Towards an interpretable fuzzy approach to experimental design

Olivier Rousselle[0000−0002−7494−2135], Jean-Philippe Poli[0000−0003−2429−6187], and Nadia Ben Abdallah[0000−0002−9884−2745]

Université Paris-Saclay, CEA List, F-91120, Palaiseau, France
olivier.rousselle@cea.fr
jean-philippe.poli@cea.fr
nadia.ben-abdallah@cea.fr

Abstract. We present in this article an interpretable fuzzy approach to experimental design under constraints that can be used with few data. It is mainly intended (but not limited) to materials science. The goal is to provide experimentalists with an interpretable and explainable algorithm, allowing them to sample optimally the design space. We detail the different steps of our algorithm that consists in recommending the next experiment to perform and building a Sugeno fuzzy rule base. We then present some results on a toy and a real-world datasets. As the method is inspired from Bayesian optimization, we also compare the fuzzy approach to the Bayesian one.

Keywords: Active learning · experimental design · fuzzy rules · optimization · interpretability.

1 Introduction

Experimental sciences require exploring an often very large space of possibilities to approach an optimum. Different sampling methods are used in experimental research, such as random sampling, factorial sampling, the response surface method, Bayesian optimization [6], the optimal coverage algorithm, etc. Without loss of generality, we take as example materials discovery, which aims at producing high-performance materials for a targeted use. These materials are generally produced from a mixture of initial compounds subjected to a certain manufacturing process. In recent years, Artificial Intelligence has accelerated innovation in this area [10]. In this context, the objective of our work is to develop a method based on fuzzy logic applied to experimental design. We define our problem as testing different sets of parameters (i.e., the composition of a material and the process parameters) to maximize a given property (e.g., the robustness of this material). In particular, we are interested in finding an automatic method to iteratively sample experimental parameters optimally.

Our motivation is to provide a tool to help experimentalists determining what are the next sets of parameters to test and that works with few data. We aim at reducing the number of experiments to achieve a target performance,

to both reduce the waste of raw materials and converge more quickly towards an innovative material. To keep the Human in the loop, we pay attention to interpretability, giving clues to the user about the choice of the next experimental setup. Indeed, explainability/interpretability aims at making the operation and results of the model more intelligible and transparent to humans, to strengthen confidence in decision-making and its acceptability.

The paper is structured as follows. The next section gives an overview of the approach. Section 3 describes the regression method that approximates the objective function. Section 4 explains the process behind the selection of the next experiment to perform. We show the results and the comparison with Bayesian optimization in Section 6. As our approach is dedicated to Human experts, section 7 introduces how the end-user is considered. Finally, we draw some conclusions and perspectives.

2 Approach overview

To satisfy the needs of experimentalists, we designed an approach based on these principles:

- It must implement an adaptive sampling;
- It must be able to combine learning from data and expert knowledge;
- It must be robust, i.e. small changes in the initial points should not lead to large changes in the results and predictions;
- It must be interpretable, i.e. the steps and results of the model must be intelligible to humans, to strengthen confidence in decision-making;
- It must be scalable, i.e. able to work on high-dimensional mixture problems.

To meet these prerequisites, we have developed an approach whose different steps are detailed in Fig. 1. The selected experimental point is the one that maximizes the sampling score (see section 4). This process is repeated over several iterations, each iteration corresponding to one experiment.

In this work, we consider the material properties to have real values. We thus need to build a fuzzy rule base for regression (see section 3). We chose to use a Sugeno approach for its computational efficiency. With this choice, we favored the performances of the prediction over the interpretability. To compensate, we propose a method to extract a more interpretable model surrogate (section 5).

Note: the notations used in this article are detailed in the Appendix.

3 Fuzzy clustering-based regression algorithm

Let us consider the problem of predicting the property of a material, denoted $\hat{y}$, for each point of the input space. We based our approach on several previous works [4, 13] that share the use of a clustering method as a first step in the rule induction. Our method differs slightly in the sense it is intended for eventually high dimensional problems. Moreover, we improved the way membership functions are learnt and the computation of the regression coefficients for the Sugeno rule conclusions.

