1984

Modeling of Laboratory Scale Batch Distillations with a Desktop Computer

Dale F. Johnson
University of Central Florida

STARS Citation

Johnson, Dale F., "Modeling of Laboratory Scale Batch Distillations with a Desktop Computer" (1984). *Retrospective Theses and Dissertations*. 4672.
https://stars.library.ucf.edu/rtd/4672

MODELING OF LABORATORY SCALE BATCH DISTILLATIONS WITH A DESKTOP COMPUTER

By

DALE F. JOHNSON
B.S., North Carolina State University, 1980

RESEARCH REPORT

Submitted in partial fulfillment of the requirements
for the Master of Science degree in Industrial Chemistry in the
Graduate Studies Program of the College of Arts and Sciences
University of Central Florida
Orlando, Florida

Fall Term
1984

ABSTRACT

A program has been written that simulates the operation of a batch distillation process. The program has been implemented on a Hewlett-Packard 9845T minicomputer and will be used as an instructional aid in a graduate level chemistry course. The program is very user interactive and is capable of using the final conditions from one run as the initial values of the next simulation. The batch simulation is achieved by using a tray-by-tray algorithm and is equipped with three integration techniques; the Euler, the Modified Euler and the 4th order Runge-Kutta methods. The program has the capability for studying the effect of all distillation operational parameters on the efficiency and selectivity in the separation of mixtures containing up to five components.

ACKNOWLEDGEMENTS

I would like to thank Drs. Chris Clausen and George Hertel for their expert assistance and guidance throughout the course of this research project. I would also like to thank my wife, Margarita, whose encouragement and support has been invaluable to me during my graduate studies. Finally, I want to express my sincere gratitude to Mrs. Claire McClure for her work on the final manuscript of this research report.

TABLE OF CONTENTS

LIST OF TABLES v

LIST OF FIGURES vi

INTRODUCTION 1

I. THE PROGRAM 8

A. Introduction 8

B. Numerical Integration 11

C. Enthalpy Subroutines 21

D. Equilibrium Subroutine 24

E. Stage and Still Pot Subroutine 30

F. Condenser and Splitter Subroutines 41

G. Output Printing Subroutine 43

H. Input Section and Main Calling Program 46

I. Tests on Program Parameters and Program Limitations 53

II. TEST SIMULATIONS AND RESULTS 60

A. Comparison to Experimental Distillation 60

B. Sensitivity to Feed Composition 61

C. Distillate Product Composition Versus Reflux Ratio . 62

APPENDIX A - Listing of Program. 67

REFERENCES 75

LIST OF TABLES

1. Data for numerical integration using step sizes of τ, $\frac{3\tau}{2}$, and 2τ 15

2. Chlorobenzene mole fractions for various stage holdups 55

3. Overhead composition for different feed mixtures of chlorobenzene and ethylbenzene (compositions are in mole fractions) 61

4. Data for distillation graphs of Figure XXII 64

LIST OF FIGURES

I. General stage, n 3

II. Batch distillation column 8

III. DATA array key 9

IV. STRM array key 10

V. Graphs of data in Table 1 15

VI. (a) INTI subroutine, and (b) INT subroutine 18

VII. ENTHL subroutine 22

VIII. ENTHV subroutine 22

IX. ACTY subroutine 27

X. EQUIL subroutine 28

XI. General stage, n 30

XII. STAGE subroutine 31

XIII. Still pot 37

XIV. BOT subroutine 38

XV. TCON subroutine 41

XVI. SPLIT subroutine 43

XVII. PRL subroutine 44

XVIII. Stream numbering for a 5 theoretical plate column . . 45

XIX. DATA SECTION of main program 47

XX. Initiation section of main program 48

XXI. Main calling section of the program 52

XXII. Graphs of overhead chlorobenzene mole fraction for reflux ratios, RR, of (a) 1, (b) 5, and (c) 10 . 63

INTRODUCTION

A computer program for the simulation of a batch distillation column has been written in BASIC language for use on a Hewlett-Packard 9845T desktop computer. The program will be used for instructional purposes in a separation course in industrial chemistry where relatively short columns of an average 10 to 20 theoretical stages are used to separate mixtures of two or three components. The program is highly user interactive and allows the final conditions from one simulation, such as operation at total reflux, to be the initial conditions for the next run. The tray-by-tray calculations developed by Franks[1] for continuous distillation were adapted for the multistage batch distillation program.

Prior work in the simulation of batch distillation is not as common as for continuous distillation, but there have been some efforts[2-7] that demonstrate the interest in the problem.

Distefano[2] wrote a FORTRAN program that used a tray-by-tray approach to solve the heat and mass balance equations. He also tested several numerical integration techniques and decided to use the third order Adams-Moulton-Shell predictor-corrector method. In such a method there are two equations. First the predictor equation is applied to obtain an estimate of the value, X_2, from an initial value, X_1. The corrector equation is then used to obtain an improved value for X_2. If desired, the corrector equation can be used again in order to attempt to improve the calculated result.

Domenech and Enjalbert[3] wrote their FORTRAN program for use on a SEMS-MITRA computer with 64 kilobytes of memory. In their algorithm they first assumed that the components' heats of vaporization and heat capacities were all equal, allowing the heat balance equations to be neglected. They avoided performing the rigorous tray-by-tray calculations by further assuming that the components' relative volatilities were constant throughout the column and that there was an empirical relationship between the column's temperature profile and the number of stages.

Rooney[4] wrote a BASIC program for use on a Hewlett-Packard 85 personal computer that considers the mass and heat transfer aspects of a single stage batch still.

Guy[5] wrote a program that performs a tray-by-tray calculation with the heat and mass balance equations and also contains a mathematical model for the tray hydraulics.

In 1979 Boston[6] developed a new solution technique for the simulation of continuous distillation which he called the "inside-out algorithm". In 1980 he and Britt[7] adapted his method to multistage batch distillations. Because the technique used is an extension of that applied to continuous distillation simulations, it will be discussed later in this section.

The work on continuous distillation simulation is more numerous and more varied than for batch distillation. Some workers such as Franks[1] and Luyben[8] have used the tray-by-tray approach while others-

Wang and Henke,[9] Tomich,[10] Herchen and Hartmann,[11] Timár and Édes,[12] Boston[6] and Russell[13] have used an iterative matrix method. In the latter method the column's component balance equations are represented by a single matrix which allows the equations to be solved simultaneously. Typically, heat and total mass balances are used to check the accuracy of the solution and to predict new values for the iteration variables if necessary.

There are variations of the iterative-matrix method, but here the work of Tomich[10] will be discussed first, followed by the modifications made by Boston[6] and employed by Russell.[13]

Figure I shows a stage with a feed stream, F_n, and liquid and vapor sidedraws, LS_n and VS_n. Internal liquid and vapor streams are designated by L_n, L_{n+1}, V_n and V_{n-1}. Heat transfer rate, $\dot{Q}$, may either be added to or removed from the stage. The mass balance equation

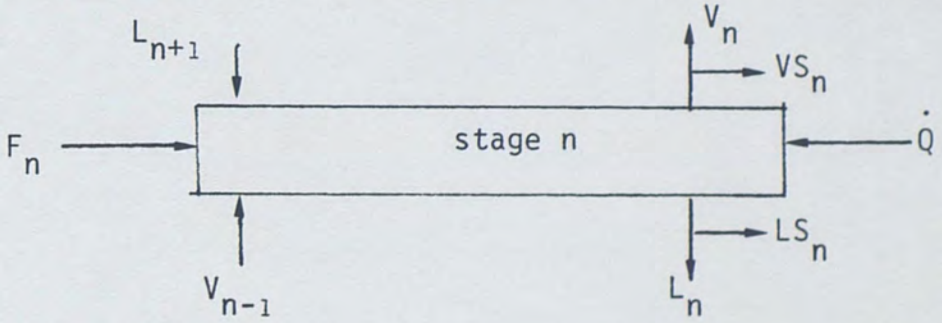


Figure I. General stage, n.

for component, i, on stage, n, will be given by:

$$X_{i,F}F + X_{i,n+1}L_{n+1} + Y_{i,n-1}V_{n-1} = X_{i,n}(L_n + LS_n) + Y_{i,n}(V_n + VS_n) \qquad (1)$$

The distribution coefficient for component, i, allows the vapor phase composition to be written in terms of the liquid phase composition:

$$Y_{i,n} = K_{i,n}X_{i,n} \tag{2}$$

The mole fraction, $X_{i,n}$, can be rewritten in terms of the component flow rate, $\ell_{i,n}$, and the total flow rate, L_n:

$$X_{i,n} = \frac{\ell_{i,n}}{L_n} \tag{3}$$

By using equations (2) and (3), equation (1) can be written in terms of component liquid flow rates:

$$f_i + \ell_{i,n+1} + \ell_{i,n-1} \cdot \left[\frac{K_{i,n-1} \cdot V_{n-1}}{L_{n-1}} \right] = \ell_{i,n} \left[(1 + \frac{LS_n}{L_n}) + \frac{K_{i,n} \cdot V_n}{L_n} (1 + \frac{VS_n}{V_n}) \right]$$

Or:

$$f_i = -\ell_{i,n+1} + \ell_{i,n} \left[(1 + \frac{LS_n}{L_n}) + \frac{K_{i,n} \cdot V_n}{L_n} (1 + \frac{VS_n}{V_n}) \right] - \ell_{i,n-1} \cdot \frac{K_{i,n-1} \cdot V_{n-1}}{L_{n-1}} \tag{4}$$

In matrix form for the component, i:

$$\begin{bmatrix} A_n & B_n & & & & & \\ -1 & A_{n-1} & B_{n-1} & & & & \\ & -1 & A_{n-2} & B_{n-2} & & & \\ & & \cdot & & & & \\ & & & \cdot & & & \\ & & & & \cdot & & \\ & & & & -1 & A_2 & B_2 \\ & & & & & -1 & A_1 \end{bmatrix} \cdot \begin{bmatrix} \ell_n \\ \ell_{n-1} \\ \ell_{n-2} \\ \cdot \\ \cdot \\ \cdot \\ \ell_2 \\ \ell_1 \end{bmatrix} = \begin{bmatrix} f_n \\ f_{n-1} \\ f_{n-2} \\ \cdot \\ \cdot \\ \cdot \\ f_2 \\ f_1 \end{bmatrix} \tag{5}$$

where

$$A_k = (1 + \frac{LS_k}{L_k}) + \frac{K_k \cdot V_k}{L_k} (1 + \frac{VS_k}{V_k}) \text{ and } B_k = -\frac{K_{k-1} \cdot V_{k-1}}{L_{k-1}}$$

Tomich used temperature and total vapor flow rates as iteration variables. With assumed values for the total vapor flow rates, the mass balance at each stage will give the total liquid flow rates for each stage. With an assumed temperature, the distribution coefficients can be calculated from:

$$K_i = \frac{Y_i}{X_i} = \frac{P_i^{*} \cdot \gamma_i}{P_{total}} \tag{6}$$

where P_i^{*} is the vapor pressure of the component and γ_i is the liquid phase activity coefficient.

With the values for the total vapor and liquid flow rates and the distribution coefficients, the tridiagonal matrix equation (5) can be solved for the interstage component liquid flow rates. Equation (3) will give the liquid phase mole fractions and then Equation (2) will give the vapor phase mole fractions. Component vapor flow rates on each stage are obtained by multiplying the vapor phase mole fraction by the initially assumed total vapor flow rate.

The total heat and mass balance relationships can be checked to determine if sufficient accuracy has been achieved. If it has not, new values for the iteration variables must be found and the process begun again.

The new iteration values are obtained from a Newton convergence method adapted to matrix techniques. Errors are obtained from the energy imbalance on each stage and the mass imbalance, i.e., the sum of the liquid phase mole fractions subtracted from one. The energy error will be used to predict the new total vapor flow rate values and the mass balance error will predict the new temperature.

Approximations to the partial derivatives of the errors with respect to the iteration variables are obtained by perturbing each variable, one at a time, and observing the change induced in the errors. These partial derivatives constitute the elements of the Jacobian matrix for the system. The inverse of the Jacobian times the error vector will give the changes to be made to the iteration variable.

For future iterations the inverse of the initial Jacobian matrix is updated by the use of Broyden's matrix update formula.[14] Broyden's formula approximates changes to the inverse of the Jacobian from the changes made to the variables and how such changes affected the errors.

Boston[6] named his new method the "inside-out algorithm" because it performs the convergence for the temperature and component liquid flow rates rapidly in an inner loop and then verifies the solution in an outer loop by comparing the relative volatilities derived from the result to those initially assumed. Rather than using the normal iteration variables of temperature and vapor flow rates, the two are combined to form a single variable called the stripping factor. The use of one variable causes the column to converge faster to a final solution. The Jacobian matrix is calculated as before in Tomich's method, and its inverse is updated with Broyden's formula as necessary.

Boston and Britt[7] were able to adapt the "inside-out" method to the unsteady state system of batch distillation because they could replace the time derivatives of the component mole fractions and stage enthalpies with linear equations. They used the backward difference formula of Gear[15] with a variable step size:

$$X_{k+1} = \beta_o \cdot \left(\frac{dX}{dt}\right)_{k+1} + \sum_{j=1}^{q} \beta_j \cdot X_{k+1-j} \qquad (7)$$

where q is the order of the method. The value of β_o will contain the step size for the increment as well as a factor that depends on the order and the previous historical data points. The values for the coefficients β_j are also obtained from the previous historical data and the method's order.

Equation (7) can be rewritten to isolate the derivative $\left(\frac{dX}{dt}\right)$, and obtain:

$$\left(\frac{dX}{dt}\right) = aX + b \qquad (8)$$

where $a = \frac{1}{\beta_o}$ and

$$b = - \sum_{j=1}^{q} \frac{\beta_j}{\beta_o} \cdot X_{k+1-j}$$

It is equation (8) that is used to replace the time derivatives in the heat and mass balance equations. Now the system of nonlinear differential equations has been reduced to a system of nonlinear algebraic equations which allows the "inside-out" technique used for continuous distillation modeling to be applied to the batch distillation.

I. THE PROGRAM

A. INTRODUCTION

The program is written using a series of subroutines and one main calling program. This organization provides a logical way for describing the program design. The two major portions of the program center around the subroutines that model the equilibrium stages. First, the mathematical model for the stage must be devised, and second, the solution of the model must be obtained by a numerical integration technique.

Figure II gives a schematic of the multistage batch distillation column modeled in this program. The physical stages are numbered starting from the bottom and proceeding upward. The still pot is considered to be stage number zero.

