COMPUTER-AIDED CHEMICAL ENGINEERING

Advisory Editor: L.M. Rose

Volume 1: Distillation Design in Practice (L.M. Rose)
Volume 2: The Art of Chemical Process Design (G.L. Wells and L.M. Rose)
Volume 3: Computer Programming Examples for Chemical Engineers (G. Ross)

COMPUTER-AIDED CHEMICAL ENGINEERING, 3

COMPUTER PROGRAMMING EXAMPLES FOR CHEMICAL ENGINEERS

GEORGE ROSS

Department of Manufacturing Engineering, Swinburne Institute of Technology, Hawthorn, Victoria, Australia

ELSEVIER
Amsterdam —Oxford — New York — Tokyo 1987

ELSEVIER SCIENCE PUBLISHERS B.V.
Sara Burgerhartstraat 25
P.O. Box 211, 1000 AE Amsterdam, The Netherlands

Distributors for the United States and Canada:

ELSEVIER SCIENCE PUBLISHING COMPANY INC.
52, Vanderbilt Avenue
New York, NY 10017, U.S.A.

ISBN 0-444-42836-4 (Vol. 3) (hardbound)
ISBN 0-444-42837-2 (Vol. 3) (paperback)
ISBN 0-444-42482-2 (Series)

Printed in The Netherlands

INTRODUCTION

Purpose

In the chemical engineering design field, computers are used in the preparation of heat and mass balances, the selection of optimum flow paths, the design of equipment items such as heat exchangers and distillation columns, and for the preparation of flow diagrams, layouts and mechanical drawings. Laboratory applications include a range of chemical analyses, determinations of particle size ranges, etc. whilst on the plant, computers and microprocessors are rapidly replacing conventional analog measurement and control devices.

How is the average chemical engineer equipped to handle these new developments? For many older engineers, the computer came along only as an afterthought in their studies, if it came at all. Current students of course, are much better equipped and receive tuition in programming, so that they are able to write programs in one or more of the common languages such as Basic, Fortran, Pascal, APL & C, but even they have been overtaken by the recent enormous increase in both power and availability of micro or desk-top computers such as the IBM PC. They are unlikely also, before completing their tertiary studies, to gain much experience writing programs related to their own engineering discipline.

Outside of teaching and research, many computers exclusively use software which has been developed by the computer manufacturers or other specialist companies. Such software packages are produced by large teams of programmers, numbering perhaps in the hundreds, working together for months under the direction of a co-ordinator. The average user of such a package is unlikely to be able to understand the program, even if he has access to it.

On entering employment, the young graduate may well find that where computers are used, such software packages are used with them. Many of these can now be run on a microcomputer equipped with a hard disc. In these circumstances, the user finds his function limited to the level of inputting data to the machine and making decisions with regard to its output, often without regard for the theoretical principles on which the software is based. I think this is a potentially dangerous situation, ultimately tending to degrade the profession, and restricting high level skills to the few.

The purpose of this book is to encourage the engineer to apply the skills in programming which he has learnt to the solution of problems in engineering. Programming such solutions is not difficult provided the programmer has a good understanding of the principles to be applied. Any gaps in this understanding will soon be uncovered! I am sure that an ability to write such programs, and the confidence which goes with it, is invaluable to the engineer working for the small company, which cannot justify the expense of the sophisticated software packages. For the engineer who does work with such packages, I hope this book will provide some insights into the logic which they employ, and encourage him also to develop his own programs.

I have selected a number of topics of interest to chemical engineers, a separate chapter being devoted to each. Each chapter presents the theoretical principles in summary form, then takes the reader through one or more manual calculations, in detail. Then a computer program is presented to perform the same task, each program including a detailed description, and being preceded by a logic flowchart.

Programming Language

The programs in this book are all written in BASIC.

I have chosen to do this for several reasons. It is the most widely used programming language and is acknowledged to be the easiest to learn, and it was the first language adopted for use on microcomputers, with which it is still widely used. Despite certain limitations with regard to speed and storage, Basic in its more developed forms is quite powerful enough for the average user.

The programs are prepared on IBM PC and PC/XT machines employing Microsoft Basic. The diskette which may be purchased with this book will of course run on these machines and on other IBM - compatible machines. The form of Basic used is in conformity with A.N.S.I. (American National Standards Institute) and so is suitable for many other Basic compilers.

In certain cases the diskette may be unsuitable; this occurs where the byte size employed in a particular machine is different from that used on the IBM machine. In such a case, unless a translator is available, there is no recourse but to type the programs in at the keyboard. Complete program listings are provided which enable this to be done.

Learning to Program

It may be that you have not yet learned to program. Don't be put off, it is not difficult. The first step is to obtain a Basic Manual appropriate to the machine you will be using. Next obtain a teaching text, of which a number are available (1) (2) (3) (4) (5). With these in hand you should be able to follow all the programs given in this book, and to extend them.

If you wish to progress to more advanced topics, there are also texts which will help you to do this (6) (7).

Logic Flowcharts

These are intended to make it easier to follow the program logic. They display the principal features of each program only, and employ symbols now standardised by A.N.S.I. and I.S.O. (International Organisation for Standardisation). The symbols used in this book are tabulated below, with brief summaries of their functions. For more information on flowcharts consult references already cited (2) (3).

