Washington State University
Electrical Engineering & Computer Science Department
Ninad Kiran Gaikwad & Dr.Anamika Dubey

Project: Building Modeling

Contents

1 Abstract

2 Introduction

3 ANN Based Black-Box Building Modeling

3.1 Model Development

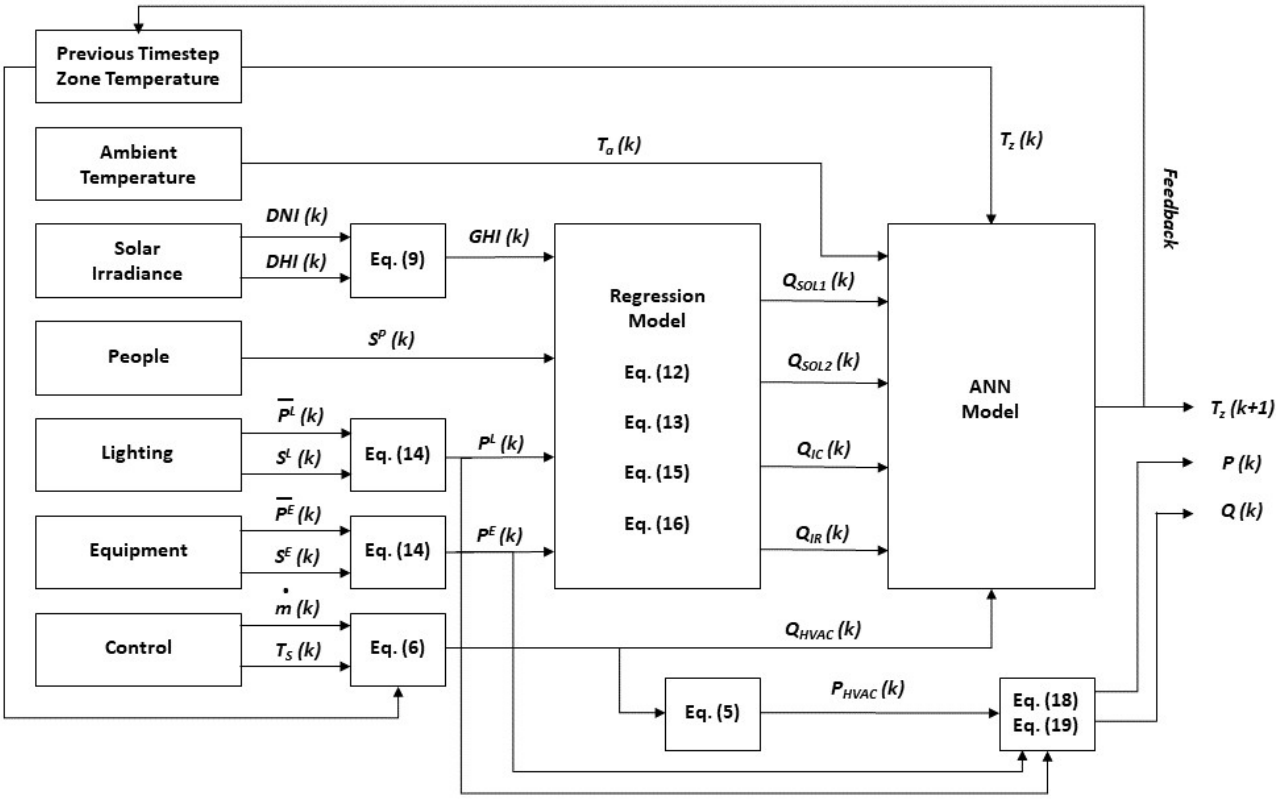


Figure 1: Single-Zone Building Model Schematic.

The Fig. 1 illustrates the schematic of the model developed for a single-zone building (zone or more precisely a thermal zone in a building is a volume within the building whose temperature can be controlled i.e. it has a thermostat associated with it). As we can see the developed model is discrete-time (with k as the time index), and it consists of seven inputs and three outputs. The first input is the feedback current time step zone temperature as it is the state of the system (further explanation will follow). The second and third inputs are the ambient temperature and solar irradiance respectively as they affect the temperature within the zone. Input four is the schedule (ratio of actual value to the rated value) of the people i.e. the number of people currently present within the zone, this is required as people give off heat which affects the zone temperature. Inputs five and six are the schedules and rated power of the lighting and other equipment (electric-based, gas-based, hot water-based, steam-based equipment if any) within the zone respectively, as they too convert a part of their consumed power into heat which affects the zone air temperature. The seventh and last input to the model is the control input which is the heating/cooling heat supplied to the zone by the HVAC system to maintain the zone air temperature within a certain range (depending on the operation policy of a building). The three outputs of the model are the next time step zone air temperature (keeps the

thermal dynamic simulation moving forward in time), real and reactive power consumption of the whole building (to be used in interfacing the building with a quasi-static power distribution system simulator). The details of this model for a general multi-zone building are discussed in the following sub-sections.

3.1.1 Multi-Zone Building Thermal Model Structure

The thermal model of a building in its most complex form is a system of heat PDEs (partial differential equations) in 4-D space-time. However, at the resolution of analyzing and controlling HVAC energy usage, it can be modeled as a system of ODEs (ordinary differential equations) in 1-D time. These ODEs take the shape of the well-known RC-Network thermal model which can consist of an arbitrary number of state variables (for a single-zone) depending on the level of modeling detail required. Based on the conclusions from [1], we have decided to go along with a second-order RC-Network model described as follows;

$$\begin{aligned} C_{z_i}\dot{T}_{z_i} =& \sum_{j}^{A_i} \frac{1}{R_{z_{ij}}}(T_{z_j} - T_{z_i}) + \frac{1}{R_{zw_i}}(T_{w_i} - T_{z_i}) + \\ & \frac{1}{R_{za}}(T_a - T_{z_i}) + (Q_{IC_i} - Q_{HVAC_i}) + A_{SOL_i}Q_{SOL1_i}, \end{aligned} \tag{1}$$

$$\begin{aligned} C_{w_i}\dot{T}_{w_i} =& \sum_{j}^{A_i} \frac{1}{R_{w_{ij}}}(T_{w_j} - T_{w_i}) + \frac{1}{R_{zw_i}}(T_{z_i} - T_{w_i}) + \\ & \frac{1}{R_{wa}}(T_a - T_{w_i}) + Q_{IR_i} + B_{SOL_i}Q_{SOL2_i}. \end{aligned} \tag{2}$$

Where $i \in I = \{1, \ldots, n\}$ is the i^{th} zone of the multi-zone building and I is the set of all n zones. Also, $A_i = \{j \in I : j \text{ is adjacent to zone } i\}$. Now, T_{z_i} and T_{w_i} are the zone air temperature and temperature of the surface (a fictitious state approximating all the surfaces of the zone) respectively, these are the states of the i^{th} zone.Q_{HVAC_i} is the heating/cooling supplied by the HVAC system, this is the control input i^{th} zone. T_a, Q_{SOL1_i}, Q_{SOL2_i}, Q_{IC_i}, and Q_{IR_i} are the ambient temperature, heat supplied to zone air from solar irradiance, heat supplied to the zone surface from solar irradiance, convective heat supplied to zone air by people and equipment within the zone, and radiant heat supplied to zone surface from people and equipment within the zone respectively; these for the disturbance to the i^{th} zone. C_{z_i}, C_{w_i}, R_{zij}, R_{wij}, R_{zwi}, R_{za}, and R_{wa} are thermal capacitances and resistances of the i^{th} zone; these along with A_{SOL_i} and B_{SOL_i} are the parameters of the i_{th} zone (in the interest of page limit we do not describe them in full). These parameters (depend on construction material and building geometry) are known as apriori in white-box modeling and are estimated using input-output data in grey-box modeling. As we are developing a discrete-time model, the continuous time building thermal model given in (1) and (2) can be easily discretized based on a method of choice to a form as follows;

$$\underline{x}(k+1) = A_d\underline{x}(k) + B_{d_1}\underline{u}(k) + B_{d_2}\underline{w}(k), \tag{3}$$

$$\underline{y}(k) = C\underline{x}(k). \tag{4}$$

Where, $\underline{x} = [T_{z_1}, \ldots, T_{z_n}, T_{w_1}, \ldots, T_{w_n}]^T$ are the states of the model. We can divide $\underline{X}$ as $\underline{x}_1 = [T_{z_1}, \ldots, T_{z_n}]^T$ (measurable state) and $\underline{x}_2 = [T_{w_1}, \ldots, T_{w_n}]^T$ (unmeasurable state). $\bar{u} = [Q_{HVAC_1}, \ldots, Q_{HVAC_n}]^T$

is the control input to the model. $\bar{w} = [T_a, Q_{SOL1_1}, \ldots, Q_{SOL1_n}, Q_{SOL2_1}, \ldots, Q_{SOL2_n}, Q_{IC_1}, \ldots, Q_{IC_n}, Q_{IR_1}, \ldots,$ is the disturbance to the model. We can divide $\bar{w}$ as $\bar{w}_1 = T_a$ (independent of zone) and $\bar{w}_1 = [Q_{SOL1_1}, \ldots, Q_{SOL1_n}, Q_{SOL2_1}, \ldots, Q_{SOL2_n}, Q_{IC_1}, \ldots, Q_{IC_n}, Q_{IR_1}, \ldots, Q_{IR_n}]^T$ (dependent on zone). $\underline{y} = [T_{z_1}, \ldots, T_{z_n}]^T$ is the output of the model, and we can see that $\underline{y} = \underline{x}$. $A_d \in \mathbb{R}^{2n \times 2n}$, $B_{d_1} \in \mathbb{R}^{2n \times n}$, and $B_{d_2} \in \mathbb{R}^{2n \times 4n+1}$ are the system matrix and input matrices respectively. $C = [C_1, C_2] \in \mathbb{R}^{1 \times 2n}$ where $C_1 = [1, \ldots, 1] \in \mathbb{R}^{1 \times n}$, and $C_2 = [0, \ldots, 0] \in \mathbb{R}^{1 \times n}$.

3.1.2 HVAC Power Consumption Model

Power consumed by the HVAC system for a particular zone (P_{HVAC_i}) is usually not measurable as all the zones are supplied by a single HVAC system. Hence, the total building HVAC power consumption (P_{HVAC}) is measurable; and for this preliminary work we assume that it is linearly dependent on the heating/cooling power supplied to a particular zone i as follows;

$$P_{HVAC_i}(k) = \frac{P_{HVAC}(k)}{n} = f_{HVAC_i}(Q_{HVAC_i}(k)), \tag{5}$$

$$Q_{HVAC_i}(k) = C_a \dot{m}_i(k)(T_{z_i}(k) - T_{s_i}(k)). \tag{6}$$

Where, C_a is the specific thermal capacity of air (kJ/kg), $\dot{m}_i$ is the mass flow rate of supply air from the HVAC system to the i^{th} zone, and T_{s_i} is the temperature of that supply air. It is to be noted that the proportion of P_{HVAC} required for each zone can vary with their volume and floor area however, we do not consider that effect in this work. Moreover, the computation model for Q_{HVAC_i} in (6) is specifically for commercial buildings, and can be modified for residential buildings to $Q_{HVAC}(k) = u_{HVAC}(k)(COP \times P_{HVAC}^{Rated})$, where $u_{HVAC} \in \{1, 0\}$ is the on-off control command to the HVAC and COP is the coefficient of performance of the HVAC.

3.1.3 Heat Source Models

We assume the relationship between GHI and both Q_{SOL1_i} and Q_{SOL2_i} to be linear and given as follows;

$$Q_{SOL1_i}(k) = f_{SOL1_i}(GHI(k)), \tag{7}$$

$$Q_{SOL2_i}(k) = f_{SOL2_i}(GHI(k)), \tag{8}$$

$$GHI(k) = DNI(k)|cos(\delta(k))| + DHI(k). \tag{9}$$

Where GHI is the global horizontal irradiance, DNI is the direct normal irradiance, DHI is the diffused horizontal irradiance, and δ is the sun's altitude angle. People, lighting and equipment with the zone contribute to Q_{IC_i} and Q_{IR_i} and we can write the following;

$$Q_{IC_i}(k) = \sum_{m}^{B} Q_{IC_i}^m(k) \tag{10}$$

$$Q_{IR_i}(k) = \sum_{m}^{B} Q_{IR_i}^m(k). \tag{11}$$

where, $m \in B = \{P, L, E\}$, P means people, L means lighting, and E means all other energy consuming equipment ((10) and (11) are self-explanatory). Now, we assume that $Q_{IC_i}^m$ and $Q_{IR_i}^m$

are linear functions of rated power ($\bar{P}_i^m$) and given schedule of operation (S_i^m); they are described as follows;

$$Q_{IC_i}^m(k) = f_{IC_i}^m(P_i^m(k)),\ m \in B - \{P\} \tag{12}$$
$$Q_{IR_i}^m(k) = f_{IR_i}^m(P_i^m(k)),\ m \in B - \{P\}, \tag{13}$$
$$P_i^m(k) = \bar{P}_i^m(k) \times S_i^m(k) \tag{14}$$
$$Q_{IC_i}^P(k) = f_{IC_i}^P(S_i^P(k)), \tag{15}$$
$$Q_{IR_i}^P(k) = f_{IR_i}^P(S_i^P(k)), \tag{16}$$

Where P_i^m is the actual power consumption based on the given schedule. The equations (12) - (16) are self-explanatory.

