Analysing Optimisation Data for Multicriteria Building Spatial Design

Koen van der Blom[1](✉), Sjonnie Boonstra[2], Hèrm Hofmeyer[2], and Michael Emmerich[1]

[1] Leiden Institute of Advanced Computer Science, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
{k.van.der.blom,m.t.m.emmerich}@liacs.leidenuniv.nl

[2] Department of the Built Environment, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
{s.boonstra,h.hofmeyer}@tue.nl

Abstract. Domain experts can benefit from optimisation simply by getting better solutions, or by obtaining knowledge about possible trade-offs from a Pareto front. However, just providing a better solution based on objective function values is often not sufficient. It is desirable for domain experts to understand design principles that lead to a better solution concerning different objectives. Such insights will help the domain expert to gain confidence in a solution provided by the optimiser. In this paper, the aim is to learn heuristic rules on building spatial design by data-mining multi-objective optimisation results. From the optimisation data a domain expert can gain new insights that can help engineers in the future; this is termed *innovization*. Originally used for applications in mechanical engineering, innovization is here applied for the first time for optimisation of building spatial designs with respect to thermal and structural performance.

Keywords: Data analysis · Building spatial design · Multicriteria optimisation · Mixed integer optimisation · Evolutionary algorithms

1 Introduction

During early phases of building design, decisions are made that significantly influence the quality of the final design. As such, optimising during the early stages can have substantial benefits. One of the first design steps entails capturing the building spatial design (BSD). Since numerous disciplines are involved in building design, multiple objectives have to be considered in the optimisation process as well. Here the focus lies on the structural and thermal performance.

Up till now, a mixed-integer representation was defined for the BSD problem in [6,8]. Based on this representation, multi-objective evolutionary algorithms

K. Deb et al. (Eds.): EMO 2019, LNCS 11411, pp. 671–682, 2019.
https://doi.org/10.1007/978-3-030-12598-1_53

have been devised, along with specialised operators in [4,5]. Further, the application of the hypervolume indicator gradient [11], to improve local search, was studied in [3], which resulted in a considerable amount of optimisation data.

Despite all this progress, the transfer of an optimisation result to a design expert is not merely a matter of stating "this solution is better than the previous one". The design expert needs to be convinced that the optimisation result is based on sensible design rules. Therefore, the optimisation result needs to be made explainable. In addition to being able to provide an optimised design, and being able to explain why it works well, it may also be possible to learn new design rules from an optimised design. Once the designer has obtained a set of proven design rules, they may apply these to similar problems, without having to endure another lengthy optimisation process. Additionally, such rules can also be integrated in co-evolutionary design algorithms like those considered in [7].

The process of learning innovative design rules from optimisation data was introduced in [10], and termed *innovization*. This concept has since been applied to a variety of problems such as clutch brake design in [10], and truss design in [1]. Later, the learning process was interleaved with the optimisation process in [13], and further automated in [1,9]. Furthermore, in [2] it was studied how an optimiser learns new concepts over time. Here it is investigated if simple techniques used to verify optimisation results may also lead to innovative insights.

This work is a first step in applying innovization in BSD. The following contributions are made: Optimisation results are verified through data analysis of a subset of the 800,000 solutions found by multi-objective optimisation in [3]. Handling a dataset of this size also results in new challenges. With this in mind, here simple and computationally inexpensive analysis techniques are applied.

From here on, this paper first introduces the problem of finding heuristic rules for building spatial design in Sect. 2. Following that, in Sect. 3 features are defined to enable the discovery of such rules. The preparation of the considered dataset is then described in Sect. 4. Section 5 evaluates the results from analysis of the data, and the implications that follow. Finally, Sect. 6 briefly summarises the study, draws conclusions, and proposes possible directions for future work.

2 Problem

Building spatial design (BSD) constitutes the arrangement of the internal and external divisions of a building. These divisions together form a number of *spaces*. A space is similar to a room. However, it also encapsulates concepts such as open kitchen-living room combinations that are not structurally separated, or hallways. This work considers the multi-objective optimisation of a BSD for structural and thermal performance. Structural performance is measured by means of compliance. This measure aggregates the total strain energy in the structural elements that constitute the structural model related to the BSD. Whereas thermal performance is taken as the total heating and cooling energy required to maintain a comfortable temperature in a given BSD.

