QAOA Solutions and Comparison to the VQE

Youfu Qian

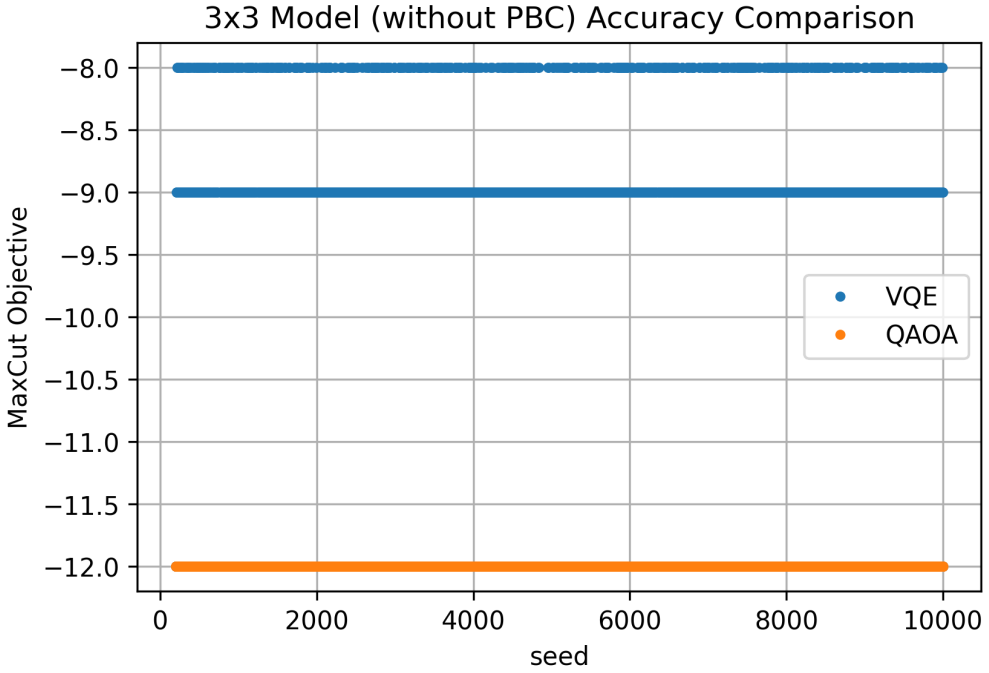
Feb 10th 2023

QAOA: Bipartite -> assign spins

VQE: assign spin at every iteration

