MRI
Task #1

Abdulla Ahmed Ali Zahran	Sec: 2	Bench: 5
Samar Ibrahim	Sec: 1	Bench: 40
Shymaa Gamal	Sec: 2	Bench: 1
Sondos Mohamed	Sec: 1	Bench: 42

Contents

Bloch Equations 1
Image Components 4

Bloch Equations

This is a representation for Bloch Equation in 3-D space using Python IDE

We have generated the following images below:

- M vector with angular frequency 30deg
- Excitation Mode with angular frequency 30deg
- Relaxation Mode with angular frequency 30deg
- Excitation - Relaxation Mode
- Excitation - Relaxation Trajectory on X-Y plane
- Relaxation - Excitation Mode
- Relaxation - Excitation Trajectory on X-Y plane

Main Equations for M vector:

- $X = sin(z) \times cos(t)$
- $Y = sin(z) \times sin(t)$
- $Z = \cos(z)$

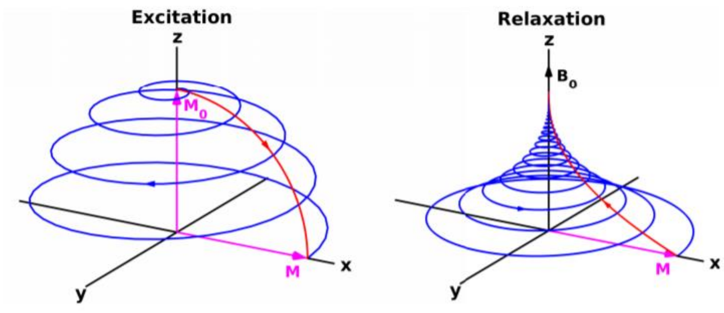
where t represents the angle of rotation, z is the angular frequency, by changing z value with respect to t time we have the following representations:

Excitation Model

- $z = \frac{pi}{2} \times \sin\left(\frac{t}{4}\right)$

Relaxation Model

- $z = \frac{pi}{2} \times cos\left(\frac{t}{4}\right)$

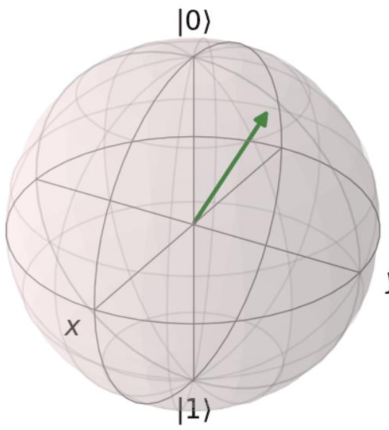


Also, you can set numberOfLoops = 1 to generate single mode [Excitation or Relaxation] only or = 2 to generate dual modes.

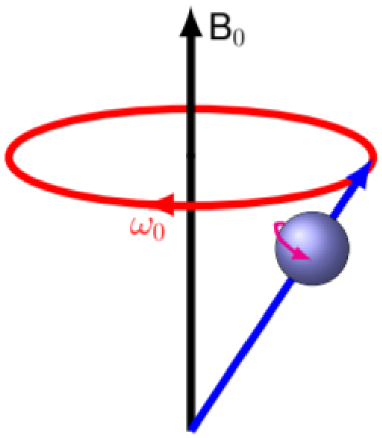
Python Code:

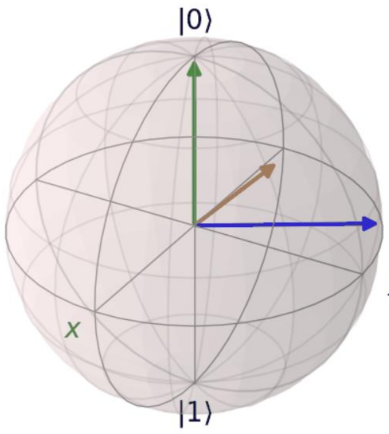
```python
import numpy as np
import imageio
from qutip import Bloch
def animate_bloch(vectors, duration=0.1, save_all=False):
    numberOfLoops = 1
    if numberOfLoops == 1:
        maxAngle = 2*np.pi
    elif numberOfLoops == 2:
        maxAngle = 4*np.pi
    mode = 1
    omega = np.pi/6
    z = 0
    sqAngle = np.pi/2
    a = 5
    vectorM = Bloch()
    images=[]
    for t in np.arange(omega, maxAngle, 0.1):
        if mode == 0:
            z = np.pi/2 * np.sin(t/(4))
            if t == 2*np.pi:
                mode = 1
        elif mode == 1:
            z = np.pi/2 * np.cos(t/(4))
            if t == 2*np.pi:
                mode = 0
        else:
            pass
        vectorM.clear()
        vectorM.add_vectors([np.sin(omega)*np.cos(t), np.sin(omega)*np.sin(t
), np.cos(omega)])
        vectorM.add_vectors([np.sin(z)*np.cos(a*t), np.sin(z)*np.sin(a*t), n
p.cos(z)])
        vectorM.add_vectors([np.sin(sqAngle)*np.cos(t), np.sin(sqAngle)*np.s
in(t), np.cos(sqAngle)])
        filename = 'temp_file.png'
        vectorM.save(filename)
        images.append(imageio.imread(filename))
    imageio.mimsave('bloch_anim.gif', images, duration=duration)
vectors = []
animate_bloch(vectors, duration=0.1, save_all=False)
```

Images Generated from The Previous Code:

