DEVELOPMENT OF A NEW EQUATION-BASED PROCESS FLOWSHEETING SYSTEM: NUMERICAL STUDIES

A new equation-based flowsheeting system is described. The system uses a simultaneous linearization approach and employs powerful sparse matrix routines. The equation generation procedure is efficient and makes flowsheet input easy. The flowsheet input format is very much like the format used in conventional sequential modular simulators. The current system is a prototype that is being used to study a number of fundamental computational problems. Results of such numerical studies are reported.

MARK A. STADTHERR

Chemical Engineering Department
University of Illinois
Urbana, IL 61801

and

COURTLAND M. HILTON

Intel Corporation
Aloha, Oregon 97006

Current methods for chemical process simulation and design (flowsheeting) are typically based on the sequential-modular approach, in which the computations for each type of unit operation are organized into modules and solved sequentially. However, this approach has inherent limitations that make it ineffective in dealing with large and complex processes. Since processes of the future are likely to be increasingly complex, as various schemes for conserving material and energy and for controlling effluents are implemented, the development of effective flowsheeting systems for such processes is particularly timely. One strategy for overcoming the current limitations is the equation-based approach. In this case, the computational modules are done away with and all the equations describing a process solved simultaneously. One approach to solving the very large equation system that results is algebraic decomposition (or tearing). In this case, the system may be solved by iterating on relatively few variables. Though this intuitively seems desirable, serious computational problems can arise, as noted by Lin and Mah (1978), who advocate a simultaneous linearization approach as potentially more powerful. In this case, all the equations are linearized and all variables iterated on simultaneously using a Newton-Raphson or quasi-Newton approach. Although in the past, equation-based flowsheeting calculations were based primarily on algebraic decomposition, there has been considerable recent interest (Gorczynski and Hutchison, 1978; Gorczynski et al., 1979; Westerberg and Berna, 1978; Lin and Mah, 1978; Benjamin et al., 1981) in the simultaneous linearization approach. This is the approach used here. The new system is a prototype that is being used in studies of a number of fundamental computational problems. In this paper, we describe the prototype system and present results for some of the studies made using the system.

One study reported here involves the reliability with which convergence is obtained from the "worst-case" initial guess situation. This is the case in which the user supplies no initial guess values and all must be generated internally by the program. We compare the convergence behavior for three schemes for internally generating an initial guess, and for several strategies for linearizing the nonlinear equations. Also presented is a study involving the reliability of convergence from a "best-case" initial guess situation using several different linearization strategies. Different strategies for computing thermophysical properties are also considered.

BACKGROUND

Extensive reviews of past work in the area of process flowsheeting have been provided by Motard et al. (1975) and Hlavacek

This work was performed at the University of Illinois. Correspondence should be addressed to Prof. Stadtherr.

0065-8812-82-5927-0124-$2.00

(1977), and more recent work has been reviewed by Rosen (1980), and Evans (1981). These reviews discuss many of the advantages and disadvantages of the modular and equation-based approaches to process flowsheeting. Also, the recent monograph of Westerberg et al. (1979), is particularly useful as an introduction to the field of process flowsheeting.

One can distinguish between two general approaches to equation-based flowsheeting. The first of these, typified by the SPEEDUP system (Leigh et al., 1974), involves tearing. One guesses, or tears, values for a number of variables sufficient to permit values for the remaining variables to be found by solving a sequence of small, usually one-variable problems. The remaining equations (tear equations) may then be solved for new values of the tear variables, and some sort of successive substitution procedure used, providing of course that the tear equations contain the tear variables explicitly. If this is not the case, the residuals in the tear equations can be used in connection with other standard root-finding procedures. Thus in effect one is able to solve a large system of nonlinear equations by iterating on only a few tear variables, thereby drastically reducing the dimensionality of the problem.

The key step in the tearing approach is the development of an appropriate solution strategy (information flow pattern) for the particular problem at hand. That is, one must decide which variables to tear, which equations to solve for which variables (output set), and in which sequence to solve them (precedence order). Furthermore, since most equation systems describing chemical processes will be underconstrained, it is also often necessary to designate certain variables as design variables. What is needed then is a systematic and efficient procedure for finding a solution strategy that will converge reliably and rapidly to the solution. Over the past several years, a variety of techniques have been devised with this as a goal. Many are designed to produce a solution strategy involving a minimum number of tear variables. Others also try to account for the relative difficulty of the single-variable problems that must be solved. Still others involve sensitivity considerations, for reasons discussed in more detail shortly. Many such techniques are prone to combinatorial problems and thus are not reliably efficient. Moreover, there is little degree of certainty that the solution strategies obtained will reliably converge. Thus, a basic premise of the tearing approach, namely that a small-dimensioned problem is easier to solve than a large-dimensioned one, is not necessarily correct. This has been emphasized by Lin and Mah (1978), who point out that because a very long chain of computations typically exists between the guessed tear values and the residuals in the tear equations, sensitivity problems can arise that may cause divergence even for initial guesses very near the solution. For this reason, methods for employing sensitivity considerations in choosing a solution strategy have been developed, as mentioned above. Compared to the sequential-modular approach, the tearing approach is typically faster and capable of solving more complex flowsheeting problems. Nevertheless, because the problem of efficiently choosing a reliable solution strategy has not been completely solved, there has been reluctance to adopt this approach.

A second and more promising approach to equation-based flowsheeting is the quasilinear approach. This involves the simultaneous linearization of all the equations and iteration on all the variables, using the Newton-Raphson method, a quasi-Newton method, or some hybrid thereof. Thus in each iteration we are faced with solving a huge set of sparse linear equations, involving perhaps several thousand variables. At this point the quasilinear approach has been proven very successful in dealing with some specialized problems, such as flows in pipe networks (Mah, 1974; Bending and Hutchison, 1973) and simulation of distillation columns (Hutchison and Shewchuk, 1974; Kubicek et al., 1976). These applications have been summarized by Westerberg et al. (1979), who also emphasize the promising aspects of this approach. Application of the quasilinear approach to flowsheeting problems in general is a very recent development. The work of Mah and Lin (1978), who apply this approach to a flowsheeting problem involving the simulation of a natural gas liquefaction process, is indicative of the potential of this approach, but they do not provide a generalized flowsheeting system. Hutchison and coworkers at the University of Cambridge in Great Britain have recently described a flowsheeting package called QUASILIN (Gorczynski et al., 1979) that may be regarded as a prototype of such a generalized quasilinear flowsheeting system. The ASCEND II package (Benjamin et al., 1981) developed by Westerberg and co-workers at Carnegie-Mellon is equally noteworthy in this regard.

The most fundamental computational problems with the quasilinear approach involve the strategy to be used to converge the nonlinear equations, and the strategy to be used to solve the huge sparse linear systems that arise after linearization. The sparse matrix strategy used determines in effect the limit on the size of problems that can be solved. For instance, consider the problem of solving the linear system $Ax = b$. The usual solution procedure can be represented by the factorization $A = LU$ where L is lower triangular and U is upper triangular. The elements of L and U are usually found using Gaussian elimination or some variation thereof. If A is large and sparse, the number of nontrivial nonzeros in L and U may greatly exceed the number of nonzeros in A. This loss of sparsity, or "fill-in", may lead to excessive storage requirements and unacceptably long computational times. Without the use of sparse matrix strategies to reduce this fill-in, only very small flowsheeting problems could be handled by the quasilinear approach. Since one of the main reasons for adopting an equation-based approach is its ability to handle large and complex problems, the need for effective sparse matrix strategies is particularly important. Westerberg et al. (1979) emphasize that while current sparse matrix software is capable of handling problems involving one or two thousand equations and sometimes more, realistic chemical plant flowsheeting problems will require sparse matrix strategies capable of handling equation systems larger by an order of magnitude or more. An important feature of the new flowsheeting system we are developing is that it is interfaced to a set of powerful new sparse matrix routines that can handle problems involving several thousand equations without resort to decomposition. This represents significant progress toward the goal cited above.

