Data Predictive Control for building energy management

Achin Jain[1], Madhur Behl[2] and Rahul Mangharam[1]

Abstract— **Decisions on how to best optimize energy systems operations are becoming ever so complex and conflicting, that model-based predictive control (MPC) algorithms must play an important role. However, a key factor prohibiting the widespread adoption of MPC in buildings, is the cost, time, and effort associated with learning first-principles based dynamical models of the underlying physical system. This paper introduces an alternative approach for implementing finite-time receding horizon control using control-oriented data-driven models. We call this approach Data Predictive Control (DPC). Specifically, by utilizing separation of variables, two novel algorithms for implementing DPC using a single regression tree and with regression trees ensembles (random forest) are presented. The data predictive controller enables the building operator to trade off energy consumption against thermal comfort without having to learn white/grey box models of the systems dynamics. We present a comprehensive numerical study which compares the performance of DPC with an MPC based energy management strategy, using a single zone building model. Our simulations demonstrate that performance of DPC is comparable to an MPC controller, with only 3.8% additional cost in terms of optimal objective function and within 95% in terms of $\mathbf{R^2}$ score, thereby making it an alluring alternative to MPC, whenever the associated cost of learning the model is high.**

I. INTRODUCTION

Control-oriented predictive models of an energy system's dynamics and energy consumption, are needed for understanding and improving the overall energy efficiency and operating costs. With a reasonably accurate forecast of future weather and building operating conditions, dynamical models can be used to predict the energy needs of the building over a prediction horizon, as is the case with Model Predictive Control (MPC) [11]. However, a major challenge with MPC is in accurately modeling the dynamics of the underlying physical system. The task is much more complicated and time consuming in case of a large building and often times, it can be even more complex and involved than the controller design itself. After several years of work on using first principles based models for peak power reduction, and energy optimization for buildings, multiple authors [11], [13] 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. The user expertise, time, and associated sensor costs required to develop a model of a single building is very high. This is because a building modeling domain expert typically uses a software tool to create the geometry of a building from the building design and equipment layout plans, add detailed information about material properties, about equipment and operational schedules. There is always a gap between the modeled and the real building and the domain expert must then manually tune the model to match the measured data from the building [10]. Moreover, the modeling process also varies from building to building with the construction and types of installed equipment. Another major downside with physics-based modeling is that enough data is not easily available and guesses for parameter values have to be made, which also requires expert know how.

The alternative is to use black-box, or completely data-driven modeling approaches, to obtain a realization of the system's input-output behavior. The primary advantage of using data-driven methods is that it has the potential to eliminate the time and effort required to build white and grey box building models. Listening to real-time data, from existing systems and interfaces, is far cheaper than unleashing hoards of on-site engineers to physically measure and model the building. Improved building technology and better sensing is fundamentally redefining the opportunities around smart buildings. Unprecedented amounts of data from millions of smart meters and thermostats installed in recent years has opened the door for systems engineers and data scientists to analyze and use the insights that data can provide, about the dynamics and power consumption patterns of these systems.

The challenge now, with using data-driven approaches, is to close the loop for real-time control and decision making for both small and large scale buildings. We address these challenges by introducing an alternative approach (to grey-box MPC) for finite receding horizon control of building energy systems using data-driven control oriented models. We call this *Data Predictive Control*. While still being model based, DPC involves using scalable and interpretable models for the building's dynamics. In particular, we utilize modified regression trees and regression trees ensembles to implement such control.

In our previous work [7], we developed and evaluated DPC using multi-output regression trees as predictive models. In this paper, we present two new approaches with significant improvements. This work has the following contributions:

1) We address the limitations of our previous work, and present a data predictive control with single-output regression trees (DPC-RT) algorithm for finite receding horizon control. DPC-RT bypasses the cost and time prohibitive process of building high fidelity models of buildings that use grey and white box modeling ap-

[1]Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104, USA {achinj, rahulm}@seas.upenn.edu

[2]Department of Computer Science, University of Virginia, Charlottesville, VA 22903, USA madhur.behl@virginia.edu

This work was supported in part by TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA.

proaches while still being suitable for receding horizon control design (like MPC).

2) While DPC-RT provides comparable performance to a MPC controller, we extend the algorithm to work with an ensemble of regression trees. The ensemble data predictive control, (DPC-En) is the first such method to bridge the gap between ensemble predictive models (such as random forests) and receding horizon control.

We present a comprehensive case study to demonstrate how DPC can achieve comparable performance as MPC but without utilizing a dynamical model of the system. We begin with description of a realistic building model used for our case study in Sec. II. Sec. III defines the finite receding horizon control problem with MPC framework. Sec. IV describes the training and control algorithms for DPC with regression trees and random forests along with model validation. We compare the performance of DPCRT to MPC and discuss the challenges associated with DPC in Sec. V. We conclude the paper with a summary of the results and a brief discussion on the future work in Sec. VI.

II. MODELING

For testing our algorithms, we use a realistic model of the building obtained from the HAMLab ISE tool [12]. HAMLab is an all-in-one collection of models and tools suitable for MatLab and/or Simulink. It provides a library of realistic building and equipment models. The model under consideration is linear and uses a state-space representation. It captures the essential dynamics governing the zone-level operation while considering external and internal thermal disturbances. The building in question is a single zone building, the parameters of which are identified by finding an equivalent RC network, the schematic of which is shown in Fig. 1.

