Large-Scale DAE Optimization Using a Simultaneous NLP Formulation

A. Cervantes and L. T. Biegler
Dept. of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15232

The differential-algebraic equation (DAE) optimization problem is transformed to a nonlinear programming problem by applying collocation on finite elements. The resulting problem is solved using a reduced space successive quadratic programming (rSQP) algorithm. Here, the variable space is partitioned into range and null spaces. Partitioning by choosing a pivot sequence for an LU factorization with partial pivoting allows us to detect unstable modes in the DAE system, which can now be stabilized without imposing new boundary conditions. As a result, the range space is decomposed in a single step by exploiting the overall sparsity of the collocation matrix, which performs better than the two-step condensation method used in standard collocation solvers. To deal with ill-conditioned constraints, we also extend the rSQP algorithm to include dogleg steps for the range space step that solves the collocation equations. The performance of this algorithm was tested on two well known unstable problems and on three chemical engineering examples including two reactive distillation columns and a plug-flow reactor with free radicals. One of these is a batch column where an equilibrium reaction takes place. The second reactive distillation problem is the startup of a continuous column with competitive reactions. These optimization problems, which include more than 150 DAEs, are solved in less than 7 CPU minutes on workstation class computers.

Introduction

We consider the general differential-algebraic equation (DAE) optimization problem

$$\min_{z(t),y(t),u(t),t_f,p} \varphi\left[z(t_f),y(t_f),u(t_f),t_f,p\right]$$

s.t. DAE model

$$F\left[\frac{dz(t)}{dt},z(t),y(t),u(t),t,p\right]=0 \tag{1}$$

$$G[z(t),y(t),u(t),t,p]=0 \tag{2}$$

initial conditions

$$z(0)=z^0 \tag{3}$$

point conditions

$$H_s[z(t_s),y(t_s),u(t_s),t_s,p]=0 \tag{4}$$

bounds

$$\begin{aligned} z^L \le z(t) \le z^U \\ y^L \le y(t) \le y^U \\ u^L \le u(t) \le u^U \\ p^L \le p \le p^U \\ t_f^L \le t_f \le t_f^U \end{aligned} \tag{5}$$

where φ is defined as scalar objective function, F differential equation constraints, G the algebraic equation constraints (we assume these are formulated to index one), H_s the additional

point conditions at times t_s, z the differential state profile vectors, z^0 the initial values of z, y the algebraic state profile vectors, u the control profile vectors, and p the time-independent parameter vector.

This problem can be solved by applying a nonlinear programming (NLP) solver to the DAE model. The methods which use this approach fall into two groups, sequential and simultaneous strategies. In the sequential strategy, an efficient nonlinear programming strategy is coupled to an existing DAE solver. Sensitivities are obtained from this solver and a small optimization problem is solved in the space of the control variables and optimization parameters (see Vassiliadis, 1993, for a review of these methods). In contrast, the simultaneous approach couples the solution of the DAE system with the optimization problem, and, therefore, requires a large-scale optimization procedure.

Despite the need for large-scale algorithms, the simultaneous approach has advantages for problems with state variable (or path) constraints and for systems where instabilities occur for a range of inputs. In addition, the simultaneous approach solves the DAE system only once, at the optimum point, and, therefore, can avoid intermediate solutions that may not exist, are difficult to obtain, or require excessive computation effort.

For solving simultaneous dynamic optimization problems, successive quadratic programming (SQP) is often the method of choice. Here, large-scale SQP methods for dynamic optimization problems can be classified as *full-space* or *reduced space*, and both have advantages for certain classes of problems. Full space methods are especially well suited for dynamic problems with many degrees of freedom (see, for example, Betts and Huffman, 1992; Betts and Frank, 1994). These methods take advantage of problem structure and the overall sparsity of the problem. For this reason, analytic second derivatives of the objective and constraint functions are usually required and special precautions are necessary to ensure descent properties for this approach.

On the other hand, the reduced space approach (rSQP) is easier to implement, but becomes inefficient when many degrees of freedom are present. With this approach, projected Hessian matrices (or their quasi-Newton approximations) are used and an efficient algorithm is constructed by decoupling the algorithm into range and null spaces. This approach allows us to incorporate Newton-based procedures instead of direct equation based process models.

Tanartkit and Biegler (1995, 1996, 1997) have adapted a simultaneous reduced Hessian successive quadratic programming (rSQP) algorithm to an implementation that incorporates the DAE solver COLDAE (Ascher et al., 1994). In this approach, COLDAE discretizes the state variables and linearizes the DAEs at given points in the time horizon, using orthogonal collocation on finite elements. The resulting matrix of the linearized system is decomposed by COLDAE with a two-step condensation method, obtaining a Newton step for the state variables. Here, unstable modes can be detected and appropriate boundary conditions are imposed in order to perform a stable decomposition. Moreover, for the previous and current studies we assume that the system (1-2) is index 1 without loss of generality, because there are several ways to reformulate high index DAE systems to index 1. Nevertheless, the saturation of state variable bounds in Eq. 5 can still lead to high index systems for (DAE1). To handle these, we rely on the stability properties of implicit Runge Kutta methods for higher order DAEs (see Brenan et al., 1996).

The search step for the discretized control variables and parameters is then obtained by solving a quadratic programming (QP) problem. However, this implementation is limited by the inability to handle large-scale problems, and this deficiency is a result of the two-step condensation method in COLDAE. Here the first step is performed by COLDAE using LINPACK subroutines DGEFA and DGESL (see Dongarra, 1979), which are dense linear system solvers. The system solved during the second step is in almost block diagonal (ABD) form, and a modification of the sparse SOLVEBLOCK routine by de Boor and Weiss (1980) is used. The main disadvantage of the COLDAE approach is that it does not exploit the sparsity of the DAEs. For this reason, the computational effort increases almost cubically with the number of DAEs.

The objective of this study is to implement a new strategy for the decomposition of the linearized system of DAEs and demonstrate its effectiveness on challenging process optimization problems. We describe this decomposition and apply COLDAE only to discretize and linearize the DAE system. The proposed decomposition is performed in a new rSQP subroutine with an efficient sparse matrix solver and a new interface. We show how the unstable modes are detected and eliminated automatically using the variables of the original problem and by exploiting some aspects of the rSQP algorithm. This allows us to perform the decomposition without the necessity of imposing additional boundary conditions. A demonstration of this approach is presented on several examples. The resulting optimization approach leads to much faster performance for process models with several hundred DAEs.

DAE Optimization

The continuous time problem is converted into an NLP by approximating the profiles to a family of polynomials on finite elements. In this work, we use a monomial basis representation (Bader and Ascher, 1987) for the differential profiles. This representation is recommended because of smaller condition number and smaller rounding errors. For a first-order ordinary differential equation, this monomial representation is

$$z(t) = z_{i-1} + h_i \sum_{q=1}^{\text{ncol}} \Omega_q\left(\frac{t - t_{i-1}}{h_i}\right)\frac{dz}{dt_{i,q}} \tag{6}$$

where z_{i-1} is the value of the differential variable at the beginning of element i, h_i is the length of element i, $dz/dt_{i,q}$ is the value of the first derivative coefficients in element i at the collocation point q, and Ω_q is a polynomial of order *ncol*, satisfying

$$\Omega_q(0) = 0 \quad \text{for } q = 1, \ldots, \text{ncol}$$

$$\frac{d}{dt}\Omega_q(\rho_r) = \delta_{q,r} \quad \text{for } q = 1, \ldots, \text{ncol}$$

where ρ_r is the rth collocation point within each element. In this work we use Radau points as they allow us to set constraints easily at the end of each element. The control and algebraic profiles are approximated using the same monomial basis representation which for a zero-order DAE takes the form

$$y(t) = \sum_{q=1}^{\text{ncol}} \psi_q\left(\frac{t - t_{i-1}}{h_i}\right) y_{i,q} \tag{7}$$

$$u(t) = \sum_{q=1}^{\text{ncol}} \psi_q\left(\frac{t - t_{i-1}}{h_i}\right) u_{i,q} \tag{8}$$

where $y_{i,q}$ and $u_{i,q}$ represent the values of the algebraic and control variables, respectively, in element i at collocation point q. Here, ψ_q is a Lagrange polynomial of order ncol satisfying

$$\psi_q(\rho_r) = \delta_{q,r} \tag{9}$$

The differential variables are required to be continuous throughout the time horizon, while the control and algebraic variables are allowed to have discontinuities at the boundaries of the elements. In this work we assume that the number of finite elements (ne) and their lengths are predetermined, although the adaptive strategy described in Tanartkit and Biegler (1997) can also be applied directly. With this assumption, substitution of Eqs. 6 to 8 into DAE1 leads to a nonlinear programming problem described below.

