Industrial Artificial Intelligence (2025) 3:4
https://doi.org/10.1007/s44244-025-00025-1

RESEARCH

Surrogate-assisted evolutionary multi-objective optimisation of office building glazing

Alexander E. I. Brownlee[1] · **Ernest R. O. Vanmosuinck**[1]

Received: 11 October 2024 / Accepted: 15 April 2025

Abstract
The quantity and positioning of glazing on a building's facade has a strong influence on the building's heating, lighting, and cooling performance. Evolutionary algorithms have been effective in finding glazing layouts that optimise the trade-offs between these properties. However, this is time-consuming, needing many calls to a building performance simulation. Surrogate fitness functions have been used previously to speed up optimisation without compromising solution quality; our novelty is in the application of a surrogate to a binary encoded, multi-objective, building optimisation problem. We propose and demonstrate a process to choose a suitable model type for the surrogate: a multilayer perceptron (MLP) is found to work best in this case. The MLP is integrated with the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) algorithm, and experimental results show that the surrogate leads to a significant (400x) speedup. This allows the algorithm to find solutions that are better than the algorithm without a surrogate in the same timeframe. Updating the surrogate at intervals improves the solution quality further with a modest increase in run time.

Keywords Simulation · Optimisation · Evolutionary algorithm · Surrogate

1 Introduction

Window positioning and shading is an important consideration in the design of a building envelope, having a large impact on the energy use associated with the building's artificial lighting, heating and cooling. Well-designed windows can reduce energy consumption by providing natural lighting and ventilation. Conversely, windows can increase energy consumption—in warm weather solar heat gain requires additional artificial cooling and in cold weather heat losses necessitate more heating. Earlier work [1, 2] has shown how this can be formulated as an optimisation problem that seeks to locate the Pareto-optimal front for two objectives: minimisation of the projected energy requirements of the operational building and minimisation of the construction cost.

The large number of explicit and implicit variable interactions, and non-convex optimisation function led to the use of evolutionary algorithms (EAs) to approximate the Pareto-optimal front by Brownlee et al. [1] and Wright et al. [2]. While this approach works well, each function evaluation is relatively time-consuming, making a call to a building performance simulation (specifically, *EnergyPlus*), and a typical run requires thousands. Thus the present paper explores the use of surrogate fitness functions [3, 4] for this problem. A surrogate is a machine learning model trained on previous fitness evaluations, that can then take the place of the long-running simulation in order to speed up the search. Surrogates have been demonstrated for several EA-based optimisations of building designs in the past (e.g., [5–18]), but in the vast majority of cases these follow a continuous or mixed-integer solution representation (that is, the variables in the optimisation problem are reals or integers). The glazing problem we focus on uses a binary encoding so, as a novelty in contrast to earlier work, we explore surrogate models suited to binary encodings. A further novelty is that, for the problem at hand, the number of bits set *true* in a solution (i.e. the total number of glazed cells) has a strong influence on the objectives, so we propose a structured strategy to generate the training data for the surrogate in place of uniform sampling of the space. Existing literature applying surrogate models within building performance optimisation typically

✉ Alexander E. I. Brownlee
alexander.brownlee@stir.ac.uk

Ernest R. O. Vanmosuinck
ernest.vanmo@gmail.com

[1] University of Stirling, Stirling, UK

Published online: 14 May 2025

uses one model type (e.g., artificial neural networks, or Kriging models), without a broader exploration of what might be most suitable. This motivates us to also propose a simple, systematic, approach to exploration of the models, that should be adopted for any application involving a surrogate to ensure best performance.

Our key contributions are as follows:

- a novel systematic methodology rooted in established best practice in machine learning for comparing different model types, hyperparameter tuning, and choosing the model best suited to this specific problem
- demonstration of the methodology through a case study featuring an implementation and application of a surrogate for a binary-encoded, multi-objective, building optimisation problem, to which surrogates have not previously been applied
- a novel process for generating training data with a good spread over the search space for binary-encoded problems where the overall count of false/true values has some impact on the optimisation objectives
- experimental results showing the impact on run time and solution quality of updating the model at fixed intervals, with the conclusion that the model should indeed be updated at intervals for best performance.

We show that the addition of surrogate modelling to the optimisation algorithm is also effective for a problem with binary-encoded variables, and that modest improvements are to be gained by updating the model over the course of the run. We begin with an overview of related work in Sect. 2. We then present the background to the glazing problem in our study in Sect. 3, before presenting our methodology and experimental results in Sects. 4 and 5. We discuss, summarise and present directions for future work in Sect. 6.

2 Related work

Evolutionary algorithms have been applied to a wide range of real-world applications [19], and for costly fitness functions, surrogate fitness functions have been a common approach to improving search efficiency [4]. It is unsurprising, then, that there is also a wide and growing body of existing research bringing together building design, including glazing, and multi-optimisation by evolutionary algorithms, with some examples also showing the use of surrogate fitness functions. Here we provide a brief summary of the relevant literature at the intersection of these themes.

Evins [20] presents work using a genetic algorithm with strength Pareto fitness assignment to optimise building designs finding the trade-off between solar gain reduction and construction cost. Similar to this work, it also used EnergyPlus for the design simulations. Suga et al. [21] used a multi-objective GA to optimise a window design problem, varying window positions, sizes and glass types. Rather than using an off-the-shelf simulation, custom functions were used to compute energy use and building performance. Caldas and Norford [22] used a GA to optimise window sizing and placement for minimum building energy use, but the windows were restricted to being rectangular in shape. The simulation software used was DOE2.1E [23]. Vukadinović et al. [24] used NSGA-II to optimise building construction (window-to-wall ratios, glazing types, shading) with mixed variable types for minimisation of three objectives: energy for heating and cooling, and hours of thermal discomfort. All objectives were computed using EnergyPlus.

Most relevant to this paper, the work described by Wright and Mourshed [25] introduced the concept of geometric window design. There, the problem was formulated as a single-objective constrained optimisation, and experiments were performed using a traversal method and a genetic algorithm. A detailed analysis of various aspects of the problem was also performed. The problem was then extended in complexity with the addition of window overhangs and a cost objective by Wright et al. [2] and Brownlee et al. [1]. The present paper builds on that earlier series of work—primarily by integrating the use of surrogate fitness functions.

Over the past few years several applications of surrogate fitness functions to speed up optimisation processes in building design applications have been described. Westermann and Evins [26] give an excellent review of the use of surrogate models for sustainable building design, including recommendations for model choice and approaches to implementation. However, most examples found by that review used the surrogate as part of a sensitivity or uncertainty analysis via factorial or Latin hypercube style sampling, with 22 focused on optimisation and a sub-sample of those using evolutionary or other search algorithms. We now summarise the most relevant examples, including those publications published since that review that make use of multi-objective optimisation and surrogate models in building design. Support Vector Machines were used by Eisenhower et al. [12] to build several meta-models of an EnergyPlus model. Sensitivity analysis was used to choose specific variables for optimisation. Optimisation using the meta-models was shown to find results close to those obtained by using EnergyPlus model alone. Ghaderian and Veysi [27] applied NSGA-II to optimisation of several continuous variables related to HVAC controls for minimisation of two objectives: gas and electricity consumption. The objectives were initially computed over set of configurations obtained through design-of-experiments; these were then used to fit linear regression surrogate models that were called by the optimisation algorithm. Yigit [28] used Latin Hypercube Sampling to create a training set that was evaluated using EnergyPlus

