# Evaluating the Suitability of Regression-Based Emulators of Building Performance in Practice: A Test Suite

Parag Rastogi *,[1, 2, 3], Mohammad Emtiyaz Khan[2], and Marilyne Andersen[3]

[1]arbnco Ltd., Glasgow, UK
[2]RIKEN Center for Advanced Intelligence Project, Tokyo, Japan
[3]LIPID laboratory, École Polytechnique Fédérale de Lausanne, Switzerland

DD MMM 2021

## Abstract

Building Performance Simulation (BPS), a useful tool to assess the operational performance of buildings and systems, can often be computationally expensive. The use of BPS is cumbersome for problems where the speed of response is important, e.g., real-time control, uncertainty quantification, parametric exploration, or stock modelling. Emulators, such as those based on regression, offer a faster substitute, but their reliability can be questionable. This paper proposes seven tests to check if an emulator *is a suitable replacement for simulation* in practice. The tests are categorized using four criteria: accuracy, speed, generalisability, and ease of use. The tests can be included in the process of setting up an emulator-based workflow. A use case is provided for emulators based on linear and non-linear regression (Gaussian Process models). This work aims to enable a practitioner to reliably conduct performance assessment for buildings using emulators.

***Keywords***— non-linear regression, building simulation, test suite, regression model

*Corresponding author: contact@paragrastogi.com

## 1 Introduction

Building Performance Simulation (BPS) is a tool to quantify the impact of decisions about a building's design, specification, or operation on its performance, i.e., changes in thermal or visual conditions, energy use, or other physical quantities. This is useful when measured data cannot be obtained from a real building and its systems, or when the systems being modelled are too complex for manual calculations. BPS provides estimates of building performance under hypothetical conditions such as a future climate, changes in building operations, or the impact of retrofits. The ensembles of physics-based equations that make up BPS are usually deterministic: given a set of inputs, simulators will always give the same outputs. These outputs are also precise, i.e., simulators do not typically estimate the uncertainty in outputs. A BPS program can often be computationally-expensive and many design exercises require hundreds of runs for each decision, both of which are difficult to implement in practice and slow down decision-making. Important use cases where the simulator's speed of response is consequential include:

- Monte Carlo (MC) sampling for uncertainty or sensitivity quantification – simulating a building or system with several plausible values of an unknown or poorly characterised input like weather,
- parametric design exploration – testing the impact of several different variations of the

design or specification of a component like window sizes,

- stock modelling – estimating the performance of large groups of buildings, like modelling housing archetypes at an urban or national scale,
- early design phase exploration – when several consequential decisions must be made while it is impractical to set up a full simulation for each decision since too many aspects of the final building design are unknown.

To be practical in time-bound decision-making workflows, either individual simulations must be sped up or computational power increased sufficiently. While big datasets have been analysed with supercomputers for some applications [e.g., 1], most users only have access to single computers or small-scale cloud services like JEPlus [1] or NREL's PAT [2]. This means that individual simulations must be sped up. Options exist in most simulation programs to use simpler variants of underlying algorithms, these achieve only limited efficiencies. One option for speeding up individual estimates is the use of so-called *emulators*, alternative models that estimate the same quantities as the original BPS. How an emulator may be judged to be a suitable replacement for a simulator is the problem addressed in this paper.

Emulators replace, and usually simplify, the complex ensemble of physics-based equations that make up a simulator. The inputs for emulators need not be exactly the same as those used by physics-based simulators. For regression models, these inputs are called features. That is, those quantifiable features or characteristics of a dataset that can be used to characterise the variety or variance seen in the dataset. If the dataset is then representative of the physical system being examined, the features can be said to describe the physical system as well.

While the use of emulators is motivated by the need to reduce computational burden, an emulator must accurately represent the physical behaviour of the building systems being modelled to be useful. That is, the difference between predictions of performance by an emulator and simulator should be acceptably low. The objective of this work is to provide a set of tests that can be used to determine whether a simpler, faster mathematical model (an emulator) is a suitable and viable substitute for BPS.

This paper lays the groundwork for a test suite to evaluate emulators for a given problem, along with its application to a simple problem of predicting energy use. This could be extended with the development of a catalogue of results using common design problems and emulators showing, for example, how emulators will improve solutions for some problems and not for others, where they are more applicable, and whether the selection of specific emulators/algorithms can be generalised to a class of problems. The original code used for this paper is available online[3], and is open for implementation in tools.

To reduce computational time without compromising the usefulness of performance estimates, any replacement for BPS must be (i) **accurate**, (ii) **fast**, (iii) **generalisable**, and (iv) **easy** to use and setup. These criteria can be used to judge whether an emulator is a sufficiently useful replacement for a simulator. In this paper, we provide a suite of seven tests that can be used to evaluate emulators against these criteria. Namely, the model should fulfil the following conditions, further developed in Section 3.1:

(1) error compared to simulator outputs is acceptably low,

(2) performance improves with more training data (lower error),

(3) performance improves with more complex or varied training data,

(4) performance does not degrade (error does not increase) too much for a specific test case,

(5) performance is consistent across different test data sets,

(6) emulator is computationally cheaper than a simulator,

[1] `www.jeplus.org`

[2] `http://nrel.github.io/OpenStudio-user-documentation/reference/parametric_analysis_tool_2/`

[3] `www.github.com/author/repository`

(7) performs appreciably better (lower error) than a simpler model.

These tests can be used to both identify whether a specific emulator is usable or not (pass/fail) and to compare different emulators, e.g., ranking candidates by error on the test data. The tests are not a substitute for expert judgement and may give conflicting answers, e.g., a model may generalise poorly (test 5) but perform very well for a specific test case (test 4), a model may show very low error (tests 2-5) but have an unacceptably high computational burden (test 6), etc. It is also possible for a model to fail on a particular test, e.g., failing to perform consistently well across different test data sets (test 5). It is difficult to offer a generalisable rule in these cases, and users must consider the nature of the problem. What is important to a given problem determines the importance of a given test: generalisability, accuracy, speed, or ease. An emulator should do well on tests 6 and 7, since there is little point to replacing a simulator with a more expensive emulator or using a complex emulator when a simple one will do.

In the next section, we describe possible candidates for emulators, existing work, and introduce regression models. After that, Section 3 lays out the context for how and when these emulators are useful, and the example dataset used in this paper. This example dataset consists of four subsets named Breadth (B), Depth (D), Home (H), and Urban (U). Each subset will be used to demonstrate a different test. In Section 4, we show a use case: applying a class of models known as Gaussian Process (GP) Regression to the example dataset. Note that the use of GP regression or the specific dataset have no bearing on the test suite itself; they are only convenient examples chosen in this paper to illustrate the applicability of the tests. Finally, we conclude with a discussion of possible use cases and limitations of this approach. We discuss how the tests are generally applicable, and the use case is meant to serve as an example for how the proposed test suite may work in practice.

# 2 Background

In this section we outline the mathematical background for the models and tests, as well as existing work and how it relates to this paper.

BPS is best characterised as a non-linear, stochastic, causal system [2]. Linear regression models are popular emulators [e.g. 3] since they are easy to fit, use and interpret, but they could be inaccurate since the simulator they are trying to model is a non-linear system itself. For such non-linear systems, emulators based on non-linear regression are more appropriate.

## 2.1 Simulators

A typical simulation uses, as inputs, parameters such as building geometry; building envelope characteristics; Heating, Ventilation, and Air Conditioning (HVAC) system specifications; operation schedules and control strategies. We denote a set of inputs with a vector $\boldsymbol{\theta}$ containing all these parameters. To assess the performance of a building design $\boldsymbol{\theta}$, a BPS simulation also requires plausible operating conditions that the building might experience, e.g., local weather and internal heat gains from lighting, occupants and equipment loads. We denote these operating conditions by a vector $\boldsymbol{z}$. The set of inputs, therefore, is equal to $\boldsymbol{\theta}$ and $\boldsymbol{z}$ which we denote by $\boldsymbol{x} := \{\boldsymbol{\theta}, \boldsymbol{z}\}$.

Given an input vector $\boldsymbol{x}$, the goal of BPS is to estimate performance indicators, such as temperature trends, comfort indicators, energy demand, etc. In this paper, we use a common BPS problem: predicting the energy performance of a building as the sum of the hour-by-hour energy demand (power draw) over the year. This gives us a scalar energy output which we denote by $y$. Denoting the simulator by a function $f$ that takes $\boldsymbol{x}$ as input and outputs $y$, we can express the Simulator as:

$$y = f_s(\boldsymbol{x}), \quad \text{where } \boldsymbol{x} = \{\boldsymbol{\theta}, \boldsymbol{z}\}. \tag{1}$$

Since the output depends on $\boldsymbol{z}$, the choices of operating conditions is extremely important. The future operating conditions are unknown, but can still be obtained using other sources of information. For example, future weather conditions can assumed to be random draws from a probability distributions $p(\boldsymbol{z})$ which can be estimated using past weather data [4, ch. 2]. A reliable prediction of performance can then be obtained using the Monte

Carlo (MC) method:

$$\hat{y}_{MC} := \frac{1}{N}\sum_{n=1}^{N} f_s(\boldsymbol{\theta}, \boldsymbol{z}^{(n)}), \quad (2)$$

where $\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \ldots, \boldsymbol{z}^{(N)}$ are $N$ operating conditions drawn from a statistical model $\hat{p}(\boldsymbol{z})$ obtained using past weather data. The above quantity gives a reliable estimate of the energy performance because it accounts for a wide variety of operating conditions. A design process built upon such estimates results in a robust building design that is less likely to fail under extreme operating conditions.

This approach, the MC method, which might be time-consuming to complete with a simulator since each simulation may itself take hours. Even though multiple simulations may be run in parallel, the design process itself is sequential and iterative, necessitating the repetition of the $N$ simulations for each design parameter vector $\boldsymbol{\theta}$ investigated. A more common practice is to use 'typical' (average, median, representative) values for $\boldsymbol{z}$ and only perform one or two simulations. This would result in performance estimates that are low variance but are heavily biased towards typical operating conditions. A design process relying on such estimate will be less robust, since it might miss important operating conditions under which a building may perform poorly or even break down. Emulators are models of the simulator which can predict the outputs of simulations quickly and, therefore, enable the use of computationally-expensive techniques in several situations.

## 2.2 Emulators

Fitting or training emulators requires overcoming three principal challenges: the requirement of a large, varied, and representative database for training; the time and effort to specify the form and compute the parameters of the models; and inflexibility in real-world application, usually indicated by an inability to predict well on test sets [5, 6].

The emulators we discuss here are based on regression models and take the following form:

$$\hat{y} = f_e(\boldsymbol{x}), \quad (3)$$

where the emulator is represented by the function $f_e(\cdot)$, which must be estimated, and $\hat{y}$ is an estimate of the simulator output $y$. The best estimate of $f_e$ is that function in the set of functions $\mathcal{F}$ that minimizes a cost function, e.g., Mean Squared Error (MSE),

$$f_e^* = \arg\min_{f_e \in \mathcal{F}} \ \mathbb{E}_{p(y,\mathbf{x})}\Big[\big(y - f_e(\boldsymbol{x})\big)^2\Big], \quad (4)$$

where $p(y, \boldsymbol{x})$ is the joint distribution of $y$ and $\boldsymbol{x}$. Since this distribution is unknown, we approximate the expectation using the sample mean over input-output pairs for $y$ and $\boldsymbol{x}$ observed in practice. The set $\mathcal{F}$ is usually the set of all continuous and differentiable functions.

