SIAM J. SCI. COMPUT.
Vol. 43, No. 3, pp. B831–B853

A SUPERVISED LEARNING APPROACH INVOLVING ACTIVE SUBSPACES FOR AN EFFICIENT GENETIC ALGORITHM IN HIGH-DIMENSIONAL OPTIMIZATION PROBLEMS*

NICOLA DEMO†, MARCO TEZZELE†, AND GIANLUIGI ROZZA†

Abstract. In this work, we present an extension of genetic algorithm (GA) which exploits the supervised learning technique called active subspaces (AS) to evolve the individuals on a lower-dimensional space. In many cases, GA requires in fact more function evaluations than other optimization methods to converge to the global optimum. Thus, complex and high-dimensional functions can end up extremely demanding (from the computational point of view) to be optimized with the standard algorithm. To address this issue, we propose to linearly map the input parameter space of the original function onto its AS before the evolution, performing the *mutation* and *mate* processes in a lower-dimensional space. In this contribution, we describe the novel method called ASGA, presenting differences and similarities with the standard GA method. We test the proposed method over n-dimensional benchmark functions—*Rosenbrock*, *Ackley*, *Bohachevsky*, *Rastrigin*, *Schaffer N. 7*, and *Zakharov*—and finally we apply it to an aeronautical shape optimization problem.

Key words. genetic algorithm, active subspaces, high-dimensional optimization

AMS subject classifications. 65K10, 90C26, 90C31

DOI. 10.1137/20M1345219

1. Introduction.

Genetic algorithm (GA) is a well-known and widespread methodology, mainly adopted in optimization problems [30, 32]. It emulates the evolutive process of natural selection by following an iterative process where the individuals are selected by a given objective function and subsequently they mutate and reproduce [24, 3, 11, 33]. This gradient-free technique is particularly effective when the objective function contains many local minima: thanks to the stochastic component, GA explores the domain without being blocked into local minima. The main disadvantage of such an algorithm is the (relatively) high number of required evaluations of the objective function during the evolution to explore the input space [37], which makes this method unfeasible in several industrial and engineering contexts for the global computational cost.

In this work, we propose a novel extension of standard GA, exploiting the emerging active subspaces (AS) property [13, 14] for the dimensionality reduction. AS is a supervised learning technique which allows the approximation of a scalar function with a lower-dimensional one, whose parameters are a linear combination of the original inputs. AS has been successfully employed in naval engineering applications [52, 49, 48, 50], coupled with reduced order methods such as POD-Galerkin

*Submitted to the journal's Computational Methods in Science and Engineering section June 12, 2020; accepted for publication (in revised form) March 31, 2021; published electronically June 29, 2021.

https://doi.org/10.1137/20M1345219

Funding: This work was partially supported by an industrial Ph.D. grant sponsored by Fincantieri S.p.A. (IRONTH Project), by the MIUR through the FARE-X-AROMA-CFD project, by the INdAM-GNCS 2019 project "Advanced intrusive and non-intrusive model order reduction techniques and applications," and by the European Union Funding for Research and Innovation—Horizon 2020 Program—in the framework of the European Research Council Executive Agency: H2020 ERC CoG 2015 AROMA-CFD project 681447 "Advanced Reduced Order Methods with Applications in Computational Fluid Dynamics" P.I. Professor Gianluigi Rozza.

†Mathematics Area, mathLab, SISSA, I-34136 Trieste, Italy (nicola.demo@sissa.it, marco.tezzele@sissa.it, gianluigi.rozza@sissa.it).

in biomedical applications [47, 42], POD with interpolation [19] in structural and CFD analysis, and dynamic mode decomposition [51] in CFD contexts. Other applications include aerodynamic shape optimization [35], artificial neural networks to reduce the number of neurons [16], nonlinear structural analysis [26], and AS for multivariate vector-valued model functions [54]. Several nonlinear AS extensions have been proposed recently. We mention active manifold [8] and kernel-based active subspaces (KAS) [40], which exploits the random Fourier features to map the inputs in a higher-dimensional space. We also mention the application of artificial neural networks for nonlinear reduction in parameter spaces by learning isosurfaces [55]. Despite these new nonlinear extensions of AS, in this work we exploit the classical linear version because of the possibility to map points in the reduced space onto the original parameter space.

The main idea of the proposed algorithm is to force the individuals of the population to evolve along the AS, which has a lower dimension, avoiding evolution along the meaningless directions. Further, the high number of function evaluations that characterize the GA is exploited within this new approach for the construction (and refinement) of the AS, making these techniques—GA produces a large dataset of input-output pairs, whereas AS needs large datasets for an accurate subspace identification—particularly suited together. This new method has the potential to improve the existing optimization pipeline involving both input and model order reduction.

A similar approach has been proposed in [12], where an active subspace is constructed in order to obtain an efficient and adaptive sampling strategy in an evolution strategy framework. This approach shares with the one we are proposing the idea of efficiently exploring the input space by constructing a subspace based on the collected data. In contrast with our approach in [12], the subspace construction is done with a singular value decomposition based method, and the optimization technique is completely different, even if evolution strategy methods and GA present some analogies. To the best of the authors' knowledge, the current contribution presents a novel approach, not yet explored in the literature. For a similar approach, we cite also random subspace embeddings for unconstrained global optimization of functions with low effective dimensionality that can be found in [53, 10], while for evolutionary methods and derivative-free optimization we mention [43, 39], respectively. For a survey on linear dimensionality reduction in the context of optimization programs over matrix manifolds, we mention [17].

The outline of this work is the following: the proposed method is described in section 4, while sections 2 and 3 are devoted to recalling the general family of genetic algorithms and the AS technique, respectively. Section 5 presents the numerical results obtained applying the proposed extension to some popular benchmark functions for optimization problems and then to a typical engineering problem where the shape of a NACA airfoil is morphed to maximize the lift-to-drag coefficient. Finally, section 6 summarizes the benefits of the method and proposes some extensions for future developments.

2. Genetic algorithms. In this work, we propose an extension of the standard genetic algorithm (GA). We start by recalling the general method in order to easily let the reader understand the differences. We define GA as the family of computational methods that are inspired by Darwin's theory of evolution. The basic idea is to generate a population of individuals with random genes, and make them evolve through mutations and crossovers, mimicking the evolution of living beings. Iterating this process by selecting at each step the best-fit individuals results in the optimization—

according to a specific objective function—of the original population. As such, this method can be easily adopted as a global optimization algorithm.

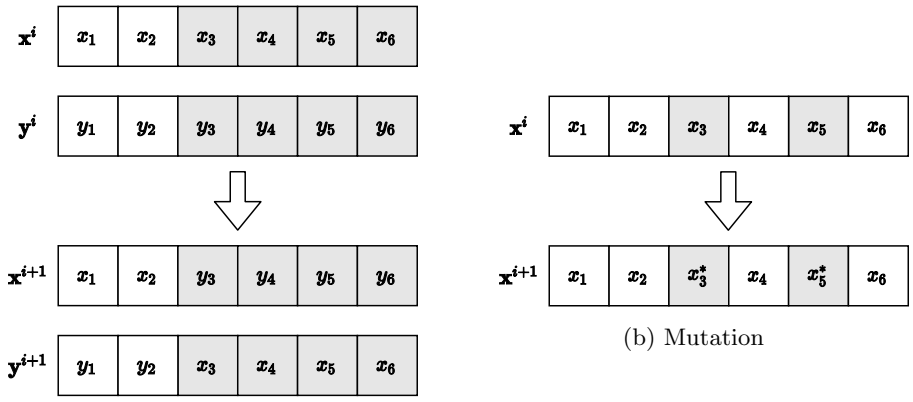
Initially proposed by Holland in [29], GA has had several modifications during the years (see, for example, [32, 22, 45, 23, 21]), but it keeps its fundamental steps: *selection*, *mutation,* and *mate*.

Let us define formally the individuals: a population composed by N individuals $\mathbf{x}_i \in \mathbb{R}^P$ with P genes is defined as $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$. We express the fitnesses of such individuals with the scalar function $f : \mathbb{R}^P \to \mathbb{R}$. The first generation $\mathbf{X}^1$ is randomly created—with possible constraints—and the fitness is evaluated for all the individuals: $\mathbf{y}_i = f(\mathbf{x}_i)$ for $i = 1, \dots, N$. Then the following iterative process starts:

Selection: The best individuals of the previous generation $\mathbf{X}^i$ are chosen accordingly to their fitnesses to breed the new generation. For the selection, several strategies can be adopted depending on the problem and on the cardinality of the population N.

Mate: Finally, the selected individuals are grouped into pairs and, according to a mate probability, they combine their genes to create new individuals. The process, also called *crossover*, emulates the species reproduction. These individuals form the new generation $\mathbf{X}^{i+1}$. An example of a crossover method is sketched in Figure 1a.

Mutation: The individuals evolve by changing some of their genes. The mutation of an individual is usually controlled by a mutation probability. In Figure 1b, we show an illustrative example where two genes have randomly mutated.



(a) Mate

(b) Mutation

FIG. 1. *Graphical example of mate and mutation where* $\mathbf{x}^i$ *and* $\mathbf{y}^i$ *indicate two generic individuals of the* i*th generation.*

After the mutation step, the fitness of the new individuals is computed and the algorithm restarts with the selection of the best-fit individuals. In this way, the population evolves, generation after generation, towards the optimal individual, avoiding getting blocked in a local minimum thanks to the stochastic component introduced by mutation and crossover. Thus, this method is very effective for global optimization where the objective function is potentially nonlinear, while standard gradient-based methods can converge to local minima. However, GA usually requires a high number of evaluations to perform the optimization, making this procedure very expensive in the case of computational costly objective functions.

