# Autoregressive Linear Thermal Model of a Residential Forced-Air Heating System with Backpropagation Parameter Estimation Algorithm

*Eric Burger*
*Scott Moura*
*David E. Culler*

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2017-28
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2017/EECS-2017-28.html

May 5, 2017

**Autoregressive Linear Thermal Model of a Residential Forced-Air Heating System with Backpropagation Parameter Estimation Algorithm**

by Eric Burger

## Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

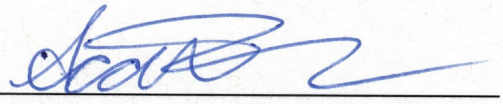
**Committee:**

Professor David E. Culler
Research Advisor

5/5/17

(Date)

* * * * * * *



Professor Scott J. Moura
Second Reader

5/5/17

(Date)

# Autoregressive Linear Thermal Model of a Residential Forced-Air Heating System with Backpropagation Parameter Estimation Algorithm

Eric M. Burger, Scott J. Moura, David E. Culler

**Abstract**

Model predictive control (MPC) strategies show great potential for improving the performance and energy efficiency of building heating, ventilation, and air-conditioning (HVAC) systems. A challenge in the deployment of such predictive thermostatic control systems is the need to learn accurate models for the thermal characteristics of individual buildings. This necessitates the development of online and data-driven methods for system identification. In this paper, we propose an autoregressive with exogenous terms (ARX) model of a building. To learn the model, we present a backpropagation approach for recursively estimating the parameters. Finally, we fit the linear model to data collected from a residential building with a forced-air heating and ventilation system and validate the accuracy of the trained model.

*Keywords:* Thermal modeling, Control-oriented model, Autoregressive with exogenous terms (ARX) model, Backpropagation, Demand response, Heating, ventilation, and air-conditioning (HVAC)

## 1. Introduction

Heating, ventilation, and air-conditioning (HVAC) account for 43% of commercial and 54% of residential energy consumption [1]. Space heating alone accounts for 45% of all residential energy use. HVAC systems are an integral part of buildings responsible for regulating temperature, humidity, carbon dioxide, and airflow, conditions which directly impact occupant health and comfort. Estimates suggest that component upgrades and advanced HVAC control systems could reduce building energy usage by up to 30% [2]. Such intelligent systems can improve the efficiency of building operations, better regulate indoor conditions to improve air quality and occupant comfort, and enable buildings to participate in demand response services to improve power grid stability and reduce energy related carbon emissions [3, 4, 5, 6, 7, 8].

To effectively control the operation of an HVAC system, it is essential that a model predictive controller incorporate an accurate mathematical representation of a building's thermal dynamics. The processes that determine the evolution of temperatures within a building are complex and uncertain. A reliable model improves the ability of a controller to forecast conditions and meet cost, efficiency, and/or comfort objectives [9, 10]. Simulation software, such as EnergyPlus and TRNSYS, is capable of high fidelity modeling of building HVAC systems. These mathematical models play a crucial role in the architectural and mechanical design of new buildings, however, due to high dimensionality and computational complexity, are not suitable for incorporation into HVAC control systems [9, 11].

The American Society of Heating, Refrigeration, and Air-Conditioning Engineers (ASHRAE) handbook [12] describes how to determine the thermal resistance values of a building surface given it materials and construction type. However, for existing buildings, details about the materials in and construction of walls and windows may be difficult to obtain or non-existent [13]. Additionally, modifications to the building or changes brought about by time and use (e.g. cracks in windows or walls) further diminish the potential for characterizing a building based on design or construction information.

Therefore, an ideal control-oriented model would capture the predominant dynamics and disturbance patterns within a building, enable accurate fore-

casting, adapt to future changes in building use, provide a model structure suitable for optimization, and be amenable to real-time data-driven model identification methods. For these reasons, low order linear models are widely employed for control-oriented thermal building models [13, 14, 15]. Such models trade complexity and accuracy for simplicity and efficiency.

In this paper, we present an autoregressive with exogenous terms (ARX) model for the thermostatic control of buildings and a recursive backpropagation method for parameter estimation. The structure of the linear model enables the approximate identification of unmodeled dynamics, in particular higher-order dynamics and time delays related to changes in the mechanical state of the system. By employing a recursive parameter estimation technique, we are able to perform online data-driven learning of the model.

We do not model heating from solar gain, building occupants, or equipment. This does not restrict the applicability of this work because the model structure can be extended for such cases. By estimating these effects with a single time-varying gain, we produce a simpler model better suited for predictive control.

This paper is organized as follows. Section 2 presents our autoregressive exogenous thermal model and Section 3 overviews the parameter estimation problem. Section 4 formulates our recursive parameter estimation approach employing backpropagation and stochastic gradient descent. Section 5 provides numerical examples of our proposed model and algorithm for the parameter estimation of an apartment with a forced-air heating and ventilation system. Finally, Section 6 summarizes key results.

## 2. Building Thermal Model

### *2.1. Linear Thermal Model*

In this paper, we focus on the modeling of an apartment with a forced-air heating system. To begin, we consider a simple linear discrete time model [16, 17, 5, 4]

$$T^{k+1} = \theta_a T^k + \theta_b T_\infty^k + \theta_c m^k + \theta_d \qquad (1)$$

where $T^k \in \mathbf{R}$, $T_\infty^k \in \mathbf{R}$, and $m^k \in \{0, 1\}$ are the indoor air temperature (state, °C), outdoor air temperature (disturbance input, °C), and heater state (control input, On/Off), respectively, at time step $k$.

The parameters $\theta_a$ and $\theta_b$ correspond to the thermal characteristics of the conditioned space as defined by $\theta_a = \exp(-\Delta t/RC)$ and $\theta_b = 1 - \exp(-\Delta t/RC)$, $\theta_c$ to the energy transfer due to the system's mechanical state as defined by $\theta_b = (1 - \exp(-\Delta t/RC))RP$, and $\theta_d$ to an additive process accounting for energy gain or loss not directly modeled.

The linear discrete time model (1) is a discretization of a RC-equivalent continuous time model and thus derived from (very basic) concepts of heat transfer. As noted in [17, 5], the discrete time model implicitly assumes that all changes in mechanical state occur on the time steps of the simulation. In this paper, we assume that this behavior reflects the programming of the systems being modeled. In other words, we assume that the thermostat has a sampling frequency of $1/(3600\Delta t)$ Hz or once per minute.

### *2.2. Autoregressive Exogenous Thermal Model*

The linear discrete time model (1) is capable of representing the predominant thermal dynamics within a conditioned space. Unfortunately, because it does not capture any higher-order dynamics or time delays related to changes in the mechanical state of the system, the model is fairly inaccurate in practice. Research into higher-order RC models, in particular multi-zone network models and the modeling of walls as 2R-1C or 3R-2C elements, have shown potential for producing higher fidelity building models [13, 14, 15]. However, this comes at the cost of increasing the model complexity and the need for temperature sensing (in particular, within interior and exterior walls).

In this paper, we present an autoregressive exogenous (ARX) model capable of approximating dynamics related to trends in the ambient temperature and to changes in the mechanical state of the system. We note that the linear discrete time model (1) is, by definition, a first-order ARX model. The distinguishing characteristic of the ARX model presented below is that the model is higher-order with respect to the exogenous input terms. By increasing the number of exogenous input terms, we can better approximate observed dynamics in the systems. However, we will not pursue a physics-based justification for the number of exogenous terms and thus the ARX model represents a slight departure from

the practice of increasing the model order through RC-equivalent circuit modeling.

Our autoregressive exogenous (ARX) thermal model is given by

$$T^{k+1} = \theta_a T^k + \sum_{i=0}^{s-1} (\theta_{b,i} T_\infty^{k-i} + \theta_{c,i} m^{k-i}) + \theta_d \quad (2)$$

where $T^k \in \mathbf{R}$, $T_\infty^k \in \mathbf{R}$, and $m^k \in \{0, 1\}$ are the indoor air temperature (state, °C), outdoor air temperature (disturbance input, °C), and heater state (control input, On/Off), respectively, at time step $k$. The order of the exogenous terms (and thus the number of $\theta_b$ and $\theta_c$ parameters) is given by $s$.

The ARX model can be expressed more compactly as

$$T^{k+1} = \theta_a T^k + \theta_b^T \mathbf{T}_\infty^k + \theta_c^T \mathbf{m}^k + \theta_d \quad (3)$$

where $T^k \in \mathbf{R}$, $\mathbf{T}_\infty^k \in \mathbf{R}^s$, and $\mathbf{m}^k \in \{0, 1\}^s$ are the indoor air temperature (state, °C), previous outdoor air temperatures (disturbance input, °C), and previous heater states (control input, On/Off), respectively, at time step $k$. Lastly, $\theta_b \in \mathbf{R}^s$ and $\theta_c \in \mathbf{R}^s$ are the parameters of the exogenous terms.

## 3. Parameter Estimation Background

A fundamental machine learning problem involves the identification of a linear mapping

$$y^k = \theta^T x^k \quad (4)$$

where variable $x^k \in \mathbf{R}^X$ is the input, $y^k \in \mathbf{R}^Y$ is the output, and the linear map is parameterized by $\theta \in \mathbf{R}^{X \times Y}$. Additionally, $X$ and $Y$ are the number of inputs and outputs, respectively.

