Soft Comput
DOI 10.1007/s00500-016-2408-3

METHODOLOGIES AND APPLICATION

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

Peter A. N. Bosman[1] · **Marcus Gallagher**[2]

Abstract We consider a scalable problem that has strong ties with real-world problems, can be compactly formulated and efficiently evaluated, yet is not trivial to solve and has interesting characteristics that differ from most commonly used benchmark problems: packing n circles in a square (CiaS). Recently, a first study that used basic Gaussian EDAs indicated that typically suggested algorithmic parameter settings do not necessarily transfer well to the CiaS problem. In this article, we consider also AMaLGaM, an enhanced Gaussian EDA, as well as arguably the most powerful real-valued black-box optimization algorithm to date, CMA-ES, which can also be seen as a further enhanced Gaussian EDA. We study whether the well-known performance on typical benchmark problems extends to the CiaS problem. We find that although the enhancements over a basic Gaussian EDA result in superior performance, the further efficiency enhancements in CMA-ES are not highly impactful. Instead, the most impactful features are how constraint handling is performed, how large the population size is, whether a full covariance matrix is used and whether restart techniques are used. We further show that a previously published version of AMaLGaM that does not require the user to set the the population size parameter is capable of solving the problem and we derive the scalability of the required number of function evaluations to solve the problem up to 99.99 % of the known optimal value for up to 30 circles.

Keywords Black-box optimization · Evolutionary computation · Estimation-of-distribution algorithms · Parameter tuning · Constraint handling · Circles-in-a-square packing

Communicated by V. Loia.

✉ Peter A. N. Bosman
Peter.Bosman@cwi.nl

Marcus Gallagher
marcusg@uq.edu.au

[1] Peter A.N. Bosman Centrum Wiskunde & Informatica (CWI), P.O. Box 94079, 1090 GB, Amsterdam, The Netherlands

[2] School of Information Technology and Electrical Engineering, The University of Queensland, 4072 Brisbane, Australia

1 Introduction

Estimation-of-distribution algorithms (EDAs) build and use probabilistic models during optimization in order to automatically discover and exploit structural features of an optimization problem (Larranaga 2001; Pelikan et al. 2006). EDAs are especially applicable in black-box optimization (BBO) where no assumptions are made on the problem being solved. This is frequently the case when solving complex real-world problems. In the design of real-valued BBO algorithms, important advances have been made and some well-designed sets of benchmark problems now exist that are commonly used to compare and evaluate the performance of algorithms. An excellent example of such a benchmark is known as the black-box optimization benchmarking (BBOB) (Hansen et al. 2010) set, which has been carefully designed to facilitate meaningful algorithm comparisons across artificial problems that are scalable in dimensionality and have known structural features. The BBOB set has been widely adopted in the evolutionary computation and metaheuristics communities. Across a large number of algorithms tested on BBOB, an enhanced Gaussian EDA known as AMaLGaM (Bosman et al. 2013) and the well-known and related CMA-ES (Hansen 2006) algorithm produced among the best results.

Published online: 15 October 2016

Despite these advances in the experimental evaluation of algorithms, there remains a significant gap between results on artificial benchmarks and the successful application of new algorithms to real-world problems. Real-world problems impose conditions and requirements that are often abstracted away in benchmark studies, where the primary concern is to better understand the algorithm. In addition, there is always the question of whether benchmark sets such as BBOB are sufficiently representative of real-world problems. Here, we aim to take a step in the direction of bridging this gap by considering a problem that ticks many of the boxes of things that are generally considered to be important when evaluating evolutionary algorithms (Whitley et al. 1996). The problem we consider is scalable, has strong ties with real-world problems, can be compactly formulated and efficiently evaluated, yet is not trivial to solve and has interesting characteristics that differ from commonly used benchmark problems: packing n circles in a square (CiaS). When faced with a complex, real-world problem that can only be tackled from a BBO perspective, the underlying problem difficulty could well have been that of the more complex problems in the BBOB benchmark suite, but it could also have been that of CiaS, which we argue has different, yet important properties. Nevertheless, our focus is still firmly on evaluating algorithms rather than claiming state-of-the-art performance for specific real-world packing problems. In other words, it should be clear that the focus of this article is not to design and experiment with (adaptations of) evolutionary algorithms that are specifically meant for and achieve state-of-the-art results on CiaS. For this, specific properties of the CiaS problem would have to be considered and exploited as is done by many specific heuristics already exist. Instead, while knowing the problem is CiaS, we still take a BBO approach, assuming only that we know the types of variables and their ranges and that these ranges must be considered as hard constraints.

Recently, a first study that used elementary Gaussian EDAs indicated that typically suggested algorithmic parameter settings do not necessarily transfer well to the CiaS problem (Gallagher 2012). Given that AMaLGaM was designed to overcome certain drawbacks in basic Gaussian EDAs and that, especially on smooth, quadratic surfaces, CMA-ES is even more efficient, we consider whether the issues observed with basic Gaussian EDAs when solving CiaS are overcome by these algorithms. Especially within the EDA community, the topic of factorizing distributions to limit model complexity has received much attention. We therefore consider also the benefits of estimating a Gaussian distribution with a full covariance matrix over using a univariate factorization for solving CiaS. We further consider the added value of a restart strategy and the impact of handling constraints since the CiaS problem has simple (bounding box), but hard constraints. To study these issues, we will use all the aforementioned algorithms to conduct various experimental analyses. In the context of real-world representative problems, we aim to highlight the experimental choices that must be made in practice but yet are often overlooked or not described in detail in papers that use artificial benchmark problems.

2 Background

In this section, we present a brief overview on the general notion of BBO in the case of continuous, real-valued variables, the evolutionary algorithms that we consider in this article as well as the specific optimization problem at hand: packing n circles in a square.

2.1 Continuous black-box optimization

Since we take a black-box optimization (BBO) perspective, we consider the following general formulation of a continuous/real-valued optimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^\ell} \{f(\mathbf{x})\} \textbf{ s.t. } c(\mathbf{x}) = 0 \tag{1}$$

where ℓ is the number of real-valued problem variables and it is further assumed that the only information available is the ability to evaluate the objective function, f and the constraint function c, at a candidate solution, $\mathbf{x}$. In Eq. 1 we assume, without loss of generality, that objective function f is to be minimized. Function c is 0 if and only if a solution is feasible. This is the most general definition of a BBO problem in the real-valued domain as we make no further assumptions on either f or c. The latter assumption is, however, often violated slightly even in what is still considered BBO, assuming that (1) we know at least the allowed range of values that each variable x_i can take (often a contiguous interval $[x_i^{\min}, x_i^{\max}] \subseteq \mathbb{R}$) and (2) we have a way to ensure that function c is not just a binary indicator function but that it also provides a notion of *how much* a solution violates the constraints, making function c, and consequently the entire problem, far more searchable. Making further assumptions moves the problem further away from BBO. Although this may well be possible for a real-world problem, in this article we assume that this is not the case. The reason for this is that for the best performance of the optimization algorithm, it is typically very important to exploit such assumptions as much as possible. This consequently also makes the designed algorithms problem specific, which is specifically not the aim of this article. We specifically want to consider algorithms meant to target BBO, to which additions can be made should more problem-specific information be available.

2.2 Estimation-of-distribution algorithms

All the algorithms that we consider are evolutionary algorithms (EAs) that maintain a population of n solutions upon which selection and variation are performed. Furthermore, they all follow, at the top level, the same principle: maintain an ℓ-dimensional Gaussian distribution and sample new solutions from this distribution as a means of generating future search points. All algorithms can thus be considered to be of the estimation-of-distribution algorithm (EDA) family and we use a single framework for them. At each generation, $\lfloor \tau n \rfloor$ of the best solutions are selected and the Gaussian distribution is updated using these solutions. Here, τ is the so-called selection percentile ($\tau \in [\frac{1}{n}, 1]$). For all algorithms considered, $n - 1$ new solutions are then generated, replacing the $n - 1$ worst solutions and keeping only the currently best solution in the population. The main difference between all considered algorithms then is how the Gaussian distribution is updated and used to generate new solutions.

2.2.1 MaLGaM

Perhaps, the most straightforward approach is to use well-known maximum likelihood estimates for the parameters (i.e., the mean vector and the covariance matrix) of the Gaussian distribution. This basic Gaussian EDA, with and without factorizations of the distribution, has been studied before within the framework known as IDEA (Bosman and Thierens 2000) as well as separately, under the name EMNA (Larrañaga et al. 2001; Larranaga 2001). Here, we refer to this algorithm as maximum-likelihood Gaussian model (MaLGaM) for its relation to AMaLGaM and iAMaLGaM, which are, respectively, adapted and incremental adapted versions of MaLGaM.