The strategy used to converge the nonlinear equations determines, to a great extent, the reliability and speed with which a given problem can be solved. Straightforward Newton-Raphson is not particularly attractive because, as discussed below, it may be difficult to supply a good enough initial guess. There are a variety of approaches to improving convergence from a poor initial guess. One promising approach is that described by Gorczynski and Hutchison (1978). They note that since a second-order linearization method, such as Newton-Raphson, provides speed at the expense of reliability, while first-order methods provide reliability at the expense of speed, a blend of the two linearization strategies would seem to be appropriate. Indeed they seem to have had some success with such blended linearizations. Unfortunately, there is little published regarding such important details as the actual values of the blending parameters used. Another approach to the convergence problem is to use "hybrid" or "dogleg" methods (Powell, 1970; Westerberg and Director, 1978) that consider the steepest-descent direction in addition to the Newton-Raphson direction. A hybrid method recently described by Chen and Stadtherr (1981) appears to be very reliable. Though the current code based on this method is written for full matrix problems, it can also be extended to sparse systems of nonlinear equations. This work is currently in progress. The need for nonlinear equation solvers capable of converging from poor initial guesses could of course be ameliorated considerably given an efficient procedure for generating a good initial guess. The difficulty here is that since all variables are iterated on, they all require initial values. For relatively small problems, the user may be quite capable of supplying a good initial guess; however, on larger problems, the user may not be able to provide initial guesses for all of the several thousand variables that may be involved. Thus one needs algorithms for systematically generating "good" initial guesses given little or no user input. Gorczynski and Hutchison (1978) outline a simple initialization scheme, though again little numerical detail is provided, and it is unclear how effective it is.

DESCRIPTION OF FLOWSHEETING SYSTEM

The system described here, which we have dubbed SEQUEL, is a prototype of a new equation-based process flowsheeting system. As the name implies, SEQUEL is an outgrowth of previous work in process flowsheeting and represents a new chapter in the continuing effort to design and simulate chemical processes by equation-based methods. It must be emphasized that SEQUEL has been designed primarily as a tool for use in developing and evaluating computational strategies for equation-based flowsheeting. Thus, SEQUEL is lacking several features that would be important in a production code. For instance, although flowsheet input to SEQUEL is very easy, friendliness to the user has in general been a secondary concern. Also, the number of components in the physical property data base is quite small and the thermodynamic models used are very simple and are inadequate for

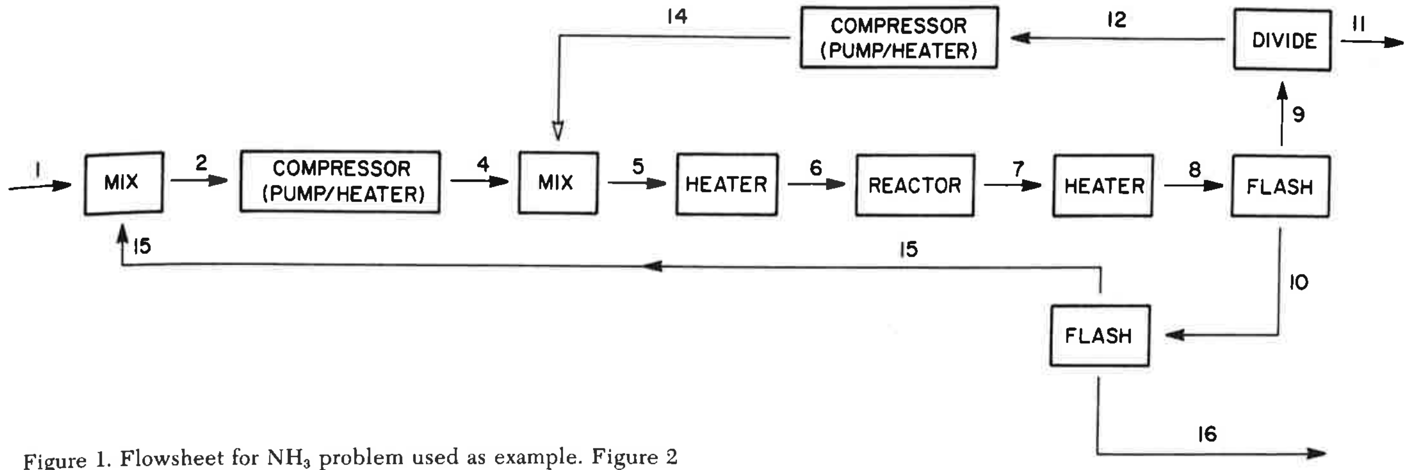


Figure 1. Flowsheet for NH_3 problem used as example. Figure 2 shows the flowsheet input file for this flowsheet.

some problems.

Equation Generation

One problem commonly cited (e.g., Evans, 1981) in connection with the equation-based approach is that a complex executive routine is needed to generate the equations describing a particular flowsheet. Two ideas have appeared in recent years that aid in overcoming this difficulty. The first such idea is modularity (Gorczynski _et al_., 1979; Benjamin _et al_., 1981). SEQUEL is organized in a modular fashion, with the modules corresponding to the various standard process units. Of course in this context, the modules are not procedures for equation solving, as in the sequential-modular case, but instead are used for generating the equations for a particular process unit. The modular architecture is a natural response to the need for flexibility and ease of user input, as well as the desire for an efficient and easily understood code. The second idea (Gorczynski _et al_., 1979; Mah and Lin, 1978; Book and Ramirez, 1978) for simplifying the equation generation code involves the observation that the equations representing a process, typically involve a rather limited number of equation types. In SEQUEL, this is exploited by using a "library" of standard equation types. It is assumed that the flowsheet can always be represented by a set of equations of these standard types. When necessary, the library of standard types can be augmented by the user. The use of standard equation types is a very efficient way to store the information required for Jacobian and function evaluation. Rather than maintain a copy of this information for each occurrence of a particular equation type, only one copy is required. It is also worth noting that by adopting this approach, we avoid the use of algebraic equation-manipulation programs, whose use in this context has sometimes been proposed (Hanyak, 1980; Coup _et al_., 1981; Kubicek _et al_., 1976).

```
"M+E"

16  1 1 1 1 1 1 1 2 1 0 1 1 1 1 1 0

 5  7 8 9 1 10

"THERMO"  2  "SUB"

"MIX" 1    3  1 15  2
"PMPVAL"   2  2  2  3
"HTR" 3    2  3  4
"MIX" 4    3  4 14  5
"HTR" 5    2  5  6
"RXTR" 6   2  6  7
"NH3"     -3 -1  0  0  2 .25 100.0
"HTR" 7    2  7  8
"FLASH" 8  3  8  9 10
"SPLT" 9   3  9 11 12
"PMPVAL" 10  2 12 13
"HTR" 11   2 13 14
"FLASH"  12  3 10 15 16

"END"
```

Figure 2. Flowsheet input file for NH_3 problem shown in Figure 1. See text for additional detail.

Flowsheet input to the equation generator uses standard FORTRAN free-format conventions. Figure 1 is an example of a chemical process flowsheet; the input file describing this flowsheet is shown in Figure 2. There are six types of statements in the input file:

1. The first statement specifies whether the equation set will describe only a material balance of the process or a complete material and energy balance.

2. The second input is the total number

of streams in the flowsheet together with a specification of the phase of each stream. The user may specify the stream as all vapor, all liquid, or of unknown state.

