Application IV: Variational Quantum Eigensolvers for NISQ Devices

Winter School on Quantum Computing at Emory (WiSQCE)

Linghua Zhu
Economou & Barnes Groups, Virginia Tech

Jan 10, 2020
OUTLINE

• NISQ
• VQE
• QAQA
NISQ
Noisy Intermediate-Scale Quantum (NISQ)

- Quantum Computing is powerful
  - Best known example: Factoring
  - The power of quantum computing is limited: Can not efficiently solve worst-case instances of NP(Non-Polynomial)-hard optimization problems
    → Quantum complexity

- Quantum Computing is scalable
  - Qubits interact with the environment
    → Quantum error correction

J. Preskill, arXiv:1801.00862 (based on a keynote address at Quantum Computing for Business)
J. Preskill, arXiv:1801.00862 (based on a keynote address at Quantum Computing for Business)
NISQ

• When will quantum computers really be realized?
  → Not sure, maybe long

• How to solve this problem
  → A quantum system with ~50 qubits, already outperforms the class computer, has huge potential

50 qubits is a significant milestone

✓ Noisy Intermediate Scale Quantum.
  the size of quantum computers which will be available in the next few years, with a number of qubits ranging from 50 to a few hundreds
NISQ Disadvantages

- Lots of Noise
  (wrong gate applied, etc.)

- Lifetime for qubits is limited
  (lost the information was stored → number of operations you can perform in this time is limited)

- Number of qubits is limited

Algorithm Requirements

- Shallow circuit depth
- Robustness
- Limited number of qubits
Algorithms for NISQ

First ways to use NISQ machines:

- Variational Quantum Eigensolver (VQE) \(\leftrightarrow\) had an immediate impact

- Quantum Approximation Optimization Algorithm (QAOA)
  \(\rightarrow\) solve combinatorial optimization problems on a NISQ computer
  - Flexible to the circuit depth
  - Resistant to noise
  - Do not require a lot of qubits
VQE
VQE

P. Gokhale (based on a talk at ATPESC 2018)

• Guess, Check, Repeat

Based on Variational Principle:

$$\frac{\langle \psi(\hat{\theta})|H|\psi(\hat{\theta}) \rangle}{\langle \psi(\hat{\theta})|\psi(\hat{\theta}) \rangle} \geq E_{gs}$$

Classical-Quantum hybrid algorithm:

• Parameters $\hat{\theta}$ optimized with classical computer
• $|\psi(\hat{\theta})\rangle$ stored in qubits and $\langle E \rangle_{\hat{\theta}}$ obtained by measurements

Accuracy of variational method depends on form of $|\psi(\hat{\theta})\rangle$ critically
VQE Ansätze

- Parametrized wavefunction $|\psi(\vec{\theta})\rangle$ implemented on qubits by:
  \[ |\psi(\vec{\theta})\rangle = U(\vec{\theta})|\psi_{ref}\rangle \]
  $|\psi_{ref}\rangle$ : trivial product state

- Quantum gates: transform the states of system, represented by
  Unitary operators $\rightarrow U^\dagger U = I$

- Typical heuristic for ansatz: predetermined
  Chemistry Inspired $\rightarrow$ Unitary Coupled Cluster Singles and Doubles (UCCSD)
  Heuristic Inspired $\rightarrow$ RY, SWAPRZ

- Depth and two-qubit gate count are (related) key factors when assessing ansätze performance under noisy conditions

VQE Workflow

✓ Second quantized Hamiltonian \( \Rightarrow \)
\[
H = \sum h_{ij} \hat{a}_i \hat{a}_j + \sum h_{ijkl} \hat{a}_i \hat{a}_j \hat{a}_k \hat{a}_l^\dagger
\]

✓ Mapping to Qubit Hamiltonian \( \Rightarrow \)
\[
H = \sum h_{\alpha} \sigma^\alpha_i + \sum h_{\alpha\beta} \sigma^\alpha_\alpha \sigma^\beta_\beta + \sum h_{\alpha\beta\gamma} \sigma^\alpha_\alpha \sigma^\beta_\beta \sigma^\gamma_\gamma + \ldots
\]

✓ Store states \( |\psi(\tilde{\theta})\rangle \) & Measure expected energy value \( \langle E \rangle_{\tilde{\theta}} \)

