Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Monterrey

MA3014
Estancia de Investigación Control de Qbits para Cómputo Cuántico
(Grupo 577)

Actividad: Qubit Control 1

Profesores:

PhD. Marco Benjamín Enríquez Flores
PhD. Alfonso Isaac Jaimes Nájera

Alumna:

Andrea Catalina Fernández Mena | A01197705

Monterrey, México. 18 de Agosto del 2023

Follow the instructions

Quantum objects must be controlled in order to implement some operations on them. The purpose of this activity is to model the dynamics of a qubit interacting with a uniformed magnetic field. This constitutes a first step in the quantum control to implement quantum gates.

The energy of a qubit is electric field can be modeled using the following operator

$$H = \frac{\Delta}{2} Z + V(t) X_+ + \overline{V}(t) X_-$$

Where the operators

$X_+ = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$ and $X_- = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$

The constant Δ might represent the magnitude a constant field in the z-direction and the complex time-dependent function V is known as the control field that will be useful to manipulate the dynamics.

In order to find the time-evolution of the qubit the Schrödinger equation

$$i \frac{d}{dt} |\psi\rangle = H |\psi\rangle$$

(Note that we are in a free-units system) This is second order matrix differential equation. This means that given an initial state

$|\psi(0)\rangle = a_0 |0\rangle + b_0 |1\rangle$, where $a_0 = \cos\theta_0, \quad b_0 = e^{i\varphi_0} \sin\theta_0$ (0)

The solution of equation () will provide the state of the system at any time, that is to say,

$$|\psi(t)\rangle = a(t)|0\rangle + b(t)|1\rangle$$

Where the coefficients as time-dependent functions provide a curve in the Bloch sphere.

The Schrödinger equation is not exactly solvable for any control field V. Thus, in this activity we provide a solution for the simple case $V(t) = 0$, that is to say,

$$H = \frac{\Delta}{2} Z \tag{1}$$

The correspondent time-dependent qubit can be computed as

$$|\psi(t)\rangle = U(t)|\psi(0)\rangle$$

Where U is the time-evolution operator and reads

$$U(t) = \exp(-iHt) \tag{2}$$

A) Using the Taylor series of the exponential function show that U can be written as

$$U(t) = \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k}}{(2k)!} H^{2k} - i \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k+1}}{(2k+1)!} H^{2k+1} \tag{3}$$

Andrea Catalina Fernández Mena A01197705

Control de Qbits 1

A) Using the Taylor series of the exponential function show that U can be written as:

$$U(t) = \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k}}{(2k)!} H^{2k} - i \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k+1}}{(2k+1)!} H^{2k+1} \quad (3)$$

Note:
U is the time-evolution operator and its defined as

$$U(t) = \exp(-iHt) = e^{-iHt}$$

Usaremos la serie de Taylor ó MacLaurin fuction para funciones exponenciales de la siguiente manera

$$e^x = \cancel{\frac{x^0}{0!}}\, 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots \quad \text{desde } [0, \infty)$$

$x = -iHt$

$$e^{-iHt} = \frac{(-iHt)^0}{0!} + \frac{(-iHt)}{1!} + \frac{(-iHt)^2}{2!} + \frac{(-iHt)^3}{3!} + \dots$$

Para facilitar la estructura de la solución separaremos la operación en la suma de la agrupación de todos los pares y de los impares.

$$= \left(\frac{(-iHt)^0}{0!} + \frac{(-iHt)^2}{2!} + \frac{(-iHt)^4}{4!} \right) + \left(\frac{(-iHt)}{1!} + \frac{(-iHt)^3}{3!} + \frac{(-iHt)^5}{5!} \right)$$

$$= \left((-1)^0 + \frac{(-1)^2 H^2 t^2}{2!} + \frac{(-1)^4 H^4 t^4}{4!} \right) + i \left(\frac{(-1) Ht}{1!} + \frac{(-1)^3 H^3 t^3}{3!} + \frac{(-1)^5 H^5 t^5}{5!} \right)$$

Igualamos valores de (-1) para ambos conjuntos

$$= \left((-1)^0 + \frac{(-1)^2 H^2 t^2}{2!} + \frac{(-1)^4 H^4 t^4}{4!} \right) + i(-1) \left(\frac{(-1)^0 Ht}{1!} + \frac{(-1)^2 H^3 t^3}{3!} + \frac{(-1)^4 H^5 t^5}{5!} \right)$$

$$= \left((-1)^0 + \frac{(-1)^2 H^2 t^2}{2!} + \frac{(-1)^4 H^4 t^4}{4!} \right) - i \left(\frac{(-1)^0 Ht}{1!} + \frac{(-1)^2 H^3 t^3}{3!} + \frac{(-1)^4 H^5 t^5}{5!} \right)$$

