Boundary Value Approach for Dynamic Optimization

P. Tanarkit & L.T. Biegler

EDRC 06-170-94

Boundary Value Approach for Dynamic Optimization

P. Tanartkit and L. T. Biegler*

Department of Chemical Engineering,
Engineering Design Research Center,
Carnegie Mellon University, Pittsburgh, PA 15213.
email: biegler@cmu.edu

Abstract, Process engineering provides a wealth of applications for dynamic optimization problems. This problem is usually solved by transforming it to a nonlinear programming (NLP) problem with either sequential or simultaneous approaches. However, both approaches can still be inefficient to tackle large problems. In addition, many problems in chemical engineering arc naturally boundary value problems (BVP) which suffer from instability if we utilize a decomposition based on single shooting solvers. In this paper, we will introduce a simple extension to the simultaneous approach that will alleviate the dimensionality problem as well as ensure stability for BVP's. Many numerical aspects of the problem will be discussed, especially the discretization of the differential equations and the index problem. By using Radau collocation, the algorithm has favorable stability properties for high index problems and by exploiting the structure of the resulting system, a stable and efficient decomposition algorithm results. Here solution of this NLP formulation is considered through a reduced Hessian *Successive* Quadratic Programming (SQP) approach, where linearized state variables are eliminated and reduced quadratic programming (QP) subproblems update the control variables. Although this study primarily addresses fixed element problems, another key aspect of the success of the DAE optimization is the element placement. In order to enforce accuracy in the solution profiles, highly nonlinear constraints have to be added and these further complicate the solution formulation. As a result, the formulation turns out to be very sensitive to initializations. To address these problems, we will introduce a new framework that will decouple the element placement from the optimal control procedure. This framework consists of two layers of optimization, the inner and outer problems. The inner problem is a traditional optimal control problem with fixed element sizes and the outer problem is then used to update them solely via error control criteria and optimality conditions. We will also present an example to illustrate the element placement via bilevel optimization.

Key Words, Dynamic Optimization, Optimal Control, Successive Quadratic Programming, Finite Difference, Multiple Shooting

1 Introduction

The interest in dynamic optimization has been expanded to diverse areas of applications in various fields of engineering. Within chemical engineering, typical examples involve dynamic simulation, reactor network synthesis [6] and optimal control for various units. The motivation for this research comes from the fact that steady state simulations are not adequate for many processes. Common problems that require the solution of dynamic optimization are:

- **Control and scheduling of batch processes, due to their transient nature.**

***to whom all correspondence should be addressed.**

- **Processes with tight transient requirements, for example for safety, waste, and operability.**
- **Sophisticated control schemes, e.g., predictive, optimal or adaptive controls.**
- **Procedures for start-up and shut-down periods.**

The dynamic optimization problem can be formulated using a differential-algebraic equation (DAE) formulation. The DAE system consists of differential equations that describe the behavior of the system, such as mass and energy balances, and algebraic constraints that ensure thermodynamic consistency or other physically meaningful relations, such as rate constants or temperature limits.

A general DAE optimization problem (DAE1) can be stated as:

$$\min_{u(t),p} \phi(z(t_f),y(t_f),u(t_f),t_f,p) \tag{1}$$

$$s.t.: \quad F(z(t),\dot{z}(t),y(t),u(t),t,p) = 0 \tag{2}$$

$$G(z(t),y(t),u(t),t,p) \le 0 \tag{3}$$

$$G_s(z(t_s),y(t_s),u(t_s),t_s,p) \le 0 \tag{4}$$

where ϕ **is a scalar objective function,**
G **are algebraic inequality constraints,**
G_s **are point condition constraints (e.g., initial or final conditions) at times** t_{si}
z, y **are state profile vectors,**
u **are control profile vectors,**
t_f **is final time, and**
p **is time-independent parameter vector.**

The solution techniques for DAE optimization problems have been investigated since the 1960's (for a review see [19]). Most of the numerical techniques in this area can be classified into two major approaches.

The first attempt to solve the optimal control problem employed the idea of calculus of variation that was introduced in the 1960's (Pontryagin Maximum Principle [22]). In the variational formulation approach the optimization problem is transformed into a two-point boundary value problem (TPBVP). This approach works very well for unconstrained problems, however, the solution of the TPBVp is still difficult to solve, especially with the presence of inequalities in the model.

Another approach is to apply a nonlinear programming (NLP) solver to the DAE model. In order to use an existing solver, a modification has to be made to transform a DAE optimization problem to an NLP. Most methods used in this approach fall into two groups, namely sequential and simultaneous strategies.

Sequential Strategy

This approach has been applied to the optimal control problem for over two decades (s£e {28] for review). The idea is to discretize the control profile into a piecewise polynomial on finite elements. Then the state and sensitivity equations are integrated using standard DAE solvers for given control profiles ancl this yields function and gradient values for the NLP solver. An optimization routine is then applied in an outer loop to update the control actions. The advantage of this strategy lies in its reduced dimensionality of the optimization problem. However, state variable constraints cannot be enforced directly. Moreover, the integration step might be infeasible or too expensive to converge at intermediate trial points.

Simultaneous Strategy

In the simultaneous strategy [19, 27], on the other hand, the DAE solution and optimization are simultaneously converged through discretization of both the state and control profiles. Vasantharajan and Biegfer [27] proved the equivalence of this formulation to the calculus of variations and suggested a decomposition scheme for solving the resulting NLP. The advantages of this approach are the treatment of profile constraints as w&U as the elimination of expensive and possibly hifeasible intermediate solutions. The main drawback of this approach is its explosive problem size. To easS the dimensionality issue, an NLP decomposition was introduced [19], corresponding to a linearized single shooting method. However like the sequential method, this approach is not guaranteed to be stable for some boundary value problems (BVP) [1]. In addition, the formulation generated has a high degree of nonlinearity due to the nonlinear element placement constraints introduced in the process. As a result, it can be very sensitive to starting points and requires careful initializations. We will base our analysis on the simultaneous strategy with a new decomposition that will overcome these difficulties.

To summarize, the difficulties associated with the resulting NLP can be categorized into four main areas:

- the presence of algebraic equations (index problem),
- the determination of the step size for discretization,
- the high dimensionality of the discretized system, and
- the integration of the state equations.

Although the last two issues will be the main foci of this study, in the last section we will briefly discuss our preliminary work concerning step size selection. In the next section we discuss the advantages of a BVP approach to DAE optimization over current methods. This motivates the problem formulation in section 3 that are based on COLDAE and the reduced Hessian SQP method. This is demonstrated on several examples in section 4 and future work on element placement is described in section 5.

2 Differential Algebraic Equation System

In order to transform a DAE optimization problem to an NLP problem, we discretize the state and control profiles with a suitable polynomial to implicitly integrate the DAE. In the 1970's, de Boor and co-workers [5] demonstrated that the collocation method has a high order of convergence and can be factored by an efficient algorithm. The collocation method requires that the approximation profiles satisfy the m^{ih} order differential equations at M collocation points by finding the coefficients of the polynomials in the form

$$z(t) = \sum_{j=1}^{A/+m} ajTjity \qquad (5)$$

where Fj is an independent polynomial basis function.

