Received: 12 April 2022 | Revised: 25 July 2023 | Accepted: 15 September 2023

DOI: 10.1111/exsy.13494

ORIGINAL ARTICLE

Expert Systems WILEY

Towards explainable metaheuristics: Feature extraction from trajectory mining

Martin Fyvie[1] | John A. W. McCall[1] | Lee A. Christie[1] | Alexander E. I. Brownlee[2] | Manjinder Singh[2]

[1]National Subsea Centre, Robert Gordon University, Aberdeen, UK

[2]Division of Computing Science and Mathematics, University of Stirling, Stirling, UK

Correspondence
Martin Fyvie and John A. W. McCall, National Subsea Centre, Robert Gordon University, Aberdeen, UK.
Email: M.Fyvie@rgu.ac.uk and j.mccall@rgu.ac.uk

Alexander E. I. Brownlee, Division of Computing Science and Mathematics, University of Stirling, Stirling, UK.
Email: alexander.brownlee@stir.ac.uk

Funding information
BT Group; The Data Lab

Abstract

Explaining the decisions made by population-based metaheuristics can often be considered difficult due to the stochastic nature of the mechanisms employed by these optimisation methods. As industries continue to adopt these methods in areas that increasingly require end-user input and confirmation, the need to explain the internal decisions being made has grown. In this article, we present our approach to the extraction of explanation supporting features using trajectory mining. This is achieved through the application of principal components analysis techniques to identify new methods of tracking population diversity changes post-runtime. The algorithm search trajectories were generated by solving a set of benchmark problems with a genetic algorithm and a univariate estimation of distribution algorithm and retaining all visited candidate solutions which were then projected to a lower dimensional sub-space. We also varied the selection pressure placed on high fitness solutions by altering the selection operators. Our results show that metrics derived from the projected sub-space algorithm search trajectories are capable of capturing key learning steps and how solution variable patterns that explain the fitness function may be captured in the principal component coefficients. A comparative study of variable importance rankings derived from a surrogate model built on the same dataset was also performed. The results show that both approaches are capable of identifying key features regarding variable interactions and their influence on fitness in a complimentary fashion.

KEYWORDS
evolutionary algorithms, explainability, PCA, population diversity

1 | INTRODUCTION

Population-based metaheuristics such as evolutionary algorithms (EA) have seen an increase in applications that involve end-user interactions. These are typically found in areas such as transport and logistics (Abduljabbar et al., 2019), medical applications (e.g., Ochoa et al., 2020) and engineering (Brownlee et al., 2020). These metaheuristics use high-level search metaphors, often nature-inspired, implemented as complex, non-deterministic sequences of operations on solution populations, leaving the solution process intractable to explanations based on analogies to human reasoning. This increase has highlighted the need for the decision processes behind these system to be more understandable and

interpretable by end-users. This, in turn, may help build a level of trust in the solutions generated by these systems, as seen in conclusions and recommendations of the public health genetics foundation (Ordish et al., 2020).

The field of explainable artificial intelligence (XAI) aims to develop techniques that enable us to systematically derive knowledge or decision-making insights from machine learning models. With this adoption of AI methods such as EAs, early studies undertaken help define explainability's core concepts and taxonomies (Adadi & Berrada, 2018; Arrieta et al., 2020). Numerous definitions of explainability exist, and for the purpose of this article, we mean it to regard the explanation of otherwise black-box systems.

Interpretability of models and AI systems is key to achieving this goal. As noted by Murdoch et al. (2019), this can be difficult to define but can be understood broadly as the 'extraction of relevant knowledge from a machine-learning model concerning relationships either contained in data or learned by the model' which also applies to metaheuristics. Two significant metaheuristic approaches are genetic algorithms (GA) and estimation of distribution (EDA) algorithms. Both explore a solution space using a population-based search metaheuristic. As a GA explores the search space its natural evolution inspired operators are used to evolve from one set of solutions to the next. Each successive population is created through natural-selection, breeding and mutation of higher (or lower) fitness candidate solutions with an overall aim of improving population fitness until convergence has occurred. In doing so, these populations of solutions represent an implicitly learned interpretation of the structure of the optimisation problem being solved. The EDA can be considered an iteratively-focused estimation of the distribution of high-value solutions. Each successive population of solutions is created by the sampling of an explicit probabilistic model of the problem structure, updated based on the distribution of candidate solutions. Here, problem structure refers to the graphical dependency relationship between solution variables and their joint influence on fitness value. This has been variously interpreted in the EDA literature through Bayesian, Markov or Gaussian probabilistic models (Shakya et al., 2012).

While the search processes of these metaheuristics can be easily explained, their non-deterministic nature means that the quality of the end solutions returned cannot be predicted or explained. We seek to mine features from these processes that are explanatory of why the final set of solutions arrived at by these metaheuristics are of high quality.

For both GAs and EDAs, each generation of solutions created can be considered to contain implicit knowledge gained by an algorithm at specific points in the optimisation processes. In this article, we collect all visited solutions in the search space, separated by generation and optimisation run. Each generation can be said to represent the general position of the algorithm as it samples multiple non-distinct solutions. Using this method, we can define a trajectory as a collection of solution populations, ordered by generation, representing the movement of the search algorithm over the course of an optimisation run.

We hypothesize that the trajectories generated in this process can be mined for valuable information regarding population changes that can aid in generating explanations for our end-user target audience of researchers and algorithm developers. Our approach involves the projection of the high-dimension solutions space to a lower dimension space that can be used to generate more easily understood visualisations and provide a possible source of new metrics. This is accomplished through the application of principal components analysis (PCA). The results can then be used to generate explanations with the aim of increasing an end-user understanding of the problem being solved and the process by which the algorithms have arrived at the provided set of solutions. We also explore a further method of explanation support involving the use of surrogate modelling to derive feature importance values from the population of solutions. This process attempts to describe the variation in the internal fitness model generated by a surrogate over the course of an optimisation run to isolate feature importance.

The rest of the article is structured as follows. Section 2 highlights related work in the area of search trajectory analysis through applied visualisation techniques. Section 3 outlines the concept of entropic divergence and how this is used as measure of population diversity change. This is followed by the background method used to translate the algorithm trajectories into a lower dimension space as well as the new metrics derived from that space for comparison to the entropic divergence measurement. Section 4 outlines the experimental setup that covers the algorithms used, the problems they were used to solve and the construction of a surrogate model based on the metaheuristic trajectories. Section 5 highlights the results and performance between the newly created population metrics and the entropic divergence are shown and discussed. This section also highlights the findings regarding problem structure, post-PCA projection variable loadings, and a comparison of the PCA loadings with the results of mining surrogate models. Section 6 sets out our conclusions based on the results of these tests and outlines planned continuations of this work.

2 | RELATED WORK

In this section, we review work related to the use of algorithm search trajectories in the XAI field in two categories: the visualisation of search trajectories using dimension reduction and analysis of the search landscape, and the deriving of feature importances from generated solutions.

2.1 | Visualisation

Previous work covering the visualisation of algorithm trajectories using PCA can be seen in Collins (2003). In this work, PCA is used to attempt to attribute algorithm design to the search path taken and compare algorithm solution quality. Other examples of work involving the exploration of

algorithm paths via dimension reduction include work by Pohlheim (2006) in which Sammon mapping is explored as a method of reduction for visualisation and Michalak (2019) in which Euclidean embedding is applied. These works focus on the visualisation of an algorithm through the search space as a means of exploration and how these visualisations may, in some way, aid in the understating of the search methods employed. Their aims are somewhat similar to those in this article however our approaches also have the potential to be used in extracting features from algorithm paths.

Landscape analysis, a survey of which can be seen in Malan (2021), can be considered a key set of tools for the generation of explanations regarding algorithm behaviour. Recent methods of visualisation and analysis of the landscape involve the creation of search trajectory networks (Ochoa et al., 2021a; 2021b) which highlight algorithm behavioural differences through their search of the problem landscape using Local Optima Networks. These networks chart the ability of a search algorithm to move from one local optimum or basin of attraction to another. By doing so, algorithm search behaviours can be compared by reducing the search landscape to a connected network of visited local optima. The concept of STNs has since been extended to multi-objective evolutionary algorithms (Lavinas et al., 2022). More recently, a move towards 'explainable landscape-aware optimization' and analysis (Trajanov et al., 2021; 2022) developments have seen a move towards the prediction of algorithm performance based on landscape analysis.

2.2 | Feature importance

The method of feature extraction explored in this article is the use of surrogate models (Jin, 2011). This involves the creation of an explicit model of the solution populations to attempt to capture any sensitivities the fitness function has to specific problem variables. The methods used in this article are similar to those of Singh et al. (2022) and Wallace et al. (2021) in which feature importance is captured by a surrogate model trained at various stages of a search trajectory. These features can then help support explanations by highlighting learning steps in the algorithm run and identifying solution variable patterns that describe the fitness function. Further approaches that aim to develop some level of understanding regarding algorithm behaviour include sensitivity analysis (SA). In SA, the aim is to measure how different values of an independent variable impact a particular dependent variable under a given set of assumptions. Applications of this method can be seen in Cortez and Embrechts (2013) and Wright et al. (2012), In which SA is used to calculate solution fitness sensitivity to changes in variable values. Similarly, SA and visualisation techniques have been used together to provide valuable insights into feature importance, making models more interpretable and transparent (Cortez & Embrechts, 2011; 2013). The introduced visualisation techniques aim to further enhance the understanding of these models.

GAs, however, are often utilised as tools to enhance XAI techniques such in fuzzy logic systems Duygu Arbatli and Levent Akin (1997) and counterfactual creation Sharma et al. (2020) and are rarely the focus of such analyses. Thus, the contributions of this article are as follows:

1. A proposal to use PCA to mine and visualise explanations of metaheuristic behaviour from search trajectories.
2. Application of the approach to two metaheuristics (GA and EDA).
3. Analysis of the effects of algorithm configuration choice on recovered features by varying the selection operators used.
4. Comparison with a related approach to mining explanations, surrogate modelling, that generates fitness models on the same algorithmic runs.

3 | FEATURE EXTRACTION

As higher quality solutions are found by the population-based metaheuristics used in this article, population diversity tends to reduce as the search converges on a set of optimal or near-optimal solutions. As the search process continues, key learning's regarding the fitness function are represented implicitly by the populations of solutions. By monitoring the change in population diversity throughout the optimisation runs, we aim to generate a set of features capable of relating variable patterns to the fitness function. There exist several metrics used to measure the change in population diversity over the course of an optimisation run by these search methods. In Hien and Hoai (2006) a brief review of many of these metrics can be found. The metrics covered include the hamming distance, the sum of pair-wise comparison in the number of variable differences between two solutions and moment of inertia Morrison & Jong, 2001. This provides a '…single method of computing population diversity that is computationally more efficient than normal pair-wise diversity measures for medium and large sized EA problems'. An alternative method for population diversity measurement is the Kullback–Leibler entropic divergence (KL_d) distance measure. This measure is based around the concept of information gain and Shannon's entropy (Shannon, 1948) in which '…the entropy of a random variable is defined in terms of its probability distribution' (Cutello et al., 2010). The authors of that work also note that this method has the potential to quantify the transition from the exploration to exploitation. The ability to associate changes in energy to the generation of higher quality solutions by an algorithm supports our search for explanatory features and so KL_d was selected for this article.

3.1 | Entropic divergence

The Kullback–Leibler entropic divergence is a population diversity distance as defined Equation (1).

$$KL_d(P\|Q) = \sum_{x \in \mathcal{X}} P(x) \log\left(\frac{P^{(t)}(x)}{Q^{(t_0)}(x)}\right), \tag{1}$$

where P and Q are vectors of marginal probabilities for two different populations in the trajectory (MacKay, 2003).

Using Equation (1) it is possible to track the information gain from the initial population generated by the algorithms as $Q(x)$ remains constant as the probability vector of the generation $t = 0$. This metric is called the 'global learning' and it measures the total information gain from initial population to the population at any given $\boldsymbol{t}$. The expected behaviour for this metric is to increase over time until a 'steady state' is arrived at.

It is also shown in Cutello et al. (2010) that it is possible to use this equation to measure the information gain between two consecutive populations where $Q^{(i)}(x)$ and $P^{(i+1)}(x)$ are used. This is known as 'local learning' with the values calculated expected to increase until the search has reached a near-optimal solution. At this point, as the remaining population diversity is reduced and the population of solutions converges, the 'local' information gain values should show a bell-curve or peak. This peak may show the point at which the algorithm transitions from primarily exploring the search space to utilising implicit problem structure to improve solution quality. When this happens, it is expected that the values will decrease as the diversity within the population decreases.

