1995

Stable decomposition for dynamic optimization

Purt Tanartkit
Carnegie Mellon University

Lorenz T. Biegler

Carnegie Mellon University.Engineering Design Research Center.

Stable Decomposition for Dynamic Optimization

Purt Tanartkit and Lorenz T. Biegler

EDRC 06-193-95

Stable Decomposition for Dynamic Optimization

Purt Tanartkit and Lorenz T. Biegler*

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

Abstract, Dynamic optimisation problems axe usually solved by transforming them to nonlinear programming (NLP) problems with either sequential or simultaneous approaches. However, both approaches can still be inefficient to tackle complex problems. In addition, many problems in chemical engineering have unstable components which lead to unstable intermediate profiles during the solution procedure. If the numerical algorithm chosen utilises an initial value formulation, the error from decomposition or integration can accumulate and the Newton iteration* then faiL On the other hand, by njiiigsmUhk^ecomposition, either through Multiple Shooting or collocation, our algorithm has favorable numerical characteristics for both stable and unstable problems; 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 Quadratk Programming (SQP) approach. The routine chosen for the decomposition of the system equations is CX)LDAE, in which the stable multipk shooting scheme is implemented. To address the mesh selection, we will introduce a new bilevd framework that wiD decouple the element placement from the optimal control procedure. We will also provide a proof for the connection of our algorithm and the calculus of variations.

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 and optimal control for various units. The motivation for this research comes from the fact that steady state simulation is 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.
- Processes with tight transient requirements, for example for safety, waste, and operability.
- Sophisticated control schemes, e.g., predictive, optimal or adaptive control.
- 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

*to whom all correspondence should be addressed.

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:

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

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

$$GW*), y(t)_f tt(0, \text{«}, p) \leq 0 \tag{3}$$

$$G_9(z\{t_9 | y(t_a), u[t_s), t_9, p) < Q \tag{4}$$

where $<p$ is **a** scalar objective function,

F **are** differential-algebraic equation constraints expressed in **an** implicit form,

tf is final time,

G are algebraic inequality constraints,

G_a **are point condition constraints (e.g., initial or final conditions) at** times t_9 including */,

z, y **are state profile vectors,**

u **are control profile vectors, 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 Logsdon and Biegler (1992)). 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 variations that was introduced in the 1960's (Pontryagin Maximum Principle (Pontryagin et al., 1962)). In the variational 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 addition of profile inequalities in the problem.

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 (see Vassiliadis (1993) 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 and this yields function and gradient values for the NLP sblver. An optimization routine is then applied in an outer loop to update the control actions. The advantage of this strategy lies in the 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 (Logsdon and Biegler, 1992; Vasantharajan and Biegler, 1990), on the other hand, the DAE solution and optimization are simultaneously converged through discretization of both the state and control profiles. Vasantharajan and Biegler (1990) 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 well as the elimination of expensive and possibly infeasible intermediate solutions.

The main drawback of this approach is its explosive problem size. To ease the dimensionality issue, an NLP decomposition was introduced, 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) and for DAEs with unstable components (Ascher et aL, 1988). In addition, the formulation 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:

1. the presence of algebraic equations (index problem),
2. the high dimensionality of the discretized system,
3. the solution of the state equations especially with unstable components, and
4. the determination of the finite elements for discretization.

Treatment of the first issue is discussed briefly in the next section and relies on previous work. The next two issues are the main foci of this study, although 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 is based on COLDAE (Ascher and Spiteri, 1992) 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 (Bader and Ascher, 1987) 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^{th} order differential equations at M collocation points by finding the coefficients of the polynomials in the form

$$z(t) = \sum_{j=1}^{Af+m} ajTjit) \qquad (5)$$

where Tj is an independent polynomial basis function.

The collocation method is equivalent to the Runge-Kutta method and its accuracy depends on the number and location of the collocation points, and the range of integration (for further details on stability of the Runge-Kutta method see Ascher et aJ. (1988)). In practice, the number of collocation points is usually small, because high degree polynomials tend to oscillate. In addition, global interpolation is 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 state 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 and control variables, the

optimization routine also determines the finite element lengths. The sizes of the finite elements are governed by stability, accuracy and the optimal locations of breakpoints for control profiles.

In this work, the monomial basis is used to approximate all the state profiles as recommended in Bader and Ascher (1987) because of its smaller condition number and smaller rounding errors. On the other hand, the control and algebraic profiles are evaluated using Lagrange polynomials to accommodate the discontinuities at the element boundaries. For an m^{th} order ODE, the state profiles are approximated by

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

where Zij , $W\{j$ are unknown polynomial coefficients, and
7j is a polynomial of order $M + rn$ satisfying,

$$\frac{d^{l-1}}{dt^{l-1}} \gamma_j(0) = 0 \quad /or/=1,\ldots,ro; \quad j = 1,\ldots,M \tag{7}$$

$$\frac{d^m}{dt^m} \gamma_j(\varrho_r) = \delta_{jr} \quad /orr = 1,\ldots,M; \quad j = 1,\ldots,M \tag{8}$$

where g_T is the collocation point within each element. Here, unlike Lagrange polynomials, the variables *zu* represent values of the state variable and the first m - 1 derivatives at the beginning of element *i* while tuy represent the m^{ih} derivative of the states in element *i* and collocation point

Although the local error can be kept small by using small element sizes or high order integration schemes, several researchers (Ascher et aL, 1988) 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 (Gear, 1986). The index can also be viewed as the order of the state constraints in the calculus of variations plus one (see Bryson and Ho (1975)).

There have been several approaches to develop a systematic way for detecting the structural index of the problem. For instance, Pantelides (1988) proposed an algorithm using graph representation, and Chung and Westerberg (1990) 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 and Petzold (1991), is to use an appropriate integration scheme, such as Radau collocation or projected Runge-Kutta collocation on the original formulation (Logsdon and Biegler, 1989). As a result, the algebraic constraints are enforced directly and this avoids the need to employ the stabilization scheme for the invariants. 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.

2.1 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. Common numerical algorithms for solving ordinary differential equations

are usually based on one of the following two ideas (see Figure 1):

- Single Shooting (SS), in which the solution is sought sequentially in time from the initial conditions (or guessed initial conditions). For the boundary value problem, a Newton iteration is then used to adjust the *guessed* initial condition so that the boundary conditions are equal to the boundary conditions given.
- Multiple Shooting (MS), in which the integration horizon is divided into smaller subintervals. Then the system equations are solved by collocation or a direct integration is performed along a nominal trajectory on each subinterval and the resulting sections are combined to find the solution. Newton iteration is also needed to enforce the continuity between elements as well as to adjust the nominal trajectory.

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 small 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 tends to accumulate in a disastrous fashion and even good problem initializations are often not adequate for successful solutions. In addition, many problems in chemical engineering are naturally BVP problems, including kinetic and transport problems in various geometries. These difficulties can be reduced 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 the finite element representation that we used for approximating the profiles.

