2018 Building Performance Modeling Conference and
SimBuild co-organized by ASHRAE and IBPSA-USA
Chicago, IL
September 26-28, 2018

# DIGITAL TWINS FOR EFFICIENT MODELING AND CONTROL OF BUILDINGS AN INTEGRATED SOLUTION WITH SCADA SYSTEMS

Achin Jain[1], Derek Nong[1], Truong X. Nghiem[2], and Rahul Mangharam[1]
[1]Flexergy AI, Philadelphia, PA
[2]Northern Arizona University, Flagstaff, AZ

## ABSTRACT

We develop an integrated solution for incorporating "digi- tal twins" of real buildings into existing SCADA systems, which enables real-time prediction and advanced control. These digital twins are either EnergyPlus (E+) or data- driven (D+) building models, whose input and output vari- ables are mapped to analogous real building OPC tags and track the real-time operation of the building. An E+ dig- ital twin can be used to provide predictions of the build- ing's performance in different weather, usage, and energy pricing scenarios, which allows for accurate assessment of different control strategies. However, it is not suitable for optimization and predictive control due to its complexity. We develop scalable D+ digital twin based on Gaussian Processes (GP) for accurate prediction and advanced con- trol. A D+ digital twin is much easier, faster, and less ex- pensive to train than developing and tuning an E+ model, while still providing accurate power forecasts and being suitable for control. Data-driven Model Predictive Con- trol (MPC) optimizes control inputs of the predictive D+ model for energy curtailment with thermal comfort guar- antees in demand response applications. The MPC con- troller is integrated into the SCADA environment, demon- strating real-time in-the-loop control of D+ digital twins.

## INTRODUCTION

Efficient control of buildings requires high fidelity models that capture the evolution of the state of the building with time, for example, how the power consumption and zone temperature are affected when the chilled water or the supply air temperature set-points are changed with time or when outside weather conditions are different. Model Predictive Control (MPC) uses such models to predict the state of the building over a finite horizon and optimize the performance with a given objective like load curtailment while meeting thermal comfort and operation constraints.

To this end, the classes of models that are most widely studied in the literature use first principles based on physics. These include the *white box* models typically based on high fidelity simulation software like EnergyPlus (E+) (Deru et al. 2011) and TRNSYS (University of Wis- consin Madison 1975), and the *grey box* models based on Resistor-Capacitance (RC) networks (Deng et al. 2010). The user expertise, time, and associated sensor costs re- quired to develop such models of a single building are very high. This is because such models require detailed information about the geometry of a building, design and equipment layout plans, material properties, and equip- ment and operational schedules. Moreover, the model- ing process also varies from building to building with the construction and types of installed equipment. Af- ter several years of work on using first principles based models for peak power reduction and energy optimization for buildings, multiple authors (Sturzenegger et al. 2016; Žáčeková, Váňa, and Cigler 2014) have concluded that the biggest hurdle to mass adoption of intelligent building control is the cost and effort required to capture accurate dynamical models of the buildings.

We take an alternative route to the physics-based ap- proach, i.e. *black box* modeling based on machine learn- ing algorithms to learn a digital twin of the underlying physical system – a building in this case. Our approach re- duces the cost and time to model the buildings by an order of magnitude (Jain et al. 2018a; Smarra et al. 2018; Jain et al. 2018b; Jain, Smarra, and Mangharam 2017; Jain, Behl, and Mangharam 2017; Nghiem and Jones 2017; Behl, Nghiem, and Mangharam 2015). We learn data- driven (D+) models using only historical data available via sensors already installed in the buildings – thermostats, multimeters – and historical weather data. The D+ models can not only be used for prediction but also for real-time MPC. These models are scalable and integrate seamlessly to the existing Supervisory Control and Data Acquisition (SCADA) systems or the Building Energy Management Systems (BEMS).

Although expensive to build, the E+ models are useful to simulate the behavior of the building. For example, build- ing operators use E+ as an isolated testbed to analyze dif- ferent control strategies and receive immediate feedback without having to implement the strategies on the real building. The drawback of E+ models is that they can- not be used for advanced control like MPC. On the other hand, as we will show later, the D+ models are much less expensive to build and they can be used for simulating the response of the real building as well as for real-time MPC. While the D+ models for control is a novel approach in its own right, the use of D+ and E+ models as real-time digital twins is limited in practice because of lack of inte- gration with existing SCADA systems.

