Parameter Estimation Techniques for Forced Systems Governed by Second Order Linear Ordinary Differential Equations

Cormac Molyneaux

August 2025

1 Abstract

This report evaluates various methods for estimating the parameters in a second-order linear ordinary differential equation (ODE) from noisy and discrete measurements of a system's input, $f(t)$ and output, $x(t)$. In this report, the problem is broken into two parts: first, the data is pre-processed and the derivative terms are estimated using three techniques, the Butterworth low-pass filter, univariate spline fitting and a Gaussian process. Estimating these derivative terms without amplifying the noise in the measurements is a key challenge. Second, the system's parameters are estimated using linear regression. Ordinary least squares (OLS) and orthogonal regression (ODR) are both evaluated here. The performance of each combination of methods is assessed using Monte Carlo simulations. The Gaussian process provided the most accurate derivative estimations out of the three techniques, leading to the best parameter estimation when paired with OLS. Overall, OLS outperformed ODR when the derivative terms were estimated very well or very badly, while ODR provided less bias and variance in the parameter estimates for close but imperfect derivative estimates, such as those generated by the low-pass filter method. These findings highlight using a Gaussian process and OLS as a robust method for estimating the parameters in a second-order linear ODE and provide a robust and evidence-based workflow for model selection in systems identification.

2 Introduction

Systems identification is a broad and mature field. At its core, systems identification deals with making mathematical models of a system based on observations (Ljung 1998). This is a fundamental aspect of all sciences and allows for a richer understanding of the world around us. For physical systems, these models encompass the underlying physical laws governing a system's movement.

This allows a system's output to be simulated for any input or for the required input to achieve the desired output to be calculated.

This report focuses on systems with governing equations that can be written as second-order linear ordinary differential equations. This is an important subset of systems, describing mass-spring-damper mechanisms and RLC circuits. It also serves as a good approximation for many more systems. For these systems, equation 1 can fully describe the relationship between the input, f(t), and the output, x(t), with three parameters, m, c and k.

$$m\ddot{x}(t) + c\dot{x}(t) + kx(t) = f(t) \tag{1}$$

In the context of mass-spring-damper systems, $x(t)$ is the displacement of a mass and $f(t)$ is the external force acting on it. k represents the spring constant or the reaction force per unit length, c is the damping coefficient or the reaction force per unit velocity and finally, m is the mass. The presence of the mass term is a result of Newton's second law, F = ma (Meirovitch 2011).

These parameters can be uniquely determined for a given $f(t)$ and $x(t)$. If this equation form correctly describes the system, with one input and one output, the output for any other input and initial conditions can be determined. This report aims to highlight methods that can be used to predict these three parameters from noisy and discrete samples of a particular $f(t)$ and the resulting $x(t)$.

Typically, system identification focuses on generating models that are fit for the particular purpose or application (Ljung 1998) rather than formally understanding the accuracy of the model. This report aims to generate estimates for m, c and k with more statistical rigour. Simulated data and Monte-Carlo simulations will be used to assess the predictive performance of the proposed methods and quantify the uncertainty in these predictions.

3 Literature Review

Estimating m, c and k from noisy x and f measurements can be broken down into two tasks: estimating $\dot{x}(t)$ and $\ddot{x}(t)$ and performing regression to estimate the parameters. This section breaks down the theory behind the methods that will be used to attempt these tasks.

3.1 Context: Challenges in Data-Driven Dynamics from Noisy Observations

Modern data-driven equation discovery techniques such as SINDy (Brunton, Proctor & Kutz 2016) can uncover the governing equations behind complex dynamic systems with adequate data. However, for systems governed by differential equations, these methods require accurate measurements of the derivatives. In practice, accurately measuring these terms can be very challenging. It would therefore be advantageous to estimate these terms where possible. This has its

own challenges, as the base function is measured with noise, which is amplified by differentiation.

3.2 Advanced Approaches for Robust ODE Discovery in Noisy Regimes

This section examines novel methods that aim to estimate state-time derivatives to recover ODE's from data. These methods aim to achieve the same goal as the method outlined later in this report, but with different strategies and priorities.

3.2.1 Derivative-based SINDY (DSINDy) for Robust Equation Discovery (Wentz & Doostan 2023)

This paper seeks to address the difficulty of estimating state-time derivatives in noisy regimes, taking a matrix-based approach. This is done with a two-step procedure, using projection-based state denoising (PSDN) to clean the data before formulating an iterative re-weighting, second-order cone problem (IRW-SOCP). PSDN leverages the assumed basis to denoise the data and provide the hyperparameters needed for IRW-SOCP. The procedure is carefully put together to ensure that the optimisation problem is convex and minimal hyperparameter tuning is required. This model focuses on predictive performance, outperforming methods such as Gaussian processes. However, this method does not provide uncertainty quantification and relies on the assumption that all noise is uncorrelated.

3.3 Derivative Estimation from Noisy Time Series

Given the importance of accurate derivative estimation, this report outlines, employs, and compares several distinct methodologies. In contrast with DSINDy, these methods are based on established signal processing and statistical techniques. This approach allows for an analysis of the strengths and weaknesses of each method.

3.3.1 Butterworth Filter and Numerical Differentiation

The Butterworth filter is a specific low-pass filter; like with any low-pass filter, the aim is to remove frequencies above a cutoff frequency, while preserving lower frequencies. The Butterworth filter does this to achieve a maximally flat pass band (Butterworth 1930).

The derivative terms are then computed using central differences in the interior of x(t) and one-sided differences at the boundaries. This is done according to the standard for computing central, forward and backwards finite difference derivatives (Yew 2011), (Equations 2, 3 and **??**).

$$\dot{x}(t) \ = \frac{x(t+\Delta t) - x(t-\Delta t)}{2\Delta t} \tag{2}$$

$$\dot{x}(t) = \frac{x(t+\Delta t) - x(t)}{\Delta t} \tag{3}$$

$$\dot{x}(t) = \frac{x(t) - x(t-\Delta t)}{\Delta t} \tag{4}$$

This method is relatively straightforward to implement, with only the cutoff frequency and filter order needing to be optimised; however, with sub-optimal choices for these hyperparameters, the smoothed x(t) can lag behind the true values, and the real signal can be removed, not just noise. Numerical differentiation also inherently amplifies any noise not removed by the filter. This contrasts with many approaches that seek to avoid data pre-processing.

3.3.2 Univariate-Spline Fitting

Univariate-spline fitting is a powerful tool for fitting a smooth curve to noisy data and subsequently obtaining derivatives of this curve. This method relies on fitting polynomial functions in a piecewise way, smoothly joined at points called knots. The implementation in this paper uses the `SciPy`.Interpolate package. Here, there are two hyperparameters to tune: the degree of the spline, k, and the smoothing factor, s. The spline, $\hat{x}(t)$, is fitted to the noisy measurement $\tilde{x}$ over n data points to satisfy the optimisation criterion, equation 5, (Virtanen, Gommers, Oliphant, Haberland, Reddy, Cournapeau, Burovski, Peterson, Weckesser, Bright, van der Walt, Brett, Wilson, Millman, Mayorov, Nelson, Jones, Kern, Larson, Carey, Polat, Feng, Moore, VanderPlas, Laxalde, Perktold, Cimrman, Henriksen, Quintero, Harris, Archibald, Ribeiro, Pedregosa, van Mulbregt & SciPy 1.0 Contributors 2020).

$$\min_{\hat{x}} \left(\sum_{i=1}^{n} (\tilde{x}_i - \hat{x}(t_i))^2 + s \int_{t_1}^{t_n} (\hat{x}^{(k+1)}(t))^2 dt \right) \tag{5}$$

The number and placement of knots are handled internally by SciPy's `UnivariateSpline` function. It initially considers each data point as a knot; however, their effective influence is weighted by the smoothing parameter s to satisfy the optimisation criterion (Equation 5). With an optimal choice of hyperparameters, $\hat{x}(t)$ converges to the true displacement $x(t)$. Consequently, its derivatives, $\dot{\hat{x}}(t)$ and $\ddot{\hat{x}}(t)$, converge to $\dot{x}(t)$ and $\ddot{x}(t)$ respectively, albeit at a slower rate.

3.3.3 Gaussian Process Regression for Derivative Estimation

Gaussian Processes (GPs) are a non-parametric, Bayesian method for approximating functions from underlying data. Prior knowledge of the function can be encoded in the kernel; for this report, the kernel consists of a smooth signal function and a white noise function. This flexible framework and the ability to encode prior knowledge of the data make GPs a powerful tool for accurately recovering $x(t)$ from $\tilde{x}$, while also providing uncertainty quantification for this estimate at every time point (Rasmussen & Williams 2006). For this report, the Gaussian process is implemented through the sklearn Gaussian process package,

(Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, Blondel, Prettenhofer, Weiss, Dubourg et al. 2011).

