Solving Optimal Control Problems using SIMSCAPE® Models for State Derivatives

Technical Report UIUC-ESDL-2014-01

Daniel R. Herber*
Engineering System Design Lab
University of Illinois at Urbana-Champaign

July 20, 2014

Abstract

This technical report outlines an approach to calculate derivative functions for SIMSCAPE® models and use them to solve optimal control problems. Although this approach is less efficient than analytic expression for the derivatives, not every problem will have these directly available due to a variety of reasons, including multidomain, multibody, large-scale, automatically generated, or proprietary models. A step-by-step procedure is presented to assist utilizing this approach. The canonical Bryson-Denham state-constrained double integrator optimal control problem is used as a test optimal control problem. A number of control formulations are compared to demonstrate the computational expense of this approach compared to analytic expressions of the state derivatives and additional benefits including improved final solutions and execution time over more traditional formulations. In particular, direct transcription solutions are decidedly more efficient than the common shooting approach. Coupled with an optimal control toolbox, the user will no longer need to worry about expressing complex derivative equations or the implementation details of their optimal control problem, allowing the focus to be shifted towards solving more complex problems.

*Ph.D pre-candidate in Systems and Entrepreneurial Engineering, Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, herber1@illinois.edu

Contents

List of Figures

List of Tables

1 Motivation for using Simscape® Derivatives

Consider the following general optimal control problem with a Bolza objective:

$$\min_{\mathbf{u}} \quad \Psi = \int_{t_0}^{t_f} \mathcal{L}\left(\boldsymbol{\xi}(t), \mathbf{u}(t), t\right) dt + \mathcal{M}\left(\boldsymbol{\xi}(t_0), t_0, \boldsymbol{\xi}(t_f), t_f\right) \tag{1}$$

$$\text{subject to:} \quad \boldsymbol{\phi}_{\min} \leq \boldsymbol{\phi}\left(\boldsymbol{\xi}(t_0), t_0, \boldsymbol{\xi}(t_f), t_f\right) \leq \boldsymbol{\phi}_{\max} \tag{2}$$

$$\mathbf{C}_{\min} \leq \mathbf{C}\left(\boldsymbol{\xi}(t), \mathbf{u}(t), t\right) \leq \mathbf{C}_{\max} \tag{3}$$

$$\dot{\boldsymbol{\xi}}(t) = \mathbf{f_d}\left(\boldsymbol{\xi}(t), \mathbf{u}(t), t\right) \tag{4}$$

where $\boldsymbol{\xi}(t)$, $\mathbf{u}(t)$, and t represent, respectively, system state, control, and time [1–4]. Equation (1) is the cost functional that is to be minimized. The dynamic constraints are given in Eqn. (4). The boundary constraints are given in Eqn. (2), and the inequality path constraints are given in Eqn. (3). A classical or indirect approach for solving this type of problem is to apply optimality conditions, such as Pontryagin's maximum principle [5]. If an analytic solution to the optimality conditions cannot be found, the resulting boundary value problem can be solved numerically. Path constraints are extremely challenging to incorporate in this solution approach. Additional issues motivate alternative solution procedures [4].

Instead of finding the optimality conditions directly, we can first discretize the problem and then transcribe it to a nonlinear programming (NLP) formulation. In other words, the infinite-dimensional optimal control problem is transcribed to a finite-dimensional NLP. A shooting approach discretizes the control trajectory only and predicts the dynamics through a forward simulation [1]. A computational model such as a SIMSCAPE® model can be utilized to provide a forward simulation the dynamics of the system. The SIMULINK DESIGN OPTIMIZATION™ toolbox uses this approach to tune design parameters to improve system performance or model estimation.[1] However, this can be extremely computationally expensive and is typically unsuited for complex engineering problems such as co-design problems due to convergence issues [3, 4]. The optimal control formulation is modified to express the forward simulation of the computational model to calculate the dynamics:

$$\begin{aligned}
\min_{\mathbf{u}} \quad & \Psi = \int_{t_0}^{t_f} \mathcal{L}\left(\boldsymbol{\xi}(t), \mathbf{u}(t), t\right) dt + \mathcal{M}\left(\boldsymbol{\xi}(t_0), t_0, \boldsymbol{\xi}(t_f), t_f\right) \\
\text{subject to:} \quad & \boldsymbol{\phi}_{\min} \leq \boldsymbol{\phi}\left(\boldsymbol{\xi}(t_0), t_0, \boldsymbol{\xi}(t_f), t_f\right) \leq \boldsymbol{\phi}_{\max} \\
& \mathbf{C}_{\min} \leq \mathbf{C}\left(\boldsymbol{\xi}(t), \mathbf{u}(t), t\right) \leq \mathbf{C}_{\max} \\
\text{where:} \quad & \dot{\boldsymbol{\xi}}(t) = \mathbf{f_d}\left(\boldsymbol{\xi}(t), \mathbf{u}(t), t\right)
\end{aligned} \tag{5}$$

Another approach within the class of direct methods of optimal control is direct transcription (DT). Differing from shooting methods, DT discretizes both the state and control and the NLP algorithm simultaneously solves the system state equations and the system optimization problem, eliminating the need

[1] http://www.mathworks.com/products/sl-design-optimization/

for forward simulation [3]. Some of the advantages of DT include numerical stability, efficient solution procedures due to the problem structure and sparsity pattern, natural inclusion of path constraints, and solving singular optimal control problems [4]. A number of commercial software packages are available that solve problems using DT, typically with pseudospectral methods, including $\mathbb{GPOPS}-\mathbb{II}$ [2, 6] and *PROPT* [7]. This approach requires the optimization of both the control and state variables, while the dynamic constraints are expressed as defect constraints ($\boldsymbol{\zeta}$):

$$\begin{aligned} \min_{\boldsymbol{\xi},\mathbf{u}} \quad & \Psi = \int_{t_0}^{t_f} \mathcal{L}\left(\boldsymbol{\xi}(t), \mathbf{u}(t), t\right) dt + \mathcal{M}\left(\boldsymbol{\xi}(t_0), t_0, \boldsymbol{\xi}(t_f), t_f\right) \\ \text{subject to:} \quad & \boldsymbol{\phi}_{\min} \leq \boldsymbol{\phi}\left(\boldsymbol{\xi}(t_0), t_0, \boldsymbol{\xi}(t_f), t_f\right) \leq \boldsymbol{\phi}_{\max} \\ & \mathbf{C}_{\min} \leq \mathbf{C}\left(\boldsymbol{\xi}(t), \mathbf{u}(t), t\right) \leq \mathbf{C}_{\max} \\ & \boldsymbol{\zeta}\left(\boldsymbol{\xi}(t), \mathbf{u}(t), t\right) = \dot{\boldsymbol{\xi}}(t) - \mathbf{f_d}\left(\boldsymbol{\xi}(t), \mathbf{u}(t), t\right) = 0 \end{aligned} \tag{6}$$

This approach **requires evaluation of state derivative values**. In system optimization problems based on sophisticated system models, it may be difficult to derive the state derivative function. In the past, this has limited the type of problems that could be solved using DT. This limitation is the motivation for investigating how SIMSCAPE® models could be used as a basis for DT problem solution implementations, increasing the sophistication of models that may be used with DT.

