8-1-2014

# Development of a Black-Box Transient Thermal Model for Residential Buildings

Andrew Cross
*University of Nevada, Las Vegas*, andrewgcross@gmail.com

# DEVELOPMENT OF A BLACK-BOX TRANSIENT THERMAL MODEL

# FOR RESIDENTIAL BUILDINGS

By

Andrew G. Cross

Bachelor of Science – Mechanical Engineering

University of Kansas

2009

A thesis submitted in partial fulfillment of the requirements for the

Master of Science in Engineering – Mechanical Engineering

Department of Mechanical Engineering

Howard R. Hughes College of Engineering

The Graduate College

University of Nevada Las Vegas

August 2014

THE GRADUATE COLLEGE

We recommend the thesis prepared under our supervision by

**Andrew G. Cross**

entitled

**Development of a Black-Box Transient Thermal Model for Residential Buildings**

is approved in partial fulfillment of the requirements for the degree of

**Master of Science in Engineering - Mechanical Engineering**

**Department of Mechanical Engineering**

Robert Boehm, Ph.D., Committee Chair

Hsuan-Tsung Hsieh, Ph.D., Committee Member

Joon Soo Lee, Ph.D., Committee Member

Wolfgang Bein, Ph.D., Graduate College Representative

Kathryn Hausbeck Korgan, Ph.D., Interim Dean of the Graduate College

**August 2014**

ABSTRACT

**Development of a Black-Box Transient Thermal Model for Residential Buildings**

by

Andrew Cross

Dr. Robert Boehm, Examination Committee Chair
Distinguished Professor of Mechanical Engineering
University of Nevada, Las Vegas

Heavily populated metropolitan areas located in cooling-dominated climates, as are found in the Desert Southwest, pose a challenge to electrical utilities that service these areas. During the late afternoons of the summer months, residents of these metropolitan areas require larger than normal amounts of power to run their homes' air conditioning systems, at significant expense to the utilities. In the study reported here, interior temperature and power consumption data, accumulated over the course of a year and a half from seven houses within a Las Vegas neighborhood, are used to develop a predictive black-box statistical model for residential thermal transience. The model is able to predict when a collection of homes' air conditioners will either cycle on or off based on multiple measured inputs. When used in conjunction with a series of residential thermostats located in roughly the same area, the model can be used as a predictive controller to manipulate those homes' thermostats' setpoints in an attempt to level the homes' electrical demand by preventing the air conditioners from all running simultaneously, and alleviate utility expenses associated with producing power during peak demand periods.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

# Chapter 1 – Introduction

## 1.1 – Background

The Southwest region of the United States is a particularly hot and arid corner of the world. Vast swaths of Nevada, Arizona, New Mexico, Utah, and portions of California are covered by deserts that, according to the National Oceanic and Atmospheric Administration, are observed to be both the warmest and most solar irradiated areas of the country [1]. Despite these conditions, some of the United States' largest metropolitan areas are located within this region. The greater Las Vegas, Nevada area boasts nearly two million persons, and Phoenix, Arizona more than doubles this population with over four million residents. Trends identified by the 2010 Census suggest that both of these populations will continue to increase.

These heavily-populated areas currently pose increasingly significant problems for electric utility companies, particularly during the summer months when residents stave off sweltering heat by running air conditioning systems in their homes nearly constantly. The U.S. Energy Information Administration (EIA) 2009 Residential Energy Consumption Survey (RECS) indicates that 25% of the energy consumed by homes in Arizona is used strictly for air conditioning, or four times the national average [2]. The aggregate effect of large portions of the population simultaneously demanding electricity creates an undesirable strain on the electric utilities' generators. This effect is commonly referred to as peak demand, or peak load, and it commonly occurs during the late summer afternoons when the business day overlaps with people returning home from work, as demonstrated in Figure 1.

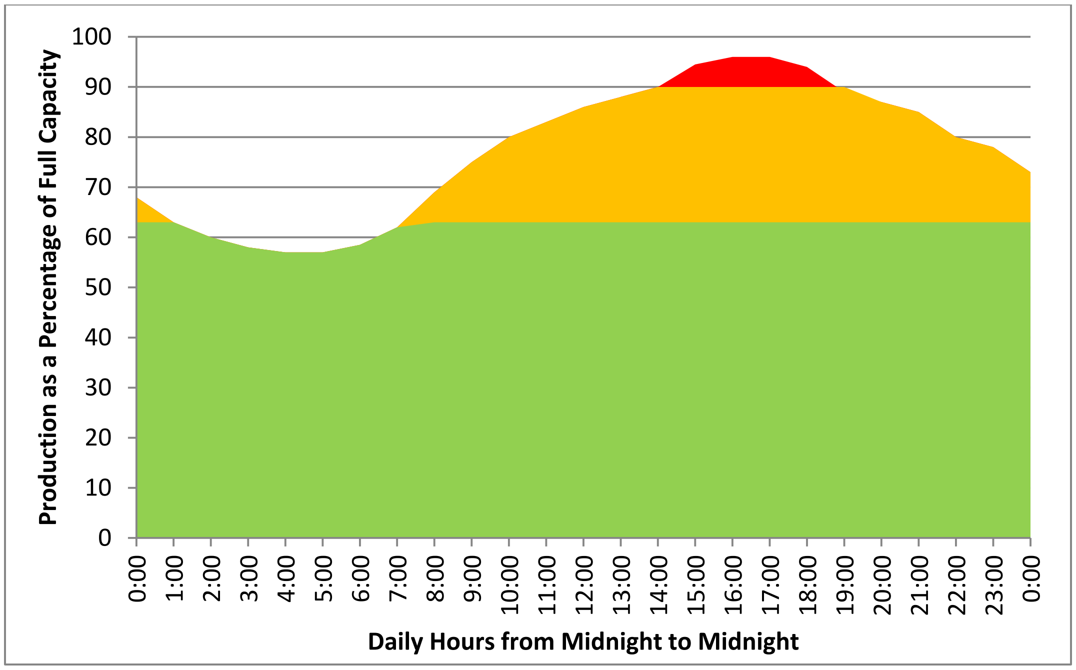


**FIGURE 1 – ILLUSTRATIVE DAILY ELECTRIC UTILITY LOAD CURVE**

Air conditioning isn't the only culprit to create peak load problems, though. Record cold spells in the state of Texas have resulted in its independently-operated grid breaking peak consumption records multiple times within the last five years. It is generally understood that space conditioning, from both residential and commercial buildings, during extreme weather conditions is largely responsible for sharp electrical peak demands around the world [3]. With the limited existence of practical utility-scale energy storage systems, utility companies are forced to meet these increased demands (roughly) in real time in one of three ways. The preferred course of action is to bring additional power plants on-line, often referred to as 'peaker plants' that, due to their supplementary nature, are often more costly to operate and utilize natural resources less efficiently than their primary power plant, or 'base load plant', counterparts. Less preferred is the option to import electricity from another utility, usually at significant cost. As a last resort, when the utilities determine that the cost of meeting the higher-than-normal demand is too great, they implement rolling blackouts to limit demand and match it to their generating capabilities.

Unfortunately, the United States' peak demand continues to rise. The North American Electric Reliability Corporation (NERC) in conjunction with the Energy Information Administration (EIA) publishes this data seasonally [4], and supplements it by offering five-year projections [5]. They anticipate that the national summer peak load will increase almost 7.5% from 2012 to 2017 (Figure 2). That number amounts to nearly 56.5 GW of additional generating capacity potentially needing to be constructed. Considering that utilizing peaker plants is the utilities' most preferred method for handling this load, and that their capacity is generally on the order of 75-100 MW, this projection suggests that as many as 750 additional (rarely utilized) plants would need to be built over the next several years to combat this peak load increase.

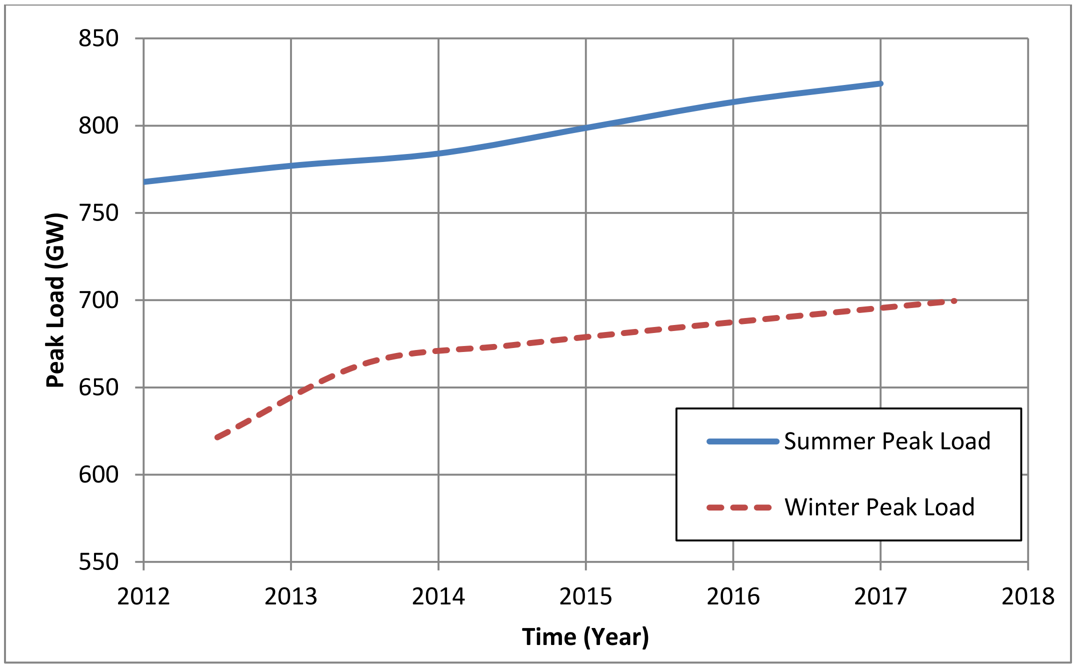


**FIGURE 2 – NORTH AMERICAN PEAK LOAD PROJECTION [4][5]**

Alternatively, the resources that prospectively could be used to build these plants could instead be spent on reducing *consumption*, thereby addressing the issue at its root cause, rather than attempting to meet steadily increasing demand with costly additional generators. Suffice to say, the United States Southwest is not the only area of the world that could stand to benefit from measures aimed at reducing this peak electrical demand. Multiple energy management techniques and advanced technologies are currently being developed by researchers across the globe. These advancements are generally referred to as being contributions to the 'smart-grid', and have direct implications on the ability of utilities and customers alike to curb electrical consumption during critical periods. So beneficial are these improvements to the well-being of the global electrical market that spending in this industry is estimated to reach $65 billion annually by 2017 [6].

## 1.2 – Load Management Strategies

Utilities and energy providers have been throttling the production of electricity up and down to meet demand for the past century, but it wasn't until the energy crisis of the 1970's that utilities began to take a serious look at instituting policies and programs to better manage demand rather than just production. Planners and those that outlaid capital spending expenditures realized that there were really only a few hours out of the entire calendar year where the system load approached being within 5% of the peak load (Figure 3).

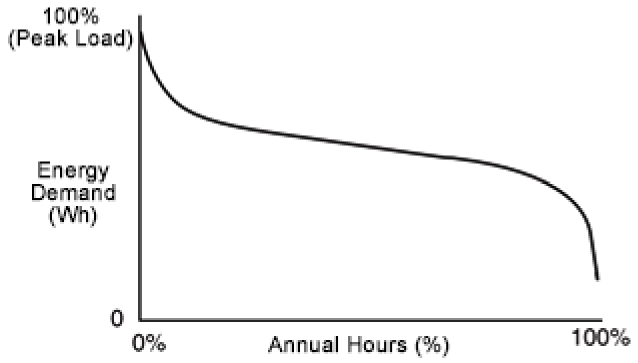


**FIGURE 3 – ILLUSTRATIVE ANNUAL ELECTRICAL LOAD [7]**

With better predictive tools, and improved control over loads during these times, the costs associated with the building, operating, and maintaining of expensive peaking generators could be significantly reduced, if not eliminated entirely.

The industry is divided on some of the classifications used to categorize the differing load management strategies that have been developed in the last 40 years, but a convenient delineation can be made between utility-controlled and customer-controlled strategies. Utility-controlled strategies include supply-side power pooling agreements between utilities that help stabilize electrical loads over a larger network, and all kinds of large scale energy storage systems. Additionally, some utility programs maintain direct control over certain customers' loads and work directly with commercial and industrial entities to establish "interruptible" loads that can be shed when necessary, or economically lucrative. These later strategies fall under the classification of demand side management (DSM) programs, as do the rest of the customer-controlled strategies.

DSM essentially involves any action, policy, or program that aims to alter end users' consumption habits by a reduction or change in the customers' patterns of use [8]. A couple obvious DMS programs include well-known public-relation campaigns that aim to encourage consumers to keep lights off in unoccupied rooms, and to swap incandescent light bulbs with significantly more efficient LED alternatives. These programs can be considered load reduction strategies, as they ultimately aim to reduce overall consumption. Supplementing these efforts is a technique known as curtailment that aims to reduce the amount of power demanded from the utility, without necessarily reducing overall consumption, by substituting grid-supplied power with power produced by on-site solar collection devices or alternative sources of power. This can also be achieved by relatively small battery storage systems that provide an attractive capability to DSM programs in that they allow electricity stored during off-peak hours to be used by the consumer during critical peak times. These batteries have seen an increase in (international) popularity recently as the technology matures and effective battery controllers are developed. The last generalized DSM strategy is demand deferral. A demand deferral program attempts to influence *when* electricity is demanded by either relying on customers to modify their consumption by offering (statically) tiered pricing structures for different times of the day (known as "time of use", or TOU) or by simply volunteering to use resource-intensive appliances and equipment during off-peak time periods.

## 1.3 – Demand Response

Another load management program that is of particular interest to this paper is somewhat of a combination of the aforementioned strategies. So called “demand response” (DR) is defined by the DOE as

> “Changes in electric usage by demand-side resources from their normal consumption patterns in response to changes in the price of electricity over time, or to incentive payments designed to induce lower electricity use at times of high wholesale market prices or when system reliability is jeopardized.” [9]

By this definition, demand response is seemingly an obvious demand deferral strategy, and therefore a customer-controlled DSM. However, the feasibility and usefulness of such a system relies heavily on the utilities’ ability to notify customers of the *dynamic pricing* changes that ultimately drive customer behavior modifications. Therefore, the industry generally refers to any load reduction strategy that is based on monetary incentives or compensation for electricity consumption reduction during peak times as a being a demand response effort.

Dynamic pricing is a concept that is closely related to the aforementioned TOU tiered pricing structure – the cost of electricity varies as a function of time, but unlike a rigidly structured TOU program, dynamic prices are allowed to fluctuate throughout the day. This fluctuation can be managed autonomously through the use of controllers and algorithms that monitor demand and adjust prices accordingly, but due to the difficulty in communicating these pricing changes to the consumer in a way that would influence consumption habits, the fluctuations are generally "triggered" by a utility controller and only instituted during projected periods of peak demand, usually around 24 hours in advance. In this sense, both the consumer and the utility share some burden of responsibility when it comes to implementing DR in an effective manner.

Despite the difficulties associated with negotiating this burden, several influential entities view DR as potentially providing the greatest immediate benefit of all of the DSM programs, including the Federal Energy Regulatory Commission (FERC) and the U.S. Department of Energy (DOE). At the request of Congress as stated in section 571 of the Energy Independence and Security Act of 2007 (EISA), FERC and the DOE developed a national assessment and separate action plan on the implementation of DR in the United States. This assessment was delivered to Congress in June of 2009 and was one of the first national analyses to examine DR on a state-by-state basis and to capture the regional differences (critical for capturing the influence of air conditioning) of peak loading [10]. Following this assessment, per the EISA, a national action plan on demand response (NAPDR) was commissioned and delivered to Congress two years later [11]. Per this implementation proposal, DR can be differentiated into five approaches:

- Dynamic pricing without enabling technology: As described above, this DR type is based purely on the concept of dynamically changing electricity prices influencing consumer behavior.

- Dynamic pricing with enabling technology: Dynamic pricing is still a major component of this type of DR program, but “enabling technology” automatically takes advantage of changes in a dynamic price. Perhaps one of the most obvious candidates for this type of technology is programmable thermostats that communicate directly with the utility to stay abreast of dynamic pricing changes. These thermostats would have the capability of adjusting heating, ventilation, and air conditioning (HVAC) usage to operate as cost effectively as possible during critical peak pricing time periods.
- Direct Load Control: This DR program is, as previously described, an agreement between the consumer and the utility whereby the utility directly controls consumer demand during critical periods.
- Interruptible tariffs: Often also referred to as interruptible rates (INTR), these contracted agreements between consumer and utility establish an agreed-upon amount of energy the consumer is willing, or able, to shed quickly upon notification from the utility. This results in a monetary incentive of some sort that benefits the consumer, but as cumbersome as these agreements can be, they are often only available to commercial, industrial, and governmental buildings of at least a modest size.
- Other DR programs: Other, more intricate, DR strategies are available strictly to consumers of large amounts of electricity, including capacity and demand bidding whereby the consumer submits frequent load reduction bids to their utility in exchange for a tariff, rebate, or other incentive. These programs can be either price-initiated or instead based on reliability concerns.

The NAPDR simulated these five different approaches to DR implementation by breaking down their rates of adoption into four indicative scenarios. The distinction between these scenarios is mostly based on dynamic pricing participation, realizing that any substantial amount of dynamic pricing participation is predicated on the installation of an infrastructure that is capable of bilateral communications between the utility and consumer.

- Business-As-Usual (BAU) constitutes a steady continuation of current DR programs.
- Expanded BAU (EBAU) is the scenario by which current DR programs are maintained and expanded to all states. Critically, this scenario does not consider the impact of dynamic pricing.
- Achievable Participation (AP) assumes that dynamic pricing with enabling technology is offered to nearly all consumers, with at least 60% of said consumers choosing to participate in such a program. This scenario is considered to be the most realistically achievable.
- Full Participation (FP) is the scenario that attempts to simulate the maximum possible benefit of DR. It assumes that 100% of consumers mandatorily participate in DR.

By examining data from surveys and case studies across the country, the NAPDR projected the potential impact of these DR programs and extrapolated them to 2019, as shown in Figure 4. These simulations were created by examining data for the 15 highest load days of the year, and analyzing these days as if they had utilized DR programs for just four hours each day. BAU shows a modest 4% peak reduction is potentially possible by 2019, but remembering that the peak load is projected to increase by nearly 7% over this same time period, BAU is not enough of a strategy to effectively reduce the country's peak load. However, by simply expanding current programs and implementing them across all 50 states, a 9% peak reduction could be achieved by 2019. This would offset the estimated natural peak load growth, but considering the error associated with such projections, the program would likely not be a sustainable long-term solution to minimizing peak load. The NAPDR suggests, in accordance with the DOE's definition of demand response, that the true benefit of DR lies in dynamic pricing. By achieving a 60% marketplace penetration of dynamic pricing programs, the peak load could be reduced nearly 9%. When taken into consideration with the other DR approaches, 14% of the peak load could be reduced by 2019 – nearly twice as much as its anticipated growth. Although not realistically achievable by 2019, the peak could be reduced 20%, mostly by taking advantage of dynamic pricing simultaneously with enabling technology.

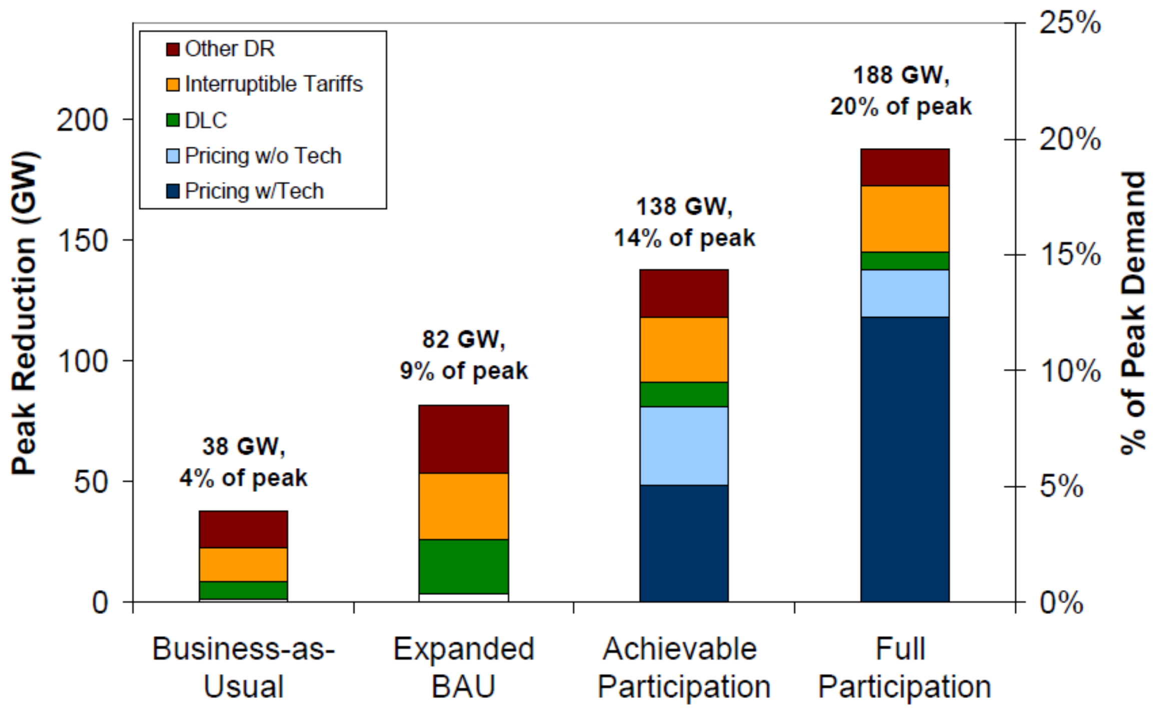


**FIGURE 4 – U.S DEMAND RESPONSE POTENTIAL BY PROGRAM TYPE (2019) [10]**

Additionally, the NAPDR also categorized their simulation results as a function of building type, rather than by DR program (Figure 5). Their results indicate that within the realistic AP scenario, essentially half of the total 14% peak reduction would be due to residences taking advantage of dynamic pricing.

One of the conclusions drawn from the NAPDR was that, although the majority of DR efforts in effect today come from large industrial and commercial consumers, it is the residential class of buildings that represents the largest per-customer potential for DR benefit stemming from the adoption of dynamic pricing programs. It is this conclusion that forms the premise of this paper.

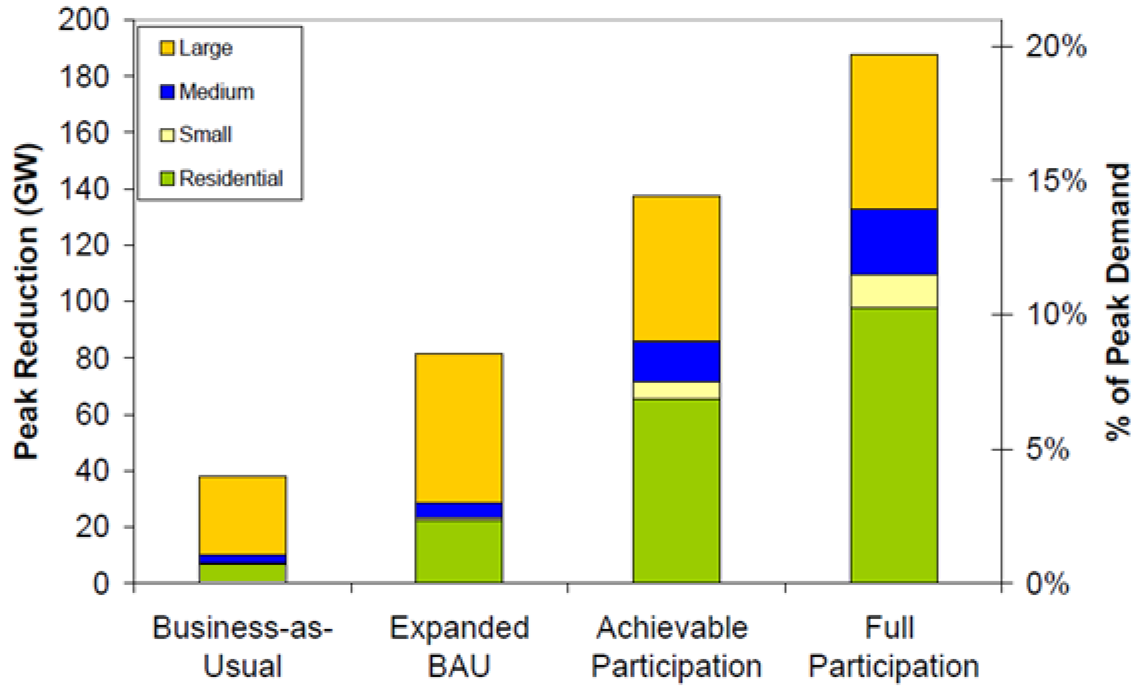


**FIGURE 5 – U.S. DEMAND RESPONSE POTENTIAL BY CLASS (2019) [10]**

## 1.4 – Project Details

Villa Trieste is a 185-unit housing development located near the western edge of Las Vegas, Nevada that was constructed specifically to research the efficacy of peak shifting/reduction strategies in a cooling-dominated climate. This was made possible through a Department of Energy (DOE) grant and the subsequent creation of a partnership between the University of Nevada Las Vegas, Pulte Homes (home builders), and NV Energy (electrical utility company). The primary objective of the DOE's grant is to achieve peak load reductions of up to 65%, as measured at the substation that services Villa Trieste, when compared to a standard-production housing development. To assist researchers in addressing this objective, each home at Villa Trieste has been built with a 1.8 kW PV array located on its roof. In addition, a limited networking infrastructure is included within each home to facilitate the eventual installation of "smart" bilaterally communicating thermostats that would have the capability of connecting to an off-site server. Each home is comprised of two finished stories, and is one of four available floor plans ranging in size from 1,487-1,960 $ft^2$. As of early 2014, nearly every one of the almost 200 Villa Trieste homes is occupied and making use of its included PV array, but logistical issues have impeded the installation of the aforementioned thermostats.

An important detail of the project that needs to be mentioned is the fact that the investigations included within this paper were not part of the original project proposal. As such, certain system elements were not necessarily optimized with the objectives of this paper in mind. Some of these shortcomings will be expanded upon within this paper, but the fact remains that the Villa Trieste housing development provides an excellent source of empirical data for the investigations included herein.

## 1.5 – Objectives

As discussed, the premise of this paper stems from the assertion that DR implementation and the adoption of dynamic pricing within the residential building sector is an excellent strategy for managing peak loading. Within the context of the Villa Trieste housing development, which is located in the cooling-dominated climate of Las Vegas, DR is largely synonymous with air conditioning management. Large commercial and industrial facilities have been minimizing the costs associated with air conditioning utilization since the early 1980's when the concept of 'intelligent buildings' was introduced. One way in which these facility managers minimize operating costs is by constructing intricate computer models that allow them to understand, in advance, the impact a certain amount of cooling (or heating) of a particular zone will have on said zone's temperature. Equipped with this knowledge, facility managers can, for example, schedule a chiller plant to operate at a time optimized to allow the building to reach a certain temperature before employees arrive to the building in the morning. This approach reduces the waste associated with unnecessarily conditioning the building when it is without occupants, and eliminates the amount of guesswork involved with manual operation. Despite the successes of these sorts of transient thermal models, they are not often used within the residential sector due to a variety of limitations. This paper examines these limitations, and attempts to overcome them while at the same time providing a method for accurately predicting the transient thermal response of a residential building.

In addition to providing the method for predicting these buildings' transient thermal responses, several applications for these models are introduced and discussed. One such application is a server-side controller that attempts to utilize thermal response predictions to schedule and *coordinate* the air conditioning loading cycles of a certain sample of houses. This level of coordination attempts to ensure that, at any given time, a number of houses are *not* utilizing their air conditioners, and therefore not contributing to a near-peak load. Another possible application is server-side tracking of a home's heating and cooling characteristics. As will be shown, a home's transient responses are predictable enough that, given a large enough sample size, the rate at which it takes an air conditioner to cool down a home could be monitored over time to potentially identify HVAC equipment failures and provide 'early alert' type warnings to home owners[12].

# Chapter 2 – Thermal Response Models and Controllers

As has been alluded to, it was the energy crisis of 1973 that motivated professionals to emphasize the importance of energy efficiency in building design. This particular period of time also happened to coincide with the creation of the microprocessor and the rapid development of consumer-level computers. This greatly increased researchers' ability to quickly perform complex calculations. With this level of computational power at their disposal, researchers began developing more advanced dynamic thermal models to simulate building performance. Over the next twenty-five years, over three hundred different types of models were developed [13]. The one thing that nearly all these models had in common was that they were constructed to evaluate the heat transfer equations for conduction, convection, and radiation under varying ambient and interior conditions. In fact, one of these early models, TRNSYS, developed by researchers at the University of Wisconsin, is in its seventeenth revision and still widely used today to simulate thermal systems [14].

As the amount of this research grew through the 1980's, certain models' characteristics that were identified during a building's design phase began to ultimately influence the operation and autonomous control of said building's mechanical equipment. With the advent of microcontrollers and increasing commercial viability of embedded circuits, analog equipment controls were phased out in favor of digital replacements that allowed for previously unprecedented levels of logical control over mechanical systems. During this time, an entire industry materialized that focused on developing building models that integrated with, and ultimately controlled, buildings' HVAC equipment. This became known as model predictive control (MPC).  Although certain MPCs may rely on different types of input and output data, and the methods by which they utilize said data may differ, they all attempt to predict the future behavior of a building in an attempt to maximize some sort of cost function over a specified duration. Nearly all the research that has been reported on HVAC-related MPCs indicate that they achieve higher levels of energy efficient operation over non-predictive control methods [15].

Of course, the types of buildings initially being subjected to MPC were the ones that stood to monetarily benefit the most from the extensive engineering analyses – large commercial and industrial facilities. This has remained relatively unchanged for the last 30 years. Residential buildings really have yet to benefit from advanced modeling and control for a variety of reasons including their size, type of mechanical equipment, predictability, and benefit [16]. Residential buildings have a much smaller thermal mass when compared to industrial and commercial buildings, and therefore exhibit transient thermal responses that are more difficult to capture within the framework of a model [17]. In addition, occupancy patterns are much more irregular in a home than in a Monday through Friday, 9-to-5, commercial facility, which ultimately impacts the nearly immeasurable internal gains of a building. However, perhaps the largest obstacle facing advanced residential HVAC control is the level of autonomy necessary to make it practical. Within the context of an industrial or commercial building, a facilities engineer is usually ultimately responsible for the operation of the HVAC equipment. They generally have access to very specific physical thermal characteristics of their building that allows them to build intricate and accurate predictive models. The same engineers are on-site when systems fail and need to be manually overridden, or models need to be updated and improved. An advanced controller within a residential home is not afforded this luxury, and must function indefinitely, as intended, and without systemic errors. This can be particularly problematic considering that a single control error could potentially negate hours, or even days, worth of operational savings [18]. With the rising cost of energy and the recent drive for adoption of DR initiatives within the residential sector, research teams have earnestly begun developing comprehensive and accurate house-specific thermal and energy models and controllers [19].

Broadly speaking, there are two distinct types of models defined by the American Society of Heating, Refrigerating, and Air-Conditioning Engineers (ASHRAE) [20] – forward models and data-driven models. The various types of data-driven models are the most relevant to this paper, for reasons that follow.

## 2.1 – Forward Models

Also referred to as the classical approach, a forward model attempts to predict a certain output based on knowledge of forcing inputs (usually ambient conditions such as dry bulb temperature, solar radiation, relative humidity, etc.) and a detailed understanding of the physical building (geometry, wall material and thickness, HVAC equipment specifics and their operating schedules, etc.). The building itself need not necessarily be built to create these models, which is why they are abundantly used during the design stage of building development. The aforementioned TRNSYS code is one such type of model, as are other popular simulation tools such as SPARK, EnergyPlus, and EnergyPlus's predecessors BLAST and DOE-2 [21].

These models are of little use to this paper's objectives, though. In addition to generally being computationally intensive and time-consumingly individually tailored, they rely too much on buildings' physical parameters to be practically applied to a wide variety of residences. They're simply not optimized for dealing with HVAC control and the minimization of cooling costs [18]. Fundamentally, the 'knowns' of a forward model are the forcing inputs (electrical consumption, HVAC operation, etc.) and the physical characteristics, while the unknowns are the outputs (temperature and humidity versus time); in the case of the Villa Trieste homes, and more generally *any* home with a bilaterally communicating thermostat, the 'knowns' are the inputs and the outputs and the ‘unknowns’ are physical characteristics of the building. Thus, a different sort of model is used when describing the physical parameters of a building. These types of models rely on empirical output data, and known as data-driven models.

## 2.2 – Data-Driven Models

These models are also often referred to as inverse models. Unlike forward models, they use the outputs of a building, or system, in conjunction with its inputs to deduce, or quantify, the building’s parameters and their effect on the forcing inputs. These are the types of models used once a building has been constructed and performance data is available. Since this method relies on empirical data, the derived models are generally more simple to use, they are easier to validate than forward models, and they often result in more accurate predictions of future performance than forward models [20].

Although the form and exact specifics of a building’s performance data vary from method to method, the data itself can be obtained in one of two ways–either passively under a building’s normal operation, in which case the collected dataset is referred to as ‘nonintrusive’, or while the building is being subjected to an experimental and predetermined set of conditions in which case the dataset is considered to be ‘intrusive’. Intrusive datasets generally result in more accurate models than nonintrusive dataset since they guarantee the model is subjected to a wider range of conditions than are generally encountered during normal operation–the variety of the sample size is greater.

Data-driven models can be further classified by the frequency and method by which they are trained. ‘Offline’ models are trained from a static historical dataset; they are then completely retrained when subjected to a new dataset. ‘Online’, or real-time, models differ in that they do not necessarily rely on a fixed dataset for training. Real-time data can be processed by the model, allowing it to make incremental changes to itself as it is fed new information. Online models can be computationally very expensive–they essentially continuously optimize some sort of cost function, whereas offline models optimize a comparable cost function much less frequently. For this reason, offline models tend to be easier to implement and are represented more within the literature [22] .

### *2.2.1 – White-Box (Physical) Models*

Data-driven models can be further distinguished by the degree to which they depend on an understanding of a given building's physical system. On one end of the spectrum, white-box, or physical, models require a *complete* understanding of a building's physical characteristics. They involve a physical description that generally includes features such as building geometry, materials, geographic location, and HVAC details. They are generally regarded more as forward models than data-driven models, but are occasionally used within the data-driven realm for relatively simple systems due to their high level of accuracy [23]. The effort involved in collecting and establishing all of a building's parameters is too prohibitive to allow white-box models to be regularly used in practice, and the models they render are generally fairly complex.

### *2.2.2 – Black-Box (Empirical or Parametric) Models*

On the other end of the spectrum, black-box models require little-to-no knowledge of a building's physical characteristics. They are essentially stochastic models; they work by parameterizing certain aspects of a system either statistically or in terms of differential equations and transfer functions. Quite often the resulting model is formulated in such a way that it is difficult to relate it to any sort of physical meaning. Black-box models are 'trained' with historical data, and often don't perform well when the forcing inputs deviate from the set of training data. Despite this, black-box models are the most widely used data-driven approach for evaluating demand-side management programs [20]; their predictions are relatively accurate and they require significantly less computational power than their white-box model counterparts.

#### *2.2.2.1– Linear Regression Models*

These types of models vary considerably in their approach, and are either completely statistically based, or rooted in some sort of physics formulation. What they generally all have in common is that they are primarily based on linear differential equations, although most systems exhibit some sort of nonlinearity. Coefficients are strategically added to the differential equation, and then regressed in a variety of fashions to best-fit the DE to a set of training data. Most regression methods attempt to find functional relationships between weather variables and building outputs [24]. One of the earliest efforts into digital, automatic, and predictive thermal control of small-scale buildings dates back to 1988 [25], and utilized this method. Shapiro et al. used a thermal resistance and capacitance (RC) model (that described the building with just five parameters), analogous to an electrical circuit, in conjunction with forecasted weather data to project anticipated zone temperatures and adjust pre heating/cooling strategies accordingly. Recently, a similar RC method was used to predict cooling loads of an interior room without occupants, but instead used a non-linear regression algorithm [17][26].

Another approach abandons the RC analogy, and instead relies on the weighted parameterization of integrated input variables [18]. The method outlined by Rabl et al. relies on just four inputs: non-solar heat input to the building (internal gains), solar radiation, exterior temperature, and interior temperature as well as their corresponding coefficients. An analysis of this method examined the accuracy of a first-order versus that of a second-order model. What the analysis revealed was that the error associated with the first-order model was equivalent to the magnitude of its time constant – there was little benefit to using the more complex second-order model for their purposes. However, they include a caveat that first-order models *may* not work as well for other applications.

#### *2.2.2.2 – Artificial Neural Network Models*

Artificial neural networks, or often just abbreviated 'neural networks' (NN), differ considerably, in principle, from regression models.  That is, unlike regression models, the structure of a NN is not derived or deduced from any sort of understanding of physics or heat transfer. Instead, they are models that are completely inferred from sets of data – they are *true* black-boxes in this sense. Without the fundamental physical basis of regression models, NNs are not at all dependent on linear differential equations. This is beneficial when dealing with the inherent nonlinearity of transient thermal problems. In fact, NNs were born from the desire to improve on the existing legacy of linear models throughout all of the physical sciences. Early developers of the NN were intent on mimicking the behavior and function of a biological brain–they wanted to develop algorithms that were capable of 'learning' a system over time. Many references will still refer to this analogy with the operation of a brain, but in reality, present-tense neural networks bear little resemblance to biology, and are rather simply advanced statistical models.

Neural networks are composed of multiple interconnected node 'layers', with weights associated with each of the interconnections. As data flows into the model from the input layer, they are multiplied by their respective connection's weight, $w_i$, and summed at a 'neuron', or node (Figure 6). This summation is then compared against an activation value, $w_{i0}$, or threshold weight, that is a property of the node itself and is always just multiplied by one. When the summed data, $h_i$ satisfies the threshold, they are run through an activation function, generally a sigmoid, and the node generates an output value, $y_i$. Mathematically, this is described by Eqn.(2.1).

To expand the NN, the input values are sent to more nodes, and those nodes feed into (potentially) more layers. This ultimately creates a multilayered neural network (Figure 7), described by Eqn. (2.2). It is the construction of these intermediary, or hidden, layers between the inputs and outputs that proves to be the most challenging task when building NNs.

This structure has no real way of 'learning' and self-adjusting its weights, though. There are multiple techniques for training NNs, but within the building modeling literature, backpropagation, or backprop, seems to be the most popular through the use of the Levenberg-Marquardt method. Essentially, after the NN calculates output values, they are compared against the *expected* output and a difference between the values is calculated. This error is then back propagated through the network, and weights are adjusted in such a way to reduce the error. This learning algorithm is advantageous due to its speed, as well as its autonomy. It has also been proven to be successful for multiple MPC applications [24][27][28][29][30][31][32][33][34].

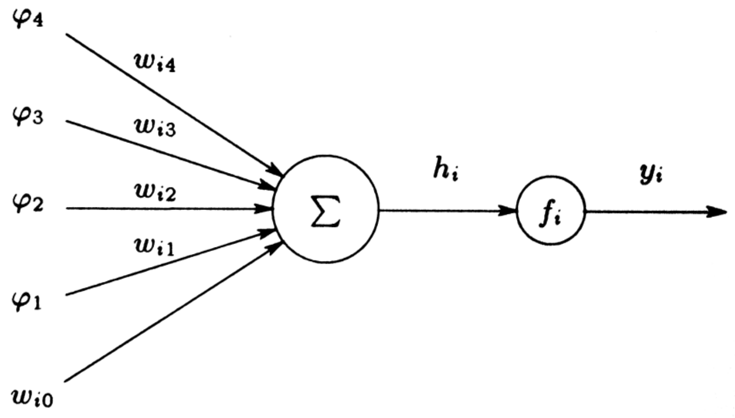


**FIGURE 6 – A SINGLE NEURAL NETWORK NODE** [35]

After the network has been exposed to a relatively small amount of training data, the iterative changes made to the weights by back propagation are minimal. One of the advantages of this approach is that nearly *anything* can be used as an input to the system. When it comes to modeling residential energy consumption, for example, some of the obvious inputs would be ambient temperature, solar radiation, etc., but NNs are flexible enough to consider less dynamic inputs, as well. Aydinalp has investigated constructing NNs in such a way to include socio-economic factors such as household income, dwelling type, size of the house, and employment status of homes' adults [28].

$$y_i = f_i(h_i) = f_i\left(\sum\nolimits_{j=1}^{n} w_{i,j} + w_{i,0}\right) \tag{2.1}$$

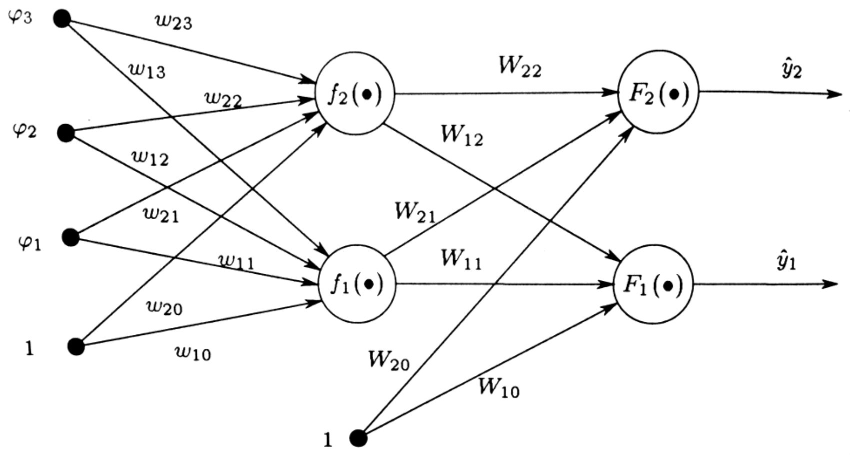


**FIGURE 7 – MULTILAYERED NEURAL NETWORK** [35]

$$\hat{y}_i(t) = g_i[\varphi, \theta] = F_i\left[\sum^{n_h} W_{i,j} f_j\left(\sum^{n_\varphi} w_{j,l}\varphi_l + w_{j,0}\right) + W_{i,0}\right] \tag{2.2}$$

During the 1990's, the American Society of Heating, Refrigeration, and Air-Conditioning Engineers (ASHRAE) sponsored a pair of competitions known as *The Great Energy Predictor Shootout I & II*. The purpose of both competitions was to pit black-box models against each other to find the most effective approach for modeling commercial building energy usage, with the end goal being to accurately predict the operational cost savings of retrofits. One of the conclusions shared between both competitions was that neural networks provide the most accurate model of a building's energy use [36], though their accuracy is highly dependent on the construction of the network and the arrangement of nodes.

Neural networks had been gaining popularity amongst the building modeling community before The Great Energy Predictor Shootouts [24] (on a larger scale, at least 35 utilities had adopted NNs for short-term load forecasting by 1998 [29]), but it has only been during the last 15 years that they have been incorporated into full-blown HVAC-related MPC environments [37]. A couple of related studies explored using NNs for predicting the optimal start/stop times of heating systems  [27][38], and a similar method has been applied to deducing the optimal start/stop times of air conditioners [39]. However, these studies relied on training data that was generated from forward-models.

Ruano et al. were able to take a comparable approach to HVAC start/stop times, but instead opted to utilize an adaptive, rather than static, NN like the prior study [40]. A static, or off-line, NN is 'taught' strictly from its initial training set, which can cause accuracy issues when inputs vary substantially outside of the bounds of the training data. The network simply has no experience on how to handle the differing inputs, and no way of correcting itself. This (usually) undesirable effect could be observed by training a model with Spring/Fall ambient weather conditions then subjecting it to Summer/Winter data. This issue can be circumvented by utilizing an adaptive, or on-line, NN with the tradeoff being a bit of computational power. Adaptive NNs never really turn off of training mode. Instead, for each time step they consider a certain amount of historical data as training data, and recalculate their weights accordingly.

