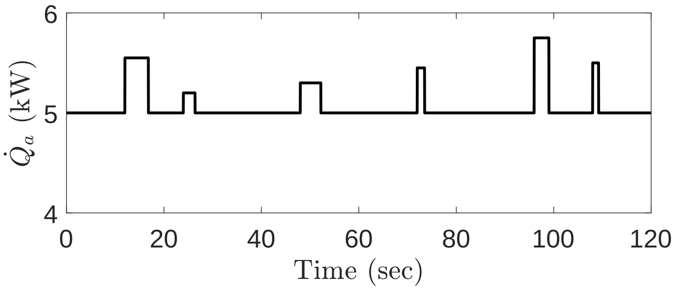
Iteration	$D_{p,1}$ (m)	$D_{p,2}$ (m)	$m_{p,1}$ (kg)	$m_{p,2}$ (kg)	$\bar{\dot{m}}_{p,1}$ (kg/s)	$\bar{\dot{m}}_{p,2}$ (kg/s)	$\dot{S}_{gen,p,1}$ (W/K)	$\dot{S}_{gen,p,2}$ (W/K)
1	0.298	0.298	6.84	6.84	0.235	0.235	8.54	8.54
2	0.275	0.304	5.79	7.10	0.183	0.248	9.17	8.39
3	0.273	0.310	5.71	7.36	0.179	0.262	9.22	8.25

The case study presented in this section illustrates how the proposed HCCD algorithm can be used to design robust systems with different design objectives. In the following section, I summarize the main takeaways and contributions of the algorithm.

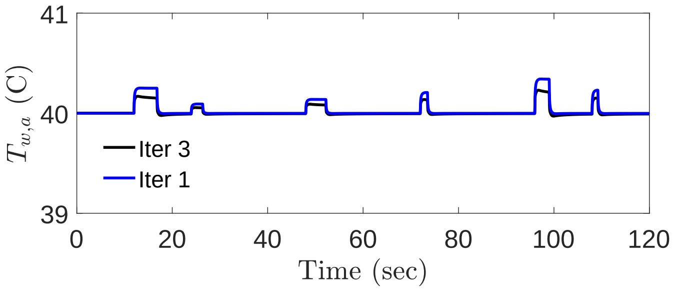
4.5 Chapter Summary

In this chapter, I presented a new hierarchical CCD methodology that combines CCD with high-fidelity component design optimization. The algorithm is designed to optimize the transient performance and robustness of TMSs in addition to optimizing a combination of component level design properties. Key takeaways are summarized as follows:

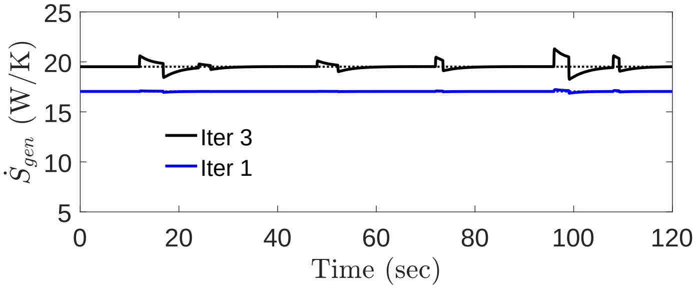
1. The algorithm integrates system level (SL) co-design with high-fidelity component level (CL) design optimization.



(a) Heat load disturbance



(b) Cold plate (heat addition) surface temperature

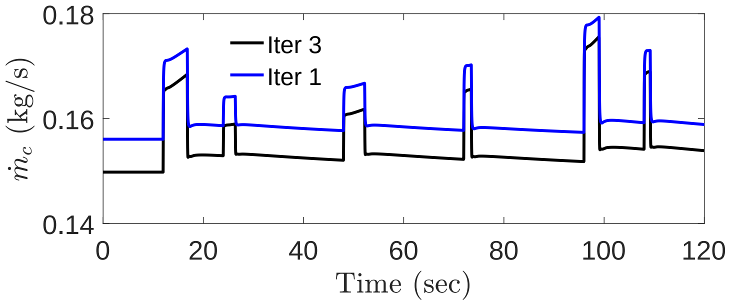


(c) Entropy generation rate

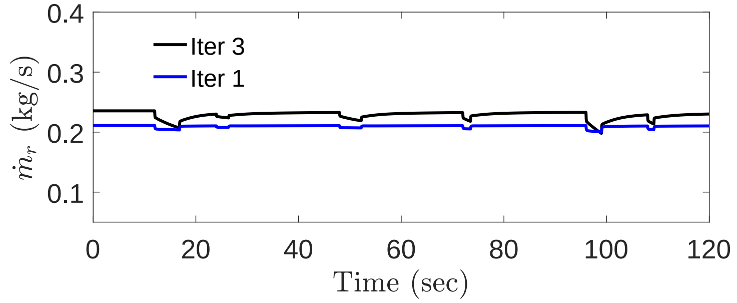
Figure 4.8. : Case M: Comparison of robustness performance for initial and final iterations.

2. The lower-fidelity SL co-design algorithm incorporates feedback control into the design of a complex system to ensure controllability and robust transient response to exogenous disturbances.

3. Multiple higher-fidelity CL design optimization algorithms solved in parallel provide a way of designing detailed components to achieve the desired performance needed at the SL.



(a) Primary fluid mass flow rate



(b) Secondary fluid mass flow rate

Figure 4.9. : Case M: modulated control inputs.

4. A model tuning analysis preceding the SL CCD optimization ensures the reduced order SL model dynamics closely match those of the optimal components designed at the CL.
5. Key specifications are passed back and forth between levels of the hierarchy at each iteration to converge on an optimal design that is responsive to desired objectives at each level.

5. HCCD SOLUTION VALIDATION

The HCCD algorithm presented in Chapter 4 is ultimately a model-based design process. Therefore, the work in this chapter seeks to validate that one can build the system that was modeled and see the behavior which was predicted by the HCCD algorithm. For the validation, I use the thermal modeling software ATTMO (**A**FRL **T**ransient **T**hermal **M**anagement **O**ptimization). ATTMO is an expansive toolset, initially designed by P.C. Krause and Associates (PCKA [80]) in collaboration with Air Force Research Laboratory (AFRL) and the University of Illinois, for transient modeling of thermal systems. The toolset supports the rapid development of dynamic thermal system models and investigation of advanced control algorithms necessary to support the dynamic cooling and power requirements associated with advanced aircraft architectures. In recent years, ATTMO has been used extensively in testing and developing transient models [81, 82] and model-based controllers [55] for thermal management systems.

5.1 ATTMO Toolset and System Construction

Creating a high-fidelity TMS model in ATTMO is the first step in validating the solution to the hierarchical CCD algorithm. ATTMO interfaces with the Simulink environment in MATLAB to enable quick connection of drag and drop components. Figure 5.1 shows an ATTMO construction of the TMS architecture optimized in Chapter 4.

Similar to the TMS architecture in this thesis, the primary fluid (water) is held in the storage tank and used to cool the surface of the cold plate, or heat addition component. The coolant is then routed through a shell and tube heat exchanger, or heat rejection component, where the absorbed heat is rejected to ambient through the

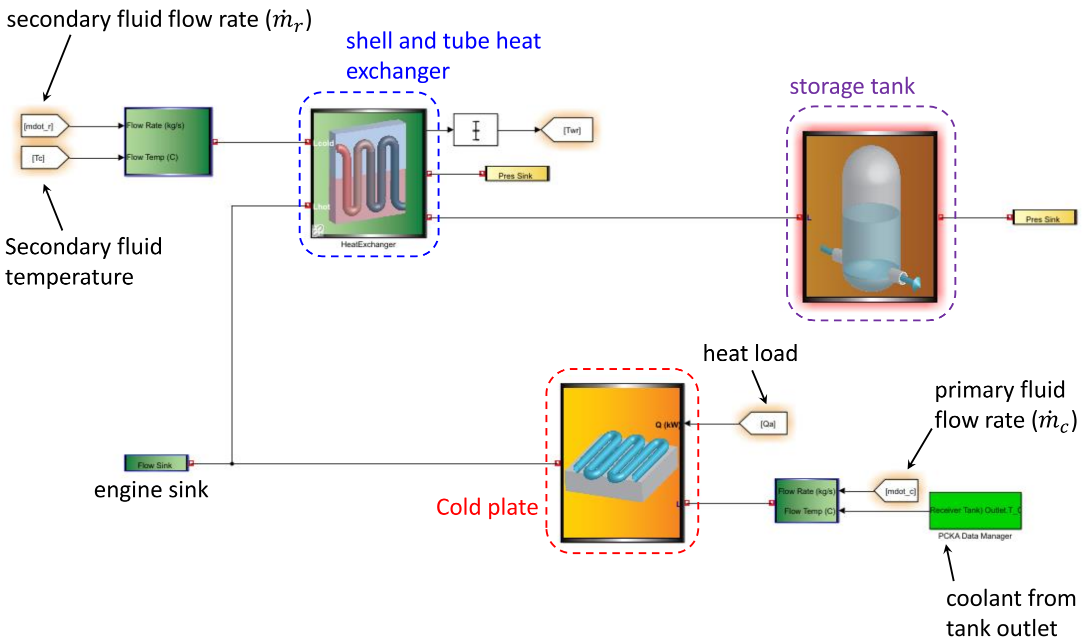
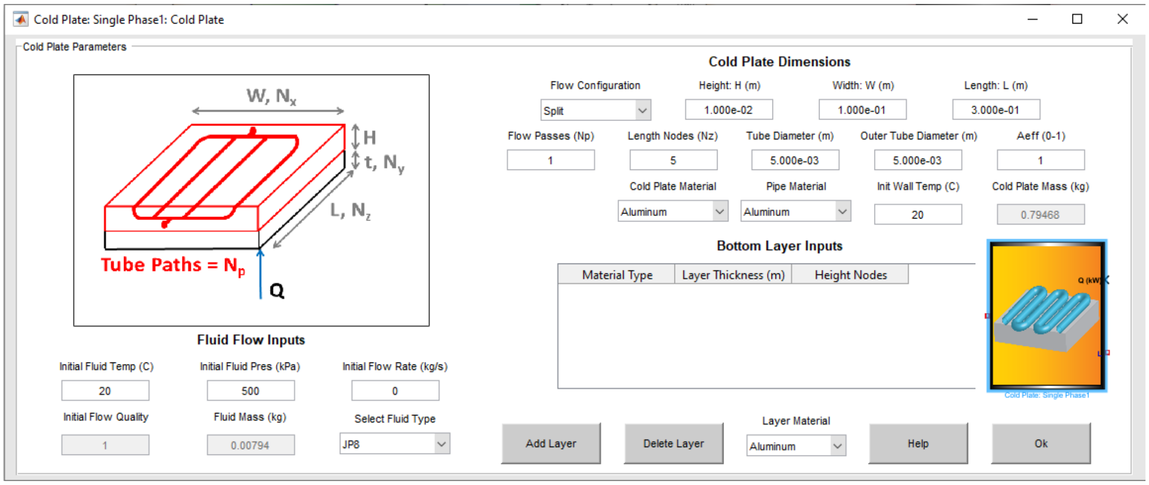


Figure 5.1. : Model of TMS architecture in ATTMO environment.

secondary fluid. The primary fluid is then returned to the storage tank to complete the cycle.

For the intents of the model and control validation in this chapter, the pumps are removed from consideration, and the primary and secondary fluid mass flow rates are instead modulated as commanded by the H_∞ controller. In practice, a map between pump speed and mass flow rate might be used to modulate the pump speed to achieve the desired flow rates to control the system. This would also dictate pressure differentials across the pump. However, due to the single-phase working fluid in this system, tracking and maintaining pressure is not considered.

The ATTMO cold plate block has many user-selected inputs as shown in Figure 5.2. Users can set quantities such as the physical dimensions of the plate, along with tube diameter for the channels of fluid used to cool the plate. For comparison to the TMS system model used in the hierarchical CCD algorithm, the cold plate length and number of flow passes are mapped to the decision variable $L_{tot,a}$, and the initial (nominal) temperature at steady-state is the temperature T_a in Figure 2.1.



Cold Plate: Single Phase1: Cold Plate

Cold Plate Parameters

Fluid Flow Inputs

Initial Fluid Temp (C): 20
Initial Fluid Pres (kPa): 500
Initial Flow Rate (kg/s): 0
Initial Flow Quality: 1
Fluid Mass (kg): 0.00794
Select Fluid Type: JP8

Cold Plate Dimensions

Flow Configuration: Split
Height: H (m): 1.000e-02
Width: W (m): 1.000e-01
Length: L (m): 3.000e-01
Flow Passes (Np): 1
Length Nodes (Nz): 5
Tube Diameter (m): 5.000e-03
Outer Tube Diameter (m): 5.000e-03
Aeff (0-1): 1
Cold Plate Material: Aluminum
Pipe Material: Aluminum
Init Wall Temp (C): 20
Cold Plate Mass (kg): 0.79468

Bottom Layer Inputs

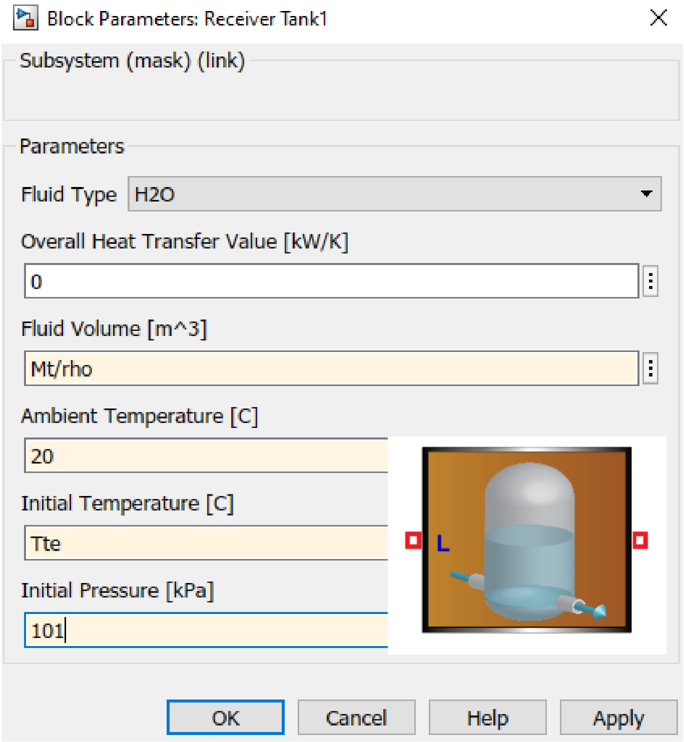
Material Type	Layer Thickness (m)	Height Nodes

Layer Material: Aluminum

Figure 5.2. : ATTMO heat addition component inputs.

Similarly, the storage (receiver) tank has five user-selected inputs as shown in Figure 5.3: overall heat transfer to ambient, initial (nominal) fluid volume, ambient temperature, initial (nominal) fluid temperature, and initial pressure. The initial fluid volume is proportional to the optimal decision variable M_T in the hierarchical CCD algorithm and the initial temperature is equal to T_t in Figure 2.1. Similar to the HCCD SL model, the model here assumes constant pressure within the receiver, constant ambient temperature, and neglect heat transfer between the tank and ambient.

