Quantum Computing for Data Analysis in High-energy Physics[*]

Andrea Delgado,[†] Kathleen E. Hamilton, and Prasanna Date
Oak Ridge National Laboratory, Oak Ridge, Tennessee USA

Jean-Roch Vlimant[‡]
California Institute of Technology

Duarte Magano
Instituto Superior Técnico, Universidade de Lisboa, Portugal and
Instituto de Telecomunicações, Physics of Information and Quantum Technologies Group, Lisbon, Portugal

Yasser Omar
Instituto Superior Técnico, Universidade de Lisboa, Portugal
Instituto de Telecomunicações, Physics of Information and Quantum Technologies Group, Lisbon, Portugal and
Portuguese Quantum Institute, Portugal

Pedrame Bargassa
Portuguese Quantum Institute, Portugal and
Laboratorio de Instrumentação e Fisica Experimental de Particulas, Lisbon, Portugal

Alessio Gianelle and Lorenzo Sestini
INFN Sezione di Padova, Padova, Italy

Donatella Lucchesi
Università degli Studi di Padova, Padova, Italy and
INFN Sezione di Padova, Padova, Italy

Davide Zuliani
Università degli Studi di Padova, Padova, Italy
INFN Sezione di Padova, Padova, Italy and
European Organization for Nuclear Research (CERN), Geneva, Switzerland

Davide Nicotra, Jacco de Vries, Domenica Dibenedetto, and Miriam Lucio Martinez
Universiteit Maastricht, Maastricht, Netherlands

Eduardo Rodrigues
University of Liverpool, Liverpool, United Kingdom

Carlos Vazquez Sierra and Sofia Vallecorsa
European Organization for Nuclear Research (CERN), Geneva, Switzerland

Anthony Francis
Institute of Physics, National Yang Ming Chiao Tung University, Hsinchu 30010, Taiwan and
Theoretical Physics Department, CERN, CH-1211 Geneva 23, Switzerland

Jesse Thaler
Center for Theoretical Physics, Massachussetts Institute of Technology, Cambridge, MA

Carlos Bravo-Prieto
Quantum Research Centre, Technology Innovation Institute, Abu Dhabi, UAE and
Department de Fisica Quantica i Astrofisica and Institut de Ciencies
del Cosmos (ICCUB), Universitat de Barcelona, Barcelona, Spain

Su Yeon Chang
European Organization for Nuclear Research (CERN), Geneva, Switzerland and
Institute of Physics, Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland

Jeffrey Lazar

Department of Physics & Laboratory for Particle Physics and Cosmology, Harvard University, Cambridge, MA and Department of Physic, University of Wisconsin-Madison, Madison, WI

Carlos A. Argüelles
Department of Physics & Laboratory for Particle Physics and Cosmology, Harvard University, Cambridge, MA

[illegible]

[illegible]

Submitted to the Proceedings of the US Community Study
on the Future of Particle Physics (Snowmass 2021)

CONTENTS

[*] This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

[†] delgadoa@ornl.gov
[‡] jvlimant@caltech.edu

I. INTRODUCTION

Particle physics has the ambitious goal of uncovering the most fundamental constituents of the Universe and deciphering the rules that mediate their interactions. Vast and complex accelerators are being developed to elucidate the dynamical basis of these fundamental constituents of matter. At these large-scale facilities, high-performance data storage and processing systems are needed to store, access, retrieve, distribute, and process experimental data. Experiments like the ones at the Large Hadron Collider (LHC) are incredibly complex, involving thousands of detector elements that produce raw experimental data at rates over a Tb/sec, resulting in the annual production of datasets in the scale of hundreds of Terabytes to Petabytes. Beyond collider physics, a significant amount of data is also expected from upcoming neutrino experiments, *e.g.* DUNE or IceCube-Gen2, and cosmological surveys, *e.g.* DESI. In addition, manipulating these complex datasets into summaries suitable for the extraction of physics parameters and model comparison is a time-consuming and challenging task.

The high-energy physics (HEP) community has a long history of working with large datasets and applying advanced statistical techniques to analyze experimental data in the energy, intensity, and cosmic frontiers. With the ever-increasing volume of data generated by HEP experiments, the community needs a significant breakthrough in the information processing systems to continue its successful journey into understanding the fundamental components of our universe. Tools developed in quantum information science (QIS) could provide a viable solution. Recently, alternative methods for detector simulation and data analysis tasks have been explored, like machine learning applications and QIS. QIS is a rapidly developing field focused on understanding information analysis, processing, and transmission using quantum mechanical principles and computational techniques. QIS can address the conventional computing gap associated with HEP-related problems, specifically those computational tasks that challenge CPUs and GPUs, such as efficient and accurate classification and simulation schemes. In addition, quantum computing offers unique advantages over classical computing in machine learning and optimization. Nonetheless, adapting these new technologies to the analysis of HEP data requires developing domain-specific tools and algorithms, such as quantum machine learning (QML) algorithms tailored to HEP applications. In References [1, 2], a review focusing on QML applications to HEP data analysis is presented. Nonetheless, the scope of this manuscript is broader, to include optimization algorithms.

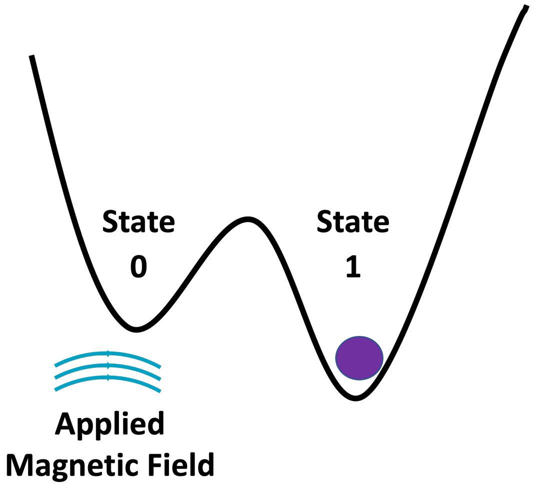
In the following, we discuss the status and prospects for quantum computing for data analysis in HEP. The emphasis is on prospects, starting from a thorough review of the current results and the related literature, thus illustrating the current status in each of the presented categories. The aim is to underline the challenges faced and to highlight possible directions for future studies. The manuscript is divided into sections covering a particular application and diving into the algorithms and models used. We start with a short introduction to the most common physical realization of quantum computing in Section II. The subsequent sections deal with applications of quantum computing to typical HEP tasks such as object reconstruction (Section III), classification (Section IV), and data generation or augmentation (Section V). In Section VI, we present a discussion of quantum-inspired algorithms. In Section VII, we provide a comprehensive list including the challenges encountered in the application of quantum computing to classical data from HEP experiments. Finally, in Section VIII, we present our views on the future of advancing HEP with quantum computing.

II. QUANTUM COMPUTING

Quantum computing involves using the quantum mechanical properties of matter, such as entanglement and superposition, to process information. Furthermore, it is based on the *qubit*, or quantum bit, instead of the classical two-state binary bits. Unlike bits, qubits can be in a superposition of both states, “1” and “0”, leading to an exponential increase in the amount of information encoded into these fundamental information processing units. For this reason, the information processing capacity of quantum computing is significantly superior to traditional computational technologies, allowing (in theory) for the reduction in execution time for specific computing tasks. Current-day quantum processors are prone to noise and small in scale, limiting the number of applications these devices can be used. In what follows, a short description of the leading quantum technologies available to domain scientists is presented.

A. Quantum Circuit Model

In the early 1980s, Richard Feynmann introduced the concept of quantum computing as a universal quantum simulator [3]. The materialization of this idea did not happen until 1985 when the concept of a *quantum Turing machine* was coined by David Deutsch [4]. This formal definition of quantum computing is based on what we know today as a *universal quantum computer*, based on qubits and unitary transformations or quantum gates. In this model, we begin in an initial “zero” state, apply a sequence of quantum gates chosen from a set of allowed gates, each acting on one or two qubits at a time, and finally output the outcome of a measurement in the computational basis.



[illegible]

B. Quantum Annealing

Another paradigm of quantum computing is quantum annealing (QA) [5, 6]. QA is a technique to solve combinatorial optimization problems by encoding the solution as the ground state of some Hamiltonian. The ground state solution is reached by initializing the quantum system in the ground state of another known, and easy to control, Hamiltonian. We then let the system evolve by slowly changing the Hamiltonian to the target one. The quantum adiabatic theorem [7] guarantees that there is a minimum evolution time for which the final state is as close to the solution state as we desire; this time depends on the smallest energy gap attained between the ground state and the first excited state of the slowly changing Hamiltonian.

C. Continuous-Variable Quantum Computing

Most of the quantum computing applications to data analysis in HEP reviewed in this manuscript focus on implementations in quantum annealers or gate-based models. While these models are very successful, another scheme explored much less than the other two QC paradigms is the continuous-variable (CV) model [8]. CV differs from its use of *qumodes* over qubits. The qumodes constitute an infinite-dimensional object instead of discrete qubits and constitute a more natural choice to simulate bosonic and continuous systems. Programming photonic quantum devices proceed in a very similar way to qubit-based devices, with both allowing the construction of circuits from quantum gates.

III. OBJECT RECONSTRUCTION

As mentioned in Section I, experimental HEP deals with the analysis of large amounts of experimental data produced at HEP experiments. One of the main tasks in any HEP data analysis workflow is the construction of *physics objects* amenable to analysis from the signals produced in a particle detector - *i.e.*, how to translate the raw detector data into an object whose properties reflect the kinematics of the initial interaction (a $p-p$ or a heavy-ion collision, or a neutrino event).

A. Tracking

Hadron colliders, such as the LHC, accelerate counter-rotating beams of hadrons in tightly packed bunches. These hadron bunches cross at designated interaction regions surrounded by tracking detectors. At each bunch crossing, several hadron-hadron collisions take place along the direction of the beam axis. Every hadron collision produces a myriad of secondary particles scattered in all directions. As the produced charged particles cross the detector's multiple layers of sensors, they leave signals of their passage, known as *hits*. The collection of the hits left by a particle is called that particle's *track*. The *tracking* problem is to recover the tracks from a set of hits. It constitutes a key task in the analysis of accelerator experiments, essential to studying the underlying physics.