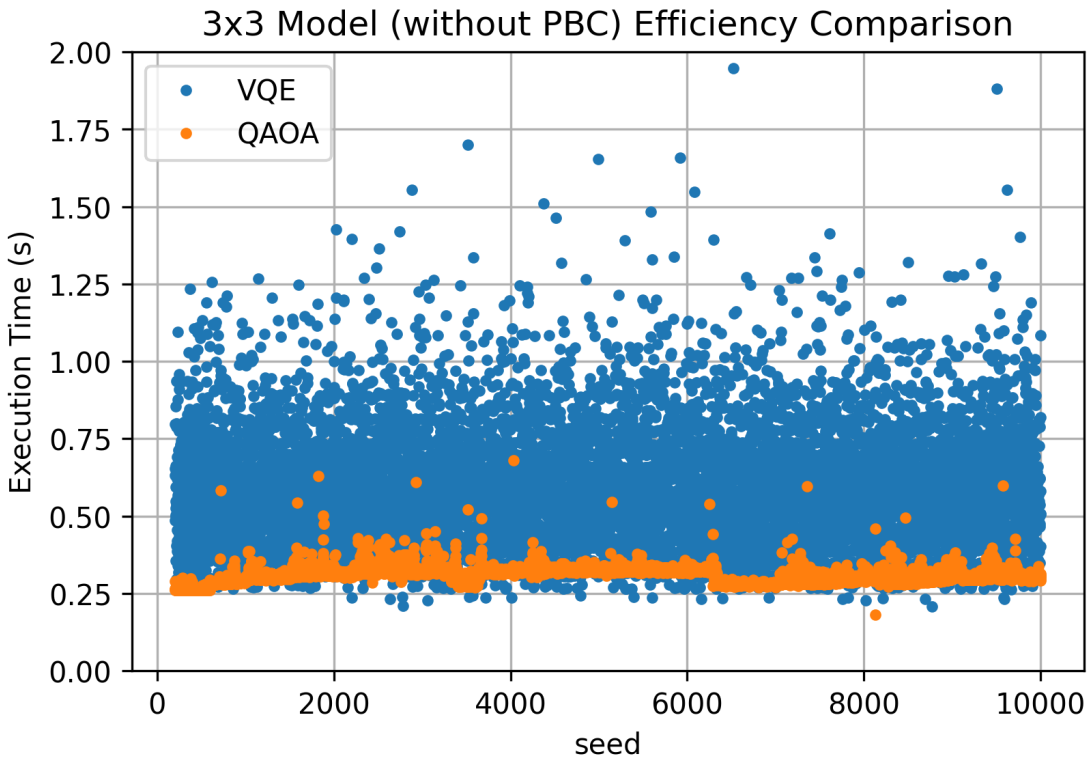
3x3 Model Without PBC

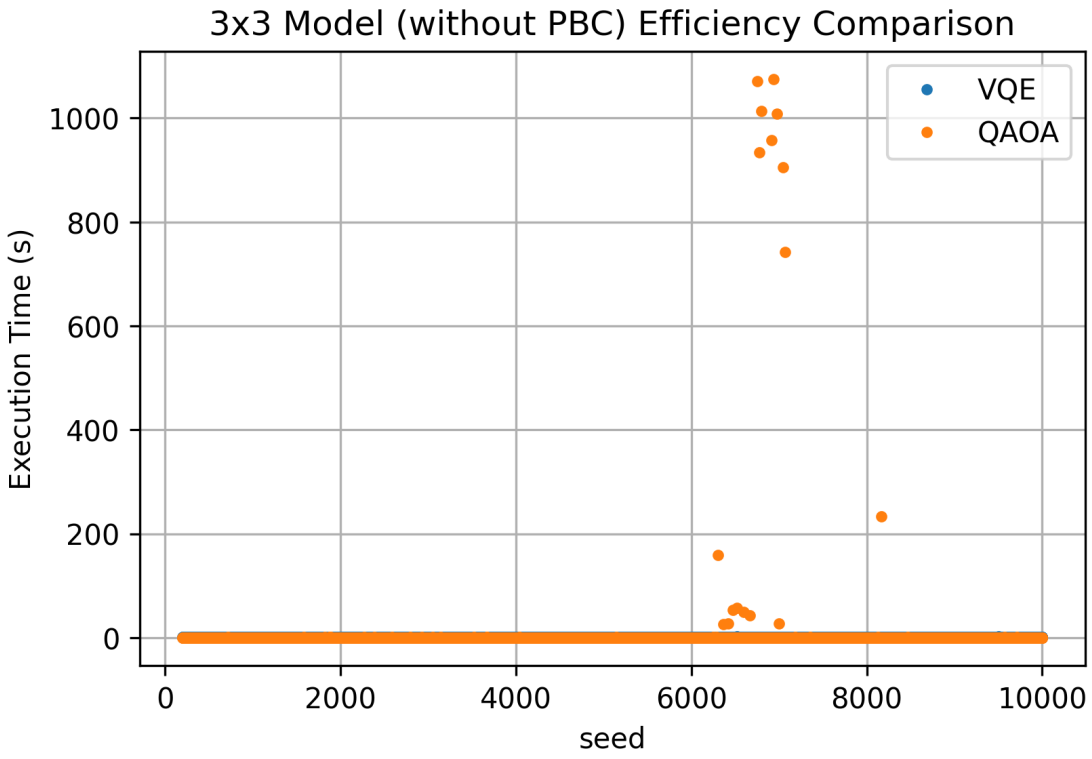
Accuracy



Algorithm	Backend	Optimizer
QAOA	aer	COBYLA
VQE	statevector	SLSQP

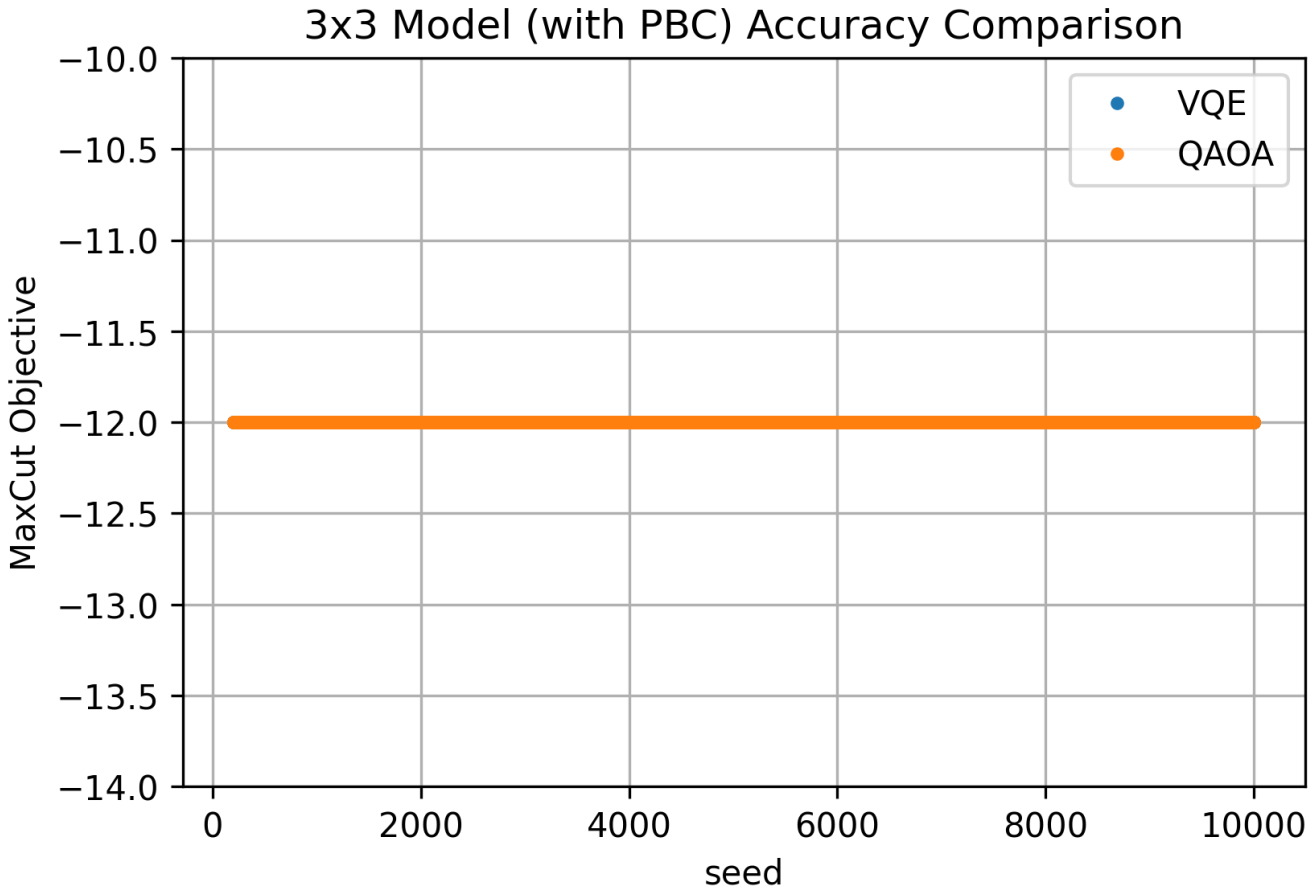
Efficiency


3x3 Model (without PBC) Efficiency Comparison
VQE
QAOA
Execution Time (s)
seed
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
0
2000
4000
6000
8000
10000