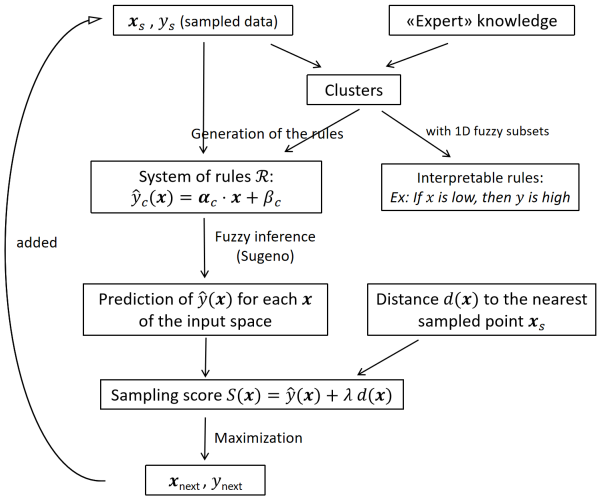

Fig. 1. The different steps of the proposed approach.

3.1 Clustering

First, we perform a multidimensional fuzzy clustering of the points already sampled, denoted $\boldsymbol{x}_s$, to highlight different groups. We remind the particularity of fuzzy clustering is that a point may belong to several clusters, with different membership degrees. The fuzzy clustering applies to both the input and output variables, and each cluster c includes a center noted $\boldsymbol{m}_c$. We note C the set of clusters. The number of clusters n_c is often an hyperparameter, whose value must be set by the user. The number of clusters can remain constant during the different iterations of the experiment or can increase regularly in stages regarding the number of points already sampled.

We are now interested in measuring the membership degree of a given point $\boldsymbol{x}$ in the input space to each cluster, denoted $\mu_c(\boldsymbol{x})$. We chose to build a membership function that depends on several input variables. The benefit is to take into account the interaction between variables and to get a multi-dimensional strong partition:

$$\forall \boldsymbol{x} \ , \ \sum_{c \in C} \mu_c(\boldsymbol{x}) = 1. \tag{1}$$

The membership degrees can be calculated using the FCM (Fuzzy Clustering Means) algorithm [3], i.e. minimizing the objective function [12]:

$$\sum_{\boldsymbol{x}_s} \sum_{\boldsymbol{m}_c} \mu_c(\boldsymbol{x}_s) ||\boldsymbol{x}_s - \boldsymbol{m}_c||^2, \tag{2}$$

with $\boldsymbol{m}_c$ the centers of each cluster and $\mu_c(\boldsymbol{x}_s)$ the coefficient of membership of the point $\boldsymbol{x}_s$ to the cluster c. The degrees of membership obtained are:

$$\mu_c(\boldsymbol{x}_s) = \sum_{c' \in C} \left(\frac{||\boldsymbol{x}_s - \boldsymbol{m}_c||^2}{||\boldsymbol{x}_s - \boldsymbol{m}_{c'}||^2} \right)^{-\frac{2}{m-1}}, \tag{3}$$

with $m \in \ ]1; +\infty[$ the "fuzzifier" parameter that influences the fuzziness of the partition. It ranges from 1 excluded (sharp partition) to $+\infty$ (totally fuzzy partition). Generally, m is chosen equal to 2. Each point $\boldsymbol{x}$ is then assigned a degree of membership to each cluster. Examples of membership degrees are illustrated in the ternary diagram in Fig. 2 with 3 clusters and with 3 input variables x_1, x_2, x_3. To read the ternary diagram, the red cross indicates, for example, the point $(x_1, x_2, x_3) = (0.55, 0.2, 0.25)$.

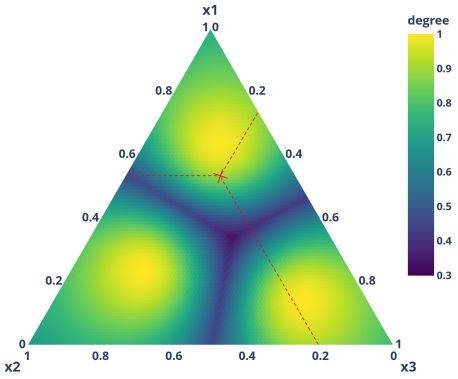

Fig. 2. Membership degrees - case study with 3 compositional variables and 3 clusters.

