An Analysis of Dimensionality Reduction Techniques for Visualizing Evolution

Andrea De Lorenzo
DIA - Università di Trieste
Trieste, Italy
andrea.delorenzo@units.it

Eric Medvet
DIA - Università di Trieste
Trieste, Italy
emedvet@units.it

Tea Tušar
Department of Intelligent Systems - Jožef Stefan Institute
Ljubljana, Slovenia
tea.tusar@ijs.si

Alberto Bartoli
DIA - Università di Trieste
Trieste, Italy
bartoli.alberto@units.it

ABSTRACT

We consider the problem of visualizing the population dynamics along an evolutionary run using a *dimensionality reduction technique* for mapping individuals from the original search space to a 2-D space. We quantitatively assess four of these techniques in terms of their ability to preserve useful information about (a) population movements and (b) exploration-exploitation trade-off. We propose two compact visualizations aimed at highlighting these two aspects of population dynamics and evaluate them qualitatively. The results are very promising as the proposed framework is indeed able to represent crucial properties of population dynamics in a way that is both highly informative and simple to understand.

CCS CONCEPTS

•Computing methodologies → *Genetic algorithms;* **•Mathematics of computing → Dimensionality reduction; •Human-centered computing → Visualization techniques;**

KEYWORDS

Exploration-Exploitation, Diversity, Ancestry, Evolutionary Algorithms, Visualization

ACM Reference format:
Andrea De Lorenzo, Eric Medvet, Tea Tušar, and Alberto Bartoli. 2019. An Analysis of Dimensionality Reduction Techniques for Visualizing Evolution. In *Proceedings of Genetic and Evolutionary Computation Conference Companion, Prague, Czech Republic, July 13–17, 2019 (GECCO '19 Companion),* 9 pages.
DOI: 10.1145/3319619.3326868

GECCO '19 Companion, Prague, Czech Republic

978-1-4503-6748-6/19/07…$15.00
DOI: 10.1145/3319619.3326868

1 INTRODUCTION AND RELATED WORKS

Visualizing the positions in the search space of individuals which are being evolved may be of great help to Evolutionary Computation (EC) practitioners and researchers. Different tasks related to the analysis and debugging of an evolutionary run may be performed more easily and intuitively if the reasoning is supported by a visualization: e.g., verifying if there is a diversity issue, analyzing the exploration-exploitation trade-off, checking if the population gets trapped in (multiple) local optima. Indeed, several visualizations with these aims have been proposed or used in the literature [1–4, 6, 7, 9, 12, 13, 15, 16, 19, 21, 25, 31], but they are mostly specific to single Evolutionary Algorithms (EAs) or limited to specific representations (with the exception of [6]).

One way to make a visualization method or tool applicable to a larger class of EAs is to first employ a *dimensionality reduction technique* to map individuals from high-dimensional search spaces (be it continuous or discrete) to a Cartesian plane and then visualize them in that plane. In the past, several methods have already used dimensionality reduction techniques for visualization purposes. For example, [16] employs simple projections to selected coordinates of the search space producing separate scatter plots and entire scatter plot matrices. Other approaches have utilized more sophisticated mappings on continuous search spaces, such as Principal Component Analysis (PCA) [4], Sammon mapping [11, 14, 24], Self-Organizing Maps (SOMs) [1], and, most recently, t-Distributed Stochastic Neighbor Embedding (t-SNE) [22]. For discrete spaces, in [27] the projection to a Cartesian plane is done by means of quadcodes, while [8] uses Sammon mapping on binary strings.

In this work, we propose a general framework which may, in principle, work with any dimensionality reduction technique and support different visualizations. We experimentally compare four of these techniques and different design options for the visualization. We evaluate the resulting visualization variants in terms of their ability to highlight two specific aspects of the population dynamics: (a) where and how the population moves and (b) if it is doing exploration or exploitation, two antagonistic cornerstones of search based optimization [5].

We applied each variant to a discrete optimization problem and a continuous optimization problem, each tunable in the number of optima and dimensionality of the search space. We quantitatively assessed the dimensionality reduction techniques in terms of the two aspects mentioned above and we qualitatively evaluated the

resulting plots along the same axes. The results are very promising, in particular when using Multidimensional Scaling (MDS) [30] for dimensionality reduction. The proposed framework is indeed able to represent crucial properties of population dynamics in a way that is both highly informative and simple to understand.

2 OVERVIEW OF OUR METHODOLOGY

Let S be the *search space* where individuals are defined. A *dimensionality reduction* function $m : S^h \to (\mathbb{R} \times \mathbb{R})^h$ takes a sequence of h points in S and outputs a sequence of h points in $\mathbb{R} \times \mathbb{R}$. Many functions of this form may be defined. Those considered in this work will be presented in the next section, along with their properties. An important characteristic shared by all these functions is that their definition depends on the full set of points to be mapped, i.e., given a point $s \in S$, its mapping in $\mathbb{R} \times \mathbb{R}$ depends on the *entire set* of points in S that have to be mapped.

Our methodology consists of two phases. In the first phase, we consider the full evolutionary run and map each individual that has existed during the evolution in the 2-D space (i.e., $\mathbb{R} \times \mathbb{R}$). In the second phase, we visualize the collected information in a compact (and possibly interactive) form.

In detail, the first phase is as follows. (i) We execute the full evolutionary run and keep track of all the individuals that have existed along with their corresponding fitness. (ii) Let P_i be the sequence of individuals (i.e., the *population*) at the i-th generation sorted based on some total order, usually the fitness of its individuals and let $i \in \{1, \dots, n_{\text{gen}}\}$, n_{gen} being the number of generations of the evolutionary run. We build P as the sequence of individuals obtained by concatenating all the sequences of individuals $P_1, \dots, P_{n_{\text{gen}}}$, sorted by increasing generation index. (iii) We map P to a sequence $Y = m(P) \in (\mathbb{R} \times \mathbb{R})^{|P|}$ of 2-D points applying the dimensionality reduction technique m. (iv) We split Y in n_{gen} sequences of 2-D points $Y_1, \dots, Y_{n_{\text{gen}}}$ such that $\forall i, |Y_i| = |P_i|$.

In the visualization phase, we build an interactive visualization of the evolution using the data in $Y_1, \dots, Y_{n_{\text{gen}}}$, the fitness of individuals, and, possibly, other ancillary information as, e.g., the ancestry. In each i-th frame corresponding to the i-th generation, points from Y_i are plotted on a Cartesian plane. The bounds of the Cartesian plane are the same for all the frames and are chosen by considering all the points in Y. Color is used to denote the fitness of the individuals. Before mapping them to a color, the fitness values are normalized according to the best and worst fitness values of all individuals in P. In addition to the current generation i, we can choose to show on the same i-th frame also some information on the previous generations. For example, we can visualize (a) points from Y_{i-1} (using a different color/size/transparency to discern them from the points from Y_i), (b) ancestry between the individuals from the previous generation and the current generation by connecting "parent points" from Y_{i-1} to the "offspring points" from Y_i, (c) points from Y_j for all $j < i$ (again using a different color/size/transparency to discern them from the points from Y_i), or (d) the mapped trajectory of the best individual in the population for generations $j, 1 \le j \le i$.

