Complexity Classes and 2D Short-Range Entangled States

November 12, 2025

1 Hamiltonian ground state

In this lecture, we shift our focus from single-qubit sensing to the intricate landscape of quantum many-body physics. A **local Hamiltonian** acting on n qubits takes the form

$$H = \sum_i h_i$$

where each term h_i is a Hermitian operator acting nontrivially on at most a constant number of qubits, with the identity on all others. The ground state $|\psi_g\rangle$ is the eigenvector of H with minimum eigenvalue E_g. A fundamental challenge in physics is the **Ground State Energy Problem**:

Problem: Given a local Hamiltonian $H = \sum_i h_i$ and two real numbers $a < b$ with a promise that the ground state energy E_g is either $E_g < a$ or $E_g > b$, decide which is the case.

A fundamental question is the following:

Q: How computationally difficult is this problem?

The difficulty depends on the nature of the local terms h_i:

- If h_i are diagonal in the computational basis (e.g., from the set $\{I, Z\}^{\otimes n}$), the problem is known to be **NP-complete**. (Can simulate everything in NP.)

  Example: 3-SAT is the classic satisfiability problem where the Boolean formula is in conjunctive normal form (CNF) and each clause has exactly three literals. For instance, consider

  $$(x_1 \vee \neg x_2 \vee x_3) \ \wedge \ (\neg x_1 \vee x_2 \vee \neg x_3) \ \wedge \ (x_2 \vee x_3 \vee \neg x_4). \tag{1}$$

  This is satisfiable if we can find Boolean x_i's such that the above equals zero. We can encode this into a classical Hamiltonian problem as

  We can encode this into a classical Hamiltonian problem as follows. Let $x_i \in \{0, 1\}$ (with $1 = \text{True}$). For each clause, add a penalty that is 1 iff the clause is violated and 0 otherwise:

  $$P_1 = (1 - x_1)\, x_2\, (1 - x_3), \tag{2}$$
  $$P_2 = x_1\, (1 - x_2)\, x_3, \tag{3}$$
  $$P_3 = (1 - x_2)\, (1 - x_3)\, x_4. \tag{4}$$

  Define the total energy

  $$E(x_1, x_2, x_3, x_4) \ = \ P_1 + P_2 + P_3. \tag{5}$$

By construction, $E \in \{0,1,2,3\}$ equals the number of violated clauses, so the instance is satisfiable iff $\min E = 0$.

Quantum Hamiltonian (projector form). Work on four qubits in the computational basis. Let $P_0 = |0\rangle\langle 0| = \frac{I+Z}{2}$ and $P_1 = |1\rangle\langle 1| = \frac{I-Z}{2}$. Each clause contributes the projector onto its unique violating assignment (acting as identity on untouched qubits):

$$H_{C_1} = P_0^{(1)} \otimes P_1^{(2)} \otimes P_0^{(3)} \otimes I^{(4)} = \tfrac{1}{8}\,(I+Z_1)(I-Z_2)(I+Z_3) \otimes I_4, \tag{6}$$

$$H_{C_2} = P_1^{(1)} \otimes P_0^{(2)} \otimes P_1^{(3)} \otimes I^{(4)} = \tfrac{1}{8}\,(I-Z_1)(I+Z_2)(I-Z_3) \otimes I_4, \tag{7}$$

$$H_{C_3} = I^{(1)} \otimes P_0^{(2)} \otimes P_0^{(3)} \otimes P_1^{(4)} = \tfrac{1}{8}\, I_1 \otimes (I+Z_2)(I+Z_3)(I-Z_4). \tag{8}$$

Sum them to get the Hamiltonian

$$H = H_{C_1} + H_{C_2} + H_{C_3}. \tag{9}$$

For any computational-basis state $|x_1x_2x_3x_4\rangle$, the energy $\langle H\rangle$ equals the number of violated clauses. Hence the formula is satisfiable iff the ground energy $E_0(H) = 0$ (*e.g.*, for the k-Local Hamiltonian promise problem one may take thresholds $a = 0$ and $b = \frac{1}{2}$).