To illustrate the impact of the boundary conditions and the integration procedure, consider a linear problem given as,

$$\dot{V}2 = r*yi+g(t) \quad (9)$$

$$\dot{V}i = V2 \quad (10)$$

As seen here, the eigenvalues (poles) of the homogenous part of the problem are r, —r, and therefore this problem has both increasing and decreasing modes. Bock and coworkers (Bock, 1983) have considered this problem with,

$$g(t) = (*^2\text{-}r^2)\sin 0r< \quad (11)$$

The solution of this problem (9-11), for any value of r, is given by

$$yi(t) = \sin(x*) + c_x \exp(-r*) + c_2 \exp(r*) \quad (12)$$

$$y_2\{t) = 7r\cos(7r<)\text{-}ci\exp(-r*) + c_2\exp(r*) \quad (13)$$

The constants (ci,c_2) are determined later using the side conditions. Equation (9-11) can be formulated as an initial value problem (IVP1) with the following conditions,

$$y_1(0) = 0 \quad (14)$$

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

Let 6i,€2 be the round-off errors for the initial conditions yi(0) and jfe(0) respectively, then the solutions to this set of initial conditions (assume that there is no other error in the integration) are:

$$y_1(t) = \sin(\pi t) + (\epsilon_1/2)(\exp(\tau t) + \exp(-\tau t))$$

$$+(\epsilon_2/(2r))(\exp(rt) - \exp(-rt)) \quad (16)$$

$$y_2(t) = \pi \sin(\pi t) + \tau(\epsilon_1/2)(\exp(\tau t) - \exp(-$$
$$+(\epsilon_2/2)(\exp(\tau t) + \exp(-\tau t)) \quad (17)$$

A direct substitution of the initial conditions into the solution (16-17) shows that the sensitivities of the solutions to these conditions axe exponential functions with respect to t, r. As a consequence, the errors of the solution will propagate rapidly even for very small round-off errors. The error for SS and MS from round-off in the initial conditions then can be written as:

$$[SS] \quad e_{,,}(l) - O(Tap(r))\|\epsilon_U \epsilon_2\| \quad (18)$$

$$[MS] \quad e_m,(l) \sim 0(rexp(r/iV))\|£_1,6_2\| \quad (19)$$

Here, we assume the time horizon [0,1] is divided into N equal distance intervals for the MS integration. Note the error for the MS method is only the N^{th} root of the SS method.

On the other hand, if we specify boundary conditions (20 -21) for this problem (IVP1) instead of initial values (14-15), we have (BVP1):

$$y_1(0) = 0 \quad (20)$$

$$Sfi(l) = 0 \quad (21)$$

Note here that the solutions for both (IVP1) and (BVP1) are the same. Using the same procedure as the previous case reveals that the error generated by the round-off in the conditions has a linear dependency with respect to both t, and r.

$$y_1(t) = \sin(\pi t) + \frac{\epsilon_2 - \epsilon_1 \exp(-\tau)}{\exp(\tau) - \exp(-\tau)} \exp(\tau t) +$$
$$\frac{-}{\exp(r) - \exp(-\tau)} \frac{\epsilon_1 \exp(\tau)}{} \exp(-\tau t) \quad (22)$$

$$y_2(t) = \pi \cos(\pi t) - \tau \frac{\epsilon_2 - \epsilon_1 \exp(-\tau)}{\exp(\tau) - \exp(-r)} \exp(\tau t) +$$
$$\tau \frac{-\epsilon_2 + \epsilon_1 \exp(\tau)}{\exp(\tau) - \exp(-\tau)} \exp(-\tau t) \quad (23)$$

In this formulation, i.e. with boundary condition, only the MS method is applicable and the error can be approximated as:

$$\epsilon_{ms}(1) \sim O(\tau)\|\epsilon_1, \epsilon_2\| \quad (24)$$

2.2 Problem Conditioning

As seen above, the (IVP1) formulation can be extremely sensitive to the structure of the side conditions. In order to understand the computational aspects of the problem, we separate the origin of the failure into two major sources. The first is the algorithm-dependent instability, i.e. errors, either roundoff or discretized errors, can be magnified along the trajectory. 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 *Mr conditioned.* This second cause cannot be avoided simply by selecting a more stable method.

To illustrate the influence of the problem formulation, we consider the following linear BVP (25-26) and provide an analysis from Ascher et al. (1988).

$$\mathbf{y}' = A(x)\mathbf{y} + \mathbf{q}(x) \tag{25}$$

$$B_a\mathbf{y}(a) + B_b\mathbf{y}(b) = \beta \tag{26}$$

$$*\text{€}[a, \&] \tag{27}$$

The solution **to** this problem can **be** written as:

$$\mathbf{y} = *(x)(5_o*(a) + 5_6*(6))^{-1}/? + \int_{Ja} tG(x,t)q(t)dt \tag{28}$$

where *9(x)* is a fundamental solution, and
G(x,t) is the Green's function of the problem.

From **(28), a bound** on error can be obtained as:

$$\|\mathbf{e}\| \leq \kappa(\|\delta\beta\| + \|\delta\mathbf{q}\|) \tag{29}$$

$$\kappa \geq \|\Psi(x)(B_a\Psi(a) + B_b\Psi(b))^{-1}\| \tag{30}$$

$$K \geq \max_{*} \int_{Ja}^{b} \|G(a:, t)\|dt \tag{31}$$

where *6(3* and *Sq* are perturbations on /? 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* (*) to quantify this phenomenon, 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 Ascher et al. (1988).

For (BVP1) above (equation (9-11) with boundary conditions (20-21)), the corresponding BVP problem is well-conditioned with condition number of the problem about *O(r)* for r » 1. Thus, if any stable integration scheme is applied, the solution is stable and accurate for a moderate r. For the multiple shooting solver, we employ the COLDAE routine, a general purpose boundary value solver based on multiple shooting. COLDAE is publicly available through NETLIB (netlib@ornl.gov). The discussion of this routine as well as its algorithm will be given later in section 3.2.

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

Now, with the MS method and IVP formulation the condition number of the problem still remains the same, however the domain of the integration is drastically shortened and that leads to a smaller error compared to the SS method. To illustrate this point, the (IVP1) formulation is solved using both SS (LSODE (Hindmarch, 1983) with tolerance of 10^{-6}) and MS (COLDAE (Ascher and Spiteri, 1992) with tolerance of 10^{-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 *w* is correct up to 9 digits. The result is not surprising because the problem

is ill-posed, and (IVP1) cannot be solved with any IVP integrator. The results from MS method are almost identical for both (BVP1) and (IVP1) and both cannot be seen as separate lines on the Figure 2, although the computational time of the MS method with (IVP1) is about double that of the (BVP1) formulation.

This example has implications for the choice of optimal control method, because any method based on single shooting will fail on this problem. For IVP 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 $L \sim r$. Also, note that the SS algorithm will be uniformly stable if L has a negative real part. We must also point out that the MS method only reduces the length of the integration, and therefore may not work for all unstable problems either.

As seen, with this small and linear system, the stability and accuracy of the solution depend inevitably on not only the solution method but also the formulation. The unstable system in chemical engineering problem is not uncommon, especially in control and simulation. For example, in model predictive control and in process identification, the prediction or model evaluation steps can encounter unstable solutions , or the dynamic simulator may have to integrate unstable intermediate trial points. IVP methods may not have the capability to handle these situations, the problem then cannot be solved at all even when the final solution is stable. It is clear from the example that in order to solve realistic process problems, the numerical algorithm needs to be able to deal with not only stable problems but also unstable systems as well.

In this section we demonstrated that the problem formulation, specific ally boundary conditions, can affect the performance 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 analytical 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. In the next section we will develop a decomposition for the dynamic optimization based on the ideas discussed in this section.