✓ Parameter optimization \( \tilde{\theta} \)

Image from Physics 11, 14 (2018)

VQE Example -- Maxcut

Weight Maxcut Problem:

- An \( n \)-node undirected graph \( G = (V, E) \) where \(|V| = n\) with edge weights \( w_{i,j} \neq 0, w_{i,j} = w_{j,i} \) for \( i, j \in E \)
- A partition of the original set \( V \) into two subsets, sum of weights of edges crossing the cut reach the maximum.
- Assigning \( x_i = 0 \) or \( 1 \) to each node \( i \), the global profit function which try to maximize:

\[
C(x) = \sum_{i,j} w_{i,j} x_i (1 - x_j)
\]

- Map to Ising Hamiltonian

\[
x_i = (1 - Z_i)/2 \\
H = \sum_i w_i Z_i + \sum_{i<j} w_{i,j} Z_{i,j}
\]

https://github.com/Qiskit/qiskit-tutorial
VQE Example

- Guess, Check, Repeat

\[
H = \sum_i w_i Z_i + \sum_{i<j} w_{i,j} Z_{i,j}
\]

Evaluate:

\[
C(\tilde{\theta}) = \langle \psi(\tilde{\theta}) | H | \psi(\tilde{\theta}) \rangle = \sum_i w_i \langle \psi(\tilde{\theta}) | Z_i | \psi(\tilde{\theta}) \rangle + \sum_{i<j} w_{i,j} \langle \psi(\tilde{\theta}) | Z_{i,j} | \psi(\tilde{\theta}) \rangle
\]

\[
|\psi(\tilde{\theta})\rangle = [U_{\text{single}}(\tilde{\theta}) U_{\text{entangler}}]^m |+\rangle
\]

\[
U_{\text{single}}(\tilde{\theta}) = \sum_{i=1}^n Y(\theta_i)
\]

Depth

C-Phase gates (fully entangling gates)

https://github.com/Qiskit/qiskit-tutorial
from qiskit import BasicAer
from qiskit.tools.visualization import plot_histogram
from qiskit.optimization.ising import max_cut, tsp
from qiskit.aqua.algorithms import VQE, ExactEigensolver
from qiskit.aqua.components.optimizers import SPSA
from qiskit.aqua.components.variational_forms import RY
from qiskit.aqua import QuantumInstance
from qiskit.optimization.ising.common import sample_most_likely

# setup aqua logging
import logging
from qiskit.aqua import set_qiskit_aqua_logging
set_qiskit_aqua_logging(DEBUG)  # choose INFO, DEBUG to see the log

from qiskit import IBMQ
# provider = IBMQ.load_account()

n=4  # Number of nodes in graph
G=nx.Graph()
G.add_nodes_from(np.arange(0,n,1))
elist=[(0,1,1.0),(0,2,1.0),(0,3,1.0),(1,2,1.0),(2,3,1.0)]
# tuple is (i,j,weight) where (i,j) is the edge
G.add_weighted_edges_from(elist)

colors = ['r' for node in G.nodes()]
pos = nx.spring_layout(G)
default_axes = plt.axes(frameon=True)
nx.draw_networkx(G, node_color=colors, node_size=600, alpha=.8, ax=default_axes, pos=pos)
from docplex.mp.model import Model
from qiskit.optimization.ising import docplex

# Create an instance of a model and variables.
mdl = Model(name='max_cut')
x = {i: mdl.binary_var(name='x_{0}'.format(i)) for i in range(n)}

# Object function
max_cut_func = mdl.sum(w[i,j]* x[i] * ( 1 - x[j] ) for i in range(n) for j in range(n))
mdl.maximize(max_cut_func)

seed = 10598

spsa = SPSA(max_trials=300)
ry = RY(qubitOp.num_qubits, depth=5, entanglement='linear')
vqe = VQE(qubitOp, ry, spsa)

backend = BasicAer.get_backend('statevector_simulator')
quantum_instance = QuantumInstance(backend, seed_simulator=seed, seed_transpiler=seed)

result = vqe.run(quantum_instance)
x = sample_most_likely(result['eigvecs'][0])
print('energy:', result['energy'])
print('time:', result['eval_time'])
print('max-cut objective:', result['energy'] + offset)
print('solution:', max_cut.get_graph_solution(x))
print('solution objective:', max_cut.max_cut_value(x, w))

colors = ['r' if max_cut.get_graph_solution(x)[i] == 0 else 'b' for i in range(n)]
nx.draw_networkx(G, node_color=colors, node_size=600, alpha = .8, pos=pos)

https://github.com/Qiskit/qiskit-tutorial
VQE for $n = 3$ qubits and depth $m = 2$