Reduced-Hessian successive quadratic programming (rSQP)

It has been shown (Tanartkit and Biegler, 1995) that rSQP is an efficient method for solving DAE optimization problems, especially when the dimension of the state variables is much larger than that of the control variables (that is, few degrees of freedom). rSQP also adds robustness to the solution procedure by performing local factorizations. This allows us to detect ill-conditioning due to unstable modes and to preserve and exploit the structure of the problem. The overall structure of our algorithm is shown in Figure 1. In the remainder of this section we include the description of the main components of our algorithm. Most of them are included in the rSQP subroutine OPT developed by Ternet and Biegler (1997). The nonlinear programming problem can be written as

$$\min_{x \in \Re^n} f(x) \tag{10}$$

$$\text{s.t.} \quad c(x) = 0 \tag{11}$$

$$x^L \le x \le x^U \tag{12}$$

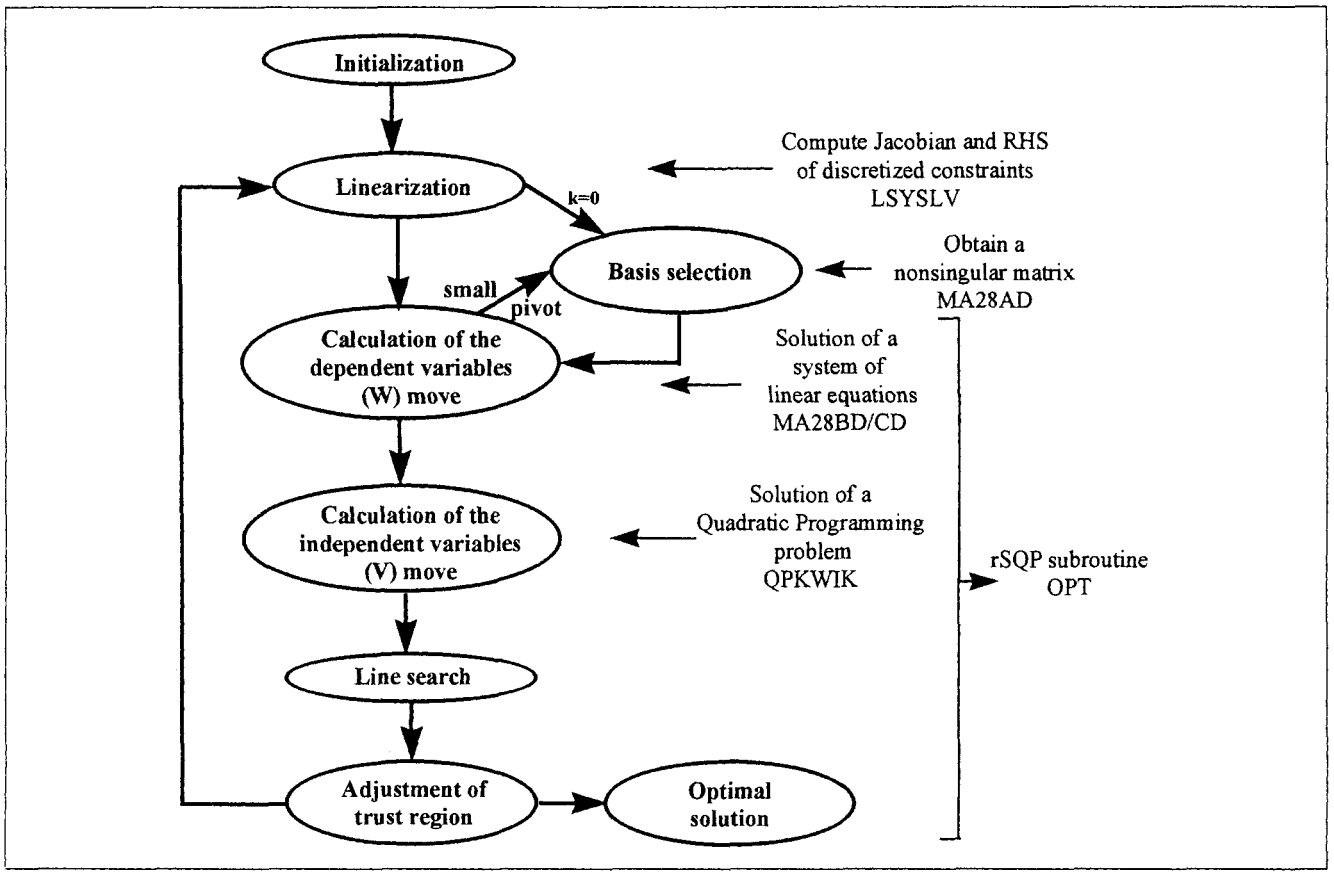

Figure 1. Structure of the algorithm.

where

$$x = \left(\frac{dz}{dt_{i,q}}, z_{i-1}, y_{i,q}, u_{i,q}, t, p \right)^T \text{ (variable vector)},$$

$$f: \Re^n \to R \quad \text{and} \quad c: \Re^n \to \Re^m$$

At each iteration k, a search direction d_k is obtained by solving a quadratic programming subproblem created from Eqs. 10–12

$$\min_{d \in \Re^n} \; g(x_k)^T d_k + \frac{1}{2} d_k^T B(x_k) d_k \tag{13}$$

$$\text{s.t. } c_k + A_k d_k = 0 \tag{14}$$

$$x^L \le x_k + d_k \le x^U \tag{15}$$

where g is the gradient of f, B denotes the Hessian of the Lagrangian function or a quasi-Newton approximation, $A_k = A(x_k)$ is the $m \times n$ Jacobian of the constraints at iteration k, and $c_k = c(x_k)$. The variables are partitioned into m dependent (W space) and n-m independent (V space) variables (basis selection in Figure 1). The independent variable space defines the null space of A, while the dependent variable space occupies the range space. Here the control variables and parameters are not necessarily the independent variables; this will depend on the basis selection that we will discuss later. With this partition, A takes the form

$$A(x) = [C(x) N(x)] \tag{16}$$

where the $m \times m$ basis matrix $C(x)$ is nonsingular. (If $C(x)$ is singular, dependent rows could be treated using the algorithm presented in Schmid and Biegler (1993).) With this partition, the search direction is written as

$$d_k = W_k p_W + V_k p_V \tag{17}$$

where the matrix V satisfies

$$A_k V_k = 0 \tag{18}$$

and we choose

$$V_k = \begin{bmatrix} -C(x_k)^{-1} N(x_k) \\ I \end{bmatrix} \quad \text{and} \quad W_k = \begin{bmatrix} I \\ 0 \end{bmatrix} \tag{19}$$

The range space direction p_W can now be determined as

$$p_W = -[A_k W_k]^{-1} c_k \tag{20}$$

and the null space direction p_V is obtained from the following reduced QP subproblem

$$\min_{p_V \in \Re^{n-m}} \left(V_k^T g_k + V_k^T B_k W_k p_W \right)^T p_V + \frac{1}{2} p_V^T \left(V_k^T B_k V_k \right) p_V \tag{21}$$

$$\text{s.t. } x^L - x_k - W_k p_W \le V_k p_v \le x^U - x_k - W_k p_W \tag{22}$$

The reduced QP subproblem is solved using the QPKWIK algorithm by Ternet and Biegler (1997), which is an improved version of the original one by Schmid and Biegler (1993). This algorithm uses a dual-space solution procedure, which is particularly efficient for problems with many inequality constraints, most of them being inactive. Another important characteristic of this subroutine is that the computational effort often increases quadratically with respect to the number of degrees of freedom.

The new version incorporates a trust region, which restricts the stepsize of d_k to an area around x_k in which we can trust the Taylor series approximations to the Lagrangian and the constraints. The addition of a trust region can improve the convergence of the algorithm, especially when the 2nd order information for the problem is poor. This implementation adds a trust region constraint $\|d_k\| \le \Delta_k$ to Eqs. 20–22, moving the search direction from a Newton step direction to a steepest descent direction. The direction of the Wp_w step is not affected by the trust region. Only its length changes, as this step is generated before the solution of QP.

Linearization and the range space move

In order to obtain the Jacobian A of the constraints, the discretized DAE system should be linearized. As mentioned before, the Jacobian A is an $m \times n$ matrix, these dimensions can be expressed in terms of the dimensions of the original DAE system

$$n = (ne + 1) \times nz + \text{ncol} \times (nz + ny + nc) \times ne + np \tag{23}$$

$$m = (ne + 1) \times nz + \text{ncol} \times (nz + ny) \times ne + na \tag{24}$$

where nz, ny and nc represent the number of differential, algebraic and control variables, np the number of parameters, and na the number of additional point equality constraints.

