COMP90084 Quantum Software Fundamentals

Semester 2, 2022, Assignment 1, Parts 2 & 3

This is the Part 2 and Part 3 of Assignment 1. This is a programming-based assignment. You must submit your programming and code running results within a single Jupyter Notebook file (.ipynb). Your code should have clear comments and variable/function/class names. You should also describe what you are doing in **Markdown** cells of your Jupyter Notebook. The total mark for this assignment is **20**. Your submission will be mainly judged by the print outputs and figure/diagram displays in the Jupyter Notebook. Due date: **14 October 2022 - 17:00 AEDT**.

1 Basic Quantum Algorithms

1.1 Single Qubit Gate Teleportation (1 mark)

You will need to answer the questions about single-qubit gate teleportation in this question. We will teleport a T gate onto an arbitrary single-qubit state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$:

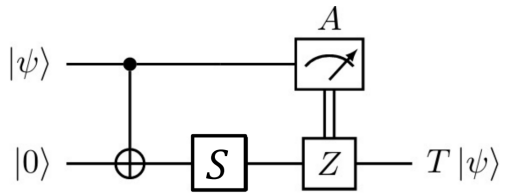

Figure 1: Teleport a T gate onto $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$

where

$$\mathrm{S} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \mathrm{T} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} \tag{1}$$

and

$$\begin{aligned} |A\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle + e^{i\pi/4}|1\rangle\right) \\ \left|A^{\perp}\right\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle - e^{i\pi/4}|1\rangle\right) \end{aligned} \tag{2}$$

The double-line connecting the Z gate and the measurement performed on $|A\rangle$ and $\left|A^{\perp}\right\rangle$ basis means that the Z operation will be applied to the second if $|A\rangle$ or $\left|A^{\perp}\right\rangle$, depending on your basis transform prior to the measurement, is measured.

Hint: $\left|A^{\perp}\right\rangle$ corresponds to the state that needs Z correction.

Hint 2: Basis change from $|A\rangle$ and $\left|A^{\perp}\right\rangle$ to computational basis:

$$U = |0\rangle\langle A| + |1\rangle\langle A|^{\perp}$$

Question (a) Program the quantum circuit shown in Fig. 1 with Qiskit. Be aware, that in Qiskit you can only measure on the computational basis (Z basis). To measure $|A\rangle$, you will need to perform proper single-qubit gates on the first qubit before the measurement operation. The circuit

should be defined within a **function**, which takes α and β as input arguments and produce the circuit diagram accordingly. Also, the state of the **second qubit** at the end of the circuit will need to be printed out, and compared with $T|\psi\rangle$.

1.2 The Deutsch–Jozsa Algorithm (1.5 marks = 0.5 + 1)

Consider a set of functions f_i defined over two-qubit (n=2) inputs as follows:

$$\begin{aligned} f_1(00) &= 0 \\ f_1(01) &= 0 \\ f_1(10) &= 0 \\ f_1(11) &= 0 \end{aligned} \tag{3}$$

$$\begin{aligned} f_2(00) &= 1 \\ f_2(01) &= 1 \\ f_2(10) &= 1 \\ f_2(11) &= 1 \end{aligned} \tag{4}$$

$$\begin{aligned} f_3(00) &= 0 \\ f_3(01) &= 1 \\ f_3(10) &= 1 \\ f_3(11) &= 0 \end{aligned} \tag{5}$$

$$\begin{aligned} f_4(00) &= 1 \\ f_4(01) &= 0 \\ f_4(10) &= 0 \\ f_4(11) &= 1 \end{aligned} \tag{6}$$

The functions are either constant or balanced. If you are given an unknown function f_i where i is one of the 4 values from the set (in decimal) 1, 2, 3, 4, your job is to develop and program a quantum algorithm, which can determine whether the function is constant or balanced.

Question (a) Describe a quantum algorithm that solves the above problem in just 1 query. You can write your answer for the description with **Markdown** cells in Jupyter Notebook. Give the quantum circuits for each function and write the corresponding quantum program.