The model has 4 states: floor temperature T_{fl}, internal facade temperature T_{if}, external facade temperature T_{ef} and internal zone temperature T_{in} such that $x := [\mathsf{T}_{\mathrm{fl}}, \mathsf{T}_{\mathrm{if}}, \mathsf{T}_{\mathrm{ef}}, \mathsf{T}_{\mathrm{in}}]^T$, 1 control input in the form of heat (in Watts) rejected/added in the zone Q_{in} such that $u := \mathsf{Q}_{\mathrm{in}}$, and 3 uncontrollable inputs or disturbances: external temperature T_{ex}, internal heat gain due to occupancy Q_{oc} and solar heat radiation Q_{sl} such that $d := [\mathsf{T}_{\mathrm{ex}}, \mathsf{Q}_{\mathrm{oc}}, \mathsf{Q}_{\mathrm{sl}}]^T$. The mathematical model can be represented as

$$x_{k+1} = Ax_k + Bu_k + B_d d_k \qquad (1)$$
$$k = 0, \ldots, T$$

where A, B and B_d are time-invariant state space matrices calculated at sampling time $T_s = 300\,\mathrm{s}$. The input u is constrained between $\underline{u} = -500\,\mathrm{W}$ (where negative values denote cooling) and $\bar{u} = 1000\,\mathrm{W}$ and the states x between $\underline{x} = -30\,^{\circ}\mathrm{C}$ and $\bar{x} = 50\,^{\circ}\mathrm{C}$. ISE also provides 30 years (1971-2000) of hourly data for the atmospheric disturbances, which is an estimate of disturbances using actual meteorological data. It is assumed that the disturbance vector d is precisely known for the purpose of simulations in Sec. V. In our future work, we will also address the problem of disturbance uncertainty. For more details on physical modeling and value of identified parameters that form A, B and B_d, we refer the reader to [12].

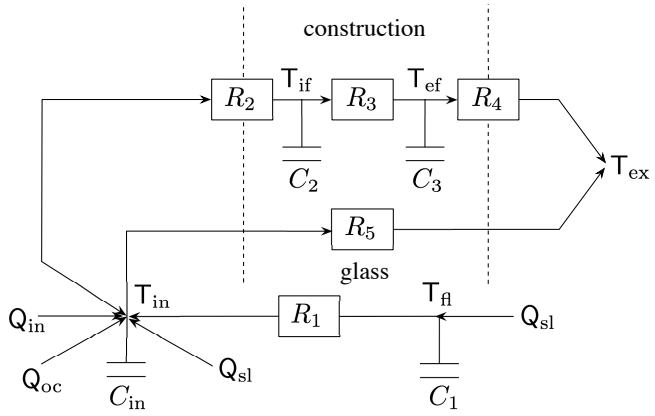


Fig. 1: RC network representation of the building model [12].

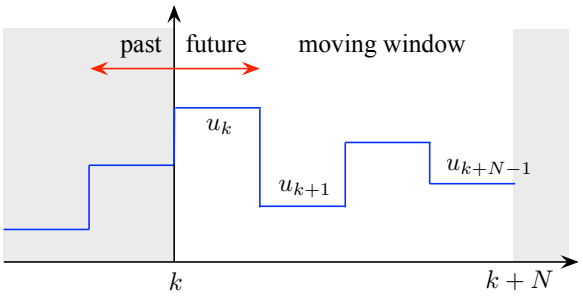


Fig. 2: Finite-horizon moving window of MPC: at time k, the MPC optimization problem is solved for a finite length window of N steps and the first control input u_k is applied; the window then recedes one step forward and the process is repeated at time $k+1$.

The results in this paper are presented for this single zone, however, the algorithms described next are easily scalable to multiple zones. In [1], we have successfully modeled a 12 storey, 70 zone building with our data-driven algorithms. The focus of this work is on comparing the performance of DPC with MPC and using a single zone suffices for that comparison as it eliminates any concomitants in the performance comparison.

III. MODEL PREDICTIVE CONTROL

We use a finite receding horizon MPC controller as a benchmark for comparison. The finite receding horizon control (RHC) approach involves optimizing a cost function subject to the dynamics of the system and the constraints, over a finite horizon of time [9]. After an optimal sequence of control inputs are computed, the first input is applied, then at the next step the optimization is solved again as shown in Fig. 2.

MPC for the same single zone model has been previously implemented in [4]. In this paper, we use MPC for comparison against the DPC algorithm by defining the following objective function. The objective of the controller

(supervisory) is to minimize the energy usage, i.e Q_{in} while maintaining a desired level of thermal comfort. Therefore, at time step k, we want to determine the optimal sequence of inputs $[\mathsf{Q}_{\mathrm{in,k}}, \ldots, \mathsf{Q}_{\mathrm{in,k+N-1}}]$ that satisfies

$$\begin{aligned} \text{minimize} \quad & \sum_{j=0}^{N-1} \mathsf{Q}_{\mathrm{in,k+j}}^2 + \lambda \sum_{j=1}^{N} (\mathsf{T}_{\mathrm{in,k+j}} - \mathsf{T}_{\mathrm{ref}})^2 \\ \text{subject to} \quad & x_{k+j} = Ax_{k+j-1} + Bu_{k+j-1} + B_d d_{k+j-1} \\ & \underline{\mathsf{Q}}_{\mathrm{in}} \leq \mathsf{Q}_{\mathrm{in,k+j-1}} \leq \bar{\mathsf{Q}}_{\mathrm{in}} \\ & \underline{\mathsf{T}}_{\mathrm{in}} \leq \mathsf{T}_{\mathrm{in,k+j}} \leq \bar{\mathsf{T}}_{\mathrm{in}} \\ & j = 1, \ldots, N \end{aligned} \tag{2}$$

where λ is a tuning parameter and $\mathsf{T}_{\mathrm{ref}}$ is the reference zone temp that maintains thermal comfort. The parameter λ helps trade-off energy savings against discomfort.

