1 Introduction

In recent years, due to the advancement of computational capabilities, mathematical modeling has become a major tool used by mechanical engineers in the design and analysis of physical systems. The mathematical model governing the dynamics of a system is usually known, particularly in the case of multibody systems. Mathematical modeling has become a major tool used by engineers in the design and analysis of multibody systems. The equations describing the motion of multibody systems are generally highly nonlinear due to the geometrical nonlinearities associated with rigid body rotations, which makes the problem of parameter identification particularly challenging in such systems.

The equations describing the motion of multibody systems are generally highly nonlinear due to the geometrical nonlinearities associated with rigid body rotations, which makes the problem of parameter identification particularly challenging in such systems. Only a few studies have addressed the estimation of parameters for general multibody systems [4–8]; some work has also been done on parameter estimation techniques for specific robotic applications (see references in Ref. [6]) and biomechanical systems [9,10]. Pilipchuk and Tan [11] proposed a technique using Lie series solutions and used it to monitor the degradation of joints in a simple nonlinear structural system by estimating their time-varying stiffnesses. In this work, we use the Lie series technique to estimate parameters in multibody systems. Symbolic computational procedures are used to generate the equations of motion, the associated Lie series solutions, and a symbolic objective function that minimizes the difference between the experimental and simulated data, thereby providing an automated approach to parameter identification in multibody systems. We also take advantage of the fact that such a series solution is very accurate (neglecting truncation error) in the neighborhood of the point about which it is expanded.

There are two main advantages of using Lie series solutions for parameter identification. First, the series solution is symbolic in the system parameters, which facilitates the symbolic generation of the Jacobian matrix of the objective function. The Jacobian matrix can be used to improve the convergence characteristics of the optimization procedure used for estimating the parameters. Second, the estimation algorithm can be used to estimate slowly-varying nonlinear parameters or nonlinear parameters that depend on the system states [11], which cannot be estimated using linear regression methods [5–8]. Thus, the Lie series approach is a more general method and can be applied to a wider variety of parameter identification problems. In Sec. 2, we describe the method used to obtain a Lie series solution for open- and closed-loop-topology systems. We then consider three examples in Sec. 3 to demonstrate the application of this technique to parameter identification in multibody systems. Some general challenges encountered when estimating parameters in practice are also discussed. Conclusions and future work are outlined in Sec 4.

2 Mathematical Modeling

In this section, we first describe the method of obtaining a Lie series solution for a dynamic system governed by ordinary differential equations (ODEs). Since a Lie series solution can only be obtained for systems of ODEs, we present one approach for eliminating the Lagrange multipliers and converting a system of differential-algebraic equations (DAEs) into ODEs, thus allowing us to apply the Lie series approach to constrained mechanical systems as well. With the exception of the work by Serban and Freeman [4], most of the existing techniques also involve eliminating the Lagrange multipliers before estimating the system parameters [5–10]. Serban and Freeman used an objective function that is based on the difference between the experimental data and the data obtained from a numerical simulation performed over the same length of time. Thus, a series of simulations is performed, each of which updates the parameters in the direction required to

1 Corresponding author.

Contributed by the Design Engineering Division of ASME for publication in the JOURNAL OF COMPUTATIONAL AND NONLINEAR DYNAMICS. Manuscript received June 29, 2010; final manuscript received February 14, 2011; published online April 28, 2011. Assoc. Editor: Hamid M. Lankarani.
minimize the objective function. This technique can be described as optimization with global integration since each optimization step involves integrating the system equations over the entire length of time for which experimental data were collected. The approach used herein can be described as optimization with local integration since we optimize the objective function at every time step of the simulation to obtain the parameters.

2.1 Solving ODE Systems Using the Lie Series. Presently, we shall assume that the mathematical model of the physical system is governed by the following system of ODEs, which are nonlinear in the general case:

\[ \mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{p}) \]  

where \( \mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T \) are the states of the system, and the external loads and gyroscopic terms are contained in vector \( \mathbf{f} = [f_1, f_2, \ldots, f_n]^T \); it is assumed that we wish to estimate \( np \) parameters \( \mathbf{p} = [p_1, p_2, \ldots, p_{np}] \). The initial conditions are given as follows:

\[ \mathbf{x}(t_0) = \mathbf{x}_0 = [x_{10}, x_{20}, \ldots, x_{n0}]^T \]  

