Fitness Landscape Analysis in Data-Driven Optimization: An Investigation of Clustering Problems

Marcus Gallagher[1]

Abstract— **Data-driven optimization problems such as clustering provide a real-world representative source of instances for benchmarking continuous metaheuristic optimization algorithms. Although many metaheurstics and other algorithms have been applied to clustering problems, relatively little research has attempted to explore the structure of the problem space and/or fitness landscape for these problems. In contrast, problem-specific analysis and insights for several classes of combinatorial problems have been developed. This paper investigates the structure of the fitness landscapes of clustering problems, focusing on the fundamental parameters that define problem instances (the dimensionality, number and distribution of the data points and the number of clusters). The paper also provides a general method for active, targeted generation of problem instances based on real-world datasets. The results provide a number of new insights into this family of continuous optimization problems as well as methods and guidelines intended to facilitate better experimental evaluation and comparison of continuous optimization algorithms.**

I. INTRODUCTION

A major focus within Evolutionary Computation and Metaheuristics is the solving of continuous optimization problems assuming only black-box information. Much of this research has focused on the development of algorithms for solving such problems. Experimental evaluation and comparison of these algorithms relies on the application of sound empirical methods and analysis, in conjunction with benchmark or test problem instances.

The performance of an algorithm on a problem instance is clearly dependent on the structure or characteristics of the problem fitness landscape. Exploratory Landscape Analysis [1] is concerned with developing features and techniques to describe the structure of problems. In combinatorial optimization, a catalog of well-defined problem classes exists. For several of these classes, problem-specific features have been used to analyse the fitness landscape [2]. This work provides insights into the complexity of problem instances, for example notions of problem hardness or difficulty with respect to specific algorithms [3], [4].

In continuous black-box optimization however, definitions of problem classes are much less specific. For example, convexity is a foundational concept in mathematical optimization where derivative information about the objective function is used, however black-box optimization is primarily concerned with non-convex problems.

This paper considers clustering as a class of continuous optimization problems from the area of data analysis and machine learning. While a large amount of previous work exists on clustering, the work here is novel in that it provides analysis of the fitness landscape of clustering problems in terms of the key parameters of the problem class. The work contributes new, foundational insights into the nature and complexity of clustering problems. In addition, it provides methods and guidelines for researchers who are interested in improving the evaluation of algorithms with new benchmark problems that offer several important benefits.

An outline of the remainder of the paper is as follows. Section II defines sum of squares clustering problems and reviews the relevant literature. Section III provides some analysis of the structure of some artificial clustering problems in terms of their key parameters. Section IV suggests a general method for active generation of problem instances using real-world datasets. Section V summarizes the key findings of the paper and discusses implications and future work.

II. SUM OF SQUARES CLUSTERING

The Euclidean sum of squared error (SSE) clustering problem (see, e.g. [5], [6]) can be stated as follows. Given a set $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subseteq I\!R^d$ of n data points, determine the locations of a set of k cluster centers in the data space, $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_k\} \in I\!R^d$ to minimize:

$$f(\mathcal{C}|\mathcal{X}) = \sum_{i=1}^{n}\sum_{j=1}^{k} b_{i,j}||\mathbf{x}_i - \mathbf{c}_j||^2 \tag{1}$$

where

$$b_{i,j} = \begin{cases} 1 & if||\mathbf{x}_i - \mathbf{c}_j|| = \min_j ||\mathbf{x}_i - \mathbf{c}_j|| \\ 0 & otherwise. \end{cases} \tag{2}$$

The problem variables are the coordinates of the cluster centres in the data space. Denote the d-dimensional coordinates of the j^{th} cluster centre as $\mathbf{c}_j = (y_j, y_{j+1}, \ldots, y_{j+d})$, then the problem can be rewritten as an unconstrained, continuous optimization problem of dimensionality dk:

$$\min f(\mathbf{y}), \mathbf{y} \in I\!R^{dk} \tag{3}$$

A specific clustering problem instance is therefore defined by a choice of distance metric, a dataset ($\mathcal{X}$) and a value of k. The dimensionality of the underlying optimization problem grows in proportion to d and k, but is independent of the number of data points, n. Clustering is a fundamental task in machine learning and data analysis and so has received considerable attention [5], [6]. Clustering problems have also been widely tackled in evolutionary computation [7] and in the global optimization literature (e.g. [8], [9]).