This matrix takes the form shown in Figure 2, which is almost block diagonal. Here, I represents the identity matrix, and D_i is a matrix containing the coefficients of the continuity equations in each element i. Z_i^q, DZ_i^q, Y_i^q, U_i^q and P_i^q represent the Jacobian of the collocation equations with respect to z_i, $dz/dt_{i,q}$, $y_{i,q}$, $u_{i,q}$ and p at collocation point q and element i. The linearization of the collocation equations is performed using a modified version of the subroutine LSYSLV in COLDAE (Ascher and Spiteri, 1994), a well tested algorithm for collocation on finite elements. The modifications in LSYSLV allow us to treat the control variables as algebraic variables and to handle directly the nonsquare Jacobian of the original DAE system. At the same time, the linearized system is stored in sparse form. This subroutine also obtains the righthand sides $c(x)$ of all the constraints.

Before obtaining the range space move, the matrix A should be partitioned into C and N, where C has to be a nonsingular and well-conditioned square matrix. Here we assume that if C is well-conditioned it should yield a dichotomous and stable system. This task is performed with the Harwell subroutine MA28 and will be discussed in detail in the next section.

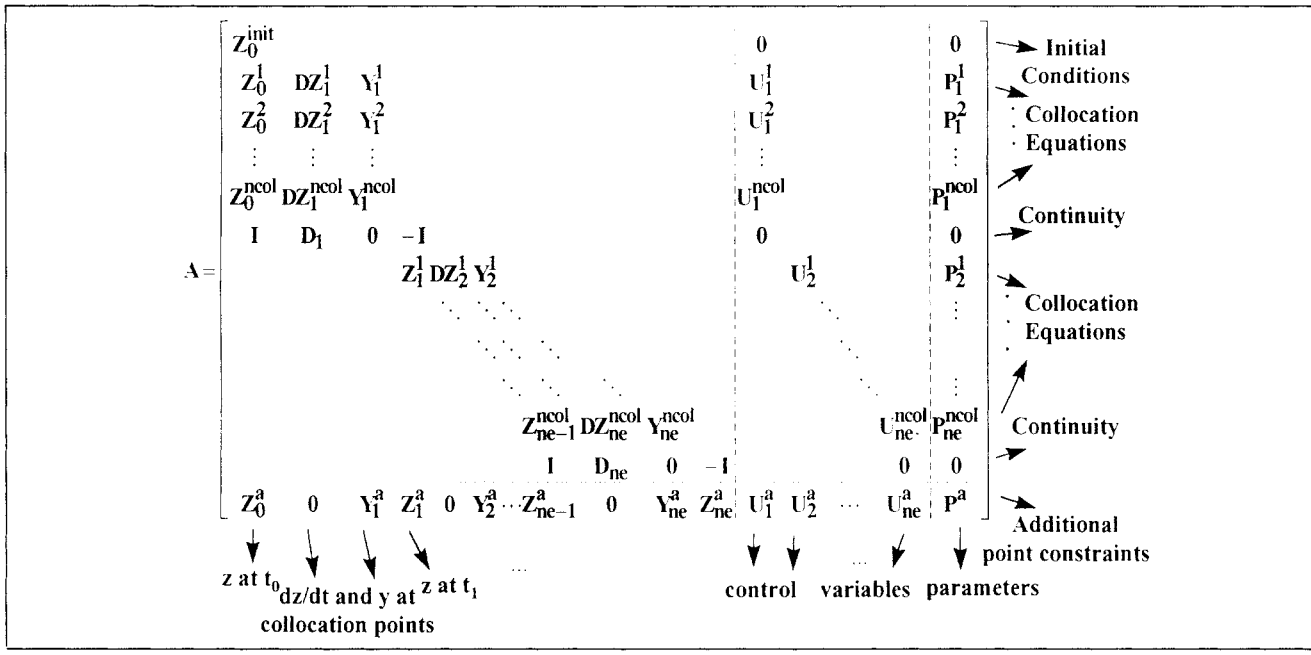

Figure 2. Discretized and linearized constraints.

We then perform a sparse LU factorization of C in one step. Here, we do not exploit directly the almost block diagonal structure of the linearized system but we are exploiting the sparsity of the matrices Z, DZ, Y, U and P, which have the same degree of sparsity as the original DAE system. This decomposition performs better for sparse DAE systems (as we will show in the examples) than the decomposition in two steps perform in COLDAE. The two-step decomposition exploits directly the block diagonal form of the linearized system but not the sparsity of the DAE. Here, the LU factorization is performed with the Harwell subroutine MA28, a well tested sparse linear solver, and is shown to perform, at most, quadratically with respect to the number of equations in sparse systems. Similarly, a number of more recent sparse codes could be employed very effectively (such as Davis and Duff, 1997 (UMFPACK) or Duff and Reid, 1996 (MA48)).

Basis Selection

We consider the partition of A into columns

$$A = \begin{bmatrix} A_z & A_{z'} & A_y & A_u & A_p \end{bmatrix} \tag{25}$$

where A_z, $A_{z'}$, A_y, A_u and A_p represent the columns corresponding to all z_i, $dz/dt_{i,q}$, $y_{i,q}$, $u_{i,q}$ and p respectively, at collocation point q and element i. For this system to have a solution, the following partition must exist

$$\begin{bmatrix} A_z & A_{z'} & A_y & A_u & A_p \end{bmatrix} = \begin{bmatrix} C & N \end{bmatrix} \tag{26}$$

with C square, nonsingular, and well-conditioned. A straightforward way to perform this partition is to define C and N as

$$C = \begin{bmatrix} A_z & A_{z'} & A_y \end{bmatrix} = C^0 \quad \text{and} \quad N = \begin{bmatrix} A_u & A_p \end{bmatrix} = N^0 \tag{27}$$

This means we set all the discretized control variables and parameters as independent variables. However, this partition of A works well only if the DAE system does not have unstable modes. In case the DAE is unstable, this partition will give us an ill conditioned C matrix, leading to numerical problems in the Wp_w step. Tanartkit and Biegler (1996) worked with this partition of A, but they showed that by imposing dummy final conditions, C^0 can be transformed into a well-conditioned matrix. In their approach a new dummy variable is added for each unstable mode in the DAE. In order to solve this problem without imposing these final conditions, we exchange some columns between C^0 and N^0. Before getting into the details of this operation, we first discuss the importance of C being well-conditioned.

Dichotomy and well conditioning

de Hoog and Mattheij (1987, 1989) investigated the relationships between the conditioning of a boundary value problem (BVP) and the growth behavior of its fundamental solution. They have shown that if the problem is moderately well-conditioned, then the homogeneous solution space must be dichotomic, which means that it can be split into a subspace of increasing modes and a complementary subspace of nonincreasing modes. To illustrate this equivalence, consider the first-order linear system of differential equations as discussed by de Hoog and Mattheij (1987)

$$\Gamma z \equiv z' - \bar{A}z = \hat{f}, \quad 0 < t < 1 \tag{28}$$

subject to the following boundary conditions (BC)

$$B'z \equiv B_0 z(0) + B_1 z(1) = b \tag{29}$$

This BVP has a unique solution if and only if $B'\bar{Y}$ is nonsin-

gular, where $\bar{Y}$ is any fundamental solution of Γ. The solution to this problem is

$$z(t) = \bar{Y}(t)\left[B_0\bar{Y}(0) + B_1\bar{Y}(1)\right]^{-1}b + \int_0^1 \bar{G}(t,s)\hat{f}(s)ds \tag{30}$$

where $\bar{G}$ is the Green's function of the problem.

A problem is strongly dichotomic if there exists a constant κ and a projection $\bar{P}$ matrix such that for a fixed fundamental solution $\bar{Y}$

$$|\bar{Y}(t)\bar{P}\bar{Y}^{-1}(s)| \le \kappa, \quad t \ge s, \tag{31}$$

$$|\bar{Y}(t)(I-\bar{P})\bar{Y}^{-1}(s) \le \kappa, \quad t < s. \tag{32}$$

For a BVP with separable boundary conditions, κ is defined by

$$\sup_{t,s}|\bar{G}(t,s)|$$

and the projection matrix provides the dichotomic subspaces.

