Review Article

Quantum Computing for Genomics: Conceptual Challenges and Practical Perspectives

Aurora Maurizio [1,*] and Guglielmo Mazzola [2,†]
[1] Center for Omics Sciences, IRCCS San Raffaele Scientific Institute, 20132 Milan, Italy
[2] Department of Astrophysics, University of Zürich, Winterthurerstrasse 190, 8057 Zürich, Switzerland

(Received 7 August 2025; accepted 1 October 2025; published 30 October 2025)

We assess the potential of quantum computing to accelerate computation of central tasks in genomics, focusing on often-neglected theoretical limitations. We discuss state-of-the-art challenges of quantum search, optimization, and machine learning algorithms. Examining database search with Grover's algorithm, we show that the expected speedup vanishes under realistic assumptions. For combinatorial optimization prevalent in genomics, we discuss the limitations of theoretical complexity in practice and suggest carefully identifying problems genuinely suited for quantum acceleration. Given the competition from excellent classical approximate solvers, quantum computing could offer a speedup in the near future only for a specific subset of hard enough tasks in assembly, gene selection, and inference. These tasks need to be characterized by core optimization problems that are particularly challenging for classical methods while requiring relatively limited variables. We emphasize rigorous empirical validation through runtime scaling analysis to avoid misleading claims of quantum advantage. Finally, we discuss the problem of trainability and data-loading in quantum machine learning. This work advocates for a balanced perspective on quantum computing in genomics, guiding future research toward targeted applications and robust validation.

DOI: 10.1103/h49j-bsc6

I. INTRODUCTION

Genomics inherently presents a suite of computationally challenging problems due to the sheer volume and complexity of biological data involved. The ability to accelerate these computations is of paramount importance for deepening our understanding of fundamental biological processes, uncovering disease mechanisms, and advancing personalized medicine [1,2]. In the past decade, quantum computing [3], an emerging computing paradigm leveraging the principles of quantum mechanics, has garnered significant attention as a potential platform for achieving substantial speedups across various scientific and technological domains [4–8]. This naturally raises the compelling question of whether the unique capabilities of quantum computation can be effectively harnessed to address the demanding computational bottlenecks within the field of genomics.

There is a growing body of literature discussing the prospects of quantum computation for biological and healthcare applications [9–22]. While useful for navigating the zoo of quantum algorithms [23], they necessarily may lack a detailed view of genomics problems.

For instance, given the inherent big-data nature of many problems within genomics, it is tempting to directly assign potential quantum speedups to tasks like sequence alignment by leveraging textbook algorithms such as Grover's search [8,24]. Similarly, considering claims regarding the computational advantage of quantum annealing hardware over classical solvers [25–27], one might be inclined to foresee the quantum advantage across all genomics applications that rely on core combinatorial optimization routines. However, as we will discuss, these direct and perhaps overly optimistic associations often overlook crucial practical limitations and theoretical nuances that significantly impact the potential for genuine quantum acceleration in real-world genomic analyses.

The goal of this manuscript is to offer a critical perspective on the usefulness of quantum algorithms in genomics. We believe that this is particularly useful to temper some of the hype surrounding quantum computing. Rather than being a compilation of works discussing the literature on *quantum genomics* [12], we focus on the algorithms and their practical implementations. Here we specifically discuss quantum algorithms for database search, optimization, sampling, and machine learning (ML).

The conceptual map of the work is presented in Fig. 1. The manuscript is organized as follows: In Sec. II, we introduce digital quantum computing in an accessible way, along with its motivation and the definition of scaling advantage, while Sec. III briefly introduces the field of genomics. In Sec. IV, we discuss Grover's algorithm for database search, highlighting the profound challenges connected with data loading and sketching possible workarounds. In Sec. V, we turn to the computationally heavier task of optimization, which is

*Contact author: maurizio.aurora@hsr.it
†Contact author: guglielmo.mazzola@uzh.ch

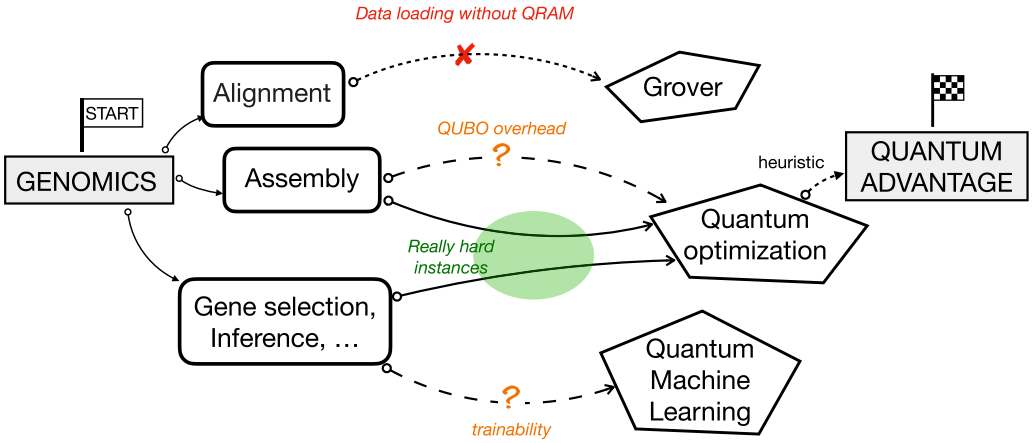

FIG. 1. We schematically sketch the main connection and topics discussed in the manuscript. Concerning the genomics side, we discuss mainly three applications: alignment, assembly, and problems beyond that (here briefly labeled as gene selection and inference). We discuss possible quantum algorithms that have already been proposed to tackle these problems. Here, for the first time, we discuss in detail also the main conceptual and practical limitations (red, orange). We speculate that quantum advantage may be found if one identifies really hard optimization problems that do not suffer from typical overhead connected with embedding into a quantum algorithm and hardware (green pathway). However, quantum search algorithms and quantum ML approaches may suffer from practical and conceptual limitations.

inherent in many genomic tasks, from assembly to inference. We discuss the applicability of complexity theory in real-world instances, and propose problems that are truly hard in practice. Finally, we discuss current limitations of quantum ML in Sec. VI. We draw our conclusions in Sec. VII. In Appendix A we provide a Table I of used acronyms to assist the reader.

II. BRIEF OVERVIEW OF QUANTUM COMPUTING

The goal of this section is to introduce the key concepts regarding the fault-tolerant quantum computation, quantum gate frequency, and scaling advantage. In Appendixes B and C we present basic concepts of quantum computation and a set of common misconceptions.

In this manuscript, we mostly focus on digital, fault-tolerant quantum computation [3], i.e., assuming no hardware noise. We will adopt the convention that a qubit is a logical qubit, not a physical one, subject to decoherence and noise [28]. Generally, it is expected that to fabricate one logical qubit, several physical qubits need to be bundled, with a set of operations designed to detect and correct errors. Two promising methods to achieve error correction in superconducting qubit platforms are the "surface" code (pursued by Google) [29,30] and the "bicycle" code (pursued by IBM) [31,32]. In short, the surface code [33,34] requires local physical qubit connectivity, and a sustain a high error threshold at the cost of requiring many physical qubit to encode a logical one. This means that algorithms requiring only hundreds of logical qubits will, in turn, necessitate millions of physical qubits [35]. Bicycle codes, however, aim for much lower qubit overhead and potentially better performance, often at the expense of more complex qubit connectivity and decoding.

The possibility of achieving quantum advantage with heuristic algorithms and noisy analog machines [36], perhaps in specific models and instances, is not ruled out. However, here we will mostly consider fault-tolerant quantum computation because it is hardware agnostic, allowing us to discuss algorithms at a high level without the need for modeling the microscopic quantum processes (e.g., quantum tunneling, decoherence, etc.) occurring in the hardware [37,38].

Scaling advantage versus gate time

There are two kinds of quantum algorithms [8,39]: those with a provable scaling advantage over their best-performing classical counterparts, and those that offer only a heuristic speed-up, i.e., that cannot be demonstrated mathematically. A scaling advantage means that the runtime of the quantum algorithm, T_Q, scales with the system size N of the problem in a more favorable manner than the runtime of a classical algorithm, T_C.

Let us consider, as an example, the task of database search over a list of N elements, the worst-case performance for the conventional approach requires N lookups, while Grover's algorithm only requires $\sqrt{N}$ lookups, realizing an appealing quadratic speedup. However, there are two potential issues to consider in more detail: (a) the quantum database lookup must be efficient and should not scale with N in a way that negates the speed-up, and (b) even if the lookup can be performed with a number of operations independent of N (in jargon, a constant-depth oracle), thereby retaining the full quadratic speed-up, the fault-tolerant quantum logic gates operate at a much slower clock frequency compared to conventional computers [35,40–42].

The typical quantum gate frequency will depend on the architecture and error correction code implemented in the future, but estimates suggest around 10 kHz [40]. This implies that quantum advantage, if any, will only occur for sufficiently large system sizes. The size at which the quantum algorithm outperforms the classical one is referred to as the threshold size. This situation is shown in Fig. 2, and discussed in the context of genome search in Sec. IV B.

To conclude, the existence of a provable quantum advantage is not always sufficient to predict practical quantum advantage in real-life use cases. A polynomial speed-up can be overshadowed by the large prefactor due to gate times,

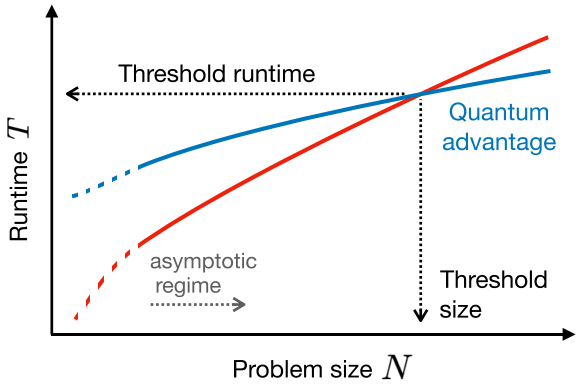

FIG. 2. Illustrative example of the scaling of runtimes T for quantum (blue line) and classical (red line) algorithms in a case where quantum speed-up is possible. In the asymptotic regime, i.e., for sufficiently large sizes, the two runtimes scale differently with N. Here we consider an unspecified example in which we can precisely define the runtime needed to reach a well-defined metric for considering the problem solved (e.g., accuracy or success probability).

revealing itself only for problem sizes that are unrealistically large and therefore not practical.

III. BRIEF OVERVIEW OF GENOMICS

Genomics focuses on decoding and interpreting the complete DNA sequence of an organism. Modern sequencing technologies produce large numbers of *reads*, which are short fragments of DNA whose sequence is determined experimentally. Typical reads consist of a few hundred nucleotides, far shorter than the full human genome ($\sim 3 \times 10^9$ nucleotides). The central challenge is to reconstruct biologically meaningful information from this fragmented data. A central resource is the *reference genome*, a previously assembled consensus sequence that provides a standardized coordinate system for interpretation. Individual reads can be computationally aligned to the reference, i.e., placed at the most likely genomic position from which they originate. Alignment is fundamental in medical applications: it allows the identification of genetic variants such as single-nucleotide polymorphisms (SNPs), insertions, deletions, or structural rearrangements, all of which can underlie diseases. Accurate alignment is essential in cancer genomics, where distinguishing somatic mutations (nonheritable genetic changes that accumulate in specific tissues) from germline inherited variation is critical, and in clinical diagnostics, where it enables the identification of genetic variants responsible for inherited or rare diseases.

In cases where no reliable reference is available, e.g., for newly discovered species or nonmodel organisms, one may attempt a *de novo* assembly, where the genome is reconstructed directly from overlaps between reads. Assembly provides an unbiased view of the genome but is computationally demanding and more challenging for large or repetitive genomes.

Once genetic variants are identified, their statistical association with observable traits can be investigated through various methods, including *genome-wide association studies* (GWAS) [43,44], *linkage analysis* in family cohorts [45], *quantitative trait locus* mapping [46], and integrative approaches that combine genomic data with transcriptomic, epigenomic, or proteomic information [47]. In GWAS, for instance, statistical models are applied to test whether the presence of a variant is significantly correlated with a phenotype across a large population. This strategy has been instrumental in identifying genetic risk factors for complex diseases such as diabetes, cardiovascular disorders, and neurodegenerative conditions.

IV. QUANTUM SEARCH IN GENOMICS

A. Adapting the Grover algorithm for DNA alignment

The idea of obtaining a quadratic speed-up for search problems, using the Grover algorithm, is particularly appealing in applications such as sequencing and alignment. The first mention of such applications dates back to 2000 [48], with the initial use of the phrase *quantum bioinformatics*. The task involves finding the location of a target sequence (or string) of length M within a database of length $N \gg M$. Examples include searching within the human DNA, where the string represents a sequence of DNA bases (A, T, G, C), or at higher levels, a sequence of amino acids, and so on. Several works propose a quantum algorithm solution for the alignment problem [49–53]. However, in this section, we present arguments questioning the real scalability of such methods in a practical setting.

1. Data encoding

First, we need to consider how to encode our problem-specific variables into binary form. For instance, in the task of finding a string made of M DNA bases, such as ATGAAC..., within a larger DNA segment of N elements, we need a way to represent the four DNA bases using binary encoding. Since each DNA base can take one of four values, we require two qubits to encode this information. We can assign the following convention: A = "00," T = "01," G = "10," C = "11." Similarly, other data types can be encoded in a similar way. For example, encoding amino acids would require 5 bits since there are 20 different amino acids, and $2^4 < 20 < 2^5$.

In DNA read alignment, it is necessary to find the pattern and its location in the genome. The data is thus the labeled set of $N - M + 1$ of all possible strings of length M, inside the full genome of length N (see Fig. 3).

At this point, there are basically two possible directions in which the Grover algorithm can be adapted to this problem.

2. Grover without QRAM

This method was originally proposed in Ref. [48], and also adapted in Ref. [51]. The core idea of this approach is to entangle the qubit register of the indexes with the qubit register of the data. The index can be encoded in binary form, therefore requiring $q_i = \log_2(N)$ qubits (assuming N being a power of 2 for simplicity). The data register requires $q_d = 2M$ qubits, assuming the encoding above relating DNA elements and bits. For the human genome case, $N \approx 3 \times 10^9$ such that $\log_2(N) \approx 32$, while typical read lengths are in the range $M \sim 50$–100.