[1]M. Gallagher is with the School of Information Technology and Electrical Engineering, University of Queensland, Q. 4072, Australia marcusg@uq.edu.au

In general the SSE clustering problem is np-hard [10]; in practice algorithms are applied to find a good solution in an acceptable amount of time. The k-means algorithm is perhaps the most well-known algorithm for clustering [11]. k-means performs a (non black-box) local search and typically converges very quickly to a solution. However, previous work suggests that the fitness landscape of clustering problems can contain a large number of local optima and basins of attraction [12]. As expected, this has a major impact on the performance of a local search algorithm. Indeed, many different techniques have been used to initialize k-means [13], notably the k-means++ technique [14]. These initialization techniques can be viewed as methods of incorporating prior knowledge (or bias) to determine good starting points for the (local) optimization.

In the meantime, the Covariance Matrix Adaptation Evolution Strategy (CMA-ES), using black-box information only is able to find much better solutions than k-means on some clustering problem instances [12]. Evolutionary, metaheuristic and global optimization algorithms have been applied to different clustering problem instances in the literature. However, very little attention has been paid to analysing and understanding the properties of the solution space or fitness landscape of clustering problems. This paper does not aim to show the superior performance of any specific algorithm (evolutionary or otherwise) on a set of clustering problems. Rather the aim is to better understand problems, so that better benchmark problems can be developed to provide more informative experimental evaluations of specific algorithms.

It is of considerable interest in and of itself to better understand the characteristics and complexity of optimization problems. This leads firstly to a better understanding of the behaviour and performance of metaheuristics and nature-inspired algorithms, and ultimately to improvements in the efficiency and quality of solutions to real-world optimization problems. These goals are behind research in exploratory landscape analysis, automated algorithm selection and configuration and benchmarking.

For combinatorial problems such as the travelling salesman problem, knapsack and graph colouring, researchers have studied the nature of problem instances by defining problem features that are specific to that problem class [2]. Given a distance metric, clustering problem instances are at the most general level a function of d, k, n and the distribution/location of the data points, $\mathcal{X}$.

III. LANDSCAPE ANALYSIS

A. General and Previously Examined Properties

Some details of the clustering optimization problem are worth noting. Although the problem is typically formulated as an unconstrained problem (1), this allows for the possibility that one or more cluster centers are not nearest to any of the data points (i.e. for some $\mathbf{c}_j, b_{i,j} = 0 \; \forall \; i$). This means that the landscape is flat in regions far from the dataset [12]. Also, the formulation above (1)-(3) collects cluster centre coordinates into a vector. However, a black-box algorithm has no knowledge of the components ($\mathbf{c}_j$) within this vector. Cluster centres are not labelled, creating a symmetry of $k!$ equivalent wedges in the landscape [12].

Intuitively, each term in the summation of (1) potentially contributes a quadratic (due to Euclidean distance) basin of attraction on the problem landscape. The location and number of data points creates a large number of these quadratic basins of different sizes. Because of the "hard" (min) ownership of data points to cluster centres (2), these quadratic bowls will be connected/intersected with sharp (discontinuous) ridges. The landscapes are therefore typically complex and non-linear, with a large number of local optima and basins of attraction.

B. Clustering/Facility Layout in a Circle

To begin, a specific, simple problem scenario is considered: clustering using n points generated uniformly (for each problem instance) at random inside the unit circle (i.e. $d = 2$). Since there is no "clustered" structure in the data (except by chance), it is useful to think of problem instances as facility location problems (clustering is a specific case of facility location) [12], [15], [16]. Good solutions correspond to spreading out cluster centres (facilities) to "cover" the area of the points. When n is large ($\rightarrow \infty$), the points densely cover the unit circle. For certain values of k we can observe that the global optimum approximates the optimal packing of equal circles inside the unit circle [17].

The fitness landscape of a similar problem (packing equal circles in the unit square), was previously studied in [18], [19]. This work indicates that the landscapes of such problems can contain a variety of complex structure. Experimental results also suggest that the number of circles (cf. k values) effects the performance of algorithms but the relationship is non-monotonic (e.g. an algorithm may have good performance for 6 circles and for 8 circles, while having poor performance for 7 circles). Circles in a square/circle packing problems have some excellent properties for algorithm benchmarking, however they are formulated as constrained problems and so require algorithms to use some form of constraint handling [19]. The corresponding clustering problems are formulated as unconstrained problems, however the cardinality (n) and distribution of the data set brings additional factors to the problem class.