3. Active subspaces for minimization on a lower-dimensional parameter space. The active subspaces (AS) [13] method is a dimensionality reduction approach for parameter space studies, which falls under the category of supervised learning techniques. AS tries to reduce the input dimension of a scalar function $f(\boldsymbol{\mu}) : \Omega \subset \mathbb{R}^k \to \mathbb{R}$ by defining a linear transformation $\boldsymbol{\mu}_M = \mathbf{A}\boldsymbol{\mu}$. This method requires the evaluation of the gradients of f since $\mathbf{A}$ depends on the second moment matrix $\mathbf{C}$ of the target function's gradient, also called the uncentered covariance matrix of the gradients of f. This matrix is defined as follows:

$$\mathbf{C} = \mathbb{E}\left[\nabla_{\boldsymbol{\mu}} f \, \nabla_{\boldsymbol{\mu}} f^T\right] = \int (\nabla_{\boldsymbol{\mu}} f)(\nabla_{\boldsymbol{\mu}} f)^T \rho \, d\boldsymbol{\mu}, \tag{3.1}$$

where with the symbol $\mathbb{E}[\cdot]$ we denote the expected value, $\nabla_{\boldsymbol{\mu}} f \equiv \nabla f(\boldsymbol{\mu}) \in \mathbb{R}^k$, and $\rho : \mathbb{R}^k \to \mathbb{R}^+$ is a probability density function which represents the uncertainty in the input parameters. In practice, the matrix $\mathbf{C}$ is constructed with a Monte Carlo procedure, and the gradients if not provided can be approximated with different techniques, such as local linear models, global models, finite difference, or a Gaussian process—for a comparison of the methods and corresponding errors, the reader can refer to [13, 14, 15]. The uncentered covariance matrix can be decomposed as

$$\mathbf{C} = \mathbf{W}\Lambda\mathbf{W}^T, \tag{3.2}$$

where $\mathbf{W}$ stands for the orthogonal matrix containing the eigenvectors, and Λ stands for the eigenvalues matrix arranged in descending order. The spectral gap [13] is used to bound the error on the numerical approximation with Monte Carlo. We can decompose these two matrices as

$$\Lambda = \begin{bmatrix} \Lambda_1 & \\ & \Lambda_2 \end{bmatrix}, \qquad \mathbf{W} = \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 \end{bmatrix}, \qquad \mathbf{W}_1 \in \mathbb{R}^{k \times M}, \tag{3.3}$$

where $M < k$ is the dimension of the active subspace. M should be chosen looking at the energy decay (the tail in the ordered eigenvalue sum) as in POD, or it can be prescribed a priori for the specific task. We can exploit this decomposition to map the input parameters onto a reduced space.

We define the active subspace of dimension M as the principal eigenspace corresponding to the eigenvalues prior to the major spectral gap. We also call the active variable $\boldsymbol{\mu}_M$ and the inactive variable $\boldsymbol{\eta}$. They are defined as $\boldsymbol{\mu}_M = \mathbf{W}_1^T\boldsymbol{\mu} \in \mathbb{R}^M$ and $\boldsymbol{\eta} = \mathbf{W}_2^T\boldsymbol{\mu} \in \mathbb{R}^{k-M}$.

In this work, we address the constrained global optimization problem of a real-valued continuous function, in the context of genetic algorithms, defined as

$$\min_{\boldsymbol{\mu} \in \Omega \subset \mathbb{R}^k} f(\boldsymbol{\mu}). \tag{3.4}$$

To fight the curse of dimensionality and speed up the convergence, we exploit the AS property of the target function to select the best individuals in the reduced parameter space, to mutate and mate them, and successively to map them in the full parameter space. This translates in the following optimization problem for each generation of individuals:

$$\min_{\substack{\boldsymbol{\mu}_M \in \mathcal{P} \subset \mathbb{R}^M \\ \boldsymbol{\mu} \in \Omega}} g(\boldsymbol{\mu}_M = \mathbf{W}_1^T\boldsymbol{\mu}), \tag{3.5}$$

where $\mathcal{P}$ is the polytope in $\mathbb{R}^M$—we assume the ranges of the parameters to be intervals—defined by AS as $\mathcal{P} := \{\boldsymbol{\mu}_M = \mathbf{W}_1^T\boldsymbol{\mu} \mid \boldsymbol{\mu} \in \Omega\}$. We remark that there are many choices for the profile g. In this work, we consider the following profile:

$$g(\mathbf{y}) := f(\mathbf{x_y}) \qquad \forall \mathbf{y} \in \mathcal{Y}, \tag{3.6}$$

$$\mathcal{Y} := \{\mathbf{y} \in \mathcal{P} \mid \exists\, \mathbf{x_y} \in \Omega \text{ s.t. } \mathbf{y} = \mathbf{W}_1^T\mathbf{x_y}\}. \tag{3.7}$$

We emphasize that the projection map onto the active subspace is a surjective map because $\mathbf{W}_1^T$ is defined as a linear projection onto a subspace, and hence it is surjective by definition. So the back-mapping from the active subspace onto Ω is not trivial. Given a point $\boldsymbol{\mu}_M^*$ in the active subspace, we can find B points in the original parameter space which are mapped onto $\boldsymbol{\mu}_M^*$ by $\mathbf{W}_1^T$. Recalling the decomposition above, we have that

$$\boldsymbol{\mu} = \mathbf{W}_1\mathbf{W}_1^T\boldsymbol{\mu} + \mathbf{W}_2\mathbf{W}_2^T\boldsymbol{\mu} = \mathbf{W}_1\boldsymbol{\mu}_M + \mathbf{W}_2\boldsymbol{\eta} \qquad \forall \boldsymbol{\mu} \in \Omega, \tag{3.8}$$

with the additional constraint coming from the rescaling of the input parameters needed to apply AS: $-\mathbf{1} \le \boldsymbol{\mu} \le \mathbf{1}$, where $\mathbf{1}$ denotes the vector in $\mathbb{R}^k$ with all elements equal to 1. We exploit this to sample the inactive variable $\boldsymbol{\eta}$ so that

$$-\mathbf{1} \le \mathbf{W}_1\boldsymbol{\mu}_M^* + \mathbf{W}_2\boldsymbol{\eta} \le \mathbf{1}, \tag{3.9}$$

or equivalently

$$\mathbf{W}_2\boldsymbol{\eta} \le \mathbf{1} - \mathbf{W}_1\boldsymbol{\mu}_M^*, \qquad -\mathbf{W}_2\boldsymbol{\eta} \le \mathbf{1} + \mathbf{W}_1\boldsymbol{\mu}_M^*. \tag{3.10}$$

These inequalities define a polytope in $\mathbb{R}^{k-M}$ from which we want to uniformly sample B points. The inactive variables are sampled from the conditional distribution $p(\boldsymbol{\eta}|\boldsymbol{\mu}_M^*)$, and we show how to perform it for the uniform distribution. For a more general distribution, one should use Hamiltonian Monte Carlo. In particular, we start with a simple rejection sampling scheme, which finds a bounding hyperbox for the polytope, draws points uniformly from it, and rejects points outside the polytope. If this method does not return enough samples, we try a *hit-and-run* method [46, 6, 34] for sampling from the polytope. This method, starting from the center of the largest hypersphere within the polytope, selects a random direction and identifies the longest segment lying inside the polytope. A new sample is randomly drawn along this segment. The procedure continues starting from the last sample until enough samples are found. If also that does not work, we use B copies of a feasible point computed as the Chebyshev center [7] of the polytope. In Figure 2, we depicted these strategies at different stages of the sampling.

4. The proposed ASGA optimization algorithm. In this section, we are going to describe the proposed AS extension of the standard GA, named ASGA. Before starting, we emphasize that in what follows, we will maintain the selection, mutation, and mate procedures—presented in section 2—as general as possible, without going into technical details, given the large variety of different options for these steps. In fact, the proposed extension is independent of the chosen evolution strategies, and we only perform them in a lower dimension exploiting AS. In Algorithm 4.1, we summarize the standard approach, while in Algorithm 4.2 we highlight the differences introduced by ASGA. We also present an illustration for both of the methods in Figure 3, where the yellow boxes indicate the main steps peculiar to ASGA. In both cases, the first step is the generation of the random individuals composing the initial

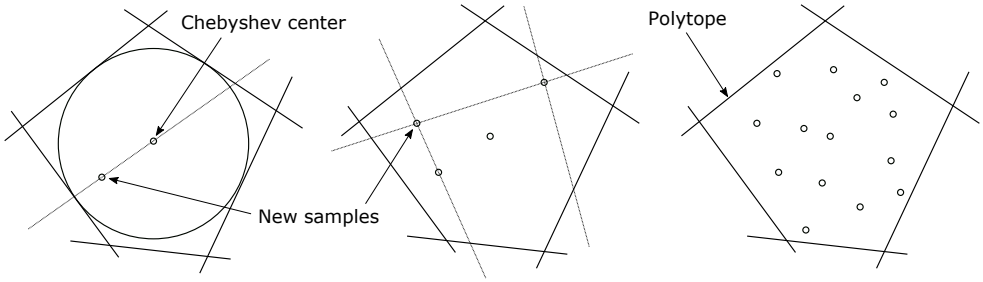


FIG. 2. *Graphical representation of the inactive variable sampling strategy. We emphasize the Chebyshev center, the selection of the next sample using the* hit-and-run *method, and the polytope defined by* (3.10).

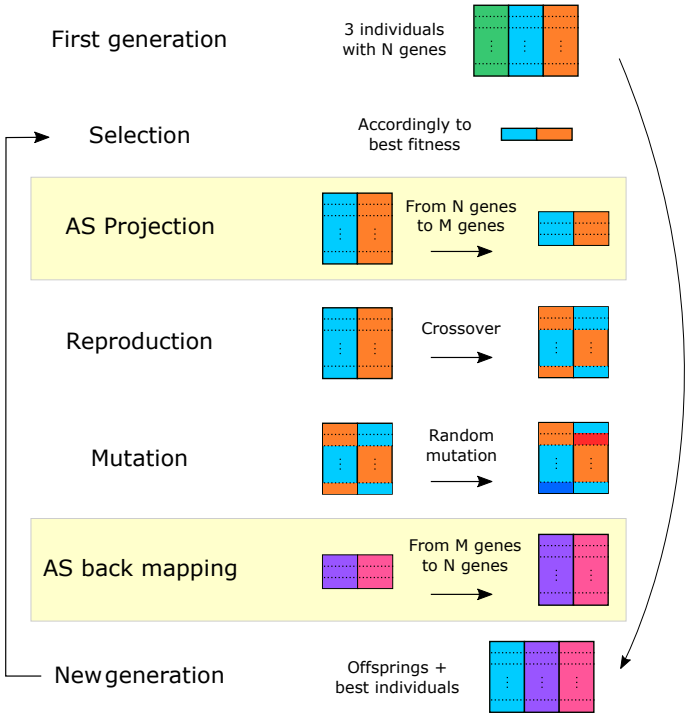


FIG. 3. *Active subspaces-based genetic algorithm scheme. The main steps of the classical GA are depicted from top to bottom. The yellow boxes (color available online only) represent projections onto and from lower dimension active subspace, which are specific to ASGA.*

population and the sequential evaluation of all of them. For ASGA, these individuals and their fitnesses are stored in two additional sets, $\mathbf{X}^{\mathrm{AS}}$ for the individuals and $\mathbf{y}^{\mathrm{AS}}$ for the fitness. We will exploit them as the input-output pair for the construction of AS. After the selection of the best-fit individuals, the active subspace of dimension M is built and the selected offspring is projected onto it. The low-dimensional individuals mate and mutate in the active subspace. Thanks to the reduced dimension and to the fact that we retain only the most important dimensions, these operations are much more efficient. Thus, even if AS of dimension M does not provide an accurate approximation of the original full-dimensional space, the active dimensions

will provide preferential directions for the evolution, making the iterative process smarter and faster.

