Efficient Implementation of VLE Procedures in Equation-Oriented Simulators

Modern equation-oriented simulators solve systems of equations and procedures using Newton's method or its variants. Partial derivatives of the output variables of procedures with respect to their input variables are required and are usually approximated by perturbation or quasi-Newton methods. Here it is shown how this partial derivative information can be computed exactly for procedures in general, with particular applications to flash and multicomponent distillation procedures. Results indicate that 1. physical property procedures should provide analytical derivatives of properties in addition to point values; 2. commonly used procedures for vapor-liquid equilibria calculation can easily be modified, and should be, to also return output/input gradients; and 3. the use of exact derivatives at all levels results in significant improvements in efficiency over present methods.

S. Macchietto, G. I. Maduabueke, R. Szczepanski
Department of Chemical Engineering
Imperial College
London, SW7 2BY England

Introduction

In the equation-oriented approach to process flowsheeting all mass and energy balances, specifications, design constraints, etc., describing a process are written as a set of equations or a mixed set of equations and procedures to be solved simultaneously (Perkins, 1984). One reason for solving a subset of equations as a procedure is that physical or mathematical knowledge of a particular process unit can be used to devise specifically tailored solution algorithms. For instance, a flash unit procedure could be written using a free energy minimization approach, or the Rachford-Rice tearing method. Many procedures are already available for individual unit operations (distillation, etc).

Process calculations usually involve the repeated use of physical and thermodynamic properties, and so these are also typically computed in procedures from a physical properties package. We therefore identify here three levels of calculations: thermophysical properties level, unit operation level, and flowsheet level as shown in Figure 1. Besides facilitating tailored solution algorithms, the use of procedures gives additional advantages: ease of tracking pathological situations, reduction in the number of variables/equations to be solved simultaneously at the flowsheet level, localization of diagnostic information, and ease of initialization (Perkins, 1984). However it is the simultaneous solution at the flowsheet level which yields the great flexibility typical of the equation-oriented approach. More experience is needed to decide whether and which equation subsets should be isolated and solved as procedures. In the meanwhile a pragmatic approach is to allow for the most efficient use of whatever procedures may already be available.

A solution of mixed systems of equations and procedures at the flowsheet level is typically achieved by Newton's method or its variants (Sargent, 1978). These use at each iteration a linearized model of equations and procedures for which first-order partial derivatives must be obtained. While the required derivatives of analytic equations can be calculated efficiently by symbolic differentiation (Pantelides, 1988), several techniques have been proposed for calculating procedure derivatives: finite-difference, diagonal perturbation, and quasi-Newton or hybrid Newton–quasi-Newton methods (see review in Macchietto, 1985).

The need for derivatives, however, is not restricted to equation-oriented simulators. Simultaneous-modular simulators need the same information (Stadtherr and Chen, 1984), and even the traditional sequential-modular simulators may use procedure gradients, in conjunction with chain-ruling, for optimization calculations (Shivaram and Biegler, 1983). Exact derivatives are also needed in other areas, for example for phase stability analysis (Michelsen, 1982). Stadtherr and Chen state that a finite-differencing method (what they call block perturbation), where each unknown input variable is perturbed in turn by a suitable step, is the most reliable and efficient scheme. However the choice of a step size is always a compromise, since perturbations must be small enough for a linearization to be valid and

Correspondence concerning this paper should be addressed to S. Macchietto.

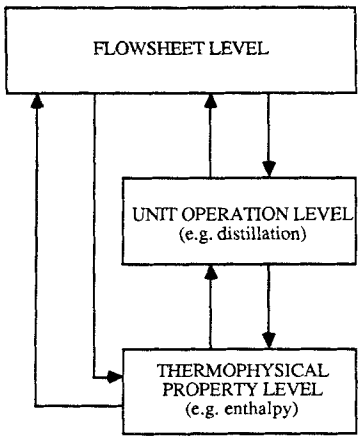

Figure 1. Levels of calculations and information flows.

large enough to eliminate the effects of roundoff and noise on the derivatives. In addition, perturbation can still be expensive when each procedure evaluation results in long and complex calculations, as with multicomponent distillation. Diagonal perturbations are in general not reliable. Quasi-Newton methods can also be used to approximate procedure derivatives. In fact a hybrid method that combines analytic derivatives of equations and quasi-Newton approximations for procedures has been found to be quite efficient for steady state equation-oriented simulation (Pantelides, 1988). However, while quasi-Newton approximations may be good enough for steady state simulation, optimization requires exact gradients (Biegler et al., 1985).

In all these methods it is assumed that procedure derivatives are not available and therefore they must be either approximated or estimated. This results from the fact that most thermodynamic packages and process unit procedures were written for a sequential-modular flowsheeting environment that did not require derivatives. In this paper, rather than looking for yet another clever way of getting around the lack of derivative information from procedures we try to answer to two simple questions: can we easily calculate exact derivatives from procedures of interest, and is this of advantage in a flowsheeting environment? We will try to show in the following that the answer to both questions is an emphatic yes.

Exact Procedure Derivatives

Let us define the set of $m + n$ equations to be solved in a procedure as

$$f(x, u, y) = 0 \tag{1}$$

with $x \in R^{\ell}$ a vector of input variables, $y \in R^{m}$ a vector of output variables, $u \in R^{n}$ a vector of variables internal to the procedure, and assume that some procedure P is available to calculate the solution y (and u) of Eqs. 1 corresponding to a given value of x; that is, $y = y(x)$ and $u = u(x)$. What we are interested in is the matrix $\partial y/\partial x$. This matrix represents the constrained gradients (sensitivities) of the outputs y with respect to the inputs x, that is, a first-order approximation to the change in y for a small change in x subject to Eqs. 1.