At the LHC, the proton bunches cross with an interval of twenty-five nanoseconds [9]. In the conditions of the Run2 of the LHC, there are around thirty-five proton-proton collisions at each bunch crossing, and each collision may produce a few thousand hits. As a result, vast amounts of data are generated, making its analysis one of the most computationally demanding activities in experimental HEP [10]. Moreover, this demand is expected to grow dramatically after 2026 with the upcoming high-luminosity phase of the LHC [11] (HC-LHC), when we will typically have around two hundred proton-proton collisions per bunch crossing, and even more so in future accelerator machines, such as the Future Circular Collider [12].

Therefore, it is crucial to develop efficient tracking methods, possibly requiring completely novel technological paradigms. In this section, we review the existing proposals for track reconstructions based on quantum computing.

1. *Tracking with Amplitude Amplification*

The Combinatorial Kalman Filter method is at the base of several reconstruction programs in high-energy physics [9, 13, 14]. Ref. [15] analyzed this technique from a computational complexity perspective, identifying four fundamental routines: seeding, track building, cleaning, and selection. The seeding stage forms initial rudimentary track candidates, known as seeds, with just a few hits. Then, the track building stage extrapolates the seeds' trajectories along the expected path of the particles, forming track candidates by adding compatible hits from successive detector layers. To avoid having multiple tracks describing the same particle, a cleaning process is applied to remove the ones that are too similar. Finally, we select only the tracks that respect some quality criteria based on the quality of the fit between the trajectory and the corresponding hits.

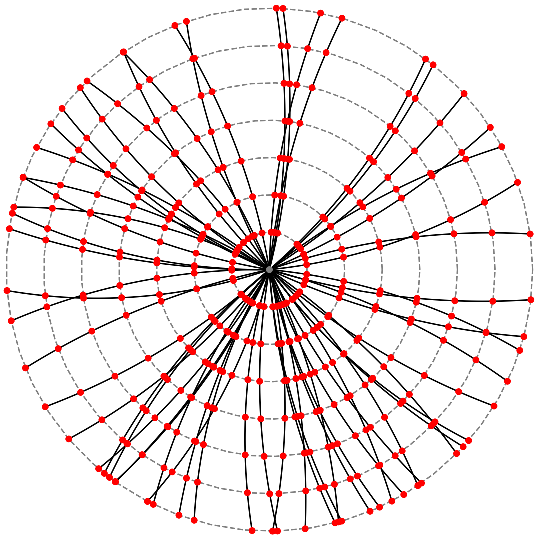
Ref. [15] proposes a quantum implementation of the Combinatorial Kalman Filter, relying on a fundamental quantum routine known as amplitude amplification [16] (a generalization of Grover's algorithm [17]), which allows for polynomial speedups for certain unstructured search problems. The general idea is that we can use this technique to find the seeds or hits that satisfy specific properties faster than what would be possible with classical brute-force search. For both the seeding and track building stages, it is possible to reproduce the same output as the Combinatorial Track Finder (up to bounded error probability) with lower quantum computational complexity. Ref. [15] further shows that if we do not register the outputs of the individual stages, but are only interested in the final reconstructed tracks, then an even stronger quantum advantage can be reached.

Ref. [15] provides a rigorous proof of quantum speedup for HEP data analysis. However, its results refer to the asymptotic regime of infinitely-many hits, which may be of limited interest for real instances of track reconstruction. Moreover, this proposal depends on the availability of a quantum random access machine (QRAM) [18] that allows coherent access to the hits' data. Although there have been proposals of physical architectures for QRAM [19], there are still significant challenges to overcome before such a device can be realized in practice. Finally, the reached speedups have been found to be mild; quantitatively, $O(n^4)$ classically versus $O(n^3)$ quantumly under certain assumptions, where n denotes the number of recorded hits.

As the algorithms proposed in [15] require fault-tolerant computing, it has not been possible to test them in real quantum hardware. Therefore, it remains unclear at which scales the complexity advantage may actually yield an advantage in processing time. In general, fault-tolerance and QRAMs lie beyond the NISQ era, so it is hard to estimate when this type of strategy might have a practical impact in HEP.

2. *Tracking with Quantum Annealing*

The work of Ref. [15] consists of a direct "quantization" of a classical algorithm (the Combinatorial Kalman Filter), which might not be the best path to establishing a significant advantage in quantum computing for HEP problems. Instead, a more successful approach could be obtained by breaking the direct correspondence with the classical setting and designing completely new tracking algorithms that inherently take advantage of the features of quantum processors. Some of the first attempts in this direction [20–23] were conceived within the quantum annealing model.



[illegible]

In particular, Ref. [20] and Ref. [21] have formulated track reconstruction as a quadratic unconstrained binary optimization (QUBO) problem, which can be naturally mapped to a quantum annealer. In Ref. [20], the QUBO variables correspond to all possible edges between hits. The idea is that, in the optimal solution, the variables assigned with +1 connect hits that are left by the same particle. For this purpose, Ref. [20] adapts the energy function of the Denby-Peterson network method [24, 25] to HL-LHC configuration. In contrast, in Ref. [21] the binary variables correspond to hit triplets and introduce a compatibility measure between hit triplets to define the QUBO problem. The use of hit triplets instead of doublets comes with a greater classical pre-processing cost while possibly increasing the performance of the reconstruction.

With both approaches, there are polynomially many coupling coefficients that need to be classically pre-processed to form the QUBO Hamiltonian – $O(n^3)$ for Ref. [20] and $O(n^5)$ for Ref. [21], where n is the number of hits. Moreover, the annealing time is likely to scale exponentially with respect to the number of hits. Hence, it remains unclear whether these methods could exhibit quantum speedup over the classical counterpart (*i.e.*, simulated annealing). Nevertheless, the simulations done at the D-Wave's machines [26] show competitive efficiency and purity for some toy datasets.

Ref. [22] addresses the problem of track clustering, not track reconstruction *per se*. This is the first step in reconstructing the positions of hadronic interactions, also known as primary vertices. The algorithm takes as input a set of positions z_i, where i is the track index, along the beam axis corresponding to the points where the reconstructed tracks approach it most closely. The goal is to associate each track with an element of a list of possible primary vertices. In the algorithm of Ref. [22], the annealing variables are the entries of a binary association matrix between the tracks and the candidate vertices, and the annealing Hamiltonian penalizes assignments of tracks i and j to the same vertex if $|z_i - z_j|$ is large. In total, one needs to pre-compute $O(n_v n_t(n_v + n_t))$ coupling coefficients, where n_v is the number of candidate vertices and n_t is the number of tracks. As in Ref. [20, 21], there is no guarantee on scaling the required annealing schedule. Ref. [22] tests their algorithm with small datasets on a D-Wave's quantum computer, reporting clustering precisions close to the results obtained with simulated annealing.

In contrast, Ref. [23] studies the problem of track classification, that is, the problem of distinguishing between signal and background events from the hits' data. This seeks to accelerate track reconstruction by isolating the signals of interest. For this purpose, Ref. [23] utilizes both associative memory models and content addressable memory models in conjunction with quantum annealing. In this setting, a collision event is characterized by a detection pattern – the detector is divided into discrete positional segments and we record the number of hits in segment. Given a history of detection patterns, the QUBO coefficients are assigned based on a specified learning rule. The computational complexity of training the model scales as $O(N^3p^2)$, where N is the length of the patterns and p is the number of training patterns. Again, it is difficult to estimate how the annealing time scales with the model parameters. The performance of the memory recall is dependent on N and p, as well as on the noise levels and the efficiency of the detections. Running simulations on D-Wave's quantum hardware, Ref. [23] report that, for small problem instances, these models can reach accurate classification results.

In summary, none of these proposals based on quantum annealing manage to conclusively establish a quantum advantage for track reconstruction over the classical counterparts. However, the simulations on currently available quantum hardware, particularly on D-Wave's annealing machines, already reveal competitive results, as long as one is restricted to small problem sizes. A complete characterization of the performance and scaling of these annealing-based solutions will require larger programmable quantum annealers.

3. *Tracking with Neural Networks*

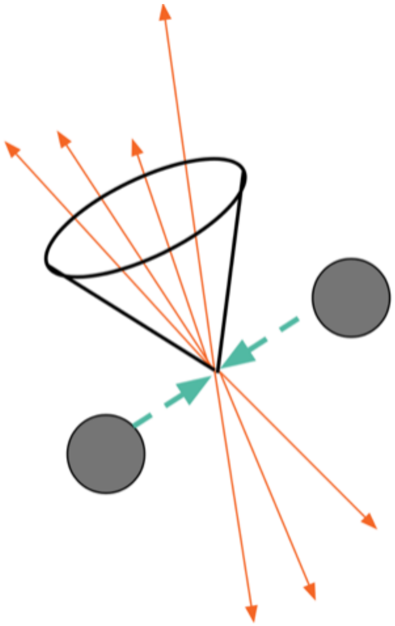
Ref. [27] seeks to leverage the sparse nature of the tracking data with hybrid quantum-classical neural networks. As a pre-processing step, they generate a graph from the detector's data, where the particle's hits become the nodes, and the possible track segments between hits become the links. Then, a neural network model takes this graph as input and gives as output the probability of each edge linking two consecutive hits left by the same particle. This neural network combines both classical and quantum layers, the latter realized in the form of parametrized quantum circuits.

Evidently, the reconstruction performance depends on the number of hidden classical n_C and quantum n_Q dimensions, and on the number of training iterations n_I. For small values of n_C, n_Q, and n_I, Ref. [27] shows that the hybrid networks closely match the results of the classical model, at least for the simplified datasets that were considered. Given the current limitations in quantum hardware, it is difficult to assess whether increasing the size of the network (n_C, n_Q, and n_I) would significantly improve the results. It also remains unclear whether neural network algorithms will ever achieve the precision of the Combinatorial Kalman Filter methods.

Finally, the complexity of the algorithm depends on the network parameters, and as such these, become hyperparameters in the optimization scheme. A further discussion on this topic can be found in Section IV B.