This way to encode the data and the indexes together provides a practical way to construct the circuit but introduces a

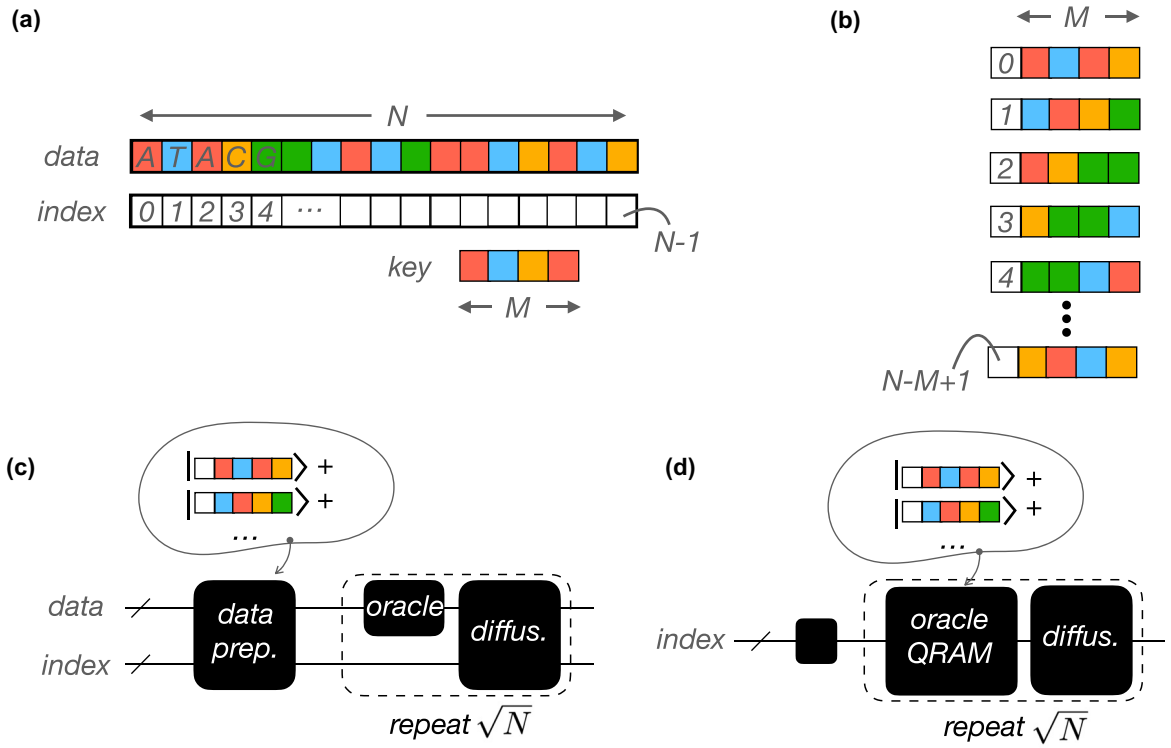

FIG. 3. DNA reads alignment using the Grover algorithm. (a) Sketch of the DNA portion of length N. Each position is labeled by an index. The task is to find the "key" string of length M inside the genome of lenght N. (b) The problem is translated into finding the "key" element in a database of $N - M + 1$ strings of length M. (c) Grover algorithm implementation that does not require QRAM. It features a *index* register, made of $\log_2(N)$ qubits, and a *data* register, made of $2M$ qubits. The entangled state that encodes the full ordered database is created using a data preparation unitary, which requires N operations to encode the data exactly. (d) Grover algorithm implementation with QRAM. In this case, the task of loading the data in superposition is offloaded to the Grover oracle.

costly operation for creating the initial state. A toy model of this problem is provided in Appendix E. Unfortunately, given that the genome data is not structured, the depth of the circuit needed to load the data cannot scale more favorably than N.

The data-loading problem also arises in quantum linear algebra applications [23]. In this case, the HHL quantum algorithm [54] solves a system of n equations in $\log(n)$ steps, thus realizing an exponential speed-up compared to the classical method. However, the exponential advantage vanishes if one considers the loading of the $\mathcal{O}(n^2)$ matrix elements. The usual way forward is to assume the existence of a quantum random access memory (QRAM) [55,56], which is a procedure capable of loading a bunch of classical data into superposition. In some rare cases, it can be argued that all variables can be autogenerated using a much smaller number of inputs [57]. However, this seems hardly the case, to say the least, for the human genome dataset.

3. Grover with QRAM

The second approach to genome search using Grover's algorithm is to utilize QRAM [55] within the oracle. Given two registers, $|i\rangle_{\text{index}}|0\rangle_{\text{data}}$ as before, one first create a superposition of all addresses. This is inexpensive, requiring just one layer of Hadamard gates for each of $\log_2(N)$ qubits (if N is a power of two, otherwise see Ref. [58]), $1/\sqrt{N}\sum_{i=0}^{N-1}|i\rangle_{\text{index}}|0\rangle_{\text{data}}$, and then obtain the superposition of values

$$\frac{1}{\sqrt{N}}\sum_{i=0}^{N-1}|i\rangle_{\text{index}}|0\rangle_{\text{data}} \xrightarrow{\text{QRAM}} \frac{1}{\sqrt{N}}\sum_{i=0}^{N-1}|i\rangle_{\text{index}}|d_i\rangle_{\text{data}}. \quad (1)$$

The marked entry is then identified within the oracle, and a minus sign is applied to the corresponding index. The rest of the Grover algorithm proceeds as described in textbooks.

The crucial weak point of this black box procedure is that it is still under debate whether QRAM can be experimentally realized, even in the distant future [56]. For a detailed explanation of the conceptual and experimental challenges of QRAM, we refer to the recent review [59].

A possible solution could be approximate loading of the data [60,61] using a parameterized low-depth quantum circuit. The parameters in the variational circuit could be obtained just once through an expensive learning or optimization process, but could then be transferable. Given that the core primary sequence of the human reference genome is fundamentally stable and major updates occur only every few years [62–64], this idea could, in principle, make sense. Nonetheless, It is not clear whether effective variational training can be performed if the target distribution lacks structure [65,66]. However, this is an active area of research. Reference [67] discusses the

gate requirements for approximate loading of the genomes of bacteriophages and SARS-CoV2 protein.

B. Constraining quantum runtimes for a simple alignment task

Finally, we check if a quadratic speed-up would still provide a meaningful advantage, assuming a fault-tolerant gate time and QRAM. Most bioinformatics tools typically address the full problem of alignment, inherently dealing with factors such as sequencing errors. On a typical high-performance computing setup with multiple CPU cores (e.g., 16–32 cores) and sufficient memory (e.g., 64–128 GB RAM), the alignment process for a whole genome sequencing of a human sample at 120× coverage and 100 bp paired end reads using, e.g., Burrows-Wheeler Aligner might take anywhere from a few hours to a day, depending on the available resources and system load.

In contrast, if one considers the idealized core problem of simply finding an exact string of length M within a database of length N, the classical runtime is significantly shorter. Therefore, a meaningful comparison of quantum and classical performance should be confined to this subtask. Modern bioinformatics tools employ highly optimized string-matching algorithms and data structures. For the task of exact matching a $M = 100$bp string over the whole human genome, $N = 3 \times 10^9$ bp long, these tools may take less than 1 minute. A quadratic speed-up implies a number of "oracle" calls in Grover's algorithm, which is $N_{\text{calls}} = \sqrt{N} \approx 6 \times 10^4$. We therefore ask the total "quantum" runtime, $T_Q = t_{\text{call}} \times N_{\text{calls}}$, where t_{call} is the physical runtime to execute one oracle call (see, e.g., Fig. 6), to be shorter than 1 minute. This yields a $t_{\text{call}} = 1$ ms time to execute one step of Grover's algorithm.

For a logical gate frequency of 10 kHz, quantum advantage would only be achievable if a single step of the Grover oracle consisted of an unrealistically short circuit with a depth of 10 (given that 1/(10 kHz) × 10 = 1 ms). Conversely, an error-correcting architecture that could enable a much more optimistic logical gate frequency of 10 MHz would permit circuits as deep as 10 000 layers.

V. OPTIMIZATION PROBLEMS IN GENOMICS AND QUANTUM ADVANTAGE

A. Optimization problems in genomics

Hopes of achieving quantum advantage are more realistic for harder computational problems in genomics, which are connected with combinatorial optimization. Many computational problems in genomics can be recast as optimization problems. Below, we provide some popular examples.

1. Sequence alignment

DNA sequencing is becoming faster and cheaper at a rate far beyond Moore's Law. Modern sequencing technology can now generate millions of reads from a sample, leading to an explosion of biological data that presents significant computational challenges [68].

Since memory access is a bottleneck for quantum algorithms, we expect that a practical quantum advantage is more likely for combinatorial problems defined on a relatively small input set N, but for which the time to solution grows exponentially, i.e., $\exp(N)$, or at least with a very steep polynomial degree.

Not all optimization problems fit this description. For instance *pairwise sequence alignment* using dynamic programming algorithms, such as Needleman-Wunsch [69] and Smith-Waterman [70], has a time and space complexity of $\mathcal{O}(NM)$, where N and M are the lengths of the two sequences. Moreover, there exist heuristic methods such as the basic local alignment search tool (BLAST) that achieve almost linear scaling [71]. These algorithms can be further accelerated by using of special-purpose architectures such as GPUs or FPGAs. Therefore, we do not expect an advantage from quantum computing in this case [41,72].

Multiple sequence alignment (MSA) is instead a challenging NP-hard problem, where the time to find the *exact* optimal solution grows exponentially with sequence number and length, making it a candidate for quantum advantage. Despite this, approximate methods, though not optimal, are widely used in tasks like *de novo* assembly. These rely on heuristics to manage hidden combinatorial complexity.

This problem also connects to graph similarity tasks. Quantum algorithms using Hamiltonian simulation [73] or support vector machines with graph kernels [74] have been proposed. For example, the overlap-layout-consensus (OLC) approach [75–78] compares all read pairs to find overlaps. The layout phase, i.e., finding a Hamiltonian path, is NP-hard, so heuristics are used.

Alternatively, graph-based methods like de Bruijn graphs are especially effective for assembling large volumes of short reads [79,80].

2. Genome-wide association studies

Computationally hard problems in genomics extend beyond alignment and assembly tasks. For instance, GWAS involve complex search and optimization tasks [81] as they aim to identify genetic SNPs that are associated with diseases.

3. Phylogenetic inference

Another genomics subfield of interest is *phylogenetic inference*, also known as phylogenetic tree reconstruction. This is the process of inferring the evolutionary relationships between a set of biological entities (e.g., species, cells, genes) based on their DNA, RNA, or protein sequences [82,83]. The core optimization problem in phylogenetic inference lies in searching the vast space of possible tree topologies (the branching patterns) and branch lengths (representing the amount of evolutionary change) to find the tree that best explains the observed DNA, RNA data under a specific model of sequence evolution. The number of possible unrooted binary trees grows exponentially with the number of leaves, making an exhaustive search computationally infeasible. Again, powerful heuristic search strategies are applied, these include approximate maximum likelihood [84], branch and bound [85], distance-based methods [86].

4. Haplotype phasing

NP-hard optimization also appears in the *haplotype phasing* problem [87], which aims to determine the specific allele combinations on chromosome copies. Using sequencing reads

spanning multiple heterozygous sites, researchers infer which alleles are inherited together. However, the number of possible haplotype configurations grows exponentially with the number of such sites.

This problem can be reformulated as a *max-cut* problem [88,89], where a fragment graph is built: vertices represent alleles, and edges connect those co-occurring in reads. Edge weights indicate whether the co-occurrence supports the same (positive) or different (negative) haplotypes. The goal is to partition the graph into two sets (haplotypes), maximizing the cut weight.

As max-cut is NP-hard, approximate solutions are used [90], including Monte Carlo methods [91]. Due to its natural mapping to the Ising Hamiltonian [92], max-cut is a key benchmark in quantum optimization research [93–96].

B. Complexity theory classes versus practical problems

1. NP-hard problems

Quantum methods are expected to demonstrate competitive advantages earlier in problems that are harder than polynomial [41]. From a purely theoretical point of view, an exponential quantum advantage is not expected for optimization problems belonging to the NPO-hard (Nondeterministic Polynomial-time Optimization) class. However, this does not exclude the possibility of large speed-ups in practical problems. Indeed, complexity theory typically deals with *worst-case* scenarios. In practical applications, one deals mainly with *average-case* instances [97,98].

Moreover, we suggest that many problems in genomics also defy conventional complexity classes since (i) we may be interested in reasonably good solutions rather than the exact one, and (ii) the input data itself can contain experimental errors, such that the exact solution for the slightly misdefined problem is not expected to be exact for the "true" data.

This strongly supports the use of heuristic algorithms, which, while lacking formal guarantees, often strike an excellent balance between cost and accuracy. The potential benefits of such algorithms must be assessed empirically, depending not only on the problem class but also on the specific encoding used.

2. Polynomial time approximations

Use cases in genomics should be carefully chosen in advance to enhance the likelihood of achieving a potential quantum advantage. Indeed, not all NPO-hard optimization problems are challenging in the same way, in practice. Most known complexity theory results hold for the task of finding the exact solution of a problem, but in practice, if we relax this goal to simply find a good approximation, then the same problem may belong to a different, "easier" class [97]. For instance, the traveling salesman problem (TSP), whose general formulation is NP-hard, may indeed change their complexity class in polynomial-time approximation scheme (PTAS) in some structured cases [99]. Practice also shows that the TSP can be approximately solved for up to millions of nodes [97].

C. From genomics to optimization and back

In this section, we aim to briefly review previous attempts at solving problems in genomics using quantum optimization ("bottom-up"), and to propose a new "top-down" pathway for connecting the two disciplines.

1. Bottom-up studies

So far, quantum computing has been proposed for solving optimization problems arising from genomic tasks, such as sequence alignment [100], *de novo* assembly [101–104], phylogenetic trees [105,106], transcription factor-DNA binding [107], epistasis analysis [108], codon optimization [109], densest k-subgraph [110], and secondary structure prediction of mRNA sequences [111]. The common strategy is to map these problems into quadratic unconstrained binary optimization (QUBO) problems, which, being defined in terms of binary variables, are well-suited for implementation in quantum software and hardware [92].

QUBOs correspond to Hamiltonians that can be implemented on quantum annealers and have been the subject of empirical studies for more than a decade [112–115]. However, QUBO problems can also be executed on digital quantum devices and tailored to algorithms such as the quantum approximate optimization algorithm (QAOA) [116]. The QUBO method requires minimizing the following cost function

$$H(x) = \sum_i h_i x_i + \sum_{i<j} J_{ij} x_i x_j, \tag{2}$$

where x_i are binary variables with values in ± 1, and h_i, J_{ij} are system dependent couplings. A real-world optimization problem, defined using its "native" variables, weights, and constraints, needs to be recast in this format [117].

