Towards explaining metaheuristic solution quality by data mining surrogate fitness models for importance of variables

Aidan Wallace, Alexander E.I. Brownlee, and David Cairns

University of Stirling
{a.l.wallace,alexander.brownlee,david.cairns}@stir.ac.uk
http://www.cs.stir.ac.uk/~alw ~sbr ~dec

Abstract. Metaheuristics are randomised search algorithms that are effective at finding "good enough" solutions to optimisation problems. However, they present no justification for generated solutions and these solutions are non-trivial to analyse in most cases. We propose that identifying the combinations of variables that strongly influence solution quality, and the nature of this relationship, represents a step towards explaining the choices made by a metaheuristic. Using three benchmark problems, we present an approach to mining this information by using a "surrogate fitness function" within a metaheuristic. For each problem, rankings of the importance of each variable with respect to fitness are determined through sampling of the surrogate model. We show that two of the three surrogate models tested were able to generate variable rankings that agree with our understanding of variable importance rankings within the three common binary benchmark problems trialled.

Keywords: metaheuristics · surrogates · optimisation · explainability

1 Introduction

With the uptake and utilisation of Artificial Intelligence (AI) driven systems across multiple sectors, we are seeing an increase in the presence of such systems within many businesses and organisations. As this continues these systems are being tasked to handle a larger number of decisions at greater levels of importance and with fewer checkpoints of human interference and guidance. It is thus vital that the decisions they are making can not only be trusted but more importantly understood by the end user as much as possible. It is hoped that, if individuals are able to build confidence in the suggestions of AI systems, they will be more likely to employ these systems and act on the solutions they provide. For this reason, enabling these processes to be understood and explained is paramount to maximising the real world utility of many machine driven approaches [18].

One branch of AI, known as metaheuristics, comprises search-based methods that have been shown to efficiently find well performing solutions to difficult optimisation problems. Metaheuristics look for solutions that minimise or maximise a set of objectives that are often related to cost or performance. Optimisation problems are prevalent in many branches of industry and have an impact on numerous aspects of everyday life for many, from Computing Science and Engineering to the less traditional fields of Medicine and Retail [8, 22, 24], and can take many forms.

This is a post-peer-review, pre-copyedit version of an article published in Bramer M & Ellis R (eds.) *Artificial Intelligence XXXVIII*. Lecture Notes in Computer Science, 13101. 41st SGAI International Conference on Artificial Intelligence, AI 2021, Cambridge, 14.12.2021-16.12.2021. Cham, Switzerland: Springer. . The final authenticated version is available online at: http://www.springer.com/gp/book/9783030910990

Whilst metaheuristics are able to produce well-performing solutions, the process by which they arrive at the suggested final solution remains largely unexplained to the end-user. This is problematic for a number of reasons. Firstly, a poorly understood solution will reduce the confidence an end user has in the highlighted solution that can have a knock-on effect in its uptake and implementation. Secondly, there can be additional criteria that were not included in the formal problem definition (e.g., aesthetics of a design, or having person X's working pattern accommodate their family life). Greater understanding of how the problem was solved could lead to a refinement of the problem that will, in turn, produce better performing solutions which can then lead to insight learned from this problem being applied to other similar problems. Thirdly, metaheuristics follow random processes that can lead to noise in the final outputs. It would be helpful to know what characteristics of a solution might simply be the result of this noise and can be eliminated with impacting solution quality. Finally, metaheuristics are very good at finding loopholes in the problem definition [19]. It would be helpful to know whether the solutions genuinely solve the problem or have just found a weakness in the specification.

These points together can reduce trust in an algorithm, meaning that suboptimal approaches may remain in practice that ultimately lead to wasted resources and inefficient practice. Whilst metaheuristics follow a relatively clear framework by which candidate solutions for a given problem are repeatedly generated, evaluated and modified, the search processes underpinning them are sufficiently complex as to be non-interpretable to humans and are regarded as a "blackbox". The problems metaheuristics solve are encompassed using a fitness function as a means to evaluate solutions, this is also difficult to interrogate directly for human understandable explanations and can take a variety of forms from truly blackbox examples driven by complex simulations to the more "grey box" optimisation examples where fitness is determined by a series of mathematical formulae.