Los números pares podemos expresarlos como $\sum_{k=0}^{\infty} = 2k$
y los impares como $\sum_{k=0}^{\infty} = 2k+1$
Por lo que podemos definir:

$$\boxed{= \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k}}{(2k)!} H^{2k} - i \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k+1}}{(2k+1)!} H^{2k+1}}$$

B) Write the operator $\boldsymbol{H}$ in its matrix form and show

$$H^{2k} = \begin{pmatrix} \Delta^{2k}/2^{2k} & 0 \\ 0 & \Delta^{2k}/2^{2k} \end{pmatrix}$$

$$H^{2k+1} = \begin{pmatrix} \Delta^{2k+1}/2^{2k+1} & 0 \\ 0 & -\Delta^{2k+1}/2^{2k+1} \end{pmatrix}$$

B) Write the operator H in its matrix form and show

$$H^{2k} = \begin{pmatrix} \frac{\Delta^{2k}}{2^{2k}} & 0 \\ 0 & \frac{\Delta^{2k}}{2^{2k}} \end{pmatrix}$$

$$H^{2k+1} = \begin{pmatrix} \frac{\Delta^{2k+1}}{2^{2k+1}} & 0 \\ 0 & \frac{\Delta^{2k+1}}{2^{2k+1}} \end{pmatrix}$$

Considering that

$V(t) = 0$

$$H = \frac{\Delta}{2}\cdot Z + \cancel{V(t)X_+}^{\,0} + \cancel{V^*(t)X_-}^{\,0}$$

$$H = \frac{\Delta}{2}\cdot Z$$

← Producto con Matriz de Pauli

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

$$H = \frac{\Delta}{2}\cdot\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} \frac{\Delta}{2} & 0 \\ 0 & -\frac{\Delta}{2} \end{pmatrix}$$

Tomando en cuenta la propiedad de exponentes aplicados a matrices con diagonal:

$$H^x = \begin{pmatrix} a^x & 0 \\ 0 & b^x \end{pmatrix}$$

entonces:

$$H^{2k} = \begin{pmatrix} \left(\frac{\Delta}{2}\right)^{2k} & 0 \\ 0 & \left(-\frac{\Delta}{2}\right)^{2k} \end{pmatrix} \qquad H^{2k+1} = \begin{pmatrix} \left(\frac{\Delta}{2}\right)^{2k+1} & 0 \\ 0 & \left(-\frac{\Delta}{2}\right)^{2k+1} \end{pmatrix}$$

Realizamos expansión de los exponentes (exp 2k cambia signo debido a la doble negación)

$$H^{2k} = \begin{pmatrix} \frac{\Delta^{2k}}{2^{2k}} & 0 \\ 0 & \frac{\Delta^{2k}}{2^{2k}} \end{pmatrix} \qquad H^{2k+1} = \begin{pmatrix} \frac{\Delta^{2k+1}}{2^{2k+1}} & 0 \\ 0 & -\frac{\Delta^{2k+1}}{2^{2k+1}} \end{pmatrix}$$

C) Substitute the former expressions into (3) and show that the time-evolution operator can be written as

$$U(t) = \begin{pmatrix} \cos(\Delta t/2) + i\sin(\Delta t/2) & 0 \\ 0 & \cos(\Delta t/2) - i\sin(\Delta t/2) \end{pmatrix}$$

(You should remember which the infinite series of functions cos and sin)

c) Substitute the former expressions into (3) and show that the time-evolution operator can be written as:

$$U(t) = \begin{pmatrix} \cos(\Delta t/2) + i\sin(\Delta t/2) & 0 \\ 0 & \cos(\Delta t/2) - i\sin(\Delta t/2) \end{pmatrix}$$

Sustitute expressions from B) into equation from A)

$$U(t) = \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k}}{(2k)!} H^{2k} - i \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k+1}}{(2k+1)} H^{2k+1}$$

$$H^{2k} = \begin{pmatrix} \frac{\Delta^{2k}}{2^{2k}} & 0 \\ 0 & \frac{\Delta^{2k}}{2^{2k}} \end{pmatrix} \qquad H^{2k+1} = \begin{pmatrix} \frac{\Delta^{2k+1}}{2^{2k+1}} & 0 \\ 0 & \frac{\Delta^{2k+1}}{2^{2k+1}} \end{pmatrix}$$