VQE Advantages

- Qubits efficiently represent the molecular wavefunction
- Suitable for noisy intermediate-scale quantum devices
- Extended to multiple qubits to calculate more complex problem

_On the path for achieving a quantum advantage for real-world applications_
VQE Challenges

- Ansatz performance sensitive to ansatz structure
- Quantum coherence is very limited
- Practical constraints, noise, etc…
- Classical optimization is not infinitely powerful
Current Results (VQE)


Unitary partition \(\rightarrow\) additional unitary

“Measurement reduction in variational quantum algorithms”

“Efficient quantum measurement of Pauli operators”

“Variational circuit compiler for quantum error correction”

"Quantum computational chemistry."

"Quantum chemistry in the age of quantum computing."
Estimating the expectation value → Fundamental task → Hard to estimates directly

- Decompose observable operator → $\sum \{P_i\}_{i=1}^{t} \leftrightarrow$ measure each $P_i$ separately
- Assembles into commuting subsets

$$C_i := \{P_{ij}\}_{j=1}^{m_i}, i = 1, \ldots, N$$

All Paulis in a group can be measured at the same time

Any group be simultaneously diagonalised by unitary $U$:

$$\langle P \rangle_\psi = \langle \Lambda(P) \rangle_{U\psi}, \quad \langle \Lambda(P) \rangle := U P U^\dagger$$

Inefficient → very large number of Paulis $\sim n^4$
Efficient quantum measurement Pauli operators

Rotation Construction:

✓ CZ-construction
✓ CNOT-construction

Grouping Strategy:

Hamiltonian: \( H = \sum_{i=1}^{N} H_i = \sum_{i=1}^{N} \sum_{j=1}^{m_i} a_{ij} P_{ij} \)

• placing the operators with the largest \( |a_{ij}|^2 \) coefficients in the same groups

Ratio Matrix: \( \hat{R} := \left( \frac{\sum_{i=1}^{N} \sum_{j=1}^{N_i} |a_{ij}|}{\sqrt{\sum_{i=1}^{N} \sum_{j=1}^{N_i} |a_{ij}|^2}} \right)^2 \)

✓ SORTED INSERTION outperforms four conventional greedy coloring algorithms & seek the minimum number of groups
Quality of a VQE simulation is only as good as the ansatz.

Performance improved
SYMMETRY PRESERVATION VQE
Q: How many variational parameters are required to potentially explore the full Hilbert space of trial states (for n qubits)?

A: $2^{n+1} - 2$

(-2 for normalization and a global phase)

Q: What if you know the number of particles in the ground state? (for n qubits and m electrons)

E.g. $|0101\rangle$ has n=4 and m=2

A: $2 \binom{n}{m} - 2$

Q: What if we also know what the $S_z$ and $S^2$ are for our ground state?

A: Much less than the two above scaling

$$\dim(H_{n,m,s,s_z}) = \sum_{k=0}^{m-2-s} \binom{n}{\frac{n}{2} - k} \binom{\frac{n}{2} - k}{m - 2k} \times \frac{(2s + 1)(m - 2k)!}{\left(\frac{m}{2} - k - s\right)! \left(\frac{m}{2} - k + s + 1\right)!}$$
ASWAP Ansatz & Minimal Parameters

- Construct ansätze that saturate the lower bounds on the number of needed variational parameters
- Enforce information about known symmetries of the system in the ansätze themselves
- A handy tool will be the $A(\theta, \phi)$ gate, it preserves particle number, and $A(\theta, 0)$ has time reversal symmetry (real coefficients)

$$A = \begin{pmatrix}
|00\rangle & |01\rangle & |10\rangle & |11\rangle \\
|00\rangle & 0 & \cos \theta & e^{i\phi} \sin \theta & 0 \\
|01\rangle & 0 & e^{-i\phi} \sin \theta & -\cos \theta & 0 \\
|10\rangle & 0 & 0 & 0 & 1 \\
|11\rangle & 0 & 0 & 0 & 1
\end{pmatrix}$$

$$R(\theta, \phi) = R_Z(\phi + \pi)R_Y(\theta + \frac{\pi}{2})$$


to appear in: *Npj Quantum Inf.*
**Ansatz for Fixed Particle Number**

- **Strategy:** Put register into appropriate, separable basis state, e.g. $|0101\ldots0101>$
- Apply layers of A gates until $\binom{n}{m}$ A gates are placed
- Fix any two of the $\phi$ parameters
- ... It works!