### *3.1. Batch Parameter Estimation*

Learning can be performed in a batch manner by producing $\hat{\theta}$, an estimate of the model parameters, given a training set of observed inputs and desired outputs, $\{x, y\}$. The goal of a parameter estimation algorithm is to minimize some function of the error between the desired and estimated outputs as given by $e^k = y^k - \hat{\theta}^T x^k$.

The least squares problem is given by

$$\underset{\hat{\theta}}{\text{minimize}} \ \frac{1}{2} \sum_{i=1}^{N} (\hat{\theta}^T x_i - y_i)^2 \quad (5)$$

with variables $x_i \in \mathbf{R}^n$, the model input for the $i$-th data point, $y_i \in \mathbf{R}$, the $i$-th observed response, $\hat{\theta} \in \mathbf{R}^n$, the weighting coefficients, and $i = 1, \ldots, N$, where $N$ is the number of data samples and $n$ is the number of features in $x_i$.

### *3.2. Recursive Parameter Estimation*

The least squares problem can be solved recursively with stochastic gradient descent as given by

$$\hat{\theta} := \hat{\theta} - \eta(\hat{\theta}^T x_k - y_k) \quad (6)$$

with variables $x_k \in \mathbf{R}^n$, the model input for at time step $k$, $y_k \in \mathbf{R}$, the observed response at time step $k$, $\hat{\theta} \in \mathbf{R}^n$, the weighting coefficients, and $\eta$, the learning rate.

## 4. Backward Propagation of Errors

A fundamental limitation of least squares regression when applied to autoregressive models of dynamical systems is that the optimization only minimizes the error of the output at one time step into the future. Thus, the model may produce a small error when employed to predict the state in the next time step but perform poorly when used to recursively produce a multiple time step forecast. To address this issue, we can represent the system as a multilayer neural network where each layer shares the same set of weights. By training the neural network with backpropagation and stochastic gradient descent, we can produce an estimate of the system's parameters that minimizes the output error multiple time steps into the future.

Backward propagation of errors, or backpropagation, is a technique commonly used for training multilayer artificial neural networks. The method consists of propagating an input forward through the layers of the neural network until the output layer is reached. The estimated output is then compared to the desired output to calculate an error value according to a loss function. Next, the error value is propagated backwards in order to calculate the relative contribution of each neuron in each layer to the network's estimated output. These relative contributions are used to calculate the gradient of the loss function with respect to the weights in the network. Finally, the weights of the network are updated according to a gradient-based optimization method, such as stochastic gradient descent, so as to minimize the loss function.

In this paper, we employ backpropagation to train the ARX thermal model (3) according to the optimization problems presented below. In each

case, we represent the system as a multilayer neural network where each layer shares the same set of weights. Thus, for a network with $\ell$ layers,

$$\begin{aligned}
\hat{T}^{k+1} &= \theta_a T^k + \theta_b^T \mathbf{T}_\infty^k + \theta_c^T \mathbf{m}^k + \theta_d \\
e_1^k &= T^{k+1} - \hat{T}^{k+1} \\
\hat{T}^{k+2} &= \theta_a \hat{T}^{k+1} + \theta_b^T \mathbf{T}_\infty^{k+1} + \theta_c^T \mathbf{m}^{k+1} + \theta_d \\
e_2^k &= T^{k+2} - \hat{T}^{k+2} \\
\hat{T}^{k+3} &= \theta_a \hat{T}^{k+2} + \theta_b^T \mathbf{T}_\infty^{k+2} + \theta_c^T \mathbf{m}^{k+2} + \theta_d \\
e_3^k &= T^{k+3} - \hat{T}^{k+3} \\
&\vdots \\
\hat{T}^{k+\ell} &= \theta_a \hat{T}^{k+\ell-1} + \\
&\qquad \theta_b^T \mathbf{T}_\infty^{k+\ell-1} + \theta_c^T \mathbf{m}^{k+\ell-1} + \theta_d \\
e_\ell^k &= T^{k+\ell} - \hat{T}^{k+\ell}
\end{aligned} \tag{7}$$

where $\hat{T}^{k+i}$ is the output of layer $i$ (i.e. the estimated temperature $i$ time steps from $k$) and $e_i^k$ is the error of the layer $i$ output (i.e. the error of the estimated temperature $i$ time steps from $k$). Note that the output of the first layer, $\hat{T}^{k+1}$, is a function of the measured temperature, $T^k$, whereas the output of each subsequent layer, $\hat{T}^{k+i+1}$, takes the output of the previous layer, $\hat{T}^{k+i}$, as an input.

Unlike a typical neural network, the activation function of each layer in our model is linear and we can express the output, $\hat{T}^{k+\ell}$, in terms of the measured temperature, $T^k$, as

$$\begin{aligned}
\hat{T}^{k+\ell} &= (\theta_a)^\ell T^k \\
&\quad + \sum_{i=1}^{\ell} (\theta_a)^{i-\ell} (\theta_b^T \mathbf{T}_\infty^{k+i-1} + \theta_c^T \mathbf{m}^{k+i-1}) \\
&\quad + \theta_d \sum_{i=1}^{\ell} (\theta_a)^{i-\ell} \\
e_\ell^k &= T^{k+\ell} - \hat{T}^{k+\ell}
\end{aligned} \tag{8}$$

Therefore, the neural network model is linear with respect to the inputs but nonlinear with respect to the parameters. This nonlinearity, as well as the forward propagation of noise, is a central challenge with respect to training the network. Furthermore, each of the training approaches presented in this paper are essentially nonlinear least square problems. However, for convenience, we employ the terminology used for multilayer neural networks to describe each parameter estimation approach.

Next, we present 3 approaches for training the multilayer neural network so as to produce estimates of the ARX model (3) that perform well when used to product multiple time step forecasts.

### *4.1. Final Error Backpropagation*

In our first training approach, we define our objective function so as to minimize the error of the final output layer of the neural network as given by

$$\underset{\hat{\theta}}{\text{minimize}} \ \frac{1}{2} \sum_{k=1}^{N} (e_\ell^k)^2 \tag{9}$$

with variables $e_\ell^k \in \mathbf{R}$, the output error of the final output layer (as defined in (7)) given the input and output data samples at time step $k$, $\hat{\theta} \in \mathbf{R}^{2s+2}$, the model parameter estimates (i.e. $\hat{\theta} = [\hat{\theta}_a, \hat{\theta}_b^T, \hat{\theta}_c^T, \hat{\theta}_d]^T$), and $k = 1, \ldots, N$, where $N$ is the number of data samples.

We solve the optimization program (9) recursively using backpropagation and stochastic gradient descent. Therefore, at each time step $k$, the stochastic gradient descent update equation is

$$\hat{\theta} \coloneqq \hat{\theta} - \eta \frac{\delta (e_\ell^k)^2}{\delta \hat{\theta}} \tag{10}$$

and the gradient of the loss function with respect to the parameters is

$$\begin{aligned}
\frac{\delta (e_\ell^k)^2}{\delta \hat{\theta}} &= \sum_{i=1}^{\ell} \frac{\delta (e_\ell^k)^2}{\delta \hat{T}^{k+\ell}} \frac{\delta \hat{T}^{k+\ell}}{\delta \hat{T}^{k+i}} \frac{\delta \hat{T}^{k+i}}{\delta \hat{\theta}} \\
&= \sum_{i=1}^{\ell} (e_\ell^k)(\hat{\theta}_a)^{\ell-i} \frac{\delta \hat{T}^{k+i}}{\delta \hat{\theta}} \\
&= \sum_{i=1}^{\ell} e_\ell^k (\hat{\theta}_a)^{\ell-i} \circ x_{i-1}^k
\end{aligned} \tag{11}$$

where

$$\begin{aligned}
x_0^k &= [T^k, (\mathbf{T}_\infty^k)^T, (\mathbf{m}^k)^T, 1]^T \\
x_{i-1}^k &= [\hat{T}^{k+i}, (\mathbf{T}_\infty^{k+i})^T, (\mathbf{m}^{k+i})^T, 1]^T \\
&\quad \forall i = 2, \ldots, \ell
\end{aligned} \tag{12}$$

Note that with this training approach, we only backpropagate the error of the final output layer. The assumption is that by minimizing the final output error, we will minimize the error of every layer in the network. In the following training approaches, we incorporate the output errors of multiple layers into the loss function in an effort to improve the robustness of the model training.

*4.2. All Error Backpropagation*

In our second training approach, we define our objective function so as to minimize the error of each layer in the neural network as given by

$$\underset{\hat{\theta}}{\text{minimize}} \ \frac{1}{2}\sum_{k=1}^{N}\sum_{i=1}^{\ell}(e_i^k)^2 \tag{13}$$

with variables $e_i^k \in \mathbf{R}$, the output error of each layer $i$ (as defined in (7)) given the input and output data samples at $k$, $\hat{\theta} \in \mathbf{R}^{2s+2}$, the model parameter estimates (i.e. $\hat{\theta} = [\hat{\theta}_a, \hat{\theta}_b^T, \hat{\theta}_c^T, \hat{\theta}_d]^T$), and $k = 1, \ldots, N$, where $N$ is the number of data samples.