The collocation method is equivalent to the Runge-Kutta method and its accuracy depends on the number and locations of the collocation points, and the range of integration (for further details on stability of the Runge-Kutta method see [1]). In practice, the number of collocation points is usually small, because high degree polynomials tend to oscillate. In addition, global interpolations are not suited for steep or discontinuous profiles. Instead, the finite element method, where the time horizon is divided into subintervals, is usually the method of choice.

Using the finite element method, the sfate profiles are required to be continuous throughout the time horizon. On the other hand, control and algebraic variable profiles are allowed to have discontinuities at the boundaries of the elements. In addition to state an4 control variables, the optimization routine also determines the finite element lengths. The sizes of the finite elements are governed by stability, accuracy syid the optimal locations of breakpoints for control profiles. In this work, the monomial basis is used as recommended in [5] because of its smaller condition number and smaller rounding errors. For an m^{th} order ODE, the state profiles are approximated by

$$z(t) = \sum_{j=1}^{m} z_{ij} \frac{(t - t_i)}{(j-1)!} + h_i^m \sum_{j=1}^{M} w_{ij} \gamma_j \left(\frac{(t - t_i)}{h_i} \right) \tag{6}$$

where $z^{\wedge}$, wij are unknown polynomial coefficients, aud
$7j$ is a polynomial of order $M + m$ satisfying,

$$^{\wedge}{}_{rT}7i(0) = 0 \quad /or/ = l,\ldots,m; \quad j = l,\ldots,M \tag{7}$$

$$\frac{d^m}{dt^m}\gamma_{J\bullet}(ffr) = *j> \quad for\ r = !,\bullet\ldots,M; \quad j = l,\ldots,Af \tag{8}$$

where g_r is the collocation point within each element.

Although the local error can be kept small by using small element sizes or high order integration schemes, several researchers [3] observed that the error in DAE systems can propagate and be amplified along the trajectory. In particular, the presence of algebraic constraints in the differential equation system not only makes the integration unstable, but also causes problems in initialization. The Index problem, associated with integration from consistent initial values, can be resolved by symbolically differentiating the algebraic constraints that caused the index, but the index structure first has to be detected before differentiation can be carried out. Then the invariants, resulting from differentiation must be enforced with appropriate stabilizing schemes [3].

There have been several approaches to develop a systematic way for detecting the structural index of the problem. For instance, Pantelides [21] proposed an algorithm using graph representation, and Chung and Westerberg [13] provided a general treatment for index and near-index problems. For the optimal control problem, however, we cannot apply these techniques directly because we do not know a priori which constraints will be active along the optimal trajectory. Examples of profiles with different index for different parts within the solution trajectory can be found in section 5. Another way to solve a high index DAE problem, as proposed by Ascher et al [2], is to use an appropriate integration scheme, such as Radau collocation. Here we assume consistent initial conditions are given. This method coincides with the projected implicit Runge-Kutta method that has favorable stability properties even for high index problems.

2A Multiple Shooting Method vs. Single Shooting

As discussed in the previous section, the solution to the DAE system is effected by solving the collocation equations. The two most important approaches that are normally used are single shooting (SS) and multiple shooting (MS).

In the single shooting method, an initial value problem (IVP) is explicitly integrated by guessing the missing initial values. The appeal of this method is that the storage space required is considerably smaller and that only the local Jacobian matrices have to be inverted. The major disadvantage of single shooting is the poor stability for solving boundary value (BVP) problems that have inherently unstable modes in the forward direction. The error in BVP's solved using SS

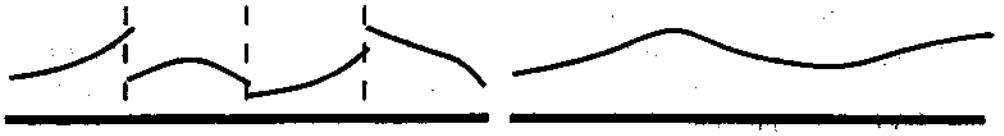

Figure 1: Multiple shooting (a) and single shooting (b)

tends to accumulate in a disastrous fashion and even good problem initializations are often not adequate for successful solutions. Unfortunately, many problems in chemical engineering are naturally BVP problems, for example kinetic and transport problems in various geometries. These difficulties can be redjice<d by using multiple shooting approaches if the problem is well-conditioned. We will define the condition of the problem in the next section. In MS. the domain of the integration will be divided into subintervals that correspond to t^e finite element representation that we used for approximating the profiles.

Several difficult parameter estimation examples solved by Bock and coworkers [9] demonstrate the advantages of this concept. For example, consider a problem [9] given as follows,

$$\dot{2/2} = T^2 y_1 - (\tau^2 + 7r^2)\sin(;r?) \quad (9)$$

$$\dot{Vi} = V2 \quad (10)$$

The solution of this problem, for any value of r, is given by

$$\mathbf{y}(t) = (\sin(\pi t), \pi\cos(\pi t))^T \quad (11)$$

This problem can be solved as a boundary value problem (BVPl) with the following conditions,

$$y_1(0) = 0 \quad (12)$$

$$y_1(1) = 0 \quad (13)$$

On the other hand, if we specify initial conditions (14-15) for this problem (IVP1) instead of boundary values (12-13), we have:

$$y»(0) = o \quad (14)$$

$$y_2(0) = 7T \quad (15)$$

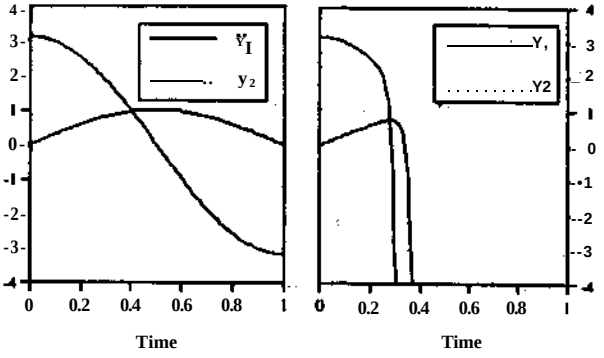

Figure 2: Multiple shooting for BVP and IVP (a) and single shooting for IVP (b)

The (IVP1) formulation is solved using both SS (LSODE [17] with tolerance of $1(T^6)$ and MS (C0LDAE [4] with tolerance of $10\sim^6$) methods with r set to 60. As seen in Figure 2, the SS method

tracks the solution only for one-fourth of the time horizon and then diverges, even though the value of 7r is correct up to 9 digits. The result from MS method are almost identical for both (BVP1) and (IVP1), although the computational time of the MS method with (IVP1) taken is about double that of the (BVP1) formulation.

2.2 Problem Conditioning

As seen, the (IVP1) formulation is extremely sensitive to the structure of ttteSide conditions. In order to understand the computational aspects of the problem, we separate the origiti of the failure into two major sources. The first is the algorithm-depetident instability, i.e. errors, either roundoff *ot* discretized errors, can be magnified along the trajettory. This can be circumvented by using a more stable or more accurate numerical method. On the other hand, roundoff error can also be magnified if the problem is too sensitive to parameters used, or *ill-conditioned.* This second cause cannot be avoided by simply selecting a more stable method.

