ELSEVIER

Journal of Computational and Applied Mathematics 120 (2000) 41–57

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

A stable elemental decomposition for dynamic process optimization

Arturo M. Cervantes, Lorenz T. Biegler *

Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA

Received 17 December 1998; received in revised form 1 June 1999

Abstract

In Cervantes and Biegler (A.I.Ch.E.J. 44 (1998) 1038), we presented a simultaneous nonlinear programming problem (NLP) formulation for the solution of DAE optimization problems. Here, by applying collocation on finite elements, the DAE system is transformed into a nonlinear system. The resulting optimization problem, in which the element placement is fixed, is solved using a reduced space successive quadratic programming (rSQP) algorithm. The space is partitioned into range and null spaces. This partitioning is performed by choosing a pivot sequence for an LU factorization with partial pivoting which allows us to detect unstable modes in the DAE system. The system is stabilized without imposing new boundary conditions. The decomposition of the range space can be performed in a single step by exploiting the overall sparsity of the collocation matrix but not its almost block diagonal structure. In order to solve larger problems a new decomposition approach and a new method for constructing the quadratic programming (QP) subproblem are presented in this work. The decomposition of the collocation matrix is now performed element by element, thus reducing the storage requirements and the computational effort. Under this scheme, the unstable modes are considered in each element and a range-space move is constructed sequentially based on decomposition in each element. This new decomposition improves the efficiency of our previous approach and at the same time preserves its stability. The performance of the algorithm is tested on several examples. Finally, some future directions for research are discussed.

Keywords: Dynamic optimization; Nonlinear programming; DAE stability

1. Introduction

Over the past two decades, major efforts in chemical process modeling had been oriented to the development of steady-state simulation an optimization tools. The success of these tools has

* Corresponding author. Tel.: 001-610-481-5306; fax: 001-610-481-2446.
E-mail address: biegler@cmu.edu (L.T. Biegler).

0377-0427/00/$ - see front matter
PII: S0377-0427(00)00302-2

produced an increasing interest in dynamic simulation and optimization, as the demand for solving more advanced problems has grown. Common problems include control and scheduling of batch processes; startup, upset, shutdown and transient analysis; safety studies and the evaluation of control schemes.

Chemical processes are modeled dynamically using differential-algebraic equations (DAEs). The DAE formulation consists of differential equations that describe the dynamic behavior of the system, such as mass and energy balances, and algebraic equations that ensure physical and thermodynamic relations. Although a lot of work has been done in the area of dynamic simulation, the direct applicability to dynamic optimization is still limited. The use of initial-value formulations for DAEs, which can be unstable in many cases, are examples of these limiting factors. As a result, there is an increasing necessity for more efficient and reliable optimization techniques in order to extend the capabilities of the dynamic optimization tools to the dynamic optimization area.

The general dynamic optimization problem can be stated as follows:

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

s.t. DAE model

$$F\left(\frac{\mathrm{d}z(t)}{\mathrm{d}t}, z(t), y(t), u(t), t, p\right) = 0, \tag{2}$$

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

initial conditions:

$$z(0) = z^0, \tag{4}$$

point conditions:

$$H_\mathrm{s}(z(t_\mathrm{s}), y(t_\mathrm{s}), u(t_\mathrm{s}), t_\mathrm{s}, p)) = 0, \tag{5}$$

bounds:

$$\begin{aligned} &z^\mathrm{L} \leqslant z(t) \leqslant z^\mathrm{U}, \\ &y^\mathrm{L} \leqslant y(t) \leqslant y^\mathrm{U}, \\ &u^\mathrm{L} \leqslant u(t) \leqslant u^\mathrm{U}, \\ &p^\mathrm{L} \leqslant p \leqslant p^\mathrm{U}, \\ &t_\mathrm{f}^\mathrm{L} \leqslant t_\mathrm{f} \leqslant t_\mathrm{f}^\mathrm{U}, \end{aligned} \tag{6}$$

where, φ is a scalar objective function, F the differential equation constraints, G the algebraic equation constraints, H_s the additional point conditions at fixed 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 is a time-independent parameter vector. For free end time problems, t_s can also be chosen as elements of this vector.

This problem can be solved either by the variational approach or by applying a nonlinear programming (NLP) solver to the DAE model. The first approach is an extension of the calculus of variations, and works well for problems without bounds. However, if the problem requires the handling of active constraints, finding the correct switching structure and suitable initial guesses for state and adjoint variables tends to be difficult in practice. In this case, an NLP approach is recomended.

The methods that apply NLP solvers also fall into two groups, namely sequential and simultaneous strategies. In the sequential strategy, the control functions are parametrized using a finite set of control parameters. The objective and constraint functions are then evaluated for a given set of parameters by integration of the dynamic model using an existing DAE solver. The gradients with respect to the parameters (sensitivities) are obtained from this solver and a small optimization problem is solved in the space of the parameters. Several studies describe this approach; their algorithms differ in the integration technique and the method for obtaining sensitivities they use (see [20], for a review of these methods).

The simultaneous approach, in contrast, couples the solution of the DAE system with the optimization problem and therefore requires a large-scale optimization procedure. Despite this characteristic, 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, be difficult to obtain, or require excessive computational effort.

For solving simultaneous DAE optimization problems, successive quadratic programming (SQP) is often the method of choice. Here large-scale SQP methods for DAE optimization problems can be classified as full-space or reduced space. Full-space methods take advantage of the DAE optimization problem structure and the overall sparsity of the model. These are especially well suited for problems with many degrees of freedom [4,5], as the optimality conditions can be stored and factored very efficiently. An important feature of these methods is that second derivatives of the objective and constraint functions are usually required, and special precautions are necessary to ensure descent properties.