After the evolution, the low-dimensional offspring is mapped back to the original space. In section 3, we describe how for any point in the active subspace we can find several points in the original space which are mapped onto it. So we select, for any individual in the offspring, B full-dimensional points which correspond to the individual in the active subspace. We emphasize that to preserve the same dimensionality of the offspring between the original GA and the AS extension, in the proposed algorithm we select the N/B best individuals, instead of selecting N. In this way, after the back-mapping, the offspring has dimension N in both versions. The number of back-mapped points B and the active subspace dimension M—that can be a fixed parameter or dynamically selected from the spectral gap of the covariance matrix $\mathbf{C}$—represent the new (hyper-)parameters of the proposed method.

Finally, the fitnesses of the new individuals, now in the full-dimensional space, are evaluated. To make AS more precise during the iterations, the evaluated individuals and their fitnesses are added to $\mathbf{X}^{\mathrm{AS}}$ and $\mathbf{y}^{\mathrm{AS}}$. The process restarts from the selection of the offspring from the new generation, continuing as described above until the stopping criteria are met.

We stress that the structure of the algorithm is similar to the original GA approach, with the difference that the gradients at the sample points are approximated in order to identify the dimensions with the highest variance. Even if such information about the function gradient is used, the ASGA method is different from gradient-based methods: numerically computing the gradient with a good accuracy at a specific point—which is the fundamental step of gradient-based methods to move on the solution manifold—is a very expensive procedure, especially in a high-dimensional space. In ASGA, we avoid such computation, exploiting instead the already collected function evaluations. Further, gradient-based techniques converge (relatively) fast to optimum, but they get blocked into local minima, contrarily to the ASGA approach. It is important to remark that, for each generation, AS is rebuilt from scratch, losing efficiency but gaining more precision due to the growing number of elements in the two sets $\mathbf{X}^{\mathrm{AS}}$ and $\mathbf{y}^{\mathrm{AS}}$. We also remark that the samples are generated with a uniform distribution only at the first generation. After that, due to the ASGA steps the distribution changes in a way which cannot be known a priori. For the computation of the expectation operator in (3.1), in this work we assume a uniform distribution. Even if this may introduce an unknown error, the numerical results achieved by ASGA seem to support such a choice. Of course the numerical estimates present in the literature for the uniform distribution do not apply in such a case. This method can be viewed as an active learning procedure in a Bayesian integration context, where the maximized acquisition function is heuristic and given by the application of AS and GA steps. Another interpretation is that we are enriching the local information near the current minimum to feed the AS algorithm, so it can be viewed as a weighted AS.

Algorithm 4.1. Standard GA.

Require: initial population size N_0
Require: population size N
Require: selection routine SELECT
Require: mutation routine MUTATE
Require: mate routine MATE
Require: objective function FOBJ
Require: stop criteria
Ensure: final population $\mathbf{X}^{\text{end}}$
$g \leftarrow 0$
$\mathbf{X}^g \leftarrow$ random pop of size N_0
$\mathbf{y}^g \leftarrow \text{FOBJ}(\mathbf{X}^g)$
repeat
  $g \leftarrow g + 1$
  $\mathbf{X}^* \leftarrow \text{SELECT}(\mathbf{X}^{g-1}, \mathbf{y}^{g-1}, N)$
  $\mathbf{X}^* \leftarrow \text{MATE}(\mathbf{X}^*)$
  $\mathbf{X}^g \leftarrow \text{MUTATE}(\mathbf{X}^*)$
  $\mathbf{y}^g \leftarrow \text{FOBJ}(\mathbf{X}^g)$
until stop criteria reached
$\mathbf{X}^{\text{end}} \leftarrow \mathbf{X}^g$
return $\mathbf{X}^{\text{end}}$

Algorithm 4.2. Proposed ASGA.

Require: initial population size N_0
Require: population size N
Require: active dimension M
Require: number backward B
Require: selection routine SELECT
Require: mutation routine MUTATE
Require: mate routine MATE
Require: objective function FOBJ
Require: stop criteria
Ensure: final population $\mathbf{X}^{\text{end}}$
$g \leftarrow 0$
$\mathbf{X}^g \leftarrow$ random pop of size N_0
$\mathbf{y}^g \leftarrow \text{FOBJ}(\mathbf{X}^g)$
$\mathbf{X}^{\text{AS}} \leftarrow \mathbf{X}^g$
$\mathbf{y}^{\text{AS}} \leftarrow \mathbf{y}^g$
repeat
  $g \leftarrow g + 1$
  $\mathbf{X}^* \leftarrow \text{SELECT}(\mathbf{X}^{g-1}, \mathbf{y}^{g-1}, \frac{N}{B})$
  build AS($\mathbf{X}^{\text{AS}}, \mathbf{y}^{\text{AS}}, M$)
  $\mathbf{X}^*_M \leftarrow \text{FORWARD}(\mathbf{X}^*)$
  $\mathbf{X}^*_M \leftarrow \text{MATE}(\mathbf{X}^*_M)$
  $\mathbf{X}^*_M \leftarrow \text{MUTATE}(\mathbf{X}^*_M)$
  for $\mathbf{x}$ in $\mathbf{X}^*_M$ **do**
    for $i \leftarrow 1$ to B **do**
      $\mathbf{X}^g \leftarrow \text{BACKWARD}(\mathbf{x})$
    end for
  end for
  $\mathbf{y}^g \leftarrow \text{FOBJ}(\mathbf{X}^g)$
  $\mathbf{X}^{\text{AS}} \leftarrow \mathbf{X}^{\text{AS}} \cup \mathbf{X}^g$
  $\mathbf{y}^{\text{AS}} \leftarrow \mathbf{y}^{\text{AS}} \cup \mathbf{y}^g$
until stop criteria reached
$\mathbf{X}^{\text{end}} \leftarrow \mathbf{X}^g$
return $\mathbf{X}^{\text{end}}$

5. Numerical results. In this section, we are going to present the results obtained by applying the proposed algorithm, first to some test functions that are usually used as benchmarks for optimization problems. Since this method is particularly suited for high-dimensional functions, we analyze the optimization convergence for three different input dimensions (2, 15, and 40), i.e., the number of genes of each individual. The second test case we propose is instead a typical engineering problem, where we optimize the lift-to-drag coefficient of a NACA airfoil which is deformed using a map $\mathcal{M} : \mathbb{R}^{10} \to \mathbb{R}$ defined in subsection 5.2. In this example, we opted for the use of a surrogate model only to evaluate the individuals' fitnesses for computational considerations since we just want to compare ASGA with GA. We do not rely on the surrogate for the gradient approximation. In [18], instead, we apply ASGA on a naval engineering hydrodynamics problem, where we do not rely on a surrogate model of the target function, but instead we exploit data-driven model order reduction methods to reconstruct the fields of interest and then compute the function to optimize.

In both of the test cases, in order to collect a fair comparison, we adopt the same routines for the selection, the mutation and the crossover steps. In particular, we do the following:

- For the *mate*, we use the blend BLX-alpha crossover [25] with $\alpha = 1.0$, with a mate probability of 50%. With this method, the offspring results:

$$\begin{cases} \mathbf{x}_a^i = (1-\gamma)\mathbf{x}_a^{i-1} + \gamma\mathbf{x}_b^{i-1} \\ \mathbf{x}_b^i = \gamma\mathbf{x}_a^{i-1} + (1-\gamma)\mathbf{x}_b^{i-1} \end{cases} \qquad \text{for } a, b = 1, \dots, N, \tag{5.1}$$

where $\mathbf{x}_a^{i-1}$ and $\mathbf{x}_b^{i-1}$ refer to the parent individuals (at the $i-1$th generation),

$\mathbf{x}_a^i$ and $\mathbf{x}_b^i$ are the mated individuals, N is the cardinality of population, and γ is a random variable chosen in the interval $[-\alpha, 1+\alpha)$. We mention that (5.1) can recover the graphical description of mating in Figure 1a if γ is taken to be discrete, either 0 or 1, and applied componentwise.

- For the *mutation*, a Gaussian operator [28] has been used with a mutation probability of 50%. This strategy changes genes by adding a normal noise. Since we do not have any knowledge about the low-dimensional space, tuning the variance of such a mutation may result in a nontrivial procedure. This quantity in fact has to be set in order to explore the input space but, at the same time, producing minimal differences between parents and offspring. A fixed variance for both the spaces may cause a too big distance—in the l_2 sense—between parents and offspring, inhibiting the convergence. To overcome this potential problem, we correlate the Gaussian variance with the genes themselves, ensuring a reasonable mutation in both spaces. The adopted mutation method is

$$\mathbf{x}_a^i = \mathbf{x}_a^{i-1} + \varepsilon\, \mathbf{x}_a^{i-1} \qquad \text{for } a = 1, \dots, N, \tag{5.2}$$

where ε is a random variable with probability distribution $\mathcal{N}(\mu, \sigma^2)$, that is, $\varepsilon \sim \mathcal{N}(\mu, \sigma^2)$, with $\mu = 0$ and $\sigma^2 = 0.1$.

Regarding the *selection*, because of the limited number of individuals per population, we adopt one of the simplest criteria, by selecting the N best individuals in terms of fitness.

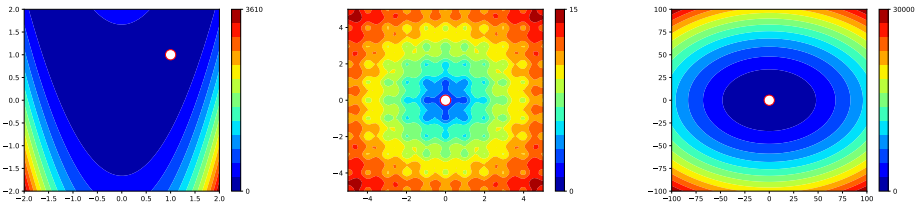
We also keep fixed the additional parameters for the AS extension: the number of active dimensions M is set to 1, while the number of back-mapped points is 2. All the gradient computations are done using local linear models. For the actual computation regarding AS, we used the ATHENA[1] Python package [41]. The only varying parameters are the size of the initial population N_0, the size of the population during the evolution N, and the number of generations in the evolutive loop, which are chosen empirically based on the objective function. We emphasize that, due to the stochastic nature of these methods, we repeated the tests 15 times, with different initial configurations, presenting the mean value, the minimum, and the maximum over the 15 runs.