In this paper, we present an end-to-end architecture

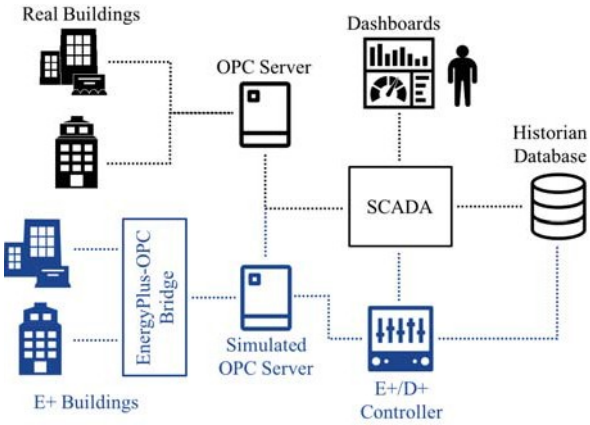


*Figure 1: A typical architecture of a SCADA system working with a real building is shown in black, and our contributions in blue.*

for efficient modeling and HVAC control of large scale buildings using machine learning, see Figure 1. We explain and demonstrate with examples our complete pipeline starting from tools for data acquisition from existing SCADA/BEMS to learning accurate control-oriented data-driven D+ models using Gaussian Processes (GP) to using D+ models for real-time predictive con- trol with high confidence. We use an EnergyPlus-OPC bridge that connects E+ models to the SCADA system through a simulated OPC server. We design a predictive controller using D+ models learned from the historical data obtained from a Historian database. For testing the controller with E+ models, the controller communicates via the OPC server. For testing the controller on a real building, secure and direct communication with SCADA is possible.

## DATA ACQUISITION & COMMUNICATION

In a building automation system, a Supervisory Control and Data Acquisition (SCADA) system is commonly used by the operators to manage individual buildings or a cam- pus of buildings. It interfaces directly with building sen- sors and controllers through open source protocols like BACNet or OPC. SCADA software also provides a dash- board interface for operators to view live or historical data feeds from sensors and an easy way for operators to change control setpoints remotely. Additionally, it may also offer a Historian database to store historical data val- ues for future reference or data analysis. See Figure 1 for an illustration of a typical SCADA system.

There are two main limiting factors in bridging digital twins with existing SCADA software. First, most SCADA software are self-contained and the features are limited to those provided by the vendor. Building operators cannot

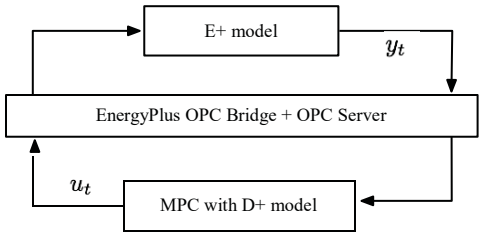


*Figure 2: The MPC controller runs in Python using the D+ models and applies the optimal inputs to the E+ model. The communication is made possible using the pyEplibrary and the OPC connection.*

view the results from E+/D+ models on the same SCADA software used for real-time monitoring because it is not clear how we can communicate between the E+/D+ mod- els and existing SCADA software to acquire the data tags required by E+/D+ models for simulation and control, and to show the generated results on the dashboards. Second, if the digital twin is an E+ model, we need an external li- brary like MLE+ (Bernal et al. 2012) to design a controller in a scripting language such as MATLAB. This is because EnergyPlus only allows manually coded rule-based con- trol strategies. Since Python is a popular programming language for data science and machine learning, we need an alternative to MLE+ for Python.

In this work, we use the open interface OPC to connect EnergyPlus with any existing SCADA software that sup- ports OPC for real-time data communication. We call this the EnergyPlus-OPC bridge. By representing inputs to and outputs from EnergyPlus as OPC tag structures, we make the integration into existing SCADA software sig- nificantly easier since the simulated building will appear as a real building to the SCADA software. Furthermore, since our machine learning models are written in Python, we develop a library to interface Python and EnergyPlus called pyEp, an equivalent to MLE+ for Python. We show how this library allows for intelligent control of buildings using D+ models and testing on E+ models. The transi- tion from testing on E+ models to testing on real buildings can be made seamlessly through the SCADA system.

The case studies presented later in this paper use the setup shown in Figure 2, in which a data-driven controller based on a D+ model acts on an E+ model as a plant. The EnergyPlus-OPC bridge allows us to interface an E+/D+ model to the OPC server and thus to the SCADA software for real-time monitoring and closed-loop control of the building. The setpoints obtained from the controller and the corresponding responses from the digital twin can be viewed in real time in a commercial SCADA dashboard.

**EnergyPlus-OPC Bridge**

Our EnergyPlus-OPC bridge provides the EnergyPlus in- put and output variables as OPC tags to be read by any OPC client. The user is able to configure the simulation for any number or type of buildings and can run each in- dividually on different schedules. To see the simulation in progress, the operators only need to view the tag using an OPC client, as they would for any other data source. By writing to one of the input tags, the operators can change the input setpoints to EnergyPlus and see the response of the building. For more advanced control of the building, operators can use our MPC controller based on D+ mod- els. The service supports the running of multiple Ener-
gyPlus instances, creating a campus of isolated buildings. The buildings are simulated synchronously, so that their simulations are always kept at the same time. This ca- pability can be useful when looking at aggregate power consumption and synthesizing control strategies involv- ing multiple-building curtailment. For communication be- tween EnergyPlus and Python we use pyEp.