When solving dynamic optimization problems in process engineering, the number of state variables is usually much larger than the number of control variables (degrees of freedom < 100). Here, a full-space algorithm which exploits the almost block diagonal structure of the DAE optimization problem was developed [1]. This approach decoupled the optimality conditions for each block of the quadratic programming (QP) subproblem using an affine transform. In this way, the first-order conditions in the state and control variables can be solved recursively, turning the effort of solving it linear with the number of blocks. On the other hand, for many of these examples a reduced space approach (rSQP) is easier to implement. With this approach, either projected Hessian matrices or their quasi-Newton approximations may be used, thus avoiding the necessity of second derivatives. An efficient algorithm can be constructed by decoupling the algorithm into range and null spaces, solving a smaller QP subproblem at every iteration. A similar partially reduced strategy was developed by Schulz [14]. In addition, specialized decomposition procedures that take advantage of the structure of the Hessian were explored by Steinbach [15].

In a previous study [9], we presented a simultaneous reduced-Hessian successive quadratic programming (rSQP) algorithm. Here, the DAE system is discretized with a modified sparse version of the LSYSLV subroutine (from the DAE solver COLDAE). The variables are then partitioned into dependent and independent variables. A Newton step for the dependent variables is obtained by

solving a square system of equations in the range space. The step for the discretized independent variables is obtained by solving a quadratic programming (QP) problem. For this system, unstable modes are detected by selecting a numerically stable pivot sequence for the LU factorization of the collocation matrix. Therefore, unstable modes are dealt with by selecting state variables as decisions in the partitioning step. The system is thus stabilized without imposing additional boundary conditions.

In this approach, the LU factorization exploits the overall sparsity of the collocation matrix, but not its almost block diagonal structure. This is important for process engineering problems, because these blocks can be large and sparse as well. Therefore, to handle much larger DAE optimization problems, this elemental structure needs to be exploited. With many elements, the storage of the A matrix requires considerable amounts of memory, which can slow down the algorithm and even cause its failure. The objective of this work is to present a decomposition strategy that exploits the almost block diagonal structure of the collocation matrix and at the same time preserves the stability properties of our previous decompostion.

In the following section we briefly describe the DAE discretization as well as the general nonlinear programing algorithm we are using. Section 3 then includes the methodolgy used to apply the element by element decomposition under this general framework. In Section 4 we present some examples, and Section 5 concludes the paper and outlines some future directions.

2. DAE optimization

The DAE optimization problem is converted into an NLP by approximating state and control profiles to a family of polynomials on finite elements. Here, we use a monomial basis representation [3] for the differential profiles, as follows:

$$z(t) = z_{i-1} + h_i \sum_{q=1}^{\text{ncol}} \Omega_q \left(\frac{t - t_{i-1}}{h_i} \right) \frac{\mathrm{d}z}{\mathrm{d}t_{i,q}}, \tag{7}$$

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, $\mathrm{d}z/\mathrm{d}t_{i,q}$ is the value of its first derivative in element i at the collocation point q, and Ω_q is a polynomial of order ncol, satisfying

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

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

where ρ_r is the rth collocation point within each element. Radau points are used as they allow setting constraints easily at the end of each element and they stabilize the system more efficiently if unstable modes are present. This will be discussed in the next section. The monomial representation is recommended because it leads to smaller condition numbers and smaller rounding errors [3].

The control and algebraic profiles are approximated using the same monomial basis representation, which, in the absence of continuity across elements, takes the form

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

$$u(t) = \sum_{q=1}^{\text{ncol}} \psi_q \left(\frac{t - t_{i-1}}{h_i} \right) u_{i,q}. \tag{9}$$

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

$$\psi(\rho_r) = \delta_{q,r}. \tag{10}$$

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. It should be mentioned that with representation (7), the bounds on the differential variables can only be enforced directly at element boundaries. However, they can be enforced at the collocation points by writing appropiate point constraints.

Here it is assumed that the number of finite elements *ne*, and their lengths are pre-determined. With this assumption, the substitution of Eqs. (7)–(9) into (1)–(6) leads to the following nonlinear programming problem (NLP):

$$\min_{x \in \Re^n} f(x) \tag{11}$$

$$\text{s.t.} \quad c(x) = 0, \tag{12}$$

$$x^{\mathrm{L}} \leqslant x \leqslant x^{\mathrm{U}}, \tag{13}$$

where $x = (\mathrm{d}z/\mathrm{d}t_{i,q}, z_i, y_{i,q}, u_{i,q}, t, p)^{\mathrm{T}}$,

$f : \Re^n \to \Re$ and $c : \Re^n \to \Re^m$.

2.1. Reduced-Hessian successive quadratic programming (rSQP)

In [9,16] we have shown 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 ($n \gg m$). rSQP also adds robustness to the solution procedure by performing local factorizations. This allows us to preserve and exploit the structure of the problem and to detect ill-conditioning due to unstable modes. At each iteration k, a search direction d_k is obtained by solving a quadratic programming subproblem (QP1) created from (11) to (13):

$$\min_{d \in \Re^n} g(x_k)^{\mathrm{T}} d_k + \tfrac{1}{2} d_k^{\mathrm{T}} B(x_k) d_k \tag{14}$$

$$\text{s.t.} \quad c_k + A_k d_k = 0, \tag{15}$$

$$x^{\mathrm{L}} \leqslant x_k + d_k \leqslant x^{\mathrm{U}}, \tag{16}$$

where g is the gradient of f, B denotes the Hessian of the Lagrangian function, and $A_k = A(x_k)$ is the $m \times n$ Jacobian of the constraints at iteration k, and $c_k = c(x_k)$.