3.1.4 Multi-Zone Building Complete Model based on ANN

We utilize the model described in (21) and (22) for understanding a physics-based input-output model of the multi-zone building thermal dynamics, through which we design the input and output of the ANN model we want to develop as follows;

$$\underline{x}_1(k+1) = f(\underline{x}_1(k), \underline{u}(k), \underline{w}(k); \theta), \tag{17}$$
$$P(k) = \sum_{i}^{I} P_{HVAC_i}(k) + \sum_{m}^{B-\{P\}} \sum_{i}^{I} P_i^m(k), \tag{18}$$
$$Q(k) = P(k) tan(cos^{-1}(p.f(k))). \tag{19}$$

(17) tries to learn the input-output relationship described in (21) and (22) using an ANN parameterized by weights θ. We should observe that the ANN model does not require the unmeasurable state $\underline{x}_2$ to compute $\underline{x}_1$, and it is assumed that it internally computes $\underline{x}_2$ as hidden latent variables. (18) and (19) compute the real and reactive power consumed by the building, where $p.f$ is the power factor of the building. Hence, (17), (18), and (19) completely describe the thermal dynamics and power consumption of a multi-zone building.

3.2 Zone Aggregation Method

From [1] we know that a single-zone thermal zone can be effectively represented as a second-order ODE. However, we do not have a measure of what is the optimal number of zones that should be used to effectively represent a multi-zone building. On the other hand, it is clear that the computational resources required to represent a multi-zone building with all its zones increase with the increase in the number of zones. We propose that there is a possibility of representing a multi-zone building with fewer zones (these are constructed by aggregations of the original zones) effectively for the purposes mentioned in Section 2.

Let us consider 2^I to be the power set of the zones present in I. Then, $\exists\ \tilde{I} \subset 2^I$ and $\tilde{I} = \{\tilde{Z} \in 2^I : \tilde{Z} \neq \emptyset, \forall\ i \in I\ \exists!\ \tilde{Z}$ s.t. $i \in \tilde{Z}\}$, $\tilde{I}$ is the set of how the original zones within a building are aggregated. Now, let us consider an arbitrary $\tilde{I} = \{\tilde{Z}_1, \ldots, \tilde{Z}_p\}$ s.t. $p \leq n$ where p is the number of the aggregated zones. For an arbitrary $\tilde{Z}_i \in \tilde{I},\ i = \{1, \ldots, p\}$, we can aggregate the states, control

inputs, and disturbances of the original zones as follows;

$$\tilde{x}_i = \frac{1}{|\tilde{Z}_i|}\sum_{j}^{\tilde{Z}_i} x_j, \tilde{u}_i = \frac{1}{|\tilde{Z}_i|}\sum_{j}^{\tilde{Z}_i} u_j, \tilde{w}_i = \frac{1}{|\tilde{Z}_i|}\sum_{j}^{\tilde{Z}_i} w_j. \tag{20}$$

Where, $\tilde{x}_i$, $\tilde{u}_i$, $\tilde{w}_i$, x_j, u_j, and w_j are components of the vectors $\underline{\tilde{x}}_i$, $\underline{\tilde{u}}_i$, $\underline{\tilde{w}}_i$, $\underline{x}_j$, $\underline{u}_j$, and $\underline{w}_j$ respectively. We can now have a discrete-time system for the aggregated zones as follows;

$$\underline{\tilde{x}}(k+1) = \tilde{A}_d\underline{\tilde{x}}(k) + \tilde{B}_{d_1}\underline{\tilde{u}}(k) + \tilde{B}_{d_2}\underline{\tilde{w}}(k), \tag{21}$$

$$\underline{\tilde{y}}(k) = C\underline{\tilde{x}}(k). \tag{22}$$

Where, $\underline{\tilde{x}}$, $\underline{\tilde{u}}$, $\underline{\tilde{w}}$, and their sub-divisions $\underline{\tilde{x}}_1$, $\underline{\tilde{x}}_1$, $\underline{\tilde{w}}_2$, and $\underline{\tilde{w}}_2$ have the exact same description as, in Section 3.1, the only difference being these quantities describe the aggregated zones. We can now develop an ANN model as described earlier for learning the thermal behavior of the aggregated zones as follows;

$$\underline{\tilde{x}}_1(k+1) = \tilde{f}(\underline{\tilde{x}}_1(k), \underline{\tilde{u}}(k), \underline{\tilde{w}}(k); \tilde{\theta}). \tag{23}$$

Now (23), (18), and (19) form the complete model of the building behavior in terms of both thermal dynamics and power usage.

3.3 Experiment Study Setup

The experiment study set up is explained in the following sub-sections.

3.3.1 Data Generation

Pacific Northwest National Laboratory (PNNL) has developed 16 prototype commercial building models and 2 prototype residential building models, whose EnergyPlus [2] input files (IDF) are available at [3]. Along with the building models PNNL also has TMY3 (typical meteorological year) weather files for 19 different climate zones within the US. For this work, we have used the five-zone small office building model along with the TMY3 weather file for Seattle (mixed marine climate zone). The data was generated by providing these two files (building and weather files) to EnergyPlus (a building simulation software) and running the simulation at a 5-minute time-step. All the necessary data for training both regression and ANN models as described in Section 3.1 was generated from this process. For both regression and ANN models the entire year simulated data obtained from EnergyPlus was divided into a 40-week training set and a 12-week testing set (77/23 % split).

3.3.2 Zone Aggregation

The small office building has 5 zones: Core, Perimeter-1, Perimeter-2, Perimeter-3, and Perimeter-4, where the Core zone is surrounded by the other Perimeter zones. The original zone structure is used as the unaggregated case (referred to as 5Z). We have aggregated these 5 zones in three different ways: 2Z (aggregation of all the Perimeter zones and the original Core zone forms the two zones), 3Z (aggregation of Perimeter-1 with Perimeter-2, Perimeter-3 with Perimeter-4, and the original Core zone forms the three zones), and 1Z (aggregation of all the five original zones forms a single zone).

3.3.3 ANN Architectures

From the (23) it is clear that the input space increases linearly with an increase in the number of aggregated zones. Moreover, in our preliminary experiments, we found that the temperature output from an ANN based on (23) for more than one aggregated zone was following the trend but had large oscillations (more work needs to be done in order to get smooth results). Now in order to keep the input space small we have come up with two other ANN architectures: Independent and Dependent. In the independent architecture, each aggregated zone is treated independently (zones do not affect each other's temperatures) of other zones and we train an ANN for each aggregated zone (the temperature, control input, and disturbances related to the respective aggregated zone are the only inputs considered to train the ANN). In the dependent architecture (zones affect each other's temperature), we train a single ANN for all the aggregated zones such that the input consists of the individual temperatures and control inputs of all the aggregated zones, while the disturbances are aggregated across all the original zones, thus making the input space smaller as compared to that of (23).

3.3.4 Computation

The system used had EnergyPlus (version 9.0.1) has been used to generate the raw data from the building and weather files obtained from PNNL repository. We have used Python (version 3.10.6) programming language for the entire work, the important packages used were: *opyplus* (for running/-manipulating Energyplus from within Python), *numpy+pandas* (for data processing), and *tensorflow 2.0* (for developing regression and ANN models). The computation machine utilized was a desktop Windows computer with 32GB RAM and a 3.0 GHz $\times$ 18 CPU and one Nvidia RTX A4000 16Gb GPU.The raw data from EnergyPlus was generated at a resolution of 5-minutes for one year, with a training/testing split of 77/23 we had 80,941/24,177 training and testing points respectively, hence the time required for training any model in this work was never more than 15 minutes (for our max number of epochs).

3.4 Training Parameters

For training the *loss function* used is Mean Squared Error for regression models, but an additional l_2 regularizer is utilized for ANN models. The *optimizer* used is SGD (Stochastic Gradient Descent) *learning rate* = 0.001 for both regressions and ANN models. Moreover, the *batch size* is 100 and 25, and the *epochs* are 10 and 100 for the regression and ANN models respectively. *Normalization* of input is done for both regression and ANN models, all of the ANN models have been designed with a single *hidden layer* of 100 neurons with RelU *activation function*.

3.5 Results and Discussion

The important results are presented and discussed in the following sub-sections;

3.5.1 Regression Model Results

Fig. 2 reflects the regression model between Q_{SOL1} and GHI. From the plot, we can see that it is highly non-linear to GHI. This is because it requires additional information, such as building geometry and position. However, we do not consider those in our research; consequently, this model

is weak. Therefore, the average absolute error shown in Table 1 for Q_{SOL} is high. On the other hand, the Q_{IR} and Q_{IC} are linearly dependent on the schedule of people, lights, and equipment. Therefore the regression model predicts very well in these cases and reflects linearity. Hence the average absolute errors shown in Table 1 for both of those are also low. Fig. 3 shows the scatter plot between Q_{IR}^{P} and the schedule of people as a reference.

Table 1: Average absolute error for regression model

Regression model	**Average Absolute Error (%)**
Q_{SOL}	27.03
Q_{IC}	1.56
Q_{IR}	2.18

3.5.2 T_z Prediction

The comparison between the actual temperature and the temperature predicted by the ANN model for various aggregate zone models is portrayed in Fig. 10. The results are presented for one-week data to improve the readability of the results. Table 2 contains the average absolute error of the T_z prediction for every zone model. As both the independent and dependent architectures of the 1Z model are identical, only one set of predicted temperature data is plotted Fig. 6. It is evident that the predicted temperature is learned through the ANN model with an error prediction of 2.78%. For 2Z models shown in Fig. 7, the predicted temperature is shown for aggregated perimeter zones. In this case, it can be seen that the efficacy of the dependent architecture is better than the independent architecture. This is more apparent in both 3Z and 5Z model shown in Fig. 8 and Fig. 9 respectively where the predicted temperature under the dependent architecture is more accurate than the independent one. Here, the combination of Perimeter-3 and Perimeter-4 is selected for 3Z plot and Perimeter-4 is selected for 5Z plot. The average absolute error displayed in Table 2 also indicates that the dependent architecture outperforms the independent architecture. This is due to the fact that the dependent architecture takes into consideration the interdependence between zone temperatures and control inputs. On the contrary, the independent architecture does not consider the zone inter-dependency to each other and that is not the practical case.

3.5.3 P_{HVAC} Prediction

The scatter plot shown in Fig. 4 reflects the non-linearity between P_{HVAC} and Q_{HVAC}. As Q_{HVAC} depends on T_s, T_z and $\dot{m}$ as per (6) and , hence P_{HVAC} is a nonlinear function. However, we are approximating it to be linear and hence it fails to capture the non-linearity and makes the model weaker. The average absolute error for P_{HVAC} prediction is displayed in Table 3.

Table 2: Average absolute error for T_z prediction

Zone Type	**Average Absolute Error (%)**		
	Independent	***Dependent***	***Dependent 2***
1Z	3.28	NA	NA
2Z	2.89	2.86	3.59
3Z	6.13	0.54	16.77
5Z	3.61	0.63	2.55

Table 3: Average absolute error for P_{HVAC} prediction

Zone Type	**Average Absolute Error (%)**		
	Independent	***Dependent***	***Dependent 2***
1Z	1.05	NA	NA
2Z	1.08	1.78	10.47
3Z	4.55	2.51	8.01
5Z	0.30	2.40	5.60

3.6 Conclusion

In this preliminary work, we have developed a principled way of utilizing the structure of a simple RC-Network-based building thermal model to develop ANN-based building thermal models. We not only develop an ANN model but also the power consumption model (both active and reactive power) based on linear regression for the building to provide a complete (controllable and encompassing effects of disturbances) building model in a form that can be integrated into a co-simulation framework along with a power distribution simulator. Our building model's input-output mimics that of EnergyPlus, where we utilize the basic inputs of weather, rated power/operation schedule of equipment, schedule of people, and HVAC control (we need control to be computed externally, while EnergyPlus has an in-built base controller for temperature control) to develop regression-based models which compute the intermediate inputs of the different heating/cooling supplied to the building by different sources. These intermediate inputs along with ambient temperature and the fed-back state (zone temperature) form the inputs to the ANN-based building thermal model. For our regression models, our assumptions of linear models for Q_{IC}, and Q_{IR} are observed to be correct, while the linear model assumption for Q_{SOL1}, Q_{SOL2} and P_{HVAC} are incorrect and they seem to affect the model performance to some degree. Between the independent and dependent ANN architectures, the performance of the latter is better as expected since it takes into account the effect of different zone temperatures and control inputs on each other.

For future work, we would first like to solve the oscillation in the output problem for a full ANN model based on (17). Moreover, in this work, we did not force variation in input to the EnergyPlus model and hence the raw data did not have the required variance which we think has been the biggest bottleneck for learning. Moving forward we would like to design an optimal input with increased variance to be given to EnergyPlus for producing rich raw data. We would be trying out some

variants of RNNs (Recurrent Neural Networks) as opposed to MLPs that we have used in this work. Moreover, the non-linear relationships of Q_{SOL1} and Q_{SOL2} with GHI, and that of P_{HVAC} with $\dot{m}$, T_z and T_s are needed to be studied further to develop more accurate models. Finally, we would like to test the complete building model in co-simulation with OpenDSS (power distribution simulator).