3.2 | Principal components analysis

The benchmark problems used in this article are comprised of relatively simple, low-level variable interactions. To generate a large enough dataset consisting of a sufficient number of generations before convergence a problem size of 40 was selected. As our aim is to detect features with some explanatory power relating to the fitness function, PCA was used to for the decomposition of these search trajectories. This decomposition allowed us to reduce the overall dimensionality of the resulting datasets for visualisation purposes. The process of reducing the dimensionality of the algorithm trajectory population datasets can be achieved through the use of PCA. This allows us to project the higher dimensional space of the solutions to a three-dimensional subspace as 'PCA produces linear combinations of the original variables to generate the axes, also known as principal components, or PCs' (Holland, 2019).

The components are an mutually orthogonal set of vectors calculated such that the first will account for the maximum possible variation in the original data. Each subsequent component will account for the next highest possible variation. This can be continued until the number of components is equal to the dimensions of the original data. A summary of the calculation of linear combination and weights from Holland (2019) can be seen in Equation (2).

$$\begin{aligned} PC_1 &= a_{11}X_1 + a_{12}X_2 + \cdots a_{1p}X_p \\ PC_1 &= a_1^t X \\ PC &= XA. \end{aligned} \tag{2}$$

Here we show how a component is created from the linear combination of the variables in a solution and the resulting eigenvectors. In Equation (2), **X** is the matrix of variables in a solution and matrix **A**denotes the matrix of eigenvectors of the PCA decomposition. The elements of the matrix **A** are also known as the 'loadings.' These describe the contribution each variable has to a given component. These values can be either positive or negative, representing the correlation between the component and variable. The larger the absolute value of the loading, the stronger that relationship is. This feature of the components can be used to help explain aspects of the optimisation problem solved. The resulting datasets were then mined with the intent of finding these features.

3.3 | Sub-space derived features

3.3.1 | Population cluster centres

The dimensionality of the dataset is reduced through the projection of the data into a lower dimension set based on the principal components calculated using Equation (2). In this article, we project into a three-dimensional sub-space to help visualise the population as a cluster. This is achieved by creating a three-dimensional subspace from the first three components calculated. Figure 1a is an example of a single search trajectory visualisation post-PCA projection. Each point in Figure 1a represents a solution. The centre of each population of solutions can be found by

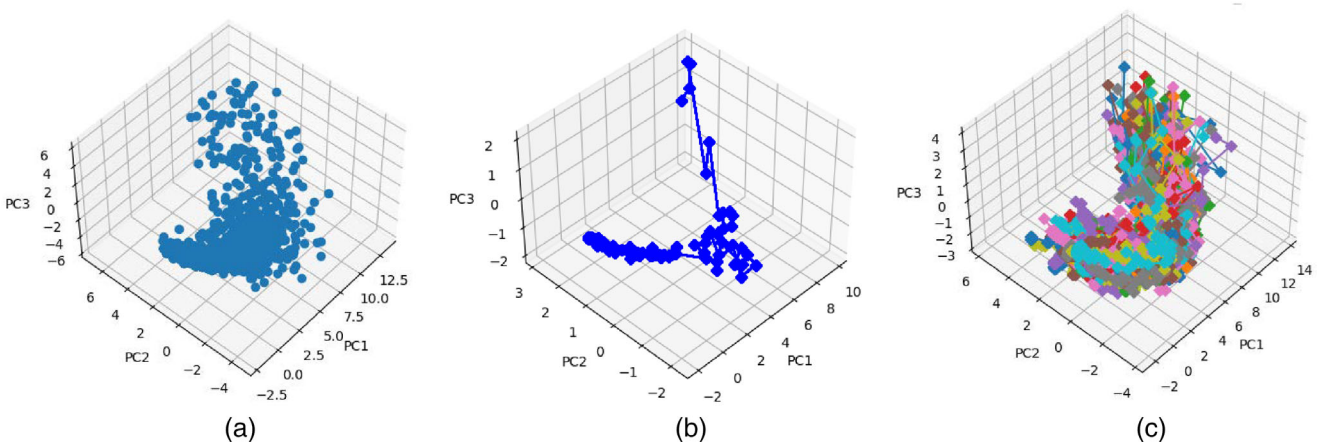

FIGURE 1 Illustrative example of algorithm trajectory projected along top three principal components. (a) Single run cloud (b) single run centres (c) 100 run centres.

calculating the point that minimizes the sum of squared Euclidean distances between itself and each point in the set. This process results in a set of points in 3D space that represents the algorithm's search trajectory from the initial population to the final population as shown in Figure 1b,c is an illustrative example of 100 search trajectories presented in the same subspace. Here, the centres of each population are used as in Figure 1b, with each colour representing a separate optimisation run and resulting search trajectory.

It is important to note that this method does not chart the algorithm trajectory in objective space and does not explicitly reflect the fitness landscape but instead can be used to measure the direction and magnitude of changes in population diversity after being projected into this subspace.

3.3.2 | Angle from origin

The projection of the search trajectories into the lower dimension, PCA derived subspace results in a series of population centres or 'centroids.' These represent the overall position of the a given generation of solutions in the subspace. Each optimisation run will generate a series of these centroids, ordered by generation number, as seen in Figure 1b. As these sets are ordered by generation number, it is possible to calculate features that depend on this ordering, much like the local and global information gain values. The angle from trajectory origin measures the angle between the centroid of the initial starting population in the trajectory and each subsequent population that was created. Each of the two points in the space are represented by the centroids coordinates as a vector of (PC1, PC2, PC3) coefficients in place of x, y and z coordinates. In order to calculate the acute angle α between two vectors we use the inverse cosine of vector products as seen in Equation (3). It is also possible to calculate the angle between clusters by altering Equation (3) using C_i and C_{i+1}, where ($i <= 0 <= maxGen$). This allows for the angle between consecutive populations to be calculated.

$$\begin{aligned} \alpha \quad &= \arccos\left(\frac{C_0 \cdot C_i}{\|C_0\|\|C_i\|}\right) \\ C_0, C_i \ &= \text{Cluster Centroids}\,(x, y, z). \end{aligned} \tag{3}$$

4 | EXPERIMENTAL SETUP

Detailed in this section are the parameters used to generate the necessary datasets for analysis for the purpose of feature extraction as outlined in Section 3. Here we highlight the selection criteria used, algorithm run settings, problems sets tested and details covering the generation and mining of surrogate models.

4.1 | Selection methods

The selection methods used by GAs and other EAs determine which subset of solutions from a population are used to generate the next population. It is possible that the selection methods used by the algorithms could be attributed to some level of variable dependency detected in the

TABLE 1 Selection operators.

Selection type	Pool size	Relative selection pressure
Truncation 20%	Top 20% by fitness	High
Truncation 50%	Top 50% by fitness	Low
Tournament	Randomly selected 5	High

results. To avoid bias in the results due to a single selection method being used, it was decided to gather results using three selection profiles as seen in Table 1. This was done to use methods of differing selection pressure to determine the potential impact of the selection operator on the results.

The selection methods used tend to favour solutions with a higher fitness than those with a lower fitness value. Because of this it is possible that certain variable dependencies would only be captured due to the selection method behaviour. To address this, the selection pressure placed on high fitness solutions was varied by applying both a top 50% and 20% selection criteria in the truncation selection method as well as a tournament size of 5 in the tournament selection method. This provided a set of results containing a variety of selection pressure values to be used in determining the specific impact of the selection operator in the experiments.

4.2 | Algorithm runs

Two population-based solvers were selected to generate a series of population trajectories for use in this study. These were a GA and a modified population based incremental learning (PBIL) algorithm (Baluja, 2002; Baluja & Caruana, 1995) in which a negative mutation rate and mutation shift value is introduced. These algorithms were selected for the purpose of comparing the results of a univariate solver and a more traditional GA on problems with different structure. Each algorithm was run on the set of outlined problems in order to generate the trajectories used in the analysis phase of this trial.

A length of 40 bits was selected for all problems being solved. This was done due to the relative simplicity of the optimisation problems being used in this article. A length of 40 bits allowed for sufficiently long optimisation runs to generated enough populations of solutions for our analysis. A population size of 100 was selected to provide a large enough selection pool for all selection methods trialled. Each optimisation run would then run for a total of 100 generations. With the aim of mining features from a large number of search trajectories a total of 100 optimisation runs for each algorithm, problem and selection combination was used. The aim of this article is to mine features capable of relating variable values and fitness from the search trajectories. Due to this, no further attempts are made to tune the run-time parameters of the algorithms beyond some preliminary runs as the algorithm specifics are not the focus of this article.

4.2.1 | Genetic algorithm

The GA used was an adaptation of the canonical genetic algorithm (Holland, 1992). Figure 2 shows the main steps involved as the GA generates new populations during an optimisation run.

Table 2 outlines the values used in the running of the GA during these experiments. The GA-specific run setting of mutation rate was set to 0.005 after some preliminary tuning runs.

P shows the number of solutions in each population. **maxGen** is the maximum number of generations before termination. **mutRate** is the mutation rate within the GA. **Selection** is the selection method used within the GA for solution comparison and reproduction. **Crossover** is the crossover type used in this trial with a rate of 1 and so occurs each generation. **N** is the problem length. **runs** is the number of runs for each problem the GA ran for.

4.2.2 | Population based incremental learning

PBIL is a form of EDA. The probability vector is updated and mutated each generation as seen in Equations (4) and (5):

$$p(X_1,\ldots,X_N) = \prod_{i=1}^{N} p(X_i), \tag{4}$$

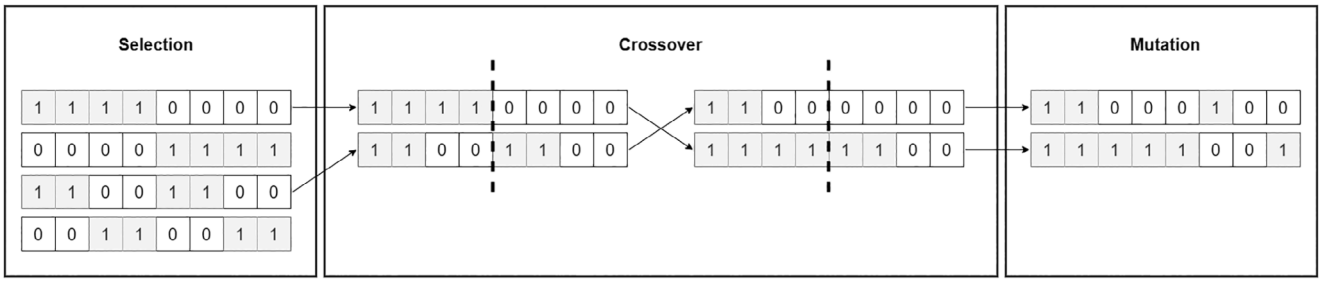

FIGURE 2 Genetic algorithm operators.

TABLE 2 GA run specifications.

P	N	runs	maxGen	mutRate	Selection	Crossover
100	40	100	100	0.005	Tournament and truncation	Random 1-Point

TABLE 3 PBIL run specifications.

P	N	runs	maxGen	mutRate	mutShift	learnRate	Selection
100	40	100	100	0.005	0.05	0.1	Tournament and truncation

$$p(X_i) = \frac{1}{N}\sum_{j=1}^{N} x_{ij}. \tag{5}$$

Here the vector of marginal probabilities $P_V = (p(X_1),\ldots,p_(X_n))$ is created by calculating the arithmetic mean of each variable X in a population of size *N*. As the solutions are comprised of bit strings we will see that values will range from 0 to 1.

Table 3 outlines the values used for the PBIL algorithm, seen in Algorithm 1, in this trial. Here, *N* is the population size, additional to these, the PBIL used a **mutShift** value that was applied to mutated probability vector values. **learnRate** is the learning rate of the algorithm.

The algorithm-specific run settings of mutation rate, mutation shift, learning rate and negative learning rate were selected based on the results of Baluja (1994).