B. Jets

In collider particle physics, a jet is a collection of particles collimated into a roughly cone-shaped region. Jets arise from the fragmentation of quarks and gluons produced in high-energy collisions. During the collision, the QCD confinement the quarks and gluons are subjected to, is broken, yielding a spray of color-neutral particles that can be experimentally measured in particle detectors. Jet clustering algorithms are employed to estimate the kinematics of the particle that initiated the jet. These clustering schemes combine the observed particles into a collective jet object for further study. In electron-positron collisions, the dominant event topology involves two back-to-back jets resulting from the fragmentation of a quark and an anti-quark. This motivates the partitioning of the event particles into two hemisphere jets, which can be accomplished using event shapes like *thrust* [28, 29]. The calculation of thrust can be very expensive computationally, scaling like $O(N^3)$ [30] for an event with N particles, though using a method introduced in Ref. [31], it is possible to improve this to $O(N^2 \log N)$. On a universal quantum computer, thrust can be computed in $O(N^2)$ [32] using a strategy based on Grover search. Alternatively, Ref. [32] showed how thrust can be phrased as a QUBO problem suitable for quantum annealing.



[illegible]

In Ref. [33], the thrust-based quantum annealing for jet clustering is benchmarked on the D-Wave Advantage 1.1 QPU. Algorithmic improvements like reverse annealing showed promise on this problem, but the most significant gains came from tuning the annealing parameters such as annealing time, the number of anneals, and relative chain strength. The QPU performance was also compared to hybrid and classical solving strategies, with simulated annealing always performing the best. Nonetheless, for event sizes smaller than 22 particles, the classical and quantum annealing solvers display a similar performance. These results hint at the limitations of current quantum annealing devices in terms of connectivity. Problems involving many spin variables and all-to-all connectivity, like the thrust problem, perform poorly on this device.

Based on the quantum annealing-based algorithm proposed in [32], an extension to the QUBO formulation for thrust calculation is presented in Ref. [34]. This extension is based on the angular distance between two particles in a given event, penalizing the assignment of two particles located on the same hemisphere of the true partition. This approach displays a better performance when tested on the D-Wave's QPU and when compared to the original formulation in Ref. [32]. It should be noted that these benchmark studies were limited to a low number of annealing runs due to limited access to the QPU.

On the other hand, algorithms based on digital quantum computing have also been proposed in Refs. [32, 35], but are not suitable for implementation on NISQ devices due to the need for a QRAM architecture to access the information about the particles on the event for further processing.

As a summary for this section, track reconstruction and jet clustering tasks in HEP deal with the clustering of detector information into higher-level objects that contain information about the interaction that triggered the detector response. The main challenge is the number of elements to be clustered. Quantum computing applications to the problems of track reconstruction and jet clustering are limited by the size and connectivity of currently available devices. Notably, these tasks require either a quantum annealer with all-to-all connectivity or a QRAM-like protocol for accessing the information of individual elements in the clustering sample in coherent superposition.

IV. CLASSIFICATION

Once detector data has been processed and higher-level objects have been constructed, the next step in a HEP analysis is using statistical techniques to extract the signal of interest and suppress background as much as possible. Thus, efficient classification tools are crucial in HEP data analyses. Classification algorithms are not limited to the discrimination events of interest (signal) from other processes (background). In addition, reconstructed particles also undergo a process of classification to be assigned a label or identification according to their type and kinematic properties. Jets need to be classified according to their origin.

Classification tasks are usually addressed with Multivariate Analysis Techniques (MVA). Current physics analyses employ analytic methods like the Matrix-Element Method [36], as well as Boosted Decision Trees (BDTs) and Neural Networks (NN). In this section, new approaches to classification tasks based on different quantum computing architectures and algorithms are reviewed, which may yield better performance than their classical counterparts.

A. Quantum annealing applications

One of the main and first quantum computing models used for classification in HEP is quantum annealing. In this binary approach, each discriminating variable x_i is transformed into a weak classifier c_i, and has an associated spin s_i. A strong classifier is then built, which is the linear combination of all weak classifiers and spins. The classification is then expressed in terms of the optimal set of spins minimizing the energy of a fully connected Ising Hamiltonian. This set of spins is found after an iterative annealing process where qubits are strongly coupled in a chain to represent a spin s_i. Correlations among the discriminating variables are possible via the $c_i \cdot c_j$ coupling terms in the Hamiltonian, and are rendered further present via operations among them, which are then transformed in weak classifiers. Fixing variable scheme and a cutoff on the coupling terms are used to reduce the size of the Ising model to be encoded on the annealer. This approach is followed in Ref. [37] where the signal from Higgs boson decays is separated from the background. In this work, quantum annealing achieves a better result for a small number of training events but is outperformed by classical machine learning (ML) tools for a large number of events. The basic approach of quantum annealing for classification [37] is further enhanced in Ref. [38], where the procedures of *zooming* and *augmentation* are applied. While the zooming shifts and narrows the region of search in the space of spins, the augmentation multiplies the number of binary weak classifiers c_i based on the shape of each variable x_i, effectively providing better discrimination. These two steps significantly improve the result of the Higgs classification problem; however, they are still outperformed by classical ML approaches. The principle of zoomed and augmented annealing is applied to the new classification problem of supersymmetric top quark versus SM events [39]. Here, the choice of the discriminating variables is based on a metric incorporating the full statistical and systematic uncertainties of a counting experiment in HEP. This results in a relatively reduced (only 17) but well-performing set of discriminating variables. Different such sets of variables are tested with different augmentation schemes. Finally, in order to place the quantum-based classifier on a footing as equal as possible to the classical Boosted Decision Trees (BDT), the discriminating variables are decorrelated by being passed through a principal component analysis before being fed to the quantum annealer. The study of Ref. [39] achieves a classification that is at least as good as the best-known classical ML tool, here the BDT. It has to be noted that this result is attained for a rather large number of events in the training sample ($5 \cdot 10^4$), which is the typical size of samples used in HEP, and for which the first quantum based classifications are outperformed by classical approaches. The results of Ref. [37–39] are all obtained with the Chimera graph of D-Wave. If a chain is broken within this device, the measure of the qubit chain is performed through a majority vote, which can lead to the selection of non-optimal sets of spins, and therefore to a possible loss of discriminating information. This is a limitation of these first-generation quantum annealers. A larger number of couplers in future machines will render each chain more stable and less prone to be broken, which will, in turn, allow more effective use of the discriminating information.

Quantum annealing can also be used to identify the topology of a signal event, where there is no hypothesis about the latter and where the mass of the new particles is inferred from their decay products. Ref. [40] looks at cases where two new objects would be produced at an LHC collision, with each decaying in a number of known particles. The problem to solve is combinatorial, where the correct association of two groups of observed particles has to be made to reconstruct the invariant mass of the new particles. The spin s_i indicates whether a particle i is the decay product of one or the other new particle. The kinematic constraints of the new particles are formulated in the Ising Hamiltonian, which is encoded on a Pegasus graph of D-Wave, and whose energy is minimized. The energy landscape of such a problem is complicated, and classical annealing will have a local minimum problem. The combinatorial problem solved by quantum annealing reaches an accuracy higher than a classical algorithm for three different processes, thus indicating a quantum advantage.

The study in Ref.[41] also uses an Ising Hamiltonian-based approach to perform a model-independent search for physics beyond the SM. The kinematic space is partitioned in bins i, where the difference between the simulated SM prediction and observed data is measured, and where each spin s_i can either be aligned (*i.e.* of the same sign) or anti-aligned with the measured difference. The Hamiltonian is expressed such that the ground state energy of the system is a measure of the goodness-of-fit. Linear spin terms in the Hamiltonian are only sensitive to the difference mentioned above in each bin i. On the other hand, $s_i \cdot s_j$ terms capture the interaction between neighboring bins, therefore being sensitive to spatial correlations between different kinematic regions, which in turn helps to differentiate between random noise and new physics signals. This work relies on simulated annealing as a method for minimizing the energy of the Hamiltonian. Toy experiments are generated where signal-plus-background and background hypotheses are tested. The goodness-of-fit of different approaches is tested in terms of true- versus false-positive rates. Both with one- and two-dimensional toy experiments, the goodness-of-fit test statistic based on this quantum algorithm performs better than classical methods. It has to be noted that this improved capacity to detect an anomaly versus an expectation is free on any assumption on the signal, and is thus model-independent.

The work presented in Ref. [42] explores a classification application of importance in cosmology, a galaxy morphology classification by training Restricted Boltzmann Machines (RBMs) in a quantum annealing device. RBMs are, generally speaking, generative models and will be discussed in this context in Section V A. In the classical setting, an RBM is a stochastic neural network that can learn a probability distribution over its set of inputs. The study in Ref. [42] found that for small datasets and limited numbers in training repetitions, quantum annealing-based RBMs performed very well and outperformed the alternative classical algorithms studied, namely logistic regression and gradient boosted trees. However, outside of these rather special training scenarios, RBMs (regardless of the classical or quantum nature of the training algorithm) did not outperform the gradient boosted tree algorithm.

B. Variational quantum circuits

Variational Quantum Circuits (VQCs) are hybrid quantum-classical algorithms aiming to harness the strength and scalability of both computational paradigms. In this architecture, classical computers are used for optimization and quantum computers for specific complicated tasks, such as calculating expectation values.