### *2.2.3 – Grey-Box Models*

As discussed, white-box models are completely based on the physical characteristics of a building and the underlying principles of physics, and black-box models are generally more statistically based with no considerations for specific building characteristics, so grey-box, or semi-physical, models are, naturally, a combination of both approaches. Like white-box models, they make use of physical characteristic parameters, but rather than being exact representations of the building, they're generalized from a rough framework. Similar to black-box models, these parameters are then fit to the system's known empirical data with black-box techniques such as linear regression or other statistical methods. This sort of hybrid model had been suggested before [41], but Deque et al. were some of the first to use the 'grey-box' phrase and present a general methodology [42].Their examination of an unoccupied French home during the summer months revealed that their grey-box model was within 5% of accurately predicting the home's cooling load when compared to a much more computationally intensive physical model. Almost simultaneously, Oussar and Dreyfus presented several arguments for the utility of such a an approach within the realm of engineering and industrial controls [43]. Another study used a physics-based approach in conjunction with an ARMA algorithm to achieve a model accuracy of around 1% for a California office building [44]. Yet another set of papers utilized the familiar thermal circuit analog with non-linear regression and parameter estimation techniques to successfully predict optimal temperature setpoints that ultimately limited peak demands in several commercial buildings between 33%-51% [45][46].

# Chapter 3 – Mechanical Systems and Data-Collection Hardware

Regarding some of the physical details of the buildings being analyzed within this thesis, one of the unique advantages to the Villa Trieste project is that the development was constructed specifically as a means for collecting real-world empirical data on the impact of different demand reduction strategies. This chapter will develop an understanding of some of the mechanical systems included within the Villa Trieste homes, as well as a summary of the devices that are used to control and collect data on said systems.

## 3.1 – Data Collection Hardware, Phase I

The first phase took place from early February 2011 to late September 2012, and was a preliminary data collecting effort undertaken while the specifics of the second phase were being developed. The owners of seven homes participated in this initial pilot, and agreed to have their energy consumption, and solar panel generation, monitored. Additionally, temperature measurements were recorded in each of these homes for 'representative' rooms on both first and second floors. Data was recorded at a rate of once per minute for the duration of the experiment.

### *3.1.1 – Energy Monitoring*

At each of the seven participating homes, project boxes were installed next to each home's respective utility service panel, either within a garage or outside next to the panel. Each box was loaded with five WattNode Pulse watt-hour transducers[1] and a wireless ZigBee module (Figure 8). The community-wide installation of these devices resulted in the creation of a local ZigBee mesh network. A single GSM[2] modem was added to this network, which was configured to

[1] Model WNB-3Y-208-P produced by Continental Control Systems LLC
[2] The Global System for Mobile communications (GSM) is an international communications standard most often used by cellular devices.

communicate, through the Internet, with a data-logging database server located at UNLV's campus (Figure 9).

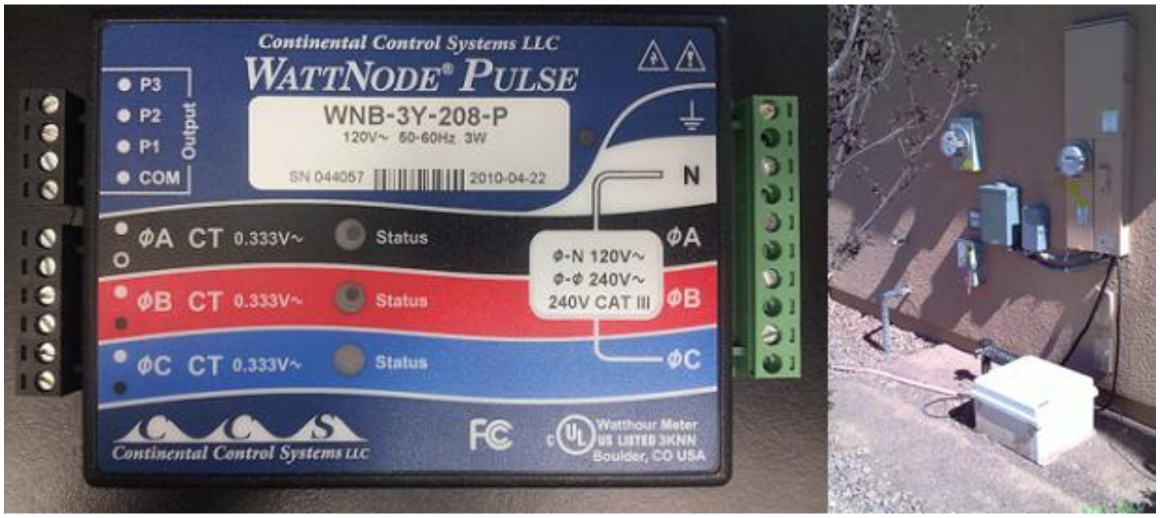


FIGURE 8 – WATTNODE MODEL WNB-3Y-208-P AND INSTALLED PROJECT BOX

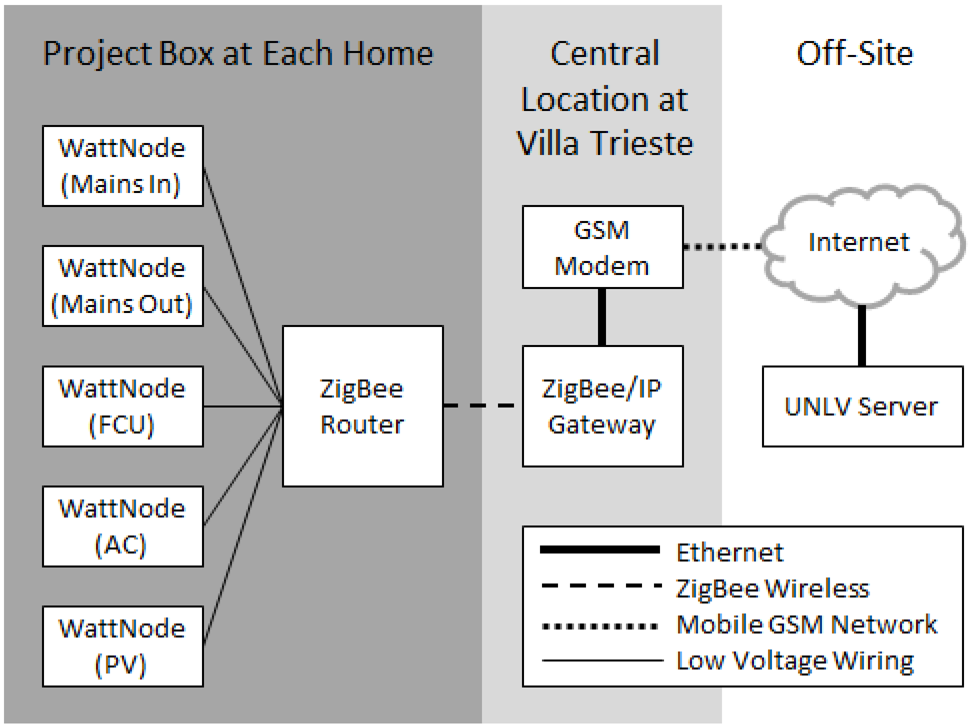


FIGURE 9 – ENERGY CONSUMPTION/GENERATION DATA LOGGING DIAGRAM

The CTs attached to each WattNode device were wired so that they could monitor the circuits described in Table 1. However, not every installation went in with consistently-sized CTs. The CT sizes that were installed in each home are shown in Table 2. The only appreciable difference in these installations is the scaling factor that is ultimately used to convert the WattNode measurements to energy or power units, as shown in Chapter 4.

**TABLE 1 – CURRENT TRANSDUCER APPLICATIONS FOR ENERGY MONITORING**

| Circuit Description | Database Designation |
|---|---|
| Mains In | data1 |
| Mains Out | data2 |
| Fan Circulating Unit | data3 |
| Air Conditioner | data4 |
| PV Array | data5 |

**TABLE 2 – SENSOR NAME DESIGNATIONS AND CT RATINGS**

| | | Temperature Sensor ID | | CT Ratings (Amps) | | | | |
|---|---|---|---|---|---|---|---|---|
| House | ZigBee ID | First Floor | Second Floor | Mains In | Mains Out | FCU | Cond | PV |
| 1 | PWR4 | TEMP5 | TEMP 6 | 150 | - | 30 | 50 | 30 |
| 2 | PWR 7 | TEMP 7 | TEMP 8 | 150 | 30 | 50 | 150 | 50 |
| 3 | PWR 6 | TEMP 9 | TEMP 10 | 150 | - | 30 | 50 | 30 |
| 4 | PWR 8 | TEMP 11 | TEMP 12 | 150 | 30 | 50 | 150 | 50 |
| 5 | PWR 3 | TEMP 13 | TEMP 14 | 150 | 30 | 50 | 150 | 50 |
| 6 | PWR 2 | TEMP 15 | TEMP 16 | 150 | 30 | 50 | 150 | 50 |
| 7 | PWR 5 | TEMP 17 | TEMP 18 | 150 | 30 | 50 | 50 | 50 |

### *3.1.2– Temperature Monitoring*

The ZigBee mesh network that was just described also extended within the homes where temperature sensors were located. Each home included two Texas Instruments TMP102 digital temperature sensors, so that the temperatures within both first and second stories could be monitored. The sensors were stored inside enclosures that were created to be plugged in to standard 120V wall outlets. These enclosures not only provided power to the sensors, as well as the supplementary ZigBee hardware, but they also kept the equipment stationary at approximately a foot off the floor in all locations. Every enclosure measured temperature, and a select few additionally measured humidity. Devices were identified and tabulated in the database by the nomenclature TEMPX where X designates a number between 5 and 18 (Table 2).

## 3.2 – Data Collection Hardware, Phase II

Phase II was developed to serve slightly different purposes in a less intrusive, more interactive way. Naturally, there are some notable differences in the equipment used during this phase when compared to the first. During Phase I, equipment was installed in seven Villa Trieste homes to record their respective interior temperatures from February 2011 to September 2012, as well as power generation and consumption. Phase II differs in this regard. Power measurements are not included within the scope of this experiment, and temperature sensors do not exist in the form of stand-alone enclosures plugged in to electrical outlets. Instead, temperature sensors are replaced with a functional thermostat that not only monitors room temperatures and controls HVAC equipment through conventional means, but also acts as a two-way communicating device with UNLV's on-campus server. The only hardware supplementing this thermostat is a gateway in each home that bridges communications between the thermostats' ZigBee mesh network and the Internet.

Researchers at UNLV constructed an intuitive web-based interface that allows homeowners to monitor, schedule, and control their thermostats' setpoints. A restricted portion of the same website includes controls for the utility company to implement temperature setbacks during DR events. The intent is that the system will allow homeowners to use familiar thermostat hardware, while at the same time providing a DSM environment that allows the utility to make progress toward the “Full Participation” scenario outlined by NAPDR.

## 3.3– HVAC

It has been previously established that demand response initiatives in cooling-dominated climates are generally synonymous with air-conditioning management programs. Many of the thermal models reference during Chapter 2 allude to this, but the majority of them are focused on the mechanical optimization of large commercial facilities. A lot of the same concepts are still applicable, but there is at least one critical difference between the operation of these large buildings and small homes–the HVAC systems themselves. In large buildings, the air conditioners can be quite complex and composed of many individual integrated components (chillers, cooling towers, etc.) that can each be purchased, serviced, and replaced when necessary. Their controllers tend to be just as complex in terms of their ability to coordinate all this equipment, but due to the size of the buildings they service, they are generally set to run as steady-state systems for extended periods of time.

Residential HVAC systems are much different. A little over 60% of homes in the United States now feature central air units [47] as their primary means of space cooling, but across the Mountain South region that includes portions of Arizona, Nevada, and New Mexico, this number jumps to nearly 80%. These popular units are their own self-contained vapor-compression-cycle-based systems, and most combine all the components of a holistic HVAC system. For obvious reasons, they are controlled much more simplistically than large building HVAC systems. Typically, a thermostat located within the home is the controller responsible for dictating when heating, ventilation, or air-conditioning components turn on and off. This type of control strategy treats the entire home as a single zone, and is referred to as bang-bang or hysteretic control. It is not a particularly efficient way to run a mechanical system, but variable speed motor controllers have yet to make an appreciable impact on the central air-conditioning market in the United States [48]. Often, logic is included within these thermostats to prevent units from rapid cycling, which could be damaging to components. These units are also often oversized, and the combination of all these factors results in these systems displaying a much more cyclical behavior than their large-building counterparts. Combined with the fact that smaller buildings are categorically skin-dominated when compared to larger buildings (which makes them more susceptible to ambient condition changes), it is easy to understand why thermal and mechanical modeling is not nearly as well represented within the literature for residential buildings as it is for larger facilities.

## 3.4– Thermostats

As just discussed, thermostats are a residential building's primary means of autonomous HVAC control. Modern microprocessor and thermistor-based thermostats are actually quite simple, and have not varied substantially in their functionality from their bi-metallic electromechanical predecessors. The basic thermostat allows a user to specify a desired zone temperature, or 'setpoint'. It then compares the localized temperature measurement from its temperature sensing element to this user-specified setpoint, and activates *cooling* when the temperature exceeds the setpoint. Similarly, it initiates *heating* when the temperature falls below the setpoint. As presented, this would result in heating and cooling equipment battling nearly constantly to maintain a consistent setpoint temperature. The reality is that the user is expected to manually place the thermostat in either heating or cooling mode. For example, while in cooling mode, the thermostat recognizes that it is only responsible for cooling the home, and that it will passively heat back up. Some modern thermostats are capable of operating in *auto* mode, whereby no manual intervention is necessary and the controls logic of the thermostat includes provisions to prevent the air conditioner and heater from working against each other.

In terms of activating the heating and cooling modes, most modern thermostats accomplish this by interfacing with an external 24 V transformer and relay board that is generally included as part of the central air system, as seen in Figure 10. These connections can be slightly more complex depending on the heating and cooling sources used, but the basic wiring diagram of Figure 10 is adequate to develop an understanding of the systems located in Villa Trieste.

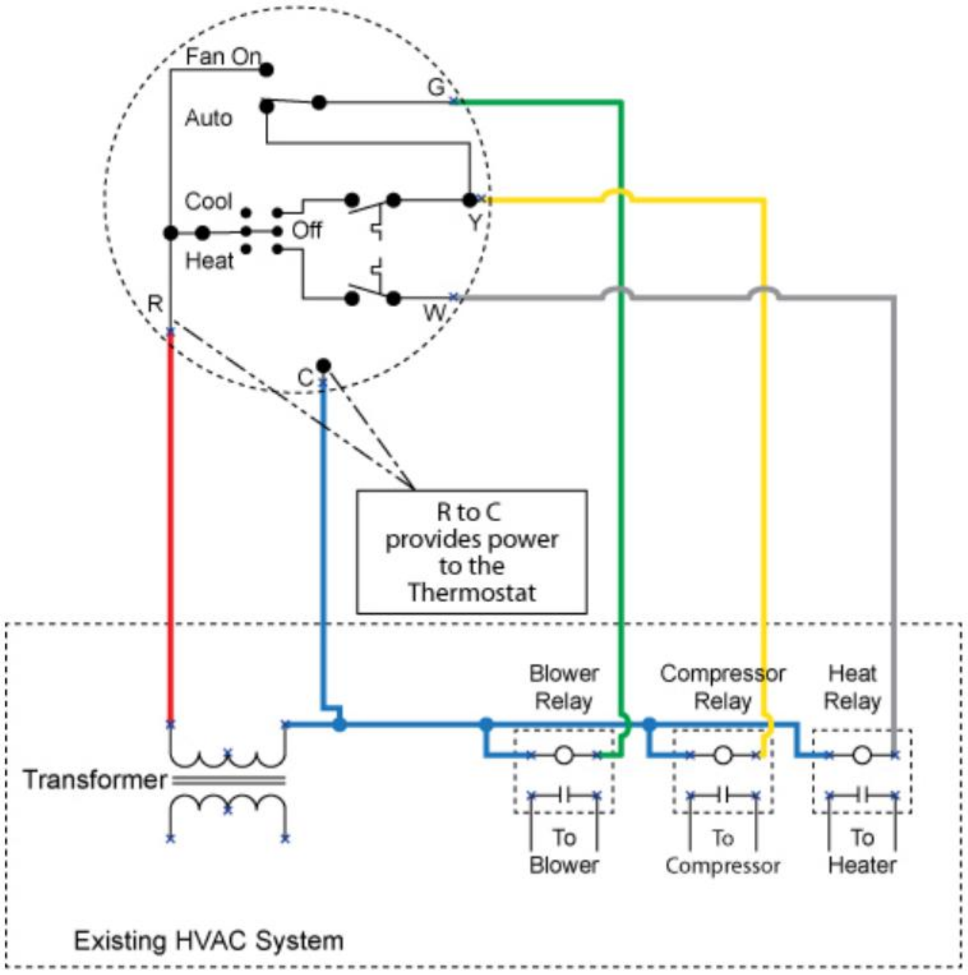


**FIGURE 10 – BASIC THERMOSTAT CIRCUIT** [49]

These 24 V 'low-voltage' systems are not the only type of thermostats on the market – millivolt and line voltage systems have some niche applications, as well – but the National Electrical Manufacturers Association's (NEMA) publication of NEMA DC 3 for the first time in 1972 established a standard for governing low-voltage thermostats, cementing their place in the residential market. One of the accomplishments of this standard was that it established an alphanumeric code (Table 3) for the terminals shown in Figure 10, as well as a collection of additional terminals that accommodate the slightly more complex installations. This is an important, if not explicitly stated, fact of the development of Phase II's thermostat – modern thermostats are essentially completely interchangeable. The method by which they interact with HVAC equipment does not substantially differ from model to model, so when choosing or developing a thermostat to place in the Villa Trieste homes for Phase II, external communication and logic capabilities are essentially the only criteria by which *any* thermostats need be compared.

**TABLE 3 – BASIC HEAT/COOL SYSTEM TERMINAL DESIGNATIONS** [50]

| **Function** | **Terminal Marking** |
|---|---|
| Power | R |
| Heat transformer power | Rh |
| Cool transformer power | Rc |
| Common | C |
| 1st stage cool | Y or Y1 |
| 2nd stage cool | Y2 |
| 1st stage heat | W or W1 |
| 2nd stage heat | W2 |
| 3rd stage heat | W3 |
| Fan | G |
| Active in heat (i.e., damper, etc.) | B, or O/B |
| Active in cool (i.e., damper, etc.) | O, or O/B |

With this in mind, UNLV researchers decided upon a ZigBee-enabled thermostat[3]. The system was developed to operate as is seen in Figure 11, with each thermostat maintaining a connection to a ZigBee router. A router's purpose is to act as an intermediary between UNLV's servers and the thermostats – transcoding TCP/IP and ZigBee signals back and forth. Though outside the scope of this paper, this system has been shown to successfully and reliably enable two-way communications between the thermostats and the UNLV server.

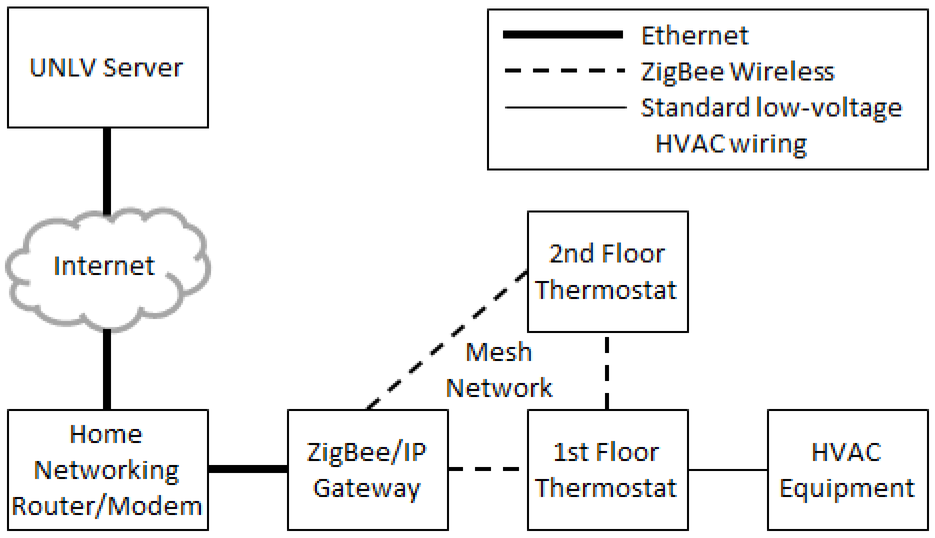


**FIGURE 11 – THERMOSTAT CONNECTIONS DIAGRAM**

With communications between UNLV's servers and the thermostats in place, the system has been configured to log the appropriate data that would make Phase I's temperature measuring equipment unnecessary. The most obvious measurement is temperature, but with that being said, the temperature measurements collected from thermostats differ in not-insignificant ways from the traditional measurement approach taken during Phase I. While the sensors used in Phase I were designed to measure and report data at 1-minute intervals, the thermostats are event-based devices. This means that instead of logging uniformly reported data, the

[3] Manufactured by RCS Technology, the thermostat model is TZB45.

thermostats will only report back to UNLV's servers whenever the temperature it senses *changes*. This is further complicated by the fact that thermostats, and not just the particular model used in the study, have an internal averaging algorithm in place to damper out transient temperature changes. In addition, unlike Phase I's sensors, the thermostats' resolution is only to the nearest degree.

Another issue that is inherent with all residential thermostats [51] is that they are quite often placed in locations where their temperature measurements are poor representations of the whole-zone temperature. Residents combat this by decreasing, or increasing, their thermostat's setpoint to achieve a particular "comfortable" temperature in a particular area of the home that is not necessarily reflected on the thermostat. This would negatively impact thermal models that were heavily reliant on accurate temperature readings, but as will be discussed in Chapter 5, this misrepresentation of the "true" zone temperature, has no impact on the proposed model's efficacy.

These distinctions may seem problematic from a measurements standpoint, but for the purposes of this research, event-based reports offer several advantages. Since the thermostats are capable of communicating *exactly* when they trigger the various relays described above, the UNLV server can log HVAC equipment cut-in and cut-out times with essentially no error. This is critical for the proposed model.

There is a tradeoff with thermostats replacing Phase I equipment, though. The CTs of Phase I located near utility service panels provide valuable information on the amount of energy the air-conditioning and fan controller units consume, which cannot be accomplished with the thermostats' limited capabilities. However, as will be shown in Chapter 4, the equipment's energy consumption can be fairly accurately estimated with some simple techniques.

# Chapter 4 – Villa Trieste Power Consumption

At this point, it is worth reiterating the difference between power and energy. Though often used interchangeably with no ill effects in many contexts, the distinction between the two is important here. Whereas power is an instantaneous measure of the rate at which work is done, energy is power integrated over a duration. This is why electrical power is most commonly expressed in terms of watts, and energy is expressed as watt-hours. While an expanded goal of DR, as outlined in Chapter 1, is to ultimately save *energy* as the technology develops, the original goal of DR is still to reduce peak demand, which, strictly speaking, equates to a reduction in the maximum *power* draw. As described in Chapter 1, power plants are capable of supplying sustained power draws (energy); it is the instantaneous, or short-term, draws that are extremely costly for utilities.

As discussed in Chapter 3, Phase I of the Villa Trieste experiment involved installing WattNode Pulse devices to monitor electricity consumption and generation in seven homes from early February 2011 to late September 2012. Current transducers were installed at each of these homes to measure and record incoming energy, PV generation, outgoing energy, AC consumption, and fan circulation consumption.

## *4.1.1 – Assessing the Quality of Data*

Before undertaking a full analysis of the data, a preliminary examination of the recorded energy data showed some irregularities; certain sensors exhibited long periods with no reported data. An example of the database's structure, along with some data is shown in Table 4. A script was written to help visualize and quantify the inconsistencies in this data, which can be found in Appendix A. Each sensor was examined in weekly increments over the duration of the experiment, and plots were generated to examine the amount of time between recorded measurements. Ideally, all sensors would plot at one minute intervals on the y-axis, indicating there being just 1 minute between data records, but the plots revealed a significant amount of missing data for all sensors across the entire experiment period. A sample plot from a week during June 2011 can be seen in Figure 12. In addition, the database contains no data from the sensor identified as PWR4. This effectively limits the sample size to 6 homes.

**TABLE 4 – WATTNODE PULSE MEASUREMENTS EXAMPLE DATA**

| id | datetime | powerSensor | data1 | data2 | data3 | data4 | data5 |
|---|---|---|---|---|---|---|---|
| 24 | 2/2/2011 13:57 | PWR8 | 0 | 4 | 6 | 0 | 22 |
| 23 | 2/2/2011 13:57 | PWR3 | 0 | 0 | 0 | 0 | 22 |
| 22 | 2/2/2011 13:57 | PWR5 | 2 | 0 | 10 | 0 | 20 |
| 21 | 2/2/2011 13:57 | PWR7 | 0 | 20 | 0 | 0 | 22 |
| 20 | 2/2/2011 13:56 | PWR3 | 0 | 0 | 0 | 0 | 21 |
| 19 | 2/2/2011 13:56 | PWR5 | 2 | 1 | 5 | 0 | 21 |
| 18 | 2/2/2011 13:56 | PWR7 | 0 | 20 | 1 | 0 | 22 |
| 17 | 2/2/2011 13:55 | PWR8 | 0 | 0 | 0 | 0 | 21 |
| 25 | 2/2/2011 13:58 | PWR7 | 0 | 20 | 0 | 0 | 22 |
| 26 | 2/2/2011 13:58 | PWR5 | 3 | 1 | 10 | 0 | 21 |
| 27 | 2/2/2011 13:58 | PWR3 | 0 | 0 | 0 | 0 | 21 |

The missing data could be attributed to a number of problems, but with no local cache or data recording “handshake” confirmation built in to the test setup, some form of communications error would seem to be the likely root cause for the missing data. The irregularities in the reporting between sensors suggests that it was not the GSM mobile network connection at fault. Rather, it was likely the ZigBee mesh network that failed to relay certain sensors’ data to the GSM router. A review of the physical map of Villa Trieste confirms that the steady signals shown in Figure 12 (PWR3, PWR5, and PWR7) were clustered in such a way that they were part of a “chain” of nodes that were in close proximity to the GSM modem, while PWR2 and PWR8 were the nodes furthest away from the mesh network.

Before the start of the 2012 summer season, extenuating circumstances necessitated the GSM modem be moved from its previous location. While this did not completely eliminate the gaps in the recorded data, it did appear to, in general, result in fewer missing pieces of data, as shown by comparing Figures 12 and 13.

By utilizing the script located in Appendix A and modifying it to examine daily, rather than weekly, data, several days during the 2012 DR season were identified that feature relatively few missing records, as seen in Table 5. The next section will use this subset of days in a portion of its analysis.

**TABLE 5 – MINUTES OF POWER DATA MISSING FROM THE RECORD FOR THE DAYS GIVEN**

| | **2012** | | | | | |
|---|---|---|---|---|---|---|
| | **07/02** | **07/08** | **07/09** | **07/17** | **07/22** | **08/08** |
| PWR2 | 1.28 | 1.12 | 14.22 | 2.03 | 4.77 | 30.75 |
| PWR3 | 1.13 | 0.55 | 0.63 | 0.62 | 1.67 | 0.75 |
| PWR5 | 0.60 | 0.55 | 0.57 | 0.60 | 0.68 | 0.57 |
| PWR6 | 0.52 | 2.05 | 1.82 | 2.62 | 6.82 | 0.60 |
| PWR7 | 0.60 | 0.50 | 1.07 | 0.52 | 0.57 | 0.45 |
| PWR8 | 0.52 | 0.53 | 1.57 | 6.52 | 0.57 | 0.53 |

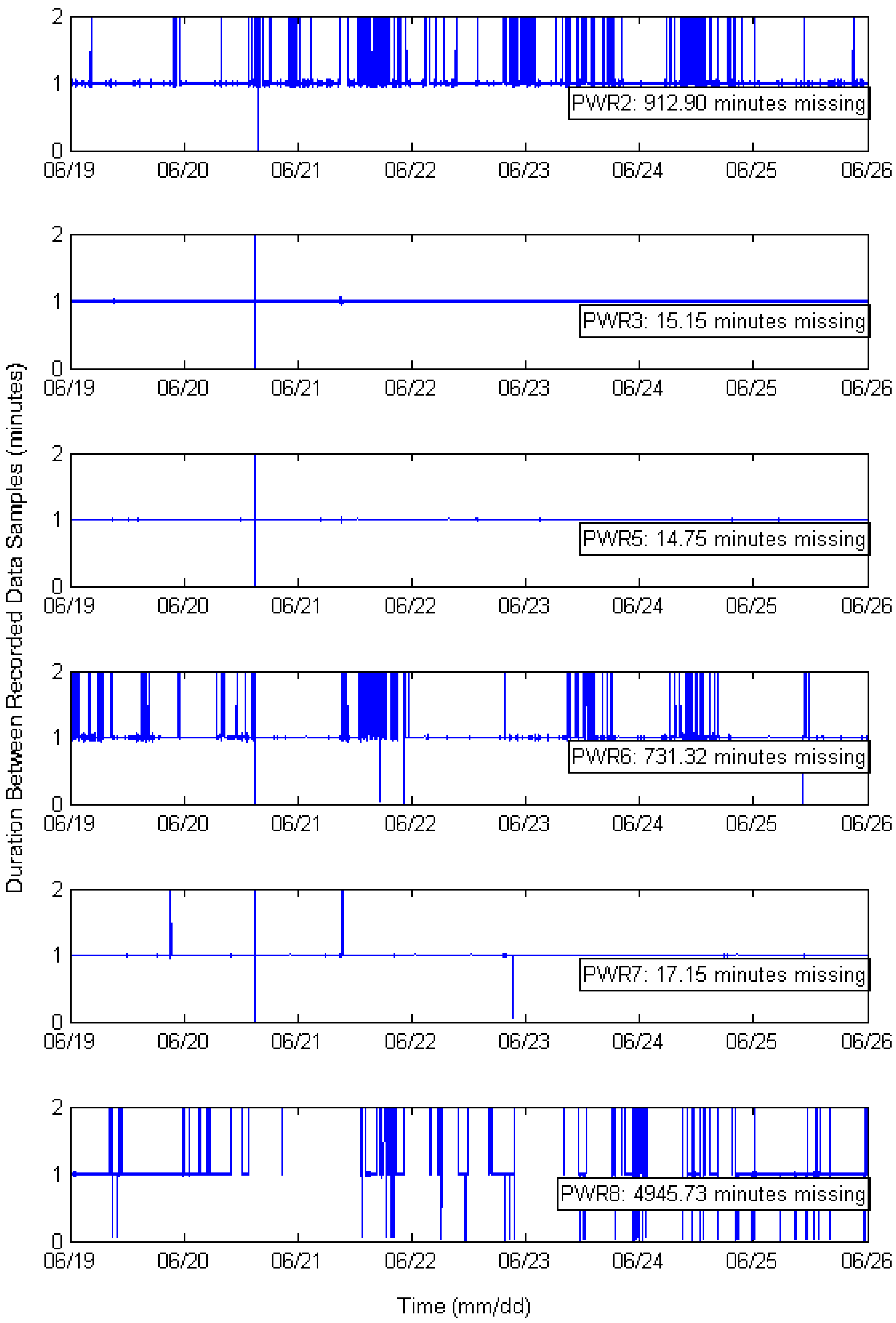


**Figure 12 – Power Data Quality for A June Week in 2011**

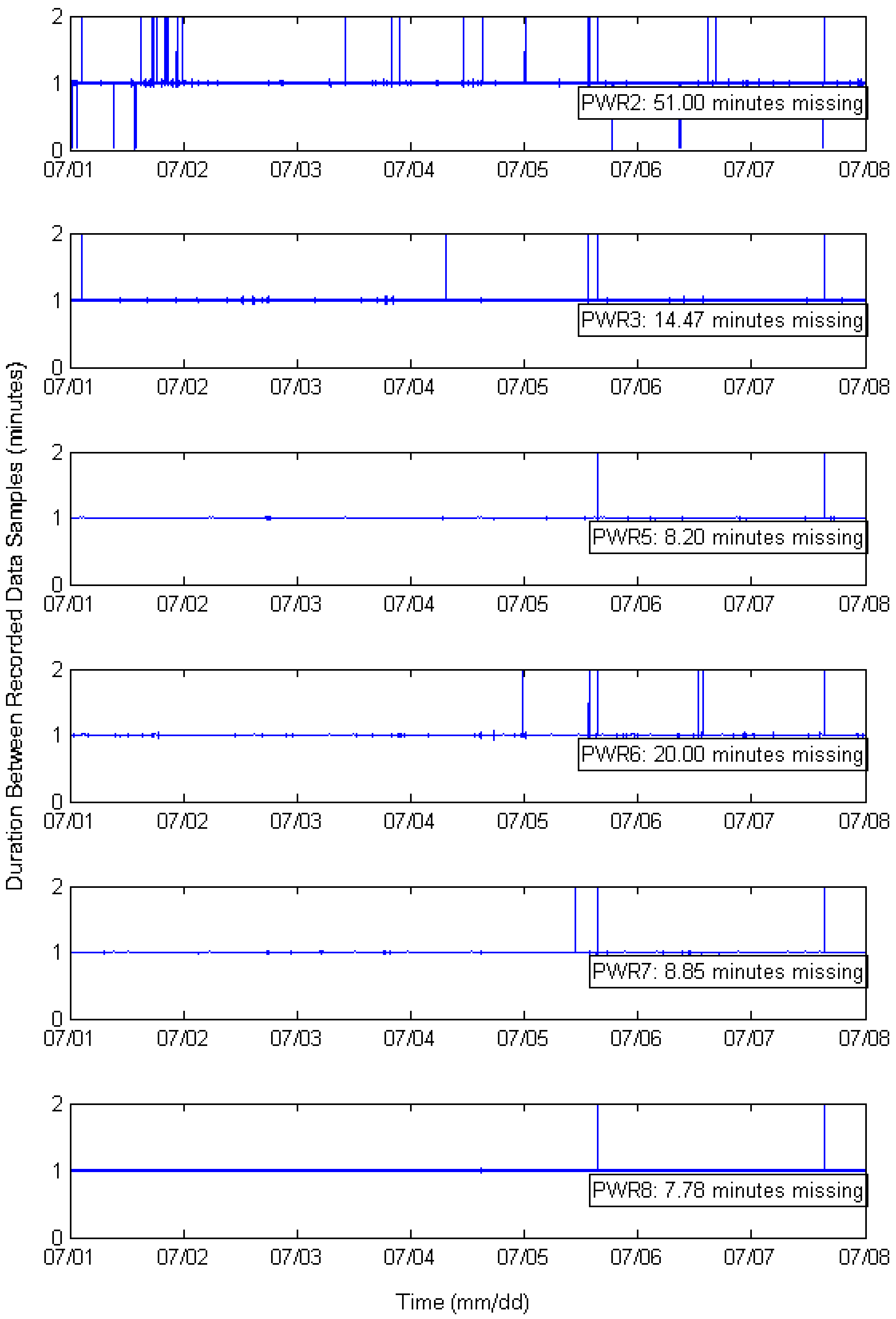


**FIGURE 13 – POWER DATA QUALITY FOR A JUNE WEEK IN 2012**

## *4.1.2 – Pulse Output Conversion*

The WattNode Pulse that was used for these energy measurements is a commercial watt-hour transducer. It works by pulsing a low-voltage output signal (through the use of opto-isolated solid state relays) that is proportional to a given amount of energy flowing through a monitored wire, as measured by a CT. It converts current measurements to energy measurements by assuming a nominal voltage that is determined by the wiring of the device. The proportionality constant that scales the pulses to energy units is dependent on the rating of the CT used during the measured period. To determine the proportionality constant (in watt-hours per pulse) for a given CT, Eqn.(4.1) is used. It is set by the particular WattNode model, which is consistent across each of the installations.

$$WhPP = \frac{Watt\text{-}hours}{pulse} = \frac{CT\ Rating\ (Amps)}{40} \tag{4.1}$$

Since the system was configured to record data at 1-minute intervals, this results in pulses being accumulated, then stored, for each WattNode over a given minute. Understanding that these pulses represent energy measurements over a consistently short duration, an estimation for the average power per pulse over a given minute can be calculated as shown in Eqn.(4.2).

$$\overline{kW}PP = \frac{WhPP \times 60}{1000} \tag{4.2}$$

As Table 2 indicates, three different CT sizes were installed at Villa Trieste: 30 A, 50 A, and 15 A. Using Eqns.(4.1) and (4.2), constant scale factors can be calculated to transform WattNode pulses into both energy and power units, as seen in Table 6.

Equipped with the proportionality constants recorded in Table 6 for scaling WattNode pulses to both energy and power, the power consumption tendencies for these six sensors over the dates indicated in Table 5 were examined. The analysis involved making use of MATLAB's built-in *timeseries* object. This object made it computationally efficient to sync all of the sensors' data to a common time domain, thereby making their 'signal' summations extremely quick. A linear interpolation method was used when resampling each of the sensors' 'signals' to the common one-minute incremented time vector, as demonstrated below by Figure 14.

**TABLE 6 – WATTNODE PROPORTIONALITY CONSTANTS**

| Valid for WNB-3Y-208-P Only | | |
|---|---|---|
| **CT Rating (Amps)** | **WhPP** | $\overline{\textbf{kWPP}}$ |
| 30 | 0.75 | 0.045 |
| 50 | 1.25 | 0.075 |
| 150 | 3.75 | 0.225 |

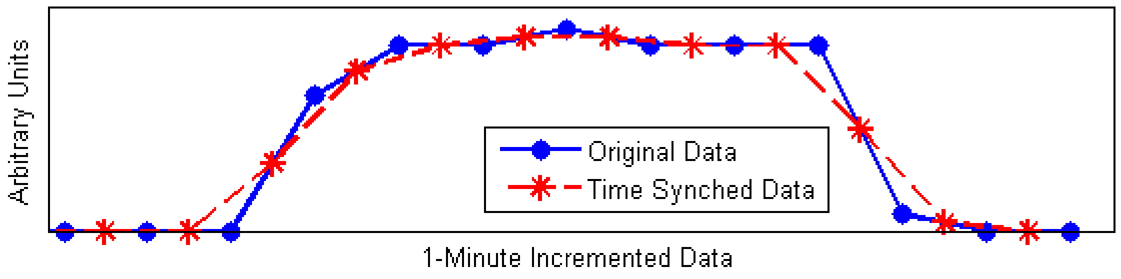


**FIGURE 14 – GENERALIZED LINEAR RESAMPLING METHOD**

In terms of the script's resulting plots, each figure features two subplots; the top plot indicates the total amount of power (averaged over minute intervals) the six homes collectively draw, and the bottom plot reflects the percentage of air conditioners that are simultaneously on at a given time. The bottom subplot also includes the ambient outdoor temperature, so that observations can be made when the peak demand occurs, and how this relates to the maximum daily ambient temperature. The script used for this examination can be found in Appendix B, and its output for the subset of days identified in Table 5 can be seen in Figures 15-20.