Hardness peaks near the phase transition when the clause-to-variable ratio $\alpha = m/n$ is about 4.26 (goes from mostly satisfiable to mostly unsatisfiable).

- If h_i can be arbitrary (e.g., from $\{I, X, Y, Z\}^{\otimes n}$), the problem is **QMA-complete**, the quantum analogue of NP-complete believed to be hard even for quantum computers.

Given this hardness, we focus on a specific, physically relevant class of ground states that are more tractable: **Short-Range Entangled (SRE) states**.

2 Short-Range Entangled (SRE) States

SRE states have a limited amount of entanglement, confined to local regions. From a quantum information perspective, SRE states can be defined as states that can be prepared from a simple product state, like $|0\rangle^{\otimes n}$, by a **constant-depth quantum circuit**. Imagine starting with all qubits in the $|0\rangle$ state. We then apply a sequence of layers of local quantum gates (e.g., two-qubit gates). The "depth" of the circuit is the number of such layers. For an SRE state, this depth is a constant, $O(1)$, meaning it does not grow as the number of qubits n increases. This structural property is key to their simplicity. The key question for this lecture is whether such states can be efficiently simulated on a classical computer.

Q: Are 2D SRE states and 2D shallow QNNs classically easy?

Simulating these systems is believed to be classically hard. **Given that they are always quantumly easy, this provides the foundation for demonstrating quantum advantage in machine learning tasks.**

A brief review of complexity classes

To understand this hardness, we need to introduce some powerful computational complexity classes.

- BPP and BQP: These are the standard complexity classes associated with the sets of decision problems that can be solved in polynomial time by randomized classical algorithms (BPP) and quantum algorithms (BQP).
- PostBPP: The class of problems solvable by a probabilistic classical computer with the (physically unrealistic) ability of postselection: forcing a random bit to yield a specific outcome.
- PostBQP: The class of problems solvable by a quantum computer with the (physically unrealistic) ability of postselection: forcing a measurement to yield a specific outcome. A landmark result by Scott Aaronson shows that $\mathsf{PostBQP} = \mathsf{PP}$.
- PP (Probabilistic Polynomial Time): A problem is in PP if there is a randomized algorithm that accepts with probability $> 1/2$ for "yes" instances and $< 1/2$ for "no" instances. Unlike BPP, the gap can be exponentially small, making it akin to counting solutions (counting how many possible assignments of the random bits correspond to "yes" vs "no").
- The **Polynomial Hierarchy (PH)**: An infinite tower of complexity classes that generalizes NP. We define $\Sigma_0 = \mathsf{P}$, $\Sigma_1 = \mathsf{NP}$, $\Sigma_2 = \mathsf{NP}^{\mathsf{NP}}$, $\Sigma_k = \mathsf{NP}^{\Sigma_{k-1}}$, and so on. It is strongly believed that this hierarchy is infinite, that is, $\Sigma_k \neq \Sigma_{k+1}$ for any k. If PH were to "collapse" to a finite level, it would be a revolutionary result in computer science.

To gain a bit of intuition for why the Polynomial Hierarchy is believed to be an infinite tower of progressively harder problems, we can look at its lower levels. The notation $\mathsf{C}_1^{\mathsf{C}_2}$ denotes an "oracle" complexity class: the set of problems solvable by a machine that runs in the time complexity of C_1 but has access to a magical subroutine (an oracle) that can instantly solve any problem in class C_2. Giving a machine access to an oracle can substantially increase its power. For example, giving a polynomial-time machine a polynomial-time oracle (P^{P}) doesn't help: the machine could have just solved the subroutine itself, so $\mathsf{P}^{\mathsf{P}} = \mathsf{P}$.