The dataset used to build a regression model should by independent and identically distributed (*i.i.d.*), though this is difficult to achieve in practice. One naive method is to use past measurements of $\boldsymbol{z}$ (if available, e.g., past weather data) as samples from $p(\boldsymbol{z})$ and run the simulator with these to obtain samples from the distribution of $y$ (e.g., Figure 9). Specifically, given $N$ measurements $\boldsymbol{z}^{(n)}$ for $n = 1, 2, \ldots, N$, we can calculate the corresponding energy outputs $y^{(n)}$ by running the simulator.

The fitting and testing of regression models to training datasets consists of four steps, as outlined in Figure 1: (A) picking a subset of data of size $N_{\text{train}} < N$; (B) estimating the hyper-parameters ($\boldsymbol{\psi}$) with this training dataset; (C) picking an additional set of data of size $N_{\text{pred}}$ for prediction; (D) using the model to predict on the test dataset and calculate error of prediction.

The structure of linear and non-linear functions, especially the GP regression models used in this paper, is discussed in Appendix A.

## 2.3 Existing Work

The existing BPS literature, including our previous work on GP regression [7], focussed on proving the utility of a specific regression method or emulator type to tackle a specific problem in BPS. A large variety of approaches have been proposed for a variety of outputs, and these can be broadly divided into two classes: 'grey box' models and regression-based models. The so-called grey box or 'reduced order' models, which use simplified physics to approximate performance [8]. These are simple to use but inflexible since a single grey-box model is

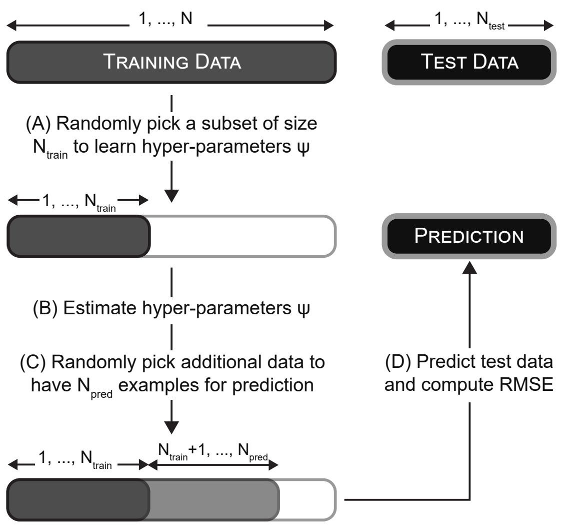



**Figure 1:** *A schematic of the method used in this paper to train and test regression models. (A)-(D) represent steps in the procedure, while the boxes represent data sets. When testing on real-world data, we use the term validation dataset to denote the test dataset.*

appropriate only for the specific system it approximates. The second class of emulators is based on regression models, constructed by fitting the model to a database of inputs and outputs obtained from a simulator or measured data [e.g., 3, 9]. Confusingly, regression models are also sometimes referred to as 'reduced order' models.

Most existing work on emulators to supplement BPS uses regression-based models. The published work largely focusses on characterising the relationship of some performance metric like energy consumption for heating or cooling to parameters/properties that may be controlled by designers, like insulation levels and window area [4, 8, 10, 11]. Examples of these kinds of studies include those that address:

(i) general uncertainty and sensitivity analysis for performance analysis and what-if analyses [4, 12–14],

(ii) computational cost, such as energy optimisation for large-scale retrofits [9], grid-scale demand prediction from buildings [15], benchmarking [16],

(iii) lack of sufficient information to run simulators reliably, e.g., prediction of the potential to harvest solar energy for neighbourhoods [17, 18], statistical evaluation of the energy performance of different office designs [19–24],

(iv) lack of certainty about future weather, e.g., prediction of indoor conditions using a small number of measured parameters [25], correlating probabilistic climate projections with office cooling demand and overheating analyses in the UK [26–28], development of "climate change amplification coefficients" to estimate resilience [29],

(v) calibration of building energy models, and fault detection and control [30–32].

# 3 Method

In this section we described the proposed tests in detail and the dataset used to show how they might be used in practice. These tests are based on general principles of statistical learning outlined in texts such as Hastie et al. [5], Rasmussen and Williams [33]

## 3.1 Proposed Test Suite

As described in Section 1, we use the following four criteria to judge if, for a given problem, an emulator is a good-enough replacement for a simulator: accuracy, speed, generalisability, and ease of use. The logic for these criteria relates to the use cases and issues discussed in the preceding sections. There are usually trade-offs between accuracy and generalisability, accuracy and speed, and between ease of use and the others. In addition, speed and ease of use are somewhat subjective and based on the problem at hand.

The tests that can be used to evaluate these criteria are all described here in terms of *error*, i.e., the difference between values (simulation outputs of interest) predicted by an emulator and those output by a simulator for the same input. We will also use the concept of test and train datasets, i.e., datasets used to fit (train) a regression model and test its performance. These tests are described for use in the process of training a regression model for a given problem. When an emulator is 'deployed' for use in a given problem, it cannot be tested except by running a small number validation simulations. The use of error to train a regression model is outlined later in Figure 1, section 2.2, and appendix A.2. We begin by summarising the tests here and describing their relevance to predicting building performance.

#### 3.1.1 Accuracy

Accuracy implies that for a given set of inputs, the emulator predicts the simulator output well. The need to predict simulator outputs accurately is obvious: if an emulator is not sufficiently accurate, it cannot replace the simulator. Given that the simulator itself is an estimate of actual performance during the lifetime of a building, introducing an unacceptably large additional error will degrade the utility of simulation-aided decision-making. We propose that the accuracy of a model during the training process can be assessed using the following tests.

**Test 1:** Error on a validation dataset is less than some acceptable tolerance of error, i.e.,

$$f_e^*(y - \hat{y}) \leq \varepsilon, \tag{5}$$

where $y$ is the output of a simulator and $\hat{y}$ is the output of an emulator for the same input, $f(\cdot)$ is some function to aggregate the differences between emulator and simulator outputs like Root Mean Square Error (RMSE) or Mean Absolute Error (MAE), and $\varepsilon$ is some acceptable tolerance for the error.

The error tolerance is a decision for the practitioner, and it may vary based on the context. For example, small random errors in the performance of individual buildings in a large stock energy model, like the Breadth dataset, will make little difference to evaluating the effectiveness of large-scale application of retrofit measures. On the other hand, error tolerance for the Home dataset, consisting of a single-family home, would be considerably smaller. The tighter the design requirements and lower the average consumption, in general, the lower this tolerance would be.

**Test 2:** Error improves with increased training data, i.e., as the number of observations available for training increase, the error on the validation set reduces. This can be expressed as an inverse correlation:

$$f_e^*(y - \hat{y}) \propto \frac{1}{N_{train}} \tag{6}$$

where $N_{train}$ is the number of observations in the training dataset.

Since the training dataset must be obtained from a simulator, with its attendant computational cost and effort, the increased investment must be justified by improvement in prediction. When applying these tests to our sample dataset in Section 4, we will show how the return on investment can diminish as the size of the training dataset increases.

#### 3.1.2 Generalisability

**Test 3:** Error improves with more complex or varied training data:

$$f_e^*(y - \hat{y}) \propto \frac{1}{\sigma_{\boldsymbol{x}}^2} \tag{7}$$

where $\sigma_{\boldsymbol{x}}^2$ is the variance of the features or inputs corresponding to the observations in the training dataset.

This test is a check against over-fitting to dataset representing a narrow set of inputs, potentially unrepresentative of the problem. For example, a model trained entirely on variations in window-to-wall ratio while everything else is held constant, like part of the Home dataset, is unlikely to estimate changes in insulation levels accurately. If the prediction on the test set improves as the training dataset includes more building or system options, designs, or scenarios, then the user has more confidence that the emulator will more accurately predict over the variety of designs and scenarios.

The training set must represent the problem to be explored. When a problem is narrowly-defined, i.e., only a limited aspect of design or uncertainty is to be explored, this test may not matter.

**Test 4:** Error does not degrade too much when moving from predicting on a general dataset during training to a validation set that is more specific to the problem at hand. In the context of this paper, this would mean training on the Breadth dataset, and predicting on the Depth dataset. When the change/increase in error is too much depends on the magnitude of the initial training error relative to that of the average prediction and the problem being studied. If an emulator fails Test 1 on a specific validation set for example, i.e., the error surpasses the tolerance set by the user, the emulator is likely unsuitable.

A dataset is more specific if it deals with only one aspect of a given design problem or exercise. For example, after training a model on variations in layouts and several building systems (Breadth dataset), we task the emulator to predict only on variations of one system or component in a specific building (Depth). If the initial error,

$$f_e^*(y_1 - \hat{y}_1) \leq \varepsilon,$$

then,

$$f_e^*(y_2 - \hat{y}_2) \leq \varepsilon, \tag{8}$$

where $y_1$, $y_2$ are outputs from two different validation datasets and $y_2$ is the result of testing on a validation dataset $\boldsymbol{x}_2$ more specific to a problem than $\boldsymbol{x}_1$, the corresponding validation dataset for $y_1$.

**Test 5:** Error does not degrade too much when testing on a more complex validation dataset, i.e., one with more variety of inputs. If the initial error,

$$f_e^*(y_1 - \hat{y}_1) \leq \varepsilon,$$

then,

$$f_e^*(y_2 - \hat{y}_2) \leq \varepsilon, \tag{9}$$

such that,

$$\sigma^2_{\boldsymbol{x}2} > \sigma^2_{\boldsymbol{x}1},$$

where $y_1$, $y_2$ are outputs from two different validation sets, $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ respectively, and $\sigma^2_{\boldsymbol{x}}$ is the variance of an input validation set. Here, $\boldsymbol{x}_1$ may represent a single archetype building used for training a model while $\boldsymbol{x}_2$ would represent a portfolio of buildings that should conform to the same archetype but with a variety of designs and systems. For example, creating variations on one archetype office building by varying the properties of different systems as in Depth dataset.

#### 3.1.3 Speed

**Test 6:** Emulator is computationally cheaper than a simulator. Regardless of the computing infrastructure being used, an emulator should be cheaper, and therefore faster, to run in order to justify accepting the increased error in estimation of output. If an emulator performs particularly well on this test, it may also be suitable for implementation in Building Management (Automation) Systems (BMS) controllers for Model Predictive Control (MPC) and similar low-resource applications that require rapid response. The computational complexity of the emulator,

$$\mathcal{O}_e(aN^b_{\text{val}}) \leq \mathcal{O}_s(cN^d_{\text{val}}), \tag{10}$$

where $\mathcal{O}_e(aN^b_{\text{val}})$ is the complexity of the emulator that scales with the number of observations in the validation dataset with some exponent, and $\mathcal{O}_s(cN^d_{\text{val}})$ is the complexity of the simulator predicting over the same observations.

