MODEL FREE REAL-TIME OPTIMIZATION FOR VAPOR COMPRESSION SYSTEMS

BY

BRYAN DAVID KEATING

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Mechanical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2017

Urbana, Illinois

Adviser:

Professor Andrew Alleyne

ABSTRACT

A vapor compression system's optimal input settings vary according to changes in environmental conditions. Tracking these optimal input trajectories can be challenging when insufficient information for a reliable system model is available. An alternative set of optimization approaches use system measurements. This thesis focuses on one such approach, extremum seeking control, which uses performance index measurements to determine optimal system settings.

Forgoing system model knowledge and relying exclusively on data allows an optimization approach to function well on many different plants. However, this added adaptivity comes at a performance cost. Using prior system model knowledge can be helpful for ensuring that a controller design works from the start of operation and inputs can be changed as soon as information about environmental conditions is updated. By contrast, data based methods may require the control designer to spend a time generating data in order to obtain enough information about the system to make good decisions online. A central theme of this work is addressing the trade off between using prior system model knowledge and ensuring sufficient adaptability of the extremum seeking optimization approach.

Two main factors in the extremum seeking design are considered: the choice of extremum seeking control law and the choice of extremum seeking control input. Extremum seeking control laws come from the field of mathematical optimization; this thesis considers the pros and cons of choosing between gradient descent and Newton descent. Both simulations and experimental results show that while Newton descent extremum seeking is less reliant on model knowledge, but slower to find optimal inputs than gradient descent extremum seeking. Because of extremum seeking's adaptability to different plants, many different inputs can be chosen for implementation. However, using an approach known as self-optimizing control, knowledge

about the plant's behavior can help choose set points with optimal values that are insensitive to changes in environmental conditions. Finding these special inputs turns the input tracking problem into a regulation problem. Both simulation and experimental results confirm that combining self-optimizing control and extremum seeking control can help improve tracking even as environmental conditions change.

To my parents, for their love and support.

ACKNOWLEDGMENTS

The work in this thesis is the product of an effort far greater than what I could have achieved on my own. I am indebted to Dr. Andrew Alleyne for his patience, kindness, expert advice, and willingness to support my research career. I am grateful for his care in assembling a research group where there is little distinction between friends and colleagues. The members of the Alleyne research group have been a foundation of my wonderful graduate school experience; I thank Tim Deppen, Justin Koeln, Matt Williams, Herschel Pangborn, Katie Amber, Lindsey Gonzales, Kaveh Kianfar, José Enrique Alonso Alfaya, Nate Weir, Ashley Armstrong, Spencer Kieffer, Malia Kawamura, Pamela Tannous, Oyuna Angatkina, Xavier Moya, Sunny Sharma, Spencer Ingram, and Sarah Garrow for their friendship and collaboration.

Tim Deppen, Herschel Pangborn, and Justin Koeln have been role models for helping me think through research problems and draft publications. Thanks to Matt, Pamela, Malia, and Nate for helping me through TAM 541, Math Methods; to Herschel for struggling through ME 561, Convex Methods in Control, with me; to Pamela for helping me understand the Kalman filter in AE 498, Estimation and Data Assimilation. A special thanks to Jon Hoff, who helped me through five classes.

I am grateful for the professors who have provided me with invaluable background knowledge for graduate study. Thanks to Dr. Andrew Alleyne for his succinct and practical ME 460 presentations; to Dan Block and Dr. Dusan Stipanovic for their friendly introduction to digital control systems in GE 420; to Dr. Geir Dullerud for his unrelentingly challenging classes based on linear systems theory, ECE 515 and ME 561; to Dr. Negar Kiyavash for introducing me to the field of optimization in ECE 490; to Dr. Prashant Mehta for giving a broad and intuitive overview of linear algebra, complex analysis, and differential equations in TAM 541; to Dr. Carolyn Beck for teaching the fundamentals of system identification which have been invalu-

able in my research efforts; and finally, to Dr. Sri Namachchivaya for his elegant presentation of the Kalman Filter in AE 498.

I am indebted to my friends outside of lab who have made living in Urbana-Champaign feel like home. Thanks to Jon and Bethany Hoff, Peter Lee, Courtney Allen, Anna Lapp, Shelby Hallman, Ayush Saxena, Adnan Akhtar, and Peter Knapp for their friendship; to Herschel Pangborn, Christine McCarthy, and Harith Kassar who built my self-defense skills through Tang Soo Do instruction; to Gokul Pathikonda and Sohan Sundir for their enthusiasm about squash and willingness to suffer through anaerobic conditioning; to Rishi Singh and Adam Wood for hanging out and playing pickup basketball; and again to Jon Hoff, for the great conversation, unwaivering friendship, and for suffering with me through weightlifting sessions.

Finally, I am deeply grateful to my girlfriend, Relicque Lott, and my family for their unconditional love and support through my graduate career.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ESC	Extremum seeking control
SOC	Self-optimizing control
RTO	Real-time optimization
LQG	Linear quadratic Gaussian
LQR+I	Linear quadratic regulator with integral action
gLTI	Gradient descent extremum seeking control using LTI filters with demodulation
gLTI	Gradient descent extremum seeking control using time varying filters
nLTI	Newton descent extremum seeking control using LTI filters with demodulation
nTV	Newton descent extremum seeking control using time varying filters
VCS	Vapor compression system
HVAC	Heating, ventilation, and air conditioning
CL	Closed loop

A	Dynamics matrix
B	Input matrix
C	Output (sensor) matrix
$\dot{W}$	Power consumption
$\dot{Q}$	Heat transfer rate
RH	Relative Humidity

T	Temperature
P	Pressure
u	Feedback controller input
v	Optimization input
w	Disturbance
x	System state
y	Output
J	Steady state optimization performance index
S	Dynamic optimization performance index
SH	Superheat
$\dot{m}$	Mass flow rate
$\dot{V}$	Volume flow rate
ρ	Density or cost function weighting
ω	Frequency
a	Amplitude
K	Scaling matrix
Q	Weighting matrix for Kalman Filter or LQR
R	Weighting matrix for Kalman Filter or LQR
G	Gain
H	Measurement combination matrix or output matrix
L	Observer gain
τ	Time constant
λ	Forgetting factor
ω	Frequency

CHAPTER 1

INTRODUCTION

Vapor compression systems (VCSs) are machines that exploit the principles of thermodynamics to achieve efficient energy transfer using a repeated cycle of evaporation, compression, condensation, and expansion. They act as crucial subsystems in many practical applications that involve some form of temperature regulation, from room air conditioning to natural gas liquefaction. Moving energy from low to high temperature comes at a cost which can determine the return on investment for an air conditioning system or affect the amount of profit extracted from an industrial process. Several studies in the literature have shown that finding efficient operation settings of subsystems such as fans, pumps, and compressors can minimize the cost of system operation. This thesis addresses "real-time optimization", where measurements are used to determine the system's optimal settings.

Real-time optimization (RTO) is often synonymous with optimization near steady state conditions. Choosing RTO input values requires some system knowledge. Just how much system knowledge is used depends on confidence in the model and how much effort the control designer is willing to spend. Developing and tuning a physics based model often requires hand tuning effort to achieve an acceptable match between physical outputs and simulation outputs. Poor tuning can create situations where the model's optimal inputs are not the true ones. Furthermore, the real-time optimizer may not have full knowledge of the system's operating conditions due to a lack of measurements. This is unavoidable for aging systems that experience changes in the true model parameters. However, one advantage of using a physics based model is that it does not require much time to select new optimal inputs.

Model free optimization is a form of RTO that does not rely on a system's governing equations to find the optimal inputs. Extremum seeking control (ESC) is one such model free approach. Inherently a feedback based adaptive

control algorithm, it estimates derivatives from plant measurements and uses these derivatives in an optimization algorithm to reach extremum points of a system's steady state performance index. A key assumption is that the system can be described by a nonlinear program with a local extremum point. One of the most attractive features of extremum seeking control is its ability to optimize complicated systems with minimal knowledge of system dynamics. This makes extremum seeking a promising tool for vapor compression systems when optimization using a system model might fail to match the true optimal inputs over the entire operating envelope. However, ignorance of the system model comes at a price; a drawback of extremum seeking control is that it inherently exchanges prior model knowledge for a slow convergence time and optimality guarantees. This is because it must experiment with the system's performance index and operate slowly enough to make correct decisions about the optimal value.

Given the advantages of using a system model to obtain knowledge about system behavior and the guarantee that ESC is capable of finding optimal inputs without this knowledge, a main feature of this work is the exploration of how physics based modeling and extremum seeking control can complement one another. This aspect is explored in two ways: using a physical system model to assist extremum seeking control design and using model based optimization in parallel with extremum seeking control.

1.1 Literature

The idea of ESC has been traced to the work of Leblanc back in 1922, but ESC only became popular as a research topic near the turn of the millennium when [4] published a proof of stability for the classical extremum seeking algorithm. Since then, many ESC algorithms have emerged as solutions for handling real-time optimization problems across a range of applications from formation flight to photovoltaic maximum power point tracking. A comprehensive review of recent ESC publications, applications, and basic theoretical principles can be found in [1].

[5] gives an overview of a unifying framework for the development of black box and gray box extremum seeking algorithms. Black box extremum seeking algorithms (used exclusively in this thesis) rely on steady state input to

output relationships, while grey box algorithms exploit knowledge of system dynamics. Whether black box or gray box, all extremum seeking algorithms are meant to estimate derivatives of a steady state performance index.

A key theme in the theoretical study of extremum seeking algorithms is assessing stability of the feedback interconnection of the plant, derivative estimation algorithm, and the control law. In terms of black box algorithm development, where the system's parameterization is assumed to be static, major contributions to ESC theory have been from novel parameterizations for derivative estimation, the corresponding estimation scheme, and the proof of closed loop system stability [4, 2, 3, 6]. An interesting hybrid black/gray box approach that has emerged in the literature is from [7, 8]. It uses an input affine parameterization of the plant and uses the ESC algorithm to estimate a Lie derivative for gradient descent.

In the last 10 years, the HVAC control systems community has become interested in ESC as a RTO solution following the publication of [9], which discussed the role of controls and optimization in energy efficient HVAC systems. Several high fidelity simulations and experiments have validated ESC's ability to optimize the operation of a variety of thermal management systems including airside economizers, chillers, thermoacoustic coolers, and home air conditioners [10, 11, 12, 13, 14, 15, 9, 16, 17, 18, 19, 8]. Because extremum seeking is used to exploit the trade offs inherent in the movement of mass and energy, almost all of the cost functions presented in these works are convex. Either some form of the classical perturbation based algorithm [4], its newton update extension [6], time varying extremum seeking [3], or proportional integral extremum seeking [7] have been applied to these thermal systems.

One lingering question about application of ESC to thermal systems is how model information should be incorporated into the controller design. Forgoing model knowledge requires a more conservative, but less system specific approach to optimization. Another aspect of these works is the method of ESC implementation. Up until now, many authors have implemented ESC using intuition for what constitutes a good input; there has been no justification for input choice in any of the following works: In [11, 17, 10, 19, 16, 14].

The method of choosing this input can be formalized by self-optimizing control (SOC), an approach to choosing optimal controlled variables for real-time optimization. When these special self-optimizing variables are regulated to constant set points, the plant stays close to its optimal operating condi-

tions [20]. In this sense, SOC is an optimization strategy that compensates for disturbances that are modeled, but are not measured online. Such an approach can be valuable when disturbances not economically measurable or the disturbance measurement is unreliable. Of the various methods available for choosing these controlled variables, this work focuses on the null space method [21].

While originally model based, SOC has recently been extended to be a data based technique. The original implementation of SOC involved using a time invariant linearized steady state model with a cost function approximated by a convex quadratic to select an optimal measurement combination [22, 21]. Where a sufficiently accurate system model is unavailable, a self-optimizing measurement combination can be extracted from process data [23, 24]. An advantage of using process data is that the information from governing physics informs the data analysis, but the resulting measurement combination is specific to the physical system that generated the data.

SOC has been applied to vapor compression cycles on a handful of occasions in the literature. [25] first applied SOC to static models of an Ammonia cycle and a CO_2 cycle to analyze the best controlled variables for steady state efficiency. [26] built on this study by implementing null space method SOC on the same Ammonia cycle model. In [27], the authors used model based SOC combined with MPC.

An alternative implementation of ESC is to choose an extremum seeking controlled variable using SOC. Because SOC uses set point regulation tools, it operates at the time scale of the closed loop system. In [28], the authors used a continuously stirred tank reactor simulation model to test the combined ESC and SOC against RTO controllers that used either ESC or SOC. The combined ESC and SOC showed superior performance in terms of operational profit.

1.2 Organization of Thesis

The thesis is organized in the following way: Chapter 2 discusses the operation, modeling, and optimization of the four component vapor compression cycle used for cooling. To build intuition for the real-time optimization tools in this thesis, Chapter 3 reviews formulations for both extremum seeking

and self optimizing control and applies them to example systems. Chapter 4 discusses the choice between Newton descent and gradient descent extremum seeking algorithms for application to a vapor compression system and how system model knowledge can affect this choice. Chapter 5 illustrates the value of using self-optimizing control combined with extremum seeking control to provide fast optimization for a vapor compression system. Both Chapter 4 and Chapter 5 emphasize comparisons between simulation and experiment. Finally, Chapter 6 presents the contributions of this work and provides ideas for future exploration.

CHAPTER 2

OPERATION, MODELING, AND REAL-TIME OPTIMIZATION OF VAPOR COMPRESSION SYSTEMS

Before applying any kind of control or optimization to a system, the following questions arise:

1. What are the basic principles of the system's operation? What does it mean for the system to operate under normal circumstances?
2. How can the system be modeled and analyzed?
3. What is a sensible index for scoring the system's performance?

This chapter answers the questions above for a specific type of vapor compression system: the four component cycle consisting of an evaporator, a compressor, a condenser, and a valve. The same basic answers can often be applied to vapor cycle systems with additional components. The interested reader is referred to studies that support this premise [10, 29, 25].

The questions posed above are answered in order: Section 2.1 introduces the four component cycle's operational principles and the physical system used as a test case; Section 2.2 shows the steps taken to build a physical system model of the test rig described in Section 2.1; Section 2.3 describes a general way to index vapor compression system performance and applies it to a special case for the system in Section 2.1; finally, Section 2.4 presents concluding remarks. Section 2.5 provides full details of the vapor compression system optimization problem considered in this work.

2.1 Operation

2.1.1 Four Component Cycle

Vapor compression systems (VCSs) can be used for heating, cooling, or both. This section focuses on cooling mode, which is relevant to room air condi-

tioners, building chillers, and natural gas liquefaction cycles [30, 13, 29].

Cooling arises from a heat exchange between the low temperature refrigerant such as ammonia, carbon dioxide, or a hydro-fluorocarbon, and a relatively higher temperature secondary fluid such as air or water (Points 4-1 in Figure 2.1). Compressing the refrigerant to a high enough temperature (Points 1-2) allows a rejection of heat to a relatively lower temperature secondary fluid stream (Points 2-3). As the refrigerant passes through a valve, both refrigerant temperature and pressure drop and the cycle repeats (Points 3-4).

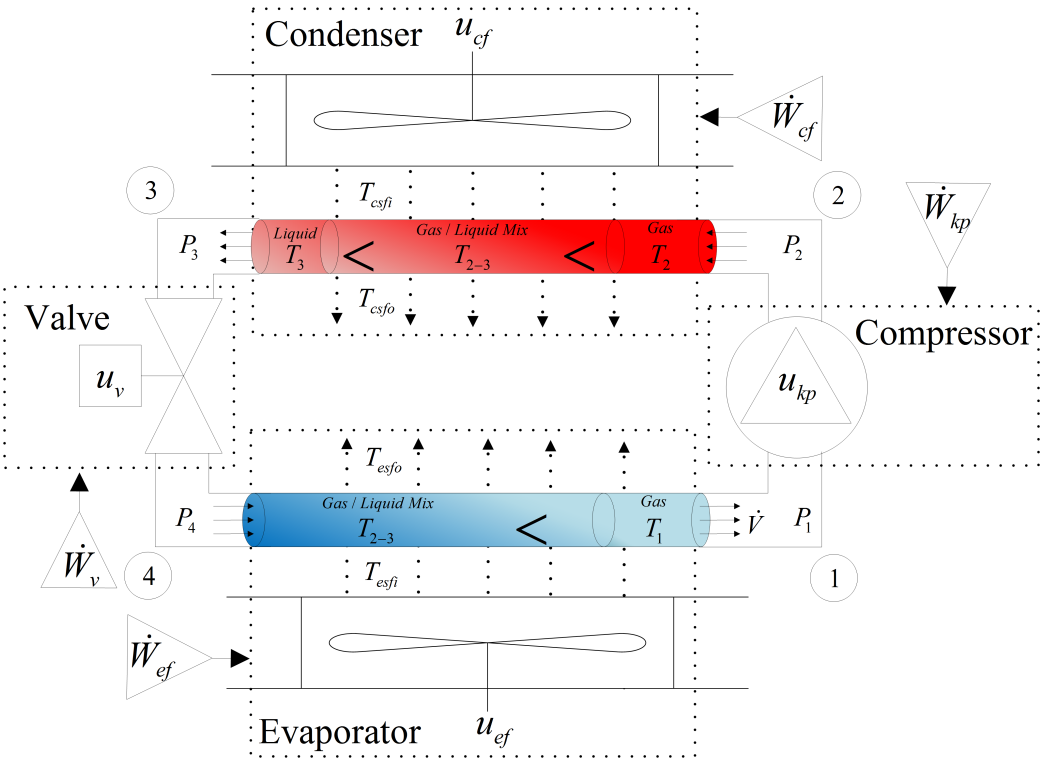

Figure 2.1: Schematic of a vapor compression system that cools air at T_{esfi} using a compressor, a valve, and two heat exchangers to drive the refrigerant cycle. While the secondary fluid shown is air, it could also be pumped water or refrigerant from another VCS stage.

Broadly speaking, the four component VCS's operational principles have been highlighted, but not in precise terms of inputs and measurements, which enable system monitoring and feedback control. Commercial VCSs are typically outfitted with thermocouples and pressure transducers to capture a subset of these available measurements, while the inputs can either be fixed speed or variable speed.

In addition to the measurements and inputs, the system's design (geometry and material) plays a key role in determining the "normal" operation regime. While worth mentioning, system design involves considerations of heat transfer and fluid dynamics, which are outside of this thesis's scope. Given this fact, the steps below define variables of the cycle's component subsystems, the evaporator, compressor, condenser, and valve.

4-1 (Evaporation):

-Refrigerant evaporates at low pressure and temperature, P_1 and T_4, and exchanges heat with the secondary fluid, which has inlet temperature, T_{esfi}, and outlet temperature, T_{esfo}.

-Refrigerant exits the evaporator with a positive superheat, defined as $T_{SH} \equiv T_1 - T_4$. Maintaining this temperature difference protects the compressor from ingesting liquid refrigerant. Compression of liquid refrigerant is not catastrophic, but it is known to cause premature failure [31].

-A fan or pump, with a constrained input command, u_{ef}, and input power, $\dot{W}_{ef}$, enhances the heat exchange process.

1-2 (Compression):

-The compressor, with a constrained input command, u_{kp}, and input power, $\dot{W}_{kp}$, increases the refrigerant pressure and temperature from P_1 and T_1 to P_2 and T_2 respectively.

2-3 (Condensation):

-Gaseous refrigerant emerging from the compressor at temperature, T_2, and pressure, P_2, is cooled by the secondary fluid at inlet temperature, T_{csfi}, and exit temperature, T_{csfo}. Temperature T_{2-3} represents a constant temperature section where the refrigerant undergoes a phase change by condensation. The refrigerant temperature at the condenser exit is $T_3 < T_{2-3}$, which corresponds to a subcooled liquid. While a positive evaporator superheat is not always achieved, $T_2 > T_{2-3}$ is almost guaranteed by the system design.

-A fan or pump, with a constrained input command, u_{cf}, and input power, $\dot{W}_{cf}$, enhances the heat exchange process.

-Refrigerant exits the condenser with a positive subcooling temperature, $T_{SC} \equiv T_{2-3} - T_3$.

3-4 (Expansion):

-The electronic expansion valve, with input u_v, and input power, $\dot{W}_v$, decreases the temperature and pressure of the liquid refrigerant exiting the condenser from T_3 to T_4 and P_2 to P_1 respectively.

Although the cycle's steps are distinct, some of the measurements described above are coupled by thermodynamic laws and/or system assumptions. For instance, pressure losses in the evaporator and condenser pipes are negligible. Thus, $P_2 \approx P_3$ and $P_1 \approx P_4$. Table 2.1 defines the measurements in Figure 2.1 and the relationships between them.

Table 2.1: Description of VCS temperature, pressure, and power consumption measurements.

Measurement	Coupling	Description
Refrigerant Pressures		
P_1	$P_1 \approx P_4 \approx P_{sat}(T_4)$	Evaporator outlet
P_2	$P_2 \approx P_3 \approx P_{sat}(T_{2-3})$	Condenser inlet
P_3	See P_2	Condenser outlet
P_4	See P_1	Evaporator inlet
Refrigerant Temperatures		
T_1	–	Evaporator outlet
T_2	–	Condenser inlet
T_3	–	Condenser outlet
T_4	See P_1	Evaporator inlet
Secondary Fluid Temperatures		
T_{esfi}	–	Evaporator inlet
T_{esfo}	–	Evaporator outlet
T_{csfi}	–	Condenser inlet
T_{csfo}	–	Condenser outlet
Flow Rate		
$\dot{V}$	–	Refrigerant volume flow rate
Power Consumption		
$\dot{W}_{kp}$	See $\dot{W}_{sys}$	Compressor
$\dot{W}_{cf}$	See $\dot{W}_{sys}$	Condenser Fan
$\dot{W}_v$	$\dot{W}_v \approx 0$	Valve
$\dot{W}_{ef}$	See $\dot{W}_{sys}$	Evaporator Fan
$\dot{W}_{sys}$	$\dot{W}_{sys} \approx \dot{W}_{kp} + \dot{W}_{cf} + \dot{W}_{ef}$	Total system power

In total, there are up to 14 unique measurements available on the simple cycle: 2 pressures, 3 refrigerant temperatures, 1 volume flow rate, 4 secondary fluid temperatures, and up to 4 power consumption measurements. Choosing which measurements are implemented on an actual system depends on the operational goals and the cost of sensors. Fortunately, the experimental test stand used by this study is equipped with all of the pressure, temperature, and flow rate measurements above as well as the total system power

measurement.

2.1.2 Experimental Test Stand

The HVAC test stand in Figure 2.2 has many components and possible configurations [32]. This study selected the four component cycle corresponding to Figure 2.1.

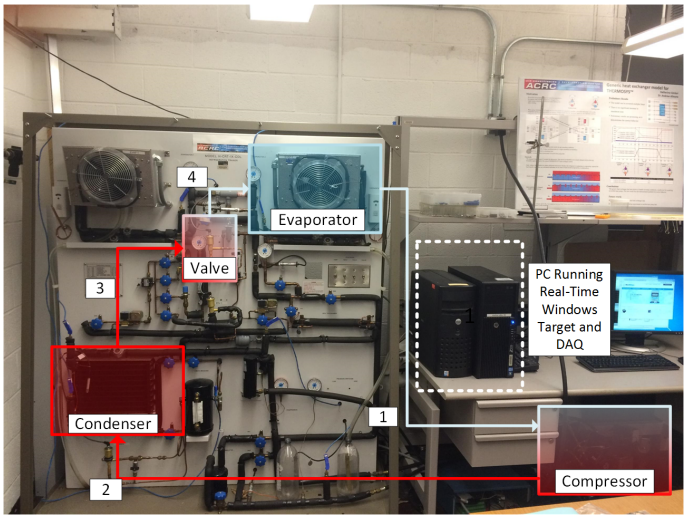

Figure 2.2: Experimental vapor compression system running a simple cycle refrigerant loop. The numbered points on the photograph correspond to the cycle points in Figure 2.1.

Some system details follow: the refrigerant is R134a and the secondary fluid is the room air; the room's HVAC system maintains an air temperature set point around $22°C$; temperatures, pressures, the flow measurement, and the power measurement come from immersion thermocouples, pressure transducers, a volume flow sensor, and a power meter, respectively. Data collection and control are coordinated by a DAQ and MATLAB's Real-Time Windows Target.

Earlier, it was stated that the VCS's goal was to achieve a set amount of cooling. Because the testbed is situated in a room with recirculating air, there is little control over the room air temperature. Instead, the refrigerant side cooling capacity, $\dot{Q}_{evap}$, is a stand-in for the indicator of achieved cooling. The cooling capacity can be expressed as a function of system measurements. Table 2.2 gives the experimental system measurements in use, as well as the

formulas for calculating mass flow, $\dot{m}$, cooling capacity, $\dot{Q}_{evap}$, superheat, T_{SH}, and refrigerant boiling temperatures, T_4 and T_{2-3}. Instances of h and ρ represent specific enthalpy and density respectively. Instances of P represent pressures and instances of T represent temperatures.

Table 2.2: Formulas for VCS experimental measurements.

Direct Measurements		Calculated Outputs	
Pressures [kPa]	P_1, P_2	$\dot{m}[kg/s]$	$\dot{V}\rho(P_2, T_3)$
Temperatures $[^\circ C]$	T_1, T_2, T_3	$\dot{Q}_{evap}[kW]$	$\dot{m}(h_1(P_1, T_1) - h_4(P_2, T_3))$
Volume Flow Rate $[m^3/s]$	$\dot{V}$	$T_{SH}[^\circ C]$	$T_1 - T_{sat}(P_1)$
Power [kW]	$\dot{W}_{sys}$	$T_4[^\circ C]$	$T_{sat}(P_1)$
		$T_{2-3}[^\circ C]$	$T_{sat}(P_2)$

Some of the measurements in Table 2.2 may be unavailable in practice. For example, volume flow rate sensors are costly and seldom used on commercial products. However, this wealth of measurements available on the experimental testbed makes the system suitable for testing various modeling and control strategies. The next section will review and give results from the modeling study undertaken.

2.2 Modeling

2.2.1 Physics Based Simulation in Thermosys

Access to a diverse set of system measurements provided all of the necessary details for calibration of a physics based VCS simulation model. Such models are useful because they help explain fundamental system behavior and execute case studies up to 1400 times faster than real time.

This study employed Thermosys [33], a vapor compression system dynamic modeling toolbox. Thermosys uses a switched moving boundary approach, which models the distinct fluid zones in Figure 2.1 as control volumes. Subjecting the control volumes to mass and energy balances gives a system of low order nonlinear dynamic equations that have proven useful for vapor

compression system modeling. For details on the moving boundary approach, consult [34].

To achieve some level of accuracy between a real system and the Thermosys model, a calibration procedure consisting of the following two steps should be performed using an experimental system. First, collect data from an experiment where the VCS's inputs are perturbed. Next, enter steady state system measurements and hand tune Thermosys parameters to match the dynamic response and steady-state behavior of the experimental data.

Figure 2.3 shows input perturbations centered about the nominal values. Figure 2.4 shows that air inlet temperatures, which are uncontrollable system inputs, are nearly constant throughout the test.

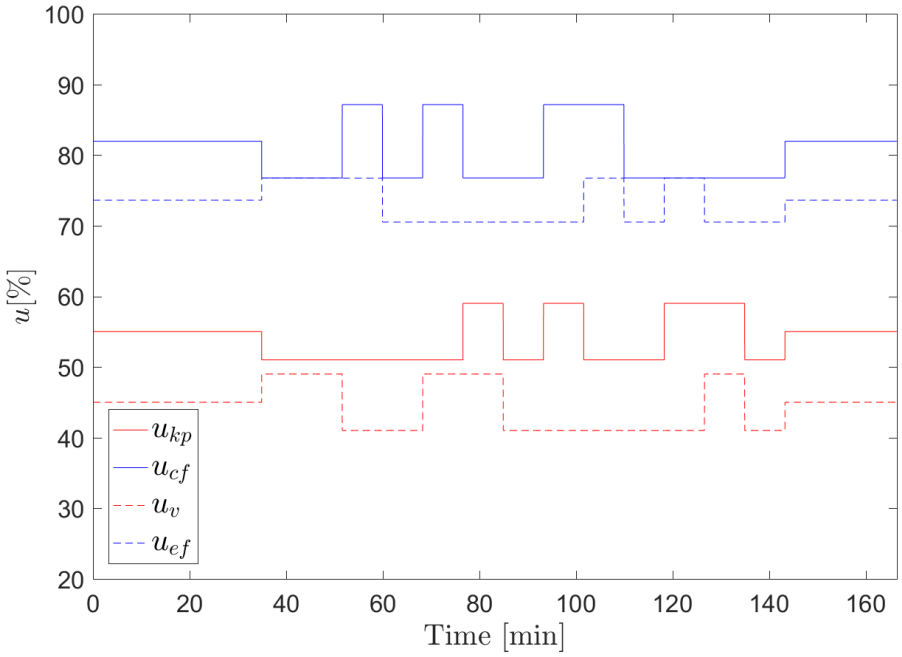

Figure 2.3: Controllable inputs were perturbed using pseudo random binary sequences generated by MATLAB's system identification toolbox.

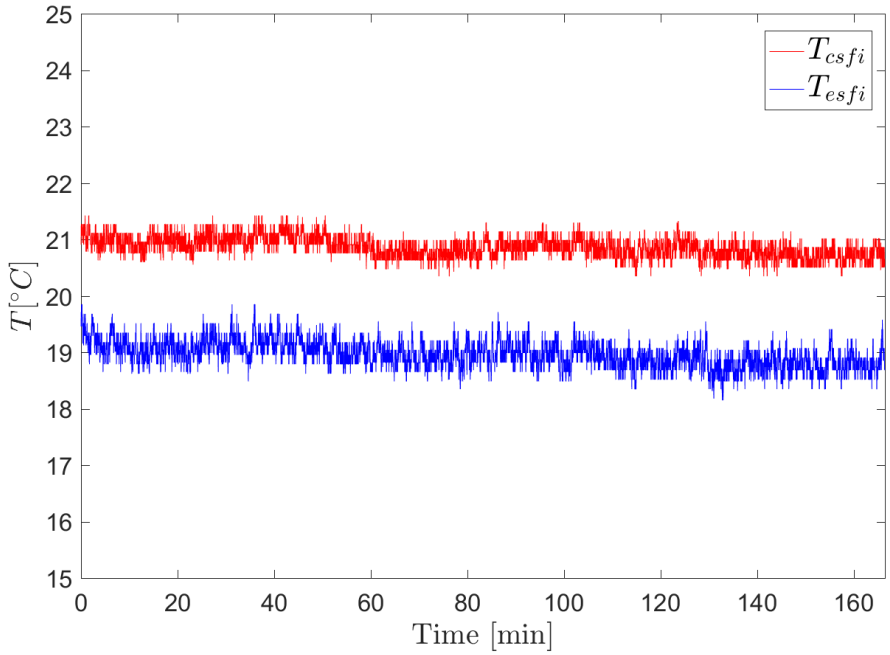

Figure 2.4: Uncontrollable air inlet temperatures remained close to constant throughout the test.

To evaluate the Thermosys model's predictive capability, Figures 2.5 through 2.8 compare the dynamic responses of the variables shown in Figure 2.1 from simulation and experiment. Apart from the discrepancy between T_2 responses in Figure 2.7, many of the simulation's outputs show trends that move in the same direction as the experimental outputs. Although the Thermosys model may not be suitable for precisely predicting the experimental system's output, the dynamic similarity between the data below suggests that the Thermosys model might be able to gage the success of control approaches. The criteria for an acceptable Thermosys model is ambiguous; experience should be used to determine the physical model's acceptability.

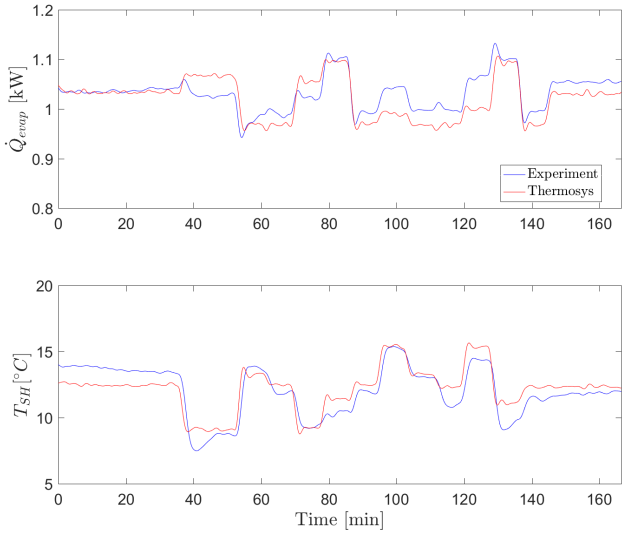

Figure 2.5: Comparison between simulated and experimental responses of the vapor compression cycle's key control outputs, $\dot{Q}_{evap}$ and T_{SH}.

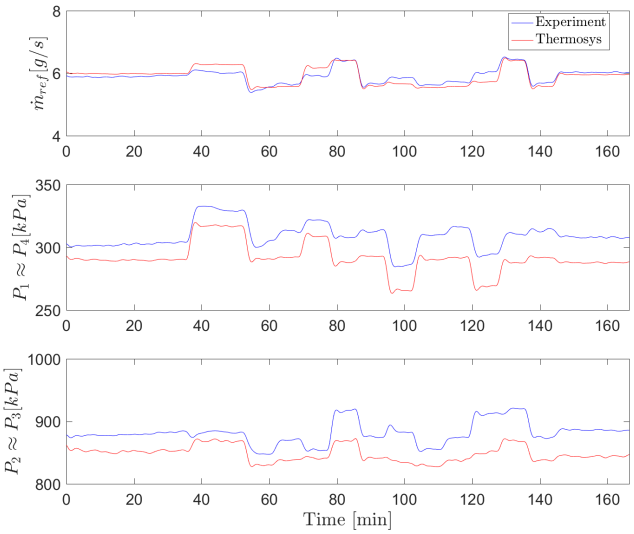

Figure 2.6: Comparison between refrigerant fluid dynamic responses observed in experiment and in Thermosys, respectively. The cycle's fundamental fluid dynamic variables are the refrigerant mass flow, $\dot{m}_{ref}$, low side pressure, P_1, and high side pressure, P_2.

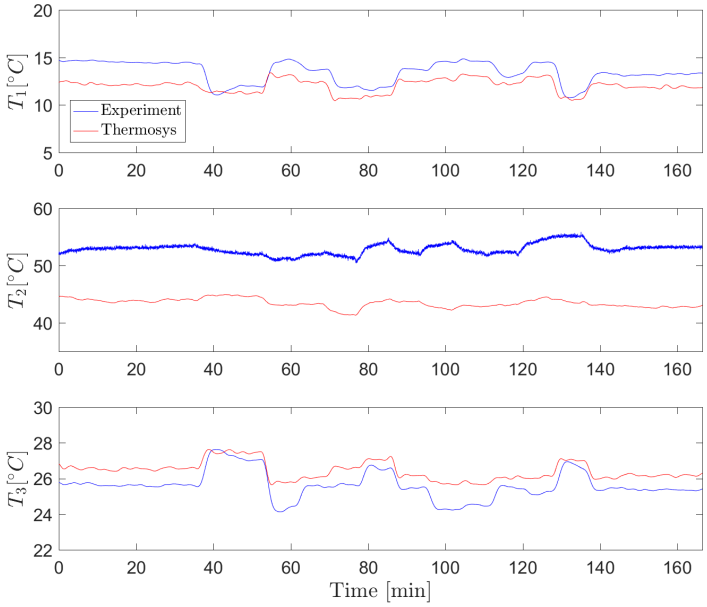

Figure 2.7: Comparison between refrigerant temperatures observed in experiment and in Thermosys, respectively. T_4 is not included because it is thermodynamically coupled to P_1.

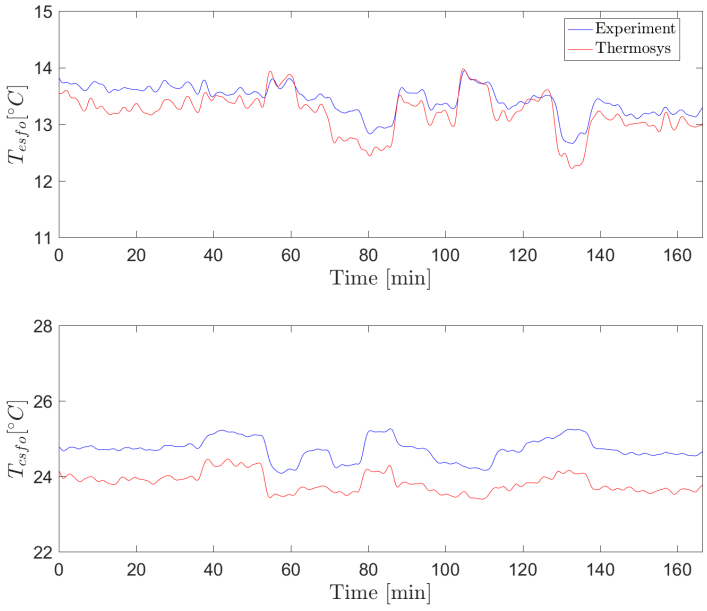

Figure 2.8: Comparison between heat exchanger air outlet temperatures observed in experiment and in Thermosys, respectively.

Building a dynamic model that is both high fidelity and accuracy can be useful for implementing intelligent control strategies. Although the subject of this thesis is model-free optimization, using a physics based system model helps build intuition which can (in theory) be applied to a wide range of systems with similar transient behavior, but dissimilar steady state values. Therefore, the main goal of the vapor compression system modeling effort was to generally capture important dynamic trends in the system without emphasis on prediction quality.

2.2.2 Power Consumption Modeling

The introduction to this chapter discussed "scoring" a system's performance using an index. While the previous section gave details about the modeling process for a VCS's characteristic pressures, mass flow, and temperatures, power consumption was conspicuously missing. In terms of scoring a VCSs efficiency, power consumption is required because it constitutes a significant operational cost and it can be measured. This section illustrates a procedure for modeling power consumption of the VCS's fans and compressor shown in Figure 2.2. Since these are generic components, the power consumption modeling procedure discussed here could likely be applied to other VCSs.

Building a model for the system's total power consumption, given by Table 2.2, involves assuming a relationship between system inputs and measurements and the component power consumption outputs. The simplest components to model are the fans; evaporator fan power, $\dot{W}_{ef}$, and condenser fan power, $\dot{W}_{cf}$, are assumed to be quadratic functions of their respective speed commands, u_{ef} and u_{cf}, given by (2.1).

$$\begin{aligned} \dot{W}_{ef} &= f_{ef}(u_{ef}) = k_{e,0} + k_{e,1}u_{ef} + k_{e,2}u_{ef}^2 \\ \dot{W}_{cf} &= f_{cf}(u_{ef}) = k_{c,0} + k_{c,1}u_{cf} + k_{c,2}u_{cf}^2 \end{aligned} \tag{2.1}$$

While $\dot{W}_{ef}$ and $\dot{W}_{cf}$ depend exclusively on the fan input signal, the compressor power consumption, $\dot{W}_{kp}$ is a function of its input signal, u_{kp}, and the pressures at the refrigerant inlet and outlet pipes, P_1 and P_2. Extending the approach to modeling fan power consumption, the compressor power consumption is modeled using a second order polynomial function of u_{kp}, P_1,

and P_2 given in (2.2) [**?**].

$$\begin{aligned}\dot{W}_{kp} = f_{ef}(u_{kp}, P_1, P_2) = k_{kp,0} + k_{kp,1}u_{kp} + k_{kp,2}P_1 + k_{kp,3}P_2 \\ + k_{kp,4}u_{kp}P_1 + k_{kp,5}u_{kp}P_2 + k_{kp,6}P_1P_2 \\ + k_{kp,7}u_{kp}^2 + k_{kp,8}P_1^2 + k_{kp,9}P_2^2\end{aligned} \tag{2.2}$$

The fast electromechanical dynamics of the fans and compressor can be neglected. Figure 2.9 shows the decomposition of the power model into evaporator fan, condenser fan, and compressor component subsystems. While the power consumption of the fans is assumed to depend on the current input, the compressor power model relies on a combination of the current compressor input and P_1 and P_2 outputs generated by the Thermosys model.

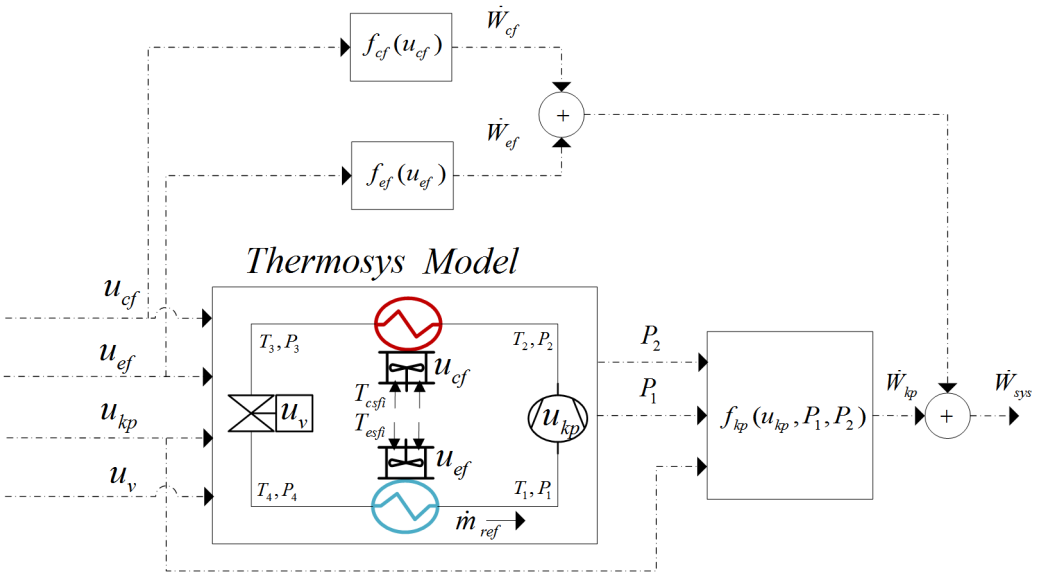

Figure 2.9: Power modeling framework implemented in Thermosys. Adding the power consumed by the fans and compressor gives the total power consumption. The electronic expansion valve's contribution is ignored.

On the system in Figure 2.2, the fan and compressor inputs each range from $0 - 100\%$, or a command that produces between 0% and 100% of the maximum capable speed. The data in in Figure 2.11 were collected by individually ramping the fan speed inputs over their full ranges. Likewise, experimental data generated for the compressor model was obtained by sweeping over a region of compressor and valve commands shown in Figure 2.10. Coefficients in (2.1) and (2.2) were found using a least-squares regression.

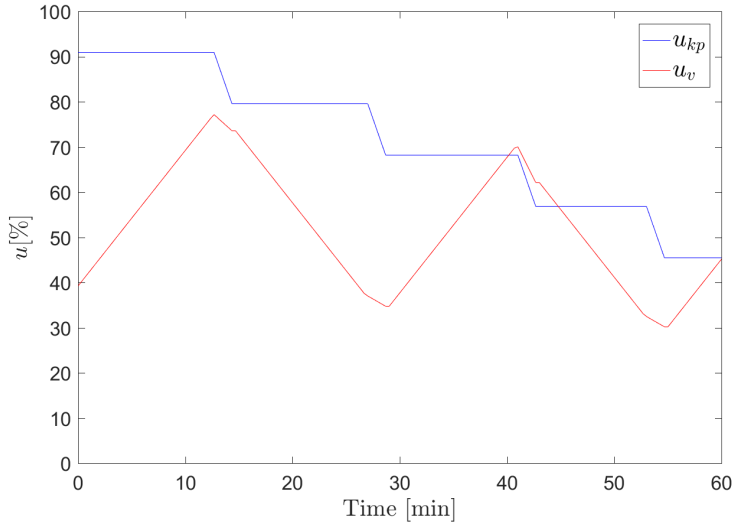

Figure 2.10: Sweep of valve and compressor inputs used to generate rich u_{kp}, P_1, and P_2 data for estimation of coefficients in (2.2).

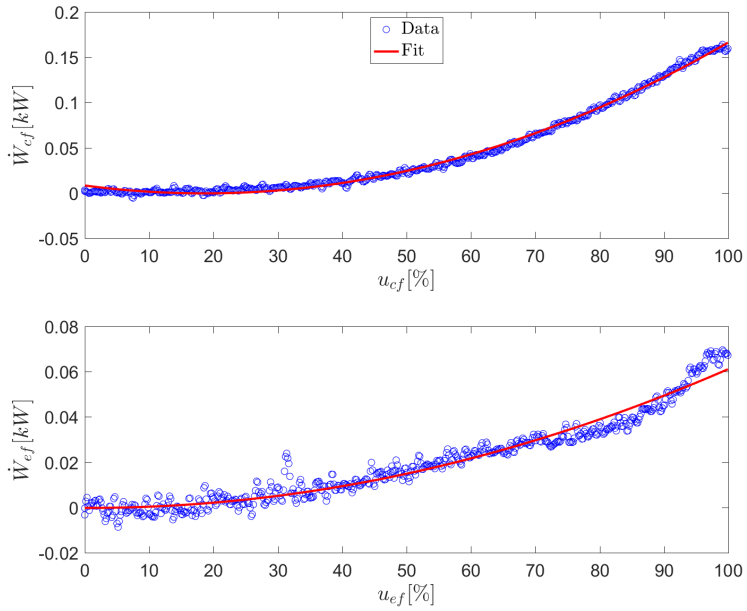

Figure 2.11: Fan power consumption fitting results. The top plot shows a second order polynomial relationship between u_{cf} and $\dot{W}_{cf}$, while the bottom plot shows a similar relationship between u_{ef} and $\dot{W}_{ef}$.

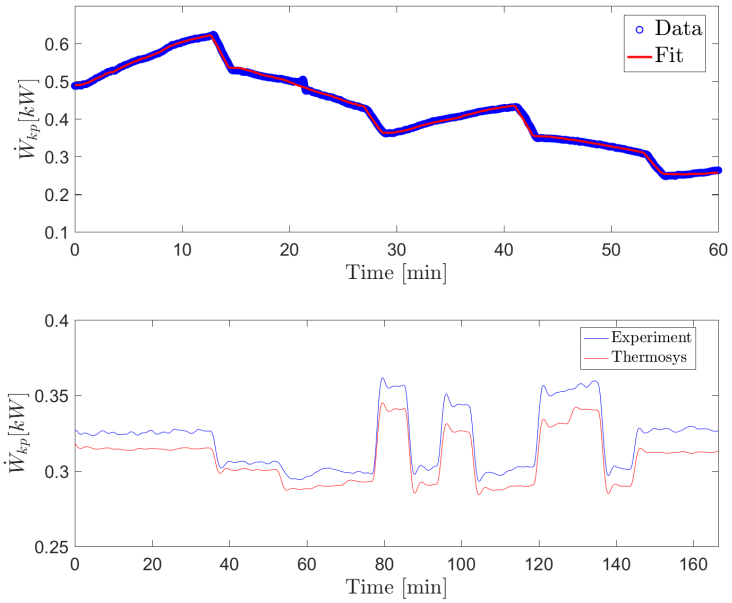

Figure 2.12: Compressor power consumption modeling results. The top plot shows a comparison between compressor power data and a polynomial fit consisting of P_1, P_2, and u_{kp} terms. The bottom plot shows a comparison between experimental data and power consumption dynamics predicted in Thermosys.

The results from Figure 2.11 and Figure 2.12 show that the experimental data validates the simulation power consumption model in two ways: the static fan models capture the quadratic trend in the experimental fan power data and the simulation captures the fundamental compressor power consumption dynamic trends observed in the pseudo random binary input perturbation experiment.

The input-output relationships in this section help "score" system operation according to a performance index, the topic of the next section.

2.3 Vapor Compression System Performance Quantification and Optimization

The process of scoring, or quantifying performance of any system requires knowledge about its operational goals and its operating environment. Starting with a general optimization problem formulation for dynamic systems,

this section shows how to simplify the problem and classify optimization of a vapor compression system's operation. Section 2.3.1 gives an optimization problem formulation that could describe a large class of dynamic systems. Section 2.3.2 shows how to reduce an optimization problem for an exponentially stable dynamic system into a corresponding steady state approximation. Finally, Section 2.3.3 applies the methodology from Section 2.3.2 to derive minimization of a vapor compression system's power consumption and compares the resulting performance index data from simulation and experiment.

2.3.1 Dynamic Optimization

Optimization problems such as the one shown below are convenient tools for determining the ways a dynamic system's performance could be quantified. Eq. (2.3) gives a formulation where a performance index, S, should be minimized using a controllable input, v. Converting to a maximization problem is simple; multiply S by -1; there is no loss of generality.

$$\begin{aligned} \min_{v} \quad & S(x, v, w) \\ \text{s.t.} \quad & \dot{x} = f(x, v, w), x(0) = x_0 \\ & y = h(x) \\ & \phi(v, y) \leq 0 \\ & \psi(v, y) = 0 \end{aligned} \tag{2.3}$$

S maps the n_x dimensional state vector, x, the n_v vector of controllable inputs, v, and the n_w vector of disturbances, w, to a scalar value that quantifies performance. This fact can be expressed using the following notation: $S : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \times \mathbb{R}_{n_w} \mapsto \mathbb{R}$. The functions below S are constraints on the optimization problem, which define relationships that must be met in order for the performance score, S, to make sense. Below, the definitions of these constraints are given:

- $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \times \mathbb{R}^{n_w} \mapsto \mathbb{R}^{n_x}$ represents a dynamic map. It is included here because optimization of any physical system obeys these dynamics. In simulation, f is a known function that must be solved numerically. In experiment, f represents the system's behavior.