simulations, to which a Gradient Boosting Model was then fitted and used as a surrogate model for NSGA-II. Artificial Neural Networks (ANNs) were used by [8, 10, 15] in place of simulation runs as part of optimisation by a single-objective genetic algorithm, particle swarm optimisation, and ant colony optimisation, respectively. In each case, a building performance simulation was run on large numbers of example designs, and the data from this was used to train and validate the ANN model. A similar approach combining ANNs with NSGA-II for two and three objective multi-objective optimisation was described by by Magnier and Haghighat [16]; Gossard et al. [14]; Aijazi and Glicksman [6]; Asadi et al. [5]; Bre et al. [29]; Zou et al. [30]. Chegari et al. [31] combined ANNs with Multi-Objective Particle Swarm Optimisation algorithm (MOPSO) to optimise several continuous building envelope variables for energy consumption and indoor thermal comfort. Yue et al. [32] found that an ANN achieved better accuracy as a surrogate for EnergyPlus calculations of a gymnasium's degree-hours in naturally ventilated seasons and energy consumption in air-conditioning seasons compared to a Support Vector Machine, Random Forest Regression, and Radial Basis Function neural network. The ANN surrogate was called by NSGA-II to optimise discretised versions of mixed variable types representing the construction and lighting configurations.

The above approaches all used a single training stage: the surrogate models were trained and validated on data generated by simulation runs, then the optimisation proceeded to run using only the surrogate model. This works reasonably well, but the obvious problem is that errors in the surrogate model can lead the optimisation algorithm into a sub-optimal region of the search space. In a few cases, methods to update the model have been explored. A Kriging surrogate model was trained on EnergyPlus performance estimates in [17], with a GA then calling the model in place of the simulation. Once the GA could progress no further, the solution generated by it was evaluated with EnergyPlus, the model retrained, and the optimisation run again. A case study showed the approach to find good solutions efficiently. Another approach [7] used an ensemble (referred to as a "committee") of ANN surrogate models within multi-objective Particle Swarm Optimisation (PSO) to minimise the energy consumption, improve the thermal comfort of the occupants and increase the energy self-sufficiency of residential buildings. If there was a high level of disagreement between the models in the ensemble when estimating the objectives for a solution, the solution would be re-evaluated by the simulation and the models retrained.

Chen and Shi [11] used a three-layer radial basis function (RBF) neural network to estimate energy consumption and indoor comfort. NSGA-II was used to seek the trade-off between these two objectives, using the surrogate model to filter promising solutions. The surrogate model was then updated each generation. Brownlee and Wright [9] also used RBFs in place of long-running simulations to filter solutions within the NSGA-II algorithm. In that case, the surrogate was checked for accuracy every generation, and updated if the correlation between the surrogate's predictions and the results of the full simulation fell below a certain threshold.

Kriging has the ability to estimate the uncertainty on predictions. This was employed by Safarzadegan Gilan et al. [33] as a means of sample selection to improve the model. A recent work by by Østergård et al. [34] comparing the accuracy (and several other characteristics) of six meta-modelling approaches, including linear regression, Gaussian processes (Kriging), ANNs, support vector regression, random forest and multivariate adaptive regression splines, on 13 problems, showed Kriging to work best for smaller problems and ANNs to work best on larger ones.

In the present work, we focus on a binary-encoded problem with multiple objectives. A surrogate model is used to in place of some simulation runs, and the surrogate is updated at regular intervals using feedback from the simulation. None of the existing literature above has considered this combination of factors in the building performance domain. Furthermore, most approaches adopt, from the outset, a single machine learning model as their surrogate. While some explicitly refer to cross-validation for tuning hyperparameters (e.g., Bamdad Masouleh et al. [8]; Bamdad et al. [7]), only Østergård et al. [34] and Yue et al. [32] appear to have performed a systematic comparison of model types, and both pieces of work focused on a problem with mixed variable types rather than the binary encoding that we consider here.

3 Case study optimisation problem

The problem forming our case study was first presented by Brownlee et al. [1] and Wright et al. [2], and for further details the interested reader is referred to the latter of those earlier works. Here, we summarise the main points relevant to understand the optimisation task at hand.

Glazing and its design have a significant influence on the energy performance of buildings; it impacts on daylight penetration, artificial lighting energy use, and heating and cooling energy use. The approach taken by Brownlee et al. [1] and Wright et al. [2] allows variation in the shape and position of the windows, in addition to the overall amount of glazing. Those previous works demonstrated that genetic algorithms are effective for finding near-optimal designs with respect to minimising building energy use and the capital cost of construction.

Thus, again here we seek to optimise the size, shape and position of windows and associated shading placed on a building; the goal is a design which minimises operational energy use (for heating, lighting and cooling) and capital cost

(for construction). These objectives conflict with each other: a building with minimal cost has no glazing, so requires more energy for lighting, and does not benefit from solar gain so potentially requires more energy to heat. The picture is further complicated by the increased energy requirement for air conditioning if there is extensive glazing. This led the earlier works to explore multi-objective optimisation algorithms that approximate the Pareto-optimal trade-off between the objectives. In the rest of this section we now detail the specific building design scenario considered in our experiments, and the formulation of the optimisation problem's objectives and variables. We will conclude this section with the approach to measuring run times, which provides the main motivation for the novel use of surrogates in the present paper.

3.1 Building configuration and weather

The wall measures 15 m wide by 8.2 m high and is divided into a grid of 120 $1m^2$ cells, 15 wide by 8 high, and each cell may be either glazed or unglazed solid wall. The atrium behind the wall is 15 m deep. In Fig. 1, we see the fully glazed building with shading overhangs on all windows. This approach to the design can result in solutions reminiscent of those developed by Le Corbusier for the Chapel of Notre Dame du Haut, where the South façade, known as the "wall of light", has windows of varying size that are randomly placed on the wall. In our target building, each window also has an optional shading overhang, a physical shade designed to reduce the light and heat entering the building via the window. Inclusion of these adds extra complexity to the optimisation problem as each overhang affects the energy impact of the window it is associated with and has corresponding increase in overall cost.

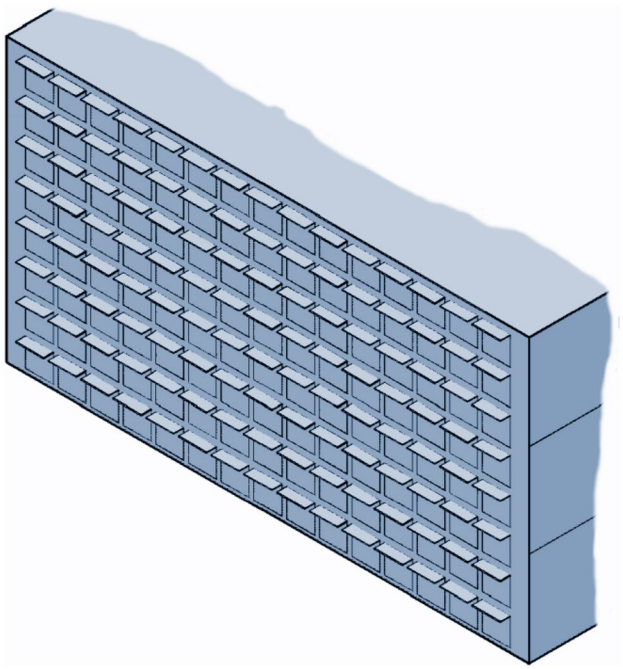

Fig. 1 Fully glazed facade

In common with the previous studies on this application the building in question is based on an atrium of a commercial building located in Chicago, USA. Chicago has interesting weather for such a study, with high temperatures in the summer and low temperatures in the winter. The atrium is 15 m wide by 15 m long by 8.1 m high with only the southern façade being exposed to the external environment. The other three sides of the atrium are connected to interior spaces that are controlled to have the same thermal conditions as the atrium. The roof, internal partition walls and the external façade have a light-weight construction; the floor is constructed from uninsulated concrete; and the window cells have a double-glazed construction.