2.2.2 AMaLGaM

AMaLGaM uses the same maximum likelihood estimates for the model parameters as MaLGaM, but it increases the values in the covariance matrix adaptively, based on how far from the current model mean the improvements were found. It samples solutions directly from the Gaussian distribution, but it additionally attempts to move a subset of these new solutions along the direction of the anticipated mean shift, which is given by the difference between the current mean and the mean estimated in the previous generation. These additions, respectively, prevent the EDA from converging prematurely and accelerate its movement along improving directions. The proposed value for the selection parameter is $\tau = 0.35$, i.e., the fraction of best solutions in the population retained for subsequent model building. For more details, including the required formulas, we refer the interested reader to the relevant literature (Bosman et al. 2013).

2.2.3 iAMaLGaM

Although AMaLGaM has been found to be more robust than MaLGaM and unlike MaLGaM capable of scalably solving problems such as the Rosenbrock function, the minimum required population size for AMaLGaM (to solve a benchmark set of quadratic functions) is relatively large. The proposed guideline in this case is $n \geq 3\ell^{1.5} + 17$. For this reason, the covariance matrix and the anticipated mean shift are estimated incrementally in iAMaLGaM, meaning that the existing values for these parameters are multiplied by a discount factor and additively combined with the maximum likelihood estimates of the current generation. As a result, the minimally required population size is considerably reduced and the proposed guideline becomes $n \geq 10\ell^{0.5}$, effectively reducing the expected number of required function evaluations on the aforementioned benchmarks. However, this reduction in population size was also observed to result in less reliable behavior on highly multimodal problems. The proposed selection percentile is again $\tau = 0.35$. For more details, including the required formulas, we refer the interested reader to the relevant literature (Bosman et al. 2013).

2.2.4 CMA-ES

The covariance matrix adaptation evolution strategy (CMA-ES) has most in common with iAMaLGaM in that the parameters of the search distribution are updated incrementally. However, in CMA-ES the updates are far more involved. A key difference is that instead of using maximum likelihood estimates from the set of selected solutions, the mean that is used in the covariance matrix estimate is that of the previous generation rather than the current generation. Furthermore, all solutions are weighted on the basis of their fitness ranks, both in the estimate of the mean and the covariance matrix, and an evolution path is maintained that can be seen as a more involved variant of the anticipated mean shift. The formulas in CMA-ES reduce the minimally required population size to a very small number, aligned with the intent of CMA-ES, to first achieve fast (optimal), competitive behavior on the functions like the sphere function, before robust behavior, which is more specific of AMaLGaM. The guideline for the population size in CMA-ES is $n \geq 4 + \lfloor 3\log(\ell) \rfloor$. The proposed selection percentile is $\tau = 0.5$. For more details, including the required formulas, we refer the interested reader to the relevant literature (Hansen 2006).

2.3 The circles-in-a-square packing problem

The CiaS problem is a well-studied two-dimensional geometric packing problem. Conceptually, it is defined as follows.

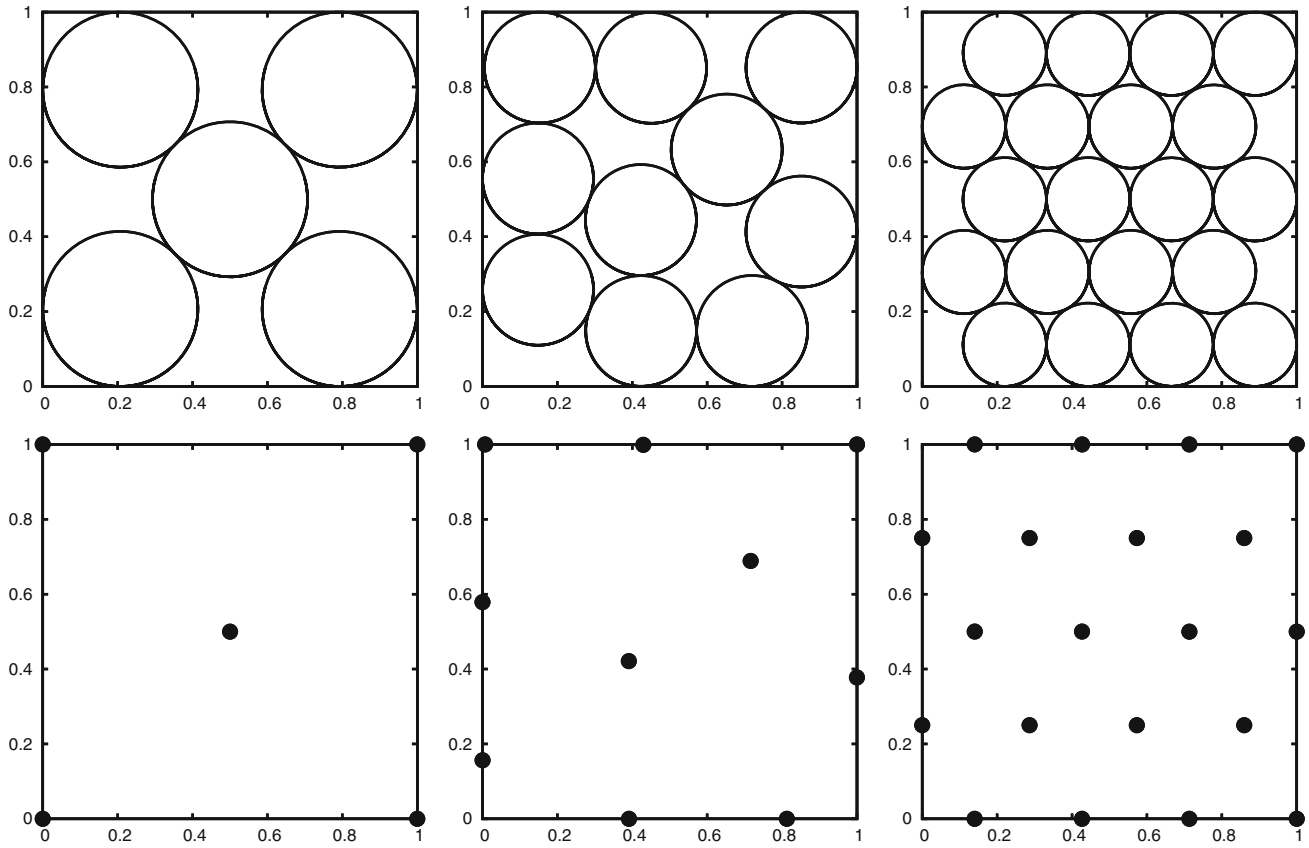

Fig. 1 *Top row* optimal circle packings for $n_c = 5, 10, 20$ circles. *Bottom row* corresponding optimal point scatterings

Determine the maximum radius for placing n_c circles inside a unit square such that they all have that same radius and do not overlap or fall outside of the unit square. In other words, position the circles and compute the radius of the circles such that the circles occupy the maximum possible area within the unit square. The solution for $n_c = 1$ is trivial (a circle placed at the center of the square with the maximum radius of $\frac{1}{2}$). For $n_c \geq 2$, the problem is equivalent to find an optimal scattering of n_c points in the unit square, meaning that the smallest distance between any pair of points is maximized (Addis et al. 2008). Examples of optimal circle packings and their corresponding point scatterings are shown in Fig. 1.

2.3.1 Problem formulation

It is the point scattering formulation that we shall directly use in our optimization algorithms. Moreover, because we take a BBO perspective, we formulate the optimization problem here in a manner that can be used straightforwardly, i.e., in terms of ℓ real-valued variables $x_0, x_1, \ldots, x_{l-1}$. Note that the CiaS problem thereby is only defined for $\ell \geq 4$ and even. We encode the horizontal and vertical position of the ith point by variables x_{2i} and x_{2i+1}, respectively. The problem to be solved then becomes

$$\max_{\mathbf{x}\in\mathbb{R}^l} \{f_{\text{Scatter}}(\mathbf{x})\} \ \textbf{s.t.}\ x_i \in [0; 1],\ i \in \{0, 1, \ldots, l-1\} \tag{2}$$

where $f_{\text{Scatter}}(\mathbf{x}) = \min_{0\leq i<j<\frac{l}{2}} \left\{\|(x_{2i}, x_{2i+1}) - (x_{2j}, x_{2j+1})\|\right\}$ and $\|\mathbf{x}\|$ is the Euclidean norm, or length, of vector $\mathbf{x}$. It is known that for any solution $\mathbf{x}$, the value for the original CiaS problem is given by

$$f_{\text{CiaS}}(\mathbf{x}) = \frac{f_{\text{Scatter}}(\mathbf{x})}{2(f_{\text{Scatter}}(\mathbf{x}) + 1)} \tag{3}$$

Castillo et al. (2008) present a survey of applications that involve circle packing: including cutting, container loading, cylinder packing, facility dispersion, communication networks and facility and dashboard layout problems. Circle packing problems can be considered as simplified versions of such real-world problems and have received a large amount of attention in the literature (see Castillo et al. (2008) for a recent overview). Provably optimal packings have been found for every $n_c \leq 30$ as well as for some special cases with more

circles, using either theoretical or computational approaches (see Szabo et al. (2005) and the references therein). For larger values of n_c, finding provably optimal packings in general becomes increasingly difficult and time-consuming. The Packomania Web site (Specht 2013) maintains a large list of the optimal (or best known) packings (currently with most values from $n_c = 2$ up to $n_c = 9996$), along with references and other related resources.

