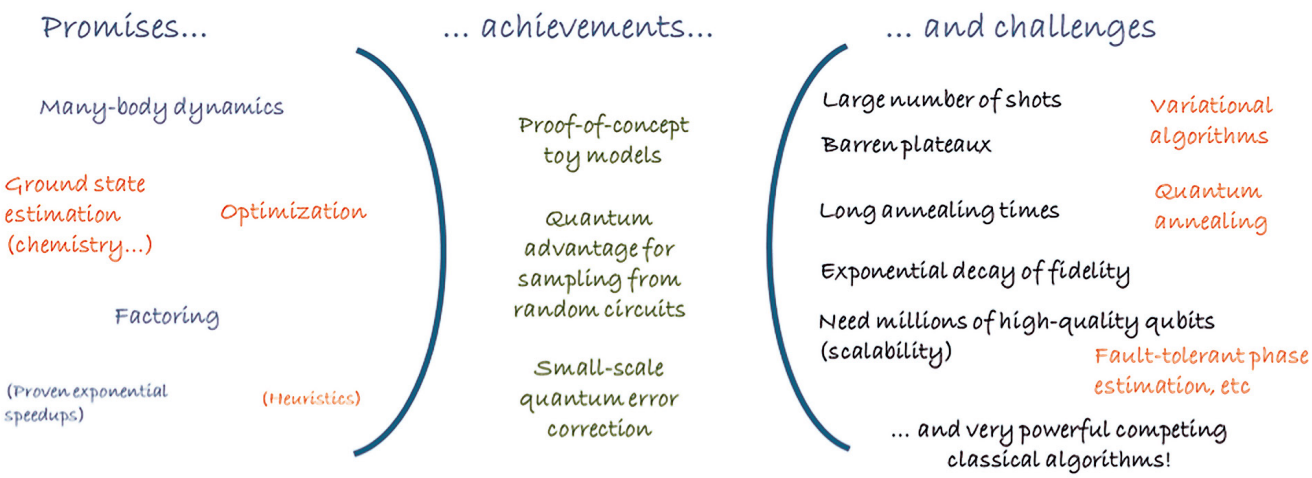
QUANTUM COMPUTING: PROMISES, ACHIEVEMENTS AND CHALLENGES

Thomas AYRAL
Eviden Quantum Lab, 78340 Les Clayes-sous-Bois, France
thomas.ayral@eviden.com



Quantum computers regularly make the headlines, with optimistic claims (often issued by companies large and small) alternating with pessimistic rebuttals (often by academic labs): sometimes they supposedly solve outstanding hard computational problems, sometimes their performances are dwarfed by classical machines. The goal of this article is to shed light on this back-and-forth, and explain what quantum computing could really be useful for.

https://doi.org/10.1051/photon/202513166

WHAT IS A QUANTUM COMPUTER?

The term of quantum computer encompasses a large variety of physical implementations. All have in common the (more or less precise) manipulation of individual objects with quantum properties: the spin of electrons, the energy levels of atoms, the polarization of photons, or even current loops in electrical circuits. Most quantum computing systems are engineered so that only two states — usually called $|0\rangle$ and $|1\rangle$ — of these individual objects are reachable during a computation. What is quantum about these objects is that they can not only be in state $|0\rangle$ or in state $|1\rangle$ (as would be the case for classical bits), but also in an arbitrary superposition of both: $|\psi\rangle = \alpha|0\rangle + \beta|0\rangle$, with α and β two complex numbers.

More importantly, these two-level systems, usually called qubits (for quantum bits), can be coupled to one another by special operations: two neighboring atoms can be coupled *via* a van der Waals interaction, two electrical circuits by a capacitive coupling,

