Winter 3-23-2017

Building Energy Model Calibration for Retrofit Decision Making

Nicolas R. Johnson
Portland State University

Building Energy Model Calibration for Retrofit Decision Making

by

Nicolas R. Johnson

A thesis submitted in partial fulfillment of the
requirements for the degree of

Master of Science
in
Mechanical Engineering

Thesis Committee:
Sung Yi, Chair
Elliott Gall
Chien Wern

Portland State University
2017

Abstract

Accommodating the continued increase in energy demand in the face of global climate change has been a worldwide concern. With buildings in the US consuming nearly 40% of national energy, a concerted effort must be given to reduce building energy consumption. As new buildings continue to improve their efficiency through more restrictive energy codes, the other 76.9 billion square feet of current building stock falls further behind. The rate at which current buildings are being retrofit is not enough and better tools are needed to access the benefits of retrofits and the uncertainties associated with them. This study proposes a stochastic method of building energy model calibration coupled with a monthly normative building simulation addressed in ISO 13890. This approach takes advantage of the great efficiency of Latin Hypercube Sampling (LHS) and the lightweight normative building simulation method, to deliver a set of calibrated solutions to asses the effectiveness of energy conservation measure, making uncertainty a part of the modeling process. A case study on a mixed-use university building is conducted to show the strength and performance of this simple method. Limitations and future concerns are also addressed.

Acknowledgments

Thank you to Molly Meyer, Melissa Scholl, Andrew Wollman, Dr. Huafen Hu, and PSU MME Faculty, Staff, and Students.

This study was funded by Portland General Electric.

Contents

Abstract **i**

Acknowledgments **ii**

List of Tables **vi**

List of Figures **vii**

Nomenclature **viii**

Acronyms **xi**

1 Introduction **1**

1.1 Motivation 1

1.2 Building Energy Modeling 3

1.3 Calibration 5

2 Overview **7**

2.1 Calibration 7

2.2 Reddy 9

2.3 Stochastic LHS Method 11

3 Methods **14**

3.1 Current Calibration Practice 14

3.2 Building Simulation Software 16

3.3 Sensitivity Analysis 20

3.4 Latin Hypercube Sampling 21

3.5 Sample Size 22

3.6 Calibration Criteria 25

3.7 Process 27

4 Case Study **28**

4.1 Nueberger Hall 28

4.2 Sensitivity Analysis 32

4.3 Settings 33

4.4 Results 34

5 Analysis **37**

5.1 Large Sample Size 37

5.2 Refinement 40

6 Conclusion **41**

6.1 Future Work 41

6.2 Considerations 43

6.3 Final Thoughts 43

Bibliography **45**

Appendix A Building Inputs **49**

Appendix B ISO 13790 MATLAB Code **53**

Appendix C Internal Gains and Solar Calculations **57**

Appendix D Morris Method MATLAB Functions **69**

Appendix E Latin Hypercube Sampling MATLAB Functions **77**

Appendix F Calibration MATLAB Scripts **82**

List of Tables

3.1 Table from Reichmuth and Turner, listing the key parameters analyzed with the First View inverse modeling tool. 15

4.1 Utility data provided by Portland State University (PSU) F&P department for the years 2003-2006. 30

4.2 List of input parameters 31

4.3 Parameter list, post sensitivity analysis 33

4.4 Calibration settings for price weighting (A), even weighting (B), and strict criteria (C). 34

4.5 goodness of fit (GOF) distribution for settings A-D for each energy source. Calibrated for 2005-2006 and validated with the 2003-2004 and 2004-2005 years. 35

5.1 Large sample size results for Strict criteria enforcement (strict) criteria with $*$ indicate the calibration year and the other years are for validation. 39

List of Figures

1.1 Figure from Olgayay and Seruto, showing the trend of carbon emissions for current commercial buildings, a buisness as usual approach to retrofits, the Architecture 2030 goals, and the US Copenhagen target. 2

1.2 Figures shown from Turner et. al 2008. 4

2.1 Figure from Reddy et al. displays the triangular variable distribution used to discretize each parameter for the mid-point Latin Hypercube Monte Carlo (LHMC) method. 10

2.2 Flowchart of Calibration Process 13

3.1 Figure from Reichmuth and Turner, displaying the results for a medium-sized Chicago office building from the First View inverse modeling tool. 15

3.2 Illustration of Latin Hypercube Sampling. 22

4.1 Photo of Neuberger Hall 29

5.1 Boxplots displaying the interquartile range of Heating and Electricity predictions for strict weight solutions for the 2005-2006 year, $n = 5x10^5$. 38

Nomenclature

a_C Calculated building time constant.

$a_{C,0}$ Dimensionless Parameter for Cooling.

a_H Calculated building time constant.

$a_{H,0}$ Dimensionless Parameter for Heating.

C_m Internal Heat Capacity.

CoV_{MC} Coefficient of Variation of a Monte Carlo Simulation.

$\eta_{C,ls}$ Heat Loss Utilization Factor.

$\eta_{H,gn}$ Heat Gain Utilization Factor.

γ_C Heat Loss Ratio.

γ_H Heat Gain Ratio.

H_{tr} Heat Transfer Coefficient due to Transmission.

H_{ve} Heat Transfer Coefficient due to Ventilation.

μ True mean of a sample.

μ^* Absolute mean of elementary effects.

n Number of Samples.

Q_C Total Cooling Energy Demand.

$Q_{C,gn}$ Heat Gains while in Cooling Mode.

$Q_{C,ht}$ Total Heat Transfer While Cooling.

Q_H Total Heating Energy Demand.

$Q_{H,gn}$ Heat Gains while in Heating Mode.

$Q_{H,ht}$ Total Heat Transfer While Heating.

σ Standard Deviation.

σ_x Absolute mean of elementary effects.

$\sigma_{\bar{X}}$ Absolute mean of elementary effects.

τ Calculated building time constant..

$\tau_{C,0}$ Initial building time constant.

$\tau_{H,0}$ Initial building time constant.

θ Confidence Range.

w_{kWh} Weighting Factor for Electricity.

$w_{CV_{RMSE}}$ Weighting Factor for Coefficient of Variance of the $RMSE$.

w_{NMBE} Weighting Factor for Mean Bias Error.

w_{Therm} Weighting Factor for Gas Heating.

$\bar{X}$ Sample Mean.

$Z_{\frac{1-\delta}{2}}$ Z Statistic.

Acronyms

CV_{RMSE} coefficient of variance of root mean square error.

AHU air handling unit.

ASHRAE American Society of Heating, Refrigeration, and Air-conditioning Engineers.

CAV constant vir volume.

CDF cumulative distribution function.

CoV Coefficient of Variation.

ECM energy conservation measure.

EE elementary effect.

EEM energy efficiency measure.

EERE Energy Efficiency and Renewable Energy.

ESCO energy service company.

EUI energy use intensity.

GIGO garbage in garbage out.

GOF goodness of fit.

HVAC Heating Ventilation and Air Conditioning.

LEED Leadership in Energy & Environmental Design.

LHMC Latin Hypercube Monte Carlo.

LHS Latin Hypercube Sampling.

M&V measurement and verification.

MC Monte Carlo.

NBI New Buildings Institute.

NCEP National Center for Environmental Prediction.

NH Neuburger Hall.

NMBE normalized mean bias error.

PDF probability density function.

PSU Portland State University.

strict Strict criteria enforcement.

Chapter 1

Introduction

1.1 Motivation

Worldwide energy demand is at an all-time high and continues to grow. From 1984 to 2004 primary energy has grown by 49% and CO2 emissions have grown by 43% [1], with the increase in energy consumption growing faster than population growth. In 2010 the US DOE reports that buildings account for 70% of electricity and 50% of natural gas use, with the 76.9 billion square feet of current building stock [2, 3]. And in 2004, US buildings alone accounted for more emissions than any other countries emissions other than China [4]. Because more than half of the energy in the US comes from coal and buildings are consuming 41% of total primary energy [5] building standards are becoming stricter, resulting in more efficient new buildings. What is left behind is the 76.9 billion square feet of current building stock, nearly all of which can use some improvement.

Due to these concerns the building sector is the focus of many policy changes, building performance labeling, and challenges to reduce energy and emissions of the building stock for the future. The white house would like to see a 20% energy reduction in commercial buildings by the year 2020 [6]. The Leadership in En-

ergy & Environmental Design (LEED) program requires each building must meet an energy efficiency requirement, among other conservation and carbon–reducing requirements, before being granted a LEED rating for a new building or retrofit project [7]. The 2030 Challenge is a project which suggests reducing carbon emissions of new buildings and retrofits by 60% and increases this standard every 5 years such that by 2030 all new buildings and retrofits are carbon neutral [6]. Continued improvement within these goals can only be accomplished through a concerted effort to create new, more efficient buildings, and implementation of energy conservation measure (ECM) on the existing building stock [3, 4, 8, 9].

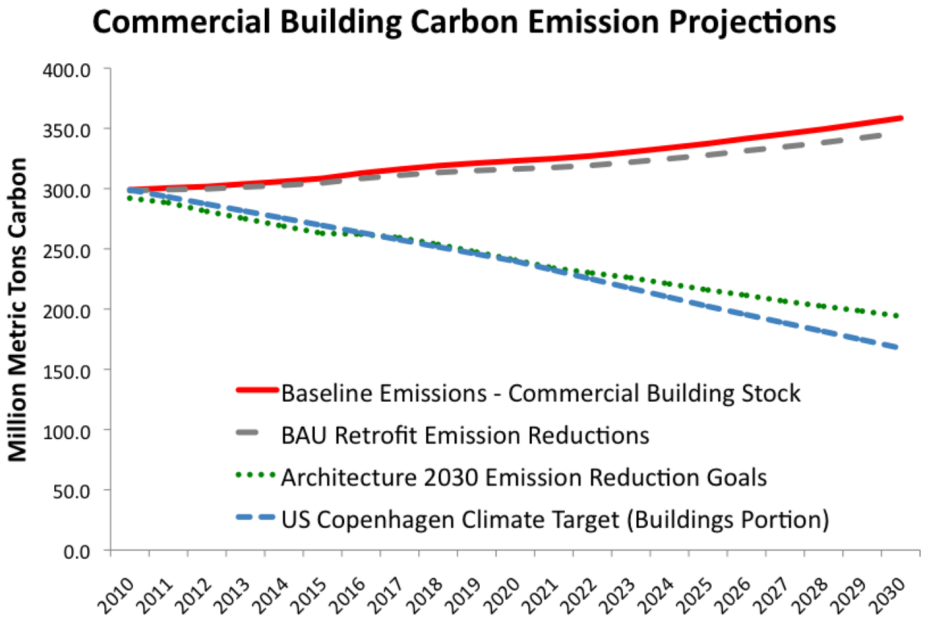

Figure 1.1: Figure from Olgayay and Seruto, showing the trend of carbon emissions for current commercial buildings, a buisness as usual approach to retrofits, the Architecture 2030 goals, and the US Copenhagen target.

Buildings are currently being retrofit at a rate of 2.2% per year with a median energy savings of 11% for those retrofits [3]. At this rate half of the building stock

will be retrofit by the year 2054. A large part of reaching goals to reduce emissions and conserve energy is to either retrofit more square-footage per year or to make each building retrofit more efficient.

1.2 Building Energy Modeling

Predictive building energy modeling have been in development since the 70s as part of the effort to reduce building energy usage. These simulations model the physics of all the interactions which take place within a buildnig such as heat gains from people, lights, electronics, and sunlight; the heat transfer through the building envelope, the outside air coming into the building, weather patterns, and building set points and schedules. The accuracy of these models in quantitatively predicting actual buildings performance has been in question due to wild fluctuations between model predictions ad measured energy consumption. A recent study by the New Buildings Institute (NBI) shows this disparity in a study of LEED certified buildings, comparing the measured vs. predicted energy use intensity (EUI) of a host of buildings, with many LEED buildings performing worse than the energy standards at the time, shown in Figure 3.1 [10].

Disparities are not uncommon when modeling new buildings as behavior patterns must be assumed, systems installed are assumed to be operating as designed, and in some instances sytems are materials modeled do not make it into the final product [11]. Despite the inconsistencies from a comparative view, building energy models still provide valuable insight on which materials or components have the largest effect on energy consumption and proper building management.

The growing need in buildings retrofits provide an interesting opportunity for

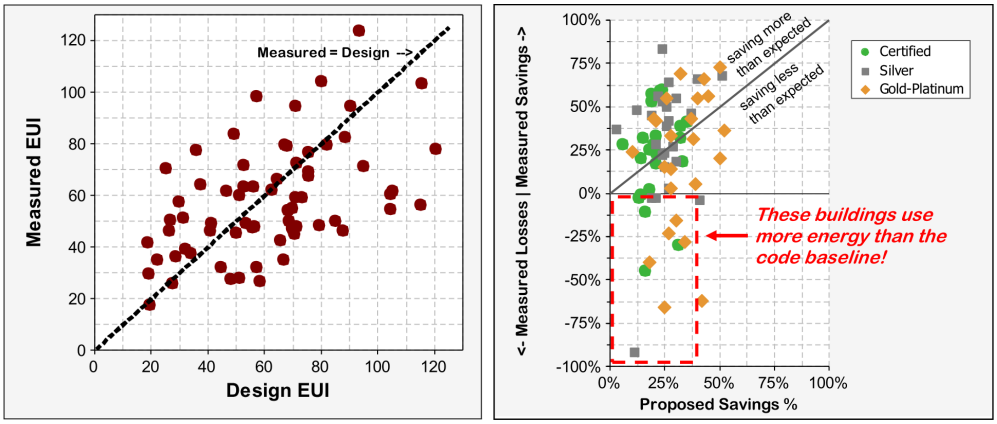

Figure 1.2: Figures shown from Turner et. al 2008.

energy modeling. Compared to its application to new building design, modelers dealing with existing buildings have access to operation schedules, building materials, and equipment. Many times a base model of the building will be developed, an ECM is implemented in the model, and the energy difference is estimated as the expected savings. A similar problem still arises that the energy savings can still vary depending on weather patterns, assumptions made in modeling (where required), and how the ECM is implemented. Managing the uncertainty related to the retrofit process is becoming more important and LEED has initiated a measurement and verification (M&V) incentive for new buildings and retrofits. Other organizations that have published M&V guidelines include the American Society of Heating, Refrigeration, and Air-conditioning Engineers (ASHRAE), the , and Energy Efficiency and Renewable Energy (EERE). Applying the principles of M&V to building energy modeling is known as building energy model calibration, comparing the model with utility data and tuning it to get an acceptable fit.

1.3 Calibration

Calibration methods for building energy models have been in development since the mid-1980s and have been utilized for retrofit decision making since then [12]. These methods can be classified by two general categories: manual calibration and robust computer based calibration. Due to the large number of building inputs in a building energy model, building model calibration is essentially an undetermined problem[1]. Therefore the quality of a manual calibration is highly dependent on the experience and expertise of the analysts performing the calibration. With this method one can get any calibration result desired but the solution that results may not be the true solution. Robust computer calibration uses mathematical algorithms and computer analysis to locate the best fit for an existing building model. Regardless of the specific method used, all calibration employs a certain level of optimization. The nonlinearity of a building model and its nature of being an undetermined problem makes robust computer calibration computation intensive and complicated. A detailed review of these calibration methods has been conducted by Reddy [13]. Given the undetermined nature of building model calibration, Reddy et al. have proposed to shift the calibration goal from finding a best fit to locating the best set of plausible fits in their recently completed ASHRAE project RP-1051 [13] [14] [15]. A mid-point Latin Hypercube Monte Carlo (LHMC) method is developed and demonstrated through case studies. This mid-point LHMC method provides a new perspective to building model calibration. Another recently completed thesis, comparing manual calibration and the mid-point LHMC method, concludes that the mid-point LHMC is most appropriate for models with a high number of uncertain

[1] i.e. the number of unknowns are larger than the given knowns

building input parameters and when only monthly utility data is available [16].

In light of the mid-point LHMC method, this study presents a simplified and yet effective approach to support decision making in existing building retrofits based on limited data collection through a walk-through site audit. The method couples a normative building modeling method with a general Latin Hypercube Sampling (LHS) method. The monthly normative building energy model, based on ISO13790 [17], is to solve a set of quasi-steady state formulas of heat balance equations to estimate monthly building energy consumption. The normative energy modeling method requires much less building input compared to detailed dynamic building simulations and is developed to provide the right balance between accuracy and data collection costs. Its minimal requirement in computation allows LHS to scan the entire feasible space with a large number of sample runs. Therefore the proposed method is able to deliver a sound calibrated model for alternative retrofit evaluation. A case study with an institutional building, built in the 1960s, is presented to demonstrate the proposed approach.

Chapter 2

Overview

Building energy models are primarily used as a means to estimate the total heating, cooling, and electricity load in a building. This is done by inputting the physical properties of the building, the internal loads, operation schedules, Heating Ventilation and Air Conditioning (HVAC) system, and weather data. Traditionally energy service companies (ESCOs) have used building energy models to evaluate the total energy use of new buildings or the effectiveness of energy efficiency measures (EEMs) in older buildings. Due to wide variation between simulated results and actual energy use, M&V practices are becoming more popular. These practices incorporate calibration procedures into the modeling process ensuring a higher quality of modeled results.

2.1 Calibration

Calibration refers to the process of adjusting a set of inputs within a model to reach a desired output. The calibration of a building energy model compares energy use between observed and modeled behavior over a given time period. With the ideal product of a calibration having the correct set of inputs, so that the output from the model is representative of the observed energy use for the given time frame.

Benefits of calibrated simulations include providing information to building owners and utilities about energy usage patterns, better prediction of EEMs post-retrofit energy usage, and increased system knowledge for building owners [13].

However, within the calibration process, it is important not to simply have a calibrated model through garbage in garbage out (GIGO)[1]; but to have a model meet the desired result by design. This situation can best be summarized by Hornberger and Spear [18]

> "... most simulation models will be complex, with many parameters, state-variables and non linear relations. Under the best circumstances, such models have many degrees of freedom and, with judicious fiddling, can be made to produce virtually any desired behavior, often with both plausible structure and parameter values."

