Reformulating Ill-Conditioned DAE Optimization Problems

Purt Tanartkit and Lorenz T. Biegler

EDRC 06-205-95

Reformulating Ill-Conditioned DAE Optimization Problems

P. Tanartkit and L. T. Biegler *
Department of Chemical Engineering,
Engineering Design Research Center,
Carnegie Mellon University, Pittsburgh, PA 15213.
email: biegler@cmu.edu

August 31, 1995

Abstract

This paper discusses numerical issues in Differential-Algebraic Equation **(DAE) optimization concerning the stability and** accuracy **of** the **discretized Nonlinear Programming Problems (NIP).** First, a brief **description of the solution strategy based on reduced-Hessian** Successive **Quadratic Programming (rSQP) is described, focusing on** the **decomposition step of the DAE constraints. Next, some** difficulties associated **with unstable DAE problem formulations are exposed** via examples. A **new procedure for detecting ill-conditioning and problem** reformulation **is then presented. Furthermore, some properties of** this procedure as **well as its limitations are also discussed. Numerical** examples are provided, **including a flowsheet optimization problem** with an unstable **reactor.**

1 Introduction

The optimization of a Differential-Algebraic equation (DAE) system has **enjoyed a dramatic increase in research activities, largely** due to its diverse **applications. A wide range of chemical engineering** problems naturally **belong to this class, for example batch processes,** various types of control **problems, and flowsheet optimization, particularly** with reactors and other

*Author to whom all correspondence should be addressed

non-perfect mixing unit operations. These problems pose an interesting challenge because the solution procedure involves several disciplines, i.e., operation research, control, numerical analysis, and linear algebra, to address these problems. **As** the **understanding in these** disciplines improves, **we now have tools to produce a general solution framework.**

In this article, we consider the problem (DAE1)

$$\min_{z(t),y_{(ft)\ll(t)t/P} \quad <p(z(tf)Mtf)Mtf),t_f,p) \tag{1}$$

$$s.t: \quad F(z(t),z(t),y(t)Mt),t,P)=O \tag{2}$$

$$G(z(t),y(t)Mtht,p)<0\_ \tag{3}$$

$$O\#W*.)\text{»»}(*\#).\ll(*t),*j,p)\leq 0 \tag{4}$$

where $<p$ is a scalar objective function,

- F are differential-algebraic equation constraints expressed in an implicit form,
- tf is final time,
- G are algebraic inequality constraints,
- G_B are point condition constraints (e.g., initial or final conditions) at times t_8 including tj,
- z, y are state profile vectors,
- u are control profile vectors, and
- p is time-independent parameter vector.

Typically, these problems are solved by transforming them into nonlinear programming (NLP) problems. Then the solutions are sought by either feasible or infeasible path strategies. Detailed discussion can be found in [14] and [15]. In both approaches, the solution profiles are determined by optimization routines which simultaneously satisfy the DAE constraints. In general, this process can be done as a direct integration of the **DAE** as in the feasible path approach or by including the discretized DAE as constraints in the NLP in the infeasible path approach.

However, these approaches have an implicit assumption that the DAE formulation is stable or well-conditioned; however, many of the applications can have potentially unstable modes. For instance, the determination of the control actions in model predictive control schemes usually involve prediction steps and optimization steps that are not always guaranteed to be stable. Consequently, the goal of this paper is to examine the numerical aspects and to present a modification that can be made to alleviate or improve the conditioning of unstable problems by manipulating the boundary

conditions in the decomposition step. This also raises the possibility of a unified framework for both stable and unstable problems.

2 DAE Optimization

In this section we review some concepts of Successive Quadratic Programming (SQP). To use the optimization tools, the continuous time problem (DAE1) first has to be converted into an NLP. The standard technique proceeds by approximating the profiles by a family of polynomials on finite elements. Therefore the variables in NLP also include all of the coefficients of these polynomials (see [14] for review), while the DAE constraints are replaced by collocation equations. In this work we assume that the sizes of the finite elements are pre-determined and the approximation based on these elements will result in an accurate representation of the problem. The treatment of the relaxation of the assumption can be found in [14]. With this transformation, we have the following problem statement (NLP1):

$$\min_{z_i, \dot{z}_i, y_{ij}, u_{ij}, p, t_f} \Phi(z_i, \dot{z}_i, y_{ij}, u_{ij}, p, t_f)$$

s. t. : Discretized DAE model:

$$Gizi.ii.yij.Uij.p) \leq 0$$

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

point conditions at $t^{\wedge}$:

$$GkizuŻi.yij.Uij.p) \leq 0$$

$$for\ k = 1, \ldots, ns$$

bounds:

$$z^L \leq Zii \leq z^u$$

$$v^L \leq yij \leq y^u$$

$$P^L \leq P \leq P^U$$

$$t^{*1} < tf \leq t^u$$

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

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

In general, problems in chemical engineering are stated with a set of initial conditions. This is due to the fact that the initial conditions of the system are usually known as opposed to the final conditions. This is true in the cases of control problems and design problems. In this paper, we will assume the following problem structure (NLP2) without loss of generality.

$$\min_{z_i, \dot{z}_i, y_{ij}, u_{ij}, p} \Phi(z_i, \dot{z}_i, y_{ij}, u_{ij}, p)$$

s. t. : Discretized DAE model:

$$C(z_i, \dot{z}_i, y_{ij}, u_{ij}, p) = 0$$

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

initial conditions:

$$z(0) = zo$$

bounds:

$$z^L \leq za \leq z^v$$
$$V^L \leq Vij \leq V^U$$
$$u^L \leq tHj \leq u^u$$
$$P^L \leq P \leq P^U$$

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

2.1 Successive Quadratic Programming (SQP)

In [14], we discussed an extension to the infeasible path approach by combining the idea of both feasible and infeasible path approaches. This strategy has been shown to be efficient for the problems with special structure in the constraint Jacobian. By using existing solvers that take advantage of these structures, the reduced-Hessian SQP (rSQP) not only decreases the computational effort and storage but also makes the solution procedure more robust. This robustness is due to the fact that the factorizations are

done locally. Therefore, ill-conditioning in any part of the problem can be immediately located and appropriate modifications can then be made.

Without loss of generality, we can further simplify the problem:

$$\min_x \; f(x) \tag{5}$$

$$s.t. \quad c(s)=0 \tag{6}$$

$$x \in [x^L, x^u) \tag{7}$$

where the inequalities are converted to equalities, with bounded slack variables. Now the quadratic programming subproblem (QP1) corresponding to the above problem:

$$\min_d \quad V^*\{x^k)^T d + l/2 < FB^k d \tag{8}$$

$$s.t. \quad c(x^k) + Vc\{x^k)^T d = 0 \tag{9}$$

$$d \in [x^L - x^k, x^u - x^k) \tag{10}$$

$$x = [\, z, \, i, \, y, \, u, \, p, \, t_f \,]^T \tag{11}$$

With the maturity of sparse matrix solvers, a valid argument can be made for solving QP1 (8-11) with a full space method instead of a reduced space method. However we are convinced that the both approaches complement one another. An important advantage of the full-space approach is that it may require less data storage than reduced-dimension problems. Also, the full space technique works well with problems whose stability is not a major concern. On the other hand, the decomposition approach provides us with more information and tools that allow us to address the stability characteristics of the underlying problem. First, with the detection. we can identify the unstable component(s) in the DAE. For an unstable problem, the corresponding reduced-space QP can be ill-conditioned. As a result, a more stable factorization procedure then can be applied only to the reduced-space QP, once the detection identifies the instability. Finally, the decomposition also leads to a systematic procedure for problem formulation and identification of local singularity. In this paper, we deal mainly with ill-conditioned problems and our approach will be based on the reduced-space method.

