CHEMICAL PROCESS OPTIMIZATION USING NEWTON-LIKE METHODS

A. Lucia and J. Xu
Department of Chemical Engineering, Clarkson University, Potsdam, NY 13676, U.S.A.

(Received 22 December 1988; final revision received 31 July 1989; received for publication 29 August 1989)

Abstract—Various interrelated issues that effect the reliability and efficiency of Newton-like methods for chemical process optimization are studied. An algorithm for solving large, sparse quadratic programming (QP) problems that is based on an active set strategy and a symmetric, indefinite factorization is presented. The QP algorithm is fast and reliable. A simple asymmetric trust region method is proposed for improving the reliability of successive QP methods. Ill-defined QP subproblems are avoided by adjusting the size of the trust region in an automatic way. Finally, it is shown that reliable initial values of the unknown variables and multipliers can be generated automatically using generic problem information, short-cut techniques and simulation tools. Many relevant numerical results and illustrations are presented.

1. INTRODUCTION

A variety of successive quadratic programming (SQP) methods have been developed and used for solving nonlinear programming problems for the past 25 yr (Wilson, 1963; Murray, 1969; Biggs, 1972; Han, 1976; Powell, 1978; Coleman and Conn, 1984; Nocedal and Overton, 1985). This list is by no means complete. Moreover, current interests in the nonlinear programming area have shifted to large-scale problems, and a variety of methods, including large-scale SQP methods, are emerging (Gill, 1989; Toint, 1989; Prieto and Murray, 1989; Gould et al., 1989; Nickel and Tolle, 1989). Nevertheless, it is only relatively recently that SQP methods have found use in chemical process optimization (Berna et al., 1980; Jirapongphan et al., 1980; Biegler and Hughes, 1982; Locke et al., 1983; Biegler and Cuthrell, 1985; Chan and Prince, 1986; Vasantharajan and Biegler, 1987; Kumar and Lucia, 1987; Lucia and Kumar, 1988). This is probably due to the inherent publicity given to Han's seminal 1976 paper by Powell (1978). Again, the list of application papers in chemical process optimization is incomplete.

When the number of variables is large, there are basically two ways to apply SQP methods; either decomposition techniques can be used or natural problem structure can be exploited. Decomposition techniques use the equality constraints to eliminate variables and reduce the size of the quadratic programming (QP) subproblems that must be solved at each iteration; usually only small dense QP subproblems have to be solved. The reasons for advocating decomposition are that QP subproblems can be expensive to solve and that many chemical process optimization problems have a small number of degrees of freedom, even though the total number of unknown variables can be large. However, decomposition techniques make it difficult to use analytical second derivative information and the associated QP subproblems can be large and dense for problems in which the number of degrees of freedom is large. Berna et al. (1980), Locke et al. (1983) and, more recently, Vasantharajan and Biegler (1987) have used decomposition techniques and have reported good results for a variety of chemical process optimization examples. On the other hand, Kumar and Lucia (1987) and Lucia and Kumar (1988) have exploited natural problem structure in applying SQP methods to chemical process optimization. In this approach, large and sparse QP subproblems must be solved at each iteration and a mixture of analytical second derivatives and many small, dense quasi-Newton updates is used to build approximations to the Hessian matrix of the Lagrangian function in the full space of the variables. The motivation for this approach is that more accurate second derivative information improves both the reliability and efficiency of SQP methods and that the resulting reduction in the number of iterations compensates, at least in part, for the computational cost associated with solving large QP subproblems. Good numerical results for this approach have also been reported for a number of distillation optimization problems with as many as 178 unknown variables.

The objective of this paper is to address certain interrelated issues that pertain to the reliability and efficiency of solving chemical process optimization problems by SQP (or Newton-like) methods. These issues include the relative cost of solving QP problems (particularly large ones), the automatic generation of starting points, the need for accurate second derivative information and the utility of stabilization procedures such as line searching and trust region methods. Accordingly, this manuscript is organized in the following way. A general SQP algorithm is given in Section 2. In Section 3, methods for QP are reviewed...
and a new QP algorithm based on an active set strategy and a sparse, symmetric, indefinite factorization is presented. In Section 4, line searching and trust region methods for nonlinearly constrained optimization are surveyed and a simple trust region strategy for chemical process optimization is presented. This trust region method is incorporated directly into the QP subproblems and defines the size of the trust region in a natural way. Various ways of approximating the Hessian matrix of the Lagrangian function are studied in Section 5. These approaches include Newton's method, hybrid methods, which use a mixture of analytical second derivatives and quasi-Newton updates, and pure quasi-Newton updates, suitable for use in the context of a hybrid method, are presented. These new updates are scale invariant in a subspace of the variables. Finally, in Section 6 many numerical results for a variety of chemical process optimization examples, including maximum entropy problems, three-phase and reaction equilibrium calculations and distillation optimization problems, are presented. The primary emphasis of this last section of the paper is on the automatic generation of reliable starting points and the impact of various aspects of QP algorithms on computational efficiency.

The main conclusions of this paper are that:

1. Large QP subproblems can be solved in a reliable and efficient manner, provided both the QP algorithm and the associated re-entry facilities are properly designed. In particular, the sparse QP algorithm presented in this paper is at least an order of magnitude faster than existing public domain software for the same task.
2. Trust regions can be constructed that avoid common reliability difficulties encountered by line searching in process optimization and show that line searching methods are unnecessary in many applications.
3. Nonlinearities in the objective function and/or constraints can be quite strong and, as a consequence, accurate, fast-tracking second derivatives can be required. Newton and hybrid methods provide this derivative information in a reliable manner and frequently give better numerical performance than the (modified) BFGS update.
4. For many chemical process optimization problems, it is possible to generate good starting points based on generic problem information, short-cut techniques and simulation tools.
5. Careful attention to the major subtasks of SQP methods and to initialization procedures for the unknown variables (and multipliers) results in a reliable and efficient overall algorithm capable of solving large problems with a small or large number of degrees of freedom.

2. SUCCESSIVE QUADRATIC PROGRAMMING METHODS

SQP (or Newton-like) methods are a class of methods used to solve the nonlinear programming problem:

$$\min f(x),$$

subject to

$$c_j(x) = 0, \quad j = 1, \ldots, m',$$
$$c_j(x) \leq 0, \quad j = m' + 1, \ldots, m,$$

where $$x \in \mathbb{R}^n$$ is a vector of unknown variables, $$f(x)$$ is the objective function, $$c(x)$$ is a vector of constraint functions, $$m'$$ is the number of equality constraints, and $$m$$ is the total number of constraints. In chemical process optimization, typical examples of the objective function are the entropy function, process utility functions and venture profit functions. The constraint equations, on the other hand, usually include the equations describing the conservation of mass, the conservation of energy, phase equilibrium, bounds on variables, design constraints, and others.

Newton-like methods are based on a quadratic approximation to the Lagrangian function:

$$L(x, \lambda, \mu) = f(x) + \sum_{j=1}^{m'} \lambda_j c_j(x) + \sum_{j=m'+1}^{m} \mu_j c_j(x),$$

and linearized constraint functions. The quantities $$\lambda$$ and $$\mu$$ in equation (4) are vectors of Lagrange and Kuhn–Tucker multipliers, respectively. It is also assumed that the objective function and constraints are twice continuously differentiable.

With this, we give the steps of a general SQP algorithm.

**SQP algorithm:**

1. Initialize $$x, \lambda, \mu$$ and the Hessian matrix of the Lagrangian function, say $$B$$. Set $$k = 0$$ and specify a convergence tolerance $$\varepsilon > 0$$.
2. Compute $$f(x_k), g(x_k), c(x_k)$$ and $$J(x_k)$$, where $$g$$ is the gradient of the objective function and $$J$$ is the Jacobian matrix of the constraints. The $$j$$th column of $$J^T$$ is the constraint normal $$\nabla c_j$$.
3. Check the Kuhn–Tucker conditions:

$$\|\nabla L\|_2 = \|\nabla_J L [c(x)]^T \mu^T c(x)\|_2 < \varepsilon,$$
$$\mu_j \geq 0, \quad j = m' + 1, \ldots, m.$$  

If satisfied, stop. Otherwise go to Step 4:
4. Solve the QP subproblem:

$$\min g^T(x_k) d_k + \frac{1}{2} d_k^T B_k d_k,$$

subject to

$$c_j(x_k) + \nabla c_j^T (x_k) d_k = 0, \quad j = 1, \ldots, m',$$
$$c_j(x_k) + \nabla c_j^T (x_k) d_k \leq 0, \quad j = m' + 1, \ldots, m,$$

where $$d_k$$ is the change in the unknown variables and $$B_k$$ is some approximation to $$\nabla_{xx} L$$. 

A. LUCIA and J. XU
5. Calculate a step in the unknown variables, say $s_k$ based on $d_k$.
6. Compute $x_{k+1} = x_k + s_k$ and $f(x_{k+1})$, $g(x_{k+1})$, $c(x_{k+1})$ and $J(x_{k+1})$, if necessary.
7. Calculate $B_{k+1}$ in some manner, set $k = k + 1$ and go to Step 3.

The statement of the algorithm is not intended to be vague but rather general and flexible. Furthermore, it is precisely the details of the steps of the algorithm that are the subject of this paper. We begin with Step 4, solving the QP subproblem.

3. METHODS FOR QUADRATIC PROGRAMMING

In SQP methods, the purpose of solving a QP subproblem is to define an iterative change in the unknown variables and to provide estimates of the Lagrange and Kuhn–Tucker multipliers. In this section, methods for QP are briefly reviewed and a new QP algorithm based on an active set strategy and a sparse, symmetric, indefinite factorization is presented.

3.1. Existing algorithms

Linear programming (LP)-based QP methods convert a QP problem into an LP problem and solve the resulting linear program by (revised) simplex techniques (Beale, 1955; Barankin and Dorfman, 1958; Wolfe, 1959). These methods possess certain computational advantages when compared to other methods. They permit the use of artificial variables for determining feasible solutions to the constraints, and they are well-suited for parametric programming. However, there are disadvantages as well. LP-based QP methods increase the number of variables in the problem because they must introduce surplus and slack variables to handle inequality constraints. This is a serious disadvantage for large QP problems. LP-based QP methods are also inefficient when applied to problems with large degrees of freedom.