Along with the GIGO approach, other drawbacks to calibration preventing its widespread adoption is the time and labor it takes to calibrate a model. There is also a high dependency on the skill, experience, and judgment of the modeler [13]. Therefore the inherent errors in the calibration procedure must be understood and accounted for in any calibration procedure to realize the full benefit of a calibrated model.

Of the many ways to calibrate a model, manual calibration is the most common. This is an iterative procedure in which the modeler 'tunes' the input parameters to improve the model. The simulation is carried out and the results are analyzed to determine the if what was done had a positive or negative impact on the result. The modeler will go back to tuning until the desired level of accuracy between observed and modeled performance is achieved. This method of calibration requires the most modeler experience, knowledge of subject, and proper justification for parameter

[1]Random alteration to reach the desired result.

changes made.

More robust computer based calibration approaches use statistics based algorithms to determine a best fit. Other methods of calibration include graphical methods, energy signature analysis, statistical methods, and others. A detailed study on these methods has been carried out by Reddy, et. al [13].

2.2 Reddy

All calibration methods attempt to perform an optimization of the model to search for a best solution. Reddy et al. have seen this as a flawed approach as building energy models are a largely undetermined problem, with hundreds or thousands of potential inputs to obtain a small number of outputs with which to perform the calibration. This means that each output is not unique and could have many confounding errors creating a suitable output by chance. The proposed solution Reddy et al. suggest is to look at all the feasible solutions, and those which satisfy the calibration criteria are the seen as a best set of solutions. From this, Reddy et al. developed a process of calibration based on a mid-point LHMC sampling method to search through the inputs for sets of solutions satisfying the calibration criteria.

This builds uncertainty into the calibration itself, providing a range of possible savings for the various ECMs. The method is demonstrated in the recently completed ASHRAE project RP-1051 [13–15], another study done comparing the mid-point LHMC method with manual calibration has been done and concludes that the mid-point LHMC method is most appropriate for models with a high number of uncertain building inputs and when only monthly utility data is available [16].

The mid-point LHMC method discretizes each input parameter into three groups

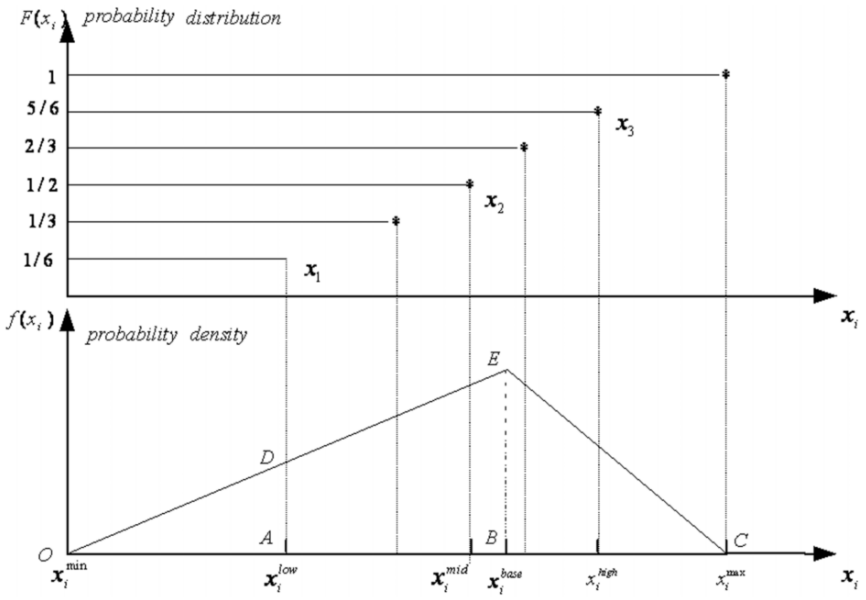

Figure 2.1: Figure from Reddy et al. displays the triangular variable distribution used to discretize each parameter for the mid-point LHMC method.

low, mid, and high; based on the probability density function (PDF) of the parameter. Creating a triangular distribution of the cumulative distribution function (CDF) for each parameter(figure 2.1), using LHMC sampling to generate n samples for each group. From 3^n samples, m are selected at random without replacement[2] to represent one set of possible inputs, this process is repeated until n solution sets are generated. These solutions sets are simulated and compared with observed data, all solutions that are within the calibration criteria are then accepted as feasible solutions[3]. After solution sets are generated a sensitivity analysis to filter strong and weak parameters is carried out on the promising solution sets, using a χ^2 (Chi-

[2]Where p is the total number of parameters $(p_1 \dots p_m)$.

[3]Reddy et. Al later reduced the calibrated solutions to the 20 best calibrations out of thousands of samples.

Square) distribution. The sensitivity analysis allows the weak variables to be fixed, increasing resolution that the strong variables are searched, within the same number of n sample runs. The last step is a fine grid search which uses the results from the best solutions to narrow the parameter search for select variables. Variables which strongly favor one of the three discrete groups (low, mid, high) are further discretized within that group. For example, if the parameter p_1 favors the low group, that parameter will be become $p_{1_{low}}$ and another set of low, mid, high within the low group. This increases the accuracy for those parameters allowing a finer search of that area to reflect a more accurate representation of that parameter's true value. The end result is a set of calibrated solutions that represent the best solutions of the model and can be simulated with selected ECMs to determine the savings and uncertainties associated with each ECM.

2.3 Stochastic LHS Method

Inspired by the mid-point LHMC method, this study presents the stochastic LHS method which shares some features of the mid-point LHMC method. The differences between the methods lie in sampling procedure, methods for sensitivity analysis, building simulation software used, and general order of operating procedure. The process of the stochastic LHS method is as follows:

1. From climate data, building characteristics, operations schedules, and heuristic knowledge a set of input parameters is generated. Parameters which must be estimated due to lack of data, are put through a sensitivity analysis and separated into significant parameters and weak parameters. The weak parameters are fixed, as they have little effect on results whereas the significant

parameters must remain.

2. Significant parameters are assigned a PDF of uniform or normal depending on knowledge of the parameter. Then are put into the LHS generator to create n samples sets to be compiled by the building simulation.

3. All n sample sets are simulated and output data from the simulation is compared with observed performance of the building. The fit between modeled and actual performance for each model is then accepted or rejected by the calibration criteria; models accepted become part of the final set of feasible solutions.

4. The last step is to select ECMs to implement using the feasible solution set. Once simulated, the savings and uncertainty for each EEM can be calculated and used for retrofit decision making.

The entire method is outlined in Figure 2.2, with details of each process further explained in the following chapter.

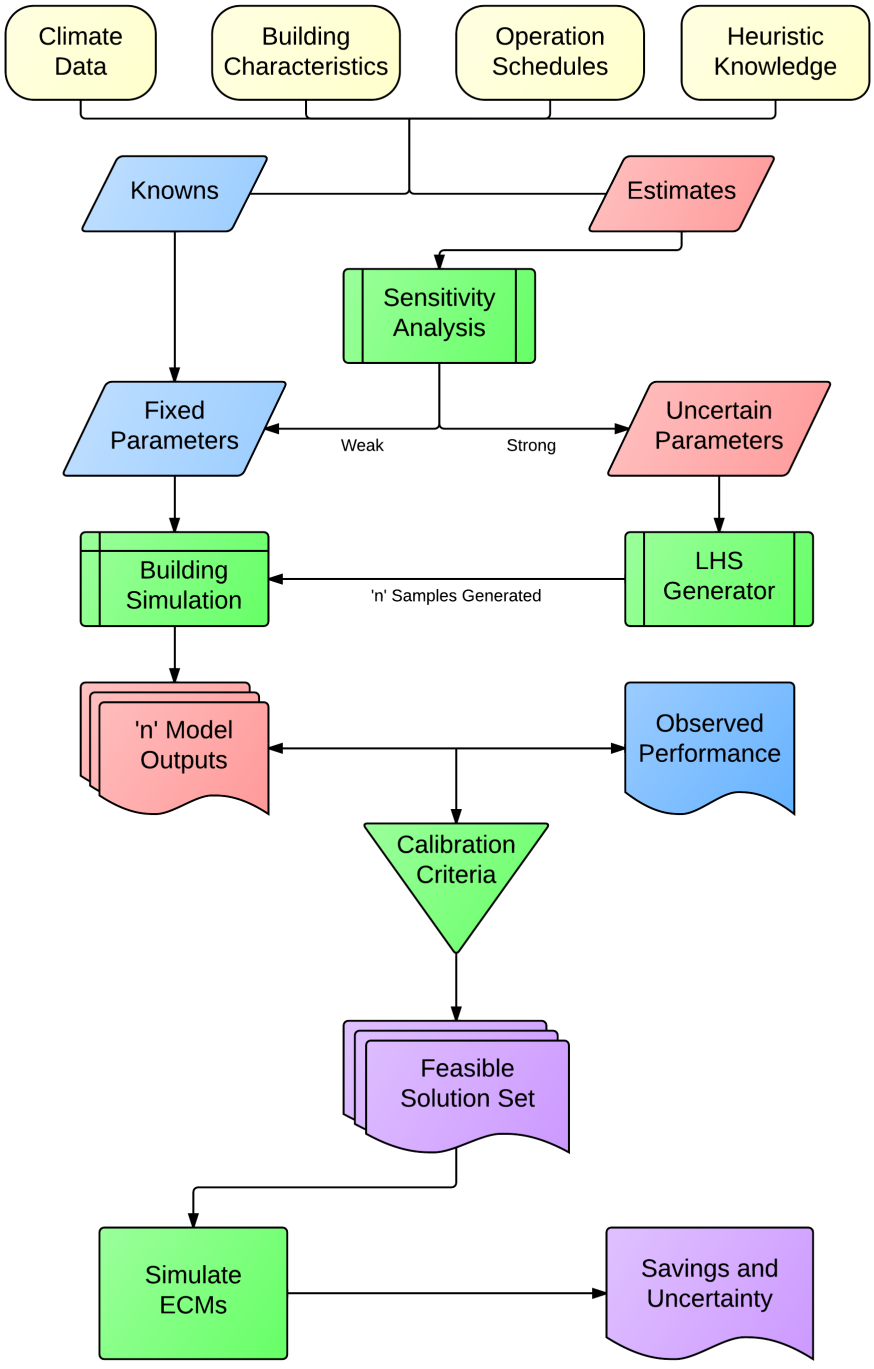

Figure 2.2: Flowchart of Calibration Process

Chapter 3

Methods

3.1 Current Calibration Practice

Of the many ways to calibrate a model, manual calibration is the most common. This is an iterative procedure in which the modeler 'tunes' the input parameters to improve the model. The simulation is carried out and the results are analyzed to determine the impact the changes had on the result. The modeler will go back to tuning until the desired level of accuracy between observed and modeled performance is achieved. This method of calibration requires the most modeler experience, knowledge of subject, and proper justification for parameter changes made.

More robust computer based calibration approaches use statistics based algorithms to determine a best fit. Other methods of calibration include graphical methods, energy signature analysis, statistical methods, and others.

One technique knowns as 'inverse modeling' is wher utility data is used to create a model based on weather data and regression parameters is currently in use through software knwon as First View, develolped by NBI [19, 20]. First View analyzes 8 parameters (table 3.1) to use as performance indicators for the building. By analyzing the results (figure 3.1) and comparing profiles with typical ranges First

Parameter, symbol	Units	Notes
Internal Gain, **Qin**	W/ft2	Solved
External Energy, **Qext**	W/ft2	Fixed ratio of internal gain
Aggregate Normalized UA, **UAn**	BTU/deg F-hr-ft2	Solved
Heating Efficiency, **Eh**	No units	Assumed to be 0.75
Cooling Efficiency, **COP**	No units	Solved
Service Water Heating, **SWH**	Gal/day/ft2	Solved
Heat Intercept, **Ht**	T deg F	Solved
Cool intercept, **Ct**	T deg F	Solved

Table 3.1: Table from Reichmuth and Turner, listing the key parameters analyzed with the First View inverse modeling tool.

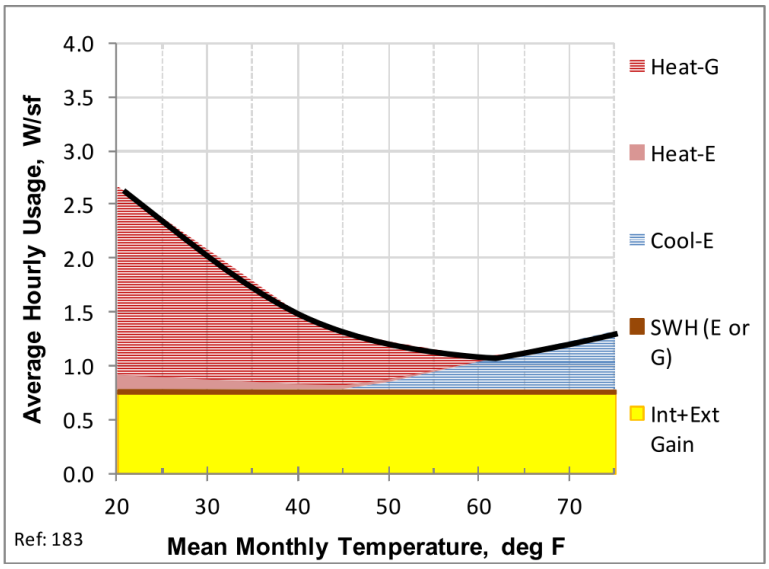

Figure 3.1: Figure from Reichmuth and Turner, displaying the results for a medium-sized Chicago office building from the First View inverse modeling tool.

View detects inefficiencies and makes commissioning or retrofit recommendations based on the findings. The method does not provide a detailed model of the building, nor does it estimate energy savings and uncertainties associated with the suggested improvements.

3.2 Building Simulation Software

The monthly normative energy calculation procedure, specified in ISO13790 [17], uses a utilization factor approach which has been validated through a few studies [21].

The monthly heating demand is calculated by:

$$Q_H = Q_{H,ht} - \eta_{H,gn} Q_{H,gn} \tag{3.1}$$

Where $Q_{H,ht}$ is the heat demand from transmission and ventilation heat transfer, $Q_{H,gn}$ are the heat gains from solar and internal sources, and the $\eta_{H,gn}$ is the gain utilization factor for the heating model. $\eta_{H,gn}$ accounts for the heat gains that are utilized, decreasing overall heating demand.

Similarly, the monthly cooling demand is calculated by:

$$Q_C = Q_{C,gn} - \eta_{C,ls} Q_{C,ht} \tag{3.2}$$

Where $Q_{C,ht}$ is the heat transfer through transmission (through building envelope) and ventilation, during the cooling season, $Q_{C,gn}$ is the heat gains from solar and internal sources, and the $\eta_{C,ls}$ is the loss utilization factor for the cooling model. The loss utilization factor plays a similar role as the gain utilization factor in heating demand calculation. It accounts for the fact that only part of the heat loss through transmission and ventilation is used to decrease cooling load.

$$Q_{ht} = Q_{tr} + Q_{ve} \tag{3.3}$$

$$Q_{tr} = H_{tr}(\theta_{int} - \theta_e)t \tag{3.4}$$

$$Q_{ve} = H_{ve}(\theta_{int} - \theta_e)t \tag{3.5}$$

The above equations illustrate the effect of the temperature difference between the mean internal temperature θ_{int} and the external temperature θ_e. It is important to note that with this normative method the temperature dependence is based on a monthly mean internal and external temperature. The calculation of H_{tr} and H_{ve} are detailed below.

$$H_{tr} = \sum_{i=1}^{n} A_i U_i \tag{3.6}$$

$$H_{ve} = \rho_a c_p \dot{V} \tag{3.7}$$

These equations demonstrate that each i^{th} building element must be taken into account for the overall conduction heat gain/loss coefficient.

$$Q_{gn} = Q_{int} + Q_{sol} \tag{3.8}$$

$$Q_{int} = \left[\sum_{k} \Phi_{int} \right] t \tag{3.9}$$

$$Q_{sol} = \left[\sum_{k} \Phi_{sol} \right] t \tag{3.10}$$

Solar gains and internal gains combine to account for the buildings total gains. Internal gains being from occupants, lighting, and misc. equipment loads within the space. Solar gains are a combination of the short wave radiation gains from the sun, and the long wave radiation losses from each surface to the sky. The below equations demonstrate this.

$$\Phi_{sol} = F_{sh} A_{sol,k} I_{sol,k} - F_{t,k} \Phi_{r,k} \tag{3.11}$$

$$\Phi_r = R_{se} U_c A_c h_r \Delta\theta_{er} \tag{3.12}$$

For opaque elements:

$$A_{sol} = \alpha_{s,c} R_{se} U_c A_c \tag{3.13}$$

For glazing elements:

$$A_{sol} = F_{sh,gl} g_{gl} (1 - F_F) A_{w,p} \tag{3.14}$$

Going back to Equation 3.1, the $\eta_{C,ls}$ is the loss utilization factor for the cooling model. The loss utilization factor plays a similar role as the gain utilization factor in heating demand calculation. It accounts for the fact that only part of the heat loss through transmission and ventilation is used to decrease cooling load. The gain/loss utilization factor is a function of the gain/loss ratio γ_C (or the loss/gain ratio in cooling season) and of a regression parameter a_C that depends on building inertia:

For heating:

$$if \;\; \gamma_H > 0 \;\; and \;\; \gamma_H \neq 1: \;\; \eta_{H,gn} = \frac{1-\gamma_H^{a_H}}{1-\gamma_H^{a_H+1}} \tag{3.15}$$

$$if \;\; \gamma_H = 1: \quad \eta_{H,gn} = \frac{a_H}{a_H+1} \tag{3.16}$$

$$if \;\; \gamma_H < 0: \;\; \eta_{H,gn} = \frac{1}{\gamma_H} \tag{3.17}$$

$$and \;\; \gamma_H = \frac{Q_{H,gn}}{Q_{H,ht}} \tag{3.18}$$

the dimensionless parameter a_H may be calculated by:

$$a_H = a_{H,0} + \frac{\tau}{\tau_{H,0}} \tag{3.19}$$

For cooling:

$$if \;\; \gamma_C > 0 \;\; and \;\; \gamma_C \neq 1: \;\; \eta_{C,ls} = \frac{1-\gamma_C^{a_C}}{1-\gamma_C^{a_C+1}} \tag{3.20}$$

$$if \;\; \gamma_C = 1: \quad \eta_{C,ls} = \frac{a_C}{a_C+1} \tag{3.21}$$

$$if \;\; \gamma_C < 0: \;\; \eta_{C,ls} = 1 \tag{3.22}$$

$$and \;\; \gamma_C = \frac{Q_{C,ht}}{Q_{C,gn}} \tag{3.23}$$

the dimensionless parameter a_H may be calculated by:

$$a_C = a_{C,0} + \frac{\tau}{\tau_{C,0}} \tag{3.24}$$

Where $a_{H,0}$ and $a_{C,0}$ are user determined dimensionless parameters, with default values of $a_{H,0} = 1$ and $a_{C,0} = 1$ for continuously heated/cooled buildings in monthly calculations. $\tau_{H,0}$ and $\tau_{C,0}$ are the reference time constants, with default values of $\tau_{H,0} = 15\,hours$ and $\tau_{C,0} = 15\,hours$ for continuously heated/cooled building in monthly calculations. The building time constant, τ , can be calculated as:

$$\tau = \frac{C_m}{H_{tr} + H_{ve}} \tag{3.25}$$

Where C_m is the building internal heat capacity, H_{tr} and H_{ve} are the transmission and ventilation heat transfer coefficients, respectively.