In order for human users to adopt the recommendations of these automated decisions, it may be enough to provide explanation in the form of justifications for an individual outcome as opposed to interrogating and describing the inner processes by which an algorithm ultimately performs [1]. Research in explainability of deep learning can be seen to follow this type of approach, however there is little work of this type concerning metaheuristics, which address a different niche of problems than their deep learning siblings. When deciding whether to adopt a solution found by a metaheuristic, we suggest [7] that the decision maker is likely to want the focus to be on two main insights. Firstly: does the proposed solution actually solve the problem or have we stumbled upon a loophole in the problems definition; and, secondly: can we identify which characteristics of a solution are related to its performance and optimality for the problem, and which are simply an unrelated result of the random processes inherent to metaheuristics. The first of these points is more broadly applicable to any optimisation approach (including mathematical programming and brute-force optimisations), although here we focus purely on metaheuristics because of the ease with which the proposed solution approach (surrogate models) can be integrated with them.

Our focus within this paper is to offer some first steps towards making these insights. We propose a new approach to identify important characteristics of metaheuristic derived solutions to a series of binary benchmark problems, particularly which variables strongly influence solution quality and which are less important or can be ignored. The approach exploits *surrogate fitness functions* [9, 16], a well-established technique for improving metaheuristic search efficiency. A computationally cheap model is trained in parallel with the opti-

mization process, with the majority of the calls to a costly fitness function being replaced by a call to the surrogate model. Crucially, the surrogate is an explicit model of what the algorithm has learned about high-fitness solutions. Our approach takes a high-fitness solution returned by the metaheuristic and probes it with respect to the surrogate model to determine which variables are important, and their impact on fitness.

We begin in Section 2 by reviewing related work on trust and explanation in metaheuristics, before highlighting the role surrogates currently play in optimization problem solving and how we plan to utilise them for explainability in Section 3. The approach to mining the surrogate for problem information in the form of variable rankings is described in Section 4. We then demonstrate the approach with some easily understood simple benchmark problems being solved by a surrogate-assisted genetic algorithm in Section 5 and reflect on the results found in Section 6. Finally, we draw conclusions and set out our immediate plans for future work in this area in Section 7.

2 Related Work

Within the past decade, *Explainability* as a topic has seen a growing level of interest within the deep learning community as well as across other Machine Learning based approaches to problem solving. However there has been a noticeable gap in the research concerning metaheuristics and search algorithms with the closest examples being innovization proposed by Deb et al. in their 2014 paper [12] and illumination algorithms such as MAP-Elites discussed in a 2019 paper by Urquhart et al [25].

Deb et al [11, 12] proposed "innovization" to generate additional problem based knowledge alongside the normally generated near-optimal solutions, by identifying common principles among Pareto-optimal solutions for multi-objective optimisation problems. This is based on the fact most optimization techniques adopted within industry are used to yield a single or small selection of optimal solutions and builds on it to allow these optimization techniques to yield additional problem-based knowledge alongside the generated optimal solution(s) via the innovization process. This process takes a generated set of high-performing trade-off solutions and looks for common principles concealed within them. The philosophy here is that principles common amongst this set of high performing solutions will represent properties that ensure Pareto-Optimality and are by extension valuable properties related to the problem as a whole.

More recently Urquhart's 2019 paper looked at using MAP-Elites [21] to increase trust in metaheuristics. This paper was based on the common complaint from end users when presented with a solution constructed via a metaheuristic approach that they themselves had no role in the solution construction. In order to address this the authors applied MAP-Elites, which creates a structured archive of high performing solutions that are mapped onto a set of solution characteristics defined by the user such as cost or time. These solutions are generated by mutation and recombination and each solution is then assigned a bin within the solution space; should a generated solution be assigned to an already occupied bin solution only the solution with the highest fitness is retained within this bin. MAP-Elites provides the end user with a structured archive of high performing yet diverse solutions: Urquhart et al presented this archive via an interactive decision-making tool from which the user is ultimately responsible for choosing the preferred solution. It can be seen that MAP-Elites serves as

filter of the solution space for an end user that takes a large and impractical search space and refines it to a diverse set of high-quality solutions that are more readily usable for the end user to interact with.

In 2017, Gaier et al. [13] described using surrogate modelling alongside MAP-elites, in order to speed up the MAP-Elites process by reducing the need for such a large number of evaluations normally required for MAP-Elites to produce worthwhile results, this was known as Surrogate-assisted illumination (SAIL). This is achieved by integrating approximate models in the form of surrogates as well as intelligent sampling of the objective function in question. Similarly to the original MAP-Elites approach the search space is partitioned into bins each holding a design with a different configuration of feature values. A surrogate is constructed on an initial population of candidate solutions and their fitness scores, MAP-elites then produces solutions seen to maximise the acquisition function in every region of the feature space producing our acquisition map. New solutions are then sampled from this map and sampled with the additional observations being used to improve the model. Repetition of the process results in increasingly accurate models of high fitness regions of the feature space. The performance predictions are then used by MAP-elites in place of the original objective function to produce a prediction map of estimated near-optimal designs.