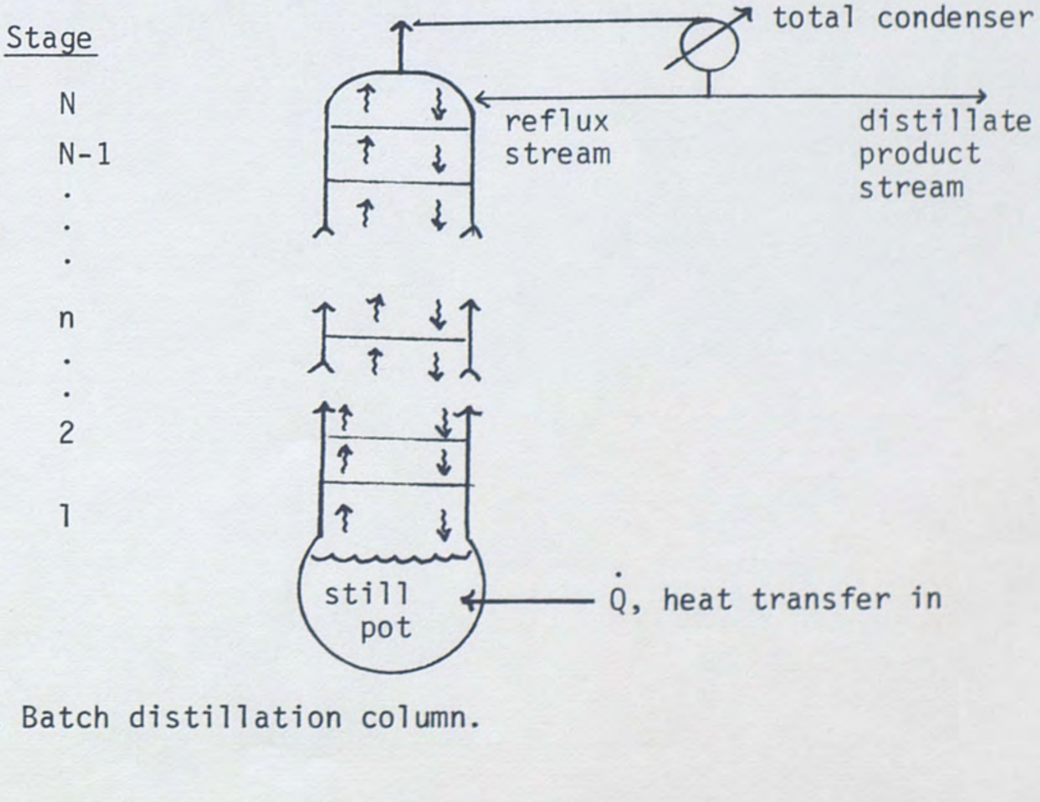


Figure II. Batch distillation column.

After having obtained the pertinent data for the components and information about the column from the user, the program will determine all the initial conditions it needs for the first set of calculations. Starting with the still pot, vapor and liquid phase compositions and flow rates are computed for each stage. After the top stage the overhead vapor stream is totally condensed and split in accordance with the reflux ratio. At this point the "clock" for the program is incremented and the tray-by-tray calculations are repeated starting at the still pot. The program run is finished when the time reaches the user-defined stopping time.

Throughout the program there are two main arrays that are used. The DATA array contains all the physical property data for the components. The array is specified as DATA (20,14) allowing for 20 components and 14 pieces of data. Figure III illustrates locations.

Location	Physical Property Data
1	C_1: Antoine Coefficient
2	C_2: Antoine Coefficient
3	C_3: Antoine Coefficient
4	A_v: Vapor Enthalpy Coefficient
5	B_v: Vapor Enthalpy Coefficient
6	ΔH_{vap} : Latent Heat of Vaporization at 0^oC
7	A_ℓ: Liquid Enthalpy Coefficient
8	B_ℓ: Liquid Enthalpy Coefficient
9	γ: Activity Coefficient
10	A_{ij}: Terminal Activity Coefficient

Figure III. DATA array key.

The second important array in the program is the STRM array which is specified as STRM (300,24), allowing for 300 streams of flowing vapor or liquid as well as the nonflowing stream in the still pot. For each stream 24 physical properties can be defined with the first 20 being the mole fractions of the components. The remaining four correspond to the flow rate in moles per minute (or simply moles for the still pot), temperature, enthalpy and finally pressure. Figure IV illustrates physical property locations of the STRM array.

Location	Physical Property
1-20	Mole fractions
21	Flow rate ($\frac{\text{moles}}{\text{min}}$), or Moles
22	Temperature
23	Enthalpy
24	Pressure

Figure IV. STRM array key.

In each multistage distillation column as shown in Figure I there will be a vapor stream and a liquid stream leaving each of the N physical stages. There also exists the condensate stream exiting the total condenser and the reflux and distillate product streams. Finally there are the still pot contents (nonflowing stream) and the vapor stream leaving the still pot. All totalled, for a column with N physical stages there will be 2N + 5 streams.

An entire listing of the computer program can be found in Appendix A. The following subsections provide discussions on each of the subroutines and sections of the program as well as a discussion of the limitations of the program.

B. NUMERICAL INTEGRATION

This program is capable of using what can be considered the three most common numerical integration methods; the Euler, the Modified Euler, and the 4th order Runge-Kutta methods. All three of these are single step methods, meaning that only the current value for the dependent variable, X_n, is used in finding the new value, X_{n+1}.

The Euler Method computes the new value by the equation:

$$X_{n+1} = X_n + \Delta t \cdot f(t_n) \tag{9}$$

where $f(t_n) = \frac{dX}{dt}$ evaluated at t_n.

Because only one derivative is computed, the Euler Method is called a first order integration method.

The Modified Euler Method determines the new value by using an average value for the derivative:

$$X_{n+1} = X_n + \Delta t \cdot \frac{f(t_n) + f(t_{n+1})}{2} \tag{10}$$

In order to obtain the derivative at the new time, $f(t_{n+1})$, the Euler Method is used to step from t_n to t_{n+1} and compute an estimate for X_{n+1}. Because there are two derivatives computed, the Modified Euler Method is a second order integration method.

The Fourth Order Runge-Kutta Method gets its name because there are four derivatives computed when calculating for the new value X_{n+1}:

$$X_{n+1} = X_n + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)\cdot\Delta t \text{ where}$$

$$K_1 = f(t_n, X_n)$$

$$K_2 = f(t_n + \frac{1}{2}\Delta t,\ X_n + \frac{1}{2}\Delta t\cdot K_1)$$

$$K_3 = f(t_n + \frac{1}{2}\Delta t,\ X_n + \frac{1}{2}\Delta t\cdot K_2)$$

$$K_4 = f(t_n + \Delta t,\ X_n + \Delta t\cdot K_3)$$

Gear[15] demonstrates how each of the methods are related to a Taylor Series expansion:

$$X_{n+1} = X_n + \Delta t\cdot\frac{dx}{dt} + \frac{1}{2}(\Delta t)^2\cdot\frac{d^2x}{dt^2} + \frac{1}{6}(\Delta t)^3\frac{d^3x}{dt^3} + \ldots + \frac{1}{n!}(\Delta t)^n\frac{dx^n}{dt^n} \qquad (12)$$

The Euler Method consists of the first two terms of the expansion; the Modified Euler (when applied to one differential equation) can be derived from the first three terms; and the Runge-Kutta Method can be taken from the first five terms. Because each method is only a portion of the full Taylor series, there exists a truncation error inherent in the formulas. For each stepwise computation, the local truncation error can be approximated by the first term following the truncation. Thus for the Euler Method the local error resulting from truncation can be approximated by the term $\frac{1}{2}(\Delta t)^2\ \frac{d^2x}{dt^2}$. For the Runge-Kutta Method, this error would be $\frac{1}{5}(\Delta t)^5\ \frac{d^5x}{dt^5}$.

When computations are taken from some starting value, X_o, to an ultimate value, X_n, there have been n steps of Δt size. The net error resulting from the formula's truncation can be approximated by n times the local error in moving one step. This assumption neglects any effects an error in one step has upon future steps but results in the overall error being (for the Euler Method):

$$\text{Error} = n \cdot \frac{1}{2}(\Delta t)^2 \cdot \frac{d^2x}{dt^2} = \frac{X_n - X_o}{\Delta t} \cdot \frac{1}{2}(\Delta t)^2 \quad \frac{d^2x}{dt^2}$$

$$\text{or Formula Error} = (X_n - X_o) \cdot \frac{1}{2} \cdot \Delta t \cdot \frac{d^2x}{dt^2}$$

Although this result is not exactly correct, it shows that the overall formula error is proportional to the integration step size raised to the power of the integration order of the method. A maximum bound on the formula error can be obtained by considering the maximum value for the second derivative, $\frac{d^2x}{dt^2}$, that occurs over the interval $[X_o, X_n]$, and then assuming this value as max $\left|\frac{d^2x}{dt^2}\right|$ for each computation step.

Also of concern in the discussion of errors are those that result from the fact that only a finite number of significant figures can be retained in each computation. The resulting error is referred to as a round-off error and depends upon the precision of the computer.

The round-off error also depends on the total number of computations performed and hence is inversely proportional to integration step size. In order to minimize the truncation error, a smaller integration step size can be chosen. However, the smaller step size will increase the round-off error. Thus, there will be some optimum integration step size that achieves the greatest accuracy for a given precision.

Also of importance when numerically integrating differential equations are the problems of stability and stiffness. To demonstrate the concept of stability, the solution for the first order equation (13) will be used.

$$\frac{dy}{dt} = \frac{1}{\tau}(A-y) \; ; \; y(o) = 0 \tag{13}$$

The analytical solution to the differential equation is

$$y = A(1-e^{-t/\tau}) \tag{14}$$

The time constant, τ, will dictate the stability behavior of the integration. This concept is illustrated using the Euler Method where an integration step size of 2τ results in critical stability, meaning the calculated values will oscillate about the true solution with a constant magnitude. For values less than 2τ, the integration is stable and will converge to the true solution. Figure V; a, b, and c show graphs of the analytical solution, equation (14), compared to graphs using integration step sizes of τ, $\frac{3\tau}{2}$ and 2τ. Table 1; a, b, and c provides the data points plotted for each of the three cases.

Table 1

Data for numerical integration using step sizes of τ, $\frac{3\tau}{2}$, and 2τ.

Case a: $dt=\tau$

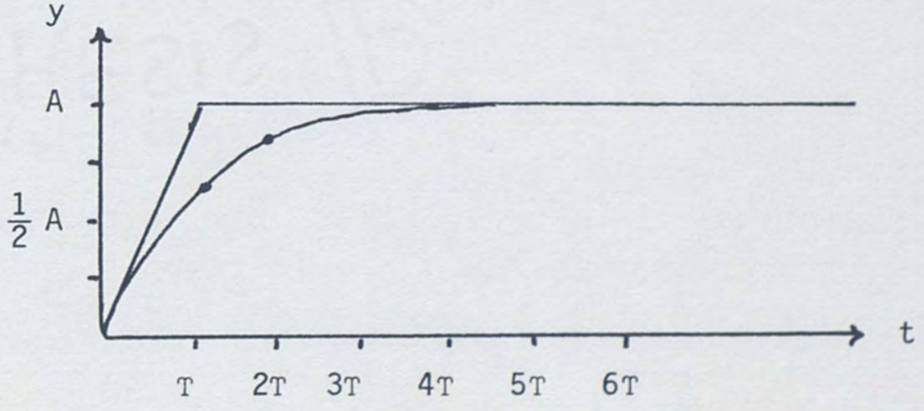
t	y	$\frac{dy}{dt}$
0	0	$\frac{A}{\tau}$
τ	A	0
2τ	A	0
3τ	A	0
4τ	A	0
5τ	A	0

Case b: $dt=\frac{3\tau}{2}$

t	y	$\frac{dy}{dt}$
0	0	$\frac{A}{\tau}$
$\frac{3\tau}{2}$	$\frac{3}{2}A$	$-\frac{1}{2}\frac{A}{\tau}$
3τ	$\frac{3}{4}A$	$\frac{1}{4}\frac{A}{\tau}$
$\frac{9\tau}{2}$	$\frac{9}{8}A$	$-\frac{1}{8}\frac{A}{\tau}$
6τ	$\frac{15}{16}A$	$\frac{1}{16}\frac{A}{\tau}$

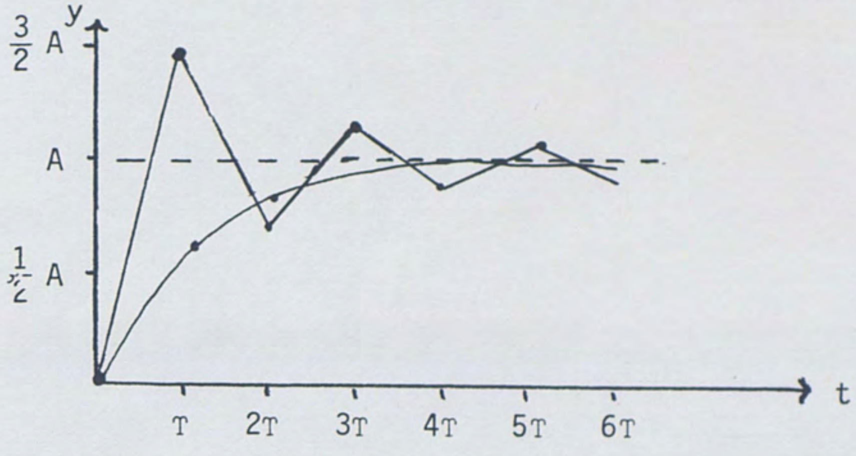
Case c: $dt=2\tau$

t	y	$\frac{dy}{dt}$
0	0	$\frac{A}{\tau}$
2τ	$2A$	$-\frac{A}{\tau}$
4τ	0	$\frac{A}{\tau}$
6τ	$2A$	$-\frac{A}{\tau}$

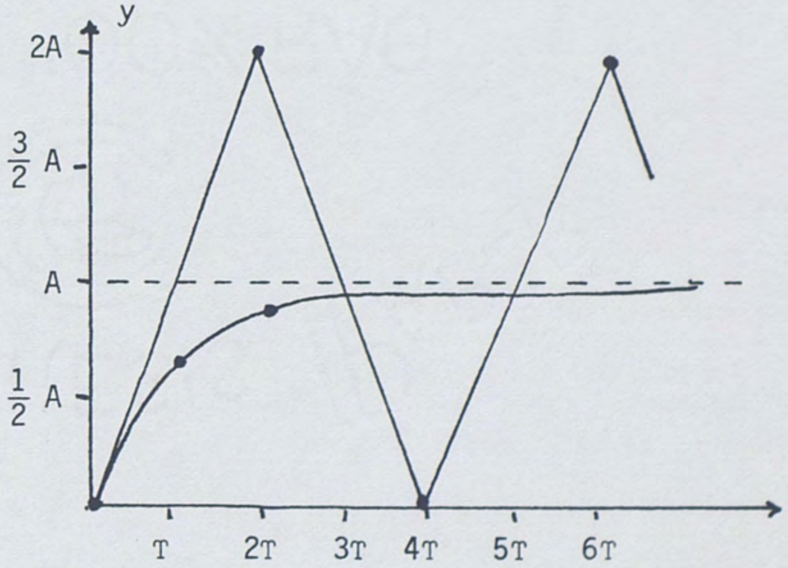


Case a: $dt = \tau$

Figure V. Graphs of data in Table 1 (continued on following page).



Case b: dt= $\frac{3}{2}$ T



Case c: dt= 2t

Figure V (continued). Graphs of data in Table 1.