**pyEp: A Python EnergyPlus Interface**

Currently, EnergyPlus supports external programs through the Building Controls Virtual Testbed (BCVTB), built on top of Ptolemy II, and the Functional Mockup Interface (FMI) standard. Using BCVTB, the users can couple and define data flows between various modeling and simulation programs, such as TRNSYS or Simulink. These simulation environments, while comprehensive, are still constrained by the capabilities of the software. MLE+ provides a solution to this problem, allowing the end-users to directly control the progress of an EnergyPlus simulation by writing MATLAB code. In recent years, Python has become a popular language for data science and machine learning, in academia and especially in industry. The pyEp library connects the myriad of Python libraries with existing technologies in the building modeling and simulation communities. For example, in our case studies depicted in Figure 2, we obtain data from EnergyPlus to learn D+ models, which are used for synthesizing building control strategies, then evaluate these strategies in closed-loop simulation with EnergyPlus. These steps are made possible by pyEp.

The pyEp library is designed to be lightweight and flexible. The core class is ep process, which pro- vides simple read and write capabilities with EnergyPlus. Each ep process instance corresponds to one Energy- Plus building, and is independent of all other ep process instances. This allows for multiple EnergyPlus models being run together in a campus-like co-simulation. An example using the Department of Energy (DoE) provided LargeOffice building model is included in the installation. For an EnergyPlus IDF model file to be used with pyEp, it

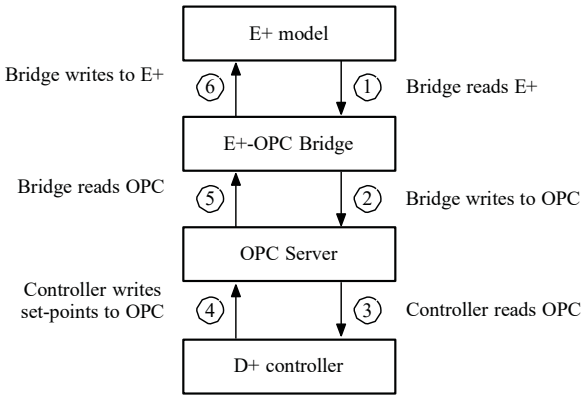


*Figure 3: Communication sequence for data exchange.*

must have the ExternalInterface configured, as well as an associated variables.cfg, specifying the inputs and outputs to the ExternalInterface. pyEp is available on the Python Package Index (PyPI) and can be installed with the command pip install pyEp. Its documentation can be found at https://github.com/mlab-upenn/pyEp.

**System Architecture**

The EnergyPlus-OPC bridge requires two processes to start and control a simulation over OPC. The first is the bridge itself, which can be started once and left in the background indefinitely. The second is a controller, which determines which setpoints to write to the bridge at what time during the simulation. The role of the bridge is to handle communication between EnergyPlus and the OPC server. The role of the controller is to control the inputs at every time step of the EnergyPlus co-simulation by writ- ing to the OPC server. The communication sequence for data exchange is shown in Figure 3. The same process fol- lows for the next building, until all have been incremented forward by one time step. The communication protocol ensures that every input and output are read to the cor- rect EnergyPlus building, and that delays in the network communications do not cause the controller and bridge to become out-of-sync with each other. The exchange of information is not real-time dependent, so human opera- tors can change the inputs time step by time step at any pace. The controller can also preemptively stop a simula- tion by terminating the controller process. Changes to a prescribed schedule can also be made, and the simulation restarted again without needing to restart the bridge pro- cess. This allows for faster and easier changes with less time overhead between simulation runs. Specific syntax can be found in the documentation. The bridge should only be restarted if different EnergyPlus buildings need to be used.

This bridge-controller design provides great flexibility in how the user can make use of EnergyPlus. Users can freely modify the controller to customize the simulation parameters. A simple schedule based controller can be made with basic knowledge of Python. Alternatively, more complex model-based controllers like MPC as in Section can also be implemented and evaluated. Two ex- ample controllers are included in the pyEp library. The first controller implements a setpoint schedule in Python and shows how to read/write from the controller to En- ergyPlus. The second controller implements a setpoint schedule based on a formatted csv file.

### Requirements

The provided controllers use the OpenOPC library to con- nect to an OPC server, but other methodologies may be used if the communications paradigm is followed. See pyEp documentation for more details. Additionally, the free Matrikon OPC Server Simulator is used as the de- fault server. Included with pyEp is a server configuration XML generator that automatically creates the correct OPC Tree Tag structure for the Matrikon Server. The pyEp core module, linking EnergyPlus to Python, is supported for Python 2.7 and 3.x, while the EnergyPlus-OPC bridge re- quires Python 2.7.

