



Hybrid Quantum Computing

Technical White Paper

How hybrid quantum computing
creates business benefits today

By Terra Quantum AG
terraquantum.swiss

May, 2022

Practical Application-Specific Advantage through Hybrid Quantum Computing

Michael Perelshtein, Asel Sagingalieva, Karan Pinto, Vishal Shete, Alexey Pakhomchik,
Artem Melnikov, Florian Neukart, Georg Gesek, Alexey Melnikov, Valerii Vinokur
Terra Quantum AG, 9400 Rorschach, Switzerland
QWare AG, 9400 Rorschach, Switzerland

Quantum computing promises to tackle technological and industrial problems insurmountable for classical computers. However, today's quantum computers still have limited demonstrable functionality, and it is expected that scaling up to millions of qubits is required for them to live up to this touted promise. The feasible route in achieving practical quantum advantage goals is to implement a hybrid operational mode that realizes the cohesion of quantum and classical computers. Here we present a hybrid quantum cloud based on a memory-centric and heterogeneous multiprocessing architecture, integrated into a high-performance computing data center grade environment. We demonstrate that utilizing the quantum cloud, our hybrid quantum algorithms including Quantum Encoding (QuEnc), Hybrid Quantum Neural Networks and Tensor Networks enable advantages in optimization, machine learning, and simulation fields. We show the advantage of hybrid algorithms compared to standard classical algorithms in both the computational speed and quality of the solution. The achieved advance in hybrid quantum hardware and software makes quantum computing useful in practice today.

I. INTRODUCTION

Explosive development of quantum technologies imposes a challenge to correctly identify the approach most effective in exploiting the potential of quantum computing and successful in addressing the required industry-relevant problems in full-scale. The present state of the quantum art is attested as Noisy Intermediate-Scale Quantum (NISQ) technology, in which quantum computers comprising 50-100 qubits are on their way of surpassing the capabilities of today's classical digital computers but in which quantum device decoherence, measurement imperfections, control errors and architectural limitations retard the further size growth of quantum circuits that can be reliably implemented [1, 2]. The most promising way to achieve the desirable development of computing technology on the practical applications level, is to hybridize the powers of the available NISQ devices with state-of-the-art classical high-performance computing capabilities [3–6]. This hybridization is needed not

only for combining the powers of the different types of processors, quantum processing units (QPUs), central processing units (CPUs) and graphical processing units (GPUs), but also for a classical control, optimization, calibration, and error-correction of NISQ devices. Classical devices are important for verification of quantum devices [7, 8], and for providing logical quantum devices [9, 10]. The task is thus to integrate all components into a single platform, so that classical and quantum computing units exchange information locally and allow for efficient high-speed device-to-device and device-to-memory connections.

Here we develop a Hybrid Quantum High-Performance Computing (HQC) cloud, with an in-memory computing model and complete software stack including hardware, operating system, middleware, and application programming interfaces (API), that fuses together the best of today's classical computing elements and the emerging quantum components into a hybrid model for productive use. Our hybrid quantum cloud, QWare, that we introduce here, provides a private cloud-based platform and containerized environment in a memory-centric compute architecture where research partners and industrial organizations can build and deploy their own hybrid quantum applications at scale [11]. Our state-of-the-art GAIA-X-compliant data centers combine high-performance classical infrastructure, simulated QPUs [12–15] and advanced machine learning tools for efficient production grade deployment of hybrid quantum-classical algorithms. There are three key innovations of the QWare cloud, which are shown schematically in Fig. 1. First, a memory-centric compute architecture, instead of the von Neumann architecture [16, 17], is used. It supports heterogeneous processing with QPUs, CPUs and GPUs as well as hybrid quantum compute, which includes all processes and procedures involved in a hybrid quantum computation of a hybrid quantum application in a container. Second, a unified information theoretical model for both, classical and quantum information, represented as the memory patterns in our central main memory and integrated into our operating system. This model allows for efficient QPU simulation through implementing a hardware type agnostic intermediate representation of the quantum circuits (see Fig. 1, and e.g., Ref. [18]). The Intermediate Representation here reflects a generic model of the quantum circuits as part of the hybrid quantum algorithms. Our HPC and Quantum Simulation workloads are powered by Intel Xeon Platinum CPUs and NVIDIA A100 GPUs. Third, a unified in memory communication pro-



tocol, which makes sure that the right hardware accesses the relevant information pattern.

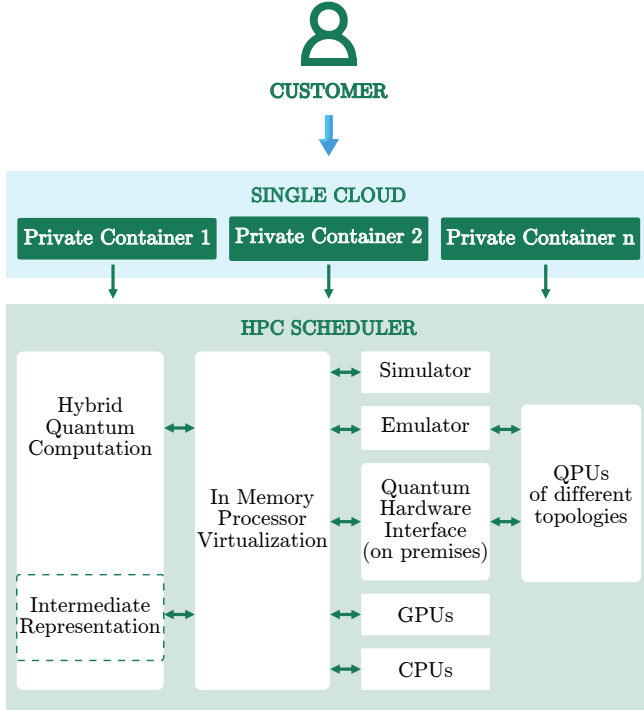


FIG. 1. QMware hybrid quantum cloud: architecture diagram. A customer accesses hardware resources via one of the private containers.

Here we focus on the hybrid algorithms, demonstrating how the combination of classical and simulated quantum resources realizes hybrid-exclusive advantages in optimization, machine learning, and simulation. In optimization, we demonstrate that our hybrid solution to the MaxCut problem showcases a performance advantage over the commercial high-performance CPLEX solver. In machine learning, we show the application-specific advantage of the hybrid quantum neural networks over their classical counterparts by studying two standard problems in the classification and regression domains. In simulation, we exhibit that the quantum-inspired tensor network differential equations' solution is more scalable than a classical conjugate gradient solution.

The paper is structured as follows. First, we introduce the hybrid quantum cloud and describe its capabilities to run hybrid quantum algorithms. Next, we demonstrate applications of hybrid quantum algorithms to optimization, machine learning, and simulation. In the summary, we discuss the impact of our developments.

II. HYBRID QUANTUM COMPUTING

To build optimal full-scale hybrid solutions, we focus on implementing quantum algorithms in a high-

performance classical environment, while evaluating the scaling on NISQ devices. While the speed-up of many quantum algorithms can be demonstrated analytically, e.g., Shor's algorithm [19], most algorithm developing initiatives benefit considerably from numerical experiments implemented within sufficiently powerful environments. For instance, variational algorithms for quantum chemistry, optimization, and machine learning that are particularly adapted for NISQ devices [20] also involve a classical heuristic optimization routine that must be analysed numerically. The key ingredients of the emerging hybridized quantum computing directions are as follows:

1. Combining optimally quantum and classical hardware resources for algorithm execution, within the given limitations of both units.
2. Implementing quantum solutions with classical improvements to eliminate bottlenecks, and using classical algorithms for optimizing quantum sub-routines.
3. Synchronizing the processing and storage of the quantum and classical information across the stack

Incorporating these directions addresses scalability, latency and memory-related challenges limiting overall performance today. The QMware cloud drives progress in these three areas and provides a system for disruptive application development around the most demanding computational problems in optimization, machine learning, and simulation, see Fig. 2.

A. Hybrid Quantum Architecture

Quantum circuits are hard to simulate classically as the time and memory needed for simulations and computational cost scales exponentially with the number of qubits [1]. Notably, however, there are classes of algorithms that scale more favourably, although still exponentially, for specific sets of quantum circuits, e.g., tensor networks [21]. Reports on several high-performance quantum circuit simulators have been published, including full state vector codes built for the CPUs [12, 22], and/or GPUs, and those that use a mix of algorithm types [23].