3.2 Generation of the fuzzy rules

Each cluster/region c is at the origin of a rule R. Each rule R is characterized by its multidimensional antecedent region and its regression coefficients $(\boldsymbol{\alpha}_c, \beta_c)$, in the form:

$$\hat{y}_c(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_{c_i} x_i + \beta_c = \boldsymbol{\alpha}_c \cdot \boldsymbol{x} + \beta_c. \tag{4}$$

The coefficients $(\boldsymbol{\alpha}_c, \beta_c)$ of each rule are determined by the optimization process described later in this article.

Expert rule A rule can also come from human expert knowledge or knowledge from the literature. For example, an expert can indicate another region of interest by adding another center $\boldsymbol{m}_c$. This cluster will also generate a rule characterized by its antecedent region and its regression coefficients $(\boldsymbol{\alpha}_c, \beta_c)$. This rule will then be added to the system of rules already generated from the data.

3.3 Optimization of regression coefficients

The output of our model is generated by merging the linear functions of the rules through a Takagi Sugeno Kang (TSK) model:

$$\hat{y}(\boldsymbol{x}) = \sum_c \mu_c(\boldsymbol{x})\ \hat{y}_c(\boldsymbol{x}), \tag{5}$$

with $\hat{y}_c(\boldsymbol{x}) = \boldsymbol{\alpha}_c \cdot \boldsymbol{x} + \beta_c$.

We determine the optimal regression coefficients $(\boldsymbol{\alpha}_c, \beta_c)$ by minimizing the squared deviation between the predicted values of the points already sampled $\hat{y}$ and their actual values y. We therefore seek to minimize the loss function:

$$\begin{aligned} L(\boldsymbol{\alpha}, \beta) &= \sum_{\boldsymbol{x}_s} \left(\hat{y}(\boldsymbol{x}_s) - y(\boldsymbol{x}_s)\right)^2 \\ &= \sum_{\boldsymbol{x}_s} \left(\sum_c \mu_c(\boldsymbol{x}_s)\ (\boldsymbol{\alpha}_c \cdot \boldsymbol{x}_s + \beta_c) - y(\boldsymbol{x}_s) \right)^2 . \end{aligned} \tag{6}$$

3.4 Prediction of the output value for each entry point

For each input point $\boldsymbol{x}$, we predict the output value $\hat{y}$. Figure 3 shows a ternary diagram illustrating an example of predicted values $\hat{y}$ obtained with our algorithm for a case with 3 input variables.

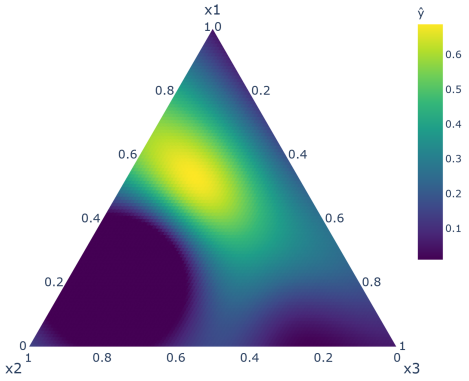

Fig. 3. Example of predicted values $\hat{y}$ for a case with 3 input variables.

We can evaluate the accuracy of the inference model using the points already sampled. For each sampled point, we predict the output value $\hat{y}$ using the inference system, which we compare to the real value y. We can then calculate the mean square deviation RMSE and the coefficient of determination R^2 to judge the quality of the model. The results are presented in part 6.

Moreover, the authors of [13] have showed that the fuzzy clustering-based regression algorithm with gradient descent is more efficient than neural networks for a regression problem (prediction of the shape of a function) for the 1D case, and with much less hyperparameters.

4 Selection of the next experiment

The proposed method also chooses the next point to sample in a deterministic manner, as a compromise between exploitation and exploration. Exploitation means favoring regions with high potential, i.e. with high predicted values $\hat{y}$, while exploration means exploring the regions not yet sampled.

We need to introduce a variable that captures this exploration part. A natural variable for that is the Euclidean distance to the nearest sampled point, denoted d. The computation of the distances between each point in the input space and the closest already sampled point is done using the KD-tree algorithm [9]. Then, for each input point, $\hat{y}$ and d are computed; $\hat{y}$ encodes the exploitation, and d encodes the exploration. To find a compromise between exploitation and exploration, we introduce a new variable called "sampling score" and denoted S:

$$S(\boldsymbol{x}) = \hat{y}(\boldsymbol{x}) + \lambda \, d(\boldsymbol{x}), \tag{7}$$

where $\hat{y}$ and d are here normalized, and where λ is an hyperparameter. λ can be chosen to be constant or it can change as the number of experiments carried out increases. Increasing λ will favor the exploration over the exploitation.