Most of these studies take a "bottom-up" approach, directly seeking evidence of quantum speedup for use cases of interest. While the merit of these studies lies in demonstrating how real-world problems can be—in principle—mapped onto a given quantum platform, they usually do not prove or disprove a quantum advantage. In reality, even empirical demonstrations of scaling advantages require careful numerical experiments, making the task highly nontrivial.

In Appendix F we present a detailed framework to rigorously assess the scaling advantage by comparing the optimal time to solution and avoiding artificially favoring quantum methods at small sizes.

2. Practical limitations of QUBO formalism

A NP-hard problem can be reformulated as another with only polynomial overhead [92,118]. While complexity theory considers polynomial overhead negligible, in practice, where powerful classical heuristics exist, such overheads may make a critical difference. In Fig. 4 we qualitatively illustrate this scenario. In practice, mapping an original optimization problem into a QUBO often introduces significant memory and time overhead [117]. This is particularly true when the original problem includes constraints that rule out large regions of the computational space, as these would encode unphysical or invalid solutions.

For example, the TSP, defined over N cities (nodes), requires N^2 variables (hence qubits) when mapped into a QUBO formulation [92,118]. The same quadratic overhead affects existing QUBO-based approaches to genome assembly [101,102]. Therefore, solving the TSP using its "native"

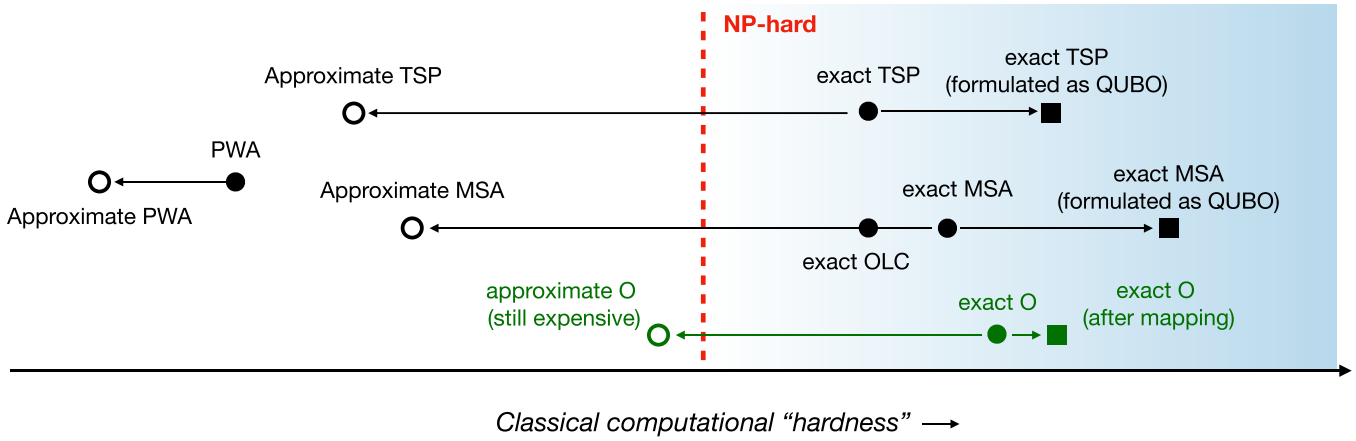

FIG. 4. Optimization problems in genomics, classical hardness, and approximations. We sketch the comparative computational hardness of several optimization problems discussed in the text, including pairwise alignment (PWA), multiple sequence alignment (MSA), and the traveling salesman problem (TSP). The exact formulations (indicated with solid circles) of MSA and TSP belong to the NP-hard class (whose boundary is drawn as dashed-red line), while PWA can be solved in polynomial time. All these problems can also be approached with approximate methods (indicated with empty circles), which trade accuracy for speed. For example, PWA can be solved with nearly linear scaling using BLAST, and the overlap-layout-consensus (OLC) problem, while still theoretically NP-hard, can also be solved approximately in practice. We further illustrate that these native NP-hard problems can be translated into QUBO form (solid squares), typically incurring a polynomial overhead. This implies that a quantum algorithm must solve a version of the problem that is harder than the original formulation. Last, we sketch the likely necessary conditions for an generic optimization problem (labeled as "O") to be amenable to quantum advantage (green symbols): (i) all known classical approximations remain computationally challenging or perform poorly in terms of accuracy, and (ii) the problem is already formulated in a way that minimizes the overhead associated with quantum embedding.

formulation is much more efficient than doing so via a QUBO mapping.

Another significant challenge arises when applying the QUBO problem to quantum machines with limited connectivity, such as current quantum annealers. Due to the sparse connectivity of the hardware, a mapping of the original problem onto the machine is necessary for fully connected QUBO problems. Embedding fully connected graphs introduces another quadratic overhead, which compounds the previously mentioned overhead [119]. This can be particularly frustrating since the all-to-all connectivity requirement stems from the QUBO formulation itself, rather than the "native" problem!

However, there are also counter-examples where the QUBO formulation provides an enhancement already at the problem-formulation level. For a lattice protein design task, the resulting QUBO model solved on classical machines outperforms previous classical state-of-the-art solvers [120]. Reference [121] compares the performance of quantum and classical optimization, including a genomic problem, with different QUBO mappings.

3. Top-down studies

An alternative, "top-down," strategy could be to first identify which general classes of optimization problems are particularly amenable to quantum speedup and then determine whether such problems appear in genomic pipelines. Recently, a portion of the optimization community proposed benchmarks for evaluating progress in quantum optimization [97]. The proposed criteria are: (i) The problems must be empirically known to be sufficiently hard even at limited sizes. (ii) The problems can be casted in binary format, and (iii) approximate classical solvers perform comparatively worse in these cases. Examples directly quoted in Ref. [97] include the quadratic assignment problem, the knapsack problem, the maximum independent set problem, low autocorrelation binary sequences, sports timetabling problems, and spin glasses. The fact that these problems become hard at relatively small sizes enables a reliable scaling analysis of runtimes. Recently, a library featuring instances belonging to these classes have been published [122].

4. Possible genomics applications of very hard optimization problems

While assessing whether quantum advantage can be established for these classes of problems is interesting from a fundamental perspective, it is important to determine whether there are connections to problems that actually occur in genomics. In what follows, we survey known instances where such problems have been studied in the context of genomics, to the best of our knowledge, and point out possible connections in cases where dedicated literature is not yet available.

Quadratic assignment problem (QAP). The mathematical formulation of a QAP typically involves minimizing the sum over all pairs of nodes of the product of the "flow" between them [123]. The QAP does not have an approximation algorithm running in polynomial time, making it a good candidate for quantum advantage. Since QAP models problems where the cost of an assignment is determined by the relationships between pairs of entities, they seem relevant for genomic problems. However, the use of QAP in genomics seems so far to be limited to enhancing visualization of microarray data based on gene expression similarity [124,125], potentially revealing intricate relationships in gene expression data that

might not be readily apparent through more localized clustering methods.

Knapsack problem. The Knapsack problem is a well-known optimization problem that involves selecting a subset of items, each characterized by a weight and a value, such that the total weight of the chosen items does not exceed a given capacity, while simultaneously maximizing the total value of the selected items. We attempt to draw some connections to gene selection, which is a critical task involving the identification of a subset of genes that are most pertinent to a particular biological inquiry [126,127]. This includes identifying genes associated with a specific disease, genes that play a key role in a biological pathway, or genes that can effectively classify different biological samples.

The weight associated with selecting a gene could reflect experimental costs, technical difficulty, or a penalty for redundancy, such as including genes that are highly correlated with others in the set, aiming to promote diversity. The capacity in this context may be defined by constraints such as a fixed panel size, budget caps, or sequencing coverage, all of which impose a limit on how many and which genes can be included.

Maximum independent set (MIS). An independent set in a graph is defined as a collection of vertices wherein no two vertices are adjacent. The MIS problem seeks to identify such a set with the largest possible number of vertices. The structure of the MIS problem, which centers on the selection of noninteracting elements, shares similarities with several challenges within genomics. Many problems in genomics involve the identification of sets of entities, such as genetic markers that are mutually incompatible or nonoverlapping in some sense. Interestingly, MIS problems can be mapped to sparse QUBO ones [97]. Some problems, such as identifying unrelated individuals for genetic analysis, have been recast as a MIS [128,129]. This is of particular importance in GWAS, where a fundamental assumption for many statistical tests is the independence of the samples being analyzed. However, human populations often exhibit complex patterns of relatedness, which, if not accounted for, can lead to spurious associations between genetic variants and the traits or diseases under investigation. The MIS problem provides an elegant solution to identify the largest possible subset of individuals within a dataset who are statistically considered "unrelated." In this application, the first step involves calculating the pairwise relationship between all individuals in the study, based on SNPs. A graph is constructed where each individual is represented as a vertex. An edge is drawn between two individuals if their estimated relatedness exceeds a predefined threshold, indicating that they are considered "related" (based on a threshold) for the purposes of the study. Then the problem of identifying the largest set of unrelated individuals becomes equivalent to finding the MIS in this graph.

Further, MIS has also been proposed in gene selection [130].

Finally, the problem of identifying nonrepetitive promoters has been recast as an MIS problem in Ref. [131].

D. Quantum enhanced Monte Carlo

Many genomics analyses (e.g., phylogenetic tree inference [132] and population genetics models [133]) rely heavily on Monte Carlo methods. Speeding these up with quantum algorithms could be transformative. A novel quantum algorithm capable of accelerating sampling was initially introduced by one of us in 2021 [134]. Later, Ref. [135] demonstrated a polynomial speed-up in the simulation of an all-to-all connected spin glass model, in real hardware.

Obviously, the potential to accelerate Monte Carlo simulations could have a significant impact on a wide range of genomics tasks involving statistical inference, such as those tackled today with hidden Markov chain. Moreover, this approach is directly relevant to the optimization of NP-hard problems, as it could enhance the performance of simulated annealing [136].

However, this method is still relatively new, and its strengths and limitations [137,138] have yet to be thoroughly evaluated before concrete applications in genomics can be expected. Finally, quantum-enhanced sampling can also be achieved using annealers. For instance, they have been successfully applied to efficient QUBO models for DNA packaging [139].

VI. QUANTUM MACHINE LEARNING

A. Classical machine learning in genomics

ML methods have already demonstrated significant impact in genomics, with several applications now considered state of the art and new ones continuing to emerge. Here we provide a short list of representative use cases of ML in genomics and refer the reader to recent comprehensive reviews for broader coverage [140–143].

In genomics, deep learning approaches have been used to improve the accuracy of genetic variant calling [144] and to advance variant effect prediction [145], extending beyond coding regions to the noncoding genome.

Within transcriptomics, and specifically in single cell RNA sequencing analysis, unsupervised and supervised ML-based tools, as well as large language models have become essential for clustering, batch effect identification and correction, and cell-type annotation [146,147].

Generative models are emerging as promising tools for multimodal omics data integration [148].

In addition to analytical tasks, with ML it is possible to address ethical and logistical challenges in genomics, particularly around data privacy. Federated learning enables institutions like hospitals and laboratories to collaboratively train models without exchanging sensitive patient genomic data. Instead of moving data to a central server, the model is sent to local sites for training, preserving privacy while supporting large-scale research [149].

B. Challenges of quantum machine learning

Given the importance of ML in genomics one could expect quantum machine learning (QML) [150–153] as a potential candidate to speed-up several workflows [22,154]. QML is a rapidly growing field, and it is outside the scope of this work to provide an in-depth discussion of its status and latest achievements. However, to provide a coherent picture of the conceptual challenges of quantum algorithms in genomics, it is worth mentioning the general issues of QML.

First, it should be noted that, to date, no quantum speedup for ML on *real-world* datasets has been observed or even postulated. Any scalability advantages have only been demonstrated on artificial datasets [155,156]. Whether such an "inherently quantum-hard" structure exists in real-world, naturally occurring datasets (i.e., those for which we actually want to apply ML) is still an open question. Further, Ref. [156] methods are as demanding as performing Shor's algorithm.

Alternatively, an empirical approach might involve developing heuristic strategies that establish a practical advantage considering several datasets. However, in QML, there is an additional challenge. Quantum neural networks (QNNs), or similar methods based on optimizable architectures, are significantly harder to train than their classical counterparts. In current literature, this issue is often referred to as the *barren plateaus* problem, where the optimization landscape becomes exponentially difficult to optimize as system size increases [65,157]. Much of the current research aims to understand the occurrence of barren plateaus and identify ways to mitigate them [158,159]. Barren plateaus do not occur if circuits are sufficiently shallow [160] or highly structured [161], but this could also make these circuits easier to simulate classically, potentially negating any quantum advantage [162–164]. Overall, a quantum learning model cannot demonstrate a true quantum advantage if it lacks the necessary trainability, expressivity, or capacity for generalization that surpasses its classical equivalents.

Finally, data-loading can also be a bottleneck. In this context, there are generally two practical approaches: The first is data encoding via angles of parameterized rotation gates [155]. While this approach is widespread, the number of qubits generally scales with the dataset size (though this is not strictly fixed). For example, for image data with millions of real-valued variables, one would need a similarly large circuit to encode all the information.

The second is amplitude encoding. This technique maps classical data into the amplitudes of a quantum state, offering an exponential reduction in memory, since M variables could be encoded using only $\log_2(M)$ qubits. However, preparing a quantum state that precisely represents the data generally requires an exponential number of operations. This is the same fundamental data-loading problem discussed earlier, for example, in the context of Grover's algorithm.

VII. CONCLUSION

We discussed several possible applications of quantum computing in the field of genomics, with a particular focus on the roadblocks and challenges that were typically neglected in previous literature. Note that we focus exclusively on the high-level theoretical challenges associated with the ideal execution of the proposed quantum algorithms in the context of genomics. Additional low-level issues arising from hardware noise or problem-specific implementations are beyond the scope of this discussion.

We began by discussing the conceptual problems related to the use of Grover's algorithm for database search. We showed that the hoped-for, and usually claimed, quadratic advantage vanishes under the realistic assumption of the lack of a QRAM. We also discussed, for the first time in this context, some approximate workarounds, but without any guarantee of accuracy for this important task.

