THE NUMERICAL SOLUTION OF NONLINEAR EQUATIONS REPRESENTING CHEMICAL PROCESSES

by

Jorge Ricardo PALOSCHI

July 1982

A thesis submitted for the degree of Doctor of Philosophy of the University of London and for the Diploma of Membership of the Imperial College.

Department of Chemical Engineering
and Chemical Technology,
Imperial College of Science and Technology,
London SW7

To Dorita, Richi, Moncho and Diego.

ACKNOWLEDGEMENT

I would like to express my gratitude to Dr J Perkins for his invaluable help and encouragement during the present work.

I am very grateful to the following institutions for the financial support received:

- Consejo Nacional de Investigaciones Cientificas y Tecnicas (Argentina)
- Universidad Nacional del Sur (Bahia Blanca - Argentina)
- United Nations Industrial Development Organisation.

The assistance received from the SPEEDUP team at Imperial College as well as the discussions held with D Bogle are particularly appreciated.

Finally, a last word of gratefulness for all those who made possible my coming to Imperial College: Ing M J Urbicain, Dr E Rotstein, Prof J R A Pearson and all my colleagues at Planta Piloto.

ABSTRACT

The solution of systems of nonlinear algebraic equations is a fundamental mathematical problem in steady state process design and simulation in Chemical Engineering. This thesis is related to this problem.

Some new ideas for implementing Quasi-Newton methods are discussed. In particular, a modification to Bennett's algorithm (to obtain an LU factorization of Jacobian approximations) is proposed to avoid numerical singularities.

Conditions for having scale invariant Quasi-Newton methods are presented as well as the convergence properties. Four scale invariant methods are proposed.

The problem of the numerical conditioning of Quasi-Newton methods is discussed and some ideas to improve it are proposed. In particular, an internal scaling procedure is devised.

All our proposals are tested and compared with others. For this a wide standard set of mathematical examples is used as well as Chemical Engineering flowsheeting examples.

CONTENTS

NOMENCLATURE

			first used	
			page	equation
Scalars:	λ_k	step reduction factor	27	2.9
	ξ_{ik}	component i of vector x_k	32	
	π_{ik}	" " p_k	32	
	ν_{ik}	" " y_k	55	
	κ	condition number	67	4.1
Vectors:	e_k	k-th column of the unit matrix	8	1.5
	x_k	variables vector	16	
	x_*	solution vector	17	1.19
	f_k	functions vector	27	2.4
	p_k	Quasi-Newton step	17	1.25
	y_k	the vector $(f_{k+1}-f_k)$	27	2.5
	v_k	the vector which determines the Quasi-Newton method	27	2.8
Matrices:	I_k	unit matrix	35	
	$F'(x_k)$	Jacobian of $f(x_k)$	17	
	B_k	approximation to $F'(x_k)$	17	
	H_k	inverse of B_k	30	2.15
	L_k	lower triangular matrix	31	2.17
	U_k	unitary upper triagular matrix	31	2.17
	D_f	external scaling matrix(functions)	20	1.28
	D_x	" (variables) (diagonal in sections 3.2 and 3.3)	20	1.27
	D_1	diagonal internal scaling matrix for the functions	78	
	D_2	diagonal internal scaling matrix for the variables	78	
General:	{}	denotes a sequence	16	
	|| ||	denotes a norm(Frobenius norm for matrices and Euclidean norm for vectors unless otherwise specified)	15	
	^	denotes the scaled space	20	

CHAPTER 1

INTRODUCTION

The solution of systems of non-linear algebraic equations is a fundamental mathematical problem in steady-state process design and simulation in Chemical Engineering.

While we can study the problem from a mathematical point of view, specialized approaches have been developed for solving flowsheeting problems.

We will use a very simple chemical process, shown in Fig 1.1, in order to help the presentation of our discussion.

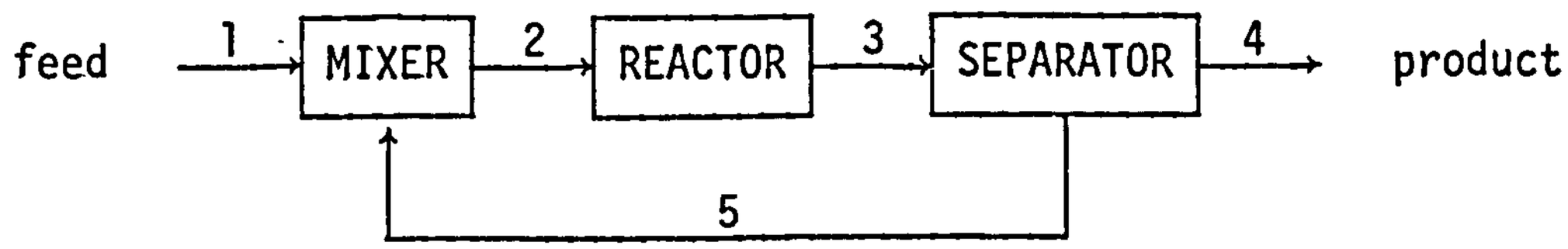


Fig 1.1 A simple chemical process

If we consider only a mass balance for our example, we can establish an approximate mathematical model for the process. Assume there are m chemical components involved in this process, we can then use vectors a_i of m components to represent the molar flows for stream i. For the mixer we can write m equations as

$$a_2 = a_1 + a_5 \qquad 1.1$$

using vector arithmetic $\mathbb{R}^m$.

To represent the reactor we need an additional vector $s \in \mathbb{R}^m$ containing the stoichiometric coefficients for the particular reaction taking place plus a scalar γ representing the conversion factor of the key component (which we will assume is the first). With this we can write for the reactor

$$a_3 = a_2 + \gamma(e_1^T a_2)s \qquad 1.2$$

which gives us another m equations.

Finally, if we use a vector p of m components to represent the fraction of the feed to the separator going as product, we can write

$$a_5 = a_3 - a_4 \qquad 1.3$$

$$a_4 = p \cdot a_3 \qquad 1.4$$

where the vector operation $p \cdot a_3$ means the arithmetic product of the components, ie

$$e_i^T(p \cdot a_3) = (e_i^T\, p)(e_i^T\, a_3) \qquad 1.5$$

Therefore we have 4m equations involving 7m + 1 variables which means our mathematical model has 3m + 1 degrees of freedom.

In order to have a consistent model we will have to eliminate the 3m + 1 degrees of freedom and then if we are doing a strictly simulation problem we will fix a_1, s, p and γ and find out a_2, a_3, a_4 and a_5 using equations 1.1 to 1.4.

We can see that in fact we have a mathematical problem to solve, that is to solve a system of 4m algebraic equations (which for our particular example happen to be linear, but it will not be so in general).

Even for our very simple example, we can see that if m is large then the size of the mathematical problem to be solved becomes an important factor. Two decades ago, when computers began to be used for solving simulation problems, storage was a critical factor (as well as number of operations) and then specialised approaches were devised to tackle the simulation and design problem.

1.1 THE SEQUENTIAL MODULAR APPROACH

Since solving a flowsheeting problem by simultaneous solution of all the equations involved resources which were not available two decades ago (numerical methods, computer storage, speed of computations), the sequential modular approach was proposed. MOTARD, SHACHAM and ROSEN

(1975) reviewed the state of the art for this approach at the beginning of the 70's when several packages were available as well as a considerable amount of literature.

In this approach each unit is represented in the computer as a subroutine in such a way that the output streams are evaluated provided the input streams plus the equipment parameters are known.

For our particular example three subroutines should be available as:

$$(a_2)\ \text{MIXER}\ (a_1, a_5) \qquad 1.6$$

$$(a_3)\ \text{REACTOR}\ (a_2, s, \gamma) \qquad 1.7$$

$$(a_4, a_5)\ \text{SEPARATOR}\ (a_3, p) \qquad 1.8$$

where the arguments on the left of the name of the subroutine are the output parameters while the ones on the right are the input parameters.

Since the input parameters must be known in order to use each subroutine, due to the recycle stream 5, it is not possible to use the subroutines since none of them has all its inputs known.

The key element for this approach was the introduction of a vector x and the replacement of 1.6 by

$$(a_2)\ \text{MIXER}\ (a_1, x) \qquad 1.9$$

If we have a value for x we can, by going sequentially through 1.9, 1.7 and 1.8, obtain a value for a_5, ie we can define a function $g : \mathbb{R}^m \Rightarrow \mathbb{R}^m$ such that

$$a_5 = g(x) \qquad 1.10$$

Now, if we can solve the mathematical problem

$$x = g(x) \qquad 1.11$$

we can solve the original flowsheeting problem since the solution of 1.11 coincides with the solution in the flowsheet.

The mathematical problem 1.11 is known as a fixed point problem and it has been well studied (see ORTEGA and RHEINBOLDT (1970), Chapter 5) from a theoretical point of view.

Sufficient conditions for existence of a solution to 1.11 are simple to enunciate. If $C \subseteq \mathbb{R}^m$ is a convex, compact set and $g(x)$ is continuous in C then 1.11 will have at least one solution in C provided

$$g(x) \in C, \forall\ x \in C \qquad 1.12$$

Unfortunately, it is not simple to show that a general function g satisfies 1.12.

In solving flowsheeting problems the first method used to solve 1.11 was successive substitution in which a sequence $\{x_n\}$ is generated, given an initial estimate x_o, as

$$x_{n+1} = g(x_n) \qquad 1.13$$

It was found that the convergence of this method was generally slow (linear) and then acceleration methods were used. WEGSTEIN's (1958) method was first used. ORBACH and CROWE (1971) proposed an accelerating procedure, later modified by CROWE and NISHIO (1975), known as the dominant eigenvalue method which relies on the linear convergence of 1.13 to extrapolate a value for x_∞ using it as a new approximation. These accelerating procedures can be applied not only to the successive substitution method but to any method having a linear rate of convergence.

While this approach succeeds in reducing considerably the size of the mathematical problem to be solved, it presents difficulties in solving design problems. Suppose that instead of fixing a_1, s, p and γ in our example, we want to fix a_1, s, p and $e_1^T a_4$, ie we want a desired product rate for the first component and to find out what conversion we should have in the reactor (γ).

To handle this problem with this approach control blocks were introduced. An additional subroutine is available as

$$(\gamma)\ \text{CONTROL}\ (\alpha_{41}) \qquad 1.14$$

where $\alpha_{41} = e_1^T a_4$, which for a given α_{41} solves the previous simulation problem obtaining a value for γ. This allows us to define a new function $\psi(\alpha_{41})$ for which the solution of our design problem will be

$$\gamma = \psi(\alpha_{41}) \qquad 1.15$$

The problem 1.15 is a general non-linear equation which can be solved with any method. The important point is that to obtain one value of 1.15 we must solve one problem like 1.11 and since a method for solving 1.15 will be in general iterative this means we are adding another level of iteration.

It has been proposed by JOHNS (1970) that 1.11 and 1.15 should be solved simultaneously in order to avoid nested iterations.

METCALFE and PERKINS (1978) and PERKINS (1979b) proposed the use of BROYDEN's (1965) method to converge simultaneously 1.11 and 1.15. It was tried successfully solving flowsheeting problems using CHESS (MOTARD and LEE (1971)). More results were later obtained by MALATHRONAS and PERKINS (1980).

GROSSMANN and del ROSAL (1978) did a comparison of successive substitution, Wegstein, Broyden and the dominant eigenvalue method and found the best results for the last two.

In the last ten years there has been a considerable amount of work in finding methods for solving the mathematical problem 1.11 (see the text book by TODD (1976) and SAIGAL (1979)). There are not yet general codes available for testing them on flowsheeting problems but this offers an interesting field for research.

1.2 THE EQUATION ORIENTED APPROACH

The limitations of the sequential modular approach in coping with design problems, plus the advance in computer hardware which has made it possible to solve problems faster and using considerable storage needs has encouraged the development of the equation oriented approach.

In this approach, each unit in a flowsheet is represented using equations, as we did at the beginning of this chapter, and then a system of non-linear algebraic equations must be solved.

In our particular example, the system to be solved will consist of the equations defined by 1.1 to 1.4. The important aspect of this approach is that there is no difference between a simulation or a design problem since the system of equations will always be the same. The difference will reside in which variables in the equations will be considered unknowns. In a simulation problem we will have as variables a_2, a_3 a_4 and a_5; in a design problem we can have as variables, for instance, a_2, a_3, the last m-1 components of a_4, a_5 and the conversion γ. The only restrictions are that the number of variables is the same as the number of equations and that the system is consistent (ie it has at least one solution).

KUBICEK, HLAVACEK and PROCHASKA (1976) proposed the use of the Newton-Raphson method in solving flowsheeting problems using a library automatically generating residual vectors (ie the equations representing each unit) and Jacobian matrices.

VANEK, HLAVACEK and KUBICEK (1977) solved separation problems using the equation oriented approach and a non-linear block successive relaxation method while SALGOVIC, HLAVACEK and ILAVSKY (1981) used parameter imbedding techniques to solve the simulation of counter current separation processes.

All these works implement the equation oriented approach based on automatic generation of equations and derivatives which, while being very convenient for particular cases, is very restrictive for general problems.

A quasi-linear approach has been proposed by GORCZYNSKI and HUTCHISON (1978) in which the equations representing the flowsheet are written as

$$A(x)\ x = b(x) \qquad 1.16$$

where the matrix A(x) and the vector b(x) are considered constants, solving 1.16 and then re-evaluating them. This technique has been tested by HILTON and STADTHERR (1981) in the SEQUEL equation oriented package.

When the system of equations to be solved is large, it will generally be sparse. If we define an occurrence matrix for the system as a matrix having as many rows as equations and columns as variables, an entry being 1 if the corresponding variable appears in the corresponding equation 0 otherwise. If the system is sparse then this occurrence matrix will have a low density of ones (say O(n), where n is the number of equations). Sometimes, it will be possible to re-number the equations and variables such that it is possible to decompose the original system into smaller subsystems which can be solved sequentially.

A detailed discussion of this was given by SARGENT (1977).

Another important aspect of this approach is that we can add costing equations to the ones representing the units and then solve a more general optimization problem.

$$\begin{aligned} &\text{maximize} \quad \gamma(x) \\ &\text{subject to} \quad f(x) = 0 \\ &\text{and} \quad g(x) \leq 0 \end{aligned} \qquad 1.17$$

where f(x) will be our original system of equations representing the units and $\gamma(x)$ will be the objective function plus a set of restrictions given by g(x).

1.3 MIXED APPROACHES

This approach is a blending of the two approaches we mentioned before using the best of each of them. ROSEN (1962) proposed to use a combination of simple linear models and the equation oriented approach plus rigorous models given as procedures. His algorithm could be briefly put as (see MAHALEC, KLUZIK and EVANS (1979)):

1 Assume properties for all streams (provides inputs for all process units).

2 Compute outputs of every unit using rigorous models given as procedures.

3 Develop linear models for each unit.

4 Based on 3 form a linear system of equations representing the entire process and solve it (this provides new values for all process streams).

5 Test for convergence and stop if achieved.

6 Substitute the linear solution for the stream variable (obtained in 4).

7 Go to 2.

Rosen proposed to use diagonal matrices for the linear models neglecting then interaction between variables. This causes instabilities according to MAHALEC, KLUZIK and EVANS (1979). They proposed to use diagonal matrices for those units in which the interaction between variables is small or nil (mixers, dividers, heat exchangers, pumps, distillation units, etc) and full matrices for those with interaction (such as reactors). These matrices were obtained by finite difference approximations.

The algorithm proposed in this latter case can be stated as:

1 Assume initial values for process variables.

2 Test for convergence and stop if achieved.

3 Solve the linearized systems of equations

$$B\Delta x = -f$$

(where B is the linear approximation, f the function vector and Δx the current step for the variables vector).

4 If $||f(x + \Delta x)|| < ||f(x)||$ go to 7.

5 Define $f_0 = ||f(x)||$, $f_1 = ||f(x + \Delta x)||$, $f_2 = ||f(x + 0.5\Delta x)||$ fit a quadratic function through f_0, f_1 and f_2 and find the coefficient α such that the quadratic has an extreme value (ie, zero gradient).

6 If $||f(x + \alpha\Delta x)|| > ||f(x)||$ go to 8.

7 $x = x + \alpha\Delta x$, go to 2.

8 Abandon current linearization B. Make one sequential pass through the flowsheet, obtain new values for process variables, find the new linearization B and go to 2.

9 $x = x + \Delta x$, go to 2.

Thus this approach first uses the equation oriented approach with the simpler linear models and only switches to the procedures implementing the rigorous models when it is not making good progress.

We should put also in this section the idea of METCALFE and PERKINS (1978) to iterate simultaneously on stream and design variables (ie, eliminating the nested iterations of the control loop in the sequential modular approach). This was tested with general problems by PERKINS (1979b) and MALATHRONAS (1979) using the simulation package CHESS (MOTARD and LEE (1971)) and Broyden's method.

We should also mention here the idea of JOHNS (1970) of iterating simultaneously on recycle streams and internal iterations within units.

JOHNS and SHEPPARD (1975) implemented this idea to solve distillation column iterations simultaneously with flowsheet evaluations. This was later extended by LEE (1980) who solved simultaneously the set of torn streams, design equation and the internal equations of the distillation unit. For this he used CHESS plus the sparse Quasi-Newton method proposed by SCHUBERT (1970). The same was done by WONG (1981) for heat exchanger units using also CHESS and Broyden's method.

The simulation package SPEED-UP (SARGENT, PERKINS and THOMAS (1982)), developed at Imperial College, is another example of a mixed approach. By allowing the use of procedures, as well as equations, to represent the units, it is possible to describe a problem as a sequence of procedures (by having a procedure for each unit) or as an equation oriented problem (having all units represented by equations). In addition we have all possible intermediate steps.

This package makes possible the use of procedures written for sequential modular packages, allowing users to interface them to SPEEDUP very easily (all that is needed is to describe the parameter list). This is a very important aspect of a package since most industrial software for flowsheeting is written for sequential modular packages.

Whatever approach is chosen, the basic mathematical problem remains the same, ie solving a system of non-linear algebraic equations.

In particular, the requirements in this sense of the simulation package SPEED-UP have motivated the work presented in this thesis.

Our aim has been to obtain a code for solving the general problem.

$$f(x) = 0 \qquad 1.18$$

for a given function $f : \mathbb{R}^n \Rightarrow \mathbb{R}^n$ whose dependence on data that has to be provided(initial guesses, scaling)is minimized.There are intrinsic difficulties in Chemical Engineering problems which we would like to overcome such as singularities(due probably to poor initial data such as in a flash having a temperature below the bubble point)or scaling(having to deal with values differing by orders of magnitude between them).
We will use mathematical examples in order to find out which methods of the ones we will propose give good performance. The reason for doing this is the cost involved in trying to test thoroughly different methods using examples arising from flowsheeting problems. Once the best methods are identified, we will test them on flowsheeting problems.

1.4 SOLVING SYSTEMS OF NONLINEAR ALGEBRAIC EQUATIONS

Methods for solving 1.18 are usually iterative. They generate a sequence $\{x_k\}$, $x_k \in \mathbb{R}^n$, which if the method converges satisfies

$$\lim_{k \to \infty} x_k = x_* \qquad 1.19$$

$$f(x_*) = 0 \qquad 1.20$$

If we use the notation F'(x) for the Jacobian matrix of f evaluated at x it was observed that if f is a linear function, the solution x_* can be found as

$$x_* = x + p^N \qquad 1.21$$

where

$$p^N = -F'(x)^{-1} f(x) \qquad 1.22$$

provided $F'(x)^{-1}$ is well defined.

If f is not a linear function but x_k is a point sufficiently near x_*, we can expect that defining

$$x_{k+1} = x_k + p_k^N \qquad 1.23$$

will provide a better approximation to x_*, ie

$$\|x_{k+1} - x_*\| < \|x_k - x_*\| \qquad 1.24$$

The iteration process 1.23 is the Newton method and thus p_k^N is called the Newton step.

This method has been the basis for many successful methods developed in the past. The computation of F'(x) is often expensive or even impossible, therefore methods have been devised for avoiding it.

Instead of $F'(x_k)$ an approximation B_k to it has been used, giving then instead of 1.23

$$p_k = -B_k^{-1} f(x) \qquad 1.25$$

$$x_{k+1} = x_k + p_k \qquad 1.26$$

A first method was devised by taking B_k as a finite difference approximation to $F'(x_k)$. (Discrete Newton's method.)

BROWN (1966) proposed a method similar to the discrete Netwon's method but requiring less function evaluations and BRENT (1973) proposed a method with similar characteristics.

BROYDEN (1965) proposed a method in which B_k, while aiming to approximate $F'(x_k)$, is not obtained by finite differences. This method is a particular case of a more general family which has received the name Quasi-Newton methods. DENNIS and MORE (1977) have given a detailed study of this family. For the purpose of our discussion we will define three families of methods according to the sharing of similar characteristics.

D-family: discrete Newton's method, Newton-Raphson method (using 1.22 instead of 1.25).

B-family: Brown and Brent's methods.

Q-family: Broyden's method, all Quasi-Newton methods.

We will now analyse the three families according to the characteristics which are important to us.

Convergence: This is related to 1.19 and with the "speed" of the method to reach x_*. Convergence properties are normally local, ie applicable to some open set containing the solution x_* (which means x_o must be in that open set). The speed of convergence is often measured in terms of a sequence $\{\alpha_k\}$ and a scalar β such that

$$\|x_{k+1} - x_*\| \leq \alpha_k \|x_k - x_*\|^\beta$$

for a given norm.

Three rates of convergence are often used:

linear $\alpha_k = \alpha,\ 0 < \alpha < 1,\ \forall_k,\ \beta = 1$

superlinear: $\lim_{k\to\infty} \alpha_k = 0,\ \beta = 1$

quadratic: $\alpha_k = \alpha,\ \forall_k,\ \beta = 2$

Family D has quadratic convergence provided the finite difference steps are chosen carefully, (ORTEGA and RHEINBOLDT (1970)) as well as family B (MORE and COSNARD (1979)). Family Q satisfies superlinear convergence (DENNIS and MORE (1977)). Also 2n - step quadratic convergence, ie

$$\| x_{k+2n} - x_* \| < \alpha \| x_k - x_* \|^2$$

(see GAY (1979)) has been proved for Broyden's method.

Function evaluations: All methods will require the evaluation of 1.18 at a certain number of points. Some methods require the evaluation of each component of the vector f separately and other the evaluation of all components simultaneously. This is of primary importance when solving flowsheeting problems in chemical engineering since the evaluation of 1.18 involves normally a considerable amount of work. When physical properties are used, the time used for the function evaluation will overtake considerably the overhead time used for iteration for the method. Families D and Q require the evaluation of the whole vector f while family B requires the evaluation of each component separately. This difference is very important since there is normally a fixed basic time which is shared by all the components if evaluated simultaneously (especially when physical properties are being used).

In order to compare the methods we will use component function evaluations. The requirements for each method per iteration are:

Family	Component function evaluations
D	$n^2 + n$
B	$(n^2 + 3n)/2$
Q	n

The component function evaluations for family D are for the discrete Newton's method.

If the Jacobian is provided analytically then only n component function evaluations are needed.

Storage: This is related to the amount of computer storage used for implementing the method, the unit is the memory needed for storing one real number. Families D and Q require n^2 storage locations while on family B Brown's method can be implemented using only $(n^2/4)$ and Brent's method using n^2 (see COSNARD (1975)). This storage is basically for storing the Jacobian, all methods use in addition O(n) storage locations for auxiliary vectors.

Operations count: This is related to the number of arithmetic operations performed per iteration needed for implementing the method and it is directly related to the computer time to be used by the method.

The following is the operation count for each family.

Family	Number of arithmetic operations
D	$O(n^3)$
B	$O(n^3)$
Q	$O(n^2)$

Linear subsystem: This is an important aspect in chemical engineering. If part of the function 1.18 is composed of linear functions then all methods will satisfy them (see MORE and COSNARD (1979) for families D and B, PERKINS (1979a) for family Q).

Since most mass balances will be represented by linear equations we can expect a significant part of 1.18 to be linear when solving flow-sheeting problems. The linear subsystems property guarantees that every iterate generated satisfies these equations.

Scaling: An ideal code should be independent of the scaling used for the variables or the functions. We define a change of scale for the variables as

$$\hat{f}(\hat{x}) = f(D_x^{-1}\,\hat{x}) \qquad 1.27$$

for a nonsingular matrix D_x, and a change of scale for the function as

$$\hat{f}(x) = D_f\, f(x) \qquad 1.28$$

And more generally, for both simultaneously

$$\hat{f}(\hat{x}) = D_f\, f(D_x^{-1}\, \hat{x}) \qquad 1.29$$

We can define scale independence properly as

DEFINITION 1: Scale invariant methods.
An iterative method for solving 1.5 will be scale invariant if for nonsingular matrices D_x and D_f defining the change of scale 1.29, the sequence $\{x_k\}$ generated by the method satisfies

$$\hat{x}_k = D_x\, x_k \qquad 1.30$$

Methods in family D are scale invariant, family B is not scale invariant (see MORE and COSNARD (1977)) and neither is family Q(MALATHRONAS and PERKINS(1980)).Having a scale invariant method is particularly important in Chemical Engineering since is not reasonable to ask a user to scale the original problem in order to present it in a suitable scale for the code.It is very common in Chemical Engineering to find values in a simulation differing by several orders of magnitude(as in flowrates for instance).
In addition to the theoretical aspects of the families we should also consider the numerical results available in the literature.

Authors have generally stressed efficiency when presenting numerical results, ie how fast does a method converge. A very important aspect to us is robustness, ie the ability of a method to converge from initial points far from the solution and in bad numerical conditions (which are often closely related). Our previous discussion on convergence is related to a neighbourhood of the solution and thus is not applicable in the first stage when we are far from the solution. This means that more important than having a different rate of convergence (say between superlinear and quadratic) will be having a better radius of convergence(or being more robust).We are interested in robustness since while it is important to obtain the solution quickly it is even more important to obtain it at all(even if this is not a fast process). Tests for robustness are comparatively rare.

Three recent studies on robustness are available. BUS (1975) tested various codes finding the best results for the discrete-Newton, BROWN (1966) and BROYDEN (1965) method.

PALOSCHI (1980) obtained good results also for BROYDEN (1965), BROWN (1966) and for the implementation of GRAGG and STEWART (1974) of the secant method. (While results were very good regarding robustness for this latter method, it was found to be poor in efficiency and it requires $4n^2$ storage locations.)

A method specially devised for robustness was proposed by BROYDEN(1969) based on continuation methods.This was tested by PALOSCHI(1980) without obtaining satisfactory results.In appendix C',numerical results obtained for this method using the same set of examples as in the next chapters are presented.These appear to confirm our earlier findings that the method is not very robust.

The most comprehensive test was done by HIEBERT (1980). Eight codes implementing four different methods were tested. Two sets of problems were considered: a set of 57 "mathematical problems" and one consisting of 22 chemical equilibrium problems. Tests for the effect of scaling on the performance of the codes were designed for the mathematical problems. For testing the performance on badly scaled variables the original problems were modified to

$$\hat{f}(\hat{x}) = f(D^{-1}\,\hat{x})$$

and for badly scaled functions

$$\hat{f}(x) = D\,f(x)$$

with D being a diagonal matrix whose elements were between 10^{-5} and 10^5.

The Hybrid method tested by Hiebert can be described very briefly as follows;

Define the gradient step

$$g_k = -B_k^T\,f(x) \qquad 1.31$$

This is the steepest descent direction and then a hybrid step p_k^h is obtained from

$$p_k^h = \alpha_k\,p_k + \beta_k\,g_k \qquad 1.32$$

where p_k is the Quasi-Newton step 1.25.

The parameters α_k and β_k are chosen according to the "progress" which the code is making, ie if the code is making good progress

(ie $\|f_{k+1}\| < \|f_k\|$) then $\beta_k \to 0$ and $\alpha_k \to 1$, $\beta_k \neq 0$ is only used when the code is not behaving very well. The idea is that the gradient step is more robust while the Quasi-Newton step is more efficient.

Regarding the theoretical properties of this method, they are the same as family Q for convergence, function evaluations and operations count. The storage required for this method is $\frac{3}{2} n^2$ for the implementation tested by Hiebert (but it can be implemented using only n^2 as in family Q, see CHEN and STADTHERR (1981)). If in 1.32 $\beta_k \neq 0$ then the method will not satisfy linear subsystems as the following shows:

Consider the linear function

$$f(x_k) = Ax_k + b$$

and take $B_o = A$ (ie, the exact Jacobian). Assume $\alpha_k = 0$, $\beta_k = 1$ in 1.32

$$p_o^h = -A^T f_o$$

and then

$$x_1 = x_o - A^T f_o$$

$$f_1 = f_o - A A^T f_o = (I - A A^T) f_o$$

and then unless $f_o = 0$ or $A^{-1} = A^T$, $f_1 \neq 0$ (while if we take $\alpha_k = 1$ and $\beta_k = 0$ then it is easy to see that $f_1 = 0$).

This method is also dependent on the scaling being used and in the code tested by Hiebert an internal scaling is provided which results in a scale invariant code but this option causes the performance of the code to deteriorate instead of improving it (this was found by HIEBERT (1980) and confirmed by CHEN and STADTHERR (1981)).

A summary of results is presented in Table 1.1.

Table 1.1: Summary of results obtained by Hiebert (1980)

		Percentage of problems solved		
		Math problems		Chemical Engineering Problems
Method	Implementation	Well scaled	Poorly scaled	
BRENT (1973)	MINPACK (1980)	74	49	83
BROWN (1966)	IMSL (1979)	67	50	42
BROYDEN (1965)	SANDIA (1975)	42	29	42
POWELL (1970)	MINPACK (1980)	86	57	50

The main conclusion from the Hiebert study is that while there are very robust codes for well scaled problems, the performance for all codes is badly affected by poor scaling. This result is of interest to us since a flowsheeting problem involves quantities which differ by orders of magnitude (as do for instance, molar fractions and flowrates).

From the analysis of the three families we conclude that the most attractive for solving flowsheeting problems is family Q. It has good convergence rate, it is the least expensive in function evaluations and operations count (the difference with other families becoming dramatically different when n becomes large and/or f is expensive to evaluate), it requires reasonable storage and its only apparent disadvantages are the numerical results obtained by Hiebert for Broyden's method and its dependence on the scaling.

We will show in chapter 2 that the first disadvantage is due to a poor implementation of the method. We will propose some implementation details which will improve considerably the performance of Broyden's method as well as a modification to Bennett's algorithm for obtaining LU factors which avoids numerically singular approximations B_{k+1}. Numerical results will be presented showing that our implementation of Broyden's method has better overall performance than the best method tested by Hiebert (ie the hybrid method).

Regarding the second disadvantage, the dependence on the scaling, we will propose in chapter 3 a sub-family of family Q which is scale invariant. We will show that its convergence is superlinear and numerical examples will show that the methods we propose have equivalent behaviour to Broyden's method.

The numerical results in chapter 3 will show that implementations of theoretically scale invariant algorithms are not scale invariant. This is due to the numerical conditioning of the problem.

The numerical conditioning of family Q will be studied in chapter 4 and ways of improving it will be proposed. In particular an internal scaling procedure will be proposed and numerical results will show how the performance of the methods is improved with it.

Finally, in chapter 5, we will apply our best methods to flowsheeting problems using the simulation package SPEED-UP. The chemical engineering problems will show the behaviour of our code in coping with small but difficult to solve problems as well as large problems when physical properties are used (when the computer time becomes critical).

CHAPTER 2

THE IMPLEMENTATION OF QUASI-NEWTON METHODS

We will discuss in this chapter the development of an implementation of Quasi-Newton methods which, for Broyden's method , gives numerical results substantially better than those published so far.

We will first introduce the family of rank-one Quasi-Newton methods and an algorithm implementing it. Implementation details for this algorithm will then be discussed. They consist of how to choose the initial approximation B_0 to the Jacobian and the convenience of its re-initialisation, the choosing of an LU factorization, how to find a suitable policy for the step reduction factor and how it is possible to avoid a numerically singular approximation B_k by using a modification of Bennett's algorithm for the updating of LU factors.

The chapter will end with the presentation of numerical results obtained for Broyden's method using the implementation discussed here.

2.1 RANK-ONE QUASI-NEWTON METHODS

Different methods have been proposed in the past based on how B_k in 1.25 is obtained. BROYDEN (1965) proposed a whole family of methods based on the following updating formula.

$$B_{k+1} = B_k + a_k\, b_k^T \qquad 2.1$$

The two vectors a_k and b_k were chosen according to the following relation (secant relation)

$$B_{k+1}\, p_k = f(x_{k+1}) - f(x_k) \qquad 2.2$$

which arises from the fact that if f is a linear function then its Jacobian $F'(x_{k+1})$ will satisfy

$$F'(x_{k+1})\, p_k = f(x_{k+1}) - f(x_k) \qquad 2.3$$

Using the notation

$$f_k = f(x_k) \qquad 2.4$$

and by defining

$$y_k = f_{k+1} - f_k \qquad 2.5$$

we can see that in order to satisfy 2.2, a_k and b_k must be such that

$$a_k = y_k - B_k\, p_k \qquad 2.6$$

$$b_k^T\, p_k = 1 \qquad 2.7$$

It is then possible to characterize the family of rank-one Quasi-Newton methods with a sequence $\{v_k\}$ of vectors in $\mathbb{R}^n$ using the update formula

$$B_{k+1} = B_k + (y_k - B_k\, p_k)\frac{v_k^T}{v_k^T\, p_k} \qquad 2.8$$

In particular BROYDEN's(1965) method is a member of this family, in which $v_k = p_k$ .More methods have been proposed in the past choosing different v_k.BARNES(1965) proposed to choose v_k orthogonal to the previous steps p_k.This assures that linear systems are solved in one iteration. This method was implemented and tested by GAY and SCHNABEL(1978) and by PALOSCHI(1980).

It has been proposed originally by BROYDEN (1965) that instead of using 1.26, the following equation should be used

$$x_{k+1} = x_k + \lambda_k\, p_k \qquad 2.9$$

where the parameter λ_k is chosen according to a particular policy. Broyden proposed to choose λ_k such that

$$\|f_{k+1}\| < \|f_k\| \qquad 2.10$$

We can formalize this family of methods with the following algorithm:

ALGORITHM 1: Rank-one Quasi-Newton methods.

1 Given x_o, B_o, ε

2 Set k = o

3 If $\|f(x_k)\| < \varepsilon$ then stop

4 Choose a suitable value for λ_k

5 $p_k = -\lambda_k B_k^{-1} f(x_k)$

6 $x_{k+1} = x_k + p_k$

7 $y_k = f(x_{k+1}) - f(x_k)$

8 $B_{k+1} = B_k + (y_k - B_k p_k)\frac{v_k^T}{v_k^T p_k}$

(v_k is determined by the particular method)

9 k = k + 1

10 Go to 3

2.2 IMPLEMENTATION OF ALGORITHM 1

2.2.1 The initial approximation to the jacobian

Two approaches have been proposed in the past for the selection of the initial approximation to the Jacobian, ie the matrix B_o in step 1 of algorithm 1. ROSEN (1966) proposed the use of $B_o = I$ and BROYDEN (1965) suggested the use of a finite difference approximation to $F'(x_o)$ obtaining B_o as

$$B_o e_j = \frac{f(x_o + \delta_j e_j) - f_o}{\delta_j} \qquad 2.11$$

with

$$\delta_j = 0.01 \max \{|e_j^T x_o|, \quad 10^{-6} \}$$

We have chosen to implement Broyden's suggestion. This alternative has the disadvantage of requiring n additional function evaluations but is has been found to perform better than $B_o = I$ in the past by METCALFE and PERKINS (1978) for flowsheeting problems and by PALOSCHI (1979) for general non-linear equations. BOGLE (1979) found the use of $B_o = I$ to perform better, for some flowsheeting problems, regarding efficiency but 2.11 was found better for robustness. An important property of B_o obtained using 2.11 is that it will satisfy

$$\hat{B}_o = D_f B_o D_x^{-1} \qquad 2.12$$

for a change of scale of the form 1.29 provided x_o is such that all its components are not zero. This property is required to ensure scale invariance as will be discussed in the next chapter.Another important property obtained using 2.11 is that if the problem 1.18 involves any linear subsets then all linear equations belonging to that subset will be satisfied by the sequence x_k for $k > 0$(PERKINS(1979a)). While it is not necessary to have approximations B_k being close to $F'(x_k)$ to ob - tain convergence in practice (DENNIS and MORE(1977)),theoretically su perlinear convergence is guaranteed only if B_o is sufficiently close to $F'(x_*)$,where x_* is the solution to 1.18. This suggests that if the algorithm is not making any improvement (ie,the norm of the function is not being reduced) the behaviour could be improved by having B_k clo- ser to $F'(x_*)$. For this ,a re-initialisation of B_k by finite differen- ces could be made using some x_k being closer to x_* than was x_o.This has been used in the past by CHEN and STADTHERR(1981) using the following procedure.

Calculate a new jacobian if both of the following conditions hold:

(a) The norm of the function has been reduced by a factor of 2 since last jacobian evaluation

(b) $$\frac{\|f_{k+1}\|^2}{\|f_{k-4}\|} > \frac{\|f_{k-4}\|}{\|f_{k-9}\|}$$

We found the re-initialisation very useful but using instead the following procedure:

Calculate a new jacobian using the best point x_k available (ie, the one with minimum norm for f_k), if both the following conditions hold:

(a) After 10 + n consecutive iterations, the norm of the function has not been reduced at least by a factor of 0.9

(b) Since the last re-initialisation, the norm of the function has, at least once, been reduced by a factor of 0.9.

This re-initialisation is particularly suitable when solving flow-sheeting problems because if the function is sufficiently sparse it is possible in 2.11 to perturb more than one variable simultaneously and thus reducing considerably the number of function evaluations needed to obtain B_0 (see CURTIS, POWELL and REID (1974)).

2.2.2 Evaluation of the step p_k

Step 5 of algorithm 1 involves the solution of a linear system for finding the step p_k as

$$B_k p_k = -\lambda_k f(x_k) \qquad 2.13$$

For this we have three alternatives:

(a) invert B_k

(b) have available B_k^{-1}

(c) have available a factorisation of B_k.

The first alternative is not practical. BROYDEN (1965) suggested the use of alternative b. For this the use of the SHERMAN -MORRISON (1949) formula was proposed for obtaining instead of 2.8

$$H_{k+1} = H_k + \frac{(p_k - H_k y_k)\, v_k^T H_k}{v_k^T H_k y_k} \qquad 2.14$$

where

$$H_k = B_k^{-1} \qquad 2.15$$

There are numerical problems related to the use of 2.14 ie, having small denominators or obtaining numerically singular approximations H_{k+1}. In addition, preliminary tests for this approach did not show promising results, and therefore it was abandoned. It is possible to have alternative (c) using operations of the same order as alternative (b). GILL and MURRAY (1972) describe a method for having a factorisation

$$B_k = Q_k R_k \qquad 2.16$$

where Q_k is orthogonal and R_k is upper triangular. This approach has been used by MORE, GARBOW and HILLSTROM (1980). While this is a very safe procedure numerically it has the disadvantage of using $\frac{3}{2} n^2$ storage locations (opposed to n^2 necessary for alternative (b)).

It is possible instead to use an LU factorisation

$$B_k = L_k U_k \qquad 2.17$$

by using the updating algorithm due to BENNETT (1965). This requires $O(n^2)$ operations (same as alternative (b)) and also n^2 storage locations. It has been used by CHEN and STADTHERR (1981).

We have selected alternative (c) for our implementation.

In section 2.3 we will present a modification to Bennett's algorithm which will allow us to avoid numerically singular approximations B_{k+1}.

2.2.3 Choosing a suitable λ_k

Step 4 of algorithm 1 has been proposed in the past for improving the performance of the methods.

It has been shown that provided $\{\lambda_k\}$ converges to 1 then the superlinear convergence properties of the method remain unmodified (see DENNIS and MORE (1974)).

BROYDEN (1965) proposed to choose λ_k such that

$$\|f_{k+1}\| < \|f_k\| \qquad 2.18$$

METCALFE and PERKINS (1978), when implementing Broyden's method using 2.14, found that if

$$\|f_{k+1}\| \gg \|f_k\|$$

then the approximation H_{k+1} becomes numerically singular. They then suggested the use of λ_k to ensure that

$$\|f_{k+1}\| \leq 10\|f_k\| \qquad 2.19$$

Numerical results have shown that 2.19 is better than 2.18 (see BOGLE (1979)). MORE and COSNARD (1979) suggested using λ_k in order to keep a control on $\|p_k\|$. They proposed to define

$$\delta_0 = \max (10, 10\ \|x_0\|)$$
$$\delta_{k+1} = \max (\delta_k, 10\ \|x_k\|),\ k > 1 \qquad 2.20$$

and then to use λ_k to ensure that

$$\|p_k\| \leq \delta_k \qquad 2.21$$

Numerical results for various methods confirmed the value of this rule (PALOSCHI (1980)). We have used for our implementation modified versions of 2.19 and 2.21.

Define

$$x_k^T = (\xi_{1k}, \xi_{2k}, \ldots, \xi_{nk})$$

$$p_k^T = (\pi_{1k}, \pi_{2k}, \ldots, \pi_{nk})$$

$$\delta_{ik} = \begin{cases} 50 & \text{if } \xi_{ik} = 0 \\ 50\ |\xi_{ik}| & \text{if } \xi_{ik} \neq 0 \end{cases} \qquad 2.22$$

for $i = 1, 2, \ldots, n$ and $k \geq 0$.

Our selection of λ_k is made to assure simultaneously

$$|\pi_{ik}| \leq \delta_{ik}, \qquad i = 1, 2, \ldots, n \qquad 2.23$$

$$\|f_{k+1}\| \leq 100\|f_0\| \qquad 2.24$$

The reason for using 2.23 instead of 2.21 is that this choice is scale invariant for changes of the form 1.29 (when $\xi_{ik} \neq 0$).

Attempts were made to obtain, instead of 2.24, a control being scale invariant but none was found to work as well as 2.24. We should note that 2.24 will make the code fail if x_0 is inside a region containing a non-zero local minimum of $\|f\|$.

2.2.4 Bounds on the variables

A facility has been provided in our code for allowing the variables to be bounded.

The iterates generated by the code satisfy the following relation.

$$\alpha_i \leq \xi_i \leq \beta_i, \quad i = 1, 2, \ldots, n$$

To ensure this, λ_k in 2.9 is used if possible. The reason for using λ_k is because we want to keep the direction of the Quasi-Newton step. In doing this we ensure that if x_k satisfies a linear subsystem then, since x_{k+1} (taken with the full Quasi-Newton step) also will satisfy the linear subsystem, any x'_{k+1} taken with $\lambda_k \neq 1$ will also satisfy the linear subsystem.

If the point x_k is already on the bound and x_{k+1} is predicted outside, no reduction of the step is possible and then we have no choice but to abandon the Quasi-Newton direction. In this case we project the step onto the boundary (see figs 2.1 and 2.2).

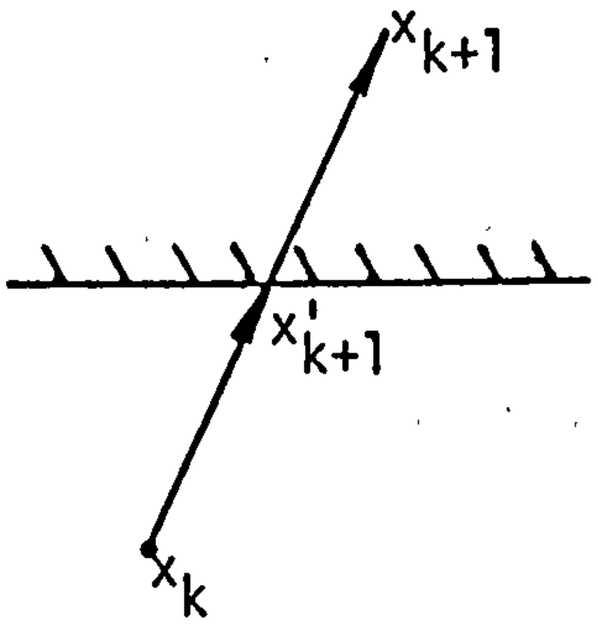


fig 2.1: Using λ_k to keep the Q-N direction

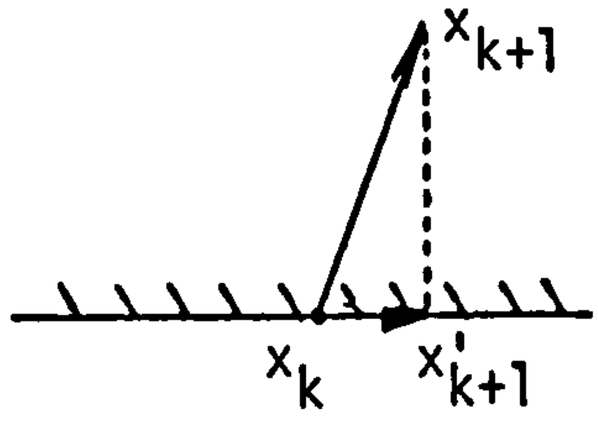


fig 2.2: Projecting the step on the bound

2.3 AVOIDING NUMERICALLY SINGULAR APPROXIMATIONS TO THE JACOBIAN

We are interested in the LU factorization of B_i, that is, having

$$B_i = L_i U_i \qquad 2.25$$

where L_i is a lower triangular matrix while U_i is a unitary upper triangular matrix. Two matrices L_{i+1} and U_{i+1} are obtained such that

$$B_{i+1} = L_{i+1} U_{i+1} \qquad 2.26$$

An algorithm for obtaining L_{i+1} and U_{i+1} from 2.8 has been presented by BENNETT (1965).

It may happen that B_{i+1}, as obtained using 2.8 is singular or its LU factorization has very small elements in the diagonal of the matrix L. In both cases the solution of 2.13 will be very difficult numerically or even impossible.

We propose here a modification of Bennett's algorithm which produces an updated factorization of the Jacobian approximation which satisfies the secant relation and is non-singular. In cases where the update 2.8 and the Bennett algorithm produce a non-singular matrix, our algorithm gives the same matrix. If 2.8 or the original Bennett algorithm gives a singular matrix, our algorithm gives a matrix which is in some sense close to that matrix, but which is non-singular.

In section 2.3.2 we describe very briefly Bennett's algorithm. The modification is proposed in section 2.3.3 with its theoretical justification. The implementation details are given in section 2.3.4 where we show how to choose between the different alternatives. An example of use of the modification is presented in section 2.3.5.

2.3.1 Bennett's algorithm

Suppose we have an LU factorization of a matrix A

$$A = LU \qquad 2.27$$

and a matrix B is obtained using the updating formula

$$B = A + ab^T \qquad 2.28$$

where a and b are two given vectors.

If we are going to obtain the LU factors of B using a standard technique, without using the fact that B is obtained using 2.8, we need $O(n^3)$ operations. BENNETT (1965) proposed an algorithm to obtain the LU factors of B satisfying 2.8 using just $O(n^2)$ operations.

We will very briefly describe the algorithm.

Define L_i as the unit matrix with the ith column replaced by the ith column of the matrix L and U_i in the same way, using U instead of L and interchanging the role of rows and columns.

These matrices will satisfy

$$L = \prod_{i=1}^{n} L_i \qquad 2.29$$

$$U = \prod_{i=n}^{1} U_i \qquad 2.30$$

Define $A_1 = A$, $B_1 = B$, $a_1 = a$, $b_1 = b$ and for each k, $1 \leq k < n$, define A_{k+1} such that

$$A = \prod_{i=1}^{k} L_i \begin{pmatrix} I_k & 0 \\ 0 & A_{k+1} \end{pmatrix} \prod_{i=k}^{1} U_i \qquad 2.31$$

A_{k+1} is of order $(n-k)$.

Then, at each stage k of the algorithm we will have obtained B_k such that

$$B_k = A_k + a_k b_k^T \qquad 2.32$$

and then L_{k*}, U_{k*}, a_{k+1} and b_{k+1} will be obtained such that

$$B = \prod_{i=1}^{k} L_{i*} \begin{pmatrix} I_k & 0 \\ 0 & B_{k+1} \end{pmatrix} \prod_{i=k}^{1} U_{i*} \qquad 2.33$$

and

$$B_{k+1} = A_{k+1} + a_{k+1}\, b_{k+1}^T \qquad 2.34$$

B_{k+1} is of order $(n-k)$.

When stage $n-1$ is finished, it is very easy to obtain L_{n*} and $U_{n*} = I$. The factorization of B will then be

$$B = \prod_{i=1}^{n} L_{i*} \prod_{i=n}^{1} U_{i*} \qquad 2.35$$

2.3.2 Proposed modification

In the special case of Quasi-Newton methods, the vector b in 2.28 is varied to give different methods. In fact, varying v_k in 2.8, we can obtain a family of rank-one updates for Quasi-Newton methods. In general, all Quasi-Newton methods satisfy the following secant relation

$$Bp = y \qquad 2.36$$

where p and y are two given vectors.

Suppose we have obtained B for a particular method and it is singular. Since B will later be used to solve a linear system, this particular method will clearly fail in this case. When this problem arises, we propose to modify B to B' such that

$$B'p = y \qquad 2.37$$

We will require B' to be non-singular and also not too different from B.

To achieve this, we propose a modification to Bennett's algorithm, to be added at stage k, the stage when singularity is detected.

The next theorem will show the only case when our proposed modification will not be applicable.

Theorem 2.1: If p and y are such that

$$p = \begin{pmatrix} p_k \\ 0 \end{pmatrix} \qquad 2.38$$

$$y = \begin{pmatrix} 0 \\ y_k \end{pmatrix} \qquad 2.39$$

with p_k being of order k and y_k of order n - k and $p \neq 0$, $y \neq 0$.

Then it is not possible to find L and U, L non-singular, such that

$$L\,U\,p = y \qquad 2.40$$

Proof: assume there exist L non-singular and U satisfying 2.40.

Define the following partitions

$$L = \begin{pmatrix} L_{11} & 0 \\ C_{21} & L_{22} \end{pmatrix}$$

$$U = \begin{pmatrix} U_{11} & C_{12} \\ 0 & U_{22} \end{pmatrix}$$

where L_{11}, U_{11} are of order k, L_{22} and U_{22} of order n - k, C_{21} is (n - k) x k and C_{12} is k x (n - k).

From 2.40 it follows

$$L_{11}U_{11}p_k = 0$$

$$C_{21}U_{11}p_k = y_k$$

Since L_{11} is non-singular then $U_{11}p_k = 0$ and then $y_k = 0$ which contradicts our hypothesis $y \neq 0$.