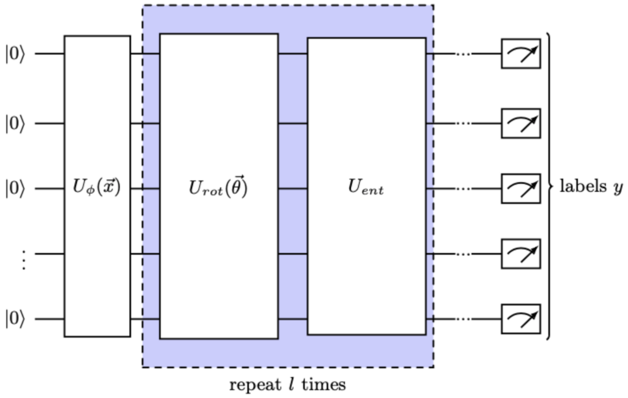
A VQC can be viewed as a Quantum Neural Network (QNN) where the encoded quantum state goes through a circuit with different layers of gates that depend on parameters that will minimize a loss function through training. In a VQC, once an initial state has been prepared through the encoding of classical data into a quantum state, a series of unitary transformations are applied through a circuit with layers that depend on trainable parameters $\vec{\theta}$ and act serially, mimicking the forward pass of a neural network, as first noted in [43]. As a classifier, a VQC can be trained from labeled data to classify new samples or, in a generative setting, to model correlations in the input data. Recent studies have reported the application of variational architectures in the field of classification [44], function approximation [45], generative machine learning [46, 47], metric learning [48], deep-reinforcement learning [49] and sequential learning [50].

In the gate-based quantum computing model, variational algorithms are implemented using quantum circuits composed of a network of single and two-qubit operations, with rotation angles serving as variational parameters. The implementation of a VQC usually takes place in three steps:

1. First, an initial state is prepared using a feature map or unitary transformation $U_\phi(\vec{x})$ to encode the classical input data $\vec{x}$, into a quantum state. The input feature vectors become the rotational gates' arguments and remain fixed during the circuit evaluation. Some standard techniques include angle and amplitude encoding; see Ref. [51] for more complex approaches.

2. The second block of unitaries is a parameterized quantum circuit or variational form. It is given by $U(\vec{\theta})$ parameterized by gate angles θ and includes alternating layers of entangling and rotation gates. Each layer consists of a sub-circuit that depends on trainable parameters and an entanglement sub-circuit. The learnable parameters $\vec{\theta}$ will be optimized through gradient-based methods. For example, the commonly used gradient-based optimizer is Adam [52].

3. Then, a quantum measurement is performed on a subset (or all) of qubits to retrieve the information. If we run the circuit once and perform a single quantum measurement, it will yield a binary string, and it generally differs from what we will get if we prepare the circuit again and perform another quantum measurement due to the stochastic nature of quantum systems. However, if we prepare the same circuit and perform the quantum measurement several times, we will get the expectation values on each qubit. Furthermore, different bases can be used for the measurement. In most cases, the expectation value of the σ_Z Pauli operator is used to obtain measurements on the computational basis.



[illegible] $U_\phi(\vec{x})$, (2 [illegible] $U_{rot}(\langle 0|\theta|0\rangle))$ [illegible] $U_{ent})$ [illegible] l [illegible] (3 [illegible]

In most cases, as we will see in some of the HEP applications discussed below, the VQC performance is evaluated by looking at the Receiver Operating Characteristics (ROC) curve and the Area Under the Curve (AUC). In what follows, applications of VQCs to the classification of HEP experimental datasets will be discussed.

The work in Ref. [53] studies the classification of chargino-pair production via a Higgs boson versus the SM background. In this study, two circuit designs or *Ansatz* are tested to demonstrate the feasibility of QML for the event classification task in HEP data analysis. The first choice is the circuit proposed in Ref. [45] and constructed using a time-evolution gate, denoted as e^{-iHT}, with the Hamiltonian H of an Ising model with random coefficients and the series of R_X, R_Z and R_X gates, whose rotational parameters are updated during training. This layout is repeated three times after applying a unitary that encodes the classical input data as rotational parameters in the R_Y and R_Z gates. The second choice of VQC is based on the circuit proposed in Ref. [54], which uses a layer of Hadamard and R_Z gates to prepare the input state, followed by an entangling layer sandwiched between two layers of rotational R_Y and R_Z gates with trainable parameters. Measurement is performed on the first two and all the qubits using the Pauli Z operators for the first and second circuit designs, respectively.

The VQC is trained by minimizing the cross-entropy cost function as implemented in the scikit-learn package [55] and using the COBYLA optimization algorithm [56].

The performance of the proposed VQCs is benchmarked against two classical ML algorithms: a BDT, as implemented on the Keras package [57], and a Deep Neural Network model based on a fully-connected feed-forward network composed of 2-6 hidden layers with 16-256 nodes each, implemented using the Tensorflow package [58]. The hyperparameters in the classical models are chosen to match the capabilities of the quantum models, and both methods are applied to the same task and dataset size. It has to be noted that this classification is based on a small number of discriminating variables. The total number of internal parameters is 12, 20, and 28 for the 3-, 5-, and 7-variable classification, respectively. Concerning benchmarking on hardware, only the circuit design based on single and two-qubit gates is tested on a quantum backend. The VQC is run on the 20-qubit IBM Q Network quantum computer. It is noteworthy that this VQC, in the present stage, does not perform better with a larger number of features.

In Ref. [59], a VQC approach is applied for the classification of $t\bar{t}H$ where a pair of top quarks are produced with a Higgs boson. In this study, two Auto-Encoder NNs (see Section V A) are used to reduce the dimension of the feature space, and thus, ease the classical input feature encoding into quantum states. This compressed representation is the input for the hybrid architectures introduced and the benchmark against non-quantum techniques.

A *data re-uploading* technique, introduced in Ref. [60], is used in an effort to reduce the number of quantum registers needed to encode the classical data into a quantum state. In this approach, the quantum classification circuit comprises several repetitions of the traditional VQC scheme, each with its own classical inputs in the feature map and trainable parameters. This study is implemented in circuits comprising different number of qubits. Using the data re-uploading technique, the feature map based on one and two-qubit gates is applied to load the classical variables. Finally, the qubits are measured and used to classify the input.

The study in Ref. [61] is a VQC approach to classify $t\bar{t}$ events versus the rest of the SM processes, using only two kinematic features. An angle encoding scheme is used to prepare the two-qubit initial state. The trainable block of the VQC consists of two layers of single rotation gates, followed by two CNOT gates, entangling the qubits in the circuit. Finally, a Pauli Z operator is applied on the first qubit, where the expectation value is taken. The circuit is run repeatedly to obtain an estimate of the latter, where the size of the training sample is relatively small (1500 events). Two different instantiations of the VQC are tested, where a classical and a quantum gradient descent are run and compared to the performance of NN with a gradient descent. The quantum gradient optimization shows a faster convergence than the traditional gradient descent optimization and the classical neural network. In terms of classification, the VQC is observed to perform better than the NN in high purity regimes, *i.e.* when cutting on high values of the classifier's output. The VQC with a quantum gradient descent performs slightly better than the one with the classical descent, always in the high regimes.

In Ref. [62], a VQC is applied to the analysis of Higgs boson production in association with a top-quark pair and a Higgs boson decay to two muons. The goal is to train a supervised learning model able to discriminate between two processes, the signal events, $H \to \mu^+\mu^-$ and the background events; namely, a $Z/\gamma^* \to \mu^+\mu^-$ interaction. A VQC-type model is proposed and trained based on thirteen kinematic variables associated with the process of interest. A pre-processing step to reduce the feature space dimensionality is employed using the Principal Component Analysis (PCA) method. A ten-qubit quantum circuit is set up using the variational circuit design in Ref. [54]. Only half of the qubits are measured to reduce the potential errors associated with measurement. The circuit is trained through the noisy simulation of a ten-qubit system using the IBM's *qasm_simulator*. An error mitigation scheme is applied to correct measurement errors through a relation matrix between the ideal and noisy results, and applied to the noisy results. The loss function is defined by the error probability of incorrect assignment compared to the exact solutions available for the training dataset. During the training, the loss function is minimized to penalize missassignment and to optimize variational parameters using the SPSA optimizer. The VQC performance is benchmarked against classical ML models, including a Support Vector Machine (SVM) constructed using the *scikit-learn* package and a BDT set up using the *XGBoost package* [63]. A hyperparameter tuning was performed on the classical models. The authors report an agreement between the results obtained when training on hardware and noisy simulation setups when error mitigation techniques are applied. Furthermore, a similar performance is observed on models trained on simulated noisy settings, combined with error mitigation techniques and classical ML models.

In Ref. [64], a VQC approach has been used to identify the flavour of jets produced in proton-proton collisions at the LHCb experiment. The identification of jets plays an important role in physics at hadron colliders, and the ability to distinguish the flavour of the quark generating a jet is fundamental in order to search for New Physics processes and find possible deviations from the Standard Model. In this paper, jets initiated by b or $\bar{b}$ quarks are identified using a VQC, and its performance is compared with an exclusive algorithm used so far at LHCb, the so-called *muon tagging* algorithm, which infers the jet flavour by means of the charge of the muon found inside the jet, and a standard Deep Neural Network (DNN). LHCb simulated data are analysed, and jet substructure information is used as input features for the classifiers; for each jet, five types of particles (muon, kaon, electron, pion, and proton) with the greatest p_T inside the jet are chosen. For each type of particle, three variables are considered: the transverse momentum with respect to the jet axis ($p_{\mathrm{T}}^{\mathrm{rel}}$), the distance in the pseudorapidity-azimuthal plane to the jet axis (ΔR) and the charge of the particle (q); a global variable, the weighted charge of the jet (Q), is also considered, accounting for a total of 16 variables. Two circuit geometries have been studied: the Angle Embedding geometry, where n input features are mapped to angles θ of rotational gates applied to n different qubits, thus resulting in a n-qubit circuit, and the Amplitude Embedding geometry, where input features are embedded in the amplitudes of a state vector of a $\lceil \log_2 n \rceil$-qubit state. The complexity of the circuit depends on the number of *strongly entangling* layers, consisting of general rotational gates and CNOT *entangling* gates, chosen in the variational part of the circuit. In order to understand the impact of the considered variables in the classification, two versions of the dataset have been used: the *muon dataset*, where only variables coming from the muon inside the jet plus the jet charge Q are used, and the *complete dataset*, which uses all variables. In this way, for the *muon dataset* the Angle Embedding (Amplitude Embedding) geometry is a circuit of 4 (2) qubits, while for the *complete dataset* the Angle Embedding (Amplitude Embedding) geometry uses 16 (4) qubits. Results show that the Angle Embedding geometry works better than the Amplitude Embedding geometry, reaching the same performance of the DNN for the *muon dataset* while for the *complete dataset* the Angle Embedding classifier has slightly worse performance than the DNN. The study also shows that by increasing the number of strongly entangling layers, the classification accuracy increases up to a certain value where there is no further improvement, giving interesting suggestions on the optimal number of strongly entangling layers. Analysis of the number of training events shows that the quantum classifiers perform better than the DNN when using fewer training events while reaching similar performance when increasing the number of training events. Finally, the impact of noise on the circuit geometries has been considered using simulations that account for circuit noise contribution: no evident degradation in performance is found, suggesting that these circuits are rather robust and, in principle, can run on hardware.