An example of the ISO13790 implementation used in this study can be viewed in Appendix B.

3.3 Sensitivity Analysis

To increase accuracy and reduce the number of variables included in the Latin Hypercube Sampling process a sensitivity analysis is performed with the method of Elementary effects (EEs). This method evaluates the significance of each variable by determination their respective elementary effect. By examining the mean and standard deviation of these elementary effect the significance of each variable may be determined.

To determine the elementary effects consider k independent variables varied by p levels within a k-dimensional cube, each elementary effect will then be represented by 3.26 below [22]:

$$EE_i = \frac{y\left(x_1, x_2, \ldots, x_{i-1}, x_i + \Delta, \ldots x_k\right) - y\left(x_1, x_2, \ldots, x_k\right)}{\Delta} \tag{3.26}$$

Where Δ is a value in $[\frac{1}{p-1}, \ldots, 1 - \frac{1}{p-1}]$.

The morris method of sensitivity analysis uses the method of elementary effects to experiment with different combinations of variables and their values to rank each variable according to their largest effect. A flowchart of this process is located in Appendix D; the MATLAB code for the morris experiment is also located in Appendix D.

3.4 Latin Hypercube Sampling

The LHS method is an evolution of the stratified sampling method. The stratified sampling method divides the probability distribution of the target variable into K strata of equal probability and each stratum has n random samples generated. The drawback of the stratified sampling method for high-dimensional problems is that the total number of strata grows exponentially as the number of variables m increases. The sample size N of a stratified sampling method is: $N = K^m xn$. The LHS method samples on the m-dimensional hypercube in a way that only the marginal distributions are stratified. Figure 3.2 (b) shows an illustration. Compared to the stratified sampling shown in Figure 3.2(a) not all cells in Figure 3.2(b) have the same number of samples (Figure 3.2 taken from Kroese, Taimre et al. [23]). Instead, both the horizontal and vertical coordinates are stratified into K^1 stratum with 30 samples each. A previous study by [24] shows that the LHS method, compared to random sampling and stratified sampling, is more robust and leads to less variance

[1] $K = 5$ in the case illustrated.

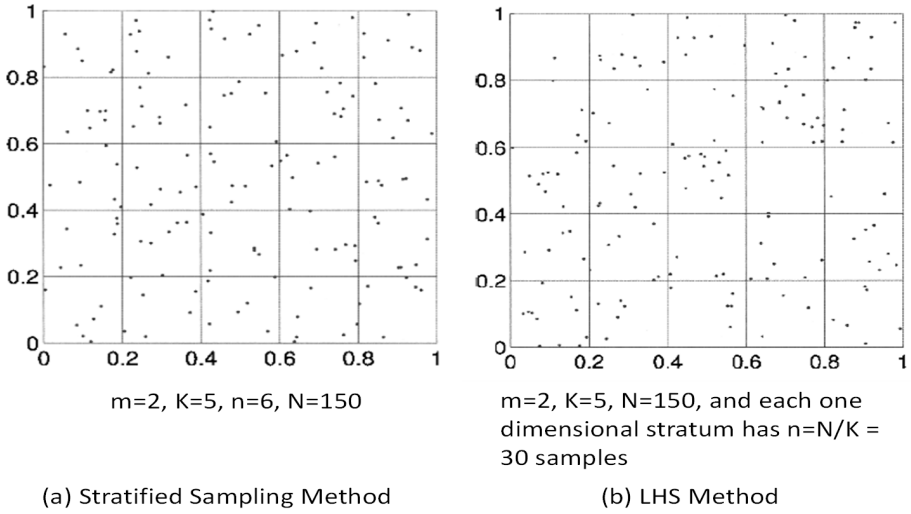

Figure 3.2: Illustration of Latin Hypercube Sampling.

in the expected mean in building simulation related applications.

The MATLAB code in Appendix E was used to generate the n samples required for the simulation.

3.5 Sample Size

Sample size recommendations from previous studies by Ricker [16] and Reddy [14] are relatively arbitrary and suggest samples within the range of 2000 up to 10000. For this study there will be a minimum sample size based on the central limit theorem and a confidence interval associated with that sample size. The goal being to choose a sample large enough so the error of the sampling method itself does not compound with the uncertainty of the modeling method and results.

Though the stochastic LHS method is the most robust sampling technique, compared to random sampling and stratified sampling method, a proper sample size is

required to achieve a certain level of accuracy. A common measure for the level of accuracy of a Monte Carlo Monte Carlo (MC) simulation is called the coefficient of variation Coefficient of Variation (CoV) of the sample mean, as shown in Equation 3.27.

$$CoV_{MC} = \frac{\sigma_{\bar{X}}}{\bar{X}} \tag{3.27}$$

Where $\sigma_{\bar{X}}$ is the standard deviation of the sample mean, $\bar{X}$. According to the central limit theory, $\sigma_{\bar{X}}$ can be estimated through Equation 3.28:

$$\sigma_{\bar{X}} = \frac{\sigma_x}{\sqrt{n}} \tag{3.28}$$

Where σ_x is the standard deviation of the population x, and n is the sample size. Let $\bar{X}$ represent a sample mean of a monthly building energy consumption (in any energy source type) and μ represent the true monthly building energy consumption. The confidence interval of μ at a confidence level of $(1-\delta)$ can be calculated as:

$$Pr(\bar{X} - Z_{\frac{1-\delta}{2}} \times \sigma_{\bar{X}} \leq \mu \leq \bar{X} + Z_{\frac{1-\delta}{2}} \times \sigma_{\bar{X}}) = 1 - \delta \tag{3.29}$$

Theoretically a student t distribution should be used in Equation 3.29 instead of the normal distribution since the standard deviation of monthly building energy consumption is unknown. But the difference between the student t distribution and the normal distribution becomes practically negligible when sample size grows larger than 30. Past research shows a sample of size 100 is often required for a MC simulation to converge in building applications. Therefore the normal distribution is used in Equation 3.29 for simplicity. Based on Equation 3.29 the confidence range

(θ) of monthly energy consumption μ can be calculated as:

$$\theta = 2 \times Z_{\frac{1-\delta}{2}} \times \sigma_{\bar{X}} \tag{3.30}$$

As the calibration criterion for normalized mean bias error (NMBE) is 5% specified by ASHRAE Guideline 14 [25], measures have to be taken to ensure that θ for any monthly energy consumption is significantly smaller than the required calibration criterion for NMBE. A conservative measure is taken in this study: the confidence range θ of any monthly energy consumption has to be no larger than some safety factor SF of the NMBE calibration criterion. That is:

$$\theta = 2 \times Z_{\frac{1-\delta}{2}} \times \sigma_{\bar{X}} \leq 2 \times \frac{5\%}{SF} \times \mu \tag{3.31}$$

Since the true monthly building energy consumption μ is unknown, the sample mean $\bar{X}$ is used to approximate: $\mu \approx \bar{X}$. Thus the proper sample size has to be at least large enough to meet the accuracy level specified in Equation 3.32 at a given level of confidence $(1 - \delta)$.

$$CoV_{MC} \leq \frac{\frac{5\%}{SF}}{Z_{\frac{1-\delta}{2}}} \tag{3.32}$$

In accordance with Equation 3.29, the level of confidence and value of the produce different required CoV values. This study aims for a minimum confidence of 68% and of $1/10^t h$, to lessen the impact of sampling error, which corresponds with a sample size of $n = 5000$.

To reach a 99% confidence interval with a of $1/100^t h$ would require a half million samples in this instance. Due to computing restrains this would be an unreasonably

large sample size for most applications, however with the LHS method it is possible to simulate a sample this large within an hour. But, to keep the method applicable to other building simulation software packages, 5000 will be the minimum sample size.

3.6 Calibration Criteria

The quality of a calibration can be considered a goodness of fit (GOF) between the model prediction and the measurements. Statistics used to measure the goodness of fit are the NMBE and the coefficient of variance of root mean square error (CV_{RMSE}) [25, 26].

$$NMBE = \frac{\sum_{i=1:n} Actual_i - Modeled_i}{(n-1)\ Mean(Actual)}\ 100\% \tag{3.33}$$

$$CV_{RMSE} = \sqrt{\frac{\sum_{i=1:n}(Actual_i - Modeled_i)^2}{n-1}} \div \bar{x} \times 100\% \tag{3.34}$$

Actual represents metered utility data and $Modeled$ is the simulation result. The calibration residual is defined as the difference between the modeled and measured building performance. NMBE measures the mean of calibration residuals, showing how much a calibrated model over or under estimates building performance compared to the actual measurement. NMBE alone is not sufficient to measure the GOF, as the monthly fluctuations may cancel each other and result in a small NMBE. The CV_{RMSE} is introduced to account for these fluctuations and is essentially the standard deviation of calibration residuals; showing how wide the variation of the residual is. Thus it is possible to have an acceptable NMBE and reject the CV_{RMSE} or the opposite [27]. Since a building often consumes energy from multiple

energy sources, including electricity, gas, etc., one needs an aggregated index to measure the overall goodness of fit. A common practice is to weigh the calibration statistics, NMBE and CV_{RMSE}, by the contribution each energy source makes to annual building energy cost. The weighted indices can be calculated as follows:

$$Overall\ NMBE = \sqrt{\frac{w_{kWh}^2 NMBE_{kWh}{}^2 + w_{Therm}^2 NMBE_{Therm}{}^2}{w_{kWh}^2 + w_{Therm}^2}} \tag{3.35}$$

$$Overall\ CV_{RMSE} = \sqrt{\frac{w_{kWh}^2 CV_{kWh}^2 + w_{Therm}^2 CV_{Therm}^2}{w_{kWh}^2 + w_{Therm}^2}} \tag{3.36}$$

Where $w_{_{kWh}} + w_{_{Therm}} = 1$

Aggregating once more, the overall GOF of a calibration is calculated by combining the NMBE and CV_{RMSE} with a different set of weighting factors. There is no established rule on what values to assign to either weight factor, however the recommendation from ASHRAE Guideline 14-2002 [25] is to use a 1:3 weight for CV_{RMSE}:NMBE , reflecting the preference of building energy managers to capture the annual energy consumption more accurately than the monthly variation. The study by Reddy, Maor et al. [13] also uses a weight of 1:3 for CV_{RMSE}:NMBE.

$$Overall\ GOF = \sqrt{\frac{w_{NMBE}^2 NMBE_{Overall}^2 + w_{CV_{RMSE}}^2 CV_{Overall}^2}{w_{NMBE}^2 + w_{CV_{RMSE}}^2}} \tag{3.37}$$

Where $w_{CV_{RMSE}} + w_{NMBE} = 1$.

Multiple organizations adopting M&V guidelines, including ASHRAE, EERE, and EVO, have all published recommendations on the acceptable levels of NMBE and CV_{RMSE} [25, 26, 28]. EERE adopts the standards set by ASHRAE Guideline

14 [25] which suggests $NMBE \leq 5\%$ and $CV_{RMSE} \leq 15\%$ for calibration on a monthly basis[2]. This translates to a $GOF \leq 11\%$ which is considered a calibrated solution in the ASHRAE Project RP-1051 [13].

The implementation of the calibration criteria uses MATLAB and the source code can be viewed in Appendix F.

3.7 Process

To sample and simulate each of these runs 5000 runs parameters are "injected" into the simulation via an $(n \times k)$ matrix and then stepped through 'n' iterations where each 'k' variable is unpacked to the $x(i^{th})$ value.

Before injecting, the parameters are 'packed' such that array '$x(n, k)$' has a unique position for each variable 'k'.

Consequently $x(1, k)$ is the i^{th} variable and as long as $a = x(i)$, $b = x(i+1)$, etc. as the iteration continues; then each variable is assigned its appropriate position within the simulation.

This process is repeated for each of 'n' unique iterations for the test required. With a simulation being completed and results output for each iteration.

This study begins with a morris sensitivity analysis; then a LHS simulation, a calibration process, and the retrofit ECMs being implemented at the end.

[2]For hourly calibration ASHRAE Guideline 14 suggests $NMBE \leq 10\%$ and $CV_{RMSE} \leq 30\%$

Chapter 4

Case Study

To demonstrate the calibration methods outlined in Chapter 3 a building that represents the aging building stock in need of retrofit is presented.

4.1 Nueberger Hall

Neuburger Hall (NH) at Portland State University (PSU) is a six story multi-use building with classrooms, administrative service offices, computer labs, and other facilities; originally built in 1960 and expanded in 1966. The footprint measures 200x200ft and contains a basement, four main floors, three mezzanine floors, and a mechanical penthouse. Structurally, Neuburger Hall is primarily concrete with single pane glazing, aluminum insulating panels and brick veneer.

Neuburger Hall is served by PSU's central steam and chilled water loop, circulated throughout NH via a constant vir volume (CAV) air handling unit (AHU). Multiple air source heat pumps located on the roof serve the computer labs on the fourth floor which are not serviced by the main AHU[1].

Occupant and building operation schedules are based on observed university schedules, and private communications with campus facility personnel. Internal

[1]This is marked by grey shade in Figure 4.1

Figure 4.1: Photo of Neuberger Hall

gains are calculated based on lighting, equipment, and occupancy estimates from site visits, university course catalogs, building layout, and university office schedules [29, 30].

Fresh air ventilation for is determined in accordance with ASHRAE 62.1 [31], however different ventilation requirements in the 1960s may have been, NH is known to be over ventilated[2]. Power ratings for major mechanical systems components, such as fans, pumps, and the like; are acquired from a building audit report conducted by Interface Engineering [32].

Ideally, on-site weather data for the 2003-2006 academic years is used, however there is none. Instead, weather data for these years comes from a commercial weather service, based on a new coupled global National Center for Environmental

[2]Malfunctioning air dampers according to site visits and communications with PSU facilities [29]

Prediction (NCEP) reanalysis at unprecedented spatial[3], vertical[4], and temporal[5] resolution [33].

Utility data is available intermittently from the years 2003-2011, condensate water is available for 2003-2010, while electricity is available from 2003-2006 and 2010-2011. There is also no record of chilled water use. Due to these limitations this research bases its calibration studies upon electricity consumption and space heating energy use for the 2003-2006 academic years.

Table 4.1: Utility data provided by PSU F&P department for the years 2003-2006.

	Condensate (KGAL)			Electricity (kWh)		
	2003-2004	2004-2005	2005-2006	2003-2004	2004-2005	2005-2006
JUL	1	0	0	275,062	277,090	277,062
AUG	0	0	0	289,939	282,472	289,839
SEP	0	0	0	271,057	287,143	270,457
OCT	45	67	83	306,590	309,546	306,590
NOV	122	118	131	282,132	285,088	282,132
DEC	119	84	155	247,817	250,773	247,817
JAN	192	176	137	289,682	292,638	289,682
FEB	120	102	129	276,530	279,486	276,530
MAR	87	90	108	283,470	286,426	283,470
APR	63	83	89	292,667	295,623	292,667
MAY	30	33	33	304,613	307,569	304,613
JUN	0	13	0	300,717	302,281	297,605

Notice internal temperature is an uncertain parameter, this is because the model simulates the difference between the exterior and interior. The set-point temperature of the thermostat will regulate heating and cooling, but does not indicate what the actual internal temperature may be and thermostats sometimes have a dead band

[3] $0.5^\circ \times 0.5^\circ$

[4] 37 pressure levels for the atmosphere and 40 levels for the ocean

[5] Hourly

Table 4.2: List of input parameters

Discrete Parameters	Continous Paramters
Fan Power	Outside Air Flow Rate (Building)
Pump Power	Outside Air Flow Rate (Lab)
Occupant Internal Gains (Building)	Thermal Bridge (Building)
Occupant Internal Gains (Lab)	Thermal Bridge (Lab)
Lab Electricity Usage	Heat Exchanger Efficiency
Monthly Schedule (Building)	SHGC
Monthly Schedule (Lab)	Lighting Power Density
Lighting/Plug Load Schedule	Plug Load Power Density
North Shading	Internal Temp H
East Shading	Internal Temp C
South Shading	External Convection Coef.
West Shading	

in which they do not operate.