The following theorem shows that it is possible to modify the algorithm to achieve what we want and also how to.

Theorem 2.2: If at stage k of Bennett's algorithm we have

$$\prod_{i=1}^{k-1} L_{i*} \text{ is non-singular for } k > 1$$

2.38 and 2.39 are not true simultaneously and

$$|e_1^T B_k e_1| = 0 \qquad 2.41$$

$$|e_1^T A_k e_1| \geq \varepsilon \qquad 2.42$$

Then there exist L'_{i*}, $i = 1, 2, \ldots, k-1$ for $k > 1$ and B'_k such that if

$$B' = \prod_{i=1}^{k-1} L'_{i*} \begin{pmatrix} I_{k-1} & 0 \\ 0 & B'_k \end{pmatrix} \prod_{i=k-1}^{1} U_{i*} \quad \text{if} \quad k > 1 \qquad 2.43$$

or

$$B' = B'_1 \quad \text{if} \quad k = 1 \qquad 2.44$$

then

$$B'p = y \qquad 2.45$$

and

$$\prod_{i=1}^{k-1} L'_{i*} \text{ is non-singular for } k > 1$$

and

$$|e_1^T B'_k e_1| = \varepsilon \qquad 2.46$$

Proof: Case (a). Assume first 2.38 is not true. We will show that it is possible to find b'_k such that

$$e_1^T (A_k + a_k b_k'^T) e_1 = \varepsilon \qquad 2.47$$

and

$$b_k'^T p_{k2} = b_k^T p_{k2} \qquad 2.48$$

where p_{k2} is of order n-k+1 and such that

$$p = \begin{pmatrix} p_{k1} \\ \\ p_{k2} \end{pmatrix} \qquad \text{if } k > 1$$

$$p = p_{k2} \qquad \text{if } k = 1 \qquad 2.49$$

If we use the following notation

$$a_k = \begin{pmatrix} \alpha_k \\ \\ \overline{a}_k \end{pmatrix}$$

$$b_k = \begin{pmatrix} \beta_k \\ \\ \overline{b}_k \end{pmatrix}$$

2.46 is satisfied if we take

$$\beta'_k = (\varepsilon - e_1^T A_k e_1)/\alpha_k \qquad 2.50$$

and $\alpha_k \neq 0$ because of 2.41, 2.42 and 2.32.

Since 2.38 is not true, it is possible to find a $\overline{b}'_k$ so that, with β'_k from 2.50, a b'_k satisfying 2.48 can be found.

If we now take $B'_k = A_k + a_k b_k'^T$ and $L'_{i*} = L_{i*}$, i = 1, 2, ..., k-1 (if k > 1) it is clear that 2.46 is satisfied.

It remains to show that 2.45 is satisfied and for this define the following partitions if k > 1

$$\prod_{i=1}^{k-1} L_{i*} = \begin{pmatrix} L_{k1} & 0 \\ C_{k1} & I_{n-k-1} \end{pmatrix} \qquad 2.51$$

$$\prod_{i=k-1}^{1} U_{i*} = \begin{pmatrix} U_{k1} & D_{k1} \\ 0 & I_{n-k+1} \end{pmatrix} \qquad 2.52$$

$$y = \begin{pmatrix} y_{k1} \\ y_{k2} \end{pmatrix} \qquad 2.53$$

where y_{k1}, L_{k1}, U_{k1} are of order k-1 and C_{k1} is (n-k+1) x (k-1), D_{k1} is of order (k-1) x (n-k+1). Since Bp = y it follows

$$L_{k1} (U_{k1} p_{k1} + D_{k1} p_{k2}) = y_{k1} \qquad 2.54$$

$$C_{k1} (U_{k1} p_{k1} + D_{k1} p_{k2}) + B_k p_{k2} = y_{k2} \qquad 2.55$$

Since 2.48 is true and

$$B'_k = A_k + a_k b'^{T}_k \qquad 2.56$$

it follows

$$B'_k p_{k2} = B_k p_{k2}$$

And then 2.45 is satisfied.

The case k=1 is straightforward.

Case (b). Assume now that 2.38 is true. Take β'_k as defined by 2.50 and $\bar{b}'_k = \bar{b}_k$. We will show that $y_{k1} \neq 0$ for k > 1.

Assume $y_{k1} = 0$, from 2.54 it follows

$$(U_{k1} p_{k1} + D_{k1} p_{k2}) = 0 \qquad 2.57$$

because L_{k1} is non-singular and then from 2.55 it follows

$$B_k p_{k2} = y_{k2} \qquad 2.58$$

and since 2.38 is true, using 2.41 we can obtain

$$e_1^T y_{k2} = 0 \qquad 2.59$$

but this implies with $y_{k1} = 0$ that 2.39 is true and this is a contradiction.

It then follows

$$(U_{k1}\, p_{k1} + D_{k1} p_{k2}) \neq 0 \qquad 2.60$$

and then it is possible to find C'_{k1} such that

$$C'_{k1}(U_{k1}\, p_{k1} + D_{k1}\, p_{k2}) + B'_k\, p_{k2} = y_{k2} \qquad 2.61$$

giving then that 2.45 is satisfied and also 2.46.

The same reasoning is valid for k = 1.

2.3.3 Implementation

We see from the last section that it is possible to update LU factors achieving two of our objectives, viz, to avoid singular Jacobian approximations while satisfying the secant relation. The necessary relations which the updating algorithm must satisfy are given in Theorem 2.2. In particular, they are equations (2.48) and (2.50) in Case (a) and equations (2.50), (2.56) and (2.61) in Case (b). A study of these conditions reveals that they define a whole family of algorithms. In this section, we make suggestions of reasonable choices from the family. These choices minimize in some sense the correction required to the original Quasi-Newton update to remove singularity.

Using partitions defined by 2.51 and 2.52 it follows

$$B = \begin{pmatrix} L_{k1} & 0 \\ C_{k1} & I_{n-k+1} \end{pmatrix} \begin{pmatrix} I_{k-1} & 0 \\ 0 & B_k \end{pmatrix} \begin{pmatrix} U_{k1} & D_{k1} \\ 0 & I_{n-k+1} \end{pmatrix} \qquad 2.62$$

or

$$B = \begin{pmatrix} L_{k1}U_{k1} & L_{k1}D_{k1} \\ C_{k1}U_{k1} & C_{k1}D_{k1} + B_k \end{pmatrix} \qquad 2.63$$

If we use Case (a) of theorem 2.2 then

$$B' = \begin{pmatrix} L_{k1}U_{k1} & L_{k1}D_{k1} \\ C_{k1}U_{k1} & C_{k1}D_{k1} + B'_k \end{pmatrix} \qquad 2.64$$

and if Case (b) is used then

$$B' = \begin{pmatrix} L_{k1}U_{k1} & L_{k1}D_{k1} \\ C'_{k1}U_{k1} & C_{k1}D_{k1} + B'_k \end{pmatrix} \qquad 2.65$$

Assume that B' is defined by 2.64, it follows using 2.63

$$\|B' - B\| = \|B'_k - B\| \qquad 2.66$$

and from 2.32 and 2.56

$$\|B' - B\| = \|a_k\| \ \|b'_k - b_k\| \qquad 2.67$$

The first component of the vector b'_k is uniquely defined by 2.50 but there is some freedom for the choice of $\bar{b}'_k$ (the remaining components).

We will make use of the following Lemma due to DENNIS and SCHNABEL (1978).

Lemma 2.1: Let $\alpha \epsilon \mathbb{R}$, $v \epsilon \mathbb{R}^n$, $v \neq 0$. Then the unique solution to

$$\min_{x \epsilon \mathbb{R}^n} \|x\| \quad \text{subject to } v^T x = \alpha$$

is

$$x = \frac{\alpha v}{v^T v} \qquad 2.68$$

Since b'_k has to satisfy 2.48 in addition to 2.50 it follows that by applying Lemma 2.1 with $x = \bar{b}'_k - \bar{b}_k$, $v = \bar{p}_{k2}$ and $\alpha = (\beta_k - \beta'_k)e_1^T p_{k2}$ we can obtain B' such that $\|B' - B\|$ is a minimum. Thus, the following equation defines $\bar{b}'_k$.

$$\bar{b}'_k = \bar{b}_k + \frac{(\beta_k - \beta'_k)(e_1^T p_{k2})\bar{p}_{k2}}{\bar{p}_{k2}^T \bar{p}_{k2}} \qquad 2.69$$

Assume now that B' is given by 2.65. In this case k > 1 and

$$\|B' - B\|^2 = \|(C'_{k1} - C_{k1})U_{k1}\|^2 + \|(C'_{k1} - C_{k1})D_{k1} + (B'_k - B_k)\|^2 \qquad 2.70$$

where C'_{k1} must satisfy 2.61 and B'_k is defined by 2.56 with $\bar{b}'_k = \bar{b}_k$ and β'_k given by 2.50. We can see that B'_k is uniquely defined but there are some degrees of freedom in choosing C'_{k1} satisfying 2.51.

If we define

$$X = C'_{k1} - C_{k1} \qquad 2.71$$

using ξ, μ, δ and β for referring to the elements of X, U, D and B and dropping the subscripts we can see that

$$\|X\ U\|^2 = \sum_{i=1}^{n-k+1} \sum_{j=1}^{k-1} \left(\sum_{\ell=1}^{k-1} \xi_{i\ell}\, \mu_{\ell j} \right)^2 \qquad 2.72$$

$$\|X\ D + (B' - B)\|^2 = \sum_{i=1}^{n-k+1} \sum_{j=1}^{n-k+1} \left(\sum_{\ell=1}^{k-1} \xi_{i\ell}\delta_{\ell j} + (\beta'_{ij} - \beta_{ij}) \right)^2 \qquad 2.73$$

Since C_{k1} satisfies 2.55 and C'_{k1} must satisfy 2.61 it follows that C'_{k1} must satisfy

$$(C'_{k1} - C_{k1})(U_{k1}\, p_{k1} + D_{k1}\, p_{k2}) + (B'_k - B_k)p_{k2} = 0 \qquad 2.74$$

Dropping the subscript k and defining

$$s = U_{k1}p_{k1} + D_{k1}p_{k2} \qquad 2.75$$

2.74 becomes

$$X\,s = (B_k - B_k')p_{k2} \qquad 2.76$$

In order to minimize 2.70 subject to 2.76 we define the Lagrangian

$$g(X) = \|B' - B\|^2 + \sum_{i=1}^{n-k+1} \lambda_i e_i^T (Xs - (B_k - B_k')p_{k2}) \qquad 2.77$$

where we have introduced Lagrange multipliers λ_i for the constraints 2.76.

We will now obtain the partial derivatives of 2.77.

$$\frac{\partial}{\partial \xi_{rs}} \|XU\|^2 = \sum_{j=1}^{k-1} 2 \left(\sum_{\ell=1}^{k-1} \xi_{r\ell}\, \mu_{\ell j} \right) \mu_{sj} \qquad 2.78$$

$$\frac{\partial}{\partial \xi_{rs}} \|XD + (B' - B)\|^2 = \sum_{j=1}^{n-k+1} 2 \left(\sum_{\ell=1}^{k-1} \xi_{r\ell}\, \delta_{\ell j} + (\beta'_{rj} - \beta_{rj}) \right) \delta_{sj} \qquad 2.79$$

$$\frac{\partial}{\partial \xi_{rs}} \|B' - B\|^2 = \sum_{\ell=1}^{k-1} 2\, \xi_{r\ell} \left(\sum_{j=1}^{k-1} \mu_{\ell j}\mu_{sj} + \sum_{j=1}^{n-k+1} \delta_{\ell j}\delta_{sj} \right)$$

$$+ \sum_{j=1}^{n-k+1} 2\, (\beta'_{rj} - \beta_{rj})\delta_{sj} \qquad 2.80$$

If we define

$$h(X) = \sum_{i=1}^{n-k+1} \lambda_i\, e_i^T (Xs - (B_k - B_k')p_{k2}) \qquad 2.81$$

and by naming the elements of p_{k2} as π_j, $i \le j \le n-k+1$

$$h(X) = \sum_{i=1}^{n-k+1} \lambda_i \left(\sum_{j=1}^{k-1} \xi_{ij}\, r_j + \sum_{j=1}^{n-k+1} (\beta'_{ij} - \beta_{ij})\pi_j \right) \qquad 2.82$$

where r_j are the elements of the vector s.

It follows

$$\frac{\partial}{\partial \xi_{rs}} h(X) = \lambda_r r_s \qquad 2.83$$

$$\frac{\partial}{\partial \lambda_r} h(X) = \sum_{j=1}^{k-1} \xi_{rj}\Gamma_j + \sum_{j=1}^{n-k+1} (\beta'_{rj} - \beta_{rj})\pi_j \qquad 2.84$$

and then

$$\frac{\partial g(x)}{\partial \xi_{rs}} = \sum_{\ell=1}^{k-1} 2\xi_{r\ell}\left(\sum_{j=1}^{k-1} \mu_{\ell j}\mu_{sj} + \sum_{j=1}^{n-k+1} \delta_{\ell j}\delta_{sj}\right)$$

$$+ \sum_{j=1}^{n-k+1} 2(\beta'_{rj} - \beta_{rj})\delta_{sj} + \lambda_r\Gamma_s \qquad 2.85$$

$$\frac{\partial g(x)}{\partial \lambda_r} = \sum_{j=1}^{k-1} \xi_{rj}\Gamma_j + \sum_{j=1}^{n-k+1} (\beta'_{rj} - \beta_{rj})\pi_j \qquad 2.86$$

In order to minimize 2.70 we should solve 2.85 and 2.86 equated to zero for all possible ξ_{rs} and λ_r. This requires solving n-k+1 linear systems of order k-1 and is not practical. In order to simplify the problem, we restrict C'_{k1} such that it only differs from C_{k1} in the p-column. In this case C'_{k1} becomes uniquely defined from 2.61.

$$\xi_{ip} = (\beta_{i1} - \beta'_{i1})\frac{\pi_1}{\Gamma_p} \qquad i = 1, 2, \ldots, n-k+1 \qquad 2.87$$

We still have some freedom in the choice of p such that $\Gamma_p \neq 0$ and we will use it for minimizing 2.70.

With our choice of C'_{k1} 2.70 becomes

$$\|B' - B\|^2 = \sum_{i=1}^{n-k+1} \left(\sum_{j=1}^{k-1} \xi_{ip}^2\mu_{pj}^2 + \sum_{j=1}^{n-k+1} (\xi_{ip}\delta_{pj} + \beta'_{ij} - \beta_{ij})^2\right) \qquad 2.88$$

since $\beta'_{ij} = \beta_{ij}$ if $j > 1$ and by defining

$$\mu_p = \sum_{j=1}^{k-1} \mu_{pj}^2 \qquad 2.89$$

$$\delta_p = \sum_{j=2}^{n-k+1} \delta_{pj}^2 \qquad 2.90$$

$$\|B' - B\|^2 = \sum_{i=1}^{n-k+1} \xi_{ip}^2 \mu_p + (\xi_{ip}\delta_{p1} + \beta'_{i1} - \beta_{i1})^2 + \xi_{ip}^2 \delta_p \qquad 2.91$$

and 2.87 gives

$$\|B' - B\|^2 = \sum_{i=1}^{n-k+1} (\beta_{i1} - \beta'_{i1})^2 \left[\frac{\pi_1^2 \mu_p}{\Gamma_p^2} + \left(\frac{\pi_1 \delta_{p1}}{\Gamma_p} - 1 \right)^2 + \frac{\pi_1^2}{\Gamma_p^2} \delta_p \right] \qquad 2.92$$

and then the problem is reduced to finding

$$\min \left[\frac{\pi_1^2}{\Gamma_p^2} (\mu_p + \delta_p + \delta_{p1}^2) - \frac{2\pi_1 \delta_{p1}}{\Gamma_p} + 1 \right]; \quad \Gamma_p \neq 0, \quad 1 \leq p < k \qquad 2.93$$

The use of equation 2.69 in obtaining $\bar{b}'_k$ is not safe numerically because $(\beta_k - \beta'_k)$ may be very small. We propose to use instead

$$\bar{b}'_k = \bar{b}_k + \frac{(\beta_{11} - \varepsilon)(e_1^T p_{k2})\bar{p}_{k2}}{\alpha_k \bar{p}_{k2}^T \bar{p}_{k2}} \qquad 2.94$$

which can also be obtained from 2.32 and 2.50.

The same is valid also for equation 2.87 and we will use instead

$$\xi_{ip} = \frac{(\beta_{11} - \varepsilon)}{\alpha_k} (e_i^T \bar{a}_k) \frac{\pi_1}{\Gamma_p} \qquad 2.95$$

2.3.4 Example of use

We will show with one example how our proposed modification improves the performance of Quasi-Newton methods. Consider Broyden's (1965) good method ($v_i = p_i$ in 2.8).

We will take the function

$$f(x) = x \qquad 2.96$$

As the initial point we take

$$x_o = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \qquad 2.97$$

and as the initial approximation to the Jacobian

$$B_o = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \qquad 2.98$$

B_o is easily factorized as

$$B_o = L_o U_o = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad 2.99$$

Using Broyden's method we will obtain

$$p_o = -B_o^{-1} f_o = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \qquad 2.100$$

and then

$$x_i = x_o + p_o = \begin{pmatrix} 2 \\ 0 \end{pmatrix} \qquad 2.101$$

From 2.8 it follows

$$B_1 = B_o + \frac{f_1 p_o^T}{p_o^T p_o} = \begin{pmatrix} 0 & -1 \\ 0 & 1 \end{pmatrix} \qquad 2.102$$

And then B_1 is singular and Broyden's method will have to be aborted since we will not be able to solve 2.96 for $i = 1$.

If instead of the original Bennett's algorithm, which will give us B_1 as in 2.68, we apply our modification (Case a) of Lemma 2.2 for any $\varepsilon > 0$, we will obtain

$$B_1' = \begin{pmatrix} \varepsilon & \varepsilon-1 \\ 0 & 1 \end{pmatrix} \qquad 2.103$$

and then

$$x_2^T = \begin{pmatrix} 2 - \frac{2}{\varepsilon} & 0 \end{pmatrix}$$

$$B_2 = \begin{pmatrix} 1 & \varepsilon-1 \\ 0 & 1 \end{pmatrix}$$

$$x_3 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

and then x_3 is the solution, obtained with any $\varepsilon > 0$.

2.4 NUMERICAL RESULTS

We will present in this section numerical results obtained with the implementation discussed in this chapter of Broyden's method (ie, $v_k = p_k$ in 2.8). We will use the same set of examples as the report by HIEBERT (1980). There are basically two sets of examples:a "general set" of mathematical examples and a "chemical equilibrium" set; they are described in Appendix A.

As was done by HIEBERT (1980), we will use a diagonal matrix S_{mn} for testing the behaviour of the code under different scaling conditions.

The diagonal elements of the matrix S_{mn} are defined by

$$\log_{10} \sigma_{mi} = (m((2i - n - 1)/(n - 1))) \qquad 2.104$$

for i = 1, 2, ..., n.

For creating a set with variables badly scaled we use

$$\hat{f}(\hat{x}) = f(S_{5n}x) \qquad 2.105$$

and for functions badly scaled

$$\hat{f}(x) = S_{5n} f(x) \qquad 2.106$$

We will compare our implementation of Broyden's method against Powell's method (MINPACK (1980) implementation) which was the one with best results in the report by Hiebert.

In Table 2.1 we present results for the general set and in table 2.2 those for the chemical equilibrium set.

The chemical equilibrium problems are already badly scaled and thus no additional scaling is used for it.

In the tables we present the number of problems for which each code failed. The general set has 54 problems while the chemical equilibrium set has 22.

Table 2.1: Number of failures for the general set (54 problems)

	Broyden	Hybrid
Original scale	12	5
Variable badly scaled	13	21
Function badly scaled	15	22
TOTAL	40	48

Table 2.2: Number of failures for the chemical equilibrium set (12 problems)

	Broyden	Hybrid
Original scale	3	9

We present in table 2.3 a comparison for the implementation with and without the reinitialisation mentioned in 2.2.1.

Finally, table 2.4 shows a summary of results in percentage of success for the Broyden code tested by Hiebert, the hybrid code and our implementation.

Table 2.3: Number of failures for the general set showing the effect of reinitialisation (54 problems)

	With Reinitialisation	Without
Original scale	12	21
Variable badly scaled	13	23
Function badly scaled	15	26
TOTAL	40	70

Table 2.4: Percentages of success for both sets

	General Set	Chemical Equilibrium
Hybrid code	70	25
Broyden (tested by Hiebert)	33	42
Broyden (our implementation)	75	86

2.5 CONCLUSION

The numerical behaviour of a method is extremely dependent on its implementation.

Our implementation of Broyden's method provides a code which gives numerical results competitive with those published so far.

The implementation of Broyden's method can be greatly improved by using a reinitialisation procedure. This is particularly useful when solving flowsheeting problems since the cost of evaluating the initial Jacobian can be reduced considerably by making use of the sparsity of the problem.

CHAPTER 3

SCALING AND QUASI-NEWTON METHODS

We will discuss in this chapter the relation of scaling and Quasi-Newton methods.

First, we will establish sufficient conditions for having scale invariant methods, then we will propose Quasi-Newton methods belonging to the rank-one update family which are scale invariant. We will show that three of the new methods are also least-change secant updates as defined by DENNIS and SCHNABEL (1978). It will also be shown that three of the proposed methods have similar local convergence properties to Broyden's method (ie, superlinear).

Finally, the new methods will be compared with Broyden's method (our implementation mentioned in chapter 2) and the hybrid method tested by Hiebert.

The same set of problems as in chapter 2 will be used plus another set specially selected to investigate the change in behaviour of the methods as the scale of the variables is made worse.

3.1 THE CONDITIONS FOR SCALE INVARIANCE

We have already defined the property of scale invariance in chapter 1.

One important property of Newton's method (the basic method from which most others have been derived) is its scale invariance for changes of scale of the form 1.29.

The basic property of Newton's method which provides scale invariance is the following relation satisfied by the Jacobian

$$\hat{F}'(\hat{x}_k) = D_f \, F'(x_k) D_x^{-1} \qquad 3.1$$

because since $\hat{x}_o = D_x \, x_o$ from 1.22 it follows that

$$\hat{p}^N = D_x \, p^N \qquad 3.2$$

and from 1.23

$$\hat{x}_{k+1} = D_x \, x_{k+1} \qquad 3.3$$

As we mentioned before, Newton's method is implemented normally (when derivatives are not available analytically) using an approximation B_k to $F'(x_k)$ obtained using 2.11 and we can characterize a family of methods, from which Newton's method will be a member using the following relation for B_k

$$B_k = Q_k \, P_k^{-1} \qquad 3.4$$

where Q_k and P_k are obtained from n points x_{ki} and n steps p_{ki} and the corresponding function changes

$$q_{ki} = f(x_{ki} + p_{ki}) - f(x_{ki}), \quad i = 1, 2, \ldots, n \qquad 3.5$$

as

$$P_k = (p_{k1}, p_{k2}, \ldots, p_{kn}) \qquad 3.6$$

$$Q_k = (q_{k1}, q_{k2}, \ldots, q_{kn})$$

For Newton's method (the discrete version) $x_{ki} = x_k$ for all i and $p_{ki} = \lambda e_i$ (λ being the length of the perturbation).

It can be easily verified that any method which corresponds to 3.4 will satisfy

$$\hat{B}_k = D_f \, B_k \, D_x^{-1} \qquad 3.7$$

and this will give us a sufficient condition for scale invariance as the following Lemma shows.

Lemma 3.1: Any method for solving 1.18 which is defined by the following iterative procedure, given x_o and B_o,

$$p_k = -B_k^{-1} \, f_k \qquad 3.8$$

$$x_{k+1} = x_k + p_k \qquad 3.9$$

with B_{k+1} obtained in such a way that for a given change of scale of the form 1.29 it satisfies 3.7,will be scale invariant.

Proof: from 3.8

$$\hat{p}_k = -\hat{B}_k^{-1}\ \hat{f}_k = -(D_x\ B_k^{-1}\ D_f^{-1})\ D_f\ f_k$$

$$= D_x\ p_k$$

and then from 3.9

$$\hat{x}_{k+1} = D_x\ x_{k+1}$$

And then the methods defined by 3.4 are scale invariant. They have been called secant methods.

We have mentioned already the inconvenience of Newton's method regarding its requirements for function evaluations. Methods of the form 3.4 have been proposed but comparisons are not available. PALOSCHI (1980) tested an implementation of a method of this form due to GRAGG and STEWART (1974) which gave good results regarding robustness but its storage requirement makes it unsuitable for solving large problems. (Its efficiency was also poor.) The method of successive substitutions for solving problems of the form 1.11 (fixed point problems) is also scale invariant (in this case $D_x = D_f$, see MALATHRONAS (1979)).

The BROWN (1966) and BRENT (1973) methods are not scale invariant (except in the trivial case $D_x = \alpha I$, see MORE and COSNARD (1979)).

The two Quasi-Newton methods proposed by BROYDEN (1965) are also not scale invariant. The one considered "good" requires D_x to be orthogonal (ie, $D_x^T = D_x^{-1}$, this includes all possible permutations of the variables; see MALATHRONAS and PERKINS (1980)), while D_f should be orthogonal in the case of the "bad" method.

We will devise in the next section Quasi-Newton methods which are scale invariant if D_x is a diagonal matrix.

3.2 SCALE INVARIANT QUASI-NEWTON METHODS

We have already defined in chapter 2 the family of rank-one Quasi-Newton methods. This family, in general, will not be scale invariant as the following analysis shows.

Consider the update formula 2.8 which characterizes this family. If we do a change of scale of the form 1.29 then the new update on the new scale will be

$$\hat{B}_{k+1} = \hat{B}_k + (\hat{y}_k - \hat{B}_k\hat{p}_k)\frac{\hat{v}_k^T}{\hat{v}_k^T\hat{p}_k}$$

$$= \hat{B}_k + D_f(y_k - B_k p_k)\frac{\hat{v}_k^T}{\hat{v}_k^T D_x p_k}$$

and since according to lemma 3.1, B_k must satisfy 3.7 for all k, it follows

$$\hat{B}_{k+1} = D_f B_k D_x^{-1} + D_f(y_k - B_k p_k)\frac{\hat{v}_k^T}{\hat{v}_k^T D_x p_k}$$

$$= D_f\left[B_k + (y_k - B_k p_k)\frac{\hat{v}_k^T D_x}{\hat{v}_k^T D_x p_k}\right] D_x^{-1} \qquad 3.10$$

We can then see that unless v_k is a chosen properly, 3.7 will not be satisfied by B_{k+1}.

The following lemma will state sufficient conditions for rank-one Quasi-Newton methods to be scale invariant.

Lemma 3.2: A rank-one Quasi-Newton method defined with the update formula 2.8 will be scale invariant under changes of scale of the form 1.29 if the sequence of vectors $\{v_k\}$, which characterises the method, is such that

$$\hat{v}_k = D_x^{-1} v_k \qquad 3.11$$

provided that B_o is such that

$$\hat{B}_o = D_f B_o D_x^{-1} \qquad 3.12$$

and D_x is a diagonal matrix.

Proof: Using 3.10 it follows that if B_k satisfies 3.7 then from 3.11, B_{k+1} satisfies 3.7. Using 3.12 it follows by induction that 3.7 is satisfied for all k, and lemma 3.1 then applies.

We will now show the existence of methods satisfying lemma 3.2.

The following notation will be used.

$$x_k = (\xi_{k1}, \xi_{k2}, \ldots, \xi_{kn})$$

$$p_k = (\pi_{k1}, \pi_{k2}, \ldots, \pi_{kn})$$

$$v_k^i = (v_{k1}^i, v_{k2}^i, \ldots, v_{kn}^i)$$

Definition: Define the pseudoinverse α^+ of a real number α as

$$\alpha^+ = \begin{cases} 0 & \text{if } \alpha = 0 \\ 1/\alpha & \text{if } \alpha \neq 0 \end{cases}$$

Given the sequence $\{x_k\}$ define the sequences $\{v_k^1\}$, $\{v_k^2\}$, $\{v_k^3\}$ and $\{v_k^4\}$ such that for each i, $i \leq i \leq n$

$$v_{ki}^1 = \xi_{k+1\ i}^+ \qquad 3.13$$

$$v_{ki}^2 = \pi_{ki}(\xi_{ki}^+)^2 \qquad 3.14$$

$$v_{ki}^3 = \pi_{ki}(\pi_{oi}^+)^2 \qquad 3.15$$

$$v_{ki}^4 = \pi_{ki}((\xi_{ki} - \xi_{oi})^+)^2 \qquad 3.16$$

Lemma 3.3: Each one of the sequences $\{v_k^1\}$, $\{v_k^2\}$, $\{v_k^3\}$ and $\{v_k^4\}$ satisfies 3.11.

Proof: If we denote by δ_i the i-th element in the diagonal of the matrix D_x we can see that

$$\hat{\xi}_{ki} = \delta_i \xi_{ki}$$

$$\hat{\pi}_{ki} = \delta_i \pi_{ki}$$

$$\hat{\xi}_{ki}^{+} = \xi_{ki}^{+} / \delta_i$$

$$\hat{\pi}_{oi}^{+} = \pi_{oi}^{+} / \delta_i$$

$$(\hat{\xi}_{ki} - \hat{\xi}_{oi})^{+} = (\xi_{ki} - \xi_{oi})^{+} / \delta_i$$

and 3.11 follows.

DENNIS and SCHNABEL (1978) proposed the theory for least change secant updates for Quasi-Newton methods. We will show now how it is possible to relate some of the scale invariant methods we are proposing to this theory.

The Frobenius norm for matrices is defined as

$$\|A\|_F = \left(\sum_{i=1}^{n} \sum_{j=1}^{n} (e_i^T A e_j)^2 \right)^{\frac{1}{2}} \qquad 3.17$$

for a matrix $A \in L(\mathbb{R}^n)$.

Quasi-Newton methods are basically characterized by the updating formula for obtaining the approximation B_{k+1} and, as we have mentioned in chapter 2, the basic relation which rules how to obtain B_{k+1} is the secant relation 2.2.

We can define a set of matrices, for two given vectors y and s, as

$$Q(y, s) = \{M \in L(\mathbb{R}^n) : Ms = y\} \qquad 3.18$$

In particular, if $s = p_k$ and $y = y_k$, $Q(y, s)$ will be the set of matrices satisfying the secant relation, ie B_{k+1} must be taken from $Q(y_k, p_k)$. Dennis and Schnabel have shown that the Broyden update (ie, $v_k = p_k$ in 2.8) is the solution to the problem

$$\min_{B \in Q(y_k, p_k)} \|B - B_k\|_F \qquad 3.19$$

They also have shown that if we extend the Frobenius norm by defining the <u>weighted Frobenius norm</u> as

$$\|A\|_{W_1, W_2} = \|W_1 A W_2\|_F \qquad 3.20$$

for two given non-singular matrices W_1 and W_2, then the solution to the problem

$$\min_{B \in Q(y_k, p_k)} \|B - B_k\| \bar{W}, W'' \qquad 3.21$$

is solved by taking $B = B_{k+1}$ with B_{k+1} given by 2.8 where

$$v_k = W^{-T} W^{-1} p_k \qquad 3.22$$

We can then see that some rank-one updates can be obtained as least change updates (that is if it is possible to express v_k as 3.22).

The matrix W can also be changed at each iteration allowing then the more general equation

$$v_k = W_k^{-T} W_k^{-1} p_k \qquad 3.23$$

We can now show that three of the methods we are proposing can be expressed as least change updates under the Dennis and Schnabel theory. For all of them we can find a diagonal matrix W_k such that v_k can be obtained using 3.23. If we express the diagonal elements of the matrix W_k as a vector then this is $\{x_k\}$ for $\{v_k^2\}$, p_o for $\{v_k^3\}$ (we can in this case apply directly 3.22), and $\{x_k - x_o\}$ for $\{v_k^4\}$.

While we have proposed only four different scale invariant methods it is possible to find many more methods satisfying 3.11.

For instance, the family of methods defined by

$$\nu_{ki} = \xi_{ki}^{+} |\pi_{ki}/\xi_{ki}^{+}|^{\alpha} \qquad 3.24$$

with α being independent of the scaling (ie, $\hat{\alpha} = \alpha$) satisfies 3.11, as well as

$$\nu_{ki} = (\xi_{ki} - \xi_{oi})^{+} |\pi_{ki}/(\xi_{ki} - \xi_{oi})^{+}|^{\alpha} \qquad 3.25$$

$$\nu_{ki} = \pi_{oi}^{+} |\pi_{ki}/\pi_{oi}^{+}|^{\alpha} \qquad 3.26$$

The ones we have proposed are special cases of these families.

The two methods $\{v_k^3\}$ and $\{v_k^4\}$ are also invariant under affine changes of the more general form

$$\hat{f}(\hat{x}) = D_f \, f(D_x^{-1}(\hat{x} - b)) \qquad 3.27$$

and so are the families 3.25 and 3.26.

If we assume that W_k is a diagonal matrix then by defining the diagonal matrix D_k as

$$D_k = W_k^{-T} \, W_k^{-1} \qquad 3.28$$

we can use instead of 3.23

$$v_k = D_k \, p_k \qquad 3.29$$

The condition for scale invariance which D_k should satisfy can be deduced from 3.11.

A sufficient condition is

$$\hat{D}_k = D_x^{-1} \, D_k \, D_x^{-1} \qquad 3.30$$

We will show in the next section that rank-one Quasi-Newton methods satisfying our requirements for scale invariance have the same local convergence properties, ie superlinear, as Broyden's method.

3.3 CONVERGENCE RESULTS

In this section we will establish conditions for local superlinear convergence of scale invariant Quasi-Newton methods.

The following theorem, due to BROYDEN, DENNIS, MORE (1973), will be the basis for our convergence results.

Theorem 3.4: Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be differentiable in the open, convex set D and assume that for some x_* in D, $f(x_*) = 0$ and $F'(x_*)$ is nonsingular, and for some $\beta > 0$.

$$\|F'(x) - F'(x_*)\| \leq \beta \|x - x_*\| \qquad 3.31$$

for all $x \in D$.

Assume that there exist $\mu_1 \geq 0$ and a nonsingular symmetric matrix M such that

$$\|Mv_k - M^{-1}p_k\| \leq \mu_1 \|M^{-1}p_k\| \max(\|x_{k+1} - x_*\|, \|x_k - x_*\|) \qquad 3.32$$

for all $x_k \in D$ and B_k such that $\|B_k - F'(x_*)\| < \varepsilon$ for some $\varepsilon > 0$, where p_k is defined by 3.8 and x_{k+1} by 3.9.

Then the method defined by 2.8 is well defined in a neighbourhood of x_* and $F'(x_*)$ and the sequence $\{x_k\}$ is locally and superlinearly convergent at x_*.

Based on our deductions in the previous section we will assume that v_k in 2.8 can be expressed as 3.29.

We can now state the following theorem.

Theorem 3.5: Let $f : \mathbb{R}^n \to \mathbb{R}^n$ satisfy the conditions of theorem 3.4 and let $\{D_k\}$ be a sequence of diagonal matrices such that for a fixed positive definite diagonal matrix D_* the following relation is satisfied for all k.

$$\|D_k - D_*\| \leq n \max(\|x_{k+1} - x_*\|, \|x_k - x_*\|) \qquad 3.33$$

Then the method defined by 2.8, 3.8, 3.9 and 3.29 is locally and superlinearly convergent provided, for a change of scale of the form 1.29, D_k is such that

$$\hat{D}_k = D_x^{-1} D_k D_x^{-1}$$

Proof: Define the change of scale

$$\hat{x}_k = D_*^{\frac{1}{2}} x_k$$

We can see that

$$\|\hat{v}_k - \hat{p}_k\| = \|D_*^{-\frac{1}{2}} v_k - D_*^{\frac{1}{2}} p_k\| = \|D_*^{-1}(D_k - D_*) D_*^{\frac{1}{2}} p_k\|$$

and then

$$\|D_*^{-\frac{1}{2}} v_k - D_*^{\frac{1}{2}} p_k\| \leq \|D_*^{-1}\| \; \|D_k - D_*\| \; \|D_*^{\frac{1}{2}} p_k\|$$

now defining

$$\mu_1 = n\|D_*^{-1}\|$$

and from 3.33

$$\|D_*^{-\frac{1}{2}} v_k - D_*^{\frac{1}{2}} p_k\| \leq \mu_1 \max(\|x_{k+1} - x_*\|, \|x_k - x_*\|) \; \|D_*^{\frac{1}{2}} p_k\|$$

We now apply theorem 3.4, identifying M and $D_*^{-\frac{1}{2}}$.

With the following lemma we will show that three of the scale invariant methods we are proposing satisfy the conditions of theorem 3.5.

Lemma 3.5: The sequences of vectors $\{v_k^2\}$, $\{v_k^3\}$ and $\{v_k^4\}$ satisfy condition 3.33, provided for $\{v_k^2\}$ x_* has no zero components and for $\{v_k^4\}$ $e_i^T(x_* - x_o) \neq 0$, $i = 1, n$.

Proof: We will use the notation δ_{ki} for i-th diagonal element of the matrix D_k.

For $\{v_k^2\}$ define D_* such that

$$\delta_{*i} = (\xi_{*i}^+)^2$$

In this case D_k is such that

$$\delta_{ki} = (\xi_{ki}^+)^2$$

and then

$$\delta_{ki} - \delta_{*i} = (\xi_{*i}^2 - \xi_{ki}^2)(\xi_{*i}^+)^2(\xi_{ki}^+)^2$$

Since 2.27 must be satisfied only for an open set containing x_*, and since we are assuming x_* to have all its components away from the origin, then it is possible to show that there exists a constant γ_1 such that

$$(\xi_{*i} + \xi_{ki})^2(\xi_{*i}^+)^4(\xi_{ki}^+)^4 \leq \gamma_1$$

and then

$$(\delta_{ki} - \delta_{*i})^2 \leq \gamma_1(\xi_{*i} - \xi_{ki})^2$$

it now follows

$$\|D_k - D_*\| \leq \gamma_1^{\frac{1}{2}}\|x_k - x_*\|$$

and 3.33 is immediate.

For $\{v_k^3\}$ define D_* such that

$$\delta_{*i} = (\pi_{oi}^+)^2$$

Since $D_* = D_k$ in this case the proof is trivial.

And finally for $\{v_k^4\}$ define D_* such that

$$\delta_{*i} = ((\xi_{*i} - \xi_{oi})^+)^2$$

D_k will in this case be such that

$$\delta_{ki} = ((\xi_{ki} - \xi_{oi})^+)^2$$

and again, as we did for $\{v_k^2\}$, we can find γ_2 such that

$$(\delta_{ki} - \delta_{*i})^2 \leq \gamma_2(\xi_{*i} - \xi_{ki})^2$$

and 3.33 follows as before.

We have shown in 3.2 that least change secant updates, as defined by DENNIS and SCHNABEL (1978), can be expressed using

$$v_k = W_k^{-T} W_k^{-1} p_k$$

If the matrices W_k are such that

$$D_k = W_k^{-T} W_k^{-1}$$

is a diagonal matrix and if for a change of scale of the form 1.29 it satisfies

$$\hat{D}_k = D_x^{-1} D_k D_x^{-1}$$

then we can apply our convergence results for any least change secant update which can be expressed in this way.

3.4 NUMERICAL RESULTS

We will present in this section numerical results obtained testing the methods we have proposed in this chapter.

The new scale invariant methods will be compared against our implementation of Broyden's method presented in chapter 2 and the hybrid method tested by Hiebert.

We will use the same sets of mathematical examples as in chapter 2 plus an additional set which is described in Appendix A and which we have named "general subset".

This set has been constructed by taking from the general set those problems for which, with the standard initial point, all methods considered converged. For studying the behaviour of the different methods under a gradual deterioration of the scaling we will use the change of scale

$$\hat{f}(\hat{x}) = f(S_{mn}\ \hat{x})$$

for m = 0, 4, 8, 12, and 16; with S_{mn} as defined in chapter 2.

We will refer to the different methods using the following nomenclature.

SI1 Scale invariant method using $\{v_k^1\}$

SI2 " " " " $\{v_k^2\}$

SI3 " " " " $\{v_k^3\}$

SI4 " " " " $\{v_k^4\}$

BRO Our implementation of Broyden's method (as in chapter 2)

HYB The hybrid method tested by Hiebert

To compare the methods regarding efficiency a number c_j is defined for each method on each problem as:

$$c_j = \begin{cases} \text{not defined} & \text{if the method has failed to converge} \\ n_j/n_o & \text{if the method has converged} \end{cases}$$

where j indicates the method, n_j is the number of function evaluation used for method j and n_o is the number of function evaluations used for the most efficient method of all on this particular problem.

The results obtained are condensed in tables 3.1 to 3.3 for the three considered sets, the general set, the general subset and the chemical equilibrium set.

The first half of the table shows the number of failures for each method while the second half gives the averages for the efficiency number c_j. Note that average is based on number of successes for particular methods.

Table 3.1: Summary of results for the GENERAL SET

		SI1	SI2	SI3	SI4	BRO	HYB
Fails	Unscaled	14	12	13	14	12	5
	Var badly scaled	12	14	13	13	13	21
	Func " "	18	18	14	14	15	22
	TOTAL	44	44	40	41	40	48
Averages of c_j	Unscaled	1.49	2.10	1.71	1.73	1.84	1.33
	Var badly scaled	1.55	1.81	1.77	1.65	1.50	1.33
	Func " "	1.77	1.26	1.27	1.50	1.44	1.41
	TOTAL	1.60	1.75	1.59	1.63	1.60	1.35

Table 3.2: Summary of results for the GENERAL SUBSET

	m	SI1	SI2	SI3	SI4	BRO
Fails	0	0	0	0	0	0
	4	0	0	0	0	0
	8	3	0	0	1	3
	12	9	8	5	5	10
	16	14	14	11	12	14
	TOTAL	26	22	16	18	27
Averages of c_j	0	1.40	1.37	1.12	1.27	1.24
	4	1.53	1.37	1.11	1.20	1.27
	8	1.63	1.57	1.28	1.17	1.28
	12	6.07	3.82	1.37	1.20	9.55
	16	2.75	1.40	3.11	1.16	1.04
	TOTAL	2.15	1.76	1.35	1.21	2.19

Table 3.3: Summary of results for the CHEMICAL EQUILIBRIUM SET

	SI1	SI2	SI3	SI4	BRO	HYB
Fails	3	3	3	3	3	9
Averages of c_j	1.28	1.53	1.34	1.40	2.94	1.05

Finally, in table 3.4, we give a complete summary showing percentages of success for all methods on all sets.

Table 3.4: Percentage of success of all methods on all sets

	SI1	SI2	SI3	SI4	BRO	HYB
GENERAL SET	73	73	75	75	75	70
GENERAL SUBSET	68	73	80	78	66	-
CHEM EQUILIB SET	75	75	75	75	75	25

The hybrid code was not tested with the general subset because it has shown already very bad behaviour when variables are badly scaled.

The results for the general set (table 3.1) show no significant difference between our scale invariant method SI3, SI4 and BRO (Broyden's method). The results for the remaining scale invariant methods do not differ too much. As before, all Quasi-Newton methods show better robustness (specially on the badly scaled cases) and slightly less efficiency than the hybrid code.

The results for the general subset show clearly the advantage of the scale invariant code, all of them failing in less cases. SI3 and SI4 give the best results in efficiency and robustness.

For the chemical equilibrium set all scale invariant methods fail on three cases as does BRO. There is an important difference in efficiency but it is caused by a particular problem for which Broyden's method takes many more iterations than the rest.

Finally, table 3.4 shows the global superiority of the scale invariant methods.

3.5 CONCLUSIONS

We have presented in this chapter scale invariant Quasi-Newton methods which are competitive with Broyden's method. The results obtained for

the general subset (when the variables are gradually badly scaled) show clearly the advantage of using the theoretically scale-invariant methods. Scale invariance, in practice, is only achieved if infinite precision arithmetic is used. Since this is not the case different results are obtained with different scales as the tables show. There is still room for improvement in the numerical conditioning of the methods in order to minimize the effects of finite precision arithmetic being used. We will show in the next chapter possible ways to deal with this problem.

The scale invariant methods SI3 and SI4 are the best alternatives according to our numerical results.

CHAPTER 4

THE NUMERICAL CONDITIONING OF QUASI-NEWTON METHODS

We will discuss in this chapter some ideas for optimizing the numerical conditioning of Quasi-Newton methods.

We will first present the concept of condition number for general matrices and its relation to the numerical conditioning of systems of non-linear equations.

Some ideas for optimizing the numerical conditioning will be proposed and tested.

First, an explicit expression for the condition number of the approximation to the Jacobian for general Quasi-Newton methods will be derived which will allow us to select Quasi-Newton methods with optimally conditioned updates. Secondly, an internal scaling procedure will be proposed for optimizing the condition number of both the problem and the approximation to the Jacobian.

Finally we will present numerical results obtained with the proposals of this chapter.

4.1 THE CONDITION NUMBER

The condition number has been introduced as a measure of the numerical conditioning for general matrices (see TODD (1966)).

For a non-singular matrix $A \in L(\mathbb{R}^n)$ the condition number $\kappa(A)$ is defined as

$$\kappa(A) = \|A\| \, \|A^{-1}\| \tag{4.1}$$

for a given matrix norm $\| \ \|$.

If A defines a system of linear equations in $\mathbb{R}^n$

$$A x = b \tag{4.2}$$

it is a well known result (ORTEGA RHEINBOLDT (1970)) that if $B \in L(\mathbb{R}^n)$ is close to A in the sense that

$$\|A^{-1}\| \; \|B - A\| < 1 \qquad 4.3$$

then B is also non-singular, and for $b \neq 0$ the solutions x_* of 4.2 and y_* of

$$B\,x = c \qquad 4.4$$

satisfy the estimate (see RHEINBOLDT(1974))

$$\frac{\|x_* - y_*\|}{\|x_*\|} \leq \frac{\kappa(A)}{1 - \kappa(A)\|B - A\|/\|A\|} \left(\frac{\|B - A\|}{\|A\|} + \frac{\|b - c\|}{\|b\|} \right) \qquad 4.5$$

As an example of the use of this number let us assume that in solving numerically the equation 4.2 we have found an approximation y_* to x_* (the exact solution).

If $\kappa(A)$ is a large number then the fact that $Ay_* \cong b$ does not mean that y_* is close to x_*; we can deduce this using 4.5 to obtain

$$\frac{\|x_* - y_*\|}{\|x_*\|} \leq \kappa(A) \frac{\|b - Ay_*\|}{\|b\|}$$

In general we can say that the smaller $\kappa(A)$, the better the result obtained in solving numerically 4.2.

This important result has led to finding ways for transforming 4.2 into an equivalent linear system having the same solution but smaller condition number.

This concept of condition number for linear systems has been generalized to systems of non-linear equations by RHEINBOLDT (1974) as follows:

For a given function $f : D \subset \mathbb{R}^n \Rightarrow \mathbb{R}^n$, closed set $C \subset D$ and point $z \in C$ define

$$\mu(f, C, z) = \sup \{t \in [0, \infty); \|f(x) - f(z)\| \geq t\|x-z\|, \forall x \in C\} \qquad 4.6$$

$$\nu(f, C, z) = \inf \{t \in [0, \infty); \|f(x) - f(z)\| \leq t\|x-z\|, \forall x \in C\}$$

and then, define the localized condition number

$$\kappa(f, C, z) = \begin{cases} \dfrac{\nu(f, C, z)}{\mu(f, C, z)} \text{ if } 0 < \mu(f, C, z), \nu(f, C, z) < \infty \\ \\ \infty \qquad \text{otherwise} \end{cases} \qquad 4.7$$

It can be shown that 4.7 reduces to 4.5 if f is a linear function.

REINBOLDT (1974) has shown that if f is a continuous function in D and if the Jacobian F'(x) of f is nonsingular in D then for any $\varepsilon > 0$ there is a $\delta > 0$ such that if

$$C = \{x \in \mathbb{R}^n, \ \|x - z\| \leq \delta\} \subset D \qquad 4.8$$

then

$$|\nu(f, C, z) - \|F'(z)\| \ | \leq \varepsilon \qquad 4.9$$

$$|\mu(f, C, z) - \|F'(z)^{-1}\|^{-1}| \leq \varepsilon \qquad 4.10$$

and then, asympotically near z, the conditioning of the non-linear function f and its Jacobian F'(z) are the same.

An equivalent formula to 4.5 is obtained for the non-linear case and then the condition number for systems of non-linear equations plays a similar role as it does for linear systems.

4.2 QUASI-NEWTON METHODS AND THE CONDITION NUMBER

The numerical performance of Algorithm 1 in chapter 2 is affected by the condition number in two different ways.

(a) The conditioning of the problem 1.18 itself, as explained in section 4.1.

(b) The conditioning of B_k since step 5 of the algorithm implies solving the linear system

$$B_k\, p_k = -\lambda_k\, f_k \qquad 4.11$$

We will propose in section 4.2.1 a way of dealing with (b) and in section 4.2.2 how to deal simultaneously with (a) and (b).

4.2.1 Optimizing the condition number using different methods

Step 5 in algorithm 1 involves the solution of a linear system. According to our discussion in section 4.1 we should try to optimize the condition number of B_k in order to obtain better numerical results. For this, we will make use of the degrees of freedom provided by v_k in step 8 of the algorithm, choosing it such that the condition number of B_{k+1} is minimized.

The following property will allow us to obtain an explicit expression for $\kappa(B_{k+1})$.

Property 1: For any $A \in L(\mathbb{R}^n)$, $b \in \mathbb{R}^n$, $c \in \mathbb{R}^n$, the Frobenious norm for matrices and the Euclidean norm for vectors.

$$\|A + bc^T\|_F^2 = \|A\|_F^2 + \|b\|_2^2\, \|c\|_2^2 + 2b^TAc \qquad 4.12$$

Proof:

$$\|A + bc^T\|_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} (e_i^TAe_j + e_i^Tbc^Te_j)^2$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} \Big((e_i^TAe_j)^2 + (e_i^Tb)^2(e_j^Tc)^2 + 2(e_i^Tb)(e_i^TAe_j)(c^Te_j) \Big)$$

$$= \|A\|_F^2 + \|b\|_2^2\, \|c\|_2^2 + 2b^T \sum_{i=1}^{n} \sum_{j=1}^{n} (e_ie_i^TAe_je_j^T)c$$

$$= \|A\|_F^2 + \|b\|_2^2\, \|c\|_2^2 + 2b^TAc$$

In what follows we will assume $\| \ \|_F$ for matrices and $\| \ \|_2$ for vectors.