Our hybrid computing hardware is based on a unique memory-centric architecture, shown in Fig. 2, whereby the same main memory is accessible by all the different processing units harnessed for problem-solving, including CPUs, GPUs, QPUs, and the like, in a similar manner and to its whole extent. This uniform computing model tackles the challenges around effectively storing and processing both quantum and classical information in one system through implementing a memory bus system for synchronisation between the physical processing units and the main memory. The in-memory processor

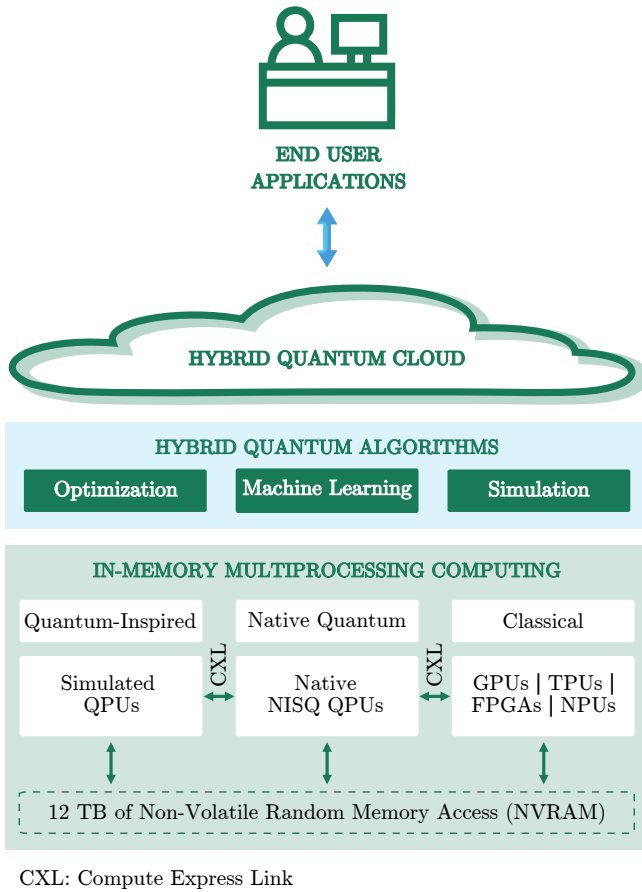


FIG. 2. QMware hybrid quantum cloud: accessing applications with quantum algorithms capabilities. Multiprocessing and in-memory compute allows for application-specific practical advantage. The integrated simulated QPUs are able to simulate up to 40 fault-tolerant qubit circuits. Native QPUs of different topologies and emulations of them are supported by our SDK, but are currently not yet integrated in the hybrid cloud.

virtualization enables the realization of simulators, emulators and physical hardware level integrations with native QPUs of all topologies from across the ecosystem in a highly efficient manner. The logical unification through an intermediate representation prevents time consuming copying of information in between process cycles. Through a system that unifies hybrid multi-processing compute, QMware enables the processor units to independently access the same central main memory. It also facilitates the scalable heterogeneous processing which is highly desirable for specific applications, e.g., for image classification using GPUs and QPUs through realizing the Compute Express Link (CXL) between processing units to scale performance.

While many hybrid algorithms feature a continuous loop of classical-quantum-classical interactions as the most time-consuming subroutine, our approach optimizes the quantum-classical interface to provide speed-ups through a hybrid quantum computing pipeline. Fur-

thermore, due to the algorithmic universality and large shared memory capabilities (12 TB per node, next version 32 TB), any quantum circuit built on our hybrid quantum cloud with virtualized qubits can also be run on upcoming native QPUs from across the ecosystem. The other way around, the QMware cloud is capable to import and run any outside developed algorithm. Additionally, the intermediate representation is downwards compatible for future hardware. This gives end users the ability to build modular hybrid quantum algorithms today while maintaining hardware flexibility in the long term.

B. Introducing Quantum into Classical

While many use-cases can be tackled using advanced quantum inspired computing, the exponential growth of the required classical resources limits the size of the simulated quantum computational space.

In the case of optimization problems, a quantum annealing [24] or a quantum approximate optimization algorithm (QAOA) [25], executed on a 40-qubit device, can handle a problem with 40 binary variables. Partitioning larger problems to be handled by these approaches significantly compromises solution quality. However, using more sophisticated encoding in the hybrid quantum algorithm allows for tackling problems with many thousands of variables, without compromising the solution quality.

Further extension of the problem complexity requires the native QPUs of the size and quality that are expected to appear on the market in forthcoming years. Many machine learning models benefit from the exponentially larger parameter space while keeping learning simple, e.g., kernel methods [26]. Enlarging the model using native quantum devices allows for more complex data analysis [27].

Quantum advantage has been demonstrated on QPUs programmed to execute random instructions that were used to mimic a quantum algorithm [28]. In the previous case, the QPU took 200 seconds to sample a quantum circuit a million times, while the execution time of the same task on a classical supercomputer is several orders of magnitude higher. Recently, a similar experiment was repeated utilizing more qubits and more complex circuits [29, 30]. Besides, Ref. 31 has implemented a quantum algorithm for a linear system solution that can be used for demonstrating an advantage of quantum phase estimation procedure, which we also use in a quantum sensing protocol [32]. Such a dramatic speed-up certainly counts as experimental evidence that for specific use cases hybrid quantum computers will surpass purely classical computers that reigned in the past.

III. APPLICATION-SPECIFIC HYBRID QUANTUM ADVANTAGE

Here, we address three industry-valuable domains, namely optimization, machine learning and simulation that posit heavy challenges and demonstrate how hybrid quantum computing can improve solution performance. These challenges and limitations are faced by classical algorithms and existing hardware across these three domains. For instance, the inability to find the global minima in feasible time for optimization challenges [33], the large energy intensiveness when training deep learning models [34] and the inability to simulate large scale complex systems [35].

A. Optimization

The problem of multiparameter optimization in the presence of multiple constraints arises in various subroutines of large-scale business management, such as an optimal resource allocation for improving efficiency, reducing risks and costs, and increasing profit. Many *discrete* optimization problems are NP-hard [36], rendering the creation of efficient methods for finding the optimal solution impossible on large enough scales. The computational challenges that appear when solving such problems include, among others, exponential increases in computational cost with increasing dimensionality and the number of local minima [33].

One of the most advanced classical software approaches to performing large-scale discrete optimizations are combinations of Simplex, the interior point-based methods, Branch and Bound algorithms, mixed-integer linear and quadratic programming, and mixed-integer constrained programming [37, 38]. These methods are implemented in commercially available solvers, such as CPLEX by IBM [39] or Gurobi Optimizer by Gurobi [40]. Many important problems have been solved by utilizing these solvers, but their large-scale performance is limited due to the likely exponential increase in optimization complexity [41]. Even without the exponential increase in complexity, the proper tuning of optimizers becomes unfeasible on a larger scale since the landscape of the cost function becomes much too complex. The latter is instrumental in solving real-life problems, such as applications in industry, where time is often a constraint and compute resources are scarce. Quantum algorithms can help in finding solutions in a faster and more accurate manner [25, 42–44].

Indeed, in the quantum computing community, discrete optimization is considered to be one of the leading candidates to demonstrate a quantum advantage in NISQ devices. Quantum annealers, in contrast to universal quantum computers, are special-purpose hardware developed explicitly for solving optimization problems and sampling tasks [24]. De facto, quantum annealing is an algorithm for solving Ising spin glasses [45] inspired

by classical simulated annealing, and quantum annealing systems implement this algorithm in quantum hardware. Such machines implement a noisy version of the Quantum Adiabatic Algorithm, the mathematical equivalent used for formulating the problem mostly being the Quadratic Unconstrained Binary Optimization (QUBO). Solving a QUBO minimizes a polynomial function of binary variables, with a degree at most two. The QUBO model is an NP-hard discrete optimization problem that lies in a cost function minimization $\min(C) = \min_{\vec{x}}(\vec{x}^T Q \vec{x})$, where $x_i \in \{0, 1\}$ is the component of a vector $\vec{x}$ of n_c binary variables and Q is a real and upper triangular matrix. Many known binary-constrained problems can be reduced to QUBO using penalties in the cost function. Remarkably, it was shown in Ref. [46] that many NP-complete and NP-hard problems, including all of Karp's 21 NP-complete problems, can be reduced to QUBO in a polynomial time.