Returning to the circles-in-a-circle packing problems, the $k = 1$ case is simple; the fitness landscape is a convex (quadratic) bowl with the minimum at the circle center, $(0, 0)$. When $k = 2$, the optimal solution is to position the cluster centres opposite each other on a diameter of the circle, centre $(0, 0)$ and radius equal to 0.5. For example, Fig. 1 shows one such optimal circle packing. The packing is invariant to rotations about the centre of the circle in the data space, leading to an infinite number of global optima. For a finite value of n, we have an approximation of this packing problem. Fig. 2 shows 10 example (good but not optimal) solutions for instances of this problem. These were the best solutions found by CMA-ES with default parameter settings on 10 trials. The dataset was regenerated for each of the 10 runs (one of these datasets is shown). This was (4,8) CMA-

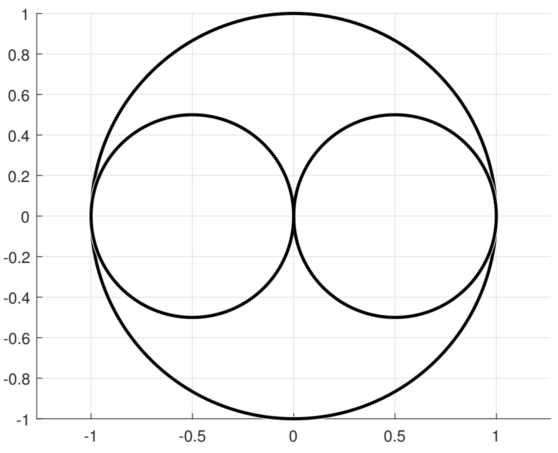

Fig. 1. An optimal packing of two equal circles within the unit circle (based on [17]).

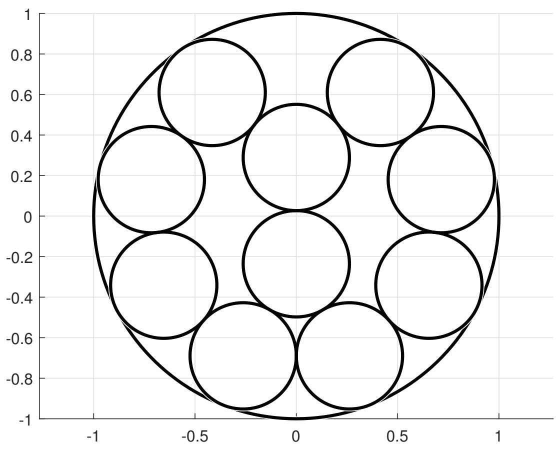

Fig. 3. An optimal packing of 10 equal circles within the unit circle (based on [17]).

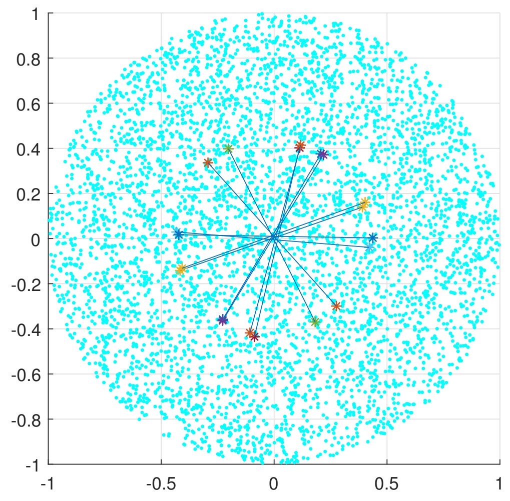

Fig. 2. Example solutions for the SSE clustering problem, with $k = 2$ and data ($n = 5000$) generated uniformly inside the unit circle. The cluster centres of each solution are connected with a line. 10 solutions obtained by CMA-ES are shown.

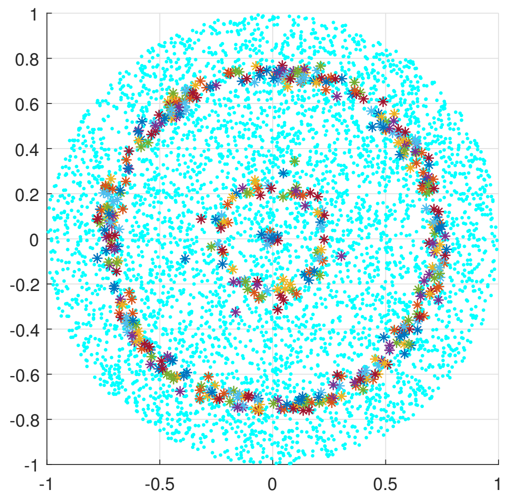

Fig. 4. Example solutions for the SSE clustering problem, with $k = 10$ and data ($n = 5000$) generated uniformly inside the unit circle. 50 solutions obtained by CMA-ES are shown.

