Rujipat Siyakhan 6532142421

Assignment III

1. **Finding Eigenvectors by Hand**
   This is a tedious exercise. But I want you to go through to practice the skill of finding eigenvectors. We can also measure the electron spin in $\hat{y}$ direction by setting up an external magnetic field in the $\hat{y}$ direction. It is given that the corresponding operator is $\sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$. Again, we are using the $|0\rangle/|1\rangle$ basis. Find the eigenvalues and eigenvectors in the $|0\rangle/|1\rangle$ basis.

2. **Finding Eigenvectors Using Python**
   Use Google Colab to check your answer. What you need is the function to find the eigenvectors and eigenvalues. The following code first builds the matrix and then find the eigenvectors and eigenvalues:

```
import numpy as np
sigmay=np.array([[0,-1j],[1j,0]])
print(sigmay)
eigenvalues, eigenvectors = np.linalg.eig(sigmay)
print(eigenvalues) print(eigenvectors)
```

3. **Pauli Matrix Multiplication by Hand and Using Python**
   Use hand calculation to show that ${\sigma_y}^2 = I$. Then verify with Colab using the following codes. Note that np.matmul is for matrix multiplication.

```
import numpy as np
sigmay=np.array([[0,-1j],[1j,0]])
print(sigmay)
print(np.matmul(sigmay, sigmay))
```

4. **Pauli Matrix Multiplication by Hand and Using Python**
   Repeat the previous question for ${\sigma_z}^2 = I$ using both hand calculation and Colab.

5. **Proof of Hermitianity**
   Show that σ_y is Hermitian. Hints: find ${\sigma_y}^{*T}$ and show that it is the same as σ_y. Verify using Colab.

6. **Operation of Paul Matrix on Bra and Ket**
   Find $\sigma_y|0\rangle$, $\langle 0|\sigma_y$. Are they the same? Why? Verify using Colab. You can use np.dot for matrix-vector multiplication. The following shows how to do $\sigma_y|0\rangle$.

```
zerospin=np.array([[1],[0]])
print(zerospin.shape)
np.dot(sigmay,zerospin)
```

1. Finding Eigenvectors by Hand

This is a tedious exercise. But I want you to go through to practice the skill of finding eigenvectors. We can also measure the electron spin in $\hat{y}$ direction by setting up an external magnetic field in the $\hat{y}$ direction. It is given that the corresponding operator is $\sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$. Again, we are using the $|0\rangle/|1\rangle$ basis. Find the eigenvalues and eigenvectors in the $|0\rangle/|1\rangle$ basis.

Assume: eigenvector $v = \begin{pmatrix} a \\ b \end{pmatrix}$, eigenvalue λ

$$\sigma_y v = \lambda v$$

$$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \lambda \begin{pmatrix} a \\ b \end{pmatrix}$$

$$\begin{pmatrix} -bi \\ ai \end{pmatrix} = \lambda \begin{pmatrix} a \\ b \end{pmatrix}$$

$$-bi = \lambda a \quad \dots (1)$$

$$ai = \lambda b \quad \dots (2)$$

From (2) sub (1): $-\left(\frac{a}{\lambda} i\right) i = \lambda a$

$$-\left(\frac{i^2}{\lambda}\right) a = \lambda a$$

$$(1)\, a = \lambda^2 a$$

$$1 = \lambda^2$$

$$\therefore \lambda = \pm 1 \quad \sim \text{Eigenvalue}$$

(1): $-bi = \lambda a$

$$b = \frac{\lambda a}{-i}$$

$$= \frac{\lambda}{-i} \times \frac{-i}{-i} \times a$$

$$\therefore b = \lambda a i$$

Thus $v = \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a \\ \lambda a i \end{pmatrix}$

Normalize eigenvector :

$$v' = \frac{v}{\sqrt{\sum_{i=0}^{n-1} |a_i|^2}}$$

$$= \frac{v}{\sqrt{|a|^2 + |\lambda a i|^2}}$$

$$= \frac{v}{\sqrt{a^2 + |\lambda|^2 |a|^2 |i|^2}}$$

$\lambda^2 = 1$: eigenvalue
$|i| = 1$

$$= \frac{v}{\sqrt{a^2 + (1) a^2 (1)}}$$

$$= \frac{v}{\sqrt{2} \cdot a}$$

$$v' = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ \lambda i \end{pmatrix} \quad ; \quad \lambda = \pm 1$$

∴ Eigenvalue $\lambda = +1, -1$ ✱

Eigenvector $v_{\lambda=1} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ i \end{pmatrix}$, $v_{\lambda=-1} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -i \end{pmatrix}$ ✱

