Stochastic Simulation of Langevin Dynamics in Optical Systems

Nathan Ngo and Supervisors: Barry Sanders & David Feder
University of Calgary
(Dated: August 2024)

I. INTRODUCTION

This document describes a simulation of a driven, underdamped harmonic oscillator with two distinct types of noise:

1. **Random phase shift noise** in the driving force.
2. **Random Gaussian noise** affecting the system's damping.

The dynamics of the system are governed by the following equation:

$$\ddot{\zeta} + 2r\dot{\zeta} + \omega_m^2\zeta + \frac{\xi(t)}{m} = -\frac{F(t)}{m}$$

where:

- ζ (denoted as x) is the **position**, restricted to oscillation along the x-axis.
- r is the **damping coefficient**.
- ω_m is the **natural angular frequency**.
- $\xi(t)$ is the **random Gaussian noise** affecting the system's damping.
- $F(t)$ is the **external driving force**, which includes a **random phase shift noise** component.

II. FEATURES

- **Phase Shift Noise**: Affects the phase of the driving force $F(t)$, introducing random shifts after a few cycles.
- **Gaussian Noise**: Affects the damping term $\xi(t)$, adding random perturbations to the system's dynamics.
- **RK45 Integration**: The system is solved using the *Runge-Kutta 45* method with adaptive step sizing and error control.
- **Exponential Envelope Fitting**: The code fits an exponential decay envelope to the oscillator's motion and calculates the **relaxation time** T_1.
- **Visualization**: The simulation generates position-time plots, Fourier Transform graphs, and phase space diagrams.

III. GOVERNING EQUATION

The system follows this equation of motion:

$$\ddot{\zeta} + 2r\dot{\zeta} + \omega_m^2\zeta + \frac{\xi(t)}{m} = -\frac{F(t)}{m}$$

- $\zeta = x$: Position along the x-axis.
- r: Damping coefficient.
- ω_m: Natural angular frequency.
- $\xi(t)$: Gaussian noise on the damping term.
- $F(t)$: External driving force with random phase shift noise.

IV. WORKFLOW

1. The `RK45` solver computes the system's position and velocity over time.
2. The results are *interpolated* for smooth plotting.
3. A *Fourier Transform* is performed on the oscillator's position data to analyze frequency components.
4. The *phase space diagram* (position vs. momentum) is plotted to visualize the system's trajectory.
5. The *exponential envelope* of the position data is fitted to calculate the **relaxation time** T_1.

V. RELAXATION TIME CALCULATION

The exponential decay of the oscillator's motion is fitted using the envelope function:

$$Ae^{-\gamma t}$$

where γ is the damping coefficient, and the **relaxation time** T_1 is calculated as:

$$T_1 = \frac{1}{\gamma}$$

The relaxation time is printed as output:

```
Relaxation time T1: X.XXXX seconds
```

VI. HOW TO RUN THE CODE

A. Prerequisites

Ensure that you have the following Python packages installed:

```
pip install numpy scipy matplotlib
```

B. Running the Code

To run the simulation, execute the main script. You will be prompted to enter a **detuning factor** for the driving frequency. For example, to detune by half a linewidth, enter:

```
Enter the detuning factor (e.g., 0.5 for half
a linewidth): 0.5
```

After running, the code will generate the specified plots and save them in the working directory.

C. Output

- **Position-Time Plot**: Shows the position of the oscillator over time with and without noise.
- **Fourier Transform Plot**: Displays the frequency analysis of the oscillator's motion.
- **Phase Space Diagram**: Visualizes the system's trajectory in phase space.
- **Exponential Envelope Fitting**: Displays the fitted exponential envelope and calculates the relaxation time T_1.