

Proof-of-concept Quantum Use Case



Simulation

In the current **NISQ (noisy intermediate-scale quantum) era**, the most feasible and promising use cases for quantum computing are optimization, simulation, and machine learning.

Optimization

From minimizing risk to maximizing throughput, optimization problems are ubiquitous across many industries, especially in *finance*, *logistics*, and *engineering*.

Simulation

Simulating quantum systems is a notoriously difficult but powerful tool for solving problems in *chemistry*, *physics*, *material science*, and the *pharmaceutical industry*.

Machine Learning

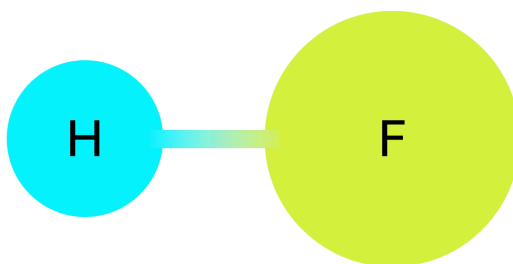
From *anomaly detection* and *classification problems* to *generating text and images*, machine learning has found countless applications touching almost every sector of society.

To tackle problems in these areas, most NISQ algorithms take a hybrid approach by using *classical optimization techniques* to tune the parameters of *variational quantum circuits*. In this proof-of-concept use case, we will consider a **simulation problem in chemistry** and then walk through, step by step, a variational quantum algorithm which solves the problem.

This proof-of-concept use case is part of a series produced by DeiC's Quantum Department as part of the Q-Access initiative to make quantum use-case development more accessible to Danish academia and industry. You can find the proof-of-concept use cases for **optimization** and **machine learning** at <https://www.deic.dk/q-access>. We will assume a basic familiarity with the quantum circuit model of quantum computing, but we will also try to give as many details as possible.

Problem: Estimating the Ground-State Energy of a Small Molecule

Suppose that we have a small molecule with a particular geometry, and we would like to calculate its lowest energy level, or **ground-state energy**. In chemistry, this is an essential reference for quantitative predictions of molecular stability, reactivity, and spectroscopic signatures. In this case, we will consider the extremely dangerous hydrogen flouride atom HF at its equilibrium bond length, depicted below.



We will consider the Bohr-Oppenheimer approximation, which treats the nuclei as fixed and treats the electrons as dynamic, and we will set the bond length between the hydrogen and flourine atoms to the equilibrium bond length. With this setup, there is a variational quantum algorithm called a **variational quantum Eigensolver (VQE)**, which can estimate the ground-state energy. VQE was originally proposed in [1] and can be applied to a variety of problems ranging from the chemistry problem we are considering to problems in optimization and machine learning.

Precise Formulation of Our Problem

A **Hamiltonian** H in quantum mechanics is an operator that represents the total energy of a system. From a “spectral decomposition” of the Hamiltonian, one can find the possible energy levels of the system, which will be a finite set $\{E_0, E_1, E_2, \dots, E_n\}$ typically ordered from smallest to largest. The **ground state** $|\Psi_0\rangle$ of a quantum system is the lowest energy state, which will have energy E_0 and satisfy

$$H|\Psi_0\rangle = E_0|\Psi_0\rangle. \quad (1)$$

The **excited states** $\{|\Psi_1\rangle, |\Psi_2\rangle, \dots, |\Psi_n\rangle\}$ are the states with energy greater than the ground state, and they satisfy $H|\Psi_i\rangle = E_i|\Psi_i\rangle$.

So, if we want find the ground state and corresponding ground-state energy of our atomic system, we must find its spectral decomposition, also known as its “eigendecomposition.” Indeed, this problem is a special case of the more general problem of finding **eigenvalues** (corresponding to the energy levels) and **eigenvectors** (corresponding to the states). Finding the full eigendecomposition is computationally expensive, but we are only looking for the smallest eigenvalue and its corresponding eigenvector, which is exactly what a variational quantum eigensolver (VQE) is designed to do.

VQE can be applied to wide range of Hamiltonians, and for the purpose of this tutorial we will assume that the Hamiltonian is given, though we will give a sketch of how it can be obtained. For hydrogen fluoride, as mentioned above, we will consider the Bohr-Oppenheimer (BO) approximation, which treats the nuclei as fixed and the electrons as dynamic. In fact, we will consider only the “electronic structure” Hamiltonian of our system and not the full rotational, vibrational, and nuclear-spin problem. However, the BO electronic structure Hamiltonian allows the electrons to vary continuously, which is not immediately compatible with the gate-based quantum computing model. To discretize the problem so that it can be mapped to a qubit system, we will consider the “Jordan-Wigner map of the second-quantization,” which gives us an operator acting on a finite orbital basis. Each orbital will then corresponding to a pair of qubits, with one qubit representing the presence of a spin-up electron and the other a spin-down electron.

