1.

Find the eigenvalues, eigenvectors, and diagonal representation of the Pauli matrices. Solve the first by hand, you may do the other 2 with (commented) computer calcs:

$$\hat{X} = \hat{\sigma_1} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \hat{Y} = \hat{\sigma_2} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \hat{Z} = \hat{\sigma_3} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Calculate $\hat{X}$ by hand. First, use $\det(A - \lambda I) = 0$ to find the eigenvalues.

$$\det \begin{pmatrix} 0-\lambda & 1 \\ 1 & 0-\lambda \end{pmatrix} = \lambda^2 - 1 = 0 \tag{1}$$

$$\lambda = \pm 1 \tag{2}$$

Second, use $\hat{A}\vec{v} = \lambda \vec{v}$ to find the eigenvectors.

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} v_2 \\ v_1 \end{bmatrix} \tag{3}$$

$$= +1 \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \implies v_1 = v_2 \tag{4}$$

$$= -1 \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \implies v_1 = -v_2 \tag{5}$$

$$\tag{6}$$

Finally, use normalization, $\vec{v} \cdot \vec{v} = 1$ to find the eigenvectors.

$$v_{x+} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad v_{x-} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{7}$$

Third, use the eigenvectors to find the diagonal representation. $S^{-1}AS = \Lambda$, where S is eigenvector matrix. To invert S, use simple 2x2 matrix inversion rule $A^{-1} = \frac{1}{|A|} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$.

$$\Lambda = S^{-1}AS = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}^{-1}\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \tag{8}$$

$$= \frac{1}{2}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \tag{9}$$

$$= \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix} \tag{10}$$

Repeat for Y and Z using Python.

$$\hat{Y}, \quad \lambda = \pm 1, \quad \vec{v}_{y+} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\ i\end{bmatrix}, \quad \vec{v}_{y-} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\ -i\end{bmatrix}, \quad \Lambda = \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix} \tag{11}$$

$$\hat{Z}, \quad \lambda = \pm 1, \quad \vec{v}_{z+} = \begin{bmatrix}1\\ 0\end{bmatrix}, \quad \vec{v}_{z-} = \begin{bmatrix}0\\ 1\end{bmatrix}, \quad \Lambda = \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix} \tag{12}$$

```python
import qutip
import numpy as np

# prettier printing
def pprint(matrix):
    return '\n'.join(['\t'.join([str(cell) for cell in row]) for row in
     ↪  matrix])

# get pauli matrices
sx = qutip.sigmax()
sy = qutip.sigmay()
sz = qutip.sigmaz()

# for each pauli, print it's eigenvalues and eigenvectors
for op in [sx, sy, sz]:
    print("Operator:\n {}".format(pprint(op)))

    # qutip built-in method
    eigvals, eigvecs = op.eigenstates()

    print("Eigenvalues:\n {}".format(eigvals))
    print("Eigenvectors:\n {}".format(pprint(eigvecs)))
    # qutip built-in method for diagonalization
    # eigvals_diag = np.diag(eigvals)

    # calculate diagonal matrix explicitly using eigenvector matrix
    eigvec_matrix = np.array([vec.full().T[0] for vec in eigvecs]).T
    eigvals_diag = np.linalg.inv(eigvec_matrix) @ op.full() @ eigvec_matrix
    print("Diagonal matrix of eigenvalues:\n
     ↪  {}\n".format(pprint(eigvals_diag)))
```

2.

Determine if the matrices given below are unitary and Hermitian. Solve the first by hand, you may use computer for the rest. For all, explain as best you can why these matrices have the properties they do.

Check if the matrix is unitary by checking if $U^\dagger U = I$ and $UU^\dagger = I$. Check if the matrix is Hermitian by checking if $A = A^\dagger$. To explain unitarity, the matrix will preserve norms and consequently the matrix should have orthonormal rows/columns. To explain why a matrix is Hermitian, a requirement is that no imaginary terms are on the diagonal (or needs to swap with its conjugate), in other words, the observables must be real-valued. These properties can be used to understand why the following matrices may be unitary or Hermitian.