The variables are further partitioned into m dependent (W space) and $n-m$ independent (V space) variables. The independent variable space occupies the null space of A while the dependent variable space occupies the range space of A^{T}. Note that the control variables and parameters are not necessarily the independent variables; this will depend on the basis selection that is discussed later. With this partition A takes the form

$$A(x) = [C(x)\ N(x)], \tag{17}$$

where the $m \times m$ basis matrix $C(x)$ is nonsingular. With this partition, the search direction is written as

$$d_k = W_k p_W + V_k p_V, \tag{18}$$

where the matrix V satisfies

$$A_k V_k = 0. \tag{19}$$

Here we choose

$$V_k = \begin{bmatrix} -C(x_k)^{-1} N(x_k) \\ I \end{bmatrix} \tag{20}$$

and

$$W_k = \begin{bmatrix} I \\ 0 \end{bmatrix}. \tag{21}$$

The range space direction p_W can now be determined as

$$p_W = -[A_k W_k]^{-1} c_k \tag{22}$$

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

$$\min_{p_V \in \Re^{n-m}} (V_k^{\mathrm{T}} g_k + V_k^{\mathrm{T}} B_k W_k p_W)^{\mathrm{T}} p_V + \tfrac{1}{2} p_V^{\mathrm{T}} (V_k^{\mathrm{T}} B_k V_k) p_V \tag{23}$$

$$\text{s.t.} \quad x^{\mathrm{L}} - x_k - W_k p_W \leqslant V_k p_v \leqslant x^{\mathrm{U}} - x_k - W_k p_W. \tag{24}$$

The reduced QP subproblem is solved using the active set algorithm QPKWIK [18], which is an improved version of the original one by Schmid and Biegler [13]. This version incorporates a trust region, which restricts the stepsize of d_k to an area around x_k where the Taylor series approximations to the Lagrangian and the constraints provide an adequate model for the nonlinear problem. The addition of a trust region can improve the convergence of the algorithm, especially when the second-order information for the problem is poor. This implementation adds a trust region constraint $\|d_k\| \leqslant \Delta_k$ to (23)–(24) and moves the search direction from a Newton step direction to a steepest descent direction. The trust region does not affect the direction of the $W p_w$ step. Only its length changes, as this step is generated before the solution of the QP.

3. Element by element decomposition

In this section we consider a more detailed decomposition for the range and null space steps. We first observe that the Jacobian of the discretized system of equations is represented by

$$A = \begin{bmatrix}
Z_{\text{init}}^0 & & & & & & & & & & & & & & 0 & & & & & & & & 0 \\
Z_1^0 & DZ_1^1 & Y_1^1 & & & & & & & & & & & & U_1^1 & & & & & & & & P_1^1 \\
Z_2^0 & DZ_2^1 & Y_2^1 & & & & & & & & & & & & U_2^1 & & & & & & & & P_2^1 \\
\vdots & \vdots & \vdots & & & & & & & & & & & & \vdots & & & & & & & & \vdots \\
Z_{\text{ncol}}^0 & DZ_{\text{ncol}}^1 & Y_{\text{ncol}}^1 & & & & & & & & & & & & U_{\text{ncol}}^1 & & & & & & & & P_{\text{ncol}}^1 \\
Z_{\text{a}}^0 & D_{\text{a}}^1 & Y_{\text{a}}^1 & & & & & & & & & & & & U_{\text{a}}^1 & & & & & & & & P_{\text{a}}^1 \\
I & D^1 & 0 & -I & & & & & & & & & & & 0 & & & & & & & & 0 \\
 & & & Z_1^1 & DZ_1^2 & Y_1^2 & & & & & & & & & 0 & U_1^2 & & & & & & & P_1^2 \\
 & & & & \ddots & \ddots & & & & & & & & & & \ddots & & & & & & & \vdots \\
 & & & & & \ddots & \ddots & & & & & & & & & & \ddots & & & & & & \vdots \\
 & & & & & & Z_{\text{ncol}}^{ne-1} & DZ_{\text{ncol}}^{ne} & Y_{\text{ncol}}^{ne} & & & & & & & & & & & U_{\text{ncol}}^{ne} & P_{\text{ncol}}^{ne} \\
 & & & & & & Z_{\text{a}}^{ne-1} & D_{\text{a}}^{ne} & Y_{\text{a}}^{ne} & & & & & & & & & & & U_{\text{a}}^{ne} & P_{\text{a}}^{ne} \\
 & & & & & & I & D^{ne} & 0 & -I & & & & & & & & & & & & &
\end{bmatrix}, \tag{25}$$

where I represents the identity matrix, and D^i is a matrix containing the coefficients of the continuity equations of the ith element. Z_q^i, DZ_q^i, Y_q^i, U_q^i and P_q^i represent the Jacobian of the collocation equations with respect to z^i, $\mathrm{d}z/\mathrm{d}t^{i,q}$, y_q^i, u_q^i and p, at collocation point q and element i. Z_{a}^i, D_{a}^i, Y_{a}^i, U_{a}^i and P_{a}^i, correspond to the Jacobian of the additional constraints. As indicated in (25), it is assumed that these constraints can be separated by elements.

In Cervantes and Biegler [9] we decompose this matrix A in a single step and exploit the overall sparsity but not the almost block diagonal structure. However, as the number of DAEs and elements is increased, the necessity of exploiting this feature becomes evident. In particular, storage and full factorization of A can slow down the algorithm or even cause its failure when there is not enough memory available.