To illustrate the influence of the problem formulation, we consider the following linear BVP (16-17) and provide an analysis from [1].

$$\mathbf{y}' = A(x)\mathbf{y} + \mathbf{q}(x) \tag{16}$$

$$B_a y(a) + B_b y(b) = /? \tag{17}$$

$$x \in [a, b]$$

The solution to this problem can be written as:

$$\mathbf{y} = Y(x)(B_a Y(a) + B_b Y(b)y^l(3 + \int_{Ja} t\, G\{x,t\}q\{t\}dt \tag{18}$$

where $Y(x)$ is a fundamental solution, and
$G(x,t)$ is the Green's function of the problem.

From (18), a bound on error can be obtained as:

$$\|\mathbf{e}\| \leq \kappa(\|\delta\beta\| + \|\delta\mathbf{q}\|) \tag{19}$$

$$K \geq \|Y(x)(B_a Y(a) + BiYib))^{-1}\| \tag{20}$$

$$K \geq \max_x \int_{Ja}^{rb} \|G(x,t)\|dt \tag{21}$$

where $6/3$ and $6q$ are perturbations on $/3$ and q, respectively.

The derivation above shows that the integration error (e) can be arbitrary large, even with small initial errors, because of the quantity *K.* Moreover, the bound on error may vary substantially depending on the problem formulation (or the boundary conditions). We use the notion of *condition number (K)* to quantify this phenomena, as in solving linear systems. Intuitively, the problem is well-conditioned if a small change in parameters produces a corresponding small error in the result, in other words, the problem has a moderate condition number. This condition number is therefore, used to determine if the problem, not the method, is well(or ill)-conditioned. For a detailed procedure to calculate this condition number, see [1].

For (BVP1) above, the corresponding BVP problem is well-conditioned with condition number of the problem about 5 [1] for r >> 1. Thus, if any stable integration scheme is applied, the solution is stable and accurate. The resulting profile using COLDAE integrator is also given in Figure 2.

The resulting initial value problem (IVP1), as it turns out, is not as well-conditoned as (BVP1). The condition number of this problem, K ~ exp(r), becomes much larger than the BVP counterpart when $r >> 1$, even if the analytical solutions for both cases are identical.

This example has implications for t^e choice of optimal control method, because any method based on single shooting will fail on this problem. Recall that for an IVP shooting integration, an approximate upper bound on roundoff error is given by $eexp^{Lh}$ where e is the machine precision, h is the length of integration interval, and L is the largest eigenvalue of the problem and in this case $L \sim T$. Also, note that the SS algorithm will be uniformly stable if L has a negative real part.

In this section, we demonstrated that the problem formulation, specifically boundary conditions, can affect the performace of the method. Although it is not possible to determine whether a set of side conditions agrees with the underlying characteristics of the outcome without the analyical solution, it is still much better to use the MS method, at least to reduce the integration interval, h and counter the exponential growth of roundoff error.

3 Fixed-Mesh Problems

If we adopt the multiple shooting approach and substitute a collocation appoach in each element and we assume for the moment that the element lengths are fixed, we have the following problem statement (NLP1)

$$\min_{z_i, \dot{z}_i, \text{you'u}>\text{p't/}} \$(zi, z\dot{i}, yij_r uij, p, fii, t_f) \tag{22}$$

s. t. : Discretized DAE model:

$$\dot{z}_{ij} - \bar{h}_i \bar{F}(zi,\ Zi,\ y_{ijy}\ u_{ijy} p) = 0 \tag{23}$$

$$G(zuZi, y_{i5},\ tiij, p) \leq 0 \tag{24}$$

for $i = 1, \ldots,$ ne; $j = 1, \ldots, ncol$

point condition:

$$G_k(z_i, \dot{z}_{i\,j} y_{ij}, u_{ij}, p) \leq 0 \tag{25}$$

for $k = 1, \ldots, ns$

bounds:

$$z^L \leq za \leq z^v \tag{26}$$

$$V^L \leq Vij \leq V^U \tag{27}$$

$$u^L \leq mj \leq u^u \tag{28}$$

$$P^L \leq P \leq P^U \tag{29}$$

$$t^L \leq t_f < t^u \tag{30}$$

for $i = 1, \ldots, ne;\ j = 1, \ldots, ncol$

where *ncol* is the number of collocation points,
ne is the number of elements, and
ns is the number of point conditions.

For this formulation $\bar{hi}$'s are fixed, and we will discuss the variation of $\bar{hi}$'s in section 5

3.1 Successive Quadratic Programming (SQP)

It is often argued that SQP methods are best suited to process problems compared to other methods, because they require fewer function and gradient evaluations. Here, the dimension of the state variables is generally much larger than that of the control variables (degrees of freedom). Hence, to solve large-scale problems efficiently, we consider a reduced space SQP algorithm. At each iteration *k*, a quadratic programming subproblem (QP1) is created from (NLP1) and solved to obtain a search direction *d* for .r, of the form:

$$\min_{p} \quad V\$(x^k)^T d + l/2 d^T B^k d \tag{31}$$

$$s.t. \quad c(x^k) + Vc(x^k f d = 0 \tag{32}$$

$$d \in [x^L - x^k, x^u - x^k] \tag{33}$$

$$x = [z \; zy \; uptf]^T \tag{34}$$

A decomposition algorithm can then be applied by partitioning variables into decision (control) variables and dependent (state) variables. Thus, the search directidn is divided[1] into:

$$d = (Y p_y)^k + (Z p_z)^k \tag{35}$$

with the range space direction (Yp_y) obtained by :

$$(Vc^T Y) p_y = \sim c(x^k) \tag{36}$$

and the following reduced QP subproblem (QP2) for the null space direction:

$$\min_{p} \quad V\$(x^k)^T p + l/2 p_z^T (Z^T B Z) p_2 + l/2 (Z^T \; B Y p_y)^T p_z \tag{37}$$

$$s.t. \quad x^k + Y p_y + Z p_2 \text{ e } [x^L . x^u] \tag{38}$$

Here $(Z^T BZ)$ is a quasi-Newton approximation to the reduced Hessian of the Lagrange function and $(Z^T B Y p_y)$ can be approximated through various options [8]. The QP subproblem was solved using the QPKWIK algorithm described in (25). QPKWIK is specifically tailored to solve large scale problems by updating the inverse Cholesky factors of the reduced Hessian $(Z^T BZ)$; therefore, it is $0(n^2)$ with respect to the degrees of freedom of the problem. Another advantage of QPKWIK over a conventional QP solver is its handling of highly-constrained QP subproblems, which often occur in DAE optimization problems.

3.2 Determination of the *Y* space move

A key issue related to reduced Hessian SQP is the determination of the *Y* space move, or equivalently, a Newton step for the system equations. The routine chosen for evaluating this move is COLDAE [4]. Along with its antecedents, COLSYS and COLNEW, this code is a well-tested implementation of collocation on finite elements and offers the following advantages. First, it is an efficient collocation code that exploits the almost block diagonal (ABD) structure of the collocation equations through the use of monomial basis functions and a sparse and stable decomposition. Second, it allows flexible specification of the DAE system, boundary conditions and approximation order. Third, as a Newton-based solver it allows an easy interface to reduced Hessian SQP methods and thus ensures all the benefits of a simultaneous strategy.