#### 3.1.4 Ease of use

**Test 7:** Emulator is appreciably better than simpler methods. This final test ensures that the simplest method that delivers adequate performance is used. As discussed in Figures 10 and 11 and appendix A.1, more complex models will generally tend to overfit to a given dataset. The error of an emulator

$$f_{e1}^*(y - \hat{y}) \leq f_{e2}^*(y - \hat{y}), \tag{11}$$

where $f_{e1}^*(\cdot)$ is a higher-order model (more complex, more parameters) than $f_{e2}^*(\cdot)$. Models with fewer parameters are both cheaper to train, so would also be quicker to train and deploy (Equation (10)).

### 3.2 Data

To demonstrate the use of the test suite proposed in this paper, we generated a large dataset by simulating numerous combinations of buildings and weather conditions from four different simulation exercises, labelled ‘Breadth’, ‘Depth’, ‘Home’, and ‘Urban’. This dataset does not comprehensively represent all the possible use cases for emulators discussed in Section 1, and it does not need to. Instead, the complete dataset gives enough variety of

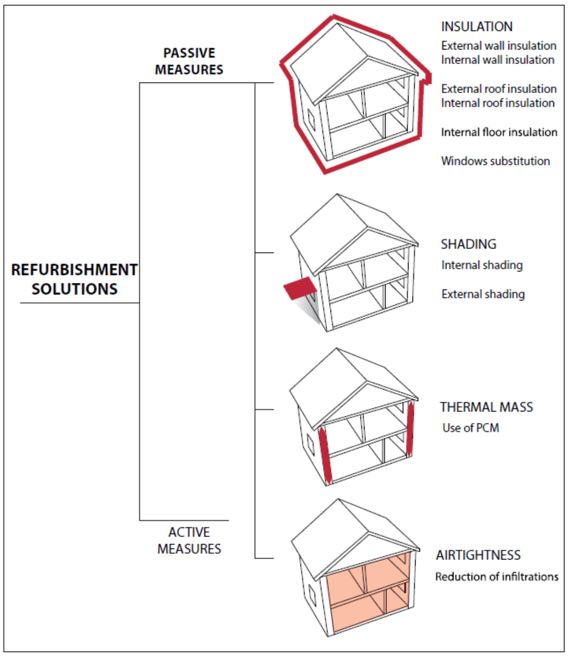



**Figure 2:** *The envelope variations simulated for the single-family home (H). Original figure in Rastogi [4].*

use cases to demonstrate the test suite proposed in this paper. Specific subsets are also used for individual tests, e.g., Breadth and Depth for Test 4 (Section 4.4).

These combinations results in approximately 600,000 data points, details of which are in Table 1. Each simulation calculated the space heating and cooling requirement for the given combination of building design and weather. The total energy for heating and cooling was divided in each case by the area of the building (normalised by area) to make the outputs of all the simulations comparable. These simulations are also described in detail in separate publications and summarised in Rastogi [4, sec. 4.2][4].

The size of the datasets used in this paper is not an indication of the minimum sizes required for each test. As the figures show in Section 4, the result for a test is usually obvious using a fraction of the data shown in this paper for demonstration.

The Breadth (B) and Depth (D) case studies are from the United States Department of Energy (USDOE) commercial buildings reference database [34]. The Breadth dataset consists of 16 different building types. This dataset is a representation of a national or regional building 'stock', i.e., a representative sample of buildings. The Depth set consists of simulations on one of the building types in the USDOE database: the 'medium office'. This dataset is a simulation of a design exercise varying envelope properties. These properties were: U-value (practically, changing the thickness of insulation material), thermal mass (nominal quantity of internal mass), shading (obtained by varying the depths of overhangs and fins), permeability (changing infiltration levels), and transparency (changing the Window-to-Wall ratio).

The Urban case (U) is composed of a set of buildings constructed over a century (1900-2010) in the centre of Geneva, Switzerland [35]. This case was chosen because it includes the urban context surrounding the buildings. We expect the urban context to add noise to the data, since we have not included any features that explicitly describe the influence of the surroundings. The buildings were all modelled in at least two variants: with the original envelope and with an envelope upgraded to the latest Swiss standards for infiltration, insulation, etc.

Finally, the single-family home case (H) is an example of a very simple simulation study, one where we expect the response to be characterized well-enough by a linear regressor. The changes to the house are described in Figure 2. The simulation model of the house is based on an actual home in north-central Germany [36].

The weather data used is of three types: recorded data from the Integrated Surface Database[5], typical years [38, 39], and synthetic weather time series [4, 40, 41].

[4]The data may be downloaded from `https://doi.org/10.5281/zenodo.291858`

[5]`https://www.ncdc.noaa.gov/isd`

**Table 1:** *List of datasets. The size indicated here may be slightly different from the amount of data used in the scripts due to the presence of invalid data entries. Less than 5% of the entries were invalid in any dataset. Mean values for annual sum of Heating and Cooling loads were calculated over all valid values in* $kWh/m^2$.

| **Name** | **Size** | **Description** | **Ref.** | **Mean Heat** | **Mean Cool** |
|---|---|---|---|---|---|
| Breadth (B) | 88,242 | USDOE commercial reference buildings (all building types) | [4, 34] | 131.63 | 48.97 |
| Home (H) | 77,934 | Single-family home, Central European construction | [4, 36, 37] | 23.41 | 74.84 |
| Depth (D) | 445,334 | Variations on the medium office building from the USDOE database | | 102.90 | 30.98 |
| Urban (U) | 6,003 | Mixed-use buildings in Geneva, Switzerland (with surrounding buildings) | [35] | 134.27 | 0.00 |

# 4 Results

In this section we show the application of the proposed test suite to judge the suitability of GP regression models with linear and non-linear kernels, using the dataset described in Section 3.2. The kernels are listed and described in Table 2.

The fitting and testing of GP models to large training datasets consists of four steps, as outlined in Figure 1: (A) picking a subset of data of size $N_{\text{train}} < N$; (B) estimating the hyper-parameters ($\boldsymbol{\psi}$) with this training dataset; (C) picking an additional set of data of size $N_{\text{pred}}$ for prediction; (D) using the model to predict on the test dataset and calculate error of prediction. Before beginning the fitting procedure, we set aside a portion of the total data available to us as 'test' data of size $N_{\text{test}}$ (approx. 60% of the dataset). The amount of training data $N_{\text{train}}$ used to estimate the hyper-parameters ($\boldsymbol{\psi}$) was varied from 50 to 4000 for each dataset and model (except for models fit to the Urban dataset individually). For a given training size $N_{\text{train}}$, we draw that many observations from the large overall training set and use it to train a model. We repeat this process 100 times to obtain an empirical distribution of the RMSE of each model's predictions on the test set. For Test 3 (Section 4.3), results from fitting to datasets much larger than 4000 are presented (e.g., Figure 5). In this case, for runs with $N_{train} > 2000$, only $N_{train} = 2000$ was used to calculate hyper-parameters, while a separate subset of the training dataset, $N_{pred}$, was used to fit the model (see Appendix A.4.2 for details). In this case, both $N_{pred}$ and $N_{train}$ are training datasets. This allowed for models fit to much larger datasets than our computer could handle if the entire process were carried out with these very large datasets.

We use Root Mean Square Error (RMSE), the square root of the MSE term from Equation (4), to quantify errors. An advantage of RMSE is that it has the same units as the original outputs, in this example kWh/$m^2$, which makes it easier to understand and judge magnitudes of error. Readers are invited to compare plotted errors against the means of the datasets given in Table 1, and against the overall mean of 51.49 kWh/$m^2$ and 64.31 kWh/$m^2$ for annual heating and cooling loads respectively. In the figures, we present the distribution of the RMSE calculated over a hundred subsets of the test set as an additional check on the reliability of a model: if the distribution of RMSE over subsets of a test set is too wide, the model does not reliably represent the range of possible values of inputs to be tested, i.e., the test dataset. The test dataset is entirely separate from the training dataset (Figure 1). The RMSE plotted in each graph below was calculated solely on the test dataset, i.e., test error.

Each test is presented as it would be applied to the problem of quantifying the uncertainty in predicted energy use of a building design for space conditioning due to lack of knowledge about fu-

**Table 2:** *List of models compared in this study.*

| Model | Description |
|---|---|
| Mean | Mean of the outputs $y_n$ |
| Lin-ISO | Linear model with isometric kernel |
| Lin-ARD | Linear model with automatic relevance determination |
| NonLin-ISO | Squared-exponential isometric kernel |
| NonLin-ARD | Squared-exponential kernel with automatic relevance determination |

ture weather conditions. Each design is intended to represent a choice available to a designer at construction or renovation, and the output of interest is the impact of the design choice on annual whole-building energy performance over the lifetime of the building. The uncertainty would be quantified using the MC method, sampling plausible weather scenarios and simulating them for a given building design to obtain a reliable estimate of energy performance (Equation (2)). While there is no definitive number of operating conditions (weather files) that must be simulated for an estimate to be reliable, we have found that stable results could be obtained with about a hundred simulations with random weather conditions per building design. Depending on the complexity of the building designs and systems used in this paper, each simulation took 15 minutes to almost 2 hours.

## 4.1 Test 1: Error on validation set

The acceptable level of error depends on the problem being explored. Fitting regression models always involves a trade-off between minimising the error on the available training dataset and ensuring that this does not overfit the model to the specific dataset. This is a form of the bias-variance trade-off common to all statistical learning approaches [5], as discussed in Section 2.1.

## 4.2 Test 2: Error with larger training dataset

This test is unambiguous and straightforward for this example: the validation error improves with size of training set for all cases described in Table 1. For example, see the changes in prediction error when predicting heating loads on the Breadth and

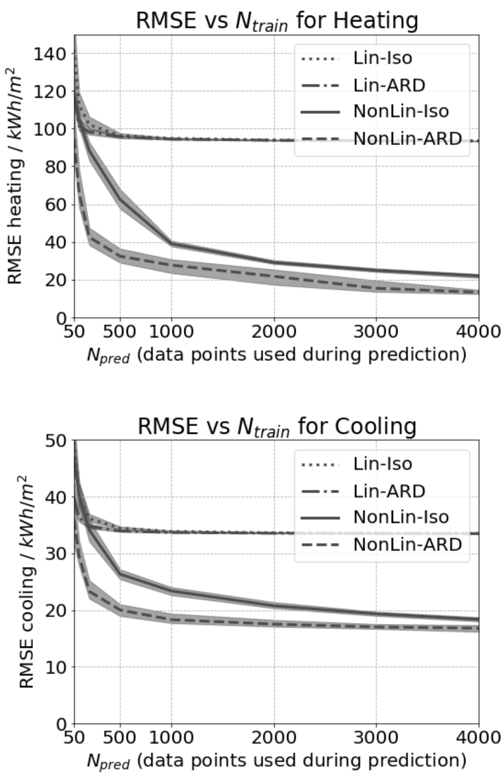


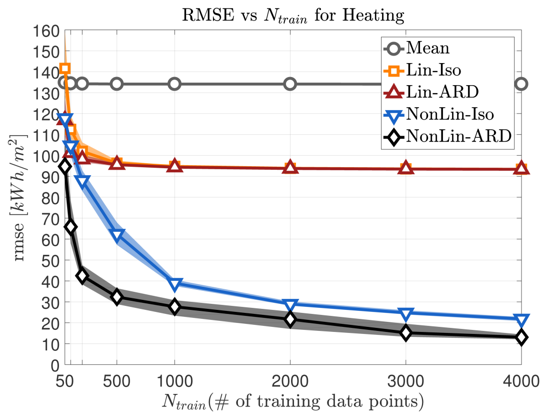

**Figure 3:** *RMSE for heating [top] and cooling [bottom] models fit to the Breadth database. The (test) errors are calculated on approx. 52,940 points, and plotted against the size of the data set used at prediction (same as the dataset used to learn hyper-parameters in this case, i.e., $N_{pred} == N_{train}$). The lines indicate median errors, and the filled areas are bounded by the 25th and 75th percentiles.*

Home datasets, plotted in Figure 4 for each model type. Compare the errors with those from the simplest possible predictive model: the mean of the training outputs, $\bar{y}_{\text{train}}$.