Hopes for quantum advantage are connected with the solution of problems that are harder than linear in the dataset size [41,72]. For this reason, we turned our attention to genomics problems that contain combinatorial optimization tasks. These are problems that require exponential time, with respect to the input size, to be solved exactly. However, not all combinatorial optimization problems that are hard in theory are equally difficult in practice [97]. We discuss how standard complexity theory may be misleading when guiding us in real-world cases, as sometimes polynomial-time approximation methods are effective in practice even for NP-hard problems.

We suggest that problems amenable to quantum speedup need to be carefully identified *a priori*. In particular, they should correspond mathematically to models that are considered good benchmarks for quantum optimization, because (i) they become hard at much smaller sizes, and (ii) conventional approximate methods perform comparatively worse [97,122]. Additionally (iii), they should be mapped into a QUBO problem with minimal overhead.

Among these, we identify the following optimization problems: max-cut, maximum independent set, knapsack, and quadratic assignment problems to be relevant for (or possibly connected to) genomics. It would also be interesting to discover if some genomics tasks admit a formulation as low autocorrelation binary sequence problems, given that there is some experimental evidence of a scaling advantage using QAOA [165]. Examples of application of the LABS problem in genomics may include gene selection for cancer subtype classification where binary feature selection can benefit from low autocorrelation to reduce redundancy, and guide RNA (gRNA) design, where binary encoding of candidate sequences could be optimized to minimize off-target effects.

Finally, we discuss QML and its trainability and data-loading problems.

So far, there is no evidence of quantum advantage in ML beyond artificially constructed datasets (cf. the recent and very instructive Ref. [166]), although some works show that graph kernels can be successfully used in quantum support vector machines [74,167,168].

In conclusion, the goal of this manuscript is to clearly discuss the theoretical challenges of commonly proposed quantum computing solutions for genomics. We are neither overly pessimistic nor optimistic about the synergies between quantum information and genomics. However, it is time for this field to adopt a mature stance and embrace a balanced perspective on this potentially transformative computational platform. We hope that our work can stimulate new directions and be helpful to researchers in carefully planning their numerical experiments.

ACKNOWLEDGMENTS

We acknowledge discussions with Francesco Tacchino, Zoe Holmes, and Massimiliano Incudini. We acknowledge useful feedback on an earlier version of the manuscript from Cristian Micheletti, Frederik Flöther, and Gemma Solomon. G.M. acknowledges financial support from the Swiss National Science Foundation (Grant No. PCEFP2_203455).

DATA AVAILABILITY

No data were created or analyzed in this study.

APPENDIX A: LIST OF ACRONYMS

In this Appendix, we provide a table containing definitions of the abbreviations most commonly used in the main text.

TABLE I. Acronyms/abbreviations with definitions.

Acronym	Full form	Definition
DNA	Deoxyribonucleic acid	The hereditary molecule that carries genetic information essential for an organism to develop, function, survive, and reproduce.
RNA	Ribonucleic acid	A molecule that plays a central role in the regulation and expression of genetic information, often serving as a messenger between DNA and proteins.
BLAST	Basic local alignment search tool	A widely used bioinformatics program for comparing biological sequences (DNA, RNA, protein). It identifies regions of local similarity between sequences, enabling fast database searches and approximate sequence alignments.
SNP	Single-nucleotide polymorphism	A genetic variant at one nucleotide position in DNA; one of the key types of variation tested in GWAS.
FPGA	Field-programmable gate array	A type of reconfigurable integrated circuit that can be programmed after manufacturing. FPGAs are often used for accelerating computationally intensive tasks, including bioinformatics and genomics workloads.
GWAS	Genome-wide association studies	Studies investigating associations between genetic variants across many genomes and traits/diseases; typically large-scale statistical tests.
LABS	Low autocorrelation binary sequences	Problem class involving binary sequences with low autocorrelation.
MIS	Maximum independent set	In a graph, an independent set is a set of vertices no two of which are adjacent. The MIS problem asks for the largest such set. Used in genomics to select mutually non-related or non-overlapping entities, e.g., unrelated individuals in GWAS or nonredundant gene sets.
MSA	Multiple sequence alignment	Aligning more than two sequences simultaneously; known to be NP-hard in its exact form.
NPO	Nondeterministic polynomial-time optimization (problems)	A complexity class generalizing NP to optimization problems. NPO-hard problems are at least as hard as the hardest NP problems, and many genomics optimization tasks (e.g., multiple sequence alignment) belong to this class.
OLC	Overlap-layout-consensus	A method for genome assembly: (i) find overlaps between sequencing reads; (ii) arrange them in a layout graph; (iii) derive consensus sequence. Used in contexts of *de novo* assembly.
PTAS	Polynomial-time approximation scheme	A class of algorithms: for any $\epsilon > 0$, an algorithm in PTAS produces a solution within $(1 + \epsilon)$ of optimal in polynomial time.
PWA	Pairwise (sequence) alignment	The problem of aligning two sequences, e.g., two DNA or protein sequences; solvable in polynomial time with dynamic programming.
QA	Quantum annealing	An analog quantum optimization method that uses quantum resources to find low-energy states of a Hamiltonian.
QAOA	Quantum approximate optimization algorithm	A heuristic variational quantum algorithm for solving combinatorial optimization problems (often QUBO), balancing circuit depth and solution quality.
QAP	Quadratic assignment problem	A combinatorial optimization problem where objects are assigned to locations and cost depends on pairwise interactions.
QML	Quantum machine learning	Use of quantum algorithms or quantum-enhanced methods in machine learning tasks.
QRAM	Quantum random access memory	Hypothetical memory structure that allows classical data to be loaded into quantum superposition efficiently.
QUBO	Quadratic unconstrained binary optimization	An optimization problem over binary variables with a quadratic cost function (pairwise interaction terms), with no hard constraints explicitly; many genomic optimization tasks are mapped onto QUBO to use quantum annealing or related algorithms.
TSP	Traveling salesman problem	Classical NP-hard optimization problem: find shortest cyclic route visiting each node once. Appears as a benchmark/mapped problem in the paper.
TTS	Time to solution	A performance metric: the total time required (including repeated runs, etc.) to reach a desired success probability in a optimization method.
VQE	Variational quantum eigensolver	Another variational algorithm used on quantum hardware for finding eigenvalues/eigenvectors of Hamiltonians; mentioned in the context of optimization/genomics mappings.

APPENDIX B: MYTHS ON QUANTUM COMPUTING

In this Appendix, we aim to dispel some frequently made claims regarding the power of quantum computing. Some of these points may be obvious, while others may be less so. We believe this discussion may be particularly useful for nonexpert readers.

1. Claim: Quantum gates are faster than classical gates

Quantum digital gates cannot operate faster than the classical electronics that controls them [169]. Moreover, in the fault-tolerant quantum era, each logical qubit will be encoded in several physical qubits, and each logical gate will require multiple elementary operations on the individual physical qubits. As a result, the effective quantum clock frequency will be orders of magnitude lower than that of conventional classical processors [33,40].

2. Claim: A single qubit can store an arbitrarily large amount of information

This claim stems from the fact that a qubit state is defined by its coefficients, α and β, which are complex numbers. In principle, the binary expansion of either coefficient could contain an infinite number of digits, suggesting infinite information content.

However, the same argument applies to any classical analog method of storing information. For instance, one could imagine encoding a telephone number by cutting a string to a precise length such that its measurement in millimeters reproduces the exact decimal expansion of the number. Clearly, this is impractical, as such an encoding would be highly susceptible to uncontrollable errors in both preparation and readout. This is precisely why digital computing replaced analog machines.

In the case of qubits, the only possible measurement outcomes are binary, either "0" or "1." Extracting information about the amplitudes α and β with arbitrary precision requires performing many measurements in different bases. Therefore, it is impossible to retrieve the exact values of the coefficients in a single shot.

3. Claim: Quantum algorithms solve a problem by trying all possible inputs in superposition

This claim, once again, neglects the fundamental issue of how to extract information from a quantum state through measurements.

It is easy to prepare a quantum state defined on n qubits (all initialized to zero) that contains an equal superposition of all 2^n basis states using just one layer of Hadamard gates:

$$\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle. \tag{B1}$$

Suppose that all these binary strings are valid inputs to our problem. The task is to find the unique bitstring that satisfies a given set of logical propositions. In simpler terms, we assume that one string is the solution (which we need to find) to an optimization problem.

We imagine having access to a quantum function (oracle) that requires an ancillary qubit. The ancillary qubit, initially prepared in the state $|0\rangle$, flips to $|1\rangle$ only if the input x is the solution to our problem. The process reads as

$$\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle|0\rangle \rightarrow \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle|f(x)\rangle, \tag{B2}$$

where $f(x)=1$ if x is the solution x_{sol}, and $f(x)=0$ otherwise.

We would be tempted to claim victory by asking to find the n-qubit register associated with the value of 1 in the ancilla qubit, $|x_{\mathrm{sol}}\rangle|1\rangle$. However, this is not possible, and at this stage, the only way to extract the information is to measure the register. Given that all components have the same amplitude, we would obtain a random bit string readout, and the "wanted" readout would appear only with a vanishing probability, $1/\sqrt{2^n}$.

At this point, we would have the same efficiency as sampling random input states and checking whether they are solutions to our problem. The key step to make this rough idea work is to find a way to amplify the wanted component in such a way as to enhance its probability of being read out. Therefore, the claim overly trivializes the behavior of quantum algorithms (see main text).

4. Claim: Quantum advantage is directly related to entanglement

One may be tempted to claim that any operation producing a sufficiently large entangled state results in a quantum state that is not amenable to classical simulation, and thus satisfies a necessary condition for quantum advantage. However, if a circuit consists only of quantum gates from the Clifford set, it can be efficiently simulated classically in polynomial time. Here, by simulation, we mean that it is possible to sample from the distribution of measurement outcomes [170]. A current area of research is the theoretical investigation of how many non-Clifford gates—sometimes referred to as magic—are necessary to produce states that are strictly nonclassically simulatable [171–173]. Given that Clifford operations can produce entangled states (for instance, a GHZ state can be prepared using only Hadamard and CNOT gates starting from the "00..0" state) entanglement alone is not a necessary and sufficient condition for quantum advantage.

APPENDIX C: COMPUTING IN QUANTUM SUPERPOSITION

1. Qubits

The elementary logic unit of quantum computing is the qubit, which, unlike a conventional bit, can exist in a superposition of both basis states $|0\rangle$ and $|1\rangle$ [3]. Mathematically, these basis states form an orthonormal basis for the complex vector space $\mathbb{C}^2$, such that a general qubit state, or wave function, reads $|\psi\rangle = c_0|0\rangle + c_1|1\rangle$. The complex coefficients, c_0 and c_1 associated with these states must satisfy the normalization condition, $|c_0|^2 + |c_1|^2 = 1$ which derives from the laws of quantum mechanics. These coefficients, squared, represent

the probability of measuring the qubit in either the state "0" or "1," therefore their sum, squared, must equal to one.

Quantum computing with a single qubit involves applying a unitary transformation, U to the initially prepared state to produce a final state, $|\psi\rangle \to U|\psi\rangle = |\psi'\rangle = c'_0|0\rangle + c'_1|1\rangle$. The computation must conclude with a measurement process that collapses the wave function onto one of the basis states. For instance, now the wave function will collapse on the state '0' with probability $|c'_0|^2$ and on "1" with probability $|c'_1|^2$.

Single-qubit gates are mathematically defined as unitary transformations in the $\mathbb{C}^2$ space, and in practice they are represented by 2×2 matrices. Two-qubits gates are represented by a $2^2 \times 2^2$ unitary matrices, while three-qubit gates by $2^3 \times 2^3$ objects, and so on. Each hardware implementation has different capabilities regarding specific gate sets, but canonical gate sets are commonly adopted in quantum information textbooks [3]. In this Appendix we will use for illustrative purposes just a few gates: the NOT gate, X, which acts on the basis states as $\mathsf{X}|0\rangle = |1\rangle$ and $\mathsf{X}|1\rangle = |0\rangle$, the gate, Z, which acts on the basis states as $\mathsf{Z}|0\rangle = |0\rangle$ and $\mathsf{Z}|1\rangle = -|1\rangle$, and their controlled versions, the CNOT (controlled-X) gate and the C-Z (controlled-Z) gate. See Appendix D for a formal definition of all the gates used in this section, and their matrix representation. Due to the linearity of quantum mechanics, it is sufficient to define the action of the gates on the basis vectors, their action on the general state reads, for instance, $\mathsf{Z}|\psi\rangle = c_0|0\rangle - c_1|1\rangle$. For universal computation, the unitary transformations in the gate set must approximate any other unitary transformation arbitrarily well. If these conditions are met, quantum circuits can be constructed, with the assurance that the unitary operations can be executed with a controllable error. However, creating an arbitrary multiqubits unitary operation is not guaranteed to be efficient, meaning it may scale exponentially with the number of qubits, n. We will return to this important point later.

2. Quantum circuits and superposition

To achieve practical computation, one must work with more than one qubit. In this case, the total computational space increases exponentially with the number of qubits, leading to the popular statement that a quantum computer can achieve exponential memory compression. For example, the basis states of a $n = 3$ qubit computational space are: $|000\rangle$, $|001\rangle$, $|010\rangle$, $\cdots$, $|111\rangle$, such that a general three qubit state is defined via $2^3 = 8$ complex coefficients, and reads $|\psi\rangle = c_{000}|000\rangle + c_{001}|001\rangle + c_{010}|010\rangle + \cdots + c_{111}|111\rangle$, where the sum of the coefficients squared, $|c_j|^2$, is one.

In reality, the power of quantum computation lies not just in memory compression but in the ability to apply a function to each component of the wave function simultaneously, due to superposition. Classically, to determine the action of a function U on each possible bitstring, one would need to apply that 2^n times, i.e., once for each bitstring.

In the quantum setup, the function U, which operationally is represented by a particular gate or sequence of gates, is applied to all 2^n components of the wave function at once, thanks to the linearity of quantum mechanics. For instance, let us consider the simplest function: flipping the leftmost bit in our previous 3-qubit example. To accomplish this, one would apply the X gate to the first qubit while leaving the other qubits unchanged. Mathematically, since the full unitary operation is defined over the three-qubit space, U would be the tensor product of the X gate acting on the first qubit and identity, I, operations on the second and third qubits, namely $U = \mathsf{I} \otimes \mathsf{I} \otimes \mathsf{X}$. Assuming, little-endian convention about bit-ordering, $U|000\rangle = |100\rangle$ and so on. The action of U on the state defined above produces the new state $|\psi'\rangle = c_{000}|100\rangle + c_{001}|101\rangle + c_{010}|110\rangle + \cdots + c_{111}|011\rangle$. For instance, the new coefficient multypling the state $|000\rangle$ is the one previously "attached" to $|100\rangle$, so $c'_{000} = c_{100}$. The corresponding, simple, circuit is shown in Fig. 5(a).

