Derivations inspired from Lukin notes, Ran et al. [1], Perplexity and Bankim's code.

Aditya

November 6, 2024

1 Hamiltonian

1.1 System Definition

- $|1\rangle$: Ground state
- $|2\rangle$: Middle state
- $|3\rangle$: Highest excited state

$$H = \hbar\Delta_1|2\rangle\langle 2| + \hbar(\Delta_1+\Delta_2)|3\rangle\langle 3| + \hbar\Omega_1(|1\rangle\langle 2| + |2\rangle\langle 1|) + \hbar\Omega_2(|2\rangle\langle 3| + |3\rangle\langle 2|) \tag{1}$$

1.2 Lindblad Operators

$$L_1 = \sqrt{\gamma_{21}}|1\rangle\langle 2| \tag{2}$$

$$L_2 = \sqrt{\gamma_{32}}|2\rangle\langle 3| \tag{3}$$

$$L_3 = \sqrt{\gamma_{31}}|1\rangle\langle 3| \tag{4}$$

1.3 Master Equation

$$\frac{d\rho}{dt} = -\frac{i}{\hbar}[H,\rho] + \sum_k \left(L_k \rho L_k^\dagger - \frac{1}{2}\{L_k^\dagger L_k, \rho\} \right) \tag{5}$$

1.4 Bloch Equations

The complete set of Bloch equations for the density matrix elements:

$$\frac{d\rho_{11}}{dt} = \gamma_{21}\rho_{22} + \gamma_{31}\rho_{33} + i\Omega_1(\rho_{21} - \rho_{12}) \tag{6}$$

$$\frac{d\rho_{22}}{dt} = -(\gamma_{21} + \gamma_{32})\rho_{22} + i\Omega_1(\rho_{12} - \rho_{21}) + i\Omega_2(\rho_{32} - \rho_{23}) \tag{7}$$

$$\frac{d\rho_{33}}{dt} = -(\gamma_{31} + \gamma_{32})\rho_{33} + i\Omega_2(\rho_{23} - \rho_{32}) \tag{8}$$

$$\frac{d\rho_{21}}{dt} = -(\frac{\gamma_{21}}{2} - i\Delta_1)\rho_{21} - i\Omega_1(\rho_{22} - \rho_{11}) - i\Omega_2\rho_{31} \tag{9}$$

$$\frac{d\rho_{32}}{dt} = -(\frac{\gamma_{32}}{2} - i\Delta_2)\rho_{32} - i\Omega_2(\rho_{33} - \rho_{22}) + i\Omega_1\rho_{31} \tag{10}$$

$$\frac{d\rho_{31}}{dt} = -(\frac{\gamma_{31}}{2} - i(\Delta_1 + \Delta_2))\rho_{31} + i\Omega_1\rho_{32} - i\Omega_2\rho_{21} \tag{11}$$

The remaining equations can be obtained using

$$\rho_{ij} = \rho_{ji}^*$$

and

$$\rho_{11} + \rho_{22} + \rho_{33} = 1$$

1.5 Assumptions

We make the following assumptions:

1. All population is in the ground state: $\rho_{11} \approx 1$
2. Excited states are empty: $\rho_{22} = \rho_{33} \approx 0$
3. No coherence between empty states: $\rho_{23} = \rho_{32} \approx 0$

1.6 Steady-State Equations

Our goal is to calculate coherence ρ_{21}, which also requires ρ_{31}. So for that we apply the assumptions mentioned above and solve the relevant steady-state equations Eq. 9 and Eq. 11 become:

$$0 = -(\frac{\gamma_{21}}{2} - i\Delta_1)\rho_{21} + i\Omega_1 - i\Omega_2\rho_{31} \tag{12}$$

$$0 = -(\frac{\gamma_{31}}{2} - i(\Delta_1 + \Delta_2))\rho_{31} + i\Omega_2\rho_{21} \tag{13}$$

1.7 Solution

From the Eq. 12, we can express ρ_{31} in terms of ρ_{21}:

$$\rho_{31} = \frac{i\Omega_2\rho_{21}}{\frac{\gamma_{31}}{2} - i(\Delta_1 + \Delta_2)} \tag{14}$$

Substituting this into the Eq. 13:

$$0 = -(\frac{\gamma_{21}}{2} - i\Delta_1)\rho_{21} + i\Omega_1 - i\Omega_2 \frac{i\Omega_2\rho_{21}}{\frac{\gamma_{31}}{2} - i(\Delta_1 + \Delta_2)} \tag{15}$$

Solving for ρ_{21}, we get:

$$\rho_{21} = \frac{i\Omega_1(\frac{\gamma_{31}}{2} - i(\Delta_1 + \Delta_2))}{(\frac{\gamma_{21}}{2} - i\Delta_1)(\frac{\gamma_{31}}{2} - i(\Delta_1 + \Delta_2)) + \Omega_2^2} \tag{16}$$

This is the steady-state solution for ρ_{21} under the given assumptions, taking into account all relevant couplings and decay processes in the three-level system.