IV. DATA PREDICTIVE CONTROL

Our goal is to find data-driven functional models that relates the value of the response variable (i.e. zone temperature, in this case) to control inputs and disturbances. When the data has lots of features, as is the case in large buildings, which interact in complicated, nonlinear ways, assembling a single global model, such as linear or polynomial regression, can be difficult, and can lead to poor response predictions. An approach to non-linear regression is to partition the data space into smaller regions, where the interactions are more manageable. We then partition the partitions again; this is called recursive partitioning, until finally we get to chunks of the data space which are so tame that we can fit simple models to them. Therefore, the global model has two parts: the recursive partition, and a simple model for each cell of the partition. Regression trees is an example of an algorithm which belongs to the class of recursive partitioning algorithms. The seminal algorithm for learning regression trees is CART as described in [3].

The primary reason for this modeling choice is that regression trees are highly interpretable, by design. Interpretability is a fundamental desirable quality in any predictive model. Complex predictive models like neural-networks, support vector regression etc. go through a long calculation routine and involve too many factors. It is not easy for a human engineer to judge if the operation/decision is correct or not or how it was generated in the first place. Building operators are used to operating a system with fixed logic and rules. They tend to prefer models that are more transparent, where it is clear exactly which factors were used to make a particular prediction. At each node in a regression tree a simple, *if this then that*, human readable, plain text rule is applied to generate a prediction at the leafs, which anyone can easily understand and interpret. Making machine learning algorithms more interpretable is an active area of research, one that is essential for incorporating human centric models for building energy management.

The central idea behind DPC is to obtain control-oriented models using machine learning, and formulate the control problem in a way that RHC can still be applied and the optimization problem (2) can be solved efficiently. In Sec. IV-B, we describe our training algorithm where we use separation of variables to fit models on controllable and uncontrollable variables separately. In Sec. IV-C and IV-D, we present two algorithms for implementation of DPC, namely DPC-RT which utilizes a single regression tree, and DPC-En which uses an ensemble of regression trees as the underlying model.

A. Training Data

We simulate the HAMLab model given by (1) with a simple rule based strategy for a period of 3 months (May'99 - July'99) so that we obtain a rich enough training data set. The forecast of disturbances T_{ex}, Q_{oc} and Q_{sl} are obtained from the ISE data base. Heat due to occupancy Q_{oc} is a discrete variable which is defined to be $500\,\mathrm{W}$ from $8\,\mathrm{am}$ to $6\,\mathrm{pm}$ and $0\,\mathrm{W}$ otherwise. An example of disturbances is shown in Fig. 6(a).

In order to build regression trees we need to train on time-stamped historical data. The following feature variables are used for training the model:

1) **Weather Data** W**:** This includes measurements of the outside air temperature T_{ex}, solar radiation Q_{sl}. Since we are interested in predicting the power consumption for a finite horizon, we include the weather forecast of the complete horizon in the training features.
2) **Building Data** B**:** The state of the building is given by the zone air temperature T_{in}. This is typically known from temperature sensors in the building. We use autoregressive (lagged) terms of T_{in} as features and the future prediction(s) of T_{in} as the response variable(s). The heat gain into a zone due to occupancy Q_{oc} is also recorded and used for model training.
3) **Controller Data** C**:** This includes current and future control actions Q_{in}.

We learn a model f which predicts future zone air temperature given current and future weather predictions, occupancy heat gain, and past zone air temperature:

$$\begin{pmatrix} \mathsf{T}_{\mathrm{in,k+1}} \\ \mathsf{T}_{\mathrm{in,k+2}} \\ \vdots \\ \mathsf{T}_{\mathrm{in,k+N}} \end{pmatrix} = f(\mathsf{W}_k, \ldots, \mathsf{W}_{k+N-1}, \mathsf{T}_{\mathrm{in,k}}, \ldots, \mathsf{T}_{\mathrm{in,k-\delta}}, \mathsf{C}_k, \ldots, \mathsf{C}_{k+N-1}), \tag{3}$$

where δ is the order of autoregression, or in shorthand notation $\mathsf{Y} = f\left(\mathsf{X}^1, \ldots, \mathsf{X}^n\right)$ with n being the number of features. Note that $\mathsf{T}_{\mathrm{in,k}}$, and C_k in (3) are same as 4^{th} component of x_k and u_k in (2), respectively. It is assumed that perfect forecasts of disturbances for the entire length of the horizon is available to both MPC and DPC. In reality, the forecasts also have an associated uncertainty but analyzing its effect of control performance is a future research direction and not the focus of this work.

B. Training Algorithm

Even though it is possible to directly learn a model based on (3) using algorithms like regression trees, random forests

or neural networks, the resulting models are not suitable for optimization in (2) because of multiple reasons. First, the gradient is not defined for such models so we may have to settle with sub-optimal solution using evolutionary algorithms [8]. Second, because of autoregression, it will lead to state space explosion.

The next step is to modify the regression trees and make them suitable for synthesizing the optimal values of the control variables in real-time. Given building data (X, Y), we can separate the controllable (or manipulated) variables X_c and uncontrollable (or non-manipulated / disturbance) variables X_d in the feature set such that $\mathsf{X}_c \cup \mathsf{X}_d \equiv \mathsf{X}$. Applying this separation of variables, the regression trees are learned only on the non-manipulated variables or disturbances $(\mathsf{X}_d, \mathsf{Y})$. We obtain a model in the following form:

$$\mathsf{Y} = f_{\text{tree}}\left(\mathsf{X}_d^1, \ldots, \mathsf{X}_d^n\right). \tag{4}$$

In the leaf R_i of the trees, we fit a linear parametric model which is a function only of the controllable/manipulated variables:

$$\mathsf{Y}_{R_i} = \beta_i^T \left[1, \mathsf{X}_c\right]^T. \tag{5}$$

Any parametric regression model is valid in the leaves. We choose linear models because they simplify the optimization problem when added as constraints in (7). We validate this assumption in Sec. IV-E.