3.2 Optimisation objective 1: energy

The first objective is to minimise the energy use of the building. This is the unweighted sum total of the energy used by heating, cooling and lighting systems over a specified period in a particular set of environmental circumstances. These figures are computed by the EnergyPlus building simulation package [35] (v9.3) and the process is simply that the design specification is generated then passed to EnergyPlus, which produces a text CSV containing the energy performance data. The total energy consumption $f_{energy}(x)$ for a specific design configuration x of the building is extracted from this CSV. The full specification of the building's design and simulation configuration is given in Sections 5.1 and 5.2 of [2]. The EnergyPlus simulation simulates the building's performance over an entire year, using a publicly available weather data set for the location. However, this process can take tens of seconds per run, motivating the exploration of surrogates in the present study (indeed, with more complex building models, minutes or hours per run are typical).

3.3 Optimisation objective 2: cost

The second objective is the minimisation of the construction cost for the building given the specified window and shading configuration x. The size of each glazed cell is fixed at $1m^2$ so this is a straightforward linear function of the cost of the opaque concrete construction, the number of windows $n_w(x)$ and shading overhangs $n_o(x)$ and does not involve the simulation software, so does not need to be replaced by the surrogate. The total cost $f_{cost}(x)$ is defined in equations (1) to (3).

The total cost of the windows c_w is £350 per glazed cell, plus the base cost of £112 for the concrete construction of each non-glazed cell:

$$c_w = 112(120 - n_w(x)) + 350n_w(x). \quad (1)$$

The total cost of overhangs c_o is based on them being £128 each:

$$c_o = 128n_o(x). \quad (2)$$

The total capital cost $f_{cost}(x)$ is the sum of unglazed cells, glazed cells and overhangs:

$$f_{cost} = c_w(x) + c_o(x). \quad (3)$$

These cost figures are taken from Spon [36] and are somewhat old, but are kept for consistency with the earlier work on this same building [1, 2].

3.4 Optimisation variables and encoding

The problem naturally lends itself to a binary solution representation; this is the approach adopted by Brownlee et al. [1] and Wright et al. [2] which we replicate here. The wall is divided into 120 cells which may be glazed or unglazed: this translates into a 120 variable bit string in which a bit is set to true for a glazed cell and false for an unglazed one. A second string of 120 bits represents the presence of shading overhangs on each window. When compiling the building design for input to the EnergyPlus simulation, a shading is only included if its corresponding cell is glazed—otherwise, it is omitted. This is a restriction imposed by the simulation which will only allows shading above glazed areas. It does mean that for individuals with unglazed areas there are corresponding redundant variables—this could be avoided by using a variable-length bit string encoding but it was felt that this represented excessive complexity. It would be interesting in future work to try an encoding which avoids this redundancy, possibly by using variables with three possible values of unglazed, glazed, or glazed + overhang.

3.5 Optimisation problem

Bringing together the previous two sections, we can now formally define the optimisation problem. The goal is to find a solution x in the space of 240 bit vectors $X = \{0, 1\}^{240}$ that simultaneously minimises $f_{energy}(x)$ and $f_{cost}(x)$:

$$minf_{energy}(x), f_{cost}(x), \quad where \;\; x \in X = \{0, 1\}^{240}. \quad (4)$$

3.6 Timing

The justification for using the surrogate is the long running time of the EnergyPlus simulation. To illustrate this point, we report the results of a simple timing experiment whereby the run times of EnergyPlus on randomly generated solutions are given.

We generated 100 solutions; in each solution $1/n$ bits were true (i.e. glazed or shading applied) where n was the solution number (i.e. $n = [1, 100]$). The specific glazed cells are chosen at random. The evaluations were performed on a Ryzen 5 3600 CPU @3.6GHz. The run times for evaluations were from 1.6 s for the solutions with very little glazing to 24.5 s for solutions near 100% glazed. The mean run time over the 100 solutions explored was 11.2 s. Given that a typical run of an EA will take 1000 s of evaluations, the resulting run times run in to many hours, even days. Each of the repeat runs in our experiments using EnergyPlus alone took around one week of CPU time. This obviously increases to 30 weeks of CPU time with the repeat runs necessary to add confidence to the results. While this can be sped up using parallelism and faster CPUs, we propose that this can be supplemented through the use of surrogate models for additional improvements to run times. Occupying a designer's machine for weeks to months of time (or cloud compute instances charged per CPU hour) for a single modestly sized building project is clearly infeasible. More importantly, as our results show, the surrogate EA is allowed to run for more iterations in the same time frame than would be possible in years' worth of full simulations, leading to still better results.

4 Methodology

We now explain the overall approach of our proposed methodology for integrating surrogates within building performance optimisation. We begin with the selection of a suitable surrogate model and its hyperparameters before summarising the evolutionary algorithm we used (NSGA-II) and the integration of the model with it.

4.1 Surrogate model selection and development

The choice of model type, and its hyperparameters, can have a large effect on the model's accuracy. For example, Multilayer Perceptrons are sensitive to the hyperparameters used during training: the learning rate, momentum, training time and number of hidden layer neurons are only a few factors that can influence accuracy. Similar is true for most model types. Choice of model type itself also has a large influence: linear regression is simple and transparent but assumes linear relationships. Regression Tree models are efficient and relatively transparent, but prone to overfitting. Multilayer Perceptrons can model non-linear dependencies well, but the backpropagation algorithms used to train them can be slow and can get caught in local optima. No single model is best for all problems. It is thus important to explore as broad a combination of these systematically to choose a model with the highest accuracy and generality for a given problem. Thus we propose the following procedure to select a suitable surrogate model.

Following standard practice in machine learning [37], we considered several potential models and tested their performance as surrogates for this problem using k-fold cross-validation. This well-established technique trains several copies of a model, validating its accuracy on unseen data, by splitting the data set into several 'folds'. Each fold serves as the validation data for one copy of the model, with the rest serving as training data. Average performance over all folds is used to compare different model types and hyperparameter configurations, in order to determine how well each particular model configuration fits the overall distribution of the data.

First, 500 solutions were generated to serve as training data. For this specific problem, an additional consideration is that the objectives are both sensitive to the total number of glazed cells. Uniform sampling, which would generally create solutions with around 50% glazing, would consequently miss a large part of the search space. So, we propose a structured scheme to ensure that the training data contained a wide spread of glazing amounts from 0% to 100%, set out in Algorithm 1. This procedure could go further still by ensuring that the locations of glazing are adequately varied as well, but as the results shortly will show, good results are obtained with the procedure as it stands. These solutions were evaluated for energy consumption by EnergyPlus as per Sect. 3.