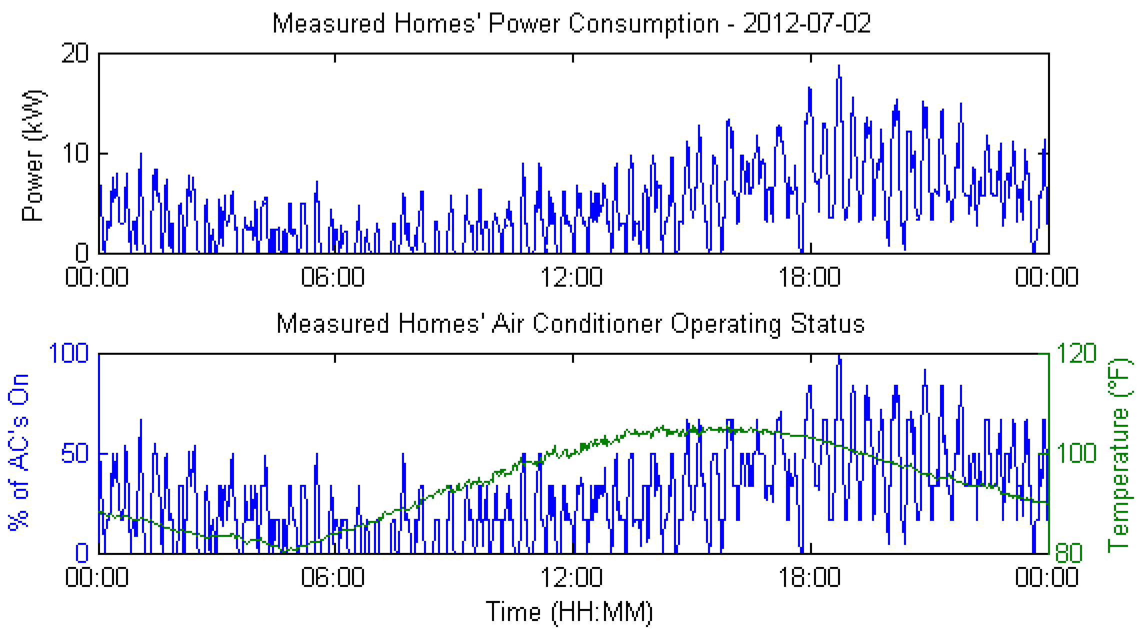


**FIGURE 15 – MEASURED HOMES' POWER CONSUMPTION FOR 2012-07-02**

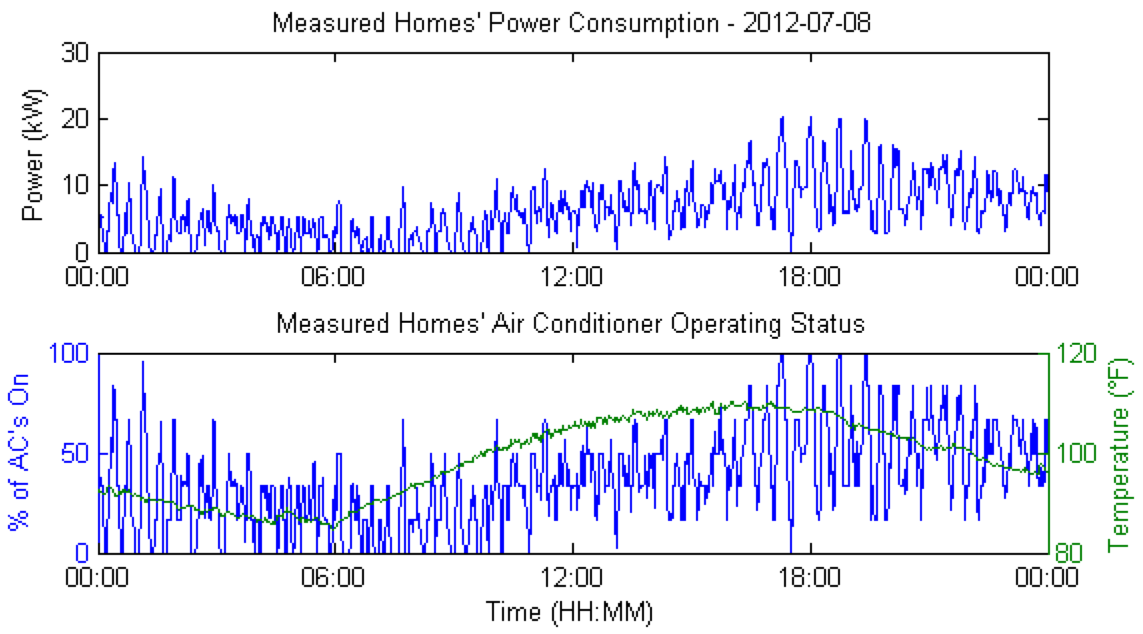


**FIGURE 16 – MEASURED HOMES' POWER CONSUMPTION FOR 2012-07-08**

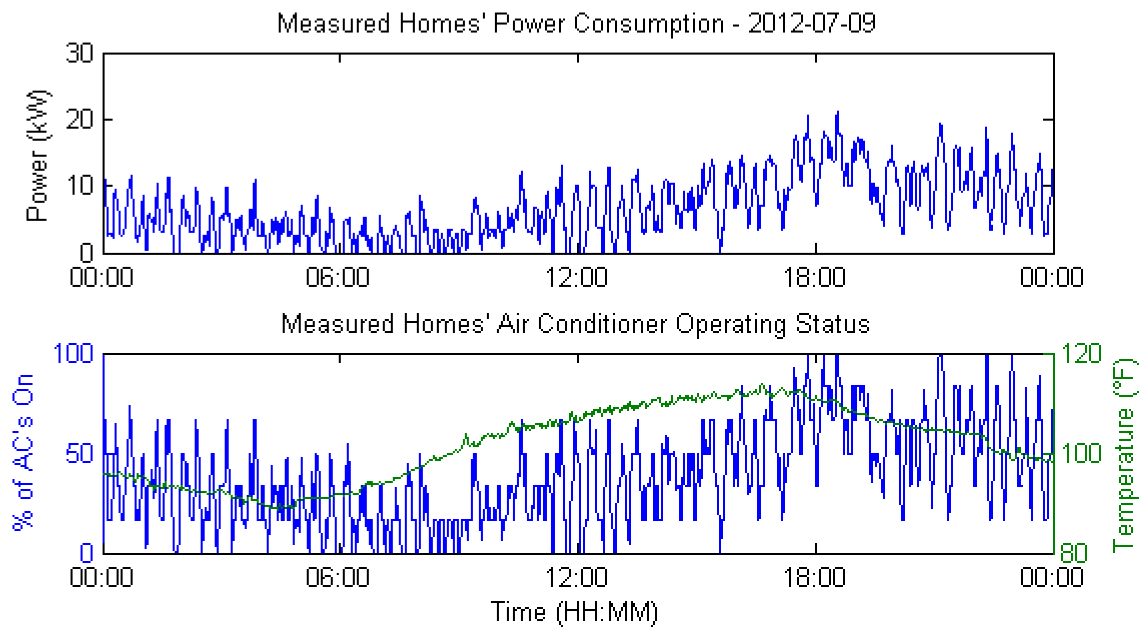


**FIGURE 17 – MEASURED HOMES' POWER CONSUMPTION FOR 2012-07-09**

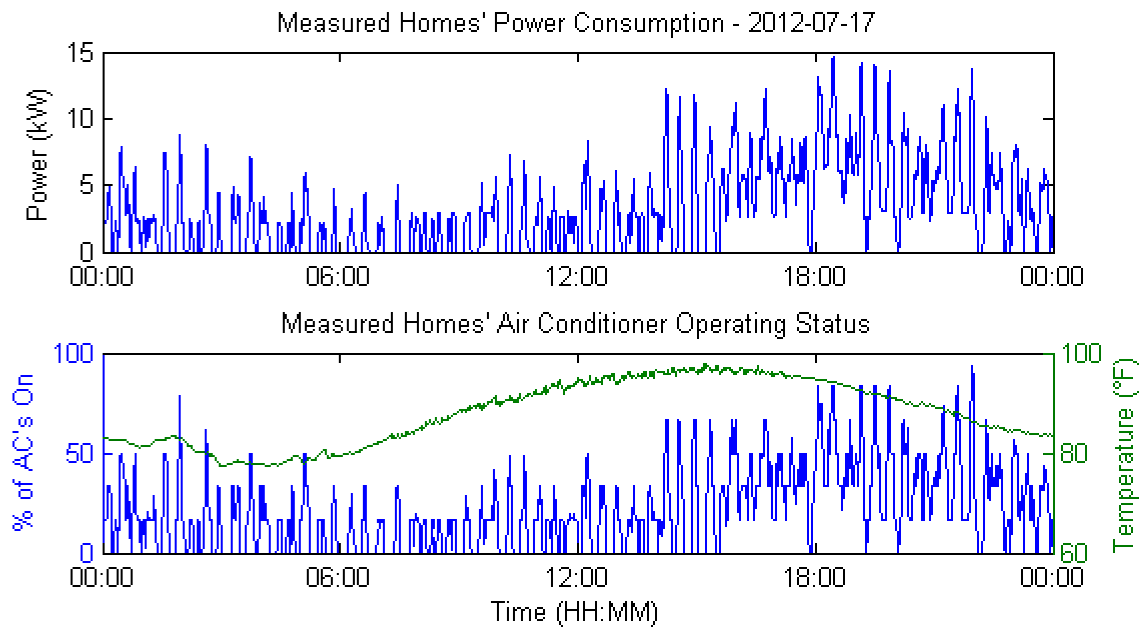


**FIGURE 18 – MEASURED HOMES' POWER CONSUMPTION FOR 2012-07-17**

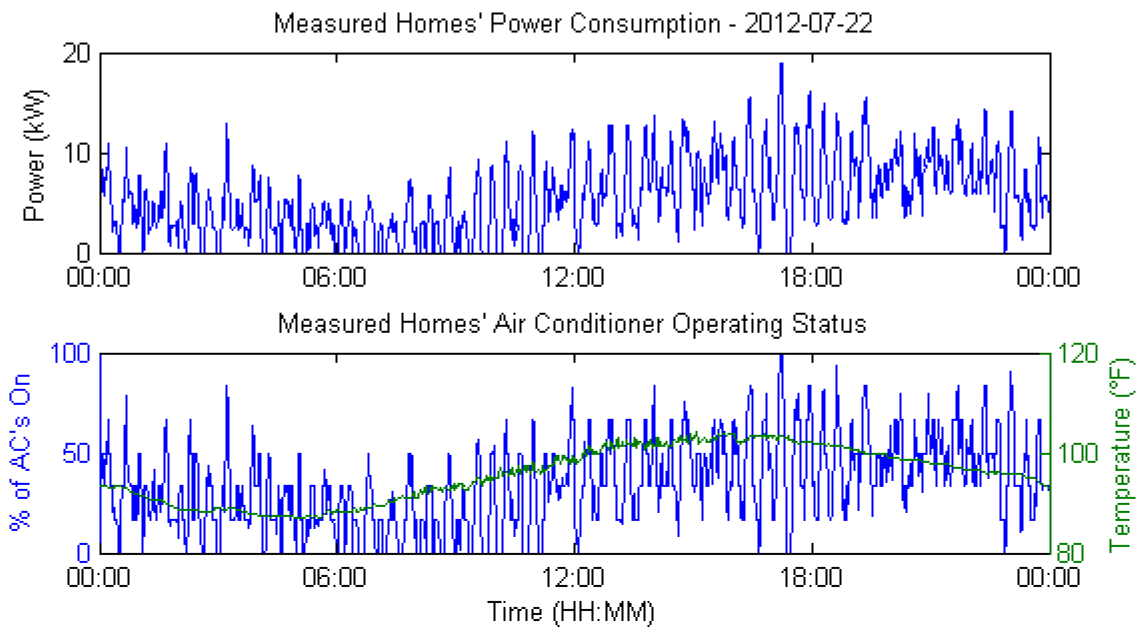


**FIGURE 19 – MEASURED HOMES' POWER CONSUMPTION FOR 2012-07-22**

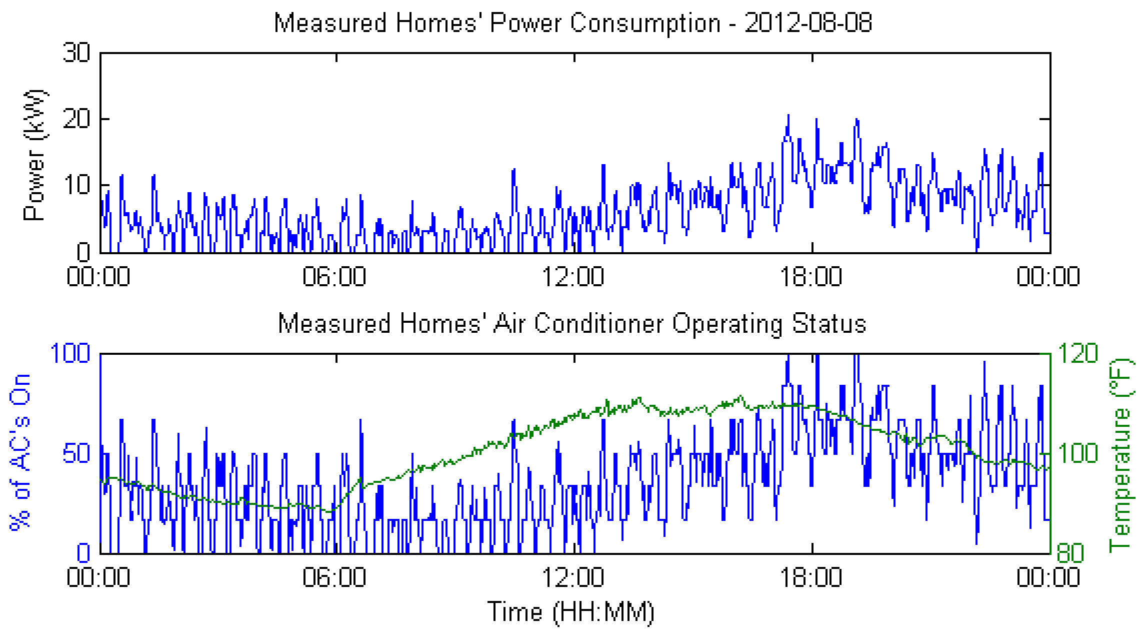


**FIGURE 20 – MEASURED HOMES' POWER CONSUMPTION FOR 2012-08-08**

Immediately, several observations can be made about these plots. First and foremost, the lag between elevated air conditioner usage and the hottest part of the day is several hours in every case. This corroborates the published literature introduced earlier in this paper, and validates the assertion that DR programs are most effectively applied from the late afternoon to early evening hours. Additionally, it is observed that the only day out of the batch that did not feature all air conditioners running simultaneously was the only day in which the ambient temperature never exceeded 100°F (Figure 18); this is why DR events are typically planned in advance when the ambient temperature is projected to exceed a given threshold. The measures are simply otherwise unnecessary for the strictest implementations of DR that only aim to reduce maximum power draws.

Lastly, the plot for July 22nd (Figure 19) is a telling example of the sample's air conditioners inadvertently synchronizing their operation, collectively resulting in an extremely variable load from the utility's point of view. From about 4:00 PM to 8:00 PM the percentage of ACs that are on varies cyclically between 20% to 80% (once hitting 100%). Figure 20 exhibits a lesser degree of the same behavior over the same daily time period, but a baseline of around half of the ACs remain on for the duration, resulting in far less variable electrical demand. Though it will be impossible to test this conjecture given the data available at this time, the author hypothesizes that the uniformity of the homes, given that they were built within weeks of each other by the same contractors in the same housing development, contributes in some way to this synchronization. A collection of less homogenous homes, with significantly different architectural aspects, mechanical equipment, and thermal capacities, may inherently exhibit a tendency toward more regular power demands.

Though there is evidence in all of the plots above, the excessively variable load of July 22nd illustrates exactly how a thermal model may be used to reduce peak demand by shifting the electrical usage of several air conditioners either forward or backward in time by manipulating thermostat setpoints as part of a MPC scheme. Right at about 5:30 PM, it can be seen that all 6 air conditioners are running simultaneously before they all shut off at nearly the same time. This trend continues, though not quite to the same degree, for the next hour and a half. If, for the several minutes leading up to 5:30 PM, each home’s thermal model was able to accurately predict its air conditioner’s cut-in time, UNLV’s central controller could quickly derive a plan for offsetting the operation of several air conditioners by increasing their thermostats’ setpoints and delaying the next air conditioner ON event.

# Chapter 5 – Thermal Model Development

Since the Villa Trieste experiment was not designed specifically for this particular thrust of research, certain aspects of the model development differ slightly from the methods generally followed in the published literature. For example, the Phase I data recording methods included no provisions for ensuring a *complete* minute-by-minute record of the homes' temperature and power consumption be stored in a database; without a local cache on the recording devices, any data that were unsuccessfully transmitted immediately following a minute-long recording period was lost. The incompleteness of this data, as well as the incompleteness of ambient temperature records, complicates the model development process; an alternative source of weather data was required.

Another major difference between the process described herein and the majority of the published literature is that these models have no controlling component–the individual house models can only be validated against already-existing data, and in their current form cannot be used as a MPC; the *impact* of the model-driven control strategy will not be able to be quantified at this time.

Other than the differences described above, much of the model development process was comparable to the methods followed by previous researchers. The general process is as follows:

1. Validation and pre-processing of the data
2. Selection of a model framework with which to work within from the options listed during a discussion of the differing modeling approaches (Chapter 1 – Thermal Response Models and Controllers)
3. Model creation
4. Evaluation of the model's relative accuracy

Borrowing terminology from control theory, this process of building analysis through model creation is known as 'system identification'. Considering the application of this research, 'house identification' is a much more suitable term, as others throughout the literature have mentioned [52].

## 5.1– Input Data Validation and Pre-Processing

Before undertaking the selection of the model, it is useful to understand exactly what data are available, their basis, and where they come from.

### *5.1.1 – Home Sensor Data*

The initial model utilizes a dataset that was created from measurements from the homes of Phase I of the experiment (3.1– Data Collection Hardware, Phase I). This dataset was placed into an online structured query language (SQL) database located on the campus of UNLV. A SQL database provides several computational advantages, including the ability to store/access/and write specific pieces of information to a non-volatile location. In addition, SQL databases are often used in web applications, so although the processes and methods described herein are prototyped and written in MATLAB, they could be relatively easily ported to a web-based language for online use.

An excerpt of this data as it is stored within the database, which has been placed in table 'temperature', has been included in Table 7. Temperatures are measured as dry bulb temperatures and are stored in degrees Fahrenheit (°F), while humidity values are relative humidity (RH) and expressed as percentages. Not every home was outfitted with a humidity sensor.

**TABLE 7 – INTERIOR TEMPERATURE SQL FORMAT EXAMPLE**

| Row Name | id | datetime | tempSensor | temp | hum |
|---|---|---|---|---|---|

| SQL Class | int(11) | datetime | text | float | float |
|---|---|---|---|---|---|
| | 32 | 2011-02-02 13:58:03 | TEMP17 | 76.32 | 0 |
| | 31 | 2011-02-02 13:57:44 | TEMP5 | 69.24 | 0 |
| | 30 | 2011-02-02 13:57:35 | TEMP14 | 73.4 | 0 |

With each of the 14 devices (one per story for each of the seven participating homes) configured to a sample at a rate of once per minute over the nearly 20-month test period, the size of the combined data totals almost 450 megabytes (MB) and nearly 11.5 million rows within the database.

Unfortunately, an examination of this recorded data shows some inconsistencies. Certain sets of data exhibit signs of stepwise measurements for periods of time that could be the result of damaged sensors or misconfiguration. Examples of untrusted, and therefore unused, temperature measurements taken over a random two-day summer period are shown in Figure 21. A sampling of trusted measurements taken over the same time period is shown in Figure 22 for comparison. A summary of the trusted and disregarded dataloggers is documented in Table 8.

One of the advantages provided to this research over the existing literature is that the total number of independent thermodynamic variables can be reduced by one, thanks to the environmental conditions; with almost no relative humidity and extremely low variability in the humidity that is present, the impact of latent heat can be ignored. Only sensible heat is considered.

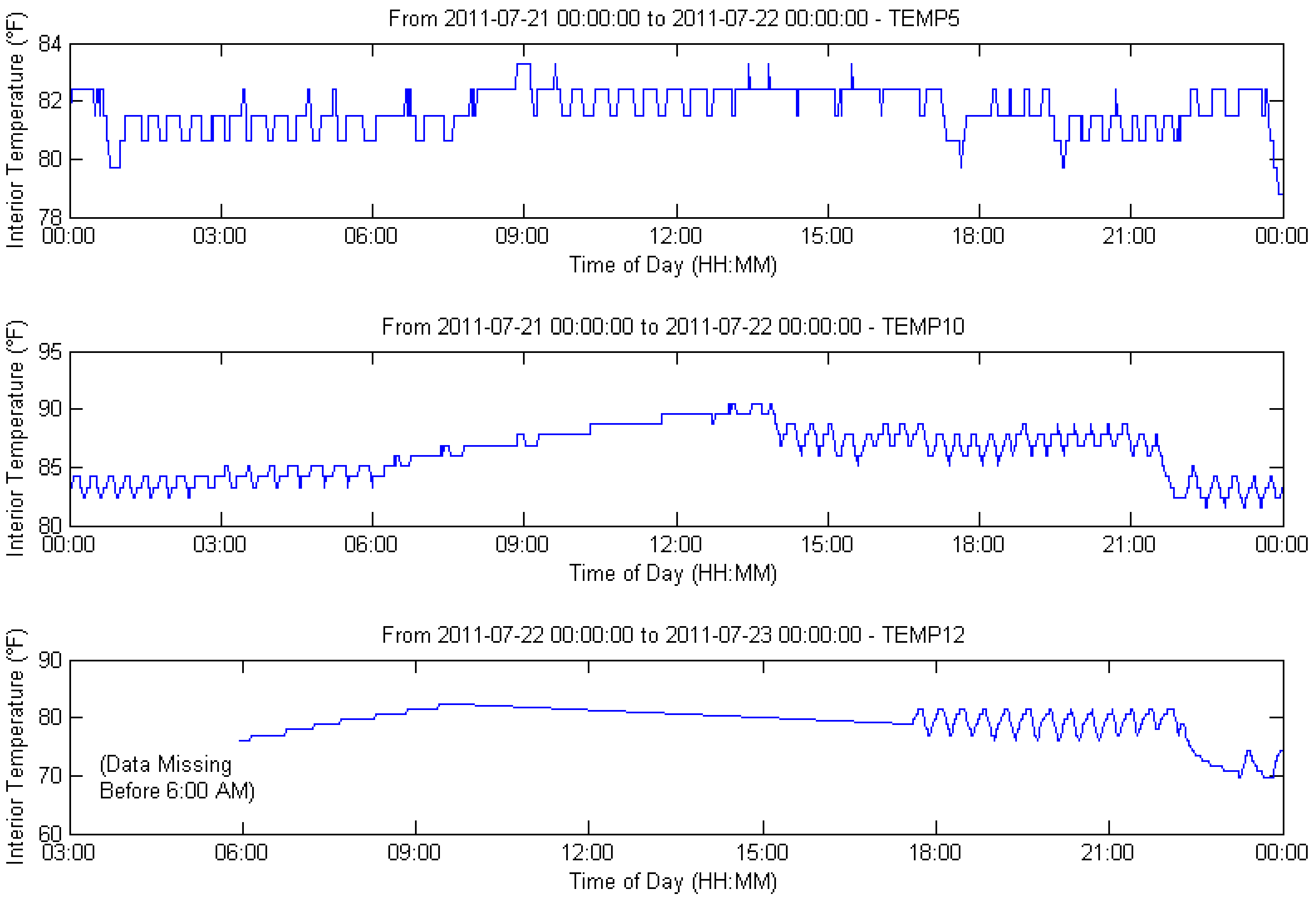


**Figure 21 – Sample of Data Loggers Showing Stepwise Temperature Measurements**

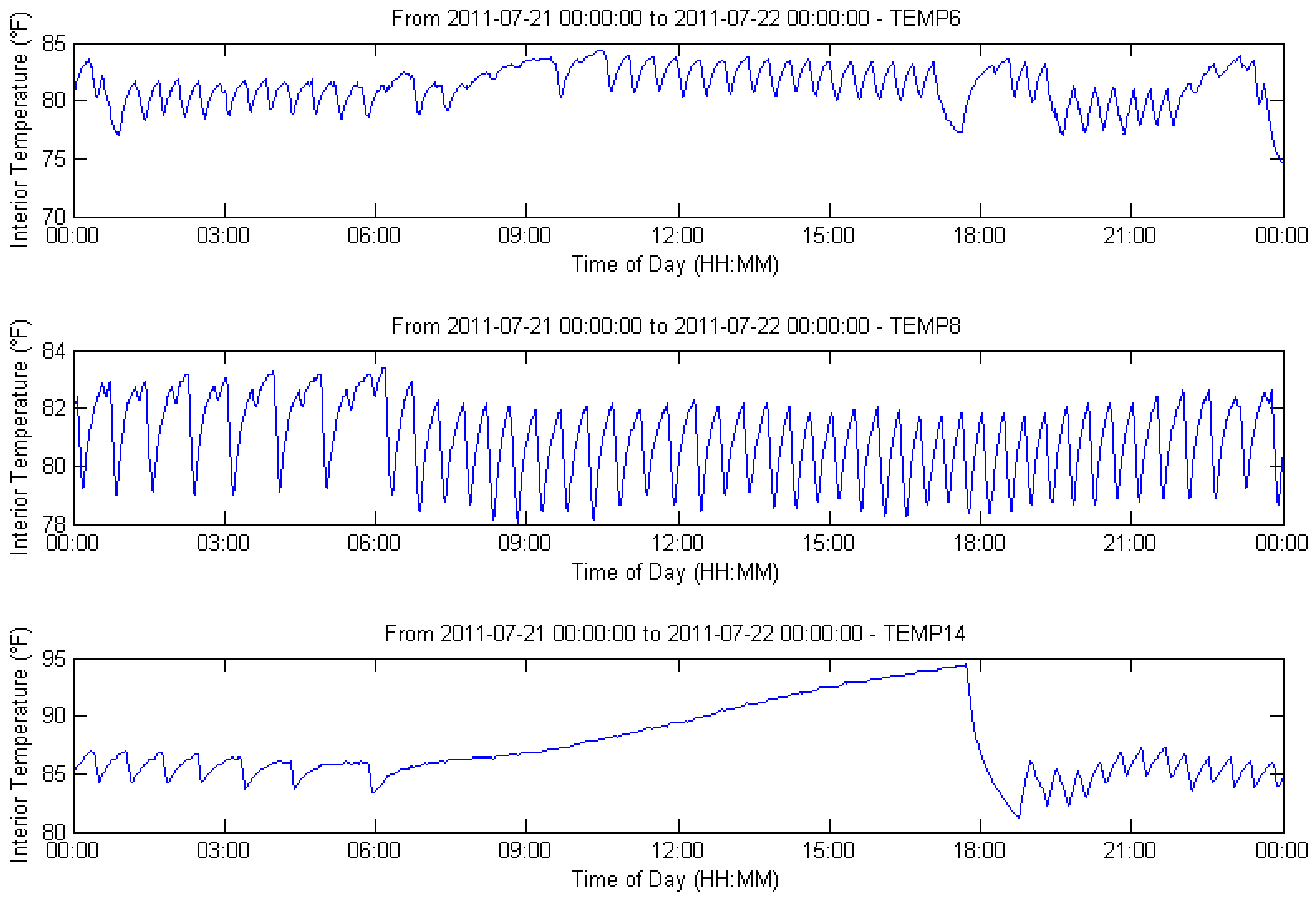


**Figure 22 – Sample of Data Loggers Showing Trusted Temperature Measurements**

**TABLE 8 – SUMMARY OF TRUSTED AND DISREGARDED DATALOGGERS**

| Trusted Sensors | | | Disregarded Sensors | | |
|---|---|---|---|---|---|
| Home | First floor | Second Floor | Home | First floor | Second Floor |
| 1 | - | TEMP6 | 1 | TEMP5 | - |
| 2 | TEMP7 | TEMP8 | 2 | - | - |
| 3 | - | TEMP10 | 3 | TEMP9 | - |
| 4 | - | TEMP12 | 4 | TEMP11 | - |
| 5 | TEMP13 | TEMP14 | 5 | - | - |
| 6 | TEMP15 | TEMP16 | 6 | - | - |
| 7 | TEMP17 | TEMP18 | 7 | - | - |

### *5.1.2 – Ambient Environmental Data*

Since the thermal model is also dependent on ambient conditions, a weather dataset was added to the same database. A temporary weather station was installed at Villa Trieste even prior to Phase I of the experiment, but its data were found to be unusable. An analysis of its recorded information shows that large chunks of data are missing from periods of time that coincide with the first phase experiment (Figure 23). However, extremely localized weather data were not necessarily critical for the purposes of this research. Ambient conditions, especially solar insolation, do not vary substantially over relatively small distances of several miles. Multiple researchers have used weather data that have been collected away from the immediate vicinity of the building they were analyzing [31][44][53]– in some cases, up to 18 miles away [54] – with no reported ill effects. In addition, for the peak shifting controls investigated within this paper to be adopted on a utility-wide scale, it would be impractical to install multiple weather stations at every housing development. A central, or generalized, weather station must be utilized.

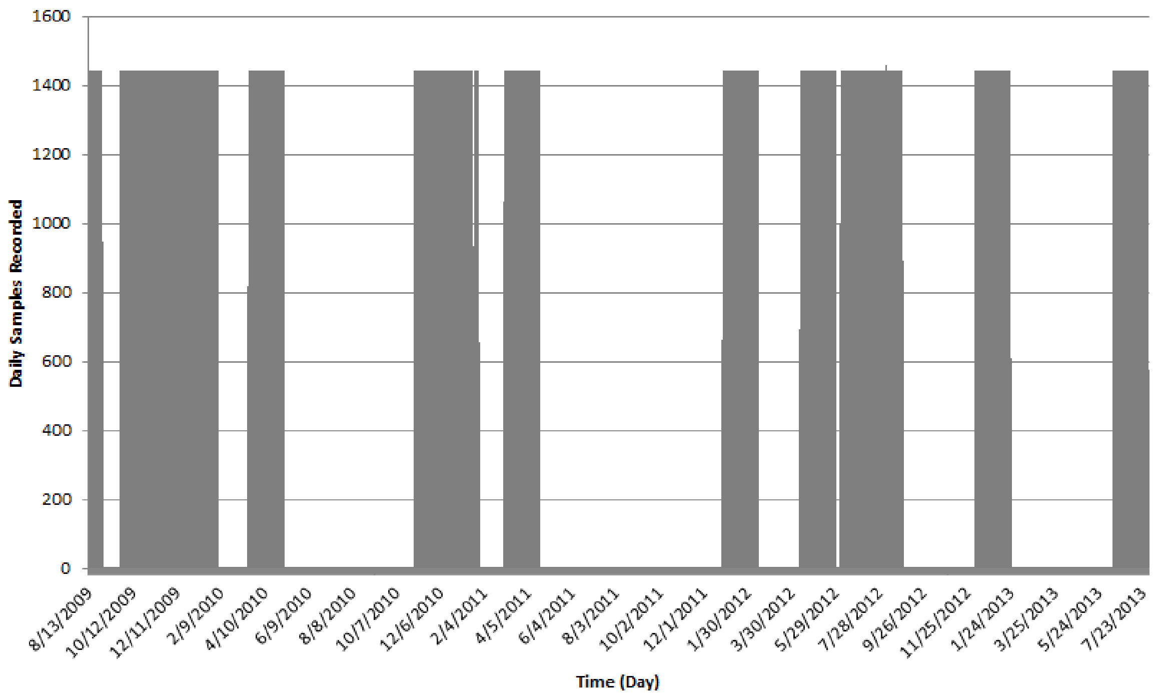


**Figure 23 – Daily Samples Recorded By Villa Trieste Weather Station**

Fortunately, a weather station maintained by the Center for Energy Research (CER) personnel, and administered by the Measurement and Instrumentation Data Center (MIDC) of the National Renewable Energy Laboratory (NREL), is located on the campus of UNLV, a little over 11 miles east of Villa Trieste. It extensively records historical ambient conditions on a per-minute basis dating from the present back to March 16, 2006, and makes this information publically accessible through an assisted database querying interface[4]. Three different solar measurements were extracted from this weather station's historical data (global horizontal irradiance [GHI], direct normal irradiance [DNI], and a calculated diffuse horizontal irradiance [DHI]), in addition to the dry bulb temperature from February 2011 through the end of September 2012. This raw dataset was then processed for formatting and added to a 'weather' table located within the same database as the interior temperature data (Table 9).

[4] https://www.nrel.gov/midc/apps/go2url.pl?site=UNLV

TABLE 9 – AMBIENT CONDITIONS SQL FORMAT EXAMPLE

| Row Type | date datetime | globalHoriz decimal(10,6) | directNormal decimal(10,6) | diffuseHoriz decimal(9,6) | ambientTemp decimal(7,4) |
|---|---|---|---|---|---|
| | 2011-02-01 06:55:00 | 10.724 | 4.15061 | 10.5769 | 48.722 |
| | 2011-02-01 06:56:00 | 11.4441 | 32.2975 | 10.2012 | 48.74 |
| | 2011-02-01 06:57:00 | 13.0352 | 148.092 | 6.8829 | 48.776 |

Each of the solar measurements is specified in terms watts per square meter (W/m$^2$), and temperature is expressed as degrees Fahrenheit (°F). The pyranometer responsible for the solar measurements is a Kipp & Zonen CM3 and is accurate within 25 W/m$^2$ when properly calibrated. The resistance temperature detector (RTD) is a weatherproof Young 41342 Platinum RTD that is accurate within ±0.3°C at 0°C. All sensors are assumed to be accurate.

## 5.2 – Model Output Requirements

Although this research does not explicitly include the development of the *controls* portion of a model predictive controller, it is the intention of the author that this predictive model be extended to someday control the thermostats that were described in Chapter 3. As Chapter 4 illustrated, this thermal predictive model could ultimately be used to decrease the collective variability of air conditioner power demands over a set of homes. Knowledge of exactly how long particular houses would take to heat up or cool down to a given threshold, or setpoint, would allow researchers to derive a cost function for optimizing both thermal comfort and electrical stability.

Considering the input data available, the model's output should not simply be a prediction of how long a particular heating or cooling cycle will last, but rather the *rate* at which the house's interior temperature changes. By deriving a rate, rather than a strict duration, the length of a given heating or cooling cycle can be calculated based on different future temperatures (optimized setpoints).

Given the scope of this research, and the historical datasets that are available, the creation of an online model would not be feasible. An offline model that was updated nightly would offer the benefit of a semi-dynamic training dataset, without the computational encumbrance of an online model's real-time optimization. If UNLV's server was able to train the model at night and simply *use* the input-output model during the day, it should computationally be able to control a relatively large collection of thermostats.

## 5.3 – Model Selection

Quite a few different methods for creating thermal models were discussed within Chapter 2. Almost all of these different modeling approaches have been shown to be successful under certain circumstances. However, quite a few of these thermal models were conceived for use on large commercial facilities, and even more were developed to investigate prolonged energy use over an extended period of time rather than to predict transient thermal responses. Many of these models would not be practical, in a broad sense, for distributing peak shifting over multiple residential buildings. Thus, the criteria for model selection are listed below.

1. The thermal model must be self-adaptive and make use of the empirical dataset that is available.
2. It should require extremely limited human interaction, whether that be up-front data entry or maintenance.
3. It must be capable of making short-range transient predictions rather than longer trend energy consumption estimations.

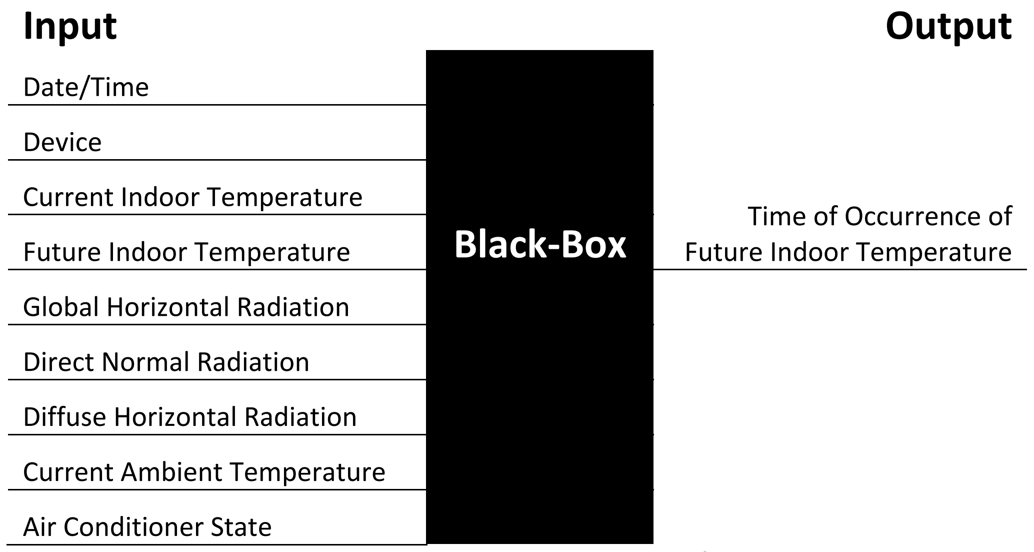
A summary of the potential candidates for model framework selection are listed in Table 10. Forward models are of no use in this application, nor are white-box data-driven models. Several black-box techniques could potentially be useful, but ARMAX and OE/Box-Jenkins methods are not conducive to the type of analysis that needs to be done. The two most viable methods for investigation are regression and neural networks.

**TABLE 10 – MODEL SELECTION DECISION**

| | | Forward | Data-Driven | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | \| | White-Box | Black-Box | | | | Grey-Box |
| | | \| | \| | Regression | ARMAX | OE/Box-Jenkins | NN | \| |
| Criteria | 1 | | | X | X | X | X | X |
| | 2 | | | X | X | X | X | |
| | 3 | | | X | | | X | |

## 5.4 – House Identification

Regardless of the specifics of the black-box, whether it be a regression method or a neural network model, the inputs and outputs of the system that are *available* to be used do not vary. In this sense, the black box analogy is very fitting (Figure 24). Mathematically, this black-box can be conceptualized as a simple function Eqn. (5.1).



Input
Output
Date/Time
Device
Current Indoor Temperature
Future Indoor Temperature
Global Horizontal Radiation
Direct Normal Radiation
Diffuse Horizontal Radiation
Current Ambient Temperature
Air Conditioner State
Black-Box
Time of Occurrence of Future Indoor Temperature

**FIGURE 24 – BLACK BOX ANALOGY INPUT/OUTPUT**

$$t^{*} = function[T_i(t), T_i(t^{*}), I_{GHI}(t), I_{DNI}(t), I_{DHI}(t), T_a(t), \theta] \tag{5.1}$$

Where: $t^{*}$: Future time when the interior temperature reaches a given threshold (HH:MM)

$t$: Current time (HH:MM)

$T_i$: Interior temperature (°F)

$I_{GHI}$: Global Horizontal Irradiance during $t$ (W/m$^2$)

$I_{DN}$: Direct Normal Irradiance during $t$ (W/m$^2$)

$I_{DHI}$: Horizontal Irradiance during $t$ (W/m$^2$)

$T_a$: Ambient dry bulb temperature (°F)

$\theta$: Air Conditioner State (ON or OFF)

### *5.4.1 – Air Conditioner State Estimation*

There is one input from the black-box schematic illustrated above that is significantly different from the others – the on/off state of the air conditioning system. Rather than being a numerical measurement, it is a discrete state that can be determined from the air conditioners' power consumption measurements, as introduced in Chapter 4. Two different methods are presented for converting granular minute-by-minute data to an event-based record of the air conditioners' status.

### *5.4.1.1 – As Recorded by Current Transducers*

Chapter 4 introduced the methods by which the recorded WattNode pulses could be converted to either power or energy units. However, the examination made during said chapter was focused on multiple homes' simultaneous electric consumption over the course of a day rather than ascertaining an individual home's HVAC state to be used as an input for the thermal model. When examining the power data at the daily scale, the minute-by-minute resolution is adequate. However, when attempting to predict transience, a more accurate record of exactly *when* the AC state transitions occur is desired.

Figure 14 visually presented a generalized AC cycle, while Table 11 shows an actual excerpt of data from the script included in Appendix B (data is selectively queried from the database). Both of these inclusions indicate the steady-state nature of the air conditioner when it is in operation.

**TABLE 11 – AN EXCERPT FROM THE RECORDED POWER DATA**

| Device | Timestamp | Pulses |
|---|---|---:|
| 'PWR2' | '2012-07-02 12:12:46.0' | 0 |
| 'PWR2' | '2012-07-02 12:13:46.0' | 9 |
| 'PWR2' | '2012-07-02 12:14:46.0' | 13 |
| 'PWR2' | '2012-07-02 12:15:46.0' | 13 |
| 'PWR2' | '2012-07-02 12:16:46.0' | 14 |
| 'PWR2' | '2012-07-02 12:17:46.0' | 13 |
| 'PWR2' | '2012-07-02 12:18:46.0' | 13 |
| 'PWR2' | '2012-07-02 12:19:46.0' | 13 |
| 'PWR2' | '2012-07-02 12:20:46.0' | 6 |
| 'PWR2' | '2012-07-02 12:21:46.0' | 0 |

This tendency is consistent with any constant-speed compressor-driven air conditioner [55], though the power required during a given cycle is also dependent on the temperature of the refrigerant that enters the compressor, which is in turn dependent on the ambient temperature. This fact can be observed by an examination of Figures 25-27, which plot ambient temperature versus the average power draw per AC cycle. Each of these plots confirms that air conditioner efficiency is dependent on the ambient temperature. As the ambient temperature increases, so does the power–in a nearly linear fashion.

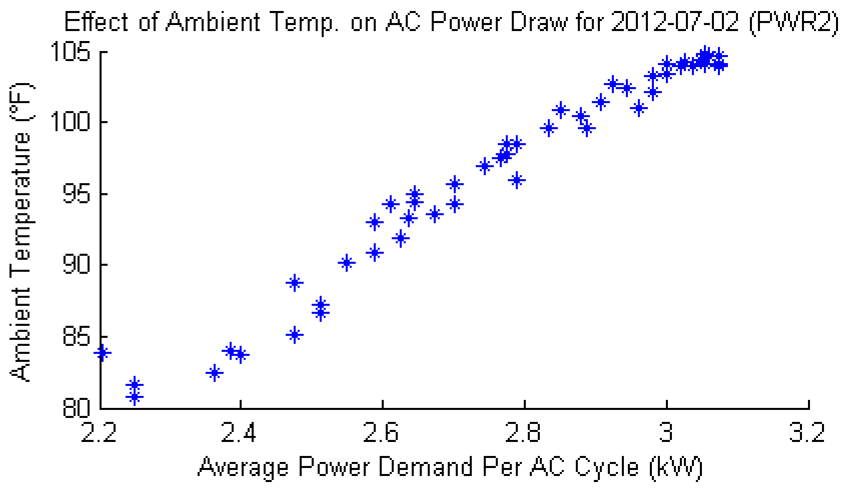


**FIGURE 25 – AMBIENT TEMP. VS. AC POWER CONSUMPTION FOR PWR2 ON 7/2/2012**

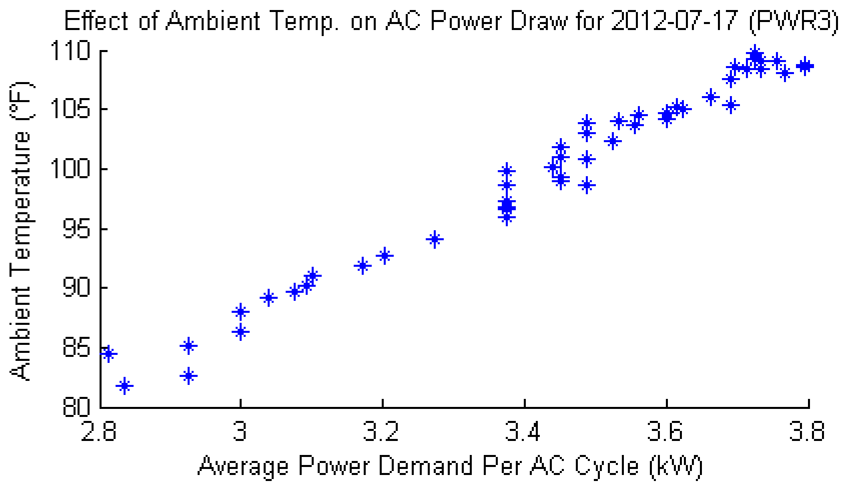


**Figure 26 – Ambient Temp. vs. AC Power Consumption for PWR3 on 7/17/2012**

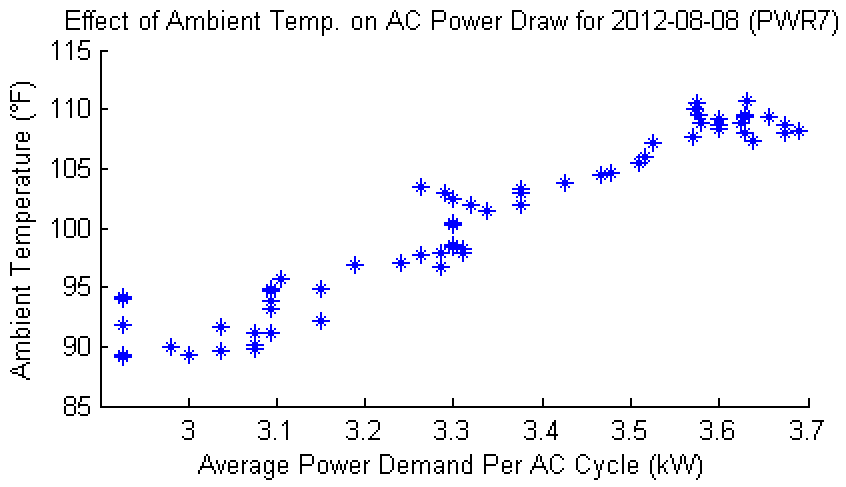


**Figure 27 – Ambient Temp. vs. AC Power Consumption for PWR7 on 8/8/2012**

Although operating in steady-state, this difference can be attributed to the sample rate of the recordings not being synchronized with the pulse rate of the WattNode. To identify the exact power demand during this AC-on cycle, the total number of pulses can be averaged over the sample period. In this case, that is 79 pulses over a period of 6 minutes, resulting in a pulse every 4.557 seconds. If this same rate is applied to the shoulder periods, it can be calculated that the AC turns on approximately 19 seconds into the first 'shoulder' period, and it turns off around 27 seconds into the second 'shoulder' period. This method is further described by Eqns.(5.2) through (5.4) below.

$$t_{ON} = t_{i+1} - P(n_1) \times s_p \tag{5.2}$$

$$t_{OFF} = t_{x-1} + P(n_x) \times s_p \tag{5.3}$$

$$s_p = \frac{60(x-2)}{\sum_{i=2}^{x-1} P(n_i)} \tag{5.4}$$

Where: $n_i$: The time sample corresponding to the 'shoulder' period when the AC turns ON

$n_x$: The time sample corresponding to next 'shoulder' period when the AC turns OFF

$P(n)$: Number of a pulses as a function of a given time sample

$t$: Timestamp

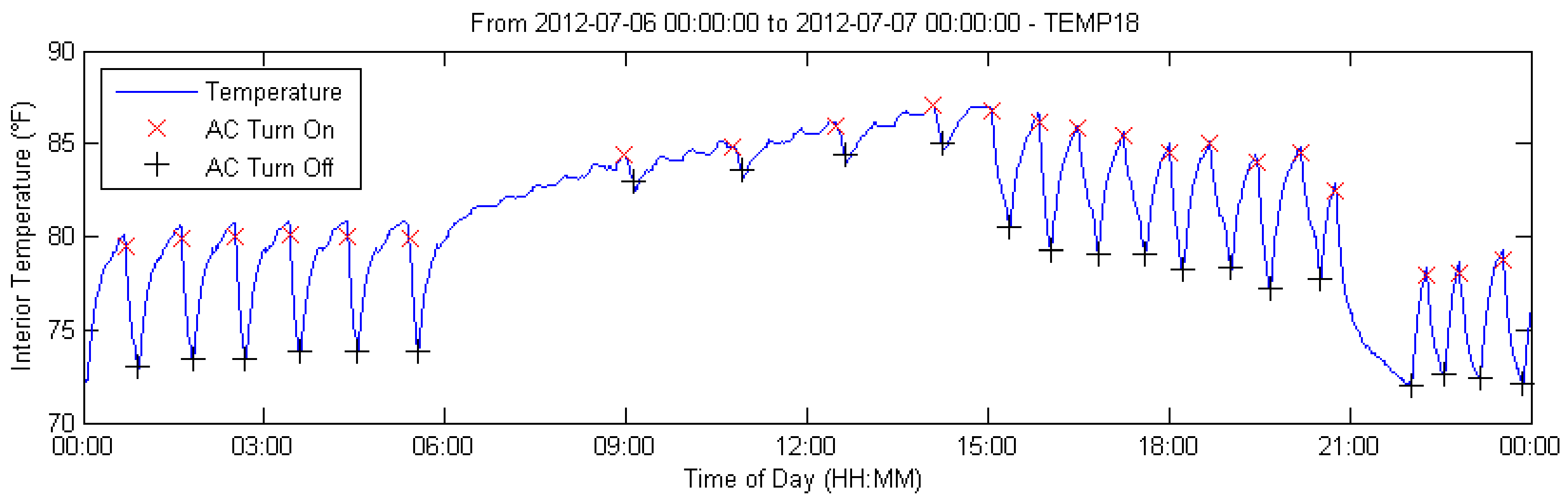
$s_p$: Amount of time between power pulses (seconds)

The above equations were utilized by the program included in Appendix E. This program was written to produce a plot, for each temperature sensor on a given date, that would illustrate exactly when the corresponding air conditioner was turning on and off, and how the AC's operation affected the home's measured temperature. A selection of these plots has been included in Figures 28-29.

Figure 28 shows a couple examples in which the AC-event-transition timestamps derived from WattNode pulses corresponded well with the homes' transient temperature changes. On the other hand, Figure 29 shows several examples of similarly derived timestamps not particularly matching up well with the temperature swings. Of these poor examples, both plots include temperature sensors from within the same house. The leftmost plot, which includes TEMP7's measurements, shows the AC-on transitions correlating with the temperature profile well enough, but also appears to consistently show the measured temperature continuing to decrease for several minutes after the AC-off transition. The rightmost plot, which shows TEMP8, illustrates the exact opposite behavior–AC-off transitions at the appropriate times, and AC-on transitions that occur after the temperature has already begun to decrease.