Consecutive frames can be viewed as a series of 2-D visualizations for different generations or can be stacked vertically to form a single 3-D visualization with the first generation on the bottom. In the latter case, points from previous generations are placed to their corresponding frames and there is no need to denote them differently to the ones from the i-th generation. By changing the current generation i, the user can simultaneously explore the i-th frame of the 2-D visualization as well as all stacked frames j, $1 \le j \le i$, in the 3-D visualization.

2.1 Dimensionality reduction

The rationale of dimensional reduction is to preserve the structures of a high-dimensional space in a lower dimensional space. Different techniques may differ in the kind of structures they attempt to preserve [17]. In these terms, dimensional reduction techniques can be roughly divided into two categories: (a) those that focus on preserving the structure concerning distant points and (b) those that seek to preserve the structure of nearby points. In our settings, we are in principle interested in both. On the one hand, we want to highlight distances between different (groups of) individuals which explore different regions of the search space. On the other hand, we are also interested in showing "slow drifts" of (groups of) individuals which converge towards an optimum.

We considered 4 techniques, evenly divided in the two categories: PCA [10] and MDS [30] in the former, t-Distributed t-SNE [17] and Uniform Manifold Approximation and Projection (UMAP) [18] in the latter. All but PCA require that a *dissimilarity function* $d : S \times S \to \mathbb{R}^+$ is defined in the search space, according to which the larger $d(s_1, s_2)$, the less similar the two individuals s_1 and s_2. On the other hand, PCA is applicable only if the original search space is numerical (i.e., if $S = \mathbb{R}^l$).

Other dimensionality techniques other than the four here considered could have been used (e.g., Sammon mapping [26], Isomap [29], low-dimensional Euclidean embedding (LDEE) [22])—we leave this investigation to future work.

3 TEST CASES

We wanted to verify the applicability of the methodology described in Section 2 to different representations, EAs, and problems. To this end, we considered two synthetic optimization problems with tunable representation and a simple tunable EA. The representation can be tuned in terms of whether the optimization problem is discrete or continuous and in terms of the dimensionality of the search space. The EA can be tuned in terms of the exploration-exploitation trade-off, by means of a diversity promotion scheme. The problems can be tuned also in the number of optima.

3.1 Bit string optimization

In this problem, an individual $\mathbf{x}$ is a bit string of l bits, i.e., $\mathbf{x} \in \{0, 1\}^l$. The fitness of an individual is defined as the distance d from the closest optimum among a predefined set $X^\star = \{\mathbf{x}_1^\star, \dots, \mathbf{x}_n^\star\}$ of n optima, where $\mathbf{x}_i^\star = (x_{i,1}^\star, \dots, x_{i,l}^\star)$ and $x_{i,j}^\star = 1$, if $l\frac{i-1}{n} < j \le l\frac{i}{n}$, and $x_{i,j}^\star = 0$, otherwise. Hence, the fitness $f(\mathbf{x}) \in \mathbb{R}^+$ of an individual $\mathbf{x}$ is $f(\mathbf{x}) = \min_{\mathbf{x}^\star \in X^\star} d(x, x^\star)$ and the goal is to minimize its value.

The parameters of this problem are the individual size l, the number n of optima, and the distance function d. It can be seen

that, when $n = 1$ and d is the Hamming distance, this problem essentially corresponds to the One Max problem.

3.2 Continuous optimization

This problem resembles the one of Section 3.1, but the search space is continuous. An individual $\mathbf{x}$ is a numerical vector of size l, i.e., $\mathbf{x} \in \mathbb{R}^l$. The fitness of an individual is defined as the distance d from the closest optimum among a predefined set $X^\star = \{\mathbf{x}_1^\star, \ldots, \mathbf{x}_n^\star\}$ of n optima. The optima of $X^\star$ lie on a l-dimensional hypersphere with radius $r = 1$ and centered in $\mathbf{0}$, i.e, for $\mathbf{x}_i^\star = (x_{i,1}^\star, \ldots, x_{i,l}^\star)$:

$$x_{i,j}^\star = \begin{cases} \cos\left(i\frac{2\pi}{n}\right)\left(\sin\left(i\frac{2\pi}{n}\right)\right)^{j-1} & \text{if } j \neq l \\ \left(\sin\left(i\frac{2\pi}{n}\right)\right)^{j-1} & \text{otherwise} \end{cases} \quad (1)$$

The fitness $f(\mathbf{x}) \in \mathbb{R}^+$ of an individual $\mathbf{x}$ is $f(\mathbf{x}) = \min_{\mathbf{x}^\star \in X^\star} d(x, x^\star)$ and the goal is to minimize its value.

The parameters of this problem are the individual size l, the number n of optima, and the distance function d.

3.3 Evolutionary Algorithm

We considered a simple EA with a non-overlapping generational model which evolves a fixed-size population of n_{pop} individuals for n_{gen} generations. Selection for reproduction is done using a tournament of size n_{tour}.

The population initialization procedure and the genetic operators are different for bit string and continuous optimization. The population initialization procedure consists, in both cases, in randomly generating n_{pop} individuals: in the bit string optimization problem, each gene $x_{i,j}$ of an individual $\mathbf{x}_i$ is randomly set to 0 or 1 with equal probability; in the continuous optimization problem, each gene $x_{i,j}$ is set by randomly sampling the interval $[-1, 1]$ with uniform probability (i.e., $x_{i,j} \sim U(-1, 1)$).

Concerning the genetic operators, we used the bit flip mutation (with probability equal to $\frac{1}{l}$) and the uniform crossover for the bit string optimization problem and the Gaussian mutation and line recombination [23] for the continuous optimization problem. The Gaussian mutation consists in adding a random noise sampled from $N(0, \sigma)$ to each element of the individual; the line recombination corresponds in taking a random point on the (extended) segment connecting the two parents, i.e., $\text{crossover}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1 + \lambda(\mathbf{x}_2 - \mathbf{x}_1)$, where $\lambda \sim U(-p_{\text{cross}}, 1+p_{\text{cross}})$. We used $\sigma = 0.01$ and $p_{\text{cross}} = 0.25$.

3.3.1 Diversity promotion. In order to better investigate to which degree the visualization tool captures the exploration-exploitation trade-off in an evolutionary run, we augmented the EA of Section 3.3 with a diversity promotion scheme. We adopted a fitness sharing scheme [28] in which the fitness of an individual is rescaled based on how densely populated is the region of the search space in which the individual is placed.