Circle packing problems form a challenging class of optimization problems. They can generally not be solved using analytical approaches or via gradient-based optimization. These problems are also believed to contain an extremely large number of local optima. For the problem of packing equal circles into a larger circular region, a computational approach was previously used to estimate the number of local optima by running a local optimization algorithm over a large number of trials (Grosso 2010). Although a conservative estimate, results indicate that the number of local optima grows unevenly but steadily, with at least 4000 local optima for $n_c = 25$ and more than 16,000 local optima for $n_c = 40$.

2.3.2 Constraint handling

A key difference with many benchmark problems (including those in BBOB) is that CiaS has hard constraints, i.e., every variable must always lie in the unit range [0; 1]. It is fair to assume that real-world problems typically have such simple, but hard constraints and that this information is known even in a BBO setting. A straightforward practical option to handle these constraints is to continue to resample solutions until all constraints are met. However, as the dimensionality of the problem increases, this is inefficient and time-consuming. Therefore, we will use general constraint-handling techniques that are commonly encountered in the evolutionary computation literature (Coello 2002). The techniques we will consider are straightforward to incorporate in any algorithm. However, apriori it is unclear which method is likely to work best and whether different methods have a different impact at all. Significant effort could be devoted to fine-tuning a specific constraint-handling technique, but this reduces the generality of the results as well as calling into question the contribution of the algorithm itself, which we have declared here to be the focus of the study. It is therefore important to study different, but general constraint-handling techniques here. We will consider three of them.

- **Boundary repair (BR)** Upon evaluation of a solution $\mathbf{x}$, every $x_i < 0$ is changed to $x_i = 0$ and every $x_i > 1$ is changed to $x_i = 1$.
- **Constraint domination (CD)** (Deb 2000) The fitness of a solution is computed regardless of its feasibility. In addition, the amount of constraint violation is penalized quadratically. For CiaS, we use the sum over all variables of $(0 - x_i)^2$ if $x_i < 0$ and $(x_i - 1)^2$ if $x_i > 1$. With CD, the ranking of solutions is altered. When comparing two solutions, if both are infeasible, the solution with the smallest amount of constraint violation is preferred. If only one solution is infeasible, the solution that is feasible is preferred. Finally, if both solutions are feasible, the original ranking (i.e., based on fitness) is used.
- **Random repair (RR)** Upon evaluation of a solution $\mathbf{x}$, every $x_i < 0$ and every $x_i > 1$ is randomly reset, meaning that it is given a new value randomly uniformly drawn from [0; 1].

3 Initial landscape analysis

Although many approaches to solving the CiaS problem have appeared in the literature, relatively little is done on landscape analysis (from a BBO perspective) (Morgan and Gallagher 2014). To gain some insight into the type of landscapes that CiaS problems have, we produced pairwise variables heat maps for two CiaS instances: for $n_c = 5$ (Fig. 2) and $n_c = 10$ (Fig. 3). These heat maps were produced by taking a single solution, either the known optimal solution or a randomly generated solution, and subsequently varying two of the variables, keeping all others fixed. The observed function values are color-coded in a heat map, with black being the worst function value and bright yellow the best. In case, the two variables are the same (i.e., the diagonal in Fig. 2), the heat map becomes one-dimensional, in which case we replace the heat map with a simple line graph. In each case, the location of the selected solution itself is also shown, using a black dot.

The heat maps illustrate how the main challenges in CiaS problems are different from typical real-valued benchmark problems that are often variants of sums of quadratic functions, potentially superimposed with many local optima. The CiaS problem has many ridges and valleys along which the fitness is constant. This seems to bear some similarity to the Michalewicz function, which is also not often found in benchmark sets, but has recently been demonstrated to be a hard problem for typical Gaussian EDAs (Bosman et al. 2013). The Michalewicz function is not efficiently solved by exponentially increasing the population size. However, it can still be efficiently solved by considering each problem dimension separately, which is not the case for CiaS where there are clearly strong, often nonlinear dependencies between problem variables. This can for instance be seen in Fig. 2 for the combination of variables x_0 and x_1 in both the random and the optimal solution that shows an uneven distribution of circular shapes in the heat map that causes the optimal value of one variable to be (very) different for different values of the other variable.

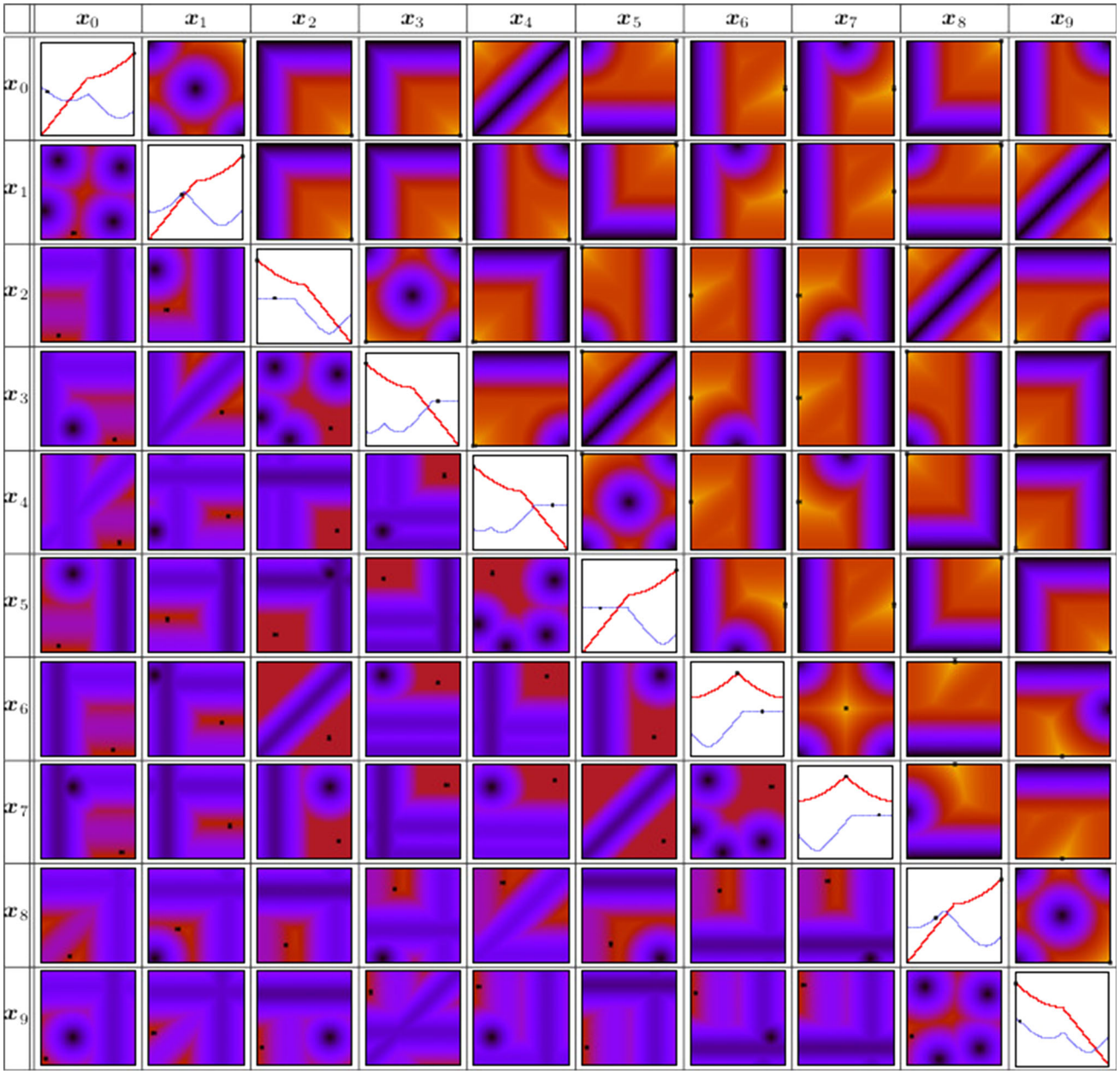

Fig. 2 Heat maps (low values: *black*, high values: *yellow*) for the $n_c = 5$ CiaS problem instance for all variable pairs in case of the optimum (*top right* of matrix) and in case of a random solution (*bottom left* of matrix). *Black dots* show solution locations. *Diagonal* shows one-dimensional landscape plots (*red*: optimum solution, *blue*: random solution)

