Literature Review and Implementation of Parameter Identification Methods for Multibody Systems Governed by Differential Algebraic Equations

Christopher Frewin
University of Applied Sciences Upper Austria, Wels, Austria

August 1st, 2013

Abstract

A variety of parameter identification literature is reviewed and considered for implementation on a pendulum system. The parameter identification methods reviewed represent a wide range of processes, including system-specific methods, stochastic methods, methods in which state space equations are the governing equations of motion, and methods in which differential algebraic equations (DAEs) with Lagrange multipliers are the governing equations of motion. A large amount of literature reduce second order DAEs of motion into ordinary differential equations (ODEs). The focus of this study is on parameter identification methods that deal directly with the DAEs of motion, without requiring the algebraic constraint equations to be eliminated. Only three pieces of literature were found that meet this requirement. One involving a first order sensitivity analysis is implemented, in which a sensitivity ODE is formulated directly from partial derivatives of the DAEs of motion. This method is compared to a finite differencing method, in which numerical approximations are used to calculate the gradient vector and Hessian matrix. The DAEs of motion for a coupled spring-damper pendulum system are derived and used an example system for both parameter identification methods. APMonitor Optimization Suite software is used together with MATLAB to solve the reduced order model of the pendulum system and perform all other computations. The benefits and drawbacks of both parameter identification methods are shown by rigorously testing and comparing each. For the pendulum system, the finite differencing approximation method is slightly more efficient at identifying optimum parameters than the first order sensitivity analysis method.

Contents

1 Introduction

Multibody system modeling is essential for designing, testing, and inverse modeling of complex mechanical systems. Generally, the equations of motion for a multibody dynamic system are second order nonlinear DAEs. For a particular model, the numerical properties of the DAEs are determined by the model's topology and by parameters including involved bodys' masses and inertias, springs' stiffnesses, dampers' damping coefficients, and other components in the system, such as moments of inertia and dimensions of components. Assuming a meaningful topology, the system's response is determined by the numerical value of these various parameters. Differences from simulated data and measured data are due to deviations between the numerical parameters of the model and the values of the real parameters present in the actual system. A common problem is that in complex multibody systems, specific system parameters like spring stiffness and damping coefficients are difficult and/or expensive to determine experimentally. Additionally, data provided by manufacturers for specific components of a multibody system typically do not represent the full dynamic behavior of the component under consideration, especially when many of these components are coupled together as is often the case in multibody systems.

The discipline of parameter identification deals with the determination of such parameters using a numerical simulation based on a model of the system under consideration. This avoids the otherwise more expensive and time-consuming method of experimentally determining specific system parameters one by one. The typical process for determining the parameters with a model of a system is by using optimization methods to find particular objective functions' minima or maxima. The objective functions represent the difference between measured data from an experiment and simulated data from a model, and are most commonly found in the literature as of the Gaussian least squares type. As a relatively new science, the majority of publications on the topic of parameter identification have been produced within the last 30 years.

2 Problem Statement

2.1 Overview

Parameter identification requires a comparison of experimental states responses $\boldsymbol{q}_{exp}$ with mathematical-model determined states $\boldsymbol{q}_{mod}$. The $\boldsymbol{q}_{exp}$ state responses can be determined directly from a real experiment, or artificial fabricated by using a mathematical model of an experimental system. $\boldsymbol{q}_{mod}$ values are found by solving either DAEs or ODEs that represent the multibody system under consideration. To best match $\boldsymbol{q}_{mod}$ and $\boldsymbol{q}_{exp}$, parameters are changed in value, and therefore the modeled system responses $\boldsymbol{q}_{mod}$ also change. The change in the value of parameters is the aspect which parameter identification methods deal with. A variety of methods have different applications to the types of systems they can be applied to, the type of experimental measurements on the system that much be taken, and the type of error associated with the experimentation on the system.

For many pieces of literature, a state space system of either second or first order ODEs is required, in which the algebraic constraint equations of the more commonly found second order DAE equations of motion are removed. Section 2.3 describes the process of index reduction which removes the algebraic constraint to reduce second order DAEs of motion to second order ODEs.

2.2 General Formulation and Parameter Identification Process

The most general description of the equations of motion for a mechanical system is given by the following DAE system:

$$\mathbf{M}(\boldsymbol{q},\, \boldsymbol{p})\ddot{\boldsymbol{q}} + \mathbf{J}(\boldsymbol{q})^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{Q}(t,\, \boldsymbol{q},\, \dot{\boldsymbol{q}},\, \boldsymbol{p}) \tag{1}$$

$$\boldsymbol{\Phi}(t,\, \boldsymbol{q},\, \boldsymbol{p}) = \mathbf{0} \tag{2}$$

where $\mathbf{M}$ is the mass matrix of the system, $\ddot{\boldsymbol{q}}$ is the generalized acceleration vector, $\mathbf{J}$ is the Jacobian matrix, $\boldsymbol{\lambda}$ is a vector of the Lagrange multipliers, and $\mathbf{Q}$ is the generalized force vector. $\boldsymbol{\Phi}$ is the vector of constraint equations. In some publications, the Jacobian matrix is referred to as $\boldsymbol{\Phi}_{\boldsymbol{q}}$, as it can be found by calculating $\frac{\partial \boldsymbol{\Phi}}{d\boldsymbol{q}}$. Vector $\boldsymbol{p}$ is the vector of p_1, p_2, p_3, ..., p_j unknown parameters which are to be identified, sometimes known as design variables. The unknown parameters are also referred to as vector $\boldsymbol{b}$ or $\boldsymbol{\theta}$ in some literature. Vector $\boldsymbol{q}$ is the vector of q_1, q_2, q_3, ..., q_n states responses of the system.

All DAE-based parameter identification methods seek to minimize the least squares type objective function, V, in which the difference between n experimental system states $\boldsymbol{q}_{exp}$, and n simulated states, $\boldsymbol{q}_{mod}$, are squared. In discrete form with time steps k and total time of the experiment t_i, the objective function is given as:

$$V(\boldsymbol{p}) = \frac{1}{2}\sum_{i=1}^{k}(q_{exp,i} - q_{mod,i}(\boldsymbol{p}))^2 \tag{3}$$

The objective function is also referred to as $\Phi(\boldsymbol{p})$, $\psi(\boldsymbol{p})$, or the 'cost function' in some literature. By minimizing the objective function, unknown parameters can be identified. In this project, no actual system was used to determine experimental responses, nor is noise from data acquisition devices considered. So to determine $\boldsymbol{q}_{exp}$, parameters values are selected and the response of the states in the system found by solving the DAE using those parameters is treated as the experimental data. $\boldsymbol{q}_{mod}$ values are determined from the same DAE but with a different parameters. For parameter identification methods using DAEs as the governing equations of the system, which is the focus of this work, the general process for identifying parameters is:

1. Derive equations of motion for the system under consideration (ODEs or DAEs)
2. Obtain experimental data from system under consideration (data of states $\boldsymbol{q}_{exp}$), either by real experiment or virtual experiment
3. Solve ODEs or DAEs to get $\boldsymbol{q}_{mod}$ values
4. Change $\boldsymbol{p}$ according to parameter identification method being used, and thus $\boldsymbol{q}_{mod}$ and the objective function Eq. (3)
5. Resolve DAEs or ODEs with new parameters $\boldsymbol{p}$ to get new $\boldsymbol{q}_{mod}$ values
6. Repeat steps 4 and 5 until Eq. (3) is at a minimum. The solution of best parameters is $\boldsymbol{p}^* = \boldsymbol{p}$

2.3 Index Reduction of Differential Algebraic Equations

Consider a single iteration of Eqs. (1) and (2) in the parameter searching process, in which a given vector of parameters $\boldsymbol{p}$ is held constant:

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{J}(\boldsymbol{q})^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{Q}(t,\, \boldsymbol{q},\, \dot{\boldsymbol{q}}) \tag{4}$$

$$\boldsymbol{\Phi}(\boldsymbol{q}) = \mathbf{0} \tag{5}$$

In order to change the second order DAE system described by Eqs. (4) and (5) into to a second order ODE system, the constraint Eq. (5) is twice differentiated with respect to time. The first time derivative is the constraint equation times the inner derivative of the states:

$$\boldsymbol{\Phi}(\boldsymbol{q})\dot{\boldsymbol{q}} = \mathbf{0} \tag{6}$$

The second time derivative requires the use of the chain rule:

$$\boldsymbol{\Phi}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{\Phi}}(\boldsymbol{q})\dot{\boldsymbol{q}} = 0 \tag{7}$$

Or, writing Eq. (7) so only the $\ddot{\boldsymbol{q}}$ term is on the left hand side of the equation:

$$\boldsymbol{\Phi}(\boldsymbol{q})\ddot{\boldsymbol{q}} = -\dot{\boldsymbol{\Phi}}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{8}$$

Combining Eq. (8) with Eq. (4), the DAE system in Eqs. (4) and (5) can be written instead as a second order ODE, where no algebraic constraint equation is required:

$$\begin{bmatrix} \mathbf{M} & \mathbf{J}^{\mathrm{T}} \\ \mathbf{J} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{Q}(\boldsymbol{q}, \dot{\boldsymbol{q}}, t) \\ -\dot{\boldsymbol{\Phi}}\dot{\boldsymbol{q}} \end{bmatrix} \tag{9}$$

This type of system is commonly found in the literature and many solution methods are available to determine state versus time, $\boldsymbol{q}(t)$ values, which are required for parameter identification methods. As will be discussed in the literature review, papers by Schiehlen et. al [11], Blanchard et al. [12], and Vyasarayani et. al. [16] [17] use this type of system of equations as the governing equations of motion for the parameter identification process. However, for complex mechanical systems with Langrange multipliers, it is sometimes difficult or impossible to reduce the system to a state-space model, since second order time derivatives of the constraint equations can be difficult to determine. For this reason, this project focuses on methods that use only the original DAEs of motion, i.e. Eqs. (1) and (2), and do not require index reduction.

3 Literature Review

3.1 Parameter Identification Methods Considered

3.1.1 System-Specific Parameter Identification Methods

Literature considering general mechanical parameter identification methods is the focus of this study. A large amount of work in parameter identification has been completed on specific mechanical systems and other electro-chemical systems, but these studies do not include enough theoretical framework to allow the parameter identification process to be utilized on any other system. Two good examples of literature that use very specific types of parameter identification methods on specific systems are by Ok et al. [1], and Kim et al. [2].

Ok et al. [1] examined parameter identification specifically for rubber brushing elements of vehicle suspension. Rubber brushing elements are damping elements that have a dependence on their past history of expansion and compression. Therefore Ok et al. fit a non-linear hysteretic model, the Bouc-Wen differential model [3] [4], on the brushing system. The identification method uses a finite differencing equation of values of the root mean square of the previous parameter estimate and the new parameter estimate. A total of nine parameters are identified from the brushing element, first tested using a simple compression-force model, and then with a 47 degree of freedom system modeling a rear suspension of an SUV. The proposed hysteric model improves upon the Kelvin-Voight damping model [5] which is used in many commercial dynamics programs.

Kim et al. [2] studied parameter identification for structural damping, frictional damping, and viscoelastic damping on a beam. Kim et al. consider three main types of damping models: linear damping, quadratic damping, and Coulomb damping. This work also considers the Kelvin-Voight model [5] and an improved method using the Bouc-Wen [3] [4] model. It was found for structural damping that linear damping methods were sufficient, while for more specific methods such as damping on a flat plate including air resistance, quadratic damping was required. For frictional damping, Coulomb damping was best.

3.1.2 Genetic Algorithm Methods

A very recent solution to the problem of parameter identification produced in 2013 utilizes genetic algorithms, also known as genetic optimization. These methods are reviewed and described by Ludwig et al. [6], where a cost function, i.e. the simulation error, is utilized instead of iteratively minimizing the more commonly found Gaussian least squares objective function. In this process, the entire simulation is run and then the total simulation error is calculated and stored. To select the next parameters, a normal distribution of possible parameters candidates establish the next possible selection of parameters. This method is highly advantageous for robotic systems which utilize control systems to maintain their movement accuracy.

A further benefit of the genetic algorithm is that it does not require the use of the equations of motion of the system, only the output of the system, so the equations of motion can be treated as a ‘black box’. In these types of computations, the minimum allowed distance between each subsequent 'generation' of parameter selection is a critical value in terms of convergence time and accuracy. In general, a smaller distance value provides better optimum parameters, but a longer convergence time. Two other studies use this genetic optimization to identify parameters.

Ludwig et al. [7] use the genetic method to calculate parameters for a theoretical belt drive system to identify the axial stiffness and controller gain parameters. The genetic method is also used to measure the time delay of the measured torque of a real robot. In this more realistic complex system with the robot, a total of five parameters are determined, including the measurement delay, gear stiffness, relative gear damping, tool mass, and drive inertia of the driving motor.

Reischl et al. [8] use the genetic parameter identification method to identify parameters in a nonlinear viscoelastic model for the injection process in molding machines. As described in [6] and [7], objective function values are calculated for a variety of parameter sets and then organized according to best (lowest) objective function values. A total of nine parameters are identified for the injection process, six of which are during the filling phase and three of which are during the packing phase. Only two experimental measurements from the entire molding process are required for the identification process.

3.1.3 Inertial and Essential Parameter Identification Methods

Other examples of literature in parameter identification can be found in Ebrahimi et al. [9], in which inertial parameters are estimated using a unit-homogenized regression matrix of the system under consideration. The regression matrix is used to determine the essential parameters, which are linear combinations of the actual parameters. These essential parameters provide the knowledge of which parameters in the model are most critical for the equations of motion to accurately match the actual experimental response of a system. Ebrahimi et al. consider a system in which the equations of motion are reformulated in terms of the inertial parameters. A six degree of freedom dual-pantograph robotic system is used as the testing device. Seven essential parameters are studied with the system to understand their sensitivities with respect to the dynamics of the system. An advantage to this method is that the physical aspects critical to the system functioning properly can be determined. A drawback is that specific parameters such as a component mass or involved springs' spring constant cannot be determined.

Shome et al. [10] describe a similar process, in which dynamic and inertial parameters are separated during the derivation and identification process of finding the optimum parameters. Single value decomposition is used to determine the essential parameters and minimum parameters. A simulation of a single degree of freedom, four-bar system is used as the test system. Actual parameter values of the simulation are given, followed by the essential parameters and the minimum parameters. If the mathematical model is only required to give a response to the experimental data as possible, then the minimum parameters can be used to decrease computation time and model complexity, since these minimum parameters account almost the entirety of the system's response. The drawback of this method is that although the response of the model using the minimum parameters will be very close to the experimental response, these parameters may take on non-physical values, such as negative damper values and negative masses.

These types of studies are excluded from this work, as essential parameters are linear combinations of the more specific actual parameters (such as combining both a spring constant and mass into one term, etc.) The main focus for this project are parameter identification methods that are capable of finding specific parameters of the system such as spring constants, damper constants, and masses.

3.1.4 Stochastic and Statistical Parameter Identification Methods

Studies that utilize stochastic, statistical and virtual work to solve the parameter identification problem are not considered in this paper. The work by Schiehlen et al. [11] is an example of parameter identification using correlation-based techniques. Schiehlen et al. use the residual error of the state equation as the model error. Schiehlen et al. include the consideration of both measured error from an experiment and noise from data acquisition methods. The example system used for parameter identification is a Duffing oscillator, which is a second order non-linear differential algebraic system. The four parameters to be identified are the mass of the oscillator, the damping constant of the damper on the oscillator, the spring constant of the oscillator, and the coefficient on the displacement. Discussion is included on optimum implementation of methods depending on white noise versus other types of colored noise. An advantage of this method is that coarse time responses from an experiment can be used because the estimation of values quickly converge to stable values. The correlation-based method is compared to the Gaussian least squares method for the example system.

Blanchard et al. [12], use both stochastic collocation and polynomial chaos theories to solve the problem of parameter identification. Any unknown parameter can be considered a second order random process and then treated in terms of polynomial chaos. Blanchard et. al. use a Kalman filter to determine the proper polynomial chaos expansion for the mathematical model of the system. The extend Kalman filter in [12] is optimized for use with Gaussian least squares type parameters, but can work with other types of uncertainties. The process is extensively studied by using a roll plane model of a vehicle with four degrees of freedom. The benefit of this method is that polynomial chaos methods converge much faster than the similar type Monte Carlo simulations. However, when the extend Kalman filter is used with too high of a sampling frequency for obtaining the experimental data, the solution can diverge.

3.1.5 Ordinary Differential Equations and First Order State Space Ordinary Differential Equations Methods

There is a wide variety of literature that utilize first order ODE state space equations as the main mathematical model representing the system under consideration. Parameter identification methods commonly found in literature are those which convert DAEs to ODEs by use of certain techniques, by index reduction as described in Section 2.3, Baumgarte reduction and stabilization [13], or linear graph theory and virtual work to generate symbolic equations of motion [14] [15]. All of these methods eliminate the Langrange multipliers. Some methods can utilize these second order ODEs produced by index reduction, while other methods require yet another step to form first order state-space equations i.e. $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$. Like second order ODEs, the solution to state-space equations is also well known and reviewed in the literature. However, the conversion from DAE to first order differential equation may not be possible for complex systems.

The Lie Series method [16] utilizes state-space equations. In this method, the Lie series creates a closed form analytical series solution of the ODE form of the equations of motion of the system under consideration. A forced spring-mass system with nonlinear stiffness is used as the example system. There are two main advantages of the Lie series method: the first is that the series solution can be described symbolically, which can be used to increase the efficiency of finding optimum parameters. Second, the Lie series is able to identify non-linear parameters, both independently non-linear and state dependent, which cannot be done with linear regression methods such as the Gaussian least-squares type methods.

A homotopy method [17] also requires the use of a state equation, in which a generated function is slowly morphed into the objective function by the use of a coupled gain variable. Multiple examples are provided: a linear pendulum system, a second order non-linear oscillator with two degrees of freedom, and a 14 degree of freedom vehicle model. The advantage of the homotopy method is that by slowly morphing from the generated homotopy function to the objective function, the search for the minimum spans the entire range of parameter values for the objective function for the system under consideration. Because of this, the global minimum of the objective function is always found. Classical minimization methods often incorrectly report the optimum parameters because they only are able to find local minimum.

Recall the stochastic method in Blanchard et. al. [12]. Blanchard et al. also require the use of a state space system in generation of all formulas. The Kalman filter formulation requires displacement verse time values as well as velocity verse time values, i.e. a first order ODE function. However, Blanchard et al. state that an unconstrained second order system can be reduced to the form which is required for the Kalman filter.

A disadvantage of all these methods is that they cannot be directly applied to systems which are governed by second order DAEs, so it is required to reduce the DAEs into ODEs. Equations of motion that are generated from even the most simply multibody systems are DAEs, so in general, this extra step is always required for these parameter identification processes. Furthermore, this step may be challenging for complex mechanical systems.

3.1.6 Differential Algebraic Equations Methods

Identification methods implemented in this work include only those which directly use the DAEs of motion that describe the dynamics of the system under consideration. The work done by Gerdin et al. [18] does provide a general analytical solution to the parameter identification problem for over-constrained DAEs, but includes an example of the process on an electro-mechanical system and is thus outside the scope of the focus of this project. Notation for a first order sensitivity analysis and generating consistent is presented in Serban et. al. [19], however, it is unclear how actual partial derivatives in the sensitivity equations are derived. Serban et al. [20] is cited to assist with finding partial derivatives but Serban et al. [20] has no inclusion of partial derivatives with respect to parameter vector $\boldsymbol{p}$ in its derivations, which are needed in Serban et. al. [19]. The multiple shooting method described by Gerdts [21] is a highly detailed paper covering a broad range of multibody systems, using a state grid and multiple shooing method to search for parameters to minimize the objective function. However, due to time constraints, this method was not implemented.

3.2 Parameter Identification Methods Implemented

3.2.1 Finite Differencing Approximation

As one of the most basic ways of performing parameter identification for systems governed by DAEs, a finite approximation method was used in which the gradient and Hessian were approximated numerically using forward differencing. There are numerous methods of finite differencing, all of which have different effects on convergence rate and computation time depending on the system under consideration. These are all discussed in Smith [22]. For the purposes of simplicity, forward differencing was selected.

3.2.2 First Order Sensitivity Analysis

The paper created by Ding et al. [23] was utilized in this project and Section 4.4 describes the method. Ding et al. includes a detailed derivation and formulation of both a first order sensitivity analysis and second order sensitivity analysis. Due to a large mathematical effort required for the second order sensitivity analysis, only the first order sensitivity analysis was implemented.