When systems of differential equations are to be integrated, the various equations will have their own individual time constants and hence could use different step sizes. In the case of the simulation of a batch distillation column, the time constants of the equations describing the conditions on the trays will be much smaller than those for the still pot. As a result, the integration step size must be chosen to be small in order for the calculations for the stages to be stable. Because the step size is so small, conditions on the stages will change at a rapid pace in comparison to those of the still pot. It is this characteristic of the simulation that gives rise to the term stiffness.

Figure VI (a and b) gives the two subroutines INTI and INT. INTI establishes the values for the independent variable to be used in the integration and INT is the subroutine where the actual integration is performed.

The INTI subroutine has two primary purposes. First, it updates the independent variable (time) and thereby acts as the clock for the program. Second, the subroutine controls the indices JN, JS and JS4. JN is used to transfer values for the dependent variable and its derivatives into internal arrays of the INT subroutine. JS and JS4 are indices that will properly direct each call on the INT routine. In some other subroutines JS and JS4 are used to indicate when a full integration step has been completed. This will prevent operations from being performed with the intermediate results generated during the Modified Euler and Runge-Kutta integration schemes.

Common to both subroutines is the integration order, IO, which can have values of 1, 2 or 4 corresponding respectively to the Euler, Modified Euler and Runge-Kutta methods. The program has an assumed integration order of 1 but can be changed by the user as part of input.

```
(a)  SUB INTI (TD,DTD,IOD)
     COM . . . .
     IO=IOD
     JN=0
     ON IO GOTO 1,2,3,3
1    JS=2
     GOTO 5
2    JS=JS+1
     IF JS=3 THEN JS=1
     IF JS=2 THEN SUBEXIT
5    DT=DTD
8    TD=TD+DTD
     T=TD
     SUBEXIT
3    JS4=JS4+1
     IF JS4=5 THEN JS4=1
     IF JS4=1 THEN GOTO 6
     IF JS4=3 THEN GOTO 7
     SUBEXIT
6    DT+DTD/2
     GOTO 8
7    TD=TD+DT
     DT=2*DT
     T=TD
     SUBEND
(b)  SUB INT (X,DX)
     COM . . . .
     JN=JN+1
     ON IO GOTO 1,2,3,3
1    X=X+DX*DT
     SUBEXIT
2    ON JS GOTO 4,5
4    DXA(JN)=DX
     X=X+DX*DT
     SUBEXIT
5    X=X+(DX-DXA(JN))*DT/2
     SUBEXIT
3    ON JS4 GOTO 6,7,8,9
6    XA(JN)=X
     DXA(JN)=DX
     X=X+DX*DT
     SUBEXIT
7    DXA(JN)=DXA(JN)+2*DX
     X=XA(JN)+DX*DT
     SUBEXIT
8    DXA(JN)=DXA(JN)+2*DX
     X=XA(JN)+DX*DT
     SUBEXIT
9    DXA(JN)=(DXA(JN)+DX)/6
     X=XA(JN)+DXA(JN)*DT
     SUBEND
```

(For the variables declared in the COMMON statements for all subroutines, see the program listing in Appendix A)

Figure VI. (a) INTI subroutine, and (b) INT subroutine.

The INT routine contains the dummy variables, TD, DTD and IOD because the actual variables T, DT, and IO cannot appear in both the declaration statement of the subroutine and the COMMON statement of the program. INTI is always called before the INT routine. If there are N differential equations to be integrated, INTI will be called once (in order to increment the time and to update the indices JS and JS4) for every N calls made on the INT routine. Every time the INTI routine is called the index JN is set equal to zero, but with each call of INT, JN is increased by 1. Thus for N equations, JN will start with a value of zero but will run from 1 to N. When the INTI routine is called again in order to increment the time, JN will be reinitialized to zero.

The operations of the two subroutines will now be discussed for all three of the integration methods. Considering first the Euler Method (IO=1), each time INTI is called JS is set equal to 2 and the time is incremented by adding the step size, DT, to the current value of time, T. When INT is called with IO=1, the new value for the dependent variable, X, is calculated by multiplying its derivative by the integration step size and adding the result to the current value that was transferred to the subroutine in the CALL statement.

When the Modified Euler Method is to be used, IO will equal 2. At the beginning of an integration step, INTI will first assign a value of 1 to JS and then increment the time as done for the Euler method. With a second call, JS will become 2 which indicates the integration procedure is in the second and final step. Time is not to be changed and thus after incrementing JS, INTI is exited. In the INT subroutine, IO=2 directs the program to line 2 where the value of JS dictates the next steps. With JS=1, the current value of the derivative is stored in the derivative array, DXA. Next a simple Euler calculation gives the intermediate value for the variable X. This value is retained in the XA array, and when the next sequence of calls comes, JS=2 and the final result for the dependent variable is computed on line 5. The equation programmed on line 5 results from eliminating X_o from the two equations:

$$\begin{aligned} X_i &= X_o + DX_o \cdot DT \qquad \text{and} \\ X_f &= X_o + DT \cdot \frac{DX_o + DX_f}{2} \quad . \end{aligned} \tag{15}$$

The first equation computes the intermediate estimate, X_i, for the variable X while the second equation is the modified Euler formula computing the final value, X_f, from an average of the derivatives at the beginning and end of the interval.

If the integration order is 4, the routines will be directed to perform the 4th order Runge-Kutta technique. In the INTI subroutine the index JS4 will cycle through the values of 1 through 4 in each complete integration step. Initially, with JS4=1, the integration step size provided by the user will be divided in half and time incremented by the result. When JS4=2, the subroutine is exited. With JS4=3, time is incremented again and the step size is restored to its original value. When JS4=4, the subroutine is exited. In short, the integration step is $\frac{1}{2}$ of that specified by the user for JS4=1 and JS4=2, but has been restored to its full value for JS4=3 and JS4=4.

With JS4=1 the INT routine stores the values of the dependent variable and its derivative in the arrays XA and DXA. Next a Euler calculation is made with the derivative to get an intermediate value for X. With JS4=2, twice the value of the derivative is added to the initial value of the derivative (which was stored in the DX array). Also with this derivative a new value for the dependent variable is calculated. With JS4=3, twice the value of the derivative is added to what is stored in the DX array. With this derivative a computation for the estimate at the end of the integration step is made (DT has been restored to full size at this point). When JS4=4 the final weighted average for the derivative is computed on line 9 and used to compute the ultimate result for the dependent variable.

C. ENTHALPY SUBROUTINES

The enthalpies of the liquid and vapor streams are needed in the heat balance equations to determine the stage vapor flow rates. The subroutines that calculate the enthalpies are called ENTHL and ENTHV and their listings are given in Figures VII and VIII.

```
SUB ENTHL (I)
COM . . . .
HV=0
FOR N=NCF TO NCL
HV=HV+STRM(I,N)*(DATA(N,7)+DATA(N,8)*STRM(I,22))
NEXT N
STRM(I,23)=HV*STRM(I,22)
SUBEND
```

Figure VII. ENTHL subroutine.

```
SUB ENTHV (I)
COM . . . .
HV=0
FOR N=NCF TO NCL
HV=HV+STRM(I,N)*((DATA(N,4)+DATA(N,5)*STRM(I,22))*
     STRM(I,22)+DATA(N,6))
NEXT N
STRM(I,23)=HV
SUBEND
```

Figure VIII. ENTHV subroutine.

In both subroutines the heat capacity has been assumed to be a linear function of temperature:

$$C_p = a + bT \qquad (16)$$

The change in enthalpy between two temperatures is simply the integral of equation (16). If the lower limit temperature is taken as 0^oC, the enthalpy change will be obtained as follows:

$$\int_{H_o}^{H} dH = \int_{0^oC}^{T} CpdT = \int_{0^oC}^{T} (a+bT)dT$$

$$\Delta H = aT + \frac{1}{2} bT^2 \qquad (17)$$

For vapor phases, the latent heat of vaporization at 0^oC (the reference temperature from the integration) must also be included. The equation on a per mole basis for the liquids and vapors respectively are:

$$H_\ell = (A_L + B_L T)T \qquad (18)$$

$$\text{and } H_v = (A_v + B_v T)T+C \qquad (19)$$

where A is the heat capacity at 0^oC,

B is $\frac{1}{2}$ x the temperature coefficient of heat capacity, and

C is the latent heat of vaporization at 0^oC.

Because there is more than one component in each stream, subroutines ENTHL and ENTHV use a loop to add up all the contributions of each component. Thus, the program's calculations come from the following summation equations:

$$H_\ell = T* \sum_{i=1}^{n} X_i (A_{L,i}+B_{L,i}T) \qquad (20)$$

$$H_v = \sum_{i=1}^{n} X_i [(A_{v,i}+B_{v,i}T)T + C_i] \qquad (21)$$

D. EQUILIBRIUM SUBROUTINE

The program treats each stage as an equilibrium stage meaning that the vapor and liquid phases leaving each tray are assumed to be in thermodynamic equilibrium. The equilibrium conditions must therefore be predicted by an appropriate mathematical model.

Distribution coefficients are defined as the ratio of the composition of a component in the vapor phase to its composition in the liquid phase:

$$K_i = \frac{Y_i}{X_i} \tag{22}$$

Values for distribution coefficients for ideal gas and liquid mixtures can be obtained by Raoult's Law which states that the partial pressure of a gas in equilibrium with a liquid phase would equal the mole fraction of the component in the liquid phase times the vapor pressure of the component at that temperature:

$$P_i = X_i P^*_{i,T} \tag{23}$$

Dalton's Law of Partial Pressure states that the partial pressure of a gas in a mixture is equal to the mole fraction of the component in the vapor phase times the total pressure of the mixture:

$$P_i = Y_i\ P_{total} \tag{24}$$

Thus, combining Raoult's and Dalton's laws, the distribution coefficient can be expressed as:

$$K_i = \frac{y_i}{x_i} = \frac{P^*_{i,T}}{P_{total}} \quad \text{or} \quad Y_i = \frac{X_i P^*_{i,T}}{P_{total}} \tag{25}$$

This resulting expression holds true under the assumption of an ideal liquid solution and ideal gas mixture-conditions which are very rare.

Smith[16] states that if the total pressure is less than 30 psia, the vapor phase can be assumed to form an ideal gas mixture.

Because nonideal liquid solutions are often encountered, the equation (25) must be modified to accommodate non-ideality in the liquid phase. At low pressures the introduction of the activity of the component in the liquid phase, rather than its mole fraction (i.e. concentration) is sufficient. The activity of the component, a_i, will simply equal its mole fraction, X_i, times the activity coefficient, γ_i:

$$a_i = \gamma_i X_i \tag{26}$$

Raoult's Law then becomes:

$$Y_i = \frac{\gamma_i X_i P^*_{i,T}}{P\ \mathrm{total}} \tag{27}$$

The values for activity coefficients depend upon the components in the mixture, the temperature and the pressure. As discussed by Smith[16] the equation above can be rewritten as:

$$\gamma_i = \frac{Y_i\ P_{\mathrm{total}}}{X_i P^*_{i,T}} \tag{28}$$

Equation (28) shows that the activity coefficient indicates the agreement with, or departure from, Raoult's Law. The numerator of the expression gives the actual partial pressure of the component in the vapor phase while the denominator gives the partial pressure predicted (at some temperature) by Raoult's Law. If the actual pressure is greater than that predicted by Raoult's Law, the activity coefficient will be greater than one. The logarithm of the activity coefficient will be positive giving rise to the name "positive deviation from

Raoult's Law". On the other hand, if the actual partial pressure is less than that from Raoult's Law, γ will be less than one, and its logarithm negative, resulting in a "negative deviation from Raoult's Law".

The most common correlations for activity coefficients occur for binary mixtures and are given by the Margules and Van Laar equations. Treybal[17] has used Wohl's method to show that the Margules and Van Laar equations can be derived from an empirical relation between the excess molar free energy of a nonideal solution and interaction terms for clusters of two, three, four or more molecules. If the empirical equation is set up with clusters of two molecules assumed, the resulting equation for the activity coefficient will be second order, i.e. terms of X_1^2, X_2^2 or X_1X_2. Such a second order correlation is also called a two suffix relation. Three molecules in the interaction clusters will result in a third order relation, or a three-suffix equation. The two-suffix Van Laar equations were chosen for use in this program:

$$\log \gamma_1 = \frac{A_{12}\, X_2^2}{\left(X_2 + \frac{A_{12}}{A_{21}} X_1\right)^2} \qquad (29)$$

$$\log \gamma_2 = \frac{A_{21}\, X_1^2}{\left(X_1 + \frac{A_{21}}{A_{12}} X_2\right)^2}$$

where A_{12} and A_{21} are respectively the logarithms of the activity coefficients for component 1 at infinite dilution in component 2 and component 2 at infinite dilution in component 1.

Figure IX shows the ACTY subroutine that programs the binary Van Laar equations.

```
SUB ACTY (N)
COM . . . .
A12=DATA(1,10)
A21=DATA(2,10)
X1=STRM(N,1)
X2=STRM(N,2)
DATA(1,9)=10^(A12*X2^2/(X1*A12/A21+X2)^2)
DATA(2,9)=10^(A21*X1^2/(X2*A21/A12+X1)^2)
SUBEND
```

Figure IX. ACTY subroutine.

The values for A12 and A21 are obtained from the DATA array and these values were initially supplied by the user. Next the mole fractions are obtained from the STRM array for the particular stream, N, indicated in the subroutine declaration statement. In the last two steps, the activity coefficients are calculated and stored in location 9 of the DATA array for components 1 and 2. It is important to note that the two nonideal components are assigned the first two component numbers.

The subroutine EQUIL listed in Figure X will take the liquid phase compositions and compute the vapor phase compositions in equilibrium with that liquid stream. The calculation converges to final correct vapor compositions by performing a Newton-Raphson convergence with temperature as the iteration variable.

```
      SUB EQUIL (IL,IV)
      COM . . . .
3     SM=Ø
      SDY=Ø
      IF LACT=1 THEN GOTO 1
      CALL ACTY(I,L)
1     FOR N=NCF TO NCL
      STRM(IV,N)=10^(DATA(N,1)-DATA(N,2)/(STRM(IL,22)+DATA(N,3)))*
           STRM(IL,N)/STRM(IL,24)*DATA(N,9)
      SM=SM+STRM(IV,N)
      DY=STRM(IV,N)*2.3025851*DATA(N,2)/(STRM(IL,22)+DATA(N,3))^2
      SDY=SDY+DY
      NEXT N
      YER=1-SM
      STRM(IL,22)=STRM(IL,22)+YER/SDY
      IF ABS(YER)<.0001 THEN GOTO 2
      GOTO 3
2     STRM(IV,22)=STRM(IL,22)
      SUBEND
```

Figure X. EQUIL subroutine

The LACT variable will have values of either 1 or 2 as designated by the user during the initial input phase of the program. A value of 1 indicates that all components are ideal and in such a case the call statement for the ACTY subroutine is bypassed. If LACT=2, the activity coefficients for the first two components will be calculated. In the input section of the program all the activity coefficients are initialized with values of one; hence no user action is necessary for ideal solutions except to enter that LACT is 1.