## MODELING WITH GAUSSIAN PROCESSES

As discussed in the Introduction section, conventional building modeling methods based on first principles are time-consuming and cost-prohibitive. Data-driven mod- eling approaches, based on machine learning techniques, have been shown to be fast, economical, and accurate al- ternatives. This section presents a building modeling ap- proach with Gaussian Process (GP), starting with a brief introduction to GP and its application in controls, and is adapted from (Jain et al. 2018a).

### Introduction to Gaussian Process

**Definition 1** *A Gaussian Process is a collection of ran- dom variables, any finite number of which have a joint Gaussian distribution.*

Consider an unknown function $f : \mathrm{R}^n \to \mathrm{R}$ with noisy ob- servations $y = f(x) + \mathcal{N}(0, \sigma^2)$. A GP of $y$ is specified by its mean function $\mu(x)$ and covariance function $k(x, x^t)$,

$$\mu(x;\theta) = \mathrm{E}[f(x)] \tag{1}$$
$$k(x, x^t;\theta) = \mathrm{E}[(f(x)-\mu(x))(f(x^t)-\mu(x^t))] + \sigma^2 \delta(x, x^t)$$

where $\delta(x, x^t)$ is the Kronecker delta function. These functions are parameterized by the hyperparameter vector $\theta$. The covariance function $k(x, x^t)$ specifies how the out- puts at $x$ and $x^t$ are correlated. In other words, a GP model of $y$ specifies the relationship between the input variables,

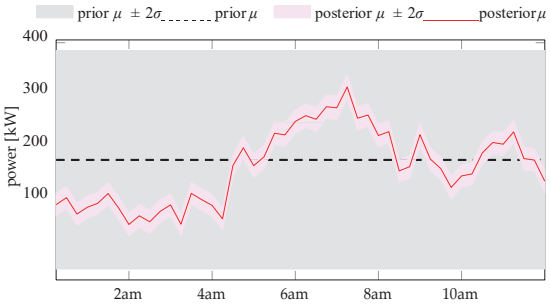


*Figure 4: Example of priors calculated using* (1) *and pos- teriors using* (2) *for predicting power consumption of a building for* 12 *hrs. Initially the mean is constant because* $\mu(x)$ *is constant, and we observe a high variance. The posterior agrees with the actual power consumption with high confidence.*

through the covariance function, rather than a fixed struc- tural input–output relationship from $x$ to $y$.

Given the regression vectors $X = [x_1, \ldots, x_N]^T$ and their corresponding observed outputs $Y = [y_1, \ldots, y_N]^T$, the dis- tribution of the output $y$ corresponding to a new input vector $x$ is a Gaussian distribution $N(\bar{y}, \sigma^2)$,

$$\bar{y} = g_m(x) := \mu(x) + K K^{-1}(Y - \mu(X)) \tag{2a}$$
$$\sigma^2 = g_v(x) := K - K K^{-1} K^T, \tag{2b}$$

where $K = [k(x, x_1), \ldots, k(x, x_N)]$, $K = k(x, x)$, and $K$ is the covariance matrix with elements $K_{ij} = k(x_i, x_j)$. The hyperparameters $\theta$ can be learned by maximizing the likelihood: $\mathrm{argmax}_\theta \Pr(Y|X, \theta)$. An example of GP prior and posterior is shown in Fig. 4.

GP models have several advantages over other machine learning models, that make them more suitable for identi- fication of dynamical systems.

1. GPs provide predictive variances, which carry uncer- tainty information of the predictions. The full predic- tive distribution, which includes both the mean and variance, can be used in a meaningful way, e.g., to estimate a 95% confidence bound for the prediction.

2. GPs work well with small data sets due to its stochas- tic nature, which is generally useful for any modeling application.

3. GPs allow incorporating domain knowledge of the system into the model to improve its accuracy, by defining priors on the hyperparameters or using a particular structure of the covariance function.

More details on GPs and their applications can be found in (Rasmussen and Williams 2006).

### Gaussian Processes for Dynamical Systems

Consider a nonlinear dynamical system with control input $u$, exogenous disturbance input $w$, and output $y$. At a current time step $t$, time-delayed input and output signals are called *autoregressive*, for example: $y_{t-1}$, $u_{t-1}$, $w_{t-1}$. By feeding autoregressive inputs and outputs to a GP as regressors, we can model the dynamic behavior of the system (Kocijan 2016). Specifically, the regressor vector $x_t$ at time step $t$ would be

$$x_t = [y_{t-l}, \ldots, y_{t-1}, u_{t-m}, \ldots, u_t, w_{t-p}, \ldots, w_{t-1}, w_t].$$

where $l$, $m$, and $p$ are respectively the lags for autoregressive outputs, control inputs, and disturbances. Note that $u_t$ and $w_t$ are the current control and disturbance inputs. The dynamical GP $y_t = f(x_t)$ can then be trained from data in the same way as any other GPs.