4 Parameter Identification Process

4.1 General Iteration Process for Methods Implemented in this Project

The objective function V is a function of both the model responses $\boldsymbol{q}_{mod}$, and the experimental responses $\boldsymbol{q}_{exp}$, which are determined by solving the DAE of motion of the system under consideration for a specific set of parameters $\boldsymbol{p}$. Therefore the general iteration process is:

$$given\,\boldsymbol{p} \rightarrow solve\,DAEs \rightarrow get\,state\,verse\,time\,responses\,\boldsymbol{q}(t) \rightarrow get\,value\,of\,V(\boldsymbol{q})$$

To change the value of V, the time responses $\boldsymbol{q}(t)$ are modified by modifying parameters $\boldsymbol{p}$ in search direction vector $\boldsymbol{d}$, and then the DAEs of motion are resolved with the new $\boldsymbol{p}$:

$$given\,\boldsymbol{p}+\boldsymbol{d} \rightarrow resolve\,DAEs \rightarrow get\,new\,state\,verse\,time\,responses\,\boldsymbol{q}_{\Delta d}(t) \rightarrow get\,value\,of\,V(\boldsymbol{q}_{\Delta d})$$

4.2 General Description of the Newton Method Applied to Parameter Identification

To determine the search direction $\boldsymbol{d}$, the Newton method is utilized, requiring a Taylor Series expansion of the objective function: the value of the objective function at the original position, then the slope of the function in direction $\boldsymbol{d}$, the second derivative of the function, and so on:

$$V(\boldsymbol{q}_{\Delta d}) = V(\boldsymbol{q}) + \frac{\partial V}{\partial d}(\boldsymbol{q})\boldsymbol{d} + \frac{1}{2}\frac{\partial^2 V}{\partial d^2}(\boldsymbol{q})\boldsymbol{d}^2 + \frac{1}{6}\frac{\partial^3 V}{\partial d^3}(\boldsymbol{q})\boldsymbol{d}^3 \dots \frac{1}{n!}\frac{1}{2}\frac{\partial^{(n)} V}{\partial d^{(n)}}(\boldsymbol{q})\boldsymbol{d}^n \tag{10}$$

Beyond a second order analysis, i.e., $n = 2$, the derivatives of the objective function are very complex due to the complex nature of mechanical systems equations of motion. Additionally, a second order sensitivity analysis has only been studied by Ding et al. [23] thus far. To find the minimum of Eq. (10), the derivative of Eq. (10) with respect to $\boldsymbol{d}$ is set to zero:

$$\frac{\partial V}{\partial d} = \frac{\partial V}{\partial d}(\boldsymbol{q}) + \frac{\partial^2 V}{\partial d^2}(\boldsymbol{q})\boldsymbol{d} = 0 \tag{11}$$

The first derivative of the objective function, $\frac{\partial V}{\partial d}(\boldsymbol{q})$, is the gradient, $\mathbf{G}$, of the objective function, and the second derivative of the objective function, $\frac{\partial^2 V}{\partial d^2}(\boldsymbol{q})$, is the Hessian, $\mathbf{H}$, of the objective function allowing Eq. (11) to be rewritten as:

$$\mathbf{G} + \boldsymbol{d}\mathbf{H} = 0 \tag{12}$$

Eq. (12) can be rearranged as:

$$\boldsymbol{d}\mathbf{H} = -\mathbf{G} \tag{13}$$

It can be noted that Eq. (13) is mathematically identical to the common linear algebraic problem:

$$\mathbf{Ax} = \mathbf{b} \tag{14}$$

Where $\mathbf{A}$ is a square $n \times n$ coefficient matrix, and solution vector $\mathbf{x}$ and right hand side coefficient vector $\mathbf{b}$ are both $1 \times n$ sized vectors. A large amount of literature has been written on this topic. Solving for $\boldsymbol{d}$, Eq. (13) can be written as:

$$\boldsymbol{d} = \mathbf{H}^{-1}(-\mathbf{G}) \tag{15}$$

Because the parameter identification process is most commonly a nonlinear process, and since Eq. (14) describes a linear system, Eq. (15) must likewise be linear. Small step approximations must be made in $\mathbf{H}$ and $\mathbf{G}$ in order to match the linear nature of Eq. (14). These approximations are described in subsections 4.3.1 and 4.3.2.

4.3 Newton Method using Finite Difference Approximations for the Gradient and Hessian

For the first approach implemented to solve the problem of parameter identification, finite difference approximations were used to approximate the values of the gradient and Hessian required for the Newton method. While this method is the most simple solution to calculating the gradient and Hessian, it is also the most basic.

4.3.1 Gradient Approximation

The gradient for the i^{th} iteration of parameters c and d is:

$$\mathbf{G}_i = \begin{bmatrix} \frac{\partial V_i}{\partial c} \\ \frac{\partial V_i}{\partial d} \end{bmatrix} \tag{16}$$

Where $\frac{\partial V}{\partial c}$ is numerical approximated using forward differencing, approximated as:

$$\frac{\partial V_i}{\partial c} \approx \frac{V(\boldsymbol{q}_{mod+\Delta c,i}) - V(\boldsymbol{q}_{mod,i})}{\Delta c} \tag{17}$$

For $V(\boldsymbol{q}_{mod+\Delta c,i})$ to be calculated, the values of $\boldsymbol{q}_{mod+\Delta c,i}$ verse time must be calculated first, found by the time solution of the DAE system when in parameter vector $\boldsymbol{p}$ the value of c has been slightly perturbed by the value of Δc:

$$\boldsymbol{p}_{mod+\Delta c,i} = \begin{bmatrix} p_{c,i} \\ p_{d,i} \end{bmatrix} + \begin{bmatrix} \Delta c \\ 0 \end{bmatrix} \tag{18}$$

The DAE system must then be re-solved to determine the new $\boldsymbol{q}_{mod,+\Delta c,i}$ vector using the slightly changed parameter vector:

$$\boldsymbol{q}_{mod+\Delta c,i} = solveDAE(\boldsymbol{p}_{mod+\Delta c,i}) \tag{19}$$

Then for each of the n states, each of which have k time responses:

$$\boldsymbol{q}_{exp} = \begin{bmatrix} \boldsymbol{q}_{1,exp} \\ \boldsymbol{q}_{2,exp} \\ \boldsymbol{q}_{3,exp} \\ \vdots \\ \boldsymbol{q}_{n,exp} \end{bmatrix} = \begin{bmatrix} q_{1,exp,1} & q_{1,exp,2} & q_{1,exp,3} & \cdots & q_{1,exp,k} \\ q_{2,exp,1} & q_{2,exp,2} & q_{2,exp,3} & \cdots & q_{2,exp,k} \\ q_{3,exp,1} & q_{3,exp,2} & q_{3,exp,3} & \cdots & q_{3,exp,k} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ q_{n,exp,1} & q_{n,exp,2} & q_{n,exp,3} & \cdots & q_{n,exp,k} \end{bmatrix}$$

And for a given i^{th} iteration in $\boldsymbol{q}_{mod}$:

$$\boldsymbol{q}_{mod,i} = \begin{bmatrix} \boldsymbol{q}_{1,mod,i} \\ \boldsymbol{q}_{2,mod,i} \\ \boldsymbol{q}_{3,mod,i} \\ \vdots \\ \boldsymbol{q}_{n,mod,i} \end{bmatrix} = \begin{bmatrix} q_{1,mod,i,1} & q_{1,mod,i,2} & q_{1,mod,i,3} & \cdots & q_{1,mod,i,k} \\ q_{2,mod,i,1} & q_{2,mod,i,2} & q_{2,mod,i,3} & \cdots & q_{2,mod,i,k} \\ q_{3,mod,i,1} & q_{3,mod,i,2} & q_{3,mod,i,3} & \cdots & q_{3,mod,i,k} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ q_{n,mod,i,1} & q_{n,mod,i,2} & q_{n,mod,i,3} & \cdots & q_{n,mod,i,k} \end{bmatrix}$$

So then:

$$\begin{aligned} V(\boldsymbol{q}_{mod+\Delta c,i}) = \tfrac{1}{2}\Big[&(q_{1exp,1} - q_{1,mod+\Delta c,i,1})^2 + (q_{1exp,2} - q_{1,mod+\Delta c,i,2})^2 \\ &+ (q_{1,exp,3} - q_{1,mod+\Delta c,i,3})^2 + ... + ((q_{1,exp,k} - q_{1,mod+\Delta c,i,k})^2 \\ &+ (q_{2,exp,1} - q_{2,mod+\Delta c,i,1})^2 + (q_{2,exp,2} - q_{2,mod+\Delta c,i,2})^2 \\ &+ (q_{2,exp,3} - q_{2,mod+\Delta c,i,3})^2 + ... + (q_{2,exp,k} - q_{2,mod+\Delta c,i,k})^2 \\ &+ (q_{n,exp,1} - q_{n,mod+\Delta c,i,1})^2 + ... + (q_{n,exp,2} - q_{n,mod+\Delta c,i,2})^2 \\ &+ (q_{n,exp,3} - q_{n,mod+\Delta c,i,3})^2 + ... + (q_{n,exp,k} - q_{n,mod+\Delta c,i,k})^2\Big] \end{aligned} \tag{20}$$

and:

$$\begin{aligned} V(\boldsymbol{q}_{mod,i}) = \tfrac{1}{2}\Big[&(q_{1exp,1} - q_{1,mod,i,1})^2 + (q_{1exp,2} - q_{1,mod,i,2})^2 \\ &+ (q_{1,exp,3} - q_{1,mod,i,3})^2 + ... + ((q_{1,exp,k} - q_{1,mod,i,k})^2 \\ &+ (q_{2,exp,1} - q_{2,mod,i,1})^2 + (q_{2,exp,2} - q_{2,mod,i,2})^2 \\ &+ (q_{2,exp,3} - q_{2,mod,i,3})^2 + ... + (q_{2,exp,k} - q_{2,mod,i,k})^2 \\ &+ (q_{n,exp,1} - q_{n,mod,i,1})^2 + ... + (q_{n,exp,2} - q_{n,mod,i,2})^2 \\ &+ (q_{n,exp,3} - q_{n,mod,i,3})^2 + ... + (q_{n,exp,k} - q_{n,mod,i,k})^2\Big] \end{aligned} \tag{21}$$

The same is done for $\frac{\partial V_{q_i}}{\partial d}$, where $\boldsymbol{q}_{mod+\Delta d,i}$ is calculated instead.

4.3.2 Hessian Approximation

The Hessian matrix for the i^{th} iteration of parameter c and d is

$$\mathbf{H}_i = \begin{bmatrix} \frac{\partial^2 V_i}{\partial c \partial c} & \frac{\partial^2 V_i}{\partial c \partial d} \\ \frac{\partial^2 V_i}{\partial d \partial c} & \frac{\partial^2 V_i}{\partial d \partial d} \end{bmatrix} \tag{22}$$

Because there is no coupling of parameters in V, c and d are both independent of each other and thus both $\frac{\partial^2 V_i}{\partial c \partial d}$ and $\frac{\partial^2 V_i}{\partial d \partial c}$ are 0. The remaining partial derivative $\frac{\partial^2 V_i}{\partial c \partial c}$ can be approximated using forward differencing:

$$\frac{\partial^2 V_i}{\partial c \partial c} \approx \frac{V(\boldsymbol{q}_{mod+2\Delta c,i}) - 2V(\boldsymbol{q}_{mod+\Delta c,i}) + V(\boldsymbol{q}_{mod,i})}{(\Delta c)^2} \tag{23}$$

Where

$$\begin{aligned}V(\boldsymbol{q}_{mod+2\Delta c,i}) = \tfrac{1}{2}\Big[&(q_{1exp,1} - q_{1,mod+2\Delta c,i,1})^2 + (q_{1exp,2} - q_{1,mod+2\Delta c,i,2})^2\\ &+ (q_{1,exp,3} - q_{1,mod+2\Delta c,i,3})^2 + ... + ((q_{1,exp,k} - q_{1,mod+2\Delta c,i,k})^2\\ &+ (q_{2,exp,1} - q_{2,mod+2\Delta c,i,1})^2 + (q_{2,exp,2} - q_{2,mod+2\Delta c,i,2})^2\\ &+ (q_{2,exp,3} - q_{2,mod+2\Delta c,i,3})^2 + ... + (q_{2,exp,k} - q_{2,mod+2\Delta c,i,k})^2\\ &+ (q_{n,exp,1} - q_{n,mod+2\Delta c,i,1})^2 + ... + (q_{n,exp,2} - q_{n,mod+2\Delta c,i,2})^2\\ &+ (q_{n,exp,3} - q_{n,mod+2\Delta c,i,3})^2 + ... + (q_{n,exp,k} - q_{n,mod+2\Delta c,i,k})^2\Big]\end{aligned}$$

$V(\boldsymbol{q}_{mod+\Delta c,i})$ and $V(\boldsymbol{q}_{mod,i})$ are calculated in the same way as in the gradient. The same is done for parameter d, where $\boldsymbol{q}_{mod+2\Delta d,i}$ must be calculated.

4.3.3 Strengths of Finite Differencing:

- Easy to implement computationally
- Can be tailed to specific system depending on forward, backward, or central differencing

4.3.4 Weaknesses of Finite Differencing:

- Computationally costly since each iteration requires five calculations of the objective function: two for the gradient and three for the Hessian
- May require the use of very small discrete steps for systems that have many parameters that need to be identified

4.4 Levenberg-Marquardt Trust Region with Second Order Sensitivity Analysis

Ding et al. [23] describe a process that uses second order sensitivity to find the optimum parameters of a mechanical system. While the mathematics involved are complex, the process provides an additional degree of accuracy not present in any other parameter identification method to date. The Levenberg-Marquardt trust region method is used as an improvement to the convergence and stability of the minimization procedure presented in Section 2.2. The trust region around the i^{th} iteration of parameters $\boldsymbol{p}^{(i)}$ is:

$$\Omega^{(i)} = \{\boldsymbol{p}|\ ||\boldsymbol{p}-\boldsymbol{p}^{(i)}|| \le h^{(i)}\} \tag{24}$$

The following ratio can be used to predict the quality of the approximation:

$$r^{(i)} = \frac{\Phi(\boldsymbol{p}^{(i)}) - \Phi(\boldsymbol{p}^{(i)}+\boldsymbol{d}^{(i)})}{\varphi^{(i)}(\mathbf{0}) - \varphi^{(i)}(\boldsymbol{d}^{(i)})} \tag{25}$$

Where $\varphi^{(i)}$are the position initial conditions of the DAEs, given in [23] by the form:

$$\boldsymbol{\varphi}(\boldsymbol{q}^1,\, \boldsymbol{p},\, t^1) = \mathbf{0} \tag{26}$$

$\bar{\boldsymbol{\varphi}}$ are the velocity initial conditions:

$$\bar{\boldsymbol{\varphi}}(\dot{\boldsymbol{q}}^1,\, \boldsymbol{q}^1,\, \boldsymbol{p},\, t^1) = \mathbf{0} \tag{27}$$

Step size $\boldsymbol{d}^{(i)}$ can be found by solving:

$$\min_{b\,\epsilon\,\mathbb{R}^{n\times m}} \{\varphi^{(i)}(\boldsymbol{d}) = \Phi(\boldsymbol{p}^{(i)}) + (\mathbf{G}^{(i)})^{\mathrm{T}}\boldsymbol{d} + \frac{1}{2}\mathbf{d}^{\mathrm{T}}\mathbf{H}^{(i)}\boldsymbol{d}\} \tag{28}$$

Where $\mathbf{G}^{(i)}$ and $\mathbf{H}^{(i)}$ are the first and second derivative of the objective function, respectively.

4.4.1 First Order Sensitivity Analysis

The gradient of Eq. (3) is

$$\mathbf{G}(\boldsymbol{p}) = \nabla V(\boldsymbol{p}) = -\sum_{i=1}^{n_m}[(y_{ie} - y_i(t_i)) \cdot \nabla y_m(t_i)] \tag{29}$$

where

$$\nabla y_m(t_i) = f_{\ddot{\boldsymbol{q}}}\ddot{\boldsymbol{q}}_{\boldsymbol{b}} + f_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{\boldsymbol{b}} + f_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{b}} + f_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{\boldsymbol{b}} + f_{\boldsymbol{b}} \triangleq Df \tag{30}$$

It is evident that six variables, $\ddot{\boldsymbol{q}}_{\boldsymbol{b}}$, $\dot{\boldsymbol{q}}_{\boldsymbol{b}}$, $\boldsymbol{q}_{\boldsymbol{b}}$, $\boldsymbol{\lambda}_{\boldsymbol{b}}$, $\dot{\boldsymbol{q}}^{\mathbf{1}}_{\mathbf{b}}$, $\boldsymbol{q}^{\mathbf{1}}_{\boldsymbol{b}}$ are still unknown. The initial values $\dot{\boldsymbol{q}}^{\mathbf{1}}_{\boldsymbol{p}}$, $\boldsymbol{q}^{\mathbf{1}}_{\boldsymbol{p}}$, are found by taking the derivatives of the initial condition Eqs. (31) and (32), which are:

$$\begin{pmatrix} \mathbf{J}^1 \\ \varphi^1_q \end{pmatrix} \boldsymbol{q}^1_{\boldsymbol{p}} = -\begin{pmatrix} \mathbf{J}^1_{\boldsymbol{p}} \\ \boldsymbol{\varphi}^1_{\boldsymbol{p}} \end{pmatrix} \tag{31}$$

$$\begin{pmatrix} \mathbf{J}^1 \\ \boldsymbol{\varphi}^1_{\dot{\boldsymbol{q}}} \end{pmatrix} \dot{\boldsymbol{q}}^1_{\boldsymbol{p}} = -\begin{pmatrix} \dot{\boldsymbol{\Phi}}^1_{\boldsymbol{p}} \\ \bar{\boldsymbol{\varphi}}^1_{\boldsymbol{p}} \end{pmatrix} - \begin{pmatrix} \dot{\boldsymbol{\Phi}}_{q^1} \\ \bar{\boldsymbol{\varphi}}^1_{\boldsymbol{q}^1} \end{pmatrix} \boldsymbol{q}^1_{\boldsymbol{p}} \tag{32}$$

Then to find the acceleration, velocity, position vectors, and Lagrange multiplier sensitivity variables $\ddot{\boldsymbol{q}}_{\boldsymbol{b}}$, $\dot{\boldsymbol{q}}_{\boldsymbol{b}}$, $\boldsymbol{q}_{\boldsymbol{b}}$, $\boldsymbol{\lambda}_{\boldsymbol{b}}$ the following matrix equation can be solved:

$$\begin{pmatrix} \mathbf{M} & \mathbf{J}^{\mathrm{T}} \\ \mathbf{J} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{q}}_{\boldsymbol{p}} \\ \boldsymbol{\lambda}_{\boldsymbol{p}} \end{pmatrix} = -\begin{pmatrix} -\mathbf{F}_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{\boldsymbol{p}} + \tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + \tilde{\boldsymbol{\Pi}}_{\boldsymbol{p}} \\ 2[(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}} + \mathbf{J}_t + \alpha\mathbf{J}]\dot{q}_{\boldsymbol{p}} + \bar{\mathbf{J}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + \bar{\mathbf{J}}_{\boldsymbol{p}} \end{pmatrix} \tag{33}$$

$\tilde{\mathbf{\Pi}}$ is defined as:

$$\tilde{\boldsymbol{\Pi}} \triangleq \mathbf{M}\tilde{\ddot{\boldsymbol{q}}} + \mathbf{J}\tilde{\boldsymbol{\lambda}} - \boldsymbol{Q} \tag{34}$$

and $\bar{\mathbf{J}}$ is defined as:

$$\bar{\mathbf{J}} \triangleq \mathbf{J}\ddot{\boldsymbol{q}} + [(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}} + 2\mathbf{J}_t + 2\alpha\mathbf{J}]\dot{\boldsymbol{q}} + \mathbf{J}_{tt} + 2\alpha\mathbf{J}_t + \beta^2\mathbf{J} \tag{35}$$

All unknown variables have been calculated and Eq. (29) can be evaluated to find the value of the first derivative of the objective function.