Using COLDAE, the collocation systems have two types of unknowns: global and local variables. Local variables are eliminated by decomposition of collocation equations in each element;

consequently, the size of the linear system that has to be factored for each iteration is drastically reduced. The resulting linear system of global variables is cast in an almost block diagonal form and solved by de Boor's sparse SOLVEBLOK routine [10]. From the definition of Yp_y (36), the limiting case of $Yp_y = 0$ leads to a reduced gradient *(GRG)* or a feasible path approach. Another case is the infeasible path approach, where Yp_y is the Newton step from the state equations. Finally, we modified COLDAE to accommodate Radau collocation as well as feasible andUnfeasible path optimization algorithms.

Feasible **Path**

In the feasible path, or *GRG* approach, the state equations are satisfied exactly; therefore, the *Y* space movement is zero. This approach has the same concept as the sequential approach described in section 1, which separates integration and optimization steps. As a result, this reduces the amount of the data storage and the need for Lagrange multipliers in the linesearch function. The *GRG* approach performs reasonably well for small, mostly linear problems, because only a few Newton iterations are required to converge with moderate steps in the *Z* space (control profile space).

Infeasible **Path**

In this approach, the algorithm is given in [26] which converges both the integration and optimization simultaneously. This algorithm has been proved to have a desirable one-step superlinear convergence property. The main challenge in this approach]& the evaluation of the Lagrange multipliers (A) for the state equation. At a Kuhn-Tucker point, the Lagrange multipliers can be approximated by,

$$A = -(1 \wedge V c)^{"1}\}'^{7}\wedge \tag{39}$$

Unfortunately, using COLDAE, the matrix $(Y^TVc)\sim^l$ is not readily available. As a result, a new linesearch algorithm must be investigated, instead of the Augmented Lagrange f7] function used in feasible path. Han [16] showed that using the exact penalty function ($(x) + /j||c(a;)||i) as the linesearch merit function, the SQP converges to a local optimizer from poor starting points. However, the penalty value (/z) has to satisfy the inequality,

$$/* > Plloo \tag{40}$$

Biegler [25] showed that a different penalty parameter can be used instead of *p.* to obtain the same global convergence property, without individual multipliers explicitly calculated. Here */.i'* is defined by (41).

$$/*' > \|^T c\|/\|c\|U \tag{41}$$

In addition, as shown in [26], the term $|A^r c|$ can be obtained easily by the factorized matrices from COLDAE. The exact penalty function method is usually slower [7] to converge than with other linesearch functions, so to accelerate convergence, we use a *watchdog* technique [11] that allows intermediate increases in the merit function but requires a reduction in the exact penalty every *q* steps (in our study $q = 2$).

4 Examples

In this section, we consider common problems that can be formulated into a DAE optimization problem. The Kuhn-Tucker errors for all examples are $10\sim^6$, and all CPU times were obtained on a DECstation 5000. For feasible path integration, the equation tolerances are $10\sim^8$. Except where indicated, all problems use a uniform element mesh.

4^1 Polymerization Problem

This example is ^n optimal control problem for a batch chain addition polymerization. Chain addition polymerization is one of the most common polymerization processes where polymers are formed by addition of one monomer unit at a time. The details of the model and the kinetic data are given in the appendix.

In this study, we consider the minimum reaction time for the polystyrene (PS) reactor as our objective; temperature *(T(-))* and initial initiator concentration (/) are control variables and final monomer conversion and number average chain length are specified. Many investigations [12, 24] have been done to obtain the result analytically and numerically using the maximum principle. However, most of them face difficulties with control constraints, such as temperature limits.

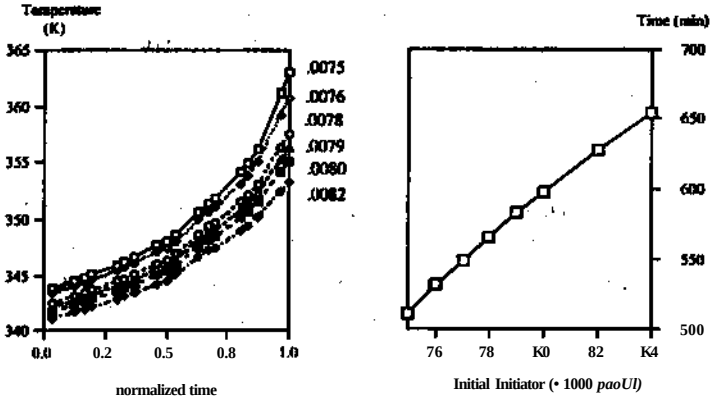

Figure 3: Optimal control profiles and final times for PS polymerization problem. The numbers right to the Figure (a) are the initial initiator concentration (gmol/1).

Table 1: Computational statistics for batch PS polymerization problem.

Initiator cone.	Number of SQP **iterations**		**CPU** time (s)		Objective (min)
(gmol/1)	feasible	infeasible	**feasible**	infeasible	function
.00820	39	**21**	26	12	627
.00806	40	29	27	16	607
.00800	35	26	20	14	598
.00790	39	29	22	17	583
.00780	30	29	20	16	566

The computational results are reported in Table 1 for both feasible and infeasible path approaches. The optimal initial initiator concentration is at the lower bounds for every case.

4.2 Van der Pol oscillator problem

This problem is formulated in [15]. The model is described by,

$$\min y_3(5) \tag{42}$$

$$s.t. : \quad \dot{y}_x = (1 - y\%)yi - y_2 + u \tag{43}$$

$$\dot{y}_2 = y_1 \tag{44}$$

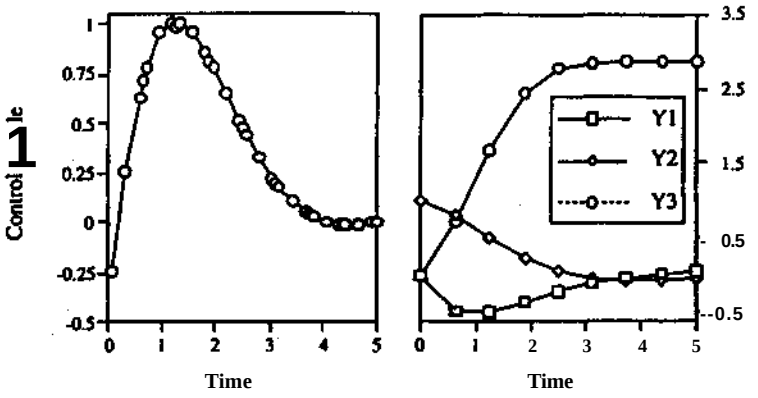

Figure 4: Optimal control state profiles for unconstrained Van der Pol oscillator problem.

Table 2: Computational statistics for unconstrained Van der Pol oscillator problem.