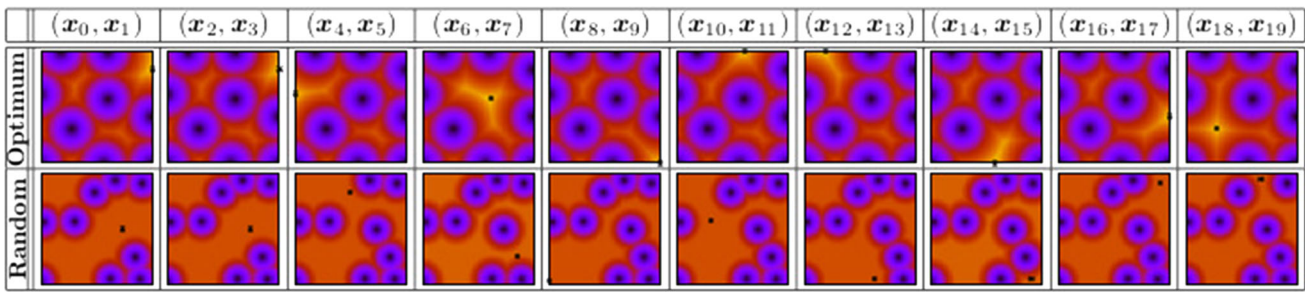

Fig. 3 Heat maps for $n_c = 10$ CiaS problem instance for pairs of variables that form circles only. Further details are similar to those in Fig. 2

The heat maps for pairs of variables that form circles also clearly show that a component of the underlying structure is the Voronoi diagram of the points, which forms a set of ridges along which the fitness is constant. Furthermore, there are many symmetries as in any given solution any pair of variables that form a circle can be interchanged with any other pair of variables that form a circle. Solutions can thus be rotated, flipped and mirrored at will. Another interesting component to CiaS is one that cannot be seen from the heat maps: the hard constraint that every point must lie in the unit box. Dealing with constraints can be done in many ways and great care must be taken. Repair mechanisms for instance have the advantage that solutions always remain feasible, but where solutions end up after a repair operation can be of vital importance to the process of model building. Although in future work more advanced landscape analyses would be interesting to perform, we can already conclude from this rudimentary analysis that the CiaS problem is highly nonlinear, has many isolated local optima, ridges and strong variable interactions as well as hard constraints.

4 Influence of population size

A key parameter of any EA is the population size. We vary the population size on the $n_c = 10$ CiaS instance and run all algorithms until convergence without a limit on the number of evaluations, but with a fitness variance tolerance of 10^{-30}. As a basis of comparison, we consider the performance ratio, defined as $f(\mathbf{x})/f(\mathbf{x}^*)$ where $f(\mathbf{x}^*)$ is the known optimal value. We also consider the probability of success, where we define a success as reaching 99.99 % of the optimal value (equivalent to a performance ratio of 1.00010001).

Figure 4 shows the average and the interdecile range (i.e., 10th–90th percentiles) of the results using error bars, computed over 100 runs. It is quite clear that the additions for enhanced convergence in AMaLGaM and CMA-ES also prove useful for solving CiaS problems. MaLGaM requires far larger population sizes to get a low performance ratio (Fig. 4, center row), but even then, the performance is well below the 99.99 % success criterion (i.e., is not shown in Fig. 4, top row since all values were zero). The comparison between CMA-ES and AMaLGaM is far less outspoken. Ignoring for now, the number of evaluations (discussed further in Sect. 6), the probability of success for CMA-ES is much smaller than it is for iAMaLGaM when BR is used, and similar when CD is used. In case of RR, the performance ratio of CMA-ES is near that of MaLGaM.

Figure 4 also includes results for the use of the univariate factorization compared to full multivariate Gaussian models. For the univariate factorization, the covariance matrix is diagonal; i.e., only the variances are estimated. As a result, model building and sampling new solutions can be done much faster. On CiaS, a reasonable performance ratio is obtainable using a univariate factorization. However, it is clear that using a univariate factorization is inferior to using the full covariance matrix if the true optimum is desired to be found (Fig. 4, bottom row). Thus, the results on CiaS problems suggest that it is important to consider the processing of dependencies, also known as linkages, between problem variables and how we can best exploit them. This has been a major focus of the EDA literature, but there have been few empirical validations of this reported on anything but artificial benchmark problems.

These results confirm existing results obtained using MaLGaM regarding the sensitivity of parameter settings (Gallagher 2012). The literature guidelines for population sizes are 373 for AMaLGaM, 44 for iAMaLGaM and 12 for CMA-ES. Whereas for AMaLGaM these values are quite good (relatively high probability of success and relatively low performance ratio), they are not optimal for iAMaLGaM and nowhere near good for CMA-ES.

Regarding constraint-handling techniques, it is clear that this choice has a dramatic influence on the performance results. Using RR gives the worst results by far among all considered techniques. This underlines the importance of being aware of the potential influence of a constraint-handling technique, in addition to issues that are typically given much more attention (e.g., how to best build the Gaussian model). With reflection on some (non-black-box) specific knowledge of the problem, the reason for the worse behavior with RR is likely to be that in an optimal solution for CiaS always at least one point lies on a boundary of the feasible solution space. When approaching a boundary, solutions will be sampled on either side, but those on the infeasible side will be changed a lot and will typically end up far away from the boundary, making converging to boundaries hard.

A second, interesting effect that results from the use of constraint-handling techniques is that the probability of success does not continue to increase as the population size increases. In fact, for CiaS, increasing the population size eventually leads to a decrease in success. The average performance ratio stagnates and in some cases even becomes worse. Clearly, a source of deception exists for the considered algorithms. As the population size increases, more infeasible solutions are generated. The proportion of infeasible solutions will also likely increase as the variance tends to increase with a larger population size, at least up to a certain point. The constraint-handling techniques then increasingly interfere with converging toward a solution as many solutions are repaired and relocated near local optima, toward which in the end with high probability the algorithm converges. Given that the performance ratio does not degrade

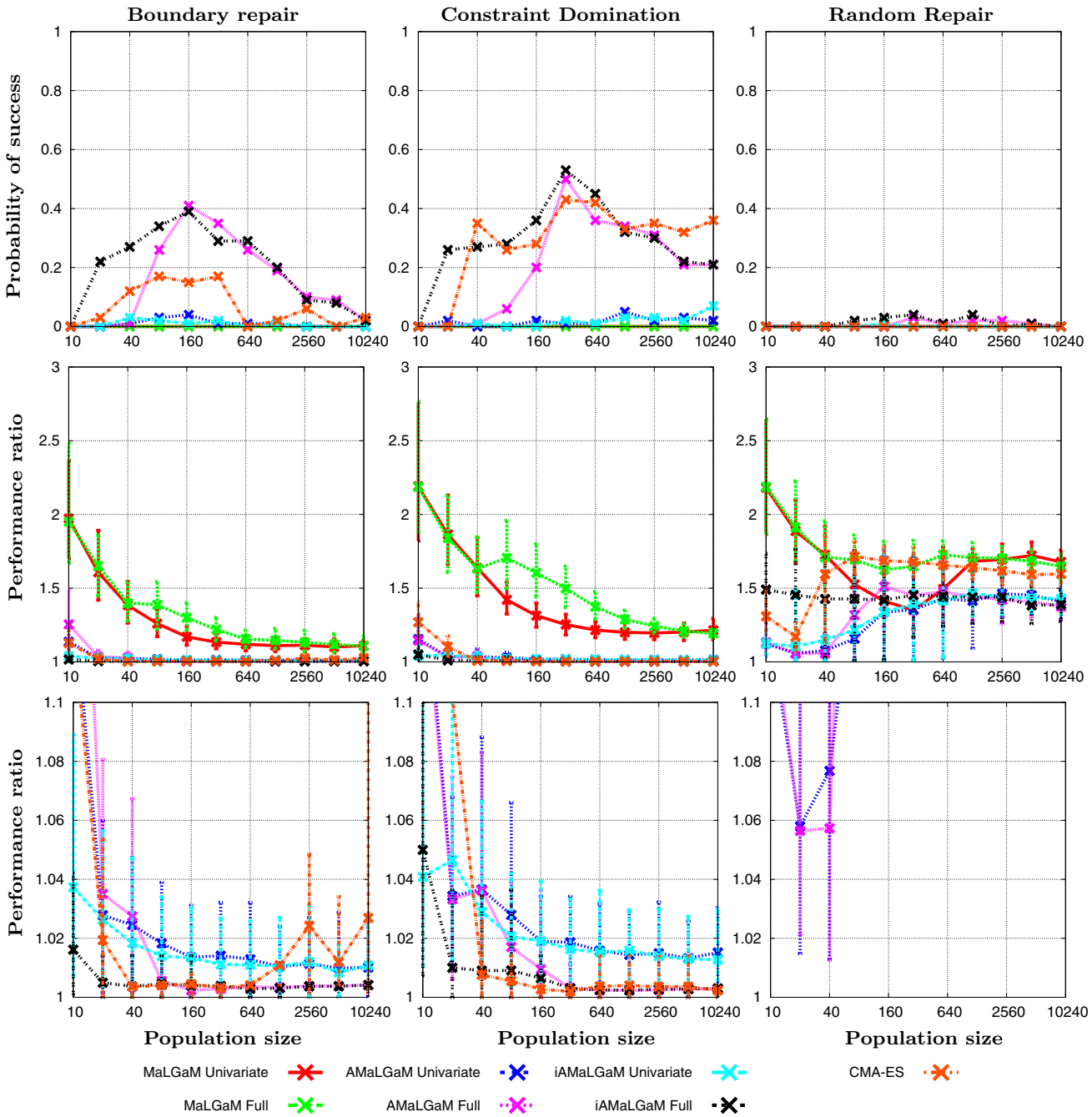

Fig. 4 Results of all tested EAs with typical literature-reported parameter settings except for population size, which is varied here (*horizontal axis* in every graph) (different *graph lines*), for every constraint-handling method (different *columns*) on the $n_c = 10$ CiaS instance, averaged over 100 runs. *Bars* show interdecile range. *Top row* probability of success, defined as reaching 99.99 % of the known optimal function value. *Center row* performance ratio upon convergence caused by population fitness variance dropping below 10^{-30}. *Bottom row* the same as the center row, but zoomed into a smaller range for the performance ratio (*vertical axis*)