$h : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_y}$ represents a map from the state vector to measured system outputs, y. A VCS's y vector could include temperatures, pressures, and power consumption.

$\phi : \mathbb{R}^{n_v} \times \mathbb{R}^{n_y} \mapsto \mathbb{R}^{n_\phi}$ ϕ represents a combination of measurements and outputs that should not exceed an upper bound. As an example, ϕ can be constructed to restrict feasible system inputs between their lower and upper bound values.

$\psi : \mathbb{R}^{n_v} \times \mathbb{R}^{n_y} \mapsto \mathbb{R}^{n_\psi}$ ψ represents an invariant combination of inputs and outputs that must hold during operation. As an example, ψ can be used to represent fixed speed inputs.

Optimization problems of the form in (2.3) can be difficult to solve, particularly if little is known about the f mapping. The next section focuses on a simplification technique that is valid when f is governed by fast, exponentially stable dynamics.

2.3.2 Steady State Optimization

This section converts the dynamic optimization problem from (2.3) into a static one where the performance index is a function of v and w only. When the dynamic equations in (2.3) are both fast and exponentially stable to equilibrium point, x^*, the following steady state approximation can be made:

$$f(x^*, v, w) = 0. \tag{2.4}$$

This approximation converts (2.3) from a problem of finding a function, v over an indefinite range of time, t, to finding an optimal v at a single time instant. Further assuming that x^* is unique for a given pair of (v, w), x^* can be expressed as a function of the system inputs as in (2.5).

$$x^* = l(v, w) \tag{2.5}$$

Substituting (2.5) into S permits a new definition: $J(v, w) \equiv S(x^*, v, w) = S(l(v, w), v, w)$, where $J : \mathbb{R}^{n_v} \times \mathbb{R}^{n_w} \mapsto \mathbb{R}$. A new equation can also be defined as a map for y: $g(v, w) \equiv h(x^*) = h(l(v, w))$, where $g : \mathbb{R}^{n_v} \times \mathbb{R}^{n_w} \mapsto \mathbb{R}$. The resulting static objective function, J, and the output function, g,

are now functions of the current controllable input and disturbance vectors only. Because the constraints in (2.3) are functions of v and y, the ϕ and ψ functions remain unchanged. Given these new definitions, the resulting steady state optimization can be written according to (2.3.2). The dynamic map, always equal to zero at steady state, is omitted.

$$\begin{aligned} \min_{v} \quad & J(v,w) \\ \text{s.t.} \quad & y = g(v,w) \\ & \phi(v,y) \leq 0 \\ & \psi(v,y) = 0 \end{aligned} \tag{2.6}$$

In the next section, (2.6) is a reference for illustrating an optimization problem for the system in Figure 2.2.

2.3.3 Steady State Vapor Compression System Optimization

Typical goals of VCS operation include meeting a cooling demand, protecting equipment, and minimizing total energy consumption. Optimizing operation involves tradeoffs: cooling provided versus total energy consumption [35] or regulating to a set point versus minimizing component wear [34].

This thesis focuses on minimization of total power consumption $\dot{W}_{sys}$ as the steady state VCS performance index. This optimization is subject to meeting a cooling capacity demand, $\dot{Q}_{evap,ref}$, and a minimum superheat, $T_{SH,ref}$, value for compressor protection [17, 12]. Because reducing superheat to its minimum allowable value is optimal [29], the optimization problems in this section will be subject to a constant superheat at the minimum value.

The performance index, J, the variables that constitute vectors, v, w, y, the constraint functions ϕ and ψ, and the full optimization problem are defined in Section 2.5.

To summarize, the four free inputs, u_{kp}, u_{cf}, u_v, and u_{ef}, are divided into separate tasks: u_{kp} and u_v are used to maintain $\dot{Q}_{evap} = \dot{Q}_{evap,ref}$ and $T_{SH} = T_{SH,ref}$, while up to 2 remaining fan speed inputs, u_{cf} and u_{ef}, are used for minimization of $\dot{W}_{sys}$. Two optimization cases were studied to compare the Thermosys simulation and with the experimental system:

Case 1 : u_{ef} is fixed at its maximum value and a single input, u_{cf}, is allowed to

change. Figure 2.13 shows the performance indexes generated in simulation and experiment by testing different values of u_{cf}. Using feedback control, the cooling capacity and superheat set points were fixed at $\dot{Q}_{evap,ref} = 1kW$ and $T_{SH,ref} = 15°C$. The other uncontrollable inputs, T_{esfi} and T_{csfi} were also unchanged. This case helps visualize a fundamental tradeoff shown in Figure 2.14: if constant $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$ are maintained, increasing the condenser fan's power consumption decreases the compressor's power consumption. The fact that a range of u_{cf} values can be chosen while meeting the performance objective enables the formulation of an optimization problem.

Case 2 : Both u_{ef} and u_{cf} can change. Figure 2.15 shows the performance indexes generated in simulation and experiment by testing different values of u_{cf} and u_{ef} under the same conditions used in Case 1.

Figure 2.13 and Figure 2.15 exhibit 2 noticeable trends: both functions appear to be convex and can be approximately described by 2nd order polynomials; and while there is disagreement on the minimum value of $\dot{W}_{sys}$, the functions have similar curvature. These trends give further confirmation that success of optimization approaches in simulation might translate to similar experimental results.

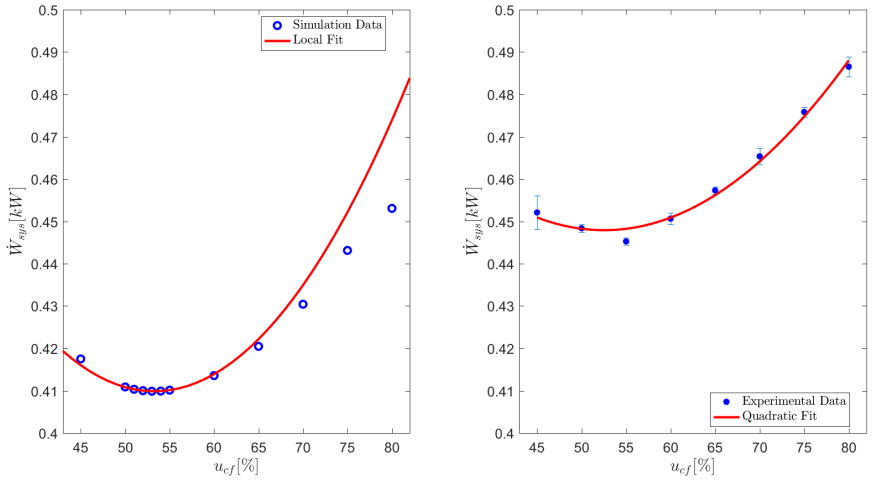

Figure 2.13: Performance index $\dot{W}_{sys}$ when u_{ef} is fixed at 100% and u_{cf} can change.

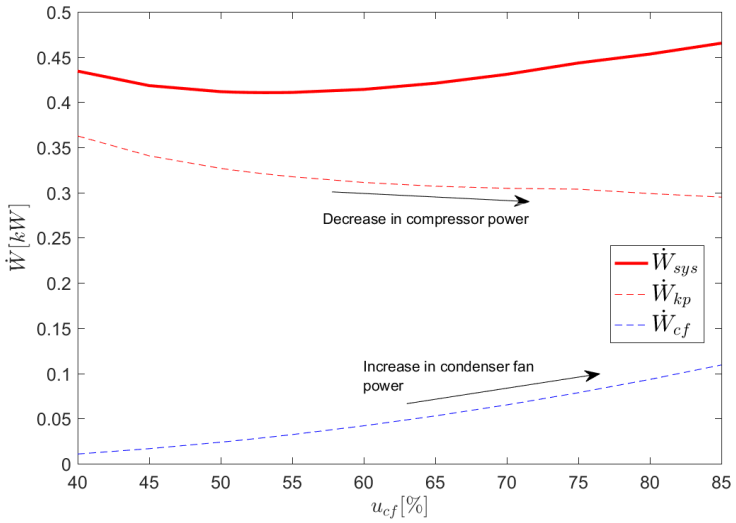

Figure 2.14: Performance index $\dot{W}_{sys}$ is broken into its components: $\dot{W}_{kp}$ and $\dot{W}_{cf}$. As fan power increases with u_{cf}, the amount of work required by the compressor to meet a given load and superheat decreases. This same relationship holds true for Case 2, where the total power varies as a function of two fan speeds.

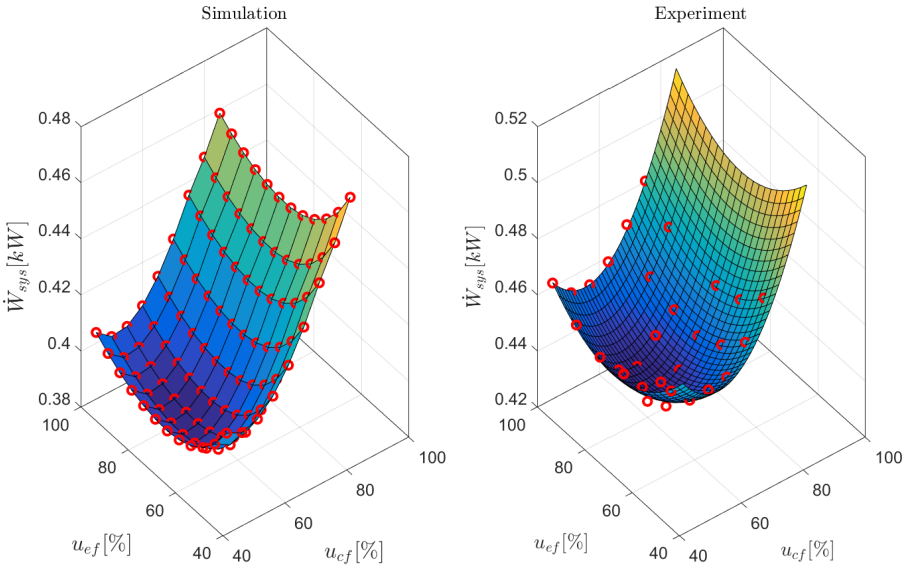

Figure 2.15: Performance index, $\dot{W}_{sys}$, when both u_{ef} and u_{cf} can change.

2.4 Conclusion

Similarities between the dynamics and performance indexes of the Thermosys model and experimental system indicate that the basic principles of VCS operation are understood, a sensible normal operation regime has been defined, and any analysis using the system model will be useful for confirmation of experimental results. In addition, total power consumption establishes a convex optimization problem where the unconstrained and global minimizing inputs are the same. Now that the optimization problems have been set up, we turn to a discussion of the model free optimization techniques studied in this thesis.

2.5 Supplemental Material

This section gives full details of the optimization problem that generates Figure 2.13 and Figure 2.15 in simulation and experiment. This study considers only unconstrained optimization, leaving the constrained case of nontrivial ϕ and ψ functions for future work. Below, the components of (2.3.2) are described for the VCS optimization problem that is a focus of this study.

J: Total system power, $\dot{W}_{sys}$.

v: Optimization inputs. In this study, the fan speed commands perform optimization, giving $v \equiv [u_{cf}\ u_{ef}]^T$.

w: -Commanded output values, $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$.
-Uncontrollable environmental conditions, T_{esfi}, T_{csfi}, and relative humidity, RH (not shown in Figure 2.1).
-Unknown changes to the system.
-In terms of known variables, $w \equiv [\dot{Q}_{evap,ref}\ T_{SH,ref}\ T_{esfi}\ T_{csfi}\ RH]^T$.

y: -Performance objectives, temperatures, pressures, and flow rate. The measurements used for this purpose are $\dot{Q}_{evap}$, T_{SH}, T_4, and T_{2-3}.
-Inputs used in closed loop feedback. In this study, u_{kp} and u_v are used to meet the commanded $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$.
-To summarize, $y = [u_{kp}\ u_v\ \dot{Q}_{evap}\ T_{SH}\ T_4\ T_{2-3}]^T$.

ϕ: -Constraints on lower and upper bounds of the system's inputs. This study focuses exclusively on unconstrained optimization, which means that ϕ can be eliminated.

ψ: -Invariant combinations of inputs and outputs. Case 1 requires using ψ to set the following constraint: $u_{ef} - u_{ef,max} = 0$. In Case 2, ψ is ignored.

It may seem counter-intuitive that inputs, u_{kp} and u_v, are system outputs. When governed by a control law to meet $\dot{Q}_{evap,ref}$ and T_{SH}, u_{kp} and u_v no longer take arbitrary values. However, unlike the temperature or cooling capacity outputs, which are system measurements by definition, any combination of n_u system inputs can be used to meet n_r commanded output values provided that $n_r \leq n_u$. In order to achieve equality between system outputs and commanded outputs at steady state, the feedback control of u_{kp} and u_v should include integral action.

$$\begin{aligned} \min_{v} \quad & \dot{W}_{sys}(v,w) \\ \text{s.t.} \quad & \begin{bmatrix} u_{kp} \\ u_v \\ \dot{Q}_{evap} \\ T_{SH} \\ T_4 \\ T_{2-3} \end{bmatrix} = \begin{bmatrix} g_1(v,w) \\ g_2(v,w) \\ g_3(v,w) \\ g_4(v,w) \\ g_5(v,w) \\ g_6(v,w) \end{bmatrix}, \; g_3(v,w) = \dot{Q}_{evap,ref}, \; g_4(v,w) = T_{SH,ref} \end{aligned} \tag{2.7}$$

Although optimization is unconstrained, the equality constraint representing $y = g(v,w)$ arises from the steady state approximation seen in Section 2.3.2 and invoking the assumption that $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$ are always within a feasible, but unknown range. For the single input optimization in Figure 2.13, an additional constraint, $u_{ef} - u_{ef,max} = 0$ should be included to show that the evaporator fan is always at max speed.

CHAPTER 3

REAL-TIME OPTIMIZATION STRATEGIES

Synonymous with steady state optimization, real-time optimization (RTO) is a term frequently used to describe a control problem for a stable system that operates near steady state for extended periods of time [20]. The previous chapter showed that reducing dynamic optimization to RTO converted the problem of finding an input function over time in (2.3) into a problem of finding a single optimal input value in (2.6). This chapter addresses the RTO problem by introducing the following techniques: extremum seeking control, self-optimizing control, and their combination.

Figure 3.1 represents a system that could be reduced to its steady state approximation if its dynamics are fast enough. The system has a performance index, J, process measurements, y, disturbances, w, and controllable inputs, v.

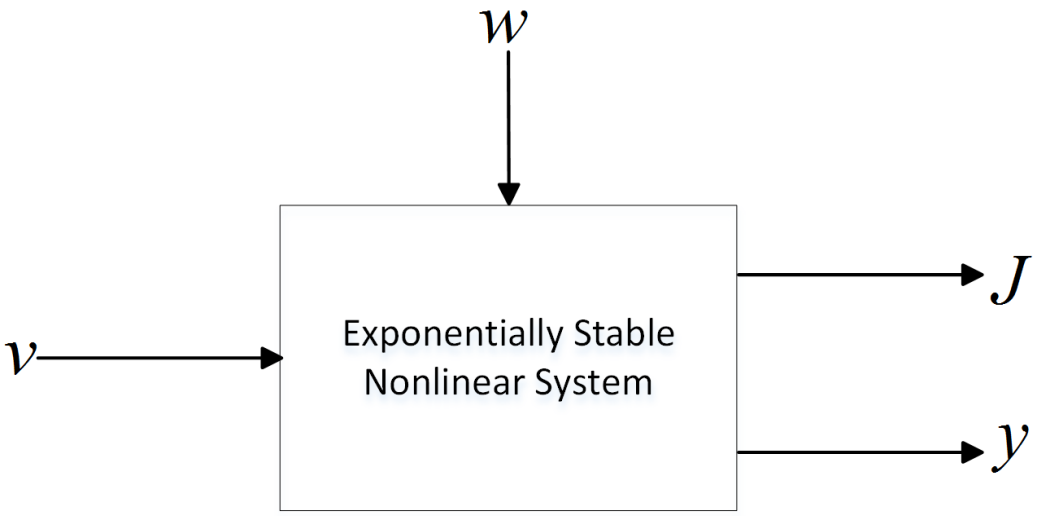

Figure 3.1: Nonlinear system with inputs and outputs that allow application of real-time optimization.

This performance index often involves some trade-off between low and high input values, which lead to a bowl-shaped steady-state optimization

problem represented by Figure 3.2, which is similar to the performance index calibrations in Figure 2.13 and Figure 2.15. While RTO problems are restricted to a steady state operation regime, the optimal input values can still be shifted by low frequency components of w. Figure 3.2 represents a typical effect of disturbances on a system's optimal inputs and performance index; the optimal v and the minimum or maximum J value are determined by w.

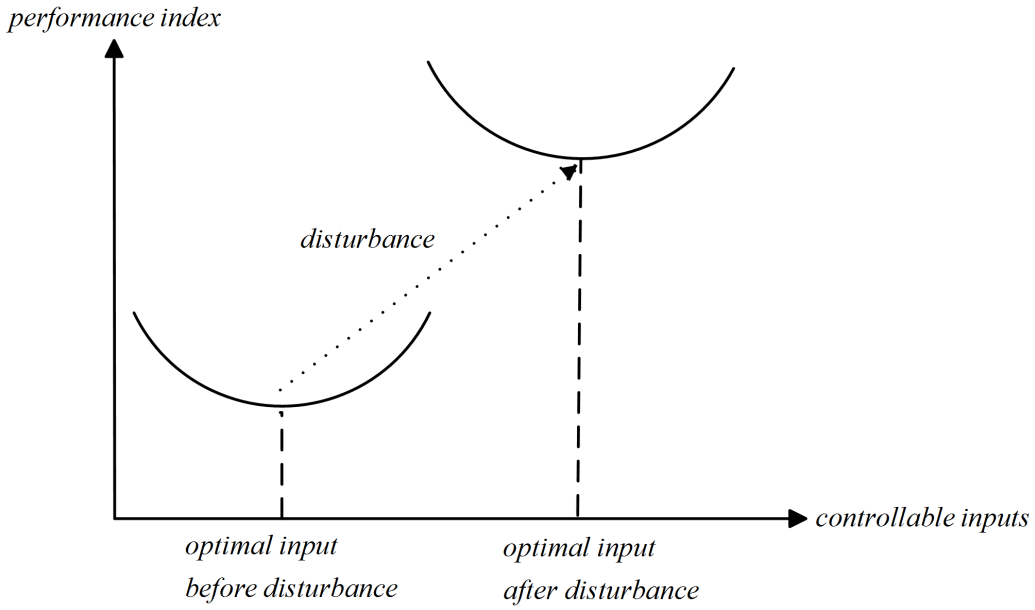

Figure 3.2: Illustration of the effect of controllable inputs and disturbances on a performance index.

Physical system models can be used to predict the optimal v for a given w. But when w is not measurable because of cost, inputs, system aging, and faults, there may be mismatch between the true optimal input and the one chosen.

Where the effects of w are challenging to capture with system model knowledge, extremum seeking control (ESC) is a possible solution. ESC works in a "model free" fashion using the process described below and depicted in Figure 3.3.

- Perturbing v
- Numerically finding derivatives of J with respect to v
- Using the derivatives to find one of the plant's local optima

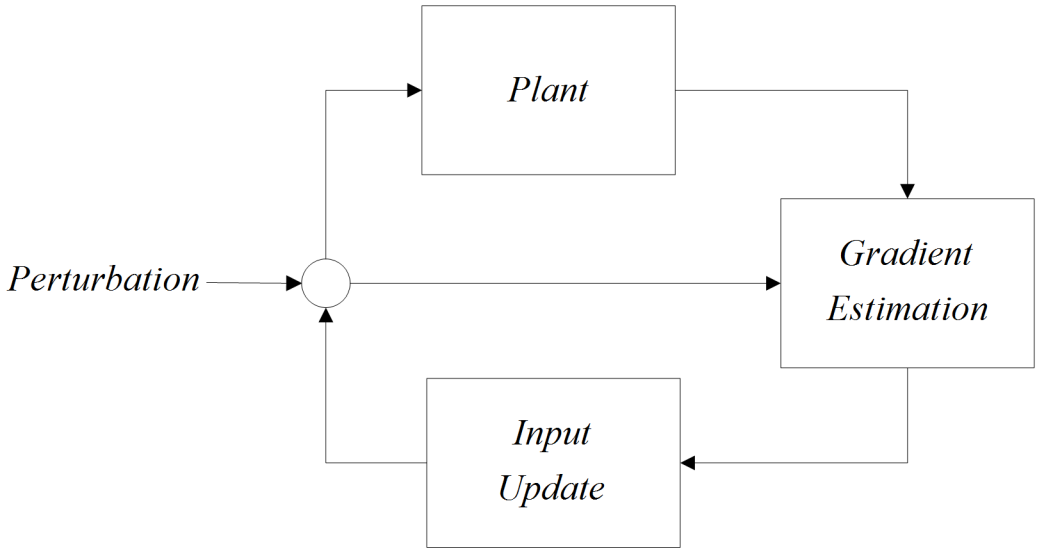

Figure 3.3: Summary of fundamental components of many extremum seeking algorithms in the literature [1].

Once the ESC reaches achieves the necessary conditions for a local, unconstrained optimum in (3.1), it hovers around the optimal input for as long as its perturbation signal continues.

$$\begin{aligned} \frac{\partial J}{\partial v} &= 0 \\ \frac{\partial^2 J}{\partial v^2} &\succ 0 \end{aligned} \tag{3.1}$$

When a disturbance changes the optimal input, the ESC reacts by adjusting its gradient estimate until it again produces a v that achieves (3.1). In the sections that follow, further details about the basic idea of extremum seeking will be given and the algorithms used in future chapters will be developed.

In contrast to ESC, self-optimizing control (SOC) is a model-based optimization method that exploits an equivalence between RTO and regulation of a combination of process outputs to a constant set-point [20]. SOC is implemented using a physical system model to perform an offline analysis of mappings between disturbances and the optimal input. Once these mappings are known, they can be related to changes in the process outputs.

A central focus of this chapter is testing unconstrained RTO approaches using simple example systems. Because the previous chapter's simulation and experimental results revealed an approximately convex and quadratic vapor compression system performance index, all of the example systems in

this chapter are convex RTO problems and most are quadratic functions.

The chapter has the following organization: Section 3.1 introduces extremum seeking in the context of optimization and estimation, discusses challenges to ESC performance, details relevant algorithms and tuning rules, and evaluates ESC performance using example systems; Section 3.2 shows how disturbances affect a system's optimal input, introduces self-optimizing control from [21], explains how a self-optimizing controller can be designed using optimal process data and combined with extremum seeking, and evaluates SOC approaches using example systems. In all sections, simulation parameters will be provided by tables preceding the plotted results. Section 3.3 offers concluding remarks.

For interested readers, all of the MATLAB code and simulation models from this chapter can be accessed using the following URL:

https://uofi.app.box.com/s/nyehk5mj927vs2qa0um1006oxpydryuh

3.1 Extremum Seeking Fundamentals

The "control" in ESC is synonymous with optimization; ESC attempts to regulate the system to zero gradient (3.1). Section 3.1.1 will develop the control aspect of ESC and focus on convergence. Section 3.1.2 will give details of a simple finite difference derivative estimation algorithm and then develop the equations and tuning rules for the two algorithms considered in this study: extremum seeking using LTI filters with demodulation and extremum seeking using time-varying filters.

3.1.1 Control

3.1.1.1 Gradient Descent

Suppose that a system with performance index, J, and vector input, $v \in \mathbb{R}^{n_v}$, gives a vector output, $z \in \mathbb{R}^{n_v}$, such that $z = \frac{\partial J}{\partial v}$ as in the figure below. In implementing control, it is desired to achieve a local minimum of $J(v)$.

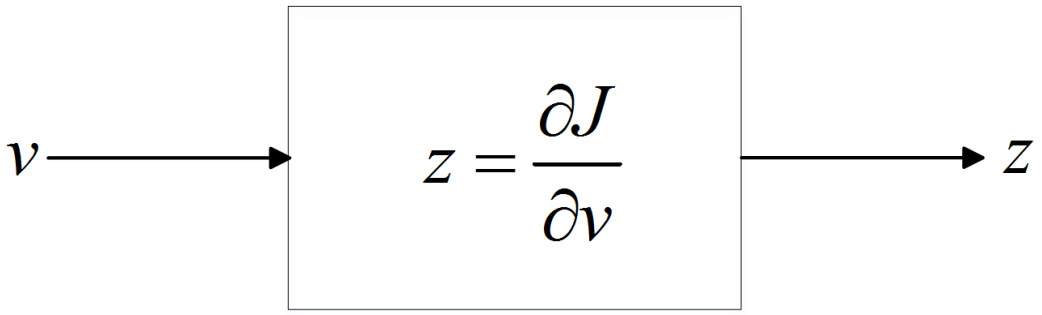

Figure 3.4: System that gives its gradient as an output.

Working in discrete-time, the input can be controlled to a local minimum point by multiplying $\hat{z}$ by a scaling matrix, K_g, and subtracting the product from the current input, v_k, to create a new input, v_{k+1}. In continuous time, the time derivative, $\dot{v}$ is set equal to the scaled gradient. Both discrete and continuous time cases are given by (3.2). The gain matrix, K_g, is a positive definite matrix. Often, K_g is a diagonal matrix, which means that v_k will decrease by some amount proportional to the steepest direction of decrease in J.

$$\begin{aligned} &Discrete\ Time: \ v_{k+1} = v_k - K_g z_k \\ &Continuous\ Time: \ \dot{v} = -K_g z \end{aligned} \tag{3.2}$$

Suppose further that J can be represented by a quadratic function with optimal input, v^*, such that $J(v) = J_0 + b^T v + \frac{1}{2} v^T Q v$, where v^T denotes the transpose of the multidimensional vector, v, and $v^* = Q^{-1}b$. When $Q = Q^T$ and all of its eigenvalues are greater than zero, Q is positive definite (denoted by $Q \succ 0$) and v^* is a global minimum point.

A quadratic $J(v)$ allows fairly straightforward convergence analyses using the fact that $z = Qv + b$. In discrete time, starting with (3.2), and setting $\tilde{v}_k \equiv v_k - v^*$, the convergence of the algorithm can be derived using the steps in (3.3).

$$\begin{aligned} &v_{k+1} = v_k - K_g(Qv_k + b) \\ &rearranging... \\ &v_{k+1} = (I - K_g Q)v_k - K_g b \\ &K_g b = K_g Q Q^{-1} b = K_g Q v^* \\ &v_{k+1} - v^* = (I - K_g Q)v_k - K_g Q v^* - v^* \\ &\Rightarrow \tilde{v}_{k+1} = (I - K_g Q)\tilde{v}_k \end{aligned} \tag{3.3}$$

The main result is the last line, which predicts that the error between the current input, v_k, and the optimal input, v^*, converges according to the eigenvalues of $(I - K_g Q)$. This fact illustrates that when implementing gradient descent, knowing the convergence rate requires an approximation of Q. A similar analysis shows that the continuous time algorithm converges according to (3.4), where $\tilde{v} \equiv v - v^*$.

$$\dot{\tilde{v}}(t) = -K_g Q v \tag{3.4}$$

It is convenient to decompose K_g into a product of the Hessian inverse estimate, $\hat{Q}^{-1}$, and a diagonal matrix, W, such that $K_g = W\hat{Q}^{-1}$. If $\hat{Q}^{-1} = Q^{-1}$, then the convergence rates in (3.3) and (3.4) can be fixed by choosing W [28], as shown in (3.5) for the discrete and continuous time cases. Below, W_d represents the discrete time version of W, while W_c represents the continuous time version of W.

$$\begin{aligned} &Discrete\ Time : \tilde{v}_{k+1} = (I - W_d\hat{Q}^{-1}Q)\tilde{v}_k \approx (I - W_d)\tilde{v}_k \\ &Continuous\ Time : \dot{\tilde{v}} = -W_c\hat{Q}^{-1}Qv \approx -W_c\tilde{v} \end{aligned} \tag{3.5}$$

The convergence rate of all inputs are governed by a single time constant, τ, when $W_c = \frac{1}{\tau}I$. In discrete time, choosing $W_d = \frac{T}{\tau}I$ will have the same effect, where T is the sample time of the discrete-time system.

3.1.1.2 Newton Descent

The previous section showed that Q must be approximately known in order to design the matrix, K_g, while achieving a desired time constant, τ, for the closed loop system's convergence. Now suppose that a system similar to the one in Figure 3.4 is given, with one major difference: the system outputs both the gradient, $z = \frac{\partial J}{\partial v}$ and the Hessian inverse, $\Gamma = (\frac{\partial^2 J}{\partial v^2})^{-1}$ as in Figure 3.5.

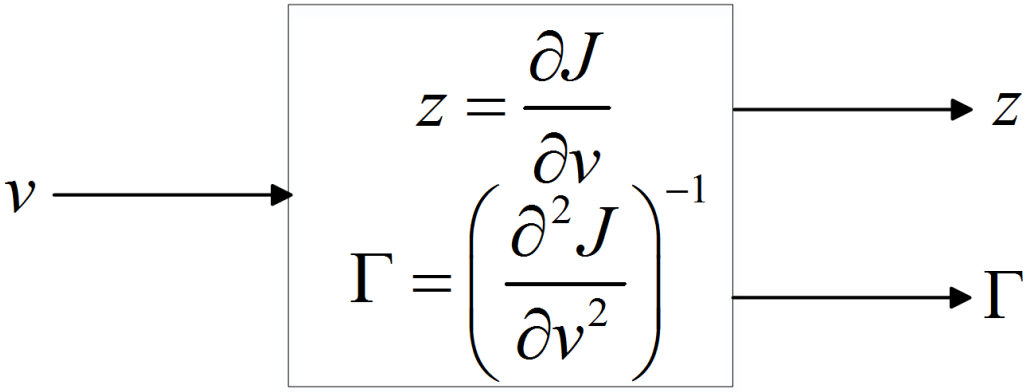

Figure 3.5: System that gives its first derivative and the inverse of its matrix of second derivatives as outputs.

Using the analysis in (3.5) from Section 3.2, including the Γ output in the control law allows specification of a convergence rate. Setting $K_g = W_d\Gamma$ in the discrete-time case or $K_g = W_c\Gamma$ in the continuous time case gives (3.6). The only difference between the analysis in this section and the analysis in Section 3.1.1.1 is that here the Hessian inverse is assumed to be perfectly known.

$$\begin{aligned} &Discrete\ Time: \ v_{k+1} = v_k - TW_d\Gamma z_k \\ &Continuous\ Time: \ \dot{v} = -W_c\Gamma z \end{aligned} \tag{3.6}$$

Making the same assumption as in the previous section that $f(v) = f_0 + b^T v + \frac{1}{2}v^T Q u$ means that $\Gamma = Q^{-1}$ is time-invariant, while $z_k = Qv_k + b$ and $z = Qv + b$. Again using the definition of $\tilde{v}$ from the previous section, performing convergence analyses on the systems in (3.6) shows that convergence can be determined entirely by choice of the W matrix as in (3.7).

$$\begin{aligned} \tilde{v}_{k+1} &= (I - W_d)\tilde{v}_k \\ \dot{\tilde{v}}(t) &= -W_c\tilde{v}(t) \end{aligned} \tag{3.7}$$

Using the Hessian inverse in the optimization law is known as Newton descent, which the same as applying Newton-Raphson root finding to a function defined by J's gradient vector, $\frac{\partial J}{\partial v}$. To show this, (3.8) applies the Newton-Raphson method to finding zeros of $f : \mathbb{R}^n \mapsto \mathbb{R}^n$, where $x \in \mathbb{R}^n$.

$$x_{k+1} = x_k - \frac{\partial f}{\partial x}^{-1} f(x_k) \tag{3.8}$$

Substituting v_k for x_k and $\frac{\partial J}{\partial v}$ for $f(x)$ recovers the Newton descent algorithm.

While Newton descent can improve convergence properties, these improvements come with the expense of using a matrix inverse. For systems where Γ is close to singular, Newton descent can lead to poor optimization performance.

The guidelines for choosing a convergence rate given in Section 3.1.1.1 are informative for choosing an extremum seeking control law. However, the systems in Figure 3.4 and Figure 3.5 are not physically realistic. The gradient, z, and the Hessian inverse, Γ, are never available for direct measurement. However, these variables can be estimated using output data. The design of these estimators will be discussed in the next sections.

3.1.2 Including Estimation

The goal of extremum seeking estimation schemes is to effectively transform the nonlinear plant equipped with a controllable input, v, and a performance index output, J, into the plants in either Figure 3.4 or Figure 3.5, as shown in Figure 3.6.

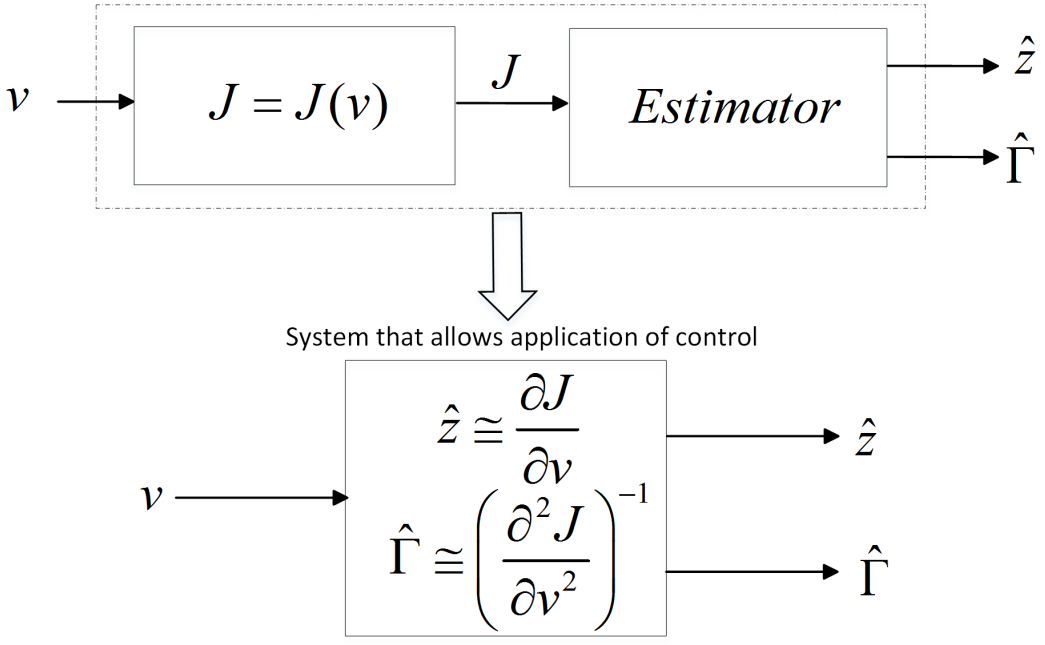

Figure 3.6: Combining the plant with an estimator creates a resulting system similar to the ones in Figure 3.4 and Figure 3.5.

Section 3.1.1 demonstrated that the minimum requirement for implementation of extremum seeking control is a system that has a gradient output. Therefore, the first part of this discussion will focus on estimation of the gradient using a simple approach.

Suppose $J = f(v)$ is a scalar performance index output of a system with scalar input, v. The gradient, $\frac{\partial J}{\partial v}$ can be estimated by perturbing the system's input by a value, δ, and using the finite difference method [36] based on the current output, J_k, and the previous output, J_{k-1}. Given a gradient estimate, the control law from Section 3.1.1.1 can be applied. This process is described by (3.9).

$$\begin{aligned} &\textit{Perturbation step}: \ v_{k-1} = v_{k-2} + \delta \\ &\textit{Estimator step}: \ \frac{\partial J}{\partial v} \approx \hat{z} = \frac{J_{k-1} - J_{k-2}}{\delta} \\ &\textit{Gradient descent step}: \ v_k = v_{k-1} - K_g \hat{z} \\ &\textit{Set } v_k = v_{k-2} \textit{ and repeat} \end{aligned} \tag{3.9}$$

In order to make the average perturbation value equal to zero over some window of time, the sign of δ should be different from one time step to the next. Figure 3.7 shows the finite difference algorithm applied to the function, $J_k = v_k^2$. While δ perturbation produces a square wave signal, the gradient descent step drives the average value of the input to $v_k = 0$.

Table 3.1: Parameters used to implement the finite difference based extremum seeking algorithm in Figure 3.7.

Parameter	Value
T	0.5
δ	0.05
K_g	0.1
v_0	1

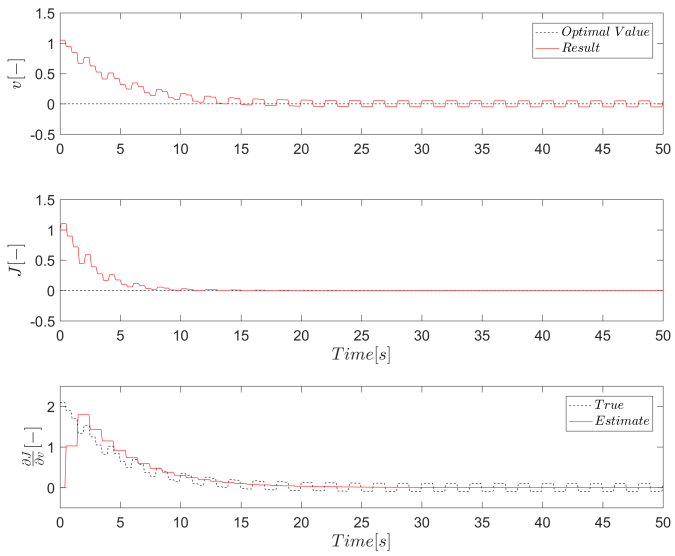

Figure 3.7: Results of finite difference extremum seeking applied to a static map, $f(v) = v^2$. After an initial transient, the gradient estimate is able to closely track the true value and find the global minimum input.

Eq. (3.10) shows that the algorithm in (3.9) can be extended to accommodate multiple inputs. For each input added, an additional perturbation step is required. Here, δ_i changes the ith component of v_k. A drawback of the approach below is that the number of time steps required for the perturbation procedure increases linearly with the number of inputs. Therefore, the algorithm's convergence speed decreases as the number of inputs increases.

$$
\begin{aligned}
& v_k \equiv [v_k^1 \ \ldots \ v_k^n] \\
& Input\ 1\ Perturbation\ step: \ v_{k-n} = v_{k-(n+1)} + \delta_1 \\
& Input\ 1\ estimator\ step: \ \frac{\partial J}{\partial v^1} \approx \frac{J_k - J_{k-1}}{\delta_1} \\
& \vdots \\
& Input\ n\ Perturbation\ step: \ v_{k-1} = v_{k-2} + \delta_n \\
& Input\ n\ estimator\ step: \ \frac{\partial J}{\partial v^n} \approx \frac{J_k - J_{k-1}}{\delta_n} \\
& \hat{z} = \left[\frac{\partial J}{\partial v^1} \ \ldots \ \frac{\partial J}{\partial v^n}\right]^T
\end{aligned}
\tag{3.10}
$$

$$
\begin{aligned}
&\textit{Gradient descent step}: \ v_k = v_{k-1} - K_g \hat{z} \\
&\textit{Set } v_k = v_{k-(n+1)} \textit{ and repeat}
\end{aligned}
$$

The algorithm can also be modified to produce an estimate of both the gradient, $\hat{z}$ and the Hessian, $\hat{\Gamma}^{-1}$, at the cost of additional perturbations before an input update. For the single input case, the algorithm is given by (3.11).

$$
\begin{aligned}
&\textit{First perturbation step}: \ v_{k-2} = v_{k-3} + \delta \\
&\textit{Second perturbation step}: \ v_{k-1} = v_{k-2} - \delta \\
&\textit{Gradient Estimator step}: \ \frac{\partial J}{\partial v} \approx \hat{z} = \frac{J_{k-1} - 2J_{k-2} + J_{k-3}}{2\delta} \\
&\textit{Hessian Estimator step}: \ \frac{\partial^2 f}{\partial v^2} \approx \hat{M} = \frac{J_{k-1} - 2J_{k-2} + J_{k-3}}{\delta^2} \\
&\textit{Newton descent step}: \ v_k = v_{k-1} - W_d \hat{\Gamma} \hat{z} \\
&\textit{Set } v_k = v_{k-3} \textit{ and repeat}
\end{aligned}
\tag{3.11}
$$

For the general n input case, the number of terms in the Hessian matrix is $\frac{n(n+1)}{2}$, while the number of gradient terms is equal to n. For a Newton update, the total number of terms required before a descent step increases according to $n + \frac{n(n+1)}{2}$. Simple reasoning has shown that increasing the number of extremum seeking inputs can increase convergence time. The next section will discuss other factors that can deteriorate extremum seeking performance.

3.1.3 Effects of Noise, Dynamics, and Disturbances

The analyses of Section 3.1.2 disregarded complications to the derivative estimation process arising from noise, dynamics, and disturbances that occur in practice. If these factors are ignored in controller implementation, then the extremum seeking algorithms may be acting on misleading information.

Noise

Noise on the performance index output is a result of random variations in sensor readings and physical system behavior that obscure the true output value (assuming that a true value exists). An issue with (3.9) is that J_k is assumed to be a true representation of J; using a single sample will place too much weight on the measurement when J is noisy.

Figure 3.8 shows that when the finite difference based extremum seeker is applied to $J = v^2$ with Gaussian noise added to the output, the controller hovers around the minimizing input as the fluctuating gradient estimates average to nearly zero.

Table 3.2: Parameters used to implement the finite difference based extremum seeking algorithm in Figure 3.8.

Parameter	Value
T	0.5
δ	0.05
K_g	0.03
v_0	1

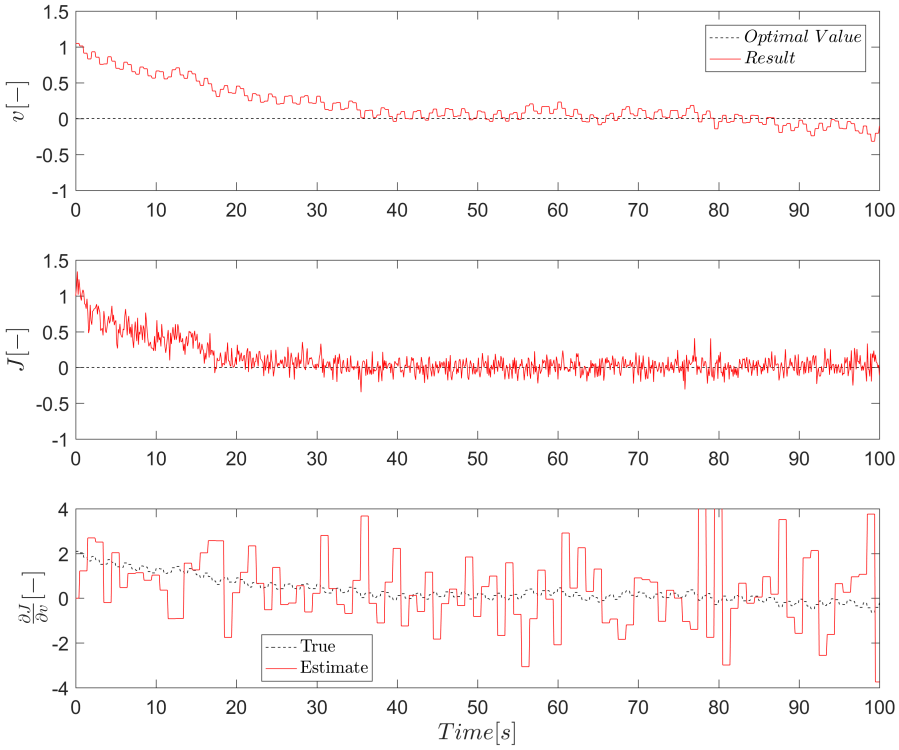

Figure 3.8: Finite difference extremum seeking applied to a static map with noise added to the output. Compared to Figure 3.7, convergence is slower and the optimal input is not always achieved.

Comparing the parameters in Table 3.1 with Table 3.2, the gradient descent gain has been reduced in Figure 3.8 in order to "average out" the effects of measurement noise. While this strategy allows convergence to the optimal input, there is a penalty on the convergence rate. One additional solution that could be explored is to low pass filter J at the expense of using a larger sampling time to wait for steady state before calculating a new gradient estimate.

Nonlinear Dynamics

Up until this point, it has been assumed that extremum seeking acts on static functions, i.e. the current output is a function of the current input. In practice, extremum seeking is applied to systems that can generally be described by (3.12). For such systems, the output's time variation following an input change must be accounted for in gradient estimation.

$$\begin{aligned} \dot{x} &= f(x, v) \\ S &= S(x, v) \end{aligned} \tag{3.12}$$

For systems with exponentially stable dynamics, steady state is approached as time goes to infinity. In practice, waiting for an infinite time to calculate the derivatives means that the extremum seeking algorithm will never converge. However, by defining an appropriate time-scale at which the system converges close enough to steady state, (3.12) can be reduced to (3.13) using the steady state approximation from the previous chapter.

$$\begin{aligned} 0 &= f(l(v), v) \\ J &= S(l(v), v) = J(v) \end{aligned} \tag{3.13}$$

Applying this strategy to the finite difference algorithm in (3.9) translates to applying a step perturbation, δ, and waiting T seconds for the system to settle, where the finite difference algorithm's sample time, T, is long enough for (3.12) to nearly reach steady state. As an example, consider the following simple system given in (3.14).

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} -10 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} -10 \\ -1 \end{bmatrix} v \\ J &= (e^{-x_1} + e^{x_2}) - 2 \end{aligned} \tag{3.14}$$

The system's slowest 2% settling time is approximately $4s$. Setting $T =$

$5s$ and applying the finite difference extremum seeking method from the previous section produces promising results shown in Figure 3.9.

Table 3.3: Parameters used to implement the finite difference based extremum seeking algorithm in Figure 3.9.

Parameter	Value
T	5
δ	0.05
K_g	0.2
v_0	1

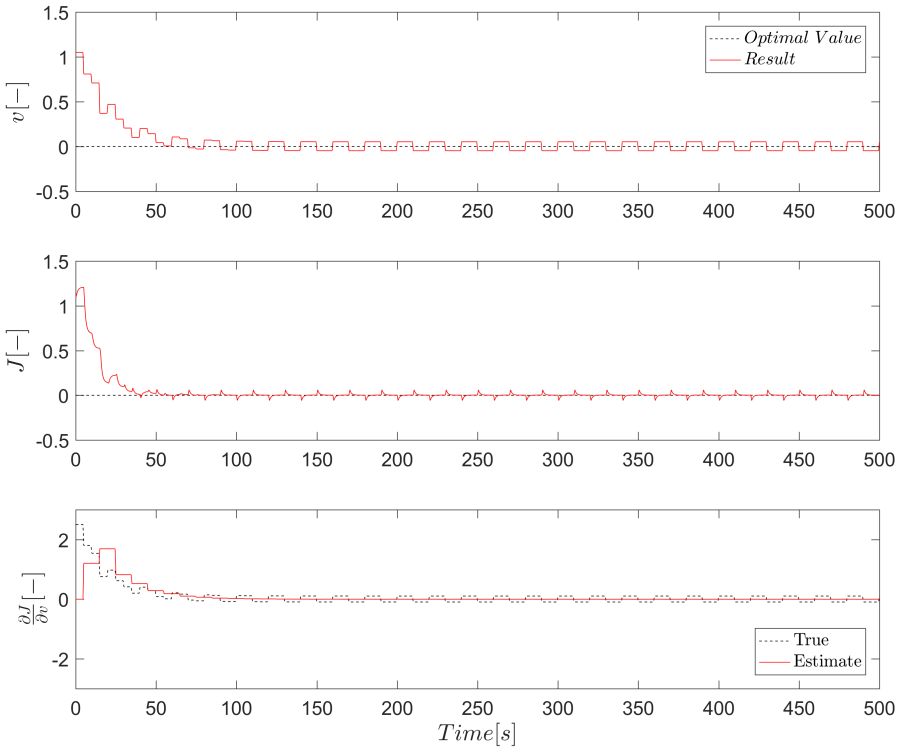

Figure 3.9: Finite difference extremum seeking applied to a dynamic system with a well chosen sample time. Despite the presence of dynamics, the optimizer is able to converge to the desired value.

Despite that the system is dynamic, waiting until the dynamics settle and assuming static behavior allows the extremum seeking algorithm to reach

the optimal input of $v = 0$. Figure 3.10 shows that when T is divided by 10, giving $T = 0.5$, the extremum seeking performance suffers because the gradient is not accurately estimated.

Table 3.4: Parameters used to implement the finite difference based extremum seeking algorithm in Figure 3.10.