for instance. This coupling generates a quintessentially quantum property called entanglement: the joint state of two entangled qubits cannot be described by specifying the individual state of each qubit. Hence, the state of two qubits, described by the superposition $\alpha_0|00\rangle + \alpha_1|01\rangle + \alpha_2|10\rangle + \alpha_3|11\rangle$ can in general not be factorized as $(\alpha_1|0\rangle + \beta_1|1\rangle)(\alpha_2|0\rangle + \beta_2|1\rangle)$. n qubits are thus described by 2^n coefficients. Conversely, N complex numbers could *a priori* be stored in $\log_2 N$ qubits! This exponential storage capacity can be leveraged in some algorithms, with an important caveat: reading out the information stored in the coefficients is not straightforward. Indeed, measuring a qubit in state $\alpha|0\rangle + \beta|1\rangle$ will only return 0 or 1 with respective probabilities given by the squared modulus of α and β. More than that, it will project the state to $|0\rangle$ or $|1\rangle$. Learning the precise value of α and β thus requires more work than meets the eye.

HOW TO PROGRAM A QUANTUM COMPUTER?

A quantum program is a list of instructions that evolve the state $|\Psi\rangle$ of the quantum computer from an initial state to a desired final state, which one can subject to quantum measurements in order to read off the solution to the problem at hand. From a physical perspective, these instructions essentially define a time-dependent Hamiltonian which, through Schrödinger's equation, completely determines the evolution of the system. The sequence of these instructions is commonly represented as a quantum circuit: a diagram whose horizontal lines represent qubits, and boxes represent the instructions, aka quantum gates that act on one, two or more qubits (lines) in a given order. For instance, the circuit used to implement a Fourier transform on a quantum computer is displayed in Fig. 1 for five qubits. Some gates (like the so-called Hadamard gate H) act on one qubit, corresponding to physical operations that act only on one qubit. They can put the qubit in a superposition state, but do not create entanglement. Some others (like the controlled-phase gates), act on two qubits, and may create entanglement.

A major theoretical advantage of quantum computers is that their quantum properties — superposition and entanglement — should afford them a computational advantage over classical processors. We can look at the Fourier transform circuit of Fig. 1 to understand this. On a quantum computer, executing a gate corresponds to a single operation, while on a classical computer, a generic quantum gate corresponds to a matrix-vector multiplication $U \cdot |\psi\rangle$. Since the vector in question is generally represented with size 2^n ● ● ●

in a classical computer memory, this operation has an exponential cost in the number n of qubits. Bookkeeping all operations in the circuit, we arrive at a cost that scales as $n2^n$ on a classical computer (the cost of a fast Fourier transform), while it scales as n^2 on the quantum computer: this is an exponential gain!

Yet, this impressive gain has strings attached, with a theoretical limitation and a practical one. Let us first dwell on the theoretical one. Supposing the coefficients α_0, α_0,..., α_{2^n-1}, of the Fourier transformed state $|\psi\rangle$ correspond to the values of pixels of an image we would like to recover. Given the probabilistic nature of quantum measurements, to obtain these very coefficients, we need at least 2^n readouts to learn the histogram (which, in fact, is only enough to learn the modulus of the α_i's)... And because of the projective nature of quantum measurements, this means repeating the circuit at least 2^n times... causing us to lose the exponential advantage. This rule has only one exception: if, from all the coefficients, only a few (let us say only one, the kth one, α_k) are nonzero, then the result of reading out the final state is always the same: it returns k with probability 1. Thus, only one circuit execution and readout are required. If moreover, the solution of our problem was to find k, then we have indeed obtained a speedup. In other words, in many quantum algorithms, acceleration can be reached only when the final distribution of coefficients is highly skewed (peaked), and the solution can be read off the peaks of the distribution. This is typically what Peter Shor's famous factoring algorithm does [1]. This is also what makes it difficult to invent efficient quantum algorithms for machine learning: it usually involves reading out a lot of information (in additional to loading a lot of training data, which is also costly) [2].

The second limitation is practical: quantum states are fragile to external classical influence. This means that the longer a computation, the more likely it is to be destroyed by external influence. This deleterious influence, called decoherence, degrades the quality of quantum states, called fidelity, exponentially with the number of gates. Hence, with current processors, only 100-1000 gates can be applied before too much harm happens. The quantum Fourier transform we mentioned above, with its n^2 gates, is already out of reach: with $n = 100$ qubits, it would require about 10 000 gates!