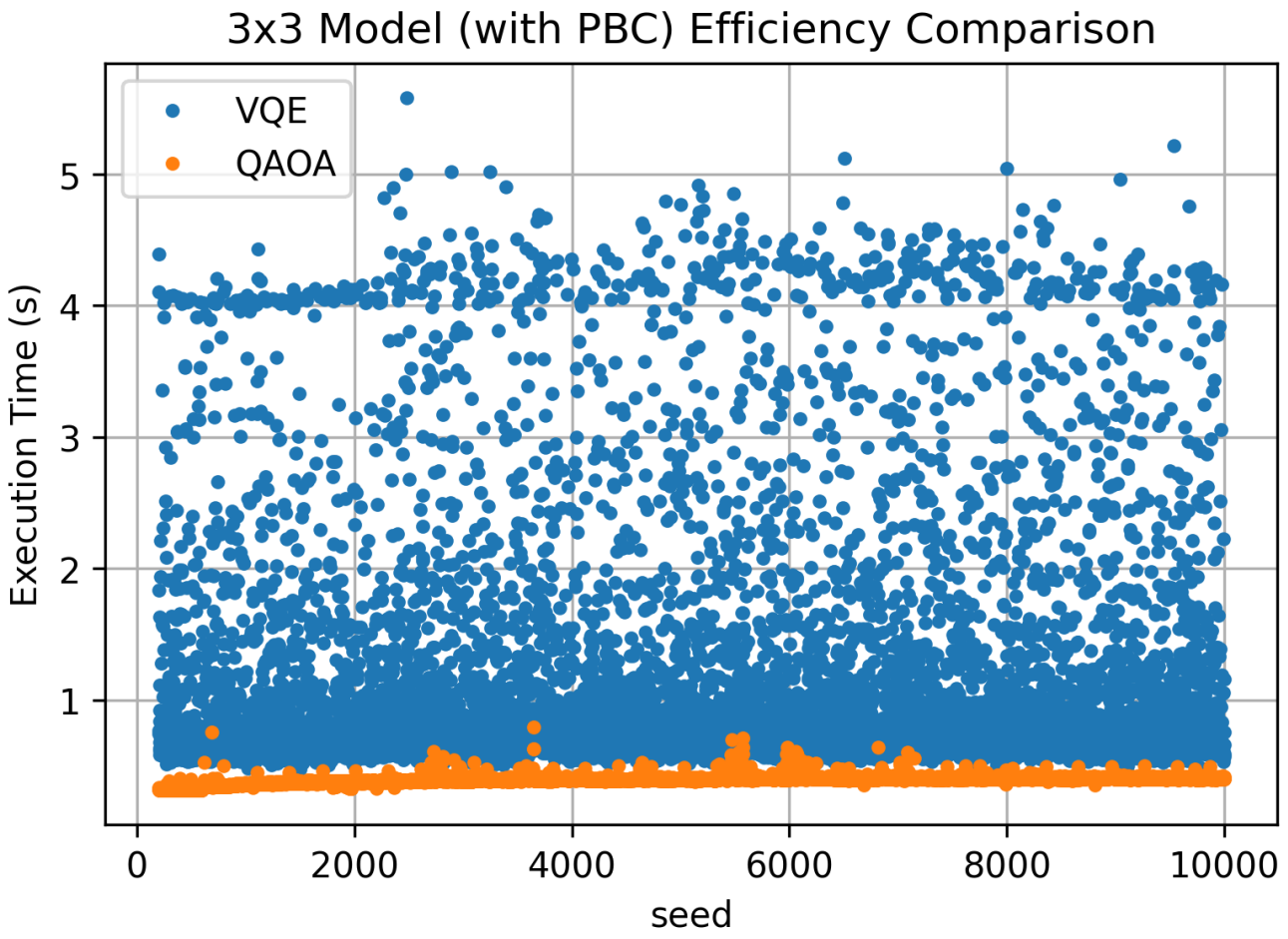

3x3 Model (without PBC) Efficiency Comparison
VQE
QAOA
Execution Time (s)
seed
0
200
400
600
800
1000
0
2000
4000
6000
8000
10000