2.2 Reduced Hessian SQP

The procedure begins with the partition of variables into dependent *(y)* and independent variables *(z)*. The basis matrices for both spaces are obtained

using a coordinate basis to preserve the sparsity pattern of the Jacobian. The independent variable space occupies the null space of the constraint **Jacobian** ($\mathbf{Vc^T}$) **and** the **dependent variable** ispace involves the corresponding **range** space. Then the search direction can be presented as:

$$d = (Yp_y)^k + (Zp_z)^k \tag{12}$$

where matrix Z has to satisfy:

$$V<?Z = [\ V_y c^T\ \ V_z<?\ )Z = 0. \tag{13}$$

We then choose

$$z = \begin{bmatrix} \cdot & -\nabla_y c^{-T} \nabla_z c^T \\ & I \end{bmatrix} \text{ " } \bullet \text{ HI} \tag{14}$$

with the range space direction (Yp_y) obtained by :

$$(Vc^T Y)p_y = -c(x^{fc}) \tag{15}$$

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

$$\min_p \quad \mathbf{V^*(x^*)^r p} + l/2pl(Z^T BZ)p_2 + l/2(Z^T BYp_y)^T{}_{Pz} \tag{16}$$

$$s.t. \qquad x^k + Yp_y + Zp_2 \text{ € } [\mathbf{x^L}, x^u] \tag{17}$$

Since (QP2) has many inequality constraints, most of which are inactive, a dual-space solution procedure seems to be more effective compared to the primal-space procedure. In this study, we employ the QPKWIK routine [12] which is based on the dual-space approach. Moreover, the computational expense of QPKWIK usually varies only quadratically with respect to the number of independent variables.

In the next section, we will examine the calculation of the dependent variable contribution as well as its numerical characteristics. These analyses will be helpful to understand the issues involved because it reflects the underlying behavior and possibly numerical instability.

2.3 Y space move calculation

In this section, we explore the *Y* space move and consider the reason why it needs attention and formal examination. As discussed above, the determination of this move is equivalent to the Newton step to solve the constraints, in this case, the discretized DAE system.

To approximate the DAE system, we apply collocation on finite elements. Alternately, we can apply a multiple shooting approach to the DAE systems. Both approaches have similar structure and can easily be modified to accommodate a parallel computation architecture [16]. The shooting points in our procedure are finite element locations and this can also be shown to be identical to the finite difference integration scheme. By applying those algorithms, we obtain the following recursive linear system (18), which connects these individual elements, in the space of state variables at the beginning of the elements. Here we assume, without loss of generality, that the boundary conditions are in the format of separate two-point boundary conditions (see [2]). Note that the algebraic equations in the system are projected out, in other words, the index of the DAE system should be less than two (for details on the procedure see [6]). We also assume that the given side conditions are consistent (for details see [11]).

$$\begin{bmatrix} Ba & & & & & \\ Si & -I & & & & \\ & S_2 & -/ & & & \\ & & \ddots & \ddots & & \\ & & & S_k & -I & \\ & & & & B_b & \end{bmatrix} \begin{bmatrix} \Delta s_1 \\ \Delta s_2 \\ \text{AS3} \\ \vdots \\ \Delta s_{ne} \\ \Delta s_{ne+1} \end{bmatrix} = \begin{bmatrix} da \\ d_{p1} \\ d_{p2} \\ \vdots \\ dpnc \\ d_b \end{bmatrix} \tag{18}$$

where B_a is the Jacobian of the initial conditions,
B_b is the Jacobian of the end conditions,
Δs_i are the dependent variable moves, or Newton steps. at the beginning of element i,
$\text{As}_{nc}+i$ are the dependent variables moves, or Newton steps, at the end of time horizon,
d_{p_i} is the constraint violation in element i,
d_a is the constraint violation in the initial conditions.
d_b is the constraint violation in the end conditions,
Si is the matrix resulting from multiple shooting in clement i.

The solution of system (18) then will be added to the independent variable move (Zp_z) to form the overall search direction. Because the number of dependent variables is normally larger than that of independent variables, this factorization is usually the bottleneck of the algorithm. As a result, the success or failure of the algorithm relies heavily on the efficiency and robustness of the decomposition.

Our first task is then to investigate if the particular problem is well-conditioned. The robustness or stability of the linear system is measured by the *condition number.* If the condition number is large, the matrix is said to be an ill-conditioned matrix. In other words, any small perturbation of the right hand side (rhs.) or in the matrix itself will lead to a large change in the solution. For linear systems, we can naturally group problems into either well-conditioned or ill-conditioned problems. For well-conditioned problems, several algorithms have been developed and efficiently implemented. For example, Wright [16] described a general algorithm for a parallel computing architecture using cyclic reduction. On the other hand, modification will have to be made to be able to use the existing computational tools for the unstable cases. This will be the main focus of the rest of the paper.

As expected, system (18) and the original DAE system are closely related, and the conditioning of the linear system depends not only on the discretization scheme used but also on the conditioning of the original DAE. For review of the conditioning of continuous DAE's , see Chapter 3 in [2]. If the original problem formulation is ill-conditioned, this linear system will also be ill-conditioned and this *cannot be* overcome by just applying new discretization schemes. To illustrate this concept, consider an initial value formulation given as:

$$\begin{bmatrix} I & & & & \\ Si & -/ & & & \\ & S_2 & -I & & \\ & & \ddots & \ddots & \\ & & & S_k & -I \end{bmatrix} \begin{bmatrix} \Delta s_1 \\ AS2 \\ \Delta s_3 \\ \vdots \\ \Delta s_{nc+1} \end{bmatrix} = \begin{bmatrix} d_a \\ d_{p1} \\ d_{p2} \\ \vdots \\ d_{p4} \end{bmatrix} \tag{19}$$

with the following system [5} that will be used to develop our algorithm in the article:

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = A \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} g(t) \\ 0 \end{bmatrix} \tag{20}$$

$$= \begin{bmatrix} \tau^2 z_2 + g(t) \\ z_1 \end{bmatrix} \tag{21}$$

with:

$$g\{t) = (\pi^2 - \tau^2)\sin(\pi t) \tag{22}$$

This system yields a general analytical solution:

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} sin\{nt) + c\backslash \exp(—rt) + 02 \exp(\mathrm{rt}) \\ ir \cos(*\mathrm{rt}) - c\backslash \exp(-\mathrm{rt}) + c* exp(rt) \end{bmatrix} \quad (23)$$

The constants (ci, c_2) are determined by the side conditions. The eigenvalues (poles) of this problem are r, - r, and that makes the problem unstable and ill-conditioned regardless of the values of ci, c_2 and function *g(t).* When we calculate the condition number of the resulting matrix (19), it increases exponentially with respect to number of finite elements and r as shown in Figure 1. Since the terms *Si* are identical to e^{A6t}, where *A* is the matrix in (20) and *St* is the step size, and since one of the eigenvalues of the problem is positive, the error in the problem will be amplified by e^{*61} at every time step in the forward direction, regardless of the non-homogeneous rhs function *g(t)* and the final results. If Gaussian elimination is applied with block row partial pivoting, the pivoting elements will only be the submatrix on the diagonal. As a result, the factorization routine will decompose* this matrix without any warning of ill-conditioning. With the initial value problem formulation, this factorization is analogous to the forward integration and this results in a growth of errors.

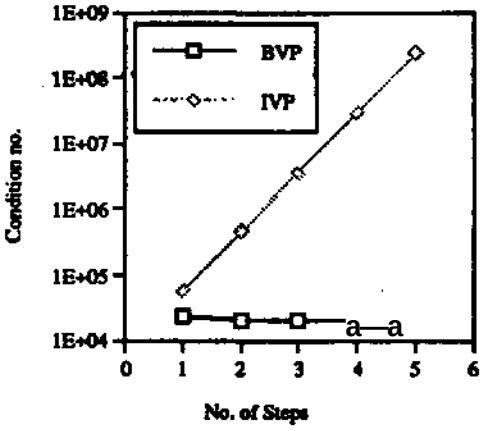

Figure 1: Condition number of the example.

Consequently, the *Y* space move is corrupted and this leads to an unreliable search direction. This phenomenon is not restricted to initial value problems. As described in [2], for well-posed problems, the initial conditions need to control the decreasing modes, while the end conditions need

to control the increasing modes, in order to get a stable decomposition.