The art of quantum algorithmics thus boils down to finding creative ways to extract computational advantage despite these limitations.

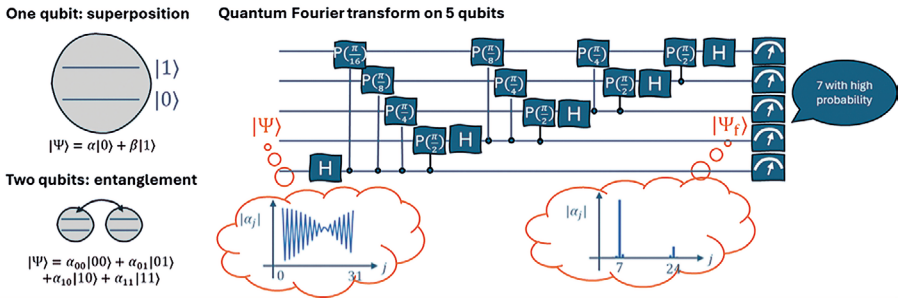


Figure 1. Quantum bits and quantum circuits. Top left: a single qubit can be in superposition of $|0\rangle$ and $|1\rangle$. Bottom left: two qubits can be entangled. Right: Quantum circuit representing a Fourier transform on $n = 5$ qubits, corresponding to a classical discrete FT on a vector of $N = 32$ points. The input wave function requires a potentially long preparation circuit. The output wavefunction (which contains the Fourier spectrum) is not directly accessible: only probabilistic measurements give access to the largest amplitudes.

WHAT USES FOR QUANTUM COMPUTERS?

The first concern of Richard Feynman, when he advocated the use of machines with quantum inside, was however not these intrinsically quantum limitations, but those of classical computers [3]. He had in mind a major conundrum of modern physics called the many-body problem. Ubiquitous in materials science, quantum chemistry, or nuclear physics, this problem arises in systems where interactions between particles matter. For instance, in solids, interactions between electrons are suspected to be the main origin of high-temperature superconductivity. Yet, interactions are also precisely the reason why these problems are difficult to tackle with classical computers: so-called mean-field approaches fail, and the more advanced methods that have been developed in the last fifty years all reach an exponential wall in some regime. For instance, tensor network techniques are sensitive to the amount of entanglement in the problem: their price scales exponentially with this entanglement. Monte-Carlo methods suffer from so-called sign problems that lead to statistical errors that diverge exponentially with system size or at low temperature [4].

Feynman pointed out that quantum computers, on the other hand, would be free from those ills, as information is directly stored in the system, and time evolution happens naturally through Schrödinger's equation, not *via* costly matrix vector multiplications (as in tensor networks) or high-dimensional integrals (computed in Monte-Carlo algorithms). In a way, by trying to simulate directly, namely with an artificial many-body system, the many-body physics that one is interested in, one does away with the problems of classical processors [5].

Quantum many-body problems are thus often believed to be among the first applications of quantum computers. As it turns out, outstanding computational problems

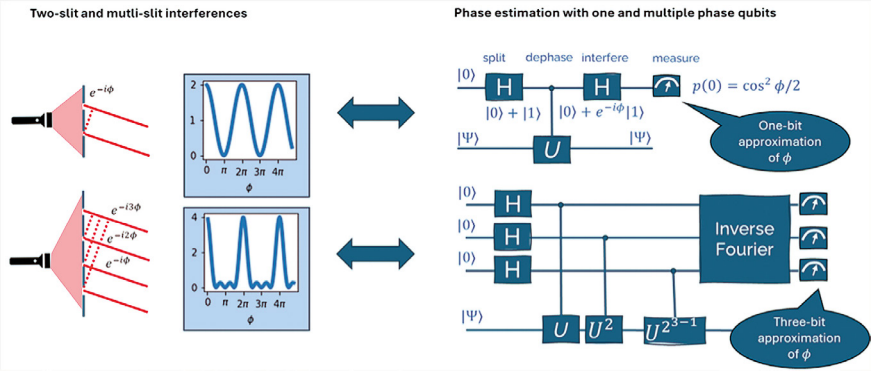


Figure 2. Analogy between interferometry in optics and quantum phase estimation with a quantum circuit. Left: interference between more waves leads to enhanced accuracy (sharper peaks) in the determination of the phase. Likewise, more phase qubits lead to a better precision.