The second major class of methods for QP problems are those based on the concept of an active set, and the direct solution of the Kuhn–Tucker conditions for the QP problem (Fletcher, 1971; Gill and Murray, 1978). These methods are iterative and define a set of active constraints (i.e. a set of equality constraints plus inequality constraints that hold as equalities) at each iteration of the calculations. The solution to the Kuhn–Tucker conditions for the QP problem defined by the current active set, which is a system of linear equations, is usually determined by partitioned Gaussian elimination or factorization. If this solution also satisfies the Kuhn–Tucker conditions for the original QP problem, then the problem is solved. If not, the set of active constraints is updated (i.e. one inequality constraint is either added to or deleted from the set) using solution information from the current active set and the QP problem is solved again. This procedure is repeated until the solution to the QP problem defined by some active set solves the original QP problem. Recursion formulae are used for updating the corresponding changes in the generalized inverse of the constraint Jacobian and the inverse of the projection of the Hessian matrix (or appropriate factorizations), and these are an essential part of the computational efficiency of any active set method. Also, some simplex-based method is needed to determine an initial feasible point, which serves to define the initial active constraint set.

Finally, other methods for QP problems have been proposed (Thiele and van de Panne, 1960; Fletcher and Jackson, 1974). However, these methods usually have additional problem restrictions, such as convexity or constraints in the form of upper and lower bounds only (Fletcher and Jackson, 1974), and are therefore not applicable in the general case.

3.2. Available software

Fortran subroutines for both LP-based and active set methods for QP problems are readily available and, as a result, are widely used within SQP algorithms for chemical process optimization. The most widely used QP software packages are QPSOL, VE02AD and VE06AD.

QPSOL is available through Stanford University (Office of Technology Licensing) and is an active set method based on an algorithm given in Gill and Murray (1978). It contains an LP phase to determine an initial feasible point and uses a QR factorization of the Jacobian matrix of the active constraint set and a modified Cholesky factorization of the projection of the Hessian matrix, respectively. The factors are updated in a way that avoids complete refactorization, and the method is intended for problems in which the Jacobian and Hessian matrices are dense matrices. QPSOL has been used by Kisala et al. (1986) and Vasantharajan and Biegler (1987) in chemical process optimization applications, as well as others in chemical engineering.

VE02AD is a set of Fortran programs that is based on the general QP algorithm of Fletcher (1971). Like QPSOL, it uses a simplex method to determine an initial feasible point and is intended for problems in which the Jacobian and Hessian matrices are dense. However, unlike QPSOL, VE02AD is based on block Gaussian elimination, and recursion relations are used for modifying the generalized inverse of the constraint Jacobian and inverse of the projection of the Hessian matrix. Again no restrictions, other than symmetry, are placed on the Hessian matrix. VE02AD is available in the Harwell Subroutine Library and has been used in chemical process optimization applications by Chan and Prince (1986).

Finally, Kumar and Lucia (1987) have used the sparse QP routine VE06AD in solving separation process optimization problems by SQP methods. VE06AD is also available in the Harwell Subroutine Library and is an LP-based QP method. It is intended for applications in which the constraint Jacobian and
Hessian matrices of the QP problem are large and sparse matrices. The Hessian matrix is assumed to be positive definite (although this is not strictly necessary) and symmetric, and the underlying LP problem is solved using LU factorization in conjunction with the simplex method. VE06AD is, to our knowledge, the only software package generally available for solving large, sparse QP problems.

### 3.3. A quadratic programming method

In this section, a method for large sparse quadratic programming that is based on an active set strategy and a modification of the symmetric, indefinite factorization of Bunch and Parlett (1971) is outlined.

The key to develop a sparse QP method was motivated by our experience with the LP-based algorithm VE06AD on large problems. While VE06AD is extremely reliable, for problems involving 100 or more unknown variables, it requires very large amounts of computer time to solve each iterative QP subproblem. For example, for the acetone-acetonitrile separator studied by Lucia and Kumar (1988), in which there are 121 unknown variables, the CPU time required by VE06AD to solve each of the seven QP subproblems ranged from 629 to 1143 s on an IBM 4341 computer. Similar high computational demands have been observed for other large problems and this, in our opinion, is unacceptable.

In the material that follows, an outline of the sparse QP method is presented. For convenience, consider the subproblem defined in Step 4 of the SQP algorithm in the form:

$$\min g^T d + \frac{1}{2} d^T B d,$$

subject to

$$Jd + c \leq 0,$$

where the Hessian matrix $B$ is symmetric but not necessarily positive definite. Also, the dependence of $g$, $B$, and $J$ on $x_k$ can be neglected.

**QP algorithm:**

1. Set $g = g(x_k)$, $B = B_k$, $J = J(x_k)$, $c = c(x_k)$ and $j = 0$. For $k = 0$, determine an active set of constraints using a phase I linear programming method, say $A_1$. Set $A_j = A_k$.

2. Solve the system of linear equations:

$$\begin{bmatrix} B & J^T_j \\ J_j & 0 \end{bmatrix} \begin{bmatrix} d_j \\ \lambda_j \end{bmatrix} = \begin{bmatrix} -g_j \\ -c_{A_j} \end{bmatrix},$$

for $d_j$, $\lambda_j$, and $\mu_j$.

3. Check the convergence condition:

$$\mu_i \geq 0, \quad \text{for } i \in A_j, \quad (13)$$

$$Vc_i^T d_j < -c_i, \quad \text{for } i \notin A_j, \quad (14)$$

for $i = m' + 1, \ldots, m$. If satisfied, set $A_{k+1} = A_j$ and stop. Otherwise, go to Step 4.

4. Find the index $i$ that corresponds to $\max|\mu_i|$ and $\mu_i < 0$. Let $i_{\text{max}}$ denote this index.

5. For each inequality constraint that is in the active set $A_j$, if:

$$\mu_i \geq 0, \quad \text{then } i \in A_{j+1}, \quad (15)$$

$$\mu_i < 0, \quad \text{then } i = i_{\text{max}} \notin A_{j+1}. \quad (16)$$

6. For each inequality constraint that is not in the active set $A_j$, if:

$$Vc_i^T d_j < -c_i, \quad \text{then } i \notin A_{j+1}, \quad (17)$$

$$Vc_i^T d_j \geq -c_i, \quad \text{then } i \in A_{j+1}. \quad (18)$$

Let $m_{A_{j+1}}$ denote the number of active inequality constraints in the set $A_{j+1}$.

7. If $m_{A_{j+1}} \leq n - m'$, go to Step 8. Otherwise:

$$\max_{x_{A_{j+1}}} [-c_i Vc_i^T d_j] i \notin A_{j+1}. \quad (19)$$

until $m_{A_{j+1}} = n - m'$.

8. Set $j = j + 1$ and go to Step 2.

Step 1 of the QP algorithm determines an initial feasible solution to the constraints on the first iteration of the SQP method. We use a phase I linear programming method in which artificial variables are introduced and an appropriate LP problem is solved (see p. 43 in Luenberger, 1973). This also allows us to identify either redundant equations or situations in which no feasible solution to the constraints exists. This is important because we intend to use trust region methods to improve the reliability of Newton-like methods for nonlinearly constrained optimization and it is well-known that trust region methods can produce ill-defined QP subproblems in which no feasible solution to the linearized constraints and trust region inequality exist (see Sections 4.2 and 4.3). The sparsity of the Jacobian matrix presents no difficulty in implementing a phase I LP method and can be handled in a straightforward manner using $LU$ decomposition. On iterations other than the first SQP iteration, we initialize the active set to the active set at the solution of the previous QP subproblem, as indicated in Step 3. We only use the phase I LP technique to ensure that the constraints have a feasible solution. Phase I methods are well-known (Luenberger, 1973; Gill et al., 1981) are used by Gill and Murray (1978) in QP SOL and will not be discussed further.

In Step 2 of the QP algorithm, the Kuhn-Tucker conditions for the current active set are solved. Unlike Fletcher (1971) and Gill and Murray (1978), we do not partition the coefficient matrix in equation (12). Instead, the symmetric, indefinite factorization of Bunch and Parlett (1971) modified for sparsity is used to decompose the coefficient matrix. $LDL^T$ factorization, where $D$ is a block diagonal matrix with $1 \times 1$ and $2 \times 2$ nonzero diagonal blocks, is used. Complete diagonal pivoting is needed to maintain numerical stability, the factorization is
computationally fast, and the $L$ and $D$ factors are updated based on changes in the active set so that complete refactorization is avoided. The reasons that we use a symmetric, indefinite factorization are because the coefficient matrix in equation (12) is not partitioned and because the Hessian matrix $H$ is not required to be positive definite. The reader is referred to the paper by Bunch and Parlett (1971) for a detailed description of the factorization method.

Step 3 checks the Kuhn--Tucker conditions for the current active set. If they are satisfied, a local solution to the given QP subproblem has been found, the current active set indices are saved to start the next iteration, and the QP algorithm returns to the Newton-like method. Otherwise, the current active set is modified based on information that is available from the solution to equation (12).

Steps 4--7 constitute the active set strategy. In Step 4, the inequality index that corresponds to the largest negative Kuhn--Tucker multiplier is identified. This index is used when it is necessary to delete a constraint from the active set. Step 5 checks the signs of the Kuhn--Tucker multipliers for those inequality constraints in the current active set. If the multiplier for the $i$th inequality constraint is nonnegative, then that constraint is included in the next active set $A_{i+1}$. If the multiplier is negative, then only the constraint with the largest negative multiplier is dropped from the active set. In Step 6, the inequality constraints that were not in the active set $A_i$ are tested to see if the solution to equation (12) violates any of these inactive inequalities. If the $i$th inequality constraint satisfies equation (17), then it is still inactive and is not included in the set $A_{i+1}$. If, on the other hand, any inactive inequality constraint is violated, then it is included in $A_{i+1}$. We note here that condition (16) permits only one constraint to be deleted from the active set at a time. This avoids cycling of the QP algorithm. However, equation (18) allows multiple constraints to be added to the active set in a single iteration of the QP algorithm. Because multiple constraints can be added to the active set, it is necessary to include the test in Step 7 of the algorithm in order to avoid the situation in which the number of active inequality constraints exceeds the maximum allowable number of inequalities, $n - m'$. Remember, equality constraints are included in any active set. If the number of active inequalities exceeds the maximum that is allowed, then one or more is deleted from the set until the number of active inequalities is maximum. The test given by equation (19) shows that those constraints whose violations are largest in magnitude are not included in $A_{i+1}$. For all changes in the active set, precautions are included to avoid linearly dependent and infeasible constraint sets.