Parameter	Value
T	0.5
δ	0.05
K_g	0.2
v_0	1

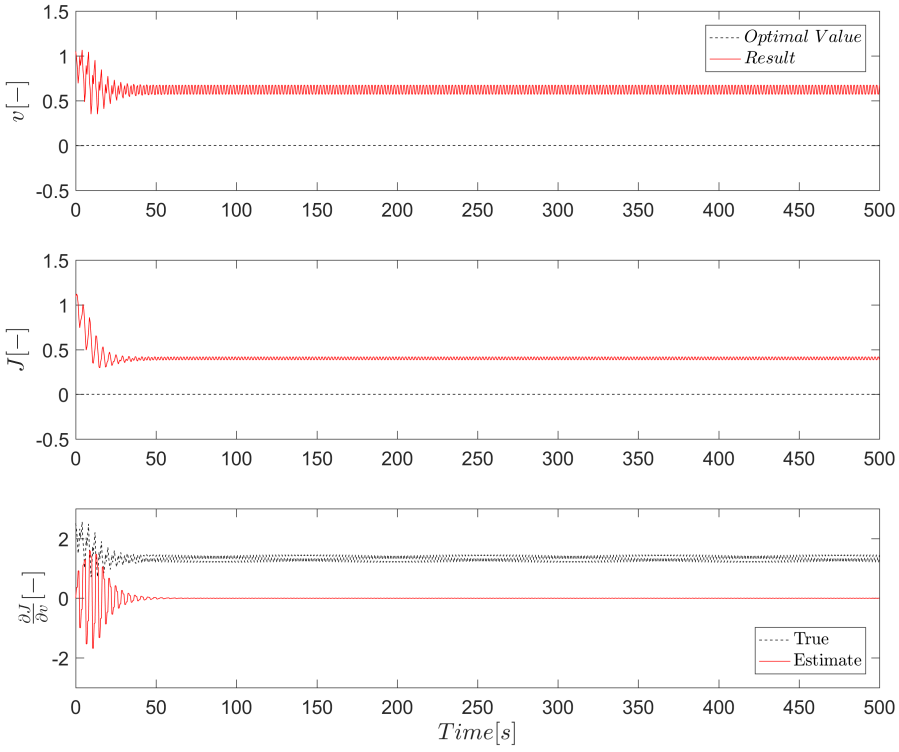

Figure 3.10: Finite difference extremum seeking applied to a dynamic system with a poorly chosen sample time. The slow system dynamic does not have sufficient time to settle, resulting in poor gradient estimates and steady state error.

Examining (3.14) shows why the ESC reached an incorrect steady state value in Figure 3.10. The x_1 component of the state vector settles after approximately $0.4s$, meaning that the contribution of e^{-x_1} will be accurately estimated when $T = 0.5$. Meanwhile, the contribution of e^{x_2} will be attenuated because the short sampling time means that there is insufficient time for x_2 to settle. Therefore, the finite difference algorithm believes that the contribution of x_2 is smaller than it is at true steady state and the convergence is biased toward $v > 0$. By perturbing the system sufficiently slowly, (3.12) behaves like (3.13) and the effects of dynamics can be circumvented at the cost of slower convergence.

Disturbances

In this section, the plant's derivatives have been assumed to be functions of solely the system input, v. In practice (as shown in Figure 3.2), w exerts influence on performance index as well. Consider the following system defined by a quadratic function of v and w:

$$\begin{aligned} J &= f(v,w) = v^2 + vw + \frac{1}{2}w^2 \\ \frac{\partial J}{\partial v} &= 2v + w, \frac{\partial J}{\partial w} = v + w \end{aligned} \tag{3.15}$$

Considering (3.15) in the context of (3.9), w may change between the measurements and result in a change in J that is different from the one produced by changing v alone, introducing model error. Figure 3.11 shows that when w changes, the gradient estimate error at around $28s$ results in an incorrect adjustment of the input. In addition to corrupting the gradient, the action of w also changes the optimal value of v, meaning that the extremum seeking algorithm must repeat its convergence to a new optimizing input.

Table 3.5: Parameters used to implement the finite difference based extremum seeking algorithm in Figure 3.11.

Parameter	Value
T	1
δ	0.05
K_g	0.2
v_0	1

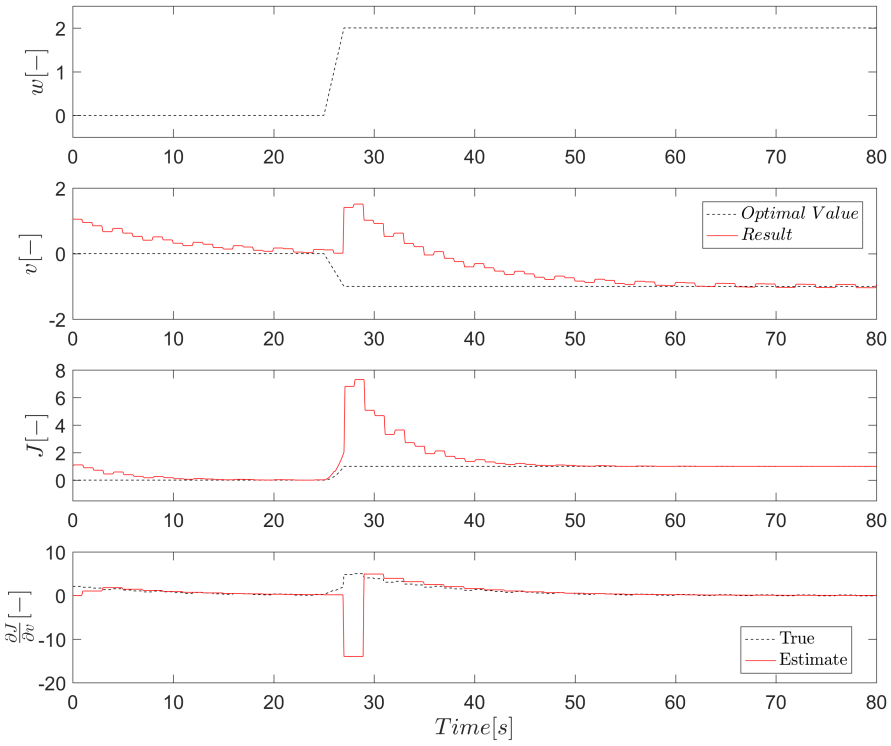

Figure 3.11: Finite difference extremum seeking result for the system in (3.15), where the action of a disturbance changes the optimal input. The change in w results in a momentarily incorrect derivative estimate.

This section demonstrated that noise, dynamics, and disturbances can trouble derivative estimation and place fundamental limitations on extremum seeking performance. The effects of noise was reduced by decreasing K_g and dynamics could be tolerated by increasing the perturbation period, T. Reducing K_g can also mitigate model uncertainty arising from disturbances, but reduces the speed of convergence to a new optimal input. The practical challenges introduced by noise, dynamics, and disturbances are fundamental limitations to extremum seeking performance that are easiest to see when estimating the gradient using finite differences.

However, the finite difference based algorithm is a simplified version of extremum seeking with a limited ability to handle measurement noise, as seen in Figure 3.8. The fact that all practical systems have measurement uncertainties motivates the exploration of more advanced extremum seeking approaches.

Relating Figure 3.3 to the discussions about gradient descent, Newton descent, estimation of gradients, and handling of plant dynamics highlights a three-part design process: first, determine a perturbation scheme that is slow enough to allow the plant to settle to a steady state; next, perform the gradient estimation at a speed slow enough to "average out" the effects of noise; finally, design the gradient or Newton descent rate to be slower than the combined perturbation and estimation scheme shown, for example, in Figure 3.6. One of the most widely used gradient estimation algorithms in the literature is based on classical LTI filter theory and signal processing [4]. A comparison between this approach, denoted *gLTI* and covered in (3.1.4.1), and the finite difference algorithm, denoted *FD*, is given in Figure 3.12, where both controllers are applied to the system, $J = v^2$.

Table 3.6: Parameters for the finite difference (*FD*) and classical extremum seeking (*gLTI*) algorithms used in Figure 3.12.

FD		*gLTI*	
Parameter	Value	Parameter	Value
T	$\pi/2$	a_1	0.1
δ	0.05	ω_1	1
K_g	0.05	ω_{HPF}	0.2
v_0	1	ω_{LPF}	0.2
		K_g	0.025
		η_0	1
		$\hat{v}_0$	1

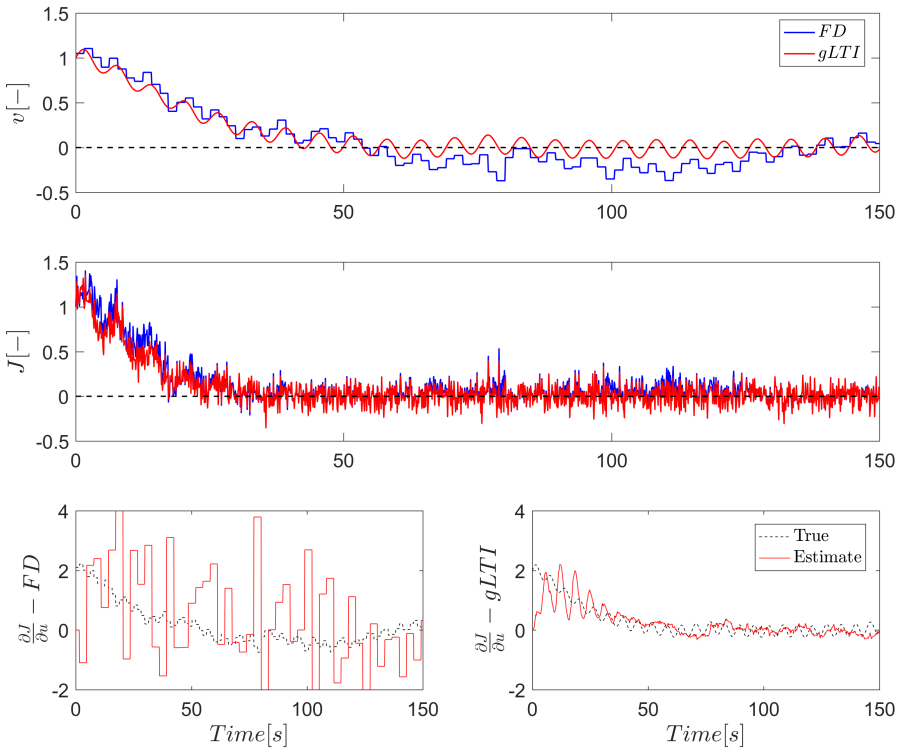

Figure 3.12: Using more advanced extremum seeking algorithms such as the *gLTI* can improve the extremum seeking performance. While the finite difference algorithm has noisy gradient estimates, the *gLTI* algorithm's estimate remains close to the true gradient.

While both the *gLTI* and *FD* algorithms converge at similar rates, *gLTI*'s input oscillates about the optimal value, indicating superior robustness to noise and demonstrating the advantage of alternative gradient estimation approaches. To understand why the *gLTI* improves on the *FD*'s performance, Section 3.1.4 explains the basics behind *gLTI* and its extension to Newton descent, denoted *nLTI*. Section 3.1.5 will detail gradient, *gTV*, and Newton, *nTV*, extremum seeking algorithms using time varying recursive least squares filters that offer additional flexibility in the perturbation and gradient estimation approaches.

3.1.4 Extremum Seeking Using LTI Filters and Demodulation

Using LTI filters with demodulation for derivative estimation is one of the most recognizable and popular extremum seeking schemes in the literature [1]. A central reason for its attractiveness is that the *gLTI*'s basic principles of operation are tractable using a combination of harmonic analysis and frequency domain filter design.

Similar to the finite difference method in the previous sections, estimation of first and second order derivatives requires the addition of a perturbation signal, $d(t)$, and a method for extracting gradient and Hessian information from the output signal. Figure 3.13 shows a possible scheme for estimating the first and second derivatives of a static map represented by $J = f(v)$. As the analysis based on [4] will show, the system below achieves (on average) the properties of Figure 3.6 to which gradient or Newton descent from Section 3.1.1.1 and Section 3.1.1.2 apply. Only the *gLTI* approach will be described in detail. For further information about estimation of the Hessian for Newton descent, consult [6].

3.1.4.1 Algorithm Overview

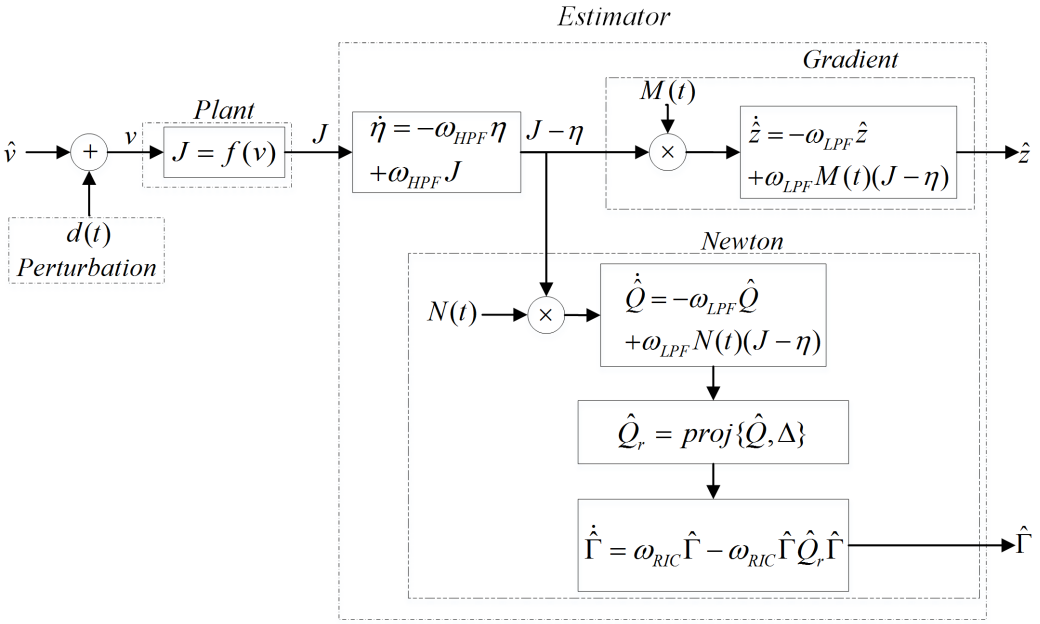

Figure 3.13: Scheme based on LTI filters with demodulation for estimation of the gradient, $\hat{z}$, and the Hessian inverse, $\hat{\Gamma}$.

In Figure 3.13, $\hat{v}$ represents the nominal system input about which perturbations are added. The total input, $v = \hat{v} + d(t)$, is fed to the plant, which produces an output signal, J. The signal, $d(t)$ represents an $\mathbb{R}^n \times \mathbb{R} \mapsto \mathbb{R}^n$ vector of sine waves such that $d(t) = [a_1 sin(\omega_1 t) \dots a_n sin(\omega_n t)]$. A simplified explanation of the gradient estimation process involves assuming that the static map can be locally represented by the first order Taylor series in (3.16).

$$J = f(v) \approx f(\hat{v}) + \frac{\partial J}{\partial v}^T d(t) \tag{3.16}$$

Eq. (3.16) shows that J is approximately the sum of a constant component, $f(\hat{v})$ and an oscillating component, $\frac{\partial J}{\partial v}^T d(t)$. If the high pass filter's cut-off frequency satisfies $\omega_{HPF} << \omega_{i,min}$, where $\omega_{i,min}$ represents the smallest perturbation frequency, then the oscillating component of the plant output will not be significantly attenuated. The high pass filter estimates $\eta \approx J(\hat{v})$, the constant term in the output signal, and removes it before demodulation by a vector of sine waves, $M(t) = [\frac{2}{a_1} sin(\omega_1 t) \dots \frac{2}{a_n} sin(\omega_n t)]^T$. Multiplying by $M(t)$ produces a vector of harmonic terms carrying signals proportional to the gradient components as shown in (3.17).

$$\frac{\partial J}{\partial v}^T d(t) M(t) = (\sum_{i=1}^{n} a_i sin(\omega_i t) \ \frac{\partial J}{\partial v_i})[\frac{2}{a_1} sin(\omega_1 t) \dots \frac{2}{a_n} sin(\omega_n t)]^T \tag{3.17}$$

Using the trigonometric identity, $sin(\omega_i t) sin(\omega_j t) = \frac{1}{2}[cos((\omega_i + \omega_j)t) + cos((\omega_i - \omega_j)t)]$, a constant $\frac{1}{2}$ term emerges from demodulation when $\omega_i = \omega_j$. Multiplication by $\frac{2}{a_i}$ in $M(t)$ produces a constant estimate of the gradient vector, $\frac{\partial J}{\partial v}$. The remaining terms in (3.17) are oscillating as in (3.18).

$$\frac{\partial J}{\partial v}^T d(t) M(t) \approx \frac{\partial J}{\partial v} + (oscillating\ terms) \tag{3.18}$$

When $\omega_{LPF} << \{minimum\ frequency\ of\ oscillating\ terms\}$, low pass filter attenuates the oscillating terms, producing $\hat{z} \approx \frac{\partial J}{\partial v}$. The low pass filter on the Hessian estimation side performs a similar function by accounting for the Harmonics present in the $N(t)(J - \eta)$ signal. Given the estimation of $\hat{z}$, continuous time gradient descent can be implemented according to (3.2).

Understanding the operation of the Newton descent algorithm requires beginning with a second order approximation to $f(v)$. To estimate $\hat{\Gamma}$, ad-

ditional steps beyond demodulation, low pass filtering, and direct inversion are required. The Hessian estimate, $\hat{Q}$, produced by the scheme in Figure 3.13 might be noisy and poorly conditioned. The authors in [37] used regularization techniques, shown by $\hat{Q}_r = proj\{\hat{Q}, \Delta\}$ in Figure 3.13, to address ill conditioned Hessians. Authors in [6] proposed a dynamic Hessian inversion scheme to avoid direct inversion of a singular $\hat{Q}$, where $\dot{\hat{\Gamma}} = \omega_{RIC}\hat{\Gamma} - \omega_{RIC}\hat{\Gamma}\hat{Q}\hat{\Gamma}$. Both approaches are discussed in the next section.

Given an introduction to the basic operational principles of the *gLTI* algorithm, full controller equations and tuning rules are provided in Section 3.1.4.2.

3.1.4.2 Controller Equations and Tuning Rules

In this section, the equations and tuning rules for gradient descent, *gLTI*, and Newton descent, *nLTI*, algorithms are presented for the general case of n system inputs in (3.19) [4, 6]. These equations can be used to implement the algorithms on an analog circuit or a computer program equipped with a differential equation solver. A discrete time version of the algorithm is also available, though this section considers the continuous time case only. The variables with 0 subscripts denote initial conditions for the controller states.

$$
\begin{aligned}
&Performance\ Index\ Output: \\
&J = J(v) \\
&Perturbation: \\
&v = \hat{v} + d(t) \\
&d = \begin{bmatrix} a_1 sin(\omega_1 t - \phi_1) & \dots & a_n sin(\omega_n t - \phi_n) \end{bmatrix} \\
&Derivative\ Estimation: \\
&\dot{\xi} = -\omega_{LPF}\xi + \omega_{LPF}(J - \eta)M \\
&\dot{\eta} = -\omega_{HPF}\eta + \eta J \\
&\dot{\hat{Q}} = -\omega_{LPF}\hat{Q} + \omega_{LPF}N(J - \eta)
\end{aligned}
\tag{3.19}
$$

$$
\begin{aligned}
&\textit{Hessian Regularization}: \\
&\hat{Q}_r = proj\{\hat{Q}, \Delta\}, \Delta \equiv \{Q \in \mathbb{S}^n : Q \succeq \delta I\} \\
&proj\{\hat{Q}, \Delta\}: \\
&\hat{Q} = V \Lambda V^* \\
&\overline{\Lambda}_i = max\{\Lambda_i, \delta\},\ i = \{1, \ldots, n\} \\
&\hat{Q}_r = V \overline{\Lambda} V^*
\end{aligned}
$$

$$
\begin{aligned}
&\textit{Dynamic Inversion}: \\
&\dot{\hat{\Gamma}} = \omega_{RIC} \hat{\Gamma} - \omega_{RIC} \hat{\Gamma} \hat{Q}_r \hat{\Gamma} \\
&\textit{Demodulation}: \\
&M = \begin{bmatrix} \frac{2}{a_1} sin(\omega_1 t) & \ldots & \frac{2}{a_n} sin(\omega_n t) \end{bmatrix} \\
&N_{i,i} = \frac{16}{a_i^2}(sin^2(\omega_i t - \phi_i) - \frac{1}{2}) \\
&N_{i,j} = \frac{4}{a_i a_j} sin(\omega_i t - \phi_i) sin(\omega_j t - \phi_j)\ i \neq j \\
&\textit{Descent}: \\
&\dot{\hat{v}} = -W_c \hat{\Gamma} \hat{z} \\
&\textit{Initialization}: \\
&\hat{v}(0) = \hat{v}_0,\ \hat{z}(0) = \hat{z}_0,\ \eta(0) = \eta_0,\ \hat{Q}(0) = \hat{Q}_0 \succ 0,\ \hat{\Gamma}(0) = \hat{Q}_0^{-1}
\end{aligned}
$$

In (3.19), ϕ_i are phase lags, $\mathbb{S}^n$ is the set of real symmetric matrices, and $\succeq$ is a non strict matrix inequality.

The *gLTI* controller can be recovered from the equations above by setting $\omega_{RIC} = 0$, $\hat{\Gamma}_0$ to a best guess of the Hessian inverse, and W_c to the desired convergence rate. To see why, setting ω_{RIC} means that $\dot{\hat{\Gamma}} = 0$ and $\hat{\Gamma}$ will not be changed from its initial value, which forms part of K_g. The following tuning heuristics based on [38] and [17] can be used to design *gLTI* and *nLTI* controllers. Because thenTV's tuning procedure is more restrictive than the gTV's, tuning rules specific to the Newton algorithm will be indicated by italics.

1. **Perturbation and Demodulation:** Dither frequencies should be as high as possible while keeping the steady state approximation valid. Let τ_{CL} from Section A.2 define the plant's 2% settling time. A corresponding estimate of the closed loop steady-state frequency threshold is $\omega_{ss} = 2\pi/\tau_{CL}$. The ESC's largest perturbation frequency, w_{max}, is

separated from ω_{ss} by a small positive constant, $0 < \sigma \leq 1$, to ensure that the extremum seeking perturbation acts in a sufficiently low frequency range.

$$w_{max} = \sigma\omega_{ss} \tag{3.20}$$

Next, dither frequencies should be distinct because each one carries a component of the gradient vector. Upon choosing ω_{max}, the remaining frequencies, indexed by distinct (i, j, k, l) should enhance identification of the gradient estimates on each channel [17]. The conditions in (3.21) can be derived by expanding the signal components in (3.17) and using trigonometric identities to identify bias terms that would corrupt the gradient estimate.

$$\begin{aligned} &\omega_{max} \geq \omega_i \\ &\omega_i \neq \omega_j \\ &\omega_i \neq 2\omega_j \\ &\omega_i \neq \omega_j + \omega_k \end{aligned} \tag{3.21}$$

nLTI Restrictions

$$\begin{aligned} &\omega_i \neq 3\omega_j \\ &\omega_i \neq \frac{1}{2}(\omega_j + \omega_k) \\ &\omega_i \neq \omega_j + \omega_k \\ &\omega_i \neq \omega_j + \omega_k \pm \omega_l \end{aligned} \tag{3.22}$$

These conditions on the dither frequency selection lead to a constrained optimization problem given in (3.23). In each case, the objective functions are given by $\Omega_{gLTI}(\bullet)$ as in (3.24) and $\Omega_{nLTI}(\bullet)$ as in (3.25). The goal is to maximize the minimum difference in conflicting harmonics of the demodulated signal.

$$\begin{aligned} \max_{\left[\omega_1 \ldots \omega_n\right]} \quad & \Omega(\left[\omega_1 \ldots \omega_n\right]) \\ \text{s.t.} \quad & \omega_i \leq \omega_{max} \\ & -\omega_i \leq 0 \end{aligned} \tag{3.23}$$

$$\Omega_{gLTI} = min\{\omega_i, \omega_i - \omega_j, \omega_i - 2\omega_j, \omega_i - (\omega_j + \omega_k)\} \tag{3.24}$$

$$\Omega_{nLTI} = min\{\Omega_{gLTI}(\left[\omega_1 \dots \omega_n\right]), \omega_i - 3\omega_j, \omega_i - \frac{1}{2}(\omega_j + \omega_k), \\ \omega_i - (\omega_j + \omega_k \pm \omega_l)\} \tag{3.25}$$

For the case of single input extremum seeking, choosing $\omega_1 = \omega_{max}$ is optimal. However, as the number of inputs increases, the dither frequency selection process is not so easy. Once frequencies have been selected, the resulting perturbation time scale is given by the slowest harmonic that must be filtered from the performance index output signal.

$$\tau_{Pert} = \frac{2\pi}{\Omega(\left[\omega_1^* \dots \omega_n^*\right])} \tag{3.26}$$

The perturbation amplitudes for each channel, a_i, are intuitively chosen large enough to produce a noticeable change in the output, J, but small enough to converge to a reasonable neighborhood of the optimal input. Delays in the perturbation signals, $\phi_1, \dots, \phi_n$ can be chosen to compensate for known delays in the plant.

2. **Estimation:** The high-pass and low-pass filter cutoff frequencies are chosen using a small positive constant, μ, and the optimal value of the dither frequency objective function, $\Omega(\left[\omega_1^* \dots \omega_n^*\right])$, as in (3.27).

$$\omega_{HPF} = \omega_{LPF} = \mu\Omega(\left[\omega_1^* \dots \omega_n^*\right]) \tag{3.27}$$

The resulting estimation time scale is given by (3.28).

$$\tau_{EST} = \frac{2\pi}{\mu\Omega(\left[\omega_1^* \dots \omega_n^*)\right])} \tag{3.28}$$

nLTI Restrictions

The Newton based ESC algorithm requires selection of a Ricatti equation frequency, ω_{RIC}, which must be separated from ω_{LPF} by another positive constant, $0 \leq \gamma \leq 1$, as in (3.29).

$$\omega_{RIC} = \gamma\omega_{LPF} \tag{3.29}$$

The convergence of the Hessian inverse estimation scheme is sensitive

to the value of ω_{RIC}. A small ω_{RIC} results in an accurate Hessian estimate, but slow convergence. Choosing ω_{RIC} too large can cause instability, especially when the measurement of J is noisy.

3. **Regularization [37]:** Regularization is a term for modifying a Hessian that is known to have incorrect properties. For instance, if a Hessian is supposed to be positive definite but the estimate is not, then regularization can be used to convert the Hessian estimate into a positive definite matrix. The regularization algorithm from (3.19) projects the Hessian estimate onto the set of matrices with a minimum eigenvalue greater than or equal to δ.

   Thus, tuning the regularization involves choosing δ, the lower bound eigenvalue of the $\hat{Q}$ estimate. For systems where the Hessian is known to be positive definite, $\delta \geq 0$ should hold. Information from a system model can be exploited to choose a reasonable $\delta > 0$ that places an upper limit on the descent rate.

4. **Descent:** In (3.19), the gradient descent gain is presented using the decomposition from (3.5): $K_g = W_c\hat{\Gamma}$. While $\hat{\Gamma} = \hat{\Gamma}_0$ for the *gLTI* algorithm, the *nLTI* algorithm continuously provides a new estimate of $\hat{\Gamma}$.

5. **Initialization:** Reasonable values must be chosen for the initial input, $\hat{v}_0$, according to the plant inputs. The high pass filter state, η_0, should be initialized to a guess of J. Setting $\hat{z}_0 = 0$ is logical because the controller should be initialized at the best guess of the optimal input.

   nLTI Initialization

   If there is significant uncertainty about the system's Hessian, $\hat{Q}_0 = \hat{\Gamma}_0^{-1}$ can be initialized to an unrealistically large value to prevent the Newton algorithm's gain, $K_g = W_c\hat{\Gamma}$, from being too large once the controller is activated. The *nLTI* algorithm compensates by correcting $\hat{\Gamma}$.

Choosing appropriate dither frequencies affects the rest of the controller design. Practical modifications to these algorithms have included using individual band pass filters in place of a high pass filter [14, 39, 40], different low pass filter designs, and other ways of regularizing the Hessian [12].

Example System

To visualize and compare the *gLTI* and *nLTI* algorithms, they are applied to the static map, $J(v) = v^2$. Setting $\hat{\Gamma}_0$ to a small initial value means that the *nLTI*'s convergence rate starts slow. However, *nLTI* can adjust its $\hat{\Gamma}$ estimate to achieve the desired convergence rate, K_g, used by the *gLTI* algorithm.

Table 3.7: Parameters for the gradient descent (*gLTI*) and Newton descent *nLTI* classical extremum seeking algorithms used in Figure 3.14.

gLTI		*nLTI*	
Parameter	Value	Parameter	Value
a_1	0.1	a_1	0.1
ω_1	1	ω_1	1
ω_{HPF}	0.2	ω_{HPF}	0.05
ω_{LPF}	0.2	ω_{LPF}	0.05
K_g	0.025	ω_{RIC}	0.05
η_0	1	η_0	1
$\hat{v}_0$	1	$\hat{v}_0$	1
		W_c	1/50
		$\hat{\Gamma}_0$	0.05
		δ	0

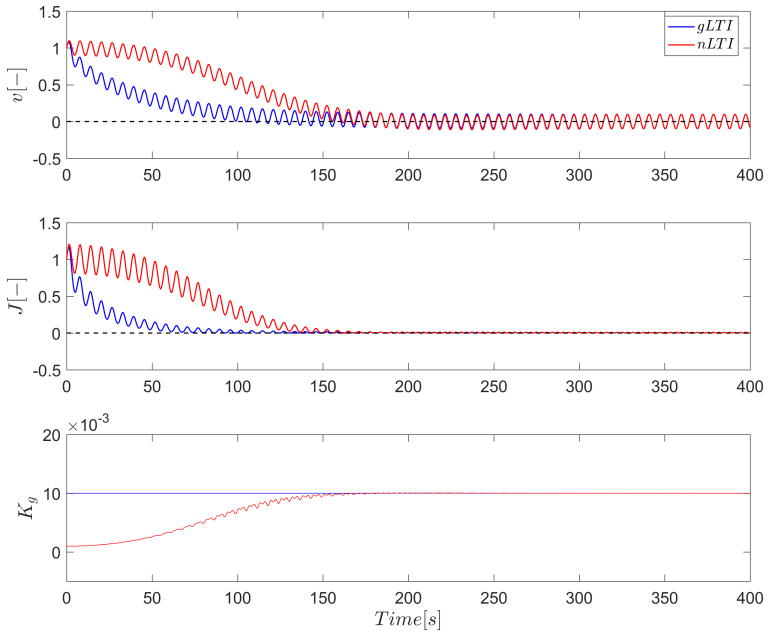

Figure 3.14: Comparison between the *gLTI* and *nLTI* algorithms. While the optimization gain must be known a-priori for the *gLTI* algorithm, the *nLTI* algorithm chooses its gain automatically.

The results show that both controllers converge to the correct optimal input value, but that the *nLTI* can do so without prior knowledge of the performance index's curvature. By contrast, the curvature must be known in order to select K_g for the gradient based algorithm. The penalty paid by the *nLTI* algorithm comes in convergence speed; *nLTI* is handicapped by less prior knowledge of the system, illustrating a fundamental trade off between the gradient based and Newton based approaches. To help address this trade off and alleviate the burden of estimating additional parameters, a time-varying derivative estimation approach is introduced in the next section.

3.1.5 Extremum Seeking with Time-Varying, Discrete Time Filters

While the *gLTI* algorithm has been studied over the last 15 years [1], extremum seeking approaches using time-varying filters have emerged relatively recently [3, 41, 2]. Unlike *gLTI* and *nLTI*, time-varying extremum seeking

algorithms use regressions on the input/output data for derivative estimation.

In this section, two regressions will be considered: a linear regression and a quadratic regression. Linear regression involves fitting a plane to the performance index output data. The planes's slope parameters can be set equal to $\hat{z}$ and used for gradient descent. This approach is denoted gTV. The quadratic regression involves fitting a second order polynomial to performance index output data. An estimate of both the gradient, $\hat{z}$, and the Hessian, $\hat{Q}$, can be constructed using the coefficients of the first and second order regression terms, respectively. This latter approach is denoted nTV. Section 3.1.5.1 will explain the basics of gTV operation to give context for the details of controller equations and tuning rules in Section 3.1.5.2.

3.1.5.1 Algorithm Overview

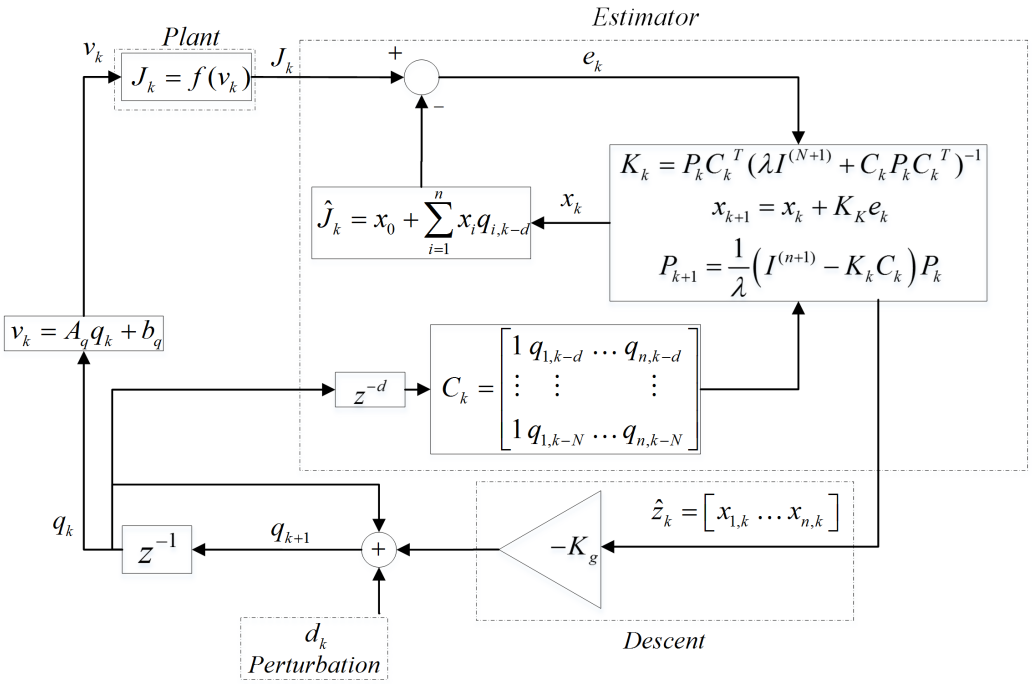

Figure 3.15: Schematic of the gTV estimation scheme. A linear regression using scaled inputs, q_k predicts the performance index output, $\hat{J}_k$, at the current time step. The algorithm presented above is essentially a hybrid of the approaches from [2] and [3].

Figure 3.15 depicts the gTV's perturbation, estimation, and descent processes. While the gTV algorithm relies on the total change in the scaled input, $q_{k+1} = q_k - K_g\hat{z}_k + d_k$, for perturbation, the $gLTI$ and $nLTI$ estimation algorithms depicted in Figure 3.13 implicitly assume that the input change comes only from the dither signal. Coupling the descent and perturbation means that input changes from the gradient descent, $K_g\hat{z}_k$, can rival input changes from the dither signal, d_k, without affecting the algorithm's ability to perform estimation. Following the perturbation, q_k undergoes an affine transformation, $v_k = A_u q_k + b_u$, to a value suited for input to the plant.

The fundamental assumption of Figure 3.15 is that the plant's sampled outputs, $[J_k, .., J_{k-N}]$, can be described by a linear combination of monomial basis functions representing a constant term and the input, $1, q_1, ...q_n$. The linear combination of these basis functions forms a plane with y intercept, $x_{0,k}$, and slope parameters, $x_{1,k}, ..., x_{n,k}$, leading to the linear regression in (3.30).

$$\begin{bmatrix} \hat{J}_k \\ \vdots \\ \hat{J}_{k-N} \end{bmatrix} = \begin{bmatrix} 1 & q_{1,k} & \dots & q_{n,k} \\ \vdots & & & \\ 1 & q_{1,k-N} & \dots & q_{n,k-N} \end{bmatrix} \begin{bmatrix} x_{0,k} \\ x_{1,k} \\ \vdots \\ x_{n,k} \end{bmatrix} = \hat{Y}_k = C_k x_k \quad (3.30)$$

The input changes from d_k and $-K_g\hat{z}$ lead to variation in the system's output data, $[J_k, ..., J_{k-N}]^T = Y_k$. The parameters, $x_{0,k}...x_{n,k}$, can be estimated by collecting at least $N > n + 1$ samples of J_k, and solving the optimization problem in (3.31).

$$\begin{aligned} \min_x \quad & \frac{1}{2}\sum_{i=k-N}^{N}(J_i - \hat{J}_i)^2 = \frac{1}{2}(Y_k - C_k x)^T(Y_k - C_k x) \\ & \frac{\partial}{\partial x}\frac{1}{2}(Y_k - C_k x)^T(Y_k - C_k x) = C_k^T C_k x_k - C^T Y_k = 0 \\ & \rightarrow x_k = (C_k^T C_k)^{-1} C^T Y_k \end{aligned} \quad (3.31)$$

A major issue with this approach is that $J_k = J(v_k)$ needs to satisfy $\frac{\partial^2 J}{\partial v^2} > 0$ to achieve optimality, though clearly the linear model predicts that $\frac{\partial^2 J}{\partial v^2} = 0^{n\times n}$, where $0^{n\times n}$ is an $n \times n$ matrix of zeros. This fact demonstrates that (3.30) is a local fit rather than a global one. The fit can be modified to emphasize local data by introducing a weighting parameter, $0 < \lambda \leq 1$, to

the performance index in (3.31). In (3.32), $\Lambda_W = diag[1\ \lambda\ \ldots\ \lambda^N]$.

$$\begin{aligned}\min_x \quad & \frac{1}{2}\sum_{i=k-N}^{N}\lambda^{N-i}(J_i-\hat{J}_i)^2 = \frac{1}{2}(Y_k-C_kx)^T\Lambda_W(Y_k-C_kx)\\ & \frac{\partial}{\partial x}\frac{1}{2}(Y_k-C_kx)^T\Lambda_W(Y_k-C_kx) = C_k^T\Lambda_WC_kx_k - C^T\Lambda_WY_k = 0\\ & \rightarrow x_k = (C_k^T\Lambda_WC_k)^{-1}C^T\Lambda_WY_k\end{aligned} \tag{3.32}$$

Using (3.32), errors due to data from the distant past will be discounted because of the $N-i$ term, rendering the fit responsive to changes in the components of x_k. Choosing λ closer to 1 helps preserve information from measurements in the past, while choosing λ closer to 0 emphasizes recent measurements.

While applying a time-dependent weighting factor makes the parameters more adaptable to time variations, one issue with (3.32) is that both the amount of data and the number of mathematical operations required to perform optimization grows with each time step. A solution is to use the recursive least-squares algorithm from [42], given by (3.33) and Figure 3.15, which preserves the memory of previous samples in the current state estimate without having to explicitly store the data.

$$\begin{aligned}e_k &= J_k - C_kx_k\\ K_k &= P_kC_k^T(\lambda I^{(N+1)} + C_kP_kC_k^T)^{-1}\\ x_{k+1} &= x_k + K_ke_k\\ P_{k+1} &= \frac{1}{\lambda}(I^{(1+n+\frac{n(n+1)}{2})} - K_kC_k)P_k\end{aligned} \tag{3.33}$$

In (3.33), P_k represents the covariance of x_k and K_k represents a gain matrix. Here, C_k can be limited to a finite number of rows, N, while data older than $k-N$ need not be stored.

As shown in Figure 3.15, the current estimate, x_k, is used to determine $\hat{J}_k$, the current estimate of J_k. The prediction error, $e_k = J_k - \hat{J}_k = J_k - C_kx_k$ is used in (3.33) to update x_k. The gradient, $\hat{z}_k$, can be extracted from the coefficients corresponding to the first order polynomial terms in the regression.

The scaled input, q_k, related to the current input, v_k, by the affine transformation in (3.34) mapping $q_k \in [-1,1]^n \mapsto v_k \in [\underline{v},\overline{v}]^n$, where $\underline{v} \in \mathbb{R}^n$

represents the anticipated lower of v_k and $\overline{v} \in \mathbb{R}^n$ represents an anticipated upper bound for v_k. The lower and upper bounds, $\underline{v}$ and $\overline{v}$, are chosen using intuition and are not necessarily representative of physical limits. The scaling is meant to improve numerical conditioning of the estimate because the monomials, $\{1, q_{1,k}, ..., q_{n,k}\}$ are orthogonal over the interval $[-1, 1]$.

$$\begin{aligned} v_k &= A_q q_k + b_q \\ A_q &\equiv diag\left[\tfrac{\overline{v}_1 - \underline{v}_1}{2} \dots \tfrac{\overline{v}_n - \underline{v}_n}{2}]\right] \\ b_q &\equiv \left[\tfrac{\overline{v}_1 + \underline{v}_1}{2} \dots \tfrac{\overline{v}_n + \underline{v}_n}{2}\right]^T \end{aligned} \tag{3.34}$$

Extending the gTV algorithm to its nTV counterpart involves adding second order monomial basis functions to the regression. This extension will be covered in the next section, which gives full controller formulations for both the gTV and nTV and guidelines for choosing tuning parameters. A simple simulation example using the static map, $J = v^2$, is given to help the reader reproduce both the gTV and nTV controllers.

3.1.5.2 Controller Equations and Tuning Rules

Leveraging the discussion from the previous section, the equations governing the gTV controller are succinctly presented in (3.35).

gTV Formulation

$$\begin{aligned} J_k &= f(v_k) \\ v_k &= A_q q_k + b_q \\ A_q &\equiv diag\left[\tfrac{\overline{v}_1 - \underline{v}_1}{2} \dots \tfrac{\overline{v}_n - \underline{v}_n}{2}]\right] \\ b_q &\equiv \left[\tfrac{\overline{v}_1 + \underline{v}_1}{2} \dots \tfrac{\overline{v}_n + \underline{v}_n}{2}\right]^T \\ e_k &= \begin{bmatrix} J_k \\ \vdots \\ J_{k-N} \end{bmatrix} - \begin{bmatrix} \hat{J}_k \\ \vdots \\ \hat{J}_{k-N} \end{bmatrix} \\ \begin{bmatrix} \hat{J}_k \\ \vdots \\ \hat{J}_{k-N} \end{bmatrix} &= \begin{bmatrix} 1 & q_{1,k} & \dots & q_{n,k} \\ \vdots & & & \\ 1 & q_{1,k-N} & \dots & q_{n,k-N} \end{bmatrix} \begin{bmatrix} x_{0,k} \\ x_{1,k} \\ \vdots \\ x_{n,k} \end{bmatrix} = C_k x_k \end{aligned} \tag{3.35}$$

$$
\begin{aligned}
&K_k = P_k C_k^T (\lambda I^{(N+1)} + C_k P_k C_k^T)^{-1} \\
&x_{k+1} = x_k + K_k e_k \\
&P_{k+1} = \frac{1}{\lambda}(I^{(1+n+\frac{n(n+1)}{2})} - K_k C_k) P_k \\
&\hat{z}_k = \begin{bmatrix} x_{0,k} & x_{1,k} & \dots & x_{n,k} \end{bmatrix}^T \\
&q_{k+1} = q_k - K_g \hat{z}_k + d_k \\
&d_k = \begin{bmatrix} a_1 sin(\omega_1 kT) & \dots & a_n sin(\omega_n kT) \end{bmatrix}^T \\
&Initial\ Conditions: \ P_0,\ x_0,\ q_0,\ C_0,\ \hat{J}_0
\end{aligned}
$$

The next section will give details about the persistence of excitation condition that must be met by d_k, which means that the components of d_k should vary enough to make the columns of C_k linearly independent.

nTV Formulation

Here, the nTV's governing equations will be developed. A natural extension of the gTV parameterization is to fit a quadratic function to the plant's output data.

$$J(v_k) \approx c + b^T v_k + \frac{1}{2} v_k^T Q v_k \tag{3.36}$$

The performance index parameters, $c \in \mathbb{R}$, $b \in \mathbb{R}^n$, and $Q \in \mathbb{S}_+^n$ (where $\mathbb{S}_+^n$ is the space of positive definite matrices) lead to a monomial parameterization with $1 + n + \frac{n(n+1)}{2}$ unknowns. As in the previous section, the regression is performed with the scaled input, q_k, which results in a modified quadratic equation given by (3.37).

$$
\begin{aligned}
\hat{J}(q_k) &\approx \bar{c} + \bar{b}^T q_k + \frac{1}{2} q_k^T \bar{Q} q_k \\
\bar{c} &\equiv c + b^T b_q + \frac{1}{2} b_q^T b_q \\
\bar{b} &\equiv b^T A_q + c^T Q A_q \\
\bar{Q} &\equiv A_q^T Q A_q
\end{aligned}
\tag{3.37}
$$

Extending the parameterization from [2] to the quadratic case, the unknown polynomial coefficients can be parameterized using a regression based on monomials of q_k as in (3.38), where ϕ creates a vector of regression polyno-

mials and $x_k \in \mathbb{R}^{1+n+\frac{n(n+1)}{2}}$ represents the polynomial coefficients.

$$\begin{aligned} \hat{J}(q_k) &= \phi(q_k)^T x_k \\ \phi(q_k) &= \begin{bmatrix} 1 & q_{1,k} \dots & q_{n,k} & q_{1,k}^2 & q_{1,k}q_{2,k} \dots & q_{n,k}^2 \end{bmatrix}^T \end{aligned} \tag{3.38}$$

Following the work of [43, 44], the numerical properties of (3.38) can be improved by applying a transformation, $V \in \mathbb{R}^{(1+n+\frac{n(n+1)}{2})\times(1+n+\frac{n(n+1)}{2})}$, to $\phi(q_k)$ that results in a family of orthogonal polynomials. As an example, given the set of Chebyshev polynomials that are orthogonal over $[-1,1]^n$, $\{1, q_{k,1}, \dots q_{k,n}, 2q_{k,1}^2 - 1, q_{k,1}q_{k,2}, \dots, 2q_{k,n}^2 - 1\}$, the regression in (3.38) can be modified such that x_k represents the estimate of the Chebyshev basis function coefficients as in (3.39).

$$\hat{J}(q_k) = (V\phi(q_k))^T x_k \tag{3.39}$$

For the case where $q_k \in \mathbb{R}^2$, the transformation $V\phi(q_k)$ is given by (3.40)

.

$$V\phi(q_k) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ q_{1,k} \\ q_{2,k} \\ q_{1,k}^2 \\ q_{1,k}q_{2,k} \\ q_{2,k}^2 \end{bmatrix} \tag{3.40}$$

The choice of V can be modified to perform regression over other families of polynomials such as Legendre or Hermite.

Using a strategy from [41], the regression in (3.39) can be further generalized to the case in (3.41) where multiple rows of delayed regression monomials are stacked into a matrix, $C_k \in \mathbb{R}^{(N+1)\times(1+n+\frac{n(n+1)}{2})}$ to enhance observability

of the unknown parameters, x_k, at each time step.

$$\begin{aligned} y_k &= C_k V^T x_k \\ y_k &\equiv \begin{bmatrix} \hat{J}(q_k) \\ \vdots \\ \hat{J}(q_{k-N}) \end{bmatrix} \\ C_k &\equiv \begin{bmatrix} \phi(q_k)^T \\ \vdots \\ \phi(q_{k-N})^T \end{bmatrix} \end{aligned} \tag{3.41}$$

The regression in (3.41) admits the application of TV estimation algorithms. Eq. (3.42) gives a modified version of (3.33) where V represents the choice of polynomial basis functions, and I^m represents the $m \times m$ identity matrix.

$$\begin{aligned} e_k &= y_k - C_k V^T x_k \\ K_k &= P_k V C_k^T (\lambda I^{(N+1)} + C_k V^T P_k V C_k^T)^{-1} \\ x_{k+1} &= x_k + K_k e_k \\ P_{k+1} &= \frac{1}{\lambda}(I^{(1+n+\frac{n(n+1)}{2})} - K_k C_k V^T) P_k \end{aligned} \tag{3.42}$$

The current parameter estimate, x_k, contains the coefficients for the orthogonal polynomial regressors, $C_k V^T$. Because it is easier to work with monomials when calculating gradients, the vector of monomial coefficients, x_k^m, is recovered using (3.43).

$$x_k^m = V^T x_k \tag{3.43}$$

Estimates of $\bar{c}$, $\bar{b}$, and $\bar{Q}$ can be directly extracted from x_k^m; the first element of x_k^m corresponds to $\hat{\bar{c}}$, the estimate of $\bar{c}$, elements $\{2 : n+1\}$ correspond to $\hat{\bar{b}}$, the estimate of $\bar{b}$, and elements $\{n+2 : 1+n+\frac{n(n+1)}{2}\}$ correspond to the unique elements of $\hat{\bar{Q}}$, the estimate of $\bar{Q}$. Referring to (3.37), the estimated gradient of $\hat{J}$ can be computed according to (3.44).

$$\hat{z} = \hat{\bar{Q}}_k q_k + \hat{\bar{b}}_k \tag{3.44}$$

Next, the eigenvalues of $\hat{\bar{Q}}_k$ are projected onto the set, Δ, defined in the *nLTI* section, to give the regularized Hessian estimate, $\hat{\bar{Q}}_{r,k}$. Implementing

the Hessian inverse scheme from [6] in the discrete time case involves inverting the difference equation, (3.45), for $\hat{H}$, a low pass filtered estimate of $\hat{\bar{Q}}_r$,

$$\hat{H}_{k+1} = \alpha \hat{H}_k + (1-\alpha)\hat{\bar{Q}}_{r,k}. \tag{3.45}$$

Setting $\hat{\Gamma}_k = \hat{H}_k^{-1}$ and using the Woodbury matrix identity [42] gives the Riccati equation for the filtered matrix inverse in (3.46).

$$\begin{aligned} \Gamma_{k+1} &= \begin{cases} (\hat{\bar{Q}}_{r,k})^{-1}, & \alpha = 0 \\ \frac{1}{\alpha}\hat{\Gamma}_k - \frac{1}{\alpha}\hat{\Gamma}_k M_k (I^n + M_k \frac{1}{\alpha}\hat{\Gamma}_k M_k)^{-1} M_k \frac{1}{\alpha}\hat{\Gamma}_k, & \alpha \neq 0 \end{cases} \\ M_k &\equiv \sqrt{(1-\alpha)\hat{\bar{Q}}_{r,k}} \end{aligned} \tag{3.46}$$

Given $\hat{\Gamma}_k$, the resulting Newton update is given by (3.47), where $W_d = TW_c$ connects the discrete time Newton descent law to the continuous time Newton descent law in (3.19).

$$\begin{aligned} q_{k+1} &= q_k - W_d \hat{\Gamma}_k (\hat{\bar{Q}} q_k + \hat{\bar{b}}) + d_k \\ v_k &= A_q q_k + b_q \end{aligned} \tag{3.47}$$

The perturbation signal, d_k, must be chosen such that the persistence of excitation condition on C_k in (3.48) is met [42].

$$\exists T \ \& \ \gamma \ s.t. \ \frac{1}{T} \sum_{i=k}^{k+T-1} V C_k^T C_k V^T > \gamma I^{1+n+\frac{n(n+1)}{2}} \tag{3.48}$$

As in the gTV case, the perturbation signal, d_k, is chosen to be a vector of sine waves. Tuning rules will now be given for the gTV and nTV algorithms. Additional guidelines for the nTV parameters will be indicated by italics.

