# Supercomputer Assisted Generation of Machine Learning Agents for the Calibration of Building Energy Models

Jibonananda Sanyal*
Oak Ridge National Laboratory
1 Bethel Valley Road
Oak Ridge, TN - 37831
sanyalj@ornl.gov

Joshua New†
Oak Ridge National Laboratory
1 Bethel Valley Road
Oak Ridge, TN - 37831
newjr@ornl.gov

Richard Edwards‡
Min H. Kao Building, Suite 401
1520 Middle Drive
Knoxville, TN 37996
redwar15@utk.edu

## ABSTRACT

Building Energy Modeling (BEM) is an approach to model the energy usage in buildings for design and retrofit purposes. EnergyPlus is the flagship Department of Energy software that performs BEM for different types of buildings. The input to EnergyPlus can often extend in the order of a few thousand parameters which have to be calibrated manually by an expert for realistic energy modeling. This makes it challenging and expensive thereby making building energy modeling unfeasible for smaller projects. In this paper, we describe the "Autotune" research which employs machine learning algorithms to generate agents for the different kinds of standard reference buildings in the U.S. building stock. The parametric space and the variety of building locations and types make this a challenging computational problem necessitating the use of supercomputers. Millions of EnergyPlus simulations are run on supercomputers which are subsequently used to train machine learning algorithms to generate agents. These agents, once created, can then run in a fraction of the time thereby allowing cost-effective calibration of building models.

## Categories and Subject Descriptors

I.2 [**Distributed Artificial Intelligence**]: Intelligent agents; C.5.1 [**Large and Medium ("Mainframe") Computers**]: Super (very large) computers

## General Terms

Design, Experimentation, Performance, Standardization

*Dr. Sanyal is with the Building Technologies Research and Integration Center at the Oak Ridge National Laboratory.

†Dr. New is with the Building Technologies Research and Integration Center at the Oak Ridge National Laboratory.

‡Mr. Edwards is a Ph.D. degree candidate at the Department of Electrical Engineering and Computer Science at the University of Tennessee in Knoxville.

## Keywords

Building Energy Modeling, Supercomputer, Parametric Ensemble, Machine Learning, Calibration, Big Data

## 1. INTRODUCTION

In the year 2010, buildings consumed 41% of the primary energy (up from 39% in 2006 [18]) constituting a larger share than either transportation (28%) or industry (31%) [19]. This is approximately $400 billion and up from $220 billion in 2006. Buildings consumed 74% of all electrictity and 34% of all natural gas produced in the United States thereby contributing 40% of the carbon dioxide emmissions in the United States [19]. The Department of Energy's (DOE) Building Technologies Office (BTO) has a significant stake in improving the energy footprint and efficiency of the buildings sector for economic, social, and environmental benefits.

The accurate modeling of buildings can serve as an important tool in understanding the energy footprint for both designing new buildings as well as retrofitting existing buildings for energy savings. The large number of existing buildings that do not employ energy efficient technologies present a low-hanging fruit that could significantly and cost- effectively contribute to energy savings across the entire country by the application of retrofit packages. Primary to this objective is the realistic modeling of the buildings that accurately reflect potential energy savings for different scenarios [2, 3, 6, 15].

Each building is unique and the model representation of the building tends to be a one-off activity. The accurate modeling of all building properties is laborious and requires expertise. In the "business-as-usual" model, experts manually adjust different input parameters until satisfactory results are obtained. The time required for this can sometimes be in the order of a few months. Clearly, the cost of such an activity makes building energy modeling unfeasible for smaller buildings.

While buildings may be one-off, they still must comply to building code. DOE has standard reference models of buildings that are used nationwide and are representative of the U.S. building stock [12]. These building models are used for normative analysis to determine how policy changes would affect energy consumption in the US, determine tax tradeoffs, design building codes, trade-off incentives, and eval-

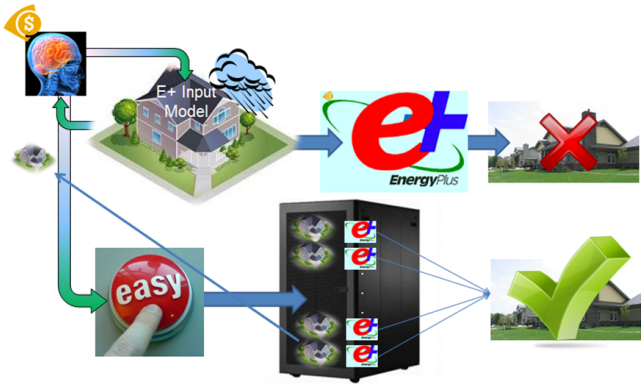


**Figure 1: Autotune workflow for E+ building energy models as a cost-effective solution for generating accurate input models.**

uation of the effect of climate change on buildings. To a large extent, this addresses the challenge of accurately modeling and scaling the building types across American Society of Heating, Refrigerating, and Air-Conditioning Engineers (ASHRAE) climate zones [2, 3, 15] and projecting how specific policies or retrofit packages would maximize return-on-investment with subsidies through federal, state, local, and utility tax incentives, rebates, and loan programs.

