EPJ Quantum Technology
a SpringerOpen Journal

RESEARCH Open Access

Trainability maximization using estimation of distribution algorithms assisted by surrogate modelling for quantum architecture search

Vicente P. Soloviev[1*], Vedran Dunjko[2,3], Concha Bielza[1], Pedro Larrañaga[1] and Hao Wang[2,3]

*Correspondence:
vicente.perez.soloviev@fi.upm.es
[1]Computational Intelligence Group (CIG), Universidad Politécnica de Madrid, Madrid, Spain
Full list of author information is available at the end of the article

Abstract

Quantum architecture search (QAS) involves optimizing both the quantum parametric circuit configuration but also its parameters for a variational quantum algorithm. Thus, the problem is known to be multi-level as the performance of a given architecture is unknown until its parameters are tuned using classical routines. Moreover, the task becomes even more complicated since well-known trainability issues, e.g., barren plateaus (BPs), can occur. In this paper, we aim to achieve two improvements in QAS: (1) to reduce the number of measurements by an online surrogate model of the evaluation process that aggressively discards architectures of poor performance; (2) to avoid training the circuits when BPs are present. To detect the presence of the BPs, we employed a recently developed metric, information content, which only requires measuring the energy values of a small set of parameters to estimate the magnitude of cost function's gradient. The main idea of this proposal is to leverage a recently developed metric which can be used to detect the onset of vanishing gradients to ensure the overall search avoids such unfavorable regions. We experimentally validate our proposal for the variational quantum eigensolver and showcase that our algorithm is able to find solutions that have been previously proposed in the literature for the Hamiltonians; but also to outperform the state of the art when initializing the method from the set of architectures proposed in the literature. The results suggest that the proposed methodology could be used in environments where it is desired to improve the trainability of known architectures while maintaining good performance.

Keywords: Quantum architecture search; Evolutionary algorithm; Information content; Barren plateaus; Estimation of distribution algorithm; Multi-level optimization; Variational quantum eigensolver

1 Introduction

Variational quantum algorithms (VQAs) [1] have become prominent tools in the noisy intermediate-scale quantum (NISQ) era, where quantum computers face limitations due to noise and connectivity issues. A well-known example of this type of approaches is the variational quantum eigensolver (VQE) [2]. Its adaptability and ability to efficiently explore solution spaces make them valuable tools for quantum computation, offering promising applications in areas such as quantum chemistry [2], optimization [3], and machine learning [4, 5], despite the challenges presented by the NISQ era hardware.

VQAs employ (i) an objective cost function to be minimized, (ii) a quantum parametric circuit (henceforth called as *ansatz*), and (iii) a classical optimization technique that tunes the *ansatz*.

First, a Hamiltonian (H) is a quantum Hermitian operator that describes a physical system, yielding the energy of a quantum state, which is often used as the objective cost function to be minimized in VQAs. Finding the global minima of the Hamiltonian (ground energy) implies finding a ground state of the quantum system. Although the literature proposes other objective functions such as the conditional value at a risk [6], or the Gibbs objective function [7], the most widely used one is the expectation value, often simplified as,

$$\min_{\boldsymbol{\theta}} \quad \langle H \rangle_{U(\boldsymbol{\theta})}, \tag{1}$$

where $\boldsymbol{\theta}$ is the variational parameter, to be optimized classically, and $\langle H \rangle_{U(\boldsymbol{\theta})}$ describes the measurements of a quantum system as,

$$\langle H \rangle_{U(\boldsymbol{\theta})} = \langle 0 | \, U^T(\boldsymbol{\theta}) H U(\boldsymbol{\theta}) \, | 0 \rangle, \tag{2}$$

where $U(\boldsymbol{\theta})$ is the unitary state generated by an *ansatz*, parameterized by $\boldsymbol{\theta} \in [0, 2\pi]^d$, where d is the number of parameters.

Second, an *ansatz* is a quantum circuit which is parameterized by a set of parameters $\boldsymbol{\theta}$, and its quantum state is denoted as,

$$|\Psi(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta}) \, |\Psi_0\rangle, \tag{3}$$

where $|\Psi_0\rangle$ is the given initial state, typically set to the $|0\rangle$ state, i.e., $|00\cdots 0\rangle^{\otimes n}$ state, where n is the number of qubits of the system.

The *ansatz* found in the literature are traditionally classified into *problem-inspired* or *hardware-efficient*, depending on its design [1]. The former considers the intrinsic physics of the problem to be solved for its design, and it has been shown to achieve good performance in terms of quality and convergence. An example is the quantum approximate optimization algorithm [8]. However, the latter proposes *ansatzes* that fit to the hardware limitations underlining a quantum device, i.e., available quantum gates or quantum connectivity.

Third, the overall performance of the VQA heavily depends on both, *ansatz* selection and the parameter optimization. Thus, the literature proposes a wide range of approaches to tune the parameters, which are typically classified into *gradient-based* or *gradient-free optimizers*. Some examples of the former include gradient descent [9] and limited Broyden-Fletcher-Goldfarb-Shanno [10]; while some examples of the latter include evolutionary algorithms (EAs) [11, 12] and reinforcement learning [13], among others.

When choosing an *ansatz* for a problem and optimizing its parameters, we assume that the *ansatz* is expressive enough to converge to the ground state of our Hamiltonian. Finding the ideal *ansatz* for a given H but also the parameters $\boldsymbol{\theta}$ becomes a multi-level optimization problem [14] in which each proposed *ansatz* also involves a new optimization task regarding the parameters of the specific architecture. Some approaches are presented in the literature using heuristics, where most of them involve too many measurements, and therefore lead to an increase of the computational resources and time. This is crucial for the feasibility of the algorithm in NISQ devices as the number of available measurements is limited before the device is re-configured. Overcoming these limitations leads us to the quantum architecture search (QAS) research topic, where some authors have proposed different ideas. Further approaches regarding QAS are reviewed in Sect. 2.

The training/optimization of the variational parameters is known to be a non-trivial task for deep circuits, since we might face quite a few challenging trainability issues, e.g., BPs and traps [15]. BPs are typically described as vanishing gradients close to zero in the landscape, where the classical optimization becomes challenging, i.e., non-trainable or hard-to-train *ansatz*. Several works are found in the state of the art where this phenomenon is studied in order to analyze the trainability of the *ansatz* [16, 17]. However, computing these gradients involves the parameter optimization of the *ansatz*, and thus increasing the number of quantum simulations, as we need to estimate the variance of the partial derivatives over the entire parameter space (exponential complexity). These tasks becomes more difficult with the number of qubits. Recently, Pérez-Salinas et al. [18] have shown that the information content (IC) metric can reliably estimate the average (over the parameter space) norm of the gradient with a small number of evaluations of parameters of the *ansatz*.

In this paper we propose a domain-agnostic approach based on EAs in which, given a set of *ansatzes*, for which a good performance is expected, we seek to find a new set of *ansatzes* similar to the initial one, but which are easier to train, and therefore are more likely to avoid the presence of BPs. The number of quantum simulations are drastically reduced by implementing a surrogate model which predicts the performance of the *ansatz*, and the IC is used to maximize the trainability of the proposed architectures avoiding the presence of BPs. Experimental results are shown in noisy environments for different problems. Thus, the main contributions of the paper are:

- The use of surrogate models to rank the *ansatz* proposed by the EA without any measurements.
- The maximization of the trainability during the optimization process by using the IC.
- The use of multi-objective optimization to optimize the IC and the score provided by the surrogate model.

To the best of our knowledge this is the first work in which IC is optimized for quantum *ansatz* design, and we conjecture this approach can pave the way to bridging the gap towards an ideal training-free approach.

The rest of the paper is organized as follows. Section 2 reviews the QAS literature. In Sect. 3 we provide a theoretical background for evolutionary approaches, IC for the approximation of the average norm of the gradients, and surrogate modelling. The proposed methodology is presented in Sect. 4 and Sect. 5 shows some experimental results. Section 6 rounds the paper off with some further conclusions and future open research lines.

2 Related work

This section reviews some of the existing works regarding QAS in the literature.

Regarding reinforcement learning (RL), [19] uses a multi-level optimization process in which the agent proposes new architectures while a classical secondary optimizer tunes the parameters of the *ansatz*. In [20], a RL approach is proposed with a different purpose: given an *ansatz*, return an optimized structure in terms of circuit depth and used gates. A RL approach is proposed [21] where an agent systematically modifies the *ansatz* and achieves shallow circuits for chemical domains. More recently, a novel approach based in RL is proposed in [22] with competitive results.

Regarding EAs, [23] proposes a multi-level genetic algorithm where a multi-objective approach is used to minimize the energy of the VQE while minimizing the number of CNOT gates, and the parameter optimization is performed by CMA-ES optimizer. In [24] the authors use a genetic algorithm to optimize a weighted single-objective cost function combining the energy of the proposed *ansatz*, its depth, and number of two-qubit gates. Recently, GA4QCO framework [25] is proposed in which a single-objective optimization is performed by a genetic algorithm, and compared to random instances.

Regarding chemistry simulation, AdaptiveVQE [26] is a methodology that systematically grows an *ansatz* for chemical simulation; and RotoSelect and RotoSolve methods [27] are two efficient methods for jointly optimizing *ansatz* structure and parameters.