Consider now a parameter estimation problem based on this dynamic 8jrstem where we estimate the parameter $p = ir$ given 30 measured data with T = 60. The detailed IVP formulation is given in (24-28). With the side conditions, the analytical solution of this problem is given by cj, $ci = 0$

$$\min_p \; R = \sum_{i=1}^{N} (z_m(i) - z(i))^2 \tag{24}$$

$$s.t. : \quad \text{£\ ss } r^2 z_2 - (r^2 + p^2)\sin(pt) \tag{25}$$

$$\dot{z}_2 = z\backslash \tag{26}$$

$$*i(0)-0 \; ; \; Z*(0)= \pi \tag{27}$$

$$-10 \leq z_u z_2 \leq 10 \tag{28}$$

As seen in Figure 2, the optimization fails to converge even when the parameter and profiles are initialized to the analytical solution, correct up to 9 digits. This is due to an unbounded growth of the round-off error in the unstable forward mode. However, if we reformulate the problem as a bound-

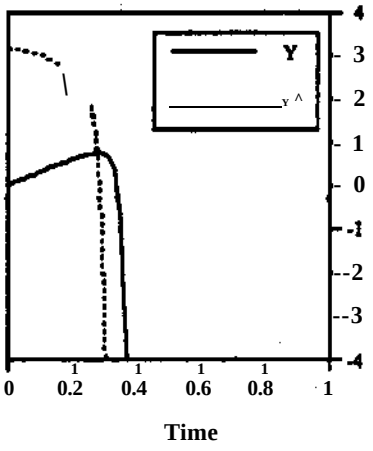

Figure 2: Result profile for (IVP).

ary value problem with (29) instead of (27), the condition nmnbors of the constraint matrix change dramatically. The condition number of the constraint Jacobian remains almost constant for increasing time intervals (see Figure 1) and varies linearly with respect to r.

$$B_a z(0) = z_2(0) = \pi$$
$$B_b z(1) = z_1(1) = 0 \tag{29}$$

If we construct the previous example using the boundary condition in (29), the problem (BVP) is given **as:**

$$\min_p \tag{30}$$

$$a.t.: \quad i\backslash = \mathbf{T^{2}*_{2}} - (T^2 + \mathbf{p^2}) \sin(pt) \tag{31}$$

$$\dot{z_2} = z_1 \tag{32}$$

$$z_2(0) = \pi \; ; \; z_1(1) = 0 \tag{33}$$

$$-10 \leq z_1, z_2 \leq 10 \tag{34}$$

Note **that** both **(IVP) and (BVP) have the same analytical** solution. The (BVP) problem converges to the optimal solution **and** the result is given in Figure 3 as a result of a well-conditioned constraint matrix.

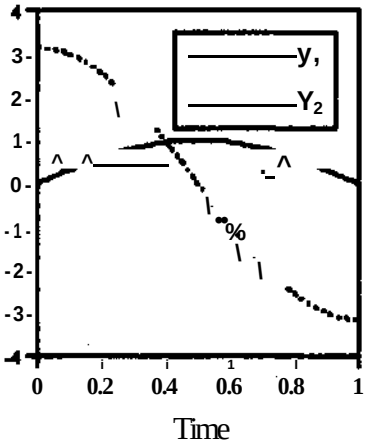

Figure 3: Result profile for (BVP).

As seen, poor performance of optimization is a direct result of the problem formulation. In this section we demonstrated that the problem formulation, specifically the boundary conditions, does have a strong impact on the ability to solve (DAE1). Briefly stated, compatible side conditions must be specified such that the corresponding mode is not an increasing mode starting from these conditions. However, for nonlinear problems, we do **not** know this a priori and it can also change during the optimization process due to changes in parameters and control profiles.

This is also important in the context of general model predictive control, which requires an open loop behavior of the models with initial conditions.

In absence of a stabilizing input, these models can also exhibit unstable behavior. This instability can also occur during the solution procedure of the control determination. Hence, to be able to address dynamic optimization systematically, it is vital that the algorithm must handle both stable and unstable problems in a uniform and consistent fashion.

In the next section we discuss the detection of unstable modes and the substitution of compatible boundary conditions to stabilize them. These boundary conditions are imposed as optimization variables and allow a stable decomposition of the dependent variables (Y space move).

3 Problem Detection

In order to detect potentially unstable problem behavior, there are several alternative procedures that can be selected, ranging from calculation of the condition number of the problem directly, calculation of the growth of the condition number with respect to a sequential decomposition, or calculating the fundamental modes of the problem explicitly. Howrvor. because the procedure has to be done at every intermediate step during the optimization process, we need to develop a procedure that requin s much lass computational effort.

As seen in the previous example, the condition of the problem can be characterized as sensitivities of the final profiles to given sido conditions. A direct application of this observation is then to detect the growth in the sensitivities of the profiles with respect to the side conditions. An obvious alternative is simply to trace all the sensitivities of the profiles with respect to the boundary conditions. This task can be accomplished by replacing the rhs of the factorized matrix of the first and the last blocks with identity matrices for both the initial and boundary conditions. For a stable* problem, the sensitivities must be small with respect to the boundary conditions.

$$\begin{bmatrix} B_a & & & & \\ Si & -J & & & \\ & S_2 & -I & & \\ & & \ddots & \ddots & \\ & & & & B_b \end{bmatrix} \begin{bmatrix} ds_1/dB_a & ds_x/dB_b \\ ds_2/dB_a & ds_2/dB_b \\ ds_3/dB_a & dsz/dBf, \\ \vdots & \vdots \\ ds_{ne+1}/dB_a & ds_{ne+1}/dB_b \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & \end{bmatrix} \quad (35)$$

Recall the example given earlier, the sensitivities of the IVP formulation are extremely large ($\geq 10^{10}$ or c^T). As a result, the formulation cannot be used in the optimization. In contrast, an appropriate BVP decomposition

yields the sensitivities that decrease as the number of elements is increased and the biggest component is reasonably small.

The computational expense of this procedure is reasonably small because the matrix has already been decomposed; therefore, it is equivalent to adding multiple rhs. Because there is no need to enforce continuity of the algebraic profiles, the procedure is applied to only the differential part. In short, this procedure is recommended for first iteration or after any iteration with large step in parameters and control variables. In the next section we will discuss some of the heuristic alternatives that are less expensive. They can provide a useful option for iterations where we have gained fairly good knowledge about the problem.

3.1 Problem Screening

To further reduce the computational time, a preliminary heuristic screening of the matrix can be done by considering only the first block or a single series of blocks using the following assumption:

- The ODE has an exponential dichotomy, with / decreasing fundamental modes. At each stage *i* of the elimination algorithm exactly *n - I* rows are exchanged by row pivoting, between the *i*-th block row and the (*i* + l)-th block row. Moreover, the collection of n - / rows that is swapped out of the first block row is passed down, intact, through block row $i = 2,3,\ldots,ne$ [16].

With this assumption, we assume that the fundamental modes of the problem do not change along the time horizon and the same pivoting sequences are used for each block in the problem. Using this assumption, we can consider only one block, or a set of blocks per iteration. Furthermore, the increasing modes in the problem will be seen in terms of the growth in the solution, in other words, some of the components are amplified from block to block. As a result, the main idea of the screening scheme is to recognize this growth. In particular, if the sensitivities of components with respect to the previous parts are greater than 1, the problem has unstable modes.

The implication of the assumption is that the behaviors of the profiles remain the same throughout the horizon. Any increasing mode in the problem can be detected by only considering the first block or a fixed set of blocks. If all the sensitivities of the variables with respect to the side conditions

(both initial and end conditions) remain reasonably small, the problem is well-conditioned.

Apart from the sensitivity approach, another alternative which can be employed for linear-time-invariant (LTI) systems is to determine the singular values of the transition matrix by consider the following solution to the dynamic system.

$$s_{i+1} = \Phi_i s_i + Gi, \quad or \tag{36}$$

$$= e^{A\delta t} 8i + Gi \tag{37}$$

where \$ is the transition matrix,

G{ is the inhomogeneous part involving controls, parameters and forcing functions.