3x3 Model With PBC

Accuracy

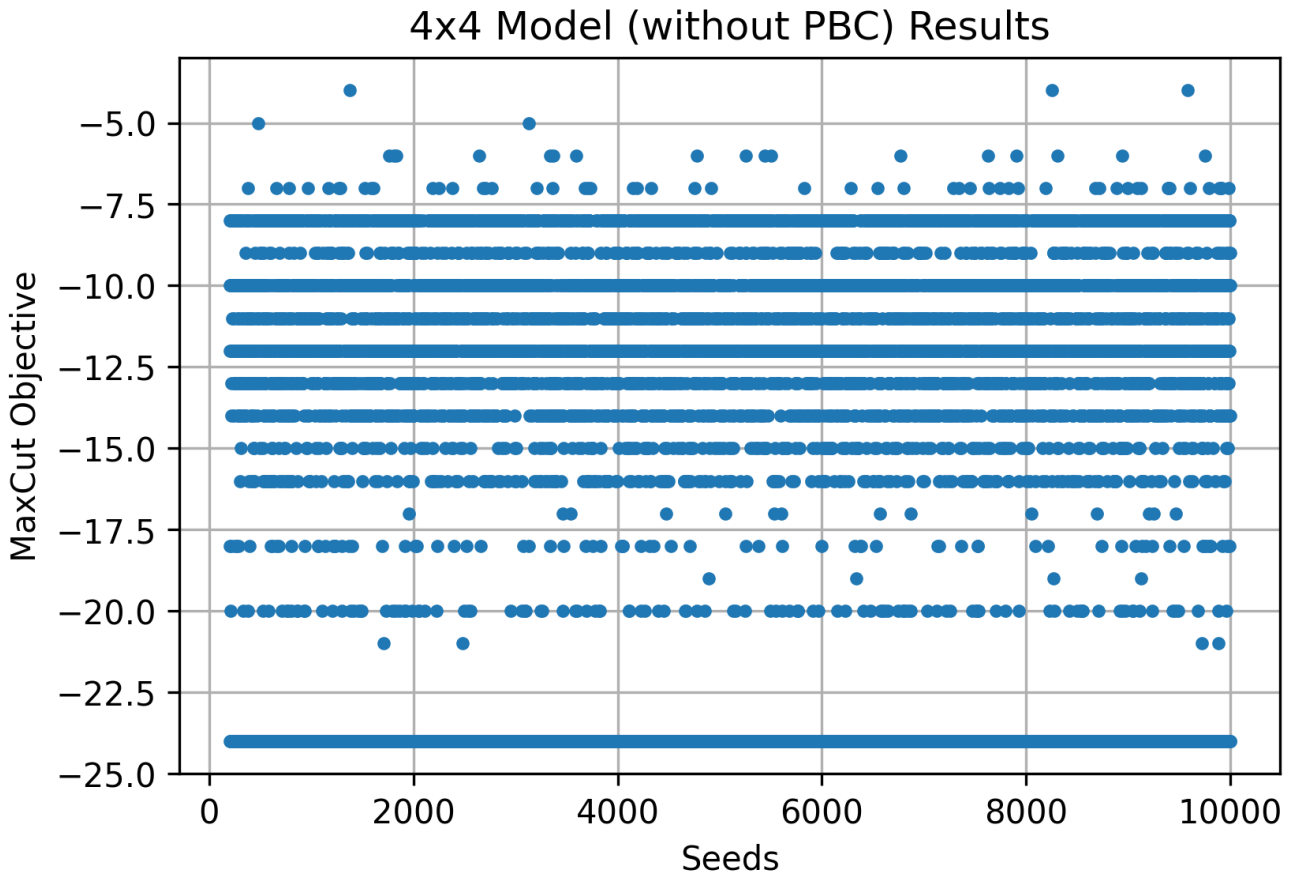

3x3 Model (with PBC) Accuracy Comparison
VQE
QAOA
MaxCut Objective
−10.0
−10.5
−11.0
−11.5
−12.0
−12.5
−13.0
−13.5
−14.0
0
2000
4000
6000
8000
10000
seed

Efficiency


3x3 Model (with PBC) Efficiency Comparison
VQE
QAOA
Execution Time (s)
1
2
3
4
5
0
2000
4000
6000
8000
10000
seed

4x4 Model Without PBC

QAOA Results: WILD!



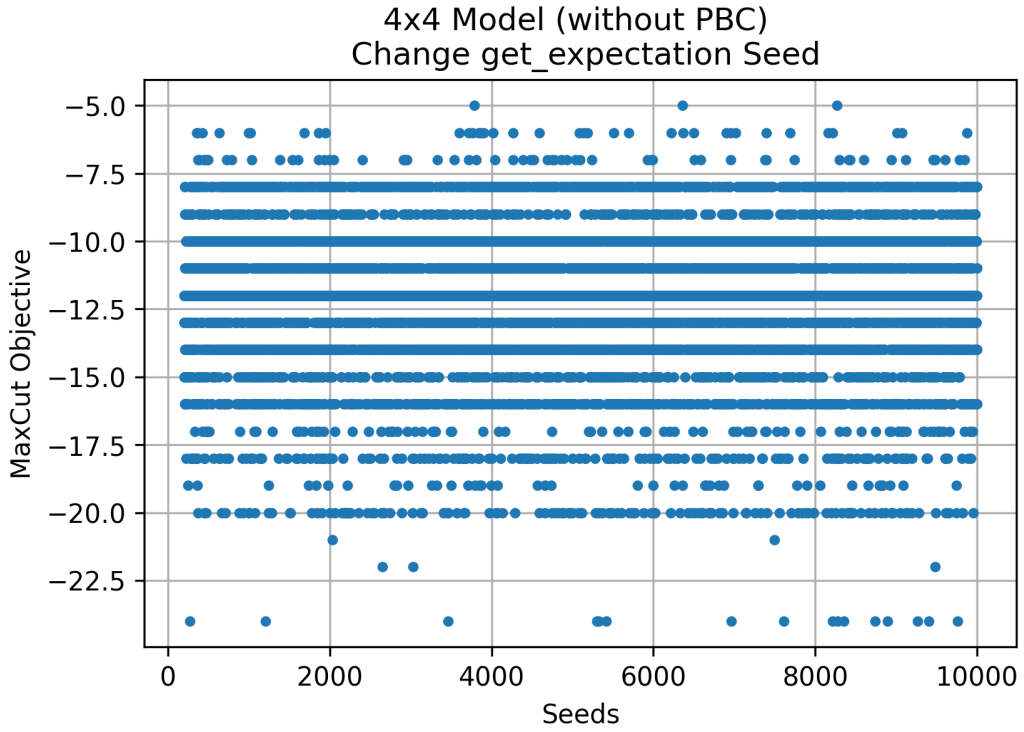
get_expectation seed = 10, backend.rum seed = vary