However, the situation changes when the oracle is more powerful. Consider the relationship between NP and its complement, coNP. A problem is in NP if "yes" instances have short, verifiable proofs. The classic example is the **Boolean Satisfiability Problem (SAT)**: given a logical formula like $(x_1 \vee \neg x_2) \wedge x_3$, is there at least one assignment of "True" or "False" to the variables that makes the whole formula true? If the answer is "yes," the proof is simple: the satisfying assignment itself. You don't have to check every possibility, just the one that works. A problem is in coNP if "no" instances have short proofs. The canonical example here is the **Tautology Problem (TAUTOLOGY)**: is a given logical formula true for *every* possible assignment of its variables? If the answer is "no," the proof is again simple: a single counterexample assignment that makes the formula false.

It is widely believed that $\mathsf{NP} \neq \mathsf{coNP}$ because there is a fundamental asymmetry between these two problems. Finding a single satisfying assignment for SAT (a proof for a "yes" answer) seems computationally much easier than proving that a formula is a tautology, which requires showing it's true for *all* assignments. The short proof for one problem doesn't seem to help you find a short proof for the other. However, they are indeed closely related. It is well known that NP and coNP are both contained within P^{NP}.

The containment of coNP is particularly instructive. Let's trace the algorithm for solving a coNP problem (like TAUTOLOGY) using a polynomial-time machine with access to an NP oracle.

1. Our machine is given a Boolean formula ϕ and must decide if it is a tautology.
2. The key logical step is this: ϕ is a tautology if and only if its negation, $\neg\phi$, is *unsatisfiable*.
3. The question "Is $\neg\phi$ unsatisfiable?" is a coNP problem. However, its complement, "Is $\neg\phi$ *satisfiable*?" is an instance of SAT, which is in NP.
4. Our machine constructs the formula $\neg\phi$ (a simple, polynomial-time operation).
5. It then asks the NP oracle: "Is this new formula, $\neg\phi$, satisfiable?" The oracle, which can solve any NP problem, answers "yes" or "no" instantly.
6. Our machine takes the oracle's one-bit answer and flips it to get the final result:
   - If the oracle says "yes" ($\neg\phi$ is satisfiable), it means there's a counterexample to ϕ being a tautology. Our machine outputs "no."
   - If the oracle says "no" ($\neg\phi$ is unsatisfiable), it means there are no counterexamples, so ϕ must be a tautology. Our machine outputs "yes."

This algorithm runs in polynomial time, proving that any problem in coNP can be solved in P^{NP}. Since P^{NP} contains both NP and coNP, it is considered strictly more powerful than either class alone (assuming they are different). The next level of the hierarchy, $\Sigma_2 = \mathsf{NP}^{\mathsf{NP}}$, allows a nondeterministic machine to access an NP oracle. This is believed to be a significant leap in computational power over plain NP, providing the intuition that each level of the hierarchy is genuinely harder than the last.

A celebrated result by Seinosuke Toda, known as **Toda's Theorem**, provides a shocking link between these classes:

$$\mathsf{PH} \subseteq \mathsf{P}^{\mathsf{PP}}$$

This means that any problem in the entire Polynomial Hierarchy can be solved efficiently by a classical computer that has access to a "PP oracle", a black box that can solve any problem in PP instantly. This shows the immense power of counting-based complexity.

Proving $\mathsf{PostBPP} \neq \mathsf{PostBQP}$ assuming PH does not collapse

Why do we need all of these complexity classes? The main usage is to derive that $\mathsf{PostBPP} \neq \mathsf{PostBQP}$, contingent on a standard conjecture in complexity theory. The argument proceeds by contradiction.

