Assisting Convergence Behaviour Characterisation with Unsupervised Clustering

Helena Stegherr*[a], Michael Heider*[b] and Jörg Hähner*[c]
Universiät Augsburg, Am Technologiezentrum 8, Augsburg, Germany
fi

Keywords: Metaheuristics, Behaviour Analysis, Convergence, Explainability.

Abstract: Analysing the behaviour of metaheuristics comprehensively and thereby enhancing explainability requires large empirical studies. However, the amount of data gathered in such experiments is often too large to be examined and evaluated visually. This necessitates establishing more efficient analysis procedures, but care has to be taken so that these do not obscure important information. This paper examines the suitability of clustering methods to assist in the characterisation of the behaviour of metaheuristics. The convergence behaviour is used as an example as its empirical analysis often requires looking at convergence curve plots, which is extremely tedious for large algorithmic datasets. We used the well-known K-Means clustering method and examined the results for different cluster sizes. Furthermore, we evaluated the clusters with respect to the characteristics they utilise and compared those with characteristics applied when a researcher inspects convergence curve plots. We found that clustering is a suitable technique to assist in the analysis of convergence behaviour, as the clusters strongly correspond to the grouping that would be done by a researcher, though the procedure still requires background knowledge to determine an adequate number of clusters. Overall, this enables us to inspect only few curves per cluster instead of all individual curves.

1 INTRODUCTION

Empirical studies are a common approach to compare metaheuristics, to analyse their performance, and to examine their search behaviour. However, every empirical study requires a well-wrought, rigorous design, so that the results are valid and unbiased (Bartz-Beielstein et al., 2020). This goes for selecting the algorithmic configurations, but also for gathering the required data and applying a suitable analysis methodology. Depending on the research questions, this can be quite demanding, and the problem is exacerbated when the number of algorithmic configurations to be examined grows.

Especially when the goal is the analysis of algorithmic behaviour, large scale empirical studies are quite common (see e.g. (Vermetten et al., 2022b)), which can be based on typical benchmarking experiments or on specialised experiment designs (Bartz-Beielstein et al., 2020). These analyses are important to not only understand how a metaheuristic achieves its performance, but also to derive more general insights into its workings and explain which part of its configuration, i.e. which operators or hyperparameters, is responsible for a certain behaviour. In such studies, often lots of data is collected, in terms of different configurations and several runs, as well as the information that is gathered during the search (e.g. the current best value, but also the objective values of other solutions or the solutions itself). Utilising all data in a comprehensive analysis can advance the explainability of the search process (Bacardit et al., 2022), e.g. by the creation of behavioural profiles of metaheuristics and their operators. These can then be used to help non-experts to configure an algorithm for the problem at hand, based on the provided knowledge of the behaviour under specific circumstances.

However, the amount of information, and therefore also the amount of data, that needs to be analysed and presented for this is extremely large. Specifically, data that is often analysed based on some kind of visualisation becomes a problem. While visualisations are often easily interpretable, it becomes difficult when trying to compare more than a few dozen algorithms based on visual information. Large studies, however, easily produce data of thousands or even

[a] https://orcid.org/0000-0001-7871-7309
[b] https://orcid.org/0000-0003-3140-1993
[c] https://orcid.org/0000-0003-0107-264X
*Authors contributed equally to this paper.

Stegherr, H., Heider, M. and Hähner, J.
Assisting Convergence Behaviour Characterisation with Unsupervised Clustering.
DOI: 10.5220/0012202100003595
In *Proceedings of the 15th International Joint Conference on Computational Intelligence (IJCCI 2023)*, pages 108-118
ISBN: 978-989-758-674-3; ISSN: 2184-3236

tens of thousands different algorithmic configurations (see e.g. (van Stein et al., 2021; Vermetten et al., 2022a)). Even when several configurations can be combined in the visualisation of specific behavioural characteristics, analysing them is not feasible by humans. An effective way around this is to break the visualisations down to numbers. This can be done utilising summarising metrics, or by looking at specific measures that indicate differences between the configurations. Again, however, this can easily get out of hand, when many different measures need to be compared for analysing one specific behavioural characteristic in depth. Furthermore, it is not always clear up front which of these measures are the most important to make any distinction, leading to either a high number of preliminary studies or including unnecessary measures in the analysis.

To alleviate these problems, we want to determine the suitability of another approach, utilising machine learning (ML) to summarise, differentiate, and—ultimately—explain algorithmic configurations by their behavioural characteristics. There are many ML techniques that could be supportive in facilitating the analysis of behavioural characteristics under consideration of configuration-specific peculiarities. Especially unsupervised ML approaches are of interest here, as these do not require data labelling, which would again constitute a large additional effort and defeats the purpose of not having to check all individual curves. Within unsupervised learning, the use of traditional clustering algorithms is the logical first step. Utilising machine learning techniques to assist in behavioural analysis is not yet common. There are, however, some statistical approaches that can be employed, e.g. for examining exploration and exploitation behaviour (Eftimov and Korošec, 2019). Also, there are general as well as specialised frameworks and statistical tools that can assist in such cases, e.g. (Bartz-Beielstein et al., 2017; Eftimov et al., 2020; Wang et al., 2022; Vermetten et al., 2022b). ML is used for evaluating metaheuristics in general, for example for predicting their performance using regression models (Eftimov et al., 2021) or for algorithm selection (see e.g. (Tanabe, 2022)). It has to be considered that ML-assisted behaviour characterisation may not be able to provide a full picture of all details that differentiate algorithms. However, it should at least vastly facilitate getting an initial overview. Cases of interest, where the ML-assisted analysis is not comprehensive enough, can still be analysed individually by looking at specific metrics, measures or visualisations, but the data-related overhead is reduced drastically.