More precisely, the rescaled fitness $f'(\mathbf{x})$ of an individual $\mathbf{x}$ is defined as:

$$f'(\mathbf{x}) = \frac{f(\mathbf{x})}{1 + \sum_{\mathbf{x}' \in K} d(\mathbf{x}, \mathbf{x}')} \quad (2)$$

where $K \subset P_i$ is the set of the n_{NN} individuals (of the same i-th generation to which $\mathbf{x}$ belongs) closest to $\mathbf{x}$, i.e., its n_{NN} nearest neighbors. The larger the distance from the closest individuals, the larger the value of the denominator, and hence the lower $f'(\mathbf{x})$—recall that we are considering minimization problems.

It can be seen that, when $n_{\text{NN}} = 0$, no diversity promotion actually occurs, since K is empty and the fitness is not rescaled.

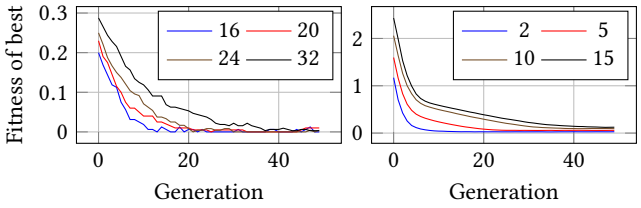

Figure 1: Fitness (mean across the 10 repetitions for each problem) of the best individual during the evolution for the two problems with different values of l and $n = 1$, $n_{\text{NN}} = 0$: bit string optimization on left and continuous optimization on right.

4 EXPERIMENTAL EVALUATION

We aimed at answering the following research questions concerning the proposed methodology for visualizing an evolution:

- RQ1 Which dimensionality reduction technique is the best in capturing the movements of the population in the search space?
- RQ2 Which dimensionality reduction technique is the best in capturing the exploration-exploitation trade-off?
- RQ3 What information can be meaningfully plotted? How?

To this end, we executed several evolutionary runs by varying some of the parameters of the problems and the EA. Namely, concerning the problems, we experimented with $l \in \{16, 20, 24, 32\}$ for the bit string optimization and with $l \in \{2, 5, 10, 15\}$ for the continuous optimization and varied n in $\{1, 2, 3, 4\}$. Concerning the EA, we varied n_{NN} in $\{0, 1, 2, 3, 4\}$ and set $n_{\text{pop}} = 50$, $n_{\text{gen}} = 50$, and $n_{\text{tour}} = 3$. As for the distances, we used the relative Hamming distance (i.e., the Hamming distance divided by length l of the bit string) for the bit string optimization problem and the Euclidean distance for the continuous optimization problem. In both cases, we used the same distance function for computing the fitness function, in the diversity promotion scheme, and for the dimensionality reduction, when applicable.

To provide an overview of the considered problems, Figure 1 shows the fitness of the best individual (mean across 10 independent evolutionary runs) during the evolution for different values of l and with $n = 1$, $n_{\text{NN}} = 0$ (i.e., one single optimum and no diversity promotion) for the two cases: bit string (left) and continuous (right) optimization.

Figure 1 highlights that the size of the individual l has an impact on the speed of convergence in both cases: the larger the individual, the slower the convergence. Moreover, the figure also suggests that in the bit string optimization the convergence is less smooth than in the continuous optimization problem, i.e., there are noticeable, yet small, variations in the fitness of the best individual during the entire evolution. We recall that our EA employs a non-overlapping generational model: monotony in the trend of the fitness of the best individual is hence not guaranteed.

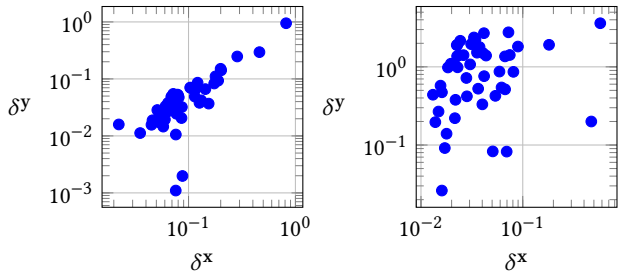

Figure 2: Scatter plot of the inter-generation best distances in the search space ($\delta^{\mathbf{x}}$) and the 2-D space ($\delta^{\mathbf{y}}$) for two runs on the continuous opt. problem: $n = 1$, $l = 10$, $n_{\text{NN}} = 1$ with MDS (left) and $n = 1$, $l = 5$, $n_{\text{NN}} = 1$ with t-SNE (right).

4.1 RQ1: Population movements

Ideally, by observing a plot of an evolution, it should be possible to see precisely where, in the search space, are the individuals at each generation during the evolution. In practice, some loss of that information occurs while performing the dimensionality reduction which, however, allows to obtain a compact and practical visualization. Measuring exactly how much of that information is lost and to which degree the loss affects the possibility of observing population movements is difficult and can vastly vary for different consumers of the visualization.

Rather than attempting to quantify the loss of information or computing the trustworthiness of a dimensionality reduction technique (see [32]), we verified if the movements in the search space along the evolution of one significant individual (the one with the best fitness) are well described in the 2-D space. To this end, we proceeded as follows. (i) We considered the sequence $\mathbf{x}^{*,1}, \ldots, \mathbf{x}^{*,n_{\text{gen}}}$ of n_{gen} best individuals (i.e., the best individual for each generation) and the corresponding sequence $\mathbf{y}^{*,1}, \ldots, \mathbf{y}^{*,n_{\text{gen}}}$ of 2-D points—the two sequences corresponding to the trajectories of the best individual in the search space and in the 2-D space. (ii) We obtained the inter-generation best distances, i.e., in each of the two spaces, the sequences $\Delta^{\mathbf{x}} = (\delta^{\mathbf{x}}_1, \ldots, \delta^{\mathbf{x}}_{n_{\text{gen}}-1})$ and $\Delta^{\mathbf{y}} = (\delta^{\mathbf{y}}_1, \ldots, \delta^{\mathbf{y}}_{n_{\text{gen}}-1})$ of the distance between the best individual at the i-th generation and the best individual at the $(i+1)$-th generation, with $\delta^{\mathbf{x}}_i = d(\mathbf{x}^{*,i+1}, \mathbf{x}^{*,i})$ and $\delta^{\mathbf{y}}_i = \|\mathbf{y}^{*,i+1} - \mathbf{y}^{*,i}\|$, where d is the distance function in the search space S. (iii) We measured the Pearson's correlation between $\Delta^{\mathbf{x}}$ and $\Delta^{\mathbf{y}}$: the closer correlation to one, the stronger the linear dependence of the inter-generation best distance in the 2-D space on that in the search space.

We repeated the procedure above for all the runs, problems, values of n_{NN}, and dimensionality reduction techniques.

Figure 2 shows, in the form of scatter plots, the collected inter-generation best distances for two different evolutionary runs with the continuous optimization problem: $n = 1$, $l = 10$, $n_{\text{NN}} = 1$ with MDS (left) and $n = 1$, $l = 5$, $n_{\text{NN}} = 1$ with t-SNE (right). Values are plotted using log scales.