Question (b) Devise a quantum algorithm to solve the above problem for n=3 which takes only 1 query. Ensure that the format of the balanced function for n=3 is such that the function takes the same values for the first two and the last two inputs [like the example above for n=2]:

$$\begin{aligned} f_1\{000,001,010,011,100,101,110,111\} &\rightarrow \{0,0,0,0,0,0,0,0\} \\ f_2\{000,001,010,011,100,101,110,111\} &\rightarrow \{1,1,1,1,1,1,1,1\} \\ f_3\{000,001,010,011,100,101,110,111\} &\rightarrow \{0,0,1,1,1,1,0,0\} \\ f_4\{000,001,010,011,100,101,110,111\} &\rightarrow \{1,1,0,0,0,0,1,1\} \end{aligned} \tag{7}$$

Draw quantum circuits for each f_i and demonstrate their working by computing quantum states at each time step through the circuit. Write a quantum program that executes your algorithm.

1.3 Grover's Algorithm (1.5 marks)

Compose a circuit for 3-qubit Grover's algorithm which marks the state 101. Suppose we assume errors in the circuit can be approximated by a rotation about the Z-axis after every gate in the circuit. The error is randomly selected (uniformly) between -0.1π and 0.1π. Write a quantum program that simulates this circuit with errors. **Since for all shots in a single simulation in Qiskit, the circuit will be fixed, so to reflect the randomness of errors in your circuit, you will need to run the simulation of the circuit many times with the error angles re-sampled each time, and set the number of shots equal to 1 for each simulation.** Run your program many times with and without errors and provide an estimate of the error's impact on the quantum algorithm's fidelity. What error correction or mitigation scheme would you use in this scenario to recover the fidelity of the quantum algorithm? Justify your answer.

1.4 The [[4,2,2]] Quantum Error Detection Code (2 marks)

The code space of the [[4,2,2]] quantum error detection code, which uses 4 physical qubits to encode 2 logical qubits, with code distance 2, is given by

$$\mathcal{C}_{[[4,2,2]]} = \text{span}\left\{\begin{array}{l} |00\rangle_L = \frac{1}{\sqrt{2}}(|0000\rangle + |1111\rangle) \\ |01\rangle_L = \frac{1}{\sqrt{2}}(|0110\rangle + |1001\rangle) \\ |10\rangle_L = \frac{1}{\sqrt{2}}(|1010\rangle + |0101\rangle) \\ |11\rangle_L = \frac{1}{\sqrt{2}}(|1100\rangle + |0011\rangle) \end{array}\right\}. \tag{8}$$

The logical X and Z operators on the first and second logical qubits are given by

$$\mathcal{L}_{[[4,2,2]]} = \left\{\begin{array}{l} \overline{X}_1 = X_1 X_3 \\ \overline{Z}_1 = Z_1 Z_4 \\ \overline{X}_2 = X_2 X_3 \\ \overline{Z}_2 = Z_2 Z_4 \end{array}\right\}. \tag{9}$$

Since on the physical qubits, we have

$$\begin{aligned} CNOT_{1,2} &= |00\rangle\langle 00| + |01\rangle\langle 01| + |11\rangle\langle 10| + |10\rangle\langle 11| \\ &= \frac{1}{2}(I \otimes I + Z \otimes I + I \otimes X - Z \otimes X) \end{aligned} \tag{10}$$

$$\begin{aligned} CNOT_{2,1} &= |00\rangle\langle 00| + |11\rangle\langle 01| + |10\rangle\langle 10| + |01\rangle\langle 11| \\ &= \frac{1}{2}(I \otimes I + I \otimes Z + X \otimes I - X \otimes Z) \end{aligned} \tag{11}$$

Your task for this section is to develop the logical CNOT gates between the first and the second logical qubits, $\overline{CNOT}_{1,2}$ and $\overline{CNOT}_{2,1}$, representing them as linear combinations of tensor products of single-qubit Pauli operators, and justify your answer by applying the logical CNOTs to the four basis states of the code space $|00\rangle_L$, $|01\rangle_L$, $|10\rangle_L$, and $|11\rangle_L$. Write your answer in a **Markdown** cell of your Jupyter Notebook.

2 Quantum Principal Component Analysis (6 marks = 1 + 3 + 2)

In this section, we will work with the well-known Iris dataset, which can be loaded from `scikit-learn`, see `https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html`. This dataset contains measurements of 150 Iris flowers from three different species (Iris Setosa, Iris Versicolour & Iris Virginica). The four measurements are the sepal width & length (in cm) and the petal width & length (in cm).