Finding analytic or closed-form solutions for the state derivatives can be challenging or inefficient for a number of problems:

- *Multidomain* models spanning multiple energy domains can produce models that are both challenging and inefficient to find the analytic state derivatives.[2]
- *Multibody* models for mechanical systems such as robotics typically have highly nonlinear dynamics, but can be easily represented in simulation environments such as SIMMECHANICS™.[3]
- *Large-scale* models where the number of components prevents an efficient process of writing closed-form solutions for the state derivatives.
- *Automatically generated* models such as ones used in system architecture studies where the efficiency of the study requires the algorithm to operate on a large number of models with varying architectures and therefore state derivatives if DT is used [3].
- *Proprietary or closed-source* models such as some SIMSCAPE® libraries where the user is not allowed to view the analytic equations, but can run simulations.

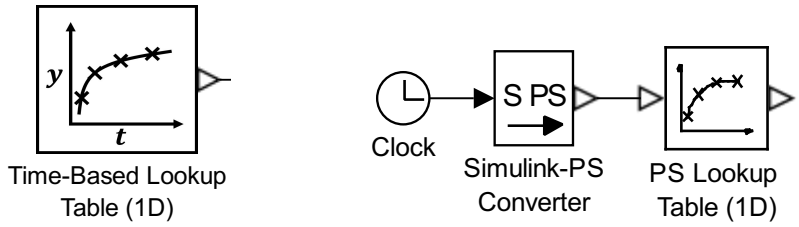
[2] http://www.mathworks.com/physical-modeling/model-multidomain-physical-systems.html
[3] http://www.mathworks.com/products/simmechanics/

This report introduces a new strategy for using sophisticated mutlidomain models with DT, presents a case study using this technique, and demonstrates the value of this solution approach in terms of numerical efficiency when compared to standard optimization methods.

2 Implementation of Simscape® Derivatives

Step 1: Create a SIMSCAPE® model (this may work with SIMULINK® models but has not been tested by the author). Include control through a custom Time-Based Lookup Table with its mask shown in Fig. 1a. This is a slightly modified version of the PS Lookup Table (1D) with the input value along the first direction valued as the internal time variable of the `tablelookup` function[4]:

```
O == tablelookup(x_t, y_t, time);
```



(a) Custom time-based lookup table. **(b)** SIMSCAPE® foundation equivalent.

Figure 1 Blocks for adding open loop control to SIMSCAPE® models.

Step 2: Determine the ordering of the input vector for the `model` command by observing the results from a test run.[5] One method to accomplish is this using the Performance Advisor. The Performance Advisor can be accessed through Simulink Editor, select `Analysis>Performance Tools>Performance Advisor` or by the following command:[6]

```
performanceadvisor(p.model)
```

Once the Performance Advisor is open, select `Check to view baseline signals and set their tolerances` in the `Create Baseline` test. Then run the test and observe the ordering of the signals. This also gives you the signal names. Refer to the page linked to in the footnote if the test will not run.[7]

Step 3: Execute the compilation phase of the model in p.model before the optimization program is called.

```
eval([p.model,'([],[],[],''compile'');'])
```

[4] http://www.mathworks.com/help/physmod/simscape/lang/tablelookup.html
[5] http://www.mathworks.com/help/simulink/slref/model_cmd.html
[6] http://www.mathworks.com/help/simulink/ug/getting-started-with-performance-advisor.html
[7] http://www.mathworks.com/help/simulink/slref/simulink-checks_bth9tg2-2.html

Step 4: From inside the function that requires derivative values (such as the nonlinear constraint function), extract the states and control matrices from the optimization vector. This function will be problem-specific.

```
[t,uMat,sMat] = extractStatesControl(x,p);
```

Step 5: Initialize the derivative matrix where the row number corresponds to the time index and the column number correspond to either the state or control index. The control needs to be added since the SIMSCAPE® model has states for the custom lookup table. Then a loop is created to add the derivative values row by row. Inside this loop, the input vector needs to be specified, xIn (this is also problem specific and one must ensure the states and control are ordered properly, see Step 2). Finally, the `model` command is called using with the flag set to output the derivatives.

```
fdMat = zeros(p.nt,p.ns+p.nc);
for i = 1:p.nt
  xIn = [sMat(i,:),uMat(i,:)];
  fdMat(i,:) = eval([p.model,'(','t(i),','xIn,','[],''derivs'');']);
end
```

Step 6: Next, remove control derivatives since they are always equal to zero and will not be needed where p.uLoc is the locations in xIn of the control variables:

```
fdMat(:,p.uLoc) = [];
```

Step 7: Finally, terminate the model using the `model` command so that the model can be used in the future.

```
eval([p.model,'([],[],[],''term'');'])
```

Shown below are some sample commands for the SIMSCAPE® model named BrysonDenhamModel:

```
BrysonDenhamModel([], [], [], 'compile')              % step 3
fdMat(i,:) = BrysonDenhamModel(t(i),xIn,[],'derivs'); % step 5
BrysonDenhamModel([], [], [], 'term')}                % step 7
```

3 Bryson-Denham State-Constrained Double Integrator Optimal Control Problem

3.1 Problem Statement

The Bryson-Denham state-constrained double integrator optimal control problem is a simple canonical test problem that will be used to demonstrate optimal control implementations that utilize SIMSCAPE® derivatives. A fully specified Bryson-Denham problem is formulated as:

$$\min_{u(t)} \quad \frac{1}{2}\int_0^1 u^2 dt \tag{7}$$

subject to:

$$\dot{\boldsymbol{\xi}}(t) = \begin{bmatrix}\dot{x}\\ \dot{v}\end{bmatrix} = \begin{bmatrix}v\\ u\end{bmatrix} \tag{8}$$

$$\boldsymbol{\xi}(0) = \begin{bmatrix}x(0)\\ v(0)\end{bmatrix} = \begin{bmatrix}0\\ 1\end{bmatrix} \tag{9}$$

$$\boldsymbol{\xi}(1) = \begin{bmatrix}x(1)\\ v(1)\end{bmatrix} = \begin{bmatrix}0\\ -1\end{bmatrix} \tag{10}$$

$$x(t) \leq \ell = \frac{1}{9} \tag{11}$$

where Eqn. (7) is the objective function, Eqn. (8) are the dynamic constraints, Eqns. (9) and (10) are the boundary conditions, and Eqn. (11) is the state-inequality path constraint. This dynamic system is analogous to moving a point mass. The closed-form solution to this problem when $0 \leq \ell \leq \frac{1}{6}$ is presented in Ref. [5, p. 122]:

$$u(t) = \begin{cases} -\frac{2}{3\ell}\left(1-\frac{t}{3\ell}\right) & 0 \leq t \leq 3\ell \\ 0 & 3\ell \leq t \leq 1-3\ell \\ -\frac{2}{3\ell}\left(1-\frac{1-t}{3\ell}\right) & 1-3\ell \leq t \leq 1 \end{cases} \tag{12}$$

$$v(t) = \begin{cases} \left(1-\frac{t}{3\ell}\right)^2 & 0 \leq t \leq 3\ell \\ 0 & 3\ell \leq t \leq 1-3\ell \\ -\left(1-\frac{1-t}{3\ell}\right)^2 & 1-3\ell \leq t \leq 1 \end{cases} \tag{13}$$

$$x(t) = \begin{cases} \ell\left(1-\left(1-\frac{t}{3\ell}\right)^3\right) & 0 \leq t \leq 3\ell \\ \ell & 3\ell \leq t \leq 1-3\ell \\ \ell\left(1-\left(1-\frac{1-t}{3\ell}\right)^3\right) & 1-3\ell \leq t \leq 1 \end{cases} \tag{14}$$

The value of the cost functional at the optimal solution is:

$$\Psi = \frac{4}{9\ell} \tag{15}$$

3.2 Simscape® Model

The SIMSCAPE® model for this example is displayed in Fig. 2. The Time-Based Lookup Table (1-D) block is used for the control and is fed into an ideal Force Source that acts on a Mass block. An additional sensor is required to log the position of the mass. Finally the SIMSCAPE® Solver block is added to solve the problem. The Simulation Data Inspector output for this model is shown in Fig. 3. As explained in Sec. 2 Step 2, the ordering of the states and control is based on this output.