Finally, the shell and tube heat exchanger has many user-defined inputs that corresponds to specific design variables optimized in the hierarchical CCD case study in Chapter 4. Shown in Figure 5.4, a variety of characteristics and properties can be chosen for the heat exchanger. The simulations executed here consider parallel flow, copper walls, and water as both the primary and secondary fluid. Additionally, the tubeside fluid is chosen to be the hot fluid. The red box in Fig. 5.4 highlights the user-defined geometric quantities related to the physical structure of the heat exchanger. Specifically, user-defined parameters include the tube type (square pitch), tube pitch, shell diameter and thickness, tube inner and outer diameter, a baffle cut percentage, tube length, and baffle spacing. Most of these quantities are analogous



Block Parameters: Receiver Tank1

Subsystem (mask) (link)

Parameters

Fluid Type: H2O

Overall Heat Transfer Value [kW/K]: 0

Fluid Volume [m^3]: Mt/rho

Ambient Temperature [C]: 20

Initial Temperature [C]: Tte

Initial Pressure [kPa]: 101

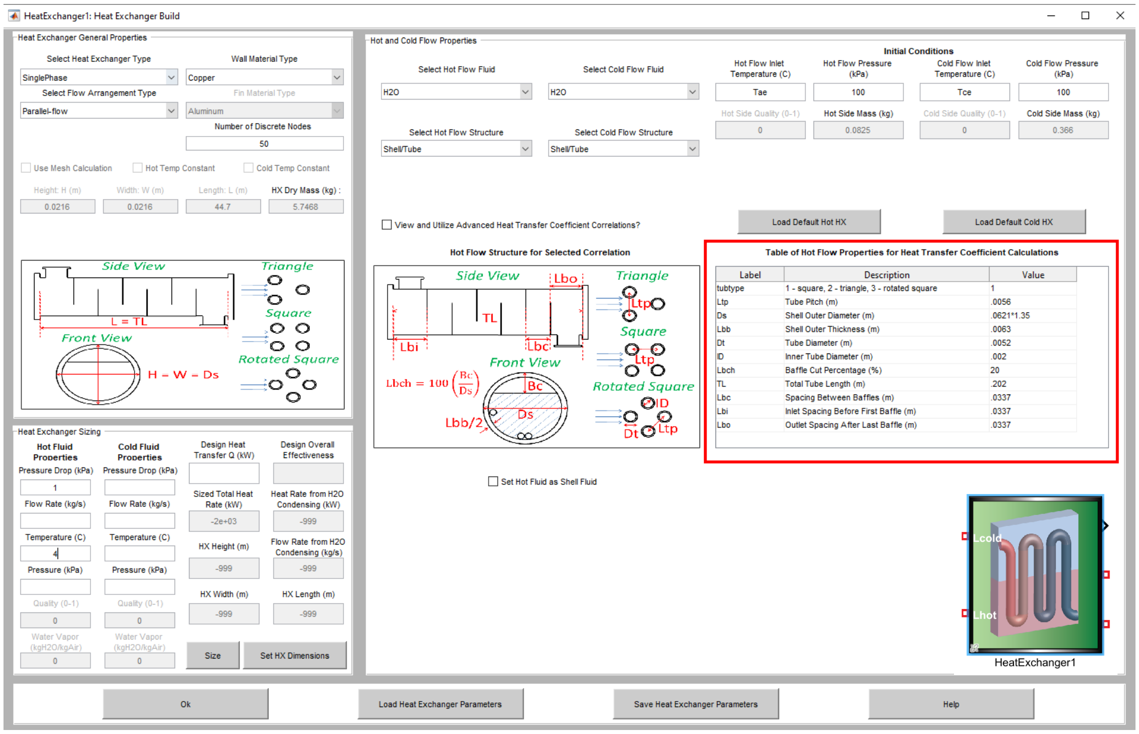
Figure 5.3. : ATTMO storage (receiver) tank inputs.

to quantities in the decision variable vector of the dynamic HX optimization used in the hierarchical CCD algorithm. Additionally, initial values for the tubeside and shellside inlet temperatures, given by T_a and T_c, respectively, in Figure 2.1, must be declared.

The system is first parameterized to closely resemble the optimal solution for Case M of the hierarchical CCD case study in Chapter 4. Table 5.1 gives a value for each user-defined quantity.

5.2 ATTMO Component Model Validation

A first step in model validation is to examine the nominal operating points of each heat exchanger. Here, I consider the nominal point of the ATTMO system and components to be ground truth. To ensure the heat exchanger models used in the



HeatExchanger1: Heat Exchanger Build

Heat Exchanger General Properties

Select Heat Exchanger Type: SinglePhase
Wall Material Type: Copper
Select Flow Arrangement Type: Parallel-flow
Fin Material Type: Aluminum
Number of Discrete Nodes: 50
☐ Use Mesh Calculation ☐ Hot Temp Constant ☐ Cold Temp Constant
Height: H (m): 0.0216
Width: W (m): 0.0216
Length: L (m): 44.7
HX Dry Mass (kg) : 5.7468

Heat Exchanger Sizing

Hot Fluid Properties
Pressure Drop (kPa): 1
Flow Rate (kg/s):
Temperature (C): 4
Pressure (kPa):
Quality (0-1): 0
Water Vapor (kgH2O/kgAir): 0

Cold Fluid Properties
Pressure Drop (kPa):
Flow Rate (kg/s):
Temperature (C):
Pressure (kPa):
Quality (0-1): 0
Water Vapor (kgH2O/kgAir): 0

Design Heat Transfer Q (kW):
Design Overall Effectiveness:
Sized Total Heat Rate (kW): -2e+03
Heat Rate from H2O Condensing (kW): -999
HX Height (m): -999
Flow Rate from H2O Condensing (kg/s): -999
HX Width (m): -999
HX Length (m): -999

Hot and Cold Flow Properties

Select Hot Flow Fluid: H2O
Select Cold Flow Fluid: H2O
Select Hot Flow Structure: Shell/Tube
Select Cold Flow Structure: Shell/Tube

Initial Conditions
Hot Flow Inlet Temperature (C): Tae
Hot Flow Pressure (kPa): 100
Cold Flow Inlet Temperature (C): Tce
Cold Flow Pressure (kPa): 100
Hot Side Quality (0-1): 0
Hot Side Mass (kg): 0.0825
Cold Side Quality (0-1): 0
Cold Side Mass (kg): 0.366

☐ View and Utilize Advanced Heat Transfer Coefficient Correlations?

Hot Flow Structure for Selected Correlation

Table of Hot Flow Properties for Heat Transfer Coefficient Calculations

Label	Description	Value
tubtype	1 - square, 2 - triangle, 3 - rotated square	1
Ltp	Tube Pitch (m)	.0056
Ds	Shell Outer Diameter (m)	.0621*1.35
Lbb	Shell Outer Thickness (m)	.0063
Dt	Tube Diameter (m)	.0052
ID	Inner Tube Diameter (m)	.002
Lbch	Baffle Cut Percentage (%)	20
TL	Total Tube Length (m)	.202
Lbc	Spacing Between Baffles (m)	.0337
Lbi	Inlet Spacing Before First Baffle (m)	.0337
Lbo	Outlet Spacing After Last Baffle (m)	.0337

☐ Set Hot Fluid as Shell Fluid

Figure 5.4. : ATTMO shell and tube heat exchanger inputs.

CCD algorithm are accurate physical representations of the detailed heat exchanger models available in ATTMO, model validation tests are run with the ATTMO cold plate and shell and tube heat exchanger components. For the set of nominal inputs from Case M of Chapter 4 ($\dot{m}_c = 0.1498$ kg/s $\dot{m}_r = 0.2354$ kg/s, $T_c = 10$ °C, $\dot{Q}_a = 5$ kW) both heat exchanger components are simulated with step signals in each input, and I compare the true nominal states of the ATTMO model to those from the HCCD system model.

5.2.1 Cold Plate Model Validation

For the cold plate model validation, the setup in ATTMO is shown in Figure 5.5 and the nominal fluid outlet temperature and nominal surface temperature are compared in Table 5.2. At steady-state, the fluid outlet temperature and surface

Table 5.1. : Parameter set for ATTMO validation of hierarchical CCD solution.

Heat Addition Component

Quanity	**Value**	**Quantity**	**Value**
Cold plate length (m)	0.1	Number of flow passes	10
Tube diameter (m)	0.0063	Initial temperature (C)	36.8

Storage Tank

Quanity	**Value**	**Quantity**	**Value**
Fluid volume (m^3)	0.002	Overall HT value (kW/K)	0
Ambient temperature (C)	20	Initial temperature (C)	28.8

Shell and tube heat exchanger

Quanity	**Value**	**Quantity**	**Value**
Tube pitch (m)	0.0056	Shell outer diameter (m)	0.1064
Shell thickness (m)	0.0064	Tube diameter (m)	0.0048
Inner tube diameter (m)	0.0016	Baffle cut percentage	10
Tube length (m)	0.2265	Baffle spacing (m)	0.02
Initial tubeside temp (C)	28.8	Initial shellside temp (C)	10

temperature for the ATTMO model converge to within 1% of the values of the optimal HCCD model in Case M of Chapter 4.

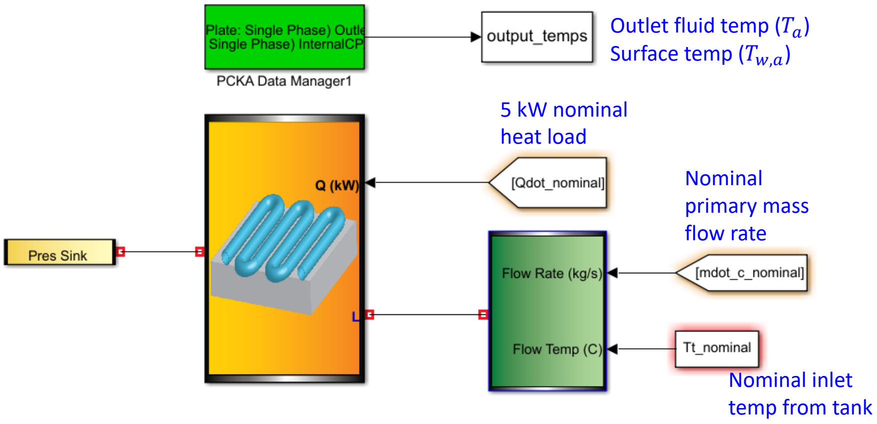


Figure 5.5. : ATTMO setup for cold plate component model validation.

Table 5.2. : Comparison of cold plate nominal conditions for HCCD prediction and ATTMO model.

Quantity	**Description**	Nominal Value (C) **HCCD**	 **ATTMO**
T_a	CP fluid outlet temp	36.83	36.82
$T_{w,a}$	CP surface temperature	40.0	40.1

5.2.2 Shell and Tube Heat Exchanger Model Validation

For the shell and tube heat exchanger validation, the setup in ATTMO is shown in Figure 5.6 and the nominal tubeside and shellside outlet temperatures are listed in Table 5.3. Here some discrepancy is observed between the ATTMO model's outlet temperatures at steady-state and the HCCD optimal system's outlet temperatures. Given the nominal flow rates and inlet temperatures from the HCDD model as inputs, the ATTMO shell and tube heat exchanger has a tubeside outlet temperature roughly three degrees lower than the HCCD model and a shellside outlet temperature roughly two degrees higher. This means that, at steady-state, the ATTMO heat exchanger rejects more heat than the required 5 kW load.

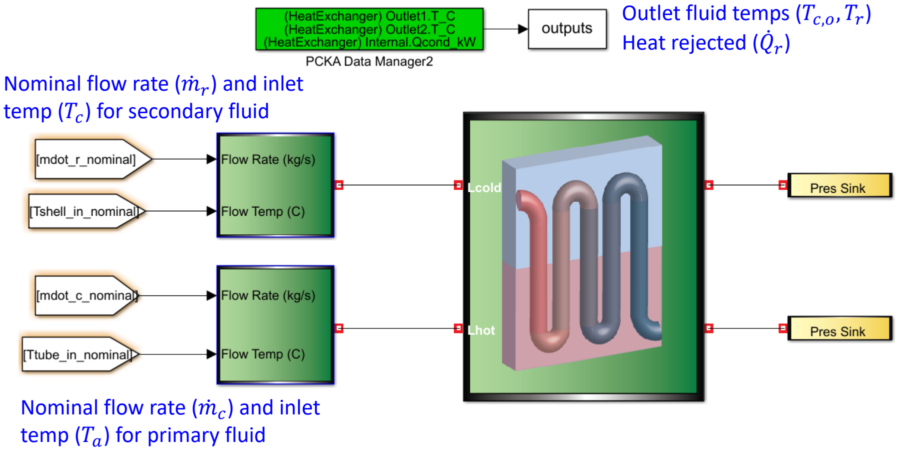

(HeatExchanger) Outlet1.T_C
(HeatExchanger) Outlet2.T_C
(HeatExchanger) Internal.Qcond_kW
PCKA Data Manager2
outputs
Outlet fluid temps ($T_{c,o}, T_r$)
Heat rejected ($\dot{Q}_r$)
Nominal flow rate ($\dot{m}_r$) and inlet temp (T_c) for secondary fluid
[mdot_r_nominal]
[Tshell_in_nominal]
Flow Rate (kg/s)
Flow Temp (C)
[mdot_c_nominal]
[Ttube_in_nominal]
Flow Rate (kg/s)
Flow Temp (C)
Lcold
Lhot
Pres Sink
Pres Sink
Nominal flow rate ($\dot{m}_c$) and inlet temp (T_a) for primary fluid

Figure 5.6. : ATTMO setup for cold plate component model validation.

Table 5.3. : Comparison of shell and tube HX nominal conditions for HCCD prediction and ATTMO model.

		Nominal Value (C)	
Quanity	**Description**	**HCCD**	**ATTMO**
$T_{t,o}$	Tubeside outlet temp	28.9	25.8
$T_{s,o}$	Shellside outlet temp	15.07	17.02

The discrepancy is likely due to differences between the shell and tube HX model used in the HCCD algorithm and the detailed modeling equations used in ATTMO. For example, the detailed HCCD HX model in Chapter 3 uses a number of parallel tubes (N_T) as an input whereas the ATTMO model calculates a value for the number of tubes based on other tube bundle and shell characteristics. Therefore, the analogous system in ATTMO will have a slightly different nominal operating point compared to what was predicted by the HCCD algorithm. In practice, a level of discrepancy between model-predicted performance and actual physical performance is tolerable. Once a physical system is constructed, the true nominal operating point can then be determined and the controller can be redesigned for the given physical system.

As such, the next step is to implement the optimal H_∞ feedback controller from Case M in the HCCD case study on the ATTMO system. With the controller in place, the HCCD design and control method can be validated.