4.1.1 Utility Data Reduction

According to the PSU Master Plan [30] steam is supplied at $10.3 psig$ and condensate water cools to approximately $200°F$ where it is metered. Using the steam tables h_g for the saturated steam is $2699.55\frac{kJ}{kg}$ and h_f for condensate is $419.1\frac{kJ}{kg}$. The difference being the net energy supplied to the building per kilogram of steam. Condensate water is given in kilo-gallons ($KGAL$), converting gallons to m^3 by dividing by $264.2\frac{gal}{m^3}$ then multiplying by the density of the condensate will give kilograms. Then the product of kilograms and the difference between the enthalpies of the steam and condensate are converted to joules to be used in the calibration.

$$Q_{heat} = (h_g - hf) \times 1000 \times \frac{GAL}{264.2\frac{gal}{m^3}} \times \rho_{condensate} \tag{4.1}$$

4.2 Sensitivity Analysis

The sensitivity analysis analyzes the unknowns that are sampled and determines which unknown parameters are significant with respect to the output. This reduces the total number of parameters required for each sample and sampling less parameters allows the sample generator to search more efficiently, increasing the resolution of the significant parameters [34].

Each parameters significance is determined by the method of EEs. This method is applied by generating a sample matrix containing each of the unknown parameters. Each sample is simulated and the outputs recorded[6]. The elementary effects for each parameter are calculated according to Equation 3.26, and the primary indicator of a parameters significance is the μ^* of the EEs for that parameter and output. There will be a μ^* of a parameter for each output[7]. Once complete, the parameters with the largest μ^* values are significant to their respective outputs they represent.

However, the final selection of significant parameters must contain the same level of variation as the original set of parameters. Without this step, the new parameter set will not truly represent the original parameter set. This is done by comparing the variances of the outputs from the original set of parameters with the variances of the modified set of parameters. With this complication, reducing the number of parameters becomes a trial and error process.

After selecting variables with large and absolute significance, there are a large amount of variables with relatively small influence, but do play an important role in the validation procedure presented. Most of these are parameters that vary each

[6]In this case monthly and annual electricity and heat energy are the outputs

[7]The outputs in this case are heating and electricity for each month and an annual value for each. Twenty-four per parameter

month, but may have the same mean value and standard deviation as estimates for each month. Reducing these parameters from discrete to continuous creates one or two significant parameters rather than multiple less significant parameters. This reduces the original list of 23 parameters with 155 inputs to just 8 parameters with 16 total inputs.

Table 4.3: Parameter list, post sensitivity analysis

Input Parameters	
Outside Air Flow Rate	Monthly Schedule (Building)
Fan Power	Plug Load Power Density
Pump Power	Lighting/Plug Load Schedule
Thermal Bridge (Building)	Internal Temp H

A graphic illustrating this process can be found in Appendix D.

4.3 Settings

As noted in Equations 3.35 and 3.36 weighting factors may be assigned to each energy source to adjust the calibration to favor a better fit for one energy source over another. Building managers may wish to have a balanced fit between all energy sources or may be interested in identifying a fit that emphasizes energy costs. Accomplishing these two goals is done by having the the energy factors, w_{kWh} and w_{Therm}, equal one another for an even weight, or have them fluctuate as a ratio of price against one another such that $\frac{w_{kWh}}{w_{Therm}} + \frac{w_{Therm}}{w_{kWh}} = 1$ is true.

Once price or even weighting factors are established the GOF weights, w_{NMBE} and $w_{CV_{RMSE}}$, can be adjusted to to the ratio desired as well. As in the ASHRAE project a balance between these variables satisfying a $GOF \leq 11\%$ may be used[8].

[8]This will strictly average the NMBE and CV_{RMSE}

Or, prior to the GOF calculation, the NMBE and CV_{RMSE} may be restricted to 5% and 15%, respectively.

Altering the calibration with the calibration criteria through these weighting factors changes the results achieved dramatically as is demonstrated in Section 4.4. Table 4.4 displays these options.

Table 4.4: Calibration settings for price weighting (A), even weighting (B), and strict criteria (C).

	A	B	C
GOF	11%	11%	11%
MBE	5%	5%	5%
CV RMSE	15%	15%	15%
wkWh	3/4	1/2	-
wTherm	1/4	1/2	-

4.4 Results

Simulating 5000 samples generated for each case from Table 4.4 gives the results displayed in Table 4.5. All simulations were done with the same year, 2005-2006, and found 20 solutions for case A, 15 for case B, 920 for case C, and 162 for case D. Years 2003-2004 and 2004-2005 are simulated with the same solution sets generated by each case for validation.

Table 4.5 represents a GOF of the individual energy sources, such that only the NMBE and CV_{RMSE} for electricity are used in Equation 3.37. This is done for the sake of simplicity. Electricity generally has a good fit no matter the setting, with a slightly better fit for setting D, where a $NMBE_{Overall}$ and $CV_{_{RMSE}Overall}$ of 5% and 15% are allowed, including price weighting. The values for setting C

are the worst for heating, which also includes price weighting, and allows 15% for either $NMBE_{Overall}$ or $CV_{RMSEOverall}$. Settings A and B have similar results for both energy sources, each setting giving equal weight to the energy sources, and the difference between them being the $NMBE_{Overall}$ of 15% for A and 5% for B.

Table 4.5: GOF distribution for settings A-D for each energy source. Calibrated for 2005-2006 and validated with the 2003-2004 and 2004-2005 years.

		Heating			Electricity		
Year	Set	5th	Median	95th	5th	Median	95th
2005-2006	A	9.9%	12.3%	14.5%	2.6%	4.2%	9.1%
2004-2005	A	13.3%	16.9%	24.1%	2.5%	3.5%	9.8%
2003-2004	A	10.6%	15.7%	22.6%	2.7%	4.2%	9.1%
2005-2006	B	9.4%	12.3%	14.4%	2.5%	3.8%	8.6%
2004-2005	B	13.3%	16.7%	24.9%	2.5%	3.3%	9.8%
2003-2004	B	10.9%	16.6%	23.7%	2.6%	3.8%	8.7%
2005-2006	C	16.3%	29.0%	64.3%	2.5%	4.4%	8.4%
2004-2005	C	17.5%	38.8%	74.4%	2.4%	3.8%	8.5%
2003-2004	C	17.0%	30.0%	65.4%	2.6%	4.5%	8.4%
2005-2006	D	13.5%	20.7%	25.3%	2.4%	3.7%	6.2%
2004-2005	D	15.3%	25.5%	33.6%	2.5%	3.2%	6.4%
2003-2004	D	14.0%	21.4%	29.3%	2.6%	3.8%	6.2%

Given the results in Table 4.5, this shows that while electricity may carry more weight due to price, the fit of electricity is not greatly improved by enforcing this within the calibration criteria. As mentioned earlier in Section 2.2, the idea of this calibration method is a small set of quality calibrated building models to mitigate the uncertainties associated with building retrofits. So while price weighting the calibration will give a larger number of results[9] the better fit represented by moving to an equal price weight provides a smaller set of higher quality calibrations.

[9] 920 and 162 solutions for C and D settings vs. 20 and 15 solutions for settings A and B.

Due to the good overall fit of electricity in all cases and years, it is likely that when electricity is aggregated with heating it produces an acceptable result. If these energy sources were not aggregated, and instead, were to be evaluated on their individual GOFs most solutions would not meet this criteria, and in fact only 2 solutions meeting this criteria for 2005-2006 were found. To analyze a sample set without combining electricity and heating, but have them meet the criteria independent of one another; a much larger sample size would be required.

Chapter 5

Analysis

Understanding each sample population from price weighting, even weighting, or enforcing the calibration criteria for each energy source individually, is important to reducing uncertainty in calibrated building models.

Recall from the results in Section 4.4 that there are very few Strict criteria enforcement (strict) criteria enforcement solutions generated from the minimum sample size, $n = 5000$. To increase the number of strict solutions there are two general methods. To use better data, using measure values instead of uncertain parameters (where possible). The other options are to use a more efficient sampling algorithm or increase the number of samples.

5.1 Large Sample Size

The results in section 4.4 used the minimum sample requirement of 5000, determined in section 3.5, for which only 1 or 2 samples may meet the requirements of the strict criteria, in some cases no samples will meet this criteria. To generate a large enough solution set for analysis the number of samples are increased from 5×10^3 to 5×10^5. This is still a small size relative to the possible combinations of variables, but offers larger resolution to search more parameter combinations.

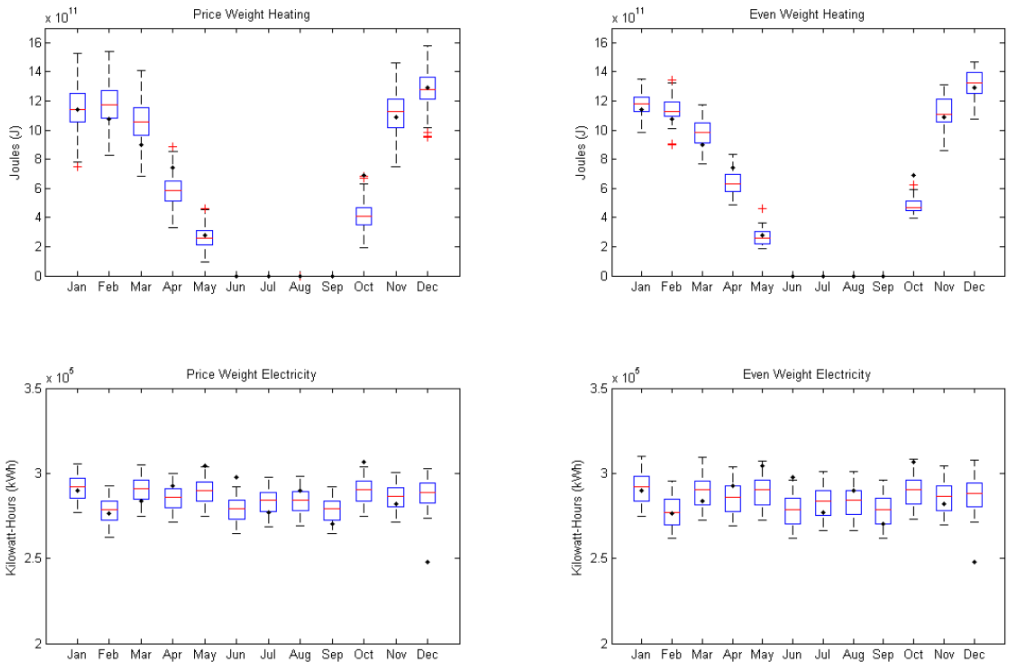

Figure 5.1: Boxplots displaying the interquartile range of Heating and Electricity predictions for strict weight solutions for the 2005-2006 year, $n = 5x10^5$.

Results from the large sample simulations for the strict solutions are in Table 5.1. These show marked improvement for the calibration year, however comparing these results with Table 4.5 for settings A and B there is only marginal improvement for the 2003-2004 validation and poor performance with the 2004-2005 validation years. By themselves the validations do not show good agreement for heating, but there is better agreement for electricity again compared with Table 4.5.

Increasing the resolution of the sampling procedure with a larger sample size shows there are a larger number of strict solutions, however their validation between years for heating leaves room for improvement. Ensuring the strict solutions between years do not belong to the same sample population a hypothesis test is used.

Table 5.1: Large sample size results for strict criteria with $*$ indicate the calibration year and the other years are for validation.

	Heating			Electricity		
Year	5th	Median	95th	5th	Median	95th
2005-2006*	7.4%	10.2%	11.0%	2.9%	4.6%	6.7%
2004-2005	11.3%	15.9%	20.3%	2.5%	3.4%	6.8%
2003-2004	11.2%	14.9%	19.5%	3.0%	4.7%	6.7%
2005-2006	11.2%	17.4%	27.7%	2.6%	4.1%	6.9%
2004-2005*	8.2%	9.9%	10.8%	2.5%	3.9%	5.4%
2003-2004	14.4%	21.6%	36.0%	2.7%	4.2%	6.9%
2005-2006	12.1%	15.9%	20.0%	3.4%	5.5%	7.5%
2004-2005	12.5%	17.3%	21.8%	2.5%	4.3%	7.8%
2003-2004*	7.5%	9.9%	10.8%	3.5%	5.5%	7.5%

The Hotelling T^2 Test is a multivariate hypothesis test of the means between two sample populations to determine if the samples belong to the same population. It is useful in determining if the resultant solution set from one year belong to the same population from another year, or in determining if the solution sets found by the various calibration settings are members of the same population. It is being used for its robustness and compatibility with large sample sizes from multiple variables for each sample set. Ideally sample sets being tested should be normally distributed but this test handles non-normal distributions very well as noted by Everitt [35].

Using this test confirms the results seen in Table 5.1, that the strict solutions for each year are coming from different populations (rejecting the null hypothesis that $H_o = \mu$). Most variables in the solutions sets tested are normally distributed, save 2-4 variables, for some sets.

5.2 Refinement

From the previous section there is a suitable population of solutions from each of the calibration criteria to analyze and compare with one another. This would not have been possible without increasing the sample size from $n = 5x10^3$ to $n = 5x10^5$. This may indicate that the uncertain variable table has ranges which are too broad to locate a suitable amount of strict solutions and a possible method of correction to this table is through the refinement process.

The refinement process involves taking the population of solutions from the strict solutions from Section 3.5 and assigning a normal distribution to each parameter with a mean and standard deviation. These become the new uncertainty table that are sampled from with the minimum sample size and analyzed to determine the effectiveness of this correction. Effectiveness is determined by the GOFs for the energy sources and how other years calibrate and validate with this new uncertainty table.

Solution sets generated from this procedure display marked improvement in the quantity of strict criteria solutions achieved, but the overall validation ranges are similar to those displayed in Table 5.1.

Chapter 6

Conclusion

In review this calibration method relies on LHS and morris method sensitivity analysis to sample and determine significance in unknown parameters. The parameters are then simulated with a calibration year and deemed solutions based on criteria defined in ASHRAE Guideline 14 for $NMBE$ and CV_{RMSE} depending on the various weighting parameters used. These solutions are checked for validity between other years of data and then could be used to simulated a set of ECMs.

6.1 Future Work

6.1.1 Retrofits and Uncertainty

After generating a validated solution set the retrofit energy simulations can be done. Capturing the uncertainty associated with these results should be done to understand the limitations of the data being presented and increasing its usefulness.

According to ASHRAE Guideline 14,

> 'Good data' does not describe data that yield the desired answer; it describes data that yield a result within the intended uncertainty interval.

Fulfillment of the uncertainty calculations for the proposed ECMs is done in two ways:

1. Usage of the mean result for a set of solutions within the solution populations to serve as the uncertainty.

2. Use uncertainty equations developed for single calibrated models in ASHRAE Guideline 14, where the mean values would serve as the inputs.

Since the LHS method may stand on its own, the comparison of these two uncertainty calculations is interesting. The stochastic LHS method of calculating the uncertainty based on the range of model results from each population and the method developed by Reddy and Claridge [36] utilizing the CV_{RMSE}, savings, and number of months post retrofit modeled to gather uncertainty.

6.1.2 Additional Research

Firstly, the method must be applied to a broader selection of buildings including office buildings, multi-family residential, and such. Buildings with adequate utility data, simple HVAC systems, and are in need of some form of energy improvement would be the best candidates. Most of the effort for this study has been on the method itself and centered around a single multi-use university building.

Subsequently, work may also be done on improving the algorithms within the study too. This can include but is not limited to applying parts of stochastic LHS method to other building simulation programs such as eQuest, EnergyPlus, and others. Also conducting comparative studies between the aforementioned simulation programs and the ISO13790 simulation approach will provide insight into how best to implement the method as there are multiple ways to use the ISO13790 approach including: single zone, core and shell, multi-zone, seasonal or monthly timestep, or an hourly timestep.

Focusing more on the sampling and analysis by providing a streamlined simulation, analysis, and calibration approach integrated with neural networks and deep learning algorithms would also be a huge improvement in this stochastic LHS process.

6.2 Considerations

The stochastic LHS approach is a step forward in the process of building energy model calibration. The strength of the method is derived from taking a closer look at the assumptions that go into an energy model and filtering out the interactions that either have no effect on the result, or interactions that would invalidate the model.

This can lead to mixed results, with utility data pointing the way towards the 'correct' answer. It is possible to arrive at the desired result according to the data, however if the data itself is not accurate, then the model will be flawed as a result. Looking back at Table 4.1 in Chapter 4 this is absolutely the case. Close inspection of the electricity usage for the 03-04 year compared with 05-06 shows almost identical energy usage for the two years which is most likely due to human error in collecting the data.

6.3 Final Thoughts

Packaged calibration tools are still in development and there is no single 'best' option. Compared with traditional building energy modeling retrofit projects, calibration provides the much needed focus on uncertainty. Results generated without regard to uncertainty have little significance comparatively.

This stochastic LHS method presents a new trend in the development of tools for MV. The importance of uncertainty analysis and risk assessment in the face of commercial building retrofits cannot be understated. As the method matures, this simple straightforward tool may prove invaluable for building owners and ESCOs.

As countries strive to be more energy independent and while new building energy consumption is on the decline it is important to bring the current building stock up to modern standards. With the increasing capabilities of computers, more efficient modeling techniques are being developed such as the one presented in this study. With the improvement of building energy model calibration retrofits can make the transition to transparency and quality investments, saving energy, money and resources over the long term.

Bibliography

[1] L Perez-Lombard, J Ortiz, and C Pout. A review on buildings energy consumption information. *Energy and buildings*, 40:394–398, 2008.

[2] Corey Cattano, Rodolfo Valdes-Vasquez, and Leidy Klotz. Barriers to the Delivery of Building Renovations for Improved Energy Performance: A Literature Review and Case Study. *ICSDC 2011@ sIntegrating ...*, (864):203–210, 2012.

[3] Victor Olgyay and Cherlyn Seruto. Whole-building retrofits: A gateway to climate stabilization. *ASHRAE Transactions*, 116:1–8, 2010.

[4] Brian Coffey, Sam Borgeson, Stephen Selkowitz, Joshua Apte, Paul Mathew, and Philip Haves. Towards a very low-energy building stock: modelling the US commercial building sector to support policy and innovation planning. *Building Research & Information*, 37(5-6):610–624, nov 2009.

[5] US DOE Energy Efficiency and Renewable Energy. Buildings Energy Databook, 2012.