1. *Support vector machine*

Quantum Support Vector Machines (QSVMs) [65] share many similarities with VQCs. Both systems encode the classical input states in a Hilbert space by applying unitary designs tailored to the application. While this encoding is conceptually equivalent in both approaches, the two models differ in how the quantum state is handled once it is prepared. VQCs can be formulated as quantum kernel methods [66].

The SVM algorithm maps $\vec{x}$ into a higher dimensional feature space, where it measures the similarity between any two data instances, denoted as "kernel entries," $k(\vec{x_i}, \vec{x_j})$. The SVM algorithm then optimizes a hyperplane that separates data points into two categories. A main limitation of the classical SVM algorithm is that evaluating kernel entries in a large feature space can be computationally expensive. Thus, the QSVM is expected to leverage the quantum state space as a direct representation of the feature space, giving rise to kernel functions that are hard to evaluate classically. From this foundation, quantum feature maps can be designed and tested on practical datasets, potentially leading to better classification results than classical feature maps and kernels. Once a suitable quantum feature map is chosen, the kernel matrix element is constructed by sampling the probability of measuring $|0\rangle$:

$$k(\vec{x_i}, \vec{x_j}) = |\langle 0^{\otimes n}| U^\dagger(\vec{x_i}) U(\vec{x_j}) |0^{\otimes n}\rangle|^2. \tag{1}$$

Another main difference with VQCs is that the loss function of a QSVM depends on the inner product of the feature vectors, *i.e.*, the goal is to maximize

$$L(c_1 ... c_n) = \sum_{i=1}^{n} c_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i c_i (\vec{x_i} \cdot \vec{x_j}) y_j c_j, \tag{2}$$

subject to $\sum_{i=1}^{n} c_i y_i = 0$, and $0 \leq c_i \leq \frac{1}{2n\lambda} \equiv C$ for all i. Here, c_i are the independent variables of the loss, $\vec{x_i}$ and $\vec{x_j}$ are feature vectors of a given pair of data points, i and j and y_i, y_j their corresponding labels. Finally, n is the number of encoded features, and λ is a regularization parameter that tunes the trade-off between misclassification and width of the SVM margin.

Furthermore, in QSVM-like circuit designs, all quantum registers need to be measured to construct the quantum kernel, whereas in VQCs this is not a requirement.

The Higgs classification problem of Ref. [59] also studied a classification scheme based on the implementation of a QSVM. Again, auto-encoders are used to reduce the dimension of the feature space, as explained in Sec. IV B. In terms of feature maps, two data embedding circuits are tested: 1. An amplitude encoding circuit with N qubits, capable of encoding 2^N features, for a 4-qubit and 6-qubit architecture; 2. An 8-qubit architecture, which is more suitable for an implementation on a NISQ device. The 4- and 8-qubit models have a similar performance to a classical SVM with a radial base function kernel. Likewise, the 6-qubit QSVM with amplitude encoding performs similar to an SVM with a linear kernel.

In Ref. [67], a QSVM model is used again for the discrimination of $t\bar{t}H$ events in the $H \to \gamma\gamma$ decay channel from non-resonant two-photon production events. Three classical kernels are considered to benchmark the classical SVM method: the linear kernel, the polynomial kernel, and the radial basis function (RBF) kernel. Both quantum and classical SVM models are trained using 15 kinematic variables resulting from the compression of the original 23 kinematic variables available in the dataset, by using a PCA method. The 20-qubit model is trained on a noise-less simulator and showed similar performance compared to the classical SVM. When deployed on hardware, the system size is reduced to 15 qubits, and the authors report similar performance compared to the noiseless simulation.

Another implementation of a QSVM algorithm for classification is described in Ref. [68]. Here, the authors designed and implemented a QSVM approach for the signal-background classification task in B meson decays. Three different encodings were tested to study the impact of the choice of feature map in the classification performance. In particular, the Bloch encoding circuit, which the authors designed for encoding particle data, outperformed the other two and used fewer resources for the same task. When using a limited number of inputs, the QSVM outperformed classical methods in simulations.

In Ref. [69], an extension to the method of machine learning based on quantum kernel methods is extended up to 17 hardware qubits requiring only nearest-neighbor connectivity. This circuit structure is used to prepare a kernel matrix for a classical SVM to learn patterns in 67-dimensional supernova data for which competitive classical classifiers fail to achieve 100% accuracy. Furthermore, the circuit design is justified based on its ability to produce large kernel magnitudes that can be sampled to high-statistical certainty with relatively short experimental runs. This experiment is, by far, the largest classification model deployed on hardware and demonstrates similar performance to non-quantum techniques for the classification of a HEP dataset.

2. *Quantum Convolutional Neural Networks*

In Section IV B, we introduced the concept of VQCs and compared the tunable parameters to the weights in a classical neural network. We now introduce the concept of quantum convolutional neural network (QCNN). QCNN is the framework that uses VQCs to perform the convolutional operations in a classical CNN. Convolutional filters or kernels are replaced with VQCs to harvest the expressive power granted by quantum entanglements. The quantum convolutional kernels will sweep through the input data and transform them into a representation vector of lower dimensions by performing measurements. A stack of VQCs will ensure features of varied length scales that are captured in different layers. In Ref. [70], a QCNN framework for the classification of HEP events from the simulated data from the DUNE experiment is proposed. Based on the success of classical CNNs for the classification of images, a QCNN is proposed for the classification of neutrino events as detected in a Liquid Argon Time Projection Chamber (LArTPC). This technology allows for detecting neutrino events through high-resolution images of particle interactions within the detector volume as the ionized electrons drift towards the multiple sensing wire planes. The goal of the QCNN is to predict the types of different particles by analogy with those performed via classical CNN. The target labels correspond to the four possible particle types to be detected, an electron, a muon, a pion, or a proton. The authors report that the quantum model can learn faster and reach better testing accuracy with fewer training epochs, when using a similar number of parameters in the classical and quantum CNN benchmarks on noiseless simulations.

In Ref. [71], a QCNN based technique aimed at understanding how QML models can provide advantages over classical models when dealing with sparse data, which is common in scientific data. This work introduces a hybrid quantum-classical graph convolutional neural network (QGCNN) framework applied to the same classification problem in Ref. [70]. This study compares the performance of a QGCNN to a classical multilayer perceptron and CNNs and reports that the 10-qubit quantum model requires fewer parameters to achieve comparable performance as the classical models. Furthermore, they compare the QGCNN model to the QCNN model proposed in Ref. [70]. Both quantum models can achieve similar performance, but the QGCNN requires half the number of parameters used in the QCNN, highlighting the importance of models tailored for sparse datasets.

3. *Anomaly detection models*

The search for new physics depends on our ability to separate Standard Model events from the background's much rarer and complex signal events. A new approach for finding events consistent with beyond-the-standard-model physics is through data-driven search, where no assumptions on the new physics scenario are made. While this allows for a search on a broader phase space, it requires a classification technique that can successfully learn and identify the important features in the background data to discriminate them from rare signal events which do not share the same properties. In classical ML, autoencoder architectures have been designed for this purpose [72]. Autoencoder architectures consist of an encoder step that compresses the feature space into a latent space with reduced dimensionality. Subsequently, the latent space is decoded into an output of the same dimensionality as the input feature space. The entire network is then trained such that the loss function, which evaluates how well the output resembles the input, is minimized. Because quantum mechanics can generate patterns with properties beyond classical physics, a quantum computer should be able to recognize patterns beyond the capabilities of classical machine learning. Thus, the motivation for a quantum autoencoder is that such a model would allow us to efficiently perform the dimension reduction of quantum data. In classical architectures, the necessary compression and expansion of data in the encoding and decoding steps are manifestly non-unitary, which has to be addressed by the QAE using entanglement operations and reference states which disallow information to flow from the encoder to the decoder. A model capable of effectively reducing the dimension of a quantum state was initially proposed in Ref. [73]. In the context of anomaly detection, the assumption is that the minimal dimension of the latent space for which the input features can still be reconstructed corresponds to the dimensionality of the information space required to describe the training sample, here, the SM background processes. If the signal is kinematically sufficiently different from the background samples, the loss function or reconstruction error will be larger for signal than for background events. In Ref. [74], an alternative approach based on continuous-variable quantum computing is proposed. The authors use gaussian boson sampling to embed classical data into a feature vector and use an autoencoder scheme for model-independent searches through anomaly detection techniques. A continuous variable alternative to the classical k-means clustering algorithm is devised and denominated Q-means clustering, potentially scaling to large feature vectors more efficiently. The Q-means algorithm has a complexity in the order of $\log(N)$, with respect to the size N of the feature vector. This represents an improvement from the classical k-means algorithm with a complexity in the order of N. The model is yet to be deployed in a continuous variable architecture, and the authors limit their implementation to discrete qubit quantum computing. In Ref. [75], the authors study quantum autoencoders based on VQCs for the problem of anomaly detection at the LHC. For a QCD $t\bar{t}$ background and a resonant heavy Higgs signal, they find that a simple quantum autoencoder outperforms dense classical autoencoders for the same input space. In Sec. V A, QVAEs are discussed in the context of generative tasks, although no application in HEP has been reported.