We solve the optimization program (13) recursively using backpropagation and stochastic gradient descent. Therefore, at each time step $k$, the stochastic gradient descent update equation is

$$\hat{\theta} \coloneqq \hat{\theta} - \eta \frac{\delta \sum_{i=1}^{\ell}(e_i^k)^2}{\delta \hat{\theta}} \tag{14}$$

and the gradient of the loss function with respect to the parameters is

$$\frac{\delta \sum_{i=1}^{\ell}(e_i^k)^2}{\delta \hat{\theta}} = \sum_{i=1}^{\ell}\sum_{j=1}^{i} e_i^k (\hat{\theta}_a)^{i-j} \circ x_{j-1}^k \tag{15}$$

where $x_{i-1}^k$ is defined in (12).

*4.3. Partial Error Backpropagation*

An issue with the Final Error Backpropagation and All Error Backpropagation methods presented above is that, for large values of $\ell$, we are propagating the errors backwards over many time steps. However, given that we are using the neural network model to represent a dynamical system, there may be very little signal between the input at time step $k$ and the output at time step $k+\ell$. This potential lack of signal between the input and output is a well known issue with training deep artificial neural networks using backpropagation and gradient-based optimization methods and can result in what is often described as the vanishing (or exploding) gradient problem.

In our case, the issue stems from the exponential terms in (11) and (15). Specifically, small values of $\hat{\theta}_a$ may cause the gradient to "vanish" while large values may cause the gradient to "explode". To address this, our third training approach will backpropagate the errors of each layer a maximum of $\beta$ time steps. As with the All Error Backpropagation method, the objective function is defined so as to minimize the error of each layer in the neural network as given by (13) and the stochastic gradient descent update equation is (14).

However, for the Partial Error Backpropagation approach, we approximate the gradient of the loss function with respect to the parameters as

$$\frac{\delta \sum_{i=1}^{\ell}(e_i^k)^2}{\delta \hat{\theta}} \approx \sum_{i=1}^{\ell}\sum_{j=f(i,\beta)}^{i} e_i^k (\hat{\theta}_a)^{i-j} \circ x_{j-1}^k \tag{16}$$

where $x_{i-1}^k$ is defined in (12) and $f(i,\beta)$ is given by

$$f(i,\beta) = \max(1, i - \beta + 1) \tag{17}$$

Note that with the Partial Error Backpropagation method, the output error $e_i^k$ of each layer $i$ is backpropagated a maximum of $\beta$ layers (i.e. backwards $\beta$ time steps).

*4.4. Growing the Neural Network*

When training the neural network using the 3 methods described above, poor initial estimates of the parameter values will cause the algorithm to diverge. Therefore, it is necessary to start with a shallow network and gradually increase the depth as the parameter estimates improve. In other words, when training the model, we start with a small value of $\ell$. Once the algorithm has converged, we increase the value of $\ell$ and continue to recursively update the parameters. We repeat this procedure until the neural network has reached the desired depth (i.e. desired value of $\ell$).

## 5. Residential Heating System Parameter Estimation Experiments

In this section, we present parameter estimation results for an 850 sq ft apartment with a forced-air heating and ventilation system. The apartment is located in Berkeley, California and equipped with a custom thermostat designed and built for this research. Therefore, we are able to control the operation of the heating system and to measure the indoor air temperature. Local weather data, specifically ambient air temperature, is retrieved from the Internet service, Weather Underground [18].

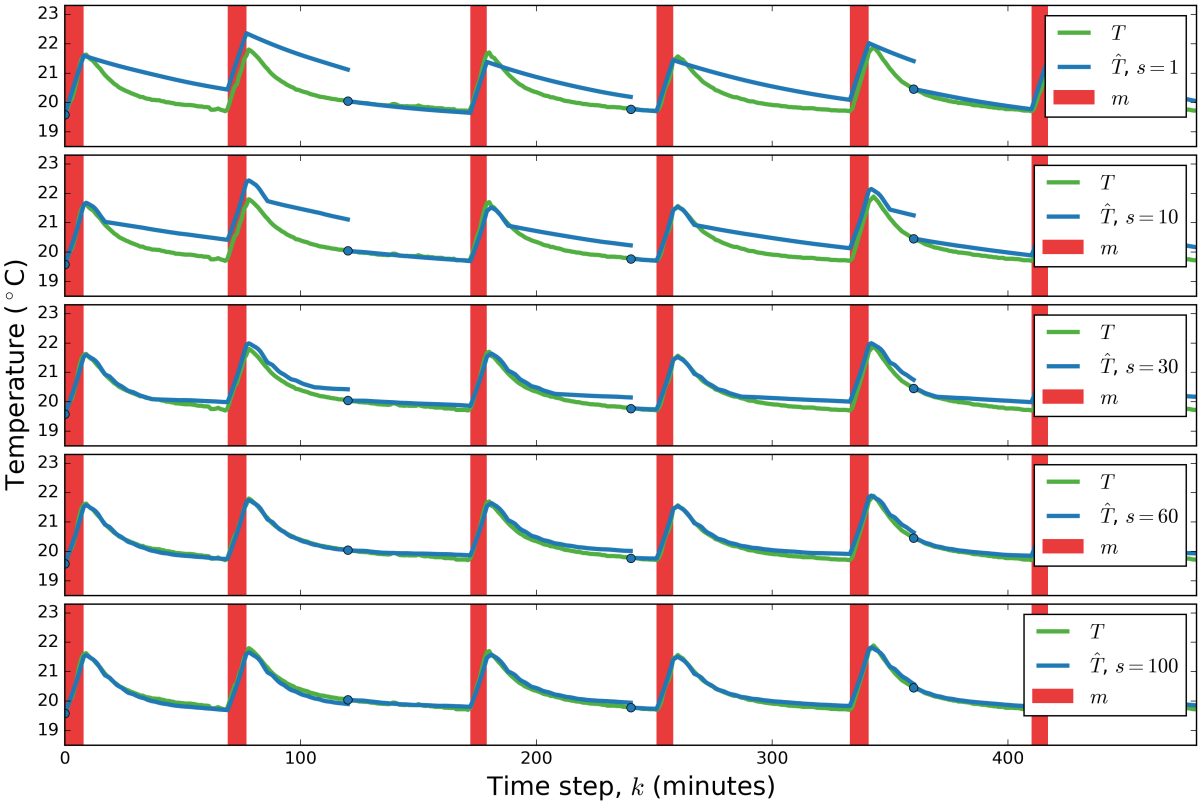



Figure 1: Examples of 2 hour temperature forecasts over 8 hours of test data generated by ARX models with varying numbers of exogenous input terms

Data was collect at a time-scale of one minute for 6 weeks during December and January of 2015-2016. With this data, we are able to perform recursive parameter estimation of the ARX thermal model (2). The results presented in this section focus of quantifying and qualifying the advantages of the ARX model and the backpropagation parameter estimation methods presented above.

### *5.1. Increasing Model Order*

With the ARX model, we are able to adjust the number of exogenous terms, $s$, based on the dynamics of a particular conditioned space. Increasing the number of exogenous terms increases the computational cost of training and employing the ARX model. Therefore, we want to find the minimum value of $s$ such that the model performs well for a specific system.

To evaluate the sensitivity of the ARX model to the number of exogenous terms, we have trained the model using different values of $s$. In each case, the model is trained using batch least squares on 80% of the sensor data (i.e. training data) and the model performance is evaluated by producing multi-hour forecasts with the remaining 20% of the data (i.e. test data).

Figures 1 and 2 present examples of 2 hour temperature forecasts produced by ARX models with varying numbers of exogenous input terms. The top subplots show forecasts from an ARX model with $s = 1$, which is equivalent to the linear thermal model in (1). As shown, the model is simply incapable of representing the evolution of the indoor air temperature. Most notably, the forecasts poorly account for the thermal dynamics immediately after the heating system turns off. These dynamics are related to the interaction between the air and the other thermal masses (walls, furniture, etc.) within the conditioned space.

By increasing $s$ to 10, the ARX model is able to better represent the dynamics immediately after the heating system turns off. However, we observe an elbow in the temperature forecasts at 10 time steps after the heating system turns off, as