As a proof of concept and to find the strengths, weaknesses and prerequisites of the clustering approach, we focus—for now—on the convergence behaviour of metaheuristics. We assume there to be a distinct number of "types of convergence curves" that are largely similar to each other. While this number might not exactly be known, we can infer some sensible numbers based on expert knowledge and let the algorithm sort the data accordingly. After constructing such clusters of similar curves, practitioners can pick a few individuals from the respective clusters for analysis, thereby reducing the number of curves that need initial manual checking by (multiple) orders of magnitude.

In the following, we examine K-Means—an unsupervised clustering algorithm—in the context of a smaller example of behavioural analysis to determine its applicability and restrictions. Therefore, we look at the convergence behaviour of metaheuristics (note that the focus is on solving real-parameter minimisation problems). We give an introduction to the characteristics of convergence behaviour that are important to distinguish different algorithmic approaches (Section 2). Then, different configurations of K-Means to assist in analysis based on these considerations are presented and evaluated (Section 3), with a focus on relating the results to our expectations. Finally, we discuss other potential ML approaches and subsequent analyses that can allow to draw further conclusions.

2 CONVERGENCE BEHAVIOUR AND CHARACTERISTICS

The behaviour of metaheuristics during the search process depends on the algorithmic configuration as well as the optimisation problem. While the configuration determines the general strategy the algorithm uses to traverse the search space, the optimisation problem itself can influence the algorithm if objective function values are utilised in its internal decisions. A detailed analysis of the algorithmic behaviour can help to identify which factors constitute to the respective behaviour, and explain the search process. Furthermore, this information can be used to determine which configuration might be appropriate for an unknown problem. The overall algorithmic behaviour can be divided into subgroups that can be examined independently, e.g. convergence-, performance-, exploration-, or exploitation-related behaviour. For each of these subgroups, it is required to look at several characteristics to examine the behaviour in enough detail to relate it to features of the algorithmic configuration.

This section explores the typical characteristics related to the convergence behaviour of an algorithm. They are summarised in a way we expect that researchers would look at them when analysing empirical data to compare different metaheuristics. Furthermore, these characteristics provide a hypothetical foundation for Section 3 to interpret the results of the machine learning techniques.

Typical convergence metrics, measures and plots and their interpretations are summarised in (He and Lin, 2016; Chen and He, 2021; Halim et al., 2020) and statistical convergence criteria, i.e. ones that determine if the algorithm has converged, can be found in (Campelo, 2015). While they relate to the characteristics of convergence behaviour, there are, to our knowledge, no specific publications on that topic, and no general guidelines on how to identify such characteristics. There is, however, an approach that uses characteristics of the convergence curve to improve the algorithm, though not to understand the behaviour (Azad, 2019).

For a rough inspection of differences in convergence behaviour, common metrics and measures are used. These include the *convergence rate*, *average convergence rate* and the point where the algorithm is converged to 5% of the optimum (He and Lin, 2016; Chen and He, 2021; Halim et al., 2020). While these mostly utilise the objective value, other convergence measures are based on the diversity of the population, e.g. (Bosman and Engelbrecht, 2014). Additionally, search trajectory networks offer more detailed visualisations and metrics for convergence analysis (Ochoa et al., 2021).

Differences in convergence behaviour are often examined in detail by comparing the plotted convergence curves of the algorithmic runs. These curves allow for two levels of detail: purely considering the visual shape of the plotted curve, and extracting several points of interest to compare the respective values. While in benchmarking studies, empirical cumulative distribution functions (ECDFs) are the typical visualisation (Hansen et al., 2016), in specialised experiments, which are still common and useful in behaviour analysis and which are the focus of this work, convergence is visualised by plotting the best objective value[1] per time step.

The curve types can give a general overview on distinct convergence behaviour but without considering the scale, i.e. the range of objective function values during the run. Figure 1 presents convergence curve types typically encountered when plotting the best objective function value per time step. Note that these are only examples for the general curve form and the final objective value is not represented meaningfully, i.e. the y-axis can be any scale.

The curves can be categorised in several main types, with smaller distinctions within those types. The main categories can be labelled as follows:

Normal: Fast at the beginning, with decreasing improvement over time.

Fast: Converged within first few steps.

Linear: Almost linear convergence until final value is reached. (1 additional variation)

Fast to Slow: Initially fast, then rate of improvement decreases, sometimes with a visible point of change. (1 additional variation)

Slow to Fast: Initially slow, then rate of improvement increases, sometimes with a visible point of change. (1 additional variation)

Steps: Alternation between faster and slower episodes. (2 additional variations)

Suboptimal: Converges to any other value. (all of the above: 10 additional variations)

None: Almost no improvement. (2 additional variations)

Presumably, not all types are equally common, at least in typical thought-through experiments. Especially the *None* type should not occur at all when the experiment includes rigorously configured algorithms. It can, however, be present when the goal is to inspect different configurations and the resulting behaviour in relation to the performance, no matter if it is good or bad. We expect one of the most common curve type to be the *Normal* one, which shows a gradually decreasing rate of improvement of the objective function value. The *Fast* type should be frequent when problems with low modality are considered, and *Suboptimal* should be common for problems with many local minima or plateaus in their fitness landscape. The other four types are presumably the most interesting ones when analysing the algorithmic behaviour, as they either—for the *Linear* type—show a constant improvement step size, or have distinct points in the search process where changes in the convergence behaviour occur. Determining the factors of the algorithmic configuration or of the problem landscape responsible for this behaviour might provide valuable insights.