A linearized model of the procedure about a base point x_b is then given by:

$$\tilde{f}(x, y) = y - y(x_b) + \left(\frac{\partial y}{\partial x}\right)_b (x - x_b) = 0 \tag{1a}$$

where $(\partial y/\partial x)_b$ is the sensitivity matrix at x_b. The m Eqs. 1a may then be solved simultaneously with other linearized procedure and analytic equations for new values of x, y. The internal variables u do not appear explicitly in Eqs. 1a, however, they are implicitly taken into account via the sensitivity matrix.

If Eqs. 1 are simple enough and the procedure gives us an explicit solution $y = g[u(x), x]$, then exact derivatives are obtained simply by analytic differentiation:

$$\frac{\partial y_i}{\partial x_j} = \frac{\partial g_i}{\partial x_j} + \sum_{k=1}^{n} \frac{\partial g_i}{\partial u_k} \frac{\partial u_k}{\partial x_j} \tag{2}$$

The derivations need to be done only once, manually or with the help of automatic symbolic manipulation programs, and the resulting expressions for the derivatives can be coded together with point value calculations. Although there is still the issue of whether derivatives are calculated more efficiently this way or by numerical perturbation, this is really a trivial case. What we are really interested in is the case when procedure P is iterative in nature and thus an explicit solution of Eqs. 1 is not available. Under the usual assumptions of continuity, a Taylor series expansion of Eqs. 1 about a current solution gives (neglecting higher order terms):

$$f(x + \delta x, u + \delta u, y + \delta y) = f(x, u, y) + Q\delta y + R\delta u + S\delta x \tag{3}$$

where

$$Q \equiv \partial f/\partial y, Q \in R^{m+n,m}$$
$$R \equiv \partial f/\partial u, R \in R^{m+n,n}$$
$$S \equiv \partial f/\partial x, S \in R^{m+n,\ell}$$

(all matrices computed at the current solution).

At a current solution all the residuals $f(x, u, y)$ equal zero. Also, we want the perturbed point to satisfy Eqs. 1; thus we set the term on the lefthand side of Eq. 3 to zero and obtain:

$$Q\delta y + R\delta u + S\delta x = 0$$

If we divide by δx, take the limit for $\delta x \to 0$, and use the definition of partial derivative, we have

$$Q\,\partial y/\partial x + R\,\partial u/\partial x = -S \tag{4}$$

or, in matrix form

$$[Q{:}R]\begin{bmatrix}\partial y/\partial x \\ \partial u/\partial x\end{bmatrix} = -S \tag{5}$$

The desired output-input sensitivities $\partial y/\partial x$ are obtained by solving Eq. 5, that is, a set of $m + n$ linear equations with ℓ righthand sides. The sensitivities of the internal variables with

respect to the inputs, $\partial u/\partial x$, are also obtained in the process. The following tasks will have to be performed in order to secure procedure derivatives through our working Eq. 5:

1. For a given value of x, solve Eqs. 1 according to procedure P to some tight tolerance
2. At the solution found in step 1 generate matrices Q, R, and S from the original Eqs. 1 (obtain derivatives of lower level procedures if required)
3. Obtain the LU factors of $[Q:R]$
4. Solve Eq. 5 for $\partial y/\partial x$ (and $\partial u/\partial x$)

Step 1 could be accomplished using any suitable numerical solution algorithm, which may even involve reformulation or transformation of the original equations. This method of evaluating procedure derivatives is noniterative, the storage requirement is small, the need for perturbations is eliminated, and above all exact procedure derivatives are obtained.

Application to VLE Procedures

In this section we want to assess the efficiency of evaluating procedure derivatives according to the method presented in the preceding section. We consider typical procedures frequently used in vapor-liquid equilibria (VLE) calculations and measure the time required to generate procedure derivatives relative to the time for a single procedure evaluation.

As a representative iterative procedure at the physical property level we chose the Soave-Redlich-Kwong (SRK) equation of state to compute the vapor phase fugacity coefficients for vapor-liquid equilibria. Derivatives of the fugacity coefficients with respect to composition, pressure, and temperature were obtained in two ways: by analytic differentiation (A) and numerical perturbation (P). Care was taken to skip repeated calculations whenever possible, for example in the evaluation of temperature-dependent parameters, and especially when perturbations were performed. Relative times for these derivatives by methods A and P are presented in Table 2 for several mixtures of from five to 16 components, given in Table 1. Also in Table 2 we present the equivalent number of fugacity coefficient calls required to evaluate a full set of $NC + 2$ procedure derivatives by the two methods, where NC is the number of components in the mixture. For example, for mixture C calculating the 10 fugacity coefficient derivatives analytically and by perturbation is equivalent to 2.1 and 6.6 fugacity coefficient point evaluations, respectively. Results in Table 2 indicate that the time for analytic evaluation of derivatives is equivalent to only 2–2.5 point evaluations, is only about one-third of the time required for numerical evaluation, and is roughly independent of the number of components in the mixture. We should add that a very efficient coding of the analytic derivatives was required to obtain these results and that significantly worse performance was achieved when implementation details were neglected.