Among the wide range of the QUBO problems, the MaxCut on an arbitrary graph [47] is often used to analyse the performance of quantum algorithms. The MaxCut problem is a search of the partition of the graph's nodes into two complementary sets, such that the sum of the weighted edges between these two sets is as large as possible. The MaxCut problem is related to logistics and planning, such as machine scheduling, traffic message management, computer-aided design, image recognition, and unsupervised machine learning problems, such as clustering and various financial optimizations. The MaxCut problem can be formulated as the QUBO problem, which lies in a minimization of the following quadratic function:

$$E = - \sum_{i,j=1}^{n_c} d_{ij} (x_i - x_j)^2, \quad (1)$$

where d_{ij} is the weight of the edge between i -th and j -th nodes in the studied graph. The solution is a binary string $\vec{x}$ of nodes' indicators that show the correspondence to one of two sets. The elements of the QUBO matrix, in turn, are $Q_{ij} = 2d_{ij}$ ($i > j$) and $Q_{ii} = -\sum_j d_{ij}$.

Inspired by quantum annealing, the optical Coherent Ising Machine (CIM) was developed to find solutions using light pulses in a hybrid electro-optical loop [48]. One of the most significant advantages of such a machine is the effective full connectivity that encodes the whole problem without sacrificing the majority of qubits to overcome the QPU connectivity issue. For instance, Ref. [49] presents a comparison between D-Wave and CIM in the MaxCut problem solution: the significant time-to-solution difference for graphs with over 50 nodes (up to 200 nodes) was observed for dense problems where CIM demonstrates better results. The difference in performance between the sparsely connected D-Wave machine and the fully connected CIMs provides strong experimental support for increasing the qubit-connectivity on quantum annealers. In general, CIM offers intriguing and prospective platforms for studying discrete optimization problems powered by the speed of light and effective

electrical feedback. However, implementing a CIM comes with formidable engineering challenges [50].

Besides specially designed hardware, one of the most promising approaches in tackling discrete optimization on universal quantum devices is variational quantum algorithms. In such algorithms, a parameterised quantum network is iteratively optimized using classical computing. A paradigm of variational circuits is close to the neural networks approach, where a deep network captures the fundamental features of the problem. For several decades, research in machine learning was focused on models that can provide theoretical guarantees for their performance. However, in recent years, methods based on heuristics have become dominant, especially for deep models, partly due to an abundance of data and computational resources. Similarly, a formal proof of quantum advantages of variational quantum algorithms was not found yet, but applications utilizing NISQ-devices to solve real-world problems using such algorithms are already being explored [20].

For instance, algorithms such as QAOA have been applied to solve NP-hard QUBO problems. However, recent experiments have highlighted the challenges in implementing the QAOA on problem graphs that differ from the native hardware topology, even for small system sizes. The latest solution of the MaxCut on a dense problem processes a 24-node graph using QAOA [51]. There are other emerging quantum approaches such as the Filtering Variational Quantum Eigensolver (F-VQE) for combinatorial optimization problems that have proved to be more performant than the original Variational Quantum Eigensolver (VQE) algorithm and QAOA [52].

Most solutions based on modern variational algorithms, e.g., QAOA, or quantum annealing, require a significant number of qubits to solve real-world MaxCut related problems. Not considering the chip topology and qubit interconnectivity, the number of qubits necessary is equal to the number of classical variables. Such an encoding requires quantum resources that current NISQ devices can't provide due to a limited number of qubits and limited qubit-interconnectivity.

Therefore, we explore novel encoding and optimization techniques for variational algorithms in a gate-based computing framework. We utilize the hardware-efficient ansatz [20], and amplitude encoding scheme [53] to transfer the classical optimization problem into the optimal quantum state search. The algorithm is called *QuEnc* because of the algorithm's underlying *Quantum Encoding* method. Inspired by available quantum machine learning tools [54], we learn the circuit parameters that provide the desired quantum state, corresponding to the optimal classical solution. The expressibility of the quantum circuit is set by the number of layers that control the number of circuit parameters. Here, we do not need to perform the full state tomography since we operate only the Z-projection of the state. The whole scheme is described in detail in Ref. [55].

We apply QuEnc to the fully connected MaxCut problem and, leveraging the amplitude encoding, we solve MaxCut with hundreds and thousands of nodes – a much larger scale than has previously been possible with quantum annealing or QAOA. Since QuEnc processes much larger problems and, therefore, can not be compared with existing quantum alternatives, we use high-performance mathematical programming solver CPLEX to analyse the performance of the QuEnc. As the solution's performance heavily depends on the classical hardware, we consider two cases: (i) the average local computing setup with 32 GB of RAM and 6 CPU cores, and (ii) the advanced hardware provided by QMware with 100 CPU cores and 12 TB of RAM. In both cases, the quantum algorithm was implemented and simulated using the <basq> Python SDK with highly efficient C++ kernels [11].

To define a MaxCut problem, we create a random weighted graph with weights of edges laying in $[0.01, 1]$, and minimize the energy function from Eq. 1. Schematic drawing of the graph is shown in Fig. 3(a).

In case (i), we focus on 256-node fully connected graph and compare CPLEX and simulated QuEnc approaches. We need to bear in mind that the exact solution of such a complex problem is impossible since we face 2^{256} possible solutions. We apply QuEnc algorithm with the fully entangled circuit presented in Fig. 3(b). Due to the amplitude encoding QuEnc provides a great advantage to solving larger problems operating as an encoder – we reduce the discrete n_c -parameter optimization to continuous $O(\log n_c)$ -parameter problem. Its convergence, reduction of the energy (cost) as a function of learning iterations, is presented in Fig. 3(c). Different colors correspond to different numbers of layers – the increase in layers leads to an increase in the number of optimized parameters. The simulated 20-layer QuEnc finds the solution with cost $-8,500$ in 30 minutes, while CPLEX finds the cost $-8,360$ in 5 hours, indicating superior performance over the hybrid QuEnc. The transfer of the simulated QuEnc to real QPU provides a further increase in speed (2-3x for that problem considering superconducting QPU). While the presence of noise is expected to limit the accuracy of the algorithm, remarkably, arising errors may even help to avoid local minima during the convergence that only benefits the accuracy. The comprehensive study of such an important issue is a subject of further work.

In case (ii), CPLEX as a well-tuned and flexible solver leveraging the full power of QMware can compete with QuEnc and finds the solution with even better cost value. Therefore, we are able to consider a larger scale problem with a 1024-node graph, where the optimization landscape is much more complex. Using encoding techniques helped us to find the solution with good cost value very fast, but usually that solution could be improved upon even more. Here, to leverage the whole power of QuEnc we introduce the hybrid pipeline with the high-performance classical solver. Mainly, we presolve the