ES, which used around 2000-3000 function evaluations per run.

A higher-dimensional circle packing problem instance ($k = 10$) is shown in Fig. 3. For a corresponding SSE clustering problem ($k = 10$), the best solutions found by default CMA-ES over 50 different runs/problem instances are shown in Fig.4. This was (6,12) CMA-ES, which used around 5000-6000 function evaluations per run before terminating. We can see that many of the solutions are close to circle centre positions in the optimal packing (allowing for rotation about the origin). However, there are some outlying points, as well as points at the origin, showing the algorithm has sometimes converged to different solutions.

In summary, we see that even with a very simple data distribution (uniform in the unit circle), clustering problem instances are interesting and useful for algorithm benchmarking. They inherit (approximately) several properties from circle packing problems, can contain many local/global optima and symmetries. Varying the value of k generates problems of different dimensionality. For large n the approximation to the circle packing problem will be good, but as n decreases the approximation will become noisy and problem instances will become sensitive to the specific points in the dataset. It is also useful to be able to visualize the solutions by algorithms in the data space, which is often difficult with other real-world optimization problems.

C. *Exploring the Relationship Between Problem Complexity, k and n*

Given a dataset drawn uniformly in the unit circle, the distribution of the data is fixed. Problem instances are then completely parameterized by the values of k (which defines the dimensionality of the problem, $2k$) and n. From (1), we can see that the fitness/objective function value of candidate solutions will (on average) increase as n increases because of the summation. That is, given a k value, different problem instances with the same dimensionality can be generated, but, e.g. the objective function value of a random solution will tend to decrease. To remove this effect, the objective function can be normalized with respect to n and we have mean squared error rather than SSE:

$$f(\mathcal{C}|\mathcal{X}) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} b_{i,j}||\mathbf{x}_i - \mathbf{c}_j||^2 \tag{4}$$

Changing the value of k means changing the dimensionality of the optimization problem. For a given dataset, candidate solutions will (on average) decrease as k increases, since the distances between points and cluster centers will typically decrease as more cluster centers are added to the solution.

What is less obvious is the relationship between k and n. To investigate this, experiments were carried out by generating problems instances for a range of k and n values. We cannot directly visualise the fitness landscapes because they are high dimensional; possible approaches include using visualization techniques to reduce the dimensionality or to employ landscape features from exploratory landscape analysis. Alternatively, we can look at the effect of the changing landscape structure via the behaviour of an algorithm (random search could be thought of as the "algorithm" underlying most exploratory landscape analysis research). The k-means algorithm, while not black-box, performs a local search and has a reputation for converging rapidly (e.g. after $O(10)$ function evaluations). The algorithm also has a hard convergence, stopping at a local optimum. This makes it easy to record the number of functions evaluations per experiment. k-means was applied to problems over a grid of k and n values. At each (k,n) value pair, 10 different problem instances (i.e. datasets) were generated. k-means depends on a random initialization point, hence the algorithm was run for 50 trials on each problem instance. For initialization, cluster centers were set at a random subset of k points from the dataset. Results were averaged across the 10 different problem instances.

The number of iterations/function evaluations taken to converge by k-means were collected. Fig. 5 shows a very interesting trend in the time the algorithm takes to find a local optimum. For a given value of n, we see that the number of iterations goes through a peak in a specific range of k values. For small values of n this peak is not significant, but seems to become increasingly pronounced as n increases. The peak occurs approximately in the range $n \approx [100k, 1000k]$.

These results show that, at reasonably large values of k and n (e.g. compared to datasets typically used in clustering), there is a critical region of the problem space where the fitness landscape (in some sense) becomes significantly more complex. This is reflected in the behaviour of k-means, which takes much longer to converge to a local optima than for problems outside this region. A possible explanation is that, around this critical value, the "data density" (in terms of the number of points occurring in a given area of the data space) causes a large amount of fluctuation in the minimum distances between points and cluster centers, which determines the updates/steps in k-means. Further work is required to investigate this hypothesis.