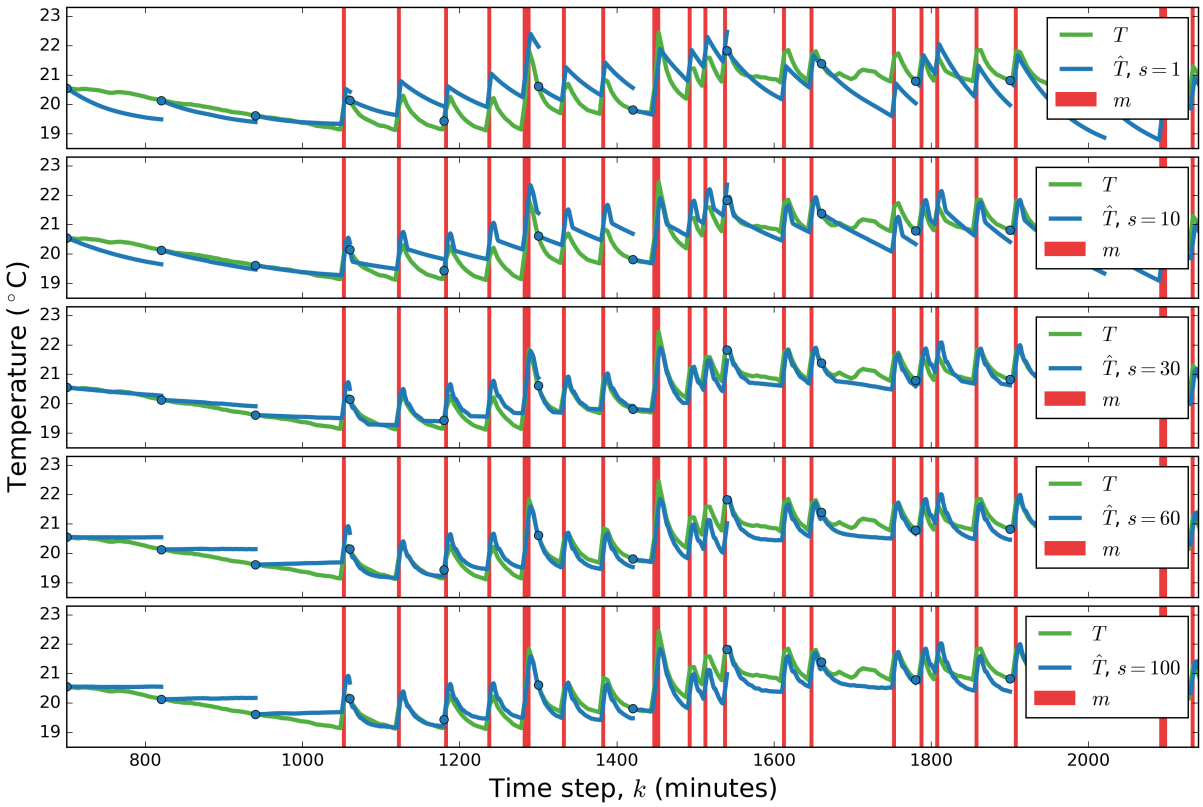


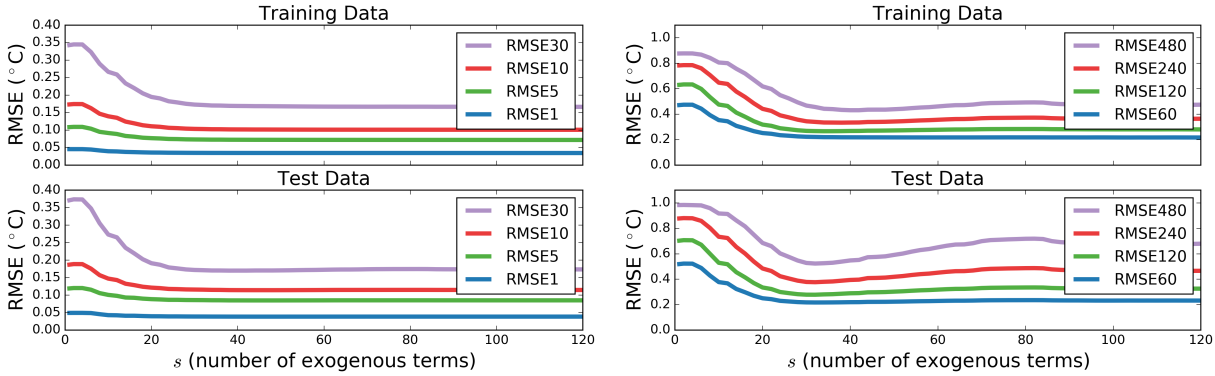

Figure 2: Examples of 2 hour temperature forecasts over 24 hours of test data generated by ARX models with varying numbers of exogenous input terms

shown in the second subplot. This suggests that the conditioned space is still responding to the change in state of the heating system, but that the ARX model no longer has any knowledge of the state change and thus cannot estimate its impact on the indoor air temperature.

By increasing $s$ to 30, the model is able to better represent the dynamics of the conditioned space from the time the heating system turns off until it turns on again. This is an intuitive result and indicates that $s$ must be sufficiently large so as to capture a full cycle of the heating system. Increasing $s$ to 60 and 100, as shown in the bottom 2 subplots, does not significantly improve the accuracy of the 2 hour forecasts. In other words, each additional exogenous input increases the complexity of the ARX model but provides less information than the previous input.

Figure 3 shows the performance of ARX models with varying numbers of exogenous terms, $s$, when used to generate forecasts of 1, 5, 10, 30, 60, 120, 240, and 480 time steps. Each ARX model was trained using batch least squares on 80% of the sensor data (i.e. training data) and the model performance was evaluated by producing forecasts with the remaining 20% of the data (i.e. test data). The performance of each model is measured as the root mean squared error (RMSE) of all multiple time step forecasts of a certain length. In other words, the RMSE60 is the RMSE of all 60 time step forecasts over a given data set. For comparison, Figure 3 includes the performances of each ARX model when used to produce forecasts on both the training data and test data.

As shown in Figure 3, the RMSEs of the ARX models over horizons of 1, 5, 10, and 30 time steps decrease as $s$ increases from 1 to 30 and level off at around 40. The RMSEs of the 240 and 480 time step forecasts also decrease at first, but begin to increase as $s$ increases from 40 to 80, particularly for the test data. A simple (though imprecise) explanation of this behavior is that we are underfitting the model when $s$ is less than 30 and overfitting when $s$ is greater than 40. The lowest RMSE1




(a) RMSE of 1, 5, 10, and 30 time step forecasts

(b) RMSE of 60, 120, 240, and 480 time step forecasts

Figure 3: Performance (RMSE) of ARX models with varying numbers of exogenous input terms on training and test data when used to generate forecasts of 1, 5, 10, 30, 60, 120, 240, and 480 time steps

(i.e. the RMSE of all 1-minute forecasts) on the training data is 0.0343°C when $s = 120$ and on the test data is 0.0384°C when $s = 48$. Since the least squares optimization problem minimizes the 1 time step ahead error, it is no surprise that each additional exogenous terms reduces the RMSE1 of the training data. By contrast, the lowest RMSE480 (i.e. the RMSE of all 8-hour forecasts) on the training data is 0.431°C when $s = 40$ and on the test data is 0.523°C when $s = 32$. With the longer forecast horizon, we see more agreement between the training and test performances with respect to the optimal number of exogenous terms.

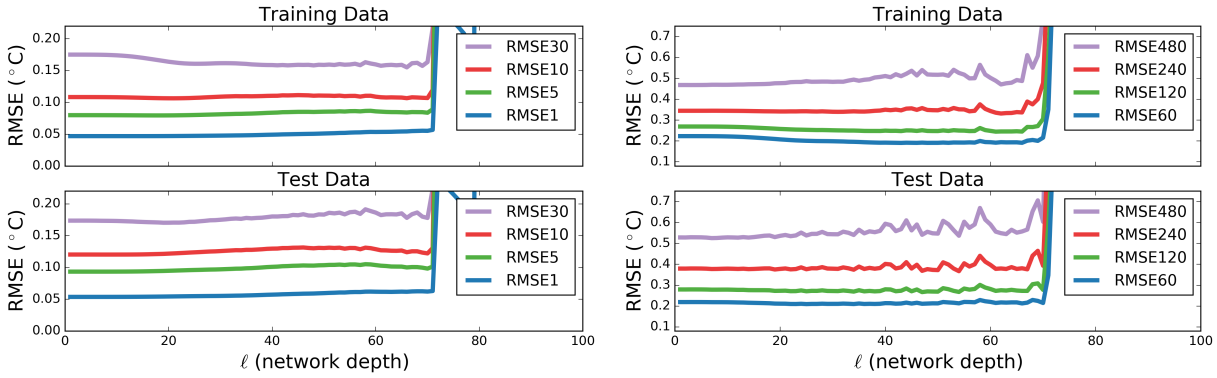
### *5.2. Backpropagation Methods and Increasing Neural Network Depth*

In this section, we present results from training the ARX model using the 3 backpropagation methods: Final Error Backpropagation, All Error Backpropagation, and Partial Error Backpropagation. Once again, we use 80% of the sensor data collected from the apartment as training data and the remaining 20% of the data as test data. For each backpropagation method, we train ARX models with 30, 60, and 100 exogenous terms. Additionally, each model is trained with different numbers of neural network layers, $\ell$. Increasing the number of layers increases the computational cost of training the model and therefore, we want to find the minimum value of $\ell$ such that the model performs well for a specific system.

As previously noted, poor initial parameter estimates will cause the training algorithm to diverge. Therefore, when training a network with depth $\ell$, we initialize the parameters with estimates from a network of depth $\ell - 1$. For a network of depth $\ell = 1$, we train the model using least squares rather than backpropagation and stochastic gradient descent. Lastly, to reduce the likelihood that the stochastic gradient descent algorithm diverges for large values of $\ell$, we set a small learning rate, $\eta$, of $3 * 10^{-9}$ and limit the number of iterations (i.e. number of stochastic gradient descent updates) to 200,000.

Results from training the ARX models using the Final Error Backpropagation method are presented in Figures 4, 5, and 6. For the ARX model with $s = 30$ exogenous terms, we observe little to no improvement in the forecast error as a result of the backpropagation training method. In fact, the lowest RMSE480 on the training data is 0.468°C when $\ell = 3$ and on the test data is 0.525°C when $\ell = 7$. As the depth of the neural network increases, the accuracy of the forecasts remain relatively stable until $\ell$ reaches about 40. With an $\ell$ of 70, we start to experience exploding gradients resulting in poor parameter estimates and a sharp increase in the RMSEs of the forecasts.