- Numerically verified by sampling states in given subspace and optimizing the overlap between the final state and the sampled state

Ansatz for Fixed $S_Z$ and $S^2$

- Strategy: Parameterize fixed particle number subspace with hyperspherical coordinates.
- Eliminate unwanted states outside the given subspace by choosing $u_i$ appropriately.
- Construct unitary/circuit that places the register in the appropriate subspace (several methods).

\[\begin{align*}
\vec{r}_i &= \sum_j \sin u_j \cdot \vec{p}_j, \\
\vec{r}_i &= \sum_j \cos u_j \cdot \vec{p}_j
\end{align*}\]

\[\begin{bmatrix}
|0\rangle_{\vec{r}_1}, |0\rangle_{\vec{r}_2}, \ldots, |0\rangle_{\vec{r}_s} \\
|0\rangle \end{bmatrix}
\]
We find performance improvements over other, heuristic based ansatz.

We also generally find improvements in both the number of CNOTs and the number of parameters.
New Challenges to Consider

• Real devices don’t have full connectivity
  Introduces overhead of SWAP gates
  (more CNOTs)
• Real devices have mixed error rates
  Choosing a good layout is important
• Errors introduce a noisy objective function
  How do you optimize a function that lies to you?
• Other people also want to use the devices!!!
  Queue lines are long

IBMQ Poughkeepsie 20 Qubit Device (error rates on Sept 19)
Richardson Extrapolation

• Not specific to QIS, well established in early 20th century
• Core idea: Increase noise in the system to then extrapolate backwards to a noiseless case
• Rigorous in theory, but rigor is often lost in implementation

Dumitrescu et al. PRL 120, 210501 (2018)
Symmetry Preservation in Noisy System

The ASWAP ansatz consistently preserves each desired symmetry.
The improvements in error rates are enough to achieve chemical accuracy.

G. Barron, et al. *in preparation*
- Symmetry preservation VQE $\rightarrow$ predefined problem $\rightarrow$ predetermined ansatz
- ADAPT-VQE $\rightarrow$ universal problem $\rightarrow$ grow ansatz
Adaptive Derivative Assembled Pseudo-Trotter (ADAPT)

Iteratively grow the ansatz in a form of

\[ |\psi_n^{ADAPT}(\vec{\theta})\rangle = e^{\theta_n \hat{T}_n} \ldots e^{\theta_2 \hat{T}_2} e^{\theta_1 \hat{T}_1} |\psi^{HF}\rangle \]

|\psi^{HF}\rangle: Hartree-Fock product state.
\[ \hat{T}_i = -\hat{T}_i^\dagger \]: chosen one at a time from a predefined pool \{\hat{T}_i\}

Selection Criteria: choose the operator that modifies the energy the most

\[ \frac{\partial}{\partial \theta_i} \langle E \rangle = \lim_{\theta_i \to 0} \left( \langle \psi_n^{ADAPT} | e^{-\theta_i \hat{T}_i} (\mathbf{-} \hat{T}_i) H e^{\theta_i \hat{T}_i} |\psi_n^{ADAPT}\rangle + \langle \psi_n^{ADAPT} | e^{-\theta_i \hat{T}_i} (+ \hat{T}_i) H e^{\theta_i \hat{T}_i} |\psi_n^{ADAPT}\rangle \right) \]

\[ = \langle \psi_n^{ADAPT} | [\hat{H}, \hat{T}_i] |\psi_n^{ADAPT}\rangle \]

\[ \leftrightarrow \text{can be measured in a quantum processor} \]

\[ |\psi_n^{ADAPT}(\vec{\theta})\rangle \text{ changing with } n, \text{ measured each iteration} \]
ADAPT-VQE