An illustration of S is presented in Fig. 4 for an example with 3 input variables. The areas in blue are the ones around the points already sampled, and the areas in yellow are the regions of interest. A network structure can be observed, due to the compromise between exploitation and exploration.

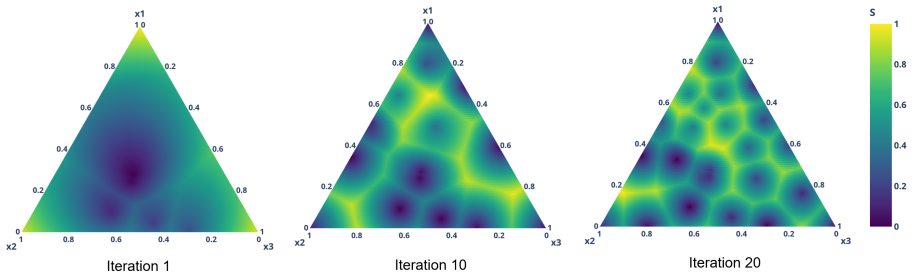

Fig. 4. Ternary diagrams illustrating the evolution of the sampling score S at different iterations, for a case with 3 input variables.

The next point proposed by our algorithm will be the one that maximizes $S(\boldsymbol{x})$. For example, the next point to test could be:

$$\{x_1 = 0.35 \ , \ x_2 = 0.5 \ , \ x_3 = 0.15\} \, . \tag{8}$$

Alternatively, our algorithm can also propose a region of interest to the experimentalist. This region includes all the points with a sampling score S higher than a given threshold (for example 0.9). For example, a region of interest could be:

$$\begin{cases} 0.3 \leq x_1 \leq 0.425 \\ 0.46 \leq x_2 \leq 0.535 \\ 0.11 \leq x_3 \leq 0.21 \end{cases} \tag{9}$$

and then the experimentalist will choose a point in this region. An explanation can also be given to justify the point/region proposed. For example: "this point/region is proposed to explore next to an area already exploited and that gave good results."

The proportion of exploitation / exploration of the next point tested can also be provided to the experimentalist:

$$p_{\text{exploitation}} = \frac{\hat{y}(\boldsymbol{x}_{next})}{\hat{y}(\boldsymbol{x}_{next}) + \lambda \; d(\boldsymbol{x}_{next})}, \tag{10}$$

$$p_{\text{exploration}} = 1 - p_{\text{exploitation}}. \tag{11}$$

Finally, constraints can be taken into account to restrict the input space; for example $x_2 < 0.5$. Experimentally, those constraints could be imposed by the experimental setup, for example to reflect the limits of the experimental machine or physico-chemical laws (such as the law of miscibility) in the case of a mixture of materials.

5 Interpretability

The benefits of having multidimensional rules vs rules based on 1D fuzzy sets include the possibility to control the desired number of rules, the avoidance of the explosion of the number of rules regarding the number of dimensions, and the capture of interactions between variables. However, multidimensional rules are less interpretable than the one-dimensional case. We can make the multidimensional system more interpretable by establishing different linguistic categories per variable (e.g. low/medium/high) and projecting the centers of each cluster onto each variable axis [11]. Our interpretability algorithm follows the following steps. First, we divide each input and output variable into different triangular fuzzy sets f, whose summits correspond to the centers of the clusters projected on each axis. If two fuzzy sets are too close (e.g. distance $<$ threshold distance), we merge them. Then, for a variable x_i ($i \in [1; N+1]$), by noting m_{c_i} the i^{th} component of the center $\boldsymbol{m}_c$, the subset associated with the cluster c is the one with the highest membership degree $\mu^f(m_{c_i})$.

A cluster will be associated with $N+1$ fuzzy sets/classes (one per variable). Figure 5 illustrates the system of interpretable rules obtained for a case with 1 input variable x and 3 clusters. The 1D rules are used as surrogates of the multidimensional rules to help the user understand this model.

