A Problem Solving Environment for Dynamic Optimization of Partial Differential-Algebraic Equation Systems *

Linda Petzold[†] Radu Serban[‡] Shengtai Li[§] Soumyendu Raha[¶] Andrew Strelzoff[‖]

March 1, 2000

Abstract

We report on our progress in developing a problem-solving environment for simulation, sensitivity analysis and dynamic optimization of partial differential-algebraic equation (PDAE) systems. The basic mathematical approach is outlined, along with research issues ranging from the use of problem structure to obtain efficiency and the need for tools to provide advice and diagnostic information on problem formulation, to user interface and revision management. Results and experiences with several challenging engineering applications are described.

Key words: simulation, sensitivity analysis, dynamic optimization, partial differential equations, differential algebraic equations.

AMS subject classifications: 65K10, 65L80, 65M20, 68U99.

1 Mathematical Approach and Computational Challenges

We consider the differential-algebraic equation (DAE) system

$$\begin{aligned} \mathbf{F}(t, \mathbf{x}, \mathbf{x}', \mathbf{p}, \mathbf{u}(t)) &= 0 \\ \mathbf{x}(t_1, \mathbf{r}) &= \mathbf{x}_1(\mathbf{r}) \end{aligned} \tag{1}$$

where the DAE is index one (see [4] or [1]) and the initial conditions have been chosen so that they are consistent (so that the constraints of the DAE are satisfied). The control parameters $\mathbf{p}$ and the vector-valued control function $\mathbf{u}(t)$ must be determined such that the objective function

$$\int_{t_1}^{t_{\max}} \Psi(t, \mathbf{x}(t), \mathbf{p}, \mathbf{u}(t))\, dt \qquad \text{is minimized}$$

and some additional inequality constraints

$$G(t, \mathbf{x}(t), \mathbf{p}, \mathbf{u}(t)) \geq 0$$

*This research was supported by the NSF Multidisciplinary Challenge Grant CCR-98-9896198, by the NSF KDI grant ATM-9873133, by the NSF and DARPA Virtual Integrated Processing program, by DOE DE-F603-98ER25354 and by LLNL ISCR 00-15

[†]University of California Santa Barbara, Dept. of Mechanical and Environmental Engineering, petzold@engineering.ucsb.edu

[‡]University of California Santa Barbara, Dept. of Mechanical and Environmental Engineering, radu@engineering.ucsb.edu

[§]University of California Santa Barbara, Dept. of Computer Science, shengtai@engineering.ucsb.edu

[¶]University of California Santa Barbara, Dept. of Computer Science, raha@engineering.ucsb.edu

[‖]University of California Santa Barbara, Dept. of Computer Science, strelz@engineering.ucsb.edu

16th IMACS World Congress (© 2000 IMACS)

are satisfied. The optimal control function $\mathbf{u}^*(t)$ is assumed to be continuous. In several of our applications, the DAE system is large-scale. Thus, the dimension N_x of $\mathbf{x}$ is large. However, the dimension of the control parameters and of the representation of the control function $\mathbf{u}(t)$ is much smaller. To represent $\mathbf{u}(t)$ in a low-dimensional vector space, we use piecewise polynomials on $[t_1, t_{\max}]$, their coefficients being determined by the optimization. For ease of presentation we can therefore assume that the vector $\mathbf{p}$ contains both the parameters and these coefficients (we let M denote the combined number of these values) and discard the control function $\mathbf{u}(t)$ in the remainder of this section. Also, we consider that the initial states are fixed and therefore discard the dependency of $\mathbf{x}_1$ on $\mathbf{r}$. Hence we consider

$$\mathbf{F}(t, \mathbf{x}, \mathbf{x}', \mathbf{p}) = 0, \qquad \mathbf{x}(t_1) = \mathbf{x}_1, \tag{2}$$

$$\int_{t_1}^{t_{\max}} \psi(t, \mathbf{x}(t), \mathbf{p})\, dt \qquad \text{is minimized}, \tag{3}$$