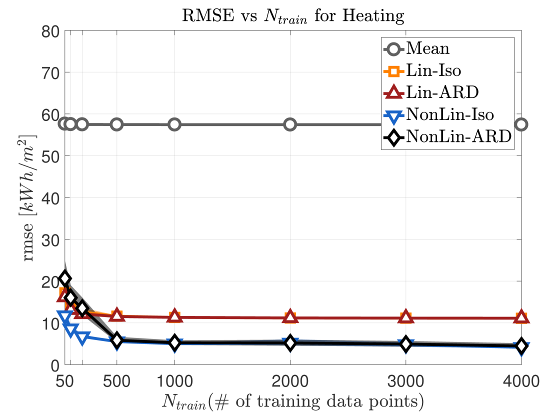
### 4.3 Test 3: Error improves with training data variance

For this test, we combined all datasets to represent the case where a complex, varied problem is to be modelled and large number of simulation runs are available. Ideally, we would like to use all the data available for training. However, learning hyper-parameters is infeasible for large numbers of data points because each step in the learning procedure requires the inversion of an $N_{\text{train}} \times N_{\text{train}}$ matrix. We use a simple trick to make use of the additional data, outlined in Figure 1 and based on the proposal in Chalupka et al. [42]. We learn the hyper-parameters on a dataset of size $N_{\text{train}} = 2,000$, but during prediction we use a much larger dataset $N_{\text{pred}} = 2000, \ldots, 12000$. Since prediction requires only one matrix inversion, the latter step is still feasible for dataset sizes of about 10,000 on the hardware we used for our study.

In Figure 5, we present the results of fitting and testing a non-linear GP model to the whole dataset (i.e., all subsets described in Table 1). Thus the differences between the results presented in Figure 3 and those presented in Figure 5 are that, firstly, in the latter figure we use a larger dataset that is more representative of real-world outcomes, and secondly, we show the additional advantage of using a larger dataset for prediction during training ($N_{\text{pred}} > 0$ in Step C of Figure 1). In Figure 3, the RMSE values were plotted against the number of data points used to learn hyper-parameters and subsequently fit the model to the same dataset. In Figure 5, the RMSE values are plotted against the size of the prediction dataset (different from the fixed number used in learning hyper-parameters). The solid lines (with shaded curves) show the RMSE obtained when 2,000 data points are used for learning but a larger set is used during prediction. We see that RMSE decreases as the number of data points used for prediction is increased. Linear models are not presented here since this procedure makes no difference to their performance.




**(a)** *Breadth*




**(b)** *Home*

**Figure 4:** *Evolution of RMSE for heating loads with increasing $N_{train}$. Non-linear models perform better than linear models for all datasets, and also show more improvement with increased training data. A separate, larger dataset was not used for prediction in this case since the sizes of $N_{train}$ were still tractable.*

We see that the use of non-linear GP models on a large dataset is both feasible and accurate. We will now check the performance of these models on two sets of problems: predicting for a specific building (individual) or on a variety of buildings together.

### 4.4 Test 4: Error on a specific validation set

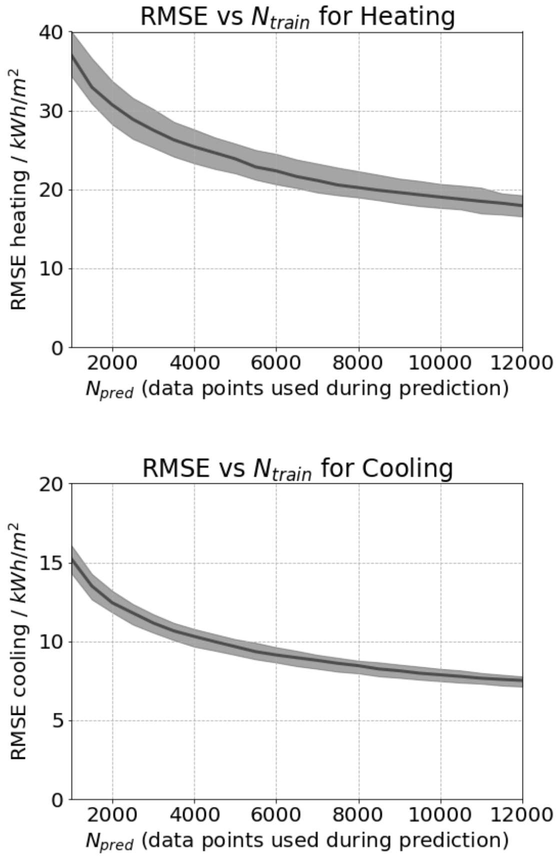



**Figure 5:** *These figures show the RMSE for heating and cooling loads for the combined dataset, plotted against the size of the training data set used at prediction (Appendix A.4.2). The lines indicate median errors, and the filled areas are bounded by the 25th and 75th percentiles. The non-linear models outperform linear models, and the predictions improve with size of training data set.*

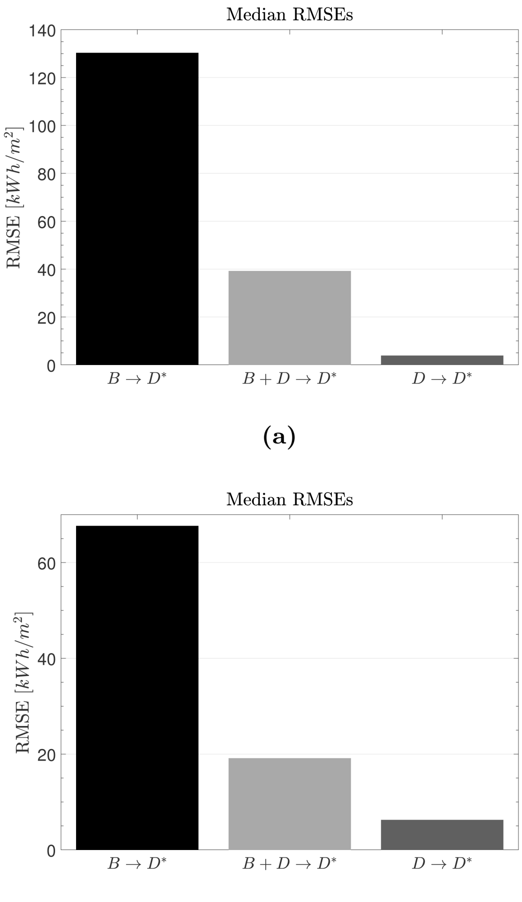



**Figure 6:** *Heating [top] and cooling [bottom] predictions when predicting on a specific building. $B \to D$ means training hyper-parameters on the Breadth dataset and testing on Depth, $B + D \to D$ implies training on a combination of the two and predicting on Depth, while $D \to D$ implies both training and testing on exclusively the Depth dataset.*

In many applications, the designer might be interested in predicting the performance of only a specific type of building. In that case, it is possible that an emulator trained on a variety of building types may not perform well. We show that a non-linear model performs well if it has 'seen' enough buildings that are similar to the one we want to predict. The results presented in this section establish the satisfactory performance of the GP regression models used here for this test.

We consider the task of predicting the performance of a building in the Depth dataset, the results of which are shown in Figure 6. This dataset contains only a specific type of building: a medium-sized office. We used 60% of the Depth dataset as the test set ($N_{\text{test},D^*} = 247,304$). We trained three models: the first using only the Breadth dataset ($N_{\text{train}} = N_{pred} = 1000$), referred to as '$B \to D^*$' in Figure 6; the second using only the Depth dataset ($N_{\text{train}} = N_{pred} = 200$), '$D \to D^*$'; and the third using a combination of the Breadth and Depth datasets (1000 data points from Breadth and 200 data points from Depth), '$B + D \to D^*$'.

We see that when using only the Breadth dataset for training and prediction, the model performs badly. Adding some points from the Depth dataset significantly improves the performance of the model, since this addition reduces the influence of those points in the Breadth set that do not come from the office building. Using only the Depth dataset gives the best performance. This shows that training an emulator with a small number of examples for a specific building is enough to predict well for that building ($N_{\text{train}} = 200$ in this case). When learning on a variety of buildings and predicting on a specific one, the performance is poor with naive selection of points during prediction [left-most bar in each graph]. Performance improves by adding simulations from the specific building [middle bar]. Performance is best when using training data only from that building [right-most bar].

A GP model predicts on a new test point by correlating the test inputs to training inputs. The kernel is supposed to encode the influence of different training points in predicting the test point: the more closely related a group of training points is to a test point, the more influence they ought to have on the prediction. We find that this is not the case for our study when we try to use training points exclusively from the Breadth dataset to predict on a test set from the Depth dataset. However, the results improve dramatically if new training points are added from the Depth case. This suggests that explicitly encoding, perhaps with a categorical variable representing building type/usage, the 'closeness' of a new test point to a subset of training points, should improve prediction.

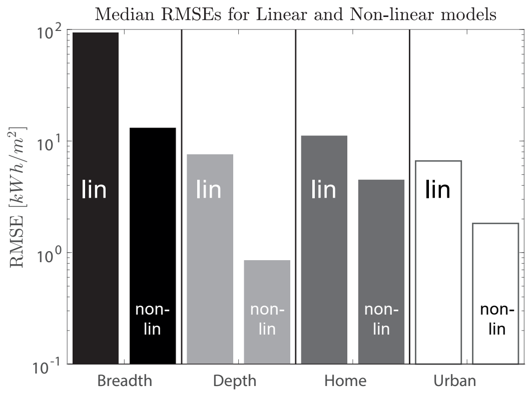
### 4.5 Test 5: Error on a varied validation set

We now present results to show that the non-linear emulators presented here can be trained to obtain accurate predictions for a variety of datasets, e.g., different buildings in different climates.