3.4 Parameter Regression Techniques

Once all derivative terms are estimated, the problem is reduced to linear regression. However, it is important to select an appropriate regression method. This is particularly tricky as, after the pre-processing and derivative estimation step, all of the variables involved have error. $\tilde{x}(t)$ and $\tilde{f}(t)$ are assumed to be measured with Gaussian noise. However, no assumptions can be made about the error in the processed or estimated terms.

3.4.1 Ordinary Least Squares (OLS)

Ordinary Least Squares (OLS) is the most prevalent method for linear regression. It aims to minimise the sum of squared differences between the dependent variable and the values predicted by the model. In the context of the second-order ODE, equation 1, $f(t)$ is treated as the dependent variable, and the assumed form for the model is:

$$\hat{m}\hat{\ddot{x}}(t) + \hat{c}\hat{\dot{x}}(t) + \hat{k}\hat{x}(t) = \tilde{f(t)} + \epsilon \tag{6}$$

This relationship can be concisely expressed in matrix form as $Y = \mathbf{X}\hat{\boldsymbol{\beta}} + \epsilon$, where:

- Y is the $n \times 1$ vector of observed input forces at each time point:

$$\tilde{\mathbf{f}} = \begin{pmatrix} \tilde{f}(t_1) \\ \tilde{f}(t_2) \\ \vdots \\ \tilde{f}(t_n) \end{pmatrix}$$

- $\mathbf{X}$ is the $n \times 3$ design matrix, comprising the estimated derivatives and measured displacement at each time point:

$$\mathbf{X} = \begin{pmatrix} \hat{\ddot{x}}(t_1) & \hat{\dot{x}}(t_1) & \hat{x}(t_1) \\ \hat{\ddot{x}}(t_2) & \hat{\dot{x}}(t_2) & \hat{x}(t_2) \\ \vdots & \vdots & \vdots \\ \hat{\ddot{x}}(t_n) & \hat{\dot{x}}(t_n) & \hat{x}(t_n) \end{pmatrix}$$

- $\hat{\boldsymbol{\beta}}$ is the 3×1 vector of unknown parameters to be estimated:

$$\hat{\boldsymbol{\beta}} = \begin{pmatrix} \hat{m} \\ \hat{c} \\ \hat{k} \end{pmatrix}$$

$\hat{\beta}$ is computed to minimise the sum of squared residuals in 6. Assuming the only source of error in the model is white noise in $\tilde{f}(t)$ and X^TX is invertible, $\hat{\beta}$ has the following closed form solution (Montgomery 2021):

$$\beta = (X^TX)^{-1}X^TY \tag{7}$$

This method is simple and computationally inexpensive, but if the pre-processing step doesn't recover the true $x(t)$, the assumptions made in OLS will be violates. However, this method still provides an interesting benchmark.

3.4.2 Orthogonal Regression (Total Least Squares)

Orthogonal regression (ODR) relaxes some of the assumptions made in OLS. In particular, none of the variables are assumed to be measured without error. ODR fits a hyperplane within the multi-dimensional space defined by the observed and estimated variables $\hat{x}$, $\hat{\dot{x}}$, $\hat{\ddot{x}}$ and $\tilde{f}$. This is achieved by minimizing the sum of the squared orthogonal distances between each observed data point (representing a point in this multi-dimensional space) and the hyperplane that represents the solutions to the underlying model (Van Huffel & Vandewalle 1991). In this report ODR will be implemented using SciPy's ODR package. This package solves for $\hat{\beta}$ iteratively with a modified Levenberg-Marquardt algorithm, (Virtanen et al. 2020).

This approach is more computationally expensive than OLS due to the lack of a closed-form solution. It also provides a more statistically sound approach, as it doesn't make the false assumption that x, $\dot{x}$ and $\ddot{x}$ are measured without noise. The SciPy.odr package also allows for the regression problem to be weighted, this can be leveraged to improve predictive accuracy if certain measurements have less error than others.

4 Methods

This section will highlight the methods used to generate data, estimate the derivative terms from this data, estimate the parameters and evaluate model performance with a Monte-Carlo simulation.

4.1 Data Generation

All data is simulated using SciPy.integrate.odeint, (Virtanen et al. 2020), to solve equation 1 for $x(t)$, $\dot{x}(t)$ and $\ddot{x}(t)$, where $f(t) = Fcos(\omega t)$. The values of these functions from $t = 0$ to $t = n\Delta t$, where n is the number of simulated data points, are recorded in lists. $x(0) = 0$ and $\dot{x}(0) = 0$ were used for simplicity. $n = 1000$ and $\Delta t = 0.005s$ were chosen to emulate sensor measurements taken over a five-second interval. A sinusoidal input function was used to ensure the system would be sufficiently excited, and $F \neq 0$ is used to ensure the parameter estimation problem has a unique solution.

Gaussian noise can then be added to any of these functions using the NumPy random.normal function, (Harris, Millman, van der Walt, Gommers, Virtanen, Cournapeau, Wieser, Taylor, Berg, Smith, Kern, Picus, Hoyer, van Kerkwijk, Brett, Haldane, del Río, Wiebe, Peterson, Gérard-Marchant, Sheppard, Reddy, Weckesser, Abbasi, Gohlke & Oliphant 2020). The noise added has $\mu = 0$ and a constant standard deviation that is chosen for each run as a percent of the max absolute value of the function the noise is being added to. The noise was added by generating a random list with the same length as those recording the signal and adding these two lists together. The functions after adding noise are defined as $\tilde{x} = x + \epsilon_x$ and $\tilde{f} = f + \epsilon_f$, where $\epsilon_x \sim \mathcal{N}(\prime, \sigma_{\S}^{\in})$ and $\epsilon_f \sim \mathcal{N}(\prime, \sigma_{\wr}^{\in})$

4.2 Derviative Estimation

Here $\hat{x}$, $\hat{\dot{x}}$ and $\hat{\ddot{x}}$ are generated from $\tilde{x}$. The primary challenge in implementing these methods is hyperparameter optimisation. In this section will therefore focus primarliy on how these parameters were chosen for each method.

4.2.1 Butterworth Low Pass Filter

The Butterworth filter requires two hyperparameters, cut-off frequency and filter order. Determining the optimal choice for these hyperparameters is critical to filter the data correctly. These parameters are optimised through a grid search combined with K-fold cross-validation. The SkLearn KFold function (Pedregosa et al. 2011) is used to break $\tilde{x}$ into five shuffled splits. The performance is evaluated using the following cost function:

$$cost = \frac{1}{n}(\sum_{i=1}^{n}(\hat{x}(t_i) - \tilde{x}(t_i))^2 + \lambda \sum_{i=1}^{n}(\hat{\dot{x}}(t_{i+1}) - \hat{\dot{x}}(t_i))^2) \tag{8}$$

This cost is calculated for all five splits for every combination of λ, cut-off frequency and filter order in the pre-defined search space. The combination with the lowest average cost over the five splits is chosen as the optimal hyperparameters to recover the signal in $\tilde{x}$.

4.2.2 Spline Fitting

Similarly to the low pass filter, two hyperparameters need to be optimised for the spline fitting so that x can be accurately recovered from $\tilde{x}$. The spline degree, k, must be at least four to approximate $\ddot{x}$ as a curve. K-fold cross-validation combined with a grid search will be used to choose suitable values for s and k. Here, the SkLearn GridSearchCV function is used to optimise the hyperparameters, penalising the mean squared error. Once these have been chosen, $\hat{x}(t)$ is fitted to $\tilde{x}(t)$ and this function can be directly differentiated to generate $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$.

4.2.3 Gaussian Process

For the Gaussian Process implementation, all of the optimisation is done by the SkLearn Gaussian Process package. The kernel is defined as a radial basis function (RBF) with white noise added. The length scale of the RBF and noise level of the white noise are optimised over ten restarts. This kernel is used to generate $\hat{x}$ from $\tilde{x}$. $\hat{\dot{x}}$ and $\hat{\ddot{x}}$ are then generated using finite difference differentiation. These could instead be estimated directly in the Gaussian process; however, an implementation of this is beyond the scope of this report and computationally expensive.

4.3 Parameter Regression

Implementing the parameter regression methods does not require any hyperparameter optimisation, making it more straightforward than the pre-processing steps. OLS is implemented directly using equation 7, while ODR is implemented using SciPy.odr. However, the pre-processing techniques tend to lose accuracy near the start and end of the data when estimating the derivative terms. To account for this a data points are removed from the start of $\hat{x}$, $\hat{\dot{x}}$, $\hat{\ddot{x}}$ and $\tilde{f}$ and b data points are removed from the end. For this report, $a = b = 100$. This was chosen to ensure inaccuracies in $\hat{\ddot{x}}$ will be removed while keeping 800 points for the regression. For the ODR case, the standard error in $\hat{x}$, $\hat{\dot{x}}$, $\hat{\ddot{x}}$ and $\tilde{f}$ can be used to weight the regression in favour of more reliable signals. Only the error in $\tilde{f}$ is available but the standard error in $\tilde{x}$ is used for $\hat{x}$ and the standard error's in $\hat{\dot{x}}$ and $\hat{\ddot{x}}$ are treated as five and twenty times that in $\tilde{x}$ respectively. This accounts for the reduced accuracy in the derivative estimates, but could likely be improved with a weighted least squares approach.