These problems can be addressed by applying the original decomposition strategy in each finite element. In this way, the necessity of storing the complete matrix A is avoided. The factorization is performed over smaller matrices, each one representing a finite element. In most cases these matrices will have the same structure and consequently allow re-use of the sparse matrix pivot sequence. To explore this decomposition, consider the rows and columns of A, corresponding to element i,

$$A^i = \begin{bmatrix}
Z_1^{i-1} & DZ_1^i & Y_1^i & 0 & U_1^i & P_1^i \\
Z_2^{i-1} & DZ_2^i & Y_2^i & 0 & U_2^i & P_2^i \\
\vdots & \vdots & \vdots & 0 & \vdots & \vdots \\
Z_{\text{ncol}}^{i-1} & DZ_{\text{ncol}}^i & Y_{\text{ncol}}^i & 0 & U_{\text{ncol}}^i & P_{\text{ncol}}^i \\
Z_{\text{a}}^{i-1} & D_{\text{a}}^i & Y_{\text{a}}^i & 0 & U_{\text{a}}^i & P_{\text{a}}^i \\
I & D^i & 0 & -I & 0 & 0
\end{bmatrix}. \tag{26}$$

Here, it is assumed that the additional point constraints (the last row in (26)) can be separated by elements. Now if no parameters p are present, the decomposition of this matrix can be performed directly, as all the variables can be eliminated locally. In the case that a parameter is present, the last column of A^i, which corresponds to the parameters, will be coupled to the entire system. In this case, we create dummy parameters for each finite element. An extra constraint for each finite element is also added in the QP, requiring all of these dummy parameters to be equal. This idea is similar to the technique applied in [19] for treating design variables in multiperiod problems. The matrix A^i will look the same after adding the dummy parameters, but now the last column will correspond to local variables.

Each of these small matrices A^i is obtained using a modified version of the subroutine LSYSLV in COLDAE [2] a well-tested algorithm for collocation on finite elements. These modifications allow us to work with rectangular matrices which are stored directly in sparse form.

3.1. Basis selection and ill-conditioning detection

In order to apply the rSQP algorithm described in Section 2, each matrix A^i is partitioned into a range and null space basis. This partition is performed by applying an LU factorization with partial pivoting on the rectangular system A^i. Following [10], this LU factorization will yield a dichotomous system in each element. If an unstable mode is present in the DAE, A^i 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, $\mathrm{d}z_j/\mathrm{d}t_{\mathrm{ncol}}$ would be specified and would correspond to a column in the null space. Correspondingly, a column corresponding to a control variable or a parameter would be added to the range space. By considering the variables that span the columns of the null space to be specified as decisions, the decomposition approach is equivalent to solving a discretized, linear BVP.

Consider now the partition of A^i into columns,

$$A^i = [A^i_{z^0}\ A^i_{\mathrm{d}z}\ A^i_y\ A^i_{z^f}\ A^i_u\ A^i_p]. \tag{27}$$

This matrix will have the same structure for all the elements, and the pivot selection of the LU factorization will partition the variables of each element i into dependent and independent variables. Here there are a number of ways to obtain a stable partition. First, one can perform the basis selection over more than one element or even on the whole matrix A, as in [9]. This approach provides a time horizon that is long enough to detect unstable modes. After working with these larger matrices, we can then apply the same variable partition to all the matrices A^i. However, this procedure can be expensive and should be considered only if elemental partitioning does not capture all of the unstable modes. As a result, we instead consider partitioning within each element.

For linear systems the partition in each element would be the same and this allows us to re-use the sparse matrix factorization. For nonlinear systems the partition can be different and there are two possibilities to consider. First, when a dummy parameter or a control variable is selected as dependent in element i and it was not selected in the elements $i-1,\ldots,1$, we force it to be dependent for all elements; and the corresponding state will always be selected as independent. The second possibility is that in successive elements, different state variables are chosen to leave the basis while the same control or parameter enters the basis. In this case a different factorization has to be used for each element. In order to apply the elemental decomposition, the independent variables should

only include columns from A_u^i, A_p^i and A_{dz}^i. This partitioning is performed following the procedure described next.

To partition A^i, ill-conditioning is avoided by selecting a pivot sequence for the LU factorization of the A^i matrix [11]. The objective of the pivot selection is to minimize the fill-in during the factorization but at the same time achieve numerical stability. Here the Harwell subroutine MA28 is used for the selection of the sequence. This subroutine uses Markowitz's ordering strategy with threshold pivoting for numerical stability. That is, it will choose the matrix element a_{mj} as a pivot if it minimizes the quantity:

$$(r_m - 1)(c_j - 1) \tag{28}$$

over all entries of the reduced matrix that satisfy the inequality

$$|a_{mj}^{(k)}| \geqslant \hat{u} \max_{l \geqslant k} |a_{ml}^{(k)}|. \tag{29}$$

Here r_m is the number of entries in row m of the reduced matrix, c_j is the number of entries in column j and $\hat{u}$ is a preset threshold pivoting parameter in the range $0 < \hat{u} \leqslant 1$.

Eq. (29) therefore allows selection of columns from A_u^i and A_p^i to replace the unstable modes of the DAE, by selecting a pivot sequence with numerical stability. A suitable modification of this strategy also allows to restrict the variables that can be selected as independent. For example, 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 sufficiently large number. In this way, MA28 automatically sets a pivot in those columns (of A_y^i) and chooses these variables as dependent. The same procedure is applied to ensure that the variables corresponding to $A_{z^0}^i$ and $A_{z^f}^i$ are selected as dependent variables.