Two Places Have Seed Parameter

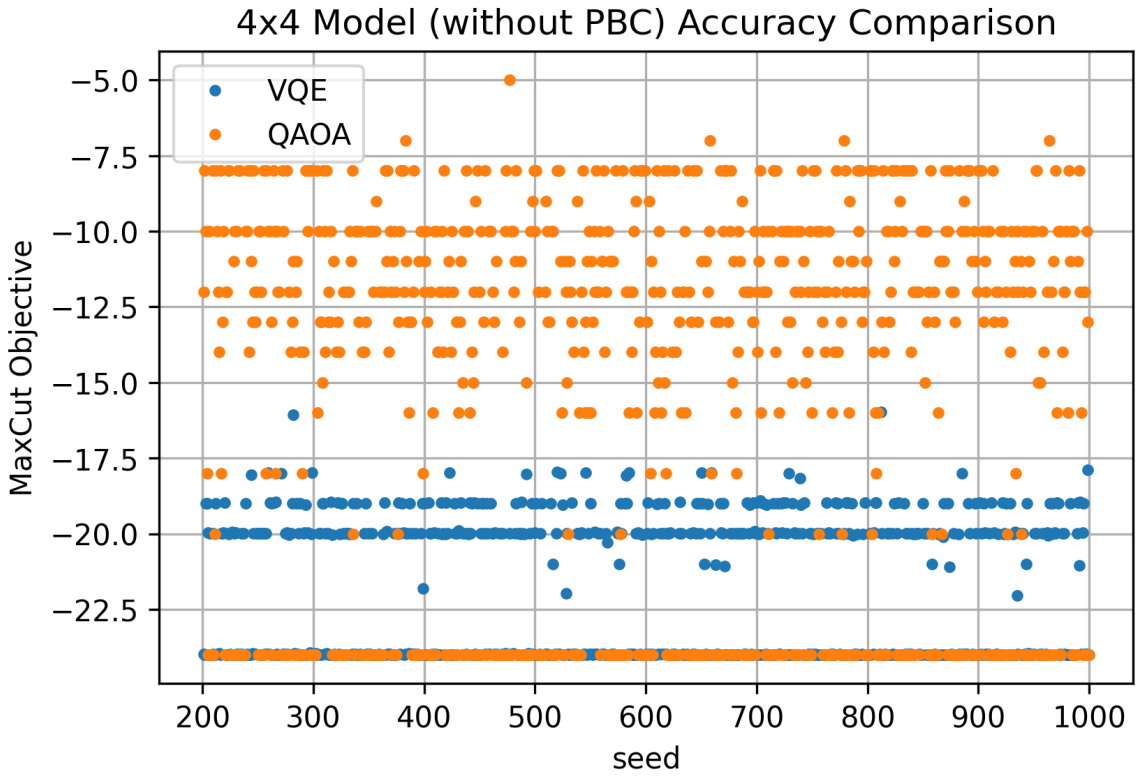
```python
# Finally we write a function that executes the circuit on the chosen backend
def get_expectation(G, seed, shots=1024):

    """
    Runs parametrized circuit

    Args:
        G: networkx graph
        p: int,
           Number of repetitions of unitaries
    """

    backend = Aer.get_backend('qasm_simulator')
    backend.shots = shots

    def execute_circ(theta):

        qc = create_qaoa_circ(G, theta)
        counts = backend.run(qc, seed_simulator = seed,
                             nshots=512).result().get_counts()

        return compute_expectation(counts, G)

    return execute_circ
```

```python
for seed in seeds:
    #begin recording time
    tic = time.time()

    # change seed here
    expectation = get_expectation(G, seed)

    res = minimize(expectation,
                           [1.0, 1.0],
                           method = optimizER)

    qc_res = create_qaoa_circ(G, res.x)

    result = backend.run(qc_res, seed_simulator=201).result()

    counts = result.get_counts()

    # stop recording time
    toc = time.time()

    time_taken = toc - tic

    counts = dict(counts)
    prob, state = max(counts.values()), max(counts, key = counts.get)
```