Algorithm 1 Method to generate a uniform distribution of solutions (note: total is capped at 500, so in practice the "all zero" unglazed solution replaces one of the 90%+ glazed ones)

Input: R Random number generator: e.g., $x \in_R X$; x takes a value in the range of possible values for X sampled uniformly at random

$\Pi = \emptyset$ /* solution set */
$\Pi \leftarrow x = \{0, \ldots, 0\}$ /* add a solution with no glazing */
for $r = 0.1$ **to** 1 **step** 0.1 **do** /* for each glazing proportion */
  for $s = 1$ **to** 50 **step** 1 **do** /* 50 solutions for each proportion */
    $x = \{\}$ /* new solution */
    for $i = 0$ **to** 119 **step** 1 **do** /* over cell grid */
      $x_i = (x \in_R [0, 1]) < r$ /* glazed at random, weighted by r */
    end
    for $i = 120$ **to** 239 **step** 1 **do** /* over cell grid */
      $x_i = (x \in_R [0, 1]) < 0.5$ /* shading overhang uniformly at random */
    end
    $\Pi \leftarrow x$ /* add solution to set */
  end
end

Several models were then fitted to this data following a fivefold cross-validation. Most machine learning models have multiple hyperparameters that control their training convergence and generalisation capabilities. It is important to tune such parameters for optimal performance on a given target data set. Two approaches are commonly used for this in the machine learning literature: Grid Search and Random Search. The former uses a full factorial coverage of all combinations of possible hyperparameters, and while guaranteeing to find the best configuration from those available, is time-consuming to run. A frequently used compromise is Random Search, whereby a fixed number of configurations drawn uniformly at random from the space of possible hyperparameters are tested. Thus, in our study a Random Search over 500 configurations was used to tune the hyperparameters for each. All models were the implementations from the WEKA Java machine learning library.[1] The mean r^2 over 5-folds for each model is given in Table 1, in addition to the best configuration found by parameter tuning. We use r^2 because this captures the model's ability to rank solutions. The optimisation algorithm we use employs tournament selection. When comparing solutions tournament selection is only concerned with their performance relative to each other, thus the model is only required to rank solutions accurately.

Following this experiment, it was clear that the multilayer perceptron offered good performance on this problem. Despite the fact that 500 solutions represent a very sparse coverage of the full search space (2^{240}), the MLP achieves an average r^2 of 0.992 over all folds. The MLP had also proved robust; the random search hyperparameter tuning exercise on the hidden layer sizes, momentum and learning rate did not improve the results beyond the default configuration. As a result, our surrogate implementation used the default multilayer perception configuration in WEKA: learning rate

[1] Version 3.8.5—https://www.cs.waikato.ac.nz/ml/weka/

Table 1 Mean r^2 over 5-folds for each model tested on training data of 500 solutions

Model	Description and best performing configuration after tuning	Mean r^2 over 5-folds
Decision stump	Single node tree as sanity check; no hyperparameters to tune	0.524
Decision table	Simple majority DT; with global table majority and forward search	0.888
Linear regression	Akaike criterion for feature selection, ridge with constant of 10^8	0.611
M5P tree	Uses pruning, smoothed predictions, and min 4 instances per leaf	0.820
Multilayer perceptron	Learning rate 0.3; momentum 0.2; single hidden layer of 240; stochastic gradient descent with 500 epochs	**0.992**
REP tree	Builds a regression tree using variance, pruned by reduced-error pruning (with backfitting); min 2 instances per leaf; no max depth	0.858
Simple linear regression	Picks the attribute that results in the lowest squared error	0.465
SMOReg	Support vector machine for regression. Trained using RegSMOImproved; complexity 1; PolyKernel	0.688

0.3; momentum 0.2; single hidden layer size of (number of features), i.e. 240; trained using stochastic gradient descent with 500 epochs.

4.2 NSGA-II

The Non-dominated Sorting Genetic Algorithm II [38]. Deb's algorithm is one the best-known and frequently implemented MOEAs. More importantly, although easily outperformed for problems with 3 or more objectives, it remains competitive for 2-objective problems [19], and was found to outperform several other algorithms for this glazing problem [1]. In common with most evolutionary algorithms, NSGA-II starts with a randomly generated population of candidate solutions. Each iteration of the algorithm, a new population is generated, gradually improving in quality until the population converges on an approximation to the Pareto front for the problem. To generate a new population, promising solutions are *selected*, then *recombined* ("crossover") and *mutated* (these operations being probabilistic). The process is repeated until either a predefined maximum number of generations is reached or a predefined maximum number of solutions evaluated. The selection of promising solutions is based on the solutions' *fitness*; to determine fitness, NSGA-II uses non-dominated sorting of individuals in the population, with a crowding distance penalty applied to individuals to maintain a diverse Pareto front. There is no external archive to hold the Pareto front, so the optimal front found is simply the set of non-dominated individuals in the final population.

As with the machine learning model's hyperparameters, it is generally best practice to explore different configurations for the genetic algorithm. The precise configuration of NSGA-II in our experiments was determined empirically. Following Brownlee et al. [1], where further details for implementation of crossover and mutation for this problem can be found, population sizes of {50,100,200}, crossover rates of {0.5,0.9,0.99,1}, and mutation rates of {0.25,0.5,1,2}/*n* were tested, and the best configuration comprised:

- Population size of 100
- Tournament selection
- Elitist generation replacement, whereby the fittest half of the union of the previous generation and newly generated offspring are carried to the next generation
- Uniform crossover with a rate of 100%
- Bit-flip mutation with a gene rate of 0.25/*n*.

The same configuration was used for the algorithm with and without the surrogate.

4.3 Integrating the surrogate

Integration of the surrogate with NSGA-II is relatively straightforward. Figure 2 shows the overall workflow of the base algorithm, and the workflow for the algorithm with surrogate. The main loop in both cases evaluates the solutions, then applies non-dominated sorting to locate the fronts (starting with the Pareto front, then identifying the front lying 'behind' it, and so on). Standard genetic algorithm selection, crossover, and mutation operators are then applied to generate a new population. The key difference is that, with the surrogate, a model is trained at the beginning of the run, and takes the place of calls to the EnergyPlus simulation. The final population in both algorithm configurations is always passed to EnergyPlus to ensure that the evaluation of the results that are output is consistent. We also explored a variation whereby the surrogate model is retrained at fixed intervals through the algorithm's run: the intervals being after 1/4, 1/2, and 3/4 of the iterations had completed. For retraining, EnergyPlus is called to evaluate the current population, and the results are used as training data to retrain the surrogate model from scratch. We also explore the impact of this retraining process later in the paper. The concept of retraining can be taken further. For example, Brownlee and Wright [9] proposed a measure based on model accuracy that was used to trigger the retraining process. In our proposed methodology, we wish to keep the overall framework