As stated above, the categorisation of the curve type as a convergence behaviour characteristic is not sufficient for a detailed convergence behaviour analysis. Therefore, some common measures of the convergence process are usually considered as additional characteristics. Some of them are:

[1]The fitness of the best individual in population-based methods.

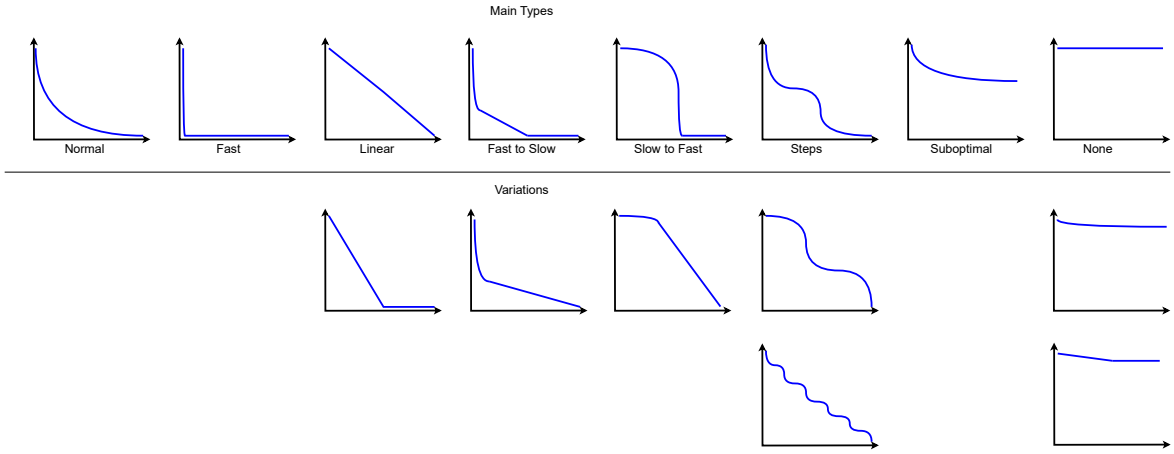


Figure 1: Examples of different expected convergence curve types and some of their common variations.

Final Value: Final value to which the algorithm converges to (optimum or other value).

50% Objective: When has the objective function value reached 50% of its initial value?

75% Objective: When has the objective function value reached 75% of its initial value?

90% Objective: When has the objective function value reached 95% of its initial value?

50% Budget: What is the objective function value after 50% of the given budget?

75% Budget: What is the objective function value after 75% of the given budget?

95% Budget: What is the objective function value after 95% of the given budget?

Number of Steps: How many "steps" are there in the convergence curve?

Which additional measures are of interest strongly depends on the research question and the curve type. Therefore, there can be additional ones to those mentioned above. For example, when initialisation strategies are taken into the overall consideration, the starting point in terms of the initial best objective function value[2] is of interest. Conversely, there can also be fewer measures of interest if a more coarse grained analysis is sufficient. It is important to note that these measures alone do also not show the whole picture when empirically analysing convergence behaviour. The curve types provide additional information, making it necessary to use both, at least to some extent. Regardless, multi-faceted and detailed analyses enhance a user's (or researcher's) understanding of the optimiser, improving the explainability further.

[2]The initial elitists' objective function value.

3 CLUSTERING CONVERGENCE CURVE DATA

As Section 2 illustrates, there are lots of characteristics that need to be considered when analysing the convergence behaviour—though the exact number depends on the specific research goal. For larger experiments, i.e. often multiple thousand algorithmic configurations and the resulting data, it is not feasible to look at each plot and measure individually. There are several approaches on how to facilitate the analysis of such experiments. We focus on utilising unsupervised ML techniques, particularly clustering methods, to group the data, and through that the different algorithmic configurations, based on their convergence behaviour characteristics.

Utilising clustering techniques to aid in convergence behaviour analysis necessitates that they provide results that are comparable to those that researchers would get when performing the analysis themselves. Therefore, this section provides a detailed examination of the process required to appropriately employ clustering for this case and of the resulting information and its practicality. We focus particularly on discussing the necessary prior knowledge and the advantages, disadvantages and short-comings of the approach. Furthermore, a short overview on subsequent and alternative strategies is provided.

We have three assumptions the clustering approach should verify to be considered applicable:

1. The clusters reflect the convergence curves.
2. The similarities within a cluster and the differences between clusters can be related to the characteristics a researcher would look for.

3. There should be differences in the allocation of examples per cluster—even for a higher number of clusters—as not all curve types should be equally common.

3.1 Data Gathering

To adequately evaluate clustering for convergence behaviour analysis, a representative dataset of algorithmic runs is required. Representative, in this case, means including at least the most common different convergence curve types and show differences in the measures presented in the previous section. Furthermore, enough examples need to be present in the dataset.