3.3 Results

This problem has been solved previously using a number of optimal control software packages including $\mathbb{GPOPS}-\mathbb{II}$[8] and *PROPT*[9], along with the closed-form solution. Six different methods were employed here to solve the problem. The first 2 methods (M1 and M2) used single shooting (S) with 101 equally spaced optimization variables to represent the control trajectory. The former used an analytic derivative function with `ode23tb` as the solver. The latter directly used the SIMSCAPE® model to determine the states.

The other four methods use DT. M3 and M4 use trapezoidal (T) collocation with 101 equally spaced collocation points. These methods differ by the calculation method for the derivatives: M3 uses analytic expressions, i.e., Eqn. (8), and M4 uses SIMSCAPE® derivatives. M5 and M6 use the software package $\mathbb{GPOPS}-\mathbb{II}$[10] (a commercial implementation of pseudospectral (PS) methods, which are a subclass of DT) [2]. Again the former uses analytic expressions and the latter uses SIMSCAPE® derivatives.

The results are shown in Fig. 5 and summarized in Table 1. Figure 6 plots the absolute error between the method and the closed-form solution in Eqns. (12) – (14). All the methods were able to find a similar result to the closed-form solution, but a number of differences were noted.

3.3.1 Use of the Derivative Function

Each method had different levels of use of the available derivative function. First, S methods tend to need the derivative function more often because for each perturbation in the optimization algorithm, the entire time horizon needs to be simulated (which requires lots of calls to the derivative function). M1 and M3 had nearly the same number of derivative function calls and total time spent calculating the derivatives. Therefore, the time per derivative function call was nearly the same between the methods, implying that the analytic derivative function was only slightly faster to call than the SIMSCAPE® version (1.2×10^{-4} s compared to 1.3×10^{-4} s). The PS methods required fewer derivative function

[8] `http://www.gpops2.com/Examples/Bryson-Denham/Bryson-Denham.html`

[9] `http://tomopt.com/docs/propt/tomlab_propt018.php`

[10] **Warning!** $\mathbb{GPOPS}-\mathbb{II}$ is not provided with this submission but can be purchased from `http://www.gpops2.com/Purchase/Purchase.html`. M5 and M6 will produce errors if the files are not located in the path.

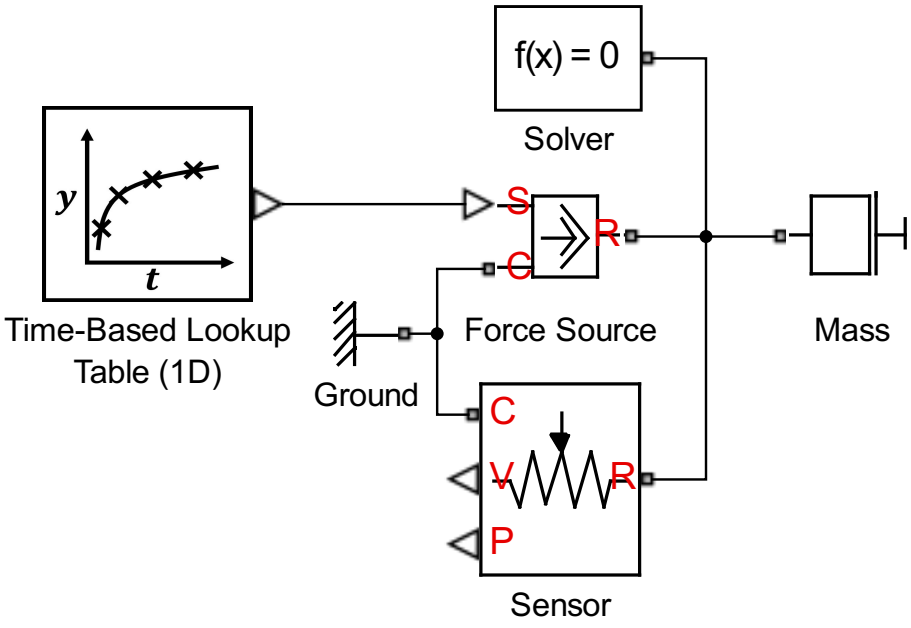


Figure 2 SIMSCAPE® model (BrysonDenhamModel.slx).

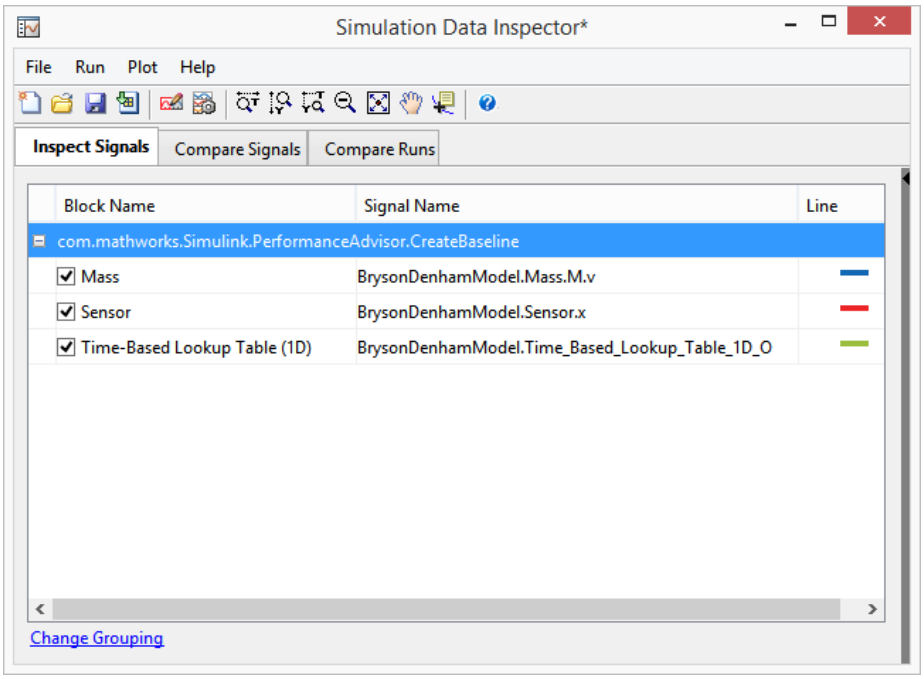


Figure 3 Simulation Data Inspector for viewing the internal state and control ordering for the Bryson-Denham problem.