1. **Dither Frequency Selection:** The most fundamental requirement on the dither signal is that the persistence of excitation condition in (3.48) is met. To achieve this requirement, no two frequencies, ω_i and ω_j, $i \neq j$ should be the same. Because the time varying extremum seeking algorithms do not rely on harmonics and LTI filters to estimate derivatives, there is more flexibility in choosing dither signals.

2. **Filter Design:** The main tuning parameter for the recursive least squares algorithm is the "forgetting factor," λ. Choosing a λ closer to

1 results in slower derivative estimates because there is less distinction between prediction errors in the past and present. However, there will also be less sensitivity to noise. By contrast, choosing λ closer to 0 helps the algorithm adapt to changing parameters, but can increase sensitivity to noise.

A secondary tuning parameter is the number of rows in C_k, which may increase the observability of monomial coefficients [41].

nLTI estimator for $\hat{\Gamma}_k$: The sole tuning parameter for the Hessian inverse estimator is $0 \leq \alpha < 1$, which has an effect similar to ω_{RIC} from the *nLTI* case. As α approaches 0, the Hessian inverse estimator approaches direct inversion. By contrast, choosing α close to 1 adds robustness.

Initialization: As in the LTI filter based ESC case, the nominal input should be chosen according to what is suitable for the plant. A large initial covariance, P_0, can help reduce the parameter estimation gain at the start of the recursive least squares algorithm; x_0 should set the initial gradient to zero; C_0 should be initialized according to nominal input guesses; and $\hat{J}_0$ should be a guess of the initial performance index output. For initialization of $\hat{\Gamma}_0$, use the same rules presented for the *nLTI* case.

Simulation Example

Figure 3.16 shows the *gTV* and *nTV* algorithms acting on the static map, $J_k = v_k^2$. Similar to the corresponding results for the *gLTI* and *nLTI* algorithms, the *nTV* algorithm automatically discovers the *gTV*'s optimization gain, despite a conservative initial guess of the system's Hessian.

Table 3.8: Parameters for the gTV and nTV extremum seeking algorithms used in Figure 3.16.

gTV		nTV	
Parameter	Value	Parameter	Value
a_1	0.01	a_1	0.01
ω_1	1	ω_1	1
λ	0.5	λ	0.5
N	4	N	10
T	0.1	T	0.1
A_u	1	A_u	1
b_u	0	b_u	0
K_g	0.25	W_d	$\frac{1}{50}$
v_0	1	v_0	1
x_0	$[1\ 0]^T$	x_0	$[1\ 0\ 0]^T$
P_0	$I^{(2\times 2)}$	P_0	$I^{(3\times 3)}$
		Basis	Tchebyshev
		α	0.995
		δ	0
		$\hat{\Gamma}_0$	0.05

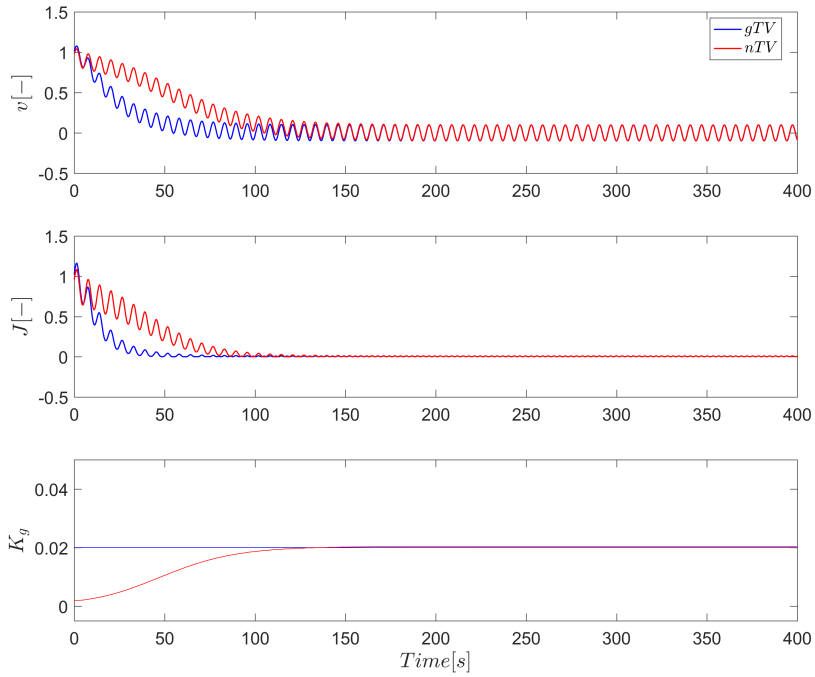

Figure 3.16: Comparison between the gTV and nTV algorithms. Similar to the comparison in Figure 3.14, while the optimization gain must be known a-priori for the gTV algorithm, the nTV algorithm chooses its gain automatically according to its estimate of the Hessian.

3.1.6 Algorithm Testing

At the close of Section 3.1.3, $gLTI$ extremum seeking was demonstrated to be less affected by measurement noise than its finite difference extremum seeking counterpart in Figure 3.12. The suggestion was that assuming proper tuning rules for the $gLTI$ approach are followed, it is a form of extremum seeking that is almost always preferred over finite differences.

Given the presentation of $nLTI$, gTV, and nTV algorithms in the previous sections, is there preferred approach? To find out, this section assesses the response of each algorithm to measurement noise and disturbances using simulation examples. Dynamics are ignored because it is assumed that their effects could be neglected by choosing sufficiently slow perturbation signals. To establish a fair comparison between approaches, the maximum dither frequency is $1 rad/s$.

The following test systems can be classified by number of inputs and whether the Hessian is a function of disturbances.

1. **Single Input:** All single input tests use (3.49), which is a function of scalars, v and w.
$$J = v^2 + vw + \frac{1}{2}w^2 \tag{3.49}$$

2. **Dual Input, Constant $\frac{\partial^2 J}{\partial v^2}$:** Defining $v \equiv [v_1 \ v_2]^T$ and $w \equiv [w_1 \ w_2]^T$, $\frac{\partial^2 J}{\partial v^2}$ of the two input system in (3.50) does not change when w changes.
$$J = \frac{1}{2}\begin{bmatrix} u_1 \\ u_2 \\ w_1 \\ w_2 \end{bmatrix}^T \begin{bmatrix} 2 & 0 & 0.55 & 0.84 \\ 0 & 2 & 0.15 & 0.62 \\ 0.55 & 0.15 & 0 & 0 \\ 0.84 & 0.62 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ w_1 \\ w_2 \end{bmatrix} \tag{3.50}$$

3. **Dual Input, Changing $\frac{\partial^2 J}{\partial v^2}$:** The static map given by (3.51) is identical to (3.50) when $[w_1 \ w_2]^T = [0 \ 0]^T$. However, nonzero w values change the entries of $Q = \frac{\partial^2 J}{\partial v^2}$.
$$J = \frac{1}{2}\begin{bmatrix} u_1 \\ u_2 \\ w_1 \\ w_2 \end{bmatrix}^T \begin{bmatrix} 2 - 0.6w_1 & 0.2w_1 + 0.2w_2 & 0.55 & 0.84 \\ 0.2w_1 + 0.2w_2 & 2 + 0.6w_2 & 0.15 & 0.62 \\ 0.55 & 0.15 & 0 & 0 \\ 0.84 & 0.62 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ w_1 \\ w_2 \end{bmatrix} \tag{3.51}$$
According to the analyses in Section 3.1.1.1, (3.51) should change the gradient descent algorithm's convergence rate. By contrast, Newton descent algorithms could conceivably react to the changing $\frac{\partial^2 J}{\partial v^2}$ values by adjusting their gains once a change in the Hessian is detected.

3.1.6.1 Single Input Comparison Between *gLTI* and *gTV*

A central observation of Section 3.1.5.1 was that time-varying extremum seeking algorithms were capable of using both the change of input from gradient descent and the dither signal to estimate the gradient parameters. By contrast, the *gLTI* algorithm assumes that only the dither perturbation to the nominal input is responsible for input changes.

To evaluate these claims, this section compares *gLTI* and *gTV* using the single input example system for cases with and without measurement noise. In each case, the algorithms were hand tuned to converge as quickly as possible to the correct average optimal input without any overshoot. The disturbance, w, steps from 0 to 2 at 50s to assess how the controllers react to a change in the optimal v brought about by an unknown input.

Figure 3.17 shows that the *gLTI* algorithm's convergence rate is fundamentally limited by its perturbation frequency. By contrast, the *gTV* is able to converge almost within one perturbation cycle. Table 3.9 gives the parameters used for the case of no additive noise.

Table 3.9: Parameters for the (*gLTI*) and *gTV* used in Figure 3.17.

gLTI		*gTV*	
Parameter	Value	Parameter	Value
a_1	0.1	a_1	0.01
ω_1	1	ω_1	1
ω_{HPF}	0.2	λ	0.5
ω_{LPF}	0.2	N	4
K_g	0.05	T	0.1
η_0	1	A_u	1
$\hat{v}_0$	1	b_u	0
		K_g	0.25
		v_0	1
		x_0	$[1\ 0]^T$
		P_0	$I^{(2\times 2)}$

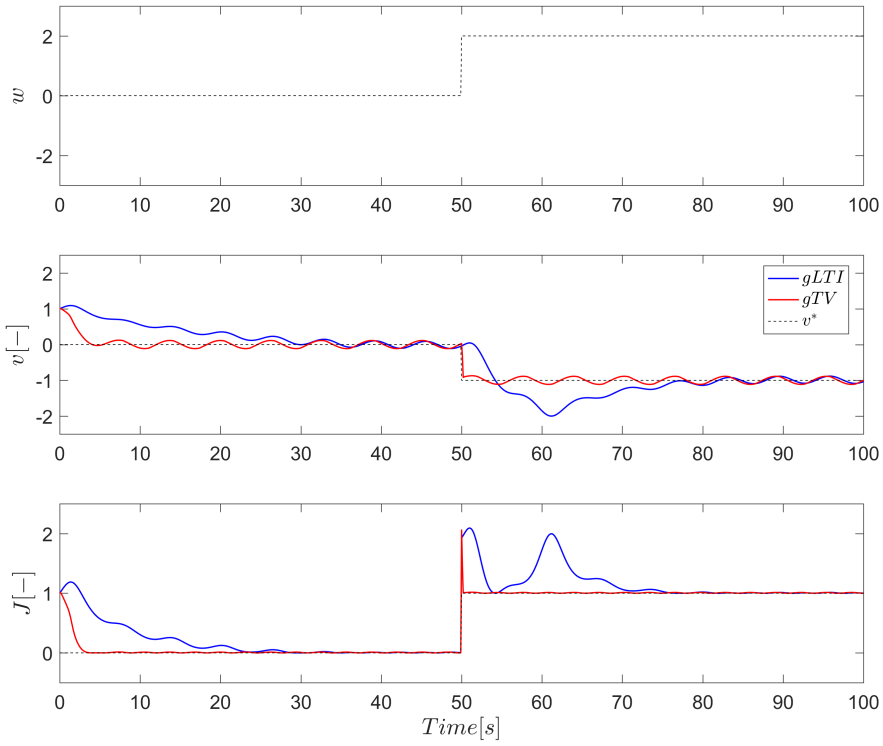

Figure 3.17: Comparison between gTV and $gLTI$. Estimation of the gradient can happen arbitrarily fast for the gTV algorithm because both the sine wave perturbation and the gradient descent contribute to excitation of the system.

The start up transient from $0 \leq t \leq 50$ confirms that the gTV algorithm can converge to the optimizing input, v^*, almost within a single perturbation cycle. The slower $gLTI$ algorithm takes multiple perturbation cycles to converge in both the $0 \leq t \leq 50$ and $50 \leq t \leq 100$ transients. While the gTV algorithm handles the change to the optimal input much better than the $gLTI$, it is hard to make any general claims about algorithm performance because both algorithms are time varying dynamic systems.

The test scenario in Figure 3.17 was repeated for the case where noise was added to J in order to obscure its true value. While the $gLTI$'s convergence rate remains the same, increasing the gTV's forgetting factor leads to a corresponding reduction of K_g and a decrease in convergence rate shown in Figure 3.18. Uncertainty in the performance index value decreases the convergence rate in both cases, reducing the gTV's advantage of being able to use the gradient descent change to the input for estimation.

Table 3.10: Parameters for the ($gLTI$) and gTV used in Figure 3.18.

$gLTI$		gTV	
Parameter	Value	Parameter	Value
a_1	0.1	a_1	0.01
ω_1	1	ω_1	1
ω_{HPF}	0.2	λ	0.9
ω_{LPF}	0.2	N	4
K_g	0.05	T	0.1
η_0	1	A_u	1
$\hat{v}_0$	1	b_u	0
		K_g	0.15
		v_0	1
		x_0	$[1\ 0]^T$
		P_0	$I^{(2\times 2)}$

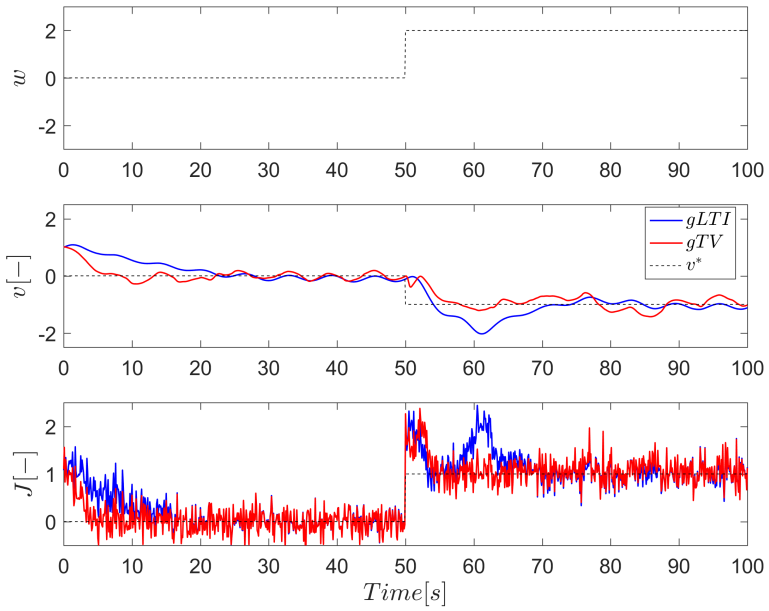

Figure 3.18: Comparison between gTV and $gLTI$ convergence rates in the presence of output noise. The gTV's convergence rate decreases relative to Figure 3.17 to compensate for the uncertainty in performance index measurements.

3.1.6.2 Multi-Input Comparison Between *gLTI* and *gTV*

This section compares the extremum seeking algorithms in a dual input optimization setting using the systems from (3.50) and (3.51) and the disturbance scenario depicted in both Figure 3.19 and Figure 3.21.

In each case, the algorithms were tuned such that they converged as quickly as possible without significant overshoot. Figure 3.19 and Figure 3.20 show that the *gTV* algorithm outperforms the *gLTI* in the multi-input test. Because the *gTV* has full knowledge of the system input, it is able to achieve a better gradient estimate and direct convergence to the optimal value in Figure 3.20. By contrast, the *gLTI* intentionally decouples frequencies on each channel using LTI filter design, which results in less accurate estimates and curved input trajectories in Figure 3.20.

Comparing Table 3.10 and Table 3.11 confirms that increasing the number of inputs decreases the estimation and convergence rates. In the *gLTI* case, increasing the number of inputs means that slower filters must be used to isolate the harmonics in different input channels. Increasing the number of inputs for the *gTV* means that there is a longer time period necessary for persistent excitation, leading to a decrease in λ.

Table 3.11: Parameters for the *gLTI* and *gTV* used in Figure 3.19, Figure 3.20, Figure 3.21, and Figure 3.22.

gLTI		*gTV*	
Parameter	Value	Parameter	Value
a_1	0.1	a_1	0.01
$[\omega_1\ \omega_2]^T$	$[1\ 0.66]^T$	$[\omega_1\ \omega_2]^T$	$[1\ 0.66]^T$
ω_{HPF}	0.2	λ	0.95
ω_{LPF}	0.2	N	4
K_g	$0.023I^{2\times2}$	T	0.1
η_0	1	A_u	$I^{2\times2}$
$\hat{v}_0$	$[1\ -1]^T$	b_u	$[0\ 0]^T$
		K_g	$0.05I^{2\times2}$
		v_0	$[1\ -1]^T$
		x_0	$[0\ 0\ 0]^T$
		P_0	$I^{(3\times3)}$

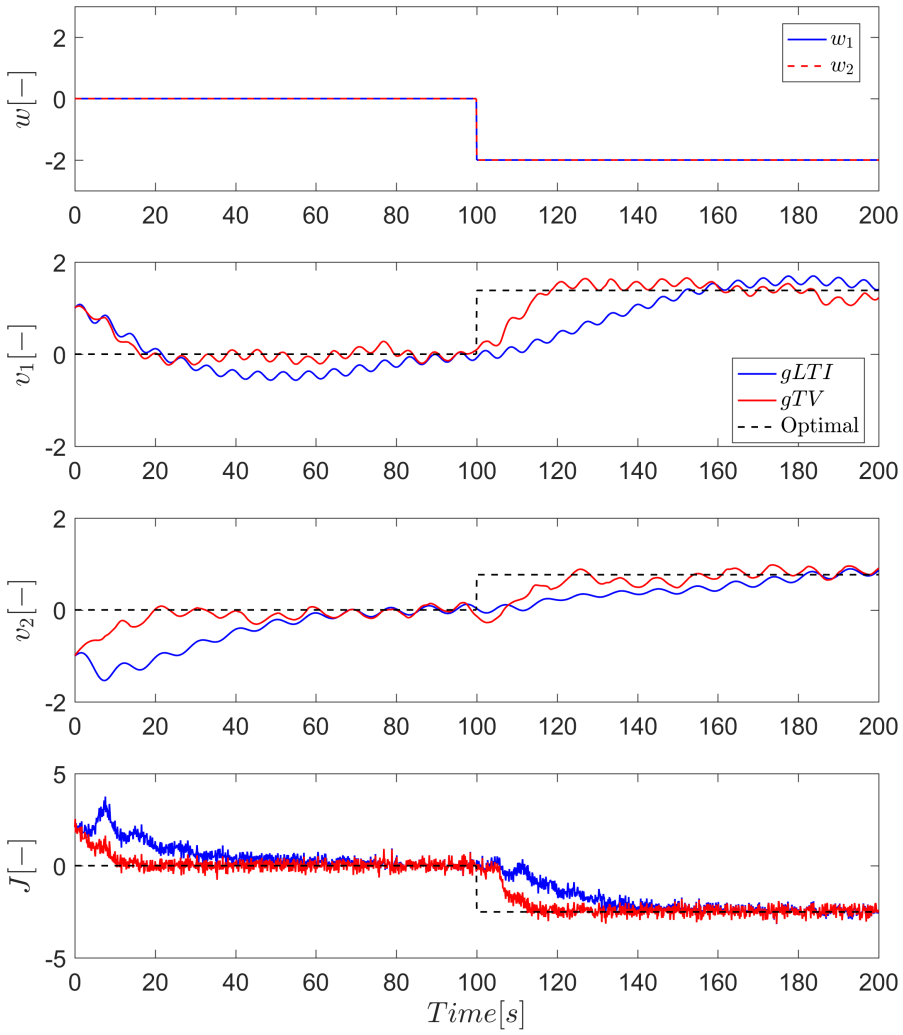

Figure 3.19: Time based comparison between the performance of gTV and $gLTI$ using the dual input system with a disturbance-invariant Hessian.

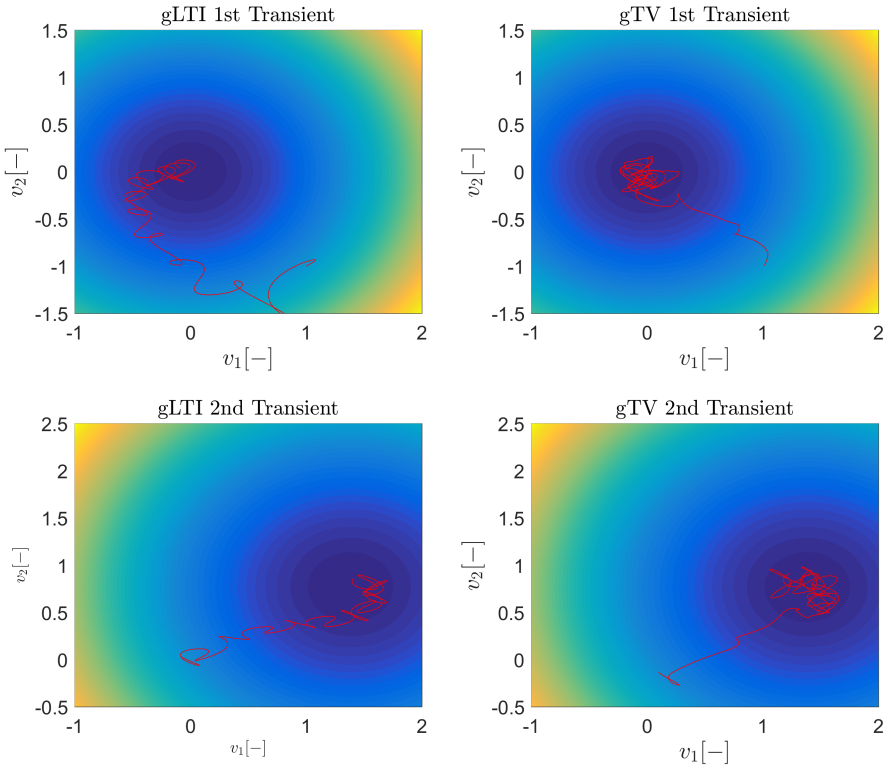

Figure 3.20: Comparison between the *gTV* and *gLTI* input trajectories.

The case of a changing Hessian poses another challenge for algorithm convergence because the performance index from 3.51 introduces coupling between channels following the disturbance step in Figure 3.21. While the convergence rates during the first transient are identical to those in Figure 3.19, the *gLTI* struggles when the cross coupling terms are introduced. By contrast, the *gTV* algorithm continues to converge on smooth gradient descent trajectories shown in Figure 3.22.

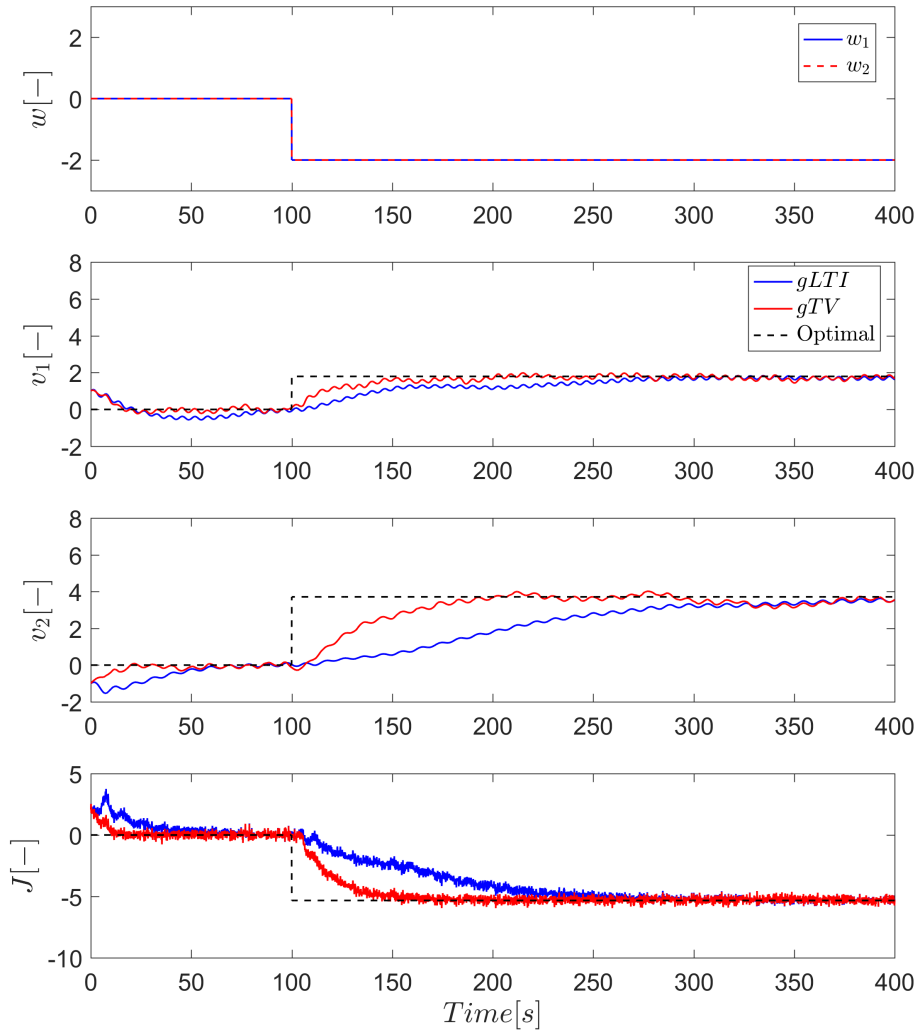

Figure 3.21: Time based comparison between the $gLTI$ and gTV convergence rates using the dual input, changing Hessian system. The changing Hessian results in a slower descent rate and indirect convergence to the optimal inputs in both cases, but less so for the gTV.

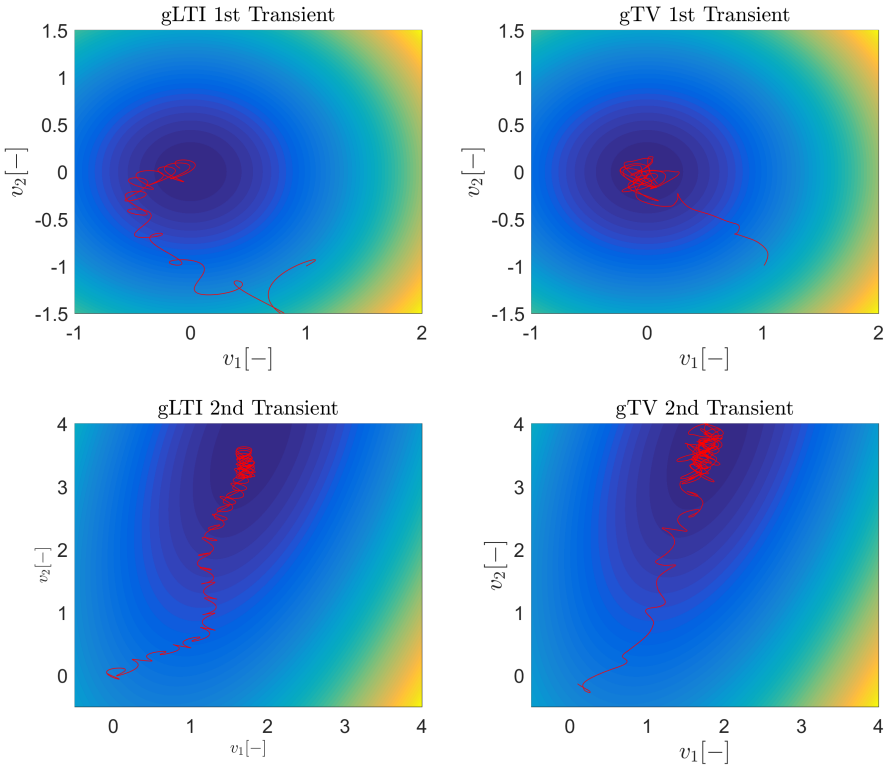

Figure 3.22: Comparison between the *gLTI* and *gTV* input trajectories using the dual input, changing Hessian system. The *gTV* takes close to the steepest descent path, while the *gLTI* takes this path following an initial transient.

The results show that the *gTV* follows a steepest descent trajectory more closely than *gLTI* in both cases. A reason for this is that the *gTV* explicitly accounts for the coupling in its regression; both components of the input are present in the estimation of $\hat{J}_k$. In addition, using the recursive least squares algorithm gives some guarantee of an optimal estimate because it attempts to minimize the sum of square errors, while using LTI filters does not.

3.1.6.3 Single-Input Comparison Between *gTV*, *nLTI*, and *nTV*

The tests from Section 3.1.6.1 and Section 3.1.6.2 show that when tuned properly, the *gTV* algorithm may be preferable to the *gLTI* in terms of achievable convergence rate. In the this section, the *gTV* will be compared to the Newton algorithms, *nLTI* and *nTV*. Assuming that the *gTV* best represents the advantages of gradient descent ESC, comparing it to New-

ton based algorithms will highlight trade offs between gradient and Newton descent.

Figure 3.23 shows that the gTV algorithm's prior knowledge of the system's Hessian helps it converge faster than both Newton descent algorithms; in this sense, the gTV approach represents an upper bound on Newton descent performance. Comparing the Newton algorithms shows that time varying estimation helps the nTV converge faster than $nLTI$.

Table 3.12: Parameters for the gTV, $nLTI$, and nTV extremum seeking algorithms used in Figure 3.23 and Figure 3.24.

gTV		$nLTI$		nTV	
Parameter	Value	Parameter	Value	Parameter	Value
a_1	0.01	a_1	0.1	a_1	0.01
ω_1	1	ω_1	1	ω_1	1
λ	0.5	ω_{HPF}	0.05	λ	0.5
N	4	ω_{LPF}	0.05	N	10
T	0.1	ω_{RIC}	0.05	T	0.1
A_u	1	η_0	1	A_u	1
b_u	0	$\hat{v}_0$	1	b_u	0
K_g	0.25	W_c	1/50	W_d	$\frac{1}{50}$
v_0	1	$\hat{\Gamma}_0$	0.05	v_0	1
x_0	$[1\ 0]^T$	δ	0	x_0	$[1\ 0\ 0]^T$
P_0	$I^{(2\times 2)}$			P_0	$I^{(3\times 3)}$
				Basis	Tchebyshev
				α	0.995
				δ	0
				$\hat{\Gamma}_0$	0.05

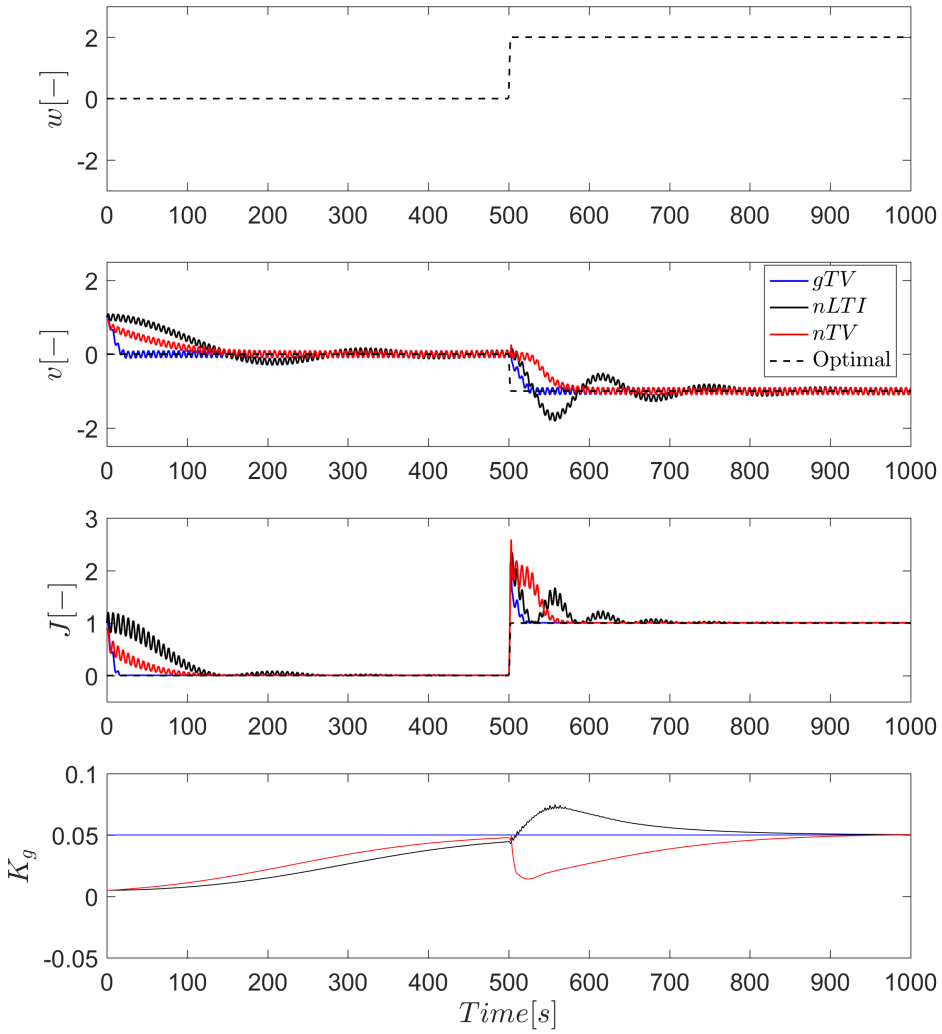

Figure 3.23: Comparison between the gTV, $nLTI$, and nTV applied to the system described by (3.49) in the absence of measurement noise. As seen in Figure 3.14 and 3.16, the $nLTI$ and nTV algorithms are able to converge to the gTV's optimization gain during the first transient from $0s \leq t \leq 50s$. The gTV's fixed gain allows it to converge quickly to the optimal input, while the optimization gains of the Newton algorithms are forced to recalibrate.

Figure 3.24 shows results similar to Figure 3.23 despite the addition of measurement noise to the output, J. The exception is that the K_g estimates are no longer smooth, suggesting that Newton algorithms are sensitive to measurement noise. To smooth the fluctuations in K_g, decrease the $nLTI$'s

ω_{RIC} value and increase the nTV's α. Both of these approaches prolong convergence times, but place less weight on faulty individual estimates.

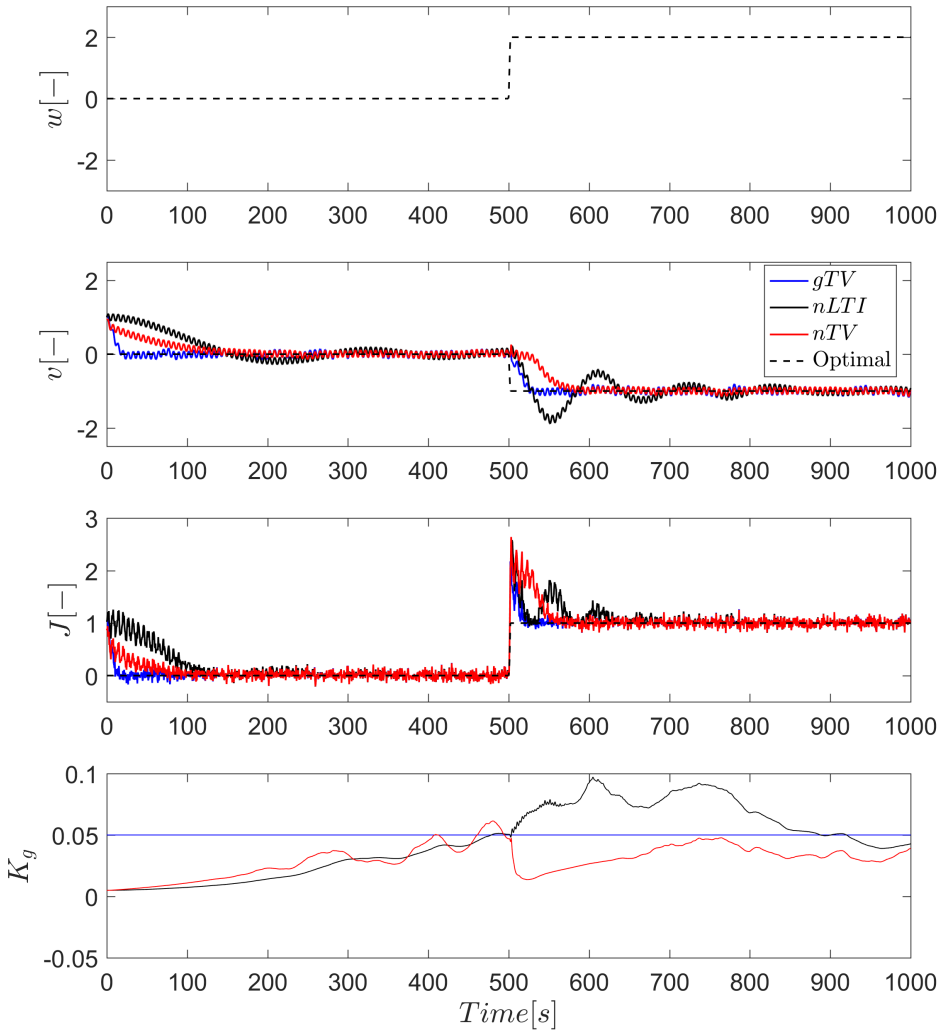

Figure 3.24: Comparison between the gTV, $nLTI$, and nTV responses to the system described by (3.49) with measurement noise acting on the performance index output. Because of the output noise, neither the $nLTI$ nor the nTV clearly converge to the gTV's K_g value.

3.1.6.4 Multi-Input Comparison Between gTV, $nLTI$, and nTV

The results from Section 3.1.6.2 indicated that adding additional inputs adversely affected ESC convergence rates because of the increase in unknown

parameters. While there are $1+n$ unknown parameters in gradient descent, where n is the number of inputs, there are an additional $\frac{n(n+1)}{2}$ unknown parameters in Newton descent. This fact partly explains why gTV can converge most quickly of the three algorithms in Figure 3.25. Comparing the Newton descent algorithms shows that relative to the nTV's performance, the $nLTI$ converges both more slowly and with more overshoot.

Table 3.13: Parameters for the gTV, $nLTI$, and nTV extremum seeking algorithms used in Figure 3.25, Figure 3.26, Figure 3.27, Figure 3.28, Figure 3.29, Figure 3.30.

gTV		$nLTI$		nTV	
Param	Value	Param	Value	Param	Value
a_1	$[0.01\ 0.006]^T$	a_1	$[0.1\ 0.1]^T$	a_1	$[0.01\ 0.006]^T$
$[\omega_1\ \omega_2]^T$	$[1\ 0.6]^T$	$[\omega_1\ \omega_2]^T$	$[1\ 0.6]^T$	$[\omega_1\ \omega_2]^T$	$[1\ 0.6]^T$
λ	0.95	ω_{HPF}	0.01	λ	0.999
N	4	ω_{LPF}	0.01	N	10
T	0.1	ω_{RIC}	0.01	T	0.1
A_u	$I^{2\times 2}$	η_0	1	A_u	$I^{2\times 2}$
b_u	$[0\ 0]^T$	$\hat{v}_0$	$[1\ -1]^T$	b_u	$[0\ 0]^T$
K_g	$0.05I^{2\times 2}$	W_c	$\frac{1}{20}$	W_d	$\frac{1}{20}$
v_0	$[1\ -1]^T$	$\hat{\Gamma}_0$	0.05	v_0	$[1\ -1]^T$
x_0	$[0\ 0\ 0]^T$	δ	0	x_0	$[0\ 0\ 0\ 20\ 0\ 20]^T$
P_0	$I^{3\times 3}$			P_0	$I^{6\times 6}$
				Basis	Tchebyshev
				α	0.999
				δ	0
				$\hat{\Gamma}_0$	$0.05I^{2\times 2}$

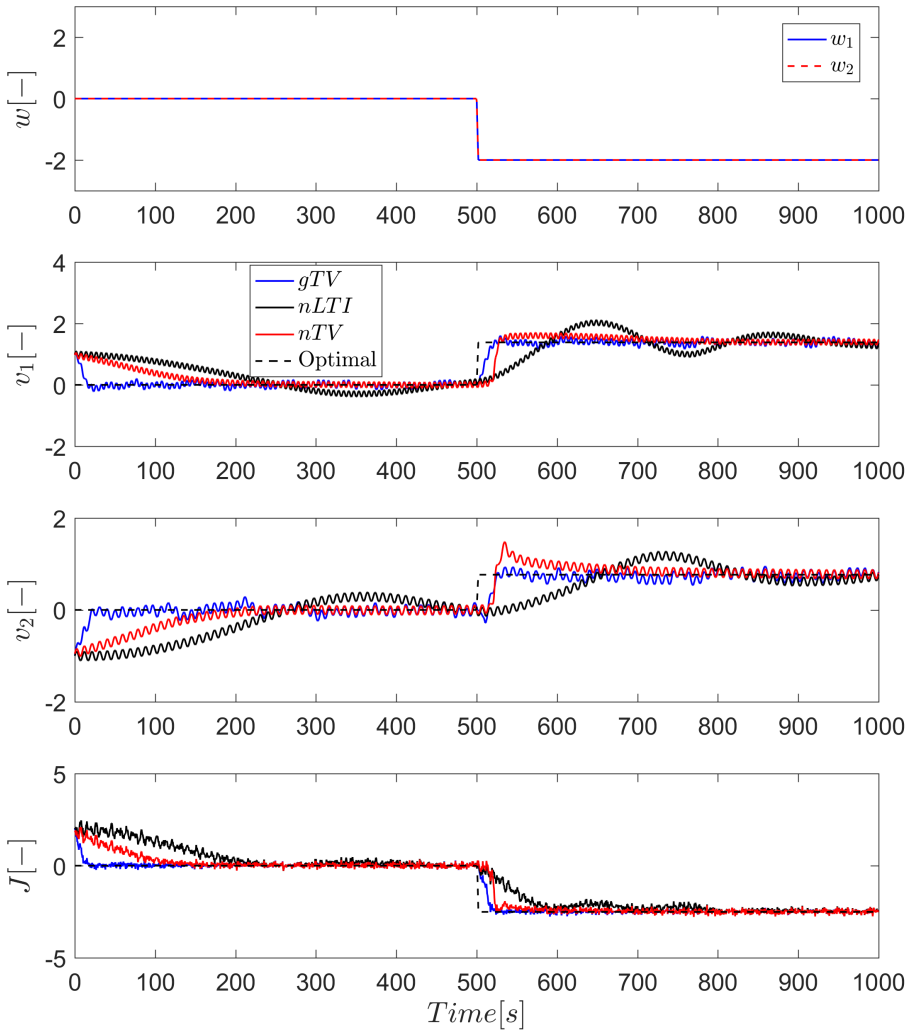

Figure 3.25: Comparison between the gTV, $nLTI$, and nTV for the case of the constant Hessian system in (3.50). Similar to the results in Section 3.1.6.3, the gradient algorithm's prior knowledge of the Hessian helps it surpass the Newton descent algorithms' convergence rates. Meanwhile, the nTV's time varying filter results in a performance improvement over the $nLTI$ case.

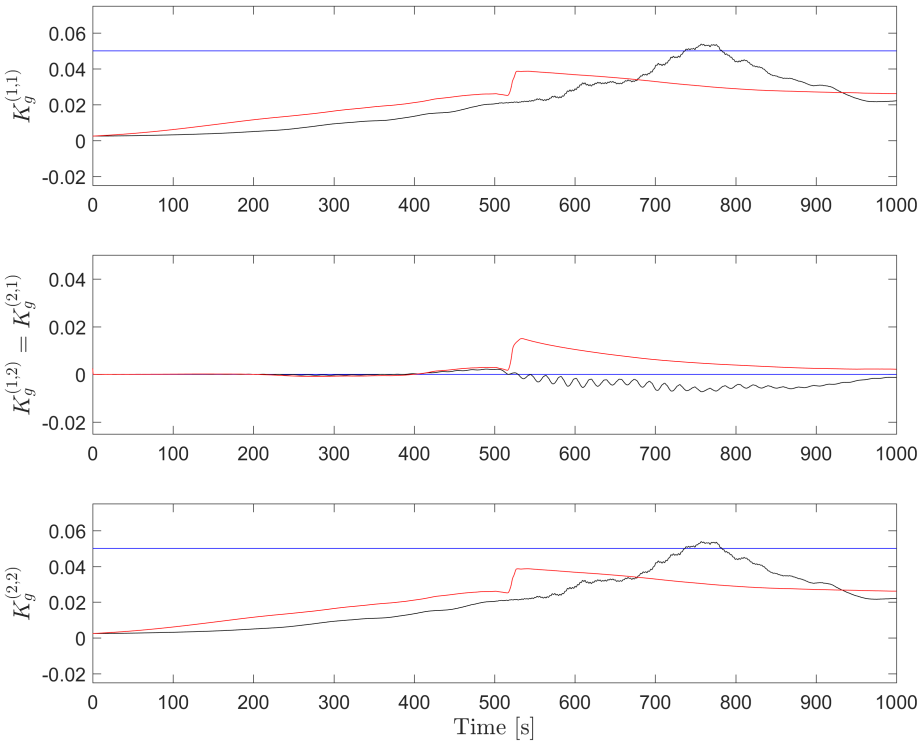

Figure 3.26: Comparison between optimization gains of the gTV, $nLTI$, and nTV for the case of the constant Hessian performance index in (3.50). While in Figure 3.24 the desired gain of the Newton algorithms were set equal to the gain of the gTV algorithm, the desired gains were lower here to prevent overshoot.

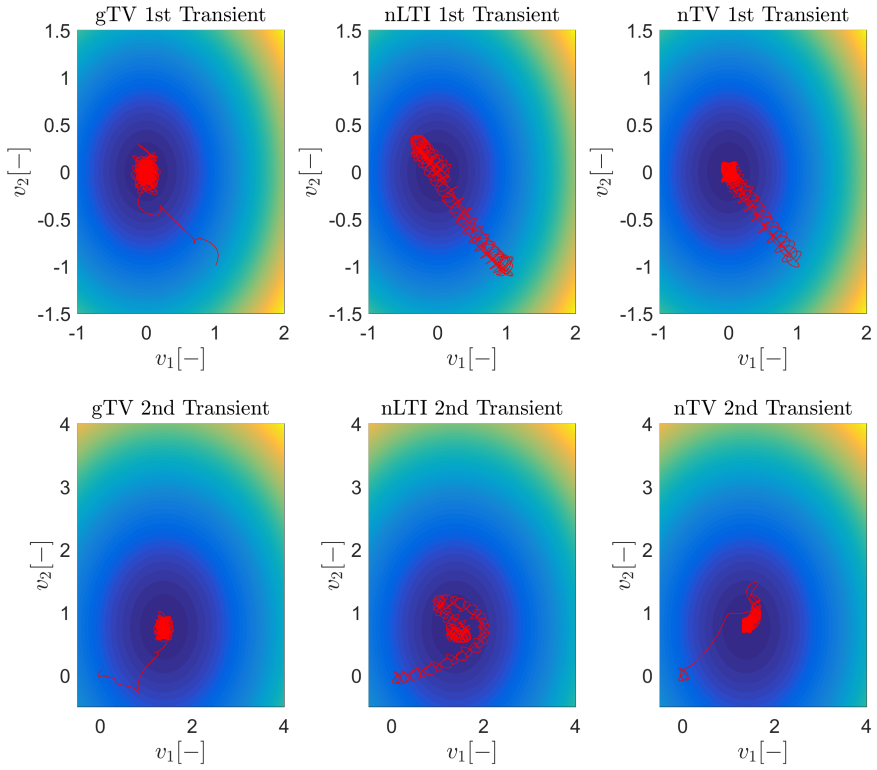

Figure 3.27: Comparison of input trajectories for the gTV, $nLTI$, and nTV. The $nLTI$'s trajectories tend to be curved, indicating overshoot. By contrast, the gTV and nTV converge almost directly to their respective optimal inputs.

While knowing the Hessian helps calibrate the gTV's convergence time, Figure 3.21 shows that when the Hessian is changed by disturbances, the gTV's convergence rate may be adversely affected. The next simulation results, applied to the system in (3.51), evaluates Newton descent as an approach for recalibrating the optimization gain after the Hessian is altered.

While the first transient from $0 \leq t \leq 500$ yields results that are identical to those in Figure 3.25, the disturbance change from $500 \leq t \leq 1000$ changes the Hessian, slowing gTV's convergence rate. In principle, convergence rates of the nTV and $nLTI$ algorithms should surpass the gTV's convergence rate if the Hessian change is significant enough. Here, only the nTV is able to adjust its K_g matrix to keep up with the gTV approach, while the $nLTI$ is outperformed.

