Quantum Computing, Quantum Machine Learning and Quantum Information Theories

Morten Hjorth-Jensen[1,2]

Department of Physics, University of Oslo[1]

Department of Physics and Astronomy and Facility for Rare Isotope Beams, Michigan State University[2]

February 19-23, 2024

Solving quantum mechanical problems

1. Repetition from last week and discussion of the first part of the project
2. Further discussions of gates and measurements
3. Introducing the Variational Quantum Eigensolver (VQE) and discussion of project 1
4. **Note on project work:** feel free to collaborate on the projects
5. Video of lecture
6. Whiteboard notes

Paths for project 2

Depending on your interests, we can tailor project 2 in several different ways

1. discussions and work on other algorithms than those used for solving quantum mechanical problems, Grover, Shor, Quantum Fourier Transforms etc
2. work on quantum machine learning (could combine with FYS5429)
3. other possibilities

Readings

1. For the discussion of one-qubit, two-qubit and other gates, sections 2.6-2.11 and 3.1-3.4 of Hundt's book **Quantum Computing for Programmers**, contain most of the relevant information. We will repeat some of the these properties during the first part of the lecture.
2. The VQE algorithm is discussed in Hundt's section 6.11
3. See also the VQE review article by Tilly et al.

Gates and measurements

This material is covered by whiteboard notes from lecture February 21. The material will be updated later this week.

The Qiskit material from last week

```
# Install alternative provider package
# If you have problems with the original notebook, try this package
# pip install -q qiskit-ibm-provider
# How to use it
from qiskit_ibm_provider import IBMProvider

api_token = 'Your token'

# Load your IBM Quantum account
IBMProvider.save_account(api_token, overwrite=True)  # This stores you
provider = IBMProvider()

import qiskit as qk
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
from qiskit.visualization import plot_histogram
from qiskit_aer import AerSimulator

import matplotlib.pyplot as plt
# Then go ahead and use it like this with whatever backend, i.e. osaka
backend = provider.get_backend('ibm_osaka')

# Create a quantum circuit with two qubits
bell_circuit = QuantumCircuit(2, 2)
```

Variational Quantum Eigensolver

One initial algorithm to estimate the eigenenergies of a quantum Hamiltonian was quantum phase estimation. In it, one encodes the eigenenergies, one binary bit at a time (up to n bits), into the complex phases of the quantum states of the Hilbert space for n qubits. It does this by applying powers of controlled unitary evolution operators to a quantum state that can be expanded in terms of the Hamiltonian's eigenvectors of interest. The eigenenergies are encoded into the complex phases in such a way that taking the inverse quantum Fourier transformation (see Hundt sections 6.1-6.2) of the states into which the eigen-energies are encoded results in a measurement probability distribution that has peaks around the bit strings that represent a binary fraction which corresponds to the eigen-energies of the quantum state acted upon by the controlled unitary operators.

The VQE

While quantum phase estimation (QPE) is provably efficient, non-hybrid, and non-variational, the number of qubits and length of circuits required is too great for our NISQ era quantum computers. Thus, QPE is only efficiently applicable to large, fault-tolerant quantum computers that likely won't exist in the near, but the far future.
Therefore, a different algorithm for finding the eigen-energies of a quantum Hamiltonian was put forth in 2014 called the variational quantum eigensolver, commonly referred to as VQE. The algorithm is hybrid, meaning that it requires the use of both a quantum computer and a classical computer. It is also variational, meaning that it relies, ultimately, on solving an optimization problem by varying parameters and thus is not as deterministic as QPE. The variational quantum eigensolver is based on the variational principle:

Expectation value of Hamiltonian

The expectation value of a Hamiltonian H in a state $|\psi(\theta)\rangle$ parameterized by a set of angles θ, is always greater than or equal to the minimum eigen-energy E_0. To see this, let $|n\rangle$ be the eigenstates of H, that is

$$H|n\rangle = E_n|n\rangle.$$

Expanding in the eigenstates

We can then expand our state $|\psi(\theta)\rangle$ in terms of said eigenstates

$$|\psi(\theta)\rangle = \sum_n c_n |n\rangle,$$

and plug this into the expectation value to yield

$$\langle\psi(\theta)|H|\psi(\theta)\rangle = \sum_{nm} c_m^* c_n \langle m|H|n\rangle = \sum_{nm} c_m^* c_n E_n \langle m|n\rangle = \sum_{nm} \delta_{nm} c_m^* c_n E_n$$