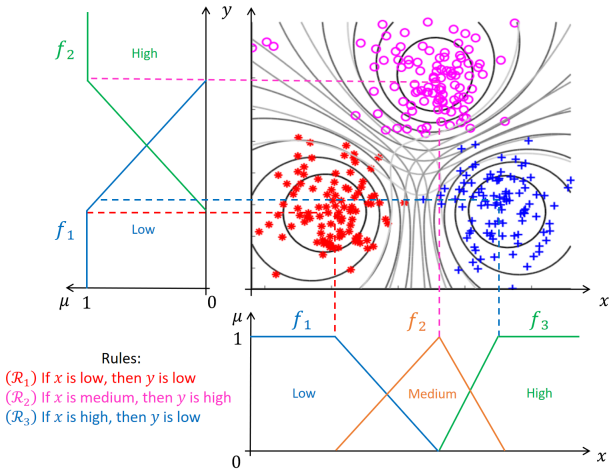

Fig. 5. Illustration of 3 clusters with FCM membership degrees represented with black level lines (taken from [7]) and fuzzy sets f_i of variables x and y . The association of each cluster center to these 1D fuzzy sets makes the rules more interpretable.

6 Experimental results

In this section, we present the results of different tests to characterize our algorithm. For reproducibility, we first give some details about the implementation.

6.1 Implementation considerations

To perform the multidimensional clustering, we hybridized two approaches: we use the hierarchical clustering method [8] to get the centers of the clusters and we use the end of the fuzzy c-means approach to determine the membership functions. Indeed, the hierarchical clustering has the advantage to be fully deterministic. Without loss of generality, we determined the number of clusters empirically for each dataset.

The optimization of the cost function Eq. 6 can be implemented by Trust Region Reflective (TRF) method [5], recursive least squares algorithm [4] or gradient descent [13]. We have used the TRF algorithm since it is a robust method, well suited to complex problems with nonlinear residuals, suitable for large sparse problems with bounds, and it does not need extra hyperparameters.

6.2 Toy datasets

We first evaluated our method on a toy dataset that we generated from a sinus function with possibly several inputs. A small number of inputs helps us to visualize the results to qualify them, while a large number helps to validate our algorithm.

Sinus function with 3 input variables We tested our algorithm for the following sinus objective function with 3 input variables:

$$f(\boldsymbol{x}) = \left| \prod_{k=1}^{3} \sin(k\pi x_k) \right|. \tag{12}$$

This objective function has been chosen because it presents a non-trivial shape with several maxima and one global maximum, with output values ranging between 0 and 1, and because it can be plotted in a ternary diagram (see Fig. 6). It also helps us to characterize the approach since we will never have a complete experimental plan from a real-world application. Each axis x_i contains 100 values from 0 to 1, so in total we have 5151 points in the input space. Our goal is to converge as fast as possible towards the optimal value. We choose initially 5 random points and we perform 50 iterations (with one point tested per iteration). The efficiency of our algorithm is measured with the criteria of the number of iterations M needed to reach 80% of the optimal value of y (which is 0.984 in our study case). Figure 6 shows the best y value obtained among the points tested until a given iteration; 80% of the optimal value is reached at the iteration $M = 24$.

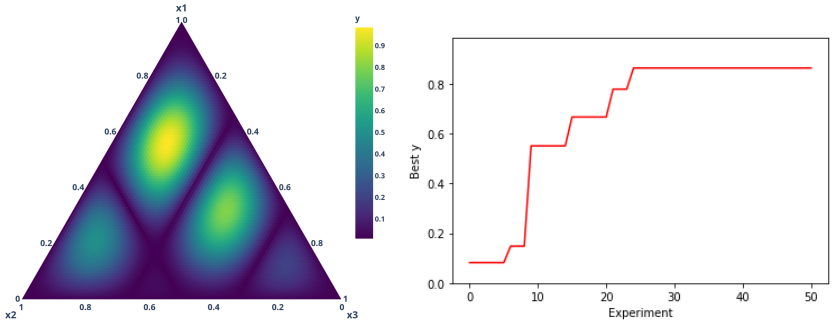

Fig. 6. Left: Ternary diagram of the objective function f given by eq (12). Right: Best value of y obtained for each iteration (i.e. experiment).