The present work differs from the above approaches. In contrast to *Innovization*, we target single-objective optimisation problems. In contrast to the illumination methods using *MAP-elites*, we do not seek to present many solutions to the end user but, rather, seek to explain a single solution that was chosen by the algorithm.

3 Background: Surrogate Models

Metaheuristics, including evolutionary algorithms, follow a well-known general framework whereby a population of solutions are generated at random, then evaluated against a *fitness function*, and new solutions are generated, usually biased towards solutions known to be high in fitness. The fitness function measures solution quality and is used to guide the search but for many applied problems is costly, especially when a long running simulation or human-in-the-loop evaluation is used.

A common approach to speed up these metaheuristics is to build a surrogate model of this fitness function [4,15,16] and use this to suggest where near-optimal solutions fall within the decision space. The surrogate model uses machine learning techniques to approximate the true fitness function, ideally retaining some of its key properties such as regions of high fitness, but is capable of evaluating solutions at a faster rate than the fitness function. As well as offering a quicker way to evaluate potential solutions the surrogate represents an explicit model of the population and thus can be exploited to gain insight into the algorithm's understanding of the problem at hand. The outline framework for a surrogate-assisted genetic algorithm that forms the subject of our study is illustrated in the flow diagram in Figure 1. In our work, the GA alternates between using the surrogate to evaluate solutions and using the true fitness function, with the surrogate being updated as new evaluations are carried out by the true fitness function. Many strategies for management and application of surrogate models exist and the interested reader is directed to [10, 15, 16] for more detail.

Surrogates are usually constructed through training a model in parallel with the optimization run, with the motivation for their use being an improvement

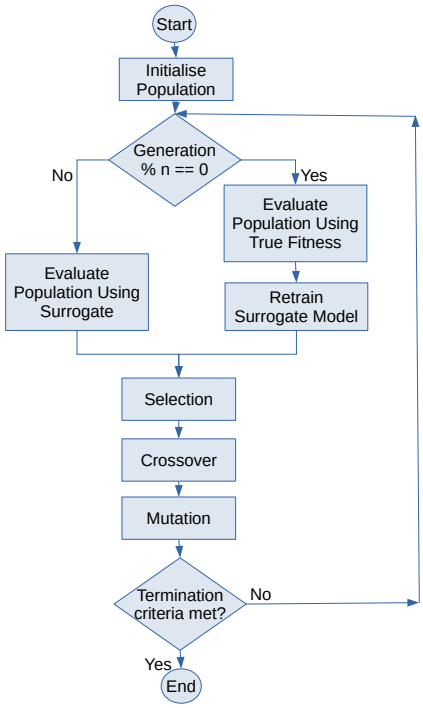

Fig. 1. The outline of the surrogate assisted genetic algorithm that we use on our experiments. In our study, n==5, i.e., the surrogate is retrained every 5 generations.

in the speed of our search. For this reason any extra problem knowledge we are able to extract from them can largely be viewed as "free" in terms of additional resources and CPU time at least. The explicit model of the population in the form of our surrogate can then be mined, similarly to regression analysis in principle [14, 17, 20]. This mining of models has already been demonstrated when looking at Estimation of Distribution Algorithms (EDAs), which construct probabilistic models reflecting the distribution of high fitness solutions within the population, then sample this distribution to yield solutions likely to be high in fitness. With existing work highlighting how useful information can be extracted from these probabilistic models [3, 6], it has been observed [7] within some real world problems the additional information gleamed from the EDA can prove just as advantageous as the optimisation results themselves.

It is important to note that the usefulness of this additional information is highly problem dependent and entirely reliant on the nature of the surrogate model used. More opaque black box modelling approaches involving neural nets are likely to be much harder to mine than any of the more transparent methods like functions derived using linear regression [23]. Our major hypothesis is that some semblance of explanation for the global optimum can be provided in the form of variable importance rankings, extracted by mining surrogate fitness models and visualisation of the results.