The Lie series solution [11,12] of the above system (Eq. (1)) is defined as follows:

\[ \mathbf{x}(t) = e^{Dt}\mathbf{x}_0 = \mathbf{x}_0 + \sum_{j=1}^{\infty} \frac{(t-t_0)^j}{j!} D\mathbf{x}_0 \]  

where \( D \) is the Lie operator:

\[ D = \frac{\partial}{\partial t_0} + \sum_{i=1}^{k} f_i(\mathbf{x}_0, \mathbf{p}) \frac{\partial}{\partial x_{i0}} \]  

The Lie series solutions (Eq. (3)) are closed-form analytical series solutions of the ODE system (Eq. (1)) and are exact when an infinite number of terms is considered in the series. Since it is impossible to retain an infinite number of terms in the series expansion, we use a finite series truncation and maintain the integrity of the solution by expanding locally, using the states at time \( t_1 \) to predict the states at time \( t_1 + \delta t \). A Lie series solution is very accurate in the neighborhood of the point about which it is expanded since the only error incurred is the truncation error. Although it has been found that a Lie series integrator performs better than other numerical integrators for certain time-varying systems [12], we are not using the series solution in this manner, in part due to the growth of the truncation error over time.

In this work, we are attempting to estimate the parameters of a system given a set of experimental data. Let us assume that we have measured the states of the system experimentally, \( \mathbf{x}_e = [x'_1, x'_2, \ldots, x'_n]^T \), using a sampling interval of \( \delta t \). We now use the experimental data \( \mathbf{x}(t_i) \) at time \( t_i \) to predict the states \( \mathbf{x}(t_{i+1}) \) at time \( t_{i+1} \) using the Lie series solution. We are essentially using the Lie series solution to take a single time step of size \( \delta t \) from the known experimental states. Since we also have experimental data \( \mathbf{x}(t_{i+1}) \) at time \( t_{i+1} \), we attempt to minimize the following objective function to estimate the parameters:

\[ F_{obj} = \frac{1}{2} [\mathbf{x}(t_{i+1}) - \mathbf{x}(t_{i+1})]^T [\mathbf{x}(t_{i+1}) - \mathbf{x}(t_{i+1})] \]  

Note that \( \mathbf{x}(t_{i+1}) \) is a symbolic function of the system parameters, so minimizing the above objective function will provide an approximation of the system parameters at a given time. Equation (5) can be minimized at all time points for which experimental data have been collected, and the resulting time-varying parameters can then be averaged over the duration of the simulation to obtain the final parameter estimates. Since the objective function is both symbolic and a function of the system parameters, its Jacobian can be obtained symbolically and used in gradient-based minimization procedures [13,14] to improve the convergence characteristics of optimization algorithms.

2.2 Converting DAEs to ODEs. If joint coordinates are used, the governing dynamic equations for an open-loop-topology multibody system are pure ODEs. The presence of closed kinematic chains, however, adds nonlinear algebraic constraint equations to the ODEs of motion, thereby producing a set of DAEs. Since a Lie series solution can only be generated for ODEs, it is first necessary to convert the DAEs governing the motion of constrained mechanisms into ODEs. In this work, DYNAPLEXPRO is used to automatically generate the governing dynamic equations symbolically for planar and spatial multibody systems of arbitrary topology. DYNAPLEXPRO uses linear graph theory [15] and symbolic computation to generate computationally efficient equations of motion. If the resulting dynamic equations are differential-algebraic in nature, we convert them into ODEs using projection and Baumgarte stabilization, as described below.

An augmented formulation is used in this work, which results in \( n_q \) ODEs (one for each generalized coordinate) and \( n_c \) nonlinear algebraic constraint equations for a multibody system with \( f=n_q-n_c \) degrees-of-freedom:

\[ \mathbf{M}\ddot{\mathbf{q}} + \mathbf{F}_{\Phi} \lambda = \mathbf{F} \]  

\[ \mathbf{F} = 0 \]  

where \( \mathbf{M} \) is the mass matrix, \( \mathbf{q} \) is the generalized coordinate vector, \( \mathbf{F}_{\Phi} \) is the Jacobian matrix, \( \lambda \) are the Lagrange multipliers, \( \mathbf{F} \) is the generalized force vector, and \( \Phi \) are the constraint equations. The generalized coordinates \( \mathbf{q} \) are partitioned into independent and dependent sets (\( \mathbf{q}_d \) and \( \mathbf{q}_c \), respectively) and can be transformed into the former as follows [16]:

\[ \dot{\mathbf{q}} = \mathbf{B}\dot{\mathbf{q}} \]  