It can be seen that in the first case (with MDS) there is a good correlation, the actual value being 0.96, between the inter-generation best distance in the search space and in the 2-D space: the trajectory of the best individual during the evolution is hence likely a good approximation of the actual trajectory in the search space. Differently, for the run with t-SNE, the correlation is lower (0.43).

Table 1 shows the value of the Pearson's correlation of inter-generation best distances for the problems with $n = 1$ and $n_{\text{NN}} = 0$, i.e., one single optimum and no diversity promotion. For brevity, we omit the other results: the qualitative findings are similar.

It can be seen from the table that MDS (and PCA, for the continuous problems) deliver much better results than t-SNE and UMAP, the difference being consistent over all the combinations. Moreover, by observing the standard deviation of the correlation across the independent runs for each combination, it can be seen that the variability of this index is lower for MDS and PCA than for t-SNE and UMAP.

4.2 RQ2: Exploration-exploitation trade-off

Exploration and exploitation are two antagonistic cornerstones of search-based optimization: in the former, new regions of the search space are visited, whereas in the latter regions where good solutions already exist are the focus of the search [5]. Although this definition is intuitive and well established, it does not corresponds to a single and widely accepted practical procedure for objectively determining if, or to which extent, an evolution is "doing" exploration or exploitation. It follows that it is not easy to assess the ability of our proposed visualization to reveal if an evolution is in the exploration or exploitation phase.

We here apply the operative definition of exploration/exploitation proposed in [5]. In brief, the tendency to exploration/exploitation is first measured at the level of the single birth: then, an aggregate measure is obtained for each generation by considering all the corresponding births. For determining if a birth corresponds to exploration or exploitation, Črepinšek et al. proposed in [5] to compare the distance of the new individual to the closest individual in the history of evolution up to the previous generation with a predefined threshold. We applied this procedure both in the search space and in the 2-D space and compare the results.

In detail, given an evolutionary run, we proceeded as follows. (i) At each birth of an individual $\mathbf{x}$ occurring at the i-th generation, we measured its (dis)Similarity[1] to the Closest Neighbor in the search space $\text{SCN}^{\mathbf{x}}(\mathbf{x}) = \min_{j<i} \min_{\mathbf{x}' \in P_j} d(\mathbf{x}, \mathbf{x}')$, where d is the distance function in the search space and P_j is the population at the j-th generation. We did the same in the 2-D space, obtaining $\text{SCN}^{\mathbf{y}}(\mathbf{y}) = \min_{j<i} \min_{\mathbf{y}' \in Y_j} \|\mathbf{y} - \mathbf{y}'\|$, where Y_j is the set of 2-D points corresponding to the population at the j-th generation. Note that for $\text{SCN}^{\mathbf{y}}$ we took the distance $\|\mathbf{y} - \mathbf{y}'\|$ to the closest neighbor $\mathbf{y}'$ in the 2-D space, rather than the distance to the 2-D point corresponding to the closest neighbor in the search space: the reason is that we wanted to verify if the exploration/exploitation trade-off which is perceived by observing the points in the 2-D space is a good proxy for the actual exploration/exploitation trade-off in the search space. (ii) We computed the medians $\overline{\text{SCN}}^{\mathbf{x}}$ and $\overline{\text{SCN}}^{\mathbf{y}}$ of the values of $\text{SCN}^{\mathbf{x}}$ and $\text{SCN}^{\mathbf{y}}$, respectively, for all the births of the evolutionary runs. (iii) For each i-th generation, we measured the *exploration rate* $\tau^{\mathbf{x}}_i$ in the search spaces as the rate of births of that generation for which $\text{SCN}^{\mathbf{x}} > \overline{\text{SCN}}^{\mathbf{x}}$: the larger $\tau^{\mathbf{x}}_i \in [0, 1]$,

[1] Although the name includes "similarity", the SCN index proposed by Črepinšek et al. in [5] is large for dissimilar individuals and hence measures dissimilarity.

	Bit string optimization				Continuous optimization			
	$l = 16$	$l = 20$	$l = 24$	$l = 32$	$l = 2$	$l = 5$	$l = 10$	$l = 15$
Dim. red.	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$
PCA					**1.00**±0.000	**0.98**±0.013	0.93±0.072	0.92±0.065
MDS	**0.95**±0.043	**0.91**±0.092	**0.95**±0.049	**0.93**±0.040	**1.00**±0.000	0.96±0.031	**0.94**±0.041	**0.93**±0.086
t-SNE	0.59±0.135	0.58±0.074	0.63±0.175	0.71±0.090	0.71±0.179	0.22±0.146	0.39±0.209	0.44±0.247
UMAP	0.35±0.108	0.52±0.207	0.59±0.187	0.56±0.166	0.65±0.181	0.21±0.112	0.43±0.198	0.43±0.168

Table 1: Pearson's correlation (mean μ and standard deviation σ across the 10 repetitions for each problem) of the inter-generation best distances measured in the search space and in the 2-D space, for the two problems and $n = 1$, $n_{\text{NN}} = 0$: the closer to 1, the better. For each problem and value of l, the best mean value is highlighted in bold.

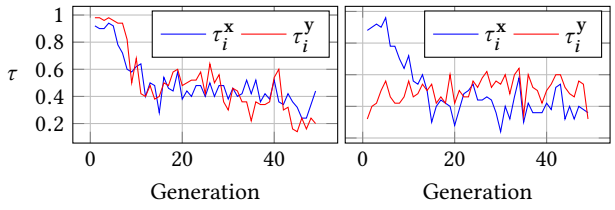

Figure 3: Exploration rate in the search space ($\tau_i^{\mathbf{x}}$) and in the 2-D space ($\tau_i^{\mathbf{y}}$), during the evolution, for two evolutionary runs on the continuous optimization: $l = 15$, $n = 1$, $n_{\text{NN}} = 1$ with MDS (left) and $l = 15$, $n = 2$, $n_{\text{NN}} = 2$ with t-SNE (right).

the more the evolution at the i-th generation is exploring, rather than exploiting. We repeated the same procedure in the 2-D space for obtaining $\tau_i^{\mathbf{y}}$. (iv) We measured the root-mean square error between the values of $\tau_i^{\mathbf{x}}$ and $\tau_i^{\mathbf{y}}$ during the evolutionary run:

$$\text{RMSE} = \sqrt{\frac{1}{n_{\text{gen}}} \sum_{i=1}^{i=n_{\text{gen}}} \left((\tau_i^{\mathbf{x}})^2 - (\tau_i^{\mathbf{y}})^2 \right)} \tag{3}$$

We remark that the choice of the median $\overline{\text{SCN}}^{\mathbf{x}}$ used for computing the exploration rate $\tau_i^{\mathbf{x}}$ is arbitrary. Choosing an appropriate threshold value for discriminating between exploration and exploitation is not trivial [5]: we think that, for the purpose of evaluating the visualization methodology here proposed, our choice is sound.