Number of collocation pt.	Number of elements	Number of SQP iterations	CPU time (s)	Objective function	Approach
3	8	39	36.5	2.8710	feasible path
3	8	20	15.7	2.8695	infeasible path

$$\dot{y}_3 = V_1^2 + y_2^2 + u^2 \quad (45)$$

$$y(0) = [0.0 \ \ 1.0 \ \ 0.0]^r \quad (46)$$

$$«(0 \ € \ (-0.3 \ \ 1.0) \quad (47)$$

As expected, the infeasible path approach performs much better for both Van der Pol (Figure 4 and Table 2) and PS polymerization examples (Figure 3 and Table 1). Therefore, for the rest of the examples, the infeasible path approach will be employed.

We also modify the problem by adding the following path constraint, by which the problem changes to index 2 instead of index 1, whenever (48) is active.

$$y \geq -0.4 \quad (48)$$

Using the fixed-mesh formulation, the constraint is easily added to the NLP as inequality constraints. As expected, the computational time for both bounded and unbounded problems are almost the same. Results are reported in Table 3 and Figure 5. These results are also compared with the solution obtained with an IVP approach by Vassiliadis [28].

4.3 Parameter Estimation Problem

This problem is a BVP problem taken from {9]. The system involves two differential equations and one unknown parameter. The state profiles are initialized to 1.0 and the unknown parameter (p =r ?r) is set to 2.0. The model of this example can be found in section 2 (9-10), with r set to 60, and the boundary conditions are

$$yi(0) = 0 \ ; \ \ yi(l) = O \quad (49)$$

The objective of this problem is to estimate the values for the parameter equal to *w* given true values corrupted with random noise *(N(0,a))* at data points 0,0.1,..,1.0. The results agree very

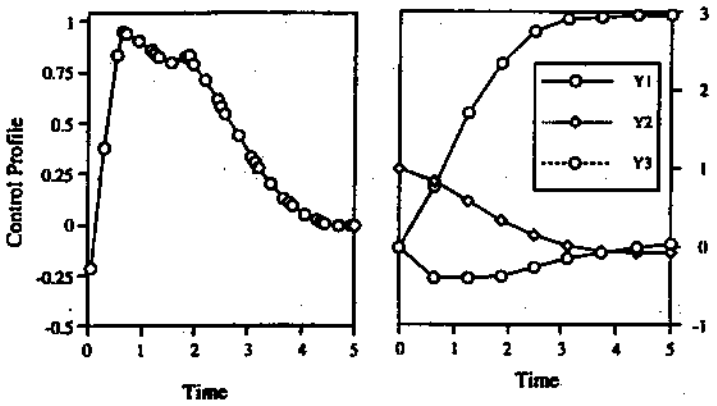

Figure 5: Optimal control and state profiles for Van der Pol oscillator problem with path constraints.

Table 3: Computational statistics for Van der Pol oscillator problem with and without path constraints.

Number of collocation pt.	Number of elements	Number of SQP iterations	CPU time (•)	$y\backslash$ bound	Objective(min) function
3	6	20	13.4	no	2.8735
3	8	20	15.7	no	2.8695
4	8	28	25.5	no	2.8670
3	9	28	30.9	no	2.8684
3	10	21	30.0	no	2.8680
Vassiliadis [28]				no	2.8681
3	6	23	12.8	yes	2.9834
3	8	20	15.5	yes	2.9549
4	8	21	26.0	yes	2.9537
3	9	24	27.9	yes	2.9540
3	10	20	28.0	yes	2.9552
Vassiliadis [28]				yes	2.95539

well with the true value as shown in Figure 6. Note also that the IVP problems require slightly more effort due to problem conditioning. This difference increases with increasing r.

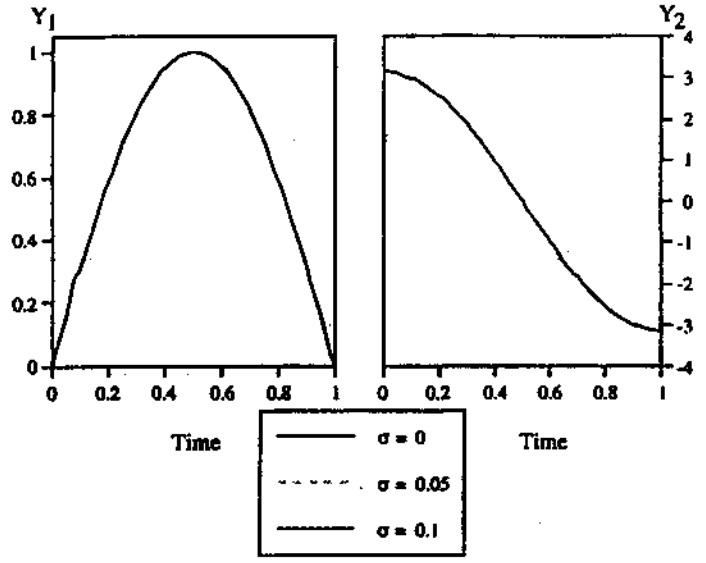

Figure 6: Optimal control and state profiles for Parameter estimation problem.

Table 4: Computational statistics for Parameter estimation problem.

Number of collocation pt.	Number of elements	Number of iterations	CPU time (s)	*a*	*w*
3	10	8	0.9	0.05	3.14140
4	10	9	1.2	0.05	3.14195
3	10	9	1.0	0.10	3.14121
4	10	9	1.2	0.10	3.14230
3	10	8	0.8	0.00	3.14159
4	10	9	1.2	0.00	3.14159
4$	10	8	1.5	0.00	3.14159
4$	10	10	1.9	0.05	3.14159
4$	10	14	2.3	0.10	3.14157

($ using IVP formulation (14-15))

4.4 Zeolite Distribution Problem

As previously described in the introduction, many problems in chemical engineering are naturally two-point boundary value problems. This last example deals with finding an optimal distribution of zeolite in a matrix of silica-alumina. The reaction system involved in this problem is the triangular reaction network, as shown in Table 7.

Both zeolite and the matrix catalyze all three reactions, but the rates of reaction are not equal. Zeolite is highly active and gives a better selectivity compared to the matrix, however, it also has higher resistance to diffusion. The purpose of the problem is to find an optimal distribution profile to maximize the selectivity for B as proposed in [20]. Further details of the model and data are

given in the appendix.

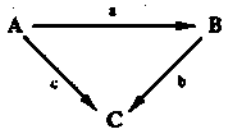

Figure 7: Schematic of Triangular reaction.

Table 5: Computational statistics for Zeolite distribution example.

Cases	Number of collocation pt.	Number of .elements	Number of iterations	CPU time . (•)	Objective (max) function
base	3	6	21	17.6	0.6507
1	3	6	18	17.8	0.6629
2	3	6	16	14.3	0.6692
3	3	6	14	16.5	0.6470

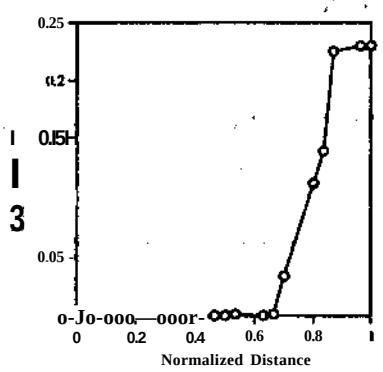

Figure 8: Optimal Zeolite profiles for base case.