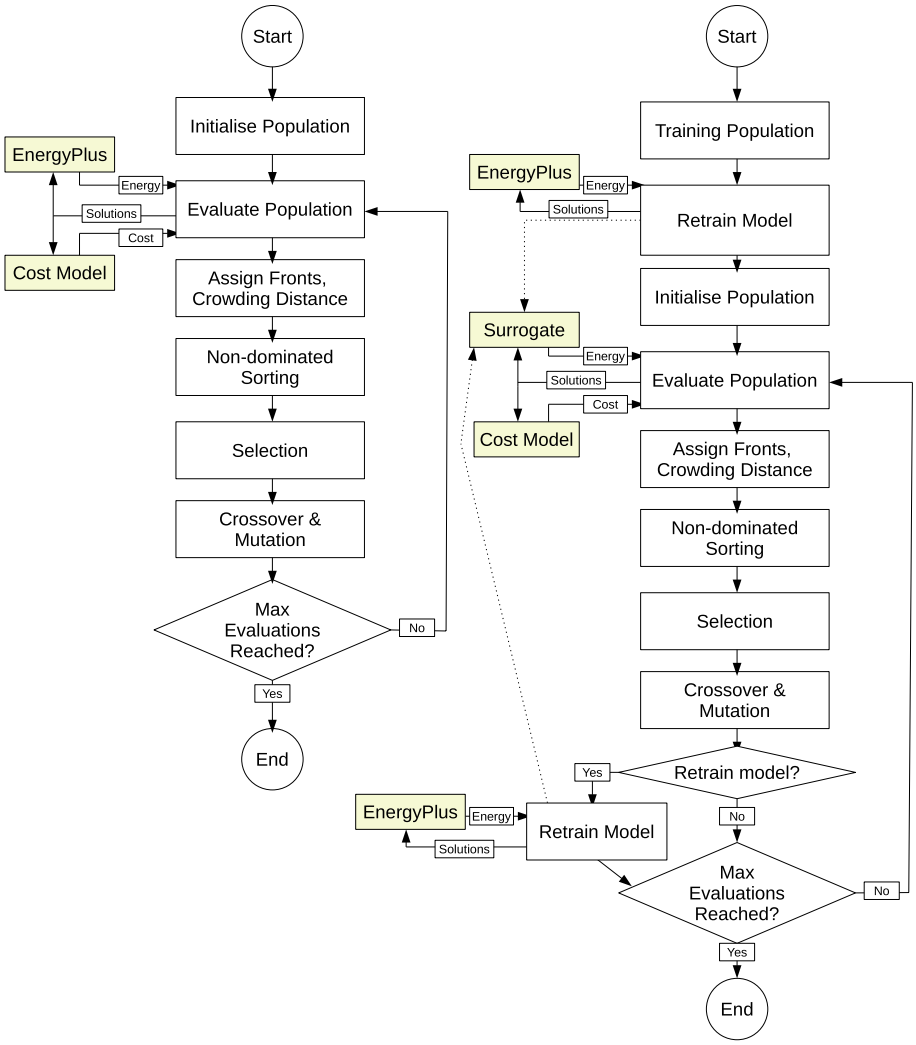

Fig. 2 The overall framework for NSGA-II (left) and NSGA-II with the surrogate model integrated (right)

as simple as possible so omit such a scheme, but it should be considered should the model error for a given problem increase unacceptably during the optimisation run.

5 Experiments and results

We now present the results of our experiments. We begin with the quality of solutions returned by the optimisation runs with different algorithm configurations. We then summarise the impact on run times of using the surrogate. Latterly, we look at a sample of solutions generated by the optimisation process, and the accuracy of the surrogate models.

5.1 Optimisation: solution quality

Three configurations of NSGA-II were each run on this problem 30 times, with each repeat using different randomly generated starting populations. The three configurations were: (1) without the surrogate, using EnergyPlus to evaluate all solutions; (2) with the surrogate, with no retraining after the surrogate was fitted at the start of the run; and (3) with the surrogate, being retrained as per Sect. 4.3. The random number generator seed was set so that the same 30 starting populations were used for both algorithm configurations. The algorithms were otherwise configured as described in Sect. 4.2, and ran for 50 000 evaluations (500 iterations). "evaluations" is the number of solutions evaluated by the algorithm as the search proceeds. These are either calls to EnergyPlus or to the surrogate. Runs of EnergyPlus used to train the surrogate are distinct from this as they do not directly form part of the search process. Run times appear in the next section, so it is also worth noting that all EnergyPlus simulations were run in parallel split across the 6 cores of a Ryzen 5 3600 CPU.

Summary attainment surfaces [39] for the three algorithms are shown in Fig. 3. These figures are similar in concept to boxplots, but designed for Pareto fronts, and show the aggregate performance of the algorithm runs. The solid lines show the parts of the objective space found by at least half of

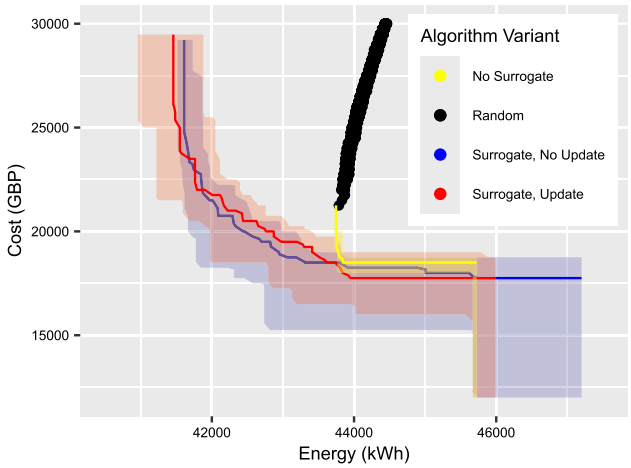

Fig. 3 Attainment surfaces for the three algorithm configurations, for runs limited to 50 000 evaluations. Each solid line indicates the region of the objective space reached by at least half of the runs of the algorithm, with the shading representing the region reached by all runs and by at least one run. The black points are the best of those visited random search limited to 50 000 evaluations

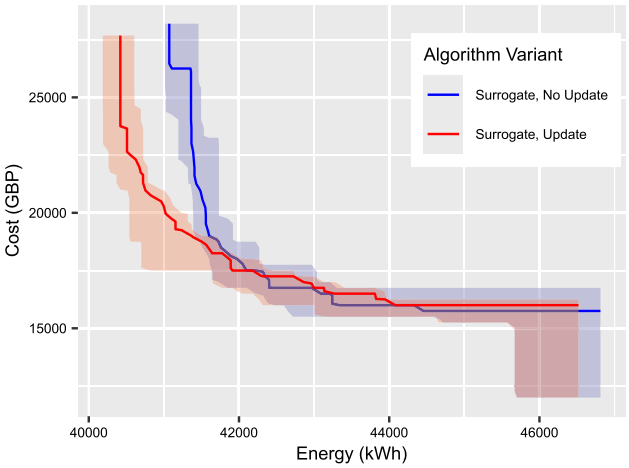

Fig. 4 Attainment surfaces for the three algorithm configurations, for runs limited to 100 000 iterations. Each solid line indicates the region of the objective space reached by at least half of the runs of the algorithm, with the shading representing the region reached by all runs and by at least one run

the repeat runs of the algorithm, with the shaded area showing variation: the regions reached by all runs and by at least one run. Also shown are, as a baseline, the best points generated by a random search of 50 000 evaluations, the same number visited by NSGA-II. Poorer (higher cost/energy) points found this way are omitted to allow the scale to concentrate on the attainment surfaces. NSGA-II improves on the random search in all the repeat runs. The algorithm using no surrogate shows consistent performance: a very narrow shaded area. The median attainment surface (solid line) for the algorithms using the surrogate are to the left and below those for the algorithm without the surrogate, indicating improved performance on average. Likewise, most of the shaded area for the surrogate runs is below the shaded area for the non-surrogate runs, indicating that most of the time the algorithms with the surrogate find better Pareto fronts that the algorithm without. It is also clear that the surrogate has allowed a greater exploration of the space; both variants of the algorithm including the surrogate consistently finding lower cost and lower energy solutions at the extremes of the fronts. The median run of the algorithm without the surrogate found solutions of 43 746kWh energy and £18 500 capital cost. The corresponding figures for the surrogate-assisted runs were 41 611kWh and £17 750 without model updates and 41 458kWh and £17 750 with updates. There is little difference in performance between the algorithm where the surrogate is updated and the algorithm where it is not.