We can then evaluate from 2.8 and Property 1

$$\|B_{k+1}\|^2 = \|B_k\|^2 + \frac{\|y_k - B_k p_k\|^2 \cdot \|v_k\|^2}{(v_k^T p_k)^2}$$

$$+ \frac{2(y_k - B_k p_k)^T B_k v_k}{v_k^T p_k} \qquad 4.13$$

If we define $H_k = B_k^{-1} V_k$ and using the SHERMAN and MORRISON (1949) formula, we can obtain from 2.8.

$$H_{k+1} = H_k + \frac{(p_k - H_k y_k) v_k^T H_k}{v_k^T H_k y_k} \qquad 4.14$$

and then using again Property 1

$$\|H_{k+1}\|^2 = \|H_k\| + \frac{\|p_k - H_k y_k\|^2 \cdot \|v_k^T H_k\|^2}{(v_k^T H_k y_k)^2}$$

$$+ \frac{2(p_k - H_k y_k)^T H_k H_k^T v_k}{v_k^T H_k y_k} \qquad 4.15$$

We can now evaluate, using 4.13 and 4.15,

$$\kappa(B_{k+1})^2 = \|B_{k+1}\|^2 \ \|H_{k+1}\|^2 \qquad 4.16$$

and thus we have found an explicit way of evaluating the condition number for the approximation to the Jacobian obtained using rank-one Quasi-Newton methods.

Define now

$$\alpha_1 = \|y_k - B_k p_k\|^2$$

$$\beta_3 = \|p_k - H_k y_k\|^2$$

$$a_1^T = (y_k - B_k p_k)^T B_k$$

$$b_4^T = (p_k - H_k y_k)^T H_k$$

$g_1 : \mathbb{R}^n \Rightarrow \mathbb{R}$ as

$$g_1(v) = \|B_k\|^2 + \alpha_1 \frac{v^T v}{(v^T p_k)^2} + 2 \frac{a_1^T v}{v^T p_k} \qquad 4.17$$

and $g_2 : \mathbb{R}^n \Rightarrow \mathbb{R}$ as

$$g_2(v) = \|H_k\|^2 + \beta_3 \frac{v^T H_k H_k^T v}{(v^T H_k y_k)^2} + 2 \frac{b_4^T H_k^T v}{v^T H_k y_k} \qquad 4.18$$

If we define $g : \mathbb{R}^n \Rightarrow \mathbb{R}$ as

$$g(v) = g_1(v)\, g_2(v)$$

then it follows

$$\kappa(B_{k+1}) = g(v_k) \qquad 4.19$$

If we are interested in minimizing the condition number of B_{k+1} using v_k we should solve:

$$\min_{v \in \mathbb{R}^n} g(v) \qquad 4.20$$

Since the only way to solve this problem is to do it numerically and this will be an iterative process itself (with no guarantee of convergence in a finite number of steps) it should be discarded as a way of optimizing 4.19.

We propose instead to restrict v_k such that the optimization problem 4.20 can be solved explicitly or very easily.

FLETCHER (1970), in dealing with the Quasi-Newton methods for optimization, introduced the concept of "dual" formulas. He noted that the secant relation satisfied by all Quasi-Newton methods

$$B_{k+1}\, p_k = y_k \qquad 4.21$$

could be thought of as a mapping giving y_k from p_k and then a dual formula could be obtained from any formula satisfying 4.21 replacing B_{k+1} by H_{k+1}, p_k by y_k and y_k by p_k obtaining a formula satisfying

$$H_{k+1}\, y_k = p_k \qquad 4.22$$

which is equivalent to 4.21. In the case of Broyden's "good" formula ($v_k = p_k$ in 2.8) the corresponding "dual" is the "bad" Broyden's formula ($v_k^T = y_k^T B_k$).

If we define an updating formula by H^o and its corresponding dual by H^d, Fletcher defined a family of formulae obtained from

$$H = (1 - \theta)H^o + \theta H^d \qquad 4.23$$

He then chose θ optimizing in some sense the conditioning of H.

DAVIDON (1975) using the same concept introduced new formulas choosing θ to optimize the condition number of $H_k^{-\frac{1}{2}}\, H_{k+1} H_k^{-\frac{1}{2}}$ (in this case H is symmetric and positive definite).

Following the same idea we will use a parameter θ for the optimization of 4.20.

Instead of 4.23 we will define v in terms of θ as

$$v_k = p_k + \theta u_k \qquad 4.24$$

Note that, with v_k given by 4.24, if we define

$$\varepsilon_{00} = \|B_k\|^2 \qquad \varepsilon_{20} = \|p_k\|^2$$

$$\varepsilon_{10} = \alpha_1 \|p_k\|^2 \qquad \varepsilon_{21} = u_k^T p_k$$

$$\varepsilon_{11} = 2\alpha_1 u_k^T p_k \qquad \varepsilon_{30} = 2a_1^T p_k$$

$$\varepsilon_{12} = \alpha_1 \|u_k\|^2 \qquad \varepsilon_{31} = 2a_1^T u_k$$

equation 4.13 becomes

$$\|B_{k+1}\|^2 = \varepsilon_{00} + \frac{\varepsilon_{10} + \varepsilon_{11}\theta + \varepsilon_{12}\theta^2}{(\varepsilon_{20} + \varepsilon_{21}\theta)^2} + \frac{\varepsilon_{30} + \varepsilon_{31}\theta}{\varepsilon_{20} + \varepsilon_{21}\theta} \qquad 4.25$$

and by defining

$$\delta_{00} = \|H_k\|^2 \qquad \delta_{20} = p_k^T H_k y_k$$

$$\delta_{10} \equiv \|p_k - H_k y_k\|^2 \, \|p_k^T H_k\|^2 \qquad \delta_{21} = u_k^T H_k y_k$$

$$\delta_{11} = 2\|p_k - H_k y_k\|^2 \, p_k^T H_k H_k^T u_k \qquad \delta_{30} = 2(p_k - H_k y_k)^T H_k H_k^T p_k$$

$$\delta_{12} = \|p_k - H_k y_k\|^2 \, \|H_k^T u_k\|^2 \qquad \delta_{31} = 2(p_k - H_k y_k)^T H_k H_k^T u_k$$

equation 4.15 becomes

$$\|H_{k+1}\|^2 = \delta_{00} + \frac{\delta_{10} + \delta_{11}\theta + \delta_{12}\theta^2}{(\delta_{20} + \delta_{21}\theta)^2} + \frac{\delta_{30} + \delta_{31}\theta}{\delta_{20} + \delta_{21}\theta} \qquad 4.26$$

We will then solve, instead of 4.20, the simpler problem

$$\min_{\theta \in \mathbb{R}} g(v_k) \qquad 4.27$$

with v_k given by 4.24.

We still have to choose u_k and from this two methods will result.

First we choose

$$u_k^T = y_k^T B_k - p_k^T \qquad 4.28$$

With this choice, v_k is such that for $\theta = 0$ it gives the Broyden's "good" formula while $\theta=1$ it gives its dual.The method resulting from solving 4.27 will be called method 1a; in addition we will define <u>method 1b</u> as the one solving instead of 4.27.

$$\min_{\theta=0 \text{ or } \theta=1} g(v_k) \qquad 4.29$$

The determination of θ for method 1b is immediate.

In the case of method 1a a typical graph of $g(v_k)$ versus θ is presented in fig 1

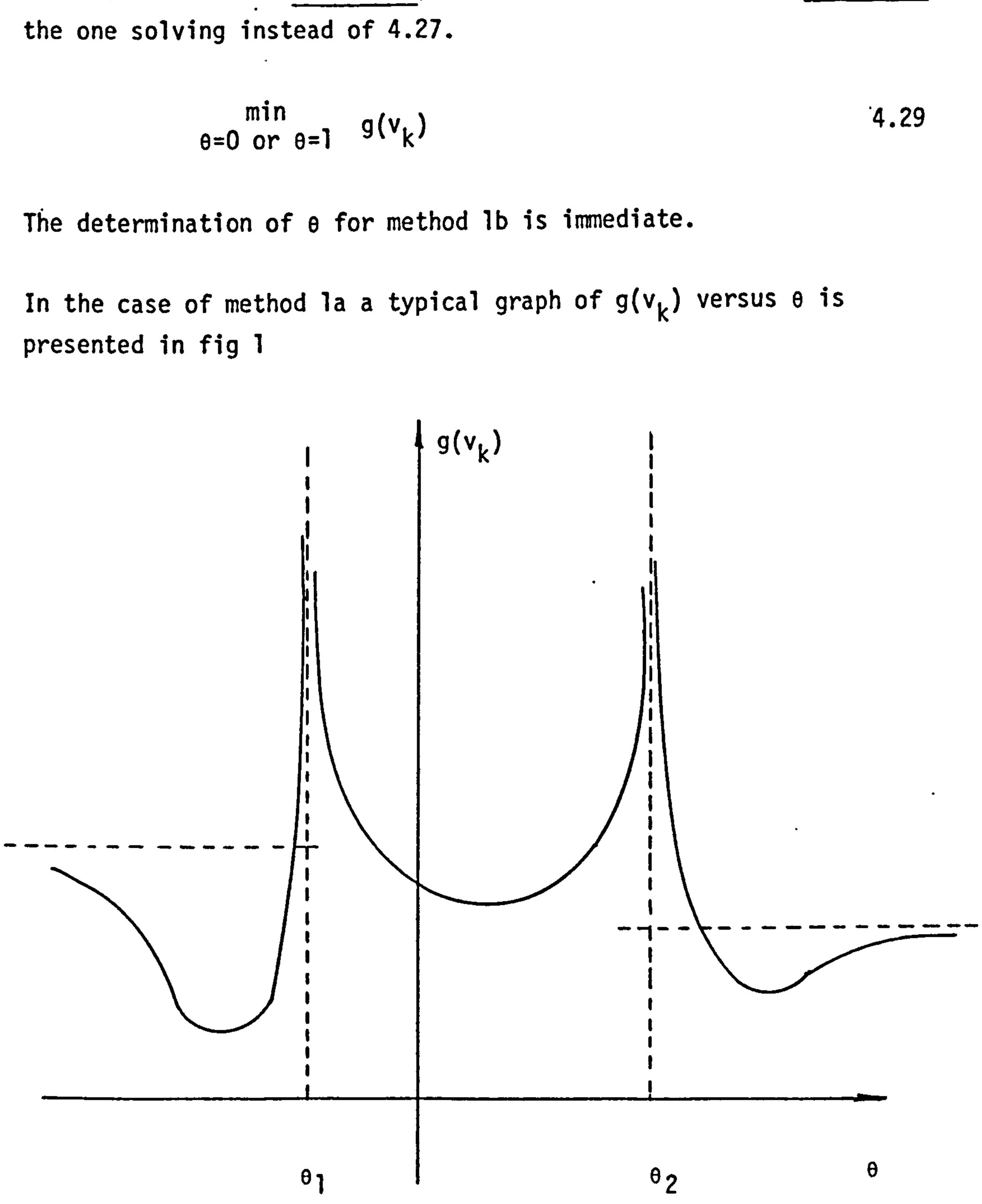


Fig 1: Typical graph of $g(v_k)$

The value θ_1 and θ_2 correspond to the annihilation of the denominators of 4.25 and 4.26. We found the value for θ solving 4.27 using Newton's method which converges very quickly to each of the three local minima , the initial points were taken as $\theta_1 - \varepsilon$, $\theta_1 + \varepsilon$ and $\theta_2 + \varepsilon$ for some $\varepsilon > 0$.

Finally, we will define <u>method 2</u> choosing u_k such that $\varepsilon_{21} = 0$ and $\delta_{21} = 0$ because this will enable us to solve 4.27 explicitly (in this case $g(v_k)$ is a polynomial in θ of order 4). In this case we need

$$u_k^T p_k = 0$$

and

$$u_k^T H_k y_k = 0$$

which can be obtained taking u_k as the projection on the subspace orthogonal to p_k and $H_k y_k$ of the vector u_k for the method 1a. This restricts this method to problems for which $n > 2$.

4.2.2 <u>Optimizing the condition number using scaling</u>

We have shown in the previous section how we can optimize the condition number of the martix used in step 5 of algorithm 1, but in doing that the condition number of the problem 1.18 remains unchanged.

We will show in this section how we can achieve both things simultaneously.

If the first approximation B_o in algorithm 1 is obtained by using finite differences (as we have proposed in chapter 2) it follows from 3.7 that scaling the variables (or function) at the first iteration is equivalent to postmultiplying (premultiplying for the function) the approximation B_o by the scaling matrix.

The following Lemma will show that scaling the function every iteration is equivalent to premultiplying the Jacobian approximation by the scaling matrix.

Lemma 4.1: If we use the change of scale

$$\hat{f}_k = D_f f_k$$

in algorithm 1, B_o is such that

$$\hat{B}_o = D_f B_o$$

and

$$\hat{v}_k = v_k$$

then

$$\hat{B}_k = D_f B_k, \forall_k$$

Proof: assume

$$\hat{B}_k = D_f B_k, \; k \leq m$$

From 2.8 it follows

$$\hat{B}_{m+1} = \hat{B}_m + (\hat{y}_m - \hat{B}_m \hat{p}_m) \frac{\hat{v}_m^T}{\hat{v}_m^T \hat{p}_m}$$

and then from the hypothesis

$$\hat{B}_{m+1} = D_f B_m + D_m (y_m - B_m p_m) \frac{v_m^T}{v_m^T p_m}$$

and hence

$$\hat{B}_{m+1} = D_f B_{m+1}$$

The proof follows by induction.

This Lemma shows that by altering the scaling matrix D_f at each iteration we premultiply B_k by a different matrix each iteration.

The following theorem due to BAUER (1963) will be the basis for our choice of the internal scaling.

Theorem 4.2: For a nonsingular matrix $A \in L(TR^n)$ and nonsingular diagonal matrices D_1 and D_2, using the maximum norm for matrices,

$$\min_{D_1} \kappa(D_1 A) \qquad 4.30$$

and

$$\min_{D_2} \kappa(A D_2) \qquad 4.31$$

are achieved for D_1 and D_2 determined from

$$|A^{-1}|e = D_2\, e \qquad 4.32$$

$$|A|\, e = D_1^{-1}\, e \qquad 4.33$$

($|A|$ means the original matrix with all its elements taken in absolute value.)

Our strategy for the internal scaling will be as follows:

(a) At the first iteration scale the variables using

$$\hat{x}_o = D_2\, x_o$$

where the diagonal matrix D_2 is obtained from

$$|B_o^{-1}|\, e = D_2\, e$$

(b) At every iteration scale the function using

$$\hat{\hat{f}}_k = D_1\, f_k$$

where the diagonal matrix D_1 is obtained at each iteration from

$$|B_k|\ e = D_1^{-1}\ e$$

The variables are scaled the first iteration in order to compensate for any bad scaling related to them.

If we want to scale the variables every iteration then we have to be able to change all our data from the old scale to the new scale. This means we have to be able, given B_k, to obtain $\hat{B}_k$ for the new scale. If the method used is not scale invariant then this means that

$$\hat{B}_k \neq B_k\ D_2^{-1} \qquad 4.34$$

in general (because we will have different updates for different scales) and then we cannot obtain $\hat{B}_k$ based only on B_k and D_2.

Even in the case of scale invariant methods (for which 4.34 becomes an equality) since we need B_k^{-1} in order to obtain D_2 from 4.32, it will make the scaling very expensive.

Lemma 4.1 shows that since all updates of the form 2.8 are scale invariant for function scaling, it is possible to scale the function every iteration. In this case

$$\hat{B}_k = D_1\ B_k$$

and it is very simple to change the scales. Also since B_k is needed to obtain D_1 using 4.33, it is not an expensive procedure.

With this choice of the internal scaling we are minimizing the condition number of B_k at each iteration (ie, we obtain an effect similar to the one of section 4.2.1). If B_k is not far from the real Jacobian $F'(x_k)$ (which is not necessarily true as has been shown by DENNIS and MORE (1977), even when convergence is achieved) we will then also minimize the condition number of the problem 1.18 as we have explained at the beginning of this chapter.

4.3 NUMERICAL RESULTS

We will present in this section numerical results for our proposals.

The same sets of mathematical examples as in chapters 2 and 3 will be used, ie the "general set", "general subset" and the "chemical equilibrium set". In addition a new set called "restricted set" will be used and its description can be found in Appendix 1. It has been obtained from the "general set" taking only those problems with $n > 2$ (this is because one of the methods we have proposed in section 4.2.1 requires $n > 2$).

The table of results have the same structure as those of chapter 3, ie, we will present a number of failures and the averages for the coefficients c_j. The methods will be named as in chapter 3 plus the ones we have proposed in section 4.2.1 which will be named as follows.

C1B Method 1a of section 4.2.1

C1B Method 1b of section 4.2.1

C2 Method 2 of section 4.2.1

In table 4.1 we present the results for the methods proposed in section 4.2.1. They are compared against our implementation of Broyden's method of chapter 2. The results clearly show that the optimization of the condition number of B_k as we have proposed is not a good idea.

Table 4.1: Summary of results for the RESTRICTED SET

		C1A	C1B	C2	BRO
Fails	Unscaled	15	15	14	11
	Var badly scaled	23	17	19	10
	Func badly scaled	20	15	13	11
	TOTAL	58	47	46	32
Averages of c_j	Unscaled	3.40	3 .76	3.84	1.79
	Var badly scaled	2.29	1.66	2.16	1.73
	Func badly scaled	2.93	2.43	3.37	1.55
	TOTAL	2.93	2.64	3.18	1.69

In the following three tables we test the effect of the internal scaling we proposed in 4.2.2 when used for implementing Broyden's method as presented in chapter 2 and the scale invariant methods proposed in chapter 3. Results are also shown, for comparison, for the hybrid code mentioned in the previous comparisons (which as we mentioned before has a facility for an internal scale which is not used since it does not improve its performance).

Table 4.2: Summary of results for the GENERAL SET using internal scaling

		SI1	SI2	SI3	SI4	BRO	HYB
Fails	Unscaled	14	10	13	10	13	5
	Var badly scaled	13	10	14	12	13	21
	Func badly scaled	12	8	13	11	12	22
	TOTAL	39	28	40	33	38	48
Averages of c_j	Unscaled	1.46	1.42	1.42	1.69	1.27	1.31
	Var badly scaled	1.54	1.65	1.40	1.66	1.31	1.30
	Func badly scaled	1.47	1.34	1.45	1.47	1.58	1.42
	TOTAL	1.49	1.47	1.42	1.60	1.39	1.34

Table 4.3: Summary of results for the GENERAL SET
Using internal scaling

	m	SI1	SI2	SI3	SI4	BRO
Fails	0	0	0	0	0	0
	4	0	0	0	0	1
	8	2	1	1	1	1
	12	7	6	6	5	6
	16	14	12	11	11	8
	TOTAL	23	19	18	17	16
Averages of c_j	0	1.28	1.11	1.08	1.08	1.05
	4	1.34	1.14	1.12	1.27	1.08
	8	1.55	1.30	1.17	1.35	1.21
	12	4.82	3.78	1.30	1.38	1.27
	16	1.00	5.49	1.60	2.29	1.28
	TOTAL	1.91	1.89	1.19	1.37	1.16

Table 4.4: Summary of results for the CHEMICAL EQUILIBRIUM SET Using internal scaling

	SI1	SI2	SI3	SI4	BRO	HYB
Fails	1	2	1	1	1	9
Averages of c_j	2.16	1.38	2.00	1.29	1.93	1.04

In table 4.5 we present a summary of results showing the percentage of success for all methods on all sets using the internal scaling. For comparison, the percentages of success obtained without the scaling are shown.

Table 4.5: Percentages of success for all methods on all sets using the internal scaling

	SI1	SI2	SI3	SI4	BRO	HYB
GENERAL SET	76 (73)	83 (73)	75 (75)	80 (75)	77 (75)	70
GENERAL SUBSET	71 (68)	76 (73)	78 (80)	79 (78)	80 (66)	-
CHEM EQUILIB SET	92 (75)	83 (75)	92 (75)	92 (75)	92 (75)	25

( ) is the percentage of success without the internal scaling.

It is clear from the results that the use of the internal scaling improves the robustness of the codes.

4.4 CONCLUSIONS

We have suggested ways of improving the numerical conditioning of code for solving the problem 1.18. The optimization of the condition number for the approximation to the Jacobian using different Quasi-Newton methods is not a good idea while optimizing it using internal scaling gives excellent results.

CHAPTER 5

CHEMICAL ENGINEERING EXAMPLES

In previous chapters we have tested all our proposals using mathematical examples. This has allowed us to compare thoroughly the different alternatives on a wide set of examples and identify those which perform best.

In this chapter we will present numerical results obtained testing our best methods using Chemical Engineering flowsheeting examples.

The examples were taken from various sources and two of them, in particular, are benchmark problems proposed by the European Federation of Chemical Engineers to test the performance of different steady state simulation packages.

The set of problems will involve the following features:

- Involving only mass balances.
- Involving energy and mass balances (hence physical properties such as enthalpy, K-values are needed).
- Physical properties provided as procedures.
- Physical properties provided as equations (ie, no internal iterations are required).
- Strictly simulation problems (all feeds and equipment parameters known).
- Design problems.
- Mixing of equations and procedures.

The simulations package to be used will be the SPEEDUP system developed at Imperial College.

5.1 THE SIMULATION PACKAGE SPEEDUP

We will run our examples using the simulation package SPEEDUP developed at Imperial College (SARGENT, PERKINS and THOMAS (1982)) which was mentioned in chapter 1.

In this package, each unit in the flowsheet is represented by equations, procedures or any combination of both. All procedures have the following format:

(output parameters list) NAME OF PROCEDURE (input parameters list)

Physical properties are available from the package FLOWPACK1 of ICI, a Chao Seader correlation being used.

Our numerical methods presented in the previous chapters, are available as options in SPEEDUP.

SPEEDUP, based on a flowsheet description for the connection between the units, a description of the models representing the units plus an assignment between models and units will create a FORTRAN subroutine for evaluating the residual vector. Once the known variables have been set, SPEEDUP checks for a consistent system to solve (ie, same number of variables and equations) and then will decompose the problem, if possible, into irreducible blocks as explained in section 1.2. The problem will then be solved, block by block, with the numerical method chosen by the user.

An important facility of SPEEDUP which has been very useful to us is the possibility of "typing" the variables and give to each type a default initial guess, a lower bound and an upper bound.

The form of the facility we used is for instance:

TEMP = 300 : 250 : 350

where 300 is the default initial guess, 250 the lower bound and 350 the upper bound for the variable type TEMP.

The initial approximation to the Jacobian in SPEEDUP is obtained by finite differences using the algorithm of CURTIS, POWELL and REID (1974) which, for sparse systems, reduces the number of function evaluations (n+1 is the maximum required) used to obtain the approximation. COLEMAN and MORE (1981) have recently presented an algorithm to find the minimum of those function evaluations but the code implementing it is not available to us.

5.2 FLOWSHEETING PROBLEMS

We will describe briefly in this section the flowsheeting problems which will be used for testing our methods. A more detailed description is given in Appendix B.

5.2.1 Small networks involving heat exchangers

Two problems representing small heat exchanger networks will be used. We will refer to them by HEX1 and HEX2.

They were both taken from GROSSMANN (1978) and are used as modified by EDWARDS (1982).

HEX1 is a small network of four heat exchangers and one condenser. The flowsheet is presented in figure 5.1.

HEX2 consists of three heat exchangers, a splitter and a mixer. In figure 5.2 its flowsheet is presented.

We have solved these problems using two approaches: using only equations to model the heat exchanger and using procedures. Constant stream specific heats will be assumed.

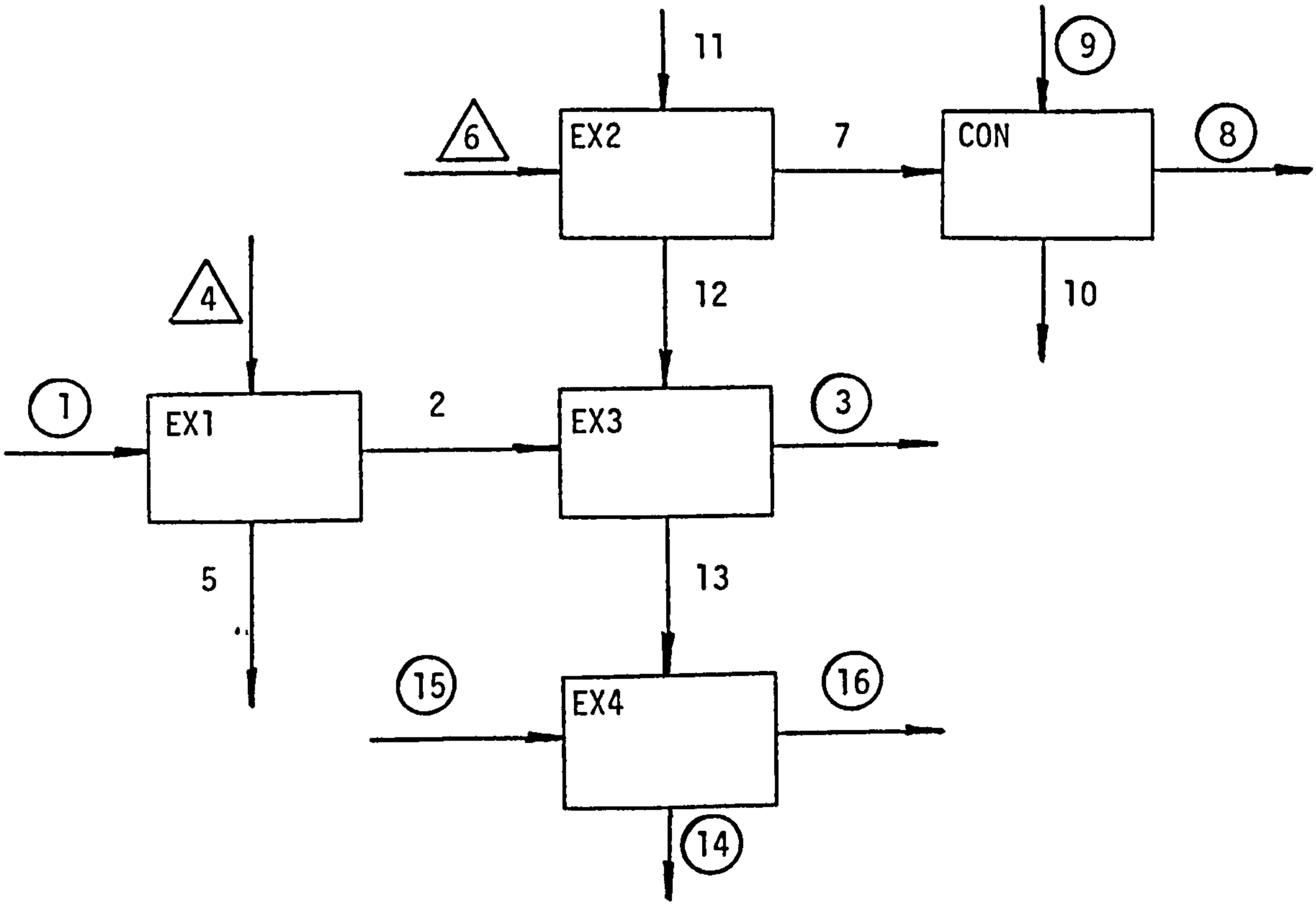


Fig 5.1: Flowsheet for HEX1 and HEX1M

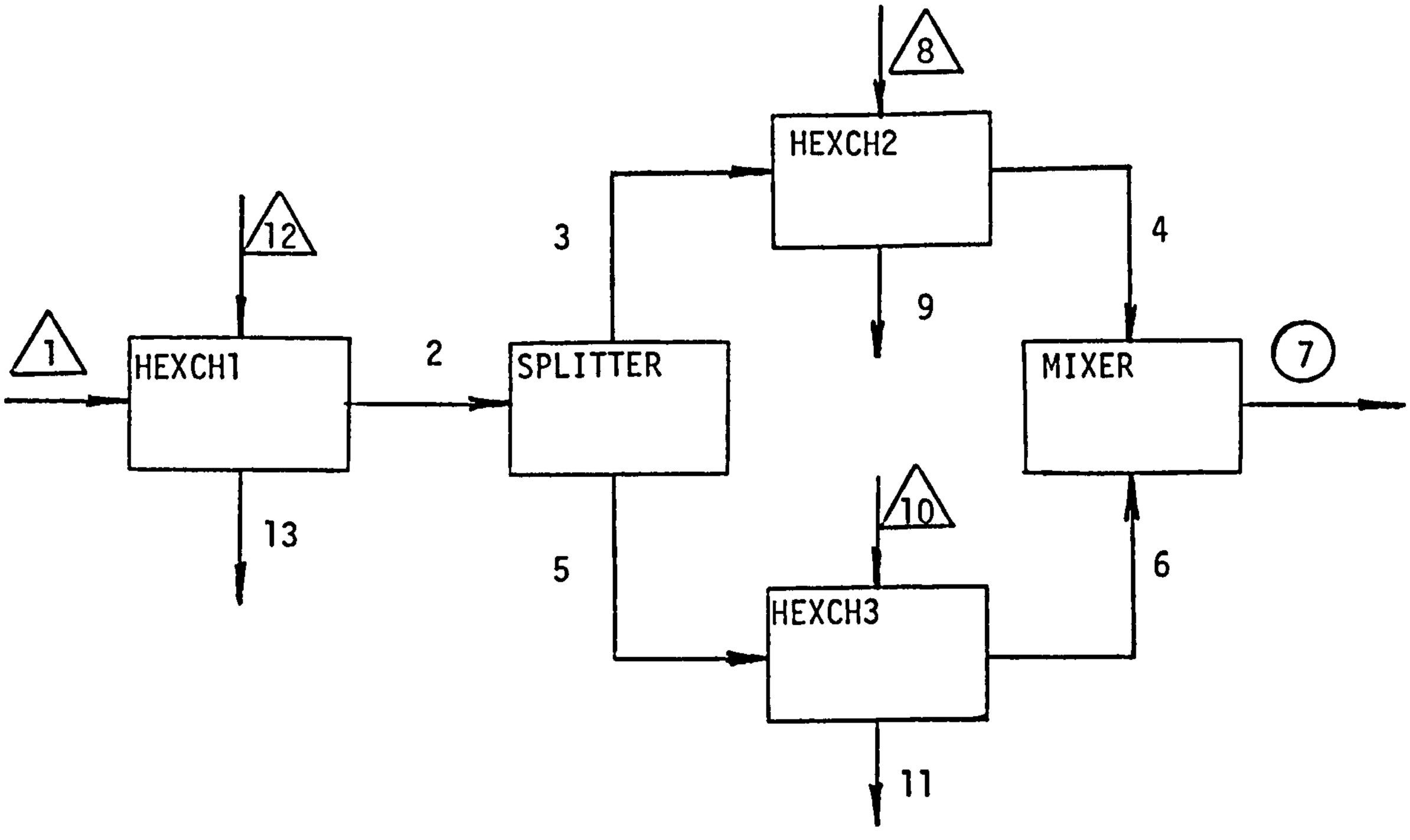


Fig 5.2: Flowsheet for HEX2 and HEX2M

Temperature specified

Temperature and flow specified

Heat transfer coefficients and area specified for all units

Consider the heat exchanger of Figure 5.1.

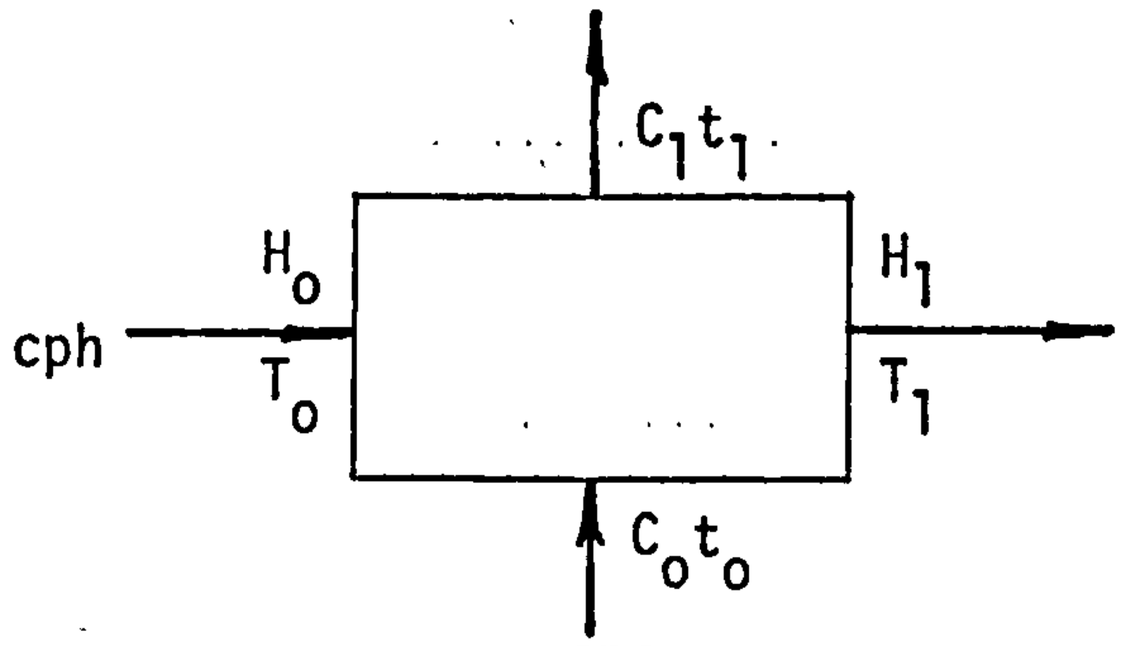


Figure 5.1: A heat exchanger unit

A traditional way of representing a heat exchanger with equations is:

$$C_1 = C_o \qquad 5.1$$

$$H_1 = H_o \qquad 5.2$$

$$Q = cph\ H_o\ (T_o - T_1) \qquad 5.3$$

$$Q = cpc\ C_o\ (t_1 - t_o) \qquad 5.4$$

$$Q = UA.\Delta TLM \qquad 5.5$$

$$ARG = (T_1 - t_o)/(T_o - t_1) \qquad 5.6$$

$$\Delta TLM = ((T_1 - T_o) - (T_o - t_1))/\ell n\ (ARG) \qquad 5.7$$

where C refers to the flowrate in the cold stream

H " " hot "

t " temperature cold "

T " " hot "

Q is the interchanged heat

cph and cpc are the specific heats of the hot and cold streams.

It is possible to obtain explicit expressions for T_1 and t_1 using the NTU approach (see KAYS and LONDON (1964)) as follows:

If we define

$$c_1 = cph\ H_o$$

$$c_2 = cpc\ C_o$$

$$c_3 = UA$$

$$c = c_2/c_1$$

from 5.3 and 5.4 we can see that

$$T_1 = T_o + c(t_o - t_1) \qquad 5.8$$

from 5.4, 5.5 and 5.7 it follows

$$c_2(t_1 - t_o) = c_3((T_1 - t_o) - (T_o - t_1))/\ell n\ (ARG)$$

and using 5.8 we can see that

$$\ell n\ (ARG) = \frac{c_3}{c_2}(1 - c)$$

And now defining

$$\alpha = \exp\ (c_3(1 - c)/c_2)$$

Using 5.6 and 5.8 we can obtain

$$t_1 = ((1 - \alpha)T_o - (1 - c)t_o)/(c - \alpha) \qquad 5.9$$

We can see then that we can replace equations 5.1 to 5.7 with a procedure which evaluates sequentially 5.9 and 5.8.

In SPEEDUP the procedure will be of the form

$$(T_1, t_1)SIMHEX(H_o, T_o, C_o, t_o, cph, cpc, UA) \qquad 5.10$$

The advantage of using the procedure is that if all the input data are known then the procedure returns immediately the solution while the equations 5.1 to 5.7 are, even in this case, a system of nonlinear equations which has to be solved iteratively.

Also, there are less numerical problems involved with equations 5.8 and 5.9 than there are with equation 5.7 which involves a logarithm.

We will consider both cases, ie, using only equations (5.1 to 5.7) for modelling the units (these problems will be called HEX1 and HEX2) and using equations and the procedure SIMHEX (which will be called HEX1M and HEX2M).

In HEX1 and HEX1M all specific heats are specified and some input flows and temperatures. The UA coefficient is given for all units as well as the latent heat of steam for the condenser.

For HEX2 and HEX2M all feeds are specified (flowrate, temperature and specific heat) but the splitting fraction is left unspecified. The output temperature for the cooling stream is specified instead.

The initial points used were as follows:

HEX1: All temperatures were initialized with 500 : 100 : 900 (initial guess, lower bound and upper bound). All flows with 1 : 0 : 1.E10. The exceptions are listed in Table 5.1.

HEX2: All temperatures were initialized with 170 : 80 : 250, the flows with 1 : 0 : 1.E10 with the exceptions listed in Table 5.2.

UNIT	VARIABLE	SOLUTION	INITIAL POINTS		
			1	2	3
EX3	Hot input temp	388	450	500	410
	Cold input temp	278	210	1.80	250
CON	Cold input temp	401	480	300	430
EX1	Cold input flow	2.05	3.00	3.50	2.60
EX2	Hot input flow	2.45	1.70	3.00	1.90
EX5	Cold input flow	2.54	1.20	2.50	1.50

Table 5.1: Solutions and initial points for HEX1

UNIT	VARIABLE	SOLUTION	INITIAL POINTS		
			1	2	3
SPL	Input temp	115	125	140	155
	Split factor	0.70	0.60	0.75	0.90
MIX	First input temp	195	200	180	160
	Second input temp	175	170	190	195

Table 5.2: Solutions and initial points for HEX2

HEX1M and HEX2M have the same initial point as HEX1 and HEX2 but without any exception, ie, all variables have the standard initial point.

For these problems we have developed in SPEEDUP a special initialisation procedure which we will call INIT.

We can describe it very briefly as:

- Of all the n variables initialize a subset of m variables.
- Eliminate from the original problem m equations involving the m initialized variables.
- Solve the reduced system of (n-m) equations (for our problems, it reduces to solving blocks containing only linear equations).

This procedure is equivalent to doing one pass of the flowsheet using a sequential modular approach and it can easily be implemented in SPEEDUP.

We will present results using this procedure for HEX1 and HEX2. After having used the procedure INIT we have all variables initialized as a function of the first n guessed variables . We can then proceed as usual but with a hopefully better initial point.

It was not necessary to use this procedure for HEX1M and HEX2M.

5.2.2 Cavett's process

This well known problem was originally proposed by CAVETT (1963). We have used the modified version of MALATHRONAS (1979).

The flowsheet is presented in figure 5.3.

We have used for this problem the physical properties facilities in SPEEDUP. We have considered two cases for the modelling of the isothermal flashes:

Case (a): Representing the flash by equations describing the mass balance plus an equilibrium equation using the K values and the vapour and liquid molar fractions, and a procedure for evaluating the K values given temperature (T), pressure (P) and liquid composition(BOTTOM).The procedure available in SPEEDUP is

(K_i) KVALU (T, P, BOTTOM)

Case (b): Representing the flash using only a procedure available in SPEEDUP as

(TOP, BOTTOM) SPHASE (T, P, FEED)

which given temperature, pressure and feed composition evaluates the vapour and liquid composition.

The procedure SPHASE contains internally an iteration loop which calls KVALU on each iteration. Case (a) avoids this inner iteration loop.

The problem was originally proposed as a simulation problem, ie, all temperatures and pressures are given for all flashes plus the feed flowrate. We will call this problem CAVSIM. In addition, we have also considered a design problem in which the temperature of the 4th flash is not given, specifying instead the product flowrate for the second component in stream 12. This problem was also considered by MALATHRONAS (1981) and will be called CAVDES.

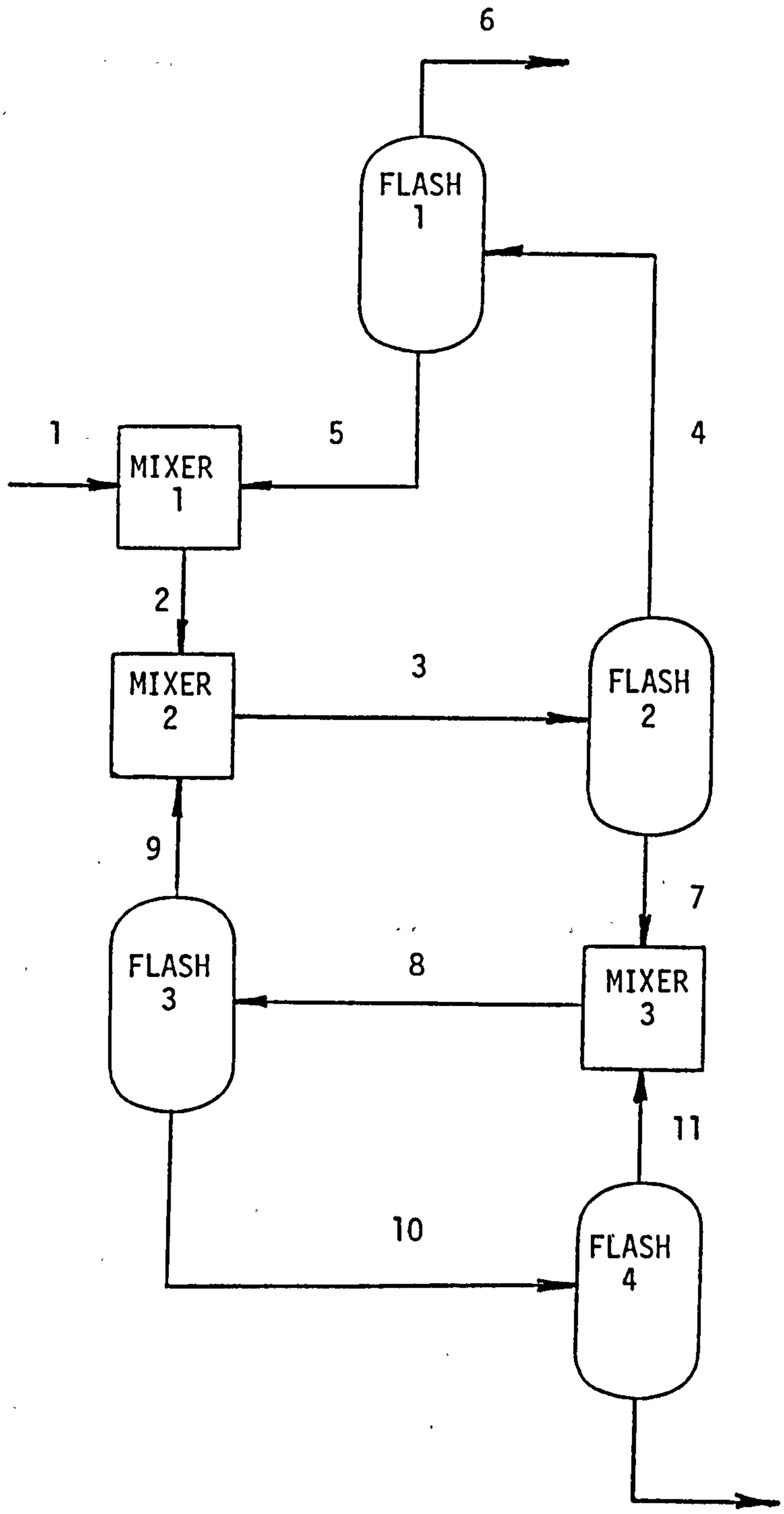


Figure 5.3: Flowsheet for CAVSIM and CAVDES

For both problems all flowrates are initialized to 1 : 0 : 10 with the exceptions listed in table 5.3 .

UNIT	VARIABLE	SOLUTION	INITIAL POINTS CAVSIM	CAVDES	
			1	1	2
FL1	Input flow	0.942	0.512*	0.512	0.512
	Five components	0.715	0.363	0.363	0.363
		0.211	0.121	0.121	0.121
		0.0262	0	0	0
		0.000286	0	0	0
FL4	Temperature	303	not appli-cable	290	340

Table 5.3: Solution and initial points for CAVSIM and CAVDES

* This initial point is obtained copying the first three components of the feed.

5.2.3 Benchmark problems from EFCE CHEMCOMP 82

These two problems were proposed by the organisers of the conference of the European Federation of Chemical Engineers in Antwerp (May 1982) in order to test different steady state simulation packages.

The first problem, CHEMCO1, is a process of separation of a hydrocarbon mixture. The flowsheet is presented in figure 5.4.

To implement this problem we have developed a special library of models which is presented in Appendix C. The isothermal flash is modelled using the procedure SPHASE (as it was for Cavett's process) of SPEEDUP while the adiabatic flash is modelled using the procedure SCALTP available in SPEEDUP as:

(TOP, BOTTOM) SCALTP (H, P, FEED)

which given enthalpy (H), pressure (P) and feed composition (FEED) evaluates the temperature (T), vapour (TOP) and liquid (BOTTOM) compositions.

Figure 5.4: Flowsheet for CHEMCOl

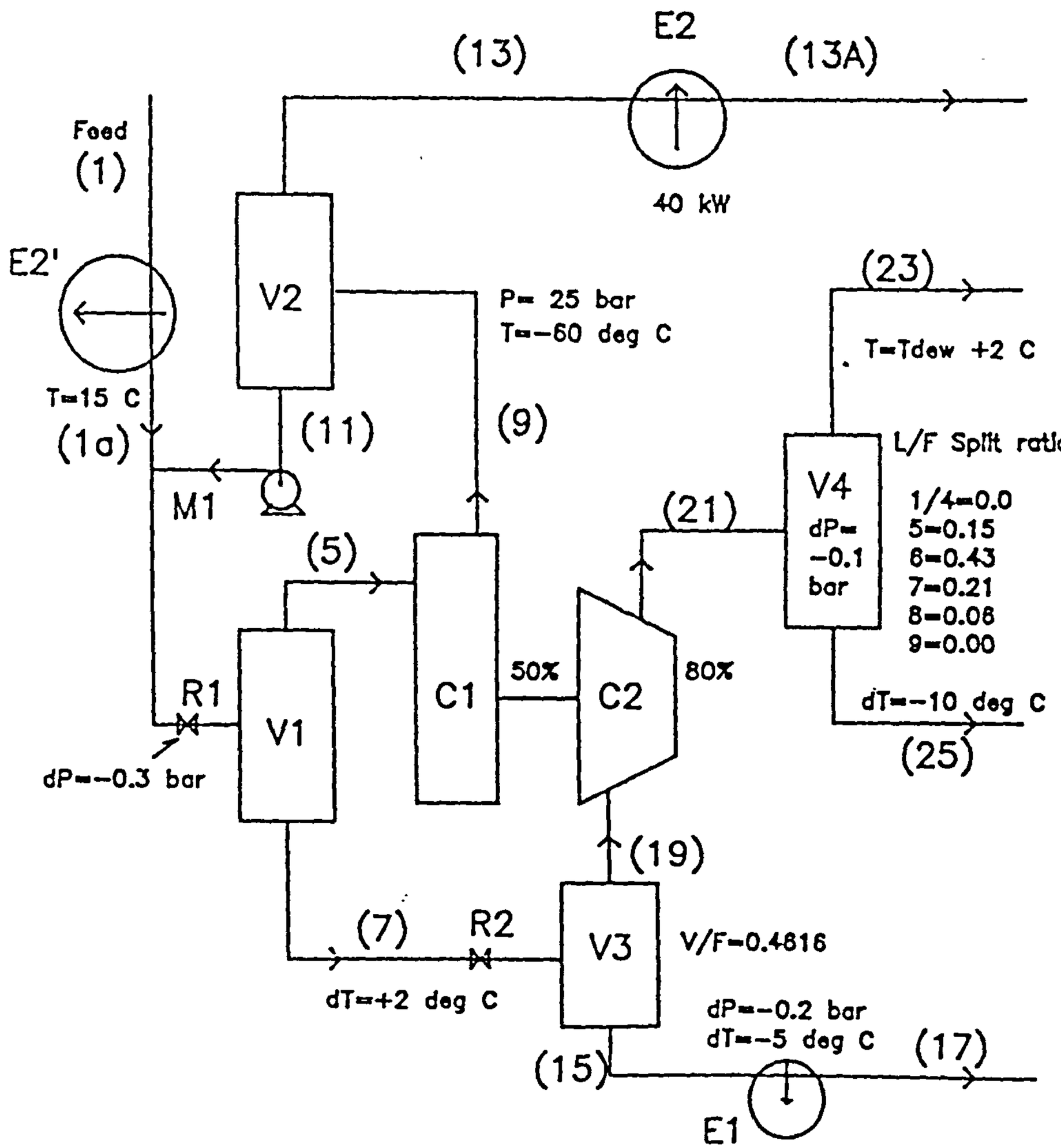
Data for this problem:

Feed: 3000 act m3/day (L+V)
T=40 C, P=75 bar
Composition in % WT

1=N2	=	9.0
2=methane	=	remainder
3=ethane	=	11.2
4=propane	=	6.2
5=butane	=	5.4
6=XXXX	=	3.0
7=hexane	=	8.1
8=heptane	=	13.3
9=octane	=	2.1

Comp 6=XXXX is pentane but must be loaded by user.
K-values by RKS or Chao-Seader.

V1=ad flash calculation with P
V2=isoth flash calc with given T,P
V3=adiab flash, V/F given
V4=black box
C1=expander + cooler/condenser
C2=compressor
E1,E2,E2'=heat exchangers



To model the compressor and the expander the following equations were used (taken from JONES and HAWKINS (1960)):

T	temperature
P	pressure
H	enthalpy
η	efficiency
γ	Cp/Cv
W	work

Subscripts: in input
out output
id ideal

For the expander:

$$\frac{T_{id}}{T_{in}} = \left(\frac{P_{out}}{P_{in}}\right)^{\frac{\gamma-1}{\gamma}}$$

$$\eta = \frac{(H_{in} - H_{out})}{(H_{in} - H_{id})}$$

For the compressor:

$$\eta = \frac{(H_{in} - H_{id})}{(H_{in} - H_{out})}$$

$$W = (H_{out} - H_{in})$$

For this problem the initial point used was the standard provided by SPEEDUP, which was:

flowrates	50 :	0 :	1000	kmol/hr
flowrates	5000 :	0 :	500000	kg/hr
temperatures	300 :	200 :	900	^{o}K
pressures	40 :	1 :	300	bars
fractions	0.5 :	0 :	1	

deltas	0 :	-1.E10 :	1E10	
enthalpies	10 :	-1E10 :	1E10	GJoules/hr

The second problem, CHEMCO2, is a process of production of methanol. The flowsheet is presented in figure 5.5.

Only mass balances are performed for this problem so no physical properties are needed. A temperature approach to equilibrium is used to specify the performance of the reactor.