Of note, TEMP7 is the downstairs temperature sensor, and the TEMP8 sensor is located upstairs. The majority of these offsets can be used to infer different mechanical and physical characteristics of the home. The temperature sensor located on the first floor is likely very close to an air register–the temperature of TEMP7 trends downward very quickly after the air conditioner is turned on. After the AC is turned off, the cooled air that is already within the home's ductwork continues to influence the temperature of the room for a short period of time. This amount of time is longer during the night, and shorter during the day–indicating the impact of ambient temperature and solar irradiance. The second floor is comparable in this regard; the temperature decreases momentarily after the AC is turned off. The temperature of this zone consistently begins to trend upward again more quickly than the first floor, suggesting that the second floor is either more susceptible to the influences of the ambient temperature than the first floor, or its general thermal mass is smaller.

The only troubling offset can be seen in the early morning time period of TEMP8. The plot suggests that the temperature of the second floor decreases up to a full degree before the AC switches on. Though not included here, the rest of the generated plots show the same general behavior.  An examination of the data shows that, in several instances, the fan control unit turns on several minutes before the AC, but not consistently for all hours of the early morning. This is the only temperature sensor that shows this level of disconnect between the AC turning on and the temperature of the zone decreasing; further investigations are necessary.



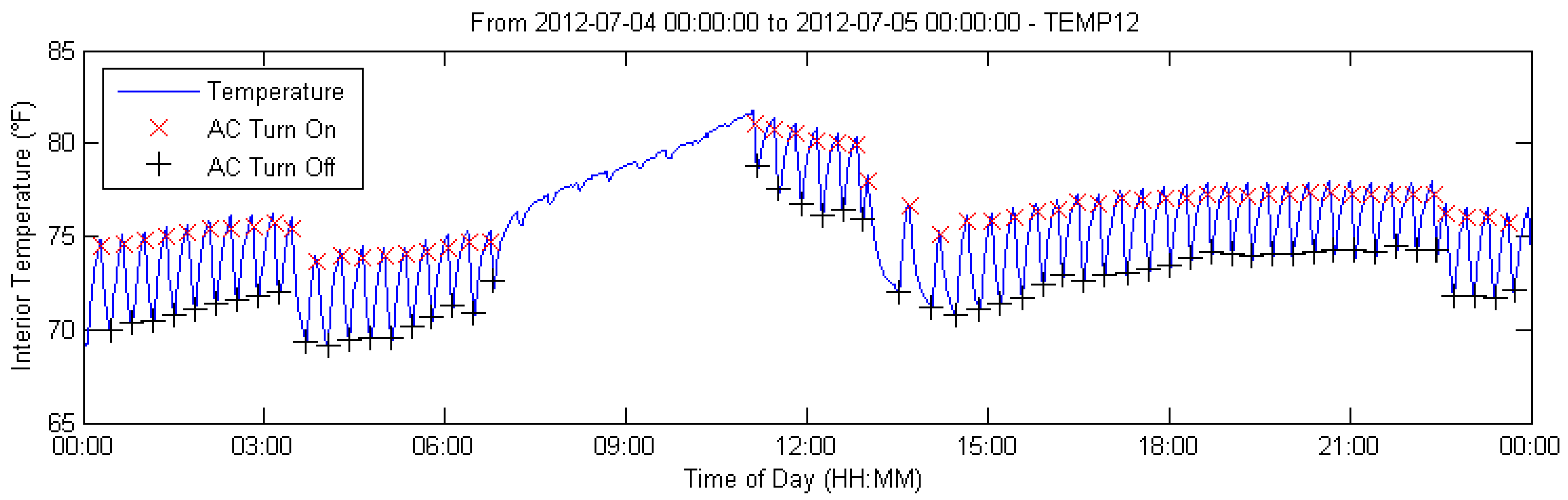
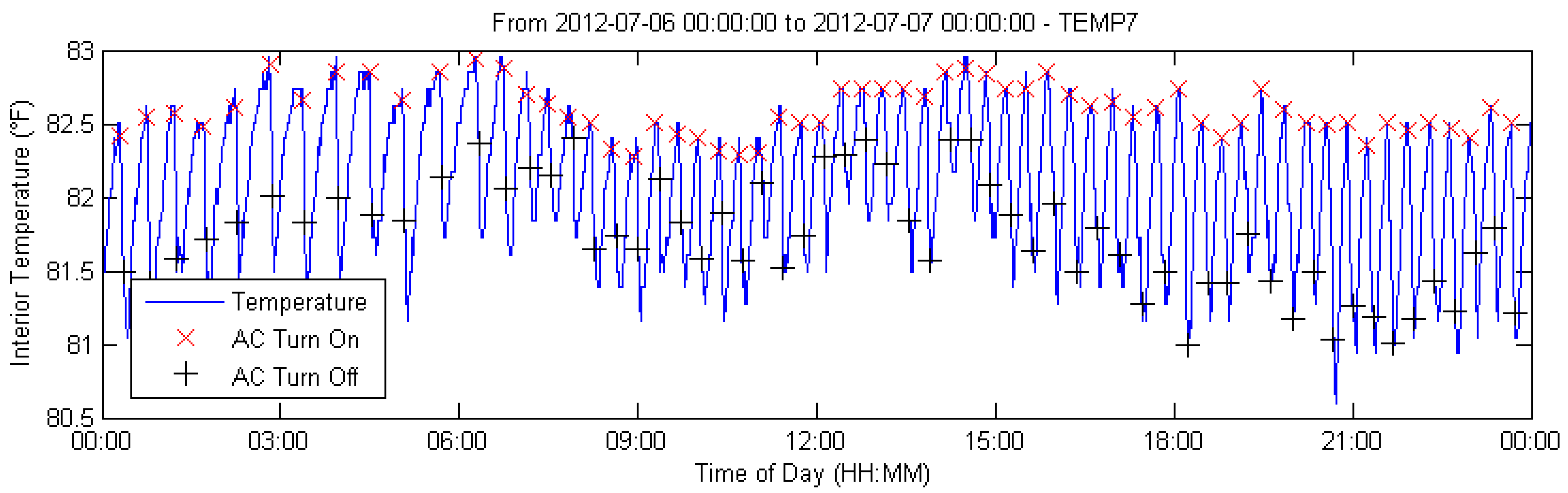


**Figure 28 – Accurate Calculations of AC Event Transitions and Interior Temperatures**



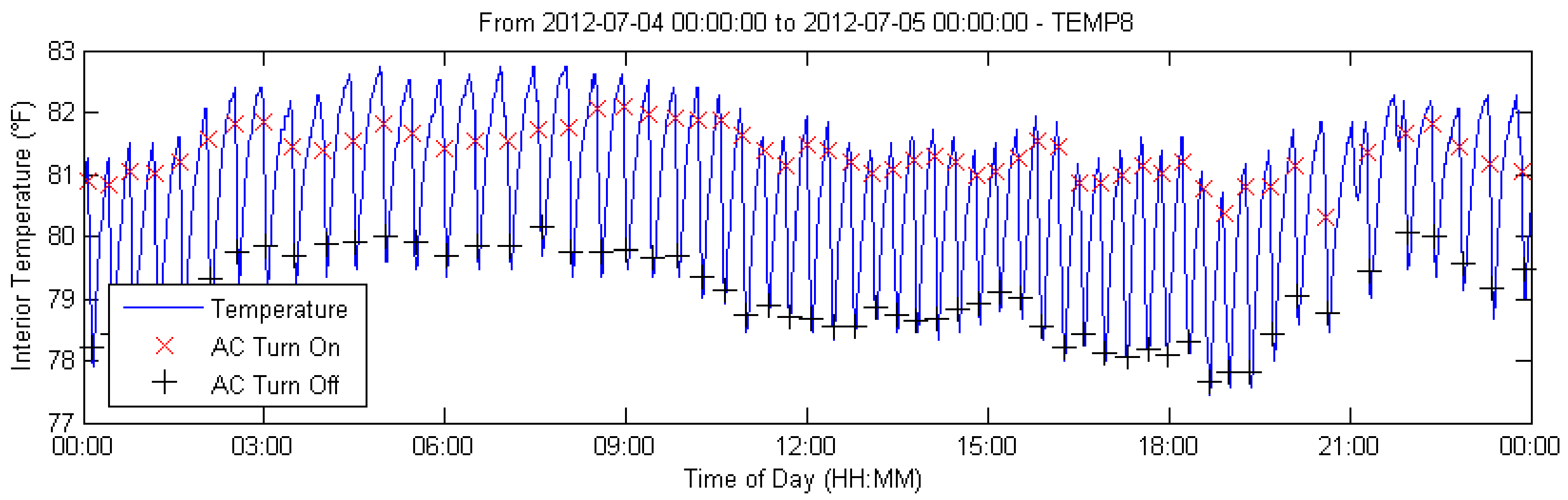


**FIGURE 29 – INACCURATE CALCULATIONS OF AC EVENT TRANSITIONS AND INTERIOR TEMPERATURES**

Further processing of the data could result in estimates for the air conditioners' energy consumption profiles. Similarly, trendlines could be generated for each individual air conditioner's efficiency and its sensitivity to the ambient temperature. Though not a primary objective of this paper, the author hypothesizes that this sensitivity to the ambient temperature could be monitored over the lifetime of the air conditioner and potentially be used as a fault detection method for identifying when the unit's performance begins to trend negatively–a possible sign of low refrigerant levels or some sort of mechanical issue.

#### *5.4.1.2 – As Inferred by Temperature Measurements*

The method outlined above, in 5.4.1.1, for determining AC status based on current transducer measurements is the method generally followed by the relevant literature. Although accurate, this method is costly in several senses. The CTs, as well as their supporting hardware, are affordable individually, but can be quite expensive when scaled up for housing-development-sized installations. Additionally, these line-voltage devices require the services of a knowledgeable electrician when they are installed, and may also require an additional enclosure be installed near a home's utility box. In an attempt to potentially mitigate these costs for future research, a method was investigated for inferring the cyclical operation of a home's air conditioner based on just temperature sensor readings, since temperature sensors are extremely inexpensive.

Referring to just the first day of the range that was examined during 5.1.1 – Home Sensor Data (July 21, 2011), Figure 30 shows the second floor measured temperature response for house 7. This sample shows some very clear air-conditioning behavior, with the following observations dictating the formulation of the AC state estimation:

1. From around 10:00 PM to 6:00 AM, the thermostat is set to ‘night’ mode where the temperature stays between 73°F and 79°F. From an understanding of how thermostats behave (3.4 – Thermostats), the ‘night’ setting is likely 77°F.
2. From 7:00 AM to 2:30 PM, the response suggests a setback temperature is in effect.
3. The extended period of time required for the house to heat up during the setback temperature suggests that the home *may* be equipped with a two-stage air conditioning system.
4. From 2:30 PM to 8:00 PM the thermostat is likely set to ‘evening’ mode, where the air conditioner maintains a temperature between 79°F and 84°F, implying a thermostat setting of 82°F.
5. From approximately 8:00 PM to 10:00 PM the slope of the downward trending temperature seems to shallow significantly, perhaps once again, suggesting two-stage air-conditioning performance.
6. The temperature response resembles a sawtooth signal. Indeed, there appears to be very little thermal inertia within the house. This is consistent with the remarks made about differences between large commercial facilities and residential homes. As soon as the constantly downward sloping temperature reaches a particular point, the house appears to tend to warm back up immediately. The same is true for the upward temperature trends; the air conditioner appears to have an immediate impact on the temperature of the room in which the sensor is located.

Each pronounced downward trend indicates the operation of the air conditioner. From the limits suggested by points 1 and 4, noise and transient downward trends, as seen around 11:30 AM and 2:00 PM can be distinguished from active cooling by filtering out downward $\Delta T$ trends that are less than a given threshold. This condition is sufficient for identifying *when* the air conditioner is operating, but it doesn't provide enough context for determining exactly when the *transition* from on to off, and vice versa, occurs. Point 6, from above, implies that the identification of the signal's local maxima and minima would indicate the beginning and end times of the air conditioner's operation.

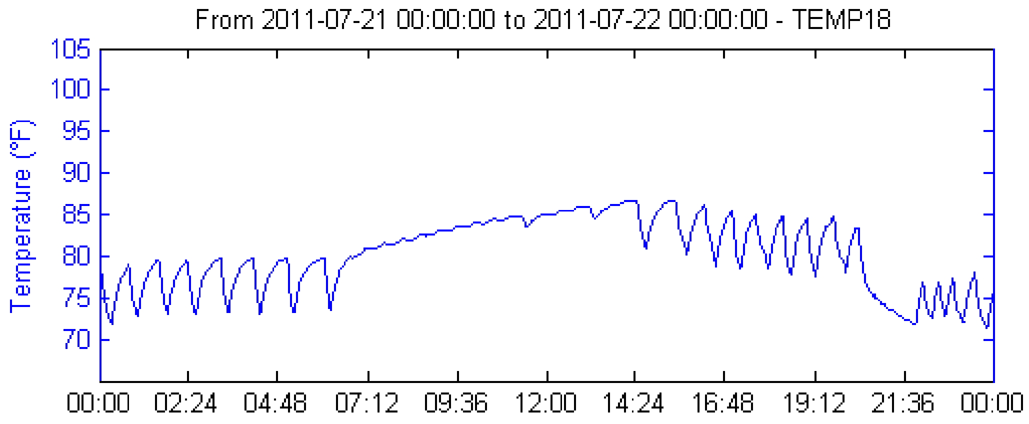


**FIGURE 30 – TYPICAL MEASURED DAY TEMPERATURE FOR DEVICE TEMP18**

The script that was written to produce plots of the internal temperatures versus AC cycling (as determined by the current transducers) was modified and expanded to also estimate the approximate operation of the air conditioner (Appendix C) through a sliding-window approach. For each time sample $t$, a window of length $\omega$ corresponding to $S_{t-(\omega-1)}$ is evaluated in several ways. This continues for $t = \omega, \dots, n$. By method of trial and error, a window size of 15 minutes provided an excellent level of accuracy without being too computationally intensive.

The algorithm begins by assuming the status of the AC is off. Whenever the AC is off, the algorithm's objective is to identify when it turns back on. Similarly, the priority of the algorithm when the status is on is to identify when it transitions to being back off. The behavior of the air conditioner is defined by Eqn.(5.5).

$$\theta = \begin{cases} 1, & \text{AC-on} \\ 0, & \text{AC-off} \end{cases} \tag{5.5}$$

Whenever the AC is off, the script looks to identify Eqn.(5.6). That is, when a temperature drop across the window exceeds the threshold $\Delta T_{AC,ON}$ (Figure 31), it registers the AC as being on, and scans backward through the window looking for a local maximum that indicates exactly *when* this transition occurs. It then immediately begins looking for the AC-off transition. This is accomplished by examining each window for a local minimum. If the local minimum exists at $t$, the algorithm assumes that $t + 1 < t$ and moves on to the next window. This continues until the minimum is identified and recorded, and the process starts over again, attempting to identify the AC-on transition. Several more considerations are programmed in for instances where “plateaus” exist and transience impacts the calculations, but the algorithm ultimately makes a record of all the AC-on/AC-off transitions and when they occur within the given time range. These generalized temperature trends can be plotted ,along with the measured interior temperature for visual inspection (Figure 32).

$$T(t) - T(t - [\omega - 1]) < -\Delta T_{AC,ON} \rightarrow \theta = 1 \tag{5.6}$$

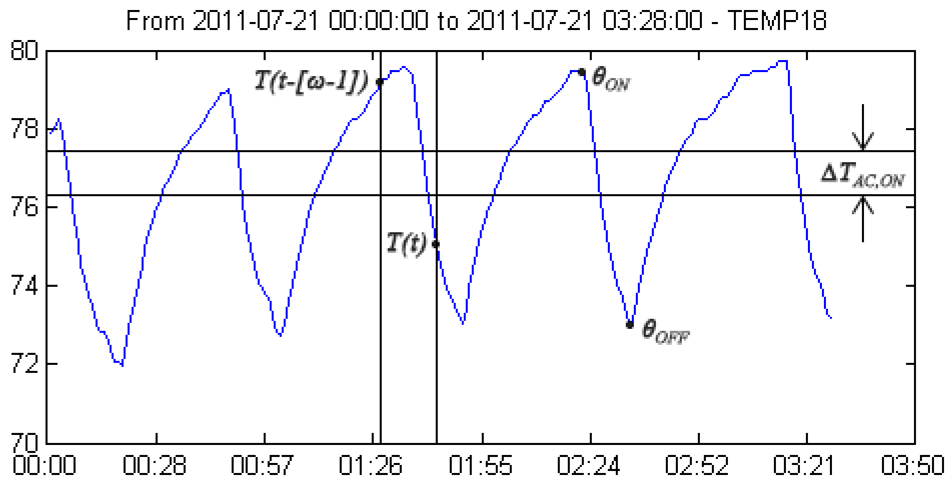


**FIGURE 31 – AIR CONDITION STATUS DETERMINATION**

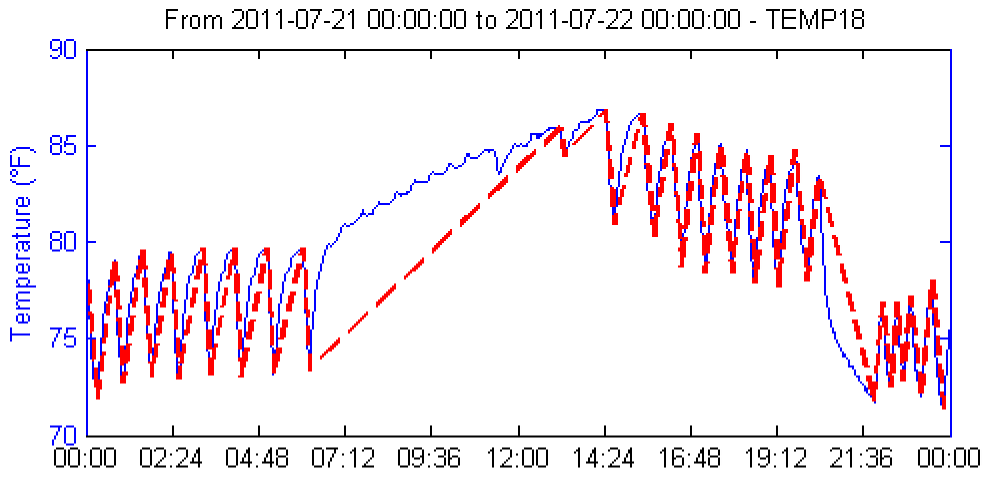
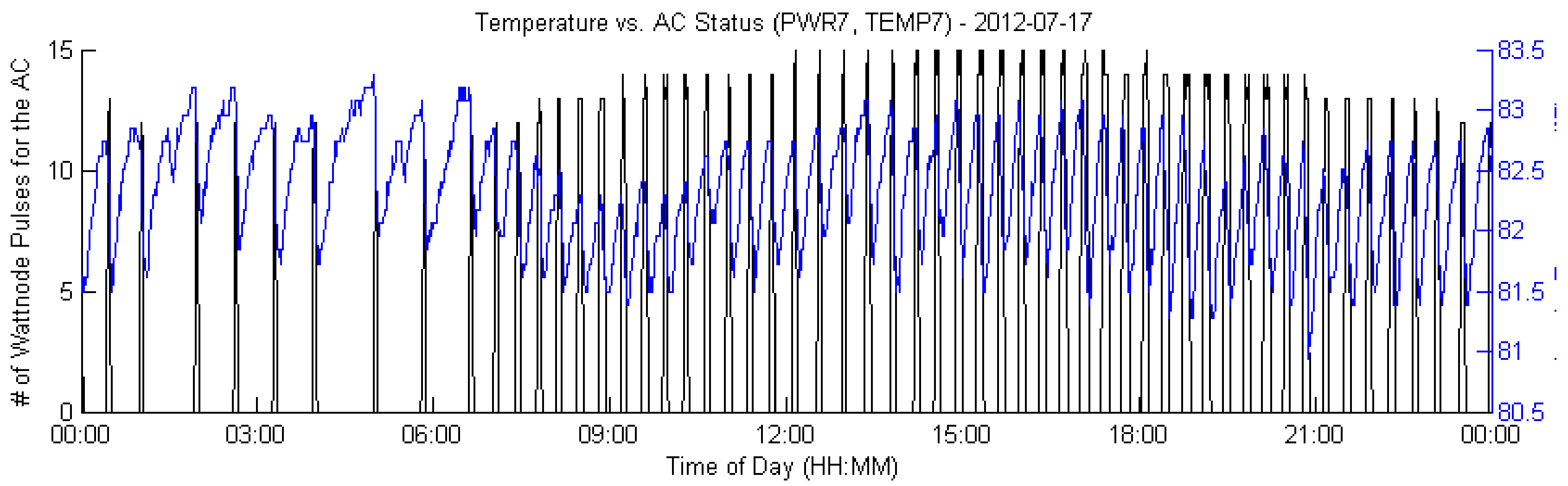


**FIGURE 32 – GENERALIZED TEMPERATURE TRENDS**

The justification for exploring this alternate determination of the AC state is based on some of the characteristics of residential buildings that were discussed in Chapter 1, most notably the relatively small amount of thermal mass these buildings possess, the tendency for residential AC units to be oversized, and square footages small enough to ensure (for all intents and purposes) that the temperature sensor is located in proximity to a cold-air register. These factors should contribute to the measured temperature within a home shifting extremely quickly after an AC event transition occurs, as demonstrated by Figures 28-29. To assess the validity of this assumption, a program was written to compile the necessary data and produce a series of plots overlaying air conditioner CT measurements and interior temperature readings; this program was later expanded, but the plot-producing functionality was not removed. It can be found in Appendix C.

A pair of these plots, generated from first-floor temperature measurements, can be seen in Figure 33. The leftmost sample indicates that TEMP8 is extremely reactive to the AC's operational status – pronounced downward temperature trends correspond extremely well to the points in time in which the WattNode measured the air conditioner as being on. The rightmost plot displays similar same behavior, though not every AC cycle corresponds to a pronounced downward trend in the measured temperature. Regarding the second-floor correlation between temperature and the AC operational status, Figure 34 shows two representative plots. They also demonstrate the immediate effect an air conditioner has on the temperature of a room. The author hypothesizes that the second-floor data may even be, in general, a better indicator of the air conditioner's operation than the first-floor data. The second floor would likely be more prone to heating up quickly due to its proximity to the roof and its associated solar gains, thereby making the downward temperature trends more distinguishable from transience than the first-floor. In addition, fewer internals gains from appliances and residents would result in a second-floor behaving more like a steady-state system than a first-floor, which would be easier to analyze. Tables 12 and 13 appear to back up this assertion–the number of transitions derived from the second-floor temperatures is much closer to the number derived from the CTs.



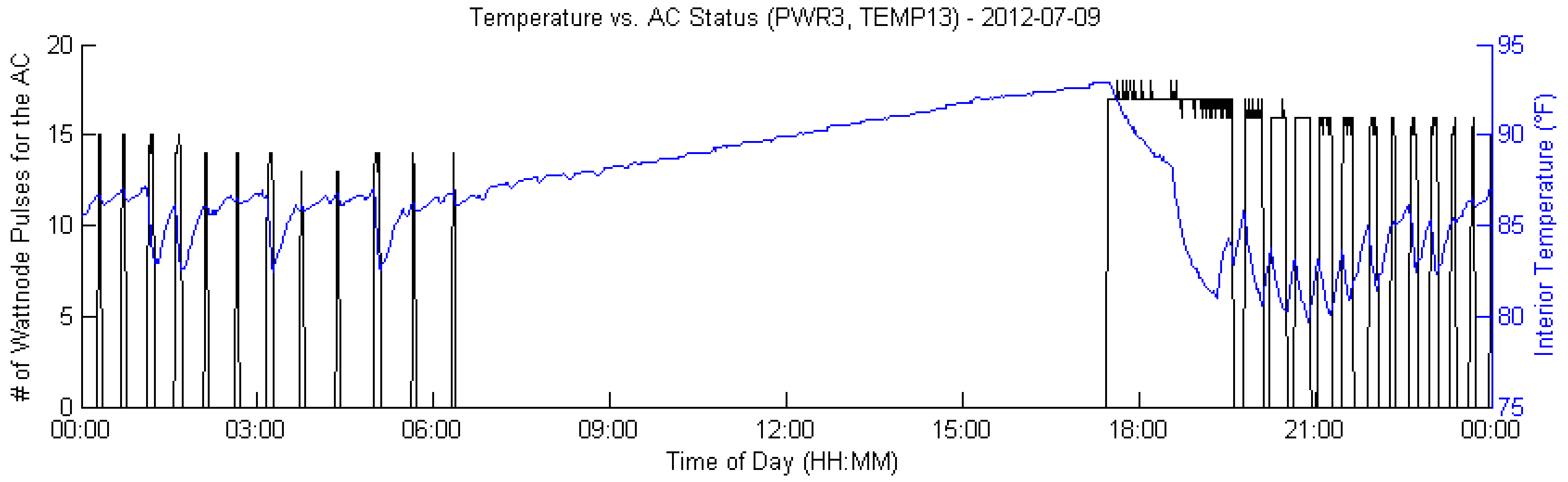


**Figure 33 – Correlation Between First-Floor Interior Temperature and AC Operation**

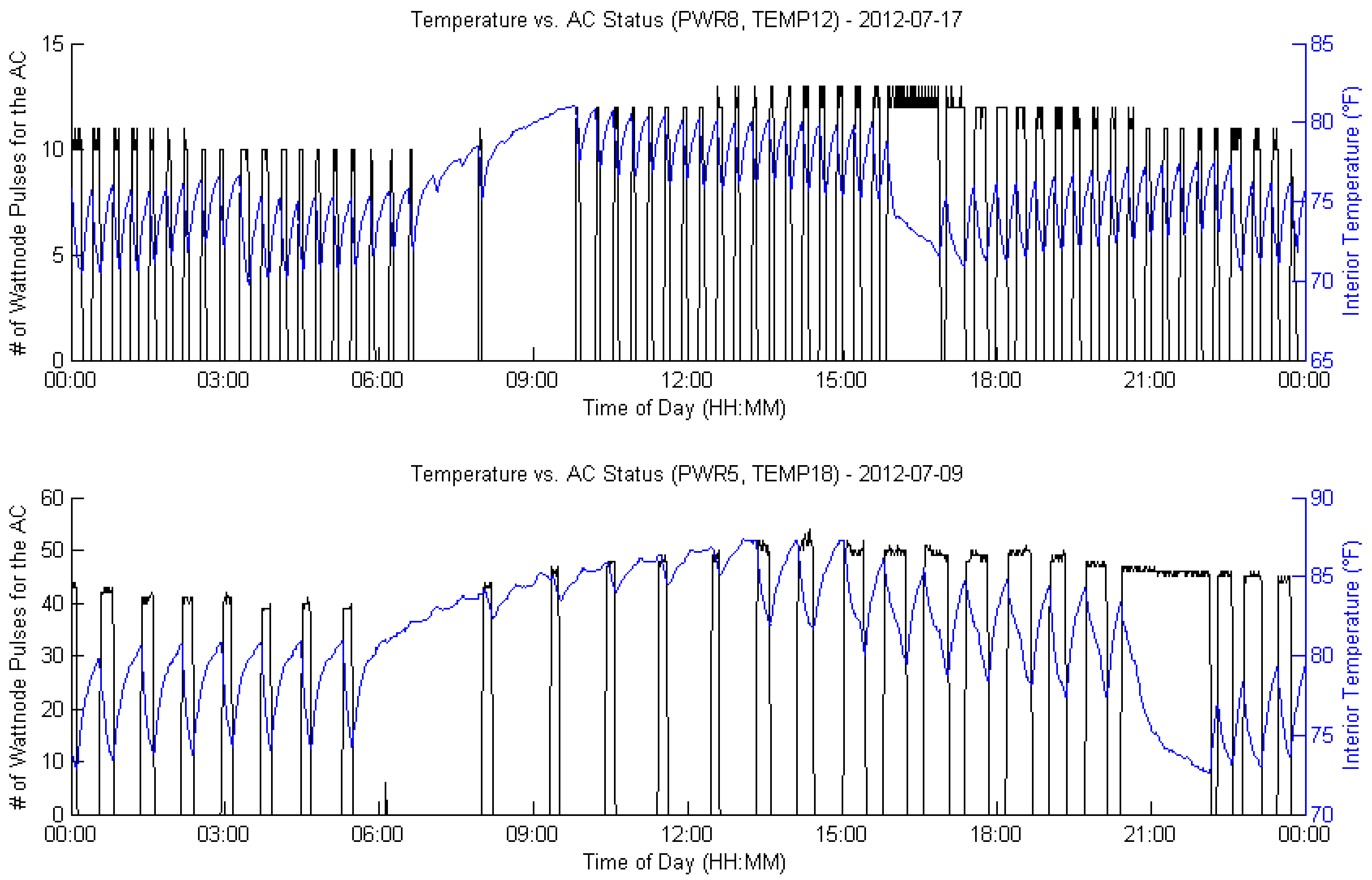


**FIGURE 34 – CORRELATION BETWEEN SECOND-FLOOR INTERIOR TEMPERATURE AND AC OPERATION**

TABLE 12 – AC EVENT TRANSITION COUNTS FOR 07/22/2012 FROM DIFFERENT SOURCES

| Determined by CTs | | Determined by First-Floor Temperatures | | | Determined by Second-Floor Temperatures | | |
|---|---|---|---|---|---|---|---|
| ID | Count | ID | Count | Error | ID | Count | Error |
| PWR2 | 119 | TEMP15 | 99 | 20 | TEMP16 | 97 | 22 |
| PWR3 | 99 | TEMP13 | 64 | 35 | TEMP14 | 92 | 7 |
| PWR5 | 49 | TEMP17 | 48 | 1 | TEMP18 | 48 | 1 |
| PWR6 | 101 | TEMP9 | 0 | 101 | TEMP10 | 34 | 67 |
| PWR7 | 131 | TEMP7 | 12 | 119 | TEMP8 | 132 | -1 |
| PWR8 | 101 | TEMP11 | 33 | 68 | TEMP12 | 99 | 2 |

TABLE 13 – AC EVENT TRANSITION COUNTS FOR 07/02/2012 FROM DIFFERENT SOURCES

| Determined by CTs | | Determined by First-Floor Temperatures | | | Determined by Second-Floor Temperatures | | |
|---|---|---|---|---|---|---|---|
| ID | Count | ID | Count | Error | ID | Count | Error |
| PWR2 | 99 | TEMP15 | 81 | 18 | TEMP16 | 74 | 25 |
| PWR3 | 35 | TEMP13 | 28 | 7 | TEMP14 | 38 | -3 |
| PWR5 | 45 | TEMP17 | 45 | 0 | TEMP18 | 41 | 4 |
| PWR6 | 83 | TEMP9 | 2 | 81 | TEMP10 | 48 | 35 |
| PWR7 | 125 | TEMP7 | 4 | 121 | TEMP8 | 125 | 0 |
| PWR8 | 103 | TEMP11 | 30 | 73 | TEMP12 | 103 | 0 |

A numerical validation of this temperature-derived AC state scheme would consist of comparing its timestamps of the AC event transitions against the comparable timestamps that were generated from the current transducers' data (5.4.1.1). Using the current transducers' data as a baseline, the amount of the time temperature-derived timestamps lag behind, or in front of, the baseline timestamps could be calculated to find out exactly how accurate this method is. A cursory glance at this type of analysis suggests that temperature sensors could very viably, and cost-effectively replace current transducers in a large-scale experimental environment for the purposes of identifying AC event transitions within the context of a residential thermal model.

### *5.4.2 – AC Event Database Storage*

Both methods for reasonably approximating AC event transitions ultimately funnel into a section of code that is intended to 'compress' the minute-by minute data down to a record that is akin to an event-based arrangement. Instead of including records for every minute of a given day, the `*signatures*` database contains detailed specifics for only the AC on/off cycles. The preliminary version of this piece of code computationally intensively looped through the data, and for every AC cycle, it calculated the mean dry bulb temperature and mean global horizontal solar radiation over the event duration. These averaged values were originally intended to be used as inputs to the black-box model, but it was observed through an examination of the code's results that the ambient conditions a house was exposed to over an AC cycle did not vary substantially from the instantaneous measurements at the event's start (with the exception of lengthy setback-cycles). Figure 35 demonstrates this with horizontal lines representing the averaged ambient conditions for each AC on/off event duration throughout a day. In addition, for the model to adhere to the premise of Eqn. (5.1), ambient conditions would only be available to the model at time *t*–the model would not have access to future measurements to determine mean values over the length of the upcoming event.

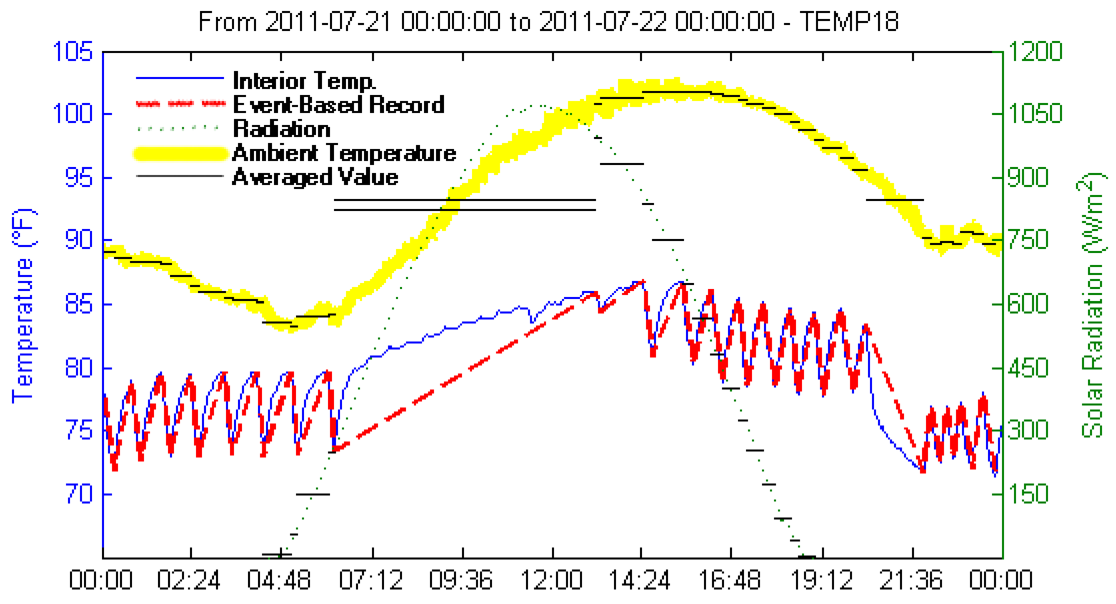


**FIGURE 35 – MEAN AMBIENT CONDITIONS DURING TEMPERATURE TREND INTERVALS**

In lieu of the results presented later in 5.4.4.2 that show no real benefit in the use of averaged ambient values, the final version of the 'compression' script only considers cycle characteristics as they are observed at time $t$. This significantly improves the speed of the 'compression', as the method is able to take advantage of the MATLAB timeseries class. This timeseries class is responsible for synchronizing interior temperature, ambient temperature, and global horizontal radiation measurements to the time domain of the previously calculated event transitions table. It accomplishes this by the linear interpolation method introduced visually by Figure 14.

The script then performs several calculations to classify and quantify temperature trends in the form of, what essentially amounts to, the slope of the change in interior temperature over time. The duration, in seconds, of each AC event is tabulated, then divided by the change in temperature over the given event duration[5], which is also recorded. Lastly, the air conditioner's status is added for each time period and uploaded to the database. An excerpt record from this database can be found in Table 14 (records split for formatting), followed by a description for each field in Table 15.

[5] The inverse slope of the vector is recorded since the duration (in seconds) of the vector is orders of magnitude higher than the change in temperature (in degrees), and floating point numbers small enough to accommodate the true value of the slope are inefficient to store within any database scheme.

**TABLE 14 – AN EXCERPT FROM THE DATABASE CONTAINING 'COMPRESSED' EVENT-BASED DATA**

| Row Type | id int(11) | tempSensor char(6) | periodStart datetime | periodEnd datetime | duration smallint |
|---|---|---|---|---|---|
| | 4015 | TEMP10 | 2012-07-27 03:51:32 | 2012-07-27 04:22:53 | 1881 |
| | 4016 | TEMP10 | 2012-07-27 04:22:53 | 2012-07-27 04:25:06 | 133 |
| | 4017 | TEMP10 | 2012-07-27 04:25:06 | 2012-07-27 05:00:20 | 2114 |
| | 4018 | TEMP10 | 2012-07-27 05:00:20 | 2012-07-27 05:02:25 | 125 |
| | 4019 | TEMP10 | 2012-07-27 05:02:25 | 2012-07-27 06:08:26 | 3961 |
| | 4020 | TEMP10 | 2012-07-27 06:08:26 | 2012-07-27 06:10:24 | 118 |

| Row Type | startTempIn decimal(5,2) | endTempIn decimal(5,2) | invSlope decimal(9,3 | startRad decimal(7,3 | startTempAmb decimal(6,3) | mode tinyint(1 |
|---|---|---|---|---|---|---|
| | 85.2 | 86.09 | 2115.067 | 0 | 83.575 | 0 |
| | 86.09 | 85.16 | -142.043 | 0 | 83.362 | 1 |
| | 85.16 | 86.07 | 2320.527 | 0 | 83.253 | 0 |
| | 86.07 | 85.31 | -166.113 | 18.127 | 82.003 | 1 |
| | 85.31 | 86.31 | 3964.965 | 19.585 | 81.945 | 0 |
| | 86.31 | 85.6 | -165.498 | 237.203 | 83.32 | 1 |

**TABLE 15 – FIELD DESCRIPTIONS FOR THE 'COMPRESSED' EVENT-BASED DATABASE**

| | |
|---|---|
| id | Database-assigned unique identification |
| tempSensor | Temperature sensor identification |
| periodStart | Timestamp of the AC event's start (yyyy-mm-dd HH:MM:SS) |
| periodEnd | Timestamp of the AC event's end (yyyy-mm-dd HH:MM:SS) |
| duration | Duration of the AC event (seconds) |
| startTempIn | Interior temperature at the AC event's start (°F) |
| endTempIn | Interior temperature at the AC event's end (°F) |
| invSlope | Duration/change in interior temperature over AC event duration |
| startRad | Global horizontal radiation at the AC event's start (W/m$^2$) |
| startTempAmb | Ambient dry bulb temperature at the AC event's start (°F) |
| mode | AC ON/OFF status |

### *5.4.3 – Linear Regression*

The formulation of this black-box model follows the process investigated by Rabl et al. [18][56], and later Jang [52]. It attempts to establish a governing differential equation based on a physical model, discretize the equation, then parameterize and weigh the inputs to best fit the measured output data.

The model is based on a simplified understanding of the three basic heat transfer modes experienced by any building – conduction Eqn.(5.7), infiltration/ventilation (convection) Eqn.(5.8), and radiation Eqn.(5.9). Conduction, in this case, encompasses the heat loss that occurs through walls, windows, doors, floors, etc. It can be considered to be a function of the surface area of the building, $A$, the building's steady-state overall loss coefficient, $U$, as well as the temperatures inside, $T_{in}$, and outside, $T_{out}$. Infiltration and ventilation represent a significant contributor to building heat loss, but they are quite difficult values to accurately quantify. Traditionally, they are considered to be primarily influenced by the difference between $T_{in}$ and $T_{out}$ [57]. The total amount of heat loss then also becomes a function of the specific heat capacity of air, $c_p$, the air's density, $\rho$, and the volume of air that is displaced between the interior and the ambient, $q_v$. The amount of radiant heat energy that is added to the house is understood to be just a function of global horizontal radiation, $I_{GHI}$. Direct normal and diffuse horizontal radiation are not treated as independent parameters. An assumption is made that there is no lag between the time radiation falls on the house, and the time the temperature within the house adjusts accordingly.

$$Q_{conduction} = AU(T_{out} - T_{in}) \tag{5.7}$$

$$Q_{infiltration,ventilation} = c_p \rho q_v (T_{out} - T_{in}) \tag{5.8}$$

$$Q_{radiation} = f(I_{GHI}) \tag{5.9}$$

Additionally, two more heat sources are considered – internal gains Eqn.(5.10), and heat added by mechanical systems Eqn.(5.11), which in this case is simply the air conditioner. Internal gains can be affected by numerous influences including the number of people in the house, the appliances running at a given time, lighting, and more, and are therefore extremely impractical to attempt to quantify in a dynamic fashion; internal gains are assumed to be constant throughout the day. Likewise, the contributions made by air conditioners are also assumed to be constant, when operating.

$$\frac{dQ_{internal}}{dt} = 0 \tag{5.10}$$

$$\frac{dQ_{mechanical}}{dt} = 0 \tag{5.11}$$

By further assuming that conduction, infiltration, and ventilation are linear functions of the difference between $T_{in}$ and $T_{out}$ , the rest of the heat gains and losses described above can be used to form a black-box model that predicts how long it takes a house to either heat up, or cool down, to a known threshold Eqn.(5.12).

$$\Delta \mathrm{t} = \left(T_i(t^*) - T_i(t)\right) \times (\alpha[T_a(t) - T_i(t)] + \beta I_{GHI}(t) + \delta\theta + \gamma) \tag{5.12}$$

Where: Δt: Amount of time it will take for the house to reach the next $\theta_{ON}/\theta_{OFF}$ transition (s)

$T_i(t^*)$: A known future temperature, as determined by the thermostat setting (°F)

$\alpha$: Coefficient weighting the impact of the difference between $T_{in}$ and $T_{out}$

$\beta$: Coefficient weighting the impact of radiation

$\delta$: Coefficient weighting the impact of mechanical HVAC systems (air conditioning)

$\gamma$: Regression error

With the black-box model now established, and historical input data available for the variables included in Eqn.(5.12), a script was written (Appendix D) to solve the equation's coefficients through the ordinary least squares method (OLS) for a given house over a specified period of time. This curve-fitting process constitutes the model's training. This approach is repeated for each home, so that each home is associated with its own unique coefficients for the time period considered. Differences between these coefficients from home to home somewhat quantify how the homes respond differently to the forcing inputs; these numbers also represent the conclusion of the system identification, or house identification, process that was introduced at the beginning of the chapter.

In terms of the model training process itself, Eqn.(5.12) is regressed in a two-step process. The first step consists of carefully constructing a subset of data from the original dataset for the time period considered. The purpose of this first step is to eliminate as many independent variables as possible from Eqn.(5.12), and establish the impact of the difference in temperature between the interior and the ambient, $\alpha$, when as many external influences can be ignored. A good subset of data for this purpose includes night-time hours when there is no cooling supplied by the AC system, or when $I_{GHI}$ and $\theta$ both go to 0. Equation (5.12) can then be reduced to Eqn.(5.13), and the inverse slope of a night-time AC-off event suddenly becomes a linear function of the difference in temperatures between the interior and ambient.

$$\alpha[T_a(t) - T_i(t)] + \gamma_1 = \frac{\Delta \mathrm{t}}{T_i(t^*) - T_i(t)} \tag{5.13}$$

Some additional processing of this subset of training data eliminates records where the right side of Eqn.(5.13) evaluates to slope values that are drastically different than the rest of the dataset, allowing the OLS to more accurately represent the dataset's true trend; any outliers outside of 1.5 standard deviations from normal are eliminated. This 'corrected' dataset is then linearly regressed, through the use of the OLS method, to Eqn.(5.13), and best-fit values for the coefficients $\alpha$ and $\gamma$ are determined.

The second step in the model training process begins by applying the coefficients that were determined for $\alpha$ and $\gamma$ during the first step to Eqn.(5.12), and rearranging the equation once again so that one side includes the unknown coefficients, and the other side can be evaluated, allowing another linear regression to occur. This is equation (5.14).

$$\beta I_{GHI}(\bar{t}) + \delta\theta + (\gamma_2 + \gamma_1) = \frac{\Delta t}{T_i(t^*) - T_i(t)} - \alpha[T_a(\bar{t}) - T_i(t)] \tag{5.14}$$

Similarly to the first regression, the dataset is 'corrected' to remove rows of data that include statistically outlying slope values that are at least 1.5 standard deviations from normal. This statistical pruning is applied separately to the data subsets corresponding to time periods when the air conditioner is either on or off to improve the effectiveness of the method. The necessity for this step can be seen between the differences in Figures 36-37.