Table 1. Vapor Phase Mixtures Used for Fugacity Coefficient Tests

Component	Mixture, kmol A	B	C	D
Nitrogen	—	—	451.974	20.036
Carbon dioxide	511.825	1,361.319	6,637.164	1,356.206
Hydrogen sulfide	—	—	—	206.723
Methane	—	2,253.665	3,776.686	456.122
Ethane	361.334	—	2,772.741	1,273.325
Propane	—	782.162	1,510.926	1,341.338
Isobutane	—	—	—	203.049
Butane	90.315	189.435	474.940	387.847
Isopentane	—	—	—	86.455
Pentane	—	—	—	53.733
Hexane	21.088	12.831	113.042	49.759
Heptane	—	—	—	24.515
Octane	—	—	—	6.052
Nonane	—	0.121	—	1.869
Decane	69.503E-3	—	0.278	0.320
Undecane	—	—	—	0.162
Temp., K	322.0	311.0	311.0	309.0
Press., bar	19.0	56.2	56.2	4.39

Table 2. Relative Times for Computation of Fugacity Coefficients (SRK Eq. of State) and Derivatives

	Feed Mixture, No. Components A 5	B 6	C 8	D 16
Fugacity Coeff.	1.0*	1.2	1.6	3.9
Fugacity Coeff. Derivatives				
Analytic (A)	2.0 (2.0)**	2.5 (2.1)	3.4 (2.1)	9.2 (2.4)
Perturbation (P)	5.7 (5.7)	7.1 (5.9)	10.5 (6.6)	32.2 (8.3)
Ratio A/P	0.35	0.352	0.323	0.286

*Actual time for 100 points = 0.060 CPU s on CDC CYBER 855 (Nos. 2, 4)
**Numbers in parentheses are equivalent number of fugacity coefficient calculations

As a representative iterative procedure at the unit operation level an isothermal flash algorithm was coded. The basic equations for the VLE flash problem are very simple but many solution schemes have been proposed. In this work we use a Rachford-Rice type algorithm that combines ideas of Ohanomah and Thompson (1984) and Rohl and Sudall (1967), with minor modifications. The algorithm is iterative, but the iteration loop is very simple and only K values are required, not their derivatives.

From this flash procedure we return the output variables (component flow rates of vapor and liquid streams) and, if requested, their exact derivatives with respect to input values (feed component rates, flash temperature, and pressure) calculated as outlined previously and detailed in Appendix A. The algorithm presented in Appendix A includes a slight modification to reduce the size of the linear system to be solved to NC (number of components in the mixture). This is more efficient than a straightforward application of the method in the previous section. K-value derivatives from lower level thermodynamic procedures are needed for evaluation of the Q and S matrices and were obtained either analytically or by perturbation from the fugacity coefficient procedure described above. The SRK equation of state was used in all cases.

Typical computer times for an isothermal flash solution and derivative generation are given in Table 4 for a five-component mixture (mixture E, Table 3). Convergence of the main iterative loop is fast, with 75% of the base point solution time spent in the evaluation of K values. Setting up matrices Q and S involves getting derivatives from physical property level routines (roughly half to 70% of the total time for flash derivatives) and solving a

Table 3. Feed Mixtures and Flow Conditions for Flash Problems

Component	Mixture, kmol/h				
	E	F	G	H	I
Nitrogen	—	—	457.0	20.5	—
Carbon dioxide	511.92	1,482.4	6,878.0	1,546.9	6.0
Hydrogen Sulfide	—	—	—	304.1	24.0
Methane	—	2,581.8	3,861.0	488.3	66.0
Ethane	361.48	—	2,944.0	1,723.3	3.0
Propane	—	1,164.5	1,710.0	2,891.4	1.0
Isobutane	—	—	—	745.6	—
Butane	90.58	380.4	607.0	1,831.8	—
Isopentane	—	—	—	864.2	—
Pentane	—	—	—	1,180.1	—
Hexane	21.50	63.5	245.0	1,813.5	—
Heptane	—	—	—	2,633.3	—
Octane	—	—	—	1,851.6	—
Nonane	—	3.7	—	1,671.3	—
Decane	0.12	—	5.0	832.1	—
Undecane	—	—	—	1,214.7	—
Temp. K	322.0	311.0	311.0	309.0	225.0
Pres. bar	19.0	56.2	56.2	4.39	60.78

linear system of $NC = 5$ equations for $NC + 2 = 7$ righthand sides. The total time for exact flash procedure derivatives is slightly less and slightly larger than the time for the base point calculation when K-value derivatives are obtained analytically and by perturbation, respectively. The effort for generating flash derivatives with several mixtures is quantified in Table 5 as an equivalent number of base point calculations, that is, taking the time for a base point solution as reference. This being a relative measure, it tends to overestimate the cost of analytic derivatives when the base point solution is achieved in few iterations, which therefore results in a most unfavorable situation for the new method. Even so, analytic derivatives of the flash procedure are calculated significantly faster than by perturbation of the $NC + 2$ inputs to the flash even for the "easy" examples (mixtures E to H). Mixture I in Table 3 represents a more difficult problem in the retrograde region, and converging the flash for each perturbed condition takes on average 9.6 thermodynamic (TD) calls. In this case the time for analytic flash derivatives using our method is 30 times smaller than by perturbation.