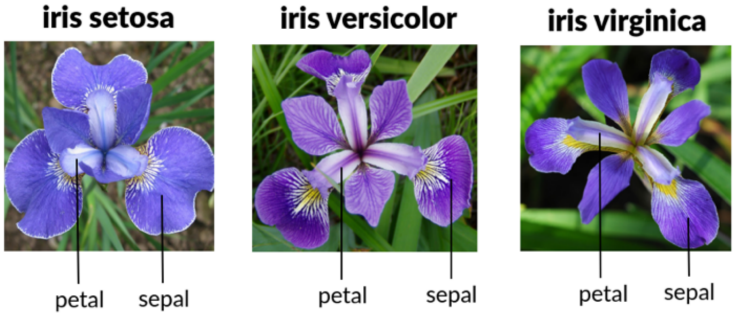

Figure 2: Three kinds of Iris flowers.

Question (a) Load the Iris dataset, calculate (with **NumPy**) and print out the 4×4 covariance matrix σ for all 150 entries in the dataset.

The first step of qPCA algorithm is to implement the operator $e^{i\rho t}$ from many copies of ρ, where ρ is the normalised physical state corresponding to the covariance matrix σ:

$$\rho = \frac{\sigma}{\mathrm{Tr}[\sigma]} \tag{12}$$

We used a partial SWAP operation in the lectures to implement $e^{i\rho t}$ iteratively. Since ρ is a two-qubit density operator, we will assume we already have an implementation for $e^{i\rho t}$ (as given in the lecture slides and tutorial materials).

Calculate (print out) the 4×4 matrix $e^{2\pi i\rho}$ for the Iris density matrix, along with the eigenvalues and eigenvectors. You can use **NumPy** for this question.

In the following parts of this section, we will use the Quantum Phase Estimation (QPE) algorithm to estimate the principal components (eigenvalues & eigenvectors) for the Iris dataset. To simulate this, you can use python or some other computing language. If you choose to use standard linear algebra packages such as python's **NumPy** and **SciPy**, you will not need to decompose the matrix $e^{2\pi i\rho}$ into CNOTs and single-qubit gates. However, in this case, you will need to implement controlled operations explicitly. If you instead choose to use **Qiskit**, or something similar, you will need to decompose $e^{2\pi i\rho}$ explicitly, but will be able to use the inbuilt controlled gate features. Note, decomposing $e^{2\pi i\rho}$ into a standard gate set (CNOTs and single-qubit gates) should not take a long time. There are packages freely available for such a task.

Question (b) In this part we will use the QPE algorithm with three phase estimation qubits to iteratively calculate the principal eigenvector for the Iris dataset.

Using the matrix (or set of gates) for $U_\rho = e^{2\pi i\rho}$ from part (a), find (and show) a first approximation for the principal eigenvector with the QPE circuit below. In this first instance, set

$|\psi_{\text{in}}\rangle = |+\rangle\,|+\rangle$. The state $\left|\psi_{\text{out}}^{(1)}\right\rangle$ will be the first order approximation when we detect $|111\rangle$ after the inverse Quantum Fourier Transform.

Next, run the same QPE circuit again, but now set $|\psi_{\text{in}}\rangle = \left|\psi_{\text{out}}^{(1)}\right\rangle$, to find a second order approximation for the principal eigenvector $\left|\psi_{\text{out}}^{(2)}\right\rangle$. . Repeat this four times until we have a fourth-order approximation $\left|\psi_{\text{out}}^{(4)}\right\rangle$. Calculate the fidelity between each approximate eigenvector $\left|\psi_{\text{out}}^{(i)}\right\rangle$ and the actual principal eigenvector of ρ. Print out your results. *Note: there may be a global phase difference between the output* $\left|\psi_{\text{out}}^{(i)}\right\rangle$ *and the true eigenvector.*

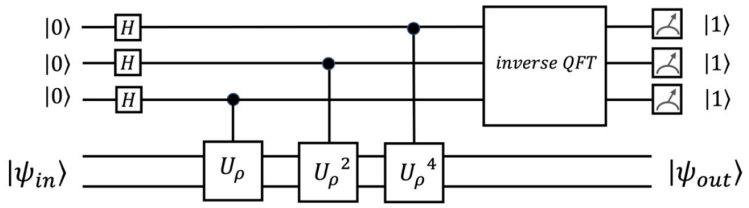

Figure 3: Circuit diagram for Quantum Phase Estimation.