4.4 Monte-Carlo Simulation and Performance Evaluation

Monte-Carlo simulations allow for the methods above to be evaluated under controlled conditions. This repeated approach to model evaluation gives a more well-rounded impression of the model's accuracy and robustness. Each trial involves adding new random noise to the data, generating $\hat{x}$, $\hat{\dot{x}}$ and $\hat{\ddot{x}}$ and estimating the parameters. Each simulation includes 300 or 100 of these independent studies.

Each combination of pre-processing and parameter estimation methods will be tested for $\sigma_x = 0.01(max(|x|))$ and $\sigma_f = 0.01(max(|f|))$, for three combinations of m, c and k. This will be done with $F = 1$ and $\omega = 5$. Each combination of methods and parameters will be run 300 times with different random noise added each time, except when using the Gaussian process, which will be run 100 times as it is much more computationally expensive than the low-pass filter and spline fitting methods. In total, eighteen simulations will be run, with each study consisting of 1000 data points captured across five seconds. The three combinations of m, c and k correspond to an under-damped, critically-damped and over-damped mass-spring-damper system. The underdamped case

uses combinations of $m = 1$, $c = 2$ and $k = 9$, the critically damped case uses combinations of $m = 1$, $c = 6$ and $k = 9$, and the overdamped case uses combinations of $m = 2$, $c = 6$ and $k = 2$. This way, the performance of each method can be evaluated over a diverse range of system characteristics.

The performance will largely be evaluated on the histogram of results for $\hat{m}$, $\hat{c}$ and $\hat{k}$ for each simulation. The aim is to produce normally distributed histograms centred around the true value, while maintaining predictive accuracy indicated by a small range of values on the histogram.

5 Results

This section outlines the findings derived from the Monte Carlo simulation described above. For each pre-processing technique, the pre-processing accuracy for each m, c, k combination is followed by the performance of both regression models will be assessed. The pre-processing accuracy will be visualised using time-series plots of the last study in the ODR simulation. A good pre-processing technique should produce estimates very close to the true terms for $x(t)$, $\dot{x}(t)$ and $\ddot{x}(t)$. The performance of the regression models will be visualised using histograms showing the parameter estimates. If both steps are done well, the histograms should be centred around the true value with a small range.

5.1 Low Pass Filter

Figures 1, 2 and 3 show time series plots for the under-, critically- and overdamped cases, including. The true values, $x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$ and $f(t)$, the noisy observations, $\tilde{x}$ and $\tilde{f}$ and the estimates generated using the low pass filter and finite differencing $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$, are shown.

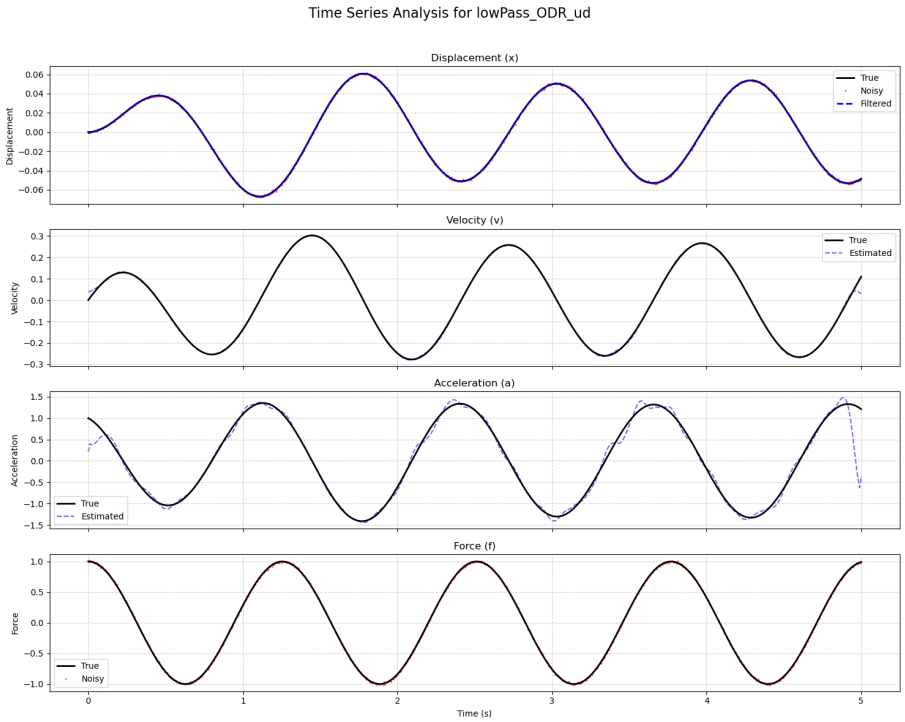

Figure 1: Time-series plots for the under damped system ($m = 1, c = 2, k = 9$). The plot shows a comparison of the true signals ($x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$) with the noisy measured displacement $\tilde{x}(t)$ and the estimated terms $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ obtained using the Butterworth filter.

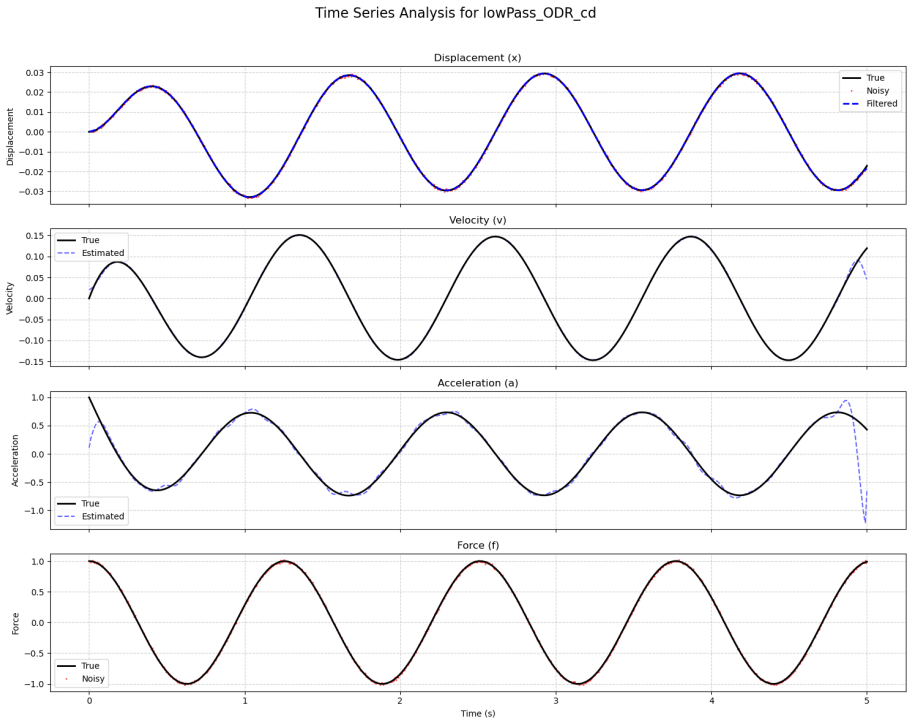

Figure 2: Time-series plots for the critically damped system ($m = 1, c = 6, k = 9$). The plot shows a comparison of the true signals ($x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$) with the noisy measured displacement $\tilde{x}(t)$ and the estimated terms $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ obtained using the Butterworth filter.