For an optimised BSD to be used, the solution must be trusted by the design expert. To inspire such confidence in the optimised design, the optimised results should be made explainable. This can be achieved by learning heuristic design rules from the optimisation data. Given such rules, it becomes clear why the design is effective. Ideally, not only known rules that experts trust and understand are obtained, but also new insights. By combining known and new design rules it is possible for experts, and automated (e.g. co-evolutionary [7]) design systems, to improve their design process. These improved design processes can then be applied to similar problems, without another lengthy optimisation procedure.

3 Features

The supercube representation introduced in [6,8] is a mixed-integer representation of the building spatial design (BSD) problem, consisting of binary and positive real numbers. Raw data in this format is difficult to interpret in terms of building properties, making it difficult to learn directly from this data. To ease this process, this section introduces elementary features that allow building engineers to characterise a BSD. Such features are necessarily domain specific. However, the same process may be applied in other domains.

Since the supercube representation is key to understanding the dataset and features it is briefly introduced in the following. Since it is used for BSD, the supercube representation considers a number of spaces that together form the BSD. Each space is defined as a cuboid (3D rectangle), such that the whole building consists of rectangular surfaces, like in Fig. 1. Additional constraints ensure that the floors of all spaces are connected with the soil via other spaces, that is, in the given representation no floating or overhanging spaces may exist.

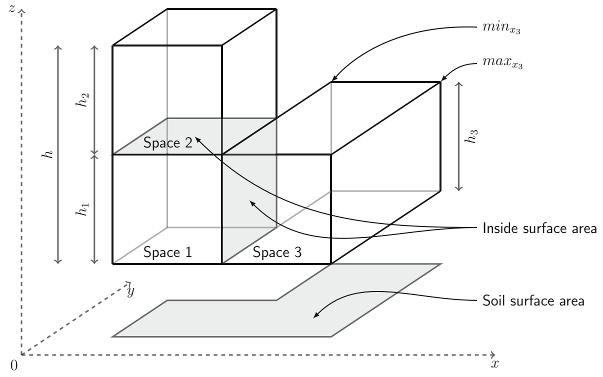

Fig. 1. Example building spatial design, annotated with a selection of features

All considered features are listed in Table 1 with their definitions and explanations. Except for the last three, all other features are computed both for the building and for individual spaces. Since the ordering of spaces is arbitrary,

Table 1. Features, definitions, and explanations

Feature	Definition	Explanation
vol	$w \times d \times h$	Volume of the space, or sum of spaces for the full BSD
short	$min(w, d)$	Shortest horizontal edge, indicator of span
long	$max(w, d)$	Longest horizontal edge, indicator of span
height	$max_z - min_z$	Height of the space or the full BSD
out	$sum(out\_area)$	Outside surface area, indicator of energy flow
in	$sum(in\_area)$	Inside surface area, indicator of energy flow
soil	$sum(soil\_area)$	Soil (ground floor) surface area, indicator of spread
horz	$sum(horz\_area)$	Horizontal surface area, indicator of total wall area
vert	$sum(vert\_area)$	Vertical surface area, indicator of floor and roof area
in_out	$in/(in + out)$	Ratio between inside- and outside surface area
out_vol	out/vol	Ratio between outside surface area and volume
long_short	$long/(long + short)$	Ratio between longest- and shortest horizontal edge
meanh	$sum(h \times roof\_area)/soil$	Mean height of the building
meanh_h	$meanh/height$	Ratio between the mean height and the height
height_soil	$height/soil$	Ratio between the height and the soil area

including values for each of them in the feature set would be of little use. Therefore, statistics are taken over all spaces in a building for each feature. In particular, the min, max, mean, median, range, standard deviation and Gini index (average deviation from the mean) are considered. Since the last three features in Table 1 do not make sense for individual spaces (e.g. mean height of a space is equal to its height), they are only computed for the building as a whole.