Question (c) In this part we will use the QPE algorithm with five phase estimation qubits to estimate the principal eigenvalues for the Iris dataset. Use a circuit similar to that in part (b), but with five phase estimation qubits. Set $|\psi_{\text{in}}\rangle$ to the corresponding true eigenvector of ρ and assume arbitrary measurement outcomes for the top five qubits. What measurement outcomes correspond to the two largest eigenvalues of ρ, where the estimated eigenvalue is related to the measurement outcome via $\lambda = 0.b_1b_2b_3b_4b_5 = \frac{b_1}{2^1} + \frac{b_2}{2^2} + \frac{b_3}{2^3} + \frac{b_4}{2^4} + \frac{b_5}{2^5}$ (with $b_1b_2b_3b_4b_5$ referring to the measurement outcome)?

3 Variational Quantum Algorithms

3.1 Load a Probability Distribution (4 marks = 1 + 2 + 1)

The first step of applying quantum computing to finance like option pricing is to load a probability distribution, like the spot price distribution when pricing the European call option. In this section, your task is to load a probability distribution with variational quantum circuits:

$$P\left(S_T\right) = \frac{1}{S_T\sigma\sqrt{2\pi T}} \exp\left(-\frac{\left(\ln\left(S_T\right) - \ln\left(S_0\right) - T\left(r - \frac{\sigma^2}{2}\right)\right)^2}{2\sigma^2 T}\right) \tag{13}$$

Where

$$r = 4\%, \sigma = 12.5\%, S_0 = 2, T = 300/365 \tag{14}$$

And S_T is the spot price at maturity. The task of the loading circuit is to encode the probability of the strike prices into the amplitudes of the states:

$$|0\rangle^{\otimes n} \rightarrow \sum_{i=0}^{2^n-1} \sqrt{p_i}|i\rangle \tag{15}$$

For the quantum part of this section, we will use **Pennylane**(`https://pennylane.ai/`) for composing and training variational quantum circuits.

Question (a) Discretize the function $P(S_T)$, with $S_{T,min} = 1.5$ and $S_{T,max} = 2.5$. Since there will be three qubits in our circuit, the number of data points need is 8, including $S_{T,min}$ and $S_{T,max}$. Plot $P(S_T)$ with respect to the discrete S_T, $S_{T,i}, i \in \{0,1,2,3,4,5,6,7\}$. Label the discrete $S_{T,i}$ with $|i\rangle$, plot the corresponding amplitudes $\sqrt{P(S_{T,i})}$ with respect to the states $|i\rangle$. *Note: You need to normalize* $\sum P(S_{T,i})$ *to 1 before calculating the amplitudes.*

Question (b) In (a), we've already constructed our target state

$$|\psi\rangle_{\text{target}} = \sum_{i=0}^{7} \sqrt{P(S_{T,i})}|i\rangle \tag{16}$$

We denote our state at the end of the variational circuit with

$$|\varphi(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta})\,|000\rangle \tag{17}$$

Where $\boldsymbol{\theta}$ denotes the parameters in the variational circuit. The structure of $U(\boldsymbol{\theta})$ is shown in Fig. 4:

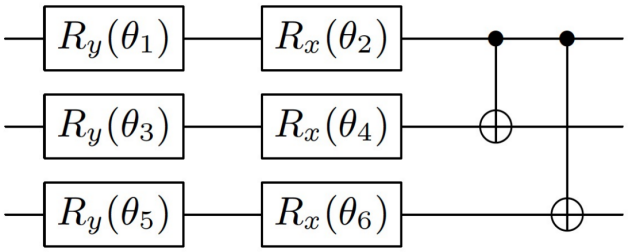

Figure 4: Structure (one layer) for the variational circuit needed for loading the probability distribution.