This paper describes the "Autotune" methodology [17] being developed at the Oak Ridge National Laboratory which automatically calibrates a building energy model using supplied utility data for building retrofit purposes. This is achieved by using trained machine agents which are generated from large parametric simulations run on supercomputing systems using machine learning algorithms. Supercomputing systems are used to run millions of parametric simulations for different building residential and commercial building types across various climate zones for training the machine agents. A public portal (http://autotune.roofcalc.com) provides access to all the simulated data (~270 TB) for open science as well as capabilities are being built to offer Autotune as a web-service.

The development of an autotuning capability (figure 1) to intelligently adapt building models or templates to building performance data would significantly facilitate market adoption of energy modeling software, aid in accurate use cases such as the effective retrofit strategies for existing buildings, and promote BTP's goals of increased market penetration for energy modeling capabilities.

## 2. AUTOTUNE METHODOLOGY

There are a large number of software tools (~20 major ones) to perform Building Energy Modeling with each having certain strengths and weaknesses. Among these, EnergyPlus (E+) is the flagship DOE whole building energy simulation tool and has a large community of users. The source comprises about ~600,000 lines of FORTRAN code. A typical input to the simulation engine can contain over 3,000 input parameters for a regular residential building which must be tuned to reflect the building properties. Besides tuning, another challenge is obtaining quality data or representative reference values for the different material properties. Various experiments [7] have established that there are significant mismatches between the supplier's product specifications and those provided by the ASHRAE handbook. There are often deviations in material properties of the product against those on its label and as specified by the supplier. These and various construction factors cause the finished building to deviate from the original model of the building. Retrofit packages are useful for existing buildings since such efforts provide a significant return on investment for existing buildings.

To be able to reasonably model the above scenarios, a building energy model must be adjusted to match measured data. This matching is typically performed manually by a building modeling expert and is a non-repeatable, time-consuming, expensive, and laborious process. A very large number of building models start out with the DOE reference buildings (which are most representative of the U.S. building stock) and go through the manual adjustment of geometry, HVAC properties, insulation, fenestration, infiltration properties, etc. The high cost of building energy modeling makes it unfeasible to apply to smaller projects thereby creating a gap in the energy assessment for a large majority of smaller buildings. "Autotune" provides a methodology to cost effectively perform building energy model calibration providing practitioners in the field with an option to apply BEM to smaller buildings.

The goal of the "Autotune" project is to save the time and effort spent by energy modelers in adjusting simulation input parameters to match the measured data by providing an 'easy' button (figure 1). An initial model of the building is provided to Autotune along with a set of measured sensor data. Autotune then spins off the trained machine learning agents and returns a tuned model of the building.

The individual trained machine agents are generated by performing machine learning on large parametric simulations for the different classes of standard DOE buildings [12] across different climate zones. At the core of the Autotune methodology is a set of multi-objective machine learning algorithms that characterize the effect of individual variable perturbations on EnergyPlus simulations (figure 2) and adapts the given model to match its output to the supplied sensor data (figure 3). Once machine learning agents are tuned and available, the computational cost of tuning a typical user's building model is reduced to matter of a few hours using widely available desktop computational resources.

The system is currently being demonstrated to match a subset of 250 sensors of 15-minute resolution data in a heavily instrumented residential building in addition to DOE's standard reference building models [12] for a medium sized office, a warehouse, and a stand-alone retail building. Further, the simulations comprise of three vintages (old, new, and recent) of the DOE commercial reference buildings across 16 different cities representing the different ASHRAE climate zones and sub-zones.

### 2.1 Parametric Space

EnergyPlus requires about ~3,000 input parameters. The computational space for performing parametric analysis on