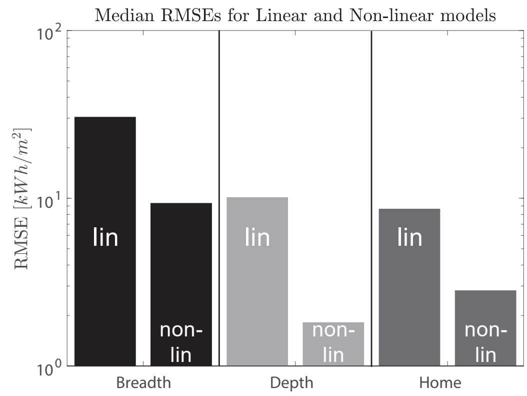
Figure 7 shows the results for all datasets separately using $N_{\text{train}} = N_{\text{pred}} = 4000$ data points, of which 60% are used as the test set for each case. We present results for two models: Linear automatic relevance determination (ARD) (*lin*) and Squared Exponential ARD (*non-lin*). The RMSE obtained from using the non-linear model is of the same order of magnitude for all datasets (1-10 kWh/$m^2$) and uniformly better than RMSE from linear models. These results demonstrate the flexibility of the non-linear model compared to an equivalent linear model.

### 4.6 Test 6: Computational Expense and Time

Figure 8 shows the distribution of computer time taken by each simulation in the dataset used for this paper. A single iteration of the Monte Carlo method for a single design, i.e., about 100 simulations, would take between 25 and 200 hours (0.9e5 to 7.2e5 s). In addition, the time taken for each iteration would not change, since the simulations cannot be reused. For comparison, GP regression takes 1 microsecond (1e-6 s) to provide one estimate, so each MC iteration would take about 10 milliseconds (1e-3 s). This comparison is for run times, i.e., assuming that the regression model has already been fitted. As we saw in Sections 4.1 to 4.4, reaching a satisfactory error rate in this example requires a training dataset of at least 1000-1500 simulations. Assuming a fresh start for each problem, a worst case scenario where no knowledge from previous simulations is transferable to a new problem, would




**(a)** *Heating*




**(b)** *Cooling*

**Figure 7:** *Median RMSE at $N_{train} = 4000$. The y-axis uses a log scale. For each dataset, the bar on the left is for the linear model and on the right for the non-linear model. Non-linear models perform than linear models for all datasets. The Urban case was modelled for Geneva, where buildings do not typically include cooling systems (air-conditioning).*

require the user to use the simulator for some part of the experiment.

When accounting for the cost of using an emulator, both the cost of obtaining training data (usually from the simulator) and of fitting the model should be included. This means that emulators are not suitable for short and quick design exercises unless that exercise is part of or similar to a problem for which a model has been trained. This creates a strong incentive for the use of pre-trained libraries of models suitable for specific problems, especially in situations where there is insufficient information to create simulation models (Section 2.3).

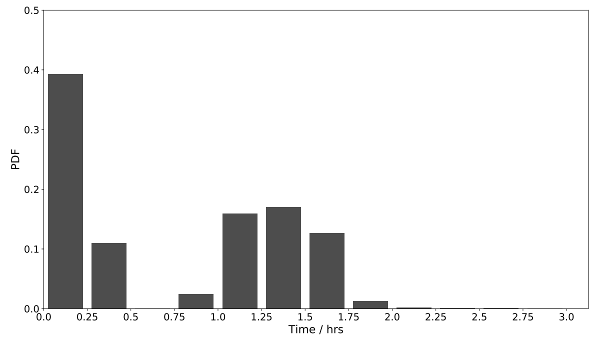



**Figure 8:** *Probability Distribution Function (PDF) of the approximate time taken to run one simulation in a simulation exercise carried out by the authors using EnergyPlus v8.8* (https://energyplus.net/).

### 4.7 Test 7: Simplest emulator

Non-linear models outperform linear models in all of the results presented here (Figures 3 to 5), and both model types outperform the mean of the training set outputs. The predictions also improve with size of training data set. Non-linear GP-based emulators perform equally well when predicting on a diverse set of buildings (Breadth) as a dataset consisting of a single building (Depth, Home) or small set of very similar buildings (Urban).

## 5 Discussion

In this section we discuss the applicability of this approach, limitations of data-based approaches for emulators, and practical issues around selecting data for training.

## 5.1 Generalisability of Approach

Designers and analysts of buildings and their systems develop professional judgement and intuition about the physics of the systems they study. This allows experts to identify common errors in numerical simulations and improves the reliability of results. The approach proposed in this paper presupposes some knowledge of how appropriate datasets may be acquired. Data-based methods are not substitutes for knowledge about the physics of the problem. A tool like regression, which does not use the physics of BPS, will require users to learn new skills in handling and interpreting statistical learning models. An example of this is the difficulty of physically interpreting the dimensions of the (mathematical) space of features, i.e., the numerical representations of the values of different building properties or characteristics along numerical axes. When collecting data on complex problems, the characteristics or features of interest may be many, which makes it difficult to maintain intuition about which designs (combinations of features) are similar to others in the feature space. This cannot easily be overcome with human judgement, a problem we encountered in the creation of the dataset used here as well. Thus, the use of these techniques will benefit from the development of services and tools which suggest methods for efficient data-gathering and training. While some progress has been made recently in moving more simulation programs to cloud-based services that remove much of the complexity of setting up models for the user, the possibility of augmenting these services with regression models to improve their utility for computationally-intensive problems remains to be explored.

The tests were demonstrated here using a single output: energy use for space conditioning. The use of these tests could be more complicated for multiple outputs of interest, e.g., comfort and energy use. Using multiple outputs could be handled with additional ranks or weights for different priorities, combining the results of the tests for different outputs for a single decision. However, given that the tests are comparing outputs from the *same physical system*, it may not always be the case that the results of applying these tests to different outputs would be different. Additionally, since the user would already have invested the effort to fit models at that point, they can also use different models for each output. There is no reason to suppose that different emulators that work for different outputs would deliver inconsistent decisions.

## 5.2 Limitations of Data-driven Methods

Data-driven methods are not fail-proof; the model learnt on one dataset does not necessarily translate perfectly to another (Sections 3.1.2, 4.4 and 4.5). An emulator does not incorporate any knowledge about the physics of the problem being simulated, which means that emulators are, by construction, usually less flexible than the simulator. Regression inputs may not be representative or may not explain the variation in the data properly, which leads to inaccurate predictions and the inability to generalise. Finally, the dataset used for training might itself have a systematic bias. That is, a dataset that does not represent the problem properly, or is not a good proxy for real-world problems.

The best regression model contains just the right predictive inputs, for the selection of which there are no fixed rules that will apply to every problem. The automatic relevance determination (ARD) procedure [33, sec. 5.1, and references therein] used in this paper allows the user to begin with a large set of input variables that might be important to a problem, letting the GP estimation procedure trim that number. However, ARD does not necessarily follow the physics of the problem either and may not, therefore, generalise to other problems. It is important to include all of the design parameters (input variables) which are expected to be relevant in the modelling of the energy performance (output). This is both to ensure a good fit to data and relevance to the design problem. Including too many inputs, however, makes it difficult to obtain a good fit with a manageable size of training dataset(the so-called *curse of dimensionality*).

## 5.3 Collecting Data for Training

The quality of a dataset is determined by how faithfully it represents the true distribution of data $p(\boldsymbol{x}, y)$ (assuming this distribution exists). However, since the true distribution is usually unknown, it is impossible to measure the quality of a dataset objectively. To obtain a good-quality dataset of simulations of building designs, we can rely on designers (domain experts) who may, for example, choose realistic building designs and weather conditions from their portfolio and experience. The quality of dataset is, therefore, defined by the goal of the designer. For example, a designer interested in predicting performance over a variety of design parameters and weather conditions might need to acquire a dataset that contains similar examples in the training set. On the other hand, a designer who is interested in studying only a particular type of building might want to limit the dataset to that particular building. The results in this paper show that if the training dataset is too general, i.e., it contains too many examples dissimilar to the one considered during prediction, then the emulator does not perform well. This underlines the need to train separate models on different design problems. Thus, the objectives of Test 3 are often in conflict with the objectives of Tests 4 and 5.

In principle, there should be no correlation with, or effect of, building type, usage, or location on the input variables $\boldsymbol{\theta}$. However, if the training sample is taken from realistic buildings in a given climate, the distribution of the input variables will be influenced by prevailing architectural idioms and other cultural and practical factors. For example, sampling many houses from a single region will not necessarily cover all possible values of wall conductance (U-value), since houses from a region typically follow local trends and laws concerning insulation level [e.g., 43]. Similarly, while modern office buildings may have up to 85% window-to-wall ratio ($WWR$), homes with the same proportion of window area are rare. At the other extreme, a value of less than 10% $WWR$ is theoretically possible for any kind of building, but windowless buildings are so rare as to be statistically insignificant. Sampling from a particular type of building does not necessarily mean that the distribution of input variables will be identical to sampling from a different type/usage. In addition, each type/usage has cultural or regional limits on the values of input variable seen in practice, e.g., buildings will probably not include a layer of insulation in the walls when it is not appropriate for a given climate.

An example of selection bias is in the Breadth dataset used in this paper (Figure 9). There is a preponderance of lower values of annual heating and cooling energy usage because the set has more moderate climates than extreme ones. The climates were selected based on data availability [4], prioritising cities with several years of recorded data. These tended to be urban areas with major airports in continuous operation for decades either due to large, established populations or strategic reasons. The cities in the database have a combined population in excess of 200 million, though future work should incorporate weather from a wider selection of world climates.

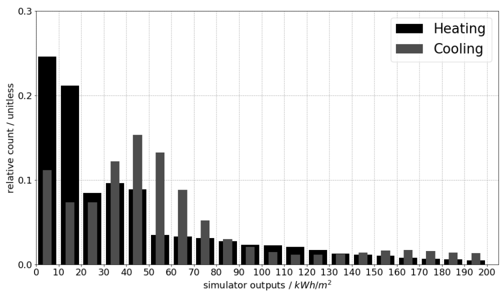



**Figure 9:** *Distribution of simulator outputs (heating and cooling loads) from the overall dataset. There are more moderate loads than extreme ones in the dataset.*

These limitations in data quality and representativeness can be overcome by including a large amount of simulated data and/or updating the models using measured data. The flexibility of regression-based emulators, as demonstrated in this paper, means that different datasets can be easily integrated into the model to improve its results. This is in contrast to the original simulator, where the results of one simulation do not have any impact on the results of another.

# 6 Conclusion

This paper has proposed a new test suite for standardising the evaluation of emulators as suitable replacements for building performance simulators in a variety of use cases, especially uncertainty quantification. The use of emulators is promising for applications where the speed of response from an evaluation is important, provided the emulators are sufficiently accurate. Thus, the test suite is presented in the context of evaluating an emulator using four criteria: accuracy, generalisability, speed, and ease of use. We do not propose a specific model or class of models for the dataset used here, or any real-world problem exemplified by this dataset. Rather, we show how emulators may be evaluated in a given context, regardless of the structure of the problem or the dataset used to characterise it.