It can be shown that if a fundamental solution satisfies (Eqs. 31 and 32), then there exist matrices $\tilde{B}_0$ and $\tilde{B}_1$ so that

$$\|z\|_\infty \le \tilde{\beta}|\tilde{B}z| + \tilde{\alpha}\|Gz\|_1 \tag{33}$$

with

$$\tilde{\beta} \le \sqrt{2}\,\kappa, \quad \tilde{\alpha} \le \kappa + 4\kappa^2 \tag{34}$$

$$\tilde{B}z = \tilde{B}_0 z(0) + \tilde{B}_1 z(1) \tag{35}$$

where the boundary conditions are scaled such that

$$\tilde{B}_0\tilde{B}_0^T + \tilde{B}_1\tilde{B}_1^T = I \tag{36}$$

These conditions for boundedness and stability of the BVP are also achieved when the BVP is discretized and solved as a linear system with a well-conditioned C matrix. Consequently, we can make the equivalence of well conditioning and dichotomy from both linear algebra and functional analysis perspectives.

Detection of ill-conditioning

If an unstable mode is present in the DAE, A is required to be partitioned so that the end conditions of any increasing mode are fixed or become decision variables. Here, if a differential variable z^j has an increasing mode, either z^j_{ne} or $dz^j/dt_{ne,\mathrm{ncol}}$ could be specified and this would correspond to a column of N. Correspondingly, a column corresponding to a control variable or a parameter would be added to C. Here we see that specifying the variables that span the columns of N is equivalent to solving the corresponding discretized, linear BVP.

To partition into the C and N matrices, the detection for good conditioning is performed by (the necessary condition of) selecting a pivot sequence for the LU factorization of the A matrix (see Duff et al., 1992). As we are solving a DAE system using an LU factorization, it is important to reconsider the error analysis in this algorithm. LU factorization with partial pivoting is equivalent to solving the system $(\tilde{A}+E)\hat{x} = b$ exactly, where

$$\|E\|_\infty \le 8n^3\rho\|\tilde{A}\|_\infty\nu + O(\nu^2) \tag{37}$$

$$\frac{\|\tilde{x}-\hat{x}\|_\infty}{\|\tilde{x}\|_\infty} \le 32n^3\rho\nu\kappa_\infty(\tilde{A}) + O(\nu^2), \tag{38}$$

ν is the machine precision $\nu = 10^{-\delta}$ and the growth parameter ρ is defined as

$$\rho = \max_{i,j,k}\frac{|\tilde{a}_{ij}^{(k)}|}{\|\tilde{A}\|_\infty} \tag{39}$$

Two issues of concern are the values of $\kappa_\infty(\tilde{A})$ and ρ. The first constant can be bounded by scaling rows and columns, although this was not necessary in this work. Also, in practice, ρ is usually of order 10 but can be as large as 2^{n-1}. In our work, an excessive growth parameter was not encountered, but it can be handled by applying complete pivoting or using iterative refinement.

The objective of the pivot selection is to minimize the fill-in during the factorization, but also to achieve numerical stability. Here, we use the Harwell subroutine MA28 for the selection of the sequence. This subroutine uses Markowitz's ordering strategy with threshold pivoting for numerical stability. That is, it will choose a_{ij} as pivot if it minimizes the quantity

$$(r_i - 1)(c_j - 1) \tag{40}$$

over all entries of the reduced matrix that satisfy the inequality

$$|a_{ij}^{(k)}| \ge \hat{u}\max_{l \ge k}|a_{il}^{(k)}| \tag{41}$$

where r_i is the number of entries in row i of the reduced matrix, c_j is the number of entries in column j and $\hat{u}$ is a preset and fixed parameter in the range $0 < \hat{u} \le 1$.

Equation 41 therefore allows us to select columns from A_u and A_p to replace the unstable modes of the DAE, by selecting a pivot sequence with numerical stability. Moreover, modification of this strategy also allows us to restrict the variables that can be selected as independent. For example, the index one algebraic variables y can always be selected as dependent variables as they are already associated with the modes of the differential variables. This restriction is added by simply multiplying the desired columns by a sufficient large number. In this way, MA28 will automatically set a pivot in those columns (of A_y) and choose these variables as dependent. In the next section, we present some examples in which this kind of scaling could be useful.

Choosing good pivot sequences is a necessary condition for obtaining a well-conditioned C matrix. Nevertheless, we also need to consider the possibility of C still being ill-conditioned after applying the pivot selection technique. This can also happen as a result of an improper initial guess, for ex-

ample. To address this problem, we have implemented an optional dogleg step (Powell, 1970) for the range space move. With this implementation, the p_w step from Eq. 20 is transformed to a hybrid step, which is the combination of a Newton direction and a steepest descent direction. Using the definition of W, this step takes the form

$$p_w = (\eta)p_w^N + (1-\eta)\beta p_w^{sd} \tag{42}$$

where

$$p_w^N = -C^{-1}c_k \tag{43}$$

$$p_w^{sd} = -C^T c_k \tag{44}$$

$$\beta = \frac{\|p_w^{sd}\|^2}{\|Cp_w^{sd}\|^2} \tag{45}$$

and η is defined such that $x + p_w$ stays inside a given trust region (see also Dennis and Schnabel, 1996). With these equations, we can guarantee the uniqueness of the W space move, even if C is ill-conditioned. It is clear that this technique can also be applied during the initialization of the problem. If a feasible solution is required to initialize the problem, a hybrid algorithm instead of a Newton algorithm can be implemented to converge the discretized system of equations for given values of the control variables and parameters. If a hybrid step is taken during the optimization, the penalty parameter (μ) of the line-search function must be modified in order to apply the multiplier free version (Schmid and Biegler, 1993) of the rSQP algorithm. Here, it can be shown that a descent direction for

$$f(x) + \mu\|c(x_k)\|_2 \tag{46}$$

is obtained if

$$\mu = \frac{|(g+\nu')^T W p_w|\|c_k\|_2 + 3\rho'\|c_k\|_2^2}{\eta\|c_k\|_2^2 + \beta(1-\eta)\|p_w^{sd}\|_2^2} \tag{47}$$

for some $\rho' > 0$ and where ν' are the multipliers calculated from the bound constraints in the QP. Note that the l_2 norm has to be used instead of the l_1 norm used in Schmid and Biegler (1993).

Examples

We first demonstrate our approach on two small parameter estimation problems and on three chemical engineering examples that include continuous and batch reactive distillation columns, as well as a plug-flow reactor optimization. The CPU times for all examples were obtained on an HP 777/110. The last three problems were initialized with a feasible solution, and the CPU time includes this computation.

Parameter estimation problems

This is a parameter estimation problem discussed in Tanartkit and Biegler (1995), which was originally taken from Bock (1983). This problem is formulated as an initial value problem with two state variables and one parameter

$$\min \sum_{i=1}^{n} \left(z_{2,m}(i) - z_2(i)\right)^2 \tag{48}$$

$$\text{s.t.} \quad \frac{dz_2}{dt} = \tau^2 z_1 - (\tau^2 + p^2)\sin(pt) \tag{49}$$

$$\frac{dz_1}{dt} = z_2 \tag{50}$$

$$z_1(0) = 0; \qquad z_2(0) = \pi \tag{51}$$

The solution to the ODE system is given by

$$z_1(t) = \sin(pt) + \frac{\pi - p}{2\tau}(\exp(\tau t) - \exp(-\tau t)) \tag{52}$$

$$z_2(t) = p\cos(pt) + \frac{\pi - p}{2}(\exp(\tau t) + \exp(-\tau t)) \tag{53}$$

and inspection of Eqs. 52 and 53 shows that the error will propagate exponentially as a function of t, τ whenever $p \neq \pi$. As a result, the system is unstable and ill-conditioned.

We solve this problem as stated, without imposing final boundary conditions. The state profiles are initialized to 0.1, and the unknown parameter p is set to 2. With τ set to 100, we estimate the parameter given true values for z_2 corrupted with random noise (σ) at data points equal to the mesh points.