4 Building Thermal Models based on RC-Network Structure

We will discuss RC-Network based thermal models for commercial and residential buildings.

4.1 2-State Zone Model Development

A generalized two-state thermal model for a zone is shown in eq.(25) and eq.(**??**). Here we have assumed that the convective part of the internal heat affects the zone air temperature, while the radiant part of the internal heat affects the fictitious wall temperature.

$$\dot{T}_z = \frac{1}{R_{za}C_z}(T_a - T_z) + \frac{1}{C_z}(Q_{INT_C} - Q_{HVAC}) + \frac{1}{R_{zw}C_z}(T_w - T_z) + \frac{A_{SOL}}{C_z}Q_{SOL1}\ , \tag{24}$$

$$\dot{T}_w = \frac{1}{R_{zw}C_w}(T_z - T_w) + \frac{1}{R_{za}C_w}(T_a - T_w) + \frac{Q_{INT_R}}{C_w} + \frac{B_{SOL}}{C_w}Q_{SOL2}\ . \tag{25}$$

Where,
T_z : Zone Air Temperature
T_w : Zone Internal Surface Temperature
T_a : Site Outdoor Air Dry bulb Temperature
C_z : Capacitance of Zone Air
C_w : Capacitance of Surfaces and Internal Mass
R_{za} : Resistance between Outside and Zone
R_{zw} : Resistance between Zone Air and Internal Surfaces and Internal Mass
R_{wa} : Resistance between Internal Mass and Outside Air
Q_{INTC} : Internal Heat Gain from People, Equipment, Lights etc. [Convective]
Q_{INTR} : Internal Heat Gain from People, Equipment, Lights etc. [Radiant]
Q_{AC} : Heating and Cooling Gain from HVAC
Q_{SOL1}, Q_{SOL2} : Heat Gain from Solar Radiation
A_{SOL}, B_{SOL} : Solar Radiation Gains

4.2 4-State Single-Family House Model Development

[4] have come up with a four-state thermal model for a single-family house, which is given as follows;

$$\dot{T}_{wall}(t) = \frac{T_{sol,w}(t) - T_{wall}(t)}{C_w R_w/2} - \frac{T_{wall}(t) - T_{ave}(t)}{C_w R_w/2} \tag{26}$$

$$\dot{T}_{ave}(t) = \frac{T_{wall}(t) - T_{ave}(t)}{C_{in} R_w/2} + \frac{T_{attic}(t) - T_{ave}(t)}{C_{in} R_{attic}} + \frac{T_{im}(t) - T_{ave}(t)}{C_{in} R_{im}} + \frac{T_{am}(t) - T_{ave}(t)}{C_{in} R_{win}} + C_1 Q_{IHL} + C_2 Q_{AC} + Q_{venti} + Q_{infil} \tag{27}$$

$$\dot{T}_{attic}(t) = \frac{T_{sol,r}(t) - T_{attic}(t)}{C_{attic} R_{roof}} - \frac{T_{attic}(t) - T_{ave}(t)}{C_{attic} R_{attic}} \tag{28}$$

$$\dot{T}_{im}(t) = \frac{T_{im}(t) - T_{ave}(t)}{C_{im} R_{im}} + C_3 Q_{solar} \tag{29}$$

Where,
T_{wall} : Wall Surface Temperature
T_{ave} : House Average Air Temperature
T_{attic} : Attic Air Temperature
T_{im} : Internal Mass Temperature
C_w : Thermal Capacitance of Wall
C_{in} : Thermal Capacitance of Indoor Air
C_{attic} : Thermal Capacitance of Air in Attic
C_{im} : Thermal Capacitance of Internal Mass
R_w : Thermal Resistance of Walls
R_{attic} : Thermal Resistance of Attic
R_{roof} : Thermal Resistance of Roof
R_{im} : Thermal Resistance of Internal Mass
R_{win} : Thermal Resistance of Windows
Q_{IHL} : Internal Heat Gain from People and Equipment
Q_{AC} : Heating and Cooling gain from AC
Q_{venti} : Heat gain from Ventilation
Q_{infil} : Heat Gain from Infiltration
Q_{solar} : Heat Gain from Solar Radiation
C_1, C_2, C_3 : Heat Gain Fractions

4.3 Single and Multi-Zone Building Thermal Models

Let us consider a commercial building with N_z zones. For a single zone a simplified thermal dynamics model can be given as follows in (31) and (32);

$$\dot{T}_{z_i} = \frac{1}{R_{za_i}C_{z_i}}(T_a - T_{z_i}) + \frac{1}{C_{z_i}}(Q_{INT_{C_i}} - Q_{HVAC_i}) + \frac{1}{R_{zw_i}C_{z_i}}(T_{w_i} - T_{z_i}) + \frac{A_{SOL_i}}{C_{z_i}}Q_{SOL1_i} \tag{30}$$

$$\dot{T}_{w_i} = \frac{1}{R_{zw_i}C_{w_i}}(T_{z_i} - T_{w_i}) + \frac{1}{R_{za_i}C_{w_i}}(T_a - T_{w_i}) + \frac{Q_{INT_{R_i}}}{C_{w_i}} + \frac{B_{SOL_i}}{C_{w_i}}Q_{SOL2_i} \tag{31}$$

Where T_{z_i}, T_{w_i}, C_{z_i} and R_i are the i^{th} zone's air temperature, i^{th} zone's fictitious wall temperature belonging to the i^{th} zone, thermal capacitance of the i^{th} zone and thermal resistance between the i^{th} surface and the i^{th} zone air.

From (31) and (32) we can see that the i^{th} zone's air temperature dynamics is not dependent on the air temperatures of other zones. Hence, this model assumes the zone to be thermally insulated from the other zones. However, this is an assumption and in reality a zone's temperature depends on the temperatures of its adjacent zones. We can construct a multi-zone model taking into account this effect of adjacent zones on the thermal dynamics of a zone as shown in (33) and (34);

$$\dot{T}_{z_i} = \sum_{j=1\ i\neq j}^{N_z} \frac{(T_{z_j} - T_{z_i})}{R_{z_{ij}}} + \frac{1}{R_{za_i}C_{z_i}}(T_a - T_{z_i}) + \frac{1}{C_{z_i}}(Q_{INT_{C_i}} - Q_{HVAC_i}) + \frac{1}{R_{zw_i}C_{z_i}}(T_{w_i} - T_{z_i}) + \frac{A_{SOL_i}}{C_{z_i}}Q_{SOL1_i} \tag{32}$$

$$\dot{T}_{w_i} = \sum_{j=1\ i\neq j}^{N_z} \frac{(T_{w_j} - T_{w_i})}{R_{w_{ij}}} + \frac{1}{R_{zw_i}C_{w_i}}(T_{z_i} - T_{w_i}) + \frac{1}{R_{za_i}C_{w_i}}(T_a - T_{w_i}) + \frac{Q_{INT_{R_i}}}{C_{w_i}} + \frac{B_{SOL_i}}{C_{w_i}}Q_{SOL2_i} \tag{33}$$

Where T_{z_j} and $R_{z_{ij}}$ are the j^{th} zone's air temperature and the thermal resistance between the j^{th} and the i^{th} zone respectively. As we can see there will be N_z ODEs similar to (33) and (34)for each zone, and together they will constitute a multi-zone thermal dynamics model of the commercial building.

Finally, there is an important point to be noted, the heating/cooling gains: $Q_{INT_{C_i}}$, $Q_{INT_{R_i}}$, Q_{SOL1i}, Q_{SOL2_i}, and Q_{HVAC_i} in (33) and (34) ideally should be data that we obtain, or there is a parametric form which is already trained from relevant input/output relationships for them, in both these cases the heating/cooling gains appear as scalar values in the thermal model not affecting its structure in terms of the number of parameters to be estimated and the linear structure of the thermal ODEs. On the other hand, in most real-world applications neither data nor a trained parametric form of these heating/cooling gains is available. In such a situation we can develop parametric forms with relevant proxy inputs such that they have some causal relationship with these heating/cooling gains, and the training of the parameters of these parametric forms has to be done within the Bayesian

framework. The parametric forms for the heating/cooling gains can be represented as;

$$Q_{INT_{C_i}} = f_{INT_{C_i}}(x_{INT_{C_i}} \mid \theta_{INT_{C_i}}) \,, \tag{34}$$

$$Q_{INT_{R_i}} = f_{INT_{R_i}}(x_{INT_{R_i}} \mid \theta_{INT_{C_i}}) \,, \tag{35}$$

$$Q_{SOL1_i} = f_{SOL1_i}(x_{SOL1_i} \mid \theta_{SOL1_i}) \,, \tag{36}$$

$$Q_{SOL2_i} = f_{SOL2_i}(x_{SOL2_i} \mid \theta_{SOL2_i}) \,, \tag{37}$$

$$Q_{HVAC_i} = f_{HVAC_i}(x_{HVAC_i} \mid \theta_{HVAC_i}) \,. \tag{38}$$

The structure of the model changes at least in the number of parameters to be estimated (as each parametric model for the heating/cooling gains comes with its own set of parameters), while the linear structure of the thermal ODEs is converted to a nonlinear structure with even if one of the parametric heating/cooling models is nonlinear.

4.4 ODE Discretization Methods

To be studied ...

5 Building Thermal Model - Grey Box ODE Model Estimation Techniques

This section will describe the different parameter estimation techniques for RC-Thermal Network Models.

5.1 Nonlinear Optimization based Least Squares Estimation

We know that through the application of 1-D heat flow concepts, we can develop LTI state-space models of a building zone thermal dynamics of varying degrees of detail, however, The problem is computing the parameters of this model. In the white-box modeling approach these parameters are calculated through thorough experimentation for each material used in the building construction and a repository of these parameters is used in tools and software employing the white-box approach. There arise cases where these parameters have not been computed, or a simpler (reduced-order) model of the thermal dynamics of a building or its zones is required (generally for control purposes) in which the parameters are not known. In such situations these parameters are to be computed optimally from data gathered through sensors installed in the building, this process is called parameter estimation (also called system identification).

Let us consider the parameter estimation problem for the thermal dynamics system described by a linear ODE representing an RC-Network structure. Let the parameter vector which needs to be estimated for this system be $\theta = [R_{con}\ R_{conv}\ C_z\ C_w]$. Also, we have N discrete data samples collected at a sampling period of t_s consisting of the zone temperature (T_z), outside air temperature (T_o), solar heat gain ($\dot{Q}_{solar}$), HVAC heat gain ($\dot{Q}_{hvac}$) and internal heat gain (Q_{ihg}), hence we know $y(k)$ and $u(k)$ for all $k = 1, \cdot, N$. We also know the initial state $x_0 = x(0) = [T_z(0)\ T_1(0)\ T_2(0)]^T$ Now the parameter estimation problem can be formulated as an optimization problem as follows;

$$\min_{\theta} \sum_{k=1}^{N} \left[y(k) - \tilde{y}(k)\right]^2, \tag{39}$$

subject to the following constraints:

$$\tilde{x}(k+1) = \tilde{x}(k) + t_s \left[A(\theta)\tilde{x}(k) + B(\theta)u(k)\right], \tag{40}$$

$$\tilde{y}(k) = C\tilde{x}(k) + Du(k), \tag{41}$$

$$\tilde{x}(k) \in \mathbb{X}, \tag{42}$$

$$\theta \in \Theta, \tag{43}$$

For the parameter estimation problem the cost function given in (39) minimizes the sum of squared error between the actual measured output y and output produced by the estimated system $\tilde{y}$. the equality constraint (40) is the discretized (Euler Forward) system state equation, while the equality constraint (41) is discretized system output equation. (42) and (43) constrains the states $\tilde{x}$ which are part of the decision variables of the optimization problem and the parameter vector θ respectively, to a physically meaningful set of values.

The above optimization problem is non-convex as the output vector y is non-linear with respect to the parameter vector θ, see (**??**). This optimization problem is usually solved using iterative

optimization algorithms like: Nonlinear Least-Squares, Steepest Descent, Gauss-Newton, or Levenberg–Marquardt. As the problem is non-convex, the global optimality of the solution is highly dependent on the initialization of the parameter vector θ.

One other important point to note is that the above method doesn't consider the noise due to process error or noise due to measurement error. This method believes that the model used is perfect as well as the measurements it is being trained on are error-free, which is not the case and this becomes a very naive simplification of the problem at hand. In the next section, we will discuss techniques developed to handle the process and measurement noises; hence, the setup of the problem will be based in a probabilistic setting leading us to a more complete formulation of the problem.