Values for w, d, h are found by taking $max_* - min_*$, where $*$ corresponds to x, y, z respectively. In other words, they are simply the distance between the minimal and maximal coordinates of a given dimension. For example, the min_x and max_x of space 3 are marked in Fig. 1. Note that these values are computed for the full design, as well as for individual spaces, as indicated for height in Fig. 1 with h for the complete BSD, and h_1, h_2, h_3 for each space.

To differentiate between various surfaces, the following surface area definitions are used. First, to distinguish between different locations of the surfaces, a non-overlapping division is made between inside (in_area), outside (out_area), and soil ($soil\_area$) surface area. Exterior surfaces are considered as outside, while interior surfaces are considered as inside. The ground floor which connects with the soil is excluded from the outside surface area, and taken as soil surface area. In Fig. 1, examples of inside and soil surface area are highlighted (the rest is outside surface area). Second, to distinguish between walls and floors/ceilings, a division between horizontal ($horz\_area$) and vertical ($vert\_area$) surface area is made. The horizontal surface area includes all floors and ceilings, so also the ground floor, while the vertical surface area consists of all walls, regardless of them being interior or exterior. Finally, the $roof\_area$ considered for $meanh$ is a part of the roof area in the BSD positioned at equal height.

Note that when considering a building as a whole, each surface is counted only once per considered distinction (e.g. horizontal/vertical). However, on the space level, surfaces are sometimes counted twice. That is, for two neighbouring spaces, both count their connecting surface as being part of, for instance, their horizontal surface area. As a result, the sum of the surface areas of all spaces is not (necessarily) equal to the total surface area of the building.

A few different features measure the same thing. For instance, the outside surface area of a building has an equal distribution (but not value) to the mean outside surface area of the spaces. Despite this, such features are kept to simplify data processing. In the analysis, only one representative should be used for these equivalent features, unless the differing values provide additional insights.

Additionally, some features may result in distributions similar to each other. This is particularly common for the range, standard deviation, and Gini index. However, even small differences may make one of them more valuable in distinguishing between solution classes than the other. Since, a priori, it is not known which is more useful in which situation, all of them are included.

Finally, it is noted that NAN values may appear in a few cases. Some spaces may be disconnected (meaning they do not share a wall with another space). As a result, it can occur in a building design that none of the spaces has a neighbour, from which it follows that their inside surface area is zero. In these cases, the Gini indices of the interior surface area, and of the ratio between inside and outside surface areas will be undefined and marked as NAN (the Gini index divides by the sum of the set of spaces, which is zero in this case). However, since these are very low quality solutions, they are not considered in the analysis in the rest of the paper. This will become clear in the next section.

4 Data Preparation

In order to learn heuristic rules for building spatial design, the dataset from the optimisation experiments in [3] is used. The dataset is a Pareto front and an archive from a building design optimisation that aimed for a BSD consisting of three spaces, with a total volume of 300 m^3. Note that while these may seem like simple BSDs, they already require 9 continuous and 81 binary variables to encode with the supercube representation [6,8], leading to a large search space. The optimisation runs resulted in a dataset of around 800,000 solutions. Here the data is prepared for analysis in the following five steps. First, classes are defined to enable the discovery of different qualities in different groups of solutions. Second, the non-dominated (ND) set is identified. Third, the knee point solution is identified. Fourth, solutions are assigned labels to link them to a class, based on the previously identified ND set and knee point. Fifth, a procedure is described to equalise the number of solutions in each class for those analysis techniques that demand this. Note that all steps are defined such that they should at least be generalisable for two-dimensional convex Pareto fronts.

To be able to learn from the features defined in the previous section, the data is split into different classes. This is accomplished based on objective values, rather than features. Classification based on objective values allows for the verification of the optimisation procedure: Do design experts agree that the designs with good objective values are indeed good? In addition, it is often a combination of features that indicate a certain quality in the BSD, making feature based classification more complex. Further, by classifying on known good qualities of a BSD, finding innovative design rules would become very unlikely. Here, four categories of solutions are considered: the knee point area (KP), good in the compliance objective (F1), good in the heating/cooling energy objective (F2), and relatively bad solutions (BD). The aim is to data-mine for heuristic design rules that make it possible to differentiate between all of these distinct classes. For more objectives additional classes F* can be added as needed.