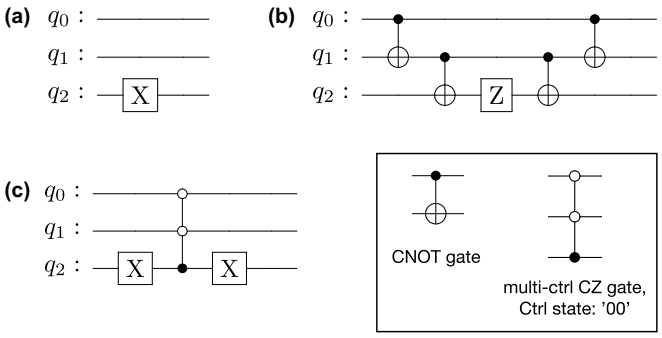

FIG. 5. Example of circuits representing the unitary functions described in the main text. (a) A circuit that flips the value of the leftmost bit; (b) a circuit that changes the sign of the amplitudes of the components with an odd number of "1"; (c) a circuit that changes the sign of the amplitude of the component corresponding to the "000" state. The multiqubit gates used are defined in Appendix D.

As a second example, we consider the following "function": detecting whether a bitstring has an even or odd number of "1"s, and flipping the sign of the corresponding coefficients in the latter case. A quantum circuit that corresponds to this *parity check* unitary is depicted in Fig. 5(b). This unitary acts on the superposition of bitstrings, simultaneously flipping the signs of all the relevant coefficients. This unitary yields the output state $U|\psi\rangle = |\psi'\rangle = c_{000}|000\rangle - c_{001}|001\rangle - c_{010}|010\rangle + c_{011}|011\rangle + \cdots - c_{111}|111\rangle$ A final example is a unitary that selectively flips the sign of the coefficient for only a specific target bitstring, such as the $|000\rangle$ state. The corresponding circuit is shown in Fig. 5(c), and the resulting state would be $U|\psi\rangle = |\psi'\rangle = -c_{000}|000\rangle + c_{001}|001\rangle + c_{010}|010\rangle + \cdots + c_{111}|111\rangle$.

At this point, some readers may raise the concern that these examples seem not useful at all, while others might already glimpse a connection with the task of database search. Both perspectives are valid. Indeed, some of these functions are useless in isolation, but they serve as building blocks for more complex and useful algorithms. For instance, the third example can be understood as an oracle that *marks* the "000" string. Oracles are essential components of many algorithms, such as Grover's algorithm [3,24,174]. They act as black boxes, performing specific operations.

In Grover's algorithm, each oracle application can be viewed as a database query. The algorithm's asymptotic speed-up is measured by the number of times the database must be consulted to find the desired entry. Therefore, the

quadratic speed-up of Grover's algorithm, often cited as a quantum advantage for database search, pertains to the number of oracle queries rather than the total number of elementary operations. This concept is crucial and will be explored in greater detail later.

Returning to our last two examples [Figs. 5(b) and 5(c)], marking a state by changing the sign of its amplitudes would not result in an effective search algorithm, as it does not alter the measurement outcome in the computational basis. Recall that the amplitude squared represents the probability of measuring a specific bit string.

The common theme of successful quantum algorithms is to perform operations in such a way that the target bit string—whether it represents the data to be found or the solution to an optimization problem—is measured with high probability. In Appendix E we see that the third function defined above is actually Grover's oracle in the particular case of "marking" a database entry encoded with the "00" bitstring, but an additional unitary is required to amplify its amplitude.

APPENDIX D: QUANTUM GATES

This Appendix provides the definitions of the quantum gates used in this work. The matrix representations are given in the computational basis $\{|0\rangle, |1\rangle\}$ for single-qubit gates and their tensor product extensions for multiqubit gates.

The Pauli-X gate is a single-qubit gate represented by the following 2×2 matrix:

$$\mathsf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Its action on the computational basis states is

$$\mathsf{X}|0\rangle = |1\rangle, \quad \mathsf{X}|1\rangle = |0\rangle.$$

The Pauli-Z gate is a single-qubit gate represented by the following 2×2 matrix:

$$\mathsf{Z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Its action on the computational basis states is

$$\mathsf{Z}|0\rangle = |0\rangle, \quad \mathsf{Z}|1\rangle = -|1\rangle.$$

The Hadamard gate is a single-qubit gate represented by the following 2×2 matrix:

$$\mathsf{H} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Its action on the computational basis states is

$$\mathsf{H}|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle),$$
$$\mathsf{H}|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

The Controlled-NOT (CNOT) gate is a two-qubit gate. The first qubit is the control qubit, and the second qubit is the target qubit. The CNOT gate flips the state of the target qubit if and only if the control qubit is in the state $|1\rangle$. Its 4×4 matrix representation in the basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ is

$$\mathsf{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Its action on the computational basis states is

$$\mathsf{CNOT}|00\rangle = |00\rangle,$$
$$\mathsf{CNOT}|01\rangle = |01\rangle,$$
$$\mathsf{CNOT}|10\rangle = |11\rangle,$$
$$\mathsf{CNOT}|11\rangle = |10\rangle.$$

The multicontrolled-CZ gate with 3 qubits, where the first two qubits are the control qubits and the third is the target qubit. In this case we define the control string to be "00." This applies a Pauli-Z gate to the target qubit only when the first two control qubits are in the state $|00\rangle$. The 8×8 matrix representation in the basis $\{|000\rangle, |001\rangle, |010\rangle, |011\rangle, |100\rangle, |101\rangle, |110\rangle, |111\rangle\}$ is

$$\mathsf{CZ}_{00}^{(1,2|3)} = \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Its action on the computational basis states is

$$\mathsf{CZ}_{00}^{(1,2|3)}|000\rangle = -|000\rangle,$$
$$\mathsf{CZ}_{00}^{(1,2|3)}|001\rangle = -|001\rangle,$$
$$\mathsf{CZ}_{00}^{(1,2|3)}|010\rangle = |010\rangle,$$
$$\mathsf{CZ}_{00}^{(1,2|3)}|011\rangle = |011\rangle,$$
$$\mathsf{CZ}_{00}^{(1,2|3)}|100\rangle = |100\rangle,$$
$$\mathsf{CZ}_{00}^{(1,2|3)}|101\rangle = |101\rangle,$$
$$\mathsf{CZ}_{00}^{(1,2|3)}|110\rangle = |110\rangle,$$
$$\mathsf{CZ}_{00}^{(1,2|3)}|111\rangle = |111\rangle.$$

In general, for a multicontrolled gate with control qubits $c_1, c_2, \ldots, c_n$ and a target qubit t, conditioned on a control string $s = s_1 s_2 \ldots s_n$, the target qubit is acted upon by a single-qubit gate U if and only if the control qubits are in the state $|s_1\rangle|s_2\rangle \ldots |s_n\rangle$. For the controlled-CZ gate with control string "00," the single-qubit gate U is the Z gate.

APPENDIX E: GROVER ALGORITHM FOR A TOY DATABASE SEARCH WITHOUT QRAM

Suppose we have the following list of elements, $\{T, A, T, G\}$, and we want to find the position of the element A using Grover's algorithm. To encode the data, we need two qubits (see main text), but we also require an additional qubit register to store the indices $\{0, 1, 2, 3\}$ of the elements. Therefore, we use four qubits: two for encoding the indices

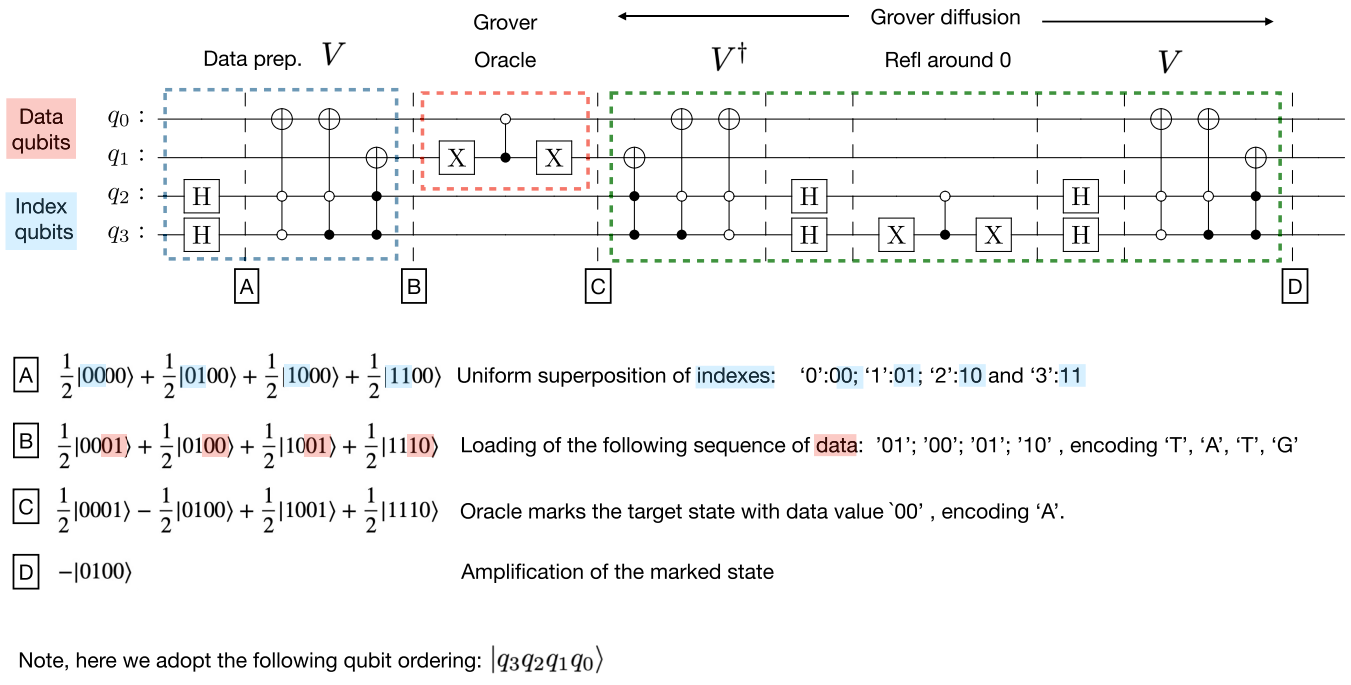

FIG. 6. Grover's algorithm with index and data encoding.

and two for encoding the data. A generic quantum state is the tensor product of the two registers $|q_3 q_2\rangle \otimes |q_1 q_0\rangle = |q_3 q_2 q_1 q_0\rangle$

Most textbook examples of Grover's algorithm initialize the list using a uniform superposition of strings. However, in this case, the process is more nuanced because we need to encode both the indices and the data together, meaning the data must be tied to the corresponding indices. Our goal is to initialize the quantum state in such a way as to encode the full input $\{(0, T), (1, A), (2, T), (3, G)\}$.

The first step is to create a uniform superposition of indices in the bottom register (see Fig. 6) $1/2(|00\rangle + |01\rangle + |10\rangle + |11\rangle) \otimes |00\rangle$. Then we suitably entangle each data entry with its corresponding index: $1/2(|00\rangle \otimes |'T'\rangle + |01\rangle \otimes |'A'\rangle + |10\rangle \otimes |'T'\rangle + |11\rangle \otimes |'G'\rangle)$, where we used the data values A, T, G, with a bit of abuse of notation, to better distinguish the two conceptually different bit positions. This is achieved through the state preparation unitary V, which initializes the entire qubit register from the initial $|0000\rangle$ state. Notice that in Fig. 6 the bottom register (qubits $q3, q2$) encodes the index, while the top register (qubits $q1, q0$) encodes the data in binary format. We adopt the little-endian convention such that the bottom qubit are the leftmost in the bitstring representation.

In this example, we use a sequence of multicontrolled CNOT gates. For instance, to entangle the "T" entry with the first index position "00," we place a multicontrolled CNOT gate acting on the first data qubit, q_0, controlled by the index string "00." Similarly, to entangle the last entry "G" with the position "11," we use a multicontrolled CNOT gate acting on the second data qubit, controlled by the index string "11." To entangle the data entry "A" with the index "01" we simply do nothing because our encoding assigns the "00" value to A, so no bit has to be flipped. Multicontrolled gates can be compiled using CNOT gates [3]. In Fig. 6 we avoid doing for the sake of visualization. Notice that here we use a purely quantum mechanical feature of controlling the execution of a function, in superposition.

For all practical purposes, this Appendix could conclude here. The data loading unitary V necessarily contains order of N gates, given that, for each index, we had to design a suitable multicontrolled CNOT gate to entangle the corresponding data [175].

If the data sequence is pseudo-random, lacks any regular periodicity, or cannot be mathematically modeled, each entry must be loaded individually using an order of N operations. This means that the data loading phase already negates the quadratic speed-up.

For the sake of example, we proceed further with the Grover circuit. The Grover operator is divided into two steps that need to be iterated. The crucial point is that the number of iterations required to amplify the amplitude of the desired state |index, data⟩ is $\sqrt{N}$ rather than N.

The first operation is usually called "reflection" around the marked state' and is typically performed by an oracle in textbooks. In our specific implementation, since we encode both the index and the data, we can explicitly provide a circuit for this oracle. The marked state is contained in the "data register," and in our encoding, it corresponds to the "00" state. This information is known to us *a priori*. Therefore, the circuit implements the function discussed in the main text [see Fig. 5(c)], i.e., a unitary that flips the sign of the components with "00" in the data register.

The second operator is called diffusion or reflection around the initial state. If the initial state is simply a uniform superposition of all four qubit states, the circuit is straightforward. However, in this case, it is not, since we need to use the reverse unitary, $V^\dagger$, that prepared the entangled state, apply a reflection around "00" in the index register, and then apply V again.