To obtain this form of the Hamiltonian for H^{HF} from our problem, we will use the Qiskit Nature library [2]. For this we only need to specify the 3-dimensional coordinates of the hydrogen and flourine atoms in our system, which will be given in angstroms (Å) as follows.

Atom	x	y	z
H	0.0	0.0	0.0
F	0.0	0.0	0.9168

With this geometry, the hydrogen and flourine are 0.9168Å apart, which is the equilibrium bond length according to the NIST Computational Chemistry Comparison and Benchmark DataBase [3]. The full expression for H^{HF} is given in terms of 631 Pauli strings on 12 qubits, which is too long to display here, but its general form is given below along with a few terms.

$$H^{\text{HF}} = \sum_{P_i \in \{I, X, Y, Z\}^{\otimes 12}} c_i P_i = \begin{cases} -65.947 I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \\ +16.182 I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes Z \\ +2.338 I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes Z \otimes I \\ \vdots \\ +0.004 Y \otimes Z \otimes Z \otimes Z \otimes Z \otimes Y \otimes Y \otimes Z \otimes Z \otimes Z \otimes Y \otimes I \\ \vdots \\ +0.155 Z \otimes Z \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \end{cases} \quad (2)$$

Flourine has 5 spatial orbitals and Hydrogen has 1. For each spatial orbital, there is either a spin-up or spin-down electron, which is why the above Hamiltonian acts on 12 qubits. Now, since we are considering neutral HF, Flourine will contribute 9 electrons and Hydrogen 1 for 10 total electrons. Moreover, since HF is a “spin singlet” it will have an equal number of spin-up and spin-down electrons. If we just filled up the orbitals with 5 spin-up and 5 spin-down electrons in order of their energy from lowest to highest in the basis provided by Qiskit Nature, we would obtain the Hartee-Fock state, given below.

$$\Psi^{\text{Hartree-Fock}} = |111110111110\rangle \quad (3)$$

Now, to solve our problem, we need to find an approximation of ground state $|\Psi_0^{\text{HF}}\rangle$ and ground-state energy E_0^{HF} for H^{HF} , which satisfies

$$H^{\text{HF}}|\Psi_0^{\text{HF}}\rangle = E_0^{\text{HF}}|\Psi_0^{\text{HF}}\rangle. \quad (4)$$

The Hartree-Fock state is generally not the ground state of the full interacting Hamiltonian, but it will serve as a good starting point for our approximation.

Motivation from Variational Quantum Simulation

The variational principle is the basis for VQE and is a useful tool for establishing upper bounds on ground-state energies and, along with classical optimization of tunable parameters, can be used to approximate ground-state energies.

Variational Principle

For a Hamiltonian H that describes the a quantum system and a “trial state” $|\Psi\rangle$ with $\langle\Psi|\Psi\rangle = 1$, the ground-state energy E_0 is always less than or equal to the expectation value of the energy of the state of $|\Psi\rangle$. That is, we have the following.

$$E_0 \leq \langle\Psi|H|\Psi\rangle \quad (5)$$

Moreover, if we have

$$E_0 = \langle\Psi|H|\Psi\rangle \quad (6)$$

then $|\Psi\rangle = |\Psi_0\rangle$ is the ground state of the system described by H . Thus, by **varying** Ψ until the expectation value $\langle\Psi|H|\Psi\rangle$ is minimized, we can obtain approximations of the ground-state and the ground-state energy.

Now, we want to consider a tunable trial state, so we will introduce parameters $\theta = (\theta_1, \dots, \theta_N)$ and consider a state $|\Psi(\theta)\rangle$ that depends on these parameters. This in turn gives us a tunable energy function

$$E(\theta) = \langle\Psi(\theta)|H|\Psi(\theta)\rangle. \quad (7)$$

Our problem then becomes to find an approximation of the set of parameters which minimizes this energy function, i.e.

$$\theta_{\text{optimal}} \approx \underset{\theta}{\text{argmin}} E(\theta). \quad (8)$$

This optimal set of parameters gives us an approximation of the ground-state energy

$$E_0^{\text{HF}} \approx E(\theta_{\text{optimal}}) \quad (9)$$

and an approximation of the ground-state

$$|\Psi_0^{\text{HF}}\rangle \approx |\Psi(\theta_{\text{optimal}})\rangle. \quad (10)$$

In order to prepare such a tunable state for VQE, we will use a variational quantum circuit.

