Multiple Parameter Estimation of Couple Oscillator Model

409220055 楊長茂、409235037 蘇唯善、411220006 陳奕安

Abstract:

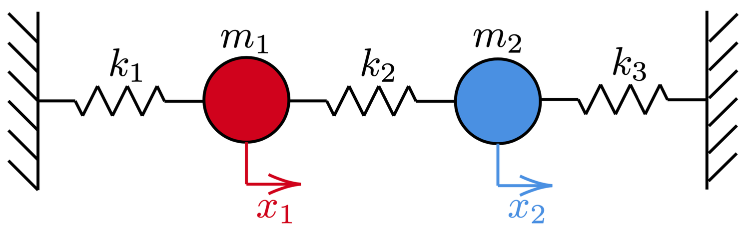
In this experiment, we utilized the Lagrangian method to calculate the exact solution of a Coupled Oscillators, serving as an analytical model. Given the known weights of two objects, we employed MCMC to simulate the model and experimental data, thereby obtaining the spring constants of the three springs. To measure the positions of the two objects during the experiment, we designed an Arduino circuit for measurement and data transmission. The data was accessed and retrieved using a custom-designed webpage, and subsequently analyzed and subjected to MCMC simulations using Python.

Introduction:

The Couple Oscillator is an important model in Physics and Chemistry that describes the vibrational motion of molecules. Using classical dynamics, we can analytically calculate the solutions to the equations of motion for the couple oscillator. To validate these solutions, we have developed an instrument that demonstrates the behavior of the coupled oscillator. By measuring the displacement of two objects, we can graph the results of the coupled oscillator. Additionally, we can fit the analytical solution of the coupled oscillator to determine the spring constants in our instrument.

Theorem:

The coupled oscillator can be interpreted as following



The Lagrangian of this system is given by following

$$L = K - V,$$

where the kinetic energy and potential energy are

$$K = \frac{1}{2} m_1 \frac{d^2 x_1}{dt^2} \frac{1}{2} m_2 \frac{d^2 x_2}{dt^2},$$

$$V = \frac{1}{2} k_1 x_1^2 + \frac{1}{2} k_2 (x_2 - x_2)^2 + \frac{1}{2} k_3 x_2^2.$$

Solving to Euler-Lagrange equation we obtain the second order differential equations

$$\begin{pmatrix} \ddot{x}_1 \\ \ddot{x}_2 \end{pmatrix} = \begin{pmatrix} -\dfrac{k_1 + k_2}{m_1} & \dfrac{k_2}{m_1} \\ \dfrac{k_2}{m_2} & -\dfrac{k_2 + k_3}{m_2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}.$$

Define the 2 by 2 matrix as F, we can solve this system this system analytically. The eigenvalues of matrix is

$$\lambda = \frac{\mathrm{tr}(F) \pm \sqrt{\mathrm{tr}(F)^2 - 4\det(F)}}{2},$$

where $\mathrm{tr}(F)$ and $\det(F)$ are the trace and determinant of matrix respectively, and eigenvectors which are also the solution of this system are given by

$$\mu = \begin{pmatrix} \dfrac{k_2}{m_1} \\ \lambda + \dfrac{k_1 + k_2}{m_1} \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} \lambda + \dfrac{k_2 + k_3}{m_2} \\ \dfrac{k_2}{m_2} \end{pmatrix}.$$

The general solution are the linear expansion of above eigenvectors, which is

$$\psi(t) = \sum_{j=0}^{2} C_j e^{i\omega_j t} \mu_j,$$

where $\omega_j = \sqrt{|\lambda|}$ are the corresponding frequencies, and the coefficients C_j can be determine by the initial value. Therefore, given a initial position of this system, we have the position of two masses with respect to time.

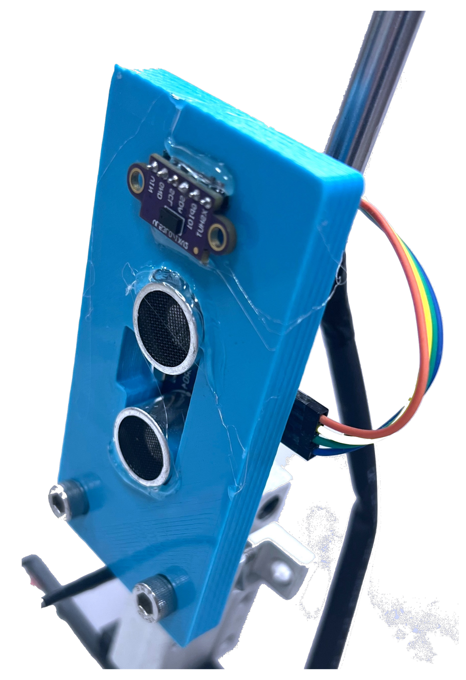
Experimental Equipment:

Our experimental setup is shown in the diagram below. It includes a typical physics experiment's pulley track and additionally utilizes springs with varying spring constants made from white iron wire.

We have created 3D-printed frames for ultrasonic and laser distance measuring devices on the far left and far right. These devices are connected to a central circuit board via wires and utilize Arduino to read the values. The readings are then transmitted to a connected computer. Additionally, we have developed a dedicated web page in our GitHub repository for accessing the measured values.