In a multistep simulation of a dynamical GP, the autoregressive outputs fed to the model beyond the first step are random variables, resulting in more and more complex output distributions as we go further. Therefore, it involves uncertainty propagation through the model, which would complicate the computation of the model significantly. (Nghiem and Jones 2017) showed that a simple simulation method called *zero-variance method*, which replaces the autoregressive signals with their corresponding expected values, could achieve sufficient prediction accuracy while benefitting from computational simplicity. In this paper, the zero-variance method was selected for predicting future outputs in optimization formulations.

### Modeling Building's Power Demand and Zone Temperatures with Gaussian Processes

We use a U.S. Department of Energy's Commercial Reference Building (DoE CRB) simulated in EnergyPlus as the virtual test-bed building. It is a large 12-story office building consisting of 19 zones with a total area of 498,588 sq.ft. Under peak load conditions the office can consume up to 1.4 MW of power. Our aim is to model the building's power demand and zone temperature behavior from data that can be measured directly from installed sensors such as thermostats, multimeters and weather forecast. We learn two GP models: $M_1$ for predicting the power demand and $M_2$ for predicting the temperature of a particular zone in the building. The following feature variables are used for training:

- *Weather variables* $d^w$*:* such as outside temperature and humidity, which are derived from historical weather data.
- *Proxy variables* $d^p$*:* such as time of day and day of week, which indicate occupancy and periodic trends.
- *Control variables u:* such as cooling, supply air temperature and chilled water setpoints – these variables will be optimized in MPC.
- *Output variable y:* total power consumption for $M_1$ and zone temperature for $M_2$ – this is the output of interest which we will predict using all the above features in the model.

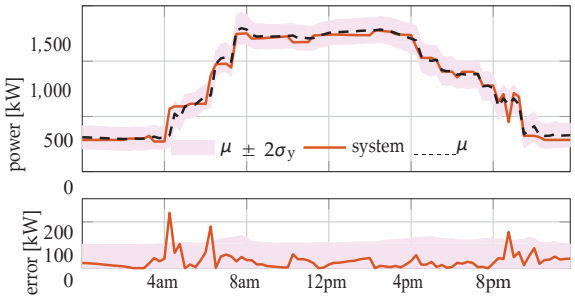


*Figure 5: Rolling forecast of the power demand for one particular day using GP model $M_1$. The actual demand is almost always within the 95% confidence interval.*

Using these measurement data, we learn autoregressive GP models $M_1$ and $M_2$, and use the zero-variance method to predict the future output $y_{t+\tau}$, where $t$ is the current time and $\tau \geq 0$. Specifically,

$$y_{t+\tau} \sim N\left(\bar{y}_{t+\tau} = g_m(x_{t+\tau}), \sigma^2_{y,t+\tau} = g_v(x_{t+\tau})\right), \quad (3)$$
$$x_{t+\tau} = [\bar{y}_{t+\tau-l}, \ldots, \bar{y}_{t+\tau-1}, u_{t+\tau-m}, \ldots, u_{t+\tau}, w_{t+\tau-p}, \ldots, w_{t+\tau-1}, w_{t+\tau}],$$

in which $w := [d^w, d^p]$. It is assumed that at time $t$, $w_{t+\tau}$ are available $\forall \tau$ from forecasts or fixed rules as applicable. For the GP, we use a constant mean function and the special covariance function proposed in our previous work (Nghiem and Jones 2017) to capture both the temporal pattern of the energy usage and the effect of non-temporal features, such as weather conditions and temperature setpoints, on the power demand. We optimize the hyperparameters $\theta$ of the GP model using GPML (Rasmussen and Nickisch 2010).

The mean prediction $\bar{y}$ with 95% confidence interval $\bar{y} \pm 2\sigma_y$ for a particular day are shown in Figure 5. With only 3 weeks of training data, we obtain a prediction accuracy of 94% in terms of normalized root mean square error (NRMSE) and an RMSE of 47kW for a building with peak demand 1400kW and mean power demand 816kW.

## MPC WITH GAUSSIAN PROCESSES

This section presents a data-driven predictive control approach for buildings using GPs. Suppose that GP models of a building's power demand and zone temperatures are already developed, as discussed in the previous section. The predicted power demand at any future time $t+\tau$