2) Operator pool
\[ \hat{A}_m = \left\{ \hat{\tau}_p^q + \hat{\tau}_p^q, \hat{\tau}_{pq}^r + \hat{\tau}_{pq}^r, \hat{\tau}_{pq}^r + \hat{\tau}_{pq}^r \right\} \]

3) Initialize
\[ |\psi^{(0)}\rangle = |\psi^{\text{HF}}\rangle \]

4) Prepare states
\[ n = n + 1 \]

Select operators from pool

5) Measure gradients
\[ \frac{\partial E^{(n)}}{\partial \theta_1} = \langle \psi^{(n)} | [\hat{H}, \hat{A}_1] | \psi^{(n)} \rangle \]
\[ \frac{\partial E^{(n)}}{\partial \theta_2} = \langle \psi^{(n)} | [\hat{H}, \hat{A}_2] | \psi^{(n)} \rangle \]
\[ \vdots \]
\[ \frac{\partial E^{(n)}}{\partial \theta_N} = \langle \psi^{(n)} | [\hat{H}, \hat{A}_N] | \psi^{(n)} \rangle \]

6) Converged?
Yes
No
Select operator with largest gradient

7) Grow ansatz
\[ |\psi^{(n+1)}\rangle = e^{\hat{A}_{n+1}} |\psi^{(n)}\rangle \]

8) VQE: Re-optimize all parameters
\[ E^{(n+1)} = \min_{\theta^{(n+1)}} \langle \psi^{\text{HF}} | e^{-\theta_{n+1} \hat{A}_1} \cdots e^{-\theta_{n+1} \hat{A}_n} \hat{H} e^{\theta_{n+1} \hat{A}_n+1} \cdots e^{\theta_{n+1} \hat{A}_1} | \psi^{\text{HF}} \rangle \]

\[ \theta^{(n+1)}_{\text{guess}} = \{ \theta^{(n)} , 0 \} \]
ADAPT-VQE Performance

- Use minimal number of parameters and operator for a given pool
- Gradient selection gives ‘correct’ ordering of operators

<table>
<thead>
<tr>
<th>PES curve</th>
<th>Error (log scale)</th>
<th>Number of parameters</th>
</tr>
</thead>
</table>


\[
\begin{align*}
\epsilon_1 &= 0.1 \\
\epsilon_2 &= 0.01 \\
\epsilon_3 &= 0.001
\end{align*}
\]
Qubit ADAPT-VQE

- Performance depends on the operator pool, which is predefined
- Each fermionic operator gives $O(n)$ gates
  - Qubit pool: use individual Pauli strings as elements in the pool
  - Pool of operators can be dictated by hardware
  - Shorter circuits

Quantum Approximate Optimization Algorithm (QAOA)

- Standard method for combinatorial optimization with a gate-based approach
- Variational parameters of the circuit allow for the solution
- Most promising approaches towards using near-term quantum computers for practical application
- Two different Hamiltonians, mixer & cost Hamiltonian → alternation → approach the ground state of the cost Hamiltonian
- Special case of a chemistry problem

\[ |\vec{y}, \vec{\beta}\rangle = U(M, \beta_p)U(C, \gamma_p) \cdots U(M, \beta_1)U(C, \gamma_1)|s\rangle \]

Weight Maxcut Problem:

- An $n$-node undirected graph $G = (V, E)$ where $|V| = n$ with edge weights $w_{i,j} \neq 0$, $w_{i,j} = w_{j,i}$ for $i, j \in E$.
- A partition of the original set $V$ into two subsets, sum of weights of edges crossing the cut reach the maximum.
- Assigning $x_i = 0$ or $1$ to each node $i$, the global profit function which try to maximize:

$$C(x) = \sum_{i,j} w_{i,j} x_i (1 - x_j)$$

- Map to Ising Hamiltonian

$$x_i = (1 - Z_i)/2 \quad \quad H = \sum_i w_i Z_i + \sum_{i<j} w_{i,j} Z_{i,j}$$

https://github.com/Qiskit/qiskit-tutorial
QAOA Example

• Hamiltonian:

\[
H = \sum_i w_i Z_i + \sum_i w_i Z_i Z_{i+1}
\]