We used a genetic algorithm to generate the data, implemented in the MAHF software framework (Stegherr et al., 2023), with tournament selection, uniform crossover, Gaussian mutation and elitism. We varied the hyperparameters, utilising common values as well as values that provoke runs not converging to the global optimum. Five typical benchmark functions were optimised, with two settings for their respective dimensions, generating a total of 11500 configurations, which were run once each. The best objective function value was logged every ten iterations for a total of 5000 iterations. The final data examples then contain a sequence of 500 objective values, i.e. the current best objective function value within the population for each logged iteration. Note that we only want to test the capabilities of the clustering methods. In this case, the number of runs per configuration does not matter as we aim to produce a diverse set of curves rather than meaningful insights about this particular (and probably rather well known) GA which is easier with different configurations than multiple runs of the same configuration. Furthermore, while budgeting in terms of function evaluations is a more appropriate approach than having a fixed number of iterations, it does not matter here as the goal is to provide different convergence plots not find the best objective function values. However, we want to stress that for specific hyperparameter/operator/algorithm behaviour analyses, multiple runs and function evaluation–based budgeting should be part of a good workflow (Bartz-Beielstein et al., 2020).

3.2 Clustering Based on Convergence Curve Information

After conducting a set of experiments, a practitioner that wants to understand the behaviour of chosen algorithmic configurations (i.e. hyperparameters, operators or general structure of the metaheuristic) should not only evaluate based on the achieved objective function value, but also based on the path that was taken towards that value. This can be easily visualised by plotting the best value at each time step which always follows a monotone function. While these curves can be analysed manually following the considerations presented in Section 2, more comprehensive tests often create too many convergence curves for manual analysis. However, many of those curves show similar patterns.

We propose the use of unsupervised ML to cluster those curves. These techniques do not require labels, which would be very labour-intensive to acquire, but rather find patterns in the data to correlate similar data points. In particular, we examine the well-known K-Means algorithm[3]. K-Means separates data into specified numbers of clusters minimizing the in-cluster variance. It exhibits good runtime scaling to large sample sizes—which is particularly useful for large-scale behavioural analyses on a variety of problems—and has the advantage of being able to sort newly generated data into the existing clusters, which can be quite interesting when performing continued analyses.

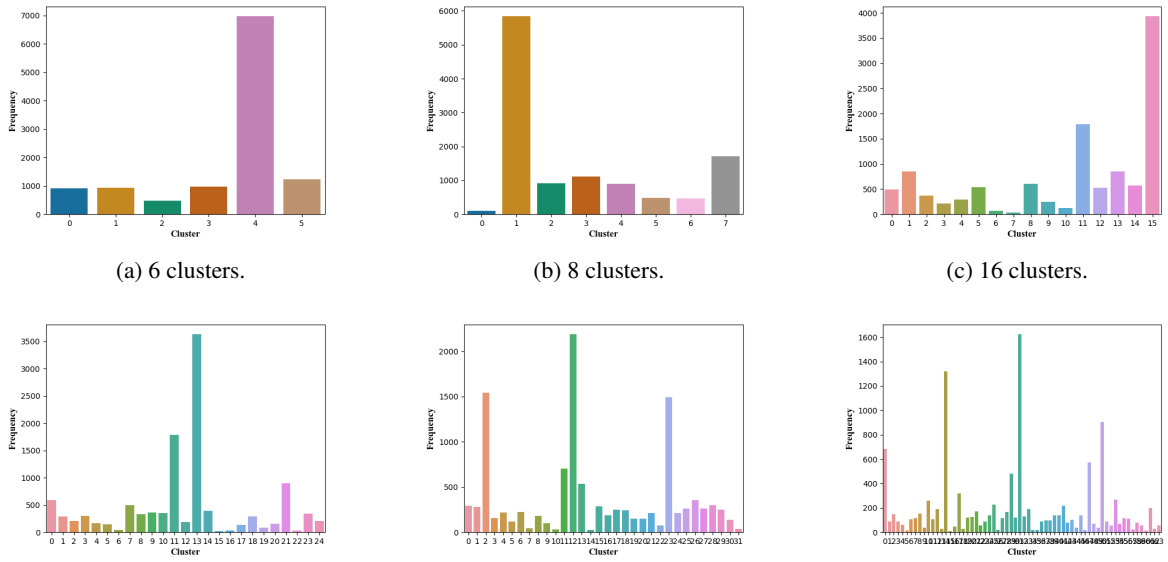
For the implementation in this paper we normalise our curves individually, i.e. the value of the first elitist is set to 1 and the value of the reachable optimum is set to 0 with everything in-between scaled accordingly. Note that this can be adapted when the optimum is unknown, using the minimum any algorithmic configuration has found for a specific test function. Given the stochastic nature of the K-Means optimiser, we report the clusters of the minimal reached inertia after 10 consecutive runs which is the standard workflow with this algorithm.

In contrast to many other clustering techniques, K-Means also allows us to predefine the number of clusters we expect based on expert knowledge common among researchers and is rather insensitive to hyperparameters, which we expect to be very valuable for those that rarely use ML-techniques (and even for those that do). However, we do not expect the ability of our proposed approach to differ considerably from other clustering techniques, e.g. DBSCAN or OPTICS, should they be configured appropriately.