This problem has only one degree of freedom. If we select the parameter p as the independent variable, the matrix for obtaining the Wp_w step becomes ill conditioned, resulting in a very large step and leading to failure of the algorithm. Instead, if we select the basis with MA28 we obtain a well-conditioned matrix by choosing one of the state variables at the last element as the independent variable. As a result, the problem has the stability properties of a BVP. The results are reported in Table 1 for different numbers of elements and

Table 1. Results for Parameter Estimation Problem: 1 Parameter

Basis Selection	No. of Elements	Discr. Vars.	$\sigma = 0.01$		$\sigma = 0.10$	
			Iter.	CPU (s)	Iter.	CPU (s)
MA28	2	19	11	0.1	11	0.1
π Independent	2	19	14	0.1	14	0.1
MA28	8	67	10	0.2	10	0.2
π Independent	8	67	35	0.7	36	0.7
MA28	16	131	5	0.3	5	0.3
π Independent	16	131	Failed Wp_w = 10E19		Failed Wp_w = 10E19	
MA28	30	243	5	0.4	5	0.5
π Independent	30	243	Failed Wp_w = 10E29		Failed Wp_w = 10E29	

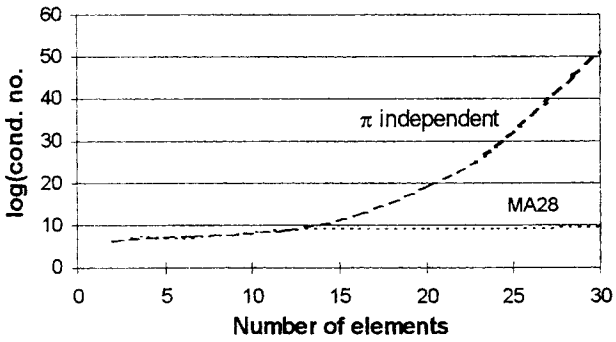

Figure 3. Condition number for parameter estimation problem.

values of the standard deviation σ. The same results were obtained by scaling the columns of the matrix before the basis selection with MA28.

We also studied the condition number of the C matrix, as we increase the number of elements, using the LINPACK subroutine DGECO. Setting p as the independent variable, the condition number of the matrix grows exponentially. If we select the basis with MA28, the condition number grows only linearly, as shown in Figure 3.

Consider the following parameter estimation problem, modified from Wright (1992) and also studied by Tanartkit and Biegler (1996)

$$\min \sum_{i=1}^{n} \left(z_{1,m}(i) - z_1(i)\right)^2 + \sum_{i=1}^{n} \left(z_{2,m}(i) - z_2(i)\right)^2 \quad (54)$$

$$\text{s.t.} \begin{bmatrix} \frac{dz_1}{dt} & \frac{dz_1}{dt} & \frac{dz_1}{dt} & \frac{dz_1}{dt} & \frac{dz_1}{dt} \end{bmatrix}^T = A\begin{bmatrix} z_1 & z_2 & z_3 & z_4 & z_5 \end{bmatrix}^T + f(t) \quad (55)$$

$$\begin{bmatrix} z_1(0) & z_2(0) & z_3(0) & z_4(0) & z_5(0) \end{bmatrix}^T = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T \quad (56)$$

where

$$A = \begin{bmatrix} -\Psi_1 \cos(2\omega_1 t) & 0 & \omega_1 + \Psi_1 \sin(2\omega_1 t) & 0 & 0 \\ 0 & -\Psi_2 \cos(2\omega_2 t) & 0 & \omega_2 + \Psi_2 \sin(2\omega_2 t) & 0 \\ -\omega_1 + \Psi_1 \sin(2\omega_1 t) & 0 & \Psi_1 \cos(2\omega_1 t) & 0 & 0 \\ 0 & -\omega_2 + \Psi_2 \sin(2\omega_2 t) & 0 & \Psi_2 \cos(2\omega_2 t) & 0 \\ 0 & 0 & 0 & 0 & \Psi_3 \end{bmatrix} \quad (57)$$

and $f(t)$ is chosen such that the solution of the ODE is

$$\begin{bmatrix} z_1(t) & z_2(t) & z_3(t) & z_4(t) & z_5(t) \end{bmatrix}^T = \begin{bmatrix} e^t & e^t & e^t & e^t & e^t \end{bmatrix}^T \quad (58)$$

The objective is to estimate the parameters Ψ_1, Ψ_2 and Ψ_3 ($\Psi_i^* = (10)^i$) given 30 measured data sets corrupted with random noise (σ) for z_1 and z_2. The parameters are initialized to 12, 120, and 1,200, respectively. This system is ill-conditioned because of 3 unstable modes.

Table 2. Results for Parameter Estimation Problem: 3 Parameters

No. of ODEs	σ	Discr. Eq./Vars.	Basis Selection	Iter.	CPU (s)
5	0.05	605/608	MA28	25	5.2
5	0.05	605/608	Parameters independent	Failed	$Wp_w \approx 10^{36}$
5	0.1	605/608	MA28	30	5.7
5	0.1	605/608	Parameters independent	Failed	$Wp_w \approx 10^{36}$

This problem cannot be solved by setting the parameters as independent variables, but if we select the basis with MA28, with or without prior scaling, we are able to solve the problem formulated as an IVP. We used 30 elements and 3 collocation points. The results are reported in Table 2.

Reactive distillation optimization

Reactive distillation is an attractive alternative to conventional processing schemes. When reactions are equilibrium limited, reactive distillation continuously removes products from the reaction zone, which dramatically increases the overall conversion. The technique may also increase selectivity in certain competing reaction systems by continuously separating products from reactants. Lastly, reactive distillation may prove economically attractive by reducing the number of processing units and providing direct heat integration between reaction and separation.

Most of the research in reactive distillation has focused upon computing a single steady-state solution to the material balances, but little work has been done with dynamic models, especially in the dynamic optimization area. In this work, we formulate and solve dynamic index one optimization problems for both batch and continuous reactive distillation columns.

Batch Reactive Distillation. In the chemical industry, interest in batch process optimization has increased in the last years as low-volume, high-value chemicals are considered. The main advantage of batch processes is their high flexibility. One important advantage of batch reactive distillation is that equilibrium reactions can be carried out better by removal of products. This characteristic is very attractive for the production of esters or for hydrolysis, isomerization, chlorination, and alkylation reactions. For these systems, it is also important to consider optimal operating schemes, which allows us to obtain the maximum conversion to the desired products. While some work has been done in dynamic simu-

lation of batch reactive distillation, little research has been conducted in the optimization of these operations. As an example for our approach, we first construct the index one DAE model and then consider a batch separation for ethyl acetate.

The dynamic model is developed from the following equations. First, the molar holdup M_i on tray i in the distillation column can be expressed as (see Ruiz et al., 1995)

$$\frac{dM_i}{dt} = F_i + V_{i+1} + L_{i-1} - V_i - L_i + \sum_{n=1}^{nr} R_{i,n} \qquad (59)$$

where V_i and L_i are the vapor and liquid flow rates, respectively, F_i is the feed flow rate, nr is the number of reactions, and $R_{i,n}$ is the difference between the rates of production and consumption of each reaction n. The liquid flow rates are obtained as a function of the molar holdups and the geometry of each tray. The expression for the liquid mole fraction x of each component j is

$$\begin{aligned} M_i\frac{dx_{i,j}}{dt} &= F_i(z_{i,j} - x_{i,j}) \\ &\quad + V_{i+1}(y_{i+1,j} - x_{i,j}) + L_{i-1}(x_{i-1,j} - x_{i,j}) \\ &\quad - x_{i,j}\sum R_{i,N} - V_i(y_{i,j} - x_{i,j}) + \sum_{n=1}^{nr} \nu_{j,n}\frac{M_i}{\rho_i} r_{i,n} \end{aligned} \qquad (60)$$

where ρ_i is the molar density of the liquid mixture, $z_{i,j}$ is the molar fraction of j in the feed, $y_{i,j}$ is the vapor mole fraction, $\nu_{j,n}$ is the stoichiometric coefficient, and $r_{i,n}$ is the rate of production per volume unit. Here we assume a constant ρ_i.