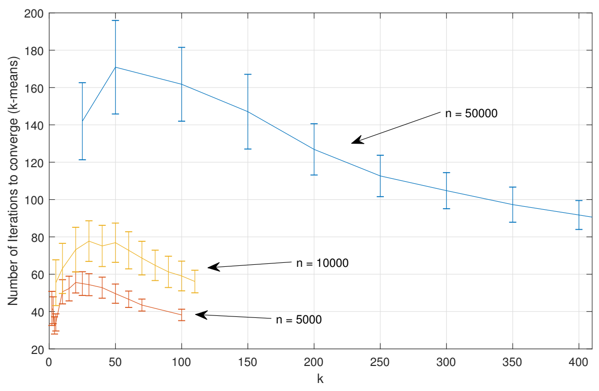

Fig. 5. The number of iterations/function evaluations taken by k-means to converge as a function of k, for several values of n. Data is generated uniformly in the unit circle. Lines are averages over trials and error bars are one standard deviation.

D. *Tuning Problem Complexity by Varying the Data Distribution*

Another way of exploring the complexity of clustering problems is to systematically vary the distribution of the data. Consider the scenario where data is (densely) generated from two circles with unit radii, separated by a distance s between the circle centers along the x-axis (Fig. 6). Now consider the case where $k = 2$. When the circles are well-separated from each other, the globally optimal solution is (approximately) to position a cluster centre at the centre of each circle. If we let $s \to 0$ (i.e. until the circles completely overlap), then we effectively have the one-circle problem and the optimal solution changes to that discussed above for packing two circles in the unit circle (see Figs. 1 and 2). In between these cases, at some critical value of s, the objective function values of the one-circle and the two-circle global optima will become equal. To see this, datasets ($n = 1000$) were generated from the two circles for a range of values $s = [0, 3]$. Fig. 7 shows the values of these two solutions as a function of s. The critical value is $f \approx 0.389$ at $s \approx 0.560$. We can see that the one-circle optimum becomes monotonically worse as s increases, growing quadratically due to the form of f. The two-circle optimal solution initially improves its value as s increases (when many points are in the overlap of the circles) and attains its minimum value at (or very close to) the critical point. As s increases further,

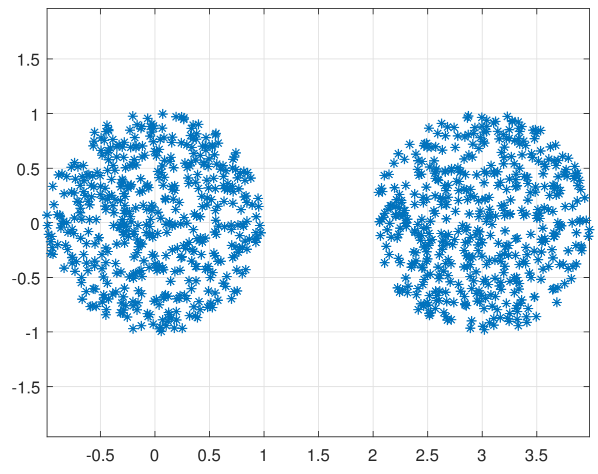
Fig. 6. Data generated from two circles with unit radii. Shown here is when $s = 3$ and each circle contains 500 data points.

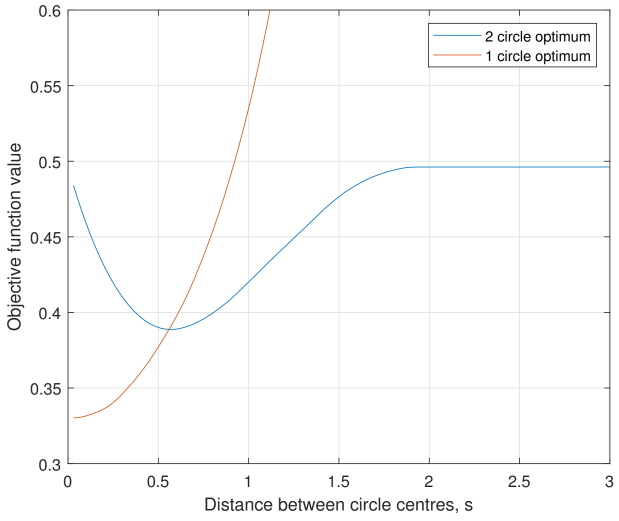

Fig. 7. Objective function values of one and two-circle optimal solutions as a function of separation between the circles.

the overlap between the circles disappears and the objective function value grows, until the circles become completely separated ($s \approx 2$). For $s > 2$, the "ownership" of points to cluster centers does not change, hence the objective function value becomes constant. This is the effect of the unconstrained problem formulation discussed previously.