From our prior knowledge we know that:

1. we need at least six clusters based on the convergence curves, based on their similarity (combining *Normal*, *Fast*, and *Fast to Slow*, and *Suboptimal* and *None*),

[3]We use the implementation in the Python package scikit-learn



(a) 6 clusters. (b) 8 clusters. (c) 16 clusters.

(d) 25 clusters. (e) 32 clusters. (f) 64 clusters.

Figure 2: Frequency of examples per cluster for different Ks.

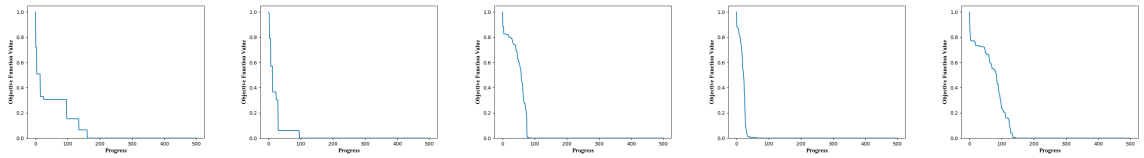
2. we want to try eight clusters, one for each main type,
3. we might want to test 25 clusters, one for each possible curve,
4. we should also try some value in between, perhaps: 16,
5. fewer than 32 clusters could still be a bit too few so we should also run that, and,
6. finally, we want to, additionally, test 64 clusters, one for each main curve type and for the different measures.

Based on these considerations, we performed K-Means clustering with $K \in \{6, 8, 16, 25, 32, 64\}$. The respective distributions of curves into these clusters can be found in Figure 2. We find that, for all sizes, a large number of runs can be clustered together, e.g. the curves of cluster 1 for $K = 8$ are all converging very fast towards a good objective value. Note that the order of clusters is arbitrary. A number of interesting examples of clusters and misclustering can be found in the Figures 3 to 8 and they are discussed in Section 3.3.

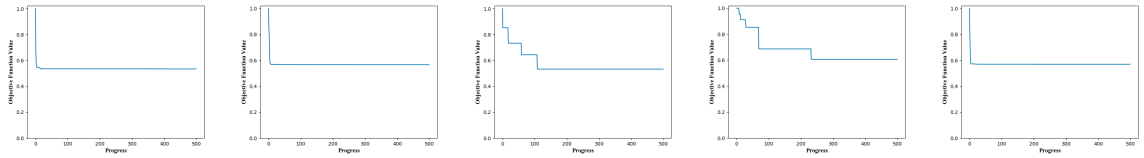
3.3 Discussion

In order to determine the usefulness of clustering for analysing the convergence behaviour according to specific characteristics, the results from the clustering approaches have to be examined in the context of the aims of the analysis. We assumed that clustering might be useful to group different algorithmic configurations by their convergence behaviour, which would ease the in-depth analysis of these configurations as only the distinct clusters rather than all curves would need to be analysed in detail. Furthermore, we expected the clustering methods to utilise similar characteristics as a researcher would (see Section 2). Finally, we surmised that some convergence curve types should be more common than others, possibly not only in the dataset used in this approach but in general (e.g. a metaheuristic should perform any search, so the *None* type should only result from extremely unsuitable configurations).

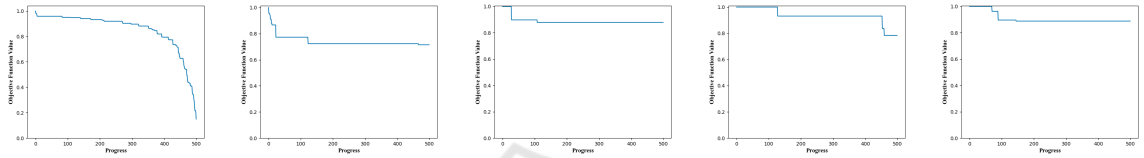
In general, clustering can be used for that purpose. However, there are considerations that have to be made up front. For K-Means clustering, as employed in this work, the number of clusters has to be specified up front. Therefore, either expert knowledge is required to estimate the expected number of clusters, or the number has to be determined by trial-and-error. Additionally, the number of clusters also depends on the research question. In this case, we wanted the clustering to reflect the grouping we would get when analysing all convergence curves visually, while keeping in mind the convergence curve characteristics. By looking at examples from the clusters, we found that for 6, 8, and 16 clusters, the differentiation is not fine-grained enough as we could still find convergence curves within the same cluster that we would identify as clearly different (see Figures 3, 4 and 5). For 25 clusters, there were still some slight differences (cf. Figure 6), while for 32 clusters, there



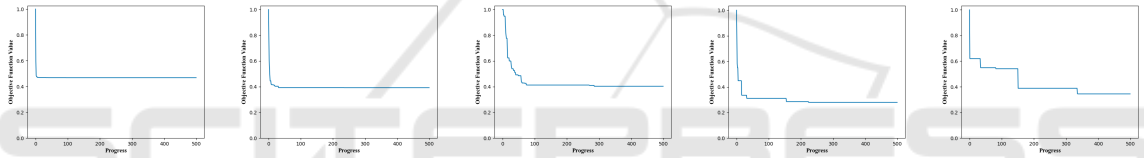
(a) Cluster 0.