5.2 Bayesian Estimation Framwork

We will be following [5] for this section. Let us consider the simpler problem first of state estimation, we have a set of unobservable hidden states $x_{0:T}$ which are observed through the measurements $y_{1:T}$ as follows;

$$x_{0:T} = \{x_0, \ldots, x_T\} \,, \tag{44}$$

$$y_{1:T} = \{y_1, \ldots, y_T\} \,. \tag{45}$$

Now, we would like to compute the hidden states from the available measurements, in the Bayesian sense this is to compute the joint posterior distribution $p(x_{0:T} \mid y_{1:T})$ as follows;

$$p(x_{0:T} \mid y_{1:T}) = \frac{p(y_{1:T} \mid x_{0:T})\; p(x_{0:T})}{p(y_{1:T})}, \tag{46}$$

where $p(x_{0:T})$ is the prior distribution defined by the process dynamic model, $p(y_{1:T} \mid x_{0:T})$ is likelihood model defined through the measurement model, and $p(y_{1:T})$ is the normalization constant defined as;

$$p(y_{1:T}) = \int p(y_{1:T} \mid x_{0:T})\; p(x_{0:T})\; dx_{0:T}. \tag{47}$$

However, the full posterior formulation has the following challenges:

1. The full posterior distribution has to be recomputed entirely with every new incoming measurement.
2. As we can see as the number of time steps increases the dimensionality of the full posterior increases leading to increased computational complexity.

On the other hand, we can overcome the disadvantages of the full posterior if we make the following relaxations;

1. To utilize selected marginal distribution as opposed to the full posterior distribution.
2. To compute marginal distributions with relative ease we can restrict our models to be probabilistic Markov sequences.

Our state-measurement model will consist of the following:

1. $p(x_0)$: Initial distribution of state x_0 at time step $k = 0$.
2. $p(x_k \mid x_{k-1})$: Transition probability distribution which describes the state dynamics and its uncertainties.
3. $p(y_k \mid x_k)$: Measurement model distribution which describes the conditional probability of measurement given state.

Now, based on the above models we define the Bayesian Filtering, Prediction and Smoothing distributions as follows, which correspond to the selected marginal distributions mentioned before:

1. *Filtering Distribution*: This distribution computes the conditional distribution of state x_k given current and all the previous measurements $y_{1:k}$;

$$p(x_k \mid y_{1:k}) \,, \qquad k = \{1, \ldots, T\} \,. \tag{48}$$

2. *Prediction Distribution*: This distribution computes the conditional distribution of the future state x_{k+n} given current and all the previous measurements $y_{1:k}$;

$$p(x_{k+n} \mid y_{1:k}) \ , \qquad k = \{1, \dots, T\} \ , \qquad n = \{1, 2, \dots\} \ . \tag{49}$$

3. *Smoothing Distribution*: This distribution computes the conditional distribution of state x_k given current, all the previous, and future measurements $y_{1:T}$;

$$p(x_k \mid y_{1:T}) \ , \qquad k = \{1, \dots, T\} \ , \qquad T > k \ . \tag{50}$$

We will describe recursive algorithms to compute the above marginal distributions in the subsequent subsections. These algorithms form the basis for Bayesian Estimation techniques.

At this point, we can claim that we understand the general setup for the state estimation problem through Bayesian techniques. Here, we assume that we precisely know the parameters of the model for which we want to carry out state estimation. However, in most of the cases, we would not have the precise parameters of our model and we will have to estimate the parameters before we perform state estimation. In general, when the model parameters are unknown the problem gets more complicated as it becomes one where we have to estimate both the parameters and states jointly because of their interdependence. Once, we have some nominal parameters estimated we can then use them in our model to perform just state estimation.

Let us now formulate the Parameter estimation problem in the Bayesian setup. For the probabilistic model of the state and measurement process from before, we say the model is parameterized by the unknown parameter θ. Then in the Bayesian sense, the problem now becomes one of computing the joint posterior of the hidden states ($x_{0:T}$) and parameter (θ) given the measurements ($y_{1:T}$) given as follows;

$$p(x_{0:T} \ , \ \theta \mid y_{1:T}) = \frac{p(y_{1:T} \mid x_{0:T} \ , \ \theta) \ p(x_{0:T} \mid \theta) \ p(\theta)}{p(y_{1:T})}, \tag{51}$$

we can see that computing the above posterior distribution is more complicated than the previous one for the state estimation problem. Also, it has the same disadvantages as mentioned before. Therefore, we would like to work on a selected marginal distribution and restrict our model class to probabilistic Markov sequences. The Bayesian Filtering and Smoothing algorithms help us compute the following marginal and joint marginal distributions for the joint state-parameter estimation problem;

$$p(\theta \mid y_{1:T}) \propto p(\theta) \ \prod \ p(y_k \mid y_{1:k-1} \ , \ \theta) \ , \tag{52}$$

$$p(x_k \ , \ \theta \mid y_{1:T}) = p(x_k \mid y_{1:T} \ , \ \theta) \ p(\theta \mid y_{1:T}). \tag{53}$$

Where, $p(y_k \mid y_{1:k-1} \ , \ \theta)$ and $p(x_k \mid y_{1:T} \ , \ \theta)$ are by-products of the Bayesian Filtering algorithm, and the Bayesian Smoothing distribution for a fixed θ respectively. Also, the probabilistic models for joint state-parameter estimation are given as follows:

1. $p(\theta)$: Prior distribution on the parameters which describe the prior knowledge about the parameters.

2. $p(x_0 \mid \theta)$: Initial distribution of state x_0 at time step $k = 0$ given the parameters.

3. $p(x_k \mid x_{k-1}\ ,\ \theta)$: Transition probability distribution which describes the state dynamics and its uncertainties given the parameters.

4. $p(y_k \mid x_k\ ,\ \theta)$: Measurement model distribution which describes the conditional probability of measurement given state and parameters.

There are multiple methods to compute θ from the above forms which we will discuss in the subsequent sections after the discussion on Bayesian Filtering, Prediction, and Smoothing algorithms is complete.

Now, we discuss the general Bayesian Filtering Framework with known parameters θ;

Bayesian Filtering Equations: The recursive equations for computing the predictive distribution $p(x_k \mid y_{1:k-1})$ and the filtering distribution $p(x_k \mid y_{1:k})$ at the time-step k are given by the following;

1. *Initialization:* The recursion stats with the prior distribution $p(x_0)$.

2. *Prediction Step:* The predictive distribution of the state x_k at the time-step k, given the dynamic model $p(x_k \mid x_{k-1})$, can be computed by the Chapman-Kolmogorov equation,

$$p(x_k \mid y_{1:k-1}) = \int p(x_k \mid x_{k-1})\ p(x_{k-1} \mid y_{1:k-1})\ dx_{k-1}\ . \tag{54}$$

3. *Update Step:* Given the measurement y_k at time-step k the posterior distribution of the state x_k can be computed by Baye's Rule,

$$p(x_k \mid y_{1:k}) = \frac{1}{Z_k} p(y_k \mid x_k)\ p(x_k \mid y_{1:k-1})\ , \tag{55}$$

$$Z_k = \int p(y_k \mid x_k)\ p(x_k \mid y_{1:k-1})\ dx_k\ . \tag{56}$$

Now, we discuss the general Bayesian Smoothing Framework with known parameters θ;

Bayesian Smoothing Equations: The backward recursive equations for computing the smoothed distribution $p(x_k \mid y_{1:T})$ for any time-step $k < T$ are given as follows;

1. *Initialization:* The recursion starts from the last time-step T, hence the filtering distribution and the smoothed distribution at time-step T are equivalent.

2. *Filtering Step:* At time-step k the predictive distribution fat time-step $k+1$ is computed from the Bayesian Filtering equations as,

$$p(x_{k+1} \mid y_{1:k}) = \int p(x_{k+1} \mid x_k)\ p(x_k \mid y_{1:k})\ dx_k\ . \tag{57}$$

3. *Smoothing Step:* At time-step k the smoothed distribution is computed as,

$$p(x_k \mid y_{1:T}) = p(x_k \mid y_{1:k})\ \int \left[\frac{p(x_{k+1} \mid x_k)\ p(x_{k+1} \mid y_{1:T})}{p(x_{k+1} \mid y_{1:k})} \right]]\ dx_{k+1}\ . \tag{58}$$

For the remainder of Section **??** we will restrict ourselves to the following general stochastic discrete-time linear and nonlinear models in order to present the different Bayesian Filtering algorithms;

The Linear State-Measurement Model:

$$x_k = A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + q_{k-1} \ , \tag{59}$$
$$y_k = H_k x_k + r_k \ . \tag{60}$$

The Nonlinear State-Measurement Model:

$$x_k = f_k(x_{k-1}, u_{k-1}) + q_{k-1} \ , \tag{61}$$
$$y_k = h_k(x_k) + r_k \ . \tag{62}$$

Where;

1. x_k : State vector at time-step k s.t. $x \in \mathbb{R}^n$.
2. y_k : Measurement vector at time-step k s.t. $y \in \mathbb{R}^m$.
3. u_k : Input vector at time-step k s.t. $u \in \mathbb{R}^p$.
4. A_k : System Matrix at time-step k s.t. $A \in \mathbb{R}^{n \times n}$.
5. B_k : Input Matrix at time-step k s.t. $B \in \mathbb{R}^{n \times p}$.
6. H_k : Measurement Matrix at time-step k s.t. $H \in \mathbb{R}^{m \times n}$.
7. f_k : State dynamic function at time-step k s.t. $f : \mathbb{R}^n \times \mathbb{R}^p \Rightarrow \mathbb{R}^n$.
8. h_k : Measurement function at time-step k s.t. $h : \mathbb{R}^n \Rightarrow \mathbb{R}^m$.
9. q_k : Additive process noise s.t. $q_k \in \mathbb{R}^n$ and $q_k \sim N(0, Q_k)$, where $Q_k \in \mathbb{R}^{n \times n}$ is process noise covariance matrix.
10. r_k : Additive measurement noise s.t. $r_k \in \mathbb{R}^m$ and $r_k \sim N(0, R_k)$, where $R_k \in \mathbb{R}^{m \times m}$ is measurement noise covariance matrix.

5.2.1 Kalman Filter and Smoother

The Kalman Filter (KF) has the following characteristics:

1. It works for the linear model given in (59) and (60).
2. As the Gaussian process/measurement noise is transformed through a linear model all the resulting distributions of the filter are Gaussian.
3. It is the optimal filter for a linear system with Gaussian process/measurement noise.
4. It computes the following distributions in closed form;

$$\begin{aligned} p(x_k \mid y_{1:k-1}) &= N(x_k \mid m_k^-, P_k^-) \ , \\ p(x_k \mid y_{1:k}) &= N(x_k \mid m_k, P_k) \ , \\ p(y_k \mid y_{1:k-1}) &= N(y_k \mid H_k m_k^-, S_k) \ . \end{aligned} \tag{63}$$

Kalman Filter Algorithm:

1. *Initialization:* Recursion starts with prior mean m_0 and covariance P_0.
2. *Prediction Step:*

$$\begin{aligned} m_k^- &= A_{k-1} m_{k-1} + B_{k-1} u_{k-1} \ , \\ P_k^- &= A_{k-1} P_{k-1} A_{k-1}^T + Q_{k-1} \ . \end{aligned} \tag{64}$$

3. *Update Step:*

$$\begin{aligned} v_k &= y_k - H_k m_k^- \ , \\ S_k &= H_k P_k^{-1} H_k^T + R_k \ , \\ K_k &= P_k^{-1} H_k^T S_k \ , \\ m_k &= m_k^- + K_k v_k \ , \\ P_k &= P_k^- - K_k S_k K_k^T \ . \end{aligned} \tag{65}$$

The Kalman Smoother (KS) has the following characteristics:

1. It works for the linear model given in (59) and (60).
2. As the Gaussian process/measurement noise is transformed through a linear model all the resulting distributions of the smoother are Gaussian.
3. It is the optimal smoother for a linear system with Gaussian process/measurement noise.
4. It computes the following distributions in closed form;

$$p(x_k \mid y_{1:T}) = N(x_k \mid m_k^s, P_k^s) \ . \tag{66}$$

Kalman Smoother Algorithm:

1. *Initialization:* The backward recursion starts with $m_T^s = m_T$ and $P_T^s = P_T$ coming from the last step of the Kalman filter.