The classification considers two primary aspects: (1) It should clearly distinguish between the classes in the objective space, and (2) It should be computationally efficient to enable processing of the large dataset of ca. 800,000 points. The computational efficiency should also allow the proposed methods to generalise to larger BSDs than those considered here.

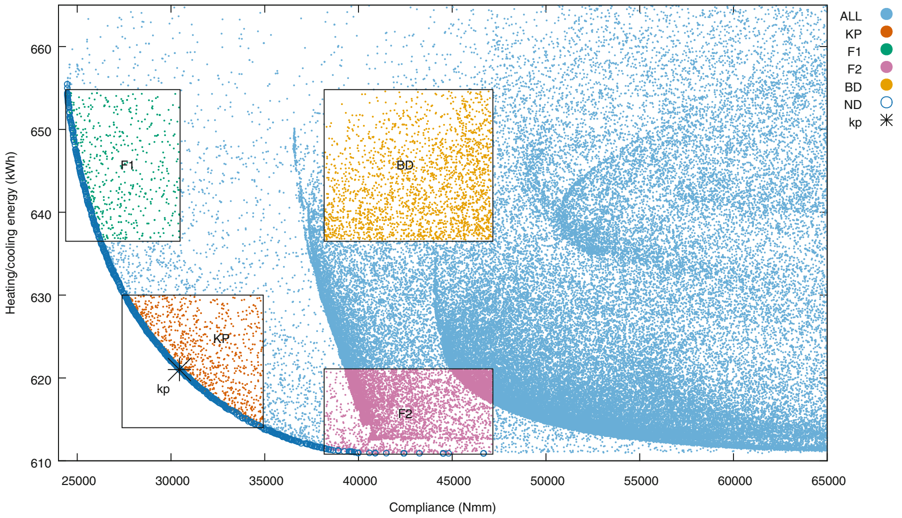

Fig. 2. Division of data into different classes: All points (ALL), knee point area (KP), objective one (F1), objective two (F2), bad solutions (BD); relative to the non-dominated set (ND), and the knee point (kp). A subset of the full dataset is shown

Since the considered classes are defined based on the non-dominated (ND) set and the knee point, these have to be identified first. For ND set computation the well-known log-linear time algorithm based on sorting is employed [12]. Based on the ND set, the knee point is derived as follows. First the objective values of the ND set are normalised to a $[0, 1]$ range, where outliers beyond 1.5 times the interquartile range are set to the appropriate boundary value. Next, the Euclidean distance to the origin $(0, 0)$ is computed for each normalised ND point. The point with the smallest distance is then taken as the knee point (indicated with ‘kp’ in Fig. 2), which is a reasonable approximation for the given dataset.

The data is then classified based on the knee point $p = (p_1, p_2)$, and the ND set. For this, the ND set is first reduced to the ND points that were not considered an outlier after normalisation, but the non-normalised values are used. In order to classify in a computationally efficient manner, each class is defined by a bounding box. These bounding boxes are found based on the range of the ND set in objective one r_1, and objective two r_2. For the knee point area class (KP) the lower bound of the box is set to $(0, 0)$, while the upper bound is set to $(p_1 + r_1 \times 0.2, p_2 + r_2 \times 0.2)$. For class F1 a lower bound of $(p_1 + r_1 \times 0.35, 0)$, and an upper bound of $(p_1 + r_1 \times 0.75, p_2)$ are taken. Similarly, F2 is found with the bounds $(0, p_2 + r_2 \times 0.35)$, and $(p_1, p_2 + r_2 \times 0.75)$. Lastly, BD uses the bounds $(p_1 + r_1 \times 0.35, p_2 + r_2 \times 0.35)$, and $(p_1 + r_1 \times 0.75, p_2 + r_2 \times 0.75)$. Following this, points are assigned a label based on the box they fall in. Any remaining unlabelled points are excluded from the analysis.