Starting with an initial value for temperature a loop is used to compute the vapor phase compositions of each component in the stream. The vapor mole fractions are calculated from equation (27):

$$Y_i = \frac{\gamma_i X_i \; P^*_{i,T}}{P_{total}}$$

where $P^*_{i,T}$, the component vapor pressure, is given by Antoine's equation:

$$\log (P^*_{i,T}) = C_1 - \frac{C_2}{T + C_3} \tag{30}$$

where the vapor pressures are in units of mmHg and the temperatures are in degrees Centigrade. The derivative of the vapor composition with respect to temperature is needed for the Newton-Raphson convergence technique. The temperature dependence of the activity coefficient is neglected.

$$\frac{dY_i}{dT} = \frac{\gamma_i X_i}{P_{total}} \cdot \frac{d[10^{C_1 - \frac{C_2}{T+C_3}}]}{dT}$$

$$\frac{dY_i}{dT} = \frac{\gamma_i X_i}{P_{total}} \cdot 10^{C_1 - \frac{C_2}{T+C_3}} \cdot \ln(10) \cdot \frac{C_2}{(T+C_3)^2}$$

$$\frac{dY_i}{dT} = Y_i \cdot \ln(10) \cdot \frac{C_2}{(T+C_3)^2} \tag{31}$$

Thus for all N components of the mixture, the vapor compositions and their derivatives with respect to temperature are computed and accumulated respectively in the variables SM and SDY. The error in the vapor composition is obtained and if it is not below a tolerance of 0.0001, another iteration is performed with the new equilibrium temperature determined by the following equation:

$$T_{new} = T_{old} + \frac{YER}{SDY} \tag{32}$$

As the iterations approach the final values for vapor compositions, SDY will attain a constant value, YER will approach zero and the difference between the old and new temperatures will go to zero. The value of 0.0001 as the tolerance for YER assures a nominal 99.99% accuracy and should be adequate for most all cases.

E. STAGE AND STILL POT SUBROUTINE

An equilibrium stage, n, is shown in Figure XI with the liquid and vapor streams leaving the stage represented by IL and IV respectively. Streams I1 and I2 represent the vapor stream from the stage below and the liquid stream from the stage above.

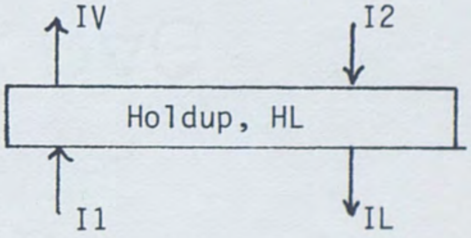


Figure XI. General stage, n.

The main purpose of the STAGE subroutine in Figure XII is to calculate the vapor and liquid flow rates leaving the tray and the liquid phase composition. Vapor phase compositions will be determined when the subroutine EQUIL is called on the next pass through the program.

```
      SUB STAGE (I1,I2,IL,IV,H,HL,HTC)
      COM . . . .
      CALL EQUIL (IL,IV)
      CALL ENTHL (IL)
      CALL ENTHV (IV)
      FLIN=STRM (I1,21)+STRM(I2,21)+RCT(21)
      HIN=STRM(I1,23)*STRM(I1,21)+STRM(I2,21)*STRM(I2,23)+H+RCT(22)
      STRM (IV,21)=(HIN-FLIN*STRM(I1,23))/(STRM(IV,23)-STRM(IL,23))
      DL=(FLIN-STRM(IV,21)-STRM(I1,21))/HTC
      FOR N=NCF TO NCL
      HK=STRM(IV,N)/STRM(IL,N)
      CNIN=STRM(I1,21)*STRM(I1,N)+STRM(I2,21)*STRM(I2,N) +RCT(N)
      IF HK>5 THEN GOTO 1
      DERN=(CNIN-FLIN*STRM(IL,N)-STRM(IV,21)*(STRM IV,N)-STRM(IL,N)))/HL
2     CALL INT(STRM(IL,N), DERN)
      NEXT N
      CALL INT (STRM(IL,21),DL)
      SUBEXIT
1     STRM(IV,N)=CNIN/(STRM(IV,21)+STRM(IL,21)/HK)
      STRM(IL,N)=STRM(IV,N)/HK
      DERN=0
      GOTO 2
      SUBEND
```

Figure XII. STAGE subroutine.

The heat balance equation for the stage, equations (33) and (34), is used to calculate the vapor flow rate leaving the stage.

$$\frac{d(HL \cdot H_{\ell})}{dt} = \text{Rate of Heat In - Rate of Heat Out} \qquad (33)$$

$$HL_n \frac{dH_{\ell,n}}{dt} + H_{\ell,n} \cdot \frac{dHL_n}{dt} = V_{n-1} \cdot H_{v,n-1} \cdot H_{\ell,n+1} + RCT(22) - V_n \cdot L_n \cdot H_{\ell,n} + H \qquad (34)$$

where H is the external heat added or removed on the stage and RCT(22) accounts for the heat generated or absorbed by any chemical reactions.

The first term on the left side of equation (34) is the product between the stage molar holdup and the change in sensible heat for the stage. This term will be neglected, an assumption that Franks[1] explains is valid so long as there is a substantial vapor flow rate throughout the column and the temperature gradient from top to bottom is not large. If the temperature gradient is large (and especially if it changes abruptly over a few stages) and the liquid enthalpies are of comparable magnitude to the vapor enthalpies, the change in sensible heat cannot be neglected. In such a case the amount of heat needed to change the temperature of the stage and its liquid stream is comparable to the heat content in the vapor stream and will significantly affect the vapor flow rate.

With the change in sensible heat neglected, no heat added or removed and no reaction heat, equation (34) becomes:

$$H_{\ell,n} \cdot \frac{dHL_n}{dt} = V_{n-1} \cdot H_{v,n-1} + L_{n+1} \cdot H_{\ell,n+1} - V_n \cdot H_{v,n} - L_n \cdot H_{\ell,n} \qquad (35)$$

The derivative of the holdup is given by:

$$\frac{dHL_n}{dt} = V_{n-1} + L_{n+1} - V_n - L_n \tag{36}$$

The term "heat in", HIN, is given by:

$$HIN = V_{n-1} \cdot H_{v,n-1} + L_{n+1} \cdot H_{\ell,n+1} \tag{37}$$

and the term "flow in", FLIN, is given by:

$$FLIN = V_{n-1} + L_{n+1} \tag{38}$$

Substituting equations (36)-(38) into (35) and solving for V_n gives

$$V_n = \frac{HIN - FLIN*H_{\ell,n}}{H_{v,n} - H_{\ell,n}} \tag{39}$$

After computing the vapor flow rate, the subroutine computes the rate of change of the liquid flow rate. The time delay that occurs for the liquid flowing onto the stage and that flowing off it is modeled using a hydraulic time constant, HTC. Equation (40) is the derivative equation and equation (41) is the form that is programmed.

$$\tau_{HTC} \cdot \frac{dL_n}{dt} = V_{n-1} + L_{n+1} - V_n - L_n \tag{40}$$

$$\frac{dL_n}{dt} = \frac{FLIN - V_n - L_n}{HTC} \tag{41}$$

The value for the time constant depends upon the dimensions of the plate and the value used by Franks[1] is 0.1 minute. Bolles[18] performs the elaborate tray hydraulics calculations and has determined a typical residence time to be near 4 seconds. Equation (40) could be rewritten in the following way if a constant vapor boilup rate is assumed:

$$\frac{dL_n}{dt} = \frac{L_{n+1} - L_n}{\tau_{HTC}} \tag{40A}$$

If the derivative is replaced by an average rate of change, $\frac{\Delta L_n}{\Delta t}$, and the stream L_{n+1} flows onto the stage at some time, t_1, and stream L_n flows off it at a later time, t_2, the time constant can be set equal to the change in time, Δt. This time difference can be approximated by the residence time on the stage.

Next the component balance is applied to find the rate of change of component composition. The component balance equation is

$$\frac{d(HL \cdot X_i)_n}{dt} = V_{n-1} \cdot Y_{i,n-1} + L_{n+1} \cdot X_{i,n+1} + RCT(X_i) - L_n X_{i,n} - V_n Y_{i,n} \tag{42}$$

The "component in" term is given by:

$$CNIN = V_{n-1} \cdot Y_{i,n-1} + L_{n+1} \cdot X_{i,n+1} + RCT(X_i) \tag{43}$$

Using equations (36)-(38) and (43), equation 42 can be solved for $\frac{dX_i}{dt}$:

$$\frac{dX_{i,n}}{dt} = \frac{CNIN - FLIN \cdot X_{i,n} - V_n (Y_{i,n} - X_{i,n})}{HL_n} \tag{44}$$

This is the form of the mass balance equation programmed in the subroutine STAGE. By using the distribution coefficient, $Y_{i,n}$ can be eliminated from equation (42) and with the rate of change of the holdup neglected the result is:

$$\frac{dX_{i,n}}{dt} = \frac{V_{n-1} \cdot Y_{i,n-1} + L_{n+1} \cdot X_{i,n+1} - V_n \cdot K_{i,n} \cdot X_{i,n} - L_n \cdot X_{i,n}}{HL_n} \tag{45}$$

which can be written in the form:

$$\frac{dX_{i,n}}{dt} = \frac{1}{\tau}\,(F_i - X_i)$$

$$\text{where } \tau = \frac{HL_n}{K_{i,n} \cdot V_n + L_n} \text{ and} \tag{46}$$

$$F_i = \frac{(V_{n-1} \cdot Y_{i,n-1} + L_{n+1} \cdot X_{i,n+1})HL_n}{K_{i,n} \cdot V_n + L_n}$$

As discussed earlier, the time constant τ will dictate the stability and hence integration step size permissible when solving the equation. Using the value for τ in equation (46) was determined to be accurate by Distefano[2] who showed that the critical step size would lie between $\frac{HL}{K \cdot V_n + L_n}$ and $2 * \frac{HL}{K \cdot V_n + L_n}$.

On each stage the values for the holdup and the two flow rates will be the same for all components, however the distribution coefficients will not be the same. For high boiling components, the value will be close to zero, and for more volatile components it may become quite large. When the distribution coefficient is nearly zero the time constant will be approximately $\frac{HL_n}{L_n}$ and does not cause serious problems in the integration. On the other hand, when the distribution coefficient becomes large, the time constant will decrease and result in instability as τ approaches the value specified for the integration step size.

Rather than having to select an extremely small time constant, it can be realized that equation (46) can be expressed as

$$\tau \cdot \frac{dX_{i,n}}{dt} = F_i - X_{i,n} \tag{47}$$

and with τ being very small, the term $\tau \cdot \frac{dX_{i,n}}{dt}$ will be small in comparison to $X_{i,n}$ and F_i. Thus, the differential equation can be approximated by a steady state equation and the mole fractions determined algebraically.

$$\tau \cdot \frac{dX_{i,n}}{dt} \cong 0 = F_i - X_{i,n}$$

$$0 = V_{n-1} \cdot Y_{i,n-1} + L_{n+1} \cdot X_{i,n+1} - Y_{i,n} \cdot V_n - X_{i,n} \cdot L_n$$

With $K_{i,n} = \dfrac{Y_{i,n}}{X_{i,n}}$:

$$0 = V_{n-1} \cdot Y_{i,n-1} + L_{n+1} \cdot X_{i,n+1} - V_n \cdot Y_{i,n} - L_n \cdot \frac{Y_{i,n}}{K_{i,n}}$$

Solving for $Y_{i,n}$:

$$Y_{i,n} = \frac{V_{n-1} \cdot Y_{i,n-1} + L_{n+1} \cdot X_{i,n+1}}{V_n + \dfrac{L_n}{K_{i,n}}} \qquad (48)$$

With $Y_{i,n}$ evaluated, $X_{i,n}$ can be determined from the distribution coefficient.

The value of τ at which the steady state approximation can be made is an arbitrary one. τ can be calculated by selecting a limiting value for the time constant and using estimated values for the tray holdups and stream flow rates. The subroutine STAGE uses the steady state calculations for the compositions if the distribution coefficients exceed 5.

If this bypass feature is used the INT subroutine is still called so that the updating of the index, JN, is not disrupted and derivatives for one component are not used when integrating for the composition of another.

The STAGE subroutine is applicable to all the physical stages of the column except for the still pot. Figure XIII gives a sketch of the still pot showing the vapor stream leaving, IV, the liquid stream entering from the first stage, LI, and the heat transfer rate in, $\dot{Q}$. The still pot stream is represented by LO.

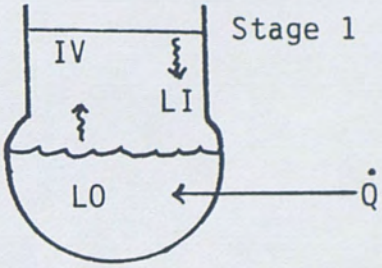


Figure XIII. Still pot.

For the modeling of the still pot the subroutine BOT is used and its listing is given in Figure XIV.

```
     SUB BOT (LI,LO,IV,Q)
     COM . . . .
     IF (JS4=4) OR (JS=2) OR (LSTR=1) THEN (STRM(LO,20)=STRM(LO,23))

     CALL EQUIL(LO,IV)
     CALL ENTHL(LO)
     CALL ENTHV(IV)
     DENL=(STRM(LO,23)-STRM(LO,20))*STRM(LO,21)/DT
     QP=Q+STRM(LI,21)*STRM(LI,23)+RCT(22)
     DHL=STRM(LI,21)-STRM(IV,21)+RCT(21)
     DEN=STRM(IV,23)-STRM(LO,23)
     STRM(IV,21)=(QP-STRM(LO,23)*(STRM(LI,21)+RCT(21))-DENL)/DEN
     IF STRM(IV,21)<0 THEN STRM(IV,21)=0
     FOR N=NCF TO NCL
     HK=STRM(IV,N)/STRM(LO,N)
     IF HK>20 THEN GOTO 1
     FNI=STRM(LI,21)*STRM(LI,N)+RCT(N)
     FNO=STRM(IV,21)*STRM(IV,N)
     DERN=(FNI-FNO-STRM(LO,N)*DHL)/STRM(LO,21)
2    CALL INT(STRM(LO,N),DERN)
     NEXT N
     CALL INT (STRM(LO,21),DHL)
     SUBEXIT
1    STRM(IV,N)=(STRM(LI,21)*STRM(LI,N)+RCT(N))/STRM(IV,21)

     STRM(LO,N)=STRM(IV,N)/HK
     DERN=0
     GOTO 2
     SUBEND
```

Figure XIV. BOT subroutine.

At first glance, the BOT subroutine appears similar to STAGE, but this is not so because there is no liquid stream leaving the pot; the pot holdup is large compared to the tray holdups, and there is heat transferred into the pot.

The heat balance relation is again used for determining the vapor flow rate leaving the still pot. Because the holdup in the still pot is large, the term containing the change in the sensible heat cannot be neglected. The derivative is computed by dividing the integration step size into the difference between the current enthalpy and that corresponding to the beginning of the integration step. The latter value of enthalpy is stored in dummy location 20 of the STRM array for the still pot and is updated at the end of each integration step, i.e. when JS4=4 or JS=2.