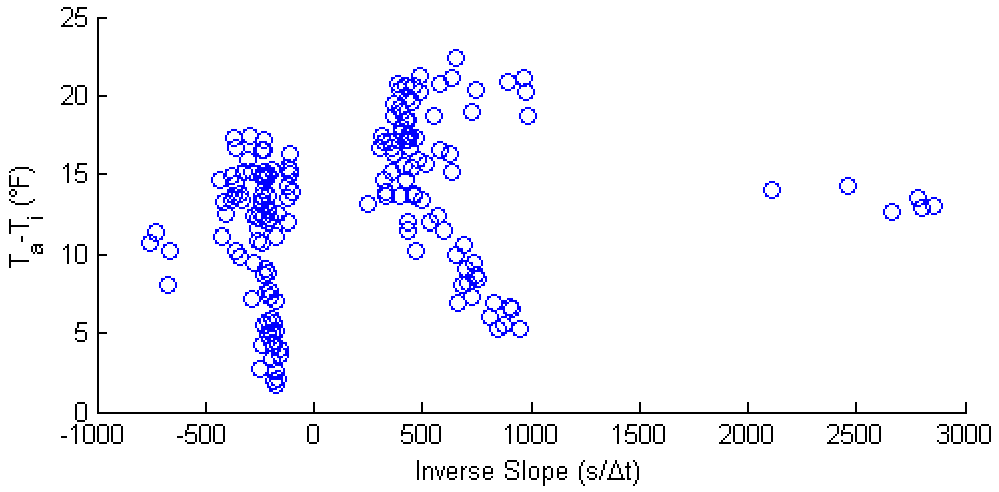


**FIGURE 36 – GENERALIZED TRAINING DATA WITH STATISTICAL OUTLIERS**

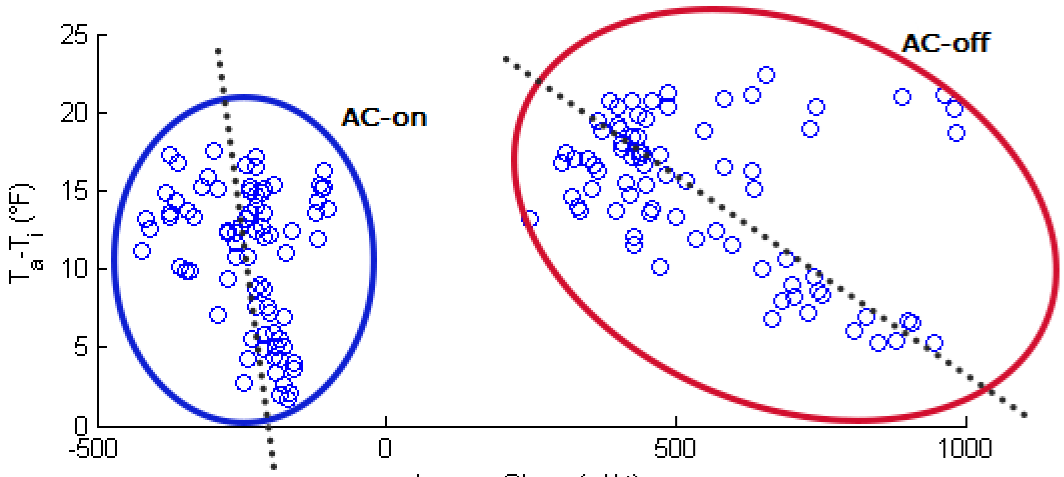


**FIGURE 37 – GENERALIZED TRAINING DATA WITH STATISTICAL OUTLIERS REMOVED**

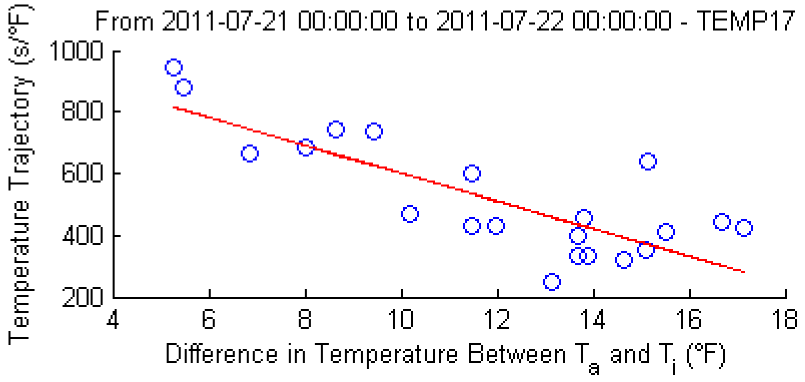
Additionally, Figure 37 illustrates the distinct groupings of the AC-on and AC-off temperature trajectories. They are grouped on either side of zero (a zero slope in this case would be no measured temperature change occurring during an AC cycle, which would not be physically possible based on the control method used by thermostats as discussed in 3.4), as expected. These groupings also indicate that the rate at which this particular house cools down is fairly consistent regardless of the difference between $T_a$ and $T_i$ – the slope of the best-fit curve is relatively steep and the inverse slope does not vary much across the $\Delta T$ range. This justifies the decision to treat $Q_{mechanical}$ as a constant. Conversely, AC-off shows a much higher dependence on the temperature difference between the interior and ambient. This makes sense from a physical standpoint.

At this point, the second regression is performed, and the coefficients $\beta$ and $\delta$ are determined. This finalizes the house identification for the time period considered. Adhering to the offline model ideal, once the coefficients' values have been determined, they are added to the database. Later, another script will rely on these values when it attempts to utilize the model for making AC event duration predictions, as discussed in 5.4.4.2.

An important component to this linear regression method is the formulation of the dataset that is used to train the model. A larger dataset should theoretically result in a more accurate model, but at the cost of increased computational power. In an attempt to balance these considerations, varying lengths of training data were examined for their impact on the accuracy of the best-fit linear regression. Table 16 includes figures and corresponding statistical data for differing dataset durations. Each figure in this table is representative of the first step in the regression process, during the night when the AC is off.

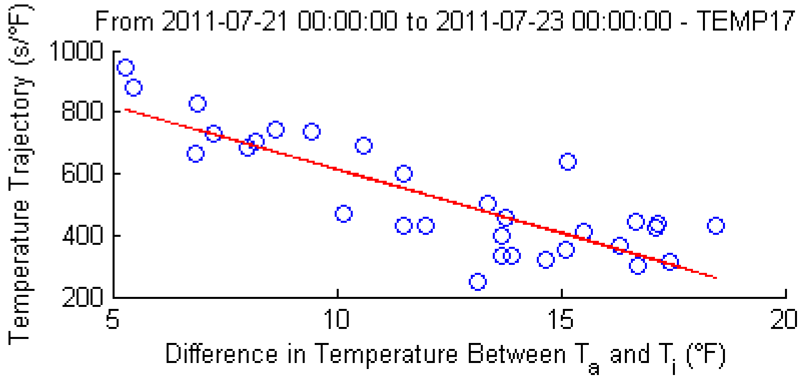
Increasing the number of days included within the training set appears to do very little with regards to increasing the accuracy of the OLS fit; the coefficient of determination decreases after just a few days of training data. For this reason, for the rest of this research, the dataset that will be responsible for training the model will be restricted to the three days prior to the day in which the model will ultimately be applied. As the amount of information within the training set increases, the F statistic continues to rise while its p value correspondingly falls. This confirms that there is certainly a general relationship between the temperature trajectory and the indoor-outdoor temperature difference during the night when the air conditioner is off.

**TABLE 16 – RESULTS OF VARYING LENGTHS OF TRAINING SETS ON REGRESSION ACCURACY**



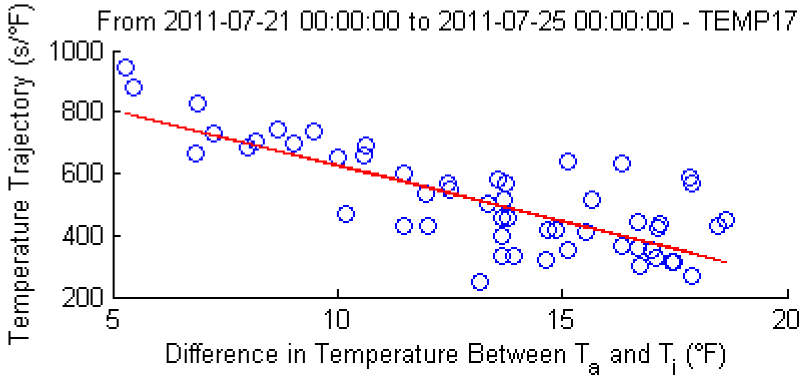
**Days of Data:** 1
**Observations:** 23

**$R^2$ statistic:** 0.644
**F statistic:** 37.92
**p value:** 4.14e-06



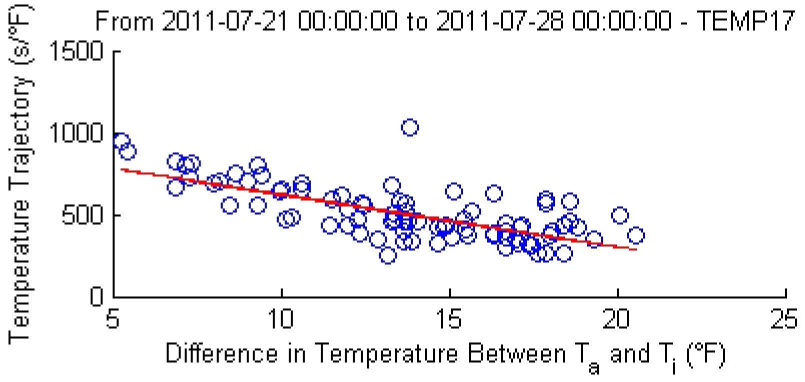
**Days of Data:** 2
**Observations** : 33

**$R^2$ statistic:** 0.681
**F statistic:** 66.13
**p value:** 3.49e-09



**Days of Data:** 4
**Observations:** 57

**$R^2$ statistic:** 0.593
**F statistic:** 80.1
**p value:** 2.54e-12



**Days of Data:** 7
**Observations:** 91

**$R^2$ statistic:** 0.502
**F statistic:** 89.6
**p value:** 4.08e-15

## *5.4.4– Linear Regression Model Validation*

The performance of this linearly regressed model is assessed slightly differently than the validation of most models within the literature. In the majority of published papers, a building model is extended to provide the basis for a control strategy. The building's mechanical systems are then manipulated per this control strategy, and the building's outputs are measured and compared against building outputs that were generated from the baseline model-less control strategy. That method of validation is not yet available for this research, though it could potentially be pursued during Phase II of the Villa Trieste experiment, as discussed in 3.2.

However, the model *can* still be evaluated for its predictive accuracy, thanks in large part to the nature of the model itself. An intermediary output of the model is an interior temperature trajectory, or slope, and the intention of this output is to be used in conjunction with a desired future temperature (what would normally be a thermostat setpoint) to ultimately determine an AC event duration. Since the dataset is historical, and 'future' temperatures of the next AC on/off transition, $T_i(t^*)$, are known, the model can be validated by how closely it predicts the durations of these already-occurred AC events–Eqn. (5.12) is solved and compared against the measured results. In the MPC sense, the amount of time until the next AC on/transition, $t^* - t$, would be unknown, but $T_i(t^*)$ would be understood to be either $T_{high}$ or $T_{low}$. This is shown visually in Figure 38.

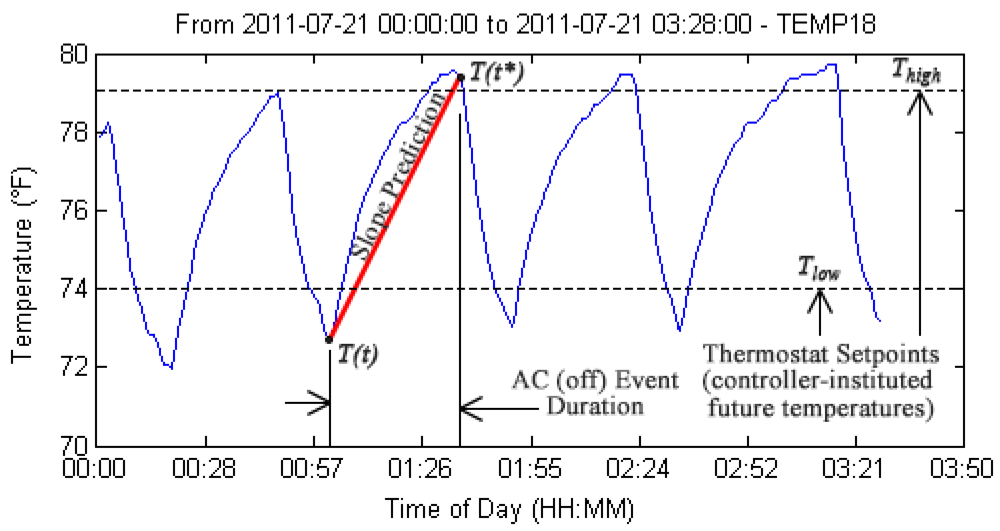


**FIGURE 38 – VISUAL REPRESENTATION OF THE MODEL'S OUTPUT**

### *5.4.4.1 – Regression Coefficients*

The results of the OLS linear regression for July 21, 2011 are shown in Table 16 as they were stored in the database. The coefficients are relatively consistent for each set of sensor data. Magnitudes are comparable, as are signs, with the exception of TEMP16's alpha value. It is positive, where every other sensor's alpha coefficient is negative. An examination of the training set that was used to determine these coefficients for TEMP16 shows that the temperature trajectories were not truly representative of the home's thermal response. This was an error rooted in the AC state estimation script (as inferred by temperature measurements, not as recorded by the current transducers), not necessarily the regression analysis. This particular error was not corrected, but the improved method introduced in the next section does utilize CT-based AC state calculations instead.

**Table 17 – Results of Regression For Each Sensor on July 21, 2011**

| device | regressionDay | samples | alpha | gamma | beta | delta | lambda |
|---|---|---|---|---|---|---|---|
| char(6) | date | int(5) | float | float | float | float | float |
| TEMP6 | 2011-07-21 | 527 | -15.9117 | 563.384 | 0.147663 | -584.067 | 8.025 |
| TEMP7 | 2011-07-21 | 325 | -42.0777 | 1256.14 | 0.133543 | -954.803 | 19.449 |
| TEMP8 | 2011-07-21 | 671 | -18.4602 | 684.784 | 0.043119 | -568.802 | 10.230 |
| TEMP13 | 2011-07-21 | 218 | -41.8922 | 1045.93 | 0.087686 | -867.774 | 5.084 |
| TEMP14 | 2011-07-21 | 866 | -27.9218 | 790.027 | 0.102956 | -744.915 | 8.118 |
| TEMP15 | 2011-07-21 | 147 | -27.0789 | 1349.99 | 0.806008 | -2229.1 | 45.591 |
| TEMP16 | 2011-07-21 | 157 | 9.11428 | 1384.72 | 0.45235 | -1986.22 | 10.146 |
| TEMP17 | 2011-07-21 | 168 | -39.2035 | 1014.37 | 0.490345 | -954.568 | 11.13 |
| TEMP18 | 2011-07-21 | 153 | -13.964 | 491.977 | 0.122909 | -518.122 | 10.755 |

### *5.4.4.2 – Trajectory Prediction*

As mentioned, the OLS regression relies on three day-long datasets, and the coefficients that are a result of this analysis are valid for just the day immediately following the training set. When these coefficients are used to evaluate Eqn.(5.12), the model is effectively 'offline' since the coefficients are applied, without being updated, until the next day. A demonstration of a single day's predicted trajectories, and subsequent AC event durations, is shown in Figure 39.

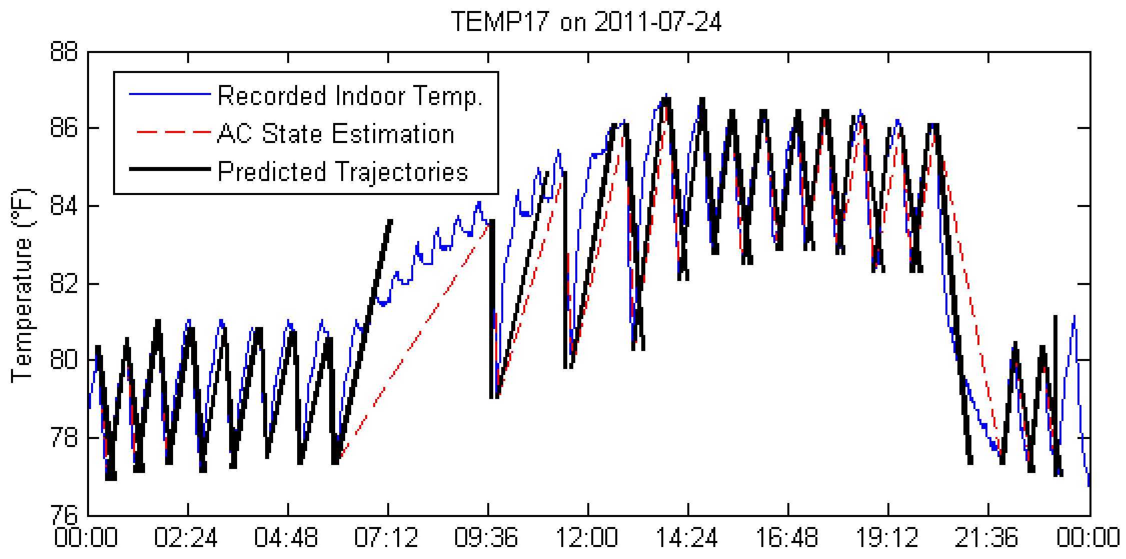


**Figure 39 – Linear Regression Predicted Trajectories**

To determine the accuracy of a given trajectory, the predicted AC event duration, $\Delta t_P$, is compared to the actual state duration, $\Delta \mathrm{t}_A$, and the difference is expressed as a percentage of $\Delta \mathrm{t}_A$. Additionally, the predicted durations were also calculated by using averaged radiation and ambient temperature values, $\Delta t_{\bar{P}}$, as described in 5.4.2. The results of these evaluations can be found in Table 18. As can be seen, there are no substantial improvements gained by using averaged values, thus justifying the use of instantaneous $I_{GHI}$ and $T_a$ values at time $t$ rather than averaged values from $t$ to $t^*$.

There are two very notable data samples within this table. The model does a very poor job of predicting the trajectories that begin at 05:58:41 and again at 20:18:47. These appear to be intervals of time in which thermostat setpoints are changing.

Within Table 18, the critical time period from early afternoon to late evening has been highlighted as the typical demand response period. The amount of error shown during this time ranges from an underestimation of 43% of the actual state duration all the way up to an overestimation of 81%. The model performs much more reliably during the late evening and early morning hours; however this does little to promote the utility of this model, as predictions during these times do not assist in minimizing critical peak demands.

Table 18 also exhibits some signs of bias. For this particular sensor on the day examined, the model seems to be more accurate when the air conditioner is off and the algorithm is attempting to predict when it turns back on, as seen in Figure 40. There could be several reasons for this occurrence:

1. The overly-simplified model could potentially be missing a critical independent variable.

**TABLE 18 – LINEAR REGRESS. TRAJECTORY LENGTH PREDICTION ERROR FOR TEMP17 ON 2011-07-24**

| Trajectory Start | $\Delta t_A$ | $\Delta t_P$ | $\Delta t_{\bar{P}}$ | $\Delta t_P - \Delta t_{\bar{P}}$ | $\Delta t_P - \Delta t_A$ | $\frac{\Delta t_P - \Delta t_A}{\Delta t_A}$ |
|---|---|---|---|---|---|---|
| (HH:MM:SS) | (s) | (s) | (s) | (s) | (s) | (s) |
| 00:15:39 | 900 | 1389 | 1381 | 8 | 489 | 54.3% |
| 00:30:39 | 1560 | 1585 | 1629 | -44 | 25 | 1.6% |
| 00:56:39 | 840 | 1247 | 1260 | -13 | 407 | 48.5% |
| 01:10:39 | 1801 | 1823 | 1889 | -66 | 22 | 1.2% |
| 01:40:40 | 1020 | 1137 | 1148 | -11 | 117 | 11.5% |
| 01:57:40 | 1980 | 1725 | 1834 | -109 | -255 | -12.9% |
| 02:30:40 | 840 | 1065 | 1042 | 23 | 225 | 26.8% |
| 02:44:40 | 1980 | 1976 | 2021 | -45 | -4 | -0.2% |
| 03:17:40 | 720 | 900 | 806 | 94 | 180 | 25.0% |
| 03:29:40 | 2341 | 2159 | 2238 | -79 | -182 | -7.8% |
| 04:08:41 | 660 | 509 | 581 | -72 | -151 | -22.9% |
| 04:19:41 | 2280 | 2081 | 2159 | -78 | -199 | -8.7% |
| 04:57:41 | 600 | 526 | 425 | 101 | -74 | -12.3% |
| 05:07:41 | 2460 | 2344 | 2358 | -14 | -116 | -4.7% |
| 05:48:41 | 600 | 309 | 312 | -3 | -291 | -48.5% |
| 05:58:41 | 13322 | 4770 | 4621 | 149 | -8552 | -64.2% |
| 09:40:43 | 540 | -93 | -64 | -29 | -633 | -117.2% |
| 09:49:43 | 5760 | 4620 | 4339 | 281 | -1140 | -19.8% |
| 11:25:43 | 600 | 125 | 69 | 56 | -475 | -79.2% |
| 11:35:43 | 4683 | 4576 | 3754 | 822 | -107 | -2.3% |
| 12:53:46 | 658 | 1384 | 1453 | -69 | 726 | 110.3% |
| 13:04:44 | 3000 | 3702 | 2615 | 1087 | 702 | 23.4% |
| 13:54:44 | 1020 | 1553 | 1613 | -60 | 533 | 52.3% |
| 14:11:44 | 1921 | 1960 | 2082 | -122 | 39 | 2.0% |
| 14:43:45 | 960 | 1017 | 1154 | -137 | 57 | 5.9% |
| 14:59:45 | 1620 | 1735 | 1670 | 65 | 115 | 7.1% |
| 15:26:45 | 960 | 1476 | 1507 | -31 | 516 | 53.8% |
| 15:42:45 | 1860 | 1588 | 1645 | -57 | -272 | -14.6% |
| 16:13:45 | 1200 | 1390 | 1402 | -12 | 190 | 15.8% |
| 16:33:45 | 1621 | 1551 | 1681 | -130 | -70 | -4.3% |
| 17:00:46 | 720 | 1303 | 1344 | -41 | 583 | 81.0% |
| 17:12:46 | 1740 | 1529 | 1533 | -4 | -211 | -12.1% |
| 17:41:46 | 1020 | 1441 | 1463 | -22 | 421 | 41.3% |
| 17:58:46 | 2100 | 1431 | 1375 | 56 | -669 | -31.9% |
| 18:33:46 | 1020 | 1790 | 1755 | 35 | 770 | 75.5% |
| 18:50:46 | 2341 | 1329 | 1390 | -61 | -1012 | -43.2% |
| 19:29:47 | 960 | 1564 | 1564 | 0 | 604 | 62.9% |
| 19:45:47 | 1980 | 1473 | 1529 | -56 | -507 | -25.6% |
| 20:18:47 | 5821 | 3381 | 2980 | 401 | -2440 | -41.9% |
| 21:55:48 | 1020 | 1050 | 1087 | -37 | 30 | 2.9% |
| 22:12:48 | 1260 | 1599 | 1599 | 0 | 339 | 26.9% |
| 22:33:48 | 1140 | 1119 | 1133 | -14 | -21 | -1.8% |
| 22:52:48 | 1260 | 1603 | 1562 | 41 | 343 | 27.2% |
| 23:13:48 | 1500 | 1490 | 1497 | -7 | -10 | 0.6% |

2. The time it takes for the house to cool down is significantly shorter than the time it takes for it to heat up (Figure 41). The shorter duration makes the prediction more difficult from a percentage-based metric.
3. The inconsistencies of $T_i(t^*)$ and $T_i(t)$ demonstrate behavior unlike a thermostat with consistent set points and dead band offsets. This may be particularly relevant to the set-back time period where it is difficult to discern what temperature the air conditioner is attempting to allow the house to cool to.
4. The assumption made by Eqn.(5.11) that the energy contribution of the air conditioner is steady and consistent may not be valid for the purposes of the model.
5. Either the internal gains assumption made by Eqn.(5.10) may be invalid, or the internal gains dominate the other heating modes to the point that its inherent variability is too ill-suited for an offline prediction model.

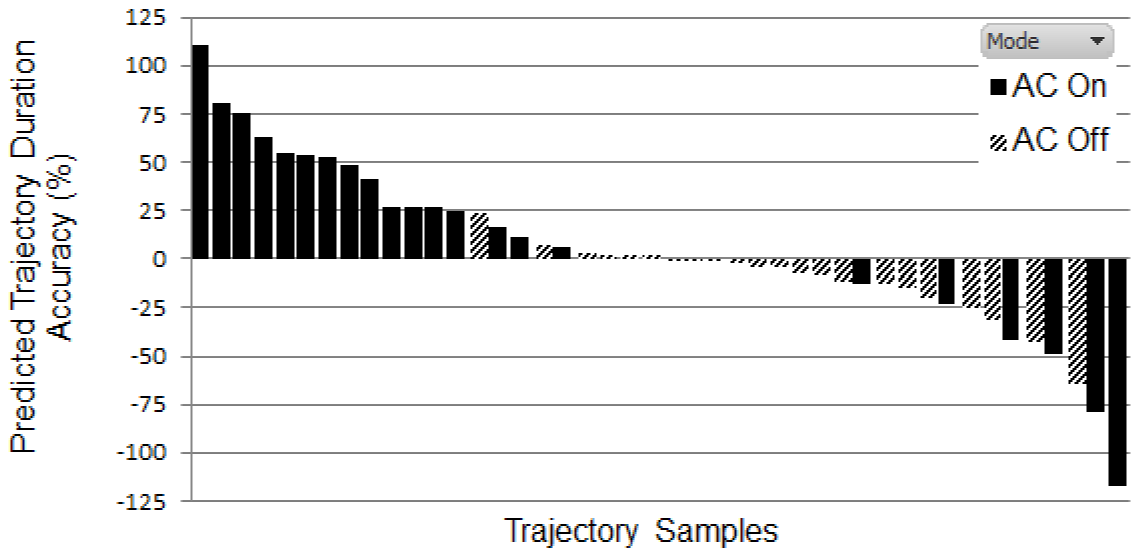


**FIGURE 40 – ERROR BIAS IN THE LINEAR REGRESSION PREDICTION MODEL**

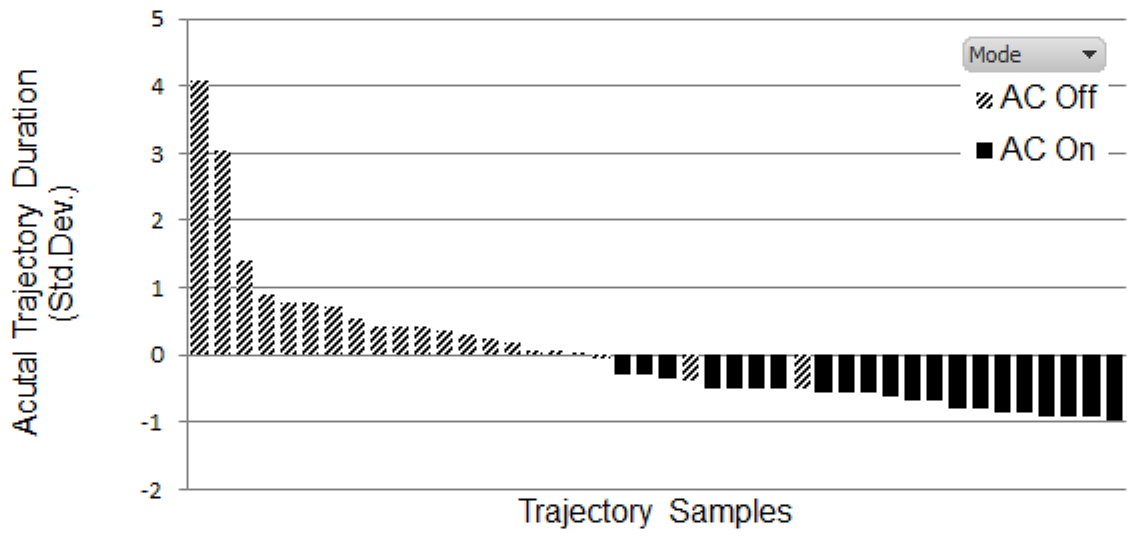


**Figure 41 – Normalized Trajectory Durations for AC On/Off States**

To confirm that the selected sensor on the given day was representative of the model's accuracy, predictions were also run for multiple sensors on several other summer days. The results were averaged and summarized in Table 19.

**Table 19 – Average Event Duration Error Of Several Devices over Multiple Days**

| 6/18/2011 | | | | | 9/13/2011 | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Device/ Status | $\Delta t_A$ (s) | $\Delta t_P$ (s) | Error (s) | Error (%) | Device/ Status | $\Delta t_A$ (s) | $\Delta t_P$ (s) | Error (s) | Error (%) |
| TEMP6 | | | | | TEMP8 | | | | |
| AC ON | 411 | 620 | 209 | 50.4% | AC ON | 500 | -872 | -1372 | -281.6% |
| AC OFF | 1359 | 962 | -397 | -8.5% | AC OFF | 5224 | 3672 | -1551 | -19.9% |
| TEMP8 | | | | | TEMP12 | | | | |
| AC ON | 466 | 277 | -188 | -40.1% | AC ON | 609 | 433 | -176 | -13.2% |
| AC OFF | 2093 | 1741 | -352 | -10.1% | AC OFF | 313 | 1061 | 748 | -6.6% |
| TEMP17 | | | | | TEMP13 | | | | |
| AC ON | 809 | 797 | -12 | -6.7% | AC ON | 360 | -355 | -715 | -199.5% |
| AC OFF | 3632 | 2738 | -895 | -5.2% | AC OFF | 352 | 3841 | 3489 | -22.3% |
| TEMP18 | | | | | TEMP18 | | | | |
| AC ON | 844 | 1009 | 165 | 27.4% | AC ON | 889 | 503 | -387 | -28.6% |
| AC OFF | 4116 | 1536 | -2580 | -45.8% | AC OFF | 808 | 3039 | 2231 | 16.9% |

Table 19 indicates that the ability of the model to accurately predict trajectory duration is highly variable from sensor to sensor. Several daily predictions averaged out to be within 10% of the true duration, while others were so inaccurate that they predicted *negative* durations. In this state, the model is unusable in any capacity.

### *5.4.5 – Curvilinear Regression*

Several shortcomings of the first-order linear regression approach to trajectory prediction were discussed within the last section. Namely, natural processes are usually nonlinear (that is, their best-fit curves are defined by power functions), and linearized models tend to do a poor job of generalizing the relationship between independent and dependent variables. Nonlinear, or curvilinear, models are typically computationally intensive due to the involvement of an exponential term in the regression, but one type of curvilinear model can still be best-fit with the OLS approach (meaning the *regression method* is still technically linear, even though the results of the model are free to take on a parabolic shape) – a polynomial model. This simply means that input variables are allowed to be second-order terms, as shown in Eqn.(5.15).

$$y = \alpha_1 + \alpha_2 x_1 + \alpha_3 {x_1}^2 \tag{5.15}$$

These second-order input variables can also be explained with the introduction of *interaction terms*. Interaction terms allow a model to consider the influence two or more input variables have on each *other*, rather than just how they affect the dependent variable. This is accomplished by assigning a regression coefficient to the *product* of two or more inputs, as shown in Eqn.(5.16). In this sense, a second-order input can be considered to be just another interaction term–between the input and itself, as shown in Eqn.(5.17).

$$y = \alpha_1 + \alpha_2 x_1 + \alpha_3 {x_1}^2 + \alpha_4 x_2 + \alpha_5 x_1 x_2 + \alpha_6 {x_2}^2 \tag{5.16}$$

$$\alpha_3 {x_1}^2 = \alpha_3 x_1 x_1 \tag{5.17}$$

Taking these interaction terms into consideration during model development allows for a significant assumption improvement to be made over the linear model; no longer do equations (5.11) and (5.12) necessarily apply. The energy contributions of an air conditioner do not need to be constant throughout the day. Instead, the air conditioner can be influenced by both the difference in temperature between indoors and outdoors and the amount of solar radiation falling on the house at a given time. This is shown by the inclusion of these interaction terms in the curvilinear regression model of Eqn.(5.18).

$$\begin{aligned}\frac{\Delta \mathrm{t}}{T_i(t^*) - T_i(t)} = {} & \alpha[T_a(t) - T_i(t)]^2 + \beta[T_a(t) - T_i(t)] + \gamma + \delta I_{GHI} + \varepsilon\theta \\ & + \zeta\theta[T_a(t) - T_i(t)]^2 + \eta\theta[T_a(t) - T_i(t)] + \lambda\theta I_{GHI}\end{aligned} \tag{5.18}$$

However, the tradeoff between the curvilinear and linear regression models is that the complexity of the model increases with the additional terms. Instead of relying on four coefficients, the model now includes eight coefficients that must be regressed. Fortunately, the method for resolving these values is very similar to the linear approach. The first regression step is once again performed for the dataset that includes only time periods of no solar radiation (night time), and when the air conditioner is off. This simplifies Eqn.(5.18) to the form of Eqn.(5.19), and the coefficients $\alpha, \beta$, and $\gamma$ are determined through the OLS approach.

$$\frac{\Delta \mathrm{t}}{T_i(t^*) - T_i(t)} = \alpha[T_a(t) - T_i(t)]^2 + \beta[T_a(\bar{t}) - T_i(t)] + \gamma \tag{5.19}$$

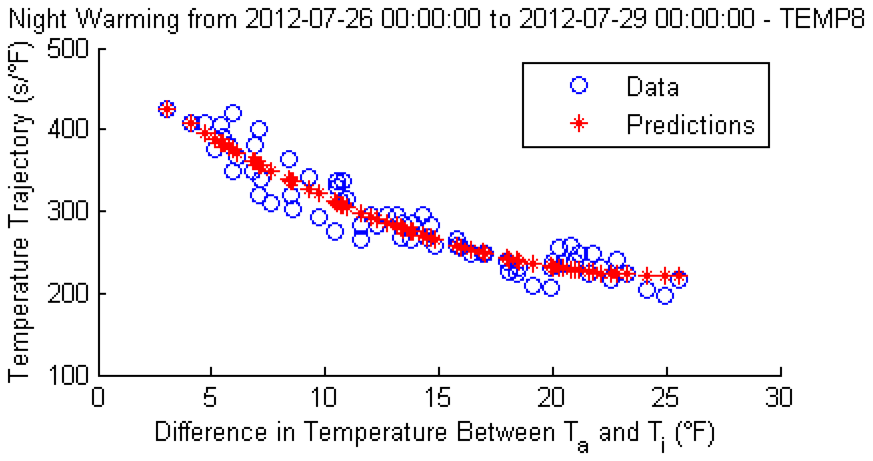
With the initial coefficients now established, the remaining coefficients can be determined through an OLS regression of the full three-day dataset, for both air conditioner operating states

Eqn.(5.20). These coefficients are added to the database in a manner identical to those outlined in Table 17.

$$\frac{\Delta t}{T_i(t^*) - T_i(t)} - \alpha[T_a(t) - T_i(t)]^2 - \beta[T_a(t) - T_i(t)] - \gamma = \delta I_{GHI} + \varepsilon\theta + \zeta\theta[T_a(t) - T_i(t)]^2 + \eta\theta[T_a(t) - T_i(t)] + \lambda\theta I_{GHI} \tag{5.20}$$

### *5.4.6– Curvilinear Regression Validation*

For initial illustrative purposes, the validation of the curvilinear approach examines a sensor that exhibits a particularly cyclical interior temperature, and doesn't appear to feature any thermostat setpoint changes–TEMP8. This allows for a high level of consistency in the data, which is made evident by regarding the tight grouping of trajectories shown in the figure included as part of Figure 42.



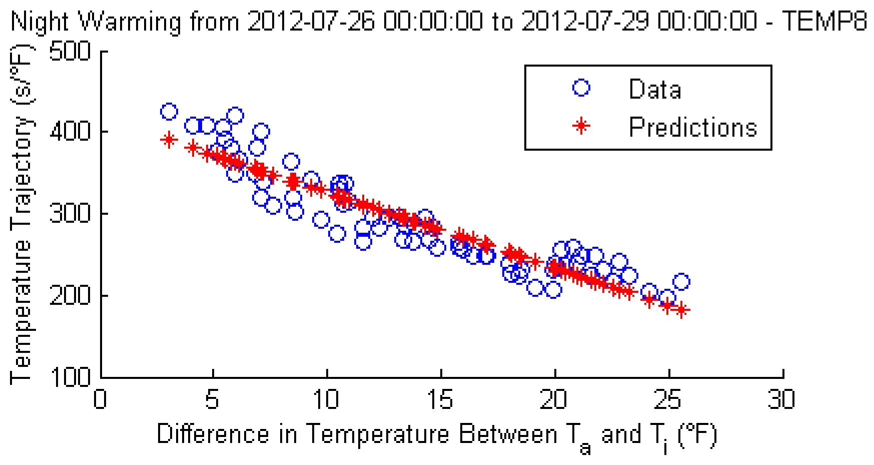
**Method:** Curvilinear

**Days of Data:** 3
**Observations:** 69

**$R^2$ statistic:** 0.90
**F statistic:** 312.1
**p value:** 2.25e-34

**FIGURE 42 – COMPARATIVE REGRESSION RESULTS FOR NIGHT WARMING (CURVILINEAR)**

Additionally, the trajectory groupings of Figure 42 show a definite dependence on the difference in temperature between the interior and the ambient. With a coefficient of determination of 0.90, the second-order nature of Eqn.(5.20) tends to fit the data better than the linear model shown in Figure 43. The residuals for both methods are shared in Figure 44.

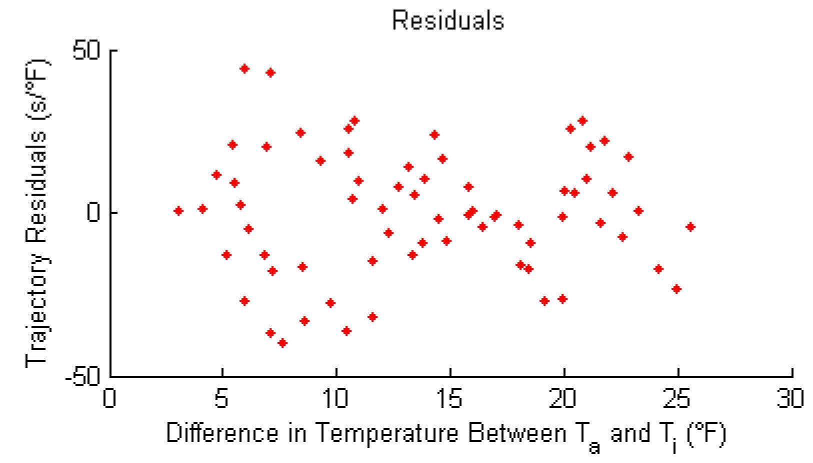


**Method:**
Linear

**Days of Data:** 3
**Observations:** 69

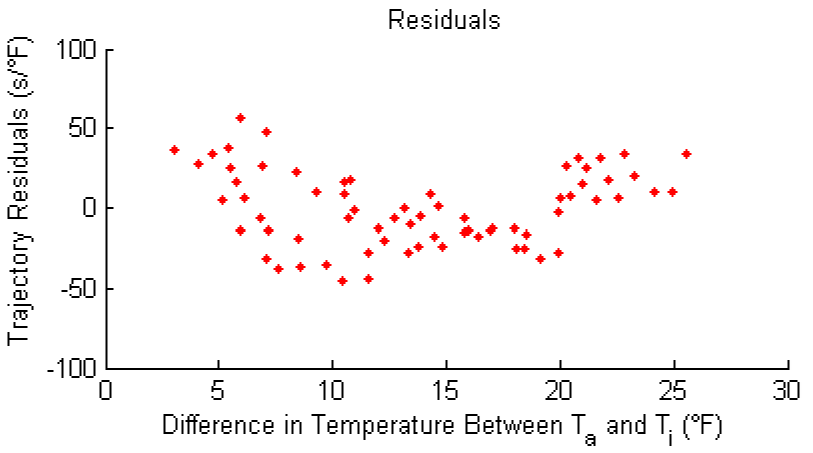
**$R^2$ statistic:** 0.85
**F statistic:** 379.9
**p value:** 2.58e-29

FIGURE 43 – COMPARATIVE REGRESSION RESULTS FOR NIGHT WARMING (LINEAR)



**Method:**
Curvilinear

**Norm of Residuals:**
155.69



**Method:**
Linear

**Norm of Residuals:**
194.94

FIGURE 44 – ACCURACY OF CURVILINEAR VS. LINEAR REGRESSION FOR NIGHT WARMING

The second-order bias of the linear residuals, qualitatively measured by the decrease in the normality of residuals, further justifies the adoption of the curvilinear model as the default regression method.

Of course, the curvilinear validation must also extend into the full dataset–not just the night-warming training subset. Figure 45 is an extension of Figures 43-44. It still shows the relationship between temperature trajectories and the interior/ambient temperature differential, but its axes have been inverted for formatting. It also includes the full dataset and further illustrates the distinct groupings of night, AM, and PM trajectories. As expected, the largest temperature differential takes place in the afternoon hours and the longest trajectories occur when the air conditioner is off and solar radiation is non-existent. Figure 45 becomes even more interesting when the inclusion of solar radiation is considered, as shown in Figure 46.

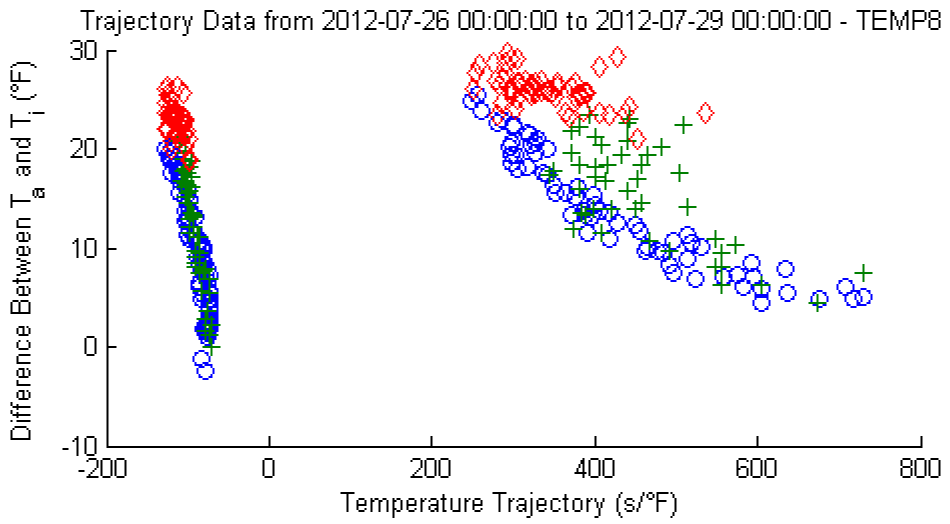


**FIGURE 45 – 2D DISTINCT DAY/NIGHT TEMPERATURE TRAJECTORY GROUPINGS**

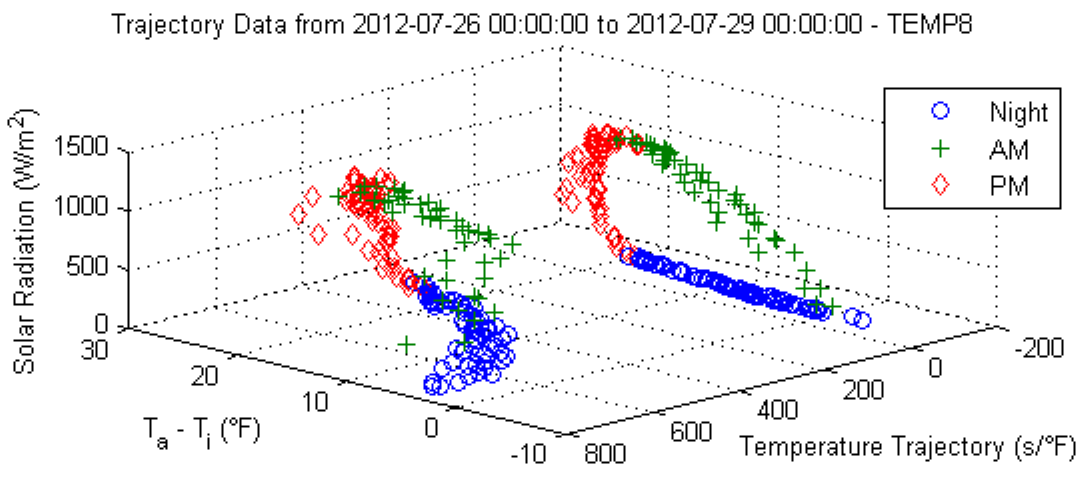


**FIGURE 46 – 3D DISTINCT DAY/NIGHT TEMPERATURE TRAJECTORY GROUPINGS**

By also plotting the temperature trajectories versus solar radiation, multiple seemingly non-linear patterns become clearly visible. It is these non-linear trends that highlight the superiority of the curvilinear over linear regression method for this particular application. The results of the model's training have been added to the plot seen in Figure 47.