In this manner, we train the regression trees using only X_d, and then in each leaf we fit a parametric (linear) model which is a function only of X_c. At run time, to solve the control problem when only X_d is known, we navigate to an appropriate leaf R_i and use the linear constraint in the optimization problem. In the following section, we demonstrate how to define this optimization problem when we have a separate regression tree for each prediction step $k+1$ to $k+N$. In Sec. IV-D we extend this to an ensemble of regression trees (random forest) to decrease the variance in predictions.

C. DPC-RT: DPC with Regression Trees

To replace dynamics constraints (1) in (2), we build N trees to predict the states of the system, i.e. $\mathcal{T}_j$ is used to predict $\mathsf{Y}^j := \mathsf{T}_{\text{in,k+j}}$ where $j = \{1, \ldots, N\}$. While each tree is trained only on X_d, in the i^{th} leaf of the j^{th} tree we fit a linear model:

$$\mathsf{Y}_{R_i}^j = \beta_i^T \left[1, \mathsf{X}_c^k, \ldots, \mathsf{X}_c^{k+j-1}\right]^T. \tag{6}$$

Eq. (6) implies that the prediction of zone temperature at time $k+j$ is an affine combination of control inputs from time k to $k+j-1$. Each tree contributes to a linear constraint in the optimization problem. Thus, the DPC-RT counterpart of (2) becomes

$$\begin{aligned} \text{minimize} \quad & \sum_{j=0}^{N-1} \mathsf{Q}_{\text{in,k+j}}^2 + \lambda \sum_{j=1}^{N} \left(\mathsf{T}_{\text{in,k+j}} - \mathsf{T}_{\text{ref}}\right)^2 \\ \text{subject to} \quad & \mathsf{T}_{\text{in,k+j}} = \beta_i^T \left[1, \mathsf{Q}_{\text{in,k}}, \ldots, \mathsf{Q}_{\text{in,k+j-1}}\right]^T \\ & \underline{\mathsf{Q}}_{\text{in}} \leq \mathsf{Q}_{\text{in,k+j-1}} \leq \bar{\mathsf{Q}}_{\text{in}} \\ & \underline{\mathsf{T}}_{\text{in}} \leq \mathsf{T}_{\text{in,k+j}} \leq \bar{\mathsf{T}}_{\text{in}} \\ & j = 1, \ldots, N. \end{aligned} \tag{7}$$

We solve this optimization in the same manner as finite receding horizon control to determine $[\mathsf{Q}_{\text{in,k}}, \ldots, \mathsf{Q}_{\text{in,k+N-1}}]$, choose the first control input $\mathsf{Q}_{\text{in,k}}$ and proceed to the next time step $k+1$.

Our algorithm for DPC with regression trees is summarized in Algo. 1 and a schematic is shown in Fig. 3(a). During training process, trees are trained only on uncontrollable variables with linear models in the leaves which are a function only of controllable variables. During control step, at time k, the uncontrollable features $\mathsf{X}_d(k)$ are known and thus the leaf $R_i(k)$ of each tree is known. The linear models in $R_i(k)$ act as constraints in optimization problem. From the control action $[\mathsf{X}_c^*(k), \ldots, \mathsf{X}_c^*(k+N-1)]^T$, the first input $\mathsf{X}_c^*(k)$ is applied to the system. The resulting output $\mathsf{Y}(k)$ which is a feature for the next time step is fed back to determine $\mathsf{X}_d(k+1)$.

Algorithm 1 Data Predictive Control with Regression Trees

```
DESIGN TIME
procedure MODEL TRAINING USING SEPARATION OF VARIABLES
    Set X_c ← manipulated features
    Set X_d ← non-manipulated features
    Build N predictive trees T_j with (Y^j, X_d) using (4)
    for all trees T_j do
        for all regions R_i at the leaves of T_j do
            Fit Y^j_{R_i} = β_i^T [1, X_c^k, ..., X_c^{k+j-1}]^T
        end for
    end for
end procedure
RUN TIME
procedure PREDICTIVE CONTROL
    while t < t_stop do
        Determine the leaf and region R_i(k) using X_d(k)
        Obtain the linear model at R_i(k)
        Solve optimization in (7) to determine optimal
        control action [X_c^*(k), ..., X_c^*(k+N-1)]^T
        Apply the first input X_c^*(k)
    end while
end procedure
```

D. DPC-En: DPC with Ensemble Models

Regression trees obtain good predictive accuracy in many domains. However, the simple models used in their leaves have some limitations regarding the kind of functions they are able to approximate. The problem with trees is their high

variance and that they can over fit the data. It is the price to be paid for estimating a simple, tree-based structure from the data. Often a small change in the data can result in a different series of splits. The main reason behind this is the hierarchical nature of the process: the effect of an error in the top split is propagated down to all of the splits below it. While pruning and cross validation can help reduce over fitting, we use ensemble methods for growing more accurate trees.

The goal of ensemble methods is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability and robustness over a single estimator. Random forests or *tree-bagging* are a type of ensemble method which makes predictions by averaging over the predictions of several independent base models. The essential idea is to average many noisy but approximately unbiased trees, and hence reduce the variance. Injecting randomness into the tree construction can happen in many ways. The choice of which dimensions to use as split candidates at each leaf can be randomized, as well as the choice of coefficients for random combinations of features. Another common method for introducing randomness is to build each tree using a bootstrapped or sub-sampled data set. In this way, each tree in the forest is trained on slightly different data, which introduces differences between the trees. More explicitly, training features X^p with $p < n$ and the data itself (in-bag samples) are different for each tree in the forest. Not all estimators can be improved by shaking up the data like this. However, highly nonlinear estimators, such as trees, benefit the most. For a more comprehensive review we refer the reader to [5].