4.4.2 Second Order Sensitivity Analysis

The Hessian of Eq. (3) is

$$\mathbf{H}(\boldsymbol{p}) = \nabla^2 V(\boldsymbol{p}) = \sum_{i=1}^{n_m}(\nabla y_m(t_i))^{\mathrm{T}}\nabla y_m(t_i) - \sum_{i=1}^{n_m}[(y_i - y_m(t_i)) \cdot \nabla^2 y_m(t_i)] \tag{36}$$

Where

$$\nabla^2 y_m(t_i) = f_{\ddot{\boldsymbol{q}}}\ddot{\boldsymbol{q}}_{pp} + f_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{pp} + f_{\boldsymbol{q}}\boldsymbol{q}_{pp} + f_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{pp} + Df_{,\ddot{\boldsymbol{q}}}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} + Df_{,\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{\boldsymbol{p}}$$

$$+Df_{,\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + Df_{,\boldsymbol{\lambda}}\boldsymbol{\lambda}_{\boldsymbol{p}} + Df_{,\boldsymbol{p}} \triangleq f_{\ddot{\boldsymbol{q}}}\ddot{\boldsymbol{q}}_{pp} + f_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{pp} + f_{\boldsymbol{q}}\boldsymbol{q}_{pp} + f_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{pp} + Df_{\boldsymbol{p}} \tag{37}$$

A small time step size ranging from t^1 and t^2 for integration is considered. The adjoint variable method in combination with integration by parts is used to evaluate the Hessian of Eq. (36), ultimately becoming:

$$\mathbf{H}(\boldsymbol{p}) = -\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{DE}_{4,\ddot{\boldsymbol{q}}^1}\ddot{\boldsymbol{q}}^1_{\boldsymbol{p}} - (\zeta^{1\mathrm{T}}\boldsymbol{DE}^1_{2,\dot{\boldsymbol{q}}^1} + \alpha^{\mathrm{T}}\boldsymbol{DE}_{3,\dot{\boldsymbol{q}}^1} + \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{DE}_{4,\dot{\boldsymbol{q}}^1})\dot{\boldsymbol{q}}^1_{\boldsymbol{p}}$$

$$-\zeta^{2T}\boldsymbol{DE}^2_{2,\dot{\boldsymbol{q}}^2}\dot{\boldsymbol{q}}^2_{\boldsymbol{p}} - (\boldsymbol{\eta}^{1\mathrm{T}}\boldsymbol{DE}^1_{1,\boldsymbol{q}^1} + \zeta^{1\mathrm{T}}\boldsymbol{DE}^1_{2,\boldsymbol{q}^1} + \alpha^{\mathrm{T}}\boldsymbol{DE}_{3,\boldsymbol{q}^1} + \beta^{\mathrm{T}}\boldsymbol{DE}_{4,\boldsymbol{q}^1})\boldsymbol{q}^1_{\boldsymbol{p}}$$

$$-(\eta^{2\mathrm{T}}\boldsymbol{DE}^2_{1,\boldsymbol{q}^2} + \zeta^{2\mathrm{T}}\boldsymbol{DE}^2_{2,\boldsymbol{q}^2})\dot{\boldsymbol{q}}^2_{\boldsymbol{p}} - \eta^{1\mathrm{T}}\boldsymbol{DE}^1_{1,\boldsymbol{p}} - \zeta^{1\mathrm{T}}\boldsymbol{DE}^1_{2,\boldsymbol{p}} - \alpha^{\mathrm{T}}\boldsymbol{DE}_{3,\boldsymbol{p}} - \beta^{\mathrm{T}}\boldsymbol{DE}_{4,\boldsymbol{p}}$$

$$-\boldsymbol{\eta}^{2T}\boldsymbol{DE}^2_{1,\boldsymbol{p}} - \zeta^{2T}\boldsymbol{DE}^2_{2,\boldsymbol{p}}$$

$$+\int_{t^1}^{t^2}[Df_{\boldsymbol{p}} - \boldsymbol{\mu}^T(\bar{\mathbf{\Pi}}_{\ddot{\boldsymbol{q}}}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} + \bar{\boldsymbol{\Pi}}_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{\boldsymbol{p}} + \bar{\boldsymbol{\Pi}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + \bar{\boldsymbol{\Pi}}_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{\boldsymbol{p}} + \bar{\boldsymbol{\Pi}}_{\boldsymbol{p}}) - \boldsymbol{v}^T(\boldsymbol{DE}_{1,\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + \boldsymbol{DE}_{1,\boldsymbol{p}})]\mathrm{d}t \tag{38}$$

Where:

$$\boldsymbol{DE_1} \triangleq \mathbf{J}\boldsymbol{q}_{\boldsymbol{p}} + \mathbf{\Phi} \tag{39}$$

$$\boldsymbol{DE_2} \triangleq \mathbf{J}\boldsymbol{q}_{\boldsymbol{p}} + \mathbf{J}\boldsymbol{q}_{\boldsymbol{p}} + \mathbf{\Phi} \tag{40}$$

$$\boldsymbol{DE_3} \triangleq \boldsymbol{\varphi}^1_{\boldsymbol{q}^1}\boldsymbol{q}^1_{\boldsymbol{p}} + \boldsymbol{\varphi}^1_{\boldsymbol{p}} \tag{41}$$

$$\boldsymbol{DE_4} \triangleq \boldsymbol{\varphi}^1_{\dot{\boldsymbol{q}}^1}\dot{\boldsymbol{q}}^1_{\boldsymbol{p}} + \bar{\boldsymbol{\varphi}}^1_{\boldsymbol{q}^1}\boldsymbol{q}^1_{\boldsymbol{p}} + \bar{\boldsymbol{\varphi}}^1_{\boldsymbol{p}} \tag{42}$$

$\bar{\mathbf{\Pi}}$ is defined as:

$$\bar{\mathbf{\Pi}} \triangleq \mathbf{M}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} + \mathbf{J}^{\mathrm{T}}\boldsymbol{\lambda}_{\boldsymbol{p}} - \mathbf{F}_{\dot{\boldsymbol{q}}}\boldsymbol{q}_{\boldsymbol{p}} + \tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + \tilde{\boldsymbol{\Pi}}_{\boldsymbol{p}} = \mathbf{0} \tag{43}$$

$\tilde{\mathbf{\Pi}}$ is defined as:

$$\tilde{\mathbf{\Pi}} \triangleq \mathbf{M}\tilde{\ddot{\boldsymbol{q}}} + \mathbf{J}^T\tilde{\boldsymbol{\lambda}} - \mathbf{F} \tag{44}$$

Notice that equations (39) - (44) are only supporting definition equations that provide condensed notation of the already complex Hessian, Eq. (38). The remaining 12 adjoint variables $\boldsymbol{\mu}$, $\boldsymbol{\mu}^1$, $\dot{\boldsymbol{\mu}}^1$, $\boldsymbol{\mu}^2$, $\dot{\boldsymbol{\mu}}^2$, $\boldsymbol{\zeta}^1$, $\boldsymbol{\zeta}^2$, η^1, η^2, $\boldsymbol{v}$, $\boldsymbol{\beta}$, and $\boldsymbol{\alpha}$ can be found through the following equations (45) - (51). $\boldsymbol{\mu}^2$ and $\boldsymbol{\zeta}^2$ can be solved from Eq. (45) and (46) at time t^1:

$$\mathbf{M}^1\boldsymbol{\mu}^1 - (\mathbf{J}^1)^{\mathrm{T}}\boldsymbol{\zeta}^1(\bar{\boldsymbol{\varphi}}^1_{\dot{\boldsymbol{q}}^1})^{\mathrm{T}}\boldsymbol{\zeta}^1 - (\boldsymbol{\varphi}^1_{\boldsymbol{q}^1})^{\mathrm{T}}\boldsymbol{\beta} = -[(y^1 - y_m(t^1))f^1_{\ddot{\boldsymbol{q}}^1}]^{\mathrm{T}} \tag{45}$$

$$\mathbf{M}^2\boldsymbol{\mu}^2 + (\mathbf{J}^2)^{\mathrm{T}}\zeta^2 = -[(y^2 - y_m(t^2))f^2_{\ddot{\boldsymbol{q}}^2}]^{\mathrm{T}} \tag{46}$$

Then, $\dot{\boldsymbol{\mu}}^2$and $\boldsymbol{\eta}^2$ can be determined through Eqs. (47) and (48) at time t^2:

$$\begin{gathered}\mathbf{M}^2\boldsymbol{\mu}^2 + (\mathbf{M}^2 + \mathbf{F}^2_{\dot{\boldsymbol{q}}^2})^{\mathrm{T}}\boldsymbol{\mu}^2 + (\mathbf{J}^2_{\boldsymbol{q}^2})^{\mathrm{T}}\boldsymbol{\eta}^2 \\ +(\dot{\mathbf{J}}^2_{\boldsymbol{q}^2})^{\mathrm{T}}\zeta^2 = -(y^2 - y_i(t^2))(-f^2_{q^1} + \tfrac{d}{dt^2}f^2_{q^2})^{\mathrm{T}}\end{gathered} \tag{47}$$

$$\mathbf{J}\boldsymbol{\mu} = -(y_{ie} - y_i(t))f^{\mathrm{T}}_{\lambda} \tag{48}$$

Backward integration of Eq. (49) with initial conditions of $\dot{\boldsymbol{\mu}}^2$ and $\boldsymbol{\eta}^2$ yields $\boldsymbol{\mu}$ and $\boldsymbol{v}$ and hence $\dot{\boldsymbol{\mu}}$ and $\dot{\boldsymbol{\mu}}$:

$$\begin{gathered}\mathbf{M}\ddot{\boldsymbol{\mu}} + (2\dot{\mathbf{M}} + \mathbf{F}^{\mathrm{T}}_{\dot{\boldsymbol{q}}})\dot{\boldsymbol{\mu}} + (\ddot{\mathbf{M}} + \tfrac{d}{dt}\mathbf{F}_{\dot{\boldsymbol{q}}} + \boldsymbol{\Pi}_{\boldsymbol{q}})^{\mathrm{T}}\boldsymbol{\mu} \\ +\mathbf{J}^{\mathrm{T}}v = -(y_{ie} - y_i(t))(f_{\boldsymbol{q}} - \tfrac{d}{dt}f_{\dot{\boldsymbol{q}}} + \tfrac{d^2}{dt^2}f_{\ddot{\boldsymbol{q}}})^{\mathrm{T}}\end{gathered} \tag{49}$$

Finally, the variables $\boldsymbol{\zeta}^1$, $\boldsymbol{\beta}$, $\boldsymbol{\eta}^1$, and $\boldsymbol{\alpha}$ can be solved for from Eqs. (50) and (51):

$$\mathbf{M}^1\boldsymbol{\mu}^1 - (\mathbf{J}^1_{q^1})^{\mathrm{T}}\boldsymbol{\zeta}^1 - (\bar{\boldsymbol{\varphi}}^1_{\dot{\boldsymbol{q}}^1})^{\mathrm{T}}\boldsymbol{\beta} = -[(y^1_{ie} - y_i(t^1))f^1_{\ddot{\boldsymbol{q}}^1}]^{\mathrm{T}} \tag{50}$$

$$\mathbf{M}^2\boldsymbol{\mu}^2 + (\mathbf{J}^2_{q^2})^{\mathrm{T}}\boldsymbol{\zeta}^2 = -[(y^2_{ie} - y_i(t^2))f^2_{\ddot{\boldsymbol{q}}^2}]^{\mathrm{T}} \tag{51}$$

All adjoint variables have been found and the second derivative of the objective function can be found from Eq. (36). A more detailed derivation of these equations can be found in [23].

4.4.3 Levenberg-Marquardt Trust Region Algorithm

The algorithm to find the optimum parameters $\boldsymbol{p}$ using the Levenberg-Marquardt trust region algorithm with a second order sensitivity analysis is defined below.

Second Order Sensitivity Analysis Algorithm:

```
Given:
Initial parameter vector: $\boldsymbol{p}^{(1)}$
Initial step size: $\sigma^{(1)} > 1$
Tolerance: $\varepsilon > 0$
Constant: $c_2 < 1$
Constant: $c_1 < c_2$ and $\geq 0$
Initial iteration: $i = 1$

*ComputeHk*
If $\boldsymbol{H}^{(i)} + \sigma^{(i}\boldsymbol{I}$ is positive definite
    *ComputeDk*
    Compute $\boldsymbol{d}^{(i)} = -[\mathbf{H}^{(i)} + \sigma^{(i)}\boldsymbol{I}]^{-1}\mathbf{G}^{(i)}$
    If $||\boldsymbol{d}^{(i)} \leq \varepsilon||$
        Optimum solution can be returned as $\boldsymbol{p}^{(i)}$
    Else
```

```
            Compute $r^{(i)}$from Eq. (25)
            Choose $\sigma^{(i+1)}$ as:
                $\sigma^{(k+1)} = \left\{ \begin{array}{cc} 4\sigma^{(i)} & r^{(i)} < c_1 \\ \frac{1}{2}\sigma^{(i)} & r^{(i)} > c_2 \\ \sigma^{(i)} & \text{otherwise} \end{array} \right\}$
            If $r^{(i)} \leq 00$
                Set $\boldsymbol{p}^{(i+1)} = \boldsymbol{p}^{(i)}$
                k = k + 1
                Return to ComputeDk
            Else
                Set $\boldsymbol{p}^{(i+1)} = \boldsymbol{p}^{(i)} + \boldsymbol{d}^{(i)}$
                i = i + 1
                Return to ComputeHk
Else
    While $\mathbf{H}^{(i)} + \sigma^{(i)}\mathbf{I}$ is not positive
    Update: $\sigma^{(i)} = 4\sigma^{(i)}$
```

4.4.4 Strengths of Second Order Sensitivity Analysis:

- Non-linear searching and faster convergence rate lead to shorter computation time
- Efficient even when applied to very large scale systems
- Applicable to complex, second order systems with Langrange multipliers

4.4.5 Weaknesses of Second Order Sensitivity Analysis:

- Requires an unconstrained system; constrained systems must be converted into unconstrained by the augmented Langrange method
- Further studies of removing experimental noise with second order sensitivities need to be further developed

5 Derivation of Equations of Motion for a Coupled Spring-Damper Pendulum System

The system to which the parameter identification processes will be applied to is shown in Figure 1:

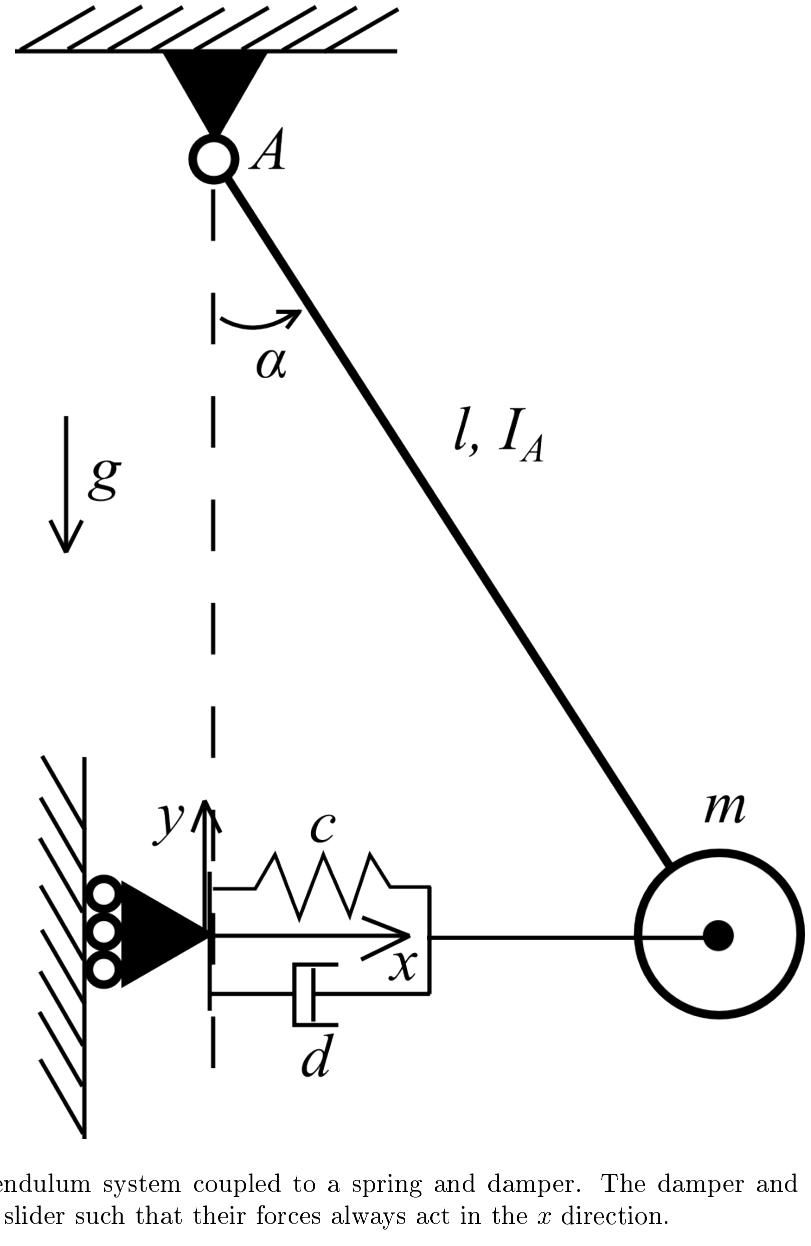

Figure 1– A pendulum system coupled to a spring and damper. The damper and spring move on a frictionless slider such that their forces always act in the x direction.

The parameters to be identified are the spring constant, c, and the damper constant, d. Their values are unknown. It is assumed that the length of the rigid rod l, the mass of the ball m, the acceleration of gravity g, and the first moment of inertia I_A are known. I_A is the first moment of inertia of the mass at distance l about point A. States α and y are the generalized states selected for the parameter identification process. Although this is somewhat of an artificial example of a system, it properly fulfills enough non-linearity while keeping fairly basic equations to easily follow the implementation process of the two parameter identification methods.