If we denote

$$P_i = |i\rangle\langle i| \quad i \in \{0,1,2,3,4,5,6,7\} \tag{18}$$

and the expectation value of P_i with respect to $|\varphi(\boldsymbol{\theta})\rangle$ (can be achieved with `expval`(op) function in Pennylane, see `https://docs.pennylane.ai/en/stable/introduction/measurements.html`) as

$$\langle\varphi(\boldsymbol{\theta})|\,P_i\,|\varphi(\boldsymbol{\theta})\rangle = \langle P_i\rangle_{\boldsymbol{\theta}} \tag{19}$$

which is the learned state probability. The learned state will be like

$$|\psi\rangle_{\text{learned}} = \sum_{i=0}^{7} \sqrt{\langle P_i\rangle_{\boldsymbol{\theta}}}\,|i\rangle \tag{20}$$

We can define the objective function to minimize as

$$\mathcal{L}(\boldsymbol{\theta}) = \text{Abs}[1 - \sum_{i=0}^{7} \sqrt{\langle P_i\rangle_{\boldsymbol{\theta}}}\sqrt{P(S_{T,i})}] \tag{21}$$

where Abs$[\cdot]$ denotes the absolute value. The fidelity between the target and learned state is

$$\mathcal{F}(|\psi\rangle_{\text{target}}, |\psi\rangle_{\text{learned}}) = |\psi\rangle_{\text{target}}\,\langle\psi|_{\text{learned}}\,|\psi\rangle_{\text{learned}}\,\langle\psi|_{\text{target}} \tag{22}$$

You can also define your own loss and fidelity function, as long as you provide a clear explanation of your approach in a **Markdown** cell in your Jupyter Notebook.

For this question, your task is to construct a variational circuit with one layer shown in Fig. 4 and train the variational circuit to load the discretized probability distribution. The recommended number of training steps is 500. Print out the value of the objective function every 50 steps during the training process and the final values of the objective/fidelity function after training. Plot the values of the objective function and fidelity function against training steps (just like what you do when training a neural network). Plot the learned probabilities and the target probabilities together.

Question (c) Repeat the process in (b), but increase the number of layers to **two**. Keep other settings unchanged. Print out the value of the objective function every 50 steps during the training process and the final values of the objective/fidelity function after training. Plot the values of the objective function and fidelity function against training steps (just like what you do when training a neural network). Plot the learned probabilities and the target probabilities together.

3.2 Variational Quantum Eigensolver (2 marks = 1.5 + 0.5)

In this section, you will solve the ground state energy problem of the Hydrogen molecule with variational quantum circuits. Generally speaking, solving the ground energy problem with quantum computers is an application of the variational principle:

$$E_0 \leq \frac{\langle \tilde{0}|H|\tilde{0}\rangle}{\langle \tilde{0} \mid \tilde{0}\rangle} \tag{23}$$

where H is the system Hamiltonian, $|\tilde{0}\rangle$ is the "trail ket", or ansätz, trying to mimic the real wave function at ground state with energy E_0, which is the smallest eigenvalue of the system Hamiltonian H. Starting from $|0^{\otimes n}\rangle$ for an $n-$qubit system, the "trial ket" can be written as a function of a set of (real) parameters θ:

$$\left|\tilde{0}\right\rangle = |\varphi(\theta)\rangle = U(\theta)\left|0^{\otimes n}\right\rangle \tag{24}$$

Given an ansätz, optimisation aims to find a set of parameters θ that minimises the right-hand side of Eqn 23.

The Hamiltonian of H_2, after second quantization and qubit mapping and reduction[1], is given as:

$$H = g_0 + g_1 Z_0 + g_2 Z_1 + g_3 Z_0 Z_1 + g_4 X_0 X_1 \tag{25}$$

The values of g_0 to g_4 differ at different H-H bond lengths. Those values can be found at TABLE 1 of `https://journals.aps.org/prx/supplemental/10.1103/PhysRevX.8.011021/Supplementary.pdf`, the supplemental of `https://journals.aps.org/pra/abstract/10.1103/PhysRevA.102.022427`. The variational circuit should be based on the following ansätz:

$$|\psi(\theta)\rangle = e^{-i\theta Y_0 X_1}|00\rangle \tag{26}$$

The circuit diagram is shown in Fig. 5.