Transformation matrix \( \mathbf{B} \) is defined as follows:

\[ \mathbf{B} = \begin{bmatrix} -\mathbf{\Phi}^\top \mathbf{\Phi} & 1 \end{bmatrix} \]  

where \( \mathbf{\Phi} = \frac{\partial \mathbf{\Phi}}{\partial \mathbf{q}_d}, \mathbf{\Phi} = \frac{\partial \mathbf{\Phi}}{\partial \mathbf{q}_c}, \) and \( 1 \) is an identity matrix of dimension \( f \). Since \( \mathbf{B} \) is an orthogonal complement to the Jacobian matrix \( \mathbf{\Phi}_q \), premultiplying Eq. (6) by \( \mathbf{B}^\top \) eliminates the Lagrange multipliers from the dynamic equations:

\[ \mathbf{B}^\top \mathbf{M}\ddot{\mathbf{q}} = \mathbf{B}^\top \mathbf{F} \]  

The kinematic constraints (Eq. (7)) can be enforced by twice differentiating them with respect to time and integrating the resulting acceleration-level constraints simultaneously with Eq. (10). To avoid significant position-level constraint violations, the acceleration-, velocity-, and position-level constraint equations can be combined into a single expression and integrated along with the equations of motion (see Baumgarte [17]):

\[ \ddot{\mathbf{\Phi}} + 2\alpha_{\Phi}\dot{\mathbf{\Phi}} + \beta_{\Phi}\mathbf{\Phi} = 0 \]  

where appropriate values for Baumgarte parameters \( \alpha_{\Phi} \) and \( \beta_{\Phi} \) must be determined for each system. Combining Eqs. (10) and (11), we obtain the following system of \( n_q+n_c \) ODEs:

\[ \begin{bmatrix} \mathbf{B}^\top \mathbf{M} & \mathbf{B}^\top \mathbf{F} \\ \mathbf{\Phi}'_q \end{bmatrix} \dot{\mathbf{q}} = \begin{bmatrix} \mathbf{B}^\top \mathbf{F} \\ -2\alpha_{\Phi}\mathbf{\Phi} - \beta_{\Phi}\mathbf{\Phi} \end{bmatrix} \]  

Note that the projection technique outlined above may not be a practical choice for large multibody systems. Alternative modeling approaches such as the penalty approach [18], where the joints are modeled as stiff springs, always generate systems of ODEs and may be more appropriate for large systems. In Sec. 3, we demonstrate the use of the Lie series for estimating parameters in multibody systems.

"DYNAPLEXPRO is now part of the Multibody package in MAPLE, a multidomain physical modeling tool developed by Maplesoft."
3 Results and Discussion

In this section, we present several examples of estimating system parameters. In lieu of experimental data, we first numerically integrate the dynamic equations with a set of ideal system parameters. We then use these data to estimate the system parameters using the Lie series approach described above. To demonstrate the process in detail, we first consider a single-degree-of-freedom structural system; two examples involving multibody systems follow. Note that SI units are used throughout.

3.1 Estimating Parameters of a Structural System. The equation of motion governing the dynamics of a forced spring-mass system with nonlinear stiffness can be expressed as follows:

\[ m \ddot{x} + k x^n = F \sin(\omega t) \]  

with known initial conditions \( x(t_0) \) and \( \dot{x}(t_0) \). Equation (13) can be written in state-space form:

\[ \dot{x}_1 = f_1 = x_2 \]  

\[ \dot{x}_2 = f_2 = -\frac{k}{m} x_1^n + \frac{F}{m} \sin(\omega t) \]  

with initial conditions \( x_1(t_0) = x_{10} \) and \( x_2(t_0) = x_{20} \). The Lie operator for the above dynamic system can be expressed as follows:

\[ D = f_1 \frac{\partial}{\partial x_1} + f_2 \frac{\partial}{\partial x_2} + \frac{\partial}{\partial t_0} \]  

The Lie series solutions for Eqs. (14) and (15) can now be generated:

\[ x_1(t) = x_1(t_0) + (t - t_0) x_2(t_0) + \frac{(t - t_0)^2}{2!} D^2 x_1(t_0) + \cdots \]  

\[ x_2(t) = x_2(t_0) + (t - t_0) x_3(t_0) + \frac{(t - t_0)^2}{2!} D^2 x_2(t_0) + \cdots \]  