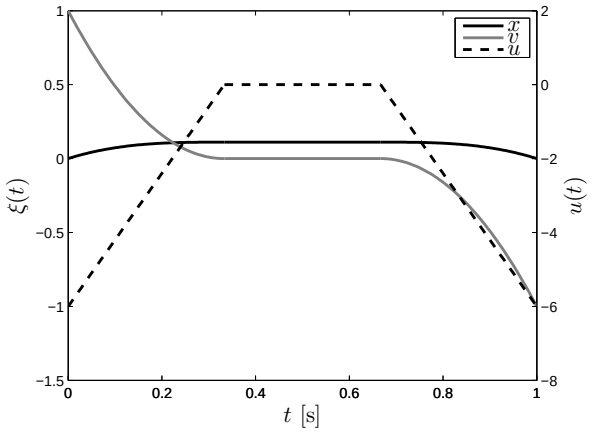


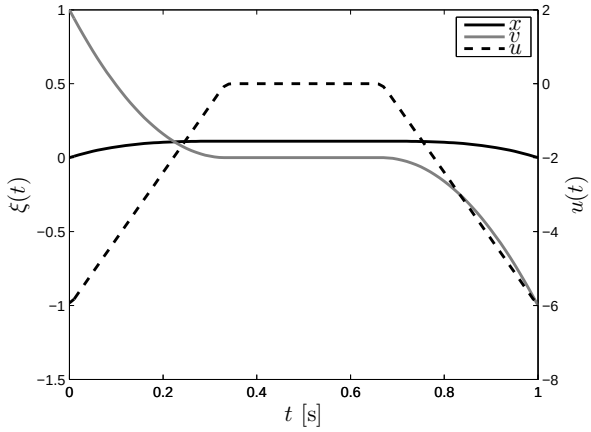
Figure 4 Closed-form solution from Ref. [5].

calls than either S or T, even with mesh refinement (discussed in the next section).

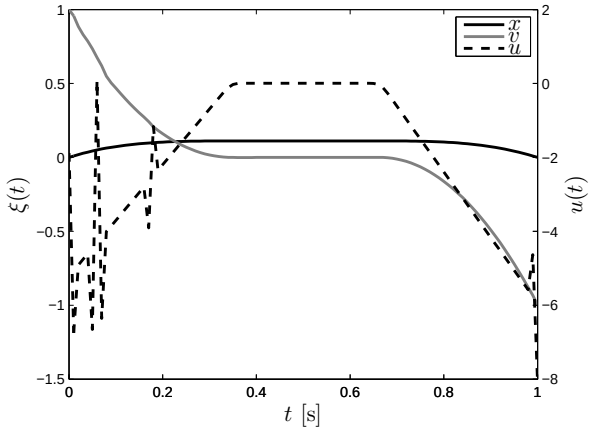
A major factor explaining the number of derivative calls required was the vectorization of the analytic derivatives. Vectorization uses matrix and vector operations to produce the desired calculation instead of loop-based, scalar-oriented code. Vectorization is typically more computationally efficient than loops. This is evident in the number of calls needed to the analytic derivative function compared to the SIMSCAPE® derivative function for the same control implementation (3344 compared to 337744 and 396 compared to 7151). Because a basic `for` loop was used, the parallel computing resources of the machine were not utilized when calculating the SIMSCAPE® derivatives. Using a `parfor` could be more efficient, but parallel compiled builds of the model did not appear functional for the `model` command. This factor would be even more influential for more complex problems and derivative functions. Running parallel simulations can also improve the computation time of a S method but does not reduce the number of derivative function calls.

3.3.2 Total Time

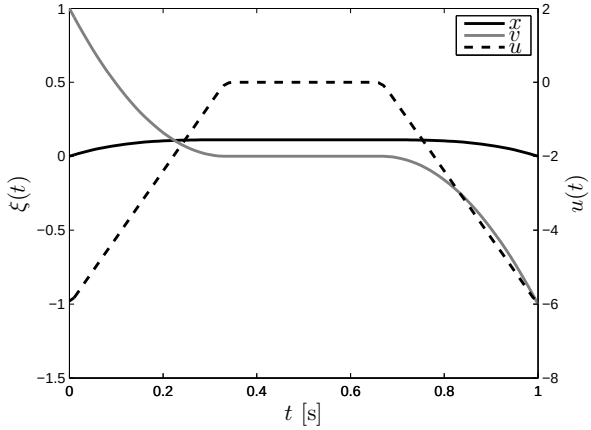
The total time to solve the optimization problem varied greatly between the solutions. By far the slowest was the shooting methods (M1, M2). The methods based on trapezoidal collocation (M3, M4) were faster, as were the methods using $\mathbb{GPOPS} - \mathbb{II}$ (M5, M6). The SIMSCAPE® derivatives were approximately 4541× slower compared to analytic derivatives for M3 and M4, while this approach was only 503× slower for M5 and M6. Even with this difference, overall computation time was only 32× and 2.5× slower using SIMSCAPE® derivatives, still making this an attractive approach. Most importantly, the DT solutions that use SIMSCAPE® derivatives (M4, M6) were 16× and 646× faster than the



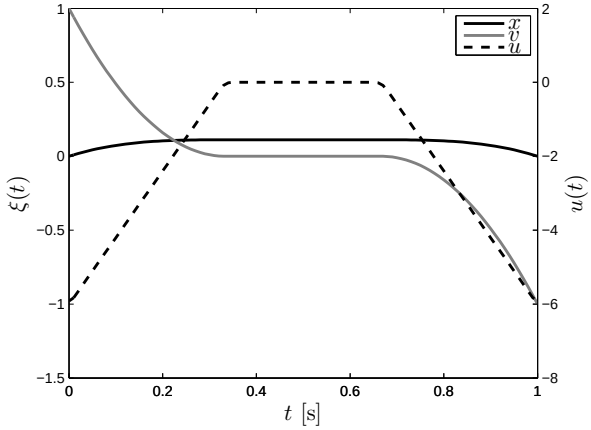
(a) Results for shooting method with analytic derivatives (method = 1).



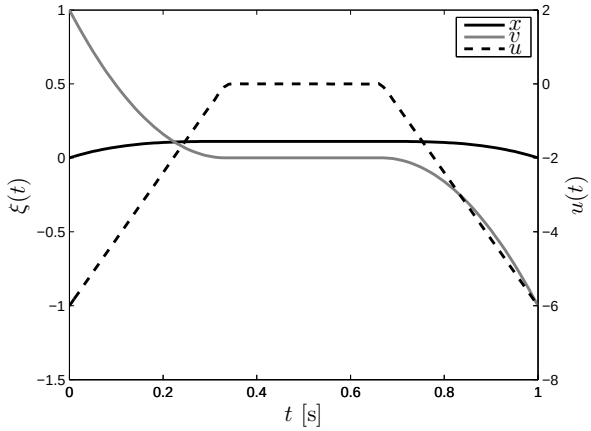
(b) Results for shooting method with a SIMSCAPE® model (method = 2).



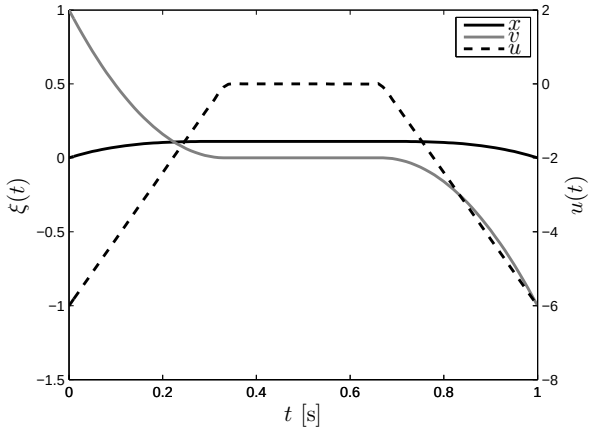
(c) Results for direct transcription with trapezoidal rule and analytic derivatives (method = 3).