which implies that we can minimize over the set of angles θ and arrive at the ground state energy E_0

$$\min_\theta \langle\psi(\theta)|H|\psi(\theta)\rangle = E_0.$$

Basic steps of the VQE algorithm

Using this fact, the VQE algorithm can be broken down into the following steps

1. Prepare the variational state $|\psi(\theta)\rangle$ on a quantum computer.
2. Measure this circuit in various bases and send these measurements to a classical computer
3. The classical computer post-processes the measurement data to compute the expectation value $\langle\psi(\theta)|H|\psi(\theta)\rangle$
4. The classical computer varies the parameters θ according to a classical minimization algorithm and sends them back to the quantum computer which runs step 1 again.

This loop continues until the classical optimization algorithm terminates which results in a set of angles θ_{min} that characterize the ground state $|\phi(\theta_{\text{min}})\rangle$ and an estimate for the ground state energy $\langle\psi(\theta_{\text{min}})|H|\psi(\theta_{\text{min}})\rangle$.

Expectation values

To execute the second step of VQE, we need to understand how expectation values of operators can be estimated via quantum computers by post-processing measurements of quantum circuits in different basis sets. To rotate bases, one uses the basis rotator R_σ which is defined for each Pauli gate σ to be

$$R_\sigma = H, \text{if } \sigma = X, \tag{1}$$

and

$$HS^\dagger, \text{if } \sigma = Y, \tag{2}$$

and

$$I, \text{if } \sigma = Z. \tag{3}$$

Measurements of eigenvalues of the Pauli operators

We can show that these rotations allow us to measure the eigenvalues of the Pauli operators. The eigenvectors of the Pauli X gate are

$$|\pm\rangle = \frac{|0\rangle \pm |1\rangle}{\sqrt{2}},$$

with eigenvalues ± 1. Acting on the eigenstates with the rotation in eq. (12) gives

$$H|+\rangle = +1|0\rangle,$$

and

$$H|-\rangle = -1|1\rangle.$$

Single-qubit states

Any single-qubit state can be written as a linear combination of these eigenvectors,

$$|\psi\rangle = \alpha|+\rangle + \beta|-\rangle.$$

We then have the following expectation value for the Pauli X operator

$$\langle|X|\rangle = \langle\psi|X|\psi\rangle = |\alpha|^2 - |\beta|^2.$$

However, we can only measure the qubits in the computational basis. Applying the rotation in eq. (12) to our state gives

$$H|\psi\rangle = \alpha|0\rangle - \beta|1\rangle.$$

Interpretations

This tells us that we are able to estimate $|\alpha|^2$ and $|\beta|^2$ (and hence the expectation value of the Pauli X operator) by using the rotation in eq. (12) and measure the resulting state in the computational basis. We can show this for the Pauli Z and Pauli Y similarly.

Reminder on rotations

Note the following identity of the basis rotator

$$R_\sigma^\dagger Z R_\sigma = \sigma,$$

which follows from the fact that $HZH = X$ and $SXS^\dagger = Y$.

Arbitrary Pauli gate

With this, we see that the expectation value of an arbitrary Pauli-gate σ in the state $|\psi\rangle$ can be expressed as a linear combination of probabilities

$$\begin{aligned} E_\psi(\sigma) &= \langle\psi|\sigma|\psi\rangle \\ &= \langle\psi|R_\sigma^\dagger Z R_\sigma|\psi\rangle = \langle\phi|Z|\phi\rangle \\ &= \langle\phi|\left(\sum_{x\in\{0,1\}}(-1)^x|x\rangle\langle x|\right)|\phi\rangle \\ &= \sum_{x\in\{0,1\}}(-1)^x|\langle x|\phi\rangle|^2 \\ &= \sum_{x\in\{0,1\}}(-1)^x P(|\phi\rangle\to|x\rangle), \end{aligned} \tag{4}$$

where $|\phi\rangle = |R_\sigma\phi\rangle$ and $P(|\phi\rangle \to |x\rangle$ is the probability that the state $|\phi\rangle$ collapses to the state $|x\rangle$ when measured.

