Exercises for Introduction to Quantum Computing (physics7514) Wise 2025

L. Funcke, A. Bulgarelli and N. Dichter

---

1 Quantum simulation of free particle in 1+1D

Consider the Hamiltonian $\hat{H} = \frac{\hat{p}^2}{2}$ for a single free particle with $m = \hbar = 1$ and the corresponding time evolution operator

$$U(\delta t) = e^{-i\hat{p}^2 \delta t/2}$$

on a one-dimensional spatial lattice with $N = 2^n$ points. We define the computational basis $|x\rangle = |x_0 x_1 ... x_{n-1}\rangle$ with $x = \sum_{k=0}^{n-1} x_k 2^{n-1-k}$ as the eigenvalues of the position operator. We set the lattice spacing $a = 1$ and define the momentum operator differently from the one defined in Set06, i.e., we multiply $\hat{x}$ by $\delta p = \frac{2\pi}{N}$ and shift the spectrum to be in the range $[-\pi, \pi]$,

$$\hat{p}_x = qFT^\dagger(-\pi + \delta p\, \hat{x}) qFT.$$

a) Show that

$$e^{-i(-\pi+\delta p\hat{x})^2 \delta t/2} |x\rangle = e^{-i\,a\,\delta t/2} \left(\prod_{j=0}^{n-1} e^{i2\,b\,x_j 2^{n-1-j}\,\delta t/2} \right) \prod_{j,l=0}^{n-1} e^{-ic\,x_j x_l 2^{2n-2-l-j}\,\delta t/2} |x\rangle$$

and derive the expressions for a, b, c *(3 points).*

b) Implement a gate in Qiskit that performs the operation $e^{-i(-\pi+\delta p\hat{x})^2 \delta t/2}$, using only one- and two-qubit gates *(3 points).*

c) On a quantum register with six qubits, study the time evolution of the wave function in position space, starting from a δ-function in position space. For this, you can set $\delta t = 1$ and plot the wave function for $n = \{1, 2, 3, 4\}$ Trotter steps. Remember that the binary representation and therefore the "qFT" implementation in Qiskit, which you may use, is different compared to the one adopted in part a). *(4 points).*

Hint:
You may want to directly access the wavefunction. This can be achieved as follows:

a) You first have to append the instruction, to save the current statevector/wave function, to your circuit:

```
from qiskit_aer.library import SaveStatevector

ssv_name = "MyStatevectorName"
```

```
# n is the amount of qubits in qc
ssv_instr = SaveStatevector(n, label=ssv_name)
qc.append(ssv_instr, range(0, n))
```

b) You have to explicitly use the `statevector` simulator of `qiskit_aer`:

```
backend = AerSimulator(method="statevector")
```

c) Finally you have to use the `run` method of the `AerSimulator`, instead of an explicit Sampler:

```
result = backend.run(isa_circuit, shots=1).result()
qiskit_statevector = result.data(0)[ssv_name]
statevector = qiskit_statevector.data
```

`statevector` then contains the complex amplitudes with respect to each basis state ordered according to the Qiskit convention. `qiskit_statevector` is a Statevector.