outside of physics can also be regarded as many-body problems: a number of challenging optimization problems, like the famous travelling salesperson problem (find the shortest route to visit each city once and only once in a road network), can also be expressed as "interacting" Hamiltonians. Here, instead of physical interactions, interactions translate the fact that the different conditions on the sought-after solutions are interdependent. Thus, quantum time-evolution algorithms that relax the system to its resting state — which is hopefully the solution to the problem — were developed in this field of combinatorial optimization. Specialized computers called quantum annealers were even specifically constructed for tackling these very problems, with a major limitation: in principle, reaching the resting state takes very long times...

A second large class of quantum algorithms combines the natural time evolution afforded by quantum processors with another central physical phenomenon called interference: as in optics, adding two (or more) coherent waves yields a signal where the phase difference between the waves is easy to read out (see Fig. 2). Likewise, quantum computers engineer interferences between two (or more) signals whose phase difference contains the solution to a hard problem [6]. This algorithm, called quantum phase estimation, underlies Shor's factoring algorithm: the fact that the final distribution is peaked, as mentioned earlier, is a direct result of having many signals interfere in a smart way. Surprisingly, this phenomenon can also be used to invert systems of linear equations $Ax = b$. In this algorithm, the inversion is realized in a time that is exponentially faster than inverting the same linear system with classical algorithms [7]. Since linear systems are central to many application fields like solving partial differential equations, this has prompted many industrial companies to enter the field of quantum computing.

WILL QUANTUM COMPUTERS BEAT CLASSICAL COMPUTERS?

With the increasing availability of prototype quantum processors at the turn of the 2010s, these optimistic ideas were put to the test of reality in the last decade. In particular, practical implementations all face the exponential fidelity wall that we discussed above. With the error rates of current prototypes, between 1% and 0.1%, the number of gates that can be executed before decoherence sets in is limited to a few hundreds or thousands. This rules out all algorithms based on interference, which use a quantum Fourier transform and/or long, and therefore gate-intensive, time evolutions: applications like factoring numbers or inverting linear systems of equations are out of reach due to current (and mid-term) noise levels. In fact, even drastic improvements will not ● ● ●

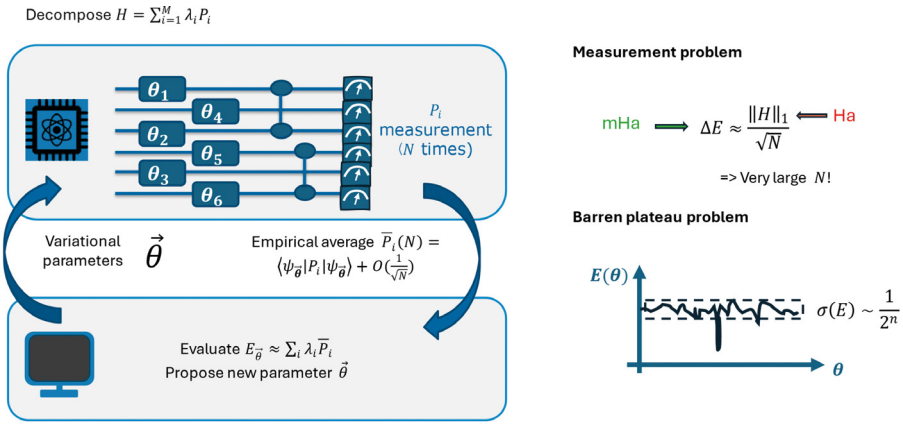


Figure 3. The variational quantum eigensolver algorithm... and its challenges. Left: VQE is a hybrid algorithm where the quantum processor (top) prepares a parameterized quantum state and measures the average values of the various Pauli terms that are contained in the Hamiltonian of the problem. These averages are combined into an estimate of the energy, which is used by a classical optimization algorithm to propose new parameters. The empirical average comes with a statistical error ΔE that leads to the so-called measurement problem of VQE (top right): the number N of samples required to reach chemical accuracies (1mHa) is very large. The energy landscape tends to be very flat for deep enough variational circuits, leading to trainability issues: this is the barren plateau problem (bottom right).

help much without the help of quantum error correction, a concept that Peter Shor borrowed from classical computers in the mid-1990s to fight against decoherence [8], and that we will touch on later.