(d) Results for direct transcription with trapezoidal rule and SIMSCAPE® derivatives (method = 4).

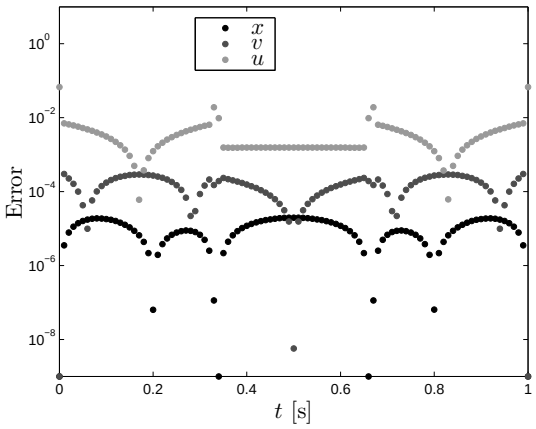


(e) Results for pseudospectral method with analytic derivatives (method = 5).

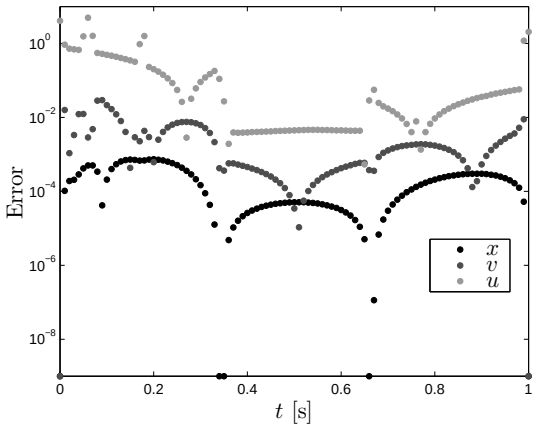


(f) Results for pseudospectral method with SIMSCAPE® derivatives (method = 6).

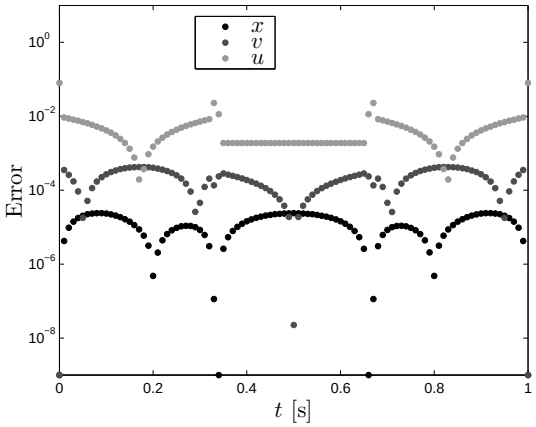
Figure 5 Results for each method.



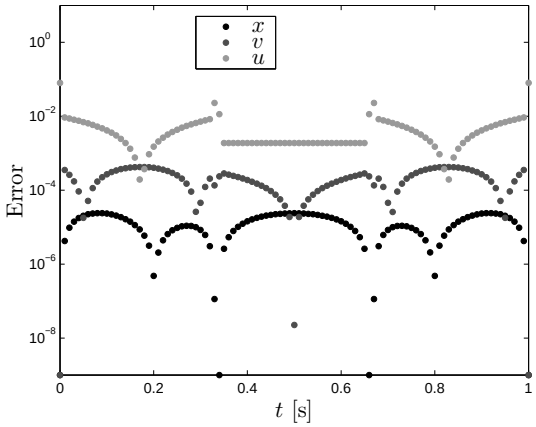
(a) Error for shooting method with analytic derivatives (method = 1).



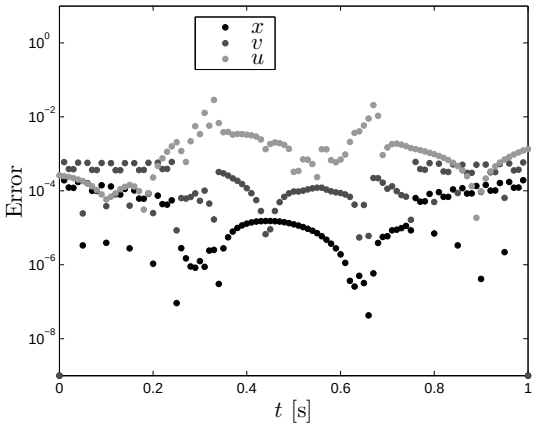
(b) Error for shooting method with SIMSCAPE® model (method = 2).



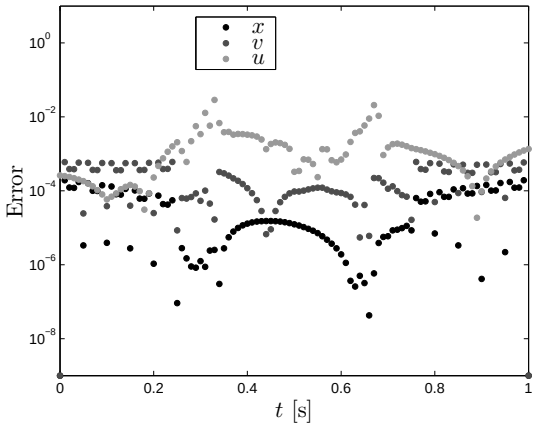
(c) Error for direct transcription with trapezoidal rule and analytic derivatives (method = 3).



(d) Error for direct transcription with trapezoidal rule and SIMSCAPE® derivatives (method = 4).



(e) Error for pseudospectral method with analytic derivatives (method = 5).



(f) Error for pseudospectral method with SIMSCAPE® derivatives (method = 6).

Figure 6 Absolute error relative to the closed-form solution for each method.

S approach that used the SIMSCAPE® model (M2)!

One contributing factor to the difference between T and PS was the large number of time points used in the T formulation. Since no mesh accuracy calculations were performed, the mesh needed to be dense enough to implicitly satisfy a specified mesh tolerance (i.e., trapezoidal collocation is accurate enough at the mesh resolution of 10^{-6}). On the other hand, $\mathbb{GPOPS} - \mathbb{II}$ utilizes adaptive mesh refinement, and it therefore efficiently provides a mesh for the problem through iterative solution. More optimization variables increase the required number of derivative function calls, explaining why $\mathbb{GPOPS} - \mathbb{II}$ is more efficient than DT with trapezoidal collocation.

3.3.3 Absolute Error Relative to Closed-Form Solution

Here we will discuss the differences between the final solution found for each method and the closed-from solution in Eqns. (12) – (14). First, most of the control trajectories are smooth with the exception of M2 (which directly uses the SIMSCAPE® model). Shooting methods tend to have these stability issues [1]. One might expect M1 and M2 to have the same results, but the differences are most likely due to the differences in solver parameters. The solvers used in SIMSCAPE® may have different settings to improve robustness or solution efficiency at the cost of accuracy. Since this is a fairly simple optimal control problem, the S methods perform reasonably well, but this is not the general case [3].