5.1. Benchmark test functions. We applied the optimization algorithm to six different n-dimensional test functions, which have been chosen to cover a large variety of possible shapes. For all the functions, the results of the proposed method are compared to the results obtained using the standard genetic approach. In detail, the functions we tested are the so-called *Rosenbrock*, *Ackley*, *Bohachevsky*, *Rastrigin*, *Schaffer, N.* 7, and *Zakharov* test functions. In Figure 4, we depict the test functions in their two-dimensional form. In the following paragraphs, we briefly introduce them before presenting the obtained results. For a complete literature survey on benchmark functions for global optimization problems, we suggest [31].

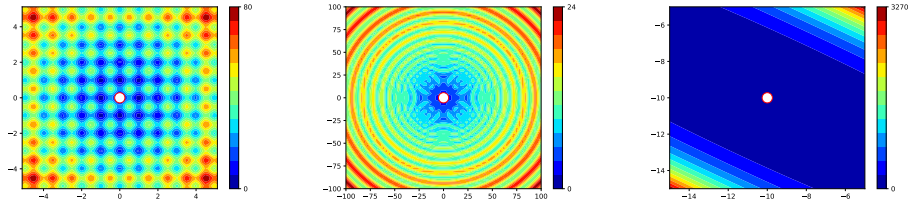
(a) *Rosenbrock function.* The Rosenbrock function is a widespread test function in the context of global optimization [20, 5, 38]. We choose it as representative of the valley-shaped test functions. The general d-dimensional formulation is the following:

$$f(x) = \sum_{i=1}^{d-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]. \tag{5.3}$$

[1] This is freely available online from https://github.com/mathLab/ATHENA.



(a) Rosenbrock function (b) Ackley function (c) Bohachevsky function



(d) Rastrigin function (e) Schaffer N. 7 function (f) Zakharov function

FIG. 4. *Benchmark test functions representation in 2D. White dots indicate the global minima.*

Its global minimum is $f(x^*) = 0$ at $x^* = (1, 1, \ldots, 1)$. As we can see from Figure 4a, the minimum lies on an easy to find parabolic valley, but the convergence to the actual minimum is notoriously difficult. We evaluated the function in the hypercube $[-5, 10]^d$.

(b) *Ackley function.* The Ackley function is characterized by many local minima, making it difficult to find the global minimum, especially for hillclimbing algorithms [4, 2]. The general d-dimensional formulation is the following:

$$f(x) = -a \exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\sqrt{\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)}\right) + a + \exp(1), \tag{5.4}$$

where a, b, and c are set to 20, 0.2, and 2π, respectively. Its global minimum is $f(x^*) = 0$ at $x^* = (0, 0, \ldots, 0)$. As we can see from Figure 4b, the function is nearly flat in the outer region, with many local minima, and the global minimum lies on a hole around the origin. The function has been evaluated in the domain $[-15, 30]^d$.

(c) *Bohachevsky function.* The Bohachevsky function is a representative of the bowl-shaped functions. There are many variants, and we chose the general d-dimensional formulation as the following:

$$f(x) = \sum_{i=1}^{d-1}(x_i^2 + 2x_{i+1}^2 - 0.3\cos(3\pi x_i) - 0.4\cos(4\pi x_{i+1}) + 0.7). \tag{5.5}$$

Its global minimum is $f(x^*) = 0$ at $x^* = (0, 0, \ldots, 0)$. As we can see from Figure 4c, the function has a clear bowl shape. This function has been evaluated in the domain $[-100, 100]^d$.

(d) *Rastrigin function.* The Rastrigin function is another difficult function to deal with for global optimization with GA due to the large search space and its many local

minima [36]. The general d-dimensional formulation is the following:

$$f(x) = 10d + \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i)]. \tag{5.6}$$

Its global minimum is $f(x^*) = 0$ at $x^* = (0, 0, \ldots, 0)$. As we can see from Figure 4d, the function is highly multimodal with local minima regularly distributed. We evaluated this function in the input domain $[-5.12, 5.12]^d$.

(e) *Schaffer N. 7 function.* The Schaffer N. 7 function [44] is a stretched V sine wave. The general d-dimensional formulation is the following:

$$f(x) = \sum_{i=1}^{d-1} (x_i^2 + x_{i+1}^2)^{0.25} \left[\sin^2(50(x_i^2 + x_{i+1}^2)^{0.10}) + 1\right]. \tag{5.7}$$

Its global minimum is $f(x^*) = 0$ at $x^* = (0, 0, \ldots, 0)$. As we can see from Figure 4e, the function presents many local minima. The optimization has been performed in the hypercube $[-100, 100]^d$.

(f) *Zakharov function.* The Zakharov function is a representative of the plate-shaped functions. It has one global minimum and no additional local minima. The general d-dimensional formulation is the following (after a shift):

$$f(x) = \sum_{i=1}^{d} (x_i + 10)^2 + \left(\sum_{i=1}^{d} \frac{i}{2}(x_i + 10)\right)^2 + \left(\sum_{i=1}^{d} \frac{i}{2}(x_i + 10)\right)^4. \tag{5.8}$$

We emphasize that we used a shifted version with global minimum $f(x^*) = 0$ at $x^* = (-10, -10, \ldots, -10)$. This choice is made to prove that the proposed method is not biased towards minima around the origin. We can see from Figure 4f the function for $d = 2$. We evaluated the Zakharov function in the domain $[-15, 0]^d$.

All the test cases presented share the same hyperparameters described at the beginning of this section, except for the population size. For the two-dimensional benchmark functions, the two algorithms are tested creating $N_0 = 200$ random individuals for the initial population and then keeping an offspring of dimension $N = 100$. Figure 5 shows the behavior for all the test functions. For this space dimension, the two trends are very similar: the usage of the proposed algorithm does not make the optimization faster, and it adds the computational overhead for the AS construction. Despite that, the results after 10 generations are very similar, and we can consider this as a worst case scenario, where a clear reduction in the parameter space is not possible.

The ASGA performance gain changes drastically, increasing the number of dimension to $d = 15$, as demonstrated in Figure 6. For such a dimension, the two parameters N_0 and N are set to 2000 and 200, respectively. Starting from this dimension, it is possible to note a remarkable difference between the standard method and the proposed one. The greater the input dimension, the greater the gain produced by ASGA, due to the exploitation of the AS reduction. All the benchmarks show a faster decay, but we can isolate two different patterns in the evolution: *Rosenbrock* and *Ackley* show a very steep trend in the first generation gain, while for the next generations the population is not able to decrease its fitness as much as before, showing a quasi-constant behavior. The difference with the standard GA is maximized in the first generation, but even if the evolution using ASGA is not so effective after the

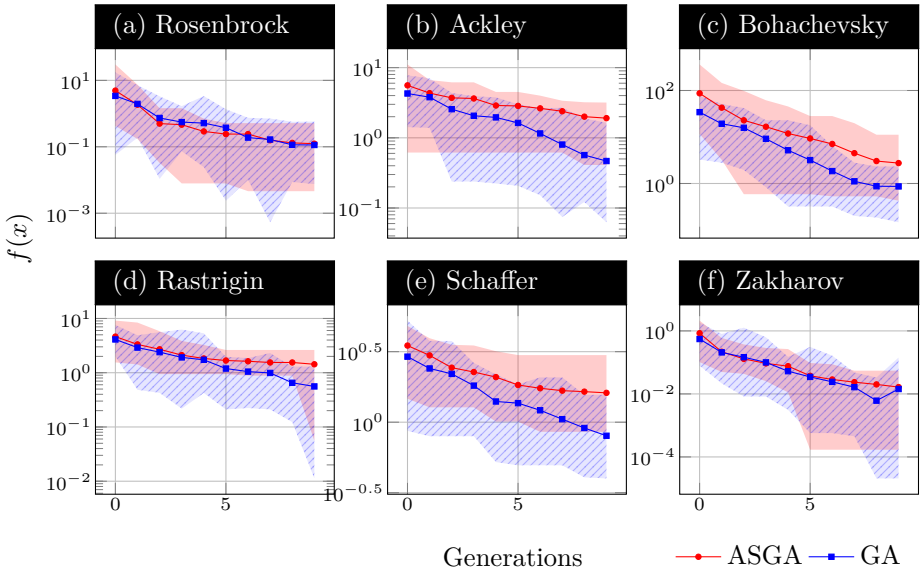


FIG. 5. *Results of the optimization of the benchmark functions in a space of dimension* $d = 2$. *We compare the standard GA (in blue square dots) with the proposed algorithm ASGA (in red circle dots) using an initial population of size* 30*, while the dimension for each generation is fixed to* 10. *The solid lines represent the mean, over* 15 *runs, of the objective function corresponding to the best individual at each generation. The shaded areas show the interval between minimum and maximum (blue with lines for GA and red for ASGA). Color is available online only.*

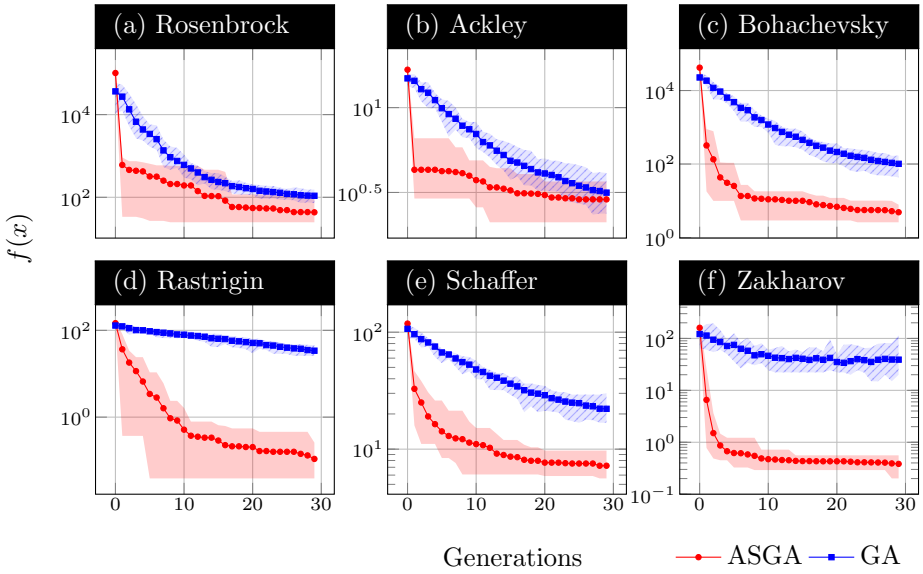


FIG. 6. *Results of the optimization of the benchmark functions in a space of dimension* $d = 15$. *We compare the standard GA (in blue square dots) with the proposed algorithm ASGA (in red circle dots) using an initial population of size* 2000*, while the dimension for each generation is fixed to* 200. *The solid lines represent the mean, over* 15 *runs, of the objective function corresponding to the best individual at each generation. The shaded areas show the interval between minimum and maximum (blue with lines for GA and red for ASGA). Color is available online only.*

first generation in these two cases, the proposed method is anyway able to achieve a better result (on average) after 30 generations. The other benchmarks instead show a much smoother decay, gradually converging to the optimum. Despite the lack of the initial step, for these benchmarks the gain with respect to the standard approach becomes bigger, even if after several generations the convergence rate decreases. In order to better understand these differences, we investigate the spectra of the AS covariance matrices for all the benchmarks, which are reported in Figure 7. The patterns individuated in the optimizations are partially reflected in the eigenvalues: *Rosenbrock*, *Ackley*, and *Zakharov* have an evident gap between the first and the second eigenvalues, which results in a better approximation (of the original function) in the one-dimensional subspace. However, the order of magnitude of the first eigenvalue is different between the three functions: for *Rosenbrock* and *Ackley*, the magnitude is greater than 1×10^{-1}, whereas for *Zakharov* it is around 5×10^{-2}.