5.1 Determining Differential Algebraic Equations of Pendulum System

5.1.1 Lagrangian Formula

To find the DAEs of the system, the Lagrangian is used:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial T}{\partial \dot{\boldsymbol{q}}_i}\right) - \frac{\partial T}{\partial \boldsymbol{q}_i} + \frac{\partial V}{\partial \boldsymbol{q}_i} + \frac{\partial \Phi}{\partial \boldsymbol{q}_i}\lambda = \boldsymbol{Q}_i \tag{52}$$

Where T is the kinetic energy of the system, V is the potential energy of the system, and $\boldsymbol{q}_i$ and $\dot{\boldsymbol{q}}_i$ represent the generalized coordinates and generalized velocities, respectively. To provide a sufficient coupling of states and non-linearity for the pendulum system, the generalized coordinates are selected as angles α and coordinate y:

$$\boldsymbol{q}_i = \begin{bmatrix} \alpha \\ y \end{bmatrix} \tag{53}$$

The other possible coordinate selection, x, is related to α by Eq. (54):

$$x = l\sin(\alpha) \tag{54}$$

Eq. (54) eliminates the need for three state variables. The velocity $\dot{x}$ of the piston then is given by taking the time derivative of Eq. (54):

$$\dot{x} = \frac{\mathrm{d}}{\mathrm{d}t}x = l\cos(\alpha)\dot{\alpha} \tag{55}$$

The kinetic energy and potential energy equations must then be generated. The kinetic energy that exists in the system results from the both the inertial energy of the ball around point A, and the kinetic energy of the ball:

$$T = \frac{1}{2}I_A\dot{\alpha}^2 + \frac{1}{2}m\left(\dot{x}^2 + \dot{y}^2\right) \tag{56}$$

Substituting the velocity Eq. (54) into Eq. (56), the kinetic and potential energy equations are given only in terms of the generalized coordinates:

$$T = \frac{1}{2}I_A\dot{\alpha}^2 + \frac{1}{2}ml^2\cos^2(\alpha)\dot{\alpha}^2 + \frac{1}{2}m\dot{y}^2 \tag{57}$$

The only potential energy existing in the system is produced by the gravitational potential of the pendulum mass, at height y, since the origin of the system is selected as the point when the pendulum is at its lowest point:

$$V = mgy \tag{58}$$

5.1.2 Kinetic Energy Partial Derivatives

The partial derivative of the kinetic energy equation T, Eq. (57), with respect to generalized velocity $\dot{\alpha}$ is:

$$\frac{\partial T}{\partial \dot{q}_\alpha} = I_A\dot{\alpha} + ml^2\cos^2(\alpha)\dot{\alpha} \tag{59}$$

so the time derivative of Eq. (59), $\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial T}{\partial \dot{q}_\alpha}\right)$ is:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial T}{\partial \dot{q}_\alpha}\right) = I_A\ddot{\alpha} - 2ml^2\cos(\alpha)\sin(\alpha)\dot{\alpha}^2 + ml^2\cos^2(\alpha)\ddot{\alpha} \tag{60}$$

the partial derivative of the kinetic equation T, Eq. (57), with respect to generalized velocity $\dot{y}$ is:

$$\frac{\partial T}{\partial \dot{q}_y} = m\dot{y} \tag{61}$$

so the time derivative of Eq. (61), $\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial T}{\partial \dot{q}_y}\right)$ is:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial T}{\partial \dot{q}_y}\right) = m\ddot{y} \tag{62}$$

The partial derivative of the kinetic energy equation T, Eq. (57), with respect to generalized coordinate α is: $\frac{\partial T}{\partial q_\alpha}$ is:

$$\frac{\partial T}{\partial q_\alpha} = -ml^2\cos(\alpha)\sin(\alpha)\dot{\alpha}^2 \tag{63}$$

and the partial derivative of the kinetic energy equation T with respect to generalized coordinate y is: $\frac{\partial T}{\partial q_y}$ is:

$$\frac{\partial T}{\partial q_y} = 0 \tag{64}$$

5.1.3 Potential Energy Partial Derivatives

The partial derivative of the potential energy equation, Eq. (58), V with respect to generalized coordinate α is:

$$\frac{\partial V}{\partial q_\alpha} = 0 \tag{65}$$

the partial derivative of the potential energy equation, Eq. (58), V with respect to generalized coordinate y is:

$$\frac{\partial V}{\partial q_y} = mg \tag{66}$$

5.1.4 Generalized Forces

There is only one generalized force, $\boldsymbol{Q}_x$, acting always in the x direction, produced by the spring and damper:

$$\boldsymbol{Q}_x = \begin{bmatrix} -cx - d\dot{x} \\ 0 \end{bmatrix} \tag{67}$$

Substituting the position Eq. (54) and velocity Eq. (55), this force vector becomes two generalized forces:

$$Q_\alpha = -cl\sin(\alpha) - dlcos(\alpha)\dot{\alpha} \tag{68}$$

$$Q_y = 0 \tag{69}$$

5.1.5 Constraint Equation

Because of the rigid pendulum, the y component of the pendulum length and the coordinate y itself add up to the actual length of the pendulum, l, providing the constraint equation:

$$\Phi = l - l\cos(\alpha) - y = 0 \tag{70}$$

the partial derivative of constraint Eq. (70) with respect to the coordinate $\frac{\partial \Phi}{\partial q_\alpha}$ is:

$$\frac{\partial \Phi}{\partial q_\alpha} = l\sin(\alpha) \tag{71}$$

and $\frac{\partial \Phi}{\partial q_y}$ is:

$$\frac{\partial \Phi}{\partial q_y} = -1 \tag{72}$$

5.2 Differential Algebraic Equations

By properly combining Eqs. (63) and (64) , the partial derivatives of the kinetic energy equations with respect to the generalized positions, Eqs. (59) and (61), the partial derivatives of the kinetic energy equations with respect to the generalized velocities, Eqs. (65) and (66), the partial derivatives of the potential energy equation, Eqs. (68) and (69), the generalized forces, and Eqs. (71) and (72), the constraint equations, into the general Lagrangian, Eq. (52), the two Lagrangian equations are:

$$I_A\ddot{\alpha} - 2ml^2\cos(\alpha)\sin(\alpha)\dot{\alpha}^2 + ml^2\cos^2(\alpha)\ddot{\alpha} + ml^2\cos(\alpha)\sin(\alpha)\dot{\alpha}^2 + 0 + l\sin(\alpha)\lambda = -cl\sin(\alpha) - dlcos(\alpha)\dot{\alpha} \tag{73}$$

for α and:

$$m\ddot{y} - 0 + mg - 1\lambda = 0 \tag{74}$$

for y.

Recollecting $\ddot{\alpha}$ terms, and simplifying, Eq. (73) can be expressed as:

$$\left(I_A + ml^2\cos^2(\alpha)\right)\ddot{\alpha} - ml^2\cos(\alpha)\sin(\alpha)\dot{\alpha}^2 + l\sin(\alpha)\lambda = -cl\sin(\alpha) - dlcos(\alpha)\dot{\alpha} \tag{75}$$

Simplifying, Eq. (74) can be expressed as:

$$m\ddot{y} + mg - \lambda = 0 \tag{76}$$

In matrix form, Eqs. (71), (75), and (76) form the following DAE system:

$$\begin{bmatrix} I_A + ml^2\cos^2(\alpha) & 0 \\ 0 & m \end{bmatrix} \begin{bmatrix} \ddot{\alpha} \\ \ddot{y} \end{bmatrix} + \begin{bmatrix} l\sin(\alpha) \\ -1 \end{bmatrix} \lambda = \begin{bmatrix} \frac{1}{2}ml^2\sin(2\alpha)\dot{\alpha}^2 - cl\sin(\alpha) - dlcos(\alpha)\dot{\alpha} \\ -mg \end{bmatrix} \tag{77}$$

$$\Phi = l - l\cos(\alpha) - y = 0 \tag{78}$$

This format is identical to that of the general description of multibody system dynamic equations, Eqs. (1) and (2). The components are:

$$\mathbf{M} = \begin{bmatrix} I_A + ml^2\cos^2(\alpha) & 0 \\ 0 & m \end{bmatrix} \tag{79}$$

$$\ddot{\boldsymbol{q}} = \begin{bmatrix} \ddot{\alpha} \\ \ddot{y} \end{bmatrix} \tag{80}$$

$$\mathbf{J}^{\mathrm{T}} = \begin{bmatrix} l\sin(\alpha) \\ -1 \end{bmatrix} \tag{81}$$

$$\boldsymbol{Q} = \begin{bmatrix} \frac{1}{2}ml^2\sin(2\alpha)\dot{\alpha}^2 - cl\sin(\alpha) - dlcos(\alpha)\dot{\alpha} \\ -mg \end{bmatrix} \tag{82}$$

6 MATLAB Implementation of Differential Algebraic Equation

6.1 Restructuring of Differential Algebraic Equations for Computational Implementation

The second order DAE system in Eqs. (77) and (78) can be reduced to multiple first order differential equations, which is the form required by almost all computational DAE solvers. Note that this order reduction is markedly different than the index reduction process outlined in Section 2.2, in which the system can maintain second order index. The ODE for computational implementation is of the form of a series of first order differential equations and constraint equations:

$$\dot{\boldsymbol{u}} = \boldsymbol{f}(t,\, \boldsymbol{u},\, \boldsymbol{v}) \tag{83}$$

$$\boldsymbol{0} = \boldsymbol{g}(t,\, \boldsymbol{u},\, \boldsymbol{v}) \tag{84}$$

such that

$$\dot{\boldsymbol{u}} = \boldsymbol{v} \tag{85}$$

Eqs. (83) - (85) are the form described by MATLAB's ODE15 solver [24]. For the pendulum system:

$$\boldsymbol{u} = \begin{bmatrix} \alpha \\ y \end{bmatrix}, \text{ and } \boldsymbol{v} = \begin{bmatrix} \dot{\alpha} \\ \dot{y} \end{bmatrix} \tag{86}$$

So the total combined system is:

$$\begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \alpha \\ y \\ \dot{\alpha} \\ \dot{y} \end{bmatrix} \tag{87}$$

Taking the time derivative of the $\boldsymbol{u}$ and $\boldsymbol{v}$, second order index in both α and y is realized:

$$\begin{bmatrix} \dot{\boldsymbol{u}} \\ \dot{\boldsymbol{v}} \end{bmatrix} = \begin{bmatrix} \dot{u}_1 \\ \dot{u}_2 \\ \dot{v}_1 \\ \dot{v}_2 \end{bmatrix} = \begin{bmatrix} \dot{\alpha} \\ \dot{y} \\ \ddot{\alpha} \\ \ddot{y} \end{bmatrix} \tag{88}$$

This time derivation of u_1, u_2, v_1, and v_2 determines the four first order differential equations that need to be derived. The first and second equations, $\dot{u}_1$ and $\dot{u}_2$, are related directly to the third and fourth variables:

$$\dot{u}_1 = \dot{\alpha} = v_1 \tag{89}$$

$$\dot{u}_2 = \dot{y} = v_2 \tag{90}$$

By subtracting the Jacobian from the generalized force vector and premultiplying by the inverse of the mass matrix in Eq. (77), the accelerations $\ddot{\alpha}$ and $\ddot{y}$, and thus the third and fourth equations, $\dot{v}_1$ and $\dot{v}_2$, can be found:

$$\begin{gathered} \dot{v}_1 = \ddot{\alpha} = \mathrm{M}_1{}^{-1}\left[Q_1 - \mathrm{J}_1{}^T\lambda\right] \\ = \tfrac{1}{I_A + ml^2\cos^2(\alpha)}\left[\tfrac{1}{2}ml^2\sin(2\alpha)\dot{\alpha}^2 - cl\sin(\alpha) - dl\cos(\alpha)\dot{\alpha} - l\sin(\alpha)\lambda\right] \end{gathered} \tag{91}$$

$$\dot{v}_2 = \ddot{y} = \mathrm{M}_2{}^{-1}\left[Q_2 - \mathrm{J}_2{}^T\lambda\right] = \frac{1}{m}\left[-mg + \lambda\right] \tag{92}$$

In order to meet the requirements of the form in Eqs. (83) - (85), the final equation is the constraint equation, set equal to zero:

$$\Phi = l - l\cos(u_1) - u_2 = 0 \tag{93}$$

Note that by the notation in Eq. (84), there could be more constraint equations in a model of a more complex system. In total, the differentiation in Eqs. (89) - (93) yield the four differential equations:

$$\boldsymbol{f} = \begin{bmatrix} \dot{u}_1 \\ \dot{u}_2 \\ \dot{v}_1 \\ \dot{v}_2 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ \frac{1}{I_A + ml^2\cos^2(u_1)} \left[\frac{1}{2}ml^2\sin(2u_1)v_1^2 - cl\sin(u_1) - dl\cos(u_1)v_1 - l\sin(u_1)\lambda\right] \\ \frac{1}{m}\left[-mg + \lambda\right] \end{bmatrix} \tag{94}$$

and the constraint equation:

$$g = 0 = l - l\cos(u_1) - u_2 \tag{95}$$

This is a system of five equations and five unknowns: u_1, u_2, v_1, v_2, λ. These five equations are implemented in the APMoniter language, which are numerically integrated so that the responses $u_1(t)$ and $u_2(t)$, i.e., $\alpha(t)$ and $y(t)$, can be determined.

6.2 Numerical Simulation

Eqs. (94) and (95) were implemented in MATLAB [24] with use of the APMonitor Modeling Language Optimization Suite software [25]. The following variable values are held constant for each simulation:

$$\begin{gathered} m = 1 \\ l = 1 \\ I_A = 1 \\ g = 9.81 \end{gathered} \tag{96}$$

Consistent initial values for state variables are used for each simulation, representing the pendulum being dropped from the position when rod l is on the horizontal:

$$q_0 = \begin{bmatrix} \alpha_0 = \frac{\pi}{2} \\ y_0 = l \\ \alpha_0 = 0 \\ y_0 = 0 \\ \lambda_0 = 0 \end{bmatrix} \tag{97}$$

The time range for the simulation is:

$$t = [0.0,\ 0.1,\ 0.2,\ ...,\ 8.0] \tag{98}$$

Five test cases are run to determine the reliability of the DAE system. Variable $\zeta = \frac{d}{2\sqrt{mc}}$ determines the type of oscillations that will occur in the system. Because $m = 1$, $\zeta = \frac{d}{2\sqrt{c}}$. Harmonic, underdamped, critically damped, and overdamped cases are all tested.

Oscillation Test Case	**Parameter Values**	**Value of ζ**
Harmonic	$c = 0$ $d = 0$	0.00
Underdamped	$c = 2$ $d = 1$	0.35
Critically Damped	$c = 1$ $d = 2$	1.0
Overdamped	$c = 1$ $d = 3$	1.5

Table 3 - Parameter values for each oscillation case.

6.3 Simulation Results

For the first simulation, both the damping constant and spring constant are consider to be 0, leading to a harmonic oscillator response:

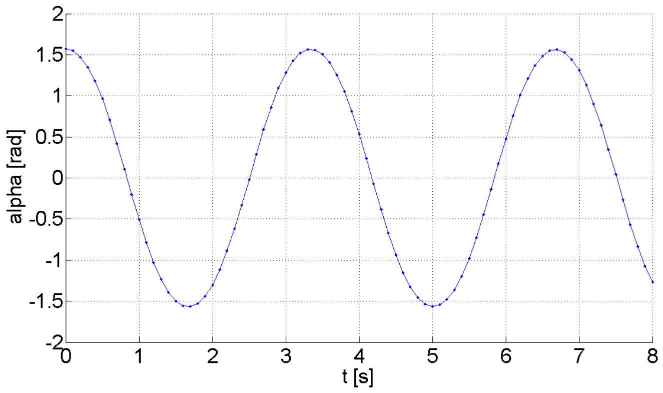

Figure 2 - Response of angle α versus time t for harmonic oscillation response.

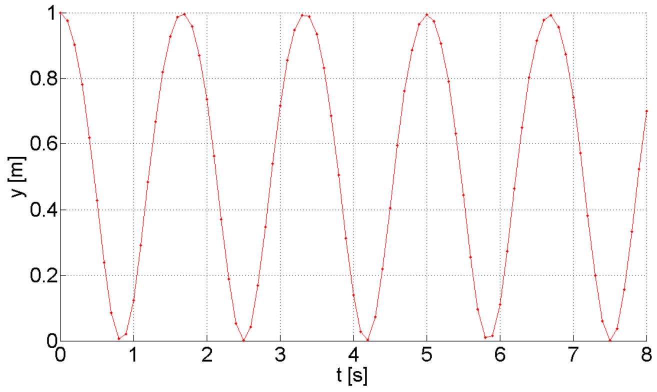

Figure 3 - Response of height y of pendulum versus time t for harmonic oscillation response.

The next simulation considers spring constant $c = 2$ and damper constant $d = 1$, leading to an underdamped response:

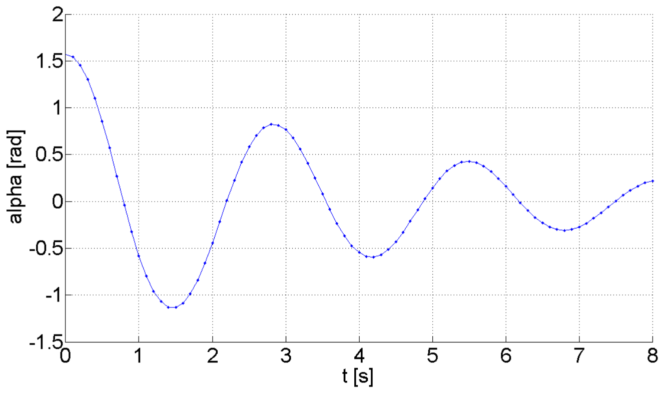

Figure 4 - Response of angle α versus time t for underdamped response.

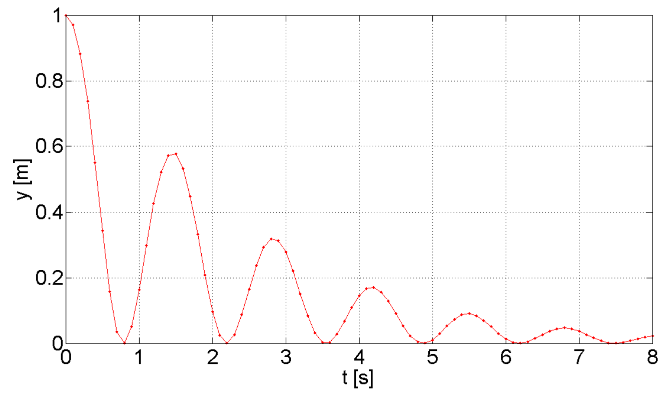

Figure 5 - Response of height y of pendulum versus time t for underdamped response.

The next simulation considers spring constant $c = 1$ and damper constant $d = 2$, leading to a critically damped response:

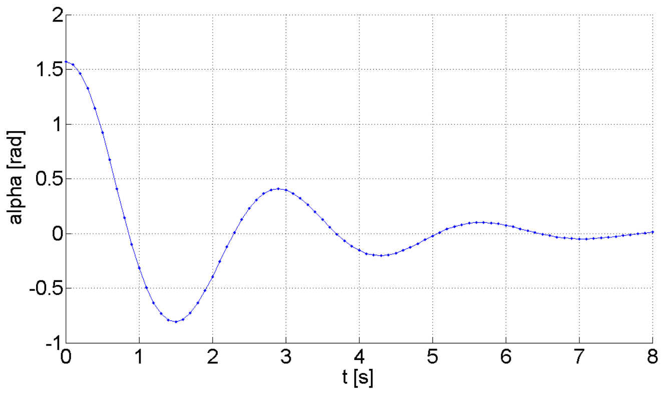

Figure 6 - Response of angle α versus time t for a critically damped response.

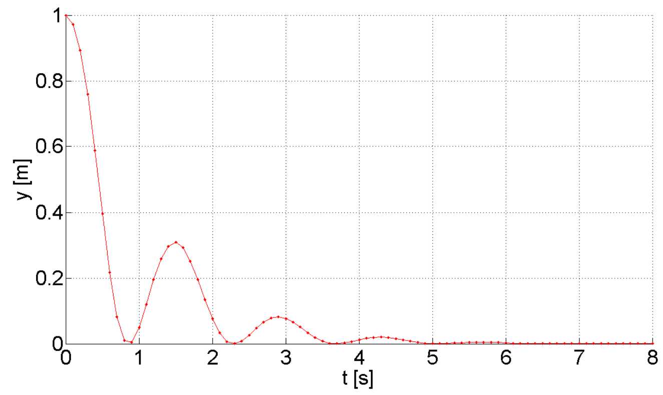

Figure 7 - Response of height y of pendulum versus time t for a critically damped response.

The final simulation considers spring constant $c = 1$ and damper constant $d = 3$, leading to a overdamped response:

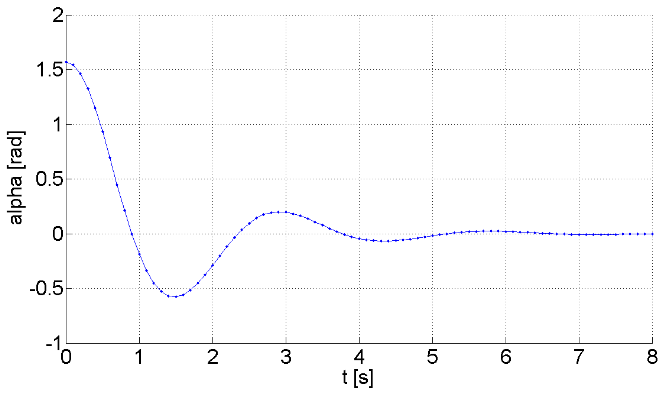

Figure 8 - Response of angle α versus time t for an overdamped response.

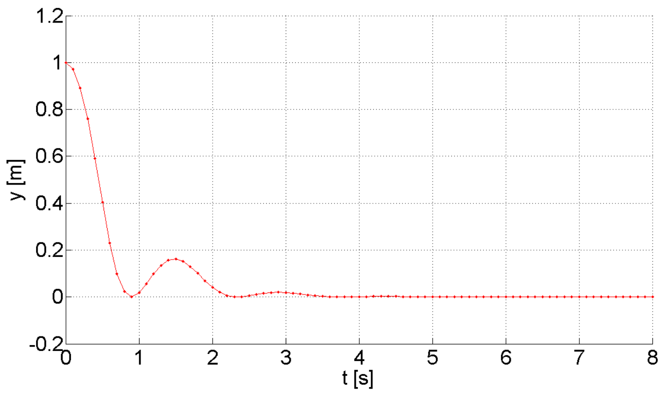

Figure 9 - Response of height y of pendulum versus time t for an overdamped response.