Many of the plots in Fig. 5 are similar because the results are near the closed-form solution. However, Fig. 6 shows that each method has different absolute error patterns and magnitude. First note that (M3, M4) and (M5, M6) have the same errors since the SIMSCAPE® derivatives were the exact same as the analytic derivatives (demonstrating that these approaches are equivalent). M2 had the largest error, especially the control values. The errors for M1, M3, and M4 were all very similar in both magnitude and pattern. The errors for each of these methods can be approximately summarized by $e_u \approx 10^{-3}$, $e_v \approx 10^{-4}$, and $e_x \approx 10^{-5}$. The error pattern for the methods which use a PS method were decidedly more chaotic. The magnitude of the errors were similar when the path constraint was active while the control error was much smaller in the other region. Therefore, the PS method was more accurate, but this is due to the fact that mesh refinement was used. Overall on a pointwise basis, each method found a reasonably close solution.

3.3.4 Final State Error Using `ode15s`

The resulting optimal control trajectory for each method was simulated using the `ode15s` to observe the practical effectiveness of each solution result. This was measured by comparing the final states from the simulation to the final desired states described in the problem. `ode15s` is commonly used for stiff systems and is an adaptive solver, so the order of accuracy will be higher than the implicit trapezoidal rule previously used in M1–M4. The errors are shown

Table 1 Comparison between the methods.

		$\dot{\xi}$ (derivative function)					error		
#		type	calls	time (s)	time/call	total (s)	f	x_f	v_f
1	S	analytic	399434	49.040	1.2×10^{-4}	83.06	4.003	-1.5×10^{-4}	1.3×10^{-4}
2	S	SIMSCAPE	826678	753.888	9.1×10^{-4}	757.55	3.752	9.1×10^{-2}	1.0×10^{-1}
3	T	analytic	3344	0.010	3.0×10^{-6}	1.53	4.004	4.1×10^{-4}	3.1×10^{-3}
4	T	SIMSCAPE	337744	45.411	1.3×10^{-4}	48.62	4.004	4.1×10^{-4}	3.1×10^{-3}
5	PS	analytic	396	0.002	5.1×10^{-6}	0.47	4.000	-6.3×10^{-4}	6.4×10^{-4}
6	PS	SIMSCAPE	7151	1.005	1.4×10^{-4}	1.17	4.000	-6.3×10^{-4}	6.4×10^{-4}

in Table 1. Each method, except for M2, perform well (i.e., errors less the 4×10^{-3}). M2 had error values that were two orders of magnitude worse. Even though the solution for M2 looked similar to the closed-form solution, the result was simply not accurate enough (not forgetting the extremely long computation time).

3.3.5 Objective Function Values

Even though each method had the same base problem statement, they produced slightly different values for the objective function (and optimization variables). The closed-form optimal cost is 4 (see Eqn. (15)). With the same optimization algorithm tolerances, M2 had a final objective function value of 3.752 (lower than the closed-form solution). This is because the solver was not accurate enough when calculating the states, and therefore produced a solution that is not feasible with a higher-order method (see Sec. 3.3.3 and 3.3.4). In a sense, this solution approach capitalized on the implementation inaccuracy. The other methods that proved to be more accurate had objective function values between 4.000 and 4.004. $\mathbb{GPOPS} - \mathbb{II}$ methods produced the lower value of 4.000 (nearly the same as the closed-form cost).

3.3.6 Convergence Behavior

Differences in the convergence behavior between each method can also be observed in the command window outputs (see App. A). The S methods require at least two major iterations to find a feasible solution while DT using T only requires one major iteration. Generally, DT requires fewer iterations to converge to both a feasible and optimal solution (recall that this was initially posed as a infinite-dimensional problem in Sec. 3.1). Additional behavior, such as the decrease in mesh error through the mesh refinement process, can be seen for M5 and M6.

4 Concluding Remarks

This technical report outlined an approach to calculate derivative functions for SIMSCAPE® models and use them to solve optimal control problems. Although this approach is less efficient than analytic expression for the derivatives, not every problem will have these directly available due to a variety of reasons. Coupled with an optimal control toolbox such as $\mathbb{GPOPS} - \mathbb{II}$, the user will no longer need to worry about expressing complex derivative equations or the implementation details of their optimal control problem, allowing the focus to be shifted towards solving more complex problems that demand the use of sophisticated dynamic system models.

References

[1] BETTS, J T. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming,* 2nd edition, SIAM, 2010.
DOI: 10.1137/1.9780898718577

[2] PATTERSON, M A and A V RAO. $\mathbb{GPOPS} - \mathbb{II}$ *Version 2.0: A General-Purpose MATLAB Software for Solving Multiple-Phase Optimal Control Problems*, May 2014.
URL: http://www.gpops2.com/resources/gpops2QuickReference.pdf

[3] ALLISON, J T and D R HERBER. "Multidisciplinary Design Optimization of Dynamic Engineering Systems." *AIAA Journal.* ***52*** (4), Apr. 2014, pp. 691–710.
DOI: 10.2514/1.J052182

[4] HERBER, D R. "Dynamic System Design Optimization of Wave Energy Converters Utilizing Direct Transcription." M.S. Thesis. University of Illinois at Urbana-Champaign: Urbana-Champaign, IL, USA, May 2014.
URL: http://systemdesign.illinois.edu/publications/Her14a.pdf

[5] BRYSON, A E and Y-C HO. *Applied Optimal Control: Optimization, Estimation, and Control,* revised edition, Taylor & Francis, 1975.
ISBN: 0891162283

[6] PATTERSON, M A and A V RAO. "$\mathbb{GPOPS} - \mathbb{II}$: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using *hp*–Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming." *ACM Transactions on Mathematical Software.* ***39*** (3), July 2013, pp. 1–41.
URL: http://vdol.mae.ufl.edu/JournalPublications/TOMS-GPOPS-II-August-2013.pdf

[7] RUTQUIST, P E and M M EDVALL. *PROPT - Matlab Optimal Control Software*, Tomlab Optimization, Apr. 2010.
URL: http://tomopt.com/docs/TOMLAB_PROPT.pdf

Acknowledgments

I would like to thank Assistant Professor James Allison[11] at the University of Illinois at Urbana-Champaign who provided much assistance when preparing this report. Also thanks to Jason McDonald for his feedback.

A Command Window Outputs

A.1 method = 1

Shooting method with analytic derivatives (only select portions are shown).

```
                                                        Norm of First-order
 Iter F-count            f(x) Feasibility  Steplength        step  optimality
    0     102    2.000000e+00   1.389e-01                          2.000e-02
    1     204    4.022316e+00   6.260e-07   1.000e+00   2.021e+01   3.901e+00
    2     306    4.021748e+00   9.875e-11   1.000e+00   1.426e-02   9.785e-03
    3     408    4.020783e+00   5.349e-11   1.000e+00   6.950e-02   9.542e-03
    4     510    4.016638e+00   6.871e-10   1.000e+00   3.247e-01   8.133e-03
    5     612    4.005824e+00   7.582e-10   1.000e+00   1.294e+00   3.621e-03
    6     714    4.002948e+00   2.699e-09   1.000e+00   9.924e-01   1.138e-03
    7     816    4.002792e+00   6.548e-11   1.000e+00   4.401e-02   1.108e-03
    8     918    4.002630e+00   1.004e-10   1.000e+00   3.917e-02   7.249e-04
    9    1020    4.002628e+00   2.074e-11   1.000e+00   1.651e-02   1.946e-04
   10    1122    4.002628e+00   6.841e-11   1.000e+00   9.352e-03   1.886e-04
   11    1224    4.002627e+00   4.420e-11   1.000e+00   4.354e-03   1.836e-04
   12    1326    4.002626e+00   3.196e-10   1.000e+00   1.247e-02   1.573e-04
   13    1428    4.002624e+00   1.962e-10   1.000e+00   1.241e-02   1.132e-04
   14    1530    4.002620e+00   1.128e-10   1.000e+00   1.661e-02   8.599e-05
   15    1632    4.002619e+00   4.964e-10   1.000e+00   1.608e-02   4.178e-05
   16    1734    4.002618e+00   7.280e-12   1.000e+00   1.024e-02   6.292e-06
   17    1836    4.002618e+00   1.146e-11   1.000e+00   2.441e-03   1.630e-06
   18    1938    4.002618e+00   1.555e-12   1.000e+00   2.876e-04   3.731e-07

Local minimum found that satisfies the constraints.
```