The main idea here is to replace each tree in Algo. 1 by a forest

$$\mathsf{Y} = f_{\text{forest}}\left(\mathsf{X}_d^1, \ldots, \mathsf{X}_d^n\right) \tag{8}$$

which, again, is trained only on X_d, and then fit a linear regression model using X_c in every leaf of every tree of every forest. We build N forests for N prediction steps such that the forest $\mathcal{R}_j$ uses a linear model

$$\mathsf{Y}_{R_i}^j = \Theta_i^T \left[1, \mathsf{X}_c^k, \ldots, \mathsf{X}_c^{k+j-1}\right]^T \tag{9}$$

in the leaf R_i of every tree, where $j = \{1, \ldots, N\}$. With a slight abuse of notation, here $(\mathsf{X}_c, \mathsf{Y})$ correspond to the in-bag samples (in-bag samples correspond to the data samples on which the tree was trained) for the trees.

While the offline training burden in DPC-En is slightly increased compared to DPC-RT, in the control step we exploit the better accuracy, and lower variance properties of the random forest. If a forest has t number of trees, given the forecast of disturbances, we have t set of linear coefficients. We simply average out all the coefficients from all the trees to get one linear model represented by $\hat{\Theta}_i$ for each forest. Note that the averaging step can only be done in run-time because the leaf of each tree can be narrowed down only when X_d is known. Thus, for N forests, we again have exactly N linear

Algorithm 2 Data Predictive Control with Random Forests

```
RUN TIME
procedure PREDICTIVE CONTROL
    while t < t_stop do
        for all forests do
            Determine the leaves R_i(k) using X_d(k)
            Obtain all linear models at R_i(k)
            Average out the linear coefficients Θ̂_i
        end for
        Solve optimization in (10) to determine optimal
        control action [X_c*(k), ..., X_c*(k+N-1)]^T
        Apply the first input X_c*(k)
    end while
end procedure
```

equality constraints in the optimization problem below.

$$\begin{aligned} \text{minimize} \quad & \sum_{j=0}^{N-1} \mathsf{Q}_{\text{in,k+j}}^2 + \lambda \sum_{j=1}^{N} \left(\mathsf{T}_{\text{in,k+j}} - \mathsf{T}_{\text{ref}}\right)^2 \\ \text{subject to} \quad & \mathsf{T}_{\text{in,k+j}} = \hat{\Theta}_i^T \left[1, \mathsf{Q}_{\text{in,k}}, \ldots, \mathsf{Q}_{\text{in,k+j-1}}\right]^T \\ & \underline{\mathsf{Q}}_{\text{in}} \leq \mathsf{Q}_{\text{in,k+j-1}} \leq \bar{\mathsf{Q}}_{\text{in}} \\ & \underline{\mathsf{T}}_{\text{in}} \leq \mathsf{T}_{\text{in,k+j}} \leq \bar{\mathsf{T}}_{\text{in}} \\ & j = 1, \ldots, N. \end{aligned} \tag{10}$$

Algo. 2 summarises the control step. The ensemble data predictive control, (DPC-En) is the first such method to bridge the gap between ensemble predictive models (such as random forests) and receding horizon control.

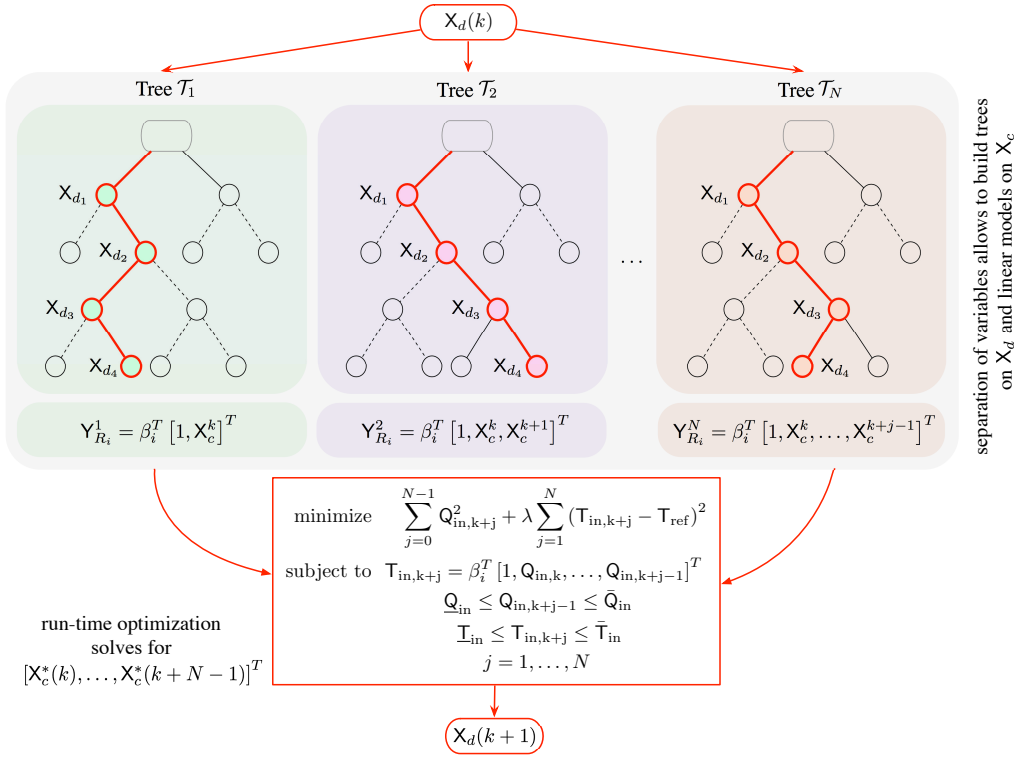
E. Validation

For horizon length $N = 6$ and order of autoregression $\delta = 6$, we run a simulation on June 1, 2000 using a random sequence of inputs. Since all disturbances as well as control inputs are known, we can predict the zone temperature $\mathsf{T}_{\text{in,k+1}}$ to $\mathsf{T}_{\text{in,k+6}}$ at every time k using both trees and ensembles. We have shown these 6 predictions at different times in the simulation in Fig. 4. At time k_1, we make next 6 predictions of zone temperature and compare it to the actual zone temperature that is obtained by applying the same control input. We repeat this process at k_2 to k_6. The normalized root mean-squared error (NRMSE) obtained with single regression trees and regression tree ensembles are shown in top and bottom, respectively. Both achieve close to 99% accuracy, but on an average, predictions with ensembles are more accurate and have a lower variance.