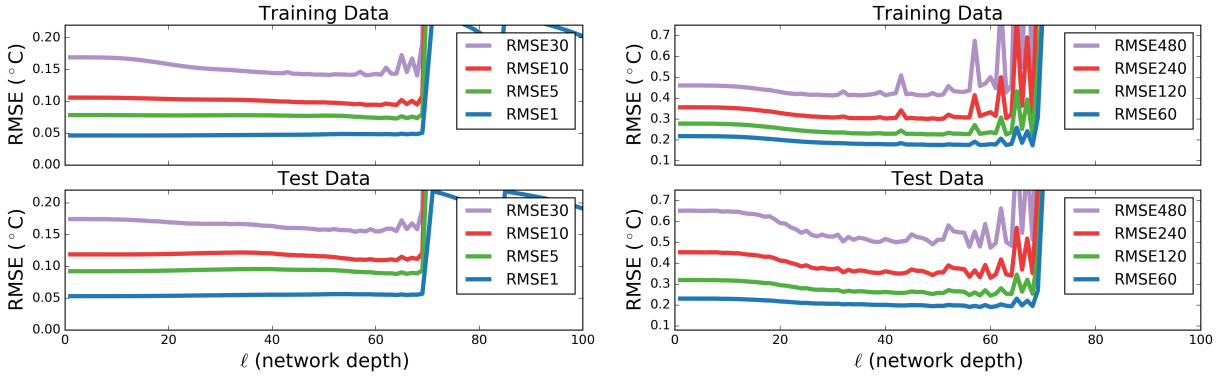
Figures 5 and 6 present results from ARX models with $s = 60$ and $s = 100$ exogenous terms. As discussed in the previous section, training models with such large numbers of exogenous terms using least squares caused overfitting and an increase in the RMSE240 and RMSE480. Using the Final Error Backpropagation method, we are able to improve the performance of both models on the training and test data. In fact, we are able to produce 8-hour forecasts that are, on average, more accu-




(a) RMSE of 1, 5, 10, and 30 time step forecasts

(b) RMSE of 60, 120, 240, and 480 time step forecasts

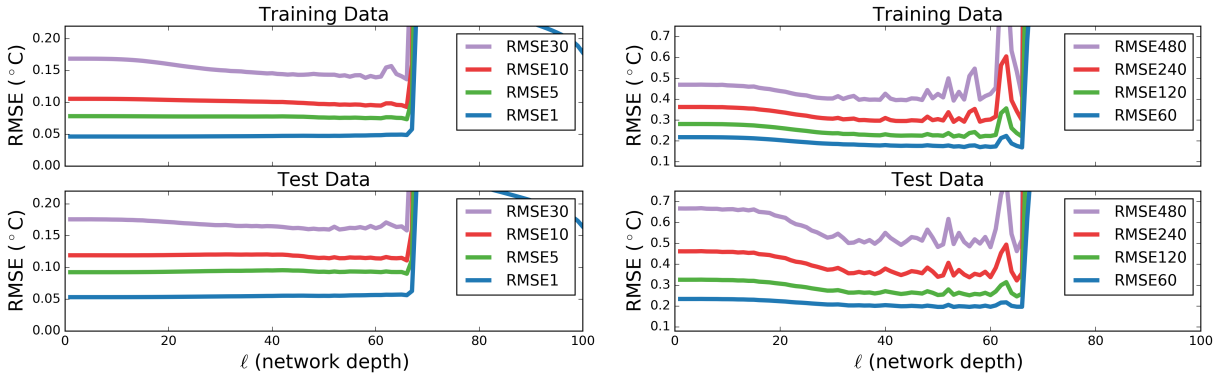
Figure 4: Performance (RMSE) of ARX model with $s = 30$ exogenous input terms when trained using Final Error Backpropagation with varying neural network depth, $\ell$




(a) RMSE of 1, 5, 10, and 30 time step forecasts

(b) RMSE of 60, 120, 240, and 480 time step forecasts

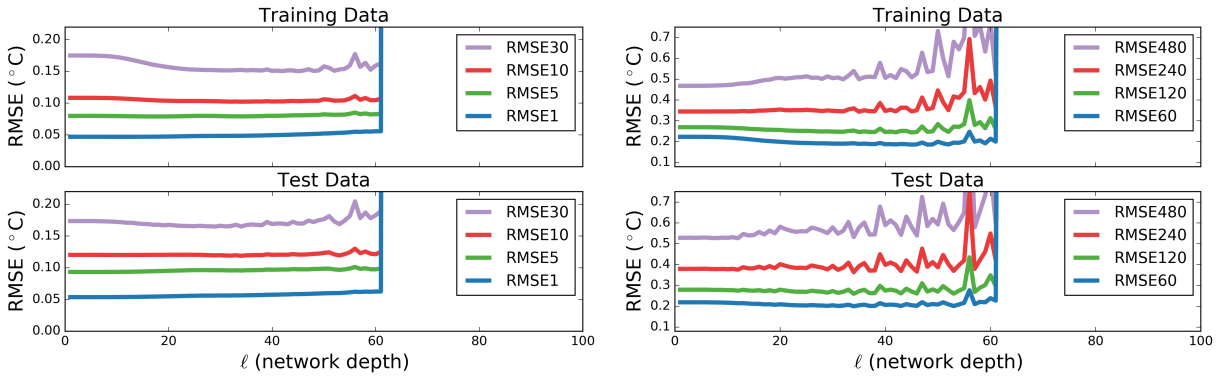
Figure 5: Performance (RMSE) of ARX model with $s = 60$ exogenous input terms when trained using Final Error Backpropagation with varying neural network depth, $\ell$




(a) RMSE of 1, 5, 10, and 30 time step forecasts

(b) RMSE of 60, 120, 240, and 480 time step forecasts

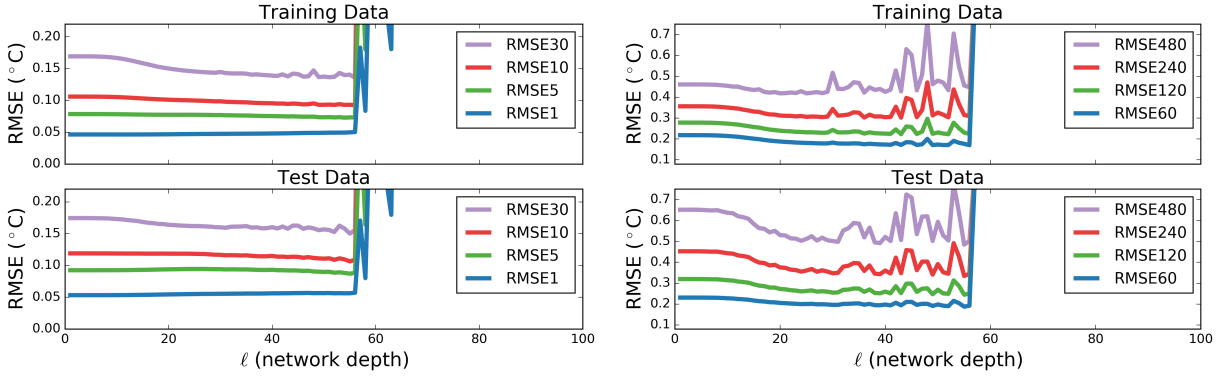
Figure 6: Performance (RMSE) of ARX model with $s = 100$ exogenous input terms when trained using Final Error Backpropagation with varying neural network depth, $\ell$




(a) RMSE of 1, 5, 10, and 30 time step forecasts

(b) RMSE of 60, 120, 240, and 480 time step forecasts

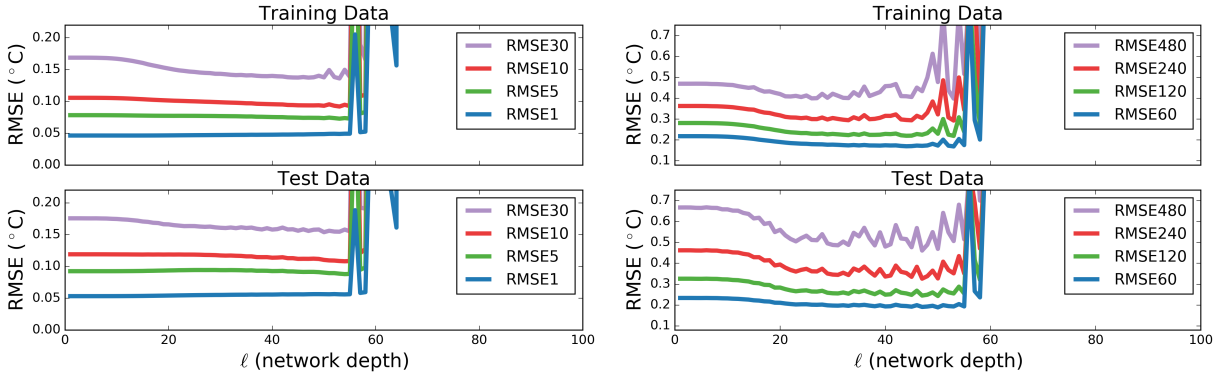
Figure 7: Performance (RMSE) of ARX model with $s = 30$ exogenous input terms when trained using All Error Backpropagation with varying neural network depth, $\ell$