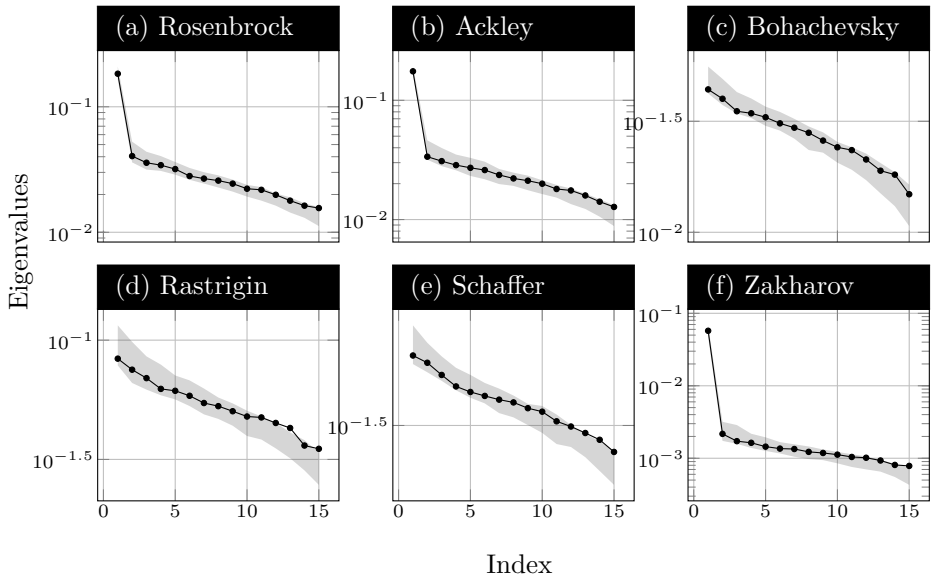


FIG. 7. *Eigenvalues estimates of the matrix* $\mathbf{C}$ *in* (3.1) *for all the benchmarks, at the first generation, for* $d = 15$. *The black dots in the plot indicate the eigenvalues, while the grey area is defined by the bootstrap intervals.*

Since for all the tests the ASGA approach performs better than its classical counterpart despite the absence, in some cases, of an evident spectral gap in the AS covariance matrix, we perform further investigations. In particular, we use the same tests as before (15-dimensional functions) but with a different number of active dimensions, i.e., $M = \{2, 5\}$, instead of $M = 1$. In Figure 8, we show the comparison between the classical GA and the ASGA outcomes. It is possible to note that by increasing the active dimension, the differences between the performances of the two methods become smaller. Only for the Rosenbrock and for the Ackley functions can we see that ASGA with $M = 5$ is not able to reach the same order of magnitude reached by GA (we remark that the original space has dimension $d = 15$).

Increasing the input dimension to $d = 40$ shows a much clearer benefit in using the proposed method, as we can see in Figure 9. Here we set $N_0 = 5000$ and $N = 1000$. We specify that we set the active dimension $M = 1$. Also with this dimensionality, we

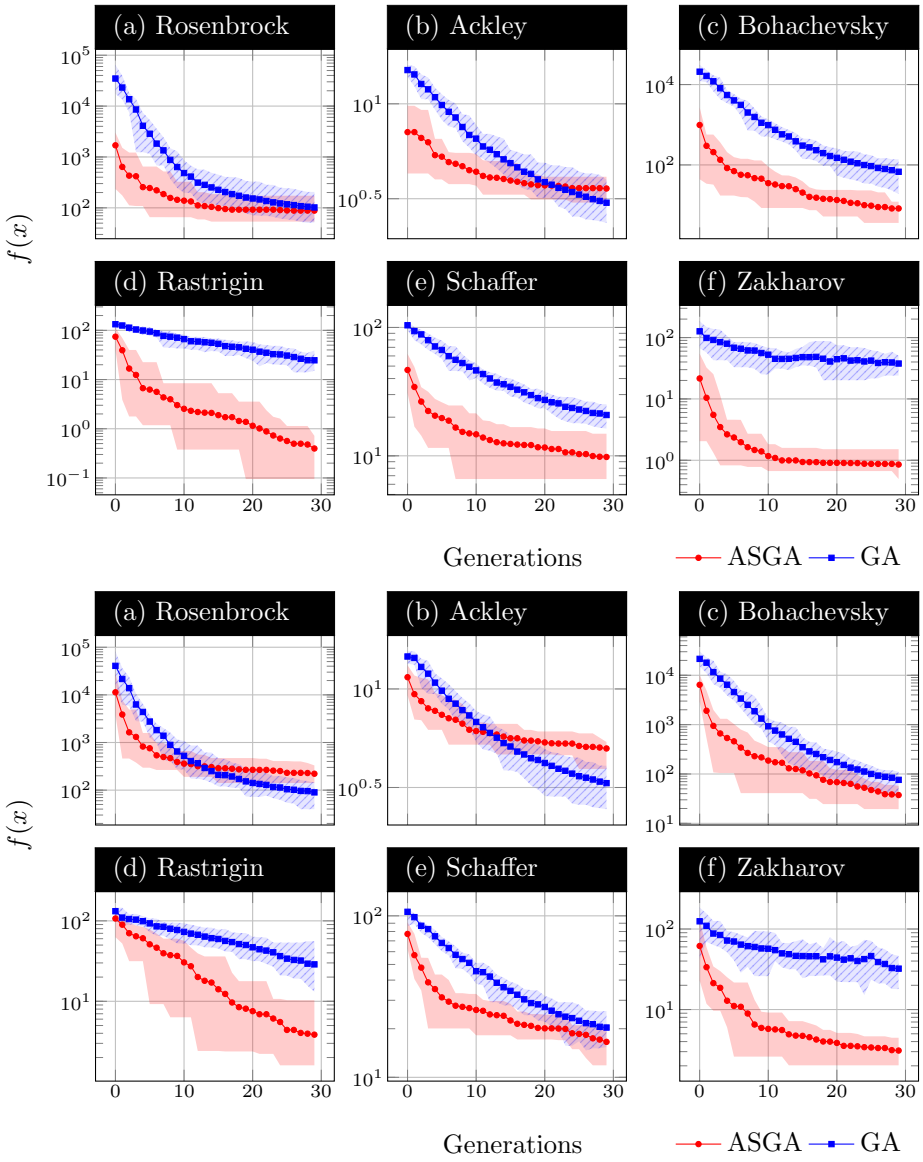


FIG. 8. *Results of the optimization of the benchmark functions in a space of dimension* $d = 15$*, with active dimensions* $M = 2$ *(top) and* $M = 5$ *(bottom). We compare the standard GA (in blue square dots) with the proposed algorithm ASGA (in red circle dots) using an initial population of size* 2000*, while the dimension for each generation is fixed to* 200*. The solid lines represent the mean, over* 15 *runs, of the objective function corresponding to the best individual at each generation. The shaded areas show the interval between minimum and maximum (blue with lines for GA and red for ASGA). Color is available online only.*

are able to isolate two main behaviors in the convergence of the six benchmarks: a very steep trend in the first generation, and a more smooth one, but still equally effective. The interesting thing is that some benchmarks do not reflect the behavior collected with $d = 15$. While *Rosenbrock*, *Rastrigin*, and *Zakharov* show a similar convergence rate for ASGA, the other benchmarks present a change in the slope. The different

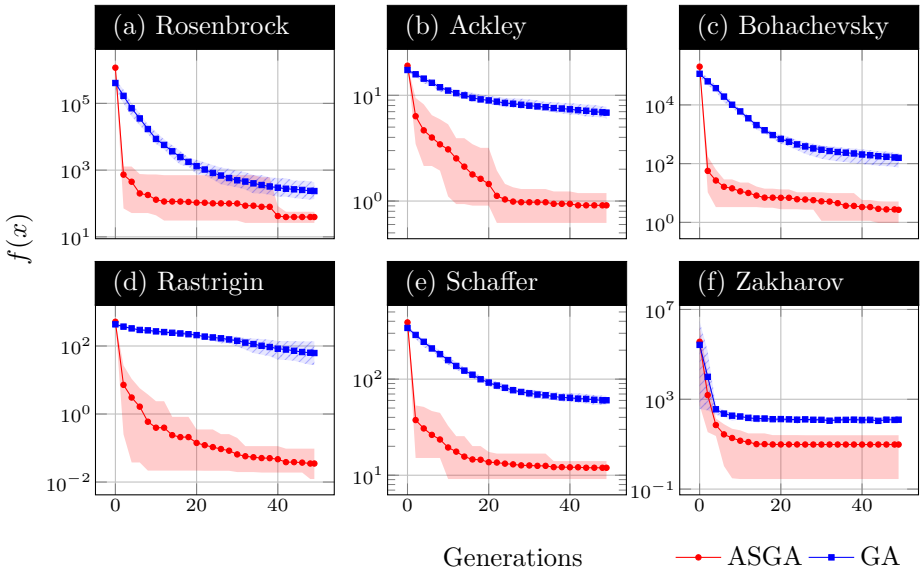


FIG. 9. *Results of the optimization of the benchmark functions in a space of dimension* $d = 40$. *We compare the standard GA (in blue square dots) with the proposed algorithm ASGA (in red circle dots) using an initial population of size* 5000*, while the dimension for each generation is fixed to* 1000. *The solid lines represent the mean, over* 15 *runs, of the objective function corresponding to the best individual at each generation. The shaded areas show the interval between minimum and maximum (blue with lines for GA and red for ASGA). Color is available online only.*

behaviors observed for the same benchmarks evaluated at different input dimensional spaces is due to the fact that the method is sensitive to the approximation accuracy of the gradients of the model function with respect to the input data. This is an issue inherited by the application of AS. Moreover, since we are keeping just one active variable, we are discarding several pieces of information, and thus the representation of the function along the active subspace could present some noise. So the genetic procedure enhanced by AS is able to converge fast to the optimum, but this optimum may be—for the space simplification—distant to the true optimum. From the tests with higher active dimensions, we note that the improvement in the first iterations is not as rapid as by using a one-dimensional AS. Also, keeping more active dimensions, the performance of ASGA becomes similar to the standard GA. We can conclude that with one (or a few) active dimension, ASGA reaches the global minimum with fewer function evaluations, but we get stuck with the projection error introduced by AS, whereas by increasing the active dimension we reduce the projection error but we lose the effectiveness of the evolution steps in a reduced space. A possible solution for this problem can be a smarter (and dynamical) strategy to select the number of active dimensions.