1. **Equalities and Inclusions:** From landmark results, we know:
   - $\mathsf{PostBQP} = \mathsf{PP}$ (Aaronson's Theorem).
   - $\mathsf{PostBPP} \subseteq \mathsf{BPP}^{\mathsf{NP}}$ (See *Threshold Computation and Cryptographic Security*; Intuition: an NP oracle can be used to find accepting paths to postselect upon, hence probabilistic classical machines with postselection can be simulated by a probabilistic classical machines with access to a magical machine that can solve any NP problems).
   - $\mathsf{BPP} \subseteq \mathsf{NP}^{\mathsf{NP}}$ (Sipser-Gowers-Lautemann Theorem).
2. **Placing PostBPP in the Polynomial Hierarchy:** We can combine the inclusions above.
$$\mathsf{PostBPP} \subseteq \mathsf{BPP}^{\mathsf{NP}} \subseteq (\mathsf{NP}^{\mathsf{NP}})^{\mathsf{NP}} = \mathsf{NP}^{\mathsf{NP}} = \Sigma_2$$
   This places classical postselection firmly within the second level of the Polynomial Hierarchy.
3. **The Power of a PostBPP Oracle:** If we give a polynomial-time machine an oracle for PostBPP, its power is contained within that of a Σ_2 oracle:
$$\mathsf{P}^{\mathsf{PostBPP}} \subseteq \mathsf{P}^{\Sigma_2} = \mathsf{P}^{\mathsf{NP}^{\mathsf{NP}}}$$
   This class, $\mathsf{P}^{\mathsf{NP}^{\mathsf{NP}}}$, is contained in the third level, Σ_3, of the Polynomial Hierarchy.
4. **Contradiction:** Let us assume for a moment that $\mathsf{PostBPP} = \mathsf{PostBQP}$.
   - If this were true, then since $\mathsf{PostBQP} = \mathsf{PP}$, we would have $\mathsf{PostBPP} = \mathsf{PP}$.
   - This implies that a polynomial-time machine with a PostBPP oracle is as powerful as one with a PP oracle: $\mathsf{P}^{\mathsf{PostBPP}} = \mathsf{P}^{\mathsf{PP}}$.
   - From Toda's Theorem, we know that the Polynomial Hierarchy is contained in P^{PP}.
   - Combining these points, our assumption leads to: $\mathsf{PH} \subseteq \mathsf{P}^{\mathsf{PP}} = \mathsf{P}^{\mathsf{PostBPP}} \subseteq \Sigma_3$.

   The result, $\mathsf{PH} \subseteq \Sigma_3$, means that the entire, supposedly infinite, Polynomial Hierarchy collapses to its third level. This is a major, widely disbelieved conjecture. Therefore, our initial assumption $\mathsf{PostBPP} = \mathsf{PostBQP}$ must be false.

This leads to the final result: assuming the Polynomial Hierarchy does not collapse, we must have

$$\mathsf{PostBPP} \neq \mathsf{PostBQP}.$$

Coming up next, we will use this relation to prove that 2D SRE states and 2D shallow QNN offer quantum advantage (classically hard but quantumly easy).