The major advantage of the surrogate is the reduced time to evaluate solutions compared to running the EnergyPlus simulation every time. This allows for much longer runs to be made, so a second experiment ran the two algorithms using a surrogate to 100 000 iterations. The attainment surfaces for these are shown in Fig. 4. In both objectives, the results are improved on those seen for the 50 000 evaluation runs. Updating the surrogate model has resulted in better results for energy, with similar results for cost for both algorithm variants. The minimum figures for solutions in median fronts for these runs were 41 072kWh and £15 750 without model updates and 40 420kWh and £16 000 with updates. The explanation for the difference between the objectives is that only the energy objective uses the surrogate in place of the EnergyPlus simulation; cost does not. Imperfections in the surrogate can be corrected over the course of the run through retraining, and without this correction the algorithm is misled into a slightly poorer part of the space.

To add a numerical comparison of the Pareto fronts, we use the hypervolume (S) measure of Zitzler [40]. While not without its faults, Knowles and Corne [41] note that hypervolume provides a good balance between measuring the different traits of a Pareto front: convergence to the true Pareto front, spread of solutions and extent of the front. It also does not require the true Pareto front to be known in advance for comparison. Hypervolume is simply the multi-dimensional value between the Pareto front and a specific reference point. To compute hypervolumes, we normalised all objective values to [0, 1], using the minimum and maximum from the complete set of all solutions found over all the experiments. For energy, the range was [40186.55, 45670.30] and for cost the range was [12000, 28750]. The reference point for hypervolume was then (1, 1) in the normalised space. Table 2 shows the median hypervolumes computed for the Pareto fronts found by each algorithm. Larger values are preferable. A Wilcoxon signed-rank test on all pairs of algorithms

Table 2 Hypervolumes for each algorithm configuration

Evaluations	Surrogate, no update	Surrogate, update	No surrogate
50 000	$0.409_{0.029}$	$0.388_{0.048}$	$0.214_{0.013}$
10 000 000	$0.560_{0.024}$	$0.625_{0.040}$	n/a

Large figures are medians over the 30 runs, with inter-quartile ranges in subscript

Table 3 Run times in seconds for each algorithm configuration

Evaluations	Surrogate, no update	Surrogate, update	No surrogate
50 000	$112.02_{3.00}$	$397.00_{16.25}$	46259_{1010}
10 000 000	$476.65_{14.78}$	$701.50_{29.75}$	n/a

Large figures are medians over the 30 runs, with inter-quartile ranges in subscript

(using Bonferroni correction for multiple tests) indicated that the difference in hypervolume for was statistically significant with $p < 0.05$ in almost all cases. The only exception was for the 50 000 evaluation runs with the surrogate with updates vs the same algorithm without updates.

5.2 Optimisation: run times

Table 3 reports the median run times for each algorithm configuration. The algorithm without surrogate was not run for 100 000 iterations as it was estimated to take around 104 days to complete. The time saved by using the surrogate is clear: over 400x speedup for 50 000 evaluations. The additional EnergyPlus simulations used by the algorithm when updating the model does extend the run time. The previous section showed that this does lead to improved results over the algorithm without updates to the surrogate: there is obviously a trade-off here but given that the difference is a few minutes it would appear to be worthwhile including some updates to the model over the run.

5.3 Example solutions

The overall goal of the optimisation process is exploration of the design space. We now briefly consider how the results in the objective space relate to actual designs for the façade. Figure 5 shows solutions taken from the Pareto front with the highest hypervolume returned by the 100 000 iteration run with the surrogate and retraining. The layout on the left represents a solution with the lowest energy consumption for that configuration. The layout on the right represents the lowest cost solutions. The layout in the middle lies at about the mid-point of the trade-off between energy and cost. The rightmost solution is the minimal cost scenario (no glazing).

It is interesting that the glazed cells are not laid out in a specific pattern. The windows appear to spread all over the façade, rather than clustered in a specific area. This implies that the absolute locations of the glazed cells do not matter much, with the overall amount of glazing being more important. This is relevant to the analysis in the next section.

5.4 Model updates and accuracy

The accuracy of the surrogate is important: if it inaccurately predicts the objective values, then the optimisation algorithm may be 'misled' in the wrong direction and become unlikely to find an optimal solution. This is reflected by the improved results seen when updating the surrogate at intervals. It can also be quantified. For the final populations of the 100 000 iteration runs, we computed the Spearman correlation between the predicted energy values and the values computed by EnergyPlus. The correlation for the surrogate that had been updated was 0.407; for the non-updated surrogate it was 0.368. It is interesting that these correlations are rather weak, but still the surrogate was able to improve the performance over the course of the optimisation run. (We would expect this correlation to be much lower than the r^2 values seen when choosing the model in Sect. 4.1, because the final population of the optimisation run is genuinely unseen data for the model: the optimisation process is intended to find solutions quite different from the uniform sampling that generated the training data.)

We now consider whether the surrogate is better at predicting some types of solutions than others. The absolute error (difference between surrogate prediction and EnergyPlus) was calculated for each solution from the five Pareto fronts with the highest hypervolume for the 100 000 iteration runs. These errors are plotted against the number of glazed cells in the solutions in Fig. 6. The more shaded windows there are in a layout, the bigger the absolute error. This variation of model accuracy with cell count retrospectively provides some additional motivation for following the procedure we used to generate training solutions when choosing the model in the first place. The algorithm to generate the data was designed to ensure that solutions with cell counts from near zero to nearly fully glazed would be generated. The absolute error is larger still for the retrained model: despite this, the Spearman correlation was better with retraining. This indicates that, although the error does vary with the amount of glazing, the ability of the model to rank solutions by energy use is still strong. Ranking is all that is required for the optimisation process (which simply proceeds on the basis of whether one solution is better than another, not by how much), so the surrogate is still effective for all solutions in the population. That the algorithm with