The problem is solved using 6 elements and 3 collocation points. As shown in Figure 8, the optimal profiles are "dilute surface" distributions with most of the zeolite concentrated at the surface, and the results compare very well with the literature [20].

5 Nested Bilevel Framework

We consider the following DAE optimization problem with 2 differential equations, tha,t represent equations of motion. This "car" problem has the objective to minimize the time for an object to cover the distance of 300 *m* and start and stop at zero velocity with a speed limit at 10 *mfs.* This problem has an analytical solution given in [19].

$$\min_{u(\cdot)} \; tj \tag{50}$$

$$S.t. \qquad \dot{z}\backslash = u \tag{51}$$

$$\dot{Z}*i = Z\backslash \tag{52}$$

$$z(0) = [0.0C\backslash.0]^T \tag{53}$$

$$z(f_f) = [0.0\ 300.0]^r \quad (54)$$

$$u(t)\ e\ [-2.0\ 1.o] \quad (55)$$

Table 6: Computational statistics for Car problem

Number of collocation	Number of finite elements	Number of SQP iterations	CPU time (s)	Mesh	Objective (min) function
3	3	8	2.6	uniform	39.04
3	6	10	6.2	uniform	37.89
3	7	11	9.2	uniform	37.68
4	10	9	15.2	uniform	37.62
3	3	10	1.9	optimal	37.50

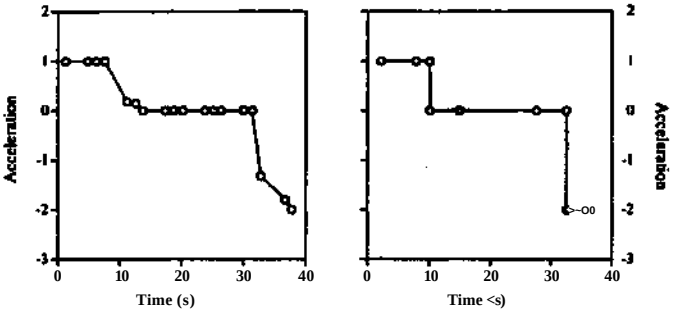

Figure 9: Optimal acceleration profiles for 6-uniform (a) and 3-optimal (b) meshes for car problem.

The results are obtained by using 6-uniform and 3-optimally spaced (from analytical solution) elements and are given in Figure 9 and Table 6. Here, the objective from the optimal mesh is always better, even when it is compared to 10-uniform elements. In addition, the uniform case required much more computation time than the optimal case. Note that the result for the optimal mesh problem is identical to the analytical solution.

5.1 Element Placement and Error Control

As seen in the example above, element placement plays a crucial role on the final result and efficiency of the method, and the optimal placement has significant impact on both the solution and the computational time. The more elements the problem has, the more time-consuming it will be. In addition, in order to estimate and bound the discretized and roundoff errors, a wide spectrum of element placement constraints can be used, ranging from highly nonlinear constraints, based on the residual at noncollocation points, to simple ad hoc bounds on elements. An extensive review of these constraints can be found in [23]. In this study, we incorporate the following relation,

$$\|e(t)\| = C\|\xi(t_{nc})\|h + O(h^{k+1}) \quad (56)$$

where $h\{$ **is step size,**

IMOU ^{ks} **approximated local error, and**

$\|f(<nc)\|$ **is the norm of the residual at non-collocation point**

as proposed in Russell [23]. Next we enforce constraint (56) without the $0(h^{k+l})$ term by bounding the residual for each equation, including algebraic equations, at non-collocation points as,

$$\& = \dot{2}\mathrm{t}(*\mathrm{nc}) - /\mathrm{t}(*\mathrm{nc}) \tag{57}$$

where (£;)/ is an interpolated residual for equation / at element *i.* As we include the error control constraints, the NLP problem is then as follows (NLP2) :

$$\min \quad \$(zi, Zij, yi_j, Uij, pJi_i, t_f) \tag{58}$$

s. t. : Discretized DAE model:

$$\dot{z}_{ij} - h_i \bar{F}(z_i, \dot{z}ij, yij, Uij, p, t_f) = 0 \tag{59}$$

$$G((z_i, \dot{z}ij.yij.Uij.pJjj\hat{}O \tag{60}$$

$$for\ i = 1,\ldots, \mathrm{ne};\ j = 1,\ldots, ncol$$

$$\sum_{i=1}^{ne} h_i = t_f \tag{61}$$

side condition:

$$Gkizi.Zi.yij.u\hat{}pJf) \leq 0 \tag{62}$$

$$for\ k = 1,.\ .,7\mathrm{i}6$$

bounds:

$$z^L \leq z\hat{} \leq z^u \tag{63}$$

$$y^L \leq yij \leq y^u \tag{64}$$

$$u^L < Uij \leq u^u \tag{65}$$

$$p^L < p < p^U \tag{66}$$

$$t^L < tj < t^u \tag{67}$$

$$for\ i = 1,.\ .,\mathrm{ne};\ j = 1,.\ .,ncol$$

element placement constraint:

$$-e \leq \& = zi(t_{nc}) - hi\bar{F}\{zi, zi, yij, u_{ti}; /;, t_f) \leq c \tag{68}$$

where *i* is the element,
j is the collocation point, and
hi: is the element length of.the element *i*

5-2 Formulation

After the discretization and the addition of the element placement constraints, the problem (DAE1) is transformed into a nonlinear programming problem (NLP2). Problem (NLP2) can then be solved with any large scale NLP solver as in [27]. However, element placement constraints (57) make the problem very nonlinear and sensitive to initialization. Even in simple cases, where the DAE model is linear and has no discontinuity in the control profile, the NLP is still difficult to solve.

To overcome this difficulty, we will propose a new framework that will accommodate the element placement with bilevel programming. The motivation for the framework comes from the fact that

element^.placpmept constraints do not influence the determination of the optimal control profile directly. In particular, if the control is differentiate throughout the horizon, the element placement role is only to ensure accuracy of the profiles. This task can be posed as a bilevel mathematical program with the outer problem determining the finite element lengths and the fixed-mesh (inner) problem determining the control and state variables once the element sizes are fixed. The outer problem (NLP3) can be written as:

$$\min_{hi} \quad \Phi(\alpha_i, z^*(h_i), y^*(h_i), u^*(h_i), \lambda^*(h_i)) \tag{69}$$

$$s.t. \qquad Hi\,hi = I \tag{70}$$

$$hi \in [h^L, h^u) \tag{71}$$

$$-\{\text{-error control contraints} \tag{72}$$

where ti*,zVy%iy and A* are implicit functions of the /i«'s defined by the inner (or fbced-mesh) problem (NLP1). While the inner(fixed-mesh) problem was addressed in previous sections, many aspects of the outer problem still need to be addressed. In particular, gradients for (NLP3) need to be computed using the solution of (NLP1).

In this preliminary study, we use a finite difference scheme to obtain the gradients for the outer problem (NLP3), i.e., perturb the element sizes and then re-solve the inner problem (NLP1) again. Future work will deal with an approach based on NLP sensitivities. With a finite difference scheme, we present small examples to illustrate the inner-outer problem framework.