The equilibrium between the vapor and liquid phases is represented by

$$y_{i,j} = K_{i,j}x_{i,j}$$

where

$$K_{i,j} = f_j(T_i) \qquad (61)$$

and the sum of the vapor fractions on each tray is required to be equal to one

$$\sum_{j=1}^{nc} y_{i,j} = 1 \qquad (62)$$

The vapor rate is calculated at every tray from the energy balance

$$\begin{aligned} M_i\frac{dh_i^l}{dt} &= V_{i+1}(h_{i+1}^v - h_i^l) + L_{i-1}(h_{i-1}^v - h_i^l) \\ &\quad - h_i^l\sum R_{i,N} - V_i(h_i^v - h_i^l) + \sum_{n=1}^{nr} \frac{M_i}{\rho_i} r_{i,n}\Delta H_n^R \end{aligned} \qquad (63)$$

where $h^v(T_i, x_{i,j}, y_{i,j})$ and $h^l(T_{i,j}, x_{i,j})$ are the vapor and liquid enthalpies, and ΔH_n^R is the heat of reaction. However, if the problem is formulated with Eq. 63, an index two problem is obtained. This can be easily identified because the vapor flow rate, which is an algebraic variable, does not appear in the algebraic Eqs. 61–62. In order to reduce the index, we can write (see Gear, 1988)

$$\frac{dh_i^l}{dt} = \sum_{j=1}^{nc} \frac{\partial h_i^l}{\partial x_{i,j}}\frac{dx_{i,j}}{dt} + \frac{\partial h_i^l}{\partial T_i}\frac{dT_i}{dt} \qquad (64)$$

The derivative of the temperature can be obtained from Eqs. 61 and 62 as

$$\frac{d}{dt}\left(\sum_{j=1}^{nc} K_{i,j}x_{i,j}\right) = \sum_{j=1}^{nc} K_{i,j}\frac{dx_{i,j}}{dt} + \left(\sum_{j=1}^{nc} x_{i,j}\frac{dK_{i,j}}{dT_i}\right)\frac{dT_i}{dt} = 0 \qquad (65)$$

By combining Eqs. 64 and 65, the terms in Eq. 63 can be replaced by a more complex algebraic expression (Eq. 66), leading to an index one DAE system

$$\begin{aligned} &\frac{1}{M_i}\left(V_{i+1}(h_{i+1}^v - h_i^l) + L_{i-1}(h_{i-1}^v - h_i^l) - V_i(h_i^v - h_i^l)\right. \\ &\qquad \left. + \sum_{n=1}^{nr} \frac{M_i}{\rho_i} r_{i,n}\Delta H_n^R\right) = \text{RHS} \end{aligned} \qquad (66)$$

where

$$\text{RHS} = \sum_{j=1}^{nc} \frac{\partial h_i^l}{\partial x_{i,j}}\frac{dx_{i,j}}{dt} - \frac{\partial h_i^l}{\partial T_i}\frac{\sum_{j=1}^{nc} K_{i,j}\frac{dx_{i,j}}{dt}}{\sum_{j=1}^{nc} x_{i,j}\frac{dK_{i,j}}{dT_i}}$$

and $dx_{i,j}/dt$ is obtained from Eq. 60.

In this example we consider the reversible reaction between acetic acid and ethanol (Wajge and Reklaitis, 1995)

$$CH_3COOH + CH_3CH_2OH \leftrightarrow CH_3COOCH_2CH_3 + H_2O \qquad (67)$$

The model consists of $12+5nt$ differential and $6+5nt$ algebraic equations, where nt is the number of trays. The column, in all cases, was fed with an equimolar mixture of ethanol, acetic acid, ethyl acetate, and water.

First, we considered only the simulation of the column in order to compare the performance of the new decomposition strategy for obtaining the Wp_w or Newton step. The model simulates an hour of operation. During the first half hour, the column is operated under total reflux. After this, the reflux ratio is set to a value of 3. We use two collocation points and five elements. We perform the simulation for a variable number of trays. In all cases the nonlinear solver converged in 9 iterations. In Figure 4, we can see that the CPU time grows only linearly, for the decomposition using MA28, but grows almost cubically for COLDAE. This illustrates the advantage of the current approach over the previous application.

Next, we consider the optimal control problem. Here, we maximize the amount of distillate D produced within one

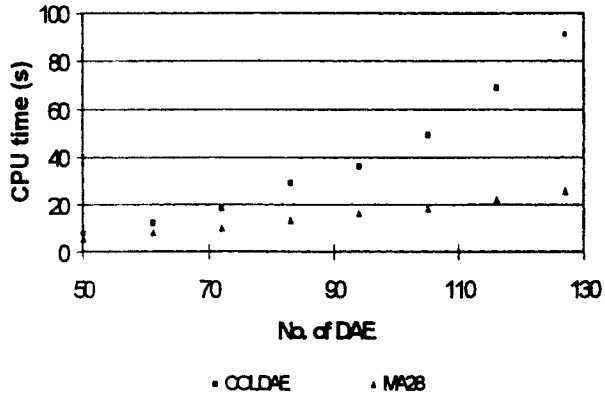

Figure 4. Newton (range space) step calculation as a function of number of DAES.

hour by manipulating the reflux ratio as a function of time, as follows

$$\max \int_0^{t_f=1} D dt \tag{68}$$

$$\text{s.t. DAEs (Eqs. 59–62), (Eq. 66)}$$

$$x_D^{\text{Ester}} \geq 0.4600$$

Here, the concentration of ethyl acetate in the final distillate should be at least 0.46. We solved this problem for 8 and 15 trays. In both cases the reflux ratio is at its upper bound at the beginning, and at its lower bound at the end, after obtaining the adequate composition of the distillate. Figure 5 shows this behavior while Figure 6 shows the mole fractions for the 8 tray column. For the optimization we used 5 elements and 3 collocation points.

In Table 3 we report the CPU times for the 8 and 15 tray columns. For these cases, if we select the discretized reflux ratio (control variables) as independent variables we obtain better performance than with MA28 itself. But, if we scale the columns of the system (MA28s) before selecting the basis with MA28, we obtain the same results. In all cases the problem was initialized with a feasible point with a constant reflux ratio of 20.

Continuous Reactive Distillation. In this system, water and ethylene oxide react to produce ethylene glycol. A parallel reaction takes place, producing diethylene glycol according to

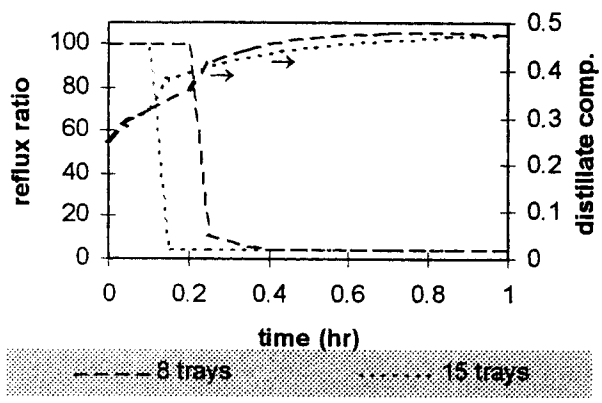

Figure 5. Reflux ratio for batch column.

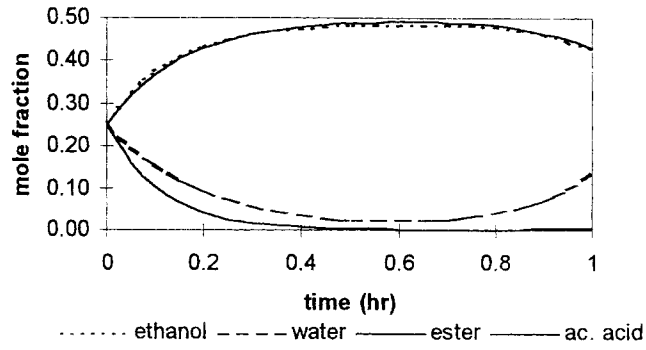

Figure 6. Condenser composition (8 trays).

$$\begin{aligned} C_2H_4O + H_2O &\rightarrow C_2H_6O_2 \\ C_2H_4O + C_2H_6O_2 &\rightarrow C_4H_{10}O_3 \end{aligned} \tag{69}$$

The column consists of ten trays, a total condenser and a reboiler. Water is fed at tray one and the ethylene glycol is fed from trays one to 6, as shown in Figure 7. The column design was taken from Ciric and Gu (1994).

