AQAlg: Advanced Quantum Algorithms **2023 - 2024**

Lecture 4: Quantum chemistry and simulation

Lecturer: Simon Apers (apers@irif.fr)

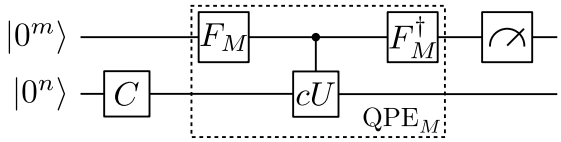
In previous lectures we discussed the use of quantum algorithms for solving essentially "non-quantum" problems such as factoring and optimization. However, the most natural (and maybe most promising) application of quantum computers is the simulation of quantum systems and quantum chemistry.

1 Ground state energy

A canonical problem in quantum chemistry is that of estimating the ground state energy of a molecular or condensed matter Hamiltonian. In full generality, this problem is QMA-hard (see next lectures by A. Grilo), and so we do not expect to solve this efficiently even on a quantum computer. However, there is hope that more "natural" Hamiltonians do allow for efficient quantum algorithms, while still being hard for classical algorithms.

1.1 Quantum phase estimation

There is a very natural quantum algorithm for estimating the ground state energy of an n-qubit Hamiltonian H that is based on quantum phase estimation. It corresponds to the following circuit:



Some details:

1. C is an n-qubit circuit that prepares an "ansatz state" $|\Psi_0\rangle$. Ideally this corresponds to a rough approximation of the ground state $|\phi_0\rangle$ of H.
2. The algorithm then applies quantum phase estimation on $|\Psi_0\rangle$ with operator $U = e^{iH}$ and precision $2^{-m} = 1/M$.
3. Finally, the phase estimation register is measured. The measurement result yields an estimate of the ground state energy.

Assume H has eigenvalue decomposition $H = \sum_{i=0}^{2^n-1} \lambda_i |\phi_i\rangle$, with eigenvalues $\lambda_0 \leq \cdots \leq \lambda_{2^n-1}$, and the ansatz has decomposition $|\Psi_0\rangle = \sum_i \alpha_i |\phi_i\rangle$. We can track the evolution in the circuit:

$$\begin{aligned} |0^m\rangle |0^n\rangle &\xrightarrow{C} |0^m\rangle |\Psi_0\rangle = \sum_i \alpha_i |0^m\rangle |\phi_i\rangle \\ &\xrightarrow{\mathrm{QPE}_M} \sum_i \alpha_i |\tilde{\lambda}_i\rangle |\phi_i\rangle \\ &\xrightarrow{\text{meas.}} |\tilde{\lambda}_i\rangle |\phi_i\rangle \quad \text{with probability} \quad p_i = |\alpha_i|^2. \end{aligned}$$

Here $\tilde{\lambda}_i$ is an estimate such that $|\tilde{\lambda}_i - \lambda_i| \leq 1/M$.

The *complexity* of the circuit is dominated by (i) the complexity of preparing the ansatz (i.e., implementing C), and (ii) the complexity of doing Hamiltonian simulation with Hamiltonian H for time $O(M)$.

Correctness of the algorithm hinges on the overlap $|\alpha_0|^2 = |\langle \Psi_0 | \phi_0 \rangle|^2$ of our ansatz state $|\Psi_0\rangle$ with the actual ground state $|\phi_0\rangle$. The probability of actually outputting an estimate of the ground state energy is $|\alpha_0|^2$. If this is sufficiently large (e.g., $|\alpha_0|^2$ constant), then after a constant number of iterations, the algorithm will output a correct estimate of the ground state energy. On the other hand, if we have no clue about the ground state $|\phi_0\rangle$, then we could pick as ansatz a uniformly random initial state, but then the overlap would only be $|\alpha_0|^2 = 1/2^n$ in expectation. The algorithm then returns an estimate of a uniformly random energy level of the Hamiltonian, which is typically useless.

Common ansatzes are based on the quantum adiabatic algorithm, classical approximations such as Hartree-Fock, or variational quantum circuits, as we discuss next.

1.2 VQE

Apart from needing a good ansatz, the phase estimation algorithm also requires that we can efficiently simulate the problem Hamiltonian H (i.e., implement e^{iHt}). While this would typically be feasible on a full-fledged quantum computer (see last lecture), there is currently interest in using "near-term" quantum computing devices for which Hamiltonian simulation might be too hard. The "Variational Quantum Eigensolver" (VQE) is a variational quantum algorithm that tries to address both the ansatz and the Hamiltonian simulation problem.

VQE is similar to QAOA, but it is designed to minimize the energy of a general Hamiltonian (rather than a diagonal cost Hamiltonian). I.e., it aims to solve

$$\min_{\langle \psi | \psi \rangle = 1} \langle \psi | H | \psi \rangle \,. \tag{1}$$

By the variational principle of quantum mechanics, the minimum equals the ground state energy λ_0 of H, and it is achieved when $|\psi\rangle$ is a ground state of H. Similar to QAOA, VQE uses a parameterized circuit to generate $|\psi\rangle$. The parameters of the circuit are tuned so as to minimize the energy (1).