3. The third input is the number of chemical species in the process together with their property data bank identification numbers.

4. The fourth gives the type of thermodynamic model to be used for predicting vapor-liquid equilibria, and whether the thermophysical properties are to be evaluated in external subroutines or as part of the overall matrix.

5. Fifth, the unit operations involved are listed followed by their user identification number, the number of streams connected to the unit operation, and the numeric designator of each such stream. The unit operations may be listed in any order.

6. Sixth, the input list is terminated with an END statement.

Like in many other flowsheeting programs, each connecting stream i is described by a vector $\underline{S}_i$ comprising molar component flowrates, total stream flowrate, temperature, pressure, and total stream enthalpy. That is, $\underline{S}_i = (n_i, F_i, T_i, P_i, H_i)$. Component flowrates were chosen as primitive variables rather than mole fractions because material balances around a unit could then be represented by linear equations. The total stream flow was included for clarity and to simplify the structure of some equations. Of course, the inclusion of both component flows and total stream flow as variables requires an additional equation for each stream, namely the stream balance that equates total flow to the sum of the component flows. After reading the number of streams and number of components, SEQUEL creates a variable vector $\underline{X}$ that comprises all of the stream vectors. That is $\underline{X} = (\underline{S}_1, \underline{S}_2, \ldots, \underline{S}_M)$ where M is the number of streams. Since the order of the variables within each stream vector is the same, it is easy to locate any individual variable within the X-vector. After initialization, the X-vector contains only the external variables describing flows between units. As discussed below, internal variables are appended to the X-vector as they arise in generating the equations.

As each unit operation in the input file is read, control within the equation generator branches to the module for that type of unit operation. This module subsequently issues calls to a subroutine library that creates general balance equations as well as an equation set unique to the unit operation and stream configuration specified. Figure 3 illustrates the general program flow during equation generation.

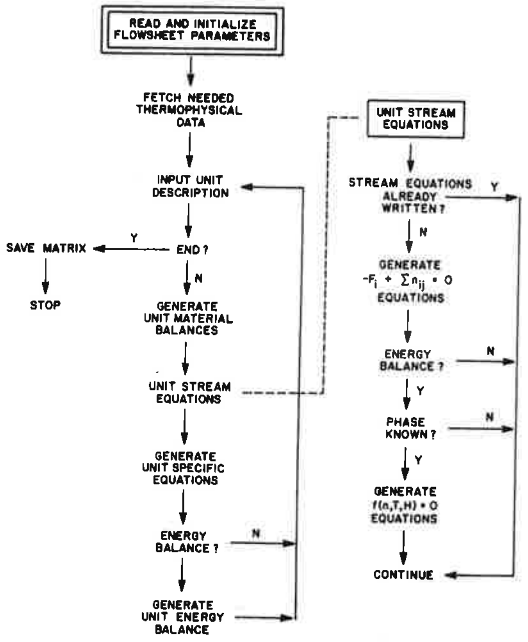

READ AND INITIALIZE FLOWSHEET PARAMETERS
FETCH NEEDED THERMOPHYSICAL DATA
INPUT UNIT DESCRIPTION
END ?
Y
SAVE MATRIX
STOP
N
GENERATE UNIT MATERIAL BALANCES
UNIT STREAM EQUATIONS
GENERATE UNIT SPECIFIC EQUATIONS
ENERGY BALANCE ?
N
GENERATE UNIT ENERGY BALANCE
UNIT STREAM EQUATIONS
STREAM EQUATIONS ALREADY WRITTEN ?
Y
N
GENERATE $-F_i + \sum n_{ij} = 0$ EQUATIONS
ENERGY BALANCE ?
N
Y
PHASE KNOWN ?
N
Y
GENERATE $f(n,T,H) = 0$ EQUATIONS
CONTINUE

Figure 3. Information flow in equation generation routine.

The equations representing any given unit operation are a subset of a "library" of standard equation types, as shown in Figure 4. In defining standard equation types, care must be taken to structure the equations so that numerical difficulties such as division by zero are avoided. Additional equation types can be added as necessary.

As the equations are generated, occurrence matrix information is stored by sequentially filling an array CI with the position in the X-vector of each variable contained in an equation. A pointer array RLI is maintained that indicates where in CI each row (equation) begins. An additional vector EQN is prepared that contains numerical values specifying the equation type corresponding to each row of the occurrence matrix. If an equation type involves coefficients that vary from row to row, these constants are stored in another vector. As an example of the equation generation process, we show in Figure 5 the arrays produced when the material balances around a heater are generated.

Some equations involve the use of internal variables such as K-values, split fractions, and heat loads. As these variables

Equation Type	
1	$-X_0 + \sum_i X_i = 0$
2	$-X_0 + c = 0$
3	$-cX_0 + X_1 = 0$
4	$-X_0 + \sum_i X_i + c = 0$
5	$\sum_i X_i + c = 0$
6	$-X_0 + \min[X_1, X_2, \ldots, X_i] = 0$
7	$-X_0 X_1 + X_2 = 0$
8	$-X_0 X_1 X_2 + X_3 X_4 = 0$
9	$-X_0 X_1 + \exp\left[c_1 - \frac{c_2}{c_3 + X_2}\right] = 0$

Equation Type	
10	$-X_0 + \sum_{i=1}^{N} b_i X_i + \sum_{i=1}^{N} X_i c_{pLi}(c_i - c_0) + \sum_{i=1}^{N} X_i \int_{c_i}^{N_{X+1}} C_{pVi} dX_{N+1} = 0$ (vapor)
	$-X_0 + \sum_{i=1}^{N} X_i C_{pLi}(X_{N+1} - c_0) = 0$ (liquid)
	where:
	$C_{pLi} = a_i \left[d_1 + d_2 (1 - \frac{X_{N+1}}{e_i})^4 + d_3 (1 - \frac{X_{N+1}}{e_i})^{-1} \right]$
11	$-X_0 X_1 + \exp\left[c_1 + \frac{c_2}{X_2} + c_3 \ln X_2 + c_4 \frac{X_0 X_1}{X_2^2}\right] = 0$
12	$-X_0 + X_1 + c_0 \min\left[\frac{X_1}{c_1}, \frac{X_2}{c_2}, \ldots\right] = 0$
13	$-X_0 + X_1 + X_2 + c_0 \min\left[\frac{X_3}{c_3}, \frac{X_4}{c_4}, \ldots\right] = 0$

Figure 4. Partial list of standard equation types used in SEQUEL. There are currently 22 standard equation types. The X_i's are the variables; the c's are constants.

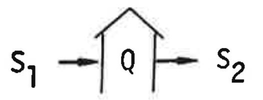
arise in generating the equations they are sequentially assigned a position in the X-vector. An identification vector is also maintained containing the type of variable at each position for use when subjecting the X-vector to global constraints, as discussed below.

The arrays produced by the equation generator are used directly when performing function evaluations and when creating and updating the linearized model at each iteration. When a call is issued by the solution portion of SEQUEL to evaluate functions or derivatives, EQN is sequentially scanned. For each equation I, the program branches to the procedure specified by EQN (I), and using the occurrence matrix information in CI in conjunction with the values in the X-vector, the desired expressions are evaluated. As previously stated, this type of program, in which each procedure (equation type) is coded only once, results in a tremendous reduction in core requirements, and makes equation generation efficient even for very large problems.