V. DATA GENERATION/AUGMENTATION

A crucial element of any analysis in HEP involves the simulation of the physical processes and interactions taking place at HEP facilities to develop new theories and models to explain experimental data and characterize background, study detector response, and plan for detector upgrades. The corresponding simulations start with a Lagrangian, compute the hard scattering in perturbative QCD, the parton shower in resummed QCD, model hadronization based on reliable precision measurements, and finally, a full detector simulation. Technically, all of the processes involved rely on Monte Carlo techniques. These simulations are often computationally intensive, taking up a significant fraction of the computational resources available to physicists. The production of reliable and statistically significant samples represents a huge computational footprint both in terms of CPU usage and disk space. Currently, the best estimates suggest that the simulation of a single collider event already takes several minutes [10], with $O(109)$ events to be generated (on average) for each simulation campaign in the LHC.

Generative models are trained to prepare a target distribution that can accordingly be used to generate new samples. This is more challenging than the discriminative tasks described in Sec. IV since it requires one to efficiently learn, represent, and sample from high-dimensional probability distributions, but quantum processors are well-suited for this task. Many kinds of generative models used in classical machine learning use neural networks and are parameterized by weights and biases (*e.g.* Boltzmann Machines (BMs) [76] and RBMs [77], autoencoders and variational autoencoders ((V)AEs) [78] and Generative Adversarial Networks (GANs) [79]). Training generative models can be based on minimizing the energy of the model [76, 80], minimizing the error when sampling from a target posterior and the model posterior [78], or through adversarial methods [79]. These neural network models have been translated into quantum models (q(R)BMs, q(V)AEs, qGANs) either as standalone VQCs (see Section IV B) or as a component in a hybrid network [73]. Many of the training workflows can be adapted as variational algorithms. The training leverages classical methods and hybrid workflows to optimize the parameterized quantum models.

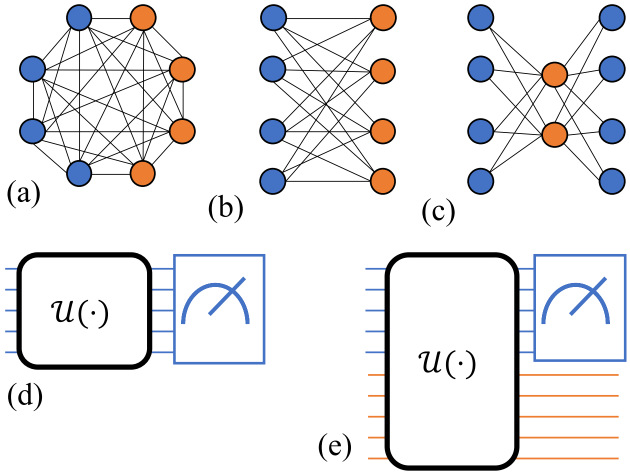
The ability of quantum information processors to represent vectors in N-dimensional spaces using $\log(N)$ qubits, and to perform manipulations of sparse and low-rank matrices in time $O(\text{poly}(\log(N)))$ [81] motivates the exploration of quantum generative models as an alternative to classical generative models for generative tasks in HEP. Quantum generative models are expected to exhibit an advantage over classical generative models in runtime and the number of parameters needed to learn data distributions due to their strong expressive power. Generative modeling has been used as near-term applications for quantum processors [82], but as of yet not all models have been used in HEP applications. In the remainder of this section, we will overview the state-of-the-art results for two of the models above – the q(R)BM and the q(V)AE. In separate follow-on sections, we provide an in-depth discussion of two quantum generative models, the qGAN and the quantum circuit Born machine (QCBM), that have been used in HEP applications.

A. Quantum Generative Models

The (R)BM is a physically motivated neural network capable of generating new samples similar to the training data [76], which was previously introduced in Section IV. A network of classical spins are connected by weighted edges (w_{ij}) and on-site biases (b_i), which are optimized during training to find the ground state of the system's Hamiltonian

$$H_{(R)BM} = \sum_i b_i z_i - \sum_{\langle i,j \rangle} w_{ij} z_i z_j. \tag{3}$$

BMs a natural fit for quantum annealing [83–85] and quantum computing [86–88]; in the latter Ref. [88] shows



[illegible] (a): [illegible] (b): [illegible] (c):(Variational) [illegible] $(U)(\cdot)$: (d): [illegible] (e): [illegible]

polynomial speed-ups relative to classical training. Starting from a classical (R)BM, the neurons (or classical spins) are replaced by qubits and the Hamiltonian is translated into a unitary circuit or Ising model couplings.

VAEs [78] are artificial neural networks typically used for generative modeling applications. The word *autoencoder* refers to the fact that its architecture, comprising an encoder and a decoder, resembles a traditional autoencoder. Nonetheless, there are significant differences, especially in the mathematical formulation.

The goal of an autoencoder [89] is learning to map some input vector x to a compressed space vector z. In the language of generative modeling, this compressed space is known as the *latent space*. This latent space vector is fed into another network that reconstructs the input. The network that compresses the input vector x is known as the encoder, whereas the network that maps the latent vector z to the initial input x is the decoder. A figure of merit for the loss function training is simply the error of the reconstructed input state, which shall be minimized.

Notice, however, that there are no constraints on the latent variables z. This is the main reason why traditional autoencoders are useful for compressing information but lack a generative structure. Namely, the latent variables z rarely capture relevant information, or this information is not organized. This is the precise issue that VAEs tackle. To be specific, VAEs deal with the problem of the latent space irregularities by making the encoder return a simple distribution over the latent space, generally a Gaussian distribution, by adding extra terms in the loss functions to ensure a better organization of the latent space [90].

The encoder and decoder of a qVAE model are gates which transform an underlying wavefunction. At any point in the network, the wavefunction defines a distribution over a set of basis states meaning that non-Gaussian distributions can be prepared in the latent space. In Section it was discussed how qAEs have been used for anomaly detection IV. Other applications for qAEs include quantum data compression [73] and qVAEs have been used for generative modeling of classical data. In Ref. [91] a quantum generative process is used to produce handwritten digits, with competitive results on par with other classical methods.

B. Quantum Generative Models for HEP

For HEP applications, two quantum generative models have been used: the qGAN and the QCBM. There are many GAN architectures. However, all of them contain as basic components a generator, a discriminator, and an adversarial training procedure as suggested in the initial work of Ref. [92]. The qGAN follows the general structure of a classical GAN, whereas the QCBM uses only a parameterized circuit.

In general the generator G transforms samples from some prior, simple-to-sample noise distribution $z \sim p_{\text{prior}}(z)$ into new samples $G(\phi_g)$, thus mapping $p_{\text{prior}}(z)$ to a different distribution p_{fake} of generated data. The discriminator takes as input samples x, from the real distribution in the form of data, and tries to distinguish between fake data from the generator and this real data p_{real}. The adversarial training procedure alternates between the generator suggesting samples that are accepted by the discriminator and the discriminator detecting those samples suggested by the generator. Once this adversarial game reaches a stable equilibrium over training epochs, the generator robustly suggests new, previously unseen, samples of the target distribution.

In practice, often the binary cross-entropy is used as optimization objective. The individual loss functions can then be defined as

$$\mathcal{L}_G(\phi_g, \phi_d) = -\mathbb{E}_{z\sim p_{\text{prior}}(z)}[\log D(\phi_d, G(\phi_g, z))]\,, \tag{4}$$

$$\begin{aligned}\mathcal{L}_D(\phi_g, \phi_d) &= \mathbb{E}_{x\sim p_{\text{real}}(x)}[\log D(\phi_d, x)] \\ &\quad + \mathbb{E}_{z\sim p_{\text{prior}}(z)}[\log(1 - D(\phi_d, G(\phi_g, z)))]\,,\end{aligned} \tag{5}$$

for the generator and discriminator, respectively. As such the training corresponds to a minimax two-player game,

$$\min_{\phi_g} \mathcal{L}_G(\phi_g, \phi_d)\,, \quad \max_{\phi_d} \mathcal{L}_D(\phi_g, \phi_d)\,, \tag{6}$$

where the optimum uniquely corresponds to the Nash equilibrium between the loss functions.

The neural network architecture, input data, and complexity of the distributions all play a role in successfully training a GAN, both in the classical and the quantum context. Here, in particular, the potential to create more complex, varied outcomes from simpler network architectures in the quantum case could prove valuable. Studying this connection between quantum network architecture and complex outputs in detail provides interesting future opportunities. For example, in Ref. [93] a robust training advantage was established in the context of supervised learning.

In HEP data analysis, two applications stand out for their intensive computing resource requirements. These are the simulation of the detector and its various calorimeters on the one hand and the generation of process events via Monte-Carlo on the other. Both fields are under intensive research using classical methods due to the potential gains from any speed-up. Finding a path towards quantum advantage and speed-up could lead to a jump in progress in the future.

1. *Detector simulations*

The traditional Monte-Carlo-based simulations of HEP detectors are highly time-consuming. As an example, it is estimated that during the last years before the last LHC shutdown, the LHC experiments devoted more than 50 % of their WLCG computing resources to Monte-Carlo production [94]. At the HL-LHC, given the higher detector granularity, the larger complexity of physics events, and the statistics needed for accurate precision measurement, Monte-Carlo production is expected to increase by at least two orders of magnitude [94]. In this context, deep generative models and, in particular, Generative Adversarial Networks, have been investigated in a range of different use cases as possible alternatives for fast simulations. Examples include 3DGAN [95, 96] or CaloGAN [97].

In most cases, the GAN-based prototypes reach a high level of accuracy and produce realistic datasets. Given the increasing interest in quantum generative modeling, it is only natural to explore the possibility of replacing classical generative models using quantum or hybrid classical-quantum architectures. Example applications exist, proposing hybrid quantum GAN, where a quantum generator is trained against a classical discriminator to reproduce the output of electromagnetic calorimeters [88]. The Dual-Parameterized Quantum Circuit (PQC) GAN, is characterized by a classical discriminator and two quantum generators that take the form of PQCs, and it is capable of simulating reduced size pixelated images of calorimeter volumes [98]. The model was trained to reproduce one-dimensional energy distribution, generated inside the calorimeter volume by single electrons. The studies in Ref. [99] show that the results are stable on both simulated and real quantum hardware. Similarly, the work in Ref. [100] explores the performance of a hybrid prototype (based again on a quantum generator and a classical discriminator) implemented using photonic-based quantum computing [101], which has the great advantage of allowing natural representation of continuous variables in quantum states.