$$\mathbf{g}(t, \mathbf{x}(t), \mathbf{p}) \geq 0. \tag{4}$$

There are a number of well-known methods for direct discretization of this optimal control problem, for the case that the DAEs can be reduced to ordinary differential equations (ODEs) in standard form. The *single shooting method* solves the ODEs (2) over the interval $[t_1, t_{\max}]$, with the set of controls generated at each iteration by the optimization algorithm. However, it is well-known that single shooting can suffer from a lack of stability and robustness [2]. Moreover, for this method it is more difficult to maintain additional constraints and to ensure that the iterates are physical or computable. The *finite-difference method* or *collocation method* discretizes the ODEs over the interval $[t_1, t_{\max}]$ with the ODE solutions at each discrete time and the set of controls generated at each iteration by the optimization algorithm. Although this method is more robust and stable than the single shooting method, it requires the solution of an optimization problem which for a large-scale ODE system is enormous, and it does not allow for the use of adaptive ODE or (in the case that the ODE system is the result of semi-discretization of PDEs) PDE software.

We thus consider the *multiple-shooting method* for the discretization of the optimal control problem. In this method, the time interval $[t_1, t_{\max}]$ is divided into subintervals $[t_{itx}, t_{itx+1}]$ $(itx = 1, \ldots, N_{tx})$, and the differential equations (2) are solved over each subinterval, where additional intermediate variables $\mathbf{X}_{itx}$ are introduced. On each subinterval we denote the solution at time t of (2) with initial value $\mathbf{X}_{itx}$ at t_{itx} by $\mathbf{x}(t, t_{itx}, \mathbf{X}_{itx}, \mathbf{p})$.

Continuity between subintervals is achieved via the continuity constraints

$$\mathbf{C}_1^{itx}(\mathbf{X}_{itx+1}, \mathbf{X}_{itx}, \mathbf{p}) \equiv \mathbf{X}_{itx+1} - \mathbf{x}(t_{itx+1}, t_{itx}, \mathbf{X}_{itx}, \mathbf{p}) = \mathbf{0}.$$

For the DAE solution to be defined on each multiple shooting subinterval, it must be provided with a set of initial values which are consistent (that is, the initial values must satisfy any algebraic constraints in the DAE). This is not generally the case with initial values provided by methods like SQP because these methods are not feasible (in other words, intermediate solutions generated by the optimizer do not necessarily satisfy constraints in the optimization problem although the final solution does). To begin each interval with a consistent set of initial values, we first project the intermediate solution generated by SNOPT onto the constraints, and then solve the DAE system over the subinterval. In the case of index-1 problems with well-defined algebraic variables and constraints such as the problem considered in this paper, this means that we perturb the intermediate initial values of the algebraic variables so that they satisfy the constraints at the beginning of each multiple shooting subinterval.

The additional constraints (4) are required to be satisfied at the boundaries of the shooting intervals

$$\mathbf{C}_2^{itx}(\mathbf{X}_{itx}, \mathbf{p}) \equiv \mathbf{g}(t_{itx}, \mathbf{X}_{itx}, \mathbf{p}) \geq \mathbf{0}.$$

Following common practice, we write

$$\Phi(t) = \int_{t_1}^{t} \psi(\tau, \mathbf{x}(\tau), \mathbf{p})\, d\tau, \tag{5}$$

which satisfies $\Phi'(t) = \psi(t, \mathbf{x}(t), \mathbf{p})$, $\Phi(t_1) = 0$. This introduces another equation and variable into the differential system (2). The discretized optimal control problem becomes

$$\min_{\mathbf{X}_2, \ldots, \mathbf{X}_{Ntx}, \mathbf{p}} \Phi(t_{\max}) \tag{6}$$

subject to the constraints

$$\mathbf{C}_1^{itx}(\mathbf{X}_{itx+1}, \mathbf{X}_{itx}, \mathbf{p}) = \mathbf{0}, \tag{7}$$