CAN WE NEVERTHELESS SALVAGE SOMETHING FROM CURRENT PROCESSORS?

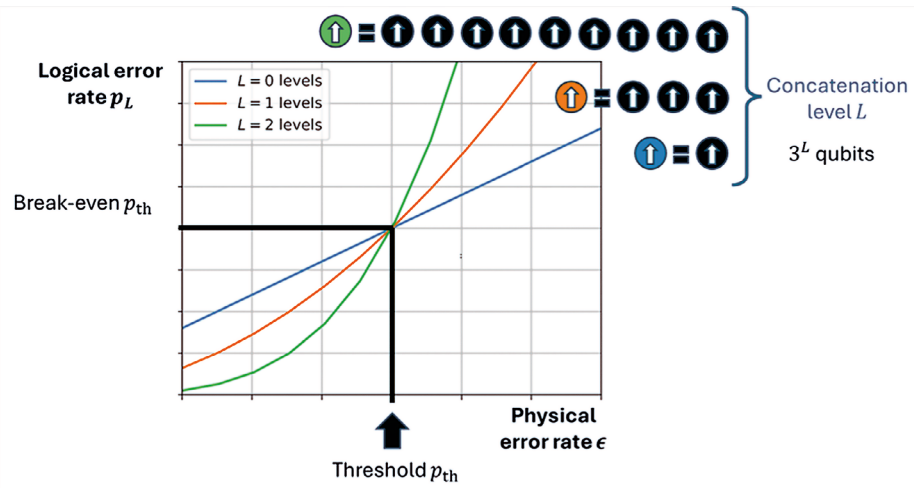
This goal is pursued by many researchers and engineers, with efforts to create algorithms that are short enough to beat decoherence, while at the same time overpowering classical processors.

To this aim, an old method, the variational method, was revisited with a quantum twist: to minimize the energy $\langle\psi(\vec{\theta})\,|H|\psi(\vec{\theta})\rangle$of a family of parameterized states, one uses a quantum computer to prepare a state $|\psi(\vec{\theta})\rangle$ and measure its energy, and a classical processor to propose new parameters $\vec{\theta}$ to reach a minimum of the energy landscape, as illustrated in Fig. 3. If the quantum computer can prepare states $|\psi(\vec{\theta})\rangle$ that are out of the reach of the best classical algorithms and measure their energy with high accuracy, this method, dubbed the variational quantum eigensolver (VQE [9]), could lead to some practical advantage. VQE, however, comes with two intrinsic limitations (in addition to decoherence): the probabilistic measurement of the energy requires many samples, and the training itself of the variational parameters turns out to be plagued with plateaus and hence exponential slowdowns [10].

The difficulties of VQE did not prevent "quantum advantage" claims on current processors. In fact, they were made without using VQE. For instance, the Google company resorted to random quantum circuits — which are known to produce very entangled states with a small amount of quantum gates — to argue they had reached "quantum supremacy" over classical machines [11]. Their first claims were rebutted by tensor-network based computations [12], but the newest generation of processors likely reached the initial goal [13]. This is however very far from any useful application.

Claims for useful quantum advantage were made by the IBM company in 2023 on a dynamical evolution problem [14], but they were quickly rebutted by classical computations, some of which were also based on tensor networks [15]. The relative ease with which classical computations reproduced or surpassed quantum computers can be attributed to the fact that physical systems usually obey constraints (like symmetries, conservation laws) that limit the growth of correlations or entanglement, and therefore make them tractable by classical algorithms, up to a certain point. Currently, the point where classical algorithms cease to work is

Figure 4. Principle of quantum error correction: by grouping several physical qubits into one logical qubit, one makes more noise-robust qubits, provided the physical (individual) error rate is lower than a certain threshold.



still beyond the point where decoherence makes quantum algorithms useless.