5.3 Model-Based Tracking State Estimation

In order to implement the optimal controller designed using the HCCD algorithm, an estimator is required for the tracking state ζ. While this state is undetectable, all other states, inputs, and disturbances are measurable, or at least observable, in practice. Therefore it is straightforward to construct an open loop estimator to be used by the H_∞ controller derived in Section 2.2.2. Figure 5.7 gives an overview of the process.

$$\dot{S}_{gen} = f(\overbrace{M_T, T_T, T_r, T_a, T_{w,r}, T_{w,a}}^{x}, \overbrace{\dot{m}_c, \dot{m}_r}^{u}, \overbrace{\dot{m}_E, \dot{Q}_a, T_c}^{d})$$

$$\frac{d\zeta}{dt} = \dot{S}_{gen} - r$$

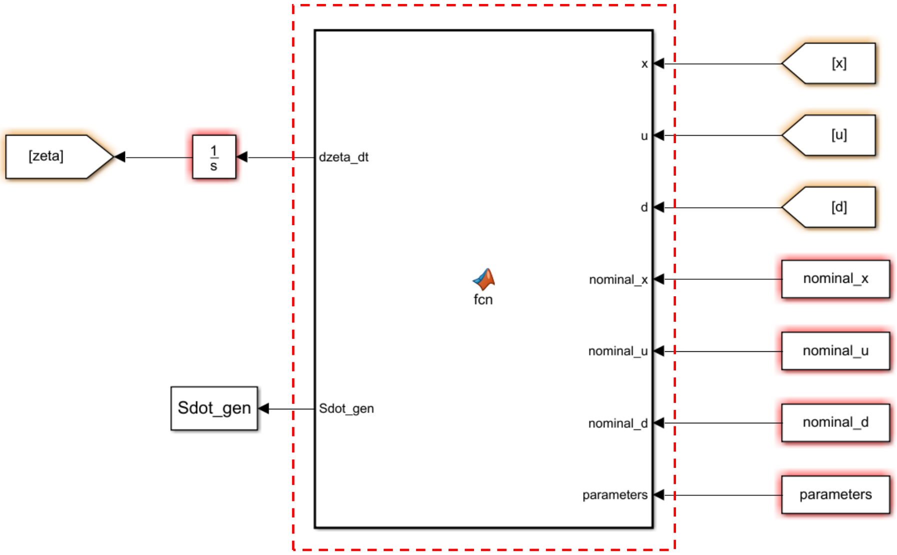


Figure 5.7. : Open loop estimation of the tracking state ζ in ATTMO/Simulink environment.

The open loop estimator takes the state (x), input (u), and disturbance (d) signals at the last sample instant, along with model-based constant parameters and the nominal operating point, to generate a model-based estimate of the system's entropy generation rate. An estimate of the tracking state ζ is computed by numerically integrating the estimated quantity $d\zeta/dt = \dot{S}_{gen} - r$. The control inputs at the current sample are then computed using the available measurements and tracking state estimate, along with the static gain matrix K as shown in Figure 5.8. Note that saturation limits are also applied to each control input.

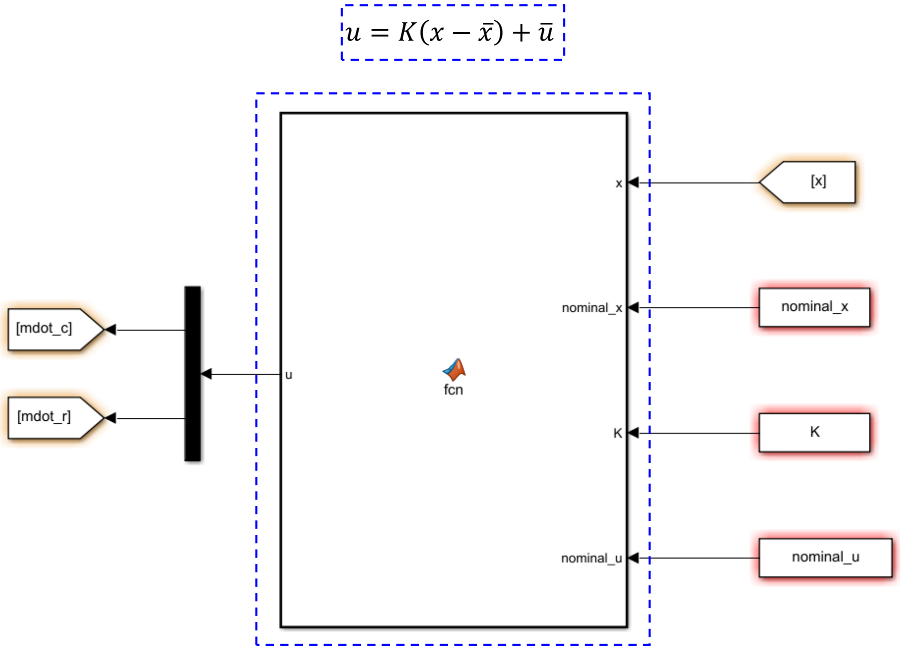


Figure 5.8. : Computation of control inputs in ATTMO/Simulink environment.

5.4 Optimal Controller Design in ATTMO

To test the optimal controller from Case M in Chapter 4, the H_∞ control problem in Eqn. (2.17) must be solved for an optimal set of controller gains for the ATTMO system. Because the H_∞ controller is a linear regulator, it requires a nominal (equilibrium) point to linearize about. However, as mentioned previously, the ATTMO system has a different nominal point than the one predicted by the HCCD system model. Therefore, the nominal point used by the linear controller in the inner loop of the HCCD algorithm should be updated based on the dynamics of the true ATTMO model. To determine the new nominal point, an open loop simulation is conducted where all nominal input values are applied as step inputs at time $t = 0$ in ATTMO. The final values of all state temperatures from the model can then be used as the true system nominal point for the linear controller model. Note that in practice, open loop experimental data would be collected, and the model-based controller could then be tuned to match physical data before updating the feedback controller gains.

Once a true nominal point is extracted from the high fidelity ATTMO model, a model tuning analysis, similar to the analysis performed between CL and SL iterations in the HCCD algorithm, is used to update the parameters of the physics-based system model in Eqn. (2.1). More specifically, MATLAB's *fmincon* is used to run the following unconstrained optimization problem:

$$\min_{PAF_s} \quad J = \sum_{i=1}^{6} |T_i^{ATTMO} - T_i^{CCD}|. \tag{5.1}$$

The decision variables are parameter adjustment factors (PAFs) which scale the heat transfer coefficients (α_c, α_r, α_a) in the SL model of the CCD algorithm. Within each iteration of Eqn. (5.1), the nominal point of the SL model is determined analytically given the set of scaled heat transfer coefficients. The cost at that iteration is then given by a summation of the differences between the updated HCCD model nominal temperatures and the ATTMO model nominal temperatures. Using this tuning analysis, the HCCD model's nominal condition matches ATTMO to within one degree Celsius.

With a tuned physics-based model of the ATTMO dynamics, the optimal controller gains are updated by solving the H_∞ control optimization given in Eqn. (2.17). Note that the updated model parameters are also used for the online tracking state estimation described previously. Now, the controller can be tested on the optimal CCD system built in ATTMO. The full process to update the controller gains is shown in Figure 5.9.

5.5 ATTMO Simulation: Validation of Optimal System Design

With the updated optimal controller designed for the ATTMO system, along with the estimator for the tracking state, I can validate the optimal system design in ATTMO. To do so, the ATTMO system is simulated for a series of heat pulses applied to the surface of the cold plate (see Figure 5.1). The controller is designed

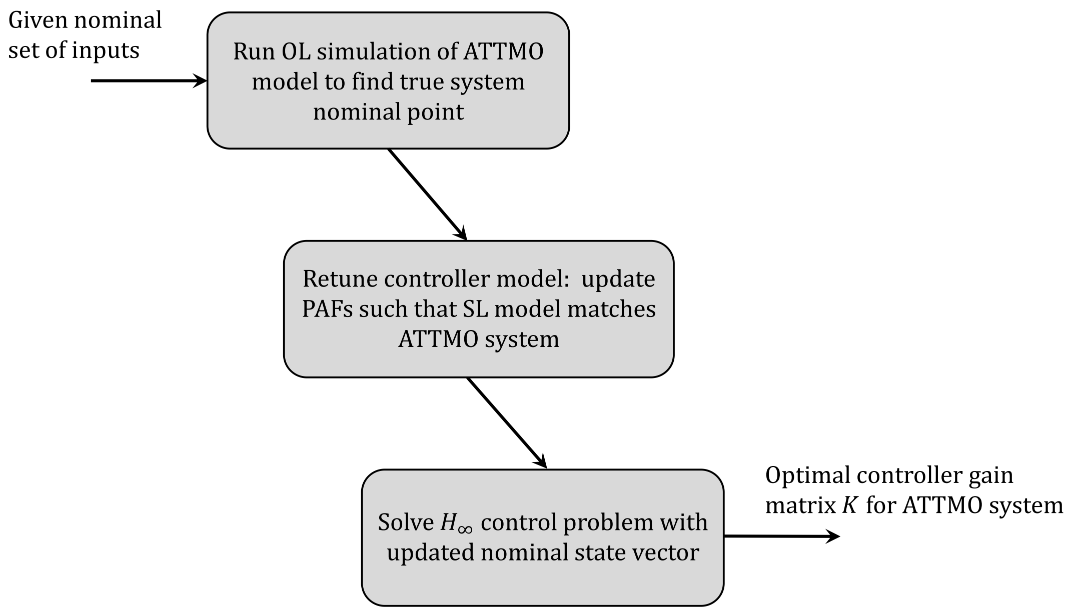
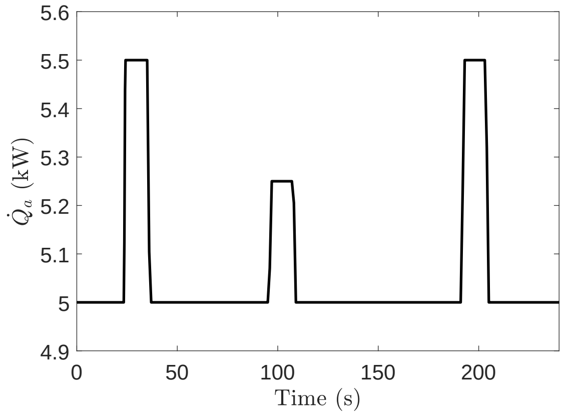

Given nominal set of inputs
Run OL simulation of ATTMO model to find true system nominal point
Retune controller model: update PAFs such that SL model matches ATTMO system
Solve H_∞ control problem with updated nominal state vector
Optimal controller gain matrix K for ATTMO system

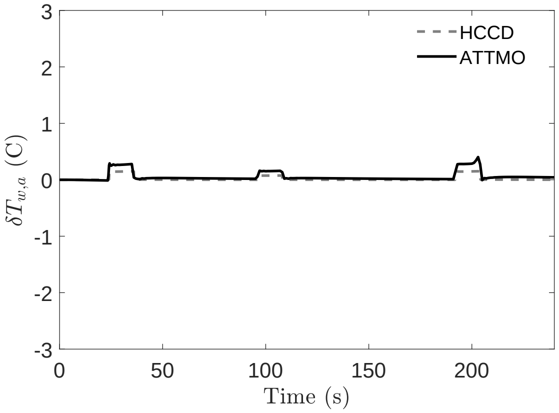
Figure 5.9. : Process used to solve for H_∞ controller gains for regulating ATTMO system model.

to regulate the cold plate surface temperature to its nominal value. The heat load impinged on the cold plate is given in Figure 5.10a and the regulated cold plate surface temperature is shown in Figure 5.10b-5.10c. The controlled inputs are shown in Figure 5.11. In each figure, the predicted performance from Case M of the HCCD optimization case study in Chapter 4 is overlaid for comparison. Recall that the actual nominal point for the ATTMO model is different than that predicted by the HCCD algorithm. Therefore, the vertical axes in Figure 5.10b-5.10c represent perturbations from the equilibrium point of both the ATTMO system and the HCCD system.

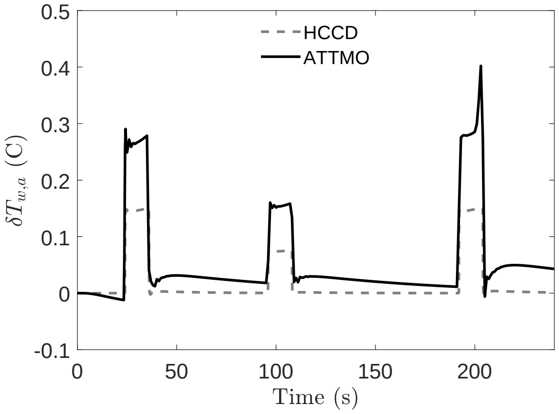
The surface temperature of the cold plate in ATTMO is regulated to within a half degree Celsius in the presence of heat pulses 10% above the nominal heat load using the optimal controller implemented in ATTMO. Figure 5.11 illustrates the similarity in behavior of the ATTMO system controller and the HCCD controller. In either case, the primary fluid mass flow rate is modulated during the heat pulses to absorb the requisite amount of transient heat necessary to maintain the cold plate surface temperature. Figure 5.10 shows that the HCCD system overestimates the robust-



(a) Heat load disturbance pulses

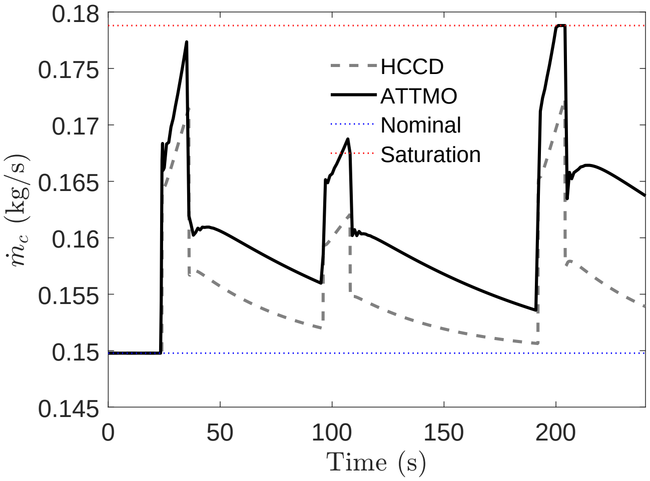


(b) Cold plate surface temperature

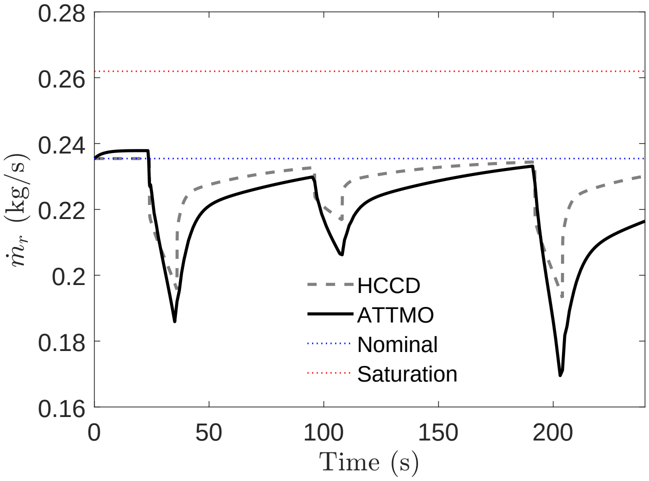


(c) Cold plate surface temperature (zoomed in)

Figure 5.10. : Comparison of ATTMO system performance with HCCD predicted performance. The δ terms on the vertical axis represent a difference from the nominal point.



(a) Primary fluid mass flow rate



(b) Secondary fluid mass flow rate

Figure 5.11. : Comparison of ATTMO system performance with HCCD predicted performance: modulated control inputs.

ness of the ATTMO system by roughly 0.15 degrees Celsius. This is due to fidelity differences between the ATTMO cold plate model and the HCCD cold plate model; the HCCD model treats the surface of the cold plate as a lumped mass, whereas the ATTMO model discretizes the cold plate into a number of individual control volumes, both along the cold plate's length and width axes. The surface temperature $T_{w,a}$ being plotted for the ATTMO simulation results is the *maximum* surface temperature of all nodes of the cold plate. Because nodes closer to the application of the heat pulse along the width axis exhibit a higher temperature increase, it makes physical sense that the cold plate surface temperature will deviate farther away from the nominal in the ATTMO model than in the HCCD model where the lumped cold plate has a higher thermal capacitance.