6.4 Properties of the Objective Function for the Pendulum System

The objective function used in parameter identification methods is of the Gaussian least-squares type, so it always has a concave parabolic shape with a single global minimum at 0. Using the system presented in Eqs. (77) and (78) as an example, c is varied from 0 to 20, with the experimental values of c and d being 10, while d is held constant at 10 Ns/m:

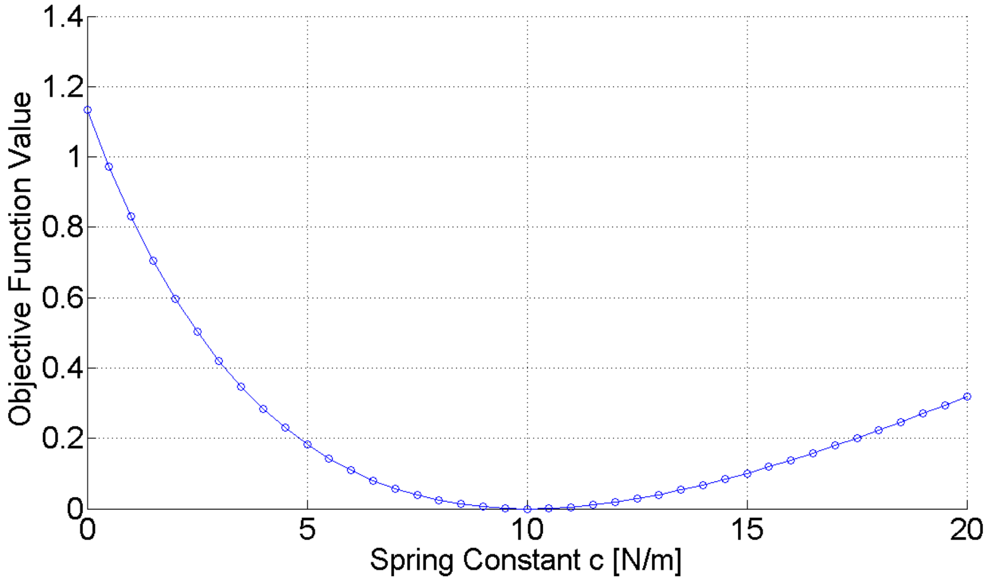

Figure 10 - Objective function values produced by varying spring constant c.

The same is done for parameter d, while c is held constant at 10 N/m:

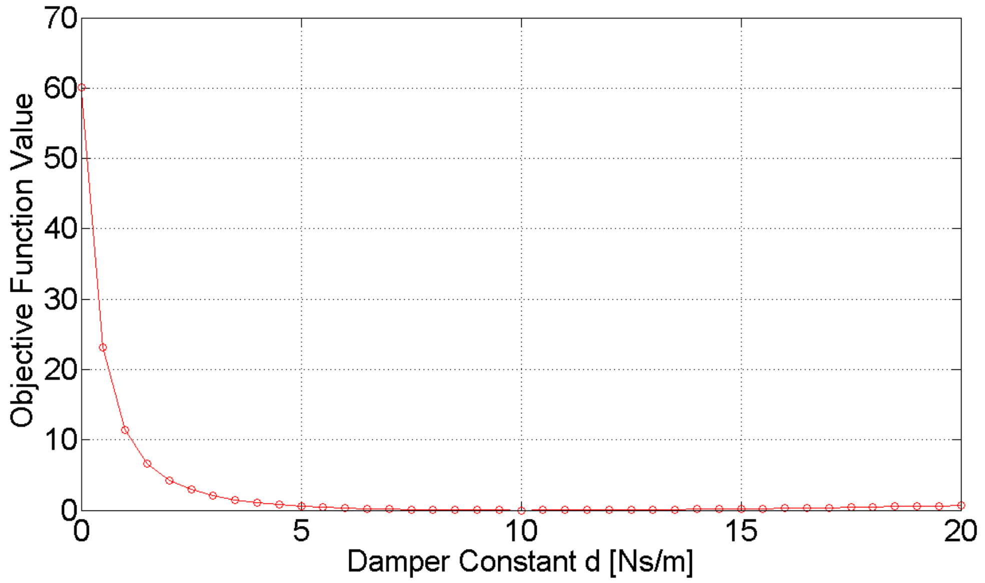

Figure 11 - Objective function values produced by varying damper constant d.

It is clear that in both cases, as c or d are modified to a value further from the experimental $c^*$and p^* values, the objective function value is > 0. Note that these curves are only accurate when the parameter not being varied is held constant at the experimental * value. It might be expected that although not zero, the objective function should still have a minimum at the * value. However, this is not the case, because parameters c and d are coupled in the system, so for example, if d was held constant at its non

* value while c was searched for, the minimum objective function value would never quite be 0, because d is still at an incorrect value, even if it is held at a constant value.

Also note that in Figure 11, changes to parameter d as the value of d becomes closer to 0 has a large impact on the value of the objective function, while around the experimental value of $d_{exp} = 10$ the objective function changes little in magnitude. In fact, beyond $d \approx 8$, any changes in c or d in this region have little effect in the response of the pendulum system, and therefore little change in the objective function value. This is because at large values of d, the system is in the overdamped region. This makes parameter identification for the pendulum system particularly difficult for parameter d, specifically because the gradient of the objective function will be small in magnitude, affecting the effectiveness of search direction $\boldsymbol{d}$.

To further understand the topology of the objective function with values of various parameters, c and d are then varied from 0 to 200 and the value of V was plotted at each parameter combination in increments of 10, as seen in Figure (12):

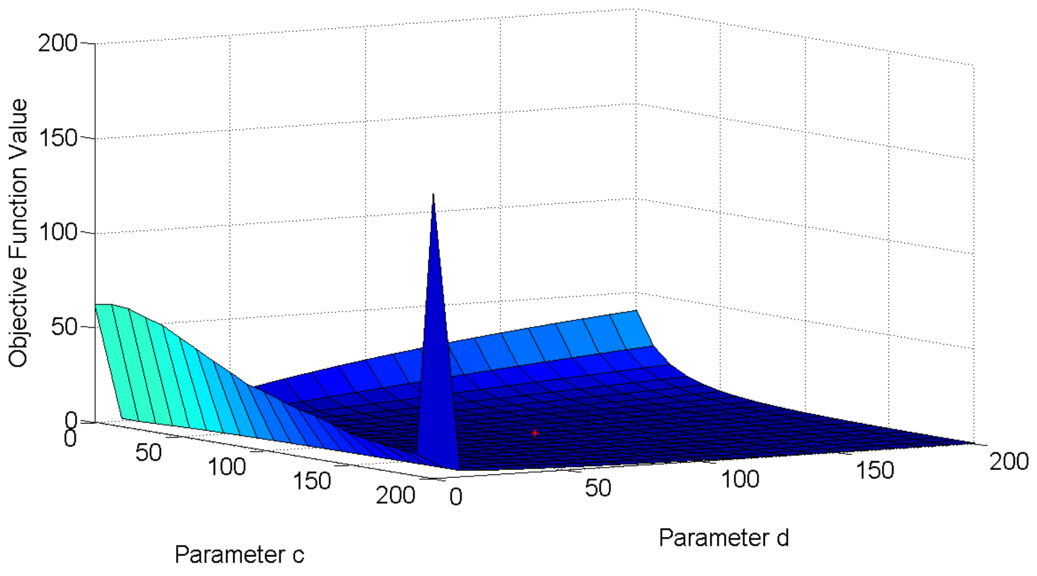

Figure 12 - Objective function value for various c and d combinations for $\boldsymbol{p}_{exp} = \begin{bmatrix} c = 100 \\ d = 100 \end{bmatrix}$.

The objective function is quite large particularly when either c or d are are close to 0. This is expected as the response of the system when either the damper or spring (or both) are completely eliminated is very different. Additionally, although the minimum of the objective function is indeed at 100 and 100, surrounding combinations of c and d produce a similar objective function value. The reason that the objective function has such a topology is the nature of the overdamped system. When d holds a large value, the system is in the overdamped region. Observe Figure 13, where two y responses from two highly overdamped cases are compared. The responses of both systems are nearly identical. Parameter identification methods for such overdamped systems are not valuable, because even a large variation of c or d in the overdamped region produces nearly the same response.

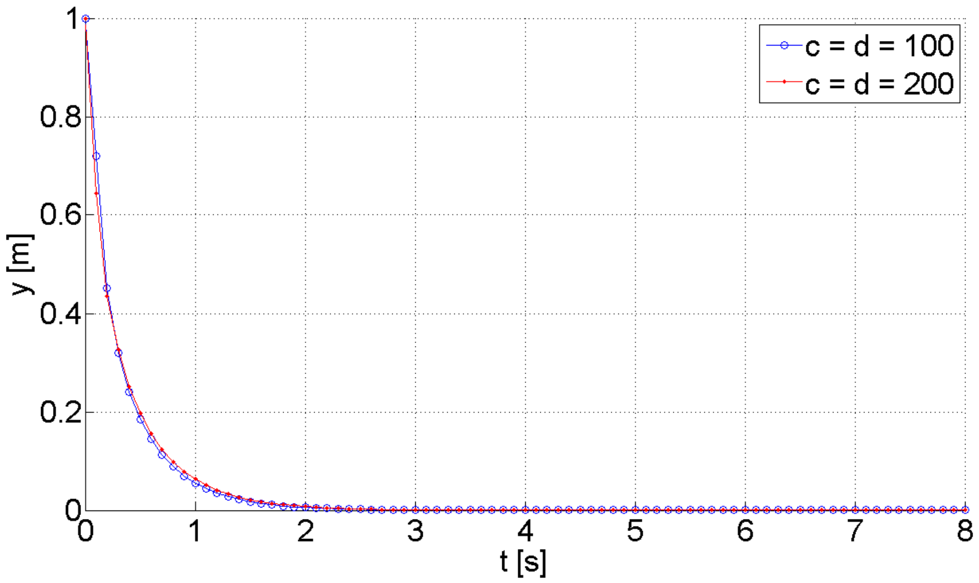

Figure 13 - y response for two overdamped cases.

Even with both parameters varied with 100% difference, the response is almost identical. In order to avoid the problem of parameter identification in the overdamped region, a more meaningful set of parameters was chosen as a reference solution to the system, valued at:

$$\boldsymbol{p}_{exp} = \begin{bmatrix} c = 5 \\ d = 1 \end{bmatrix} \tag{99}$$

This system has a much more meaningful topology:

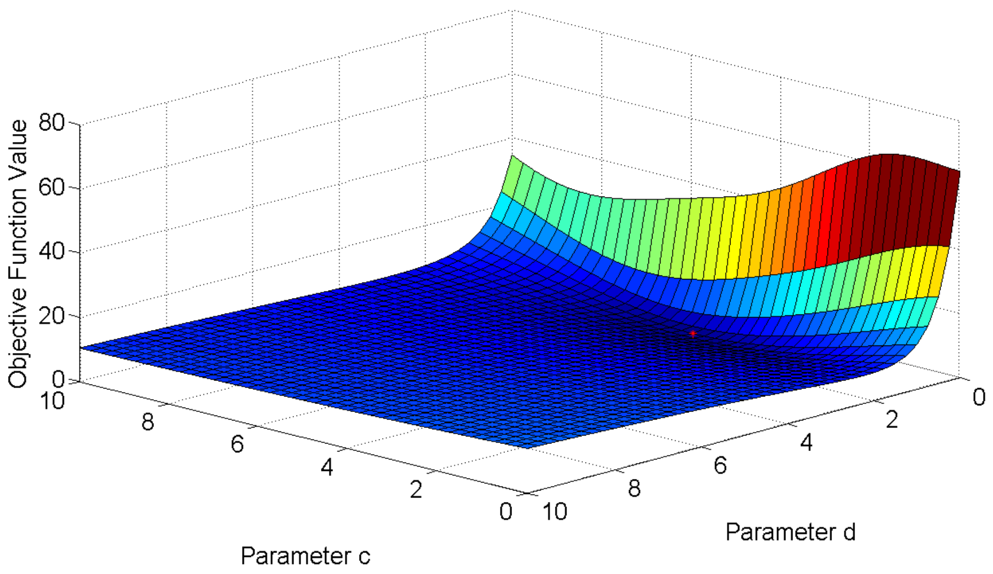

Figure 14 - Objective function values for various c and d combinations for $\boldsymbol{p}_{exp} = \begin{bmatrix} c = 5 \\ d = 1 \end{bmatrix}$.

The objective function value produced by $\boldsymbol{p}_{exp}$ is marked by a red asterisk. As expected, the objective function is zero at this value, and it is clear in all directions around the minimum sink towards the optimum parameter values.

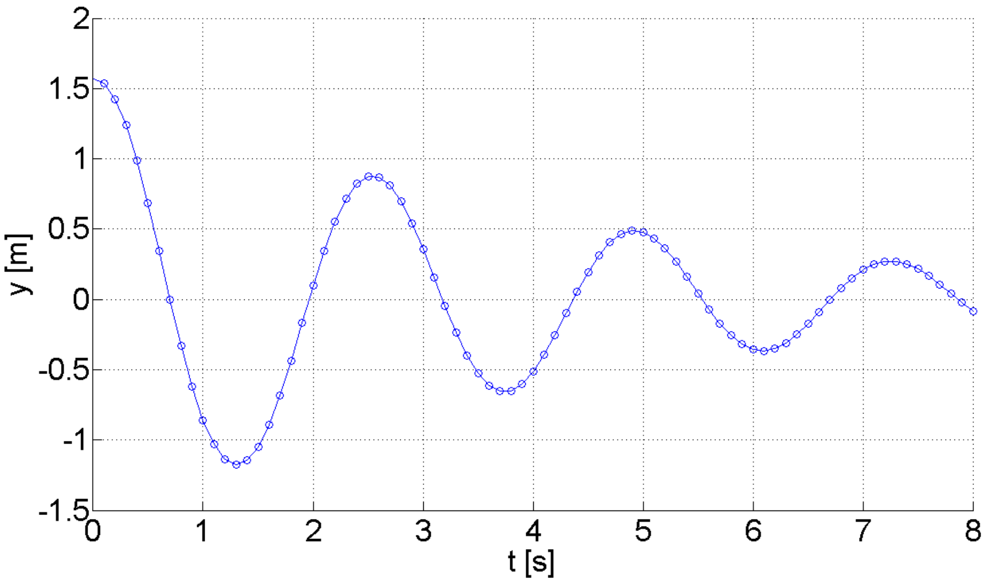

Figure 15 - y response for more meaningful parameter selection.

As can be seen in Figure 15, the response of the system is much more meaningful than the overdamped response response shown in Figure 13. This objective function topology is much more meaningful is a better test case for using the Newton method. Still, however, note that as d becomes larger, the objective function topology continues to become more and more flat, making parameter identification in the overdamped region very difficult.

6.5 Finite Differencing Newton Method Implementation

With a process of solving the DAEs that describe the pendulum system, the approximate differencing process described in Section 4.2 can now be implemented. The experimental system is described as the system response when parameters are valued at:

$$\boldsymbol{p}_{exp} = \begin{bmatrix} c_{exp} \\ d_{exp} \end{bmatrix} = \begin{bmatrix} 6 \\ 2 \end{bmatrix} \tag{100}$$

These parameters are used to generate $\boldsymbol{q}_{exp}$, the experimental response of the system. Note that the time responses of α and y in $\boldsymbol{q}_{exp}$ are held constant for the entire parameter identification process of k discrete time steps:

$$\boldsymbol{q}_{exp} = \begin{bmatrix} \alpha_{exp,0},\ \alpha_{exp,1},\ \alpha_{exp,2},\ ...,\ \alpha_{exp,k} \\ y_{exp,0},\ y_{exp,1},\ y_{exp,2},\ ...,\ y_{exp,k} \end{bmatrix} \tag{101}$$

The initial guess for the model parameters are selected as a close estimate to the actual parameters. Damper constant d is overvalued at 6.2 and spring constant c is undervalued at 1.8.

$$\boldsymbol{p}_1 = \begin{bmatrix} 6.2 \\ 1.8 \end{bmatrix} \tag{102}$$

To find corresponding coordinates versus time of the DAE system of the i^{th}iteration, starting at $i = 1$:

$$\boldsymbol{q}_{mod,i} = \begin{bmatrix} \alpha_{mod,i,0},\ \alpha_{mod,i,1},\ \alpha_{mod,i,2},\ ...,\ \alpha_{mod,i,k} \\ y_{mod,i,0},\ y_{mod,i,1},\ y_{mod,i,2},\ ...,\ y_{mod,i,k} \end{bmatrix} \tag{103}$$

the reduced order system presented in Section 6.1 is used and solved numerically using APMonitor Optimization Suite.

6.5.1 First Order Newton Method Algorithm

The Newton method algorithm is as follows:

Newton Method Algorithm

```
    Initialize
    $i = 1$
    $\epsilon = 1 \times 10^{-4}$
    $\Delta c = 0.01$
    $\Delta d = 0.01$
    $\alpha = 1$
    Calculate initial $V(\boldsymbol{q}_i)$
    While $V(\boldsymbol{q}_i) > \epsilon$
        Approximate $\mathbf{G}_i$ using $\boldsymbol{q}_i$, $\boldsymbol{q}_{\Delta c,i}$, $\boldsymbol{q}_{\Delta d,i}$
        Approximate $\mathbf{H}_i$ using $\boldsymbol{q}_i$, $\boldsymbol{q}_{\Delta c,i}$, $\boldsymbol{q}_{2\Delta c,i}$, $\boldsymbol{q}_{\Delta d,i}$, $\boldsymbol{q}_{2\Delta d,i}$
        Calculate search direction $\boldsymbol{d}_i = \mathbf{H}_i^{-1}(-\mathbf{G}_i)$
        Update parameter vector $\boldsymbol{p}_{i+1} = \boldsymbol{p}_i + \alpha \boldsymbol{d}_i$
        Check signs of parameters in $\boldsymbol{p}$ in function *checkP*
        Calculate $\boldsymbol{q}_{i+1}$ by solving DAE using new parameters $\boldsymbol{p}_{i+1}$in function *solveDAE*
        Calculate $V(\boldsymbol{q}_{i+1})$ using $\boldsymbol{q}_{i+1}$in function *getCostFunction*
        Update $i = i + 1$
    Return $\boldsymbol{p}^* = \boldsymbol{p}_i$
```

Note that this algorithm maintains that c, d, ϵ : $\mathbb{R} \geq 0$, because negative values of c and d have no physical meaning, and the solution to the DAE will not converge.

Function checkP

```
    If $\boldsymbol{p}_{i+1}(1,1) > 0$
        $\boldsymbol{p}_{i+1}(1,1) = 0$
    If $\boldsymbol{p}_{i+1}(2,1) > 0$
        $\boldsymbol{p}_{i+1}(2,1) = 0$
```

Function solveDAE

```
    Define constants:
        $c = \boldsymbol{p}(1, , i)$
        $d = \boldsymbol{p}(2, , i)$
        $m = 1$
        $I_A = 1$
        $l = 1$
        $g = 9.81$
    Set initial values of variables:
        $\alpha = \frac{\pi}{2}$
        $y = l$
        $\dot{\alpha} = 0$
        $\dot{y} = 0$
        $\lambda = 0$
    Differential and constraint equations:
        $\frac{\mathrm{d}}{\mathrm{d}t}\alpha = \dot{\alpha}$
        $\frac{\mathrm{d}}{\mathrm{d}t}y = \dot{y}$
        $\frac{\mathrm{d}}{\mathrm{d}t}\dot{\alpha} = \frac{1}{I_A + ml^2\cos^2(\alpha)}\left[\frac{1}{2}ml^2\sin(2\alpha)\dot{\alpha}^2 - cl\sin(\alpha) - dl\cos(\alpha)\dot{\alpha} - l\sin(\alpha)\lambda\right]$
        $\frac{\mathrm{d}}{\mathrm{d}t}\dot{y} = \frac{1}{m}\left[-mg + \lambda\right]$
        $0 = l - l\cos(\alpha) - y$
    Return $\boldsymbol{q} = \begin{bmatrix} [t,\ \alpha] \\ [t,\ y] \end{bmatrix}$
```

```
**Function getCostFunction**
        for $\boldsymbol{q}_i$
        for $\boldsymbol{t}_j$
            $r = q_{exp,i,t_j} - q_{mod,i,t_j}$
            $V = V + r^2$
        $V = \frac{1}{2}V$
    Return $V$
```

With $i = 38$ iterations, $\boldsymbol{p}^*$ is returned as:

$$\boldsymbol{p}^* = \begin{bmatrix} 5.99 \\ 1.99 \end{bmatrix} \tag{104}$$

Figure 12 shows the strength of the Newton searching algorithm that can simultaneously find parameters c and d:

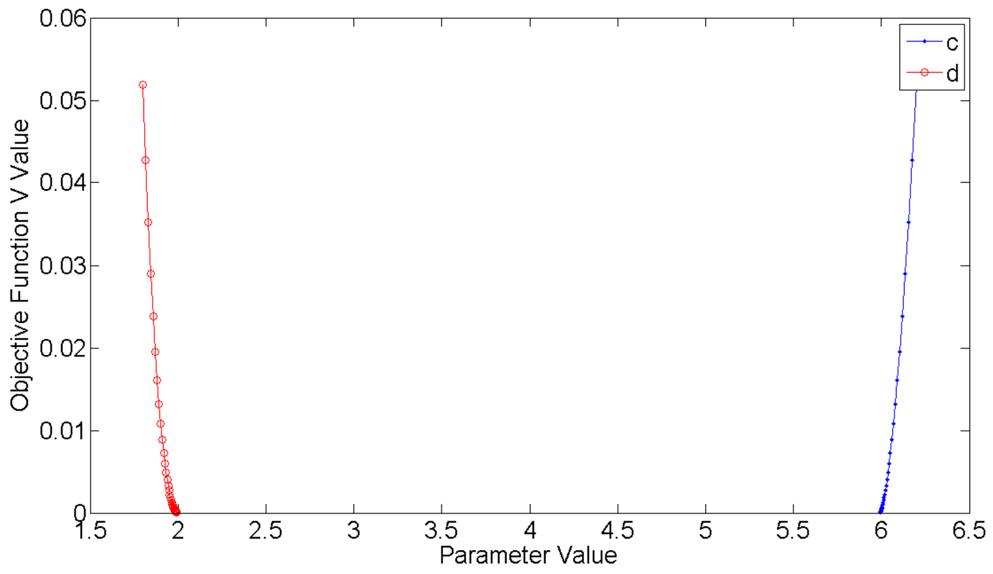

Figure 12 - Parameter values vs. the objective function value.