Table 4. Breakdown of Computer Times for a Typical Isothermal Flash Solution (Mixture E)

	Time % for K Values Derived			
	Analyt.		By Perturb.	
Flash Calculation	52		40	
Rigorous K-value		75		69
Arithm. Ops.—in iterative loop		25		31
Flash Derivatives	48		60	
Rigorous K-Value + K-Value derivatives		47		70
Generation of matrices and other arithm. ops.				
Linear eq. solver		21		10
Complete Flash Procedure	100*		100**	

*100% = 0.086 CPU's, CDC CYBER 855
**100% = 0.105 CPU's, CDC CYBER 855

Table 5. Equivalent Number of Isothermal Flash Evaluations Required to Generate Output Sensitivities with Respect to all $NC + 2$ Input Variables

	Feed Mixture and No. Components NC				
	E 5	F 6	G 8	H 16	I 5
Exact					
A**	0.9 (2)*	0.7 (2)	0.9 (2)	4.8 (2)	0.1 (2)
B†	1.5 (7)	1.0 (8)	1.1 (10)	8.5 (18)	0.2 (7)
Perturbation	2.7 (22)	2.3 (40)	2.6 (53)	6.3 (37)	3.0 (67)
No. iterations/base point	8	17	20	5	38
No. TD calls/base point	9	18	21	6	39

*() Equiv. no. of thermodynamic (TD) calls for derivatives
**Analytic K value derivatives
†K value derivations by perturbation

As a second representative unit operation procedure we modified a slightly inefficient but widely available standard Naphtali-Sandholm type code for multicomponent distillation (Fredenslund et al., 1977). The algorithm uses Newton's method to solve simultaneously mass and energy balances and equilibrium relations, input variables being all feeds, the rate of all side products, distillate rate, and reflux ratio. A block tridiagonal linear system is solved at each column iteration, and factorization is performed on the diagonal blocks only. Modifications were made to allow for any suitable routine for K values and enthalpy, with or without derivatives being passed from physical property level procedures, and to compute exact sensitivities of the outputs (distillate and bottoms composition and heat loads) with respect to any of the inputs. In order to increase efficiency, in a given run sensitivities are computed only with respect to those input variables that are specifically indicated (unknown variables in a given run). The Q and R matrices needed for generating the procedure derivatives are partitions of the Jacobian matrix used in the main iterative loop for the column, thus the code to generate them is already at hand. The effort for setting up the S matrix is small and in fact exactly the same block tridiagonal code can be used to solve the linear system, Eq. 5, for sensitivities that was used to solve for a correction step in each Newton iteration for the column.

Typical computer times for a single column solution (calculation of output variables and their derivatives with respect to two of the input variables: distillate rate and reflux ratio) are shown in Table 6 for the initial column evaluation of example problem 6 discussed in the next section. Analytic derivatives of all physical properties were used here. Even more than in the isothermal flash case, the effort required to generate column procedure derivatives is minimal. It is roughly equivalent to a single extra iteration of the main convergence loop. Since even the most efficient perturbation would require at least one pass through the loop to determine the sensitivities to each input variable, our method is bound to be very efficient.

Table 6. Computer Times for Initial Column Evaluation of Problem 6 (Method NAA)

	Time CPU	
Base Point Calculation		
Initialization	0.001	
Iterative calc., 6 cols. iterations	0.562	
Thermod. calc., K-values and enthalpy	0.145	
		0.708
Procedure Derivatives Calculation		
Thermod. calc.	0.040	
Solution of linear system	0.117	
		0.157
Total Time for Distillation Procedure		0.865

*Procedure derivatives computed with respect to two input variables: distillate rate and reflux ratio

Column specifications given in Table 12.

Flowsheeting Examples

In this section we utilize the procedure derivatives developed in the previous section for the solution of several steady state flowsheet simulation examples. All flowsheet calculations are performed using the SPEEDUP equation-oriented simulator, using two of the currently available options (Pantelides, 1988).

SPEEDUP solves sets of equations and procedures using variations of Newton's method. Equations present at the flowsheet level are differentiated symbolically and exact analytic derivatives are obtained. As to procedures, with the NEWTON option (here called N) procedure derivatives are used at all iterations. If these are obtained within the procedure, using the method illustrated previously, they can be passed back to the solver (method A). Otherwise they are computed automatically by perturbation of those inputs to the procedure that are unknowns at the flowsheet level (method P). For each base point or perturbation the entire vector of residuals is calculated (a "function evaluation") with the number of evaluations at each iteration minimized using the Curtis-Powell-Reid algorithm (Curtis et al., 1974).

With the HYBRID option (here called H) procedure derivatives are only calculated (by perturbation) at the very first flowsheet iteration and when a restart is required. Quasi-Newton updates of just these derivatives are then used in subsequent flowsheet iterations.

In summary we have the following options:

- Solution method at the flowsheet level—Newton (N) and hybrid (H)
- Calculation of procedure derivatives at the unit operation level—analytic (A) and perturbation (P)
- Calculation of procedure derivatives at the thermodynamic property level—analytic (A) and perturbation (P)

With NAP we indicate a solution using Newton's method at the flowsheet level with analytic unit operation procedure derivatives and thermophysical property derivatives by perturbation, etc. The criteria for assessing the performance of the different strategies are:

1. Simulation time
2. Number of calls to the thermodynamic routines
3. Number of flowsheet iterations and function evaluations

The first example utilizes the standard four-flash Cavett flowsheet shown in Figure 2. Tables 7 and 8 provide a summary of the specifications for five problems. Problems 1, 3, 4, and 5 are simulation calculations whereas problem 2 is a design problem. Equations for the mixer units are given explicitly at the flowsheet level, and the isothermal flash procedure previously described is used for each flash unit, with K values from the SRK equation of state. The four combinations we have tested are NP, HP, NAA, and NAP.