The description of the reactor is as follows:

Reaction: $CO + 2H_2 \rightleftarrows CH_3OH$

$CO + H_2O \rightleftarrows CO_2 + H_2$

Reaction equilibria:

$$K_{p1} = \frac{P_{CH_3OH}}{P_{CO} (P_{H_2})^2}$$

$$K_{p2} = \frac{P_{CO} P_{H_2O}}{P_{CO_2} P_{H_2}}$$

with $P_{()}$ being the partial pressure of ().

Equilibrium constants:

$$K_{j1} = 0.6 + 1.5 \times 10^{-3} (T_1 - 473)$$

$$\log_{10} K^o_{p1} = 9.218 + \frac{3971}{T_1} - 7.492 \log_{10} T_1 + 1.77 \times 10^{-3} T_1 - 3.11 \times 10^{-8} T_1^2$$

$$K_{j2} = 0.89 + 4 \times 10^{-4} (T_2 - 473)$$

$$\log_{10} K^{o}_{p2} = 1664 - \frac{1850}{T_2}$$

where T_1 and T_2 are specified temperatures at which the products would be at equilibrium with respect to reactions (1) and (2).

$$K_{p1} = \frac{K^{o}_{p1}}{K_{j1}}$$

$$K_{p2} = \frac{K^{o}_{p2}}{K_{j2}}$$

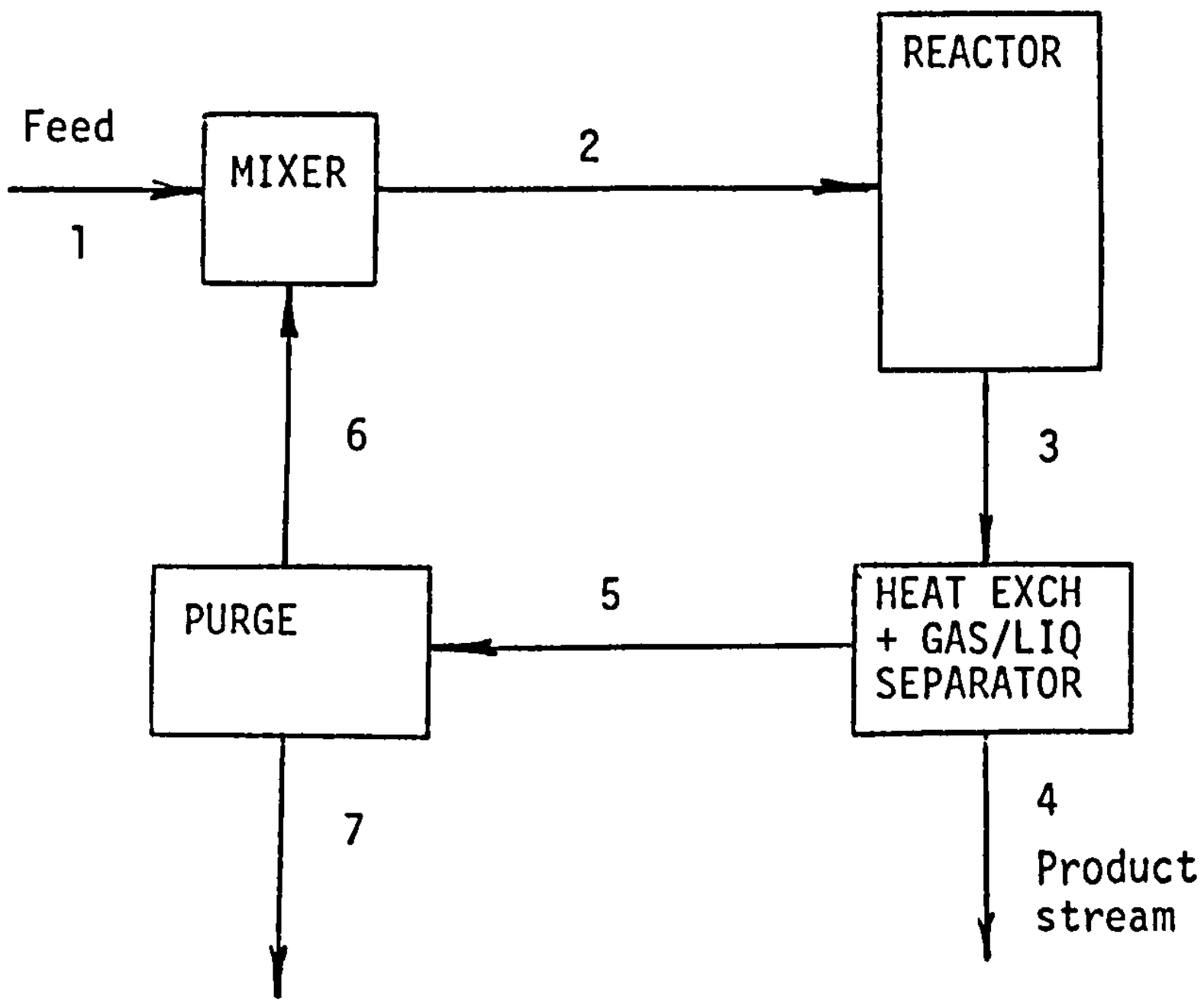


Figure 5.5: Flowsheet for CHEMCO2

The initial point for this problem is the one suggested by the proposers in which the composition of stream 2 is set to the same as stream 1. All the other flowrates are initialized to 10000 : 0 : 1000000 (Kmoles/hr).

5.2.4 Nitric acid plant

This problem was also taken from MALATHRONAS (1979).

The flowsheet for this problem is presented in figure 5.6.

It consists of the simulation of a plant producing nitric acid and only mass balances are performed (hence no physical properties are needed). After decomposition we are left with a block of 199 variables, being the largest problem we have solved (regarding storage requirements). We have used the library of standard models performing mass balances available in SPEEDUP.

The initial point used was the standard provided by SPEEDUP for all flowrates (10 : 0 : 1000).

We will refer to this problem as NITRIC.

5.2.5 Crude preheat train

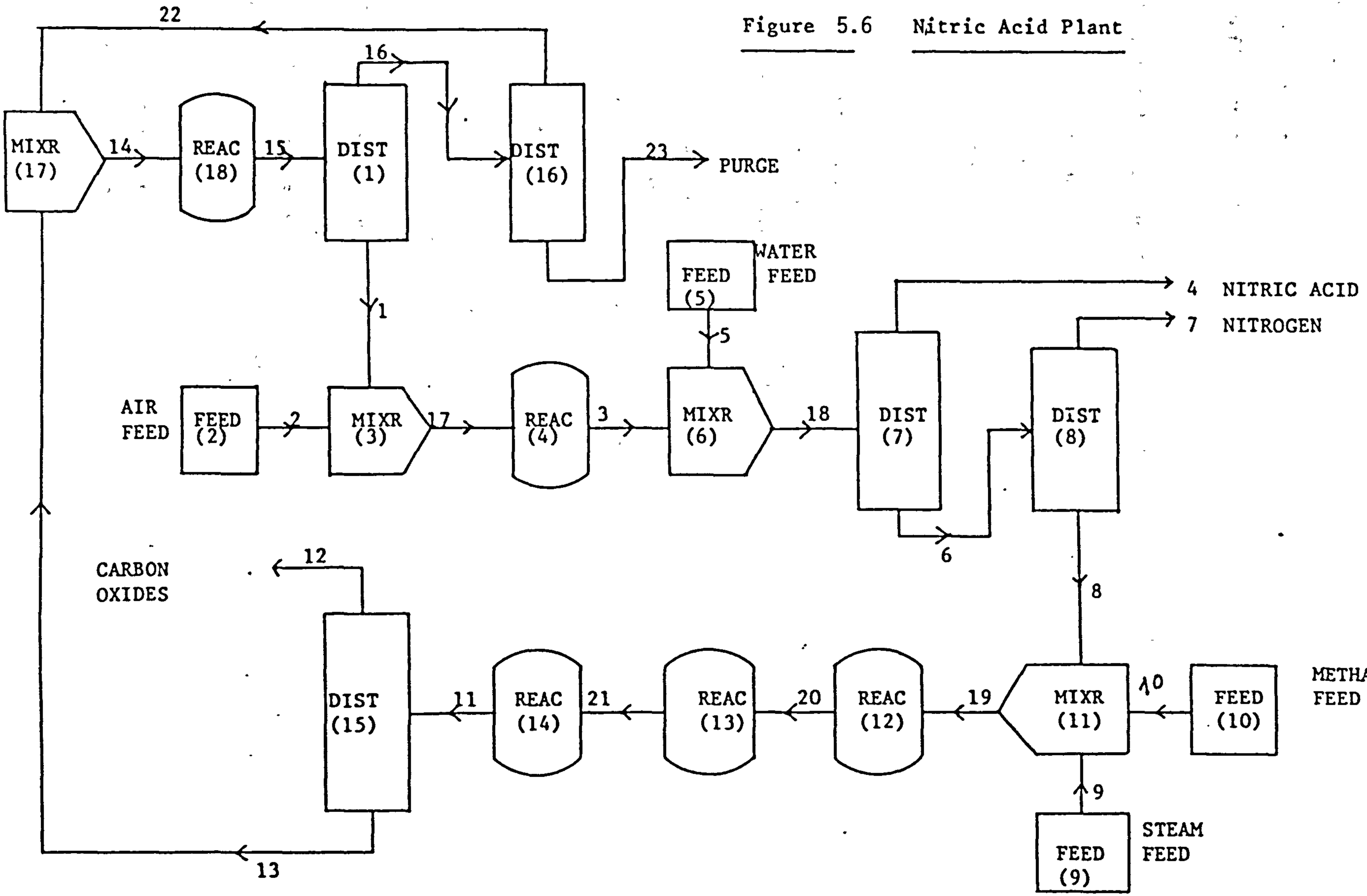
This problem was taken from WONG (1981).

It consists of a complicated network involving 20 heat exchangers and 12 streams. The flowsheet is presented in figure 5.7a.

The original problem was a simulation problem and we have modified it such that all feed flowrates, specific heats and temperatures are given as well as some measured temperatures. With this information, heat transfer coefficients are calculated for all units as well as intermediate temperatures and flows.

We will use, for this problem, a model of heat exchangers using only equations and also one using the procedure SIMHEX as we did for the heat exchangers of section 5.2.1. We will call the problems respectively BPNET and BPNETM.

Figure 5.6 Nitric Acid Plant



The initial point used in these cases was the standard provided by SPEEDUP which was:

flowrates	100 :	0 :	1000	(Kmoles)
temperatures	200 :	-100 :	500	(^{o}C)
coefficients (UA)	100 :	0 :	1000	

Only for BPNET, the following temperatures are initialized:

STREAM	SOLUTION	INITIAL POINT
51	306	300
52	215.3	250
61	219.5	250

This problem has been very interesting from the point of view of the decomposition phase of SPEEDUP. We have marked in the flowsheet for this problem (see Figure 5.7b) the two largest blocks found. By slightly altering the specifications we can obtain a very different decomposition. For instance, if instead of specifying stream 36, on the heat exchanger number 5, we specify stream 61 on the same unit, the decomposition will give us a block consisting only of the units which are connected to the flowsheet by stream 33 (see Figure 5.7b).

5.2.6 Simple ethylene plant

This problem has been taken from PERKINS (1979).

The flowsheet is presented in Figure 5.8. Only mass balances are performed and, as we did for the nitric acid plant, the standard library of models performing mass balances was used.

We have considered a simulation problem (all feeds and equipment parameters known) and 3 design problems:

HEAT EXCHANGER NETWORK (Crude Preheat Train)

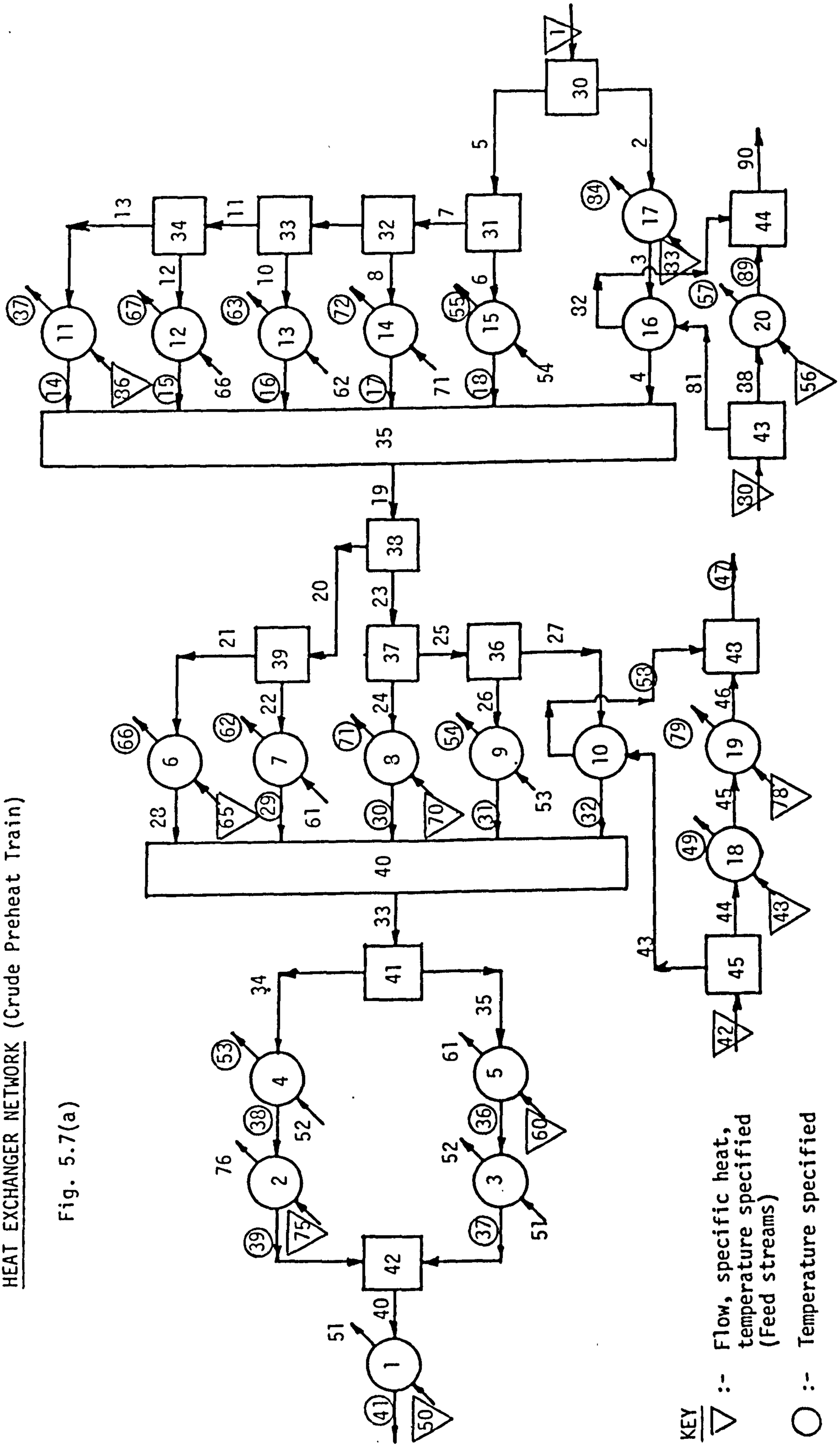
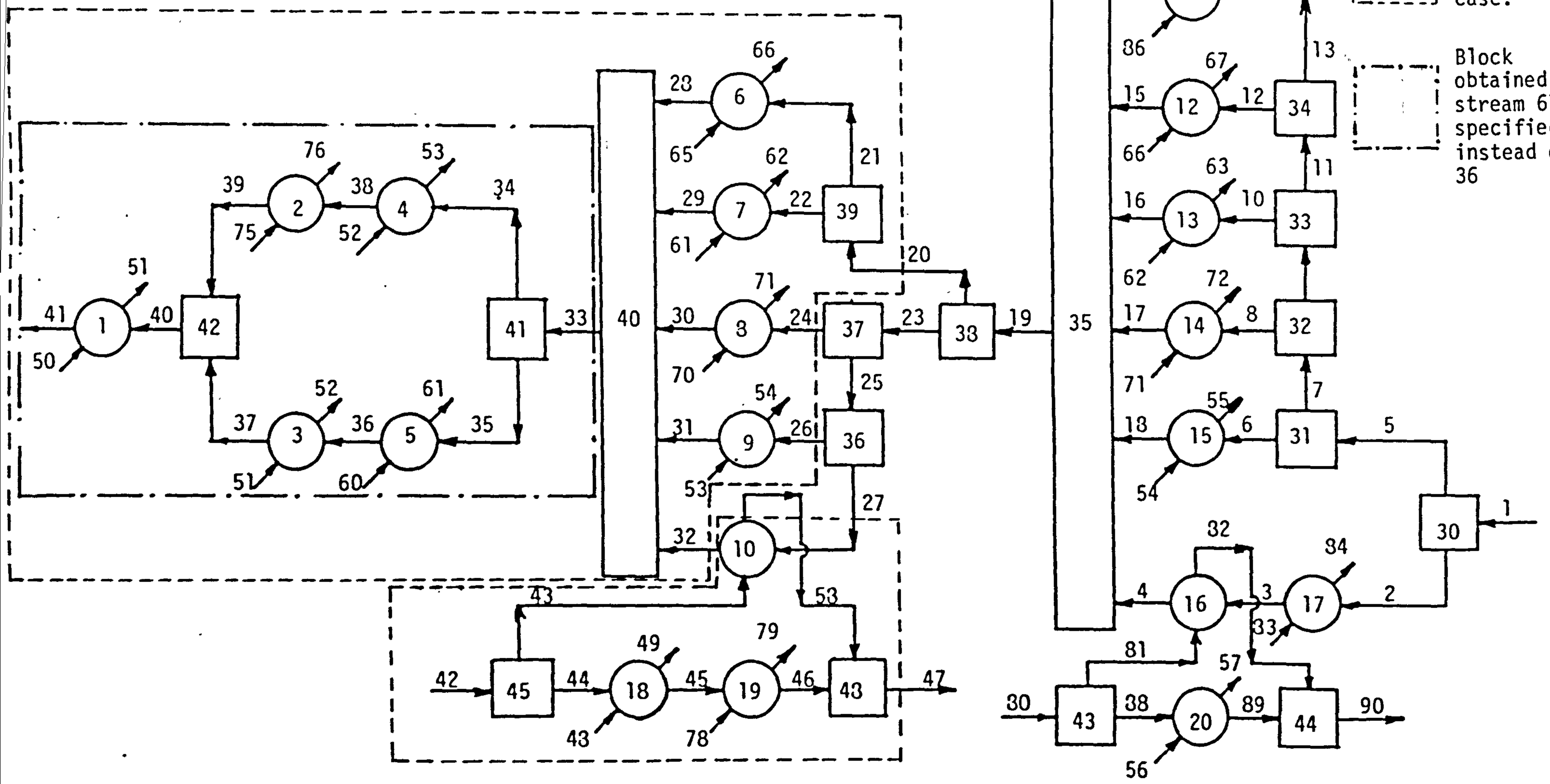


Fig. 5.7(a)

HEAT EXCHANGER NETWORK (Crude Preheat Train)

Fig. 5.7(b)



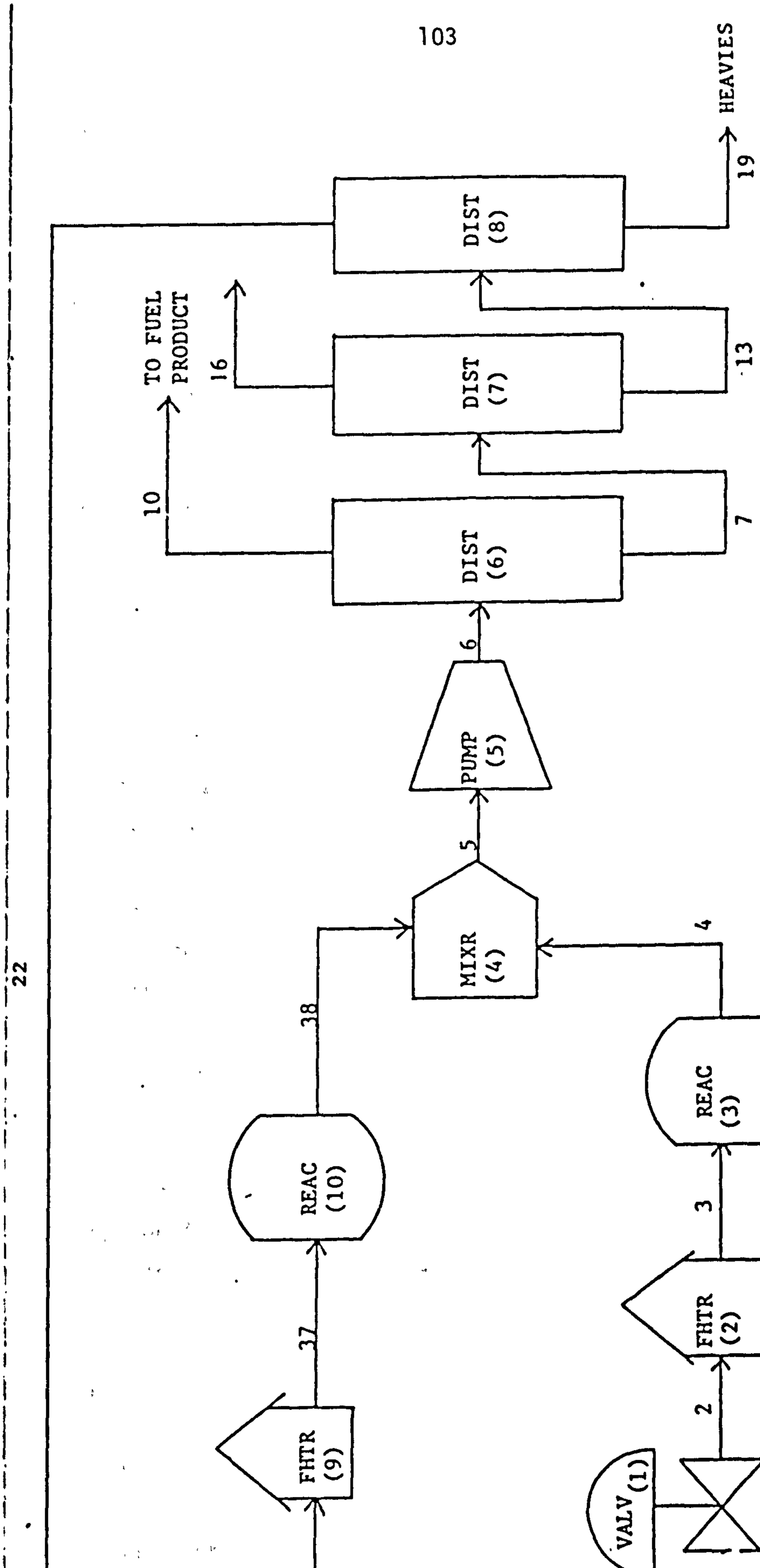


Figure 5.8 Simple Ethylene Plant

CASE	DESIGN VARIABLES	DESIGN SPECIFICATIONS
1	Conversion in REAC10	Flow of ethylene in stream 16
2	Conversion in REAC10	Flow of ethylene in stream 16
	" " REAC3	" propylene " 19
3	Conversion in REAC10	Flow of ethylene in stream 16
	" " REAC3	" methane " 10
	TOTAL FEED FLOW	Ratio of propane/butane in stream 19

We will name the simulation problem SIMETH and the design problems DESETH.

A standard initial point of 1 : 0 : 100 was used for all flowrates.

5.3 NUMERICAL RESULTS

We will present in this section the numerical results obtained for the problems mentioned in section 5.2.

Four methods will be considered, our implementation of Broyden's method mentioned in chapter 2 (BRO) and three of the scale invariant methods mentioned in chapter 3 (SI2, SI3 and SI4).

We will give for each problem the number of function evaluations used by the method on the biggest block.

The reason for this is because whenever a failure occurred, it happened in the biggest block for all our problems as well as because regarding efficiency all methods perform equally in the smaller blocks.

Turning first to the results for the small heat exchanger networks, we should note that the use of equations for implementing the heat exchanger is not numerically safe because of the logarithm in equation 5.7. If the equation is implemented in this way we risk having a zero denominator. We have used instead

$$\Delta TLM \quad \ell n(ARG) = (T_1 - t_o) - (T_o - t_1)$$

The problem when using this equation is that it creates trivial solutions (which are not solutions to the original problem). This trivial solution is

$$ARG = 1$$

$$T_1 - t_o = T_o - t_1$$

All cases for HEX1 converge to trivial solutions in at least one heat exchanger and the results are presented in table 5.4. In table 5.6 we present the results for HEX2; in this case no trivial solutions were found. The use of the procedure avoids this problem of trivial solutions and the results for these cases are presented in tables 5.5 and 5.7.

INIT POINT	INTERNAL SCALING	INIT FACILITY	METHODS BRO	SI2	SI3	SI4
1	NO	NO	**	**	**	**
1	YES	NO	**	**	**	**
1	NO	YES	63	52	83	46
1	YES	YES	42	52	93	65
2	NO	YES	54	55	124	58
2	YES	YES	61	55	69	78
3	NO	YES	31	32	69	32
3	YES	YES	40	32	69	40

Table 5.4: Function evaluations for the biggest block for HEX1
** failed to converge

INTERNAL SCALING	METHODS BRO	SI2	SI3	SI4
NO	36	28	59	28
YES	79	28	59	28

Table 5.5. Function evaluations for the biggest block for HEX1M

INIT POINT	INTERNAL SCALING	INIT FACILITY	METHODS BRO	SI2	SI3	SI4
1	NO	NO	**	**	**	**
1	YES	NO	**	**	**	**
1	NO	YES	91	29	29	34
1	YES	YES	34	29	44	109
2	NO	YES	86	117	92	75
2	YES	YES	**	159	93	213
3	NO	YES	**	**	**	**
3	YES	YES	88	81	**	285

Table 5.6: Function evaluations for the biggest block for HEX2

INTERNAL SCALING	INIT FACILITY	METHODS BRO	SI2	SI3	SI4
NO	NO	71	105	83	78
YES	NO	70	42	134	152
NO	YES	23	26	62	21
YES	YES	26	26	54	34

Table 5.7: Function evaluations for the biggest block for HEX2M

These problems show clearly:

- The use of a procedure for implementing the heat exchangers is preferable to the use of equations. It allows the use of standard initial guesses in SPEEDUP.

- The INIT facility for obtaining an initial point is essential for this problem represented solely as equations. A solution can not be obtained without it.

- The use of the internal scaling, while it seems to cause the efficiency to deteriorate in some cases, helps robustness.

All methods failed to solve the Cavett's process problem for case (a) ie, when equations are used (plus the procedure KVALU). For

case (b) most methods did very well and the results are presented in table 5.8. The methods BRO and SI2 performed best while SI3 and SI4 failed for some design cases.

PROBLEM	INITIAL TEMP	SCALING	METHODS BRO	SI2	SI3	SI4
CAVSIM	-	NO	66	64	98	67
	-	YES	100	81	101	107
CAVDES	290	NO	77	97	**	65
	290	YES	85	108	**	**
	340	NO	86	82	**	**
	340	YES	90	195	**	105

Table 5.8: Function evaluations for the biggest block for CAVSIM and CAVDES

Our results are consistent with those obtained by HILTON and STADTHERR (1981) when considering Cases (a) and (b) with the equation oriented package SEQUEL. They also found case (b) to perform better than case (a).

Table 5.9 shows the results obtained with the benchmark problems of EFCE CHEMCOMP 82. All codes succeeded in finding the solution and the use of the internal scaling does not alter the results much.

PROBLEM	INTERNAL SCALING	METHODS BRO	SI2	SI3	SI4
CHEMC01	NO	39	40	77	67
	YES	41	40	78	75
CHEMC02	NO	74	105	110	109
	YES	85	99	145	87

Table 5.9 : Function evaluations for the biggest block for the benchmark problems of EFCE CHEMCOMP 82

We present in table 5.10, the results for the crude preheat train.

PROBLEM	INTERNAL SCALING	METHOD			
		BRO	SI2	SI3	SI4
BPNET	NO	36	30	57	31
	YES	33	30	46	30
BPNETM	NO	61	69	**	155
	YES	94	69	**	212

Table 5.10: Function evaluations for the biggest block for the crude preheat train

Contrary to the results obtained for the Cavett's process and HEX1 and HEX2, for this problem the use of only equations for modelling the heat exchanger gives better results than using the procedure. All methods converge very fast and with the standard initial point.

The reason for this is that because most temperatures are known, the argument of the logarithm in the equations is known eliminating the main disadvantages of the use of equations. This also avoids the problem of trivial solutions which we found for HEX1.

Again SI3 fails for BPNETM while BRO and SI2 perform best for all cases.

Due to the internal decomposition in SPEEDUP, the problems NITRIC and all the cases for the simple ethylene plant are such that they are decomposed into only linear blocks (which our code solves in one iteration). This is due to the fact that the nonlinearity is only caused by products of two variables (ie, the function is bilinear), which are solved for in separate blocks. Thus they become linear blocks.

We do not present any results for these problems since all methods solve them in one iteration.

We summarize in table 5.11 the percentages of success for all methods on the 18 problems considered.

INTERNAL SCALING	BRO	SI2	SI3	SI4
NO	83	83	67	78
YES	83	89	67	83
TOTAL	83	86	67	81

Table 5.11: Percentage of success for all problems

The methods BRO and SI2 are those which had the best performance for the Chemical Engineering problems. Globally, the internal scaling solves more problems.

We present in Table 5.12 the sizes of the problems solved.

PROBLEM	TOTAL NUMBER OF VARIABLES	NUMBER OF BLOCKS	IN BIGGEST BLOCK VARIABLES	IN BIGGEST BLOCK NONLINEAR EQUATIONS
HEX1	36	14	20	14
HEX1M	21	11	11	8
HEX2	27	9	15	9
HEX2M	18	9	9	5
CAVSIM	55	1	55	40
CAVDES	55	1	55	40
CHEMCO1	251	157	57	38
CHEMCO2	99	48	52	18
NITRIC	425	227	199	0
BPNETM	207	161	26	14
BPNET	267	236	25	13
SIMETH	65	22	8	0
DESETH1	65	28	7	0
DESETH2	65	26	7	0
DESETH3	87	36	33	0

Table 5.12 Size of the Chemical Engineering problems solved

5.4 CONCLUSIONS

We have available a code implementing our proposed methods whose behaviour on chemical engineering flowsheet problems is very good when used in conjunction with the simulation package SPEEDUP.

Our conclusion is based on the fact that we have tried a wide variety of problems, all considered before by different authors (two of them being benchmark problems) and the initial points used were, in general, the standard provided by the simulation package (and the exceptions are a very small proportion of the total number of variables being initialized).

It seems desirable to incorporate in SPEEDUP an initialization such as we have used for the problems HEX1 and HEX2. It is clear that it will help robustness as well as efficiency.

The importance of the exploitation of the structure of the problem is demonstrated by our results. Two problems (ETHYL and NITRIC) which are nonlinear became linear due to the decomposition. The problem BPNET shows how useful, for reducing the size of the problem, the decomposition is. The decomposition into smaller blocks also helps the debugging of a problem, helping to locate the source of difficulties (such as badly posed problems or getting better initial guesses). The problems HEX1M and HEX2M also show how exploiting the structure (in this case by having explicit expressions for some variables) helps to overcome very difficult problems.

It is unfortunate that avoiding internal loops in the physical properties calculations by linking them to the nonlinear system gives such bad results. The use of special procedures for the physical properties seems to be essential.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

This thesis has been concerned with the study of the problem of numerical solution of systems of algebraic nonlinear equations in Chemical Engineering.

Our conclusions from this study follow.

Our implementation of the method of BROYDEN (1965) (Chapter 2) is a considerable improvement over the codes published so far.

We have presented Quasi-Newton scale invariant methods which have shown numerical results which puts them as an excellent alternative to our implementation of Broyden's method, with the additional advantage of being theoretically scale invariant (Chapter 3). We have proposed only 4 particular scale invariant methods but we have shown that many more could be devised. Our ideas for scale invariance could be extended for Quasi-Newton methods generated by updates other than rank-one.

We have proposed and tested with good numerical results an internal scaling procedure which relies on a reasonable theoretical justification. The use of this procedure can be extended to any code for solving non-linear algebraic equations for which an approximation to the Jacobian is used.

Our codes have shown particularly good numerical results when used for solving steady-state flowsheeting problems. A wide set of examples have been used for the testing and the initial points used were in general well away from the solution. This shows clearly the robustness of our codes.

We have confirmed, with our results, previous findings by other authors regarding the deterioration of the numerical performance of codes when the physical properties inner iterations are considered together with the main iteration for solving the non-linear system.

Since there is a great advantage involved in avoiding inner loops for physical properties, it seems to us that special attention should be given to this.

We believe that our initialisation procedure for obtaining better initial points (section 5.2.1) in SPEEDUP should be incorporated as a standard feature. Our examples have shown that much can be gained in robustness when it us used.

Finally, our experience shows that the combination of our codes and the simulation package SPEEDUP provides a very useful tool for performing steady-state flowsheeting problems in Chemical Engineering.

REFERENCES

ASPEN PROJECT (1980), "Aspen User Manual", MIT Cambridge, MA.

BARNES J G P (1965), "An algorithm for solving nonlinear equations based on the secant methods", The Computer Journal, 8, pp 66-72.

BAUER, F L (1963), "Optimally scaled matrices", Num Math, 5, pp 73-87.

BENNETT J M (1965), "Triangular factors of modified matrices", Num Math, 7, pp 217-221.

BOGLE I D L (1979), "Numerical methods for nonideal flowsheeting problems, MScDiss, Imperial College.

BRENT R P (1973), "Some efficient algorithms for solving systems of nonlinear equations", SIAM J on Num Anal, 10, No 2.

BROYDEN C G (1965), "A class of methods for solving non-linear simultaneous equations", Math of Comp, 19, p 577.

BROYDEN C G (1967), "Quasi-Newton methods and their application to function minimisation", Math of Comp, 21, p 368.

BROYDEN C G (1969), "A new method of solving nonlinear simultaneous equations", Comp Journal, 12.

BROYDEN C G, DENNIS J E, MORE J J (1973), "On the local and superlinear convergence of Quasi-Newton methods", J of the Inst of Math and Appl, 12, p 223.

BROWN K M (1966), "A quadratically convergent method for solving simultaneous non-linear equations", PhDDiss, Purdue Univ.

BUS J C P (1975), "A comparative study of programs for solving non-linear equations", Mathematisch Centrum Report NW 25/75, Amsterdam, Holland.

CAVETT R H (1963), "Application of numerical methods to the convergence of simulated processes involving recycle loops", American Pet Inst, Preprint No 04-63.

CHEN H S, STADTHERR M A (1981), "A modification of Powell's dogleg method for solving systems of non-linear equations", Comp and Chem Eng, 5, No 3.

COLEMAN T F, MORE J J (1981), "Estimation of sparse Jacobian matrices and graph coloring problems", Argonne Nat Lab Report, ANL-81-39.

COSNARD M Y (1975), "A comparison of four methods for solving systems of non-linear equations", Report TR75-248, Cornell Univ, Dept Comp Science.

CROWE CM, NISHIO M (1975), "Convergence promotion in the simulation of chemical processes - the general dominant eigenvalue method", AIChE J, 21, No 3.

CURTIS A R, POWELL M J D, REID J K (1974), "On the estimation of sparse Jacobian matrices", J Inst Math Appl, 13, p 117.

DAVIDON W C (1975), "Optimally conditioned optimization algorithms without line searches", Math Progr, 9, pp 1-30.

DENNIS J E, MORE J J (1974), "A characterization of superlinear convergence and its application to Quasi-Newton methods", Math Comp, 28, p 549.

DENNIS J E, MORE J J (1977), "Quasi-Newton methods, motivation and theory", SIAM Review, 19, No 1.

DENNIS J E, SCHNABEL R B (1978), "Least change secant updates for Quasi-Newton methods", Report TR78-344, Cornell Univ, Dept Comp Science.

EDWARDS D W (1982), "Robust decomposition techniques for process design and simulation", PhD Thesis, Univ London.

FLETCHER R (1970), "A new approach to variable metric algorithms", The Comp Journal, 13, No 3.

GAY D M, SCHNABEL R B (1978), "Solving systems of non-linear equations by Broyden's method with projected updates", From: Nonlinear Prog 3, Ed, Mangasarian, O L and others, Academic Press.

GILL P E, MURRAY W (1972), "Quasi-Newton methods for unconstrained optimisation", J Inst Math Applic, 9, pp 91-108.

GORCZYNSKI E W, HUTCHISON H P (1978), "Towards a quasilinear process simulator - I: fundamental ideas", Comput Chem Eng, 2, p 189.

GRAGG W B, STEWART G W (1974), "A stable variant of the secant method for solving nonlinear equations", Carnegie Mellon Univ Report.

GROSSMANN I (1975), "Optimum design of heat exchanger networks", MSc Thesis, Univ London.

GROSSMANN I, del ROSAL R (1978), "Comparacion de methodos de convergencia en el calculo de reciclos", XVIII Conv. Nac del IMIQ Merida.

HIEBERT K L (1980), "A comparison of software which solves systems of nonlinear equations", Sandia Lab Report, SAND 80-0181.

HILTON C, STADTHERR M A (1981), "Development of a new equation based process flowsheeting system", Paper at 75th AIChE meeting, New Orleans.

IMSL (1979), "Library Reference Manual", Ed 7.

JOHNS W R (1970), "Mathematical considerations in preparing general purpose computer programs for the design or simulation of chemical processes", Paper at EFCE Conf "The use of computers in studies preceding the design of chemical plants", Florence, Italy.

JOHNS W R, SHEPPARD A J (1975), "Multi-component multi-stage absorption and distillation calculations for use with flowsheeting programs", Symp, Computers in the design and erection of chemical plants, Karlovy Vary, Czechoslovakia.

JONES J B, HAWKINS G A (1960), "Engineering thermodynamics", John Wiley.

KAYS W M, LONDON A L (1964), "Compact heat exchangers", McGraw-Hill.

KUBICEK M, HLAVACEK V, PROCHASKA, F (1976), "Global modular Newton-Raphson technique for simulation of an interconnected plant applied to complex rectification columns", Chem Eng Sci, 31, p 277.

LEE N (1980), "Simultaneous solution of design, torn streams and multi-component distillation equations in a sequential modular flowsheeting package", MSc Thesis, Univ of London.

MAHALEC V, KLUZIK H, EVANS B L (1979), "Simultaneous modular algorithm for steady state flowsheet simulation and design", CACE 79, EFCE, Montreux.

MALATHRONAS J P (1979), "Solution of simulation and design problems using Broyden's method in sequential modular flowsheeting packages", MScDiss, Imperial College.

MALATHRONAS J P, PERKINS J D (1980), "Solution of design problems using Broyden's method in a sequential modular flowsheeting package", Paper at CHEM PLANT 80, Heviz, Hungary.

METCALFE S R, PERKINS J D (1978), "Information flow in modular flowsheeting systems", Trans IChemE, 56, p 210.

MINPACK (1980), "Documentation", Argonne Nat Lab, App Math Div.

MORE J J, COSNARD M Y (1979), "Numerical solution of nonlinear equations", ACM TOMS, 5, No 1.

MORE J J, GARBOW B S, HILLSTROM K E (1980), "User guide for Minpack-1", Report ANL-80-74, Argonne Nat Lab.

MOTARD R L, LEE H M (1971), "CHESS User's Guide", 3rd Ed, Dept of Chem Eng, Univ of Houston.

MOTARD R L, SHACHAM M, ROSEN E M (1975), "Steady-state chemical process simulation", AIChE J, 21, p 417.

ORBACH O, CROWE C M (1971), "Convergence promotion in the simulation of chemical processes with recycle - the dominant eigenvalue method", The Canadian J of Chem Eng, 49.

ORTEGA J M, RHEINBOLDT W C (1970), "Iterative solution of nonlinear equations in several variables", Academic Press.

PALOSCHI J R (1980), "A comparative study of algorithms for solving systems of nonlinear equations", Imperial College Report.

PERKINS J D (1979a) Personal communication.

PERKINS J D (1979b) "Efficient solution of design problems using a sequential modular flowsheeting programme", Paper at 12th Symp on Comp Appl in Chem Eng, Montreux.

POWELL M J D (1970), "A hybrid method for nonlinear equations", In "Numerical methods for nonlinear algebraic equations", P Rabinowitz, ed, Gordon and Breach.

RHEINBOLDT W (1974), "On measures of ill conditioning for nonlinear equations", Tech Report TR330, Univ of Maryland, Comp Sci Center.

ROSEN E M (1962), "A machine computation method for performing material balances", Chem Eng Progress, 58, No 10.

ROSEN E M (1966), "A review of Quasi-Newton methods for nonlinear equation solving and unconstrained optimization", Proc 21st ACM Nat Meet, Washington.

SAIGAL R (1979), "On solving large structural fixed point problems", Paper at Symp on fixed point alg and complementarity, Southampton Univ.

SALGOVIC A, HLAVACEK V, ILAVSKY J (1981), "Global simulation of counter current separation processes via one-parameter imbedding techniques", Chem Eng Sci, 36, No 10

SANDIA (1975), "Mathematical Subroutine Library", Sandia Lab.

SARGENT R W H (1977), "The decomposition of systems and procedures and algebraic equations", Paper at Dundee Conf on NA.

SARGENT R W H, PERKINS J D, THOMAS S (1982), "SPEEDUP: a computer program for steady-state and dynamic simulation of chemical processes", Paper at IChemE Jubilee Symp.

SCHUBERT L K (1970), "Modification of Quasi-Newton method for nonlinear equations with a sparse Jacobian", Math of Comp, 23, p 27.

SHERMAN J, MORRISON W J (1949), "Adjustment of an inverse matrix corresponding to changes in the elements of a given column as a given row of the original matrix", Ann Math Statist, 20, p 621.

TODD M J (1966), "On condition numbers", Paper at Prog on Math Num, Besancon.

TODD M J (1976), "The computation of fixed points and applications", Springer-Verlag, New York.

VANEK T, HLAVACEK V, KUBICEK M (1977), "Calculation of separation columns by nonlinear block successive relaxation methods", Chem Eng Sci, 32, p 839.

WEGSTEIN J (1958), "Accelerating convergence of iteration processes", Comm ACM, 1, p 9.

WONG M P F (1981), "Efficient solution of flowsheets with heat exchangers using Broyden's method in a sequential modular package", MSc Thesis, Univ of London.

APPENDIX A

THE SETS OF MATHEMATICAL EXAMPLES

We describe here the sets of mathematical examples used for the comparisons. All of them are taken from HIEBERT (1980).

The basic set of examples consist of 18 different "mathematical" problems, each one having a standard initial point (with the exception of the last three having more than one initial point).

The list of problems can be found in Table A1 and they are collected in the MINPACK (1980) test routines VECFEC and INITP listed in Appendix D.

Problems F to N are of variable dimension.

From the basic set of examples four different sets of problems will be configured as follows.

GENERAL SET: Consists of 54 problems taken from the basic set, 21 having as initial point x_o (the standard initial point), 18 with $10x_o$, and 15 with $100x_o$. In table A2 we summarize the problems for this set.

GENERAL SUBSET: This is a subset of the general set. If consists of 16 problems, all having the standard initial point and it is listed in table A3. All the problems for this set were selected such that all methods considered converged for the standard initial point.

CHEMICAL EQUILIBRIUM SET: Consists of 12 problems taken from O to Q from the basic set and taking different initial points. It is listed in table A4.

RESTRICTED SET: This is just the general set where all problems with $n \leq 2$ were removed. It consists of 46 problems.

PROBLEM	DESCRIPTION	DIMENSION
A	Rosenbrock's function	2
B	Powell's singular function	4
C	Powell's badly scaled function	2
D	Wood's function	4
E	Helical valley function	3
F	Watson's function	variable
G	Chebyquad function	"
H	Brown's almost linear function	"
I	Discrete boundary value problem	"
J	Discrete integral equation function	"
K	Trigonometric function	"
L	Variable dimensioned function	"
M	Broyden's tridiagonal function	"
N	Broyden's banded function	"
O	Chemical equilibrium problem 1	2
P	" " " 2	6
Q	" " " 3	10

Table A1: List of problems and dimensions

Prob	Orig Prob	Dimen	Prob	Orig Prob	Dimen	Prob	Orig Prob	Dimen
1	A	2	19	L	10	37	L	10
2	B	2	20	M	10	38	M	10
3	C	2	21	N	10	39	N	10
4	D	4	22	A	2	40	A	2
5	E	3	23	B	4	41	B	4
6	F	6	24	C	2	42	D	4
7	F	9	25	D	4	43	E	3
8	G	5	26	E	3	44	G	5
9	G	6	27	F	6	45	G	6
10	G	7	28	F	9	46	G	7
11	G	9	29	G	5	47	H	10
12	H	10	30	G	6	48	I	10
13	H	30	31	G	7	49	J	2
14	H	40	32	H	10	50	J	10
15	I	10	33	I	10	51	K	10
16	J	2	34	J	2	52	L	10
17	J	10	35	J	10	53	M	10
18	K	10	36	K	10	54	N	10

Table A2: List of problems and dimensions for the general set.

Prob	Orig Prob	Dimen
1	A	2
2	B	4
3	C	2
4	F	6
5	F	9
6	G	5
7	G	6
8	G	7
9	H	10
10	H	30
11	I	10
12	J	2
13	J	10
14	L	10
15	M	10
16	N	10

Table A3: List of problems and dimensions for the general subset.

Prob	Orig Prob	Dimen
1	0	2
2	0	2
3	P	6
4	P	6
5	P	6
6	P	6
7	Q	10
8	Q	10
9	Q	10
10	Q	10
11	Q	10
12	Q	10

Table A4: List of problems and dimensions for the chemical equilibrium set.

APPENDIX B

DESCRIPTION OF THE CHEMICAL ENGINEERING PROBLEMS

HEX1 and HEX1M

The flowsheet is shown in fig B1. We will use the following notation.

T_i temp of stream i

F_i flowrate of stream i

cp_i specific heat of stream i

The known data for this problem are:

$T_1 = 140$ $T_3 = 320$ $T_4 = 320$

$T_6 = 240$ $T_8 = 500$ $T_9 = 560$

$T_{14} = 280$ $T_{15} = 100$ $T_{16} = 160$

$F_4 = 2.7778$ $F_6 = 2.306$

$cp_1 = 0.7$ $cp_4 = 0.6$ $cp_6 = 0.5$

$cp_9 = 0.5$ $cp_{11} = 0.8$ $cp_{15} = 1.0$

	UA
EX1	3.975
EX2	1.665
EX3	0.81
EX4	0.81

The latent heat of steam for the condenser is 656.6.

HEX2 and HEX2M

The known data are

$F_1 = 1$ $F_8 = 0.6$ $F_{10} = 0.4$

$F_{12} = 0.7$ $T_1 = 80$ $T_7 = 190$

$T_8 = 250$ $T_{10} = 220$ $T_{12} = 180$

$cp_1 = 0.8$ $cp_8 = 0.7$ $cp_{10} = 0.6$

$cp_{12} = 1$

	UA
HEX1	0.44
HEX2	1.20
HEX3	0.31

CAVSIM

Five components: CO_2 ETHANE N-BUTANE N-HEXANE N-DECANE

Feed: 0.5124, 0.3625, 0.1205, 0.0932, 0.0266.

	T	P
FL1	311	56.2
FL2	322	19.6
FL3	309	4.39
FL4	303	1.91

CAVDES

Same as CAVSIM except instead of T on FL4 the data is the flowrate for the second component on stream. 12 is 2.723×10^{-3}.

CHEMCOl

Feed hydrocarbon-inert gas mixture (1) is mixed with liq (11) returning from vessel V2 in mixer Ml.

This stream is expanded somewhat through valve Rl and separated by simple flash in vessel Vl.

Top vapour of Vl (5) is expanded to given temp and pres (9) and is then separated by flash at same pressure and temperature in V2. Top vapour of V2 (13) is gas product 1.

The liquid bottoms of Vl (7) picks up heat resulting in a small temperature rise indicated and then passes through reducing valve R2.

The pressure after R2 is such that the subsequent adiabatic flash produces vapour in the ratio V/F as given.

The liquid bottom product of V3 (15) is cooled further by exch El to stream (17) = the liquid product 1.

The top vapour of V3 (19) is compressed by C2. The power needed for this compression is provided by expander C1.

The compressor C2 outlet (21) is split in blackbox V4.

Operations taking place in V4 can be represented by the indicated component split ratios.

The resulting streams (23) and (25) have the same pressure drop as indicated. Stream (23) is at its dewpoint temperature and stream (25) has the indicated temperature difference from (21).

Feed is stream 1.

Products are: Streams (13) = vap, (23) = vap, (17) = liquid and
(25) = liq/vapour mix
Recycle is: stream (11) = liquid.

The values for the known data are shown together with the flowsheet in figure 5.4.

CHEMCO2

Input data:

Feed: 6000 k moles/hr

Feed composition:

CO_2	6	vol %
CH_4	3.2	vol %
CO	14.6	vol %
N_2	72	vol %
H_2	4.1	vol %
H_2O	0.1	vol %

Recycle ratio (Str nr 6/Str nr 1) 3.5 mole basis.
Reactor pressure 90 bar.
Temperature reactor outlet 270^{o}C.

T in degree Kelvin.

Carry out the reactor calculation as follows:

Consider watergas reaction (2) in chemical equilibrium at reactor outlet conditions.

Consider methanol reaction (1) in equilibrium 10^{o}k below the temperature at reactor outlet.

Model G/L separator

One stage simple flash.

Outlet temperature 40^{o}C, no pressure drop.

Only methanol and H_2O are in liquid phase $K_{methanol}$ = 3.8 E-3 (40^{o}C)

K_{H_2O} = 8.E-4.(40^{o}C).

Nitric acid plant

Problem specification.

Components:

HNO_3
N_2
O_2
H_2O
CH_4
CO
CO_2
NH_3

UNIT (1) SIMPLE DISTILLATION

100% N_2 & H_2 are taken overhead and an unknown percentage of NH_3.

UNIT (2) FEED

Unknown total flow of air composition

79% N_2, 21% O_2.

UNIT (3) MIXER

UNIT (4) REACTOR

Reaction taking place $NH_3 + 2O_2 \rightarrow HNO_3 + H_2O$

100% conversion of NH_3.

UNIT (5) FEED

Unknown total flow of H_2O.

UNIT (6) MIXER

UNIT (7) SIMPLE DISTILLATION

100% HNO_3 and 100% H_2O taken overhead.