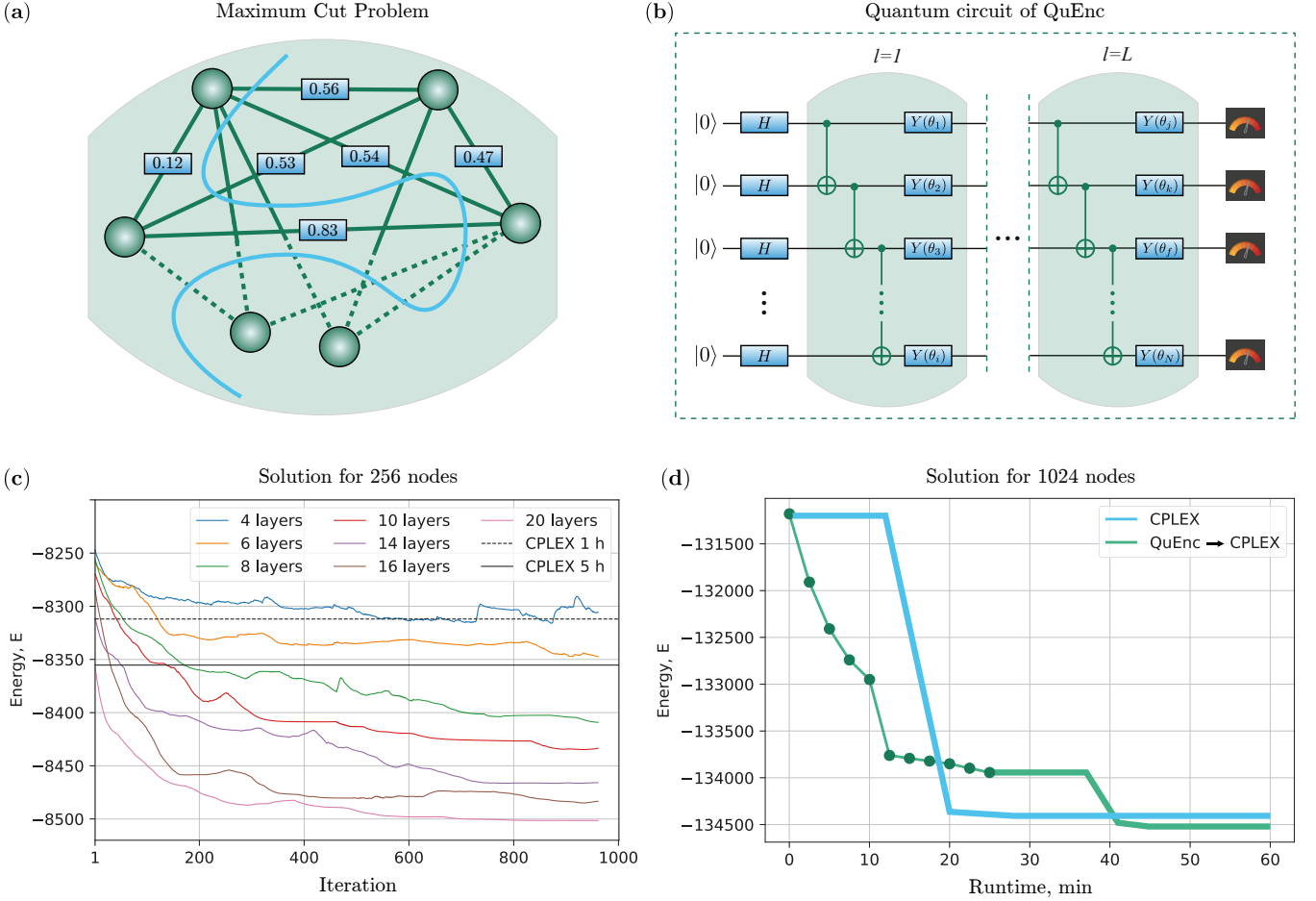


FIG. 3. (a) The schematic drawing of the weighed fully connected graph. The cut that gives the largest value is called Maximum Cut (orange line) and the search of such a cut is an NP-hard problem. (b) The quantum circuit used in the QuEnc algorithm with fully entangled quantum state. The circuits parameterised are optimized to sample the classical solution with the lowest energy (cost) value. (c) The QuEnc and CPLEX performance on complete graphs with 256 nodes. Our quantum algorithm with ≥ 20 layers finds significantly more accurate solution with lower energy in 1 minute than the CPLEX in 5 hours. (d) The cost function obtained via quantum algorithm and high-performance CPLEX on complete weighted graphs with 1024 nodes as function of runtime. The solution obtained by simulating quantum circuit with 50 layers in 25 minutes and utilization the resulting solution as an initial point for the CPLEX solver, which improves the obtained solution for 35 minutes. Such a hybrid pipeline finds more accurate solution with lower energy in 1 hour than the pure CPLEX.

problem using QuEnc obtaining the solution with good cost and then use that solution as the initial point for the CPLEX solver.

The QuEnc and CPLEX convergence, reduction of the energy (cost) as function of execution time, is presented in Fig. 3(d). Green dots show the QuEnc solution convergence, which reach $-13,400$ in 25 minutes. The green line is the convergence of the CPLEX that starts from the point that was found by QuEnc. The plato arises due to the CPLEX internal processes, such as tree building. The blue line corresponds to the pure CPLEX solution started at a random point, whose cost is close to the initial cost of QuEnc that also started from a random point. The QuEnc $\rightarrow$ CPLEX pipeline finds better cost of $-134,520$ than CPLEX, $-134,406$, providing 0.085% improvement. We expect that the improvement would be

more prominent with the increase in the problem size and enhancing QuEnc that falls within the scope of future work.

B. Machine Learning

Quantum machine learning, both improving quantum technologies with artificial intelligence and enhancing classical machine learning utilizing quantum effects, vastly illustrates the power of hybrid quantum computing.

First, classical machine learning is becoming increasingly more important for a variety of tasks in quantum information technologies [57–59]. For instance, reinforce-

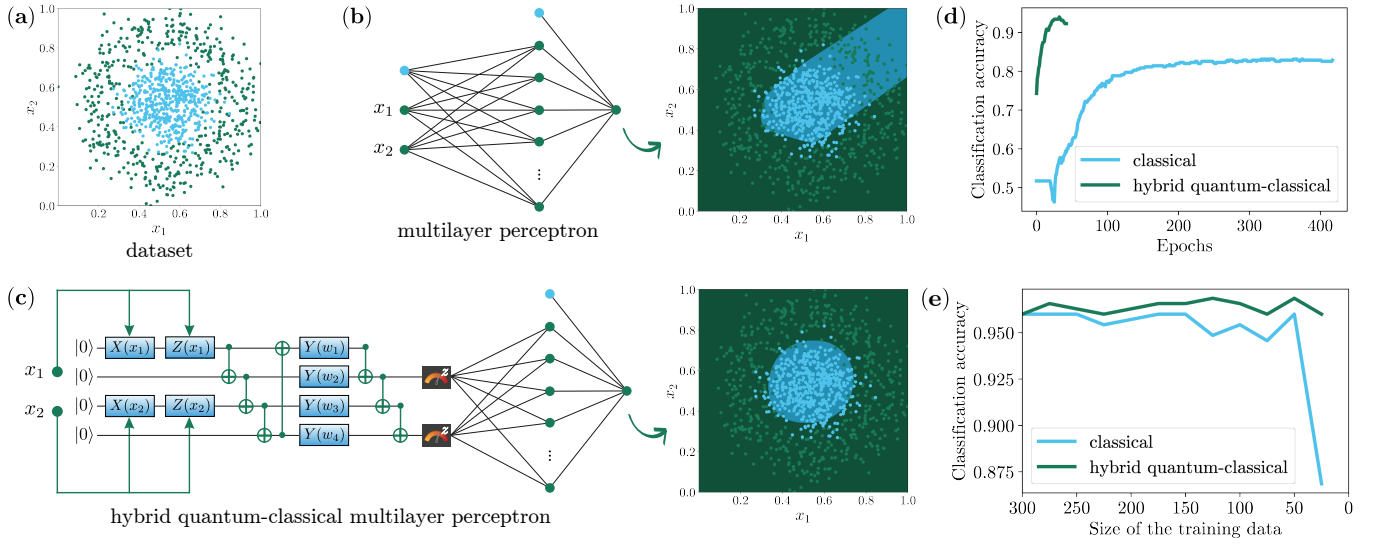


FIG. 4. (a) Circles dataset from Scikit [56]. 1000 points are generated: 300 for training procedure and 700 for evaluating model’s performance. (b) Classic neural network’s architecture. A classification function learned by this network is a dependence of the output class on the coordinates x_1 and x_2 . (c) Hybrid quantum neural network’s architecture. A classification function learned by this network is a dependence of the output class on the coordinates x_1 and x_2 . (d) Accuracy on test data for each epoch during the training procedure. Results are averaged over 10 independent models. (e) Dependence of the classification accuracy on the training data size.

ment learning agents are being used for control [60–62], error-correction [63, 64], and designing new quantum protocols and experiments [65–67]. The latter is motivated by the unknown reachability of various configurations in quantum experiments [68, 69]. These works show that machine learning can offer dramatic advances in how complicated experiments are generated. In addition to reinforcement learning, supervised learning systems were found useful for, e.g., studying quantum advantage over a classical approach [70–72], reconstructing quantum states of physical systems [73–76], and learning compact representations of these states [77, 78]. These studies have revealed that deep learning networks can identify complex patterns and trends in data. It would not be possible without powerful computers and special-purpose hardware capable of implementing deep networks with billions of parameters [79]. If machine learning implemented on HPC could substantially improve quantum devices, the potential impact would be tremendous.

Second, quantum technologies can massively assist the most advanced machine learning frameworks – highly autonomous systems that outperform humans at most economically valuable work require considerable computational resources, limiting their performance. Quantum computing models can potentially improve the training process of existing classical models [57, 80–82], which allows for finding better extreme points in an objective function landscape or the same optima with fewer iterations. These methods allow for polynomial speedups, which are crucial for large and complex problems, where minor improvements give noticeable gains. Besides, the recent experiments show that quantum models can sam-

ple intricate probability distributions in a polynomial time [83], while the same classical sampling could be exponentially difficult. Among many other methods, the most promising are quantum neural networks [54, 84–87] and quantum kernels [88] that are expected to beat classical models with current noisy quantum devices.

Since NISQ devices limit the freedom in the machine learning model choice, we focus on the hybrid pipelines for classification and regression as the most suitable approaches. Hybrid quantum-classical solvers implemented in the hybrid quantum cloud can provide higher efficiency in training by requiring lesser iterations, and can show a higher prediction accuracy as we demonstrate next.

1. Hybrid Machine Learning Advantages in Classification

To illustrate the difference that quantum circuits introduce in machine learning, we first consider a standard benchmark Scikit dataset [56] used to test small-scale classification algorithms. The dataset can be visualized as a large circle containing a smaller circle in 2D, see Fig. 4(a).