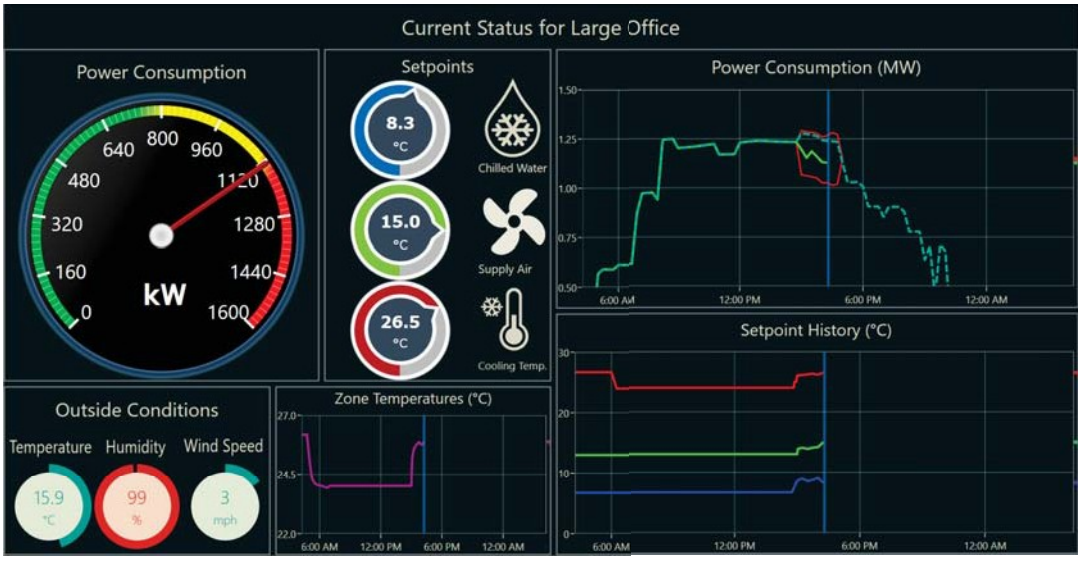


*Figure 6: Status of the LargeOffice as seen on SCADA system during Demand Tracking Control. The blue vertical line shows the current time of operation. Power consumption: the baseline power consumption is shown in dashed line, the curtailment is shown in solid green and the 95% confidence bounds are shown in solid red. Since a fixed curtailment is tracked without thermal comfort constraints in problem* (4)*, the zone temperature shoots up by* $2^{\circ}C$.

in a window of $N$ time steps starting from the current time $t$, for $\tau \in \{0,\ldots, N-1\}$, is given by Equation (3). The output at time $t+\tau$ depends upon the control inputs $u_{t+\tau-m},\ldots, u_{t+\tau}$. We next show the flexibility of using these models in different applications.

**Demand Tracking Control**

This predictive control formulation fits demand response applications where a customer – a building in this case – must curtail its power demand by a certain amount from its baseline, or must track an Area Control Signal (ACS) sent from the grid operator. For Demand Tracking Con- trol we only require GP model $M_1$. We are interested in solving the following optimization problem

$$\begin{aligned} \text{minimize} \quad & \sum_{\tau=0}^{N-1} (\bar{y}_{t+\tau} - y_{\text{ref},t+\tau})^2 + \lambda \sigma^2_{y,t+\tau} \qquad (4) \\ \text{subject to} \quad & \bar{y}_{t+\tau} = \mu(x_{t+\tau}) + K\ K^{-1}(Y - \mu(X)) \\ & \sigma^2_{t+\tau} = K\ \ - K\ K\ \ ^{-1}K^T \\ & u_{t+\tau} \in U \\ & \Pr(y_{t+\tau} \in Y) \geq 1-\varepsilon \end{aligned}$$

where the constraints hold for all $\tau \in \{0,\ldots, N-1\}$. Here, $K = [k(x_{t+\tau}, x_1),\ldots, k(x_{t+\tau}, x_N)]$, $K\ = k(x_{t+\tau}, x_{t+\tau})$. The signal $y_{\text{ref}}$ is a reference power demand trajectory the building should follow as closely as possible. The last constraint is probabilistic, which states that the building's power demand must stay inside a set $Y$ with a probability of at least $1-\varepsilon$. For instance, this constraint can control the quality of tracking the reference $y_{\text{ref}}$.

We solve the optimization problem (4) to compute optimal $u_t^*,\ldots, u^*_{\ t+N-1}$, apply $u_t^*$ to the system and proceed to time $t+1$. Although all the constraints in (4) are of analytical forms, the optimization can be computation- ally hard to solve due to the high nonlinearity and computational burden of the GP. We use the nonlinear solver IPOPT (Wächter and Biegler 2009) and the optimization modeling framework CasADi (Andersson 2013) to solve (4).

We show the real-time results on the SCADA system in Figure 6. The dashboard is configured to show the current power consumption, current setpoints, current weather conditions, a 12 hour history of the temperature of one of the zones (CoreMid), a 12 hour history of the setpoints – cooling temperature (red), supply air temperature (green), chiller water temperature (blue) – and a 12 hour history of the power consumption (solid green) along with a 12 hour forecast of the baseline consumption (dashed). In this example, the controller follows a rule-based strategy until 3 pm. A Demand Response is scheduled between 3-5 pm when D+ controller based on (4) is used. The controller provides a sustained curtailment of 100kW from the base- line. Since a fixed curtailment is desired without thermal comfort constraints, the zone temperature shoots up by $2^{\circ}C$.