much, these local optima are of good quality in the case of BR. This is to be expected since good solutions will have points on the boundary. Figure 5 shows a convergence graph for iAMaLGaM with two different settings for the population size. For the smaller population size setting, 38 runs out of 100 were successful, whereas for the larger population size setting, only 6 runs out of 100 were successful. This confirms that there are one or more local attractors to which iAMaLGaM tends to converge (if BR is used). Given how RR works, it is unlikely that a solution will be repaired and relocated near a local optimum of high quality if RR is used. Among the considered techniques, CD is the least disruptive.

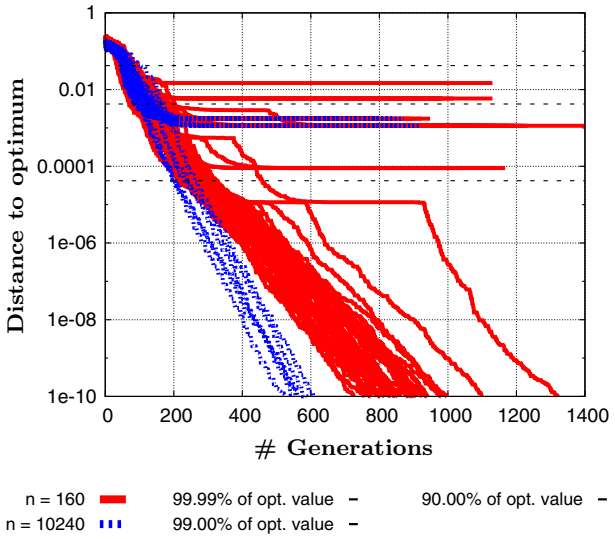

Fig. 5 Convergence graph showing 100 runs of iAMaLGaM with typical literature-reported parameter settings except for population size, which is varied here, on the $n_c = 10$ CiaS instance. Successful runs show lines reaching a distance to the optimum of $1e-10$. Runs that level off at a higher distance, have converged to a local optimum

This is to be expected since this technique does not relocate solutions. However, the performance of iAMaLGaM-Full shows that even with CD the chance of not ending up at the global optimum increases as the population size becomes overly large. Concluding, the choice of constraint-handling technique is of great importance and care should be taken when drawing conclusions about an algorithms' performance without considering which constraint-handling technique is used.

5 Influence of multi-starts and restarts

Increasing the population size is a commonly used approach to increasing an EA's resources and, often considered correspondingly, its optimization power. However, as shown in the previous section, due to the constraints of the CiaS problem, increasing the population size above some point is not helpful and can even be detrimental. Another way to increase the resources of an EA (also of non-population-based optimization algorithms) is a random restart procedure, i.e., run the EA multiple times and report the best result found over all runs. In AMaLGaM (Bosman et al. 2013), an extension of this procedure was proposed that uses an increasing number of multiple restarts that are not fully independent. In the case of k restarts, kn solutions are generated randomly and then partitioned into k clusters. These clusters then form the initial populations for k otherwise independent runs. If the solution space is globally deceptive or hard, this approach allows a natural subdivision of the space and potentially a faster increase in the probability of success than randomly restarting. A related technique found in EAs is niching (Mahfoud 2000; Yu and Suganthan 2010). We consider here whether such an approach is useful for the CiaS problem. Since the procedure was proposed only for AMaLGaM, we tested it only with the AMaLGaM algorithm variants. For AMaLGaM and iAMaLGaM, their respective guideline parameter settings were used. For MaLGaM, 10 times the population sizing guideline for AMaLGaM was used, i.e., $n = 3730$.

The results are shown in Fig. 6. The light dotted lines correspond to the probability of success if the number of populations on the horizontal axis was independent random restarts, i.e., $1 - (1 - p_s)^k$ where p_s would be the probability of success for a single run, and k is the number of restarts. These results should be compared with those in Fig. 4 showing the effects of population size. Clearly, restarting is a far more effective approach to increasing the probability of success than increasing the population size. This is good news from a viewpoint of parallel computation without minutely parallelizing the algorithm but running a single algorithm in parallel multiple times. From these results, there seems little additional advantage to using the clustering approach described above to partition the restarts when solving CiaS, because the results closely follow the lines that would have been achieved with pure random restarts. If the initial partitioning would be additionally helpful, the observed probabilities of success for the actual algorithm experiments should grow faster than the light dotted lines.

Other results are in line with observations made in Sect. 4. In particular, the RR constraint-handling technique leads to the worst results, the enhancements in AMaLGaM help substantially and the univariate factorization substantially diminishes the probability of success. The main reason for AMaLGaM outperforming iAMaLGaM in these experiments is that at the guideline population size AMaLGaM (initially) has a larger probability of success.

6 Influence of problem size

In this Section, we consider the influence that the size of the problem has on the performance of the considered algorithms from two different perspectives. First, we consider the case where the maximum computational budget is fixed. Second, we consider the case where the desired approximation quality is fixed.

6.1 On a 10,000,000 evaluations budget

We first consider the influence of the problem size given a fixed budget of 10^7 function evaluations. We consider problem instances $2 \leq n_c \leq 30$, for which the globally optimal values are known. Because we know from the results obtained