2. *Filtering Step:*

$$\begin{aligned} m_{k+1}^- &= A_k m_k + B_k u_k \ , \\ P_{k+1}^- &= A_k P_k A_k^T + Q_k \ . \end{aligned} \tag{67}$$

3. *Smoothing Step:*

$$\begin{aligned} G_k &= P_k A_k^T \left[P_{k+1}^-\right]^{-1} \ , \\ m_k^s &= m_k + G_k \left[m_{k+1}^s - m_{k+1}^-\right] \ , \\ P_k^s &= P_k + G_k \left[P_{k+1}^s - P_{k+1}^-\right] G_k^T \ . \end{aligned} \tag{68}$$

5.2.2 Extended Kalman Filter and Smoother

The Extended Kalman Filter (EKF) has the following characteristics:

1. It works for the nonlinear model given in (61) and (62).
2. As the Gaussian process/measurement noise is transformed through a nonlinear model all the resulting distributions of the filter are non-Gaussian, hence we approximate a Gaussian distribution.
3. It is not the optimal filter for a nonlinear system with Gaussian process/measurement noise.
4. It computes the following approximate distributions in closed form with an assumption that the process/measurement model is linearized at the current best estimate of the state;

$$\begin{aligned} p(x_k \mid y_{1:k-1}) &\approx N(x_k \mid m_k^-, P_k^-) \ , \\ p(x_k \mid y_{1:k}) &\approx N(x_k \mid m_k, P_k) \ , \\ p(y_k \mid y_{1:k-1}) &\approx N(y_k \mid H_k m_k^-, S_k) \ . \end{aligned} \tag{69}$$

Extended Kalman Filter Algorithm:

1. *Initialization:* Recursion starts with prior mean m_0 and covariance P_0.
2. *Prediction Step:*

$$\begin{aligned} m_k^- &= f_k(m_{k-1}, u_{k-1}) \ , \\ P_k^- &= F_{k_x}(m_{k-1}, u_{k-1}) P_{k-1} F_{k_x}^T(m_{k-1}, u_{k-1}) + Q_{k-1} \ . \end{aligned} \tag{70}$$

Where,

$$\begin{aligned} [F_{k_x}(m_{k-1}, u_{k-1})]_{j,j'} &= \left. \frac{\partial f_{k_j}(x,u)}{\partial x_{j'}} \right|_{x=m_{k-1}, u=u_{k-1}} \ , \ F_{k_x}(m_{k-1}, u_{k-1}) \in \mathbb{R}^{n \times n} \ , \\ [H_{k_x}(m_{k-1}, u_{k-1})]_{j,j'} &= \left. \frac{\partial h_{k_j}(x,u)}{\partial x_{j'}} \right|_{x=m_{k-1}, u=u_{k-1}} \ , \ H_{k_x}(m_{k-1}, u_{k-1}) \in \mathbb{R}^{m \times n} \ . \end{aligned} \tag{71}$$

3. *Update Step:*

$$\begin{aligned} v_k &= y_k - h_k(m_k^-) \ , \\ S_k &= H_{k_x}(m_k^-) P_k^{-1} H_{k_x}^T(m_k^-) + R_k \ , \\ K_k &= P_k^{-1} H_{k_x}^T(m_k^-) S_k \ , \\ m_k &= m_k^- + K_k v_k \ , \\ P_k &= P_k^- - K_k S_k K_k^T \ . \end{aligned} \tag{72}$$

The Extended Kalman (EKS) Smoother has the following characteristics:

1. It works for the nonlinear model given in (61) and (62).

2. As the Gaussian process/measurement noise is transformed through a linear model all the resulting distributions of the smoother are Gaussian.

3. It is not the optimal smoother for a nonlinear system with Gaussian process/measurement noise.

4. It computes the following distributions in closed form;

$$p(x_k \mid y_{1:T}) \approx N(x_k \mid m_k^s, P_k^s) \ . \tag{73}$$

Extended Kalman Smoother Algorithm:

1. *Initialization:* The backward recursion starts with $m_T^s = m_T$ and $P_T^s = P_T$ coming from the last step of the Extended Kalman filter.

2. *Filtering Step:*

$$\begin{aligned} m_{k+1}^- &= f(m_k, u_k) \ , \\ P_{k+1}^- &= F_{k_x}(m_k, u_k) P_k F_{k_x}^T(m_k, u_k) + Q_k \ . \end{aligned} \tag{74}$$

Where,

$$[F_{k_x}(m_k, u_k)]_{j,j'} = \left. \frac{\partial f_{k_j}(x, u)}{\partial x_{j'}} \right|_{x=m_k, u=u_k} , \ F_{k_x}(m_k, u_k) \in \mathbb{R}^{n \times n} \ . \tag{75}$$

3. *Smoothing Step:*

$$\begin{aligned} G_k &= P_k F_{k_x}^T(m_k, u_k) \left[P_{k+1}^- \right]^{-1} \ , \\ m_k^s &= m_k + G_k \left[m_{k+1}^s - m_{k+1}^- \right] \ , \\ P_k^s &= P_k + G_k \left[P_{k+1}^s - P_{k+1}^- \right] G_k^T \ . \end{aligned} \tag{76}$$

5.2.3 Unscented Kalman Filter and Smoother

The Unscented Kalman Filter (UKF) has the following characteristics:

1. It works for the nonlinear model given in (61) and (62).

2. As the Gaussian process/measurement noise is transformed through a nonlinear model all the resulting distributions of the filter are non-Gaussian, hence we approximate a Gaussian distribution from a set of $2n+1$ deterministically chosen points called Sigma Points. Now let us consider a random variable x distributed as a Gaussian $x \sim N(m, P)$. We can deterministically compute the sigma points for this distribution as;

$$\begin{aligned} \mathcal{X}^{(0)} &= m \ , \\ \mathcal{X}^{(i)} &= m + \sqrt{n+\lambda}\left[\sqrt{P}\right]_i \ , \\ \mathcal{X}^{(i+n)} &= m - \sqrt{n+\lambda}\left[\sqrt{P}\right]_i \ . \end{aligned} \tag{77}$$

   Where, $[.]_i$ denotes the i^{th} column of the matrix, λ is a scaling parameter given by;

$$\lambda = \alpha^2(n+k) - n \ . \tag{78}$$

   Where α and k determine the spread of Sigma points around the mean, and $P \triangleq \sqrt{P}\sqrt{P}^T$.

3. Once we have the Sigma points, we can propagate them through a nonlinear function $y = g(x)$ to obtain transformed Sigma points as follows;

$$\mathcal{Y}^{(i)} = g(\mathcal{X}^{(i)}) \ , \ i = 0, \ldots, 2n \ . \tag{79}$$

4. The mean $E[g(x)]$ and covariance $Cov[g(x)]$ can be computed from the transformed Sigma points as follows;

$$\begin{aligned} E[g(x)] \approx \mu_U &= \sum_{i=0}^{2n} W_i^{(m)} \mathcal{Y}^{(i)} \ , \\ Cov[g(x)] \approx S_U &= \sum_{i=0}^{2n} W_i^{(c)} (\mathcal{Y}^{(i)} - \mu_U)(\mathcal{Y}^{(i)} - \mu_U)^T \ . \end{aligned} \tag{80}$$

   Where the weights $W_i^{(m)}$ and $W_i^{(c)}$ are given as;

$$\begin{aligned} W_0^{(m)} &= \frac{\lambda}{n+\lambda} \ , \\ W_0^{(c)} &= \frac{\lambda}{n+\lambda} + (1 - \alpha^2 + \beta) \ , \\ W_i^{(m)} &= \frac{1}{2(n+\lambda)} \ , i = 1, \ldots, 2n \ , \\ W_i^{(c)} &= \frac{1}{2(n+\lambda)} \ , i = 1, \ldots, 2n \ . \end{aligned} \tag{81}$$

   Where β is an additional parameter that can be used to incorporate prior information.

5. It is not the optimal filter for a nonlinear system with Gaussian process/measurement noise.
6. It approximates distributions in closed form as Gaussians, by directly computing the mean and covariances of a nonlinear function with respect to the Gaussian distribution represented non-parametrically by the Sigma points.
7. It approximates the Filtering distributions as a Gaussian as follows;

$$\begin{aligned} p(x_k \mid y_{1:k-1}) &\approx N(x_k \mid m_k^-, P_k^-) \ , \\ p(x_k \mid y_{1:k}) &\approx N(x_k \mid m_k, P_k) \ , \\ p(y_k \mid y_{1:k-1}) &\approx N(y_k \mid \mu_k, S_k) \ . \end{aligned} \tag{82}$$

Unscented Kalman Filter Algorithm:

1. *Initialization:* Recursion starts with prior mean m_0 and covariance P_0.
2. *Prediction Step:*
   (a) Form Sigma points:

$$\begin{aligned} \mathcal{X}_{k-1}^{(0)} &= m_{k-1} \ , \\ \mathcal{X}_{k-1}^{(i)} &= m_{k-1} + \sqrt{n+\lambda}\left[\sqrt{P_{k-1}}\right]_i \ , \\ \mathcal{X}_{k-1}^{(i+n)} &= m_{k-1} - \sqrt{n+\lambda}\left[\sqrt{P_{k-1}}\right]_i \ , \ i = 1, \ldots, n \ , \end{aligned} \tag{83}$$

   where, λ is given by (78).
   (b) Propagate the Sigma points through the dynamic model:

$$\hat{\mathcal{X}}_k^{(i)} = f(\mathcal{X}_{k-1}^{(i)}, u_{k-1}) \ , \ i = 0, \ldots, 2n \ . \tag{84}$$

   (c) Compute the predicted mean m_k^- and the predicted covariance P_k^-:

$$\begin{aligned} m_k^- &= \sum_{i=0}^{2n} W_i^{(m)} \hat{\mathcal{X}}_k^{(i)} \ , \\ P_k^- &= \sum_{i=0}^{2n} W_i^{(c)} (\hat{\mathcal{X}}_k^{(i)} - m_k^-)(\hat{\mathcal{X}}_k^{(i)} - m_k^-)^T + Q_{k-1} \ , \end{aligned} \tag{85}$$

   where, the weights $W_i^{(m)}$ and $W_i^{(c)}$ are given by (81).
3. *Update Step:*
   (a) Form the Sigma points:

$$\begin{aligned} \mathcal{X}_{k-1}^{-(0)} &= m_k^- \ , \\ \mathcal{X}_{k-1}^{-(i)} &= m_k^- + \sqrt{n+\lambda}\left[\sqrt{P_k^-}\right]_i \ , \\ \mathcal{X}_{k-1}^{-(i+n)} &= m_k^- - \sqrt{n+\lambda}\left[\sqrt{P_k^-}\right]_i \ , \ i = 1, \ldots, n \ , \end{aligned} \tag{86}$$

   where, λ is given by (78).

(b) Propagate Sigma points through the measurement model:

$$\hat{\mathcal{Y}}_k^{(i)} = h(\mathcal{X}_k^{-(i)}) \ , \ i = 0, \ldots, 2n \ . \tag{87}$$

(c) Compute the predicted mean of the measurement μ_k, the predicted covariance of the measurement S_k, and the cross-covariance of the state and the measurement C_k:

$$\begin{aligned} \mu_k &= \sum_{i=0}^{2n} W_i^{(m)} \hat{\mathcal{Y}}_k^{(i)} \ , \\ S_k &= \sum_{i=0}^{2n} W_i^{(c)} (\hat{\mathcal{Y}}_k^{(i)} - \mu_k)(\hat{\mathcal{Y}}_k^{(i)} - \mu_k)^T + R_k \ , \\ C_k &= \sum_{i=0}^{2n} W_i^{(c)} (\mathcal{X}_k^{-(i)} - m_k^-)(\hat{\mathcal{Y}}_k^{(i)} - \mu_k)^T \ , \end{aligned} \tag{88}$$

where, the weights $W_i^{(m)}$ and $W_i^{(c)}$ are given by (81).

(d) Compute the filter gain K_k, the filtered state mean m_k, and the covariance P_k, conditional on the new measurement y_k:

$$\begin{aligned} K_k &= C_k S_k^{-1} \ , \\ m_k &= m_k^- + K_k[y_k - \mu_k] \ , \\ P_k &= P_k^- - K_k S_k K_k^T \ . \end{aligned} \tag{89}$$

The Unscented Kalman (UKS) Smoother has the following characteristics:

1. It works for the nonlinear model given in (61) and (62).
2. As the Gaussian process/measurement noise is transformed through a nonlinear model all the resulting distributions of the smoother are non-Gaussian.
3. It is not an optimal smoother for a nonlinear system with Gaussian process/measurement noise.
4. It computes the following distributions in closed form;

$$p(x_k \mid y_{1:T}) \approx N(x_k \mid m_k^s, P_k^s) \ . \tag{90}$$

Unscented Kalman Smoother Algorithm:

1. *Initialization:* The backward recursion starts with $m_T^s = m_T$ and $P_T^s = P_T$ coming from the last step of the Unscented Kalman filter.
2. *Filtering Step:*

(a) Form the Sigma points:

$$\begin{aligned} \mathcal{X}_k^{(0)} &= m_k \ , \\ \mathcal{X}_k^{(i)} &= m_k + \sqrt{n+\lambda} \left[\sqrt{P_k} \right]_i \ , \\ \mathcal{X}_k^{(i+n)} &= m_k - \sqrt{n+\lambda} \left[\sqrt{P_k} \right]_i \ , \ i = 1, \ldots, n \ , \end{aligned} \tag{91}$$

where, λ is given by (78).

(b) Propagate Sigma points through the dynamic model:

$$\hat{\mathcal{X}}_{k+1}^{(i)} = f(\mathcal{X}_k^{(i)}, u_k) \ , \ i = 0, \ldots, 2n \ . \tag{92}$$

(c) Compute the predicted mean m_{k+1}^-, the predicted covariance P_{k+1}^-, and the predicted cross-variance D_{k+1}:

$$\begin{aligned} m_{k+1}^- &= \sum_{i=0}^{2n} W_i^{(m)} \hat{\mathcal{X}}_{k+1}^{(i)} \ , \\ P_{k+1}^- &= \sum_{i=0}^{2n} W_i^{(c)} (\hat{\mathcal{X}}_{k+1}^{(i)} - m_{k+1}^-)(\hat{\mathcal{X}}_{k+1}^{(i)} - m_{k+1}^-)^T + Q_k \ , \\ D_{k+1} &= \sum_{i=0}^{2n} W_i^{(c)} (\mathcal{X}_{k+1}^{(i)} - m_k)(\hat{\mathcal{X}}_{k+1}^{(i)} - m_{k+1}^-)^T \ , \end{aligned} \tag{93}$$

where, the weights $W_i^{(m)}$ and $W_i^{(c)}$ are given by (81).