This mixed situation of current devices has prompted intense experimental efforts to make quantum error correction (QEC) work on the leading hardware platforms. QEC consists in protecting qubits against decoherence by spreading the information of one "logical" qubit over many "physical" qubits, and performing regular local measurements to detect and then correct local errors (see Fig. 4). Such a procedure is beneficial — the so-obtained logical qubit is better than the individual physical qubits — only if the individual qubits' error rates are below a certain threshold. Recent experiments have shown error rates below this threshold, opening perspectives for future QEC. However, the number of physical qubits required to implement algorithms such as time evolution, phase estimation or Shor's algorithm exceeds one million, far from the number of qubits (100-1000) available in today's prototypes. Going to these numbers will pose formidable scalability issues that make any prediction as to the first QEC-enabled quantum advantage a very tall order [16].

Whether near-term, uncorrected hardware will already provide quantum advantage on niche applications like many-body dynamics, or if this will be achieved by quantum error corrected hardware with the more traditional, gate-intensive quantum algorithms, is an open question. In fact, it could very well be that a clever blend of both paradigms delivers on the promises of quantum computers.

REFERENCES

[1] P. W. Shor, Proceedings 35th Annual Symposium on Foundations of Computer Science **1**, 124 (1995), https://doi.org/10.1109/SFCS.1994.365700

[2] M. Schuld, N. Killoran, PRX Quantum **3**, 030101 (2022), https://doi.org/10.1103/PRXQuantum.3.030101

[3] R. P. Feynman, Int. J. Theor. Phys. **21**, 467 (1982). https://doi.org/10.1007/BF02650179

[4] T. Ayral, C. R. Physique **26**, 25 (2025). https://doi.org/10.5802/crphys.229

[5] T. Ayral, P. Besserve, D. Lacroix, E. A. Ruiz Guzman, Eur. Phys. J. A **59**, 227 (2023), https://doi.org/10.1140/epja/s10050-023-01141-1

[6] R. Cleve, A. Ekert, C. Macchiavello, M. Mosca, Proc. R. Soc. A **454**, 339 (1998), https://doi.org/10.1098/rspa.1998.0164

[7] A. W. Harrow, A. Hassidim, S. Lloyd, Phys. Rev. Lett. **103**, 150502 (2009). https://doi.org/10.1103/PhysRevLett.103.150502

[8] P. W. Shor, Phys. Rev. **A 52**, R2493 (1995). https://doi.org/10.1103/PhysRevA.52.R2493

[9] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, J. L. O'Brien, Nat. Commun. **5**, 4213 (2013). https://doi.org/10.1038/ncomms5213

[10] M. Larocca, S. Thanasilp, S. Wang, K. Sharma, J. Biamonte, P. J. Coles, L. Cincio, J. R. McClean, Z. Holmes, M. Cerezo, "A Review of Barren Plateaus in Variational Quantum Computing," arXiv:2405.00781 (2024). http://arxiv.org/abs/2405.00781

[11] F. Arute et al., Nature **574**, 505 (2019). https://doi.org/10.1038/s41586-019-1666-5

[12] T. Ayral, T. Louvet, Y. Zhou, C. Lambert, E. M. Stoudenmire, X. Waintal, PRX Quantum **4**, 020304 (2023). https://doi.org/10.1103/PRXQuantum.4.020304

[13] A. Morvan et al., Nature **634**, 328 (2024). https://doi.org/10.1038/s41586-024-07998-6

[14] Y. Kim et al., Nature **618**, 500 (2023). https://doi.org/10.1038/s41586-023-06096-3

[15] J. Tindall, M. Fishman, E. M. Stoudenmire, D. Sels, PRX Quantum **5**, 010308 (2024). https://doi.org/10.1103/PRXQuantum.5.010308

[16] M. Mohseni et al., "How to Build a Quantum Supercomputer: Scaling Challenges and Opportunities," arXiv:2411.10406 (2024). http://arxiv.org/abs/2411.10406