Car Problem

In this section, we will solve the car problem again but with the bilevel approach. The main challenge here is to see whether the outer problem can detect the discontinuous points in acceleration. The starting profiles used are a constant acceleration at 0.5 m/s^2 and all state profiles are initialized to zero. All cases listed in Table 7 converged to the analytic solutions. In contrast, the algorithm used in [19] which simultaneously solved the optimal control and mesh selection failed to converge from this starting point. Note that the CPU times reported are not excessive considering the inefficient finite difference and feasible path inner problem implementation.

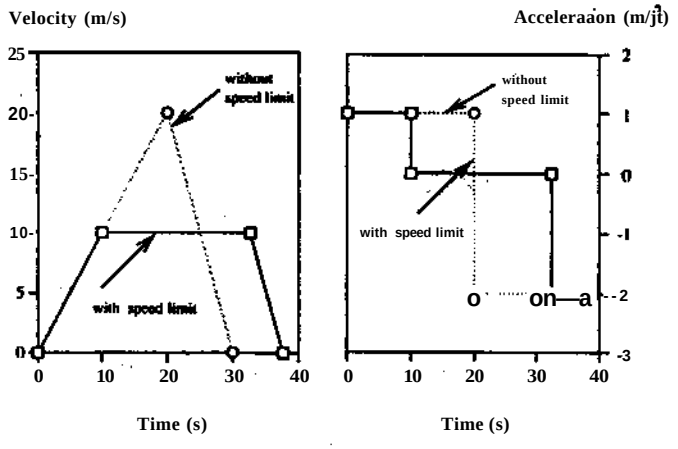

Figure 10: Optimal acceleration and velocity profiles.

We also re-solved the polymer reactor example in section 4 using the inner-outer problem formulation. But in this case, the element lengths do not change and optimization for the outer problem converges in one iteration, because the optimal control profiled are continuous and the errors in the integration are well within the tolerance.

Table 7: Computational statistics for Cat[4] problem using nested strategy with feasible path

Number of collocation	Number of finite elements	Number of iter. (outer)	CPU time (s)	Speed limit at 10 unit/s	Objective function
3	2	5	14.5	no	30.00
3	3	4	31.1	no	30.00
3	3	5	51.4	yes	37.50
3	4	6	61.3	yes	37.50

There are numerous efforts to solve the bilevel optimization problem, particularly in economics and planning. Kolstad and Lasdon [18] proposed a method for extracting derivative information for the-outer problem based on the Kuhn-Tucker conditions on the inner problem. Their properties fpUow from Fiacco [14] and require assumptions that the solution of the inner problem is unique, the gradients of the active constraints are not degenerate, and the complementary condition holds. Consequently, the derivatives of the outer problem cap be obtained as,

$$\nabla_x L) = V_{hx}L + V_{xx}LV_hx + \nabla_x c\nabla_h\lambda = 0 \tag{73}$$

$$\frac{d}{dh^y}\Big|_{\bar{}} - \nabla_h c + V_x cV_h x = 0 \tag{74}$$

$$V_h x^* = 0 \tag{75}$$

where L is the Lagrange function

x are the state and control variables

x^* are components of x at bounds

c are the active constraints

h are the element lengths

A range and null space decomposition can also be utilized (73-75) to make this approach less expensive (as shown in [29]) and this follows from,

$$\nabla_x c^T Z = 0 \tag{7G}$$

$$\nabla_h x = Y p_y + Z p_z \tag{77}$$

$$\begin{bmatrix} Z^T V_{rx}LZ & (2^T V_{XX}LY) \\ 0 & VrC^T Y \end{bmatrix} \begin{bmatrix} p_z \\ p_y \end{bmatrix} = \begin{bmatrix} Z^T \nabla_x L \\ \nabla_h c \end{bmatrix} \tag{78}$$

Although this method is very promising, an efficient sensitivity algorithm still needs to be applied to the inner problem.

As seen in the car problem, the performance of the method depends heavily on the size of the problem, i.e. the number of the elements. At the present time, there is no automatic procedure for the selection of the elements. Most of the strategies are based on qualitative knowledge concerning the problems or heuristics in addition to the trial and error procedure.

Therefore, the next step for a complete method is to develop a systematic element addition procedure based on adaptive element addition in [27]. With this goal, our inner-outer strategy looks promising, because the decoupling of the profile variables and the element sizes gives us a straightforward way to find the influence of the elements. Still, an extensive study of this issue is required.

6 Conclusion

Through an implementation of COLDAE, an efficient TPBVP solver, we have developed and compared feasible and infeasible path approaches for the fixed-mesh problem, using reduced Hessian SQP. To obtain the *Y* space move, the algorithm employs a stable decomposition scheme, based on multiple shooting to solve the linear system resulting from the collocation equations, and therefore, instabilities associated with poor problem initializations of (DAE1) are avoided. Also included is a specialized line search procedure that does not require the explicit calculation of Lagrange multipliers. Furthermore, the efficiency of the algorithm is further enhanced by applying the new QP solver, QPKWIK. Several approaches have been introduced for modifying the SQP algorithm. Most of them fall into two classes: feasible and infeasible path approaches. For relatively simple to moderate size models, the feasible and infeasibie path approaches require almost identical CPU times. On the other hand, the infeasible path approach outperforms the feasible path counterpart for large and more nonlinear examples.

In the last section, we also proposed a general solution framework for the DAE optimization problem by using the idea of bilevel optimization. This process decouples the problem into two levels: the inner and outer problems, and can result in a more robust algorithm. Furthermore, we showed that our algorithm can be utilized as an inner problem solver of the bilevel DAE optimization framework. A preliminary study indicates significant increases in robustness with a reasonable CPU time, even with a finite difference approximation for the derivative evaluation in the outer problem.

Even though the results for the problems, solved using this framework, are very encouraging, the success of the framework critically depends on the outer problem performance, in particular, the outer problem gradient evaluation. This will be the focus of a future study.

7 ACKNOWLEDGMENTS

Financial support from the National Science Foundation through grant ASC-9213149 is gratefully acknowledged.

A Model Descriptions

A.I Model description for PS Polymerization problem

Here, we make the following assumptions that

1. quasi-steady approximation for radical species can be used,
2. and the density of the reactor content is constant.

The details of the derivation and kinetic data are given in [24]. Based on the above assumptions and mechanism, a mass balance for the monomer (m), the initiator (i), and the first moment of polymer (£) can be written as,

$$mintj \tag{79}$$

$$s.t. : \quad \dot{m} = 1.16 * 10^{9} * (2//)^{1}/^{2}\exp -21620/\#I\backslash l - c)^{l\cdot 2}(l - m)/g(m) \tag{80}$$

$$\dot{c} = 1.58 * 10^{15} * \exp -30800/\#T(l - c) \tag{81}$$

$$£ = 1.09 * 10^{14} * / * \exp -30800/i?T(l - c) \tag{82}$$

Table 8: Parameter values used in the Zeolite distribution example.