$$\mathbf{C}_2^{itx}(\mathbf{X}_{itx}, \mathbf{p}) \geq \mathbf{0}. \tag{8}$$

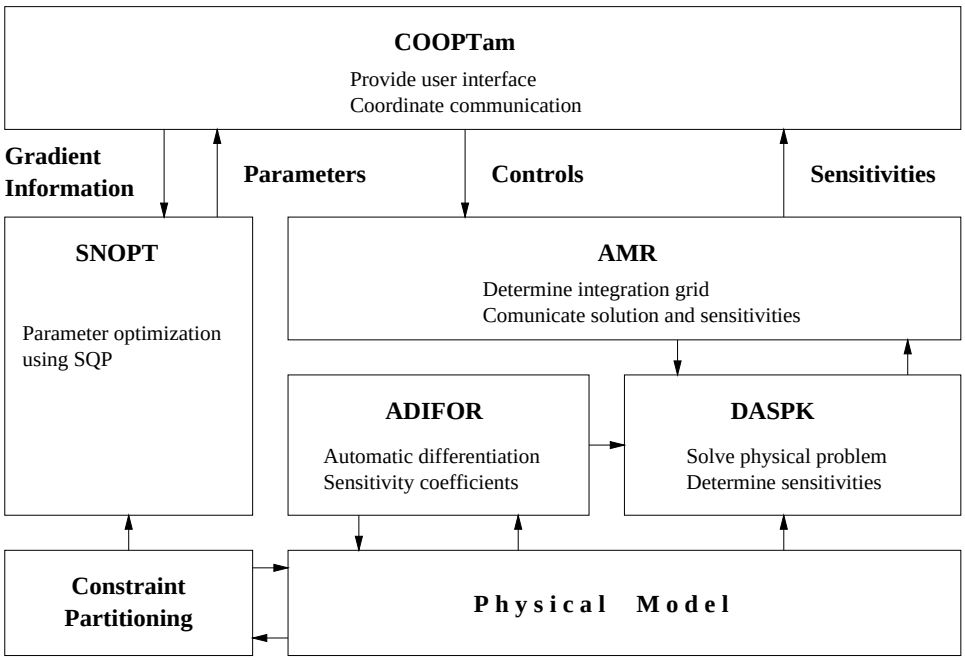

Figure 1: COOPT flow chart.

This problem can be solved by an optimization code. We use the solver SNOPT [6], which incorporates a sequential quadratic programming (SQP) method (see [7]). The SQP methods require a gradient and Jacobian matrix that are the derivatives of the objective function and constraints with respect to the optimization variables. We compute these derivatives via differential-algebraic equation (DAE) sensitivity software DASPK3.0 [8]. Further details on the sensitivity analysis will be given in the lecture. The sensitivity equations to be solved by DASPK3.0 are generated via the automatic differentiation software ADIFOR [3]. Our basic algorithms and software for the optimal control of dynamical systems are described in detail in [15].

This basic multiple-shooting type of strategy can work very well for small-to-moderate size ODE systems, and has an additional advantage that it is inherently parallel. However, for large-scale ODE and DAE systems there is a problem because the computational complexity grows rapidly with the dimension of the ODE system. The difficulty lies in the computation of the derivatives of the continuity constraints with respect to the variables $\mathbf{X}_{itx}$. The work to compute the derivative matrix $\partial \mathbf{x}(t)/\partial \mathbf{X}_{itx}$ is of order $\mathcal{O}(N_x^2)$, and for the problems under consideration N_x can be very large (for example, for an ODE system obtained from the semi-discretization of a PDE system, N_x is the product of the number of PDEs and the number of spatial grid points). In contrast, the computational work for the single shooting method is of order $\mathcal{O}(N_x N_p)$ although the method is not as stable, robust or parallelizable.