The DAE model consists of 50 differential and 20 algebraic equations. The basic equations are the same as in the batch reactive column (Eqs. 59–62 and Eq. 66), except that now liquid density is a function of temperature. Both the reboiler and the condenser have negligible holdups, the column is originally full of water, and at time zero the ethylene oxide and the water are fed. The objective is to minimize the heat required to start up the column. The control variable is the fraction of the bottoms (α), which is recirculated to the reboiler. Also, the final time (t_f) is treated as a parameter. The final concentration of ethylene glycol at the bottoms is required to be 0.86 and the final control variable is required to be 0.958, which are the design values. The optimization problem can be stated as

$$\min_{t_f} \int_0^{t_f} Q_R dt \tag{70}$$

$$\text{s.t. DAEs (Eqs. 59–62), (Eq. 66)}$$

$$x_{t_f}^{\text{EG}} = 0.8600$$

$$\alpha_{t_f} = 0.9580$$

We solve the problem with 6 elements and 3 collocation points. The bottoms composition of ethylene glycol and wa-

Table 3. Results Reactive Batch Column

No. of Trays	No. of DAEs	Discr. Eq./Vars.	Basis Selection	Iter.	CPU (s)
8	98	1,788/1,798	MA28	29	105.8
8	98	1,788/1,798	Control vars. independent	19	88.4
8	98	1,788/1,798	MA28s	19	89.1
15	168	3,048/3,058	MA28	46	362.5
15	168	3,048/3,058	Control vars. independent	29	215.4
15	168	3,048/3,058	MA28s	29	216.2

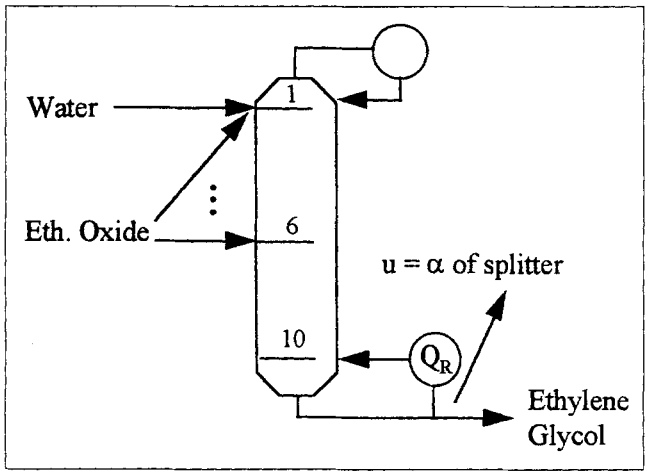

Figure 7. Continuous reactive distillation.

ter, as well as the profile for α, are shown in Figure 8. In Figure 9 we also present the temperature and holdup profiles for the last tray. The computational results are reported in Table 4. In this case the algorithm was slower when we select the basis using MA28 without scaling. However, the same results were obtained with scaling and choosing the controls as independent.

Plug-flow reactor optimization

Optimization of plug-flow reactors is especially important in the chemical industry. One example is the olefin production industry, where various hydrocarbon feedstocks react in a cracking furnace. The optimization of this kind of unit is of special interest in order to increase profitability and product quality. Some work has been done in this area using modular approaches, but only recently have equation oriented methods been applied. This example is a challenging problem because of the presence of free radicals in the reactive schemes. For this example, the reaction system includes six molecules, three free radicals, and seven reactions as shown below

$$\begin{aligned}
&C_2H_6O \rightarrow 2CH_3\bullet\\
&CH_3\bullet + C_2H_6 \rightarrow CH_4 + C_2H_5\bullet\\
&H\bullet + C_2H_6 \rightarrow H_2 + C_2H_5\bullet\\
&C_2H_5\bullet \rightarrow C_2H_4 + H\bullet\\
&C_2H_5\bullet + C_2H_4 \rightarrow C_3H_6 + CH_3\bullet\\
&2C_2H_5\bullet \rightarrow C_4H_{10}\\
&H\bullet + C_2H_4 \rightarrow C_2H_5\bullet
\end{aligned} \tag{71}$$

The model also includes the heat balance and the pressure drop equation (Chen et al., 1996). This gives a total of eleven differential equations as follows:

The molar flow F for each component j along the dimension x of the reactor is given by

$$\frac{dF_j}{dx} = \frac{\pi d_t^2}{4} R_j \tag{72}$$

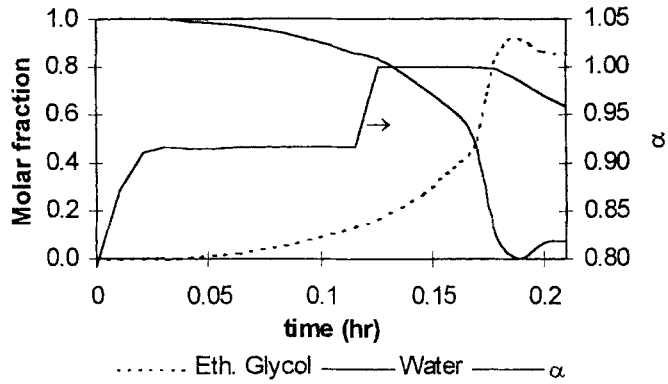

Figure 8. Bottoms composition and control profiles.

d_t is the tube diameter and R_j is defined as

$$R_j = \sum_i \nu_{ij} r_i \tag{73}$$

where ν_{ij} is the stoichiometric coefficient for component j in reaction i and

$$r_i = k_i \prod_j C_j^{\nu_{ij}} \tag{74}$$

Here, C_j is the concentration of each component which is a function of temperature, pressure and molecular weight of the mixture, and k_i is the reaction constant of reaction i, which is a function of temperature. The temperature along the reactor is given by

$$\frac{dT}{dx} = \frac{1}{\sum F_j Cp_j}\left[\pi d_t q(x) + \frac{\pi d_t^2}{4}\sum(-\Delta H_i) r_i\right] \tag{75}$$

where q is the heat flux along the reactor, ΔH_i is the heat of reaction, and Cp_j is the heat capacity of component j. The change in molecular weight M_m can be expressed as

$$\frac{d}{dx}\left[\frac{1}{M_m}\right] = \frac{\sum_j \frac{dF_j}{dx}}{G\frac{\pi d_t^2}{4}} \tag{76}$$

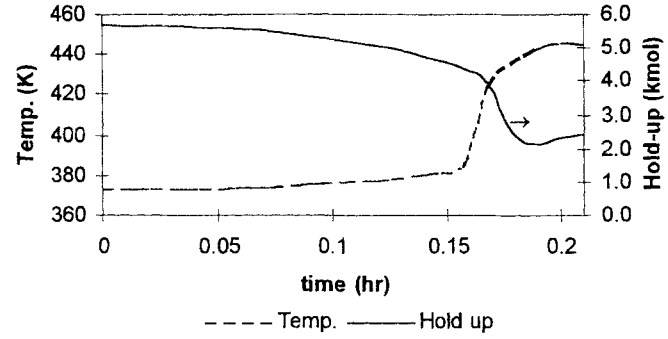

Figure 9. Last tray temperature and holdup.

Table 4. Results for Continuous Reactive Distillation

No. of Trays	No. of DAEs	Discr. Eq./Vars.	Basis Selection	Iter.	CPU (s)
10	70	1,618/1,631	MA28	∞	∞
10	70	1,618/1,631	MA28s	38	427.2
10	70	1,618/1,631	Control vars. independent	38	425.9

Table 5. PFR Results

No. of ODEs	Discr. Eq./Vars.	Basis Selection	Iter.	CPU (s)
11	1,618/1,631	MA28	66	282.5
11	1,618/1,631	MA28s	77	287.3
11	1,618/1,631	Control vars. independent	Failed	

where G is the mass-flow rate. Finally, the pressure p_t can be obtained from

$$\frac{dp_t}{dx} = \frac{\frac{d}{dx}\left[\frac{1}{M_m}\right] + \frac{1}{M_m}\left[\frac{1}{T}\frac{dT}{dx} + \left(\frac{2f}{d_t} + \frac{\xi}{\pi r_b}\right)\right]}{\frac{1}{M_m p_t} - \frac{p_t}{G^2 RT}} \tag{77}$$

where f, r_b and ξ are constants. The objective is to find an optimal profile for the heat flux along the reactor in order to maximize the production of ethylene

$$\max \; F_{\text{exit}}^{C_2H_4} \tag{78}$$

$$\text{s.t. DAEs (Eqs. 72–77)}$$

$$T_{\text{exit}} \le 1{,}180 \text{ K}$$

The problem (Eq. 78) is solved using 15 elements and 3 collocation points. Figure 10 shows the profiles of concentrations of ethane, ethylene, and hydrogen radical, as well as the heat flux profile. Note that the stiffness of the problem, due to the sharp increase of the hydrogen radical (Chen et al., 1996) is handled satisfactorily.

In this example, the optimization algorithm fails to converge (line-search failure) at iteration 50 if the discretized heat fluxes along the reactor (control variables) are chosen as independent variables. On the other hand, if we let MA28 select the basis the algorithm converges after 66 iterations, showing better stability. If we scale the system before selecting the basis with MA28 (MA28s), we obtain a similar result. The computational results are reported in Table 5.

Conclusions

In this work we present an efficient and stable decomposition method for solving DAE optimization problems. Here, orthogonal collocation is used within a sparse rSQP framework in order to obtain the control profiles and the parameters given a fixed element placement. We have shown that the sparse decomposition of the discretized system is more efficient than in previous approaches. This new strategy also allows us to detect unstable modes automatically by a simple selection of pivots. The system is stabilized using the same variables of the problem, and we suppressed the necessity of imposing final conditions and adding dummy variables as in Tanartkit and Biegler (1996).