If an operator is unitary, then it's conjugate transpose can be used to "reverse" the operator. If an operator is Hermitian, then it is it's own reverse.

a. Pauli matrices

$$\hat{X}^\dagger \hat{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I \tag{13}$$

$$\hat{X}\hat{X}^\dagger = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I \tag{14}$$

$$\hat{X} = \hat{X}^\dagger \tag{15}$$

$$\hat{Y}^\dagger\hat{Y} = \begin{bmatrix}0 & -i\\ i & 0\end{bmatrix}\begin{bmatrix}0 & -i\\ i & 0\end{bmatrix} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} = I \tag{16}$$

$$\hat{Y}\hat{Y}^\dagger = \begin{bmatrix}0 & -i\\ i & 0\end{bmatrix}\begin{bmatrix}0 & -i\\ i & 0\end{bmatrix} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} = I \tag{17}$$

$$\hat{Y} = \hat{Y}^\dagger \tag{18}$$

$$\hat{Z}^\dagger\hat{Z} = \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} = I \tag{19}$$

$$\hat{Z}\hat{Z}^\dagger = \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} = I \tag{20}$$

$$\hat{Z} = \hat{Z}^\dagger \tag{21}$$

For each Pauli, Unitary: True, Hermitian: True

b. $\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}$

U: True, H: True

c. $\begin{bmatrix}1 & 0\\ 0 & i\end{bmatrix}$

True, False

d. $\begin{bmatrix}1 & 0\\ 0 & \exp\left(\frac{i\pi}{4}\right)\end{bmatrix}$

True, False

e. $\begin{bmatrix} 1 & e^{i\theta} \\ e^{-i\theta} & 1 \end{bmatrix}$

$$\hat{A}^\dagger \hat{A} = \begin{bmatrix} 1 & e^{i\theta} \\ e^{-i\theta} & 1 \end{bmatrix} \begin{bmatrix} 1 & e^{-i\theta} \\ e^{i\theta} & 1 \end{bmatrix} = \begin{bmatrix} 2 & 2e^{i\theta} \\ 2e^{-i\theta} & 2 \end{bmatrix} \neq I \tag{22}$$

False, True

f. $\begin{bmatrix} 1 & 1+i \\ 1+i & 0 \end{bmatrix}$

False, False

```python
def check_unitary_hermitian(op):
    # qutip built-in methods
    print("Is unitary: {}".format(op.isunitary))
    print("Is hermitian: {}\n".format(op.isherm))

    # calculate unitary and hermitian explicitly
    # check if unitary by checking if self * self^dagger = identity
    print("Is unitary: {}".format(np.allclose(op.full() @ op.dag().full(),
    ↪  np.eye(2))))
    # check if hermitian by checking if self^dagger = self
    print("Is hermitian: {}\n".format(np.allclose(op.dag().full(),
    ↪  op.full())))

if __name__ == '__main__':
    #a
    print("a")
    op_list = [qutip.sigmax(), qutip.sigmay(), qutip.sigmaz()]
    list(map(check_unitary_hermitian, op_list))
    #b
    print("b")
    check_unitary_hermitian(1/np.sqrt(2) * qutip.Qobj([[1, 1], [1, -1]]))
    #c
    print("c")
    check_unitary_hermitian(qutip.Qobj([[1, 0], [0, 1j]]))
    #d
    print("d")
    check_unitary_hermitian(qutip.Qobj([[1, 0], [0, np.exp(1j * np.pi /
↪   4)]]))
    #e
    print("e")
    theta = np.random.uniform(0, 2 * np.pi) #rest of proof in doc
    check_unitary_hermitian(qutip.Qobj([[1, np.exp(1j * theta)],
↪   [np.exp(-1j* theta), 1]]))
    #f
    print("f")
    check_unitary_hermitian(qutip.Qobj([[1, 1+1j], [1+1j, 0]]))
```

3.