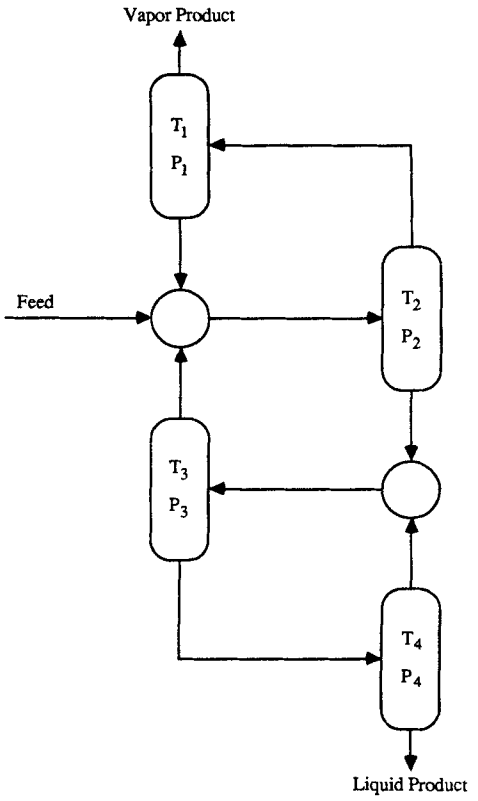

Figure 2. Flowsheet of Cavett process.

Results for problems 1 to 5 are summarized in Tables 9 to 11. When the flash procedure derivatives are computed by perturbation, the hybrid method (HP) is always faster than Newton's method (NP). It also uses fewer function evaluations and significantly fewer accesses to the thermophysical property package,

Table 7. Feed Mixtures Used in Cavett Problems

	Feed Mixture, kmol/h			
Component	L	M	N	O
Nitrogen	—	—	452.0	358.2
Carbon dioxide	512.0	2,257.0	6,698.0	4,965.6
Hydrogen sulfide	—	—	—	339.4
Methane	—	1,362.0	3,782.0	2,995.5
Ethane	363.0	—	3,025.0	2,395.5
Propane	—	1,041.0	2,893.0	2,291.0
Isobutane	—	—	—	604.1
Butane	121.0	700.0	2,707.0	1,539.9
Isopentane	—	—	—	790.4
Pentane	—	—	—	1,129.9
Hexane	93.0	802.0	7,944.0	1,764.7
Heptane	—	—	—	2,606.7
Octane	—	—	—	1,884.5
Nonane	—	759.0	—	1,669.0
Decane	27.0	—	7,020.0	831.7
Undecane	—	—	—	1,214.5

Table 8. Specifications for Cavett Problems

	Problem No. 1	2	3	4	5
	Feed Mixture and No. Components *NC*				
	L 5	L 5	M 6	N 8	O 16
Temp., K					
*T*1	322	342*	311	311	311
*T*2	311	311	322	322	322
*T*3	301	301	309	309	309
*T*4	293	293	303	303	303
Press., bar					
*P*1	19.0	19.0	56.2	56.2	56.2
*P*2	10.0	10.0	19.6	19.6	19.6
*P*3	3.5	3.5	4.39	4.39	4.39
*P*4	1.013	1.013	1.91	1.91	1.91
C10 in vapor product, kmol/h	—	0.070	—	—	—

*Initial guess
Component flow rates in streams 2–11 (Figure 2) were initialized to 1 kmol/h, except stream 5 in problems 1–4 and stream 4 in problem 5, which were initialized to be equal to the feed stream

although it may take a few more iterations to converge, Tables 9–11. When exact procedure derivatives are used at all levels (method NAA), computer times are up to 50% smaller than those obtained with HP and always significantly smaller than those with NP. Even when derivatives of thermodynamic quantities are not available and must be obtained by perturbation (method NAP), times are almost as good as with NAA. For all the Newton method options (NP, NAA, and NAP) solutions are obtained after the same number of iterations, Table 10, which confirms that accurate procedure derivatives are computed by the method proposed previously. That procedures derivatives generated by the hybrid method are only approximations good in the quasi-Newton sense is reflected in the larger number of iterations to converge. Of all the methods tested NAA and NAP converged in significantly fewer function evaluations than the other methods, since perturbation of the flash procedure was not required.

In terms of calls to the thermophysical properties package, Table 11, NAA is by far the best method, followed by NAP, HP, and NP in that order. Factors from Table 2 were used to calculate equivalent calls for property derivatives.

When a standard flash routine from the PPDS package is used simulation times are two to three times larger than those obtained with our flash algorithm implementation. Function

Table 9. Solution Time CPU, CDC CYBER 855

Source of Flash Routine	Solution Algorithm	Problem No. 1	2	3	4	5
This work	NP	0.998	0.973	2.548	3.447	14.958
	HP	0.778	0.818	1.791	2.768	10.896
	NAA	0.502	0.469	1.339	1.490	6.234
	NAP	0.580	0.539	1.934	1.626	6.780
	NP	2.915	2.835	6.578	11.089	*
PPDS	HP	1.906	2.014	3.763	6.345	*

*Not available

Table 10. Number of Iterations/Function Evaluations

Source of Flash Routine	Solution Algorithm	Problem No. 1	2	3	4	5
This work	NP	5/31	4/29	8/57	7/64	7/120
	HP	11/18	11/19	20/28	24/34	27/45
	NAA	5/6	4/5	8/9	7/8	7/8
	NAP	5/6	4/5	8/9	7/8	7/8
PPDS	NP	5/31	4/29	8/57	8/73	*
	HP	11/18	11/19	20/28	26/36	*

*Not available

evaluations and iterations for both routines are the same, indicating that inefficiencies result totally from the flash procedure.