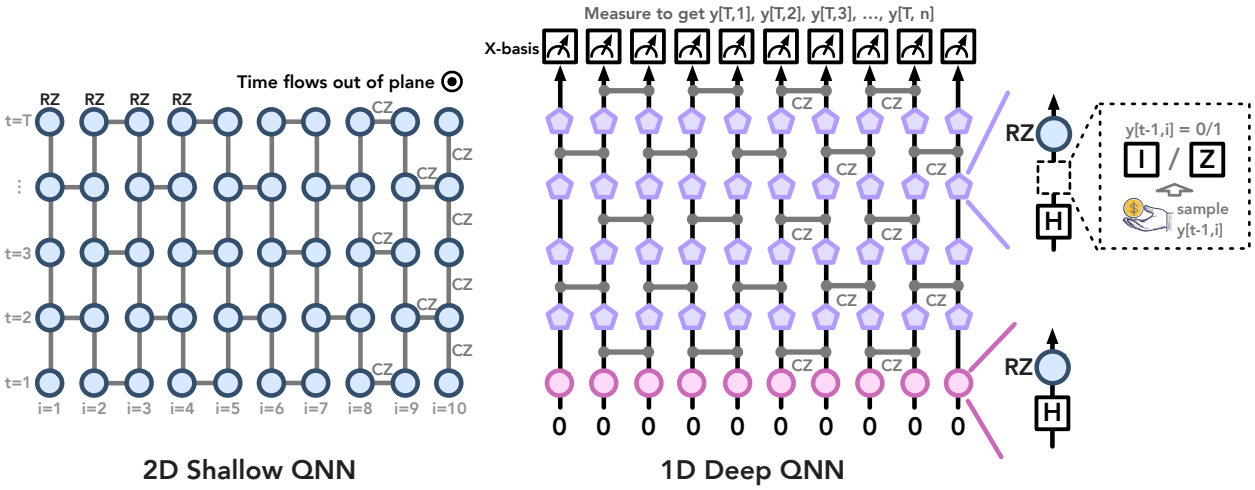

Figure 1: Equivalence between a 2D shallow QNN (left) and a 1D deep QNN (right). The shallow circuit acts on an $n \times T$ grid of qubits in constant depth. This can be reinterpreted as a deep circuit on a 1D line of n qubits, where one spatial dimension is traded for the time dimension. Rotation angles in the R_Z gates match those in the 2D shallow QNN. For pentagon-shaped gates, a coin flip determines whether to implement an identity or Z gate.

Classical Hardness of 2D Shallow QNNs

We now leverage complexity-theoretic results to establish the classical hardness of simulating 2D shallow QNNs. The argument hinges on showing that efficient classical simulation would precipitate a major collapse in complexity classes. The core of this collapse is the equivalence between shallow 2D circuits and deep 1D circuits.

Mapping 2D Shallow QNNs to 1D Deep QNNs

A key observation is that shallow 2D QNNs, despite their constant-depth structure, are computationally equivalent to deep sequential 1D circuits. Understanding this mapping, illustrated in Figure 1, is essential for the hardness proof.

2D Shallow QNN. The left side of Figure 1 depicts the 2D shallow QNN.

- **Structure:** An $n \times T$ grid of qubits, indexed by $i \in \{1, \ldots, n\}$ (space) and $t \in \{1, \ldots, T\}$ (treated as a second spatial dimension).
- **Computation:** Time flows out of the plane, meaning the entire computation consists of a constant number of gate layers applied across the 2D grid:
  1. Hadamard gates on all qubits.
  2. Single-qubit R_Z rotations and two-qubit CZ gates between adjacent qubits.
  3. Hadamard gates on all qubits.

  The number of layers remains constant in system size, hence the circuit is shallow.

When we input $|0\rangle^{n \times T}$ to the 2D shallow QNN and measure in the Z basis, the resulting bitstring distribution is equivalent to sampling from a 1D deep QNN followed by X-basis measurement. This

Algorithm 1 2D Shallow QNN

Input: Total time T, 2D graph G with nodes (t, i), model parameters $\beta_{t,i}$
Output: Bitstring measurements $\{y_{t,i}\}$

Initialize all qubits (t, i) to $|0\rangle$
Apply H to all qubits (t, i)
Apply $R_Z(\beta_{t,i})$ to all qubits (t, i)
Apply CZ gates between all neighboring qubits (neighbors defined by edges in G)
Apply H to all qubits (t, i)
Measure all qubits (t, i) in the Z basis
Assign measurement result for qubit (t, i) to classical bit $y_{t,i}$

correspondence follows from quantum teleportation. Consider an n-qubit state $|\psi\rangle$. Measuring the second qubit of the following state:

$$(\mathrm{CZ} \otimes \mathrm{Id}_{n-1})(|+\rangle \otimes |\psi\rangle) \tag{10}$$

in the X basis teleports the first qubit of $|\psi\rangle$ to the location of $|+\rangle$ with an additional Hadamard gate and I or Z gate depending on whether the measurement yields $|+\rangle$ or $|-\rangle$. Additionally, CZ commutes with $R_Z(\beta)$ (both are diagonal), allowing arbitrary reordering.

1D Deep QNN. The right side of Figure 1 shows the equivalent 1D deep QNN.