[6] White House. We Can't Wait: President Obama Announces Nearly $4 Billion Investment in Energy Upgrades to Public and Private Buildings., 2011.

[7] USGB Council. LEED 2009 for new construction and major renovations. 2008(November 2008), 2010.

[8] Architecture 2030. Architecture 2030 Challenge, 2011.

[9] S Pacala and R Socolow. Stabilization wedges: solving the climate problem for the next 50 years with current technologies. *Science (New York, N.Y.)*, 305(5686):968–72, aug 2004.

[10] Cathy Turner, M Frankel, and USGB Council. *Energy performance of LEED for new construction buildings*. 2008.

[11] Mushtaq Ahmad and CH Culp. Uncalibrated building energy simulation modeling results. *HVAC&R Research*, 12(4):1141–1155, 2006.

[12] A Rabm. Parameter Estimation in Buildings : IVIethods for Dynamic Analysis of IVIeasured Energy Use. 110(February), 1988.

[13] T Agami Reddy and Itzhak Maor. Final Report : Procedures for Reconciling Computer-Calculated Results With Measured Energy Data Investigators. Technical Report January, ASHRAE, 2006.

[14] TA Reddy, Itzhak Maor, and Chanin Panjapornpon. Calibrating Detailed Building Energy Simulation Programs with Measured Data—Part I: General Methodology (RP-1051). *HVAC&R Research*, (January 2012):37–41, 2007.

[15] TA Reddy, Itzhak Maor, and Chanin Panjapornpon. Calibrating detailed building energy simulation programs with measured data—part II: application to three case study office buildings (RP-1051). *HVAC&R Research*, (January 2012):37–41, 2007.

[16] Elizabeth Ricker. *Assessing methods for predicting retrofit energy savings in buildings: case study of a Norwegian school*. Masters thesis, Massachusetts Institute of Technology, 2008.

[17] ISO. 13790: Energy performance of buildings—Calculation of energy use for space heating and cooling. Technical report, International Orginzation for Standardization, 2008.

[18] G.M. Hornberger and R.C. Spear. Approach to the preliminary analysis of environmental systems. *Journal of Environmenal Management*, 12(1):7–18, 1981.

[19] Howard Reichmuth and Cathy Turner. A Tool for Efficient First Views of Commercial Building Energy Performance. pages 325–338, 2010.

[20] New Buildings Institute. FirstView Overview. Technical report, 2012.

[21] V. Corrado and H. E. Mechri. Uncertainty and sensitivity analysis for building energy rating. *Journal of building physics*, 33(2):125–156, jun 2009.

[22] Andrea Saltelli, Marco Ratto, Terry Andres, Francesca Campolongo, Jessica Cariboni, Debora Gatelli, Michaela Saisana, and Stefano Tarantola. *Global Sensitivity Analysis*. John Wiley & Sons Inc., Chichester, UK, 2008.

[23] Dirk P Kroese, Thomas Taimre, and Zdravko I Botev. *Handbook of monte carlo methods*, volume 706. John Wiley & Sons, 2013.

[24] IA Macdonald. Comparison of sampling techniques on the performance of Monte Carlo based sensitivity analysis. *Proceedings Building Simulation*, (3):992–999, 2009.

[25] ASHRAE. ASHRAE Guideline 14 - 2002. Technical report, American Society of Heating, Refrigeration, and Air Conditioning Engineers, Atlanta, GA, 2002.

[26] FEMP. M & V Guidelines : Measurement and Verifi cation for Federal Energy. Technical report, US DOE, EERE, Federal Energy Management Program, 2008.

[27] Jeff S Haberl, David E Claridge, and Charles Culp. ASHRAE's Proposed Guideline 14P for Measurement of Energy and Demand Savings: How to Determine What was Really Saved by the Retrofit. *First International Conference for Enhanced Building Operations*, pages 1–13, 2005.

[28] EVO. International Performance Measurement and Verification Protocol. Technical report, Efficiency Valuation Organization, 2012.

[29] PSU F&P. Neuberger Hall Standard Space Report. Technical report, Portland State University Facilities and Planning, Portland, OR, 2010.

[30] PSU. Final Campus-Wide Steam and Chilled Water Master Plan Portland State University. 2008.

[31] ASHRAE. 62.1: Ventilation for Acceptable Indoor Air Quality. Technical report, American Society of Heating, Refrigeration, and Air Conditioning Engineers, Atlanta, GA, 2010.

[32] Bo Xu and Stacey Lin. Technical Analysis of Neuberger Hall. Technical report, Interface Engineering Inc., Portland, OR, 2010.

[33] Suranjana Saha, Shrinivas Moorthi, Hua-Lu Pan, Xingren Wu, Jiande Wang, Sudhir Nadiga, Patrick Tripp, Robert Kistler, John Woollen, David Behringer, Haixia Liu, Diane Stokes, Robert Grumbine, George Gayno, Jun Wang, Yu-Tai Hou, Hui-Ya Chuang, Hann-Ming H Juang, Joe Sela, Mark Iredell, Russ Treadon, Daryl Kleist, Paul Van Delst, Dennis Keyser, John Derber, Michael Ek, Jesse Meng, Helin Wei, Rongqian Yang, Stephen Lord, Huug Van Den Dool, Arun Kumar, Wanqiu Wang, Craig Long, Muthuvel Chelliah, Yan Xue, Boyin Huang, Jae-Kyung Schemm, Wesley Ebisuzaki, Roger Lin, Pingping Xie, Mingyue Chen, Shuntai Zhou, Wayne Higgins, Cheng-Zhi Zou, Quanhua Liu, Yong Chen, Yong Han, Lidia Cucurull, Richard W Reynolds, Glenn Rutledge, and Mitch Goldberg. The NCEP Climate Forecast System Reanalysis. *Bulletin of the American Meteorological Society*, 91(8):1015–1057, 2010.

[34] Andrea Saltelli. Sensitivity Analysis for Importance Assessment. *Risk Analysis*, 22(3):579–590, 2002.

[35] Brian S Everitt. A Monte Carlo Investigation of the Robustness of Hotelling ' s One- and Two-Sample T2 Tests. 74(365):48–51, 1979.

[36] T Reddy and D Claridge. Uncertainty of "measured" energy savings from statistical baseline models. *HVACR Research*, 6(1), 2000.

Appendix A

Building Inputs

Table A.1: Simplified Internal Gains

	Quantity	Energy/Per	Fraction	Total [W]
People	1500	70	0.5	52500
Lighting	200000	1.2	0.5	120000
Plug Loads	200000	1.2	0.5	120000

Table A.2: Opaque Exterior Elements

Type	Qty.	Perimiter (m)	Area	U-Value	Azimuth
Alum Panels	96	501.3	163.5	0.29	90
Alum Panels	97	613.4	231.8	0.29	180
Alum Panels	59	360.7	133.2	0.29	0
Brick Siding	2	73.2	201.9	0.29	90
Brick Siding	1	36.9	115.8	0.29	180
Brick Siding	1	36.9	115.8	0.29	0
Other Panels	20	101.2	32.0	0.29	180
Other Panels	20	101.2	32.0	0.29	0
Pillars	2	73.5	14.3	0.29	180
Pillars	2	73.5	14.3	0.29	0
Pillars	9	330.8	64.1	0.29	90
Roof	1	243.8	2405.3	0.29	0
Veneer Concrete	1	40.5	919.8	0.29	270
Veneer Concrete	1	37.5	102.7	0.29	180
Veneer Concrete	1	37.5	102.7	0.29	0

Table A.3: Exterior Glazing Elements

Type	Qty	Perimeter	Area	U-Value	SHGC	Azimuth
1" Glass	70	468.2	189.9	5.7	0.75	270
1" Glass	36	278.7	97.3	5.7	0.75	180
1" Glass	36	278.7	97.3	5.7	0.75	0
1/4" Glass	34	254.3	118.8	5.7	0.75	270
1/4" Glass	24	180.0	83.4	5.7	0.75	180
1/4" Glass	24	180.0	83.4	5.7	0.75	0
Alum Windows	96	1683.4	295.0	5.7	0.75	90
Alum Windows	64	1196.5	192.6	5.7	0.75	180
Alum Windows	60	1052.1	182.0	5.7	0.75	0
Glass Door	3	23.4	11.4	5.7	0.75	180
Glass Door	3	23.4	11.4	5.7	0.75	0
Glass Door	2	19.9	12.3	5.7	0.75	270
Skylights	4	97.3	305.2	5.7	0.75	0

Table A.4: ASHRAE 62.1 Ventilation Calculation

Type	Floor	Area	cfm/ft^2	People	cfm/person	m^3/s
Office	Basement	11072	0.06	55	5	0.44
Lecture	Basement	10235	0.06	665	7.5	2.64
Circulation	Basement	8157	0.06	0	0	0.23
Mechanical	Basement	1432	0.12	0	0	0.08
Office	First	28465	0.06	142	5	1.14
Circulation	First	4812	0.06	0	0	0.14
Mechanical	First	153	0.12	0	0	0.01
Office	Second	2710.7	0.06	14	5	0.11
Lecture	Second	24396.3	0.06	1586	7.5	6.30
Circulation	Second	5499	0.06	0	0	0.16
Office	2nd Mezz	3803	0.06	19	5	0.15
Circulation	2nd Mezz	1622	0.06	0	0	0.05
Office	Third	4415.4	0.06	22	5	0.18
Lecture	Third	17661.6	0.06	1148	7.5	4.56
Circulation	Third	10405	0.06	0	0	0.29
Mechanical	Third	163	0.12	0	0	0.01
Office	3rd Mezz	2754	0.06	14	5	0.11
Circulation	3rd Mezz	1889	0.06	0	0	0.05
Office	Fourth	10059	0.06	50	5	0.40
Lecture	Fourth	4311	0.06	280	7.5	1.11
Circulation	Fourth	8855	0.06	0	0	0.25
Office	4th Mezz	2802	0.06	14	5	0.11
Circulation	4th Mezz	1967	0.06	0	0	0.06
Office	Fifth	1599	0.06	8	5	0.06

Table A.5: Complete List of Variables for Latin Hypercube Sampling and the Morris Method Sensitivity Analysis.

Index	Variables	Min	Max	Ref
1	Outside Air Flow Rate (Building)	11.879	25.521	18.7
2	Outside Air Flow Rate (Lab)	0.826	1.774	1.3
3	Fan Power	75834	162916	119375
4	Pump Power	8356	15519	11938
5	Occupant Internal Gains (Building)	24813	53307	39060
6	Occupant Internal Gains (Lab)	1654	3554	2604
7	Lab Electricity Usage	8715	18723	13719
8	Thermal Bridge (Building)	1.000	1.500	1
9	Thermal Bridge (Lab)	1.000	1.500	1
10	Heat Exchanger Efficiency	0.83	0.87	0.85
11	Monthly Schedule (Building)	0.335	0.719	0.527
12	Monthly Schedule (Lab)	0.463	0.591	0.527
13	SHGC	0.729	0.931	0.830
14	Lighting Power Density	1	1.5	1.000
15	Plug Load Power Density	1	1.5	1.000
16	Lighting/Plug Load Schedule	0.254	0.546	0.400
17	North Shading	0.8	1	0.800
18	East Shading	0.703	0.897	0.800
19	South Shading	0.703	0.897	0.800
20	West Shading	0.703	0.897	0.800
21	Internal Temp H	18.8	21.2	20.0
22	Internal Temp C	21.9	24.7	23.3
23	External Convection Coef.	20.4	29.6	25.0

Appendix B

ISO 13790 MATLAB Code

MATLAB Code B.1: ISO 13790 Code

```
function [Q_h_interm,Q_c_interm]=iso_13790(H_g,H_tr,H_ve,time,C_m,...
    theta_h,theta_c,f_H,f_C,a0,t0,Q_gn,theta_e)

% ****************************************************************
% This function calculates the monthly heating and cooling loads
% using the monthly heat balance method specified in ISO 13790.
% ****************************************************************
%
% Inputs:
%     H_g          : Ground Heat Transfer Coefficient [W/K]
%     H_tr         : Transmission Heat Transfer Coefficient [W/K]
%     H_ve         : Ventilation Heat Transfer Coefficient [W/K]
%     C_m          : Building Total Heat Capacity []
%     time         : Monthly Timestep [s]
%     theta_h      : Heating Setpoint Temperature [C]
%     theta_c      : Cooling Setpoint Temperature [C]
%     theta_e      : Monthly Average External Temperature [C]
%     f_H          : Fraction of Month in Heating Mode
%     f_C          : Fraction of Month in Cooling Mode
%     a0           : Dimensionless Time Constant
%     t0           : Dimesnionless Time Constant
%     Q_gn         : Monthly Total Solar and Internal Gains [J]
%
% Outputs:
%     Q_h_interm   : Monthly Heating Energy Requirements [J]
%     Q_c_interm   : Monthly Cooling Energy Requirements [J]
%
% Subfunctions:
%     util_factor : Monthly Utilization Factor Calculation
%
for i = 1:12  % run through each month (Jan to Dec)
```

```
% ------------------------------------------------------------------
% Transmission/Ventilation
    Q_tr_g = H_g * time(i) * 2; % 2 degree temp difference
    % Transmission/Ventilation Heat Transfer (Continous)
    Q_tr_h = (theta_h - theta_e(i))*H_tr*time(i) + Q_tr_g; % [J]
    Q_tr_c = (theta_c - theta_e(i))*H_tr*time(i) + Q_tr_g; % [J]
    Q_ve_h = (theta_h - theta_e(i))*H_ve*time(i);   % [J]
    Q_ve_c = (theta_c - theta_e(i))*H_ve*time(i);   % [J]
% ------------------------------------------------------------------
% Heat Balance Ratio for Heating
    Q_h_i = Q_ve_h + Q_tr_h;  % Heating for month i
    gamma_h = Q_gn(i)/Q_h_i;  % Heat Balance for heating for month i
% Heat Balance Ratio for Cooling
    Q_c_i = Q_ve_c + Q_tr_c;  % Cooling for month i
    gamma_c = Q_gn(i)/Q_c_i;  % Heat Balance in cooling for month i
% ------------------------------------------------------------------
% 12.2.1 - Utilization Factor
    [eta_h,eta_c,a_h,a_c]=CalUtilizationFactor(C_m,H_tr,...
        H_ve,a0,t0,gamma_h,gamma_c,f_H(i),f_C(i));
% ------------------------------------------------------------------
% Continous Heating/Cooling Energy
    Q_h_cont = Q_h_i - eta_h*Q_gn(i); % [J] Heating
    Q_c_cont = Q_gn(i) - eta_c*Q_c_i; % [J] Cooling
% ------------------------------------------------------------------
% 13.2.2 - Intermitent Heating/Cooling
    Q_h_interm(i) = Q_h_cont*a_h;
    Q_c_interm(i) = Q_c_cont*a_c;
% ------------------------------------------------------------------
end
```

MATLAB Code B.2: Heat Transmission Calculation

```
function [H_tr,H_g,H_ve] = tr_calc(BldgEnv,Bldg,BldgEnvG,...
                                   Vdot,EnvWind)
% ****************************************************************
% This function calculates the transmission and ventilation heat
% transfer coefficients as outlined in ISO 13790.
% ****************************************************************
%
% Inputs:
%      Bldg: Complete Building Envelope Areas and U-Values
%   BldgEnv: Opaque Building Element Areas and U-Values
%  BldgEnvG: Ground Building Element Area
%      Vdot: Ventilation Air Flow Rate Required
%   EnvWind: Glazing Building Element Areas and U-Values
%
% Outputs:
```

```
%     H_tr: Transmission Heat Transfer Coefficient [W/K]
%     H_ve: Ventilation Heat Transfer Coefficient [W/K]
%      H_g: Ground Heat Transfer Coefficient [W/K]
%

mArea = BldgEnv(:,2);
mUVal = BldgEnv(:,3);
He = sum(mArea.*mUVal);  % W/K

wArea = Bldg(EnvWind,2);
wUVal = Bldg(EnvWind,3);
Hw = sum(wArea.*wUVal);  % W/K

H_tr = He + Hw;  % W/K

% Ventilation
H_ve = 1200*Vdot;  %[W/K]

% Transmission to Ground
gArea = BldgEnvG(:,2);
gUVal = BldgEnvG(:,3);
H_g = sum(gArea.*gUVal); % W/K
```

MATLAB Code B.3: Utilization Factor Calculation

```
function [eta_h,eta_c,a_h_red,a_c_red]=util_factor(C_m,H_tr,H_ve,...
    a_0,tau_0,gamma_h,gamma_c,f_H,f_C)

% ****************************************************************
% This function calculates the monthly utilization factors
% as specified in ISO 13790.
% ****************************************************************
%
% Inputs:
%      C_m: Building Total Heat Capacity []
%     H_tr: Transmission Heat Transfer Coefficient [W/K]
%     H_ve: Ventilation Heat Transfer Coefficient [W/K]
%      a_0: Dimensionless Time Constant
%    tau_0: Dimensionless Time Constant
%  gamma_h: Heat Balance Ratio in Heating Mode
%  gamma_c: Heat Balance Ratio in Cooling Mode
%      f_H: Fraction of Month in Heating Mode
%      f_C: Fraction of Month in Cooling Mode
%
% Outputs:
%    eta_h: Gain Utilization Factor
%    eta_c: Loss Utilization Factor
```

```
%  a_h_red: Intermitent Heating Reduction Factor
%  a_c_red: Intermitent Cooling Reduction Factor

% Building Inertia
tau = (C_m/3600)/(H_tr + H_ve);
a_param = a_0 + tau/tau_0;

%Gain and Loss Utilization Factor Calculation
if gamma_h<0
    eta_h = 1/gamma_h;
elseif gamma_h>0
    if gamma_h==1
        eta_h = a_param/(a_param + 1);
    else
        eta_h = (1-gamma_h^(a_param))/(1-gamma_h^(a_param + 1));
    end
else
    eta_h = 0;
end
if gamma_c<0
    eta_c = 1;
elseif gamma_c>0
    if gamma_c==1
        eta_c = a_param/(a_param + 1);
    else
        eta_c = (1-gamma_c^-(a_param))/(1-gamma_c^-(a_param + 1));
    end
else
    eta_c = 0;
end

% Heating and Cooling Reduction Factor Calculation
a_h_red = 1 - 3*(tau_0/tau)*gamma_h*(1-f_H);
if a_h_red > 1
    a_h_red = 1;
elseif a_h_red <f_H
    a_h_red = f_H;
end
a_c_red = 1 - 3*(tau_0/tau)*cBalance*(1-f_C);
if a_c_red > 1
    a_c_red = 1;
elseif a_c_red <Cfract
    a_c_red = Cfract;
end
```