4.3 | Benchmark problems

4.3.1 | The 1D checkerboard

The 1D checkerboard function scores the chromosome based on the sum of adjacent variables that do not share the same value Baluja & Davies, 1997. The function is seen here in Equation (6).

$$\mathrm{CHECK}^{l}_{1D}(x) = \sum_{i=0}^{l-2} \begin{Bmatrix} 1, & x_i \neq x_{i+1} \\ 0, & x_i = x_{i+1} \end{Bmatrix}. \tag{6}$$

Because the function scores only adjacent variables it is possible to have two possible global maxima. As an example, for a bit string of length 5 the two possible would be [01010] and [10101]. The implementation of the problem used in this experiment also checks the first and last allele to check if they match. This allows for a total fitness value equal to the bit-string length for an ideal solution.

4.3.2 | The royal road

The royal road function scores chromosomes based on collections of variable values based on a specified set of schema that the solution must fulfil in order to score an optimal value (Forrest & Mitchell, 1993). below, Equation (7) specifies the fitness function for the royal road problem with a schema block size of 5, as used in this experiment.

Algorithm 1 **Population-based incremental learning (PBIL).**

Initialize a probability vector $p(X_1, \ldots, X_N)$ with 0.5 for each position
for $t = 1$ to maxGen **do**
  Generate population X of P binary strings based on $p(X_1, \ldots, X_N)$
  Evaluate each individual in X and find the best individual X^*
  Update $p(X_1, \ldots, X_N)$:
  for $i = 1$ to N **do**
    $p(X_i) = p(X_i) * (1 - \text{learnRate}) + X^*[i] * \text{learnRate}$
  end for
  Introduce variation to $p(X_1, \ldots, X_N)$:
  for $i = 1$ to N **do**
    if Random() < mutRate **then**
      $p(X_i) = p(X_i) * (1 - \text{mutShift}) + \textit{RandomBinary}() * \text{mutShift}$
    end if
  end for
end for

$$R_1(x) = \sum_{i=1}^{5} \delta_i(x) o(s_i), \text{ where } \delta_i(x) = \begin{cases} 1 & \text{if } x \in s_i \\ 0 & \text{otherwise} \end{cases}. \tag{7}$$

As noted by Forrest and Mitchell (1993) the equation represents the fitness function, such that R_1 is a sum of terms relating to partially specified schema. The schemata are subsets of solutions that match the partial specification, s_i. As an example, one partially specified schema with a size of 5 could be represented as [11111***** ...] where unspecified members are denoted by '*'

A given bit-string x is an instance of a specific schema s,$x \in s$ if x matches s in the defined positions within that schema. $o(s_i)$ defines the order of s_i which is the the number of defined bits in s_i. The royal road function was designed to 'capture one landscape feature of particular relevance to GAs: the presence of fit low-order building blocks that recombine to produce fitter, higher-order building blocks' (Mitchell et al., 1991).

4.3.3 | The Trap-5

The Trap-5 concatenated problem is designed to be intentionally deceptive (Goldberg, 1989a; 1989b), such that they 'deceive evolutionary algorithms into converging on a local optimum. This is particularly a problem for algorithms which do not consider interactions between variables' (Brownlee, 2009). As with the royal road problem, the bit-strings are partitioned into blocks and their fitness scored separately. Seen in Equation (8a) is the function of a trap of order k.

$$f(x) = \sum_{i=1}^{n/k} \text{trap}_k(x_{b_{i+1}} + \cdots + x_{b_{i+k}}), \tag{8a}$$

$$\text{trap}_k(u) = \left\{ f_{\text{high}} \text{ if } u = k, f_{\text{low}} - u\frac{f_{\text{low}}}{k-1} \text{ otherwise} \right\}. \tag{8b}$$

Blocks within the bit-string are scored according to the fitness function in Equation (8b). A Trap5 problem with a bit-string length of 10 would have the values n=10, k=5, $f_{\text{high}} = 5$ and $f_{\text{low}} = 4$. The further from the goal of each trap containing five 1's, the higher the fitness value, with only a maximum achieved when the whole trap is comprised of 1s, leading the algorithm away from the optimal value.

4.4 | Mining surrogate models

Our experiments also compare with another approach to explaining solution quality: mining surrogate models as proposed by Wallace et al. (2021). We briefly recap the approach here. Surrogate models (Brownlee et al., 2015; Jin, 2011) are usually used to speed up an NDA by replicating a costly true function and replacing calls to it with calls to a much cheaper model. Wallace et al. (2021) noted that the surrogate represents an explicit model of the population, and proposed mining this model to capture the sensitivity of the objective function to the problem variables. The idea is that the model is biased by the population of the NDA, and so represents another view of the algorithm's understanding of the problem.

For the experiments presented here, the final population of each algorithm's run was used as training data for a support vector regression (SVR) model, implemented in scikit-learn (version 1.0.2). We used the default parameter settings for the SVR model, the model input features are simply the problem variables X and the target for prediction is fitness $f(X)$. The purpose is to replicate the surrogate model that would be obtained at the end of an algorithm's run. The best (highest fitness) solution, which we refer to as the seed, is then taken from the population and evaluated against this model. The Hamming-1 neighbourhood of this solution is also evaluated against the surrogate, allowing us to determine the change in surrogate fitness due to mutating each variable in the solution. In a slight modification of the procedure in Wallace et al. (2021), we also set the sign of the 'importance' measure to be negative if the corresponding bit in the starting seed solution was 1: this provides an indication of the direction of the relationship between that variable and fitness. This gives us a measure of both each variable's importance, and its correlation with fitness, in the vicinity of the high-fitness solution. More formally, this probing process is as shown in Algorithm 2.

In the results section, we report the mean change in surrogate fitness for each variable, aggregated over the 100 repeated runs of each algorithm. Note that a separate surrogate model was constructed and probed for each of the repeated runs.

5 | RESULTS

We hypothesize that is it possible to derive features from algorithm trajectories that can aid in generating explanations for end-users similar in nature to existing known metrics such as the Kullback–Leibler entropic divergence values. For two population-based NDAs—a GA and a univariate population-based incremental learner—we generated a total of 100 algorithm trajectories on each of the three test function, selection operator pairs used. These trajectories were transformed using PCA to allow the projection of the populations into a lower dimension space for the purpose of visualisation and feature extraction.

5.1 | Global information gain and angle from origin

The results of the comparison of global information and angle from origin values calculated for the Trap 5 function are shown in Figure 3 for the GA and Figure 4 for the PBIL. These are split by selection operator and have been scaled between 0.0 and 1.0 to allow for a direct comparison of the

Algorithm 2 **Method for probing variables in a solution with respect to the surrogate fitness function**

In: $x = (x_0 \dots x_n), x_i = \{0, 1\}$, near-optimal *seed* solution found by GA
In: $S(X) \rightarrow f$, surrogate fitness function to estimate fitness f of a solution X
Out: $C = (c_0 \dots c_n), c_i \in \mathbb{R}$, absolute change to surrogate fitness for each variable in x
$C \leftarrow \emptyset$;
$f_{org} \leftarrow S(x)$ ▷ surrogate fitness of solution
for each variable x_i **do** $i = 0$ to $n - 1$
  $x_i \leftarrow (x_i + 1) \bmod 2$ ▷ flip variable x_i
  $\hat{f}_i \leftarrow S(x_i)$ ▷ surrogate fitness of mutated solution
  if $x_i = 1$ **then** $c_i = -1 * abs(f_{org} - \hat{f}_i)$ ▷ Change in surrogate fitness, optimum should have a 1
  else $c_i = abs(f_{org} - \hat{f}_i)$ ▷ Change in surrogate fitness, optimum should have a 0
  end if
  $C \leftarrow C \cup c_i$ ▷ add to list
end for

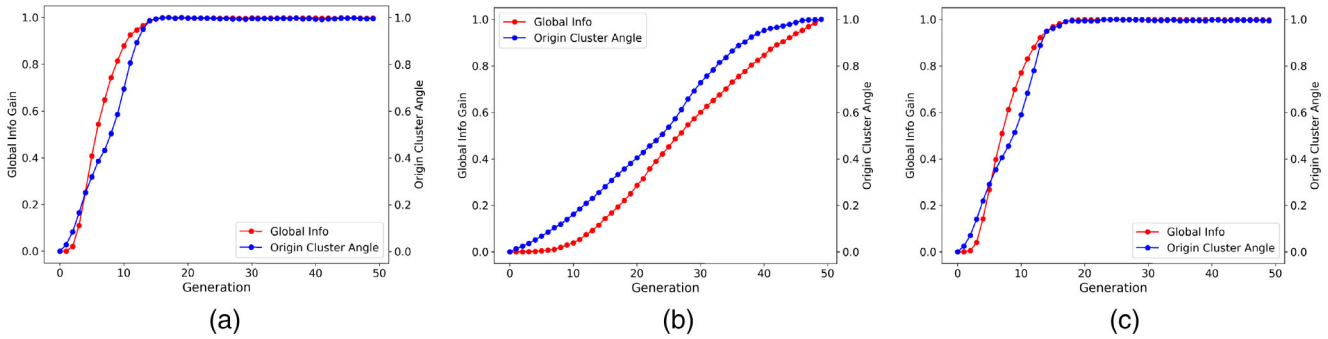

FIGURE 3 GA global information versus PCA angles on Trap5 by selection. (a) GA Trap5 Trunc20 mean global (b) GA Trap5 Trunc50 mean global (c) GA Trap5 tour mean global.

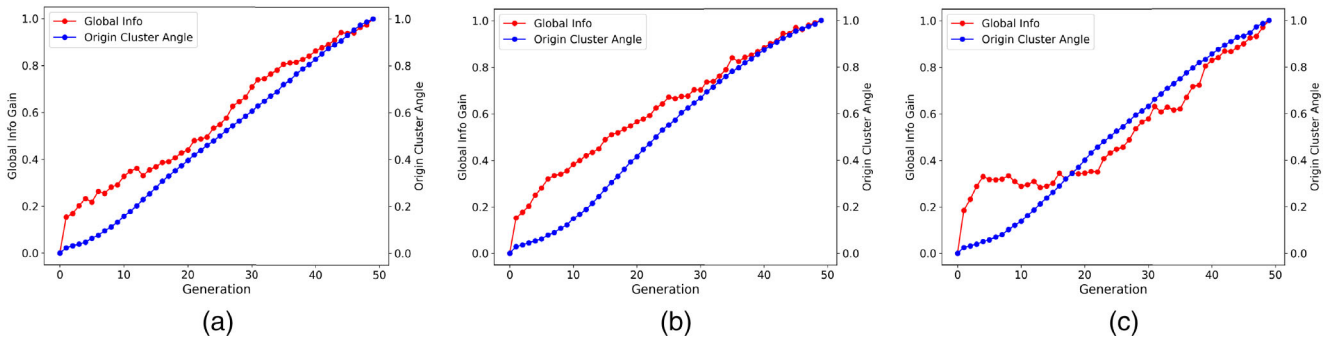

FIGURE 4 PBIL global information versus PCA angles on Trap5 by selection. (a) PBIL Trap5 Trunc20 mean global (b) PBIL Trap5 Trunc50 mean global (c) PBIL Trap5 tour mean global.

behaviour rather than the values themselves. These figures show that angle from origin values in both algorithms across selection operators closely follow that of the global information gain. Both metrics detect the increase in information gained as the algorithms solve the supplied problem. This behaviour is closely matched in the results for both the 1D checker and royal road problems, as seen in Appendix A.1 (Figures A1 and A2) and Appendix A.2 (Figures A3 and A4) for both algorithms. This is reflected in the results shown in Table 4 in which all have strongly correlated values.

5.2 | Local information gain and inter-cluster angle

The local information gain and inter-cluster angle comparison results are more varied than those of the global information and appear to be showing that learning behaviour differs between algorithms on the same problem and selection operator. Shown in Figures 5 and 6 are the results for the GA and PBIL on the Trap 5 problem respectively. The GA results show a distinct lag of approximately 10 generations before detecting the maximum angle when compared to the local information gain in the Truncation-20 and tournament results. The Truncation-50 results show that the information gain and inter-cluster angle values take considerably longer to peak before trending towards 0, showing a significantly different behaviour to that of the other two selection operators. These results are reflected in the wider range of correlation coefficients calculated for the Truncation-50 selection seen in Table 4.

The inter-cluster angle results calculated for the populations generated by the PBIL do not share the same pattern of behaviour as the local information gain. The results show a peak approximately 25 to 30 generations later than the local information gain and so these events do not co-occur at the same point in the trajectory in all problems tested. This difference in behaviour is reflected in the wider range of correlation coefficients calculated and shown in Table 4. This is seen across all selection operator results and is highly similar across all problems. The results for the 1D checker and royal road for both GA and PBIL are shown in Appendix A.3 (Figures A5 and A6) and Appendix A.4 (Figures A7 and A8).