For QAOA, the Hamiltonian H was diagonal, and so we could easily evaluate the energy of the output state by measuring all qubits and evaluating the cost function $h(z)$ for the resulting bit string z. VQE considers general Hamiltonians, and so evaluating (1) requires more work. One approach is to use quantum phase estimation, as in the previous section, but this requires simulating the Hamiltonian evolution e^{iHt} and this can be expensive. Instead, the Hamiltonian H is decomposed as a linear combination of Pauli operators,

$$H = \sum_{i_1,\ldots,i_n=0}^{3} \alpha_{i_1 \ldots i_n} \, \sigma_{i_1} \otimes \sigma_{i_2} \otimes \cdots \otimes \sigma_{i_n} .$$

We can then rewrite the energy as

$$\langle \psi | H | \psi \rangle = \sum_{i_1,\ldots,i_n=0}^{3} \alpha_{i_1 \ldots i_n} \, \langle \psi | \sigma_{i_1} \otimes \sigma_{i_2} \otimes \cdots \otimes \sigma_{i_n} | \psi \rangle \,,$$

and so it suffices to estimate the energy of $|\psi\rangle$ with respect to each of the individual Pauli terms. This can be done relatively straightforwardly (see exercises). While there can be 4^n such terms, the

sum can often be truncated to poly(n) terms. E.g., assuming 2-body interactions limits the sum to $O(n^2)$ Pauli terms.[1]

2 Gibbs states

Ground states and ground state energies often reveal useful information about a system. However, it can be argued that "thermal states" or "Gibbs states" are more relevant and prevalent in nature. The Gibbs state of a Hamiltonian H at temperature T is described by the density matrix

$$\rho_T = \frac{1}{Z_T} e^{-H/T} = \frac{1}{Z_T} \sum_i e^{-\lambda_i/T} |\phi_i\rangle \langle\phi_i| ,$$

where $Z_T = \sum_i e^{-\lambda_i/T}$ is a normalization constant (called the "partition function"). It corresponds to the equilibrium state at temperature T of a system with Hamiltonian H. Note that this is a ground state in the $T \to 0$ limit (i.e., $\lim_{T\to 0} \rho_T = |\phi_0\rangle \langle\phi_0|$).

2.1 Metropolis algorithm

First we consider a classical (i.e., diagonal) Hamiltonian of the form $H = \sum_x E(x) |x\rangle \langle x|$. The Gibbs state is then

$$\rho_T = \frac{1}{Z_T} \sum_{x\in\{0,1\}^n} e^{-E(x)/T} |x\rangle \langle x| .$$

This corresponds to a *classical* distribution over bit strings x.

For many decades, the Metropolis algorithm has been the go-to algorithm for such Gibbs states. To get some intuition as to why preparing ρ_T is nontrivial, note that we can typically evaluate the *relative* weight or probability $e^{-E(x)/T}$ of a string x, but evaluating the *absolute* probability $\frac{1}{Z_T} e^{-E(x)/T}$ requires knowledge of the partition function Z_T. Evaluating the partition function is a notoriously hard problem (e.g., naively evaluating the sum requires time 2^n).

The Metropolis algorithm avoids this problem in a clever way. It is a Markov chain algorithm, in that it encodes the goal distribution as the stationary distribution of some Markov chain over the bit strings.[2] Here a Markov chain is specified by transition probabilities $P(y|x)$ for $x, y \in \{0,1\}^n$, and a distribution π is called a *stationary distribution* if

$$\pi(x) = \sum_y P(x|y)\pi(y).$$

The Metropolis algorithm starts from an arbitrary, symmetric "proposal" Markov chain with transition probabilities $Q(x'|x) = Q(x|x')$. A typical example is the chain that picks x' by flipping a random bit of x, so that $Q(x'|x) = 1/n = Q(x|x')$ if x and x' differ in a single bit, and $Q(x'|x) = 0$ otherwise. The resulting chain has a uniform stationary distribution. The Metropolis algorithm tweaks the transition probabilities so that the stationary distribution becomes the Gibbs state. A single step of the algorithm goes as follows:

1. From x_t, generate a candidate x'_t with probability $Q(x'_t|x_t)$.
2. Set $x_{t+1} = x'_t$ with acceptance probability $A(x'_t|x_t) = \min\left\{1, e^{-E(x'_t)/T}/e^{-E(x_t)/T}\right\}$. Otherwise, reject the move and set $x_{t+1} = x_t$.

[1] The Max Cut Hamiltonian from last lecture is an example of this.

[2] This is similar in spirit to the Feynman Hamiltonian, which encodes a computation in the ground state of a Hamiltonian.

Effectively, the algorithm will reject with high probability moves that increase the energy too much, while it always accepts moves that decrease the energy. This increases the overall probability of being in a low energy state, as is the case for the Gibbs state. In the exercises, you will show that the Gibbs state is a stationary distribution of the Metropolis algorithm. In fact, under mild conditions this will be the *unique* stationary distribution of the Markov chain, and moreover the distribution of the Markov chain will converge to π as time goes by.