It is also valuable to examine how parameters c and d change with each iteration, showing that with each iteration they become closer to the experimental values.

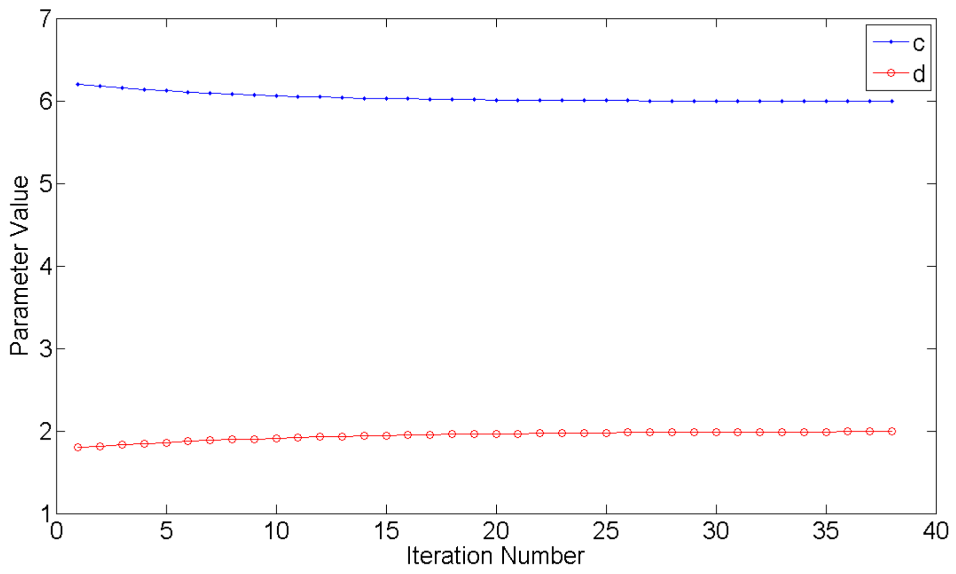

Figure 13 - Parameter values vs. iteration.

Parameters in $\boldsymbol{p}^*$ found by the algorithm provide a very accurate representation of the actual system:

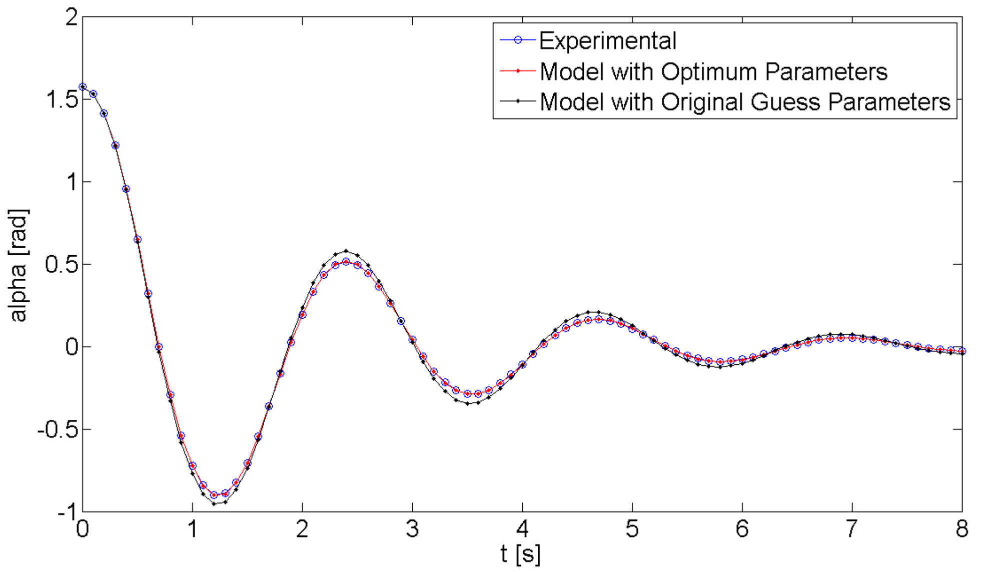

Figure 13 - Response of α from experimental and modeled system.

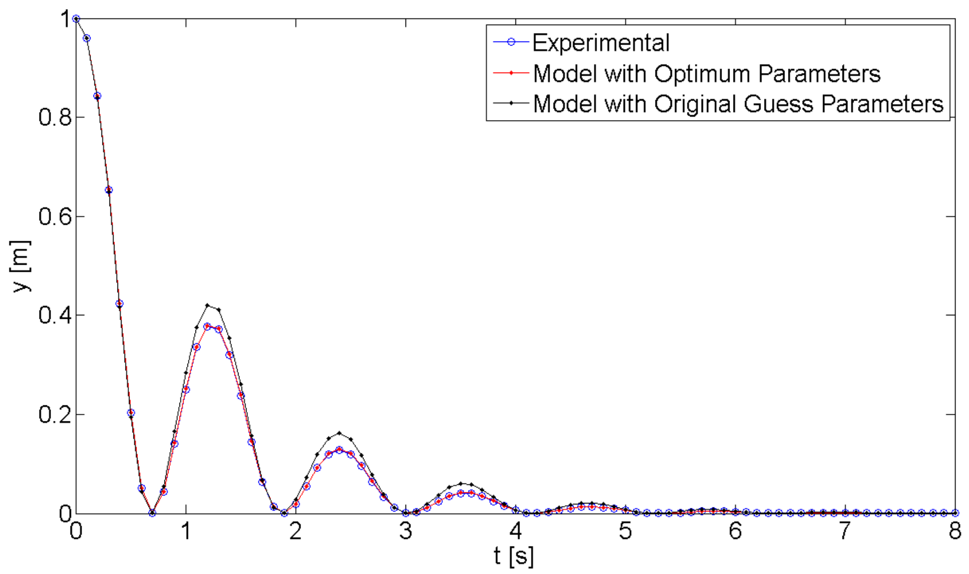

Figure 14 - Response of y from experimental and modeled system.

6.5.2 Reference Solution

As a reference solution for a worst-case identification scenario, parameters c and d are both valued initially 20, while the solution of the system is the system response when parameters c and d are valued at 2, a 900% difference from the initial guess. Using the algorithm described above, convergence requires 169 iterations to converge, returning $\boldsymbol{p}^*$ as:

$$\boldsymbol{p}^* = \begin{bmatrix} 2.0091 \\ 2.0009 \end{bmatrix} \tag{105}$$

The values of c and d at each iteration are displayed below.

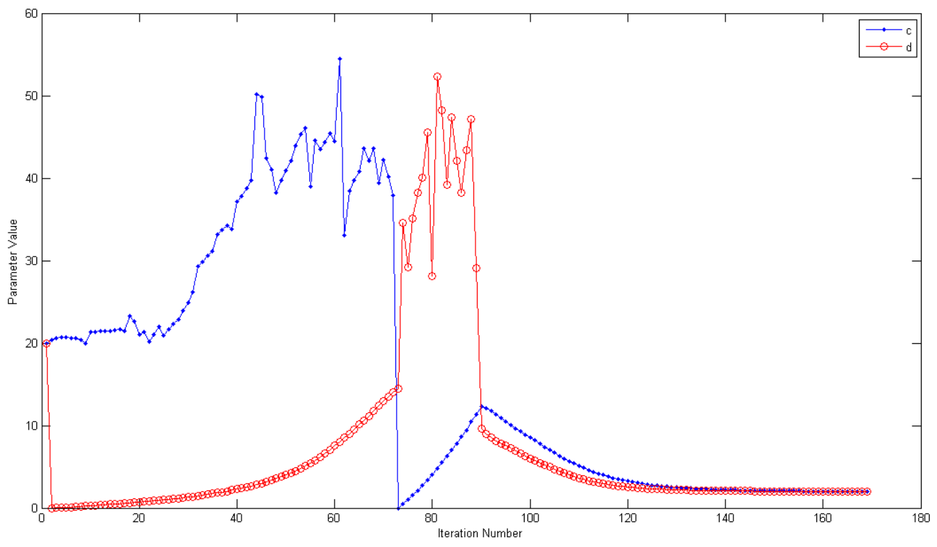

Figure 15 - Parameters c and d for the reference solution. Note that the values of the parameters remain positive or 0.

Although at first seemingly divergent, the two parameter values eventually converge to the experimental values.

6.5.3 Range of Convergence for Newton Method

To test the range of convergence, d was held constant at its experimental value, and it was found the range of convergence for c was from initial value of 0 to about 25. The same was done for the initial guess for parameter d while c was held constant.

Parameter	Range of Initial Guess for Convergence
c	0 - 25
d	1 - 25

Table 4 - Ranges of initial guess for parameters c and d.

To further verify the ability of the searching algorithm, a combination of initial starting guess for c and d was used in four test cases, where both c and d were overvalued over the experimental values, c and d both undervalued, and then the cases where one one parameter is over valued and the other is undervalued. For these four simulations, the following critical convergence values were used.

Convergence Parameter	Value
Δc,Δd	0.01
α	1
ϵ	1×10^{-4}

Table 5 - Convergence parameters critical to the convergence rate of the Newton method.

$\boldsymbol{p}_1$	$\boldsymbol{p}_{exp}$	Optimum Parameters $\boldsymbol{p}*$	Iterations	Computation Time [s]
$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 10 \\ 10 \end{bmatrix}$	$\begin{bmatrix} 9.82 \\ 9.88 \end{bmatrix}$	23	572
$\begin{bmatrix} 20 \\ 20 \end{bmatrix}$	$\begin{bmatrix} 10 \\ 10 \end{bmatrix}$	$\begin{bmatrix} 9.83 \\ 9.89 \end{bmatrix}$	26	676
$\begin{bmatrix} 0 \\ 20 \end{bmatrix}$	$\begin{bmatrix} 10 \\ 10 \end{bmatrix}$	$\begin{bmatrix} 9.83 \\ 9.88 \end{bmatrix}$	24	593
$\begin{bmatrix} 20 \\ 1 \end{bmatrix}$	$\begin{bmatrix} 10 \\ 10 \end{bmatrix}$	$\begin{bmatrix} 9.82 \\ 9.88 \end{bmatrix}$	20	488

Table 6 - Four test cases testing the full range of usability of the Newton method, starting from initial parameters $\boldsymbol{p}_1$ with experimental parameters $\boldsymbol{p}_{exp}$.

All computation times are based off the use of a Dell Latitude E6410 with an Intel Core i7 central processing unit.

6.6 First Order Sensitivity Analysis Implementation

With a method of solving the DAEs of motion for the pendulum system, the first order sensitivity analysis outlined in Section 4.4.1 can also be implemented. Because the second order component of Ding et al. [23] was not considered, the Hessian was still calculated using the finite differencing method as described in Section 4.2.2. Recall Eqs. (29) and (30) from Ding et al. [23], which describes the gradient of the objective function as:

$$\nabla V(\boldsymbol{p}) = -\sum_{i=1}^{n_m} [(y_{ie} - y_i(t_i)) \cdot \nabla y_m(t_i)]$$

with

$$\nabla y_m(t_i) = f_{\ddot{\boldsymbol{q}}}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} + f_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{\boldsymbol{p}} + f_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + f_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{\boldsymbol{p}} + f_{\boldsymbol{p}}$$

f is the function that relates the generalized measured values y_m, to the generalized coordinates of the system. In more complex systems, measured outputs may be a function of each other or have explicit time terms. In the case of the pendulum system described in Section 5, the only measured outputs are α and y. Because the two coordinates α and y are coupled in the pendulum, any change in parameters c or d will affect both outputs. Therefore, only one output needs to be selected. For simplicity, α is selected as the output function:

$$y_m(t) = f(\ddot{\alpha},\, \dot{\alpha},\, \alpha,\, \ddot{y},\, \dot{y},\, y,\, c,\, d,\, \lambda,\, t) = \alpha \tag{106}$$

Note that selecting y as the output f function, or any proper combination of α and y in f would also be possible. The remaining partial derivatives are all trivial:

$$f_{\ddot{\boldsymbol{q}}} = f_{\dot{\boldsymbol{q}}} = f_{\boldsymbol{\lambda}} = f_{\boldsymbol{p}} = 0 \tag{107}$$

the remaining $f_{\boldsymbol{q}}$ is:

$$f_{\boldsymbol{q}} = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{108}$$

therefore the $\nabla y_m(t_i)$ equation is simplified to:

$$\nabla y_m(t_i) = f_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_c \\ \boldsymbol{y}_c \end{bmatrix} = \boldsymbol{\alpha}_c \tag{109}$$

So the final gradient formula is:

$$\nabla V(\boldsymbol{p}) = -\sum_{i=1}^{n_m}[(y_{ie} - y_i(t_i)) \cdot \alpha_c(t_i)] \tag{110}$$

Therefore for calculating the gradient of the pendulum system, the only unknown required to be found is sensitivity α_c, although $\dot{\alpha}_c$ and $\ddot{\alpha}_c$ must also be solved in order to determine α_c values. To determine the α_c values, recall the matrix DAE system from Eq. (33) determined by Ding et. al.:

$$\begin{pmatrix} \mathbf{M} & \mathbf{J}^{\mathrm{T}} \\ \mathbf{J} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{q}}_{\boldsymbol{p}} \\ \boldsymbol{\lambda}_{\boldsymbol{p}} \end{pmatrix} = -\begin{pmatrix} -\boldsymbol{Q}_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{\boldsymbol{p}} + \tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + \tilde{\boldsymbol{\Pi}}_{\boldsymbol{p}} \\ 2[(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}} + \mathbf{J}_t + \alpha\mathbf{J}]\dot{\boldsymbol{q}}_{\boldsymbol{p}} + \bar{\mathbf{J}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + \bar{\mathbf{J}}_{\boldsymbol{p}} \end{pmatrix}$$

Note the subscript $\boldsymbol{p}$ on $\ddot{\boldsymbol{q}}_{\boldsymbol{p}}$, $\dot{\boldsymbol{q}}_{\boldsymbol{p}}$, and $\boldsymbol{q}_{\boldsymbol{p}}$ are different than the variables that represent the generalized coordinates: $\ddot{\boldsymbol{q}}$, $\dot{\boldsymbol{q}}$, $\boldsymbol{q}$. This means that $\boldsymbol{q}_{\boldsymbol{p}} = \begin{bmatrix} \alpha_{\boldsymbol{p}} \\ y_{\boldsymbol{p}} \end{bmatrix} = \begin{bmatrix} \alpha_c & \alpha_d \\ \alpha_c & y_d \end{bmatrix}$ are sensitivities representing the change in the responses of α and y with respect to a change in c and d at every time step of the simulation. So, $\boldsymbol{q}_{\boldsymbol{p}}$ provides the information $\frac{\partial \alpha}{\partial c}$, $\frac{\partial \alpha}{\partial d}$, $\frac{\partial y}{\partial c}$, and $\frac{\partial y}{\partial d}$ at every time step in the simulation. For the pendulum system, $\mathbf{J}_t$ is $\mathbf{0}$ since it includes no explicit time terms. and Baumgarte parameter α is valued at 0, simplifying the system to

$$\begin{pmatrix} \mathbf{M} & \mathbf{J}^{\mathrm{T}} \\ \mathbf{J} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{q}}_{\boldsymbol{p}} \\ \boldsymbol{\lambda}_{\boldsymbol{p}} \end{pmatrix} = -\begin{pmatrix} -\boldsymbol{Q}_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{\boldsymbol{p}} + \tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + \tilde{\boldsymbol{\Pi}}_{\boldsymbol{p}} \\ 2[(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}}]\dot{\boldsymbol{q}}_{\boldsymbol{p}} + \bar{\mathbf{J}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + \bar{\mathbf{J}}_{\boldsymbol{p}} \end{pmatrix} \tag{111}$$

expanding:

$$\begin{pmatrix} \mathbf{M}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} + \mathbf{J}^{\mathrm{T}}\boldsymbol{\lambda}_{\boldsymbol{p}} \\ \mathbf{J}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} \end{pmatrix} = -\begin{pmatrix} -\boldsymbol{Q}_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{\boldsymbol{p}} + \tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + \tilde{\boldsymbol{\Pi}}_{\boldsymbol{p}} \\ 2[(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}}]\dot{\boldsymbol{q}}_{\boldsymbol{p}} + \bar{\mathbf{J}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} + \bar{\mathbf{J}}_{\boldsymbol{p}} \end{pmatrix} \tag{112}$$

becoming two matrix equations:

$$\mathbf{M}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} + \mathbf{J}^{\mathrm{T}}\boldsymbol{\lambda}_{\boldsymbol{p}} = \boldsymbol{Q}_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{\boldsymbol{p}} - \tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} - \tilde{\boldsymbol{\Pi}}_{\boldsymbol{p}} \tag{113}$$

and

$$\mathbf{J}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} = -2[(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}}]\dot{\boldsymbol{q}}_{\boldsymbol{p}} - \bar{\mathbf{J}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} - \bar{\mathbf{J}}_{\boldsymbol{p}} \tag{114}$$

Parameters α_B and β_B represent Baumgarte parameters. If the constraint equation, Eq. (2) holds true, then Ding et al. [23] also find that:

$$\ddot{\mathbf{\Phi}} - 2\alpha_B\dot{\mathbf{\Phi}} - \beta_B^2\mathbf{\Phi} = 0 \tag{115}$$

is true. A rough description of the Baumgarte parameters is that they are useful for computational optimization, which is outside the scope of this project. The Baumgarte parameter values can be optimized depending on the stability of the system. For the function of the pendulum system which is already stabilized, they are both valued at 0 for simplicity.

For parameter vectors $\boldsymbol{p}$ of size greater than $n = 1$ parameters, the notation in Eq. (33) is unclear. It can be seen that for terms $\boldsymbol{Q}_{\dot{\boldsymbol{q}}}$ and $\tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}$, a vector will ultimately be produced, since $\boldsymbol{Q}_{\dot{\boldsymbol{q}}}$ and $\tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}$ are matrices and $\dot{\boldsymbol{q}}_{\boldsymbol{p}}$ and $\boldsymbol{q}_{\boldsymbol{p}}$ are vectors. For example, for i coordinates and n number of parameters, $\boldsymbol{Q}_{\dot{\boldsymbol{q}}}$ would form the $i \times n$ matrix:

$$\boldsymbol{Q}_{\dot{\boldsymbol{q}}} = \begin{bmatrix} \frac{\partial Q_{\dot{q}_1}}{\partial p_1} & \cdots & \frac{\partial Q_{\dot{q}_1}}{\partial p_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial Q_{\dot{q}_i}}{\partial p_1} & \cdots & \frac{\partial Q_{\dot{q}_i}}{\partial p_n} \end{bmatrix} \tag{116}$$

It is also unclear as to how the dimensions of $\tilde{\boldsymbol{\Pi}}_{\boldsymbol{p}}$ will be a vector, since it too is a matrix for any parameter vector greater than $n = 1$ parameters. Therefore, it is required to establish two separate versions of the DAE sensitivity system in Eq. (33), one for each parameter, c and d. These separate formulations are completed in Section 6.6.2.