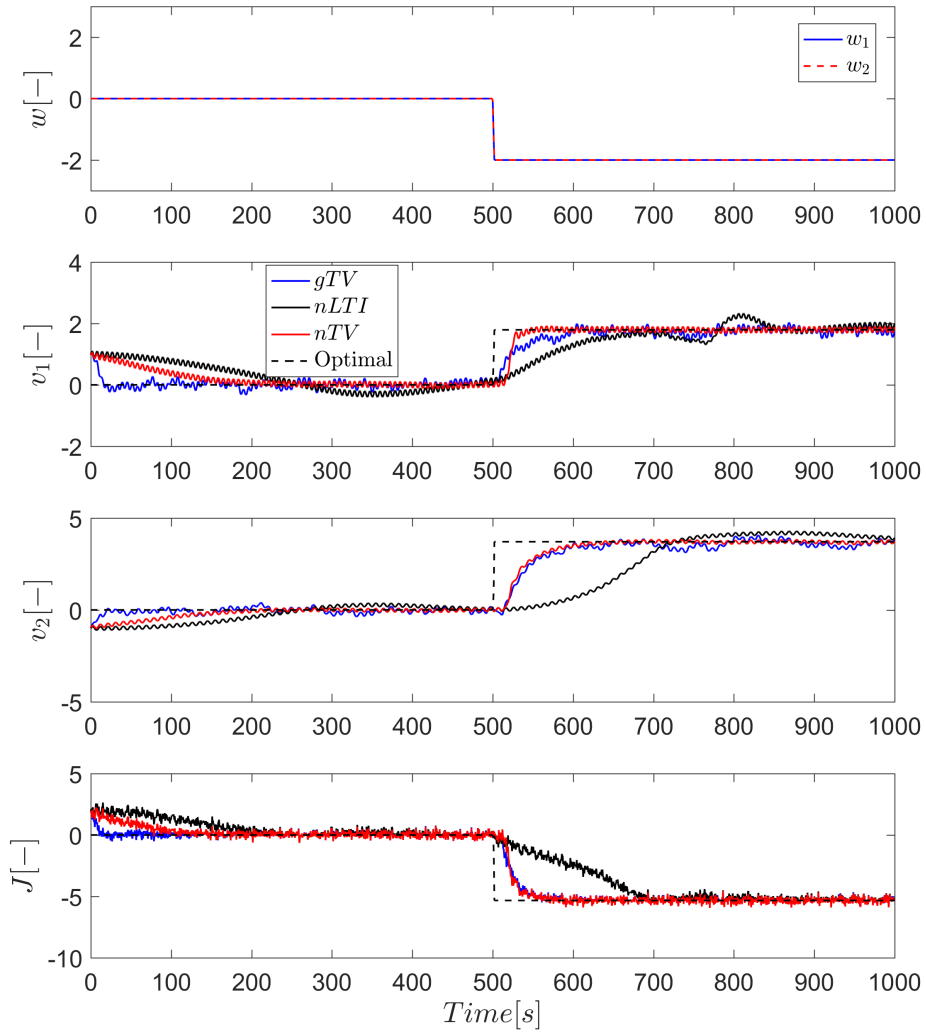

Figure 3.28: Comparison of dual input gTV, $nLTI$, and nTV algorithms for the case where the Hessian changes with disturbances. The nTV's ability to adjust its optimization gains helps it converge almost as quickly as the gTV, while the $nLTI$ is outperformed by both time varying extremum seeking approaches.

Figure 3.29 shows that the nTV adapts to the changing Hessian soon after the disturbances occur. Meanwhile, the $nLTI$'s convergence rate lags behind the algorithms equipped with time varying parameter estimators.

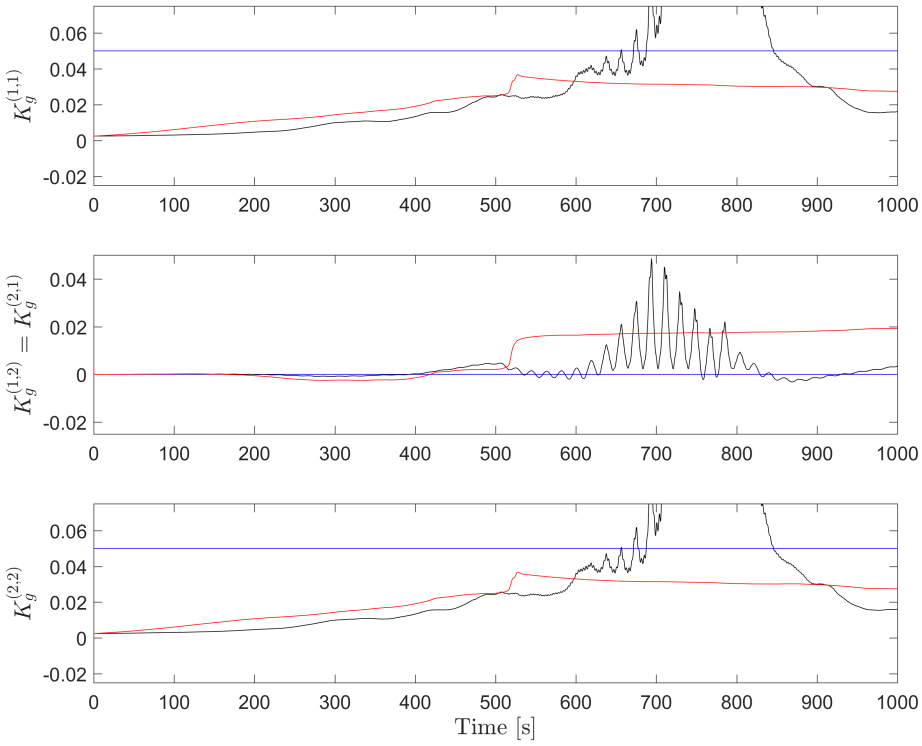

Figure 3.29: Comparison between optimization gains of the gTV, $nLTI$, and nTV for the case of the changing Hessian static map in (3.50). The nTV is able to discover the presence of off-diagonal terms in the Hessian and adjust its gain matrix accordingly. Meanwhile, the $nLTI$ produces transient oscillations in $K_g^{(1,2)}$ and struggles to keep up with the nTV's K_g estimate.

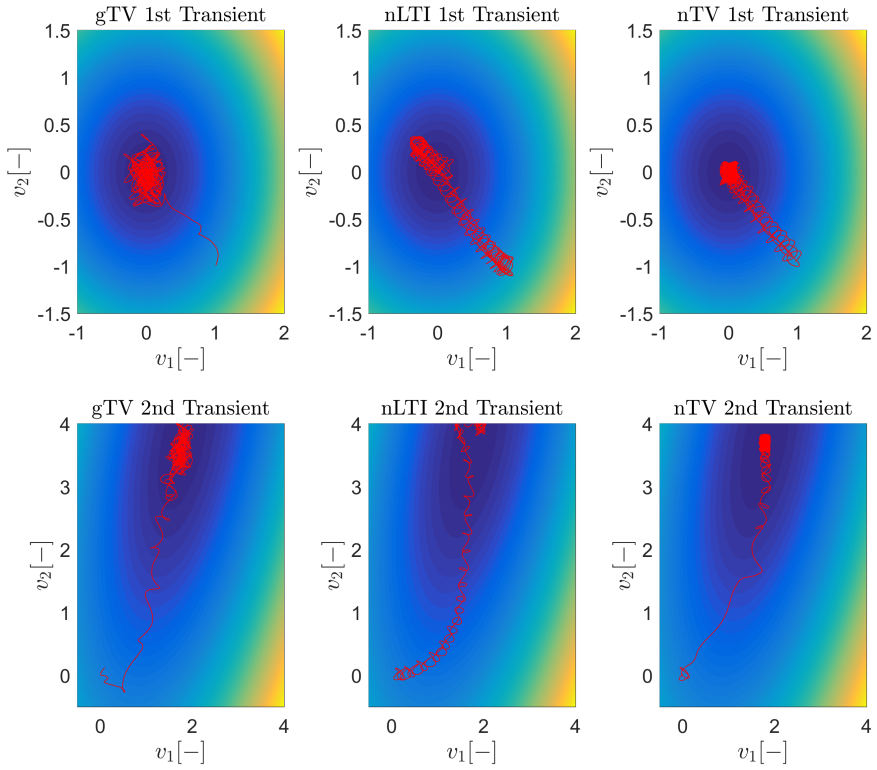

Figure 3.30: Comparison between the gTV, $nLTI$, and nTV input trajectories. The gTV and nTV approaches show similar convergence trends during the second transient. By contrast, the $nLTI$'s convergence lags behind and exhibits overshoot.

The results of this section offered further evidence that ESC using time varying parameter estimators is generally favorable to the classical demodulation and LTI filter based approach. It is recommended that when there is reasonable knowledge of system's Hessian available, the gTV approach is favorable. If Newton descent is required, using the nTV approach can lead to improvements in performance relative to the $nLTI$ case.

3.2 Self-Optimizing Control

When unknown disturbances change the optimal input, the ESC experiments with the optimizing input and adapts it using a descent law. Another RTO approach is to use a system model that predicts the optimal input using the system's current disturbance condition.

However, disturbances that cannot be directly measured pose an obstacle to implementation of model based control. Self-optimizing control solves this problem by regulating a combination of a sufficient number of measurements to a constant value [20]. Instead of having to measure the disturbances directly, the measurements are strategically chosen using H in Figure 3.31 to detect these disturbances and force the input to correct for them indirectly through regulation to the constant set point. A major challenge to implementation of self-optimizing control is obtaining a sufficiently accurate system model that captures the effects of process disturbances on the system outputs.

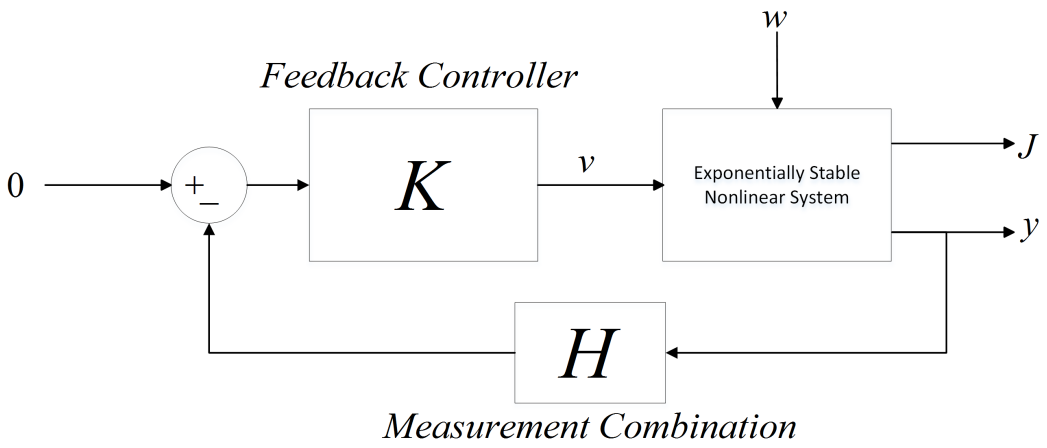

Figure 3.31: Self-optimizing control block diagram. A combination of measurements is regulated to zero in order to choose the optimal v.

Recently, however, a method introduced by [23] has shown that the self-optimizing measurement combination can be found from historical process output data when it reflects optimal choices of v. Such data could be generated first by using ESC to control v and then analyzing the data to find H.

The goal of Section 3.2.1 is to develop sufficient background in self-optimizing control for understanding the results in Chapter 5. Section 3.2.1.1 explains the relationship between disturbances and the optimal input for a system described by a quadratic performance index. Using the results from Section 3.2.1.1, Section 3.2.1.2 explains the theory behind "null space method" self-optimizing control, which is capable of achieving optimal operation provided that certain assumptions are met [21, 26]. Section 3.2.1.3 leverages theory from Section 3.2.1.2 to describe the process for implement-

ing self-optimizing control using optimal input-output data collected from a system.

While self-optimizing control has often been considered independent from extremum seeking control, [28] demonstrated that the two approaches were complementary. Section 3.2.1.4 tests their combination using the single input plant in (3.49). The results highlight benefits and drawbacks of combining the two real-time optimization approaches against implementing them separately.

3.2.1 Self-Optimizing Control Background

3.2.1.1 Mappings From Disturbances to Optimal Inputs

The systems (3.49) and (3.50) represent instances of quadratic functions in the controllable input, v, and uncontrollable disturbance, w. Suppose that given an arbitrary performance index in (3.52),

$$J = J(v, w), \tag{3.52}$$

a nominal disturbance $\overline{w}$, and a corresponding nominally optimal input, $\overline{v}^*$, the local optimality conditions are satisfied: $\frac{\partial J}{\partial v}(\overline{v}^*, \overline{w}) = 0$ and $\frac{\partial^2 J}{\partial v^2}(\overline{v}^*, \overline{w}) > 0$. The performance index, J, can be approximated by a second order Taylor series about the nominal input condition given in (3.53).

$$J \simeq J_0 + \begin{bmatrix} J_v & J_w \end{bmatrix} \begin{bmatrix} \Delta v \\ \Delta w \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \Delta v \\ \Delta w \end{bmatrix}^T \begin{bmatrix} J_{vv} & J_{vw} \\ J_{wv} & J_{ww} \end{bmatrix} \begin{bmatrix} \Delta v \\ \Delta w \end{bmatrix} \tag{3.53}$$

The variables, $\Delta w = w - \overline{w}$, $\Delta v = v - \overline{v}^*$, and $\Delta y = y - \overline{y}^*$, are deviations from nominal conditions. Subscripts of J denote partial derivatives. For instance, $J_{vw} = \frac{\partial^2 J}{\partial v \partial w}$. Given that $J_v = 0$ at $(\overline{v}^*, \overline{w})$, (3.53) can be solved for the change in the optimal input, $\Delta v^* = v^* - \overline{v}^*$, as a function of Δw, giving the linear relationship in (3.54).

$$\Delta v^*(\Delta w) = -J_{vv}^{-1} J_{vw} \Delta w \tag{3.54}$$

For the case where reliable measurements of w are available, one possible optimization strategy is to directly map disturbance measurements to the system inputs using (3.54). However, when disturbances cannot be measured,

linearity of the local disturbance to optimal input map in (3.54) can be exploited to implement a control law that achieves the same result. Section 3.2.1.2 shows how this can be done using system measurements.

3.2.1.2 Self-Optimizing Control Using the Nullspace Method

The null space method from [21] performs optimization compensation for disturbances that are not measurable. It works best for systems where a quadratic approximation like the one in (3.53) is fairly accurate. Suppose that in addition to the output J in (3.52), there are other outputs, y, unrelated to the performance index, but possibly used for fault detection. Suppose that these outputs are given by the static map in (3.55).

$$\begin{aligned} &Nonlinear\ Output\ Equation: \\ &y = h(x) \\ &After\ making\ a\ steady\ state\ approximation: \\ &x^* = l(v,w) \\ &y = h(l(v,w)) = g(v,w) \end{aligned} \tag{3.55}$$

The map of outputs, $y = g(v,w)$, related to the reduced system in (3.52) can be linearized about $\overline{v}^*$ and $\overline{w}$ to find the linear and steady state approximations of the input-output gain in (3.56).

$$\begin{aligned} \Delta y &= G_v \Delta v + G_w \Delta w \\ G_v &\equiv \frac{\partial g}{\partial v}(\overline{v}^*, \overline{w}) \\ G_w &\equiv \frac{\partial g}{\partial w}(\overline{v}^*, \overline{w}) \end{aligned} \tag{3.56}$$

By substituting (3.54) into (3.56), the change in optimal value of the process outputs, $\Delta y^* = y^* - \overline{y}^*$, becomes a linear function of the disturbance, Δw.

$$\Delta y^* = (G_w - G_v J_{vv}^{-1} J_{vw}) \Delta w \tag{3.57}$$

Defining $F = (G_w - G_v J_{vv}^{-1} J_{vw})$, (3.57) can be set to zero by combining the available outputs, Δy, using a matrix, H, that is in the left null space such

that (3.58) holds.

$$HF = 0 \tag{3.58}$$

Eq. (3.58) indicates that there must be an H such that $F \in ker(H)$ in order to guarantee zero loss. This will always be the case provided that $n_y \geq n_v + n_w$. Additional conditions for achieving zero loss apply: there is no measurement noise, the linear and convex approximation of the system is equivalent to its actual behavior, and perfect tracking of $H\Delta y = 0$ is achieved. While these assumptions may not be fully met, achieving close to zero loss is an attractive prospect. Section 3.2.1.3 will exploit this property to show that a self-optimizing H can be extracted from optimal output data.

3.2.1.3 Combining Self-Optimizing Control with Extremum Seeking Control

The previous section showed that steady state linear systems with quadratic performance indices and no measurement noise could be operated with zero loss if H was chosen properly. Suppose now that instead of knowing a system's linear gains, G_v and G_w, and Hessian components, J_{vv} and J_{vw}, a wide matrix of sampled output data from an optimal operation policy is known. This data is contained in a matrix, Y^*, which represents the outputs corresponding to achieving the optimal steady state input, v^*, for all data points. When centered about its mean, Y^* becomes $\Delta Y^* = F\Delta W$, where F is defined in the previous section and the columns of ΔW are the disturbance vectors at each sample time. Optimal operation data Suppose further that $n_y \geq n_v + n_w$. Eq. (3.58) shows that for the self-optimizing H, $HF\Delta W = H\Delta Y^* = 0$, indicating that H can be found from centered optimal data [23].

Using the following procedure from [23], the approximation of H, H_{SVD} can be extracted from nearly optimal plant data, Y.

1. Use an extremum seeking controller algorithm to generate optimal plant data.

2. Determine the most significant unmeasurable system disturbances.

3. Once disturbances are believed to have sufficiently excited the system, gather output data into a wide matrix, $Y \in R^{n_y \times n_{data}}$.

4. Find the mean value from each column of Y and center Y about this mean value.

5. If measurements are dissimilar in magnitude, scale the data to achieve similar magnitude changes. The matrix of scaled, centered data is given by $Y_{scl,0}$.

6. H_{SVD} can be determined by performing a singular value decomposition on the centered and scaled data according to (3.59).

$$Y_{scl,0} = U\Sigma V^T \tag{3.59}$$

   In the single input case, H_{SVD}^T is the last column of U, which corresponds to the minimum singular value of $Y_{scl,0}$.

After using the procedure above to approximate H, two options are available. In both options, H combines y and a regulator is designed to maintain $H\Delta y$ at its mean value in the data. The first option involves no more steps. The second possibility is to modify the $H\Delta y$ set point from 0 to a value that is closer to optimal using ESC. This strategy could be useful to reject optimal input changes caused my unmodeled disturbances. It could also be useful if Y contains suboptimal output data, in which case H is not the true self-optimizing measurement combination.

Using combined extremum seeking and self-optimizing control may require a redesign of the extremum seeking controller, including changing the optimization gain and the perturbation amplitude. If the closed loop system time constant changes significantly as a result of changing the ESC's input from v to the $H\Delta y$ set point, then the perturbation frequencies should be adjusted.

A natural question about implementing self-optimizing control using optimal operation data and its combination with extremum seeking is how well it performs in comparison to a baseline strategy with perfect model knowledge and the true optimal measurement combination, H_{true}. The following section addresses this question using simulation examples.

3.2.1.4 Evaluation of Self-Optimizing Control, Extremum Seeking Control, and Their Combination

Self-optimizing control with perfect model knowledge represents an upper bound on RTO performance when disturbances are not measurable and the performance index in question is quadratic. This section investigates how close extremum seeking control, data based self-optimizing control, and their combination can get to this upper bound.

The properties of each real-time optimization approach are investigated using the system from (3.49) with one modification: v and w act on a linear system with a measurable output, y from (3.60).

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = G_v v + G_w w = \begin{bmatrix} -1 \\ 1 \end{bmatrix} v + \begin{bmatrix} 1 \\ 1 \end{bmatrix} w \tag{3.60}$$

For the system in (3.49), $J_{vv} = 2$ and $J_{vw} = 1$. Solving for the null space of $F = (G_w - G_v J_{vv}^{-1} J_{vw})$ and setting the Euclidean norm of H to 1 gives $H = [-0.32\ 0.95]$. This is the optimal H with full knowledge of the system model.

Given the appropriate H for implementation of the baseline controller, Figure 3.32 compares the results of three real-time optimization approaches: a *gLTI* implementing extremum seeking using v, denoted "*gLTI alone*", self-optimizing control with an incorrect initial set point, "*Incorrect SOC*", and the incorrect SOC combined with extremum seeking control to find the optimal set point, denoted "*SOC and ESC*". Starting at 0, w changes to 2 after the first 50 seconds of the simulation to induce a change in the optimal input. The self-optimizing regulator (represented by K in Figure 3.31) is an integral controller with a transfer function, $\frac{k_i/G_v H}{s}$, where k_i shown in Table 3.14.

The results show that *gLTI alone* and *SOC and ESC* converge at similar rates, but *SOC and ESC* deviates less from the baseline input and cost, v_{SOC} and J_{SOC}, respectively, after the disturbance change. While the *SOC and ESC* input changes after the disturbance, it does so because the disturbance change affects the gradient estimate, not because there is a change in the optimal input. By comparison, the *gLTI alone* must rediscover the optimal input after the disturbance, a process which results in a longer convergence time. Without extremum seeking control, the *Incorrect SOC* gets neither the

initial optimal input correct, nor the optimal input following the disturbance, showing that error in choosing the initial set point incurs a performance index loss, $J - J_{SOC}$.

Table 3.14: Parameters for the *gLTI alone*, *Incorrect SOC*, and *SOC and gLTI* used in Figure 3.32.

gLTI alone		*Incorrect SOC*		*SOC and gLTI*	
Parameter	Value	Parameter	Value	Parameter	Value
a_1	0.1	a_1	–	a_1	0.08
ω_1	1	ω_1	–	ω_1	1
ω_{HPF}	0.2	ω_{HPF}	–	ω_{HPF}	0.2
ω_{LPF}	0.2	ω_{LPF}	–	ω_{LPF}	0.2
K_g	0.05	K_g	–	K_g	0.08
η_0	1	η_0	-	η_0	1
$\hat{v}_0$	1	$\hat{r}_0$	1.27	$\hat{r}_0$	1.27
H	–	H	$[-0.32\ 0.95]$	H	$[-0.32\ 0.95]$
k_i	–	k_i	10	k_i	10

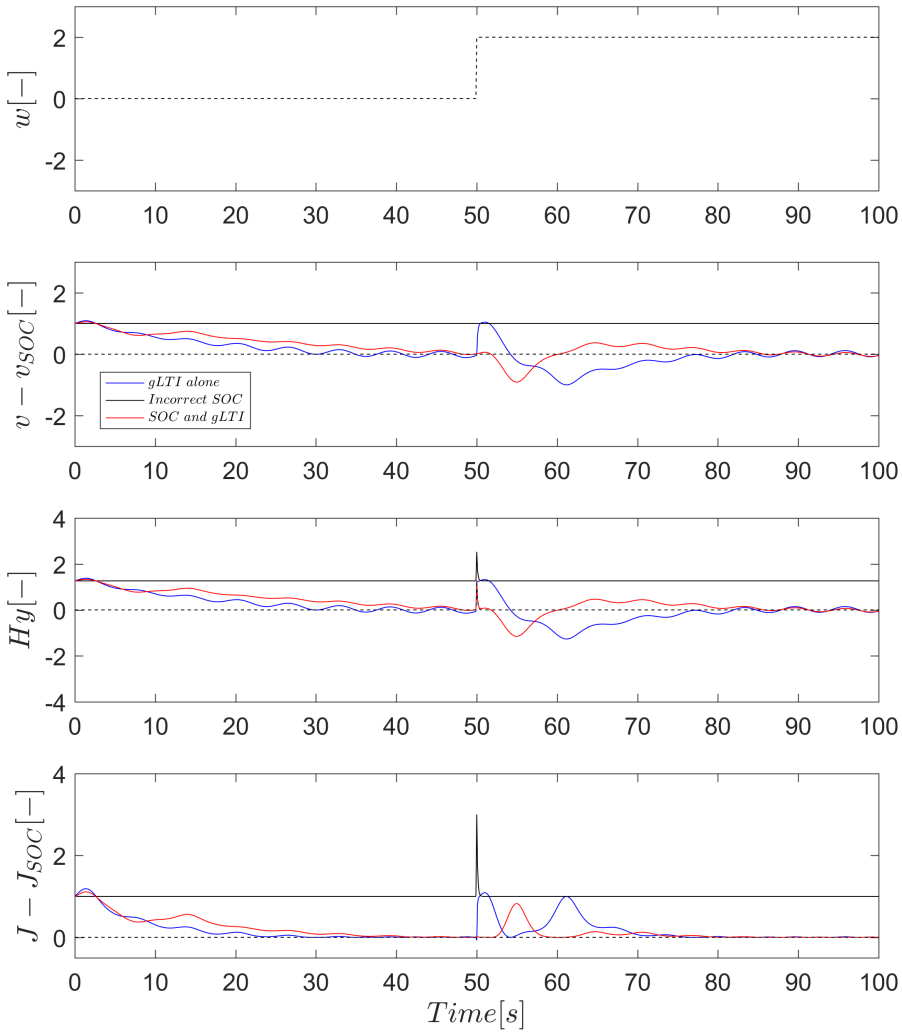

Figure 3.32: Comparison between self-optimizing control, extremum seeking, combined extremum seeking and self-optimizing control, and self-optimizing control with model error.

The next test studies how close self-optimizing control using optimal process data can get to achieving zero loss. Before the test can be done, process data must be generated and analyzed to synthesize an approximation of the optimal measurement combination, denoted H_{SVD}. Figure 3.33 shows results from three steps in w that generate variations in the outputs, y_1 and y_2. A notable feature of the simulation is that the minimizing input is not always achieved. However, Figure 3.34 shows that the values of H_{SVD} converge close

to their true values in H_{true} following the final disturbance change back to $w = 0$.

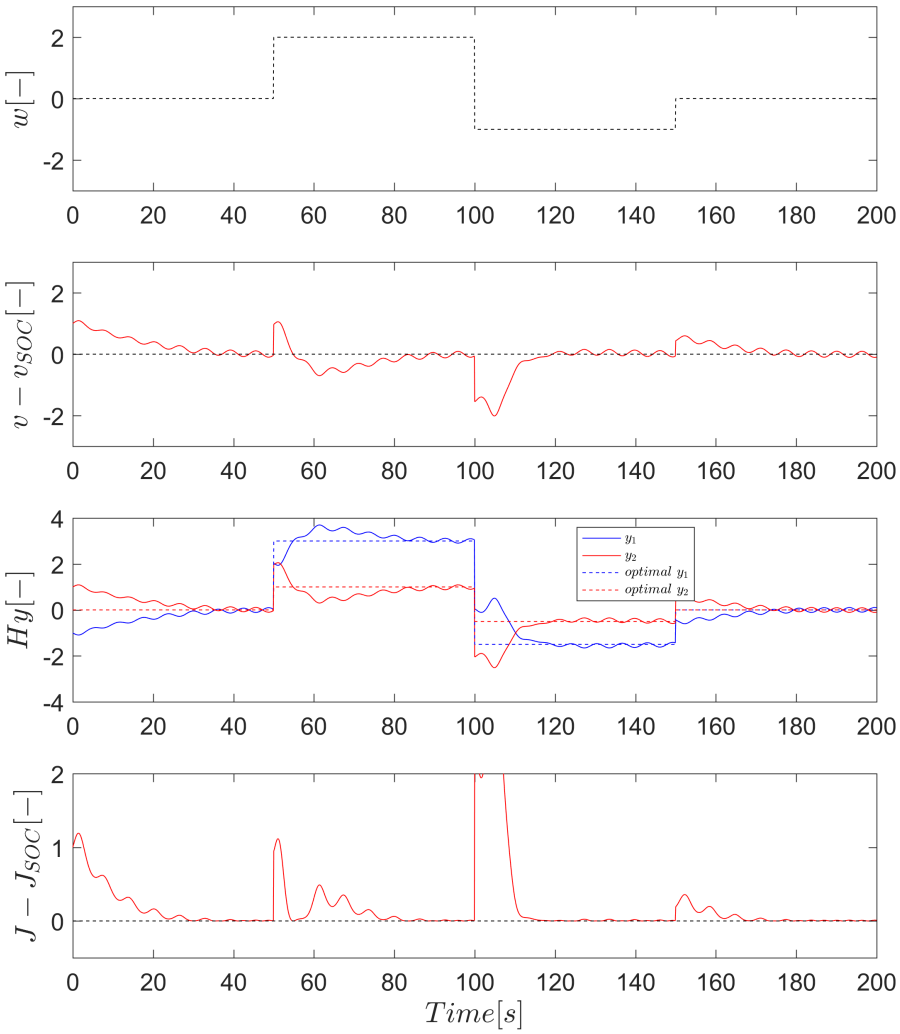

Figure 3.33: Training data used to estimate the best measurement combination.

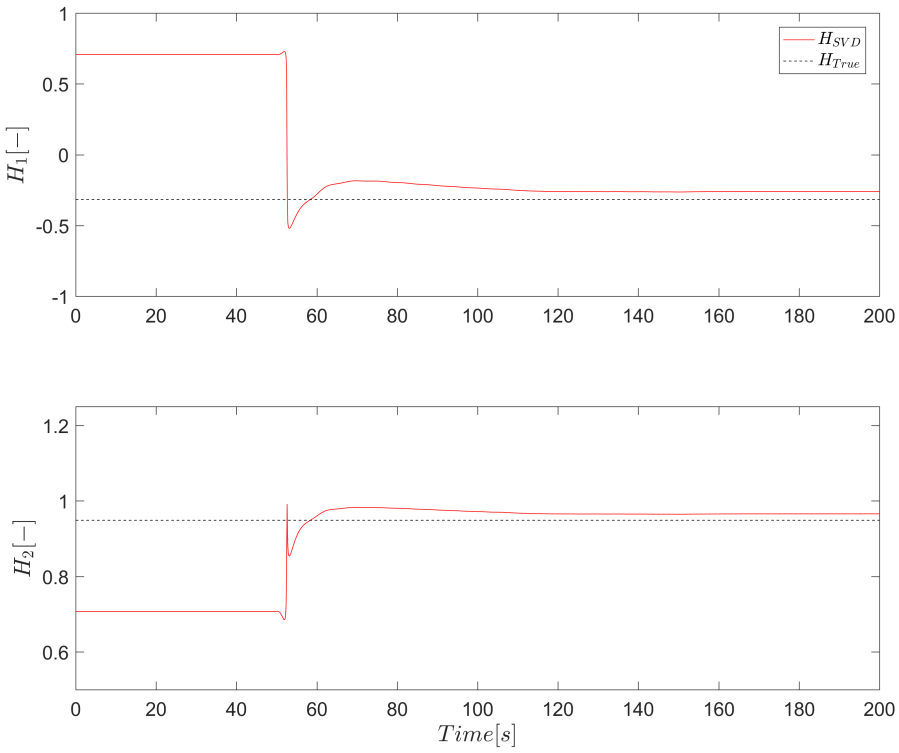

Figure 3.34: Evolution of the H_{est} components based on output data from Figure 3.33.

Figure 3.35 shows a comparison between the following strategies:

- Self-optimizing control implemented using the measurement combination from data, denoted by SOC_{est}.
- ESC combined with the true optimal measurement combination, denoted SOC_{true} *and* ESC.
- ESC combined with the measurement combination from data, denoted SOC_{est} *and* ESC.

The results show that the approximate measurement combination's performance is almost indistinguishable from the baseline performance of SOC using the true measurement combination. Likewise, using the approximate implementation of SOC combined with ESC has performance similar to the combined true SOC and ESC. This similarity indicates that the performance sacrifice from using data for SOC implementation may be small, even when suboptimal process data is included in the analysis.

Table 3.15: Parameters for the data based SOC, true SOC and ESC, and data based SOC and ESC used in Figure 3.35.

SOC_{est}		SOC_{true} and ESC		SOC_{est} and ESC	
Parameter	Value	Parameter	Value	Parameter	Value
a_1	–	a_1	0.08	a_1	0.08
ω_1	–	ω_1	1	ω_1	1
ω_{HPF}	–	ω_{HPF}	0.2	ω_{HPF}	0.2
ω_{LPF}	–	ω_{LPF}	0.2	ω_{LPF}	0.2
K_g	–	K_g	0.075	K_g	0.08
η_0	-	η_0	1	η_0	1
$\hat{r}_0$	0	$\hat{r}_0$	1.23	$\hat{r}_0$	1.27
H	$[-0.26\ 0.97]$	H	$[-0.26\ 0.97]$	H	$[-0.32\ 0.95]$
k_i	10	k_i	10	k_i	10

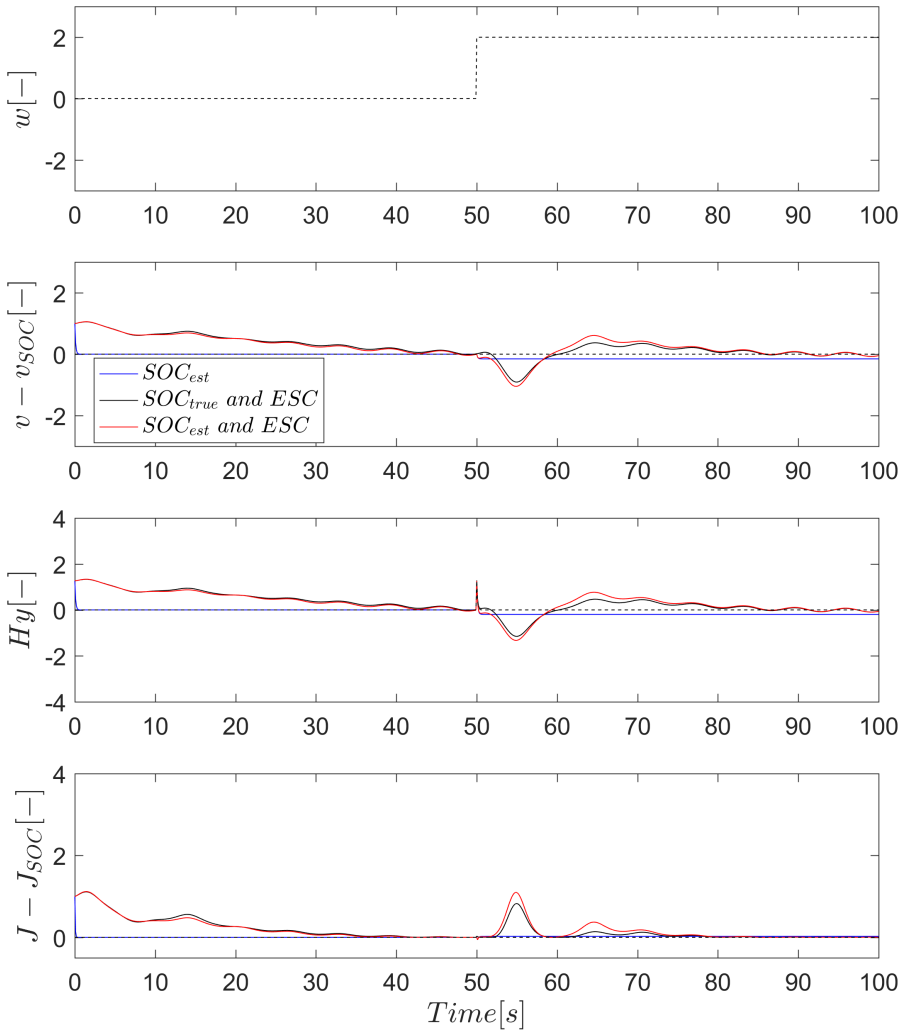

Figure 3.35: Comparison between data based SOC, true SOC and ESC, and data based SOC and ESC.

3.3 Conclusions

This chapter discussed the fundamentals of extremum seeking, a model free RTO approach, and self-optimizing control, which has both model free and model based forms. A finite difference approach was used to introduce the concept of and challenges to derivative estimation for extremum seeking. Once intuition for the finite difference approach was established, sophisti-

cated estimation approaches were introduced: a scheme using LTI filters with demodulation and a scheme using a time varying recursive least squares algorithm. These estimation approaches were paired with gradient and Newton descent ESC algorithms.

Simulation results from 3.1.6 indicated that implementing ESC using time varying parameter estimators was favorable to LTI filters with demodulation. When sufficient information is known about the curvature of a quadratic performance index, gradient descent should be used for optimization because it tends to converge faster than Newton descent alternatives, where the curvature must be estimated. However, if there is significant uncertainty about the curvature, then Newton descent is a viable solution.

In 3.2.1, self-optimizing control was introduced as both an alternative and complementary addition to ESC. Nullspace method self-optimizing control from [21] was shown to achieve zero loss under the assumption of perfect model knowledge, a linear system, a quadratic performance index, and zero measurement noise. Using the results from the null space method analysis, the model free, data based self-optimizing control implementation from [23] was introduced as both an optimization strategy on its own and in combination with extremum seeking.

Simulation results in Section 3.2.1.4 showed that combining self-optimizing control and extremum seeking reduced the sensitivity of the optimization problem by creating an ESC input with an optimal value that was invariant to disturbances. Extremum seeking was also shown to be a potentially effective method for generating optimal process data used to choose an approximately self-optimizing measurement combination.

This chapter helped develop intuition for real-time optimization approaches used in the chapters that follow. Because the systems considered in this chapter are easy to analyze, they are useful for gaining experience with extremum seeking and self-optimizing control before implementation on a vapor compression system. Like the simple examples seen earlier, implementation of RTO approaches on a vapor compression system is fundamentally challenged by measurement noise, disturbances, and dynamics.

CHAPTER 4

EXTREMUM SEEKING ALGORITHMS APPLIED TO A VAPOR COMPRESSION SYSTEM

This chapter builds on the previous chapter's results by evaluating the performance of extremum seeking algorithms applied to vapor compression systems. Two claims have been made for quadratic performance indices: using a time-varying filter for estimating derivatives improved on the performance attained using LTI filters and demodulation; Newton descent should be used only when insufficient information is available to tune the convergence rate for gradient descent. These considerations lead to the following questions:

1. What are the challenges involved in moving from example systems with no dynamics to vapor compression systems, which have complicated nonlinear dynamics?

2. Do time varying filters improve derivative estimation rates and ESC convergence speed when applied to vapor compression systems?

3. Is there significant enough uncertainty in the performance indices to warrant using Newton descent over gradient descent?

4. What are the pros and cons of using multiple optimization inputs?

Partial answers can be found by considering recent developments in the literature. In [17], the authors use the *nLTI* algorithm to choose optimal vapor compression fan speeds, but did not clearly show the algorithm's ability to adjust K_g. In [45], the authors use a form of the *gTV* algorithm to choose an optimal set point for the compressor's exit refrigerant temperature. In [11], the authors used a Thermosys model to show that a form of the *gTV* algorithm outperformed its *gLTI* counterpart. In [12], the *nLTI* algorithm was shown to outperform the *gLTI* algorithm for optimization of a chiller system's cooling tower fan speed and condenser water pump speed inputs. To summarize the unexplored work in the area of ESC applied to VCSs:

Newton descent has not been combined with time varying estimation and used for RTO; Newton descent has not been explicitly shown to adapt to a target optimization gain; and a discussion about weighting the pros and cons of Newton ESC against those of gradient ESC is missing.

This chapter addresses the issues above using both the Thermosys model and the experimental test rig from Chapter 2 as testing grounds for the gTV, $nLTI$, and nTV extremum seeking algorithms. Both single input and dual input cases shown in Chapter 2 are considered, where the performance index is the total power consumption, $\dot{W}_{sys}$, and the optimizing inputs are the fan speeds, u_{cf} and u_{ef}. Before each plot of results, the corresponding controller parameters are given in a table.

4.1 Comparison of Algorithms in Simulation

In this section, the Thermosys simulation model from Chapter 2 is used as a testbed to evaluate the three candidate extremum seeking algorithms. The extremum seeking problem is varied in two ways:

1. **Single input and dual input**: Comparing single input and dual input results helps illustrate challenges that arise from increasing the number of optimization variables.

2. **No noise versus realistic noise meant to mimic experimental conditions**: For the noise free case, the major challenge facing extremum seeking arises from the system dynamics. This challenge can largely be addressed by choosing the perturbation signal wisely. When realistic levels of measurement noise are introduced, convergence is slowed because filter parameters must be adjusted to place less emphasis on individual measurements.

Unlike the simulation results in Chapter 3, this section ignores the effects of disturbances. Instead, the algorithms are tested only by converging to the optimal input(s) from a suboptimal starting point.

4.1.1 Single Input

In Chapter 2, the single input VCS performance index evaluation showed that a range of condenser fan speeds changed the system's total power consumption while $\dot{Q}_{evap}$ or T_{SH} were kept at their desired set points. Figure 4.1 partly illustrates the control strategy that makes this possible. While the value of u_{cf} may change, the u_{kp} and u_v inputs are adapted by the controller (LQR+I) to meet the $\dot{Q}_{evap,ref}$ and T_{SH} set points. The abbreviation, LQR+I, represents "linear quadratic regulator with integral action". While the u_{ef} input could change as well, it is kept constant until the next section, which is focused on the dual input scenario.

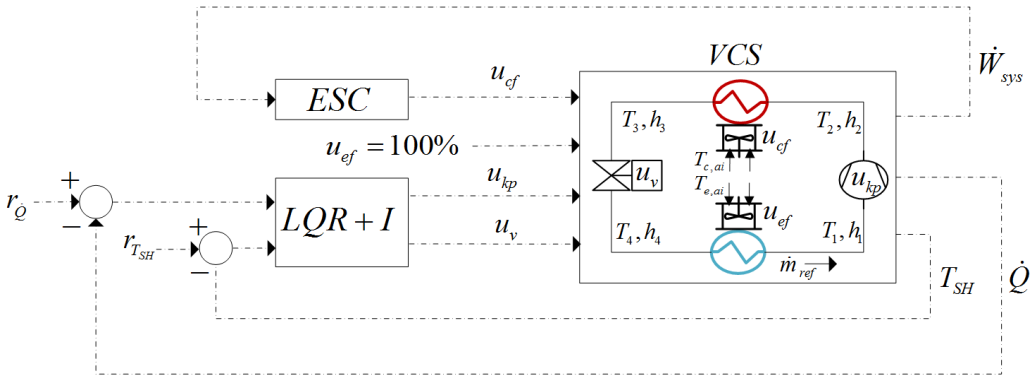

Figure 4.1: Controller architecture for the single input extremum seeking tests.

Before implementing extremum seeking, the combined closed loop LQR+I and VCS 2% settling time was determined to be 240s in Table B.2.1, corresponding to a maximum dither frequency of $0.02 rad/s$. To ensure that the ESC algorithms operated at steady state, a conservative dither frequency of $0.005 rad/s$ was chosen. A perturbation amplitude of 5% balances sufficient excitation with limited process disruption. To make sense of the other tuning parameters in the sections that follow, consult the tuning rules in Chapter 3.

4.1.1.1 No Noise Results

First, the simplest case of single input ESC and no measurement noise was tested. One of the first considerations in determining a successful ESC application to a VCS is to check if the $\dot{Q}_{evap,ref}$ and the $T_{SH,ref}$ set points were

satisfied. The top two plots in Figure 4.2 showing the $\dot{Q}_{evap}$ and T_{SH} trajectories confirm that the LQR+I is successful in maintaining the desired set points throughout the simulation.

The next consideration is to determine whether the performance index was minimized. As in Section 3.1.6, the gTV starts with a predetermined optimization gain shown in the bottom plot, while the $nLTI$ and nTV begin with overly conservative optimization gains. All three algorithms are shown to converge to the dashed lines indicating the minimum $\dot{W}_{sys}$ in the plot on the left and the minimizing u_{cf} in the plot on the right.

Table 4.1: Parameters for the gTV, $nLTI$, and nTV extremum seeking algorithms used in Figure 4.2.

gTV		$nLTI$		nTV	
Param	Value	Param	Value	Param	Value
a_1	0.0036	a_1	5	a_1	0.0036
ω_1	0.005	ω_1	0.005	ω_1	0.005
λ	0.993	ω_{HPF}	0.001	λ	0.997
N	10	ω_{LPF}	0.001	N	15
T	4	ω_{RIC}	0.001	T	4
A_u	27.5	η_0	1	A_u	27.5
b_u	-2.5	$\hat{v}_0$	$[1\ -1]^T$	b_u	-2.5
K_g	0.003	W_c	$\frac{1}{3500}$	W_d	$\frac{1}{3500}$
v_0	0.091	$\hat{\Gamma}_0$	2	v_0	0.091
x_0	$[0.4\ 0]^T$	δ	0	x_0	$[3.52\ -3.44\ 37.8]^T$
P_0	$100I^{2\times2}$			P_0	$100I^{3\times3}$
				Basis	Tchebyshev
				α	0.995
				δ	0
				$\hat{\Gamma}_0$	0.0264

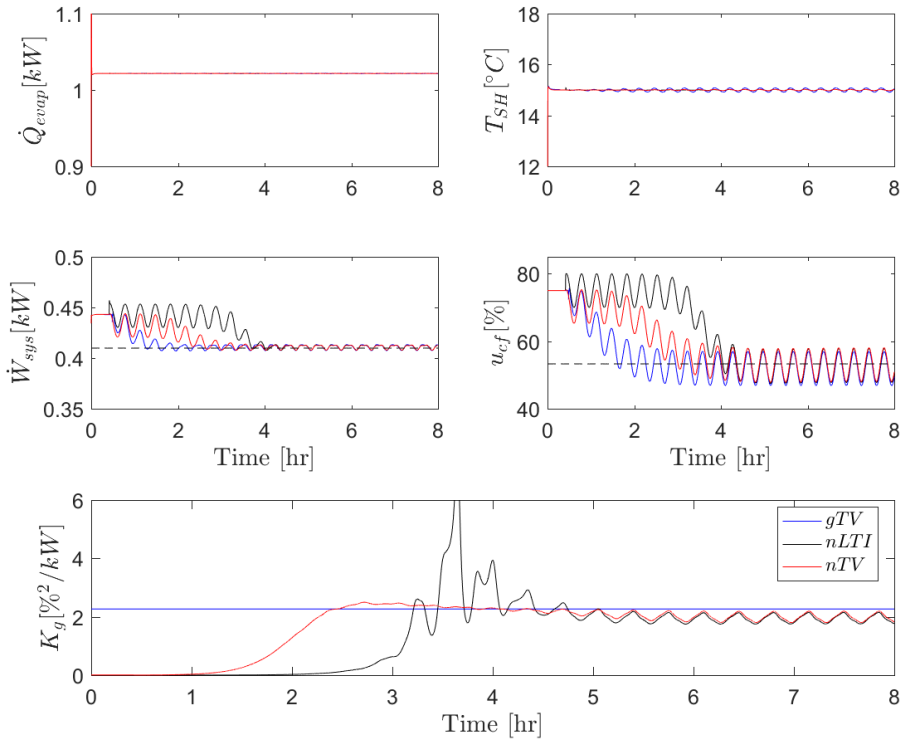

Figure 4.2: Noise-free simulation results for a single input comparison between the gTV, $nLTI$, and nTV extremum seeking algorithms. The results show that while the gradient descent's prior knowledge of the Hessian allows it to converge most quickly of the three approaches, the $nLTI$ and nTV algorithms converge to reasonable Hessian estimates after 2.5 and 5 hours respectively.

The gTV's gradient descent gain was designed according to the local quadratic fit shown in Figure 2.13. Table 4.1 shows that the desired convergence time constant was 3500s. To confirm that the Newton descent algorithms can adjust the descent rate, the bottom plot shows that the $nLTI$ and nTV converge to the gTV's preprogrammed gain. Similar to the previous chapter's results, nTV's optimization gain converges noticeably faster and with less overshoot. Although the nTV converges roughly an hour later than the gTV, the delay from estimating the Hessian may be tolerable. However, the next section shows that adding noise to the performance index output increases the Newton descent's convergence time.

4.1.1.2 Realistic Noise Scenario Results

The simulations in Section 3.1.6 showed that noise degraded extremum seeking performance. This section shows that the same holds true for ESC applied to VCSs. Comparing Table 4.1 and Table 4.2 shows that the perturbation signals remain the same, but the Newton descent algorithms' filter parameters need to be made more conservative when noise is added. Meanwhile, the gTV's parameters were not affected. Figure 4.3 confirms that measurement noise decreases the convergence rate for the $nLTI$ and nTV simulations, while the gTV's convergence rate is not significantly altered. Noise leads to an increase in the dynamic Hessian inverse filter parameters for both the $nLTI$ and nTV, slowing convergence to the desired optimization gain, K_g, in the bottom plot. Even though these parameters have been chosen more conservatively, there are still fluctuations of the optimization gains following convergence of the input. These fluctuations indicate sensitivity to noise.