We now evaluate the coefficients of the first Lie series solution, given by Eq. (17), using the Lie operator (Eq. (16)):

\[ D x_1 = f_1(t_0) \frac{\partial}{\partial x_1} + f_2(t_0) \frac{\partial}{\partial x_2} + \frac{\partial}{\partial t_0} \]  

\[ D^2 x_1 = D^2 x_1(t_0) = f_1(t_0) \frac{\partial}{\partial x_1} + f_2(t_0) \frac{\partial}{\partial x_2} + \frac{\partial}{\partial t_0} \]  

\[ = -\frac{k}{m} x_1^n + \frac{F}{m} \sin(\omega t_0) \]  

\[ = \frac{F}{m} \cos(\omega t_0) \]  

Note that the Lie series coefficients can be evaluated recursively:

\[ D^{n+1} x_0 = D^{n} x_0 \]  

A symbolic computational tool such as MAPLE is used to generate the coefficients of the Lie series solution automatically. We now substitute the series coefficients into Eq. (17) to obtain the symbolic solution for \( x_1(t) \):

\[ x_1(t) = x_{10} + (t - t_0) x_{20} + \frac{(t - t_0)^2}{2!} \left( -\frac{k}{m} x_1^n + \frac{F}{m} \sin(\omega t_0) \right) + \cdots \]  

At this point, we could repeat the entire procedure for \( x_2(t) \); however, since \( x_2(t) = \dot{x}_1(t) \), we can simply differentiate Eq. (23) with respect to time to obtain the series solution for \( x_2(t) \):

\[ x_2(t) = \ddot{x}_2(t_0) x_0 + (t - t_0) x_3(t_0) + \frac{(t - t_0)^2}{2!} D^2 x_2(t_0) + \cdots \]  

\[ = \frac{F}{m} \cos(\omega t_0) \]  

For \( k = m = n = 1, F = 0 \), and initial conditions \( x_{10} = 0 \) and \( x_{20} = 0 \), we have the following analytical solution for \( x_1(t) \) from the Lie series solution (Eq. (23)):

\[ x_1(t) = t - \frac{t^3}{3!} + \cdots = \sin(t) \]  

Since it is impossible to retain an infinite number of terms in the series expansion, we use a finite number of truncation and expand it locally to avoid the accumulation of truncation error. We illustrate the effectiveness of this approach using a Taylor series expansion of \( \sin(x) \) about a point \( x_0 \):

\[ \sin(x + x_0) = \sin(x_0) + (x - x_0) \cos(x_0) + \frac{(x - x_0)^2}{2!} \sin(x_0) \\ - \frac{(x - x_0)^3}{3!} \cos(x_0) + \cdots \]  

Retaining only two nonzero terms in the expansion and plotting the resulting polynomial against the actual sine function, as shown in Fig. 1, it is clear that the truncated series approximates the sine curve accurately near the expansion point \( x = 0 \) but deviates significantly elsewhere. One method of obtaining a more accurate representation is to increase the number of terms used in the series expansion. Another method involves periodically updating the point about which the Taylor series is expanded, using an expansion about \( x = 0 \) to represent \( \sin(x) \) in the range \( x \in [0, \pi/2] \), using an expansion about \( x = \pi/2 \) in the range \( x \in [\pi/2, \pi] \), and so on. Use of local Taylor series expansions provides a much more accurate representation and matches the actual sine curve very closely, as shown in Fig. 1. This level of accuracy can be obtained...
using only two nonzero terms because we are always evaluating a Taylor series very close to the point about which it has been expanded.

Since Eqs. (23) and (24) are nothing but the Taylor series expansions of the solutions of differential equations (14) and (15), we can use the same local expansion strategy here. A finite truncation of the above series can be used to obtain the solution at time \( t_{0} + \Delta t \), where \( \Delta t \) is a small time interval from the initial time \( t_{0} \). Once the states of the system at time \( t_{0} + \Delta t \) have been computed, the initial time \( t_{0} \) can be updated to \( t_{0} + \Delta t \), and Eqs. (23) and (24) can again be used to obtain the solution at time \( t_{0} + 2\Delta t \).

In essence, Eqs. (23) and (24) are expressed recursively and can be used as a time-marching scheme:

\[
\begin{align*}
x_{1}(t_{i+1}) &= x_{1}(t_{i}) + (t_{i+1} - t_{i})x_{2}(t_{i}) + \frac{(t_{i+1} - t_{i})^{2}}{2} \left[ -\frac{k}{m} x_{1}(t_{i})^{n} + F \cos(\omega t_{i}) \right] + \cdots \\
x_{2}(t_{i+1}) &= x_{2}(t_{i}) + (t_{i+1} - t_{i}) \left[ -\frac{k}{m} x_{1}(t_{i})^{n} + F \sin(\omega t_{i}) \right] + \frac{(t_{i+1} - t_{i})^{2}}{2} \left[ -\frac{k}{m} x_{1}(t_{i})^{n-1} + \frac{F \omega}{m} \cos(\omega t_{i}) \right] + \cdots 
\end{align*}
\]  