The heat balance relation is

$$\frac{d(HL\cdot H_{\ell})_B}{dt} = \begin{matrix}\text{rate of}\\ \text{heat transfer in}\end{matrix} - \begin{matrix}\text{rate of}\\ \text{heat transfer out}\end{matrix}$$

Or,

$$H_{\ell,B}\cdot\frac{dHL_B}{dt} + HL_B\cdot\frac{dH_{\ell,B}}{dt} = LI\cdot H_{\ell,LI}+RCT(22)+Q-IV\cdot H_{V,IV} \qquad (49)$$

Letting,

$$Q_P=Q+RCT(22)+LI\cdot H_{\ell,LI} \qquad (50)$$

$$DHL=\frac{dHL_B}{dt} = LI+RCT(21)-IV \qquad (51)$$

$$DENL=HL_B\cdot\frac{dH_{\ell,B}}{dt} \qquad (52)$$

equation (49) can be solved for the vapor boilup rate, IV:

$$H_{\ell,B}\cdot(LI+RCT(21)-IV)+DENL=Q_P-IV\cdot H_{V,IV}$$

$$IV=\frac{Q_P-DENL-H_{\ell,B}(LI+RCT(21))}{H_{V,IV}-H_{\ell,B}} \qquad (53)$$

For the component balance equations, the derivative of the composition with respect to time is obtained from:

$$\frac{d(HL\cdot X_i)_B}{dt} = LI\cdot X_{i,LI} + RCT(N)-IV\cdot Y_{i,IV} \qquad (54)$$

Or, $$HL_B \cdot \frac{dX_{i,B}}{dt} + X_{i,B} \cdot \frac{dHL_B}{dt} = LI \cdot X_{i,LI} + RCT(N) - IV \cdot Y_{i,IV}$$

Using equation (51) and letting

$$FNI = LI \cdot X_{i,LI} + RCT(N)$$

$$FNO = IV \cdot Y_{i,IV}$$

the following result is obtained.

$$\frac{dX_{i,B}}{dT} = \frac{FNI - FNO - X_{i,B} \cdot DHL}{HL_B} \qquad (55)$$

The assumption of steady state solution to equation (55) is still based on the value of the distribution coefficient. However, because the still pot's holdup is so large in comparison to the holdup of the trays, the assumption is not used unless the distribution coefficient is greater than twenty. This value again can be obtained from the time constant of the equation. Equation (54) can be rewritten as:

$$\frac{dX_{i,B}}{dt} = \frac{LI \cdot X_{i,LI} - (LI + IV(K_{i,B} - 1))X_{i,B}}{HL_B} \qquad (56)$$

Or, with $$\tau = \frac{HL_B}{LI + IV(K_{i,B} - 1)}$$

$$\frac{dX_{i,B}}{dt} = \frac{1}{\tau}(F_i - X_{i,B}) \qquad (57)$$

$$F_i = \frac{LI \cdot X_{i,LI} \cdot (HL_B)}{LI + IV(K_{i,B} - 1)}$$

For given values of holdup and flow rate, the smallest values of the time constant will occur as the distribution coefficients become large. When comparing the time constants of equations (57) and (46), it can be seen that for all practical purposes the liquid flow rates and vapor boilup rates will not vary greatly, nor will the distribution coefficient for a given component from tray to tray. Thus, the denominators of the two time constants are essentially the same for high boiling components. With this result, the difference in the time constants can be seen to depend more on their holdups. With a ratio between the holdups of 100 being common, the distribution coefficient of 20 being used to key the steady state assumption for the still pot is conservative.

F. CONDENSER AND SPLITTER SUBROUTINES.

The overhead vapor stream enters a total condenser from which emerges a condensate stream which will have the same composition as the overhead as well as the same flow rate. The condenser is assumed to subcool the condensate by two degrees. Figure XV gives the listing for the TCON subroutine. The variables I, and JO represent the inlet overhead and outlet condensate streams, respectively.

```
SUB TCON(I,JO)
COM . . . .
FOR N=NCF TO NCL
STRM(JO,N)=STRM(I,N)
NEXT N
STRM(JO,22)=STRM(I,22)-2
STRM(JO,21)=STRM(I,21)
CALL ENTHL(JO)
SUBEND
```

Figure XV. TCON subroutine.

The condensate stream flows into a splitter which determines the distillate product and reflux stream flow rates and physical properties based on the reflux ratio, RR, supplied by the user. The subroutine will establish the temperatures, enthalpies, compositions and flow rates for the two exiting streams.

The subroutine will simulate operation at total reflux if the user supplies a value for the reflux ratio greater than or equal to 1000. In such operation the program assigns values of zero to all the properties and compositions of the distillate product stream except the temperatures, which it sets equal to the condensate temperature. If the reflux ratio is zero, the program sets all the conditions of the reflux stream except temperature equal to zero.

Should the reflux ratio lie between these two extremes, the program sets the enthalpies, temperatures and compositions of the reflux and distillate streams equal to the values from the condensate stream. The flow rates are determined from the definition of the reflux ratio and the total mass balance equation:

$$RR=\frac{\text{reflux flow rate}}{\text{distillate flow rate}}=\frac{RFLX}{D} \tag{58}$$

$$C=RFLX+D \tag{59}$$

where C is the condensate flow rate. Solving for the reflux and distillate flow rates give

$$D=\frac{C}{RR+1} \text{ and } RFLX=D\cdot RR \tag{60}$$

Figure XVI lists the SPLIT subroutine.

```
      SUB SPLIT(C,RFLX,D,RR)
      COM . . . .
      IF RR>=1000 THEN GOTO 1
      STRM(D,21)=STRM(C,21)/(1+RR)
      STRM(RFLX,21)=STRM(D,21)*RR
      STRM(RFLX,22)=STRM(D,22)=STRM(C,22)
      IF RR=0 THEN GOTO 2
      STRM(RFLX,23)=STRM(C,23)
      STRM(D,23)=STRM(C,23)
      FOR N=NCF TO NCL
      STRM(RFLX,N)=STRM(C,N)
      STRM(D,N)=STRM(C,N)
      NEXT N
      SUBEXIT
1     STRM(RFLX,21)=STRM(C,21)
      STRM(D,21)=0
      STRM(RFLX,22)=STRM(C,22)
      STRM(D,22)=STRM(C,22)
      STRM(RFLX,23)=STRM(C,23)
      STRM(D,23)=0
      FOR N=NCF TO NCL
      STRM(RFLX,N)=STRM(C,N)
      STRM(D,N)=0
      NEXT N
      SUBEXIT
2     STRM(RFLX,23)=0
      STRM(D,23)=STRM(C,23)
      FOR J=1 TO NCL
      STRM(D,J)=STRM(C,J)
      STRM(RFLX,J)=0
      NEXT J
      SUBEND
```

Figure XVI. SPLIT subroutine.

G. OUTPUT PRINTING SUBROUTINE

The outputting of results is performed by the subroutine PRL listed in Figure XVII.

```
       SUB  PRL(PRI,FNR,NF)
       COM . . . .
       IF LSTR=1 THEN GOTO 1
       IF (T>=TPRNT-DT/2) AND ((JS=2) OR (JS4=4)) THEN GOTO 2

       SUBEXIT
1      LSTR=NF=0
2      TPRNT=TPRNT+PRI
       PRINTER IS 0
3      IMAGE 1X, "TIME = "; D.DDDE
       PRINT
       PRINT USING 3; T
       PRINT
       PRINT "STRM# FLOW  TEMP COMP1  COMP 2  COMP3  COMP4  COMP5"
       FOR J=1 TO NP
       PRINT USING FORMAT2; J,STRM(J,21),STRM(J,22)
       FORMAT2:  IMAGE #,XXD.DDDE
       FOR I=NCF TO NCL
       PRINT USING FORMAT3; STRM(J,I)
       FORMAT3:  IMAGE #, XX.DDDDE
       NEXT I
       PRINT
       NEXT J
       IF T>=FNR-DT/3 THEN GOTO 4
       PRINTER IS 16
       SUBEXIT
4      T=0
       TPRNT=0
       MAT XA=ZER
       PRINTER IS 16
       NF=1
       SUBEND
```

Figure XVII. PRL subroutine.

In the declaration statement the variables PRI, FNR and NF are introduced. PRI is the print interval, FNR is the stop (or finishing) time for the simulation run and NF is a parameter that indicates that the time has reached the value of FNR.

The parameter LSTR is used in calculating the boilup rate in the still pot on the first pass through the BOT subroutine. Similarly, in the PRL subroutine, LSTR is used to print the column's initial conditions. LSTR is initialized with a value of 1 in the input section of the main program, but on the first pass through the PRL subroutine, LSTR will be given a value of 0.

Whenever the time is equal to the print time, TPRNT, the conditions of the column will be printed. For each stream the flow rate (or moles for the still pot), temperature and component mole fractions will be printed.

When the time reaches the stop time of FNR all the values of time, T, print time, TPRNT, and the entries of the array XA are reset to zero and the parameter NF set equal to one. The designation "printer is 16" corresponds to the CRT of the computer used in this work and "printer is 0" corresponds to the hard copy printer.

Figure XVII shows the numbering of the streams for a column with 4 physical stages, i.e. a total of 5 theoretical plates.

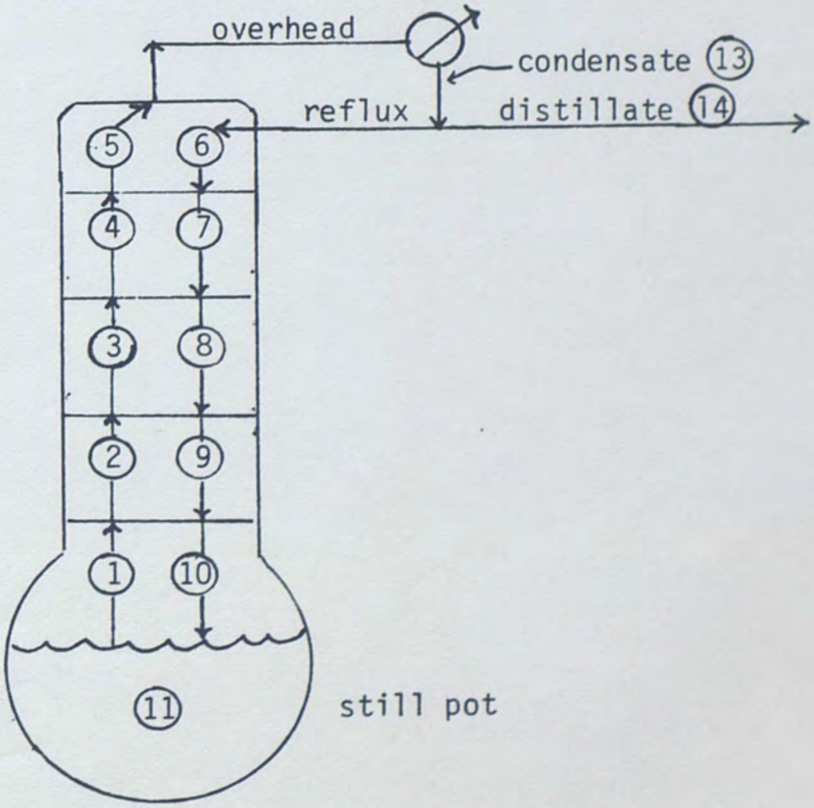


Figure XVIII. Stream numbering for a 5 theoretical plate column.

H. INPUT SECTION AND MAIN CALLING PROGRAM

In the Input section, the user will provide all the required data for the enthalpy models and Antoine and Van Laar equations as well as the characteristics of the column. Based on the given information the program will be divided into two subsections in order to facilitate their presentation.

Figure XIX gives the listing of the DATA SECTION. First, the user defines the total numbers of theoretical plates, NPLTS, for his column. The program then calculates the number of physical stages, NSTGS. The distinction between the total number of plates and stages is necessary in order to determine the number of times the STAGE subroutine must be called. The total number of streams, NP, is calculated as well as the stream number corresponding to the still pot, NPOT.

The user then tells how many components are in the feed mixture and whether they form an ideal mixture, LACT=1, or if the first two components are nonideal, LACT=2. Next the user enters the Antoine coefficients and enthalpy data for the components. The program assigns values of 1 to all the activity coefficients, but if LACT=2, the program obtains the logarithms of the terminal activity coefficients, A_{12} and A_{21}, from the user. An excellent source for the Antoine and Van Laar coefficients is Hála et al.[19]

```
     ! DATA SECTION
     DISP "(" ; NPLTS; ")";
     INPUT "What is the total number of theoretical plates (including
          the still pot)?", NPLTS
     NSTGS=NPLTS-1
     NP=2*NSTGS+5
     NPOT=2*NSTGS+3
     NCF=1
     DISP "(" ; NCL; ")";
     INPUT "What is the total number of components (maximum of 5)?",
          NCL
     DISP "("; LACT; ")";
     INPUT "Enter 1 if mixture ideal, 2 if 2 comps nonideal and rest
          are ideal", LACT
     PRINTER IS 16
     PRINT PAGE
     PRINT "Enter data for the components according to the following
          menu.  Enter the data for the nonideal components first."
     PRINT "The Antoine equation is log(P)=C1-C2/(T+C3) where P is
          pressure in mm Hg and T is temperature in Centigrade."
     PRINT TAB (10); "ITEM 1:  Antoine coefficient, C1", LIN(1)
     PRINT TAB (10); "ITEM 2:  Antoine coefficient, C2", LIN(1)
     PRINT TAB (10); "ITEM 3:  Antoine coefficient, C3", LIN(1)
     PRINT TAB (10); "ITEM 4:  Vapor enthalpy coefficient, Av", LIN(1)
     PRINT TAB (10); "ITEM 5:  Vapor enthalpy coefficient, Bv", LIN(1)
     PRINT TAB (10); "ITEM 6:  Latent heat of vaporization at 0
          Centigrade", LIN(1)
     PRINT TAB (10); "ITEM 7:  Liquid enthalpy coefficient, Aℓ", LIN(1)
     PRINT TAB (10); "ITEM 8:  Liquid enthalpy coefficient, Bℓ", LIN(1)
     FOR N=NCF TO NCL
     FOR I=1 TO 8
     DISP "Component number"; N; "Item"; I; ":"; DATA(N,I);
     INPUT DATA (N,I)
     NEXT I
     NEXT N
     FOR J=NCF TO NCL
     DATA (J,9)=1
     NEXT J
     PRINT PAGE
     IF LACT=1 THEN GOTO 2
     PRINT "The Van Laar equation is used in calculating the activity
          coefficients."
     DISP "("; DATA (1,10); ")";
     INPUT "What is the binary activity coefficient of comp 1 out of
          comp 2?", DATA (1,10)
     DISP "("; DATA (2,10); ")";
     INPUT "What is the binary activity coefficient of comp 2 out of
          comp 1?", DATA (2,10)
2    ! INITIATION SECTION
```

Figure XIX. DATA SECTION of main program.