3 Fixed-Mesh Problems

Here, we adopt the multiple shooting approach and substitute a collocation method in each element. We also assume for the moment that the element lengths are fixed, and we will later show that, with another layer of optimization, we can prove the equivalence of the algorithm to the calculus of variations. With a fixed mesh, we have the following problem statement (NLP1)

$$\min_{z_i, \dot{z}_i, y_{ij}, u_{ij}, p, t_f} \wedge\{zijZi, yijjUijyPjhi, \bar{tf}\tilde{}) \tag{32}$$

s. t. : Discretized DAE model:

$$F(z_i, \dot{z}_i, y_{ij}, t^*ij, p) = O \tag{33}$$

$$G(z_i, \dot{z}_i, y_{ij}, tx, ;, p) \leq 0 \tag{34}$$

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

point condition:

$$G_k(z\%, \dot{z}\text{»} Vijy\ v\text{>}ij, p) \leq 0 \tag{35}$$

$$for\ k = 1, \ldots_7 n5$$

bounds:

$$z^L \le *ij \le *^U \quad (36)$$

$$V^L \le Vij \le y^U \quad (37)$$

$$u^L \le un \le u^u \quad (38)$$

$$P^L \le P \le P^U \quad (39)$$

$$t^L \le t_f \le t^u \quad (40)$$

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

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 h_i 's are fixed, and we will discuss the variation of h_i '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. The structure of our algorithm is given in Figure 3. Its main components are discussed in the remainder of this section. At each iteration &, a quadratic programming subproblem (QP1) is created from (NLP1) and solved to obtain a search direction *d* for x, of the form:

$$\min_{d} \quad \nabla\Phi(x^k)^T d + \tfrac{1}{2} d^T B^k d \quad (41)$$

$$s.t. \quad c(x^k) + \nabla c(x^k)^T d = 0 \quad (42)$$

$$d \in [x^L - x^k,\ x^u - x^k] \quad (43)$$

$$x = [z,\ y,\ u,\ P,\ t_f]^T \quad (44)$$

A decomposition algorithm can then be applied by partitioning variables into decision (control *(z))* variables and dependent (state *(y))* variables with coordinate basis. Thus, the search direction is divided into:

$$d = (Yp_v)^k + (Z_A)^* \quad (45)$$

where *Z* and *Y* are defined by:

$$Vc^T Z = [\ Vc_1\ \ Vc_2\ ]Z = 0. \quad (46)$$

We then choose

with the range space direction (Yp_y) obtained by :

$$(Vc^T Y)p_y = -c(x^k) \quad (48)$$

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

$$\min_{p} \quad V\$(x^k)^T p + \tfrac{1}{2} p_z^T (Z^T B Z) p_z + \tfrac{1}{2} (Z^T B Y p_y)^T p_z \quad (49)$$

$$s.t. \quad x^k + Yp_y + Zp_z \in [x^L, x^u] \quad (50)$$

Here (Z^TBZ) is a quasi-Newton approximation to the reduced Hessian of the Lagrange function and (Z^TBYp_y) can be approximated through various options (Biegler et aL, 1994). The QP subproblem was solved using the QPKWIK algorithm described in Schmid and Biegler (1993). QPKWIK is specifically tailored to solve large scale problems by updating the inverse Cholesky factors of the reduced Hessian (Z^TBZ); therefore, it is $O(n^2)$ 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 (48), or equivalently, a Newton step for the system equations. Therefore, if the problem is ill-posed, in other words, the system is too sensitive to round-off error, then Yp_y resulted from the system equations is unreliable and that leads to unreliable overall linesearch directions as well.

By using reduced SQP, model sparsity is maintained for the simultaneous or equation oriented approach. The system is usually large when the number of equations and the number of finite elements increase. The complete system consists of a square system of *ncol* • *(ncomp* + *ny)* * ne + *ncomp* * (ne + 1) variables, where *ncomp,* ny, *ncol* and ne denote numbers of differential profiles, algebraic profiles, collocation points and finite element respectively.

While a number of integration packages is available, the routine chosen for evaluating this move is COLDAE (Ascher and Spiteri, 1992). 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 for both initial value and boundary value problems. 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. Moreover, the evaluation of *Z* (47) can be performed by employing the factorized matrix in COLDAE with different right hand sides and this further reduces the computational expense. Because COLDAE is only used to set up and to decompose the system equations for fixed elements, the error estimate and the mesh selection in the code are suppressed. The discussion of error estimate and mesh selection in our algorithm will be deferred to the future development section.

In COLDAE, the collocation systems have two types of unknowns: global and local variables. Local variables *(ncol * (ncomp+ny) * ne)* are evaluated and 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. With the independence of the local variables, this implies that we can solve them in a parallel fashion and further decrease overall computational time. The resulting linear system of global variables *(ncomp* • (ne + 1)) is cast in an almost block diagonal form and solved by de Boor's sparse SOLVEBLOK routine (de Boor and Weiss, 1980), in which the number of computational steps is only linear with respect to the number of finite elements. Moreover, SOLVEBLOK is implemented so that the elimination is done without additional storage. It is crucial to point out that both the reductions in computational effort and storage space are achieved due to the structure of the system equations that have constant blockwidth. These characteristics are vital to our ability to tackle large scale problem from the computational point of view. The implementation with COLDAE allows the following two optimization strategies.

Feasible Path

In the feasible path approach, the state equations are satisfied exactly in an internal loop;

therefore, both the *Y* space move and *c{x)* are zero. This approach has the same concept as the sequential approach described in section 1, as it 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. This 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). But for a nonlinear system, this approach might be too slow because the system of nonlinear equations has to be solved at each step.

Infeasible Path

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

$$A = -(Y^T Vc)^{-1} Y^T V^* \tag{51}$$

Unfortunately, using COLDAE, the matrix $(y^{\wedge}\bar{V}c)^{-1}$ is not readily available. As a result, a new linesearch algorithm must be investigated, instead of the Augmented Lagrange (Biegler and Cuthrell, 1985) function used in feasible path. Han (1977) showed that using the exact penalty function ($(x)+/i||c(x)||i) as the linesearch merit function, the SQP converges to a local optimizer from poor starting points. However, the penalty value (/i) has to satisfy the inequality,

$$/i > \text{IIAllloo} \tag{52}$$

Here a different penalty parameter can be used instead of *fi* to obtain the same global convergence property, without individual multipliers explicitly calculated. The penalty parameter *p!* is defined by (53).

$$if > lA^{\wedge}I/Mh \tag{53}$$

As shown in Schmid and Biegler (1994), the term $|A^T c|$ can be obtained easily by the factorized matrices from COLDAE. The exact penalty function method is usually slower (Biegler and Cuthrell, 1985) to converge than with other linesearch functions, so to accelerate convergence, we use a *watchdog* technique (Chamberlain, 1979) that allows intermediate increases in the merit function but requires a reduction in the exact penalty every *q* steps (in our study *q* = 2).

Our algorithm can therefore be summarized in Figure 3. No te that the structure is modular and relatively easy to construct and adapt to new applications.

4 Examples

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

4.1 Polymerization Problem

This example is an optimal control problem for 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 (Chen and Jeng, 1978; Sacks et al., 1972) 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.