Over all of the three test cases, where we vary the input space dimension, the performance of ASGA is better than or equal to the standard GA. In Table 1, we summarize the relative gain on average achieved after the entire evolution and after only one generation, divided by the test function, both with the GA and ASGA methods. The relative gain is computed as the mean over 15 runs of the ratio between the objective function evaluated at the best-fit individual at the beginning of the evolution ($f(x^0_{\text{opt}})$) and the objective function evaluated at the best-fit individual

after k generations, with $k = 1$ and $k = N_{\text{gen}}$, where N_{gen} is the maximum number of generations depending on the input dimension. This relative gain $G(k)$ reads as

$$G(k) = \frac{1}{15} \sum_{i=1}^{15} f(x^0_{\text{opt}_i}) / f(x^k_{\text{opt}_i}), \tag{5.9}$$

where $x^k_{\text{opt}_i}$ is the best-fit individual of the population at the ith run and at the kth generation. The highest values correspond to a more effective optimization, and for dimensions 15 and 40 we can see from Table 1 that ASGA performs better than standard GA for all the benchmarks. Even the gain after just one evolutive iteration is bigger in all the collected tests, reaching in some cases some order of magnitude of difference with respect to GA. These results suggest that despite the computational overhead for the construction of AS and the back-mapping, an application of ASGA over the standard GA produces usually better or at least comparable results for a fixed generation.

TABLE 1
Summary comparison between GA and ASGA with respect to the gain $G(k)$ defined in (5.9). *We compare the gain for the first and last generations.*

Function	Method	dim = 2		dim = 15		dim = 40	
		$G(9)$	$G(1)$	$G(29)$	$G(1)$	$G(49)$	$G(1)$
Ackley	GA	9.17	1.13	4.71	1.03	2.53	1.10
	ASGA	2.93	1.29	5.81	3.89	20.91	3.00
Bohachevsky	GA	39.58	1.78	223.86	1.22	729.05	1.81
	ASGA	31.66	2.04	8608.41	130.72	75104.33	3548.70
Rastrigin	GA	7.34	1.41	3.80	1.05	6.97	1.17
	ASGA	3.24	1.39	1343.41	4.00	14738.40	71.77
Rosenbrock	GA	30.04	1.74	335.33	1.34	1723.89	2.42
	ASGA	39.68	2.66	2343.57	167.48	29747.56	1600.24
Schaffer	GA	3.64	1.21	4.83	1.11	5.66	1.18
	ASGA	2.16	1.17	16.41	3.61	32.57	10.38
Zakharov	GA	38.59	2.65	3.11	1.07	2148.39	26.50
	ASGA	51.14	3.88	417.86	24.46	37739.61	237.48

5.2. Shape design optimization of a NACA airfoil. Here we present the shape design optimization of a NACA 4412 airfoil [1]. Since the purpose of this work is limited to the extension of GA, we briefly present the details of the complete model, with a quick overview of the application. To reproduce the full order simulations, refer to [51].

Let there be given the unsteady incompressible Navier–Stokes equations described in an Eulerian framework on a parametrized space-time domain $Q(\boldsymbol{\mu}) = \Omega \times [0, T] \subset \mathbb{R}^d \times \mathbb{R}^+$, $d = 2, 3$, with the velocity field denoted by $\mathbf{u} : Q(\boldsymbol{\mu}) \to \mathbb{R}^d$, and the pressure field by $p : Q(\boldsymbol{\mu}) \to \mathbb{R}$, such that

$$\begin{cases} \mathbf{u_t} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) - \nabla \cdot 2\nu \nabla^{\mathbf{s}} \mathbf{u} = -\nabla p & \text{in } Q(\boldsymbol{\mu}), \\ \nabla \cdot \mathbf{u} = \mathbf{0} & \text{in } Q(\boldsymbol{\mu}), \\ \mathbf{u}(t, x) = \mathbf{f}(\mathbf{x}) & \text{on } \Gamma_{\text{in}} \times [0, T], \\ \mathbf{u}(t, x) = \mathbf{0} & \text{on } \Gamma_0(\boldsymbol{\mu}) \times [0, T], \\ (\nu \nabla \mathbf{u} - p\mathbf{I})\mathbf{n} = \mathbf{0} & \text{on } \Gamma_{\text{out}} \times [0, T], \\ \mathbf{u}(0, \mathbf{x}) = \mathbf{k}(\mathbf{x}) & \text{in } Q(\boldsymbol{\mu})_0 \end{cases} \tag{5.10}$$

holds. Here we have that $\Gamma = \Gamma_{\text{in}} \cup \Gamma_0 \cup \Gamma_{\text{out}}$ is the boundary of Ω and it is composed by the inlet boundary Γ_{in}, the outlet boundary Γ_{out}, and the physical walls $\Gamma_0(\boldsymbol{\mu})$. The term $\mathbf{f}(\mathbf{x})$ stands for the stationary nonhomogeneous boundary condition, whereas $\mathbf{k}(\mathbf{x})$ indicates the initial condition for the velocity at $t = 0$. Shape changes are applied to the boundary $\Gamma_0(\boldsymbol{\mu})$ corresponding to the airfoil wall, which in the undeformed configuration corresponds to the 4-digit, NACA 4412 wing profile. Such shape modifications are associated to numerical parameters contained in the vector $\boldsymbol{\mu} \in \mathbb{R}^k$ with $k = 10$.

For a geometrical deformation map $\mathcal{M}$, we adopt the shape morphing proposed in [27], where five shape functions r_i are added to the upper and lower parts of the airfoil profile, denoted by y^+ and y^-, respectively. Each shape function is multiplied by a possible different coefficient as in the following:

$$y^+ = \overline{y^+} + \sum_{i=1}^{5} a_i r_i, \qquad y^- = \overline{y^-} - \sum_{i=1}^{5} b_i r_i, \tag{5.11}$$

where the bar denotes the reference undeformed profile. These 10 coefficients (a_i and b_i) represent the input parameters $\boldsymbol{\mu} \in \mathbb{D} := [0, 0.03]^{10}$. In Figure 10, we depict the NACA 4412 together with the five rescaled shape functions r_i. The output function we want to maximize is the lift-to-drag coefficient, one of the typical quantities of interest in aeronautical problems. To recast the problem in a minimization setting, we just minimize the opposite of the coefficient. To compute it, we model a turbulent flow passing around the two-dimensional airfoil using the incompressible Reynolds averaged Navier–Stokes equations. Regarding the main numerical settings, we adopt a finite volume approach with the Spalart–Allmaras model, with a computational grid of 46500 degrees of freedom. The flow velocity, at the inlet boundary, is set to 1 m/s, while the Reynolds number is fixed to 50000. For the detailed problem formulation, we refer the reader to the experiments conducted in [51].

Instead of running the high-fidelity solver for any new untested parameter, we optimize a radial basis function (RBF) response surface built using the initial dataset. Due to the stochastic nature of the method, also in this test case we test the methods for several initial settings—25 different runs—making the total computational load very high. Thus, we decided to build a response surface using a dataset of 333 samples, computed with the numerical scheme described above, mimicking at the same time a typical industrial workflow.

The objective function $f_{obj}(\boldsymbol{\mu}) : \mathbb{D} \subset \mathbb{R}^{10} \to \mathbb{R}$ we are going to minimize is the following:

$$f_{obj}(\boldsymbol{\mu}) = \begin{cases} g(\boldsymbol{\mu}) & \text{if } \boldsymbol{\mu} \in \mathbb{D}, \\ \alpha & \text{if } \boldsymbol{\mu} \notin \mathbb{D}, \end{cases} \tag{5.12}$$

where $g(\boldsymbol{\mu})$ is the response surface built using the RBF interpolation technique [9] over the samples, while $\alpha \in \mathbb{R}$ is a penalty constant. To prevent the evolution from creating new individuals that do not belong to $\mathbb{D}$, we impose a penalty factor $\alpha = 10$. We recall that we minimize the opposite of the lift-to-drag coefficient.

Figure 11 reports the evolution of the best-fit individual over 10 generations. Also in this case, we apply the proposed algorithm and the standard GA to 25 different initial settings, using an initial population size $N_0 = 20$ and selecting at each generation the $N = 10$ best-fit individuals for the offspring. The plot depicts the mean best-fit individual with solid lines, whereas the shaded areas show the interval between the

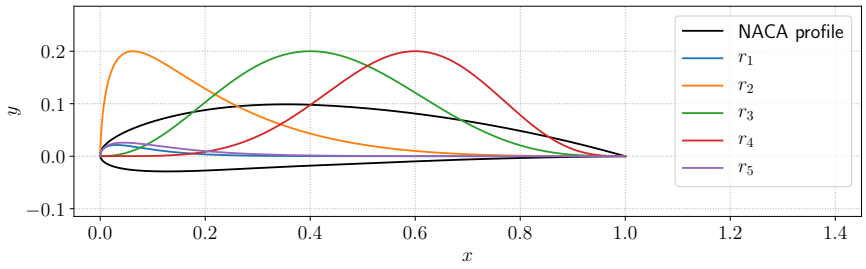


FIG. 10. *NACA* 4412 *profile with the five shape functions* r_i *rescaled by a factor equal to* 0.2.

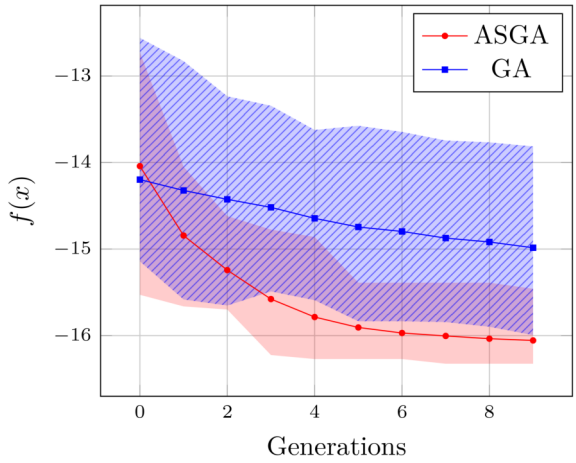


FIG. 11. *Results of the optimization of the NACA airfoil design in a* 10*-dimensional space. We compare the standard GA (in blue square dots) with the proposed algorithm ASGA (in red circle dots) using an initial population of size* 20*, while the dimension for each generation is fixed to* 10*. The solid lines represent the mean, over* 25 *runs, of the objective function corresponding to the best individual at each generation. The shaded areas show the interval between minimum and maximum (blue with lines for GA and red for ASGA). Color is available only only.*

minimum and maximum (of the 25 runs) for each generation. Even if the dimension of the parameter space is not very high (10), we can see that on average the proposed algorithm is able to converge faster. The difference between the two methods is not as remarkable as in a higher-dimensional test case, but we can see that the best run using standard GA is slightly worse than the mean optimum achieved by ASGA. This again demonstrates the value in the proposed method. Moreover, we emphasize that also in this case the decay of the objective function in the first generations with ASGA is faster.