The objective function to minimize is:

$$\begin{aligned}\mathcal{L}(\theta) = E_0(\theta) &= \langle H\rangle_\theta \\ &= \langle\psi(\theta)|\, H\,|\psi(\theta)\rangle \\ &= g_0\langle I_0 I_1\rangle_\theta + g_1\langle Z_0 I_1\rangle_\theta + g_2\langle I_0 Z_1\rangle_\theta + g_3\langle Z_0 Z_1\rangle_\theta + g_4\langle X_0 X_1\rangle_\theta\end{aligned} \tag{27}$$

[1]This process can be found in `https://journals.aps.org/prx/abstract/10.1103/PhysRevX.6.031007`, as well as a review paper at `https://doi.org/10.1021/acs.chemrev.8b00803`.

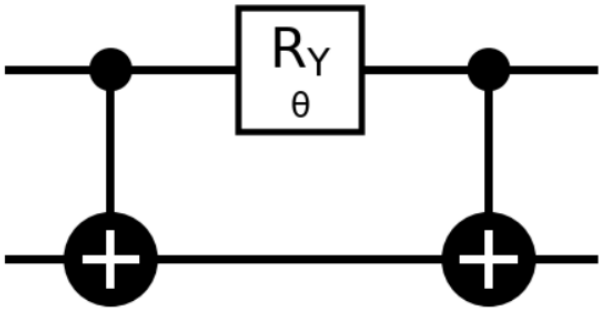

Figure 5: Variational circuit of $e^{-i\theta Y_0 X_1}$ for solving H_2 ground state energy problem

Question (a) Compose the circuit shown in Fig. 5. Draw the circuit diagram. Use it to find the ground state energy (minimal value of $\mathcal{L}(\theta)$) for g_0 to g_4 when the bond length is 0.75Å. Print out your result.

Question (b) Repeat what you did in (a), but for every bond length listed in TABLE 1 of `https://journals.aps.org/prx/supplemental/10.1103/PhysRevX.8.011021/Supplementary.pdf`. Plot the ground state energies against bond lengths. Find the bond length that gives the minimal energy. Print out your results.

3.3 Variational Quantum Linear Solver (2 marks = 0.5 + 1.5)

In this section, we will use variational quantum circuits to solve a simple set of linear equations. For a detailed description of VQLS, see `https://doi.org/10.48550/arXiv.1909.05820`. Relevant tutorials see `https://qiskit.org/textbook/ch-paper-implementations/vqls.html` and `https://pennylane.ai/qml/demos/tutorial_vqls.html`. The basic idea of VQLS is that given a matrix equation

$$\mathbf{A}x = b \tag{28}$$

we can construct a Hamiltonian[2]

$$H = A^{\dagger}(I - |b\rangle \langle b|)A \tag{29}$$

where $|b\rangle$ is the normalized version of b. Then with a variational circuit

$$|\psi(\theta)\rangle = U(\theta)\,|0\rangle \tag{30}$$

The objective function to be minimized is

$$\mathcal{L}(\theta) = \langle\psi(\theta)|\,H\,|\psi(\theta)\rangle \tag{31}$$

After the optimization is finished, the (normalized) solution x_i can be extracted with the probability of measuring $|i\rangle$. For a one-qubit case, $x_0 = \langle\psi|0\rangle\langle 0|\psi\rangle$ and $x_1 = \langle\psi|1\rangle\langle 1|\psi\rangle$.

For this section, we will consider the following simultaneous linear equations:

$$\begin{aligned} 1.2x - 0.3y &= 1 \\ -0.3x + 0.8y &= 1 \end{aligned} \tag{32}$$

[2]We only consider cases when $\mathbf{A}$ is Hermitian. Non-hermitian cases are slightly more complicated.

Question (a) Write the equations in Eqn. 32 in the form of matrix equation $\mathbf{A}x = b$. What is the solution to this set of linear equations? Show your working in a **Markdown** cell of your Jupyter Notebook. Write a Hamiltonian whose ground state corresponds to the solution of these equations.

Question (b) Using a simulator, optimise a trial circuit to find the ground state of this Hamiltonian, reporting results in the forms of plots and print-out in your Jupyter Notebook. Compare with the answer you found in (b). Do your answers agree? Comment on any discrepancies.