(27)

(28)

Figure 2 compares the time response of the system obtained using a three-term Lie series local expansion to that obtained using the ode15s integrator in MATLAB. For the numerical simulation, we have used \( F=20 \), \( k=10 \), \( m=1 \), and \( \omega=10 \). Clearly, the two solutions are in very close agreement. Although the truncation error increases as we march forward in time, we are not using the Lie series solution to solve the forward dynamic problem in this work. In fact, we are only using the Lie series solution to take a single time step from each known experimental state, so the truncation error incurred in the solution does not accumulate.

Let us assume that the experimental values of the system states are available and are given by \( \bar{x}_{1}(t) \) and \( \bar{x}_{2}(t) \). The system parameters must be found such that the objective function \( F_{obj}=(x_{1}'(t) - x_{1}(t))^{2}+(x_{2}'(t) - x_{2}(t))^{2} \) is minimized. We first write the objective function in a discrete form for each time step:

Since the objective function (Eq. (29)) at a given time is purely a function of the system parameters, it can be minimized to find the unknown parameters using an existing nonlinear least-squares/gradient-descent minimization technique. As an example, we use the above objective function to estimate the spring constant that corresponds to the numerical solution given by ode15s. In this case, we already know that the actual spring constant is \( k_{act}=10 \), but to evaluate the performance of this estimation technique, we treat \( k \) as unknown and use the solution provided by ode15s as the experimental states. The state vector is stored after every milliseconds of the simulation. Equation (29) is then minimized at each of these points using the fminunc unconstrained minimization routine in MATLAB. The estimated spring constant, once averaged over the duration of the simulation, is found to be \( k_{est}=10.026 \). The difference between the response obtained using \( k_{act} \) and \( k_{est} \) is negligible, which is an indication of the utility of this method.

Maple procedures were written for converting DAE systems into ODE systems using the method discussed in Sec. 2.2 (Algorithm 1) and for generating the required Lie series solutions (Algorithms 2 and 3). In this work, we use the optimization routines available in MATLAB to minimize the objective function. A flowchart outlining the parameter estimation process presented above is shown in Fig. 3.
Algorithm 1 — GetODEs

Input: Mass matrix (M), generalized coordinate vector (q), generalized force vector (F), and list of constraint equations (Φ)
Output: Baumgarte-stabilized ODEs of motion

#partition q into dependent and independent sets
q_j ← q_{1..dim(Φ)}
q_i ← q_{dim(Φ)+1..dim(q)}
#construct orthogonal complement to the Jacobian Φ_q
B ← −Inv(Jac(Φ, q_j)) × Jac(Φ, q_i)
#assemble ODEs in the form ̇Mq=F
M ← Transpose(B) × M
Jac(Φ, q)
  + Transpose(B) × F
  + ̇q_j × 2αgarn/D − β^2_q Φ
#generate final ODEs in the form ̇q = A
A ← Inverse(M) × F
return A

Algorithm 2 — SerSol

Input: Right-hand sides of state-space equations (eqs), state-space variables (vars), number of terms desired in series expansion (N), and integration step size (T)
Output: N-term Lie series solution for each variable in vars

for i = 1 to dim(vars) do
  sol_i ← vars_i + ∑_j=1^N (T/j × DN(j, vars_j))
end for
return sol

Algorithm 3 — DN

Input: Right-hand sides of state-space equations (eqs), state-space variables (vars), term number (n_t), and variable with respect to which D operator is to be taken (x)
Output: n_t term of D operator taken with respect to x

if n_t=0 then
  return x
else
  return ∑_j=1^{dim(eqs)} (eqs_j × 1/∂vars_j) × DN(n_t−1, x) + 1/T × DN(n_t−1, x)
end if