Figure XX gives the listing of the INITIATION SECTION of the program. In this portion of the program, the user supplies more information about the column and the program determines the initial flow rates, compositions, temperatures and pressures for each stage.

```
2     ! INITIATION SECTION
      DISP "("; STRM(NPOT,21); ")";
      INPUT "What is the initial number of moles in the still pot?",
           STRM(NPOT,21)
      DISP "("; STRM(NPOT,22); ")";
      INPUT "What is the initial temperature of the still pot?",
           STRM(NPOT,22)
      FOR N=NSTGS+2 TO NPOT-1
      STRM(N,22)=STRM(NPOT,22)
      NEXT N
      PRINT "What is the composition of the feed mixture, enter mole
           fractions and press CONT key after each entry."
      FOR N=NCF TO NCL
      DISP "Component number"; N; ":"; STRM(NPOT,N);
      INPUT STRM(NPOT,N)
      FOR J=NSTGS+2 TO NPOT+2
      STRM(J,N)=STRM(NPOT,N)
      NEXT J
      NEXT N
      PRINT PAGE
      DISP "("; DELTAP; ")";
      INPUT "What is the total pressure drop for the column; (in
           mm Hg)?", DELTAP
      PPOT=760+DELTAP
      STRM(NPOT,24)=STRM(1,24)=PPOT
      FOR J=2 TO NSTGS+1
      JD=2*(NSTGS+2)-J
      STRM(J,24)=PPOT-DELTAP/NSTGS*(J-1)
      STRM(JD,24)=STRM(J,24)
      CALL EQUIL (JD,J)
      CALL ENTHL (JD)
      NEXT J
      CALL ENTHL (NSTGS+2)

      STRM(NSTGS+2,24)=STRM(NPOT+1,24)=STRM(NPOT+2,24)=1
      TOTHL=NSTGS*.03
      DISP "(";TOTHL;")";
      INPUT "What is the total molar holdup of the column?", TOTHL
      HL=TOTHL/NSTGS
```

Figure XX. Initiation section of main program (continued).

```
      DISP "("; Q; ")";
      INPUT "What is the heat input to the pot in calories per
           minute?", Q
      DEN=0
      FOR N=NCF TO NCL
      DEN=DEN+STRM(NPOT,N)*DATA(N,6)
      NEXT N
      STRM(1,21)=Q/DEN
      IF RR<3 THEN GOTO 3
      FOR N=NSTGS+2 TO NPOT-1
      STRM(N,21)=STRM(1,21)
      NEXT N
      IF RR>3 THEN GOTO 4
3     FOR N=NSTGS+2 TO NPOT-1
      STRM(N,21)=.75*STRM(1,21)
      NEXT N
4     IO=1
      DT=.02
      PRINT "What is the integration order?  Value of 1 is assumed
           if none given."
      DISP "("; IO; ")";
      INPUT "1 is Simple Euler, 2 is Modified Euler, 4 is Runge-
           Kutta.", IO
      PRINT PAGE
      DISP "("; DT; ")";
      INPUT "What is the integration step size?  .02 minutes assumed
           if none given", DT
      DISP "("; FNR; ")";
      INPUT "At what time do you want the program to stop?", FNR
      DISP "("; PRI;")";
      INPUT "What is the print interval?", PRI
      HTC=.05
      DISP "("; HTC; ")";
      INPUT "What is the hydraulic time constant?  .05 assumed if
           none given", HTC
      TPRNT=0
      LSTR=1
      T=0
      CALL EQUIL (NPOT,1)
      CALL ENTHL (NPOT)
      INPUT "Do you want to change any inputs?  YES=1, NO=0", AGAIN
      IF AGAIN=1 THEN GOTO 1
      INPUT "Do you want the program to return for another run?
           YES=1, NO=0", RTRN
```

Figure XX. Initiation section of Main Program (continued from preceding page).

The initial estimate for the liquid flow rates are based on the initial boilup rate and the reflux ratio. If the reflux ratio is less than three, the flow rates for the liquid streams are estimated to be 75% of the estimated boilup rate. If the reflux ratio is greater than three, the initial flowrates for the liquid streams are taken to be the same as the boilup rate.

The user then defines the integration order, but the program has a default value of 1 if none is assigned. The integration step size is then obtained and a default value of 0.02 minutes is used if none is given. The user must supply the time for the program to stop and the printing interval. The final bit of information needed to run the program is a value for the hydraulic time constant which will be assumed to be 0.05 minutes if no value is supplied.

The program then initializes the clock with T=0, the printing clock, TPRNT=0, and the logical flag LSTR=1.

Finally, the program is able to return to the top of the data section in order to inspect and change the data prior to commencing the run. The very last statement asks whether the program is to return to the very beginning once the current run is completed or if the entire program is to stop. The advantage of returning to the beginning is that the data and information from the previous run has been retained. Thus, one possibility is to run the column at total reflux until the steady state is attained, and then begin a new simulation run with the total reflux results as initial conditions.

Following the two subsections of the INPUT section is the main calling portion of the program. The first part of the INPUT section is the DERIVATIVE section and its second part is the INTEGRATION section. Figure XXI gives the listing of the main calling segment of the program.

```
      ! DERIVATIVE SECTION
5     CALL BOT(NPOT-1,NPOT,1,Q)
      FOR N=1 TO NSTGS
      CALL STAGE (N,NPOT-1-N,NPOT-N,N+1,0,HL,HTC)
      NEXT N
      CALL TCON(NSTGS+1,NPOT+2)
      CALL SPLIT(NPOT+2,NSTGS+2,NPOT+1,RR)
      CALL PRL(PRI,FNR,NF)
      IF (NF=1) AND (RTRN=1) THEN PRINT "THIS RUN DONE!  READY TO
           START ANOTHER!"
      IF (NF=1) AND (RTRN=1) THEN GOTO 1
      IF NF=1 THEN GOTO 6
      ! INTEGRATION SECTION
      CALL INTI (T,DT,IO)
      GOTO 5
      PRINT "PROGRAM IS FINISHED!!  WHAT'S NEXT?"
      END
```

Figure XXI. Main calling section of the program.

The program starts calculations on each pass by first calling the BOT subroutine and then works up the column through the N stages. After TCON and SPLIT have been called, the printing subroutine PRL is called. If the current time equals the TPRNT variable, output will be printed. Also, if the current time equals the finish time, FNR, the parameter NF will be set to 1. If NF is not 1, the INTI subroutine is called and the independent variable time is incremented for the next pass when the program is directed back to the call statement of the BOT subroutine. However, if NF were 1, then after returning from the PRL subroutine one of two things will happen based on the value of the parameter RTRN. As discussed earlier, a value of 1 for RTRN means that the user wants the program to retain all information used and return to the beginning of the DATA section to start a new run. This is precisely what happens when both NF and RTRN have values of 1. If the user does not want the program to return to the beginning, then he should enter a value of 0 for RTRN during the initial input section. With only NF=1 in this case the program simply ends.

I. TESTS ON PROGRAM PARAMETERS AND PROGRAM LIMITATIONS

The limitations of the program were studied by performing a series of tests in which one parameter was allowed to vary while all the others were held constant at values equal to those used for a base standard. The parameters tested were the heat transfer rate, $\dot{Q}$, the molar holdup per stage, HL, the hydraulic time constant, HTC, the pressure drop across the column, the initial flow rate estimates and the extent of subcooling that occurs in the condenser. The standard for comparison was a 9 theoretical stage column charged with a 40/60 mole percent mixture of ethylbenzene/chlorobenzene and operating at total reflux. The heat transfer rate was 2200 calories per minute; the stage holdup was .02 moles; the hydraulic time constant was .03 minutes; the pressure drop was 60 mm Hg; the initial flow rates were .05 moles per minute for all streams; and the extent of subcooling was 1 degree Centigrade.

The heat transfer rate, $\dot{Q}$, establishes the vapor boilup rates throughout the column. As $\dot{Q}$ is increased these boilup rates increase. For this case when the flow rates were approximately .35 moles per minute (resulting from approximately 3000 calories per minute for $\dot{Q}$), the numerical integration gave oscillating, inaccurate results.

The time constant for the component balance equation of component i on stage n is given by equation (46).

$$T = \frac{HL_n}{K_{i,n} \cdot V_n + L_n} \tag{46}$$

For a column operating at total reflux, once steady state is achieved the liquid and vapor flow rates will be equal. For this test the distribution coefficients were between 1.1 and 1.2 and HL_n was .02 moles. Thus, a typical time constant for these equations can be calculated as:

$$T = \frac{.02}{(1.15)(.35)+(.35)} = .027 \text{ min}$$

This value is slightly greater than the .02 min step size used in the integration. The earlier statements that the critical step size for an equation should lie between one and two times the value of the time constant is applicable to a single equation. If the time constant is too close to the integration step size, the computed value will oscillate. Because all the equations are coupled, the oscillating values in one equation will create oscillating values in the next. The entire computation may or may not converge if allowed to run for a long enough time, but regardless of that, the results being generated along the way will be inaccurate. Thus, $\dot{Q}$ values that result in flow rates that create time constants within 50% of the specified integration step size should be avoided.

When studying the sensitivity of the simulation to the stage molar holdup, HL, it was found that the overhead composition is significantly affected by a change in the holdup and that at low holdup values the oscillating/convergence problem arises. From equation (46) it can be seen that the time constant varies directly with the stage holdup, and hence as the holdup is decreased the time constant will approach the value of the integration step size. Thus, the oscillating problem with the calculations is expected to occur for small tray holdups.

Table 2 shows the values of the chlorobenzene mole fraction in the overhead stream for various stage holdups. The tabulated values are after 5 minutes of simulation in which it takes about 15 minutes to reach steady state.

Table 2

Chlorobenzene mole fractions for various stage holdups

Stage Molar Holdup, HL	Chlorobenzene Mole Fraction
.01	.8085
.02	.7984
.024	.7921
.04	.7681

The result that separability, and hence tray efficiency, drops as the stage holdup increases is in agreement with how distillation columns are known to operate. Because of the sensitivity of the compositions, the stage holdups need to be determined as accurately as possible when comparing computer simulated results to those of an actual column. The program uses a default value of .03 moles per stage if none is provided by the user.

The hydraulic time constant, HTC, is used in determining the liquid flows leaving a stage. Liquid residence time is an excellent estimate of the time delay between the liquid flowing onto the stage and that flowing off of it. In the test runs if the value of the time constant became too small, the oscillating behavior of the calculations appeared. However, for values that gave stable results, there was no effect on the results. The program has a default value of .05 minutes, and if .10 minutes is used the results are the same. For values less than .03 minutes the calculation begins to oscillate.

The only effect of changing the column pressure drop was to change the equilibrium temperatures on each stage. There was no discernible effect on the calculated results, and no convergence problems in the integration occurred.

The sensitivity of the simulation to the initial values of the flow rates was also not noticeable. The only effect was that the number of moles charged to the still pot would increase or decrease in the first few minutes of this simulation in order to satisfy the total mass balance relationship. For example, if the initial estimates were too low (i.e. lower than when actually calculated) the number of moles in the still pot would decrease, simulating the vaporization of liquid in order to increase the flow rates and holdups on the stages. As discussed in the INPUT section earlier, the initial estimates for the flow rates are calculated from the values for the heat transfer rate, reflux ratio, feed composition and component latent heats of vaporization.

The extent of subcooling that occurs in the condenser has no effect on compositions of the streams or on the stability of the calculations. It does however affect the boilup rate of the top stage. This result is expected because the boilup rate is calculated from the enthalpy balance equation, and a cooler reflux stream returning to the top stage will have lower enthalpies. The program assumes the condensate stream is subcooled by 2 degrees Centigrade.

There are two limitations to the program based on the models used to simulate the distillation processes in the column. The first results from the assumption that the change in the sensible heat on each stage can be neglected, and the second from the assumption of a constant molar holdup per tray combined with the simple time delay model of the tray hydraulics.

As mentioned earlier, neglecting the rate of change of the sensible heat on the stages is invalid if a large temperature gradient exists from top to bottom in the column and the liquid enthalpies are of the same magnitude as the vapor enthalpies. Large temperature gradients will exist whenever components with a large difference in boiling points and heat capacities are charged to the still pot. In such a case, the low boiling component will be more concentrated on the lower stages and force the stage temperatures to be large, while the more volatile component will be found on the upper stages and tend to keep the stage temperatures low. Test runs were made and if the temperature gradient approaches $20^{\circ}C$, the simulation does not give accurate results because the vapor boilup rates and liquid flow rates oscillate. The reason is that the change in sensible heat is large on each stage and the stage liquid enthalpies should be determined by integration rather than algebraically calculated from the ENTHL subroutine. For the use of this program charges that generate temperature gradients greater than $10\text{-}15^{\circ}C$ should be avoided.

With an assumed constant molar holdup and assuming the tray hydraulics are approximated by a hydraulic time constant model, the program is incapable of predicting what conditions will cause flooding of the column. The decreased efficiency with increasing holdup, however, does depict the poorer contacting between the vapor and liquid phases on each stage. The constant molar holdup assumption implies that the component mixture has the same density for all temperatures and compositions and that the formula weights of the components are the same. The assumption of a constant volume holdup with the density modeled as a function of temperature, pressure and composition was adopted by Distefano[2] and Boston and Britt.[7] For instructional uses the constant molar holdup assumption keeps the programming simplified and still relatively accurate for most cases. Complete models for tray hydraulics can be found in Guy[5] and Bolles[18] and if a true simulation of the tray hydraulics is desired these two works are excellent sources. Just as for the constant molar holdup assumption, the simple time constant model for the tray hydraulics keeps the program both simple and accurate for its intended use as an instructional tool.

One final limitation results from the use of the rigorous tray-by-tray approach for the simulation and the fact that it creates long run times. This program operating on the HP 9845T computer is running at about $\frac{1}{4}$ second for each call of the STAGE or BOT subroutines. This value is independent of the number of components, stages, and step size and integration order. All of these factors dictate the total number of integrations that must be performed by the following equation:

$$N_{INT} = IO*N_{STGS}*N_{COMPS}*\left(\frac{\text{TIME SIMULATED}}{DT}\right) \qquad (61)$$

II. TEST SIMULATIONS AND RESULTS

Other than the test simulations on the various parameters of the program, there were three case studies made in order to

1) verify the program's accuracy with an operating experimental column, and
2) operate the simulation in the manner it would be used in the future as an instructional tool.

A. COMPARISON TO EXPERIMENTAL DISTILLATION

The agreement with experimental results was obtained by operating a laboratory batch distillation containing 9 theoretical stages (16 actual sieve trays with efficiency of 56%) at total reflux with an initial charge of 60/40 mole percent chlorobenzene and ethylbenzene.

The computer simulation charged the same amount of feed (5.5 moles) with the same composition and applied the same heat transfer rate (2100 calories/min). It used the default values for integration order, step size and tray holdups. The simulation used a hydraulic time constant of .035 minutes.

The actual column was allowed to run for three hours at which time a 1 ml sample of the overhead distillate was analyzed by gas chromatography to give a chlorobenzene composition of 78.1 mole %. The computer simulation ran for one hour and arrived at an overhead chlorobenzene composition of 81 mole %.