3. *Smoothing Step:*

(a) Compute the smoother gain G_k, the smoothed mean m_k^s, and the smoothed covariance P_k^s:

$$\begin{aligned} G_k &= D_{k+1} \left[P_{k+1}^- \right]^{-1} \ , \\ m_k^s &= m_k + G_k \left[m_{k+1}^s - m_{k+1}^- \right] \ , \\ P_k^s &= P_k + G_k \left[P_{k+1}^s - P_{k+1}^- \right] G_k^T \ . \end{aligned} \tag{94}$$

5.2.4 Gaussian Filter and Smoother

The Gaussian Filter (GF) has the following characteristics:

1. It works for the nonlinear model given in (61) and (62).

2. As the Gaussian process/measurement noise is transformed through a nonlinear model all the resulting distributions of the filter are non-Gaussian, hence we approximate the Gaussian integrals through approximate computation schemes. Now let us consider a random variable x distributed as a Gaussian $x \sim N(m, P)$ and a general nonlinear function $g(x)$. We can compute the Gaussian integral given as;

$$\int_{-\infty}^{\infty} g(x)N(x \mid m, P)\, dx \ ,$$

as an approximation by different numerical approximations. The two most commonly used numerical approximations of the Gaussian integrals are as follows:

(a) Gauss-Hermite Quadrature:

i. The unit Sigma points $\zeta^{i_{i'}},\ i' \in \{1, \ldots, p\}$ are computed as roots of p^{th} order Hermite polynomial $H_p(x)$. We can construct a p^{th} order Hermite polynomial as follows;

$$\begin{aligned} H_0(x) &= 1 \ , \\ H_1(x) &= x \ , \\ H_{p+1}(x) &= xH_p(x) - pH_{p-1}(x) \ . \end{aligned} \tag{95}$$

ii. Compute multi-dimensional weights as products of one-dimensional weights:

$$\begin{aligned} W_{i_1,\ldots,i_n} &= W_{i_1} \times \cdots \times W_{i_n} \ , \\ &= \frac{p!}{p^2[H_{p-1}(\zeta^{(i_1)})]^2} \times \cdots \times \frac{p!}{p^2[H_{p-1}(\zeta^{(i_n)})]^2} \ , \end{aligned} \tag{96}$$

where, each i_k takes values $1, \ldots, p$.

iii. Form multi-dimensional unit Sigma points as Cartesian product of the one-dimensional unit Sigma points:

$$\zeta^{(i_1,\ldots,i_n)} = \begin{pmatrix} \zeta^{i_1} \\ \vdots \\ \zeta^{i_n} \end{pmatrix} \tag{97}$$

iv. Approximate the integral as:

$$\int_{-\infty}^{\infty} g(x)N(x \mid m, P)\, dx \approx \sum_{i_1,\ldots,i_n} W_{i_1,\ldots,i_n}\, g(m + \sqrt{P}\zeta^{(i_1,\ldots,i_n)}) \ , \tag{98}$$

where, $P \triangleq \sqrt{P}\sqrt{P}^T$.

(b) Spherical Cubature:

i. Compute the unit Sigma points:

$$\zeta^i = \begin{cases} \sqrt{n}e_i & , \ i = 1, \ldots, n \ , \\ -\sqrt{n}e_{i-n} & , \ i = n+1, \ldots, 2n \ , \end{cases} \tag{99}$$

where, e_i is a unit vector in the direction of the i^{th} coordinate axis.

ii. Approximate the integral:

$$\int_{-\infty}^{\infty} g(x)N(x \mid m, P) \, dx \approx \frac{1}{2n} \sum_{i=1}^{2n} g(m + \sqrt{P}\zeta^{(i)}) \ , \tag{100}$$

where, $P \triangleq \sqrt{P}\sqrt{P}^T$.

3. It is not the optimal filter for a nonlinear system with Gaussian process/measurement noise.

4. It approximates distributions in closed form as Gaussians, by directly computing the approximate Gaussian integrals of a nonlinear function with respect to the Gaussian distribution represented non-parametrically by appropriate Sigma points.

5. It approximates the Filtering distributions as a Gaussian as follows;

$$\begin{aligned} p(x_k \mid y_{1:k-1}) &\approx N(x_k \mid m_k^-, P_k^-) \ , \\ p(x_k \mid y_{1:k}) &\approx N(x_k \mid m_k, P_k) \ , \\ p(y_k \mid y_{1:k-1}) &\approx N(y_k \mid \mu_k, S_k) \ . \end{aligned} \tag{101}$$

Gaussian Filter Algorithms:

1. *Initialization:* Recursion starts with prior mean m_0 and covariance P_0.

2. *Prediction Step:*

(a) Form Sigma points:

i. Gauss-Hermite Kalman Filter (GHKF):

$$\mathcal{X}_{k-1}^{(i_1,\ldots,i_n)} = m_{k-1} + \sqrt{P_{k-1}}\zeta^{(i_1,\ldots,i_n)} \ , i_1, \ldots, i_n = 1, \ldots, p \ , \tag{102}$$

where the unit Sigma points are defined in (97).

ii. Cubature Kalman Filter (CKF):

$$\mathcal{X}_{k-1}^{(i)} = m_{k-1} + \sqrt{P_{k-1}}\zeta^{(i)} \ , i = 1, \ldots, 2n \ , \tag{103}$$

where the unit Sigma points are defined in (99).

(b) Propagate the Sigma points through the dynamic model:

i. Gauss-Hermite Kalman Filter (GHKF):

$$\hat{\mathcal{X}}_k^{(i_1,\ldots,i_n)} = f(\mathcal{X}_{k-1}^{(i_1,\ldots,i_n)}, u_{k-1}) \ , \ i_1, \ldots, i_n = 1, \ldots, p \ . \tag{104}$$

ii. Cubature Kalman Filter (CKF):

$$\hat{\mathcal{X}}_k^{(i)} = f(\mathcal{X}_{k-1}^{(i)}, u_{k-1}) \ , \ i = 0, \ldots, 2n \ . \tag{105}$$

(c) Compute the predicted mean m_k^- and the predicted covariance P_k^-:

i. Gauss-Hermite Kalman Filter (GHKF):

$$\begin{aligned} m_k^- &= \sum_{i_1,\ldots,i_n} W_{i_1,\ldots,i_n} \hat{\mathcal{X}}_k^{(i_1,\ldots,i_n)} \ , \\ P_k^- &= \sum_{i_1,\ldots,i_n} W_{i_1,\ldots,i_n} (\hat{\mathcal{X}}_k^{(i_1,\ldots,i_n)} - m_k^-)(\hat{\mathcal{X}}_k^{(i_1,\ldots,i_n)} - m_k^-)^T + Q_{k-1} \ , \end{aligned} \tag{106}$$

where the weights are defined in (96).

ii. Cubature Kalman Filter (CKF):

$$\begin{aligned} m_k^- &= \frac{1}{2n} \sum_{i=1}^{2n} \hat{\mathcal{X}}_k^{(i)} \ , \\ P_k^- &= \frac{1}{2n} \sum_{i=1}^{2n} (\hat{\mathcal{X}}_k^{(i)} - m_k^-)(\hat{\mathcal{X}}_k^{(i)} - m_k^-)^T + Q_{k-1} \ , \end{aligned} \tag{107}$$

3. *Update Step:*

(a) Form the Sigma points:

i. Gauss-Hermite Kalman Filter (GHKF):

$$\mathcal{X}_k^{-(i_1,\ldots,i_n)} = m_k^- + \sqrt{P_k^-}\zeta^{(i_1,\ldots,i_n)} \ , i_1, \ldots, i_n = 1, \ldots, p \ , \tag{108}$$

where the unit Sigma points are defined in (97).

ii. Cubature Kalman Filter (CKF):

$$\mathcal{X}_k^{-(i)} = m_k^- + \sqrt{P_k^-}\zeta^{(i)} \ , i = 1, \ldots, 2n \ , \tag{109}$$

where the unit Sigma points are defined in (99).

(b) Propagate Sigma points through the measurement model:

i. Gauss-Hermite Kalman Filter (GHKF):

$$\hat{\mathcal{Y}}_k^{(i_1,\ldots,i_n)} = h(\mathcal{X}_k^{-(i_1,\ldots,i_n)}) \ , i_1, \ldots, i_n = 1, \ldots, p \ , \tag{110}$$

ii. Cubature Kalman Filter (CKF):

$$\hat{\mathcal{Y}}_k^{(i)} = h(\mathcal{X}_k^{-(i)}) \ , \ i = 1, \ldots, 2n \ . \tag{111}$$

(c) Compute the predicted mean of the measurement μ_k, the predicted covariance of the measurement S_k, and the cross-covariance of the state and the measurement C_k:

i. Gauss-Hermite Kalman Filter (GHKF):

$$\begin{aligned}
\mu_k &= \sum_{i_1,\ldots,i_n} W_{i_1,\ldots,i_n} \hat{\mathcal{Y}}_k^{(i_1,\ldots,i_n)} \,, \\
S_k &= \sum_{i_1,\ldots,i_n} W_{i_1,\ldots,i_n} (\hat{\mathcal{Y}}_k^{(i_1,\ldots,i_n)} - \mu_k)(\hat{\mathcal{Y}}_k^{(i_1,\ldots,i_n)} - \mu_k)^T + R_k \,, \\
C_k &= \sum_{i_1,\ldots,i_n} W_{i_1,\ldots,i_n} (\mathcal{X}_k^{-(i_1,\ldots,i_n)} - m_k^-)(\hat{\mathcal{Y}}_k^{(i_1,\ldots,i_n)} - \mu_k)^T \,,
\end{aligned} \tag{112}$$

where the weights are defined in (96).

ii. Cubature Kalman Filter (CKF):

$$\begin{aligned}
\mu_k &= \frac{1}{2n} \sum_{i=1}^{2n} \hat{\mathcal{Y}}_k^{(i)} \,, \\
S_k &= \frac{1}{2n} \sum_{i=1}^{2n} (\hat{\mathcal{Y}}_k^{(i)} - \mu_k)(\hat{\mathcal{Y}}_k^{(i)} - \mu_k)^T + R_k \,, \\
C_k &= \frac{1}{2n} \sum_{i=1}^{2n} (\mathcal{X}_k^{-(i)} - m_k^-)(\hat{\mathcal{Y}}_k^{(i)} - \mu_k)^T \,,
\end{aligned} \tag{113}$$

(d) Compute the filter gain K_k, the filtered state mean m_k, and the covariance P_k, conditional on the new measurement y_k, for both GHKF and CKF:

$$\begin{aligned}
K_k &= C_k S_k^{-1} \,, \\
m_k &= m_k^- + K_k[y_k - \mu_k] \,, \\
P_k &= P_k^- - K_k S_k K_k^T \,.
\end{aligned} \tag{114}$$

The Gaussian Smoother (GS) has the following characteristics:

1. It works for the nonlinear model given in (61) and (62).
2. As the Gaussian process/measurement noise is transformed through a nonlinear model all the resulting distributions of the smoother are non-Gaussian.
3. It is not an optimal smoother for a nonlinear system with Gaussian process/measurement noise.
4. It computes the following distributions in closed form by approximating the nonlinear distributions as Gaussian distributions via employing accurate numerical approximations to compute Gaussian integrals of nonlinear functions;

$$p(x_k \mid y_{1:T}) \approx N(x_k \mid m_k^s, P_k^s) \,. \tag{115}$$

Gaussian Smoother Algorithms:

1. *Initialization:* The backward recursion starts with $m_T^s = m_T$ and $P_T^s = P_T$ coming from the last step of the Gaussian filter.

2. *Filtering Step:*

   (a) Form the Sigma points:

      i. Gauss-Hermite Kalman Smoother (GHKS):

$$\mathcal{X}_k^{(i_1,\ldots,i_n)} = m_k + \sqrt{P_k}\zeta^{(i_1,\ldots,i_n)} \, , i_1, \ldots, i_n = 1, \ldots, p \, , \tag{116}$$

      where the unit Sigma points are defined in (97).

      ii. Cubature Kalman Smoother (CKS):

$$\mathcal{X}_k^{(i)} = m_k + \sqrt{P_k}\zeta^{(i)} \, , i = 1, \ldots, 2n \, , \tag{117}$$

      where the unit Sigma points are defined in (99).