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 taken from Vassiliadis (1993). The model is described by,

$$\text{minjfe}(5) \tag{54}$$

$$s.t.: \quad \dot{y}_t = (1 - yl)y_x - y_2 + u \tag{55}$$

$$il2 = yi \tag{56}$$

$$\dot{y}^* = yl + yZ + u^2 \tag{57}$$

$$y(0) = [0.0 \ 1.0 \ 0.0]^T \tag{58}$$

$$u(t) \in (-0.3 \ 1.0) \tag{59}$$

As expected, the infeasible path approach performs much better for both Van der Pol (Figure 5 and Table 2) and PS polymerization examples (Figure 4 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 (60) is active.

$$y \geq -0.4 \tag{60}$$

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 6. These results are also compared with the solution obtained with an IVP approach by Vassiliadis (1993).

4.3 Parameter Estimation Problem

This problem is a parameter estimation problem taken from Bock (1983). The system involves two differential equations and one unknown parameter. The state profiles are initialized to 1.0 and the unknown parameter (p = x) is set to 2.0. The model of this example can be defined as a boundary value problem (BVP) or an initial value problem (IVP) by using different side-conditions. The detail formulations are given in the Appendix, with r set to 60. The objective of this problem is to estimate the values for the parameter equal to x given true values corrupted with random noise $(N(0_y a))$ at data points 0,0.1,..,1.0.

For the BVP problem, the condition of the decomposition is linear with respect to time and the parameter r, and the resulting problem is therefore well-conditioned.

We have also solved the problem in the case that the conditions provided are given in the form of an IVP. However, as analyzed in section 2, this problem has both increasing and decreasing

modes in the solution. In other words, the problem is ill-posed, with the condition number increases exponentially with respect to both final time and the parameter r. Note that the condition of the problem is reflected in the quality of the *Y* space move and it depends solely on the side conditions in the decomposition. From the stability standpoint, this problem is much more stable if posed as a BVP; however, the final condition is not given from the IVP problem statement.. To improve the condition, we employ the BVP formulation with an end-condition as a decision variable and a constraint added to the optimization problem that relates the known initial value to this decision variable. As a result, the problem has stability properties of the BVP, even though the problem is given in IVP problem statement. This formulation (PBVP) can be found in the appendix.

The results for all the 10-element cases agree very well with the true value as shown in Figure 7. Note also that the un-modified IVP problems with 10 elements require slightly more effort but for the 30-element case the algorithm fails to converge (line search failure) due to problem conditioning as shown in Section 2.2. These difficulties in the IVP formulation increase with increasing *r* and number of steps.

Comparisons between (PBVP) and (BVP) problems show that both perform equally well as shown in Table 5. In Figure 7, all three cases that converged superimpose on the same line in the graph.

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 Figure 8.

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 Martin et al. (1987). Farther details of the model and data are given in the appendix.

The problem is solved using 6 elements and 3 collocation points. As shown in Figure 9, 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 (Martin et al., 1987).

5 Future Development

We consider the following DAE optimization problem with 2 diiferential equations, that 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 *m/s.* This problem has an analytical solution given in Logsdon and Biegler (1992).

$$\min_{u(\cdot)} \; t_f \tag{61}$$

$$\text{s.t.} \quad \dot{z}_x = u \tag{62}$$

$$i_2 = z\backslash \tag{63}$$

$$z(\dot{0}) = [0.0,\ 0.0]^T \tag{64}$$

$$z(f_f) = [0.0,\ 300.0]^T \tag{65}$$

$$u(t)\ e\ [-2.0,\ 1.0] \tag{66}$$

The results are obtained by using 6-uniform and 3-optimally spaced (from analytical solution) elements and are given in Figure 10 and Table 7. 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 element placement that is usually performed in most sequential optimal control packages is done in both the integrator for the state variables and the optimization routine for the control variables, and this leads to the solution with redundant elements. The more elements the problem has, the more time-consuming it will be.

On the other hand, if the element placement is done in one stage, i.e. in the optimization routine, the element placement then has to be included in the formulation. 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 Russell and Christiansen (1978). In this study, we incorporate the following relation,

$$\|e(t)\| = C\|\xi(t_{nc})\|h + O(h^{k+1}) \tag{67}$$

where *hi* is step size,
k is the order of the method,
||e(t)|| is approximated local error, and
j|€(inc)l! *s the norm of the residual at non-collocation point.

as proposed in Russell and Christiansen (1978). Next we enforce constraint (67) without the O(/**$^{+1}$) term by bounding the residual for each equation, including algebraic equations, at non-collocation points as,

$$6 = F(\text{«}\dot{Z}ih\ VH, {}^{u}ih\ P> {}^{f}nc) \tag{68}$$

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) :

$$\bullet \min_{*\text{€}\,i*ij\,iVij\,.**\text{›}\,>P>{}^{h}i\,\text{»}*/} H\text{^}Z\dot{i}j.yij.Uijyp.hi.tf) \tag{69}$$

s» t. : Discretized DAE model:

$$\text{^«^ii;yii^ii,p,O} = \mathbf{o} \tag{70}$$

$$G(zi,\dot{Z}i_h yij, Ui_h p,t)<\{S \tag{71}$$

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

$$\overset{nc}{5} >= *, \tag{72}$$

side condition:

$$G_k(z_i, \dot{z}_i, y_{ij}, u_{ij}, p, t_k) \leq 0 \tag{73}$$

for $k = 1,\ldots,$ ns

bounds:

$$*^{L} \leq Zij \leq z^{u} \quad (74)$$

$$\wedge\wedge ya\wedge y^{17} \quad (75)$$

$$u^{L} \leq u_{ti} \leq t\wedge \quad (76)$$

$$P^{L} \leq P < P^{U} \quad (77)$$

$$t^{L} \leq t_{f} \leq t^{u} \quad (78)$$

for i = l,...,ne; j = l,...,nco/

element placement constraint:

$$-\epsilon \leq \xi_i = F(z_i, \dot{z}_i, y_{ij}, u_{ij}, p, t_{nc}) \leq \epsilon \quad (79)$$

where t **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 (DAEl) is transformed into a nonlinear programming problem (NLP2). Problem (NLP2) can then be solved with any large scale NLP solver as in Vasantharajan and Biegler (1990). However, element placement constraints (79) 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. Moreover, the element placement constraints are not explicitly present in the calculus of variations derivation for optimality, therefore the connection is not dear.

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 placement constraints do not influence the determination of the optimal control profile directly. In particular, if the control is differentiable 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. Furthermore, with this **formulation the derivation for the equivalency of our algorithm and the** calculus **of** variations can be derived, **the** derivation is **provided in the Appendix B. The outer** problem (NLP3) can be written as:

$$\min_{hi} \quad *(ai, *(hi), jr(hi)_f u*(hi), \backslash *(hi)) \quad (80)$$

$$s.t. \quad £A = 1 \quad (81)$$

$$hie[h^{L}, h^{u}] \quad (82)$$

$$-\{-error\,control\,contraints \quad (83)$$

where $u^{m}{}_{J} z^{m}{}_{J} y^{m}{}_{J} t^{1}{}_{J}$ and A* are implicit functions of the /&,'s defined by the inner (or fixed-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.

In this section, we first consider the car problem 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 8 converged to the analytic solutions. In contrast, the algorithm used in Logsdon and Biegler (1992) 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. We also re-solved the polymer reactor example in section 4 using the bilevel 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 profiles are continuous and the errors in the integration are well within the tolerance.