• Goal:

\[
\min \langle \psi(\vec{\theta}) | H | \psi^{\text{th qubit}} \rangle
\]

• Construction:

A layer of Hadamard gates & followed by two alternating unitaries:

\[
U_C(\gamma) = e^{-i\gamma(\sum_{i=1}^{n} w_i \sigma_i^z + \sum_{i,j=1}^{n} w_{i,j} \sigma_i^z \sigma_j^z)}
\]

\[
U_B(\beta) = e^{-i\beta(\sum_{i=1}^{n} \sigma_i^x)}
\]

\[
U(\gamma, \beta) = \prod_{i=1}^{p} U_B(\beta_i)U_C(\gamma_i) | H \rangle \otimes n
\]
import PennyLane and Numpy

Define Operators

Objective Function

Optimization

QAOA Code

```
import pennylane as qml
from pennylane import numpy as np

n_wires = 4
graph = [(0, 1), (0, 3), (1, 2), (2, 3)]

# minimize the negative of the objective function
def objective(params):
    gammas = params[0]
    ...
    opt = qml.AdagradOptimizer(stepsize=0.5)

    # optimize parameters in objective
    params = init_params
    steps = 30
    for i in range(steps):
        params = opt.step(objective, params)
        if (i + 1) % 5 == 0:
            print("Objective after step {}: {:7f}\n".format(i + 1, -objective(params))
        qml.Rz(gamma, wires=wires)
        qml.CNOT(wires=[wire1, wire2])
```

https://pennylane.ai/qml/app/tutorial_qaoa_maxcut.html
ADAPT-QAOA

Original QAOA:

✓ Cost Hamiltonian: $H_C$, acts diagonally on the qubits

✓ Mixer Hamiltonian: $H_B = \sum_{i=1}^{n} X_i$

✓ Initial States: $|+\rangle^\otimes n$

✓ Parameterized quantum state:

$$|\psi_p(\vec{\gamma}, \vec{\beta})\rangle = e^{-i \beta_p H_B} e^{-i \gamma_p H_C} \ldots e^{-i \beta_1 H_B} e^{-i \gamma_1 H_C} |+\rangle^\otimes n$$

✓ Measure and minimize $\langle \psi_p(\vec{\gamma}, \vec{\beta})|H_C|\psi_p(\vec{\gamma}, \vec{\beta})\rangle$ by optimizing $\vec{\gamma}, \vec{\beta}$

ADAPT-VQE:

✓ Provides a protocol for the generation of a compact ansatz

Can we use ADAPT to create ansätze for QAOA?
**ADAPT-QAOA**

**Operator Pool:**

- For mixer Hamiltonian
- Selection Criteria: choose the mixer that modifies the energy the most

\[
\frac{\partial}{\partial \theta_i} \langle E \rangle = \langle \psi_n^{ADAPT} | [\hat{H}_c, \hat{o}_i] | \psi_n^{ADAPT} \rangle
\]

\( \hat{o}_i \): operator chosen from operator pool

**Single-qubit gate operators:**
\[ \{X_i, Y_i, \sum_{i=1}^{n} X_i, \sum_{i=1}^{n} Y_i \} \]

**Single & Entanglement operators:**
\[ \{X_i, Y_i, \sum_{i=1}^{n} X_i, \sum_{i=1}^{n} Y_i, Z_i Y_j, X_i Y_j, Z_i Z_j, X_i X_j, X_i Z_j, Y_i Y_j \} \]

*Zhu, et al. in preparation*
ADAPT-QAOA Performance

- Maxcut Problem for graphs with random edge weights
- Instead of fixing an ansatz upfront, grows it systematically
- Standard mixer is a good start, ADAPT protocol can find operators which give better performance
- Small number of parameters $\rightarrow$ shallow-depth circuits

Operator chosen: Zhu, et al. in preparation
Summary

- **Varitional Quantum Algorithms (VQE & QAOA)** → On the path for achieving a quantum advantage for real-world applications
  

- **Symmetries** play central role → efficient state preparation circuits respect *particle number, total spin, spin projection, and time-reversal symmetries* → Minimal number of variational parameters


- **ADAPT** provides a protocol for the generation of a compact ansatz → grows it systematically & improves convergence performance → shallow-depth circuits

  Zhu, et al. in preparation