With this, the active set $A_{i+1}$ is correctly defined and the algorithm resolves the Kuhn--Tucker conditions corresponding to the updated active set $A_{i+1}$. We reiterate that the $L$ and $D$ factors of the coefficient matrix are updated during any iteration in the QP algorithm. This is essential for computational efficiency. Finally, we remark that the ability to add multiple constraints to the active set at once and retaining the active set constraints indices from one SQP iteration to the next also contribute to the efficiency of the algorithm since they often significantly reduce the number of linear systems that must be solved within the QP algorithm.

4. STABILIZATION PROCEDURES

Stabilization procedures are frequently used to improve the reliability of SQP methods. This is usually accomplished by altering the magnitude and/or direction of the iterative change in the unknown variables, when necessary, in order to reduce the value of some appropriately determined merit function. The two most commonly used stabilization procedures for nonlinearly constrained optimization are line searching methods and trust region strategies. In this section, line searching and trust region methods for nonlinearly constrained optimization are briefly reviewed and their performance in chemical process optimization is highlighted. Also, a simple trust region strategy for nonlinearly constrained optimization is presented that shows that line searching is generally unnecessary.

4.1. Line searching methods

Many line searching techniques have been proposed to improve the reliability of SQP methods (Han, 1976; Powell, 1978; Schittkowski, 1981; Chamberlain et al., 1982; Fletcher, 1984; Powell and Yuan, 1986a, b). While these techniques vary somewhat in their choice of line search function (i.e. differentiable, nondifferentiable, exact penalty or augmented Lagrangian) and line search solution approximation (i.e. exact or inexact), in all cases the change in the unknown variables is the Newton-like direction. It is the magnitude of the Newton-like step that is varied to generate a monotonically decreasing sequence of values for the line search function and establish convergence.

There are two principal difficulties with line search methods that are relevant to chemical process calculations, the Maratos (1978) effect and the occurrence of near singular, nondenest and/or unbounded Newton-like directions. In either case, simple examples can be constructed that illustrate that these numerical difficulties represent legitimate practical concerns.

To illustrate the Maratos (1978) effect, consider the simple vapor–liquid maximum entropy problem involving a mixture of methanol, ethanol, acetone and water (see Section 6.2 for a description of the mathematical model). Let the feed to the flash vessel be $(35.8894, 26.1543, 29.6833, 39.6923)$ kmol h$^{-1}$ of methanol, ethanol, acetone and water, respectively at $T = 337.707$ K and $p = 1.013 	imes 10^5$ Pa. Also, fix the heat duty to the vessel at $Q = 2.32043 	imes 10^6$ J h$^{-1}$ and
the flash pressure at 1.013 \times 10^5 \text{ Pa}. The solution to this problem is

\[ x^* = (n_1^*, n_2^*, \ldots, n_k^*) = (82.627, 33.7181, 4.31893 \times 10^{-5}, 6.90608 \times 10^{-5}), \]

at which the total liquid and vapor are 116.345 and 1.12797 \times 10^{-4} \text{ kmol h}^{-1}, respectively. Using the starting point described in Section 6.2.1, Newton's method and the thermodynamically constrained hybrid method with and without line searching fail to converge because they repeatedly encounter the liquid phase boundary. This is because the Newton-like directions in the neighborhood of the solution are near-singular directions (i.e. they are in the direction predicted by the Gibbs-Duhem equation for the vapor phase) since the projected Gibbs-Hessian matrix \( (\nabla^2 G + \nabla^2 G') \) is completely dominated by \( \nabla^2 G' \) which has a Gibbs-Duhem singularity. Thus, any attempt to remove vapor from the system causes the iterates to encounter the liquid phase boundary.

Furthermore, line searching does not work because improvements in the Gibbs free energy function are difficult to measure in near singular directions. This does not mean, however, that near-singular directions are not useful; they are. In fact, Newton's method and the thermodynamically constrained hybrid method with a simple trust region strategy without line searching converge to an accuracy of \( 10^{-8} \) in 25 and 28 iterations, respectively, on this example. The rate of convergence, however, is only linear with common ratio of 1/2 (see Griewank, 1985).

### 4.2. Trust region methods

Recently, several trust region methods for nonlinearity equality constrained optimization have been proposed (Vardi, 1985; Celis et al., 1985; Fletcher, 1985; Powell and Yuan, 1986a, b), primarily because of the limitations of line searching methods and because of the relevance that trust region methods have to problems in which descent directions on the tangent subspace cannot be guaranteed (e.g. problems in which the sparsity of the Hessian matrix is important or in which positive definiteness of the projected Hessian approximation cannot be ensured).

Trust region methods have not, however, been applied to problems involving inequality constraints.

The main idea behind the trust region methods for nonlinear equality constrained optimization is straightforward. A trust region is inscribed in the full space of the variables by adding an inequality constraint of the form:

\[ \| d_k \| \leq \Delta_k, \]

(20)

to the constraint set. The quantity \( \Delta_k \) is the trust region radius, which is adjusted iteratively. The additional constraint is incorporated into the QP subproblem either directly or by linearization, thus implementation is straightforward in this respect.

Furthermore, the trust region constraint ensures that the Newton-like steps are bounded, and thereby removes the need for the projected Hessian matrix to be positive definite. Also, some type of merit function is commonly used to decide whether to accept the step that is calculated.

However, trust region methods for equality constrained optimization are not without disadvantages, the most serious of which stems from the fact that the linearization of the original constraint set may not have a solution within the trust region (see p. 578 in Vardi, 1985).

To circumvent this difficulty, the linearized equality constraints are usually modified in some way. Vardi (1985) modifies the constraints according to the rule:

\[ J_k d_k = -2c(x_k), \]

(21)
where $\alpha$ is chosen so that the intersection of the constraints and the trust region inequality are not null. To determine a suitable value of $\alpha$, the solution of additional QP subproblems are often required. Vardi also uses a nondifferentiable penalty function to decide whether to accept a step and shows that for small values of the trust region radius, the direction that is calculated is a steepest descent step in the Lagrangian function. Powell and Yuan (1986a, b), on the other hand, modify the linearized equality constraints by lumping them into a single inequality constraint of the form:

$$\| c(x_k) + J_k d_k \|_2 \leq \xi_k,$$

(22)

where $\xi_k$ is any number that satisfies:

$$\min_{|d_k| \leq h_k \Delta_k} \| c(x_k) + J_k d_k \|_2 \leq \xi_k \leq \min_{|d_k| \leq \eta_k \Delta_k} \| c(x_k) + J_k d_k \|_2,$$

(23)

where $0 < h_k \leq b_1 < 1$. The quadratic objective function given by equation (7) is minimized subject to the pair of inequality constraints defined by equations (20) and (22). Powell and Yuan emphasize that the choice for $\xi_k$ gives some freedom in the directions that are computed and also use a differentiable exact penalty function to decide whether or not to accept $d_k$. However, determining $\xi_k$ is a nontrivial task and often requires trial and error calculations.

4.3. An asymmetric trust region method

In this section, we focus attention on trust region methods for nonlinearly constrained optimization because line searching methods sometimes cause SQP methods to fail on relatively simple chemical process optimization problems (e.g. near phase boundaries) and because we intend to use sparse approximations to the Hessian matrix of the Lagrangian function which may not be positive definite on the tangent subspace. It is shown that straightforward and computationally useful trust region methods for chemical process optimization can be defined in a natural way.

In chemical process applications, line searching methods frequently cause failure because they have difficulties with discontinuities in the model and its derivatives when component flow rates tend to zero or when phase boundaries are encountered. This is true for both small and large problems using either nondifferentiable or differentiable line search functions, as illustrated in the previous section. Furthermore, the unknown variables common to chemical process optimization (e.g. component flow rates, temperatures, heat duties, pressures, etc.) are usually bounded by physical constraints such as operating regime limits, nonnegativity constraints and, indirectly, component mass balance equations. Consequently, unbounded Newton-like directions do not occur.

We define asymmetric trust regions by placing simple bounds on some or all of the components of the Newton-like step. However, these bounds take into account any natural physical bounds that exist and, as a result, can be different depending on the type of variable (i.e. component flow rate, temperature, etc.), the bounds and the application. For example, for minimum Gibbs free energy or maximum entropy calculations, we bound the changes in the component flow rates by the simple set of inequalities:

$$-\beta_i (n'_i)_k \leq (\Delta n'_i)_k \leq \beta_i (n'_i)_k,$$

(24)

where $(n'_i)_k$ is the component flow rate of the $i$th component in the $k$th phase on the $k$th iteration of the calculations, $0 < \beta_i < 1$ and $\beta_i \leq \beta_i$. Temperatures are usually bounded by the bubble and dewpoint temperatures of the feed or some other appropriate limits. Note that these constraints define an asymmetric trust region whose size changes from one iteration to the next. The phase 1 LP procedure used to generate an initial feasible solution to the QP subproblem is used to ensure that the original constraint set and the trust region bounds define a consistent set of linearized constraints. If inconsistencies occur, the trust region is enlarged based on information available from the QP algorithm frequently without resolving the QP subproblem. This is an important point, and will be illustrated shortly. Furthermore, the trust region inequalities are linear so that linearly constrained optimization problems such as minimum Gibbs free energy problems remain linearly constrained. The other important point is that the trust region avoids zero component flow rates and phase boundaries from being encountered because of the presence of the lower bounds and the way in which they are adjusted. It does not, however, prevent solutions that lie near phase boundaries or have some small values of component flow rates from being computed (see the illustrative example in the previous section).