We repeated the procedure above for all the runs, problems, values of n_{NN}, and dimensionality reduction techniques.

Figure 3 shows the collected $\tau_i^{\mathbf{x}}$ and $\tau_i^{\mathbf{x}}$ values for two different evolutionary runs with the continuous optimization problem: $l = 15$, $n = 1$, $n_{\text{NN}} = 1$ with MDS (left) and $l = 15$, $n = 2$, $n_{\text{NN}} = 2$ with t-SNE (right).

By looking at the "actual" exploration rate $\tau_i^{\mathbf{x}}$ (blue line) in both cases, it can be seen that the index follows a reasonable trend: at the beginning of the evolution (i.e., within the $\approx$ 15-th generation) there is more exploration than exploitation ($\tau_i^{\mathbf{x}} > 0.5$); thereafter, the exploration rate decreases and stays steadily below 0.5, suggesting that the evolution is doing exploitation, rather than exploration. While being similar in the trend of $\tau_i^{\mathbf{x}}$, the two plots of Figure 3 differ in $\tau_i^{\mathbf{y}}$, i.e., in the exploration rate measured in the 2-D space (red line). With MDS, $\tau_i^{\mathbf{y}}$ appears to be a qualitatively good approximation of $\tau_i^{\mathbf{x}}$, capturing its general trend. In contrast, with t-SNE the values of $\tau_i^{\mathbf{y}}$ during the evolution do not highlight the presence of the two stages of the evolution (exploration with decreasing exploration rate and then exploitation).

Table 2 shows the results for the problems with $n = 1$ and $n_{\text{NN}} = 0$ (above) and with $n = 4$ and $n_{\text{NN}} = 4$ (below). We chose to include the results in two settings (one optimum, no diversity promotion and many optima with diversity promotion) to verify if the RMSE of the exploration rate varies when conditions related to the exploration/exploitation trade-off vary.

The figures in Table 2 show that, as for the Pearson's correlation of the inter-generation best distances, PCA and MDS deliver the better results. For bit string optimization MDS is always the best option among the dimensionality reduction techniques. For continuous optimization, MDS gives the lowest RMSE in 5 on 8 cases. The difference in RMSE are, however, less apparent than those observed for the Pearson's correlation of the inter-generation best distances (see Table 1). Interestingly, t-SNE looks effective, in terms of RMSE of the exploration rate, for large continuous optimization problems ($l = 15$). We think that further experimentation is needed to confirm or refute this hypothesis.

All in all, the experiments we conducted to answer RQ1 and RQ2 suggest that MDS is the dimensionality reduction technique which best addresses, among the 4 techniques we considered, the needs related to the visualization of evolutionary runs in a 2-D space.

4.3 RQ3: Visualization choices

We developed two interactive visualizations: the first shows one 2-D frame per generation and the second visualizes all generations up to the current one in a single 3-D plot. In both, the individuals from the current generation are represented as points whose position on the Cartesian plane is obtained by means of dimensionality reduction and whose color denotes their fitness. In addition, the third coordinate of the 3-D visualization encodes the generation number. The user can interact with the visualization by changing the current generation by means of a slider: the 2-D and 3-D plots can be placed side to side and controlled with a single slider. For 3-D plots, the user can also freely rotate the space for changing the point of view.

		Bit string optimization				Continuous optimization			
		$l = 16$	$l = 20$	$l = 24$	$l = 32$	$l = 2$	$l = 5$	$l = 10$	$l = 15$
	Dim. red.	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$
$n = 1$ $n_{\text{NN}} = 0$	PCA					**0.00**±0.000	0.47±0.160	0.57±0.152	0.61±0.148
	MDS	**0.43**±0.083	**0.29**±0.107	**0.23**±0.211	**0.55**±0.085	**0.00**±0.000	**0.45**±0.147	**0.56**±0.130	0.60±0.127
	t-SNE	0.49±0.139	0.46±0.122	0.47±0.125	0.58±0.115	0.50±0.124	0.56±0.129	0.57±0.123	**0.56**±0.113
	UMAP	0.54±0.116	0.53±0.109	0.56±0.114	0.64±0.118	0.55±0.132	0.63±0.125	0.66±0.119	0.66±0.116
$n = 4$ $n_{\text{NN}} = 4$	PCA					**0.00**±0.000	0.43±0.192	**0.49**±0.162	**0.54**±0.151
	MDS	**0.37**±0.108	**0.37**±0.171	**0.36**±0.207	**0.54**±0.093	0.01±0.036	**0.41**±0.173	**0.49**±0.150	0.55±0.134
	t-SNE	0.48±0.136	0.49±0.130	0.49±0.125	0.57±0.127	0.51±0.118	0.59±0.132	0.61±0.123	**0.54**±0.128
	UMAP	0.53±0.116	0.56±0.118	0.56±0.116	0.63±0.121	0.56±0.123	0.65±0.131	0.68±0.122	0.65±0.123

Table 2: RMSE (mean μ and standard deviation σ across the 10 repetitions for each problem) of the exploration rate τ_i^{y} in the 2-D space with respect to the one τ_i^{x} in the search space, for the two problems with $n = 1$ and $n_{\text{NN}} = 0$ (above) and with $n = 4$ and $n_{\text{NN}} = 4$ (below): the lower, the better. For each problem, l, n, and n_{NN} values, the best mean value is highlighted in bold.

Visualizing population dynamics requires to show not only the current generation but also some information on the previous generation(s). While this is already achieved by the 3-D visualization (see Figure 4), there are several ways to add such information to a 2-D visualization (see Figure 5). Let i be the current generation number. We have explored visualizing the points from Y_{i-1} that correspond to individuals from the previous generation using (a) a single color (light gray) that is different from the colors used to denote the points from Y_i, (b) smaller points than the ones used to visualize points from Y_i, and (c) transparency in contrast to the non-transparent points from Y_i. Additionally, by connecting the points from Y_{i-1} with those from Y_i based on their ancestry using a gray line, we provide additional information that can be useful when, for example, investigating how genetic operators are acting. We have furthermore experimented with extending such representations to include all previous $Y_j, 1 \leq j \leq i$, and additionally with visualizing the trajectory of the best individual in the population for generations $j, 1 \leq j \leq i$, using a red line.

Next, we showcase 3-D visualizations for the four dimensionality reduction techniques in Figure 4 and the different representations used in 2-D visualizations in Figure 5.

4.3.1 3-D visualization. The plots of Figure 4 show how the 3-D visualization based on MDS (second column), which resulted the most suitable technique according the experiments discussed in previous sections, can highlight the different aspects of an evolutionary run concerning population dynamics.