   (b) Propagate Sigma points through the dynamic model:

      i. Gauss-Hermite Kalman Smoother (GHKS):

$$\hat{\mathcal{X}}_{k+1}^{(i_1,\ldots,i_n)} = f(\mathcal{X}_k^{(i_1,\ldots,i_n)}, u_k) \, , \; i_1, \ldots, i_n = 1, \ldots, p \, . \tag{118}$$

      ii. Cubature Kalman Smoother (CKS):

$$\hat{\mathcal{X}}_{k+1}^{(i)} = f(\mathcal{X}_k^{(i)}, u_k) \, , \; i = 0, \ldots, 2n \, . \tag{119}$$

   (c) Compute the predicted mean m_{k+1}^-, the predicted covariance P_{k+1}^-, and the predicted cross-variance D_{k+1}:

      i. Gauss-Hermite Kalman Smoother (GHKS):

$$\begin{aligned}
m_{k+1}^- &= \sum_{i_1,\ldots,i_n} W_{i_1,\ldots,i_n} \hat{\mathcal{X}}_{k+1}^{(i_1,\ldots,i_n)} \, , \\
P_{k+1}^- &= \sum_{i_1,\ldots,i_n} W_{i_1,\ldots,i_n} (\hat{\mathcal{X}}_{k+1}^{(i_1,\ldots,i_n)} - m_{k+1}^-)(\hat{\mathcal{X}}_{k+1}^{(i_1,\ldots,i_n)} - m_{k+1}^-)^T + Q_k \, , \\
D_{k+1} &= \sum_{i_1,\ldots,i_n} W_{i_1,\ldots,i_n} (\hat{\mathcal{X}}_k^{(i_1,\ldots,i_n)} - m_k)(\hat{\mathcal{X}}_{k+1}^{(i_1,\ldots,i_n)} - m_{k+1}^-)^T \, ,
\end{aligned} \tag{120}$$

      where the weights are defined in (96).

      ii. Cubature Kalman Smoother (CKS):

$$\begin{aligned}
m_{k+1}^- &= \frac{1}{2n} \sum_{i=1}^{2n} \hat{\mathcal{X}}_{k+1}^{(i)} \, , \\
P_{k+1}^- &= \frac{1}{2n} \sum_{i=1}^{2n} (\hat{\mathcal{X}}_{k+1}^{(i)} - m_{k+1}^-)(\hat{\mathcal{X}}_{k+1}^{(i)} - m_{k+1}^-)^T + Q_k \, , \\
D_{k+1} &= \frac{1}{2n} \sum_{i=1}^{2n} (\hat{\mathcal{X}}_k^{(i)} - m_k)(\hat{\mathcal{X}}_{k+1}^{(i)} - m_{k+1}^-)^T \, ,
\end{aligned} \tag{121}$$

3. *Smoothing Step:*

(a) Compute the smoother gain G_k, the smoothed mean m_k^s, and the smoothed covariance P_k^s for both GHKS and CKS:

$$\begin{aligned} G_k &= D_{k+1}\left[P_{k+1}^{-}\right]^{-1} , \\ m_k^s &= m_k + G_k\left[m_{k+1}^s - m_{k+1}^{-}\right] , \\ P_k^s &= P_k + G_k\left[P_{k+1}^s - P_{k+1}^{-}\right] G_k^T . \end{aligned} \tag{122}$$

5.2.5 Particle Filter and Smoother

The Particle Filter (PF) has the following characteristics:

1. It works for the nonlinear model given in (61) and (62).

2. As the Gaussian process/measurement noise is transformed through a nonlinear model all the resulting distributions of the filter are non-Gaussian, hence we approximate the non-Gaussian integrals through approximate computation schemes through the Monte Carlo framework. Now let us consider a random variable $x \in \mathbb{R}^n$ distributed as a non-Gaussian $x \sim p(x \mid y_{1:T})$ and a general nonlinear function $g(x) : \mathbb{R}^n \to \mathbb{R}^m$. We can compute the non-Gaussian integral using a non-parametrized target distribution using N samples drawn from the target distribution given as;

$$\begin{aligned} E[g(x) \mid y_{1:T}] &= \int_{-\infty}^{\infty} g(x)p(x \mid y_{1:T})\, dx \ , \\ &\approx \frac{1}{N}\sum_{i=1}^{N} g(x^{(i)}) \ , \ x^{(i)} \sim p(x \mid y_{1:T}) \ , \ i = 1, \ldots, N \ . \end{aligned} \tag{123}$$

   The convergence of Monte Carlo methods is in theory guaranteed by the Central Limit Theorem (CLT); moreover, its invariance with respect to the dimensionality of the random variable makes it unparalleled to any other method.

3. However, obtaining samples from the target distribution $p(x \mid y_{1:T})$ is usually impractical due to its complicated functional form. We can, however, sample from an importance distribution $\pi(x \mid y_{1:T})$ which we choose in a manner that sampling from it is easy. The importance sampling algorithm is given as follows:

   Importance Sampling:

   (a) Draw N samples from the importance distribution,,

$$x^{(i)} \sim \pi(x \mid y_{1:T}) \ , \ i = 1, \ldots, N \ . \tag{124}$$

   (b) Compute the unnormalized weights as,

$$w^{*(i)} = \frac{p(y_{1:T} \mid x^{(i)})p(x^{(i)})}{\pi(x^{(i)} \mid y_{1:T})} \ , \tag{125}$$

   and the normalized weights as,

$$w^{(i)} = \frac{w^{*(i)}}{\sum_{j=1}^{N} w^{*(j)}} \ . \tag{126}$$

   (c) The approximation to the posterior expectation of $g(x)$ is computed as follows,

$$E[g(x) \mid y_{1:T}] \approx \sum_{i=1}^{N} w^{(i)} g(x^{(i)}) \ . \tag{127}$$

(d) The approximation to the posterior probability distribution is given as follows,

$$p(x \mid y_{1:T}) \approx \sum_{i=1}^{N} w^{(i)} \delta(x - x^{(i)}) \ , \tag{128}$$

where, $\delta(.)$ is the Dirac delta function.

4. The above sampling method is for a general model, now considering a stochastic state-space model with states $x_k \in \mathbb{R}^n$ and measurements $y_k \in \mathbb{R}^m$ as follows,

$$\begin{aligned} x_k &\sim p(x_k \mid x_{k-1}) \ , \\ y_k &\sim p(y_k \mid x_k) \ . \end{aligned} \tag{129}$$

We can now develop a sequential version of the Importance sampling algorithm utilizing the Markovian properties of the state-space model leading to a recursive form as follows:

Sequential Importance Sampling with Resampling (SIS/SIR):

(a) Draw N samples $x_0^{(i)}$ from the prior as follows,

$$x_0^{(i)} \sim p(x_0) \ , \ i = 1, \ldots, N \ , \tag{130}$$

now set initial weights $w_0^{(i)} = 1/N$, for all $i = 1, \ldots, N$.

(b) For each $k = 1, \ldots, T$ do as follows,

i. Draw samples $x_k^{(i)}$ from the importance distribution as follows,

$$x_k^{(i)} \sim \pi(x_k \mid x_{0:k-1}^{(i)}, y_{1:k}) \ , \ i = 1, \ldots, N \ . \tag{131}$$

Note: importance distribution can be selected to be Markovian such that $\pi(x_k \mid x_{0:k-1}^{(i)}, y_{1:k}) = \pi(x_k \mid x_{k-1}^{(i)}, y_{1:k})$

ii. Calculate new weights as follows,

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(y_k \mid x_k^{(i)}) p(x_k^{(i)} \mid x_{k-1}^{(i)})}{\pi(x_k \mid x_{0:k-1}^{(i)}, y_{1:k})} \ , \tag{132}$$

then normalize to unity.

iii. If effective number of particles is too low i.e. $n_{eff} < N/10$, where,

$$n_{eff} \approx \frac{1}{\sum_{i=1}^{N} \left(w_k^{(i)} \right)^2} \ , \tag{133}$$

we perform resampling as follows,

Resampling Algorithm:

A. Interpret each weight $w_k^{(i)}$ as the probability of obtaining the sample index i in the set $\{x_k^{(i)} \ : \ i = 1, \ldots, N\}$.

B. Draw N samples from that discrete distribution and replace the old sample set with this new one.

C. Set all weights to the constant value $w_k^{(i)} = 1/N$.

Note: we perform resampling as a counter to the degeneracy problem caused by most particles having zero or near zero weights which can cause trouble for convergence.

iv. The approximation to the posterior expectation of $g(x_k)$ is computed as follows,

$$E[g(x_k) \mid y_{1:k}] \approx \sum_{i=1}^{N} w_k^{(i)} g(x_k^{(i)}) \ . \tag{134}$$

v. The approximation to the posterior probability distribution is given as follows,

$$p(x_k \mid y_{1:k}) \approx \sum_{i=1}^{N} w_k^{(i)} \delta(x_k - x_k^{(i)}) \ , \tag{135}$$

where, $\delta(.)$ is the Dirac delta function.

Particle Filter Algorithm:

1. The SIS/SIR algorithm discussed previously is called the particle filter algorithm.
2. The choice of the importance distribution plays an important role in the practical implementation of this algorithm.
3. The following distributions are used as importance distributions for practical implementation:
   (a) Using process model as importance distribution leads to a bootstrap particle filter, which might need very large number of Monte Carlo samples due to the inefficiency of the importance distribution.
   (b) The filter distributions computed from EKF, UKF, GF, and any kind of their ensembles can be used as importance distributions, leading to different flavors of the particle filter. However, it has been noted that artificially increasing the covariance of these distributions or replacing them with a Student's t distribution helps in improving convergence.
   (c) One more thing to note is that resampling introduces another problem of sample impoverishment i.e. it may lead to making large copies of a few particles with high weights. This problem of sample impoverishment can be mitigated by introducing resample-move, regularization, or MCMC steps algorithms.
   (d) During the computation of weights for SIS/SIR we shouldn't forget that,

$$\begin{aligned} p(x_k \mid x_{k-1}) &= N(f_k(x_{k-1}, u_{k-1}) \ , \ Q_{k-1}) \ , \\ p(y_k \mid x_k) &= N(h_k(x_k) \ , \ R_k) \ , \end{aligned} \tag{136}$$

   and that we can always numerically compute $p(x < x_k \mid x_{k-1]})$ and $p(y < y_k \mid x_k)$ as they are standard Gaussians.

The Particle Smoother (PS) has the following characteristics:

1. It works for the nonlinear model given in (61) and (62).
2. As the Gaussian process/measurement noise is transformed through a nonlinear model all the resulting distributions of the smoother are non-Gaussian.
3. It is not an optimal smoother for a nonlinear system with Gaussian process/measurement noise.
4. It computes the smoothing distribution in a non-parametric form using a collection of weighted and importance sampled particles, using which the expectation of any function over the smoothing distribution can be computed efficiently.

Particle Smoother Algorithm:

1. *Filtering Step:*
   (a) Run the Particle filter to obtain a weighted set of particles representing filtering distributions as follows,

$$\{w_k^i \ , \ x_k^{(i)} \ : \ i = 1, \ldots, N \ , \ k = 1, \ldots, T\} \ ,$$
$$p(x_k \mid y_{1:k}) \approx \sum_{i=1}^{N} w_k^{(i)} \delta(x_k - x_k^{(i)}) \ , \tag{137}$$

2. *Smoothing Step:*
   (a) For $j = 1, \ldots, S$ do as follows:
      i. Choose $\tilde{x}_T = x_T^{(i)}$ with probability $w_T^{(i)}$.
      ii. For $k = T-1, \ldots, 0$ do as follows:
         A. Compute new weights as,

$$w_{k|k+1}^{(i)} \propto w_k^{(i)} p(\tilde{x}_{k+1} \mid x_k^{(i)}) \ . \tag{138}$$

         B. Choose $\tilde{x}_k = x_k^{(i)}$ with probability $w_{k|k+1}^{(i)}$
         C. We develop a set of weighted particles for this j^{th} iteration as follows,

$$\{w_{k|k+1_{(j)}}^{(i)} \ , \ \tilde{x}_k^{(j)} \ : \ k = T, \ldots, 0\} \ . \tag{139}$$

   (b) We develop S sets of weighted particles as above.
   (c) For the S , $j = 1, \ldots, S$ iterations the full smoothing distribution is given by,

$$p(x_{0:T} \mid y_{1:T}) \approx \frac{1}{S} \sum_{j=1}^{S} \delta(x_{0:T} - \tilde{x}_{0:T}^{(j)}) \ . \tag{140}$$

   (d) The marginal of the full smoothing distribution for the k^{th} time-step is given as,

$$p(x_k \mid y_{1:T}) \approx \frac{1}{S} \sum_{j=1}^{S} w_{k|k+1_{(j)}}^{(i)} \delta(x_k - \tilde{x}_k^{(j)}) \ . \tag{141}$$

6 Bayesian Parameter Estimation Methods

In this section, we will describe the common methods for parameter estimation using the Bayesian Estimation Framework.