The cold plate temperature transients for each heat pulse are shown in Figure 5.10c. Toward the end of the final transient pulse, or around time $t = 200$ seconds, there is a spike in the ATTMO model's cold plate surface temperature. When looking at the primary fluid mass flow rate commanded by the ATTMO system's controller (Fig. 5.11a), it is apparent that this spike corresponds to the initial onset of input saturation. In other words, the primary fluid mass flow rate reaches its actuation limit during this heat pulse. While not applied in the simulations in this dissertation, I have developed a method for embedding notions of input saturation into the H_∞ control formulation studied here. Initial results indicate the proposed formulation results in controllers that are often less reactive during transients. Therefore, this type of formulation could be used in the SL CCD algorithm's inner loop to incorporate notions of input saturation into the HCCD algorithm. Details for the formulation can be found in the Appendix.

Several important observations can be made from the results presented in this chapter. These observations, which are summarized below, provide important insights for future research direction.

Remark *8.* In practice, a design engineer can use the proposed HCCD algorithm to select components and an overall design for a physical system. Without custom-ordered components, however, which can be expensive and come with increased lead time, it is difficult to select components with the exact desired designs. Therefore, some type of model tuning will typically be required to ensure the model used to program a model-based controller accurately depicts the dynamics of the physical system. The work in this chapter provides a systematic method of model tuning which can be used in this type of potential experimental work.

Remark 9. Due to the nature of entropy generation, a state describing these dynamics will typically be undetectable. Therefore, in practice, a model-based estimation of the system's entropy generation rate will be required. The work in this chapter also provides a method of designing an open-loop estimator for use with any state-feedback control method. This technique can easily be leveraged for future experimental work.

To summarize, the value of the work in this chapter is twofold. First, it is a necessary initial step in validation of the proposed HCCD algorithm and its utility. The efforts here provide initial validation that the proposed HCCD algorithm's solution can translate to higher fidelity modeling software, thereby moving HCCD a step closer to the experimental realm. Second, the steps taken to validate the HCCD solution in ATTMO provided important insight into challenges that would be encountered when building a physical system and programming the feedback controller. Therefore, this work can be utilized in future research aimed at experimentally validating the proposed approach. Future research directions will be discussed in more detail in the following chapter.

6. CONCLUSIONS

6.1 Summary of Research Contributions

In this dissertation, I developed a new design approach for thermal management systems (TMSs) based on the concept of combined plant and control design, or control co-design (CCD), to enable improved transient performance and efficiency for TMSs. More specifically, I developed a hierarchical CCD (HCCD) algorithm aimed at optimizing a full system design of a TMS for transient performance and robustness. The algorithm integrates system level (SL) CCD with detailed component level (CL) design optimization. The lower-fidelity SL CCD algorithm incorporates feedback control into the design of a TMS to ensure controllability and robust transient response to exogenous disturbances and the higher-fidelity CL design optimization algorithms provide a way of designing detailed components to achieve the desired performance needed at the SL.

My SL CCD algorithm for dynamic thermal management explicitly leverages system efficiency, via entropy generation rate, as a design metric for both steady-state and transient performance. This was accomplished in part by structuring the algorithm to utilize a *parameterized first-principles model* that defines entropy generation rate as a state whose dynamics are coupled to the plant and control variables. The result of the proposed CCD algorithm is a system design that is not only optimized for steady-state efficiency, but that can be designed for robustness to transient disturbances while achieving said disturbance rejection with minimal compromise to system efficiency. My component level design optimization algorithms allow optimization of specific physical variables too complex for inclusion in a CCD algorithm. For example, my design methodology for a shell and tube heat exchanger allows optimization of a detailed set of design variables to meet various static and transient objectives in the presence of specific component design constraints.

I presented a case study to illustrate the use of the HCCD architecture in designing robust TMSs with different design objectives at both the system and component levels. The software package ATTMO was used to test a solution to the proposed HCCD algorithm on high-fidelity thermal modeling software. I showed that the resulting specifications from the HCCD algorithm could be used to select components and design the controller in ATTMO while achieving temperature regulation close to that achieved by the system level HCCD model. Future work is described in the following section.

6.2 Future Research Direction

This thesis focused on developing a general framework for HCCD of transient systems. However, the proposed algorithm also lays the groundwork for multiple impactful future research directions. Important future research should include (1) extended validation of the proposed HCCD approach, (2) investigation of more complex hierarchical architectures, and (3) examination of hierarchical control synthesis and alternative feedback control methods for given systems. Each of these directions would not only expand the applicability of the HCCD approach, but also demonstrate its physical efficacy in designing and controlling integrated transient systems.

Further validation efforts for the HCCD approach should be twofold. First, the validation should involve building the optimal components, and/or modifying existing components to fit an optimal solution to a HCCD case study. This would also enable quantification of the performance benefits of the HCCD process over conventional design processes, particularly in terms of achieving real-time performance specifications and disturbance rejection. Second, future validation work should consider the study of more realistic and complex case studies, particularly those with well-defined mission profiles and/or design constraints. For example, it is typical for design specifications to include upper bounds on footprint or mass for individual components or an entire system; the proposed HCCD algorithm could be modified with a cost function that solely penalizes transient performance and treats mass and footprint as constraints rather than as additional costs to be weighted in the overall objective function. Similarly, while this thesis focused on what would be analogous to cruise operation, further study is needed to consider how the algorithm can scale to handle design of systems across highly-transient operating conditions such as take-off or landing in an aircraft context.

The proposed HCCD algorithm also lays the groundwork to begin investigating more complex hierarchies. While this thesis focused on integration of the system level and component level, future work should look at extending the hierarchy to incorporate more complexity. For example, within an aircraft, the fuel TMS is one system that makes up the larger architecture of the aircraft. The TMS must also interface with other systems such as the aircraft's propulsion system. The hierarchy could be extended to optimize design and control of multiple systems in the overall aircraft architecture. This would help illustrate how efficiency savings in the TMS would propagate to other systems on board. For example, a reduction in entropy generation within the fuel TMS would ultimately correlate to less required fuel for the propulsion system and thus less required mass on board the aircraft. Additional complexity could also be added to the TMS itself. As an example, a second heat sink could be added as part of an additional vapor compression system (VCS) on board the aircraft. The extra heat sink would provide options of either dumping heat to the original sink or to the additional aircraft VCS. In practice, this type of architecture is often utilized to accommodate different flight conditions [27] such as the lower altitudes encountered during take-off.

Extending the hierarchy to incorporate more complex architectures also facilitates the need for research on incorporating hierarchical control into the HCCD framework. For example, the aircraft propulsion system would require its own feedback controller which may have its own control objectives and need to operate on a different time scale than the controller for the fuel TMS. Therefore, the HCCD framework would lend itself well to implementing hierarchical feedback control. Similarly, future research could involve examining the use of different control algorithms within the SL CCD algorithm for the fuel TMS. While this thesis used a time delay H_∞ control method to incorporate a robustness metric into the CCD plant objective function, the CCD formulation itself is algorithm-agnostic; in other words, many control algorithms could be used in the CCD inner loop as long as there is a connection to link the inner loop objective to the outer plant loop optimization. Therefore, future work could compare system performance for different inner loop control algorithms to determine approaches that enable the most favorable transient performance.

APPENDICES

A. CONTROL INPUT SATURATION

One problem with current design and control strategies for thermal management concerns the notion of input saturation. Control methods are often blind to, or cannot adapt to, input-related constraints such as actuator saturation limits. This is a problem in the context of CCD. When designing a TMS and its controller, it is desirable to guarantee that robust performance can be achieved within the actuator limits. Without some notion of actuator limits within either the model or control formulation, it is difficult to program a controller to be intelligent enough to respond within saturation limits.

Recall the two control inputs for the nonlinear dynamic TMS model in Eqn. (2.1) are the primary fluid mass flow rate $\dot{m}_c$ and the secondary fluid mass flow rate $\dot{m}_r$. Equation (A.1) can be used to incorporate notions of input saturation into the control strategy studied in this work:

$$\dot{m}_c = \frac{1}{2} \cdot \left(\underline{\dot{m}}_c + \bar{\dot{m}}_c + \left[(\dot{m}_{c,des} - \underline{\dot{m}}_c)^2 + \varepsilon^2 \right]^{1/2} - \left[(\dot{m}_{c,des} - \bar{\dot{m}}_c)^2 + \varepsilon^2 \right]^{1/2} \right) \tag{A.1a}$$

$$\dot{m}_r = \frac{1}{2} \cdot \left(\underline{\dot{m}}_r + \bar{\dot{m}}_r + \left[(\dot{m}_{r,des} - \underline{\dot{m}}_r)^2 + \varepsilon^2 \right]^{1/2} - \left[(\dot{m}_{r,des} - \bar{\dot{m}}_r)^2 + \varepsilon^2 \right]^{1/2} \right) \ . \tag{A.1b}$$

In Eqn. (A.1), the term $\dot{m}_{des}$ represents the desired, or commanded, flow rate and $\dot{m}$ represents the actual flow rate that would result based on actuator limits. The terms $\bar{\dot{m}}$ and $\underline{\dot{m}}$ are upper and lower saturation limits on flow rate, respectively. The variable ε is a user-defined scalar that helps mimic saturation effects with a single continuous curve.

As an example, Figure A.1 compares a continuous curve represented by Eqn. (A.1) with a nonlinear, *discontinuous*, saturation function.

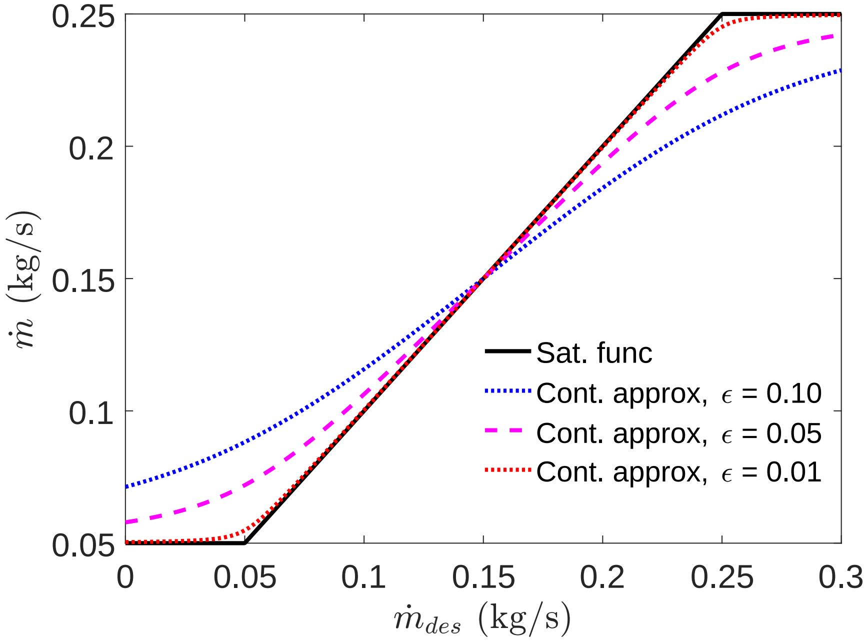


Figure A.1. : Comparison of saturation curve and continuous, linearizable approximation.

The horizontal axis in Fig. A.1 represents a desired, or commanded, flow rate such as that from a controller. The vertical axis represents the actual flow rate that would be produced based on actuation limits. The continuous curve in Eqn. (A.1) accurately approximates the discontinuous behavior of the saturation limits. In fact, as $\varepsilon \to 0$, the continuous curve approaches a perfect match with the discontinuous saturation behavior. While the approximation is very nonlinear near the saturation limits, it does provide a continuous, *and therefore easily linearizable*, curve for embedding saturation effects into the linear controller.

It is then trivial to rewrite the state equations for the dynamic TMS model as

$$\frac{dM_t}{dt} = -\dot{m}_E \tag{A.2a}$$

$$M_t \frac{dT_t}{dt} = \left(\dot{m}_c - \dot{m}_E\right)\left(T_r' - T_t\right) \tag{A.2b}$$

$$C_r \frac{dT_r}{dt} = A_{s,r}\alpha_r \left(T_{w,r} - T_r\right) + \left(\dot{m}_c - \dot{m}_E\right) c_p \left(T_a' - T_r\right) \tag{A.2c}$$

$$C_a \frac{dT_a}{dt} = A_{s,a}\alpha_a \left(T_{w,a} - T_a\right) + \dot{m}_c c_p \left(T_t' - T_a\right) \tag{A.2d}$$

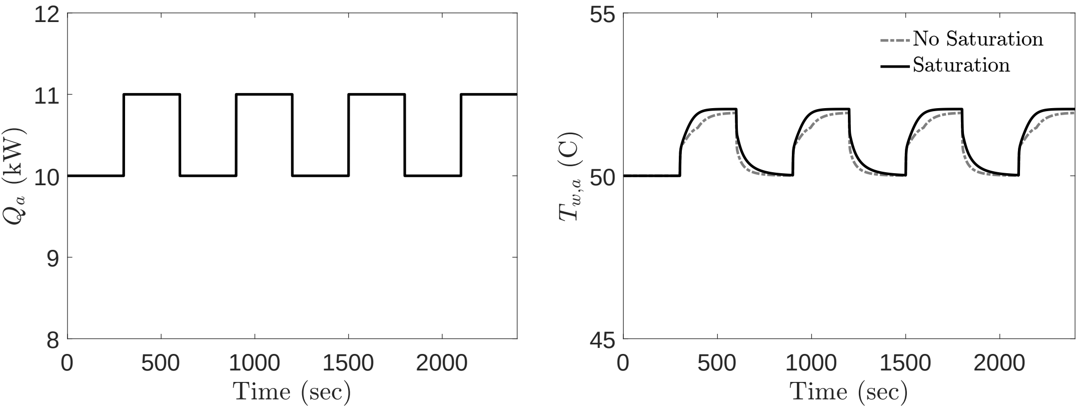
$$m_{w,r} c_{p,w} \frac{dT_{w,r}}{dt} = A_{s,r}\alpha_r \left(T_r - T_{w,r}\right) + \dot{Q}_{c\to w} \tag{A.2e}$$

$$m_{w,a} c_{p,w} \frac{dT_{w,a}}{dt} = A_{s,a}\alpha_a \left(T_a - T_{w,a}\right) + \dot{Q}_a \quad , \tag{A.2f}$$

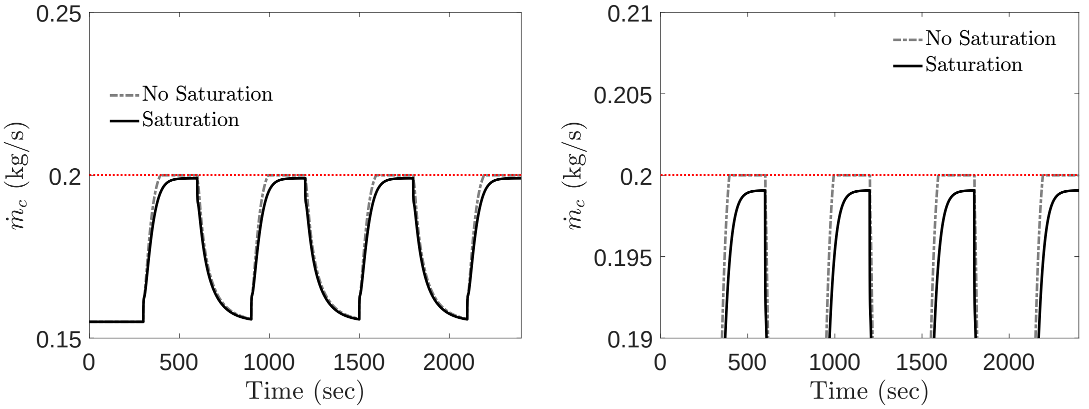
$$\dot{Q}_{c\to w} = \dot{m}_r c_{p,c} \left[T_c - \left(T_{w,r} + \left(T_c - T_{w,r}\right) e^{\frac{-\alpha_c A_{s,c}}{\dot{m}_r c_{p,c}}} \right) \right] . \tag{A.3}$$

where the continuous expressions to approximate saturation behavior are inserted into the dynamic equations. I can then re-linearize the set of state equations as described in Eqn. (2.14). I then have a linear system with A_0, A_1, B_1, and B_2 parameterized as functions of the saturation limits $\underline{\dot{m}}$ and $\bar{\dot{m}}$.