Several works are found in the literature in which neural architecture search methodologies are applied to QAS. QuantumDARTS [28] is an adaptation of classical DARTS [29] for neural network architecture search to QAS, in which two methods are proposed: one for whole architecture search, and another for promising sub-architectures. Another example is [30] in which new architectures are sampled from a probabilistic model, and gradients between the best energies found are computed.

Additionally, SuperNet structure [31], samples several architectures and its parameters are classically optimized. Based on the performance, the *ansatz* are ranked and a new architecture is constructed based on the knowledge gained from them. SuperNet has also been used to enhance VQAs on an 8-qubit superconducting quantum processor for classification tasks [32].

Our work is an EA which differs from the rest by using a multi-objective approach, reducing the complexity of the multi-level optimization task by using surrogate modeling and information content to evaluate the presence of BPs.

3 Background

3.1 Estimation of distribution algorithms

EAs are a class of optimization and search techniques inspired by the principles of natural selection and biological evolution. Rooted in the idea of survival of the fittest, these algorithms mimic the process of evolution to iteratively improve and evolve a population of candidate solutions to a problem. Traditional EAs rely on crossover and mutation operators, whereas, estimation of distribution algorithms (EDAs) [33] iteratively learn and sample unclear modelling what target probability distribution. EDAs have shown to be a power tool for optimization problems in which the number of variables to be optimized is big.

Algorithm 1 describes the baseline of EDA approaches. Given a population of size N, the ratio of the population $\alpha \in (0,1)$ to be promoted to next iteration, and the cost function $g(\cdot)$ to be optimized, the algorithm iteratively selects the top $\lfloor \alpha N \rfloor$ individuals from a set of solutions according to $g(\cdot)$ (lines 3-4), learns a probabilistic model (line 5) from these top individuals, and samples it to generate a new set of solutions (line 6). The algorithm iterates until a convergence criterion is met, and returns the best cost and solution found so far.

Algorithm 1 Estimation of distribution algorithms

Input: Population size N, selection ratio α, cost function g

Output: Best individual $\mathbf{x}'$ and cost found $g(\mathbf{x}')$

```
1: $G_0 \leftarrow N$ individuals randomly sampled or provided
2: for $t = 1, 2, \dots$ until stopping criterion is met do
3:     Evaluate $G_{t-1}$ according to $g(\cdot)$
4:     $G^S_{t-1} \leftarrow$ Select top $\lfloor \alpha N \rfloor$ individuals from $G_{t-1}$
5:     $p_{t-1} \leftarrow$ Learn a probabilistic model from $G^S_{t-1}$
6:     $G_t \leftarrow$ Sample $N$ individuals from $p_{t-1}(\cdot)$
7: end for
```

Regarding the type of probabilistic model, we can distinguish between *multivariate* EDAs and *univariate* EDAs. The former learns a joint probability distribution factorized with conditional probabilities over the variables involved in the problem. The latter learns a univariate probability distribution per variable in which no dependencies are considered, speeding up the computation and thus allowing to face bigger optimization problems, in terms of the number of variables.

Considering the set of random variables $\mathbf{X} = (X_1, X_2, \dots, X_d)$ involved in the problem, where d regards the dimension of the feature space, the joint probability distribution is approximated in the univariate EDAs as,

$$p(\mathbf{X}) = p(X_1, X_2, \dots, X_d) = \prod_{i=1}^{d} p(X_i), \tag{4}$$

where $p(X_i)$ is the marginal probability distribution of variable X_i. Note that computing the joint probability distribution of multivariate EDAs is much more costly, and thus in this approach we use univariate EDAs.

3.2 Information content for BPs diagnosis

BPs are traditionally described as exponentially vanishing gradients of the cost function where a classical optimizer is placed in a flat landscape, in which finding the global optimum becomes challenging. Avoiding this type of landscapes increases the probability of reaching better solutions. However, computing the gradients involves optimizing the *ansatz*, and thus, drastically increasing the number of quantum simulations.

Formally, BPs are characterized by the following properties,

$$\mathbb{E}_\theta(\partial_k E(\boldsymbol{\theta})) = 0, \tag{5}$$

$$\mathrm{Var}(\partial_k E(\boldsymbol{\theta})) \in \mathcal{O}(\exp(-n)), \tag{6}$$

where $\mathbb{E}(\partial_k E(\boldsymbol{\theta}))$, $k \in [1 \dots m]$, and $\mathrm{Var}(\partial_k E(\boldsymbol{\theta}))$ are the expectation and variance of the partial derivatives of the objective cost function, respectively, $\boldsymbol{\theta}$ is the set of parameters of the unitary representing the *ansatz*, and n is the number of qubits.

Recently, Pérez-Salinas et al. [18] have shown that the norm of the gradients can be bounded efficiently with a small number of quantum measurements (which grows linearly with the number of parameters), without the need of optimizing the *ansatz* parameters. This method performs a random walk in the parameter space and measures the entropy of fluctuations of cost values along the walk. The measured entropy value can be used to analytically bound the gradient of the cost function along the walk. We notice that the average of the gradient field (henceforth named as IC) can be approximated by the average along the random walk (due to Monte Carlo integration):

$$\| \nabla E \|^2 \approx \mathbb{E}_W \left(\sum_{k=1}^{m} (\partial_k E(\boldsymbol{\theta}))^2 \right) = \sum_{k=1}^{m} \mathrm{Var}_W(\partial_k E(\boldsymbol{\theta})), \tag{7}$$

where Var_W denotes the variance found in the objective cost function using m different $\boldsymbol{\theta}$ parameters generated from a random walk W. Note that this sampling is more efficient than estimating the gradients from random points.

Therefore, we propose to measure the IC metric for each candidate architecture, and maximize the IC value across the architecture search in addition to minimizing the cost value. This approach can help the architecture to generate more trainable circuits.

3.3 Surrogate modelling

Surrogate modelling is a common approach in machine learning for approximating the performance of an expensive computational task. Formally, we define a surrogate model as a function $h'(\mathbf{X})$ that approximates the output of $h(\mathbf{X})$, where $\mathbf{X} = (X_1, X_2, \ldots, X_d)$ is the input space with dimension d, and $h(\cdot)$ is a multivariate function that is time consuming to compute. The surrogate model $h'(\cdot)$ is formulated to provide a computationally efficient alternative and as a supervised approach it is constructed based on a set of observed data points $\mathcal{D} = \{(\mathbf{x}_i, h(\mathbf{x}_i))\}_{i=1}^{S}$, where $\mathbf{x}_i$ is an instance of the dataset with associated performance $h(\mathbf{x}_i)$, and S is the number of instances in the dataset.

3.4 Multi-objective optimization

Multi-objective optimization deals with the simultaneous optimization of multiple objectives $g_1(\mathbf{x}), \ldots, g_{n_o}(\mathbf{x})$ in the context of decision variables, where some of them might be conflicting. Multi-objective optimization aims at identifying a set of solutions that represent the trade-offs between different objectives, i.e. the best Pareto frontier approximation. Then, a multi-objective optimization problem is defined as

$$\begin{aligned} \max_{\mathbf{x}} \quad & G(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_{n_o}(\mathbf{x})) \\ & \text{subject to } \mathbf{x} \in \mathbb{R}^d \end{aligned} \tag{8}$$

where n_o and d are the number of objectives and variables involved in the problem, respectively, and the optimization criterion is maximization.

Quality indicators in multi-objective optimization are quantitative measures used to assess the performance and characteristics of solutions generated by multi-objective optimization algorithms [34]. An example is the hypervolume (HV) [35].

HV measures the volume of the objective space that is not dominated by a set of solutions. It quantifies how well a set of solutions covers the entire Pareto front. A higher hypervolume indicates a better spread of solutions. HV of a set $\mathbf{S}$, given a reference point

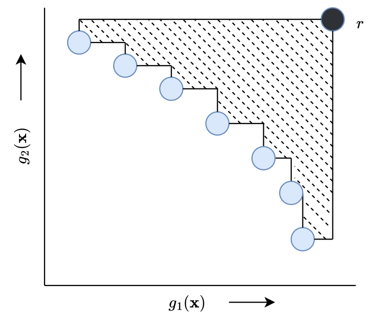


Figure 1 Example of Pareto frontier identified for two conflicting objectives. Reference point (**r**) assists the computation of the hypervolume metric

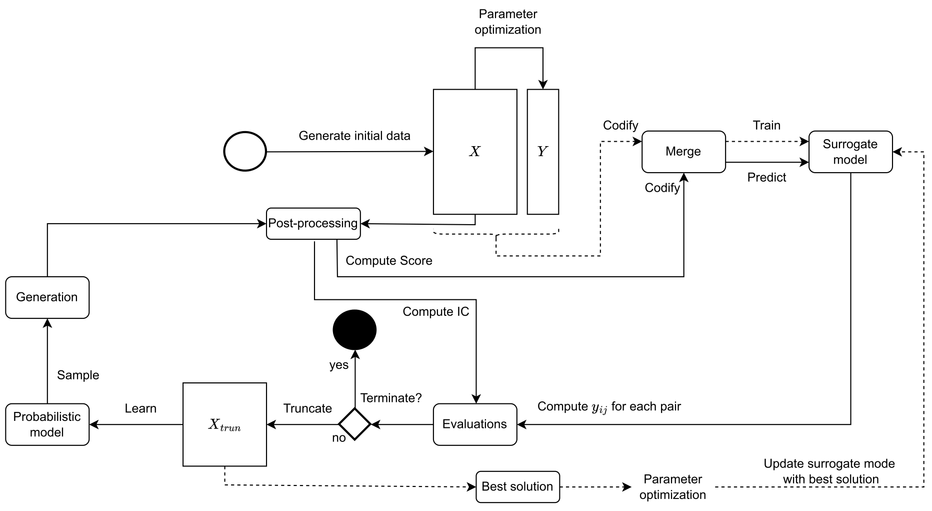


Figure 2 Flowchart of the proposed approach, starting from the white spot and finishing in the black spot one the convergence criteria is met. Dashed lines regard the train and update of the surrogate model