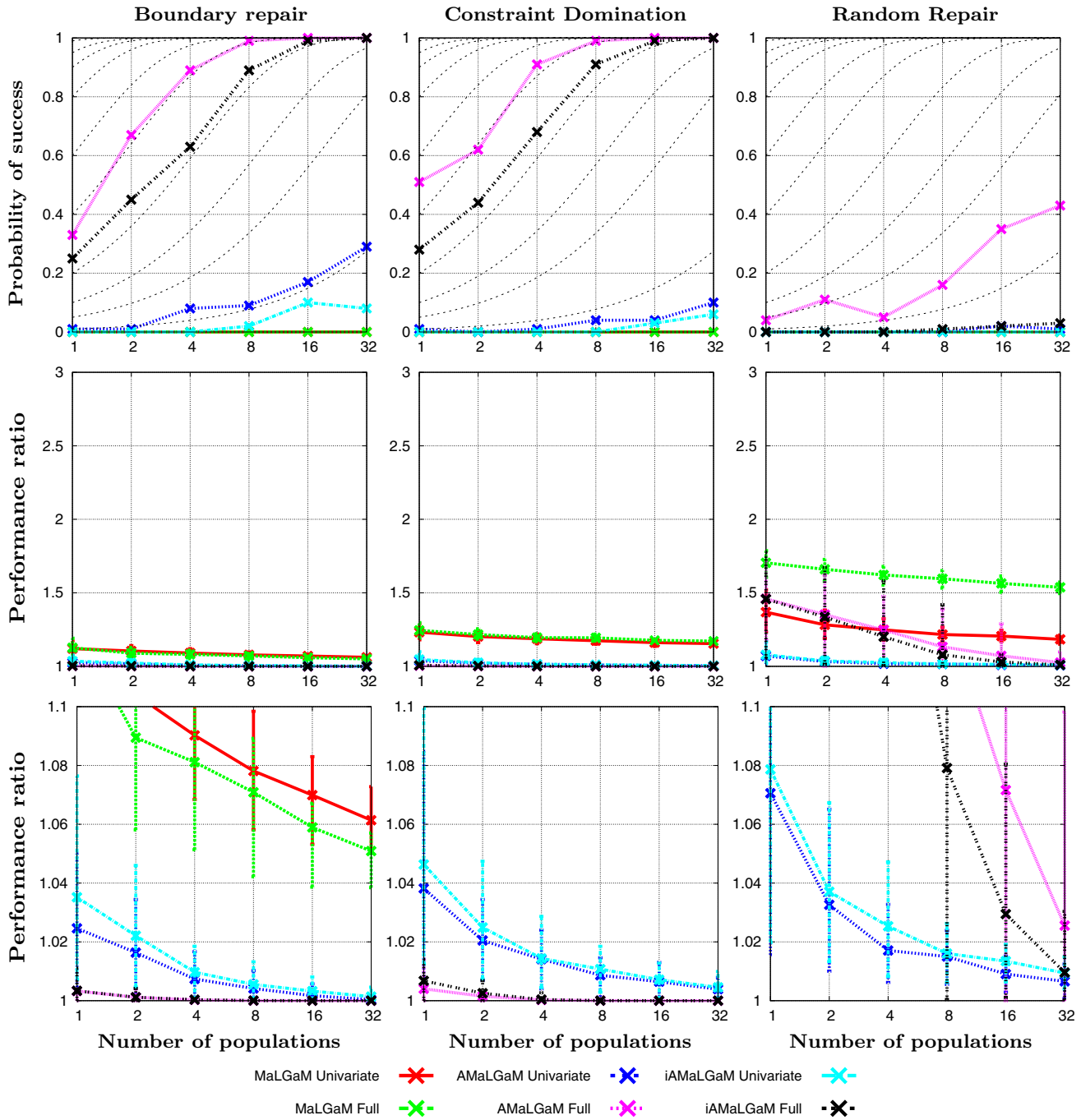

Fig. 6 Results of all tested EAs with typical literature-reported parameter settings (different *graph lines*) for every constraint-handling method (different *columns*) on the $n_c = 10$ CiaS instance for different numbers of populations that are executed in parallel, but are initialized so as to subdivide the search space (*horizontal axis* in every graph), averaged over 100 runs. *Bars* show interdecile range. *Top row* probability of success, defined as reaching 99.99 % of the known optimal function value. *Dotted lines* indicate expected progression of the probability of success with completely independent multiple restarts, starting from different probability of success values for a single start. *Center row* performance ratio upon convergence caused by population fitness variance dropping below 10^{-30}. *Bottom row* the same as the center row, but zoomed into a smaller range for the performance ratio (*vertical axis*)

so far that allowing (multi-)restarts is highly beneficial, we will consider two approaches to using the algorithms: using typical parameter settings and using parameter-less population management schemes that employ restarts.

6.1.1 Typical parameter settings

For all algorithms, we used their respective suggested algorithmic parameter settings taken from the literature. How-

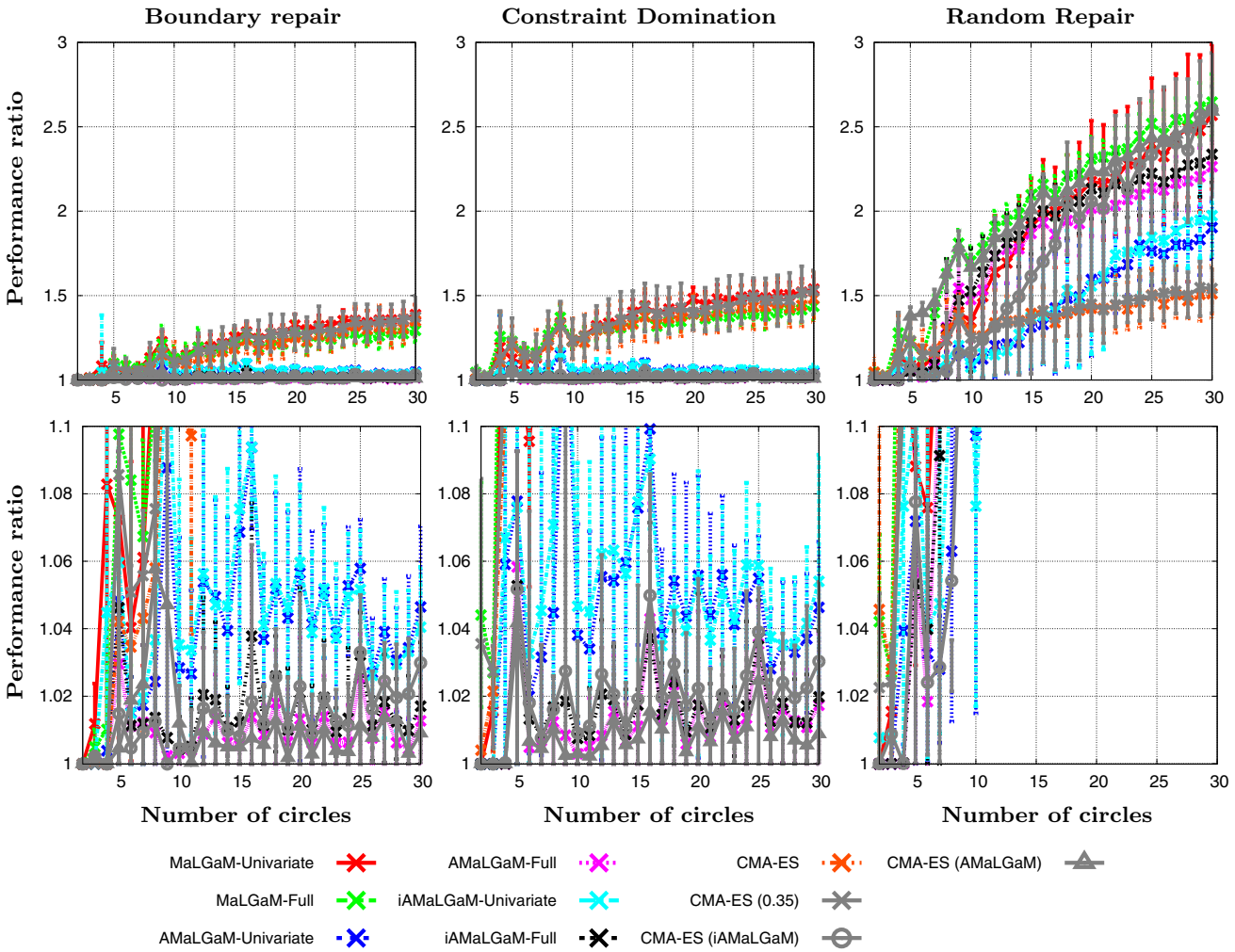

Fig. 7 Results of all tested EAs with typical literature-reported parameter settings (different *graph lines*) for every constraint-handling method (different *columns*) for different problem sizes (*horizontal axis* in every graph), averaged over 100 runs. *Bars* show interdecile range. *Top row* performance ratio upon convergence caused by population fitness variance dropping below 10^{-30}. *Bottom row* the same as the *center row*, but zoomed into a smaller range for the performance ratio (*vertical axis*)

ever, because in the previous sections we observed that the population size guideline for CMA-ES was too small, we considered running CMA-ES with the same settings as AMaLGaM and iAMaLGaM for both the truncation percentile and the population size.

The results in Fig. 7 confirm most of the observations that were summarized at the end of Sect. 5. In addition, however, we now see that the performance of CMA-ES drastically changes if different algorithmic parameters are chosen. To support some of our observations from the figures, we used statistical hypothesis testing. In particular, we have used the Mann–Whitney–Wilcoxon statistical hypothesis test for equality of medians with $p < 0.05$ to see whether the final result obtained by one EA is statistically different from that of another EA.

If the selection parameter alone is changed into that of AMaLGaM (i.e., $\tau = 0.35$ instead of $\tau = 0.5$), the results become only a little worse on average, but this is not found to be statistically significant. However, a huge, statistically significant, improvement is obtained if the population size is additionally changed into that of AMaLGaM or that of iAMaLGaM. In case of the latter, the results are a little worse on average than that of AMaLGaM and iAMaLGaM. This difference is found to be statistically significant. This is again indicative that the additional enhancements made in CMA-ES over those already present in AMaLGaM are not the most important. However, if the population size is further increased to that of AMaLGaM, the results become slightly better than that of AMaLGaM and iAMaLGaM. This difference is found to be statistically significant in comparison to iAMaLGaM, but not to AMaLGaM. These observations hold for the two least disruptive constraint-handling techniques. In case of RR, the results always become worse for CMA-ES. This is likely because the solutions no longer abide by a

key assumption under which CMA-ES performs highly efficiently, i.e., that the function being optimized is a quadratic surface. This is a known issue (Hansen 2006). Choosing the right population size can, however, still make a big difference. Note that although the best results in these graphs are now for CMA-ES, this may not necessarily remain the case if other population sizes are tested for both AMaLGaM and CMA-ES.