In actual practice, the computer program will be used to demonstrate the effects of changing reflux ratios, applied heat transfer rate, and the number of stages on the efficacy of the separation. To verify the usefulness of the program in these regards the following case studies were performed.

B. SENSITIVITY TO FEED COMPOSITION

In this study, two columns were compared in terms of their sensitivities to feed composition. The two columns were identical except that one had 9 theoretical stages while the other had 25. To each column various chlorobenzene and ethylbenzene feed mixtures were charged. Table 3 (a) shows the feed and total reflux overhead compositions for the 9 stage column, and Table 3 (b) shows the same information for the 25 stage column.

TABLE 3

Overhead composition for different feed mixtures of chlorobenzene and ethylbenzene (compositions are in mole fractions).

(a) 9 theoretical stage column

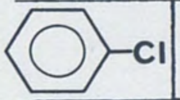
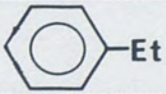
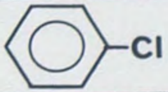
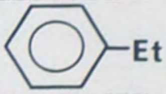
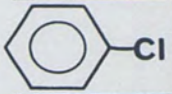
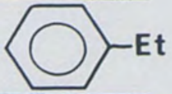
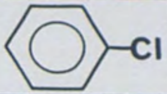
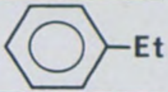
Feed Mixtures		Overhead product	
			
.60	.40	.81	.19
.50	.50	.74	.26
.40	.60	.65	.35

TABLE 3 (continued)

(b) 25 theoretical stage column

Feed Mixture	
	
.60	.40
.40	.60
.30	.70

Overhead product	
	
.965	.035
.91	.09
.87	.13

The results show that for a change from 60 mole % to 40 mole % chlorobenzene feed mixtures, the 9 stage column undergoes a much greater change in overhead chlorobenzene concentration. For the 9 stage column there is a 16 mole % drop in concentration, whereas for the 25 stage column there is only a 5.5 mole % drop. These results are summarized by noting that the separation achieved by the smaller column is more sensitive to the composition of feed mixture.

C. DISTILLATE PRODUCT COMPOSITION VERSUS REFLUX RATIO

One of the most common questions that arises when working with batch distillation is how does the distillate product vary with reflux ratio? To investigate this problem, the still pot was charged with 5.5 moles having a composition of 60/40 mole % of chlorobenzene and ethylbenzene. The program simulated a 9 stage column operating at reflux ratios of 1, 5 and 10. The output from the program is plotted in Figure XXII as the mole fraction of the initial feed distilled. Table 4 summarizes the data used

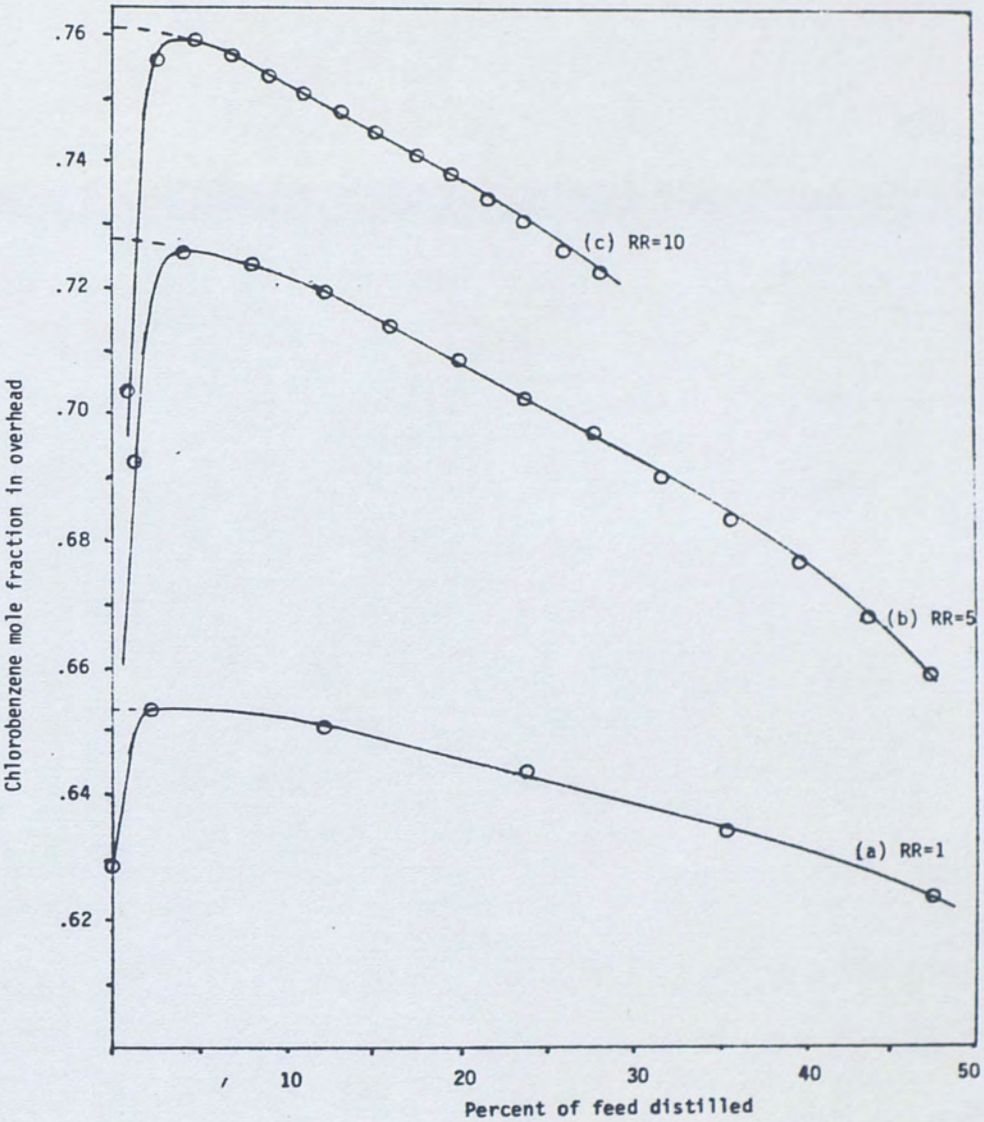


Figure XXII. Graphs of overhead chlorobenzene mole fraction for reflux ratios, RR, of (a) 1, (b) 5, and (c) 10.

Table 4

Data for distillation graphs of Figure XXII

Time	% Feed distilled	Fraction mole
(a) Reflux ratio = 1		
0	0	.6280
1	2.71	.6534
5	12.2	.6505
10	23.98	.6431
15	35.84	.6343
20	47.71	.6237
25	59.55	.6103
(b) Reflux ratio = 5		
0	0	.6280
1	1.51	.6924
5	4.62	.7255
10	8.49	.7234
15	12.38	.7187
20	16.27	.7136
25	20.16	.7083
30	24.05	.7026
35	27.96	.6966
40	31.85	.6901
45	35.76	.6832
50	39.67	.6757
55	43.56	.6676
60	47.49	.6588

Time	% Feed distilled	Fraction mole
(c) Reflux ratio = 10		
0	0	.6280
1	1.24	.7037
5	2.93	.7565
10	5.02	.7591
15	7.13	.7566
20	9.25	.7537
25	11.35	.7507
30	13.45	.7476
35	15.58	.7443
40	17.69	.7410
45	19.80	.7375
50	21.91	.7339
55	24.02	.7302
60	26.15	.7263
65	28.27	.7223

and also shows the corresponding time through the simulation.

The first drop of distillate taken is always the most concentrated for a given run, but in Figure XXII each case starts with a chlorobenzene composition of 62.8 mole %. The reason for this is that the initial feed composition is used to initialize all the stages in the column. The data can be extrapolated back to the vertical axis to give an estimate of the composition of that initial drop of distillate, and in Figure XXII this extrapolation is shown with dashed lines.

The graphical information can be used to predict what value of reflux ratio should be used to achieve a certain degree of enrichment. For example, if the experimenter wanted to collect one mole from the 5.5 moles of charge, corresponding to 18.18 % of the initial feed, a reflux ratio of 1 would give an average distillate product with a composition over 65 mole % chlorobenzene, while a reflux ratio of 10 would give a product with over 74 mole %.

For another problem, it may be desirable to know what reflux ratio to use in order to obtain a continuous distillate stream to be fed to another experiment such that the stream's composition is never less than 70 mole %. A reflux ratio of 5 will give the desired composition until 25% of the initial charge has been distilled. The reflux ratio of 10 also will give the composition above 70 mole % until nearly 40% of the feed has been removed. Since a high reflux ratio means a low distillate flow rate, the experimenter could decide to use the reflux ratio of 5 until he has collected between 20 and 25% of the initial feed and then switch to a reflux ratio of 10. In fact, the computer program can simulate this type of run by setting the finish time, FNR, to 25 to 30 minutes with a reflux ratio of 5 and RTRN set equal to 1. Once this part of the run is finished the program will return to the beginning where the user can tell the program to use all the current values for the parameters except change the reflux ratio to 10.

The inaccuracy of the graphical data in Figure XXII during the early phases of the simulation runs can be eliminated by running the program under total reflux until steady state is attained, and then again with RTRN having been set equal to 1, start a second run with the desired reflux ratio. This approach is exactly how Distefano[2] and Boston and Britt[7] designed their respective programs.

One final point can be made about the data in Figure XXII. If the reflux ratio were allowed to continue to increase the graphs would converge to a single plot. The vertical intercept would represent the column's greatest ability to enrich the feed, i.e., the distillate composition at total reflux. The column used in these runs has 9 stages and from Table 3 the overhead composition will converge to a value of 81 mole %.