$\mathbf{r} = (g_1^{ref}, g_2^{ref}, \dots, g_{n_o}^{ref})$, is the volume of the union of the hypercubes determined by each of its solutions $\mathbf{s} \in \mathbf{S}$ and $\mathbf{r}$,

$$HV(\mathbf{S}, \mathbf{r}) = \Lambda(\bigcup_{\mathbf{S}} \{[g_1(\mathbf{s}), g_1^{ref}] \times \cdots \times [g_{n_o}(\mathbf{s}), g_{n_o}^{ref}]\}), \tag{9}$$

where g_i^{ref} refers to the reference ideal point for objective function g_i and $\Lambda(\cdot)$ refers to the Lebesgue measure. Figure 1 illustrates an example of the HV computation over the Pareto frontier for $n_o = 2$ objectives.

4 Method

This section explains the proposed approach and describes each of the modules in the following subsections. Figure 2 summarizes the flowchart of the approach where the main steps of the proposed algorithm are stated.

4.1 Codification

For an *ansatz* of n qubits and maximally depth m, we propose the following integer-valued matrix representation:

$$\mathbf{X} = \begin{bmatrix} X_{11} & \cdots & X_{1m} \\ \vdots & \ddots & \vdots \\ X_{n1} & \cdots & X_{nm} \end{bmatrix} \tag{10}$$
$$\to [X_{11}, \ldots, X_{1m}, \ldots, X_{n1}, \ldots, X_{nm}],$$

where each entry $X_{ij} \in \{0, 1, \ldots, n_{gates}\}$ represents the choice of the quantum logic gate at position (i,j) of the matrix. Given a predetermined number of qubits n and maximal depth m, the architecture representation has a fixed dimension $d = nm$. This way, each column represents all the operators executed in parallel along the total depth, and each row represents a qubit.

Note that regarding two-qubit gates such as CNOT, applying a CNOT with the same control qubit, but different target qubits, are considered as different gates. This allows to restrict the evolutionary search according to hardware constraints by restricting the search space, although in this work an all-to-all connectivity is considered. In our case, $n_{gates} = (n-1)+5$, as we consider the following universal operators: $\{Rx(\cdot), Ry(\cdot), Rz(\cdot), H, I\}$ and the CNOT gate with different target qubits. Note that CNOT(i,j) denotes that i and j are the control and target qubits, respectively. Therefore, we establish the following rules for the codification of X_{ij}:

- $0 \le X_{ij} < n$ and $X_{ij} \neq i$ corresponds to the CNOT gate configurations from qubit i to the possible target qubits.
- $X_{ij} = i$ corresponds to $Ry(\cdot)$.
- $X_{ij} = n$ corresponds to $Rz(\cdot)$.
- $X_{ij} = n + 1$ corresponds to H.
- $X_{ij} = n + 2$ corresponds to I gate.
- $X_{ij} = n + 3$ corresponds to $Rx(\cdot)$.

The initial state of all the proposed architectures is set to the $|0\rangle$ state, i.e., $|00\cdots0\rangle^{\otimes n}$ state.

Figure 3 shows four examples where the following codifications are represented as *ansatzes*,

$$\mathbf{A}_1 = \begin{bmatrix} 4\,0\,1\,3 \\ 4\,4\,5\,2 \\ 2\,2\,5\,5 \end{bmatrix}, \mathbf{A}_2 = \begin{bmatrix} 4\,0\,5\,3 \\ 4\,4\,0\,5 \\ 2\,2\,5\,1 \end{bmatrix}, \tag{11}$$
$$\mathbf{A}_3 = \begin{bmatrix} 4\,1\,5\,5 \\ 5\,5\,2\,5 \\ 5\,5\,5\,5 \end{bmatrix}, \mathbf{A}_4 = \begin{bmatrix} 1\,5\,5\,2 \\ 5\,2\,5\,5 \\ 5\,5\,0\,5 \end{bmatrix},$$

where $n = 3$ and $m = 4$.

4.2 Probabilistic model

The joint probability distribution factorizes in a univariate EDA approach according to Equation (4), where $p(X_{ij})$ is the marginal probability distribution of variable X_{ij}. In this

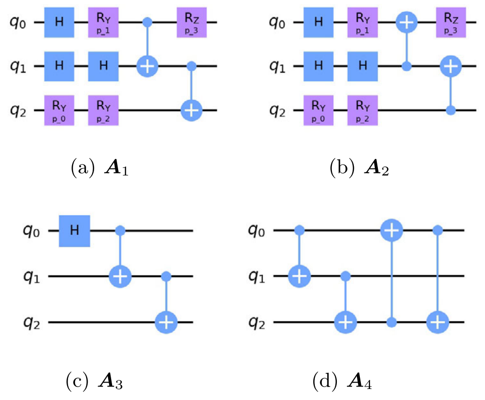


Figure 3 Four examples of the ansatz codifications (**A**$_1$, **A**$_2$, **A**$_3$, **A**$_4$) defined in Equation (11)

approach, $d = nm$, and $p(X_{ij})$ follows a multinomial distribution,

$$X_{ij} \sim \text{Mult}(n_m = \lfloor \alpha N \rfloor, k_m = (n_{gates} + 1)), \tag{12}$$

where n_m and k_m are the number of trials and mutually exclusive events that define the multinomial probability distribution, respectively.

Note that the marginal probabilities over the set of solutions are computed after the truncation process (Algorithm 1 Line 4), where the top $\lfloor \alpha N \rfloor$ solutions are selected according to the cost function to be optimized. The sampling process generates N new solutions as detailed in Algorithm 1, and duplicate *ansatz* are rejected in order to reduce redundancy. Each solution represents an *ansatz*, and the algorithm is expected to learn itself the best gates configuration during runtime.

4.3 Post-processing

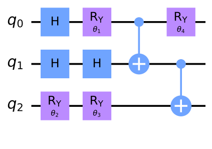
In order to restrict the search space of the QAS problem, we establish a series of hard rules to remove redundancy and simplify the *ansatz* architectures proposed in the sampling process of the EDA.

- Two consecutive H gates are removed, as they are equivalent to an I gate.
- Consecutive application of $Rx(\cdot)$ gates, are simplified as one single $Rx(\cdot)$ gate, to remove redundancy.
- Consecutive application of $Ry(\cdot)$ gates, are simplified as one single $Ry(\cdot)$ gate, to remove redundancy.
- Consecutive application of $Rz(\cdot)$ gates, are simplified as one single $Rz(\cdot)$ gate, to remove redundancy.

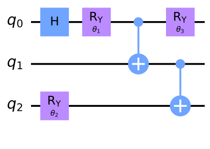
Once the algorithm samples a new set of architectures (Algorithm 1 Line 6), the post-processing step is applied to each of them. Figure 4 shows an example of the application of these hard rules, where (i) in the second qubit, both consecutive H gates were suppressed, and (ii) in the third qubit the two $Ry(\cdot)$ gates are simplified as a single gate.

4.4 Surrogate model

A characteristic of traditional EDAs is that once the solutions of the same population are ranked according to $g(\cdot)$, no matter how much better a solution is compared to others, as



(a) Original *ansatz*



(b) Post-processed *ansatz*

Figure 4 Post-processing of an *ansatz* where hard rules (Sect. 4.3) have been applied to the architecture represented in Equation (11) with $n = 3$ and $m = 4$

all solutions included in the top $\lfloor \alpha N \rfloor$ will contribute equally to the probabilistic model learning [36] (see Algorithm 1, Line 4). The surrogate model used in this approach surrogates the minimal thing needed for the EDA, that is, the ranking of solutions (line 4 Algorithm 1). This is introduced by a metric Score(A) (inspired in [37]) which measures the quality of a solution A within the rest of solutions of the population,

$$\text{Score}(A) = \sum_{B \in \mathbf{X}} (h(A,B) + 1 - h(B,A)), \tag{13}$$

where the higher Score(A), the better the quality of A, and $h(A,B)$ compares *ansatz* A to *ansatz* B as,

$$h(A,B) = \begin{cases} 0, & \text{if } \ P_B \geq P_A + \epsilon \\ 1, & \text{if } \ P_A \geq P_B + \epsilon \\ 2, & \text{otherwise} \end{cases} \tag{14}$$

where P_A, and P_B are the minimum expectation values (Equation (1)) found by a classical optimizer for architectures A and B, respectively and ϵ is a tolerance error configured by the user. Note that $h(A,B) = h(B,A) = 2$ means that two *ansatz* A and B are non comparable or very similar performance is expected.