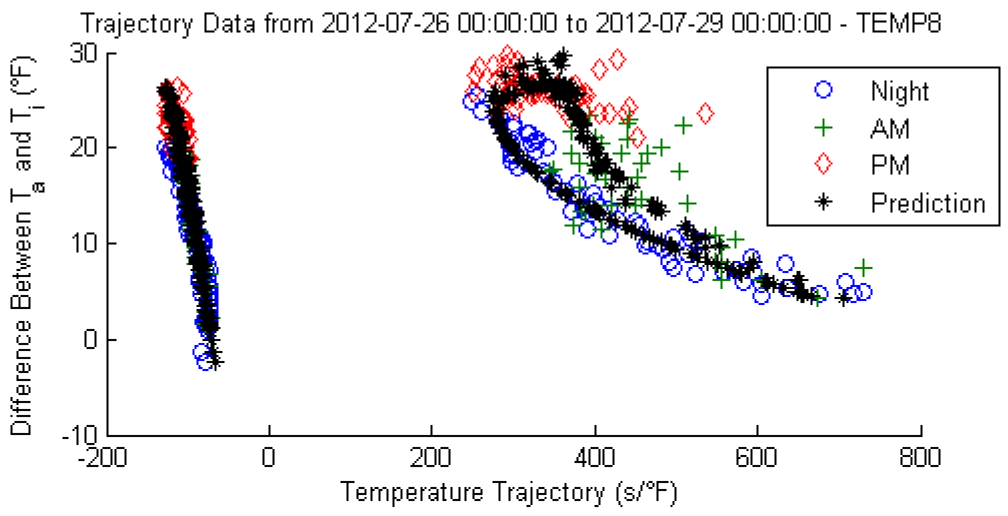


**FIGURE 47 – CURVILINEAR MODEL TRAINING RESULTS**

### *5.4.6.1 – Regression Coefficients*

Unlike the linear regression method, the coefficients determined by the curvilinear method are not particularly consistent for each sensor over the same time period. An excerpt of this table can be seen in Table 20, which is divided due to formatting constraints. Magnitudes vary tremendously for each coefficient across the sensors, and even signs are not particularly consistent. One possible reason for this is that, due to the increased order of the defining equation, there are more degrees of freedom for the resulting best-fit curves, which in turn increase the number of potential curve shapes. As cautioned in 2.2.2, however, it is ill-advised to attempt to draw physical conclusions from the parameterization weights of a black-box model.

**TABLE 20 – CURVILINEAR REGRESSION COEFFICIENTS**

| tempSensor | date | samples | alpha | beta | gamma | delta | epsilon | zeta | eda | lambda | error |
|---|---|---:|---:|---:|---:|---:|---:|---:|---:|---:|---:|
| TEMP7 | 2012-07-06 | 330 | -3.03374 | -36.9245 | 2062.13 | 0.434859 | -2405.57 | 3.11799 | 33.7726 | -0.505502 | 45.9687 |
| TEMP8 | 2012-07-06 | 315 | 0.320988 | -41.4315 | 924.214 | 0.110144 | -933 | -0.41443 | 41.6062 | -0.111426 | -74.2099 |
| TEMP9 | 2012-07-06 | 165 | -176.305 | -152.018 | 19383.9 | 7.04147 | -21286.7 | 177.259 | 233.719 | -7.99634 | 53.3492 |
| TEMP10 | 2012-07-06 | 210 | -4.95901 | -49.0768 | 1718.79 | 0.456438 | -2078.33 | 4.95627 | 45.0407 | -0.384611 | 132.164 |
| TEMP11 | 2012-07-06 | 286 | 1.73431 | -46.7201 | 1063.39 | 0.179314 | -1455.5 | -2.35255 | 46.1192 | -0.140902 | 51.2462 |
| TEMP12 | 2012-07-06 | 297 | -0.0177045 | -5.77173 | 310.863 | 0.0408463 | -397.373 | -0.0973828 | 5.34473 | -0.0254408 | 10.8726 |
| TEMP13 | 2012-07-06 | 93 | -0.317504 | -67.0371 | 1272.67 | 5.50857 | -1395.64 | -0.513102 | 57.4439 | -5.78606 | -36.0655 |
| TEMP14 | 2012-07-06 | 97 | 3.97966 | -155.691 | 1678.59 | 6.68066 | -1834.75 | -5.32683 | 157.752 | -6.87178 | 36.4424 |
| TEMP15 | 2012-07-06 | 234 | -1.24011 | -54.4381 | 1834.82 | 0.626558 | -2214.35 | 0.939941 | 53.2553 | -0.583487 | 122.503 |
| TEMP16 | 2012-07-06 | 228 | 1.37001 | -46.0447 | 799.537 | 0.0844042 | -1085.95 | -1.6924 | 44.6037 | -0.0291007 | 209.385 |
| TEMP17 | 2012-07-06 | 106 | -4.98759 | 48.6746 | 728.994 | 0.559952 | -1086.57 | 4.67081 | -53.6579 | -0.354696 | 146.671 |
| TEMP18 | 2012-07-06 | 94 | -0.582331 | 0.665858 | 432.454 | 0.445513 | -588.21 | 0.0760113 | 2.87414 | -0.45134 | 46.2695 |

### *5.4.6.2 – Model Training*

The quality of the model's trajectory predictions fundamentally relies on the training of the model itself. The quality of this training is dependent on two criteria:

1. Consistency of the trajectories within the training dataset
2. Similarity between the training dataset and the dataset the trained model is applied to

Addressing the first of these criteria, the temperature sensor and training dataset that was featured in the last section was intentionally chosen to illustrate the feasibility of the process. In reality, not every training set features the same level of consistency. Figures 48-50 show a sampling of some of these instances.

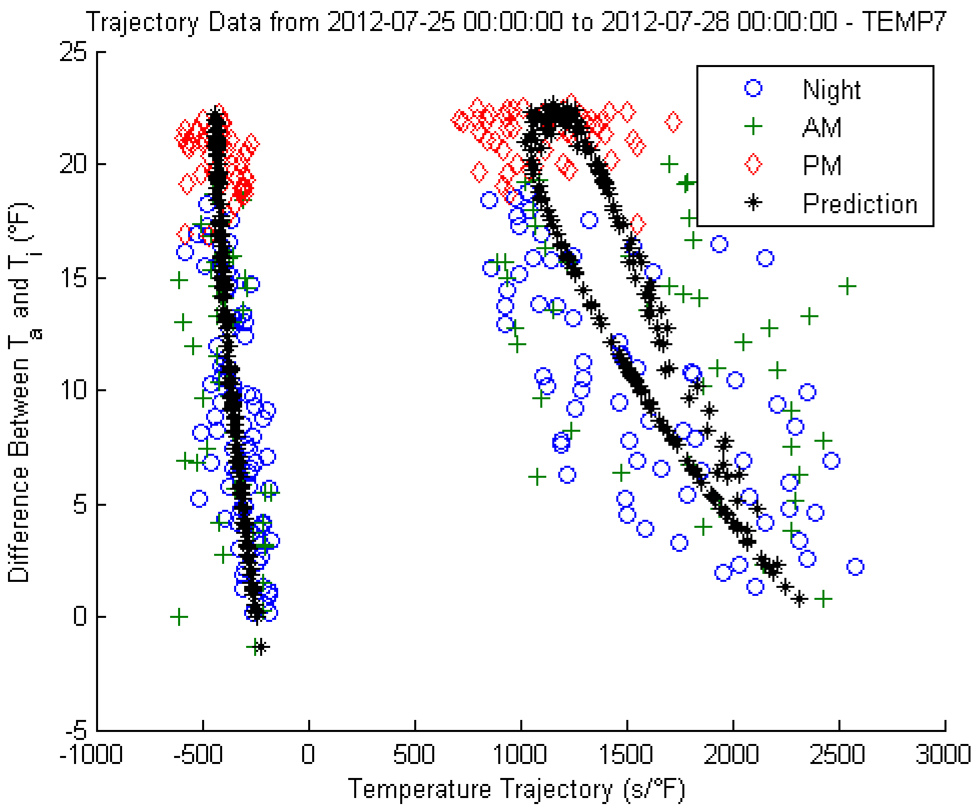


**FIGURE 48 – MODEL TRAINING RESULTS FOR TEMP7 FROM THE DATASET OF 7/26/2012-7/28/2012**

Each of these figures demonstrates slightly different problems. Beginning with Figure 48, the dataset that was used to train this particular model was taken from the same house, over the same time period, as the accurately trained model shown in Figure 47. The difference being that Figure 48 is based on TEMP7, which is the sensor located on the home's first floor. Outside of the obvious observation that the temperature trajectories are simply not nearly as well-grouped for the first floor (Figure 48) as they are for the second floor (Figure 47), there are still a couple interesting points that can be made. The temperature trajectories are simply much longer on average on the first floor. Figure 48 shows temperature trajectories as generally being between 1000 and 2500 s/°F, whereas the trajectories of Figure 47 are between 200 and 800 s/°F. Being that these models are based on sensors from within the same house, they share an air conditioner–their AC event durations have to be the same lengths. This means that the temperature changes experienced by the second floor must be higher than those on the first floor. This is confirmed by Figure 29.

This demonstrates the impact temperature swings have on the quality of the trajectories that form the training sets, and the perils of relying on non-controller-instituted temperature data. Revisiting Figure 38, the AC event durations appear to be relatively consistent. These durations form the numerator of the trajectory. What is *not* consistent is $T(t^*) - T(t)$ for each of these events, and herein lies the problem. For large numerators, small fluctuations in the denominator will result in drastically different temperature trajectories as shown in Table 21. This issue is addressed in 5.4.6.4.

**TABLE 21 – PROBLEMS CAUSED BY SMALL TEMPERATURE FLUCTUATIONS IN TRAJECTORY PREDICTIONS**

| AC Event Duration (s) | T(t*)-T(t) (°F) | Temperature Trajectory (s/°F) |
|---|---|---|
| 2100 | 1.5 | 1400 |
| 2100 | 1.1 | 1909 |

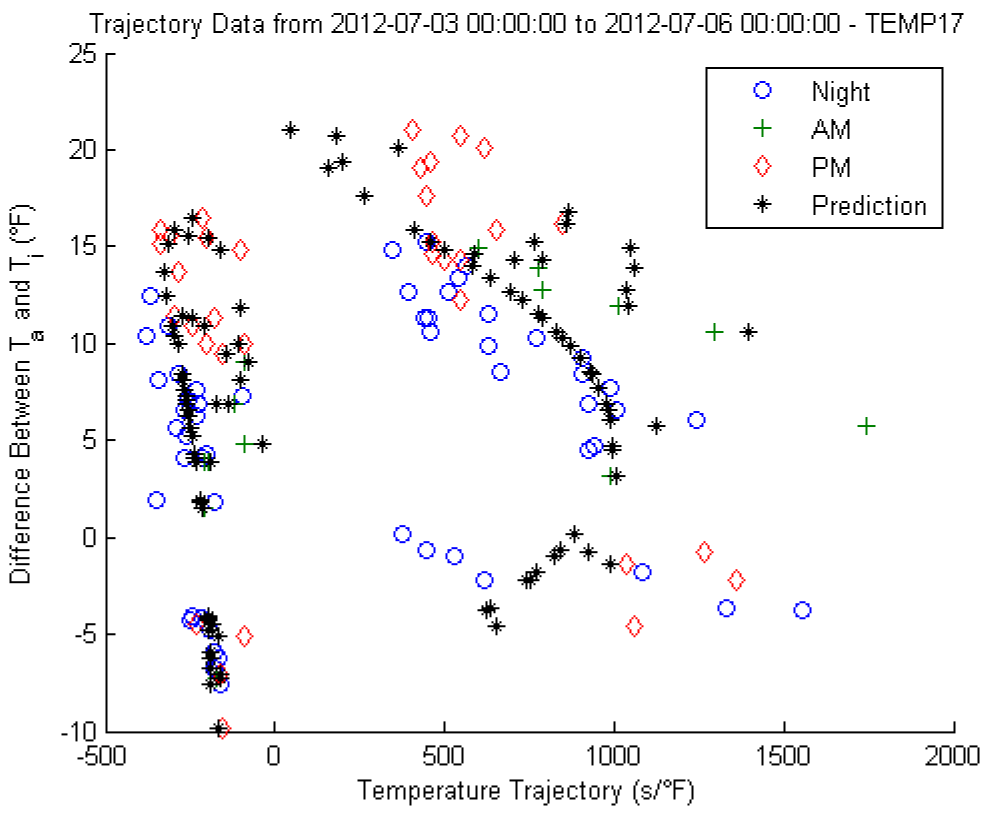


**FIGURE 49 – MODEL TRAINING RESULTS FOR TEMP17 FROM THE DATASET OF 7/2/2012-7/4/2012**

The same problems that plague the model shown in Figure 48 also plague the model shown in Figure 49, but this model also features problems of its own. It does a respectable job of curve-fitting the trajectories that correspond to the air conditioner being on, but on the other hand, the predictions for the AC-off trajectories seem to follow a path that inflects in a different direction than the other figures in the section. This is caused by the regression attempting to best-fit the data points that sit below 0 on the y-axis; they correspond to points in time where the interior temperature is *warmer* than the ambient. This intuitively makes sense that it would cause problems for the model–the direction of conductive and convective heat transfer between the house and the ambient is reversed compared to the rest of the dataset. Since the very basic regression method is predicated on curve-fitting, the more sporadic the data, the less effective this modeling approach will be.

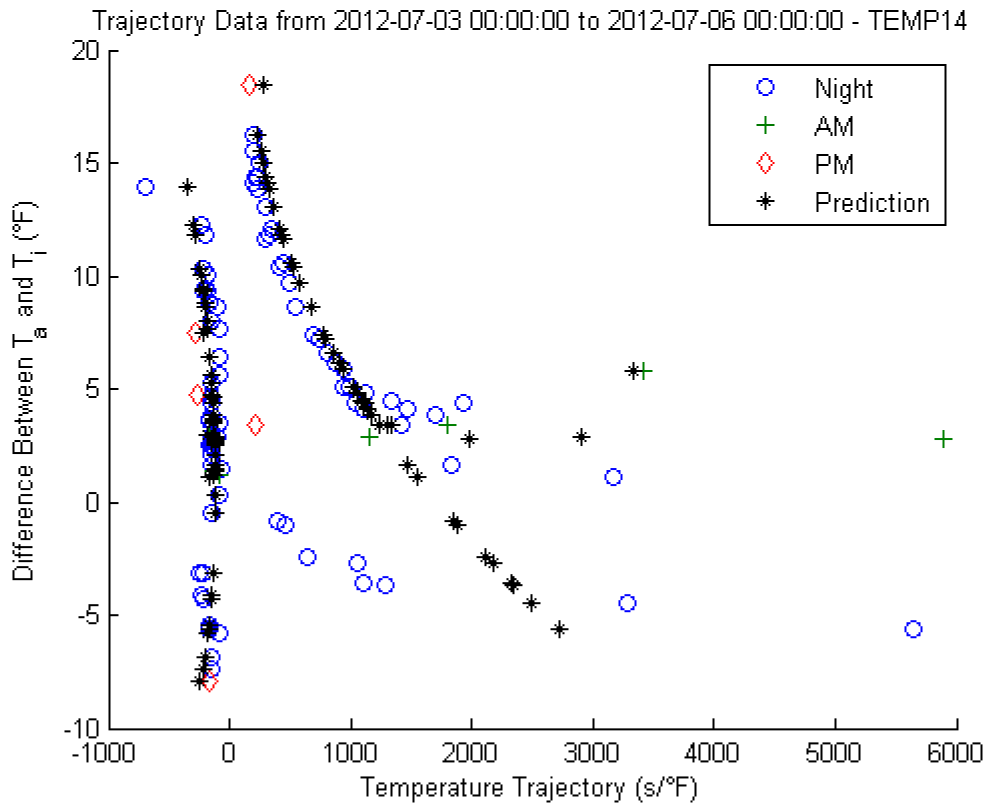


**Figure 50 – Model Training Results for TEMP14 From the Dataset of 7/3/2012-7/5/2012**

Figure 50 is affected by the same problems already mentioned, but also suffers from an extremely sparse dataset. The night-time trajectories are fairly well predicted (with the exception of the points below 0 on the y-axis), but over the three-day span of data, there are only five data points that take place during afternoon hours, and three of these five points are for AC-on trajectories. An examination of the interior temperature profile for one of the training days (Figure 51) shows that this home's AC is effectively turned off from 6:00 AM to 6:00 PM.

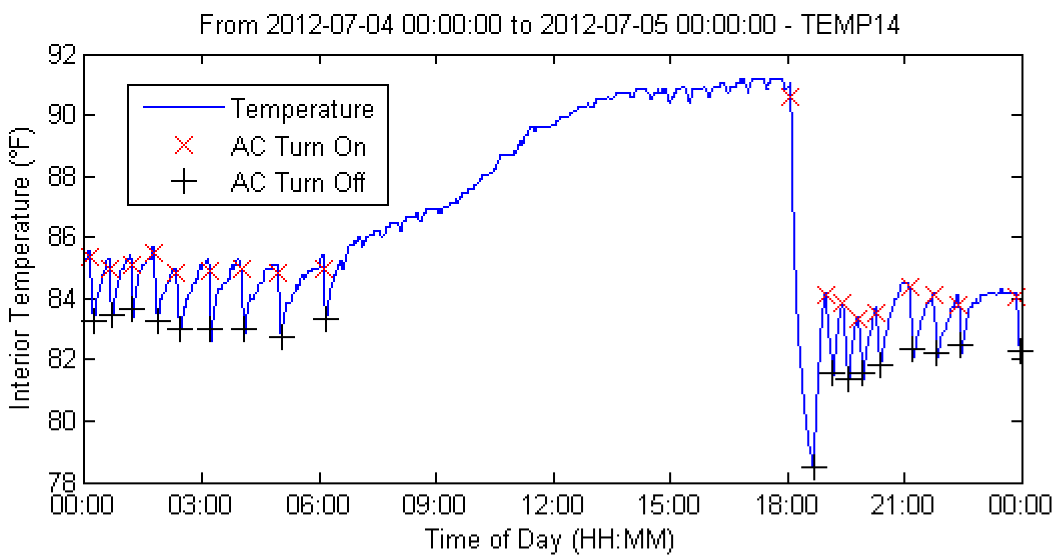


**Figure 51 – Extreme AC Setback for TEMP14 on 7/4/2012**

The regression accuracy does not suffer in this case due to the second point made at the beginning of this section–this behavior is consistent for all three days' worth of the training data. If the following day exhibits similar behavior, the model will do a decent job predicting the majority of the AC cycle durations; if the following day includes air-conditioning use throughout the afternoon, the model will significantly over-predict the duration of every AC-off cycle.

### *5.4.6.3 – Initial Model Results*

Despite the issues with the training datasets that were mentioned above, the model was still run for certain sensors, on a small sampling of summer days, as seen in Table 22. The columns within this table are partitioned to make it easy to compare the accuracy of the AC-on and AC-off cycle duration predictions. The columns are further divided to indicate the time period over which the cycle durations were averaged. The $AD$ header reflects an average from the entirety of the day, while the $PP$ columns contain averages of just those durations that occur during the peak period. Subscripts $A$ and $P$ are used to differentiate actual versus predicted, respectively.

Error columns are denoted by Δ, and show the prediction accuracies as a percentage of the actual.

For the remainder of this paper, the peak period is understood to mean the period of time from 1:30 PM through 7:00 PM. These times were selected based on the Las Vegas utility's (NV Energy) experimental pricing trial that ran from 3/14/2011 to 3/14/2013 [58], with a 30 minute lead-in to potentially accommodate pre-cooling considerations.

For the most part, there is absolutely no consistency in the accuracy of the predictions. Even for the same sensor, accuracy on a day-to-day basis fluctuates considerably. TEMP15, in particular, over-estimates AC-on durations for the first 3 dates examined, then proceeds to under-estimate the next 3. TEMP15's trained model also does a poor job of predicting AC-off trajectories, especially during the peak period. Most importantly, however, none of the sensors show peak period predictions being consistently better than all-day predictions.

**TABLE 22 – AC CYCLE DURATION PREDICTIONS**

| | TEMP8 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AC-On Average Durations (s) | | | | | | AC-Off Average Durations (s) | | | | | |
| Day | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ |
| 7/2/12 | 269 | 234 | -13.0% | 372 | 323 | -13.2% | 1097 | 1126 | 2.7% | 895 | 995 | 11.2% |
| 7/8/12 | 351 | 236 | -32.9% | 522 | 378 | -27.6% | 969 | 1160 | 19.7% | 795 | 1447 | 81.9% |
| 7/9/12 | 406 | 341 | -16.1% | 561 | 486 | -13.4% | 921 | 1020 | 10.8% | 784 | 1227 | 56.6% |
| 7/17/12 | 233 | 187 | -19.7% | 307 | 240 | -21.7% | 1302 | 1135 | -12.8% | 971 | 1161 | 19.6% |
| 7/25/12 | 300 | 293 | -2.4% | 408 | 385 | -5.6% | 1015 | 921 | -9.2% | 861 | 664 | -22.9% |
| 8/8/12 | 378 | 327 | -13.6% | 562 | 477 | -15.2% | 944 | 1017 | 7.7% | 842 | 932 | 10.6% |

| | TEMP12 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AC-On Average Durations (s) | | | | | | AC-Off Average Durations (s) | | | | | |
| Day | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ |
| 7/2/12 | 580 | 478 | -17.4% | 1096 | 679 | -38.1% | 1097 | 877 | -20.0% | 1365 | 980 | -28.2% |
| 7/8/12 | 725 | 670 | -7.6% | 732 | 824 | 12.6% | 1024 | 827 | -19.3% | 1150 | 832 | -27.6% |
| 7/9/12 | 769 | 656 | -14.7% | 1323 | 917 | -30.7% | 1016 | 803 | -20.9% | 743 | 782 | 5.2% |
| 7/17/12 | 432 | 404 | -6.4% | 784 | 522 | -33.4% | 1070 | 888 | -16.9% | 751 | 924 | 23.0% |
| 7/25/12 | 675 | 583 | -13.6% | 1304 | 873 | -33.0% | 1117 | 846 | -24.2% | 897 | 871 | -2.9% |
| 8/8/12 | 872 | 582 | -33.3% | 1914 | 957 | -50.0% | 1110 | 1115 | 0.4% | 815 | 1315 | 61.3% |

| | TEMP15 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AC-On Average Durations (s) | | | | | | AC-Off Average Durations (s) | | | | | |
| Day | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ |
| 7/2/12 | 394 | 890 | 125.6% | 571 | 1002 | 75.5% | 1361 | 1013 | -25.6% | 788 | 187 | -76.2% |
| 7/8/12 | 516 | 583 | 13.0% | 1014 | 800 | -21.1% | 896 | 990 | 10.5% | 639 | 468 | -26.8% |
| 7/9/12 | 514 | 551 | 7.2% | 781 | 801 | 2.6% | 854 | 739 | -13.5% | 597 | 542 | -9.2% |
| 7/17/12 | 314 | 76 | -75.8% | 433 | 174 | -59.9% | 1220 | 1113 | -8.8% | 930 | 1091 | 17.4% |
| 7/25/12 | 408 | 384 | -5.9% | 619 | 581 | -6.2% | 973 | 913 | -6.1% | 808 | 1157 | 43.2% |
| 8/8/12 | 560 | 538 | -4.0% | 818 | 878 | 7.4% | 1026 | 415 | -59.6% | 672 | -952 | -241.6% |

| | TEMP17 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AC-On Average Durations (s) | | | | | | AC-Off Average Durations (s) | | | | | |
| Day | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ |
| 7/2/12 | 966 | 926 | -4.1% | 892 | 1053 | 18.0% | 2838 | 2768 | -2.5% | 1811 | 1617 | -10.7% |
| 7/8/12 | 1073 | 965 | -10.1% | 1171 | 1090 | -6.9% | 2294 | 2120 | -7.6% | 1496 | 1800 | 20.4% |
| 7/9/12 | 1178 | 1028 | -12.8% | 1500 | 1235 | -17.7% | 2244 | 2413 | 7.5% | 1515 | 2685 | 77.2% |
| 7/17/12 | 846 | 927 | 9.6% | 740 | 1010 | 36.5% | 3796 | 3071 | -19.1% | 2629 | 3932 | 49.5% |
| 7/25/12 | 1012 | 1016 | 0.4% | 1035 | 1161 | 12.2% | 2634 | 2086 | -20.8% | 1792 | 1380 | -23.0% |
| 8/8/12 | 1120 | 946 | -15.6% | 1279 | 1168 | -8.7% | 2327 | 2797 | 20.2% | 1596 | 2721 | 70.5% |

### *5.4.6.4 – Training Dataset Improvements*

Remembering that the performance of this offline, curvilinear, black-box model is fundamentally reliant on how well the model is trained, this section discusses several changes that were made to the formulation of the training datasets in an attempt to improve the model's performance during the peak period.

Tackling the sources of error as they were presented in section 5.4.6.2, the first training dataset improvement deals with the issue of the sparse dataset. The purpose of the OLS method is to best-fit a curve to a collection of data points; it does this by 'weighing' each data point equally. This allows a single anomaly in the data to significantly influence the best-fit. There are two ways to handle this issue:

1. Increase the size of the dataset, so that the single anomaly has less of an impact on the OLS regression
2. Restrict the dataset to only include only worthwhile data points

Increasing the size of the dataset has negative computational implications, so, as ambiguously as the second point is presented, it is actually a more worthwhile pursuit. The results presented by Table 22 were influenced by 24-hour training data, and they reflect the model's attempt to best-fit trajectories for the entire day. Since the purpose of the model is to ultimately model the homes' thermal transience during peak periods, it makes sense to *only* train the models with data from this time period. This has no impact on the first step of the regression process, it still relies on night-time cycles, but the second step ends up being only trained by night-time cycles and those cycles that occur during the peak period. Morning cycles no longer impact the regression.

Extending the notion of only including worthwhile data points to the second source of error mentioned, the training set can also be culled of the data in which the temperature of the interior and the ambient are close. An examination of Figures 48-50, as well as several more unpublished figures, suggests that the homes' thermal responses become erratic when the interior temperature is within 5 °F, or less, of the ambient temperature.

With these changes in place, the model was rerun on TEMP8 for the same time periods as Table 22. The impact the improved dataset has on the model is substantial, as shown in Table 23. Unexpectedly, the model performed better across the entirety of the day, but it also more accurately predicted peak period AC cycle durations, as expected. The reason for the model's drastic improvement can be seen between a comparison of Figures 52-53.

**TABLE 23 – IMPACT OF TRAINING DATASET IMPROVEMENTS ON MODEL PERFORMANCE**

| | TEMP8 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AC On Average Durations | | | | | | AC Off Average Durations | | | | | |
| Day | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ |
| 7/2/12 | 269 | 216 | -19.8% | 372 | 318 | -14.6% | 1097 | 1117 | 1.8% | 895 | 979 | 9.4% |
| 7/8/12 | 351 | 355 | 1.0% | 522 | 533 | 2.0% | 969 | 1013 | 4.5% | 795 | 1056 | 32.9% |
| 7/9/12 | 406 | 374 | -7.8% | 561 | 527 | -6.0% | 921 | 960 | 4.3% | 784 | 1132 | 44.4% |
| 7/17/12 | 233 | 201 | -13.5% | 307 | 275 | -10.5% | 1302 | 1035 | -20.5% | 971 | 1029 | 6.0% |
| 7/25/12 | 300 | 296 | -1.4% | 408 | 425 | 4.3% | 1015 | 993 | -2.1% | 861 | 754 | -12.5% |
| 8/8/12 | 378 | 334 | -11.7% | 562 | 503 | -10.5% | 944 | 1043 | 10.5% | 842 | 961 | 14.1% |

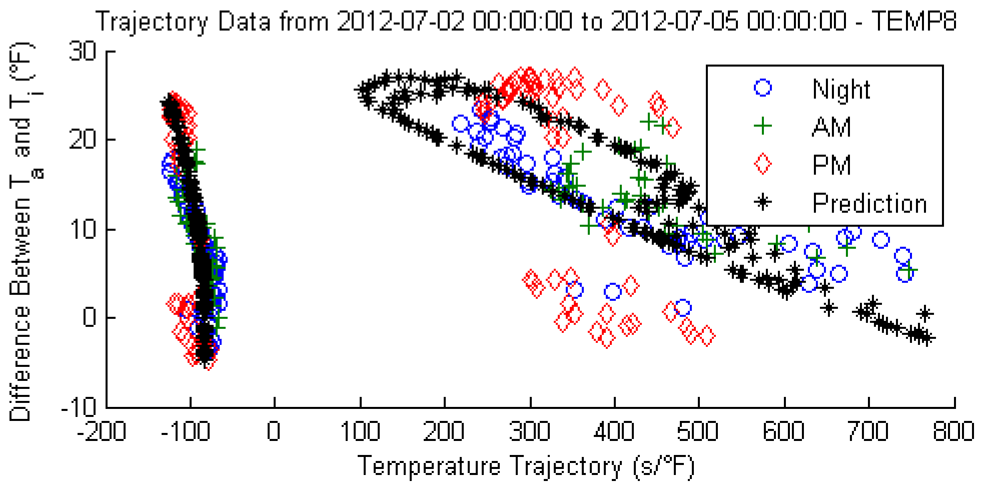


**FIGURE 52 – COMPARISON OF AN IMPROVED DATASET VERSUS ITS BASELINE (BASELINE)**

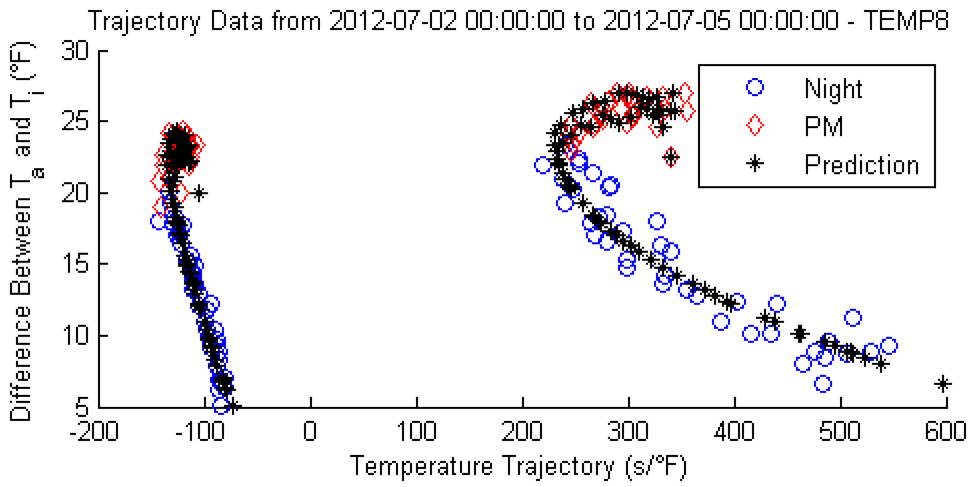


**FIGURE 53 – COMPARISON OF AN IMPROVED DATASET VERSUS ITS BASELINE (IMPROVED)**

Despite these promising results, there is still room for improvement. The last source of error suggested in section 5.4.6.2 involves the issue of fluctuating denominators in the trajectories that compose the training dataset. This problem can be further understood by taking a look at a graph of AC on/off cycle durations throughout a given day (Figure 54). As the time of day approaches late afternoon, the amount of time the air conditioner runs is on increases while the durations of its off-cycle correspondingly falls. This makes sense, as the home is requiring more cooling for the warmest part of the day. More notably, though, the trends for each cycle state are extremely consistent. The right-most plot of Figure 29 is a fairly good representation of the daily temperature trends measured by TEMP8–it features no sign of thermostat setpoint changes at any point throughout the day. This being the case, the thermostat that controls this home's AC is adhering to the same $T_{high}$ and $T_{low}$ temperature thresholds (Figure 38) throughout the entire day. Hypothetically, if this $\Delta T$ between thresholds was simply 1°F for the whole day, the day's trajectories would be equivalent to the day's cycle durations, and the trajectories would be just as consistent as the durations presented in Figure 54.

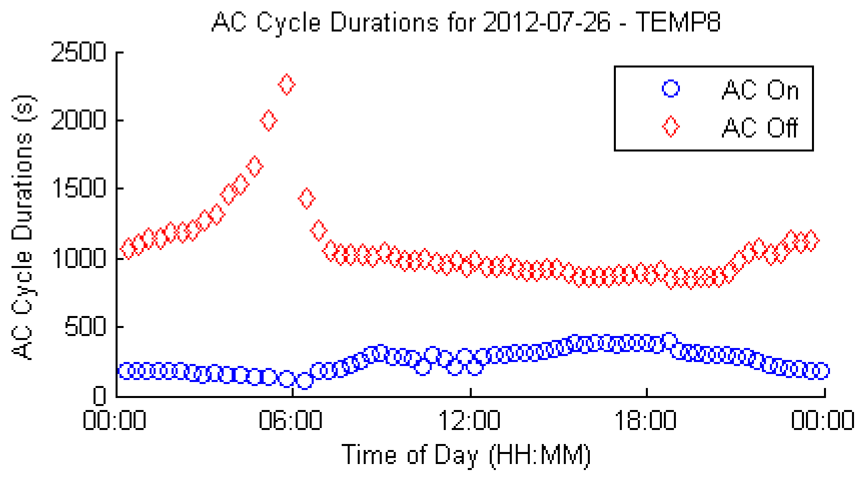
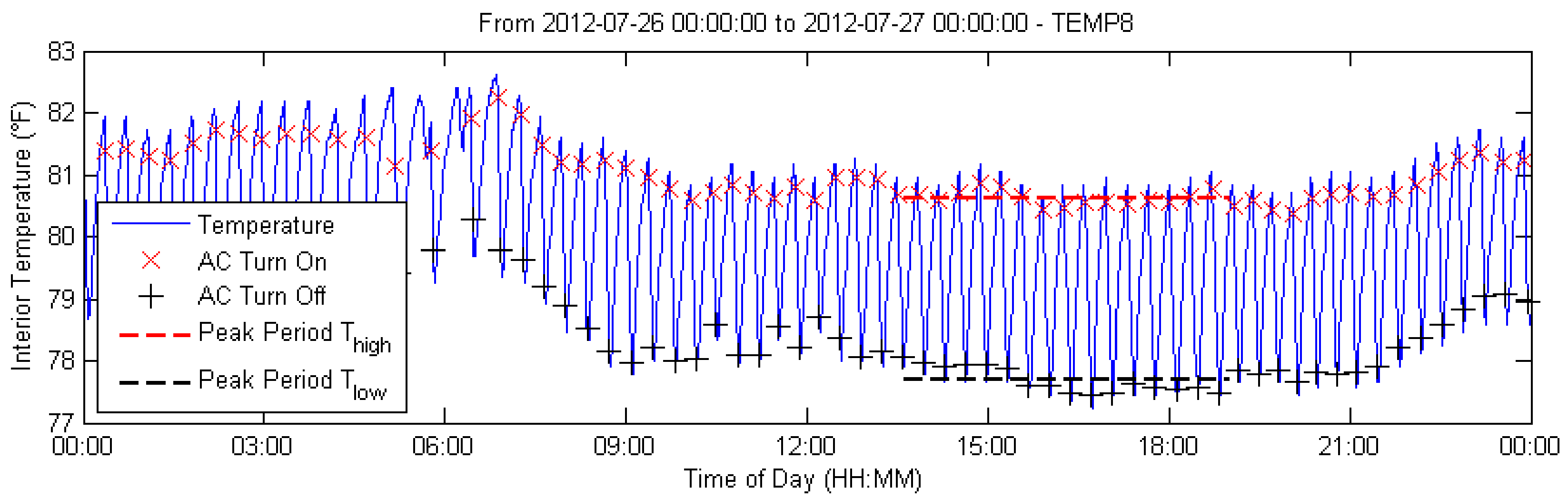


**FIGURE 54 – AC ON/OFF CYCLE DURATIONS FOR TEMP8 ON 7/26/2012**

Of course, none of the thermostats have temperature threshold differences of just 1 °F, but as long as the threshold $\Delta T$ is held constant, the regularity of the AC cycle durations will transfer to the derived trajectories, resulting in a more consistent training dataset and a subsequent improvement in the OLS regression accuracy.

In lieu of any data on the setpoints that were observed during the interior temperature measurements, approximations must be made for the $T_{high}$ and $T_{low}$ temperatures. Since the dataset has already been restricted to the peak period, the average of the AC-on and AC-off temperature measurements during this time and recalculates the temperature trajectories based on these averaged values. Two examples of this process can be seen in Figure 55.



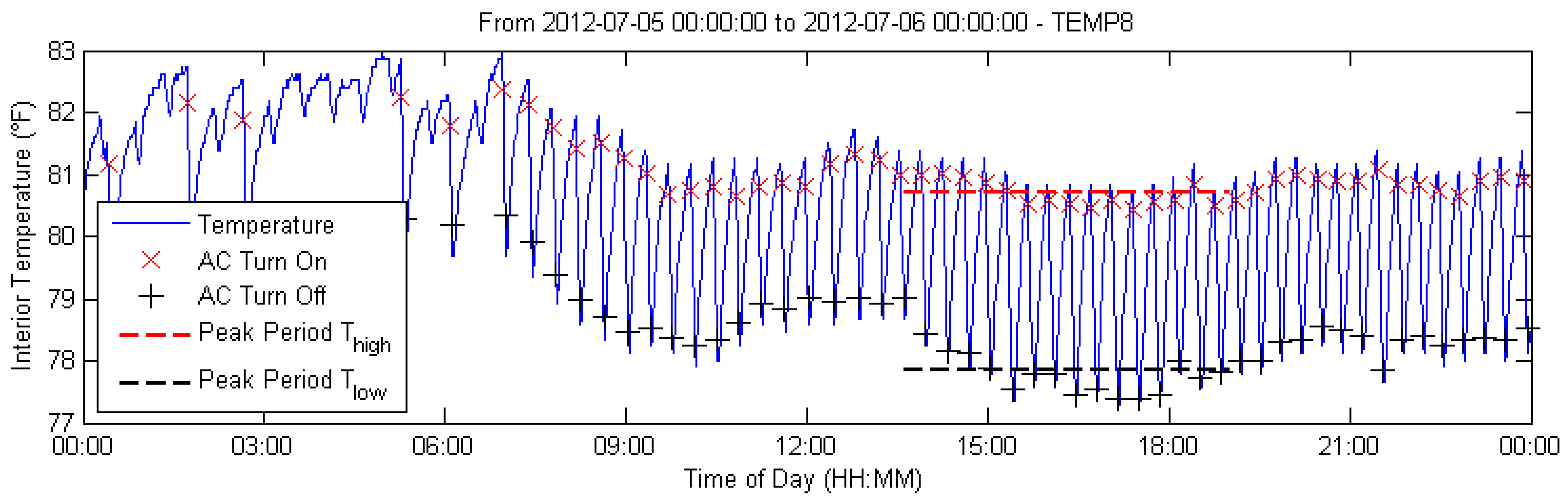


**Figure 55 – Averaged Peak Period AC-On and AC-Off Temperatures**

#### *5.4.6.5– Final Model Results*

Following the implementation of training dataset improvements that were discussed above, the program included in Appendix E was once again run for the same sensors and dates included within Table 22, so that the benefit of the training dataset improvements could be quantified. These results can be seen in Table 24.

Peak period prediction accuracies are generally improved for every day examined across every sensor, with the most notable improvements occurring for TEMP8. On average, nearly all AC-On cycle predictions for this sensor were within 8% of the actual durations. For the majority of TEMP8's AC-Off cycle predictions, the model was accurate to within 16%. TEMP12's AC-On predictions generally under-calculated the actual durations by over 30%, except for 7/8/12, in which it over-calculated by 14.2%. A review of its training dataset (Figure 56) shows inconsistent trajectories for both night-time and peak period data points, so the prediction errors are unsurprising. TEMP15 on 7/17/12 is even worse in this regard (Figure 57), but an examination of the left-most plot of Figure 58 shows that the initial interior temperature measured by the sensor was questionable. The same can be said for TEMP17, which can be seen in the right-most plot of Figure 58.

These results, though not ideal, are still encouraging. TEMP8's data, which was most closely processed to emulate event-based reporting, showed the model's ability to consistently make accurate AC cycle duration predictions within 10% of the true cycle length. Unfortunately, for the majority of the sensors, the interior temperatures they recorded could not be counted on for their accuracy; this lack of basis data makes it extremely challenging to cross-validate the results of TEMP8. At the same time, the results of TEMP8 suggest that there is some merit to this modeling approach. Further work must be done before any real conclusions can be drawn as to the model’s utility.