By controlling the basis selection and exchanging columns from A_u^i, A_p^i and A_{dz}^i we now write the partitioned matrix A^i as

$$A^i = \begin{bmatrix} T^i & C^i & 0 & N^i \\ I & \hat{C}^i & -I & \hat{N}^i \end{bmatrix}, \tag{30}$$

where the first three colums correspond to the range space, and the last one to the null space. T^i represents the blocks corresponding to Z^{i-1} and Z_a^{i-1}; C^i now contains Y^i and Y_a^i and elements from DZ^i, DZ_a^i, U^i, U_a^i, P^i and P_a^i; $\hat{C}^i$ contains elements from D^i and D_a^i, P^i and P_a^i; and $\hat{N}^i$ can contain elements from D^i and D_a^i.

3.2. Wp_w step and QP subproblem construction

As the whole matrix A is not stored anymore, we have to perform a forward elimination in order to obtain the step for the dependent variables p_w and $C^{-1}N$ for the construction of the QP subproblem.

After the basis is selected, we can now use Eq. (30) to represent the matrix A as

$$A = \left[\begin{array}{ccccccccc|ccc} I & & & & & & & & & 0 & & \\ T^1 & C^1 & & & & & & & & N^1 & & \\ I & \hat{C}^1 & -I & & & & & & & \hat{N}^1 & & \\ & & T^2 & C^2 & & & & & & & N^2 & \\ & & I & \hat{C}^2 & -I & & & & & & \hat{N}^2 & \\ & & & & T^3 & C^3 & & & & & & N^3 \\ & & & & & \ddots & \ddots & & & & & \ddots \end{array}\right] = [C \mid N] \tag{31}$$

and the corresponding right-hand sides are

$$c^{\mathrm{T}} = [0\ c^1\ \hat{c}^1\ c^2\ \hat{c}^2\ c^3\ \cdots]. \tag{32}$$

By premultiplying T^i and C^i by the inverse of C^i in each element, the matrix C becomes

$$\tilde{C} = \left[\begin{array}{ccccccccc} I & & & & & & & & \\ (C^1)^{-1}T^1 & I & & & & & & & \\ I & \hat{C}^1 & -I & & & & & & \\ & & (C^2)^{-1}T^2 & I & & & & & \\ & & I & \hat{C}^2 & -I & & & & \\ & & & & (C^i)^{-1}T^3 & I & & & \\ & & & & & \ddots & \ddots & & \end{array}\right]. \tag{33}$$

We now obtain the step for the dependent variables by performing a forward elimination for the right-hand side:

$$\left[\begin{array}{ccccccccc|c} I & & & & & & & & & 0 \\ (C^1)^{-1}T^1 & I & & & & & & & & (C^1)^{-1}c^1 \\ I & \hat{C}^1 & -I & & & & & & & \hat{c}^1 \\ & & (C^2)^{-1}T^2 & I & & & & & & (C^2)^{-1}c^2 \\ & & I & \hat{C}^2 & -I & & & & & \hat{c}^2 \\ & & & & (C^i)^{-1}T^3 & I & & & & (C^3)^{-1}c^3 \\ & & & & & \ddots & \ddots & & & \vdots \end{array}\right]. \tag{34}$$

At the same time, we can also obtain the matrix $C^{-1}N$, by performing the forward elimination for different right-hand sides

$$\left[\begin{array}{ccccccccc|ccc} I & & & & & & & & & 0 & 0 & \\ (C^1)^{-1}T^1 & I & & & & & & & & (C^1)^{-1}N^1 & 0 & \\ I & \hat{C}^1 & -I & & & & & & & \hat{N}^1 & 0 & \\ & & (C^2)^{-1}T^2 & I & & & & & & \vdots & (C^2)^{-1}N^2 & \\ & & I & \hat{C}^2 & -I & & & & & \vdots & \hat{N}^2 & \\ & & & & (C^i)^{-1}T^3 & I & & & & \vdots & \vdots & (C^3)^{-1}N^3 \\ & & & & & \ddots & \ddots & & & \vdots & \vdots & \ddots \end{array}\right]. \tag{35}$$

Table 1
Computational results for one-parameter problem

Basis	Discretized Vars		Iterations/CPU (s)	
	Global	Elemental	Global	Elemental
MA28	243	272	5/0.4	7/0.5
p indep.	243	272	Failed ($Wp_W = 10^{29}$)	Failed ($Wp_W = 10^{29}$)

The factorizations of the matrices C^i are performed with the Harwell subroutine MA48, a well-tested sparse linear solver.

4. Examples

In this section, we first present two well-known parameter estimation problems. The objective here is to show that the stability properties of our previous approach still apply for the elemental decomposition. We then demonstrate the efficiency of the new decomposition on a larger chemical engineering example. All the CPU times reported were obtained on an HP 9000/C110 workstation.

4.1. Ill-conditioned parameter estimation with one parameter

This is a parameter estimation problem with two states and one parameter due to Bock [6]. We formulate this problem as an initial value problem:

$$\min \quad \sum_{i=1}^{n} (z_2^m(i) - z_2(i))^2 \tag{36}$$

$$\text{s.t.} \quad \frac{\mathrm{d}z_1}{\mathrm{d}t} = z_2, \tag{37}$$

$$\frac{\mathrm{d}z_2}{\mathrm{d}t} = \tau^2 z_1 - (\tau^2 + p^2)\sin(pt), \tag{38}$$

$$[z_1(0)\ z_2(0)]^{\mathrm{T}} = [0\ \pi]^{\mathrm{T}}. \tag{39}$$

The objective is to estimate the parameter p ($p = \pi$) given true values for z_2 corrupted with random noise at data points equal to the mesh points. The unknown parameter p was initialized to 2 and the value of the constant τ was set to 100. We solve the problem using 30 finite elements and three collocation points.