6.1.2 Parameter-less population management schemes

In addition to testing the algorithms using guideline settings for algorithmic parameters taken from the literature, we also tested versions of the algorithms that incorporate restart mechanisms. Ultimately, it is fairest to do comparisons on the basis of complete mechanisms, wherein all algorithmic parameters are specified and a restart mechanism is incorporated to make algorithms self-contained so that no further parameter tweaking can be done to make an algorithm appear better than its non-tweaked competitors. We can then speak of a specific parameterized instance of the algorithm. For AMaLGaM and iAMaLGaM, such a parameter-less population management (PLPM) scheme has been proposed in which first a single run is performed with base guideline parameter settings. Upon convergence, the population size is increased or the number of parallel runs started from a clustered initialization is increased. In even restarts the number of parallel runs is exponentially increased. In odd restarts a single population is used but the population size is exponentially increased. More details may be found in the literature (see, e.g., Bosman et al. 2013). Moreover, source code is available from the website of the first author.[1] For CMA-ES, we use a restart strategy that was the proposed default until recently: increase the population size exponentially upon convergence (Hansen 2006). We note that the currently best-known approach for running CMA-ES is to use a strategy that uses different parameter settings in different populations, known as the bi-pop CMA-ES (Hansen et al. 2010).

For MaLGaM, a restart strategy has not been previously proposed. Although we could use the same one as in AMaLGaM, the results presented here indicate that the basic Gaussian EDA is far inferior for solving CiaS instances and we therefore disregarded it further. The results (Fig. 8) show quite clearly again the impact of using the different constraint-handling techniques. Interestingly, it can be seen that with the CD approach to constraint handling, the negative effect of increasing the population size on the probability of success as pointed out in Sect. 4, becomes much larger than with the BR approach or even the RR approach as the problem size increases. This can specifically be seen from the fact that the PLPM variant of CMA-ES that increases its population size exponentially starts to perform worse than all other algorithms for $n_c \geq 12$. Moreover, this difference becomes very distinct as n_c increases. Using the BR approach the results are much better, although still not as good as for the PLPM version of AMaLGaM, which appears to remain the most robust algorithm that we have tested. Moreover, the results are much better than those reported in the previous study on solving CiaS with EDAs (Gallagher 2012), suggesting that we can, for the first time, look into the scalability of actually solving CiaS instances using a black-box approach, which is what we turn to next.

6.2 Scalability in case of 99.99 % quality

Because it has become clear that the RR constraint-handling technique is by far the worst, we only consider the BR and CD techniques. We use the success criterion of reaching 99.99 % of the known optimal quality. To reach this criterion repeatedly and reliably up to $n_c = 30$ we only found the PLPM versions of AMaLGaM to be up to the task, so we report only on those. The results are shown in Fig. 9 on a log–log scale. The final results appear to scale polynomially, but this is not expected to extend beyond the tested range of problem sizes as most packing problems are NP-hard. Our results are limited to problem sizes up to $\ell = 60$, but it should be noted that this is still larger than usually studied in black-box optimization experiments (e.g., in the BBOB benchmark competitions the maximum considered dimensionality is 40). iAMaLGaM has the upper hand for smaller dimensionalities and this holds true longer when the BR technique is used. Both with BR and CD, AMaLGaM showcases the more robust behavior and better scalability. Estimating polynomial regression lines on the averages and in terms of ℓ instead of n_c, we find that with the CD technique scalability is slightly better with $0.76\ell^{4.3}$ for AMaLGaM and $0.05\ell^{5.2}$ for iAMaLGaM in case of BR and $1.2\ell^{3.7}$ for AMaLGaM and $0.19\ell^{5.4}$ for iAMaLGaM in case of CD. We note again that these polynomial fits likely do not remain representative well out the tested range of $2 \leq n_c \leq 30$. Moreover, even within this range, these numbers are substantial.

7 Discussion and conclusions

In this article, we considered the real-valued optimization problem of sizing and packing n_c equally sized circles in a square (CiaS) from a black-box optimization perspective. This problem is non-trivial to solve, yet can be efficiently evaluated and has many ties to real-world optimization problems. Moreover, we have pointed out various reasons why this problem is different from most (artificial) benchmarks

[1] http://www.cwi.nl/~bosman.

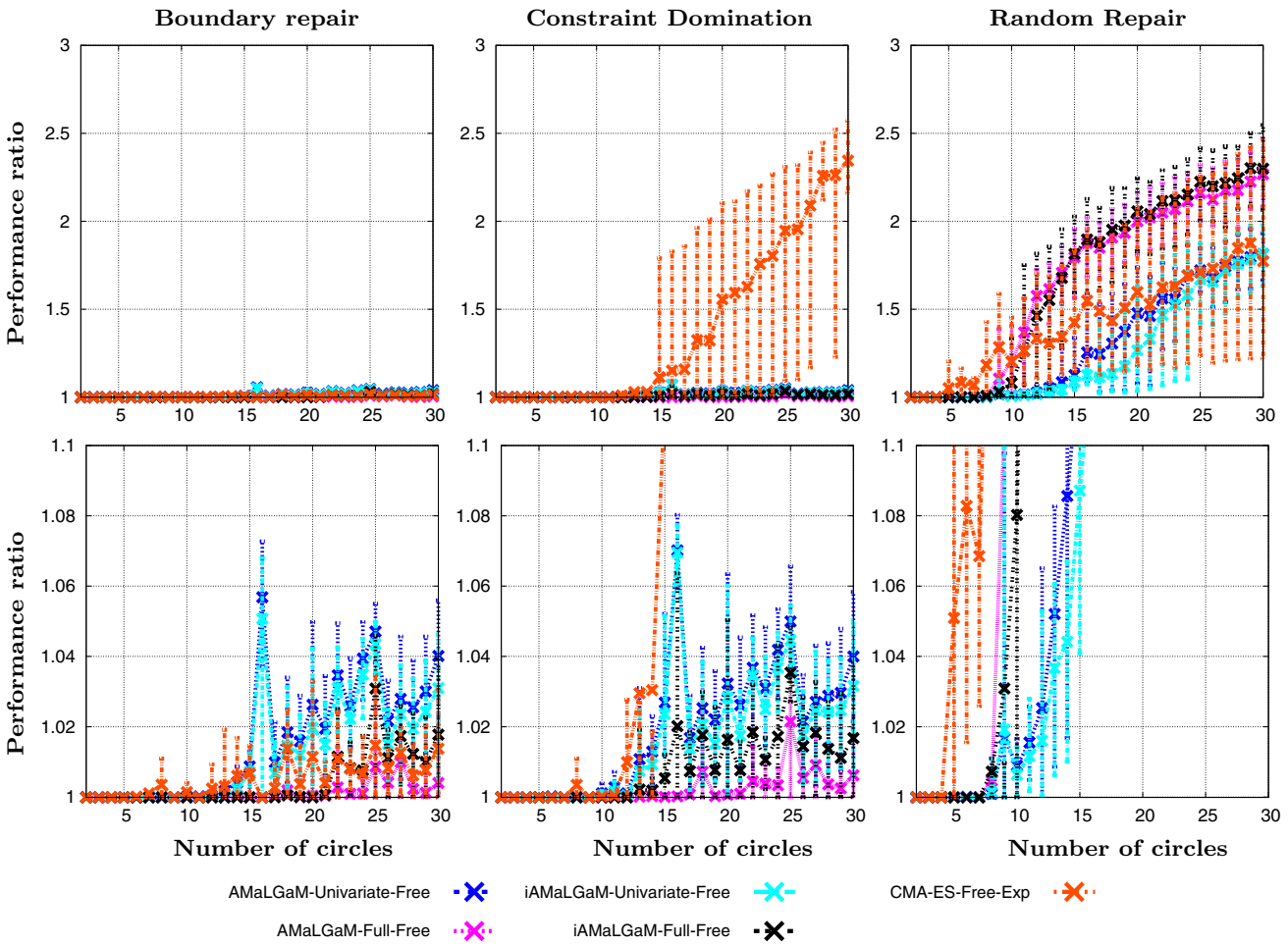

Fig. 8 Results of all tested EAs with typical literature-reported parameter settings (different *graph lines*) but with the population size parameter removed using parameter-less population management schemes, for every constraint-handling method (different *columns*) for different problem sizes (*horizontal axis* in every graph), averaged over 100 runs. *Bars* show interdecile range. *Top row* performance ratio upon convergence caused by population fitness variance dropping below 10^{-30}. *Bottom row* the same as the center row, but zoomed into a smaller range for the performance ratio (*vertical axis*)

considered in most studies on the design of real-valued evolutionary algorithms from a black-box optimization perspective. All these criteria combined lead us to propose that problems such as CiaS should be considered more frequently in the design of new (black-box) optimization algorithms as they account for different, yet important problem characteristics that demand other strengths of an optimization algorithm than the strengths that are currently being focused on most in the literature. Moreover, caution needs to be taken when generalizing the parameter settings of algorithm instances that have worked well on some benchmark problems to problems that may have significantly different characteristics. Ideally, a large, diverse set of benchmark problems (including real-world representative and artificial functions) will develop over time to provide more informative experimental results for black-box optimization algorithms. Our results indicate that most commonly accepted parameters for even some of the best-performing algorithms on other typical benchmarks do not automatically transfer well to the CiaS problem. We particularly looked at variants of Gaussian EDAs, including EMNA, AMaLGaM and CMA-ES. The type of problem structure that is exploited by these algorithms is only partially of importance when solving CiaS, which is the main reason for typical parameter settings not transferring well. The parameter settings for AMaLGaM were overall found to be the most robust, although they are unlikely to be optimal. In addition to a strong influence of parameter selection, we further found that the manner in which constraints are handled and having a restart strategy were the most impactful elements to the finally obtained solution quality. Overall, we therefore stress the importance of these issues when considering the comparison of different algorithms and of ensuring that comparisons are fair. It is easy to come to the wrong conclusions on a problem like this by virtue of straightforward usage of previously documented guidelines for algorithmic parameters.