(a) RMSE of 1, 5, 10, and 30 time step forecasts

(b) RMSE of 60, 120, 240, and 480 time step forecasts

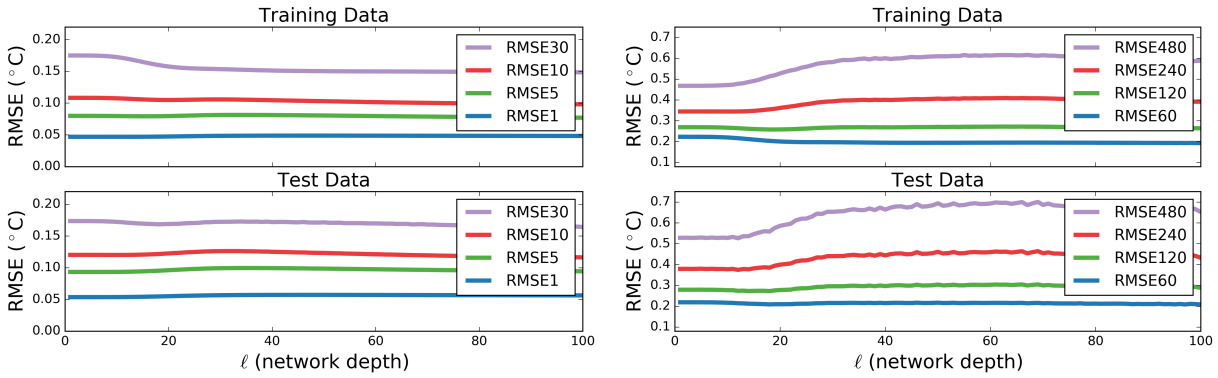
Figure 8: Performance (RMSE) of ARX model with $s = 60$ exogenous input terms when trained using All Error Backpropagation with varying neural network depth, $\ell$




(a) RMSE of 1, 5, 10, and 30 time step forecasts

(b) RMSE of 60, 120, 240, and 480 time step forecasts

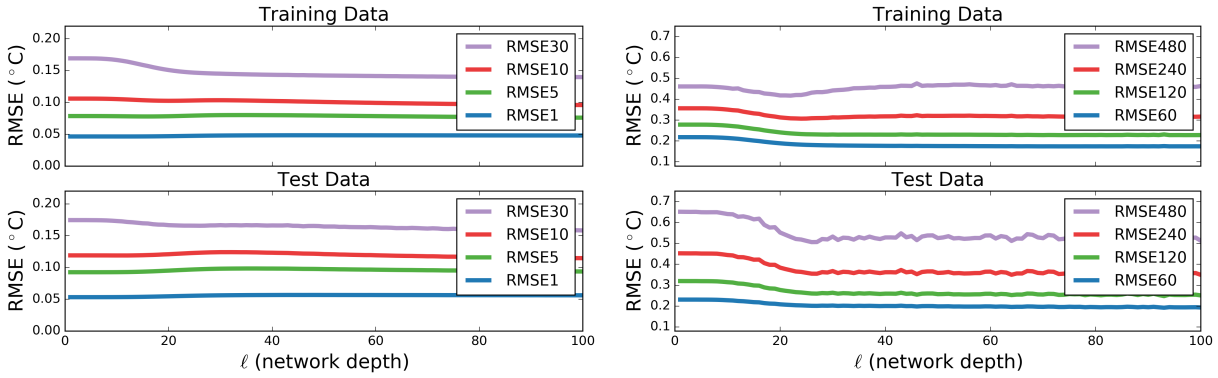
Figure 9: Performance (RMSE) of ARX model with $s = 100$ exogenous input terms when trained using All Error Backpropagation with varying neural network depth, $\ell$




(a) RMSE of 1, 5, 10, and 30 time step forecasts

(b) RMSE of 60, 120, 240, and 480 time step forecasts

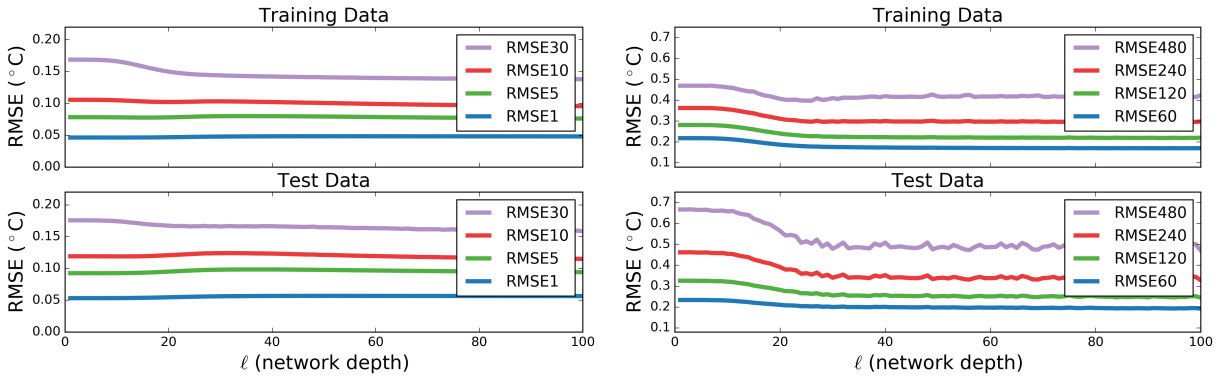
Figure 10: Performance (RMSE) of ARX model with $s = 30$ exogenous input terms when trained using Partial Error Backpropagation with a backpropagation limit of $\beta = 5$ and varying neural network depth, $\ell$




(a) RMSE of 1, 5, 10, and 30 time step forecasts

(b) RMSE of 60, 120, 240, and 480 time step forecasts

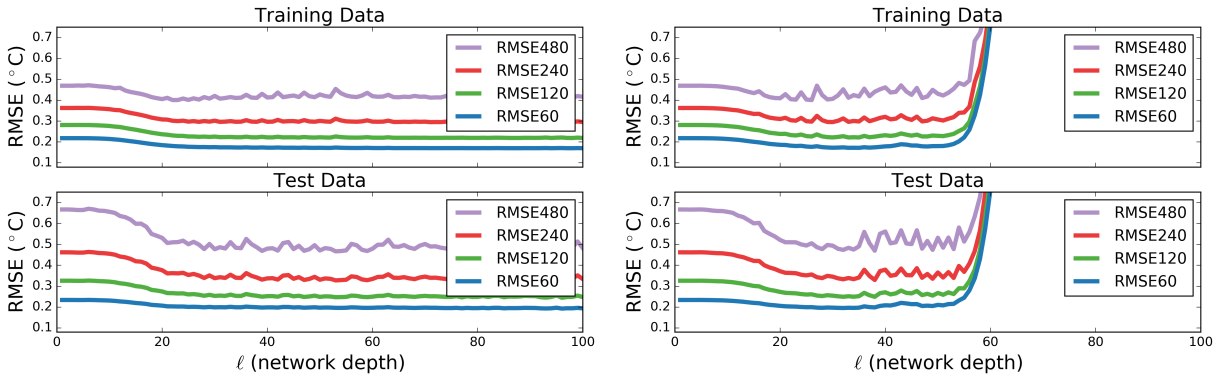
Figure 11: Performance (RMSE) of ARX model with $s = 60$ exogenous input terms when trained using Partial Error Backpropagation with a backpropagation limit of $\beta = 5$ and varying neural network depth, $\ell$




(a) RMSE of 1, 5, 10, and 30 time step forecasts

(b) RMSE of 60, 120, 240, and 480 time step forecasts

Figure 12: Performance (RMSE) of ARX model with $s = 100$ exogenous input terms when trained using Partial Error Backpropagation with a backpropagation limit of $\beta = 5$ and varying neural network depth, $\ell$




(a) RMSE of 60, 120, 240, and 480 time step forecasts with a backpropagation limit of $\beta = 10$

(b) RMSE of 60, 120, 240, and 480 time step forecasts with a backpropagation limit of $\beta = 20$

Figure 13: Performance (RMSE) of ARX model with $s = 100$ exogenous input terms when trained using Partial Error Backpropagation with a backpropagation limit of $\beta = 10$ and $\beta = 20$ and varying neural network depth, $\ell$

rate than with the $s = 30$ model. For the $s = 60$ ARX model, the lowest RMSE480 on the training data is 0.413°C when $\ell = 30$ and on the test data is 0.475°C when $\ell = 60$. For the $s = 100$ ARX model, the lowest RMSE480 on the training data is 0.394°C when $\ell = 53$ and on the test data is 0.463°C when $\ell = 65$. Once again, with an $\ell$ of 70, we start to experience exploding gradients and a sharp increase in forecast error.

Results from training the ARX models using the All Error Backpropagation method are presented in Figures 7, 8, and 9. For the ARX model with $s = 30$ exogenous terms, we observe an overall increase in forecast error as a result of the backpropagation training method. The lowest RMSE480 on the training data is 0.468°C when $\ell = 3$ and on the test data is 0.527°C when $\ell = 12$. By contrast, for the $s = 60$ and $s = 100$ ARX models, we again see an improvement in the model performance as a result of the backpropagation training method. For the $s = 60$ ARX model, the lowest RMSE480 on the training data is 0.419°C when $\ell = 25$ and on the test data is 0.485°C when $\ell = 55$. For the $s = 100$ ARX model, the lowest RMSE480 on the training data is 0.398°C when $\ell = 53$ and on the test data is 0.461°C when $\ell = 47$. The performances of the ARX models exhibit greater variability when trained with the All Error Backpropagation method than compared with the Final Error Backpropagation approach and we observe a sharp increase in forecast error at an $\ell$ of about 60 due to exploding gradients.