Data: $x = (x_0 \dots x_n), x_i = \{0, 1\}$, near-optimal solution found by GA
Data: $S(X) \rightarrow f$, surrogate fitness function to estimate fitness f of a solution X
Result: $C = (c_0 \dots c_n), c_i \in \mathbb{R}$, absolute change to surrogate fitness for each variable in x

$C \leftarrow \emptyset$;
$f_{org} \leftarrow S(x)$ `/* surrogate fitness of solution */`
for $i = 0$ **to** $n - 1$ **do** `/* for each variable` x_i `*/`
  $x_i \leftarrow (x_i + 1) \bmod 2$ `/* flip variable` x_i `*/`
  $\hat{f}_i$ `/* surrogate fitness of mutated solution */`
  $C \leftarrow |c_i|$ `/* add to list */`
end

Algorithm 1: Method for probing variables in a solution with respect to the surrogate fitness function

4 Methodology: Mining the Model

The proposed approach to determine and quantify the importance of individual variables takes the form of local sensitivity analysis [26], whereby the optimal solutions are perturbed and the resulting mutants tested against the surrogate model, following Algorithm 1. Once the metaheuristic run has completed and a surrogate fitness function built, the best-found solution over the metaheuristic run was retained. Due to the relative simplicity of binary benchmark problems this often lay on or near the true globally optimal solution s. Each binary variable within s was then flipped to form a neighbour $\hat{s}$. The fitness of $\hat{s}$ was calculated using the surrogate, allowing us to understand not only the importance a given variable may have on the overall fitness of a solution but, more importantly, whether our surrogate was able to accurately capture and utilize this information.

This method also brings insight into whether the problem contains variables which, when altered, would not impact negatively on performance. This is less relevant for benchmark problems but is significant should this approach be applied to more real life and noisy problems. This mutated fitness score was then compared to that of the original solution and the amount to which it has risen or fallen recorded. Once a value for each variable change was recorded the variables themselves were ranked in terms of absolute effect on fitness and these together constituted an experimental run. This entire process of GA and surrogate generation and subsequent surrogate model mining for variable importance was repeated 50 times to give us a set of repeat runs. From these 50 runs, the median absolute change in fitness and median ranking for each variable was calculated. An example of the outcome of these rankings can be seen in Figure 3 and Figure 4.

The idea here is that the importance of each variables is derived from the surrogate fitness function. This surrogate is biased towards high-quality solutions that have been visited by the metaheuristic, and so reflects something of the metaheuristic's understanding of the problem. Probing in this way also costs very little, because no additional calls to the fitness function are required.

5 Experiments

As a baseline, we focus on three well-known bit-string encoded benchmark functions (BinVal, AltOnes and Checkerboard) and use three problem sizes for each. In the case of BinVal and AltOnes these were 20, 50 and 100, whilst in the case of Checkerboard, a square structure was required so problems sizes of 25, 64 and 100 were used. These functions are well understood and, in each function, we have direct control over the interactions between variables and the importance of variables.

5.1 Benchmark Problems

Our experiments focus on three benchmark functions using a bit string representation. The three were chosen to give some basic variation in the importance attached to each variable and the presence or absence of interactions.

BinVal was chosen because the problem variables have a clear rank ordering of importance. In the standard version of this problem, fitness is simply the integer represented by the bit string in binary. In order to ensure that the importance of the least significant bits on fitness was not infinitesimal compared to the most significant bits, we flattened the growth in weight applied to each bit by changing the base c from 2 in the original benchmark to 1.1 in our experiments.

$$f(x) = \sum_{i=1}^{n-1} c^{ix_i} \tag{1}$$

AltOnes applies equal importance to all variables, and introduces interactions. Fitness is the count of directly neighbouring pairs of bits in the bit string that have differing values.

$$f(x) = \sum_{i=0}^{n-1} \delta(x_i, x_x + 1) \tag{2}$$

Checkerboard also introduces bivariate interactions, and though these are equally weighted in the fitness function, implicitly those in the centre of the grid have greater impact on fitness [5]. Solutions represent the rows of a s x s grid concatenated into one string, and the objective is to realise a grid with a checkerboard pattern of alternating 1s and 0s:

$$f(x) = 4(s-2)^2 - \sum_{i=2}^{s-1} \sum_{j=2}^{s-1} \left\{ \begin{array}{c} \delta\left(x_{ij}, x_{i-1j}\right) + \delta\left(x_{ij}, x_{i+1j}\right) \\ +\delta\left(x_{ij}, x_{ij-1}\right) + \delta\left(x_{ij}, x_{ij+1}\right) \end{array} \right\} \tag{3}$$

where δ is Kronecker's delta function,

$$\delta_{ij} = \left\{ \begin{array}{l} 1 \text{ if } i = j \\ 0 \text{ if } i \neq j \end{array} \right. \tag{4}$$

5.2 Experimental Procedure

When addressing each of the benchmark problems above, a similar experimental approach was undertaken. An initial random population of 100 candidate solutions was created and a Genetic Algorithm (GA) used with a view to determine a near-optimum solution. The GA has population size 100, tournament size 2, mutation gene rate of 0.1, and a crossover rate of 0.95 and ran for 100 generations. The GA also used elitism, with the single fittest solution being carried from one generation to the next without mutation.