As a classical machine learning solution to this binary classification problem, we use a multilayer perceptron model with 3 neural network layers, shown in Fig. 4(b). The first layer consists of 2 input neurons and a bias neuron, followed by a hidden layer with 40 neurons and a bias, and a single output neuron. Neural network layers are fully connected leading to 161 weights in the network.

As a hybrid quantum machine learning solution to this

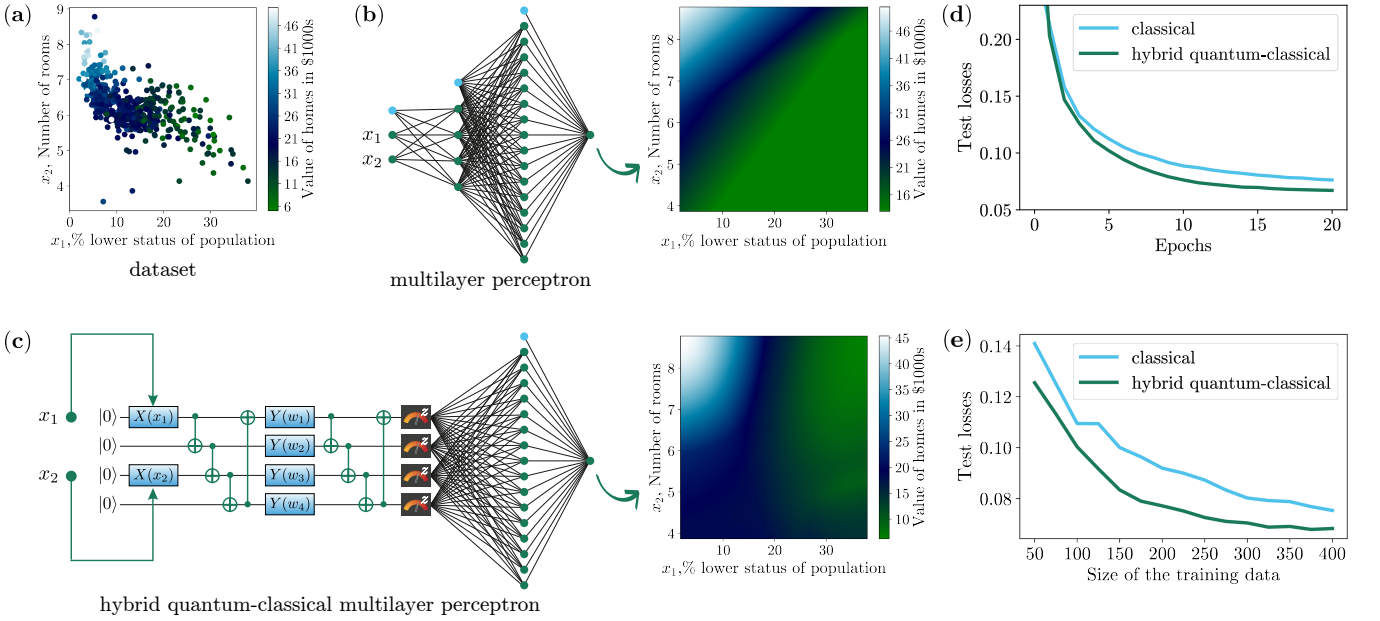


FIG. 5. (a) Boston Housing dataset from Scikit [89]. (b) Classic neural network's architecture. A function learned by this network is a dependence of the value of houses on the number of rooms and the % of lower status of population. (c) Hybrid quantum neural network's architecture. A function learned by this network is a dependence of the value of houses on the number of rooms and the % of lower status of population. (d) Loss on the test data for epoch during the training procedure. Results are averaged over 100 independent models. (e) Dependence of the test loss on the training data size.

binary classification problem, we use a hybrid quantum-classical multilayer perceptron. This hybrid quantum neural network consists of both quantum and classical neural network parts: a 4-qubit quantum circuit followed by 3 layers of neurons for a classical part, shown in Fig. 4(c). The classical part of the hybrid model is the same as of a classical multilayer perceptron, but without a bias neuron. The quantum circuit has 4 variational parameters, leading to a total of 125 weights in the hybrid quantum neural network (HQNN).

On Fig. 4(d) one can see the training procedure of the best model over hundred independent models. Both models have been trained with stochastic gradient descent, Adam optimizer with learning rate 1×10^{-2} , and the Binary cross entropy loss function. The HQNN algorithm achieves a 13% higher accuracy on the test data (0.831 and 0.940 accuracy for the classical and the hybrid model, respectively), but also converges much faster than the classical counterpart (317 and 32 epochs for the classical and the hybrid model, respectively). On Figs. 4(b)-(c) one can additionally see a qualitative difference in how well both models separated data points belonging to different classes with the training size of 25 samples.

Moreover, by reducing the size of the training data, the classical model's ability to learn decreases substantially. Meanwhile, the HQNN does not show any difference and demonstrates accuracy above 90%, as can be seen in Fig. 4(e). The obtained results are significant since the most difficult part of commercial machine learning tasks is collecting data and labelling it. Therefore,

it is very useful that this hybrid model manages to learn well even on small datasets.

The HQNN model used to solve the binary classification problem, can be used for a continuous variable output, namely it can deal with regression problems as we show next.

2. Hybrid Machine Learning Advantages in Regression

The regression problem under consideration is represented by the Boston housing dataset [89]. This dataset is used as a comparison of machine learning models, testing of various algorithms and based on information gathered by the US Census Service regarding housing in the Massachusetts capital, Boston. The dataset contains 506 samples and 13 feature variables such as average number of rooms per dwelling, pupil-teacher ratio, and per capita crime rate. Our goal is to determine the value of the median price of owner-occupied homes based on these features. The data was originally published in [90]. The title of the article suggests that people are willing to pay more for clean air and that the price of houses depends on the surrounding area. It is worth noting that house prices do not exceed 50,000\$. This is due to the fact that the Census Service censored the data and put a price cap of 50,000\$. After all these years, it is impossible to reliably know the real prices of houses.

The data was downloaded using the Scikit-learn li-

brary [89]. After analysing the data, we find a relationship between the target variable with other features and selected two features: number of rooms (average number of rooms per dwelling) and LSTAT (percentage of lower status of the population). We built a machine learning model for determining the prices of Boston houses based on two features we selected. Next, we split the dataset into training and test samples, 80% and 20%, respectively.

Similar to the classification task, we use the multilayer perceptron for solving the regression problem. The architecture of our classic network is shown in Fig. 5(b) and consists of three linear layers with the ReLU activation functions. As for the hybrid neural network, we construct it by replacing the first classical fully-connected layer with a quantum variational layer with four qubits, as shown in Fig. 5(c). Since we decided to use only two features, the dimension of the input data is equal to two. Therefore, we encoded the input information into the angles of rotations along the x -axis on the first and third qubits. It should be noted that the number of parameters in our quantum variation circuit is four versus twelve parameters in the first classical layer in the classical analogue of our hybrid network.

We train our classical and hybrid models using stochastic gradient descent, Adam optimizer with the learning rate 3×10^{-3} and compute the mean squared error loss function. Mean absolute error is used to evaluate the model's performance on the test set. To compare the classical and hybrid neural networks, in Fig. 5(d) we plot depicting the dependence of test losses on the number of epochs during the training procedure. This plot was obtained by averaging the results across one hundred independent models. The HQNN has an advantage over the classical neural network, as the test loss is 12% lower in the hybrid case (0.076 and 0.067 loss values for the classical and hybrid models, respectively).

Another interesting feature is scaling. In Fig. 5(e) we show test loss as a function of the number of training data samples used for training. Every point of this plot, taken every 50 steps in training data size, is obtained by averaging across 100 independent models. Similar to the classification problem, in the regression problem we observe that the hybrid quantum advantage is robust, and the quantum advantage is observable in an entire range of training data sizes. The advantage that we observe ranges between 12% improvement for the training set size of 400 (see Fig. 5(d)) and 16% improvement for the training set size of 125 (0.109 and 0.092 loss values for the classical and the hybrid model, respectively). Similar to the Circles Scikit dataset, we observe a better advantage for smaller dataset sizes.

C. Quantum-inspired simulation

The tantalizing goal of quantum computing is to per-

form calculations beyond the reach of any classical computer, e.g., high-performance modelling of complex physical and biological systems. The NISQ devices suffer from many sources of errors, which limit the degree of entanglement and their current performance. However, quantum-enabled algorithms, even being implemented via a tensor network on classical hardware, can provide an advantage at solving certain problems.