To illustrate that trust region methods for nonlinearly constrained optimization can give rise to ill-defined QP subproblems, consider the simple VLE equilibrium calculation of a binary mixture by minimizing the total Gibbs free energy of the system. Let the feed be $(0.91234, 20.0218) \text{ kmol h}^{-1}$ of ethanol and $n$-hexane, respectively, at $T = 334.983$ K and $p = 1.013 \times 10^5$ Pa. Let the temperature and pressure of the vessel be the same as the feed temperature and pressure. Finally, let $\beta_1 = \sqrt{2}/2$ and $\beta_2 = -\sqrt{2}$ in equation (24) be fixed for each iteration of the calculations, and let the starting point be $x_0 = (n_1^x, n_2^x, n_1^z, n_2^z) = (5.544 \times 10^{-5}, 0.2088, 0.9118, 19.813)$. The components are in the order ethanol, $n$-hexane and the starting point is based on the dewpoint of the feed. Using the thermodynamically constrained hybrid method of Kumar and Lucia (1987) and the QP algorithm described earlier, an ill-defined QP subproblem is encountered on iteration...
5 because the trust region defined by equation (24) is too small. On iteration 5, the values of the unknown variables are $(n_{1}^*, n_{2}^*, n_{3}^*, n_{4}^*) = (0.01883, 0.0924, 0.8935, 12.292)$ and the bound that causes numerical difficulty is the lower bound for $\Delta n_{1}^*$, the change in the moles of $n$-hexane in the vapor. Equation (24), the current value of $n_{1}^*$ and the value of $\beta_{i} = \sqrt{2}/2$ imply that:

$$-9.1423 = -\beta_{i} n_{1}^* \leq \Delta n_{1}^*. \quad (25)$$

Moreover, the correct active set for this iteration includes only the upper bounds on $\Delta n_{1}^*$ and $\Delta n_{2}^*$. With that correct active set, however, the change in the unknown variables is $\Delta n = (0.0266, 10.0301, 0.0266, 10.0301)$ and it is easily seen that the lower bound on $\Delta n_{1}^*$ is violated. Consequently, the QP subproblem has no solution within the specified trust region! Unlike all standard trust region methods for constrained optimization, we resolve this difficulty by modifying the trust region bounds within the feasible region to avoid ill-defined QP subproblems.

In the case of the illustrative example, the condition:

$$-12.929 = -n_{2}^* \leq \Delta n_{2}^* \quad (26)$$

follows from the nonnegativity constraints that define the feasible region. Furthermore, this inequality is trivial to calculate during the course of solution from knowledge of the size of the feasible region, and indicates that it is possible to adjust the size of the trust region (within the feasible region) to avoid an ill-defined QP subproblem. In particular, because the correct active set gives $\Delta n_{1}^* = -10.0301$ and this value of $\Delta n_{1}^*$ satisfies equation (26), it follows that the trust region should be enlarged to:

$$-10.0301(1 + \epsilon) = \beta_{i} n_{1}^* \leq \Delta n_{1}^*. \quad (27)$$

where $\epsilon$ is a small number, from which a useful value of $\beta_{i}$ is easily calculated. However, note that the QP subproblem does not have to be resolved; the step can simply be accepted. When this is done, the ill-defined QP subproblem encountered on iteration 5 is removed and the calculation proceeds without difficulty to the solution and converges to an accuracy of $10^{-5}$ in 14 iterations.

Similar behavior exists in larger problems such as those that arise from the optimization of the operating conditions for multistage separators (Lucia and Kumar, 1988 or Section 6.3). It is not uncommon for some of the component flow rates or an entire phase associated with one or more stages in a separator to tend to zero during the course of iteration. When this occurs, it usually propagates through the separator and gives physically unrealistic component material balance calculations within or around the separator. Furthermore, this behavior usually manifests itself as line searching failures. However, in our opinion, failure is really due to the associated discontinuities in the model, discontinuities in some of the elements of the Jacobian matrix and/or singularities in the Jacobian or Hessian matrices that accompany near-zero component and total flow rates. For example, in the acetone-acetonitrile separation discussed at length in Lucia and Kumar (1988), a secant-only hybrid method failed because it encountered a lower bound for the acetonitrile in the vapor stream leaving the condenser. This lower bound remained active from iteration 4 to iteration 16, caused low values of component molar flow rates for both acetonitrile and water to propagate through the rectifying section of the column and eventually caused failure on iteration 16 due to an uphill search direction. The same secant-only hybrid method also failed on an optimization problem derived from a separation originally studied by Gallun and Holland (1976). Failure in this case was due to line searching difficulties caused by very small amounts of methanol in the bottom section of the column.

In each of these cases, failure can be avoided by simply replacing the nonnegativity constraints with the simple trust region inequalities given by equation (24). This is illustrated in the section on numerical results.

Finally, note that this simple idea of using inequalities to define a trust region based on physical insight carries over verbatim to other common chemical process optimization problems as well (e.g. heat exchanger networks, multistage compression units, refrigeration cycles, etc.).

5. METHODS FOR APPROXIMATING THE HESSIAN MATRIX

For optimization problems with strong nonlinearities, accurate approximations to the Hessian matrix of the Lagrangian function are often required for the numerical performance of SQP methods to be reliable and efficient. In this section, methods for approximating the Hessian matrix in nonlinearly constrained optimization are briefly reviewed. Also, a new scale invariant quasi-Newton update for use in the context of a hybrid Newton-like method is presented.

5.1. Existing methods

The Hessian matrix of the Lagrangian function is given by:

$$\nabla_{x}^{2} L = \nabla^{2} f(x) + \sum_{j=1}^{m} \lambda_{j} \nabla^{2} c_{j}(x) + \sum_{j=m+1}^{m+n} \mu_{j} \nabla^{2} c_{j}(x). \quad (28)$$

where $\nabla^{2} f(x)$ and $\nabla^{2} c_{j}(x)$ are the matrices of second partial derivatives of the objective function and $j$th constraint function, respectively. For many applications in chemical process optimization, these matrices are sparse.

There are several well-known methods for approximating the Hessian matrix in the full space of the variables. These include Newton's (or Wilson's) method with analytical and/or finite difference derivatives, the (modified) BFGS formula and hybrid
methods that use a mixture of analytical derivatives and quasi-Newton updates. These methods, and others, can be conveniently described in a general Newton-like framework in which the approximation of the Hessian matrix of the Lagrangian function $B$, is split according to the rule

$$B = C + A,$$  

(29)

where $C$ and $A$ are the computed and approximated parts of the Hessian matrix, respectively. The symmetric matrices $C$ and $A$ each contain second derivatives of the objective function and constraints. That is:

$$C = C_0 + \sum_{j=1}^{m} \lambda_j C_j + \sum_{j=m+1}^{m'} \mu_j C_j,$$  

(30)

and

$$A = A_0 + \sum_{j=1}^{m} \lambda_j A_j + \sum_{j=m+1}^{m'} \mu_j A_j,$$  

(31)

where $C_0$ and $A_0$ denote the computed and approximated parts of $\nabla^2 f(x)$ and $C_j$ and $A_j$ are the same respective parts of $\nabla^2 c_j(x)$.

Newton's method approximates the matrices $C$ and $A$ by using analytical and/or finite difference second derivatives in equations (30) and (31). The modified BFGS update, on the other hand, builds approximations to $B$ directly using the formula:

$$B = B + \frac{\eta \eta^T}{\eta^T s} \frac{B s s^T B}{s^T B s},$$  

(32)

where

$$B = B_{k+1}, B = B_k, s = x_{k+1} - x_k,$$

and

$$\eta = \theta y + (1 - \theta) B s,$$  

(33)

$$\theta = \begin{cases} 
1 & \text{if } y^T s \geq 0.2 s^T B s, \\
0.8 s^T B s / (s^T B s - y^T s) & \text{if } y^T s < 0.2 s^T B s.
\end{cases}$$  

(34)

and where

$$y = \nabla_x L(x_{k+1}, \lambda_{k+1}, \mu_{k+1}) - \nabla_x L(x_k, \lambda_{k+1}, \mu_{k+1}).$$

For many chemical process optimization applications, the matrices $C_j$ and $A_j$ for $j = 0, 1, \ldots, m$ have many small dense symmetric blocks due to the separability of the functions usually associated with phases and streams in any process. Kumar and Lucia (1987) use the idea of Griewank and Toint (1982) and approximate the blocks of $A$ by using a symmetric, dense quasi-Newton update. Kumar and Lucia (1987) also exploit the homogeneity of thermodynamic functions and their derivatives and use the method of iterated projections (Dennis and Schnabel, 1979) and the PSB update to build thermodynamically constrained quasi-Newton approximations to the blocks of $A$. More specifically, the approximation to the $i$th block of $A$ is given by the sequence of PSB updates:

$$A_i' = A_i + \frac{(y - A s) s^T + s (y - A s)^T}{s^T s} - \frac{[(y - A s) s^T s] s^T}{(s^T s)^2}$$  

(35)

and

$$\bar{A}_i = A_i' + (r - A' z) z^T + z (y - A' z)^T/(z^T z) - \frac{[(r - A' z) z^T z] z^T}{(z^T z)^2},$$  

(36)

where $A_i = (A_i)_i, \bar{A}_i = (A_i)_i + 1$ and where the vectors $s = s_i, y = y_i, z = z_i$ and $r = r_i$ are defined for each block of $A$. Consequently, each symmetric block of $A$ satisfies a thermodynamic constraint in the form:

$$A_i z_i = r_i,$$  

(37)

at each iteration of the calculations and an appropriate secant condition

$$\bar{A}_i s_i = y_i,$$  

(38)

in the limit. See Kumar and Lucia (1987) for details.

5.2. Subspace invariant quasi-Newton updates

The BFGS formula has a history of reliable numerical performance for small problems and this good behavior is usually attributed to the fact that it is invariant under affine transformations of the variables (see Dennis and Schnabel, 1979). On the other hand, it is well-known that the numerical performance of the PSB update depends on the scale of the variables. Thus, the hybrid method of Kumar and Lucia has the potential for numerical difficulties due to poor scaling of the variables.

The obvious way to remove this potential difficulty is to use a scale invariant update like the BFGS formula to approximate the blocks of $A$. However, this does not work very well because the symmetric blocks of $A$ are usually indefinite matrices (see p. 1386 in Lucia et al., 1985). The symmetric rank one (SR1) update, which is also scale invariant, does not give consistently good numerical performance in this context either.