6.6.1 Formulation and Partial Derivatives

From the system in Eqs. (77) and (78), the following five variables $\mathbf{M}$, $\mathbf{J}$, $\mathbf{J}^T$ can be determined from DAE system:

$$\mathbf{M} = \begin{bmatrix} I_A + ml^2cos^2(\alpha) & 0 \\ 0 & m \end{bmatrix} \tag{117}$$

$$\mathbf{J} = \begin{bmatrix} lsin(\alpha) & -1 \end{bmatrix} \tag{118}$$

$$\mathbf{J}^T = \begin{bmatrix} lsin(\alpha) \\ -1 \end{bmatrix} \tag{119}$$

The remaining unknowns $\boldsymbol{Q}_{\boldsymbol{q}}$, $\mathbf{J}_t$, $\tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}$, $\tilde{\boldsymbol{\Pi}}_{\boldsymbol{p}}$, $\bar{\mathbf{J}}_{\boldsymbol{q}}$, and $\bar{\mathbf{J}}_{\boldsymbol{p}}$ need to be generated. $\boldsymbol{Q}_{\dot{\boldsymbol{q}}}$ is

$$\boldsymbol{Q}_{\dot{\boldsymbol{q}}} = \begin{bmatrix} \frac{\partial Q_1}{\partial \dot{\alpha}} & \frac{\partial Q_1}{\partial \dot{y}} \\ \frac{\partial Q_2}{\partial \dot{\alpha}} & \frac{\partial Q_2}{\partial \dot{y}} \end{bmatrix} = \begin{bmatrix} ml^2sin(2\alpha)\dot{\alpha} - dlcos(\alpha) & 0 \\ 0 & 0 \end{bmatrix} \tag{120}$$

and $\mathbf{J}_t$ is:

$$\mathbf{J}_t = \begin{bmatrix} \frac{\partial J_1}{\partial t} & \frac{\partial J_2}{\partial t} \end{bmatrix} = \begin{bmatrix} 0 & 0 \end{bmatrix} \tag{121}$$

For $\tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}$ and $\tilde{\boldsymbol{\Pi}}_{\boldsymbol{p}}$, recall Eq. (35) for $\tilde{\mathbf{\Pi}}$:

$$\tilde{\boldsymbol{\Pi}} \triangleq \mathbf{M}\tilde{\boldsymbol{q}} + \mathbf{J}^{\mathrm{T}}\tilde{\boldsymbol{\lambda}} - \boldsymbol{Q}$$

Which for the pendulum system is:

$$\tilde{\boldsymbol{\Pi}} = \begin{bmatrix} I_A + ml^2cos^2(\alpha) & 0 \\ 0 & m \end{bmatrix} \begin{bmatrix} \ddot{\alpha} \\ \ddot{y} \end{bmatrix} + \begin{bmatrix} lsin(\alpha) \\ -1 \end{bmatrix} \lambda - \begin{bmatrix} \frac{1}{2}ml^2\sin(2\alpha)\dot{\alpha}^2 - cl\sin(\alpha) - dlcos(\alpha)\dot{\alpha} \\ -mg \end{bmatrix} \tag{122}$$

Or, simplified:

$$\tilde{\boldsymbol{\Pi}} = \begin{bmatrix} I_A\ddot{\alpha} + ml^2cos^2(\alpha)\ddot{\alpha} + lsin(\alpha)\lambda - \frac{1}{2}ml^2\sin(2\alpha)\dot{\alpha}^2 + cl\sin(\alpha) + dlcos(\alpha)\dot{\alpha} \\ m\ddot{y} - \lambda + mg \end{bmatrix} \tag{123}$$

So $\tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}$ is:

$$\tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}} = \begin{bmatrix} \frac{\partial \tilde{\Pi}_1}{\partial \alpha} & \frac{\partial \tilde{\Pi}_1}{\partial y} \\ \frac{\partial \tilde{\Pi}_2}{\partial \alpha} & \frac{\partial \tilde{\Pi}_2}{\partial y} \end{bmatrix} = \begin{bmatrix} -2ml^2sin(\alpha)cos(\alpha)\ddot{\alpha} + lcos(\alpha)\lambda - ml^2cos(2\alpha)\dot{\alpha}^2 + clcos(\alpha) - dlsin(\alpha)\dot{\alpha} & 0 \\ 0 & 0 \end{bmatrix} \tag{124}$$

Recall Eq. (35) for $\bar{\mathbf{J}}$:

$$\bar{\mathbf{J}} \triangleq \mathbf{J}\ddot{\boldsymbol{q}} + [(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}} + 2\mathbf{J}_t + 2\alpha_B\mathbf{J}]\dot{\boldsymbol{q}} + \mathbf{J}_{tt} + 2\alpha\mathbf{J}_t + \beta_B^2\mathbf{J} \tag{125}$$

However, many terms in $\bar{\mathbf{J}}$ are trivial. The partial time derivatives $\mathbf{J}_t$ and $\mathbf{J}_{tt}$, both result in a 0 vector since the Jacobian is not explicitly dependent on time. Additionally, computation parameters α and β are valued at 0. With these simplifications, $\bar{\mathbf{J}}$ becomes:

$$\bar{\mathbf{J}} = \mathbf{J}\ddot{\boldsymbol{q}} + [(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}}]\dot{\boldsymbol{q}} \tag{126}$$

With the first term $\mathbf{J}\ddot{\boldsymbol{q}}$:

$$\mathbf{J}\ddot{\boldsymbol{q}} = lsin(\alpha)\ddot{\alpha} - \ddot{y} \tag{127}$$

and the second term $[(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}}]\dot{\boldsymbol{q}}$ can be determined in three steps:

$$\begin{aligned} \mathbf{J}\dot{\boldsymbol{q}} &= \;\; lsin(\alpha)\dot{\alpha} \quad -\dot{y} \\ (\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}} &= \begin{bmatrix} lcos(\alpha)\dot{\alpha} & 0 \end{bmatrix} \\ [(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}}]\dot{\boldsymbol{q}} &= lcos(\alpha)\dot{\alpha}^2 \end{aligned} \tag{128}$$

So $\bar{\mathbf{J}}$ is:

$$\bar{\mathbf{J}} = lsin(\alpha)\ddot{\alpha} - \ddot{y} + lcos(\alpha)\dot{\alpha}^2 \tag{129}$$

Note that $\bar{\mathbf{J}}$ is a scalar. $\bar{\mathbf{J}}_{\boldsymbol{q}}$ is:

$$\bar{\mathbf{J}}_{\boldsymbol{q}} = \begin{bmatrix} lcos(\alpha)\ddot{\alpha} - lsin(\alpha)\dot{\alpha}^2 & 0 \end{bmatrix}$$

6.6.2 Separate Formulation for Parameters c and d

Formulation for Spring Constant Parameter c

Due to the dimensioning problem present in Eq. (116), the the partial derivatives of $\tilde{\boldsymbol{\Pi}}$ and $\bar{\mathbf{J}}$ must be done separately with respect to each parameter to get the correct dimensions required for the DAE. Calculating these values for the two parameters simultaneously would produce a square 2×2 matrix, which does not match the vector properties in the rest of the DAE. For parameter c, $\tilde{\boldsymbol{\Pi}}$ is:

$$\tilde{\boldsymbol{\Pi}}_c = \begin{bmatrix} \frac{\partial \tilde{\Pi}_1}{\partial c} \\ \frac{\partial \tilde{\Pi}_2}{\partial c} \end{bmatrix} = \begin{bmatrix} lsin(\alpha) \\ 0 \end{bmatrix} \tag{130}$$

Scalar variables $\bar{\mathrm{J}}_c$ and $\bar{\mathrm{J}}_d$ must also be determined. $\bar{\mathrm{J}}_c$ is a scalar of value 0, since there is no dependance in $\bar{\mathrm{J}}$, Eq. (129) on parameter c:

$$\bar{\mathrm{J}}_c = 0$$

The DAE system in Eq. (33) can now be generated for parameter c. For parameters c and d the equations of motion differ by only one term due to the difference between $\tilde{\boldsymbol{\Pi}}_c$ and $\tilde{\boldsymbol{\Pi}}_d$. For parameter c, three equations are derived. Two differential equations are derived from the first line of the DAE system, which is:

$$\mathbf{M}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} + \mathbf{J}^{\mathrm{T}}\boldsymbol{\lambda}_{\boldsymbol{p}} = \boldsymbol{Q}_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{\boldsymbol{p}} - \tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} - \tilde{\boldsymbol{\Pi}}_{c} \tag{131}$$

Expanded into two scalar equations:

$$\mathrm{M}_1\ddot{q}_{p_\alpha} + {\mathrm{J}_1}^{\mathrm{T}}\lambda_{p_\alpha} = Q_{\dot{q}_1}\dot{q}_{p_\alpha} - \tilde{\Pi}_{q_1}q_{p_\alpha} - \tilde{\Pi}_{c_1} \tag{132}$$

$$\mathrm{M}_2\ddot{q}_{p_y} + {\mathrm{J}_2}^{\mathrm{T}}\lambda_{p_y} = Q_{\dot{q}_2}\dot{q}_{p_y} - \tilde{\Pi}_{q_2}q_{p_y} - \tilde{\Pi}_{c_2} \tag{133}$$

With all coefficients:

$$\begin{gathered}\left[I_A + ml^2cos^2(\alpha)\right]\ddot{\alpha}_c + lsin(\alpha)\lambda_c = \left[ml^2sin(2\alpha)\dot{\alpha} - dlcos(\alpha)\right]\dot{\alpha}_c \\ -\left[-2ml^2sin(\alpha)cos(\alpha)\ddot{\alpha} + lcos(\alpha)\lambda - ml^2cos(2\alpha)\dot{\alpha}^2 + clcos(\alpha) - dlsin(\alpha)\dot{\alpha}\right]\alpha_c - lsin(\alpha)\end{gathered} \tag{134}$$

$$m\ddot{y}_c - \lambda_c = 0 \tag{135}$$

and one constraint equation equation is derived from the second line:

$$\mathbf{J}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} = -2[(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}}]\dot{\boldsymbol{q}}_{\boldsymbol{p}} - \bar{\mathbf{J}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} - \bar{\mathbf{J}}_c \tag{136}$$

$$lsin(\alpha)\ddot{\alpha}_c - \ddot{y}_c = \left[-2lcos(\alpha)\dot{\alpha}\right]\dot{\alpha}_c - \left[lcos(\alpha)\ddot{\alpha} - lsin(\alpha)\dot{\alpha}^2\right]\alpha_c - 0 \tag{137}$$

Formulation for Damping Constant Parameter d
For parameter d, $\tilde{\boldsymbol{\Pi}}$ is:

$$\tilde{\boldsymbol{\Pi}}_d = \begin{bmatrix} \frac{\partial \tilde{\Pi}_1}{\partial d} \\ \frac{\partial \tilde{\Pi}_2}{\partial d} \end{bmatrix} = \begin{bmatrix} lcos(\alpha)\dot{\alpha} \\ 0 \end{bmatrix} \tag{138}$$

Like $\bar{\mathbf{J}}_c$, $\bar{\mathbf{J}}_d$ is also a scalar of value 0, since there is no dependance in $\bar{\mathbf{J}}$, Eq. (129) on parameter d :

$$\bar{\mathbf{J}}_d = 0 \tag{139}$$

For parameter d the two differential equations are:

$$\mathbf{M}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} + \mathbf{J}^{\mathrm{T}}\boldsymbol{\lambda}_{\boldsymbol{p}} = \boldsymbol{Q}_{\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}}_{\boldsymbol{p}} - \tilde{\boldsymbol{\Pi}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} - \tilde{\boldsymbol{\Pi}}_{d} \tag{140}$$

Because there are two responses, Eq. (140) can be expanded into two scalar equations:

$$\mathrm{M}_1\ddot{\alpha}_d + {\mathrm{J}_1}^{\mathrm{T}}\lambda_d = Q_{\dot{q}_1}\dot{\alpha}_d - \tilde{\Pi}_{q_1}\alpha_d - \tilde{\Pi}_{d_1} \tag{141}$$

$$\mathrm{M}_2\ddot{y}_d + {\mathrm{J}_2}^{\mathrm{T}}\lambda_d = Q_{\dot{q}_2}\dot{y}_d - \tilde{\Pi}_{q_2}y_d - \tilde{\Pi}_{d_2} \tag{142}$$

Fully, expanded the two equations are:

$$\begin{gathered}\left[I_A + ml^2cos^2(\alpha)\right]\ddot{\alpha}_d + lsin(\alpha)\lambda_d = \left[ml^2sin(2\alpha)\dot{\alpha} - dlcos(\alpha)\right]\dot{\alpha}_d \\ -\left[-2ml^2sin(\alpha)cos(\alpha)\ddot{\alpha} + lcos(\alpha)\lambda - ml^2cos(2\alpha)\dot{\alpha}^2 + clcos(\alpha) - dlsin(\alpha)\dot{\alpha}\right]\alpha_d - lcos(\alpha)\dot{\alpha}\end{gathered} \tag{143}$$

$$m\ddot{y}_d - \lambda_d = 0 \tag{144}$$

And the constraint equation:

$$\mathbf{J}\ddot{\boldsymbol{q}}_{\boldsymbol{p}} = -2[(\mathbf{J}\dot{\boldsymbol{q}})_{\boldsymbol{q}}]\dot{\boldsymbol{q}}_{\boldsymbol{p}} - \bar{\mathbf{J}}_{\boldsymbol{q}}\boldsymbol{q}_{\boldsymbol{p}} - \bar{\mathbf{J}}_d \tag{145}$$

Fully expanded:

$$lsin(\alpha)\ddot{\alpha}_d - \ddot{y}_d = \left[-2lcos(\alpha)\dot{\alpha}\right]\dot{\alpha}_d - \left[lcos(\alpha)\ddot{\alpha} - lsin(\alpha)\dot{\alpha}^2\right]\alpha_d - 0 \tag{146}$$

Note the only difference between the formulation of equations for parameters c and d in Sections 8.2.1 and 8.2.2 is the last term in Eq. (134) and (143), respectively. The power of this method is now apparent. Notice that the only unknowns that have to be solved are the newly introduced partial derivative variables α_c, $\dot{\alpha}_c$, $\ddot{\alpha}_c$, α_d, α_d, and α_d, while variables M_1 , J_1 , and $\tilde{\Pi}_{c_1}$ are scalar values that can be determined at every time step, assuming the original DAEs of motion in Eqs. (77) and (78) have been previously solved. Utilizing the second order sensitivity analysis present in Ding. [23], summarized in Eqs. (37) - (52), requires forward and backward integrations at multiple time steps, in addition to even more complex formulations, and this amount of technical effort is outside the scope of this project. Therefore, calculation of the Hessian is kept as a finite differencing approximation, as presented in Section 4.3.2.

6.6.3 Generation of Reduced Order System for Parameter c

For an initial test of the method presented by Ding et al. [23], parameter d was held constant at its experimental value while only c was searched for. The rewritten form of Eqs. (134) and (135) yield the four differential equations:

$$\frac{\mathrm{d}}{\mathrm{d}t}\alpha_c = \dot{\alpha}_c \tag{147}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}y_c = \dot{y}_c \tag{148}$$

$$\begin{aligned}\tfrac{\mathrm{d}}{\mathrm{d}t}\dot{\alpha}_c = \ddot{\alpha}_c = \tfrac{1}{I_A + ml^2cos^2(\alpha)}\Big[&\left[ml^2sin(2\alpha)\dot{\alpha} - dlcos(\alpha)\right]\dot{\alpha}_c \\ - \left[-2ml^2sin(\alpha)cos(\alpha)\ddot{\alpha} + lcos(\alpha)\lambda - ml^2cos(2\alpha)\dot{\alpha}^2 + clcos(\alpha) - dlsin(\alpha)\dot{\alpha}\right]&\alpha_c - lsin(\alpha) - lsin(\alpha)\lambda_c\Big]\end{aligned} \tag{149}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\dot{y}_p = \ddot{y}_p = \frac{\lambda_c}{m} \tag{150}$$

and the constraint equation from Eq. (137):

$$lsin(\alpha)\ddot{\alpha}_c - \ddot{y}_c = \left[-2lcos(\alpha)\dot{\alpha}\right]\dot{\alpha}_c - \left[lcos(\alpha)\ddot{\alpha} - lsin(\alpha)\dot{\alpha}^2\right]\alpha_c \tag{151}$$

To solve these equations, the values of α, $\dot{\alpha}$, $\ddot{\alpha}$ and λ must first be found by solving the original DAE and used properly in Eqs. (149) and (151) at each time step during solving. This means that all terms in brackets in Eqs. (149) and (151) are scalar values if the DAE Eqs. (77) and (78) have been solved.

6.6.4 Results of Sensitivity Equations

The values of α_c and y_c represent $\frac{\partial\alpha}{\partial c}$ and $\frac{\partial y}{\partial c}$ at each time step of the simulation, respectfully. A plot of these values compared to the actual response of the system provides insight as to how a change of parameter c changes the response of the system. Taking the parameters of the system to be $\boldsymbol{p}_{exp} = \begin{bmatrix} 6 \\ 2 \end{bmatrix}$, α_c and y_c were calculated, plotted below α and y, respectfully for convenience:

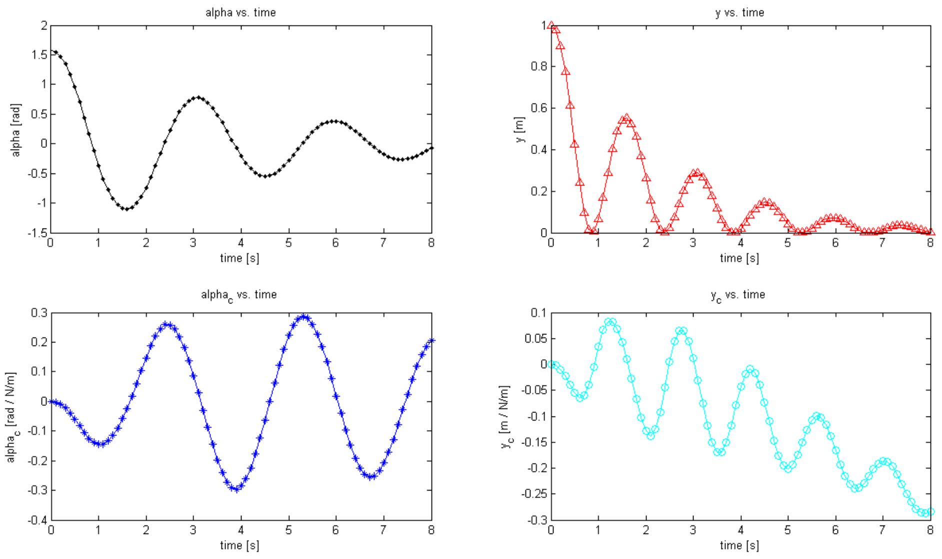

Figure 16 α_c and y_c calculated for the standard harmonic decaying case.

These charts give an excellent insight into both the pendulum system and how the sensitivity analysis works. Notice that at values of $\alpha = 0$, the sensitivity of α to parameter c, i.e., the value of α_c, is at a local maximum. This is an expected response, after observing Figure 1 displaying the schematic of the pendulum system, the spring will be under the most compressional force when angle α is at 0. It is likewise expected then that the values of y_c are at a maximum when y is at 0. To further confirm that the calculated α_c and y_c are accurate, an overdamped case of parameters valued at $\boldsymbol{p}_{exp} = \begin{bmatrix} 100 \\ 100 \end{bmatrix}$ was tested:

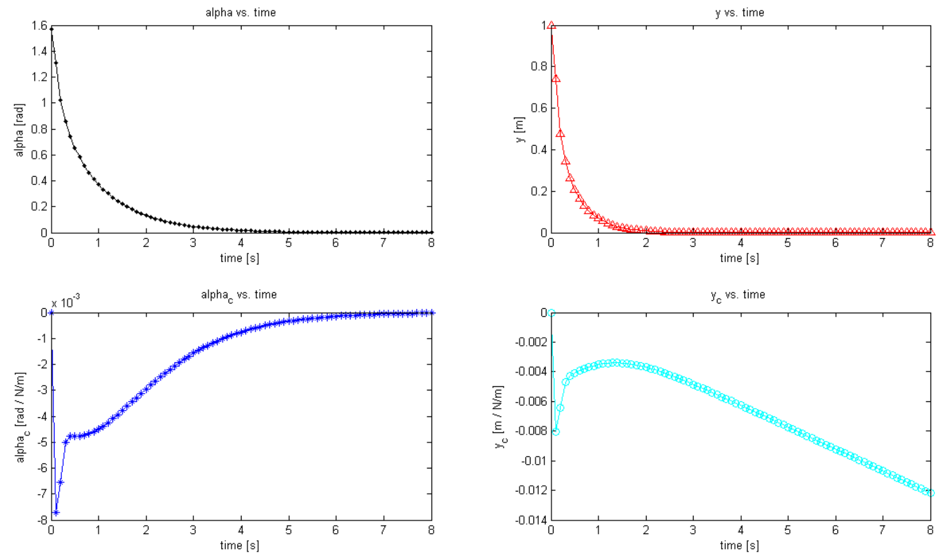

Figure 17 α_c and y_c calculated for the overdamped case.