While this method of incorporating saturation limits was not included in the HCCD results presented in this dissertation, initial investigation into its utility has been promising. Figure A.2 illustrates the performance of two separate controllers: "No Saturation" and "Saturation." In the No Saturation case, the controller model is linearized without notion of the actuator limits. Conversely, in the Saturation case, the controller model is linearized using the continuous approximations presented here. In Figure A.2cA.2d, one can see that by incorporating the actuator limits into the linearization, the controller is less aggressive and seems to respond to the notion of the limits while achieving the same level of temperature regulation (Fig. A.2b).



(a) Exogenous heat pulses (b) Cold plate surface temperature



(c) Commanded primary fluid mass flow rate (d) Commanded primary fluid mass flow rate (zoomed in on limit)

Figure A.2. : Comparison of controller performance with and without knowledge of saturation limits.

B. HCCD MATLAB CODE

The following code is is an example of how to set up the HCCD algorithm.

```
clc
clear all
close all

% THIS IS THE TOP LEVEL FUNCTION TO RUN THE HCCD ALGORITHM.  NOTE ...
    THAT
% ALL QUANTITIES ARE IN BASE UNITS (i.e. K, kg, Pa, etc...)

final_str = 'optimal_system';

% Define weighting parameters for the SL and CL level algorithms
lam_m1 = 1*10^(0); lam_v1 = 0*10^(0); gam_wt1 = 1*10^(0); Sgen_wt1 = ...
    0*10^(0); % s.s. level
m_wt = 1; eval_wt1 = 0*10^(-1); ftp_wt = 0; % hex level
mpump_wt = 1; Sgenpump_wt = 0; satpump_wt = 0;

% initialize various setpoints needed  for the algorithm(s) to run
spec_Twa = 40 + 273.15; spec_Q = 5*1000; Tclow = 10 + 273.15; delT = ...
    2; eta_min = 0;
mce_ini = 0.2; mrame_ini = 0.2; eta_range = 0.75; run_multistart = 1;...
     maxfevals = 2500;
gam_flag = 1; const_linear_coeffs = 0; Tchigh = 20 + 273.15; ...
    pump_flow_FOS = 0.9;

% weighting matrices for the controller at s.s. level
num_ss_constr = 10;
```

```
wt_Twa = 0.01; C = [0,0,0,0,0,wt_Twa,0;0,0,0,0,0,0,wt_Twa/1000]; D12 ...
   = [0.5,0;0,0.5];

% run initial component optimizations first; initialize variables ...
   needed
% later with dummy values
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% initialize optimal component algorithms
iter = 1;
opt_vars_pump = [8;8]*0.0254; % initial cond for impeller diams
map_ss_to_pump.mce = mce_ini; map_ss_to_pump.mrame = mrame_ini;

Do_tube = 1/4*0.0254; Pt = 1.01*Do_tube; wt_tube = 1/16*0.0254; Nt = ...
   100;
L_tube = 2*12*0.0254;B = L_tube/(15+1); Ds_shell = 2.5*B;
opt_vars_hex = [Pt;Do_tube;L_tube;Nt;Ds_shell;B];

% run the initial pump optimization
[PumpData,opt_vars_pump,fval_pump] = run_entire_pump_opt(iter,...
    mpump_wt,Sgenpump_wt,map_ss_to_pump,eta_range,opt_vars_pump,...
       run_multistart,maxfevals);

% package various quantities needed to run the initial HX ...
   optimization
% (map SL variables to HX optimization)
PAF_finder_inputs.dummy_var = 1; last_PAFs = [1;1];
map_ss_to_hex.mce = mce_ini; map_ss_to_hex.mrame = mrame_ini;
map_ss_to_hex.absU1min = PumpData.absU1min; map_ss_to_hex.absU1max = ...
   PumpData.absU1max;
map_ss_to_hex.absram_max = PumpData.absram_max;
map_ss_to_hex.Tce = 1/2*(Tclow+Tchigh);  map_ss_to_hex.Tre = 30 + 273...
   .15;
```

```
Cpavg = refpropm('C','T',1/2*(spec_Twa+map_ss_to_hex.Tre),'P',101,'...
    water');
map_ss_to_hex.Tae = spec_Q/(Cpavg*mce_ini) + map_ss_to_hex.Tre;

% setup the HX optimization - i.e. package up inputs needed and find ...
    a
% feasible starting point
absU1max = PumpData.absU1max;
hex_inputs = setup_hex_opt(spec_Q,m_wt,eval_wt1,ftp_wt,map_ss_to_hex,...
    opt_vars_hex);



%%% Unpack what is needed from the setup function to actually run the...
     hx
%%% optimization
feas_vars = hex_inputs.feas_vars; P_tube = hex_inputs.P_tube; P_cool ...
    = hex_inputs.P_cool;
Tshell_in = hex_inputs.Tshell_in; mdot_shell = hex_inputs.mdot_shell;...
     mdot_tube = hex_inputs.mdot_tube;
Ref_tube = hex_inputs.Ref_tube; Ref_shell = hex_inputs.Ref_shell; Np ...
    = hex_inputs.Np; N = hex_inputs.N;
epsilon_tube = hex_inputs.epsilon_tube; Method = hex_inputs.Method; ...
    static_flag = hex_inputs.static_flag;
Nb = hex_inputs.Nb; spec_Q = hex_inputs.spec_Q; dynamic_flag = ...
    hex_inputs.dynamic_flag;
num_constraints = hex_inputs.num_constraints; eps_Q = ...
    hex_inputs.eps_Q; Pt = hex_inputs.Pt;
Do_tube = hex_inputs.Do_tube; Ttube_in = hex_inputs.Ttube_in; wt_tube...
     = hex_inputs.wt_tube;
nlrhs = hex_inputs.nlrhs; nle = hex_inputs.nle; LB = hex_inputs.LB; ...
    UB = hex_inputs.UB;

% run the initial HX optimization
des_pars = feas_vars;
```

```
nlcon = @(des_pars) constr_hex(des_pars,Tshell_in,mdot_shell,...
   mdot_tube,...
    Ref_tube,Ref_shell,Np,N,epsilon_tube,spec_Q,num_constraints,...
    Ttube_in,wt_tube,absU1max,map_ss_to_hex);
        fobj = @(des_pars) obj_hex(des_pars,Tshell_in,mdot_shell,...
           mdot_tube,...
            Ref_tube,Np,N,m_wt,...
            eval_wt1,ftp_wt,0,Ttube_in,wt_tube);

opts = optiset('display','iter','maxiter',1000,'maxfeval',maxfevals,'...
   maxtime',3600);
Opt = opti('obj',fobj,'nlmix',nlcon,nlrhs,nle,'x0',...
des_pars,'solver','nomad','bounds',LB,UB,'options',opts);
if run_multistart == 1
    [opt_vars_hex(1:6,iter),fval_hex(iter),exitflag,info] = ...
       multisolve(Opt,des_pars,[2 2]);
else
    [opt_vars_hex(1:6,iter),fval_hex,exitflag,info] = solve(Opt,...
       des_pars); exitflag_hex = exitflag;
end

% initial optimal variables for the HX
Pt = opt_vars_hex(1,iter); Do_tube = opt_vars_hex(2,iter);
L_tube = opt_vars_hex(3,iter); Nt = opt_vars_hex(4,iter);
Ds_shell = opt_vars_hex(5,iter); B = opt_vars_hex(6,iter);

% post process the HX to save some relevant outputs
hex_outputs = post_process_hex(wt_tube,opt_vars_hex(:,iter),Tshell_in...
   ,Ttube_in,...
    N,spec_Q,map_ss_to_hex.mce,map_ss_to_hex.mrame,m_wt,eval_wt1,...
       ftp_wt,absU1max,map_ss_to_hex);
rhow = hex_outputs.rhow; Chex = hex_outputs.Chex;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% while loop starts the HCCD algorithm; take initial optimal ...
   components and
% run SL CCD... iterate back and forth until convergence

% initialize the convergence criterion and iteration counter
iter = 1; delJ_ss = 20; delJ_hex = 20; delJ_pump = 20;
stop_tol_ss = 0.5; stop_tol_hex = 0.5; stop_tol_pump = 0.5;
ss_conv = 100; hex_conv = 100; pump_conv = 100;

opt_vars_ss = 1; convergence = 0;
while convergence == 0 && iter < 100

    % first, map relevant quantities from CL algorithms to SL
    imp_diams = opt_vars_pump(:,end);
    pump_to_ss = map_pump_to_ss(imp_diams,map_ss_to_pump.mce,...
       map_ss_to_pump.mrame,eta_range);

    [hex_to_ss,last_PAFs] = map_hex_to_ss(Do_tube,wt_tube,Pt,Ds_shell...
       ,Chex,B,rhow,Ttube_in,L_tube,Nt,...
        mdot_tube,iter,hex_outputs,PAF_finder_inputs,Tclow,spec_Q,...
           spec_Twa,delT,pump_to_ss,mce_ini,mrame_ini);

    % RUN THE SL ALGORITHM

    % first, find a feasible starting point and output quantities ...
       needed to
    % run the SL CCD
    ss_inputs = setup_ss_opt(spec_Twa,spec_Q,Tclow,C,D12,hex_to_ss,...
       iter,opt_vars_ss,...
        num_ss_constr,pump_to_ss,mce_ini,mrame_ini,pump_flow_FOS,...
           const_linear_coeffs,Tchigh);
    feas_vars = ss_inputs.feas_vars; absU1min = ss_inputs.absU1min;
```

```
absram_max = ss_inputs.absram_max; absU1max = ss_inputs.absU1max;...
     params = ss_inputs.params;
LB = ss_inputs.LB; UB = ss_inputs.UB; nlrhs = ss_inputs.nlrhs; ...
    nle = ss_inputs.nle;

% run SL CCD optimization
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

des_pars = feas_vars; opt_flag = 1; feas_flag = 0;
opts = optiset('display','iter','maxiter',2500,'maxfeval',...
    maxfevals,'maxtime',3600);
nlcon = @(des_pars) constr_ss(des_pars,params,C,D12,absU1min,...
Tclow,spec_Twa,spec_Q,1,absram_max,absU1max,hex_to_ss,...
    num_ss_constr,...
pump_flow_FOS,const_linear_coeffs,Tchigh);
fobj = @(des_pars) obj_ss(des_pars,params,C,D12,lam_m1,lam_v1,...
    gam_wt1,absU1min,...
spec_Twa,spec_Q,1,Sgen_wt1,feas_flag,gam_flag,absram_max,absU1max...
    ,hex_to_ss,pump_to_ss,const_linear_coeffs);
Opt = opti('obj',fobj,'nlmix',nlcon,nlrhs,nle,'x0',...
     des_pars,'solver','nomad','bounds',LB,UB,'options',opts);

[opt_vars_ss(1:7,iter),fval_ss(iter),exitflag,info] = solve(Opt,...
    des_pars); exitflag_ss = exitflag;

% evaluate convergence criterion for SL objectives
if iter > 1
    delJ_ss(iter) = sensitivity_ss(opt_vars_ss,params,C,D12,...
        lam_m1,lam_v1,gam_wt1,absU1min,...
    spec_Twa,spec_Q,Sgen_wt1,0,gam_flag,absram_max,absU1max,...
        hex_to_ss,...
    pump_to_ss,const_linear_coeffs);
end

% define quantities that will be used for model tuning after CL
% optimization
PAF_finder_inputs.last_opt_vars_ss   = opt_vars_ss(1:7,iter);
PAF_finder_inputs.params             = params;
PAF_finder_inputs.Tclow              = Tclow;
PAF_finder_inputs.spec_Twa           = spec_Twa;
PAF_finder_inputs.spec_Qa            = spec_Q;
PAF_finder_inputs.delT               = delT;

% save optimal SL plant variables
Lta = opt_vars_ss(1,iter); Lar = opt_vars_ss(2,iter); Lrt = ...
   opt_vars_ss(3,iter);
Ltot_a = opt_vars_ss(4,iter);
Mte = opt_vars_ss(5,iter);
mce = opt_vars_ss(6,iter); mrame = opt_vars_ss(7,iter);

% output some SL properties to save
ss_outputs = post_process_ss(opt_vars_ss(:,iter),params,ss_inputs...
   ,spec_Q,spec_Twa,C,D12,hex_to_ss,const_linear_coeffs);
K = ss_outputs.K; gam = ss_outputs.gam; nom_x = ss_outputs.nom_x;
nom_u = ss_outputs.nom_u; Sgene = ss_outputs.Sgene;

% re-optimize components if we haven't converged yet
if delJ_ss(end) ≤ stop_tol_ss && delJ_hex(end) ≤ stop_tol_hex && ...
   delJ_pump(end) ≤ stop_tol_pump
    convergence = 1;
elseif iter == 10
    convergence = 1;
end

    % print info to command window; run new iteration if not ...
       converged
clc
ss_data = fval_ss
hex_data = fval_hex
pump_data = fval_pump
```

```
if iter > 1
    delJ_ss
    delJ_hex
    delJ_pump
end

iter_str = sprintf('system_iter%d',iter);
save(iter_str);

iter = iter + 1;

% COMPONENT LEVEL OPTIMIZATIONS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


if convergence == 0

 % now, we must pass relevant constraints/vars from sl to cl
    map_ss_to_hex.mce = mce; map_ss_to_hex.mrame = mrame;
    map_ss_to_hex.Tce = nom_u(3);
    if map_ss_to_hex.Tce ≤ Tclow
        map_ss_to_hex.Tce = Tclow;
    end
    map_ss_to_hex.Tae = nom_x(4);
    map_ss_to_hex.Tre = nom_x(3);
    map_ss_to_hex.absU1min = ss_inputs.absU1min; ...
       map_ss_to_hex.absU1max = ss_inputs.absU1max;
    map_ss_to_hex.absram_max = ss_inputs.absram_max;
    map_ss_to_pump.mce = mce; map_ss_to_pump.mrame = mrame;

    % run the pump optimization
    [PumpData,opt_vars_pump,fval_pump(iter)] = ...
       run_entire_pump_opt(...
iter,mpump_wt,Sgenpump_wt,map_ss_to_pump,eta_range,...
    opt_vars_pump,run_multistart,maxfevals);

% evaluate pump convergence
if iter > 1
     delJ_pump(iter) = sensitivity_pump(opt_vars_pump,mce,...
         mrame,mpump_wt,Sgenpump_wt,eta_range);
else
     delJ_pump(iter) = sensitivity_pump([opt_vars_pump_ini,...
         opt_vars_pump],mce,mrame,mpump_wt,Sgenpump_wt,...
         eta_range);
end

% setup new HX optimization
absU1max = PumpData.absU1max;
hex_inputs = setup_hex_opt(spec_Q,m_wt,eval_wt1,ftp_wt,...
    map_ss_to_hex,opt_vars_hex);

%%% Unpack what is needed from the setup function to actually...
     run the hx
%%% optimization
feas_vars = hex_inputs.feas_vars; P_tube = hex_inputs.P_tube;...
     P_cool = hex_inputs.P_cool;
Tshell_in = hex_inputs.Tshell_in; mdot_shell = ...
    hex_inputs.mdot_shell; mdot_tube = hex_inputs.mdot_tube;
Ref_tube = hex_inputs.Ref_tube; Ref_shell = ...
    hex_inputs.Ref_shell; Np = hex_inputs.Np; N = ...
    hex_inputs.N;
epsilon_tube = hex_inputs.epsilon_tube; Method = ...
    hex_inputs.Method; static_flag = hex_inputs.static_flag;
Nb = hex_inputs.Nb; spec_Q = hex_inputs.spec_Q; dynamic_flag ...
    = hex_inputs.dynamic_flag;
num_constraints = hex_inputs.num_constraints; eps_Q = ...
    hex_inputs.eps_Q; Pt = hex_inputs.Pt;
    Do_tube = hex_inputs.Do_tube; Ttube_in = hex_inputs.Ttube_in;...
        wt_tube = hex_inputs.wt_tube;
    nlrhs = hex_inputs.nlrhs; nle = hex_inputs.nle; LB = ...
       hex_inputs.LB; UB = hex_inputs.UB;

    %%% run new iteration of HX optimization
    des_pars = feas_vars;
    nlcon = @(des_pars) constr_hex(des_pars,Tshell_in,mdot_shell,...
       mdot_tube,...
Ref_tube,Ref_shell,Np,N,epsilon_tube,spec_Q,num_constraints,...
Ttube_in,wt_tube,absU1max,map_ss_to_hex);
    fobj = @(des_pars) obj_hex(des_pars,Tshell_in,mdot_shell,...
       mdot_tube,...
        Ref_tube,Np,N,m_wt,...
        eval_wt1,ftp_wt,0,Ttube_in,wt_tube);

    opts = optiset('display','iter','maxiter',1000,'maxfeval',...
       maxfevals,'maxtime',3600);
    Opt = opti('obj',fobj,'nlmix',nlcon,nlrhs,nle,'x0',...
    des_pars,'solver','nomad','bounds',LB,UB,'options',opts);
    if run_multistart == 1
        [opt_vars_hex(1:6,iter),fval_hex(iter),exitflag,info] = ...
           multisolve(Opt,des_pars,[2 2]);
    else
        [opt_vars_hex(1:6,iter),fval_hex(iter),exitflag,info] = ...
           solve(Opt,des_pars); exitflag_hex = exitflag;
    end

    % save new optimal hex variables
    Pt = opt_vars_hex(1,iter); Do_tube = opt_vars_hex(2,iter);
    L_tube = opt_vars_hex(3,iter); Nt = opt_vars_hex(4,iter);
    Ds_shell = opt_vars_hex(5,iter); B = opt_vars_hex(6,iter);

    % evaluate HX convergence
    if iter > 1
```

```
            delJ_hex(iter) = sensitivity_hex(opt_vars_hex,Tshell_in,...
               mdot_shell,mdot_tube,...
            Ref_tube,Np,N,m_wt,...
            eval_wt1,ftp_wt,Ttube_in,wt_tube);
        else
            delJ_hex(iter) = sensitivity_hex([opt_vars_hex_ini,...
               opt_vars_hex],Tshell_in,mdot_shell,mdot_tube,...
            Ref_tube,Np,N,m_wt,...
            eval_wt1,ftp_wt,Ttube_in,wt_tube);
        end

        % output some HX info based on new optimal component
        hex_outputs = post_process_hex(wt_tube,opt_vars_hex(:,iter),...
           Tshell_in,Ttube_in,...
    N,spec_Q,map_ss_to_hex.mce,map_ss_to_hex.mrame,m_wt,eval_wt1,...
       ftp_wt,absU1max,map_ss_to_hex);
        rhow = hex_outputs.rhow; Chex = hex_outputs.Chex;

    end

end
save(final_str); % save the final optimal components and system/...
   controller

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

The following function illustrates how to set up the obective function and constraint function evaluations for the heat exchanger optimization. Note that similar functions are designed for both the system level CCD algorithm and the component level pump algorithm.