Computing Score(A) involves $\lfloor \alpha N \rfloor - 1$ comparisons, and thus, this is clearly the main bottleneck of the task. In order to overcome this, we propose the use of support vector machines (SVMs) to approximate $h(A,B)$. We take the following input feature to the surrogate model:

$$\text{Flatten}(A + B, A - B) \tag{15}$$

where A and B are the two *ansatz* architectures to be compared, and the resultant vector size is $d = 2nm$. Thus, $h(A,B) \in \{0,1,2\}$ is approximated by $h'(\text{Flatten}(A,B)) \in \{0,1,2\}$ using SVM.

Several classification methods have been tested over some initial data randomly generated for different values of n, where SVM achieved better accuracy metrics. Results using cross-validation can be found in Appendix B.

The implementation has been obtained from LibSVM library [38].

The surrogate model is re-fitted after each iteration with the top 5 solutions in the ranking of the best solutions computed by the EDA (Sect. 4.5). Thus, in each iteration 5 classical parameter optimizations are carried out, and the number of parameter tuning processes executed during runtime is $N + 5t$, where t is the total number of iterations. Without the usage of the surrogate model approach, this number would have been $N(1 + t)$.

4.5 Evaluation

This approach aims to find the optimal *ansatz* for a given problem H in terms of trainability and expected energy. Here we define the following metrics to be computed for each proposed architecture.

First, IC (Equation (7)) maximization has been proved to be able to avoid BP in the *ansatz* parameter tuning [18]. Those architectures with low associated IC are less trainable/optimizable, compared to those with high IC. Our approach maximizes this metric through the optimization process. Here, the IC of an ansatz A is denoted as,

$$\mathrm{IC}(A) = \epsilon_M \sqrt{M}, \tag{16}$$

where ϵ_M is the ϵ associated to the norm of the gradient computed after a random walk over the parameters (Sect. 3.2), and M is the number of parameters of *ansatz* A.

Second, Score$(\cdot)$ (Equation (13)) evaluates the quality of a solution compared to a subset of solutions. Our approach implements an elite approach, in which the best solution of generation G_i also appears in generation G_{i+1}. Then finding a different best solution in G_{i+1} will lead to a best global solution in the whole optimization process. Thus, Score$(\cdot)$ is also desired to be maximized.

Maximizing both metrics becomes a multi-objective optimization problem, in which the Pareto frontier between both objectives is explored. During the optimization process defined in Algorithm 1 and Fig. 2, the truncation process ranks the solutions according to $g(\cdot)$, which is here defined as,

$$g(A) = \mathrm{HV}((\mathrm{Score}(A), \mathrm{IC}(A)), \mathbf{r}), \tag{17}$$

where HV$(\cdot)$ is the hypervolume contribution between the surrogate model output (Score(A)) and the information content computed (IC(A)), and $\mathbf{r}$ is the reference point. The $\lfloor \alpha N \rfloor$ best solutions in terms of HV$(\cdot)$ minimization are the ones that better approximate the Pareto frontier, and are the ones that promote to the next EDA iteration.

The reference point can be estimated based on the bounds of Score(A) and IC(A). In the former, the lower bound is set to zero (the worst solution within the population) and the upper bound to $2N$ (the best solution within the population). In the latter, the lower bound is set to zero (the least trainable scenario) and the upper bound to 2, based on previous experience. Then, $\mathrm{Score}(A) \in \{0, 1, \dots, 2N\}$ and $\mathrm{IC}(A) \in [0, 2] \in \mathbb{R}$, so the reference point is set to $\mathbf{r} = (2N, 2)$. We would like to remark that the reference point definition becomes more sensitive when decreasing the number of solutions selected in the truncation phase. The more solutions chosen, the less probable to discard a good solution due to wrong reference point definition.

Finally, the optimization problem is formalized as,

$$\begin{aligned} &\min_{\mathbf{X}} \quad g(\mathbf{X}) \\ &\text{subject to } \mathbf{X} \in \{0, 1, \dots, n_{gates}\}, \end{aligned} \tag{18}$$

where $\mathbf{X}$ denotes a codified *ansatz* (Equation (10)), and $g(\dot{})$ is defined at Equation (17).

5 Results

This section shows some numerical results on solving different Hamiltonians $H \in \{H_1, H_2, H_3, H_4\}$ (Appendix A), already studied in [39] for $n \in \{4, 8, 12\}$. The following sections compare the results found by the EDA approach with those presented in the dataset from [39]. In the original paper, the authors present several architectures which find similar state vectors in the search space of VQE *ansatz*, for each H_i. Henceforth, D_i^n denotes the set of architectures proposed in the dataset to solve the Hamiltonian H_i with n qubits.

Two experiments have been carried out in which, (i) the initial population of the EDA approach is initialized randomly to test if the algorithm is able to converge to similar solutions to those proposed in the dataset (Sect. 5.1), and (ii) the initial population is initialized from the *ansatzes* proposed in the dataset [39] to test if the algorithm is able to improve the given architectures (Sect. 5.2).

The size of the population, and maximum number of iterations of the EDA have been set to $N = 150$ and $t = 60$, respectively, for all the experiments. Note that, during this experimentation we will determine that an algorithm has already converged at a given iteration if there was no improvement in the best cost found in the previous 10 iterations. A difference lower than $1e - 8$ is reported as zero in the experimental results. Regarding the quantum circuit simulation, we simulate the measurement noise.

5.1 Random initialization

To randomly generate the initial population (G_0), a predefined probabilistic model is set to the algorithm, from which the set of solutions are sampled. Thus, some of the outcomes for each variable can be restricted, or boosted, decreasing or increasing the associated probabilities, respectively, as demanded by the user.

In this experiment, initially, all the possible outcomes have been set to equal probability for all the variables:

$$p(X_i = j) = \frac{1}{n_{gates} + 1}, \tag{19}$$

for all $i = 1, \dots, d$ and $j = 0, 1, \dots, n_{gates}$.

The initial population samples a set of N solutions, according to Equation (19). Each sample corresponds to a different architecture following the codification in Equation (10) and is post-processed (Sect. 4.3). The expectation value (Equation (1)) of each architecture is computed, where its parameters are classically optimized using an external optimizer. In this experiment we use COBYLA optimizer, as it has been shown to achieve good results in terms of CPU time and energy minimization [41]. Considering the set of solutions and associated expectation values, a surrogate model is trained (Sect. 4.4) and each solution is evaluated (Sect. 4.5).

The original dataset [39] proposes using dimensionality reduction to demonstrate that the minimal energy states achieved within D_i^n are very similar. Figure 5 shows the dimensional reduction using t-SNE [40] for the Hamiltonians approached, represented as clusters in two dimensions. The solutions found by the EDA approach (EDA$_i^n$, where i denotes the index of the faced Hamiltonian and n the number of qubits) are also represented by stars and different colors. Note that our approach is able to reach very similar solutions to the ones presented in the dataset.

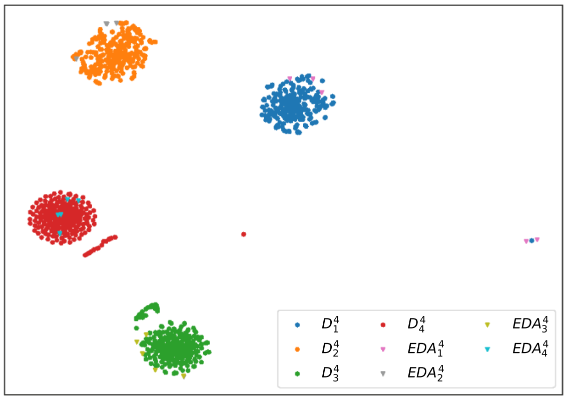


Figure 5 Visualization of the *ansatzes* found in the dataset (D_i^n) using t-SNE [40], which are colored depending on the Hamiltonian to be solved (H_i, where $i \in \{1,2,3,4\}$). Additionally, the best architectures found by the EDA approach (EDA_i^n) are represented using different colored and shaped points. Note that EDA_i^n regards the solutions found for Hamiltonian H_i. All the results shown correspond to $n = 4$

Table 1 ANOVA one-way test to reject the null hypothesis of equal means between the mean distances (Equation (20)), from the proposed by [39] *ansatzes* found by EDAs and $\{D_1^n, D_2^n, D_3^n, D_4^n\}$ proposed for $\{H_1, H_2, H_3, H_4\}$, respectively. A threshold of 5e-2 has been set to reject the null hypothesis, highlighting in bold those results below this value

H_i	$n = 4$	$n = 8$	$n = 12$
H_1	**3.0e-34**	**3.0e-2**	6.0e-1
H_2	**1.3e-4**	**1.1e-2**	1.5e-1
H_3	**1.0e-15**	3.0e-1	1.1e-1
H_4	**2.0e-8**	**5.1e-2**	2.1e-1

In the following analysis the fidelity of the lowest energy state found by the EDA approach is compared to those obtained by the *ansatzes* provided in the dataset for different problems $\{H_1, H_2, H_3, H_4\}$ and number of qubits (n), that is, by D_i^n.

The distance from each proposed *ansatz* (A) in EDA_i^n to each cluster of architectures D_i^n is computed by the arithmetic mean distance to each of the *ansatzes* belonging to D_i^n as,

$$\mathrm{dist}(A, D_i^n) = \frac{1}{|D_i^n|}\Big(\sum_{B \in D_i^n} 1 - F(|\Psi_A\rangle, |\Psi_B\rangle)\Big), \tag{20}$$

where D_i^n is the subset of *ansatzes* (with size $|D_i|$) in the dataset proposed to solve H_i with n qubits and meet $m \pm \sqrt{m}$ restriction, $F(\cdot)$ is the fidelity between two quantum states, and $|\Psi_A\rangle$ and $|\Psi_B\rangle$ are the lowest energy states achieved by *ansatzes* A and B, respectively, after classical parameter optimization.