Appendix C

Internal Gains and Solar Calculations

MATLAB Code C.1: Total monthly heat gain calculation. Comprised of solar and internal gains.

```
function Q_gn = gains(bldg_opaq,bldg_glaz,bldg_env,hoa,...
    lat,lon,weather_data,theta_e,theta_sky,time)

% ****************************************************************
% This function calculates the total internal and solar gains
% as specified in ISO 13790.
% ****************************************************************
%
% Inputs:
%             Bldg: Complete Building Envelope Data
%          BldgEnv: Opaque Element Areas and U-Values
%          EnvWind: Glazed Element Areas and U-Values
%              hoa: External Film Heat Transfer Coefficient
%              lat: Latitude of Location
%              lon: Longitude of Location
%     weather_data: Complete Hourly Weather Data
%          theta_e: Fraction of Month in Heating Mode
%        theta_sky: Fraction of Month in Cooling Mode
%
% Outputs:
%             Q_gn: Gain Utilization Factor
%
% Subfunction:
%       solar_glaz: Calculate Solar Gains from Glazed Elements
%       solar_opaq: Calculate Solar Gains from Opaque Elements
%        rad_solar: Calculate Radiated Solar Losses


% Estimate shading reduction factor
ext_shade = ones(size(bldg_glaz,1),1);

for i = 1:12
```

```
    % Solar Gains for Conditioned Space
    Phi_glaz(i) = solar_glaz(bldg_glaz,ext_shade,i,lat,...
                             lon,weather_data); %[Wh]
    Phi_opaq(i) = solar_opaq(bldg_opaq,i,lat,lon,...
                             weather_data,hoa); %[Wh]
    Phi_sol(i) = (Phi_glaz(i) + Phi_opaq(i))*3600; %[J]
    Phi_r(i) = rad_solar(bldg_env,hoa,theta_sky(i),...
                         i,theta_e(i))*time(i); %[J]
end

Q_sol_gn = (Phi_sol - Phi_r); % [J]

Q_gn = Q_sol_gn;
```

MATLAB Code C.2: Solar calculation for opaque elements.

```
function Q_sol_opaq = solar_opaq(bldg_opaq,month,lat,...
                                 lon,weather_data,hoa)

% ****************************************************************
% Subfunction to estimate the total solar gain
% through exterior opaque elements.
% ****************************************************************
%
% Inputs:
%     bldg_opaq   : Matrix containing area, SHGC, thermal
%                   absorptance, and thermal transmission
%                   of opaque elements.
%     month       : Month of the year.
%     lat         : Latitude of location.
%     lon         : Longitude of location.
%     weater_data : Matrix of hourly weather data including direct
%                   and diffuse solar radiation.
%     hoa         : Thermal resistance of thermal film on
%                   exterior surface.
%
% Outputs:
%     Q_sol_opaq  : Monthly solar gain through glazing elements [Wh]
%

Q_sol_opaq = 0; % Initialize

Fsh = 1; % Shading reduction factor

for i = 1:size(bldg_opaq,1)
    alpha_f = bldg_opaq(i,6);
    beta_f = bldg_opaq(i,7);
```

```
    SolAbs = bldg_opaq(i,5);
    Aopaq = bldg_opaq(i,2);
    U_val = bldg_opaq(i,3);
    [I_beam, I_diff] = month_solar(lat,lon,weather_data,month,...
                                   alpha_f,beta_f); %[Wh/m2]
    Asol = SolAbs/hoa*U_val*Aopaq;
    SolarOpaq = (I_beam + I_diff)*Asol*Fsh;

    Q_sol_opaq = Q_sol_opaq +  SolarOpaq;  % [Wh]
end
```

MATLAB Code C.3: Solar calculation for glazed elements

```
function Q_sol_glaz = solar_glaz(bldg_glaz,bldg_shade,month,...
                                      lat,lon,weather_data)

% ******************************************
% Subfunction to estimate the total solar gains
% through glazing envelope elements
% ******************************************
%
% Inputs:
%      bldg_glaz     : Matrix containing area, SHGC, thermal
%                      absorptance, and thermal transmission
%                      of building glazing.
%      bldg_shade    : External shading factor
%      lat           : Latitude of location
%      lon           : Longitude of location
%      weather_data  : Matrix of hourly weather data including
%                      direct and diffuse solar radiation.
% Outputs:
%      Q_sol_glaz    : Monthly solar gain from glazing
%                      elements [Wh]

Q_sol_glaz = 0; % Initialize

Ff = 0.0;  % Frame factor

for i = 1:size(bldg_glaz,1)
    alpha_f = bldg_glaz(i,6);
    beta_f = bldg_glaz(i,7);
    SHGC = bldg_glaz(i,5);
    Aw = bldg_glaz(i,2);
    [I_beam, I_diff] = month_solar(lat,lon,weather_data,month,...
                                   alpha_f,beta_f);   %[Wh/m2]
    A_sol = SHGC * (1-Ff)*Aw;
    Q_sol_beam = bldg_shade(i)*A_sol*I_beam;
```

```
    Q_sol_diff = A_sol*I_diff;

    Q_sol_glaz = Q_sol_glaz+(Q_sol_beam + Q_sol_diff);% [Wh]
end
```

MATLAB Code C.4: Monthly calculation for total solar gains.

```
function [I_beam, I_diff] = month_solar(lat,lon,weather_data,...
                                                    month,alpha_f,beta_f)

% ***************************************************************
% This is a subfunction to calculate the monthly incident solar
% radiation on any tilted surface.
% ***************************************************************
% Author    : Huafen Hu
% Edited By : Nicolas Johnson
%
% Inputs:
%       Lat: latitude of the location [deg]
%       Lon: longtitude of the location [deg]
%       alpha_f: surface zimuth measured from due south clockwise, [deg]
%       beta_f: the tilt angle [deg]
% Outputs:
%       mIb: monthly beam radiation on the tilted surface, [Wh/m^2]
%       mId: monthly diffuse radiation on the tilted surface, [Wh/m^2]

I_beam = 0;
I_diff = 0;

fdmonth = [1 32 60 91 121 152 182 213 244 274 305 335];

ydate_s = fdmonth(month);
if month==12
    ydate_e = 365;
else
    ydate_e = fdmonth(month+1)-1;
end

for j = ydate_s:ydate_e
    Ydate = j;
    for k = 1:24
        LST = k-0.5;
        nrow = (Ydate -1)* 24 + floor(LST) + 1;
        [Idbeta,Idfbeta]=solar(weather_data,lat,lon,Ydate,LST,nrow,...
                                                    alpha_f,beta_f);
        I_beam= I_beam + Idbeta;
        I_diff = I_diff + Idfbeta;
```

```
    end
end
```

MATLAB Code C.5: Calculation for direct and diffuse solar radiation on each element.

```
function [Idbeta,Idfbeta,i_beta]=solar(weather_data,Lat,Lon,Ydate,...
                                       LST,nrow,alpha_f,beta_f)

% ********************************************************************
% This is a sub-function for calculating solar gains.
% The total solar gain comes from three parts: direct radiation, sky
% diffuse radiation and ground reflection.
% ********************************************************************
%
% Author: Huafen Hu
%
% Inputs:
%     weather_data : Formatted weather data w/ solar beam and diffuse
%                    radiation.
%     lat          : Lattitude of location.
%     lon          : Longitude of location.
%     Ydate        : Day of the year.
%     LST          : Hour of the day.
%     nrow         : Row of weather data to access.
%     alpha_f      : Surface azimuth measure from due south clockwise.
%     beta_f       : Tilt angle of surface.
%
% Outputs:
%     Idbeta       : Total direct/beam radiation on the surface.
%     Idfbeta      : Total diffuse solar radiation on the surface.
%     i_beta       : The angle between the incident beam and
%                    the surface's normal vector.
%

r_g = 0.2; % ground reflectivity - assign the overcast sky

%---------------
%Albedo of some typical natural surfaces
% fresh snow cover  - (0.75-0.95)
% old snow cover - (0.4-0.7)
% densely built-up areas - (0.15-0.25)
% high dense grass - (0.18-0.20)
% Lawn: high sun, clear sky  - (0.23)
% Lawn: high sun, partly cloudy  - (0.23)
% Lawn: low sun, clear sky  - (0.25)
```

```
% Lawn: overcast sky  - (0.23)
% Dead leaves - (0.30)

Iglobal_h = weather_data(nrow,11);   % Load global horizontal radiation
                                     % from weather data
Ifh = weather_data(nrow,13); % Load diffuse horizontal radiation
                             % from weather data
Idh = Iglobal_h - Ifh;

if Iglobal_h == 0
    Idbeta = 0;
    Idfbeta = 0;
    i_beta =0;
    return;
end

plocal = weather_data(nrow,7); %atmospheric station pressure(Pa)

%-----------------------------------------------------------------%
% This part covers direct radiation

decli = 23.45*sin((280.1+(0.9863*Ydate))*pi/180); % Solar declination,
                                                  % unit: degree

% Apparent solar time, calculated according to
% ASHRAE fundamentals 07/31/2006

LSM = 120; % Local standard time meridian, CST - 90, EST - 75,
           % MST - 105, PST - 120
ET = 9.87*sin((1.978*Ydate-160.22)*pi/180)-7.53*cos((0.989*Ydate-...
     80.11)*pi/180)-1.5*sin((0.989*Ydate-80.11)*pi/180); % Equation
                                                         % of time
ts = LST + ET/60 + 4*(LSM - Lon)/60; % Apparent solar time


theta_h = 15 * (12 - ts); % The hour angle, degree
beta_s = asin(cos(Lat*pi/180)*cos(decli*pi/180)*cos(theta_h*pi/180)...
    +sin(Lat*pi/180)*sin(decli*pi/180)); % The solar altitude, radians

if sin(beta_s)<sin(3*pi/180)    % Sun is still below the horizon,
                                % assigning the tolerance to be
                                % 5 degrees
    Idbeta = 0;
    Idfbeta = 0;
    i_beta =0;
    return;
end
```

```
Isc = 1353; % the solar constant evaluated at the equinox, W/m^2

sin_alpha_s = cos(decli*pi/180)*sin(theta_h*pi/180)/cos(beta_s);
cos_alpha_s = (sin(beta_s)*sin(Lat*pi/180) - sin(decli*pi/180))...
              /cos(beta_s)/cos(Lat*pi/180);
alpha_s1 = asin(sin_alpha_s)*180/pi; % Solar azimuth
                                     % option 1 in degree
alpha_s2 = acos(cos_alpha_s)*180/pi; % Solar azimuth
                                     % option 2 in degree
Idn = Idh/sin(beta_s);

%% Do correction on solar azimuth  - 07/31/2006
if sin_alpha_s > 0.00000001
    if cos_alpha_s > 0.00000001
        alpha_s = - alpha_s1;
    else
        alpha_s = - alpha_s2;
    end
else
    if cos_alpha_s > 0.00000001
        alpha_s = - alpha_s1;
    else
        alpha_s = alpha_s2;
    end
end


w = abs(alpha_s - alpha_f); % The surface solar azimuth
i_beta = acos(sin(beta_s)*cos(beta_f*pi/180)+cos(beta_s)*...
         cos(w*pi/180)*sin(beta_f*pi/180));
% The angle between the incident beam and the surface's normal vector

if beta_f==0  %horizontal surfaces
    Idbeta = Idh;
    Idfbeta = Ifh;
    return;
elseif beta_f==180
    Idbeta = 0;
    Idfbeta = 0;
    return;
end

if cos(i_beta)<0.0000001 % the subject is behind the shadow
    Idbeta = 0;
else
    Idbeta = Idn * cos(i_beta) ;  % The direct intensity on
                                  % the inclined surface;
    if Idbeta < 0
```

```
        Idbeta = 0;
    end
end

%-------------------------------------------------------------------%
% The section deals with ground reflected component

Irv = 0.5*(1-cos(beta_f*pi/180))* Iglobal_h * r_g;
% The ground reflected total radiation incident on the
% nonvertical inclination

% calculate the air mass
% The air mass corresponding to the prevailing solar altitude
% and atmospheric pressure
m_d = sin(beta_s)+0.00176759*(beta_s*180/pi*((94.37515-beta_s...
                              *180/pi)^(-1.21563)));
m_unc = 1/m_d;
m = m_unc * plocal / 101325;

%-------------------------------------------------------------------%
% This part deals with sky diffuse component
z = (90 - beta_s*180/pi)*pi/180; % The zenith angle, in radiance
a0 = max(0, cos(i_beta));
a1 = max(cos(85*pi/180), cos(z));
% a0, a1: correct for the angle of incidence of the circumsolar
% radiation on the inclined and horizontal surface respectively

if Ifh==0
    Idfbeta = 0;
    return;
else
    epslon_m = (Ifh + Idn)/Ifh + (1.041*z^3);
    epslon_d = 1 + (1.041*z^3);
    epslon = epslon_m / epslon_d;
end

f = [
-0.0083 0.1299  0.3297  0.5682  0.8730  1.1326  1.0602  0.6777
0.5877  0.6826  0.4869  0.1875  -0.3920 -1.2367 -1.5999 -0.3273
-0.0621 -0.1514 -0.2211 -0.2951 -0.3616 -0.4118 -0.3589 -0.2504
-0.0596 -0.0189 0.0554  0.1089  0.2256  0.2878  0.2642  0.1561
0.0721  0.0660  -0.0640 -0.1519 -0.4620 -0.8230 -1.1272 -1.3765
-0.0220 -0.0289 -0.0261 -0.0140 0.0012  0.0559  0.1311  0.2506];

fcol = zeros(5,1);  % initialize the f column
if (epslon>=1.000)&&(epslon<1.065)
    fcol = f(:,1);
elseif (epslon>=1.065)&&(epslon<1.230)
```

```
    fcol = f(:,2);
elseif (epslon>=1.230)&&(epslon<1.500)
    fcol = f(:,3);
elseif (epslon>=1.500)&&(epslon<1.950)
    fcol = f(:,4);
elseif (epslon>=1.950)&&(epslon<2.800)
    fcol = f(:,5);
elseif (epslon>=2.800)&&(epslon<4.500)
    fcol = f(:,6);
elseif (epslon>=4.500)&&(epslon<6.200)
    fcol = f(:,7);
else
    fcol = f(:,8);
end

delta = m * Ifh / Isc;  % The sky's brightness ( reflect the
                        % opacity/thickness of the clouds)
F1 =max(0, (fcol(1)+fcol(2)*delta + z*fcol(3)));
% The circumsolar brightness coefficient
F2 = fcol(4) + fcol(5)*delta + z*fcol(6);
% The horizon brightness coefficient

Isbeta = Ifh * ((1-F1)*cos(0.5*beta_f*pi/180)*cos(0.5*beta_f*...
                pi/180) + F1*a0/a1 + F2*sin(beta_f*pi/180));
% The sky diffuse radiation incident on a surface of inclination

%-------------------------------------------------------------------
%-------------------------------------------------------------------
Idfbeta = Isbeta + Irv; % total diffuse radiation, W/m^2
```

MATLAB Code C.6: Radiated solar loss calculation.

```
function Phi_r = rad_solar(BldgEnvC,hoa,theta_sky,theta_e)

% ********************************************************************
% This is a function to estimate longwave radiation between
% exterior building surfaces and sky.
% ********************************************************************
%
% Author: Huafen Hu
%
% Inputs:
%      bldg_env   : Building exterior surface data
%      hoa        : Exterior surface convective heat
%                   transfer coefficient
%      theta_sky  : Sky temperature
%      theta_e    : Exterior temperature
```

```
%
% Outputs:
%       Phi_r: Total longwave solar radiation loss
%

Phi_r = 0; % Initialize
sigma_0 = 5.67e-8;   % [W/M2K4] Stephen Boltzman Constant
theta_ss = (theta_sky + theta_e)/2;

% Estimate the external radiative heat transfer coefficient
for j = 1:size(BldgEnvC,1)
    alpha_f = BldgEnvC(j,6);
    epslon_0 = BldgEnvC(j,8);
    h_r = 4*epslon_0*sigma_0*((theta_ss+273.15)^3);  % [W/M2K]
    U_val = BldgEnvC(j,3);                           % [W/M2K]
    area = BldgEnvC(j,2);
    phi_r_j = (1/hoa*U_val*area*h_r*(theta_e - Tsky))*...
              (1+cos(alpha_f/180*pi))/2;                % [W]

    Phi_r = Phi_r + phi_r_j;
end
```

MATLAB Code C.7: Sky temperature calculation.

```
function theta_sky = sky_temp(weather_data)

% *******************************************************
% This is a function to estimate sky temperature based
% on the algorithm set up in EnergyPlus, referred in
% Engineering reference PDF p.148
% *******************************************************
%
% Author: Huafen Hu & Nicolas Johnson
%
% Inputs:
%       weather_data : The TMY3 formatted weather file
%
% Outputs:
%       theta_sky    : The average monthly sky temperature, [degC]
%

theta_sky = zeros(12,1);

% constants
sigma0 = 5.67e-8; % Stefan-Boltzmann constant, [W/m2K4]
hIR = weather_data(:,10);
hTsky = (hIR/sigma0).^0.25 - 273.15;
```

```
% calculate the monthly average
fdmonth = [1 32 60 91 121 152 182 213 244 274 305 335];
for i=1:12
    ydate_s = fdmonth(i);
    if i==12
        ydate_e = 365;
    else
        ydate_e = fdmonth(i+1)-1;
    end

    row_s = (ydate_s - 1)*24 +1;
    row_e = (ydate_e - 1)*24 + 24;
    theta_sky(i) = mean(hTsky(row_s:row_e));
end
```