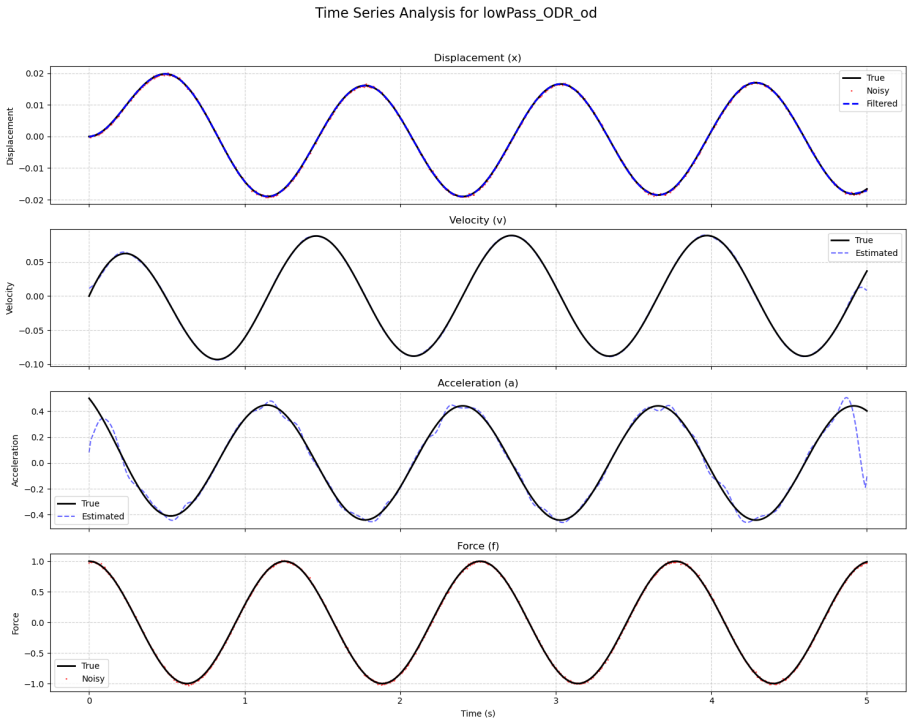

Figure 3: Time-series plots for the over damped system $(m = 2, c = 6, k = 2)$. The plot shows a comparison of the true signals $(x(t),\ \dot{x}(t),\ \ddot{x}(t))$ with the noisy measured displacement $\tilde{x}(t)$ and the estimated terms $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ obtained using the Butterworth filter.

Figures 4, 5 and 6 show the distribution of $\hat{m}$, $\hat{c}$ and $\hat{k}$ for the 300 studies performed using the low pass filter and OLS for the three damping cases.

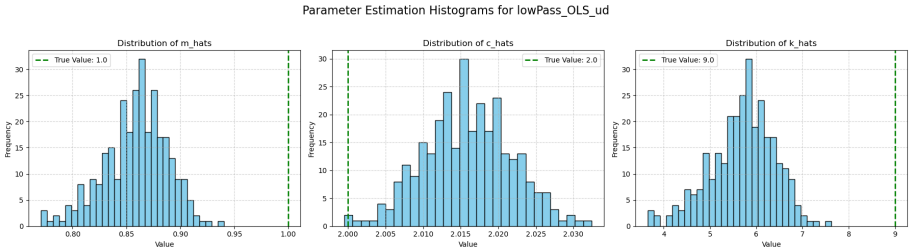

Figure 4: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the under-damped system, obtained using the Butterworth filter for derivative estimation and Ordinary Least Squares (OLS) for parameter fitting. The true parameter values were, $m = 1$, $c = 2$ and $k = 9$

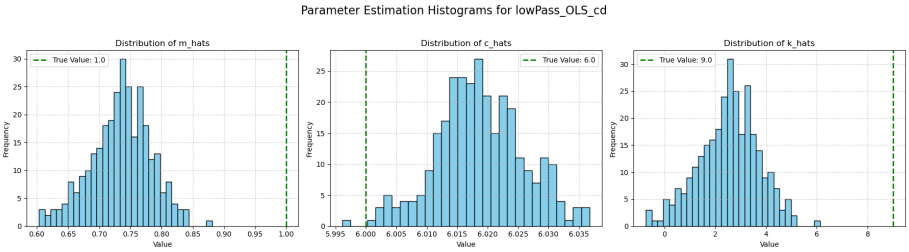

Figure 5: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the critically damped system, obtained using the Butterworth filter for derivative estimation and Ordinary Least Squares (OLS) for parameter fitting. The true parameter values were, $m = 1$, $c = 6$ and $k = 9$

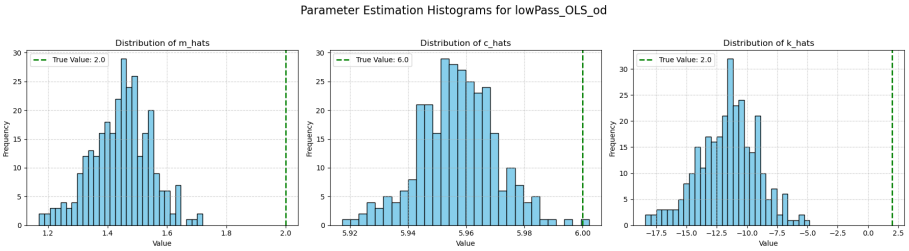

Figure 6: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the critically damped system, obtained using the Butterworth filter for derivative estimation and Ordinary Least Squares (OLS) for parameter fitting. The true parameter values were, $m = 2$, $c = 6$ and $k = 2$

Figures 7, 8 and 9 show the distribution of $\hat{m}$, $\hat{c}$ and $\hat{k}$ for the 300 studies performed using the low pass filter and ODR for the three damping cases.

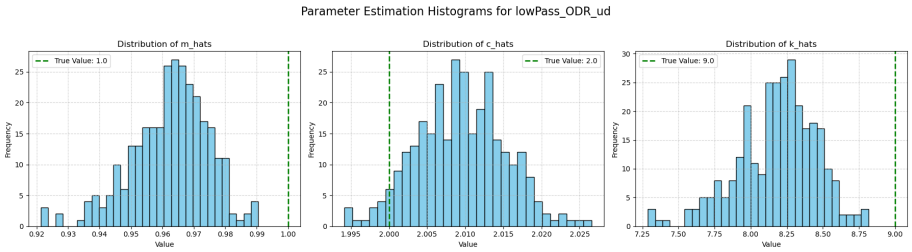

Figure 7: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the over-damped system, obtained using the Butterworth filter for derivative estimation and Orthogonal Regression (ODR) for parameter fitting. The true parameter values were, $m = 1$, $c = 2$ and $k = 9$

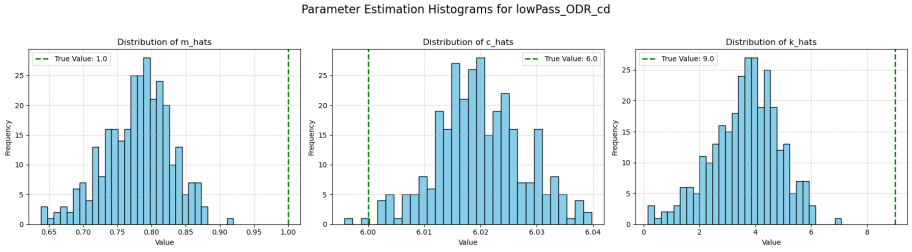

Figure 8: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the critically damped system, obtained using the Butterworth filter for derivative estimation and Orthogonal Regression (ODR) for parameter fitting. The true parameter values were, $m = 1$, $c = 6$ and $k = 9$

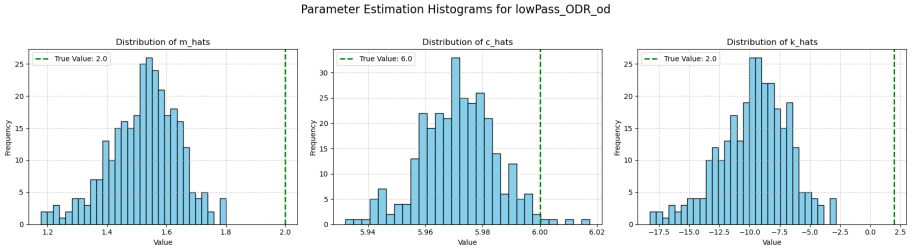

Figure 9: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the over-damped system, obtained using the Butterworth filter for derivative estimation and Orthogonal Regression (ODR) for parameter fitting. The true parameter values were $m = 2$, $c = 6$, and $k = 2$.

5.2 Spline Fitting

Figures 10, 11 and 12 show time series plots for the under-, critically- and over-damped cases including. The true values, $x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$ and $f(t)$, the noisy observations, $\tilde{x}$ and $\tilde{f}$ and the estimates generated by fitting and differentiating splines, $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$, are shown.