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 finite elements for optimal control problems. 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 (Vasantharajan and Biegler, 1990). With this goal, our bilevel 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. In particular, we have successfully demonstrated our approach on problems that fail with sequential algorithms based on IVP solvers. 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 infeasible path approaches require almost identical CPU times. On the other hand, the infeasible path approach outperforms the feasible path counterpart for larger 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. Furthermore, we showed that our algorithm can be utilized as an inner problem solver of the bilevel DAE optimization framework.

Intermediate future work will be directed to deal with large scale problems. As seen the parameter estimation problem, the formulation of the stable side conditions from unstable initial conditions is given and therefore the condition of the problem improves. However, the procedure is not general enough to deal with large scale problems, and this is also another important issue for

further investigation.

This paper therefore discussed an efficient and robust strategy for optimal control problems when finite elements are fixed. Extending this formulation to variable elements, we see that the preliminary results are very encouraging. However, the success of the framework critically depends on the outer problem performance, in particular, the outer problem gradient evaluation. This will also be the focus of a future study.

7 ACKNOWLEDGMENTS

Financial support from the National Science Foundation through grant ASC-9213149 is gratefully acknowledged. The authors thank João Albuquerque for the assistance in the stability discussion section.

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 Sacks et al. (1972). 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,

$$\min tf \tag{84}$$

$$s.t.: \quad \dot{m} = 1.16 * 10^9 \bullet (2/J)^{1/2} \exp -21620/J\&r(1 - c)^l l^2\{l - m)/g(m) \tag{85}$$

$$\dot{c} = 1.58 * 10^{15} * \exp -30800/\#T(1 - c) \tag{86}$$

$$i = 1.09 \bullet 10^{14} * / * \exp -30800/£T(1 - c) \tag{87}$$

$$[me\ f](0) = [0\ 0\ 0] \tag{88}$$

$$m(t_t) = 0.5; \quad t(t_f) = 0.005 \tag{89}$$

$$\#t/y = 10 \quad 0 \le m < 0.3 \tag{90}$$

$$= 1.522 - 1.818m \quad 0.3 \le m \le 0.8 \tag{91}$$

A.2 Model description for Zeolite distribution problem

With flat plate geometry as used in Martin et al. (1987), the model of problem can be stated as,

$$\max S_B = -\frac{D_{Br}(dB/dx)}{D_{Ar}(dA/dx)}|_R \tag{92}$$

$$s.t.: \quad D_{AT}d^2A/dx^2 = k\wedge A + k_{ar}A^2 \tag{93}$$

$$D_{Br}d^2B/dx^2 = kkrB\text{-}KrA^2 \tag{94}$$

$$\frac{1}{R}\int_0^R u dx = .05 \tag{95}$$

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

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

The boundary conditions are given by

$$\frac{d}{dx}A(0) = \frac{d}{dx}B(0) = 0 \tag{96}$$

and at the surface

$$A(R) = 1.6 * 10^{s} mol/cm^3 \tag{97}$$
$$B(R) = Qmol/cm^3 \tag{98}$$

A.3 Model description for Parameter Estimation problem

A.3.1 Initial Value Formulation (IVP)

$$\min_p \; R = \sum_{i=1}^{N} (y_m(i) - y(0)^2 \tag{99}$$
$$s.t.: \quad \dot{y_2} = T^2 V! - (r^2 + p^2) \sin(pt) \tag{100}$$
$$\dot{V}i = V2 \tag{101}$$
$$y_1(0) \leftarrow 0 \; ; \; y_2(0) \leftarrow * \tag{102}$$

A.3.2 Boundary Value Formulation (BVP)

$$\min_p \; ii = Efci(ym(0 - 2/(t))^2 \tag{103}$$
$$s.t.: \quad \dot{y_2} = r^2 y_1 - (r^2 + p^2) \sin(pt) \tag{104}$$
$$\dot{y_1} = y_2 \tag{105}$$
$$y_1(0) \leftarrow 0 \; ; \; y_1(1) \leftarrow 0 \tag{106}$$

A.3.3 Boundary Value Formulation with Initial Conditions (PBVP)

$$\min_{p4} R = £\& (y_{ro}(f) - y(i))^2 \tag{107}$$
$$s.*.: \quad \dot{y}2 = r^2_{yi} - (r^2 + p^2) \sin(pt) \tag{108}$$
$$Ji = 22 \tag{109}$$
$$y_1(0) \leftarrow 0 \; ; \; y_1(1) \leftarrow t \tag{110}$$
$$y_2(0) - \pi = f(\xi) - \pi = 0 \tag{111}$$

Note: variable ^ and equation (111) have been added to the optimization problem.

B Equivalence of Optimality and Variational Conditions for Projected Gauss Collocation Methods

A general DAE optimization problem can be given as follow (P):

$$W \quad (z(1)) \tag{112}$$

s.t.

$$\dot{z}(t) = /(s(f), «(*),*) \quad (113)$$
$$z(0) = 0 \quad (114)$$
$$z(t) \in [*',*''] \quad \text{or} \quad g(z(t)) \leq 0 \quad (115)$$
$$u(t) \in [«V] \quad \text{or} \quad q(u(t)) \leq 0 \quad (116)$$
$$t \; 6 \; [0,1] \quad (117)$$

Note that any problem can be modified to the above formulation and algebraic equations can also be included. Let $H(t) = X(tff(t) + v(t)^T g(z\{t)) + ii(t)^T q(u(t))$
Optimality Condition:

$$V_u H(t) = 0 \quad (118)$$
$$\dot{A} = -(A(<)^r V,/(t) + i/(0^T l7(z(0) + M0^r V\wedge(ti(<))) \quad (119)$$
$$\dot{z}(t) = f(z(t),u(t),t) \quad (120)$$
$$\lambda(1) = V, v^T \quad (121)$$
$$g(z(t)) \leq 0 \quad (122)$$
$$?(«(<)) \leq 0 \quad (123)$$

with the following complementary conditions:

$$\mu(t) = 0 \; *\text{-}> \; q(u)<0 \quad (124)$$
$$ft(t) \geq 0 \quad \sim \quad ?(«) = 0 \quad (125)$$
$$\nu(t) = 0 \quad \wedge \quad \wedge)<0 \quad (126)$$
$$\nu(t) \geq 0 \; \text{•-»•} \; g\{z) = 0 \quad (127)$$

In addition, we assume that the problem is of index 2, or equivalently, the problem contains only $1^{<(}$ order state variable inequality constraints. As a result, at the constraint boundary, the following conditions must be enforced.

$$9\{z) = 0 \quad (128)$$

The derivation for the problems without path constraints are given in Reddien (1979) for both the first and second order conditions. Consider now the discretized problem (DP) corresponded to the continuous problem (P) above. Note here that the error control constraints are inactive at the optimal solution, and hence they are omitted here.

$$\underset{u(t)}{xamip\{z_N\{\backslash))} \quad (129)$$

S.t.

$$\dot{z}(\&) = /(*;), «o\text{-},fo) \quad (130)$$
$$*(0) = 0 \quad (131)$$
$$z_{N,i}(^h i) = {}^2«,0 \quad (132)$$
$$z_{ij} \in [z^l, z^u]$$

$$\text{or } g(zij) \leq 0 \tag{133}$$

$$\text{or } 9(\text{«o}) \wedge 0 \tag{134}$$

$$\text{or } r(z_{fc\cdot}) = 0 \tag{135}$$

The profiles are approximated over element ṫ by the polynomial *ZN* as follows:

$$z_{N,i}(t) = z_{i,0} + \sum_{j=1}^{ncol} \dot{z}_{ij}\psi_{ij} \tag{136}$$

$$*NM = \underset{3=0}{\overset{ncol}{\text{E}}} \text{V}*\text{v} \tag{137}$$

$$UN\text{-}UM = \overset{ncol}{\text{E}}\text{u}\dot{\&}\text{i} \tag{138}$$

where *if>*, *w*, *<f>* are defined by

$$Vt1>ij(t_k i) = fc_{ilW} \tag{139}$$

$$\wedge(i(0) = 0 \tag{140}$$

$$\varpi_{ij}(t_{kl}) = \delta_{ij,kl} \quad for\ j = 0,\text{-}s,ncol \tag{141}$$

$$\phi_{ij}(t_{kl}) = \delta_{ij,kl} \quad \{oxj=l,\text{---}_t ncol \tag{142}$$

$$\tag{143}$$

In addition, w;j's, ^,j's represent weights, collocation points and the Legendre polynomials respectively for the coefficient at collocation *j* in element *i*.