```
function J = obj_hex(des_pars,Tshell_in,mdot_shell,mdot_tube,...
    Ref_tube,Np,N,m_wt,eval_wt1,ftp_wt,feas_flag,...
Ttube_in,wt_tube)

% This function computes the HX algorithm objective function value (J...
   ) for
% a given set of HX decision variables (des_pars).

% current decision variables
Pt = des_pars(1); Do_tube = des_pars(2);
L_tube = des_pars(3); Nt = des_pars(4);
Ds_shell = des_pars(5); B = des_pars(6);

% get HX geometric quantities needed
Cond = get_hex_params(Np,1,N,Pt,Do_tube,wt_tube,L_tube,Nt,B,Ds_shell)...
   ;
PAF_COOL = Cond.PAF.PAF_COOL; PAF_SH = Cond.PAF.PAF_SH;
PAF_TP = Cond.PAF.PAF_TP; PAF_SC = Cond.PAF.PAF_SC;
P_cool = Cond.Par.P_cool; Pt = Cond.Par.Pt; Di_tube = ...
   Cond.Par.Di_tube;
wt_tube = Cond.Par.wt_tube; L_tube = Cond.Par.L_tube; B = Cond.Par.B;
Nb = Cond.Par.Nb; Nt = Cond.Par.Nt; Np = Cond.Par.Np; Ds_shell = ...
   Cond.Par.Ds_shell;
Cp_wall = Cond.Par.Cp_wall; Do_tube = Cond.Par.Do_tube;  Chex = ...
   Cond.Par.Chex;
Ac_cool = Cond.Par.Ac_cool; As_c = Cond.Par.As_cool; As = ...
   Cond.Par.As_ref;
Ac_ref = Cond.Par.Ac_ref; m_wall = Cond.Par.M_wall; Vol_ref = ...
   Cond.Par.Vol_ref;
Do_tube = Cond.Par.Do_tube; De_shell = Cond.Par.De_shell;

% here, evaluate eigenvalues if we're placing penalty on transient
% performance
if feas_flag ≠ 1
    if eval_wt1 > 0
```

```
Cond = get_hex_params(Np,0,N,Pt,Do_tube,wt_tube,L_tube,Nt,B,...
    Ds_shell);
PAF_COOL = Cond.PAF.PAF_COOL; PAF_SH = Cond.PAF.PAF_SH;
PAF_TP = Cond.PAF.PAF_TP; PAF_SC = Cond.PAF.PAF_SC;
P_cool = Cond.Par.P_cool; Pt = Cond.Par.Pt; Di_tube = ...
    Cond.Par.Di_tube;
wt_tube = Cond.Par.wt_tube; L_tube = Cond.Par.L_tube; B = ...
    Cond.Par.B;
Nb = Cond.Par.Nb; Nt = Cond.Par.Nt; Np = Cond.Par.Np; ...
    Ds_shell = Cond.Par.Ds_shell;
Cp_wall = Cond.Par.Cp_wall; Do_tube = Cond.Par.Do_tube;  Chex...
     = Cond.Par.Chex;
Ac_cool = Cond.Par.Ac_cool; As_c = Cond.Par.As_cool; As = ...
    Cond.Par.As_ref;
Ac_ref = Cond.Par.Ac_ref; m_wall = Cond.Par.M_wall; Vol_ref =...
     Cond.Par.Vol_ref;
Do_tube = Cond.Par.Do_tube; De_shell = Cond.Par.De_shell;

Ref = Ref_tube;
n = 1; num_passes = Np;

% desired nominal inputs
mdot_t = mdot_tube;
P_ref_in = 101*1000; Tsat = refpropm('T','P',P_ref_in/1000,'Q...
    ',0,Ref);
T_ref_in = Ttube_in;

h_in = refpropm('H','T',T_ref_in,'P',P_ref_in/1000,Ref); % kJ...
    /kg
T_c_in = Tshell_in;
mdot_s = mdot_shell;
nom_u = [mdot_t;h_in;T_c_in;mdot_s];

% desired nominal states
P_ref0 = P_ref_in;
```

```
        H_ref0 = h_in*ones(N,1);
        T_wall0 = (1/2*(T_ref_in + T_c_in))*ones(N,1);
        nom_x0 = [H_ref0;T_wall0];

        % form the desired nominal vector
        Aeq = []; Beq = [];
        options = optimoptions('fmincon','Algorithm','interior-point'...
           ,...
            'Display','off','MaxFunEvals',1,'MaxIter',1,'TolFun',1e...
               -6,'TolX',1e-6);
        [solved_nomX,fval,exitflag,output]  = fmincon(...
           @get_hex_nominal,nom_x0,[],[],Aeq,Beq,0.85*nom_x0,1.15*...
           nom_x0,[],options,nom_u,Cond,Ref,N,P_ref0);
        nom_x = solved_nomX;

        [Amat,Bmat] = linearize_hex(nom_x,nom_u,Cond,Ref,N,P_ref0);
        evals = eig(Amat);
        max_eval1 = max(real(evals));

        nan_flag = 0;
        for o = 1:numel(Amat)
            if isnan(Amat(o)) == 1
                nan_flag = nan_flag + 1;
            end
        end
        if nan_flag > 0
            max_eval1 = 10^(-4);
        end

        if isnan(max_eval1) == 1 || max_eval1 == 0
            max_eval1 = .0001;
        end

    else
        max_eval1 = 1;
    end
end

% evaluate the objective function value
static_L = L_tube;
static_Ds = Ds_shell;
footprint = static_L*static_Ds;

wt_shell = 2*wt_tube;
Do_shell = static_Ds + 2*wt_shell; rhow = 8940;
m_shell = rhow*pi/4*static_L*(Do_shell^2-static_Ds^2);
m_wall = rhow*static_L*(Do_tube^2 - (Do_tube-2*wt_tube)^2)*pi/4*Nt;

m_tot = (m_wall + m_shell);

if feas_flag == 1
    J = 0;
else
    J = m_wt*m_tot + eval_wt1*((1/max_eval1)^2) + ftp_wt*footprint;
end

end
```

```
function [c] = constr_hex(des_pars,Tshell_in,mdot_shell,mdot_tube,...
    Ref_tube,Ref_shell,Np,N,epsilon_tube,spec_Q,...
    num_constraints,Ttube_in,wt_tube,absU1max,map_ss_to_hex)

% This function is used to evaluate feasibility (constraints) for the...
     CL HX
% algorithm

%%% The output is "c" which is the constraint evaluation vector; note...
     that
```

```
%%% constraints ≤ 0 are feasible

c = zeros(num_constraints,1);

% current decision variables values
Pt = des_pars(1); Do_tube = des_pars(2);
L_tube = des_pars(3); Nt = des_pars(4);
B = des_pars(6);
Ds_shell = des_pars(5);

% get HX geometry for current decision variables
Cond = get_hex_params(Np,1,N,Pt,Do_tube,wt_tube,L_tube,Nt,B,Ds_shell)...
   ;
PAF_COOL = Cond.PAF.PAF_COOL; PAF_SH = Cond.PAF.PAF_SH;
PAF_TP = Cond.PAF.PAF_TP; PAF_SC = Cond.PAF.PAF_SC;
P_cool = Cond.Par.P_cool; Pt = Cond.Par.Pt; Di_tube = ...
   Cond.Par.Di_tube;
wt_tube = Cond.Par.wt_tube; L_tube = Cond.Par.L_tube; B = Cond.Par.B;
Nb = Cond.Par.Nb; Nt = Cond.Par.Nt; Np = Cond.Par.Np; Ds_shell = ...
   Cond.Par.Ds_shell;
Cp_wall = Cond.Par.Cp_wall; Do_tube = Cond.Par.Do_tube;  Chex = ...
   Cond.Par.Chex;
Ac_cool = Cond.Par.Ac_cool; As_c = Cond.Par.As_cool; As = ...
   Cond.Par.As_ref;
Ac_ref = Cond.Par.Ac_ref; m_wall = Cond.Par.M_wall; Vol_ref = ...
   Cond.Par.Vol_ref;
Do_tube = Cond.Par.Do_tube; De_shell = Cond.Par.De_shell;

% call hex_design_calcs function which will evaluate constraints for
% current decision vars
[Qhot,Qcold,Qhx,c] = hex_design_calcs(des_pars,Tshell_in,mdot_shell,...
   mdot_tube,...
    Ref_tube,Ref_shell,Np,N,epsilon_tube,spec_Q,Ttube_in,wt_tube,...
       absU1max,map_ss_to_hex,num_constraints);
    c = c';

end
```

The following function solves the SL CCD inner loop optimization problem for the H_∞ control problem.

```
function [K,successes,gam] = solve_inner_loop(A0,A1,A2,B1,B2,C,D12,h1...
    ,h2)

% This function solves the SL CCD inner loop for optimal Hinf ...
    controller
% gains (K) and linking variable (gam)

% dimensions of system
n = size(A0,1); l = size(B2,2); p = size(C,1); r = size(D12,1); q = ...
    size(B1,2);
dummy = h2;

% define the decision variables
Q1 = sdpvar(n,n);
U1 = sdpvar(n,n);
U2 = sdpvar(n,n);
Q2 = sdpvar(n,n,'full');
Q3 = sdpvar(n,n,'full');
R1 = sdpvar(n,n,'full');
R2 = zeros(size(R1));
Y = sdpvar(l,n,'full');
Beta = sdpvar(1,1,'full');

F1 = zeros(size(R2,1),size(U1,1));
F2 = zeros(size(R2,1),size(U2,1));
```

```
% create the lmi constraint
sig = Q3 - Q2' + Q1*(A0' + A1' + A2') + Y'*B2';

lmi = blkdiag(Q2+Q2',-Q3-Q3',-Beta*eye(size(B1,2)),-h1*R1,...
    -eye(size(Q1*C',2)),-eye(size(Y'*D12',2)),-U1,-U1,-U2,-U2,-h1*R1)...
       ;

lmi(1:size(Q2,1),:) = [Q2+Q2',sig,zeros(size(sig,1),size(B1,2)+size(...
   R1,2)),...
    Q1*C',Y'*D12',zeros(size(sig,1),size(F1*U1,2)),Q2',zeros(size(sig...
       ,1),size(F2*U2,2)),...
    Q2',h1*Q2'*A1'];

lmi(size(Q2,1)+1:size(Q2,1)+size(Q3,1),:) = [sig',-Q3-Q3',B1,h1*R1,...
   ...
    zeros(size(R2,1),size(Y'*D12',2)+size(Q1*C',2)),...
    F1*U1,Q3',F2*U2,Q3',h1*Q3'*A1'];

for j = 1:size(lmi,1)
    lmi(j:size(lmi,1),j) = lmi(j,j:size(lmi,2))';
end

constr = [lmi ≤ -eps*eye(size(lmi)); Q1 ≥ eps*eye(size(Q1)); Beta ≥ ...
   eps];
cost = [Beta];
ops = sdpsettings('solver','bmibnb','verbose',0);
solved_sys = optimize(constr,cost,ops);

Y = value(Y);
Q1 = value(Q1);
Beta = value(Beta);
gam = sqrt(Beta);
K = Y*inv(Q1);

if solved_sys.problem == 0
```

```
    successes = 1;
    Y = value(Y);
    Q1 = value(Q1);
    Beta = value(Beta);
    gam = sqrt(Beta);
    K = Y*inv(Q1);
else
    successes = 0;
    gam = 10^5;
    K = zeros(size(B2,2),size(A0,1));
end

dummy = 1;

end
```

The following function illustrates the process for linearizing the system level model. A similar process is used to linear the HX dynamic model.