Our second example involves the solution of two design problems (that is with specification of output variables) for a distillation column. The multicomponent distillation procedure of the previous section was used, with the SRK equation of state for fugacities, a polynomial form for pure-component heat capacities, and no excess enthalpy.

In problem 6 the mole fraction of one component in the distillate is specified, and the reflux ratio must be calculated. Since reflux ratio and distillate rate are on input variables to our distillation procedure, this specification is satisfied by an iterative calculation at the flowsheet level. This problem is defined in detail in Table 12, with Table 13 showing initial conditions and solutions. Problem 7 is taken from Holland (1981) and involves the solution of two interconnected columns, Figure 3. The energy recovered from the condenser in column II is used to meet the heat requirements for the reboiler of column I. In addition the bottoms of column II is recycled back to column I. The

Table 11. Equivalent Number of Thermodynamic Calls

Source of Flash Routine	Solution Algorithm	Problem No. 1	2	3	4	5
This work	NP	713	667	1968	1990	3452
	HP	413	430	782	974	1283
	NAA	234	195	310	315	319
	NAP	306	255	561	497	817

Table 12. Definition of Example Problem 6

Specifications	
No. stages	7
Feed stage	4
Top press., bar	5.0
Press. drop/stage, bar	0.0
Feed temp., °C	−0.4
Feed press., bar	5.0
Feed flow rate, (kmol/h)	500.00
C2-Ethane	127.08
C3-Propane	159.37
C4-Butane	96.67
C5-Pentane	67.75
C6-Hexane	49.13
Partial condenser	—
Distillate rate, (kmol/h)	150
Mol frac. C2 in distillate product	0.79

Table 13. Initial Value and Solutions of Problem 6

	Initial Point	Solution
Distillate, kmol/h		
C2	127.08	118.500
C3	159.37	31.384
C4	96.67	0.115
C5	67.75	199.062E-06
C6	49.13	0.0
Bottoms, kmol/h		
C2	5.0	8.580
C3	5.0	127.986
C4	5.0	96.555
C5	5.0	67.750
C6	5.0	49.130
Reflux ratio	0.50	0.726
Distillate	150.0*	150.0*

*Set value

equations stating the equality of the two duties and of the second feed to column I and of the bottoms of column II are satisfied at the flowsheet level by an iterative calculation. A more detailed problem definition is given in Table 15, with Table 16 giving the solutions from this work and from Holland.

Internal column profiles were initialized (for the first call only) according to the standard procedure described by Fredenslund et al. (1977). Specifications were converged to a tolerance of 10^{-5} on the infinity norm of the residuals at the flowsheet level, and the distillation procedure was converged to a tight tolerance of 10^{-8} at every call. (A looser convergence tolerance of 10^{-6} had virtually no effect on the results.) We tested the following combinations of solution methods:

- No derivatives available at any level—NPP and HPP
- Analytic derivatives available only from the thermophysical property level—NPA and HPA
- Analytic derivatives only from the distillation procedure, with perturbation of physical properties—NAP
- Exact analytic derivatives available from all levels—NAA

The statistics for the two problems are summarized in Tables 14 and 17, with factors from Table 2 used to calculate an equivalent number of point value calls to the thermophysical property routines. Great care was taken in performing all calculations with the maximum efficiency. A modification to SPEEDUP was

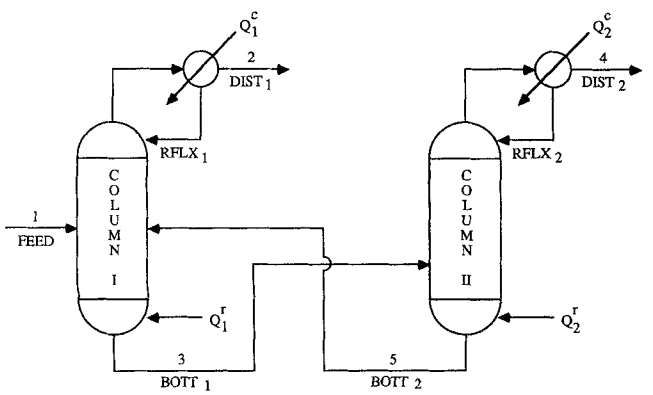

Figure 3. Coupled distillation columns with mass and energy recycles (Problem 7).

Table 14. Solution Statistics, Problem 6

Method	CPU Time s	Flowsheet Iterations	Function Evaluations	Equiv. No. Thermody. Calls
NPP	4.710	4	9	3,705
HPP	4.337	4	7	3,325
NPA	3.466	4	9	1,549
HPA	3.197	4	7	1,323
NAP	3.992	4	5	3,050
NAA	2.899	4	5	1,125

Table 15. Definition of Problem 7 (Interconnected Columns)

Feed		
Ethane	15.0 kmol/h	
Propylene	35.0 kmol/h	
Propane	30.0 kmol/h	
Isobutene	20.0 kmol/h	
Vapor frac.	0 saturated liquid	
	Col. I	Col. II
No. ideal stages	20	12
Efficiency	100%	100%
Condenser press., bar	17.23	24.12
Press. drop/stage, bar	0	0
Condenser type	partial	partial
Feed stage(s)	5, 9	6
Distillate rate, kmol/h	62.23	37.77
Reflux ratio	—	8.0