The GA takes the initial population and iterates through a cycle of assessing the fitness of the population, selecting candidate solutions to take forward to the next newly formed population and breeding these solutions through a cycle of crossover and mutation, following the outline in Figure 1. The best found solution over the entire metaheuristic run is returned to the end user once completed with no true explanation as to how it arrived at a given solution. In order to try and capture some of this explainability a surrogate model was constructed using the current GA population and calculated fitness scores at different periods throughout the GA run with retraining occurring at set intervals. Clearly for these simple benchmark functions the surrogate is unnecessary for the purpose of speeding up the process; but we aim to demonstrate its utility for mining the importance of problem variables.

For each problem, three different surrogate models were used in order to see how the problems performed over a variety of surrogate model types: linear regression; decision tree; and random forest. These were trained using the WEKA Java machine learning library[1], and used with the default hyperparameter configurations. In principle better modelling could be achieved by tuning of the hyperparameters, but as the purpose of this study is a proof of concept for mining the model, we have left this stage to future work for now.

6 Results and Discussion

Before we go onto to discuss the results, it is important to note that one of the surrogates failed to accurately grasp enough about any of the problems in question: this was the decision tree in the form of a WEKA RepTree. In the smaller sized instances of BinVal, RepTree was able to ascertain that greater importance lay in the lower numbered variables, nearer the beginning of the bit string but failed to pick up any further knowledge about variable importance. With both AltOnes and Checkerboard this surrogate did not pick up sufficient information to be of use to us. An example of the results that RepTree generated for BinVal can be seen in Figure 2

Looking firstly at the BinVal benchmark problem over the three problem sizes with linear regression and random forest as the surrogate models, we see similar results as shown in Figure 3. The average bearing a variable has on fitness diminishes as the variable number increases. This is further reinforced by Figure 4, which shows that, as the variable number increases, its rank and, by extension, its bearing on fitness, can be seen to reduce. This agrees with our understanding of the BinVal problem and paints a clear picture that surrogate mining has found information directly relevant to the quality of solutions in the optimisation problem at hand.

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

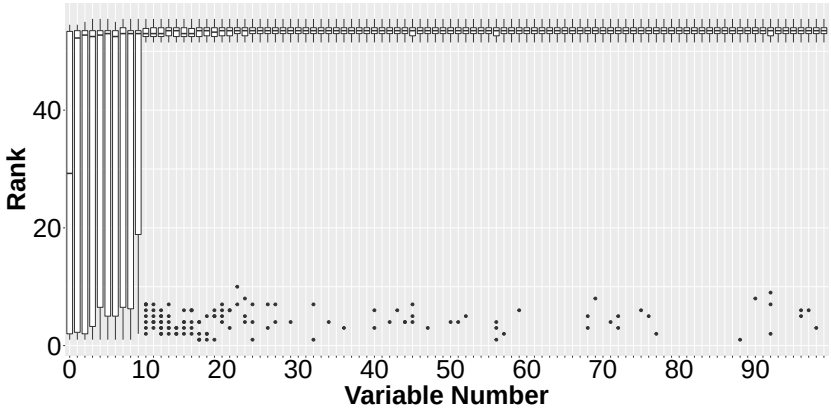

Fig. 2. This plot shows the distribution of importance ranks allocated to each variable over 50 runs (lower values indicate closer to 1st place; i.e., more important). In this case, showing the ability of the surrogate decision tree (RepTree) to capture problem knowledge. Even the most important variables (left-most) are often ranked far from first place, and after the tenth variable, rarely was any importance detected.

Similarly when looking at the AltOnes problem, where equal importance is seen across the suite of variables, it was observed both linear regression and random forest surrogates were able to detect that all variables shared the same level of change in absolute fitness. In each case, each variable had an impact of $1/n$ on overall fitness, indicating they shared a similar degree of importance in solving the problem. This was further reinforced when the variables were ranked in terms of importance and all fell on or around half of the total number of variables in a given problem, as shown in Figure 5, where a linear regression model was used as the surrogate.

Finally when looking at the Checkerboard problem, we see a similar pattern to AltOnes, with all variables sharing a similar relationship to fitness and their alteration resulting in a similar fitness change. When ranked, all variables fall on or around halfway, as seen in Figure 6. The only caveat to this is small regular pockets of apparently slightly more influential and higher ranked variables (there is a lot of noise but roughly speaking, 13–17, 21–28, 55–61, 73–79). These were determined to be variables that lay nearer the central region of our hypothetical Checkerboard and thus had a higher number of neighbours surrounding them. At this stage, we are only concerned with the individual variable importance but, in future, more work could be done to investigate this neighbourhood element of the problem and how best to capture it.