5.3 | Correlation

Table 4 displays the Spearman correlation coefficients of local and global information gain to the inter-cluster and angle from origin features extracted. Global information shows a strong positive correlation to the angle from origin feature with a range of 0.88 to 0.99 in the GA results

TABLE 4 Spearman correlation coefficient.

		PBIL		GA	
Problem	**Selection**	**Local to inter-cluster**	**Global to origin**	**Local to inter-cluster**	**Global to origin**
Trap5	Truncation 20%	**−0.73**	**1.0**	**−0.69**	**0.99**
	Truncation 50%	**−0.78**	**1.0**	0.83	**0.98**
	Tournament 5	**−0.72**	**0.97**	**0.36**	**0.96**
1D checkerboard	Truncation 20%	**−0.86**	**1.0**	**0.94**	**0.88**
	Truncation 50%	**0.39**	**0.88**	**0.39**	**0.88**
	Tournament 5	**−0.75**	**1.0**	**0.5**	**0.99**
Royal road	Truncation 20%	**−0.83**	**1.0**	**0.77**	**0.98**
	Truncation 50%	**−0.79**	**1.0**	0.23	**0.99**
	Tournament 5	**−0.81**	**1.0**	**0.82**	**0.97**

Note: Bold values indicate where the *p*-value associated with the correlation was < 0.05.

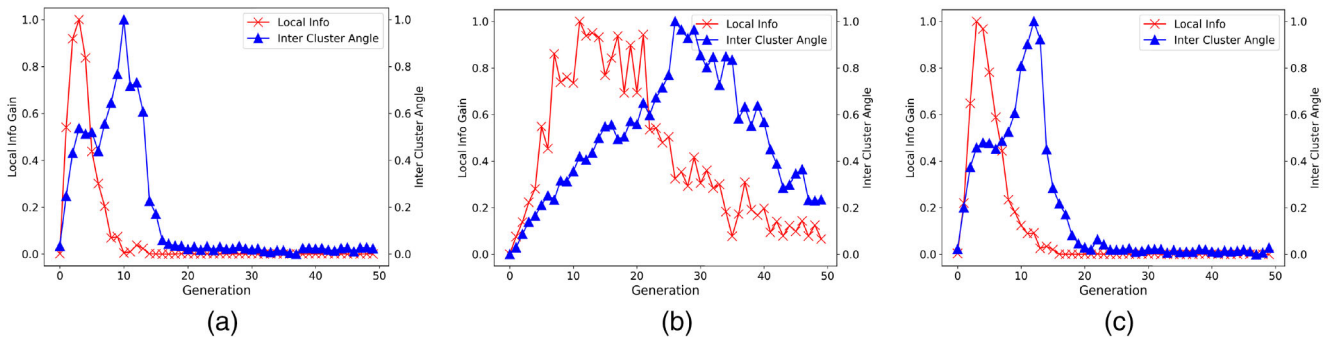

FIGURE 5 GA local information versus PCA angles on Trap5 by selection. (a) GA Trap5 Trunc20 mean local (b) GA Trap5 Trunc50 mean local (c) GA Trap5 tour mean local.

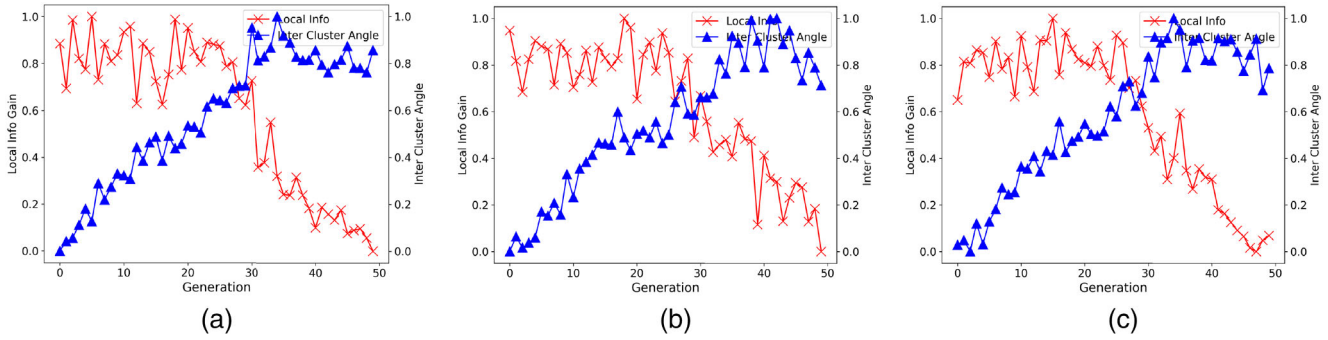

FIGURE 6 PBIL local information versus PCA angles on Trap5 by selection. (a) PBIL Trap5 Trunc20 mean local (b) PBIL Trap5 Trunc50 mean local (c) PBIL Trap5 tour mean local.

and 0.88 to 1.0 in the PBIL across all problems and selection operators. Local information correlation values show a higher degree of variation in both positive and negative directions. The PBIL results span −0.86 to 0.39 in the 1D checkerboard values, giving a range of 1.25. The GA results for the Trap5 problem show both positive and negative correction with values of −0.69 to 0.83, a range of 1.52.

The results show a clear difference between the two algorithms when local information gain is compared to the inter-cluster-angle results. This may be due to the fact that the PBIL increments the probabilistic model gradually over successive populations so local information gain accumulates before it is reflected in inter-cluster angle change. As a GA can be considered a Markov process, the probability of each population is only dependant to the current state of the system. This can also be seen when global information gain is compared to angle from origin. The PBIL reaches maximum global information later in the trajectory than the GA with a shallower, almost linear ascent in these trials. The PBIL reaches a

point at which global information gain stops increasing late in the optimisation run whereas the GA has a steeper global information gain rate, reaching the maximum value between generations 10 and 20. This may be due to the GA taking a more varied path across the search space than the PBIL which tends to have less varied performance. Together, these show that it is possible to detect differences in algorithm behaviour over the same optimisation problems through the differences in both sets of results.

5.4 | PCA explained variation

The values in Table 5 show the mean percentage of variance in the original data explained by the first three principal components, broken down by algorithm and problem.

The results show that for the PBIL, total variation explained by the first three principal components ranged 22% for Truncation-50 and 29% for Truncation-20. Similar values are found in all three problems in the PBIL results with Truncation 50 showing the lowest value and Truncation 20 showing the highest, closely followed by Tournament 5.

The explained variation results for the GA show a similar pattern to that of the PBIL with the gap between Truncation-20 and tournament values smaller, with the tournament having a marginally higher value of 41% to Truncation-20's 40% in the royal road problem results.

5.5 | Principal component loadings

The results of plotting the mean loadings calculated for each algorithm and problem pair can be seen in Figures 7–12. The Figures 7–9 contain the plotted results of the GA on each of the three test problems and Figures 10–12 show those of the PBIL.

The results for the Trap5 problem are shown in Figures 7 and 10 for the GA and PBIL respectively. Across all three selection methods, the GA results show all 8 blocks of 5 consecutive bits possess similar values and are distinct from the next block. This reflects the expected fitness function structure of the Trap5 problem in which each block of 5 bits requires the same value of 1 to achieve the highest fitness of 5. However, if that block is comprised of four 0's and one 1, then a score of 4 is achieved. The results for the Trap5 problem for the PBIL however, do not show any strong relation to the expected fitness function structure across the selection methods. Since PBIL is univariate, it cannot detect multivariate interactions between the bits in each block, preventing this structure from being detected.

TABLE 5 PCA variance explained by first three components.

			Explained variance %			
Algorithm	**Problem**	**Selection**	**PC1**	**PC2**	**PC3**	**Total**
PBIL	Trap5	Truncation 20%	20	5	3	29
		Truncation 50%	13	5	4	22
		Tournament 5	18	5	3	26
	Checker	Truncation 20%	19	5	3	27
		Truncation 50%	13	5	4	22
		Tournament 5	17	5	3	26
	Royal road	Truncation 20%	18	5	3	27
		Truncation 50%	13	5	4	22
		Tournament 5	16	5	3	25
GA	Trap5	Truncation 20%	33	8	6	47
		Truncation 50%	27	8	5	40
		Tournament 5	33	8	5	47
	Checker	Truncation 20%	29	8	6	43
		Truncation 50%	24	7	6	37
		Tournament 5	28	8	6	42
	Royal road	Truncation 20%	27	7	5	40
		Truncation 50%	25	7	5	36
		Tournament 5	28	8	5	41

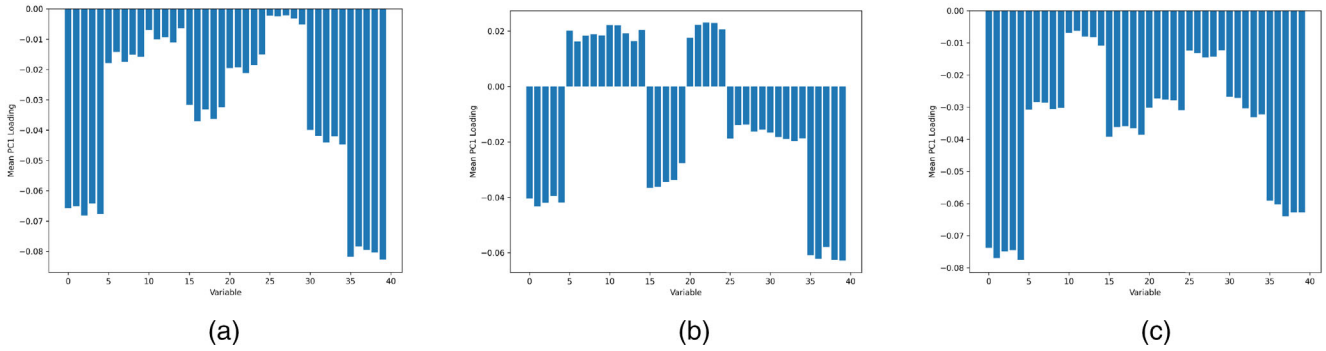

FIGURE 7 GA mean loading values information versus PCA angles on Trap5 by selection. (a) GA Trap5 Trunc20 mean loading (b) GA Trap5 Trunc50 mean loading (c) GA Trap5 tour mean loading.

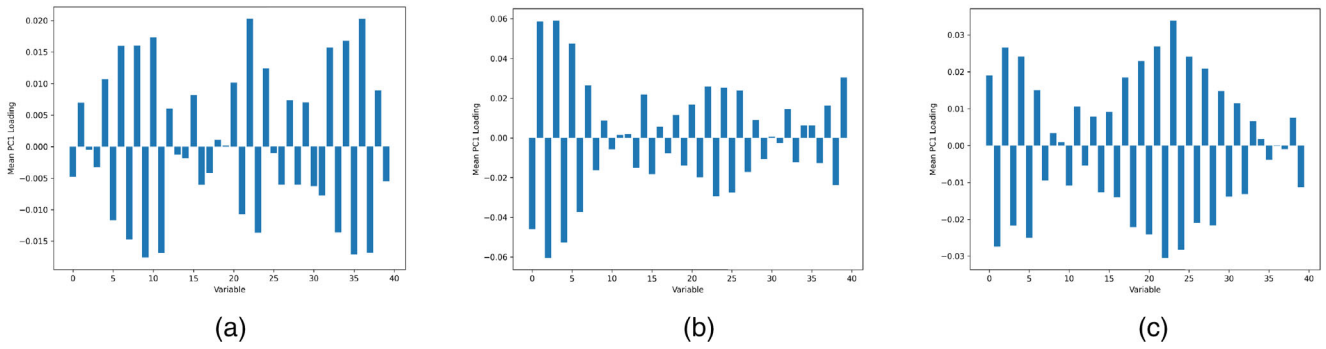

FIGURE 8 GA mean loading values information versus PCA angles on 1D checkerboard by selection. (a) GA 1DChecker Trunc20 mean loading (b) GA 1DChecker Trunc50 mean loading (c) GA 1DChecker tour mean loading.