APPENDIX A

Listing of Program

Listing of Program BATCH

```
10    !  DATA SECTION
20    COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt,
      Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
30    DISP "(";Nplts;")";
40    INPUT "What is the total number of theoretical plates (inclu
      ding the still pot)?",Nplts
50    Nstgs=Nplts-1
60    Np=2*Nstgs+5
70    Npot=2*Nstgs+3
80    Ncf=1
90    DISP "(";Ncl;")";
100   INPUT "What is the total number of components (maximum of 5)
      ?",Ncl
110   DISP "(";Lact;")";
120   INPUT "Enter 1 if mixture ideal, 2 if 2 comps nonideal and r
      est are ideal",Lact
130   PRINTER IS 16
140   PRINT PAGE
150   PRINT "Enter data for the components according to the follow
      ing menu. Enter the data for the nonideal components first."
160   PRINT "The Antoine equation is log(P) = C1 - C2/(T + C3) whe
      re P is pressure in mmHg and T is temperature in Centigrade."
170   PRINT TAB(10);"ITEM 1:  Antoine coefficient, C1",LIN(1)
180   PRINT TAB(10);"ITEM 2:  Antoine coefficient, C2",LIN(1)
190   PRINT TAB(10);"ITEM 3:  Antoine coefficient, C3",LIN(1)
200   PRINT TAB(10);"ITEM 4:  Vapor enthalpy coefficient, Av",LIN(1)
210   PRINT TAB(10);"ITEM 5:  Vapor enthalpy coefficient, Bv",LIN(1)
220   PRINT TAB(10);"ITEM 6:  Latent heat of vaporization at 0 Cen
      tigrade",LIN(1)
230   PRINT TAB(10);"ITEM 7:  Liquid enthalpy coefficient, Al",LIN
      (1)
240   PRINT TAB(10);"ITEM 8:  Liquid enthalpy coefficient, Bl",LIN
      (1)
250   FOR N=Ncf TO Ncl
260   FOR I=1 TO 8
270   DISP "Component Number";N,"Item";I;":";Data(N,I);
280   INPUT Data(N,I)
290   NEXT I
300   NEXT N
310    FOR J=Ncf TO Ncl
320    Data(J,9)=1
330    NEXT J
340   PRINT PAGE
350   IF Lact=1 THEN 410
360   PRINT "The van Laar Equation is used in calculating the acti
      vity coefficients"
370   DISP "(";Data(1,10);")";
380   INPUT "What is the  binary activity coefficient of comp 1 ou
      t of comp 2?",Data(1,10)
390   DISP "(";Data(2,10);")";
400   INPUT "What is the  binary activity coefficient of comp 2 ou
      t of comp 1?",Data(2,10)
410   ! INITIATION SECTION
```

```
420  PRINT PAGE
430  DISP "(";Strm(Npot,21);")";
440  INPUT "What is the initial number of moles in the still pot?
     ",Strm(Npot,21)
450  DISP "(";Strm(Npot,22);")";
460  INPUT "What is the initial temperature of the still pot?",St
     rm(Npot,22)
470  FOR N=Nstgs+2 TO Npot-1
480  Strm(N,22)=Strm(Npot,22)
490  NEXT N
500  PRINT "What is the composition of the feed mixture, enter mo
     le fractions and press CONT key after each entry."
510  FOR N=Ncf TO Ncl
520  DISP "Component number";N;":";Strm(Npot,N);
530  INPUT Strm(Npot,N)
540  FOR J=Nstgs+2 TO Npot+2
550  Strm(J,N)=Strm(Npot,N)
560  NEXT J
570  NEXT N
580  PRINT PAGE
590  DISP "(";Deltap;")";
600  INPUT "What is the total pressure drop for the column? (in m
     m Hg)",Deltap
610  Ppot=760+Deltap
620  Strm(Npot,24)=Strm(1,24)=Ppot
630  FOR J=2 TO Nstgs+1
640  Jd=2*(Nstgs+2)-J
650  Strm(J,24)=Ppot-Deltap/Nstgs*(J-1)
660  Strm(Jd,24)=Strm(J,24)
670  CALL Equil(Jd,J)
680  CALL Enthl(Jd)
690  NEXT J
700  CALL Enthl(Nstgs+2)
710  Strm(Nstgs+2,24)=Strm(Npot+1,24)=Strm(Npot+2,24)=1
720  Strm(Nstgs+2,24)=Strm(Npot+1,24)=Strm(Npot+2,24)=1
730  Tothl=Nstgs*.03
740  DISP "(";Tothl;")";
750  INPUT "What is the total molar holdup of the column?",Tothl
760  Hl=Tothl/Nstgs
770  DISP "(";Rr;")";
780  INPUT "What is the Reflux Ratio?",Rr
790  DISP "(";Q;")";
800  INPUT "What is the heat input to the pot in calories per min
     ute?",Q
810  Den=0
820  FOR N=Ncf TO Ncl
830  Den=Den+Strm(Npot,N)*Data(N,6)
840  NEXT N
850  Strm(1,21)=Q/Den
860  IF Rr<3 THEN 910
870  FOR N=Nstgs+2 TO Npot-1
880  Strm(N,21)=Strm(1,21)
890  NEXT N
900  IF Rr>=3 THEN 940
910  FOR N=Nstgs+2 TO Npot-1
920  Strm(N,21)=.75*Strm(1,21)
```

```
930  NEXT N
940  Io=1
950  Dt=.02
960  PRINT "What is the integration order? Value of 1 is assumed
     if none given"
970  DISP "(";Io;")";
980  INPUT "1 is Simple Euler, 2 is Modified Euler, 4 is Runge-Ku
     tta.",Io
990  PRINT PAGE
1000  DISP "(";Dt;")";
1010  INPUT "What is the integration step size? .02 minutes assum
     ed if none given",Dt
1020  DISP "(";Fnr;")";
1030  INPUT "At what time do you want the program to stop?",Fnr
1040  DISP "(";Pri;")";
1050  INPUT "What is the print interval?",Pri
1060  Htc=.05
1070  DISP "(";Htc;")";
1080  INPUT "What is the hydraulic time constant? .05 assumed if
     none given",Htc
1090  Tprnt=0
1100  Lstr=1
1110  T=0
1120  CALL Equil(Npot,1)
1130  CALL Enthl(Npot)
1140  INPUT "Do you want to change any inputs?. YES =1, NO=0",Again
1150  IF Again=1 THEN 10
1160  INPUT "Do you want the program to return for another run? Y
     ES=1, NO=0",Rtrn
1170  ! DERIVATIVE SECTION
1180  CALL Bot(Npot-1,Npot,1,Q)
1190  FOR N=1 TO Nstgs
1200  CALL Stage(N,Npot-1-N,Npot-N,N+1,0,H1,Htc)
1210  NEXT N
1220  CALL Tcon(Nstgs+1,Npot+2)
1230  CALL Split(Npot+2,Nstgs+2,Npot+1,Rr)
1240  CALL Prl(Pri,Fnr,Nf)
1250  IF (Nf=1) AND (Rtrn=1) THEN PRINT "THIS RUN DONE! READY TO
     START ANOTHER!"
1260  IF (Nf=1) AND (Rtrn=1) THEN 10
1270  IF Nf=1 THEN 1310
1280  ! INTEGRATION SECTION
1290  CALL Inti(T,Dt,Io)
1300  GOTO 1180
1310  PRINT "PROGRAM IS FINISHED!! WHAT'S NEXT?"
1320  END
1330  SUB Enthv(I)
1340 COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt,
     Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
1350  Hv=0
1360  FOR N=Ncf TO Ncl
1370  Hv=Hv+Strm(I,N)*((Data(N,4)+Data(N,5)*Strm(I,22))*Strm(I,22
     )+Data(N,6))
1380  NEXT N
1390  Strm(I,23)=Hv
1400  SUBEND
```

```
1410  SUB Enthl(I)
1420 COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt,
     Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
1430  Hv=0
1440  FOR N=Ncf TO Ncl
1450  Hv=Hv+Strm(I,N)*(Data(N,7)+Data(N,8)*Strm(I,22))
1460  NEXT N
1470  Strm(I,23)=Hv*Strm(I,22)
1480  SUBEND
1490  SUB Equil(Il,Iv)
1500 COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt,
     Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
1510  Sm=0
1520  Sdy=0
1530  IF Lact=1 THEN 1550
1540  CALL Acty(Il)
1550  FOR N=Ncf TO Ncl
1560  Strm(Iv,N)=10^(Data(N,1)-Data(N,2)/(Strm(Il,22)+Data(N,3)))
     *Strm(Il,N)/Strm(Il,24)*Data(N,9)
1570  Sm=Sm+Strm(Iv,N)
1580  Dy=Strm(Iv,N)*2.3025851*Data(N,2)/(Strm(Il,22)+Data(N,3))^2
1590  Sdy=Sdy+Dy
1600  NEXT N
1610  Yer=1-Sm
1620  Strm(Il,22)=Strm(Il,22)+Yer/Sdy
1630  IF ABS(Yer)<.0001 THEN GOTO 1650
1640  GOTO 1510
1650  Strm(Iv,22)=Strm(Il,22)
1660  SUBEND
1670  SUB Int(X,Dx)
1680  COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt
     ,Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
1690  Jn=Jn+1
1700  ON Io GOTO 1710,1730,1790,1790
1710  X=X+Dx*Dt
1720  SUBEXIT
1730  ON Js GOTO 1740,1770
1740  Dxa(Jn)=Dx
1750  X=X+Dx*Dt
1760  SUBEXIT
1770  X=X+(Dx-Dxa(Jn))*Dt/2
1780  SUBEXIT
1790  ON Js4 GOTO 1800,1840,1870,1900
1800  Xa(Jn)=X
1810  Dxa(Jn)=Dx
1820  X=X+Dx*Dt
1830  SUBEXIT
1840  Dxa(Jn)=Dxa(Jn)+2*Dx
1850  X=Xa(Jn)+Dx*Dt
1860  SUBEXIT
1870  Dxa(Jn)=Dxa(Jn)+2*Dx
1880  X=Xa(Jn)+Dx*Dt
1890  SUBEXIT
1900  Dxa(Jn)=(Dxa(Jn)+Dx)/6
1910  X=Xa(Jn)+Dxa(Jn)*Dt
1920  SUBEND
```

```
1930  SUB Inti(Td,Dtd,Iod)
1940  COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt
     ,Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
1950  Io=Iod
1960  Jn=0
1970  ON Io GOTO 1980,2000,2070,2070
1980  Js=2
1990  GOTO 2030
2000  Js=Js+1
2010  IF Js=3 THEN Js=1
2020  IF Js=2 THEN SUBEXIT
2030  Dt=Dtd
2040  Td=Td+Dt
2050  T=Td
2060  SUBEXIT
2070  Js4=Js4+1
2080  IF Js4=5 THEN Js4=1
2090  IF Js4=1 THEN 2120
2100  IF Js4=3 THEN GOTO 2140
2110  SUBEXIT
2120  Dt=Dtd/2
2130  GOTO 2040
2140  Td=Td+Dt
2150  Dt=2*Dt
2160  T=Td
2170  SUBEND
2180  SUB Prl(Pri,Fnr,Nf)
2190  COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt
     ,Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
2200  IF Lstr=1 THEN 2230
2210  IF (T>=Tprnt-Dt/2) AND ((Js=2) OR (Js4=4)) THEN 2240
2220  SUBEXIT
2230  Lstr=Nf=0
2240  Tprnt=Tprnt+Pri
2250  PRINTER IS 0
2260  IMAGE 1X, "TIME = ",D.DDDE
2270  PRINT
2280  PRINT USING 2260;T
2290  PRINT
2300  PRINT "STRM#   FLOW        TEMP       COMP 1      COMP 2      C
     OMP 3      COMP 4      COMP 5"
2310  FOR J=1 TO Np
2320  PRINT USING Format2;J,Strm(J,21),Strm(J,22)
2330 Format2:  IMAGE #,DDDX,2(XXD.DDDE)
2340  FOR I=Ncf TO Ncl
2350  PRINT USING Format3;Strm(J,I)
2360 Format3:  IMAGE #,XX.DDDDE
2370  NEXT I
2380  PRINT
2390  NEXT J
2400  IF T>=Fnr-Dt/3 THEN 2430
2410  PRINTER IS 16
2420  SUBEXIT
2430  T=0
2440  Tprnt=0
2450  MAT Xa=ZER
```

```
2460  PRINTER IS 16
2470  Nf=1
2480  SUBEND
2490  SUB Stage(I1,I2,Il,Iv,H,Hl,Htc)
2500  COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt
     ,Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
2510  CALL Equil(Il,Iv)
2520  CALL Enthl(Il)
2530  CALL Enthv(Iv)
2540  Flin=Strm(I1,21)+Strm(I2,21)+Rct(21)
2550  Hin=Strm(I1,23)*Strm(I1,21)+Strm(I2,23)*Strm(I2,21)+H+Rct(22)
2560  Strm(Iv,21)=(Hin-Flin*Strm(Il,23))/(Strm(Iv,23)-Strm(Il,23))
2570  Dl=(Flin-Strm(Iv,21)-Strm(Il,21))/Htc
2580  FOR N=Ncf TO Ncl
2590  Hk=Strm(Iv,N)/Strm(Il,N)
2600  Cnin=Strm(I1,21)*Strm(I1,N)+Strm(I2,21)*Strm(I2,N)+Rct(N)
2610  IF Hk>5 THEN GOTO 2670
2620  Dern=(Cnin-Flin*Strm(Il,N)-Strm(Iv,21)*(Strm(Iv,N)-Strm(Il,
     N)))/Hl
2630  CALL Int(Strm(Il,N),Dern)
2640  NEXT N
2650  CALL Int(Strm(Il,21),Dl)
2660  SUBEXIT
2670  Strm(Iv,N)=Cnin/(Strm(Iv,21)+Strm(Il,21)/Hk)
2680  Strm(Il,N)=Strm(Iv,N)/Hk
2690  Dern=0
2700  GOTO 2630
2710  SUBEND
2720  SUB Tcon(I,Jo)
2730  COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt
     ,Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
2740  FOR N=Ncf TO Ncl
2750  Strm(Jo,N)=Strm(I,N)
2760  NEXT N
2770  Strm(Jo,22)=Strm(I,22)-2
2780  Strm(Jo,21)=Strm(I,21)
2790  CALL Enthl(Jo)
2800  SUBEND
2810  SUB Split(C,Rflx,D,Rr)
2820  COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt
     ,Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
2830  IF Rr>=1000 THEN 2960
2840  Strm(D,21)=Strm(C,21)/(1+Rr)
2850  Strm(Rflx,21)=Strm(D,21)*Rr
2860  Strm(Rflx,22)=Strm(C,22)
2870  Strm(D,22)=Strm(C,22)
2880  IF Rr=0 THEN 3070
2890  Strm(Rflx,23)=Strm(C,23)
2900  Strm(D,23)=Strm(C,23)
2910  FOR N=Ncf TO Ncl
2920  Strm(Rflx,N)=Strm(C,N)
2930  Strm(D,N)=Strm(C,N)
2940  NEXT N
2950  SUBEXIT
2960  Strm(Rflx,21)=Strm(C,21)
2970  Strm(D,21)=0
```

```
2980  Strm(Rflx,22)=Strm(C,22)
2990  Strm(D,22)=Strm(C,22)
3000  Strm(Rflx,23)=Strm(C,23)
3010  Strm(D,23)=0
3020  FOR N=Ncf TO Ncl
3030  Strm(Rflx,N)=Strm(C,N)
3040  Strm(D,N)=0
3050  NEXT N
3060  SUBEXIT
3070  Strm(Rflx,23)=0
3080  Strm(D,23)=Strm(C,23)
3090  FOR J=1 TO Ncl
3100  Strm(D,J)=Strm(C,J)
3110  Strm(Rflx,J)=0
3120  NEXT J
3130  SUBEND
3140  SUB Bot(Li,Lo,Iv,Q)
3150  COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt
     ,Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
3160  IF (Js4=4) OR (Js=2) OR (Lstr=1) THEN Strm(Lo,20)=Strm(Lo,23)
3170  CALL Equil(Lo,Iv)
3180  CALL Enthl(Lo)
3190  CALL Enthv(Iv)
3200  Denl=(Strm(Lo,23)-Strm(Lo,20))*Strm(Lo,21)/Dt
3210  Qp=Q+Strm(Li,21)*Strm(Li,23)+Rct(22)
3220  Dhl=Strm(Li,21)-Strm(Iv,21)+Rct(21)
3230  Den=Strm(Iv,23)-Strm(Lo,23)
3240  Strm(Iv,21)=(Qp-Strm(Lo,23)*(Strm(Li,21)+Rct(21))-Denl)/Den
3250  IF Strm(Iv,21)<0 THEN Strm(Iv,21)=0
3260  FOR N=Ncf TO Ncl
3270  Hk=Strm(Iv,N)/Strm(Lo,N)
3280  IF Hk>20 THEN 3360
3290  Fni=Strm(Li,21)*Strm(Li,N)+Rct(N)
3300  Fno=Strm(Iv,21)*Strm(Iv,N)
3310  Dern=(Fni-Fno-Strm(Lo,N)*Dhl)/Strm(Lo,21)
3320  CALL Int(Strm(Lo,N),Dern)
3330  NEXT N
3340  CALL Int(Strm(Lo,21),Dhl)
3350  SUBEXIT
3360  Strm(Iv,N)=(Strm(Li,21)*Strm(Li,N)+Rct(N))/Strm(Iv,21)
3370  Strm(Lo,N)=Strm(Iv,N)/Hk
3380  Dern=0
3390  GOTO 3320
3400  SUBEND
3410  SUB Acty(N)
3420  COM Strm(300,24),Data(20,14),Rct(22),Ncf,Ncl,Lstr,Lact,T,Dt
     ,Js,Jn,Dxa(500),Xa(500),Io,Js4,Ns,Np,Tprnt,Rtrn
3430  A12=Data(1,10)
3440  A21=Data(2,10)
3450  X1=Strm(N,1)
3460  X2=Strm(N,2)
3470  Data(1,9)=10^(A12*X2^2/(X1*A12/A21+X2)^2)
3480  Data(2,9)=10^(A21*X1^2/(X2*A21/A12+X1)^2)
3490  SUBEND
```

REFERENCES

1. Franks, R.G.E. "Modeling and Simulation in Chemical Engineering," John Wiley and Sons, New York, 1972.
2. Distefano, G.P. American Institute of Chemical Engineering Journal (1968), 14, 190.
3. Domenech, S.; Enjalbert, M. Computers and Chemical Engineering (1981), 5, 181.
4. Rooney, J.M. Chemical Engineering, May 14, 1984, 61.
5. Guy, J.L. Chemical Engineering, January 10, 1983, 99.
6. Boston, J.F. Paper presented before the Division of Industrial and Engineering Chemistry, American Chemical Society, Washington D.C. meeting, "Inside-Out Algorithms for Multicomponent Separation Process Calculations," September 11-13, 1979.
7. Boston, J.F.; Britt, H.T. Proceedings of an International Conference on the Foundations of Computer-Aided Chemical Process Design, Henniker, New Hampshire, "An Advanced System for the Simulation of Batch Distillation Operations," July 6-11, 1980.
8. Luyben, W.L. "Process Modeling," McGraw-Hill, New York, 1973.
9. Wang, J.C.; Henke, G.E. Hydrocarbon Processing (1966), 45, 155.
10. Tomich, J.F. American Institute of Chemical Engineering Journal (1970), 16, 229.
11. Herchen, W.; Hartmann, H. Computers and Chemical Engineering (1979), 3, 235.
12. Tímar, L.; Édes, J. Hungarian Journal of Industrial Chemistry (1982), 10, 107.
13. Russell, R.A. Chemical Engineering, October 17, 1983, 53.
14. Broyden, C.G. Mathematics and Computers (1965), 19, 577.

15. Gear, C.W. "Numerical Initial Value Problems in Ordinary Differential Equations," Prentice-Hall, Englewood Cliffs, New Jersey, 1971.

16. Smith, B.D. "Design of Equilibrium Stage Processes," M.S. Peters, Ed., McGraw-Hill, New York, 1963.

17. Treybal, R.E. "Liquid Extraction," McGraw-Hill, New York, 1951.

18. Bolles, W.L. In "Design of Equilibrium Stage Processes," M.S. Peters, Ed., McGraw-Hill, New York, 1963.

19. Hála, E.; Wichterle, I.; Polak, J.; Boublik, T. "Vapour Liquid Equilibrium Data at Normal Pressures," Pergamon Press, Oxford, 1968.