To circumvent the potential scaling difficulties associated with the PSB update, we suggest that information contained in the computed part of the Hessian matrix be used to capture the scale of the variables. Following Dennis and Schnabel (1979), the appropriate variational calculus problem that defines
the least change quasi-Newton approximation to the
ith block of \( A \) is given by:

\[
\min \| W_i(A_i - A_i)W_i \|_F, \tag{39}
\]
subject to

\[
\begin{align*}
A_i s_i &= y_i, \tag{40} \\
A_i z_i &= r_i, \tag{41} \\
A_i &= A_i^T, \tag{42}
\end{align*}
\]
where \( W_i \) is a symmetric, positive definite matrix and \( \| \cdot \|_F \) denotes the Frobenius matrix norm. The problem defined by \( i \) has no analytical solution but the solution can be approximated using the method of iterated projections. This approximate solution is given by the sequential update:

\[
\begin{align*}
A' &= A + \frac{(y - A s)q^T + q(y - A s)^T}{q^T s} - \frac{(y - A s)q}{(q^T s)}qq^T, \tag{43} \\
\bar{A} &= A' + \frac{(r - A' z)u^T + u(r - A' z)^T}{u^T z} - \frac{(r - A' z)}{(u^T z)^2} uu^T, \tag{44}
\end{align*}
\]

where \( u = W^{-2}z \). Again, we have dropped the subscript \( i \) for convenience; however the reader should keep in mind that all quantities, including the weighting matrix \( W_i \), are associated with the \( i \)th block of \( A \).

Note that \( W^{-2}z \) gives the iterated projection sequence using the PSB update, which is defined by equations (35) and (36). We suggest that the matrix \( W^{-2}z \) be approximated using analytical second derivative information that is contained in the computed part, \( C \). Obviously, the way in which this is done can vary considerably from one application to the next. However, in the next section it is shown that there are applications in which \( W^{-2}z \) has a natural interpretation and in which the update defined by equations (43) and (44) gives better numerical performance than iterated projections with the PSB update.

6. NUMERICAL RESULTS

Our experience indicates that the initial estimates of the unknown variables and multipliers can have a significant impact on the reliability and efficiency of Newton-like methods for chemical process optimization. In fact, many numerical difficulties are either created or removed by starting-point considerations. In this section, it is shown that starting points that give reliable and efficient numerical performance of Newton-like methods can usually be generated automatically using generic problem information, shortcut methods and, where appropriate, simulation tools. These starting strategies, which are problem dependent, are combined with the techniques presented in Sections 3, 4 and 5 to form a Newton-like framework for chemical process optimization. Within this framework, a variety of Newton-like methods are used to solve some common optimization problem in chemical process engineering.

6.1. Algorithm details

The following algorithmic details are pertinent to the general SQP algorithm given in Section 2:

1. Convergence tolerance—The two-norm of the Kuhn–Tucker conditions was used as the convergence criterion and the specific value for the convergence tolerance varied depending on the application. The particular values used in these numerical experiments are explicitly given with other problem-specific information in the appropriate places in the manuscript.

2. Quadratic programming algorithms—Two methods were used to solve the sparse, iterative QP subproblems in Step 4 of the algorithm. These were VE06AD, which is the sparse, LP-based QP algorithm available in the Harwell Subroutine Library, and the active set method described in Section 3.3, which is based on a symmetric, indefinite factorization.

3. Stabilization procedures—New estimates of the unknown variables in Step 6 of the algorithm were computed by using the trust region strategy described in Section 4.3 without line searching.

4. Hessian matrix approximations—Several methods were used to build approximations to the Hessian matrix of the Lagrangian function. The methods are described below in the context of splitting the Hessian matrix into a computed and approximated part [i.e. equation (29)].

(a) Newton's method (N): the iterative blocks of \( C \) were calculated from analytical second derivatives; the blocks of \( A \) were calculated using either analytical or finite difference derivatives, depending on the application. The initial Hessian matrix was approximated by evaluating both \( C \) and \( A \) using the initial estimates of the unknown variables and multipliers.

(b) BFGS method: the Hessian matrix of the Lagrangian function in the full space of the unknown variables was approximated using the modified BFGS update. The Hessian matrix approximation was initialized in two ways; \( B_0 = I \) and \( B_0 = C(x_0, \lambda_0, \mu_0) \), where \( x_0, \lambda_0 \) and \( \mu_0 \) are the initial values of the unknown variables, Lagrange multipliers and Kuhn–Tucker multipliers, respectively.

(c) Thermodynamically constrained hybrid methods (TCH): the blocks of \( C \) were calculated analytically and each block of \( A \) was calculated using equations (43) and (44). Two choices of weighting matrix were used:
$W_f = I$ and $W_r$ calculated from application-dependent information. In the latter case, these choices are described in the appropriate places along with the associated thermodynamic constraints. In all cases the initial blocks of $C$ were evaluated analytically, while those of $A$ were set equal to the null matrix.

(d) Secant-only hybrid method (SOH): the blocks of $C$ were calculated analytically and each block of $A$ was calculated using equation (35). The matrices $C$ and $A$ were initialized in exactly the same way as they were for the TCH method.

5. Jacobian matrix of the constraints—The Jacobian matrix of the constraint functions was computed analytically.

6. Physical properties—The physical properties (i.e. activity and fugacity coefficients, enthalpies, etc.) that were needed to evaluate $f(x)$, $c(x)$, $J(x)$, $\nabla^2 L(x, \lambda, \mu)$ were calculated using the procedures and data given in Prausnitz et al. (1980) unless otherwise stated.

All calculations were done on both a Gould 9080 and an Alliant FX/8 computer using double-precision arithmetic.

6.2. Extrema of thermodynamic potential functions

The first class of problems that were studied were minimum Gibbs free energy and maximum entropy calculations for two- and three-phase equilibrium, chemical reaction equilibrium, and simultaneous phase and chemical equilibrium. This class of problems is often a common subproblem for larger chemical process optimization problems, and has some interesting characteristics. The objective function and, where present, the energy balance constraint are frequently strongly nonlinear, particularly for liquid–liquid (LLE), vapor–liquid–liquid (VLLE) and chemical equilibria. Singularities in the projected Hessian matrix can occur at azeotropic points, plait points and phase boundaries. Finally, the number of components and phases in the system are $n_c$ and $n_p$, respectively. Finally, $T_{\text{min}}$ and $T_{\text{max}}$ represent the lower and upper bound on the equilibrium temperature.

Usually the feed conditions (i.e. feed component molar flows, temperature, pressure) and the pressure of the vessel are specified. As a result, only one additional variable can be specified to fix the equilibrium state of the system. When this additional specification is the equilibrium temperature, the optimization problem is a minimum Gibbs free energy calculation. That is, the terms $-\Delta H$ in the objective function expression, the energy balance equation and the temperature bounds are dropped and the quantity $G/RT$ is minimized subject to the linear mass balance constraints and the nonnegativity conditions. If, on the other hand, the heat duty $Q$ (or equivalently the total product enthalpy) is specified, then all of the equations and inequalities are required, $T$ becomes an unknown variable, and the objective function is the quantity $-S/R$, where $S$ denotes the total system entropy. Furthermore, this problem is non-linearly constrained because of the presence of the energy balance equation. Finally, for systems that involve chemical reactions, the reaction stoichiometry is reflected in the Jacobian matrix of the mass balance constraints and $\Delta H$ accounts for enthalpy effects due to reaction.

In all cases, the nonnegativity constraints given by equation (48) were replaced by the trust region inequalities defined by equation (24) with $\beta_1 = \sqrt{2}/2$ and $\beta_2 = \sqrt{2}$ for the phase equilibrium calculations and $\beta_1 = 0.95$ and $\beta_2 = 1$ for the chemical equilibrium problems. These values were adjusted automatically to avoid ill-defined QP subproblems. Also, Newton’s method with complete analytical derivatives was used. The thermodynamic constraints that were exploited in the hybrid methods were the set of Gibbs–Duhem equations for the equilibrium phases. In this case, the $n_c \times 1$ vectors $z_i$ and $r_i$ in equation (37) are:

$$z_i = n_i^+, \quad r_i = 0, \quad k = 1, \ldots, n_p$$

and $A$, denotes the $n_c \times n_p$ matrix of activity coefficient or fugacity coefficient-component flow rate derivatives associated with the $i$th equilibrium phase.
6.2.1. Vapor–liquid equilibrium (VLE). Both minimum Gibbs free energy (G/RT) and maximum entropy (S/R) calculations for ideal and nonideal mixtures were solved. The mixtures that were studied are shown in Table 1. In all cases, the feed component flow rates were generated randomly and the feed was assumed to be at its bubble point at the flash pressure, which was 1,013 x 10^5 Pa. The remaining specification was chosen in the following ways. For the Gibbs energy calculations, the temperature was chosen randomly between the bubble point and dewpoint temperatures of the feed. For the entropy calculations, the heat duty specification was given by:

\[ Q = \alpha \left( \sum x_i \right) \left( H_{SV}^V - H_{SL}^L \right). \]  

(51)

where \( \alpha \) is a random number, and \( H_{SV}^V \) and \( H_{SL}^L \) are the molar enthalpies of the feed at saturated vapor and saturated liquid conditions, respectively. Finally, the convergence tolerance was \( \epsilon = 10^{-3} \) for the Gibbs energy calculations and \( \epsilon = 10^{-1} \) for the entropy calculations.

The starting values for the unknown variables were determined as follows. For the Gibbs free energy calculations, the liquid molar flows were calculated from:

\[ n^L = 10^{-2} \left( \sum x_i \right) x_{dp}. \]  

(52)

where \( x_{dp} \) is the dewpoint composition of the feed, and the superscript \( L \) denotes liquid. The vapor component molar flow rates were back-calculated from the component mass equations.