Thermophysical Property Models

As noted above, the models available to date in SEQUEL are very simple and will not always be adequate. In part, this is because our interest here is less in the accuracy of the models than in the computational strategy for using them. For determining such properties as K-values, dew and bubble points, enthalpies, vapor fractions, etc., one can in general identify three types of computational strategies:

Case 1. In the simplest case all thermophysical properties are evaluated in external subroutines. Therefore, the thermophysical models are not treated as part of the overall process matrix and are not subject to linearization. That is, no linearized K-value, dew and bubble point, or phase determination equations appear in the process matrix. A linearized enthalpy equation for each stream in the flowsheet is included in the matrix to allow the update of each stream's temperature. After a process matrix iteration, current values of temperature, pressure, and composition are provided to the subroutines, which in turn update, perhaps iteratively, the thermophysical property values required for function and Jacobian evaluation. The omission of the thermophysical property expressions from the matrix keeps the process matrix size smaller than would otherwise be possible. However, except for stream enthalpies, no information on the variation of the property values with temperature, pressure or composition is included. A linearized process model based on such a strategy is inaccurate and its convergence behavior may be poor for some problems. For example, in modeling a process



		Variable Slate	Position in X-vector S_1	S_2
n_{11}	n_{21}	n_1	1	7
n_{12}	n_{22}	n_2	2	8
F_1	F_2	F	3	9
T_1	T_2	T	4	10
P_1	P_2	P	5	11
H_1	H_2	H	6	12

Material Balances:

$-F_1 + F_2 = 0$

$-n_{11} + n_{21} = 0$

Stream Balances:

$-F_1 + n_{11} + n_{12} = 0$

$-F_2 + n_{21} + n_{22} = 0$

in terms of X-vector components:	Equation type
$-X_3 + X_9 = 0$	1
$-X_1 + X_7 = 0$	1
$-X_3 + X_1 + X_2 = 0$	1
$-X_9 + X_7 + X_8 = 0$	1

Vector	Position 1	2	3	4	5	6	7	8	9	10
EQN	1	1	1	1						
RLI	1	3	5	8	11					
CI	3	9	1	7	3	1	2	9	7	8

Figure 5. Arrays created when generating material and stream balance equations around a heater unit.

including an adiabatic flash, it is important to include the K-value dependence on temperature in the linearized process model. A model without this information, in general, will exhibit poor convergence behavior. On the other hand, if one were modeling a process involving only isothermal flashes, good convergence could still be expected.

Case 2. This case is like the first, with the distinction that linearized K-value equations are now included in the process matrix. This contributes to a possibly drastic increase in the size of the occurrence matrix, since multiple copies of the linearized K-value model must be included. However, the model is more accurate and may be expected to perform well in both the adiabatic and isothermal cases. The subroutines are used to update the property values, and may also be used to generate the coefficients in the linearized models. This means that within each Newton-Raphson iteration, there may be nested iteration loops in the subroutines that need to be converged.

Case 3. A third type of strategy avoids this nesting of iteration loops by eliminating the subroutines and treating the physical property models simply as equations within the overall set whose residuals are to be driven to zero by the Newton-Raphson process. In this case, the thermophysical property models are not converged until the end, and the properties are updated at each iteration by the Newton-Raphson process and not by external subroutines. In comparison to Case 2 this may mean another drastic increase in the size of the occurrence matrix. We are currently using SEQUEL to perform numerical studies comparing these three strategies.

Though the models used to date are simple, the flexible nature of SEQUEL allows the inclusion of more complex models with little difficulty. Thermodynamic models are represented as standard equation types within the library set. It should be noted that a standard equation type may in fact be a procedure involving several equations. Thus, complex models can be handled in this context. In the current version of SEQUEL, all streams, both liquid and vapor, are modeled as ideal solutions. Raoult's law is used for K-value prediction, along with vapor pressures predicted using either the Antoine equation or an equation of standard type 11 as given by

Reid, Prausnitz, and Sherwood (1977). The principal differences between the vapor pressure equations is that they are, respectively, explicit and implicit in vapor pressure, the latter thus requiring iterative solution. Heat capacities of vapor components are evaluated using third order polynomials in temperature. Liquid component heat capacities are evaluated using an equation of standard type 10. The thermophysical property data base is a subset of the data base included in Reid, Prausnitz, and Sherwood (1977) and thus is very similar to the CACHE data base. The data base is stored in a separate file and is accessed only once.

Design Specifications

After the equation generation routine has processed the flowsheet input file, the equation set generated will in general contain more variables than equations. Thus, design variables must be chosen and assigned values, or other specifications added to the equation set. Design variables are handled by adding to the equation set equations that set the design variables equal to their specified values. In this version of SEQUEL the user is required to choose the design variables and supply a consistent set of specifications. Though the user entry of this information is straightforward, the program could be made friendlier to the user by incorporating one of the procedures available for automatically selecting some or all of the design variables.

Initialization

Since all variables are iterated on, they all require initialization. Ideally these initial guesses would be provided by a user whose insight into the process at hand permits him to provide good guesses. For relatively small problems, this is not an unreasonable expectation. For larger problems, however, this may be impractical. Nevertheless, even on large problems a relatively good initialization may be available. For instance, the result of a similar problem solved earlier may be used. Also, by starting with a small part of the overall process for which a good guess can be made, one may exploit the modular nature of SEQUEL to "build" a process a unit or two at a time. This is essentially a "continuation" approach in which a sequence of problems is solved, the solution of one problem being used to generate initial guesses for the next.

The worst case with respect to initialization is that the user will be unable to supply an initial guess, requiring the program to assign initial values internally. In our numerical studies using SEQUEL one interest is in studying the performance of the quasilinear approach in this situation. Though any number of internal initialization schemes might be devised, for the purpose of the numerical studies presented here we have used three such "worst-case" schemes. In all three schemes, temperature, pressure, and enthalpy are set to preselected reference values, heat loads are set to zero, and split fractions are set equal. These values are overridden if other values are specified as design variables. The guesses for the K-values are calculated using these preset or specified temperatures and pressure values. The three schemes differ in how component flowrates are guessed:

Type 1. In this most primitive scheme all unspecified component flows are set to some arbitrarily small number (we used 0.001). Somewhat surprisingly this works remarkably well for some kinds of problems.

Type 2. In this case all component flows are set to some arbitrary "average" values. As in the previous scheme each stream will have the same component flowrate values, except where overridden by a design specification. There are any number of ways for choosing the "average values". For instance, one could simply use specified or estimated feed stream values.

Type 3. In this case a set of heuristics is used, one for each type of unit, and a guess is generated by beginning with the feed stream and moving sequentially through the process until guessed flows for all streams have been generated. Only one pass is made through the process, so there is no iteration on recycle streams. The heuristics are as follows:

1. FEED STREAMS. Feed component flows not specified are estimated using specifications for those components elsewhere in the flowsheet if possible. For a stream component for which this is possible, the feed flow is set to the average of flows specified elsewhere for that component times the number of output streams. For components that do not have a flow specified anywhere in the flowsheet, the feed stream flows are set to the average of all specified component flows.

2. STREAM MIXER. Guess the output stream by assuming that any unknown inlet flows contribute one-half the flow in the largest known (or already guessed) inlet.

3. STREAM DIVIDER. Guess outlet streams using specified outlet/inlet ratios if available. Otherwise assume inlet flow is divided equally among outlets.

4. EQUILIBRIUM FLASH. Guess outlet streams by assuming total liquid flow out equals total vapor flow out. Use the K-values guessed as described above.

5. REACTOR. If conversion is not specified, guess outlet by assuming 100% conversion of limiting reactant in inlet.

In using these heuristics, values given by design specifications always override the guessed flows and assumed parameters. Many other heuristics such as these could be used, and may be better than those given here.