2. Finding Eigenvectors Using Python

Use Google Colab to check your answer. What you need is the function to find the eigenvectors and eigenvalues. The following code first builds the matrix and then find the eigenvectors and eigenvalues:

```
import numpy as np
sigmay=np.array([[0,-1j],[1j,0]])
print(sigmay)
eigenvalues, eigenvectors = np.linalg.eig(sigmay)
print(eigenvalues) print(eigenvectors)
```

```python
import numpy as np

sigma = np.array([[0, -1j], [1j, 0]])
print(f"sigma: {sigma}")

eigvalue, eigvector = np.linalg.eig(sigma)
print(f"eigvalue: {eigvalue}")
print(f"eigvector: {eigvector}")
print(f"eigvalue[0]: {eigvalue[0]}")
print(f"eigvector[0]: {eigvector[0]}")
print(f"eigvalue[1]: {eigvalue[1]}")
print(f"eigvector[1]: {eigvector[1]}")
```

```
(quantum-computing) (base) pupipatsingkhorn@Pupipats-MacBook-Air quantum-computing % /Us
ting/.venv/bin/python /Users/pupipatsingkhorn/Developer/repositories/quantum-computing/a
sigma: [[ 0.+0.j -0.-1.j]
 [ 0.+1.j  0.+0.j]]
eigvalue: [ 1.+0.00000000e+00j -1.-1.23259516e-32j]
eigvector: [[-8.65956056e-17-0.70710678j  7.07106781e-01+0.j        ]
 [ 7.07106781e-01+0.j        -4.96458202e-17-0.70710678j]]
eigvalue[0]: (0.9999999999999996+0j)
eigvector[0]: [-8.65956056e-17-0.70710678j  7.07106781e-01+0.j        ]
eigvalue[1]: (-0.9999999999999999-1.232595164407831e-32j)
eigvector[1]: [ 7.07106781e-01+0.j        -4.96458202e-17-0.70710678j]
(quantum-computing) (base) pupipatsingkhorn@Pupipats-MacBook-Air quantum-computing %
```

Result from python:

$$\lambda = 1, -1$$

$$v = \begin{pmatrix} -\frac{1}{\sqrt{2}}i \\ -\frac{1}{\sqrt{2}} \end{pmatrix}, \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}}i \end{pmatrix}$$

$$\therefore v = \frac{1}{\sqrt{2}}\begin{pmatrix} -i \\ -1 \end{pmatrix}, \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ -i \end{pmatrix}$$

Since the answer from python and Q1. is not the same

We will show that both answer is correct due to global phase equivalent.

Two normalized vectors $|\phi\rangle$ and $|\psi\rangle$ represent the same physical state iff.

there exists a phase $e^{i\theta}$ with: $|\phi\rangle = e^{i\theta}|\psi\rangle$

$\lambda = 1$:

$$Q1\, v_{\lambda=1} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ i \end{pmatrix}, \qquad Q2\, v_{\lambda=1} = \frac{1}{\sqrt{2}}\begin{pmatrix} -i \\ 1 \end{pmatrix}$$

$$(-i)\, Q1\, v_{\lambda=1} = (-i)\cdot\frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ i \end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix} -i \\ 1 \end{pmatrix}$$

Euler: $-i = 0 + i(-1)$

$\rightarrow \cos\left(-\frac{\pi}{2}\right) + i\sin\left(-\frac{\pi}{2}\right)$

$\rightarrow e^{i\left(-\frac{\pi}{2}\right)}$

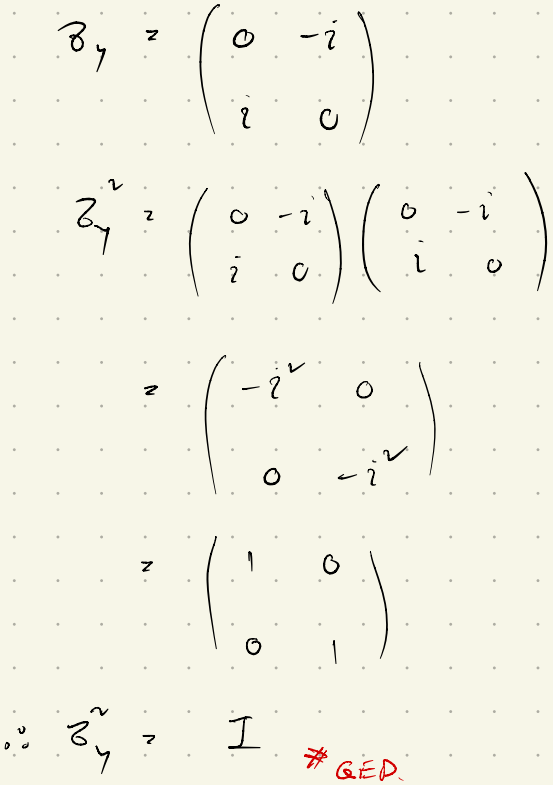
$$\therefore Q2\, v_{\lambda=1} = e^{i\left(-\frac{\pi}{2}\right)}\, Q1\, v_{\lambda=1}$$

$\therefore$ Both vectors represent the same physical state. $\rightarrow$ Both Ans. correct.