Hence we get

$$\rho_{21} = \frac{i\Omega_1}{\left(\frac{\gamma_{21}}{2} - i\Delta_1\right) + \frac{\Omega_2^2}{\frac{\gamma_{31}}{2} - i(\Delta_1 + \Delta_2)}} \tag{17}$$

2 Important points to remember while numerical computation using qutip

- Qutip does not know about the weak probe approximation, hence for analytical and numerical simulations to match, the probe value (g_1) should be set low.
- The code is written with different states labelling, so notice the change in density matrix labelling as shown in the below :

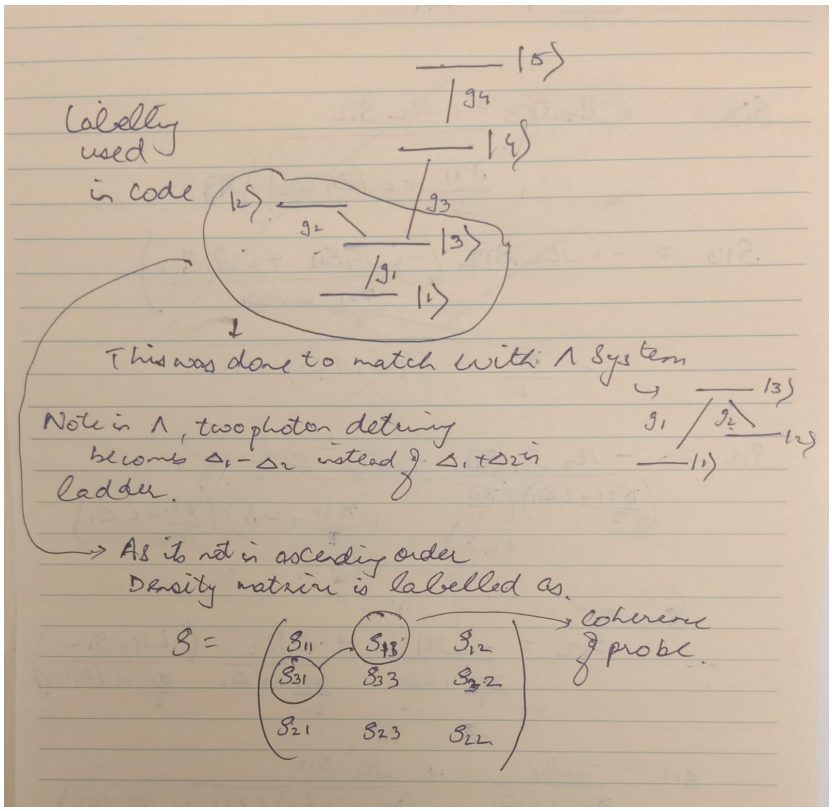

Figure 1: new labelling

- Some dispute between analytical and numerical expression because of decay as shown in this figure

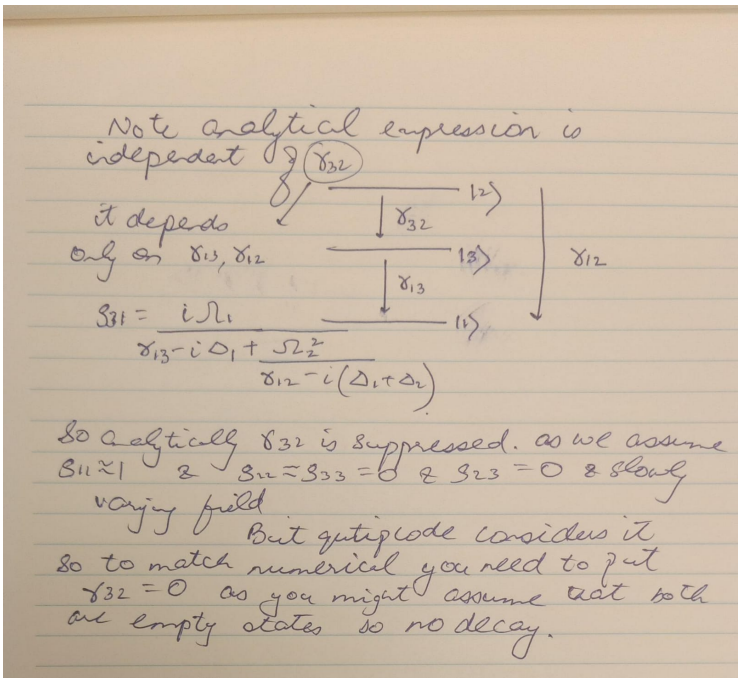

Figure 2: dispute between numerical and analytical

- In Bankim's code, or in our group we sometimes call $\gamma' = \frac{\gamma}{2}$.