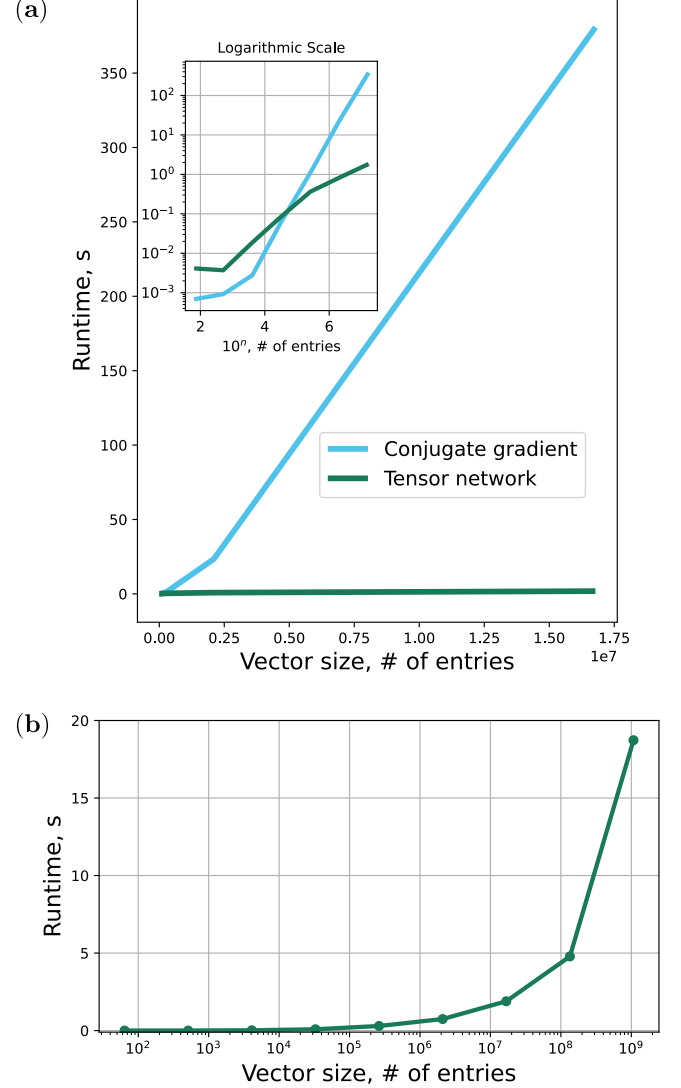


FIG. 6. (a) The runtime of the Poisson equation solution as a function of the space discretisation for the conjugate gradient method (black) and tensor networks (red). The embedding shows the log-log dependency. Tensor networks provide an exponential advantage in the solution of such a problem. (b) The runtime of the Poisson equation solution as a function of the space discretization for the tensor network-based method. Such a method allows solving problem with 10^9 points in space discretisation in less than 20 seconds using just two CPUs.

In general, tensor decompositions and tensor networks, which were initially introduced in quantum physics for multiparticle system analysis [91], are emerging as

promising methods for high-dimensional problems, simulation in particular. The main advantage of tensor networks is logarithmic scaling in the studied problem's dimension for some tasks, which is similar to scaling expected in quantum computers. Moreover, the topology of tensor networks is similar to quantum circuits architecture, which makes them an efficient tool for low-entangled quantum computer virtualization and simulation of certain physical systems [92].

Many physical and biological simulations depend on solving partial differential equations describing the physical processes behind studied systems. Using tensor decomposition, one can find an efficient way to solve differential equations saving memory and speeding up the solution.

Here, as an illustrative example, we study the solution of second-order linear partial differential equations via tensor networks. Mainly, we consider the Poisson equation, which is a generalization of Laplace's equation, that is frequently used in various areas of science and engineering, e.g., computational fluid dynamics [93], electrostatics [94], the theory of Markov chains [95], and density functional theory and electronic structure calculations [96].

We solve the Poisson equation using tensor networks by discretisation on a regular grid in Cartesian coordinates and processing the resulting system of linear equations. Leveraging the fact that the matrix corresponding to the second-order derivative can be represented as a matrix product operator [97], we can solve such a system of linear equations employing a method based on an alternating minimal energy solver [98]. Those types of algorithms provide polylogarithmic scaling in runtime and memory, offering exponential speedup in comparison with conventional classical methods that have polynomial scaling.

Let us consider the solution of the Poisson equation in a 3-dimensional space with zero boundary condition

$$-\Delta u = 1 \text{ in } \Omega = [0, 1]^3, \quad u|_{\partial\Omega} = 0.$$

We reduce the differential equation to a linear system, represent it as a tensor network, process and contract tensors to obtain a classical solution. In order to verify the solution we consider one of the most powerful tools in a linear systems solver – a conjugate gradient method. The runtime of the solution for both a tensor network and conjugate gradient methods as a function of discretization accuracy, number of grid points, is presented in Fig. 6(a). It is clear that the tensor network solver provides an exponentially faster solution in comparison with classical approaches with similar scaling as proposed quantum algorithms [99]. However, in com-

parison with NISQ-device implementation, our method allows us to solve large-scale problems in seconds, as presented in Fig. 6(b). Here, we solve the Poisson equation on 10^9 spatial points with perfect fidelity via tensor networks in less than 20 seconds using just two Intel Xeon 2.2 GHz CPUs and 12 Gb of RAM.

Leveraging the full power of the QMware hybrid quantum cloud, a tensor network approach can handle larger and more complex problems that will be studied in the future. Such a method can be extended to more complex differential equations in many areas, including computational fluid dynamics, e.g., Boltzmann equation, financial simulations, e.g., Fokker-Planck equation, aerodynamics and heat transfer, e.g., parabolic differential equations.

IV. CONCLUSION

This paper presents a hybrid quantum architecture for hardware and software targeting maximizing application-specific practical advantage today. This architecture puts forth hybrid quantum algorithms for optimization, machine learning and simulation. We demonstrate solution performance for problems across these application domains, through examples of discrete graph-based optimization, classification, regression and partial differential equations. In the MaxCut optimization problem with 256 nodes with the limited hardware, the QuEnc algorithm provided a 1.7% better solution in 1 minute than the CPLEX solver in 5 hours. The hybrid QuEnc algorithm achieved a 0.085% better solution on a 1024 node MaxCut problem at the fixed runtime. Next, in classification, the hybrid quantum neural network provided, compared to a classical analogue, a 13% higher accuracy and a much better convergence time (32 epochs instead of 317) on the Circles dataset. In regression, the hybrid quantum neural network outperformed the classical analogue by 12-16% depending on the training set size. Finally, in solving the Poisson equation, tensor networks were shown to be exponentially faster than the conjugate gradient method providing an improvement in a runtime by several orders of magnitude.

Through our hybrid quantum cloud, QMware, we hence make these demonstrated benefits accessible in an industrial context. The introduced hybrid quantum-classical approach looks to meet the need of industrial users to get the best compute results for their applications, irrespective of whether the underlying hardware is classical or quantum. This paves the way toward accelerating quantum adoption for the benefit of business and society.

[1] J. Preskill, Quantum computing in the NISQ era and beyond, *Quantum* **2**, 79 (2018).