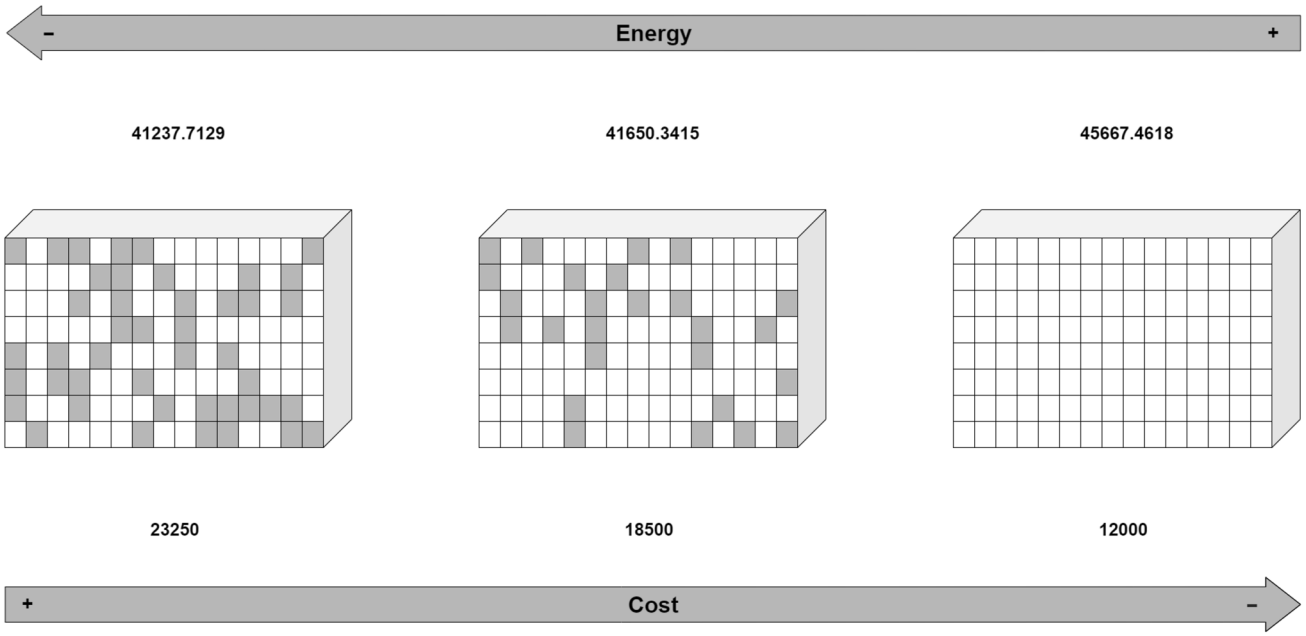

Fig. 5 Samples of the solutions from the Pareto front from the 100 000 evaluation optimisation run having the highest hypervolume

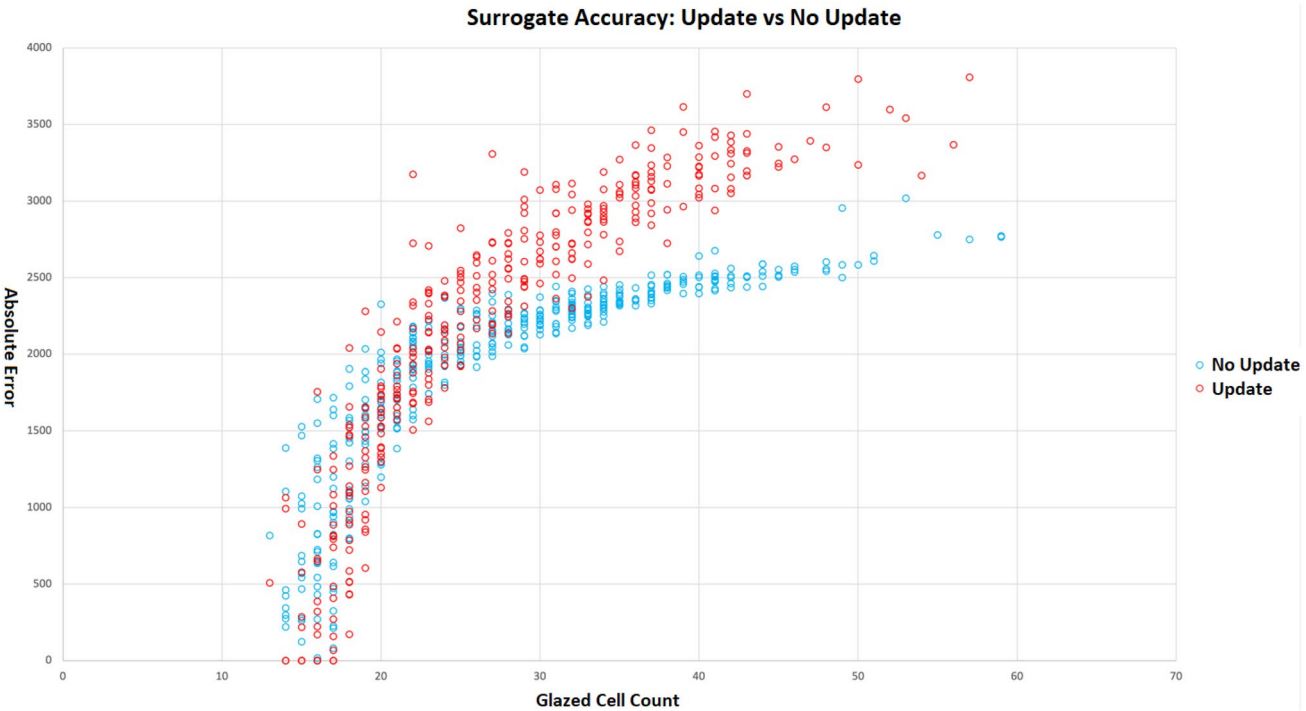

Fig. 6 Samples of the solutions from the Pareto front from the 100 000 evaluation optimisation run having the highest hypervolume

model updates is better able to rank solutions with higher glazing counts coincides with the improved performance of that algorithm in Fig. 4, where the low energy (high cost/high glazing count) end of the Pareto front dominated that of the algorithm without retraining. It is also possible that the surrogate error partly explains the improved performance: the surrogate "smoothes" the search landscape [3] allowing the NSGA-II to more efficiently explore it; further investigation in this direction would provide an interesting direction for future work.

6 Conclusions

While there have been many examples of surrogate models being used within optimisation processes for building designs, typically one model type is chosen and applied. The first contribution of this paper is the description and demonstration of a systematic process for selecting and tuning an appropriate surrogate model, in which a sample of solutions spanning the search space are generated, and models then fitted to these and compared.

Few examples exist of surrogates for binary-encoded, multi-objective, building optimisation problem such as forms our case study. Our case study also demonstrates the implementation of a surrogate integrated within NSGA-II for such a problem. We also developed a procedure to generate training data with a sufficient spread of solutions from little to extensive glazing. We have shown that the surrogate leads to a considerable speedup and improvement in solution quality for such a problem. Without the surrogate, in the median case solutions taking 43 746kWh energy and £18 500 capital cost were found in 46 259 s. With the surrogate, solutions taking 41 072kWh and £15 750 were found in 477 s without model updates and 40 420kWh and £16 000 in 702 s with updates. Thus, introducing limited updates to the model improves the quality of the final solutions returned by the optimisation process, but at the cost of a slightly longer run time. We also investigated the accuracy of the model for the optimal solutions found. Given the variability of model accuracy as increasing numbers of bits in the solutions are set true, the procedure to ensure a good spread of solutions for this training data was shown to be particularly particularly important. It was also noteworthy that despite the high mean absolute error in the model's predictions, its ability to rank solutions, a Spearman correlation of 0.407 between the predicted ranks and the true ranks as measured by the EnergyPlus simulation, was enough for the optimisation algorithm to locate high-quality Pareto fronts.

Future work in this area will include exploration of other binary encoded problems and a more systematic approach to updating the surrogate model.

Author contributions EV designed the methodology and initial experiments, performed the initial analysis, and co-wrote the manuscript. AB guided the research, extended the experimentation and analysis, and co-wrote the manuscript.

Funding Not applicable.

Data availability The datasets generated during this work cannot be shared due to permissions issues.

Declarations

Competing interests The authors declare no competing interests.