For the entropy calculations, a vapor-to-feed fraction, \( \psi = V/F \), was calculated from the equation:

\[ \psi = Q \left( \sum x_i \right) \left( H_{SV}^V - H_{SL}^L \right) \]

\[ + \left( H^V - H_{SV}^V \right) \left( H_{SL}^L - H_{SL}^V \right). \]  

(53)

The vapor or liquid component molar flow rates were calculated by:

\[ n^V = \psi \left( \sum x_i \right) x_{dp}, \]  

(54)

\[ n^L = (1 - \psi) \left( \sum x_i \right) x_{dp}, \]  

if \( \psi \leq 0.5 \),

if \( \psi > 0.5 \),

where \( x_{dp} \) is the bubble point composition. The remaining component molar flows were back-calculated from the mass balance constraints and the temperature was computed from:

\[ T = \begin{cases} T_{dp} + \psi (T_{dp} - T_{bp}), & \text{if } \psi \leq 0.5 \\ T_{dp} - (\psi - 1) (T_{dp} - T_{bp}), & \text{if } \psi > 0.5 \end{cases} \]  

(55)

Note that the energy balance equation will not be satisfied with this starting strategy.

The multipliers were initialized by setting all Kuhn–Tucker multipliers to zero. The Lagrange multipliers associated with the mass balance equations were calculated from:

\[ \lambda_i = \sum_{k=1}^{n} \frac{G_{ik}}{\eta_k}, \quad j = 1, \ldots, n_c. \]  

(56)

The Lagrange multiplier for the energy balance equation was computed by:

\[ \lambda = -1/RT. \]  

(57)

which can be deduced from an application of the Gibbs–Helmholtz equation to the necessary conditions for the maximum entropy problem. Initializing the Lagrange multipliers in this way is equivalent to finding a least squares solution to the initial Kuhn–Tucker conditions.

Numerical results for 100 random problems for each mixture are shown in Tables 2 and 3.

6.2.2. Vapor–liquid–liquid equilibrium (VLLLE). Minimum Gibbs free energy and maximum entropy problems for 10 nonideal mixtures that exhibit threephase behavior were solved. The multicomponent mixtures, feed conditions and specifications for these problems are shown in Table 4. Problems 1–8 in that table were taken from Wu and Bishnoi (1986). Problem 9 is one of several single-stage subproblems in a heterogeneous distillation problem studied by Ferraris and Morbidelli (1982). Finally, problem 10 is one that we have created which involves a mixture of hydrocarbons, aromatics and water. In all cases, the liquid phases were modelled by the UNIQUAC equation and the vapor phase was modelled by the B-truncated virial equation of state.

The VLLLE problems were solved by first solving an appropriately determined LLE subproblem and then using those results and additional bubble and dewpoint calculations to initialize the three-phase calculations. For Gibbs energy minimization calculations, the component molar flow rates for the two liquid phases \( n^L_i \) and \( n^L_j \) were initialized by identifying the dominant immiscible pair and using the condition \( n^L_i = 0.99 n^L_j \), where \( i \) denotes either of the immiscible components. The flow rate of the other immiscible component was determined from \( n^L_i = 0.01 n^L_j \). Each of the distributed component flow rates was initialized by \( n^L_i = \alpha_i f_i \), where \( \alpha_i \) is a measure of the solubility of that distributed component in the dominant immiscible component in liquid phase 1. The second liquid phase component flow rates were back-calculated using the mass balance equations and an LLE.

Table 1. Mixtures for VLE calculations

<table>
<thead>
<tr>
<th>Mixture</th>
<th>Species</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>nC10, nC10</td>
</tr>
<tr>
<td>2</td>
<td>nC10, nC10, nC10, nC10</td>
</tr>
<tr>
<td>3</td>
<td>iC1, nC1, hexene-1, methylcyclopentane, nC10, nC10</td>
</tr>
<tr>
<td>4</td>
<td>C1, C10, nC10, isoprene, cyclopentane, iC1, nC10, benzene, cyclohexane, hexene-1, methylcyclopentane, nC10, toluene, nC10, nC10, nC10</td>
</tr>
<tr>
<td>5</td>
<td>Ethanol, nC10</td>
</tr>
<tr>
<td>6</td>
<td>Methanol, ethanol, acetone, water</td>
</tr>
<tr>
<td>7</td>
<td>Carbon tetrachloride, chloroform, methanol, ethanol, acetone, methylcyclopentane, benzene, toluene</td>
</tr>
</tbody>
</table>
Gibbs free energy minimization problem was solved for the given temperature and pressure specifications.

The corresponding three-phase Gibbs energy calculations were initialized in the following way. The bubble and dewpoint temperatures and compositions for both liquid phases from the LLE subproblem solution were calculated. The vapor molar flow rates were initialized by:

\[ n^v = 10^{-2} \left( \sum f_i \right) y_{bp}, \]  

(58)

where \( y_{bp} \) is the bubble point composition of either equilibrium liquid phase, since the bubble point conditions are identical (see Kingsley, 1986). The liquid molar flows were computed from:

\[ n^L = 0.99 n^L_{LE}, \quad n^L_{LE} = 0.99 n^L_{LE}, \]  

(59)

where \( n^L_{LE} \) and \( n^L_{LE} \) denote the solution to the LLE subproblem. The results for the LLE subproblems and VLE Gibbs free energy calculations are shown in Table 5.

For the three-phase entropy maximization calculations, the liquid phase component flow rates for the LLE subproblems were initialized in exactly the same way as in the Gibbs energy calculations. The temperature was set equal to the bubble point temperature of the feed and the Kuhn–Tucker and Lagrange multipliers were initialized using equations (56) and (57). The results of the three-phase maximum entropy calculations are given in Table 6.

6.2.3. Chemical equilibrium. Six single-phase chemical equilibrium and one simultaneous chemical and two-phase equilibrium Gibbs free energy minimization problems were also solved. These problems were taken from the literature and are briefly sketched in Table 7. Complete problem information (i.e. reaction stoichiometry, free energies of formation, etc.) can be found in the appropriate references shown in that table. The convergence tolerance for the single-phase chemical equilibrium and simultaneous phase and chemical equilibrium problems were \( \epsilon = 10^{-6} \) and \( \epsilon = 10^{-5} \), respectively.

For the single-phase chemical equilibrium problems, the species mol numbers were initialized by finding a least-square solution to the set of mass balance equations and nonnegativity constraints. That is, the problem:

\[ \min \left| J_n n - \mathbf{f} \right|_2; \quad n \geq 0, \]  

(63)

is solved for the vector \( n \).

For the simultaneous phase and chemical equilibrium problem, a reaction-only chemical equilibrium problem at the given specifications. These calculations were initialized using a least-squares solution to the mass balance equations and nonnegativity constraints. The simultaneous phase and chemical
equilibrium Gibbs free energy calculations were initialized by:

$$n^V = 0.5 \left( \sum n_i^{RO} \right) x^{RO}_{BF}, \quad (64)$$

$$n^L = 0.5 \left( \sum n_i^{RO} \right) x^{RO}_{LP}, \quad (65)$$

where the superscript $RO$ denotes reaction-only, the vector $n_i^{RO}$ is the equilibrium solution of the reaction-only problem, and $x^{RO}_{BF}$ and $x^{RO}_{LP}$ are the bubble and dewpoint composition of the reaction-only solution. The initial Lagrange and Kuhn–Tucker multipliers were calculated from equation (56). The results for these calculations are shown in Table 8. The results for these calculations are shown in Table 8. The results for these calculations are shown in Table 8.

### Table 4. VLLE problem specifications

<table>
<thead>
<tr>
<th>Mixture No.</th>
<th>Species</th>
<th>Flow (kmol h⁻¹)</th>
<th>$p$ (10⁵ Pa)</th>
<th>$T$ (K)</th>
<th>$Q$ (10⁶ J h⁻¹)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Acetonitrile</td>
<td>10.73</td>
<td>1.013</td>
<td>344.90</td>
<td>1.98791</td>
</tr>
<tr>
<td></td>
<td>Water</td>
<td>30.04</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Acetonitrile</td>
<td>15</td>
<td>0.627</td>
<td>333.78</td>
<td>1.36384</td>
</tr>
<tr>
<td></td>
<td>Acrylonitrile</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Acetonitrile</td>
<td>5</td>
<td>0.900</td>
<td>341.27</td>
<td>1.40504</td>
</tr>
<tr>
<td></td>
<td>Water</td>
<td>60</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Acrylonitrile</td>
<td>35</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Furfural</td>
<td>40</td>
<td>0.098</td>
<td>300.30</td>
<td>1.38129</td>
</tr>
<tr>
<td></td>
<td>Benzene</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cyclohexane</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>224-TMP</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Furfural</td>
<td>40</td>
<td>0.083</td>
<td>299.65</td>
<td>1.61754</td>
</tr>
<tr>
<td></td>
<td>Benzene</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>224-TMP</td>
<td>40</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Furfural</td>
<td>30</td>
<td>0.099</td>
<td>300.65</td>
<td>1.67307</td>
</tr>
<tr>
<td></td>
<td>Cyclohexane</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>224-TMP</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Ethanol</td>
<td>5</td>
<td>0.176</td>
<td>298.95</td>
<td>1.66794</td>
</tr>
<tr>
<td></td>
<td>Benzene</td>
<td>45</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Water</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Ethanol</td>
<td>10</td>
<td>0.179</td>
<td>298.38</td>
<td>1.18602</td>
</tr>
<tr>
<td></td>
<td>Benzene</td>
<td>40</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Water</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Acetonitrile</td>
<td>51.97</td>
<td>1.013</td>
<td>344.07</td>
<td>−11.8489</td>
</tr>
<tr>
<td></td>
<td>Water</td>
<td>358.87</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Hydrogen</td>
<td>0.1</td>
<td>1.013</td>
<td>270.00</td>
<td>4.20199 × 10⁻³</td>
</tr>
<tr>
<td></td>
<td>Nitrogen</td>
<td>0.3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Carbon monoxide</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Methane</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ethanol</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Butene-1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>n-butane</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>i-butane</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Isoprene</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>i-pentane</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>n-pentane</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Benzene</td>
<td>67</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>n-hexane</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The temperature and pressure of the feed were set equal to the specified flash temperature and pressure, respectively.