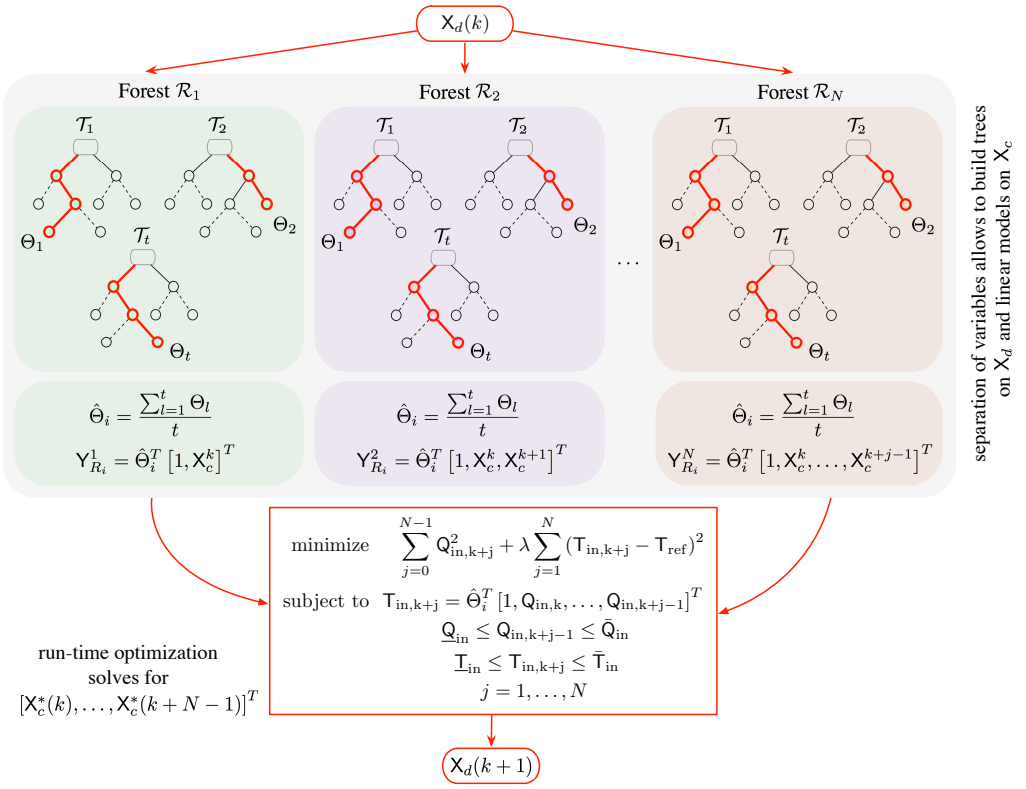
The absolute residual error in the predictions of $\mathsf{T}_{\text{in,k+1}}$ for full day is shown in Fig. 5 which again shows that random forest are more accurate. The absolute error for ensembles is bounded by $\pm 0.4\,^{\circ}\text{C}$ while for single trees it is $\pm 0.8\,^{\circ}\text{C}$.

V. COMPARATIVE STUDY

We now run DPC and MPC controller in a closed-loop simulation with the HAMLab plant model. The objective function and the box constraints on input and states are



(a) DPC-RT: At time k, the algorithm uses the forecast of disturbances $\mathsf{X}_d(k)$ to select linear models β_i in the leaves of each tree. Each linear model is added as a constraint in the optimization problem which calculates optimal sequence $[\mathsf{X}_c^*(k), \ldots, \mathsf{X}_c^*(k+N-1)]^T$, of which the first one is applied, and $\mathsf{X}_d(k+1)$ is calculated to proceed to $k+1$.



(b) DPC-En: At time k, the algorithm uses the forecast of disturbances $\mathsf{X}_d(k)$ to select linear models Θ_1 to Θ_t in the leaves of each ensemble. The linear models in each ensemble are averaged to calculate a single model represented by $\hat{\Theta}_j$ which act as constraints in the optimization problem. Again, the optimal sequence $[\mathsf{X}_c^*(k), \ldots, \mathsf{X}_c^*(k+N-1)]^T$, of which the first one is applied, and $\mathsf{X}_d(k+1)$ is calculated to proceed to $k+1$.

Fig. 3: Run-time algorithm for data predictive control with regression trees and tree ensembles.

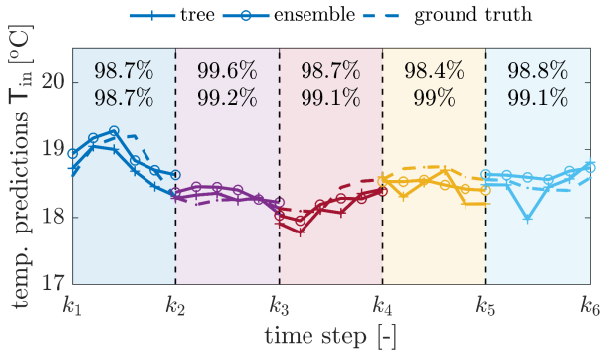


Fig. 4: Accuracy of predictions with snapshots in time: at time k_1, we make 6 predictions for time k_1 to $k_1 + 5$. The numbers represent mean accuracy in predictions for these steps for trees (top) and ensembles (bottom). This process is repeated at time $k_2, \ldots, k_5$.