```
function [A0,A1,B,Bd,nc] = linearize_ss(nom_x,nom_u,params,dparam)

% This function computes the LTI state-space matrices for a given ...
   nominal
% condition.

% Outputs are the matrices and the ctrb.
% matrix rank.

Cp = params.Cp; D = params.D; Ac = params.Ac; rho = params.rho; Cpw =...
    params.Cpw;
mu = params.mu; T0 = params.T0; beta = params.beta; k = params.k;
Cp_c = params.Cp_c; k_c = params.k_c; mu_c = params.mu_c;
Di = params.Di; wt = params.wt; Do = params.Do;
```

```
Aca = params.Aca; a1 = params.a1;
a2 = params.a2; a3 = params.a3; a4 = params.a4; a5 = params.a5;
a6 = params.a6; rhow = params.rhow;
cpf_p = params.cpf_p; cpf_c = params.cpf_c;

Lta = dparam.Lta; Lar = dparam.Lar; Lrt = dparam.Lrt;
Ltot_r = dparam.Ltot_r; Ltot_a = dparam.Ltot_a;
Cr = dparam.Cr;
Ca = dparam.Ca; Vwr = dparam.Vwr; Vwa = dparam.Vwa; Mwa = dparam.Mwa;
Mwr = dparam.Mwr; Ac_c = dparam.Ac_c;
As_c = dparam.As_c;
Asa = dparam.Asa; Asr = dparam.Asr;
hma1 = dparam.hma1; hma2 = dparam.hma2;
hm_c1 = dparam.hm_c1; hm_c2 = dparam.hm_c2;
hmr1 = dparam.hmr1; hmr2 = dparam.hmr2;

% x = [Mt, Tt, Tr, Ta, Twr, Twa, Zeta];
% xd = [0, Ttd, Trd, Tad, 0, 0, 0];
% u = [mc, mram]; d = [me, Qa, Tc, r];

Mt = nom_x(1);
Tt = nom_x(2); Tt_delay = Tt;
Tr = nom_x(3); Tr_delay = Tr;
Ta = nom_x(4); Ta_delay = Ta;
Twr = nom_x(5);
Twa = nom_x(6);
Zeta = nom_x(7);

mc = nom_u(1);
me = nom_u(2);
Tc = nom_u(3); % Tbr = Twr;
Qa = nom_u(4); % Tba = Twa;
mram = nom_u(5);
r = nom_u(6);
```

```
Z1 = eye(7);
Z1(3,3) = Cr;
Z1(4,4) = Ca;
Z1(5,5) = Mwr*Cpw;
Z1(6,6) = Mwa*Cpw;

A0mat = zeros(7,7);
A0mat(2,1) = (mc - me) * (-Tr_delay + Tt) / Mt ^ 2;
A0mat(2,2) = (-mc + me) / Mt;
A0mat(2,3) = 0; % Tr_delay
A0mat(3,3) = ((-mc + me) * hmr1 - hmr2) * Asr - (mc - me) * Cp;
A0mat(3,4) = 0; % Ta_delay
A0mat(3,5) = Asr * (hmr1 * (mc - me) + hmr2);
A0mat(4,2) = 0; % Tt_delay
A0mat(4,4) = (-hma1 * mc - hma2) * Asa - mc * Cp;
A0mat(4,6) = Asa * (hma1 * mc + hma2);
A0mat(5,3) = Asr * (hmr1 * (mc - me) + hmr2);
A0mat(5,5) = mram * Cp_c * exp(-(hm_c1 * mram + hm_c2) * As_c / mram ...
   / Cp_c) + ((-mc + me) * hmr1 - hmr2) * Asr - mram * Cp_c;
A0mat(6,4) = Asa * (hma1 * mc + hma2);
A0mat(6,6) = - Asa * (hma1 * mc + hma2);
A0mat(7,2) = Cp * (mc - me) * (-Tr_delay + Tt) / Tt ^ 2;
A0mat(7,3) = (-(hmr1 * (mc - me) + hmr2) * Asr * Twr ^ 2 + Cp * (mc -...
    me) * (Tr - Ta_delay) * Twr + (hmr1 * (mc - me) + hmr2) * Tr ^ 2...
    * Asr) / Tr ^ 2 / Twr;
A0mat(7,4) = (-Asa * (hma1 * mc + hma2) * Twa ^ 2 + Cp * mc * (Ta - ...
   Tt_delay) * Twa + Asa * (hma1 * mc + hma2) * Ta ^ 2) / Ta ^ 2 / ...
   Twa;
A0mat(7,5) = -Asr * (hmr1 * (mc - me) + hmr2) * (Tr - Twr) * (Tr + ...
   Twr) / Tr / Twr ^ 2;
A0mat(7,6) = -Asa * (Ta - Twa) * (Ta + Twa) * (hma1 * mc + hma2) / Ta...
    / Twa ^ 2;
A0 = Z1\A0mat;

A1mat = zeros(7,7);
```

```
A1mat(2,3) = (mc - me) / Mt;
A1mat(3,4) = (mc - me) * Cp;
A1mat(4,2) = mc * Cp;
A1mat(7,2) = -Cp * mc * (Ta - Tt_delay) / Ta / Tt_delay;
A1mat(7,3) = -Cp * (mc - me) * (-Tr_delay + Tt) / Tt / Tr_delay;
A1mat(7,4) = -Cp * (mc - me) * (-Ta_delay + Tr) / Tr / Ta_delay;
A1 = Z1\A1mat;

% u = [mc,mram]
Bmat = zeros(7,2);
Bmat(2,1) = (Tr_delay - Tt) / Mt;
Bmat(3,1) = -Asr * (Tr - Twr) * hmr1 - Cp * (Tr - Ta_delay);
Bmat(4,1) = -Asa * (Ta - Twa) * hma1 - Cp * (Ta - Tt_delay);
Bmat(5,1) = Asr * hmr1 * (Tr - Twr);
Bmat(5,2) = -((As_c * hm_c2 + mram * Cp_c) * exp(-(hm_c1 * mram + ...
    hm_c2) * As_c / mram / Cp_c) - mram * Cp_c) * (Tc - Twr) / mram;
Bmat(6,1) = Asa * hma1 * (Ta - Twa);
Bmat(7,1) = (-Cp * Tr * Twr * Tt * Ta * Twa * log(Ta_delay) - Cp * Tr...
     * Twr * Tt * Ta * Twa * log(Tr_delay) - Cp * Tr * Twr * Tt * Ta ...
    * Twa * log(Tt_delay) + Cp * Tr * Twr * Tt * Ta * Twa * log(Ta) +...
     Cp * Tr * Twr * Tt * Ta * Twa * log(Tr) + Cp * Tr * Twr * Tt * ...
    Ta * Twa * log(Tt) + Asa * Tr * Tt * Twa ^ 2 * Twr * hma1 + (Twr ...
    ^ 2 * hmr1 * Ta * Tt * Asr + ((((-0.2e1 * Asa * hma1 - 0.2e1 * ...
    Asr * hmr1 + (2 * a1 * mc) - 0.3e1 * Cp + a2) * Tr + Cp * ...
    Ta_delay) * Ta + Cp * Tt_delay * Tr) * Tt + Cp * Tr_delay * Ta * ...
    Tr) * Twr + hmr1 * Ta * Tr ^ 2 * Tt * Asr) * Twa + Asa * hma1 * ...
    Tr * Twr * Tt * Ta ^ 2) / Tr / Twr / Tt / Ta / Twa;
Bmat(7,2) = 2 * a4 * mram + a5;
B = Z1\Bmat;

% d = [me,Qa,Tc,r]
Bdmat = zeros(7,4);
Bdmat(1,1) = -1;
Bdmat(2,1) = (-Tr_delay + Tt) / Mt;
Bdmat(3,1) = Asr * (Tr - Twr) * hmr1 + Cp * (Tr - Ta_delay);
Bdmat(5,1) = -Asr * hmr1 * (Tr - Twr);
Bdmat(5,3) = -mram * Cp_c * (-0.1e1 + exp(-(hm_c1 * mram + hm_c2) * ...
   As_c / mram / Cp_c));
Bdmat(6,2) = 1;
Bdmat(7,1) = (Cp * Tr * Twr * Tt * log(Ta_delay) + Cp * Tr * Twr * Tt...
    * log(Tr_delay) - Cp * Tr * Twr * Tt * log(Tr) - Cp * Tr * Twr *...
    Tt * log(Tt) - Asr * Tt * Twr ^ 2 * hmr1 + (((0.2e1 * Asr * hmr1...
    + 0.2e1 * Cp) * Tr - Cp * Ta_delay) * Tt - Cp * Tr * Tr_delay) *...
    Twr - Asr * hmr1 * Tr ^ 2 * Tt) / Tr / Twr / Tt;
Bdmat(7,2) = 0;
Bdmat(7,3) = 0;
Bdmat(7,4) = -1;
Bd = Z1\Bdmat;