The proportion of exploitation/exploration of the point tested at each iteration is plotted in the histogram Fig. 7.

We observe, as expected, that the proportion of exploration decreases as the number of points sampled increases. Note that there are still some explorations even after a high number of experiments; indeed, it is better to continue exploring to avoid coming across a local maximum.

We then tested the sensibility of our approach to its initialization. Indeed, the number of experiments M needed to reach 80% of the optimal value depends on the relevance of the random initial points. We repeated the simulation 100 times where at each simulation we have 5 random initial points with $y < 0.1$

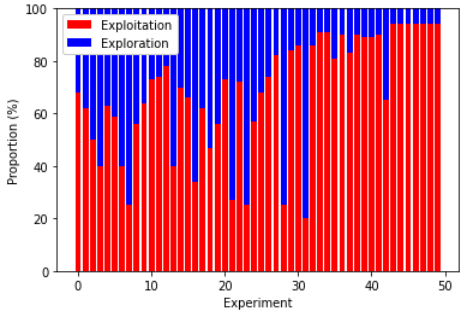

Fig. 7. Proportion of exploitation/exploration for each tested point.

(i.e. we chose non-relevant points). We evaluated that the number of iterations needed to reach 80% of the optimal value is $M = 20 \pm 12.5$. This variability is due to the fact that the algorithm is sensitive to initial points, especially when the number of initial points is small.

Sinus function with $N > 3$ input variables To get closer to a real world problem, we tested our approach with more variables. For that, we consider the following objective function with $N \geq 3$ input variables:

$$f_N(\boldsymbol{x}) = \left| \prod_{k=1}^{N} \sin(i\pi x_k) \right|, \tag{13}$$

with $\boldsymbol{x} = (x_1, x_2..., x_N)$ continuous, discrete or categorical variables. The values of y range between 0 and 1. More precisely, we study the case of $3 \leq N \leq 10$ input discrete variables (with 5 different values each). This simulates a composition optimization experiment whose values are very constrained.

We use this last toy dataset to compare our approach with Bayesian Optimization (BO). BO differs from our fuzzy algorithm for the following steps [6]: in BO, we evaluate the surrogate function using the Gaussian Process (GP) or the Tree-structured Parzen Estimator algorithm (TPE); and the compromise between exploitation and exploration is modelled by using the acquisition function called "Expected Improvement" (EI).

Analogously to our algorithm, the BO is repeated over a certain number of iterations. Each loop provides additional information until reaching an optimal value. The TPE algorithm is computationally efficient and well suited for high-dimensional optimization problems with an expensive objective function [2]. Additionally, it is well suited to optimization problems involving discrete and categorical variables, as well as continuous variables [1].

For both algorithms, we evaluate the number of iterations M needed to reach 80% of the optimal value for a given number N of input variables and for given 5 initial points (see Fig. 8). In the case of the BO, due to the random nature of the computation of the surrogate function, we had to repeat the simulation several

times to get an average value. We observe that globally our fuzzy algorithm gives better result than BO, it converges with less iterations. Moreover, BO is a non-deterministic method and we observe that the disparity of the convergence is quite important (see high standard deviation bars in blue in the figure).

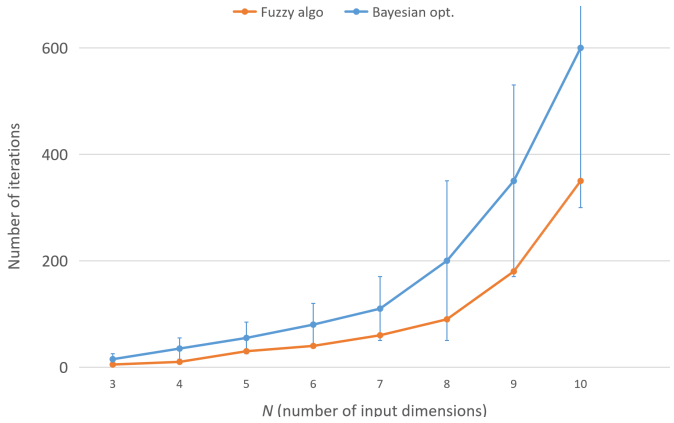

Fig. 8. Number of iterations needed to reach 80% of the optimal value - Case with N discrete variables.