Equation Linearization and Solution

Three linearizations are currently available in SEQUEL. These are: standard second-order Newton-Raphson linearization, Newton-Raphson with step size relaxation, and a hybrid linearization that blends first and second order linearizations according to user specification. The hybrid linearization, an approach suggested by Gorczinski and Hutchison (1978) is based on *a priori* analysis of the nonlinear equation types. Each nonlinear term is linearized by selecting all but one of the variables in the term to be constant. The selection was made when SEQUEL was programmed. The resulting linearization is first order. It is then straightforward to gradually shift from the first order representation to the full Newton-Raphson linearization. The user specifies the value of the blending parameter (zero to one, where one is a full Newton-Raphson linearization) and the rate at which the blending parameter is to be increased. The idea, of course, is that far from the solution the stability of the first-order linearization is desirable, while nearer the solution the speed of the second-order linearization is desirable. While requiring the user to select the blending parameter and rate may not be desirable in a production code, the requirement is consistent with our use of SEQUEL to study different computational strategies.

X-vector values may vary considerably from iteration to iteration, and values may be outside the range of physical feasibility, e.g., negative flowrates. In some cases, this may cause no difficulty, but in others it may create numerical difficulties in performing function and derivative evaluations. The values for the X-vector are therefore constrained to be within user-defined bounds of feasibility. SEQUEL utilizes global constraints as opposed to constraints on individual variables. For example, all flowrates are subject to the same constraints, rather than varying flowrate constraints from component to component and stream to stream. SEQUEL constrains pressures and flowrates to be positive, and, to prevent excursions too far from the applicable range of thermophysical property models, constrains temperatures within maximum and minimum values and enthalpies below some maximum value. When a constraint is violated, the X-vector values are reset heuristically. Convergence is monitored using the norm of the correction vector as well as the norm of the residual vector.

Sparse Matrix Processing

For solving the linearized equations, we use a two-pass approach. In the first pass, the equations and variables are reordered into a matrix form that reduces fill-in during the solution. In the second pass, the linear system is solved using Gaussian elimination or some other technique. In this case, since all the linearized problems have the same structure, the matrix reordering is done only once and is saved for use in later iterations.

Various reordering and solution methods are available to SEQUEL. For instance, reordering methods available include P^3 (Hellerman and Rarick, 1971), hierarchical partitioning (Lin and Mah, 1977), and our modifications of these methods. Solution methods available include Gaussian elimination and the methods described by Stadtherr (1979) and Stadtherr and Wood (1980). Work comparing various combinations of reordering and solution methods is in progress. Using a code based on the solution method given by Stadtherr and Wood (1980), we are able to solve systems of several thousand equations without problem decomposition. For larger problems decomposition is required. Although the current version of SEQUEL does not have a decomposition capability, methods have been described by Westerberg and Berna (1978) and Stadtherr and Hilton (1982).

NUMERICAL STUDIES

Problem Set 1

The studies reported here involve the reliability with which convergence is obtained from the so-called "worst-case" initial guess situation described above. We compare the performance in this regard of the three initial guess types given above and of several different linearization strategies. What if any effect the method of handling thermophysical properties has in this regard is also considered in preliminary fashion.

For the studies reported here, we used ten problems involving the three flowsheets shown in Figures 1, 6, 7, all of which are familiar in the flowsheeting literature. As shown in Table 1, there are two flash with recycle problems, four ammonia manufacture problems and four Cavett problems. The problems differ with respect to which variables are specified by the user. Note that problems 5, 6, 8, 9, and 10 have the feed stream and equipment parameters specified and thus can be categorized as simulation problems. The remaining problems are design problems with one or more output flows specified. The problems may also be classified based on the type of flash specification used. Problems 1, 3, 5, 7, 8, and 9 have flash temperatures specified; we refer to this as the isothermal flash case. In other problems, the flash heat loads are specified; we refer to this as the adiabatic flash case even though the heat load may be specified to be nonzero.

For the flash with recycle problem a 50% recycle rate is used. For the ammonia problem the precent conversion of the limiting component is set at 25%. Also, since this version of SEQUEL has no compressor module, pump and heater modules are specified to provide the

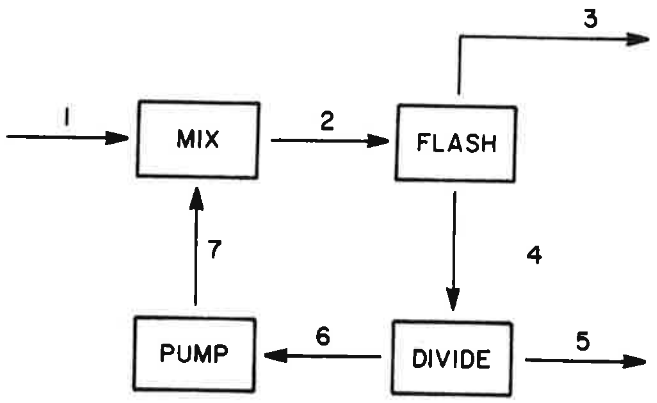


Figure 6. Flowsheet for flash with recycle problem used as example.

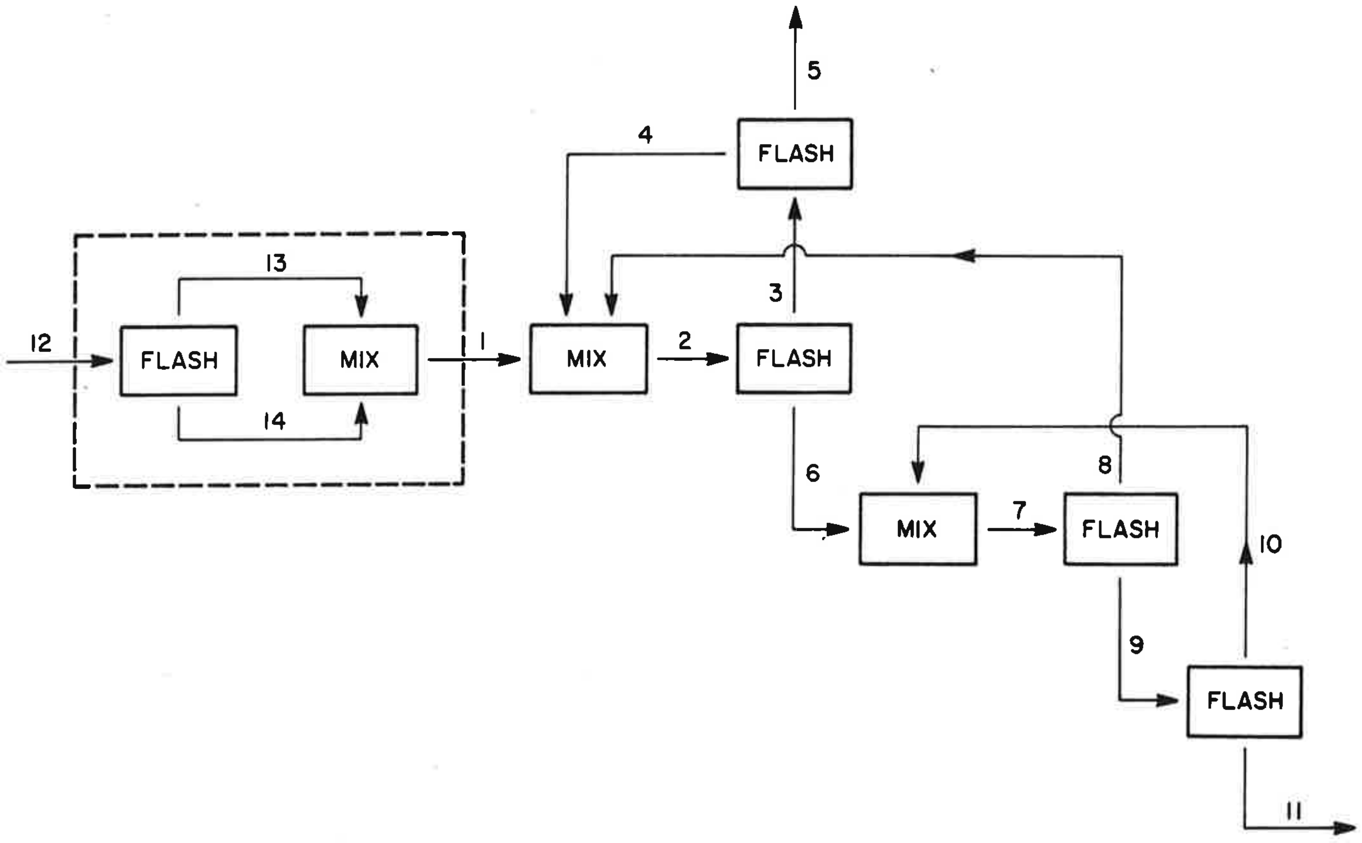


Figure 7. Flowsheet for Cavett problem used as example. Dashed area indicates flash/mix inserted to test condition of feed.