Here, the singular value decomposition (SVD) of \$ is

$$* = U^T TV \tag{38}$$

where (7, *V* are orthogonal matrices,

E isadiag(<7i,<7_2,...,(7_n)

a are the singular values of \$ with

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0 \tag{39}$$

Note that the singular values are simply the eigenvalues (A) of $(<f>^T\$)^{1/2}$. Therefore, an LTI problem, or for slowly varying LTV systems [8], with decreasing modes will have a,'s which are less than one. In the same fashion. the singular values that are larger than one correspond to the increasing modes in the problem. However, this approach still depends on where to draw the line between increasing and decreasing modes as in the sensitivity approach. Moreover, the approach can be used only to find the number of increasing and decreasing modes and then we can compare it with the given side conditions. It is not able to inform us whether the problem is well-conditioned with a particular boundary structure. Another serious drawback of this procedure is that it is expensive due to the cost of SVD. On the other hand, the sensitivity procedure uses the already-factorized matrix with different right hand sides. Moreover, if *A* is strictly diagonally dominant, for an LTI or LTV system, the modes in the problem coincide with the state variables [2]. Consequently, the problem reformulation is also easy and we will discuss about this issue in the next section.

To demonstrate the procedures with **both ideas, recall** the example given **earlier** (21). The **resulting** L, **from** LU **decomposition, factor** from the 1VP **formulation with r = 60 is:**

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -4.07 & -.065 & 1 & 0 \\ -236 & -4.07 & 0 & 1 \end{bmatrix} \begin{bmatrix} ds_{1,1}/ds_{1,1} & ds_{1,1}/ds_{1,2} \\ ds_{2,1}/ds_{1,1} & ds_{2,1}/ds_{1,2} \\ ds_{1,2}/ds_{1,1} & ds_{1,2}/ds_{1,2} \\ ds_{2,2}/ds_{1,1} & ds_{2,2}/ds_{1,2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad (40)$$

In this example, the U matrix is identity. From the system above, for instance, cki,2/<ki,i is ~ 4. As a result, the error from the factorization can potentially increase by **a** factor of 4 as the number of blocks or dements increases. As in the example, with 30 elements the error will grow to 4^{M} at the end of the last element while the correct profile is *cos(x).* On the other hand, similar analysis can be done with the boundary value version of the problem (29). The corresponding amplifying factor, as expected, is < 1, and the error will damp out along the profiles. Because this problem is an initial value problem, the procedure is completed. However, if problems also contain end conditions, then the same sensitivity procedure will have to be done for them.

Similar analysis can also be done using the SVD procedure. In this problem, the singular values of the transition matrix are 236 and .0052. Hence this problem contains both increasing and decreasing modes as indicated by the value of the singular values. As a result, this problem needs both initial and end conditions. In the next section we will discuss how to determine the new set of boundary conditions

First, we note that these procedures are valid for linear and nonlinear systems that satisfy the assumption of time invariant dichotomy. This assumption holds for many BVP's in process engineering and is satisfied by all the examples in this work. Furthermore, this procedure can be omitted if the behavior of the system does not change from iteration to iteration. Finally, the tolerance of the,test for ill-conditioning plays an important role whether to stop using the LU factors. A general observation can be made that the tolerance is based also on the number of blocks in the problem. Recall that the amplification of the error also depends on the number of finite elements. As a result, for a long horizon problem this amplification should be small. From our experiments, for a problem with 30 finite elements, this acceptable growth factor should be less than 1.5 in each element.

4 Problem Reformulation

In the previous section, we investigated the procedure to determine if the problem formulation has to be altered. The proposed algorithm assumes that the problem can be reformulated stably using separated and uncoupled boundary conditions.

The main focus of this section is to stably invert the Jacobian of the discretized DAE which is the same as the matrix in (35) Our algorithm starts by performing Gaussian elimination with total pivoting on the submatrix *S\ (nxn),* and this step stops when the next available pivot elements are small. Let r be the niunber of Gaussian pivoting steps taken, then initial conditions are added for the remaining *n* — *r* columns as identity matrix. Then the elimination process is done for the next block sequentially by adding the unused rows (n — r) from the previous block. The factorization of the rest of the problem is done via row partial pivoting to preserve the structure of the problem. The final step of the procedure is then to add end conditions to the last block. For the last block *(n — r* rows), the elimination is done via total pivoting similar to the first block, the end conditions are then added as an identity *(rxr)* matrix.

To illustrate the procedure, recall the system in (40).

$$5\,l = \begin{bmatrix} -4.07 & -.065 \\ -236 & -4.07 \end{bmatrix} \qquad \text{(41)}$$

The first pivot element in this example is —236, and after the first step the only available pivot element (in the "1,2" position) is .0052 which <^ 1. At this point, the number of the elimination step on Si is r (1). To proceed to the next column, the pivoting elements required are added as initial conditions *n* — *r* (2-1), *z%* in this example. The same procedure is also performed for the last block to obtain the necessary end condition(s). When we apply the procedure to the example, it turns out that we need to add a side condition (45) to the BVP problem and enforce the constraint on *z\(0)* in the optimization problem. The formulation after the modification is given as

$$\min_p \; R = \pounds\&i(*m(0 - z(\dot{t}))^2 \qquad (42)$$

$$s.t: \quad z\backslash = T^2 Z_2 - (T^2 + jfi)\sin(pt) \qquad (43)$$

$$\dot{z_2} - *i \qquad (44)$$

$$z_2(0) = 7r; z_1(l) = C \qquad (45)$$

$$z_1(0) = 0 \tag{46}$$

$$-10 \leq z_1, z_2 \leq 10 \tag{47}$$

The constraints to be decomposed are (43-45), and the extra constraint (46) is added to the optimization problem. The boundary conditions generated by the procedure are structured so that they can be examined separately. Furthermore, Wright [16] proved stability for a similar decomposition for parallel computation for a two-point boundary value problem that can easily be modified for our procedure. This reformulation results in accurate function and gradient evaluations. Moreover, for a linear system the boundary conditions derived using this procedure can easily be verified with the following property [2].

Property 1 *For LTI systems, a set of separate boundary conditions leads to well-conditioned decomposition iff the matrices $B_a W_\omega$ and $BtW^\wedge$ are not singular.*

where W_ω, Wt are the eigenvectors of A for the decreasing and increasing modes respectively. Because the modes and the eigenvectors of LTI remain the same throughout the horizon, the above statement is not only necessary but also sufficient. A proof can be found in the Appendix. This necessary and sufficient condition comes directly from the fact that the decoupling is exact and we can solve the decreasing and increasing parts separately. For detailed analyses on general cases, see [2]. The property above also applies to a system that is strictly diagonally dominant uniformly. In this case each mode is restricted to only one profile. As a result, the dichotomy is obvious and the reformulation step yields the desired side conditions.

Another important characteristic of the problem is that the compatible sets of boundary conditions may not be unique. As mentioned earlier, the necessary condition requires that these conditions are compatible with the proper eigenvectors of the problem. Recall the previous example, the eigenvectors of the problem are:

$$W = \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \tag{48}$$

The eigenvalues of this problem are 60, —60 (with r = 60), consequently, the eigenvector for the decreasing and increasing modes are $[1,1]^T$ and [—1,1] respectively. With the new boundary conditions (45):

$$B_a W_a^T = [0\ 1][1\ 1]^T \tag{49}$$

$$B_b W_b^T = \begin{matrix} * \\ = \\ \# \end{matrix} \begin{matrix} \mathbf{O} \\ [1\ 0][-1\ 1]^T \\ 0 \end{matrix} \tag{50}$$

Consider now the structure of the constraints with the conventional IVP formulation. First of all, the IVP structure with increasing modes will lead to an exponential increase of the error as seen in the previous section. However, with the reformulation, the pivot sequence in the LU factorization leads to a more stable decomposition. For example, in the case where we have dichotomy in the state variables and the structure is given by:

$$\Psi = \begin{bmatrix} Si & -I & & \\ & S_2 & -I & \\ & & S_3 & -I \end{bmatrix} \tag{51}$$

which is permuted to:

$$= \begin{bmatrix} E & C & If & 0 & & & & \\ B & A & 0 & I_b & & & & \\ & & E & C & 1/ & & & \\ & & B & A & & 0I_h & & \\ & & & & E & C & If & \\ & & & & B & A & 0 & I_b \end{bmatrix} \tag{52}$$

where E **is a submatrix with small components,**
A **is a submatrix that can be factored stably (via total pivoting), and**
B, C **are off-diagonal non pivot matrix elements.**

Then the pivot elements are always from A and // and they are the largest elements in the matrix. By setting B_a to // and B_b to I_b, we have therefore found a stable pivot sequence from the structure of S\. However, the partition in (52) is only one case and other pivoting cases can arise, which are still preferable to the IVP form. These are discussed in Appendix B.