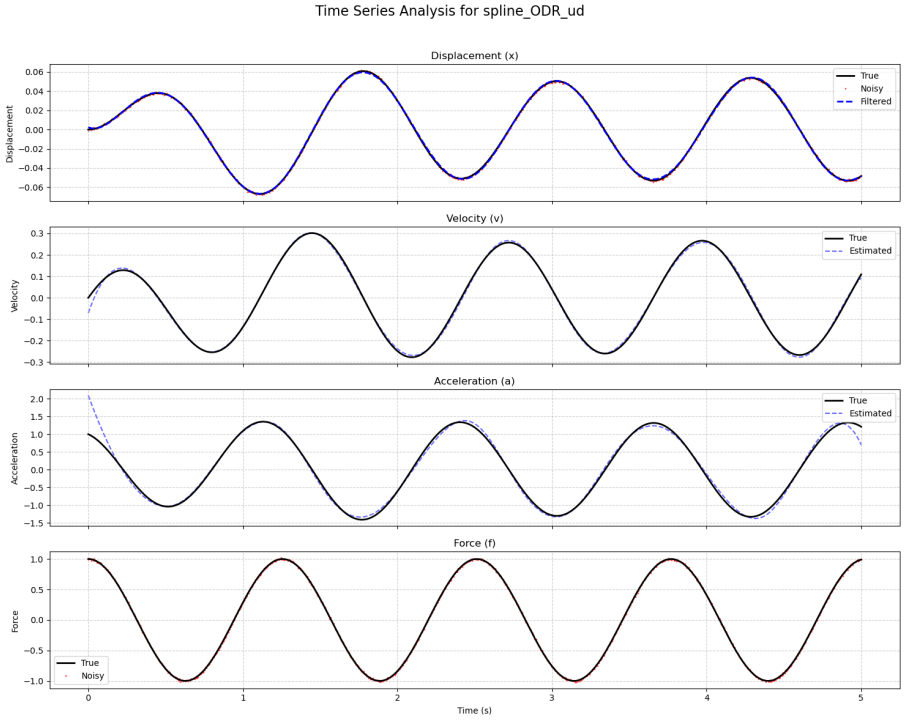

Figure 10: Time-series plots for the under damped system ($m = 1, c = 2, k = 9$). The plot shows a comparison of the true signals ($x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$) with the noisy measured displacement $\tilde{x}(t)$ and the estimated terms $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ obtained using univariate spline fitting.

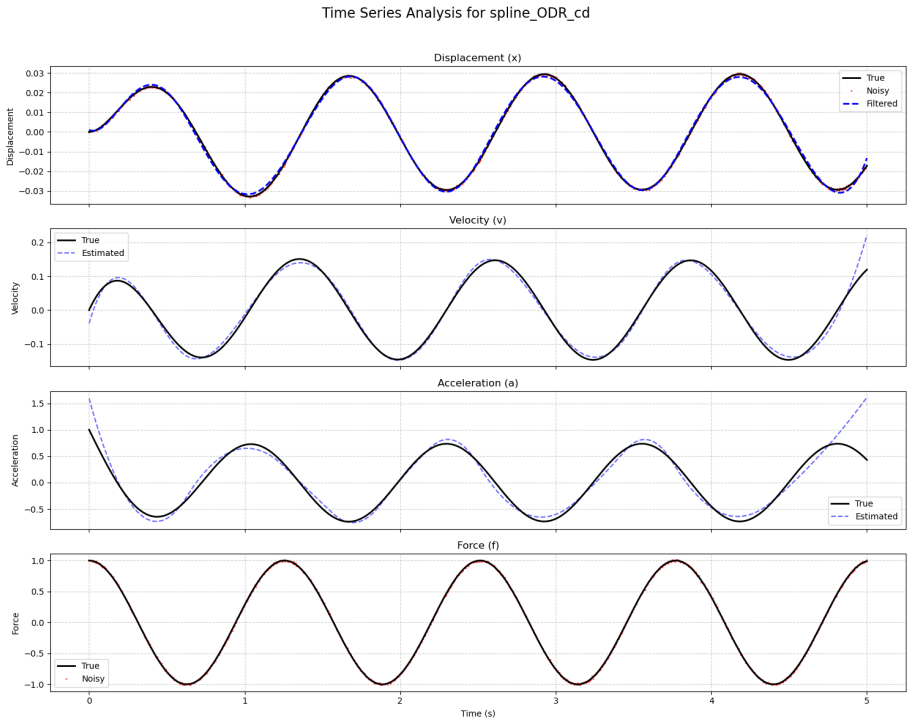

Figure 11: Time-series plots for the critically damped system ($m = 1, c = 6, k = 9$). The plot shows a comparison of the true signals ($x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$) with the noisy measured displacement $\tilde{x}(t)$ and the estimated terms $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ obtained using univariate spline fitting.

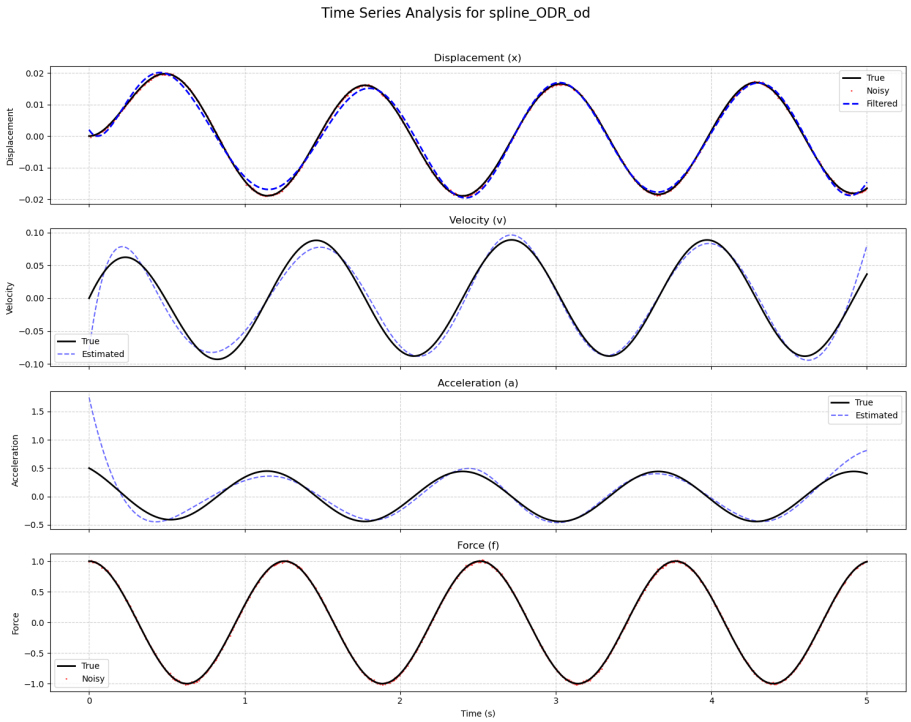

Figure 12: Time-series plots for the over damped system ($m = 2, c = 6, k = 2$). The plot shows a comparison of the true signals ($x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$) with the noisy measured displacement $\tilde{x}(t)$ and the estimated terms $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ obtained using univariate spline fitting.

Figures 13, 14 and 15 show the distribution of $\hat{m}$, $\hat{c}$ and $\hat{k}$ for the 300 studies performed using spline fitting and OLS for the three damping cases.

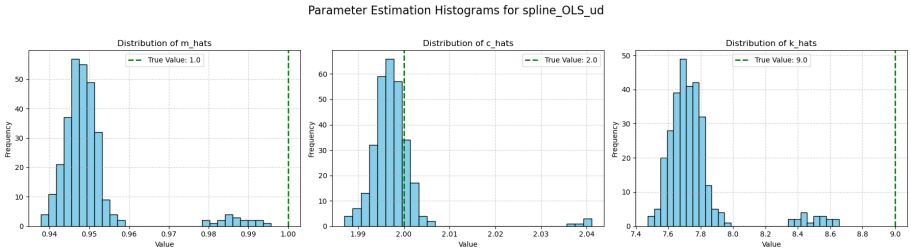

Figure 13: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the under-damped system, obtained using spline fitting for derivative estimation and Ordinary Least Squares (OLS) for parameter fitting. The true parameter values were, $m = 1$, $c = 2$ and $k = 9$

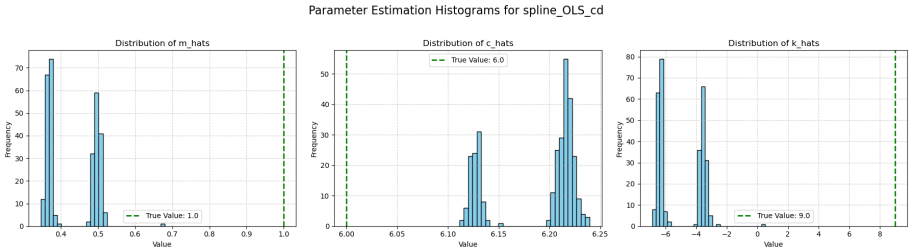

Figure 14: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the critically-damped system, obtained using spline fitting for derivative estimation and Ordinary Least Squares (OLS) for parameter fitting. The true parameter values were, $m = 1$, $c = 6$ and $k = 9$

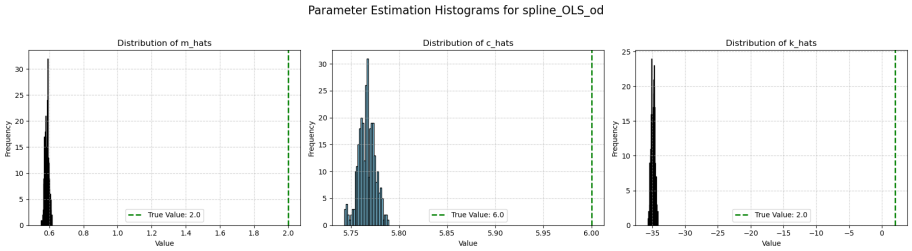

Figure 15: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the over-damped system, obtained using spline fitting for derivative estimation and Ordinary Least Squares (OLS) for parameter fitting. The true parameter values were, $m = 2$, $c = 6$ and $k = 2$

Figures 16, 17 and 18 show the distribution of $\hat{m}$, $\hat{c}$ and $\hat{k}$ for the 300 studies performed using spline fitting and ODR for the three damping cases.