[11] http://www.mathworks.com/matlabcentral/fileexchange/authors/157997

A.2 method = 2

Shooting method with a SIMSCAPE® model (only select portions are shown).

```
                                                          Norm of  First-order
 Iter F-count            f(x) Feasibility  Steplength        step  optimality
    0     102    2.000000e+00   1.389e-01                          2.000e-02
    1     204    3.822289e+00   1.135e-06   1.000e+00   1.978e+01   4.472e+00
    2     306    3.821157e+00   4.310e-07   1.000e+00   3.373e-02   2.039e-02
    3     408    3.815691e+00   7.342e-06   1.000e+00   1.671e-01   1.937e-02
    4     510    3.796440e+00   2.085e-09   1.000e+00   6.825e-01   1.519e-02
    5     612    3.766875e+00   1.064e-05   1.000e+00   2.479e+00   8.630e-03
    6     714    3.763901e+00   1.927e-09   1.000e+00   1.710e-01   8.187e-03
    7     816    3.759692e+00   1.277e-09   1.000e+00   2.407e-01   6.266e-03
    8     918    3.757821e+00   1.233e-09   1.000e+00   2.775e-01   5.804e-03
    9    1020    3.757696e+00   1.100e-09   1.000e+00   5.530e-02   5.717e-03
   10    1122    3.757368e+00   8.846e-10   1.000e+00   7.190e-02   5.680e-03
   11    1224    3.756414e+00   1.193e-09   1.000e+00   2.637e-01   5.702e-03
   12    1326    3.755500e+00   2.155e-09   1.000e+00   2.248e-01   5.116e-03
   13    1428    3.754423e+00   6.132e-10   1.000e+00   3.102e-01   3.801e-03
   14    1530    3.753618e+00   1.038e-06   1.000e+00   4.096e-01   2.015e-03
   15    1632    3.753450e+00   1.187e-09   1.000e+00   1.282e-01   1.632e-03
   16    1734    3.753290e+00   1.756e-07   1.000e+00   9.897e-02   1.869e-03
   17    1836    3.753064e+00   1.616e-09   1.000e+00   8.002e-02   2.064e-03
   18    1938    3.752627e+00   1.570e-09   1.000e+00   2.042e-01   1.656e-03
   19    2040    3.752412e+00   2.576e-09   1.000e+00   1.163e-01   9.309e-04
   20    2142    3.752350e+00   1.830e-07   1.000e+00   9.293e-02   8.962e-04
   21    2244    3.752333e+00   1.435e-10   1.000e+00   4.548e-02   6.774e-04
   22    2346    3.752318e+00   4.193e-10   1.000e+00   3.024e-02   6.711e-04
   23    2448    3.752297e+00   1.424e-09   1.000e+00   1.291e-02   6.128e-04
   24    2550    3.752260e+00   2.025e-10   1.000e+00   6.215e-02   3.991e-04
   25    2652    3.752239e+00   5.026e-10   1.000e+00   7.208e-02   1.671e-04
   26    2754    3.752236e+00   3.087e-10   1.000e+00   1.825e-02   1.195e-04
   27    2856    3.752235e+00   8.116e-11   1.000e+00   1.099e-02   2.089e-04
   28    2959    3.752235e+00   1.570e-11   1.000e+00   1.760e-03   1.277e-04
   29    3061    3.752235e+00   9.367e-12   1.000e+00   2.584e-03   1.233e-04
   30    3163    3.752234e+00   1.130e-10   1.000e+00   1.009e-02   8.765e-05

                                                          Norm of  First-order
 Iter F-count            f(x) Feasibility  Steplength        step  optimality
   31    3265    3.752233e+00   6.497e-11   1.000e+00   8.868e-03   2.354e-04
   32    3368    3.752232e+00   1.406e-10   1.000e+00   1.758e-02   1.032e-04
   33    3470    3.752232e+00   3.953e-11   1.000e+00   8.640e-03   3.410e-04
   34    3573    3.752232e+00   7.088e-12   1.000e+00   3.086e-03   6.514e-05
   35    3675    3.752232e+00   6.678e-12   1.000e+00   1.927e-03   6.675e-05
   36    3777    3.752232e+00   2.395e-12   1.000e+00   9.156e-04   6.734e-05
   37    3879    3.752232e+00   2.409e-12   1.000e+00   8.536e-04   6.681e-05
   38    3981    3.752232e+00   5.714e-12   1.000e+00   1.679e-03   6.874e-04
   39    4012    3.752232e+00   5.714e-12   1.578e-05   3.191e-06   6.874e-04

Local minimum possible. Constraints satisfied.
```

A.3 method = 3,4

Direct transcription with trapezoidal rule (only select portions are shown). Both analytic and SIMSCAPE® derivatives have the same output.

```
                                                                  Norm of First-order
 Iter F-count            f(x) Feasibility  Steplength        step  optimality
    0     304    2.000000e+00   2.000e+00                          2.000e-02
    1     608    4.022445e+00   3.610e-11   1.000e+00   2.291e+01   3.932e+00
    2     912    4.022250e+00   7.180e-13   1.000e+00   1.397e-02   9.659e-03
    3    1216    4.021296e+00   5.454e-12   1.000e+00   6.951e-02   9.410e-03
    4    1520    4.017008e+00   1.525e-11   1.000e+00   3.386e-01   8.199e-03
    5    1824    4.005247e+00   1.610e-10   1.000e+00   1.474e+00   2.926e-03
    6    2128    4.003568e+00   7.349e-11   1.000e+00   8.056e-01   2.619e-04
    7    2432    4.003567e+00   4.930e-14   1.000e+00   9.386e-04   2.594e-04
    8    2736    4.003563e+00   1.732e-13   1.000e+00   4.484e-03   2.466e-04
    9    3040    4.003547e+00   1.247e-12   1.000e+00   2.131e-02   1.852e-04
   10    3344    4.003527e+00   3.463e-12   1.000e+00   6.444e-02   1.927e-07

Local minimum found that satisfies the constraints.
```