The result of the classification process is visualised in Fig. 2. Note that gaps are left between the different classes to improve the chances of being able to distinguish between them. If the classes would directly neighbour each other, points on the border are likely to have very similar features. This would impede learning what makes a solution perform well (or not) in one objective or the other. Future work could study how these points can be included in the analysis.

In Fig. 3 a randomly selected example of a BSD is shown for each class. Although the examples for KP and F1 look similar, the design for F1 is far more elongated. This result can be expected, as the short spans (here coupled with elongated spaces) allow F1 designs to reduce the strain energy, at the cost of a larger surface area, reducing thermal efficiency. The F2 design shows the reverse, with a much more compact design. Finally, the BD design is not as well arranged in the spatial sense, and shows relatively poor performance in both objectives.

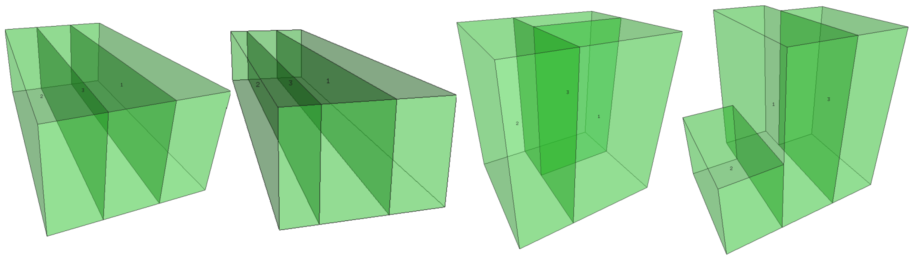

Fig. 3. Typical examples of the different classes, from left to right: KP, F1, F2, BD

After processing the dataset[1] 70,088 of the 806,430 solutions are labelled. With 5978 KP, 3400 F1, 48,482 F2, and 12,228 BD solutions respectively. Given the mixed-integer nature of the representation, multiple discrete subspaces can be seen in Fig. 2, indicated by the different curves. Since the dataset is not homogeneous, the resulting classes do not have an equal number of points. For

[1] The dataset is available under http://moda.liacs.nl/index.php?page=code.

some types of analysis, however, it is critical to have equally distributed classes. In such situations excess solutions are removed from the larger classes uniformly at random. In all other situations, all labelled data is used.

5 Results

Two techniques are used for data analysis, box plots and decision trees. Box plots give insight into the distribution of feature data for different solution classes. As such, it may be possible to identify features that allow for a clear distinction between two or more classes. Further, the decision tree can provide information about distinguishing features as well, since it generates clear rules based on such features. Moreover, it gives confidence measures for the classification of solutions to different classes. Finally, by using the learned decision tree on new data, it is possible to validate whether those rules can indeed be used reliably.

5.1 Box Plots

To generate box plots all labelled data is used, with each feature normalised to a $[0, 1]$ range, without removing outliers. In the plots, each class is then visualised by an individual box, such that any differences become clearly visible.

In Fig. 4 a subset of the features is shown that appears to allow for a significant amount of distinction between the different classes. Notice, for example, how the mean of the most extended horizontal edge (long.mean) enables differentiation between objective one (F1), and objective two (F2).

Surprisingly the soil surface area (soil.mean), and the horizontal surface area (not in the figure) showed exactly the same distribution in all of their features. This occurs because all buildings considered in the labelled dataset are single-story buildings. For such single-story buildings, the horizontal surface area is equal to the soil surface area plus the roof area. Since these two areas are equal, the horizontal surface area is exactly twice the soil surface area, which results in their equal distributions. It appears then that in general single-story buildings have a good performance for the given objectives, even if they are not necessarily optimal. After all, the labelled solutions are all relatively close the Pareto front approximation. Naturally, this result may not generalise to designs with a larger number of spaces. This also indicates it may be interesting to include an even worse class of solutions in future analysis to see how things differ with even worse solutions. Additionally, a feature indicating the number of stories a BSD has could be useful as well in this case. Even if just to identify this type of situation more easily.