MATLAB Code C.8: Monthly average temperature calculation.

```
function [theta_sky,theta_e,t,WindS]=weather(weather_data)

theta_sky = sky_temp(weather_data);

% Ic, Tdb,Tsky

mWeather = weather_data(:,4);
Jan = mean(mWeather(1:744));
Feb = mean(mWeather(745:1416));
Mar = mean(mWeather(1417:2160));
Apr = mean(mWeather(2161:2880));
May = mean(mWeather(2881:3624));
Jun = mean(mWeather(3625:4344));
Jul = mean(mWeather(4345:5088));
Aug = mean(mWeather(5089:5832));
Sep = mean(mWeather(5833:6552));
Oct = mean(mWeather(6553:7296));
Nov = mean(mWeather(7296:8016));
Dec = mean(mWeather(8017:8760));

mWind = weather_data(:,15);
wJan = mean(mWind(1:744));
wFeb = mean(mWind(745:1416));
wMar = mean(mWind(1417:2160));
wApr = mean(mWind(2161:2880));
wMay = mean(mWind(2881:3624));
wJun = mean(mWind(3625:4344));
wJul = mean(mWind(4345:5088));
wAug = mean(mWind(5089:5832));
```

```
wSep = mean(mWind(5833:6552));
wOct = mean(mWind(6553:7296));
wNov = mean(mWind(7296:8016));
wDec = mean(mWind(8017:8760));

theta_e = [Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec];

WindS=[wJan;wFeb;wMar;wApr;wMay;wJun;wJul;wAug;wSep;wOct;wNov;wDec];

ndaysMonth=[31 28 31 30 31 30 31 31 30 31 30 31];
t = ndaysMonth'*24*3600;
```

Appendix D

Morris Method MATLAB Functions

MATLAB Code D.1: Morris Method generator script. This script initializes the variables and inserts them within the morris_experiment function. Once complete a variable matrix is created for simulation.

```
% ------------------------------------------------------------------
%  This script generates a file called MorrisExperiment.csv which
%  specifies the runs for a Method of Morris screening experiment.
%  To use this script, please do the following:

%  1.  REQUIRED FILES
%  Have the following files in your working directory in Matlab:
%  - generate_experiment.m (this script)
%  - morris_experiment.m

%  2.  EDIT CODE FOR YOUR FACTORS
%  Adapt the script for the number of factors that you are analyzing.
%   - Specify your factors the comments following Line 23.
%   - Specify lower bounds for your factors in Line 31.
%   - Specify upper bounds for your factors in Line 32.
%   - Specify the number of random observations in Line 40.
% ------------------------------------------------------------------

% ------------------------------------------------------------------
% SPECIFY FACTORS
% ------------------------------------------------------------------

% -----------------------------edit-below-----------------------------
% x = [Factor1,...,Factork]
% Factor1 ranges from a to b                              --> [a, b]
% ...
% Factork ranges from c to d                              --> [c, d]
% -----------------------------edit-above-----------------------------

% -----------------------------edit-below-----------------------------
```

```
xlb = xlsread('..\16.Variables\Uncertainties.xls','Morris','C2:C156')';
xub = xlsread('..\16.Variables\Uncertainties.xls','Morris','D2:D156')';
% ----------------------------edit-above----------------------------

% ------------------------------------------------------------------
% SPECIFY NUMBER OF RANDOM OBSERVATIONS
% ------------------------------------------------------------------

% ----------------------------edit-below----------------------------
r = 10; % the number of random observations
% ----------------------------edit-above----------------------------

k = length(xlb); % the number of factors
e = morris_experiment(k,r,xlb,xub);
csvwrite('..\16.Variables\MorrisExperiment.csv',e)
m_morr = csvread('..\16.Variables\MorrisExperiment.csv');
% ------------------------------------------------------------------
```

MATLAB Code D.2: Function within the Morris Method generator script. This function generates a matrix of variable sets between their upper and lower bounds.

```
function X = morris_experiment(k, r, xlb, xub, seed)
%
% ************************************************
% Function to generate the input data matrix for the
% Morris expiriment.
% ************************************************
%
% Author   : Huafen Hu
% Edits By : Nicolas Johnson
%
% Inputs:
%     k    : Number of input factors
%     p    : Grid_level (should be even)
%     r    : The number of effects that one wants to sample
%     lb   : Optional lower bound on the x values
%     ub   : Optional upper bound ont the x values
%     seed : Optional random number generator seed
%
% Output:
%     X    : Matrix of (n,k) for model
%

m = k+1; % number of experiments per batch
n = m*r; % total number of experiments
```

```
% Pick p to be something large so that it is unlikely that
% the same grid point will be sampled twice
p = 4; %r*10000;
delta = p/(2*(p-1));

% Check for lower and upper bounds
if nargin < 4
    xlb = zeros(1,k);
    xub = ones(1,k);
end

% Seed the random number generator
if nargin==5
    rand('state',seed);
else
    rand('state',sum(100*clock));
end

% Define sampling matrix of the form
% B = [0 0 0 0;
%      1 0 0 0;
%      1 1 0 0;
%      1 1 1 0;
%      1 1 1 1];
J = ones(m,k);
B = tril(J,-1);

X = zeros(n,k);
for i=1:r
    Dstar = diag(floor(rand(k,1)*2)*2-ones(k,1));
    xstar = floor(rand(1,k)*p/2)/(p-1);
    Btemp = ones(m,1)*xstar+delta/2*((2*B-J)*Dstar+J);
    Bstar = Btemp(:,randperm(k));
    X((i-1)*m+1:i*m,:) = ones(m,1)*xlb+ones(m,1)...
                         *(xub-xlb).*Bstar;
end

return;
```

MATLAB Code D.3: First step in analyzing the simulation results from the preceeding variable matrix. Elementary effect statistics are generated for the plots and further analysis.

```
% ------------------------------------------------------------------
%  This script generates Method of Morris plots to analyze the results
%  of a Method of Morris screening experiment.  To use this script,
```

```
%  please do the following:

%  1.  REQUIRED FILES
%  Have the following files in your working directory in Matlab:
%  - generate_plots.m (this script)
%  - morris_plot.m
%  - MorrisResults.csv (a .csv file containing your experiment and
%    results.
%  It is important that this file contain only numbers, so be sure
%  there are no headers in your .csv file.)

%  2.  EDIT CODE FOR YOUR RESPONSES
%  Adapt the script for the number of responses that you are
%  analyzing.
%   - Specify the number of responses in Line 26.
%   - Add or remove plot commands following Line 45.
%   - Add or remove entries in the stats command following Line 73.
% ------------------------------------------------------------------------

% ------------------------------------------------------------------------
% NUMBER OF RESPONSES
% ------------------------------------------------------------------------

% load Morris_Results
% Qkwh =[m_mQkwh' sum(m_mQkwh,1)'];
% Qsh = [m_mQh_demand' sum(m_mQh_demand,1)'];
% xlswrite('MorrisResults.csv',Qkwh,'MorrisResults','A1');
% xlswrite('MorrisResults.csv',Qsh,'MorrisResults','N1');
%

% ----------------------------edit-below-----------------------------
m = 30; % number of responses
%
% load morResult
%
% morResult = [qH_total' sum(qH_total)' qE_total' ...
%     sum(qE_total)' g_CV_RMSE' g_MBE' e_CV_RMSE' e_MBE'];
%
% load m_morr
%
% morResults = [m_morr morResult];
%
% csvwrite('MorrisResults.csv',morResults);

% ----------------------------edit-above-----------------------------

% load experiment and response data from MorrisResults.csv
E = xlsread('MorrisResults.csv');
```

```
k = size(E,2) - m;
e = E(:,1:k);
M = E(:,(k+1):(k+m));

% ------------------------------------------------------------------
% PLOT COMMANDS
% ------------------------------------------------------------------
% The following lines contain plot commands to generate the morris
% plots. There is one plot command for each response.  Each command
% consists of three lines of code.  Add or remove plot commands as
% needed.  Be sure to index the effective mean and standard deviation
% variable names as well as the column in the M vector for each
% additional response.  Also, replace the generic "Reponse #" with a
% more descriptive name.


% ----------------------------edit-below-----------------------------
figure, [eff_mean_m1, eff_std_m1] = morris_plot(e,M(:,26));
title('Method of Morris - annual space heating')

figure, [eff_mean_m2, eff_std_m2] = morris_plot(e,M(:,13));
title('Method of Morris - annual kwh')


figure, [eff_mean_m3, eff_std_m3] = morris_plot(e,M(:,3));
title('Method of Morris - Response 3')

figure, [eff_mean_m4, eff_std_m4] = morris_plot(e,M(:,4));
title('Method of Morris - Response 4')

figure, [eff_mean_m5, eff_std_m5] = morris_plot(e,M(:,5));
title('Method of Morris - Response 5')
% ----------------------------edit-above-----------------------------

% ---------------------------Results postprocess--------------------
MorrisSum = zeros(k,m*3);
for ii = 1:m
    [eff_mean, abs_eff_mean, eff_std] = morris_cal(e, M(:,ii));
    MorrisSum(:,(ii-1)*3+1)= abs_eff_mean;
    MorrisSum(:,(ii-1)*3+2)= eff_mean;
    MorrisSum(:,(ii-1)*3+3)= eff_std;
end

% rank them here

save morSum MorrisSum

xlswrite('MorrisSum.xls',MorrisSum,1,'A1')
```

```
% ------------------------------------------------------------------
% STATS
% ------------------------------------------------------------------
% The effective means and standard deviations for each factor and each
% response are stored in the stats matrix below.  This is helpful for
% determining which dot on the plot corresponds to which factor,
% because the labels often overlap or are not sufficiently close to
% the dot.  Notes for interpreting the stats matrix are included below.
% Edit this command to include the effective means and standard
% deviations for each response that you are studying.

% ----------------------------edit-below----------------------------
% stats = [eff_mean_m1', eff_std_m1', ...
%     eff_mean_m2', eff_std_m2',...
%     eff_mean_m3', eff_std_m3',...
%     eff_mean_m4', eff_std_m4',...
%     eff_mean_m5', eff_std_m5'];
% ----------------------------edit-above----------------------------

% INTERPRETING THE STATS MATRIX
% The effective means and standard deviations are organized as follows
% in the stats matrix
% - each row corresponds to a factor
% - each pair of columns corresponds to the mean and standard
%   deviation of a response (i.e., there are 2*m columns)
%
% For example, elements (4,5) and (4,6) correspond to the mean and
% standard deviation of the fourth factor with respect to the
% third response.
% ------------------------------------------------------------------
```

MATLAB Code D.4: Final step in the Morris Method sensitivity analysis ranking variables according to their significance according to the results received from the experiment generated.

```
% ****************************************************
% This script is written to process Morris results.
% ****************************************************

clear all
resMorris = xlsread('MorrisSum.xls');
Mrank = zeros(length(resMorris),30);
Mstar = zeros(length(resMorris),30);

ind1 = [1:length(resMorris)]';
```

```
nw = 1;
for i=1:3:size(resMorris,2)
    comb = [ind1 -resMorris(:,i)];
    zero_row = find(comb(:,2)==0);
    comb(zero_row,:)=[];
    comb2 = sortrows(comb,2);
    Mrank(1:length(comb2(:,1)),nw) = comb2(:,1);
    Mstar(1:length(comb2(:,1)),nw) = -comb2(:,2);
    nw = nw+1;
end
xlswrite('MorrisSum.xls',Mrank,'Mrank','B2');
xlswrite('MorrisSum.xls',Mstar,'Mstar','B2');

morScreen = [Mrank;Mstar];

morSetup = [Mrank(:,27) Mstar(:,27) Mrank(:,28) Mstar(:,28) ...
    Mrank(:,29) Mstar(:,29) Mrank(:,30) Mstar(:,30)];

morMonth = [Mrank(:,1) Mstar(:,1) Mrank(:,2) Mstar(:,2) Mrank(:,3)...
    Mstar(:,3) Mrank(:,4) Mstar(:,4) Mrank(:,5) Mstar(:,5)...
    Mrank(:,6) Mstar(:,6) Mrank(:,7) Mstar(:,7) Mrank(:,8)...
    Mstar(:,8) Mrank(:,9) Mstar(:,9) Mrank(:,10) Mstar(:,10)...
    Mrank(:,11) Mstar(:,11) Mrank(:,12) Mstar(:,12)];

save morScreen morScreen morMonth Mrank Mstar
```

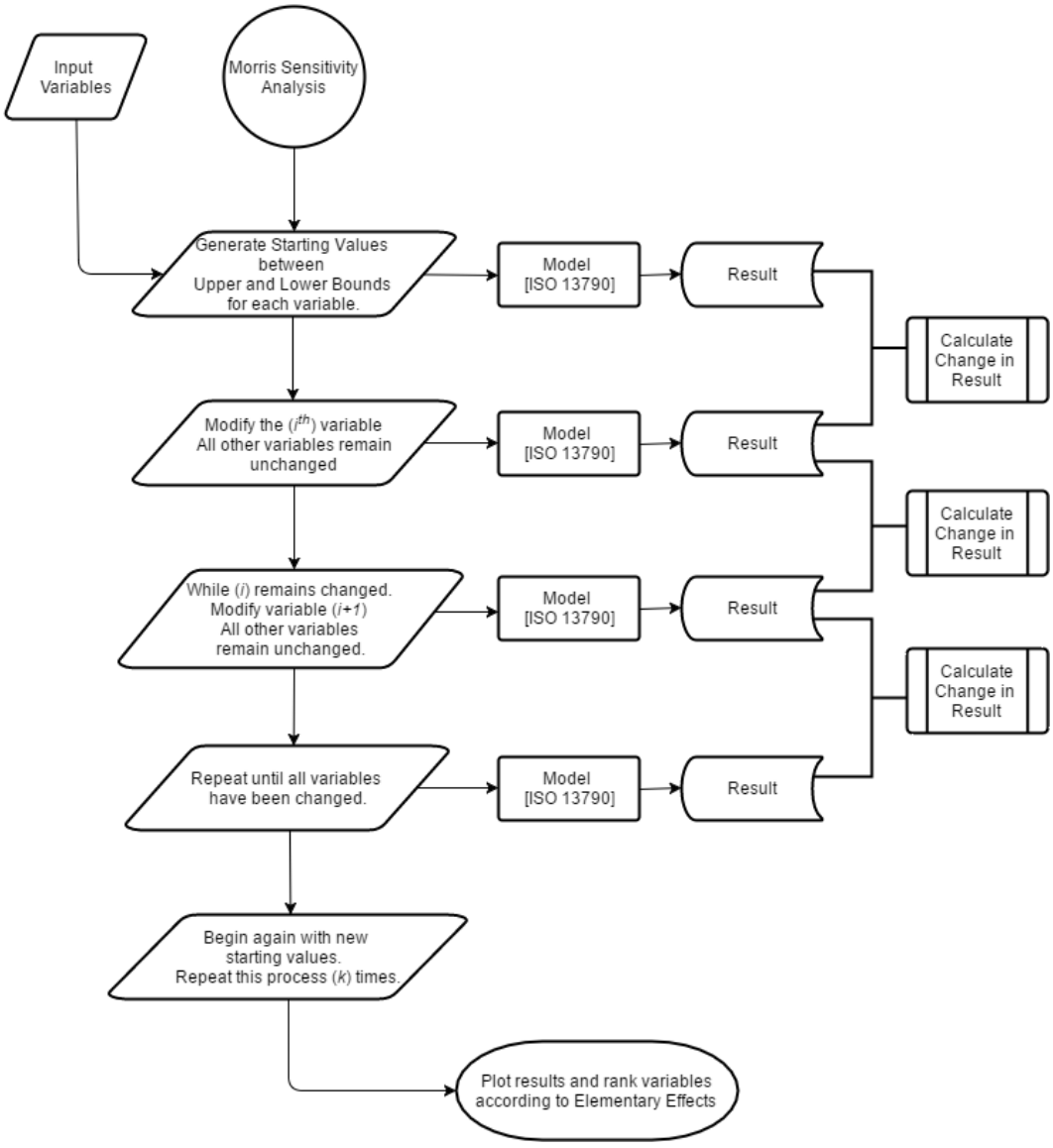

Figure 4.1: Flowchart of morris method sensitivity analysis.