7 Conclusion

This paper serves as an initial indicator for the potential to use a collection of machine learning algorithms to aid in the explainability of optimization processes utilizing metaheuristics. It focuses on the reporting of relationships between the location of variables within a given candidate solution and their bearing on this solutions ability to solve benchmark problems. It applies sensitivity analysis to

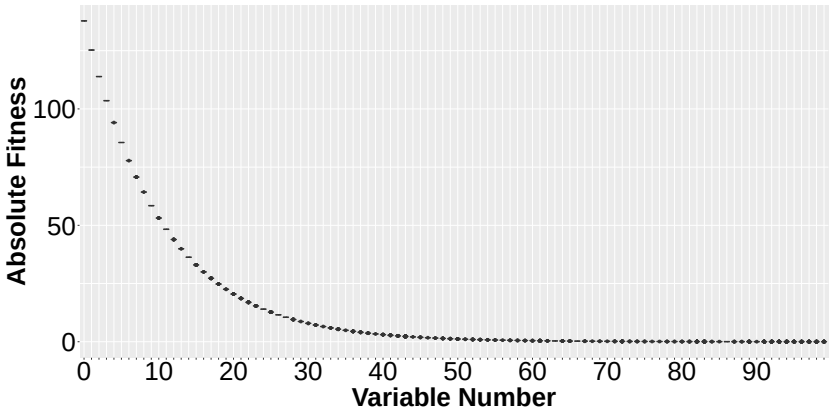

Fig. 3. The absolute fitness change resulting when individual variables are flipped with the BinVal problem of size 100, using a linear regression surrogate. Each box is the distribution of absolute fitness changes detected for each variable by mining the surrogate. A smooth drop off in importance can be observed.

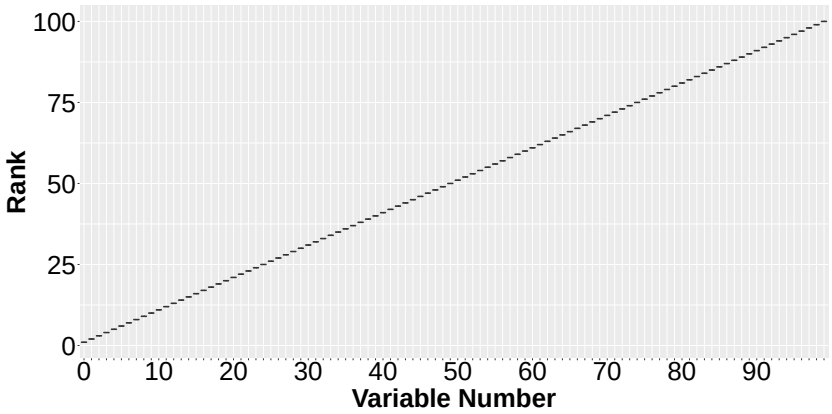

Fig. 4. The rankings of variables within the BinVal problem of size 100, using the linear regeression surrogate. A clear and consistent rank ordering of importance corresponding to the location of each variable can be observed.

a proposed solution with a surrogate model to achieve this. Some immediate benefits of this can be quickly seen:

– Outside of traditional benchmark problems, gaining knowledge about variable rankings and the sensitivity of individual variables can allow for solutions to be fine tuned for factors and constraints not initially considered during the optimization run, whilst retaining an idea of the effect a given change will have on the optimality of a given solution.

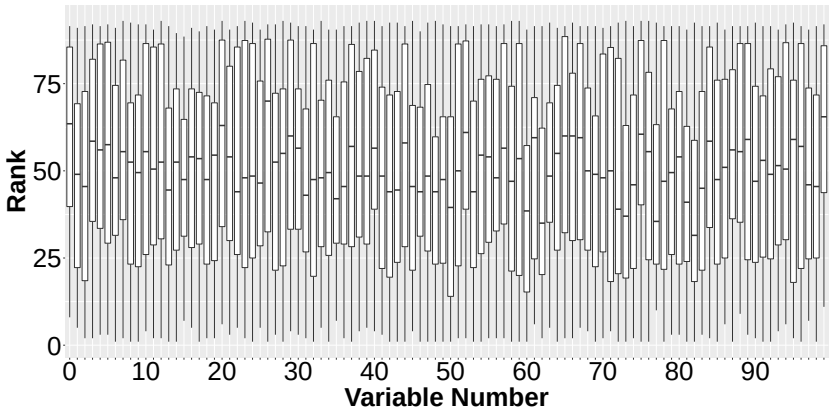

Fig. 5. Showing the variable ranking resulting when individual variables are flipped with the AltOnes problem of size 100