3.2 Estimating Parameters of a Spatial Slider-crank. In this example, we estimate the inertial parameters of the spatial slider-crank mechanism shown in Fig. 4 [1]. We use joint coordinates q = {α, β, θ, s} to model the system, thereby obtaining four second-order differential equations and three algebraic constraint equations for this single-degree-of-freedom system. We eliminate the Lagrange multipliers from the dynamic equations using the projection method presented in Sec. 2.2. The differential equations obtained after applying Baumgarte stabilization (using αg = βB = 5) are converted into state-space form, and a two-term Lie series solution is generated and converted into a MATLAB procedure using our MAPLE code. A 2-second simulation is performed to predict the motion of the mechanism, the results of which are treated as experimental data. This simulation is performed using the DYNAFLEXPRO package in MAPLE and the rkf45 numerical integrator with absolute and relative error tolerances of 10^-6. The crank is initially rotated to an angle of θ = π/4, and the mechanism is allowed to fall freely under the force of gravity. The parameters A1 = 0.1 m, A2 = 0.12 m, m1 = 0.12 kg, m2 = 0.5 kg, m3 = 2.0 kg, \( \ell_1 = 0.08 \) m, and \( \ell_2 = 0.3 \) m are assumed to be known, where the crank, connecting rod, and piston are numbered as bodies 1, 2, and 3, respectively; the estimated parameters are shown in Table 1.

The results obtained from a numerical simulation using the above parameters are used to estimate the inertias of the links. Since it is possible to obtain inertial parameters that are completely different from the actual values while still matching the simulation results exactly, we use a constrained optimization approach (fmincon in MATLAB) for this problem, again optimizing once for each millisecond of the simulation. We use lower and upper bounds of 0.8 and 1.2 times the actual parameter values, respectively; the lower bounds are used as the initial guesses for each parameter being estimated. In practice, such bounds can be estimated using computer-aided design (CAD) models or other approximation techniques. Averaging over the 2-second simulation, we obtain the estimated inertial parameters shown in the third column of Table 1. Again, the estimated values are reasonably close to the actual values, as indicated in the last column of the table. Note that the estimate for I_{x2} is significantly further from the actual value than the other identified parameters. A comparison between the simulated experimental displacements and the displacements obtained using the estimated parameters is shown in Fig. 5. Clearly, the estimated parameters produce results that are very close to the experimental results. Note that the richness of the actuation signal plays an important role; the system must be excited such that all parameters are being sought participate in generating the observed motion. In this case, the apparent insensitivity of the system to parameter I_{x2} may be related to the fact that the system is only being driven by the force of gravity.

3.3 Estimating Parameters of a Vehicle Model. In the final example, we estimate parameters for the eight-degree-of-freedom

Table 1 Estimated spatial slider-crank system parameters [1]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Actual</th>
<th>Identified</th>
<th>Difference (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I_{x1}</td>
<td>1.0 × 10^{-4} kg m²</td>
<td>9.64 × 10^{-4} kg m²</td>
<td>3.6</td>
</tr>
<tr>
<td>I_{x2}</td>
<td>4.0 × 10^{-4} kg m²</td>
<td>4.64 × 10^{-4} kg m²</td>
<td>16.0</td>
</tr>
<tr>
<td>I_{y1}</td>
<td>4.0 × 10^{-2} kg m²</td>
<td>3.91 × 10^{-2} kg m²</td>
<td>2.2</td>
</tr>
<tr>
<td>I_{y2}</td>
<td>4.0 × 10^{-2} kg m²</td>
<td>4.07 × 10^{-2} kg m²</td>
<td>1.7</td>
</tr>
</tbody>
</table>
lumped-parameter vehicle model shown in Fig. 6. This model was used by Bouazara and Richard [19] to compare different control strategies for vehicle comfort. The mass of each tire is lumped together with one-quarter of the suspension components and is considered as a single mass that can move vertically. The vehicle chassis is assumed to be a rigid body with mass $M$ and is allowed to pitch ($\phi$), roll ($\varphi$), and move vertically ($Z$). The suspensions and tire compliances are modeled using linear spring-damper combinations. Since this model has been used to study ride comfort, the dynamics of the driver and seat are included. The mass of the driver and seat are lumped together ($m_d$), and the compliance of the seat is modeled as a spring-damper combination ($k_d$ and $c_d$) that is attached to the chassis at distances of $r_x$ along the $x$-axis and $r_y$ along the $y$-axis from the chassis center of mass. Only the vertical translation of the driver seat is modeled. The geometric parameters used in the simulation are shown in Table 2; the dynamic parameters are given in Table 3.

The vehicle is driven at a constant speed of 28 m/s over the double-bump road profile shown in Fig. 7. The equations of motion are again generated using DYNAFLEXPRO, and a 7-second simulation is performed in MAPLE using the rkf45 numerical integrator with absolute and relative error tolerances of $10^{-6}$. The response of the vehicle was compared to the results reported by Bouazara and Richard [19] and was found to be in good agreement. A Lie series solution is constructed using our MAPLE code, and a MATLAB procedure is generated for evaluating the objective function during the parameter identification process. The states of the system obtained from the MAPLE simulation are treated as experimental data for the purpose of estimating all four suspension stiffnesses. We use the fminunc unconstrained optimization routine in MATLAB to minimize the objective function. The estimated stiffness parameters at each time step of the optimization process are shown in Fig. 8. It can be seen that the identified values converge to the actual parameters; however, several spikes are visible in the first 3 seconds of the simulation. As such, the averages are computed starting from $t=3$ s and are all within 0.3% of the actual parameters, as shown in Table 3.