Arbitrary string of Pauli operators

This can be extended to any arbitrary Pauli string: consider the string of Pauli operators $P = \bigotimes_{p\in Q} \sigma_p$ which acts non-trivially on the set of qubits Q which is a subset of the total set of n qubits in the system. Then

$$\begin{aligned}
E_\psi\left(P\right) &= \langle\psi| \left(\bigotimes_{p\in Q} \sigma_p\right) |\psi\rangle \\
&= \langle\psi| \left(\bigotimes_{p\in Q} \sigma_p\right) \left(\bigotimes_{q\notin Q} I_q\right) |\psi\rangle \\
&= \langle\psi| \left(\bigotimes_{p\in Q} R^\dagger_{\sigma_p} Z_p R_{\sigma_p}\right) \left(\bigotimes_{q\notin Q} I_q\right) |\psi\rangle \\
&= \langle\psi| \left(\bigotimes_{p\in Q} R^\dagger_{\sigma_p}\right) \left(\bigotimes_{p\in Q} Z_p\right) \left(\bigotimes_{q\notin Q} I_q\right) \left(\bigotimes_{p\in Q} R_{\sigma_p}\right) |\psi\rangle
\end{aligned}$$

Which gives us

$$\begin{aligned}
E_\psi\left(P\right) &= \langle\phi| \left(\bigotimes_{p\in Q} Z_p\right) \left(\bigotimes_{q\notin Q} I_q\right) |\phi\rangle \\
&= \langle\phi| \left(\bigotimes_{p\in Q} \sum_{x_p\in\{0_p,1_p\}} (-1)^{x_p} |x_p\rangle\langle x_p|\right) \left(\bigotimes_{q\notin Q} \sum_{y_q\in\{0_q,1_q\}} |y_q\rangle\langle y_q|\right) |\phi \\
&= \langle\phi| \left(\sum_{x\in\{0,1\}^n} (-1)^{\sum_{p\in Q} x_p} |x\rangle\langle x|\right) |\phi\rangle \\
&= \sum_{x\in\{0,1\}^n} (-1)^{\sum_{p\in Q} x_p} |\langle x||\phi\rangle|^2 \\
&= \sum_{x\in\{0,1\}^n} (-1)^{\sum_{p\in Q} x_p} P(|\phi\rangle \to |x\rangle), \qquad (5)
\end{aligned}$$

where $|\phi\rangle = |\bigotimes_{p\in Q} R_{\sigma_p}\psi\rangle$.

Final observables

Finally, because the expectation value is linear

$$E_\psi\left(\sum_m \lambda_m P_m\right) = \sum_m \lambda_m E_\psi(P_m),$$

one can estimate any observable that can be written as a linear combination of Pauli-string terms.

Measurement

To estimate the probability $P(|\phi\rangle \to |x\rangle)$ from the previous results, one prepares the state $|\phi\rangle$ on a quantum computer and measures it, and then repeats this process (prepare and measure) several times. The probability $P(|\phi\rangle \to |x\rangle)$ is estimated to be the number of times that one measures the bit-string x divided by the total number of measurements that one makes; that is

$$P(|\phi\rangle \to |x\rangle) \approx \sum_{m=1}^{M} \frac{x_m}{M},$$

where $x_m = 1$ if the result of measurement is x and 0 if the result of measurement is not x.

Law of large numbers aka Bernoulli's theorem

By the law of large numbers the approximation approaches equality as M goes to infinity

$$P(|\phi\rangle \to |x\rangle) = \lim_{M\to\infty} \sum_{m=1}^{M} \frac{x_m}{M}.$$

As we obviously do not have infinite time nor infinite quantum computers (which could be run in parallel), we must truncate our number of measurement M to a finite, but sufficiently large number. More precisely, for precision ϵ, each expectation estimation subroutine within VQE requires $\mathcal{O}(1/\epsilon^2)$ samples from circuits with depth $\mathcal{O}(1)$.

Implementing the VQE method, one qubit system

We start with a simple 2×2 Hamiltonian matrix expressed in terms of Pauli X and Z matrices, as discussed in the project text.
We define a symmetric matrix $H \in \mathbb{R}^{2\times 2}$

$$H = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix},$$

Rewriting the Hamiltonian

We let $H = H_0 + H_I$, where

$$H_0 = \begin{bmatrix} E_1 & 0 \\ 0 & E_2 \end{bmatrix},$$

is a diagonal matrix. Similarly,

$$H_I = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix},$$

where V_{ij} represent various interaction matrix elements. We can view H_0 as the non-interacting solution

$$H_0|0\rangle = E_1|0\rangle, \tag{6}$$

and

$$H_0|1\rangle = E_2|1\rangle, \tag{7}$$

where we have defined the orthogonal computational one-qubit basis states $|0\rangle$ and $|1\rangle$.