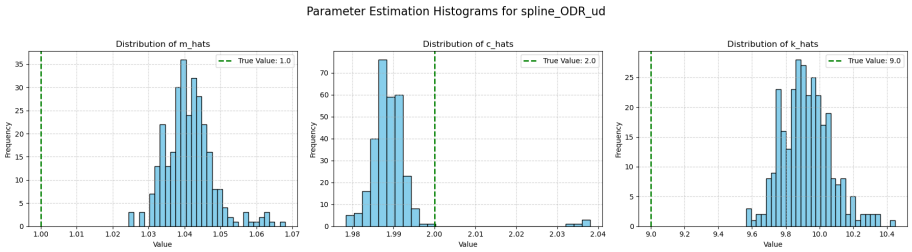

Figure 16: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the underdamped system, obtained using spline fitting for derivative estimation and Orthogonal Regression (ODR) for parameter fitting. The true parameter values were, $m = 1$, $c = 2$ and $k = 9$

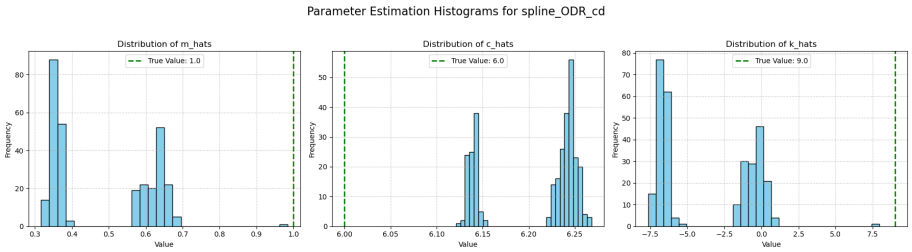

Figure 17: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the critically damped system, obtained using spline fitting for derivative estimation and Orthogonal Regression (ODR) for parameter fitting. The true parameter values were, $m = 1$, $c = 6$ and $k = 9$

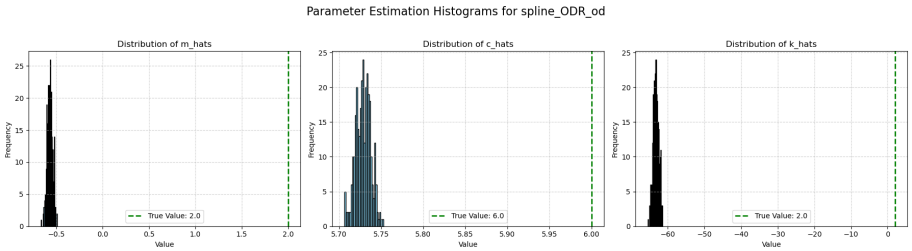

Figure 18: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the overdamped system, obtained using spline fitting for derivative estimation and Orthogonal Regression (ODR) for parameter fitting. The true parameter values were, $m = 2$, $c = 6$ and $k = 2$

5.3 Gaussian Process

Figures 19, 20 and 21 show time series plots for the under-, critically- and over-damped cases including. The true values, $x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$ and $f(t)$, the noisy observations, $\tilde{x}$ and $\tilde{f}$ and the estimates generated using a Gaussian process and finite differencing $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$, are shown.

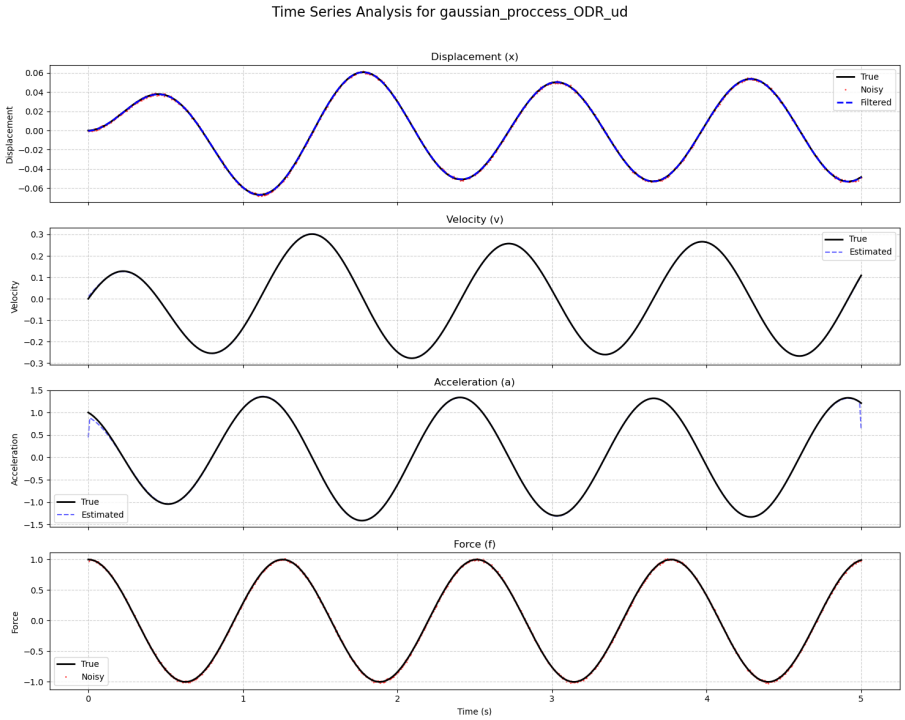

Figure 19: Time-series plots for the under damped system ($m = 1, c = 2, k = 9$). The plot shows a comparison of the true signals ($x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$) with the noisy measured displacement $\tilde{x}(t)$ and the estimated terms $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ obtained using a Gaussian process.

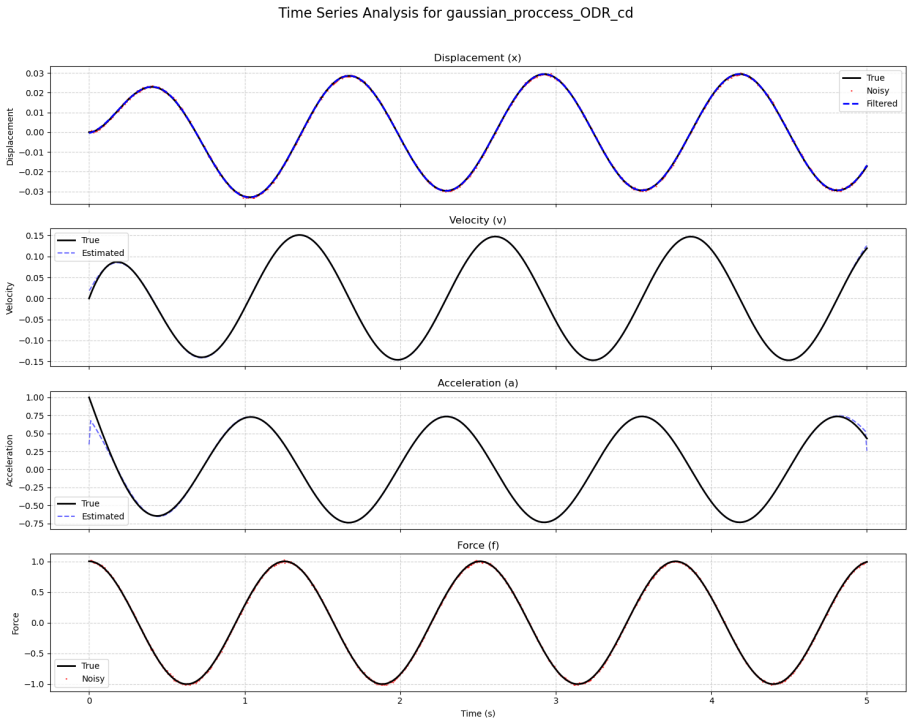

Figure 20: Time-series plots for the critically damped system ($m = 1, c = 6, k = 9$). The plot shows a comparison of the true signals ($x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$) with the noisy measured displacement $\tilde{x}(t)$ and the estimated terms $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ obtained using a Gaussian process.