UNIT (8) SIMPLE DISTILLATION

Unknown percentage of N_2 and O_2 taken overhead.

UNIT (9) FEED

Unknown total flow of H_2O.

UNIT (10) FEED

Unknown total flow of CH_4.

UNIT (11) MIXER

UNIT (12) REACTOR

The reaction taking place is $H_2O \rightarrow H_2 + \frac{1}{2}O_2$

Unknown conversion of water.

UNIT (13) REACTOR

The reaction taking place is $2CH_4 + O_2 \rightarrow 2CO + 4H_2$

Unknown conversion of O_2.

UNIT (14) REACTOR

The reaction taking place is $CH_4 + O_2 \rightarrow CO_2 + 2H_2$

Unknown conversion of O_2.

UNIT (15) SIMPLE DISTILLATION

100% CO & CO_2 are taken overhead.

UNIT (16) SIMPLE DISTILLATION

100% N_2, 99% H_2 and 100% NH_3 taken overhead.

UNIT (17) MIXER

UNIT (18) REACTOR

The reaction taking place is $N_2 + 3H_2 \rightarrow 2NH_3$

25% conversion of N_2.

DESIGN CONSTRAINTS

There are 10 design variables. The design constraints are:

(1) HNO_3 in stream 4 is 0.952 kmoles/hr

(2) H_2O in stream 4 is 2.222 kmoles/hr

(3) O_2 in stream 11 is zero kmoles/hr

(4) H_2O in stream 11 is zero kmoles/hr

(5) CH_4 in stream 11 is zero kmoles/hr

(6) N_2 in stream 6 is 99.2% of the total flow

(7) O_2 in stream 8 is 1% of the total flow

(8) CO_2 in stream 12 is 91% of the total flow

(9) N_2 in stream 14 is 25% of the total flow

(10) H_2 in stream 14 is 75% of the total flow.

Crude preheat train

Problem specifications

Feed streams:

STREAM	Flowrate	Temp (^{o}C)	C_p
1	680	25	0.555
42	480	219	2.912
48	290	159	1.393
50	250	352	0.523
56	160	41	1.068
60	110	263	0.655
65	83	206	0.659
70	39	304	0.674
75	510	315	0.686
78	130	142	0.642
80	580	152	4.482
83	160	133	1.681
86	38	170	0.701

Product Stream temperatures:

STREAM	Temp (^{o}C)	STREAM (cont)	Temp (^{o}C) (cont)
84	83.9	66	170.2
87	107.6	62	209.1
47	197.2	71	273.9
57	133.1	54	201
49	203.4		
79	199.7		
41	306.5		
72	70.7		
67	111		
63	84.9		
55	69		

Other measured temperatures:

STREAM	Temp (^{o}C)
39	296
38	215.3
53	211.3
29	162.7
30	155.6
31	151.9
32	176.9
53	193.6
14	161
15	152.2
16	136.3
17	108.6
18	120.7
4	126.8
89	142.6
37	281.7
36	219.68

Simple ethylene plant

Problem specification.

Components	Feed stream 1 (kmols/hr)
Hydrogen	0
Methane	0
Ethane	80
Propane	120
n-butane	0
Ethylene	0
Propylene	0
Temperature	295 K
Pressure	41.37 bars

UNIT (1) VALVE

UNIT (2) FIRED HEATER/FURNACE

UNIT (3) REACTOR
The reaction taking place is
$3C_2H_6 + 6C_3H_8 \quad 4H_2 + 4CH_4 + 5C_2H_4 + 2C_3H_6 + C_4H_{10}$
90 % conversion of C_3H_8.

UNIT (4) MIXER

UNIT (5) PUMP

UNIT (6) DIST/SIMPLE DISTILLATION
100% H_2 and CH_4 are taken overhead

UNIT (7) DIST/SIMPLE DISTILLATION
5% Ethane and 99% Ethylene are taken overhead

UNIT (8) DIST/SIMPLE DISTILLATION
99% of the ethane remaining and 100% of the etylene remaining are taken overhead.

UNIT (9) FIRED HEATER/FURNACE

UNIT (10) REACTOR
The reaction taking place is $4C_2H_6 \quad 2H_2 + 2CH_4 + 3C_2H_4$
80% conversion of C_2H_6

DISTORTED PAGES IN ORIGINAL

APPENDIX C

LIBRARY OF MODELS FOR CHEMCOI

```
#**********************************************************#
#*                                                        *#
#*  L I b K A R Y   O F   M O D E L S   F O R             *#
#*                                                        *#
#*            S P E E D U P                               *#
#*                                                        *#
#**********************************************************#
MODEL EXPANDER
#************#
SET NOCOMP
TYPE F AS ARRAY(NUCOMP) OF FLOWRATE
     TIN,TOU,TIDEAL AS TEMPERATURE
     PIN,POU AS PRESSURE
     HIN,HOU,HIDEAL,WORK AS ENTHALPY
     EFFIC AS FRACTION
     GAMMA AS NOTYPE
STREAM
     INPUT IS F,TIN,PIN,HIN
     OUTPUT 1 IS F,TOU,POU,HOU
     OUTPUT 2 IS WORK
EQUATION
     TIDEAL * PIN ** GAMMA = TIN * POU ** GAMMA;
     EFFIC * ( HIN - HIDEAL ) = HIN - HOU;
     WORK = HOU - HIN;
PROCEDURE
  ( HIDEAL ) ZCALEN ( TIDEAL , POU , F )
  ( HOU    ) ZCALEN ( TOU   , POU , F )
****
MODEL COMPRESSOR
#**************#
SET NOCOMP
TYPE F AS ARRAY(NUCOMP) OF FLOWRATE
     TIN,TOU,TIDEAL AS TEMPERATURE
     PIN,POU AS PRESSURE
     HIN,HOU,HIDEAL,WORK AS ENTHALPY
     EFFIC AS FRACTION
     GAMMA AS NOTYPE
STREAM
     INPUT 1 IS F,TIN,PIN,HIN
     INPUT 2 IS WORK
     OUTPUT IS F,TOU,POU,HOU
EQUATION
     TIDEAL * PIN ** GAMMA = TIN * POU ** GAMMA;
     EFFIC * ( HIN - HOU ) = HIN - HIDEAL;
     WORK = HOU - HIN;
PROCEDURE
  ( HIDEAL ) ZCALEN ( TIDEAL , POU , F )
  ( HOU    ) ZCALEN ( TOU   , POU , F )
****
```

```
MODEL HEATER_COOLER
#********************#
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF  FLOWRATE
     TIN,TOU AS TEMPERATURE
     PIN,POU AS PRESSURE
     HIN,HOU,Q AS ENTHALPY
     DELTAT,DELTAP AS DELTA
STREAM
     INPUT IS F,TIN,PIN,HIN
     OUTPUT IS F,TOU,POU,HOU
EQUATION
     TOU = TIN + DELTAT;
     POU = PIN + DELTAP;
     Q = HOU - HIN;
PROCEDURE
     ( HOU ) ZCALEN ( TOU , POU , F )
****
MODEL VALVE
#******************#
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF  FLOWRATE
     TIN,TOU AS TEMPERATURE
     PIN,POU AS PRESSURE
     H AS ENTHALPY
     DELTAP AS DELTA
STREAM
     INPUT IS F,TIN,PIN,H
     OUTPUT IS F,TOU,POU,H
EQUATION
     POU = PIN - DELTAP;
PROCEDURE
     ( TOU ) ZCALIP ( H , POU , F )
****
MODEL PUMP
#*************#
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
     PIN,POU AS PRESSURE
     HIN,HOU AS ENTHALPY
     DELTAP AS DELTA
STREAM
     INPUT IS F,T,PIN,HIN
     OUTPUT IS F,T,POU,HOU
EQUATION
     POU = PIN + DELTAP;
PROCEDURE
     ( HOU ) ZCALEN ( T , POU , F )
****
```

```
MODEL MOL_TO_KG_CONVERTER
#************************#
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
     P AS PRESSURE
     H AS ENTHALPY
     X AS ARRAY(NOCOMP) OF FRACTION
     FKG AS ARRAY(NOCOMP) OF KGFLOW
     MOLW AS ARRAY(NOCOMP) OF NOTYPE
     FTOTALKG AS KGFLOW
STREAM
     INPUT IS F,T,P,H
     OUTPUT IS F,T,P,H
EQUATION
     FKG = MOLW * F;
     FTOTALKG = SIGMA ( FKG );
     X * FTOTALKG = FKG;
****
MODEL MOLAR_FRACTION_TO_MOLES_CONVERTER
#*************************************#
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
     P AS PRESSURE
     H AS ENTHALPY
     X AS ARRAY(NOCOMP) OF FRACTION
     FTOTAL AS FLOWRATE
STREAM
     INPUT IS F,T,P,H
     OUTPUT IS F,T,P,H
EQUATION
     FTOTAL * X = F;
     FTOTAL = SIGMA ( F ) ;
****
MODEL ADIABATIC_FLASH
#********************#
SET NOCOMP
TYPE F,TOP,BOT AS ARRAY(NOCOMP) OF FLOWRATE
     T,TTOP,TBOT AS TEMPERATURE
     P AS PRESSURE
     H,HTOP,HBOT AS ENTHALPY
     FRAC AS FRACTION
STREAM
     INPUT IS F,T,P,H
     OUTPUT 1 IS TOP,TTOP,P,HTOP
     OUTPUT 2 IS BOT,TBOT,P,HBOT
EQUATION
     TBOT = TTOP;
     SIGMA(TOP) = FRAC * SIGMA(F);
PROCEDURE
     ( TTOP , TOP , BOT ) SCALTP ( H , P , F)
     ( HTOP ) ZENTHV ( TTOP , P , TOP )
     ( HBOT ) ZENTHL ( TBOT , P , BOT )
****
```

```
MODEL ISOTHERMAL_FLASH
#**********************#
SET NOCOMP
TYPE F,TOP,BOT AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
     P AS PRESSURE
     H,HTOP,HBOT AS ENTHALPY
STREAM
     INPUT IS F,T,P,H
     OUTPUT IS TOP,T,P,HTOP
     OUTPUT 2 IS BOT,T,P,HBOT
PROCEDURE
     ( TOP , BOT ) SPHASE ( T , P , F )
     ( HTOP ) ZENIHV ( T , P , TOP )
     ( HBOT ) ZENIHL ( T , P , BOT )
****
MODEL VOLUME_TO_MOLES_CONVERTER
#**************************#
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
     P AS PRESSURE
     H AS ENTHALPY
     TOTALFVOL,RHO AS NOTYPE
     TOTALF AS FLOWRATE
STREAM
     INPUT IS F,T,P,H
     OUTPUT IS F,T,P,H
EQUATION
     TOTALF = RHO * TOTALFVOL;
     TOTALF = SIGMA ( F );
****
MODEL MIXER
#**********#
SET NOCOMP
TYPE F1,F2,F AS ARRAY(NOCOMP)OF FLOWRATE
     T1,T2,T AS TEMPERATURE
     P1,P2,P AS PRESSURE
     H1,H2,H AS ENTHALPY
STREAM
     INPUT 1 IS F1,T1,P1,H1
     INPUT 2 IS F2,T2,P2,H2
     OUTPUT IS F,T,P,H
EQUATION
     F = F1 + F2;
     P = P1;
     H = H1 + H2;
PROCEDURE
     ( T ) = ZCALTP ( H , P , F )
****
```

```
MODEL BLACKBOX
SET NOCOMP
TYPE FIN,TOP,BOT AS ARRAY(NOCOMP) OF FLOWRATE
     TIN,TTOP,TBOT AS TEMPERATURE
     PIN,PTOP,PBOT AS PRESSURE
     HIN,HTOP,HBOT AS ENTHALPY
     DELTAP,DELTAT AS DELTA
     SPLITR,XTOP,XBOT AS ARRAY(NOCOMP) OF FRACTION
     COMP67 AS FRACTION
     TOTALTOP,TOTALBOT AS FLOWRATE
     TOTALTOPVOL,RHO AS NOTYPE
     BOTKG,TOPKG AS ARRAY(NOCOMP) OF KGFLOW
     MOLW AS ARRAY(NOCOMP) OF NOTYPE
     TOTALBOTKG,TOTALTOPKG AS KGFLOW
STREAM
  INPUT     IS FIN,TIN,PIN,HIN
  OUTPUT 1 IS TOP,TTOP,PTOP,HTOP
  OUTPUT 2 IS BOT,TBOT,PBOT,HBOT
EQUATION
  FIN = TOP + BOT;
  BOT = SPLITR * FIN;
  PTOP = PIN + DELTAP;
  PBOT = PTOP;
  TBOT = TIN + DELTAT;
  TOTALTOP = SIGMA(TOP);
  TOTALBOT = SIGMA(BOT);
  XTOP * TOTALTOP = TOP;
  XBOT * TOTALBOT = BOT;
  TOTALTOP = RHO * TOTALTOPVOL;
  COMP67 = XTOP(6) + XTOP(7);
  TOTALBOTKG = SIGMA(BOTKG);
  TOTALTOPKG = SIGMA(TOPKG);
  BOTKG = MOLW * BOT;
  TOPKG = MOLW * TOP;
PROCEDURE
  ( TTOP ) SDEWPT ( PTOP , TOP )
  ( HTOP ) ZENTHV ( TTOP , PTOP , TOP )
  ( HBOT ) ZENTHL ( TBOT , PBOT , BOT )
****
```

APPENDIX C'

Numerical results obtained for the general set of mathematical problems using the Broyden's version of the continuation method.The parameter λ in the in the table is as proposed by BROYDEN(1969).

		λ		
		0.5	0.1	0.05
failures	unscaled	21	21	21
	variables badly scaled	22	22	23
	functions badly scaled	30	31	32
	Total	73	74	76

A maximun of 2000 function evaluations was allowed(for this reason some problems which converged for λ =0.5 did not converge for $\lambda < 0.5$).The code used for solving each subproblem is the one described in chapter 2.

APPENDIX D

COPIED FROM MICRO FICHE

```
C**********************************************************************
C**********************************************************************
C**                                                                  **
C**                                                                  **
C**    SUBROUTINES    V E C F C N     AND     I N I T P T            **
C**                                                                  **
C**                                                                  **
C**********************************************************************
C**********************************************************************
      SUBROUTINE VECFCN(N,X,FVEC,NPROB,IER)
      INTEGER N,NPROB
      REAL X(N),FVEC(N)
C        *******
C
C     SUBROUTINE VECFCN
C
C     THIS SUBROUTINE DEFINES FOURTEEN TEST FUNCTIONS. THE FIRST
C     FIVE TEST FUNCTIONS ARE OF DIMENSIONS 2,4,2,4,3, RESPECTIVELY,
C     WHILE THE REMAINING TEST FUNCTIONS ARE OF VARIABLE DIMENSION
C     N FOR ANY N GREATER THAN OR EQUAL TO 1 (PROBLEM 6 IS AN
C     EXCEPTION TO THIS, SINCE IT DOES NOT ALLOW N = 1).
C
C     THE SUBROUTINE STATEMENT IS
C
C       SUBROUTINE VECFCN(N,X,FVEC,NPROB)
C
C     WHERE
C
C       N IS A POSITIVE INTEGER INPUT VARIABLE.
C
C       X IS AN INPUT ARRAY OF LENGTH N.
C
C       FVEC IS AN OUTPUT ARRAY OF LENGTH N WHICH CONTAINS THE NPROB
C         FUNCTION VECTOR EVALUATED AT X.
C
C       NPROB IS A POSITIVE INTEGER INPUT VARIABLE WHICH DEFINES THE
C         NUMBER OF THE PROBLEM. NPROB MUST NOT EXCEED 14.
C
C     SUBPROGRAMS CALLED
C
C       FORTRAN-SUPPLIED ... ATAN,COS,EXP,SIGN,SIN,SQRT,
C                            MAX0,MIN0
C
C     MINPACK. VERSION OF JULY 1978.
C     BURTON S. GARBOW, KENNETH E. HILLSTROM, JORGE J. MORE
C
C     **********
      INTEGER I,IEV,IVAR,J,K,K1,K2,KP1,ML,MU
      REAL C1,C2,C3,C4,C5,C6,C7,C8,C9,EIGHT,FIVE,H,ONE,PROD,SUM,SUM1,
     *     SUM2,TEMP,TEMP1,TEMP2,TEN,THREE,TI,TJ,TK,TPI,TWO,ZERO
      REAL FLOAT
      DATA ZERO,ONE,TWO,THREE,FIVE,EIGHT,TEN
     *     /0.0E0,1.0E0,2.0E0,3.0E0,5.0E0,8.0E0,1.0E1/
      DATA C1,C2,C3,C4,C5,C6,C7,C8,C9
     *     /1.0E4,1.0001E0,2.0E2,2.02E1,1.98E1,1.8E2,2.5E-1,5.0E-1,
     *      2.9E1/
      DIMENSION CEP3R(12)
      DATA (CEP3R(I),I=1,12)/2*4.,2*10.,2*40.,2*4.,2*10.,2*40./
      FLOAT(IVAR) = IVAR
C
C     PROBLEM SELECT .
C
      IF (NPROB.LE.18) GO TO 5
      IF (NPROB.LE.20) GO TO 800
      IF (NPROB.LE.24) GO TO 900
```

```
      IF (NPROB.LE.24) GO TO 1000
```

```
      IF (NPROB.LE.36) GO TO 1000
      IER=-99
      RETURN
$     CONTINUE
      GO TO (10,20,30,40,50,60,120,170,200,220,270,300,330,350,
     1       400,500,600,700,800,900,1000),NPROB
C
C     ROSENBROCK FUNCTION.
C
   10 CONTINUE
      FVEC(1) = ONE - X(1)
      FVEC(2) = TEN*(X(2) - X(1)**2)
      GO TO 380
C
C     POWELL SINGULAR FUNCTION.
C
   20 CONTINUE
      FVEC(1) = X(1) + TEN*X(2)
      FVEC(2) = SQRT(FIVE)*(X(3) - X(4))
      FVEC(3) = (X(2) - TWO*X(3))**2
      FVEC(4) = SQRT(TEN)*(X(1) - X(4))**2
      GO TO 380
C
C     POWELL BADLY SCALED FUNCTION.
C
   30 CONTINUE
      FVEC(1) = C1*X(1)*X(2) - ONE
      IF((X(1).GT.200.).OR.(X(2).GT.200.))GO TO 35
      TE1=0.
      TE2=0.
      IF(X(1).GT.-200.)TE1=EXP(-X(1))
      IF(X(2).GT.-200.)TE2=EXP(-X(2))
      FVEC(2)=TE1+TE2-C2
      GO TO 380
   35 IER=-9
      GO TO 380
C
C     WOOD FUNCTION.
C
   40 CONTINUE
      TEMP1 = X(2) - X(1)**2
      TEMP2 = X(4) - X(3)**2
      FVEC(1) = -C3*X(1)*TEMP1 - (ONE - X(1))
      FVEC(2) = C3*TEMP1 + C4*(X(2) - ONE) + C5*(X(4) - ONE)
      FVEC(3) = -C6*X(3)*TEMP2 - (ONE - X(3))
      FVEC(4) = C6*TEMP2 + C4*(X(4) - ONE) + C5*(X(2) - ONE)
      GO TO 380
C
C     HELICAL VALLEY FUNCTION.
C
   50 CONTINUE
      TPI = EIGHT*ATAN(ONE)
      TEMP1 = SIGN(C7,X(2))
      IF (X(1) .GT. ZERO) TEMP1 = ATAN(X(2)/X(1))/TPI
      IF (X(1) .LT. ZERO) TEMP1 = ATAN(X(2)/X(1))/TPI + C8
      TEMP2 = SQRT(X(1)**2+X(2)**2)
      FVEC(1) = TEN*(X(3) - TEN*TEMP1)
      FVEC(2) = TEN*(TEMP2 - ONE)
      FVEC(3) = X(3)
      GO TO 380
C
C     WATSON FUNCTION.
C
   60 CONTINUE
      DO 70 K = 1, N
         FVEC(K) = ZERO
```

```
      FVEC(K) = ZERO
   70 CONTINUE
      DO 110 I = 1, 29
         TI = FLOAT(I)/C9
         SUM1 = ZERO
         TEMP = ONE
         DO 80 J = 2, N
            SUM1 = SUM1 + FLOAT(J-1)*TEMP*X(J)
            TEMP = TI*TEMP
   80       CONTINUE
         SUM2 = ZERO
         TEMP = ONE
         DO 90 J = 1, N
            SUM2 = SUM2 + TEMP*X(J)
            TEMP = TI*TEMP
   90       CONTINUE
         TEMP1 = SUM1 - SUM2**2 - ONE
         TEMP2 = TWO*TI*SUM2
         TEMP = ONE/TI
         DO 100 K = 1, N
            FVEC(K) = FVEC(K) + TEMP*(FLOAT(K-1) - TEMP2)*TEMP1
            TEMP = TI*TEMP
  100       CONTINUE
  110    CONTINUE
      TEMP = X(2) - X(1)**2 - ONE
      FVEC(1) = FVEC(1) + X(1)*(ONE - TWO*TEMP)
      FVEC(2) = FVEC(2) + TEMP
      GO TO 380
C
C     CHEBYQUAD FUNCTION.
C
  120 CONTINUE
      DO 130 K = 1, N
         FVEC(K) = ZERO
  130    CONTINUE
      DO 150 J = 1, N
         TEMP1 = ONE
         TEMP2 = TWO*X(J) - ONE
         TEMP = TWO*TEMP2
         DO 140 I = 1, N
            FVEC(I) = FVEC(I) + TEMP2
            TI = TEMP*TEMP2 - TEMP1
            TEMP1 = TEMP2
            TEMP2 = TI
  140       CONTINUE
  150    CONTINUE
      TK = ONE/FLOAT(N)
      IEV = -1
      DO 160 K = 1, N
         FVEC(K) = TK*FVEC(K)
         IF (IEV .GT. 0) FVEC(K) = FVEC(K) + ONE/(FLOAT(K)**2 - ONE)
         IEV = -IEV
  160    CONTINUE
      GO TO 380
C
C     BROWN ALMOST-LINEAR FUNCTION.
C
  170 CONTINUE
      SUM = -FLOAT(N+1)
      PROD = ONE
      DO 180 J = 1, N
         SUM = SUM + X(J)
C
C        ADDED TO AVOID CRASHING
C
         IF (PROD.LE.1.E100) GO TO 173
           AUX1=ALOG10(ABS(PROD))
         AUX2=ALOG10(ABS(X(J)))
         IF ((AUX1+AUX2).LT.200) GO TO 173
```

```
      AUX1=ALOG10 (ABS (PROD))
      AUX2=ALOG10 (ABS (X(J)))
      IF ((AUX1+AUX2) .LT. 200) GO TO 173
      IER=-9
      RETURN
3     CONTINUE
      PROD = X(J) *PROD
180   CONTINUE
      DO 190 K = 1, N
         FVEC(K) = X(K) + SUM
190   CONTINUE
      FVEC(ND = PROD - ONE
      GO TO 380

      DISCRETE BOUNDARY VALUE FUNCTION.

200 CONTINUE
      H = ONE/FLOAT(N+1)
      DO 210 K = 1, N
         TEMP = (X(K) + FLOAT(K)*H + ONE)**3
         TEMP1 = ZERO
         IF (K .NE. 1) TEMP1 = X(K-1)
         TEMP2 = ZERO
         IF (K .NE. N) TEMP2 = X(K+1)
         FVEC(K) = TWO*X(K) - TEMP1 - TEMP2 + TEMP*H**2/TWO
210   CONTINUE
      GO TO 380

      DISCRETE INTEGRAL EQUATION FUNCTION.

220 CONTINUE
      H = ONE/FLOAT(N+1)
      DO 260 K = 1, N
         TK = FLOAT(K)*H
         SUM1 = ZERO
         DO 230 J = 1, K
            TJ = FLOAT(J)*H
            TEMP = (X(J) + TJ + ONE)**3
            SUM1 = SUM1 + TJ*TEMP
230      CONTINUE
         SUM2 = ZERO
         KP1 = K + 1
         IF (N .LT. KP1) GO TO 250
         DO 240 J = KP1, N
            TJ = FLOAT(J)*H
            TEMP = (X(J) + TJ + ONE)**3
            SUM2 = SUM2 + (ONE - TJ)*TEMP
240      CONTINUE
250      CONTINUE
         FVEC(K) = X(K) + H*((ONE - TK)*SUM1 + TK*SUM2)/TWO
260   CONTINUE
      GO TO 380

      TRIGONOMETRIC FUNCTION.

270 CONTINUE
      DO 272 J=1,N
      IF(ABS(X(J)).GT.1.E14) IER=-9
272   CONTINUE
      IF (IER.LT.0)RETURN
      SUM = ZERO
      DO 280 J = 1, N
         FVEC(J) = COS(X(J))
         SUM = SUM + FVEC(J)
280   CONTINUE
      DO 290 K = 1, N
         FVEC(K) = FLOAT(N+K) - SIN(X(K)) - SUM - FLOAT(K)*FVEC(K)
290   CONTINUE
      GO TO 380
```

```
  290    CONTINUE
         GO TO 380
C
C        VARIABLY DIMENSIONED FUNCTION.
C
  300 CONTINUE
         SUM = ZERO
         DO 310 J = 1, N
            SUM = SUM + FLOAT(J) * (X(J) - ONE)
  310    CONTINUE
         TEMP = SUM*(ONE + TWO*SUM**2)
         DO 320 K = 1, N
            FVEC(K) = X(K) - ONE + FLOAT(K)*TEMP
  320    CONTINUE
         GO TO 380
C
C        BROYDEN TRIDIAGONAL FUNCTION.
C
  330 CONTINUE
         DO 340 K = 1, N
            TEMP = (THREE - TWO*X(K))*X(K)
            TEMP1 = ZERO
            IF (K .NE. 1) TEMP1 = X(K-1)
            TEMP2 = ZERO
            IF (K .NE. N) TEMP2 = X(K+1)
            FVEC(K) = TEMP - TEMP1 - TWO*TEMP2 + ONE
  340    CONTINUE
         GO TO 380
C
C        BROYDEN BANDED FUNCTION.
C
  350 CONTINUE
         ML = 5
         MU = 1
         DO 370 K = 1, N
            K1 = MAX0(1,K-ML)
            K2 = MIN0(K+MU,N)
            TEMP = ZERO
            DO 360 J = K1, K2
               IF (J .NE. K) TEMP = TEMP + X(J)*(ONE + X(J))
  360       CONTINUE
            FVEC(K) = X(K)*(TWO + FIVE*X(K)**2) + ONE - TEMP
  370    CONTINUE
         GO TO 380
C
C        BROWN FUNCTION - BUS EXAMPLE 5.2.2
C
  400    CONTINUE
         FVEC(1)=X(1)**2-X(2)-1.
         FVEC(2)=(X(1)-2.)**2+(X(2)-0.5)**2-1.
         GO TO 380
C
C        FREUDENSTEIN & ROTH FUNCTION
C
  500    CONTINUE
         FVEC(1)=-13.+X(1)+((-X(2)+5.)*X(2)-2.)*X(2)
         FVEC(2)=-29.+X(1)+((X(2)+1.)*X(2)-14.)*X(2)
         GO TO 380
C
C        GHERI & MANCINO FUNCTION
C
  600    CONTINUE
         IALPHA=5
         IBETA=14
         IGAMMA=3
         DO 620 K=1,N
         SUM=0.
         DO 610 J=1,N
         IF(J.EQ.K)GO TO 610
         ZIJ=SQRT(X(J)**2+FLOAT(K)/J)
         SUM=SUM+(ZIJ*(SIN(ALOG(ZIJ))**IALPHA+
     1       COS(ALOG(ZIJ))**IALPHA))
  620    CONTINUE
```

```
      DO 610 J=1,N
      IF (J.EQ.K) GO TO 610
      ZIJ=SQRT(X(J)**2+FLOAT(K)/J)
      SUM=SUM+(ZIJ*(SIN(ALOG(ZIJ))**IALPHA+
     1     COS(ALOG(ZIJ))**IALPHA))
610   CONTINUE
      FVEC(K)=SUM+IBETA*N*X(K)+(K-N/2)**IGAMMA
620   CONTINUE
      GO TO 380
C
C     FOUR REACTORS FUNCTION
C
700   CONTINUE
      XS=0.5
      A0=1.
      B5=1.
      A5=0.
      B0=0.
      THETK1=4.
      XSC=1.-XS
      FVEC(1)=XS*A0+XSC*X(2)-X(1)-THETK1*X(1)*X(5)
      FVEC(2)=XS*B0+XSC*X(6)-X(5)-THETK1*X(1)*X(5)
      FVEC(3)=XS*X(1)+XSC*X(3)-X(2)-THETK1*X(2)*X(6)
      FVEC(4)=XS*X(5)+XSC*X(7)-X(6)-THETK1*X(2)*X(6)
      FVEC(5)=XS*X(2)+XSC*X(4)-X(3)-THETK1*X(3)*X(7)
      FVEC(6)=XS*X(6)+XSC*X(8)-X(7)-THETK1*X(3)*X(7)
      FVEC(7)=XS*X(3)+XSC*A5-X(4)-THETK1*X(4)*X(8)
      FVEC(8)=XS*X(7)+XSC*B5-X(8)-THETK1*X(4)*X(8)
      GO TO 380
C
C     CHEMICAL EQUILIBRIUM PROBLEM 1
C
800   CONTINUE
      FVEC(1)=X(2)-10.
      FVEC(2)=X(1)*X(2)-5.E4
      GO TO 380
C
C     CHEMICAL EQUILIBRIUM PROBLEM 2
C
900   CONTINUE
      FVEC(1)=X(1)+X(2)+X(4)-0.001
      FVEC(2)=X(5)+X(6)-55.
      FVEC(3)=X(1)+X(2)+X(3)+2.*X(5)+X(6)-110.001
      FVEC(4)=X(1)-0.1*X(2)
      FVEC(5)=X(1)-1.E4*X(3)*X(4)
      FVEC(6)=X(5)-55.*1.E14*X(3)*X(6)
      GO TO 380
C
C     CHEMICAL EQUILIBRIUM PROBLEM 3
C
1000    CONTINUE
      R=CEP3R(NPROB-24)
      TOT=0.
      DO 1010 I=1,N
1010  TOT=TOT+X(I)
      IF(X(1).LT.0.)GO TO 1050
      IF(X(2)*X(4)*TOT.LT.0.)GO TO 1050
      IF(X(4).LT.0.)GO TO 1050
      IF(X(1)*X(2)*TOT.LT.0.)GO TO 1050
      IF(X(3)*TOT.LT.0.)GO TO 1050
      FVEC(1)=X(1)+X(4)-3.
      FVEC(2)=2.*X(1)+X(2)+X(4)+X(7)+X(8)+X(9)+2.*X(10)-R
      FVEC(3)=2.*X(2)+2.*X(5)+X(6)+X(7)-8.
      FVEC(4)=2.*X(3)+X(9)-4.*R
      FVEC(5)=X(1)*X(5)-1.93*1.E-1*X(2)*X(4)
      FVEC(6)=X(6)*SQRT(X(1))-2.597*1.E-3*SQRT(X(2)*X(4)*TOT)
      FVEC(7)=X(7)*SQRT(X(4))-3.448*1.E-3*SQRT(X(1)*X(2)*TOT)
      FVEC(8)=X(8)*X(4)-1.799*1.E-5*X(1)*TOT
      FVEC(9)=X(9)*X(4)-2.155*1.E-4*X(1)*SQRT(X(3)*TOT)
      FVEC(10)=X(10)*X(4)*X(4)-3.846*1.E-5*X(4)*X(4)*TOT
      GO TO 380
1050  IER=-9
```

```
      FVEC(8)=X(8)*X(9)-1.799*1.E-5*X(1)*TOT
      FVEC(9)=X(9)*X(4)-2.155*1.E-4*X(1)*SQRT(X(3)*TOT)
      FVEC(10)=X(10)*X(4)*X(4)-3.846*1.E-5*X(4)*X(4)*TOT
      GO TO 380
1050  IER=-9
      RETURN
  380 CONTINUE
      RETURN
C
C     LAST CARD OF SUBROUTINE VECFCN.
C
      END
      SUBROUTINE INITPT(N,X,NPROB,FACTOR)
      INTEGER N,NPROB
      REAL FACTOR
      REAL X(N)
C     **********
C
C     SUBROUTINE INITPT
C
C     THIS SUBROUTINE SPECIFIES THE STANDARD STARTING POINTS FOR
C     THE FUNCTIONS DEFINED BY SUBROUTINE VECFCN. THE SUBROUTINE
C     RETURNS IN X A MULTIPLE (FACTOR) OF THE STANDARD STARTIING
C     POINT. FOR THE SIXTH FUNCTION THE STANDARD STARTING POINT IS
C     ZERO, SO IN THIS CASE, IF FACTOR IS NOT UNITY, THEN THE
C     SUBROUTINE RETURNS THE VECTOR  X(J) = FACTOR, J=1,...,N.
C
C     THE SUBROUTINE STATEMENT IS
C
C       SUBROUTINE INITPT(N,X,NPROB,FACTOR)
C
C     WHERE
C
C       N IS A POSITIVE INTEGER INPUT VARIABLE.
C
C       X IS AN OUTPUT ARRAY OF LENGTH N WHICH CONTAINS THE STANDARD
C         STARTING POINT FOR PROBLEM NPROB MULTIPLIED BY FACTOR.
C
C       NPROB IS A POSITIVE INTEGER INPUT VARIABLE WHICH DEFINES THE
C         NUMBER OF THE PROBLEM. NPROB MUST NOT EXCEED 14.
C
C       FACTOR IS AN INPUT VARIABLE WHICH SPECIFIES THE MULTIPLE OF
C         THE STANDARD STARTING POINT. IF FACTOR IS UNITY, NO
C         MULTIPLICATION IS PERFORMED.
C
C     MINPACK. VERSION OF JULY 1978.
C     BURTON S. GARBOW, KENNETH E. HILLSTROM, JORGE J. MORE
C
C     **********
      INTEGER IVAR,J
      REAL C1,H,HALF,ONE,THREE,TJ,ZERO
      REAL FLOAT
      DATA ZERO,HALF,ONE,THREE,C1 /0.0E0,5.0E-1,1.0E0,3.0E0,1.2E0/
      DIMENSION CEP1(2,2),CEP2(6,4),CEP3(10,6),NSEAR(18)
      DATA ((CEP1(I,J),I=1,2),J=1,2)/2*0.,2*10./
      DATA ((CEP2(I,J),I=1,6),J=1,4)/6*0.,6*1.,1.E-4,1.E-3,0.,1.E-4,
     1                               55.,1.E-4,6*10./
      DATA ((CEP3(I,J),I=1,10),J=1,6)/0.,1.,8.,3.,5.,1.,4*0.,1.,
     1               0.,8.,2.,7.,1.,4*0.,1.,1.,10.,3*1.,4*0.,2.,
     2               2.,10.,1.,1.,2.,4*0.,2.,5.,40.,1.,5*0.,5.,1.,
     3               1.,20.,1.,5*0.,1./
      DATA (NSEAR(I),I=1,18)/1,2,1,2,3,4,1,2,3,4,5,6,1,2,3,4,5,6/
      FLOAT(IVAR) = IVAR
C
C     SELECTION OF INITIAL POINT.
C
      IF (NPROB.LE.18) GO TO 5
      NPR=NSEAR(NPROB-18)
      IF (NPROB.LE.20) GO TO 800
      IF (NPROB.LE.24) GO TO 900
      IF (NPROB.LE.36) GO TO 1000
      WRITE(6,4)
4     FORMAT(" PROBLEM OUT OF RANGE")
```

```
C     SELECTION OF INITIAL POINT.
C
      IF (NPROB.LE.18) GO TO 5
      NPR=NSEAR(NPROB-18)
      IF (NPROB.LE.20) GO TO 800
      IF (NPROB.LE.24) GO TO 900
      IF (NPROB.LE.36) GO TO 1000
      WRITE(6,4)
4     FORMAT(" PROBLEM OUT OF RANGE")
      STOP
5     CONTINUE
      GO TO (10,20,30,40,50,60,80,100,120,120,140,160,180,180,
     1       400,500,600,700,800,900,1000),NPROB
C
C     ROSENBROCK FUNCTION.
C
   10 CONTINUE
      X(1) = -C1
      X(2) = ONE
      GO TO 200
C
C     POWELL SINGULAR FUNCTION.
C
   20 CONTINUE
      X(1) = THREE
      X(2) = -ONE
      X(3) = ZERO
      X(4) = ONE
      GO TO 200
C
C     POWELL BADLY SCALED FUNCTION.
C
   30 CONTINUE
      X(1) = ZERO
      X(2) = ONE
      GO TO 200
C
C     WOOD FUNCTION.
C
   40 CONTINUE
      X(1) = -THREE
      X(2) = -ONE
      X(3) = -THREE
      X(4) = -ONE
      GO TO 200
C
C     HELICAL VALLEY FUNCTION.
C
   50 CONTINUE
      X(1) = -ONE
      X(2) = ZERO
      X(3) = ZERO
      GO TO 200
C
C     WATSON FUNCTION.
C
   60 CONTINUE
      DO 70 J = 1, N
         X(J) = ZERO
   70    CONTINUE
      GO TO 200
C
C     CHEBYQUAD FUNCTION.
C
   80 CONTINUE
      H = ONE/FLOAT(N+1)
      DO 90 J = 1, N
         X(J) = FLOAT(J)*H
   90    CONTINUE
      GO TO 200
C
C     BROWN ALMOST-LINEAR FUNCTION.
```

```
   90    CONTINUE
      GO TO 200
C
C     BROWN ALMOST-LINEAR FUNCTION.
C
  100 CONTINUE
      DO 110 J = 1, N
         X(J) = HALF
  110    CONTINUE
      GO TO 200
C
C     DISCRETE BOUNDARY VALUE AND INTEGRAL EQUATION FUNCTIONS.
C
  120 CONTINUE
      H = ONE/FLOAT(N+1)
      DO 130 J = 1, N
         TJ = FLOAT(J)*H
         X(J) = TJ*(TJ - ONE)
  130    CONTINUE
      GO TO 200
C
C     TRIGONOMETRIC FUNCTION.
C
  140 CONTINUE
      H = ONE/FLOAT(N)
      DO 150 J = 1, N
         X(J) = H
  150    CONTINUE
      GO TO 200
C
C     VARIABLY DIMENSIONED FUNCTION.
C
  160 CONTINUE
      H = ONE/FLOAT(N)
      DO 170 J = 1, N
         X(J) = ONE - FLOAT(J)*H
  170    CONTINUE
      GO TO 200
C
C     BROYDEN TRIDIAGONAL AND BANDED FUNCTIONS.
C
  180 CONTINUE
      DO 190 J = 1, N
         X(J) = -ONE
  190    CONTINUE
         GO TO 200
C
C        BROWN FUNCTION - BUS EXAMPLE 5.2.2
C
400      CONTINUE
         X(1)=-1.
         X(2)=1.5
         GO TO 200
C
C        FREUDENSTEIN & ROTH FUNCTION
C
500      CONTINUE
         X(1)=-5.
         X(2)=0.
         GO TO 200
C
C        GHERI & MANCINO FUNCTION
C
600      CONTINUE
```

```
      IALPHA=5
      IBETA=14
      IGAMMA=3
      KEI=IBETA*N+(IALPHA+1.)*(N-1)
      C=IBETA*N-(IALPHA+1.)*(N-1)
      DO 601 I=1,N
      SUM=0.
      DO 602 J=1,N
```

```
      IGAMMA=3
      KEI=IBETA*N+(IALPHA+1.)*(N-1)
      C=IBETA*N-(IALPHA+1.)*(N-1)
      DO 601 I=1,N
      SUM=0.
      DO 602 J=1,N
      IF(J.EQ.I)GO TO 602
      ZIJ=SQRT(FLOAT(I)/J)
      SUM=SUM+ZIJ*(SIN(ALOG(ZIJ))**IALPHA+COS(ALOG(ZIJ))**IALPHA)
602   CONTINUE
      X(I)=-((I-N/2.)**IGAMMA+SUM)*(C+KEI)/(2.*C*KEI)
601   CONTINUE
      GO TO 200
C
C     FOUR REACTORS FUNCTION
C
700   CONTINUE
      X(1)=0.1
      X(2)=0.2
      X(3)=0.3
      X(4)=0.4
      X(5)=0.5
      X(6)=0.4
      X(7)=0.3
      X(8)=0.2
      GO TO 200
C
C     CHEMICAL EQUILIBRIUM PROBLEM 1
C
800   CONTINUE
      DO 810 I=1,N
810   X(I)=CEP1(I,NPR)
      GO TO 200
C
C     CHEMICAL EQUILIBRIUM PROBLEM 2
C
900   CONTINUE
      DO 910 I=1,N
910   X(I)=CEP2(I,NPR)
      GO TO 200
C
C     CHEMICAL EQUILIBRIUM PROBLEM 3
C
1000  CONTINUE
      DO 1010 I=1,N
1010  X(I)=CEP3(I,NPR)
      GO TO 200
  200 CONTINUE
C
C     COMPUTE MULTIPLE OF INITIAL POINT.
C
      IF (FACTOR .EQ. ONE) GO TO 250
      IF (NPROB .EQ. 6) GO TO 220
         DO 210 J = 1, N
            X(J) = FACTOR*X(J)
  210       CONTINUE
         GO TO 240
  220 CONTINUE
         DO 230 J = 1, N
            X(J) = FACTOR
  230       CONTINUE
  240 CONTINUE
  250 CONTINUE
      RETURN
C
C     LAST CARD OF SUBROUTINE INITPT.
C
      END
```

```
C     LAST CARD OF SUBROUTINE INITPT.
C
      END
```

****
****
****
****
****
****
****
****
****

****
****
**   **
**   **
**   D E T A I L E D   R E S U L T S   F O R   T H E   **
**   **
**   G E N E R A L   S E T   **
**   **
**   **
****
****

METHOD DESCRIPTION

1 SCALE INVARIANT METHOD 1
2 SCALE INVARIANT METHOD 2
3 SCALE INVARIANT METHOD 3
4 SCALE INVARIANT METHOD 4
5 BROYDEN'S METHOD
6 POWELL'S HYBRID METHOD (MINPACK IMPLEMENTATION)

THE NUMBERS ON THE RIGTH TABLES ARE THE EFFICIENCY COEFFICIENTS

THE ENTRIES ON THE LEFT TABLES ARE AS FOLLOWS:

FOR METHODS 1 TO 5

>0 NUMBER OF FUNCTION EVALUATIONS USED (IT MEANS THE METHOD HAS CONVERGED)
0 AFTER 500 FUNCTION EVALUATIONS CONVERGENCE HAS NOT BEEN ACHIEVED
-2 CURRENT JACOBIAN IS SINGULAR AND THIS IS NOT AVOIDABLE
-5 THE STEP HAS BEEN REDUCED 10 TIMES WITHOUT ACHIEVING A REDUCTION IN THE NORM OF THE FUNCTION
-6 THE CURRENT STEP IS TOO SMALL (I.E. IT WILL NOT MODIFY THE VARIABLES VECTOR WHEN ADDED)
-9 A VALUE FOR ONE THE VARIABLES IS SUCH THAT A FUNCTION CAN NOT BE EVALUATED (FOR INSTANCE AN ARGUMENT FOR THE EXPONENTIAL FUNCTION IS TOO BIG)

FOR METHOD 6

-6 THE CODE IS NOT MAKING ANY PROGRESS (I.E. THE NORM OF THE FUNCTION IS NOT BEEN REDUCED AT ALL).