Table 4.2: Parameters for the gTV, $nLTI$, and nTV extremum seeking algorithms used in Figure 4.3. With the exception of the gTV's K_g value, all of the same parameters are used in Figure 4.11.

gTV		$nLTI$		nTV	
Param	Value	Param	Value	Param	Value
a_1	0.0036	a_1	5	a_1	0.003
ω_1	0.005	ω_1	0.005	ω_1	0.005
λ	0.993	ω_{HPF}	0.0006	λ	0.997
N	10	ω_{LPF}	0.0006	N	15
T	4	ω_{RIC}	0.0006	T	4
A_u	27.5	η_0	1	A_u	27.5
b_u	-2.5	$\hat{v}_0$	$[1 \ -1]^T$	b_u	-2.5
K_g	0.003	W_c	$\frac{1}{3500}$	W_d	$\frac{1}{3500}$
v_0	0.091	$\hat{\Gamma}_0$	2	v_0	0.091
x_0	$[0.4 \ 0]^T$	δ	0	x_0	$[3.52 \ -3.44 \ 37.8]^T$
P_0	$100I^{2\times 2}$			P_0	$100I^{3\times 3}$
				Basis	Tchebyshev
				α	0.998
				δ	0
				$\hat{\Gamma}_0$	0.0264

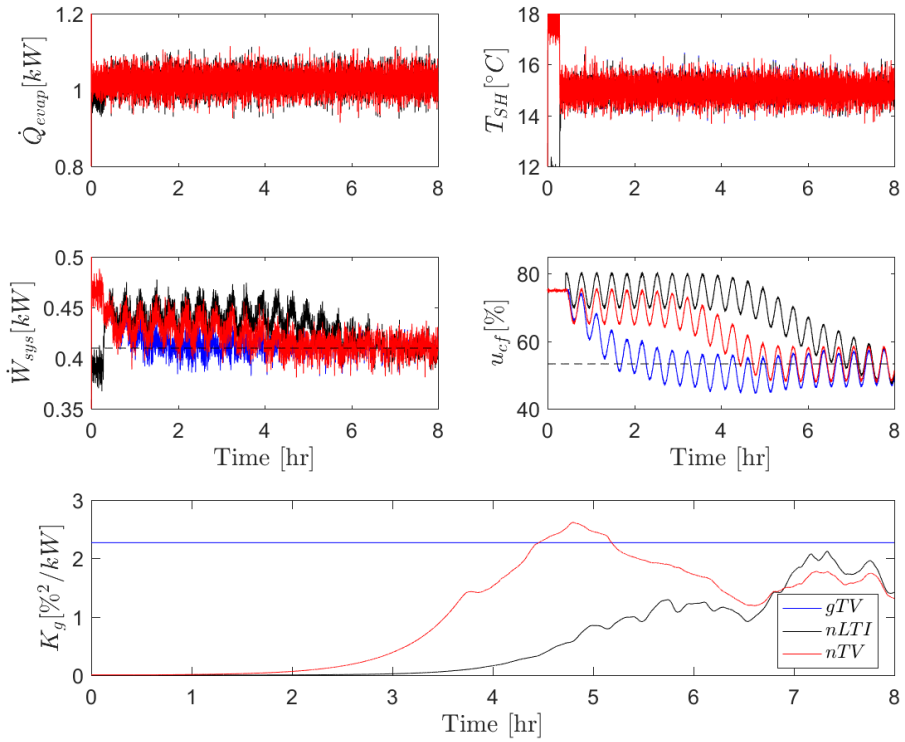

Figure 4.3: Realistic simulation results for a single input comparison between the gTV, $nLTI$, and nTV extremum seeking algorithms. The noise forces the $nLTI$ and nTV controllers to be more conservatively tuned than in the noise free case, resulting in slower convergence times. In addition, the gain estimate does not clearly settle by the end of the simulation.

4.1.2 Dual Input

Section 4.1.1 showed that nearly steady state behavior could be achieved by choosing the perturbation signals wisely, but that noise on the performance index output required tuning the derivative estimation filters conservatively. Chapter 3 suggested that adding inputs further decreases convergence speed because of an increase in unknown derivative parameters.

To test whether these trends appear in the dual input case, the control approach depicted by Figure 4.4 was implemented. Both the evaporator and condenser fans are now available for minimization of the performance index. The same (LQR+I) governs u_{kp} and u_v to achieve the nominal cooling capacity and superheat set points from Chapter 2. Similar to the single input case, there is a range of u_{cf} and u_{ef} inputs that allows the set points to be

met.

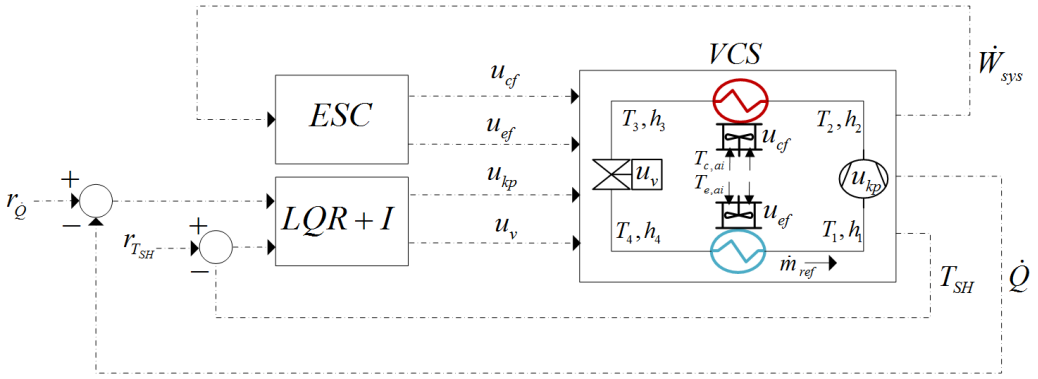

Figure 4.4: Controller architecture for the dual input extremum seeking tests.

Compared to the single input case, the maximum dither frequency was increased from $0.005 rad/s$ to $0.006 rad/s$ for the u_{cf} input. Using the dither frequency selection optimization problem from the $nLTI$ tuning rules resulted in a dither frequency of $0.004 rad/s$ for the u_{ef} input. These dither frequencies were used for all three ESC approaches. Although both inputs were initially perturbed at 5% of the speed command, the results that follow highlighted an advantage to changing this amplitude for the u_{ef} input.

4.1.2.1 No Noise Results

Similar to the single input case, Figure 4.5 shows that the $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$ set points are maintained at their desired values, despite the action of multiple fan speeds. Convergence of the Newton algorithms' inputs shows that adding an additional input does increase convergence time. By contrast, the gradient descent algorithm's convergence is not significantly affected. This is because the number of unknown parameters in the Newton descent case have doubled from 3 to 6, while the number of gradient descent parameters have increased from 2 to 3.

Table 4.3: Parameters for the gTV, $nLTI$, and nTV extremum seeking algorithms used in Figure 4.5. Many of the same parameters are to be used in Figure 4.9.

gTV		$nLTI$		nTV	
Param	Value	Param	Value	Param	Value
$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$	$\begin{bmatrix} 0.0044 \\ 0.0029 \end{bmatrix}$	$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$	$\begin{bmatrix} 5 \\ 5 \end{bmatrix}$	$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$	$\begin{bmatrix} 0.0044 \\ 0.0029 \end{bmatrix}$
$\begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix}$	$\begin{bmatrix} 0.006 \\ 0.004 \end{bmatrix}$	$\begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix}$	$\begin{bmatrix} 0.006 \\ 0.004 \end{bmatrix}$	$\begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix}$	$\begin{bmatrix} 0.006 \\ 0.004 \end{bmatrix}$
λ	0.99	ω_{HPF}	0.0004	λ	0.9995
N	10	ω_{LPF}	0.0004	N	15
T	4	ω_{RIC}	0.0004	T	4
A_u	$\begin{bmatrix} 27.5 & 0 \\ 0 & 27.5 \end{bmatrix}$	η_0	0.45	A_u	$\begin{bmatrix} 27.5 & 0 \\ 0 & 27.5 \end{bmatrix}$
b_u	$\begin{bmatrix} -2.5 \\ 22.5 \end{bmatrix}$	$\hat{v}_0$	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$	b_u	$\begin{bmatrix} -2.5 \\ 22.5 \end{bmatrix}$
K_g	$0.003I^{2\times 2}$	W_c	$\frac{1}{4000}$	W_d	$\frac{1}{4000}$
v_0	$\begin{bmatrix} 0.09 \\ -0.82 \end{bmatrix}$	$\hat{\Gamma}_0$	$2I^{2\times 2}$	v_0	$\begin{bmatrix} 0.09 \\ -0.82 \end{bmatrix}$
x_0	$\begin{bmatrix} 0.45 \\ 0 \\ 0 \end{bmatrix}$	δ	0	x_0	$\begin{bmatrix} 256.7 \\ -34.4 \\ 309.4 \\ 378.1 \\ 0 \\ 378.1 \end{bmatrix}$
P_0	$100I^{3\times 3}$			P_0	$100I^{6\times 6}$
				Basis	Tchebyshev
				α	0.999
				δ	0
				$\hat{\Gamma}_0$	$0.05I^{2\times 2}$

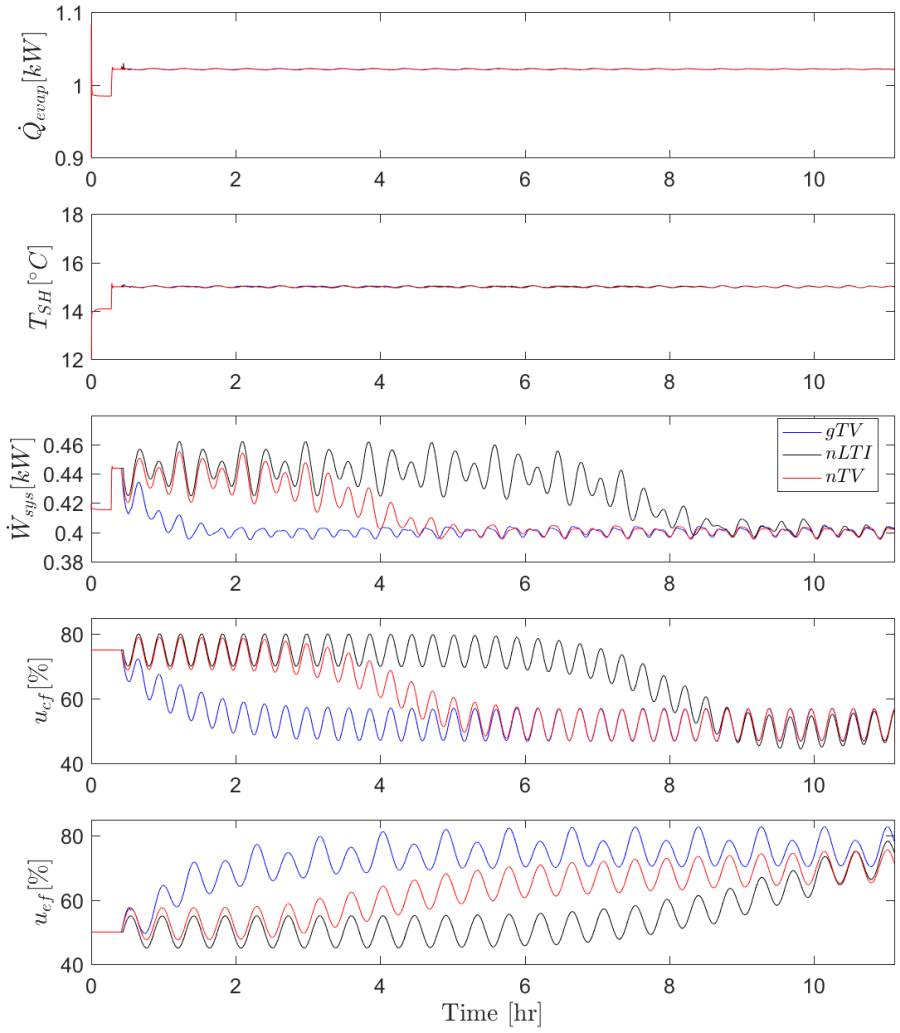

Figure 4.5: Simulation results for the comparison between dual input gTV, $gLTI$, and nTV approaches. Despite issues with K_g convergence shown in Figure 4.6, the optimizing inputs reach their desired values for both the $nLTI$ and nTV.

Figure 4.6 shows that K_g's $(2,2)$ element corresponding to u_{ef} causes large oscillations in the estimated gain for both the $nLTI$ and nTV approaches. To address this, the u_{ef} perturbation amplitude was increased to determine if enhancing the level of excitation from u_{ef} in $\dot{W}_{sys}$ quells the oscillations, while allowing the set point regulation to remain intact.

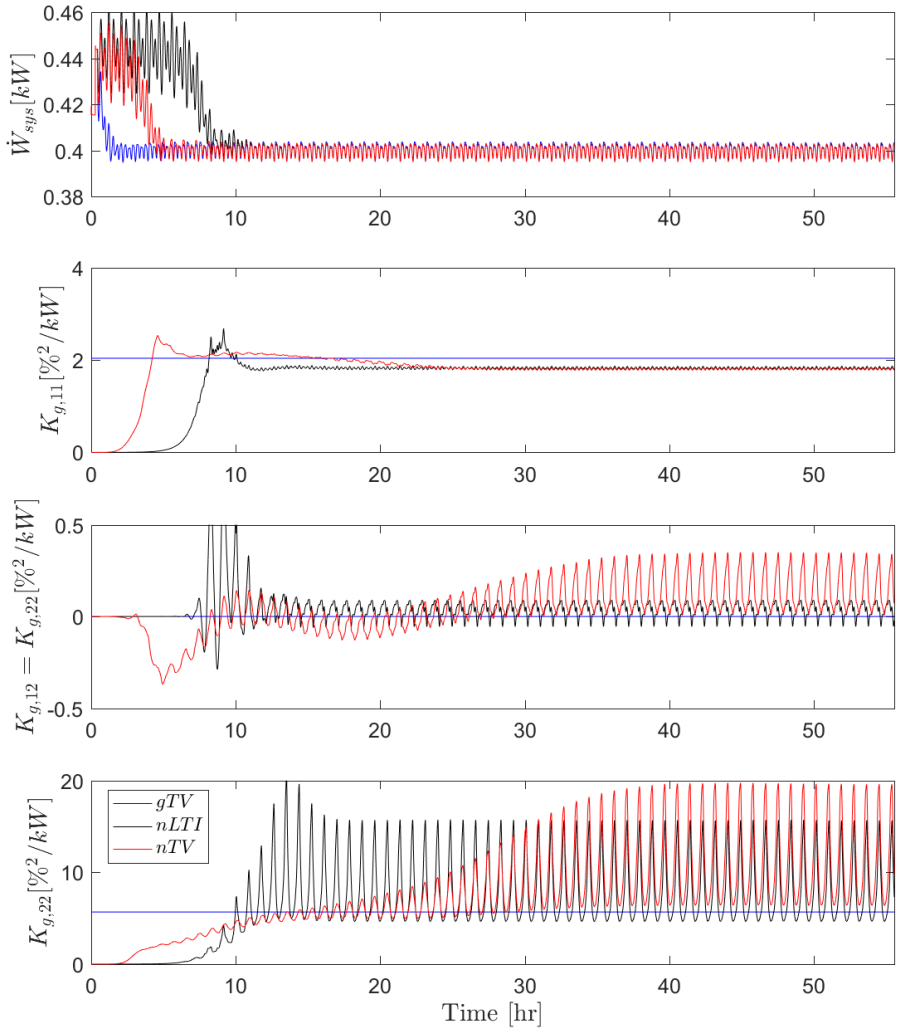

Figure 4.6: K_g trajectories for the noise free, dual input comparison of algorithms. The optimization gains do not settle to steady-state oscillations until 40 hours of simulation time has elapsed.

Figure 4.7 shows that increasing the u_{ef} perturbation amplitude has an insignificant effect on both the convergence time and the ability to meet the desired $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$ set points. Figure 4.8 shows a decrease in the steady state oscillations of K_g's $(2,2)$ value for both the $nLTI$ and nTV cases. The oscillations seen in Figure 4.6 could be a result of a numerical issue in the model or perhaps it is normal behavior when one diagonal component of the Hessian is smaller in magnitude than the other.

In addition to increasing the u_{ef}'s dither signal amplitude, the Hessian inverse filter coefficient could be chosen more conservatively. However, such a choice would sacrifice convergence time.

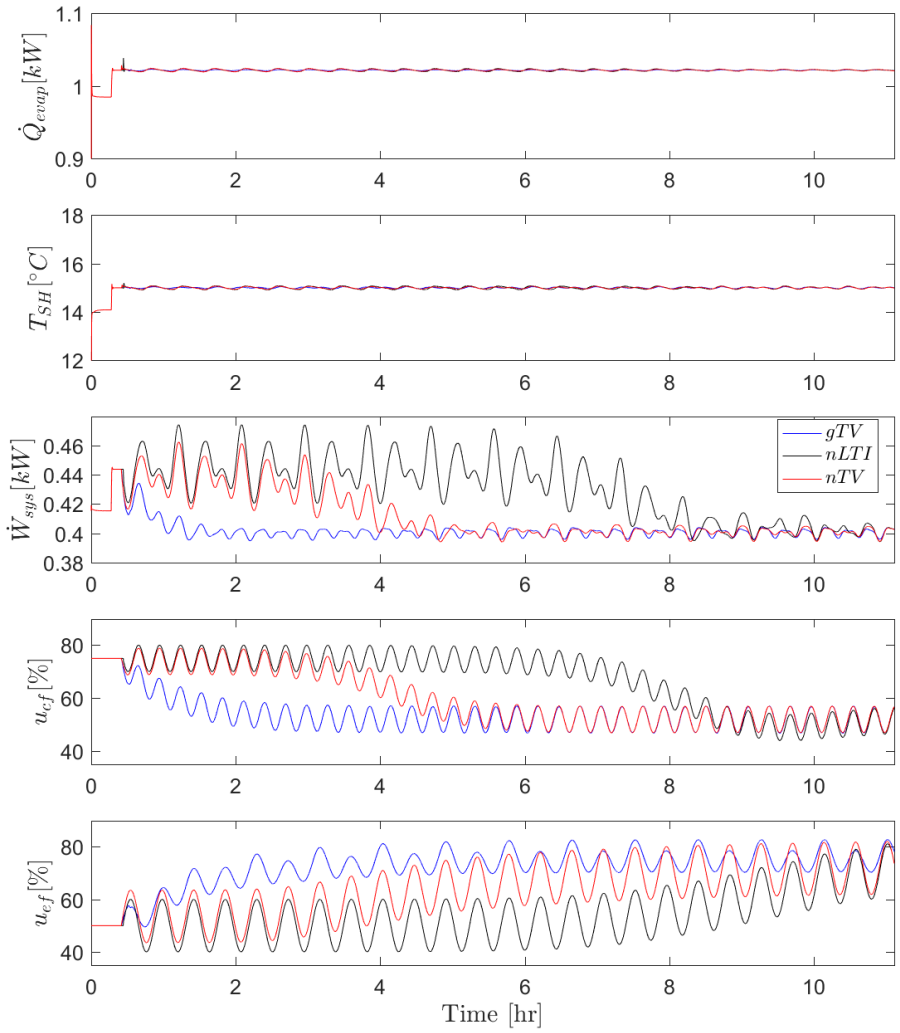

Figure 4.7: Comparison of the three algorithms after the Newton approaches' u_{ef} amplitudes have been changed from 5% to 10%. The amplitude change does not affect convergence time.

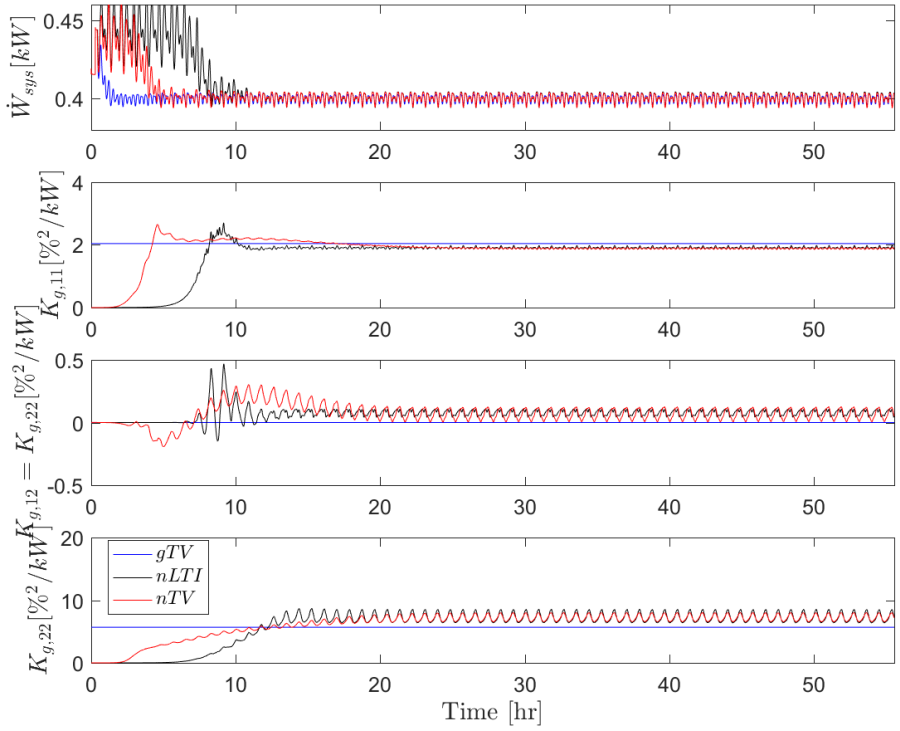

Figure 4.8: K_g trajectories for the noise free, dual input comparison of algorithms for the case where the Newton descent approaches' evaporator fan speed perturbation amplitudes have been doubled to introduce a more noticeable effect on the performance index output. Using larger evaporator fan perturbation amplitudes results in smaller oscillations of the Hessian component estimates at steady state and allows the controller to converge to an average Hessian estimate faster than in the previous case.

4.1.2.2 Realistic Noise Scenario Results

To test the effect of noise on the dual input approaches, the simulations in this section used the same noise level from Figure 4.3. For the simulation in Figure 4.9, the perturbation signals were identical to those given in Table 4.3, except that the Newton algorithms' u_{ef} perturbation amplitudes are 10%. Between Figure 4.5 and Figure 4.9, the nTV's α parameter was increased from 0.996 to 0.998. This increase results in a longer convergence time to the desired K_g, but increases robustness to noise.

Figure 4.9 shows that adding output noise further troubles convergence speed for the multi-input Newton descent algorithms and leads to fluctuations

in the K_g estimates, despite that the perturbation amplitude increase from the previous section has been kept in place. Even the nTV algorithm takes approximately 5 hours to converge to the desired gain. Meanwhile, the gTV's convergence is not significantly affected by noise, supporting the idea that it should be used for implementation on VCSs whenever possible.

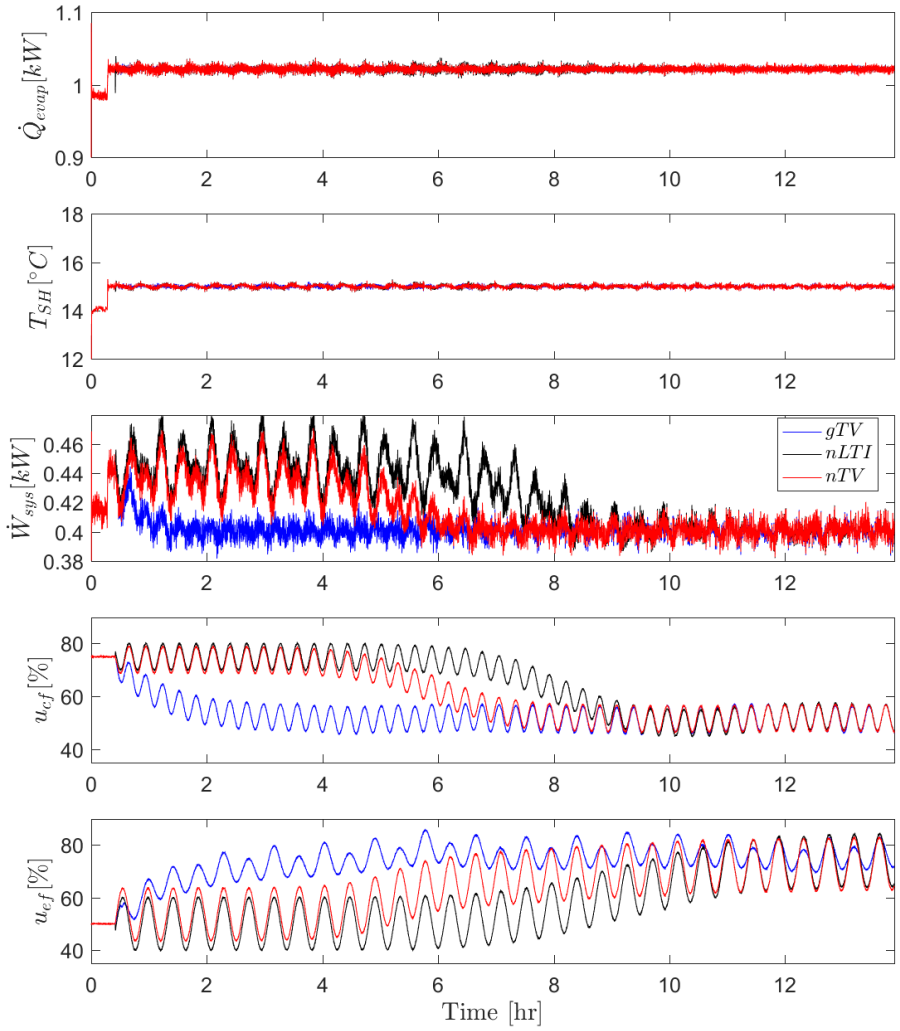

Figure 4.9: Realistic simulation results for the dual input case.

The output noise causes fluctuations in the K_g value for the $nLTI$ controller even after the performance index has apparently converged. While the nTV algorithm experiences similar transients, they appear to be less

severe, providing further evidence that using time-varying estimation provides a more favorable trade off between convergence speed and robustness to noise.

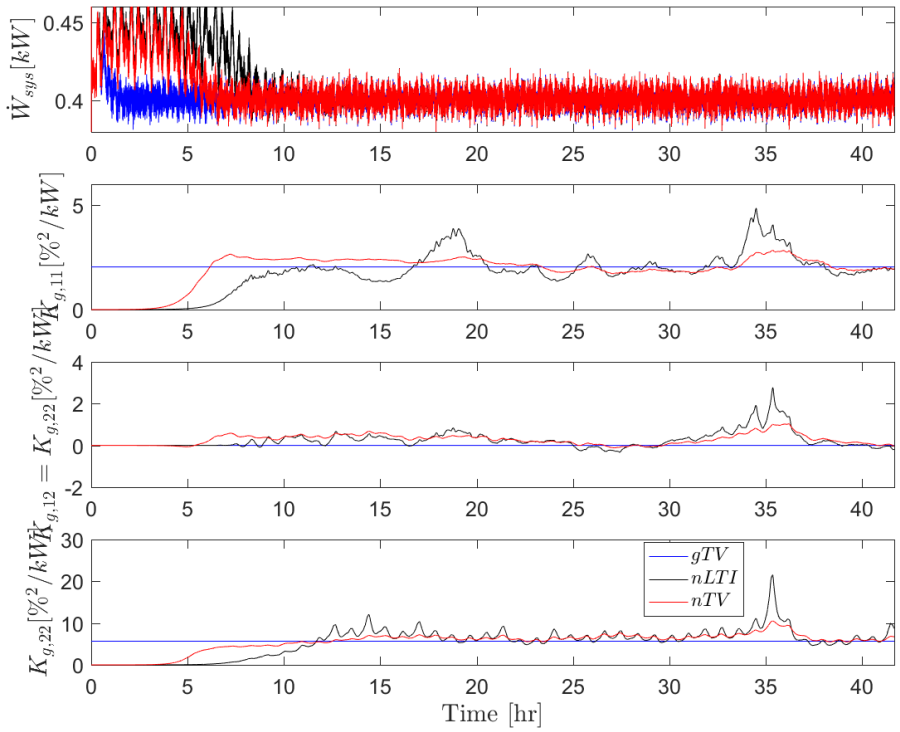

Figure 4.10: Controller gains for the realistic dual input controller simulation.

The Thermosys simulation results confirmed many of the trends observed in the case studies from Chapter 3 because the gTV outperformed both the $nLTI$ and nTV algorithms in terms of convergence time. Given a desired convergence rate, the Newton descent algorithms were able to successfully reproduce the approximate K_g matrix used by the gradient descent controller. A result absent from Chapter 3 was that increasing the perturbation amplitude can lead to a smoother estimate of the target K_g. This fact could provide an interesting direction for future work where the perturbation amplitudes are large until the target K_g is reached, at which point they reduce in magnitude and the algorithm becomes a gradient descent extremum seeking approach.

4.2 Experimental Results

This section serves two purposes: first, it provides confirmation that trends observed in simulation hold in experiment; second, it raises questions about how to transition from tuning of an extremum seeking algorithm in simulation to tuning extremum seeking parameters for a real system. The latter consideration raises the following fundamental questions for practical extremum seeking implementation:

> When can the exact same parameters from a simulation study be used in experiment?
>
> How reliable is the Thermosys modeling procedure for choosing an optimization gain?

The issues that give rise to these questions are system dynamics, measurement noise, and uncertainty about the performance index's curvature. In previous sections, system dynamics influenced the perturbation signal design, measurement noise affected derivative estimator design, and curvature affected the optimization/control law.

A dynamic simulation might give reliable knowledge of system dynamics that could be used to design the perturbation signal and a realistic level of measurement noise can be added to inform tuning of filter parameters. However, the Thermosys model's performance index curvature is affected by two factors that are difficult to control: the accuracy of component power consumption modeling and the accuracy of modeling steady state outputs.

The next two sections give experimental results to provide recommendations for the issues above. Section 4.2.1 gTV's convergence rate against the nTV's from the test in Figure 4.3. Section 4.2.2 shows convergence of the dual input gTV.

4.2.1 Single Input gTV vs. nTV

This section helps verify the simulation results from Figure 4.3 and gives insight into the tuning of ESC algorithms applied to a VCS. With the exception of the gTV, where K_g was set to 0.003 in simulation and 0.0036 in experiment, the gTV and nTV parameters were the exact same in simulation (see Table 4.2) and experiment. The gTV's gain value was chosen using

a desired convergence time constant of $3500s$ and a Hessian estimate from the single input calibration in Chapter 2.

The results in Figure 4.11 verify the realistic simulation in Figure 4.3; the nTV converged near the target K_g with some fluctuations in the gain values and the gTV's prior knowledge of the system Hessian obtained from a calibration experiment led to faster gTV convergence.

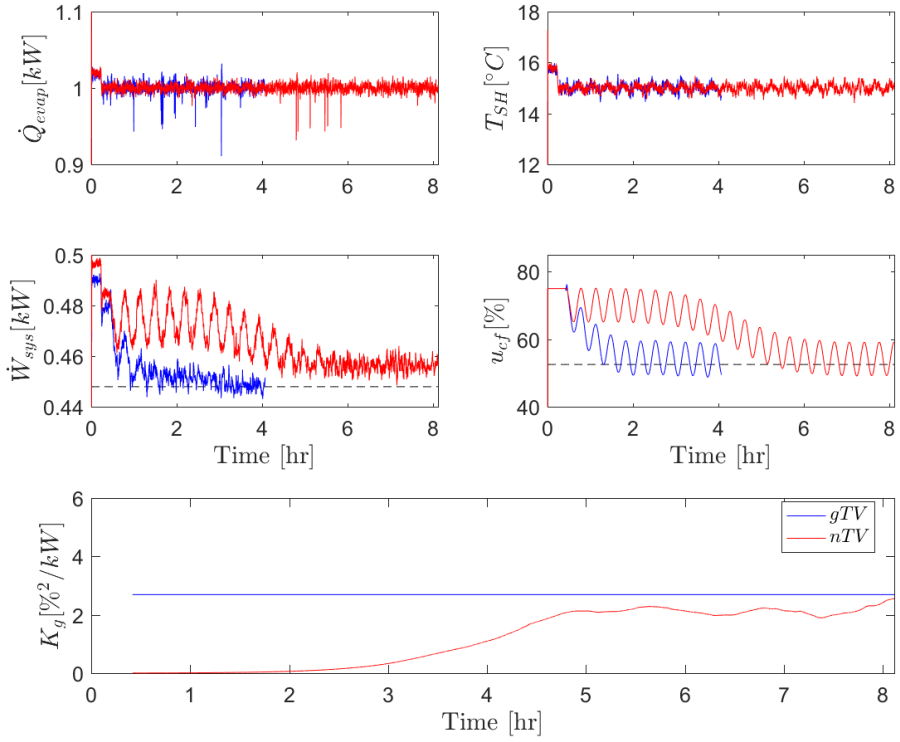

Figure 4.11: Experimental results comparing the performance of the single input gTV and nTV algorithms. Similar to the simulation results, the gTV algorithm's prior knowledge of the Hessian allows it to converge to the optimizing input much faster than the nTV algorithm can.

The middle plots show agreement between nTV's convergence time in the realistic simulation, about 5 hours, and in the experiment, about 6 hours (see Figure 4.3). One reason for the discrepancy could be that the nTV estimates a larger Hessian value than what was found in the calibration during Chapter 2.

A nice result is that both simulation and experiment agree on the trade off between obtaining prior knowledge and achieving good convergence time.

This fact confirms that a good enough dynamic model and a realistic measurement noise scenario can lead to a sound choice of tuning parameters. But while simulation results are reliable enough to predict the nTV's efficacy, can they also supply reliable information about the performance index's Hessian?

The fact that K_g increases by just 20% suggests that the simulation is reliable enough. To confirm this, the gTV simulation in Figure 4.3 was repeated with all of the same parameters in Table 4.2 except for K_g, which was changed to 0.0036 to match the value used in experiment.

Figure 4.12 shows that when the same gTV algorithm is implemented in simulation and experiment, the results are not significantly different. This confirms that tuning the gTV in simulation can be valuable for implementing the approach on a real system and avoiding the added convergence time of Newton descent. While initializing nTV with the correct Hessian value would improve the convergence time, simulation results from Chapter 3 showed that even when the Hessian changes with disturbances, Newton descent is not always advantageous.

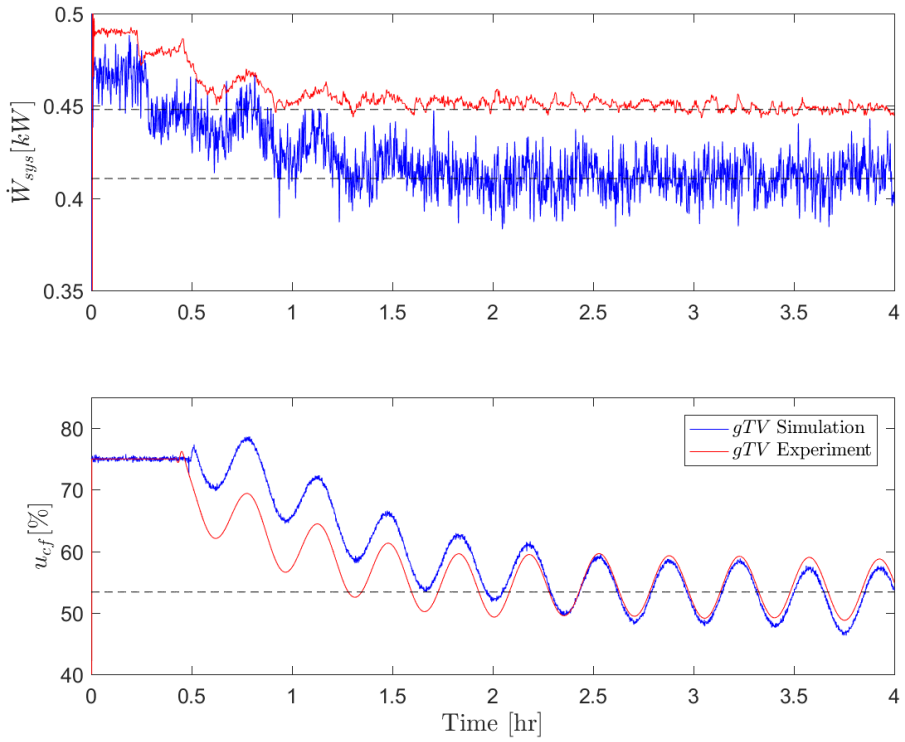

Figure 4.12: Comparison between the gTV algorithm in simulation and experiment. The gTV parameters are not changed between the two tests.

It is somewhat surprising to see that the simulation model predicted the optimal input with high precision, especially given that the minimum $\dot{W}_{sys}$ values are different between simulation and experiment. Accurately predicting this optimal input could be a result of modeling effort and some coincidence. In modeling a system from scratch, it is hard to know whether the final result will match the experiment. It should be clear that a match between optimal input values is unimportant for successful gTV convergence. Instead, the only properties that matter are performance index curvature, dynamics, and noise.

A final consideration is the opportunity cost of calibrating a system model. While calibrating K_g using localized perturbations at steady state conditions takes about 4-5 hours, calibrating the model involves several hours of system run time without attempting any kind of optimization. Before deploying extremum seeking on a real system, this fact should be considered.

4.2.2 Dual Input gTV

The goal of the dual input gTV experiment was to see if the Thermosys model would be useful for dual input ESC design. To find a K_g for the gTV that would give a convergence time constant of $4000s$, the calibration from Chapter 2 was used to generate an estimate of the Hessian. It happens that this calibrated Hessian was not drastically different from the Hessian estimate predicted by the dual input, noise free K_g results in Figure 4.8. This fact suggests that like the single input case, a single K_g value would lead to successful extremum seeking performance in both simulation and experiment. Apart from K_g, the initial inputs, and the desired convergence time, where

$$K_g = \Gamma W_d = \begin{bmatrix} 9.44 & -0.5503 \\ -0.5503 & 30.95 \end{bmatrix} \frac{1}{4000} = \begin{bmatrix} 0.0024 & 0 \\ 0 & 0.0077 \end{bmatrix}, \qquad (4.1)$$

the filter parameters and perturbation amplitude are the same in both simulation (Table 4.3) and experiment.

Figure 4.13 confirms that the $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$ setpoints are met in both cases. However, the fan perturbations seem to have a more significant

effect on steady state $\dot{Q}_{evap}$ and T_{SH} values in experiment than in simulation, where the perturbations are barely noticeable in these outputs. However, the desired $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$ values are attained on average in each case.

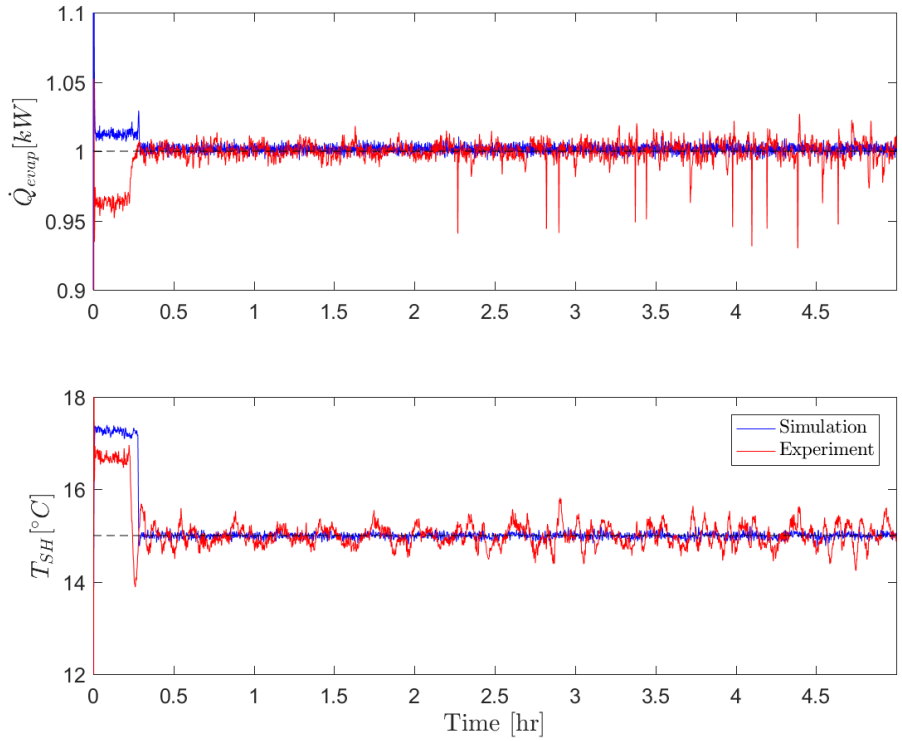

Figure 4.13: Comparison of implementations of the same gTV algorithm in simulation and experiment. Although the gTV parameters are not changed between the two tests, the results are similar.

Figure 4.14 confirms that convergence is successful in both experiment and simulation, despite the fact that identical parameters were used for each and there was risk of mismatch between the system and the model. Therefore, the simulation model is able to capture the experimental system's curvatures in both the u_{cf} and u_{ef} directions. While both the u_{cf} and u_{ef} inputs contribute to minimizing power consumption, comparing Figure 4.14 with Figure 4.12 shows that changing u_{ef} from 100% to its optimal value at about 75% does not produce a significant increase in energy savings. The dual input case's minimum $\dot{W}_{sys}$ is 0.44 kW. By comparison, the single input case achieves a minimum $\dot{W}_{sys}$ of 0.45kW, an increase of about 2%. Choosing the single input or dual input approach depends on whether the potential in-

crease in energy savings is worth decreasing the convergence rate. How much energy savings can be achieved depends on an individual system's properties.

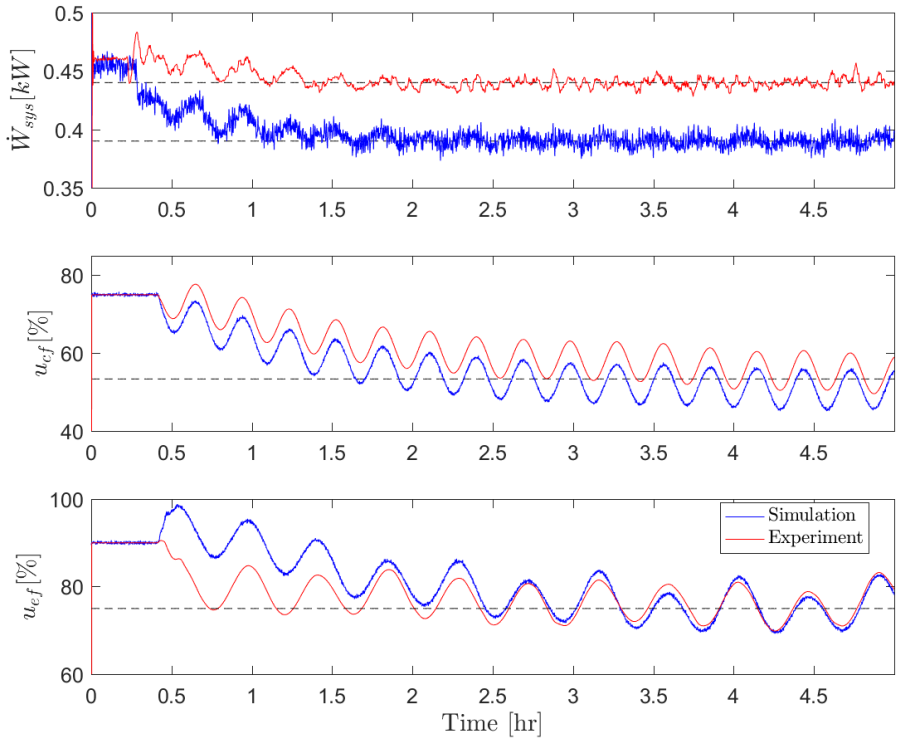

Figure 4.14: Comparison between the implementations of a gTV algorithm in simulation and experiment. The gTV parameters are not changed between the two tests.

4.3 Conclusions

This chapter's simulation and experimental results support the following conclusions from Chapter 3: time varying filters are preferable to LTI filters with demodulation for derivative estimation; gTV is preferable to both Newton descent algorithms when sufficient information about the VCS's performance index's curvature is available. Interestingly, helpful curvature information came not just from the experimental calibrations in Chapter 2, but also from the Thermosys model. However, building a model to capture curvature takes effort and system run time. When such an effort or wasted runtime is undesirable, Newton descent is an attractive option. Simulation results show

that nTV offers improved convergence speed relative to the $nLTI$ approach, while an experiment confirms that nTV is an effective form of extremum seeking. Table 4.4 summarizes the pros and cons of using gradient descent versus Newton descent extremum seeking algorithms for application to VCSs.

Table 4.4: Pros and cons of gradeit.

	Gradient Descent	Newton Descent
Pros	-Fewer parameters needed for estimation -Low sensitivity to noise	-Automatically discovers performance index curvature -Achieves desired optimization gain
Cons	-Requires identification when there is uncertainty about the Hessian	-Hessian inverse estimate is sensitive to noise -Reaching the desired gain requires a time investment

While studies in the literature such as [17, 12] indicated that Newton descent outperformed gradient descent, the Newton ESC's sensitivity to noise was not examined. For VCSs that are different from the one considered in this study, the advantages of Newton descent versus gradient descent should be evaluated for the system in question. Any indication that the curvature is highly sensitive to operating condition is a good reason to lean towards using Newton descent [6].

The main purpose of this chapter was to consider the convergence of the extremum seeking algorithm algorithms for a constant disturbance scenario. While such case studies are helpful for understanding extremum seeking behavior, they likely do not frequently occur in practice. The next chapter focuses on transient performance improvement for the case where changing the cooling capacity set point alters the optimal input.

CHAPTER 5

COMBINING SELF-OPTIMIZING CONTROL AND EXTREMUM SEEKING CONTROL FOR REAL-TIME VAPOR COMPRESSION SYSTEM OPTIMIZATION

While the previous chapter examined the challenges facing extremum seeking convergence, this chapter shows how self-optimizing control can improve extremum seeking control performance when a disturbance changes the optimal input settings. This work follows results from Section 3.2.1, which showed that combining extremum seeking and self-optimizing control can reduce the degree to which disturbances influence the optimal input.

Chapter 2 showed that there are a wealth of VCS measurements available for self-optimizing control implementation. By contrast, the previous chapter's implementation of extremum seeking was intuition based; no considerations were made regarding the effect of disturbances on the optimal fan inputs.

Using results from experiment and simulation, this chapter evaluates the benefits of combined ESC and SOC from Figure 5.1. standard ESC is a performance benchmark representing the intuition based extremum seeking approach used in the previous chapter. Both approaches guarantee that steady state inputs will be optimal, but the combined ESC and SOC is designed to use the two algorithms in concert to perform optimization. In the figure below, the combined ESC and SOC's feedback controller is designed to let the ESC use a reference input, r_{ESC}, rather than an actuator input, v. To implement the SOC feedback, H combines the process measurements, y, into a self optimizing controlled variable, y_{SOC}. As in previous sections, the disturbance input is w and the performance index output is J. The reference, r, represents performance objectives that must be met for the optimization to make sense.

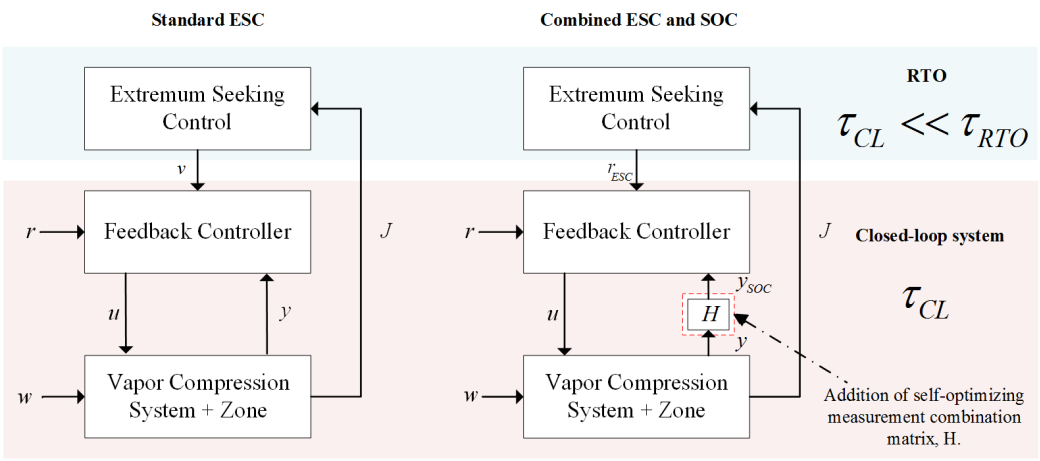

Figure 5.1: ESC acts as a RTO controller for the closed loop system. In standard ESC, the optimizing input is chosen using intuition. In combined ESC and SOC, the extremum seeking input is optimally chosen using SOC methods.

Figure 5.1 shows that while ESC performs optimization with a slow settling time, τ_{RTO}, SOC can achieve optimization at the plant's settling time, τ_{CL}. Chapter 3 showed that a well designed SOC can perform optimization once the ESC has converged. Two of the SOC design methods from Chapter 3 are reviewed below:

Combined ESC and Local SOC: The local approach involves using a system model. A numerical quadratic approximation of the cost function's Hessian matrix and a numerical linear approximation of input to output gains are used to find H.

Combined ESC and Data SOC: The data based approach relies on optimal data generated by a standard ESC to find H. Using this data, the singular value decomposition procedure from Section 3.2.1 finds invariants in the process data that produce a self-optimizing measurement combination.

In Section 5.1, the steady state optimization problems for standard ESC and combined ESC and SOC will be presented and compared. Section 5.2.3 uses a simulation to evaluate the local SOC and ESC's performance against the standard ESC's performance. Next, simulation results compare the local and data based SOC methods. In Section 5.3, the data based combined ESC

and SOC method will be evaluated against the standard ESC method in experiment. Section 5.4 gives an overview of the chapter's findings.

5.1 Problem Description

The combination of ESC and SOC in this chapter is restricted to the simple case of a scalar ESC input, $v \in \mathbb{R}$, and a scalar disturbance, $w \in \mathbb{R}$. The standard ESC depicted in Figure 5.1 is identical to the RTO approach shown in Figure 4.1, where an LQR+I maintains $\dot{Q}_{evap}$ and T_{SH} at their desired references. Eq. (5.1) represents the optimization problem from the standard ESC case, where the single optimization input is $v = u_{cf}$ and the single uncontrollable disturbance is $\dot{Q}_{evap,ref}$. The superheat set point, $T_{SH,ref}$, is not changed.

$$\begin{aligned} \min_{v} \quad & \dot{W}_{sys}(v,w) \\ \text{s.t.} \quad & \begin{bmatrix} u_{kp} \\ u_v \\ \dot{Q}_{evap} \\ T_{SH} \\ T_4 \\ T_{2-3} \end{bmatrix} = \begin{bmatrix} g_1(v,w) \\ g_2(v,w) \\ g_3(v,w) \\ g_4(v,w) \\ g_5(v,w) \\ g_6(v,w) \end{bmatrix}, \; g_3(v,w) = \dot{Q}_{evap,ref}, \; g_4(v,w) = T_{SH,ref} \\ & u_{ef} - 100 = 0 \end{aligned} \tag{5.1}$$

Setting $u_{ef} = u_{ef,max} = 100$ simplifies the optimization problem, but it could be included as an optimization input as well. As in Chapter 2, the inputs, u_{cf}, u_{kp}, and u_v are constrained to lie between 0% and 100% in normal circumstances. When the inputs might reach their constraints, it is necessary to account for these situations using anti-windup control or model predictive control. However, the simulations and experiments in this section are designed to avoid constraints, which change the self-optimizing control problem.