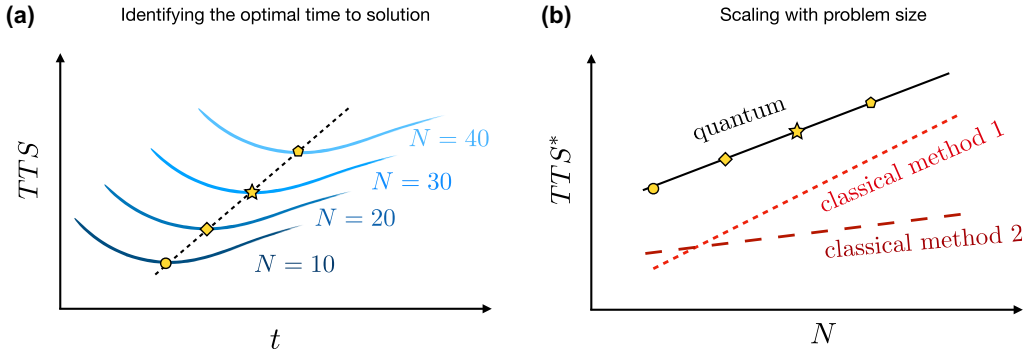

FIG. 7. Sketch of an empirical assessment of scaling advantage: (a) Identifying the optimal time to solution, TTS^* as a function of the time t of the run. For each problem size N (in this hypothetical scenario $N = 10, 20, 30, 40$) this represents the minimal cost of the simulation. (b) Scaling of the TTS^* as a function of N. This plot is used to assess the existence of an empirical scaling advantage over classical methods. In this sketch, the quantum algorithms would outperform the first classical method, but not the second.

In this simple example, the *a priori* success probability of finding the correct entry is 25%, given that there are just four elements. This is a limiting case in which the Grover algorithm returns the correct element in just one iteration. The final state produced by the circuit in Fig. 6 is indeed $|0100\rangle$, that encodes the value ("1" = 01, "A" = 00).

The total runtime of this implementation would scale as $N\sqrt{N}$, because every Grover step also requires a depth N subcircuit, thus realizing a quantum slowing-down compared to the brute force classical approach.

APPENDIX F: HEURISTIC METHODS FOR QUANTUM OPTIMIZATION

As of today, quantum optimization demonstrations feature two main strategies: quantum annealing (QA) [37,176], implemented on analog quantum hardware, and the quantum approximate optimization algorithm (QAOA) [177,178] or similar approaches based on parameterized circuits, like the variational quantum eigensolver (VQE) [179], executed on digital hardware. We refer the reader to reviews of these algorithms for definitions and details [37,38,180,181].

However, it must be noted that all these approaches are heuristic in practice. Quantum annealing yields the exact solution only in the infinite runtime limit. Moreover, analog machines are strongly affected by decoherence, allowing coherent dynamics to persist only for tens of nanoseconds, see, e.g., Ref. [26]. Current quantum annealing experiments extend beyond tens of microsecond runtimes, leading to strong coupling between the system and the environment [37,112,182]. As a result, the computational resources of these machines involve a combination of incoherent tunneling events and conventional thermal annealing [114,115].

Quantum optimization on today's digital hardware is similarly affected by gate noise, as these machines are not yet fault-tolerant [178,183–185]. QAOA can be interpreted as a discrete version of quantum annealing, where the total evolution is divided into layers or blocks, each associated with tunable parameters [177].

Recent large-scale simulations of QAOA reveal that coherent dynamics are maintained up to about 8–10 layers (for a simple disordered Ising model). Beyond that point, gate noise introduces errors that increase the cost function [184].

This means that the execution of both QA and QAOA, must be repeated multiple times in practice. In the literature, this is commonly discussed in terms of the *success probability*, p, that is, the probability of finding the exact solution or a solution whose energy cost is below a certain threshold.

Since each run occupies the quantum hardware for a physical time t, the total *time to solution* is $\mathrm{TTS} = R \times t$, where R is the number of repetitions required to achieve the desired success probability. Ideally, one would perform a single execution with $t \to \infty$. However, this is obviously impractical. The opposite limit consists of performing a huge number of repetitions, each with a very short execution time t, but this is also inefficient, since one would perform no annealing. It has been recognized that there exists a trade-off between the total annealing (or circuit) time t and the number of repetitions R required to achieve a target success probability p (for example, $p = 0.90$).

Determining the optimal balance between t and R is essential for accurately assessing the scaling of the total computational cost with the problem size [182]. Notably, the *optimal time to solution*, defined as $\mathrm{TTS}^* = \min(t\ R(t))$ depends on the size of the problem, hence $\mathrm{TTS}^* = \mathrm{TTS}^*(N)$. This is the proper quantity to plot against N to assess the scaling of the quantum time cost vs problem size. This procedure is shown graphically in Fig. 7. As clearly explained in the Refs. [112,182], it is important to identify the optimal time to solution when analyzing the scaling. Overestimating the runtime for smaller problem sizes would result in a flatter scaling curve, potentially leading to an artificial indication of quantum advantage.

While this concept is particularly transparent in the case of QA, the existence of an optimal resource usage also applies to QAOA and VQE. Here the total runtime t is proportional to the circuit depth *times* the number of shots needed to evaluate the cost function.

Historically, two-dimensional spin glasses (well-suited to the architecture of quantum annealers) have been studied extensively [112,115,186]. Reference [182], for example, identifies a class of artificial problems where quantum annealers demonstrate scaling advantages over simulated annealing (SA) [187]. Interestingly, Ref. [188] extends this observation to more realistic problems related to polymer folding on a lattice, showing a similar scaling advantage.

While these results are promising, they do not constitute conclusive evidence of quantum advantage. A definitive demonstration would require outperforming all possible classical solvers, e.g., simulated quantum annealing [189].

To conclude, seeking quantum advantage through empirical observation is certainly a legitimate and practical approach. However, careful attention must be paid to how the scaling with problem size is analyzed and to ensuring fair comparisons with the best-known classical methods for the same problem. Moreover, classical algorithms should be appropriately optimized for the task at hand. For example, if the optimization problem possesses a known structure, this prior knowledge should also be exploited by the classical algorithm. A notable case concerns Ref. [114], where an initially claimed scaling advantage of quantum annealing over SA was refuted by employing a slightly more sophisticated cluster-update SA algorithm [113].

[1] A. Alyass, M. Turcotte, and D. Meyre, From big data analysis to personalized medicine for all: Challenges and opportunities, BMC Med. Genomics **8**, 33 (2015).

[2] Y. Hasin, M. Seldin, and A. Lusis, Multi-omics approaches to disease, Genome Biol. **18**, 83 (2017).

[3] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*, Cambridge Series on Information and the Natural Sciences (Cambridge University Press, Cambridge, UK, 2000).

[4] F. Tacchino, A. Chiesa, S. Carretta, and D. Gerace, Quantum computers as universal quantum simulators: State-of-the-art and perspectives, Adv. Quantum Technol. **3**, 1900052 (2020).

[5] Y. Cao, J. Romero, J. P. Olson, M. Degroote, P. D. Johnson, M. Kieferová, I. D. Kivlichan, T. Menke, B. Peropadre, N. P. Sawaya *et al.*, Quantum chemistry in the age of quantum computing, Chem. Rev. **119**, 10856 (2019).

[6] A. Di Meglio, K. Jansen, I. Tavernelli, C. Alexandrou, S. Arunachalam, C. W. Bauer, K. Borras, S. Carrazza, A. Crippa, V. Croft *et al.*, Quantum computing for high-energy physics: State of the art and challenges, PRX Quantum **5**, 037001 (2024).

[7] R. Orús, S. Mugel, and E. Lizaso, Quantum computing for finance: Overview and prospects, Rev. Phys. **4**, 100028 (2019).

[8] A. Jayakumar, A. A. Adedoyin, J. J. Ambrosiano, P. M. Anisimov, A. Baertschi, W. R. Casper, G. Chennupati, C. J. Coffrin, H. N. Djidjev, D. O. Gunter *et al.*, Quantum algorithm implementations for beginners, ACM Trans. Quantum Comput. **3**, 1 (2022).

[9] C. Outeiral, M. Strahm, J. Shi, G. M. Morris, S. C. Benjamin, and C. M. Deane, The prospects of quantum computing in computational molecular biology, WIREs Comput. Mol. Sci. **11**, e1481 (2021).

[10] B. A. Cordier, N. P. Sawaya, G. G. Guerreschi, and S. K. McWeeney, Biology and medicine in the landscape of quantum advantages, J. R. Soc. Interface. **19**, 20220541 (2022).

[11] L. Marchetti, R. Nifosì, P. L. Martelli, E. Da Pozzo, V. Cappello, F. Banterle, M. L. Trincavelli, C. Martini, and M. D'Elia, Quantum computing algorithms: Getting closer to critical problems in computational biology, Brief. Bioinform. **23**, bbac437 (2022).

[12] K. Nalecz-Charkiewicz, K. Charkiewicz, and R. M. Nowak, Quantum computing in bioinformatics: A systematic review mapping, Brief. Bioinform. **25**, bbae391 (2024).

[13] F. F. Flöther, The state of quantum computing applications in health and medicine, Res. Direct.: Quant. Technol. **1**, e10 (2023).

[14] M. Zinner, F. Dahlhausen, P. Boehme, J. Ehlers, L. Bieske, and L. Fehring, Quantum computing's potential for drug discovery: Early stage industry dynamics, Drug Discovery Today **26**, 1680 (2021).

[15] P. S. Emani, J. Warrell, A. Anticevic, S. Bekiranov, M. Gandal, M. J. McConnell, G. Sapiro, A. Aspuru-Guzik, J. T. Baker, M. Bastiani *et al.*, Quantum computing at the frontiers of biological sciences, Nat. Methods **18**, 701 (2021).

[16] R. Ur Rasool, H. F. Ahmad, W. Rafique, A. Qayyum, J. Qadir, and Z. Anwar, Quantum computing for healthcare: A review, Future Internet **15**, 94 (2023).

[17] S. Maniscalco, E.-M. Borrelli, D. Cavalcanti, C. Foti, A. Glos, M. Goldsmith, S. Knecht, K. Korhonen, J. Malmi, A. Nykänen *et al.*, Quantum network medicine: Rethinking medicine with network science and quantum algorithms, arXiv:2206.12405.

[18] S. Basu, J. Born, A. Bose, S. Capponi, D. Chalkia, T. A. Chan, H. Doga, F. F. Flother, G. Getz, M. Goldsmith, T. Gujarati, A. Guzman-Saenz, D. Iliopoulos, G. O. Jones, S. Knecht, D. Madan, S. Maniscalco, N. Mariella, J. A. Morrone, K. Najafi *et al.*, Towards quantum-enabled cell-centric therapeutics, arXiv:2307.05734.

[19] S. McWeeney, T. Perciano, C. Susut, L. Chatterjee, M. Fornari, L. Biven, and C. Siwy, Quantum computing for biomedical computational and data sciences: A joint DOE-NIH roundtable, Technical Report, USDOE Office of Science, Washington, D.C. (2023).

[20] T. J. Durant, E. Knight, B. Nelson, S. Dudgeon, S. J. Lee, D. Walliman, H. P. Young, L. Ohno-Machado, and W. L. Schulz, A primer for quantum computing and its applications to healthcare and biomedical research, J. Am. Med. Inf. Assoc. **31**, 1774 (2024).

[21] A. M. Smaldone, Y. Shee, G. W. Kyro, C. Xu, N. P. Vu, R. Dutta, M. H. Farag, A. Galda, S. Kumar, E. Kyoseva *et al.*, Quantum machine learning in drug discovery: Applications in academia and pharmaceutical industries, Chem. Rev. **125**, 5436 (2025).

[22] F. F. Flöther, D. Blankenberg, M. Demidik, K. Jansen, R. Krishnakumar, R. Krishnakumar, N. Laanait, L. Parida, C. Y. Saab, and F. Utro, How quantum computing can enhance biomarker discovery, Patterns **6**, 101236 (2025).

[23] A. M. Dalzell, S. McArdle, M. Berta, P. Bienias, C.-F. Chen, A. Gilyén, C. T. Hann, M. J. Kastoryano, E. T. Khabiboulline, A. Kubica *et al.*, *Quantum Algorithms: A Survey of Applications and End-to-end Complexities* (Cambridge University Press, 2025).

[24] L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing* (ACM, New York, NY, 1996), pp. 212–219.