nc = rank(ctrb(A0+A1,B));
end
```

REFERENCES

REFERENCES

[1] J. Doty, K. Yerkes, L. Byrd, J. Murthy, A. Alleyne, M. Wolff, S. Heister, and T. S. Fisher. Dynamic thermal management for aerospace technology: review and outlook. *Journal of Thermophysics and Heat Transfer*, 31(1):86–98, January 2017.

[2] Wei Huang, Mircea R. Stan, Karthik Sankaranarayanan, Robert J. Ribando, and Kevin Skadron. Many-core design from a thermal perspective. In *Design Automation Conference, 2008. DAC 2008. 45th ACM/IEEE*, pages 746–749. IEEE, 2008.

[3] Srinivas Garimella, Ashlie M. Brown, and Ananda Krishna Nagavarapu. Waste heat driven absorption/vapor-compression cascade refrigeration system for megawatt scale, high-flux, low-temperature cooling. *International Journal of Refrigeration*, 34(8):1776–1785, December 2011.

[4] Mark Bodie, Greg Russell, Kevin McCarthy, Eric Lucas, Jon Zumberge, and Mitch Wolff. Thermal analysis of an integrated aircraft model. *AIAA Paper*, 20:0–288, 2010.

[5] G.C. Goodwin, S.F. Graebe, and M.E. Salgado. *Control system design*. Prentice Hall, 2001.

[6] H. Fathy, P. Y. Papalambros, and A. G. Ulsoy. On combined plant and control optimization. In *8th Cairo University International Conference on Mechanical Design and Production*. Cairo University Cairo, Egypt, 2004.

[7] Rakesh Patil, Zoran Filipi, and Hosam Fathy. Computationally efficient combined design and control optimization using a coupling measure. *IFAC Proceedings Volumes*, 43(18):144–151, 2010.

[8] Yu Jiang, Yebin Wang, Scott A. Bortoff, and Zhong-Ping Jiang. An iterative approach to the optimal co-design of linear control systems. *International Journal of Control*, 89(4):680–690, April 2016.

[9] Vineet Sahasrabudhe, Roberto Celi, Andr, and L. Tits. Integrated Rotor-Flight Control System Optimization with Aeroelastic and Handling Qualities Constraints. *Journal of Guidance, Control, and Dynamics*, 20(2):217–224, March 1997.

[10] Diane L. Peters, Panos Y. Papalambros, and A. Galip Ulsoy. Sequential co-design of an artifact and its controller via control proxy functions. *Mechatronics*, 23(4):409–418, June 2013.

[11] Hosam K. Fathy, Panos Y. Papalambros, A. Galip Ulsoy, and Davor Hrovat. Nested plant/controller optimization with application to combined passive/active automotive suspensions. In *American Control Conference, 2003. Proceedings of the 2003*, volume 4, pages 3375–3380. IEEE, 2003.

[12] James T. Allison and Sam Nazari. Combined plant and controller design using decomposition-based design optimization and the minimum principle. In *ASME 2010 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages 765–774. American Society of Mechanical Engineers, 2010.

[13] James T. Allison, Tinghao Guo, and Zhi Han. Co-design of an active suspension using simultaneous dynamic optimization. *Journal of Mechanical Design*, 136(8):081003, 2014.

[14] Yi-Hsuan Hung and Chien-Hsun Wu. A combined optimal sizing and energy management approach for hybrid in-wheel motors of EVs. *Applied Energy*, 139:260–271, February 2015.

[15] Kamal Youcef-Toumi. Modeling, design, and control integration: a necessary step in mechatronics. *IEEE/ASME Transactions on Mechatronics*, 1(1):29–38, 1996.

[16] Malcolm J. Abzug and E. Eugene Larrabee. *Airplane stability and control: a history of the technologies that made aviation possible*. Number 14 in Cambridge aerospace series. Cambridge University Press, Cambridge, UK ; New York, 2nd ed edition, 2002.

[17] Tulga Ersal, Changsun Ahn, Diane L. Peters, John W. Whitefoot, Abigail R. Mechtenberg, Ian A. Hiskens, Huei Peng, Anna G. Stefanopoulou, Panos Y. Papalambros, and Jeffrey L. Stein. Coupling between component sizing and regulation capability in microgrids. *IEEE Transactions on Smart Grid*, 4(3):1576–1585, September 2013.

[18] Min-Joong Kim and Huei Peng. Power management and design optimization of fuel cell/battery hybrid vehicles. *Journal of Power Sources*, 165(2):819–832, March 2007.

[19] Nikolaos A. Diangelakis, Baris Burnak, Justin Katz, and Efstratios N. Pistikopoulos. Process design and control optimization: A simultaneous approach by multi-parametric programming. *AIChE Journal*, 63(11):4827–4846, November 2017.

[20] Vassilis Sakizlis, John D. Perkins, and Efstratios N. Pistikopoulos. Parametric Controllers in Simultaneous Process and Control Design Optimization. *Industrial & Engineering Chemistry Research*, 42(20):4545–4563, October 2003.

[21] L.A. Ricardez Sandoval, H.M. Budman, and P.L. Douglas. Simultaneous design and control of processes under uncertainty: A robust modelling approach. *Journal of Process Control*, 18(7-8):735–752, August 2008.

[22] Adrian Bejan. Fundamentals of exergy analysis, entropy generation minimization, and the generation of flow architecture. *International Journal of Energy Research*, 26(7):0–43, 2002.

[23] P. K. Nag and S. De. Design and operation of a heat recovery steam generator with minimum irreversibility. *Applied Thermal Engineering*, 17(4):385–391, 1997.

[24] Neera Jain and Andrew Alleyne. Exergy-based optimal control of a vapor compression system. *Energy Conversion and Management*, 92:353–365, March 2015.

[25] Meysam Razmara, Mehdi Maasoumy, Mahdi Shahbakhti, and Rush D. Robinett. Exergy-based model predictive control for building HVAC systems. In *American Control Conference (ACC), 2015*, pages 1677–1682. IEEE, 2015.

[26] M. Razmara, M. Maasoumy, M. Shahbakhti, and R.D. Robinett. Optimal exergy control of building HVAC system. *Applied Energy*, 156:555–565, October 2015.

[27] Neera Jain and Brandon M. Hencey. Increasing fuel thermal management system capability via objective function design. In *American Control Conference (ACC), 2016*, pages 549–556. IEEE, 2016.

[28] Wei Luo, Qiao Wang, Junjun Guo, Zhaohui Liu, and Chuguang Zheng. Exergy-based control strategy selection for flue gas recycle in oxy-fuel combustion plant. *Fuel*, 161:87–96, December 2015.

[29] P.J. Boait, R.M. Rylatt, and A. Wright. Exergy-based control of electricity demand and microgeneration. *Applied Energy*, 84(3):239–253, March 2007.

[30] Amip J. Shah, Van P. Carey, Cullen E. Bash, and Chandrakant D. Patel. An exergy-based control strategy for computer room air-conditioning units in data centers. In *ASME 2004 International Mechanical Engineering Congress and Exposition*, pages 61–67. American Society of Mechanical Engineers, 2004.

[31] B. Erik Ydstie. Passivity based control via the second law. *Computers & chemical engineering*, 26(7-8):1037–1048, 2002.

[32] J.P. Garca-Sandoval, N. Hudon, D. Dochain, and V. Gonzlez-lvarez. Stability analysis and passivity properties of a class of thermodynamic processes: An internal entropy production approach. *Chemical Engineering Science*, 139:261–272, January 2016.

[33] J.P. Garca-Sandoval, V. Gonzlez-lvarez, and C. Caldern. Stability analysis and passivity properties for a class of chemical reactors: Internal entropy production approach. *Computers & Chemical Engineering*, 75:184–195, April 2015.

[34] J.P. Garca-Sandoval, N. Hudon, and D. Dochain. Conservative and dissipative phenomena in thermodynamical systems stability. *IFAC-PapersOnLine*, 49(24):28–33, 2016.

[35] Li Wang, Bernhard Maschke, and Arjan van der Schaft. Irreversible port-Hamiltonian Approach to Modeling and Analyzing of Non-isothermal Chemical Reaction Networks. *IFAC-PapersOnLine*, 49(26):134–139, 2016.

[36] W. Zhou, B. Hamroun, Y. Le Gorrec, and F. Couenne. Lyapunov based nonlinear control of tubular chemical reactors. *IFAC-PapersOnLine*, 48(8):1045–1050, 2015.

[37] A.K. Amjad, R. Khoshbakhi Saray, S.M.S. Mahmoudi, and A. Rahimi. Availability analysis of n-heptane and natural gas blends combustion in HCCI engines. *Energy*, 36(12):6900–6909, December 2011.

[38] C Rakopoulos and E Giakoumis. Second-law analyses applied to internal combustion engines operation. *Progress in Energy and Combustion Science*, 32(1):2–47, 2006.

[39] Feng Yan and Wanhua Su. A promising high efficiency rm-hcci combustion proposed by detail kinetics analysis of exergy losses. *SAE Technical Paper*, 2015.

[40] Meysam Razmara, Mehran Bidarvatan, Mahdi Shahbakhti, and Rush Robinett. Innovative exergy-based combustion phasing control of ic engines. *SAE Technical Paper*, 2016.

[41] Cristina Renzi. A genetic algorithm-based integrated design environment for the preliminary design and optimization of aeronautical piston engine components. *The International Journal of Advanced Manufacturing Technology*, 86(9-12):3365–3381, October 2016.

[42] Wanzhong Zhao, Zunsi Yang, and Chunyan Wang. Multidisciplinary hybrid hierarchical collaborative optimization of electric wheel vehicle chassis integrated system based on drivers feel. *Structural and Multidisciplinary Optimization*, 57(3):1129–1147, March 2018.

[43] YuanDong Liu, YiHui Yin, and ZhongZe Guo. Static and dynamic design based on hierarchical optimization for materials and structure of porous metals. *Science China Technological Sciences*, 55(10):2808–2814, October 2012.

[44] Ricardo Martins, Nuno Loureno, and Nuno Horta. Multi-objective optimization of analog integrated circuit placement hierarchy in absolute coordinates. *Expert Systems with Applications*, 42(23):9137–9151, December 2015.

[45] Chen Zheng, Peter Hehenberger, Julien Le Duigou, Matthieu Bricogne, and Benot Eynard. Multidisciplinary design methodology for mechatronic systems based on interface model. *Research in Engineering Design*, 28(3):333–356, July 2017.

[46] Shozo Takata and Yasushi Umeda, editors. *Advances in life cycle engineering for sustainable manufacturing businesses: proceedings of the 14th CIRP Conference on Life Cycle Engineering, Waseda University, Tokyo, Japan, June 11th-13th, 2007.* Springer, London, 2007. OCLC: ocn124026486.

[47] Jrgen Gausemeier and Stefan Moehringer. New Guideline Vdi 2206-a Flexible Procedure Model for the Design of Mechatronic Systems. In *DS 31: Proceedings of ICED 03, the 14th International Conference on Engineering Design, Stockholm*, 2003.

[48] Sebastian Jansen and Ewald G. Welp. Model-based design of actuation concepts: a support for domain allocation in mechatronics. In *DS 35: Proceedings ICED 05, the 15th International Conference on Engineering Design, Melbourne, Australia, 15.-18.08. 2005*, 2005.

[49] Zdenek Hadas, Vladislav Singule, Stanislav Vechet, and Cestmir Ondrusek. Development of energy harvesting sources for remote applications as mechatronic systems. In *Power Electronics and Motion Control Conference (EPE/PEMC), 2010 14th International*, pages T10–13. IEEE, 2010.

[50] Lihui Wang and Andrew Y.C. Nee, editors. *Collaborative Design and Planning for Digital Manufacturing.* Springer London, London, 2009.

[51] Riccardo Scattolini. Architectures for distributed and hierarchical Model Predictive Control A review. *Journal of Process Control*, 19(5):723–731, May 2009.

[52] R.C. Hill, J.E.R. Cury, M.H. de Queiroz, D.M. Tilbury, and S. Lafortune. Multi-level hierarchical interface-based supervisory control. *Automatica*, 46(7):1152–1164, July 2010.

[53] Samira Rahnama, Jan Dimon Bendtsen, Jakob Stoustrup, and Henrik Rasmussen. Robust Aggregator Design for Industrial Thermal Energy Storages in Smart Grid. *IEEE Transactions on Smart Grid*, pages 1–15, 2015.

[54] Jan Bendtsen, Klaus Trangbaek, and Jakob Stoustrup. Hierarchical model predictive control for resource distribution. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 2468–2473. IEEE, 2010.

[55] Herschel C. Pangborn, Justin P. Koeln, Matthew A. Williams, and Andrew G. Alleyne. Experimental Validation of Graph-Based Hierarchical Control for Thermal Management. *Journal of Dynamic Systems, Measurement, and Control*, 140(10):101016, 2018.

[56] Daniel R. Herber and James T. Allison. A Problem Class with Combined Architecture, Plant, and Control Design Applied to Vehicle Suspensions. *Journal of Mechanical Design*, 141(10):101401, 2019.

[57] Satya RT Peddada, Daniel R. Herber, Herschel C. Pangborn, Andrew G. Alleyne, and James T. Allison. Optimal flow control and single split architecture exploration for fluid-based thermal management. *Journal of Mechanical Design*, 141(8):083401, 2019.

[58] Yu Jiang, Yebin Wang, Scott A. Bortoff, and Zhong-Ping Jiang. Optimal codesign of nonlinear control systems based on a modified policy iteration method. *IEEE Transactions on Neural Networks and Learning Systems*, 26(2):409–414, February 2015.

[59] Austin L. Nash and Neera Jain. Combined plant and control co-design for robust disturbance rejection in thermal-fluid systems. *IEEE Transactions on Control Systems Technology*, 2019, submitted.

[60] Michael J. Moran and Howard N. Shapiro. *Fundamentals of engineering thermodynamics*. Wiley, Hoboken, NJ, 6 edition, 2007.

[61] E. Fridman and U. Shaked. Delay-dependent stability and H control: Constant and time-varying delays. *International Journal of Control*, 76(1):48–60, January 2003.

[62] J Currie and DI Wilson. Foundations of computer-aided process operations. *OPTI: lowering the barrier between open source optimizers and the industrial MATLAB user. Savannah, Georgia, USA*, 2012.

[63] Johan Lofberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*, pages 284–289. IEEE, 2004.

[64] Chunbao Liu, Weiyang Bu, and Dong Xu. Multi-objective shape optimization of a plate-fin heat exchanger using CFD and multi-objective genetic algorithm. *International Journal of Heat and Mass Transfer*, 111:65–82, August 2017.

[65] Houkuan Tam, Lapmou Tam, Sikchung Tam, Chouhei Chio, and Afshin J. Ghajar. New optimization method, the algorithms of changes, for heat exchanger design. *Chinese Journal of Mechanical Engineering*, 25(1):55–62, January 2012.

[66] Habib Karimi, Hossein Ahmadi Danesh Ashtiani, and Cyrus Aghanajafi. Study of mixed materials heat exchanger using optimization techniques. *Journal of Engineering, Design and Technology*, January 2019.

[67] Hong Bo Li, Xin Min Dong, Jun Guo, Yong Chen, and Ting Ting Li. Heat exchanger optimization analysis of aircraft environmental control system. *Advanced Materials Research*, 383-390:5536–5541, November 2011.

[68] Ngo Thi Phuong Thuy, Rajashekhar Pendyala, Nejat Rahmanian, and Narahari Marneni. Heat exchanger network optimization by differential evolution method. *Applied Mechanics and Materials*, 564:292–297, June 2014.

[69] Antonio C. Caputo, Pacifico M. Pelagagge, and Paolo Salini. Joint economic optimization of heat exchanger design and maintenance policy. *Applied Thermal Engineering*, 31(8-9):1381–1392, June 2011.

[70] Jiangfeng Guo and Mingtian Xu. The application of entransy dissipation theory in optimization design of heat exchanger. *Applied Thermal Engineering*, 36:227–235, April 2012.

[71] Dillip Kumar Mohanty. Application of firefly algorithm for design optimization of a shell and tube heat exchanger from economic point of view. *International Journal of Thermal Sciences*, 102:228–238, April 2016.

[72] E Fanelli, N Lovaglio, G Cornacchea, G Braccio, and V Magi. Heat exchanger design and optimization by using genetic algorithm for externally fired microturbine. In *73rd Conference of the Italian Thermal Machines Engineering Association*, Pisa, Italy, September 2018.

[73] Tarun K. Poddar and Graham T. Polley. Optimize shell-and-tube heat exchanger design. *Chemical Engineering Progress*, 96(9), 2000.

[74] Herschel Pangborn, Andrew G. Alleyne, and Ning Wu. A comparison between finite volume and switched moving boundary approaches for dynamic vapor compression system modeling. *International Journal of Refrigeration*, 53:101–114, May 2015.

[75] Joseph Fasl. *Modeling and control of hybrid vapor compression cycles.* PhD thesis, University of Illinois at Urbana-Champaign, 2013.

[76] Herschel Pangborn. *Dynamic modeling, validation, and control for vapor compression systems.* PhD thesis, University of Illinois at Urbana-Champaign, 2015.

[77] Bin Li. *Dynamic modeling, simulation, and control of transportation hvac systems.* PhD thesis, University of Illinois at Urbana-Champaign, 2013.

[78] William S Janna. *Design of Fluid Thermal Systems-SI Version.* Cengage learning, 2010.

[79] Robert W. Fox and Alan T. McDonald. *Introduction to Fluid Mechanics.* John Wiley and Sons, New York, 1994.

[80] P.C. Krause and Associates. *AFRL Transient Thermal Management Optimization (ATTMO)*, 2019.

[81] Patrick McCarthy, Nicholas Niedbalski, Kevin McCarthy, Eric Walters, Joshua Cory, and Soumya Patnaik. A first principles based approach for dynamic modeling of turbomachinery. *SAE International Journal of Aerospace*, 9(2016-01-1995):45–61, 2016.

[82] Kevin McCarthy, Patrick McCarthy, Ning Wu, Andrew Alleyne, Justin Koeln, Soumya Patnaik, Stephen Emo, and Joshua Cory. Model accuracy of variable fidelity vapor cycle system simulations. Technical report, SAE Technical Paper, 2014.

VITA

Austin Nash was born in Terre Haute, Indiana, in 1990 and raised nearby in Sullivan, Indiana. He graduated from Sullivan High School in 2009 and attended Rose-Hulman Institute of Technology in Terre Haute to study mechanical engineering. He received the Bachelor of Science degree, *summa cum laude*, in 2013 and remained at Rose-Hulman to pursue a Master of Science. His MS research, with Prof. Bradley Burchett, focused on guidance, navigation and control of military projectiles. After receiving the MS degree in 2015, he entered the mechanical engineering Ph.D. program at Purdue University. Under the advisement of Prof. Neera Jain, his research has centered on dynamic modeling, advanced control, and optimization of energy-related systems.

PUBLICATIONS

1. **A. Nash** and N. Jain, “Hierarchical Combined Plant and Control Design: A Case Study on Thermal Management Systems,” (in preparation).
2. **A. Nash** and N. Jain, “Combined Plant and Control Co-design for Robust Disturbance Rejection in Thermal-Fluid Systems,” *IEEE Transactions on Control Systems Technology.* (In Press)
3. **A. Nash**, B. Fu, T. Bird, N. Jain, and T. Fisher, “Control-Oriented Modeling of Integrated Flash Boiling for Rapid Transient Heat Dissipation,” *AIAA Journal of Thermophysics and Heat Transfer*, vol. 33, July 2019.
4. **A. Nash**, A. Badithela, and N. Jain, “Dynamic Modeling of a Sensible Thermal Energy Storage Tank with an Immersed Coil Heat Exchanger under Three Operation Modes,” *Applied Energy*, vol. 195, pp. 877-889, June 2017.

Conference Proceedings with Full Paper (* *indicates invited paper*)

1. **A. Nash** and N. Jain, “Dynamic Design Optimization for Thermal Management: A Case Study on Shell and Tube Heat Exchangers,” *Proceedings of the 2019 ASME Dynamic Systems and Control Conference*, Park City, UT, October 8-11, 2019.*
2. **A. Nash**, T. Fisher, and N. Jain, “Combined Plant and Control Design for a Flash Boiling Cooling System,” *Proceedings of the 2019 American Control Conference*, Philadelphia, PA, July 10-12, 2019.*
3. **A. Nash** and N. Jain, “Second Law Modeling and Robust Control for Thermal-Fluid Systems.” *Proceedings of the 2018 ASME Dynamic Systems and Control Conference*, Atlanta, GA, September 30 - October 3, 2018.*
4. **A. Nash** and N. Jain, “Dynamic Modeling and Performance Analysis of Sensible Thermal Energy Storage Systems,” *Proceedings of the 4th International High Performance Buildings Conference*, Purdue University, West Lafayette, IN, July 11-14, 2016.