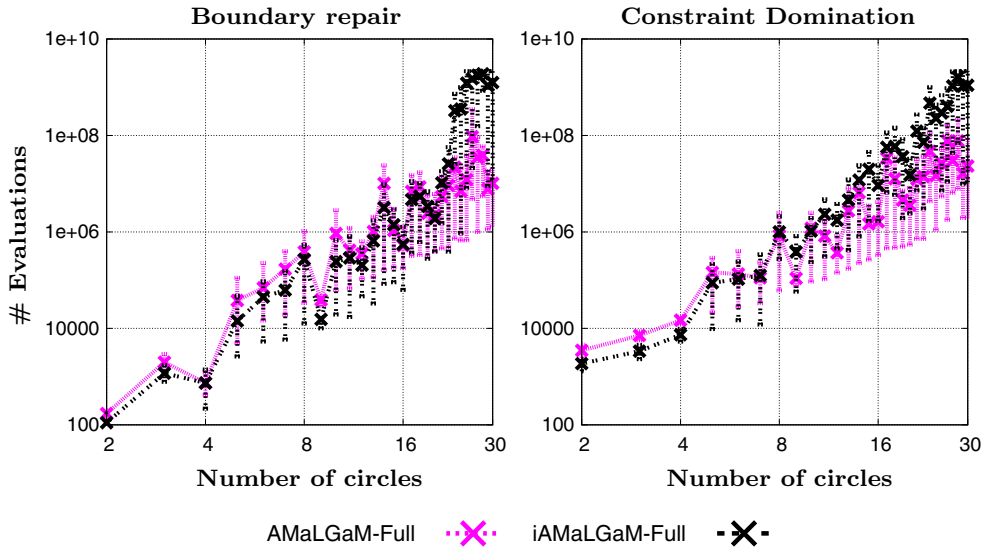

Fig. 9 Scalability in terms of the required number of function evaluations to reach a quality of 99.99 % on $2 \leq n_c \leq 30$ CiaS instances for AMaLGaM and iAMaLGaM using a full covariance matrix and a parameter-less population management scheme, for two constraint-handling methods (different *columns*) averaged over 100 runs. *Bars* show interdecile range

Given the relation of CiaS to real-world problems, it is important to study the key characteristics of CiaS more closely and to consider how particular structural features can be detected and be exploited automatically. To this end, we believe that the most important issues for further study are that the problem is constrained, that it has large, isolated local optima (instead of, e.g., many local optima superimposed on a quadratic surface) and that it has strong interactions between its variables. In addition to this, CiaS problems have high degree of symmetry in the structure of the solution space. Specifically, the order of circles in the solution vector can be permuted and the entire solution can be rotated or reflected, leading to $8n_c!$ equivalent solutions (Gallagher 2009). These and other types of symmetries occur in some types of optimization problems but to our knowledge have not been explored in an optimization context. This is beyond the scope of the work presented here but is an interesting direction for future work. In particular, the choices of solution representation and search operators in population-based metaheuristics may by strongly effected by these symmetries. It would furthermore be interesting to find and study other (real-world) problems that share similar properties.

We note that we have used only very general constraint-handling techniques that require the weakest of additional assumptions compared to the general problem formulation in Eq. 1. Because we have used a specific optimization problem in our study, it is natural to start to think about constraint-handling techniques and other extensions or adaptations of the EAs that we used in order to better solve the CiaS problem. Given the breadth of the literature on solving the CiaS problem, such improvements are most likely possible. For instance, one straightforward possibility is to rescale newly generated candidate solutions, to lie within the unit square. It should, however, be noted that such problem-specific changes are outside the scope of this article. We moreover would like to encourage other researchers to also consider solving the CiaS problem formulation in a similar fashion as we did in this article *without* introducing operators that exploit any additional problem-specific information, i.e., only using the knowledge that every variable must be in the range of [0, 1] and not assuming to know for instance which variables encode horizontal coordinates of points and which variables encode (associated) vertical coordinates.

In spite of the sensitivity to algorithmic parameter selection and constraint handling, we have shown that contrary to basic Gaussian EDAs, AMaLGaM and CMA-ES are capable of solving CiaS to optimality, underlining that the enhancements in AMaLGaM and CMA-ES over basic Gaussian EDAs are of importance also for solving CiaS. Furthermore, our results show that covariance modeling is important in order to locate the global optimum efficiently. Using univariately factorized Gaussian distributions allows obtaining good solutions quickly, but obtaining near-optimal solutions requires substantially more function evaluations. This number likely results in a larger additional computation time than building more complex models does. In future work, we shall consider this trade-off in computation time more closely.

Compliance with ethical standards

Conflict of interest Peter A. N. Bosman, Marcus Gallagher declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Addis B, Locatelli M, Schoen F (2008) Disk packing in a square: a new global optimization approach. INFORMS J Comput 20(4):516–524

Bosman P, Grahl J, Thierens D (2013) Benchmarking parameter-free AMaLGaM on functions with and without noise. Evolut Comput 21(3):445–469

Bosman PAN, Thierens D (2000) Expanding from discrete to continuous estimation of distribution algorithms: the IDEA. In: Schoenauer M et al (eds) Parallel problem solving from nature—PPSN VI. Springer, Berlin, pp 767–776

Castillo I, Kampas FJ, Pintér JD (2008) Solving circle packing problems by global optimization: numerical results and industrial applications. Eur J Oper Res 191:786–802

Coello CAC (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Comput Methods Appl Mech Eng 191(11–12):12451287

Deb K (2000) An efficient constraint handling method for genetic algorithms. Comput Methods Appl Mech Eng 186(2–4):311–338

Gallagher M (2009) Investigating circles in a square packing problems as a realistic benchmark for continuous metaheuristic optimization algorithms. In: MIC 2009: the VIII metaheuristics international conference, Hamburg, Germany

Gallagher M (2012) Beware the parameters: estimation of distribution algorithms applied to circles in a square packing. In: Coello CAC et al (eds) Parallel problem solving from nature—PPSN XII. Springer, Berlin, pp 478–487

Grosso A, Jamali A, Locatelli M, Schoen F (2010) Solving the problem of packing equal and unequal circles in a circular container. J Glob Optim 47(1):63–81

Hansen N (2006) The CMA evolution strategy: a comparing review. In: Lozano JA, Larrañaga P, Inza I, Bengoetxea E (eds) Towards a new evolutionary computation. Advances in estimation of distribution algorithms. Springer, Berlin, pp 75–102

Hansen N, Auger A, Ros R, Finck S, Pošík P (2010) Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In: Branke J et al (eds) Proceedings of the genetic and evolutionary computation conference—GECCO-2010. ACM Press, New York, pp 1689–1696

Larrañaga P, Lozano JA, Bengoetxea E (2001) Estimation of distribution algorithm based on multivariate normal and Gaussian networks. Technical Report KZZA-IK-1-01. University of the Basque Country, Department of Computer Science and Artificial Intelligence

Larrañaga P, Lozan JA (2001) Estimation of distribution algorithms. A new tool for evolutionary computation. Kluwer Academic, London

Mahfoud S (2000) Niching methods. In: Bäck DFT, Michalewicz Z (eds) Evolutionary computation 2–advanced algorithms and operations, Chap 13. IOP Publishing Ltd, Bristol, pp 87–92

Morgan R, Gallagher M (2014) Fitness landscape analysis of circles in a square packing problems. In: Dick G et al (eds) Simulated evolution and learning—SEAL-2014. Springer, Berlin, pp 455–466

Pelikan M, Sastry K, Cantú-Paz E (2006) Scalable optimization via probabilistic modeling: from algorithms to applications. Springer, Berlin

Specht E (2013) http://www.packomania.com/

Szabó PG, Markót MC, Csendes T (2005) Global optimization in geometry—circle packing into the square. In: Audet P, Hansen P, Savard P (eds) Essays and surveys in global optimization. Springer, US

Whitley D, Mathias K, Rana S, Dzubera J (1996) Evaluating evolutionary algorithms. Artif Intell 85(1–2):245–276

Yu E, Suganthan P (2010) Ensemble of niching algorithms. Inf Sci 180(15):2815–2833