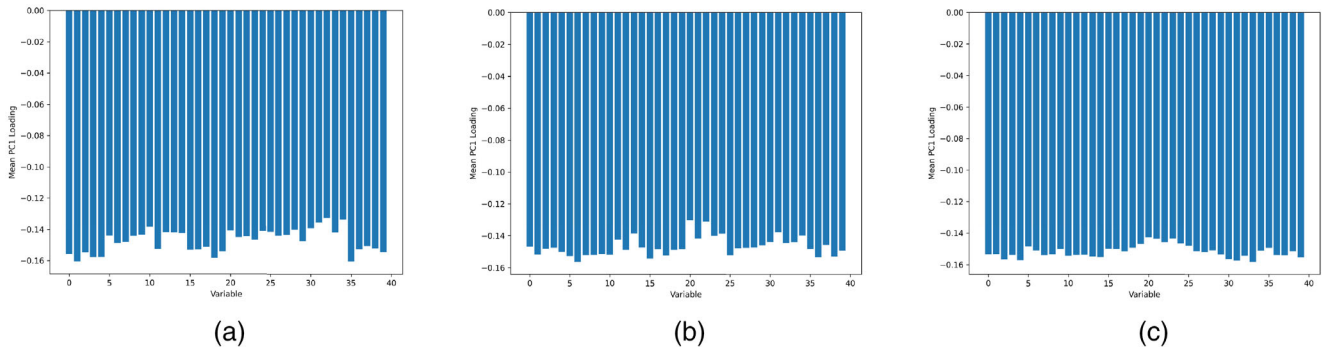

FIGURE 9 GA mean loading values information versus PCA angles on royal road by selection. (a) GA royal road Trunc20 mean loading (b) GA royal road Trunc50 mean loading (c) GA royal road tour mean loading.

The 1D checkerboard results, shown in Figures 8 and 11, show that the recorded loadings reflect the patterns of the PCA coefficients. In those plots, we can see that adjacent variables in the solutions discovered by the GA have opposing values. This pattern is also observed in the results of the PBIL on the same 1D checkerboard problem seen in Figure 11. It is important to note that the 1D checkerboard problem has two global optimal solutions. The loadings recorded match closely the mathematical structure of the fitness functions such that adjacent values in the bitstring should take opposing values. This is due to both global optimal solutions being binary complements of each other. Both algorithms however show instances in which the loadings did not conform to the expected pattern, showing a flip in the alternating sequence at three or more points in the bit-string. The Truncation-50 results for the PBIL (Figure 11b) show the highest number of non-conformities to this expected structure. The 1D checkerboard results show that some bivariate interaction was detected but this will be accidental.

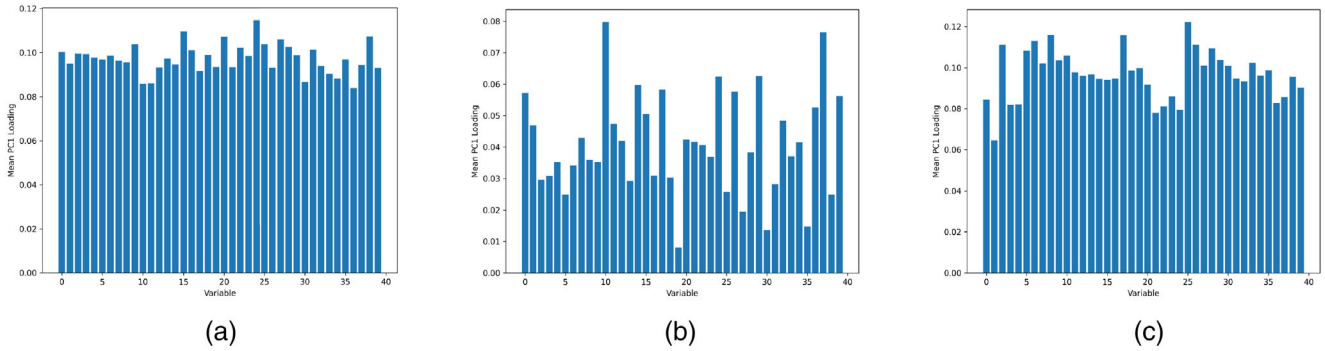

FIGURE 10 PBIL mean loading values information versus PCA angles on Trap5 by selection. (a) PBIL Trap5 Trunc20 mean loading (b) PBIL Trap5 Trunc50 mean loading (c) PBIL Trap5 tour mean loading.

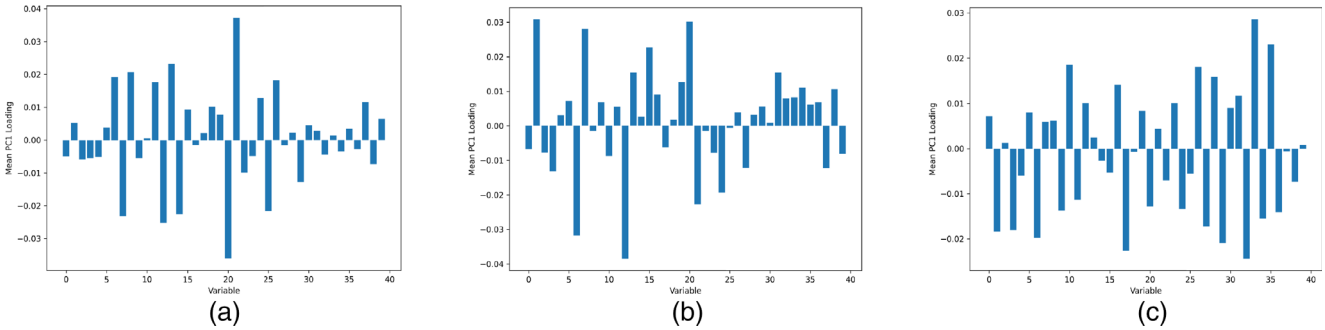

FIGURE 11 PBIL mean loading values information versus PCA angles on 1D checkerboard by selection. (a) PBIL 1DChecker Trunc20 mean loading (b) PBIL 1DChecker Trunc50 mean loading (c) PBIL 1DChecker tour mean loading.

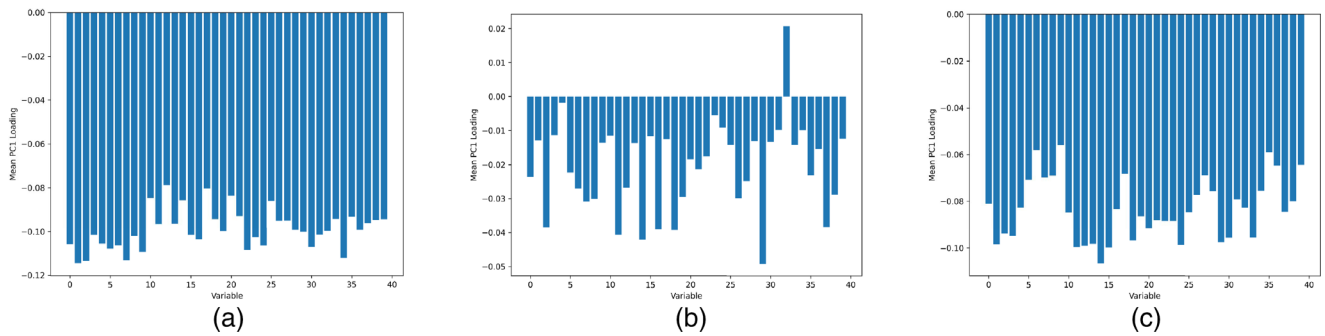

FIGURE 12 PBIL mean loading values information versus PCA angles on royal road by selection. (a) PBIL 1DChecker Trunc20 mean loading (b) PBIL 1DChecker Trunc50 mean loading (c) PBIL 1DChecker tour mean loading.

The results of the royal road problem are shown in Figure 9 for the GA and 12 for the PBIL. In this problem, like the Trap5 problem, each block of 5 bits requires the same value of 1 to achieve the best fitness however a fitness of 0 is assigned to any other combination of values in each block. The GA results show some partial problem structure detection, with consecutive blocks of 5 bits having similar values that do not match the next blocks. There are, however, only a few instances of this occurring. The PBIL results do not show any clear pattern that would match the expected structure defined by the fitness function. These results show that the algorithm trajectories reflect the simpler features of the problem structure that the algorithms have learned but the higher-order features are less likely to be recovered.

5.6 | Surrogate mining

We now consider the results of mining surrogates fitted to the final population of each of the algorithm runs. Each plot in Figures 13–18 shows the mean contribution to surrogate fitness for each variable, over the 100 surrogates constructed from the final population of each repeated run

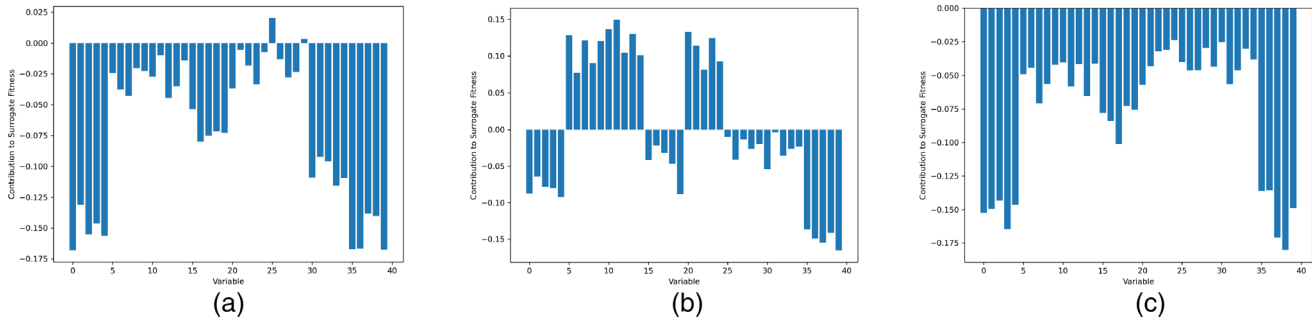

FIGURE 13 GA contribution to surrogate fitness per variable on Trap5 by selection. (a) GA Trap5 Trunc20 contribution to surrogate fitness per variable (b) GA Trap5 Trunc50 contribution to surrogate fitness per variable (c) GA Trap5 tour contribution to surrogate fitness per variable.

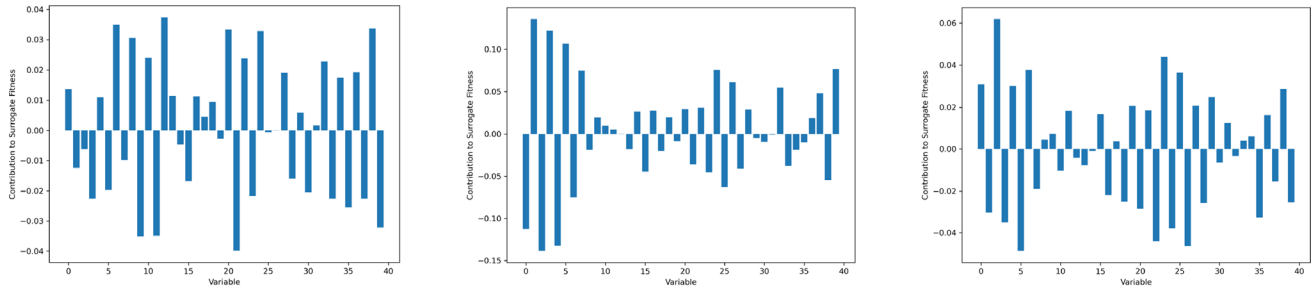

FIGURE 14 GA contribution to surrogate fitness per variable on 1D checkerboard by selection. (a) GA 1DChecker Trunc20 contribution to surrogate fitness per variable (b) GA 1DChecker Trunc50 contribution to surrogate fitness per variable (c) GA 1DChecker tour contribution to surrogate fitness per variable.