2.2 Quantum Metropolis algorithm

The quantum Metropolis algorithm is a generalization of the Metropolis algorithm to the quantum domain. The main hurdle to overcome is this: quantum Hamiltonians H are typically not diagonal, and so the corresponding Gibbs state are distributions over *quantum* states (more specifically, over eigenvectors $|\phi_i\rangle$ of H) rather than bit strings.

Similar to the Metropolis algorithm, we wish to start from an eigenstate of the Hamiltonian H. We can do so easily by picking an arbitrary initial state such as $|\psi_0\rangle = |0^n\rangle$, performing quantum phase estimation and measuring the phase estimation register. If $|\psi_0\rangle = \sum \alpha_i |\phi_i\rangle$, with $(|\phi_i\rangle, \lambda_i)$ the eigenpairs of H, then by our discussion in Section 1 this returns a random state[3]

$$|\phi_i\rangle |\lambda_i\rangle$$

with probability $|\alpha_i|^2$.

The quantum Metropolis algorithm starts from such a state with two additional registers, $|\phi_i\rangle |\lambda_i\rangle |0\rangle |0\rangle$. One step of the algorithm goes as follows:

1. Apply a "proposal" unitary U to the first register:

$$|\phi_i\rangle |\lambda_i\rangle |0\rangle |0\rangle \xrightarrow{U} (U |\phi_i\rangle) |\lambda_i\rangle |0\rangle |0\rangle = \sum_{k=0}^{2^n-1} \alpha_k^i |\phi_k\rangle |\lambda_i\rangle |0\rangle |0\rangle .$$

2. Use quantum phase estimation to estimate the new energies:

$$\sum_k \alpha_k^i |\phi_k\rangle |\lambda_i\rangle |0\rangle |0\rangle \xrightarrow{\text{QPE}} \sum_k \alpha_k^i |\phi_k\rangle |\lambda_i\rangle |\lambda_k\rangle |0\rangle$$

3. In superposition, calculate the acceptance probability $A(\lambda_k|\lambda_i) = \min\{1, e^{-(\lambda_k-\lambda_i)/T}\}$ and rotate the last qubit over an angle $\sqrt{A(\lambda_k|\lambda_i)}$:

$$\sum_k \alpha_k^i |\phi_k\rangle |\lambda_i\rangle |\lambda_k\rangle |0\rangle \xrightarrow{W} \sum_k \alpha_k^i |\phi_k\rangle |\lambda_i\rangle |\lambda_k\rangle \left(\sqrt{A(\lambda_k|\lambda_i)} |1\rangle + \sqrt{1 - A(\lambda_k|\lambda_i)} |0\rangle\right).$$

4. Measure the last qubit.

   (a) If "1" (accept), the resulting state is

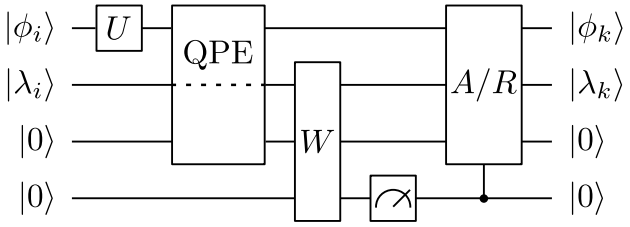
$$\sum_k \tilde{\alpha}_k^i |\phi_k\rangle |\lambda_i\rangle |\lambda_k\rangle |1\rangle$$

   with $\tilde{\alpha}_k^i \propto \alpha_k^i \sqrt{A(\lambda_k|\lambda_i)}$. Measure the second energy register. This returns state $|\phi_k\rangle |\lambda_i\rangle |\lambda_k\rangle |1\rangle$ with probability $|\alpha_k^i|^2 A(\lambda_k|\lambda_i)$. By shuffling and erasing some registers, we can transform this to the new initial state $|\phi_k\rangle |\lambda_k\rangle |0\rangle |0\rangle$.

[3]For simplicity, we assume that quantum phase estimation is exact: $\tilde{\lambda}_i = \lambda_i$.

(b) If "0" (reject), we wish to return to the original state $|\phi_i\rangle\,|\lambda_i\rangle\,|0\rangle\,|0\rangle$. While highly nontrivial, there exists a quantum routine to achieve this, but we will not detail it here.

The following circuit summarizes the algorithm:



The algorithm effectively implements the classical Metropolis algorithm on the eigenstates $\{|\phi_i\rangle\}$ of the Hamiltonian. Starting from a state $|\phi_i\rangle\,|\lambda_i\rangle$ it returns a state $|\phi_k\rangle\,|\lambda_k\rangle$ with transition probability $P(k|i) = |\alpha_k^i|^2 A(\lambda_k|\lambda_i)$.

Exercise 1. *What condition should we impose on the mixer unitary U (equivalently, on the coefficients α_k^i) so that we recover a Metropolis algorithm?*