### Table 5. Numerical results for VLLE Gibbs free energy minimization calculations

<table>
<thead>
<tr>
<th>Mixture No.</th>
<th>N</th>
<th>$B_0 - I$</th>
<th>$B_0 - C(s_0)$</th>
<th>TCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11 (11)</td>
<td>F (17)</td>
<td>32 (18)</td>
<td>16 (13)</td>
</tr>
<tr>
<td>2</td>
<td>10 (10)</td>
<td>F (21)</td>
<td>25 (23)</td>
<td>14 (11)</td>
</tr>
<tr>
<td>3</td>
<td>10 (9)</td>
<td>44 (17)</td>
<td>24 (20)</td>
<td>18 (10)</td>
</tr>
<tr>
<td>4</td>
<td>10 (9)</td>
<td>40 (12)</td>
<td>27 (23)</td>
<td>15 (10)</td>
</tr>
<tr>
<td>5</td>
<td>11 (10)</td>
<td>35 (20)</td>
<td>28 (23)</td>
<td>16 (11)</td>
</tr>
<tr>
<td>6</td>
<td>11 (7)</td>
<td>36 (16)</td>
<td>26 (17)</td>
<td>13 (8)</td>
</tr>
<tr>
<td>7</td>
<td>10 (9)</td>
<td>35 (12)</td>
<td>26 (14)</td>
<td>20 (12)</td>
</tr>
<tr>
<td>8</td>
<td>8 (10)</td>
<td>38 (15)</td>
<td>23 (16)</td>
<td>18 (15)</td>
</tr>
<tr>
<td>9</td>
<td>5 (10)</td>
<td>42 (30)</td>
<td>8 (21)</td>
<td>9 (12)</td>
</tr>
<tr>
<td>10</td>
<td>F (F)</td>
<td>F (F)</td>
<td>F (F)</td>
<td>13 (20)</td>
</tr>
</tbody>
</table>

$F$ denotes failure.
Newton and hybrid methods are more reliable than the BFGS update on this class of problems, particularly on the LLE and VLLE problems. The BFGS update sometimes converges to the trivial solution on problems that are based on generic problem information and appropriate bubble and dew point calculations. Furthermore, even though these starting points may sometimes require the solution of a subproblem, as in the case of VLE equilibria or simultaneous phase and chemical equilibrium, this effort is usually well invested.

The other principal factor effecting the reliability of Newton-like methods on this class of problems is the stabilization procedures that are used. The numerical results in Tables 2, 3, 5, 6 and 8 clearly show that simple trust region strategies can give reliable numerical performance and that line searching is generally unnecessary on this class of problems. We solved all problems without line searching and in no case in which failure occurred was that failure avoided by line searching. To illustrate the point, we removed the trust region inequalities, reinstated all of the VLE Gibbs free energy problems for the mixtures in Table 1 using the Han–Powell line searching procedure. The associated numerical results are shown in Table 7.

Table 7. Problem specifications for chemical equilibrium problems

<table>
<thead>
<tr>
<th>Mixture No.</th>
<th>T (K)</th>
<th>p (10^5 Pa)</th>
<th>Solution model</th>
<th>Species</th>
</tr>
</thead>
<tbody>
<tr>
<td>1[4.8]</td>
<td>1000</td>
<td>1.013</td>
<td>Ideal vapor</td>
<td>H₂, O₂, CO, CH₄, CO₂, C₂H₆, C₃H₈, C₄H₁₀, H₂O</td>
</tr>
<tr>
<td>2[4.10]</td>
<td>358</td>
<td>0.912</td>
<td>Nonideal liquid</td>
<td>CH₃COOH, C₃H₇OH, CH₃COOCH₂H₂H₂O</td>
</tr>
<tr>
<td>3[6.11]</td>
<td>3500</td>
<td>51.06</td>
<td>Ideal vapor</td>
<td>H, H₂, H₂O, N, N₂, NH, NO, O₂, OH</td>
</tr>
<tr>
<td>4[6.6]</td>
<td>600</td>
<td>2.026</td>
<td>Ideal vapor</td>
<td>C₁₇H₁₆, C₁₄H₁₀, C₆H₁₂, CH₂Cl₃, CH₂Cl₂, CHCl</td>
</tr>
<tr>
<td>5[6.7]</td>
<td>3000</td>
<td>20.69</td>
<td>Ideal vapor</td>
<td>H, H₂, H, Cl, C₂H₂Cl₂, F₂, N₂, C₂H₂OH, H₂, S, C₃H₂SH, (C₃H₆), S, H₂S, (C₃H₆)₂O, CH₂CHO, C₃H₁₀, C₅H₁₂, H₂O</td>
</tr>
<tr>
<td>6[6.8]</td>
<td>600</td>
<td>81.04</td>
<td>Ideal vapor</td>
<td>C₃H₁₀OH, H₂, S, C₃H₂SH, (C₃H₆), S, H₂S, (C₃H₆)₂O, CH₂CHO, C₅H₁₂, H₂O</td>
</tr>
<tr>
<td>7[1]</td>
<td>358</td>
<td>1.013</td>
<td>Nonideal vapor, liquid</td>
<td>CH₃COOH, C₃H₇OH, CH₃COOCH₂H₂H₂O</td>
</tr>
</tbody>
</table>

1Number in brackets denotes problem number in Smith and Misser (1982).
2Problem from Sanderson and Chien (1973).
for various aspects of the chemical equilibrium minimum Gibbs free energy calculations and the VLE maximum entropy problems are given. Note that the cost of solving the sparse QP subproblems using VEO6AD accounts for roughly 70-90% of the total time and this agrees with statements made by other researchers. On the other hand, the same problems can be solved in significantly less computer time if the active set method proposed in Section 3.3 is used. The total solution time using QPSLV is roughly 5-15% of that for VEO6AD and this reduction in total solution time is primarily due to the fact that the proposed active set method usually requires less than 10% of the time required by VEO6AD to solve the QP subproblems.

Chan and Prince (p. 238, 1986) note that the efficiency of SQP methods depends on both the efficiency of the QP solver and the number of iterations (i.e., QP subproblems) needed to solve a given problem. They also state that both aspects are equally important. The numerical results shown in Tables 2, 3, 5, 6 and 8 show that Newton and hybrid methods frequently require fewer iterations than the BFGS method. Thus they are often more efficient than the BFGS method. Moreover, the results in Table 2, for example, show that the number of iterations required by the BFGS method can increase as the number of degrees of freedom increases, and that this can be independent of the initial Hessian approximation.

Thus, for efficient calculations, we recommend Newton and hybrid methods on this class of problems.

6.3.1. Mathematical model. The mathematical model for these problems usually involves optimizing some measure of profitability subject to constraint equations representing the conservation of mass, conservation of energy, phase equilibrium and design constraints and inequalities that represent the nonnegativity of flow rates, limits on the range of operation and bounds on product flow rates. In this study, we posed the optimization problem as:

\[ \min 3 \times 10^{-4}(-Q_1 + Q_m) - (v_{h,i} + l_{h,i}), \] (66)

subject to the conservation of mass

\[ f_i + l_{i-1} - l_{y} - v_{y} + v_{y-1} = 0, \quad i = 1, \ldots, n_c, \] (67)

phase equilibrium relationships

\[ K_q(l_q)(\sum l_q) - (v_q)(\sum v_q) = 0, \quad i = 1, \ldots, n_c, \] (68)

Table 10. Computer times$^1$ for chemical equilibrium problems

<table>
<thead>
<tr>
<th>Mixture No.</th>
<th>VE06AD</th>
<th>Time$^2$ (s)</th>
<th>QPSLV</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.006, 0.027, 2.894, 0.023, 3.051</td>
<td>0.006, 0.027, 0.296, 0.023, 0.552</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.003, 0.016, 0.324, 0.029, 0.433</td>
<td>0.003, 0.016, 0.021, 0.029, 0.137</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.006, 0.032, 4.500, 0.035, 4.712</td>
<td>0.006, 0.032, 0.034, 0.035, 0.617</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.004, 0.028, 2.803, 0.021, 2.986</td>
<td>0.004, 0.028, 0.219, 0.021, 0.415</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.005, 0.032, 5.757, 0.038, 5.848</td>
<td>0.006, 0.032, 0.548, 0.038, 0.842</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.006, 0.039, 6.101, 0.042, 6.411</td>
<td>0.006, 0.039, 0.705, 0.038, 1.039</td>
<td></td>
</tr>
</tbody>
</table>

$^1$Calculations using Newton's method on Alliant FX/8.

$^2$Computer times for (initialization, function calls, QP solutions, second derivative calculations, problem solution), respectively.
Table 11. Computer times\(^1\) for VLLE maximum entropy problems

<table>
<thead>
<tr>
<th>Mixture No.</th>
<th>VE06AD</th>
<th>QPSLV</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(2.999, 0.451, 9.447, 0.469, 13.968)</td>
<td>(0.855, 0.443, 0.438, 0.468, 2.833)</td>
</tr>
<tr>
<td>2</td>
<td>(3.039, 0.219, 4.177, 0.216, 8.142)</td>
<td>(0.744, 0.219, 0.240, 0.216, 1.923)</td>
</tr>
<tr>
<td>3</td>
<td>(2.400, 0.286, 5.046, 0.292, 8.544)</td>
<td>(0.707, 0.286, 0.393, 0.295, 2.217)</td>
</tr>
<tr>
<td>4</td>
<td>(4.090, 0.273, 7.376, 0.386, 13.153)</td>
<td>(0.851, 0.251, 0.327, 0.386, 2.635)</td>
</tr>
<tr>
<td>5</td>
<td>(2.557, 0.160, 3.158, 0.202, 7.082)</td>
<td>(0.612, 0.161, 0.183, 0.202, 1.951)</td>
</tr>
<tr>
<td>6</td>
<td>(1.914, 0.143, 2.546, 0.142, 5.208)</td>
<td>(0.608, 0.143, 0.147, 0.143, 1.518)</td>
</tr>
<tr>
<td>7</td>
<td>(2.511, 0.186, 3.321, 0.181, 6.686)</td>
<td>(0.622, 0.177, 0.181, 0.178, 1.646)</td>
</tr>
<tr>
<td>8</td>
<td>(2.481, 0.169, 2.848, 0.161, 6.130)</td>
<td>(0.630, 0.164, 0.161, 0.160, 1.602)</td>
</tr>
<tr>
<td>9</td>
<td>(3.610, 0.382, 7.851, 0.272, 12.687)</td>
<td>(0.814, 0.374, 0.381, 0.271, 2.416)</td>
</tr>
<tr>
<td>10</td>
<td>(16.416, 1.631, 48.721, 1.941, 70.919)</td>
<td>(3.432, 1.621, 2.194, 1.941, 11.969)</td>
</tr>
</tbody>
</table>

\(^1\) Calculations using TCH method on Alliant FX/8.