Table 1 shows the p-values computed using the ANOVA test[1] to reject the null hypothesis of equal means between each *ansatz* in EDA_i^n and the different clusters D_i^n, where highlighted results are rejected. Appendix C details the distance computations statistically analyzed in this table. An increasing number of non-rejected hypotheses is observed for increasing number of qubits (n), which suggests that the EDA is proposing architec-

[1] All the data used for the ANOVA tests fit Gaussian distributions.

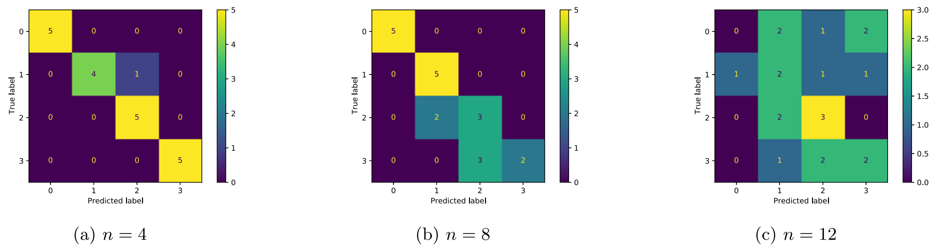


Figure 6 Confusion matrices for $n \in \{4, 8, 12\}$

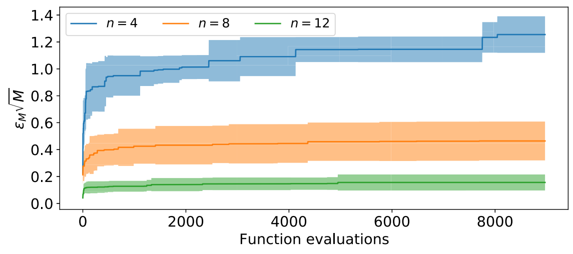


Figure 7 Mean and standard deviation of IC maximization aggregating the optimization process of different H_i for different numbers of qubits (n)

tures much different to the ones available at the dataset for $n = 12$. Increasing the number of qubits (n) also involves increasing the number of variables of the EDA optimizer. According to the results found, the population size set is not enough to generate a large number of samples which covers the increasing cardinality of the problem. Also, larger number of qubits should also involve a larger *ansatz* depth, so m should also be increased to allow more expressive quantum circuits. This suggests that the chosen configuration is valid to problems up to $n < 8$. For bigger instances, a different configuration of the hyper-parameters m and N should be chosen, although this would involve a drastic increase of the CPU time.

Assuming that a truly classified *ansatz* (A) is the case in which the closest cluster D_i^n represents H_i, and $A \in \mathrm{EDA}_i^n$ was optimized for Hamiltonian H_i as well, Fig. 6 shows the confusion matrices. The percentage of correctly classified *ansatzes* is 95%, 75% and 35% for $n = 4, 8, 12$, respectively, where a decreasing tendency is observed for increasing n; however, for $n = 12$ the EDA was not able to found any statistical significant result.

Figure 7 shows the IC convergence plot during the optimization process of the EDA approach. The associated shade shows a mean aggregation of the optimization processes regarding different $\{H_1, H_2, H_3, H_4\}$, where a maximizing monotonic tendency is observed. Regardless of the results encountered, the three scenarios show that the algorithm has converged. Note that, the mean IC found by the optimizer denotes an exponential decay with the number of qubits (n), as expected according to [16, 18].

Because Score(A) returns a metric comparing *ansatz* A with the rest of the architectures within the population to which A belongs, the trend throughout the optimization process is not an interesting fact to analyze.

Appendix D shows the Pareto frontier approximation (non-dominated solutions highlighted as orange spots) for each H_i we are facing (in columns) and different values of n

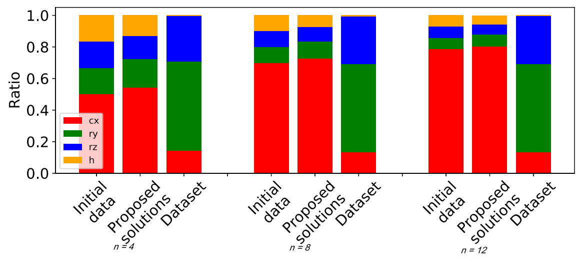


Figure 8 Ratio of $\{CNOT, RY, RZ, H\}$ gates in the *ansatz* design of the randomly generated initial data (G_0), best EDA solutions found (EDA_i^n), and dataset (D_i^n) [39], for $n \in \{4, 8, 12\}$, respectively

(in rows). It is observed how both objectives are conflicting, and maximizing one of the objectives worsens the second, and vice-versa. Thus, a trade-off between both objectives through the Pareto frontier approximation is desired. Note that the scale of the Y-axis (IC) is different for different number of qubits, as explained before.

Considering the best solutions found by the EDA, i.e., those that better approximate the Pareto frontier, we now compare the characteristics of the *ansatzes* proposals with those available in the dataset [39] with depth in the range $m \pm \sqrt{m}$ (for a fair comparison and ensure a minimum number of instances from the original dataset). A drastic increase in the number of certain quantum gates might improve the performance of the *ansatz*, however, this may lead to a poor trainability. Thus, the ratio among the gates set used, and the number of gates is further analyzed.

Figure 8 shows the ratio of the different available universal gates in the set of initial randomly generated data (G_0), the solutions found by EDA approach (EDA_i^n) and the best solutions from the original dataset (D_i^n), for different values of n. A strong correlation is observed between the initial data and the proposed solutions, independently of n, where the EDA_i^n has a slightly higher ratio of CNOT gates compared to G_0. However, comparing to D_i^n, our proposals achieve a much lower ratio of parametric gates, compensating it with superposition and two-qubit gates. Although the ratios for D_i^n seem to remain constant along n, our approach increases the number of CNOT gates with n.

Figure 9 plots the number of parameters as a function of n, in the set of initial randomly generated data (G_0), the solutions found by the EDA approach (EDA_i^n) and the original dataset (D_i^n). Although the number of gates increases linearly in the three cases, comparing the slopes found in the linear approximations of the three cases, the green function (D_i^n) denotes a coefficient approximately 6 times bigger than the other two functions. We show that our EDA is able to learn that a bigger number of parameters is needed, however, it does not increase this number drastically, as it is able to converge to simpler *ansatz*. Shallower *ansatzes* (low values in the Y-axis) are more convenient to be executed in real quantum devices due to quantum coherence and other issues of the NISQ devices.

In this experiment we tested whether our approach initialized from a random set of *ansatzes* is able to converge and find similar solutions to the ones proposed in the dataset, assumed to be optimal. Figure 5 and Table 1 show that our algorithm finds solutions with similar state fidelity as the ones in the dataset.

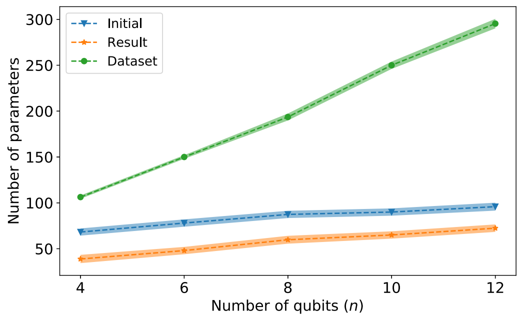


Figure 9 Mean and standard deviation of the number of parameters (Y-axis) as a function of the number of qubits (X-axis), in the *ansatzes* found in the randomly generated initial data (G_0), best EDA solutions found (EDA_i^n), and dataset (D_i^n) [39]. Note that the values for $n = 6, 10$ have been approximated through a linear regression

5.2 Initialization with the dataset

The previous results have shown that the EDA approach is able to provide trainable and well performing architectures. In this section we initialize the EDA optimizer from the *ansatzes* provided in the dataset (D_i^n) to test whether it is able to converge to better solutions. Thus, the EDA execution used to face the Hamiltonian H_i will be initialized using $G_0 = D_i^n$. In this case, D_i^n will consist of all those architectures that meet the depth constraint imposed by the EDA. Note that, in case an architecture has a depth smaller than that imposed, the coding in binary (Equation (10)) would be equivalent to fill with identity gates (I) until the desired depth is reached.

The purpose of this experiment is that, given a set of *ansatzes*, which are known to have good performance, we try to improve their trainability while maintaining a similar behavior. In order to compare the results found by the EDA, the energy (Equation (1)) using a second level classical optimizer and the IC (Equation (7)) are computed for all the *ansatzes* in all D_i^n. Results are shown in Table 6.

Figure 11 (Appendix) shows the Pareto frontier approximations for each H_i we are facing and different numbers of n. Note that, with increasing number of qubits, the conflict between both objectives becomes more drastic. However, the EDA approach is able to identify the promising solutions in the Pareto frontier. Note that the initial generation $G_0 = D_i^n$ has been also represented to establish a reference in terms of IC. However, Score(A) for the first generation should not be taken into account, as D_i^n represents similar minimal energy state vectors (Fig. 5), and thus, are not comparable.