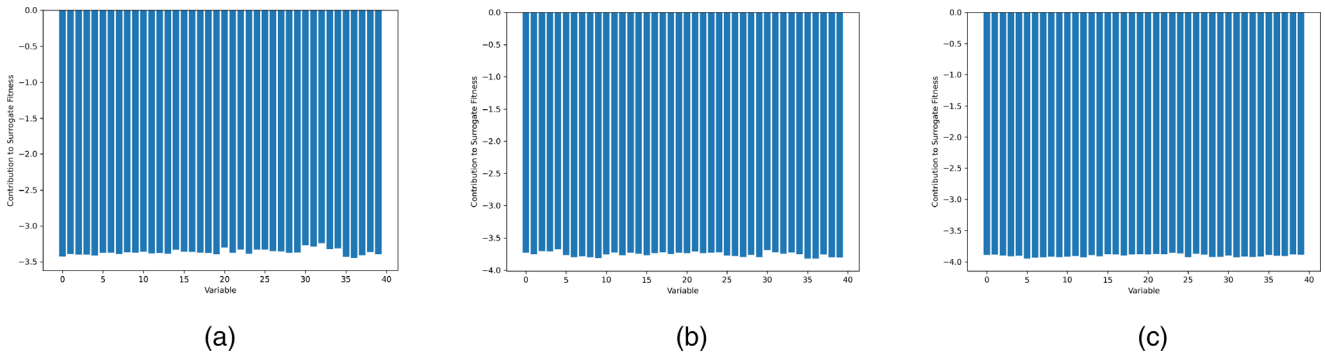

FIGURE 15 GA contribution to surrogate fitness per variable on royal road by selection. (a) GA 1DChecker Trunc20 contribution to surrogate fitness per variable (b) GA 1DChecker Trunc50 contribution to surrogate fitness per variable (c) GA 1DChecker tour contribution to surrogate fitness per variable.

of the algorithm. Note that positive values indicate that the optimal value was a 0 and negative values indicate that it was a 1 (we have chosen this way round to match the results more easily with the PCA loadings). We observe that the surrogate mining approach has also picked up key characteristics of the problem; but with some differences. For Trap5, surrogate mining (Figures 13 and 16 and Figures 7 and 10) has found all variables to be equal in importance, with the ideal value for each being a 1. The groupings are not visible because, although the surrogate is trained on the whole population, the mining procedure probes variables in isolation. In contrast, the PCA loadings reflect that some sub-optimal members of the final population have groups set to the locally-optimal 0. For royal road (Figures 15 and 18), the relative importance of variables in each problem is approximately the same, as was the case with the PCA loadings. Here, the populations have largely converged to set of solutions in which the variables have the value of 1. This is reflected in the loadings. For 1D checkerboard, both approaches show the chain pattern of alternating bits. Different weights are associated with each variable (i.e., some appear to contribute more than others to fitness), though the overall

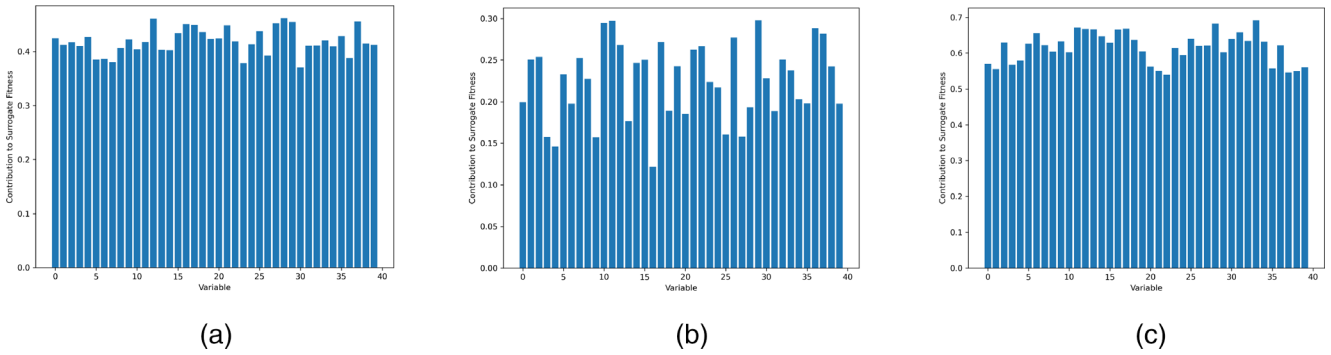

FIGURE 16 PBIL contribution to surrogate fitness per variable on Trap5 by selection. (a) PBIL Trap5 Trunc20 contribution to surrogate fitness per variable (b) PBIL Trap5 Trunc50 contribution to surrogate fitness per variable (c) PBIL Trap5 tour contribution to surrogate fitness per variable.

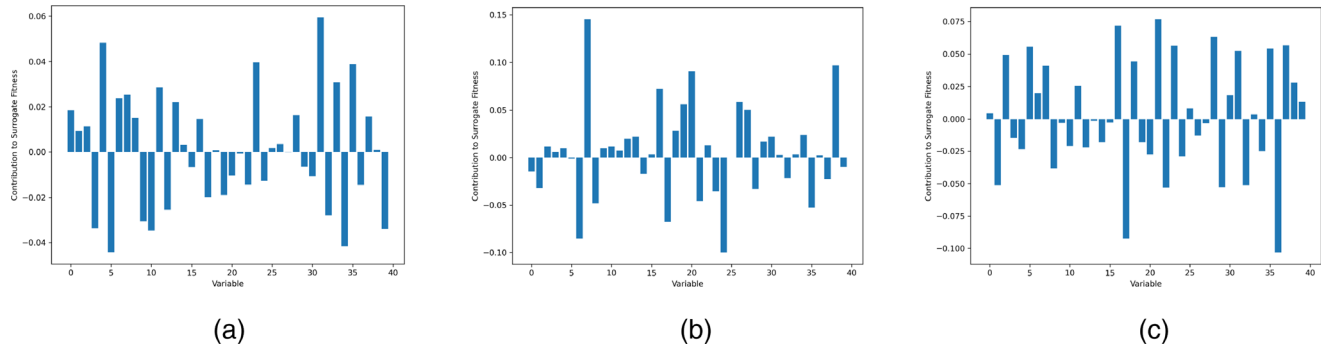

FIGURE 17 PBIL contribution to surrogate fitness per variable on 1D checkerboard by selection. (a) PBIL 1DChecker Trunc20 contribution to surrogate fitness per variable (b) PBIL 1DChecker Trunc50 contribution to surrogate fitness per variable (c) PBIL 1DChecker tour contribution to surrogate fitness per variable.

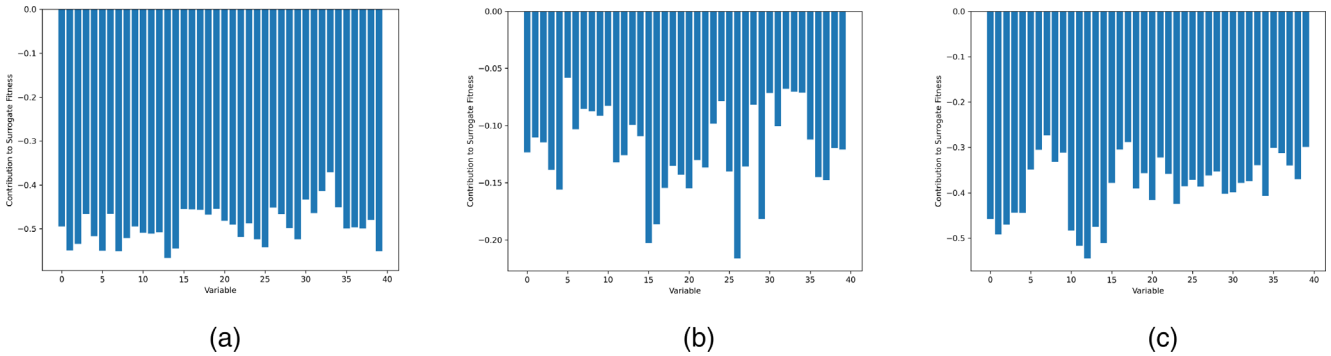

FIGURE 18 PBIL contribution to surrogate fitness per variable on royal road by selection. (a) PBIL royal road Trunc20 mean loading (b) PBIL royal road Trunc50 mean loading (c) PBIL royal road tour mean loading.

pattern of these weights is similar between the PCA loadings and the surrogate mining. The 1D checkerboard problem does not actually have the property of differing variable contributions to fitness as all bits contribute equally to the fitness function. Once the algorithm has begun to assemble chains of alternating sequences, however, their contribution may begin to vary. Bits within those sequences will contribute more as mutating them will drop two points of fitness (a point for the variable either side of the one being mutated). As such, the explanatory methods have revealed that the population has begun to detect the key component of the problem. The less clear patterns for PBIL on 1D checkerboard again reveal that algorithm's inability to capture the variable interactions that are needed to solved the problem efficiently.

A visual comparison of the results from the PCA and surrogate mining approaches suggests that both identify similar characteristics of the optimal solutions. In order to determine more precisely how similar the results are we now also consider a quantitative comparison of the

TABLE 6 Spearman rank correlation between PCA loadings and variable importance mined from the surrogates.

Alg and selection	Trap5	1D checkerboard	Royal road
GA truncation 20%	**0.891**	**0.800**	**0.760**
GA truncation 50%	**0.904**	−0.112	**0.522**
GA tournament	**0.708**	**0.905**	**0.427**
PBIL truncation 20%	0.085	0.150	0.315
PBIL truncation 50%	0.308	**0.398**	**0.571**
PBIL tournament	**0.713**	**0.537**	**0.513**

Note: Bold values indicate where the *p*-value associated with the correlation was < 0.05.

'importance' values for each variable derived from the PCA loadings and the surrogate mining approach. For each problem and algorithm, we computed the Spearman rank correlation between the PCA loading values and the contributions to surrogate fitness for all variables. The correlation results can be seen in Table 6. Most of the correlations are strong and positive, indicating that the overall trends in variable importance identified by both approaches are similar. That is, explanations generated by the PCA approach and those generated by the surrogate mining approach are similar enough to suggest that they are coming from the problem rather than artefacts of the explanation approaches themselves. This adds some confidence that both approaches are identifying something fundamental about the problem. We suspect that the lower values for PBIL are due to the more heavily converged population when using a high selective pressure, causing the surrogate to fit less well.

The two approaches give a slightly different perspective: the patterns seen for the surrogate mining approach correspond more closely to our intuitive understanding of the problems (especially for Trap5), but the patterns seen in the PCA loadings reflect the variation in the population itself. There is a broader philosophical question around whether the explanations for the optimisation problem or the particular solver are more useful: we suggest that both have an important role to play in bringing insight into the optimisation results. That the approaches are largely in agreement about the contribution of each variable to fitness provides some measure of confidence that both approaches are providing some insight into the algorithm's choice of solution.

6 | CONCLUSIONS

In this article, we presented the results of the application of PCA to the trajectories created by two population-based search metaheuristics. This was done to mine features that can enrich explanations regarding how these algorithms traverse the search space and present significant solution features detected by the algorithms. We generated a collection of algorithm trajectories by solving a set of benchmark problems with a GA and a modified PBIL algorithm and projected the resulting trajectories into a lower-dimensional space through the application of PCA. This process resulted in a dataset that was mined using a novel set of angular-based metrics. The final generations of these trajectories were also used to create a surrogate model with the aim of mining features related to the sensitivity of the objective function to the problem variables.

Our evaluation of these metrics when compared to the Kullback–Leibler entropic divergence measure of both local information and global information gain shows that there is potential to capture a similar level of detail regarding the global information learned. These metrics were used to detect differing algorithm behaviour on the same problems as seen between the PBIL and GA in the inter-cluster angle values. In this evaluation, a set of selection methods was introduced to monitor the effect of differing levels of selection pressure on the derived and established metrics. The results show that the GA is considerably more sensitive to selection pressure when generating populations with higher fitness solutions regarding global information gain. The PBIL results have shown that selection pressure plays a lesser role in higher fitness solution generation than the GA in both local and global metrics, however, its overall performance was less due to its univariate nature. The highly correlated findings relating to information gain show that the features being detected do show a relation between the variable value choices being made internally by the metaheuristic and the fitness of a candidate solution. It was shown that principal component loadings can be used to represent what the algorithms have learned in terms of variable contributions to overall fitness. This can be seen in the eigenvector values for the GA that implied the fitness function structure of the optimisation problem for the 1D checkerboard and Trap5 problem. This feature in the PBIL results shows partial structure detection only in the 1D checkerboard problem and shows that some structure has not been captured using the features used in these tests. Being univariate, PBIL is incapable of creating probability features that capture higher-level features with interactions as found in the remaining problems. Finally, we fitted surrogate models to the final populations of each algorithm run and mined these models using a probing procedure to extract a measure of each variable's contribution to fitness. The surrogate modelling results show a broad level of alignment with those of the PCA analysis with some exceptions. In the results for the Trap5 problem, the surrogate results did not detect the variable groups that were found in the PCA approach.

The results of this article have shown that the PC-derived features are associated with the algorithm learnings regarding problem structure. As a similar set of results were extracted from the surrogate model, this adds a further degree of confidence in these findings for both approaches. These techniques can be considered a stepping stone towards supporting explanations by relating changes in information gain and variable importance ranking to the discovery of specific interaction features.