Do sustitution:

$$U(t) = \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k}}{(2k)!} \begin{pmatrix} \frac{\Delta^{2k}}{2^{2k}} & 0 \\ 0 & \frac{\Delta^{2k}}{2^{2k}} \end{pmatrix} - i \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k+1}}{(2k+1)!} \begin{pmatrix} \frac{\Delta^{2k+1}}{2^{2k+1}} & 0 \\ 0 & -\frac{\Delta^{2k+1}}{2^{2k+1}} \end{pmatrix}$$

We do the corresponding operations (multiply):

$$= \begin{pmatrix} \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k}}{(2k)!} \left(\frac{\Delta^{2k}}{2^{2k}}\right) & 0 \\ 0 & \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k}}{(2k)!} \left(\frac{\Delta^{2k}}{2^{2k}}\right) \end{pmatrix} - i \begin{pmatrix} \sum_{k=0}^{\infty} \frac{(-1)^k t^{2k+1}}{(2k+1)!} \left(\frac{\Delta^{2k+1}}{2^{2k+1}}\right) & 0 \\ 0 & -\sum_{k=0}^{\infty} \frac{(-1)^k t^{2k+1}}{(2k+1)!} \left(\frac{\Delta^{2k+1}}{2^{2k+1}}\right) \end{pmatrix}$$

negative

Simplifyed with Taylor series:

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{3!} \qquad \& \qquad \sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} + \dots$$

(paired values) (odd values)

$$= \begin{pmatrix} \cos(t\Delta/2) & 0 \\ 0 & \cos(t\Delta/2) \end{pmatrix} - i \begin{pmatrix} \sin(t\Delta/2) & 0 \\ 0 & -\sin(t\Delta/2) \end{pmatrix}$$

Simplificando valores de matrices:

$$= \begin{pmatrix} \cos(\Delta t/2) - i\sin(\Delta t/2) & \emptyset \\ \emptyset & \cos(\Delta t/2) + i\sin(\Delta t/2) \end{pmatrix}$$

Dicha respuesta es equivalente a la esperada debido a la regla trigonometrica de negative angle identities. (Y un pequeño error de dedo en la act jeje)

D) Apply the operator *U* to the state (0), and find the Bloch vector coordinates

D) Apply the operator U to the state (0), and find the Bloch vector coordinates

state (0):

$$|\psi(0)\rangle = a_0|0\rangle + b_0|1\rangle, \quad \text{where } a_0 = \cos\frac{\theta_0}{2}, \quad b_0 = e^{i\varphi_0}\sin\frac{\theta_0}{2}$$

Time dependent equation $\longrightarrow |\psi(t)\rangle = U(t)|\psi(0)\rangle$

$$U(t) = \begin{pmatrix} \cos(\Delta t/2) - i\sin(\Delta t/2) & 0 \\ 0 & \cos(\Delta t/2) + i\sin(\Delta t/2) \end{pmatrix}$$

State (0) is represented as

$$\psi(0) = \begin{pmatrix} a_0 \\ b_0 \end{pmatrix} = \begin{pmatrix} \cos\left(\frac{\theta}{2}_0\right) \\ e^{i\varphi}\sin\left(\frac{\theta}{2}_0\right) \end{pmatrix}$$

Con base a dichos planteamientos y con el fin de optimizar el tiempo de desarrollo de dicha aplicación de *U* se desarrolló el siguiente código en Python utilizando los puntos antes mencionados para obtener el estado del Qubit tiempo-dependiente, así como las coordenadas <X>,<Y>,y <Z> del Vector aplicado a la Esfera de Bloch.

```
#Inciso D) Apply the operator U to the state (0), and find the Bloch
vector coordinates

#Andrea Catalina Fernández Mena

import numpy as np
```

```
#Valores iniciales

time = 0
theta = np.pi/4
phi = 0
delta = 1

#Representar valores iniciales del ket

a = np.cos(theta/ 2 )
b=  np.exp(1j * phi) * np.sin(theta/2)  #1j representa valores
imaginarios

#Definir U(t)
Ut = np.array([ [(np.cos(delta*time)/ 2) - 1j*np.sin((delta*time)/ 2) ,
0] , [0 ,(np.cos(delta*time)/ 2) + 1j*np.sin((delta*time)/ 2) ] ])

#Calcula el estado time-dependent del Qbit
   #Ket puede representarse como una matriz
   #Np.dot sirve para hacer dot producto multiplicacion de matrices
psi_t = np.dot(Ut, np.array([a,b]))

print("Time dependent qubit state: ", psi_t)

#Calculate the Bloch Vector Component
nX = 2* np.real(np.conj(psi_t[0]) * psi_t[1])  #posicion 0 es A y
posicion 1 es B
nY = 2* np.imag(np.conj(psi_t[0]) * psi_t[1])  #posicion 0 es A y
posicion 1 es B
nZ = np.abs(psi_t[0])**2 - np.abs(psi_t[1])**2

print("Bloch vector coordinates")
print("Nx = ", nX)
print("Ny = ", nY)
print("Nz = ", nZ)
```