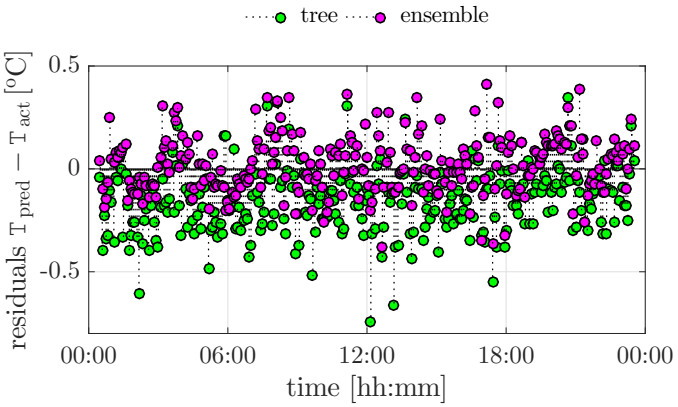


Fig. 5: Residual error for the tree $\mathcal{T}_1$ and the forest $\mathcal{R}_1$ that predict $T_{in,k+1}$ shows that the residuals for the forest are more concentrated around 0, while tree predictions have a much higher variance.

exactly same in both cases. Only difference in the optimization problem is due to system dynamics. In particular, MPC solves (2), DPC-RT solves (7) and DPC-En (10). T_{ref} in the optimization problems is set to 18 °C.

A. Simulation Settings

We test the performance of DPC controller against MPC controller on June 1, 2000. The choice of this day is arbitrary, and does not fall in the training period. For the HAMLab simulation model under consideration, the hourly weather predictions are only available from 1971-2000, however, the choice of the year does not influence the comparison. Heat gain due to occupancy Q_{oc} is discrete variable which is defined to be 500 W from 8 am to 6 pm and 0 W otherwise. The disturbances on June 1, 2000 are shown in Fig. 6(a). The sampling time T_s in MPC and model training for DPC is 5 min, while the weather data is available after every 1 h, so these disturbances are kept constant for 12 time steps. N and δ are again chosen to be 6. For solving optimization numerically, we use Gurobi Optimizer [6].

TABLE I: Quantitative comparison of R^2 score, mean value of objective function, energy consumption and mean deviance from the reference temperature.

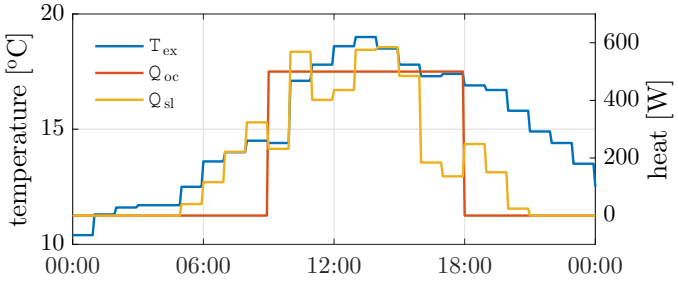
	R^2 score [−]	objective value (% change) [−]	energy [kWh]	mean deviance [°C]
MPC	–	59.70 (−)	3.72	0.23
DPC-En	95.3%	62.01 (3.8%)	3.50	0.24
DPC-RT	83.9%	64.53 (8.1%)	2.55	0.27

B. Results

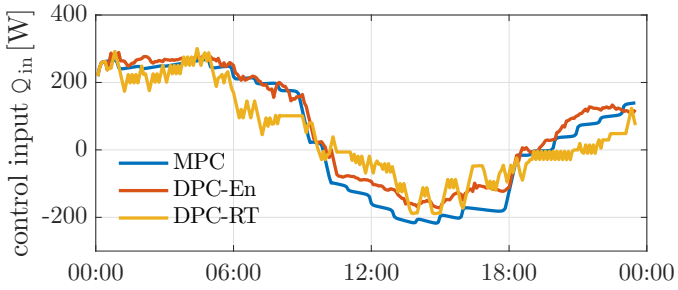
Optimal control strategies for all 3 methods are shown in Fig. 6(b). The chosen reference temperature and external disturbances require both heating and cooling at some point during the day. The solution obtained from MPC sets the benchmark that we compare to. Note that the MPC implementation uses the exact knowledge of the plant dynamics. Therefore, the associated control strategy is indeed the optimal strategy for the plant. In reality the quality of data, and modeling errors also affect the performance of the MPC controller [2]. Qualitatively, the DPC-En control input is not much different from that of MPC. Until 8 am, when the disturbances are not changing to a great extent, MPC and DPC-En are very close. Slightly after 8 am, when solar heat, external temperature and heat due to occupancy all increase abruptly, we observe that DPC-En cools less and differs slightly from the benchmark. Although DPC-RT also maintains the trend in the control input when the disturbances increase or decrease, the control strategy is quite different from that of MPC. Due to high variance in the predictions, DPC-RT also results in non-smooth inputs, which are not good for practical reasons of switching constraints on heating and cooling equipment in buildings.

Fig. 6(c) shows the plot for the zone air temperature. Again, DPC-RT shows a bigger deviation from the optimal solution while DPC-En is near-optimal. This is attributed to linear model averaging in ensemble learning which improves the model prediction. This improvement comes at the cost of reduced interpretability.