The ODE system contains one unstable mode. As seen in Table 1, if the problem is posed by setting the parameter p as independent variable, the algorithm fails to converge. On the other hand, if we select the basis with MA28 we are able to detect the unstable mode, and the algorithm converges in five iterations using our previous (global) approach and in seven iterations using the elemental decomposition. The difference in iterations is due to small differences in the QP solutions, especially in converging to a tight tolerance, 10^{-6}, on the Kuhn–Tucker conditions.

The optimal state profiles were compared with the analytical solution to ensure that the mesh selection is valid. For both variables the error is not larger that the tolerance of the solution. In this case, the utilization of an automatic integrator, such as DASSL, is not possible because of the unstable nature of the problem.

4.2. Ill-conditioned parameter estimation with three parameters

We next consider the parameter estimation problem, modified from Wright [22], and also studied in [17]:

$$\min \quad \sum_{i=1}^{n} (z_1^m(i) - z_1(i))^2 + \sum_{i=1}^{n} (z_2^m(i) - z_2(i))^2 \tag{40}$$

$$\text{s.t.} \quad \left[\frac{\mathrm{d}z_1}{\mathrm{d}t} \ \frac{\mathrm{d}z_2}{\mathrm{d}t} \ \frac{\mathrm{d}z_3}{\mathrm{d}t} \ \frac{\mathrm{d}z_4}{\mathrm{d}t} \ \frac{\mathrm{d}z_5}{\mathrm{d}t} \right]^{\mathrm{T}} = M[z_1 \ z_2 \ z_3 \ z_4 \ z_5]^{\mathrm{T}} + f(t), \tag{41}$$

$$[z_1(0) \ z_2(0) \ z_3(0) \ z_4(0) \ z_5(0)]^{\mathrm{T}} = [1 \ 1 \ 1 \ 1 \ 1]^{\mathrm{T}}, \tag{42}$$

where

$$M = \begin{bmatrix} -\Psi_1 \cos(2\omega_1 t) & 0 & \omega_1 + \Psi_1 \sin(2\omega_1 t) & 0 & 0 \\ 0 & -\Psi_2 \cos(2\omega_2 t) & 0 & \omega_2 + \Psi_2 \sin(2\omega_2 t) & 0 \\ -\omega_1 + \Psi_1 \sin(2\omega_1 t) & 0 & \Psi_1 \cos(2\omega_1 t) & 0 & 0 \\ 0 & -\omega_2 + \Psi_2 \sin(2\omega_2 t) & 0 & \Psi_2 \cos(2\omega_2 t) & 0 \\ 0 & 0 & 0 & 0 & \Psi_3 \end{bmatrix}, \tag{43}$$

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

$$[z_1(t) \ z_2(t) \ z_3(t) \ z_4(t) \ z_5(t)]^{\mathrm{T}} = [e^t \ e^t \ e^t \ e^t \ e^t]^{\mathrm{T}}. \tag{44}$$

This corresponds to

$$f(t) = \begin{bmatrix} (1 + 1000\cos(2\omega_1 t) - \omega_1 - 1000\sin(2\omega_1 t))e^t \\ (1 + 100\cos(2\omega_2 t) - \omega_2 - 100\sin(2\omega_2 t))e^t \\ (1 - 1000\cos(2\omega_1 t) + \omega_1 - 1000\sin(2\omega_1 t))e^t \\ (1 - 100\cos(2\omega_2 t) + \omega_2 - 100\sin(2\omega_2 t))e^t \\ -9e^t \end{bmatrix}. \tag{45}$$

The objective is to estimate the parameters Ψ_1, Ψ_2, Ψ_3 ($\Psi_i = 10, 100, 1000$) given 30 measured data points corrupted with random noise (σ) for z_1 and z_2. The parameters are initialized to 12, 120 and 1200, respectively. The system is unstable because of the presence of three unstable modes.

The problem was solved using 30 elements and three collocation points. The constants ω_1 and ω_2 were set to 30. The computational results are presented in Table 2. If we select the parameters Ψ_i as independent variables, the decomposition becomes unstable and the algorithm fails to converge. If

Table 2
Computational results for three-parameter problem

Basis	Discretized Vars		Iterations/CPU(s)	
	Global	Elemental	Global	Elemental
MA28	608	695	25/5.6	31/6.5
Ψ_i indep.	608	695	Failed ($Wp_W = 10^{57}$)	Failed ($Wp_W = 10^{57}$)

we let MA28 select the basis the algorithm converges in 25 iterations for the global decomposition and in 31 iterations for the elemental decomposition. Again, the difference in iterations is due to small differences in the QP solutions, especially in converging to a tight tolerance, 10^{-6}, on the Kuhn–Tucker conditions. We also compared the optimal state profiles with the analytical solution to ensure that the mesh selection was correct. For none of them was the error larger than the tolerance of the optimization problem.