WILD! Not a good approach

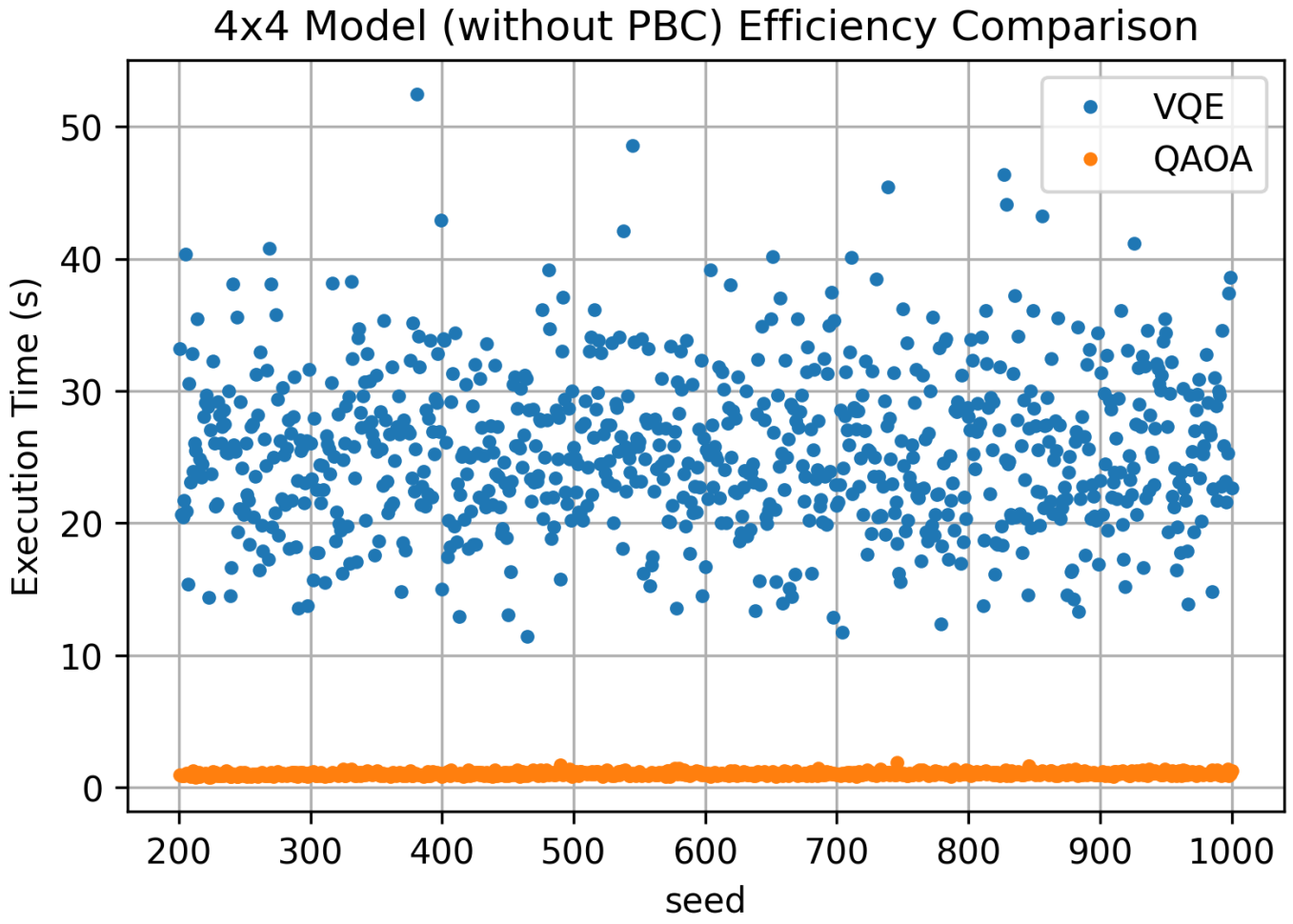


Accuracy



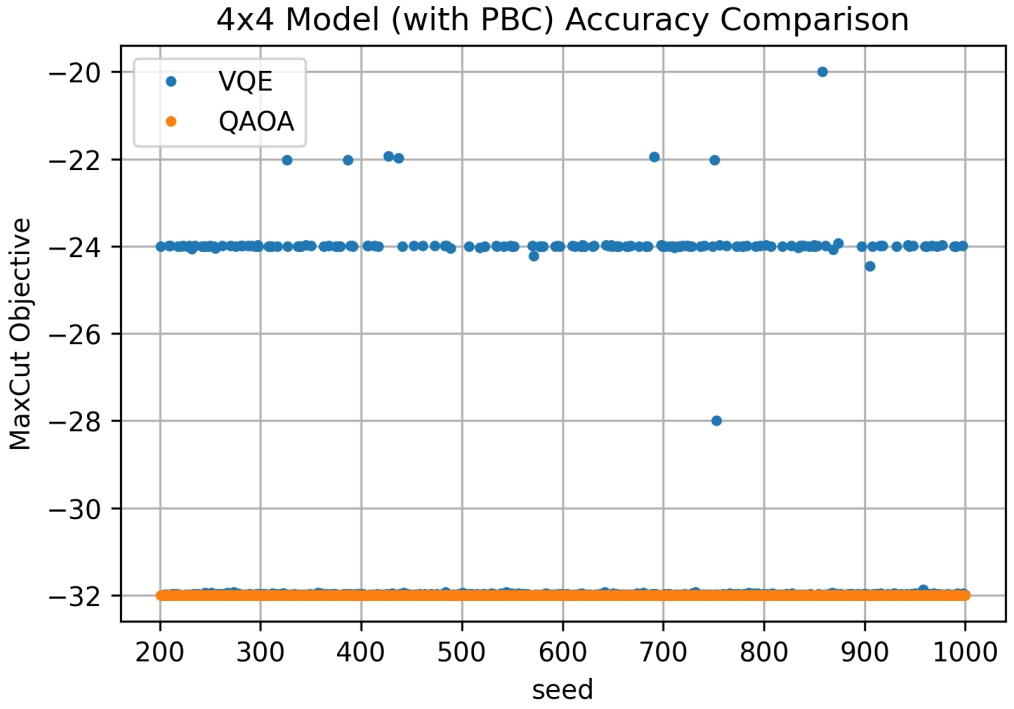
Algorithm	Backend	Optimizer
QAOA	aer	COBYLA
VQE	qasm	COBYLA