Results from training the ARX models using the Partial Error Backpropagation method are presented in Figures 10, 11, 12, and 13. Figures 10, 11, and 12 present results from training ARX models using a maximum of $\beta = 5$ layers for backpropagation and Figure 13 presents results using $\beta = 10$ and $\beta = 20$. Unlike with Final Error Backpropagation and All Error Backpropagation, we only observe divergence in the gradient descent algorithm for the $\beta = 20$ case when using Partial Error Backpropagation. For the other cases, the algorithm remains stable (or as stable as can be expected of stochastic gradient descent) even at large values of $\ell$. This suggests that by limiting the number of neural network layers through which the errors are backpropagated, we can approximate the gradient of the objective function and reduce the risk of exploding gradients.

For the ARX model with $s = 30$ exogenous terms, the lowest RMSE480 on the training data is 0.468°C when $\ell = 2$ and on the test data is 0.526°C when $\ell = 12$. These results are very close to those when trained with Final Error Backpropagation and All Error Backpropagation. For the $s = 60$ ARX model, the lowest RMSE480 on the training data is 0.417°C when $\ell = 22$ and on the test data is 0.501°C when $\ell = 93$. For the $s = 100$ ARX model, the lowest RMSE480 on the training data is 0.398°C when $\ell = 26$ and on the test data is 0.470°C when $\ell = 100$. Note that with Partial Error Backpropagation for the $s = 60$ and $s = 100$ cases, the test error is minimized with an $\ell$ greater than 90. With the previous training approaches, the gradient descent algorithm began to diverge

with an $\ell$ of around 60. If we increase $\beta$ to 10, the lowest RMSE480 of the $s = 100$ ARX model on the training data is 0.401°C when $\ell = 22$ and on the test data is 0.465°C when $\ell = 96$. By increasing $\beta$ again to 20, the lowest RMSE480 on the training data becomes 0.400°C when $\ell = 26$ and on the test data becomes 0.470°C when $\ell = 38$. As previously noted, with $s = 100$ and $\beta = 20$, the algorithm diverges at around $\ell = 60$.

Using the Final Error Backpropagation, All Error Backpropagation, and Partial Error Backpropagation approaches, the lowest RMSE480 values on the test data were 0.463°C, 0.461°C, and 0.465°C, respectively. Each of these was achieved by an ARX model with $s = 100$ exogenous terms. Given the clear potential for instability in the Final Error Backpropagation and All Error Backpropagation methods, these parameter estimation methods are poorly suited for control applications. However, given the greater stability and comparable model performances (as measured by the RMSE480 values), the Partial Error Backpropagation method presented in this paper has the greatest potential for improving the accuracy of the ARX model by minimizing the output error over multiple time steps rather than one time step into the future.

### *5.3. Control Simulations*

In the previous sections, the performance of each ARX model is quantified using the root mean squared error (RMSE) of all multiple time step forecasts of a certain length for a given data set. These RMSE values are useful for understanding the accuracy of the model (in a statistical sense) and measuring the capability of the model to estimate the air temperature of the conditioned space given the mechanical state (On/Off) of the system. For applications like temperature estimation (e.g. Kalman filtering of temperature measurements) and fault detection (e.g. detecting if the system has failed to deliver heat to the conditioned space), we would like a model with a low RMSE value. However, the RMSE of the temperature forecasts is not sufficient for quantifying the fidelity of the ARX model or its suitability for model predictive control applications.

In this section, we present results from control simulations in which various ARX models are used to estimate both the indoor air temperature, $T^k$, and mechanical state, $m^k$, given the outdoor air temperature, $T^k_\infty$, and upper and lower temperature bounds at each time step $k$. Each ARX model has 30, 60, or 100 exogenous input terms ($s$=30, 60, or 100) and is fit to the training data using batch least squares ($\ell = 1$) or Partial Error Backpropagation with $\beta = 5$ and a network depth of 20, 30, 40, or 50 layers ($\ell$=20, 30, 40, or 50). We simulate the control of the system using the outdoor air temperature, $T^k_\infty$, and temperature setpoints, $T^k_{set}$, of the test data set. The indoor temperature and mechanical state estimates are initialized with measured data (i.e. $\hat{T}^0 = T^0$ and $\hat{m}^0 = m^0$) and the evolution of the states are given by the update equations

$$\begin{aligned} T^{k+1} &= \theta_a T^k + \theta_b^T \mathbf{T}^k_\infty + \theta_c^T \mathbf{m}^k + \theta_d \\ m^{k+1} &= \begin{cases} 1 & \text{if } T^{k+1} < T^k_{set} - \frac{\delta}{2} \\ 0 & \text{if } T^{k+1} > T^k_{set} + \frac{\delta}{2} \\ m^k & \text{otherwise} \end{cases} \end{aligned} \tag{18}$$

where $T^k \in \mathbf{R}$ and $m^k \in \{0, 1\}$ are the indoor air temperature (°C) and heater state (On/Off), respectively, at time step $k$. As in (3), $\mathbf{T}^k_\infty \in \mathbf{R}^s$ denotes the previous outdoor air temperatures (°C), $\mathbf{m}^k \in \{0, 1\}^s$, the previous heater states ($\mathbf{m}^k = [m^k, m^{k-1}, \ldots, m^{k-s+1}]^T$), and $\theta_b \in \mathbf{R}^s$ and $\theta_c \in \mathbf{R}^s$, the parameters of the exogenous terms. Lastly, $T^k_{set} \in \mathbf{R}$ denotes the temperature setpoint (°C) at time step $k$ and $\delta \in \mathbf{R}$, the temperature deadband width (°C). Therefore, at each time step $k$, the upper temperature bound is $T^k_{set} + \delta/2$ and the lower temperature bound is $T^k_{set} - \delta/2$.

Examples of temperature estimates, $\hat{T}^k$, and mechanical state estimates, $\hat{m}^k$, produced by ARX models trained with batch least squares and simulated with (18) are presented in Figure 14. The top subplot shows the measured temperature, $T^k$, and mechanical state, $m^k$, and the remaining four subplots show estimates from ARX models with $s$=10, 30, 60, and 100 exogenous terms.

Figure 15 summarizes results from the control simulation tests. As shown in the subfigures, we employ four metrics to quantify the fidelity of the ARX models with respect to observations of the forced-air heating system. In Figure 15(a), we compare the number of time steps that the temperature estimates are within the upper and lower bounds and the number of time steps that the mechanical system is on according to the control simulation of each ARX model. The results are presented as errors relative to the observed number of time steps within the deadband and time steps that the system in on, respectively. In Figure 15(b), we show the root mean squared error

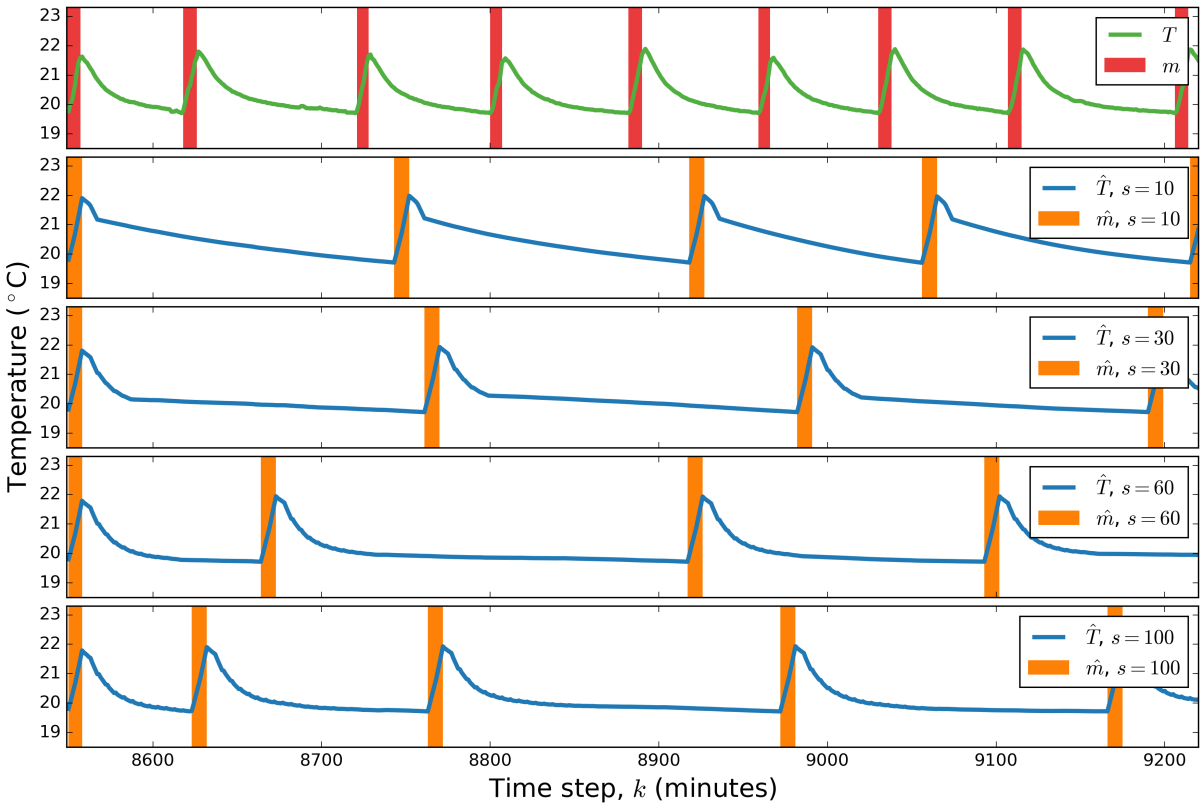


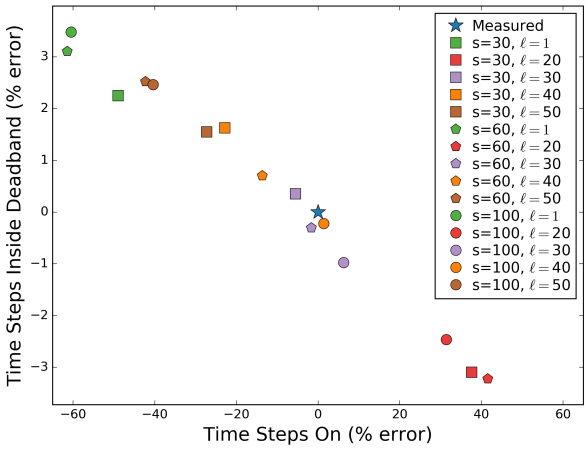

Figure 14: Examples of temperature and mechanical state estimates over test data generated by ARX models with varying numbers of exogenous input terms and trained with batch least squares

(RMSE) of temperature deviations above the upper bound ($T^{k+1} > T^k_{set} + \delta/2$) and below the lower bound ($T^{k+1} < T^k_{set} - \delta/2$). The errors are calculated relative to the upper or lower bounds and temperatures within the bounds have an error of 0.