The optimization problem in (5.1) can be reformulated for implementation of the combined ESC and SOC approach using the following steps.

1. Determine the self-optimizing measurement combination, H, using a

system model or optimal process data.

2. Use H to combine the unregulated process measurements from (5.1), y_p. In this study $y_p = [u_{kp}\ u_v\ T_4\ T_{2-3}]$; of these variables, T_4 and T_{2-3} will be used for SOC.

3. Redesign the LQR+I from Figure 4.1 to use both v and u from the standard ESC in Figure 5.1 for regulation to r_{ESC}, the self-optimizing reference input. In this study, $u = [u_{kp}\ u_v]$ in the standard ESC case. In the combined ESC and SOC case, u becomes $u = [u_{kp}\ u_{cf}\ u_v]$. The SOC's reference input is a combination of T_4 and T_{2-3}. Therefore, H can be written as a 2 dimensional row vector, $[H_1\ H_2]$.

Using the steps above, the combined ESC and SOC optimization problem becomes (5.2), where γ represents a new steady state output mapping (formerly denoted by g).

$$
\begin{aligned}
\min_{r_{ESC}} \quad & \dot{W}_{sys}(r_{ESC}, w) \\
\text{s.t.} \quad & \begin{bmatrix} u_{kp} \\ u_{cf} \\ u_v \\ \dot{Q}_{evap} \\ T_{SH} \\ H_1 T_4 + H_2 T_{2-3} \end{bmatrix} = \begin{bmatrix} \gamma_1(v,w) \\ \gamma_2(v,w) \\ \gamma_3(v,w) \\ \gamma_4(v,w) \\ \gamma_5(v,w) \\ \gamma_6(v,w) \end{bmatrix} \\
& \gamma_4(v,w) = \dot{Q}_{evap,ref} \\
& \gamma_5(v,w) = T_{SH,ref} \\
& \gamma_6(v,w) = r_{ESC} \\
& u_{ef} - 100 = 0
\end{aligned}
\tag{5.2}
$$

Eqs. (5.1) and (5.2) show the unconstrained optimization problems that are compared in this chapter. The theory from Chapter 3 predicts that SOC will lead to a scenario where latter optimization problem is less sensitive to disturbances. The next section tests this hypothesis using steps in the commanded cooling capacity, $\dot{Q}_{evap,ref}$. To create a scenario where the heat load is an unmeasured disturbance, $\dot{Q}_{evap,ref}$ will not be directly measured by the RTO scheme, necessitating the use of self-optimizing control to detect unknown disturbance changes.

5.2 Simulation Case Study

Figure 5.2 shows the simulated heat load profile that alters the VCS's optimal input settings. The other disturbances, T_{csfi} and T_{esfi} shown in Figure 2.1 and mentioned in Chapter 2, remain constant throughout the simulation. Section 5.2.1 details the extremum seeking controller parameters employed in this section. Section 5.2.2 details the controlled variable selection calculations for both the combined ESC and local SOC and the combined ESC and data SOC optimizers. Finally, Section 5.2.3 performs a head to head comparison between the standard ESC, the combined ESC and local SOC, and the combined ESC and data SOC approaches.

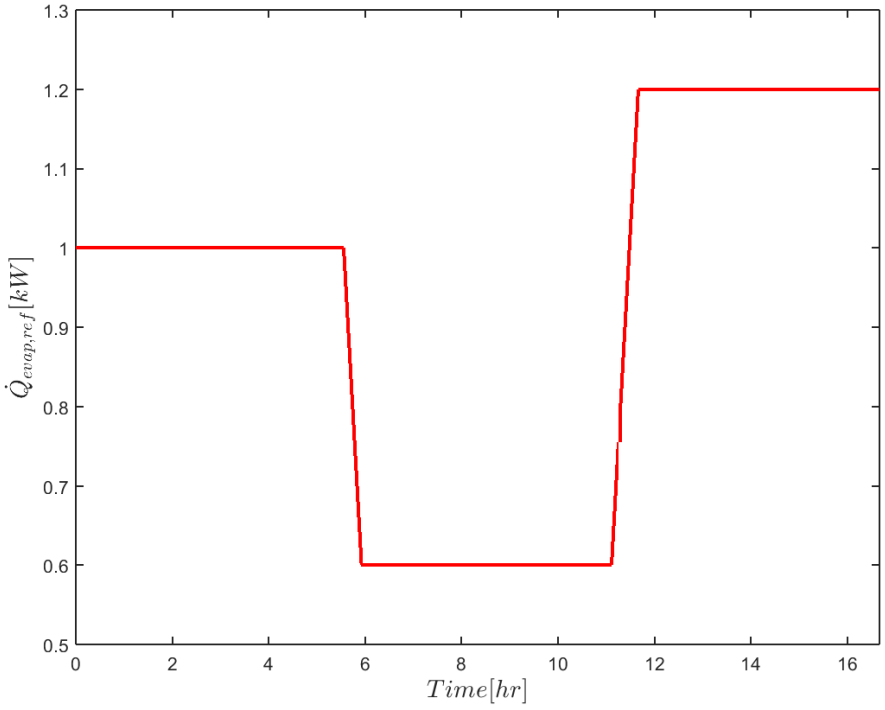

Figure 5.2: Heat load trajectory applied to each RTO controller.

5.2.1 Extremum Seeking Control

Because the focus of this chapter is how SOC can improve ESC performance, the ESCs were designed to have similar convergence rates for each test case. The *gLTI* algorithm from Section 3.1.4 was used here. Before designing the

ESCs, LQR+I feedback controllers with state augmentation from Section A.2 were used to meet the $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$ set points (and the r_{SOC} set point for the combined ESC and SOC approaches). To determine the plant's settling time, τ_{CL}, the closed loop system's 2% settling time to a step response was calculated using MATLAB's `stepinfo` command. These settling times are given by Table B.5, Table B.6, and Table B.7 respectively. The ESCs were designed to with equal perturbation frequencies to ensure a fair comparison. Likewise, the filter cutoff frequencies were chosen to be identical in order to produce the same filtering speeds and avoid giving one approach better ESC performance than the other. The optimization gains were hand tuned to achieve similar convergence rates at a nominal cooling capacity of $\dot{Q}_{evap} = 1.0kW$.

Table 5.1: Extremum seeking controller parameters.

Parameter	Standard ESC	Combined ESC and SOC
$\omega_1[rad/s]$	0.005	0.005
a_1	5[%]	0.4 [$^\circ C$]
$\omega_{HPF}[rad/s]$	0.0005	0.0005
$\omega_{LPF}[rad/s]$	0.0005	0.0005
K_g	$3.8[kW/\%]$	$0.1[kW/^\circ C]$

While the u_{cf}'s perturbation amplitude could be specified exactly for the standard ESC, the it is hard to make a reference perturbation that produces a precisely equivalent level of objective function excitation for the combined ESC and SOC approaches. To strive for similar excitation, the a_1 values were hand tuned in each case. The optimization gain, K_g, values were chosen somewhat arbitrarily by tuning the ESC convergence rates to be equal during the first transient.

5.2.2 Controlled Variable Selection

This section presents the controlled variable selection procedures necessary to implement the combined ESC and SOC approaches. Because the local SOC approach is model based, its H is close to the optimal combination for

minimizing loss. This fact is confirmed in Section 5.2.3, where the local SOC data is evaluated using a SVD.

5.2.2.1 ESC and Local SOC

Using the simulation model to find the optimal measurement combination involved first finding numerical approximations of the partitioned Hessian components, J_{vv}, $J_{vw} = J_{wv}^T$, and J_{ww}, about the optimal input value when $\dot{Q}_{evap,ref} = 1kW$. Next, the steady gains, G_v and G_w, were calculated. The measurement combination, H, is chosen to be in the nullspace of $G_w - G_v J_{vv}^{-1} J_{vw}$. Table 5.2 gives the Hessian components, gains, and resulting measurement combination, H.

Table 5.2: The simulation model was used to find J_{vv}, J_{vw}, G_v, and G_w numerically, giving all the components necessary to find the locally optimal measurement combination, H.

Variable	Value
J_{vv}	$1.9783e-4$
J_{vw}	-0.0032
J_{ww}	0.6111
G_v	$\begin{bmatrix} 0.0007 \\ -0.0907 \end{bmatrix}$
G_w	$\begin{bmatrix} -7.5539 \\ 12.3287 \end{bmatrix}$
H	$[1.44\ 1]$
y_{SOC}	$1.44T_4 + T_{2-3}$

5.2.2.2 ESC and Data SOC

Recalling the procedure in Chapter 3, a standard ESC must be used to generate a set of near optimal process data. Figure 5.3 shows this data generated in simulation. Following the procedure from 3.2.1.3, the T_4 and T_{3-2} measurements were centered about their mean values, giving ΔT_4 and ΔT_{2-3}. Next, the centered data was stacked into two rows to form $Y_{scl,0}$ and analyzed using SVD. Scaling was not necessary because ΔT_4 and ΔT_{2-3} have the same units and showed similar changes in magnitude. Figure 5.3 shows that the data based self-optimizing control analysis produced a combination of measurements with an optimal value that is less sensitive to heat load changes than the T_4 and T_{2-3} measurements individually. Table 5.3 gives the resulting U, Σ, and H.

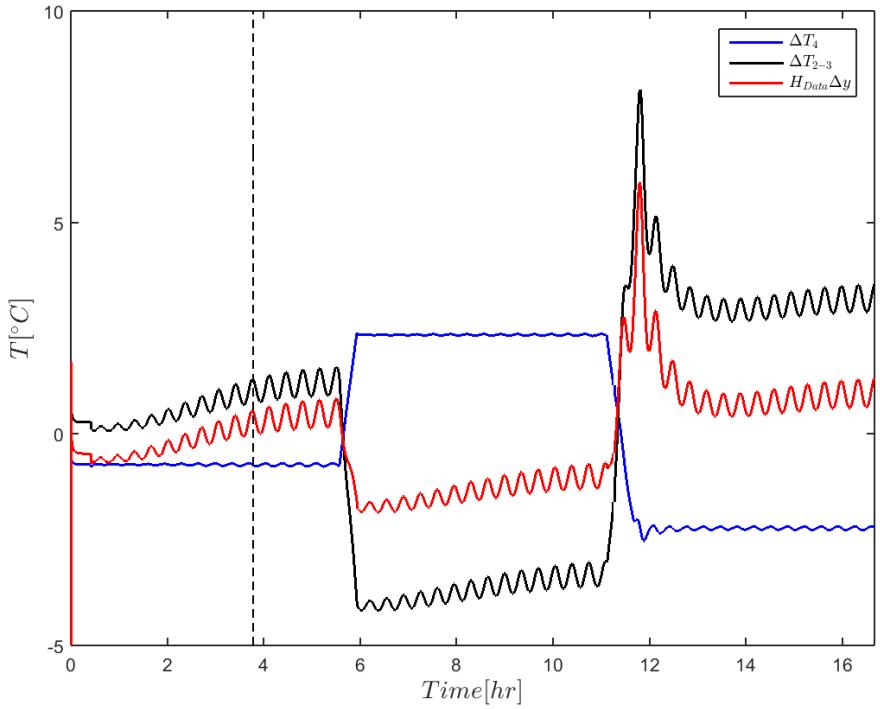

Figure 5.3: The optimal self-optimizing combination shows less variation in its optimal value following heat load changes than the T_4 and T_{2-3} variables alone.

Table 5.3: Singular value decomposition of the process data, $Y_{scl,0} = [\Delta T_4 \ \Delta T_{2-3}]^T$. The resulting measurement combination, H, is a scaled and transposed version of U's second column.

Variable	Value
U	$\begin{bmatrix} -0.5295 & 0.8483 \\ 0.8483 & 0.5295 \end{bmatrix}$
Σ	$\begin{bmatrix} 418.4724 & 0 & 0 & ... & 0 \\ 0 & 39.9242 & 0 & ... & 0 \end{bmatrix}$
H	$[1.6019 \ 1]$
y_{SOC}	$1.6T_4 + T_{2-3}$

5.2.3 Simulation Results

To evaluate the concept of combining ESC and SOC for vapor compression systems, the standard ESC's performance is compared to the ESC and local SOC first. Next, the two SOC approaches are compared to evaluate the performance loss associated with using near optimal data to choose the self-optimizing measurement combination matrix.

A prerequisite for comparing optimization performance is to show that the output constraints on $T_{SH,ref}$ and $\dot{Q}_{evap,ref}$ were met. Figure 5.4 and Figure 5.5 confirm that both combined ESC and SOC approaches and the standard ESC approach meet the desired cooling capacity and superheat set points on average, despite oscillations in the inputs. This shows that at a minimum, using the combined ESC and SOC approach does not affect the feedback controller's ability to meet the desired equality constraints.

In each test, the ESCs converge to the optimal inputs and close to the minimum power consumption following the heat load disturbances, illustrating that the RTO schemes act as they should.

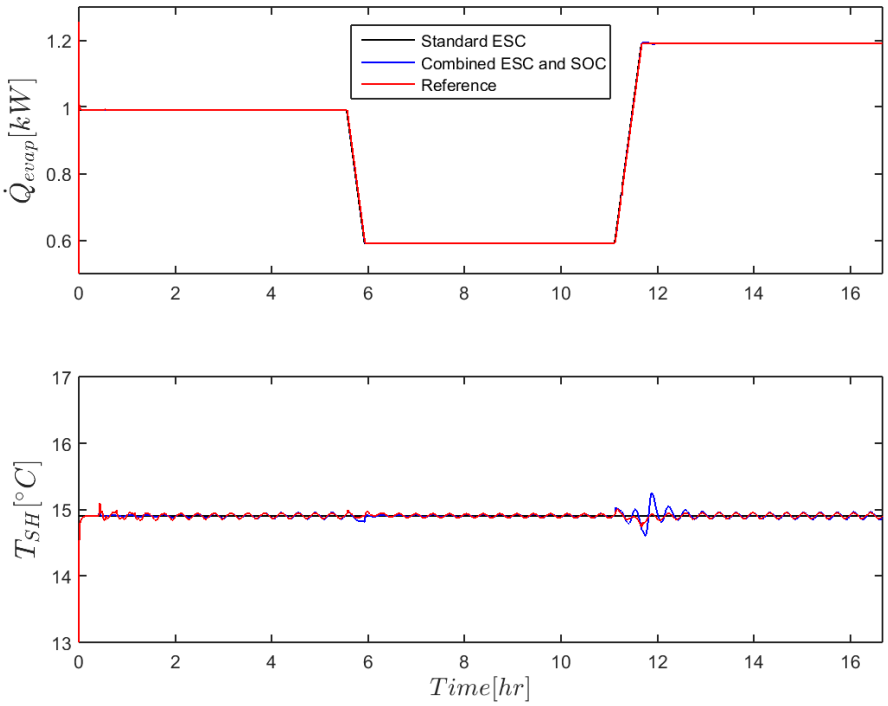

Figure 5.4: On average, the controllers are able to meet their desired $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$ values for both the standard ESC and the combined ESC and local SOC approach. Oscillations about reference values are the result of ESC perturbations.

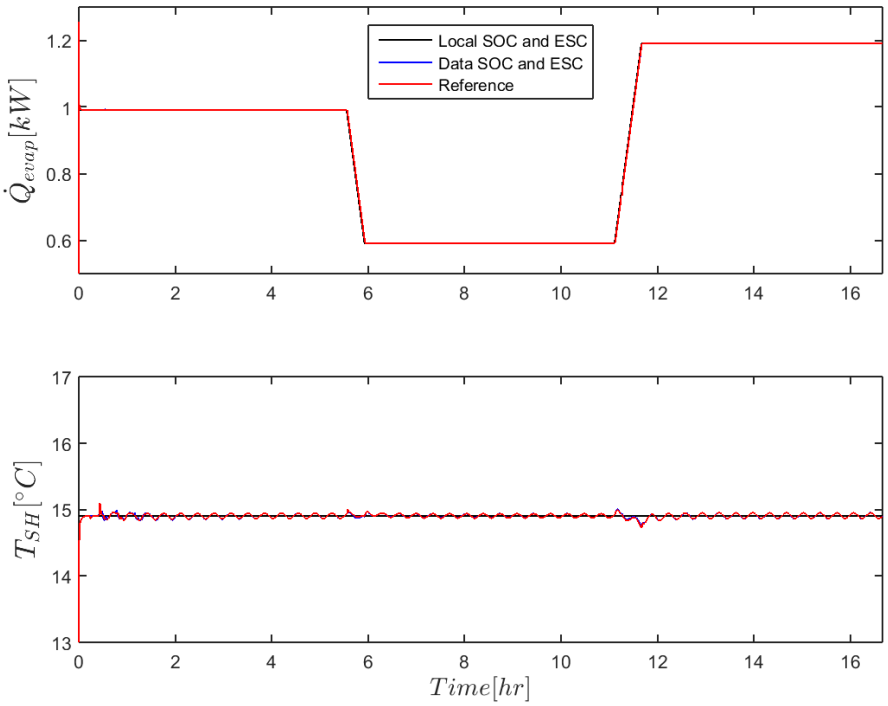

Figure 5.5: On average, the controllers are able to meet their desired $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$ values for both combined ESC and SOC approaches. Oscillations about reference values are the result of ESC perturbations.

Figure 5.6 shows that the inclusion of SOC in the RTO framework improves the transient performance of the optimizer. In the bottom plot, SOC does not help during the first convergence transient when the ESC is finding the optimal set point. Once the ESC has converged, it does not significantly deviate from the optimal set point for the rest of the simulation. This behavior is expected because the SOC measurement combination's optimal value was chosen to be invariant to the heat load disturbance.

The SOC's contribution is especially apparent during the second heat load step, where the standard ESC approach lags behind the step change in the optimal fan speed. During this transient, the value of the self-optimizing output spikes in the standard ESC simulation. There is a corresponding spike in the power consumption that indicates suboptimality of the standard ESC's input choice. Essentially, the standard ESC fails to regulate the self-optimizing combination to its optimal value and pays a penalty in performance loss. By contrast, the combined ESC and SOC turns the optimization

problem into a regulation problem during the transient and as a result, its performance loss is not noticeable.

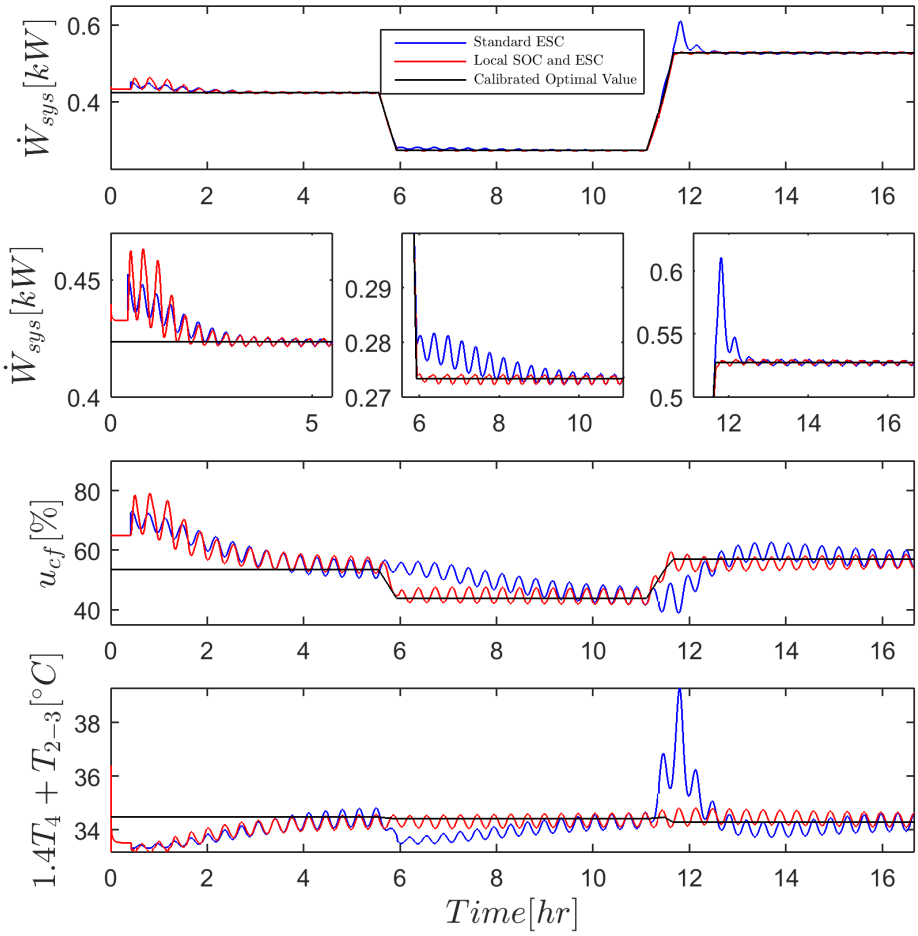

Figure 5.6: The combined ESC and local SOC's contribution to optimization is apparent from the fact that its u_{cf} signal leads the standard ESC's during the steps in $\dot{Q}_{evap,ref}$, despite that the initial convergence rates are similar.

While the combined ESC and local SOC approach is promising, it may not always be practical to use a model to choose a self-optimizing measurement combination. This is the case when plant-model mismatch is a concern in the design process.

The performance loss from using the data based SOC approach instead of the local SOC approach is evaulated in Figure 5.7. As expected, both SOCs make input adjustments to maintain the ESC's set point, but the local

SOC's input adjustment is closer to optimal because of its superior system knowledge. Comparing Figure 5.7 with Figure 5.6 indicates that the feedforward action of the combined ESC and data SOC still offers performance improvement relative to the standard ESC approach.

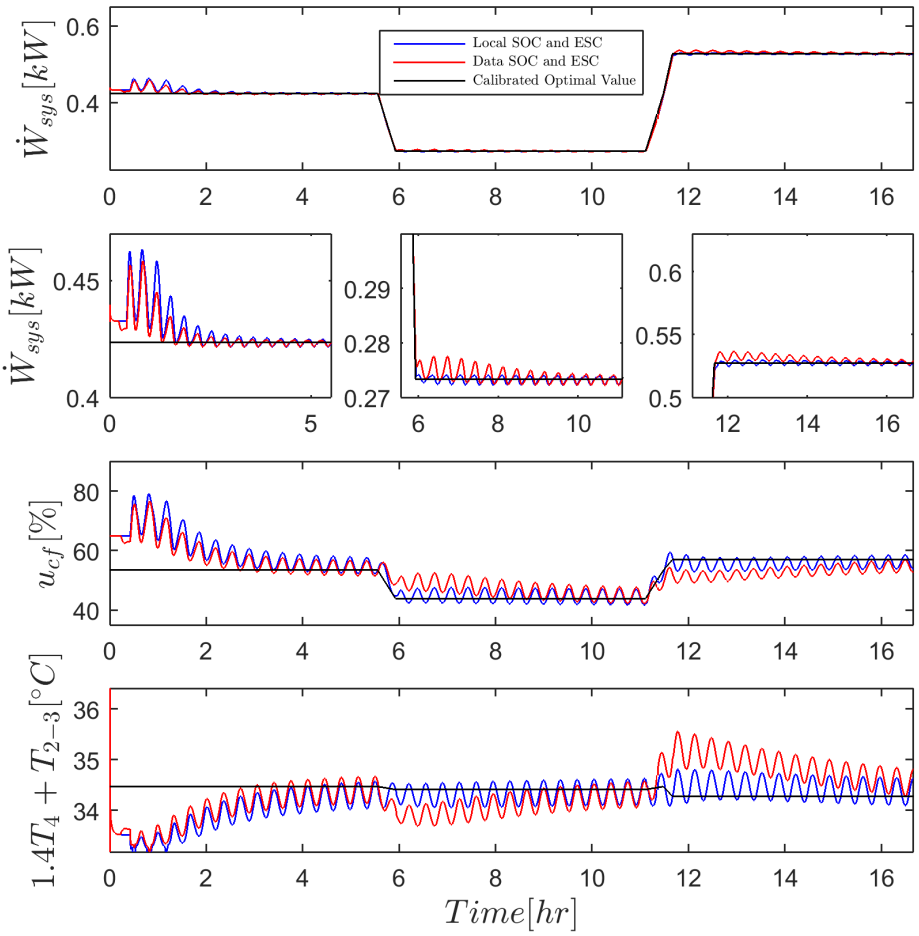

Figure 5.7: When the combined ESC and SOC approach uses a SOC measurement combination extracted from near optimal process data, the feedforward SOC action is not as accurate as the case where a locally optimal measurement combination is used.

The results for SOC designed using near optimal process data is encouraging for model free implementation of the approach; the performance improvement should be close to what could be achieved with a perfect system model. In the next section, the combined ESC and data SOC is compared to the standard ESC using a similar case study for the system in Figure 2.2.

5.3 Experimental Case Study

To evaluate the transient performance of the combined ESC and data SOC against the standard ESCin experiment, the controllers were subjected to the heat load profile shown in Figure 5.8 and a constant $T_{SH,ref} = 15°C$.

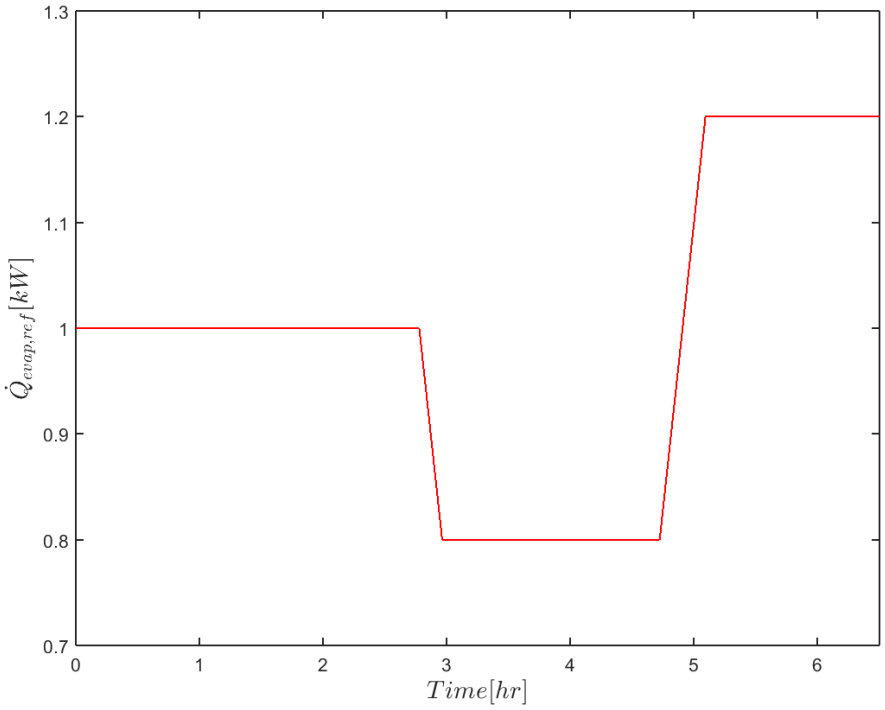

Figure 5.8: Heat load trajectory applied to each RTO controller.

As in the previous section, a combination of at least two measurements is sufficient to achieve zero loss when the null space method SOC assumptions are satisfied. Unlike the simulation case, the assumption of no noise is violated and unmodeled disturbances, T_{amb} and RH, may change during the experiment. Additionally, calibration results from C.3 show that the Hessian of $\dot{W}_{sys}$ changes with $\dot{Q}_{evap,ref}$. However, the SOC implementation from Section 5.3.2 confirms that the data based null space method can be a strong practical tool in vapor compression system applications.

5.3.1 Extremum Seeking Control

The closed loop system settling times, τ_{CL}, are given in Table C.4 and Table C.5 from Section C.2.1 and Section C.2.2 respectively. As in the simulation case, τ_{CL} was calculated by determining the closed loop system's state space realization and applying MATLAB's `stepinfo` command to find 2% settling time. A different σ was chosen for the standard ESC and combined ESC and SOC approaches to achieve equal perturbation frequencies and account for the discrepancy in τ_{CL}. The LTI filter cutoff frequencies were chosen to be identical.

Table 5.4: Extremum seeking controller parameters.

Parameter	Standard ESC	Combined ESC and SOC
$\omega_1[rad/s]$	0.007	0.007
a_1	5[%]	0.4 [$^\circ C$]
$\omega_{HPF}[rad/s]$	0.001	0.001
$\omega_{LPF}[rad/s]$	0.001	0.001
K_g	3.8[$kW/\%$]	0.1[$kW/^\circ C$]
$J_{vv}(0.8kW)$	0.96e-4 [$kW/\%^2$]	0.007 [$kW/^\circ C^2$]
$J_{vv}(1.0kW)$	1.12e-4 [$kW/\%^2$]	0.005 [$kW/^\circ C^2$]
$J_{vv}(1.2kW)$	1.58e-4 [$kW/\%^2$]	0.003 [$kW/^\circ C^2$]
$\tau_{RTO}(0.8kW)[s]$	11,456	8,022
$\tau_{RTO}(1.0kW)[s]$	9,913	5,568
$\tau_{RTO}(1.2kW)[s]$	7,211	13,192

Similar to the simulation results, the u_{cf} perturbation amplitude could be specified exactly for the standard ESC. The perturbation amplitude, a_1, was chosen for the combined ESC and SOC's reference in order to produce u_{cf} oscillations similar in magnitude to the standard ESC's perturbation signal. However, the system nonlinearity makes the actual u_{cf} perturbations subject to change for the combined ESC and SOC. The K_g values were chosen so that improving ESC convergence time could be ruled out as a source of RTO performance improvement. Hessian values were taken from the columns labeled c_3 and d_3 in Table C.6 and Table C.7 respectively. Because these Hessian values change as a function of disturbances, the convergence rates of

each ESC can vary. Compared to the simulation from the previous section, the ESC parameters were more aggressively tuned to shorten the experiment time. A faster ESC meant less noticeable performance improvement for the combined ESC and SOC approach.

5.3.2 Controlled Variable Selection

The same procedure from Section 5.2.2.2 for selecting the self-optimizing measurement combination in simulation was employed here. Table 5.5 shows the U, Σ, and H derived from the experimental data.

Table 5.5: Results from singular value decomposition of the centered process data, $\Delta Y = [\Delta T_4 \ \Delta T_{2-3}]^T$. H is a scaled and transposed version of the second column of U.

Variable	Value
U	$\begin{bmatrix} -0.5062 & 0.8624 \\ 0.8624 & 0.5062 \end{bmatrix}$
Σ	$\begin{bmatrix} 194.3 & 0 & 0 & ... & 0 \\ 0 & 33.57 & 0 & ... & 0 \end{bmatrix}$
H	$[1.7 \ 1]$
y_{SOC}	$1.7T_4 + T_{2-3}$

5.3.3 Experimental Results

In both tests, the steady state optimization problems were satisfactorily solved. Figure 5.9 demonstrates that the $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$ constraints were met by the linear quadratic controllers. At steady state, power consumption values remain close to the calibrated minimum values from Appendix B. Although the optimal u_{cf} at $\dot{Q}_{evap,ref} = 1.2kW$ varies between

tests, the extremum seeking controllers provide adequate compensation. Figure 5.11 shows that other disturbances likely do not have a significant effect on the system's power consumption, validating the assumption that $\dot{Q}_{evap,ref}$ is the only meaningful disturbance.

Figure 5.10 shows that the combined ESC and SOC improves upon the standard ESC's transient performance following the second $\dot{Q}_{evap,ref}$ step. In the first transient, the ESCs recover from suboptimal initial conditions equally well and at similar rates. Following the change in $\dot{Q}_{evap,ref}$ from $1.0kW$ to $0.8kW$, both the standard ESC and the combined ESC and SOC show equally good tracking of the minimum power consumption, a phenomenon observed in the simulation case study. The step from $0.8kW$ to $1.2kW$ leads to a power consumption spike in the standard ESC test, but the spike is absent from the combined ESC and SOC test. This can be attributed to the self-optimizing controller increasing the condenser fan speed to maintain r_{ESC}, resulting in a faster convergence to the combined ESC and SOC's eventual steady state value of u_{cf}. By contrast, the dip in u_{cf} chosen by the standard ESC corresponds to the spike in $\dot{W}_{sys}$. Figure 5.10 also shows that the optimal value of $1.7T_4 + T_{2-3}$ is nearly invariant to heat load disturbances for both experiments, though its value spikes in the standard ESC test. This fact illustrates the main advantage of using SOC for optimization: the combined ESC and SOC approach converts the standard ESC's gradient descent problem to a regulation objective for $1.7T_4 + T_{2-3}$.

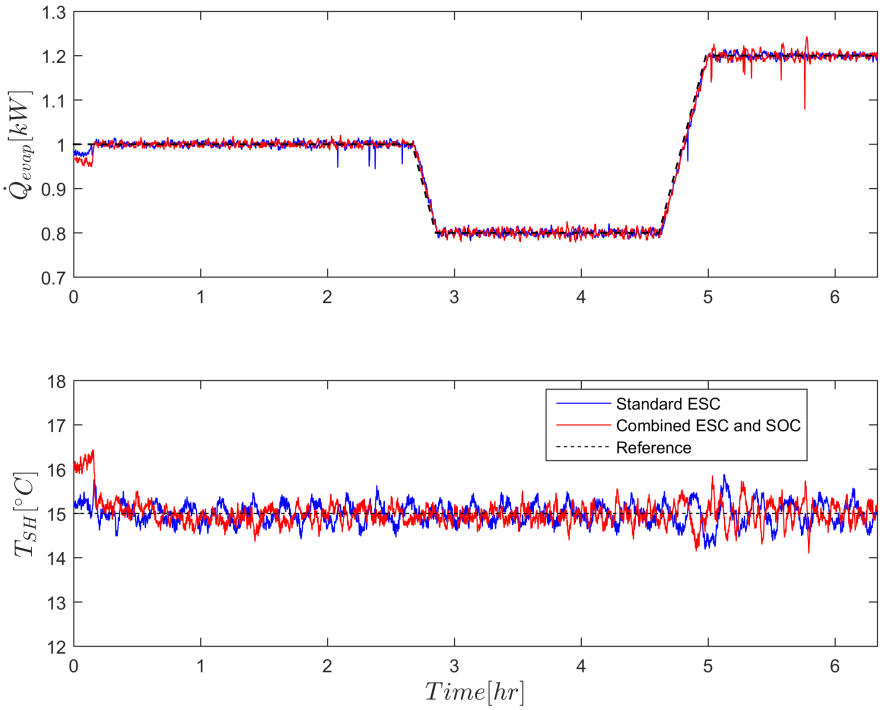

Figure 5.9: On average, the controllers are able to meet their desired $\dot{Q}_{evap,ref}$ and $T_{SH,ref}$ values in experiment. Oscillations about reference values are the result of ESC perturbations.

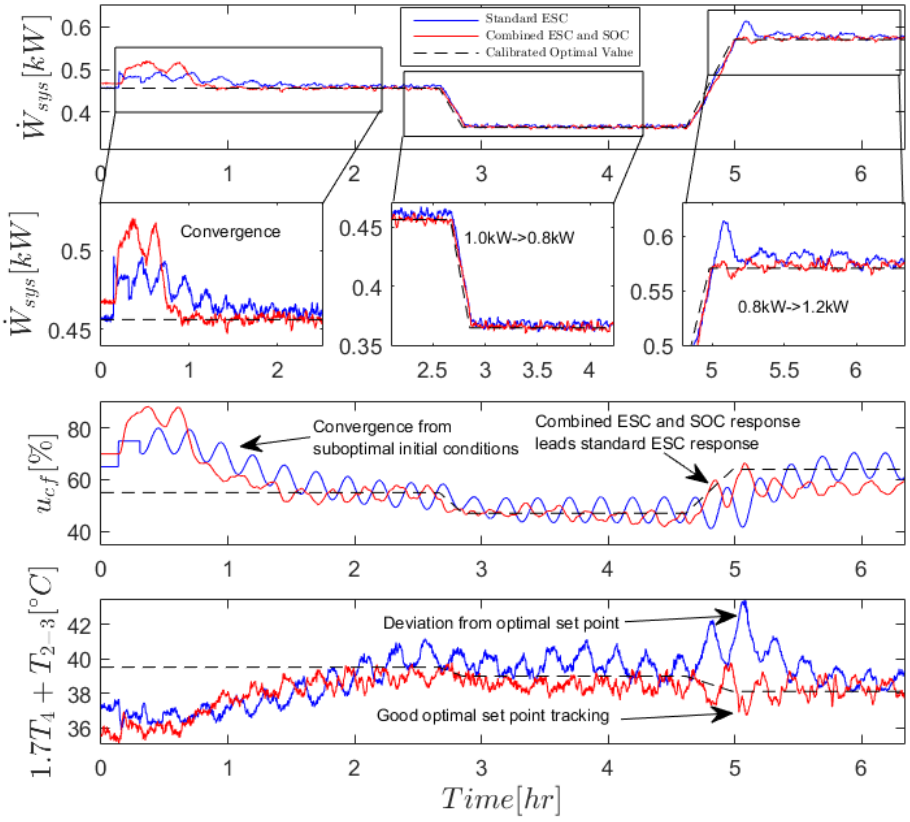

Figure 5.10: The feedforward action of the combined ESC and SOC is apparent from the fact that its u_{cf} signal leads the standard ESC's during the steps in $\dot{Q}_{evap,ref}$. At steady state the trajectories are similar but slightly offset from each other because of run to run variations in the optimal inputs.

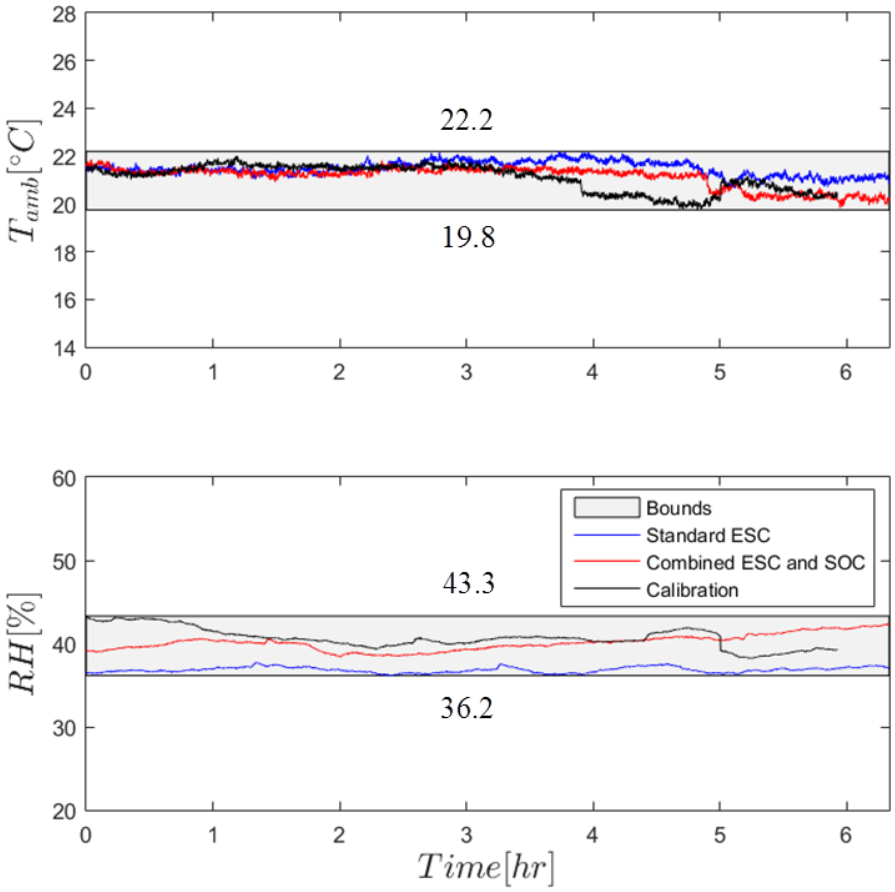

Figure 5.11: T_{amb} and RH remain within the upper and lower bounds indicated by the gray boxes during each controller test and the calibration. These disturbances could account for the slight difference in the optimal u_{cf} between tests, but do not seem to affect the overall result.

5.4 Conclusions

Both simulation and experimental results showed that combining SOC and ESC can improve the RTO's transient performance. While the settling times for each RTO were similar during the initial convergence transients, the combined ESC and SOC approaches appeared to shorten the RTO settling time by removing the requirement for ESC to update its set point. Before implementing SOC, assumptions about the number of important disturbances and the linear relationship between the disturbance and the optimal input were

invoked. Combined ESC and SOC could be implemented using the system model in simulation, but process data generated using closed loop extremum seeking control was necessary to achieve a model free combined ESC and SOC in experiment. One disadvantage of the combined ESC and data SOC approach is that controller parameters needed to be changed to accommodate the self-optimizing measurement combination. While augmenting extremum seeking control with self-optimizing control can improve transient performance, the experiments suggest that the standard ESC performs just as well for disturbances that are small in magnitude. Prior work in the HVAC field shows that standard ESC effectively handles slowly varying disturbances as well.

Finally, although the combined ESC and SOC approach using T_4 and T_{2-3} leads to transient performance improvement, it is not recommended as a suitable control strategy for all VCSs and operating scenarios. Instead, the case studies in this chapter demonstrated the following design approach:

- Determine significant process disturbances.
- Use either model based or data based nullspace SOC to choose a combination of measurements that are sufficient to reject the process disturbances.
- Use an ESC to change the SOC set point in order to compensate for unmodeled disturbances and set point errors.

The most important consideration is finding significant disturbances. In this chapter, the cooling capacity was the sole disturbance considered. In practice, a self-optimizing measurement combination meant to reject a cooling capacity disturbance might perform poorly for other disturbance changes. Analyzing a system model can help flag these issues.

CHAPTER 6

CONCLUSION

6.1 Summary of Research Contributions

This thesis covered several aspects of extremum seeking control design for vapor compression systems including physics based modeling and simulation; algorithms, tuning guidelines, and performance of control algorithms; and comparisons between simulation and experimental results of algorithm applications.

Chapter 2 reviewed the operational aspects of the four component vapor compression cycle, validated a system model against experimental data, and connected system operation to a steady state optimization problem. Of particular importance was the power consumption model, which captured the vapor compression system's performance index dynamics. Fast electromechanical dynamics could be ignored and it was easy to represent the total power consumption a summation of component subsystem power consumptions. Calibrations in simulation and experiment showed that power consumption could be approximated as a convex quadratic function.

Chapter 3 focused on overviews of two real-time optimization approaches: extremum seeking control and self-optimizing control. Their respective benefits and drawbacks could be seen from simulation case studies involving simple systems with quadratic performance indices. The most noteworthy takeaways were:

- Time varying extremum seeking algorithms generally outperform extremum seeking using LTI filters with demodulation.

- Gradient descent extremum seeking is favorable to Newton descent extremum seeking when information about the performance index's curvature is available.

- Newton descent extremum seeking is more sensitive to noise than gradient descent extremum seeking, but is not reliant on prior knowledge of the performance index's curvature.

- Combining self-optimizing control and extremum seeking control can reduce the optimal input's sensitivity to known, but unmeasured disturbances.

- The development process of obtaining optimal operation data, using it to choose a self-optimizing measurement combination, and redesigning an extremum seeking controller can improve the real-time optimizer's transient performance.

These results proved to be valuable in Chapters 4 and 5, where both simulation and experimental tests of these strategies applied to vapor compression systems supported many of the above assertions. Because the vapor compression performance indices considered in this thesis were quadratic, they share many properties with even the simple simulation examples from Chapter 3.

Chapters 4 and 5 had unique contributions of their own. Chapter 4's comparison of simulation and experimental gradient descent extremum seeking results showed that Chapter 2's system modeling effort could provide reliable information about the performance index's curvature. This is significant because many of the results in this thesis are difficult to predict without the help of a physics based model. Even though a model might not have exceptional predictive capability, there is a chance that it could provide helpful information for gradient descent extremum seeking design. Another contribution of Chapter 4 was a novel and definitive demonstration of time varying, Newton descent extremum seeking's ability to recover a desired gradient descent gain by estimating the inverse of the performance index's curvature. Finally, Chapter 5 took the novel step of applying the combined extremum seeking and data based self-optimizing control design process to the experimental test stand described in Chapter 2.

6.2 Future Work

Extremum seeking has significant potential as a practical tool for steady state vapor compression system optimization. However, its implementation could be limited by a lack of model knowledge and slow convergence rates. The lack of model knowledge could be addressed by devising ways to make the extremum seeking process more "model free". An example of this is how Newton descent extremum seeking could be used to implement a controller without prior knowledge of the performance index's curvature. The second concern could be addressed using alternative ways to represent the plant for extremum seeking control. An example of this can be found in [8], where the authors use an input affine nonlinear system (a special case of a nonlinear system) to represent the vapor compression system's dynamics. Such representations can lead to convergence rates that are significantly faster than those observed in this thesis.

Finally, this thesis focused on unconstrained real-time optimization problems. This focus could be relaxed to include constraints on the controller inputs and states. While penalty function additions to the cost function have been proposed to allow ESC to avoid loosing feedback after encountering a constraint [46], predictive algorithms are more adept at constraint handling. An interesting study could examine how ESC interacts with predictive control algorithms when the system is near its operating limits.

REFERENCES

[1] Y. Tan, W. Moase, C. Manzie, D. Nesic, and I. Mareels, "Extremum seeking from 1922 to 2010," in *29th Chinese Control Conference. Beijing, China, July 29-31*, 2010.

[2] G. Gelbert, J. P. Moeck, C. O. Paschereit, and R. King, "Advanced algorithms for gradient estimation in one- and two-parameter extremum seeking controllers," *Journal of Process Control*, vol. 22, no. 4, pp. 700–709, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.jprocont.2012.01.022

[3] M. Guay, "A time-varying extremum-seeking control approach for discrete-time systems," *Journal of Process Control*, vol. 24, no. 3, pp. 98–112, 2014.

[4] M. Krstić and H.-H. Wang, "Stability of extremum seeking feedback for general nonlinear dynamic systems," *Automatica*, vol. 36, no. 4, pp. 595–601, 2000.

[5] D. Nesic, Y. Tan, C. Manzie, A. Mohammadi, and W. Moase, "A unifying framework for analysis and design of extremum seeking controllers," *Control and Decision Conference*, pp. 4274–4285, 2012. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6244001

[6] A. Ghaffari, M. Krstić, and D. Nešić, "Multivariable Newton-based extremum seeking," *Automatica*, vol. 48, no. 8, pp. 1759–1767, 2012.

[7] M. Guay and D. Dochain, "A proportional integral extremum-seeking control approach," in *19th IFAC World Congress*, 2014.

[8] D. J. Burns, C. R. Laughman, and M. Guay, "Proportional Integral Extremum Seeking for Vapor Compression Systems," no. 2, 2016.

[9] H. Sane, C. Haugstetter, and S. Bortoff, "Building HVAC control systems - role of controls and optimization," in *Proceedings of the American Control Conference, Minneapolis, Minnesota, USA, June 14-16*, 2006.

[10] J. P. Koeln and A. G. Alleyne, "Optimal subcooling in vapor compression systems via extremum seeking control: Theory and experiments," *International Journal of Refrigeration*, vol. 43, pp. 14–25, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140700714000632

[11] M. Guay and D. J. Burns, "A comparison of extremum seeking algorithms applied to vapor compression system optimization," in *Proceedings of the 2014 American Control Conference, Portland, Oregon, USA, June 4th-6th*, 2014.

[12] B. Mu, Y. Li, J. E. Seem, and B. Hu, "A Multi-variable Newton-based Extremum Seeking Control for Condenser Water Loop Optimization of Chilled Water Plant," *Journal of Dynamic Systems, Measurement, and Control*, vol. 137, pp. 1–10, 2015.

[13] D. Burns and C. Laughman, "Extremum Seeking Control for Energy Optimization of Vapor Compression Systems," in *14th International Refrigeration and Air Conditioning Conference at Purdue, West Lafayette, IN, USA, July 16-19.*, 2012.

[14] L. Dong, Y. Li, B. Mu, and Y. Xiao, "Self-optimizing control of air-source heat pump with multivariable extremum seeking," *Applied Thermal Engineering*, vol. 84, pp. 180–195, 2015.

[15] X. Li, Y. Li, J. E. Seem, and P. Li, "Dynamic modeling and self-optimizing operation of chilled water systems using extremum seeking control," *Energy and Buildings*, vol. 58, pp. 172–182, mar 2013.

[16] W. K. Weiss, D. J. Burns, and M. Guay, "Realtime Optimization of MPC Setpoints using Time-Varying Extremum Seeking Control for Vapor Compression Machines," in *15th International Refrigeration and Air Conditioning Conference at Purdue, West Lafayette, Indiana, July 14-17*, 2014. [Online]. Available: http://docs.lib.purdue.edu/iracc/1424

[17] Y. Xiao, Y. Li, and J. E. Seem, "Multi-variable Extremum Seeking Control for Mini-split Air-conditioning System," in *15th International Refrigeration and Air Conditioning Conference at Purdue, West Lafayette, Indiana, July 14-17*, 2014.

[18] B. Mu, Y. Li, J. M. House, and T. I. Salsbury, "Experimental evaluation of anti-windup extremum seeking control for airside economizers," *Control Engineering Practice*, vol. 50, pp. 37–47, 2016. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0967066116300211

[19] Y. Li, M. A. Rotea, G. T. C. Chiu, L. G. Mongeau, and I. Paek, "Extremum Seeking Control of Tunable Thermoacoustic Cooler," *IEEE Transactions on Control Systems Technology*, vol. 13, no. 4, 2005.