Variational Quantum Eigensolver (VQE)

The VQE algorithm is a straightforward implementation of the variational principle discussed above.

Outline of VQE

Goal: Find an approximation of the ground-state energy E_0 of a given Hamiltonian H .

Strategy

1. Construct a variational quantum circuit $Q(\theta)$ depending on N parameters $\theta = (\theta_1, \dots, \theta_N)$.
2. Optimize the parameters θ , by repeatedly running the circuit from Step 1 to estimate the energy

$$E(\theta) = \langle \Psi(\theta) | H | \Psi(\theta) \rangle. \quad (11)$$

where

$$|\Psi(\theta)\rangle = Q(\theta)|0\rangle \quad (12)$$

and by updating the parameters to lower the energy.

3. Estimate the ground-state energy E_0 by sampling the final optimized circuit.

There are numerous classical optimization algorithms which could be used for Step 2 above, but we will not delve into these and instead rely on existing implementations from the open-source Python library SciPy [4]. However, this leaves a big question unanswered. How do we construct an appropriate variational quantum circuit $Q(\theta)$? There are several potential considerations, and we will discuss two possible variational forms, or **ansatzes**, for our quantum circuit below.

Finally, in variational methods like VQE ideally we start with a state that is already a decent approximation. For our problem of approximating the ground-state energy of HF, we will start with the Hartree-Fock approximation of HF from Equation (3).

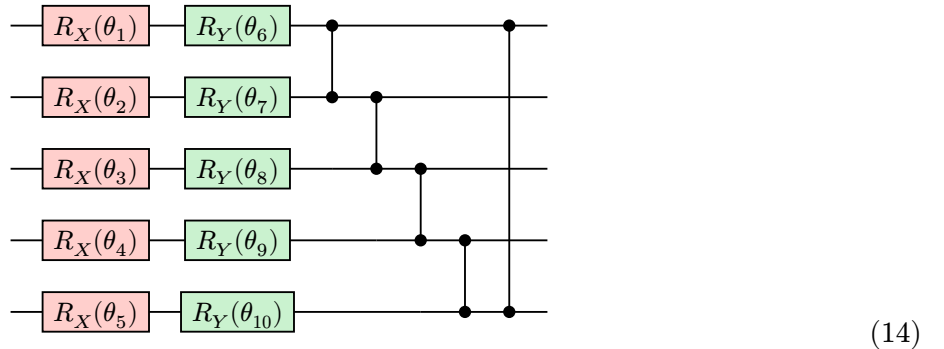
Constructing a Quantum Circuit for VQE

The variational quantum circuit for preparing the VQE ansatz $|\Psi(\theta)\rangle$ will have the following general form

$$|00\dots 0\rangle \Rightarrow \text{Initialize } |\Psi(0)\rangle \Rightarrow \boxed{\text{Layer 1} \Rightarrow \text{Layer 2} \Rightarrow \dots \Rightarrow \text{Layer } k} \Rightarrow |\Psi(\theta)\rangle \quad (13)$$

Ansatz circuit depending on θ

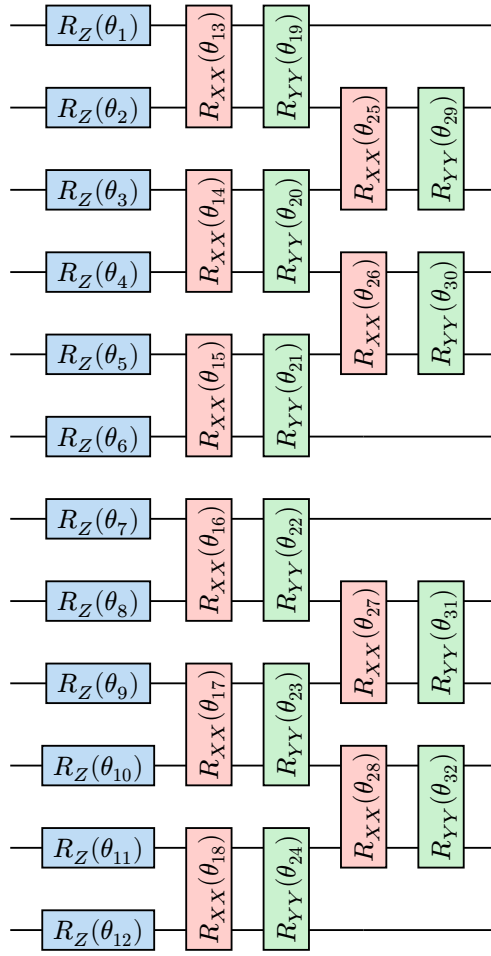
where the ansatz circuit comprises k layers. Given the depth limitations of current quantum computers, one of the most common considerations when choosing an ansatz is hardware efficiency. A typical hardware-efficient ansatz consists of parametrized single qubit gates and a fixed entangler. Below is an example of a hardware-efficient ansatz layer for 5 qubits, which uses 2 parameters per qubit per layer.