4.1 Enforcement of the Remaining Initial Conditions

After the detection and the reformulation of the constraints, the rest of the initial conditions that are not involved in the calculation of p_y are moved to the QP. This procedure also enables us to address classes of problems

in model predictive control that are infeasible or unbounded as we cannot enforce the required inequality constraints. To solve the problem, we need to further alter the structure of the problem. Instead of implementing (46) as a hard constraint, one can augment it to the objective function as penalty term. With this transformation, if the problem cannot be satisfied, it will result in offset in the side condition (46) in order to yield a feasible solution. This strategy is motivated by the work of Oliveira and Biegler [10] in the context of model predictive control.

They suggest the used of quadratic($/_2$) and exact *(l\)* penalty functions with penalty parameter *p*. This quadratic strategy is known to be equivalent to the original hard constraint problem when the penalty parameter $p \to \infty$ In contrast, the exact penalty formulation only needs *p* to be greater than the Lagrange multipliers of the original constraints. By using the exact penalty function, *p* remains bounded to maintain feasibility (unlike the quadratic penalty). As discussed in [10], the penalty parameter *(p)* has only to satisfy:

$$P \geq M_{oo} \tag{53}$$

Where *v* is the vector of Lagrange multipliers of the original problem.

To illustrate the result of the procedure, we recall the general problem (NLP2). For the problem with unbounded components, the reformulation procedure will generate a new set of boundary conditions to meet the stability and boundedness of all profiles. As a result, the revised problem statement can be given as (with *l* penalty):

$$\min_{z_i, \dot{z}_i,} \quad *(z\text{»}, i\text{>}, y\#_f m\text{^}p) + p\{t_t + \epsilon_{tt}) \tag{54}$$

s. t. : Discretized DAE model:

$$\left.\begin{array}{rcl} C(z_i, \dot{z}_i, y_{ij}, u_{ij}, p) & = & 0 \\ z_a(0) & = & z_{a0} \\ z_b(1) & = & \xi \end{array}\right\} \text{to be decomposed for } Y \text{ space move} \tag{55}$$

$$z_b(0) = z_{b0} + ei\text{-}tu \tag{56}$$

bounds:

$$\text{^} < Zij < z^U \tag{57}$$

$$V^L \leq Vij \leq y^U \tag{58}$$

$$u^L \leq iitf \leq u^{1f} \tag{59}$$

$$P\text{^} \leq P < P^U \tag{60}$$

$$\epsilon_l, \epsilon_u \geq 0 \tag{61}$$

$$far\ i = 1,\ldots,ne;\ j = 1,\ldots,nccl$$

where the components (a, 6) are determined in the reformulation step. In addition, this procedure can also be applied when the underlying problem, with given structure of side conditions, grows exponentially. This can also occur if we have side conditions which lead to unbounded solution. For example, we consider the example again, but this time with initial conditions that lead to an unbounded solution.

$$B_a z(0) : zi(0) = .1, \tag{62}$$

The analytical solution of the problem with (62) is given by (23), and, ci, c_2 are not equal to zero. Hence the profiles will grow exponentially and violate bounds on the state variables. The problem then becomes infeasible because the profiles increase exponentially and the side condition (46) cannot be satisfied.

Consider the example (30-34) again. With incorrect side conditions, the problem has an unbounded solution. We then incorporate some or all infeasible side conditions (46) into the objective function either as quadratic penalty or an exact penalty term.

Note that the decomposition in the calculation for p_y (see (12)) is the system (64-66).

$$\min_p \quad R = \pounds \text{i}\text{L}\text{i}(\text{«m}(\text{O} - z\{i\})^2 + p''Zi\text{«»} \tag{63}$$

$$a.t.: \qquad 'zi = r^2 z_2 - (\text{T}^2 + p^2)\sin(\text{pt}) \tag{64}$$

$$\text{JZ2} == \text{Z}\backslash \tag{65}$$

$$z_2(0) = \pi \ ; \ z_1(1) = \xi \tag{66}$$

$$\text{zi(O)-.l} = \text{cj-}\epsilon_u \tag{67}$$

$$ti \gtrsim 0 \tag{68}$$

$$-10 \leq \text{zi,S2} \leq 10 \tag{69}$$

The profiles for the penalty formulation in this study with $p = 50$ are bounded with an offset in the initial condition of $z\backslash$ as expected. With this formulation we are therefore assured that the NLP converges to a bounded solution.

We conclude this section with some remarks. Firstly, as stated in the assumption, we assume that the pivoting sequences in all blocks are the same. However, for some instances, the pivoting sequence during the factorization can change. From our experience if we formulate the problem so that the

number of initial conditions is equal to the decreasing modes, the sequence usually drifts towards a compatible set. Secondly, the reformulation procedure is to ensure that the function and gradient evaluation as well as the calculation of Y space move are accurate. Since (has very little effect in the projected constraint (67), this can cause ill-conditioning in the QP and can also lead to a large Z space move. For example, since the side conditions (67) in the QP are not satisfied, the search direction will adjust the stabilizing variables to assure feasibility. However, because the gradients with respect to the side conditions in the decomposition step are small, this can yield a large move in the stabilizing variables. Hence the QP (16-17) must be solved more accurately. Fortunately, effect of controls and parameters often stabilize the QP and we also employ a robust QP factorization.

5 General Algorithm

Armed with the problem detection and corresponding modification scheme from previous sections, we can now outline the main algorithm for the general dynamic optimization problem.

Step 1: *Preprocessing*

- Choose convergence tolerances for the problems.
- Choose a starting point.
- Set up positive-definite matrices for problem Hessians, possibly an identity matrix.

Step 2: *Decomposition*

- Determine the LU factor for the collocation equation system using a Newton-based solver. In this work we employ the COLDAE routine [4], which can be obtained from NETLIB (netlib@ornl.gov).
- Determine if the LU factor is stable by applying the procedure in section 3.
- If the problem is unstable, reformulate the problem using the procedure in section 4. Also reset the Hessian to identity matrix or add row(s) and column(s) for the stabilizing variable(s) with positive diagonal component(s).

Step 3: ***Optimization***

- Solve the reduced QP subproblem (QP1) to get search direction *d*
- If the convergence tolerance is reached, stop.
- Calculate the penalty parameter for the collocation equation using the multiplier-free approach (see [13] for details).

Step 4: ***Line search***

- Perform a line search using the *watchdog* technique [7] that requires a reduction in the penalty function every *t* iterations. The function and gradient evaluations are done using Step 2.
- Update the reduced Hessian using the BFGS formula. Goto Step 3.

6 Examples

The algorithm has been implemented on a DECstation 5000, and all the CPU times reported are in seconds. The tolerance for optimality in all examples is 10^{-7}. To illustrate the algorithm, we consider the following examples.

6.1 Example 1

To illustrate the procedure, we will start with a small parameter estimation problem. This example is modified from [9]. The problem contains 2 differential conditions with 2 initial conditions. The number and the locations of finite elements and sampling data points coincide and are uniform with the final time at *n*.

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} 1-\psi\cos 2t & 1+\psi\sin 2t \\ -1+\psi\sin 2t & 1+\psi\cos 2t \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + f(t) \qquad (70)$$

With the following initial conditions:

$$*i(0) = 1 \qquad (71)$$

$$z_2(0) = 1 \qquad (72)$$

where $f(t)$ is chosen such that the solution is $z(t) = [\exp(<),\exp(t)]^r$. As seen in the fundamental matrix (73), the problem has two modes with one

Tfeble 1: Computational statistics for Example 1.