Table 1. Summary of ten test problems used in Problem Set 1.

Flowsheet	Problem Number	Specifications
Flash with Recycle (6 components)	1	n_{11}, T_1, P_1, n_{33}, n_{54}, $n_{31} = 0.994\, n_{11}$, $n_{56} = .69\, F_5$; all unit parameters (T and P for flash)
	2	same as #1, but Q specified for flash, not T
NH_3 Synthesis (5 components)	3	T_1, P_1, $n_{11,1}$, $n_{11,2}$, $n_{16,3}$, $n_{16,4}$, $n_{16,5}$; all unit parameters (T and P for flashes)
	4	Same as #3, but Q specified for flashes, not T
	5	T_1, P_1, $\underline{n}_1$; all unit parameters (T and P for flashes)
	6	same as #5, but Q specified for flashes, not T
Cavett Problem (16 components)	7	T_1, P_1, F_5, F_{11}, $n_{5,1}$; $n_{11,4}$, $n_{11,5}$, ..., $n_{11,16}$; T and P specified for flashes; material balance only
	8	T_1, P_1, $\underline{n}_1$; T and P specified for flashes; material balance only
	9	same as #8, but energy balance also performed
	10	same as #9, but Q specified for flashes, not T

desired temperature and pressure changes. For the Cavett problem a sixteen component system was used. A flash-mixer combination is inserted into the feed stream as a means of testing the feed condition. It should be noted that the largest problem considered here has only 365 equations and thus comes nowhere close to taxing the sparse matrix handling capabilities of SEQUEL. Comparisons of the various sparse matrix routines available in SEQUEL will be presented elsewhere.

The linearization strategies compared are summarized in Table 2. The first strategy is pure Newton-Raphson. The next four are Newton-Raphson with step size relaxation; these differ in how the relaxation parameter r is determined. In strategy #2, r is selected based on a standard norm reduction scheme. Strategies #3-5 employ a scheme that is more primitive, but which performs better on some problems. In these strategies the user supplies an initial value of r and a percentage increment, which is implemented after each iteration in which improvement is observed. The remaining strategies involve a blending of first and second order linearizations, as discussed above. The user supplies an initial blending parameter that indicates the fraction

Table 2. Summary of linearization strategies used in Problem Set 1.

Method	Strategy Number	Parameters	
Newton-Raphson	1		
Newton-Raphson with step size relaxation	2	r set by norm reduction:	
	3	initial r = 0.8,	increment = 10%
	4	= 0.5	= 10%
	5	= 0.2	= 10%
Blended first-order and second-order linearizations	6	Initial = 0,	$\Delta\beta$ = 0.1*
	7	= 0	= 0.1
	8	= 0.5	= 0.5*
	9	= 0.5	= 0.1
	10	= 0.8	= 0.05
	11	= 0.9	= 0.1
	12	= 0	= 1.

*indicates $\Delta\beta$ applied only after iteration during which improvement occurred; otherwise $\Delta\beta$ applied after every iteration.

of a second-order linearization used. Thus $\beta = 1$ indicates a Newton-Raphson linearization and $\beta = 0$ a pure first-order linearization. The user also specifies an absolute increment $\Delta\beta$ that is applied either after every iteration or only after iterations in which improvement occurs. In our experience, it is not desirable to remain in the blended ($\beta < 1$) mode for too many iterations, thus in general the values of $\Delta\beta$ used are relatively large.

Since the comparison of the strategies

for handling thermophysical property calculation is not the principal concern of this problem set, somewhat simplified versions of the three strategies discussed above were used. In particular, no dew or bubble point calculations were performed. It is for this reason a flash/mix is inserted in the feed stream to the Cavett problem to check the phase condition. However, the fundamental differences between the three cases are still maintained, and each case presents a distinct problem to the various equation solving routines.

Each of the ten problems was solved using the twelve linearization strategies on the nine possible combinations of initial guess types and thermodynamics cases, the only exception being that the ammonia problems were not solved for thermodynamics Case 3. Thus a total of 936 solutions were attempted. Table 3 summarizes the success rate for the various guess types and thermodynamics cases. The entries in this table indicate what percentage of the twelve linearization strategies were successful on each type of problem. These success rates give a very general indication of how difficult it was to converge a given problem using the twelve linearizations selected. Of course for a different set of linearization strategies, different results might be obtained. Nevertheless, while the results here are not conclusive, they are interesting and in many respects not surprising. First of all, we note that the overall success rate was not particularly good. Since "worst-case" initial guesses were used, this is not too surprising. It does indicate however, the need for better methods of internally generating initial guesses, and the need to consider other approaches, such as the dogleg method, for solving the nonlinear equations. Comparing the initial guess schemes we see that the heuristic approach of Type 3 is in general the best. The development of a better set of heuristics might further improve the success rate.

Comparing thermodynamics Cases 1 and 2 we note that in general Case 2 has a better success rate. This is as expected, since in Case 1 the Jacobian does not include deriatives of K-values with respect to temperature or pressure and thus does not provide an accurate linearization. It is interesting that on the isothermal flash problems (1,3,5,7,8,9), for which the K vs. T information is not important, Case 1 in general seems preferable to Case 2. As expected, it is clear that except in problems for which temperatures are specified (or closely guessed) for all equilibrium stages, it is desirable to linearize the models for K-value prediction and include these in the overall equation set. For Case 3 the results are incomplete, but suggest worse convergence behavior than Case 2. This observation is discussed in more detail below.

Table 4 summarizes the success rate for the various linearization strategies. The entries in this Table indicate how successful each linearization was for solving each problem using all the different combinations of initial guess type and thermodynamics case. For instance, linearization #5 was successful in six of the nine attempted solutions of problem 1. These success rates give a general indication of how reliable each linearization type was under a variety of circumstances. Of course for a different set of initial guesses different results might be obtained. Looking at these overall results, we note that the blended linearizations #6 and #7, which use an initial $\beta = 0$ and a $\Delta\beta = 0.1$, appear the most successful. For most problems, these strategies performed the best or nearly best overall. For some problems a different linearization seems appropriate however. For instance, #9 performed best on problem 7 and #12 best on problem 4. In fact, linearization #9 was the only strategy to solve all the problems at least once. If we focus just on the attempted solutions using guess Type 3, and thermodynamics Case 2 (the best combination from Table 3), linearizations #6 and #7 solve 70 and 80 percent of the problems respectively, but the norm reduction strategy #2 solves 90% of the problems. Thus, it is difficult to make any specific conclusions regarding the different linearizations. We can say in general however, that the blending of first and second order linearizations appears to be a useful approach. While an initial $\beta = 0$ and $\Delta\beta = 0.1$ seem to be relatively good blending parameters, in general it appears that the best blending parameters will vary from problem to problem. This

Table 3. Success rate (%) for different guess types and thermodynamics cases. See text for more detailed information.