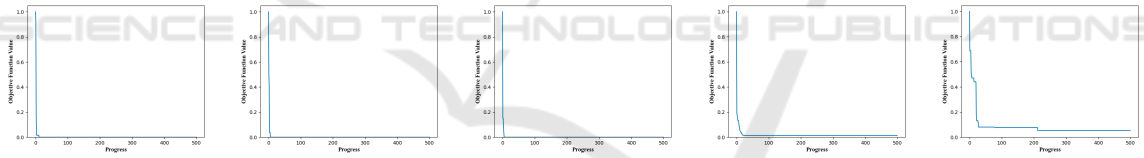
(b) Cluster 1.



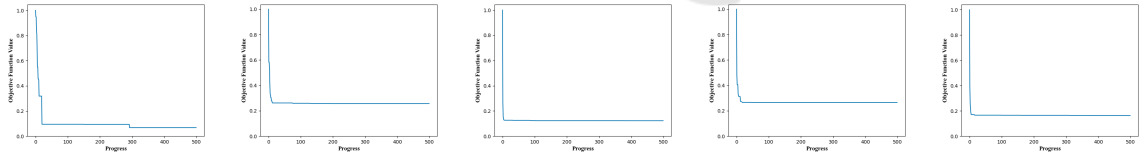
(c) Cluster 2.



(d) Cluster 3.



(e) Cluster 4.



(f) Cluster 5.

Figure 3: Five randomly selected example curves per cluster for K=6.

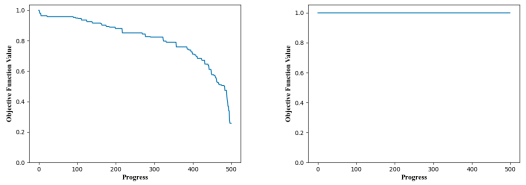


Figure 4: Clearly different curves within cluster 5 for $K = 8$.

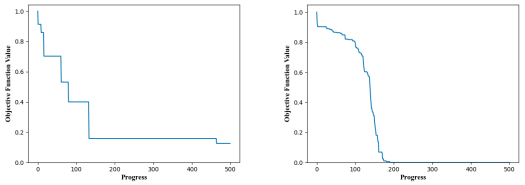


Figure 5: Different curves within cluster 10 for $K = 16$.

were convergence curves which seem similar but are located in different clusters (see Figure 7). Depending on the goal of the analysis, both might still be adequate, or any value in between. More clusters do not seem to be useful, as even here differences within a cluster can still be found, but more and more clusters are too similar (see Figure 8). Most likely, dissimilar curves in the same cluster are the result of the convex decision boundaries introduced by K-Means. However, we can assume that some curves that fall on that border are clustered in one while others cluster into another group. Having similar clusters is clearly a result of K being too high and therefore introducing new centroids that are subsequently slicing groups of closely related curves in part.

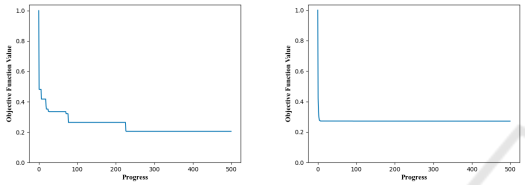


Figure 6: Slightly differing curves within cluster 18 for $K = 25$.

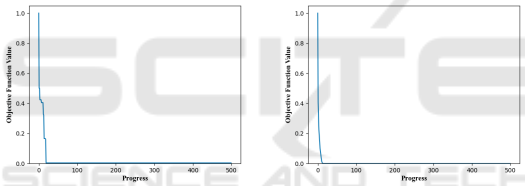
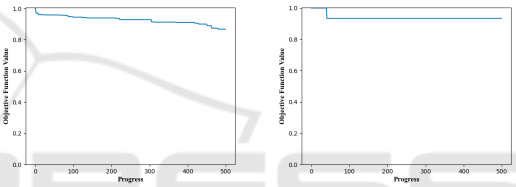


Figure 7: Similar curves split into clusters 11 and 12 for $K = 32$.

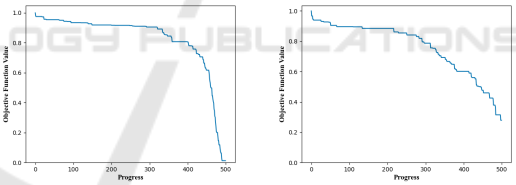
The distinction between the clusters can be related to the characteristics described in Section 2, at least in large parts. This is encouraging, as for a comprehensive analysis of the convergence behaviour, an appropriate interpretability of the clusters is necessary. For example, the cluster in Figure 3b corresponds obviously to the *Suboptimal* curve type and the entailed examples are additionally grouped by their final value. The cluster in Figure 3e corresponds to *Fast* convergence with an (almost) optimal final value, while that in Figure 3a mixes *Normal*, *Steps* and *Slow to Fast*, but its examples always reach the best possible final value. When grouping into more clusters, the differentiation is also based on more nuances in the characteristics.

In terms of examples per cluster (see Figure 2), we see that some clusters encompass more than others. For a number of 6 clusters, the most examples are grouped in cluster 4. This corresponds to *Fast* or at least really good *Normal* convergence curves (see Figure 3e), as was expected with not too difficult optimisation problems. For all K, this still stays the same, i.e. the clusters with the highest number of examples depict slightly different variations of *Fast* type curves. Conversely, the clusters with the lowest allocation correspond to the badly performing algorithmic configurations which exhibit the *None* or an extreme case of the *Suboptimal* curve type.