At some time steps, the identified parameters become negative and are, thus, nonphysical—but they are still optimal values, as can be confirmed by the objective function value at these time instants (see Fig. 9). One method of addressing the problem of obtaining nonphysical parameters is to use a constrained optimization technique. Another approach can be revealed by carefully examining Fig. 9, where it is clear that the value of the objective function is somewhat higher when the optimal parameters are nonphysical. In some cases, it may be sufficient to simply com-

### Table 2 Geometric parameters used in the vehicle model [19]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>$x$ position from center of mass (m)</th>
<th>$y$ position from center of mass (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front-left tire location</td>
<td>$l_f=1.011$</td>
<td>$b=0.761$</td>
</tr>
<tr>
<td>Front-right tire location</td>
<td>$l_f=1.011$</td>
<td>$a=-0.761$</td>
</tr>
<tr>
<td>Rear-left tire location</td>
<td>$l_r=-1.803$</td>
<td>$b=0.761$</td>
</tr>
<tr>
<td>Rear-right tire location</td>
<td>$l_r=-1.803$</td>
<td>$a=-0.761$</td>
</tr>
<tr>
<td>Driver seat location</td>
<td>$r_x=0.360$</td>
<td>$r_y=0.234$</td>
</tr>
</tbody>
</table>

![Fig. 5 Mechanism response with actual and estimated parameters](image-url)

![Fig. 6 Schematic of the vehicle model](image-url)
pute the mean of the identified parameters over all time steps, effectively averaging out the aberrant data points. In this case, we ignore the estimates for the first 3 seconds of the simulation, which are associated with relatively high objective function values as well as nonphysical parameters. To verify that the estimates are reasonable, a simulation is performed using the mean of the identified parameters in place of the actual parameters. As shown in Fig. 10, the difference between the time response obtained using the estimated parameters and that obtained using the actual parameters is negligible, again indicating the usefulness of the Lie series technique.

3.4 Challenges. Although we were able to identify the parameters in the examples presented above, several challenges are often encountered when estimating parameters in a more practical setting. In particular, we note that the issues of parameter dependence, excitation richness, unobservable states, and measurement noise can complicate the identification of parameters in practice. We briefly discuss each of these issues below.

Parameter dependence [20,21] refers to the requirement that two or more parameters maintain a particular relationship, such as a constant sum or product. Identifying such parameters can be difficult if their relationship is ignored. One possible approach is to specify bounds for the dependent parameters in the identifying algorithm. Another alternative is to account for the relationship explicitly by defining nondimensional parameter groups [22,23], thereby obtaining a minimal set of parameters for the dynamic equations. The use of dimensionless groups also avoids issues arising from incompatible dimensions in the objective function [6–8].

When generating experimental data for parameter identification purposes, it is essential that the measured output contains information relating to the parameters being sought. This requirement affects both the input that is used to excite the system, as well as the particular outputs that are measured. The input to the experimental system must be sufficiently rich so as to elicit a system response that is dependent on all the parameters being identified. A straight-line acceleration test, for example, would provide important information about the pitch dynamics of a vehicle but would be unsuitable for identifying parameters associated with vehicle roll. Appropriate inputs can be determined for systems that are linear in parameters by following some established guidelines [24–26]; however, no such guidelines exist for the general case of parameter identification for nonlinear systems.

Obtaining parameter-dependent experimental data is also hindered by the existence of unobservable states. Although we have assumed a full state measurement in the examples above, it may be difficult, expensive, or even impossible to obtain experimental data for all system states. In such cases, it may be challenging to identify parameters that only affect the unobservable states directly. A parameter and state estimator, such as the dual extended Kalman filter [27], can be used to address this issue, though the implementation of such strategies may be challenging for large systems [28].