Multiply (130) by Aj^ny,, (133) by μT_{jWijy} (134) by $\$w_{ih}$ (132) by ijg, and (135) by x?. The Lagrangian for (NLP1) then can be given as:

$$L = \text{y(»w(l))} + \underset{i=1}{\overset{ne}{\text{E}}}\underset{j=1}{\overset{ncol}{\text{E}}} \wedge \text{i}\wedge\text{Ö}^T\wedge\text{i}(\text{-\%} + /(\wedge\text{»} \text{«0>fe})) + \wedge (z_{ij}) + \mu_{ij}^T w_{ij} + vT\&NAht) - \text{W}_*(*\text{D}) + *\text{fK}\wedge(\text{iM}) \tag{144}$$

Note that the polynomial *ZN* and A//$_{ft}$ are of degree *ncol ox* degree *ncol* - 1, and from the fact that Gaussian quadrature is exact for polynomial of degree *2ncol* — 1, therefore integration by part can be taken on the underlined section of (144).

$$\underset{i=i}{\overset{ne}{Yl}}\ \underset{i=i}{\overset{ncol}{\text{XI}}}\ WijtTjiVijZNA\&j) = -\underset{t=i}{\overset{ne}{\text{Z}}})\ \underset{j=i}{\overset{ncol}{\text{Z}}})\ \text{Wtf}\wedge\text{O\&itei} + \wedge NA^h i)^{Tz} N_t i(^h i) \sim {}^x To^z io \tag{145}$$

Since $A_{it}o$ is not present in the Lagrange function (144), we define them to satisfy the following expressions.

$$\lambda_{ti0} = \begin{cases} / \text{ Ajv,t-i(/ii-i)} & \text{if state bounds are inactive.} \\ \text{| A//}_{ft}\text{-»i(/i}_t\text{-\_i)} + \wedge iV_z g^T & \text{if state bounds in element are active.} \end{cases} \tag{146}$$

$$\text{and } 7\!/_t = A_{t-0} \tag{147}$$

Let $\bar{H}ij$ = *Xfjf(zij> Uijybj) + v7jq("ij)>* then the optimality conditions of (DP) are given by

$$\mathbf{V}\wedge\mathbf{Jftf} = \mathbf{0} \quad (148)$$

$$\nabla_{z_{ij}} f(z_{ij}, u_{ij}, \xi_{ij})^T \lambda_{ij} + \dot{\lambda}_{N,i}(\xi_{ij}) + \nabla_{z_{ij}} g(z_{ij})^T \mu_{ij} = 0 \quad (149)$$

$$\dot{z}_{N,i}(\xi_{ij}) - f(z_{ij}, u_{ij}, \xi_{ij}) = 0 \quad (150)$$

$$g(\text{«i}) \leq 0 \quad (151)$$

$$q(\text{tHi}) \leq 0 \quad (152)$$

$$r(z_{N,i}(h_i)) \leq 0 \quad (153)$$

$$z_{N,i}(h_i) = *\text{«+i,o} \quad (154)$$

for $i = 1, \bullet\bullet\bullet, ne$ and $j = 1, \bullet\bullet\bullet, ncol.$ with the following side conditions:

$$\wedge N,ne = \wedge 1) \wedge \quad (155)$$

$$z_{1,0} = 0 \quad , \quad (156)$$

$$g(z_{N,k}(h_k)) = 0 \quad \text{if the constaints are active in the element } k. \quad (157)$$

The above conditions are equivalent to the discretized formulation of the problem (P).

C REFERENCES

Ascher, U.M.; Mattheij, R.M.; Russell, R.D. *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations;* Prentice-Hall: Englewood Cliffs, NJ 1988; Chapter 3.

Ascher, U.M.; Petzold, L.R. "Projected Implicit Runge-Kutta for Differential Algebraic Equations". *SIAM J. Num. Anal.* 1991 28(4),1097-1120.

Ascher, U.M.; Spiteri, R.J. *Collocation Software for Boundary Value Differential-Algebraic Equations,* Technical Report 92-18: University of British Columbia, Vancouver, B.C. December, 1992.

Bader, G.; Ascher, U.M. "A New Basis Implementation for A Mixed Order Boundary Value ODE Solver". *SIAM J. Sci. Comput.* 1987, 8(4), 483-500.

Biegler, L.T.; Cuthrell, J.E. "Improved Infeasible Path Optimization for Sequential Modular Simulators II The Optimization Algorithm". *Comput. Chem. Engng.* 1985, 9, 257-267.

Biegler, L.T.; Nocedal, J.; Schmid, C. "A Reduced Hessian Method for Large Scale Constrained Optimization" *SIAM J. Opt* 1994, to appear.

Bock, H. G., "Recent Advances in Parameter Identification Techniques for O.D.E." In *Numerical Treatment of Inverse Problem in Differential and Integral Equation;* Deuflhard, P., Hairer, E.; Birkhäuser: Heidelberg, Federal Republic of Germany, 1983.

Bryson, A.E., Jr.; Ho, Yu-Chi *Applied Optimal Control;* Hemisphere: Washington, D.C. 1075.

de Boor, C; Weiss, R "SOLVEBLOK: A Package for Solving Almost Block Diagonal Linear Systems". *ACM Trans. Math. Software* 1980, 6(1), 80-87.

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

Chen, S.; Jeng, W. "Minimum End Time Policies for Batchwise Radical Chain Polymerization" *Chem Eng.Sci.* 1978, 33, 735.

Chung, Y.; Westerberg, A.W. "A Proposed Numerical Algorithm for Solving Nonlinear Index Problems". *I & EC Research* 1990, 29(7), 1234.

Gear, C.W."Maintaining Solution Invariants in the Numerical Solution of ODEs". *SIAM* ***J.Sci.Stat.Comp.*** **1986, 7, 734-743.**

Han, S.P., UA Globally convergent Method for Nonlinear Programming" *J. Optim. Theory Applies.* 1977, 22(3), 297-309.

Hindmarch, A.C., "ODEPACK, A Systematized Collection of ODE Solvers." In *Scientific Computing:* R.S. Stepleman et al. Eds; North-Holland, Amsterdam, 1983.

Logsdon, J. S.; Biegler, L.T. "Accurate Solution of Differential-Algebraic Optimization Problems" / *& EC Research* 1989, 28(11), 1628-1639.

Logsdon, J. S.; Biegler, L.T. "Decomposition Strategies for Large Scale Dynamic Optimization Problems" *Chem. Eng. Sci.* 1992,47(4), 851.