Number of iterations	CPU time (s)	a	Obj (residual)
6	2.9	0.00	0.
7	3.0	0.05	0.150
7	3.0	0.10	0.602

increasing and one decreasing. If the problem is set up as initial value problem, it will result in an ill-conditioned constraint matrix. However, our detection procedure is able to spot the problem, and subsequently the problem is automatically reformulated and solved.

$$\Theta(t) = \begin{bmatrix} \text{sint} & \text{—cost} \\ \text{cost} & \text{sint} \end{bmatrix} diag(e^{20t}, e^{-18t}) \tag{73}$$

The side conditions generated are $z\backslash(0) = 1$ and $2\hat{}(0) = £$. We report the results on three cases (three different sets of data) in Table 1.

6.2 Example 2

This example is modified from [16]. This problem consists of five differential equations, and the objective of the example is to estimate the parameter (t/>j = 1000) given 30 measure data sets corrupted with random noise $(N(0,a))$ of $z_u z_2$.

$$\dot{z} = Az + f(t) \tag{74}$$

where we define:

$$\begin{aligned} \dot{z} &= [\dot{z}_1, \dot{z}_2, \dot{z}_3, \dot{z}_4, \dot{z}_5]^T \\ z &= [z_1, z_2, z_3, z_4, z_5]^T \end{aligned} \tag{75}$$

and A as

$$\begin{bmatrix} -\psi_1 \text{ coe2u>i£} & 0 & U\text{i} + 0\text{isin2u>it} & 0 & 0 \\ 0 & \text{—02 C06 2u)?t} & 0 & \text{W2+V>2sin2coit} & 0 \\ -\omega_1 + 0\text{isin2u;it} & 0 & 0\text{icos2u>it} & 0 & 0 \\ 0 & \text{'™"d\2 ™f' f^2 Sin 2kJ}_0\text{t} & 0 & \text{02 COS 2u>2<} & 0 \\ 0 & 0 & 0 & 0 & 03 \end{bmatrix} \tag{76}$$

With the following initial conditions:

$$\text{«(0)} = [1,1,1,1,1]^{T} \tag{77}$$

The nonhomogeneous part *f(t)* [16] is chosen so that the analytical solution is:

$$\text{z(t)} = [e^{t}, e\backslash e^{t}, e\backslash e^{t}]^{T} \tag{78}$$

The fundamental matrix of the problem is given in (81). As seen, the fundamental modes, increasing and decreasing modes, must vanish in order to get (78). As a result, the problem is ill-conditioned and cannot be solved with the original formulation. The detection step reveals that the problem is unstable, and stabilizing variables are added as well as end conditions similar to (42-47). Prom the analysis, we found that there are two decreasing modes and three increasing modes in the fundamental solution. With the stabilizing procedure, the boundary conditions in the decomposition are:

$$\begin{aligned} zz(0) &= 1 \\ *4(0) &= 1 \\ z_1(1) &= \xi_1 \\ z_2(1) &= \xi_2 \\ z_5(1) &= \xi_3 \end{aligned} \tag{79}$$

As a result, the constraints to be added to the QP are:

$$\begin{aligned} z_1(0) &= 1 \\ z_2(0) &= 1 \\ z_5(0) &= 1 \end{aligned} \tag{80}$$

Table 2 shows the computational time and iterations required to obtain the solution to this problem with the reformulation.

$$\Theta(t) = \begin{bmatrix} cos\,u)\backslash t & 0 & \sin u;it & 0 & 0 \\ 0 & \text{cosu}\text{\^{}}t & 0 & \text{sina}\text{\^{}}t & 0 \\ -\text{sinu}>it & 0 & \cos a;it & 0 & 0 \\ 0 & -\text{sina}>_2 t & 0 & \text{cosu}\text{\^{}}t & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} e^{\nu' t} \tag{81}$$

where $e^{*} = diag(e^{-\psi_1 t}, e^{-\psi_2 t}, e^{\psi_1 t}, e^{\psi_2 t}, e^{\psi_3 t})$

Table 2: Computational statistics for Example 2.

Number of iterations	CPU time	*a*	**Obj (residual)**
13	9.0	**0.00**	**0.**
14	9.1	**0.05**	**0.32**
14	9.2	**0.10**	**1.28**

6.3 Example 3

Consider a plug flow reactor (PFR) of length L with diffusion in the direction **of** the **flow,** ftg?iiTnii^gno radial mixing. The reactor is depicted in Figure 4. The series reactions in this problem are two first order irreversible! chemical reactions from reactant A to intermediate B and finally to product C. The reaction rate constants^, fcg) are to be determined by 30 measured data sets of the concentrations of the product along the length. The dynamics

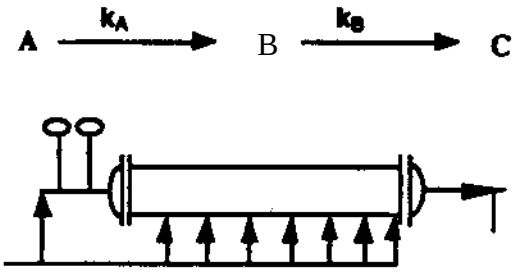

Figure 4: PFR reactor in Example 3.

of this system is governed by a second order differential equation (82-83), the side stream in this example is given such that the concentration profiles are constant.

$$D_A\frac{d^2y_A}{dx^2} - u\frac{dy_A}{dx} - k_Ay_A = f(x)$$
$$D_B\frac{d^2y_B}{dx^2} - u\frac{dy_B}{dx} + k_Ay_A - k_By_B = 0 \qquad (82)$$

with initial conditions:

$$y_A(0) = y_{A0} + e_1$$

$$dy_A/dx = dy_M/dx + e_2$$
$$VB(O) = VB0 + C3$$
$$dysjdx = dyBo/dx + e_A \quad (83)$$

where Z? is the diffusivity constant,
ȳo is the given initial concentration of i,
I/io are the correct initial conditions of t,
e is the error introduced in the initial conditions,
u is the bulk flowrate,
$k\{$ is the rate of reaction,
$f(x)$ is the expression for side streams.

The general solution of this problem contains two increasing fundamental modes and two decreasing modes for $h/D > u/D$, namely exp $(\pm xk/D)$. Hence, the problem has unstable modes and when we solve this problem as an IVP, the optimization routine fails to converge. When we apply our algorithm, the matrix is indeed ill-conditioned. However, even when the stabilizing variables are added, the problem is still infeasible. This suggests to us that the side conditions given are unstable and the problem is then modified using $l\backslash$ penalty (84).

$$\min_{p,e,\xi} R = \sum_{i=1}^{N} (y_m(i) - y(i))^2 + \rho \sum_{i=}^{8} i\text{«}i \quad (84)$$

s.t.: $Eq.82$ (85)

$$VA(0) = VA + ei\text{-}e_2; \quad dz_A\{l)/dx = ti \quad (86)$$
$$VB(0) = \bar{V}BO + €3 - e_4; \quad dz_B(l)/dx = \& \quad (87)$$
$$dyA(0)/dx = \bar{d}y_{A0}/dx + c_5 - ce \quad (88)$$
$$dy_B\{0)/dx = \bar{d}y_{B0}/dx + c_7 - \text{«}8 \quad (89),$$
$$\text{«}i \geq 0 \quad (90)$$
$$-10 \leq z x, \text{«}B \leq 10 \quad (91)$$

Again, the stabilizing variables (£) are added to the formulation and the decomposition is done with the linearized from (82),(86) and (87). The computational results are shown in Table 3. The solutions have offset in the initial conditions in all cases. However, if the accurate initial conditions are given (ej = 0), then there is no offset in the initial conditions.

Table 3: Computational statistics for Example 3.

Number of iterations	CPU time (8)	a
7	3.9	0.00
7	3.9	0.05
7	3.9	0.10

7 Application in Flowsheet Optimization

We have seen from the previous discussion that by using the above algorithm, optimization problems with DAE constraints can be stably and systematically addressed. In this section, we will incorporate it into the context of flowsheet optimization. We will briefly describe the framework based on the concept of tailoring the optimization to exploit the problem structure and make use of existing model solvers.