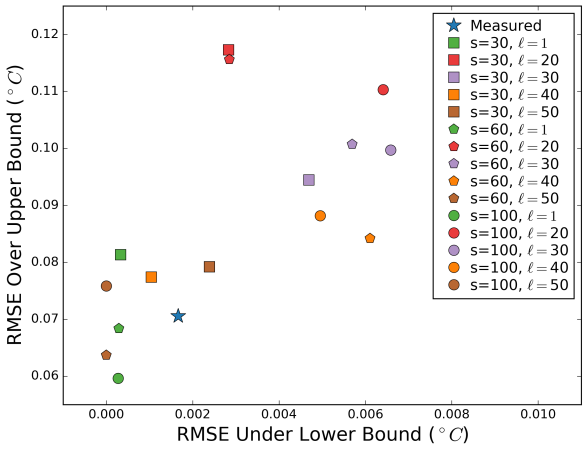
As shown in Figure 15(a), and to a lesser degree, Figure 14, the ARX models trained with batch least squares ($\ell = 1$) tend to overestimate the number of time steps that the indoor temperature is within the upper and lower bounds. Similarly, these models underestimate the number of time steps that the system is on and therefore underestimate the energy required to maintain the temperature within the conditioned space. Using Partial Error Backpropagation with $\ell = 20$, the ARX models underestimate the time steps within the deadband and overestimate the energy demand. When we increase $\ell$ to 30 and 40, the relative errors of the ARX models move closer to 0, suggesting that the temperature and mechanical state estimates better reflect observed dynamics of the system. However, increasing $\ell$ to 50 causes the ARX models to once again overestimate the time steps within the deadband and underestimate the number of time steps that the system is on.

Figure 15(b) shows the RMSE values of temperature deviations above the upper bound and below the lower bound for the temperature measurements, $T^k$, and temperature estimates, $\hat{T}^k$, produced by the control simulations. Given the formulation of the update equations in (18), some deviation outside the bounds is necessary to change the mechanical state. Accordingly, for the various models, we would like to observe RMSE values that are close to those of the temperature measurements, indicating that the ARX model accurately represents the dynamics of the system just after it turns on or off.

As shown in the subfigure, the models trained with $\ell = 1$ underestimate deviations below the lower bound but do a relatively good job of estimating deviations above the upper bound. For $\ell = 20$, the models overestimate the deviations above the upper bound. In other words, these models overshoot the upper bound, which may help to explain




(a) Percent error of estimated number of time steps that the temperature is within the upper and lower bounds (y-axis) and the heating system is on (x-axis)




(b) RMSE of temperature deviations above the upper bound (y-axis) and below the lower bound (x-axis)

Figure 15: Fidelity of ARX model with $s = 100$ exogenous input terms when trained using Partial Error Backpropagation with a backpropagation limit of $\beta = 5$ and varying neural network depth, $\ell$

why these models also overestimate the number of time steps that the system is on. While the models trained with an $\ell$ of 30 and 40 produced good estimates of the number of time steps within the deadband and the number of time steps that the system is on, they overestimate the deviations above and below the temperature bounds. Lastly, with $\ell = 50$, we achieve RMSE values that are relatively close to those of the measured data.

Based on the results presented in Figure 15, we conclude that the Partial Error Backpropagation training method does have potential to improve the fidelity of the ARX models relative to batch least squares. This improvement in model fidelity is particularly important for model predictive control applications which estimate and optimize the energy demand of residential heating systems. However, further research is necessary to optimize for model fidelity during the training of the ARX models. Potential research directions include stopping criteria which include a control simulation, the training of an ensemble of ARX models with model selection based on a control simulation, and the development of a non-linear training technique (e.g. genetic algorithm, particle swarm optimization) which optimizes both the accuracy (RMSE of multi time step forecasts) and fidelity (error of control simulations) of an ARX model.

## 6. Conclusions

This paper addresses the need for control-oriented thermal models of buildings. We present an autoregressive with exogenous terms (ARX) model of a building that is suitable for model predictive control applications. To estimate the model parameters, we present 3 backpropagation and stochastic gradient descent methods for recursive parameter estimation: Final Error Backpropagation, All Error Backpropagation, and Partial Error Backpropagation. Finally, we present experimental results using real temperature data collected from an apartment with a forced-air heating and ventilation system. These results demonstrate the potential of the ARX model and Partial Error Backpropagation parameter estimation method to produce accurate forecasts of the air temperature within the apartment.

## 7. References

[1] U.S. Department of Energy, 2010 Buildings Energy Data Book, accessed May. 2, 2014.
URL `http://buildingsdatabook.eren.doe.gov`

[2] R. Brown, Us building-sector energy efficiency potential, Lawrence Berkeley National Laboratory.

[3] T. X. Nghiem, G. J. Pappas, Receding-horizon supervisory control of green buildings, in: American Control Conference (ACC), IEEE, 2011, pp. 4416–4421.

[4] E. M. Burger, S. J. Moura, Generation following with thermostatically controlled loads via alternating direction method of multipliers sharing algorithm,

Electric Power Systems Research 146 (2017) 141–160. doi:10.1016/j.epsr.2016.12.001.
URL http://escholarship.org/uc/item/2m5333xx

[5] D. S. Callaway, Tapping the energy storage potential in electric loads to deliver load following and regulation, with application to wind energy, Energy Conversion and Management 50 (5) (2009) 1389–1400.

[6] M. Maasoumy, C. Rosenberg, A. Sangiovanni-Vincentelli, D. S. Callaway, Model predictive control approach to online computation of demand-side flexibility of commercial buildings HVAC systems for supply following, in: American Control Conference (ACC), Portland, Oregon, 2014, pp. 1082–1089.

[7] J. L. Mathieu, S. Koch, D. S. Callaway, State estimation and control of electric loads to manage real-time energy imbalance, Power Systems, IEEE Transactions on 28 (1) (2013) 430–440.

[8] A. Kelman, F. Borrelli, Bilinear model predictive control of a HVAC system using sequential quadratic programming, IFAC Proceedings Volumes 44 (1) (2011) 9869–9874.

[9] A. Aswani, N. Master, J. Taneja, A. Krioukov, D. Culler, C. Tomlin, Energy-efficient building HVAC control using hybrid system LBMPC, arXiv preprint arXiv:1204.4717.

[10] E. M. Burger, S. J. Moura, Recursive parameter estimation of thermostatically controlled loads via unscented Kalman filter, Sustainable Energy, Grids and Networks 8 (2016) 12–25. doi:10.1016/j.segan.2016.09.001.
URL http://escholarship.org/uc/item/7t453713

[11] A. Aswani, N. Master, J. Taneja, V. Smith, A. Krioukov, D. Culler, C. Tomlin, Identifying models of HVAC systems using semiparametric regression, in: American Control Conference (ACC), IEEE, 2012, pp. 3675–3680.

[12] A. Handbook-Fundamentals, American society of heating, refrigerating and air-conditioning engineers, Inc., NE Atlanta, GA 30329.

[13] Y. Lin, T. Middelkoop, P. Barooah, Issues in identification of control-oriented thermal models of zones in multi-zone buildings, in: Decision and Control (CDC), 51st Annual Conference on, IEEE, 2012, pp. 6932–6937.

[14] C. Agbi, Z. Song, B. Krogh, Parameter identifiability for multi-zone building models, in: Decision and Control (CDC), 51st Annual Conference on, IEEE, 2012, pp. 6951–6956.

[15] P. Radecki, B. Hencey, Online building thermal parameter estimation via unscented Kalman filtering, in: American Control Conference (ACC), IEEE, 2012, pp. 3056–3062.

[16] S. Ihara, F. C. Schweppe, Physically based modeling of cold load pickup, Power Apparatus and Systems, IEEE Transactions on 100 (9) (1981) 4142–4250.

[17] R. E. Mortensen, K. P. Haggerty, A stochastic computer model for heating and cooling loads., Power Systems, IEEE Transactions on 3 (3) (1998) 1213–1219.

[18] Weather Underground Web Service and API.
URL wunderground.com/weather/api/