[20] S. Skogestad, "Plantwide control: The search for the self-optimizing control structure," *Journal of Process Control*, vol. 10, no. 5, pp. 487–507, 2000.

[21] V. Alstad and S. Skogestad, "Null space method for selecting optimal measurement combinations as controlled variables," *Industrial and Engineering Chemistry Research*, vol. 46, no. 3, pp. 846–853, 2007.

[22] I. J. Halvorsen, S. Skogestad, J. C. Morud, and V. Alstad, "Optimal Selection of Controlled Variables," *Industrial & Engineering Chemistry Research*, vol. 42, pp. 3273–3284, 2003.

[23] J. Jäschke and S. Skogestad, "Controlled Variables from Optimal Operation Data," *Computer Aided Chemical Engineering*, vol. 29, pp. 754–757, 2011.

[24] J. Jaschke and S. Skogestad, "Using process data for finding self-optimizing controlled variables," in *Preprints of the 10th IFAC International Symposium on Dynamics and Control of Process Systems, Mumbai, India, December 18-20*, 2013.

[25] J. B. Jensen and S. Skogestad, "Optimal operation of simple refrigeration cycles Part I: Degrees of freedom and optimality of sub-cooling," *Computers & Chemical Engineering*, vol. 31, no. 5-6, pp. 712–721, 2007.

[26] V. Alstad, S. Skogestad, and E. S. Hori, "Optimal measurement combinations as controlled variables," *Journal of Process Control*, vol. 19, pp. 138–148, 2009.

[27] X. Yin, S. Li, and W. Cai, "Enhanced-efficiency operating variables selection for vapor compression refrigeration cycle system," *Computers & Chemical Engineering*, vol. 80, pp. 1–14, sep 2015.

[28] J. Jäschke and S. Skogestad, "NCO tracking and self-optimizing control in the context of real-time optimization," *Journal of Process Control*, vol. 21, no. 10, pp. 1407–1416, 2011.

[29] J. B. Jensen and S. Skogestad, "Optimal operation of simple refrigeration cycles Part II: Selection of controlled variables," *Computers & Chemical Engineering*, vol. 31, no. 12, pp. 1590–1601, dec 2007.

[30] Y. Li and J. E. Seem, "Extremum Seeking Control of Cooling Tower for Self-optimizing Efficient Operation of Chilled Water Systems," pp. 3396–3401, 2012.

[31] A. G. Alleyne and N. Jain, "Transient thermal systems: Dynamics and Control," *Mechanical Engineering*, vol. 136, no. 3, pp. 4–12, 2014.

[32] B. P. Rasmussen and A. G. Alleyne, "Dynamic Modeling and Advanced Control of Air Conditioning and Refrigeration Systems," Ph.D. dissertation, 2006.

[33] A. Alleyne, "THERMOSYS 4 Toolbox," *University of Illinois at Urbana-Champaign*, no. March, 2012. [Online]. Available: http://arg.mechse.illinois.edu/thermosys

[34] H. Pangborn, A. G. Alleyne, and N. Wu, "A comparison between finite volume and switched moving boundary approaches for dynamic vapor compression system modeling," *International Journal of Refrigeration*, vol. 53, pp. 101–114, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.ijrefrig.2015.01.009

[35] M. S. Elliott and B. P. Rasmussen, "Decentralized model predictive control of a multi-evaporator air conditioning system," *Control Engineering Practice*, vol. 21, no. 12, pp. 1665–1677, 2013.

[36] G. François, B. Srinivasan, and D. Bonvin, "Comparison of six implicit real-time optimization schemes," *Journal européen des systèmes automatisés*, 2013.

[37] M. Großbichler, R. Schmied, P. Polterauer, H. Waschl, and L. Re, "A robustified Newton based Extremum Seeking for Engine Optimization," pp. 3280–3285, 2016.

[38] A. Ghaffari, S. Seshagiri, and M. Krstić, "Multivariable maximum power point tracking for photovoltaic micro-converters using extremum seeking," *Control Engineering Practice*, vol. 35, pp. 83–91, 2015.

[39] B. Mu, Y. Li, T. I. Salsbury, and J. M. House, "Extremum Seeking Based Control Strategy for a Chilled-Water Plant With Parallel Chillers," in *Proceedings of the ASME 2015 Dynamic Systems and Control Conference*, 2015.

[40] L. Dong, Y. Li, T. I. Salsbury, and J. M. House, "Self-optimizing Control and Mode Switching for Multi-functional Variable Refrigerant Flow Air Conditioning Systems via Extremum Seeking," in *American Control Conference*, 2016.

[41] J. Ryan and J. Speyer, "Peak-seeking control using gradient and Hessian estimates," *American Control Conference (ACC), 2010*, pp. 611–616, 2010.

[42] K. Astrom and B. Wittenmark, *Adaptive Control*, 1st ed. Addison-Wesley, 1989.

[43] J. Deutscher and M. Bäuml, "Approximate feedback linearization using multivariable legendre polynomials," *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 17, no. 1 PART 1, pp. 4749–4754, 2008.

[44] S. Marinkov, B. De Jager, and M. Steinbuch, "Extremum Seeking Control with Adaptive Disturbance Feedforward," *19th IFAC World Congress*, pp. 3627–3632, 2014.

[45] D. J. Burns, W. K. Weiss, and M. Guay, "Realtime Setpoint Optimization with Time-Varying Extremum Seeking for Vapor Compression Systems," 2015.

[46] Y. Tan, Y. Li, and I. Mareels, "Extremum Seeking for Constrained Inputs," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2405–2410, 2013.

APPENDIX A

FEEDBACK CONTROL DESIGN PROCEDURE

The RTO algorithms implemented in Chapters 4 and 5 required a feedback control layer to regulate cooling capacities and superheat to reference values. The design of these regulators involved identification of the plant model using the `n4sid` black box method followed by discrete time linear quadratic regulator (LQR) or linear quadratic Gaussian (LQG) feedback control design. Before control design, the plant models were reformulated to penalize high actuator slew rates and steady state offset between reference and output. Section A.1 gives the standard procedure for subspace based system identification and Section A.2 gives the controller design and analysis procedure given the identified system model.

A.1 System Identification

The case studies in chapters 4 and 5 use black box state space system identification to elicit discrete time models that are useful for control design. Each controller implementation follows a standard system identification procedure:

1. **Inputs and Outputs** Identify the inputs that will be manipulated by the feedback controller and the outputs that will be measured for linear feedback control.

2. **Input Sequence** Apply a pseudorandom binary input sequence, Δu, with appropriate amplitude about a nominal input, $\bar{u}$. Large amplitude perturbations will give a high signal to noise ratio, but will result in poor linear model fits if the perturbations stray too far from nominal conditions.

3. **Data Centering and Scaling** Center the data for each output, $y_{data,i}$, about a mean value, $\bar{y}_i$, normalizing the output data by its maximum amplitude, $A_{y,i} = max(y_{data,i}) - min(y_{data,i})$, using (A.1). This gives $y_{id,i}$, which can be used for identification.

$$y_{id,i} = \frac{y_{data,i} - \overline{y}_i}{A_{y,i}} \tag{A.1}$$

4. **Model Order Selection** Apply the `n4sid` algorithm to the first half of the data and select the lowest model order that results in an acceptable normalized root mean square error for each output, $NRMSE_i$, from (A.2).

$$NRMSE_i = 100(1 - \frac{\|y_{id,i} - \hat{y}_i\|}{\|y_{id,i}\|}) \tag{A.2}$$

5. **Cross Validation** Calculate the $NRMSE$ for the model given the second half of the data to protect against the inclusion of weakly controllable or observable dynamics in the system model. If this step does not yield satisfactory results, then repeat the model order selection.

The identification procedure above produces a discrete LTI state space matrix triple, (A, B, C), with sample time T_s. D is not included in the state space formulation because it is typically zero for vapor compression systems. For cases where the number of outputs is equal to the number of states, C can be used as a similarity transformation as in (A.3) to allow full state measurement with the new matrix triple, $(\tilde{A}, \tilde{B}, I)$.

$$\begin{aligned} &\tilde{A}_D = C_D A_D C_D^{-1}, \ \tilde{B} = C_D B_D \\ &\tilde{x}_{k+1} = \tilde{A}_D x_k + \tilde{B}_D u_k \\ &y_k = \tilde{x}_k \end{aligned} \tag{A.3}$$

A.2 Linear Quadratic Controller Design

The linear quadratic controller was selected because it is simple to implement and it considers coupling between system inputs and outputs. The controller design procedure employed the state space triple, (A, B, C), from the previous section to design a state feedback regulator with knowledge of the system rate limits and with zero steady state sensitivity to changes in the reference input. Eq. (A.4) shows the state space system and cost function resulting from the addition of integral and delayed input states, η_k and u_{k-1}, to generic (A,B,C) system matrices. The modified input is defined as the difference between the current input and the delayed input, $\Delta u_k = u_k - u_{k-1}$. The following two sections give the dynamic controllers used in each case.

$$\begin{aligned}
&\min_{\Delta u_k} \quad J_{LQR}(x_0) = \sum_{k=0}^{\infty} \begin{bmatrix} x_k \\ \eta_k \\ u_{k-1} \end{bmatrix}^T \begin{bmatrix} C^TC & 0 & 0 \\ 0 & \rho_\eta I & 0 \\ 0 & 0 & \rho_u I \end{bmatrix} \begin{bmatrix} x_k \\ \eta_k \\ u_{k-1} \end{bmatrix} + \Delta u_k^T \rho_{\Delta u} I \Delta u_k \\
&\text{s.t.} \quad \begin{bmatrix} x_{k+1} \\ \eta_{k+1} \\ u_k \end{bmatrix} = \begin{bmatrix} A & 0 & B \\ C & 0 & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} x_k \\ \eta_k \\ u_{k-1} \end{bmatrix} + \begin{bmatrix} B \\ 0 \\ I \end{bmatrix} \Delta u_k
\end{aligned} \tag{A.4}$$

Controller gains from solving the discrete time optimal control problem are represented by the partitioned matrix, $[K_x\ K_\eta\ K_u]$, where K_x, K_η, and K_u are the system state, integral, and delayed input gain matrices respectively. These gains are adjusted using the tuning weights, ρ_η, ρ_u, and $\rho_{\Delta u}$. For the case where a direct measurement of x_k is unavailable, a discrete time, static gain Kalman filter was designed using covariance matrices, $\gamma_n I$ and $\gamma_x I$. The sole step in designing the LQR/LQG regulators is to adjust the tuning weights until satisfactory closed loop performance is achieved.

For implementation of extremum seeking control, knowledge of the system's settling time was desired to ensure time scale separation between the extremum seeking controller and the closed loop plant. To achieve this, the controllers were first expressed as dynamic systems. Incidentally, these forms are convenient for implementation of controllers in simulation and experiment.

For the general case where a LQG controller is used, (A.5) describes the controller dynamics. For the special case of a plant model with a full state measurement available, (A.6) gives the controller dynamics. Below, r_k represents a time varying reference input, y_k represents the current output measurement, and L is the observer gain obtained from the Kalman filter design.

$$\begin{aligned}
\begin{bmatrix} \hat{x}_{k+1} \\ \eta_{k+1} \\ u_k \end{bmatrix} &= \begin{bmatrix} (A - BK_x - LC) & -BK_\eta & B(I - K_u) \\ 0 & I & 0 \\ -K_x & -K_\eta & (I - K_u) \end{bmatrix} \begin{bmatrix} \hat{x}_k \\ \eta_k \\ u_{k-1} \end{bmatrix} \\
&+ \begin{bmatrix} L & 0 \\ -T_s I & T_s I \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y_k \\ r_k \end{bmatrix} \\
u_k &= \begin{bmatrix} -K_x & -K_\eta & (I - K_u) \end{bmatrix} \begin{bmatrix} \hat{x}_k \\ \eta_k \\ u_{k-1} \end{bmatrix} + \begin{bmatrix} 0 & 0 \end{bmatrix} \begin{bmatrix} y_k \\ r_k \end{bmatrix}
\end{aligned} \tag{A.5}$$

$$\begin{aligned}
\begin{bmatrix} \eta_{k+1} \\ u_k \end{bmatrix} &= \begin{bmatrix} I & 0 \\ -K_\eta & (I - K_u) \end{bmatrix} \begin{bmatrix} \eta_k \\ u_{k-1} \end{bmatrix} + \begin{bmatrix} -T_s I & T_s I \\ -K_x & 0 \end{bmatrix} \begin{bmatrix} y_k \\ r_k \end{bmatrix} \\
u_k &= \begin{bmatrix} -K_\eta & (I - K_u) \end{bmatrix} \begin{bmatrix} \eta_k \\ u_{k-1} \end{bmatrix} - \begin{bmatrix} K_x & 0 \end{bmatrix} \begin{bmatrix} y_k \\ r_k \end{bmatrix}
\end{aligned} \tag{A.6}$$

Given that the plant's matrices are known and part of the controller dynamics, closed loop equations can be written where the sole input is r_k. Eq. (A.7) represents the closed loop dynamics for the LQG controllers, while (A.8) represents the closed loop dynamics for the LQR.

$$
\begin{aligned}
\begin{bmatrix} x_{k+1} \\ \hat{x}_{k+1} \\ \eta_{k+1} \\ u_k \end{bmatrix} &= \begin{bmatrix} A & -BK_x & -BK_\eta & B(I-K_u) \\ LC & (A-BK_x-LC) & -BK_\eta & B(I-K_u) \\ -T_sC & 0 & I & 0 \\ 0 & -K_x & -K_\eta & (I-K_u) \end{bmatrix} \begin{bmatrix} x_k \\ \hat{x}_k \\ \eta_k \\ u_{k-1} \end{bmatrix} \\
&+ \begin{bmatrix} 0 \\ 0 \\ T_sI \\ 0 \end{bmatrix} r_k \\
y_k &= \begin{bmatrix} C & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ \hat{x}_k \\ \eta_k \\ u_{k-1} \end{bmatrix} + 0r_k
\end{aligned}
\tag{A.7}
$$

$$
\begin{aligned}
\begin{bmatrix} x_{k+1} \\ \eta_{k+1} \\ u_k \end{bmatrix} &= \begin{bmatrix} (A-BK_x) & -BK_\eta & B(I-K_u) \\ 0 & I & 0 \\ -K_x & -K_\eta & (I-K_u) \end{bmatrix} \begin{bmatrix} x_k \\ \eta_k \\ u_{k-1} \end{bmatrix} + \begin{bmatrix} 0 \\ T_sI \\ 0 \end{bmatrix} r_k \\
y_k &= \begin{bmatrix} I & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ \eta_k \\ u_{k-1} \end{bmatrix} + 0r_k
\end{aligned}
\tag{A.8}
$$

Assembling the state space representations of these systems in MATLAB and using the `stepinfo` command gives the settling times from the input to output step responses. The longest input to output step response settling time, τ_{CL}, is the closed loop plant's representative time scale.

APPENDIX B

SIMULATION DETAILS

B.1 System Identification

Table B.1: Nominal input values employed in the simulation case studies.

Input	Nominal Value [%]
u_{kp}	55
u_{cf}	65
u_v	40
u_{ef}	100

B.1.1 Standard ESC

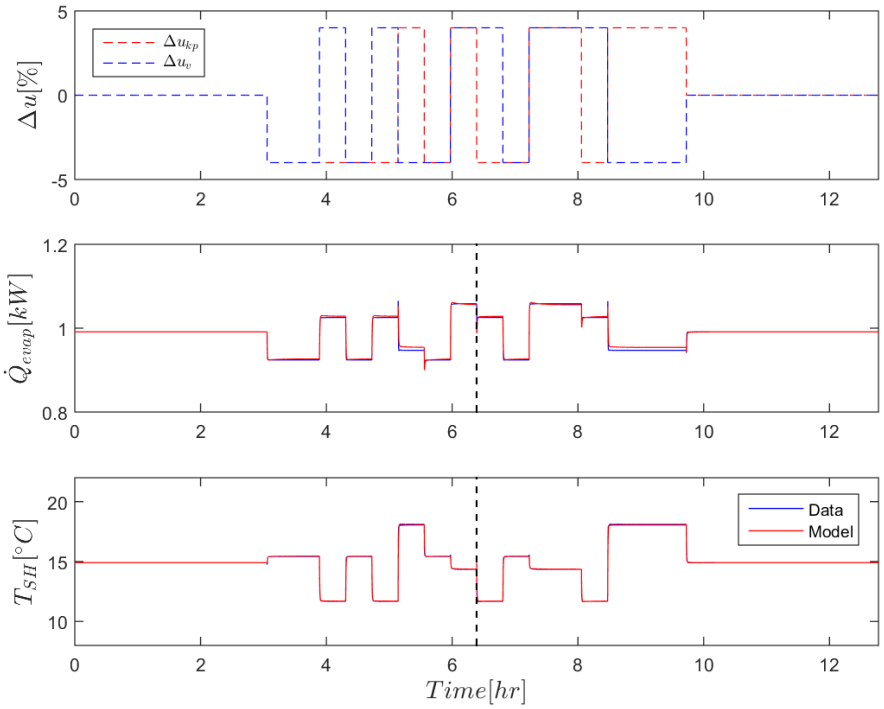

Figure B.1: Pseudorandom binary inputs of 4% magnitude were applied to the u_{kp} and u_v inputs to identify a two input two output system model for $\dot{Q}_{evap}$ and T_{SH}.The relationship between the u_{kp} and u_v inputs and $\dot{Q}_{evap}$ and T_{SH} outputs in simulation can be approximated by a fourth order linear system model.

Table B.2: System identification results from simulation for the standard ESC's linear model.

Output	$\dot{Q}_{evap}$	T_{SH}
$\overline{y}$	$0.9912[kW]$	$14.91[°C]$
A_y	$0.0829[kW]$	$3.2578[°C]$
$NRMSE$	$90[\%]$	$98[\%]$

$$A_C = \begin{bmatrix} 0.8341 & 0.0235 & 0.0292 \\ -0.2683 & 0.7080 & -0.2498 \\ -0.1419 & -0.2729 & 0.7479 \end{bmatrix}$$

$$B_C = \begin{bmatrix} 0.0005 & 0.0060 \\ -0.0256 & 0.0048 \\ -0.0235 & 0.0005 \end{bmatrix}$$

$$C_C = \begin{bmatrix} 4.9584 & -2.1864 & 0.1411 \\ -4.3714 & -1.1025 & -0.2238 \end{bmatrix}$$

$$Sample\ Time : 4s$$

B.1.2 Combined ESC and Local SOC

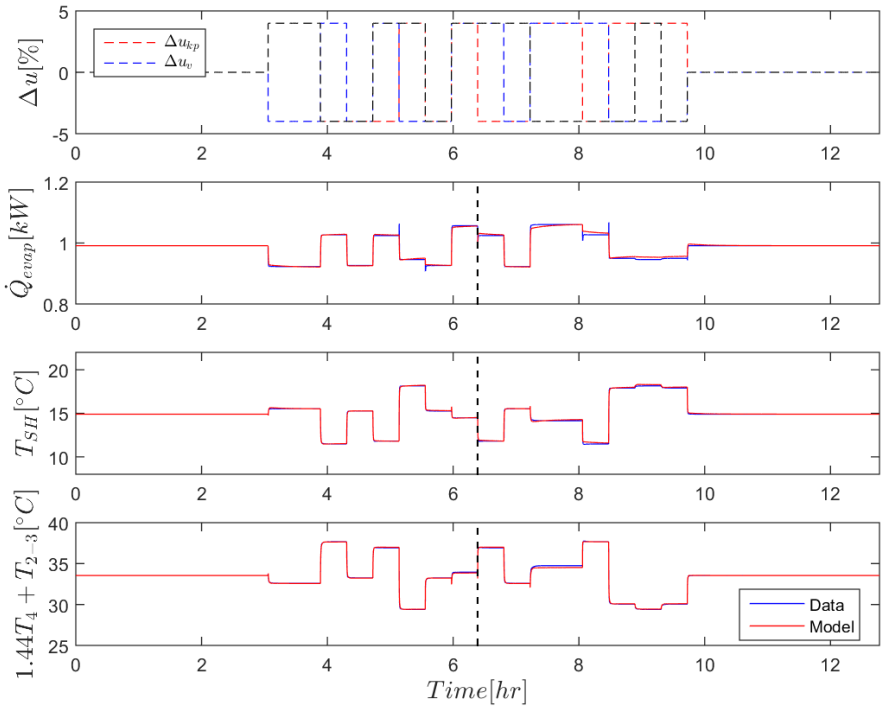

Figure B.2: The relationship between the u_{kp}, u_{cf}, and u_v inputs and $\dot{Q}_{evap}$, T_{SH}, and $1.44T_4 + T_{2-3}$ outputs can be approximated by a fourth order linear system model.

Table B.3: System identification results for the combined ESC and SOC's linear model.

Output	$\dot{Q}_{evap}$	T_{SH}	$1.7T_4 + T_{2-3}$
$\overline{y}$	$0.9912[kW]$	$14.9054[^\circ C]$	$33.5424[^\circ C]$
A_y	$0.0838[kW]$	$3.4732[^\circ C]$	$4.2208[^\circ C]$
$NRMSE$	$88.22[\%]$	$95.12[\%]$	$95.71[\%]$

$$A_C = \begin{bmatrix} 0.8195 & 0.1591 & -0.0641 & -0.0721 \\ 0.0906 & 0.8935 & -0.0311 & 0.0064 \\ 0.0580 & 0.1672 & 0.6832 & -0.4343 \\ -0.0145 & 0.2411 & -0.2097 & 0.6253 \end{bmatrix}$$

$$B_C = \begin{bmatrix} -0.0071 & 0.0084 & -0.0005 \\ 0.0000 & -0.0041 & -0.0003 \\ -0.0268 & 0.0050 & -0.0014 \\ -0.0212 & 0.0073 & -0.0007 \end{bmatrix}$$

$$C_C = \begin{bmatrix} 4.7643 & -1.4510 & -1.5172 & -0.7570 \\ -3.2599 & 2.5076 & -1.3753 & 0.1346 \\ 3.7827 & -0.9994 & 1.4728 & -0.3746 \end{bmatrix}$$

$$Sample\ Time : 4s$$

B.1.3 Combined ESC and Data SOC

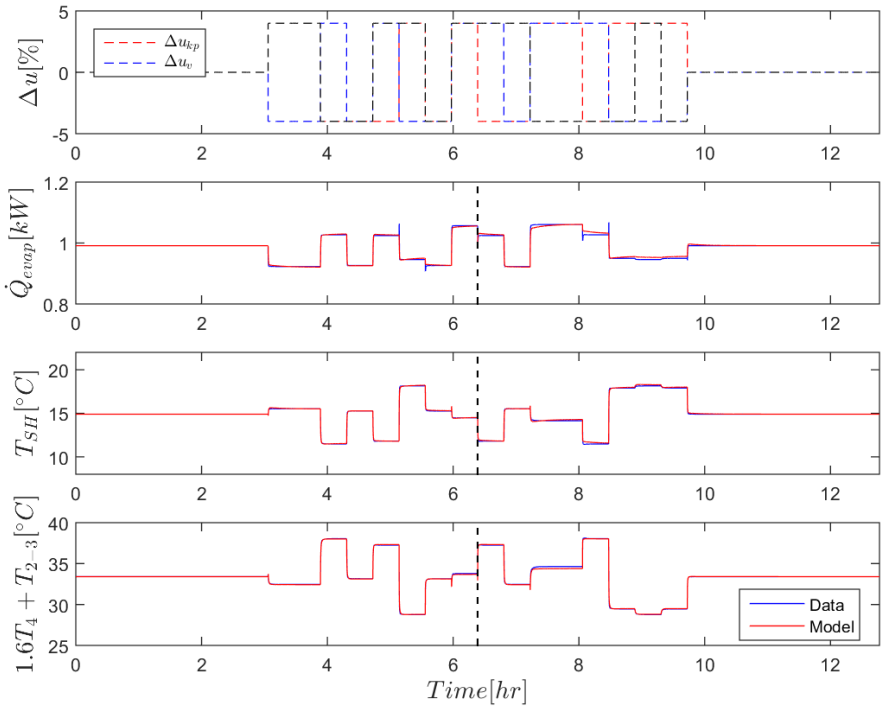

Figure B.3: The relationship between the u_{kp}, u_{cf}, and u_v inputs and $\dot{Q}_{evap}$, T_{SH}, and $1.6T_4 + T_{2-3}$ outputs can be approximated by a fourth order linear system model.

Table B.4: System identification results for the combined ESC and data SOC's linear model.

Output	$\dot{Q}_{evap}$	T_{SH}	$1.7T_4 + T_{2-3}$
$\overline{y}$	$0.9912[kW]$	$14.9054[^\circ C]$	$33.5424[^\circ C]$
A_y	$0.0838[kW]$	$3.4732[^\circ C]$	$4.2208[^\circ C]$
$NRMSE$	$88.44[\%]$	$95.08[\%]$	$95.93[\%]$

$$A_C = \begin{bmatrix} 0.8216 & 0.1512 & -0.0676 & 0.0799 \\ 0.0822 & 0.9086 & -0.0334 & -0.0070 \\ 0.0669 & 0.1228 & 0.6412 & 0.4386 \\ 0.0168 & -0.1943 & 0.2102 & 0.6475 \end{bmatrix}$$

$$B_C = \begin{bmatrix} -0.00690.0084 - 0.0005 \\ -0.0005 - 0.0036 - 0.0003 \\ -0.02700.0050 - 0.0015 \\ 0.0189 - 0.00700.0006 \end{bmatrix}$$

$$C_C = \begin{bmatrix} 4.8226 & -1.3515 & -1.4224 & 0.7039 \\ -3.2663 & 2.4442 & -1.4385 & -0.1034 \\ 3.6744 & -1.0136 & 1.5785 & 0.3797 \end{bmatrix}$$

$$Sample\ Time : 4s$$

B.2 Controller Design

B.2.1 Standard ESC

Table B.5: Quadratic weights, feedback gains, and closed loop poles for the standard ESC's LQR.

Tuning Weights
$\rho_\eta = 0.3$
$\rho_u = 0$
$\rho_{\Delta u} = 50$
$\gamma_n = 1$
$\gamma_x = 1$
Controller Gains
$K_x = \begin{bmatrix} 2.3832 & -3.1352 & 2.1819 \\ 8.5376 & -1.0985 & 1.6273 \end{bmatrix}$
$K_\eta = \begin{bmatrix} -0.0485 & -0.0476 \\ -0.0466 & 0.0464 \end{bmatrix}$
$K_u = \begin{bmatrix} 0.2300 & -0.0085 \\ -0.0085 & 0.2794 \end{bmatrix}$
$L = 0.001 \begin{bmatrix} 0.3381 & -0.5349 \\ -0.8205 & -0.5713 \\ -0.9632 & -0.1783 \end{bmatrix}$
Closed Loop Poles
$eig(A_{cl}) = \{0.4963, 0.4980, 0.8547 \pm 0.1583i,$ $0.7533, 0.7946, 0.8748 \pm 0.1170,$ $0.9928, 0.9933\}$
$\tau_{CL} : 240s$

B.2.2 Combined ESC and Local SOC

Table B.6: Quadratic weights, feedback gains, and closed loop poles for the combined ESC and local SOC's LQG regulator.

Tuning Weights
$\rho_\eta = 0.3$
$\rho_u = 0$
$\rho_{\Delta u} = 50$
$\gamma_n = 1$
$\gamma_x = 1$
Controller Gains
$K_x = \begin{bmatrix} -1.4905 & 1.5746 & -3.2647 & 3.0446 \\ 7.0772 & -0.5909 & -0.8367 & -0.8335 \\ -3.5163 & -7.4783 & 2.4538 & -2.0016 \end{bmatrix}$
$K_u = \begin{bmatrix} 0.2388 & -0.0346 & 0.0125 \\ -0.0346 & 0.2900 & -0.0058 \\ 0.0125 & -0.0058 & 0.0567 \end{bmatrix}$
$K_\eta = \begin{bmatrix} -0.0386 & -0.0414 & 0.0370 \\ -0.0553 & 0.0260 & -0.0229 \\ 0.0004 & 0.0509 & 0.0554 \end{bmatrix}$
$L = 0.001 \begin{bmatrix} 0.3557 & -0.5894 & 0.5623 \\ -0.2266 & 0.2305 & -0.2109 \\ -0.3381 & -0.5528 & 0.5751 \\ -0.0994 & -0.6039 & 0.6057 \end{bmatrix}$
Closed Loop Poles
$eig(A_{cl}) = \{0.3528, 0.3561, 0.8361 \pm 0.1667i,$ $0.7186, 0.7465, 0.8572 \pm 0.1144i, 0.9118, 0.9172$ $0.9734 \pm 0.0286i, 0.9962, 0.9931\}$
$\tau_{CL} : 660s$

B.2.3 Combined ESC and Data SOC

Table B.7: Quadratic weights, feedback gains, and closed loop poles for the combined ESC and data SOC's LQG regulator.

Tuning Weights
$\rho_\eta = 0.3$
$\rho_u = 0$
$\rho_{\Delta u} = 50$
$\gamma_n = 1$
$\gamma_x = 1$
Controller Gains
$K_x = \begin{bmatrix} -1.4713 & 2.0612 & -3.0498 & -3.0643 \\ 7.2543 & -0.3621 & -0.6722 & 1.1133 \\ -3.3979 & -8.2431 & 2.2930 & 1.9299 \end{bmatrix}$
$K_u = \begin{bmatrix} 0.2388 & -0.0349 & 0.0128 \\ -0.0349 & 0.2887 & -0.0052 \\ 0.0128 & -0.0052 & 0.0531 \end{bmatrix}$
$K_\eta = \begin{bmatrix} -0.0379 & -0.0410 & 0.0381 \\ -0.0558 & 0.0258 & -0.0221 \\ 0.0015 & 0.0515 & 0.0550 \end{bmatrix}$
$L = 0.001 \begin{bmatrix} 0.3799 & -0.5749 & 0.5479 \\ -0.2144 & 0.1839 & -0.1652 \\ -0.2884 & -0.5829 & 0.6138 \\ 0.0151 & 0.5676 & -0.5718 \end{bmatrix}$
Closed Loop Poles
$eig(A_{cl}) = \{0.3543, 0.3586, 0.8369 \pm 0.1660i,$ $0.7220, 0.7505, 0.8767 \pm 0.1144i, 0.9119, 0.9191$ $0.9757 \pm 0.0261, 0.9954, 0.9910\}$
$\tau_{CL} : 716s$

APPENDIX C

EXPERIMENT DETAILS

C.1 System Identification

Table C.1: Nominal input values employed for every test.

Input	Nominal Value [%]
u_{kp}	52
u_{cf}	65
u_v	52
u_{ef}	100

C.1.1 Standard ESC

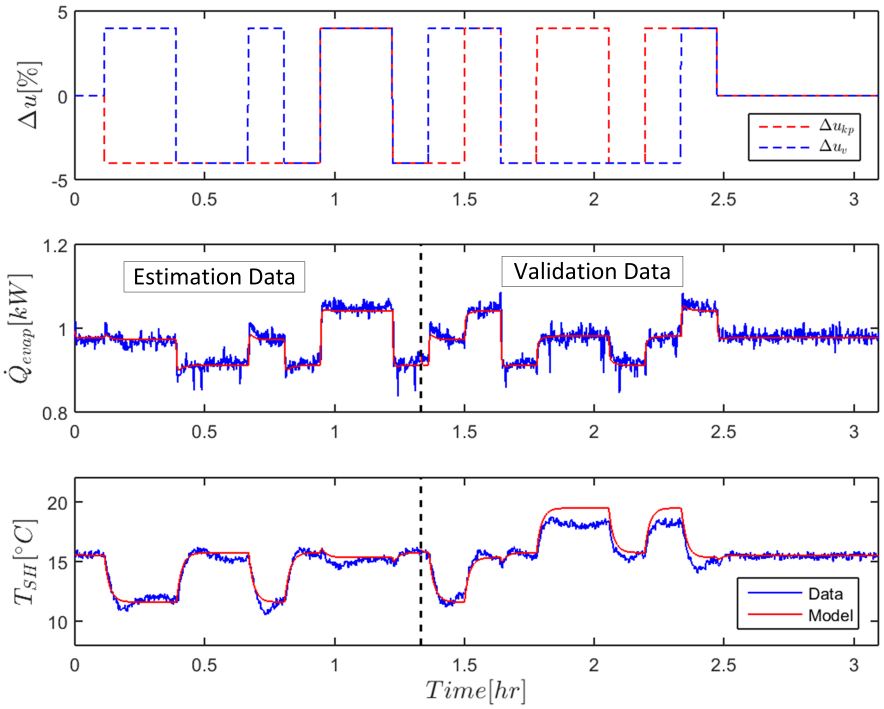

Figure C.1: Pseudorandom binary inputs of 4% magnitude were applied to the u_{kp} and u_v inputs to identify a two input two output system model for $\dot{Q}_{evap}$ and T_{SH}.The relationship between the u_{kp} and u_v inputs and $\dot{Q}_{evap}$ and T_{SH} outputs can be approximated by a second order linear system model.

Table C.2: System identification results for the standard ESC's linear model.

Output	$\dot{Q}_{evap}$	T_{SH}
$\overline{y}$	$0.9765[kW]$	$15.54[°C]$
A_y	$0.1239[kW]$	$4.0859[°C]$
$NRMSE$	$65[\%]$	$57[\%]$

$$\tilde{A}_D = \begin{bmatrix} 0.6330 & 0.0477 \\ -0.0448 & 0.9511 \end{bmatrix}$$

$$\tilde{B}_D = \begin{bmatrix} 0.0203 & 0.0286 \\ -0.0087 & -0.0034 \end{bmatrix}$$

$$\tilde{C}_D = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$Sample\ Time : 4s$$

C.1.2 Combined ESC and SOC

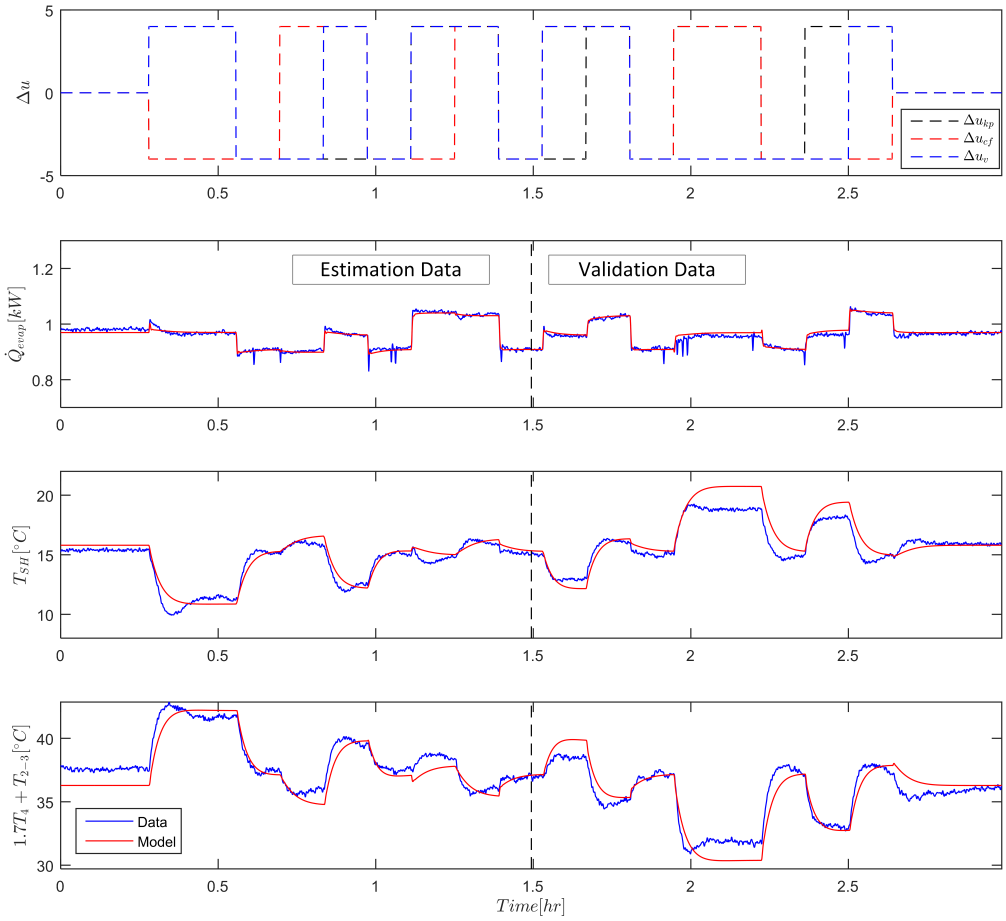

Figure C.2: The relationship between the u_{kp}, u_{cf}, and u_v inputs and $\dot{Q}_{evap}$, T_{SH}, and $1.7T_4 + T_{2-3}$ outputs can be approximated by a fifth order linear system model.

Table C.3: System identification results for the combined ESC and SOC's linear model.

Output	$\dot{Q}_{evap}$	T_{SH}	$1.7T_4 + T_{2-3}$
$\overline{y}$	$0.9695[kW]$	$15.7986[^\circ C]$	$36.2852[^\circ C]$
A_y	$0.1407[kW]$	$6.2970[^\circ C]$	$100[^\circ C]$
$NRMSE$	$62[\%]$	$33[\%]$	$65[\%]$

$$A_C = \begin{bmatrix} 0.9688 & 0.0130 & 0.0066 & 0.0433 & 0.0092 \\ -0.0523 & 0.8934 & 0.0776 & -0.1747 & 0.1124 \\ -0.0291 & -0.0092 & 0.9606 & -0.6589 & -0.0152 \\ -0.1721 & -0.2963 & 0.6283 & 0.1791 & 0.4550 \\ 0.1325 & 0.2831 & -0.1404 & -0.0849 & 0.5565 \end{bmatrix}$$

$$B_C = \begin{bmatrix} 0.0004 & 0.0000 & -0.0002 \\ -0.0008 & 0.0000 & 0.0004 \\ 0.0027 & 0.0007 & -0.0031 \\ 0.0016 & -0.0009 & 0.0085 \\ 0.0079 & 0.0014 & -0.0018 \end{bmatrix}$$

$$C_C = \begin{bmatrix} 0.1976 & -2.1482 & -2.7799 & 0.7015 & 0.1388 \\ 5.0597 & -0.8184 & 0.3534 & 0.1299 & -0.1588 \\ -5.5462 & -1.8548 & 0.3655 & 0.1016 & -0.0897 \end{bmatrix}$$

$$Sample\ Time : 4s$$

C.2 Controller Design

C.2.1 Standard ESC

Table C.4: Quadratic weights, feedback gains, and closed loop poles for the standard ESC's LQR.

Tuning Weights
$\rho_\eta = 0.3$
$\rho_u = 1$
$\rho_{\Delta u} = 250$
Controller Gains
$K_x = \begin{bmatrix} 0.0904 & 1.2357 \\ 0.3565 & -0.9069 \end{bmatrix}$
$K_\eta = \begin{bmatrix} 0.1620 & -0.0057 \\ -0.0057 & 0.1657 \end{bmatrix}$
$K_u = \begin{bmatrix} -0.0211 & -0.0237 \\ -0.0238 & 0.0209 \end{bmatrix}$
Closed Loop Poles
$eig(A_{cl}) = \{0.6393, 0.8865, 0.9151 \pm 0.0711i,$ $0.9372 \pm 0.0792i\}$
$\tau_{CL} : 392s$

C.2.2 Combined ESC and SOC

Table C.5: Quadratic weights, feedback gains, and closed loop poles for the combined ESC and SOC's LQG regulator.

Tuning Weights
$\rho_\eta = 0.3$
$\rho_u = 0$
$\rho_{\Delta u} = 50$
$\gamma_n = 1$
$\gamma_x = 1$
Controller Gains
$K_x = \begin{bmatrix} 23.2013 & 0.0817 & 0.5597 & 0.8104 & 1.2446 \\ 3.5084 & 13.8154 & -2.3205 & -1.1815 & 2.4636 \\ -14.1495 & -6.4066 & -1.6824 & 2.0347 & 0.4410 \end{bmatrix}$
$K_u = \begin{bmatrix} 0.1956 & 0.0139 & -0.0006 \\ 0.0139 & 0.0749 & -0.0196 \\ -0.0006 & -0.0196 & 0.1995 \end{bmatrix}$
$K_\eta = \begin{bmatrix} -0.0390 & -0.0455 & 0.0350 \\ 0.0018 & 0.0456 & 0.0589 \\ -0.0573 & 0.0326 & -0.0214 \end{bmatrix}$
$L = 0.001 \begin{bmatrix} 0.0309 & 0.0600 & -0.0656 \\ -0.1564 & -0.0099 & 0.0188 \\ -0.5498 & -0.0337 & 0.0743 \\ -0.4076 & 0.0190 & 0.0256 \\ 0.0546 & 0.0387 & -0.0363 \end{bmatrix}$
Closed Loop Poles
$eig(A_{cl}) = \{0.4663 \pm 0.5170i, 0.4669 \pm 0.5172i,$ $0.6820, 0.6835, 0.8895 \pm 0.1033i, 0.9341 \pm 0.0994i,$ $0.8817, 0.9737 \pm 0.0384,$ $0.9700 \pm 0.0108i, 0.9487\}$
$\tau_{CL} : 604s$

C.3 System Calibration

The system was swept over the range of u_{cf} and $\dot{Q}_{evap,ref}$ values depicted in Figure C.3 to map the system's power consumption and optimal inputs. The calibration results were used as a benchmark to evaluate the RTO performance of the candidate controllers. During the calibration, the linear quadratic regulator from the standard ESC approach was in the loop to ensure that the $\dot{Q}_{evap,ref}$ and the $T_{SH,ref}$ values were satisfied. Figure C.3 gives a time history of the calibration. The u_{cf} input was rate limited stepped by 5% increments over $10min$ intervals. Assuming that it took the system $7.5min$ to reach steady state, the power consumption value, $\dot{W}_{sys}$ value was averaged over the last $2.5min$ of the step to determine the steady state power consumption. For each $\dot{Q}_{evap,ref}$, the two quadratic least squares regressions given by (C.1) and (C.2) were performed for $\dot{W}_{sys}$. Table C.6 gives the coefficients of the least squares regressions as well as the calibrated minimum values.

$$\dot{W}_{sys} = c_1 + c_2 u_{cf} + c_3 u_{cf}^2 \tag{C.1}$$

$$\dot{W}_{sys} = d_1 + d_2(1.7T_4 + T_{2-3}) + d_3(1.7T_4 + T_{2-3})^2 \tag{C.2}$$

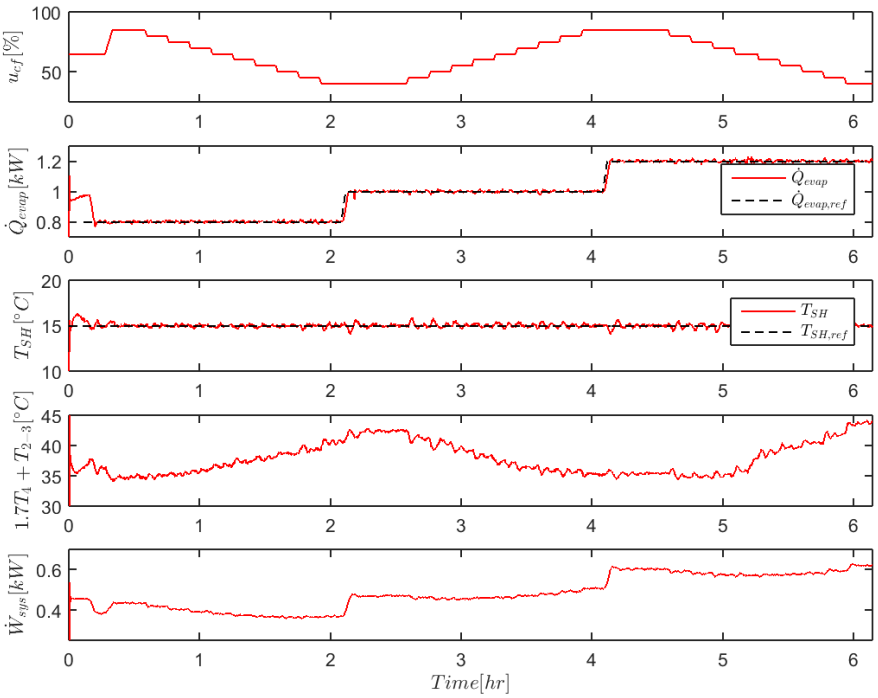

Figure C.3: Calibration inputs and outputs.

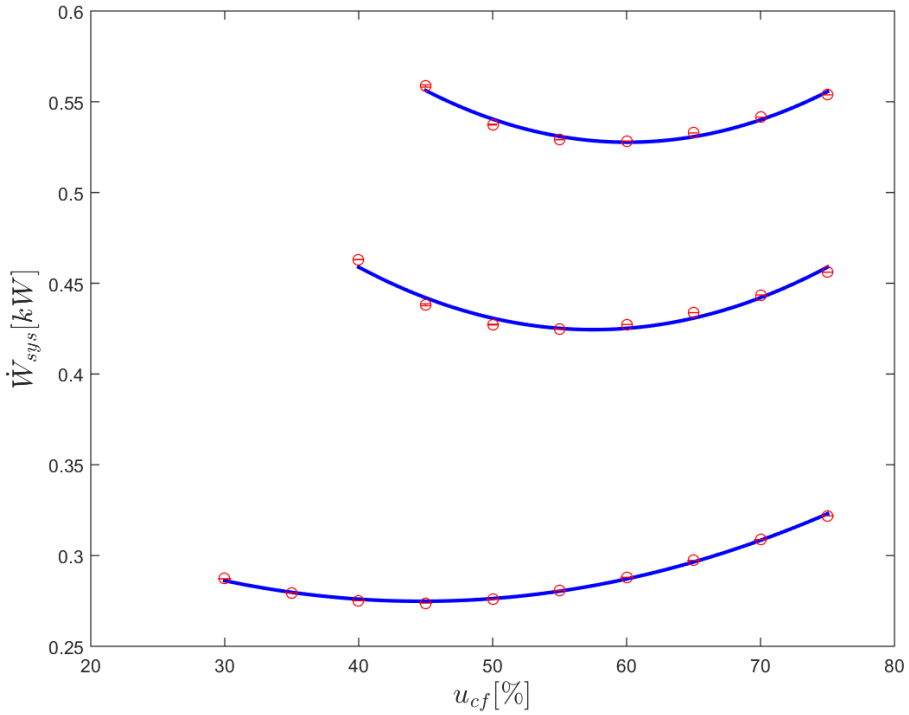

Figure C.4: Calibration of power consumption as a function of $\dot{Q}_{evap}$ and u_{cf}.

Table C.6: Regression coefficients and resulting minimum and optimal values.

$\dot{Q}_{evap,ref}[kW]$	c_1	c_2	c_3	$u^*_{cf}[\%]$	$\dot{W}_{sys,min}[kW]$
0.8	0.4718	-0.00456	4.85e-5	47	0.3646
1.0	0.6255	-0.00616	5.60e-5	55	0.4562
1.2	0.8856	-0.00985	7.70e-5	64	0.5705

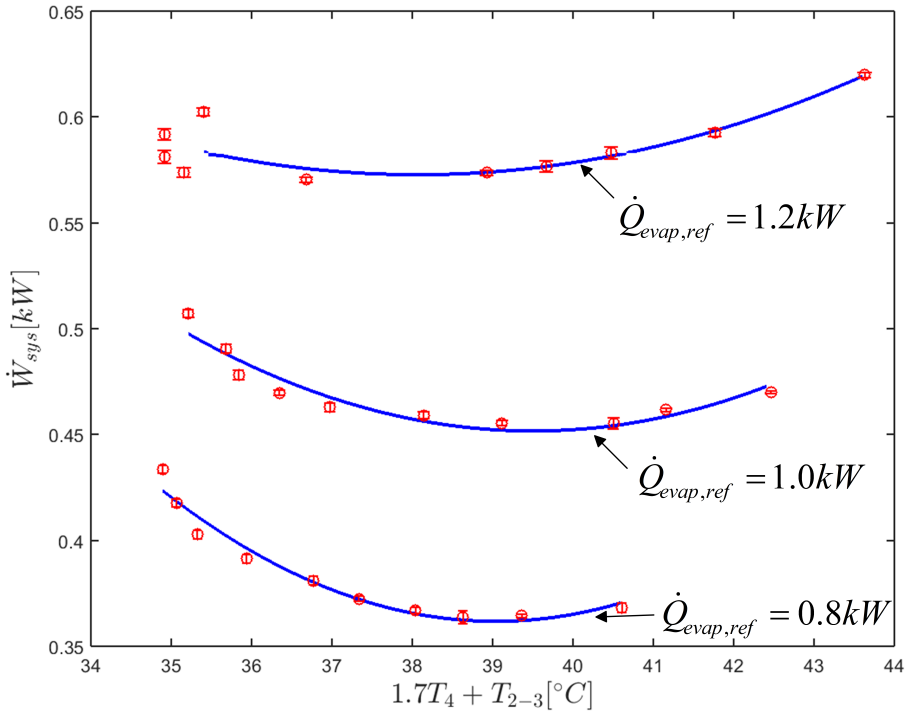

Figure C.5: Calibration of power consumption as a function of $\dot{Q}_{evap}$ and $1.7T_4 + T_{2-3}$.

Table C.7: Regression coefficients and resulting minimum and optimal values.

$\dot{Q}_{evap,ref}[kW]$	d_1	d_2	d_3	$(1.7T_4 + T_{2-3})^*[°C]$	$\dot{W}_{sys,min}[kW]$
0.8	5.8361	-0.2805	0.0036	39.0	0.3618
1.0	4.3424	-0.1970	0.0025	39.5	0.4517
1.2	2.7695	-0.1154	0.0015	38.1	0.5736