It is well known that the computational expense of the problem is dominated by the time required to generate and factorize the flowsheet equations. Furthermore, the total number of variables is usually large, however, the majority of them are dependent. Moreover, these variables can be decomposed outside the optimization via the flowsheet constraints. By taking into account these two observations, we consider the use of rSQP. A depiction of the approach is given in Figure 5. Some significant advantages of the approach from a practical point of view are that:

- Model sparsity and structure of the unit systems are completely preserved.
- Complicated units tend to be very demanding in terms of initialization and formulation. By using existing model subroutines, they provide a well-tested procedure and they also take into account model structure. This will lead to robust process modeling.
- The flexibility of reformulation and problem construction can he achieved in a straightforward fashion.
- The procedure retains the one-step superlinear convergence properties as in equation-based optimization.

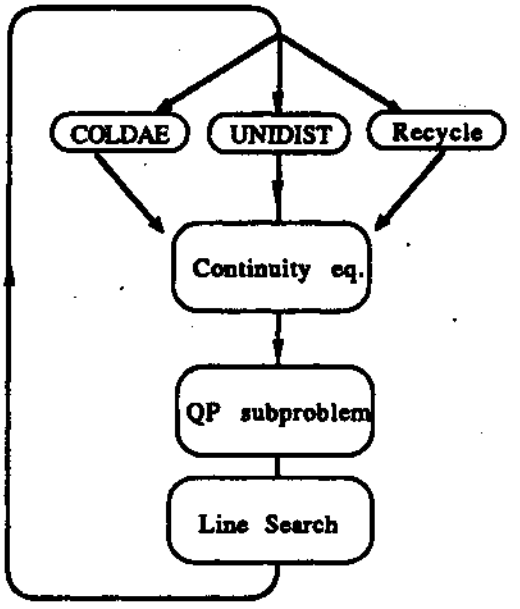

Figure 5: Reduced Hessian SQP algorithm.

The rSQP approach can be described as an iterative refinement of the operating condition variables (x_k) by taking a step a* along the search direction *dk,* until the optimal solution is reached. The step size is determined such that a merit function (exact penalty function in this work) is sufficiently reduced. The search direction d* is the solution to the rSQP. It should be noted that these unit model solvers in the procedure are general, and the modifications to be made to these routines are minimal.

7.1 Result for Flowsheet optimization with unstable reactor

The approach described above is tested with the optimization problem of the flowsheet in Figure 6. The objective function of the problem is to maximize the product production, given a fixed amount of reactant. The problem is also subject to bounds on operating conditions. This process is based on the Hydrodealkylation process. The reactor part is replaced by the PFR reactor described in Example 3 without the side streams (92-93). The final model consists of more than 350 variables with detailed models for both the reactor and the distillation parts. As seen in the previous example, this reactor cannot be solved as an IVP reactor because it contains increasing

modes. Unless it is reformulated properly, the flowsheet problem cannot be solved with either sequential or simultaneous approaches.

$$D_A \frac{<Py_A}{} \quad \frac{dy_A}{} \quad {}_{,\ ,,} \quad kAVA \quad = 0$$

$$D_B \frac{d^2 y_B}{dx^2} - u \frac{dy_B}{dx} + k_A y_A - k_B y_B = 0 \tag{92}$$

with initial conditions:

$$\begin{aligned} VA(0) &= VAO \\ dy_A/dx &= dy_A o/dx \\ VB(0) &= VBO \\ dy_B/dx &= dy_{B0}/dx \end{aligned} \tag{93}$$

To generate the Jacobian and to determine the Newton step for the reactor,

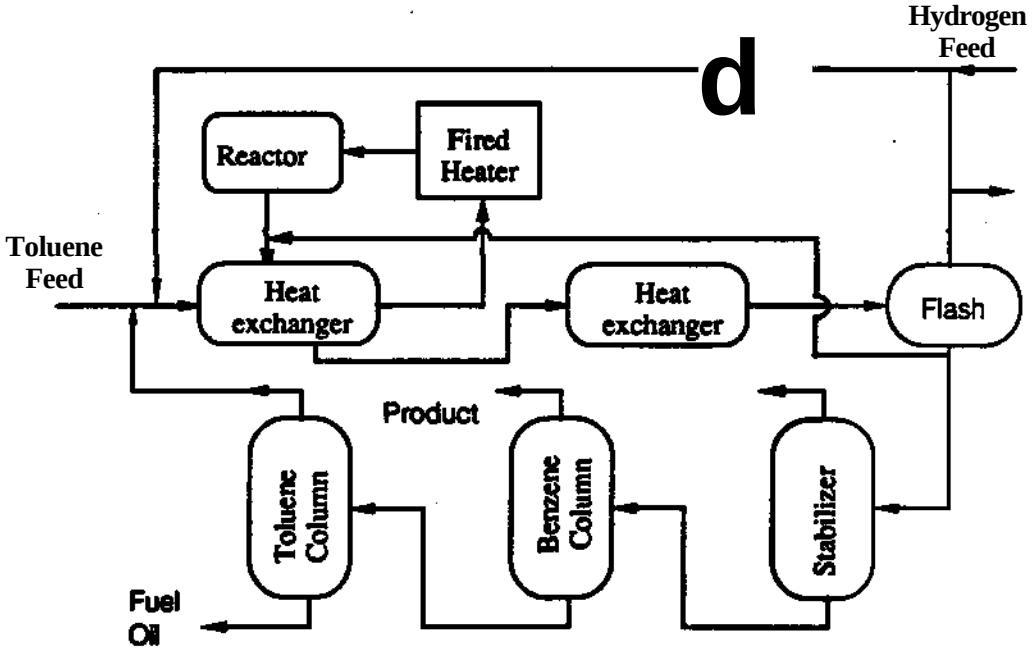

Figure 6: Flowsheet of the process.

we employ decomposition described in Section 3 and Section 4. The distillation equations are derived and then factorized with the Naphthali-Sandhohn

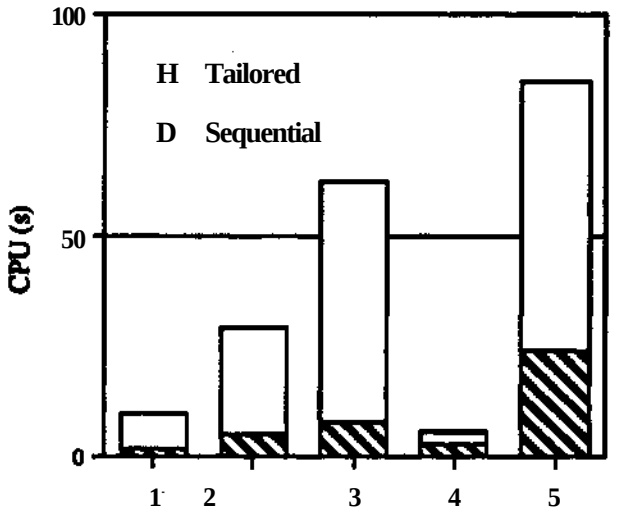

Figure 7: Computational time for the flowsheet optimization example.

model, UNIDIST, which is part of the SEPSIM process simulator [1]. The optimal solutions are obtained using both the tailored approach and the sequential modular approach for various cases. The numerical statistics arc reported in Figure 7. In the figure, Case 1 and Case 2 correspond to the case that only the toluene column and benzene column are modeled as tray by tray calculation, respectively. In case 3, both units are addressed *as* tray by tray models. In case 4, only the reactor is modeled with collocation model and both distillation columns are sharp split separators. In case 5, the reactor and the toluene column have detailed models.

Table 4: Computational statistics for flowsheet example.

Case	No. of SQP iterations		CPU time	
	Tailored	Sequential	Tailored	Sequential
1	7	9	1.9	10
2	14	21	5.1	29.3
3	12	12	7.8	62.2
4	8	8	2.8	5.5
5	26	21	24	85

As a result of simultaneous convergence, the tailored approach performs better in terms of computational time. The performance differences between them seem to be more distinct with more complex problems, at least for this example.