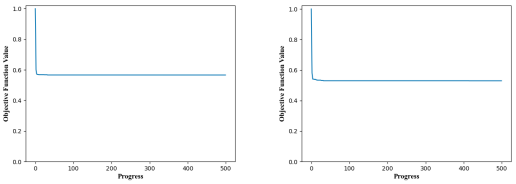
Overall, clustering can be used as a mean to facilitate the analysis of convergence behaviour on the basis of comprehensible characteristics and acts according to our expectations. It summarises similar convergence curves into the same cluster, enabling practitioners to proceed with further (visual or computational) analyses per cluster instead of per configuration, saving time and providing a first step to correlate similar behaviour to distinct configurations. However, it should be noted that it is not without flaws and especially not without putting in some effort, either to find a suitable configuration or to verify that the results are adequate for the research goal.



(a) Examples within cluster 11 show differences.



(b) Examples within cluster 18 show differences.



(c) Examples within cluster 1 and cluster 12 are similar.

Figure 8: Some examples for $K = 64$.

3.4 Different Strategies and Subsequent Approaches

In this paper, we relied on a traditional—and comparatively simple—clustering algorithm. However, we want to raise to attention the power of modern deep learning algorithms. In particular, we propose the use

of autoencoder-based architectures for future analyses. An autoencoder is a type of neural network that features a so called latent space of much smaller dimensionality than the original inputs (dimensionality reduction techniques in general have been used a few times within recent years, e.g. for population dynamics visualisation (Walter et al., 2022)). The network is trained to output the input data without any loss of information, however, by choosing a smaller latent space, it is forced to learn a compression (hence the name encoder) and a decompression function of the data. We expect the autoencoder to be able to compress the curves into very few (3-7) real-valued features. These features can then be clustered similarly to the current technique but, moreover, they can be used to describe the curves and maybe these features can even be related to how a human domain expert would describe the convergence process, e.g. "this curve is very steep", but with a hard quantification that allows ordering and many more advantages of specific numbers.

The exact design of the autoencoder is a topic of future investigation but we assume that a temporal convolution (TCN)–based setup could be advantageous over long short-term memory (LSTM)–based or, especially, fully connected feed forward networks (cf. the results of e.g. (Bai et al., 2018)) . Holstad et al. (Holstad et al., 2020) found architectures using LSTM layers able to compress real-world measurement curves similar to some of our found clusters into only three features and reasoned about the underlying physical processes based on that. In our case, LSTMs might not be an ideal choice as they tend to prioritise recent timesteps over long past ones for their feature generation. However, some convergences curves are most interesting in the beginning, while some runs converge only late in the optimisation process (cf. Figure 3). While the use of (the encoding parts of) transformer networks (Vaswani et al., 2017) could be discussed, we expect this to be unnecessarily complex for the task at hand. Although, the self-attention mechanism might yield some insights into why certain parts of a curve are of interest.

While for the current purpose K-Means seems to be a sufficient algorithm, we will test a variety of autoencoders in the near future as the potential advantages towards greater understanding and easier analysis of operators and metaheuristics in general seem intriguing.

Of course, other approaches can be used to group by a specific behaviour according to predefined characteristics would. One would be to implement a reasonable set of hard-coded rules directly, based on the respective requirements of the research goal. While, when done right, this approach will provide the best results, it requires extensive expert knowledge and a lot of time. Furthermore, if the research goal changes or is adapted, it might be necessary to manually adapt the procedure, requiring further knowledge of the new circumstances. Alternative strategies to perform or facilitate behavioural analyses of metaheuristics by reducing the amount of (visual) information that has to be examined include classic statistical approaches such as factor analysis or combined metrics. Both can be applied on their own, but also after the clustering approach, which provides the advantage of a meaningful preprocessing of the algorithmic data. Factor analysis is aimed at finding the factors (e.g. hyperparameters or operators) or their combinations that influence the criterion under investigation the most (see e.g. (Bang-Jensen et al., 2007)). However, they are data and computation heavy and require extensive examination of measures when the number of factors is high. Combined metrics, on the other hand, can be employed to summarise different measures and metrics (e.g. those mentioned in Section 2) into one single value, thus reducing the amount of information that needs to be investigated in detail (cf. e.g. (Song et al., 2013; Chignell et al., 2015)). Their disadvantage is the loss of information that can occur when not properly devised, and it may be still required to take a closer look at the individual measures.

4 CONCLUSION

This paper examined the applicability of unsupervised machine learning techniques, particularly clustering, to facilitate the analysis of the convergence behaviour of different algorithmic configurations. This is especially useful when performing large empirical studies, with several thousand configurations, where a visual analysis of all measures and plots is infeasible. We found that clustering techniques can be successfully applied to alleviate this problem and that the characteristics used for clustering even relate to those a researcher would consider. However, some prior knowledge is still required, especially when determining the number of clusters that should be used and when analysing if the clustering performs as intended. Overall, as the scale of empirical studies in the field of metaheuristics is growing and more complex relationships between configurations and algorithmic behaviour are of interest, it is worth exploring clustering, for which we demonstrated its applicability and advantages, and unsupervised machine learning in general, as well as to conceptualise other suitable approaches.