As a simple test of this critical value, default CMA-ES was run on three problem instances:

(a) The data circles completely overlap ($s = 0$),
(b) The data circles are separated (approximately) at the critical value ($s = 0.56$),
(c) The data circles are well-separated ($s = 3$).

A set of 500 data points was randomly generated from each circle ($n = 1000$) and CMA-ES was run 50 times on each problem instance. For problems (a) and (c), the algorithm converged with high precision ($\approx 10^{-7}$) to the same solution on all 50 runs. For problem (b) however, the algorithm converged to two relatively different solutions ($\approx 10^{-1}$): 43 runs converged to one solution while 7 runs converged to the other. This deserves a more significant experimental study, but nevertheless indicates that the behaviour of CMA-ES is sensitive to the value of s for these problem instances. A similar effect would be expected if the problem scenario was varied to use a different value of k.

IV. ACTIVE GENERATION OF CLUSTERING PROBLEMS FOR BENCHMARKING

A. *Perturbing Datasets to Generate Problem Instances*

In the previous Section, the distribution of the data was first specified and then problem instances were generated from that distribution. An infinite number of randomised problem instances can be generated in this way by sampling from a specified distribution. In real-world clustering problems however, only a fixed, finite dataset $\mathcal{X}$ is typically available. In this Section, one possible way of utilizing real-world datasets for algorithm benchmarking is proposed to purposefully modify the dataset to target the goal of the experiment. For example, if we have an algorithm that we would like to adversarialy "stress-test", we can modify the dataset to produce problem instances where the performance of the algorithm drops. Given a value of k, any subset of a given dataset defines a problem instance. A standard way to represent a subset selection problem is to define indicator variables, $\alpha_i, 1 \leq i \leq n$ for each data point in $\mathcal{X}$. We can then apply an algorithm to solving the (meta) binary optimization problem of determining the values of the α_i. This leads to the following method for active problem generation:

- **Given:** a dataset and a suitable performance criterion.
- Use a meta-optimization algorithm (minimize or maximize the performance criterion with respect to the α_i) to perform subset selection.
- **Returns:** a subset of the data, which defines a new clustering problem instance.

Note that any algorithm(s) or performance criterion could be used within this method. To illustrate, the Ruspini dataset ($n = 75$, $d = 2$) was selected, which has been widely used to benchmark clustering algorithms [12]. Default CMA-ES was used as the test algorithm with $k = 10$ (i.e. a 20-D optimization problem). A simple stochastic local search was used to solve the meta-problem by "pruning". Starting with the original dataset, a point was randomly removed and CMA-ES was run and its performance compared with the performance on the original dataset. If performance *decreased*, the point was left out of the dataset, otherwise it was replaced and the algorithm iterated. The performance criterion used here was the number of function evaluations used by default CMA-ES before termination, averaged over 10 trials on each problem instance.

The results of an example run of this method is shown in Fig. 8. The default was (6,12) CMA-ES, initialized from

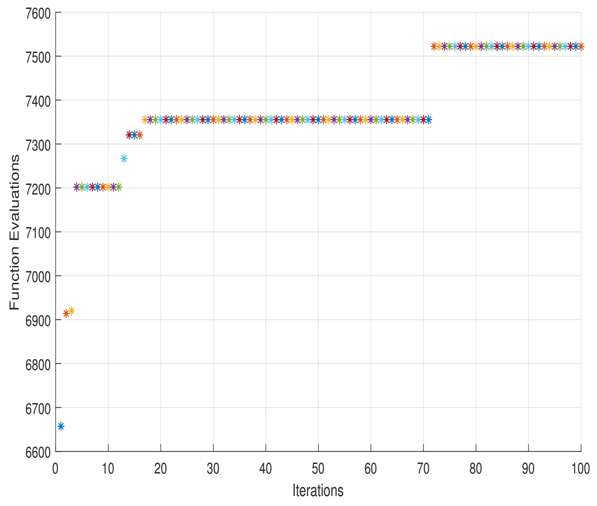

Fig. 8. Average number of function evaluations taken by (6,12) CMA-ES as a function of the meta-local search algorithm. The meta-search removes points from the Ruspini dataset to increase the average number of function evaluations taken by CMA-ES.

a Gaussian distribution centered at the average of the data with standard deviation equal to $\frac{2}{3}$ of the maximum range of the data over both dimensions. After 100 iterations, we can see that the local search has removed 7 points from the dataset. This is a small fraction of the dataset ($\approx 10\%$) but it creates a significant change in the algorithm performance, with the average number of function evaluations increasing from under 6700 to over 7500.