ALL THE OTHER CODES ARE THE SAME AS FOR METHODS 1 TO 5

W I T H O U T U S I N G T H E I N T E R N A L S C A L I N G

UNSCALED SET OF PROBLEMS

UNSCALED SET OF PROBLEMS

METH / PROB	1	2	3	4	5	6	1	2	3	4	5	6
1	6	6	6	5	6	23	1.20	1.20	1.20	1.00	1.20	4.60
2	27	27	37	34	25	41	1.08	1.08	1.48	1.36	1.00	1.64
3	74	98	29	46	66	179	2.55	3.38	1.00	1.59	2.28	6.17
4	0	227	111	74	164	96	0	3.07	1.50	1.00	2.22	1.30
5	28	433	0	246	242	27	1.04	16.04	0	9.11	8.96	1.00
6	363	129	103	125	99	91	3.99	1.42	1.13	1.37	1.09	1.00
7	113	103	145	130	111	143	1.10	1.00	1.41	1.26	1.08	1.39
8	20	16	18	32	16	18	1.25	1.00	1.13	2.00	1.00	1.13
9	28	30	29	26	26	27	1.08	1.15	1.12	1.00	1.00	1.04
10	31	23	23	27	22	34	1.41	1.05	1.05	1.23	1.00	1.55
11	61	32	0	37	32	76	1.91	1.00	0	1.16	1.00	2.38
12	35	59	51	46	104	31	1.13	1.90	1.65	1.48	3.35	1.00
13	138	224	77	173	95	71	1.94	3.15	1.08	2.44	1.34	1.00
14	0	196	126	0	106	91	0	2.15	1.38	0	1.16	1.00
15	14	14	14	14	14	15	1.00	1.00	1.00	1.00	1.00	1.07
16	6	6	6	6	6	7	1.00	1.00	1.00	1.00	1.00	1.17
17	14	14	14	14	14	15	1.00	1.00	1.00	1.00	1.00	1.07
18	74	122	0	0	33	-6	2.24	3.70	0	0	1.00	0
19	31	33	31	31	31	32	1.00	1.06	1.00	1.00	1.00	1.03
20	21	20	20	20	20	21	1.05	1.00	1.00	1.00	1.00	1.05
21	32	28	28	27	28	29	1.19	1.04	1.04	1.00	1.04	1.07
22	6	6	6	5	6	9	1.20	1.20	1.20	1.00	1.20	1.80
23	48	48	58	38	59	36	1.33	1.33	1.61	1.06	1.64	1.00
24	19	10	10	13	11	12	1.90	1.00	1.00	1.30	1.10	1.20
25	207	419	307	0	0	252	1.00	2.02	1.48	0	0	1.22
26	0	425	420	229	283	32	0	13.28	13.13	7.16	8.84	1.00
27	0	182	160	272	0	331	0	1.14	1.00	1.70	0	2.07
28	0	0	0	0	0	-6	0	0	0	0	0	0
29	0	0	0	0	0	192	0	0	0	0	0	1.00
30	0	0	0	0	0	310	0	0	0	0	0	1.00
31	0	0	0	0	0	0	0	0	0	0	0	0
32	33	71	62	68	182	31	1.06	2.29	2.00	2.19	5.87	1.00
33	16	16	16	16	16	17	1.00	1.00	1.00	1.00	1.00	1.06
34	9	9	8	8	8	10	1.13	1.13	1.00	1.00	1.00	1.25
35	17	17	17	17	17	18	1.00	1.00	1.00	1.00	1.00	1.06
36	0	0	0	0	0	82	0	0	0	0	0	1.00
37	93	40	76	92	57	36	2.58	1.11	2.11	2.56	1.58	1.00
38	99	55	54	56	54	54	1.83	1.02	1.00	1.04	1.00	1.00
39	69	70	63	63	63	45	1.53	1.56	1.40	1.40	1.40	1.00
40	8	7	7	7	7	9	1.14	1.00	1.00	1.00	1.00	1.29
41	53	53	78	40	56	40	1.32	1.32	1.95	1.00	1.40	1.00
42	0	0	0	0	0	437	0	0	0	0	0	1.00
43	81	0	212	0	265	44	1.84	0	4.82	0	6.02	1.00
44	0	0	0	0	0	492	0	0	0	0	0	1.00
45	0	0	0	0	0	244	0	0	0	0	0	1.00
46	0	0	0	0	0	0	0	0	0	0	0	0
47	179	0	75	101	74	97	2.42	0	1.01	1.36	1.00	1.31
48	65	52	50	50	47	47	1.38	1.11	1.06	1.06	1.00	1.00
49	27	28	20	23	19	21	1.42	1.47	1.05	1.21	1.00	1.11
50	65	52	50	51	47	38	1.71	1.37	1.32	1.34	1.24	1.00
51	0	0	0	0	0	-6	0	0	0	0	0	0
52	87	74	61	113	71	58	1.50	1.28	1.05	1.95	1.22	1.00
53	78	75	235	189	76	42	1.86	1.79	5.60	4.50	1.81	1.00
54	74	71	72	88	71	57	1.30	1.25	1.26	1.54	1.25	1.00
FAILS	14	12	13	14	12	5	MEAN	1.49	2.10	1.71	1.73	1.84	1.33

VARIABLES BADLY SCALED

METH / PROB	1	2	3	4	5	6	1	2	3	4	5	6

METH PROB	1	2	3	4	5	6		1	2	3	4	5	6
1	6	6	6	5	6	35		1.20	1.20	1.20	1.00	1.20	7.00
2	27	27	37	35	33	29		1.00	1.00	1.37	1.30	1.22	1.07
3	74	111	29	46	66	-9		2.55	3.83	1.00	1.59	2.28	0
4	160	0	110	427	138	99		1.62	0	1.11	4.31	1.39	1.00
5	28	369	0	0	75	-6		1.00	13.18	0	0	2.68	0
6	296	88	128	87	121	-6		3.40	1.01	1.47	1.00	1.39	0
7	270	110	127	112	135	-6		2.45	1.00	1.15	1.02	1.23	0
8	20	16	18	32	21	23		1.25	1.00	1.13	2.00	1.31	1.44
9	28	30	29	30	29	28		1.00	1.07	1.04	1.07	1.04	1.00
10	31	23	23	27	59	55		1.35	1.00	1.00	1.17	2.57	2.39
11	61	32	173	35	0	-6		1.91	1.00	5.41	1.09	0	0
12	35	59	51	46	33	27		1.30	2.19	1.89	1.70	1.22	1.00
13	78	224	77	77	162	46		1.70	4.87	1.67	1.67	3.52	1.00
14	347	95	0	96	78	-6		4.45	1.22	0	1.23	1.00	0
15	14	14	14	14	14	15		1.00	1.00	1.00	1.00	1.00	1.07
16	6	6	6	6	6	7		1.00	1.00	1.00	1.00	1.00	1.17
17	14	14	14	14	14	15		1.00	1.00	1.00	1.00	1.00	1.07
18	64	180	0	51	0	56		1.25	3.53	0	1.00	0	1.10
19	60	64	84	101	57	39		1.54	1.64	2.15	2.59	1.46	1.00
20	21	20	20	20	23	23		1.05	1.00	1.00	1.00	1.15	1.15
21	32	28	28	28	33	34		1.14	1.00	1.00	1.00	1.18	1.21
22	6	6	6	5	6	9		1.20	1.20	1.20	1.00	1.20	1.80
23	52	52	57	40	38	34		1.53	1.53	1.68	1.18	1.12	1.00
24	19	10	10	13	11	-9		1.90	1.00	1.00	1.30	1.10	0
25	202	0	452	0	0	-6		1.00	0	2.24	0	0	0
26	0	0	366	228	88	33		0	0	11.09	6.91	2.67	1.00
27	0	150	413	0	0	-6		0	1.00	2.75	0	0	0
28	0	0	0	0	0	-6		0	0	0	0	0	0
29	0	299	0	0	0	-6		0	1.00	0	0	0	0
30	0	0	0	0	0	-6		0	0	0	0	0	0
31	0	0	0	0	0	-6		0	0	0	0	0	0
32	40	70	0	68	44	39		1.03	1.79	0	1.74	1.13	1.00
33	16	16	16	16	16	17		1.00	1.00	1.00	1.00	1.00	1.06
34	9	9	8	8	8	10		1.13	1.13	1.00	1.00	1.00	1.25
35	17	17	17	17	17	18		1.06	1.00	1.00	1.00	1.00	1.06
36	0	0	0	0	-9	-6		0	0	0	0	0	0
37	89	62	38	79	101	37		2.41	1.68	1.03	2.14	2.73	1.00
38	121	55	54	56	109	38		3.18	1.45	1.42	1.47	2.87	1.00
39	69	70	63	64	80	50		1.38	1.40	1.26	1.28	1.60	1.00
40	9	7	7	7	7	12		1.29	1.00	1.00	1.00	1.00	1.71
41	59	53	64	70	43	39		1.51	1.36	1.64	1.79	1.10	1.00
42	0	0	477	0	240	-6		0	0	1.99	0	1.00	0
43	81	355	248	311	92	-6		1.00	4.38	3.06	3.84	1.14	0
44	0	0	0	0	-5	-6		0	0	0	0	0	0
45	0	0	0	0	0	-6		0	0	0	0	0	0
46	0	0	0	0	0	-6		0	0	0	0	0	0
47	140	0	82	102	139	-6		1.71	0	1.00	1.24	1.70	0
48	66	52	50	53	75	41		1.59	1.27	1.22	1.29	1.83	1.00
49	27	28	20	25	38	21		1.35	1.40	1.00	1.25	1.90	1.05
50	65	52	50	53	80	42		1.55	1.24	1.19	1.26	1.90	1.00
51	0	0	0	0	0	-6		0	0	0	0	0	0
52	78	0	116	81	57	67		1.37	0	2.04	1.42	1.00	1.18
53	78	75	180	194	67	44		1.77	1.70	4.09	4.41	1.52	1.00
54	74	71	72	88	82	60		1.23	1.18	1.20	1.47	1.37	1.00
FAILS	12	14	13	13	13	21	MEAN	1.55	1.81	1.77	1.65	1.50	1.33

FUNCTIONS BADLY SCALED

METH PROB	1	2	3	4	5	6	1	2	3	4	5	6
1	6	6	6	6	6	-6	1.00	1.00	1.00	1.00	1.00	0
2	51	53	64	55	57	0	1.00	1.04	1.25	1.08	1.12	0
3	56	54	30	30	31	0	1.87	1.80	1.00	1.00	1.03	0
4	0	0	82	417	473	0	0	0	1.00	5.09	5.77	0
5	0	0	0	0	0	-6	0	0	0	0	0	0

6	191	96	105	107	97	0		1.99	1.00	1.09	1.11	1.01	0
7	464	174	150	150	187	0		3.09	1.16	1.00	1.00	1.25	0
8	143	43	31	29	36	-6		4.93	1.48	1.07	1.00	1.24	0
9	33	34	33	32	29	124		1.14	1.17	1.14	1.10	1.00	4.28
10	55	28	30	37	28	200		1.96	1.00	1.07	1.32	1.00	7.14
11	0	0	59	108	51	-6		0	0	1.16	2.12	1.00	0
12	36	38	34	42	105	33		1.09	1.15	1.03	1.27	3.18	1.00
13	139	84	78	65	84	73		2.14	1.29	1.20	1.00	1.29	1.12
14	0	79	189	134	0	93		0	1.00	2.39	1.70	0	1.18
15	15	15	15	15	15	16		1.00	1.00	1.00	1.00	1.00	1.07
16	8	8	8	8	8	9		1.00	1.00	1.00	1.00	1.00	1.13
17	16	16	16	16	16	17		1.00	1.00	1.00	1.00	1.00	1.06
18	0	43	81	253	45	-6		0	1.00	1.88	5.88	1.05	0
19	69	70	79	80	88	63		1.10	1.11	1.25	1.27	1.40	1.00
20	25	24	25	25	25	26		1.04	1.00	1.04	1.04	1.04	1.08
21	55	34	34	34	34	36		1.62	1.00	1.00	1.00	1.00	1.06
22	6	7	7	6	7	11		1.00	1.17	1.17	1.00	1.17	1.83
23	54	57	68	77	73	43		1.26	1.33	1.58	1.79	1.70	1.00
24	19	11	12	13	10	12		1.90	1.10	1.20	1.30	1.00	1.20
25	0	0	0	0	0	0		0	0	0	0	0	0
26	272	0	0	0	140	57		4.77	0	0	0	2.46	1.00
27	0	0	220	0	0	-6		0	0	1.00	0	0	0
28	0	0	0	0	0	-6		0	0	0	0	0	0
29	0	0	0	0	0	-6		0	0	0	0	0	0
30	0	0	0	0	0	-6		0	0	0	0	0	0
31	0	0	0	0	0	-6		0	0	0	0	0	0
32	81	123	63	69	120	46		1.76	2.67	1.37	1.50	2.61	1.00
33	19	19	19	19	19	20		1.00	1.00	1.00	1.00	1.00	1.05
34	10	10	10	10	10	11		1.00	1.00	1.00	1.00	1.00	1.10
35	19	19	19	19	19	20		1.00	1.00	1.00	1.00	1.00	1.05
36	0	0	0	0	0	128		0	0	0	0	0	1.00
37	82	73	74	79	80	44		1.86	1.66	1.68	1.80	1.82	1.00
38	137	60	67	65	62	37		3.70	1.62	1.81	1.76	1.68	1.00
39	87	106	93	95	90	47		1.85	2.26	1.98	2.02	1.91	1.00
40	8	8	8	8	9	17		1.00	1.00	1.00	1.00	1.13	2.13
41	71	61	80	47	70	49		1.51	1.30	1.70	1.00	1.49	1.04
42	0	0	0	0	0	0		0	0	0	0	0	0
43	0	0	0	448	0	0		0	0	0	1.00	0	0
44	0	0	0	0	0	0		0	0	0	0	0	0
45	0	0	0	0	0	364		0	0	0	0	0	1.00
46	0	0	0	0	0	-6		0	0	0	0	0	0
47	168	0	103	147	106	49		3.43	0	2.10	3.00	2.16	1.00
48	75	58	57	53	52	82		1.44	1.12	1.10	1.02	1.00	1.58
49	28	30	22	26	21	23		1.33	1.43	1.05	1.24	1.00	1.10
50	88	63	58	57	57	39		2.26	1.62	1.49	1.46	1.46	1.00
51	0	0	0	0	0	-6		0	0	0	0	0	0
52	149	97	114	107	72	-6		2.07	1.35	1.58	1.49	1.00	0
53	77	74	63	65	63	45		1.71	1.64	1.40	1.44	1.40	1.00
54	76	74	76	77	74	78		1.03	1.00	1.03	1.04	1.00	1.05
FAILS	18	18	14	14	15	22	MEAN	1.77	1.26	1.27	1.50	1.44	1.41

FINAL STATISTICS

FAILS	44	44	40	41	40	48	MEAN	1.60	1.75	1.59	1.63	1.60	1.35

U S I N G T H E I N T E R N A L S C A L I N G

UNSCALED SET OF PROBLEMS

METH PROB	1	2	3	4	5	6		1	2	3	4	5	6
1	6	6	6	7	6	23		1.00	1.00	1.00	1.17	1.00	3.83
2	27	27	34	35	25	41		1.08	1.08	1.36	1.40	1.00	1.64
3	73	62	29	66	37	179		2.52	2.14	1.00	2.28	1.28	6.17
4	0	280	118	433	118	96		0	2.92	1.23	4.51	1.23	1.00
5	28	62	86	0	0	27		1.04	2.30	3.19	0	0	1.00

METH PROB	1	2	3	4	5	6		1	2	3	4	5	6
1	6	6	6	7	6	23		1.00	1.00	1.00	1.17	1.00	3.83
2	27	27	34	35	25	41		1.08	1.08	1.36	1.40	1.00	1.64
3	73	62	29	66	37	179		2.52	2.14	1.00	2.28	1.28	6.17
4	0	280	118	433	118	96		0	2.92	1.23	4.51	1.23	1.00
5	28	62	86	0	0	27		1.04	2.30	3.19	0	0	1.00
6	164	76	102	75	86	91		2.19	1.01	1.36	1.00	1.15	1.21
7	154	107	107	120	99	143		1.56	1.08	1.08	1.21	1.00	1.44
8	20	16	18	22	16	18		1.25	1.00	1.13	1.38	1.00	1.13
9	28	30	29	25	30	27		1.12	1.20	1.16	1.00	1.20	1.08
10	29	23	23	29	21	34		1.38	1.10	1.10	1.38	1.00	1.62
11	0	107	0	84	108	76		0	1.41	0	1.11	1.42	1.00
12	35	36	33	34	35	31		1.13	1.16	1.06	1.10	1.13	1.00
13	82	82	75	75	75	71		1.15	1.15	1.06	1.06	1.06	1.00
14	103	156	187	100	186	91		1.13	1.71	2.05	1.10	2.04	1.00
15	14	14	14	14	14	15		1.00	1.00	1.00	1.00	1.00	1.07
16	6	6	6	6	6	7		1.00	1.00	1.00	1.00	1.00	1.17
17	14	14	14	14	14	15		1.00	1.00	1.00	1.00	1.00	1.07
18	76	0	0	0	47	-6		1.62	0	0	0	1.00	0
19	31	31	31	31	31	32		1.00	1.00	1.00	1.00	1.00	1.03
20	21	20	20	20	20	21		1.05	1.00	1.00	1.00	1.00	1.05
21	32	28	28	27	28	29		1.19	1.04	1.04	1.00	1.04	1.07
22	6	6	6	6	6	9		1.00	1.00	1.00	1.00	1.00	1.50
23	49	49	50	38	48	36		1.36	1.36	1.39	1.06	1.33	1.00
24	19	10	10	13	37	12		1.90	1.00	1.00	1.30	3.70	1.20
25	0	0	0	0	0	252		0	0	0	0	0	1.00
26	71	0	0	273	0	32		2.22	0	0	8.53	0	1.00
27	0	426	227	227	125	331		0	3.41	1.82	1.82	1.00	2.65
28	0	0	0	0	0	-6		0	0	0	0	0	0
29	0	176	0	468	0	192		0	1.00	0	2.66	0	1.09
30	0	316	0	0	0	310		0	1.02	0	0	0	1.00
31	0	0	0	0	0	0		0	0	0	0	0	0
32	92	66	79	58	69	31		2.97	2.13	2.55	1.87	2.23	1.00
33	16	16	16	16	16	17		1.00	1.00	1.00	1.00	1.00	1.06
34	9	9	8	8	8	10		1.13	1.13	1.00	1.00	1.00	1.25
35	17	17	17	17	17	18		1.00	1.00	1.00	1.00	1.00	1.06
36	0	0	0	0	0	82		0	0	0	0	0	1.00
37	97	39	96	64	37	36		2.69	1.08	2.67	1.78	1.03	1.00
38	114	55	54	56	57	54		2.11	1.02	1.00	1.04	1.06	1.00
39	69	70	63	65	63	45		1.53	1.56	1.40	1.44	1.40	1.00
40	8	7	7	7	7	9		1.14	1.00	1.00	1.00	1.00	1.29
41	58	54	56	61	55	40		1.45	1.35	1.40	1.52	1.38	1.00
42	0	0	423	458	463	437		0	0	1.00	1.08	1.09	1.03
43	53	157	153	190	0	44		1.20	3.57	3.48	4.32	0	1.00
44	0	283	0	477	0	492		0	1.00	0	1.69	0	1.74
45	0	0	0	0	0	244		0	0	0	0	0	1.00
46	0	0	0	0	0	0		0	0	0	0	0	0
47	62	144	96	130	99	97		1.00	2.32	1.55	2.10	1.60	1.56
48	65	53	50	54	45	47		1.44	1.18	1.11	1.20	1.00	1.04
49	27	28	20	23	19	21		1.42	1.47	1.05	1.21	1.00	1.11
50	65	54	50	53	44	38		1.71	1.42	1.32	1.39	1.16	1.00
51	0	0	0	0	0	-6		0	0	0	0	0	0
52	120	113	61	66	57	58		2.11	1.98	1.07	1.16	1.00	1.02
53	91	75	179	208	134	42		2.17	1.79	4.26	4.95	3.19	1.00
54	86	71	72	91	71	57		1.51	1.25	1.26	1.60	1.25	1.00
FAILS	14	10	13	10	13	5	MEAN	1.46	1.42	1.42	1.69	1.27	1.31

VARIABLES BADLY SCALED

METH PROB	1	2	3	4	5	6		1	2	3	4	5	6
1	6	6	6	6	6	35		1.00	1.00	1.00	1.00	1.00	5.83
2	27	27	34	37	25	29		1.08	1.08	1.36	1.48	1.00	1.16
3	76	47	29	51	37	-9		2.62	1.62	1.00	1.76	1.28	0
4	144	249	126	115	118	99		1.45	2.52	1.27	1.16	1.19	1.00
5	28	62	147	64	0	-6		1.00	2.21	5.25	2.29	0	0

6	94	84	84	96	146	-6		1.12	1.00	1.00	1.14	1.74	0
7	389	146	154	114	139	-6		3.41	1.28	1.35	1.00	1.22	0
8	20	16	18	20	16	23		1.25	1.00	1.13	1.25	1.00	1.44
9	28	30	29	27	30	28		1.04	1.11	1.07	1.00	1.11	1.04
10	29	23	23	30	21	55		1.38	1.10	1.10	1.43	1.00	2.62
11	0	112	0	74	131	-6		0	1.51	0	1.00	1.77	0

FAILS	14	10	13	10	13	5	MEAN	1.46	1.42	1.42	1.69	1.27	1.31

VARIABLES BADLY SCALED

METH PROB	1	2	3	4	5	6	1	2	3	4	5	6
1	6	6	6	6	6	35	1.00	1.00	1.00	1.00	1.00	5.83
2	27	27	34	37	25	29	1.08	1.08	1.36	1.48	1.00	1.16
3	76	47	29	51	37	-9	2.62	1.62	1.00	1.76	1.28	0
4	144	249	126	115	118	99	1.45	2.52	1.27	1.16	1.19	1.00
5	28	62	147	64	0	-6	1.00	2.21	5.25	2.29	0	0
6	94	84	84	96	146	-6	1.12	1.00	1.00	1.14	1.74	0
7	389	146	154	114	139	-6	3.41	1.28	1.35	1.00	1.22	0
8	20	16	18	20	16	23	1.25	1.00	1.13	1.25	1.00	1.44
9	28	30	29	27	30	28	1.04	1.11	1.07	1.00	1.11	1.04
10	29	23	23	30	21	55	1.38	1.10	1.10	1.43	1.00	2.62
11	0	112	0	74	131	-6	0	1.51	0	1.00	1.77	0
12	35	36	33	37	35	27	1.30	1.33	1.22	1.37	1.30	1.00
13	82	82	75	75	75	46	1.78	1.78	1.63	1.63	1.63	1.00
14	103	243	194	0	82	-6	1.26	2.96	2.37	0	1.00	0
15	14	14	14	14	14	15	1.00	1.00	1.00	1.00	1.00	1.07
16	6	6	6	6	6	7	1.00	1.00	1.00	1.00	1.00	1.17
17	14	14	14	14	14	15	1.00	1.00	1.00	1.00	1.00	1.07
18	91	0	0	0	47	56	1.94	0	0	0	1.00	1.19
19	55	64	80	92	91	39	1.41	1.64	2.05	2.36	2.33	1.00
20	21	20	20	20	20	23	1.05	1.00	1.00	1.00	1.00	1.15
21	32	28	28	28	28	34	1.14	1.00	1.00	1.00	1.00	1.21
22	6	6	6	6	6	9	1.00	1.00	1.00	1.00	1.00	1.50
23	49	49	51	60	49	34	1.44	1.44	1.50	1.76	1.44	1.00
24	19	13	10	12	38	-9	1.90	1.30	1.00	1.20	3.80	0
25	0	0	392	0	369	-6	0	0	1.06	0	1.00	0
26	71	429	0	315	0	33	2.15	13.00	0	9.55	0	1.00
27	0	285	332	247	0	-6	0	1.15	1.34	1.00	0	0
28	0	0	0	0	0	-6	0	0	0	0	0	0
29	0	249	0	0	0	-6	0	1.00	0	0	0	0
30	0	0	0	0	0	-6	0	0	0	0	0	0
31	0	0	0	0	0	-6	0	0	0	0	0	0
32	35	65	37	49	44	39	1.00	1.86	1.06	1.40	1.26	1.11
33	16	16	16	16	16	17	1.00	1.00	1.00	1.00	1.00	1.06
34	9	9	8	8	8	10	1.13	1.13	1.00	1.00	1.00	1.25
35	17	17	17	17	17	18	1.00	1.00	1.00	1.00	1.00	1.06
36	0	0	-9	-9	0	-6	0	0	0	0	0	0
37	206	75	39	78	40	37	5.57	2.03	1.05	2.11	1.08	1.00
38	116	55	54	56	57	38	3.05	1.45	1.42	1.47	1.50	1.00
39	69	70	63	63	63	50	1.38	1.40	1.26	1.26	1.26	1.00
40	8	7	7	7	7	12	1.14	1.00	1.00	1.00	1.00	1.71
41	58	54	56	61	55	39	1.49	1.38	1.44	1.56	1.41	1.00
42	0	0	0	0	348	-6	0	0	0	0	1.00	0
43	62	134	0	162	0	-6	1.00	2.16	0	2.61	0	0
44	0	247	0	262	0	-6	0	1.00	0	1.06	0	0
45	0	0	0	0	0	-6	0	0	0	0	0	0
46	0	0	0	0	0	-6	0	0	0	0	0	0
47	72	126	102	137	117	-6	1.00	1.75	1.42	1.90	1.63	0
48	65	53	50	50	45	41	1.59	1.29	1.22	1.22	1.10	1.00
49	27	28	20	25	19	21	1.42	1.47	1.05	1.32	1.00	1.11
50	65	54	50	50	44	42	1.55	1.29	1.19	1.19	1.05	1.00
51	0	0	0	0	0	-6	0	0	0	0	0	0
52	95	88	103	193	116	67	1.42	1.31	1.54	2.88	1.73	1.00
53	91	75	195	217	111	44	2.07	1.70	4.43	4.93	2.52	1.00
54	86	71	72	73	71	60	1.43	1.18	1.20	1.22	1.18	1.00
FAILS	13	10	14	12	13	21	MEAN 1.54	1.65	1.40	1.66	1.31	1.30

FUNCTIONS BADLY SCALED

METH PROB	1	2	3	4	5	6	1	2	3	4	5	6
1	6	6	6	6	6	-6	1.00	1.00	1.00	1.00	1.00	0
2	56	56	64	47	54	0	1.19	1.19	1.36	1.00	1.15	0
3	108	65	30	67	30	0	3.60	2.17	1.00	2.23	1.00	0
4	144	99	194	85	88	0	1.69	1.16	2.28	1.00	1.04	0
5	36	0	-2	0	0	-6	1.00	0	0	0	0	0
6	172	95	102	85	99	0	2.02	1.12	1.20	1.00	1.16	0
7	151	175	155	136	158	0	1.11	1.29	1.14	1.00	1.16	0
8	24	20	22	21	20	-6	1.20	1.00	1.10	1.05	1.00	0
9	33	34	33	32	37	124	1.03	1.06	1.03	1.00	1.16	3.88
10	36	27	28	30	25	200	1.44	1.08	1.12	1.20	1.00	8.00
11	0	110	110	62	101	-6	0	1.77	1.77	1.00	1.63	0
12	36	38	34	39	36	33	1.09	1.15	1.03	1.18	1.09	1.00

METH PROB	1	2	3	4	5	6		1	2	3	4	5	6
1	6	6	6	6	6	-6		1.00	1.00	1.00	1.00	1.00	0
2	56	56	64	47	54	0		1.19	1.19	1.36	1.00	1.15	0
3	108	65	30	67	30	0		3.60	2.17	1.00	2.23	1.00	0
4	144	99	194	85	88	0		1.69	1.16	2.28	1.00	1.04	0
5	36	0	-2	0	0	-6		1.00	0	0	0	0	0
6	172	95	102	85	99	0		2.02	1.12	1.20	1.00	1.16	0
7	151	175	155	136	158	0		1.11	1.29	1.14	1.00	1.16	0
8	24	20	22	21	20	-6		1.20	1.00	1.10	1.05	1.00	0
9	33	34	33	32	37	124		1.03	1.06	1.03	1.00	1.16	3.88
10	36	27	28	30	25	200		1.44	1.08	1.12	1.20	1.00	8.00
11	0	110	110	62	101	-6		0	1.77	1.77	1.00	1.63	0
12	36	38	34	39	36	33		1.09	1.15	1.03	1.18	1.09	1.00
13	84	84	77	76	65	73		1.29	1.29	1.18	1.17	1.00	1.12
14	78	79	188	320	251	93		1.00	1.01	2.41	4.10	3.22	1.19
15	15	15	15	15	15	16		1.00	1.00	1.00	1.00	1.00	1.07
16	8	8	8	8	8	9		1.00	1.00	1.00	1.00	1.00	1.13
17	16	16	16	16	16	17		1.00	1.00	1.00	1.00	1.00	1.06
18	88	64	67	70	52	-6		1.69	1.23	1.29	1.35	1.00	0
19	68	68	67	68	67	63		1.08	1.08	1.06	1.08	1.06	1.00
20	25	24	25	25	21	26		1.19	1.14	1.19	1.19	1.00	1.24
21	42	34	34	34	35	36		1.24	1.00	1.00	1.00	1.03	1.06
22	6	7	22	6	6	11		1.00	1.17	3.67	1.00	1.00	1.83
23	61	61	70	63	59	43		1.42	1.42	1.63	1.47	1.37	1.00
24	19	14	12	13	11	12		1.73	1.27	1.09	1.18	1.00	1.09
25	0	0	0	0	0	0		0	0	0	0	0	0
26	139	121	0	133	446	57		2.44	2.12	0	2.33	7.82	1.00
27	270	438	280	242	352	-6		1.12	1.81	1.16	1.00	1.45	0
28	0	0	0	0	0	-6		0	0	0	0	0	0
29	392	300	0	304	0	-6		1.31	1.00	0	1.01	0	0
30	0	444	0	0	0	-6		0	1.00	0	0	0	0
31	0	248	0	0	0	-6		0	1.00	0	0	0	0
32	71	103	86	103	86	46		1.54	2.24	1.87	2.24	1.87	1.00
33	19	19	19	19	18	20		1.06	1.06	1.06	1.06	1.00	1.11
34	10	10	10	10	10	11		1.00	1.00	1.00	1.00	1.00	1.10
35	19	19	19	19	19	20		1.00	1.00	1.00	1.00	1.00	1.05
36	0	0	0	0	0	128		0	0	0	0	0	1.00
37	88	79	73	77	69	44		2.00	1.80	1.66	1.75	1.57	1.00
38	161	64	63	65	68	37		4.35	1.73	1.70	1.76	1.84	1.00
39	73	73	66	80	114	47		1.55	1.55	1.40	1.70	2.43	1.00
40	8	8	8	8	8	17		00	1.00	1.00	1.00	1.00	2.13
41	66	66	78	70	63	49		35	1.35	1.59	1.43	1.29	1.00
42	0	0	433	0	0	0		0	0	1.00	0	0	0
43	56	167	0	229	354	0		00	2.98	0	4.09	6.32	0
44	0	283	0	0	0	0		0	1.00	0	0	0	0
45	0	0	0	0	0	364		0	0	0	0	0	1.00
46	0	0	0	0	0	-6		0	0	0	0	0	0
47	-5	128	100	106	135	49		0	2.61	2.04	2.16	2.76	1.00
48	76	58	57	57	59	82		1.33	1.02	1.00	1.00	1.04	1.44
49	28	30	22	25	24	23		1.27	1.36	1.00	1.14	1.09	1.05
50	101	66	62	54	80	39		2.59	1.69	1.59	1.38	2.05	1.00
51	0	0	0	0	0	-6		0	0	0	0	0	0
52	134	89	127	151	98	-6		1.51	1.00	1.43	1.70	1.10	0
53	93	76	241	184	64	45		2.07	1.69	5.36	4.09	1.42	1.00
54	90	74	76	76	92	78		1.22	1.00	1.03	1.03	1.24	1.05
FAILS	12	8	13	11	12	22	MEAN	1.47	1.34	1.45	1.47	58	1.42

FINAL STATISTICS

FAILS	39	28	40	33	38	48	MEAN	1.49	1.47	1.42	1.60	1.39	1.34

****
****
**
**
** D E T A I L E D R E S U L T S F O R T H E **
**
** G E N E R A L S U B S E T **
**
**
****
****

W I T H O U T U S I N G T H E I N T E R N A L S C A L I N G

****

WITHOUT USING THE INTERNAL SCALING

METH PROB	1	2	3	4	5		1	2	3	4	5
1	6	6	6	5	6		1.20	1.20	1.20	1.00	1.20
2	27	27	37	34	25		1.08	1.08	1.48	1.36	1.00
3	74	98	29	46	66		2.55	3.38	1.00	1.59	2.28
4	363	129	103	125	99		3.67	1.30	1.04	1.26	1.00
5	113	103	145	130	111		1.10	1.00	1.41	1.26	1.08
6	20	16	18	32	16		1.25	1.00	1.13	2.00	1.00
7	28	30	29	26	26		1.08	1.15	1.12	1.00	1.00
8	31	23	23	27	22		1.41	1.05	1.05	1.23	1.00
9	36	59	51	46	104		1.00	1.69	1.46	1.31	2.97
10	138	224	77	173	95		1.79	2.91	1.00	2.25	1.23
11	14	14	14	14	14		1.00	1.00	1.00	1.00	1.00
12	6	6	6	6	6		1.00	1.00	1.00	1.00	1.00
13	14	14	14	14	14		1.00	1.00	1.00	1.00	1.00
14	31	33	31	31	31		1.00	1.06	1.00	1.00	1.00
15	21	20	20	20	20		1.05	1.00	1.00	1.00	1.00
16	32	28	28	27	28		1.19	1.04	1.04	1.00	1.04
FAILS	0	0	0	0	0	MEAN	1.40	1.37	1.12	1.27	1.24

METH PROB	1	2	3	4	5		1	2	3	4	5
1	6	6	6	5	6		1.20	1.20	1.20	1.00	1.20
2	27	27	37	37	33		1.00	1.00	1.37	1.37	1.22
3	75	70	29	46	66		2.59	2.41	1.00	1.59	2.28
4	389	102	117	96	86		4.52	1.19	1.36	1.12	1.00
5	273	228	108	118	147		2.53	2.11	1.00	1.09	1.36
6	20	16	18	32	20		1.25	1.00	1.13	2.00	1.25
7	28	30	29	29	29		1.00	1.07	1.04	1.04	1.04
8	31	23	23	27	55		1.35	1.00	1.00	1.17	2.39
9	35	59	51	46	33		1.06	1.79	1.55	1.39	1.00
10	138	224	77	77	88		1.79	2.91	1.00	1.00	1.14
11	14	14	14	14	14		1.00	1.00	1.00	1.00	1.00
12	6	6	6	6	6		1.00	1.00	1.00	1.00	1.00
13	14	14	14	14	14		1.00	1.00	1.00	1.00	1.00
14	59	76	63	84	62		1.00	1.29	1.07	1.42	1.05
15	21	20	20	20	23		1.05	1.00	1.00	1.00	1.15
16	32	28	28	28	34		1.14	1.00	1.00	1.00	1.21
FAILS	0	0	0	0	0	MEAN	1.53	1.37	1.11	1.20	1.27

METH PROB	1	2	3	4	5		1	2	3	4	5
1	6	6	6	5	6		1.20	1.20	1.20	1.00	1.20
2	136	67	63	42	30		4.53	2.23	2.10	1.40	1.00
3	85	59	28	-9	65		3.04	2.11	1.00	0	2.32
4	0	164	138	121	282		0	1.36	1.14	1.00	2.33
5	201	321	155	215	0		1.30	2.07	1.00	1.39	0
6	48	24	21	26	30		2.29	1.14	1.00	1.24	1.43
7	0	33	31	51	0		0	1.06	1.00	1.65	0
8	0	74	104	46	0		0	1.61	2.26	1.00	0
9	35	59	51	46	33		1.06	1.79	1.55	1.39	1.00
10	78	224	77	77	59		1.32	3.80	1.31	1.31	1.00
11	15	14	15	15	15		1.07	1.00	1.07	1.07	1.07
12	7	7	7	7	7		1.00	1.00	1.00	1.00	1.00
13	15	14	15	15	15		1.07	1.00	1.07	1.07	1.07
14	35	62	61	35	35		1.00	1.77	1.74	1.00	1.00
15	21	20	20	20	21		1.05	1.00	1.00	1.00	1.05
16	33	27	27	27	32		1.22	1.00	1.00	1.00	1.19
FAILS	3	0	0	1	3	MEAN	1.63	1.57	1.28	1.17	1.28

METH PROB	1	2	3	4	5		1	2	3	4	5
1	0	0	24	8	0		0	0	3.00	1.00	0
2	65	53	93	102	59		1.23	1.00	1.75	1.92	1.11
3	-9	-9	-9	-9	-9		0	0	0	0	0
4	0	0	96	71	0		0	0	1.35	1.00	0
5	0	0	272	260	0		0	0	1.05	1.00	0

METH PROB	1	2	3	4	5		1	2	3	4	5
1	0	0	24	8	0		0	0	3.00	1.00	0
2	65	53	93	102	59		1.23	1.00	1.75	1.92	1.11
3	-9	-9	-9	-9	-9		0	0	0	0	0
4	0	0	96	71	0		0	0	1.35	1.00	0
5	0	0	272	260	0		0	0	1.05	1.00	0
6	0	0	0	0	0		0	0	0	0	0
7	0	0	0	0	0		0	0	0	0	0
8	0	0	0	0	0		0	0	0	0	0
9	35	59	51	46	33		1.06	1.79	1.55	1.39	1.00
10	131	55	0	0	0		2.38	1.00	0	0	0
11	167	138	22	22	262		7.59	6.27	1.00	1.00	11.91
12	147	97	12	10	323		14.70	9.70	1.20	1.00	32.30
13	277	134	22	22	201		12.59	6.09	1.00	1.00	9.14
14	-6	397	134	138	243		0	2.96	1.00	1.03	1.81
15	96	58	33	61	0		2.91	1.76	1.00	1.85	0
16	0	0	230	204	0		0	0	1.13	1.00	0
FAILS	9	8	5	5	10	MEAN	6.07	3.82	1.37	1.20	9.55

METH PROB	1	2	3	4	5		1	2	3	4	5
1	0	0	74	11	0		0	0	6.73	1.00	0
2	0	0	194	109	0		0	0	1.78	1.00	0
3	-9	-9	-9	-9	-9		0	0	0	0	0
4	-6	0	0	0	0		0	0	0	0	0
5	0	0	0	0	0		0	0	0	0	0
6	0	0	0	0	0		0	0	0	0	0
7	0	0	0	0	0		0	0	0	0	0
8	0	0	0	0	0		0	0	0	0	0
9	128	59	74	54	37		3.88	1.79	2.24	1.64	1.00
10	130	81	303	80	86		1.63	1.01	3.79	1.00	1.07
11	0	0	0	0	0		0	0	0	0	0
12	-6	-6	13	-2	-6		0	0	1.00	0	0
13	0	0	0	0	0		0	0	0	0	0
14	0	0	0	0	0		0	0	0	0	0
15	0	0	0	0	0		0	0	0	0	0
16	0	0	0	0	0		0	0	0	0	0
FAILS	14	14	11	12	14	MEAN	2.75	1.40	3.11	1.16	1.04

FINAL STATISTICS

FAILS	26	22	16	18	27	MEAN	2.15	1.76	1.35	1.21	2.19

U S I N G T H E I N T E R N A L S C A L I N G

METH PROB	1	2	3	4	5		1	2	3	4	5
1	6	6	6	7	6		1.00	1.00	1.00	1.17	1.00
2	27	27	34	35	25		1.08	1.08	1.36	1.40	1.00
3	73	62	29	66	37		2.52	2.14	1.00	2.28	1.28
4	164	76	102	75	86		2.19	1.01	1.36	1.00	1.15
5	154	107	107	120	99		1.56	1.08	1.08	1.21	1.00
6	20	16	18	22	16		1.25	1.00	1.13	1.38	1.00
7	28	30	29	25	30		1.12	1.20	1.16	1.00	1.20
8	29	23	23	29	21		1.38	1.10	1.10	1.38	1.00
9	35	36	33	34	35		1.06	1.09	1.00	1.03	1.06
10	82	82	75	75	75		1.09	1.09	1.00	1.00	1.00
11	14	14	14	14	14		1.00	1.00	1.00	1.00	1.00
12	6	6	6	6	6		1.00	1.00	1.00	1.00	1.00
13	14	14	14	14	14		1.00	1.00	1.00	1.00	1.00
14	31	31	31	31	31		1.00	1.00	1.00	1.00	1.00
15	21	20	20	20	20		1.05	1.00	1.00	1.00	1.00
16	32	28	28	27	28		1.19	1.04	1.04	1.00	1.04
FAILS	0	0	0	0	0	MEAN	1.28	1.11	1.08	1.18	1.05

METH PROB	1	2	3	4	5		1	2	3	4	5
1	6	6	6	6	6		1.00	1.00	1.00	1.00	1.00
2	27	27	34	37	25		1.08	1.08	1.36	1.48	1.00
3	76	73	29	68	37		2.62	2.52	1.00	2.34	1.28

METH PROB	1	2	3	4	5		1	2	3	4	5
1	6	6	6	6	6		1.00	1.00	1.00	1.00	1.00
2	27	27	34	37	25		1.08	1.08	1.36	1.48	1.00
3	76	73	29	68	37		2.62	2.52	1.00	2.34	1.28
4	211	76	139	101	127		2.78	1.00	1.83	1.33	1.67
5	122	112	98	106	0		1.24	1.14	1.00	1.08	0
6	20	16	18	32	16		1.25	1.00	1.13	2.00	1.00
7	28	30	29	29	30		1.00	1.07	1.04	1.04	1.07
8	29	23	23	28	21		1.38	1.10	1.10	1.33	1.00
9	35	36	33	40	35		1.06	1.09	1.00	1.21	1.06
10	137	82	75	75	75		1.83	1.09	1.00	1.00	1.00
11	14	14	14	14	14		1.00	1.00	1.00	1.00	1.00
12	6	6	6	6	6		1.00	1.00	1.00	1.00	1.00
13	14	14	14	14	14		1.00	1.00	1.00	1.00	1.00
14	57	61	85	84	62		1.00	1.07	1.49	1.47	1.09
15	21	20	20	20	20		1.05	1.00	1.00	1.00	1.00
16	32	28	28	28	28		1.14	1.00	1.00	1.00	1.00
FAILS	0	0	0	0	1	MEAN	1.34	1.14	1.12	1.27	1.08

METH PROB	1	2	3	4	5		1	2	3	4	5
1	6	6	6	5	6		1.20	1.20	1.20	1.00	1.20
2	52	55	56	74	78		1.00	1.06	1.08	1.42	1.50
3	71	70	28	60	37		2.54	2.50	1.00	2.14	1.32
4	0	137	185	83	138		0	1.65	2.23	1.00	1.66
5	308	265	101	245	113		3.05	2.62	1.00	2.43	1.12
6	48	24	21	27	21		2.29	1.14	1.00	1.29	1.00
7	72	33	31	82	32		2.32	1.06	1.00	2.65	1.03
8	0	0	184	102	0		0	0	1.80	1.00	0
9	35	36	33	38	35		1.06	1.09	1.00	1.15	1.06
10	82	82	75	75	75		1.09	1.09	1.00	1.00	1.00
11	15	14	15	15	15		1.07	1.00	1.07	1.07	1.07
12	7	7	7	7	7		1.00	1.00	1.00	1.00	1.00
13	15	14	15	15	15		1.07	1.00	1.07	1.07	1.07
14	68	39	0	0	80		1.74	1.00	0	0	2.05
15	21	20	20	20	20		1.05	1.00	1.00	1.00	1.00
16	33	27	27	27	26		1.27	1.04	1.04	1.04	1.00
FAILS	2	1	1	1	1	MEAN	1.55	1.30	1.17	1.35	1.21

METH PROB	1	2	3	4	5		1	2	3	4	5
1	9	8	9	8	9		1.13	1.00	1.13	1.00	1.13
2	78	86	196	92	117		1.00	1.10	2.51	1.18	1.50
3	-9	-9	-9	-9	-9		0	0	0	0	0
4	0	0	116	305	79		0	0	1.47	3.86	1.00
5	0	160	0	129	0		0	1.24	0	1.00	0
6	0	0	0	0	0		0	0	0	0	0
7	0	0	0	0	0		0	0	0	0	0
8	0	0	0	0	0		0	0	0	0	0
9	35	36	33	37	35		1.06	1.09	1.00	1.12	1.06
10	137	82	75	75	75		1.83	1.09	1.00	1.00	1.00
11	197	138	22	23	21		9.38	6.57	1.05	1.10	1.00
12	184	141	12	12	11		16.73	12.82	1.09	1.09	1.00
13	178	130	22	21	22		8.48	6.19	1.05	1.00	1.05
14	127	391	135	79	149		1.61	4.95	1.71	1.00	1.89
15	72	59	33	61	70		2.18	1.79	1.00	1.85	2.12
16	0	0	0	0	0		0	0	0	0	0
FAILS	7	6	6	5	6	MEAN	4.62	3.78	1.30	1.38	1.27

METH PROB	1	2	3	4	5		1	2	3	4	5
1	13	14	13	13	13		1.00	1.08	1.00	1.00	1.00
2	0	0	361	0	112		0	0	3.22	0	1.00
3	-9	-9	-9	-9	-9		0	0	0	0	0
4	0	0	0	0	0		0	0	0	0	0
5	0	0	0	0	0		0	0	0	0	0
6	0	0	0	0	0		0	0	0	0	0
7	0	0	0	0	0		0	0	0	0	0
8	0	0	0	0	0		0	0	0	0	0
9	32	33	34	40	35		1.00	1.03	1.06	1.25	1.09
10	0	55	95	120	172		0	1.00	1.73	2.18	3.13
11	0	0	0	0	177		0	0	0	0	1.00
12	-6	245	13	78	13		0	18.85	1.00	6.00	1.00
13	0	0	0	0	158		0	0	0	0	1.00

METH PROB	1	2	3	4	5		1	2	3	4	5
1	13	14	13	13	13		1.00	1.08	1.00	1.00	1.00
2	0	0	361	0	112		0	0	3.22	0	1.00
3	-9	-9	-9	-9	-9		0	0	0	0	0
4	0	0	0	0	0		0	0	0	0	0
5	0	0	0	0	0		0	0	0	0	0
6	0	0	0	0	0		0	0	0	0	0
7	0	0	0	0	0		0	0	0	0	0
8	0	0	0	0	0		0	0	0	0	0
9	32	33	34	40	35		1.00	1.03	1.06	1.25	1.09
10	0	55	95	120	172		0	1.00	1.73	2.18	3.13
11	0	0	0	0	177		0	0	0	0	1.00
12	-6	245	13	78	13		0	18.85	1.00	6.00	1.00
13	0	0	0	0	158		0	0	0	0	1.00
14	0	0	0	0	242		0	0	0	0	1.00
15	0	0	0	86	0		0	0	0	1.00	0
16	0	0	0	0	0		0	0	0	0	0
FAILS	14	12	11	11	8	MEAN	1.00	5.49	1.60	2.29	1.28
FINAL STATISTICS											
FAILS	23	19	18	17	16	MEAN	1.91	1.89	1.19	1.37	1.16

****
****
**
**
** DETAILED RESULTS FOR THE
**
** CHEMICAL EQUILIBRIUM SET
**
**
****
****

WITHOUT USING THE INTERNAL SCALING

METH PROB	1	2	3	4	5	6		1	2	3	4	5	6
1	0	0	0	0	0	0		0	0	0	0	0	0
2	5	5	5	5	5	0		1.00	1.00	1.00	1.00	1.00	0
3	170	119	0	119	0	0		1.43	1.00	0	1.00	0	0
4	62	70	97	113	73	-6		1.00	1.13	1.56	1.82	1.18	0
5	14	14	13	13	22	-6		1.08	1.08	1.00	1.00	1.69	0
6	68	84	65	62	64	-6		1.10	1.35	1.05	1.00	1.03	0
7	0	0	0	0	0	33		0	0	0	0	0	1.00
8	0	88	64	90	434	25		0	3.52	2.56	3.60	17.36	1.00
9	59	48	41	35	36	-9		1.69	1.37	1.17	1.00	1.03	0
10	54	71	35	40	34	39		1.59	2.09	1.03	1.18	1.00	1.15
11	132	0	111	0	116	-9		1.19	0	1.00	0	1.05	0
12	154	133	176	105	117	-9		1.47	1.27	1.68	1.00	1.11	0
FAILS	3	3	3	3	3	9	MEAN	1.28	1.53	1.34	1.40	2.94	1.05
FINAL STATISTICS													
FAILS	3	3	3	3	3	9	MEAN	1.28	1.53	1.34	1.40	2.94	1.05

USING THE INTERNAL SCALING

METH PROB	1	2	3	4	5	6	1	2	3	4	5	6
1	14	14	120	14	120	0	1.00	1.00	8.57	1.00	8.57	0
2	5	5	5	5	5	0	1.00	1.00	1.00	1.00	1.00	0
3	71	67	113	68	90	0	1.06	1.00	1.69	1.01	1.34	0
4	67	80	65	84	81	-6	1.03	1.23	1.00	1.29	1.25	0
5	16	13	12	12	12	-6	1.33	1.08	1.00	1.00	1.00	0
6	67	84	72	67	66	-6	1.02	1.27	1.09	1.02	1.00	0
7	-6	0	0	-6	-6	33	0	0	0	0	0	1.00
8	276	79	72	59	71	25	11.04	3.16	2.88	2.36	2.84	1.00
9	59	47	41	35	35	-9	1.69	1.34	1.17	1.00	1.00	0

METH PROB	1	2	3	4	5	6		1	2	3	4	5	6
1	14	14	120	14	120	0		1.00	1.00	8.57	1.00	8.57	0
2	5	5	5	5	5	0		1.00	1.00	1.00	1.00	1.00	0
3	71	67	113	68	90	0		1.06	1.00	1.69	1.01	1.34	0
4	67	80	65	84	81	-6		1.03	1.23	1.00	1.29	1.25	0
5	16	13	12	12	12	-6		1.33	1.08	1.00	1.00	1.00	0
6	67	84	72	67	66	-6		1.02	1.27	1.09	1.02	1.00	0
7	-6	0	0	-6	-6	33		0	0	0	0	0	1.00
8	276	79	72	59	71	25		11.04	3.16	2.88	2.36	2.84	1.00
9	59	47	41	65	35	-9		1.69	1.34	1.17	1.86	1.00	0
10	54	61	35	40	35	39		1.54	1.74	1.00	1.14	1.00	1.11
11	185	0	111	156	108	-9		1.71	0	1.03	1.44	1.00	0
12	133	100	158	108	119	-9		1.33	1.00	1.58	1.08	1.19	0
FAILS	1	2	1	1	1	9	MEAN	2.16	1.38	2.00	1.29	1.93	1.04

FINAL STATISTICS

FAILS	1	2	1	1	1	9	MEAN	2.16	1.38	2.00	1.29	1.93	1.04

****
****
****
***
***
*** S P E E D U P D A T A F I L E S ***
***
***
****
****
****

****
****
**
**
** PROCEDURES USED IN SPEEDUP FOR THE PROBLEMS **
**
** HEX1M,HEX2M AND BPNET FOR THE SIMULATION OF **
**
** THE HEAT EXCHANGER AND CONDENSER UNITS **
**
** S I M H E X **
**
** S I M C O N **