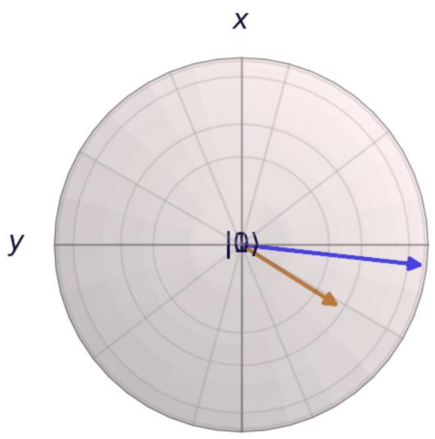


M vector on precession

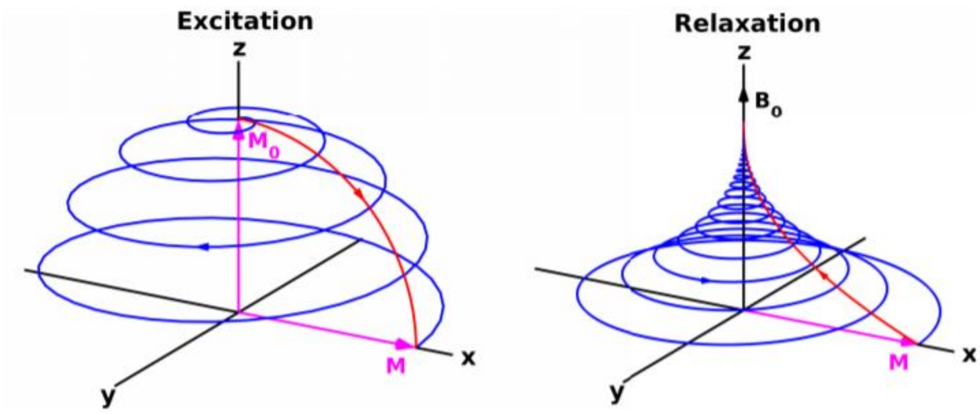




On Excitation/Relaxation



X-Y Trajectory

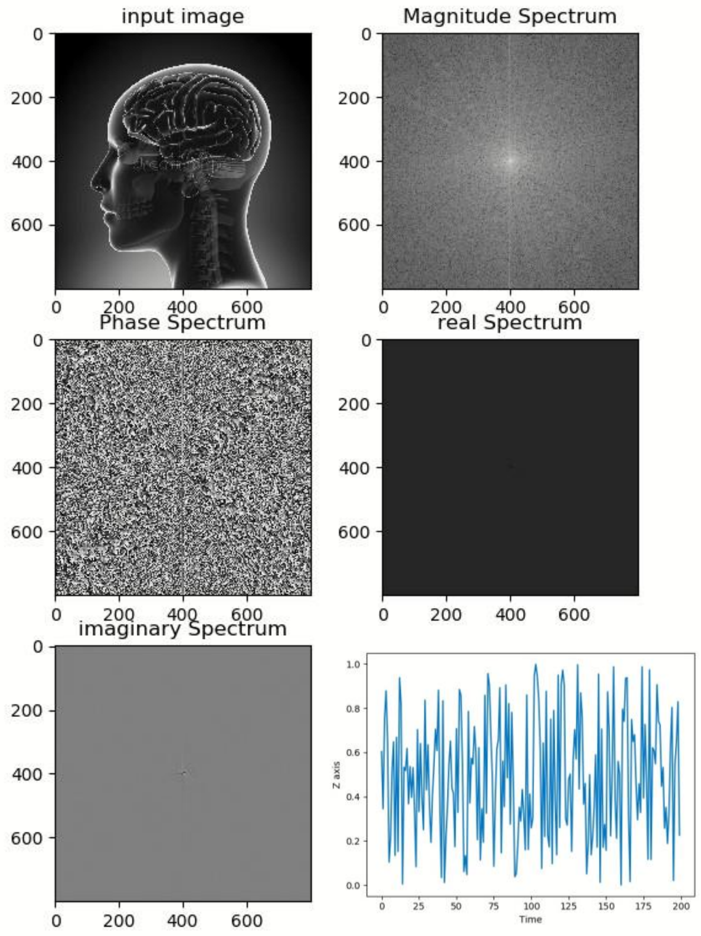


Attached with the report file GIFs animations for the previous images.

- Green: M vector on Relaxation with 0/30deg angular frequency.
- Orange: M vector on Excitation/Relaxation with B.
- Blue: M vector on Excitation with 90deg with Z vector.

Image Components

Here we use magnitude, angle, real and imaginary to show each Fourier component in Matplot show.



Python Code:

```python
import numpy as np
from cv2 import cv2 as cv
import matplotlib.pyplot as p
from numpy import random
def main ():
    img=cv.imread('1.jpg',0)
    spectrum =np.fft.fftshift(np.fft.fft2(img))
    p.subplot(321)
    p.imshow(img, cmap='gray')
    p.title('input image')
    p.subplot(322)
    p.imshow( np.log(np.abs(spectrum)) , cmap='gray')
    p.title('Magnitude Spectrum')
    p.subplot(323)
    p.imshow(np.angle(spectrum),cmap='gray')
    p.title('Phase Spectrum')
    p.subplot(324)
    p.imshow(np.real(spectrum),cmap='gray')
    p.title('real Spectrum')
    p.subplot(325)
    p.imshow(  np.imag(spectrum),cmap='gray')
    p.title('imaginary Spectrum')
    p.show()
    z = random.random(200)
    p.ylabel("Z axis")
    p.xlabel("Time")
    p.plot(z)
    p.show()

if __name__=="__main__":
    main()
```

The code, GIFs and files all uploaded on GitHub:

https://github.com/Abdolla25/MRI-Task_1