6.1 Bayesian Filter Estimation via State Augmentation

Let us consider a nonlinear process-measurement model with unknown parameter θ as follows;

$$\begin{aligned} x_k &= f_k(x_{k-1}, u_{k-1}, \theta) + q_{k-1} \ , \\ y_k &= h_k(x_k, \theta) + r_k \ . \end{aligned} \tag{142}$$

Now, the complete model with time-invariant parameter θ can be given as;

$$\begin{aligned} \theta_k &= \theta_{k-1} \ , \\ x_k &= f_k(x_{k-1}, u_{k-1}, \theta_{k-1}) + q_{k-1} \ , \\ y_k &= h_k(x_k, \theta_{k-1}) + r_k \ . \end{aligned} \tag{143}$$

Now, let us define the augmented state as $\tilde{x}_k \triangleq [x_k, \theta_k]^T$. We can now rewrite the model in terms of the augmented state as follows;

$$\begin{aligned} \tilde{x}_k &= \tilde{f}_k(\tilde{x}_{k-1}, u_{k-1}) + \tilde{q}_{k-1} \ , \\ y_k &= \tilde{h}_k(\tilde{x}_k) + r_k \ . \end{aligned} \tag{144}$$

Where we have $\tilde{q}_k \triangleq [q_k, \underline{0}]^T$, $\underline{0} \in \mathbb{R}^n$.

Now, the above model can be solved using any filtering algorithm described in the previous sections, and we will get recursive estimates of the state and parameters jointly. However, in the current form of the model there are the following disadvantages:

1. As no noise is present in the parameter equation i.e. the parameter is time-invariant, the dynamic model of the parameter is singular, which problems with non-linear filtering algorithms causing it to diverge.

2. We can use the above formulation of the model when the entire system is linear and the parameters also appear linearly in the system.

To avoid the above disadvantages for a model which is non-linear in both state and parameters, we introduce artificial noise in the parameter dynamic equation as follows;

$$\theta_k = \theta_{k-1} + q_{k-1_\theta} \ . \tag{145}$$

Now, the state augmented process/measurement model (144) with $\tilde{q}_k \triangleq [q_{k_x}, q_{k_\theta}]^T$, $q_{k_\theta} \in \mathbb{R}^n$, can be used with any of the appropriate filtering algorithms described in the previous section to provide for a more stable and robust implementation to compute recursively both the state and parameters of the model. However, even this modified problem with a time-variant parameter with a small noise process can be inapplicable but it is advised to start from this method due to its implementation simplicity before moving on to more complicated methods.

6.2 Batch Estimation

We will follow [6] for this subsection, with all the notations developed in the previous sections, so as to keep the notation homogeneous.

6.3 Maximum Likelihood Estimation (MLE)

6.4 Maximum A Posteriori (MAP) Estimation

6.5 Expectation Maximization (EM) based Estimation

7 Results and Discussion

8 Conclusion

References

[1] Y. Lin, T. Middelkoop, and P. Barooah, "Issues in identification of control-oriented thermal models of zones in multi-zone buildings," in *2012 IEEE 51st IEEE conference on decision and control (CDC)*. IEEE, 2012, pp. 6932–6937.

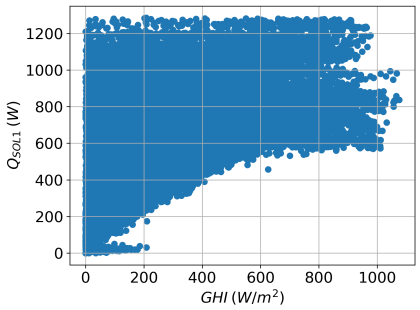
[2] [Online]. Available: https://energyplus.net/

[3] "Prototype building models." [Online]. Available: https://www.energycodes.gov/prototype-building-models

[4] B. Cui, C. Fan, J. Munk, N. Mao, F. Xiao, J. Dong, and T. Kuruganti, "A hybrid building thermal modeling approach for predicting temperatures in typical, detached, two-story houses," *Applied energy*, vol. 236, pp. 101–116, 2019.

[5] S. Särkkä and L. Svensson, *Bayesian filtering and smoothing*. Cambridge university press, 2023, vol. 17.

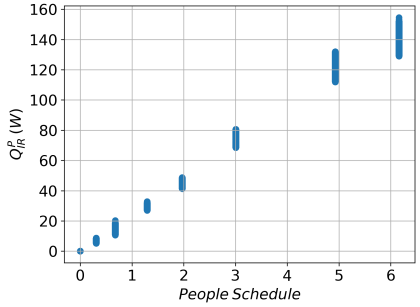
[6] D. G. Robertson, J. H. Lee, and J. B. Rawlings, "A moving horizon-based approach for least-squares estimation," *AIChE Journal*, vol. 42, no. 8, pp. 2209–2224, 1996.



0.32

(a) t

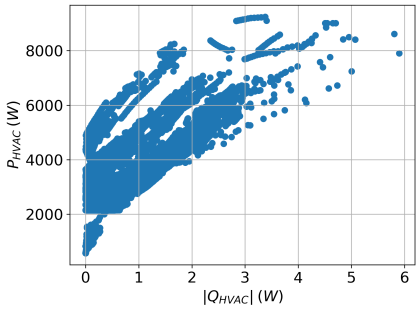
Figure 2: Q_{SOL1} vs GHI



0.32

(a) t

Figure 3: Q_{IR}^{P} vs People Schedule

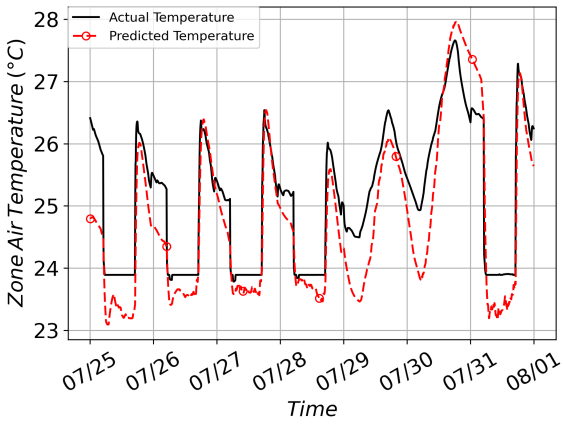


0.32

(a) t

Figure 4: P_{HVAC} vs $|Q_{HVAC}|$

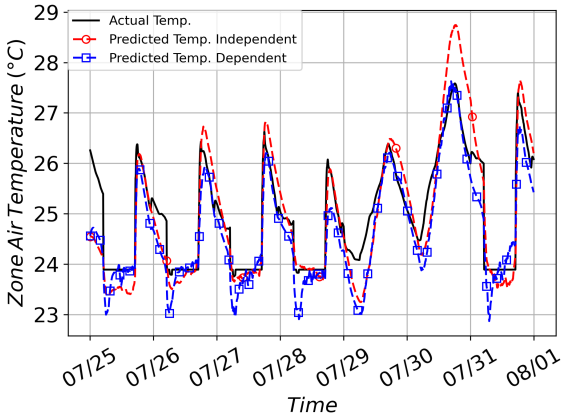
Figure 5: Scatter plots for selected variables.



0.48

(a) t

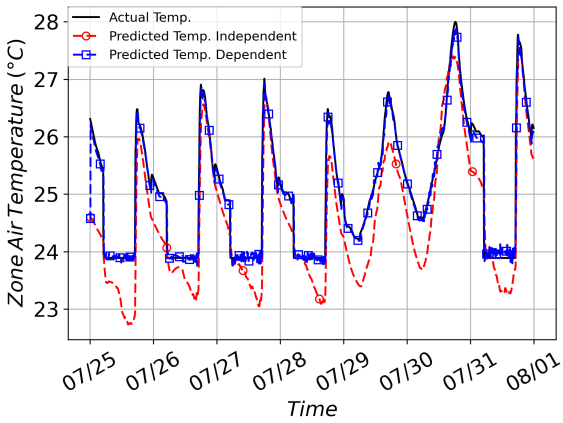
Figure 6: 1 Zone



0.48

(a) t

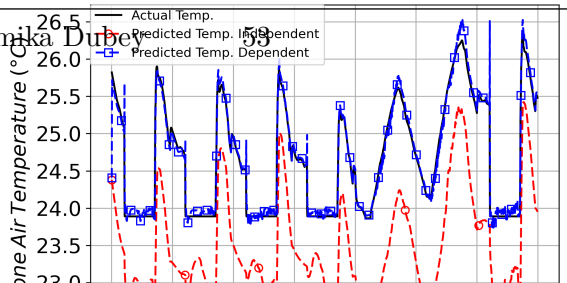
Figure 7: 2 Zone



0.48

(a) t

Figure 8: 3 Zone



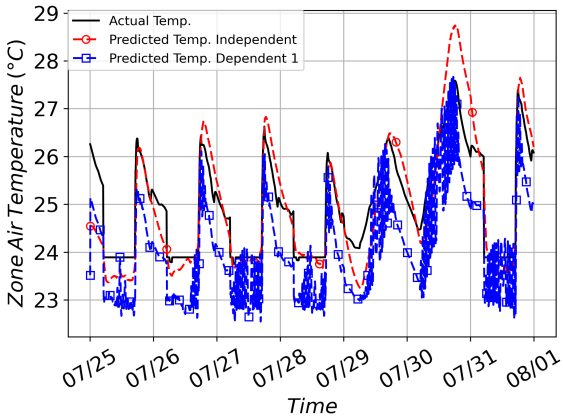
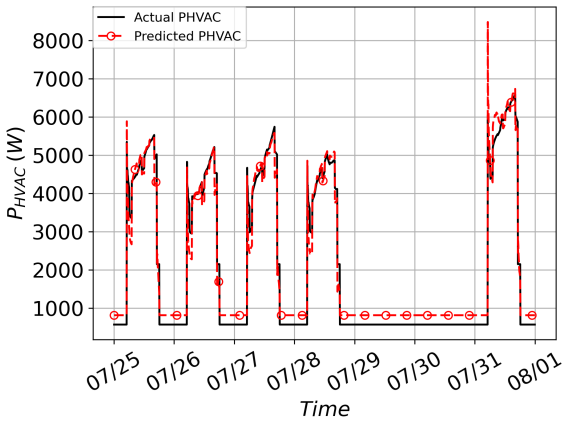


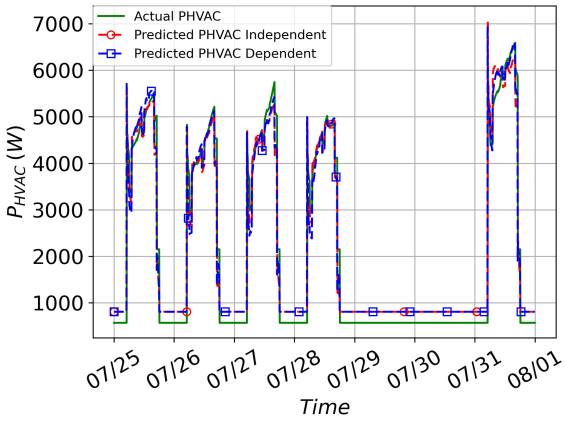
Figure 11: Actual vs predicted T_z for 2Z model.



0.48

(a) t

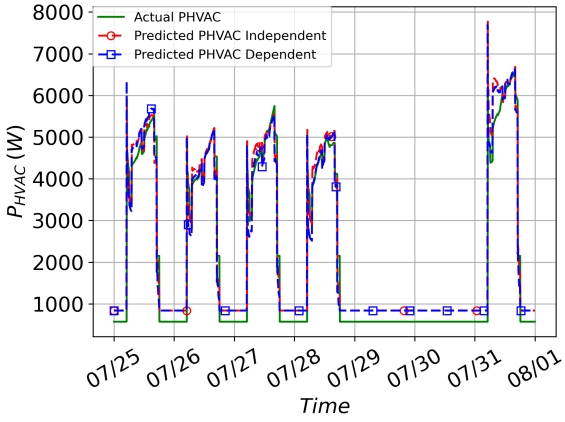
Figure 12: 1 Zone



0.48

(a) t

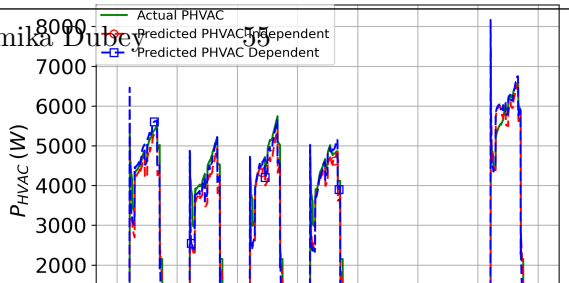
Figure 13: 2 Zone



0.48

(a) t

Figure 14: 3 Zone



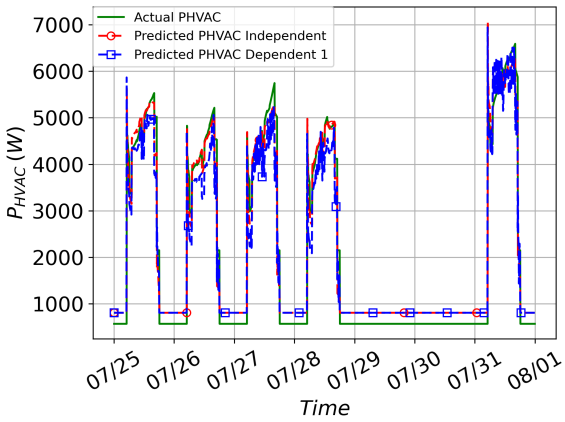


Figure 17: Actual vs predicted P_{HVAC} for 2Z model.