We reduce the computational complexity of the multiple shooting method for this type of problem by modifying the method to make use of the structure of the continuity constraints to reduce the number of sensitivity solutions which are needed to compute the derivatives[5]. To do this, we recast the continuity constraints in a form where only the matrix-vector products $(\partial \mathbf{x}(t)/\partial \mathbf{X}_{itx})\mathbf{w}_j$ are needed, rather than the entire matrix $\partial \mathbf{x}(t)/\partial \mathbf{X}_{itx}$. The matrix-vector products are directional derivatives; each can be computed via a single sensitivity analysis. The number of vectors $\mathbf{w}_j$ such that the directional sensitivities are needed is small, of order $\mathcal{O}(N_p)$. Thus the complexity of the modified multiple shooting computation is reduced to $\mathcal{O}(N_x N_p)$, roughly the same as that of single shooting. Unfortunately, the reduction in computational complexity comes at a price: the stability of the modified multiple shooting algorithm suffers from the same limitations as single shooting. However, for many DAE and PDE systems including the application described here, this is not an issue, and the modified method is more robust for nonlinear problems.

In the context of the SQP method, the use of modified multiple shooting involves a transformation of the constraint Jacobian. The affected rows are those associated with the continuity constraints and any path constraints applied within the shooting intervals. Path constraints enforced at the shooting points (and other constraints involving only discretized states) are not transformed. The transformation is cast almost entirely at the user level and requires minimal changes to the optimization software, which is important because software in this area is constantly being modified and improved. Gill *et.al.* ([5]) have shown that the modified quadratic subproblem yields a descent direction

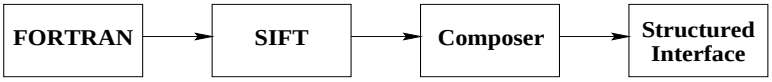

Figure 2: JMPL flow chart.

for the ℓ_1 penalty function. DAOPT is a modification to the SNOPT optimization code that uses a merit function based on an ℓ_1 penalty function.

An important issue for this type of method is the handling of constraints[13]. In particular, given an equality constraint, it can be included in the dynamic optimization problem either as part of the DAE, or to be handled directly by the optimizer. If it is included as part of the DAE, then there is a possibility that it could alter the index (mathematical structure) of the DAE, making it potentially more difficult to solve; on the other hand, if it is an important physical constraint, then its inclusion into the DAE where it will always be enforced should help the optimizer to avoid non-physical solutions. In our work, this question has been studied from the point of view of mathematical structure (index), and stability/conditioning of the DAE. It turns out that including some constraints with the DAE can alter the DAE stability, in a way which may be either favorable or unfavorable. Algorithms have been developed for partitioning the constraints to lead to a stable DAE system of index two or lower (which can be solved by the existing software). The logarithmic norm, which is closely related to the pseudoeigenvalues, is used to measure the stability for potential partitionings.

A limitation of the basic COOPT software arises because the method of lines is used to solve the PDAE. This does not allow for an adaptive grid in space, which would be needed if there are steep spatial gradients moving in time. We have developed methods and software based on adaptive mesh refinement (AMR) for systems of partial differential equations[9]. The software has been designed to be flexible, to allow the user to 'plug and play' existing non-adaptive simulation software, and to exert direct control over the adaptivity when needed. Recently we have shown how to compute sensitivity derivatives via AMR [10], and developed a capability for doing dynamic optimization including AMR for the time-dependent PDE systems. The structure of COOPTam, the extended COOPT software with adaptive mesh capabilities, is presented in Fig. 1.

2 User Interface and Revision Management

Software such as that described for dynamic optimization with adaptive solution of the DAE or PDAE has reached a level of complexity such that it is not a simple matter for the engineer or scientist to input a complete description of the problem. At the same time, we have come to expect that software in general should be natural and easy to use, for example via a graphic user interface (GUI). It is possible to develop a GUI which for example is specific to the COOPT software, but there are some problems. (1) The underlying software, including the problem specification, is constantly changing as new algorithms and capabilities are developed. (2) The user community is quite diverse, as evidenced from some of the scientific and engineering applications described later. Problems from widely different areas have different requirements for the specification and even for the vocabulary that need to be addressed in the interface.