6.3 Real world dataset

We then tested our approach on a real-world dataset from UC Irvine Machine Learning Repository called "Concrete Compressive Strength" [1].

It includes 8 input variables, and the goal is to maximize the normalized output variable "Concrete compressive strength (MPa)". This dataset includes 1030 instances. By using cross validation process (with 80% of training set and 20% of testing set), our simulations on this dataset showed that the surrogate function from our fuzzy algorithm leads to a better regression prediction than the Gaussian process from bayesian optimization: $RMSE = 8.7 \pm 2.9$ for our fuzzy algorithm, and $RMSE = 15.1 \pm 3.6$ for the Gaussian process. To determine the number of experiments to reach an optimal point, we set up the following process: we chose 5 random points from the 1030 instances with bad output value $y < 0.1$; an experiment consists here in taking a point among those instances chosen by the algorithm and we would like to converge fast to an optimal point. The full simulation is repeated several times to get an average value and standard deviation. The number of experiments needed to reach 80% of the optimal value of y is $M = 9 \pm 6.7$.

[1] https://archive.ics.uci.edu/dataset/165/concrete+compressive+strengt

7 End-user consideration

We evaluated the interaction with the end-users with a questionnaire (human-based evaluation). The panel is constituted of 29 individuals working in the field of Materials Science, from academic researchers to industrial researchers, aged from 22 to 62 years old. To collect their opinions, we used a 5-point Likert scale, from totally disagree to totally agree. The questionnaire describes a situation based on a mixture of 3 compounds (x_1, x_2, x_3) and a property (y) that ranges from 0 to 1. On a ternary diagram, 17 previous experiments are showed with the respective values of the property. We asked the panel to compare 4 different outputs:

- Type 1: the algorithm gives exactly the next values for x_1, x_2, x_3, as in Eq. 8;
- Type 2: same as Type 1 with an explanation (e.g., "to exploit an already explored zone and that gave good results");
- Type 3: the algorithm gives a region of interest as in Eq. 9;
- Type 4: same as Type 3 and with the same explanation as in Type 2.

Figure 9 shows the answers to the question : "are you satisfied by the next experiment(s) suggested by the AI?". The results are mainly positive, but the third type of output has the best marks (i.e. a region without explanation). Moreover, Figure 10 shows the answers to the question : "do you trust the choice of the AI?". The two types that do not provide any explanation have less positive results. However, the two types that provide the users with an explanation have more positive results but they also have one panelist who is totally disagree. We had a closer look to his answers and they are sometimes contradictory: it may be considered as an outlier.

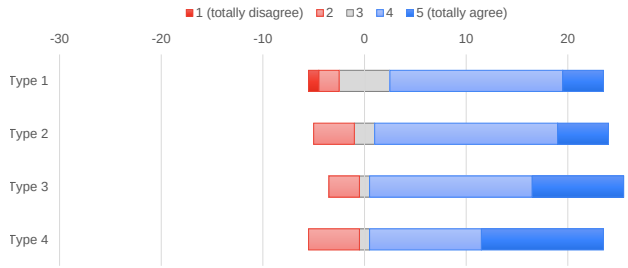

Fig. 9. Answers to the question: "Are you satisfied by the next experiment suggested by the AI?"

We asked the panelists what was their favorite output: 12 of them answered Type 4, 8 answered Type 2, 5 for Type 3 and finally 4 for Type 1. Thus, the two favorite outputs come with explanations. It is important to mention that we asked the panelists if they were afraid by the use of AI in their job: 7 panelists

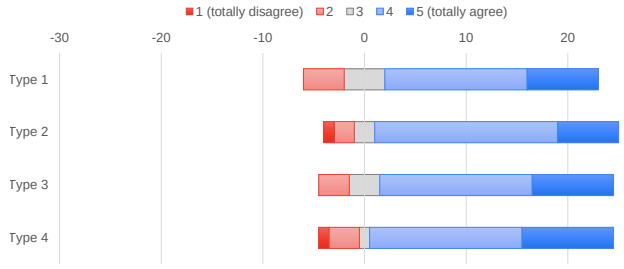

Fig. 10. Answers to the question: "Do you trust the choice of the AI?"

totally agree and 10 of them agree. 9 of them stayed neutral, meaning only 3 of them are not afraid. The results confirm what we were expecting: end-users prefer having a choice and an explanation, meaning that the final method should provide the region of the next experiment and an explanation of the recommendation. In a comment, one panelist wrote he prefers having possible experiments instead of only one, but since the AI chose a central experiment, his choice would be the same. It highlights the importance of considering the Human preferences in such tools to increase their acceptability.