[25] S. Yarkoni, E. Raponi, T. Bäck, and S. Schmitt, Quantum annealing for industry applications: Introduction and review, Rep. Prog. Phys. **85**, 104001 (2022).
[26] A. D. King, S. Suzuki, J. Raymond, A. Zucca, T. Lanting, F. Altomare, A. J. Berkley, S. Ejtemaee, E. Hoskinson, S. Huang *et al.*, Coherent quantum annealing in a programmable 2,000 qubit Ising chain, Nat. Phys. **18**, 1324 (2022).
[27] A. D. King, A. Nocera, M. M. Rams, J. Dziarmaga, R. Wiersema, W. Bernoudy, J. Raymond, N. Kaushal, N. Heinsdorf, R. Harris *et al.*, Beyond-classical computation in quantum simulation, Science **388**, 199 (2025).
[28] Z. Cai, R. Babbush, S. C. Benjamin, S. Endo, W. J. Huggins, Y. Li, J. R. McClean, and T. E. O'Brien, Quantum error mitigation, Rev. Mod. Phys. **95**, 045005 (2023).
[29] Google Quantum AI, Suppressing quantum errors by scaling a surface code logical qubit, Nature (London) **614**, 676 (2023).
[30] Google Quantum AI and Collaborators, Quantum error correction below the surface code threshold, Nature (London) **638**, 920 (2025).
[31] S. Bravyi, A. W. Cross, J. M. Gambetta, D. Maslov, P. Rall, and T. J. Yoder, High-threshold and low-overhead fault-tolerant quantum memory, Nature (London) **627**, 778 (2024).
[32] T. J. Yoder, E. Schoute, P. Rall, E. Pritchett, J. M. Gambetta, A. W. Cross, M. Carroll, and M. E. Beverland, Tour de gross: A modular quantum computer based on bivariate bicycle codes, arXiv:2506.03094.
[33] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Surface codes: Towards practical large-scale quantum computation, Phys. Rev. A **86**, 032324 (2012).
[34] D. Litinski, A game of surface codes: Large-scale quantum computing with lattice surgery, Quantum **3**, 128 (2019).
[35] M. E. Beverland, P. Murali, M. Troyer, K. M. Svore, T. Hoeffler, V. Kliuchnikov, G. H. Low, M. Soeken, A. Sundaram, and A. Vaschillo, Assessing requirements to scale to practical quantum advantage, arXiv:2211.07629.
[36] J. Preskill, Quantum computing in the NISQ era and beyond, Quantum **2**, 79 (2018).
[37] A. Das and B. K. Chakrabarti, *Colloquium*: Quantum annealing and analog quantum computation, Rev. Mod. Phys. **80**, 1061 (2008).
[38] T. Albash and D. A. Lidar, Adiabatic quantum computation, Rev. Mod. Phys. **90**, 015002 (2018).
[39] A. Montanaro, Quantum algorithms: An overview, npj Quantum Inf. **2**, 15023 (2016).
[40] C. Gidney and A. G. Fowler, Efficient magic state factories with a catalyzed $|ccz>$ to $2|t>$ transformation, Quantum **3**, 135 (2019).
[41] R. Babbush, J. R. McClean, M. Newman, C. Gidney, S. Boixo, and H. Neven, Focus beyond quadratic speedups for error-corrected quantum advantage, PRX Quantum **2**, 010103 (2021).
[42] G. Mazzola, Quantum computing for chemistry and physics applications from a Monte Carlo perspective, J. Chem. Phys. **160**, 010901 (2024).
[43] V. Tam, N. Patel, M. Turcotte, Y. Bossé, G. Paré, and D. Meyre, Benefits and limitations of genome-wide association studies, Nat. Rev. Genet. **20**, 467 (2019).
[44] E. Uffelmann, Q. Q. Huang, N. S. Munung, J. De Vries, Y. Okada, A. R. Martin, H. C. Martin, T. Lappalainen, and D. Posthuma, Genome-wide association studies, Nat. Rev. Methods Primers **1**, 59 (2021).
[45] D. W. Bowden, S. S. An, N. D. Palmer, W. M. Brown, J. M. Norris, S. M. Haffner, G. A. Hawkins, X. Guo, J. I. Rotter, Y.-D. I. Chen *et al.*, Molecular basis of a linkage peak: Exome sequencing and family-based analysis identify a rare genetic variant in the adipoq gene in the IRAS family study, Human Mol. Genet. **19**, 4112 (2010).
[46] C. T. Consortium, The nature and identification of quantitative trait loci: A community's view, Nat. Rev. Genet. **4**, 911 (2003).
[47] X. Wang, D. Fan, Y. Yang, R. C. Gimple, and S. Zhou, Integrative multi-omics approaches to explore immune cell functions: Challenges and opportunities, IScience **26**, 106359 (2023).
[48] L. C. L. Hollenberg, Fast quantum search algorithms in protein sequence comparisons: Quantum bioinformatics, Phys. Rev. E **62**, 7532 (2000).
[49] K. Prousalis and N. Konofaos, A quantum pattern recognition method for improving pairwise sequence alignment, Sci. Rep. **9**, 7226 (2019).
[50] P. Niroula and Y. Nam, A quantum algorithm for string matching, npj Quantum Inf. **7**, 37 (2021).
[51] A. Sarkar, Z. Al-Ars, C. G. Almudever, and K. L. Bertels, QiBAM: Approximate sub-string index search on quantum accelerators applied to DNA read alignment, Electronics **10**, 2433 (2021).
[52] K. K. Soni and A. Rasool, Quantum-based exact pattern matching algorithms for biological sequences, ETRI J. **43**, 483 (2021).
[53] K. Miyamoto, N. Yamamoto, and Y. Sakakibara, Quantum algorithm for position weight matrix matching, IEEE Trans. Quantum Eng. **4**, 1 (2023).
[54] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum algorithm for linear systems of equations, Phys. Rev. Lett. **103**, 150502 (2009).
[55] V. Giovannetti, S. Lloyd, and L. Maccone, Quantum random access memory, Phys. Rev. Lett. **100**, 160501 (2008).
[56] S. Arunachalam, V. Gheorghiu, T. Jochym-O'Connor, M. Mosca, and P. V. Srinivasan, On the robustness of bucket brigade quantum RAM, New J. Phys. **17**, 123010 (2015).
[57] B. D. Clader, B. C. Jacobs, and C. R. Sprouse, Preconditioned quantum linear system algorithm, Phys. Rev. Lett. **110**, 250504 (2013).
[58] R. Babbush, C. Gidney, D. W. Berry, N. Wiebe, J. McClean, A. Paler, A. Fowler, and H. Neven, Encoding electronic spectra in quantum circuits with linear T complexity, Phys. Rev. X **8**, 041015 (2018).
[59] S. Jaques and A. G. Rattew, Qram: A survey and critique, arXiv:2305.10310.
[60] H. Tezuka, K. Nakaji, T. Satoh, and N. Yamamoto, Grover search revisited: Application to image pattern matching, Phys. Rev. A **105**, 032440 (2022).
[61] K. Nakaji, S. Uno, Y. Suzuki, R. Raymond, T. Onodera, T. Tanaka, H. Tezuka, N. Mitsuda, and N. Yamamoto, Approximate amplitude encoding in shallow parameterized quantum circuits and its application to financial market indicators, Phys. Rev. Res. **4**, 023136 (2022).
[62] National Center for Biotechnology Information (NCBI), Human genome overview—Genome reference consortium (2024), https://www.ncbi.nlm.nih.gov/grc/human.

[63] E. S. Lander, Initial sequencing and analysis of the human genome, Nature (London) **409**, 860 (2001).
[64] J. C. Venter, M. D. Adams, E. W. Myers, P. W. Li, R. J. Mural, G. G. Sutton, H. O. Smith, M. Yandell, C. A. Evans, R. A. Holt *et al.*, The sequence of the human genome, Science **291**, 1304 (2001).
[65] M. Larocca, S. Thanasilp, S. Wang, K. Sharma, J. Biamonte, P. J. Coles, L. Cincio, J. R. McClean, Z. Holmes, and M. Cerezo, Barren plateaus in variational quantum computing, Nat. Rev. Phys. **7**, 174 (2025).
[66] G. Scriva, N. Astrakhantsev, S. Pilati, and G. Mazzola, Challenges of variational quantum optimization with measurement shot noise, Phys. Rev. A **109**, 032408 (2024).
[67] F. M. Creevey, H. T. Hassan, J. McCafferty, L. C. Hollenberg, and S. Strelchuk, Scalable quantum state preparation for encoding genomic data with matrix product states, arXiv:2508.06184.
[68] R. Chikhi, A. Limasset, and P. Medvedev, Compacting de bruijn graphs from sequencing data quickly and in low memory, Bioinformatics **32**, i201 (2016).
[69] S. B. Needleman and C. D. Wunsch, A general method applicable to the search for similarities in the amino acid sequence of two proteins, J. Mol. Biol. **48**, 443 (1970).
[70] T. F. Smith, M. S. Waterman *et al.*, Identification of common molecular subsequences, J. Mol. Biol. **147**, 195 (1981).
[71] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, Basic local alignment search tool, J. Mol. Biol. **215**, 403 (1990).
[72] T. Hoefler, T. Häner, and M. Troyer, Disentangling hype from practicality: On realistically achieving quantum advantage, Commun. ACM **66**, 82 (2023).
[73] A. Daskin, A. Grama, and S. Kais, Multiple network alignment on quantum computers, Quantum Info. Proc. **13**, 2653 (2014).
[74] K. Kishi, T. Satoh, R. Raymond, N. Yamamoto, and Y. Sakakibara, Graph Kernels encoding features of all subgraphs by quantum superposition, IEEE J. Emerg. Select. Top. Circ. Syst. **12**, 602 (2022).
[75] R. M. Idury and M. S. Waterman, A new algorithm for DNA sequence assembly, J. Comput. Biol. **2**, 291 (1995).
[76] M. C. Schatz, A. L. Delcher, and S. L. Salzberg, Assembly of large genomes using second-generation sequencing, Genome Res. **20**, 1165 (2010).
[77] J. R. Miller, S. Koren, and G. Sutton, Assembly algorithms for next-generation sequencing data, Genomics **95**, 315 (2010).
[78] P. Flicek and E. Birney, Sense from sequence reads: Methods for alignment and assembly, Nat. Methods **6**, S6 (2009).
[79] H. Li, J. Ruan, and R. Durbin, Mapping short DNA sequencing reads and calling variants using mapping quality scores, Genome Res. **18**, 1851 (2008).
[80] Z. Iqbal, M. Caccamo, I. Turner, P. Flicek, and G. McVean, De novo assembly and genotyping of variants using colored de Bruijn graphs, Nat. Genet. **44**, 226 (2012).
[81] J. H. Moore, F. W. Asselbergs, and S. M. Williams, Bioinformatics challenges for genome-wide association studies, Bioinformatics **26**, 445 (2010).
[82] R. Schwartz and A. A. Schäffer, The evolution of tumour phylogenetics: Principles and practice, Nat. Rev. Genet. **18**, 213 (2017).
[83] B. Rannala and Z. Yang, Phylogenetic inference using whole genomes, Annu. Rev. Genom. Hum. Genet. **9**, 217 (2008).
[84] B. Chor and T. Tuller, Maximum likelihood of evolutionary trees: Hardness and approximation, Bioinformatics **21**, i97 (2005).
[85] M. D. Hendy and D. Penny, Branch and bound algorithms to determine minimal evolutionary trees, Math. Biosci. **59**, 277 (1982).
[86] R. P. Trees, The neighbor-joining method: A new method for reconstructing phylogenetic trees, Mol. Biol. Evol. **4**, 406 (1987).
[87] S. R. Browning and B. L. Browning, Haplotype phasing: Existing methods and new developments, Nat. Rev. Genet. **12**, 703 (2011).
[88] C. H. Ding, X. He, H. Zha, M. Gu, and H. D. Simon, A min-max cut algorithm for graph partitioning and data clustering, in *Proceedings of the 2001 IEEE International Conference on Data Mining* (IEEE, Piscataway, NJ, 2001), pp. 107–114.
[89] P. Festa, P. M. Pardalos, M. G. Resende, and C. C. Ribeiro, Randomized heuristics for the max-cut problem, Optimiz. Methods Softw. **17**, 1033 (2002).
[90] Y. Yu, L. Chen, X. Miao, and S. C. Li, Spechap: A diploid phasing algorithm based on spectral graph theory, Nucleic Acids Res. **49**, e114 (2021).
[91] S. R. Browning and B. L. Browning, Rapid and accurate haplotype phasing and missing-data inference for whole-genome association studies by use of localized haplotype clustering, The American Journal of Human Genetics **81**, 1084 (2007).
[92] A. Lucas, Ising formulations of many NP problems, Front. Phys. **2**, 5 (2014).
[93] G. E. Crooks, Performance of the quantum approximate optimization algorithm on the maximum cut problem, arXiv:1811.08419.
[94] G. G. Guerreschi and A. Y. Matsuura, Qaoa for max-cut requires hundreds of qubits for quantum speed-up, Sci. Rep. **9**, 6903 (2019).
[95] Z. Wang, S. Hadfield, Z. Jiang, and E. G. Rieffel, Quantum approximate optimization algorithm for maxcut: A fermionic view, Phys. Rev. A **97**, 022304 (2018).
[96] J. Wurtz and P. Love, Maxcut quantum approximate optimization algorithm performance guarantees for $p > 1$, Phys. Rev. A **103**, 042612 (2021).
[97] A. Abbas, A. Ambainis, B. Augustino, A. Bärtschi, H. Buhrman, C. Coffrin, G. Cortiana, V. Dunjko, D. J. Egger, B. G. Elmegreen *et al.*, Challenges and opportunities in quantum optimization, Nat. Rev. Phys. **6**, 718 (2024).
[98] S. P. Jordan, N. Shutty, M. Wootters, A. Zalcman, A. Schmidhuber, R. King, S. V. Isakov, and R. Babbush, Optimization by decoded quantum interferometry, Nature **646**, 831 (2025).
[99] E. M. Arkin and R. Hassin, Approximation algorithms for the geometric covering salesman problem, Discrete Applied Mathematics **55**, 197 (1994).
[100] Y. Matsumoto and S. Nakamura, Qualign: Solving sequence alignment based on quadratic unconstrained binary optimisation, EMBnet J. **28**, e1020 (2022).
[101] A. Sarkar, Z. Al-Ars, and K. Bertels, Quaser: Quantum accelerated de novo DNA sequence reconstruction, PLoS ONE **16**, e0249850 (2021).

[102] A. Boev, A. Rakitko, S. Usmanov, A. Kobzeva, I. Popov, V. Ilinsky, E. Kiktenko, and A. Fedorov, Genome assembly using quantum and quantum-inspired annealing, Sci. Rep. **11**, 13183 (2021).

[103] K. Nalecz-Charkiewicz and R. M. Nowak, Algorithm for dna sequence assembly by quantum annealing, BMC Bioinf. **23**, 122 (2022).

[104] Y. Chen, J.-H. Huang, Y. Sun, Y. Zhang, Y. Li, and X. Xu, Haplotype-resolved assembly of diploid and polyploid genomes using quantum computing, Cell Reports Methods **4**, 100754 (2024).

[105] M. J. Dinneen, P. S. Ghodla, and S. Linz, A qubo formulation for the tree containment problem, Theor. Comput. Sci. **940**, 60 (2023).

[106] Y. Park, J. Kim, and J. Huh, Scalable guide tree construction using quantum annealing for multiple sequence alignment, bioRxiv (2024).

[107] R. Y. Li, R. Di Felice, R. Rohs, and D. A. Lidar, Quantum annealing versus classical machine learning applied to a simplified computational biology problem, npj Quantum Inf. **4**, 14 (2018).

[108] M. Hoffmann, J. M. Poschenrieder, M. Incudini, S. Baier, A. Fritz, A. Maier, M. Hartung, C. Hoffmann, N. Trummer, K. Adamowicz *et al.*, Network medicine-based epistasis detection in complex diseases: Ready for quantum computing, Nucleic Acids Res. **52**, 10144 (2024).

[109] D. M. Fox, K. M. Branson, and R. C. Walker, mRNA codon optimization with quantum computers, PLoS ONE **16**, e0259101 (2021).

[110] C. S. Calude, M. J. Dinneen, and R. Hua, Quantum solutions for densest k-subgraph problems, Journal of Membrane Computing **2**, 26 (2020).

[111] V. Kumar, D. Alevras, M. Metkar, E. Welling, C. Cade, I. Niesen, T. Friedhoff, J.-E. Park, S. Shivpuje, M. LaDue *et al.*, Towards secondary structure prediction of longer mrna sequences using a quantum-centric optimization scheme, arXiv:2505.05782.

[112] T. F. Rønnow, Z. Wang, J. Job, S. Boixo, S. V. Isakov, D. Wecker, J. M. Martinis, D. A. Lidar, and M. Troyer, Defining and detecting quantum speedup, Science **345**, 420 (2014).