We are addressing this problem by creating an environment which would allow developers of numeric and scientific software to quickly, easily and in a semi-automatic fashion create matching Java front ends for their programs. The project, which is called JMPL (Java math package launcher), is envisioned as a continuing publication environment - as new versions of the numeric software become available, JMPL should assist both the publisher in creating new front ends from old and the end user in importing old problem sets into the new software. Revision management presents some of the most interesting and challenging research problems. The project has focussed so far on two main areas of JMPL development. 'SIFT' - structured interface format, will analyze and discover program structure, and 'Composer' will construct a Java Gui interface from the structural information (Fig. 2).

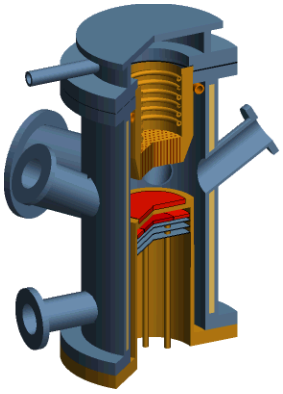

Figure 3: Stagnation-flow reactor. Metalorganic precursors enter a heated plenum and mixing chamber at the top of the reactor. A stagnation-flow region is established between a showerhead manifold and a heated wafer surface. The exhaust gases are pumped out the bottom of the reactor chamber

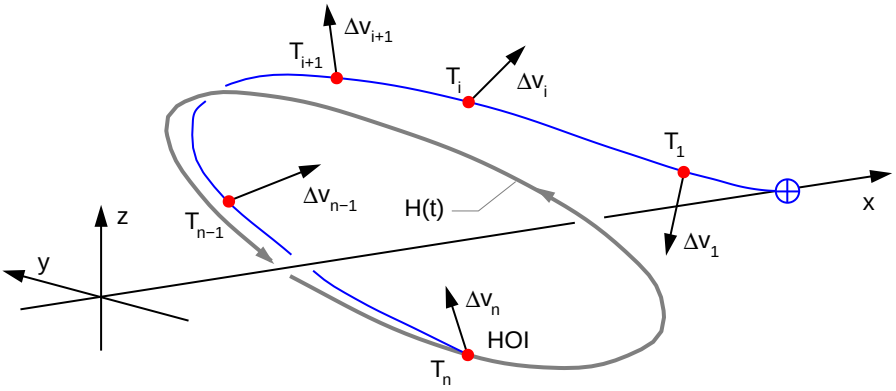

Figure 4: Transfer trajectory from Earth to the Halo Orbit around the Lagrange point L_1 ot the Sun-Earth system. Maneuvers take place at times T_i,$i = 1, 2, ..., n$.

3 Application Examples

3.1 Dynamic Optimization of Chemical Vapor Deposition Processes in Stagnation Flow Reactors

In this application[14] we are performing dynamic optimization for the optimal control of transient phenomena in chemical-vapor-deposition (CVD) processes. The algorithm is demonstrated for a stagnation-flow type reactor configuration. The equations describing chemically reacting stagnation flows are written in a transient compressible similarity form. After discretizing the spatial derivatives on a finite-volume mesh, the system becomes a set of differential-algebraic equations (DAEs). The DAEs constitute the model system for the dynamic optimization algorithm. The example demonstrates the use of dynamic optimization in the control of film stoichiometry during imposed transients in the CVD of multicomponent films. Time-varying trajectories of precursor composition at the reactor inlet are computed by the algorithm, so that correct flux ratios of yttrium, barium, and copper atoms to the surface are maintained during deposition of Yttrium-Barium-Copper-Oxide (YBCO) superconducting thin films.