Despite the encouraging initial results described above, it should be noted that the use case of detector simulation presents important challenges related to the size of the distributions the generative models learn. In most cases, this scales with the number of sensors in the detectors and it is therefore challenging to embed in quantum circuits, especially on near-term devices.

2. *Monte Carlo event generation*

The use of GANs for applications in the simulation of HEP events has been a key focus in the context of classical machine learning [102–108]. Due to their ability to generate larger samples of a distribution given through a more limited data set, they present a unique opportunity for data augmentation and signal boosting applications.

An example application is to provide distributions of key experimental processes and to generate a larger sample using a qGAN. This was successfully implemented for the process of $pp \to t\bar{t}$ production at the LHC with $\sqrt{s} = 13$ TeV in [109]. Based on the data re-uploading idea [60], there a flexible network architecture was proposed that enabled an efficient representation of the desired generative model.

In particular, through hyperparameter tuning, it was found that a network in which each input dimension of the distribution is represented by a single qubit, entangled with its neighbors, and a single layer was sufficient to successfully train the GAN with a robust minmax condition found (with a few tens of thousands of epochs of training, comparable with classical experiences).

The low complexity of the quantum circuit used enabled its deployment on both superconducting and ion-based quantum computing architectures. Good signal-to-noise properties were observed in both cases.

Even though the work in this direction is at a proof-of-principle stage, results like these give hope for successful deployment on NISQ devices.

Further refinements, such as training efficiently for other processes through transfer learning, the inclusion of error correction techniques in the generation process, or further implementation of quantum components - as e.g. the given example implements a classical discriminator - are useful avenues to follow in the future.

The use of the trained qGAN is not limited to data augmentation purposes. Additionally, it can also be used as an intermediary encoder of the process generating functions. In Ref. [110] it was successfully demonstrated that a qGAN could be used to load the underlying information of the process onto qubits. A quantum amplitude estimation (QAE) algorithm subsequently performed the integration of the elementary process in one and two dimensions on these loaded qubits. In future, it would be interesting to pursue further opportunities to combine different quantum algorithms in this way. One could envision the construction of a “quantum pipeline” from raw data to refined results for physics observables in which different quantum algorithms are connected in such a way that they benefit from practical quantum advantages. For example, one could imagine feeding in data from an efficiently trained and sampling quantum network [111, 112] into subsequent quantum analysis tools, such as the QAE, via qGANs or exact encodings.

The QCBM is an example of an implicit model for generative learning [113] that generates data by measuring the system as a Born machine [82, 114]. Unlike the q(R)BM models, the unitary circuit is not defined with respect to a specific Hamiltonian, rather a QCBM is a parameterized unitary $U(\langle 0|\theta|0\rangle)$ that prepares N-qubits in the state $|\Psi_\Theta\rangle = U(\vec{\theta})|\Psi_0\rangle$. One can obtain a classical distribution over the 2^N computational basis states by measuring $|\Psi_\Theta\rangle$ in a fixed basis (usually the computational basis). Unlike the qGAN model, QCBMs can be trained using non-adversarial methods, either using gradient-free [82] or gradient-based [115] optimization. Training a QCBM is done by minimizing a loss function $\mathcal{L}(P_{target}, P_{QCBM})$ that computes the similarity between the target distribution and the distribution sampled from the QCBM. The study in Ref. [116], uses non-adversarial gradient-based training of 8- and 12-qubit QCBMs to generate joint distributions over 2 and 3 variables to generate synthetic data of a typical HEP process.

VI. QUANTUM-INSPIRED ALGORITHMS

The fusion of quantum computing and data analytics promises a revolution in machine learning and other optimization and related computational capabilities. However, there are only a few demonstrations of quantum advantage for very specific tasks. Given the current hardware and software limitations, an exciting exploration for HEP scientists is quantum-inspired algorithms, viewing information technologies from the vantage point of quantum to make gains in conventional systems.

Quantum-inspired algorithms involve a traditional (non-quantum) computer emulating certain aspects of quantum mechanics to gain a computational advantage. To date, there have been quite a few quantum-inspired algorithms tackling problems of optimization [117], machine learning [118], and linear algebra [119]. In addition, the Fujitsu Digital Annealer [120] is a classical analog of quantum annealers such as those produced by D-Wave.

In HEP, a method of increasing popularity rooted in quantum mechanical concepts is tensor networks (TNs) [121, 122] or tensor network states. TNs have been developed to investigate quantum many-body systems on classical computers by efficiently representing the exponentially large quantum wavefunction in a compact form.

Recently, it has been shown that TN methods can also be applied to solve ML tasks very effectively [123–125], yielding comparable results when benchmarked against NNs on standard datasets [123, 125, 126]. In HEP, TNs have been explored in an ML context to gain insight into the learned data by computing quantum correlations or entangling entropy.

In Ref. [127], a TN-based supervised learning approach to the identification of the charge of b quarks (*i.e.*, b or $\bar{b}$) is presented. Simulated data of b quarks generating jets produced in high-energy proton-proton collisions at the LHCb experiment are studied. Although the TN-based method yielded similar performance compared to a deep neural network classifier, some benefits of the TN-based approach were highlighted, including the power of compressing the network while keeping a high amount of information.

Similar conclusions were reported in Ref. [128], where a TN-inspired classifier is used to discriminate top jets over QCD jets. It is also noted that the TN learns the volume and correlations of the projected geometry of topological relations in the data, which is reflected by the entanglement entropy of the network. This observation can be exploited to reduce redundant information in the input data, thereby reducing the complexity of the network while maintaining a high classification performance.

VII. CHALLENGES

In this section, we describe some of the limitations that are common to the applications discussed in this manuscript.

A. Quantum annealing-based algorithms

Compared with other quantum computing paradigms, quantum annealing has the advantage that there are already commercially available processors with thousands of qubits, and these are expected to grow further in the near term. Nevertheless, is remains elusive a clear experimental demonstration of quantum advantage for quantum annealing.

Moreover, quantum annealers are severely limited in the tasks that they can perform. Specifically, they are only designed to solve quadratic unconstrained binary optimization (QUBO) problems [129, 130], although the more general quantum adiabatic computing model is universal [131].

A common problem with annealing-based solutions for HEP data analysis is the large overhead of converting the problem at hand to the corresponding QUBO formulation, that is, computing all the QUBO coefficients. The time taken in this pre-processing step largely differs from problem to problem, but, as we have seen, often scales worse than quadratically with instance size. Another challenge is the required annealing schedule to reach the global minimum, which determines the final complexity of the algorithm. This is difficult to estimate for real-world data analysis problems, but for many applications it is expected to scale exponentially with the number of variables in the problem.

Additionally, if a problem requires a significant degree of connectivity among its binary variables (also called logical qubits), it is extremely difficult to embed it onto the physical qubits. Indeed, some instances might take exponentially long for the embedding process [132].

For HEP problems specifically, the current methods for problem decomposition such as `qbsolv` by D-Wave are inadequate because of their lack of precision required for collider experiments. These problems also require significant hyperparameter optimization in addition to the pre- and post-processing subroutines for dealing with noise, freeze out, and competing strengths between spin couplings in the Hamiltonian and the chain strengths.

B. Gate-based/universal QC algorithms

Gate-based quantum computers have already been used to establish a clear advantage over the classical counterparts [133]. However, present-day universal quantum computers are noisy and limited in the number of qubits to serve many practical applications, in particular in HEP data analysis.

In the HEP context, a significant challenge is the interface between the classical data and the quantum computing architecture. Some proposals assume access to QRAM [18], which would allowed coherent access to previously stored classical data. However, despite existing theoretical proposals for building such devices [19], it is not expected that they could be implemented in the near term.

Considering solutions that do not require fault-tolerance, the current generation of quantum variational circuits often uses the amplitude or basis encoding schemes, as mentioned in Sec. IV B. While these are important first steps, each scheme has some shortcomings. For instance, the data that can be stored using the basis embedding protocol is linear in the number of qubits being used, much smaller than the Hilbert space, which grows like 2^N [134]. On the other hand, the amplitude encoding scheme does allow for storage of 2^N variables but is less effective in certain contexts [64]. As has been pointed out in [135], this may be due to the fact that non-linear transformations of the data are difficult to design in this context. However, how to move beyond these schemes is not obvious as other encoding mechanisms require a large amount of preprocessing on the input data; see *e.g.* Ref. [51]. Going forward, we will need to find methods for efficiently and densely embedding data in quantum registers in order to take full advantage of QC's expressiveness.

In addition to the inefficiency in embedding data onto the quantum computer, there is a lack of knowledge on the kind of problems where one embedding scheme might be better suited than the other. Furthermore, the understanding about how noise might impact the performance of training is somewhat unclear—while some studies advocate for noise adding much desired stochasticity during training, other studies present noise as a hindrance. In terms of logistics, accessing quantum computers on the cloud remotely and running jobs on them is extremely inefficient—in some cases, users have to wait for days before their jobs are run on the hardware.

Finally, it remains an open challenge to find a quantum algorithm for HEP data analysis with proven *exponential* speedup. In general, such algorithms are rare [136], requiring highly structured problems [137]. Importantly, Ref. [138] argues that with the current error-correction techniques we should not see a time advantage for problems with small polynomial speedups, at least when we consider practical input sizes.

VIII. OUTLOOK

Over the last couple of years, quantum computing techniques have been developed to explore the applicability of quantum computing to data analysis in HEP, promising better algorithms for ML, optimization, and other techniques. In this manuscript, we highlighted some areas where quantum computing applications have been explored for data analysis in HEP. The major drawback of these ideas is the lack of a fully mature quantum computer. The main limitations were discussed in Section VII. Naturally, the first question that arises from this statement is whether there will ever be a demonstration of quantum advantage. The second question to be answered is if there is a short-term gain in exploring quantum computing for data analysis in HEP. While it may be sometime before quantum computers mature enough to replace classical computing in certain HEP analysis workflows, there are quantum explorations appropriate even for today's data analysts. These applications can be categorized into main areas: (1) how HEP can benefit from the use of QC, and (2) how HEP can contribute to the quantum computing ecosystem.