[113] S. Mandra, Z. Zhu, W. Wang, A. Perdomo-Ortiz, and H. G. Katzgraber, Strengths and weaknesses of weak-strong cluster problems: A detailed overview of state-of-the-art classical heuristics versus quantum approaches, Phys. Rev. A **94**, 022337 (2016).

[114] V. S. Denchev, S. Boixo, S. V. Isakov, N. Ding, R. Babbush, V. Smelyanskiy, J. Martinis, and H. Neven, What is the computational value of finite-range tunneling? Phys. Rev. X **6**, 031015 (2016).

[115] S. Boixo, T. F. Rønnow, S. V. Isakov, Z. Wang, D. Wecker, D. A. Lidar, J. M. Martinis, and M. Troyer, Evidence for quantum annealing with more than one hundred qubits, Nat. Phys. **10**, 218 (2014).

[116] E. Farhi, D. Gamarnik, and S. Gutmann, The quantum approximate optimization algorithm needs to see the whole graph: A typical case, arXiv:2004.09002.

[117] F. Glover, G. Kochenberger, and Y. Du, A tutorial on formulating and using QUBO models, arXiv:1811.11538.

[118] G. Nannicini, Performance of hybrid quantum-classical variational heuristics for combinatorial optimization, Phys. Rev. E **99**, 013304 (2019).

[119] M. S. Könz, W. Lechner, H. G. Katzgraber, and M. Troyer, Embedding overhead scaling of optimization problems in quantum annealing, PRX Quantum **2**, 040322 (2021).

[120] V. Panizza, P. Hauke, C. Micheletti, and P. Faccioli, Protein design by integrating machine learning and quantum-encoded optimization, PRX Life **2**, 043012 (2024).

[121] M. Upadhyay and M. N. Jones, Comparative studies of quantum annealing, digital annealing, and classical solvers for reaction network pathway analysis and mrna codon selection, arXiv:2509.09862.

[122] T. Koch, D. E. B. Neira, Y. Chen, G. Cortiana, D. J. Egger, R. Heese, N. N. Hegade, A. G. Cadavid, R. Huang, T. Itoko *et al.*, Quantum optimization benchmark library–the intractable decathlon, arXiv:2504.03832.

[123] R. E. Burkard, S. E. Karisch, and F. Rendl, Qaplib–a quadratic assignment problem library, J. Global Optim. **10**, 391 (1997).

[124] M. Inostroza-Ponta, A. Mendes, R. Berretta, and P. Moscato, An integrated QAP-based approach to visualize patterns of gene expression similarity, in *Progress in Artificial Life: Third Australian Conference; ACAL 2007 Gold Coast, Australia, December 4–6, 2007 Proceedings 3* (Springer, Berlin, 2007), pp. 156–167.

[125] M. Inostroza-Ponta, R. Berretta, and P. Moscato, QAPgrid: A two level QAP-based approach for large-scale data analysis and visualization, PLoS ONE **6**, e14468 (2011).

[126] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, Gene selection for cancer classification using support vector machines, Mach. Learn. **46**, 389 (2002).

[127] R. Díaz-Uriarte and S. Alvarez de Andrés, Gene selection and classification of microarray data using random forest, BMC Bioinf. **7**, 1 (2006).

[128] J. Staples, D. A. Nickerson, and J. E. Below, Utilizing graph theory to select the largest set of unrelated individuals for genetic analysis, Genetic Epidemiology **37**, 136 (2013).

[129] K. J. Abraham and C. Diaz, Identifying large sets of unrelated individuals and unrelated markers, Source Code Biol Med **9**, 6 (2014).

[130] S. C. Bull, M. R. Muldoon, and A. J. Doig, Maximising the size of non-redundant protein datasets using graph theory, PLoS ONE **8**, e55484 (2013).

[131] A. Hossain, E. Lopez, S. M. Halper, D. P. Cetnar, A. C. Reis, D. Strickland, E. Klavins, and H. M. Salis, Automated design of thousands of nonrepetitive parts for engineering stable genetic systems, Nat. Biotechnol. **38**, 1466 (2020).

[132] B. Mau, M. A. Newton, and B. Larget, Bayesian phylogenetic inference via Markov chain Monte Carlo methods, Biometrics **55**, 1 (1999).

[133] J. W. MacCluer, Monte carlo methods in human population genetics: A computer model incorporating age-specific birth and death rates, Am. J. Hum. Genet. **19**, 303 (1967).

[134] G. Mazzola, Sampling, rates, and reaction currents through reverse stochastic quantization on quantum computers, Phys. Rev. A **104**, 022431 (2021).

[135] D. Layden, G. Mazzola, R. V. Mishmash, M. Motta, P. Wocjan, J.-S. Kim, and S. Sheldon, Quantum-enhanced Markov chain Monte Carlo, Nature (London) **619**, 282 (2023).

[136] S. Arai and T. Kadowaki, Quantum annealing enhanced Markov-Chain Monte Carlo, Sci. Rep. **15**, 21427 (2025).
[137] A. Orfi and D. Sels, Bounding the speedup of the quantum-enhanced Markov-chain Monte Carlo algorithm, Phys. Rev. A **110**, 052414 (2024).
[138] J. Christmann, P. Ivashkov, M. Chiurco, and G. Mazzola, From quantum-enhanced to quantum-inspired Monte Carlo, Phys. Rev. A **111**, 042615 (2025).
[139] F. Slongo, P. Hauke, P. Faccioli, and C. Micheletti, Quantum-inspired encoding enhances stochastic sampling of soft matter systems, Sci. Adv. **9**, eadi0204 (2023).
[140] S. Whalen, J. Schreiber, W. S. Noble, and K. S. Pollard, Navigating the pitfalls of applying machine learning in genomics, Nat. Rev. Genet. **23**, 169 (2022).
[141] M. W. Libbrecht and W. S. Noble, Machine learning applications in genetics and genomics, Nat. Rev. Genet. **16**, 321 (2015).
[142] C. Miller, T. Portlock, D. M. Nyaga, and J. M. O'Sullivan, A review of model evaluation metrics for machine learning in genetics and genomics, Frontiers in Bioinformatics **4**, 1457619 (2024).
[143] A. Maurizio, Advances in liquid biopsy: Computational and artificial intelligence approaches in cancer research, Global Translational Medicine **3**, 3063 (2024).
[144] R. Poplin, P.-C. Chang, D. Alexander, S. Schwartz, T. Colthurst, A. Ku, D. Newburger, J. Dijamco, N. Nguyen, P. T. Afshar, S. S. Gross, L. Dorfman, C. Y. McLean, and M. A. DePristo, A universal SNP and small-indel variant caller using deep neural networks, Nat. Biotechnol. **36**, 983 (2018).
[145] Ž. Avsec, N. Latysheva, J. Cheng, G. Novati, K. R. Taylor, T. Ward, C. Bycroft, L. Nicolaisen, E. Arvaniti, J. Pan *et al.*, AlphaGenome: advancing regulatory variant effect prediction with a unified DNA sequence model, bioRxiv (2025).
[146] A. Butler, P. Hoffman, P. Smibert, E. Papalexi, and R. Satija, Integrating single-cell transcriptomic data across different conditions, technologies, and species, Nat. Biotechnol. **36**, 411 (2018).
[147] W. Hou and Z. Ji, Assessing GPT-4 for cell type annotation in single-cell RNA-seq analysis, Nat. Methods **21**, 1462 (2024).
[148] V. Schuster, E. Dann, A. Krogh, and S. A. Teichmann, Multi-dgd: A versatile deep generative model for multi-omics data, Nat. Commun. **15**, 10031 (2024).
[149] D. Kolobkov, S. Mishra Sharma, A. Medvedev, M. Lebedev, E. Kosaretskiy, and R. Vakhitov, Efficacy of federated learning on genomic data: A study on the UK biobank and the 1000 genomes project, Front. Big Data **7**, 1266031 (2024).
[150] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature (London) **549**, 195 (2017).
[151] M. Cerezo, G. Verdon, H.-Y. Huang, L. Cincio, and P. J. Coles, Challenges and opportunities in quantum machine learning, Nat. Comput. Sci. **2**, 567 (2022).
[152] M. Schuld, I. Sinayskiy, and F. Petruccione, An introduction to quantum machine learning, Contemp. Phys. **56**, 172 (2015).
[153] Y. Du, X. Wang, N. Guo, Z. Yu, Y. Qian, K. Zhang, M.-H. Hsieh, P. Rebentrost, and D. Tao, Quantum machine learning: A hands-on tutorial for machine learning practitioners and researchers, arXiv:2502.01146.
[154] G. Wang, J. Warrell, P. S. Emani, and M. Gerstein, Quantum variational autoencoder utilizing regularized mixed-state latent representations, Phys. Rev. A **111**, 042416 (2025).
[155] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, Nature (London) **567**, 209 (2019).
[156] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, Nat. Phys. **17**, 1013 (2021).
[157] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Barren plateaus in quantum neural network training landscapes, Nat. Commun. **9**, 4812 (2018).
[158] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, Connecting ansatz expressibility to gradient magnitudes and barren plateaus, PRX Quantum **3**, 010313 (2022).
[159] C. Ortiz Marrero, M. Kieferová, and N. Wiebe, Entanglement-induced barren plateaus, PRX Quantum **2**, 040316 (2021).
[160] H.-K. Zhang, S. Liu, and S.-X. Zhang, Absence of barren plateaus in finite local-depth circuits with long-range entanglement, Phys. Rev. Lett. **132**, 150603 (2024).
[161] A. Pesah, M. Cerezo, S. Wang, T. Volkoff, A. T. Sornborger, and P. J. Coles, Absence of barren plateaus in quantum convolutional neural networks, Phys. Rev. X **11**, 041011 (2021).
[162] M. Cerezo, M. Larocca, D. García-Martín, N. L. Diaz, P. Braccia, E Fontana, M. S. Rudolph, P. Bermejo, A. Ijaz, S. Thanasilp, E. R. Anschuetz and Z. Holmes, Does provable absence of barren plateaus imply classical simulability? Nat. Commun. **16**, 7907 (2025).
[163] P. Bermejo, P. Braccia, M. S. Rudolph, Z. Holmes, L. Cincio, and M. Cerezo, Quantum convolutional neural networks are (effectively) classically simulable, arXiv:2408.12739.
[164] F. J. Schreiber, J. Eisert, and J. J. Meyer, Classical surrogates for quantum learning models, Phys. Rev. Lett. **131**, 100803 (2023).
[165] R. Shaydulin, C. Li, S. Chakrabarti, M. DeCross, D. Herman, N. Kumar, J. Larson, D. Lykov, P. Minssen, Y. Sun *et al.*, Evidence of scaling advantage for the quantum approximate optimization algorithm on a classically intractable problem, Sci. Adv. **10**, eadm6761 (2024).
[166] J. Bowles, S. Ahmed, and M. Schuld, Better than classical? The subtle art of benchmarking quantum machine learning models, arXiv:2403.07059.
[167] L. Bai, L. Rossi, A. Torsello, and E. R. Hancock, A quantum Jensen–Shannon graph Kernel for unattributed graphs, Pattern Recogn. **48**, 344 (2015).
[168] L.-P. Henry, S. Thabet, C. Dalyac, and L. Henriet, Quantum evolution Kernel: Machine learning on graphs with programmable arrays of qubits, Phys. Rev. A **104**, 032416 (2021).
[169] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver, A quantum engineer's guide to superconducting qubits, Appl. Phys. Rev. **6**, 021318 (2019).
[170] D. Gottesman, The Heisenberg representation of quantum computers, in *Proceedings of the XXII International Colloquium on Group Theoretical Methods in Physics*, edited by S. P. Corney, R. Delbourgo, and P. D. Jarvis (International Press, Cambridge, MA, 1999), pp. 32–43.

[171] S. Bravyi and A. Kitaev, Universal quantum computation with ideal Clifford gates and noisy ancillas, Phys. Rev. A **71**, 022316 (2005).

[172] S. Bravyi and J. Haah, Magic-state distillation with low overhead, Phys. Rev. A **86**, 052329 (2012).

[173] T. Haug and M. S. Kim, Scalable measures of magic resource for quantum computers, PRX Quantum **4**, 010301 (2023).

[174] M. Boyer, G. Brassard, P. Høyer, and A. Tapp, Tight bounds on quantum searching, Fortschr. Phys. **46**, 493 (1998).

[175] D. K. Park, F. Petruccione, and J.-K. K. Rhee, Circuit-based quantum random access memory for classical data, Sci. Rep. **9**, 3949 (2019).

[176] M. W. Johnson, M. H. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk *et al.*, Quantum annealing with manufactured spins, Nature (London) **473**, 194 (2011).

[177] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv:1411.4028.

[178] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices, Phys. Rev. X **10**, 021067 (2020).

[179] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, A variational eigenvalue solver on a photonic quantum processor, Nat. Commun. **5**, 4213 (2014).

[180] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, The theory of variational hybrid quantum-classical algorithms, New J. Phys. **18**, 023023 (2016).

[181] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio *et al.*, Variational quantum algorithms, Nat. Rev. Phys. **3**, 625 (2021).

[182] T. Albash and D. A. Lidar, Demonstration of a scaling advantage for a quantum annealer over simulated annealing, Phys. Rev. X **8**, 031016 (2018).

[183] S. H. Sack and D. J. Egger, Large-scale quantum approximate optimization on nonplanar graphs with machine learning noise mitigation, Phys. Rev. Res. **6**, 013223 (2024).

[184] A. Miessen, D. J. Egger, I. Tavernelli, and G. Mazzola, Benchmarking digital quantum simulations above hundreds of qubits using quantum critical dynamics, PRX Quantum **5**, 040320 (2024).

[185] E. Pelofske, A. Bärtschi, L. Cincio, J. Golden, and S. Eidenbenz, Scaling whole-chip QAOA for higher-order Ising spin glass models on heavy-hex graphs, npj Quantum Inf. **10**, 109 (2024).

[186] H. G. Katzgraber, F. Hamze, Z. Zhu, A. J. Ochoa, and H. Munoz-Bauza, Seeking quantum speedup through spin glasses: The good, the bad, and the ugly, Phys. Rev. X **5**, 031026 (2015).

[187] S. Kirkpatrick Jr., C. D. Gelatt, and M. P. Vecchi, Optimization by simulated annealing, Science **220**, 671 (1983).

[188] C. Micheletti, P. Hauke, and P. Faccioli, Polymer physics by quantum computing, Phys. Rev. Lett. **127**, 080501 (2021).

[189] G. E. Santoro, R. Martoňák, E. Tosatti, and R. Car, Theory of quantum annealing of an Ising spin glass, Science **295**, 2427 (2002).