To conclude this section, we would like to point out that the differences and similarity of the tailored and equation-based approaches. Since both approaches are based on an infeasible path strategy, the convergence characteristics are similar. Moreover, Schxnid and Biegler [12] compared both approaches and confirmed that conclusion. However, since the formulation of the problem in the tailored approach is done through the use of existing unit solvers with specialized procedures for each unit model, it not only simplifies the problem setup and can also yield a more robust formulation. As a result, it reduces the effort needed in problem formulation and initialization that cannot be quantified simply as CPU time. Also with the modularity of the unit preserved, the tailored and sequential modular approaches can be programmed to detect the ill-conditioning in the units and adjustment can then be made. Furthermore, the pivot sequences in the tailored approach are done by taking into account the structure of the units in contrast to a general purpose pivoting strategy in the equation-based approach.

As is well known, the trend in process engineering is toward more realistic models and real-time optimization. This will lead to a large demand in terms of computational capability as well as in robust modeling. As a result, we believe that the rSQP approach can be a useful tool that addresses computational speed and meets the robustness requirement for large-scale optimization.

8 Conclusions

This article first reveals the potentially unstable nature of the dynamic optimization problem and the corresponding relationship to the resulting NLP. The paper presents a new methodology for diagnosing this problem and. through the use of problem reformulation step, yields a stable problem formulation. These procedures are integrated into a reduced-Hessian SQP and the collocation solver COLDAE. This framework provides a systematic way to address problems without assuming that the problems are stable and can be applied to various families of problems. A formulation similar to soft and hard constraint enforcement can also be included to keep the solution bounded.

The algorithm proposed can be easily incorporated into a large scale problem such as flowsheet optimization. Here a flowsheet optimization problem is considered via the tailored approach. It is shown in this work that the tailored approach offers several significant advantages over conventional strategies, namely sequential modular and equation-based approaches.

Appendix A: Proof for Property 1

Case 1: ODE and index 1 problems (LTI)

Consider an ODE of the form;

$$\dot{x} = Mx + g(t) \tag{94}$$

Note that the result below is general regardless of the inhomogeneous part. Then let

$$\mathbf{M} = W^T A W \tag{95}$$

Where A is diag(Ai, A_2, • • •, A_n), with
A_x < .. < A_a < 0,
A_a+i < .. < A_n > 0,
W are the eigenvectors of Matrix A/, and
W_a are the eigenvectors of Matrix *M* corresponding to Ai, ..A_o, and
Wb are the eigenvectors of Matrix *M* corresponding to A_o+i, ..An,

Let $s = Wxj$ then

$$\dot{s} = \mathbf{A}s \tag{96}$$

For index one problems, the algebraic variables are then computed from the result of the differential variables *x*.

As a result, we can partition the problem into a decreasing part (Ai to A_o) and an increasing part (A_a+i to A_n). Since this problem is LTI, the fundamental solution *S* can be calculated stably. Clearly for this problem to be well-conditioned, the boundary conditions of the problem have to reflect this underlying structure. Let $[\hat{B}\%\ \hat{B}\pounds]^T]$ be the boundary conditions derived from section 4. From [2], the condition number of the problem is $\backslash S([B? : 0)^T S_a$ + [0 : * f f S*)"^{1}!, where

$$B_a = [\hat{B}_a W_a \backslash \hat{B_a} W_b] \tag{97}$$

$$B_b = \hat{}[B_b W_a \backslash \hat{B}_b W_b] \quad (98)$$

$$S(t) = diagle\wedge i\wedge e\wedge e\wedge +\wedge -V,..\wedge\wedge^{1}\wedge) \quad (99)$$

$$S_a = <fca_{ff}[l,..4,e-\wedge+V.,e-^{A}«] \quad (100)$$

$$S_b = diag[e^{\lambda_1},..,e\backslash l,..,l] \quad (101)$$

Moreover, for problems with fast dynamics min|A| » 1, the condition number is bounded iff $\hat{B}_a W_a$ and $\hat{B_b} W_b$ are both nonsingular. We can then conclude directly from this result that the property 1 holds. D

Case 2: index two problems

The discussion in this part is based on the result in [3]. Consider the following linear BVP.

$$\dot{z} = G_n z + Gny \quad (102)$$

$$G_2 ix = q \quad (103)$$

Since this problem is index two, *G21G12* is nonsingular. Moreover *G12* has full row rank. We can find an orthonormal matrix *R* such that

$$RG12 = 0 \quad (104)$$

Multiply (102) by *R*,

$$R\dot{z} = RG_n z \quad (105)$$

and let

$$x = Rz \quad (106)$$

Differentiating (106) and substituting matrix *T* where *RT* = /. The final ODE of the problem can then be given as:

$$\dot{x} = [(RGxi + \dot{R})T\backslash x + \text{inhompgeneous part.} \quad (107)$$

With the final ODE, we can then use the result in Case 1. D

Appendix B: some examples in the factorization step

In general, "the structure of constraints matrices can vary from the case we present. Here, we assume that $A_y B, CjD$ are equal size. However, most

cases can be decomposed into the following subproblems. A common structure that we often face is:

$$\mathfrak{L} = \begin{bmatrix} E & C & 0 & / & & & & \\ B & A & I & 0 & & & & \\ & & EC & O & & I & & \\ & & B & A & I & 0 & & \\ & & & & E & C & 0 & I \\ & & & & B & A & I & 0 \end{bmatrix} \quad (108)$$

The matrices *A, B,* C, *E* are the same as in (52). In the case where *B*, C are zero, the growth of the sensitivity can be bounded by $\backslash(AE)^{ne}\backslash oo^{*}$ In addition, the row pivoting elements are restricted to a sequence of *(I,A,E,I). U B,C* can serve as pivots and are of moderate size, the bounds are given by $mdx(\backslash(AB^{"l}E + CJ\wedge IocK\wedge\wedge"^{1})^{TM8}\wedge)$. In this CBSe^ the sequence is */, A,* (J, *B).* In general, if the time scales of all profiles are in the same order of magnitude, $AE \sim O(1)$.

Acknowledgments

Financial support from the National Science Foundation through grant ASC-9213149 is gratefully acknowledged. The authors wish to thank Dr. Steve J. Wright for his insightful comments concerning the algorithm. The authors also gratefully acknowledge Dr. Claudia Schmid for the implementation of the flowsheet example.

References

[1] Anderson, P.M., F. Genovese and J. Perregard, "Manual for Steady State Simulator, SEPSIM," Institut for Kemiteknik, DTH, Lyngby, Denmark (1991)

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

[3] Ascher, U.M., L.R. Petzold, "Projected Implicit Runge-Kutta Methods for Differential-Algebraic Equations, *SIAM J. Numer. Anal.* 1991 28(4) pp 1097-1120.

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

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

[6] Brenan, K.E., Campbell, S.L., and L.R. Petzold, *Numerical Solution of Initial'Value Problems in Differential-Algebraic Equations,* Elsevier Science Publishing, New York, NY (1989).

[7] Chamberlain, R.M., C. Lemarechal, H.C. Pedersen, and M.J.D. Powell, 'The watchdog technique for forcing convergence in algorithms for constrained optimization". *Math.Prog.Studies* 1982, 16, pp 1-17.

[8] Desoer, C.A., "Slowly Varying Discrete System", *Elec. Letters* 1970, 6 (11), pp 339-340.

[9] Mattheij, R.M.M., "Decoupling and Stability of Algorithms for Boundary Value Problems" *SIAM Review,* 1986 27 pp 1-44.

[10] Oliveira, N.M.C., and Biegler, L.T. "Constraint Handling and Stability Properties of Model-Predictive Control" *AIChE •/.,* 1994 40 7 pp 1138-1155.

[11] Pantelides, C.C. "The Consistent Initialization of Differential-Algebraic Systems" *SIAM J.Sci.StaLCompuL,* 1988 9 2 pp 213-231.

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

[13] Tuiartkit, P., and L.T. Biegler, "A Nested, Simultaneous Approach for Dynamic Optimization Problems", Proceedings of the PSE'94, pp 145-151, Kyongju, Korea (1994).

[14] Tanartkit, P., and L.T. Biegler, "Stable Decomposition for Dynamic Optimization", *I&EC Research* 1994, 34 pp 1253

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

[16] Wright, S. "Stable Parallel Algorithms for Two-Point Boundary Value Problems" ***S1AM J. Set. Statist Comput.*** **1992,13 pp. 742-764.**