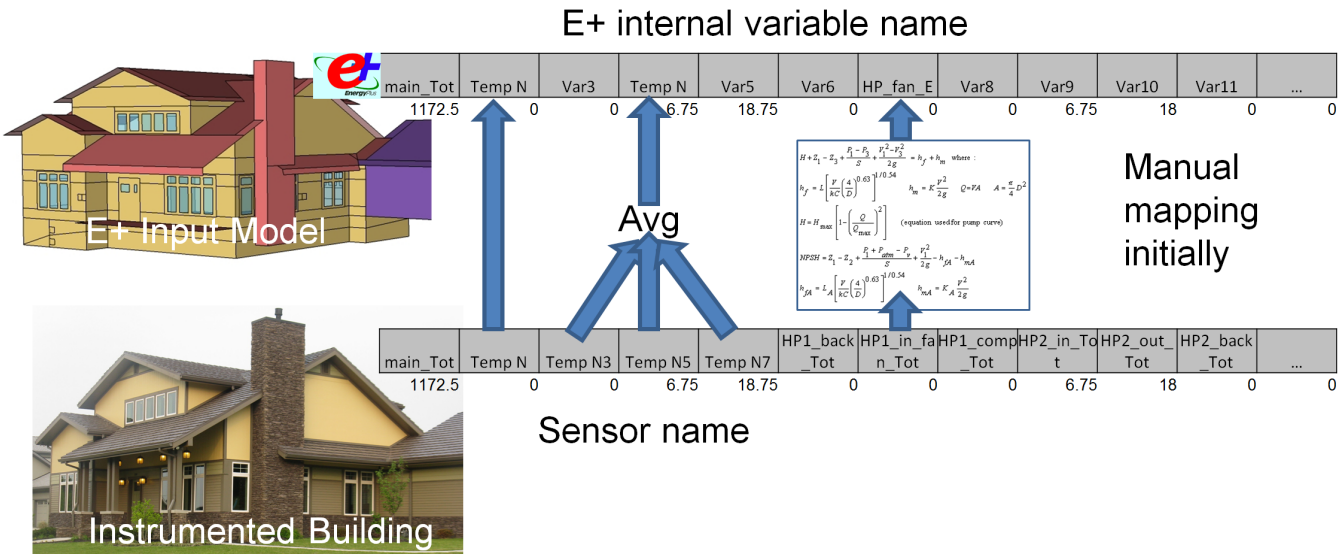


**Figure 2: A virtual building model (software space) and a real building (sensor space), when viewed as vectors of numbers, allows a mathematical mapping between vector spaces for direct comparison between simulation state and sensed world state.**

~3,000 inputs is huge. Brute-force exploration for 156 of 3000+ input parameters using only minimum, average, and maximum would require $5 \times 10^{52}$ EnergyPlus simulations. This is both impractical and not feasible. Several numerical algorithms are used which allow convergence to a solution without brute-forcing every combination of inputs and intelligent experimental design guides parametric analysis to a much smaller sample size.

## 2.2 Parametric Input Sampling

Building technology experts who routinely perform calibration of BEMs analyzed the various inputs and picked only the most important parameters in their experience. This reduces the number of parameters to 156 for the heavily instrumented residential building. These are parameters that are most commonly used by the energy modelers. The modelers further ranked into three importance categories. The experts further defined realistic bounds and incremental step size values for the parameters. Furthermore, various meta-parameters were determined which allow several individual parameters to be varied as a function of a single input parameter.

Even with ~156 input parameters and three levels of incremental values for each of the simulations, we are looking at 10 million simulations. Each individual simulation takes between 2 to 8 minutes depending on the type of building, which translates to 2 million compute hours after accounting for overheads. Using Oak Ridge National Laboratory's *Titan* supercomputer (currently ranked as the fastest supercomputer in the world), this would take 2 months of calendar time to just run the simulations, let alone manage the data, perform machine learning, and subsequent analysis. Effective, scalable methods to sample the input space is crucial.

We use the expert's grouping of important parameters to divide the sampling space into groups of relative importance. We have also used low-order Markov ordered simulations to determine variables with a monotonic effect on sensor data that can reliably be interpolated to estimate impact of a given input. The source of variance of individual variables is being used to guide sampling rates of the more sensitive inputs. Finally, experts in multi-parameter optimization will be investigating computational steering algorithms to determine the optimal sampling strategy for the remaining space beyond the brute-force sampling of higher order Markov chains of Monte Carlo simulations.

In summary, a total of 8 million simulations are planned for following building types:

- Residential: ~5 million
- Medium office: ~1 million
- Stand-alone retail: ~1 million
- Warehouse: ~1 million

## 2.3 Supercomputing

Time on several supercomputing systems have been competitively awarded and used to demonstrate the scalability of our algorithms and code for the massively parallel leadership-class computing systems. Systems include the 1024-core shared memory *Nautilus*, 2048-core *Frost*, and the 299,008-core *Titan* which is currently the world's fastest supercomputer at 20 petaflops. The *Nautilus* machine has 4 TB of global shared memory visible to every processor on the system. *Titan* (and *Frost* too) has a distributed memory model with each node having 16 processors and 32 GB of RAM. It is worthwhile to mention that bulk of the machine learning and tuning efforts have been done on Nautilus, which is an XSEDE resource. The Autotune project is one of the biggest users of Nautilus and has clocked more than 300,000 service units, or compute hours [16].