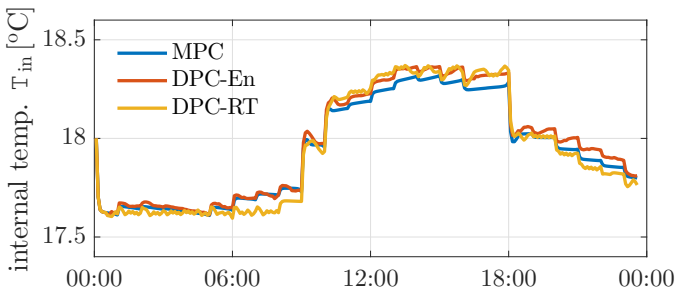
The quantitative results from the simulations are tabulated in Tab. I. We are interested in evaluating how close DPC performs in comparison to optimal benchmark set by MPC. The coefficient of determination or the R^2 score shows that 95.3% variability in MPC has been accounted for by DPC-En, while DPC-RT captures only 83.9% variability. In other words, DPC-En and MPC control inputs are 95.3% close. Fig. 6(d) shows the cumulative sum of the objective function as a function of time. At the end of the day, the cumulative sum for DPC-En is 3.8% more than MPC, and for DPC-RT it is 8.1% more than MPC. While both DPC-En and DPC-RT are near-optimal, the value of the objective function validates that DPC-En is more closer to the optimal solution. In the considered scenario, MPC tracks the reference temperature more closely offering better thermal comfort than DPC-RT or DPC-En by spending more energy.



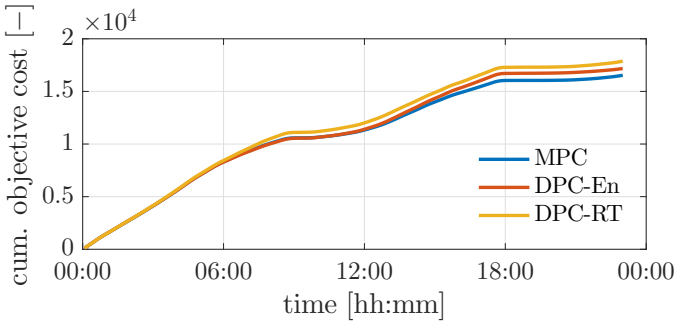
(a) Hourly weather data T_{ex} and Q_{sl} and internal disturbance Q_{in}.



(b) Optimal control strategies Q_{in}.



(c) Optimal internal zone air temperature T_{in}.



(d) Cumulative sum of the objective function.

Fig. 6: Comparison of optimal performance obtained with MPC, DPCRT and DPC-En on June 1, 2000. All 3 controllers start from the same state of the model.

VI. CONCLUSION & DISCUSSION

We present two data predictive control algorithms for finite receding horizon control using regression trees and regression trees ensembles. While DPC-RT uses a single regression tree for each step of the control horizon, DPC-En uses a random forest at each interval of the control horizon. By separating the controllable and uncontrollable features during training, and fitting a linear model on just the controllable features, the optimization is reduced to a simple convex program. The ensemble data predictive control, (DPC-En) is the first such method to bridge the gap between ensemble predictive models (such as random forests) and receding horizon control.

We implement DPC controller for energy management on a single zone building model and compare its performance with a MPC controller that uses the accurate dynamic model. We demonstrate that DPC can achieve comparable performance to MPC, with only 3.8% additional cost in terms of optimal objective function and within 95% in terms of R^2 score, while avoiding the cost and effort associated with learning a grey box/white box model of the system; and effect which magnifies at larger scale (hundreds of zones in a large building). Our current and future work is focused on demonstrating the capability of DPC on much larger and realistic building models, and using real building data and test-beds. We are also addressing the question of learning reliable data-driven models with limited functional testing of the plant, and providing stability guarantees for DPC for model switching. DPC has applications which go beyond buildings and energy systems, to industrial process control, and controlling large critical infrastructures like water networks, district heating & cooling. DPC is immensely valuable in situations where first principles based modeling cost is extremely high.

REFERENCES

[1] M. Behl, A. Jain, and R. Mangharam. Data-driven modeling, control and tools for cyber-physical energy systems. In *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS)*, pages 1–10. IEEE, 2016.

[2] M. Behl, T. Nghiem, and R. Mangharam. Model-iq: Uncertainty propagation from sensing to modeling and control in buildings. In *International Conference on Cyber Physical Systems*. IEEE/ACM, 2014.

[3] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[4] J. Drgoňa and M. Kvasnica. Comparison of MPC strategies for building control. In *Process Control (PC), 2013 International Conference on*, pages 401–406. IEEE, 2013.

[5] J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[6] I. Gurobi Optimization. Gurobi optimizer reference manual, 2015.

[7] A. Jain, R. Mangharam, and M. Behl. Data Predictive Control for peak power reduction. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, pages 109–118. ACM, 2016.

[8] A. Kusiak, Z. Song, and H. Zheng. Anticipatory control of wind turbines with data-driven predictive models. *IEEE Transactions on Energy Conversion*, 24(3):766–774, 2009.

[9] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[10] J. R. New, J. Sanyal, M. Bhandari, and S. Shrestha. Autotune e+ building energy models. *Proceedings of the 5th National SimBuild of IBPSA-USA*, pages 1–3, 2012.

[11] D. Sturzenegger, D. Gyalistras, M. Morari, and R. S. Smith. Model predictive climate control of a swiss office building: Implementation, results, and cost–benefit analysis. *IEEE Transactions on Control Systems Technology*, 24(1):1–12, 2016.

[12] A. Van Schijndel. Integrated heat, air and moisture modeling and simulation in hamlab. In *IEA Annex 41 working meeting, Montreal, May*, 2005.

[13] E. Žáčeková, Z. Váňa, and J. Cigler. Towards the real-life implementation of MPC for an office building: Identification issues. *Applied Energy*, 135:53–62, 2014.