Problem No. (reference) | No. of stages | Column pressure (MPa) | Feed at col pressure flow rates (kmol h\(^{-1}\)) | Specifications for starting point | Product bounds (kmol h\(^{-1}\))
--- | --- | --- | --- | --- | ---
1 (King, 1980) | 5 | 0.1216 | Saturated vapor, stage 5
(1k) benzene 20
(1l) toluene 40
(1h) o-xylene 40 | \(D = V_1 = 20\)
\(R = 5\) | None |
2 (King, 1980) | 8 | 0.1013 | Saturated liquid, stage 5
(1k) acetone 50
(1h) water 50 | \(d_a = 49\)
\(R = 0.288\) | \(d_a \leq 49.9\) | None |
3 (Van Winkle, 1967) | 10 | 0.1013 | Saturated liquid, stage 6
(1k) n-heptane 42
(1h) ethylbenzene 58 | \(x_{sa} = 0.99\)
\(R = 2.5\) | None |
4 (Naka et al., 1979) | 17 | 0.1013 | Saturated liquid, stage 9
(1k) acetone 10
(1l) acetonitrile 75
water 15 | \(B = 70\)
\(R = 3\) | None |
5 (Gallun and Holland, 1976) | 16 | 0.1013 | Liquid, \(T_r = 331.2\ K\), stage 5
(1k) methanol 15
(1h) chloroform 20
(1l) acetone 40
acetate 8 methylacetate 5
benzene 20 | \(B = 62\)
\(R = 2.5\) | \(d_a \leq 14.98\) | None |
6.3.3. Thermodynamic constraints. The thermodynamic constraints that were exploited in building quasi-Newton updates to the blocks of \( A \) were:

\[
[V^2 \ln \gamma_i]_j = -\ln \gamma_i, \quad i = 1, 2, \ldots, n_c,
\]

\[
[V^2 H_{ij}]_j = 0, \quad j = 1, 2, \ldots, n_c, \tag{74}
\]

and

\[
[V^2 H_{ij}]_j = 0, \quad j = 1, 2, \ldots, n_s, \tag{75}
\]

where \( \gamma_i \) is the \( i \)th component activity coefficient on the \( j \)th stage in the column, \( H_{ij} \) is the excess liquid molar enthalpy and \( i \) is a vector of liquid component flow rates associated with stage \( j \). These equations represent the \(-1 \) degree homogeneity of \( V \ln \gamma_i \) and the zero degree homogeneity of \( H_{ij} \). A complete description of the thermodynamic and secant constraints used in quasi Newton updating can be found in Kumar and Lucia (1987).

6.3.4. Numerical examples and results. Newton's method with finite difference approximations to the blocks of \( A \), a secant-only hybrid method and a thermodynamically constrained hybrid method were tested on the five distillation examples studied by Lucia and Kumar (1988). Complete problem specifications for the simulation and optimization calculations are given in Table 12, which was taken from Kumar and Lucia (pp. 416, 421, 1987). Also, the liquid phase was modelled by the UNIQUAC equation, the vapor phase was assumed to be ideal and the convergence tolerance for the Kuhn-Tucker conditions was \( 10^{-3} \).

Finally, the nonnegativity constraints were replaced by the bounds given by equation (24) with \( \beta_1 = \sqrt{2}/2 \) and \( \beta_2 = \sqrt{2} \) and these bounds were enlarged automatically to prevent ill-defined QP subproblems. These distillation examples range in size from 37 to 178 unknown variables and are strongly nonlinearly constrained due to the presence of the phase equilibrium and energy balance equations.

A summary of the numerical results for Newton's method and both hybrid methods is given in Table 13. A comparison of the performance of VE06AD and QPSLV on these problems is shown in Table 14, along with other pertinent information that has bearing on the computational efficiency of these Newton-like methods.

6.3.5. Discussion of numerical results. Lucia and Kumar (1988) experienced numerical difficulties in solving some of the example problems in Table 12 by SQP methods. In particular, both Newton's method and the secant-only hybrid method failed on two of the five problems. Newton's method failed on problems 2 and 4, while the secant-only hybrid method failed on problems 4 and 5. In both cases, the secant-only hybrid method failed because certain component flow rates activated lower bounds and resulted in line searching difficulties or uphill search directions. Newton's method, on the other hand, seemed to be converging in both cases on which it failed but was still relatively far from the solution when the calculations were terminated after 25 iterations. It is likely that Newton's method would have succeeded on problem 4 if the calculations were continued. However, on problem 2, although the unknown variables were moving toward the solution, it is, in our opinion, unlikely that Newton's method would have converged on this problem because gross qualitative errors in the signs and magnitudes of several of the Lagrange multipliers (i.e. incorrect sign and two orders of magnitude too large) occurred on iteration 4 of the calculations. These errors, particularly those in sign, persisted throughout the remainder of the calculations and, in our opinion, contributed to both poor Hessian matrix approximations and doddering numerical performance. The thermodynamically constrained hybrid method did not experience any of these difficulties.

Reliability—The reliability of Newton-like methods on this class of chemical process optimization problems is strongly dependent on the starting point that is used. Our experience shows that infeasible starting points generally lead to failure and this is the principal reason that we use feasible starting points. These feasible starting points are easily and inexpensively computed, and, in turn, are useful in obtaining qualitatively correct initial estimates of the Lagrange multipliers and good approximations to the Hessian matrix of the Lagrangian function.

The numerical results in Table 13 clearly show that improvements in reliability can result from the use of trust region methods. Both failures of the secant-only hybrid method and one of the two failures of Newton's method reported by Lucia and Kumar (1988) were removed by avoiding wild changes in the unknown variables and/or activating lower bounds with the use of trust regions. This, in turn, avoids poor approximations to the unknown variables and multipliers and/or model discontinuities.

Computational efficiency Significant improvements in the computer time required to solve the distillation examples can be obtained if the QP algorithm given in Section 3.3 is used instead of VE06AD. This is clearly shown in Table 14. Note that QPSLV requires approx. 1–5% of the time required by VE06AD. Also, note that the relative cost of generating a feasible starting point is insignificant, particularly in light of its impact on the reliability of Newton-like methods on this class of chemical process optimization problems.

Finally, we close by reiterating that, in our opinion, careful attention must be paid to the variable and multiplier initialization strategies, the QP algorithm,
Chemical process optimization

Table 14. Computer times\(^a\) for distillation optimization problems

<table>
<thead>
<tr>
<th>Problem No.</th>
<th>VE06AD</th>
<th>Time (s)(^b)</th>
<th>QPSLV</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0.866, 4.109, 108.6, 7.382, 136.1)</td>
<td>(0.866, 4.112, 5.273, 7.379, 27.831)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>(1.732, 3.464, 105.2, 3.595, 117.5)</td>
<td>(1.732, 3.514, 4.121, 3.597, 17.621)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(1.461, 4.612, 132.7, 5.280, 161.2)</td>
<td>(1.461, 4.612, 5.308, 5.271, 21.633)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(3.566, 6.714, 2330.9, 8.112, 2354.1)</td>
<td>(3.566, 6.694, 28.174, 8.091, 53.124)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(8.574, 11.284, 9361.2, 15.616, 9418.3)</td>
<td>(8.574, 11.292, 280.82, 15.615, 325.16)</td>
<td></td>
</tr>
</tbody>
</table>

\(^a\)Calculations using TCH method on Alliant FX/8.
\(^b\)Computer times for initialization, function calls, QP solutions, second derivative calculations, problem solution, respectively.

stabilization procedures and techniques for approximating the Hessian matrix of the Lagrangian function, if consistently reliable and efficient numerical performance of SQP methods is to be achieved on chemical process optimization applications. It is clearly a case of getting what you pay for. We hope that the ideas and results that were presented in this paper motivate others to move in this direction.

NOMENCLATURE

- \(A\) = Approximated part of Hessian matrix
- \(B\) = Hessian matrix of Lagrangian function
- \(c(x)\) = Vector of constraint equations and inequalities
- \(C\) = Computed part of Hessian matrix
- \(d\) = Solution from QP subproblem
- \(f, f(x)\) = Vector of feed component molar flow rates, objective function
- \(g(x)\) = Gradient of objective function
- \(G, G\) = Partial molar Gibbs free energy, total Gibbs free energy
- \(H, H_x\) = Partial molar enthalpy, enthalpy of reaction
- \(J(x)\) = Jacobian matrix of constraint functions
- \(L(x, \lambda, \mu)\) = Lagrangian function
- \(n\) = Vector of phase component molar flow rates
- \(q\) = Transformation of the vector \(s\)
- \(Q\) = Heat duty
- \(r\) = Right-hand-side of thermodynamic constraints
- \(s\) = Iterative change in unknown variables
- \(S\) = Total entropy
- \(T\) = Temperature
- \(u\) = Transformation of the vector \(z\)
- \(W\) = Weighting matrix
- \(x\) = Vector of unknown variables
- \(y\) = Right-hand-side of secant condition
- \(z\) = Vector defining thermodynamic constraints

Greek symbols

- \(\alpha\) = Random number between 0 and 1
- \(\epsilon\) = Convergence tolerance
- \(\eta\) = Vector in modified BFGS formulae
- \(\theta\) = Scalar in defining equation for \(\eta\)
- \(\lambda\) = Lagrange multipliers
- \(\mu\) = Kuhn–Tucker multipliers
- \(\psi\) = Vapor-to-feed fraction

Subscripts and superscripts

- \(BP\) = Bubble point
- \(DP\) = Dew point
- \(F\) = Feed
- \(I\) = Component index
- \(i\) = Stage index
- \(k\) = Iteration counter or phase index
- \(L\) = Liquid phase
- \(LLE\) = Liquid–liquid equilibria
- \(max\) = Upper bound
- \(min\) = Lower bound
- \(R\) = Reaction
- \(SL\) = Saturated liquid
- \(SV\) = Saturated vapor
- \(T\) = Transpose
- \(V\) = Vapor phase

REFERENCES