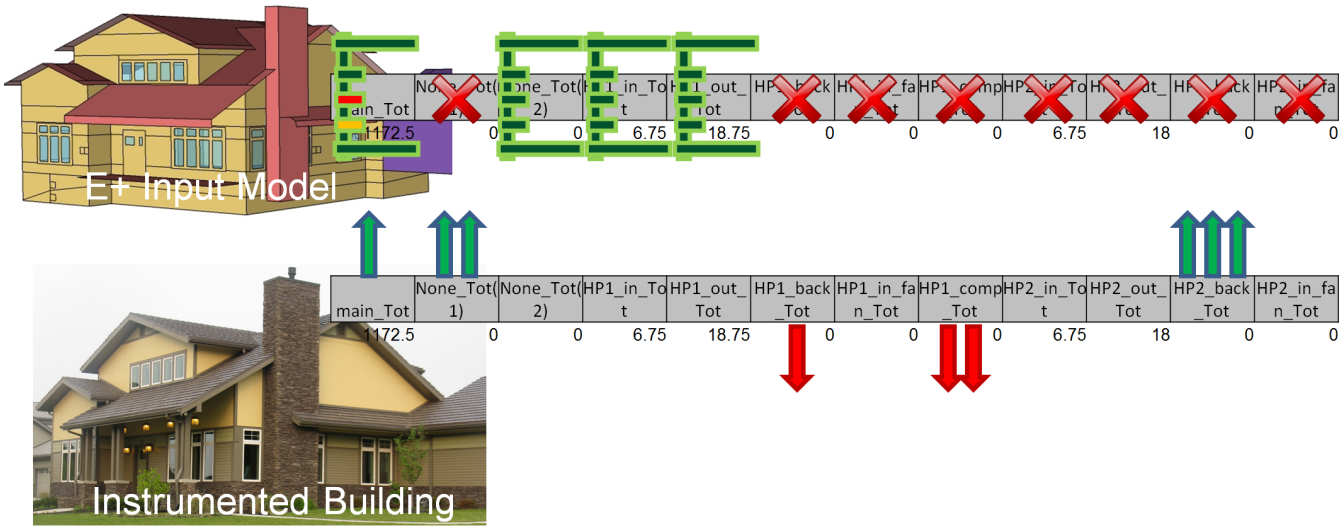


**Figure 3: Sensitivity analysis of E+ simulations mapped to their effect in sensor space.**

While in theory this is an embarrassingly parallel problem (parametric ensemble) and should be easy to parallelize, various complicating factors make this difficult to scale in practice. First, EnergyPlus was developed as a desktop application and was not supercomputer ready. In a typical execution trace for a single simulation, a sub-directory and a large number of files (12+ files amounting to 100+MB) are created. Second, constant moving and soft-linking of the files are done as the simulation workflow executes. Third, an annual simulation for warehouses with 15-minute output of 82 channels is 35MB in size and currently needs to be stored on disk for later analysis. For other types of buildings, the output data is much larger in number of variables and consequently, file size. In other words, the entire process is particularly I/O intensive, which complicates the scalability of parallel execution on supercomputers. We attempt to mitigate these issues in many ways.

A branch of the source code for the engine has been made open-source [1]; however, writing a wrapper for over 600,000 lines of code to streamline I/O for use on supercomputers is outside the scope of this work. We treat E+ as a black-box and use it to simulate our 8 million runs. In addition, the workflow depends on a number of additional executables, the source-code of which is not available.

The current job submission scripts were designed to pick up a set of input files and execute them in parallel on the number of cores requested and moving the output to a directory when complete. Initial experiments on the supercomputing systems delivered very poor scaling performance which were expectedly traced to the bandwidth and Lustre file-system saturation with the frequent number of large I/O requests. The biggest bottleneck was determined to be the communication with the metadata server and the storage targets.

In order to alleviate the filesystem bottleneck, we made use of the memory-based virtual file-system which gave us more than two orders of magnitude improvement over using the Lustre filesystem. In addition, we block-partitioned and streamlined our input and output mechanisms. To outline the steps performed:

1. EnergyPlus comes with a large number of supporting executable programs and associated files. A typical E+ simulation is essentially a workflow where multiple executables are invoked with each producing temporary files ingested by subsequent programs. We minimized the engine's folder structure to include only the binaries and libraries required for our simulations, modified the execution scripts to use relative paths, and compressed the minimized file structure to make it ready to be loaded into the virtual filesytem. We also statically compiled the main EnergyPlus engine from the open source branch [1].

2. In an effort to reduce the number of input files fetched, we performed a pre-processing step in which we grouped the inputs into blocks of 64 simulations each and packed them into compressed tarballs. This reduces the number of files fetched by a factor of 64 and reduces size by ~60%.

3. For *Nautilus*, individual jobs can be placed on individual processors using the 'dplace' command. A heavily modified job submission script allows us to request a specific number of cores and provide a count of the number of batches to run. For example, a request for 256 cores with 90 batches would start out by picking out $256/64 = 4$ blocks of compressed input files and the simulation engine, and then parallelly extract them to the virtual file-system. Each core then executes a simulation (using an explicit 'dplace' command which runs a job on a core). After completion, the data is moved to the physical file-system and the next batch of 4 compressed files is loaded. This is repeated 90 times.