In the first problem (top row), there is a fast convergence toward the single optimum and the shape of the cloud of points obtained with MDS reflects this outcome of the evolution: a thin "column" in dark colors is visible with the best individual trajectory being "inside" the column. In the second problem, with diversity promotion and a larger l, the convergence is slower (see also Figure 1) and this is reflected in the taller and wider base of the column, slower change of color than in the top row and the longer trajectory of the best individual. Finally, in the third problem, the 3-D plot shows that at the beginning of the evolution the population is pursuing two optima and after ≈ 20 generations it crowds around only one optimum.

While PCA-based 3-D plots are visually similar to the corresponding MDS-based plots, those based on t-SNE and UMAP are not. Figure 4 shows how these two techniques tend to "magnify" the regions of the space which are more crowded (an effect that PCA and MDS, being linear, do not achieve). Although this ability could, in principle, be useful for visualizing an evolution (see, e.g., the sharply separated "short column" in the UMAP plot for the third problem, which corresponds to an optimum which is abandoned by the population after ≈ 20 generations), it turns out from our experiments that it negatively impacts on both our goals: showing population movements and revealing the exploration-exploitation trade-off.

4.3.2 2-D visualization. Figure 5 presents 2-D plots at different generations of an evolutionary run of the continuous problem with $l = 15$, $n = 3$, $n_{\text{NN}} = 4$ and MDS (the 3-D visualization of this example can be found in the second plot in the middle row of Figure 4). Each row in the figure uses a different representation for points from previous generations. The top row shows the previous generation in gray, while the second row shows it using smaller points (and additionally provides ancestry information by connecting offspring to their two parents). The gray color helps to discern the two generations more easily than the use of smaller points. On the other hand, the information on the fitness of the previous individuals is lost if all points use the same color. Using together both small points and ancestry connections makes it possible to, for example, visually confirm that the low performing individuals from the previous population were not used as parents in the current one (see second plot in the second row). The next two rows visualize all generations up to the current one either using gray (third row) or transparency (bottom row). Again, while gray makes a better distinction between the current and previous generations, using transparent points keeps some information on their fitness values.

The usefulness of the listed representations depends mainly on the purpose of the visualization. For example, the movement of the population is clearer when using a different color for denoting previous populations. Finally, independently of the visualization style we can see from the colors of the points in the current generation that at the beginning of the evolution, the fitness of the population worsens and generation 10 does not contain better individuals than

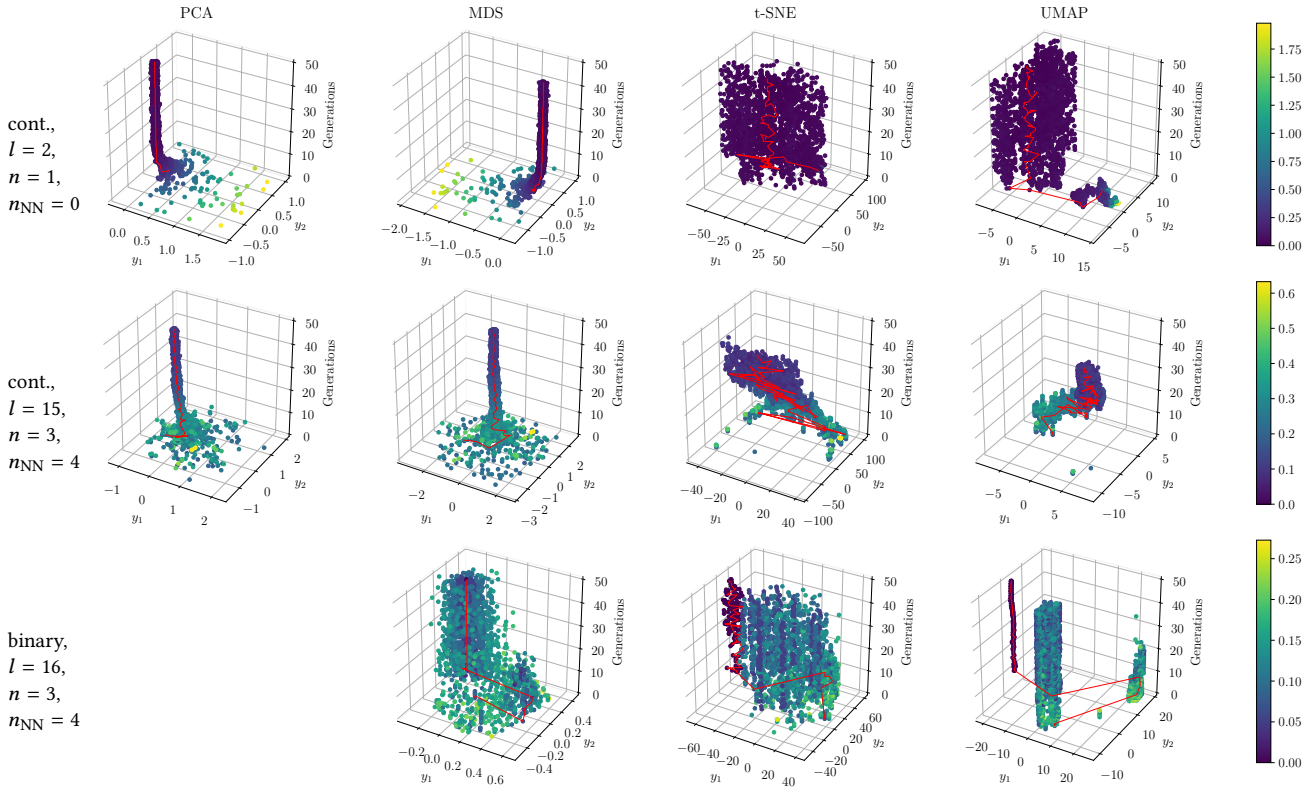

Figure 4: 3-D visualizations of entire evolutionary runs using four dimensionality reduction techniques (columns) on three different problems (rows): continuous opt. with $l = 2$, $n = 1$, $n_{NN} = 0$ (top row), continuous opt. with $l = 15$, $n = 3$, $n_{NN} = 4$ (middle row), and the bit string opt. with $l = 16$, $n = 3$, $n_{NN} = 4$ (bottom row). The red line shows the best trajectory.

the initial generation (but generation 30 does). This is probably due to the employed diversity preserving mechanism ($n_{NN} = 4$), which favors exploration to exploitation.

Comparing 3-D visualizations to 2-D ones, we can notice that the 3-D visualizations provide a nice overview of the entire evolution, while details can be better viewed in the 2-D plots. Because of this, both should be used simultaneously to facilitate acquiring insights from an evolutionary run.

4.4 Computational effort

We investigated the time taken to apply our methodology, i.e., to produce one visualization out of an evolutionary run. The largest computational effort is in the first phase (see Section 2), i.e., the dimensionality reduction. We hence measured the time taken by the different techniques.