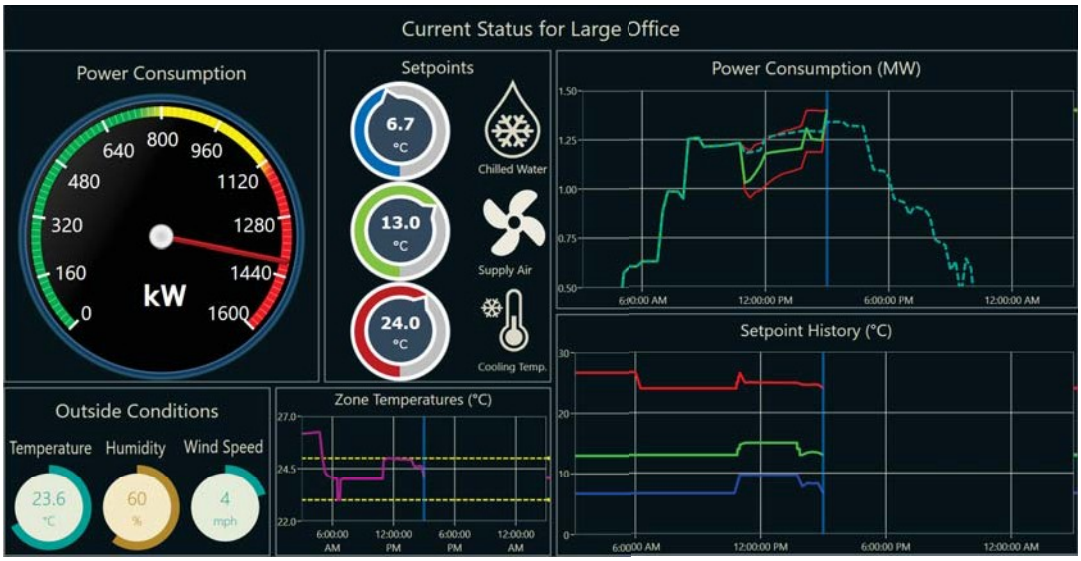


*Figure 7: Status of the LargeOffice as seen on SCADA system during Climate Control with Minimum Energy Usage. The blue vertical line shows the current time of operation. Power consumption: the baseline power consumption is shown in dashed line, the curtailment is shown in solid green and the 95% confidence bounds are shown in solid red. Zone temperature: the thermal comfort constraint keeps the temperature between a desired range* $23-25^{\circ}C$ *defined in the optimization problem* (5).

**Climate Control with Minimum Energy Usage**

Other interesting objective for energy management is of minimizing energy usage while meeting thermal comfort and operation constaints. This is always desired since the consumers want to minimize their electricity bills. When the real-time price of electricity peaks, the utilities may also ask their customers to minimize their consumption between a specified time based on the available flexibility. In this example, we require both GP models $M_1$ and $M_2$. Specifically, the model $M_2$ is used to enforce thermal comfort constraints in the optimization problem below:

$$\begin{aligned}
&\text{minimize} && \sum_{\tau=0}^{N-1} \bar{y}_{t+\tau} + \lambda \sigma^2_{y,t+\tau} && (5)\\
&\text{subject to} && \bar{y}_{t+\tau} = \mu(x_{t+\tau}) + K\ K^{-1}(Y-\mu(X)) && \text{GP } M_1\\
&&& \sigma^2_{t+\tau} = K\ -K\ K\ ^{-1}K^T \\
&&& \left.\begin{aligned}\bar{T}_{t+\tau} &= \mu(x_{t+\tau}) + K K^{-1}(Y-\mu(X))\\ \sigma^2_{T,t+\tau} &= K\ -K\ K\ ^{-1}K^T\end{aligned}\right\}\ \text{GP } M_2\\
&&& \Pr(T_{t+\tau} \in T) \geq 1-\varepsilon\\
&&& u_{t+\tau} \in U
\end{aligned}$$

Here, $\bar{y}, \sigma^2_y$ denote the mean and variance of the GP $M_1$, and $\bar{T}$, $\sigma^2_T$ of the GP $M_2$. The goal is to optimize the inputs $u$ that minimize the power consumption while maintaining thermal comfort with high probability. We again solve the optimization problem (5) to compute optimal $u_t, \ldots, u_{N-1}$, apply $u_{t+}$ to the system and proceed to time $t+1$.

We consider a Demand Response scenario when the utility company notifies this building to minimize the power consumption between 11am - 2pm. Between 2pm - 3pm, the controller switches to tracking the baseline consump- tion based on (4) to prevent a sudden kickback. The sta- tus of the SCADA system at 3pm is shown in Figure 7. The controller exploits the available energy flexibility in providing maximum curtailment while ensuring that the zone temperature is always between the prescribed range $23-25^{\circ}C$.