References

1. Brownlee AEI, Wright JA, Mourshed MM (2011) A multi-objective window optimisation problem. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO). GECCO '11, pp. 89–90. ACM, New York, NY, USA. https://doi.org/10.1145/2001858.2001910
2. Wright JA, Brownlee AEI, Mourshed MM, Wang M (2014) Multi-objective optimization of cellular fenestration by an evolutionary algorithm. J Build Perform Simul 7(1):33–51. https://doi.org/10.1080/19401493.2012.762808
3. Brownlee AEI, Woodward JR, Swan J (2015) Metaheuristic design pattern: Surrogate fitness functions. MetaDeeP Workshop, Proc. GECCO Companion. ACM Press, Madrid, Spain, pp 1261–1264
4. Li J-Y, Zhan Z-H, Zhang J (2022) Evolutionary computation for expensive optimization: a survey. Mach Intell Res 19(1):3–23
5. Asadi E, Silva MG, Antunes CH, Dias L, Glicksman L (2014) Multi-objective optimization for building retrofit: a model using genetic algorithm and artificial neural network and an application. Energy Build 81:444–456
6. Aijazi AN, Glicksman LR (2019) Application of surrogate modeling to multi-objective optimization for residential retrofit design. In: Proceedings of the Symposium on Simulation for Architecture and Urban Design, pp. 1–7
7. Bamdad K, Cholette ME, Bell J (2020) Building energy optimization using surrogate model and active sampling. J Build Perform Simul 13(6):760–776
8. Bamdad Masouleh K, Cholette M, Guan L, Bell J (2017) Building energy optimisation using artificial neural network and ant colony optimisation. In: Proceedings of the Australasian Building Simulation Conference 2017, The Australian Institute of Refrigeration, Air Conditioning and Heating (AIRAH). pp. 1–11
9. Brownlee AEI, Wright JA (2015) Constrained, mixed-integer and multi-objective optimisation of building designs by NSGA-II with fitness approximation. Appl Soft Comput 33:114–126. https://doi.org/10.1016/j.asoc.2015.04.010
10. Chen L, Fang Q-s, Zhang Z-y (2010) Research on the identification of temperature in intelligent building based on feed forward neural network and particle swarm optimization algorithm. In: 2010 Sixth International Conference on Natural Computation, IEEE. vol. 4, pp. 1816–1820
11. Chen Y, Shi X (2023) Surrogate based multi-objective optimization for energy-saving building design. In: International Conference on Green Building, Civil Engineering and Smart City, Springer. pp. 311–318.

12. Eisenhower B, O'Neill Z, Narayanan S, Fonoberov VA, Mezić I (2012) A methodology for meta-model based optimization in building energy models. Energy Build 47:292–301
13. Goethals K, Couckuyt I, Dhaene T, Janssens A (2012) Sensitivity of night cooling performance to room/system design: surrogate models based on CFD. Build Environ 58:23–36. https://doi.org/10.1016/j.buildenv.2012.06.015
14. Gossard D, Lartigue B, Thellier F (2013) Multi-objective optimization of a building envelope for thermal performance using genetic algorithms and artificial neural network. Energy Build 67:253–260
15. Kalogirou SA (2004) Optimization of solar systems using artificial neural-networks and genetic algorithms. Appl Energy 77(4):383–405
16. Magnier L, Haghighat F (2010) Multiobjective optimization of building design using TRNSYS simulations, genetic algorithm, and artificial neural network. Build Environ 45(3):739–746
17. Tresidder E, Zhang Y, Forrester AIJ (2011) Optimisation of low-energy building design using surrogate models. In: Proceedings of the Building Simulation Conference, Sydney, Australia, pp. 1012–1016
18. Tresidder E, Zhang Y, Forrester A (2012) Acceleration of building design optimisation through the use of kriging surrogate models. Proceedings of building simulation and optimization, p. 1–8
19. Coello CAC, Brambila SG, Gamboa JF, Tapia MGC, Gómez RH (2020) Evolutionary multiobjective optimization: open research areas and some challenges lying ahead. Complex Intell Syst 6(2):221–236
20. Evins R (2010) Configuration of a genetic algorithm for multi-objective optimisation of solar gain to buildings. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), ACM Press, New York, NY, USA. pp. 2003–2006
21. Suga K, Kato S, Hiyama K (2010) Structural analysis of pareto-optimal solution sets for multi-objective optimization: an application to outer window design problems using multiple objective genetic algorithms. Build Environ 45(5):1144–1152
22. Caldas LG, Norford LK (2002) A design optimization tool based on a genetic algorithm. Autom Constr 11(2):173–184
23. DOE (1993) Simulation Research Group, DOE-2 Supplement – Version 2.1E. LBL-34946
24. Vukadinović A, Radosavljević J, Đorđević A, Protić M, Petrović N (2021) Multi-objective optimization of energy performance for a detached residential building with a sunspace using the NSGA-II genetic algorithm. Sol Energy 224:1426–1444
25. Wright J, Mourshed M (2009) Geometric optimization of fenestration. In: Proceedings of the IBPSA Building Simulation Conference, Glasgow, Scotland, pp. 920–927
26. Westermann P, Evins R (2019) Surrogate modelling for sustainable building design-a review. Energy Build 198:170–186
27. Ghaderian M, Veysi F (2021) Multi-objective optimization of energy efficiency and thermal comfort in an existing office building using NSGA-II with fitness approximation: a case study. J Build Eng 41:102440
28. Yigit S (2021) A machine-learning-based method for thermal design optimization of residential buildings in highly urbanized areas of Turkey. J Build Eng 38:102225
29. Bre F, Roman N, Fachinotti VD (2020) An efficient metamodel-based method to carry out multi-objective building performance optimizations. Energy Build 206:109576
30. Zou Y, Lou S, Xia D, Lun IY, Yin J (2021) Multi-objective building design optimization considering the effects of long-term climate change. J Build Eng 44:102904
31. Chegari B, Tabaa M, Simeu E, Moutaouakkil F, Medromi H (2022) An optimal surrogate-model-based approach to support comfortable and nearly zero energy buildings design. Energy 248:123584
32. Yue N, Li L, Morandi A, Zhao Y (2021) A metamodel-based multi-objective optimization method to balance thermal comfort and energy efficiency in a campus gymnasium. Energy Build 253:111513
33. Safarzadegan Gilan S, Goyal N, Dilkina B (2016) Active learning in multi-objective evolutionary algorithms for sustainable building design. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), pp. 589–596
34. Østergård T, Jensen RL, Maagaard SE (2018) A comparison of six metamodeling techniques applied to building performance simulations. Appl Energy 211:89–103
35. Crawley DB, Lawrie LK, Winkelmann FC, Buhl WF, Huang YJ, Pedersen CO, Strand RK, Liesen RJ, Fisherm DE, Witte MJ, Glazer J (2001) EnergyPlus: creating a new generation building energy simulation program. Energy Build 33(4):319–331
36. Spon (2011) SPON'S Architects and Builders Price Book. Spon Press, London
37. Witten IH, Frank E, Hall MA (2011) Chapter 5 - credibility: Evaluating what's been learned. In: Witten, I.H., Frank, E., Hall, M.A. (eds.) Data Mining: Practical Machine Learning Tools and Techniques (Third Edition), Third edition edn. The Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann, Boston. pp. 147–187. https://doi.org/10.1016/B978-0-12-374856-0.00005-5
38. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197
39. Knowles J (2005) A summary-attainment-surface plotting method for visualizing the performance of stochastic multiobjective optimizers. In: Proceedings of the Conference on Intelligent System Design and Applications. ISDA '05, IEEE, Washington, DC, USA. pp. 552–557. https://doi.org/10.1109/ISDA.2005.15
40. Zitzler E (1999) Evolutionary algorithms for multiobjective optimization: Methods and applications. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland
41. Knowles J, Corne D (2002) On metrics for comparing nondominated sets. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC), **1**, 711–716. https://doi.org/10.1109/CEC.2002.1007013