3 Analytical and qutip code

```
import numpy as np
import qutip as qt
import matplotlib.pyplot as plt
from ipywidgets import interact, FloatSlider, Layout
from IPython.display import display

# Fixed parameter
g_probe = 0.000000001  # Weak probe coupling strength for
    ↪ transition 1-3

def analytical_coherence(delta_1, g_2, gamma_31, gamma_21, delta_2,
    ↪  g_3, gamma_41, delta_3, gamma_51, delta_4, g_4):
    nr = 1j * g_probe
    dr = (gamma_31/2 - 1j*delta_1) + g_2**2 / ((gamma_21/2 - 1j*(
        ↪ delta_2 + delta_1))) + g_3**2 / ((gamma_41/2 - 1j*(
        ↪ delta_3 + delta_1)) + g_4**2 / (gamma_51/2 - 1j*(delta_4
        ↪  + delta_3 + delta_1)))
    return nr / dr
```

```
def numerical_coherence(delta_1, g_2, gamma_31, gamma_23, gamma_21,
    ↪  delta_2, g_3, gamma_43, gamma_41, delta_3, gamma_54,
    ↪ gamma_51, delta_4, g_4):
    # Define basis states and operators
    ground_state = qt.basis(5, 0)  # Ground state |1
    excited_1 = qt.basis(5, 2)  # Highly excited state |2
    excited_2 = qt.basis(5, 1)  # Middle state |3
    excited_3 = qt.basis(5, 3)  # |4
    excited_4 = qt.basis(5, 4)  # |5

    a_13 = ground_state * excited_2.dag()  # Lowering operator for
        ↪ 1-3 transition
    a_32 = excited_2 * excited_1.dag()  # Lowering operator for 3-2
        ↪  transition
    a_12 = ground_state * excited_1.dag()  # Lowering operator for
        ↪ 2-1 transition
    a_34 = excited_2 * excited_3.dag()  # Lowering operator for 4-3
        ↪  transition
    a_14 = ground_state * excited_3.dag()  # Lowering operator for
        ↪ 4-1 transition
    a_15 = ground_state * excited_4.dag()  # Lowering operator for
        ↪ 5-1 transition
    a_45 = excited_3 * excited_4.dag()  # Lowering operator for 5-4
        ↪  transition

    H = (delta_1 + delta_3 + delta_4) * excited_4 * excited_4.dag()
        ↪  + (delta_1 + delta_3) * excited_3 * excited_3.dag() + \
        delta_1 * excited_2 * excited_2.dag() + (delta_1 + delta_2)
            ↪  * excited_1 * excited_1.dag() + \
        g_probe * (a_13 + a_13.dag()) + g_2 * (a_32 + a_32.dag()) +
            ↪  g_3 * (a_34 + a_34.dag()) + g_4 * (a_45 + a_45.dag
            ↪ ())

    c_ops = [np.sqrt(gamma_31) * a_13, np.sqrt(gamma_23) * a_32, np
        ↪ .sqrt(gamma_21) * a_12,
             np.sqrt(gamma_43) * a_34, np.sqrt(gamma_41) * a_14, np
                 ↪ .sqrt(gamma_54) * a_45, np.sqrt(gamma_51) *
                 ↪ a_15]

    rho_ss = qt.steadystate(H, c_ops)
    return rho_ss[0, 1]

def plot_coherences(g_2, gamma_31, gamma_23, gamma_21, delta_2, g_3
    ↪ , gamma_41, delta_3, gamma_43, gamma_54, gamma_51, delta_4,
    ↪ g_4, optical_depth):
    # System parameters
    delta_1_range = np.linspace(-30, 30, 80)

    # Calculate coherences
    rho_13_analytical = [analytical_coherence(delta_1, g_2,
        ↪ gamma_31, gamma_21, delta_2, g_3, gamma_41, delta_3,
        ↪ gamma_51, delta_4, g_4) for delta_1 in delta_1_range]
    rho_13_numerical = [numerical_coherence(delta_1, g_2, gamma_31,
        ↪  gamma_23, gamma_21, delta_2, g_3, gamma_43, gamma_41,
        ↪ delta_3, gamma_54, gamma_51, delta_4, g_4) for delta_1
        ↪ in delta_1_range]
```

```
    chi_analytical = optical_depth * np.array(rho_13_analytical) /
        ↪ g_probe
    chi_numerical = optical_depth * np.array(rho_13_numerical) /
        ↪ g_probe

    # Create figure with three subplots
    fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize=(18, 6))
    fig.text(0.5, 0.95, f'g_probe (fixed) = {g_probe:.9f}',
        ↪ horizontalalignment='center', fontsize=12)

    # Plot 1: Coherences
    ax1.plot(delta_1_range, np.abs(rho_13_analytical), 'b-', label=