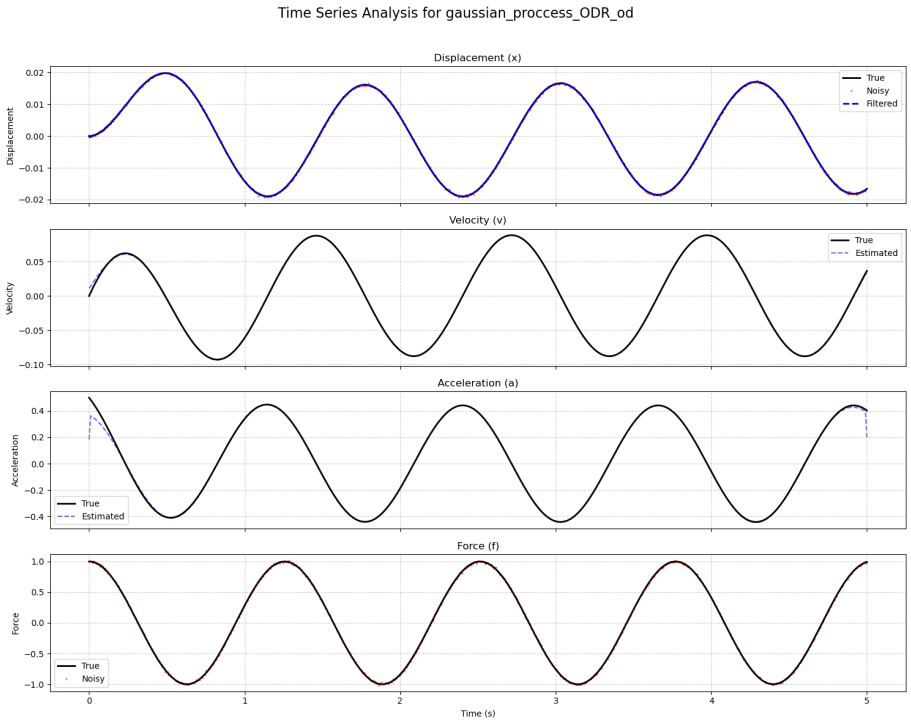

Figure 21: Time-series plots for the over damped system ($m = 2, c = 6, k = 2$). The plot shows a comparison of the true signals ($x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$) with the noisy measured displacement $\tilde{x}(t)$ and the estimated terms $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ obtained using a Gaussian process.

Figures 22, 23 and 24 show the distribution of $\hat{m}$, $\hat{c}$ and $\hat{k}$ for the 300 studies performed using a GP and OLS for the three damping cases.

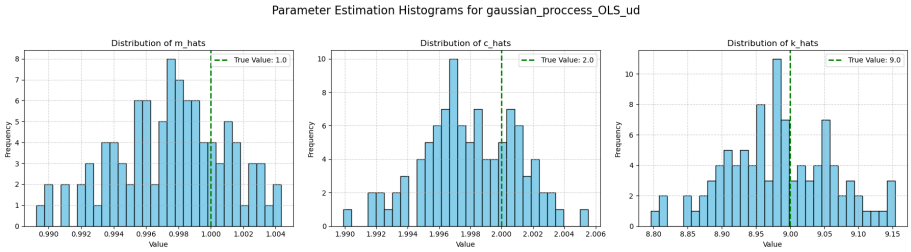

Figure 22: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the under-damped system, obtained using a Gaussian process for derivative estimation and Ordinary Least Squares (OLS) for parameter fitting. The true parameter values were, $m = 1$, $c = 2$ and $k = 9$

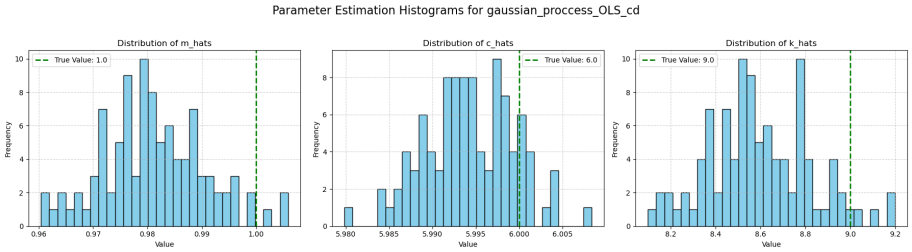

Figure 23: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the critically-damped system, obtained using a Gaussian process for derivative estimation and Ordinary Least Squares (OLS) for parameter fitting. The true parameter values were, $m = 1$, $c = 6$ and $k = 9$

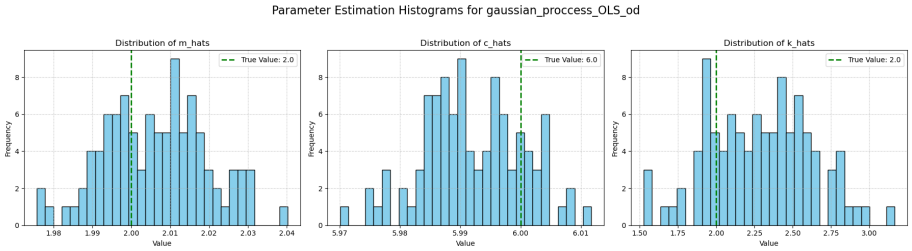

Figure 24: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the over-damped system, obtained using a Gaussian process for derivative estimation and Ordinary Least Squares (OLS) for parameter fitting. The true parameter values were, $m = 1$, $c = 6$ and $k = 9$

Figures 25, 26 and 27 show the distribution of $\hat{m}$, $\hat{c}$ and $\hat{k}$ for the 300 studies performed using a GP and ODR for the three damping cases.

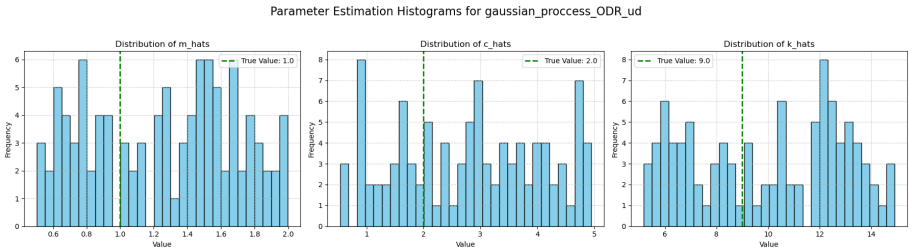

Figure 25: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the underdamped system, obtained using a Gaussian process for derivative estimation and Orthogonal Regression (ODR) for parameter fitting. The true parameter values were, $m = 1$, $c = 2$ and $k = 9$

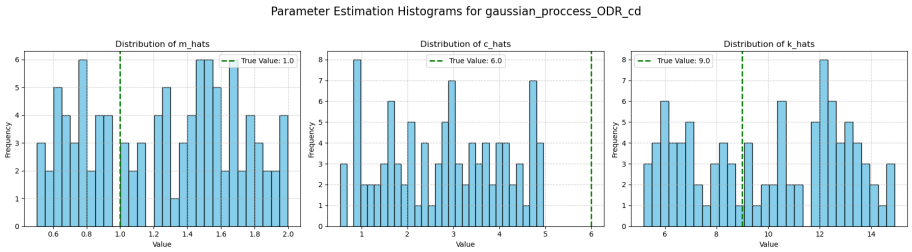

Figure 26: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the critically-damped system, obtained using a Gaussian process for derivative estimation and Orthogonal Regression (ODR) for parameter fitting. The true parameter values were, $m = 1$, $c = 6$ and $k = 9$

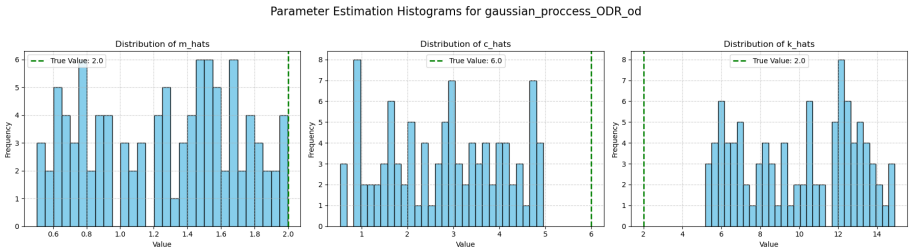

Figure 27: Histograms of the estimated parameters $\hat{m}$, $\hat{c}$, and $\hat{k}$ for the overdamped system, obtained using a Gaussian process for derivative estimation and Orthogonal Regression (ODR) for parameter fitting. The true parameter values were, $m = 2$, $c = 6$ and $k = 2$

6 Discussion

This Monte Carlo simulation provides a rich data set to evaluate the methods for parameter estimation for systems governed by equation 1. This section aims to interpret this data to evaluate the strengths and weaknesses of each method while analysing the limitations in the study and suggesting directions for future work.

6.1 Performance of derivative estimation methods

This section will interpret the time-series plots and some histograms presented in the results section to discuss the accuracy demonstrated by each of the pre-processing and derivative estimation methods highlighted by the report.