**
**
****
****

```
      SUBROUTINE SIMHEX(FH,TH0,CPH,FC,TC0,CPC,UA,TH,TC)
C
C
C     THIS SUBROUTINE CALCULATES THE OUTPUT TEMPERATURE
C     FOR BOTH STREAMS IN A HEAT EXCHANGER GIVEN ALL THE
C     INPUT DATA
C
C              FLOWRATE       TEMPERATURE     CP
C
C                             INPUT OUTPUT
C
C     HOT      FH             TH0   TH        CPH
C     COLD     FC             TC0   TC        CPC
C
C     UA IS THE PRODUCT OF THE U COEFFICIENT AND THE AREA
C
C
      COMMON/SPSTAT/LP,IPRINT,NERRS,IABORT
      C1=FH*CPH
      C2=FC*CPC
      IF(ABS(C1).LT.1.E-6)GO TO 50
      IF(ABS(C2).LT.1.E-6)GO TO 50
      GO TO 100
50    CONTINUE
      IF(IPRINT.LE.0)GO TO 60
      WRITE(LP,1000)C1,C2,UA
1000  FORMAT(" SIMHEX-W-BAD ARGUMENTS,C1=",G12.6," C2=",G12.6,
     +       " UA=",G12.6)
```

```
**********************************************************************
**********************************************************************
      SUBROUTINE SIMHEX(FH,TH0,CPH,FC,TC0,CPC,UA,TH,TC)
C
C
C     THIS SUBROUTINE CALCULATES THE OUTPUT TEMPERATURE
C     FOR BOTH STREAMS IN A HEAT EXCHANGER GIVEN ALL THE
C     INPUT DATA
C
C               FLOWRATE        TEMPERATURE      CP
C
C                               INPUT OUTPUT
C
C     HOT       FH              TH0   TH         CPH
C     COLD      FC              TC0   TC         CPC
C
C     UA IS THE PRODUCT OF THE U COEFFICIENT AND THE AREA
C
C
      COMMON/SPSTAT/LP,IPRINT,NERRS,IABORT
      C1=FH*CPH
      C2=FC*CPC
      IF(ABS(C1).LT.1.E-6)GO TO 50
      IF(ABS(C2).LT.1.E-6)GO TO 50
      GO TO 100
50    CONTINUE
      IF(IPRINT.LE.0)GO TO 60
      WRITE(LP,1000)C1,C2,UA
1000  FORMAT(" SIMHEX-W-BAD ARGUMENTS,C1=",G12.6," C2=",G12.6,
     +       " UA=",G12.6)
60    CONTINUE
      TH=TH0
      TC=TC0
      RETURN
100   CONTINUE
      BETA=C2/C1
      IF(ABS(BETA-1.0).LT.1.E-6)BETA=1.0-1.E-6
      ARG=UA/C2*(1.0-BETA)
      IF(ARG.GT.50.0)GO TO 50
      ALPHA=EXP(ARG)
      TC=((1-ALPHA)*TH0-(1-BETA)*TC0)/(BETA-ALPHA)
      TH=TH0+BETA*(TC0-TC)
      RETURN
      END
C++++++++++++++++++++++++++++++++++++++++++++++
C
C++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE SIMCON(TH,FC,TC0,CPC,UA,TC)
C
C
C     THIS SUBROUTINE CALCULATES THE TEMPERATURE FOR A CONDENSER
C     GIVEN ALL THE INPUT DATA
C
C               FLOWRATE        TEMPERATURE      CP
C
C                               INPUT OUTPUT
C
C     HOT                       TH
C     COLD      FC              TC0   TC         CPC
C
C     UA IS THE PRODUCT OF THE U COEFFICIENT AND THE AREA
C
C
      COMMON/SPSTAT/LP,IPRINT,NERRS,IABORT
```

```
      C2=FC*CPC
      C3=UA
      IF(ABS(C2).GE.1.E-6)GO TO 100
      TC=TC0
      RETURN
100   CONTINUE
      ARG=C3/C2
      IF(ARG.LT.50.0)GO TO 200
      TC=TH
      RETURN
200   CONTINUE
      TC=TH-(TH-TC0)/EXP(ARG)
      RETURN
      END
C**********************************************
C**********************************************


**********************************************************************
**********************************************************************
**                                                                  **
**                                                                  **
**        H  E  A  T  1                                             **
**        -------------                                             **
**                                                                  **
**                                                                  **
```

```
**                                                                    **
**                                                                    **
**       H E A T  1                                                   **
**       ----------                                                   **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************
TITLE
 HEAT EXCHANGER PROBLEM 1
****
MODEL HEATEXCHANGER
TYPE HOTIN,COLDIN,HOTOUT,COLDOUT AS FLOWRATE
     THOTIN,TCOLDIN,THOTOUT,TCOLDOUT AS TEMPERATURE
     ARG AS LOGARG
     Q AS NOTYPE2
     CPHOT,CPCOLD,UA,DELTATLM AS NOTYPE
STREAM
      INPUT  1 IS HOTIN,THOTIN
      INPUT 2  IS COLDIN,TCOLDIN
      OUTPUT 1 IS HOTOUT,THOTOUT
      OUTPUT 2 IS COLDOUT,TCOLDOUT
EQUATION
HOTOUT=HOTIN;
COLDOUT=COLDIN;
Q=HOTIN*CPHOT*(THOTIN-THOTOUT);
Q=COLDIN*CPCOLD*(TCOLDOUT-TCOLDIN);
Q=UA*DELTATLM;
ARG*(THOTIN-TCOLDOUT)=THOTOUT-TCOLDIN;
DELTATLM*LOG(ARG)=(THOTOUT-TCOLDIN)-(THOTIN-TCOLDOUT);
****
MODEL CONDENSER
TYPE HOTIN,COLDIN,HOTOUT,COLDOUT AS FLOWRATE
     THOTIN,TCOLDIN,THOTOUT,TCOLDOUT AS TEMPERATURE
     ARG AS LOGARG
     Q AS NOTYPE2
     L,CPCOLD,UA,DELTATLM AS NOTYPE
STREAM
      INPUT  1 IS HOTIN,THOTIN
      INPUT  2 IS COLDIN,TCOLDIN
      OUTPUT 1 IS HOTOUT,THOTOUT
      OUTPUT 2 IS COLDOUT,TCOLDOUT
EQUATION
HOTOUT=HOTIN;
COLDOUT=COLDIN;
THOTOUT=THOTIN;
Q=HOTIN*L;
Q=COLDIN*CPCOLD*(TCOLDOUT-TCOLDIN);
Q=UA*DELTATLM;
ARG*(THOTIN-TCOLDOUT)=THOTOUT-TCOLDIN;
DELTATLM*LOG(ARG)=(THOTOUT-TCOLDIN)-(THOTIN-TCOLDOUT);
****
FLOWSHEET
INPUT 1 OF EX1 IS FEED 1
INPUT 2 OF EX1 IS FEED 2
OUTPUT 1 OF EX1 IS PRODUCT 1
OUTPUT 2 OF EX1 IS INPUT 2 OF EX3
INPUT 1 OF EX2 IS FEED 3
INPUT 2 OF EX2 IS FEED 4
OUTPUT 1 OF EX2 IS INPUT 1 OF EX3
OUTPUT 2 OF EX2 IS INPUT 2 OF CON
INPUT 1 OF CON IS FEED 5
OUTPUT 1 OF CON IS PRODUCT 2
OUTPUT 2 OF CON IS PRODUCT 3
OUTPUT 1 OF EX3 IS INPUT 1 OF EX5
OUTPUT 2 OF EX3 IS PRODUCT 4
INPUT 2 OF EX5 IS FEED 6
OUTPUT 1 OF EX5 IS PRODUCT 5
OUTPUT 2 OF EX5 IS PRODUCT 6
****
UNIT EX1 IS A HEATEXCHANGER
****
UNIT EX2 IS A HEATEXCHANGER
****
UNIT EX3 IS A HEATEXCHANGER
****
UNIT EX5 IS A HEATEXCHANGER
****
UNIT CON IS A CONDENSER
****
OPERATION
SET
WITHIN EX1
  UA=3.9750,HOTIN=2.7778,THOTIN=320,CPHOT=0.6,TCOLDIN=140,CPCOLD=0.7
WITHIN EX2
  UA=1.6650,CPHOT=0.8,COLDIN=2.3060,TCOLDIN=240,CPCOLD=0.5
WITHIN EX3
  UA=0.8100,TCOLDOUT=320,CPHOT=0.8,CPCOLD=0.7
WITHIN EX5
  UA=0.8100,TCOLDIN=100,CPCOLD=1,CPHOT=0.8,THOTOUT=280,TCOLDOUT=160
```

```
  UA=3.9750,HOTIN=2.7778,THOTIN=320,CPHOT=0.6,TCOLDIN=140,CPCOLD=0.7
WITHIN EX2
  UA=1.6650,CPHOT=0.8,COLDIN=2.3060,TCOLDIN=240,CPCOLD=0.5
WITHIN EX3
  UA=0.8100,TCOLDOUT=320,CPHOT=0.8,CPCOLD=0.7
WITHIN EX5
  UA=0.8100,TCOLDIN=100,CPCOLD=1,CPHOT=0.8,THOTOUT=280,TCOLDOUT=160
WITHIN CON
  UA=1.1200,L=656.6,CPCOLD=0.5,THOTIN=560,TCOLDOUT=500
****
OPTIONS
ROUTINES ON
****
DECLARE TYPE FLOWRATE=1:0:1.E10,
             TEMPERATURE=500:100:900,
             LOGARG=1:0.001:1.E10,
             NOTYPE2=1:-1.E10:1.E10



*************************************************************************
*************************************************************************
**                                                                     **
**                                                                     **
**       H E A T 1 M                                                   **
**                                                                     **
**                                                                     **
*************************************************************************
```

```
*************************************************************************



TITLE
 HEAT EXCHANGER PROBLEM 1
****
MODEL HEATEXCHANGER
TYPE HOTIN,THOTIN,COLDIN,TCOLDIN,HOTOUT,THOTOUT,COLDOUT,
     TCOLDOUT,CPHOT,CPCOLD,UA AS NOTYPE
STREAM
      INPUT  1 IS HOTIN,THOTIN
      INPUT 2  IS COLDIN,TCOLDIN
      OUTPUT 1 IS HOTOUT,THOTOUT
      OUTPUT 2 IS COLDOUT,TCOLDOUT
EQUATION
HOTOUT=HOTIN;
COLDOUT=COLDIN;
PROCEDURE
  (THOTOUT,TCOLDOUT) SIMHEX (HOTIN,THOTIN,CPHOT,COLDIN,TCOLDIN,CPCOLD,UA)
****
MODEL CONDENSER
TYPE HOTIN,THOTIN,COLDIN,TCOLDIN,HOTOUT,THOTOUT,COLDOUT,
     TCOLDOUT,L,CPCOLD,UA  AS NOTYPE
STREAM
      INPUT  1 IS HOTIN,THOTIN
      INPUT  2 IS COLDIN,TCOLDIN
      OUTPUT 1 IS HOTOUT,THOTOUT
      OUTPUT 2 IS COLDOUT,TCOLDOUT
EQUATION
HOTOUT=HOTIN;
COLDOUT=COLDIN;
THOTOUT=THOTIN;
HOTIN*L=COLDIN*CPCOLD*(TCOLDOUT-TCOLDIN);
PROCEDURE
  (TCOLDOUT) SIMCON (THOTIN,COLDIN,TCOLDIN,CPCOLD,UA)
****
FLOWSHEET
INPUT 1 OF EX1 IS FEED 1
INPUT 2 OF EX1 IS FEED 2
OUTPUT 1 OF EX1 IS PRODUCT 1
OUTPUT 2 OF EX1 IS INPUT 2 OF EX3
INPUT 1 OF EX2 IS FEED 3
INPUT 2 OF EX2 IS FEED 4
OUTPUT 1 OF EX2 IS INPUT 1 OF EX3
OUTPUT 2 OF EX2 IS INPUT 2 OF CON
INPUT 1 OF CON IS FEED 5
OUTPUT 1 OF CON IS PRODUCT 2
OUTPUT 2 OF CON IS PRODUCT 3
OUTPUT 1 OF EX3 IS INPUT 1 OF EX5
OUTPUT 2 OF EX3 IS PRODUCT 4
INPUT 2 OF EX5 IS FEED 6
OUTPUT 1 OF EX5 IS PRODUCT 5
OUTPUT 2 OF EX5 IS PRODUCT 6
****
UNIT EX1 IS A HEATEXCHANGER
****
UNIT EX2 IS A HEATEXCHANGER
****
UNIT EX3 IS A HEATEXCHANGER
****
UNIT EX5 IS A HEATEXCHANGER
****
UNIT CON IS A CONDENSER
****
OPERATION
```

```
SET
WITHIN EX1
  UA=3.9750,HOTIN=2.7778,THOTIN=320,CPHOT=0.6,TCOLDIN=140,CPCOLD=0.7
WITHIN EX2
  UA=1.6650,CPHOT=0.8,COLDIN=2.3060,TCOLDIN=240,CPCOLD=0.5
WITHIN EX3
  UA=0.8100,TCOLDOUT=320,CPHOT=0.8,CPCOLD=0.7
WITHIN EX5
  UA=0.8100,TCOLDIN=100,CPCOLD=1,CPHOT=0.8,THOTOUT=280,TCOLDOUT=160
WITHIN CON
  UA=1.1200,L=656.6,CPCOLD=0.5,THOTIN=560,TCOLDOUT=500
PRESET
WITHIN EX3
  THOTIN=378:340:800,TCOLDIN=500:100:900
WITHIN CON
  TCOLDIN=500:100:900
WITHIN EX1
  COLDIN=2.38:0:1.E10
WITHIN EX2
  HOTIN=2.08:0:1.E10
WITHIN EX5
  COLDIN=1.66:0:1.E10
WITHIN EX1
  THOTOUT=500:100:900
WITHIN EX2
  THOTIN=500:100:900
WITHIN EX3
  THOTOUT=500:100:900
****
OPTIONS
ROUTINES ON
PRINT OPERATION 4 UNIT 4
```

```
**********************************************************************
**********************************************************************
**                                                                  **
**                                                                  **
**        H E A T 2                                                 **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************
```

```
TITLE
 HEAT EXCHANGER PROBLEM 2
****
MODEL SPLITTER
TYPE SPIN,TIN,SOU1,TOU1,SOU2,TOU2,SPLITFAC AS NOTYPE
STREAM
 INPUT  IS SPIN,TIN
 OUTPUT 1 IS SOU1,TOU1
 OUTPUT 2 IS SOU2,TOU2
EQUATION
TOU1=TOU2=TIN;
SOU1=SPLITFAC*SPIN;
SPIN=SOU1+SOU2;
****
MODEL MIXER
TYPE MIN1,TIN1,MIN2,TIN2,MOU,TOU AS NOTYPE
STREAM
 INPUT 1 IS MIN1,TIN1
 INPUT 2 IS MIN2,TIN2
 OUTPUT IS MOU,TOU
```

```
EQUATION
MOU=MIN1+MIN2;
MOU*TOU=MIN1*TIN1+MIN2*TIN2;
****
MODEL HEATEXCHANGER
TYPE HOTIN,THOTIN,COLDIN,TCOLDIN,HOTOUT,THOTOUT,COLDOUT,
     TCOLDOUT,Q,CPHOT,CPCOLD,UA,DELTATLM,ARG AS NOTYPE
STREAM
      INPUT  1 IS HOTIN,THOTIN
      INPUT 2  IS COLDIN,TCOLDIN
      OUTPUT 1 IS HOTOUT,THOTOUT
      OUTPUT 2 IS COLDOUT,TCOLDOUT
EQUATION
HOTOUT=HOTIN;
COLDOUT=COLDIN;
Q=HOTIN*CPHOT*(THOTIN-THOTOUT);
Q=COLDIN*CPCOLD*(TCOLDOUT-TCOLDIN);
Q=UA*DELTATLM;
ARG*(THOTIN-TCOLDOUT)=THOTOUT-TCOLDIN;
DELTATLM*LOG(ARG)=(THOTOUT-TCOLDIN)-(THOTIN-TCOLDOUT);
****
UNIT SPL IS A SPLITTER
****
UNIT MIX IS A MIXER
****
UNIT HEX1 IS A HEATEXCHANGER
****
UNIT HEX2 IS A HEATEXCHANGER
****
UNIT HEX3 IS A HEATEXCHANGER
****
FLOWSHEET
```

```
UNIT SPL IS A SPLITTER
****
UNIT MIX IS A MIXER
****
UNIT HEX1 IS A HEATEXCHANGER
****
UNIT HEX2 IS A HEATEXCHANGER
****
UNIT HEX3 IS A HEATEXCHANGER
****
FLOWSHEET
INPUT 1 OF HEX1 IS FEED 1
INPUT 2 OF HEX1 IS FEED 2
OUTPUT 1 OF HEX1 IS PRODUCT 1
OUTPUT 2 OF HEX1 IS INPUT OF SPL
OUTPUT 1 OF SPL IS INPUT 2 OF HEX2
OUTPUT 2 OF SPL IS INPUT 2 OF HEX3
INPUT 1 OF HEX2 IS FEED 3
OUTPUT 1 OF HEX2 IS PRODUCT 2
OUTPUT 2 OF HEX2 IS INPUT 1 OF MIX
INPUT 1 OF HEX3 IS FEED 4
OUTPUT 1 OF HEX3 IS PRODUCT 3
OUTPUT 2 OF HEX3 IS INPUT 2 OF MIX
OUTPUT OF MIX IS PRODUCT 4
****
OPERATION
SET
WITHIN HEX1
  UA=0.44823899,HOTIN=0.7,THOTIN=180,CPHOT=1,
  COLDIN=1.0000,TCOLDIN=80,CPCOLD=0.8
WITHIN HEX2
  UA=1.2033485,HOTIN=0.6000,THOTIN=250,CPHOT=0.7,CPCOLD=0.8
WITHIN HEX3
  UA=0.31086886,HOTIN=0.4000,THOTIN=220,CPHOT=0.6,CPCOLD=0.8
WITHIN MIX
  TOU=190
PRESET
WITHIN SPL
  TIN=115:90:160,SPLITFAC=0.5249:0:1
WITHIN MIX
  TIN1=220:100:240,TIN2=156:100:200
WITHIN HEX1
  THOTOUT=170:80:250
WITHIN HEX2
  THOTOUT=170:80:250,TCOLDIN=170:80:250,
  TCOLDOUT=170:80:250
WITHIN HEX3
  THOTOUT=170:80:250,TCOLDIN=170:80:250,
  TCOLDOUT=170:80:250
WITHIN HEX1
  ARG=1:0.001:1.E10
WITHIN HEX2
  ARG=1:0.001:1.E10
WITHIN HEX3
  ARG=1:0.001:1.E10
****
OPTIONS
ROUTINES ON
PRINT UNIT 4 OPERATION 4


**********************************************************************
**********************************************************************
**                                                                  **
**                                                                  **
**        H E A T 2 M                                               **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************


TITLE
 HEAT EXCHANGER PROBLEM 2
****
MODEL SPLITTER
TYPE SPIN,TIN,SOU1,TOU1,SOU2,TOU2,SPLITFAC AS NOTYPE
STREAM
 INPUT  IS SPIN,TIN
 OUTPUT 1 IS SOU1,TOU1
 OUTPUT 2 IS SOU2,TOU2
EQUATION
TOU1=TOU2=TIN;
SOU1=SPLITFAC*SPIN;
SPIN=SOU1+SOU2;
****
MODEL MIXER
TYPE MIN1,TIN1,MIN2,TIN2,MOU,TOU AS NOTYPE
STREAM
 INPUT 1 IS MIN1,TIN1
 INPUT 2 IS MIN2,TIN2
 OUTPUT IS MOU,TOU
EQUATION
MOU=MIN1+MIN2;
MOU*TOU=MIN1*TIN1+MIN2*TIN2;
****
MODEL HEATEXCHANGER
TYPE HOTIN,THOTIN,COLDIN,TCOLDIN,HOTOUT,THOTOUT,COLDOUT,
     TCOLDOUT,CPHOT,CPCOLD,UA AS NOTYPE
STREAM
```

```
MODEL HEATEXCHANGER
TYPE HOTIN,THOTIN,COLDIN,TCOLDIN,HOTOUT,THOTOUT,COLDOUT,
     TCOLDOUT,CPHOT,CPCOLD,UA AS NOTYPE
STREAM
       INPUT  1 IS HOTIN,THOTIN
       INPUT 2  IS COLDIN,TCOLDIN
       OUTPUT 1 IS HOTOUT,THOTOUT
       OUTPUT 2 IS COLDOUT,TCOLDOUT
EQUATION
HOTOUT=HOTIN;
COLDOUT=COLDIN;
PROCEDURE
  (THOTOUT,TCOLDOUT) SIMHEX (HOTIN,THOTIN,CPHOT,COLDIN,TCOLDIN,
                             CPCOLD,UA)
```

```
****
UNIT SPL IS A SPLITTER
****
UNIT MIX IS A MIXER
****
UNIT HEX1 IS A HEATEXCHANGER
****
UNIT HEX2 IS A HEATEXCHANGER
****
UNIT HEX3 IS A HEATEXCHANGER
****
FLOWSHEET
INPUT 1 OF HEX1 IS FEED 1
INPUT 2 OF HEX1 IS FEED 2
OUTPUT 1 OF HEX1 IS PRODUCT 1
OUTPUT 2 OF HEX1 IS INPUT OF SPL
OUTPUT 1 OF SPL IS INPUT 2 OF HEX2
OUTPUT 2 OF SPL IS INPUT 2 OF HEX3
INPUT 1 OF HEX2 IS FEED 3
OUTPUT 1 OF HEX2 IS PRODUCT 2
OUTPUT 2 OF HEX2 IS INPUT 1 OF MIX
INPUT 1 OF HEX3 IS FEED 4
OUTPUT 1 OF HEX3 IS PRODUCT 3
OUTPUT 2 OF HEX3 IS INPUT 2 OF MIX
OUTPUT OF MIX IS PRODUCT 4
****
OPERATION
SET
WITHIN HEX1
  UA=0.94823899,HOTIN=0.7,THOTIN=180,CPHOT=1,
  COLDIN=1.0000,TCOLDIN=80,CPCOLD=0.8
WITHIN HEX2
  UA=1.2033485,HOTIN=0.6000,THOTIN=250,CPHOT=0.7,CPCOLD=0.8
WITHIN HEX3
  UA=0.31086886,HOTIN=0.4000,THOTIN=220,CPHOT=0.6,CPCOLD=0.8
WITHIN MIX
  TOU=190
PRESET
WITHIN SPL
  TIN=170:80:250,SPLITFAC=0.5249:0:1
WITHIN MIX
  TIN1=170:080:250,TIN2=170:080:250
WITHIN HEX1
  THOTOUT=170:80:250
WITHIN HEX2
  THOTOUT=170:80:250,TCOLDIN=170:80:250,
  TCOLDOUT=170:80:250
WITHIN HEX3
  THOTOUT=170:80:250,TCOLDIN=170:80:250,
  TCOLDOUT=170:80:250
****
OPTIONS
ROUTINES ON
PRINT UNIT 4 OPERATION 4
```

```
**********************************************************************
**********************************************************************
**                                                                  **
**                                                                  **
**        C A V S I M                                               **
**                                                                  **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************
```

```
TITLE
 MODIFIED CAVETT'S PROCESS
****
    MODEL MIXERMB
    •  MIXER OF TWO STREAMS (MATERIAL BALANCE ONLY)  •
       SET NOCOMP
       TYPE FLOWIN1, FLOWIN2, FLOWOUT AS ARRAY (NOCOMP) OF FLOWRATE
       STREAM INPUT  1 FLOWIN1
              INPUT  2 FLOWIN2
              OUTPUT   FLOWOUT
       EQUATION
         •  COMPONENT MATERIAL BALANCES  •
           FLOWIN1 + FLOWIN2 = FLOWOUT;
****
DECLARE COMPONENTS 5,102,104,109,117
```

```
TITLE
 MODIFIED CAVETT'S PROCESS
****
    MODEL MIXERMB
    • MIXER OF TWO STREAMS (MATERIAL BALANCE ONLY)  •
      SET NOCOMP
      TYPE FLOWIN1, FLOWIN2, FLOWOUT AS ARRAY (NOCOMP) OF FLOWRATE
      STREAM INPUT  1 FLOWIN1
             INPUT  2 FLOWIN2
             OUTPUT   FLOWOUT
      EQUATION
        • COMPONENT MATERIAL BALANCES  •
          FLOWIN1 + FLOWIN2 = FLOWOUT;
****
DECLARE COMPONENTS 5,102,104,109,117
        TYPE FLOWRATE=1:0:10 UNIT="MMOLES",
         ENTHALPIE=10:-1.E10:1.E10,
             TEMPERATURE=320:280:350 UNIT="KELVIN",
             PRESSURE=1:0:100 UNIT="BARS"
****
  MODEL ISOFLASHMB
     SET NOCOMP
     TYPE FLOWIN, FLOWOUT1, FLOWOUT2 AS ARRAY (NOCOMP) OF FLOWRATE
          T AS TEMPERATURE
          P AS PRESSURE
     STREAM INPUT  IS FLOWIN
            OUTPUT 1 IS FLOWOUT1
            OUTPUT 2 IS FLOWOUT2
     PROCEDURE
       (FLOWOUT1,FLOWOUT2) SPHASE (T,P,FLOWIN)
****
FLOWSHEET
INPUT 1 OF MIX1 IS FEED 1
OUTPUT OF MIX1 IS INPUT 1 OF MIX2
INPUT 2 OF MIX1 IS OUTPUT 2 OF FL1
INPUT 2 OF MIX2 IS OUTPUT 1 OF FL3
OUTPUT OF MIX2 IS INPUT OF FL2
OUTPUT 1 OF FL2 IS INPUT OF FL1
OUTPUT 1 OF FL1 IS PRODUCT 1
OUTPUT 2 OF FL2 IS INPUT 2 OF MIX3
INPUT 1 OF MIX3 IS OUTPUT 1 OF FL4
OUTPUT OF MIX3 IS INPUT OF FL3
OUTPUT 2 OF FL3 IS INPUT OF FL4
OUTPUT 2 OF FL4 IS PRODUCT 2
****
UNIT MIX1 IS A MIXERMB
SET NOCOMP=5
****
UNIT MIX2 IS A MIXERMB
SET NOCOMP=5
****
UNIT MIX3 IS A MIXERMB
SET NOCOMP=5
****
UNIT FL1 IS A ISOFLASHMB
SET NOCOMP=5
****
UNIT FL2 IS A ISOFLASHMB
SET NOCOMP=5
****
UNIT FL3 IS A ISOFLASHMB
SET NOCOMP=5
****
UNIT FL4 IS A ISOFLASHMB
```

```
SET NOCOMP=5
****
OPERATION
SET
WITHIN MIX1
  FLOWIN1=(0.5124,0.3625,0.1205,0.0932,0.0266)
WITHIN FL1
  P=56.2,T=311
WITHIN FL2
  T=322,P=19.6
WITHIN FL3
  T=309,P=4.39
WITHIN FL4
  T=303,P=1.91
PRESET
WITHIN FL1 FLOWIN=(0.5124,0.3625,0.1205,0.0)
****
OPTIONS
ROUTINES ON,CHAOSEADER



**********************************************************************
**********************************************************************
**                                                                  **
**                                                                  **
**       C A V D E S                                                **
**                                                                  **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************


TITLE
 MODIFIED CAVETT'S PROCESS
****
```

```
**                                                                    **
**                                                                    **
**        C A V D E S                                                 **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************

TITLE
 MODIFIED CAVETT'S PROCESS
****
    MODEL MIXERMB
    • MIXER OF TWO STREAMS (MATERIAL BALANCE ONLY)  •
      SET NOCOMP
      TYPE FLOWIN1, FLOWIN2, FLOWOUT AS ARRAY (NOCOMP) OF FLOWRATE
      STREAM INPUT  1 FLOWIN1
             INPUT  2 FLOWIN2
             OUTPUT   FLOWOUT
      EQUATION
        • COMPONENT MATERIAL BALANCES  •
          FLOWIN1 + FLOWIN2 = FLOWOUT;
****
DECLARE COMPONENTS 5 '02,104,109,117
        TYPE FLOWRATE=1:0:10 UNIT="MMOLES",
         ENTHALPIE=10:-1.E10:1.E10,
             TEMPERATURE=320:280:350 UNIT="KELVIN",
             PRESSURE=1:0:100 UNIT="BARS"
****
  MODEL ISOFLASHMB
     SET NOCOMP
     TYPE FLOWIN, FLOWOUT1, FLOWOUT2 AS ARRAY (NOCOMP) OF FLOWRATE
          T AS TEMPERATURE
          P AS PRESSURE
     STREAM INPUT  IS FLOWIN
            OUTPUT 1 IS FLOWOUT1
            OUTPUT 2 IS FLOWOUT2
     PROCEDURE
       (FLOWOUT1,FLOWOUT2) SPHASE (T,P,FLOWIN)
****
FLOWSHEET
INPUT 1 OF MIX1 IS FEED 1

OUTPUT OF MIX1 IS INPUT 1 OF MIX2
INPUT 2 OF MIX1 IS OUTPUT 2 OF FL1
INPUT 2 OF MIX2 IS OUTPUT 1 OF FL3
OUTPUT OF MIX2 IS INPUT OF FL2
OUTPUT 1 OF FL2 IS INPUT OF FL1
OUTPUT 1 OF FL1 IS PRODUCT 1
OUTPUT 2 OF FL2 IS INPUT 2 OF MIX3
INPUT 1 OF MIX3 IS OUTPUT 1 OF FL4
OUTPUT OF MIX3 IS INPUT OF FL3
OUTPUT 2 OF FL3 IS INPUT OF FL4
OUTPUT 2 OF FL4 IS PRODUCT 2
****
UNIT MIX1 IS A MIXERMB
SET NOCOMP=5
****
UNIT MIX2 IS A MIXERMB
SET NOCOMP=5
****
UNIT MIX3 IS A MIXERMB
SET NOCOMP=5
****
UNIT FL1 IS A ISOFLASHMB
SET NOCOMP=5
****
UNIT FL2 IS A ISOFLASHMB
SET NOCOMP=5
****
UNIT FL3 IS A ISOFLASHMB
SET NOCOMP=5
****
UNIT FL4 IS A ISOFLASHMB
SET NOCOMP=5
****
OPERATION
SET
WITHIN MIX1
  FLOWIN1=(0.5124,0.3625,0.1205,0.0932,0.0266)
WITHIN FL1
  P=56.2,T=311
WITHIN FL2
  T=322,P=19.6
WITHIN FL3
  T=309,P=4.39
WITHIN FL4
  FLOWOUT2(2)=2.723E-3,P=1.91
PRESET
WITHIN FL1 FLOWIN=(0.5124,0.3625,0.1205,0,0)
WITHIN FL4 T=290
****
OPTIONS
ROUTINES QN,CHAOSEADER


************************************************************************
************************************************************************
**                                                                    **
**                                                                    **
**        C H E M C A D                                               **
```

```
**********************************************************************
**                                                                  **
**                                                                  **
**      C H E M C O 1                                               **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************


*********************************************************************
```

```
*                                                                   *
*  L I B R A R Y   O F   M O D E L S   F O R                        *
*                                                                   *
*           S P E E D U P                                           *
*                                                                   *
*********************************************************************
MODEL EXPANDER
*************
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF FLOWRATE
     TIN,TOU,TIDEAL AS TEMPERATURE
     PIN,POU AS PRESSURE
     HIN,HOU,HIDEAL,WORK AS ENTHALPY
     EFFIC AS FRACTION
     GAMMA AS NOTYPE
STREAM
     INPUT IS F,TIN,PIN,HIN
     OUTPUT 1 IS F,TOU,POU,HOU
     OUTPUT 2 IS WORK
EQUATION
     TIDEAL * PIN ** GAMMA = TIN * POU ** GAMMA;
     EFFIC * ( HIN - HIDEAL ) = HIN - HOU;
     WORK = HOU - HIN;
PROCEDURE
  ( HIDEAL ) ZCALEN ( TIDEAL , POU , F )
  ( HOU    ) ZCALEN ( TOU    , POU , F )
****
MODEL COMPRESSOR
****************
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF FLOWRATE
     TIN,TOU,TIDEAL AS TEMPERATURE
     PIN,POU AS PRESSURE
     HIN,HOU,HIDEAL,WORK AS ENTHALPY
     EFFIC AS FRACTION
     GAMMA AS NOTYPE
STREAM
     INPUT 1 IS F,TIN,PIN,HIN
     INPUT 2 IS WORK
     OUTPUT IS F,TOU,POU,HOU
EQUATION
     TIDEAL * PIN ** GAMMA = TIN * POU ** GAMMA;
     EFFIC * ( HIN - HOU ) = HIN - HIDEAL;
     WORK = HOU - HIN;
PROCEDURE
  ( HIDEAL ) ZCALEN ( TIDEAL , POU , F )
  ( HOU    ) ZCALEN ( TOU    , POU , F )
****
MODEL HEATER_COOLER
*******************
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF  FLOWRATE
     TIN,TOU AS TEMPERATURE
     PIN,POU AS PRESSURE
     HIN,HOU,Q AS ENTHALPY
     DELTAT,DELTAP AS DELTA
STREAM
     INPUT IS F,TIN,PIN,HIN
     OUTPUT IS F,TOU,POU,HOU
EQUATION
     TOU = TIN + DELTAT;
     POU = PIN + DELTAP;
     Q = HOU - HIN;
PROCEDURE
     ( HOU ) ZCALEN ( TOU , POU , F )
****
```

```
MODEL VALVE
*****************
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF  FLOWRATE
     TIN,TOU AS TEMPERATURE
     PIN,POU AS PRESSURE
     H AS ENTHALPY
     DELTAP AS DELTA
STREAM
     INPUT IS F,TIN,PIN,H
     OUTPUT IS F,TOU,POU,H
EQUATION
     POU = PIN - DELTAP;
PROCEDURE
     ( TOU ) ZCALTP ( H , POU , F )
****
MODEL PUMP
*************
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
```

```
MODEL VALVE
*****************
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF  FLOWRATE
     TIN,TOU AS TEMPERATURE
     PIN,POU AS PRESSURE
     H AS ENTHALPY
     DELTAP AS DELTA
STREAM
     INPUT IS F,TIN,PIN,H
     OUTPUT IS F,TOU,POU,H
EQUATION
     POU = PIN - DELTAP;
PROCEDURE
     ( TOU ) ZCALTP ( H , POU , F )
****
MODEL PUMP
************
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
     PIN,POU AS PRESSURE
     HIN,HOU AS ENTHALPY
     DELTAP AS DELTA
STREAM
     INPUT IS F,T,PIN,HIN
     OUTPUT IS F,T,POU,HOU
EQUATION
     POU = PIN + DELTAP;
PROCEDURE
     ( HOU ) ZCALEN ( T , POU , F )
****
MODEL MOL_TO_KG_CONVERTER
************************
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
     P AS PRESSURE
     H AS ENTHALPY
     X AS ARRAY(NOCOMP) OF FRACTION
     FKG AS ARRAY(NOCOMP) OF KGFLOW
     MOLW AS ARRAY(NOCOMP) OF NOTYPE
     FTOTALKG AS KGFLOW
STREAM
     INPUT IS F,T,P,H
     OUTPUT IS F,T,P,H
EQUATION
     FKG = MOLW * F;
     FTOTALKG = SIGMA ( FKG );
     X * FTOTALKG = FKG;
****
MODEL MOLAR_FRACTION_TO_MOLES_CONVERTER
***************************************
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
     P AS PRESSURE
     H AS ENTHALPY
     X AS ARRAY(NOCOMP) OF FRACTION
     FTOTAL AS FLOWRATE
STREAM
     INPUT IS F,T,P,H
     OUTPUT IS F,T,P,H
EQUATION
     FTOTAL * X = F;
     FTOTAL = SIGMA ( F ) ;
```

```
****
MODEL ADIABATIC_FLASH
*********************
SET NOCOMP
TYPE F,TOP,BOT AS ARRAY(NOCOMP) OF FLOWRATE
     T,TTOP,TBOT AS TEMPERATURE
     P AS PRESSURE
     H,HTOP,HBOT AS ENTHALPY
     FRAC AS FRACTION
STREAM
     INPUT IS F,T,P,H
     OUTPUT 1 IS TOP,TTOP,P,HTOP
     OUTPUT 2 IS BOT,TBOT,P,HBOT
EQUATION
     TBOT = TTOP;
     SIGMA(TOP) = FRAC * SIGMA(F);
PROCEDURE
     ( TTOP , TOP , BOT ) SCALTP ( H , P , F)
     ( HTOP ) ZENTHV ( TTOP , P , TOP )
     ( HBOT ) ZENTHL ( TBOT , P , BOT )
****
MODEL ISOTHERMAL_FLASH
**********************
SET NOCOMP
TYPE F,TOP,BOT AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
     P AS PRESSURE
     H,HTOP,HBOT AS ENTHALPY
STREAM
     INPUT IS F,T,P,H
     OUTPUT IS TOP,T,P,HTOP
     OUTPUT 2 IS BOT,T,P,HBOT
PROCEDURE
```

```
MODEL ISOTHERMAL_FLASH
*********************
SET NOCOMP
TYPE F,TOP,BOT AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
     P AS PRESSURE
     H,HTOP,HBOT AS ENTHALPY
STREAM
     INPUT IS F,T,P,H
     OUTPUT IS TOP,T,P,HTOP
     OUTPUT 2 IS BOT,T,P,HBOT
PROCEDURE
     ( TOP , BOT ) SPHASE ( T , P , F )
     ( HTOP ) ZENTHV ( T , P , TOP )
     ( HBOT ) ZENTHL ( T , P , BOT )
****
MODEL VOLUME_TO_MOLES_CONVERTER
******************************
SET NOCOMP
TYPE F AS ARRAY(NOCOMP) OF FLOWRATE
     T AS TEMPERATURE
     P AS PRESSURE
     H AS ENTHALPY
     TOTALFVOL,RHO AS NOTYPE
     TOTALF AS FLOWRATE
STREAM
     INPUT IS F,T,P,H
     OUTPUT IS F,T,P,H
EQUATION
     TOTALF = RHO * TOTALFVOL;
     TOTALF = SIGMA ( F );
****
MODEL MIXER
**********
SET NOCOMP
TYPE F1,F2,F AS ARRAY(NOCOMP) OF FLOWRATE
     T1,T2,T AS TEMPERATURE
     P1,P2,P AS PRESSURE
     H1,H2,H AS ENTHALPY
STREAM
     INPUT 1 IS F1,T1,P1,H1
     INPUT 2 IS F2,T2,P2,H2
     OUTPUT IS F,T,P,H
EQUATION
     F = F1 + F2;
```

```
     P = P1;
     H = H1 + H2;
PROCEDURE
     ( T ) = ZCALTP ( H , P , F )
****
```

```
.....................................
.   E N D    O F    L I B R A R Y   .
.....................................
FLOWSHEET
INPUT OF VM1 IS FEED
OUTPUT OF VM1 IS INPUT OF K1
OUTPUT OF K1 IS INPUT OF E2P
OUTPUT OF E2P IS INPUT 1 OF MIX
OUTPUT OF MIX IS INPUT OF R1
OUTPUT OF R1 IS INPUT OF V1
OUTPUT 1 OF V1 IS INPUT OF C1
OUTPUT 1 OF C1 IS INPUT OF T2
OUTPUT OF T2 IS INPUT OF V2
OUTPUT 1 OF V2 IS INPUT OF VM2
OUTPUT 2 OF V2 IS INPUT OF P2
OUTPUT OF P2 IS INPUT 2 OF MIX
OUTPUT OF VM2 IS INPUT OF E2
OUTPUT OF E2 IS INPUT OF K2
OUTPUT OF K2 IS PRODUCT 1
OUTPUT 2 OF V1 IS INPUT OF T3
OUTPUT OF T3 IS INPUT OF R2
OUTPUT OF R2 IS INPUT OF V3
OUTPUT 1 OF V3 IS INPUT 1 OF C2
OUTPUT 2 OF C1 IS INPUT 2 OF C2
OUTPUT 2 OF V3 IS INPUT OF E1
OUTPUT OF E1 IS INPUT OF K5
OUTPUT OF K5 IS PRODUCT 4
OUTPUT OF C2 IS INPUT OF V4
OUTPUT 1 OF V4 IS PRODUCT 2
OUTPUT 2 OF V4 IS PRODUCT 3
****
MODEL BLACKBOX
SET NOCOMP
TYPE FIN,TOP,BOT AS ARRAY(NOCOMP) OF FLOWRATE
     TIN,TTOP,TBOT AS TEMPERATURE
     PIN,PTOP,PBOT AS PRESSURE
     HIN,HTOP,HBOT AS ENTHALPY
     DELTAP,DELTAT AS DELTA
     SPLITR,XTOP,XBOT AS ARRAY(NOCOMP) OF FRACTION
     COMP6T AS FRACTION
     TOTALTOP,TOTALBOT AS FLOWRATE
     TOTALTOPVOL,RHO AS NOTYPE
     BOTKG,TOPKG AS ARRAY(NOCOMP) OF KGFLOW
     MOLW AS ARRAY(NOCOMP) OF NOTYPE
     TOTALBOTKG,TOTALTOPKG AS KGFLOW
STREAM
  INPUT    IS FIN,TIN,PIN,HIN
  OUTPUT 1 IS TOP,TTOP,PTOP,HTOP
  OUTPUT 2 IS BOT,TBOT,PBOT,HBOT
```

```
    TOTALBOTKG, TOTALTOPKG AS KGFLOW
STREAM
  INPUT    IS FIN,TIN,PIN,HIN
  OUTPUT 1 IS TOP,TTOP,PTOP,HTOP
  OUTPUT 2 IS BOT,TBOT,PBOT,HBOT
EQUATION
  FIN = TOP + BOT;
  BOT = SPLITR * FIN;
  PTOP = PIN + DELTAP;
  PBOT = PTOP;
  TBOT = TIN + DELTAT;
  TOTALTOP = SIGMA(TOP);
  TOTALBOT = SIGMA(BOT);
  XTOP * TOTALTOP = TOP;
  XBOT * TOTALBOT = BOT;
  TOTALTOP = RHO * TOTALTOPVOL;
  COMP6T = XTOP(6) + XTOP(7);
```

```
  TOTALBOTKG = SIGMA(BOTKG);
  TOTALTOPKG = SIGMA(TOPKG);
  BOTKG = MOLW * BOT;
  TOPKG = MOLW * TOP;
PROCEDURE
  ( TTOP ) SDEWPT ( PTOP , TOP )
  ( HTOP ) ZENTHV ( TTOP , PTOP , TOP )
  ( HBOT ) ZENTHL ( TBOT , PBOT , BOT )
****
UNIT VM1 IS A VOLUME_TO_MOLES_CONVERTER SET NOCOMP=9
****
UNIT K1 IS A MOL_TO_KG_CONVERTER SET NOCOMP=9
****
UNIT E2P IS A HEATER_COOLER SET NOCOMP=9
****
UNIT V2 IS A ISOTHERMAL_FLASH SET NOCOMP=9
****
UNIT VM2 IS A VOLUME_TO_MOLES_CONVERTER SET NOCOMP=9
****
UNIT E2 IS A HEATER_COOLER SET NOCOMP=9
****
UNIT K2 IS A MOL_TO_KG_CONVERTER SET NOCOMP=9
****
UNIT P2 IS A PUMP SET NOCOMP=9
****
UNIT T2 IS A HEATER_COOLER SET NOCOMP=9
****
UNIT V3 IS AN ADIABATIC_FLASH SET NOCOMP=9
****
UNIT T3 IS A HEATER_COOLER SET NOCOMP=9
****
UNIT R2 IS A VALVE SET NOCOMP=9
****
UNIT MIX IS A MIXER SET NOCOMP=9
****
UNIT C1 IS A EXPANDER SET NOCOMP=9
****
UNIT C2 IS A COMPRESSOR SET NOCOMP=9
****
UNIT V4 IS A BLACKBOX SET NOCOMP=9
****
UNIT V1 IS AN ADIABATIC_FLASH SET NOCOMP=9
****
UNIT R1 IS A VALVE SET NOCOMP=9
****
UNIT E1 IS A HEATER_COOLER SET NOCOMP=9
****
UNIT K5 IS A MOL_TO_KG_CONVERTER SET NOCOMP=9
****
OPERATION
SET
WITHIN VM1
  TOTALFVOL=125,RHO=4.634,H=0
WITHIN K1
  MOLW = ( 28 , 16 , 30 , 44 , 58 , 72 , 86 , 100 , 114 ),
  X(1)=0.09,X(3)=0.112,X(4)=0.062,X(5)=0.054,
  X(6)=0.03,X(7)=0.081,X(8)=0.133,X(9)=0.021,
  T=313,P=75
WITHIN E2P TOU=288,DELTAP=0
WITHIN K2
  MOLW = ( 28 , 16 , 30 , 44 , 58 , 72 , 86 , 100 , 114 )
WITHIN R1
  DELTAP=0.3
WITHIN P2  POU=75
WITHIN T2 DELTAP=0
WITHIN V3
```

```
  FRAC=0.4616
WITHIN T3
  DELTAT=2,DELTAP=0
WITHIN E1
  DELTAT=-5,DELTAP=-0.2
WITHIN K5
  MOLW = ( 28 , 16 , 30 , 44 , 58 , 72 , 86 , 100 , 114 )
WITHIN C1
  EFFIC=0.5,GAMMA=0.1
WITHIN C2
  EFFIC=0.8,GAMMA=0.1
WITHIN V4
  DELTAT=-10,DELTAP=-0.1,RHO=3.833,
  MOLW = ( 28 , 16 , 30 , 44 , 58 , 72 , 86 , 100 , 114 ),
```

```
  FRAC=0.4616
WITHIN T3
  DELTAT=2,DELTAP=0
WITHIN E1
  DELTAT=-5,DELTAP=-0.2
WITHIN K5
  MOLW = ( 28 , 16 , 30 , 44 , 58 , 72 , 86 , 100 , 114 )
WITHIN C1
  EFFIC=0.5,GAMMA=0.1
WITHIN C2
  EFFIC=0.8,GAMMA=0.1
WITHIN V4
  DELTAT=-10,DELTAP=-0.1,RHO=3.833,
  MOLW = ( 28 , 16 , 30 , 44 , 58 , 72 , 86 , 100 , 114 ),
  SPLITR = ( 0 , 0 , 0 , 0 , 0.15 , 0.43 , 0.21 , 0.06 , 0 )
WITHIN V2
  P=25,T=213
WITHIN VM2
  RHO=1.691
WITHIN E2 Q=0.144,DELTAP=0
****
TITLE
 PROBLEM 1 (CASE A)  FOR EFCE CHEMCOMP 82
****
DECLARE
  TYPE FLOWRATE=50:0:1000 UNIT="KMOL/HR",
       KGFLOW=5000:0:500000 UNIT="KG/HR",
       TEMPERATURE=300:200:900 UNIT="DEG KELVIN",
       PRESSURE=40:1:300 UNIT="BARS",
       ENTHALPY=10:-1.0E10:1.0E10 UNIT="GJOULES/HR",
       FRACTION=0.5:0:1,
       DELTA=0.0:-1.0E10:1.0E10
  COMPONENTS 2,101,102,103,104,106,109,114,115
****
OPTIONS
ROUTINES ON,CHAOSEADER
PRINT UNIT 2 MODEL 2 FLOWSHEET 2 OPERATION 2 DECLARE 2
****



**********************************************************************
**********************************************************************
**                                                                  **
**                                                                  **
**        C H E M C O 2                                             **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************


FLOWSHEET
   OUTPUT 1 OF SEPARATOR IS PRODUCT 1
   OUTPUT 2 OF SEPARATOR IS INPUT OF PURGE
   OUTPUT 1 OF PURGE IS PRODUCT 2
   OUTPUT 2 OF PURGE IS INPUT 2 OF MIXER
   OUTPUT OF MIXER IS INPUT OF REACTOR
   OUTPUT OF REACTOR IS INPUT OF SEPARATOR
   OUTPUT OF FEEDER IS INPUT 1 OF MIXER
****
   MODEL SPLITMB
   *  TWO STREAM SPLITTER (MATERIAL BALANCE ONLY)  *
      SET NOCOMP
      TYPE FLOWIN, FLOWOUT1, FLOWOUT2 AS ARRAY (NOCOMP) OF FLOWRATE
           TOTAL1, TOTAL2 AS FLOWRATE
```

```
        X1, X2 AS ARRAY (NOCOMP) OF MOLEFRACTION
        FRACTION AS MOLEFRACTION
    STREAM INPUT    FLOWIN
           OUTPUT 1 FLOWOUT1
           OUTPUT 2 FLOWOUT2
    EQUATION
      • STREAM SPLIT •
            FLOWOUT1 = FLOWIN * FRACTION;
      • COMPONENT MATERIAL BALANCES •
            FLOWIN = FLOWOUT1 + FLOWOUT2;
      • TOTAL FLOWS OF OUTPUT STREAMS •
            TOTAL1 = SIGMA (FLOWOUT1);
            TOTAL2 = SIGMA (FLOWOUT2);
      • MATERIAL FRACTIONS OF OUTPUT STREAMS •
          X1 * TOTAL1 = FLOWOUT1;
          X2 * TOTAL2 = FLOWOUT2;
****
MODEL     MESEPARATOR
  TYPE
    FLOWIN, FLOWOUT1, FLOWOUT2 AS ARRAY (7) OF FLOWRATE
    TOTAL1, TOTAL2 AS FLOWRATE
    X3, X4, Y3, Y4 AS MOLEFRACTION
  STREAM
    INPUT      FLOWIN
    OUTPUT 1   FLOWOUT1
    OUTPUT 2   FLOWOUT2
  EQUATION
    • OVERALL BALANCES •
      FLOWOUT1 + FLOWOUT2 = FLOWIN   ;
    • TOTAL FLOWS •
                TOTAL1 = SIGMA (FLOWOUT1)   ;
                TOTAL2 = SIGMA (FLOWOUT2)   ;
    • MOLE FRACTIONS •
              X3 * TOTAL1 = FLOWOUT1(3)   ;
              X4 * TOTAL1 = FLOWOUT1(4)   ;
              Y3 * TOTAL2 = FLOWOUT2(3)   ;
              Y4 * TOTAL2 = FLOWOUT2(4)   ;
    • EQUILIBRIUM RELATIONS •
                        Y3 = 3.8E-3 * X3   ;
                        Y4 = 8.0E-4 * X4   ;
              FLOWOUT1(1) = 0   ;
              FLOWOUT1(2) = 0   ;
              FLOWOUT1(5) = 0   ;
              FLOWOUT1(6) = 0   ;
              FLOWOUT1(7) = 0   ;
****
  MODEL FEEDERMB
  • FEED STREAM DEFINITION (MATERIAL BALANCE ONLY) •
    SET NOCOMP
    TYPE  FLOWOUT AS ARRAY (NOCOMP) OF FLOWRATE
          X AS ARRAY (NOCOMP) OF MOLEFRACTION
          TOTAL AS FLOWRATE
    STREAM
      OUTPUT IS FLOWOUT
    EQUATION
      • TOTAL FEED FLOW •
              TOTAL = SIGMA (FLOWOUT);
      • MATERIAL FRACTIONS OF FEED STREAM •
          X * TOTAL = FLOWOUT;
****
MODEL     MEREACTOR
  TYPE
    FLOWIN, FLOWOUT AS ARRAY (7) OF FLOWRATE
    TOTAL AS FLOWRATE
    EXTENT1,EXTENT2 AS EXTENT
    X AS ARRAY (7) OF MOLEFRACTION

    KJ1,KJ2,KX1,KX2,KP10,KP20,T1,T2,PRESSURE AS NOTYPE
  STREAM
    INPUT   FLOWIN
    OUTPUT  FLOWOUT
  EQUATION
    • DEFINITION OF EQUILIBRIUM CONSTANTS •
      KJ1 = 0.6 + 1.5E-3 * (T1 - 473)   ;
      KP10 = 10**(9.218 + 3971./T1 - 7.492 * LOG10(T1) + 1.77E-3 * T1 -
                  3.11E-8 * T1**2)   ;
      KX1 = KP10/KJ1 * PRESSURE**2   ;
      KJ2 = 0.89 + 4.0E-4 * (T2-473)   ;
      KP20 = 10**(1.664 - 1850./T2)   ;
      KX2 = KP20/KJ2   ;
    • OUTLET FLOWS FROM EXTENT AND STOICHIOMETRY •
      FLOWOUT(1) = FLOWIN(1) - EXTENT1 - EXTENT2   ;
      FLOWOUT(2) = FLOWIN(2) - 2 * EXTENT1 * EXTENT2   ;
      FLOWOUT(3) = FLOWIN(3) + EXTENT1   ;
      FLOWOUT(4) = FLOWIN(4) - EXTENT2   ;
      FLOWOUT(5) = FLOWIN(5) + EXTENT2   ;
      FLOWOUT(6) = FLOWIN(6)   ;
      FLOWOUT(7) = FLOWIN(7)   ;
    • TOTAL OUTLET FLOW •
          TOTAL = SIGMA (FLOWOUT)   ;
    • MOLE FRACTIONS •
      TOTAL * X = FLOWOUT   ;
    • REACTION EQUILIBRIA •
          X(3) = X(1) * X(2)**2 * KX1   ;
    X(1) * X(4) = X(5) * X(2) * KX2   ;
****
  MODEL MAKEUPMB
    • MIXER OF TWO STREAMS (MATERIAL BALANCE ONLY) •
      SET NOCOMP
      TYPE FLOWIN1, FLOWIN2, FLOWOUT AS ARRAY (NOCOMP) OF FLOWRATE
```

```
    • [illegible] •
      FLOWOUT(1) = FLOWIN(1) * EXTENT1 * EXTENT2  ;
      FLOWOUT(2) = FLOWIN(2) - 2 * EXTENT1 * EXTENT2  ;
      FLOWOUT(3) = FLOWIN(3) * EXTENT1  ;
      FLOWOUT(4) = FLOWIN(4) - EXTENT2  ;
      FLOWOUT(5) = FLOWIN(5) * EXTENT2  ;
      FLOWOUT(6) = FLOWIN(6)  ;
      FLOWOUT(7) = FLOWIN(7)  ;
    • TOTAL OUTLET FLOW •
        TOTAL = SIGMA (FLOWOUT)  ;
    • MOLE FRACTIONS •
      TOTAL * X = FLOWOUT  ;
    • REACTION EQUILIBRIA •
        X(3) = X(1) * X(2) **2 * KX1  ;
    X(1) * X(4) = X(5) * X(2) * KX2  ;
****
  MODEL MAKEUPMB
    • MIXER OF TWO STREAMS (MATERIAL BALANCE ONLY) •
      SET NOCOMP
      TYPE FLOWIN1, FLOWIN2, FLOWOUT AS ARRAY (NOCOMP) OF FLOWRATE
           TOTAL, TOTALMAKE, TOTALR AS FLOWRATE
           X A: ARRAY (NOCOMP) OF MOLEFRACTION
           R[illegible],0 AS NOTYPE
      STREAM INPUT  1 FLOWIN1
             INPUT  2 FLOWIN2
             OUTPUT   FLOWOUT
      EQUATION
        • COMPONENT MATERIAL BALANCES •
          FLOWIN1 + FLOWIN2 = FLOWOUT;
        • TOTAL FLOW OF OUTPUT STREAM •
                    TOTAL = SIGMA (FLOWOUT);
        • MATERIAL FRACTIONS OF OUTPUT STREAM •
                X * TOTAL = FLOWOUT;
        • SPECIFICATION OF RECYCLE RATIO •
                TOTALMAKE = SIGMA (FLOWIN1)  ;
                   TOTALR = SIGMA (FLOWIN2)  ;
                   TOTALR = RATIO * TOTALMAKE  ;
****
UNIT      REACTOR IS A MEREACTOR
****
UNIT      SEPARATOR IS A MESEPARATOR
****
UNIT      PURGE IS A SPLITMB
  SET NOCOMP = 7
****
UNIT      MIXER IS A MAKEUPMB
  SET NOCOMP = 7
****
UNIT      FEEDER IS A FEEDERMB
  SET NOCOMP = 7
****
OPERATION
  • COMPONENT NUMBERS : 1 - CO, 2 - H2, 3 - CH3OH,  •
                        • 4 - H2O, 5 - CO2, 6 - CH4, 7 - N2  •
SET
  WITHIN FEEDER
    TOTAL = 6000, X(1) = 0.146, X(2) = 0.720, X(3) = 0,
    X(4) = 0.001, X(5) = 0.06, X(6) = 0.032
  WITHIN MIXER
    RATIO = 3.5
   WITHIN REACTOR
     T1 = 533.15, T2 = 543.15, PRESSURE = 90
PRESET
WITHIN SEPARATOR
   Y3 = 0.0, Y4 = 0.0
   WITHIN REACTOR
     FLOWIN(1) = 876, FLOWIN(2) = 4320, FLOWIN(3) = 0.0,
     FLOWIN(4) = 6, FLOWIN(5) = 360, FLOWIN(6) = 192,
     FLOWIN(7) = 246,
     FLOWOUT(1) = 876, FLOWOUT(2) = 4320, FLOWOUT(3) = 0.0,
     FLOWOUT(4) = 6, FLOWOUT(5) = 360, FLOWOUT(6) = 192,
     FLOWOUT(7) = 246,
     EXTENT1 = 0, EXTENT2 = 0,
   X(1) = 0.146, X(2) = 0.720, X(3) = 0, X(4) = 0.001,
   X(5) = 0.06, X(6) = 0.032, X(7) = 0.041
****
DECLARE
   TYPE •DECLARATIONS FOR LIBRARY MODELS•
      FLOWRATE = 10000.0:0.0:100000.0 UNIT="KMOL/HR" ,
      EXTENT = 0:-1000000.0:1000000.0,
      MOLEFRACTION = 0.5:0.0:1.0 ,
      PRESSURE = 10.0:0.0:1000.0 UNIT="BAR"
****
OPTIONS
    ROUTINES  ON


**********************************************************************
**********************************************************************
**                                                                  **
**                                                                  **
**        N  I  T  R  I  C                                          **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************


DECLARE
   TYPE •DECLARATIONS FOR LIBRARY MODELS•
```

```
**                                                                    **
**                                                                    **
**        N  I  T  R  I  C                                            **
**                                                                    **
**                                                                    **
**********************************************************************
**********************************************************************

DECLARE
   TYPE *DECLARATIONS FOR LIBRARY MODELS*
      FLOWRATE = 10.0:0.0:1000.0 UNIT="KMOL/HR" .
      MOLEFRACTION = 0.5:0.0:1.0.
      COEFFICIENT = 0.0:-10.0:10.0
****
   MODEL MIXERMB
    * MIXER OF TWO STREAMS (MATERIAL BALANCE ONLY)  *
      SET NOCOMP
      TYPE FLOWIN1, FLOWIN2, FLOWOUT AS ARRAY (NOCOMP) OF FLOWRATE
           TOTAL AS FLOWRATE
           X AS ARRAY (NOCOMP) OF MOLEFRACTION
      STREAM INPUT  1 FLOWIN1
             INPUT  2 FLOWIN2
             OUTPUT   FLOWOUT
      EQUATION
        *  COMPONENT MATERIAL BALANCES  *
        FLOWIN1 + FLOWIN2 = FLOWOUT;
        *  TOTAL FLOW OF OUTPUT STREAM  *
                    TOTAL = SIGMA (FLOWOUT);
```

```
        *  MATERIAL FRACTIONS OF OUTPUT STREAM  *
                  X * TOTAL = FLOWOUT;
****
   MODEL SPLITMB
   *  TWO STREAM SPLITTER (MATERIAL BALANCE ONLY)  *
      SET NOCOMP
      TYPE FLOWIN, FLOWOUT1, FLOWOUT2 AS ARRAY (NOCOMP) OF FLOWRATE
           TOTAL1, TOTAL2 AS FLOWRATE
           X1, X2 AS ARRAY (NOCOMP) OF MOLEFRACTION
           FRACTION AS MOLEFRACTION
      STREAM INPUT    FLOWIN
             OUTPUT 1 FLOWOUT1
             OUTPUT 2 FLOWOUT2
      EQUATION
        *  STREAM SPLIT  *
              FLOWOUT1 = FLOWIN * FRACTION;
        *  COMPONENT MATERIAL BALANCES  *
                FLOWIN = FLOWOUT1 + FLOWOUT2;
        *  TOTAL FLOWS OF OUTPUT STREAMS  *
                TOTAL1 = SIGMA (FLOWOUT1);
                TOTAL2 = SIGMA (FLOWOUT2);
        *  MATERIAL FRACTIONS OF OUTPUT STREAMS  *
           X1 * TOTAL1 = FLOWOUT1;
           X2 * TOTAL2 = FLOWOUT2;
****
   MODEL DISTMB
   *  TWO STREAM COMPONENT SPLITTER (MATERIAL BALANCE ONLY)  *
      SET NOCOMP
      TYPE FLOWIN, FLOWOUT1, FLOWOUT2 AS ARRAY (NOCOMP) OF FLOWRATE
           TOTAL1, TOTAL2 AS FLOWRATE
           X1, X2, FRACTION AS ARRAY (NOCOMP) OF MOLEFRACTION
      STREAM INPUT    FLOWIN
             OUTPUT 1 FLOWOUT1
             OUTPUT 2 FLOWOUT2
      EQUATION
        *  COMPONENT SPLITS  *
              FLOWOUT1 = FLOWIN * FRACTION;
        *  COMPONENT MATERIAL BALANCES  *
                FLOWIN = FLOWOUT1 + FLOWOUT2;
        *  TOTAL FLOWS OF OUTPUT STREAMS  *
                TOTAL1 = SIGMA (FLOWOUT1);
                TOTAL2 = SIGMA (FLOWOUT2);
        *  MATERIAL FRACTIONS OF OUTPUT STREAMS  *
           X1 * TOTAL1 = FLOWOUT1;
           X2 * TOTAL2 = FLOWOUT2;
****
   MODEL FEEDERMB
   *  FEED STREAM DEFINITION (MATERIAL BALANCE ONLY)  *
      SET NOCOMP
      TYPE  FLOWOUT AS ARRAY (NOCOMP) OF FLOWRATE
            X AS ARRAY (NOCOMP) OF MOLEFRACTION
            TOTAL AS FLOWRATE
      STREAM
        OUTPUT IS FLOWOUT
      EQUATION
        *  TOTAL FEED FLOW  *
                TOTAL = SIGMA (FLOWOUT);
        *  MATERIAL FRACTIONS OF FEED STREAM  *
           X * TOTAL = FLOWOUT;
****
   MODEL REACTORMB
   *  REACTOR WITH CONSTANT CONVERSION (MATERIAL BALANCE ONLY)  *
      SET NOCOMP, KEY
      TYPE FLOWIN, FLOWOUT AS ARRAY (NOCOMP) OF FLOWRATE
           X AS ARRAY (NOCOMP) OF MOLEFRACTION
           TOTAL AS FLOWRATE
```

```
           NU AS ARRAY (NOCOMP) OF COEFFICIENT
           EXTENT, CONVERSION AS NOTYPE
      STREAM INPUT  FLOWIN
```

```
****
  MODEL REACTORMB
  • REACTOR WITH CONSTANT CONVERSION (MATERIAL BALANCE ONLY)  •
    SET NOCOMP, KEY
    TYPE FLOWIN, FLOWOUT AS ARRAY (NOCOMP) OF FLOWRATE
         X AS ARRAY (NOCOMP) OF MOLEFRACTION
         TOTAL AS FLOWRATE
```

```
         NU AS ARRAY (NOCOMP) OF COEFFICIENT
         EXTENT, CONVERSION AS NOTYPE
    STREAM INPUT  FLOWIN
           OUTPUT FLOWOUT
    EQUATION
      • EXTENT OF REACTION  •
                - NU(KEY) * EXTENT = FLOWIN(KEY) * CONVERSION;
      • OUTPUT FLOWS FROM EXTENT AND STOICHIOMETRY  •
                  FLOWOUT - FLOWIN = NU * EXTENT;
      • TOTAL OUTPUT FLOW  •
                             TOTAL = SIGMA (FLOWOUT);
      • MATERIAL FRACTIONS OF OUTPUT STREAM  •
                            FLOWOUT = TOTAL * X;
****
FLOWSHEET
    OUTPUT 1 OF DIST7 IS PRODUCT 1
    OUTPUT 2 OF DIST7 IS INPUT OF DIST8
    OUTPUT 1 OF DIST8 IS PRODUCT 2
    OUTPUT 2 OF DIST8 IS INPUT 1 OF MIXR11A
    OUTPUT OF FEED10 IS INPUT 2 OF MIXR11A
    OUTPUT OF MIXR11A IS INPUT 1 OF MIXR11B
    OUTPUT OF FEED9 IS INPUT 2 OF MIXR11B
    OUTPUT OF MIXR11B IS INPUT OF REAC12
    OUTPUT OF REAC12 IS INPUT OF REAC13
    OUTPUT OF REAC13 IS INPUT OF REAC14
    OUTPUT OF REAC14 IS INPUT OF DIST15
    OUTPUT 1 OF DIST15 IS PRODUCT 3
    OUTPUT 2 OF DIST15 IS INPUT 1 OF MIXR17
    OUTPUT OF MIXR17 IS INPUT OF REAC18
    OUTPUT OF REAC18 IS INPUT OF DIST1
    OUTPUT 1 OF DIST1 IS INPUT OF DIST16
    OUTPUT 1 OF DIST16 IS INPUT 2 OF MIXR17
    OUTPUT 2 OF DIST16 IS PRODUCT 4
    OUTPUT 2 OF DIST1 IS INPUT 1 OF MIXR3
    OUTPUT OF FEED2 IS INPUT 2 OF MIXR3
    OUTPUT OF MIXR3 IS INPUT OF REAC4
    OUTPUT OF REAC4 IS INPUT 1 OF MIXR6
    OUTPUT OF FEED5 IS INPUT 2 OF MIXR6
    OUTPUT OF MIXR6 IS INPUT OF DIST7
****
UNIT       REAC4 IS A REACTORMB
    SET NOCOMP = 9, KEY = 9
****
UNIT       DIST8 IS A DISTMB
    SET NOCOMP = 9
****
UNIT       REAC13 IS A REACTORMB
    SET NOCOMP = 9, KEY = 3
****
UNIT       MIXR6 IS A MIXERMB
    SET NOCOMP = 9
****
UNIT       FEED9 IS A FEEDERMB
    SET NOCOMP = 9
****
UNIT       MIXR3 IS A MIXERMB
    SET NOCOMP = 9
****
UNIT       REAC18 IS A REACTORMB
    SET NOCOMP = 9, KEY = 2
****
UNIT       DIST7 IS A DISTMB
    SET NOCOMP = 9
****
UNIT       REAC14 IS A REACTORMB
    SET NOCOMP = 9, KEY = 3
```

```
****
UNIT       DIST15 IS A DISTMB
    SET NOCOMP = 9
****
UNIT       FEED5 IS A FEEDERMB
    SET NOCOMP = 9
****
UNIT       REAC12 IS A REACTORMB
    SET NOCOMP = 9, KEY = 4
****
UNIT       MIXR11B IS A MIXERMB
    SET NOCOMP = 9
****
UNIT       MIXR17 IS A MIXERMB
    SET NOCOMP = 9
****
UNIT       DIST1 IS A DISTMB
    SET NOCOMP = 9
****
UNIT       FEED2 IS A FEEDERMB
    SET NOCOMP = 9
****
UNIT       FEED10 IS A FEEDERMB
    SET NOCOMP = 9
****
```

```
****
UNIT      MIXR17 IS A MIXERMB
   SET NOCOMP = 9
****
UNIT      DIST1 IS A DISTMB
   SET NOCOMP = 9
****
UNIT      FEED2 IS A FEEDERMB
   SET NOCOMP = 9
****
UNIT      FEED10 IS A FEEDERMB
   SET NOCOMP = 9
****
UNIT      MIXR11A IS A MIXERMB
   SET NOCOMP = 9
****
UNIT      DIST16 IS A DISTMB
   SET NOCOMP = 9
****
OPERATION
   *  COMPONENT NUMBERS - 1 : HNO3 , 2 : N2 , 3 : O2  *
   *                      4 : H2O  , 5 : CHH, 6 : CO  *
   *                      7 : CO2  , 8 : H2 , 9 : NH3 *
   SET
      WITHIN DIST1
SET
   WITHIN DIST1
      FRACTION(1) = 0, FRACTION(2) = 1, FRACTION(3) = 0,
      FRACTION(4) = 0, FRACTION(5) = 0, FRACTION(6) = 0,
      FRACTION(7) = 0, FRACTION(8) = 1
   WITHIN FEED2
      X(1) = 0, X(2) = 0.79, X(3) = 0.21, X(4) = 0, X(5) = 0,
      X(6) = 0, X(7) = 0, X(8) = 0
   WITHIN REAC4
      NU(1) = 1, NU(2) = 0, NU(3) = -2, NU(4) = 1, NU(5) = 0,
      NU(6) = 0, NU(7) = 0, NU(8) = 0, NU(9) = -1,
      CONVERSION = 1
   WITHIN FEED5
      X(1) = 0, X(2) = 0, X(3) = 0, X(4) = 1, X(5) = 0,
      X(6) = 0, X(7) = 0, X(8) = 0
   WITHIN DIST7
      FRACTION(1) = 1, FRACTION(2) = 0, FRACTION(3) = 0,
      FRACTION(4) = 1, FRACTION(5) = 0, FRACTION(6) = 0,
      FRACTION(7) = 0, FRACTION(8) = 0, FRACTION(9) = 0,
   *  DESIGN SPECIFICATIONS 1, 2, 6  *
      FLOWOUT1(1) = 0.432, FLOWOUT1(4) = 1.008, X2(2) = 0.992
   WITHIN DIST8
      FRACTION(1) = 0, FRACTION(4) = 0, FRACTION(5) = 0,
      FRACTION(6) = 0, FRACTION(7) = 0, FRACTION(8) = 0,
      FRACTION(9) = 0,
   *  DESIGN SPECIFICATION 7  *
      X2(3) = 0.01
   WITHIN FEED9
      X(1) = 0, X(2) = 0, X(3) = 0, X(4) = 1, X(5) = 0,
      X(6) = 0, X(7) = 0, X(8) = 0
   WITHIN FEED10
      X(1) = 0, X(2) = 0, X(3) = 0, X(4) = 0, X(5) = 1,
      X(6) = 0, X(7) = 0, X(8) = 0
   WITHIN REAC12
      NU(1) = 0, NU(2) = 0, NU(3) = 0.5, NU(4) = -1, NU(5) = 0,
      NU(6) = 0, NU(7) = 0, NU(8) = 1, NU(9) = 0
   WITHIN REAC13
      NU(1) = 0, NU(2) = 0, NU(3) = -1, NU(4) = 0, NU(5) = -2,
      NU(6) = 2, NU(7) = 0, NU(8) = 4, NU(9) = 0
   WITHIN REAC14
      NU(1) = 0, NU(2) = 0, NU(3) = -1, NU(4) = 0, NU(5) = -1,
      NU(6) = 0, NU(7) = 1, NU(8) = 2, NU(9) = 0,
   *  DESIGN SPECIFICATIONS 3, 4, 5  *
      FLOWOUT(3) = 0, FLOWOUT(4) = 0, FLOWOUT(5) = 0
   WITHIN DIST15
      FRACTION(1) = 0, FRACTION(2) = 0, FRACTION(3) = 0,
      FRACTION(4) = 0, FRACTION(5) = 0, FRACTION(6) = 1,
      FRACTION(7) = 1, FRACTION(8) = 0, FRACTION(9) = 0,
   *  DESIGN SPECIFICATION 8  *
      X1(7) = 0.91
   WITHIN DIST16
      FRACTION(1) = 0, FRACTION(2) = 1, FRACTION(3) = 0,
      FRACTION(4) = 0, FRACTION(5) = 0, FRACTION(6) = 0,
      FRACTION(7) = 0, FRACTION(8) = 0.99, FRACTION(9) = 1
   WITHIN MIXR17
   *  DESIGN SPECIFICATIONS 9, 10  *
      X(2) = 0.25, X(8) = 0.75
   WITHIN REAC18
      NU(1) = 0, NU(2) = -1, NU(3) = 0, NU(4) = 0, NU(5) = 0,
      NU(6) = 0, NU(7) = 0, NU(8) = -3, NU(9) = 2,
      CONVERSION = 0.25
****
TITLE
      REKLAITIS / SOOD AMMONIA - NITRIC ACID COMPLEX
****
OPTIONS  ROUTINES ON
  PRINT MODEL 2 FLOWSHEET 2 UNIT 2 OPERATION 2 DECLARE 2
****



**********************************************************************
**********************************************************************
**                                                                  **
**                                                                  **
**        B  P  N  E  T                                             **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************
```

```
**********************************************************************
**                                                                  **
**                                                                  **
**        B P N E T                                                 **
**                                                                  **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************


FLOWSHEET
INPUT OF D30 IS FEED 1
OUTPUT 1 OF D30 IS INPUT 2 OF H17
OUTPUT 2 OF D30 IS INPUT OF D31
INPUT 1 OF H17 IS FEED 2
OUTPUT 1 OF H17 IS PRODUCT 1
OUTPUT 2 OF H17 IS INPUT 2 OF H16
OUTPUT 1 OF D31 IS INPUT 1 OF H15
OUTPUT 2 OF D31 IS INPUT OF D32
OUTPUT 1 OF D32 IS INPUT 2 OF H14
OUTPUT 2 OF D32 IS INPUT OF D33
OUTPUT 1 OF D33 IS INPUT 2 OF H13
OUTPUT 2 OF D33 IS INPUT OF D34
OUTPUT 1 OF D34 IS INPUT 2 OF H12
OUTPUT 2 OF D34 IS INPUT 2 OF H11
INPUT 1 OF H11 IS FEED 3
OUTPUT 1 OF H11 IS PRODUCT 2
OUTPUT 2 OF H11 IS INPUT 1 OF M35
INPUT OF D43 IS FEED 4
OUTPUT 1 OF D43 IS INPUT 1 OF H16
OUTPUT 2 OF D43 IS INPUT 2 OF H20
OUTPUT 1 OF H16 IS INPUT 1 OF MM4
OUTPUT 2 OF H16 IS INPUT 6 OF M35
INPUT 1 OF H20 IS FEED 5
OUTPUT 1 OF H20 IS PRODUCT 4
OUTPUT 2 OF H20 IS INPUT 2 OF MM4
OUTPUT OF MM4 IS PRODUCT 5
INPUT OF D45 IS FEED 6
OUTPUT 1 OF D45 IS INPUT 1 OF H10
OUTPUT 2 OF D45 IS INPUT 2 OF H18
INPUT 1 OF H18 IS FEED 7
OUTPUT 1 OF H18 IS PRODUCT 6
OUTPUT 2 OF H18 IS INPUT 2 OF H19
INPUT 1 OF H19 IS FEED 8
OUTPUT 1 OF H19 IS PRODUCT 7
OUTPUT 2 OF H19 IS INPUT 2 OF MM8
INPUT 1 OF H1 IS OUTPUT OF MM2
INPUT 2 OF H1 IS FEED 9
OUTPUT 1 OF H1 IS PRODUCT 8
OUTPUT 2 OF H1 IS INPUT 2 OF H3
INPUT OF D38 IS OUTPUT OF M35
OUTPUT 1 OF D38 IS INPUT OF D39
OUTPUT 2 OF D38 IS INPUT OF D37
OUTPUT 1 OF D37 IS INPUT 2 OF H8
OUTPUT 2 OF D37 IS INPUT OF D36
INPUT 1 OF H8 IS FEED 10
OUTPUT 1 OF H8 IS INPUT 1 OF H14
OUTPUT 2 OF H8 IS INPUT 3 OF MM0
OUTPUT 1 OF H14 IS PRODUCT 9
OUTPUT 2 OF H14 IS INPUT 4 OF M35
OUTPUT 1 OF D39 IS INPUT 2 OF H7
OUTPUT 2 OF D39 IS INPUT 2 OF H6
INPUT 1 OF H6 IS FEED 11
OUTPUT 1 OF H6 IS INPUT 1 OF H12
OUTPUT 2 OF H6 IS INPUT 5 OF MM0
OUTPUT 1 OF H12 IS PRODUCT 10
OUTPUT 2 OF H12 IS INPUT 2 OF M35
OUTPUT 1 OF D36 IS INPUT 1 OF H9
OUTPUT 2 OF D36 IS INPUT 2 OF H10
OUTPUT 1 OF H10 IS INPUT 1 OF MM8
OUTPUT 2 OF H10 IS INPUT 1 OF MM0
INPUT 2 OF MM0 IS OUTPUT 1 OF H9
INPUT 4 OF MM0 IS OUTPUT 2 OF H7
OUTPUT OF MM0 IS INPUT OF D41
OUTPUT 1 OF D41 IS INPUT 1 OF HH
OUTPUT 2 OF D41 IS INPUT 2 OF H5
INPUT 1 OF H5 IS FEED 12
OUTPUT 1 OF H5 IS INPUT 1 OF H7
OUTPUT 2 OF H5 IS INPUT 1 OF H3
OUTPUT 1 OF H7 IS INPUT 1 OF H13
OUTPUT 1 OF H13 IS PRODUCT 11
OUTPUT 2 OF H13 IS INPUT 3 OF M35
OUTPUT 1 OF H3 IS INPUT 1 OF MM2
OUTPUT 2 OF H3 IS INPUT 2 OF HH
OUTPUT 1 OF HH IS INPUT 2 OF H2
OUTPUT 2 OF HH IS INPUT 2 OF H9
INPUT 1 OF H2 IS FEED 13
OUTPUT 1 OF H2 IS PRODUCT 12
OUTPUT 2 OF H2 IS INPUT 2 OF MM2
OUTPUT 2 OF H9 IS INPUT 2 OF H15
OUTPUT 1 OF H15 IS INPUT 5 OF M35
OUTPUT 2 OF H15 IS PRODUCT 13
OUTPUT OF MM8 IS PRODUCT 3
****
MODEL DIVIDER
TYPE F,F1,F2 AS FLOWRATE
     T,T1,T2 AS TEMPERATURE
     FRAC,C,C1,C2 AS NOTYPE
STREAM
  INPUT   IS F,T,C
  OUTPUT 1 IS F1,T1,C1
```