Given the unitary matrices below, calculate $\hat{U}_3\hat{U}_2\hat{U}_1$ and $\hat{U}_1\hat{U}_2\hat{U}_3$. What does this tell you about matrix multiplication?

$$\hat{U}_1 = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \quad \hat{U}_2 = \begin{bmatrix} 1 & i \\ -i & 1 \end{bmatrix}, \quad \hat{U}_3 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix},$$

Matrix multiplication is not commutative. It can also be shown that multiplication of unitary matrices preserves the unitary property; however, in this case $\hat{U}_3$ is not unitary.

$$\hat{U}_3\hat{U}_2\hat{U}_1 = \begin{bmatrix} 1-i & 1+i \\ 1+i & -1-i \end{bmatrix} \tag{23}$$

$$\hat{U}_1\hat{U}_2\hat{U}_3 = \begin{bmatrix} 1+i & 1-i \\ 1+i & 1-i \end{bmatrix} \tag{24}$$

```
# setup
u1 = qutip.Qobj([[1, 0], [0, 1j]])
check_unitary_hermitian(u1)
u2 =qutip.Qobj([[1, 1j], [-1j, 1]])
check_unitary_hermitian(u2)
u3 = qutip.Qobj([[1, 1], [1, -1]])
```

```
check_unitary_hermitian(u3)

# calc u3u2u1
u = u3 * u2 * u1
print(u)
check_unitary_hermitian(u)

# calc u1u2u3
u = u1 * u2 * u3
print(u)
check_unitary_hermitian(u)
```

4.

The $\hat{X}$ Pauli matrix is a π rotation about the x-axis, what is the matrix representation of a $\pi/2$ rotation about the x-axis?

Use this equation $R_x(\theta) = \begin{bmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{bmatrix}$, where θ is the angle of rotation.

$$R_x(\pi/2) = \begin{bmatrix} \cos(\pi/4) & -i\sin(\pi/4) \\ -i\sin(\pi/4) & \cos(\pi/4) \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix} \tag{25}$$

```
from qiskit.circuit.library import RXGate
print(RXGate(np.pi/2).to_matrix())
```

5.

Using the Pauli matrices, $\hat{X}$, $\hat{Y}$, and $\hat{Z}$ given above, calculate $\hat{X}\hat{X}$, $\hat{Y}\hat{Y}$, $\hat{Z}\hat{Z}$, $\hat{X}\hat{Y}$, $\hat{Y}\hat{Z}$, and $\hat{Z}\hat{X}$. Up to a constant, what can we infer about the matrix product $\hat{\sigma_i}\hat{\sigma_j} = f(\hat{\sigma_i}, \hat{\sigma_j}, \hat{\sigma_k})$?

$$\hat{X}\hat{X} = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} \tag{26}$$

$$\hat{Y}\hat{Y} = \begin{bmatrix}0 & -i\\i & 0\end{bmatrix}\begin{bmatrix}0 & -i\\i & 0\end{bmatrix} = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} \tag{27}$$

$$\hat{Z}\hat{Z} = \begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}\begin{bmatrix}1 & 0\\0 & -1\end{bmatrix} = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} \tag{28}$$

$$\hat{X}\hat{Y} = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}0 & -i\\i & 0\end{bmatrix} = \begin{bmatrix}i & 0\\0 & -i\end{bmatrix} \tag{29}$$

$$\hat{Y}\hat{Z} = \begin{bmatrix}0 & -i\\i & 0\end{bmatrix}\begin{bmatrix}1 & 0\\0 & -1\end{bmatrix} = \begin{bmatrix}0 & i\\i & 0\end{bmatrix} \tag{30}$$

$$\hat{Z}\hat{X} = \begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} = \begin{bmatrix}0 & 1\\-1 & 0\end{bmatrix} \tag{31}$$

Generallly written as,

$$\hat{\sigma_j}\hat{\sigma_k} = \delta_{jk}\hat{I} + i\sum_{l=1}^{3}\epsilon_{jkl}\hat{\sigma_l} \tag{32}$$

```
# calculate the pauli products
    # xx, yy, zz, xy, yz, zx
    pauli_products = [
        qutip.sigmax() * qutip.sigmax(),
        qutip.sigmay() * qutip.sigmay(),
        qutip.sigmaz() * qutip.sigmaz(),
        qutip.sigmax() * qutip.sigmay(),
        qutip.sigmay() * qutip.sigmaz(),
        qutip.sigmaz() * qutip.sigmax()
    ]

    # print the pauli products
    for op in pauli_products:
        print(op)
```