**TABLE 24 – AC CYCLE DURATION PREDICTIONS**

| | TEMP8 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AC On Average Durations | | | | | | AC Off Average Durations | | | | | |
| Day | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ |
| 7/2/12 | 269 | 215 | -20.1% | 372 | 326 | -12.3% | 1097 | 1086 | -0.9% | 895 | 956 | 6.7% |
| 7/8/12 | 351 | 363 | 3.2% | 522 | 537 | 2.9% | 969 | 992 | 2.3% | 795 | 1031 | 29.7% |
| 7/9/12 | 406 | 383 | -5.8% | 561 | 536 | -4.5% | 921 | 939 | 2.0% | 784 | 1097 | 40.0% |
| 7/17/12 | 233 | 210 | -9.8% | 307 | 286 | -6.8% | 1302 | 993 | -23.7% | 971 | 989 | 1.8% |
| 7/25/12 | 300 | 294 | -2.1% | 408 | 422 | 3.4% | 1015 | 971 | -4.3% | 861 | 729 | -15.3% |
| 8/8/12 | 378 | 338 | -10.6% | 562 | 518 | -7.9% | 944 | 1020 | 8.0% | 842 | 950 | 12.9% |

| | TEMP12 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AC On Average Durations | | | | | | AC Off Average Durations | | | | | |
| Day | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ |
| 7/2/12 | 580 | 479 | -17.3% | 1096 | 762 | -30.5% | 1097 | 813 | -25.8% | 1365 | 840 | -38.5% |
| 7/8/12 | 725 | 666 | -8.1% | 732 | 835 | 14.2% | 1024 | 834 | -18.6% | 1150 | 921 | -19.9% |
| 7/9/12 | 769 | 654 | -15.0% | 1323 | 866 | -34.6% | 1016 | 785 | -22.7% | 743 | 754 | 1.5% |
| 7/17/12 | 432 | 460 | 6.4% | 784 | 528 | -32.7% | 1070 | 781 | -27.0% | 751 | 759 | 1.0% |
| 7/25/12 | 675 | 578 | -14.4% | 1304 | 889 | -31.8% | 1117 | 773 | -30.8% | 897 | 778 | -13.3% |
| 8/8/12 | 872 | 602 | -31.0% | 1914 | 877 | -54.2% | 1110 | 1099 | -1.1% | 815 | 1329 | 63.0% |

| | TEMP15 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AC On Average Durations | | | | | | AC Off Average Durations | | | | | |
| Day | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ |
| 7/2/12 | 394 | 480 | 21.8% | 571 | 644 | 12.8% | 1361 | 1083 | -20.4% | 788 | 268 | -66.0% |
| 7/8/12 | 516 | 521 | 1.1% | 1014 | 789 | -22.2% | 896 | 915 | 2.1% | 639 | 341 | -46.6% |
| 7/9/12 | 514 | 495 | -3.7% | 781 | 741 | -5.1% | 854 | 648 | -24.1% | 597 | 191 | -68.0% |
| 7/17/12 | 314 | 48 | -84.6% | 433 | -35 | -108.0% | 1220 | 1790 | 46.7% | 930 | 857 | -7.8% |
| 7/25/12 | 408 | 264 | -35.2% | 619 | 443 | -28.5% | 973 | 865 | -11.1% | 808 | 988 | 22.2% |
| 8/8/12 | 560 | 606 | 8.1% | 818 | 931 | 13.9% | 1026 | 714 | -30.4% | 672 | -347 | -151.6% |

| | TEMP17 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AC On Average Durations | | | | | | AC Off Average Durations | | | | | |
| Day | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ | $AD_A$ | $AD_P$ | Δ | $PP_A$ | $PP_P$ | Δ |
| 7/2/12 | 966 | 830 | -14.1% | 892 | 829 | -7.1% | 2838 | 2966 | 4.5% | 1811 | 1729 | -4.5% |
| 7/8/12 | 1073 | 868 | -19.2% | 1171 | 896 | -23.5% | 2294 | 1934 | -15.7% | 1496 | 1790 | 19.7% |
| 7/9/12 | 1178 | 984 | -16.5% | 1500 | 1085 | -27.7% | 2244 | 2077 | -7.5% | 1515 | 2239 | 47.8% |
| 7/17/12 | 846 | 2188 | 158.6% | 740 | 994 | 34.3% | 3796 | 2261 | -40.4% | 2629 | 1823 | -30.7% |
| 7/25/12 | 1012 | 967 | -4.4% | 1035 | 1049 | 1.3% | 2634 | 2265 | -14.0% | 1792 | 1598 | -10.8% |
| 8/8/12 | 1120 | 957 | -14.5% | 1279 | 1000 | -21.8% | 2327 | 2405 | 3.4% | 1596 | 2013 | 26.1% |

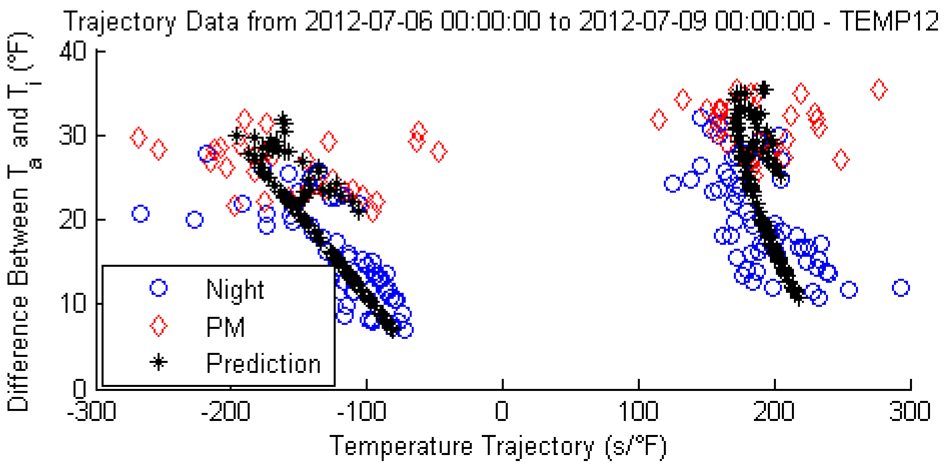


**FIGURE 56 – TRAINING DATASET FOR TEMP12 ON 7/9/2012**

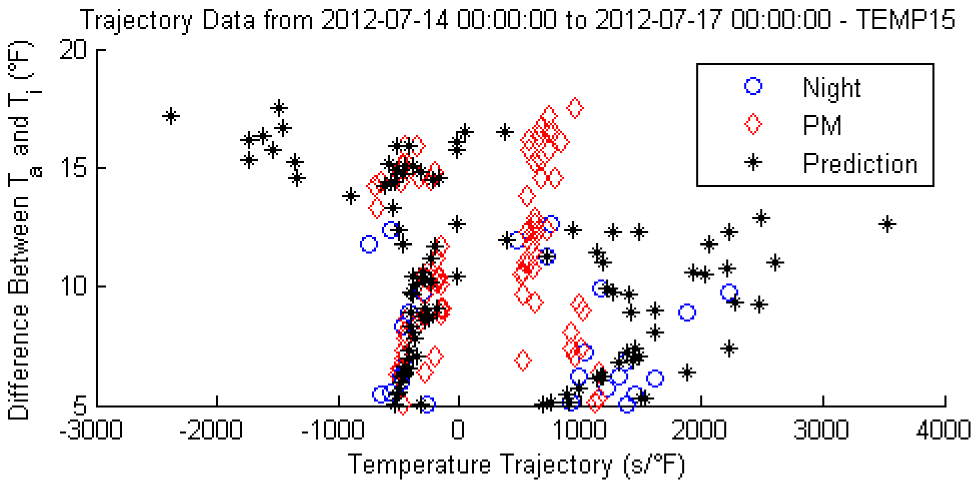


**FIGURE 57 – TRAINING DATASET FOR TEMP15 ON 7/17/2012**

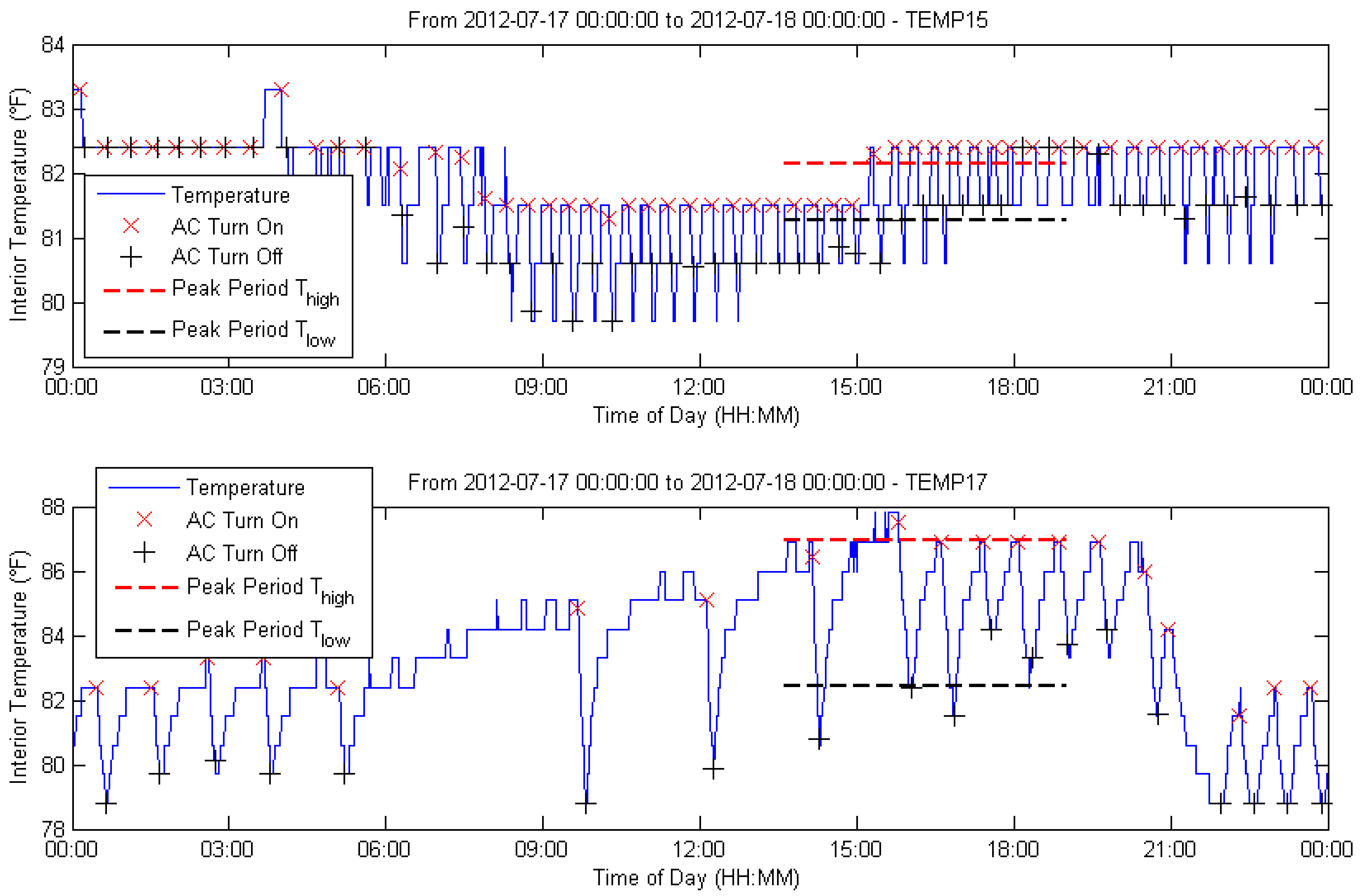


**Figure 58 – Interior Temperature Data Errors Impacting Model Results**

# Chapter 6 – Future Work

The results presented in 5.4.6.5 indicate that the black-box model that was developed was moderately effective for determining AC cycle durations for certain interiors on certain days after a significant amount of pre-processing of the data. However, the black-box model could not be validated further do to the lack of quality interior temperature measurements that resulted in wildly variable model outputs. Thus, the first task recommended for future work involves abandoning minute-by-minute temperature sensor data in favor of temperature records made by event-based thermostats. This will be achievable as the Villa Trieste project transitions from Phase I to Phase II (Chapter 3), and two-way communicating thermostats are installed within the homes.

With quality data to analyze, the curvilinear approach to model training would be able to be validated. Additionally, other, more adaptive, algorithms could be investigated for their ability to increase training efficacy and accuracy. Among the candidates for worthwhile investigation, artificial neural networks potentially show the most measurable benefit.

Once the model is validated and trained to the point that it consistently is able to make 'usefully' accurate AC cycle prediction, the model's applications are abundant. The ability to predict exactly when a home will reach a certain temperature based on current conditions can benefit electricity producers and consumers alike. With the thermal model tied into the functional operation of a controller [59], or thermostat, residents could cut down on their electrical bills by setting back their thermostats in the morning and taking advantage of the controller's model-predictive ability to return the home to a given temperature at the exact time of day specified by the resident.

Electricity producers could incorporate the thermal model into model predictive controls as well. The ability to predict exactly when a collection of air conditioners would be turning on would be extremely useful as a component to demand response initiatives [60]. When predicting time periods of critical peak loading, a utility-scale model predictive controller could quickly, and intelligently, manipulate thermostat setpoints for a certain sampling of homes to ensure the peak demand would not be reached, while at the same time only affecting homes that could still maintain their level of thermal comfort.

Potentially furthering the benefit of this research to both energy producers and consumers, the impact an air-conditioning system has on the temperature of a home could be monitored over time. If the impact were able to be quantified, and a measureable decrease in air conditioner efficiency was observed, it could provide possible early-detection equipment failure warnings to residents.

# Chapter 7 – Conclusion

Intelligent energy management is becoming an increasingly important component of the modern world. It is of particular importance for metropolitan areas in cooling-dominated climates, where energy-intensive air conditioners are prolific. The thermal model for residential buildings developed within these pages could potentially ultimately contribute to some of these important utility-wide energy management initiatives.

The thermal model is statistically based and constructed on a black-box, input-output framework. It relies on weather data, as well as interior temperature measurements and air-conditioning on/off signals, to predict how long a home's air conditioner will either stay on or off based on the amount of sunlight at a given time and the difference between the outdoor ambient temperature and the temperature within the home.

A dataset of the aforementioned measurement collected from seven homes over a period of almost two years was used to validate the model that was developed. Unfortunately, significant portions of data within this dataset were shown to be unusable, for various reasons. After a tremendous amount of pre-processing, portions of the dataset were able to be used to train the thermal model and gauge the model's efficacy. From a random selection of six days from the late-July early-August period of 2012, the model was applied to second-story zone temperature measurements from a selected house. For the most part, the model was able to predict air conditioner on-cycle durations to within 10% of their actual durations, and air conditioner off-cycles to within 30% of theirs. None of the other zones that were examined exhibited the same consistency, but they were shown to suffer from fundamentals errors associated with the collection of the interior zone temperature measurements.

Although the dataset that was available for the purposes of this research was not ideal, and proved to make it extremely difficult to validate the performance of the thermal model, there was just enough quality data to show that the thermal model could potentially be feasible. The model was built such that a future dataset that was ideally derived from thermostat measurements, rather than temperature sensors, could quickly be substituted and used for the purpose of more effectively training the model, which would ultimately lead to overall better performance of the model.

# APPENDIX A

```
%This script examined power data for each participating house over the week
%specified. It graphs the time between power records versus time. The
%purpose is to identify periods in time in which few power records were
%dropped. The total amount of time missing for each sensor during the week
%is included.

clc
clear all
warning('off','all');

tic

% ------INITIALIZE THE DATABASE

host = '[REDACTED]';  %MySQL hostname
user = '[REDACTED]';   %MySQL username
password = '[REDACTED]';    %MySQL password
dbName = '[REDACTED]'; %MySQL database name

%# JDBC parameters
jdbcString = sprintf('jdbc:mysql://%s/%s', host, dbName);
jdbcDriver = 'com.mysql.jdbc.Driver';

%# Create the database connection object
conn = database(dbName, user , password, jdbcDriver, jdbcString);

% ------ESTABLISH THE RUNTIME SPECIFICS
week = [25];
year = 2011;
NumTicks = 8;

% ------DATA PROCESSING
for m = 1:1:length(year)
    for n = 1:1:length(week)

    qry = sprintf(['SELECT powerSensor, datetime ',...
        'FROM `power` ',...
        'WHERE WEEK(datetime,0) = %u ',...
        'AND YEAR(datetime) = %u ',...
        'ORDER BY powerSensor ASC, datetime ASC'],week(n),year(m));
    rs = fetch(exec(conn, qry));
    power = get(rs, 'Data');

    if ~strcmp(power{1,1},'No Data')
```

```
power(:,3) = num2cell(datenum(cell2mat(power(:,2))));

qry = sprintf(['SELECT DISTINCT powerSensor ',...
    'FROM `power` ',...
    'WHERE WEEK(datetime,0) = %u ',...
    'AND YEAR(datetime) = %u ',...
    'ORDER BY powerSensor ASC'],week(n),year(m));
rs = fetch(exec(conn, qry));
powerSensors = get(rs, 'Data');
numSensors = length(powerSensors);

data = 0;
for i = 1:1:length(powerSensors)
%Split up each sensor
powerSeries{i} = cell2mat(power(strcmp(powerSensors(i),power(:,1)),3));

%Calculate the number of minutes between measurement recordings
for j = 2:1:length(powerSeries{i})
    powerSeries{i}(j,2) = (powerSeries{i}(j,1) - ...
                           powerSeries{i}(j-1,1))*1440;

    %Tabulate how many minutes of lost data there is
    if powerSeries{i}(j,2) > 1+(1/60)
        powerSeries{i}(j,3) = powerSeries{i}(j,2) - 1;
    end
end

%Calculate the total time lost in each day
TL(i) = round(sum(powerSeries{i}(:,3))*100)/100;
end

% ------DATA VISUALIZATION

clear title xlabel ylabel
fig = figure('Position',[200 200 575 750]);

for i = 1:1:numSensors
p{i} = subplot(numSensors,1,i);
plot(powerSeries{i}(:,1), powerSeries{i}(:,2));
set(p{i},'YLim',[0 2])
L = get(gca,'XLim');
set(gca,'XTick',linspace(L(1),L(2),NumTicks))
%legend(sprintf('%s',powerSensors{i}))
text(powerSeries{i}(end,1),0.9,sprintf('%s: %.2f minutes missing',...
     powerSensors{i},TL(i)),...
```

```
        'HorizontalAlignment','right',...
        'VerticalAlignment','top',...
        'BackgroundColor','white',...
        'EdgeColor','black',...
        'Margin',2)
    datetick('x','mm/dd','keepticks')
    set(p{i}, 'Position', [0.05, 1-(i*(1-.02)/numSensors-.05), ...
                           0.91, (1-.1)/numSensors-.05])
    end

    drawnow
    set(gcf,'PaperPositionMode','auto')
    print(fig, '-dpng', ...
          sprintf('[REDACTED]\\power_quality\\%u-%u.png',week(n),year(m)));

    end

    end
end

toc
```

# APPENDIX B

```
%This takes a look at all the houses simultaneously for the dates
%specified. It makes two plots for each day, showing total power
%consumption, utilization as a percentage of the total number of AC units,
%and the ambient temperature

clc
clear all
warning('off','all');

tic

%% ------INITIALIZE THE DATABASE

host = '[REDACTED]';  %MySQL hostname
user = '[REDACTED]';   %MySQL username
password = '[REDACTED]';    %MySQL password
dbName = '[REDACTED]'; %MySQL database name

%# JDBC parameters
jdbcString = sprintf('jdbc:mysql://%s/%s', host, dbName);
jdbcDriver = 'com.mysql.jdbc.Driver';

%# Create the database connection object
conn = database(dbName, user , password, jdbcDriver, jdbcString);

%% ------ESTABLISH THE RUNTIME SPECIFICS

%Sensor IDs
%Temperature, Power
devices = [{'TEMP5' 'PWR4'};
           {'TEMP6' 'PWR4'};
           {'TEMP7' 'PWR7'};
           {'TEMP8' 'PWR7'};
           {'TEMP9' 'PWR6'};
           {'TEMP10' 'PWR6'};
           {'TEMP11' 'PWR8'};
           {'TEMP12' 'PWR8'};
           {'TEMP13' 'PWR3'};
           {'TEMP14' 'PWR3'};
           {'TEMP15' 'PWR2'};
           {'TEMP16' 'PWR2'};
           {'TEMP17' 'PWR5'};
           {'TEMP18' 'PWR5'}];

%Current Transducer Ratings for each power measurement box
```

```
%WattNode, Main In, Main Out, Fan control unit, Air conditioner, PV
CTsizes = [{'PWR4' 150 0 30 50 30};
           {'PWR7' 150 30 50 150 50};
           {'PWR6' 150 0 30 50 30};
           {'PWR8' 150 30 50 150 50};
           {'PWR3' 150 30 50 150 50};
           {'PWR2' 150 30 50 150 50};
           {'PWR5' 150 30 50 50 50}];

%Wh per pulse = CT Size/pulseConvert
pulseConvert = 40; %Set by WNB-3Y-208-P WattNode

%The day to examine
% startDay = '2012-07-01';
% numberOfDays = 5;
dates = {'2012-07-02','2012-07-08','2012-07-09',...
         '2012-07-17','2012-07-22','2012-08-08'};

% for k = 0:1:numberOfDays
for k = 1:1:length(dates)
date = dates{k};

dateRange = {datestr(date,'yyyy-mm-dd HH:MM:SS'),
             datestr(addtodate(datenum(date),1,'day'),...
             'yyyy-mm-dd HH:MM:SS')};

%% ------DATA PROCESSING
%data4: AC WattNode pulses
qry = sprintf(['SELECT powerSensor, datetime, data4 ',...
              'FROM `power` ',...
              'WHERE datetime BETWEEN ''%s'' and ''%s'' ',...
              'ORDER BY powerSensor ASC, datetime ASC'],...
              dateRange{1},dateRange{2});
rs = fetch(exec(conn, qry));
power = get(rs, 'Data');

qry = sprintf(['SELECT DISTINCT powerSensor ',...
              'FROM `power` ',...
              'WHERE datetime BETWEEN ''%s'' and ''%s'' ',...
              'ORDER BY powerSensor ASC'],...
              dateRange{1},dateRange{2});
rs = fetch(exec(conn, qry));
powerSensors = get(rs, 'Data');

qry = sprintf(['SELECT datetime, ambientTemp ',...
            'FROM `weatherAll` ',...
            'WHERE datetime BETWEEN ''%s'' and ''%s'''],...
            dateRange{1},dateRange{2});
rs = fetch(exec(conn, qry));
weather = get(rs, 'Data');
time = timeseries(cell2mat(weather(:,end)),weather(:,1));
time.name = 'time';

data = 0;
for i = 1:1:length(powerSensors)
%Split up each sensor so that they can be synched
powerSeries{i} = power(strcmp(powerSensors(i),power(:,1)),:);

%Convert the pulses to Wh
CT = cell2mat(CTsizes(strcmp(CTsizes(:,1),powerSensors(i)),5));
WhPP = CT/pulseConvert; %Watt-hours per pulse
powerSeries{i}(:,4)=num2cell(cell2mat(powerSeries{i}(:,3)).*WhPP); %Wh
powerSeries{i}(:,5)=num2cell(cell2mat(powerSeries{i}(:,4)).*60./1000); %KW

%On/Off
powerSeries{i}(:,6)=num2cell(double(cell2mat(powerSeries{i}(:,4)).*60~=0));

%Make the timeseries
ts{i,1}=timeseries(cell2mat(powerSeries{i}(:,3:end)),powerSeries{i}(:,2));
ts{i,2}=timeseries(cell2mat(weather(:,end)),weather(:,1));
ts{i,1}.name = powerSensors{i};
ts{i,2}.name = 'weather';

%Sync the weather to the power record
ts{i,2} = resample(ts{i,2},getabstime(ts{i,1}),'linear');

  %Identify each cycle, calculate the average power demand and the average
  %ambient and interior temperatures over that time
  j=2;
  m=1;
  while j < length(ts{i,1}.data)
    %Identify when the pulses start
    if ts{i,1}.data(j,3) ~= 0 && ts{i,1}.data(j-1,3) == 0
      ss(1) = j+1; %starting steady-state index
        %Shift the index up until the AC switches OFF
        while ts{i,1}.data(j,3) ~= 0 && j < length(ts{i,1}.data)
            j=j+1;
        end
      ss(2) = j-2; %ending steady-state index
      %Average power required and the ambient temp. over the same period
      ts{i,3}{m,1} = mean(ts{i,1}.data(ss(1):ss(2),3));
      ts{i,3}{m,2} = mean(ts{i,2}.data(ss(1):ss(2),1));
      m=m+1;
    end
    j=j+1;
  end

%Plot the pulses consumption versus the ambient temperature
fig1 = figure;
set(gcf,'PaperPositionMode','auto')
set(fig1, 'Position', [400 200 440 230])
scatter(cell2mat(ts{i,3}(:,1)),cell2mat(ts{i,3}(:,2)),'Marker','*')
title(sprintf('Effect of Ambient Temp. on AC Power Draw for %s (%s)',...
               date,powerSensors{i}))
ylabel('Ambient Temperature (°F)')
xlabel('Average Power Demand Per AC Cycle (kW)')
drawnow
print(fig1,'-dpng','-r100',...
      sprintf('[REDACTED]\\AC-Eff\\%s-%s-(EFF).png',powerSensors{i},date));

%Sync the power record to the common weather record
%so that all the power records can be added
ts{i,1} = resample(ts{i,1},getabstime(time),'linear');

%Sum the power and energy measurements
data = data + ts{i,1}.data(:,2:end);
end

%% ------DATA VISUALIZATION

% if max(data(:,3)./size(powerSensors,1)) == 1

clear title xlabel ylabel
fig2 = figure('Position',[200 200 550 300]);
set(gcf,'PaperPositionMode','auto')

p1 = subplot(2,1,1);
plot(datenum(getabstime(ts{i,1})),data(:,2));
ylabel('Power (kW)');
datetick('x','HH:MM')
title(sprintf('Measured Homes'' Power Consumption - %s',date))

p2=subplot(2,1,2);
p2=plotyy(datenum(getabstime(ts{i,1})),...
          100*data(:,3)./size(powerSensors,1),...
          datenum(getabstime(ts{i,2})),ts{i,2}.data,'plot');

ylabel(p2(1),'% of AC''s On') % left y-axis
ylabel(p2(2),'Temperature (°F)') % right y-axis

xlabel('Time (HH:MM)');
datetick('x','HH:MM')
set(p2,'xtick',get(p1,'xtick'),'xticklab',get(p1,'xticklab'))
title('Measured Homes'' Air Conditioner Operating Status')

set(p1, 'Position', [0.09, 0.60, 0.83, 0.32])
set(p2, 'Position', [0.09, 0.12, 0.83, 0.32])
drawnow
print(fig2,'-dpng','-r100',...
      sprintf('[REDACTED]\\CollectivePower\\%s-(CP).png',date));

%end

end
toc
```

# APPENDIX C

```
%This script compares two methods developed for determining the
%timestamps of AC on/off transition points.

clc
clear all
warning('off','all');

tic

%% ------INITIALIZE THE DATABASE

host = '[REDACTED]';  %MySQL hostname
user = '[REDACTED]';   %MySQL username
password = '[REDACTED]';    %MySQL password
dbName = '[REDACTED]'; %MySQL database name

%# JDBC parameters
jdbcString = sprintf('jdbc:mysql://%s/%s', host, dbName);
jdbcDriver = 'com.mysql.jdbc.Driver';

%# Create the database connection object
conn = database(dbName, user , password, jdbcDriver, jdbcString);

%% ------ESTABLISH THE RUNTIME SPECIFICS

%Sensor IDs
%Temperature, Power
devices = [{'TEMP5' 'PWR4'};
           {'TEMP6' 'PWR4'};
           {'TEMP7' 'PWR7'};
           {'TEMP8' 'PWR7'};
           {'TEMP9' 'PWR6'};
           {'TEMP10' 'PWR6'};
           {'TEMP11' 'PWR8'};
           {'TEMP12' 'PWR8'};
           {'TEMP13' 'PWR3'};
           {'TEMP14' 'PWR3'};
           {'TEMP15' 'PWR2'};
           {'TEMP16' 'PWR2'};
           {'TEMP17' 'PWR5'};
           {'TEMP18' 'PWR5'}];

%Current Transducer Ratings for each power measurement box
%WattNode, Main In, Main Out, Fan control unit, Air conditioner, PV
CTsizes = [{'PWR4' 150 0 30 50 30};
```

```
           {'PWR7' 150 30 50 150 50};
           {'PWR6' 150 0 30 50 30};
           {'PWR8' 150 30 50 150 50};
           {'PWR3' 150 30 50 150 50};
           {'PWR2' 150 30 50 150 50};
           {'PWR5' 150 30 50 50 50}];

%Wh per pulse = CT Size/pulseConvert
pulseConvert = 40; %Set by WNB-3Y-208-P WattNode

%The day to examine
% startDay = '2012-07-01';
% numberOfDays = 5;
dates = {'2012-07-02','2012-07-08','2012-07-09','2012-07-17',...
         '2012-07-22','2012-08-08'};

% for k = 0:1:numberOfDays
for k = 1:1:length(dates)
date = dates{k};
%date = datestr(addtodate(datenum(startDay),k,'day'),'yyyy-mm-dd');

dateRange = {datestr(date,'yyyy-mm-dd HH:MM:SS'),...
             datestr(addtodate(datenum(date),1,'day'),...
             'yyyy-mm-dd HH:MM:SS')};

%% ------DATA PROCESSING
%data4: AC WattNode pulses
qry = sprintf(['SELECT powerSensor, datetime, data4 ',...
              'FROM `power` ',...
              'WHERE datetime BETWEEN ''%s'' and ''%s'' ',...
              'ORDER BY powerSensor ASC, datetime ASC'],...
              dateRange{1},dateRange{2});
rs = fetch(exec(conn, qry));
power = get(rs, 'Data');

qry = sprintf(['SELECT DISTINCT powerSensor ',...
              'FROM `power` ',...
              'WHERE datetime BETWEEN ''%s'' and ''%s'' ',...
              'ORDER BY powerSensor ASC'],...
              dateRange{1},dateRange{2});
rs = fetch(exec(conn, qry));
powerSensors = get(rs, 'Data');

qry = sprintf(['SELECT tempSensor, datetime, temp ',...
              'FROM `temperature` ',...
              'WHERE datetime BETWEEN ''%s'' and ''%s'' ',...
              'ORDER BY tempSensor ASC'],...
              dateRange{1},dateRange{2});
rs = fetch(exec(conn, qry));
temps = get(rs, 'Data');

%Options for the temperature AC status estimate
window = 15; %number of samples (must be odd)
windowSide = (window-1)/2;

n = 1;
for i = 1:1:length(powerSensors)
  powerSensor = powerSensors(i);

  %Split up each sensor individually
  series(i,1) = powerSensor;
  series{i,2} = power(strmatch(powerSensor,power(:,1)),2:3);

  %% ------STATE BASED ON AC TRANSDUCER

  %Go through the routine that identifies "shoulders" in the power data and
  %attempts to ascertain the timestamps of the ON/OFF transitions
  j = 2; %index
  state = 1; %ON/OFF state of the AC
  while j < length(series{i,2})
    %Identify the time period in which the AC switches ON
    if (series{i,2}{j-1,2} == 0 || j == 2)&& series{i,2}{j,2} ~= 0

      k(1) = j+1; %starting steady-state index

      %Shift the index up until the AC switches OFF
      while series{i,2}{j,2} ~= 0 && j < length(series{i,2})
        j = j+1;
      end
      k(2) = j-2; %ending steady-state index

      if k(1) <= k(2)
          %Number of pulses during the steady-state period of the cycle
          Puls = sum(cell2mat(series{i,2}(k(1):k(2),2)));

          %Seconds per pulse based on steady-state performance
          sp = etime(datevec(series{i,2}{j-1,1}),...
                     datevec(series{i,2}{k(1),1}))/Puls;

          %Create the "simplified" AC event transition record
          %Event Start
          event = addtodate(datenum(series{i,2}{k(1),1}),...
                                   -sp*series{i,2}{k(1)-1,2},'second');

          if n == 1
            series{i,3}{n,1} = datestr(event,'yyyy-mm-dd HH:MM:SS');

            %Event Stop
            event = addtodate(datenum(series{i,2}{k(2),1}),...
                                      sp*series{i,2}{k(2)+1,2},'second');
            series{i,3}{n,2} = datestr(event,'yyyy-mm-dd HH:MM:SS');
            series{i,3}{n+1,1} = datestr(event,'yyyy-mm-dd HH:MM:SS');

            series{i,3}{n,3} = state;
            state = ~state;
          else
            series{i,3}{n-1,2} = datestr(event,'yyyy-mm-dd HH:MM:SS');
            series{i,3}{n,1} = datestr(event,'yyyy-mm-dd HH:MM:SS');

            series{i,3}{n-1,3} = state;
            state = ~state;

            %Next Event Start
            event = addtodate(datenum(series{i,2}{k(2),1}),...
                                      sp*series{i,2}{k(2)+1,2},'second');
            series{i,3}{n,2} = datestr(event,'yyyy-mm-dd HH:MM:SS');
            series{i,3}{n+1,1} = datestr(event,'yyyy-mm-dd HH:MM:SS');

            series{i,3}{n,3} = state;
            state = ~state;
          end

          n=n+2;
      end
    end


    j = j+1;
  end

  %Trim off the last incomplete record
  series{i,3} = series{i,3}(1:end-1,:);

  n=1;
```

```
%% ------STATE BASED ON TEMPERATURE

%Which temperature sensors match the power sensor being examined
tempSensors = devices(strmatch(powerSensor,devices(:,2)),1);

for ii = 1:1:length(tempSensors)

  j = 1;
  passInitial = 0;
  k = 1;
  state = 0;

  tempSensor = tempSensors(ii);

  %Split up each sensor individually
  series(i,3*ii+1) = tempSensor;
  series{i,3*ii+2} = temps(strmatch(tempSensor,temps(:,1)),2:3);
  %plot(datenum(cell2mat(series{1,5}(:,1))),cell2mat(series{1,5}(:,2)))

  for p = window:1:length(series{i,3*ii+2})

  %Status indicates whether the air conditioner is on or off
  %Determine if a significant downward trend starts at the given loop
      %level by comparing the previous slope to the upcoming slope
  if state == 0 && ...
          series{i,3*ii+2}{p,2}-series{i,3*ii+2}{p-(window-1),2} < -1.15

      j = p;
      while state == 0
      if j ~= p
          %If the drop in temperature after the given point is
          %significantly more than the drop in temperature
          %before the given point
          if j == 1
              state = 1;
          elseif series{i,3*ii+2}{j+1,2}-series{i,3*ii+2}{j,2} > 0 && ...
                  series{i,3*ii+2}{j-1,2} < series{i,3*ii+2}{j,2} + ...
                  0.75*(series{i,3*ii+2}{j+1,2}-series{i,3*ii+2}{j,2})

            series{i,3*ii+3}{k,1} = series{i,3*ii+2}{j+1,1};
            if k ~= 1
              series{i,3*ii+3}{k-1,2} = series{i,3*ii+2}{j+1,1};
            end
            series{i,3*ii+3}{k,3} = ~state;
            state = 1;
            k = k+1;
          elseif series{i,3*ii+2}{j+1,2}-series{i,3*ii+2}{j,2} == 0
          %If the sample has plateaued, look further back into the
          %samples to find out if it's truly plateaued
          m = j;
          while m ~= 0 && series{i,3*ii+2}{m+1,2} - ...
                          series{i,3*ii+2}{m,2} == 0
             m = m-1;
          end
          m = m+1;

          if m-windowSide < 1
              state = 1;
          elseif series{i,3*ii+2}{m-windowSide,2} < ...
                  series{i,3*ii+2}{m,2} | ...
                  series{i,3*ii+2}{m-1,2} < ...
                  series{i,3*ii+2}{m,2} + 0.75*(series{i,3*ii+2}{p,2}-...
                  series{i,3*ii+2}{p+1,2})
              series{i,3*ii+3}{k,1} = series{i,3*ii+2}{j+1,1};
              if k ~= 1
                series{i,3*ii+3}{k-1,2} = series{i,3*ii+2}{j+1,1};
              end
              series{i,3*ii+3}{k,3} = ~state;

              state = 1;
              k = k+1;
          end
          end
      end

      j = j-1; %decrease the index that looks at the temperature
      end


  elseif state == 1 && series{i,3*ii+2}{p,2} - ...
                       series{i,3*ii+2}{p-(window-1),2} > 0

  %Find the local minimum value. It has already been established
  %that a local minimum exists
  indexOfMin = find(cell2mat(series{i,3*ii+2}(p:-1:p-(window-1),2))==...
          min(cell2mat(series{i,3*ii+2}(p:-1:p-(window-1),2))),1,'last');

  series{i,3*ii+3}{k,1} = series{i,3*ii+2}{p-indexOfMin+1,1};
  if k ~= 1
```

```
      series{i,3*ii+3}{k-1,2} = series{i,3*ii+2}{p-indexOfMin+1,1};
    end
    series{i,3*ii+3}{k,3} = ~state;

    state = 0;
    k = k+1;

    end
    end

  end

  %% ------DATA VISUALIZATION

  %Plot temperature measurements vs power transducer-derived transitions
  %for each power sensor
  fig(i) = figure;
  set(fig(i),'Position',[200 200 900 275]);
  %Power
  line(datenum(series{i,2}(:,1)),cell2mat(series{i,2}(:,2)),'color','k')
  haxes1 = gca; % handle to axes
  datetick('x','HH:MM')
  xlabel('Time of Day (HH:MM)');

  haxes1_pos = get(haxes1,'Position'); % store position of first axes
  haxes2 = axes('Position',haxes1_pos,...
              'YAxisLocation','right',...
              'Color','none',...
              'YColor','b');
  set(haxes2,'XTick',[])

  %Temperatures
  %line(datenum(series{i,5}(:,1)),cell2mat(series{i,5}(:,2)),...
  %             'Parent',haxes2,'color','b') %First floor
  line(datenum(series{i,8}(:,1)),cell2mat(series{i,8}(:,2)),...
             'Parent',haxes2,'color','b') %Second floor

  ylabel(haxes1,'# of Wattnode Pulses for the AC') % left y-axis
  ylabel(haxes2,'Interior Temperature (°F)') % right y-axis

  title(sprintf('Temperature vs. AC Status (%s, %s) -
%s',series{i,1},series{i,7},date))
  drawnow

  set(gcf,'PaperPositionMode','auto')
  print(fig(i), '-dpng', ...
        sprintf('[REDACTED]\\justify_AC_status\\(%s)-%s-%s.png',date,...
                 cell2mat(powerSensor),cell2mat(tempSensor)))

end

end
toc
```

# APPENDIX D

```
%Establish the regression method for best-fitting a curve to determined
%trajectories
clc
clear all
tic

%% ------INITIALIZE THE DATABASE

host = '[REDACTED]';  %MySQL hostname
user = '[REDACTED]';   %MySQL username
password = '[REDACTED]';    %MySQL password
dbName = '[REDACTED]'; %MySQL database name

%# JDBC parameters
jdbcString = sprintf('jdbc:mysql://%s/%s', host, dbName);
jdbcDriver = 'com.mysql.jdbc.Driver';

%# Create the database connection object
conn = database(dbName, user , password, jdbcDriver, jdbcString);

%% ------ESTABLISH THE RUNTIME SPECIFICS

%The required time data - it takes the day that's about to be predicted and
%performs the regression on the previous 3 days
regressionDates = {
    {'2012-07-30'}
};

%Sensor List
devices = {'TEMP7'};

figureCount = 1;
for datesCount = 1:1:length(regressionDates)

date = regressionDates{datesCount};
dateRange = {datestr(addtodate(datenum(date),-3,'day'),...
             'yyyy-mm-dd HH:MM:SS'),
             datestr(date, 'yyyy-mm-dd HH:MM:SS'),
             datestr(addtodate(datenum(date),1,'day'),...
             'yyyy-mm-dd HH:MM:SS')};

for deviceCount = 1:1:length(devices)

tempSensor = devices{deviceCount};
```

```
%-------------[INTERIOR TEMPERATURE DATA]------------
qry = sprintf(['SELECT datetime, temp ',...
               'FROM `temperature` ',...
               'WHERE datetime BETWEEN ''%s'' and ''%s'' ',...
               'AND tempSensor = ''%s'''],...
               dateRange{1},dateRange{2},tempSensor);
rs = fetch(exec(conn, qry));
thermostat = get(rs, 'Data');

%-------------[SIGNATURE]----------------------------
%Pull the complete given device's data over the regression period
qry = sprintf(['SELECT startTempIn, invSlope, duration, ',...
                      '(startTempIn-endTempIn), startRad, ',...
                      'startTempAmb, mode, periodStart ',...
               'FROM `signaturePwr` ',...
               'WHERE tempSensor = ''%s'' ',...
               'AND periodStart BETWEEN ''%s'' AND ''%s'' ',...
               'ORDER BY periodStart'],...
               tempSensor,dateRange{1},dateRange{2});
rs = fetch(exec(conn, qry));
sig = get(rs, 'Data');
%The mode comes in as logical
sig = [cell2mat(sig(:,1:6)),cell2mat(sig(:,7)),datenum(sig(:,8))];

%Remove the samples with daylight and when the AC is running
sigNight = sig(~logical(sig(:,7)),:); %When the AC is off
sigNight = sigNight(~(sigNight(:,5) > 5),:); %When the GHI is over 5

%------------[NIGHT REGRESSION]------------
%There are some slopes that really throw off the data. Remove the rows that
%are more than 1.5 standard deviations from the median
dev = 1.5; %Standard Deviation level

outliers = abs(sigNight(:,2)-median(sigNight(:,2)))>dev*std(sigNight(:,2));
sigNight = sigNight(~outliers,:);

choice = input('Perform (l)inear or (c)urvilinear regression? [c]:','s');

if strcmp(choice,'l') %Linear
    X = [sigNight(:,6)-sigNight(:,1),ones(size(sigNight,1),1)];
else %Curvilinear
    X = [(sigNight(:,6)-sigNight(:,1)).^2,sigNight(:,6)-sigNight(:,1),...
          ones(size(sigNight,1),1)];
end
```

```
y = sigNight(:,2); %inverse slope
[b,~,~,~,~] = regress(y,X);

%The WARM UP trajectories vs the actual temp sensor readings for the NIGHT
fig1 = figure;
set(gcf,'PaperPositionMode','auto')
set(fig1, 'Position', [100 100 440 230])
%--------AmbientTemp---startT_in--vs--invSlope
scatter(sigNight(:,6)-sigNight(:,1),sigNight(:,2))
hold on

if strcmp(choice,'l') %Linear
    scatter(sigNight(:,6)-sigNight(:,1),...
            b(1)*(sigNight(:,6)-sigNight(:,1)) + b(2),'r*')
else %Curvilinear
    scatter(sigNight(:,6)-sigNight(:,1),...
            b(1)*(sigNight(:,6)-sigNight(:,1)).^2 + ...
            b(2)*(sigNight(:,6)-sigNight(:,1)) + b(3),'r*')
end

title(sprintf('Night Warming from %s to %s - %s',...
      dateRange{1},dateRange{2},tempSensor))
ylabel('Temperature Trajectory (s/°F)')
xlabel('Difference in Temperature Between T_a and T_i (°F)')
legend('Data','Predictions')

%Store the results of the first regression test
coeff = b; %alpha, beta, gamma


%------------[FULL DAY REGRESSION]------------
%Same thing, remove the outliers, but remove them seperately for ON and OFF
%AC operation modes
statusOn = sig(logical(sig(:,7)),:);
statusOff = sig(~logical(sig(:,7)),:);

outlierOn = abs(statusOn(:,2) - median(statusOn(:,2))) > ...
            dev*std(statusOn(:,2));
outlierOff = abs(statusOff(:,2) - median(statusOff(:,2))) > ...
             dev*std(statusOff(:,2));

statusOn = statusOn(~outlierOn,:);
statusOff = statusOff(~outlierOff,:);

%Illustrates the trajectory groupings for the on/off AC states
```

```
sig = cat(1,statusOn,statusOff); %Combines the two matricies
filter = logical([]);
filter(:,1) = sig(:,5) < 5; %No radiation (night)
filter(:,2) = ~filter(:,1) & hour(sig(:,8)) < 12; %AM
filter(:,3) = ~filter(:,1) & hour(sig(:,8)) >= 12; %PM

fig2 = figure;
scatter(sig(filter(:,1),2),sig(filter(:,1),6)-sig(filter(:,1),1),'o',...
        'DisplayName','Night') %Night
hold on
scatter(sig(filter(:,2),2),sig(filter(:,2),6)-sig(filter(:,2),1),'+',...
        'DisplayName','AM') %AM
scatter(sig(filter(:,3),2),sig(filter(:,3),6)-sig(filter(:,3),1),'d',...
        'DisplayName','PM') %PM
title(sprintf('Trajectory Data from %s to %s - %s',...
      dateRange{1},dateRange{2},tempSensor))
xlabel('Temperature Trajectory (m/°F)')
ylabel('Difference Between T_a and T_i (°F)')

%3D representation of the Ta-Ti vs. radiation vs. trajectory
fig3 = figure;
scatter3(sig(filter(:,1),6)-sig(filter(:,1),1),sig(filter(:,1),2),...
         sig(filter(:,1),5),'o') %Night
hold on
scatter3(sig(filter(:,2),6)-sig(filter(:,2),1),sig(filter(:,2),2),...
         sig(filter(:,2),5),'+') %AM
scatter3(sig(filter(:,3),6)-sig(filter(:,3),1),sig(filter(:,3),2),...
         sig(filter(:,3),5),'d') %PM
xlabel('T_a - T_i (°F)')
ylabel('Temperature Trajectory (s/°F)')
zlabel('Solar Radiation (W/m^2)')
legend('Night','AM','PM')

%Determining the last coefficents beta and del
%b = regress(y,X) <-Formatting example
x1 = sig(:,5); %radiation
x2 = sig(:,7); %ACstatus
x3 = sig(:,7).*(sig(:,6)-sig(:,1)).^2; %ACstatus/delT^2 interation
x4 = sig(:,7).*(sig(:,6)-sig(:,1)); %ACstatus/delT interation
x5 = sig(:,7).*sig(:,5); %ACstatus/radiation interation
y = sig(:,2) - coeff(1).*(sig(:,6)-sig(:,1)).^2 - ...
    coeff(2).*(sig(:,6)-sig(:,1)) - coeff(3);

X = [x1,x2,x3,x4,x5,ones(length(sig),1)];
[b,~,~,~,~] = regress(y, X);
```

```
%Combine the night-regressed coefficients with the day's coefficients
coeff(4:9) = b(1:6); %delta, epsilon, zeta, eda, lambda, and the error

%Review the regression
figure(fig2)
plot(coeff(1).*(sig(:,6)-sig(:,1)).^2+coeff(2).*(sig(:,6)-sig(:,1))+...
     coeff(3)+coeff(4).*sig(:,5)+coeff(5).*sig(:,7)+...
     coeff(6).*sig(:,7).*(sig(:,6)-sig(:,1)).^2+...
     coeff(7).*sig(:,7).*(sig(:,6)-sig(:,1))+...
     coeff(8).*sig(:,7).*sig(:,5)+coeff(9).*sig(:,7),sig(:,6)-...
     sig(:,1),'k*','DisplayName','Prediction')
legend(get(fig2, 'Child'),'show')

%Upload the regression coefficients to the database
%Write the signature matrix to the database
if ~isconnection(conn)
   conn = database(dbName, user , password, jdbcDriver, jdbcString);
end

colnames = {'device','regressionDay','samples','alpha',...
            'gamma','beta','delta'};
fastinsert(conn,'regression',colnames,...
           {tempSensor,date,length(sig),C(1),C(2),C(3),C(4)});

end
end

toc
```