ACKNOWLEDGEMENTS

This work was partially funded by the *Deutsche Forschungsgemeinschaft (DFG)*. It was also partially funded by the *Bavarian Ministry of Economic Affairs, Regional Development and Energy*.

REFERENCES

Azad, S. K. (2019). Monitored convergence curve: a new framework for metaheuristic structural optimization algorithms. *Structural and Multidisciplinary Optimization*, 60(2):481–499.

Bacardit, J., Brownlee, A. E. I., Cagnoni, S., Iacca, G., McCall, J., and Walker, D. (2022). The intersection of evolutionary computation and explainable AI. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM.

Bai, S., Kolter, J. Z., and Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling.

Bang-Jensen, J., Chiarandini, M., Goegebeur, Y., and Jørgensen, B. (2007). Mixed models for the analysis of local search components. In *Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics*, pages 91–105. Springer Berlin Heidelberg.

Bartz-Beielstein, T., Doerr, C., Berg, D. v. d., Bossek, J., Chandrasekaran, S., Eftimov, T., Fischbach, A., Kerschke, P., La Cava, W., Lopez-Ibanez, M., Malan, K. M., Moore, J. H., Naujoks, B., Orzechowski, P., Volz, V., Wagner, M., and Weise, T. (2020). Benchmarking in Optimization: Best Practice and Open Issues.

Bartz-Beielstein, T., Zaefferer, M., and Rehbach, F. (2017). In a Nutshell - The Sequential Parameter Optimization Toolbox.

Bosman, P. and Engelbrecht, A. P. (2014). Diversity rate of change measurement for particle swarm optimisers. In *Lecture Notes in Computer Science*, pages 86–97. Springer International Publishing.

Campelo, F. (2015). Towards statistical convergence criteria for mutation-based evolutionary algorithms. In *2015 Latin America Congress on Computational Intelligence (LA-CCI)*. IEEE.

Chen, Y. and He, J. (2021). Average convergence rate of evolutionary algorithms in continuous optimization. *Information Sciences*, 562:200–219.

Chignell, M., Tong, T., Mizobuchi, S., Delange, T., Ho, W., and Walmsley, W. (2015). Combining multiple measures into a single figure of merit. *Procedia Computer Science*, 69:36–43.

Eftimov, T., Jankovic, A., Popovski, G., Doerr, C., and Korošec, P. (2021). Personalizing performance regression models to black-box optimization problems. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM.

Eftimov, T. and Korošec, P. (2019). A novel statistical approach for comparing meta-heuristic stochastic optimization algorithms according to the distribution of solutions in the search space. *Information Sciences*, 489:255–273.

Eftimov, T., Petelin, G., and Korošec, P. (2020). DSCTool: A web-service-based framework for statistical comparison of stochastic optimization algorithms. *Applied Soft Computing*, 87:105977.

Halim, A. H., Ismail, I., and Das, S. (2020). Performance assessment of the metaheuristic optimization algorithms: an exhaustive review. *Artificial Intelligence Review*, 54(3):2323–2409.

Hansen, N., Auger, A., Brockhoff, D., Tušar, D., and Tušar, T. (2016). COCO: Performance Assessment.

He, J. and Lin, G. (2016). Average convergence rate of evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 20(2):316–321.

Holstad, T. S., Ræder, T. M., Evans, D. M., Småbråten, D. R., Krohns, S., Schaab, J., Yan, Z., Bourret, E., van Helvoort, A. T. J., Grande, T., Selbach, S. M., Agar, J. C., and Meier, D. (2020). Application of a long short-term memory for deconvoluting conductance contributions at charged ferroelectric domain walls. *npj Computational Materials*, 6(1).

Ochoa, G., Malan, K. M., and Blum, C. (2021). Search trajectory networks: A tool for analysing and visualising the behaviour of metaheuristics. *Applied Soft Computing*, 109:107492.

Song, M.-K., Lin, F.-C., Ward, S. E., and Fine, J. P. (2013). Composite variables. *Nursing Research*, 62(1):45–49.

Stegherr, H., Luley, L., Wurth, J., Heider, M., and Hähner, J. (2023). A framework for modular construction and evaluation of metaheuristics. Technical Report 2023-01, Fakultät für Angewandte Informatik.

Tanabe, R. (2022). Benchmarking feature-based algorithm selection systems for black-box numerical optimization. *IEEE Transactions on Evolutionary Computation*, 26(6):1321–1335.

van Stein, B., Caraffini, F., and Kononova, A. V. (2021). Emergence of structural bias in differential evolution. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Vermetten, D., Caraffini, F., van Stein, B., and Kononova, A. V. (2022a). Using structural bias to analyse the behaviour of modular CMA-ES. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM.

Vermetten, D., van Stein, B., Caraffini, F., Minku, L. L., and Kononova, A. V. (2022b). BIAS: A toolbox for benchmarking structural bias in the continuous domain. *IEEE Transactions on Evolutionary Computation*, 26(6):1380–1393.

Walter, M. J., Walker, D. J., and Craven, M. J. (2022). Visualizing population dynamics to examine algorithm performance. *IEEE Transactions on Evolutionary Computation*, 26(6):1501–1510.

Wang, H., Vermetten, D., Ye, F., Doerr, C., and Bäck, T. (2022). IOHanalyzer: Detailed performance analyses for iterative optimization heuristics. *ACM Transactions on Evolutionary Learning and Optimization*, 2(1):1–29.