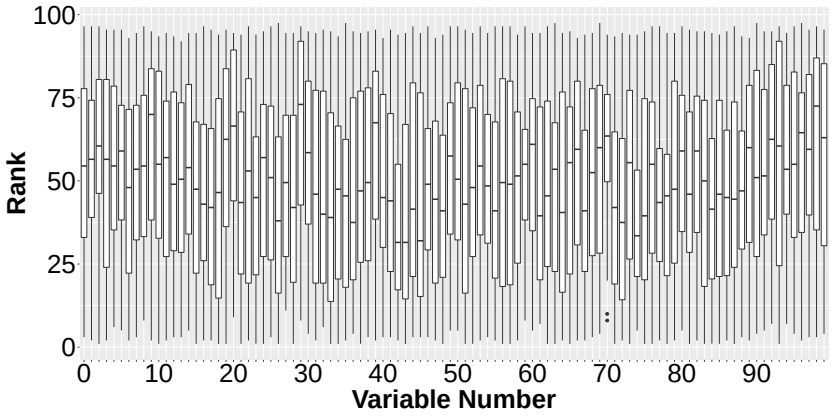

Fig. 6. Showing variable rankings within checkerboard problem

- More obviously, if the optimal solution / set of solutions returned by the metaheuristic run agrees with the conclusions drawn from our surrogate model, it gives a decision maker more confidence in the optimality of solutions in question and increases the likelihood they will be utilized.
- Finally in the case of complicated or long running optimization algorithms, the surrogate is able to point towards the likely location of well performing solutions and by extension likely global optima. This has a possible two-fold benefit. Firstly, allowing us to narrow the search space to filter out solutions we know to be of poorer quality and focus on only the higher quality solutions. Secondly, the potential to indicate where a stochastic technique such as those employed by many metaheuristic optimization techniques may have neglected or passed over a region of high fitness or perhaps even the globally optimal solution itself.

Comparing the results of the two successful surrogates (linear regression and random forest) we see very similar variable rankings generated for each of the benchmark problems. In the case of BinVal both showed a clear sliding level of variable importance as a variable is further from the beginning of the bit string. Similarly, in the case of AltOnes and Checkerboard a generally similar level of importance was shared by all variables within the bit string as expected with our own understanding of these problems.

We have looked at using surrogate fitness functions as a way to obtain additional information pertaining to variable importance rankings within a subset of common binary benchmark problems. Going forward from here additional focus needs to remain on introducing and disseminating the concept of applying surrogate model mining to a much wider range of problems particularly those traditionally solved using processes derived from metaheuristic approaches. This work has further strengthened the argument for using surrogates as a means of gaining additional problem knowledge and explaining metaheuristic optimisation results in the future.

Future work following this line could involve applying similar methods to real world and non-binary problems as well as multi-objective problems. We also wish to investigate whether taking surrogate models built at different stages of a GA run (after a set number of generations) will affect the surrogate's performance. Furthermore, we will seek to extend our current methods to work on problems involving interactions, such as the known neighbourhood structure in this benchmark problem, or following a structure learning procedure such as those set out in [2].

References

1. Adadi, A., Berrada, M.: Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). IEEE Access **6**, 52138–52160 (2018). doi:10.1109/ACCESS.2018.2870052
2. Brownlee, A.E.I., McCall, J.A.W., Shakya, S.K., Zhang, Q.: Structure Learning and Optimisation in a Markov-network based Estimation of Distribution Algorithm. In: Proc. IEEE CEC. pp. 447–454. IEEE Press, Trondheim, Norway (2009)
3. Brownlee, A.E.I., Regnier-Coudert, O., McCall, J.A.W., Massie, S.: Using a Markov network as a surrogate fitness function in a genetic algorithm. In: Proc. IEEE CEC. pp. 4525–4532. Barcelona (2010)
4. Brownlee, A.E.I., Woodward, J., Swan, J.: Metaheuristic design pattern: Surrogate fitness functions. In: MetaDeeP Workshop, Proc. GECCO Companion. pp. 1261–1264. ACM Press, Madrid, Spain (2015)
5. Brownlee, A., McCall, J., Zhang, Q.: Fitness Modeling With Markov Networks. IEEE T. Evolut. Comput. **17**(6), 862–879 (2013)
6. Brownlee, A.E.I., Regnier-Coudert, O., McCall, J.A.W., Massie, S., Stulajter, S.: An application of a GA with Markov network surrogate to feature selection. Int. J. Syst. Sci. **44**(11), 2039–2056 (2013)
7. Brownlee, A.E.I., Wallace, A., Cairns, D.: Mining Markov Network Surrogates to Explain the Results of Metaheuristic Optimisation. In: Proceedings of the SICSA XAI Workshop 2021. CEUR Workshop Proceedings, Robert Gordon University, Aberdeen, UK (2021)
8. Brownlee, A.E., Weiszer, M., Chen, J., Ravizza, S., Woodward, J.R., Burke, E.K.: A fuzzy approach to addressing uncertainty in Airport Ground Movement optimisation. Transportation Research Part C: Emerging Technologies **92**(April), 150–175 (2018). doi:10.1016/j.trc.2018.04.020, https://doi.org/10.1016/j.trc.2018.04.020