5.2 Decision Trees

In order to use decision trees to their full potential, the data should be equally distributed among the classes. As such, this is carried out as described previously. Since the smallest class contains 3400 solutions, the other classes are reduced to

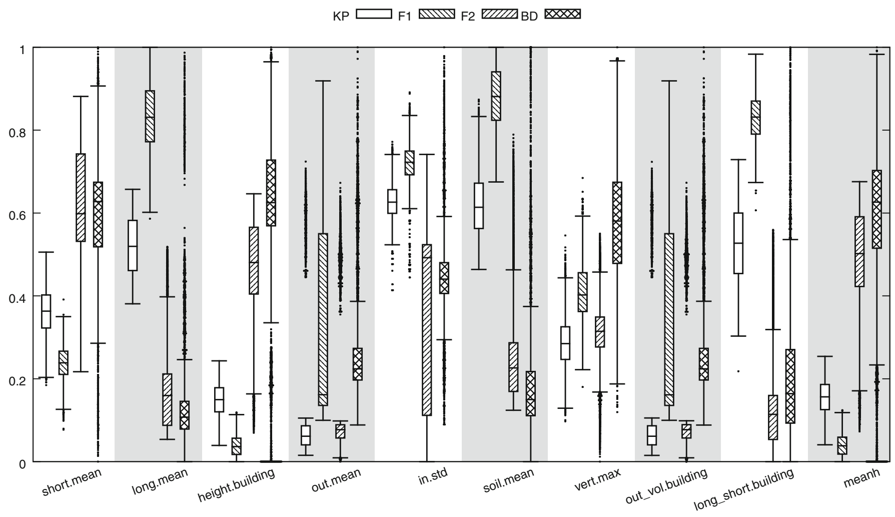

Fig. 4. Boxplot of a selection of distinguishing features

the same number of data points, resulting in a total of 13,600 solutions. This total is split into a training set of 10,200 solutions and a test set of 3400 solutions by sampling uniformly at random. Note that as a result, the representation of each class is not necessarily exactly equal in either of the training and test sets, but still sufficiently close. The training and test sets then consist of approximately 2550, respectively 850 solutions per class. Only labelled solutions are used, no normalisation is applied, and no outliers of individual features are removed. In the future it may be of interest to do the same study with unlabelled solutions to see if the generated rules generalise.

Given the prepared dataset, the decision tree in Fig. 5 was generated with the R package *rpart* [14]. From this figure, it can be found that the longest horizontal edge, the outer surface area, the ratio between the longest and shortest horizontal edge, and the ratio between the inner and outer surface area provide important information to distinguish between different classes of solutions.

These rules indicate properties of a building that contribute to qualities present in different solution classes. The first split shows that relatively long buildings (long.build) are likely to be efficient in objective one (compliance). This split intuitively makes sense, since buildings that are more stretched out are likely to have short spans. Note that this is under the assumption that not just the building is stretched out, but the spaces as well (e.g. F1 in Fig. 3).

In the other branch buildings are a bit more compact. Additionally, it can be seen that buildings where the minimal ratio of the spaces between the longest and shortest horizontal edge (long_short.min) is relatively high, are very likely to be solutions in the knee point area. This indicates that although the building as a whole is more compact, the individual spaces remain somewhat elongated to balance between the two objectives. The primary split between low quality

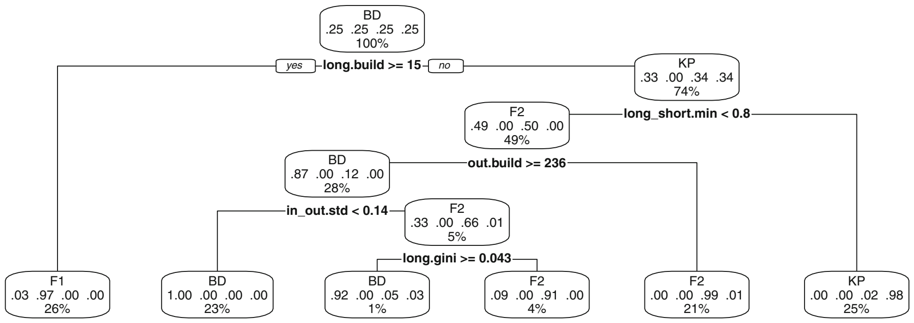

Fig. 5. Decision tree based on data

solutions and the second objective (energy) is made based on the outer surface area of the entire building (out.build). Since a larger outer surface area is an indicator of a more significant loss of energy to the outside, this appears to be a sensible rule. Further, these rules provide clear pointers on how to navigate towards the PF. It may be possible to incorporate this in specialised operators to speed up the optimisation process.