6. Conclusions. In this work, we have presented a novel approach for optimization problems coupling the supervised learning technique called active subspaces (AS) with the standard genetic algorithm (GA). We have demonstrated the benefits of such a method by applying it to some benchmark functions and to a realistic engineering problem. The proposed method achieves faster convergence to the optimum since the individuals evolve only along few principal directions (discovered exploiting the AS property). Further, from the results it emerges that the gain induced from the ASGA method is greater for high-dimensionality functions, making it particularly suited for

models with many input parameters.

This new method can also be integrated in numerical pipelines involving model order reduction and reduction in parameter space. Reducing the number of input parameters is a key ingredient to improve the computational performance and to allow the study of very complex systems.

Since the number of active dimensions is important for the accuracy of AS, future developments will focus on an efficient criterion to select dynamically the number of AS dimensions, which in the presented results are kept fixed. Future studies will also address the problem of incorporating nonlinear extensions of active subspaces into ASGA, focusing on the construction of a proper back-mapping from the reduced space to the original full parameter space.

Appendix A. On ASGA convergence. The aim of this section is to provide further insights about the convergence of the ASGA method. We perform a single run on all the benchmark functions presented above in a space of dimension $d = 2$. We kept unaltered all the ASGA numerical settings described in section 5, so for all the details we refer the reader to that section. We emphasize that we used the same hyperparameters of the two-dimensional optimization test, except for the number of generations, which we increased to 100.

We summarize in Figures 12 and 13 the spatial coordinates of the best individual after each generation using the standard GA and ASGA. The proposed method reaches the global minimum for all the test cases, performing better than the standard counterpart for the Rosenbrock and Rastrigin functions.

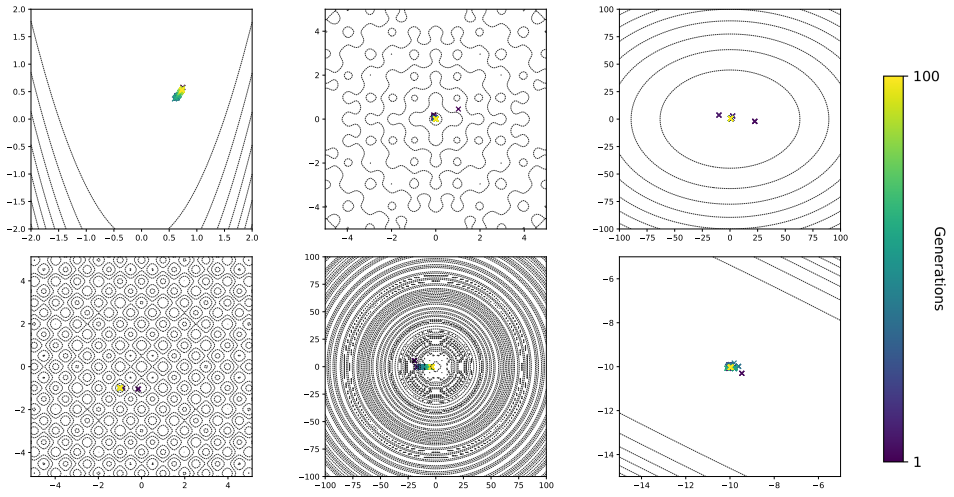


FIG. 12. *Results of the single optimization run using GA for the two-dimensional benchmark functions. The colored crosses indicate the spatial coordinates of the best individual at each generation. Black lines indicate the isolines of the functions. Color is available online only.*

We also measure the convergence as the Euclidean distance between the best individual fitness and the global optimum, and the *spatial* convergence as the Euclidean distance between the coordinates of the best individual and the coordinates of the optimal point. We kept the same numerical settings, only raising the number of generation to 1000. Figure 14 presents the plots where we compare the trend using GA and ASGA: the proposed method shows a better performance, not only thanks to the faster convergence but also because in all the cases ASGA is able to get closer than

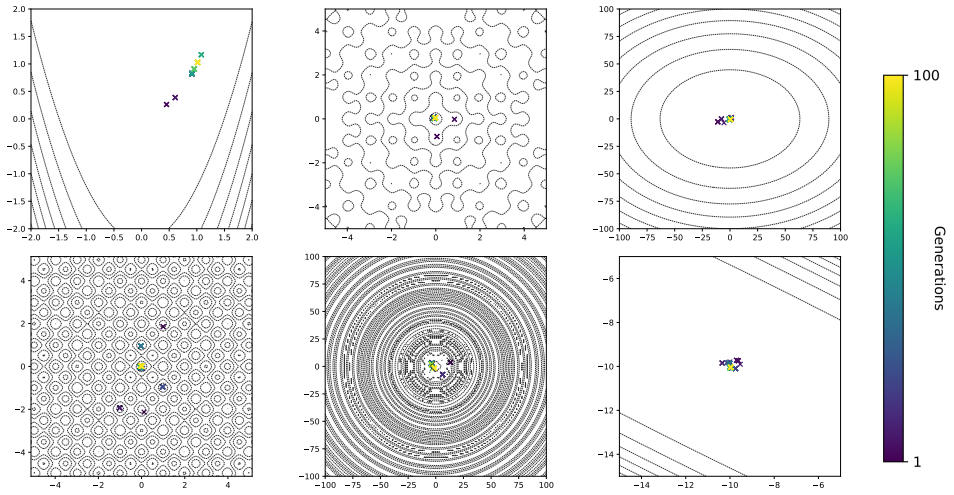


FIG. 13. *Results of the single optimization run using ASGA for the 2-dimensional benchmark functions. The colored crosses indicate the spatial coordinates of the best individual at each generation. Black lines indicate the isolines of the functions. Color is available online only.*

GA to the global optimum.

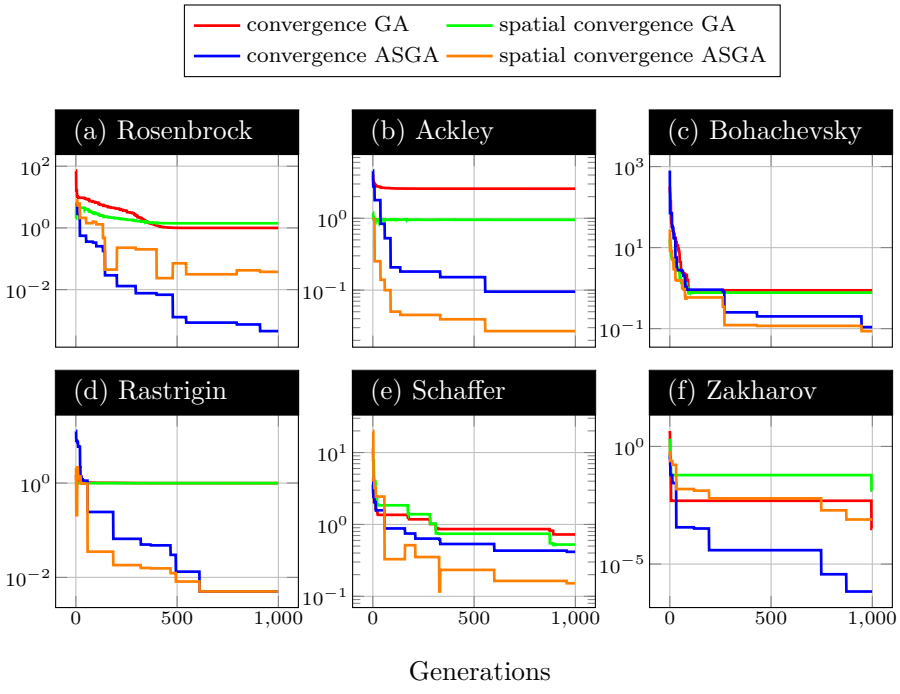


FIG. 14. *Convergence of GA and ASGA for the two-dimensional benchmark functions.*

Acknowledgment. We thank Francesco Romor for the productive discussions and comments.

REFERENCES

[1] I. H. Abbott and A. E. Von Doenhoff, *Theory of Wing Sections: Including a Summary of Airfoil Data*, Courier Corporation, North Chelmsford, MA, 2012.

[2] E. P. Adorio and U. Diliman, *Mvf-multivariate test functions library in C for unconstrained global optimization*, Quezon City, Metro Manila, Philippines, 2005, pp. 100–104.

[3] M. M. Ali, C. Khompatraporn, and Z. B. Zabinsky, *A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems*, J. Global Optim., 31 (2005), pp. 635–672.

[4] T. Back, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*, Oxford University Press, New York, 1996.

[5] J. Bect, D. Ginsbourger, L. Li, V. Picheny, and E. Vazquez, *Sequential design of computer experiments for the estimation of a probability of failure*, Stat. Comput., 22 (2012), pp. 773–793.

[6] C. J. Bélisle, H. E. Romeijn, and R. L. Smith, *Hit-and-run algorithms for generating multivariate distributions*, Math. Oper. Res., 18 (1993), pp. 255–266.

[7] N. Botkin and V. Turova-Botkina, *An algorithm for finding the Chebyshev center of a convex polyhedron*, Appl. Math. Optim., 29 (1994), pp. 211–222.

[8] R. A. Bridges, A. D. Gruber, C. R. Felder, M. Verma, and C. Hoff, *Active Manifolds: A non-linear analogue to Active Subspaces*, in Proceedings of the 36th International Conference on Machine Learning (ICML), Long Beach, CA, 2019, pp. 764–772.

[9] M. D. Buhmann, *Radial Basis Functions: Theory and Implementations*, Cambridge Monogr. Appl. Comput. Math. 12, Cambridge University Press, Cambridge, UK, 2003.

[10] C. Cartis and A. Otemissov, *A Dimensionality Reduction Technique for Unconstrained Global Optimization of Functions with Low Effective dimensionality*, preprint, https://arxiv.org/abs/2003.09673, 2020.