9. Brownlee, A.E., Wright, J.A.: Constrained, mixed-integer and multi-objective optimisation of building designs by NSGA-II with fitness approximation. Applied Soft Computing **33**, 114–126 (2015)
10. Chugh, T., Rahat, A., Volz, V., Zaefferer, M.: Towards better integration of surrogate models and optimizers. In: Bartz-Beielstein, T., Filipič, B., Korošec, P., Talbi, E.G. (eds.) High-Performance Simulation-Based Optimization, pp. 137–163. Springer International Publishing, Cham (2020). doi:10.1007/978-3-030-18764-4_7
11. Deb, K., Srinivasan, A.: Innovization: Discovery of innovative design principles through multiobjective evolutionary optimization. In: Knowles, J., et al. (eds.) Multiobjective Problem Solving from Nature: From Concepts to Applications. pp. 243–262. Springer, Berlin, Heidelberg (2008)
12. 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. Applied Soft Computing **15**, 42–56 (2014). doi:10.1016/j.asoc.2013.10.011
13. Gaier, A., Asteroth, A., Mouret, J.B.: Data-efficient exploration, optimization, and modeling of diverse designs through surrogate-assisted illumination. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 99–106 (2017)
14. Hauschild, M., Pelikan, M.: An introduction and survey of estimation of distribution algorithms. Swarm Evol. Comput. **1**(3), 111 – 128 (2011)
15. Jin, Y.: A comprehensive survey of fitness approximation in evolutionary computation. Soft computing **9**(1), 3–12 (2005)
16. Jin, Y.: Surrogate-assisted evolutionary computation: Recent advances and future challenges. Swarm Evol. Comput. **1**(2), 61–70 (2011)
17. Larrañaga, P., Lozano, J.A.: Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation. Kluwer, Boston (2002)
18. Le Bras, P., Robb, D.A., Methven, T.S., Padilla, S., Chantler, M.J.: Improving user confidence in concept maps: Exploring data driven explanations. Conference on Human Factors in Computing Systems - Proceedings **2018-April** (2018). doi:10.1145/3173574.3173978
19. Lehman, J., Clune, J., Misevic, D.: The surprising creativity of digital evolution: A collection of anecdotes from the evolutionary computation and artificial life research communities. Artificial Life **26**(2), 274–306 (2020). doi:10.1162/artl_a_00319
20. Lozano, J.A., Larrañaga, P., Inza, I., Bengoetxea, E.: Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms (Studies in Fuzziness and Soft Computing). Springer-Verlag (2006)
21. Mouret, J.B., Clune, J.: Illuminating search spaces by mapping elites. arXiv preprint arXiv:1504.04909 (2015)
22. Ochoa, G., Christie, L.A., Brownlee, A.E., Hoyle, A.: Multi-objective evolutionary design of antibiotic treatments. Artificial Intelligence in Medicine **102**(October 2019), 101759 (2020). doi:10.1016/j.artmed.2019.101759
23. Rodriguez Rafael, G.D., Solano Salinas, C.J.: Empirical study of surrogate models for black box optimizations obtained using symbolic regression via genetic programming. In: Proc. GECCO Companion. pp. 185–186. ACM (2011)
24. Sajja, P.S.: Examples and applications on genetic algorithms. In: Illustrated Computational Intelligence, pp. 155–189. Springer (2021)
25. Urquhart, N., Guckert, M., Powers, S.: Increasing trust in meta-heuristics using MAP-elites. In: Proc. GECCO Comp. pp. 1345–1348 (2019)
26. Wright, J.A., et al.: Multi-objective optimization of cellular fenestration by an evolutionary algorithm. J. Build. Perform. Sim. **7**(1), 33–51 (2014)