From the decision tree in Fig. 5 it appears classification of solutions is possible with high precision. To validate this, the tree was used to classify the 3400 solutions in the test set. Table 2 shows the resulting predictions. All assignments were made with a confidence of at least 90%, showing that it is possible to classify designs quite reliably. A particularly notable result is the classification of the majority of the solutions in the F2 and KP classes, which, for this dataset, is done with near perfect confidence. Not only does this provide confidence in the optimisation process, but these rules could even be useful during optimisation. By classifying new solutions based on these rules it may be possible to identify which solutions are more likely to perform well, such that expensive simulations might only be needed for those.

Table 2. Decision tree results on the test set. Columns relate to the predicted probability of belonging to a specific class, whereas rows refer to classes. Each cell then contains the number of solutions that belong to a solution class, with a particular probability.

Predictions	0.0000	0.0008	0.0046	0.0048	0.0051	0.0162	0.0276	0.0345	0.0483	0.0926	0.9074	0.9241	0.9655	0.9784	0.9949	0.9952
BD	1758	0	0	0	728	0	54	0	0	0	0	0	0	860	0	0
F1	1649	860	0	0	0	0	0	0	0	0	0	0	891	0	0	0
F2	891	0	0	748	0	860	0	0	54	0	119	0	0	0	728	0
KP	728	0	860	0	0	0	0	891	0	119	0	54	0	0	0	748

6 Conclusion

In summary, the results show that by using predefined features and decision trees, it is possible to apply innovization to large datasets from Pareto optimisation in building spatial design (BSD), and to obtain meaningful results from an engineering perspective. Furthermore, the obtained rules allow for high precision ($\geq$96%) classification of solutions.

Besides in generating insight, the design rules could also be useful in steering the multi-objective optimisation process. For future work, it would be interesting to investigate which moves in the optimisation process result in improvements. In other words, given an existing design, what changes to its features will, with high probability, result in an improved design. Furthermore, it may be possible to apply learned rules in co-evolutionary design processes [7]. Or, as mechanism to determine for which solutions to use expensive simulations during optimisation.

The current work analyses data for a specific type of building. To generalise the conclusions, the same methods should be evaluated on a larger variety of building types. Given the computationally efficient nature of the used approach, it is probable that larger BSDs can be handled. This must, however, still be verified. Additionally, currently only a subset of the optimisation data is labelled. As a result, it is unclear whether the learned rules generally allow the identification of, for instance, solutions that perform well in objective one. It may be the case that some areas of the objective space, that have not been considered here, have similar characteristics in some features. This should be studied in the future. A challenge here is how to do proper analysis with both sparse and dense areas in the objective space.

Based on first discussions with a design expert good heuristics are learned that accurately describe high quality BSDs. However, it remains difficult to foresee the consequences of changes in feature values with respect to the objective values. In order to improve this visual aids would be helpful. For instance, a slider controlling the weights of the structural and thermal objectives could be used to change the spatial design in real-time.

Acknowledgments. This work is part of the TTW-Open Technology Programme with project number 13596, which is (partly) financed by the Netherlands Organisation for Scientific Research (NWO).