Dicho código imprimiendo como resultados, los cuales representar coordenadas para A y B en el caso del estado del Qubit tiempo-dependiente, arroja las coordenadas de la esfera de Bloch en decimales con un valor equivalente a sus identidades trigonométricas correspondientes:

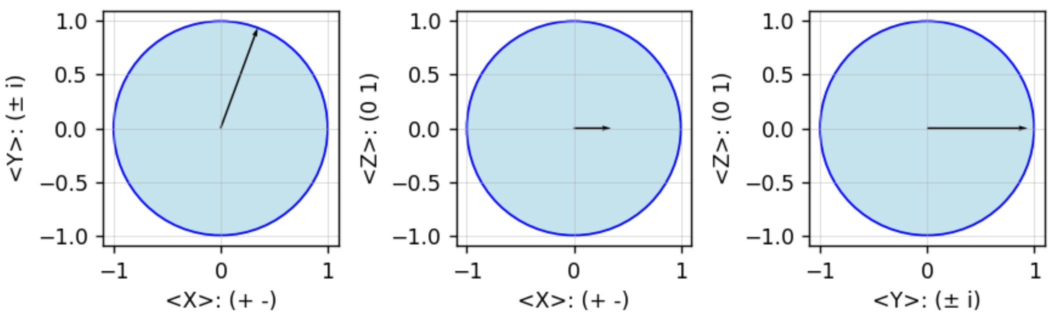
```
PS D:\Documentos\Septimo Semestre\Control de QBits para Cómputo Cuántico\Act2_A01197705> python -u "d:\Documentos\Septimo Semestre\Control de QBits para Cómputo Cuántico\Act2_A01197705\incisoD_A01197705.py"
Time dependent qubit state:  [0.46193977+0.j 0.19134172+0.j]
Bloch vector coordinates
Nx =  0.1767766952966369
Ny =  0.0
Nz =  0.17677669529663687
PS D:\Documentos\Septimo Semestre\Control de QBits para Cómputo Cuántico\Act2_A01197705>
```

E) Generate a code (Matlab, Pyton, Mathematica) to make an animation of the dynamics. Note that for each value of the time a different point in the Bloch sphere can be represented. The sequence of points will determine a trajectory in the sphere. To test your code you may use $\theta = \pi/4$, $\varphi = 0$ and $\Delta = 1$.

```
#Andrea Catalina Fernández Mena A01197705

#Bloch Sphere Dynamics Animation

import numpy as np
import matplotlib.pyplot as plt
from matplotlib.patches import Circle
from matplotlib.animation import FuncAnimation

# Condiciones iniciales para esfera de Bloch (test clauses)
delta = 1
theta = np.pi / 4
phi = 0
a = np.cos(theta)
b = np.exp(1j * phi) * np.sin(theta)

# Guardar funcionalidades de Inciso D para coordenadas dentro de
funciones lambda
U = lambda t: np.array([[np.cos((delta * t) / 2) - 1j * np.sin((delta *
t) / 2), 0],[0, np.cos((delta * t) / 2) + 1j * np.sin((delta * t) /
2)]])
psi = lambda uT: np.dot(uT, np.array([a, b]))
bloch = lambda a, b: [2*np.real(np.conj(a)*b), 2*np.imag(np.conj(a)*b),
np.abs(a)**2 - np.abs(b)**2]

#Incluir funciones lamda dentro de un método funcion
def bloch_Psi_Plot(t):
  uT = U(t)
  a, b = psi(uT)
  n = bloch(a, b)
  return n

# Subplots para proyecciones de  X-Y, X-Z, & Y-Z
```

```
fig, axes = plt.subplots(1, 3, figsize=(8, 4))
plt.suptitle('Bloch Sphere Dynamics Animation - A01197705',
fontsize=13)
plt.subplots_adjust(wspace=0.5)

#Actualización de frames para posicionamiento
def update(frame):
    t = frame / ratio_frames
    n_vector = bloch_Psi_Plot(t)
    projections = [(axes[0], "<X>: (+ -)", "<Y>: (± i)", [n_vector[0],
n_vector[1]]),
                   (axes[1], "<X>: (+ -)", "<Z>: (0 1)", [n_vector[0],
n_vector[2]]),
                   (axes[2], "<Y>: (± i)", "<Z>: (0 1)", [n_vector[1],
n_vector[2]])]

    for ax, x_label, y_label, vector in projections:
        ax.clear()
        ax.add_patch(Circle((0, 0), 1, fill=True, color='lightblue',
alpha=0.7))
        ax.add_patch(Circle((0, 0), 1, fill=False, color='b', alpha=1))
        ax.set_xlim([-1.1, 1.1])
        ax.set_ylim([-1.1, 1.1])
        ax.set_xlabel(x_label)
        ax.set_ylabel(y_label)
        ax.set_aspect('equal', adjustable='box')
        ax.grid(True, linestyle='-', linewidth=0.5, alpha=0.5)
        ax.quiver(0, 0, vector[0], vector[1], angles='xy',
scale_units='xy', scale=1, color='k')

#Animacion
ratio_frames = 10  # Adjust as needed
animation_duration_s = 10
frames_total = animation_duration_s * ratio_frames
interval_frame = 1000 / ratio_frames

animation = FuncAnimation(fig, update, frames=frames_total,
interval=interval_frame)

plt.show()
```