We performed this phase using the appropriate Python libraries (scikit-learn for PCA, MDS, and t-SNE and the reference implementation for UMAP, at https://github.com/lmcinnes/umap); we run the code on AWS EC2 and on the CINECA HPC cluster. On EC2, we used the c4.8xlarge instances (36 vCPU based on 2.9 GHz Intel Xeon E5-2666, 60 GB RAM) and the m5.metal instances (96 vCPU based on 3.1 GHz Intel Xeon Platinum 8175, 384 GB RAM). On CINECA HPC cluster, we used the nodes of the Galileo partition (18 cores on 2 2.3 GHz Intel Xeon E5-2697 v4, 128 GB RAM).

For the sake of brevity, we report here only the mean time taken by the 4 techniques on the m5.metal AWS EC2 instances for the continuous optimization problems with $l = 2$ and $l = 15$, which were 6 s and 6 s for PCA, 265 s and 620 s for MDS, 21 s and 22 s for t-SNE, and 12 s and 12 s for UMAP. These figures show that MDS is by far the most expensive technique: depending on the actual practical settings, waiting some minutes to obtain a visualization might or might not be acceptable. It can also be seen that MDS takes much longer with larger values of l than the other techniques. Among the techniques which are not limited to continuous optimization problems, UMAP is the fastest, the difference with respect to t-SNE being larger for the continuous optimization problems.

5 CONCLUDING REMARKS AND FUTURE WORK

We considered the problem of visualizing the population dynamics along an evolutionary run. We proposed a visualization in a 2-D space obtained by applying a dimensionality reduction technique to individuals in their original search space. An evolutionary run may be visualized either by a sequence of 2-D frames, one representing the population at each generation, or by a 3-D representation with one 2-D plane for each generation.

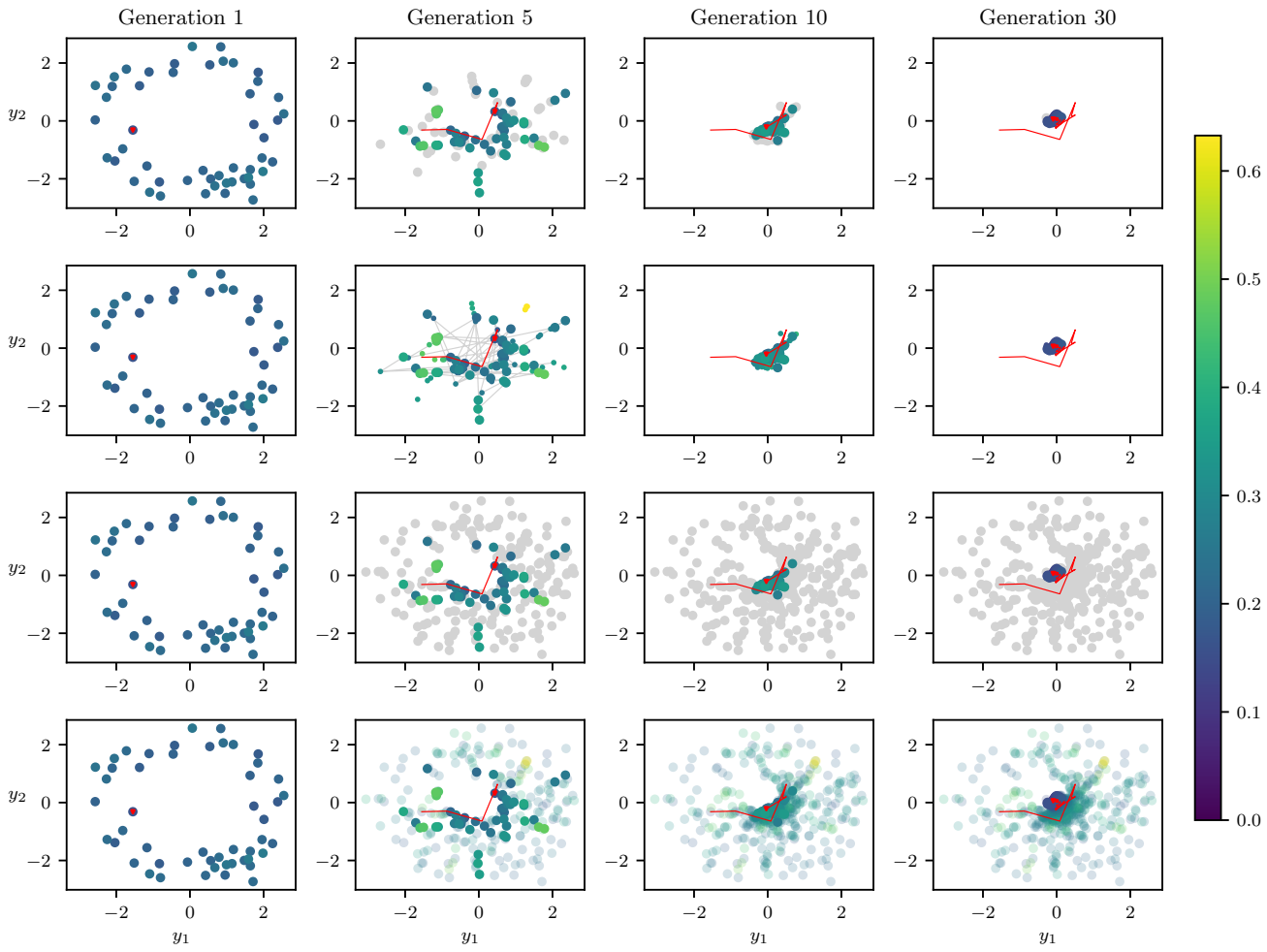

Figure 5: 2-D visualizations of an evolutionary run of the continuous problem with $l = 15$, $n = 3$, $n_{\text{NN}} = 4$ with MDS (corresponding to the second plot in the middle row of Figure 4). Columns show current generations (1, 5, 10, and 30), while the rows exhibit four ways to include information on previous generations: only the most recent previous generation in gray (top row), the most recent previous generation with smaller points and ancestry shown with gray lines (second row), all previous generations in gray (third row), and all previous generations with high transparency (bottom row). In all plots, the red triangle denotes the current best individual while the red line shows its trajectory through the generations.

We assessed numerous variants of the resulting framework and focused on the ability of the various dimensionality reduction techniques to preserve meaningful information about the population dynamics despite the unavoidable loss of information. The results are very promising as the proposed framework is indeed able to represent such crucial properties as population movements and exploration-exploitation trade-off in a way that is both highly informative and simple to understand.

As future work, we plan to extend the present work along three directions: (1) by including other dimensionality reduction techniques in the comparison (e.g., the one proposed very recently in [22]) and enlarging the set of considered problems (e.g., with symbolic regression with tree-based GP); (2) by investigating the possibility of using intrinsic dimensionality [20] for improving both the visualization and its analysis; (3) by releasing an implementation of our framework usable by EC practitioners and researchers.