Table 7 (Appendix) shows the best E and IC found by the EDA approach where COBYLA optimizer is used, for the *ansatz* parameter optimization. Note that the solutions shown in the tables are the ones that maximize HV in the Pareto frontier approximation, that is, a trade-off between both objectives in the non-dominated solutions set is found. Although in this case it is important to show the solution that optimizes the HV, it is possible to analyze each of the non-dominated solutions from the Pareto front in order to maximize any of the two metrics.

Regarding the results shown in Table 7, it is observed a good performance in terms of expectation value minimization for $n = 4$. Moreover, the IC achieved is noticeable better, which also happens in the case of $n = 8$. However, the expectation value obtained for H_3

and H_4 for $n = 8$ is worse than that described in the original dataset, which suggests that the EDA approach is not able to improve the metrics in Table 6.

In this experiment we tested whether our approach is able to improve the quality of the *ansatz* provided in the dataset, from which the EDA is initialized. Our results show that the EDA approach is able to improve them in some of the cases, and suggest that a hyper-parameter tuning should be carried out for increasing number of qubits.

6 Conclusions

In this paper we present a novel method for architecture search, in which the complexity of the multi-level optimization problem has been drastically reduced by using surrogate modelling. The EDA approach optimizes the energy estimated by the surrogate modelling by performing comparisons by pairs, and reduces the possibility of barren plateaus issues.

The experimental results showcase two different situations for optimizing different Hamiltonians: (i) the EDA is initialized from a random subset of solutions, and (ii) the EDA is initialized from the best solutions presented in the dataset. In the former case, the results show that the optimizer is able to converge to the same solutions presented in the dataset when the number of qubits is lower than $n = 8$, and the hyper-parameters should be tuned for greater values of n. In the latter case, the EDA is able to improve the state of the art in some of the cases. Our approach is able to find solutions that keep a good performance regarding energy minimization, but also improve the trainability of the *ansatzes* encountered.

The numerical results analyzed suggest that the performance of our approach worsens with the number of qubits, unless the population size (N) and the number of iterations (t) are increased. However, in order to implement a useful approach for NISQ and fault tolerant devices, the algorithm runtime for the optimization process is limited, in contrast to neural network architecture search, where the coherence of the devices do not change during time. Future work in this field would include the scalability of the algorithm to higher number of qubits (n). We suggest combining our approach with large-scale EA techniques which have been deeply studied in the last decade [42, 43].

The EDA internally uses HV for ranking the architectures to be selected. Although the IC upper bound has been set based on previous experience, future work would include a dynamic definition of the reference point for the HV computation, during runtime.

Given that this research is at an early stage, our primary focus is on showing underpinnings and initial feasibility rather than conducting exhaustive empirical comparisons with state-of-the-art methods. Comprehensive benchmarking and detailed empirical evaluations are planned for future studies.

Appendix A: Hamiltonians

This section describes the Hamiltonians used for the experimental results. Note that the following benchmarks and coefficients have been used in order to compare the results with the ones found in [39].

1D transverse-field Ising model:

$$H_1 = \sum_{i=1}^{n-1} Z_i Z_{i+1} + 2 \sum_{i=1}^{n} X_n$$

1D Heisenberg model:

$$H_2 = \sum_{i=1}^{n-1}(X_iX_{i+1} + Y_iY_{i+1} + Z_iZ_{i+1}) + 2\sum_{i=1}^{n} Z_n$$

Su-Schrieffer-Heeger model:

$$H_3 = \sum_{i=1}^{n-1}\left(1 + \frac{3}{2}(-1)^{i-1}\right)(X_iX_{i+1} + Y_iY_{i+1} + Z_iZ_{i+1}) + 2\sum_{i=1}^{n} X_n$$

J_1 - J_2 model:

$$H_4 = \sum_{i=1}^{n-1}(X_iX_{i+1} + Y_iY_{i+1} + Z_iZ_{i+1}) + 3\sum_{i=1}^{n-2}(X_iX_{i+2} + Y_iY_{i+2} + Z_iZ_{i+2})$$

Appendix B: Surrogate model prediction

Here we compare the performance of different surrogate models by comparing different *ansatzes* by pairs in a given initial data for different number of qubits.

Different architectures have been built for problems described in Appendix A and different values of n. The number of architectures have been set to $N = 37.5n$, and the circuit depth to $m = 60$. Table 2 shows the accuracy found for different models with different configurations. Results show that support vector classifier (SVC) achieves the best metrics, and thus, is used as surrogate model in our approach.

Table 2 Accuracy found after evaluating each model in a set of initial architectures using cross-validation with 15 folds. Independently of n, all the *ansatzes* have been restricted to $m = 60$, and $N = 37.5n$. Random forest with different numbers of estimators, k-nearest neighbors (KNN) with different numbers of neighbors, support vector classifier (SVC), decision tree, and naive Bayes have been tested

model	$n = 4$	$n = 8$	$n = 12$
Random_forest_20	0.76	0.77	0.75
Random_forest_50	0.81	0.82	0.80
Random_forest_80	0.82	0.83	0.80
KNN_2	0.64	0.66	0.68
KNN_5	0.72	0.74	0.75
KNN_15	0.78	0.79	0.79
SVC	**0.91**	**0.92**	**0.90**
Decision tree	0.64	0.65	0.65
Naive Bayes	0.69	0.76	0.78

Appendix C: Distance computation

Here we detail the distance comparison between all the proposed solutions within EDA_i^n and each of the clusters D_i^n by computing Equation (20). Note that index j denotes each of the 5 best results found by the EDA. Table 3-5 show the distance computations for $n \in [4, 8, 12]$, respectively.

Table 3 Distance (Equation (20)) between each *ansatz* in EDA^4_{ij} and D^4_i, where *i* denotes the Hamiltonian index and $n = 4$. Bold values represent those instances in which the closest cluster to EDA^4_{ij} is D^4_i

ansatz (EDA^4_{ij})	H_i	dist($\mathrm{EDA}^4_{1_j}, D^4_1$)	dist($\mathrm{EDA}^4_{2_j}, D^4_2$)	dist($\mathrm{EDA}^4_{3_j}, D^4_3$)	dist($\mathrm{EDA}^4_{4_j}, D^4_4$)
$\mathrm{EDA}^4_{1_1}$	H_1	**0.018**	0.998	0.990	0.999
$\mathrm{EDA}^4_{1_2}$	H_1	**0.011**	0.999	0.995	0.995
$\mathrm{EDA}^4_{1_3}$	H_1	**0.011**	0.999	0.989	0.999
$\mathrm{EDA}^4_{1_4}$	H_1	**0.027**	0.990	0.991	0.995
$\mathrm{EDA}^4_{1_5}$	H_1	**0.011**	0.999	0.989	0.999
$\mathrm{EDA}^4_{2_1}$	H_2	0.999	**0.038**	0.982	0.997
$\mathrm{EDA}^4_{2_2}$	H_2	0.999	**0.049**	0.993	0.999
$\mathrm{EDA}^4_{2_3}$	H_2	0.993	0.954	**0.233**	0.880
$\mathrm{EDA}^4_{2_4}$	H_2	0.999	**0.035**	0.976	0.990
$\mathrm{EDA}^4_{2_5}$	H_2	0.970	**0.374**	0.794	0.965
$\mathrm{EDA}^4_{3_1}$	H_3	0.993	0.999	**0.051**	0.660
$\mathrm{EDA}^4_{3_2}$	H_3	0.992	0.999	**0.058**	0.648
$\mathrm{EDA}^4_{3_3}$	H_3	0.988	0.998	**0.064**	0.646
$\mathrm{EDA}^4_{3_4}$	H_3	0.995	0.997	**0.069**	0.631
$\mathrm{EDA}^4_{3_5}$	H_3	0.987	0.999	**0.056**	0.637
$\mathrm{EDA}^4_{4_1}$	H_4	0.991	0.995	0.691	**0.077**
$\mathrm{EDA}^4_{4_2}$	H_4	0.993	0.992	0.752	**0.061**
$\mathrm{EDA}^4_{4_3}$	H_4	0.998	0.991	0.811	**0.081**
$\mathrm{EDA}^4_{4_4}$	H_4	0.992	0.997	0.702	**0.099**
$\mathrm{EDA}^4_{4_5}$	H_4	0.990	0.993	0.329	**0.011**