The result shown here is only intended to serve as proof-of-concept. The local search used is very simple and has only been run for a small amount of iterations. If a state-of-the-art evolutionary algorithm was used with more computing time, it is very likely that this result would be considerably improved. Also, the Ruspini dataset is small, which is likely to limit the capacity to optimize some criteria for the meta-search. Larger datasets should offer more scope for developing interesting problem instances.

B. Related Work

Several other researchers have taken a similar approach to generating test problem instances for benchmarking continuous metaheuristics, for example using genetic programming [20] or genetic algorithms [21]. Tunable landscape generators can also be found in combinatorial optimization (e.g. the well-known NK-landscape model [22]) as well as work on evolving problem instances [23]. Recent work also carried out gradual transformations of well-known artificial benchmark functions to evaluate problem features [24], [25]. The novelty here is that black-box optimization problem instances are generated through modifications to a dataset. This is similar to “bumping” which was proposed in the context of model selection in machine learning [26]. More generally, similar ideas exist in the context of active learning [27]. The method described above could be used beyond clustering for any data-driven optimization problems (i.e. where the objective function is defined over the points in a dataset).

V. CONCLUSIONS

This paper has presented some analysis of the fitness landscapes of sum of squares clustering problems. A number of properties of clustering problems have been discussed and developed in terms of the key parameters of the problem class. The work provides methods and guidelines for researchers who are interested in improving the evaluation of algorithms with new, tunable benchmark problems. A summary of the key findings of the paper is as follows:

- When the data distribution is non-informative (e.g. uniform), the problems have similarities with geometric packing problems. For given values of k and n, randomized problem instances can easily be generated for benchmarking.
- Problem instances with a variety of features can be generated by varying the dimensionality of the optimization problem (k and d) and the number of data points, n.
- There appears to be a critical region of the problem space with respect to $\frac{k}{d}$ where problem instances become considerably more complex (at least with respect to the time taken for k-means to converge to a local optimum).
- A spectrum of problem instances can be generated by controlled variation of the data distribution. An example is studied with data from two unit circles. Varying the number of data points controls the quality of the approximation of problem instances to the underlying data distribution.
- A method for active, controlled generation of problem instances using real-world datasets is presented, which can be applied to any black box algorithm using any suitable performance criteria.

There are several avenues for possible future work. The critical region for k-means on the one-circle problem should be investigated further. A number of combinatorial problems have been analysed and found to contain phase transitions with regions of hard and easy problems (e.g. [28], [29]). It would be interesting to see if similar structure can be identified in some sense in a continuous optimization problem class. The critical point identified when moving from the one-circle to the two-circle problem is also interesting. This phenomena should be approachable theoretically, as well as experimentally by generating problem instances around the critical value or creating similar test problem conditions (e.g. with a larger number of circles/value for k.

This paper has used the Euclidean distance metric, but clustering problems are defined using other metrics. Interesting and challenging problem instances may be found in the large existing literature on clustering. It would also be useful to carry out a large experimental study to establish some comparative results and identify a set of specific clustering problem instances which could be used by other researchers.

ACKNOWLEDGMENT

The author thanks Millie Macdonald for carrying out and validating initial experiments for some parts of this paper. This work was supported by the Dagstuhl Seminars 13721 and 16412.

REFERENCES

[1] O. Mersmann, B. Bischl, H. Trautmann, M. Preuss, C. Weihs, and G. Rudolph, "Exploratory landscape analysis," in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. ACM, 2011, pp. 829–836.

[2] K. Smith-Miles and L. Lopes, "Measuring instance difficulty for combinatorial optimization problems," *Computers & Operations Research*, 2012 (online 12 July 2011).

[3] O. Mersmann, B. Bischl, H. Trautmann, M. Wagner, J. Bossek, and F. Neumann, "A novel feature-based approach to characterize algorithm performance for the traveling salesperson problem," *Annals of Mathematics and Artificial Intelligence*, vol. 69, no. 2, pp. 151–182, 2013.