[2] K. Bharti, A. Cervera-Liarta, T. H. Kyaw, T. Haug,



- S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, *et al.*, Noisy intermediate-scale quantum algorithms, *Rev. Mod. Phys.* **94**, 015004 (2022).
- [3] S. Bravyi, G. Smith, and J. A. Smolin, Trading classical and quantum computational resources, *Phys. Rev. X* **6**, 021043 (2016).
- [4] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, The theory of variational hybrid quantum-classical algorithms, *New J. Phys.* **18**, 023023 (2016).
- [5] J. Li, X. Yang, X. Peng, and C.-P. Sun, Hybrid quantum-classical approach to quantum optimal control, *Phys. Rev. Lett.* **118**, 150503 (2017).
- [6] D. Zhu, N. M. Linke, M. Benedetti, K. A. Landsman, N. H. Nguyen, C. H. Alderete, A. Perdomo-Ortiz, N. Korda, A. Garfoot, C. Brecque, *et al.*, Training of quantum circuits on a hybrid quantum computer, *Sci. Adv.* **5**, eaaw9918 (2019).
- [7] A. Gheorghiu, T. Kapourniotis, and E. Kashefi, Verification of quantum computation: An overview of existing approaches, *Theory Comput. Syst.* **63**, 715 (2019).
- [8] J. Eisert, D. Hangleiter, N. Walk, I. Roth, D. Markham, R. Parekh, U. Chabaud, and E. Kashefi, Quantum certification and benchmarking, *Nat. Rev. Phys.* **2**, 382 (2020).
- [9] A. A. Semnani, J. Pham, B. Englert, and X. Wu, Virtualization technology and its impact on computer hardware architecture, in *8th Int. Conf. Inf. Technol. Proc.* (IEEE, 2011) pp. 719–724.
- [10] M. Bechtold, Master’s thesis (2021), Bringing the concepts of virtualization to gate-based quantum computing.
- [11] QMware: The first global quantum cloud, <https://qmware.com> (2022).
- [12] G. G. Guerreschi, J. Hogaboam, F. Baruffa, and N. P. D. Sawaya, Intel quantum simulator: a cloud-ready high-performance simulator of quantum circuits, *Quantum Sci. Technol.* **5**, 034007 (2020).
- [13] S. Mandrà, J. Marshall, E. G. Rieffel, and R. Biswas, Hybridq: A hybrid simulator for quantum circuits, in *2nd Int. Workshop on Quantum Comput. Softw.* (IEEE/ACM, 2021) pp. 99–109.
- [14] ATOS Quantum Learning Machine, <https://atos.net/en/solutions/quantum-learning-machine> (2022).
- [15] S. Efthymiou, S. Ramos-Calderer, C. Bravo-Prieto, A. Pérez-Salinas, D. García-Martín, A. García-Saez, J. I. Latorre, and S. Carrazza, Qibo: a framework for quantum simulation with hardware acceleration, *Quantum Sci. Technol.* **7**, 015018 (2021).
- [16] J. Von Neumann, First draft of a report on the EDVAC, *IEEE Ann. Hist. Comput.* **15**, 27 (1993).
- [17] M. Mariantoni, H. Wang, T. Yamamoto, M. Neeley, R. C. Bialczak, Y. Chen, M. Lenander, E. Lucero, A. D. O’Connell, D. Sank, *et al.*, Implementing the quantum von neumann architecture with superconducting circuits, *Science* **334**, 61 (2011).
- [18] QIR Alliance, <https://github.com/qir-alliance> (2022).
- [19] P. W. Shor, Algorithms for quantum computation: discrete logarithms and factoring, in *Proceedings 35th AS-FCS* (IEEE Comput. Soc. Press, 1994).
- [20] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, and P. J. Coles, Variational quantum algorithms, *Nat. Rev. Phys.* **3**, 625 (2021).
- [21] Y.-Y. Shi, L.-M. Duan, and G. Vidal, Classical simulation of quantum many-body systems with a tree tensor network, *Phys. Rev. A* **74** (2006).
- [22] J. Niwa, K. Matsumoto, and H. Imai, General-purpose parallel simulator for quantum computing, in *Unconventional Models of Computation* (Springer Berlin Heidelberg, 2002) pp. 230–251.
- [23] T. Jones, A. Brown, I. Bush, and S. C. Benjamin, QuEST and high performance simulation of quantum computers, *Sci. Rep.* **9** (2019).
- [24] S. E. Venegas-Andraca, W. Cruz-Santos, C. McGeoch, and M. Lanzagorta, A cross-disciplinary introduction to quantum annealing-based algorithms, *Contemp. Phys.* **59**, 174 (2018).
- [25] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, *arXiv:1411.4028* (2014).
- [26] G. Pillonetto, F. Dinuzzo, T. Chen, G. D. Nicolao, and L. Ljung, Kernel methods in system identification, machine learning and function estimation: A survey, *Automatica* **50**, 657 (2014).
- [27] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, *Nat. Phys.* **17**, 1013 (2021).
- [28] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, *et al.*, Quantum supremacy using a programmable superconducting processor, *Nature* **574**, 505 (2019).
- [29] Y. Wu, W.-S. Bao, S. Cao, F. Chen, M.-C. Chen, and X. Chen, Strong quantum computational advantage using a superconducting quantum processor, *arXiv:2106.14734* (2021).
- [30] Q. Zhu, S. Cao, F. Chen, M.-C. Chen, X. Chen, and T.-H. Chung, Quantum computational advantage via 60-qubit 24-cycle random circuit sampling, *arXiv:2109.03494* (2021).
- [31] M. R. Perelshtein, A. I. Pakhomchik, A. A. Melnikov, A. A. Novikov, A. Glatz, G. S. Paraoanu, V. M. Vinokur, and G. B. Lesovik, Solving large-scale linear systems of equations by a quantum hybrid algorithm, *arXiv:2003.12770* (2020).
- [32] M. R. Perelshtein, N. S. Kirsanov, V. V. Zemlyanov, A. V. Lebedev, G. Blatter, V. M. Vinokur, and G. B. Lesovik, Linear ascending metrological algorithm, *Phys. Rev. Research* **3**, 013257 (2021).
- [33] P. C. Cheeseman, B. Kanefsky, W. M. Taylor, *et al.*, Where the really hard problems are, in *IJCAI*, Vol. 91 (1991) pp. 331–337.
- [34] E. Strubell, A. Ganesh, and A. McCallum, Energy and policy considerations for deep learning in NLP, *arXiv:1906.02243* (2019).
- [35] I. M. Georgescu, S. Ashhab, and F. Nori, Quantum simulation, *Reviews of Modern Physics* **86**, 153 (2014).
- [36] R. M. Karp, Reducibility among combinatorial problems (Springer US, 1972) pp. 85–103.
- [37] E. L. Lawler and D. E. Wood, Branch-and-Bound methods: A survey, *Oper. Res.* **14**, 699 (1966).
- [38] J. Dongarra and F. Sullivan, Guest editors’ introduction to the top 10 algorithms, *IEEE Comput. Archit. Lett.* **2**, 22 (2000).
- [39] IBM ILOG CPLEX, User’s manual for CPLEX, International Business Machines Corporation **46**, 157 (2009).
- [40] Gurobi Optimization, LLC, Gurobi Optimizer Reference Manual (2021).
- [41] D. E. Knuth, Postscript about NP-hard problems, *ACM*