Table 4 Distance (Equation (20)) between each *ansatz* in EDA^8_{ij} and D^8_i, where *i* denotes the Hamiltonian index and $n = 8$. Bold values represent those instances in which the closest cluster to EDA^5_{ij} is D^8_i

ansatz (EDA^8_{ij})	H_i	dist($\mathrm{EDA}^8_{1_j}, D^8_1$)	dist($\mathrm{EDA}^8_{2_j}, D^8_2$)	dist($\mathrm{EDA}^8_{3_j}, D^8_3$)	dist($\mathrm{EDA}^8_{4_j}, D^8_4$)
$\mathrm{EDA}^8_{1_1}$	H_1	**0.973**	0.995	0.995	0.997
$\mathrm{EDA}^8_{1_2}$	H_1	**0.950**	0.996	0.996	0.994
$\mathrm{EDA}^8_{1_3}$	H_1	**0.830**	0.998	0.998	0.998
$\mathrm{EDA}^8_{1_4}$	H_1	**0.553**	0.999	0.999	0.999
$\mathrm{EDA}^8_{1_5}$	H_1	**0.942**	0.995	0.990	0.997
$\mathrm{EDA}^8_{2_1}$	H_2	0.990	**0.926**	0.968	0.991
$\mathrm{EDA}^8_{2_2}$	H_2	0.998	**0.906**	0.998	0.999
$\mathrm{EDA}^8_{2_3}$	H_2	0.998	**0.963**	0.989	0.995
$\mathrm{EDA}^8_{2_4}$	H_2	0.996	**0.992**	0.998	0.998
$\mathrm{EDA}^8_{2_5}$	H_2	0.999	**0.991**	0.999	0.999
$\mathrm{EDA}^8_{3_1}$	H_3	0.999	0.999	**0.957**	0.995
$\mathrm{EDA}^8_{3_2}$	H_3	0.999	**0.958**	0.983	0.985
$\mathrm{EDA}^8_{3_3}$	H_3	0.999	0.999	**0.522**	0.949
$\mathrm{EDA}^8_{3_4}$	H_3	0.998	0.996	**0.958**	0.983
$\mathrm{EDA}^8_{3_5}$	H_3	0.999	**0.922**	0.999	0.996
$\mathrm{EDA}^8_{4_1}$	H_4	0.999	0.999	**0.971**	0.981
$\mathrm{EDA}^8_{4_2}$	H_4	0.999	0.998	0.992	**0.945**
$\mathrm{EDA}^8_{4_3}$	H_4	0.998	0.998	**0.988**	0.996
$\mathrm{EDA}^8_{4_4}$	H_4	0.999	0.999	**0.982**	0.994
$\mathrm{EDA}^8_{4_5}$	H_4	0.999	0.999	0.999	**0.988**

Appendix D: Pareto frontier approximations

Figure 10 shows the Pareto frontier approximation for different H and number of qubits. The columns refer to the problem instances, while the rows refer to the number of qubits

Table 5 Distance (Equation (20)) between each *ansatz* in EDA_{ij}^{12} and D_i^{12}, where i denotes the Hamiltonian index and $n = 12$

ansatz (EDA_{ij}^{12})	H_i	dist($\mathrm{EDA}_{1j}^{12}, D_1^{12}$)	dist($\mathrm{EDA}_{2j}^{12}, D_2^{12}$)	dist($\mathrm{EDA}_{3j}^{12}, D_3^{12}$)	dist($\mathrm{EDA}_{4j}^{12}, D_4^{12}$)
EDA_{11}^{12}	H_1	0.999	0.999	0.999	0.999
EDA_{12}^{12}	H_1	0.999	0.999	0.999	0.999
EDA_{13}^{12}	H_1	0.999	0.999	0.999	0.999
EDA_{14}^{12}	H_1	0.999	0.999	0.999	0.999
EDA_{15}^{12}	H_1	0.999	0.999	0.999	0.999
EDA_{21}^{12}	H_2	0.999	0.999	0.999	0.999
EDA_{22}^{12}	H_2	0.999	0.999	0.999	0.999
EDA_{23}^{12}	H_2	0.999	0.999	0.999	0.999
EDA_{24}^{12}	H_2	0.999	0.999	0.999	0.999
EDA_{25}^{12}	H_2	0.999	0.999	0.999	0.999
EDA_{31}^{12}	H_3	0.999	0.998	0.999	0.999
EDA_{32}^{12}	H_3	0.999	0.999	0.999	0.999
EDA_{33}^{12}	H_3	0.999	0.999	0.999	0.999
EDA_{34}^{12}	H_3	0.999	0.999	0.998	0.999
EDA_{35}^{12}	H_3	0.999	0.999	0.999	0.999
EDA_{41}^{12}	H_4	0.999	0.999	0.999	0.999
EDA_{42}^{12}	H_4	0.999	0.999	0.999	0.999
EDA_{43}^{12}	H_4	0.999	0.999	0.999	0.999
EDA_{44}^{12}	H_4	0.999	0.999	0.999	0.999
EDA_{45}^{12}	H_4	0.999	0.999	0.999	0.999

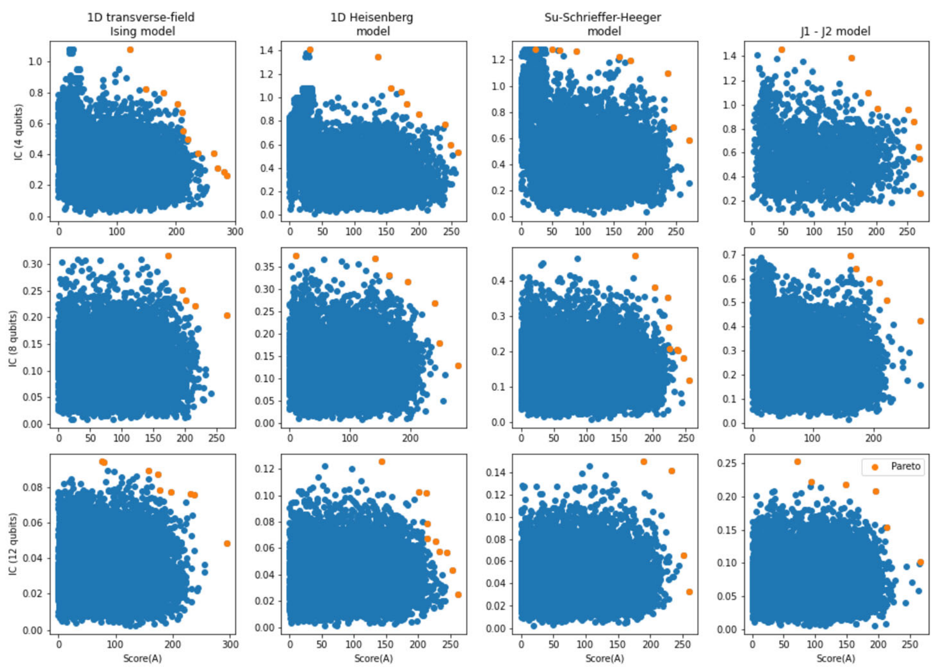


Figure 10 Pareto frontier approximation (orange spots) over all the *ansatzes* considered (blue spots) during optimization process. Columns refer to problem instances, while rows refer to number of qubits (n)

(n). Each subplot shows all the evaluated *ansatzes* (blue spots) from which the non-dominated solutions are highlighted (orange spot).

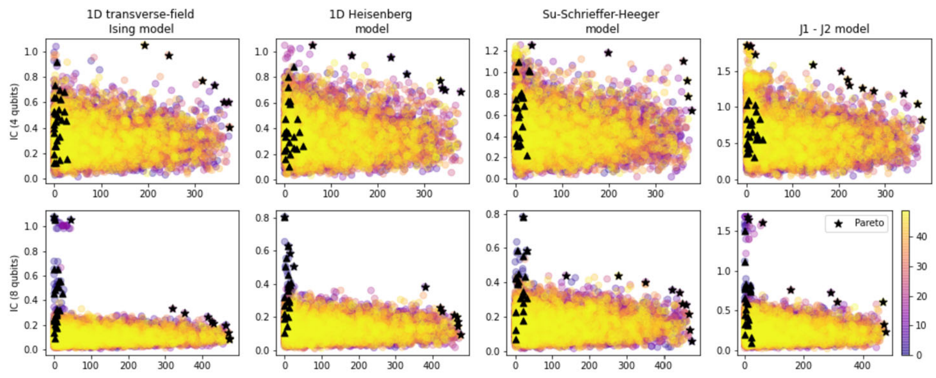


Figure 11 Pareto frontier approximation (black stars) over all the *ansatzes* considered (colored spots) during the optimization process. Black triangles regard the *ansatzes* included in the dataset. Columns refer to problem instances, while rows refer to number of qubits (n)

Table 6 Mean and standard deviation of expectation value (E) (Equation (1)) and information content (IC) (Equation (7)), respectively, found in the *ansatz* in the dataset whose depth is in the range $m \pm \sqrt{m}$, for different number of qubits n and Hamiltonian H_i

	$n = 4$		$n = 8$	
	E	IC	E	IC
H_1	−8.37 ± 0.01	0.47 ± 0.14	−16.89 ± 0.01	0.46 ± 0.16
H_2	−7.83 ± 0.01	0.51 ± 0.16	−15.92 ± 0.02	0.45 ± 0.06
H_3	−14.19 ± 1.87	0.63 ± 0.15	−30.07 ± 0.01	0.51 ± 0.07
H_4	−17.18 ± 2.20	0.80 ± 0.09	−39.05 ± 0.04	0.82 ± 0.15

Table 7 Best expectation value (E) (Equation (1)) and information content (IC) (Equation (7)) found by the EDA approach (assisted by COBYLA) for different number of qubits (n) and Hamiltonians (H_i), where HV is maximized in the best Pareto approximation

	$n = 4$		$n = 8$	
	E	IC	E	IC
H_1	−7.81	0.97	−16.18	0.56
H_2	−6.74	0.73	−13.58	0.45
H_3	−14.03	1.00	−29.28	0.43
H_4	−17.21	1.47	−26.87	1.57

Appendix E: IC and expectation values comparison

Table 6 describes the mean expectation value (Equation (1)) and IC (Equation (7)) for the *ansatzes* available in the dataset (D_i^n) for different values of n.

Table 7 describes the best expectation value and IC found by the EDA approach for different H_i and values of n, where the HV is maximized. That is, the solutions which maximize HV within EDA_i^n.