6.1.1 Low Pass Filter and Finite Differencing

This method generated $\hat{x}(t)$ and $\hat{\dot{x}}(t)$ very well, to a visual inspection recovering $x(t)$ and $\dot{x}(t)$ almost perfectly. However, there are definite issues with $\hat{\ddot{x}}(t)$. In all three damping cases, there are noticeable oscillations present in $\hat{\ddot{x}}(t)$ that aren't present in $\ddot{x}(t)$. These oscillations could be reduced with a better choice of cut-off frequency and filter order. It is unclear whether these sub-optimal hyperparameter choices are a result of the grid search, as the true optimal values may have been between two choices in the grid, or if these inaccuracies are a result of using cross-validation on the noisy data.

6.1.2 Spline Fitting

This method showed serious issues, $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ are out of phase with, and occasionally have different magnitudes to $x(t)$, $\dot{x}(t)$ and $\ddot{x}(t)$. These issues seem to be worse for the overdamped case and the least prevalent for the underdamped case. This could suggest that the method is more effective for underdamped systems than systems in the other two damping regimes. This claim is substantiated by the histograms, which show less bias in the estimates for the underdamped case compared to the other two. Similarly to the low-pass filter, these issues could be down to the choice of hyperparameters. In particular, the chosen smoothing factor s is likely too high, as the chosen splines seem to prioritise "smoothness" over fit to the original data. The hyperparameters are chosen through generalised cross-validation with a grid search. The issue could reside in the grid search approach, as the best value for s could be between two options or a result of the cross-validation technique. It is also worth noting that the histograms where spline fitting is used display two peaks for the parameter estimates. This phenomenon is likely due to different hyperparameters being selected on each run. A refinement of the Monte Carlo simulation to save the chosen hyperparameters for each study would reveal, for certain, whether or not this is the case. If it is, this highlights a worrying instability of the method.

6.1.3 Gaussian Process

The Gaussian process performed better than the other two pre-processing methods and derivative estimation techniques in all three damping regimes. Excluding the first and last 0.5s as is done for the regression, $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ recover the true functions so precisely no difference can be seen in visual inspection. The greater accuracy than the other two methods could be driven by a better choice of hyperparameters, as these are optimised within the SkLearn functions used. However, it should be noted that this method is significantly more computationally expensive than the other two.

6.2 Impact of Regression Method on Parameter Estimates

In this section, the strengths and weaknesses of ordinary least squares and orthogonal least squares for estimating m, c and k will be evaluated. This will be done through visual inspection of the histograms in the results section. This is a complex task as the differences between OLS and ODR changed for each derivative estimation technique. Where the derivative estimations were close but visibly imperfect, such as when the low-pass filter was used in the underdamped case, ODR reduced the bias and variance of predictions. When the derivative estimates were less accurate than this, such as when the spline fitting method was used, ODR gave similar or worse results than OLS. Finally, when the derivative terms are recovered very well, such as when using the GP, OLS performs much better than ODR, because the assumption that the only measurement error is in $\tilde{f}(t)$ is a good one. Overall, it seems that OLS outperforms ODR when $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ are very accurate or very inaccurate, but there is a region of accurary in which ODR is better for recovering the parameters.

6.3 Limitations of the Study and Suggestions for Future Work

The three derivative estimation techniques all require hyperparameters to be chosen. In this report, the Gaussian process had the most advanced hyperparameter optimisation algorithm as it's included in the SkLearn package and the other two algorithms were implemented with a grid search. For a fairer comparison between the three methods, better algorithms should be implemented for the low-pass filter and spline-fitting methods. The cross-validation and generalised cross-validation approaches could remain, but moving away from the grid search would likely result in better derivative estimates.

This report focuses exclusively on simulated data, emulating sensor data with discrete and noisy measurements of functions. This is much better for generating a custom data set to evaluate the performance of these methods. That being said, using experimental data would give these methods greater credibility and could uncover new challenges that certain methods are better equipped for than others.

A natural expansion would be to combine these methods with general equation discovery methods in the future, such as SINDy. This could allow these methods to be used on systems governed by any ODE.

The data analysis in this report was done just through visual inspections of graphs. However, modern statistical and data analytics techniques could offer interesting insights into the data that aren't obvious in visual inspection. This, combined with some tweaks to the data generation process, such as saving the chosen hyperparameters for each run, could allow for a more well-rounded understanding of how systems like this can be modelled from noisy and discrete measurements of their input and output functions.

7 Conclusion

This section aims to summarise the key findings in the report and provide the final conclusions that can be drawn from these findings.

7.1 Summary of Key Findings

The objective of this report was to investigate and evaluate different methods for estimating m, c and k in equation 1 from noisy and discrete measurements of $f(t)$ and $x(t)$. The Gaussian process-based method for generating $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ proved to be the most accurate, recovering the true functions almost perfectly across all three damping regimes. The Butterworth low-pass filter-based method also showed some promise. The generated $\hat{\ddot{x}}(t)$ was not as accurate as when generated using the GP, but was close in all three damping regimes and likely could be improved by moving away from grid search optimisation. The spline fitting method proved to be the least accurate and reliable. Similarly to the low-pass filter, $\hat{\ddot{x}}(t)$ could likely have been more accurate with a better hyperparameter optimisation algorithm. However, the spline fitting method had some worrying issues, such as the two peaks in some histograms, suggesting different hyperparameters could be chosen for the same signal based on the noise. This makes the method less reliable, and there's no guarantee that moving away from grid search would fix this problem.

The choice of regression technique seems to be dependent on the accuracy of $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$. The best results were generated using OLS and the Guassian Process, as when $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ are very accurate the assumption that these three variables are measured without noise is a good one. When these functions are slightly less accurate, such as when the low-pass filter is used, ODR is the best regression method. The bias and variance of predictions are reduced by acknowledging $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ contain error and that $\hat{\ddot{x}}(t)$ contains the most. For highly inaccurate derivative estimates, such as for the spline fitting method, OLS performs the same or better than ODR even though there is error in all variables.

7.2 Outlook for Future Work

This report offers a clear and robust, two-stage method for estimating m, c and k in 1 from noisy and discrete measurements of $f(t)$ and $x(t)$, using a Gaussian process for generating $\hat{x}(t)$, $\hat{\dot{x}}(t)$ and $\hat{\ddot{x}}(t)$ with OLS to generate $\hat{m}$, $\hat{c}$ and $\hat{k}$. This report also provides a method for evaluating the accuracy and robustness of different modelling techniques. With improvements to hyperparameter selection and the analysis of the data generated by the Monte Carlo simulation, this could provide a framework for more advanced work in the future.

Going forward, it would be advantageous to integrate experimental data into these methods to ensure they are usable in real-world scenarios. These methods could also be integrated into equation discovery techniques such as SINDy, making systems identification for systems governed by ordinary differential equations a lot easier.

References

Brunton, S. L., Proctor, J. L. & Kutz, J. N. (2016), 'Discovering governing equations from data by sparse identification of nonlinear dynamical systems', *Proceedings of the National Academy of Sciences* **113**(15), 3932–3937.

Butterworth, S. (1930), 'On the theory of filter amplifiers', *Wireless Engineer* **7**, 536–541.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C. & Oliphant, T. E. (2020), 'Array programming with NumPy', *Nature* **585**(7825), 357–362.
URL: *https://doi.org/10.1038/s41586-020-2649-2*

Ljung, L. (1998), *System Identification: Theory For The User*, Prentice Hall.

Meirovitch, L. (2011), *Fundamentals of Vibrations*, Waveland Press.

Montgomery, Peck, V. (2021), *Introduction to Linear Regression Analysis*, Wiley.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. et al. (2011), 'Scikit-learn: Machine learning in python', *Journal of machine learning research* **12**(Oct), 2825–2830.

Rasmussen, C. E. & Williams, C. K. I. (2006), *Gaussian Processes for Machine Learning*, MIT Press.

Van Huffel, S. & Vandewalle, J. (1991), *The Total Least Squares Problem: Computational Aspects and Analysis*, siam.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P. & SciPy 1.0 Contributors (2020), 'SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python', *Nature Methods* **17**, 261–272.

Wentz, J. & Doostan, A. (2023), 'Derivative-based sindy (dsindy): Addressing the challenge of discovering governing equations from noisy data', *Computer Methods in Applied Mechanics and Engineering* .

Yew, A. C. (2011), *Numerical Methods for Engineers and Scientists*, McGraw-Hill Education.

A Code and Data Availability

All code, data, and scripts used for the Monte-Carlo simulations and analysis in this report are publicly available on GitHub at the following URL:

`https://https://github.com/cormac005/Summer2025Research`

The repository is organised as follows:

- **/src/**: Contains all Python scripts.
- **/data/**: Contains the raw data generated from the simulations.