Martin, G.R.; White m,C.W.; Dadyburjor, D.B. "Design of Zeolite/Silica-Alumina Catalysts for Triangular Cracking Reactions" *Journal of Catalysis* 1987,**106,** 116.

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

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

Reddien G.W. "Collocation at Gauss Points as a Discretization in Optimal Control" *SIAM J. Cont. Opt* 1979,17, 298.

Russell, R.D.; Christiansen, J. "Adaptive Mesh Selection Strategies for Solving Boundary Value Problems" *SIAM J. Numer. Anal* 1978,15(1), 59-80.

Sacks, M.E.; Lee, S.; Biesenberger, J.A. "Optimal Policy for Batch, Chain Addition Polymerization" *Chem Eng.Sci.* 1972, 27, 2281.

Schmid, C; Biegler, L.T. "Quadratic Programming Methods for Tailored Reduced Hessian SQP" *Comput. Chem. Engng.* 1993,18(9), 817-832.

Schmid, C; Biegler, L.T. "A Simultaneous Approach for Flowsheet Optimization with Existing Modelling Procedures" *Trans IChemE* 1994, 72, 382-388.

Vasantharajan, S.; Biegler, L.T. "Simultaneous Strategies for Optimization of Differential Algebraic System with Enforment of Error Criteria" *Comp. Chem. Engng.* 1990, 14(10), 1083.

Vassiliadis, V. "Computational Solution of Dynamic Optimization Problems with General Differential-Algebraic Constraints". Ph.D. Dissertation. University of London, London, UK. 1993.

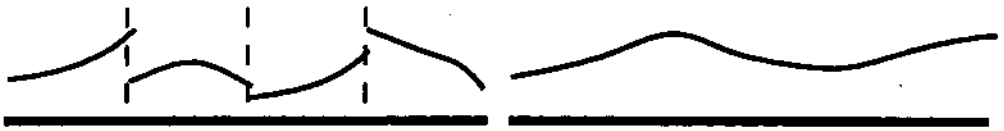

Figure 1: Multiple shooting (a) and single shooting (b)

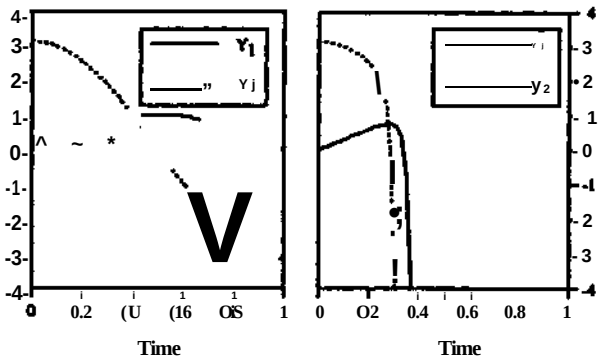

Figure 2: Multiple shooting for BVP and IVP (a) and single shooting for IVP (b)

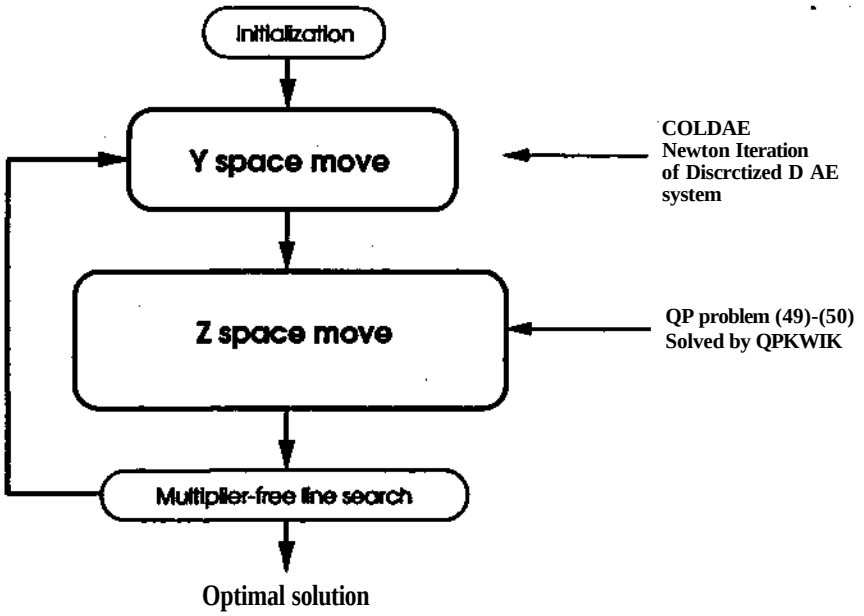

Figure 3: Reduced Hessian SQP algorithm

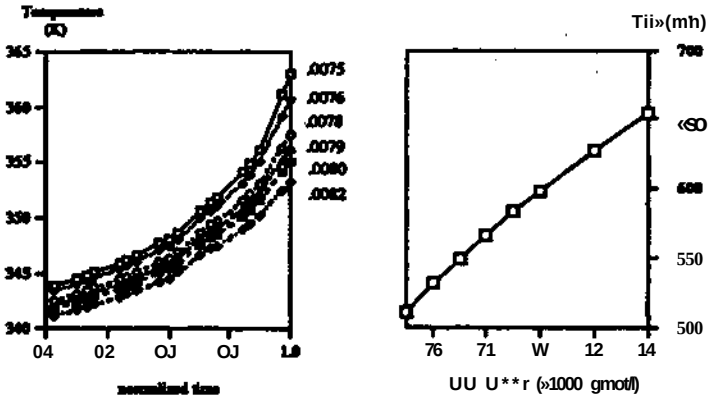

Figure 4: Optimal control profiles (a) and final times (b) for PS polymerization problem. The numbers right to the Figure (a) are the initial initiator concentration (gmol/1).

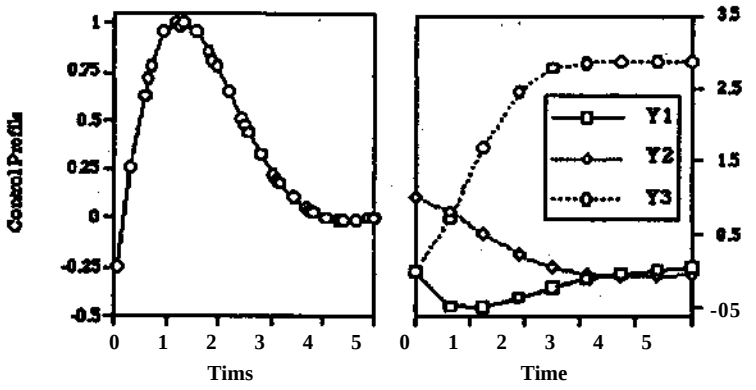

Figure 5: Optimal control state profiles for unconstrained Van der Pol oscillator problem.

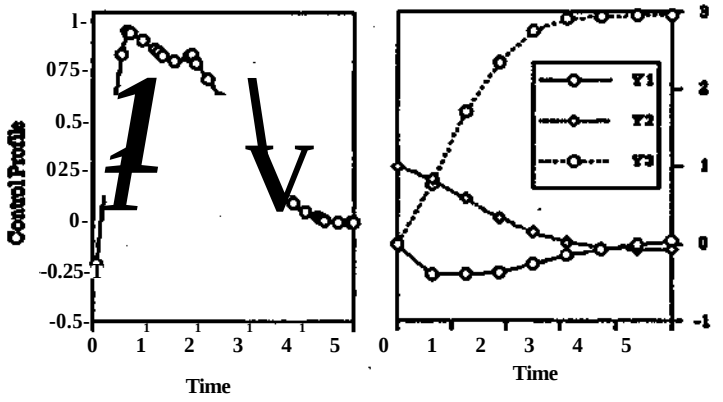

Figure 6: Optimal control and state profiles for Van der Pol oscillator problem with path constraints.