Efficiency


4x4 Model (without PBC) Efficiency Comparison
VQE
QAOA
Execution Time (s)
0
10
20
30
40
50
200
300
400
500
600
700
800
900
1000
seed

4x4 Model With PBC

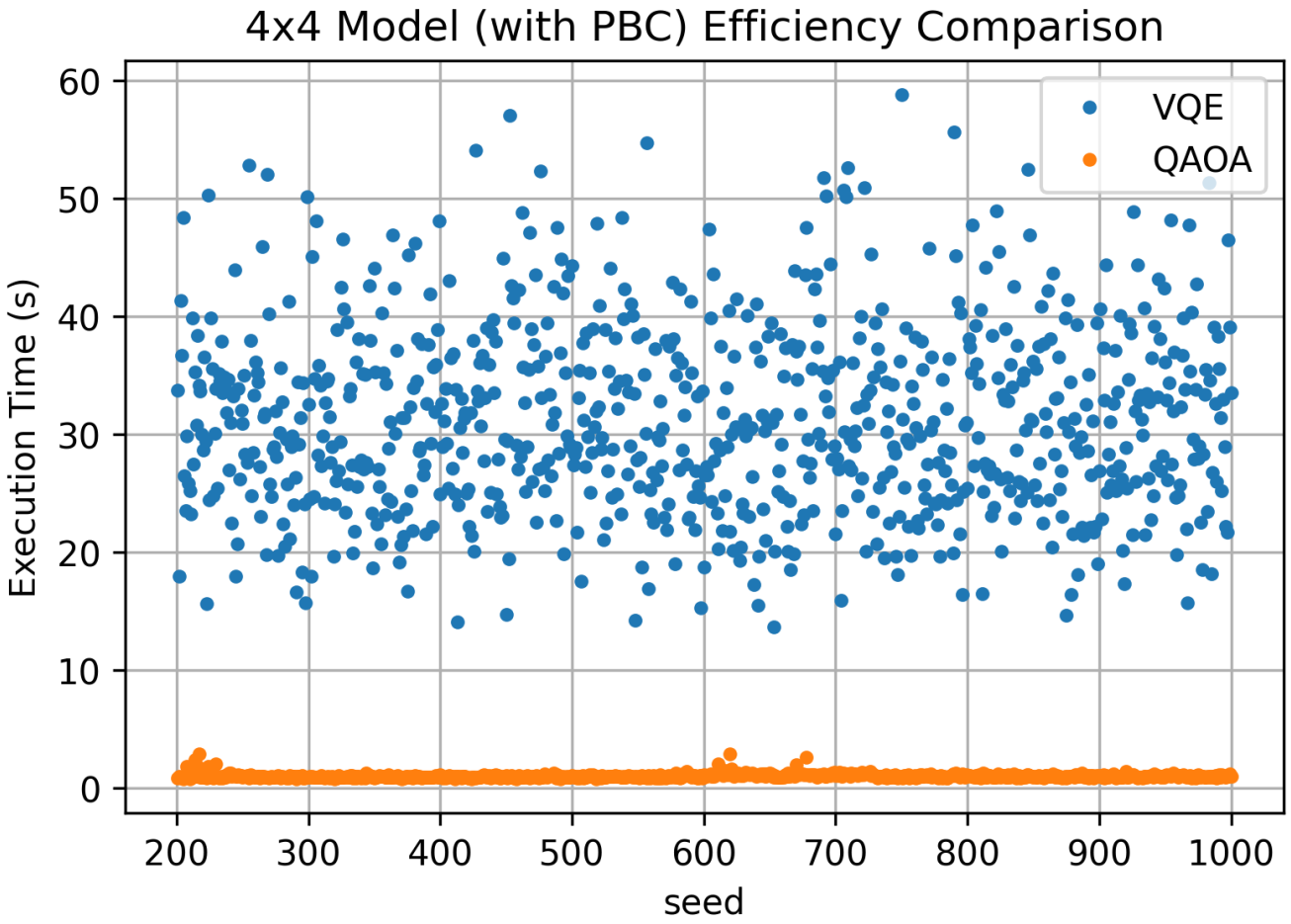
Accuracy



- get_expectation seeds fixed
- Change backend.run seed

Algorithm	Backend	Optimizer
QAOA	aer	COBYLA
VQE	qasm	COBYLA

Efficiency


4x4 Model (with PBC) Efficiency Comparison
VQE
QAOA
Execution Time (s)
seed
0
10
20
30
40
50
60
200
300
400
500
600
700
800
900
1000