For many problems, including ours, there may be problem-specific considerations that could influence our choice of ansatz. Since we are considering the electronic structure of a fixed molecule HF, there will always be the same number of spin-up and spin-down electrons.

For HF, there will always be 5 spin-up electrons and 5 spin-down electrons. The following ansatz layer for 12 qubits has 32 parameters and preserves the number of 1's in the first 6 qubits and the number of 1's in

the last 6 qubits, which exactly preserves the spin-up and spin-down qubits in the orbital basis provided by Qiskit Nature [2].



(15)

This second ansatz is the one we will implement below in Qiskit.

Qiskit Implementation of VQE for Our Problem

First, we need to import everything we need from Qiskit [5], Qiskit Nature [2], and SciPy [4].

```
1 from qiskit import QuantumCircuit
2 from qiskit.circuit import ParameterVector
3 from qiskit_aer.primitives import Estimator
4
5 from qiskit_nature.second_q.drivers import PySCFDriver
6 from qiskit_nature.second_q.mappers import JordanWignerMapper
7 from qiskit_nature.second_q.circuit.library import HartreeFock
8
9 from scipy.optimize import minimize
```

Then, we can input our problem as a string describing the geometry of HF with its equilibrium bond length and specify the number of layers in our ansatz.

```
10 HF_equilibrium_geometry = "H 0.0 0.0 0.0; F 0.0 0.0 0.9168"
11
12 layers = 2
```

Then, with Qiskit Nature, we can obtain the Jordan-Wigner map of the second quantization of the Hamiltonian for HF.

```

13 driver = PySCFDriver(atom=HF_equilibrium_geometry)
14 electronic_structure_problem = driver.run()
15
16 second_quantized_Hamiltonian = electronic_structure_problem.hamiltonian.second_q_op()
17
18 mapper = JordanWignerMapper()
19 HF_Hamiltonian = mapper.map(second_quantized_Hamiltonian)

```

Next, we use Qiskit Nature to create a circuit for generating the Hartree-Fock state, which just requires a single layer of X -gates on the appropriate qubits.

```

20 hartree_fock_state = HartreeFock(
21     num_spatial_orbitals=6,
22     num_particles=(5,5),
23     qubit_mapper=mapper
24 )

```

Then, we can define a function for generating number-preserving ansatzes as described above.

```

25 def number_preserving_ansatz(n_qubits, n_layers, param_name="θ"):
26     n_exchange_per_layer = (n_qubits // 2) + ((n_qubits - 1) // 2)
27
28     total_params = n_layers * (n_exchange_per_layer + n_qubits)
29     params = ParameterVector(param_name, total_params)
30
31     qc = QuantumCircuit(n_qubits)
32
33     param_index = 0
34     for _ in range(n_layers):
35         for q in range(n_qubits):
36             qc.rz(params[param_index], q)
37             param_index += 1
38
39         # Even pairs
40         for i in range(0, n_qubits - 1, 2):
41             qc.rxx(params[param_index], i, i+1)
42             qc.ryy(params[param_index], i, i+1)
43             param_index += 1
44
45         # Odd pairs
46         for i in range(1, n_qubits - 1, 2):
47             qc.rxx(params[param_index], i, i+1)
48             qc.ryy(params[param_index], i, i+1)
49             param_index += 1
50
51     return qc, params