ACKNOWLEDGEMENTS

The experimental evaluation of this work has been done on Amazon AWS within the "AWS Cloud Credits for Research" program and on CINECA HPC cluster within the CINECA-University of Trieste agreement. Tea Tušar acknowledges the financial support from the Slovenian Research Agency (project No. Z2-8177).

REFERENCES

[1] Heni Ben Amor and Achim Rettinger. 2005. Intelligent exploration for genetic algorithms: Using self-organizing maps in evolutionary computation. In *Companion Material Proceedings of the Genetic and Evolutionary Computation Conference,*

GECCO '05. ACM, 1531–1538.

[2] Bogdan Burlacu, Michael Affenzeller, Michael Kommenda, Stephan M. Winkler, and Gabriel Kronberger. 2013. Visualization of genetic lineages and inheritance information in genetic programming. In *Companion Material Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '13*. ACM, 1351–1358.

[3] Waldo Cancino Ticona, Nadia Boukhelifa, Anastasia Bezerianos, and Evelyne Lutton. 2013. Evolutionary visual exploration: Experimental analysis of algorithm behaviour. In *Companion Material Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '13*. 1373–1380.

[4] Trevor D. Collins. 2003. Applying software visualization technology to support the use of evolutionary algorithms. *Journal of Visual Languages and Computing* 14, 2 (2003), 123–150.

[5] Matej Črepinšek, Shih-Hsi Liu, and Marjan Mernik. 2013. Exploration and exploitation in evolutionary algorithms: A survey. *ACM Computing Surveys (CSUR)* 45, 3 (2013), 35.

[6] António Cruz, Penousal Machado, Filipe Assunção, and António Leitão. 2015. ELICIT: Evolutionary computation visualization. In *Companion Material Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '15*. ACM, 949–956.

[7] J. Drchal and M. Šnorek. 2007. Diversity visualization in evolutionary algorithms. In *Proceedings of 41th Spring International Conference, MOSIS'07*. Ostrava: MARQ, 77–84.

[8] Richard Dybowski, Trevor D. Collins, and P. R. Weller. 1996. Visualization of Binary String Convergence by Sammon Mapping. In *Proceedings of the Fifth Annual Conference on Evolutionary Programming*. MIT Press, 377–383.

[9] Emma Hart and Peter Ross. 2001. GAVEL—A new tool for genetic algorithm visualization. *IEEE Transactions on Evolutionary Computation* 5, 4 (2001), 335–348.

[10] Harold Hotelling. 1933. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology* 24, 6 (1933), 417.

[11] Guillaume Jornod, Ezequiel Di Mario, Iñaki Navarro, and Alcherio Martinoli. 2015. SwarmViz: An open-source visualization tool for Particle Swarm Optimization. In *Congress on Evolutionary Computation, CEC 2015*. IEEE, 179–186.

[12] Andreas Kerren and Thomas Egger. 2005. EAVis: A Visualization Tool for Evolutionary Algorithms. In *2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC 2005)*. IEEE, 299–301.

[13] Namrata Khemka and Christian Jacob. 2009. VISPLORE: a toolkit to explore particle swarms by visual inspection. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '09*. ACM, 41–48.

[14] Yong-Hyuk Kim, Kang Hoon Lee, and Yourim Yoon. 2009. Visualizing the search process of particle swarm optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '09*. ACM, 49–56.

[15] Shih-Hsi Liu, Matej Črepinšek, and Marjan Mernik. 2012. Analysis of VEGA and SPEA2 using exploration and exploitation measures. In *Proceedings of the 5th International Conference on Bioinspired Optimization Methods and their Applications, BIOMA 2012*. Jožef Stefan Institute, 97–108.

[16] Evelyne Lutton, Julie Foucquier, Nathalie Perrot, Jean Louchet, and Jean-Daniel Fekete. 2011. Visual analysis of population scatterplots. In *Revised Selected Papers from the 10th International Conference on Artificial Evolution (Evolution Artificielle), EA 2011 (Lecture Notes in Computer Science)*, Vol. 7401. Springer, 61–72.

[17] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[18] Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018).

[19] Nicholas Freitag McPhee, Maggie M. Casale, Mitchell Finzel, Thomas Helmuth, and Lee Spector. 2016. Visualizing genetic programming ancestries. In *Companion Material Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '16*. ACM, 1419–1426.

[20] Eric Medvet, Alberto Bartoli, Alessio Ansuini, and Fabiano Tarlao. 2018. Observing the Population Dynamics in GE by means of the Intrinsic Dimension. *arXiv preprint arXiv:1812.02504* (2018).

[21] Eric Medvet, Marco Virgolin, Mauro Castelli, Peter AN Bosman, Ivo Gonçalves, and Tea Tušar. 2018. Unveiling evolutionary algorithm representation with DU maps. *Genetic Programming and Evolvable Machines* 19, 3 (2018), 351–389.

[22] K. Michalak. 2019. Low-Dimensional Euclidean Embedding for Visualization of Search Spaces in Combinatorial Optimization. *IEEE Transactions on Evolutionary Computation* 23, 2 (2019), 232–246.

[23] Heinz Mühlenbein and Dirk Schlierkamp-Voosen. 1993. Predictive models for the breeder genetic algorithm i. continuous parameter optimization. *Evolutionary computation* 1, 1 (1993), 25–49.

[24] Hartmut Pohlheim. 2006. Multidimensional Scaling for Evolutionary Algorithms - Visualization of the Path through Search Space and Solution Space Using Sammon Mapping. *Artificial Life* 12, 2 (2006), 203–209.

[25] Hartmut Pohlheim. 2006. Understanding the Course and State of Evolutionary Optimizations Using Visualization: Ten Years of Industry Experience with Evolutionary Algorithms. *Artificial Life* 12, 2 (2006), 217–227.

[26] John W Sammon. 1969. A nonlinear mapping for data structure analysis. *IEEE Transactions on computers* 100, 5 (1969), 401–409.

[27] William B. Shine and Christoph F. Eick. 1997. Visualizing the evolution of genetic algorithm search processes. In *International Conference on Evolutionary Computation*. IEEE, 367–372.

[28] Giovanni Squillero and Alberto Tonda. 2016. Divergence of character and premature convergence: A survey of methodologies for promoting diversity in evolutionary optimization. *Information Sciences* 329 (2016), 782–799.

[29] Joshua B Tenenbaum, Vin De Silva, and John C Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *science* 290, 5500 (2000), 2319–2323.

[30] Warren S Torgerson. 1952. Multidimensional scaling: I. Theory and method. *Psychometrika* 17, 4 (1952), 401–419.

[31] Zoltán Tóth. 2003. A graphical user interface for evolutionary algorithms. *Acta Cybernetica* 16, 2 (2003), 337–365.

[32] Laurens van der Maaten, Eric Postma, and Jaap van den Herik. 2009. *Dimensionality Reduction: A Comparative Review*. Technical Report TiCC-TR 2009-005. Tilburg University Technical Report.