In Section VI, we discussed how quantum-inspired algorithms might provide a new perspective to the main computational problems facing the HEP community. Due to the requirement of analyzing vast, highly correlated data in order to exploit the full physics potential of the LHC or other large sample size particle physics experiments such as DUNE or IceCube, it becomes more and more critical to develop a fundamental understanding of the data analysis methods applied. In this context, TNs are able to extract information not easily accessible to NNs, such as correlation functions and entanglement entropy which can be used to explain the learning process and subsequent classifications, paving the way to an efficient and transparent ML tool. Furthermore, TNs have been shown to reduce prediction times. In Ref. [127], it is expected that in the near future, prediction times will be reduced to the order of MHz, having the potential for deployment on FPGAs for real-time data acquisition and selection at HEP facilities.

As the size and quality of QCs continue to advance, with large-scale entanglement achieved on a range of platforms, so does the feasibility of using quantum machines to perform typical data analysis tasks in particle physics. In particular, there are various ways in which the field of ML may benefit from the advent of QC. These benefits range from speed-ups to specific subroutines, such as gradient descent and linear algebra, to quantum analogs of classical algorithms.

Finally, quantum computing benchmarks on HEP experimental data can be used to formulate metrics and benchmarks for emerging quantum computers and other quantum technologies. For reasons of cross-platform comparison and the identification of trends to make predictions, metrics and benchmarks have an important role to play even today, when quantum computers cannot, with rare exception, outperform conventional devices. One example is jet substructure, where analytics reconstruction techniques coexist with numerical MVA methods. The combination of a large amount of available data with an excellent theoretical understanding of the underlying physics in collider phenomenology provides an ideal environment to explore novel quantum and quantum/classical techniques.

ACKNOWLEDGMENTS

This work was partially supported by the Quantum Information Science Enabled Discovery (QuantISED) for High Energy Physics program at ORNL under FWP ERKAP61. This work was partially supported by the Laboratory Directed Research and Development Program of Oak Ridge National Laboratory, managed by UT-Battelle, LLC, for the U. S. Department of Energy.

DM, YO, and PB acknowledge project *QuantHEP – Quantum Computing Solutions for High-Energy Physics*, supported by the EU H2020 QuantERA ERA-NET Cofund in Quantum Technologies and FCT – Fundação para a Ciência e a Tecnologia (QuantERA/0001/2019), and thank the support from FCT, namely through project UIDB/50008/2020. DM acknowledges the support from FCT through scholarship 2020.04677.BD. AG, DL, DN, LS, and DZ acknowledge the support from the Department of Physics and Astronomy "Galileo Galilei" of University of Padova and from INFN (Istituto Nazionale di Fisica Nucleare).

[1] [illegible] *Machine Learning: Science and Technology*, [illegible]

[2] [illegible] *Nature Reviews Physics*, [illegible]

[3] [illegible] *International Journal of Theoretical Physics*, [illegible]

[4] [illegible] *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, [illegible]

[5] [illegible] *Physical Review E*, [illegible]

[6] [illegible]

[7] [illegible] *arXiv preprint quant-ph/0411152*, [illegible]

[8] [illegible] *Phys. Rev. Lett.*, [illegible]

[9] [illegible] *Journal of Instrumentation*, [illegible]

[10] [illegible] *Computing and Software for Big Science*, [illegible]

[11] [illegible]

[12] [illegible] *Eur. Phys. J. C*, [illegible]

[13] [illegible] *European Physical Journal C*, [illegible]

[14] [illegible] *Combinatorial Kalman Filter and High Level Trigger Reconstruction for the Belle II Experiment.* [illegible]

[15] [illegible] *arXiv:2104.11583*, [illegible]

[16] [illegible] *Quantum Computation and Information*, [illegible]

[17] [illegible] *Phys. Rev. Lett.*, [illegible]

[18] [illegible] *Physical Review*

Letters, [illegible]

[illegible] *Physical Review A*, [illegible]

[illegible] *Quantum Machine Intelligence*, [illegible]

[illegible] *Computing and Software for Big Science*, [illegible]

[illegible] *arXiv:1903.08879*, [illegible]

[illegible] *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, [illegible]

[illegible] *Computer Physics Communications*, [illegible]

[illegible] *Nucl. Instrum. Meth. A*, [illegible]

[illegible]

[illegible] *Quantum Machine Intelligence*, [illegible]

[illegible] *Physics Letters*, [illegible]

[illegible] *Phys. Rev. Lett.*, [illegible]

[illegible] *Journal of Computational Physics*, [illegible]

[illegible] *Journal of High Energy Physics*, [illegible]

[illegible] *Phys. Rev. D*, [illegible]

[illegible]

[illegible] *arXiv:2012.14514*, [illegible]

[illegible]

[illegible] *Physical Review Letters*, [illegible]

[illegible] *Nature*, [illegible]

[illegible] *Physical Review A*, [illegible]

[illegible] *Physical Review D*, [illegible]

[illegible] *arXiv:2111.07806*, [illegible]

[illegible] *arXiv preprint: 2003.02181*, [illegible]

[illegible]

[illegible] *Nature Computational Science*, [illegible]

[illegible] *Quantum Science and Technology*, [illegible]

[illegible] *Physical Review A*, [illegible]

[illegible] *Physical Review A*, [illegible]

[illegible] *Quantum Machine Intelligence*, [illegible]

[illegible]

[illegible]

[illegible] *Advances in neural information processing systems*, [illegible]

[illegible]

[illegible]

[illegible]

[illegible] *Computing and Software for Big Science*, [illegible]

[illegible] *Nature*, [illegible]

[illegible] *Journal of Machine Learning Research*, [illegible]

[illegible] *Advances in Optimization and Numerical Analysis*, [illegible]

[illegible]

[illegible]

[illegible] *EPJ Web of Conferences*, [illegible]

[illegible] *Quantum*, [illegible]

[illegible] *Journal of High Energy Physics*, [illegible]

[illegible] *Journal of Physics G: Nuclear and Particle Physics*, [illegible]

[illegible] *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, [illegible]

[illegible] *b*-[illegible] *arXiv:2202.13943*, [illegible]

[illegible] *Physical Review Letters*, [illegible]

[illegible]

[illegible] *Physical Review Research*, [illegible]

[illegible] *Computing and Software for Big Science*, [illegible]

[illegible] *npj Quantum Information*, [illegible]

[illegible] *arXiv preprint: 2012.12177*, [illegible]

[illegible] *arXiv preprint: 2101.06189*, [illegible]

[illegible]

[illegible] *Quantum Science and Technology*, [illegible]

[illegible] *Journal of High Energy Physics*, [illegible]

[illegible] *arXiv preprint arXiv:2112.04958*, [illegible]

[illegible] *Cognitive Science*, [illegible]

[illegible] *Chapter 6,*, [illegible]

[illegible] *arXiv preprint arXiv:1312.6114*, [illegible]

[illegible]

[illegible] *Pattern Recognition*, [illegible]

[illegible] *Physical Review Letters*, [illegible]

[illegible]

[illegible] *npj Quantum Inf.*, [illegible]

[illegible] *Phys. Rev. A*, [illegible]

[illegible]

[illegible] *Physical Review X*, [illegible]

[illegible]

[illegible] *Phys. Rev. X*, [illegible]

[illegible] *Quantum Machine Intelligence*, [illegible]

[illegible] *AIChE journal*, [illegible]

[illegible] *IEEE transactions on pattern analysis and machine intelligence*, [illegible]

[illegible] *Quantum Science and Technology*, [illegible]

[illegible] *Advances in neural information processing systems*, [illegible]

[illegible] *Nature Physics*, [illegible]

[illegible] *Comput. Softw. Big Sci.*, [illegible]

[illegible] *J. Phys.: Conf. Ser.*, [illegible]

[illegible] *2018 25th IEEE International Conference on Image Processing (ICIP)*, [illegible]

[illegible] *Phys. Rev. D*, [illegible]

[illegible] *EPJ Web of Conferences*, [illegible]

[illegible]

[illegible]

[illegible] *Physical Review Research*, [illegible]

[illegible] *arXiv preprint arXiv:2012.11944*, [illegible]

[illegible] *SciPost Physics*, [illegible]

[illegible] *arXiv preprint arXiv:2008.08558*, [illegible]

[illegible] *SciPost Physics*, [illegible]

[illegible] *SciPost Physics Core*, [illegible]

[illegible] *SciPost Physics*, [illegible]

[illegible] *SciPost Physics*, [illegible]

[illegible] *arXiv:2110.06933*, [illegible]

[illegible] *arXiv preprint arXiv:2201.01547*, [illegible]

[illegible] *arXiv preprint arXiv:2106.12627*, [illegible]

[illegible] *Physical Review Letters*, [illegible]

[illegible]

[illegible] *npj Quantum Information*, [illegible]

[illegible] *Phys. Rev. A*, [illegible]

[illegible]

[illegible]

[illegible] *Expert Systems with Applications*, [illegible]

[illegible] *IEEE Transactions on Neural Networks and Learning Systems*, [illegible]

[illegible] *Physical Review A*, [illegible]

[illegible] *Frontiers in Physics*, 7 [illegible]

[illegible] *Journal of Physics A: Mathematical and Theoretical*, [illegible]

[illegible]

[illegible] *Quantum Science and Technology*, [illegible]

[illegible]

[illegible] *Advances in Neural Information Processing Systems*, [illegible]

[illegible]

[illegible] *npj Quantum Information*, (7), [illegible]

[illegible] *Journal of High Energy Physics*, [illegible]

[illegible] *Quantum Information Processing*, [illegible]

[illegible] *Scientific reports*, [illegible]

[illegible] *SIAM Journal on Computing*, [illegible]

[illegible] *Quantum Information Processing*, [illegible]

[illegible] *Nature*, [illegible]

[illegible] *CoRR*, [illegible]

[illegible] *Phys. Rev. Lett.*, [illegible]

[illegible] *J. ACM*, [illegible]

[illegible] *Theory of Computing*, [illegible]

[illegible] *PRX Quantum*, [illegible]