A.4 method = 5,6

Pseudospectral method (only select portions are shown). Both analytic and SIMSCAPE® derivatives have the same output.

```
|---------------------------------- Mesh iteration 1 ----------------------------------|

Nonlinear constraints      10     Linear constraints       1
Nonlinear variables        17     Linear variables         0
Jacobian  variables        17     Objective variables      7
Total constraints          11     Total variables         17

Major Minors     Step   nCon Feasible  Optimal  MeritFunction    nS Penalty
    0      8              1  9.5E-01  1.0E+00  1.4073667E+05     1           r i
    1      9  2.9E-01     2  4.7E-01  1.1E+00  9.0261036E+04     1         n rli
    2      1  3.5E-01     4  2.4E-01  8.7E-01  5.6285668E+04     1         n rli
    3      2  1.0E+00     5  1.4E-01  1.9E-01  2.6076911E+02     1         s
    4      1  1.0E+00     6  1.0E-03  6.0E-03  4.2402145E+00     1
    5      2  1.0E+00     7  4.0E-06  7.1E-03  4.1701570E+00     1
    6      1  1.0E+00     8  7.2E-05  6.6E-03  4.1430905E+00     1
    7      1  1.0E+00     9  1.2E-02  2.7E-03  4.0105429E+00     1
    8      1  1.0E+00    10  2.1E-03  1.4E-04  3.9981039E+00     1
    9      1  1.0E+00    11  5.1E-06  2.8E-06  3.9977648E+00     1
   10      1  1.0E+00    12 (4.2E-09)(9.1E-07) 3.9977647E+00     1

SNOPTA EXIT   0 -- finished successfully
SNOPTA INFO   1 -- optimality conditions satisfied

--- Analysis of Mesh in Phase 1 ---

Maximum Relative Error on Current Mesh in Phase 1 = 0.0004596
Mesh Error Tolerance is NOT satisfied in Phase 1

--- Modification of Mesh in Phase 1 ---
```

```
 Dividing Mesh Interval 1 Into 4 Mesh Intervals

 |---------------------------------- Mesh iteration 2 ----------------------------------|

 Nonlinear constraints      26     Linear constraints       1
 Nonlinear variables        41     Linear variables         0
 Jacobian  variables        41     Objective variables      7
 Total constraints          27     Total variables         41

 Major Minors     Step   nCon Feasible  Optimal  MeritFunction    nS Penalty
     0     19              1  1.9E-02  5.4E-03  3.9977647E+00     7           r
     1      1  1.0E+00      2  5.9E-03  9.0E-03  4.0042235E+00     7 5.8E+00 n r
     2      2  1.0E+00      3 (1.4E-07) 8.2E-04  4.0041105E+00     6 1.0E+01 s
     3      2  2.9E-02      6  1.4E-04  1.2E-03  4.0030091E+00     5 6.5E+00
     4      3  1.2E-01      8  2.9E-04  7.8E-04  4.0019035E+00     5 6.5E+00
     5      6  1.0E-01     10  8.2E-04  2.8E-04  4.0001191E+00     6 6.5E+00
     6      2  8.0E-02     12  1.0E-03  3.1E-04  3.9991627E+00     5 6.5E+00
     7      2  8.7E-02     14  1.0E-03  2.2E-04  3.9986885E+00     6 6.5E+00
     8      2  7.9E-02     16  9.7E-04  1.1E-04  3.9983219E+00     7 6.5E+00
     9      1  6.7E-02     18  9.3E-04  1.2E-04  3.9981921E+00     7 6.5E+00
    10      1  1.0E+00     19  2.4E-05  4.9E-05  3.9973743E+00     7 6.5E+00
    11      1  1.0E+00     20  1.6E-06 (1.0E-06) 3.9973639E+00     7 6.5E+00
    12      1  1.0E+00     21 (1.4E-09)(1.6E-07) 3.9973639E+00     7 6.5E+00   R

 SNOPTA EXIT   0 -- finished successfully
 SNOPTA INFO   1 -- optimality conditions satisfied

 --- Analysis of Mesh in Phase 1 ---

 Maximum Relative Error on Current Mesh in Phase 1 = 0.00029249
 Mesh Error Tolerance is NOT satisfied in Phase 1

 --- Modification of Mesh in Phase 1 ---

Dividing Mesh Interval 2 Into 4 Mesh Intervals
Dividing Mesh Interval 3 Into 4 Mesh Intervals

 |---------------------------------- Mesh iteration 3 ----------------------------------|

 Nonlinear constraints      62     Linear constraints       1
 Nonlinear variables        95     Linear variables         0
 Jacobian  variables        95     Objective variables      7
 Total constraints          63     Total variables         95

 Major Minors     Step   nCon Feasible  Optimal  MeritFunction    nS Penalty
     0     31              1  1.7E-03  5.4E-04  3.9973639E+00    25           r
     1      1  1.0E+00      2  1.8E-04  3.3E-04  4.0029786E+00    25 1.2E+02 n r
     2      2  1.0E+00      3 (4.4E-08) 4.7E-04  4.0029258E+00    24 2.0E+01 s
     3      2  1.0E+00      4  4.4E-04  8.8E-05  4.0002089E+00    25 1.9E+01
     4      1  1.0E+00      5  4.0E-05  1.2E-04  4.0001110E+00    25 1.9E+01
     5      1  1.0E+00      6  9.0E-06  3.1E-05  4.0000063E+00    25 1.9E+01
     6      1  1.0E+00      7  7.6E-06  2.7E-05  3.9999461E+00    25 1.9E+01
     7      1  1.0E+00      8  1.8E-06  1.2E-05  3.9999349E+00    25 1.9E+01
     8      1  1.0E+00      9 (1.7E-07) 2.1E-06  3.9999340E+00    25 1.9E+01
     9      1  1.0E+00     10 (5.9E-09)(6.8E-07) 3.9999339E+00    25 1.9E+01

 SNOPTA EXIT   0 -- finished successfully
```

```
 SNOPTA INFO   1 -- optimality conditions satisfied

 ---- Analysis of Mesh in Phase 1 ----

Maximum Relative Error on Current Mesh in Phase 1 = 4.5676e-06
Mesh Error Tolerance is NOT satisfied in Phase 1

 ---- Modification of Mesh in Phase 1 ----

Dividing Mesh Interval 3 Into 2 Mesh Intervals
Dividing Mesh Interval 8 Into 2 Mesh Intervals

 |---------------------------------- Mesh iteration 4 ------------------------------------|

 Nonlinear constraints      74     Linear constraints       1
 Nonlinear variables       113     Linear variables         0
 Jacobian  variables       113     Objective variables      7
 Total constraints          75     Total variables        113

 Major Minors     Step   nCon Feasible  Optimal  MeritFunction    nS Penalty
     0     29              1  7.0E-05  6.9E-05  3.9999339E+00    23           r
     1      1  1.0E+00      2  2.2E-06  5.2E-05  4.0000439E+00    23 1.7E+03 n r
     2      6  1.0E+00      3 (4.4E-10) 5.2E-05  4.0000433E+00    24 6.8E+01 s
     3      6  1.0E+00      4  1.2E-05  1.9E-05  3.9999952E+00    29 2.2E+01
     4      2  1.0E+00      5  1.5E-06  1.8E-05  3.9999898E+00    30 2.2E+01
     5      2  1.0E+00      6 (4.6E-07) 1.0E-05  3.9999860E+00    29 2.2E+01
     6      2  1.0E+00      7 (3.6E-07) 6.4E-06  3.9999821E+00    30 2.2E+01
     7      1  1.0E+00      8 (2.1E-07) 2.5E-06  3.9999807E+00    30 2.2E+01
     8      4  1.0E+00      9 (1.8E-08)(1.8E-06) 3.9999805E+00    30 2.2E+01

 SNOPTA EXIT   0 -- finished successfully
 SNOPTA INFO   1 -- optimality conditions satisfied

 ---- Analysis of Mesh in Phase 1 ----

Maximum Relative Error on Current Mesh in Phase 1 = 5.5764e-07
Mesh Error Tolerance IS satisfied in Phase 1
```