## CONCLUSION

This paper presented a set of methods and tools to incorporate "digital twins" of real buildings into existing SCADA systems. We proposed a data-driven modeling approach using Gaussian Process to quickly and inexpensively capture a model of a building solely from its measurement data. This type of models, called D+ models, together with EnergyPlus (E+) models of buildings serve as "digital twins" of the buildings. In addition to requiring significantly lower cost and effort to develop, compared to E+ models, D+ models are more suitable for Model Predictive Control (MPC) and more adaptive to changes in the buildings. Two applications of MPC with D+ models were formulated for demand-tracking control and climate

control with minimum energy.

We developed an EnergyPlus-Python bridge, called pyEp, to interface Python code with EnergyPlus to perform co-simulations, which is useful for implementing ad- vanced algorithms such as machine-learning-based mod- eling and optimization-based control. We also devel- oped an EnergyPlus-OPC bridge, which completes our toolchain for integrating E+ and D+ models into SCADA systems. Through OPC tag mappings, these digital twins can directly exchange data with a SCADA system, receiv- ing control commands and returning measurement val- ues as if they were real buildings. For the first time, our toolchain has enabled seamless real-time in-the-loop pre- diction and advanced control of both software buildings and physical buildings within the same SCADA environ- ment. The toolchain was demonstrated in a case study, which showed the effectiveness of both our software and our proposed data-driven MPC approach for buildings.

## REFERENCES


Andersson, Joel. 2013. "A General-Purpose Software Framework for Dynamic Optimization." PhD thesis, KU Leuven.

Behl, Madhur, Truong X. Nghiem, and Rahul Mangharam. 2015. "DR-Advisor: A Data Driven Demand Response Recommender System." *Proceedings of CISBAT*.

Bernal, Willy, Madhur Behl, Truong X. Nghiem, and Rahul Mangharam. 2012. "MLE+: a tool for integrated design and deployment of energy efficient building controls." *Proceedings of ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings (BuildSys)*. ACM, 123–130.

Deng, Kun, Prabir Barooah, Prashant G Mehta, and Sean P Meyn. 2010. "Building thermal model reduction via aggregation of states." *American Control Conference (ACC)*. IEEE, 5118–5123.

Deru, Michael, Kristin Field, Daniel Studer, et al. 2011. US Department of Energy commercial reference building models of the national building stock.

Jain, Achin, Madhur Behl, and Rahul Mangharam. 2017. "Data Predictive Control for building energy management." *Proceedings of the 2017 American Control Conference*. IEEE.

Jain, Achin, Truong X. Nghiem, Manfred Morari, and Rahul Mangharam. 2018a. "Learning and Control using Gaussian Processes." *Proceedings of the 9th International Conference on Cyber-Physical Systems (ICCPS)*. ACM/IEEE.

Jain, Achin, Francesco Smarra, Madhur Behl, and Rahul Mangharam. 2018b. "Data-Driven Model Predictive Control with Regression Trees—An Application to Building Energy Management." *ACM Transactions on Cyber-Physical Systems* 2 (1): 4.

Jain, Achin, Francesco Smarra, and Rahul Mangharam. 2017. "Data Predictive Control using Regression Trees and Ensemble Learning." *Proceedings of the 2017 Conference on Decision and Control*. IEEE.

Kocijan, Juš. 2016. *Modelling and control of dynamic systems using Gaussian process models*. Springer.

Nghiem, Truong X., and Colin N. Jones. 2017. "Data-driven Demand Response Modeling and Control of Buildings with Gaussian Processes." *Proceedings of American Control Conference (ACC)*.

Rasmussen, Carl Edward, and Hannes Nickisch. 2010. "Gaussian processes for machine learning (GPML) toolbox." *Journal of Machine Learning Research* 11 (Nov): 3011–3015.

Rasmussen, Carl Edward, and Christopher KI Williams. 2006. *Gaussian processes for machine learning*. Volume 1. MIT press Cambridge.

Smarra, Francesco, Achin Jain, Tullio de Rubeis, Dario Ambrosini, Alessandro D'Innocenzo, and Rahul Mangharam. 2018. "Data-driven model predictive control using random forests for building energy optimization and climate control." *Applied Energy*.

Sturzenegger, David, Dimitrios Gyalistras, Manfred Morari, and Roy S Smith. 2016. "Model Predic- tive Climate Control of a Swiss Office Building: Implementation, Results, and Cost–Benefit Analysis." *IEEE Transactions on Control Systems Technology* 24 (1): 1–12.

University of Wisconsin Madison. 1975. TRNSYS, a Transient Simulation Program.

Wächter, A, and L Biegler. 2009. IPOPT-an interior point OPTimizer.

Žáčeková, Eva, Zdeněk Váňa, and Jiří Cigler. 2014. "Towards the real-life implementation of MPC for an of- fice building: Identification issues." *Applied Energy* 135:53–62.