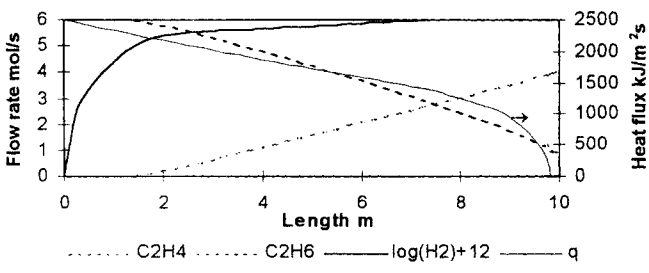

Figure 10. PFR profiles.

We have solved two well-known unstable problems and three chemical engineering examples. The first two were solved efficiently with the same stability properties as a BVP. We then consider two reactive distillation examples. These examples are of relevance, as little work has been done in the dynamic optimization of reactive distillation. Here, our algorithm also performs well, solving these larger models in a reasonable amount of time. The last problem deals with a plug-flow reactor model, which involves free radicals. In this problem, the algorithm performs well even when stiffness is present and is capable of handling the small concentrations of the free radicals (from 10^{-3} to 10^{-12} mol/s).

Two important aspects will be considered in the future. One is the element placement and approximation error control. This aspect, discussed by Tanartkit and Biegler (1997), can be implemented together with the fixed element problem using a bilevel strategy. Here, the fixed mesh problem is solved in an inner loop and the element placement and the error control are adjusted in an outer loop. The present study does not require any modification in order to deal with this bilevel strategy, as our algorithm can be used in the inner loop directly.

The second and most important aspect that we consider as future work is the storage of the A matrix and the treatment of the parameters. Our decomposition requires the storage of A, which increases with the number of elements and DAEs. To address larger problems with many finite elements, we are developing an elemental decomposition, where we will decompose separately each block of A, corresponding to a particular element. In our algorithm, the selection of a parameter as dependent variable can cause considerable fill-in, as it is present in all the elements. This problem can also be solved using an elemental decomposition by adding a dummy parameter for each element and adding the constraint that all of them should be equal to the original one. This will be the subject of a future study. Finally, we are also replacing MA28 with MA48, as it has shown to perform 15 to 30% better on these kinds of problems.

Acknowledgment

A. Cervantes is grateful for the financial support provided by "Universidad Nacional Autónoma de México." Acknowledgment is made to the donors of the Petroleum Research Fund, administered

by ACS, for partial support of this research. The authors thank Mr. Stefan Walter for development of the batch reactive distillation model.

Notation

c_k = value of equality constraints at iteration k
f = objective function for discretized NLP
$N(x)$ = n-$m \times m$ matrix from $A(x)$
t_s = times at which point constraints are applied
V = null space basis for $A(x)$
W = range space basis for $A(x)$

Greek letters

β = scaling parameter in steepest descent step
η = stepsize in doglegged step
ν = machine precision
ρ' = small positive value

Literature Cited

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

Ascher, U. M., and R. J. Spiteri, "Collocation Software for Boundary Value Differential-Algebraic Equations," *SIAM J. Sci. Comput.*, **15**(4), 939 (1994).

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).

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

Betts, J. T., and W. P. Huffman, "Application of Sparse Nonlinear Programming to Trajectory Optimization," *J. Guidance, Dyn. Cont.*, **15**(1), 198 (1992).

Betts, J. T., and P. D. Frank, "A Sparse Nonlinear Optimization Algorithm," *J. Opt. Theo. Applics.*, **82**, 543 (1994).

Bock, H. G., "Recent Advances in Parameter Identification Techniques for o.d.e.," *Numerical Treatment of Inverse Problem in Differential and Integral Equations*, P. Deuflhard and E. Hairer, eds., Birkhäuser, Heidelberg, Germany (1983).

Brenan, K. E., S. L. Campbell, and L. R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, North Holland, New York (1996).

Chen, Q., et al., "The Equation-Based SPYRO Model for Olefin-Production Applications," AIChE Meeting, New Orleans (1996).

Ciric, A., and D. Gu, "Synthesis of Nonequilibrium Reactive Distillation Processes by MINLP Optimization," *AIChE J.*, **40**(9), 1479 (1994).

Davis, T. A., and I. S. Duff, "A Combined Unifrontal/Multifrontal Method for Sparse Unsymmetric Matrices," *ACM Trans. on Mathematical Software*, in press (1997).

De Boor, C., and R. Weiss, "SOLVEBLOCK: A Package for Solving Almost Block Diagonal Linear Systems," *ACM Trans. Math. Software*, **6**(1), 80 (1980).

De Hoog, R., and M. Mattheij, "On Dichotomy and Well Conditioning in BVP," *SIAM J. Numer. Anal.*, **24**(1), 89 (1987).

De Hoog, R., and M. Mattheij, "On the Conditioning of Multipoint and Integral Boundary Value Problems," *SIAM J. Math. Anal.*, **20**(1), 200 (1989).

Dennis, J. E., and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM Publications, Philadelphia (1996).

Doherty, M. F., and G. Buzad, "Reactive Distillation by Design," *TransIChemE*, **70**, 448 (1992).

Dongarra, J. L., et al., *LINPACK User's Guide*, SIAM Publications, Philadelphia (1979).

Duff, I. S., E. M. Erisman, and J. K. Reid, *Direct Methods for Sparse Matrices*, Oxford Science Publication, New York (1992).

Duff, I. S., and J. K. Reid, "The Design of MA48: A Code for the Direct Solution of Sparse Unsymmetric Linear Systems of Equations," *ACM Trans. on Mathematical Software*, **22**(2), 187 (1996).

Gear, C. W., "Differential-Algebraic Equation Index Transformation," *SIAM J. Sci. Comput.*, **9**, 39 (1988).

Golub, H. G., and C. F. Van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins Univ. Press, Baltimore (1996).

Harwell Laboratory, *Harwell Subroutine Library Specifications* (Release 12), AEA Technology, Oxfordshire, U.K., p. 516 (1995).

Powell, M. J. D., "A Hybrid Method for Nonlinear Equations," in *Numerical Methods for Nonlinear Algebraic Equations*, P. Rabinowitz, ed., Gordon and Breach, London, p. 87 (1970).

Ruiz, C. A., M. S. Basualdo, and N. J. Scenna, "Reactive Distillation Dynamic Simulation," *TransIChemE*, 73 Part A, 363 (1995).

Schmid, C., and L. T. Biegler, "Quadratic Programming Methods for Reduced Hessian SQP," *Computers Chem. Eng.*, **18**(9), 817 (1993).

Sundaram, K. M., and G. F. Froment, "The Accuracy of the Pseudo-Steady-State Approximation for Radicals in Thermal Cracking," *Int. of Chemical Kinetics*, **10**, 1189 (1978).

Tanartkit, P., and L. T. Biegler, "A Nested Simultaneous Approach for Dynamic Optimization Problems," *Comp. and Chem. Eng.*, **20**(6,7), 735 (1996).

Tanartkit, P., and L. T. Biegler, "A Nested Simultaneous Approach for Dynamic Optimization Problems II: The Outer Problem," *Comp. and Chem. Eng.*, **21**(12), 1353 (1997).

Tanartkit, P., and L. T. Biegler, "Reformulating Ill-Conditioned DAE Optimization Problems," *Ind. Eng. Chem. Res.*, **35**(6), 1853 (1996).

Tanartkit, P., and L. T. Biegler, "Stable Decomposition for Dynamic Optimization," *Ind. Eng. Chem. Res.*, **34**, 1253 (1995).

Tanartkit, P., "Stable Computational Procedures for Dynamic Optimization in Process Engineering," PhD Diss., Dept. of Chemical Engineering, Carnegie Mellon Univ., Pittsburgh, PA (1996).

Ternet, D., and L. T. Biegler, "Recent Improvements to a Multiplier-Free Reduced Hessian Quadratic Programming Algorithm," *Comp. and Chem. Eng.*, in press (1998).

Vassiliadis, V., "Computational Solution of Dynamic Optimization Problems with General Differential-Algebriac Constraints," PhD Thesis, Univ. of London, London, U.K. (1993).

Wajge, R. M., and G. V. Reklaitis, "An Optimal Campaign Structure for Multicomponent Batch Distillation with Reversible Reaction," AIChE Meeting, Miami Beach (1995).

Wright, S. J., "Stable Parallel Algorithm for Two-Point Boundary Value Problems," *SIAM J. Sci. Stat. Comput.*, **13**(3), 742 (1992).

Manuscript received Aug. 11, 1997, and revision received Jan. 29, 1998.