- **Structure:** A 1D line of n qubits (corresponding to the $i = 1, \ldots, n$ axis).
- **Computation:** The second spatial dimension $t = 1, \ldots, T$ becomes the time axis, with computation flowing from bottom to top. This makes the circuit deep, with depth proportional to T (the height of the square lattice for the 2D shallow QNN).

Algorithm 2 1D Deep QNN

Input: Total time T, 2D graph G with nodes (t, i), model parameters $\beta_{t,i}$
Output: Bitstring measurements $\{y_{t,i}\}$

Initialize all qubits i to $|0\rangle$
Apply H to all qubits i ▷ Operations at time $t = 1$
Apply $R_Z(\beta_{1,i})$ to all qubits i
for all edges $((1, i), (1, j))$ in G **do**
    Apply CZ gate between qubits i and j
for $t = 2$ to T **do** ▷ Main loop for time steps $t = 2$ to T
    for all edges $((t, i), (t, j))$ in G **do**
        Apply CZ gate between qubits i and j
    for all qubits i **do**
        Apply H to qubit i
        Sample $y_{t-1,i} \sim \mathrm{Uniform}(\{0, 1\})$
        Apply $Z^{y_{t-1,i}}$ followed by $R_Z(\beta_{t,i})$ to qubit i
for all qubits i **do** ▷ Final measurement at time T
    Measure qubit i in the X basis and assign result to $y_{T,i}$

The Classical Hardness Proof

The crucial insight is that these two models are mathematically equivalent. The 2D shallow and 1D deep circuits in Figure 1 represent identical processes, guaranteed to produce samples from the same probability distribution. This equivalence provides the key link for the hardness argument, which proceeds by contradiction.

1. **Assumption:** Suppose there exists an efficient randomized classical algorithm $\mathcal{A}$ that samples from the output distribution of any shallow 2D QNN.

2. **Implication from Equivalence:** By the mathematical equivalence established in Algorithms 1 and 2, the algorithm $\mathcal{A}$ must also efficiently sample from the output of the corresponding randomized 1D deep QNN, where the randomness arises from the uniformly random choices of $y_{t,i} \in \{0, 1\}$ determining whether to apply I or Z gates at each intermediate step.

3. **Postselection:** Consider now an algorithm in $\mathsf{PostBPP}$ that has access to $\mathcal{A}$ as a subroutine. We demonstrate this algorithm can solve any problem in $\mathsf{PostBQP}$. Using the (physically unrealistic) power of postselection in classical randomized computation, we postselect on the event that $y_{t,i} = 0$ for all $t \in \{1, \ldots, T-1\}$ and $i \in \{1, \ldots, n\}$. This postselection eliminates the stochastic I/Z gates, reducing the 1D deep QNN to a deterministic quantum circuit.

4. **Universal Quantum Computation:** The gate set $\{H, R_Z(\theta), \mathrm{CZ}\}$ is universal for quantum computation. Therefore, by appropriately choosing the parameters $\{\beta_{t,i}\}$ and the graph structure G of the 2D shallow QNN (equivalently, the 1D deep QNN after postselection), we can implement any polynomial-size quantum circuit. Furthermore, we can postselect on the final measurement outcomes of the 1D deep QNN to simulate quantum postselection. Consequently, the randomized classical algorithm $\mathcal{A}$ with postselection can simulate any quantum computation with postselection, establishing that $\mathsf{PostBPP} = \mathsf{PostBQP}$.

5. **Contradiction:** The equality $\mathsf{PostBPP} = \mathsf{PostBQP}$ implies that the Polynomial Hierarchy collapses to its third level, directly contradicting the widely accepted conjecture that PH does not collapse. Therefore, our initial assumption must be false.

Together, we have proved that simulating 2D shallow QNNs is classically hard assuming standard complexity-theoretic conjecture. **This establishes 2D shallow QNNs as a powerful computational model and a good candidate for demonstrating quantum advantage.**