Thermodynamics:	Case 1			Case 2			Case 3		
Guess Type:	1	2	3	1	2	3	1	2	3
Problem: 1	92	92	100	92	100	100	92	100	100
2	0	8	17	67	100	100	83	100	92
3	42	17	92	42	17	92	-	-	-
4	17	17	17	33	42	75	-	-	-
5	42	0	67	42	0	67	-	-	-
6	33	0	8	50	0	42	-	-	-
7	0	17	33	25	17	33	0	0	8
8	25	17	67	25	17	33	0	17	25
9	25	17	58	25	17	33	0	17	25
10	0	0	0	25	0	17	0	17	33
Overall	28	18	46	40	30	60	29	42	47

Table 4. Success rate (%) of different linearization strategies on the test problems. See text for more detailed explanation.

Problem Type:	DESIGN					SIMULATION					OVERALL
Flash:	Isothermal			Adiabatic		Isothermal			Adiabatic		
Prob. #:	1	3	7	2	4	8	5	9	6	10	
Linearization:											
1	100	0	11	67	0	0	33	0	17	0	24
2	100	33	11	67	17	11	33	11	17	0	31
3	100	33	0	67	33	11	33	0	17	0	29
4	100	33	11	56	33	11	33	11	33	0	32
5	67	67	22	67	17	11	33	11	17	0	31
6	100	100	0	67	50	78	66	78	33	44	62
7	89	100	0	67	67	78	66	78	50	44	63
8	100	67	0	67	33	22	33	33	17	11	38
9	100	33	33	67	33	56	33	44	33	22	47
10	100	33	11	56	17	22	0	22	0	0	28
11	100	33	22	56	17	0	33	0	0	0	27
12	100	67	22	56	83	0	33	0	33	0	37

suggests that, as noted in Gorczynski et al. (1979), it may be desirable to use different blending parameters for different parts of the process, rather than one set of parameters for the overall flowsheet. The "goodness" of the initial guess is another factor that must be considered in choosing the blending parameters. In general, for better initial guesses than those used here, a wider range of parameters will be successful and numerical efficiency will govern the choice of β and $\Delta\beta$, since it will be desirable to move more quickly into the pure second-order linearization. What is needed is a systematic procedure for selecting an appropriate set of blending parameters for a given situation.

Though the studies presented here focus on the reliability with which convergence is obtained from a poor initial guess, it is also appropriate to comment on numerical efficiency. For the problems solved here, the solution times and number of iterations vary with the guess type, linearization strategy, etc. For the flash with recycle problem a typical solution time is about 0.1 second and about seven iterations are needed. For the NH_3 problem, typical figures are 0.4 seconds and 15 iterations. For the Cavett problem, about sixteen to twenty iterations are needed and times range from about one second for the isothermal cases to about three seconds for the adiabatic case. These computational times were obtained on a CDC Cyber 175 computer using the FTN compiler under OPT = 1. The number of iterations required reflects the fact that blending in a first order linearization slows the rate of convergence. For this reason, as noted above, when better initial guesses are available, one should avoid staying in the blended mode for many iterations.

Problem Set 2

In this set of problems we look at the convergence behavior from a set of "best-case" initial guesses. These guesses were generated by taking the known solution and randomly perturbing the values in it a minimum of 5% and a maximum of 10, 15, 20, 25, or 30%. Thus five levels of initial guess accuracy were considered. For each problem, three different initial guesses were generated for each level of accuracy.

The problems solved were based on the ammonia and Cavett flowsheets discussed above. The thermophysical properties were handled using either Case 2 or Case 3, thus generating four problems as shown in Table 5. For further details on these and other problems solved see Hilton (1982). For this problem set, dew and bubble point determinations are included using either the Case 2 or Case 3

Table 5. Summary of test problems (Problem set 2).

Problem Number	Flowsheet	Specifications
1	NH_3 Synthesis; 5 components; Thermo. Case 2; # of variables = 165	$n_{1,1} - n_{1,5}$, T_1, P_1; T_4, P_4 ("compressor" output); T_6, T_7, T_8; T_{14}, P_{14} ("compressor" output); P and Q of both flashes; $n_{5,2} = 0.1\, n_{5,4}$ (this fixes the inerts into the reactor, etc. to be a fixed percentage of the active components)
2	As in 1 but Thermo. Case 3; # of variables = 234	As in 1.
3	Cavett Problem; 12 components; Thermo. Case 2; # of variables = 228	$n_{1,1} - n_{1,12}$, T_1, P_1, H_1; P and Q of all flashes
4	As in 3 but Thermo. Case 3; # of variables = 506	As in 3.

strategy. Thus, the largest problem solved now has 506 equations, which again does not tax the sparse matrix capabilities of the system. The linearization strategies used are the same as for the first set of problems, except that the damped Newton-Raphson strategies 2 and 3 use somewhat more sophisticated norm reduction schemes, as detailed in Table 6. Each of the four problems was solved using the twelve linearization strategies on each of the eighteen different initial guesses. Thus a total of 864 solutions were attempted.

Table 7a summarizes the success rate for the various "best-case" guess types and the two thermodynamics cases. The entries in this table indicate how many solutions out of a possible six were successful. Table 7b compares the overall performance of the Newton-Raphson linearizations (1-5) to the hybrid linearizations (6-12) for the five guess types. Two trends are apparent. Again we note that the Case 2 strategy for computing the thermophysical properties seems more reliable than Case 3. It is not clear why this should be so. Perhaps it is due to the fact that for Case 3 the thermophysical property models are not converged until the end, while for Case 2 they are converged after every iteration. However, this is not a particularly satisfactory explanation. A more probable explanation is that Case 2 is more reliable because the thermophysical property subroutines can employ special-purpose solution algorithms known to be fairly robust for a particular problem, while for Case 3 a general-purpose solution algorithm is applied to the entire equation set. The second trend to note is that the Newton-Raphson-based strategies (1-5) in general perform better than the blended first-second order linearization (6-12). This would seem to indicate that since we are starting near

Table 6. Summary of linearization strategy changes (problem set 2).

Method: Newton-Raphson with step size relaxation

Strategy Number:	Evaluation of relaxation parameter r:
2	$r = \dfrac{(1+6n)^{0.5} - 1}{3n}$ where $n = \underline{f}_i^2/\underline{f}_0^2$ and if $r < 0.05$, $r = 0.05$
3	$r_i = \dfrac{2\underline{f}_{i-1}^2}{2(\underline{f}_i^2 + \underline{f}_{i-1}^2)}$ where if $r < 0.1$, $r = 0.1$. If the norm of f is not reduced: $r_{i+1} = \dfrac{2r_i^2\, \underline{f}_{i-1}^2}{2(\underline{f}_i^2 + \underline{f}_{i-1}^2(2r_i - 1))}$ where if $r_{i+1} < 0.1\, r_i$, $r_{i+1} = 0.1 r_i$
2-5	When norm of $\underline{f} < 10$, $r = 1$.

Table 7 a. Success rate (convergences out of 6) of different linearization strategies (problem set 2).
b. Success rate (%) for Newton-Raphson type linearizations and hybrid type linearizations.