4. For *Frost* and *Titan*, the virtual file system (*tmpfs*) is shared-memory visible within a node. Additionally, these are Cray systems and do not use 'dplace' for

**Table 1: E+ run times for a residential building on the *Nautilus* supercomputer, inclusive of disk write time.**

| Procs | E+ sims | Wall-clock Time (h:mm:ss) | Time/E+ task (mm:ss) |
|---|---|---|---|
| 32 | 32 | 0:02:08 | 2:08 |
| 64 | 64 | 0:03:04 | 3:04 |
| 128 | 128 | 0:04:11 | 4:11 |
| 128 | 1024 | 0:34:24 | 4:18 |
| 256 | 2048 | 1:25:17 | 10:40 |
| 512 | 1024 | 0:18:05 | 9:02 |

direct placement of individual jobs on a processor. We wrote a message passing interface (MPI) program that would make each node load the engine and a block of 64 runs into its shared-memory. Since each node has 16 processors and there are 64 files in a compressed block of inputs, the node makes four iterations to run all 64 simulations.

5. Once a block of simulations is complete, the output files are added to a tarball and moved to disk. This is typically about 1.5 – 5.3 GB in size depending on the type of simulation. This also reduces the number of I/O interactions with the Lustre filesystem by a factor of 64. The virtual file system is cleaned and prepared for the next block to run.

We have been able to complete a batch of 1 million E+ simulations for the warehouse building using *Nautilus* and *Frost* in under 2 weeks of continual execution (along with other users running other jobs on the systems). The theoretical runtime using average job execution metrics was estimated at about 7.6 days for the batch of 1 million simulations. We were also able to complete the 1 million Standalone Retail simulations in ~2 weeks.

*Titan* has just been made available to users and we have started using the machine for scaling to tens of thousands of cores. At the time of this writing, we have observed weak scalability upto 65,536 cores and anticipate the trend to continue as we run even larger batches. Tables 1 and 2 illustrate the scalability observed on both machines.

We expect to generate approximately 270 TB of raw data when all simulations are complete. We have estimated that this can be compressed down to about 70 TB, which is still a large amount of data. This is the size of simulation data prior to any operations performed as part of the analysis processes. There are certain logical partitions in the data such as type of building simulation, its vintage, location, and also the parameter sampling and grouping strategy which helps us in breaking down the data management space. While many database-related technologies have been tested and explored, effectively storing this data for quick retrieval and analysis remains a challenge.

We have explored a number of databases including MySQL, noSQL/key-value pair, columnar, and time-series database formats for simulation/sensor data and currently implement a hybrid solution with a part of the summarized data entered in a database and readily accessible for querying and analysis

**Table 2: E+ run times scaling chart for a commercial building on the *Titan* supercomputer, inclusive of disk write time. For comparison with table 1, 4 commercial building simulations are run for each processor.**

| Processors | Wall-clock Time (mins) | E+ tasks |
|---|---|---|
| 16 | 18:14 | 64 |
| 32 | 18:19 | 128 |
| 64 | 18:34 | 256 |
| 128 | 18:22 | 512 |
| 256 | 20:30 | 1024 |
| 512 | 20:43 | 2048 |
| 1024 | 21:03 | 4096 |
| 2048 | 21:11 | 8192 |
| 4096 | 20.00 | 16384 |
| 8192 | 26:14 | 32768 |
| 16384 | 26:11 | 65536 |
| 65536 | 44:52 | 262144 |
| 131072 | 68:08 | 524288 |

while and the raw data being fetched on demand. This data is currently provided with no guarantees since the entire data queryable with an assured turnaround time (a solution similar to a hadoop stack) for queries is currently infeasible.

We currently have an 80 TB Synology disk station which primarily serves as a data repository with fast access to individual files and are in the process of acquiring additional hardware for some of the analysis needs. Although additional hardware is being procured, it is geared towards adding compute capabilities for analysis but is not geared towards assured turn-around times for queries.

# 3. MACHINE LEARNING AGENTS

The integration of multiple machine learning algorithms and software implementations into a single HPC-enabled system, known as MLSuite [11], for data mining on large data has general extensibility to search and optimization problems across multiple domains. It is used in several specific ways as part of the Autotune project [17]. One is the exploration of a more data-driven, sensor-based energy modeling (sBEM) [10] through machine learning prediction of building performance as a function of observed data using statistical methods rather than complex, physics-based, engineering algorithms [8]. Second, usage is speeding up annaul EnergyPlus (E+) simulations by creating surrogate simulation engines that trade-off accuracy for order of magnitude speedup [9]. Third, creating inverse simulation engines that use sensor data (corresponding to simulation output) to predict an anticipated set of simulation input parameters, which can act as seed points for further exploration of the true inputs. Fourth, it is being used to both fill in missing values and detect outliers in time-series data by intelligently predicting the anticipated value as a function of temporal patterns for a specific data channel in combination with spatial patterns of other data channels. Other methods not in MLSuite are also used for quality assurance of sensor data[5] [4], sensitivity analysis, and uncertainty quantification with plans to scale the software beyond what is currently possible with existing hardware resources. Many of these analyses steps are pre-processing or integrated as part of the Autotune process which relies upon evolutionary computation[13] with metrics for relaying accuracy of the tuned virtual model to real-world

data [14]. All references in this section point to full research papers and reports developed by the Autotune project team; all interested readers are referred to these sources for further details on how these analysis methodologies are conducted.