8 Conclusion and perspectives

In conclusion, we summarize the advantages of our approach based on the results obtained. Our algorithm is transparent as a direct consequence of the interpretable nature of its parameters, the dominance of a cluster in each region of the input-output space, the lack of complexity and the linguistic nature of its fuzzy rules; it is deterministic, i.e. it converges to the same values at each execution; it is significantly faster than meta-heuristic approaches such as evolutionary algorithms; it is robust to overtraining and resilient to noise thanks to the fused contribution of clusters; finally, constraints from the literature can be applied to reduce the input search space. This approach has the merit to be interpretable, intuitive, and can be a real help to experimentalists. The originality of our algorithm includes the transformation of multidimensional fuzzy clusters into interpretable 1D fuzzy sets, and the definition of an acquisition/sampling score function from the variables $\hat{y}$ (predicted output value) and d (distance to the nearest sampled point). Possibilities for improvement include the computational speed in the high-dimensional case, better detection of local extrema, and multi-objective optimization.

Acknowledgments. This work is funded by the CEA Cross-Cutting Program on Materials and Processes Skills.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 2623–2631 (2019)
2. Bergstra, J., Yamins, D., Cox, D.: Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. Proc. of the 30th International Conference on Machine Learning (2013)
3. Bezdek, J.C., Ehrlich, R., Full, W.: Fcm: The fuzzy c-means clustering algorithm. Computers & Geosciences **10**(2), 191–203 (1984)
4. Chiu, S.L.: Fuzzy model identification based on cluster estimation. Journal of Intelligent & fuzzy systems **2**(3), 267–278 (1994)
5. Conn, A., Gould, N., Toint, P.: Trust-region methods. MPS-SIAM Series on Optimization 1. SIAM and MPS, Philadelphia (2000)
6. Greenhill, S., Rana, S., Gupta, S., Vellanki, P., Venkatesh, S.: Bayesian optimization for adaptive experimental design: A review. IEEE access **8**, 13937–13948 (2020)
7. Lesot, M.J., Rifqi, M., Bouchon-Meunier, B.: Fuzzy Prototypes: From a Cognitive View to a Machine Learning Principle, pp. 431–452. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
8. Li, L.J., Liang, Y.L.: A hierarchical fuzzy clustering algorithm. International Conference on Computer Application and System Modeling (2010)
9. Maneewongvatana, S., Mount, D.: Data Structures, Near Neighbor Searches, and Methodology, chap. Analysis of approximate nearest neighbor searching with clustered point sets. American Mathematical Society (2002)
10. Pagliaro, A., Sangiorgi, P.: AI in experiments: Present status and future prospects. Applied Sciences **13**(10415) (2023)
11. Tsekouras, G., Sarimveis, H., Kavakli, E., Bafas, G.: A hierarchical fuzzy-clustering approach to fuzzy modeling. Fuzzy sets and systems **150**(2), 245–266 (2005)
12. Türkşen, I.B.: A review of developments from fuzzy rule bases to fuzzy functions. In: 2012 Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS). pp. 1–5 (2012)
13. Viaña, J., Ralescu, S., Ralescu, A., Cohen, K., Kreinovich, V.: Explainable fuzzy cluster-based regression algorithm with gradient descent learning. Complex Engineering Systems (2022)

Index of notations

- N number of input dimensions
- $\boldsymbol{x}_s$ point already sampled ; $\boldsymbol{x}$ input space point
- $\boldsymbol{x}_{next}$ next point tested
- f fuzzy set
- c cluster ; C set of clusters ; $\boldsymbol{m}_c$ center of the cluster c ; n_c number of clusters
- μ_c membership degree to cluster c
- R fuzzy rule
- $\boldsymbol{\alpha}_c, \beta_c$ regression coefficients relative to the cluster c
- $\hat{y}_c$ output value predicted for an input $\boldsymbol{x}$, relative to the cluster c
- $\hat{y}$ global output value predicted for an input $\boldsymbol{x}$
- M number of iterations needed to reach an optimal point