As can be seen in the overdamped test, extremely small values of both α_c and y_c are produced. This means that any change in parameter c creates almost no change in outputs α or y. This is an expected result after exploring the overdamped response with the same parameters in the system in Section 6.4. It was observed that even a large change in c produced nearly the same response if d was very large. Therefore, from these two vastly different test cases, this method is accurate for determining the sensitivities of parameters c and d to the system state responses α and y.

6.6.5 Range of Convergence for First Order Sensitivity Analysis

Two test cases were tested in directions around $p_{exp} = \begin{bmatrix} 6 \\ 2 \end{bmatrix}$. Because the first order sensitivity analysis was not completed for parameter d, the convergence range is done only for parameter c. Two test cases are done, ranging from 1.5 to 10 in c with the experimental parameter being valued at $c_{exp} = 6$. Parameter d remains constant at $d_{exp} = d_1 = 10$.

$\boldsymbol{p}_1$	$\boldsymbol{p}_{exp}$	**Optimum Parameters $\boldsymbol{p}*$**	**Iterations**	**Computation Time [s]**
$\begin{bmatrix} 1.5 \\ 2 \end{bmatrix}$	$\begin{bmatrix} 6 \\ 2 \end{bmatrix}$	$\begin{bmatrix} 6.000 \\ 2 \end{bmatrix}$	4	88.2
$\begin{bmatrix} 10 \\ 2 \end{bmatrix}$	$\begin{bmatrix} 6 \\ 2 \end{bmatrix}$	$\begin{bmatrix} 6.001 \\ 2 \end{bmatrix}$	7	161

Table 7 - Two test cases testing the full range of usability of the first order sensitivity analysis, starting from initial parameters $\boldsymbol{p}_1$ with experimental parameters $\boldsymbol{p}_{exp}$.

As can be seen from Table 7, with an initial spring constant parameter c guess below 1.5 N/m or above 10 N/m, the solution would diverge. Because these results are slightly unfair to compare with the finite approximation method, since only parameter c is being searched for, the range of convergence results from the finite differencing range of convergence in Section 6.5.3 Table 6 cannot be used. To accommodate for this, the first order approximation method was re-tested while holding parameter d constant to make it a fair comparison with the results in Table 7.

$\boldsymbol{p}_1$	$\boldsymbol{p}_{exp}$	**Optimum Parameters $\boldsymbol{p}*$**	**Iterations**	**Computation Time [s]**
$\begin{bmatrix} 1 \\ 2 \end{bmatrix}$	$\begin{bmatrix} 6 \\ 2 \end{bmatrix}$	$\begin{bmatrix} 5.995 \\ 2 \end{bmatrix}$	5	89.0
$\begin{bmatrix} 10 \\ 2 \end{bmatrix}$	$\begin{bmatrix} 6 \\ 2 \end{bmatrix}$	$\begin{bmatrix} 5.985 \\ 2 \end{bmatrix}$	5	108

Table 8 - Two test cases testing the full range of usability of the first order sensitivity analysis, starting from initial parameters $\boldsymbol{p}_1$ with experimental parameters $\boldsymbol{p}_{exp}$.

By comparing Table 7 and Table 8, the two methods are very similar. However, the finite differencing method is able to converge when the initial guess is slightly lower than in the first order sensitivity analysis. Additionally, from the lower bound, the two methods have almost identical computation time, while from the upper bound, the finite differencing method converges in about 32% less time than the first order sensitivity analysis method.

7 Comparison of Approximation Method and First Order Sensitivity Method

7.1 Comparison Model Used

The example system to be compared is selected when experimental parameters are set at $\boldsymbol{p}_{exp} = \begin{bmatrix} 6 \\ 2 \end{bmatrix}$, and the initial guess parameters are $\boldsymbol{p}_1 = \begin{bmatrix} 6.2 \\ 1.8 \end{bmatrix}$. Convergence variables were kept identical to that as in Table 4, when the Newton method was used. The following Sections 7.1.1 - 7.2 compare the two methods in various ways.

7.1.1 Parameter c Convergence

The value of parameter c versus iteration was compared for each method.

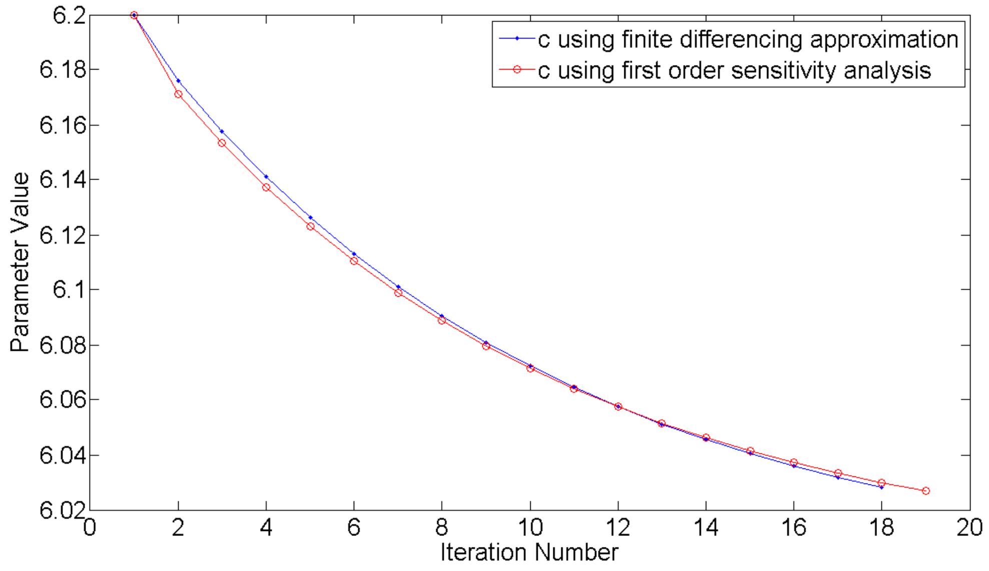

Figure 18 - Value of parameter c versus each iteration with the finite approximation method and the first order sensitivity method.

7.1.2 Objective Function Convergence

The two algorithms were compared using the same starting values and convergence parameters. Figure 19 shows their ability to minimize the objective function at each iteration.

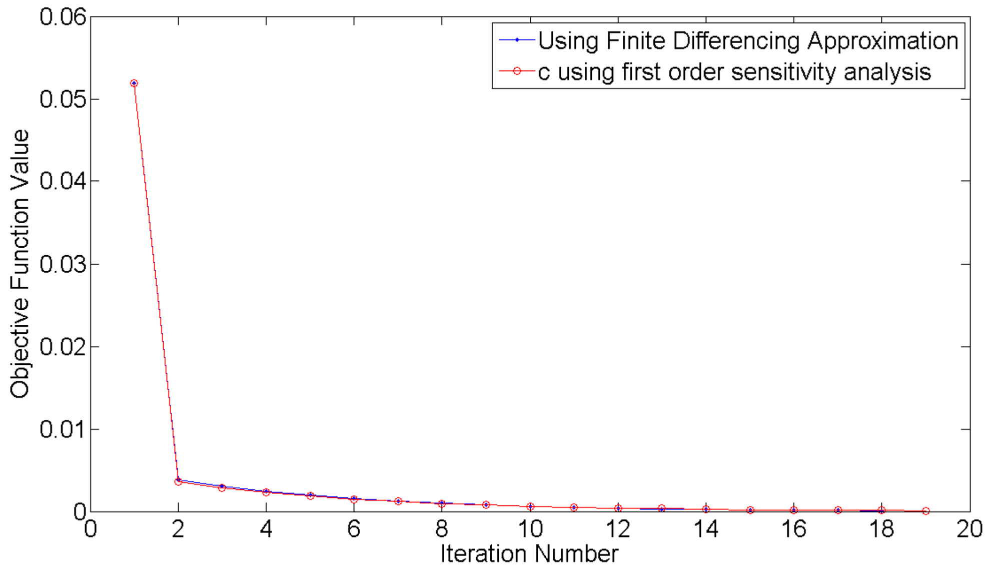

Figure 19 - Objective function value vs. iteration for both the finite approximation method and first order sensitivity analysis method.

7.1.3 System Responses Using Calculated Parameters

The original guess parameters and the final parameters found by both the finite differencing method and the first order sensitivity analysis method were then overlayed by the experimental system response, for both responses α and y.

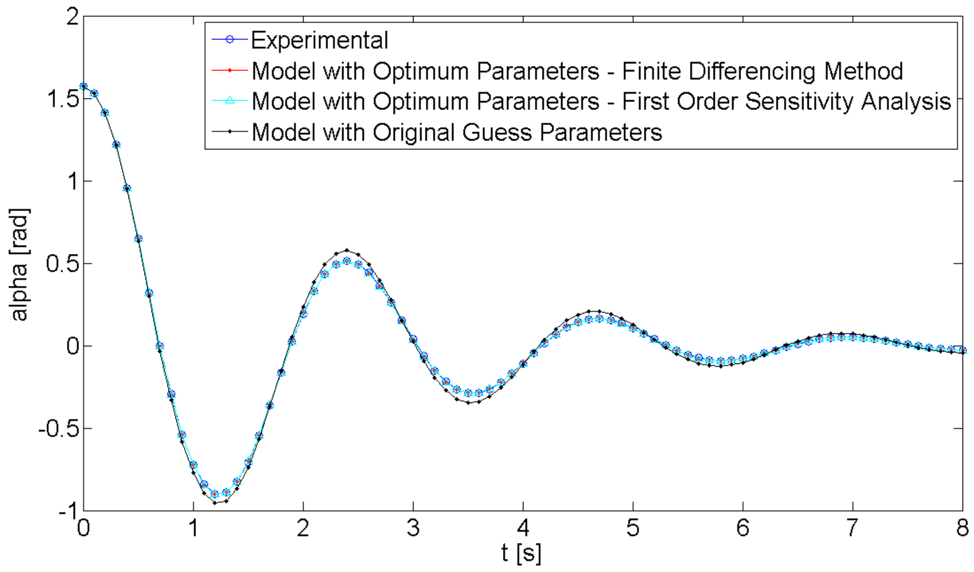

Figure 20 - α response of pendulum system with original guess parameters and response by parameters found with the two methods.

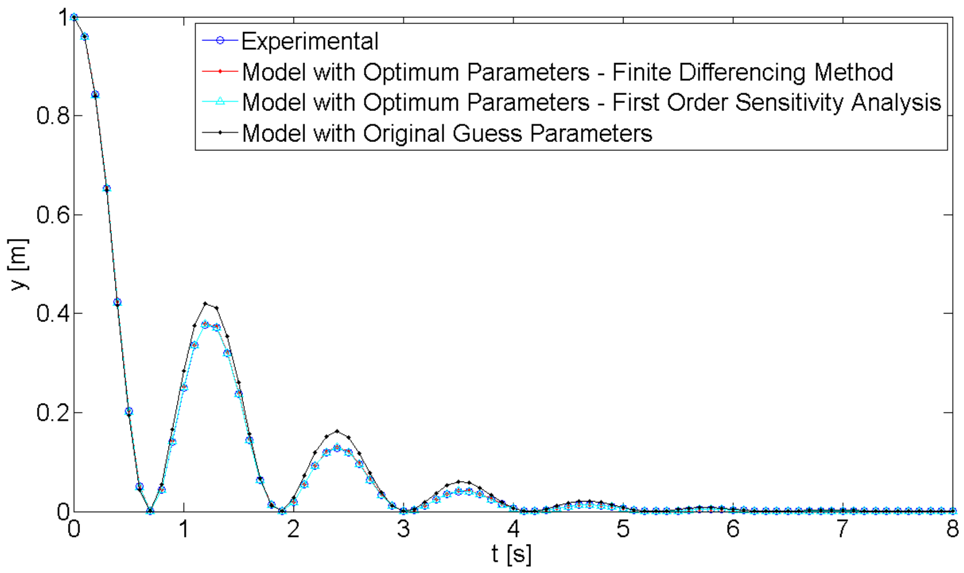

Figure 21 - y response of pendulum system with original guess parameters and response by parameters found with the two methods.

Both methods return optimum parameters that when used with the model produce almost identical results to the experimental response of the system.

7.2 Overview of Results

Table 9 includes various values that describe the efficiency of the two methods implemented.

Computational Component	Finite Approximation	First Order Sensitivity Analysis
Iterations	18	19
Computation Time [s]	335	454
Time Per Iteration [s/iteration]	18.6	23.9
Value of c_{final} [N/m]	6.028	6.026
Value of V_{final}	9.697×10^{-5}	8.845×10^{-5}

Table 9 - Comparisons of efficiency of the finite approximation method and the first order sensitivity method.

If extremely close parameter values are not necessary, then it may be more efficient to use the first order sensitivity analysis. As can be seen in Figure 18, until iteration 12, parameter c approaches its experimental value faster than the finite differencing method. Beyond iteration 12, the finite differencing method is slightly more efficient at identifying the value of parameter c. Despite this fact, as seen in Figure 19, both methods minimize the objective function almost with the exact same efficiency.

Overall, the methods are very comparable in terms of convergence values of parameter c and minimizing the objective function. However, it clear that the computation time when using the first order sensitivity analysis is higher. This is due to the fact that both the DAEs of motion of the system and the ODE sensitivity equations must be solved, where as in the finite differencing method only the DAEs of motion need to be solved. In the case of the pendulum system, the finite difference approximation method is slightly more efficient than the first order sensitivity analysis method. This may not be true for more complex systems, as finite differencing may not be accurate enough for systems where many more parameters need to be identified.

8 Discussion

Parameter identification for even the most simple of multibody systems requires high computational and mathematical effort, even with only a first order sensitivity analysis. Formulation, especially for time derivatives, can quickly become complex even for the most simple systems. More literature needs to be produced on the topic of determining partial derivatives of the system as described in Eqs. (1) and (2), without eliminating Langrange multipliers. A proper example of derivations of these equations on a complex system such as a slider crank mechanism needs to be produced. As the science of parameter identification becomes more advanced, it may become impossible to use only first order state equations to describe multibody systems. Some authors, such as Vyasarani et. al. [16] are studying possible ways to extend current ODE methods to be applicable to DAE systems.

9 Summary and Conclusions

Both a finite differencing approximation parameter identification method and a first order sensitivity analysis parameter identification method were implemented on a coupled spring-damper pendulum system, which is governed by second order differential algebraic equations. The second order differential equations were reduced in a way that they could be solved in by the use of both MATLAB and APMonitor softwares. Rigorous testing of each method displayed the strengths and weaknesses of each method.

Using finite differencing approximations for the Hessian and gradient is a viable method for parameter identification, and in the case of the pendulum system, provides a faster convergence than the first order sensitivity analysis. However, it may be that the first order sensitivity analysis as described in Ding et. al. [23] must also be used with the second order sensitivity analysis for the fastest convergence times. The value of convergence parameters in these Newton Method type processes, such as α, ϵ, and Δq have a large impact on convergence time. In some cases, if α is too large, convergence is impossible. Although slightly less efficient than the finite differencing method, the power of the sensitivity analysis, by the fact it utilizes a homogenous ODE, may prove very useful for multibody systems with much more complex equations of motion, where simple finite difference approximations may not work.

Furthermore, as is seen in Section 3, the literature review, the variety of example and test systems varies greatly between each method. A standard system should be established that can be applied to all parameter identification methods. Such a system would have to be able to be described accurately by both DAEs and ODEs, as these two governing systems cover nearly all parameter identification methods. A large review of convergence efficiency and implementation effort of each of the methods in the literature review would be insightful and informative to the discipline of parameter identification.

10 Acknowledgment

This project was supported by the Austrian Marshall Plan Foundation Scholarship No. 363 and the University of Applied Sciences Upper Austria. Special thanks to Professor Wolfgang Witteveen at the University of Applied Sciences Upper Austria, who was the mentor for this project.

11 References

[1] Ok, J. K., Sohn, J. H., Yoo, W. S., ‘Development of nonlinear coupled mode bushing model based on the Bouc-Wen hysteretic model’, *ASME*, 2007.

[2] Kim, H. J., Yoo, W. S., Ok, J. K., & Kang, D. W., ‘Parameter identification of damping models in multibody dynamic simulation of mechanical systems’, *Multibody System Dynamics*, **22.4**, 2009, 383-398.

[3] Bouc, R., ‘Forced vibration of mechanical systems with hysteresis. In: proceedings of the 4th international conference on nonlinear oscillations’, Czechoslovakia, 1967.

[4] Wen, Y.K., ‘Approximate method for nonlinear random vibration’, *J. Eng. Mech. Div.* **101.4**, 1975, 249–264.

[5] ADAMS User manual, MSC software, USA, 1998.

[6] Ludwig, R., Gerstmayr, J., ‘Automatic parameter identification for mechatronic systems’, *Multibody System Dynamics, Robotics and Control*, 2013, 193-212.

[7] Ludwig, R., Gerstmayr, J., ‘Automatic parameter identification for generic robot models’, *Proceedings of the Multibody Dynamics ECCOMAS Thematics, Brussels*, 2013, 1-16.

[8] Reischl, D., Ludwig, R., ‘Automated identification of a nonlinear viscoelastic model for injection molding machines’, *Austrian Center of Competence in Mechatronics*, 2011, 1-7.

[9] Ebrahimi, C., Kovecses, J., ‘Sensitivity analysis for estimation of inertial parameters of multibody mechanical systems’, *Mechanical Systems and Signal Processing*, **24**, 2010, 19–28.

[10] Shome, S. S., Beale, D. G., Wang, D., ‘A general method for estimating dynamic parameters of spatial mechanisms’, *Nonlinear Dynamics*, **16.4**, 1998, 349-368.

[11] Schiehlen, M., Hu, B., ‘Parameter identification of nonlinear multi-body system using correlation techniques’, *Institute B of Mechanics, University of Stuttgard*, 2001.

[12] Blanchard, E., Sandu, A., Sandu, C., ‘Parameter estimation for mechanical systems using an extended Kalman filter’, *Virginia Polytechnic University and State University*, 2008.

[13] Baumgarte, J., ‘Stabilization of constraints and integrals of motion in dynamical systems’, *Computer Methods in Applied Mechanics and Engineering*, **1.1**, 1972, 1-16.

[14] Shi, P., McPhee, J., ‘Dynamics of flexible multibody systems using virtual work and symbolic programming’, *Multibody System Dynamics*, **4.4**, 2000, 355-381.

[15] McPhee, J., Shi, P., and Piedboeuf, J.-C., ‘Dynamics of multibody systems using virtual work and symbolic programming’, *Mathematical and Computer Modeling of Dynamical Systems*, **8.2**, 2002, 137-155.

[16] Vyasarayani, C. P., Uchida, T., McPhee, J., ‘Parameter identification in multibody systems using lie series solutions and symbolic computation’, *Journal of Computational and Nonlinear Dynamics*, **6**, 2011, 4.

[17] Vyasarayani, C. P., Uchida, T., Carvalho, A., McPhee, J., ‘Parameter identification in dynamic systems using the homotopy optimization approach’, *Multibody System Dynamics*, **26**, 2011, 411-424.

[18] Gerdin, M., Glad, T., Ljung, L., ‘Parameter estimation in linear differential-algebraic equations’, *In proceedings of the 13th IFAC symposium on system identification*, 2003, 1530-1535.

[19] Serban, R., Freeman, J. S., ‘Identification and indentifiability of unknown parameters in multibody dynamic systems’, *Multibody System Dynamics*, **5**, 2001, 335-350.

[20] Serban, R., Haug, E.J., ‘Analytical derivatives for multibody system analyses’, *Mechanics of Structures and Machines*, 1998, **26.2**, 145–173.

[21] Gerdts, M., ‘Parameter identification in higher index DAE systems’, *Department of Mathematics, University of Hamburg*, 2005, 1-24.

[22] Smith, G. D., ‘Numerical solution of partial differential equations: finite difference methods’, *Oxford University Press*, 1985.

[23] Ding, J., Pan, Z., Chen, L., ‘Parameter identification of multibody systems based on second order sensitivity analysis’, *International Journal of Non-Linear Mechanics*, **47**, 2011, 1105-1110.

[24] MATLAB, 7.12.0 (R2011a), www.mathworks.com/products/matlab/.

[25] APMonitor Optimization Suite, 0.5.8e, www.apmonitor.com/wiki/index.php/Main/MATLAB.