Table 16. Simulation Results for Problem 7

	Column I		Column II	
	This Work	Holland (1981)	This Work	Holland (1981)
Q^c, GJ/h	5.082	—	6.253	—
Q^r GJ/h	6.253	—	7.134	—
Distillate, kmol/h				
Ethane	14.998	15.0	1.661E-3	6.99E-5
Propylene	29.790	30.695	5.210	4.306
Propane	17.440	16.540	12.560	13.46
Isobutene	1.566E-3	2.067E-4	19.998	20.00
Bottom, kmol/h				
Ethane	1.662E-3	6.99E-5	0.0	1.675E-10
Propylene	5.249	4.307	39.294E-3	1.659E-3
Propane	12.730	13.48	0.170	1.224E-2
Isobutene	35.757	29.97	15.759	9.975
Reflux ratio	4.526	4.0	8.0*	8.0
Distillate rate, kmol/h	62.230	62.23	37.77*	37.77
Feed 1, kmol/h				
Ethane	15.0*	15.0	1.989E-3	6.99E-5
Propylene	35.0*	35.0	5.468	4.307
Propane	30.0*	30.0	12.584	13.48
Isobutene	20.0*	20.0	40.293	29.97
Top temp., °C	35.08	—	93.95	—
Bottom temp., °C	76.33	—	113.26	—
Feed temp. 1, °C	33.09*	—	76.33	—
Feed temp. 2, °C	113.26	—	—	—

*Set values

Table 17. Solution Statistics for Problem 7

Solution Algorithm	Simulation Time CPU s	Flowsheet Iterations/ Function Evaluations	Equiv. No. TD Properties Calls
NPP	46.567	4/29	59,767
HPP	27.137	4/12	33,276
NPA	28.276	4/29	18,199
HPA	14.985	4/12	8,580
NAP	23,960	4/5	23,623
NAA	11.268	4/5	5,087

TD, thermodynamic

introduced to save internal variables in the distillation procedure, making it possible to use converged profiles from one iteration as starting values for the next. Only one internal column iteration turned out to be sufficient in most cases to converge a column for a perturbation.

The results in terms of iterations and procedure evaluations are broadly in line with the previous example. Analytic and perturbation derivatives evaluated at a given point have exactly the same numerical values, so solutions with the same flowsheet level option follow exactly the same convergence path and take the same number of iterations. The only factors affecting the computer times are therefore the number of executions of the column and physical property procedures and the way procedure derivatives are computed. If we compare methods NPP and HPP we observe that the second is faster, because it requires fewer evaluations of the column procedure.

We found this was true in general, even when HPP took a few more flowsheet iterations. Using analytic derivatives of physical properties (K values and enthalpies) rather than perturbation derivatives (methods NPA and HPA) results in significant savings in both problems. Using analytic derivatives of just the column procedure and perturbation derivatives of physical properties (method NAP) also results in time savings, even when perturbation of only one input variable to the column is performed (15% savings in problem 6). When more column perturbations are performed at each flowsheet iteration (problem 7) the time savings are much more significant (50%). When analytic derivatives are used at all levels (method NAA) the two reductions are compounded, producing the best overall result of any combination. This method outperforms even the derivative-free hybrid method. The overall reduction in computer time for NAA in problems 6 and 7 is respectively 39 and 76% with respect to NPP and 33 and 58% with respect to HPP.

Acknowledgment

We thank Costas Pantelides for implementing several modifications to the SPEEDUP simulator required for this study. G. I. Maduabueke was recipient of a scholarship from the Nigerian Government, an Overseas Research Student grant from the British Council, and a bursary from the Physical Property Data Service of the Institution of Chemical Engineers.

Notation

$f, \tilde{f}, g$ = function vectors
$\underline{FX}, \underline{FY}, \underline{FZ}$ = vectors of liquid, vapor, and feed component flow rate
K = vector of equilibrium K values
L, V = total liquid, vapor flow rate
NC = number of components
P = pressure
Q, R, S = matrices of partial derivatives
T = temperature
x, y, u = vectors of input, output, and internal variables

Subscripts

b = base point
i = index

Superscript

T = transpose

Appendix A: Derivation of Matrices Q and S for an Isothermal Flash Procedure (NC Components)

Variables

Input: feed component rates $\underline{FZ}$
flash temperature, pressure T, P

Let $\underline{x}$ be defined as the input vector: $\underline{x}^T = (\underline{FZ}^T, T, P)$

Output: liquid component rates $\underline{FX}$
vapor component rates $\underline{FY}$

Internal: none (hence $R = 0$)

Equations

Component mass balance:

$$FY_i = FZ_i - FX_i, \quad i = 1, 2, \ldots, NC$$
$$\text{i.e., } \underline{FY} = \underline{FY}(\underline{FZ}, \underline{FX}) \quad \text{(A1)}$$

Phase equilibrium:

$$f_i = K_i FX_i \Sigma_j FY_j - FY_i \Sigma_j FX_j = 0 \quad i = 1, \ldots, NC$$
$$\text{i.e., } \underline{f} = \underline{f}(\underline{FX}, \underline{FY}, \underline{x}) \quad \text{(A2)}$$

where $K_i = K_i(T, P, \underline{FX}, \underline{FY})$ is computed from a lower level thermodynamic procedure.