There is often a set of questions or concerns regarding the apparent disconnect of supercomputers needed for the aforementioned analysis steps and the need to deploy a quick, easy-to-run Autotune utility on a desktop computer (or web server) without the continued reliance on HPC resources. In an effort to address this concern, it should be noted that new scientific discoveries requires searching through a large candidate model space. For the Autotune project, as with the application of most machine learning or optimization techniques, this model space contains mathematically relationships derived from existing data. This process of searching through the model space in order to find a sufficiently accurate and fast model requires HPC resources. Once a specific model is selected (e.g. a trained agent), runtime of that agent can be deployed on a desktop machine. To provide a more explicit exegesis, HPC resources are used in combination with MLSuite to perform feature selection, ~8 machine learning algorithms, different parameter settings for each algorithm, different order of input feature vectors, different cross-validation methodologies (to prevent over-fitting), and even a limited amount of ensemble learning combinations/workflows of multiple algorithms in order to systematize the process of finding the optimal model among the overwhelmingly large space of conflicting variables (each of which affects the performance of a trained agent). As an explicit example for the Autotune project, the process for approximating E+ has taken weeks on HPC resources for even a very limited attempt at building E+ surrogate engines; however, the selected trained agent produces approximate E+ output from simulation inputs in ~15 seconds. It should also be noted that several input parameters for training of machine learning relate to building square footage, number of floors, etc., in an effort that a trained agent would be of general applicability by statistically capturing dynamics of the same physics applied by the simulation engine. Current research is deriving agents for each particular DOE building classification (residential, warehouse, office, stand-alone retail), but future research is planned to indicate the stability and applicability of trained models within or across other building types.

Tables 3, 4, 5, and 6 illustrate the various performance and scalability measures, primarily by use of the *Nautilus* supercomputer.

**Table 3: Different types of learning systems in use and their acronyms for reference in the scalability tables.**

| Learner Type | Acronym |
|---|---|
| Linear Regression | REG |
| Feed Forward Neural Network | FFNN |
| Hierarchical Mixture of Linear Regression Experts | HME-REG |
| Non-linear Support Vector Regression | SVR |
| Hierarchical Mixture of Feed Forward neural Networks | HME-FFNN |
| Hierarchical mixture of Least-Square Support Vector machines | HME-LSSVM |
| Fuzzy C-Means clustering with local models | FCM - FFNN |
| All learning systems combined (suite run) | All |

**Table 4: MLSuite: Actual run times for 1 core and 4 cores, as well as the expected run time for 90 cores with 1 core per job, and 120 cores with 4 cores per job.**

| Learner | Jobs | 1 core [hours] | 4 cores [hours] | 90 cores (1-core per job) [secs] | 120 cores (4-cores per job) [secs] |
|---|---|---|---|---|---|
| REG | 10 | 0.00759 | 0.00063 | 0.273 | 0.226 |
| FFNN | 60 | 1.37 | 0.688 | 82.295 | 82.295 |
| HME-REG | 90 | 1.329 | 1.172 | 53.17 | 140.59 |
| SVR | 10 | 0.04749 | 0.03985 | 17.097 | 14.344 |
| HME-FFNN | 540 | 482.903 | 245.633 | 21200.617 | 30376.072 |
| HME-LSSVM | 90 | 1.257 | 0.536 | 50.288 | 64.348 |
| FCM-FFNN | 420 | 395.85 | 197.908 | 1979.273 | 2770.711 |
| All | 1220 | 533.092 | 271.16 | 22924.789 | 33241.826 |

**Table 5: MLSuite: Average run time for each (and all) learner(s) using a single core per job, and four cores per job.**

| Learner | Avg-Runtime 1 Job (1 core) [secs] | Avg-Runtime 1 Job (4 cores) [secs] |
|---|---|---|
| REG | 0.273 | 0.226 |
| FFNN | 82.295 | 41.302 |
| HME-REG | 53.17 | 46.863 |
| SVR | 17.097 | 14.344 |
| HME-FFNN | 3533.436 | 1687.56 |
| HME-LSSVM | 50.288 | 21.449 |
| FCM-FFNN | 395.855 | 197.908 |
| All | 1637.485 | 810.776 |