Using Pauli matrices

We rewrite H (and H_0 and H_I) via Pauli matrices

$$H_0 = \mathcal{E}I + \Omega\sigma_z, \quad \mathcal{E} = \frac{E_1 + E_2}{2},\ \Omega = \frac{E_1 - E_2}{2},$$

and

$$H_I = c\boldsymbol{I} + \omega_z\sigma_z + \omega_x\sigma_x,$$

with $c = (V_{11} + V_{22})/2$, $\omega_z = (V_{11} - V_{22})/2$ and $\omega_x = V_{12} = V_{21}$. We let our Hamiltonian depend linearly on a strength parameter λ

$$H = H_0 + \lambda H_{\mathrm{I}},$$

with $\lambda \in [0, 1]$, where the limits $\lambda = 0$ and $\lambda = 1$ represent the non-interacting (or unperturbed) and fully interacting system, respectively. The model is an eigenvalue problem with only two available states.

Selecting parameters

Here we set the parameters $E_1 = 0$, $E_2 = 4$, $V_{11} = -V_{22} = 3$ and $V_{12} = V_{21} = 0.2$.
The non-interacting solutions represent our computational basis. Pertinent to our choice of parameters, is that at $\lambda \geq 2/3$, the lowest eigenstate is dominated by $|1\rangle$ while the upper is $|0\rangle$. At $\lambda = 1$ the $|0\rangle$ mixing of the lowest eigenvalue is 1% while for $\lambda \leq 2/3$ we have a $|0\rangle$ component of more than 90%. The character of the eigenvectors has therefore been interchanged when passing $z = 2/3$. The value of the parameter V_{12} represents the strength of the coupling between the two states.

Setting up the matrix

```
from  matplotlib import pyplot as plt
import numpy as np
dim = 2
Hamiltonian = np.zeros((dim,dim))
e0 = 0.0
e1 = 4.0
Xnondiag = 0.20
Xdiag = 3.0
Eigenvalue = np.zeros(dim)
# setting up the Hamiltonian
Hamiltonian[0,0] = Xdiag+e0
Hamiltonian[0,1] = Xnondiag
Hamiltonian[1,0] = Hamiltonian[0,1]
Hamiltonian[1,1] = e1-Xdiag
# diagonalize and obtain eigenvalues, not necessarily sorted
EigValues, EigVectors = np.linalg.eig(Hamiltonian)
permute = EigValues.argsort()
EigValues = EigValues[permute]
# print only the lowest eigenvalue
print(EigValues[0])
```

Now rewrite it in terms of the identity matrix and the Pauli matrix X and Z

```
# Now rewrite it in terms of the identity matrix and the Pauli matrix
X = np.array([[0,1],[1,0]])
Y = np.array([[0,-1j],[1j,0]])
Z = np.array([[1,0],[0,-1]])
```

Implementing the VQE

For a one-qubit system we can reach every point on the Bloch sphere (as discussed earlier) with a rotation about the x-axis and the y-axis.
We can express this mathematically through the following operations (see whiteboard for the drawing), giving us a new state $|\psi\rangle$

$$|\psi\rangle = R_y(\phi)R_x(\theta)|0\rangle.$$

Possible ansatzes

We can produce multiple ansatzes for the new state in terms of the angles θ and ϕ. With these ansatzes we can in turn calculate the expectation value of the above Hamiltonian, now rewritten in terms of various Pauli matrices (and thereby gates), that is compute

$$\langle\psi|(c+\mathcal{E})\boldsymbol{I}+(\Omega+\omega_z)\boldsymbol{\sigma}_z+\omega_x\boldsymbol{\sigma}_x|\psi\rangle.$$

We can now set up a series of ansatzes for $|\psi\rangle$ as function of the angles θ and ϕ and find thereafter the variational minimum using for example a gradient descent method.