From Eq. A1 (underscores for vectors omitted)

$$dFY = \frac{\partial FY}{\partial FX} dFX + \frac{\partial FY}{\partial x} dx \quad \text{(A3)}$$

From Eq. A2

$$df = \frac{\partial f}{\partial FX} dFX + \frac{\partial f}{\partial FY} dFY + \frac{\partial f}{\partial x} dx \quad \text{(A4)}$$

Substituting Eq. A3 in Eq. A4, rearranging, and using the definition of derivative:

$$\left[\frac{\partial f}{\partial FX} + \frac{\partial f}{\partial FY}\frac{\partial FY}{\partial FX}\right]\frac{\partial FX}{\partial x} = -\left[\frac{\partial f}{\partial x} + \frac{\partial f}{\partial FY}\frac{\partial FY}{\partial x}\right] \quad \text{(A4)}$$

or

$$Q \cdot \frac{\partial FX}{\partial x} = -S \quad \text{(A5)}$$

Solution of Eq. A5, a linear system of NC equations for $NC + 2$ righthand sides gives the sensitivities of liquid component rates $\underline{FX}$ to all input variables. The terms on the lefthand side of Eq.

A5 are as follows:

$$\frac{\partial f_i}{\partial FX_j} = V\left(\frac{\partial K_i}{\partial FX_j} FX_i + \delta_{ij} K_i\right) - FY_i$$

$$\frac{\partial f_i}{\partial FY_j} = FX_i\left(K_i + V\frac{\partial K_i}{\partial FY_j}\right) - \delta_{ij} L$$

$$\frac{\partial FY_i}{\partial FX_j} = -\delta_{ij} \quad \text{where} \quad V = \Sigma_j FX_j$$

$$\delta_{ij} = \begin{cases} 0 & i = j \\ 1 & i = j \end{cases}$$

The terms on the righthand side of Eq. A5 are:

$$\frac{\partial f}{\partial x} = \left[\frac{\partial f}{\partial FZ}\frac{\partial f}{\partial T}\frac{\partial f}{\partial P}\right]; \quad \frac{\partial FY}{\partial x} = \left[\frac{\partial FY}{\partial FZ}\frac{\partial FY}{\partial T}\frac{\partial FY}{\partial P}\right]$$

$$\frac{\partial f}{\partial FZ} = 0; \quad \frac{\partial FY}{\partial T} = 0; \quad \frac{\partial FY_i}{\partial FZ_j} = \delta_{ij}$$

$$\frac{\partial f_i}{\partial T} = FX_i V\frac{\partial K_i}{\partial T}; \quad \frac{f_i}{\partial P} = FX_i V\frac{\partial K_i}{\partial P}$$

Sensitivities of vapor component rates are then obtained from Eq. A1:

$$\frac{\partial FY_i}{\partial FZ_j} = \delta_{ij} - \frac{\partial FX_i}{\partial FZ_j}; \quad \frac{\partial FY_i}{\partial T} = -\frac{\partial FX_i}{\partial T}; \quad \frac{\partial FY_i}{\partial P} = -\frac{\partial FX_i}{\partial P}$$

Literature Cited

Biegler, L. T., I. E. Grossmann, and A. W. Westerberg, "A Note on Approximation Techniques Used for Process Optimization," *Comput. Chem. Eng.*, **9**(2), 201 (1985).

Curtis, A. R., M. J. D. Powell, and J. K. Reid, "On the Estimation of Sparse Jacobian Matrices," *J. Inst. Math. Appl.*, **13**, 117 (1974).

Fredenslund, A., J. Gmehling, and P. Rasmussen, *Vapor-Liquid Equilibria Using UNIFAC, A Group Contribution Method*, Elsevier, New York (1977).

Holland, C. D., "Fundamentals of Multicomponent Distillation," McGraw-Hill, New York (1981).

Macchietto, S., "Solution Techniques for Processes Described by Mixed Sets of Equations and Procedures," *Inst. Chem. Eng. Symp. Ser.* **92**, 377 (1985).

Michelsen, M. L., "The Isothermal Flash Problem. I: Stability Analysis," *Fluid Ph. Equil.*, **9**, 21 (1982).

Ohanomah, M. A., and D. W. Thompson, "Computation of Multicomponent Phase Equilibria. I: Vapor-Liquid Equilibria," *Comput. Chem. Eng.*, **8**(3/4), 147 (1984).

Pantelides, C. C., "Symbolic and Numerical Techniques for the Solution of Large Systems of Nonlinear Algebraic Equations," Ph.D Thesis, Univ. London (1988).

Perkins, J. D., "Equation-Based Flowsheeting," Westerberg, A. W. and H. H. Chien, eds., *Proc. 2nd Int. Conf. Foundations of Computer-Aided Process Design*, A. W. Westerberg, H. H. Chien, eds., CACHE (1984).

Rohl, J. S., and N. Sudall, "Convergence Problems Encountered in Flash Equilibrium Calculations Using a Digital Computer," *Chem. Eng. Symp. Series*, **23**, 71 (1967).

Sargent, R. W. H., "The Decomposition of Systems of Procedures and Algebraic Equations," *Numerical Analysis Proc. of Biennial Conf. Dundee, 1977*, G. A. Watson, ed., Lec. Notes in Math. No. 630, Springer, Berlin, 158 (1978).

Shivaram, S., and L. T. Biegler, "Improved Infeasible Path Methods for Sequential-Modular Optimization," *EFCE Conf. Comput. Appli. in Chem. Eng.*, Paris (1983).

Stadtherr, M. A., and H. Chen, "Strategies for Simultaneous-Modular Flowsheeting and Optimization," *Proc. Int. Conf. Foundations of Computer-Aided Process Design*, A. W. Westerberg, H. H. Chien, eds., CACHE (1984).

Manuscript received Feb. 26, 1986, and revision received Feb. 2, 1988.