Appendix E

Latin Hypercube Sampling MATLAB Functions

MATLAB Code E.1: Latin Hypercube Sampling main function to take variables and their reference values and return n random samples based on their probability density functions.

```
% *******************************************************
% Latin Hypercube Sampling main data
% input and output script.
% *******************************************************
%
% Authors: Nicolas Johnson and Huafen Hu
%
% Inputs:
%       pdfPar  : Data Structure Containing Variable Info
%       nsample : Number of Samples
%
% Output:
%       m_lhs   : Random Sample Matrix (nsample,nvar)
%
% Subfunction:
% lhs_normunitri: Subfunction to create random values based on
%                 the pdf assigned to each variable.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% DomID MUST be EQUAL to number of variables being sampled!!! %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

nsample = 5000;  % Edit to change number of samples
pdfPar = xlsread('...\PathToExcelData.xlsx','Sheet','CellRange');
refPar = pdfPar(:,1);  % Reference Parameter (Initial Value)
pdfID = pdfPar(:,3);   % PDF Identifier (Normal/Uniform)

domID = sort(1:size(pdfPar,1)); % Index of Each Variable
pdfdom = pdfID(domID);
domPar = pdfPar(domID,:);
```

```
normID = find(pdfdom==1);
unifID = find(pdfdom==2);

xmean = domPar(normID,1);
xsd = domPar(normID,2);

xmin = domPar(unifID,1);
xmax = domPar(unifID,2);

m_lhs_std = lhs_normunitri(xmean,xsd,xmin,xmax,nsample);

% Restore the Matrix
newID = [normID;unifID];
comb = [newID m_lhs_std'];
comb = sortrows(comb,1);
m_lhs = comb(:,2:size(comb,2))';
```

MATLAB Code E.2: Modification to the main LHS function to re-assign the reference values to insignificant variables as determined by the sensitivity analysis.

```
% Assign constant values for nonsignificant parameters
m_lhs = ones(nsample,k);
for j=1:k
    m_lhs(:,j) = refPar(j)*m_lhs(:,j);
end
for ii = 1:length(domID)
    dID = domID(ii);
    m_lhs(:,dID) = m_lhs_sh(:,ii);
end
```

MATLAB Code E.3: Sub function to calculate the uniform and normal distributions for each variable for the number of samples required.

```
function s=lhs_normunitri(xmean,xsd,xmin,xmax,nsample)

% ******************************************************
% Latin Hypercube Sampling function from a combination
% of uniform and normal distribution functions.
% ******************************************************
%
% Author        : Huafen Hu
% Edits by      : Nicolas Johnson
%
% Inputs:
```

```
%       xmean   : Mean of Data (1,nvar)
%       xsd     : Standard Deviation of Data (1,nvar)
%       nsample : Number of Samples
%       xmin    : Min of Data (1,nvar)
%       xmax    : Max of Data (1,nvar)
%
% Output:
%       s       : Random Sample (nsample,nvar)
%
nvar1 = length(xmean); % normal
nvar2 = length(xmin);  % uniform

ran = rand(nsample,nvar1+nvar2);
s = zeros(nsample,nvar1+nvar2);

% Stein method for normal distributions
for j=1:nvar1+nvar2
    idx = randperm(nsample);
    P=(idx'-ran(:,j))/nsample;       % Probability of the cdf
    if j<nvar1+0.5
        s(:,j) = xmean(j) + ltqnorm(P).* xsd(j);
    else
        s(:,j) = xmin(j-nvar1) + P.* (xmax(j-nvar1)-xmin(j-nvar1));
    end
end
```

MATLAB Code E.4: Sub function to provide a better approximation of the normal distribution for probabilities close to zero. Written by Peter J. Acklam.

```
function z = ltqnorm(p)
%LTQNORM Lower tail quantile for standard normal distribution.
%
%   Z = LTQNORM(P) returns the lower tail quantile for the standard
%   normal distribution function.  I.e., it returns the Z satisfying
%   Pr{X < Z} = P, where X has a standard normal distribution.
%
%   LTQNORM(P) is the same as SQRT(2) * ERFINV(2*P-1), but the
%   former returns a more accurate value when P is close to zero.

%   The algorithm uses a minimax approximation by rational functions
%   and the result has a relative error less than 1.15e-9.  A last
%   refinement by Halley's rational method is applied to achieve full
%   machine precision.

%   Author:      Peter J. Acklam
%   Time-stamp:  2003-04-23 08:26:51 +0200
```

```
%   E-mail:      pjacklam@online.no
%   URL:         http://home.online.no/~pjacklam

% Coefficients in rational approximations.
a = [ -3.969683028665376e+01  2.209460984245205e+02 ...
     -2.759285104469687e+02  1.383577518672690e+02 ...
     -3.066479806614716e+01  2.506628277459239e+00 ];
b = [ -5.447609879822406e+01  1.615858368580409e+02 ...
     -1.556989798598866e+02  6.680131188771972e+01 ...
     -1.328068155288572e+01 ];
c = [ -7.784894002430293e-03 -3.223964580411365e-01 ...
     -2.400758277161838e+00 -2.549732539343734e+00 ...
      4.374664141464968e+00  2.938163982698783e+00 ];
d = [  7.784695709041462e-03  3.224671290700398e-01 ...
       2.445134137142996e+00  3.754408661907416e+00 ];

% Define break-points.
plow  = 0.02425;
phigh = 1 - plow;

% Initialize output array.
z = zeros(size(p));

% Rational approximation for central region:
k = plow <= p & p <= phigh;
if any(k(:))
  q = p(k) - 0.5;
  r = q.*q;
  z(k) = (((((a(1)*r+a(2)).*r+a(3)).*r+a(4)).*r+a(5)).*r+a(6))...
         .*q./(((((b(1)*r+b(2)).*r+b(3)).*r+b(4)).*r+b(5)).*r+1);
end

% Rational approximation for lower region:
k = 0 < p & p < plow;
if any(k(:))
  q  = sqrt(-2*log(p(k)));
  z(k) = (((((c(1)*q+c(2)).*q+c(3)).*q+c(4)).*q+c(5)).*q+c(6))...
         ./((((d(1)*q+d(2)).*q+d(3)).*q+d(4)).*q+1);
end

% Rational approximation for upper region:
k = phigh < p & p < 1;
if any(k(:))
  q  = sqrt(-2*log(1-p(k)));
  z(k) = -(((((c(1)*q+c(2)).*q+c(3)).*q+c(4)).*q+c(5)).*q+c(6))...
          ./((((d(1)*q+d(2)).*q+d(3)).*q+d(4)).*q+1);
end
```

```
% Case when P = 0:
z(p == 0) = -Inf;

% Case when P = 1:
z(p == 1) = Inf;

% Cases when output will be NaN:
k = p < 0 | p > 1 | isnan(p);
if any(k(:))
  z(k) = NaN;
end

% The relative error of the approximation has absolute value less
% than 1.15e-9.  One iteration of Halley's rational method (third
% order) gives full machine precision.
k = 0 < p & p < 1;
if any(k(:))
  e = 0.5*erfc(-z(k)/sqrt(2)) - p(k);          % error
  u = e * sqrt(2*pi) .* exp(z(k).^2/2);        % f(z)/df(z)
  %z(k) = z(k) - u;                             % Newton's method
  z(k) = z(k) - u./( 1 + z(k).*u/2 );          % Halley's method
end
```

Appendix F

Calibration MATLAB Scripts

MATLAB Code F.1: Script used to calculate the GOF and identify solutions that meet the calibration criteria.

```
% Utility data
load (['.\01.Utilities\',Year,'.mat'])

% Calibration Criteria
cMBE = .05;
cCV = .15;

% Begin Analysis

for ii = 1:nsample
    g_Error(:,ii) = (Qh_supply - m_mQh_demand(ii,:));
    g_MBE(1,ii) = (sum(g_Error((1:(12-p)),ii))./((12-p)-1))...
                                                /mean(Qh_supply);
    g_RMSE(1,ii) = sqrt(sum(g_Error((1:(12-p)),ii).^2)/((12-p)-1));
    g_CV_RMSE(1,ii) = sqrt(sum(g_Error((1:(12-p)),ii).^2)...
                              /((12-p)-1))/mean(Qh_supply);

    e_Error(:,ii) = (Qkwh - m_mQkwh(ii,:));
    e_MBE(1,ii) = (sum(e_Error((1:(12-p)),ii))./((12-p)-1))...
                                                 /mean(Qkwh);
    e_RMSE(1,ii) = sqrt(sum(e_Error((1:(12-p)),ii).^2)/((12-p)-1));
    e_CV_RMSE(1,ii) = sqrt(sum(e_Error((1:(12-p)),ii).^2)...
                                    /((12-p)-1))/mean(Qkwh);

    % Price/Therm / Boiler Efficiency  Joules to Therms conversion
    Therm = sum(m_mQh_demand(ii,:))*1.15/.79*(1/(105.505585*10^6));
    kW = .07*sum(m_mQkwh(ii,:));      % Price/kW
    eW(1,ii) = kW/(kW+Therm);
    gW(1,ii) = (Therm)/(kW+Therm);
end
```

```
[pPrice,ePrice,sPrice,pEven,eEven,sEven,pHeating,eHeating,sHeating,...
    pElectric,eElectric,sElectric,pHdemand,eHdemand,sHdemand,...
    pEdemand,eEdemand,sEdemand,pCdemand,eCdemand,sCdemand,p_lhs,...
    e_lhs,s_lhs,allGOF] = gofAnalysis(m_mQh_demand,m_mQc_demand,...
    m_mQkwh,g_CV_RMSE,e_CV_RMSE,g_MBE,e_MBE,eW,gW,12,o_lhs,cMBE,cCV);

nrgStats = [g_MBE;g_CV_RMSE;e_MBE;e_CV_RMSE];

save ('..\04.MorrisMethd\morResult.mat','m_mQh_demand','m_mQkwh', ...
    'g_MBE','g_CV_RMSE','e_MBE','e_CV_RMSE')

resultname = ['..\',Folder,'\Results_',Year,Sim,'.mat'];
save (resultname,'pPrice','ePrice','sPrice','pEven','eEven',...
    'sEven','pHeating','eHeating','sHeating','pElectric',...
    'eElectric','sElectric','pHdemand','eHdemand','sHdemand',...
    'pEdemand','eEdemand','sEdemand','pCdemand','eCdemand',...
    'sCdemand','p_lhs','e_lhs','s_lhs','allGOF','Qh_supply',...
    'Qkwh','m_mQh_demand','m_mQkwh','nrgStats','m_mQh_demand',...
    'm_mQc_demand','m_mQkwh','g_Error','e_Error')
```

MATLAB Code F.2: Function to carry out the analysis for each GOF solution set.

```
function [pPrice,ePrice,sPrice,pEven,eEven,sEven,pHeating,...
    eHeating,sHeating,pElectric,eElectric,sElectric,pHdemand,...
    eHdemand,sHdemand,pEdemand,eEdemand,sEdemand,pCdemand,...
    eCdemand,sCdemand,p_lhs,e_lhs,s_lhs,allGOF] = gofAnalysis...
    (m_mQh_demand,m_mQC_bldg,m_mQkwh,g_CV_RMSE,e_CV_RMSE,g_MBE,...
    e_MBE,eW,gW,nMonths,o_lhs,cMBE,cCV)

nsample = size(g_MBE,2);
smallnum = 1e-3;

pMBE = zeros(1,nsample);
pCV_RMSE = zeros(1,nsample);
pGOF = zeros(1,nsample);

MBE = zeros(1,nsample);
CV_RMSE = zeros(1,nsample);
GOF = zeros(1,nsample);

e_GOF = zeros(1,nsample);
g_GOF = zeros(1,nsample);

% Price Weight vs. Even Weight
for j=1:nsample
    pMBE(1,j) = wFactor(g_MBE(1,j),gW(1,j),e_MBE(1,j),eW(1,j));
```

```
    pCV_RMSE(1,j) = wFactor(g_CV_RMSE(1,j),gW(1,j),...
                                e_CV_RMSE(1,j),eW(1,j));
    pGOF(1,j) = wFactor(pMBE(1,j),1,pCV_RMSE(1,j),1);

    MBE(1,j) = wFactor(g_MBE(1,j),1,e_MBE(1,j),1);
    CV_RMSE(1,j) = wFactor(g_CV_RMSE(1,j),1,e_CV_RMSE(1,j),1);
    GOF(1,j) = wFactor(MBE(1,j),1,CV_RMSE(1,j),1);

    e_GOF(1,j) = wFactor(e_MBE(1,j),1,e_CV_RMSE(1,j),1);
    g_GOF(1,j) = wFactor(g_MBE(1,j),1,g_CV_RMSE(1,j),1);
end

allGOF = [pGOF(1:nsample);pMBE(1:nsample);pCV_RMSE(1:nsample);...
    GOF(1:nsample);MBE(1:nsample);CV_RMSE(1:nsample);...
    e_GOF(1:nsample);e_MBE(1:nsample);e_CV_RMSE(1:nsample); ...
    g_GOF(1:nsample);g_MBE(1:nsample);g_CV_RMSE(1:nsample)];

p = 0;
e = 0;
s = 0;

for j=1:nsample
    if pMBE(1,j) < (cMBE+smallnum)
        if pCV_RMSE(1,j) < (cCV+smallnum)
            p = p + 1;
            p_mark(1,p) = j;
        end
    end
end

for j=1:nsample
    if MBE(1,j) < (cMBE+smallnum)
        if CV_RMSE(1,j) < (cCV+smallnum)
            e = e + 1;
            e_mark(1,e) = j;
        end
    end
end

for j=1:nsample
    if e_CV_RMSE(1,j) < (cCV+smallnum)
        if abs(e_MBE(1,j)) < (cMBE+smallnum)
            if g_CV_RMSE(1,j) < (cCV+smallnum)
                if abs(g_MBE(1,j)) < (cMBE+smallnum)
                    s = s + 1;
                    s_mark(1,s) = j;
                end
            end
```

```
        end
    end
end


if p == 0
    p_mark = 1;
    disp('No price solutions')
end

pGof = pGOF(1,p_mark(1:p));
[TP,OP] = sort(pGof,2);
pSol = [p_mark(1,OP(1:length(pGof))); TP(1:length(pGof))];

pPrice = zeros(4,size(pSol,2));
pEven = zeros(4,size(pSol,2));
pHeating = zeros(4,size(pSol,2));
pElectric = zeros(4,size(pSol,2));

pHdemand = zeros(nMonths,size(pSol,2));
pCdemand = zeros(nMonths,size(pSol,2));
pEdemand = zeros(nMonths,size(pSol,2));

p_lhs = nan(length(p_mark),size(o_lhs,2));

for k=1:size(pSol,2)
    p =  pSol(1,k);

    pPrice(1,k)= p;
    pPrice(2,k)= pGOF(1,p);
    pPrice(3,k)= pMBE(1,p);
    pPrice(4,k)= pCV_RMSE(1,p);

    pEven(1,k)= p;
    pEven(2,k)= GOF(1,p);
    pEven(3,k)= MBE(1,p);
    pEven(4,k)= CV_RMSE(1,p);

    pHeating(1,k)= p;
    pHeating(2,k)= g_GOF(1,p);
    pHeating(3,k)= g_MBE(1,p);
    pHeating(4,k)= g_CV_RMSE(1,p);

    pElectric(1,k)= p;
    pElectric(2,k)= e_GOF(1,p);
    pElectric(3,k)= e_MBE(1,p);
    pElectric(4,k)= e_CV_RMSE(1,p);
```

```
    pHdemand(:,k) = m_mQh_demand(:,p);
    pEdemand(:,k) = m_mQkwh(:,p);
    pCdemand(:,k) = m_mQC_bldg(:,p);

    p_lhs(k,:) = o_lhs(p,:);
end


if e == 0
    e_mark = 1;
    disp('No even solutions')
end

eGof = GOF(1,e_mark(1:e));
[TE,OE] = sort(eGof,2);
eSol = [e_mark(1,OE(1:length(eGof))); TE(1:length(eGof))];

ePrice = zeros(4,size(eSol,2));
eEven = zeros(4,size(eSol,2));
eHeating = zeros(4,size(eSol,2));
eElectric = zeros(4,size(eSol,2));

eHdemand = zeros(nMonths,size(eSol,2));
eCdemand = zeros(nMonths,size(eSol,2));
eEdemand = zeros(nMonths,size(eSol,2));

e_lhs = nan(length(e_mark),size(o_lhs,2));

for k=1:size(eSol,2)
    e = eSol(1,k);

    ePrice(1,k)= e;
    ePrice(2,k)= pGOF(1,e);
    ePrice(3,k)= pMBE(1,e);
    ePrice(4,k)= pCV_RMSE(1,e);

    eEven(1,k)= e;
    eEven(2,k)= GOF(1,e);
    eEven(3,k)= MBE(1,e);
    eEven(4,k)= CV_RMSE(1,e);

    eHeating(1,k)= e;
    eHeating(2,k)= g_GOF(1,e);
    eHeating(3,k)= g_MBE(1,e);
    eHeating(4,k)= g_CV_RMSE(1,e);

    eElectric(1,k)= e;
    eElectric(2,k)= e_GOF(1,e);
```

```
    eElectric(3,k)= e_MBE(1,e);
    eElectric(4,k)= e_CV_RMSE(1,e);

    eHdemand(:,k) = m_mQh_demand(:,e);
    eEdemand(:,k) = m_mQkwh(:,e);
    eCdemand(:,k) = m_mQC_bldg(:,e);

    e_lhs(k,:) = o_lhs(e,:);
end

if s == 0
    s_mark = 1;
    disp('No strict solutions')
end

sGof = GOF(1,s_mark(1:s));
[TS,OS] = sort(sGof,2);
sSol = [s_mark(1,OS(1:length(sGof))); TS(1:length(sGof))];

sPrice = zeros(4,size(sSol,2));
sEven = zeros(4,size(sSol,2));
sHeating = zeros(4,size(sSol,2));
sElectric = zeros(4,size(sSol,2));

sHdemand = zeros(nMonths,size(sSol,2));
sCdemand = zeros(nMonths,size(sSol,2));
sEdemand = zeros(nMonths,size(sSol,2));

s_lhs = nan(length(s_mark),size(o_lhs,2));

for k=1:size(sSol,2)
    s = sSol(1,k);

    sPrice(1,k)= s;
    sPrice(2,k)= pGOF(1,s);
    sPrice(3,k)= pMBE(1,s);
    sPrice(4,k)= pCV_RMSE(1,s);


    sEven(1,k)= s;
    sEven(2,k)= GOF(1,s);
    sEven(3,k)= MBE(1,s);
    sEven(4,k)= CV_RMSE(1,s);

    sHeating(1,k)= s;
    sHeating(2,k)= g_GOF(1,s);
    sHeating(3,k)= g_MBE(1,s);
    sHeating(4,k)= g_CV_RMSE(1,s);
```

```
    sElectric(1,k)= s;
    sElectric(2,k)= e_GOF(1,s);
    sElectric(3,k)= e_MBE(1,s);
    sElectric(4,k)= e_CV_RMSE(1,s);

    sHdemand(:,k) = m_mQh_demand(:,s);
    sEdemand(:,k) = m_mQkwh(:,s);
    sCdemand(:,k) = m_mQC_bldg(:,s);

    s_lhs(k,:) = o_lhs(s,:);
end
```

MATLAB Code F.3: Sub-function to alter the weighting of the GOF.

```
function [output] = wFactor(var1,weight1,var2,weight2)

output = ((weight1^2*var1^2 + weight2^2*var2^2)/...
                        (weight1^2+weight2^2))^(1/2);
```