# APPENDIX E

```
%This is the final program that is the culmination of this paper's efforts.
%It analyzes past interior temperatures, measured energy data, and weather
%data to to develop 'signatures' for each home studied. It then makes AC
%event duration predictions based on this 'signature'.

clc
clear all
warning('off','all');
tic
%% ---------[INITIALIZE THE DATABASE]--------------------------------------

%connection parameteres
host = '[REDACTED];  %MySQL hostname
user = '[REDACTED]';   %MySQL username
password = '[REDACTED]';    %MySQL password
dbName = '[REDACTED]'; %MySQL database name

%# JDBC parameters
jdbcString = sprintf('jdbc:mysql://%s/%s', host, dbName);
jdbcDriver = 'com.mysql.jdbc.Driver';

%# Create the database connection object
conn = database(dbName, user , password, jdbcDriver, jdbcString);

%% ---------[ESTABLISH THE RUNTIME SPECIFICS]------------------------------

date = '2011-07-21';

%CT Sizes: Temperature Sensor, WattNode, Mains-in, Mains-out, Fan, AC, PV
devices = {{'TEMP5' 'PWR4' 150 0 30 50 30};
           {'TEMP6' 'PWR4' 150 0 30 50 30};
           {'TEMP7' 'PWR7' 150 30 50 150 50};
           {'TEMP8' 'PWR7' 150 30 50 150 50};
           {'TEMP9' 'PWR6' 150 0 30 50 30};
           {'TEMP10' 'PWR6' 150 0 30 50 30};
           {'TEMP11' 'PWR8' 150 30 50 150 50};
           {'TEMP12' 'PWR8' 150 30 50 150 50};
           {'TEMP13' 'PWR3' 150 30 50 150 50};
           {'TEMP14' 'PWR3' 150 30 50 150 50};
           {'TEMP15' 'PWR2' 150 30 50 150 50};
           {'TEMP16' 'PWR2' 150 30 50 150 50};
           {'TEMP17' 'PWR5' 150 30 50 50 50};
           {'TEMP18' 'PWR5' 150 30 50 50 50}
          };
```

```
indices = [4 8 11 13]; %Range of devices to analyze for the date specified
period = 'PP'; %All day or Peak Period (PP)
results={}; %Initialize the final results cell
run = 'Y';

%% ---------[RUN THE COLLECTION OF SCRIPTS AS SPECIFIED]-------------------
while ~strcmp(run,'N')

dateRange = {datestr(date,'yyyy-mm-dd HH:MM:SS'),
             datestr(addtodate(datenum(date),1,'day'),...
             'yyyy-mm-dd HH:MM:SS')}; %date reformatted

%The first day of the regression period
date3Back = datestr(addtodate(datenum(dateRange{1}),-3,'day'),...
                    'yyyy-mm-dd HH:MM:SS');

for indexCount = 1:1:length(indices)
index = indices(indexCount);
tempSensor = devices{index}{1};
powerSensor = devices{index}{2};


%% ---------[GENERATE THE "COMPRESSED" EVENT-BASED DATA]-------------------
clear('weather','track','time','temperature','stdHigh','ssStart',...
      'ssEnd','sp','signatureRecords','sig','shStart','shEnd',...
      'setPointSched','setChanges','powerAC','power','meanHigh','m',...
      'ii','i','events','event','coeff','Puls');
%Check to see if the signature data has already been determined
%for the given day and given device
qry = sprintf(['SELECT count(id) ',...
               'FROM `signaturePwr` ',...
               'WHERE tempSensor = ''%s'' '...
               'AND date(periodStart) = ''%s'''],...
               tempSensor,date);
rs = fetch(exec(conn, qry));
signatureRecords = cell2mat(get(rs, 'Data'));

if signatureRecords ~= 0
%Delete records to ensure no duplicates are created
exec(conn, sprintf(['DELETE FROM `signaturePwr` ',...
                    'WHERE tempSensor = ''%s'' ',...
                    'AND date(periodStart) = ''%s'''],...
                    tempSensor,date));
end
```

```
%-------------[INTERIOR TEMPERATURE DATA]------------
qry = sprintf(['SELECT datetime, temp ',...
              'FROM `temperature` ',...
              'WHERE datetime BETWEEN ''%s'' and ''%s'' ',...
              'AND tempSensor = ''%s'''],...
              dateRange{1},dateRange{2},tempSensor);
rs = fetch(exec(conn, qry));
temperature = get(rs, 'Data');

%-------------[WEATHER DATA]------------
qry = sprintf(['SELECT datetime, globalHoriz, ambientTemp ',...
              'FROM `weatherAll` ',...
              'WHERE datetime BETWEEN ''%s'' and ''%s'''],...
              dateRange{1},dateRange{2});
rs = fetch(exec(conn, qry));
weather = get(rs, 'Data');

%-------------[POWER DATA]------------
%1: Main in, %2: Main out, %3: Fan circulation unit, %4: AC, %5: PV
qry = sprintf(['SELECT datetime, data1, data2, data3, data4, data5 ',...
              'FROM `power` ',...
              'WHERE datetime BETWEEN ''%s'' and ''%s'' ',...
              'AND powerSensor = ''%s'''],...
              dateRange{1},dateRange{2},powerSensor);
rs = fetch(exec(conn, qry));
power = get(rs, 'Data');

qry = sprintf(['SELECT DISTINCT data4, count(*) ',...
              'FROM `power` ',...
              'WHERE datetime BETWEEN ''%s'' and ''%s'' ',...
              'AND powerSensor = ''%s'' ',...
              'GROUP BY data4'],...
              dateRange{1},dateRange{2},powerSensor);
rs = fetch(exec(conn, qry));
powerAC = cell2mat(get(rs, 'Data'));
%plot(datenum(power(:,1)),cell2mat(power(:,5)));datetick
%most common pulse count (operational energy draw)
powerAC = powerAC(powerAC(:,2)==max(powerAC(length(powerAC)/2:end,2)),1);

%How well the power-derived ON/OFF corresponds to the measured temp.
clear title xlabel ylabel
fig3 = figure;
set(fig3,'PaperPositionMode','auto')
% set(fig3,'Position', [200 500 829 250]) %For plotting 2-column figures
% axes('Position',[.08 .15 .88 .75])
```

```
set(fig3,'Position', [200 500 829 168]) %For plotting 3-column figures
axes('Position',[.08 .23 .88 .62])
plot(datenum(cell2mat(temperature(:,1))),...
             cell2mat(temperature(:,2)),...
             'DisplayName','Temperature');
title(sprintf('From %s to %s - %s',dateRange{1},dateRange{2},tempSensor))
xlabel('Time of Day (HH:MM)')
ylabel('Interior Temperature (˚F)')
datetick; hold on; drawnow

%Create the signature data
i = 2; %index for all the temperature rows
ii = 1; %index for the signature rows
m = 1; %index for setpoint schedule
track = zeros(1,2);
sig = cell(1,12);

while i < length(temperature) && i < length(power)
if power{i,5} ~= 0 && power{i-1,5} == 0 %Identify when the pulses start

%--------[IDENTIFY AC ON CYCLE]-----------------
shStart = i; %starting shoulder index
ssStart = i+1; %starting steady-state index

%Shift the index up until the AC switches OFF
while power{i,5} ~= 0 && i < length(temperature) && i < length(power)
    i = i+1;
end

ssEnd = i-2; %ending steady-state index
shEnd = i-1; %ending shoulder index

%--------[DETERMINE AC EVENT TIMESTAMPS]----------------
if shEnd - shStart > 1 %Error check to make sure it's actually a cycle

%Number of pulses during the steady-state period of the cycle
Puls = sum(cell2mat(power(ssStart:ssEnd,5)));

%Seconds per pulse based on steady-state performance
sp = etime(datevec(power{shEnd,1}),datevec(power{ssStart,1}))/Puls;

%Create the "simplified" AC event transition record
%Event Start
event = addtodate(datenum(power{ssStart,1}),round(-sp*power{shStart,5}),...
                  'second');
```

```
if ii ~= 1
%Off Cycle End
sig{ii-1,2} = datestr(event,'yyyy-mm-dd HH:MM:SS');
end

%On Cycle Start
sig{ii,1} = datestr(event,'yyyy-mm-dd HH:MM:SS');
sig{ii,9} = logical(1); %AC status

%On Cycle End
event = addtodate(datenum(power{ssEnd,1}),round(sp*power{shEnd,5}),...
                  'second');
sig{ii,2} = datestr(event,'yyyy-mm-dd HH:MM:SS');

%Off Cycle Start
sig{ii+1,1} = datestr(event,'yyyy-mm-dd HH:MM:SS'); %event stop time
sig{ii+1,9} = logical(0); %AC status

ii=ii+2;
end

end
i = i+1;
end

%--------[SYNCHRONIZE DATA TO AC EVENT TIME DOMAIN]----------------
%Create the timeseries
temperature = timeseries(cell2mat(temperature(:,2)),temperature(:,1));
events = timeseries(cell2mat(sig(:,9)),sig(:,1));
weather = timeseries(cell2mat(weather(:,2:3)),weather(:,1));
weather.data(weather.data(:,1) < 0,1) = 0; %Correct negative rad values

%Name the timeseries
temperature.name = 'temperature';
events.name = 'events';
weather.name = 'weather';

%Synchronize the time vectors to the AC events
temperature = resample(temperature,getabstime(events));
weather = resample(weather,getabstime(events));

%Visualize the accuracy of the power-derived AC event transitions
time = datenum(getabstime(temperature));
plot(time(logical(events.data)),...
     temperature.data(logical(events.data)),...
     'LineStyle','none',...
     'Marker','x',...
     'MarkerSize',10,...
     'MarkerEdgeColor','red',...
     'DisplayName','AC Turn On')
plot(time(~logical(events.data)),...
     temperature.data(~logical(events.data)),...
     'LineStyle','none',...
     'Marker','+',...
     'MarkerSize',10,...
     'MarkerEdgeColor','black',...
     'DisplayName','AC Turn Off')

%--------[CHARACTERIZE THE AC EVENTS]----------------
sig(1:end-1,3) = num2cell(etime(datevec(sig(1:end-1,2)),...
                               datevec(sig(1:end-1,1))));%duration (sec)
sig(:,4) = num2cell(temperature.data); %starting interior temp.
sig(1:end-1,5) = num2cell(cell2mat(sig(2:end,4)) - ...
                         cell2mat(sig(1:end-1,4))); %change in temp.
sig(1:end-1,6) = num2cell(cell2mat(sig(:,3))./cell2mat(sig(:,5))); %s/deg
sig(:,7) = num2cell(weather.data(:,2)); %starting exterior temp.
sig(:,8) = num2cell(weather.data(:,1)); %starting radiation
sig(:,10) = {tempSensor};
sig = sig(1:end-1,:); %The last record isn't complete, so drop it

%T_high and T_low temperatures during night and peak periods
sig(:,11) = {logical(0)};
sig(:,12) = sig(:,6);
filter = cell2mat(sig(:,9))==1 & ...
        (datenum(datestr(sig(:,1),'HH:MM:SS')) >= datenum('13:30:00') & ...
         datenum(datestr(sig(:,1),'HH:MM:SS')) < datenum('19:00:00'));
T_high = mean(cell2mat(sig(filter,4)));
sig(filter,11) = {logical(1)};

filter = cell2mat(sig(:,9))==0 & ...
        (datenum(datestr(sig(:,1),'HH:MM:SS')) >= datenum('13:30:00') & ...
         datenum(datestr(sig(:,1),'HH:MM:SS')) < datenum('19:00:00'));
T_low = mean(cell2mat(sig(filter,4)));
sig(filter,11) = {logical(1)};

filter = cell2mat(sig(:,11)) == 1 & cell2mat(sig(:,9)) == 1; %PP AC-on
sig(filter,12) = num2cell(cell2mat(sig(filter,3))./(T_low-T_high));

filter = cell2mat(sig(:,11)) == 1 & cell2mat(sig(:,9)) == 0; %PP AC-off
```

```
sig(filter,12) = num2cell(cell2mat(sig(filter,3))./(T_high-T_low));

%Include a box that indicates the peak period and temperature bounds
x{1} = datenum(datevec(date)+[0 0 0 13 30 0]);
x{2} = datenum(datevec(date)+[0 0 0 19 00 0]);
plot([x{1} x{2}],[T_high T_high],'--r',...
                                 'LineWidth',2,...
                                 'DisplayName','Peak Period T_h_i_g_h')
plot([x{1} x{2}],[T_low T_low],'--k',...
                               'LineWidth',2,...
                               'DisplayName','Peak Period T_l_o_w')

legend(get(fig3, 'Child'),'show')
legend('location','southwest')
drawnow
print(fig3,'-dpng','-r100',...
      sprintf('[REDACTED]\\LinearEst\\%s-%s-(LIN).png',tempSensor,date));

%Upload the given day to the database
%For certain days when the measured temperature doesn't change between
%AC on/off events, the trajectory kicks back strange values. Clear out
%these rows before updating the database.
sig = sig(abs(cell2mat(sig(:,6))) < 999999,:); %Infinite of large slopes
colnames = {'tempSensor','periodStart','periodEnd','duration',...
            'startTempIn','endTempIn',...
            'invSlope','startRad','startTempAmb','mode',...
            'isPeak','impSlope'};
fastinsert(conn,'signaturePwr',colnames,...
           [sig(:,10), sig(:,1), sig(:,2), sig(:,3), sig(:,4),...
            num2cell(cell2mat(sig(:,4))+cell2mat(sig(:,5))),...
            sig(:,6), sig(:,8), sig(:,7), sig(:,9),...
            sig(:,11), sig(:,12)]);

%Create the duration vs time plot
clear title xlabel ylabel
fig4 = figure;
set(gcf,'PaperPositionMode','auto')
set(fig4, 'Position', [400 200 440 230])
scatter(datenum(sig(cell2mat(sig(:,9))==1,1)),...
        cell2mat(sig(cell2mat(sig(:,9))==1,3)),'bo','DisplayName','AC On')
hold on
scatter(datenum(sig(cell2mat(sig(:,9))==0,1)),...
        cell2mat(sig(cell2mat(sig(:,9))==0,3)),'rd','DisplayName','AC Off')
datetick
title(sprintf('AC Cycle Durations for %s - %s',date,tempSensor))
```

```
ylabel('AC Cycle Durations (s)')
xlabel('Time of Day (HH:MM)')
legend(get(fig4, 'Child'),'show')

drawnow
print(fig4,'-dpng','-r100',...
      sprintf('[REDACTED]\\Durations\\%s-%s-(DUR).png',tempSensor,date));


%% ---------[RESOLVE THE MODEL PARAMETERS FOR THE CURRENT DAY]-------------
clear('Puls','colnames','event','events','i','ii','m','meanHigh',...
      'power','powerAC','qry','setChanges','setPointSched','shEnd',...
      'shStart','sig','signatureRecords','sp','ssEnd','ssStart',...
      'stdHigh','temperature','time','track','weather');
%See if the sensor has been signature'd the 3 previous days, and if the
%current day has had its regression coefficients calculated yet
qry = sprintf(['SELECT date(periodStart) AS `date`, '...
                      'count(*) AS `count` ',...
              'FROM `signaturePwr` ',...
              'WHERE tempSensor = ''%s'' ',...
              'AND periodStart BETWEEN ''%s'' AND ''%s'' ',...
              'GROUP BY date(periodStart)'],...
              tempSensor,date3Back,dateRange{1});
rs = fetch(exec(conn, qry));
signatureRecords = get(rs, 'Data');

qry = sprintf(['SELECT alpha, beta, gamma, delta, epsilon, zeta, eda, ',...
                      'lambda, error, trainingPeriod ',...
              'FROM `regressionpolyPwr` ',...
              'WHERE tempSensor = ''%s'' ',...
              'AND trainingPeriod = ''%s'' ',...
              'AND date = ''%s'''],...
              tempSensor, period, date);
rs = fetch(exec(conn, qry));
regressionRecord = get(rs, 'Data');

if size(signatureRecords,1) == 3

%Query in the AC event records
if strcmp(period,'PP')
qry = sprintf(['SELECT startTempIn, impSlope, duration, '...
                      '(startTempIn-endTempIn), startRad, ',...
                      'startTempAmb, mode, periodStart ',...
               'FROM `signaturePwr` ',...
               'WHERE tempSensor = ''%s'' ',...
```

```
                'AND periodStart BETWEEN ''%s'' AND ''%s'' ',...
                'AND (isPeak = 1 OR startRad < 5) ',...
                'ORDER BY periodStart'],...
                tempSensor,date3Back,dateRange{1});
else %AD
qry = sprintf(['SELECT startTempIn, invSlope, duration, '...
                       '(startTempIn-endTempIn), startRad, ',...
                       'startTempAmb, mode, periodStart ',...
                'FROM `signaturePwr` ',...
                'WHERE tempSensor = ''%s'' ',...
                'AND periodStart BETWEEN ''%s'' AND ''%s'' ',...
                'ORDER BY periodStart'],...
                tempSensor,date3Back,dateRange{1});
end

rs = fetch(exec(conn, qry));
sig = get(rs, 'Data');
sig = [cell2mat(sig(:,1:6)),cell2mat(sig(:,7)),...
                             datenum(sig(:,8))]; %Mode comes in as a logical

%Create delta T column, then remove the rows where del T < 5
sig(:,9) = sig(:,6)-sig(:,1);
sig=sig(sig(:,end)>5,:);

%Remove the samples with daylight and when the AC is running
sigNight = sig(~logical(sig(:,7)),:); %When the AC is off
sigNight = sigNight(~(sigNight(:,5) > 5),:); %When the GHI is over 5

%------------[NIGHT REGRESSION]------------
%There are some slopes that really throw off the data. Remove the rows that
%are more than 1.5 standard deviations from the median
dev = 1.5; %Standard Deviation level

outliers = abs(sigNight(:,2)-median(sigNight(:,2)))>dev*std(sigNight(:,2));
sigNight = sigNight(~outliers,:);

%[b,bint,r,rint,stats] = regress(y,X)
X = [(sigNight(:,6)-sigNight(:,1)).^2,...
      sigNight(:,6) - sigNight(:,1),ones(size(sigNight,1),1)];
y = sigNight(:,2); %inverse slope
[b,~,~,~,~] = regress(y,X);

%The WARM UP trajectories vs the actual temp sensor readings for the NIGHT
fig1 = figure;
set(gcf,'PaperPositionMode','auto')
set(fig1, 'Position', [100 100 440 230])
scatter(sigNight(:,6)-sigNight(:,1),sigNight(:,2))
hold on
scatter(sigNight(:,6)-sigNight(:,1),b(1)*(sigNight(:,6) - ...
        sigNight(:,1)).^2 + b(2)*(sigNight(:,6)-sigNight(:,1))+b(3),'r*')
title(sprintf('Night Warming from %s to %s - %s',...
      date3Back,dateRange{1},tempSensor))
ylabel('Temperature Trajectory (m/°F)')
xlabel('Difference in Temperature Between T_a and T_i (°F)')
legend('Data','Predictions')
drawnow
print(fig1,'-dpng','-r100',...
      sprintf('[REDACTED]\\NightWarming\\%s-%s-%s-(NW).png',...
      tempSensor,period,date));

%Store the results of the first regression test
coeff = b; %alpha, beta, gamma

%------------[FULL DAY REGRESSION]------------
%Query that selects given time periods through the day
%Same thing, remove the outliers, but remove them seperately for ON and OFF
%AC operation modes
statusOn = sig(logical(sig(:,7)),:);
statusOff = sig(~logical(sig(:,7)),:);

outlierOn = abs(statusOn(:,2) - median(statusOn(:,2))) > ...
                                dev*std(statusOn(:,2));
outlierOff = abs(statusOff(:,2) - median(statusOff(:,2))) > ...
                                  dev*std(statusOff(:,2));

statusOn = statusOn(~outlierOn,:);
statusOff = statusOff(~outlierOff,:);

%Illustrates the trajectory groupings for the on/off AC states
sig = cat(1,statusOn,statusOff); %Combines the two matricies
filter = logical([]);
filter(:,1) = sig(:,5) < 5; %No radiation (night)
filter(:,2) = ~filter(:,1) & hour(sig(:,8)) < 12; %AM
filter(:,3) = ~filter(:,1) & hour(sig(:,8)) >= 12; %PM

fig2 = figure;
set(gcf,'PaperPositionMode','auto')
set(fig2, 'Position', [1100 100 525 230])
scatter(sig(filter(:,1),2),sig(filter(:,1),6)-sig(filter(:,1),1),'o',...
        'DisplayName','Night')%Night
```

```
hold on
scatter(sig(filter(:,2),2),sig(filter(:,2),6)-sig(filter(:,2),1),'+',...
        'DisplayName','AM') %AM
scatter(sig(filter(:,3),2),sig(filter(:,3),6)-sig(filter(:,3),1),'d',...
        'DisplayName','PM') %PM
title(sprintf('Trajectory Data from %s to %s - %s',...
      date3Back,dateRange{1},tempSensor))
xlabel('Temperature Trajectory (s/°F)')
ylabel('Difference Between T_a and T_i (°F)')
drawnow

% %3D representation of the Ta-Ti vs. radiation vs. trajectory
% fig3 = figure;
% scatter3(sig(filter(:,1),6)-sig(filter(:,1),1),...
%          sig(filter(:,1),2),sig(filter(:,1),5),'o')%Night
% hold on
% scatter3(sig(filter(:,2),6)-sig(filter(:,2),1),...
%          sig(filter(:,2),2),sig(filter(:,2),5),'+') %AM
% scatter3(sig(filter(:,3),6)-sig(filter(:,3),1),...
%          sig(filter(:,3),2),sig(filter(:,3),5),'d') %PM
% xlabel('T_a - T_i (°F)')
% ylabel('Temperature Trajectory (s/°F)')
% zlabel('Solar Radiation (W/m^2)')
% legend('Night','AM','PM')

%Determining the last coefficents beta and del
%b = regress(y,X) <-Formatting example
x1 = sig(:,5); %radiation
x2 = sig(:,7); %ACstatus
x3 = sig(:,7).*sig(:,9).^2; %ACstatus & delT^2 (interation)
x4 = sig(:,7).*sig(:,9); %ACstatus & delT (interation)
x5 = sig(:,7).*sig(:,5); %ACstatus & radiation (interation)
y = sig(:,2) - coeff(1).*sig(:,9).^2 - coeff(2).*sig(:,9) - coeff(3);

X = [x1,x2,x3,x4,x5,ones(length(sig),1)];
[b,~,~,~,~] = regress(y, X);

%Combine the night-regressed coefficients with the day's coefficients
coeff(4:9) = b; %delta, epsilon, zeta, eda, lambda, and the error

%Review the regression
figure(fig2)
plot(coeff(1).*sig(:,9).^2+coeff(2).*sig(:,9) + coeff(3) + ...
     coeff(4).*sig(:,5) + coeff(5).*sig(:,7) + ...
     coeff(6).*sig(:,7).*sig(:,9).^2 + coeff(7).*sig(:,7).*sig(:,9) + ...
     coeff(8).*sig(:,7).*sig(:,5) + coeff(9),sig(:,9),...
     'k*','DisplayName','Prediction')
legend(get(fig2, 'Child'),'show')
drawnow
print(fig2,'-dpng','-r100',...
      sprintf('[REDACTED]\\Trajectories\\%s-%s-%s-(TRAJ).png',...
      tempSensor,period,date));

% figure(fig3)
% scatter3(sig(:,9),coeff(1).*sig(:,9).^2 + coeff(2).*sig(:,9) + ...
%          coeff(3)+coeff(4).*sig(:,5)+ coeff(5).*sig(:,7) + ...
%          coeff(6).*sig(:,7).*sig(:,9).^2 + ...
%          coeff(7).*sig(:,7).*sig(:,9) + ...
%          coeff(8).*sig(:,7).*sig(:,5),sig(:,5),'k*') %PM
% legend('Night','AM','PM','Prediction')

%------------[UPDATE THE DATABASE]------------
colnames = {'tempSensor','date','samples','alpha','beta','gamma',...
            'delta','epsilon','zeta','eda','lambda','error',...
            'trainingPeriod'};

if ~strcmp(regressionRecord{1},'No Data') && ...
    sum(strcmp(period,regressionRecord(:,10))) == 1

    %Update the existing record
    update(conn,'regressionpolyPwr',colnames,...
          {tempSensor,date,length(sig),coeff(1),coeff(2),coeff(3),...
          coeff(4),coeff(5),coeff(6),coeff(7),coeff(8),coeff(9),period},...
          sprintf(['where tempSensor = ''%s'' AND date = ''%s'' ',...
                   'AND trainingPeriod = ''%s'''],tempSensor,date,period));
else
    %Write the new data to the database
    fastinsert(conn,'regressionpolyPwr',colnames,...
            {tempSensor,date,length(sig),coeff(1),coeff(2),coeff(3),...
            coeff(4),coeff(5),coeff(6),coeff(7),coeff(8),coeff(9),period});
end

end


%% ---------[USE THE MODEL PARAMETERS TO MAKE TIME PREDICTIONS]--------
clear('X','b','coeff','colnames','dev','filter','outlierOff',...
      'outlierOn','outliers','qry','regressionRecord','sig','sigNight',...
      'statusOff','statusOn','tempResult','x1','x2','x3','x4','x5','y');
```

```
%-------------[MODEL PARAMETERS]-------------------
qry = sprintf(['SELECT alpha, beta, gamma, delta, epsilon, '...
                      'zeta, eda, lambda, error ',...
               'FROM `regressionpolyPwr` ',...
               'WHERE tempSensor = ''%s'' ',...
               'AND date = ''%s'' ',...
               'AND trainingPeriod = ''%s'''],...
             tempSensor,date,period);
rs = fetch(exec(conn, qry));
coeff = cell2mat(get(rs, 'Data'));

if ~strcmp(coeff,'No Data')

%-------------[COMPRESSED TEMPERATURE DATA]---------
qry = sprintf(['SELECT periodStart, periodEnd, duration, '...
                      'startTempIn, endTempIn, ',...
                      '(startTempAmb-startTempIn), invSlope, '...
                      'startRad, startTempAmb, mode ',...
               'FROM `signaturePwr` ',...
               'WHERE periodStart BETWEEN ''%s'' and ''%s'' ',...
               'AND tempSensor = ''%s'' ',...
               'ORDER BY periodStart ASC'],...
               dateRange{1},dateRange{2},tempSensor);
rs = fetch(exec(conn, qry));
sig = get(rs, 'Data');

%-------------[INTERIOR TEMPERATURE DATA]------------
qry = sprintf(['SELECT datetime, temp ',...
              'FROM `temperature` ',...
              'WHERE datetime BETWEEN ''%s'' and ''%s'' ',...
              'AND tempSensor = ''%s'''],...
              dateRange{1},dateRange{2},tempSensor);
rs = fetch(exec(conn, qry));
temperature = get(rs, 'Data');

fig5 = figure;
set(gcf,'PaperPositionMode','auto')
set(fig5, 'Position', [600 200 795 278])
axes('Position',[.08 .13 .89 .77])
p{1} = plot(datenum(temperature(:,1)),cell2mat(temperature(:,2)));
hold on
p{2} = plot([datenum(sig(:,1)),datenum(sig(:,2))],[cell2mat(sig(:,4)),...
             cell2mat(sig(:,5))],'r','LineWidth',2);

for i=1:1:length(sig)
```

```
%Use model parameters to calculate the slope at time t
slope = coeff(1)*sig{i,6}^2 + coeff(2)*sig{i,6} + coeff(3) + ...
        coeff(4)*sig{i,8} + coeff(5)*sig{i,10} + ...
        coeff(6)*sig{i,10}*sig{i,6}^2 + coeff(7)*sig{i,10}*sig{i,6} + ...
        coeff(8)*sig{i,10}*sig{i,8};

%Use the slope with T(t*)-T(t) to calculate the event duration (seconds)
duration = (sig{i,5}-sig{i,4})*slope;

%Tabulate the prediction results
tempResult{i,1} = sig{i,3}; %actual duration
tempResult{i,2} = duration; %predicted duration
tempResult{i,3} = sig{i,10}; %AC mode

if datenum(datestr(sig{i,1},'HH:MM:SS')) >= datenum('13:30:00') && ...
   datenum(datestr(sig{i,1},'HH:MM:SS')) < datenum('19:00:00')
tempResult{i,4} = logical(1); %It's a peak period prediciton
else
tempResult{i,4} = logical(0);
end

%Plot the trajectory
p{3} = plot([datenum(sig{i,1}) addtodate(datenum(sig{i,1}),...
            duration,'second')],...
     [sig{i,4} sig{i,5}],'k','LineWidth',2);
end

datetick
title(sprintf('AC Event Duration Predictions for %s - %s',date,tempSensor))
ylabel('Temperature (°F)')
xlabel('Time of Day (HH:MM)')
legend([p{1} p{2}(1) p{3}],{'Measured Temp.',...
                            'Compressed Trajectory',...
                            'Predicted Trajectory'})
drawnow
print(fig5,'-dpng','-r100',…
      sprintf('[REDACTED]\\Results-AD\\%s-%s-%s-(RES).png',...
      tempSensor,period,date));

%Transfer the temp results prediction results to the permanent table
j = size(results,1) + 1;
results{j,1} = tempSensor;
results{j,2} = date;

%ON durations
```

```
%Actual AD
results{j,3} = mean(cell2mat(tempResult(cell2mat(tempResult(:,3))==1,1)));

%Predicted AD
results{j,4} = mean(cell2mat(tempResult(cell2mat(tempResult(:,3))==1,2)));
results{j,5} = (results{j,4}-results{j,3})/results{j,3}; %error of above

%Actual PP
results{j,6} = mean(cell2mat(tempResult((cell2mat(tempResult(:,3))==1) &...
                                       (cell2mat(tempResult(:,4))==1),1)));
%Predicted PP
results{j,7} = mean(cell2mat(tempResult((cell2mat(tempResult(:,3))==1) &...
                                       (cell2mat(tempResult(:,4))==1),2)));
results{j,8} = (results{j,7}-results{j,6})/results{j,6}; %error of above

%OFF durations
%Actual AD
results{j,9} = mean(cell2mat(tempResult(cell2mat(tempResult(:,3))==0,1)));

%Predicted AD
results{j,10} = mean(cell2mat(tempResult(cell2mat(tempResult(:,3))==0,2)));
results{j,11} = (results{j,10}-results{j,9})/results{j,9}; %error of above

%Actual PP
results{j,12} = mean(cell2mat(tempResult((cell2mat(tempResult(:,3))==0)&...
                                       (cell2mat(tempResult(:,4))==1),1)));

%Predicted PP
results{j,13} = mean(cell2mat(tempResult((cell2mat(tempResult(:,3))==0)&...
                                       (cell2mat(tempResult(:,4))==1),2)));
results{j,14} = (results{j,13}-results{j,12})/results{j,12};%error of above

end

end

%Option the user to continure running scripts for the next day
date = datestr(addtodate(datenum(date),1,'day'),'yyyy-mm-dd');
run = input(sprintf('Run scripts for %s? ([Y]/N): ',date),'s');

%Close the plot windows
close all

end
toc
```

# Bibliography

[1] National Climatic Center (U.S.) and United States National Oceanic and Atmospheric Administration, “Comparative Climatic Data For the United States Through 2012,” 2012. [Online]. Available: http://www1.ncdc.noaa.gov/pub/data/ccd-data/CCD-2012.pdf.

[2] U.S. Energy Information Administration, “Household energy use in Arizona,” *2009 Residential Energy Consumption Survey*, 2009. [Online]. Available: http://www.eia.gov/consumption/residential/reports/2009/state_briefs/pdf/az.pdf.

[3] S. B. Sadineni and R. F. Boehm, “Measurements and simulations for peak electrical load reduction in cooling dominated climate,” *Energy*, vol. 37, no. 1, pp. 689–697, Jan. 2012.

[4] U.S. Energy Information Administration, “Table 8.6.B noncoincident peak load by North American Electric Reliability Corporation assessment area,” 2012. [Online]. Available: http://www.eia.gov/electricity/annual/html/epa_08_06_b.html.

[5] U.S. Energy Information Administration, “Table 8.6.A noncoincident peak load by North American Electric Reliability Corporation assessment area,” 2012. [Online]. Available: http://www.eia.gov/electricity/annual/html/epa_08_06_a.html.

[6] “Annual smart grid spending to reach $65 billion by 2017,” *Transmission & Distribution World*, no. February, Overland Park, p. 1, May-2014.

[7] J. Casazza and F. Delea, *Understanding Electric Power; An Overview of the Technology and the Marketplace*. Hoboken, New Jersey: John Wiley & Sons, 2003, p. 47.

[8] S. Carley, “Energy demand-side management: new perspectives for a new era,” *Policy Anal. Manag.*, vol. 31, no. 1, pp. 6–32, 2012.

[9] Federal Energy Regulatory Commission, “Assessment of demand response and advanced metering,” 2013. [Online]. Available: http://www.ferc.gov/legal/staff-reports/2013/oct-demand-response.pdf.

[10] Federal Energy Regulatory Commission, “National assessment of demand response potential,” 2009. [Online]. Available: http://www.ferc.gov/legal/staff-reports/06-09-demand-response.pdf.

[11] Federal Energy Regulatory Commission and U.S. Department of Energy, “Implementation proposal for the national action plan on demand response,” 2011. [Online]. Available: http://www.ferc.gov/legal/staff-reports/07-11-dr-action-plan.pdf.

[12] H. Peitsman and V. Bakker, “Application of black-box models to HVAC systems for fault detection,” in *ASHRAE Transactions*, 1996, pp. 102(1):628–40.

[13] Q. T. Ahmad, “Review paper: validation of building thermal and energy models,” *Build. Serv. Eng. Res. Technol.*, vol. 19, no. 2, pp. 61–66, Jan. 1998.

[14] P. Almeida, M. J. Carvalho, R. Amorim, J. F. Mendes, and V. Lopes, “Dynamic testing of systems – use of TRNSYS as an approach for parameter identification,” *Sol. Energy*, vol. 104, pp. 60–70, Mar. 2014.

[15] M. Avci, M. Erkoc, A. Rahmani, and S. Asfour, “Model predictive HVAC load control in buildings using real-time electricity pricing,” *Energy Build.*, vol. 60, pp. 199–209, May 2013.

[16] M. Siemann, J. Kim, D. Oberholzer, and C. Sloop, “Performance of a residential building energy grey-box model using localized weather networks,” *ASHRAE Trans.*, vol. 119(1), pp. 1–8, 2013.

[17] Y. Ma, “Model predictive control for energy efficient buildings,” Ph.D. dissertation, Control and Design, University of California, Berkeley, CA, 2012.

[18] A. Rabl and L. K. Norford, “Peak load reduction by preconditioning buildings at night,” *Int. J. Energy Res.*, vol. 15, no. 9, pp. 781–798, 1991.

[19] H. Zhao and F. Magoulès, “A review on the prediction of building energy consumption,” *Renew. Sustain. Energy Rev.*, vol. 16, no. 6, pp. 3586–3592, Aug. 2012.

[20] American Society of Heating Refrigerating and Air-Conditioning Engineers, *2009 ASHRAE Handbook: Fundamentals*. Atlanta, GA: American Society of Heating, Refrigerating and Air-Conditioning Engineers, 2009.

[21] D. B. Crawley, J. W. Hand, M. Kummert, and B. T. Griffith, “Contrasting the capabilities of building energy performance simulation programs,” *Build. Environ.*, vol. 43, no. 4, pp. 661–673, Apr. 2008.

[22] P. Li, Z. O’Neill, and J. E. Braun, “Development of control-oriented models,” *ASHRAE Trans.*, vol. 119, no. 2, pp. 1–8, 2013.

[23] A. A. T. Maia, R. N. N. Koury, and L. Machado, “Development of a control algorithm employing data generated by a white box mathematical model,” *Appl. Therm. Eng.*, vol. 54, no. 1, pp. 120–130, May 2013.

[24] D. C. Park, M. a. El-Sharkawi, R. J. Marks, L. E. Atlas, and M. J. Damborg, “Electric load forecasting using an artificial neural network,” *IEEE Trans. Power Syst.*, vol. 6, no. 2, pp. 442–449, May 1991.

[25] M. M. Shapiro, A. J. Yager, and T. H. Ngan, “Test hut validation of a microcomputer predictive HVAC control,” *ASHRAE Trans.*, vol. 94, no. 1, pp. 644–663, 1988.

[26] Y. Ma, G. Anderson, and F. Borrelli, “A distributed predictive control approach to building temperature regulation,” in *2011 American Control Conference*, 2011, pp. 2089–2094.

[27] I.-H. Yang, M.-S. Yeo, and K.-W. Kim, “Application of artificial neural network to predict the optimal start time for heating system in building,” *Energy Convers. Manag.*, vol. 44, no. 17, pp. 2791–2809, Oct. 2003.

[28] M. Aydinalp, V. Ismet Ugursal, and A. S. Fung, “Modeling of the appliance, lighting, and space-cooling energy consumptions in the residential sector using neural networks,” *Appl. Energy*, vol. 71, no. 2, pp. 87–110, Feb. 2002.

[29] A. Khotanzad, R. Afkhami-Rohani, and D. Maratukulam, “ANNSTLF - artificial neural network short-term load forecaster - generation three,” *IEEE Trans. Power*, vol. 13, no. 4, 1998.

[30] G. Mustafaraj, J. Chen, and G. Lowry, “Thermal behaviour model identification for an office space using BMS data,” *Build. Serv. Eng. Res. Technol.*, vol. 30, no. 4, pp. 329–341, 2009.

[31] N. Chaturvedi and J. E. Braun, “Analytical tools for dynamic building control,” ASHRAE, Rep. #5031-1, 2000.

[32] B. Thomas and M. Soleimani-Mohseni, “Artificial neural network models for indoor temperature prediction: investigations in two buildings,” *Neural Comput. Appl.*, vol. 16, no. 1, pp. 81–89, Mar. 2006.

[33] W. Yu, B. Li, Y. Lei, and M. Liu, “Analysis of a Residential Building Energy Consumption Demand Model,” *Energies*, vol. 4, no. 12, pp. 475–487, Mar. 2011.

[34] J. W. Moon, “ANN-based model-free thermal controls for residential buildings,” Ph.D. dissertation, Architecture, The University of Michigan, Ann Arbor, MI, 2009.

[35] M. Norgaard, O. Ravn, N. K. Poulsen, and L. K. Hansen, *Neural Networks for Modelling and Control of Dynamic Systems*. Springer-Verlag, 2000.

[36] J. S. Haberl and S. Thamilseran, “The great energy predictor shootout II - measuring retrofit savings,” *ASHRAE J.*, vol. 40, no. 1, pp. 49–56, 1998.

[37] H. S. Hippert, C. E. Pedreira, and R. C. Souza, “Neural networks for short-term load forecasting: a review and evaluation,” *IEEE Trans. Power Syst.*, vol. 16, no. 1, pp. 44–55, 2001.

[38] I.-H. Yang and K.-W. Kim, “Prediction of the time of room air temperature descending for heating systems in buildings,” *Build. Environ.*, vol. 39, no. 1, pp. 19–29, Jan. 2004.

[39] A. E. Ben-Nakhi and M. a Mahmoud, “Energy conservation in buildings through efficient A/C control using neural networks,” *Appl. Energy*, vol. 73, no. 1, pp. 5–23, Sep. 2002.

[40] A. E. Ruano, E. M. Crispim, E. Z. E. Conceição, and M. M. J. R. Lúcio, “Prediction of building’s temperature using neural networks models,” *Energy Build.*, vol. 38, no. 6, pp. 682–694, Jun. 2006.

[41] H. Madsen and J. Holst, “Estimation of continuous-time models for the heat dynamics of a building,” *Energy Build.*, vol. 22, pp. 67–79, 1995.

[42] F. Deque, F. Ollivier, and A. Poblador, “Grey boxes used to represent buildings with a minimum number of geometric and thermal parameters,” *Energy Build.*, vol. 31, no. 1, pp. 29–35, 2000.

[43] Y. Oussar and G. Dreyfus, “How to be a gray box: dynamic semi-physical modeling.,” *Neural Netw.*, vol. 14, no. 9, pp. 1161–72, Nov. 2001.

[44] S. Wu and J.-Q. Sun, “A physics-based linear parametric model of room temperature in office buildings,” *Build. Environ.*, vol. 50, pp. 1–9, Apr. 2012.

[45] K.-H. Lee and J. E. Braun, “Development of methods for determining demand-limiting setpoint trajectories in buildings using short-term measurements,” *Build. Environ.*, vol. 43, no. 10, pp. 1755–1768, Oct. 2008.

[46] K. Lee and J. E. Braun, “Evaluation of methods for determining demand-limiting setpoint trajectories in buildings using short-term measurements,” *Build. Environ.*, vol. 43, no. 10, pp. 1769–1783, Oct. 2008.

[47] U.S. Energy Information Administration, “Air conditioning in nearly 100 million U.S. homes,” *Residential Energy Consumption Survey (RECS)*, 2009. [Online]. Available: http://www.eia.gov/consumption/residential/reports/2009/air-conditioning.cfm.

[48] J. Dieckmann and J. Brodrick, “VFDs for residential systems,” *ASHRAE J.*, vol. 52, no. 8, pp. 66–68, 2010.

[49] B. Scaringe, “Thermostatic wiring principles,” *Mainstream Engineering*, 2011. [Online]. Available: http://www.epatest.com/store/resources/images/misc/how-a-thermostat-operates.pdf.

[50] National Electrical Manufacturers Association, *NEMA DC 3-2013 Residential Controls – Electrical Wall-Mounted Room Thermostats*. 2013.

[51] T. E. Peffer, “California DREAMing: the design of residential demand responsive technology with people in mind,” Ph.D. dissertation, Architecture, University of California, Berkeley, CA, 2009.

[52] J. H. Jang, “System design and house dynamic signature identification for intelligent energy management in residential buildings,” Ph.D. dissertation, Mechanical Engineering, University of California, Berkeley, CA, 2008.

[53] J. E. Braun, K. W. Montgomery, and N. Chaturvedi, “Evaluating the performance of building thermal mass control strategies,” *HVAC&R Res.*, vol. 7, no. 4, pp. 403–428, 2001.

[54] M. A. Piette, D. Watson, N. Motegi, S. Kiliccote, and P. Xu, “Automated critical peak pricing field tests: program description and results,” Lawrence Berkeley National Laboratory, Rep. 59351, 2006.

[55] S. M. R. Sanhueza, F. L. Tofoli, F. L. de Albuquerque, J. C. de Oliveira, and G. C. Guimaraes, “Analysis and Evaluation of Residential Air Conditioners for Power System Studies,” *IEEE Trans. Power Syst.*, vol. 22, no. 2, pp. 706–716, May 2007.

[56] A. Rabl, “Parameter estimation in buildings: methods for dynamic analysis of measured energy use,” *J. Sol. Energy Eng.*, vol. 110, pp. 52–66, 1988.

[57] C. Younes, C. a. Shdid, and G. Bitsuamlak, “Air infiltration through building envelopes: A review,” *J. Build. Phys.*, vol. 35, no. 3, pp. 267–302, Oct. 2011.

[58] NV Energy, “Nevada dynamic pricing trial: residential single-family service critical peak price,” 2011. [Online]. Available: https://www.nvenergy.com/company/rates/snv/schedules/images/RS_CPP_South.pdf.

[59] M. Sinha, “Intelligent demand side management of residential building energy systems,” Ph.D. dissertation, Mechanical Engineering, The University of North Carolina at Charlotte, Charlotte, NC, 2013.

[60] M. Avci, “Demand response-enabled model predictive HVAC load control in buildings using real-time electricity pricing,” Ph.D. dissertation, Industrial Engineering, University of Miami, Miami, FL, 2013.

## Vita

Graduate College
University of Nevada, Las Vegas

Andrew G. Cross

Degrees:

Bachelor of Science, Mechanical Engineering, 2009
University of Kansas

Publications:

A. Cross, K. Hammer, R. Hurt, R. Boehm, "An Autonomous Controller for Ductless Mini-Split Heat Pumps, Residential Solar Thermal Collection, and Hydronic Floor Heating" ES-FuelCell2014-6316, ASME 2014 8th International Conference on Energy Sustainability & 12th Fuel Cell Science, Engineering and Technology Conference, June 29-July 2, 2014, Boston, Massachusetts.

Experience and Certifications:

Vertical Turbine Engineer – Pentair Engineered Flow Kansas City Operations, 2010-2012
EIT recognized by the Kansas State Board of Technical Professions, 2009

Thesis Title:

Development of a Black-Box Transient Thermal Model for Residential Buildings

Thesis Examination Committee:

Chairperson, Dr. Robert Boehm
Committee Member, Dr. Hsuan-Tsung Hsieh
Committee Member, Dr. Joon Soo Lee
Graduate College Representative, Dr. Wolfgang Bein