[4] K. Smith-Miles, B. Wreford, L. Lopes, and N. Insani, "Predicting metaheuristic performance on graph coloring problems using data mining," in *Hybrid metaheuristics*. Springer, 2013, pp. 417–432.

[5] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM computing surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.

[6] R. Xu and D. Wunsch, *Clustering*. John Wiley & Sons, 2008, vol. 10.

[7] E. R. Hruschka, R. J. Campello, A. A. Freitas *et al.*, "A survey of evolutionary algorithms for clustering," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 39, no. 2, pp. 133–155, 2009.

[8] A. M. Bagirov, "Modified global k-means algorithm for minimum sum-of-squares clustering problems," *Pattern Recognition*, vol. 41, no. 10, pp. 3192–3199, 2008.

[9] P. Hansen and B. Jaumard, "Cluster analysis and mathematical programming," *Mathematical programming*, vol. 79, no. 1-3, pp. 191–215, 1997.

[10] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, "Np-hardness of euclidean sum-of-squares clustering," *Machine learning*, vol. 75, no. 2, pp. 245–248, 2009.

[11] D. Steinley, "K-means clustering: A half-century synthesis," *British Journal of Mathematical and Statistical Psychology*, vol. 59, pp. 1–34, 2006.

[12] M. Gallagher, "Towards improved benchmarking of black-box optimization algorithms using clustering problems," *Soft Computing*, vol. 20, no. 10, pp. 3835–3849, 2016.

[13] M. E. Celebi, H. A. Kingravi, and P. A. Vela, "A comparative study of efficient initialization methods for the k-means clustering algorithm," *Expert systems with applications*, vol. 40, no. 1, pp. 200–210, 2013.

[14] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[15] J. Brimberg, P. Hansen, N. Mladenovic, and E. D. Taillard, "Improvements and comparison of heuristics for solving the uncapacitated multisource Weber problem," *Operations Research*, vol. 48, no. 3, pp. 444–460, 2000.

[16] S. Salhi and M. D. H. Gamal, "A genetic algorithm based approach for the uncapacitated continuous location-allocation problem," *Annals of Operations Research*, vol. 123, pp. 230–222, 2003.

[17] E. Specht, "Packomania," Retrieved from http://www.packomania.com/, Accessed January, 2019.

[18] R. Morgan and M. Gallagher, "Fitness landscape analysis of circles in a square packing problems," in *Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'14)*. Springer, 2014, pp. 455–466.

[19] P. A. Bosman and M. Gallagher, "The importance of implementation details and parameter settings in black-box optimization: a case study on gaussian estimation-of-distribution algorithms and circles-in-a-square packing problems," *Soft Computing*, vol. 22, no. 4, pp. 1209–1223, 2018.

[20] W. Langdon and R. Poli, "Evolving problems to learn about particle swarm optimizers and other search algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 5, pp. 561–578, Oct. 2007.

[21] K. Smith-Miles and S. Bowly, "Generating new test instances by evolving in instance space," *Computers & Operations Research*, vol. 63, pp. 102–113, 2015.

[22] S. A. Kauffman and E. D. Weinberger, "The nk model of rugged fitness landscapes and its application to maturation of the immune response," *Journal of theoretical biology*, vol. 141, no. 2, pp. 211–245, 1989.

[23] J. I. van Hemert, "Evolving combinatorial problem instances that are difficult to solve," *Evolutionary Computation*, vol. 14, no. 4, pp. 433–462, 2006.

[24] S. Saleem, M. Gallagher, and I. Wood, "A model-based framework for black-box problem comparison using gaussian processes," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2018, pp. 284–295.

[25] ——, "Direct feature evaluation in black box optimization using problem transformations," *Evolutionary computation*, pp. 1–24.

[26] R. Tibshirani and K. Knight, "Model search by bootstrap bumping," *Journal of Computational and Graphical Statistics*, vol. 8, no. 4, pp. 671–686, 1999.

[27] B. Settles, "Active learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 6, no. 1, pp. 1–114, 2012.

[28] D. Achlioptas, A. Naor, and Y. Peres, "Rigorous location of phase transitions in hard optimization problems," *Nature*, vol. 435, no. 7043, pp. 759–764, 2005.

[29] I. P. Gent and T. Walsh, "The tsp phase transition," *Artificial Intelligence*, vol. 88, no. 1-2, pp. 349–358, 1996.