Even if a sufficient number of states can be observed or estimated, the issue of measurement noise cannot be escaped. The effect of measurement noise on the performance of the proposed algorithm is difficult to address: since we only take one step forward in time from each experimental state, the propagation of noise through the system response depends on the system itself. The present algorithm, originally proposed by Pilipchuk and Tan [11], shows some robustness to measurement noise but, unfortunately, this characteristic cannot be guaranteed for arbitrary systems. One disadvantage of the proposed algorithm is that, in contrast to recursive gradient-descent or least-squares approaches [24,25], the outputs are not being integrated. As a result, the noise is not being averaged out. One way of addressing this shortcoming is to filter the experimental data, which poses no major computational issues in the present context since the parameters are being identified offline. Another possible strategy involves considering the data from more than one time step simultaneously. In general, this approach would demand the use of a higher number

---

### Table 3 Dynamic parameters used in the vehicle model [19]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Actual Value</th>
<th>Identified Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass, Vehicle (M)</td>
<td>730 kg</td>
<td></td>
</tr>
<tr>
<td>Mass, Driver (m_d)</td>
<td>75 kg</td>
<td></td>
</tr>
<tr>
<td>Front suspension (m_f)</td>
<td>40 kg</td>
<td></td>
</tr>
<tr>
<td>Rear suspension (m_r)</td>
<td>35.5 kg</td>
<td></td>
</tr>
<tr>
<td>Inertia-Stiffness, Vehicle</td>
<td>1230 kg m^2</td>
<td>0 kg m^2</td>
</tr>
<tr>
<td>Inertia-Stiffness, Front tire</td>
<td>175.5 kN/m</td>
<td></td>
</tr>
<tr>
<td>Inertia-Stiffness, Rear tire</td>
<td>175.5 kN/m</td>
<td></td>
</tr>
<tr>
<td>Inertia-Stiffness, Front-left suspension</td>
<td>14.81 kN/m</td>
<td>14.82 kN/m</td>
</tr>
<tr>
<td>Inertia-Stiffness, Front-right suspension</td>
<td>14.81 kN/m</td>
<td>14.77 kN/m</td>
</tr>
<tr>
<td>Inertia-Stiffness, Rear-left suspension</td>
<td>22.54 kN/m</td>
<td>22.53 kN/m</td>
</tr>
<tr>
<td>Inertia-Stiffness, Rear-right suspension</td>
<td>22.54 kN/m</td>
<td>22.59 kN/m</td>
</tr>
<tr>
<td>Damping, Seat</td>
<td>94.99 kN/m</td>
<td></td>
</tr>
<tr>
<td>Damping, Front suspension</td>
<td>1.38 kN/s/m</td>
<td></td>
</tr>
<tr>
<td>Damping, Rear suspension</td>
<td>1.12 kN/s/m</td>
<td></td>
</tr>
<tr>
<td>Damping, Seat</td>
<td>1.93 kN/s/m</td>
<td></td>
</tr>
</tbody>
</table>

---

**Fig. 7 Road profile used for vehicle simulation**
of terms in the series solution, which may pose computational difficulties for large systems. Finally, we note that the implementation of the least-squares algorithm used for minimizing the objective function in the proposed approach may also influence its robustness to measurement noise.

4 Conclusions and Future Work

This work explores the potential of using Lie series solutions for estimating parameters in multibody systems. Symbolic computational procedures are used to generate the equations of motion, the associated Lie series solutions, and the symbolic objective functions automatically. The Lie series approach presented herein can be applied to any nonlinear dynamic system governed by ODEs and has been used to estimate the parameters of a structural system, a spatial slider-crank mechanism, and an eight-degree-of-freedom vehicle model. The proposed technique was found to produce satisfactory results in all three example problems.
Two shortcomings of the present methodology can be identified. First, the Lie series approach can only be applied to ODE systems. Although DAEs can be converted into ODEs by eliminating the Lagrange multipliers symbolically, this approach may not be practical for large systems. Second, it may not be practical to construct high-order Lie series solutions for large systems, as a large amount of memory would be required to perform the necessary symbolic computation. To address these shortcomings of the Lie series approach, the authors are presently studying the application of the Taylor DAE integrator [29,30] to the problem of parameter identification. A Taylor series solution can be used in a similar manner as a Lie series solution, but the construction of the former requires neither the inversion of a symbolic coefficient matrix, nor the elimination of the Lagrange multipliers in systems governed by DAEs.

Acknowledgments

The financial support provided to C.P.V. by the Ontario Centres of Excellence (OCE), to T.U. by the Natural Sciences and Engineering Research Council of Canada (NSERC), and to J.M. by the NSERC/Toyota/Maplesoft Industrial Research Chair program is gratefully acknowledged, as are the helpful suggestions provided by Dr. Ken Butts and Dr. Chris Vermillion of the Model-Based Development group at Toyota Motor Engineering and Manufacturing North America (TEMA).

References