(a)

Guess Accuracy:	10%		15%		20%		25%		30%		
Thermo. Case:	2	3	2	3	2	3	2	3	2	3	Overall Success (%)
Linearization: 1	6	4	6	4	6	2	4	1	3	2	63
2	6	6	6	6	5	5	5	3	5	5	87
3	6	5	6	5	6	5	4	4	4	3	80
4	6	5	6	3	6	3	3	2	6	2	70
5	6	6	6	5	5	4	5	2	4	2	75
6	5	3	3	4	3	3	3	2	3	2	52
7	4	3	3	3	2	3	2	3	3	2	47
8	5	5	6	3	4	1	3	0	5	4	60
9	4	4	6	4	5	2	2	2	4	2	58
10	5	5	6	3	5	2	4	3	4	3	67
11	6	5	6	3	4	2	3	1	3	4	62
12	3	4	2	1	1	0	3	1	2	-0	28
Total(%)	86	76	86	61	72	44	57	33	64	43	

(b)

Guess Accuracy:	10%		15%		20%		25%		30%	
Thermo. Case:	2	3	2	3	2	3	2	3	2	3
Linearization Type:										
Newton-Raphson (Overall % Success for linearizations 1-5)	100	87	100	77	93	63	70	40	73	47
Hybrid (Overall % Success for linearizations 6-12)	76	69	76	50	57	31	48	29	57	40

the solution, the more accurate second-order linearization should be used from the start. Overall the damped Newton-Raphson strategies 2 and 3 perform quite well. However there is still room for improvements in robustness even from relatively good initializations. We believe the dogleg approach (Powell, 1970; Chen and Stadtherr, 1981) to be particularly promising in this regard.

CONCLUDING REMARKS

A new equation-based flowsheeting system, SEQUEL, is being developed. The system uses a simple executive routine that efficiently generates the equations to be solved, and is interfaced to a set of powerful sparse matrix routines capable of solving several thousand equations without problem decomposition. For larger problems decomposition may be required. Though SEQUEL does not currently have a decomposition capability, the techniques given by Westerberg and Berna (1978) and Stadtherr and Hilton (1982) may be used in this context.

SEQUEL is designed for use in studying

various computational strategies for equation-based flowsheeting. The studies discussed here indicate that simple heuristics can be developed to improve internally generated initial guesses when the user supplies no such information externally. Also the blending of first and second order linearizations appears to be a useful approach to improve convergence from poor guesses. However, choosing the best set of blending parameters is not straightforward. For good initial guesses, as well as some poor guesses, a damped Newton-Raphson approach appears quite effective.

ACKNOWLEDGEMENT

This work has been supported by the National Science Foundation under Grant CPE 80-12428.

LITERATURE CITED

Bending, M. J. and H. P. Hutchison, "The Calculation of Steady State Incompressible Flow in Large Networks of Pipes," Chem. Eng. Sci.. 28, 1857 (1973).

Benjamin, D. R., M. H. Locke, and A. W. Westerberg, "Interactive Programs for Process Design," pres. at AIChE Meeting, Detroit, August 1981.

Book, N. L. and W. F. Ramirez, "The Structural Analysis and Solution of Systems of Algebraic Design Equations," presented at AIChE National Meeting, Philadelphia, June 1978.

Chen, H. S. and M. A. Stadtherr, "A Modification of Powell's Dogleg Method for Solving Systems of Nonlinear Equations," Comput. Chem. Eng., 5, 143 (1981).

Coup, T. G., E. H. Chimowitz, A. Blonz and L. F. Stutzman, "An Equation Analyzer Package for the Manipulation of Mathematical Expressions - I," Comput. Chem. Eng., 5, 151 (1981).

Evans, L. B., "Advances in Process Flowsheeting Systems," in Foundations of Computer-Aided Chemical Process Design, Vol. 1, (ed. R. S. H. Mah and W. D. Seider), Engineering Foundation, New York (1981).

Gorczynski, E. W. and H. P. Hutchison, "Towards a Quasilinear Process Simulator: I. Fundamental Ideas," Comput. Chem. Eng. 2, 189 (1978).

Gorczynski, E. W., H. P. Hutchison, and A. R. M. Wajih, "Development of a Modularly Organized Equation-Oriented Process Simulator," Comput. Chem. Eng. 3, 353 (1979).

Hanyak, M. E., "Textual Expansion of Chemical Process Flowsheets into Algebraic Equation Sets," Comput. Chem. Eng. 4, 223 (1980).

Hellerman, E. and D. Rarick, "Reinversion with the Preassigned Pivot Sequence," Math. Programming, 1, 195 (1971).

Hilton, C. M., "Numerical Studies in Equation-Based Chemical Process Flowsheeting," Ph.D. Thesis, University of Illinois (1982).

Hlavacek, V. "Analysis of a Complex Plant - Steady State and Transient Behavior," Comput. Chem. Eng., 1, 75 (1977).

Hutchison, H. P. and C. F. Shewchuk, "Computational Method for Multiple Distillation Towers," Trans. I. Chem. Eng., 52, 325 (1974).

Kubicek, M., V. Hlavacek, and F. Prochaska, "Global Modular Newton-Raphson Technique for Simulation of an Interconnected Plant Applied to Complex Rectification Columns," Chem. Eng. Sci., 31, 277 (1976).

Leigh, M. J., G. D. D. Jackson, R. W. H. Sargent, "SPEED-UP - A Computer-Based System for the Design of Chemical Processes," presented at CAD-74, Imperial College, London, Sept. 24-27, 1974.

Lin, T. D. and R. S. H. Mah, "A Sparse Computation System for Process Design and Simulation: I. Data Structures and Processing Techniques," AIChE J., 24, 830 (1978).

Lin, T. D. and R. S. Mah, "Hierarchical Partition - A New Optimal Pivoting Algorithm," Math Programming, 12, 260 (1977).

Mah, R. S. H. "Pipeline Network Calculations Using Sparse Computation Techniques," Chem. Eng. Sci., 29, 1629 (1974).

Mah, R. S. H. and T. D. Lin, "A Sparse Computation System for Process Design and Simulation: II. A Performance Evaluation Based on the Simulation of a Natural Gas Liquefaction Process," AIChE J., 24, 839 (1978).

Motard, R. L., M. Sacham, and E. M. Rosen, "Steady State Chemical Process Simulation," AIChE J., 21, 417 (1975).

Powell, M. J. D., "A Hybrid Method for Nonlinear Equations," in Numerical Methods for Nonlinear Algebraic Equations, (ed. P. Rabinowitz), Gordon & Breach, New York (1970).

Reid, R. C., J. M. Prausnitz, and T. K. Sherwood, The Properties of Gases and Liquids, 3rd ed., McGraw-Hill, New York (1977).

Rosen, E. M., "Steady State Chemical Process Simulation - A State-Of-The Art Review," in Computer Applications to Chemical Engineering, ACS Symp. Ser. No. 124 (ed. G. V. Reklaitis and R. G. Squires), ACS, Washington, D.C. (1980).

Stadtherr, M. A., "Maintaining Sparsity in Process Design Calculations," AIChE J. 25, 609 (1979).

Stadtherr, M. A. and C. M. Hilton, "On Efficient Solution of Large-Scale Newton-Raphson-Based Flowsheeting Problems in Limited Core," Comput. Chem. Eng., in press (1982).

Stadtherr, M. A. and E. S. Wood, "Exploiting Border Structure in Solving Large Sparse Linear Systems in Bordered Triangular Form," Comput. Chem. Eng. 4, (1980).

Westerberg, A. W. and T. J. Berna, "Decomposition of Very Large-Scale Newton-Raphson Based Flowsheeting Problems," Comput. Chem. Eng. 2, 61 (1978).

Westerberg, A. W. and S. W. Director, "A Modified Least Squares Algorithm for Solving Sparse n x n Sets of Nonlinear Equations," Comput. Chem. Eng. 2, 77 (1978).

Westerberg, A. W., H. P. Hutchison, R. L. Motard and P. Winter, Process Flowsheeting, Cambridge University Press, Cambridge (1979).