As an example of how the test suite may be used, we showed that non-linear models noticeably and consistently outperform linear models in emulating example specific and broad datasets. In all cases, the non-linear regression models show a Root Mean Square Error (RMSE) between 10-15% of the mean model (output from a model which consists of only one term: the mean of the training data). The GP regression models are able to predict well on a dataset consisting of a variety of buildings as well as a dataset consisting of a specific building. We find that the predictive performance of non-linear GP regression models is stable and repeatable. We showed procedures to use large datasets for learning and predicting with the same models on unseen data.

Not all steps of a typical process require simulation, as designers make several decisions based on meeting existing laws, user needs, and functional requirements. The use of numerical simulation has expanded considerably with the advent of simulation tools or workflows offered as a service to non-specialists looking to carry out specific analyses [e.g., 44, 45], better diffusion of numerical and computational skills, and better interoperability between the models created by different professions. However, over-reliance on simulation tools for prediction rather than comparative what-if analyses, and excessive trust in results based on testing under limited operational conditions, e.g., typical weather files, can lead to a severe gap between expected and actual performance. The quantification of a possible cause of this gap can be partially addressed through the use of regression-based emulators. The use of these emulators can, in turn, become more systematic and widespread with the adoption of standard operating procedures such as the test suite proposed in this paper.

# A Regression Models

This appendix discusses the mathematical background of linear and non-linear regression models. Details are also included on the structure of GP regression models, how they may be fit to data and used, and a practical workaround for big datasets.

## A.1 Linear and Non-linear Regression Models

Non-linear regression models are more flexible and have the potential to model the output of a non-linear system more accurately. They can also account for the complex interactions of the large number of inputs that determine the outputs of a simulation. However, fitting non-linear models is computationally challenging, especially when a large amount of data is available. Ironically, a large amount of data is almost essential to obtain a good performance of the non-linear model, otherwise they might over-fit the data in hand [5, 6]. Another issue is that specification of non-linear models is difficult and requires a lot of effort and domain expertise. That is, for some problems where adequate data cannot be obtained within the budgetary or time allocation of an exercise, or the number of properties or factors for each test subject is limited because of the quality of data, non-linear models may not work [e.g., 3]. Some of these challenges are described in Section 3.2, exemplified by the data collected for this paper.

Non-linear regression models use more parameters than linear models, which increases both the time to build the model and the size of the training data set required to calculate the parameters of a model. For example, a non-linear model such as an Artificial Neural Network (ANN) could have millions of parameters, which means that the dataset required to estimate all parameters must be of the same order of magnitude. In addition to the larger number of parameters, the space of possible non-linear functions that can fit a given dataset is also larger. Thus, linear models, with fewer possible functions and fewer parameters to specify those functions, are simpler and easier to fit.

Linear models are not flexible enough to estimate non-linear systems such as a building performance simulator. This means that the estimates of linear models are precise but may be inaccurate. The flexibility of non-linear models, on the other hand, means that the data from a non-linear system can be estimated more accurately. However, this flexibility could lead to over-fitting the model to the dataset at hand. The practical consequence of this would be a failure to predict well on real-world data different from that included in the training dataset.

This problem of generalisability of models could arise from over-fitting to a small amount of data, or an unrepresentative dataset. In the context of BPS, such an unrepresentative dataset could consist, for example, of only one building type or weather (context/location). This would make the model inaccurate for other building types or locations. The use of a larger, varied dataset can reduce this problem to some extent (see Figures 10 and 11 for an example), since it would make it more likely that the model would see examples of more real-world situations.

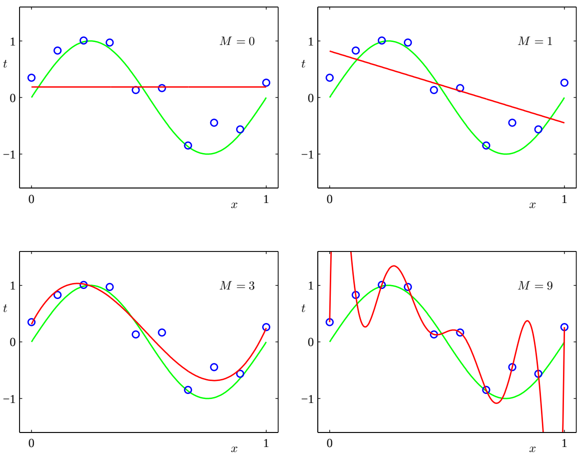



**Figure 10:** *A generic representation of the tendency of non-linear models to over-fit data from Bishop [6]. In general, the more complex a model is, the more it will over-fit the data at hand. The green curve is used to generate the data and the red curves show a polynomial fit. Polynomials of progressively higher degrees (*$M = 0, \ldots, 9$*) fit the data better but are probably over-fitting.*

## A.2 Linear Models

The function $f_e(\cdot)$ from Equation (3) may take the form of a linear model:

$$\hat{y}_l = f_l(\boldsymbol{x}) := \boldsymbol{\beta}^T \boldsymbol{x}, \qquad (12)$$

where $\hat{y}_l$ is the prediction of the output at input $\boldsymbol{x}$ obtained using a linear function $f_l(\boldsymbol{x})$ which is specified using $\boldsymbol{\beta}$, a real-valued parameter vector (of the same size as $\boldsymbol{x}$).

The parameter $\boldsymbol{\beta}$ is unknown but can be estimated using a *dataset* of the input-output pairs, e.g., obtained by running many BPS simulations on a plausible set of building designs $\boldsymbol{\theta}$ and its operating conditions $\boldsymbol{x}$. In a standard machine-learning framework, we first collect a large amount of such data: $\mathcal{D} := \{y_n, \boldsymbol{x}_n\}_{n=1}^N$ where $n$ denotes the $n$'th BPS simulation. Given a dataset $\mathcal{D}$, we may use the standard training-testing framework developed in statistics and machine learning [5] to estimate $\boldsymbol{\beta}$. In this framework, first, the $N$ observations are split into two mutually-exclusive sets: training and testing. We denote the training set by $\mathcal{D}_{train}$ which contains $N_{train}$ number of observations. Similarly, we denote the test set by $\mathcal{D}_{test}$ which may contain $N_{test}$. In this paper we use the term *validation* set to denote the dataset used for real-world testing of the model. By construction, $N = N_{train} + N_{test}$. The training set is used to *train* the linear model, i.e., to estimate $\boldsymbol{\beta}_*$ by minimizing a cost function, e.g., a mean-square error as shown below:

$$\boldsymbol{\beta}^* = \arg\min_{\beta} \frac{1}{N_{\text{train}}} \sum_{n=1}^{N_{\text{train}}} \left( y_n - \boldsymbol{\beta}^T \boldsymbol{x}_n \right)^2, \qquad (13)$$

This gives us a linear model $f_l^*(\boldsymbol{x}) := \boldsymbol{\beta}_*^T \boldsymbol{x}$ which can be used to predict the new inputs. The test set is then used to assess the *goodness-of-fit* of the estimator by computing the following cost,

$$\mathcal{L}(\hat{f}_l^*) = \frac{1}{N_{test}} \sum_{n=1}^{N_{\text{test}}} \left( y_n - \boldsymbol{\beta}_*^T \boldsymbol{x}_n \right)^2, \qquad (14)$$

This is the *test error* which could be a faithful estimate of the real-world prediction error of the model when $N_{test}$ is fairly large and representative of the real-world problem.

An advantage of a linear model is that training is easy. Equation (13) has a closed-form solution which can be obtained by using the ordinary least-squares method. This method scales well for medium-sized datasets and can also be extended to large datasets by using iterative methods such as stochastic gradient descent [46]. Another advantage of the linear model is that it is fairly straightforward to specify and interpret. An entry in the parameter $\boldsymbol{\beta}$ is a direct indicator of how important the corresponding entry in input $\boldsymbol{x}$ is for the linear model to predict well. Unfortunately, linear models are not good models of the simulator since the simulator is a non-linear model itself. As a result the test error $\mathcal{L}(\hat{f}_l^*)$ is usually quite large, except for the simplest problems as discussed in Section 2.3 above.

## A.3 Non-linear models

Estimating the $\hat{f}_e^*$ of Equation (4) is easier for linear regression models since closed-form methods like least-squares may be used [5, 6]. However, estimating the same quantity for non-linear models requires the use of iterative methods, which is time-consuming. The time to train parameters rises rapidly with the number of parameters to be estimated. As discussed in Section 2.3, several types of non-linear model types have been proposed for BPS. The recently-concluded ASHRAE Great Energy Predictor III challenge [47] alone saw 415 solutions submitted. Next, we discuss the structure of a non-linear model and the process of fitting it to a given dataset using a general-purpose model type: Gaussian Process (GP) regression.

## A.4 Gaussian Process Regression

We use the framework of GP regression to estimate the non-linear function $f_{nl}$ that minimizes Equation (14). GP regression uses Bayes' rule to compute the posterior distribution over $f_{nl}$ given sample outputs $y_n$ [33, ch. 2]. This approach works directly in the space of $f_{nl}$ and avoids both a direct estimation of $\boldsymbol{\beta}$ and also a direct specification of $\boldsymbol{\phi}(\boldsymbol{x})$. Instead, we specify a 'kernel' function which defines the inner product of $\boldsymbol{\phi}$ as

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\phi}(\boldsymbol{x}_i)^T \,\boldsymbol{\Sigma}\, \boldsymbol{\phi}(\boldsymbol{x}_j), \qquad (15)$$

where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are two inputs in our observation set. In practice, a kernel function is easier to specify than $\boldsymbol{\phi}$, even though it is sometimes unintuitive. For example, a linear model $f_l$ can be specified by choosing the linear kernel

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^T \boldsymbol{\Sigma} \boldsymbol{x}_j. \qquad (16)$$

The non-linear model used in this paper is a squared exponential function (SqE) kernel function

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \, \exp\left[ -\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \,\boldsymbol{\Sigma}\, (\boldsymbol{x}_i - \boldsymbol{x}_j)\right], \qquad (17)$$

where $\sigma_f^2 > 0$ is the signal variance. This kernel is also referred to as the radial basis function (RBF) kernel in the context of Artificial Neural Networks.

It is possible that emulator $f_{nl}$ is not able to model the output $y_n$ perfectly and, in that case, we can assume that there is noise in the estimation, i.e., $y_n = f_{nl}(\boldsymbol{x}_n) + \varepsilon_n$. Following the standard practice in GP regression, we assume that $\varepsilon_n$ are independent Gaussian random variables with zero mean and noise variance $\sigma_n^2$. Specifying this non-linear model requires estimation of the noise variance $\sigma_n^2$, the signal variance $\sigma_f^2$, and $\boldsymbol{\Sigma}$. Collectively, these quantities are referred to as "hyper-parameters" of the GP model, and we denote the set of hyper-parameters by $\boldsymbol{\psi}$ [33].

### A.4.1 Fitting and Using a GP Model

Building a GP-based emulator requires two tasks. The first task is to estimate the hyper-parameters. This is called 'learning'. The second task is to compute $\hat{f}_{nl}(\boldsymbol{x}_*)$ given a new input $\boldsymbol{x}_*$ and the estimated hyper-parameters. This is called 'prediction'.