3.2 Trajectory Planning for Halo Orbit Insertion

In this second application, we are applying dynamic optimization techniques to the problem of trajectory planning and orbit insertion in the circular restricted three-body problem (CR3BP). The algorithm is employed to design a continous trajectory from the Earth-Moon system to the periodic halo orbit around the libration point $\mathcal{L}_1$ of the Sun-Earth system[11]. This results in an impulsive optimal control problem which is cast as a parameter optimization

problem. The underlying dynamic model consists of an ODE system of dimension $6 \cdot n_M$, where n_M is the number of maneuvers (bursts) along the trajectory (Fig. 4). We minimize the fuel consumption, expressed as a function of the sizes of the maneuvers (in m/s) by selecting optimal values for the maneuver times and magnitudes[12].

The goal of this work was to decide whether optimal fuel cost solutions exist for different optimal problem constraints, such as different delays in the first maneuver ($TM1$) and different perturbations in the launching velocity. For all combinations explored (1 day $\leq TM1 \leq$ 5 days and $-7m/s \leq \epsilon_{v_0} \leq 7m/s$) the dynamic optimization algorithm found optimal trajectories that minimize fuel consumption with a minimal number of maneuvers. The dependency of the optimal solutions on these parameters was also confirmed by evaluating its sensitivities with respect to changes in the constraint levels. Current research focuses on moving the insertion point from the halo orbit to the stable manifold originating on the halo orbit.

References

[1] U. M. Ascher and L. R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, SIAM, 1998.

[2] U. M. Ascher, R. M. M. Mattheij, and R. D. Russell, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, Society for Industrial and Applied Mathematics (SIAM) Publications, Philadelphia, PA, 1995. ISBN 0-89871-354-4.

[3] C. Bischof, A. Carle, G. Corliss, A. Griewank, and P. Hovland, *ADIFOR—generating derivative codes from Fortran programs*, Scientific Programming, 1 (1992), pp. 11–29.

[4] K. E. Brenan, S. L. Campbell, and L. R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, SIAM Publications, Philadelphia, second ed., 1995. ISBN 0-89871-353-6.

[5] P. E. Gill, L. O. Jay, M. W. Leonard, L. R. Petzold and V. Sharma, *An SQP Method for the Optimal Control of Large-Scale Dynamical Systems*, to appear, J. Comp. Appl. Math.

[6] P. E. Gill, W. Murray, and M. A. Saunders, *SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization*, Numerical Analysis Report 97-2, Department of Mathematics, University of California, San Diego, La Jolla, CA, 1997.

[7] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*, Academic Press, London and New York, 1981. ISBN 0-12-283952-8.

[8] S. Li and L.R. Petzold, *Design of New DASPK for Sensitivity Analysis*, UCSB Department of Computer Science Technical Report, 1999.

[9] S. Li, *Adaptive Mesh Methods and Software for Time-Dependent PDEs*, Ph.D. thesis, Department of Computer Science, University of Minnesota, 1998.

[10] S. Li, L. R. Petzold and J. M. Hyman, *Solution Adapted Nested Grid Refinement and Sensitivity Analysis for Parabolic PDE Systems*, in preparation

[11] M. Lo and S.Ross, *Low Energy Interplanetary Transfers Using Invariant Manifolds of L1, L2, and Halo Orbits*, AAS/AIAA Space Flight Mechanics Meeting, Monterey, Ca, 9-11 February 1998

[12] R. Serban, W. S. Koon, M. Lo, J. E. Marsden, L. R. Petzold, S. D. Ross and R. S. Wilson, *Optimal Control for Halo Orbit Missions*, to appear, Proc. IFAC Workshop on Lagrangian and Hamiltonian Methods for Nonlinear Control, March 16-18, 2000, Princeton Univ.

[13] Soumyendu Raha, *Constraint Partitioning for Solution of Path-Constrained Dynamic Optimization Problems*, Ph.D. Thesis, University of Minnesota, in preparation.

[14] L. Raja, R. Kee, R. Serban, and L.R. Petzold *Dynamic Optimization of Chemically Reacting Stagnation Flows*, 1998 Electrochemical Society Conference, Boston, Ma.

[15] R. Serban, *COOPT - Control and Optimization of Dynamic Systems - Users' Guide*, UCSB, Department of Mechanical and Environmental Engineering, Report UCSB-ME-99-1,1999.