OUTPUT 1 OF H15 IS INPUT 5 OF H25

```
OUTPUT 2 OF H15 IS PRODUCT 13
OUTPUT OF MH8 IS PRODUCT 3
****
MODEL DIVIDER
TYPE F,F1,F2 AS FLOWRATE
     T,T1,T2 AS TEMPERATURE
     FRAC,C,C1,C2 AS NOTYPE
STREAM
  INPUT   IS F,T,C
  OUTPUT 1 IS F1,T1,C1
  OUTPUT 2 IS F2,T2,C2
EQUATION
  T1=T2=T;
  C1=C2=C;
  F = F1 + F2;
  FRAC * F = F1;
****
MODEL HEATEX
TYPE S1,S2,S10,S20 AS FLOWRATE
     TS1I,TS2I,TS10,TS20  AS TEMPERATURE
     CP1,CP2,CP10,CP20 AS NOTYPE
     UA AS COEFFICIENT
STREAM
  INPUT 1 IS S1,TS1I,CP1
  INPUT 2 IS S2,TS2I,CP2
  OUTPUT 1 IS S10,TS10,CP10
  OUTPUT 2 IS S20,TS20,CP20
EQUATION
  S10=S1;
  S20=S2;
  CP10=CP1;
  CP20=CP2;
PROCEDURE
  (TS10,TS20) SIMHEX (S1,TS1I,CP1,S2,TS2I,CP2,UA)
****
MODEL MIX2S
TYPE F1,F2,F AS FLOWRATE
     T1,T2,T AS TEMPERATURE
     C1,C2,C AS NOTYPE
STREAM
  INPUT 1 IS F1,T1,C1
  INPUT 2 IS F2,T2,C2
  OUTPUT  IS F,T,C
EQUATION
  F = F1 + F2;
*  MIXING EQNS. FOR C AND T (ASSUMING STREAMS OF SAME COMP.)  *
      C = C1   ;
  F * T = F1 * T1 +F2 * T2   ;
****
MODEL MIX5S
TYPE F1,F2,F3,F4,F5,F AS FLOWRATE
     T1,T2,T3,T4,T5,T AS TEMPERATURE
     C1,C2,C3,C4,C5,C AS NOTYPE
STREAM
  INPUT 1 IS F1,T1,C1
  INPUT 2 IS F2,T2,C2
  INPUT 3 IS F3,T3,C3
  INPUT 4 IS F4,T4,C4
  INPUT 5 IS F5,T5,C5
  OUTPUT  IS F,T,C
EQUATION
  F = F1 + F2 + F3 + F4 + F5;
*  MIXING OF C, T (ASSUMING STREAMS OF SAME COMP.)   *
      C = C1;
  F * T = F1 * T1 + F2 * T2 + F3 * T3 + F4 * T4 + F5 * T5   ;
****
```

```
MODEL MIX6S
TYPE F1,F2,F3,F4,F5,F6,F AS FLOWRATE
     T1,T2,T3,T4,T5,T6,T AS TEMPERATURE
     C1,C2,C3,C4,C5,C6,C AS NOTYPE
STREAM
  INPUT 1 IS F1,T1,C1
  INPUT 2 IS F2,T2,C2
  INPUT 3 IS F3,T3,C3
  INPUT 4 IS F4,T4,C4
  INPUT 5 IS F5,T5,C5
  INPUT 6 IS F6,T6,C6
  OUTPUT IS F,T,C
EQUATION
  F = F1 + F2 + F3 + F4 + F5 + F6;
*   MIXING OF C, T (ASSUMING STREAMS OF SAME COMP.)   *
      C = C1   ;
 F * T  = F1*T1 + F2*T2 + F3*T3 + F4*T4 + F5*T5 + F6*T6   ;
****
UNIT H13 IS A HEATEX
****
UNIT H18 IS A HEATEX
****
UNIT D43 IS A DIVIDER
****
UNIT H5 IS A HEATEX
****
UNIT D32 IS A DIVIDER
****
UNIT H10 IS A HEATEX
****
UNIT D37 IS A DIVIDER
****
UNIT H2 IS A HEATEX
****
UNIT H15 IS A HEATEX
```

```
UNIT H5 IS A HEATEX
****
UNIT D32 IS A DIVIDER
****
UNIT H10 IS A HEATEX
****
UNIT D37 IS A DIVIDER
****
UNIT H2 IS A HEATEX
****
UNIT H15 IS A HEATEX
****
UNIT H9 IS A HEATEX
****
UNIT D45 IS A DIVIDER
****
UNIT M35 IS A MIX6S
****
UNIT H6 IS A HEATEX
****
UNIT D34 IS A DIVIDER
****
UNIT D39 IS A DIVIDER
****
UNIT H17 IS A HEATEX
****
UNIT H12 IS A HEATEX
****
UNIT MM8 IS A MIX2S
****
UNIT H20 IS A HEATEX
****
UNIT H3 IS A HEATEX
****
UNIT MM0 IS A MIX3S
****
UNIT D31 IS A DIVIDER
****
UNIT D36 IS A DIVIDER
****
UNIT H14 IS A HEATEX
****
```

```
UNIT H19 IS A HEATEX
****
UNIT H7 IS A HEATEX
****
UNIT MM2 IS A MIX2S
****
UNIT D33 IS A DIVIDER
****
UNIT H11 IS A HEATEX
****
UNIT H16 IS A HEATEX
****
UNIT D38 IS A DIVIDER
****
UNIT MM IS A HEATEX
****
UNIT D41 IS A DIVIDER
****
UNIT H1 IS A HEATEX
****
UNIT H8 IS A HEATEX
****
UNIT D30 IS A DIVIDER
****
UNIT MM4 IS A MIX2S
****
OPERATION
SET
*  SPECIFICATION OF FEED STREAMS  *
WITHIN D30
  F=480,T=25,C=0.555
WITHIN D45
  F=480,T=219,C=2.912
WITHIN H18
  S1=290,CP1=1.393,TS11=159
WITHIN H1
  S2=280,TS21=352,CP2=0.523
WITHIN H20
  S1=160,TS11=41,CP1=1.068
WITHIN H5
  S1=110,TS11=263,CP1=0.655
WITHIN H6
  S1=83,TS11=206,CP1=0.659
WITHIN H8
  S1=39,TS11=304,CP1=0.674
WITHIN H2
  S1=510,TS11=315,CP1=0.686
WITHIN H19
  S1=130,TS11=142,CP1=0.642
WITHIN D43
  F=580,T=152,C=4.842
WITHIN H17
  S1=160,TS11=133,CP1=1.681
WITHIN H11
  S1=38,TS11=170,CP1=0.701
*   SPECIFICATION OF PRODUCT STREAM TEMPERATURES   *
WITHIN H17 TS10=83.9
WITHIN H11 TS10=107.6
WITHIN MM8 T=197.2
```

```
S1=100,TS11=153,CP1=0.701
WITHIN H11
  S1=38,TS11=170,CP1=0.701
•   SPECIFICATION OF PRODUCT STREAM TEMPERATURES   •
WITHIN H17 TS10=83.9
WITHIN H11 TS10=107.6
WITHIN MM8 T=197.2
WITHIN H20 TS10=133.1
WITHIN H18 TS10=203.4
WITHIN H19 TS10=199.7
WITHIN H1 TS10=306.5
WITHIN H14 TS10=70.7
WITHIN H12 TS10=111.0
WITHIN H13 TS10=84.9
```

```
WITHIN H16 TS20=69.0
•   SPECIFICATION OF OTHER MEASURED TEMPERATURES   •
WITHIN H2 TS20=296.0
WITHIN MM TS20=215.3,TS10=211.3
WITHIN H6 TS10=170.2
WITHIN H7 TS20=209.1, TS10=162.7
WITHIN H8 TS20=273.9,TS10=156.6
WITHIN H9 TS10=201.0,TS20=151.9
WITHIN H10 TS20=176.9,TS10=193.6
WITHIN H11 TS20=161.0
WITHIN H12 TS20=152.2
WITHIN H13 TS20=136.3
WITHIN H14 TS20=108.6
WITHIN H15 TS10=120.7
WITHIN H16 TS20=126.8
WITHIN H20 TS20=142.6
WITHIN H3 TS10=281.7
WITHIN H5 TS20=219.68
PRESET
•   INITIAL GUESSES FOR BLOCK 138   •
   WITHIN H7 TS11=250
   WITHIN H3 TS21=300, TS20=250
****
TITLE
 BP HEAT EXCHANGER NETWORK
****
DECLARE
    TYPE FLOWRATE=1.0E2:0:1.0E3  UNIT="KMOLES",
         TEMPERATURE=200:-100:500 UNIT="DEGC",
         COEFFICIENT=1E2:0:1E3 UNIT="H.T.C.",
         NOTYPE=1:0:1.0E13 UNIT="NOUNITS"
****
OPTIONS
  ROUTINES ON
****
```

```
**********************************************************************
**********************************************************************
**                                                                  **
**                                                                  **
**        B  P  N  E  T  E                                          **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************
```

```
FLOWSHEET
INPUT OF D30 IS FEED 1
OUTPUT 1 OF D30 IS INPUT 2 OF H17
OUTPUT 2 OF D30 IS INPUT OF D31
INPUT 1 OF H17 IS FEED 2
OUTPUT 1 OF H17 IS PRODUCT 1
OUTPUT 2 OF H17 IS INPUT 2 OF H16
OUTPUT 1 OF D31 IS INPUT 1 OF H15
OUTPUT 2 OF D31 IS INPUT OF D32
OUTPUT 1 OF D32 IS INPUT 2 OF H14
OUTPUT 2 OF D32 IS INPUT OF D33
OUTPUT 1 OF D33 IS INPUT 2 OF H13
OUTPUT 2 OF D33 IS INPUT OF D34
OUTPUT 1 OF D34 IS INPUT 2 OF H12
OUTPUT 2 OF D34 IS INPUT 2 OF H11
INPUT 1 OF H11 IS FEED 3
OUTPUT 1 OF H11 IS PRODUCT 2
```

```
OUTPUT 2 OF H11 IS INPUT 1 OF M35
INPUT OF D43 IS FEED 4
OUTPUT 1 OF D43 IS INPUT 1 OF H16
OUTPUT 2 OF D43 IS INPUT 2 OF H20
OUTPUT 1 OF H16 IS INPUT 1 OF MM4
OUTPUT 2 OF H16 IS INPUT 6 OF M35
INPUT 1 OF H20 IS FEED 5
OUTPUT 1 OF H20 IS PRODUCT 4
OUTPUT 2 OF H20 IS INPUT 2 OF MM4
OUTPUT OF MM4 IS PRODUCT 5
INPUT OF D45 IS FEED 6
OUTPUT 1 OF D45 IS INPUT 1 OF H10
OUTPUT 2 OF D45 IS INPUT 2 OF H18
INPUT 1 OF H18 IS FEED 7
OUTPUT 1 OF H18 IS PRODUCT 6
OUTPUT 2 OF H18 IS INPUT 2 OF H19
INPUT 1 OF H19 IS FEED 8
OUTPUT 1 OF H19 IS PRODUCT 7
OUTPUT 2 OF H19 IS INPUT 2 OF MM8
INPUT 1 OF H1 IS OUTPUT OF MM2
INPUT 2 OF H1 IS FEED 9
OUTPUT 1 OF H1 IS PRODUCT 8
```

OUTPUT 2 OF H11 IS INPUT 1 OF M35
INPUT OF D43 IS FEED 4
OUTPUT 1 OF D43 IS INPUT 1 OF H16
OUTPUT 2 OF D43 IS INPUT 2 OF H20
OUTPUT 1 OF H16 IS INPUT 1 OF M44
OUTPUT 2 OF H16 IS INPUT 6 OF M35
INPUT 1 OF H20 IS FEED 5
OUTPUT 1 OF H20 IS PRODUCT 4
OUTPUT 2 OF H20 IS INPUT 2 OF M44
OUTPUT OF M44 IS PRODUCT 5
INPUT OF D45 IS FEED 6
OUTPUT 1 OF D45 IS INPUT 1 OF H10
OUTPUT 2 OF D45 IS INPUT 2 OF H18
INPUT 1 OF H18 IS FEED 7
OUTPUT 1 OF H18 IS PRODUCT 6
OUTPUT 2 OF H18 IS INPUT 2 OF H19
INPUT 1 OF H19 IS FEED 8
OUTPUT 1 OF H19 IS PRODUCT 7
OUTPUT 2 OF H19 IS INPUT 2 OF M48
INPUT 1 OF H1 IS OUTPUT OF M42
INPUT 2 OF H1 IS FEED 9
OUTPUT 1 OF H1 IS PRODUCT 8
OUTPUT 2 OF H1 IS INPUT 2 OF H3
INPUT OF D38 IS OUTPUT OF M35
OUTPUT 1 OF D38 IS INPUT OF D39
OUTPUT 2 OF D38 IS INPUT OF D37
OUTPUT 1 OF D37 IS INPUT 2 OF H8
OUTPUT 2 OF D37 IS INPUT OF D36
INPUT 1 OF H8 IS FEED 10
OUTPUT 1 OF H8 IS INPUT 1 OF H14
OUTPUT 2 OF H8 IS INPUT 3 OF M40
OUTPUT 1 OF H14 IS PRODUCT 9
OUTPUT 2 OF H14 IS INPUT 4 OF M35
OUTPUT 1 OF D39 IS INPUT 2 OF H7
OUTPUT 2 OF D39 IS INPUT 2 OF H6
INPUT 1 OF H6 IS FEED 11
OUTPUT 1 OF H6 IS INPUT 1 OF H12
OUTPUT 2 OF H6 IS INPUT 5 OF M40
OUTPUT 1 OF H12 IS PRODUCT 10
OUTPUT 2 OF H12 IS INPUT 2 OF M35
OUTPUT 1 OF D36 IS INPUT 1 OF H9
OUTPUT 2 OF D36 IS INPUT 2 OF H10
OUTPUT 1 OF H10 IS INPUT 1 OF M48
OUTPUT 2 OF H10 IS INPUT 1 OF M40
INPUT 2 OF M40 IS OUTPUT 1 OF H9
INPUT 4 OF M40 IS OUTPUT 2 OF H7
OUTPUT OF M40 IS INPUT OF D41
OUTPUT 1 OF D41 IS INPUT 1 OF M1
OUTPUT 2 OF D41 IS INPUT 2 OF H5
INPUT 1 OF H5 IS FEED 12
OUTPUT 1 OF H5 IS INPUT 1 OF H7
OUTPUT 2 OF H5 IS INPUT 1 OF H3
OUTPUT 1 OF H7 IS INPUT 1 OF H13
OUTPUT 1 OF H13 IS PRODUCT 11
OUTPUT 2 OF H13 IS INPUT 3 OF M35
OUTPUT 1 OF H3 IS INPUT 1 OF M42
OUTPUT 2 OF H3 IS INPUT 2 OF M1
OUTPUT 1 OF M1 IS INPUT 2 OF H2
OUTPUT 2 OF M1 IS INPUT 2 OF H9
INPUT 1 OF H2 IS FEED 13
OUTPUT 1 OF H2 IS PRODUCT 12
OUTPUT 2 OF H2 IS INPUT 2 OF M42
OUTPUT 2 OF H9 IS INPUT 2 OF H15
OUTPUT 1 OF H15 IS INPUT 5 OF M35
OUTPUT 2 OF H15 IS PRODUCT 13
OUTPUT OF M48 IS PRODUCT 3

```
{****
MODEL DIVIDER
TYPE F,F1,F2 AS FLOWRATE
     T,T1,T2 AS TEMPERATURE
     FRAC,C,C1,C2 AS NOTYPE
STREAM
  INPUT   IS F,T,C
  OUTPUT 1 IS F1,T1,C1
  OUTPUT 2 IS F2,T2,C2
EQUATION
  T1=T2=T;
  C1=C2=C;
  F = F1 + F2;
  FRAC * F = F1;
{****
MODEL HEATEX
TYPE S1,S2,S10,S20 AS FLOWRATE
     TS1I,TS2I,TS10,TS20  AS TEMPERATURE
     CP1,CP2,CP10,CP20,UA AS NOTYPE
     Q,DELTATLM AS NOTYPE2
     ARG AS ARGUMENT
STREAM
  INPUT 1 IS S1,TS1I,CP1
  INPUT 2 IS S2,TS2I,CP2
  OUTPUT 1 IS S10,TS10,CP10
  OUTPUT 2 IS S20,TS20,CP20
EQUATION
  CP10=CP1;
  CP20=CP2;
  S10 = S1;
  S20 = S2;
  Q = S1 * CP1 * (TS1I-TS10);
  Q = S2 * CP2 * (TS20-TS2I);
  Q = UA * DELTATLM;
  ARG * (TS1I-TS20) = TS10-TS2I;
  DELTATLM * LOG(ARG) = (TS10-TS2I)-(TS1I-TS20)
{****
MODEL MIX2S
TYPE F1,F2,F AS FLOWRATE
     T1,T2,T AS TEMPERATURE
     C1,C2,C AS NOTYPE
STREAM
  INPUT 1 IS F1,T1,C1
  INPUT 2 IS F2,T2,C2
  OUTPUT  IS F,T,C
EQUATION
  F = F1 + F2;
{  MIXING EQNS. FOR C AND T (ASSUMING STREAMS OF SAME COMP.)  }
     C = C1    ;
  F * T = F1 * T1 +F2 * T2    ;
{****
MODEL MIX5S
TYPE F1,F2,F3,F4,F5,F AS FLOWRATE
     T1,T2,T3,T4,T5,T AS TEMPERATURE
     C1,C2,C3,C4,C5,C AS NOTYPE
STREAM
  INPUT 1 IS F1,T1,C1
  INPUT 2 IS F2,T2,C2
  INPUT 3 IS F3,T3,C3
  INPUT 4 IS F4,T4,C4
  INPUT 5 IS F5,T5,C5
  OUTPUT  IS F,T,C
EQUATION
  F = F1 + F2 + F3 + F4 + F5;
{  MIXING OF C, T (ASSUMING STREAMS OF SAME COMP.)   }
     C = C1;
```

```
  F * T = F1 * T1 + F2 * T2 + F3 * T3 + F4 * T4 + F5 * T5    ;
{****
MODEL MIX6S
TYPE F1,F2,F3,F4,F5,F6,F AS FLOWRATE
     T1,T2,T3,T4,T5,T6,T AS TEMPERATURE
     C1,C2,C3,C4,C5,C6,C AS NOTYPE
STREAM
  INPUT 1 IS F1,T1,C1
  INPUT 2 IS F2,T2,C2
  INPUT 3 IS F3,T3,C3
  INPUT 4 IS F4,T4,C4
  INPUT 5 IS F5,T5,C5
  INPUT 6 IS F6,T6,C6
  OUTPUT IS F,T,C
EQUATION
  F = F1 + F2 + F3 + F4 + F5 + F6;
{   MIXING OF C, T  ASSUMING STREAMS OF SAME COMP.)   }
     C = C1    ;
 F * T  = F1*T1 + F2*T2 + F3*T3 + F4*T4 + F5*T5 + F6*T6    ;
{****
UNIT H13 IS A HEATEX
{****
UNIT H18 IS A HEATEX
{****
UNIT D43 IS A DIVIDER
{****
UNIT H5 IS A HEATEX
{****
UNIT D32 IS A DIVIDER
{****
UNIT H10 IS A HEATEX
{****
UNIT D37 IS A DIVIDER
{****
UNIT H2 IS A HEATEX
```

```
 F • T  • F1•T1 • F2•T2 • F3•T3 • F4•T4 • F5•T5 • F6•T6    ;
****
UNIT H13 IS A HEATEX
****
UNIT H18 IS A HEATEX
****
UNIT D43 IS A DIVIDER
****
UNIT H8 IS A HEATEX
****
UNIT D32 IS A DIVIDER
****
UNIT H10 IS A HEATEX
****
UNIT D37 IS A DIVIDER
****
UNIT H2 IS A HEATEX
****
UNIT H15 IS A HEATEX
****
UNIT H9 IS A HEATEX
****
UNIT D45 IS A DIVIDER
****
UNIT M35 IS A MIX6S
****
UNIT H6 IS A HEATEX
****
UNIT D34 IS A DIVIDER
****
UNIT D39 IS A DIVIDER
****
UNIT H17 IS A HEATEX
****
UNIT H12 IS A HEATEX
****
UNIT MH8 IS A MIX2S
****
UNIT H20 IS A HEATEX
****
UNIT H3 IS A HEATEX
****
UNIT MH0 IS A MIX3S
****
UNIT D31 IS A DIVIDER
****
UNIT D36 IS A DIVIDER
****
UNIT H14 IS A HEATEX
****
UNIT H19 IS A HEATEX
****
UNIT H7 IS A HEATEX
****
UNIT MH2 IS A MIX2S
****
UNIT D33 IS A DIVIDER
****
UNIT H11 IS A HEATEX
****
UNIT H16 IS A HEATEX
****
UNIT D38 IS A DIVIDER
****
UNIT MH IS A HEATEX
****
UNIT D41 IS A DIVIDER
****
UNIT H1 IS A HEATEX
****
UNIT H8 IS A HEATEX
****
UNIT D30 IS A DIVIDER
****
UNIT MH4 IS A MIX2S
****
OPERATION
SET
•  SPECIFICATION OF FEED STREAMS  •
WITHIN D30
   F=680,T=25,C=3.555
WITHIN D45
   F=480,T=219,C=2.912
WITHIN H18
   S1=290,CP1=1.393,TS11=159
WITHIN H1
   S2=250,TS21=352,CP2=0.523
WITHIN H20
   S1=160,TS11=41,CP1=1.068
WITHIN H5
   S1=110,TS11=263,CP1=0.655
WITHIN H6
   S1=83,TS11=206,CP1=0.659
WITHIN H8
   S1=39,TS11=304,CP1=0.674
WITHIN H2
   S1=610,TS11=315,CP1=0.686
WITHIN H19
   S1=130,TS11=142,CP1=0.642
WITHIN D43
   F=580,T=152,C=4.842
WITHIN H17
   S1=160,TS11=133,CP1=1.681
```

```
WITHIN H1
  S2=250,TS21=352,CP2=0.523
WITHIN H20
  S1=160,TS11=41,CP1=1.068
WITHIN H5
  S1=110,TS11=263,CP1=0.655
WITHIN H6
  S1=83,TS11=206,CP1=0.659
WITHIN H8
  S1=39,TS11=304,CP1=0.674
WITHIN H2
  S1=510,TS11=315,CP1=0.686
WITHIN H19
  S1=130,TS11=142,CP1=0.642
WITHIN D43
  F=580,T=152,C=4.842
WITHIN H17
  S1=160,TS11=133,CP1=1.681
WITHIN H11
  S1=38,TS11=170,CP1=0.701
*   SPECIFICATION OF PRODUCT STREAM TEMPERATURES   *
WITHIN H17 TS10=83.9
WITHIN H11 TS10=107.6
WITHIN MH8 T=197.2
WITHIN H20 TS10=133.1
WITHIN H18 TS10=203.4
WITHIN H19 TS10=199.7
WITHIN H1 TS10=306.5
WITHIN H14 TS10=70.7
WITHIN H12 TS10=111.0
WITHIN H13 TS10=84.9
WITHIN H15 TS20=69.0
*   SPECIFICATION OF OTHER MEASURED TEMPERATURES   *
WITHIN H2 TS20=296.0
WITHIN HN TS20=215.3,TS10=211.3
WITHIN H6 TS10=170.2
WITHIN H7 TS20=209.1, TS10=162.7
WITHIN H8 TS20=273.9,TS10=155.6
WITHIN H9 TS10=201.0,TS20=151.9
WITHIN H10 TS20=176.9,TS10=193.6
WITHIN H11 TS20=161.0
WITHIN H12 TS20=152.2
WITHIN H13 TS20=136.3
WITHIN H14 TS20=108.6
WITHIN H15 TS10=120.7
WITHIN H16 TS20=126.8
WITHIN H20 TS20=142.6
WITHIN H3 TS10=281.7
WITHIN H5 TS20=219.68
****
TITLE
 BP HEAT EXCHANGER NETWORK
****
DECLARE
    TYPE FLOWRATE=1.0E2:0:1.0E3  UNIT="KMOLES",
         TEMPERATURE=200:-100:500 UNIT="DEGC",
         COEFFICIENT=1E2:0:1E3 UNIT="H.T.C.",
         ARGUMENT=1:1.E-10:1.E10,
         NOTYPE2=1:-1.E10:1.E10,
         NOTYPE=1:0:1.0E13 UNIT="NOUNITS"
****
OPTIONS
  ROUTINES ON
****


**********************************************************************
**********************************************************************
**                                                                  **
**                                                                  **
**        E  T  H  Y  L                                             **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************


TITLE
 SIMPLE ETHYLENE PLANT
****
MODEL REACTOR
SET KEY
ARRAY REIN(7),REOU(7),STO(7)
STREAM
  INPUT IS REIN
  OUTPUT IS REOU
EQUATION
KEYCOMP=REIN(KEY)*CONVER
REOU=REIN*STO*KEYCOMP
****
MODEL MIXER
ARRAY MIXIN1(7),MIXIN2(7),MIXOUT(7)
STREAM
  INPUT 1 IS MIXIN1
  INPUT 2 IS MIXIN2
  OUTPUT  IS MIXOUT
EQUATION
MIXOUT=MIXIN1+MIXIN2
****
MODEL SIMPLEDIST
```

```
        :NOTYPE2=1:-1.E10:1.E10
        NOTYPE=1:0:1.0E13 UNIT="MOUNITS"
****
OPTIONS
  ROUTINES ON
****



*************************************************************************
*************************************************************************
**                                                                     **
**                                                                     **
**       E T H Y L                                                     **
**                                                                     **
**                                                                     **
*************************************************************************
*************************************************************************


TITLE
 SIMPLE ETHYLENE PLANT
****
MODEL REACTOR
SET KEY
ARRAY REIN(7),REOU(7),STO(7)
STREAM
  INPUT IS REIN
  OUTPUT IS REOU
EQUATION
KEYCOMP=REIN(KEY)*CONVER
REOU=REIN+STO*KEYCOMP
****
MODEL MIXER
ARRAY MIXIN1(7),MIXIN2(7),MIXOUT(7)
STREAM
  INPUT 1 IS MIXIN1
```

```
  INPUT 2 IS MIXIN2
  OUTPUT  IS MIXOUT
EQUATION
MIXOUT=MIXIN1+MIXIN2
****
MODEL SIMPLEDIST
ARRAY DIN(7),TOP(7),BOT(7),SEP(7)
STREAM
  INPUT IS DIN
  OUTPUT 1 IS TOP
  OUTPUT 2 IS BOT
EQUATION
DIN=TOP+BOT
TOP=SEP*DIN
****
UNIT REAC3 IS A REACTOR
SET KEY=4
****
UNIT REAC10 IS A REACTOR
SET KEY=3
****
UNIT MIX4 IS A MIXER
****
UNIT DIST6 IS A SIMPLEDIST
****
UNIT DIST7 IS A SIMPLEDIST
****
UNIT DIST8 IS A SIMPLEDIST
****
FLOWSHEET
INPUT OF REAC3 IS FEED 1
OUTPUT OF REAC3 IS INPUT 2 OF MIX4
INPUT 1 OF MIX4 IS OUTPUT OF REAC10
OUTPUT OF MIX4 IS INPUT OF DIST6
OUTPUT 1 OF DIST6 IS PRODUCT 1
OUTPUT 2 OF DIST6 IS INPUT OF DIST7
OUTPUT 1 OF DIST7 IS PRODUCT 2
OUTPUT 2 OF DIST7 IS INPUT OF DIST8
OUTPUT 1 OF DIST8 IS INPUT OF REAC10
OUTPUT 2 OF DIST8 IS PRODUCT 3
****
OPERATION
SET
WITHIN REAC3
  REIN(1)=0,REIN(2)=0,REIN(3)=80,REIN(4)=120,REIN(5)=0,REIN(6)=0,REIN(7)=0,
  STO(1)=0.667,STO(2)=0.667,STO(3)=-0.5,STO(4)=-1,STO(5)=0.167,STO(6)=0.833,
  STO(7)=0.333,CONVER=0.9
WITHIN REAC10
  STO(1)=0.5,STO(2)=0.5,STO(3)=-1,STO(4)=0,STO(5)=0,STO(6)=0.75,STO(7)=0,
  CONVER=0.8
WITHIN DIST6
  SEP(1)=1,SEP(2)=1,SEP(3)=0,SEP(4)=0,SEP(5)=0,SEP(6)=0,SEP(7)=0
WITHIN DIST7
  SEP(1)=1,SEP(2)=1,SEP(3)=0.05,SEP(4)=0,SEP(5)=0,SEP(6)=0.99,SEP(7)=0
WITHIN DIST8
  SEP(1)=1,SEP(2)=1,SEP(3)=0.99,SEP(4)=0,SEP(5)=0,SEP(6)=1,SEP(7)=0
```