4.3. Batch reactive distillation

The dynamic model is an index one DAE system, which was obtained by reformulating the dynamic model of Ruiz and coworkers [12]. The general model of the column consists of the following equations:

$$\frac{\mathrm{d}M_i}{\mathrm{d}t} = F_i + V_{i+1} + L_{i-1} - V_i - L_i + \sum_{n=1}^{nr} R_{i,n}. \tag{46}$$

Here, M_i corresponds to the molar holdup on tray i, V_i and L_i are the vapor and liquid flowrates, 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 mole fraction x of each component j can be expressed as

$$M_i \frac{\mathrm{d}x_{i,j}}{\mathrm{d}t} = F_i(z_{i,j} - x_{i,j}) + V_{i+1}(y_{i+1,j} - x_{i,j}) + L_{i-1}(x_{i-1,j} - x_{i,j}) - V_i(y_{i,j} - x_{i,j}) + \hat{R}, \tag{47}$$

where

$$\hat{R} = \sum_{n=1}^{nr} v_{j,n} \frac{M_i}{\rho_i} r_{i,n} - x_{i,j} \sum_{n=1}^{nr} R_{i,n}. \tag{48}$$

ρ_i is the liquid molar density, $z_{i,j}$ is the molar fraction of j in the feed, $y_{i,j}$ is the vapor mole fraction, $v_{j,n}$ is the stoichiometric coefficient, and $r_{i,n}$ is the rate of production per unit volume. The phase equilibrium relationship is represented by

$$y_{i,j} = K_{i,j} x_{i,j}, \tag{49}$$

$$K_{i,j} = f_j(T_i), \tag{50}$$

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

$$\sum_{j=1}^{nc} y_{i,j} = 1. \tag{51}$$

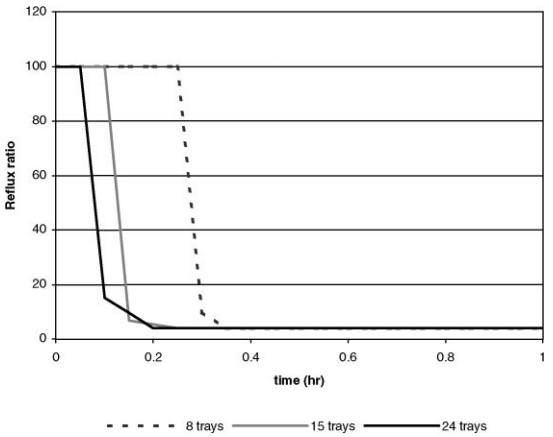

Fig. 1. Reflux ratio.

The vapor flowrates are calculated using a modified index one energy balance (see [9])

$$\frac{1}{M_i}\left(V_{i+1}(h^{\mathrm{v}}_{i+1}-h^{\mathrm{l}}_i)+L_{i-1}(h^{\mathrm{v}}_{i-1}-h^{\mathrm{l}}_i)-V_i(h^{\mathrm{v}}_i-h^{\mathrm{l}}_i)+\sum_{n=1}^{nr}\frac{M_i}{\rho_i}r_{i,n}\Delta H^{\mathrm{R}}_n\right)$$
$$=\sum_{j=1}^{nc}\frac{\partial h^{\mathrm{l}}_i}{\partial x_{i,j}}\frac{\mathrm{d}x_{i,j}}{\mathrm{d}t}-\frac{\partial h^{\mathrm{l}}_i}{\partial T_i}\frac{\sum_{j=1}^{nc}K_{i,j}(\mathrm{d}x_{i,j}/\mathrm{d}t)}{\sum_{j=1}^{nc}x_{i,j}(\mathrm{d}K_{i,j}/\mathrm{d}T_i)}, \tag{52}$$

where h^{v} and h^{l} are the vapor and liquid enthalpies, and ΔH^{R}_n is the heat of reaction.

This example considers the reversible reaction between acetic acid and ethanol [21],

$$\mathrm{CH_3COOH}+\mathrm{CH_3CH_2OH}\leftrightarrow\mathrm{CH_3COOCH_2CH_3}+\mathrm{H_2O}.$$

The model consists of $12+5\mathrm{nt}$ differential and $6+5\mathrm{nt}$ 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. Only three collocation points and five elements were required to obtain accurate state variable profiles. The objective is to maximize the amount of distillate D produced within 1 hr by manipulating the reflux ratio as a function of time, as follows:

$$\max\int_0^{t_{\mathrm{f}}=1}D\,\mathrm{d}t \tag{53}$$

$$\text{s.t.}\quad \text{DAE model (Eqs. (46)–(52))}, \tag{54}$$

$$x_D^{\mathrm{Ester}}\geqslant 0.4800.$$

Here, the mole fraction of ethyl acetate in the final distillate should be at least 0.48. We solved this problem for 8, 15 and 24 trays. In all cases the problem was initialized with a feasible point with a constant reflux ratio of 20. For this example, the discretized control variables were selected as independent variables, because the DAE system does not have unstable modes.

As seen in Figs. 1 and 2, 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.

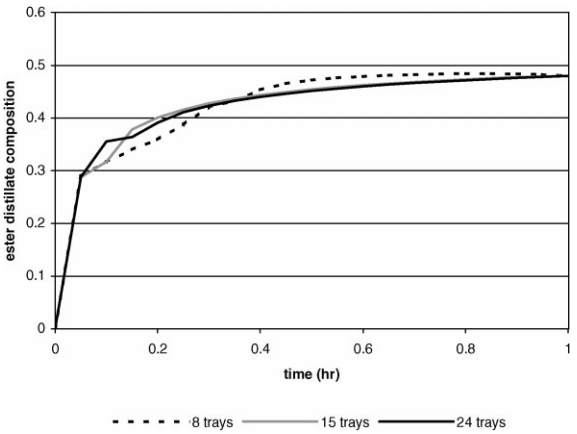

Fig. 2. Ethyl acetate distillate composition.

Table 3
Computational results for batch distillation column

Trays	DAEs	NLP Vars.	Iter.	Total CPU(s)		$Wp_W, C^{-1}N$ (s/iter)	
				Global	Elemental	Global	Elemental
8	98	1788	14	56.4	37.2	3.3	1.9
15	168	3048	32	245.7	207.5	7.7	5.6
24	258	4678	45	1083.2	659.3	22.4	12.9

The computational results are presented in Table 3. Using a Kuhn–Tucker tolerance of 10^{-6}, both decomposition strategies require the same number of SQP iterations but the elemental decomposition is almost twice as fast. Moreover, by storing only the elemental matrices, far fewer memory resources are required. In this case, we compare the optimal state profiles with those obtained with DASSL (tolerance 10^{-10}); the error was never larger than 10^{-6}.