Error Analysis:

Our Arduino distance measuring device is shown in the diagram on the right. The upper part consists of a laser distance measuring component, while the other part is an ultrasonic distance measuring component. Although the laser distance measuring device offers higher precision, it takes longer to measure distances, resulting in a lower sampling frequency compared to the system's frequency. This can lead to aliasing issues. Therefore, we ultimately chose the ultrasonic distance measuring device as the instrument for distance measurement.



Data Analysis:

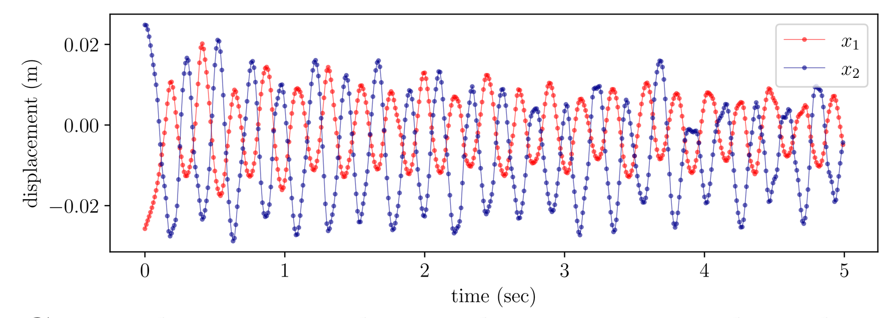
After capturing data from Arduino, we analyze it using Python. We apply the Savitzky-Golay filter and cubic spline interpolation to smooth the curve and obtain a splined curve. Additionally, we fit the curve with the analytical solution to evaluate the spring constants k_1, k_2, k_3

Fitting multiple parameters using traditional methods like non-linear least squares can be challenging. To address this, we employ Markov chain Monte Carlo (MCMC) simulation. We initialize the spring constants and determine a suitable step size. Next, we generate new parameters by random sampling within the step size interval. We calculate the Chi-square value by comparing the experimental data with both the old and new parameter models, which is

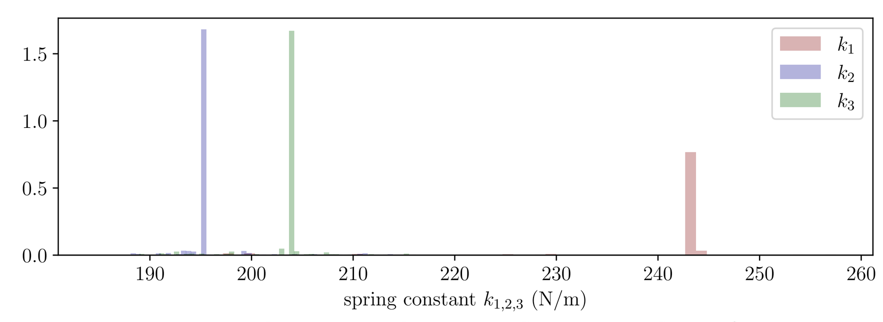
$$\chi_i^2 = \sum_{j=1}^{n} \frac{(x_{i,\text{data},j} - x_{i,\text{theory},j})^2}{x_{i,\text{theory},j}}, \quad i = 1, 2$$

If the Chi-square value for the new parameters is lower than that of the old parameters, we update the old parameters for the next iteration. If the Chi-square value is higher, we may still update the old parameters to the new ones, but this is determined by a probability ratio based on the Chi-square values.

This algorithm is known as Metropolis-Hastings. By repeating this process, we obtain the spring constants. For example, we start from a data



After using MCMC simulation, we have the posterior distribution as following



Searching for the maximum value we have the result of spring constant

$$k_1 \approx 203.1, \quad k_2 \approx 191.7, \quad k_3 \approx 204.8, \quad (\mathrm{N/m})$$

Comparing this result with the spring constants that we have actually measured, we find that the error is approximately 9.68%.

In this experiment, we given many kind of initial value problem, and the result are given below

#	k_1 (N/m)	k_2 (N/m)	k_3 (N/m)	error
1	199.19	200.73	172.77	19.84%
2	210.26	192.87	172.77	15.52%
3	194.60	211.44	172.77	5.37%
4	242.69	195.04	203.69	22.71%
5	202.21	195.98	231.24	13.75%
mean	209.79	199.21	190.64	15.44%

Conclusion:

Indeed, we can utilize MCMC to predict multiple parameters for this experiment. The simulation results closely approximate the actual results with small errors. However, due to the similar values of the spring constants for the three springs, it is challenging to determine their relative magnitudes accurately. In the MCMC simulation results, we observe a distribution with a higher density in the posterior distribution plot, indicating the effectiveness of this method.

Reference:

[1] Tong, D. (2019). Classical Dynamics: Lecture Notes. From: http://www.damtp.cam.ac.uk/user/tong/dynamics/two.pdf
[2] Gareth O. Roberts, Jeffrey S. Rosenthal (2004). General state space Markov chains and MCMC algorithms, DOI: 10.1214/154957804100000024
[3] Computational Physics, M. Newman (2013)