Acknowledgements
We would like to thank Yash J. Patel, Onur Danaci, Adrián Pérez-Salinas, Patrick Emonts, and the people from $\langle aQa^L \rangle$ group for fruitful discussions in the topic, and inviting Vicente P. Soloviev as a visitor for a few months in University of Leiden.
This work has been partially supported by the Spanish Ministry of Science and Innovation through the PID2022-139977NB-I00 project and TED2021-131310B-I00 ("Bayes-Interpret"), and by the Autonomous Community of Madrid within the ELLIS Unit Madrid framework.
This work was also partially supported by the Dutch Research Council (NWO/OCW), as part of the Quantum Software Consortium programme (project number 024.003.03), and co-funded by the European Union (ERC CoG, BeMAIQuantum, 101124342). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them

Vicente P. Soloviev has been supported by the predoctoral grant FPI PRE2020-094828 from the Spanish Ministry of Science and Innovation.

Author contributions

Vicente P. Soloviev: Conceptualization, Methodology, Software, Validation, Formal analysis, Writing – original draft. Vedran Dunjko: Project administration, Supervision, Resources, Writing – review & editing. Concha Bielza: Project administration, Supervision, Resources, Writing – review & editing. Pedro Larrañaga: Project administration, Supervision, Resources, Writing – review & editing. Hao Wang: Project administration, Supervision, Resources, Writing – review & editing.

Data Availability

No datasets were generated or analysed during the current study.

Declarations

Competing interests

The authors declare no competing interests.

Author details

[1]Computational Intelligence Group (CIG), Universidad Politécnica de Madrid, Madrid, Spain. [2]Applied Quantum Algorithms Leiden, Leiden University, Leiden, The Netherlands. [3]Leiden Institute of Advanced Computer Science (LIACS), Leiden University, Leiden, The Netherlands.

Received: 10 July 2024 Accepted: 14 October 2024 Published online: 18 October 2024

References

1. Bharti K, Cervera-Lierta A, Kyaw TH, Haug T, Alperin-Lea S, Anand A, Degroote M, Heimonen H, Kottmann JS, Menke T, et al. Noisy intermediate-scale quantum algorithms. Rev Mod Phys. 2022;94(1):015004.
2. Peruzzo A, McClean J, Shadbolt P, Yung M-H, Zhou X-Q, Love PJ, Aspuru-Guzik A, O'brien JL. A variational eigenvalue solver on a photonic quantum processor. Nat Commun. 2014;5(1):4213.
3. Soloviev VP, Bielza C, Larrañaga P. Quantum approximate optimization algorithm for Bayesian network structure learning. Quantum Inf Process. 2022;22(1):19.
4. Schuld M, Petruccione F. Supervised learning with quantum computers. vol. 17. Berlin: Springer; 2018.
5. Wiśniewska J, Sawerwain M. Variational quantum eigensolver for classification in credit sales risk. 2023. arXiv:2303.02797.
6. Kl Barkoutsos P, Nannicini G, Robert A, Tavernelli I, Woerner S. Improving variational quantum optimization using CVaR. Quantum. 2020;4:256.
7. Li L, Fan M, Coram M, Riley P, Leichenauer S, et al. Quantum optimization with a novel Gibbs objective function and ansatz architecture search. Phys Rev Res. 2020;2(2):023074.
8. Farhi E, Goldstone J, Gutmann S. A quantum approximate optimization algorithm. 2014. arXiv:1411.4028.
9. Ruder S. An overview of gradient descent optimization algorithms. 2016. arXiv:1609.04747.
10. Byrd RH, Lu P, Nocedal J, Zhu C. A limited memory algorithm for bound constrained optimization. SIAM J Sci Comput. 1995;16(5):1190–208.
11. Soloviev VP, Larrañaga P, Bielza C. Variational quantum algorithm parameter tuning with estimation of distribution algorithms. In: 2023 IEEE congress on evolutionary computation. IEEE; 2023. p. 1–9.
12. Spall JC. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. IEEE Trans Autom Control. 1992;37(3):332–41.
13. Garcia-Saez A, Riu J. Quantum observables for continuous control of the quantum approximate optimization algorithm via reinforcement learning. 2019. arXiv:1911.09682.
14. Mejía-de Dios J-A, Rodríguez-Molina A, Mezura-Montes E. Multiobjective bilevel optimization: a survey of the state-of-the-art. In: IEEE transactions on systems, man, and cybernetics: systems. 2023.
15. Anschuetz ER, Kiani BT. Quantum variational algorithms are swamped with traps. Nat Commun. 2022;13(1):7760.
16. Cerezo M, Sone A, Volkoff T, Cincio L, Coles PJ. Cost function dependent barren plateaus in shallow parametrized quantum circuits. Nat Commun. 2021;12(1):1791.
17. McClean JR, Boixo S, Smelyanskiy VN, Babbush R, Neven H. Barren plateaus in quantum neural network training landscapes. Nat Commun. 2018;9(1):4812.
18. Pérez-Salinas A, Wang H, Bonet-Monroig X. Analyzing variational quantum landscapes with information content. 2023. arXiv:2303.16893.
19. Pirhooshyaran M, Terlaky T. Quantum circuit design search. Quantum Mach Intell. 2021;3:1–14.
20. Fösel T, Niu MY, Marquardt F, Li L. Quantum circuit optimization with deep reinforcement learning. 2021. arXiv:2103.07585.
21. Ostaszewski M, Trenkwalder LM, Masarczyk W, Scerri E, Dunjko V. Reinforcement learning for optimization of variational quantum circuit architectures. Adv Neural Inf Process Syst. 2021;34:18182–94.
22. Patel YJ, Kundu A, Ostaszewski M, Bonet-Monroig X, Dunjko V, Danaci O. Curriculum reinforcement learning for quantum architecture search under hardware errors. 2024. arXiv preprint. arXiv:2402.03500.
23. Chivilikhin D, Samarin A, Ulyantsev V, Iorsh I, Oganov AR, Kyriienko O. MoG-VQE: Multiobjective genetic variational quantum eigensolver. 2020. arXiv:2007.04424.
24. Rattew AG, Hu S, Pistoia M, Chen R, Wood S. A domain-agnostic, noise-resistant, hardware-efficient evolutionary variational quantum eigensolver. 2019. arXiv:1910.09694.
25. Sünkel L, Martyniuk D, Mattern D, Jung J, Paschke A. GA4QCO: genetic algorithm for quantum circuit optimization. 2023. arXiv:2302.01303.

26. Grimsley HR, Economou SE, Barnes E, Mayhall NJ. An adaptive variational algorithm for exact molecular simulations on a quantum computer. Nat Commun. 2019;10(1):3007.
27. Ostaszewski M, Grant E, Benedetti M. Structure optimization for parameterized quantum circuits. Quantum. 2021;5:391.
28. Wu W, Yan G, Lu X, Pan K, Yan J. QuantumDARTS: differentiable quantum architecture search for variational quantum algorithms. 2023.
29. Liu H, Simonyan K, Darts YY. Differentiable architecture search. 2018. arXiv:1806.09055.
30. Zhang S-X, Hsieh C-Y, Zhang S, Yao H. Differentiable quantum architecture search. Quantum Sci Technol. 2022;7(4):045023.
31. Du Y, Huang T, You S, Hsieh M-H, Tao D. Quantum circuit architecture search: error mitigation and trainability enhancement for variational quantum solvers. 2020. arXiv:2010.10217.
32. Linghu K, Qian Y, Wang R, Hu M-J, Li Z, Li X, Xu H, Zhang J, Ma T, Zhao P, et al. Quantum circuit architecture search on a superconducting processor. 2022. arXiv:2201.00934.
33. Larrañaga P, Lozano JA. Estimation of distribution algorithms: a new tool for evolutionary computation. Dordrecht: Kluwer Academic; 2001.
34. Li M, Yao X. Quality evaluation of solution sets in multiobjective optimisation: a survey. ACM Comput Surv. 2019;52(2):1–38.
35. Beume N, Fonseca CM, Lopez-Ibanez M, Paquete L, Vahrenhold J. On the complexity of computing the hypervolume indicator. IEEE Trans Evol Comput. 2009;13(5):1075–82.
36. Larrañaga P, Bielza C. Estimation of distribution algorithms in machine learning: a survey. In: IEEE transactions on evolutionary computation. 2023.
37. Shi R, Luo J, Liu Q. Fast evolutionary neural architecture search based on Bayesian surrogate model. In: 2021 IEEE congress on evolutionary computation. IEEE; 2021. p. 1217–24.
38. Chang C-C, Lin C-J. LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol. 2011;2(3):1–27.
39. Nakayama A, Mitarai K, Placidi L, Sugimoto T, Fujii K. VQE-generated quantum circuit dataset for machine learning; 2023.
40. Van der Maaten L, Hinton G. Visualizing data using t-SNE. J Mach Learn Res. 2008;**9**(11).
41. Powell MJ. Direct search algorithms for optimization calculations. Acta Numer. 1998;7:287–336.
42. Liu S, Lin Q, Li J, Tan KC. A survey on learnable evolutionary algorithms for scalable multiobjective optimization. IEEE Trans Evol Comput. 2023;27(6):1941–61.
43. Liu J, Sarker R, Elsayed S, Essam D, Siswanto N. Large-scale evolutionary optimization: a review and comparative study. Swarm Evol Comput. 2024;58:101466.

Publisher's Note