**Table 6: Improvements made over exact Leave-One-Out (eLOO) using approximate (aLOO) for Least Squares Support Vector Machines (LS-SVM) and Kernel Ridge Regression (KRR) with minimal trade-off in accuracy [15] with example sizes going from 8,192 to 1 million [solution time in seconds].**

| # Examples | $2^{13}$ | $2^{14}$ | $2^{15}$ | $2^{16}$ | $2^{17}$ | $2^{18}$ | $2^{19}$ | $2^{20}$ |
|---|---|---|---|---|---|---|---|---|
| eLOO | 4.43 | 35.25 | 281.11 | | | | | |
| aLOO-LSSVM | 1.3 | 2.6 | 5.32 | 10.88 | 22.45 | 47.41 | 101.36 | 235.83 |
| aLOO-KRR | 0.54 | 1.06 | 2.14 | 4.3 | 8.55 | 17.28 | 35.39 | 68.22 |

# 4. PUBLIC FACING PORTAL

The parametric simulations run by supercomputers has been uploaded to a centralized database to allow public access to this data (figure 4). It is anticipated that this data would be of interest to researchers at universities, data-mining experts, entrepreneurs, industry, and several other organizations for a myriad of purposes. Several tools have been developed for easily defining and running parametric E+ simulations, compressing data, and sending to a centralized server. In addition, several methods have been made available for the general public to freely access and use this data.

The data storage and access software has been architected as a distributed, heterogeneous, client-server framework. The embarrassingly parallel nature of the independent simula-

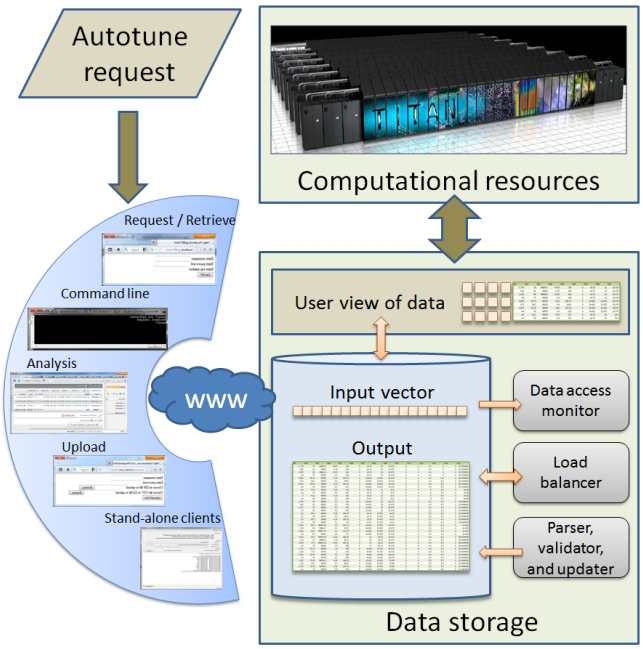


**Figure 4: Architecture of the Autotune software-as-a-service product.**

tions allows us to exploit computational resources that are remote and disparate leading to an architecture capable of collecting simulation results from individual desktop systems as well as supercomputing resources.

Data access patterns are being studied to allow rearchitecting the database and load-balancing for higher efficiency. The internal data storage format is not tied to the format of input or output E+ variables but instead uses its own generic internal naming scheme. Depending on the current set of variables and preferences of the users, a custom view of the data is provided that can be easily queried, summarized, and analyzed, providing the full benefits of a relational database system. Figure 4 shows the various software components of the Autotune database illustrating the independence of the data storage mechanism from the user view of the data, the software components on the server, and the remote web-based clients. There are several methods for accessing the data: a web-based input file reconstructor, command-line access for MySQL queries, phpMyAdmin for GUI-based interaction with a data subset, a webpage for uploading simulation data, and the software tool EplusGenius.

In the initial stage of the effort, a Qt based tool named "EplusGenius" was developed that leveraged the power of idle desktops to run EnergyPlus simulations and upload the results to a database. It was engineered to provide the choice to run only after office hours and on weekends. While effective in harnessing unused computational power from a network of computers, the computational requirements to achieve a reasonable time-to-solution necessitate the use of HPC resources. While not in use currently, we anticipate that the "EplusGenius" tool will be effective in executing the trained machine agents.

A web-based method for reconstructing an input file from a vectorized set of input parameters from the database is provided. A web-interface is also available for uploading external E+ simulations input and output files to the database. External access to this database can be provided upon request using several user validation and access methods including a command line interface, password-protected phpMyAdmin for interactive queries, drill-down, and analysis of the simulation database.

In addition to web-access to different tools, a means of providing a user with an easy way to develop a building model using available templates for different kinds of buildings is in progress. This, along with user provided sensor data will help in making Autotune available as a web-service.

## 5. CONCLUSIONS

The successful completion of the presented Autotune effort underway at the Oak Ridge National Laboratory will go a long way in alleviating the tedious task of tuning a building energy model to sensor data. The employment of machine learning agents performs a multi-objective optimization of the input parameters to provide a solution that best matches the input sensor data. The refinement and dimension reduction of the input parameter space to 156 important ones identified by the experts helps to reduce the computational space. Further, various methods to scientifically sample the input parameter space helps to reduce the computational space.