Bloch Sphere Dynamics Animation - A01197705



F) The commutator between two operators A and B is defined as

$$[A, B] = AB - BA$$

Verify the following relations

$$[X_+, X_-] = Z$$

$$[Z, X_\pm] = \pm 2X_\pm$$

Ver procedimientos en siguiente página

f) The commutator between two operators A and B is defined as

$[A, B] = AB - BA$

Verify the following relations:

$$[X_+, X_-] = Z$$

$$[Z, X_\pm] = \pm 2X_\pm$$

Considerando que:

$$X_+ = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \quad X_- = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Primer caso

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

$0 \cdot 0 + 1 \cdot 1 = 1$

$0 \cdot 0 + 1 \cdot 0 = 0$

$0 \cdot 0 + 0 \cdot 1 = 0$

$0 \cdot 0 + 0 \cdot 0 = 0$

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

$0 \cdot 0 + 0 \cdot 0 = 0$

$0 \cdot 1 + 0 \cdot 0 = 0$

$1 \cdot 0 + 0 \cdot 0 = 0$

$1 \cdot 1 + 0 \cdot 0 = 1$

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

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

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

Por lo que

$$[X_+, X_-] = Z$$

Caso dos:

$$[Z, X_\pm] = \pm 2X_\pm$$

Considerando que:

$$X_+ = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \quad X_- = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad \text{and} \quad [X_+, X_-] = Z$$

Se evaluara Z para ambos casos con X_+ + X_-

$$[A,B] = AB - BA$$

$$[Z, X_+] = +2X_+$$

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} - \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = +2X_+$$

$1\cdot0 + 0\cdot0 = 0$
$1\cdot1 + 0\cdot0 = 1$
$0\cdot0 + -1\cdot0 = 0$
$0\cdot1 + -1\cdot0 = 0$

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

$0\cdot1 + 1\cdot0 = 0$
$0\cdot0 + 1\cdot-1 = -1$
$0\cdot1 + 0\cdot0 = 0$
$0\cdot0 + 0\cdot-1 = 0$

$$\begin{pmatrix} 0 & 2 \\ 0 & 0 \end{pmatrix} = +2X_+$$

Si sacamos el 2 de la matriz veremos que

$$\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} = X_+ \quad = \quad \begin{pmatrix} 0 & 2 \\ 0 & 0 \end{pmatrix} = +2X_+$$

Para evaluar a $[Z, X_-] = -2X_-$

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = -2X_-$$

$1\cdot0 + 0\cdot1 = 0$
$1\cdot0 + 0\cdot0 = 0$
$0\cdot0 + -1\cdot1 = -1$
$0\cdot0 + -1\cdot0 = 0$

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

$0\cdot1 + 0\cdot0 = 0$
$0\cdot0 + 0\cdot-1 = 0$
$1\cdot1 + 0\cdot0 = 1$
$1\cdot0 + 0\cdot-1 = 0$

$$\begin{pmatrix} 0 & 0 \\ -2 & 0 \end{pmatrix} = -2X_-$$

Si sacamos el -2 de la matriz tenemos que

$$\begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} = X_- \longrightarrow \begin{pmatrix} 0 & 0 \\ -2 & 0 \end{pmatrix} = -2X_-$$

Y simplificado podemos decir que

$$[Z, X_\pm] = \pm 2X_\pm$$