5. Conclusions and future work

We have developed an efficient and stable method for solving DAE optimization problems. We show that IVPs can be solved with a forward elimination by transfering the unstable modes to the independent variables of the rSQP algorithm. Under this scheme, the necessity of adding final conditions is avoided. Therefore, the resulting algorithm is, in principle, as efficient as any sequential algorithm regarding the memory requirements. At the same time, the stability properties of the simultaneous approaches are preserved.

Future research will concentrate on further bottlenecks in our reduced space, simultaneous approach. One important aspect is that the presence of a large number of constraints (due to discretized profile bounds) can still lead to a huge quadratic programming problem, especially as the number of DAEs and finite elements is increased. This becomes a concern in our algorithm, as the solution

time of the QP subproblem increases. To solve this problem we consider as future work the implementation of a barrier method [7,8] which will allow us to incorporate the inequality constraints into the objective function and at the same time use the elemental decomposition presented in this work. Also, while the degrees of freedom in dynamic process engineering problems are relatively small ($n \gg n - m$), increasing the number of finite elements and collocation points also leads to large, reduced space QPs. These will be explored in the future by considering the elemental structure of the Kuhn–Tucker matrix in greater detail.

Acknowledgements

Funding from the Universidad Nacional Autónoma de México, the National Science Foundation (CTS9729075) and the American Chemical Society – Petroleum Research Fund (31243 AC9) is gratefully acknowledged.

References

[1] J. Albuquerque, V. Gopal, G. Staus, L.T. Biegler, B.E. Ydstie, Interior point SQP strategies for structured process optimization problems, Comput. Chem. Eng. 21 (1997) 283.
[2] U.M. Ascher, R.J. Spiteri, Collocation software for boundary value differential-algebraic equations, SIAM J. Sci. Comput. 15 (1994) 939–952.
[3] G. Bader, U. Ascher, A new basis implementation for mixed order boundary value ODE solver, SIAM J. Sci. Comput. 8 (1987) 483–500.
[4] J.T. Betts, P.D. Frank, A sparse nonlinear optimization algorithm, J. Optim. Theory Appl. 82 (1994) 543.
[5] J.T. Betts, W.P. Huffman, Application of sparse nonlinear programming to trajectory optimization, J. Guidance Dynamics Control 15 (1992) 198.
[6] H.G. Bock, Numerical treatment of inverse problem in differential and integral equation, in: P. Deuflhard, E. Hairer (Eds.), Recent Advances in Parameter Identification Techniques for o.d.e., Heidelberg, 1983.
[7] R.H. Byrd, J.C. Gilbert, J. Nocedal, A trust region method based on interior point techniques for nonlinear programming, Report OTC 96/02, Optimization Technology Center, Northwestern University, 1996.
[8] R.H. Byrd, M.E. Hribar, J. Nocedal, An interior point algorithm for large scale nonlinear programming, Technical Report, Northwestern University, 1997.
[9] A. Cervantes, L.T. Biegler, Large-scale DAE optimization using simultaneous nonlinear programming formulations, A.I.Ch.E. J. 44 (1998) 1038.
[10] R. De Hoog, M. Mattheij, On dichotomy and well conditioning in BVP, SIAM J. Numer. Anal. 24 (1987) 89–105.
[11] I.S. Duff, E.M. Erisman, Direct methods for Sparse Matrices, Oxford Science Publication, New York, 1992.
[12] C.A. Ruiz, M.S. Basualdo, N.J. Scenna, Reactive distillation dynamic simulation, Trans. Inst. Chem. Eng. 73 (1995) 363–378.
[13] C. Schmid, L.T. Biegler, Quadratic programming methods for reduced Hessian SQP, Comput. Chem. Eng. 18 (1993) 817–832.
[14] V.H. Schulz, Solving discretized optimization problems by partially reduced SQP methods, Comput. Visual Sci. 1 (1997).
[15] M.C. Steinbach, Fast recursive SQP methods for large-scale optimal control problems, Ph.D. Thesis, University of Heidelberg, Germany, 1995.
[16] P. Tanartkit, L.T. Biegler, Stable decomposition for dynamic optimization, Ind. Eng. Chem. Res. 34 (1995) 1253–1266.
[17] P. Tanartkit, L.T. Biegler, A nested simultaneous approach for dynamic optimization problems II: the outer problem, Comput. Chem. Eng. 21 (1997) 1365.

[18] D. Ternet, L.T. Biegler, Recent improvements to a multiplier-free reduced Hessian quadratic programming algorithm, Comput. Chem. Eng. 22 (1997) 963.
[19] D.K. Varvarezos, L.T. Biegler, I.E. Grossmann, Multiperiod design optimization with SQP decomposition, Comput. Chem. Eng. 18 (1994) 579–595.
[20] V. Vassiliadis, Computational solution of dynamic optimization problems with general differential-algebraic constraints, Ph.D. Thesis, University of London, London, UK, 1993.
[21] R.M. Wajge, G.V. Reklaitis, An optimal campaign structure for multicomponent batch distillation with reversible reaction, A.I.Ch.E. Annual Meeting, 1995.
[22] S.J. Wright, Stable parallel algorithm for two-point boundary value problems, SIAM J. Sci. Statist. Comput. 13 (1992) 742–764.