We first give details of the prediction task. We wish to compute the *predictive* distribution of the output (here: annual energy use, denoted by $y_*$) at a new input (here: the set of features that define a building and weather conditions for that particular year, denoted by $\boldsymbol{x}_*$) present in the test data, i.e., the distribution $p(y_*|\boldsymbol{x}_*, \mathcal{D}, \boldsymbol{\psi})$ where $\mathcal{D} = \{y_1, \boldsymbol{x}_1, y_2, \boldsymbol{x}_2, \ldots, y_{N_{\text{train}}}, \boldsymbol{x}_{N_{\text{train}}}\}$ and $\boldsymbol{\psi}$ is the set of hyper-parameters. For GP regression, this distribution is a Gaussian and has a closed form expression. This follows from the property that any finite number of samples drawn from a Gaussian Process are jointly Gaussian, giving the following expression for the distribution of $\boldsymbol{y} := [y_1, y_2, \ldots, y_{N_{\text{train}}}]^T$ and the $y_*$ corresponding to $\boldsymbol{x}_*$:

$$\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K} + \sigma_n^2 \boldsymbol{I} & \boldsymbol{k}_* \\ \boldsymbol{k}_*^T & k_{**} + \sigma_n^2 \end{bmatrix}\right) \qquad (18)$$

where $\boldsymbol{K}$ is a matrix whose $(i,j)$'th entry is $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\boldsymbol{k}_*$ is a vector whose $i$'th entry is

$k(\boldsymbol{x}_i, \boldsymbol{x}_*)$, $k_{**} = k(\boldsymbol{x}_*, \boldsymbol{x}_*)$, and $\boldsymbol{I}$ is an identity matrix of size $N_{\text{train}} \times N_{\text{train}}$. Using the above equation, we can write the expression for the distribution of $y_*$ given $\boldsymbol{y}$ by using the conditional distribution for a Gaussian distribution [33, pg. 16]:

$$p(y_*|\boldsymbol{x}_*, \mathcal{D}, \boldsymbol{\psi}) := \mathcal{N}(\mu_*, \sigma_*^2), \qquad (19)$$
$$\text{where } \mu_* := \boldsymbol{k}_*^T(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y}$$
$$\text{and } \sigma_*^2 := k_{**} - \boldsymbol{k}_*^T(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{k}_*.$$

The computational complexity of these operations is $\mathcal{O}(N_{\text{train}}^3)$, due to the inversion of the matrix $\boldsymbol{K} + \sigma_n^2\boldsymbol{I}$. That is, the number of operations required increases by the cube of the number of elements, so adding 2 data points, for example, would require 8 additional operations.

The distribution $p(y_*|\boldsymbol{x}_*, \mathcal{D}_t, \boldsymbol{\theta})$ depends on the specification of $\boldsymbol{\psi}$. We estimate $\boldsymbol{\psi}$ by maximizing the log-likelihood: $\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\psi})$. This is called the maximum likelihood estimation (MLE) method. The closed-form expression for the log-likelihood is

$$\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) := -\tfrac{1}{2}\log|\boldsymbol{K} + \sigma_n^2\boldsymbol{I}| - \tfrac{1}{2}\boldsymbol{y}^T(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y} - \frac{N_{\text{train}}}{2}\log(2\pi). \qquad (20)$$

This can be optimised with a numerical optimization method [33]. However, every iteration requires a matrix inversion, which could be costly if the optimization takes too many iterations. We discuss a method to reduce the computation cost in Appendix A.4.2.

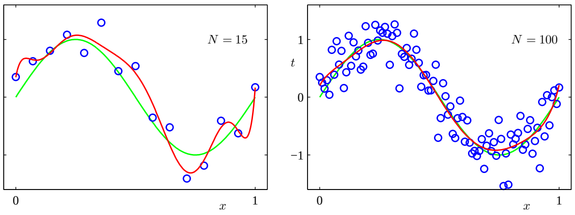
GP regression allows the specification of a 'signal noise', i.e., the variance of the uncertainty in the data itself ($\sigma_n$ in Equation (18)). This noise variance may be fixed to some appropriate value or tuned along with the other hyper-parameters. Given that we are using simulated data, we expect the noise variance to be very low. However, the models are less stable when the signal noise is low because the covariance matrices are frequently non-invertible (Equation (20)). This is because the signal noise acts as a regulariser in this inversion of the covariance matrix $\boldsymbol{K}$ when fitting a GP model. Therefore, a value of nearly zero for the signal noise foregoes the stability accorded by the regulariser. When we set a lower bound, $\sigma_n \geq 10^{-6}$, the ill-conditioning of the covariance matrices is reduced.

### A.4.2 Using Big Datasets

Generally speaking, the more data a model sees to characterize a domain, the better it is able to predict on unseen data from that domain. Fitting a GP model involves the inversion of a matrix (the covariance matrix), whose size is $N \times N$, where $N$ is the number of data points. This puts a limit on the size of dataset that can be considered for learning hyper-parameters or predicting. It is possible to work around this limitation by using the so-called 'sparse' methods, i.e., methods using sparse representations of the covariance matrix. However, these methods invariably reduce the predictive performance of the model. In this paper, we present a simple method to extend the amount of data considered, similar to a proposal in Chalupka et al. [42].

The training procedure consists of two steps: learning hyper-parameters through Maximum Likelihood Estimation (MLE) using some training data, and then predicting the output at test inputs using the same training data. These two steps correspond to Equations (19) and (20), respectively. The matrix $\boldsymbol{K}$ from Equation (19) depends on training data, while the vector $\boldsymbol{k}_*$ depends on the testing data. In our experiments, the estimates of hyper-parameters stabilize with about 1000-2000 training data points. If we continue with the policy of learning and predicting using the same dataset, we are restricted to models trained on about 5,000 points, because repeatedly inverting matrices (as part of the MLE step) of size 5,000 or more is impractical on the hardware available to us. A simple method to add more data was to increase the size of the dataset during prediction.

We modified the procedure to reduce run time by using sets of different sizes for learning and prediction (Figure 1). If learning is carried out on a smaller set of 2000 points ($N_{\text{train}} = 2000$), i.e., the matrix $\boldsymbol{K}$ in Equation (20) is defined on a dataset of 2000 points, estimates of the hyper-parameters $\boldsymbol{\psi}$ are fixed relatively rapidly. Given this estimate of $\boldsymbol{\psi}$, we proceed to increase the data size to $N_{\text{pred}} > N_{\text{train}}$ for prediction using Equation (19). Since prediction involves only one matrix inversion, we were able to handle $N_{\text{pred}}$ of size up to 12,000. This modification to the procedure gives a modest improvement in the validation error.




**(a)** *Dataset of size $N = 15$* **(b)** *Dataset of size $N = 100$*

**Figure 11:** *Using $N = 15$ [left] or $N = 100$ [right] data points to fit a polynomial of degree $M = 9$, shows that "...increasing the size of the data set reduces... over-fitting" [6].*

# Acknowledgements

The authors would like to thank Dr Minu Agarwal, Dr Giorgia Chinazzo, Dr Georgios Mavromatidis, Margaux Peltier, Dr Masashi Sugiyama, Dr Norihiro Maeda, the RIKEN-AIP team, and the Energy Systems Research Unit (ESRU) team at University of Strathclyde. For part of this work, Parag was hosted at ESRU and RIKEN-AIP, and his work was funded by the Swiss National Science Foundation's Postdoc.Mobility grant P2ELP2_168519.

# References

[1] Joshua New, Jibonananda Sanyal, Bob Slattery, Anthony Gehl, William Miller, and Aaron Garrett. Big Data Mining for Assessing Calibration of Building Energy Models. *International Journal of Computer & Software Engineering*, 3(2), September 2018. ISSN 24564451. doi: 10.15344/2456-4451/2018/136. URL `https://www.graphyonline.com/archives/IJCSE/2018/IJCSE-136/`.

[2] J. A. Clarke. A vision for building performance simulation: a position paper prepared on behalf of the IBPSA Board. *Journal of Building Performance Simulation*, 8(2):39–43, March 2015. ISSN 1940-1493, 1940-1507. doi: 10.1080/19401493.2015.1007699. URL `http://www.tandfonline.com/doi/full/10.1080/19401493.2015.1007699`.

[3] Mahnameh Taheri, Parag Rastogi, Colin Parry, and Alan Wegienka. Benchmarking Building Energy Consumption Using Efficiency Factors. In *Proceedings of BS 2019*, pages 3863–3870, Rome, Italy, 2019. doi: 10.26868/25222708.2019.210575. URL `http://www.ibpsa.org/proceedings/BS2019/BS2019_210575.pdf`.

[4] Parag Rastogi. *On the sensitivity of buildings to climate: the interaction of weather and building envelopes in determining future building energy consumption.* PhD, Ecole polytechnique fédérale de Lausanne, Lausanne, Switzerland, August 2016. URL `https://infoscience.epfl.ch/record/220971?ln=en`. doi:10.5075/epfl-thesis-6881.

[5] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition.* Springer, August 2009. ISBN 978-0-387-84858-7.

[6] Christopher M Bishop. *Pattern recognition and machine learning.* springer, 2006.

[7] Parag Rastogi, Mohammad Emtiyaz Khan, and Marilyne Andersen. Gaussian-Process-Based Emulators for Building Performance Simulation. In *Proceedings of BS 2017*, San Francisco, CA, USA, August 2017. IBPSA.

[8] Hai-xiang Zhao and Frédéric Magoulès. A review on the prediction of building energy consumption. *Renewable and Sustainable Energy Reviews*, 16(6):3586–3592, August 2012. ISSN 1364-0321. doi: 10.1016/j.rser.2012.02.049. URL `http://www.sciencedirect.com/science/article/pii/S1364032112001438`.

[9] Saleh Seyedzadeh, Farzad Pour Rahimian, Stephen Oliver, Sergio Rodriguez, and Ivan Glesk. Machine learning modelling for predicting non-domestic buildings energy performance: A model to support deep energy retrofit decision-making. *Applied Energy*, 279:115908, December 2020. ISSN 0306-2619. doi: 10.1016/j.apenergy.2020.115908. URL `http://www.sciencedirect.com/science/article/pii/S0306261920313702`.

[10] Aurélie Foucquier, Sylvain Robert, Frédéric Suard, Louis Stéphan, and Arnaud Jay. State of the art in building modelling and energy performances prediction: A review. *Renewable and Sustainable Energy Reviews*, 23:272–288, July 2013. doi: 10.1016/j.rser.2013.03.004. URL `http://linkinghub.elsevier.com/retrieve/pii/S1364032113001536`.

[11] Emilie Nault. *Solar Potential in Early Neighborhood Design. A Decision-Support Workflow Based*

*on Predictive Models*. PhD Thesis, Ecole polytechnique fédérale de Lausanne, Lausanne, Switzerland, 2016.

[12] Wei Tian. A review of sensitivity analysis methods in building energy analysis. *Renewable and Sustainable Energy Reviews*, 20:411–419, April 2013. doi: 10.1016/j.rser.2012.12.014. URL `http://linkinghub.elsevier.com/retrieve/pii/S1364032112007101`.

[13] Christina Johanna Hopfe. *Uncertainty and sensitivity analysis in building performance simulation for decision support and design optimization*. PhD, Technische Universiteit Eindhoven, Eindhoven, The Netherlands, 2009. URL `http://www.bwk.tue.nl/bps/hensen/team/past/Hopfe.pdf`.