References

1. Bandaru, S., Deb, K.: Automated innovization for simultaneous discovery of multiple rules in bi-objective problems. In: Takahashi, R.H.C., Deb, K., Wanner, E.F., Greco, S. (eds.) EMO 2011. LNCS, vol. 6576, pp. 1–15. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19893-9_1
2. Bandaru, S., Deb, K.: Temporal innovization: evolution of design principles using multi-objective optimization. In: Gaspar-Cunha, A., Henggeler Antunes, C., Coello, C.C. (eds.) EMO 2015. LNCS, vol. 9018, pp. 79–93. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15934-8_6

3. van der Blom, K., Boonstra, S., Wang, H., Hofmeyer, H., Emmerich, M.T.M.: Evaluating memetic building spatial design optimisation using hypervolume indicator gradient ascent. In: Trujillo, L., Schütze, O., Maldonado, Y., Valle, P. (eds.) NEO 2017. SCI, vol. 785, pp. 62–86. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-96104-0_3
4. van der Blom, K., Boonstra, S., Hofmeyer, H., Bäck, T., Emmerich, M.T.M.: Configuring advanced evolutionary algorithms for multicriteria building spatial design optimisation. In: 2017 IEEE Congress on Evolutionary Computation (CEC), pp. 1803–1810. IEEE (2017). https://doi.org/10.1109/CEC.2017.7969520
5. van der Blom, K., Boonstra, S., Hofmeyer, H., Emmerich, M.T.M.: Multicriteria building spatial design with mixed integer evolutionary algorithms. In: Handl, J., Hart, E., Lewis, P.R., López-Ibáñez, M., Ochoa, G., Paechter, B. (eds.) PPSN 2016. LNCS, vol. 9921, pp. 453–462. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45823-6_42
6. van der Blom, K., Boonstra, S., Hofmeyer, H., Emmerich, M.T.M.: A super-structure based optimisation approach for building spatial designs. In: Papadrakakis, M., Papadopoulos, V., Stefanou, G., Plevris, V. (eds.) VII European Congress on Computational Methods in Applied Sciences and Engineering - ECCOMAS VII, vol. 2, pp. 3409–3422. National Technical University of Athens, Athens (2016). https://doi.org/10.7712/100016.2044.10063
7. Boonstra, S., van der Blom, K., Hofmeyer, H., Emmerich, M.T.M.: Combined super-structured and super-structure free optimisation of building spatial designs. In: Koch, C., Tizani, W., Ninić, J. (eds.) Digital Proceedings of the 24th EG-ICE International Workshop on Intelligent Computing in Engineering, pp. 23–34. Curran Associates Inc., Red Hook (2017)
8. Boonstra, S., van der Blom, K., Hofmeyer, H., Emmerich, M.T.M., van Schijndel, J., de Wilde, P.: Toolbox for super-structured and super-structure free multidisciplinary building spatial design optimisation. Adv. Eng. Inform. **36**, 86–100 (2018). https://doi.org/10.1016/j.aei.2018.01.003
9. Deb, K., Bandaru, S., Greiner, D., Gaspar-Cunha, A., Tutum, C.C.: An integrated approach to automated innovization for discovering useful design principles: case studies from engineering. Appl. Soft Comput. **15**, 42–56 (2014). https://doi.org/10.1016/j.asoc.2013.10.011
10. Deb, K., Srinivasan, A.: Innovization: innovating design principles through optimization. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, GECCO 2006, pp. 1629–1636. ACM, New York (2006). https://doi.org/10.1145/1143997.1144266
11. Emmerich, M., Deutz, A.: Time complexity and zeros of the hypervolume indicator gradient field. In: Schuetze, O., et al. (eds.) EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation III. Studies in Computational Intelligence, pp. 169–193. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-319-01460-9_8
12. Kung, H.T., Luccio, F., Preparata, F.P.: On finding the maxima of a set of vectors. J. ACM **22**(4), 469–476 (1975). https://doi.org/10.1145/321906.321910
13. Ng, A.H.C., Dudas, C., Boström, H., Deb, K.: Interleaving innovization with evolutionary multi-objective optimization in production system simulation for faster convergence. In: Nicosia, G., Pardalos, P. (eds.) LION 2013. LNCS, vol. 7997, pp. 1–18. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-44973-4_1
14. Therneau, T., Atkinson, B.: rpart: Recursive Partitioning and Regression Trees (2018). https://CRAN.R-project.org/package=rpart. r package version 4.1-13