Knowing the sensitivity of the objectives to individual variables is one form of explanation for solution quality. In a real-world application, such information could enable a decision-maker to make informed refinements to the generated solutions to accommodate goals not included in the original definition of fitness. (For example, aesthetics in a design problem, or personal preferences in a rostering problem.) Furthermore, if the identified sensitivities are counter-intuitive, then it may indicate problems in the definition of the problem as represented by the fitness function. If the importance is as expected, it provides confidence that the problem solved by the algorithm accurately reflects reality.

6.1 | Future work

Trajectories record the important series of decisions made by a metaheuristic, based on the internal operators at work. Further work is planned regarding the construction of explanations based on these points of change and detectable, fitness-related features at these points. Future work can be summarized in three main objectives:

1. The continued detection of fitness-related features surrounding these points of change as seen in the local and global information behaviours.
2. To translate features derived from the search to narration to explain the decision processes that explain the final set of solutions arrived at by the metaheuristic.
3. To extend the experimental set to include additional metaheuristics and analysis techniques including SA.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Martin Fyvie https://orcid.org/0000-0001-8491-7008
John A. W. McCall https://orcid.org/0000-0003-1738-7056
Lee A. Christie https://orcid.org/0000-0001-8878-0344
Alexander E. I. Brownlee https://orcid.org/0000-0003-2892-5059
Manjinder Singh https://orcid.org/0000-0003-4720-3473

REFERENCES

Abduljabbar, R., Dia, H., Liyanage, S., & Bagloee, S. A. (2019). Applications of artificial intelligence in transport: An overview. *Sustainability*, *11*(1), 189.
Adadi, A., & Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access*, *6*, 52138–52160.
Arrieta, A. B., Díaz-Rodríguez, N., Ser, J. D., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*, 82–115.
Baluja, S. (1994). *Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning* (Technical Report).
Baluja, S. (2002). An empirical comparison of seven iterative and evolutionary heuristics for static function optimization (extended abstract).
Baluja, S., & Caruana, R. (1995). *Removing the genetics from the standard genetic algorithm*. Carnegie Mellon University.
Baluja, S., & Davies, S. (1997). Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space.
Brownlee, A. (2009). *Multivariate Markov networks for fitness modelling in an estimation of distribution algorithm* (PhD thesis). Robert Gordon University.
Brownlee, A. E., Wright, J. A., He, M., Lee, T., & McMenemy, P. (2020). A novel encoding for separable large-scale multi-objective problems and its application to the optimisation of housing stock improvements. *Applied Soft Computing*, *96*, 106650.
Brownlee, A. E. I., Woodward, J., & Swan, J. (2015). Metaheuristic design pattern: Surrogate fitness functions. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation* (pp. 1261–1264). ACM Press.
Collins, T. (2003). Applying software visualization technology to support the use of evolutionary algorithms. *Journal of Visual Languages & Computing*, *14*, 123–150.
Cortez, P., & Embrechts, M. (2013). Using sensitivity analysis and visualization techniques to open black box data mining models. *Information Sciences*, *225*, 1–17.
Cortez, P., & Embrechts, M. J. (2011). Opening black box data mining models using sensitivity analysis. In *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)* (pp. 341–348). IEEE.
Cutello, V., Nicosia, G., Pavone, M., & Stracquadanio, G. (2010). Entropic divergence for population based optimization algorithms. In *IEEE Congress on Evolutionary Computation* (pp. 1–8). IEEE.
Duygu Arbatli, A., & Levent Akin, H. (1997). Rule extraction from trained neural networks using genetic algorithms. *Nonlinear Analysis: Theory, Methods & Applications*, *30*(3), 1639–1648.

Forrest, S., & Mitchell, M. (1993). Relative building-block fitness and the building-block hypothesis. In L. D. Whitley (Ed.), *Foundations of genetic algorithms* (Vol. 2, pp. 109–126). Elsevier.
Goldberg, D. E. (1989a). Genetic algorithms and Walsh functions: Part I, a gentle introduction. *Complex Systems*, *3*, 129–152.
Goldberg, D. E. (1989b). Genetic algorithms and Walsh functions: Part II, deception and its analysis. *Complex Systems*, *3*, 153–171.
Hien, N. T., & Hoai, N. X. (2006). A brief overview of population diversity measures in genetic programming. *Proceedings of the 3rd Asian-Pacific Workshop on Genetic Programming*, 128–139.
Holland, J. H. (1992). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. The MIT Press.
Holland, S. M. (2019). *Principal components analysis PCA*. University of Georgia.
Jin, Y. (2011). Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, *1*(2), 61–70.
Lavinas, Y., Aranha, C., & Ochoa, G. (2022). Search trajectories networks of multiobjective evolutionary algorithms. In J. L. Jiménez Laredo, J. I. Hidalgo, & K. O. Babaagba (Eds.), *Applications of evolutionary computation* (pp. 223–238). Springer.
MacKay, D. J. C. (2003). *Information theory, inference, and learning algorithms*. Cambridge University Press.
Malan, K. (2021). A survey of advances in landscape analysis for optimisation. *Algorithms*, *14*, 40.
Michalak, K. (2019). Low-dimensional Euclidean embedding for visualization of search spaces in combinatorial optimization. *IEEE Transactions on Evolutionary Computation*, *23*(2), 232–246.
Mitchell, M., Forrest, S., & Holland, J. H. (1991). The royal road for genetic algorithms: Fitness landscapes and ga performance. In *Proceedings of the First European Conference on Artificial Life*. MIT Press.
Morrison, R. W., & Jong, K. A. D. (2001). Measurement of population diversity. In P. Collet, C. Fonlupt, J. K. Hao, E. Lutton, & M. Schoenauer (Eds.), *Artificial evolution* (pp. 31–41). Springer.
Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., & Yu, B. (2019). Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, *116*(44), 22071–22080.
Ochoa, G., Christie, L. A., Brownlee, A. E., & Hoyle, A. (2020). Multi-objective evolutionary design of antibiotic treatments. *Artificial Intelligence in Medicine*, *102*, 101759.
Ochoa, G., Malan, K., & Blum, C. (2021a). Search trajectories illuminated. *ACM SIGEVOlution*, *14*, 1–5.
Ochoa, G., Malan, K., & Blum, C. (2021b). Search trajectory networks: A tool for analysing and visualising the behaviour of metaheuristics. *Applied Soft Computing*, *109*, 107492.
Ordish, J., Brigden, T., & Hall, A. (2020). *Black box medicine and transparency*. PHG Foundation.
Pohlheim, H. (2006). Multidimensional scaling for evolutionary algorithms—Visualization of the path through search space and solution space using sammon mapping. *Artificial Life*, *12*(2), 203–209.
Shakya, S., Mccall, J., Brownlee, A., & Owusu, G. (2012). DEUM—Distribution estimation using Markov networks. In S. Shakya & R. Santana (Eds.), *Markov networks in evolutionary computation* (Vol. 14, pp. 55–71). Springer.
Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, *27*(3), 379–423.
Sharma, S., Henderson, J., & Ghosh, J. (2020). CERTIFAI: Counterfactual explanations for robustness, transparency, interpretability, and fairness of artificial intelligence models. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. ACM.
Singh, M., Brownlee, A. E. I., & Cairns, D. (2022). Towards explainable metaheuristic: Mining surrogate fitness models for importance of variables. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO '22* (pp. 1785–1793). Association for Computing Machinery.
Trajanov, R., Dimeski, S., Popovski, M., Korošec, P., & Eftimov, T. (2021). Explainable landscape-aware optimization performance prediction. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)* (pp. 1–8). IEEE.
Trajanov, R., Dimeski, S., Popovski, M., Korošec, P., & Eftimov, T. (2022). Explainable landscape analysis in automated algorithm performance prediction. In J. L. Jiménez Laredo, J. I. Hidalgo, & K. O. Babaagba (Eds.), *Applications of evolutionary computation* (pp. 207–222). Springer.
Wallace, A., Brownlee, A. E. I., & Cairns, D. (2021). Towards explaining metaheuristic solution quality by data mining surrogate fitness models for importance of variables. In M. Bramer & R. Ellis (Eds.), *Artificial Intelligence XXXVIII* (pp. 58–72). Springer International Publishing.
Wright, J., Wang, M., Brownlee, A., & Buswell, R. (2012). Variable convergence in evolutionary optimization and its relationship to sensitivity analysis.

AUTHOR BIOGRAPHIES

Martin Fyvie, a research assistant at the National Subsea Centre (NSC) and Robert Gordon University, focuses on Explainable Artificial Intelligence (XAI), Evolutionary Computing (EC), and non-deterministic solvers. His work includes developing methods to make algorithmic features more accessible and exploring workforce scheduling for green energy transition. Previously a Developer & Data Scientist, developed skills in data science frameworks and client-focused workflows. At NSC, Martin contributes to projects including the Data for Net Zero: UK Continental Shelf Offshore Workforce Planning and aids the Explainable AI for Evolutionary Computing workshop at the Genetic and Evolutionary Computation Conference, pairing academic research with practical solutions in workforce optimization and XAI advancements.

John A. W. McCall is Director of the National Subsea Centre (NSC) at Robert Gordon University. He has researched in machine learning, search and optimisation for 30 years, making novel contributions to a range of nature-inspired optimisation algorithms and predictive machine learning methods, including EDA, PSO, ACO and GA. He has 170+ peer-reviewed publications in books, international journals and conferences. These have received over 3000 citations with an h-index of 24. John specialises in industrially-applied optimization and decision support (Industry 4.0), working with major international companies including BT, BP, EDF, CNOOC and Equinor as well as a diverse range of SMEs.

Lee A. Christie has worked as a Research Fellow on themes of optimisation and AI at Robert Gordon University and University of Stirling. He has been working in this area of research since completing his PhD in 2016 on the topic of pseudo-Boolean optimisation. Recent work with industrial partners includes various projects building efficient transport models and optimising transport networks. He has interests in decentralised computing, machine learning, functional programming and API design. He also teaches Python for business analytics and is part of the steering group for the Aberdeen Python User Group.

Alexander E. I. Brownlee is a Senior Lecturer at the University of Stirling, where he leads the Data Science and Intelligent Systems research group. He is interested in search-based optimisation methods and machine learning, with a focus on decision support tools, and applications in civil engineering, transportation and software engineering. He has published over 80 peer-reviewed papers on these topics. He has worked with several leading businesses including BT and KLM on industrial applications of optimisation and machine learning. He has been an organiser of several workshops and tutorials at international conferences on topics including explainable AI and genetic improvement of software.

Manjinder Singh is a VP at Morgan Stanley in Montreal, and a PhD student with the University of Stirling. He holds a BSc with first class honours from Stirling, where his dissertation focused on using evolutionary computation to generate adversarial examples for NLP systems. His research interests focus on explainable and interpretable AI for metaheurstics.

How to cite this article: Fyvie, M., McCall, J. A. W., Christie, L. A., Brownlee, A. E. I., & Singh, M. (2023). Towards explainable metaheuristics: Feature extraction from trajectory mining. *Expert Systems*, e13494. https://doi.org/10.1111/exsy.13494

APPENDIX A: INFORMATION GAIN RESULTS

A.1 | GA: Global information

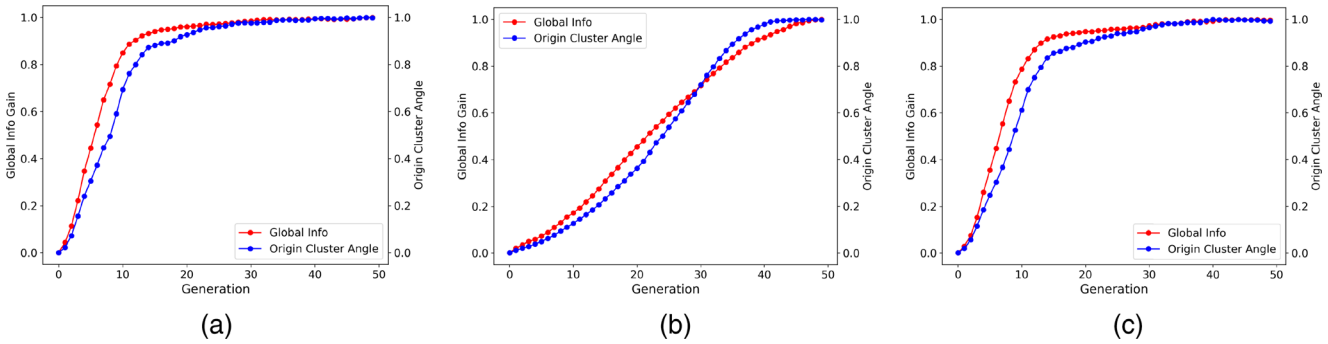

FIGURE A1 GA global information versus PCA angles on 1D checkerboard by selection. (a) GA 1DChecker Trunc20 mean global (b) GA 1DChecker Trunc50 mean global (c) GA 1DChecker tour mean global.