More on rotation operators

To do so, we need to remind ourselves about the mathematical expressions for the rotational matrices/operators.

$$R_x(\theta) = \cos\frac{\theta}{2}\boldsymbol{I} - \imath\sin\frac{\theta}{2}\boldsymbol{\sigma}_x,$$

and

$$R_y(\phi) = \cos\frac{\phi}{2}\boldsymbol{I} - \imath\sin\frac{\phi}{2}\boldsymbol{\sigma}_y.$$

Code example

```
# define the rotation matrices
# Define angles theta and phi
theta = 0.5*np.pi; phi = 0.2*np.pi
Rx = np.cos(theta*0.5)*I-1j*np.sin(theta*0.5)*X
Ry = np.cos(phi*0.5)*I-1j*np.sin(phi*0.5)*Y
#define basis states
basis0 = np.array([1,0])
basis1 = np.array([0,1])

NewBasis = Ry @ Rx @ basis0
print(NewBasis)
# Compute the expectation value
#Note hermitian conjugation
Energy = NewBasis.conj().T @ Hamiltonian @ NewBasis
print(Energy)
```

Not an impressive results. We set up now a loop over many angles θ and ϕ and compute the energies

```
# define a number of angles
n = 20
angle = np.arange(0,180,10)
n = np.size(angle)
ExpectationValues = np.zeros((n,n))
for i in range (n):
    theta = np.pi*angle[i]/180.0
    Rx = np.cos(theta*0.5)*I-1j*np.sin(theta*0.5)*X
    for j in range (n):
```

A smarter way of doing this

The above approach means that we are setting up several matrix-matrix amd matrix-vector multiplications. Although straight forward it is not the most efficient way of doing this, in particular in case the matrices become large (and sparse). But there are some more important issues.
In a physical realization of these systems we cannot just multiply the state with the Hamiltonian. When performing a measurement we can only measure in one particular direction. For the computational basis states which we have, $|0\rangle$ and $|1\rangle$, we have to measure along the bases of the Pauli matrices and reconstruct the eigenvalues from these measurements.

Using the Pauli Z matrix

From our earlier discussions we know that the Pauli Z matrix has the above basis states as eigen states through

$$\boldsymbol{\sigma}_z|0\rangle = \boldsymbol{Z}|0\rangle = +1|0\rangle,$$

and

$$\boldsymbol{\sigma}_z|1\rangle = \boldsymbol{Z}|1\rangle = -1|1\rangle,$$

with eigenvalue -1.

The Pauli X matrix

For the Pauli X matrix on the other hand we have

$$\boldsymbol{\sigma}_x|0\rangle = \boldsymbol{X}|0\rangle = +1|1\rangle,$$

and

$$\boldsymbol{\sigma}_x|1\rangle = \boldsymbol{X}|1\rangle = -1|0\rangle,$$

with eigenvalues 1 in both cases. The latter two equations tell us that the computational basis we have chosen, and in which we will prepare our states, is not an eigenbasis of the σ_x matrix.

Rewriting in terms of Pauli Z matrices

We will thus try to rewrite the Pauli X matrix in terms of a Pauli Z matrix. Fortunately this can be done using the Hadamard matrix twice, that is

$$\boldsymbol{X} = \boldsymbol{\sigma}_x = \boldsymbol{HZH}.$$

The Pauli Y matrix can be written as

$$\boldsymbol{Y} = \boldsymbol{\sigma}_y = \boldsymbol{HS}^\dagger \boldsymbol{ZHS},$$

where S is the phase matrix

$$S = \begin{bmatrix} 1 & 0 \\ 0 & \imath \end{bmatrix}.$$

Rewriting the Hamiltonian

From here and on we will denote the Pauli matrices by X, Y and Z and we can write the expectation value of the Hamiltonian as

$$\langle\psi|(c+\mathcal{E})\boldsymbol{I}+(\Omega+\omega_z)\boldsymbol{Z}+\omega_x\boldsymbol{HZH}|\psi\rangle,$$

which we can rewrite as

$$(c+\mathcal{E})\langle\psi|\boldsymbol{I}|\psi\rangle+(\Omega+\omega_z)\langle\psi|\boldsymbol{Z}|\psi\rangle+\omega_x\langle\psi\boldsymbol{H}|\boldsymbol{Z}|\boldsymbol{H}\psi\rangle.$$

The first and second term are to easy to perform a measurement on since we we just need to compute $\langle\psi|\boldsymbol{I}|\psi\rangle$ and $\langle\psi|\boldsymbol{Z}|\psi\rangle$. For the final term we need just to add the action of the Hadamard matrix and we are done.

Plans for the week of February 26-March 1

1. Reminder on basics of the VQE method and how to perform measurements for the simpler one- and two-qubit Hamiltonians
2. Simulating efficiently Hamiltonians on quantum computers with the VQE method and gradient descent to optimize the state function ansatz
3. Introducing the Lipkin model
4. Reading suggestion, VQE review article