The paper highlights the scientific contribution of automating BEM calibration and technical challenges faced in simulating a very large set of simulations using an engine that has been developed for desktop applications, and in the process, generating a large amount of data. We expect that lessons learned and software developed will be useful for researchers who intend to run large ensembles and perform data mining of very large data. We also hope that some of the challenges faced and lessons learned in analyzing, moving, and managing large data across hardware platforms will provide beneficial insight and help to researchers in planning such endeavours. Lastly, we also expect that the open availability of the parametric simulation datasets for the standard DOE reference buildings will directly benefit the building sciences community.

## 6. ACKNOWLEDGMENTS

This work was funded by field work proposal CEBT105 under the Department of Energy Building Technology Activity Number BT0201000. We would like to thank Amir Roth for his support and review of this project. We would like to thank our collaborators which include Dr. Aaron Garrett from Jacksonville State University, and Mr. Buzz Karpay from Karpay Associates. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. Our work has been enabled and supported by data analysis and visualization experts, with special thanks to Pragneshkumar Patel, at the NSF funded RDAV (Remote Data Analysis and Visualization) Center of the University of Tennessee, Knoxville (NSF grant no. ARRA-NSF-OCI-0906324 and NSF-OCI-1136246). We

would also like to thank Willan Renaud at the Oak Ridge Leadership Computational Facility for his constant help in overcoming system issues. We also thank XSEDE and the Oak Ridge Leadership Computational Facility for making the systems available for the experiments.

Oak Ridge National Laboratory is managed by UTBattelle, LLC, for the U.S. Dept. of Energy under contract DE-AC05-00OR22725. This manuscript has been authored by UT-Battelle, LLC, under Contract Number DEAC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes.

## 7. REFERENCES

[1] Big Ladder Software. Epx, a modified version of energyplus v. 7, 2013.

[2] Briggs, R., Lucas, R., and Taylor, Z. Climate classification for building energy codes and standards: Part 1–development process. *ASHRAE Transactions 109*, 1 (2003), 109–121.

[3] Briggs, R., Lucas, R., and Taylor, Z. Climate classification for building energy codes and standards: Part 2–zone definitions, maps, and comparisons. *ASHRAE Transactions 109*, 1 (2003), 122–130.

[4] Castello, C. C., and New, J. R. Autonomous correction of sensor data applied to building technologies utilizing filtering methods. *Fourth ACM International Conference on Future Energy Systems* (2012).

[5] Castello, C. C., and New, J. R. Autonomous correction of sensor data applied to building technologies utilizing statistical processing methods. *2nd Energy Informatics Conference* (2012).

[6] Deru, M., Field, K., Studer, D., Benne, K., Griffith, B., Torcellini, P., Liu, B., Halverson, M., Winiarski, D., Rosenberg, M., et al. US Department of Energy Commercial Reference Building Models of the National Building Stock.

[7] DeWit, M. *Uncertainty in predictions of thermal comfort in buildings*. PhD thesis, Technische Universiteit Deft, Netherlands, 2001.

[8] Edwards, R. E., New, J. R., and Parker, L. Predicting future hourly residential electrical consumption: A machine learning case study. *Buildings and Energy* (2012).

[9] Edwards, R. E., New, J. R., and Parker, L. Constructing large scale energyplus surrogates from large data. *Buildings and Energy* (2014).

[10] Edwards, R. E., and Parker, L. Sensor-based energy modeling final report fy2012. Tech. rep., Oak Ridge National Laboratory, TN, 2012.

[11] Edwards, R. E., and Parker, L. Mlsuite final report fy2012. Tech. rep., Oak Ridge National Laboratory, TN, 2013.

[12] Field, K., Deru, M., and Studer, D. Using DOE Commercial Reference Buildings for Simulation Studies, 2010.

[13] Garrett, A., and New, J. R. Evolutionary tuning of building model parameters. Tech. rep., Oak Ridge National Laboratory, TN, 2012.

[14] Garrett, A., New, J. R., and Chandler, T. Evolutionary tuning of building models to monthly electrical consumption. *ASHRAE Annual Conference* (2014).

[15] IECC 2009 and ASHRAE 90.1-2007. *Energy Code Climate Zones*.

[16] National Institute for Computational Sciences. Supercomputer-assisted calibration methodology enhances accuracy of energy models, 2013.

[17] New, J. R., Sanyal, J., Bhandari, M. S., and Shrestha, S. S. Autotune e+ building energy models. *5th National SimBuild of IBPSA-USA* (2012).

[18] U.S. Dept. of Energy. *Building Energy Data Book*. D&R International, Ltd., 2010.

[19] U.S. Dept. of Energy. *Building Energy Data Book*. D&R International, Ltd., 2011.