Parameters	Units	Base Cases	case numbers		
			1	2	3
f_{caz}	$*10^3 cm^3/mol/s$	10"	8.2	6.4	4.6
k_{am}	$*10^3\ cm^3/mol/s$	1	1	1	1
hz	$*10^2 5^{-x}$	8	6.6	5.2	3.8
k_{bm}	$\cdot 10V^{-1}$	1	1	1	1
k_{cz}	$*10^2\ cm^3/mol/s$	3	3	3	3
k_{cm}	$*10^2 cm^3/mol/s$	8	7	6	5
$DA,$	$*10^{-5}\ cm^2/s$	1	1	1	1
D_{Am}	$*10^{-5} cm^2/s$	103	102.4	101.8	101.2
D_{Bz}	$*10^{-4} cm^2/s$	1	1	1	1
D_{Bm}	$*10^{-4} cm^2/s$	103	102.4	101.8	101.2

$$[m\ c\ \xi](0) = [0\ 0\ 0] \quad (83)$$

$$m(t_i) = 0.5;\ \pounds\{t_f\} = 0.005 \quad (84)$$

$$g(kt/k_to) = 1.0 \qquad 0 \leq m \leq 0.3 \quad (85)$$

$$= 1.522 - 1.818m \quad 0.3 \leq \mathbf{m} \leq 0.8 \quad (86)$$

A.2 Model description for Zeolite distribution problem

With flat plate geometry as used in [20], the model of problem can be stated as,

$$\max S_B = -\frac{D_{Br}(dB/dx)}{D_{Ar}(dA/dx)}\Big|_R \quad (87)$$

$$s.t.: \quad D_{Ar}d^2A/dx^2 = kcrA + k_{aT}A^2 \quad (88)$$

$$D_{Br}d^2B/dx^2 = k_{br}B - k_{aT}A^2 \quad (89)$$

$$j_i \int_0^* udx = .05 \quad (90)$$

The kinetic and transport parameters are the weighted average of the zeolite and silica-alumina, for example

$$D_{Ar} = uD_{Az} + (1 - u)D_{Am}k_{ar} - uk_{az} + (1 - u)k_{am}$$

The boundary conditions are given by

$$\wedge A(O) = \wedge fl(O) = 0 \quad (91)$$

and at the surface

$$A(R) = 1.6 *10^{s} mo//cm^3 \quad (92)$$

$$B(R) = 0\ mol/cm? \quad (93)$$

References

[1] Ascher, U.M., R.M. Mattheij, and R.D. Russell, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, Prentice-Hall, Englewood Cliffs, NJ (1988)

[2] Ascher, U.M. and L.R. Petzold, "Projected Implicit Runge-Kutta for Differential Algebraic Equations" *SUM J. Num. Anal.*, 28, 4, p. 1097 (1991)

[3] Ascher, U.M., H. Qin and S. Reich, *Stabilization ofDAE's and Invariant Manifolds*, University of British Columbia Technical Report 92-17 (1992)

[4] Ascher, U.M., and R.J. Spiteri, *Collocation Software for Boundary Value Differential-Algebraic Equations*, University of British Columbia Technical Report 92-18 (1992)

[5] Bader, G., U.M.Ascher, "A New Basis Implementation for A Mixed Order Boundary Value ODE Solver", *SIAM J. Sci. CompuL*, 8(4) , 483 (1987)

[6] Balakrishna S., Ph.D. Dissertation, Carnegie Mellon University, Pittsburgh, PA (1992)

[7] Biegler, L.T. and J.E. Cuthrell, "Improved Infeasible Path Optimization for Sequential Modular Simulators II The Optimization Algorithm.", *Comput. Chem. Engng.* , 9, 295 (1985)

[8] Biegler, L.T.,J. Nocedal, and C. Schmid, "A Reduced Hessian Method for Large Scale Constrained Optimization", *SIAM J. Opt*, to appear.

[9] Bock, H. G., "Recent Advances in Parameter Identification Techniques for O.D.E.", *Numerical Treatment of Inverse Problem in Differential and Integral Equation*, Heidelberg, Federal Republic of Germany (1983)

[10] de Boor, C, *A Pmctical Guide to Splines*, Applied Math. Science V.27, Springer, NY (1978)

[11] Chamberlain, R.M., "Some Examples of Cycling in Variable Metric Methods for Constrained Minimization", *Math.Prog.*, 16 (3) 78 (1979)

[12] Chen, S. and W Jeng, "Minimum End Time Policies for Batchwise Radical Chain Polymerizatioiï, *Chem Eng.Sci.*, 33, 735 (1978)

[13] Chung, Y., Ph.D. Dissertation, Carnegie Mellon University, Pittsburgh, PA (1991)

[14] Fiacco, A.V., "Sensitivity Analysis for Nonlinear Programming Using Penalty Methods"*JIath.Prog.*, 10, 287 (1976)

[15] Gritsis, D., Ph.D. Dissertation, University of London, London, U.K. (1990)

[16] Han, S.P., "A Globally convergent Method for Nonlinear Programming", *J. Optim. Theory Applies.*, 22(3), 297 (1977)

[17] Hindmarch, A.C., *ODEPACK, A Systematized Collection of ODE Solvers, in Scientific Computing*, R.S. Stepleman et a!, (eds.), North-Holland, Amsterdam, (1983)

[18] Kolstad, CD. and L.S. Lasdon, "Derivative Evaluation and Computational Experience with Large Bilevel Mathematical Programs", *J. of Optimization Theory and Applications*, 65, 485 (1990)

[19] Logsdon, J. S., Ph.D. Dissertation, Carnegie Mellon University, Pittsburgh, PA (1990)

[20] Martin, G.R., C.W.White III and D.B. Dadyburjor, "Design of Zeolite/Silica-Alumina, Catalysts for Triangular Cracking Reactions", *Journal of Catalysis,* **106,** 116 (1987)

[21] Pantelides, C.C., "The Consistent Initialization of Differential-Algebraic Systems", *SIAM J.Sci.Stdt. Comput,* 9(2), 213 (1988)

[22] Pontryagin, L.S., V. Boltyanskii, R. Gamkrelidge and E. Mishchenko, *The Mathematical Theory of Optimal Processes,* Interscience Publishers Inc., New York (1962)

[23] Russell, R.D. and J. Christiansen, "Adaptive Mesh Selection Strategies for Solving Boundary Value Problems", *SIAM J. Numer. Anal.,* 15(1), 59 (1978)

[24] Sacks, M.E., S. Lee and J.A. Biesenberger, "Optimal Policy for Batch, Chain Addition Polymerization", *Chem Eng.Sci.,* 27, 2281 (1972)

[25] Schmid, C. and L.T. Biegler, "Quadratic Programming Methods for Tailored Reduced Hessian SQP", submitted to *Comput. Chem. Engng.* (1993)

[26] Schmid, C, P. Tanartkit and L.T. Biegler, Simultaneous Flowsheet Optimization Using Tailored Reduced Hessian SQP, presented at AIChE Annual Meeting (1993)

[27] Vasantharajan, S., Ph.D. Dissertation, Carnegie Mellon University, Pittsburgh, PA (1989)

[28] Vassiliadis, V., Ph.D. Dissertation, University of London, London, UK.(1993)

[29] Wolbert, D., X. Joulia, B. Koehret, and L.T. Biegler, "Optimal Flowsheet Sensitivity in a. Sensitivity Oriented Environment", *Comput. Chem. Engng.,* 17, S117-122 (1993)