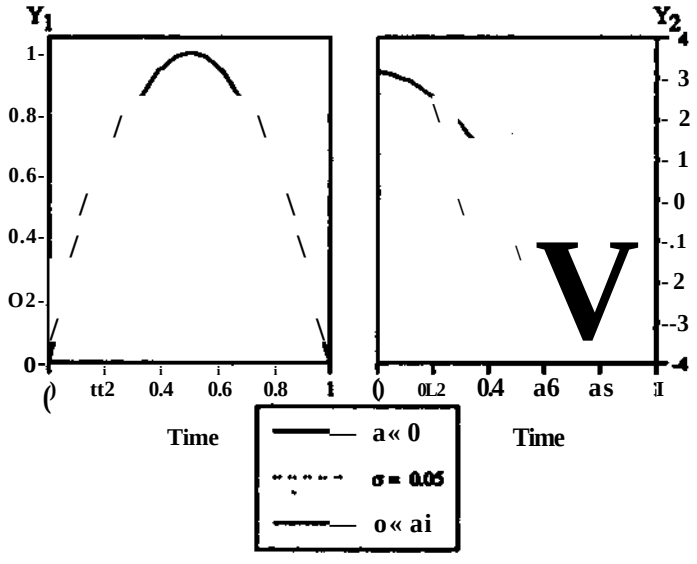

Figure 7: Optimal control and state profiles for Parameter estimation problem.

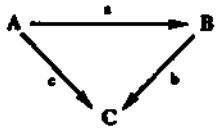

Figure 8: Schematic of Triangular reaction.

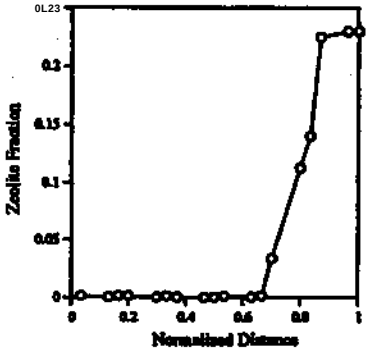

Figure 9: Optimal Zeolite profiles for base case.

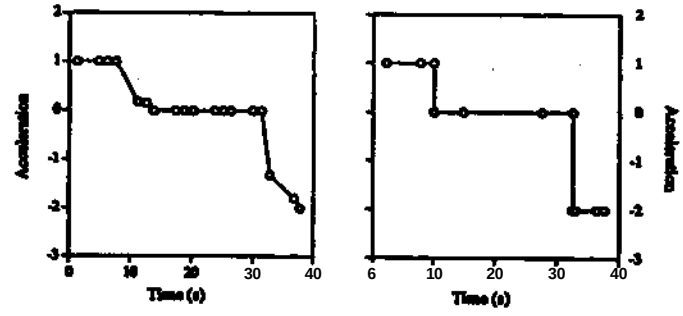

Figure 10: Optimal acceleration profiles for 6-uniform (a) and 3-optimal (b) meshes for car problem.

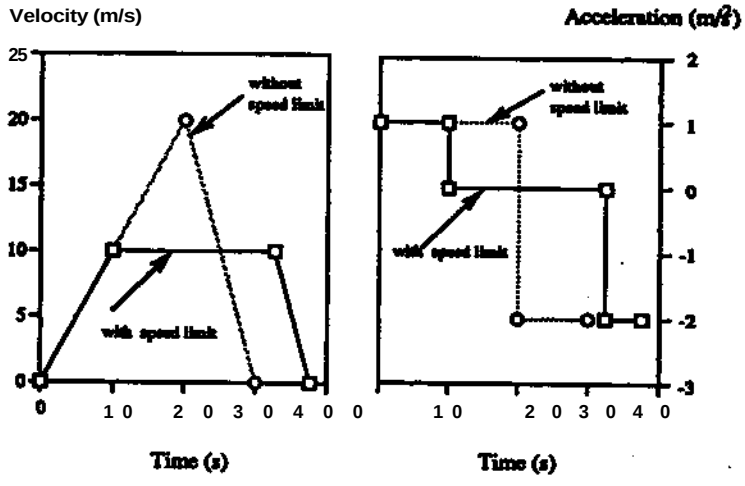

Figure 11: Optimal acceleration and velocity profiles.

Table 1: Computational statistics for batch PS polymerization problem.

Initiator cone. (gmol/1)	Number of SQP iterations		CDU time (s)		Objective (min) function
	feasible	infeasible	feasible	infeasible	
.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

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
8	8	39	36.5	2.8710	feasible path
8	8	20	15.7	2.8695	infeasible path

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 00	yi 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 (Vassiliadis, 1993)				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 (Vassiliadis, 1993)				yes	2.95539

Table 4: Computational statistics for Parameter estimation problem.

Formulation	Number of elements	Number of iterations	CPU time (s)	a	π
BVP	10	8	0.8	0.00	3.14159
BVP	10	8	0.9	0.05	3.14140
BVP	10	9	1.0	0.10	3.14121
PBVP	10	7	1.1	0.00	3.14175
PBVP	10	8	1.3	0.05	3.14178
PBVP	10	8	1.3	0.10	3.14802
IVP	10	7	2.0	0.00	3.14757
IVP	10	7	1.8	0.05	3.14178
IVP	10	7	1.9	0.10	3.14180

Table 5: Computational statistics for Parameter estimation problem (cont.).

Formulation	Number of elements	Number of iterations	CPU time 00	a	K
BVP	80	9	3.0	0.00	3.14159
BVP	80	9	3.3	0.05	3.14160
BVP	80	9	3.2	0.10	3.14220
PBVP	80	8	2.2	0.00	3.14159
PBVP	80	8	2.2	0.05	3.14159
PBVP	80	8	2.6	0.10	3.14180
IVP	30	-	-	-	-

Table 6: Computational statistics for Zeolite distribution example.

Cases	Number of collocation pt.	Number of elements	Number of iterations	CPU time 00	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

Table 7: Computational statistics for Car problem

Number of collocation	Number of finite elements	Number of SQP iterations	CPU time (•)	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

Table 8: Computational statistics for Car problem using nested strategy with feasible path

Number of collocation	Number of finite elements	Number of iter. (outer)	CPU time (•)	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

Table 9: Parameter values in the Zeolite distribution example.

Parameters	Units (parameter*unit=value)	Base Cases	case numbers		
			1	2	3
k_{az}	*l(ficm^3/mol/s	10	8.2	6.4	4.6
k_{am}	*l(Pcm^3/mol/s	1	1	1	1
hz	•IO^{2}*-^{1}	8	6.6	5.2	3.8
k_{bm}	*i0^2«-*	1	1	1	1
kCZ	*10^{2}cm?/mol/s	3	3	3	3
k_{cm}	*10*cm^3/mol/s	8	7	6	5
DA,	*10"^{5}cm^2/«	1	1	1	1
D_Am	*10-^{s}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

List of Figures