[11] R. Chelouah and P. Siarry, *A continuous genetic algorithm designed for the global optimization of multimodal functions*, J. Heuristics, 6 (2000), pp. 191–213.

[12] K. M. Choromanski, A. Pacchiano, J. Parker-Holder, Y. Tang, and V. Sindhwani, *From complexity to simplicity: Adaptive ES-active subspaces for blackbox optimization*, in Advances in Neural Information Processing Systems, MIT Press, Cambridge, MA, 2019, pp. 10299–10309.

[13] P. G. Constantine, *Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies*, SIAM Spotlights 2, SIAM, Philadelphia, 2015, https://doi.org/10.1137/1.9781611973860.

[14] P. G. Constantine, E. Dow, and Q. Wang, *Active subspace methods in theory and practice: Applications to kriging surfaces*, SIAM J. Sci. Comput., 36 (2014), pp. A1500–A1524, https://doi.org/10.1137/130916138.

[15] P. G. Constantine, A. Eftekhari, and M. B. Wakin, *Computing active subspaces efficiently with gradient sketching*, in Proceedings of the 6th International IEEE Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2015, pp. 353–356.

[16] C. Cui, K. Zhang, T. Daulbaev, J. Gusak, I. Oseledets, and Z. Zhang, *Active subspace of neural networks: Structural analysis and universal attacks*, SIAM J. Math. Data Sci., 2 (2020), pp. 1096–1122, https://doi.org/10.1137/19M1296070.

[17] J. P. Cunningham and Z. Ghahramani, *Linear dimensionality reduction: Survey, insights, and generalizations*, J. Mach. Learn. Res., 16 (2015), pp. 2859–2900.

[18] N. Demo, M. Tezzele, A. Mola, and G. Rozza, *Hull shape design optimization with parameter space and model reductions, and self-learning mesh morphing*, J. Mar. Sci. Eng., 9 (2021), 185, https://doi.org/10.3390/jmse9020185.

[19] N. Demo, M. Tezzele, and G. Rozza, *A non-intrusive approach for reconstruction of POD modal coefficients through active subspaces*, Comptes Rendus Mécanique de l'Académie des Sciences, DataBEST 2019 Special Issue, 347 (2019), pp. 873–881, https://doi.org/10.1016/j.crme.2019.11.012.

[20] L. C. W. Dixon and G. P. Szego, *The global optimization problem: An introduction*, in Towards Global Optimisation, North-Holland, Amsterdam, 1978, pp. 1–15.

[21] Z. Drezner, *A new genetic algorithm for the quadratic assignment problem*, INFORMS J. Comput., 15 (2003), pp. 320–330.

[22] T. A. El-Mihoub, A. A. Hopgood, L. Nolle, and A. Battersby, *Hybrid genetic algorithms: A review*, Eng. Lett., 13 (2006), pp. 124–137.

[23] S. M. Elsayed, R. A. Sarker, and D. L. Essam, *A new genetic algorithm for solving optimization problems*, Eng. Appl. Artif. Intell., 27 (2014), pp. 57–69.

[24] C. Emmeche, *The Garden in the Machine: The Emerging Science of Artificial Life*, Princeton

Sci. Lib. 17, Princeton University Press, Princeton, NJ, 1996.
[25] L. J. Eshelman and J. D. Schaffer, *Real-coded genetic algorithms and interval-schemata*, in Foundations of Genetic Algorithms, Vol. 2, Elsevier, Amsterdam, 1993, pp. 187–202.
[26] M. Guo and J. S. Hesthaven, *Reduced order modeling for nonlinear structural analysis using Gaussian process regression*, Comput. Methods Appl. Mech. Engrg., 341 (2018), pp. 807–826.
[27] R. M. Hicks and P. A. Henne, *Wing design by numerical optimization*, J. Aircr., 15 (1978), pp. 407–412.
[28] R. Hinterding, *Gaussian mutation and self-adaption for numeric genetic algorithms*, in Proceedings of the IEEE International Conference on Evolutionary Computation, Vol. 1, 1995, pp. 384–389.
[29] J. H. Holland, *Genetic algorithms and the optimal allocation of trials*, SIAM J. Comput., 2 (1973), pp. 88–105, https://doi.org/10.1137/0202009.
[30] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, MIT Press, Cambridge, MA, 1992.
[31] M. Jamil and X.-S. Yang, *A literature survey of benchmark functions for global optimisation problems*, Int. J. Math. Model. Numer. Optim., 4 (2013), pp. 150–194, https://doi.org/10.1504/IJMMNO.2013.055204.
[32] M. Kumar, M. Husain, N. Upreti, and D. Gupta, *Genetic Algorithm: Review and Application*, https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3529843, 2010.
[33] M. Laguna and R. Martí, *Experimental testing of advanced scatter search designs for global optimization of multimodal functions*, J. Global Optim., 33 (2005), pp. 235–255.
[34] L. Lovász and S. Vempala, *Hit-and-run from a corner*, SIAM J. Comput., 35 (2006), pp. 985–1005, https://doi.org/10.1137/S009753970544727X.
[35] T. W. Lukaczyk, P. Constantine, F. Palacios, and J. J. Alonso, *Active subspaces for shape optimization*, in Proceedings of the 10th AIAA Multidisciplinary Design Optimization Conference, 2014, 1171.
[36] H. Mühlenbein, M. Schomisch, and J. Born, *The parallel genetic algorithm as function optimizer*, Parallel Comput., 17 (1991), pp. 619–632.
[37] Y. Nesterov and V. Spokoiny, *Random gradient-free minimization of convex functions*, Found. Comput. Math., 17 (2017), pp. 527–566.
[38] V. Picheny, T. Wagner, and D. Ginsbourger, *A benchmark of kriging-based infill criteria for noisy optimization*, Struct. Multidiscip. Optim., 48 (2013), pp. 607–626.
[39] H. Qian, Y.-Q. Hu, and Y. Yu, *Derivative-free optimization of high-dimensional non-convex functions by sequential random embeddings*, in Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI), 2016, pp. 1946–1952.
[40] F. Romor, M. Tezzele, A. Lario, and G. Rozza, *Kernel-Based Active Subspaces with Application to CFD Parametric Problems Using Discontinuous Galerkin Method*, preprint, https://arxiv.org/abs/2008.12083, 2020.
[41] F. Romor, M. Tezzele, and G. Rozza, *ATHENA: Advanced Techniques for High Dimensional Parameter Spaces to Enhance Numerical Analysis*, preprint, https://arxiv.org/abs/2105.06713, 2020.
[42] G. Rozza, M. Hess, G. Stabile, M. Tezzele, and F. Ballarin, *Basic ideas and tools for projection-based model reduction of parametric partial differential equations*, in Model Order Reduction, Vol. 2, P. Benner, S. Grivet-Talocia, A. Quarteroni, G. Rozza, W. H. A. Schilders, and L. M. Silveira, eds., De Gruyter, Berlin, Boston, 2020, pp. 1–47, https://doi.org/10.1515/9783110671490-001.
[43] M. L. Sanyang and A. Kabán, *REMEDA: Random embedding EDA for optimising functions with intrinsic dimension*, in Parallel Problem Solving from Nature—PPSN XIV, Springer, Cham, 2016, pp. 859–868.
[44] J. D. Schaffer, R. Caruana, L. J. Eshelman, and R. Das, *A study of control parameters affecting online performance of genetic algorithms for function optimization*, in Proceedings of the 3rd International Conference on Genetic Algorithms, 1989, pp. 51–60.
[45] R. Sivaraj and T. Ravichandran, *A review of selection methods in genetic algorithm*, Int. J. Eng. Sci. Technol., 3 (2011), pp. 3792–3797.
[46] R. L. Smith, *The hit-and-run sampler: A globally reaching Markov chain sampler for generating arbitrary multivariate distributions*, in Proceedings of the IEEE Winter Simulation Conference, 1996, pp. 260–264.
[47] M. Tezzele, F. Ballarin, and G. Rozza, *Combined parameter and model reduction of cardiovascular problems by means of active subspaces and POD-Galerkin methods*, in Mathematical and Numerical Modeling of the Cardiovascular System and Applications, SEMA

SIMAI Springer Ser. 16, D. Boffi, L. F. Pavarino, G. Rozza, S. Scacchi, and C. Vergara, eds., Springer, Cham, 2018, pp. 185–207, https://doi.org/10.1007/978-3-319-96649-6_8.

[48] M. Tezzele, N. Demo, M. Gadalla, A. Mola, and G. Rozza, *Model order reduction by means of active subspaces and dynamic mode decomposition for parametric hull shape design hydrodynamics*, in Technology and Science for the Ships of the Future: Proceedings of NAV 2018 and the 19th International Conference on Ship & Maritime Research, IOS Press, Amsterdam, 2018, pp. 569–576, https://doi.org/10.3233/978-1-61499-870-9-569.

[49] M. Tezzele, N. Demo, A. Mola, and G. Rozza, *An Integrated Data-Driven Computational Pipeline with Model Order Reduction for Industrial and Applied Mathematics*, Special Volume of ECMI, Springer, Berlin, Heidelberg; to appear.

[50] M. Tezzele, N. Demo, and G. Rozza, *Shape optimization through proper orthogonal decomposition with interpolation and dynamic mode decomposition enhanced by active subspaces*, in Proceedings of MARINE 2019: VIII International Conference on Computational Methods in Marine Engineering, R. Bensow and J. Ringsberg, eds., 2019, pp. 122–133.

[51] M. Tezzele, N. Demo, G. Stabile, A. Mola, and G. Rozza, *Enhancing CFD predictions in shape design problems by model and parameter space reduction*, Adv. Model. Simul. Eng. Sci. , 7 (2020), 40, https://doi.org/10.1186/s40323-020-00177-y.

[52] M. Tezzele, F. Salmoiraghi, A. Mola, and G. Rozza, *Dimension reduction in heterogeneous parametric spaces with application to naval engineering shape design problems*, Adv. Model. Simul. Eng. Sci., 5 (2018), 25, https://doi.org/10.1186/s40323-018-0118-3.

[53] Z. Wang, F. Hutter, M. Zoghi, D. Matheson, and N. de Feitas, *Bayesian optimization in a billion dimensions via random embeddings*, J. Artif. Intell. Res., 55 (2016), pp. 361–387.

[54] O. Zahm, P. G. Constantine, C. Prieur, and Y. M. Marzouk, *Gradient-based dimension reduction of multivariate vector-valued functions*, SIAM J. Sci. Comput., 42 (2020), pp. A534–A558, https://doi.org/10.1137/18M1221837.

[55] G. Zhang, J. Zhang, and J. Hinkle, *Learning nonlinear level sets for dimensionality reduction in function approximation*, in Advances in Neural Information Processing Systems, MIT Press, Cambridge, MA, 2019, pp. 13199–13208.