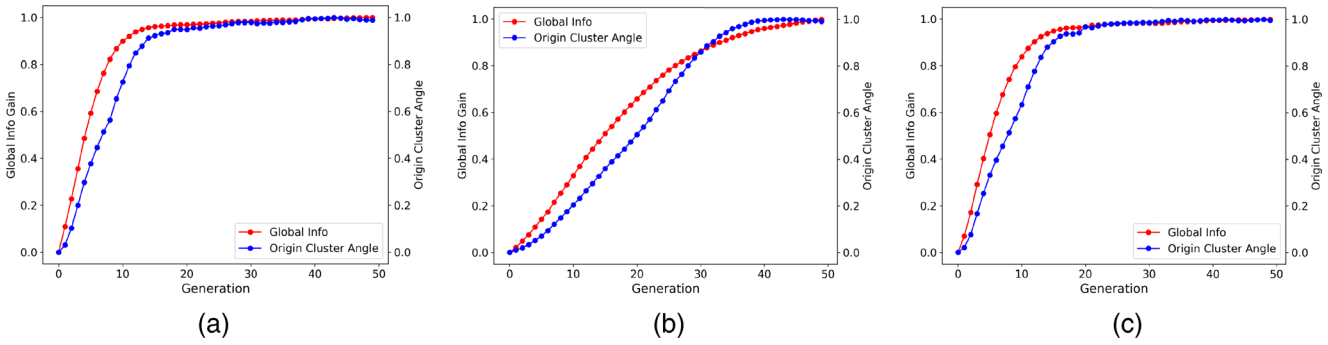

FIGURE A2 GA global information versus PCA on royal road by selection. (a) GA royal road Trunc20 mean global (b) GA royal road Trunc50 mean global (c) GA royal road tour mean global.

A.2 | PBIL: Global information

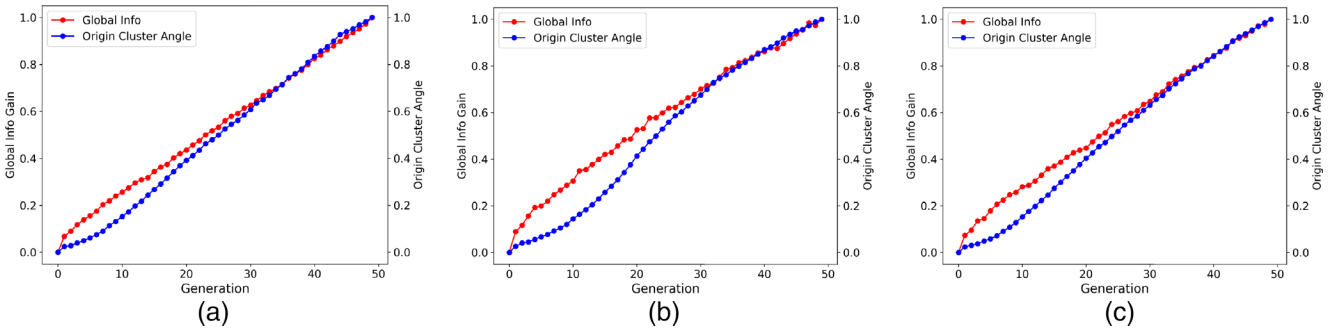

FIGURE A3 PBIL global information versus PCA angles on 1D checkerboard by selection. (a) PBIL 1DChecker Trunc20 mean global (b) PBIL 1DChecker Trunc50 mean global (c) PBIL 1DChecker tour mean global.

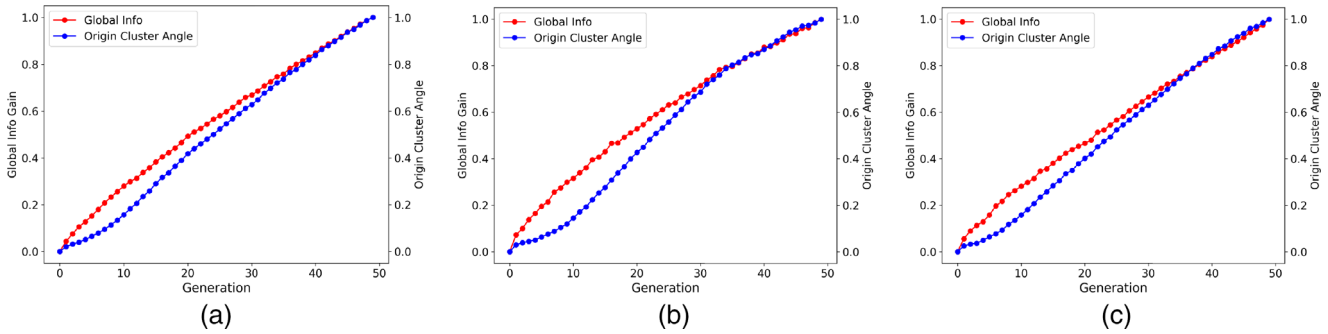

FIGURE A4 PBIL global information versus PCA angles on royal road by selection. (a) PBIL royal road Trunc20 mean global (b) PBIL royal road Trunc50 mean global (c) PBIL royal road tour mean global.

A.3 | GA: Local information

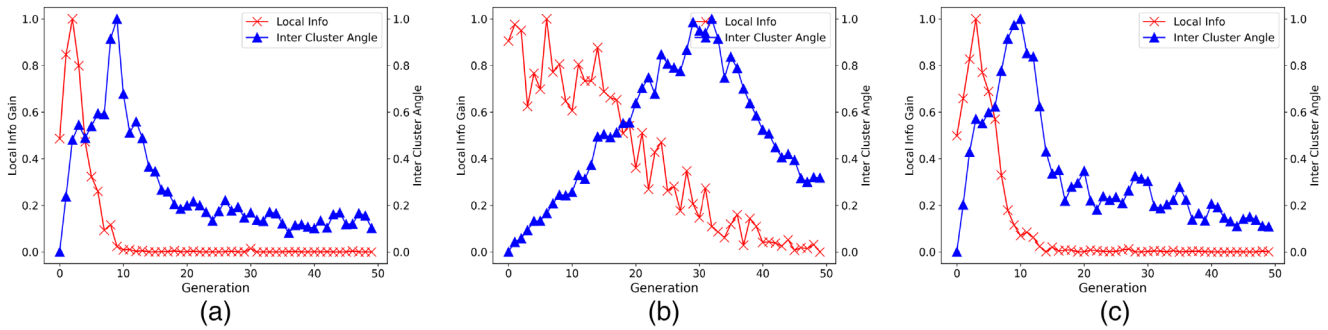

FIGURE A5 GA local information versus PCA angles on checker by selection. (a) GA 1DChecker Trunc20 mean local (b) GA 1DChecker Trunc50 mean local (c) GA 1DChecker tour mean local.

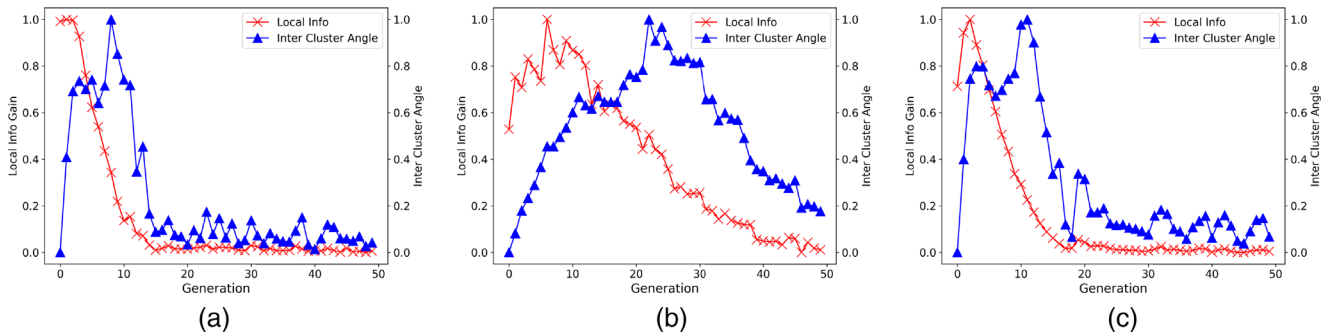

FIGURE A6 GA local information versus PCA angles on royal road by selection. (a) GA royal road Trunc20 mean local (b) GA royal road Trunc50 mean local (c) GA royal road tour mean local.

A.4 | PBIL: Local information

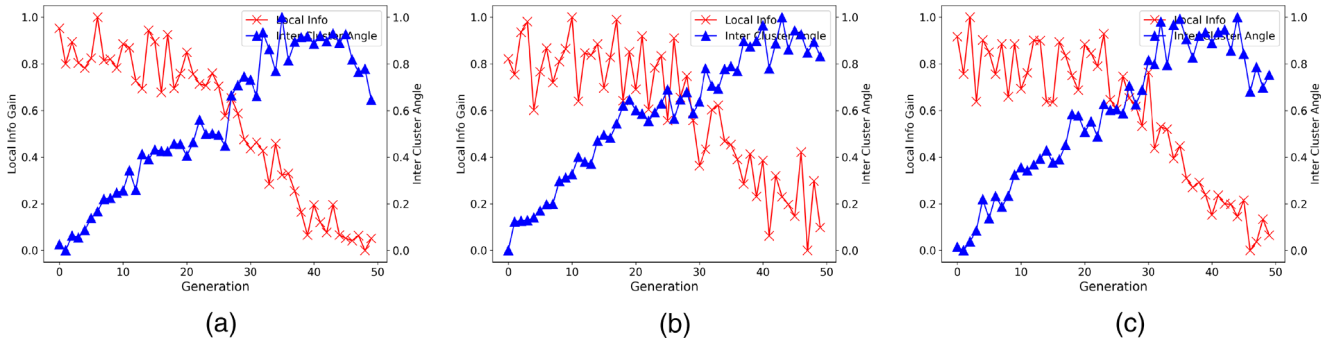

FIGURE A7 PBIL local information versus PCA angles on 1D checkerboard by selection. (a) PBIL 1DChecker Trunc20 mean local (b) PBIL 1DChecker Trunc50 mean local (c) PBIL 1DChecker tour mean local.

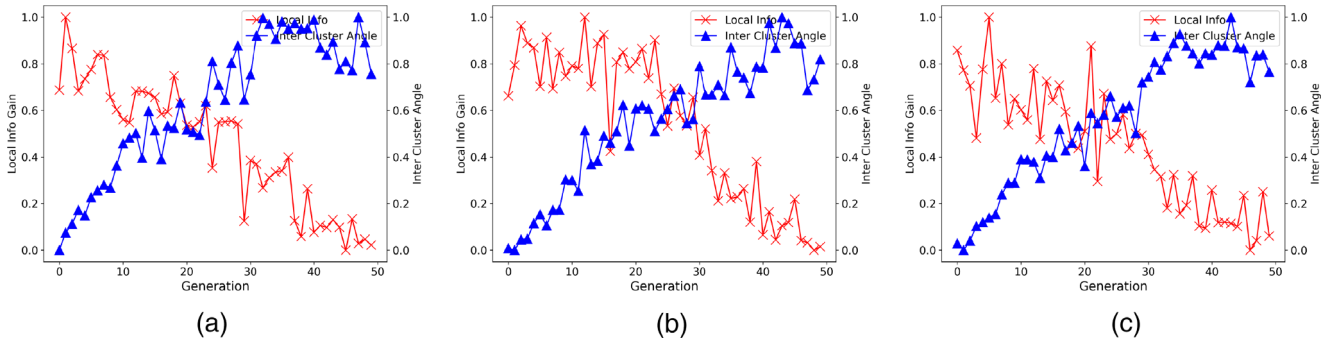

FIGURE A8 PBIL local information versus PCA angles on royal road by selection. (a) PBIL royal road Trunc20 mean local (b) PBIL royal road Trunc50 mean local (c) PBIL royal road tour mean local.