        ↪ '|rho_13| Analytical')
    ax1.plot(delta_1_range, np.abs(rho_13_numerical), 'r--', label=
        ↪ '|rho_13| Numerical')
    ax1.set_xlabel('Probe Detuning (delta_1)')
    ax1.set_ylabel('Coherence Magnitude')
    ax1.set_title('Coherences vs Probe Detuning')
    ax1.legend()
    ax1.grid(True)

    # Plot 2: Transmission
    ax2.plot(delta_1_range, np.exp(-np.imag(chi_analytical)), 'b-',
        ↪  label='Analytical Transmission')
    ax2.plot(delta_1_range, np.exp(-np.imag(chi_numerical)), 'r--',
        ↪  label='Numerical Transmission')
    ax2.set_xlabel('Probe Detuning (delta_1)')
    ax2.set_ylabel('Transmission')
    ax2.legend()
    ax2.grid(True)

    # Plot 3: Phase
    ax3.plot(delta_1_range, np.real(chi_analytical), 'b-', label='
        ↪ Analytical Phase')
    ax3.plot(delta_1_range, np.real(chi_numerical), 'r--', label='
        ↪ Numerical Phase')
    ax3.set_xlabel('Probe Detuning (delta_1)')
    ax3.set_ylabel('Phase')
    ax3.legend()
    ax3.set_title('Phase')
    ax3.grid(True)

    plt.tight_layout()
    plt.show()

    # Print system parameters
    print("System Parameters:")
    print(f"g_2 = {g_2}")
    print(f"gamma_31 = {gamma_31}")
    print(f"gamma_23 = {gamma_23}")
    print(f"gamma_21 = {gamma_21}")
    print(f"delta_2 = {delta_2}")
    print(f"Optical Depth = {optical_depth}")

# Create sliders
slider_layout = Layout(width='500px')
```

```
g_2_slider = FloatSlider(value=1, min=0, max=10, step=0.1,
    ↪ description='g_2:', layout=slider_layout)
gamma_31_slider = FloatSlider(value=0.1, min=0, max=1, step=0.05,
    ↪ description='gamma_31:', layout=slider_layout)
gamma_23_slider = FloatSlider(value=0, min=0, max=1, step=0.05,
    ↪ description='gamma_23:', layout=slider_layout)
gamma_21_slider = FloatSlider(value=0.1, min=0, max=1, step=0.05,
    ↪ description='gamma_21:', layout=slider_layout)
delta_2_slider = FloatSlider(value=0, min=-10, max=10, step=0.5,
    ↪ description='delta_2:', layout=slider_layout)
g_3_slider = FloatSlider(value=1, min=0, max=2, step=0.1,
    ↪ description='g_3:', layout=slider_layout)
gamma_41_slider = FloatSlider(value=0.5, min=0, max=1, step=0.05,
    ↪ description='gamma_41:', layout=slider_layout)
gamma_43_slider = FloatSlider(value=0, min=0, max=1, step=0.05,
    ↪ description='gamma_43:', layout=slider_layout)
delta_3_slider = FloatSlider(value=0, min=-10, max=10, step=0.5,
    ↪ description='delta_3:', layout=slider_layout)
OD_slider = FloatSlider(value=20, min=0, max=150, step=1,
    ↪ description='Optical Depth:', layout=slider_layout)
gamma_51_slider = FloatSlider(value=0.5, min=0, max=1, step=0.05,
    ↪ description='gamma_51:', layout=slider_layout)
gamma_54_slider = FloatSlider(value=0.5, min=0, max=1, step=0.05,
    ↪ description='gamma_54:', layout=slider_layout)
delta_4_slider = FloatSlider(value=0, min=-10, max=10, step=0.5,
    ↪ description='delta_4:', layout=slider_layout)
g_4_slider = FloatSlider(value=1, min=0, max=2, step=0.1,
    ↪ description='g_4:', layout=slider_layout)

interact(plot_coherences, g_2=g_2_slider, gamma_31=gamma_31_slider,
    ↪  gamma_23=gamma_23_slider,
         gamma_21=gamma_21_slider, delta_2=delta_2_slider, g_3=
             ↪ g_3_slider, gamma_41=gamma_41_slider,
         delta_3=delta_3_slider, gamma_43=gamma_43_slider, gamma_54
             ↪ =gamma_54_slider,
         gamma_51=gamma_51_slider, delta_4=delta_4_slider, g_4=
             ↪ g_4_slider, optical_depth=OD_slider)
```

References

[1] Ran Finkelstein and Ofer Firstenberg. "A practical guide to electromagnetically induced transparency in atomic vapor". In: *New Journal of Physics* 25.6 (2023), p. 063016. DOI: 10.1088/1367-2630/acbc40.