[14] Sten de Wit. *Uncertainty in predictions of thermal comfort in buildings*. PhD, Delft University of Technology, Delft, The Netherlands, June 2001. URL `http://resolver.tudelft.nl/uuid:a231bca8-ec81-4e22-8b34-4bafc062950e`.

[15] Geoffrey K. F. Tso and Kelvin K. W. Yau. Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks. *Energy*, 32(9):1761–1768, September 2007. ISSN 0360-5442. doi: 10.1016/j.energy.2006.11.010. URL `http://www.sciencedirect.com/science/article/pii/S0360544206003288`.

[16] Endong Wang, Zhigang Shen, and Kevin Grosskopf. Benchmarking energy performance of building envelopes through a selective residual-clustering approach using high dimensional dataset. *Energy and Buildings*, 75(Supplement C):10–22, June 2014. ISSN 0378-7788. doi: 10.1016/j.enbuild.2013.12.055. URL `http://www.sciencedirect.com/science/article/pii/S0378778814000048`.

[17] Emilie Nault, Giuseppe Peronato, Emmanuel Rey, and Marilyne Andersen. Review and critical analysis of early-design phase evaluation metrics for the solar potential of neighborhood designs. *Building and Environment*, 92:679–691, October 2015. ISSN 0360-1323. doi: 10.1016/j.buildenv.2015.05.012. URL `http://www.sciencedirect.com/science/article/pii/S0360132315002243`.

[18] Emilie Nault, Parag Rastogi, Emmanuel Rey, and Marilyne Andersen. The sensitivity of predicted energy use to urban geometrical factors in various climates. In *Proceedings of PLEA 2015*, Bologna, Italy, September 2015. URL `http://infoscience.epfl.ch/record/211101?ln=en`.

[19] Janelle S Hygh, Joseph F. DeCarolis, David B Hill, and S Ranji Ranjithan. Multivariate regression as an energy assessment tool in early building design. *Building and Environment*, 57:165–175, November 2012. doi: 10.1016/j.buildenv.2012.04.021. URL `http://dx.doi.org/10.1016/j.buildenv.2012.04.021`.

[20] Somayeh Asadi, Shideh Shams Amiri, and Mohammad Mottahedi. On the development of multilinear regression analysis to assess energy consumption in the early stages of building design. *Energy and Buildings*, 85:246–255, December 2014. ISSN 0378-7788. doi: 10.1016/j.enbuild.2014.07.096. URL `http://www.sciencedirect.com/science/article/pii/S0378778814007154`.

[21] Shideh Shams Amiri, Mohammad Mottahedi, and Somayeh Asadi. Using multiple regression analysis to develop energy consumption indicators for commercial buildings in the U.S. *Energy and Buildings*, 109:209–216, December 2015. ISSN 0378-7788. doi: 10.1016/j.enbuild.2015.09.073. URL `http://www.sciencedirect.com/science/article/pii/S0378778815303133`.

[22] Torben Østergård, Rasmus L. Jensen, and Steffen E. Maagaard. Early Building Design: Informed decision-making by exploring multidimensional design space using sensitivity analysis. *Energy and Buildings*, 142(Supplement C):8–22, May 2017. ISSN 0378-7788. doi: 10.1016/j.enbuild.2017.02.059. URL `http://www.sciencedirect.com/science/article/pii/S0378778817306916`.

[23] Xi Chen, Hongxing Yang, and Ke Sun. Developing a meta-model for sensitivity analyses and prediction of building performance for passively designed high-rise residential buildings. *Applied Energy*, 194(Supplement C):422–439, May 2017. ISSN 0306-2619. doi: 10.1016/j.apenergy.2016.08.180. URL `http://www.sciencedirect.com/science/article/pii/S0306261916312892`.

[24] Joshua Hester, Jeremy Gregory, and Randolph Kirchain. Sequential early-design guidance for residential single-family buildings using a probabilistic metamodel of energy consumption. *Energy and Buildings*, 134(Supplement C):202–211, January 2017. ISSN 0378-7788. doi: 10.1016/j.enbuild.2016.10.047. URL `http://www.sciencedirect.com/science/article/pii/S037877881631369X`.

[25] Siyu Wu and Jian-Qiao Sun. Multi-stage regression linear parametric models of room temperature in office buildings. *Building and Environment*, 56:69–77, October 2012. doi: 10.1016/j.buildenv.2012.02.026. URL `http://linkinghub.elsevier.com/retrieve/pii/S0360132312000716`.

[26] Sandhya Patidar, David P Jenkins, Gavin J Gibson, and P F G Banfill. Statistical techniques to emulate dynamic building simulations for overheating analyses in future probabilistic climates. *Journal of Building Performance Simulation*, 4 (3):271–284, 2011. doi: 10.1080/19401493.2010.531144.

[27] D.P. Jenkins, M. Gul, and S. Patidar. Probabilistic future cooling loads for mechanically cooled offices. *Energy and Buildings*, 66:57–65, November 2013. ISSN 0378-7788. doi: 10.1016/j.enbuild.2013.07.040. URL `http://www.sciencedirect.com/science/article/pii/S0378778813004313`.

[28] David P Jenkins, Sandhya Patidar, P F G Banfill, and Gavin J Gibson. Probabilistic climate projections with dynamic building simulation: Predicting overheating in dwellings. *Energy and Buildings*, 43(7):1723–1731, July 2011. doi: 10.1016/j.enbuild.2011.03.016. URL `http://linkinghub.elsevier.com/retrieve/pii/S0378778811000946`.

[29] David Coley and Tristan Kershaw. Changes in internal temperatures within the built environment as a response to a changing climate. *Building and Environment*, 45(1):89–93, January 2010. ISSN 0360-1323. doi: 10.1016/j.buildenv.2009.05.009. URL `http://www.sciencedirect.com/science/article/pii/S0360132309001280`.

[30] Yeonsook Heo and Victor M. Zavala. Gaussian process modeling for measurement and verification of building energy savings. *Energy and Buildings*, 53:7–18, October 2012. ISSN 0378-7788. doi: 10.1016/j.enbuild.2012.06.024. URL `http://www.sciencedirect.com/science/article/pii/S037877881200312X`.

[31] Michael C. Burkhart, Yeonsook Heo, and Victor M. Zavala. Measurement and verification of building systems under uncertain data: A Gaussian process modeling approach. *Energy and Buildings*, 75:189–198, June 2014. ISSN 0378-7788. doi: 10.1016/j.enbuild.2014.01.048. URL `http://www.sciencedirect.com/science/article/pii/S0378778814001091`.

[32] Filippo Monari. *Sensitivity Analysis and Bayesian Calibration of Building Energy Models*. PhD, University of Strathclyde, Glasgow, UK, 2016.

[33] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass, 2006. ISBN 978-0-262-18253-9.

[34] Michael Deru, Kristin Field, Daniel Studer, Kyle Benne, Brent Griffith, Paul Torcellini, Bing Liu, Mark Halverson, Dave Winiarski, Michael Rosenberg, Mehry Yazdanian, Y. Joe Huang, and Drury B. Crawley. U.S. Department of Energy commercial reference building models of the national building stock. Technical report, National Renewable Energy Laboratory (NREL), February 2011. URL `http://digitalscholarship.unlv.edu/renew_pubs/44`.

[35] Minu Agarwal, Parag Rastogi, Margaux Peltier, Luisa Pastore, and Marilyne Andersen. Examining Building Design Decisions Under Long Term Weather Variability and Microclimate Effects: A Case Based Exploratory Study. In *Proceedings of PLEA 2016*, Los Angeles, USA, July 2016.

[36] Giorgia Chinazzo. Refurbishment of Existing Envelopes in Residential Buildings: assessing robust solutions for future climate change. Master's thesis, EPFL, Lausanne, Switzerland, 2014. URL `http://infoscience.epfl.ch/record/203438`.

[37] Parag Rastogi, Sönke Frederik Horn, and Marilyne Andersen. Toward Assessing the Sensitivity of Buildings to Changes in Climate. In *Proceedings of PLEA 2013*, Munich, Germany, September 2013. URL `http://infoscience.epfl.ch/record/187507?ln=en`.

[38] Jan Remund, Stefan Mueller, Stefan Kunz, and Christoph Schilter. METEONORM Handbook Part II : Theory. Technical report, Meteotest, May 2012. URL `http://meteonorm.com/download/software/mn70/`.

[39] S Wilcox and W Marion. Users' Manual for TMY3 Data Sets. Technical report, National Renewable Energy Laboratory, May 2008. URL `http://www.nrel.gov/docs/fy08osti/43156.pdf`.

[40] Parag Rastogi and Marilyne Andersen. Embedding Stochasticity in Building Simulation Through Synthetic Weather Files. In *Proceedings of BS*

*2015*, Hyderabad, India, December 2015. URL http://infoscience.epfl.ch/record/208743.

[41] Parag Rastogi and Marilyne Andersen. Incorporating Climate Change Predictions in the Analysis of Weather-Based Uncertainty. In *Proceedings of SimBuild 2016*, Salt Lake City, UT, USA, August 2016. URL http://infoscience.epfl.ch/record/208743.

[42] Krzysztof Chalupka, Christopher KI Williams, and Iain Murray. A framework for evaluating approximation methods for Gaussian process regression. *Journal of Machine Learning Research*, 14 (Feb):333–350, 2013. URL http://www.jmlr.org/papers/v14/chalupka13a.html. 00071.

[43] CIBSE. *Sustainability - CIBSE Guide L: 2020*. CIBSE guides. The Chartered Institution of Building Service Engineers, London, UK, 2 edition, July 2020. ISBN 978-1-912034-49-9.

[44] J. A. Clarke. The role of building operational emulation in realizing a resilient built environment. *Architectural Science Review*, 61 (5):358–361, September 2018. ISSN 0003-8628. doi: 10.1080/00038628.2018.1502157. URL https://doi.org/10.1080/00038628.2018.1502157. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00038628.2018.1502157.

[45] J. A. Clarke. A simulation-based procedure for the holistic resilience testing of building performance. *IOP Conference Series: Earth and Environmental Science*, 329:012027, October 2019. ISSN 1755-1315. doi: 10.1088/1755-1315/329/1/012027. URL https://doi.org/10.1088%2F1755-1315%2F329%2F1%2F012027. Publisher: IOP Publishing.

[46] Léon Bottou. Large-Scale Machine Learning with Stochastic Gradient Descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010)*, pages 177–187, Paris, France, August 2010. Springer. URL http://leon.bottou.org/papers/bottou-2010.

[47] Clayton Miller, Pandarasamy Arjunan, Anjukan Kathirgamanathan, Chun Fu, Jonathan Roth, June Young Park, Chris Balbach, Krishnan Gowri, Zoltan Nagy, Anthony Fontanini, and Jeff Haberl. The ASHRAE Great Energy Predictor III competition: Overview and results. *arXiv:2007.06933 [cs]*, July 2020. URL http://arxiv.org/abs/2007.06933. arXiv: 2007.06933.