- SIGACT News **6**, 15 (1974).
- [42] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda, A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem, *Science* **292**, 472 (2001).
 - [43] N. Moll, P. Barkoutsos, L. S. Bishop, J. M. Chow, A. Cross, D. J. Egger, S. Filipp, A. Fuhrer, J. M. Gambetta, M. Ganzhorn, *et al.*, Quantum optimization using variational algorithms on near-term quantum devices, *Quantum Sci. Technol.* **3**, 030503 (2018).
 - [44] S. Marsh and J. B. Wang, Combinatorial optimization via highly efficient quantum walks, *Phys. Rev. Research* **2**, 023302 (2020).
 - [45] A. Das and B. K. Chakrabarti, eds., *Quantum Annealing and Other Optimization Methods* (Springer Berlin Heidelberg, 2005).
 - [46] A. Lucas, Ising formulations of many NP problems, *Front. Phys.* **2** (2014).
 - [47] G. Ausiello, A. Marchetti-Spaccamela, P. Crescenzi, G. Gambosi, M. Protasi, and V. Kann, *Complexity and Approximation* (Springer Berlin Heidelberg, 1999).
 - [48] F. Böhm, G. Verschaefelt, and G. V. der Sande, A poor man's coherent ising machine based on opto-electronic feedback systems for solving optimization problems, *Nat. Commun.* **10** (2019).
 - [49] R. Hamerly, T. Inagaki, P. L. McMahon, D. Venturelli, A. Marandi, *et al.*, Experimental investigation of performance differences between coherent ising machines and a quantum annealer, *Sci. Adv.* **5**, eaau0823 (2019).
 - [50] Y. Yamamoto, K. Aihara, T. Leleu, K. ichi Kawarabayashi, S. Kako, M. Fejer, K. Inoue, and H. Takesue, Coherent ising machines — optical neural networks operating at the quantum limit, *npj Quantum Inf.* **3** (2017).
 - [51] M. P. Harrigan, K. J. Sung, M. Neeley, K. J. Satzinger, F. Arute, *et al.*, Quantum approximate optimization of non-planar graph problems on a planar superconducting processor, *Nat. Phys.* **17**, 332 (2021).
 - [52] D. Amaro, C. Modica, M. Rosenkranz, M. Fiorentini, M. Benedetti, and M. Lubasch, Filtering variational quantum algorithms for combinatorial optimization, *arXiv:2106.10055* (2021).
 - [53] B. Tan, M.-A. Lemonde, S. Thanasilp, J. Tangpanitanon, and D. G. Angelakis, Qubit-efficient encoding schemes for binary optimisation problems, *Quantum* **5**, 454 (2021).
 - [54] M. Broughton, G. Verdon, T. McCourt, A. J. Martinez, J. H. Yoo, *et al.*, TensorFlow Quantum: A software framework for quantum machine learning, *arXiv:2003.02989* (2020).
 - [55] M. R. Perelshtein and A. I. Pakhomchik, Hardware-efficient hybrid quantum algorithm for discrete optimization, *Patent* (2021).
 - [56] Scikit, Circles Dataset, https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_circles.html (Accessed: 2021-10-21).
 - [57] V. Dunjko and H. J. Briegel, Machine learning & artificial intelligence in the quantum domain: a review of recent progress, *Rep. Prog. Phys.* **81**, 074001 (2018).
 - [58] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Machine learning and the physical sciences, *Rev. Mod. Phys.* **91**, 045002 (2019).
 - [59] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, *Nature* **549**, 195 (2017).
 - [60] T. Fösel, P. Tighineanu, T. Weiss, and F. Marquardt, Reinforcement learning with neural networks for quantum feedback, *Phys. Rev. X* **8**, 031084 (2018).
 - [61] M. Bukov, A. G. R. Day, D. Sels, P. Weinberg, A. Polkovnikov, and P. Mehta, Reinforcement learning in different phases of quantum control, *Phys. Rev. X* **8**, 031086 (2018).
 - [62] H. Xu, J. Li, L. Liu, Y. Wang, H. Yuan, and X. Wang, Generalizable control for quantum parameter estimation through reinforcement learning, *npj Quantum Inf.* **5**, 82 (2019).
 - [63] H. Poulsen Nautrup, N. Delfosse, V. Dunjko, H. J. Briegel, and N. Friis, Optimizing quantum error correction codes with reinforcement learning, *Quantum* **3**, 215 (2019).
 - [64] R. Sweke, M. S. Kesselring, E. P. van Nieuwenburg, and J. Eisert, Reinforcement learning decoders for fault-tolerant quantum computation, *Mach. Learn.: Sci. Technol.* **2**, 025005 (2020).
 - [65] A. A. Melnikov, H. Poulsen Nautrup, M. Krenn, V. Dunjko, M. Tiersch, A. Zeilinger, and H. J. Briegel, Active learning machine learns to create new quantum experiments, *Proc. Natl. Acad. Sci. U.S.A.* **115**, 1221 (2018).
 - [66] J. Wallnöfer, A. A. Melnikov, W. Dür, and H. J. Briegel, Machine learning for long-distance quantum communication, *PRX Quantum* **1**, 010301 (2020).
 - [67] A. A. Melnikov, P. Sekatski, and N. Sangouard, Setting up experimental Bell tests with reinforcement learning, *Phys. Rev. Lett.* **125**, 160401 (2020).
 - [68] M. Krenn, M. Malik, R. Fickler, R. Lapkiewicz, and A. Zeilinger, Automated search for new quantum experiments, *Phys. Rev. Lett.* **116**, 090405 (2016).
 - [69] M. Krenn, M. Erhard, and A. Zeilinger, Computer-inspired quantum experiments, *Nat. Rev. Phys.* **2**, 649 (2020).
 - [70] A. A. Melnikov, L. E. Fedichkin, and A. Alodjants, Predicting quantum advantage by quantum walk with convolutional neural networks, *New J. Phys.* **21**, 125002 (2019).
 - [71] A. A. Melnikov, L. E. Fedichkin, R.-K. Lee, and A. Alodjants, Machine learning transfer efficiencies for noisy quantum walks, *Adv. Quantum Technol.* **3**, 1900115 (2020).
 - [72] C. Moussa, H. Calandra, and V. Dunjko, To quantum or not to quantum: towards algorithm selection in near-term quantum optimization, *Quantum Sci. Technol.* **5**, 044009 (2020).
 - [73] S. Yu, F. Albarrán-Arriagada, J. C. Retamal, Y.-T. Wang, W. Liu, Z.-J. Ke, Y. Meng, Z.-P. Li, J.-S. Tang, E. Solano, L. Lamata, C.-F. Li, and G.-C. Guo, Reconstruction of a photonic qubit state with reinforcement learning, *Adv. Quantum Technol.* **2**, 1800074 (2019).
 - [74] G. Torlai, B. Timar, E. P. L. van Nieuwenburg, H. Levine, A. Omran, *et al.*, Integrating neural networks with a quantum simulator for state reconstruction, *Phys. Rev. Lett.* **123**, 230504 (2019).
 - [75] A. M. Palmieri, E. Kovlakov, F. Bianchi, D. Yudin, S. Straupe, J. D. Biamonte, and S. Kulik, Experimental neural network enhanced quantum tomography, *npj Quantum Inf.* **6** (2020).
 - [76] Y. Ding, J. D. Martín-Guerrero, M. Sanz, R. Magdalena-Benedicto, X. Chen, and E. Solano, Retrieving quantum information with active learning, *Phys. Rev. Lett.* **124**, 140504 (2020).

- [77] G. Carleo and M. Troyer, Solving the quantum many-body problem with artificial neural networks, *Science* **355**, 602 (2017).
- [78] X. Gao and L.-M. Duan, Efficient representation of quantum many-body states with deep neural networks, *Nat. Commun.* **8**, 662 (2017).
- [79] Q. V. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. S. Corrado, J. Dean, and A. Y. Ng, Building high-level features using large scale unsupervised learning, in *29th Int. Conf. Mach. Learn.* (2012).
- [80] H. Neven, V. S. Denchev, G. Rose, and W. G. Macready, QBoost: Large scale classifier training with adiabatic quantum optimization, in *Proc. Asian Conf. Mach. Learn.*, Proceedings of Machine Learning Research, Vol. 25, edited by S. C. H. Hoi and W. Buntine (PMLR, 2012) pp. 333–348.
- [81] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum support vector machine for big data classification, *Phys. Rev. Lett.* **113**, 130503 (2014).
- [82] V. Saggio, B. E. Asenbeck, A. Hamann, T. Strömberg, P. Schiansky, V. Dunjko, N. Friis, N. C. Harris, M. Hochberg, D. Englund, *et al.*, Experimental quantum speed-up in reinforcement learning agents, *Nature* **591**, 229 (2021).
- [83] A. P. Lund, M. J. Bremner, and T. C. Ralph, Quantum sampling problems, *BosonSampling and quantum supremacy*, npj Quantum Inf. **3**, 1 (2017).
- [84] E. Farhi and H. Neven, Classification with quantum neural networks on near term processors, arXiv:1802.06002 (2018).
- [85] P. Rebentrost, T. R. Bromley, C. Weedbrook, and S. Lloyd, Quantum Hopfield neural network, *Phys. Rev. A* **98**, 042308 (2018).
- [86] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babush, and H. Neven, Barren plateaus in quantum neural network training landscapes, *Nat. Commun.* **9**, 1 (2018).
- [87] K. Beer, D. Bondarenko, T. Farrelly, T. J. Osborne, R. Salzmann, D. Scheiermann, and R. Wolf, Training deep quantum neural networks, *Nat. Commun.* **11**, 1 (2020).
- [88] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, *Nature* **567**, 209 (2019).
- [89] Scikit, Boston Housing Dataset, https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_boston.html (Accessed: 2021-10-21).
- [90] D. Harrison Jr and D. L. Rubinfeld, Hedonic housing prices and the demand for clean air, *J. Environ. Econ. Manag.* **5**, 81 (1978).
- [91] S. R. White, Density matrix formulation for quantum renormalization groups, *Phys. Rev. Lett.* **69**, 2863 (1992).
- [92] Y. Zhou, E. M. Stoudenmire, and X. Waintal, What limits the simulation of quantum computers?, *Phys. Rev. X* **10** (2020).
- [93] G. K. Batchelor, An introduction to fluid dynamics, in *Cambridge University Press, Cambridge, UK* (2000).
- [94] D. J. Griffiths, Introduction to Electrodynamics, in *Prentice Hall, Upper Saddle River, NJ* (1999).
- [95] S. P. Meyn and R. L. Tweedie, Markov chains and stochastic stability, in *Cambridge University Press* (2009).
- [96] E. Engel and R. M. Dreizler, Density functional theory: An advanced course, in *Springer, New York* (2011).
- [97] V. Kazeev and B. Khoromskij, Low-rank explicit QTT representation of the laplace operator and its inverse, *SIAM J. Matrix Anal. Appl.* **33** (2012).
- [98] S. Dolgov and D. Savostyanov, Alternating minimal energy methods for linear systems in higher dimensions, *SIAM J. Sci. Comput.* **36**, 1 (2014).
- [99] S. Wang, Z. Wang, W. Li, L. Fan, Z. Wei, and Y. Gu, Quantum fast Poisson solver: the algorithm and complete and modular circuit design, *Quantum Inf. Process.* **19** (2020).



By Terra Quantum AG
May, 2022