3. Pauli Matrix Multiplication by Hand and Using Python

Use hand calculation to show that $\sigma_y{}^2 = I$. Then verify with Colab using the following codes. Note that np.matmul is for matrix multiplication.

```
import numpy as np
sigmay=np.array([[0,-1j],[1j,0]])
print(sigmay)
print(np.matmul(sigmay, sigmay))
```



$\therefore \sigma_y^2 = I$ #QED

```
import numpy as np

sigmay = np.array([[0, -1j], [1j, 0]])
print(np.matmul(sigmay, sigmay))
```

```
(quantum-computing) (base) pupipat
uting/assignmenats-03/q3.py
[[1.+0.j 0.+0.j]
 [0.+0.j 1.+0.j]]
(quantum-computing) (base) pupipat
```

4. Pauli Matrix Multiplication by Hand and Using Python

Repeat the previous question for $\sigma_z{}^2 = I$ using both hand calculation and Colab.

$$\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$\sigma_z^2 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\therefore \sigma_z^2 = I$$ ✱ QED.

```python
import numpy as np

sigmaz = np.array([[1, 0], [0, -1]])
print(np.matmul(sigmaz, sigmaz))
```

```
(quantum-computing) (base) p
uting/assignmenats-03/q4.py
[[1 0]
 [0 1]]
(quantum-computing) (base) p
```

5. Proof of Hermitianity

Show that σ_y is Hermitian. Hints: find $\sigma_y{}^{*T}$ and show that it is the same as σ_y. Verify using Colab.

$$\sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$$

$$\sigma_y^* = \begin{bmatrix} 0 & +i \\ -i & 0 \end{bmatrix}$$

$$\sigma_y^{*T} = \begin{bmatrix} 0 & -i \\ +i & 0 \end{bmatrix}$$

$\therefore \sigma_y^{*T} = \sigma_y$ Thus for σ_y is Hermitian. ∎ QED.

```python
import numpy as np

sigmay = np.array([[0, -1j], [1j, 0]], dtype=complex)

sigmay_dagger = sigmay.conj().T

print("σ_y:\n", sigmay)
print("\nσ_y^†:\n", sigmay_dagger)
print("\nHermitian? ->", np.allclose(sigmay,
sigmay_dagger))
```

```
(quantum-compu  Debug Console
uting/assignme
σ_y:
 [[ 0.+0.j -0.-1.j]
 [ 0.+1.j  0.+0.j]]

σ_y^†:
 [[ 0.-0.j  0.-1.j]
 [-0.+1.j  0.-0.j]]

Hermitian? -> True
(quantum-computing) (base) pu
```

6. Operation of Paul Matrix on Bra and Ket

Find $\sigma_y|0\rangle$, $\langle 0|\sigma_y$. Are they the same? Why? Verify using Colab. You can use np.dot for matrix-vector multiplication. The following shows how to do $\sigma_y|0\rangle$.

```
zerospin=np.array([[1],[0]])
print(zerospin.shape)
np.dot(sigmay,zerospin)
```

$$\sigma_y|0\rangle = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ i \end{pmatrix}$$

$$\langle 0|\sigma_y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} = \begin{bmatrix} 0 & -i \end{bmatrix}$$

$\therefore \sigma_y|0\rangle \neq \langle 0|\sigma_y$ ※

But $(\sigma_y|0\rangle)^\dagger = [0 \quad -i] = \langle 0|\sigma_y$ i.e. they are Hermitian conjugates of each other. ※

```python
import numpy as np

# Define sigma_y and basis states
sigmay = np.array([[0, -1j], [1j, 0]], dtype=complex)

ket0 = np.array([[1.0], [0.0]])  # |0>
bra0 = ket0.conj().T  # <0|

# σ_y |0>
ket_result = sigmay @ ket0
print("σ_y |0> =\n", ket_result)  # expect [[0],[ i]]

# <0| σ_y
bra_result = bra0 @ sigmay
print("<0| σ_y =\n", bra_result)  # expect [[0, -i]]

print("Are σ_y|0> and <0|σ_y equal? ->",
np.allclose(ket_result, bra_result))

# Check the Hermitian-conjugate relation:
lhs = ket_result.conj().T  # (σ_y |0>)^†
rhs = bra_result  # <0| σ_y
print("Are (σ_y|0>)^† and <0|σ_y equal? ->",
np.allclose(lhs, rhs))
```

```
(quantum-computing) (base) pupipatsingkhorn
uting/assignments-03/q6.py
σ_y |0> =
 [[0.+0.j]
 [0.+1.j]]
<0| σ_y =
 [[0.+0.j 0.-1.j]]
Are σ_y|0> and <0|σ_y equal? -> False
Are (σ_y|0>)^† and <0|σ_y equal? -> True
(quantum-computing) (base) pupipatsingkhorn
```