```

With the Hartree-Fock state preparation circuit and the number-preserving ansatz, we can build the VQE circuit using one number-preserving ansatz for the spin-up electrons and one for the spin-down electrons.

```

52 VQE_qc = QuantumCircuit(12)
53
54 VQE_qc.compose(hartree_fock_state, inplace=True)
55
56 ansatz = QuantumCircuit(12)
57 ansatz_A, params_A = number_preserving_ansatz(6, 2, param_name="θ_A")

```

```

58 ansatz_B, params_B = number_preserving_ansatz(6, 2, param_name="θ_B")
59
60 ansatz.compose(ansatz_A, qubits=[0,1,2,3,4,5], inplace=True)
61 ansatz.compose(ansatz_B, qubits=[6,7,8,9,10,11], inplace=True)
62
63 VQE_qc.compose(ansatz, inplace=True)
64 VQE_qc.measure_all()

```

Using the variational VQE circuit and the Hamiltonian for HF, we can estimate the energy corresponding to a given set of parameters.

```

65 estimator = Estimator()
66
67 def estimate_energy(params):
68     val = estimator.run(VQE_qc, HF_Hamiltonian,
69         parameter_values=params).result().values[0]
69     return val

```

Finally, we can use SciPy [4] to optimize the parameters and obtain our approximation of the ground-state energy.

```

70 initial_params = 0.05 * standard_normal(len(params_A) + len(params_B))
71 optimum = minimize(energy, x0=initial_params, method="COBYLA")
72
73 optimal_params = optimum.x
74 ground_state_energy = energy(optimal_params)

```

In the next section we will see how close this approximation is.

Results

From the VQE implementation in the above Qiskit code, we obtain the following estimate for the ground-state energy.

$$E_0^{\text{VQE}} = \langle \Psi_0^{\text{VQE}} | H^{\text{HF}} | \Psi_0^{\text{VQE}} \rangle \approx -103.764 \quad (16)$$

where $|\Psi_0^{\text{VQE}}\rangle = |\Psi^{\text{VQE}}(\theta_{\text{optimal}})\rangle = Q^{\text{VQE}}(\theta_{\text{optimal}})|0\rangle$. In this case, we can also compute E_0^{HF} and $|\Psi_0^{\text{HF}}\rangle$ exactly and see how well our implementation of VQE performed.

$$E_0^{\text{HF}} = \langle \Psi_0^{\text{HF}} | H^{\text{HF}} | \Psi_0^{\text{HF}} \rangle \approx -103.791 \quad (17)$$

So, our approximation is quite good without having to tune any meta-parameters! We can also compute the fidelity of our approximation for the ground state, which is given as follows.

$$|\langle \Psi_0^{\text{VQE}} | \Psi_0^{\text{HF}} \rangle|^2 \approx 0.983 \quad (18)$$

A perfect fidelity would be 1, so the fidelity is also quite good.

References

- [1] A. Peruzzo *et al.*, “A variational eigenvalue solver on a quantum processor,” *Nature Communications*, vol. 5, 4213, Jul. 2014, doi: [10.1038/ncomms5213](https://doi.org/10.1038/ncomms5213).
- [2] The Qiskit Nature developers and contributors, *Qiskit Nature 0.6.0 (0.6.0)*. doi: [10.5281/zenodo.7828767](https://doi.org/10.5281/zenodo.7828767).
- [3] NIST Computational Chemistry Comparison and Benchmark DataBase, “Experimental data for HF (Hydrogen fluoride).” [Online]. Available: <https://cccbdb.nist.gov/exp2x.asp?casno=7664393&charge=0>
- [4] P. Virtanen *et al.*, “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python,” *Nature Methods*, vol. 17, no. 3, pp. 261–272, Feb. 2020, doi: [10.1038/s41592-019-0686-2](https://doi.org/10.1038/s41592-019-0686-2).
- [5] A. Javadi-Abhari *et al.*, “Quantum computing with Qiskit,” *arXiv*, May 2024, doi: [10.48550/arXiv.2405.08810](https://doi.org/10.48550/arXiv.2405.08810).
- [6] P. J. J. O’Malley *et al.*, “Scalable quantum simulation of molecular energies,” *Physical Review X*, vol. 6, 31007, Jul. 2016, doi: [10.1103/PhysRevX.6.031007](https://doi.org/10.1103/PhysRevX.6.031007).

Appendix: Data and Resources

The Python code implementing the VQE algorithm tailored to this problem using Qiskit [5], Qiskit Nature [2], and SciPy [4] is available at <https://github.com/DeiC-Quantum/Proof-of-concept-Quantum-Use-Cases>.