Mathematics

The jobs which the chemical engineer is likely to use the computer for fall into the following categories:

1. Storage of data (for example, physical & thermodynamic properties).
2. Interpretation of data (that is, finding a mathematical relationship to fit the data).
3. Solution of problems involving stagewise processes (distillation, liquid extraction, evaporation, etc), treated as a series of lumped parameter systems.
4. Solution of problems in heat, mass and momentum transfer possibly involving transients. These are usually distributed parameter systems, but may be modelled as series of lumped parameter systems by the methods of finite differences (also by finite element methods).

The techniques employed to solve these problems are simply:

- Standard analytical mathematical methods
- Statistical methods
- Finite difference methods

Usually, large numbers of simultaneous equations are generated and have to be solved. This can be done either by Iteration, or by Matrix Algebra.

The chemical engineer who wishes to do his own programming then has to have some facility with each of the above. However, the necessary skills can be developed as the need arises.

Warnings

Commercial software includes a great deal of programming whose only function is to protect the software and to avoid user mistakes. Very little of such measures is included with the programs in this book; it makes understanding of the programs very much harder. If for instance, the program calls for values of composition in weight fraction, then the user must ensure that the sum of these values equals unity. If they do not, then errors will arise. So it is necessary to be careful in your working. If you don't like this situation then you can obtain excellent programming practice by incorporating your own safety precautions!

It is recommended that you purchase the program diskette if possible. It will save you many hours of typing and then checking for your mistakes. If you do possess the diskette, make a copy of it immediately, and use this copy when running or working on the programs. The original diskette should be retained as a master copy and only used to restore your working diskette if this has been corrupted.

The asterisk * is used in the text to indicate multiplication, in order to avoid confusion with the letter x.

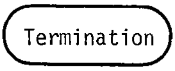
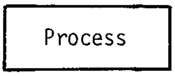
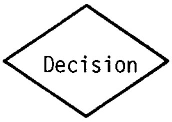
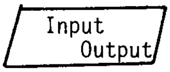
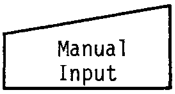
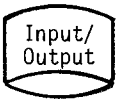
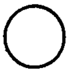
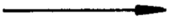
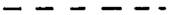
This book has been written to encourage chemical engineers to develop their programming skills. I hope you find it helpful.

G. Ross
Swinburne Institute of Technology
Melbourne 1987

REFERENCES - INTRODUCTION

1 W. Amsbury, Structured Basic and Beyond, Computer Science Press, Rockville MD, U.S.A. 1980
2 H.A. Moriber, Structured Basic Programming, Charles E. Merrill Publishing Co. Columbus, OH, U.S.A., 1984
3 R.G. Thompson, Basic - A Modular Approach, Charles E. Merrill Publishing Co. Columbus, OH, U.S.A., 1985
4 L. & J. Rosenblatt, Simplified Basic Programming: with Companion Problems, Addison - Wesley Publishing Co., Reading Mass, U.S.A. 1973
5 S.L. Marateck, Basic, Academic Press, New York, U.S.A., 1975
6 I. Stewart, R. Jones, Machine Code and Better Basic, Shiva Publishing, Nantwich, U.K. 1982
7 M.L. Lesser, Using Microsoft Compiled Basic, McGraw-Hill Book Co., New York, U.S.A., 1985

FLOW CHART SYMBOLS

Symbol	Meaning or Application
	This symbol indicates a terminal point, such as the beginning or end of a program or subroutine.
	This stands for any function or group of functions causing changes in the information flow, for example, it could be a sequence of algorithms used in a calculation.
	Statements causing branching, such as IF ... THEN or FOR ... NEXT. An abbreviated expression for the decision is placed inside the symbol with a ? below it.
	For example, input from a data statement, output to CRT or printer.
	Input from the keyboard.
	Input/Output from/to a magnetic aisc or similar device.
	Connector, used for convenience in laying out the flowchart, it has no counterpart in tne actual program.
	A line with an arrowheaa is used to link symbols; the direction of the arrow indicates the sequence of operations and the direction of the data flow.
	The dotted line has no counterpart in the actual program; its purpose is to indicate the symbols referred to in a comment or annotation appended to the flowchart.

Chapter 1

FLOWSHEETING (PROCESS SIMULATION)

Preparation of flowsheets is an important part of the work of the design chemical engineer. The flowsheet is the bridge between design calculation and plant hardware. Choice of optimum number of stages of extraction, or of heat exchange, depends upon material and energy balance considerations, and may be regarded as part of the flowsheeting process.

Computer flowsheeting is an attempt to assemble many chemical engineering design functions into one package (1) (2). Software packages of considerable complexity are now available (3) (4). Usually the simulation is limited to steady state conditions, thus greatly reducing the complexity of the problems to be dealt with.

A program for flowsheeting purposes obviously cannot be written as a simple linear program. Instead it must consist of a control program and a number of subroutines. A different subroutine is required for each step or operation in the process. A good many of these would be required, and the list might include "mix", "split", "compress", "expand", "flash", "distil", "pump","heat exchange", "extract", "absorb", "settle", "react", etc.

The control program would operate in such a way that these subroutines could be called upon by the designer as required, interactively at the terminal. The designer would assemble the complete flowsheet by appropriate connections between the subroutines.

Each subroutine should perform the function of calculating material and energy balances for the process step which it represents. In order to do this it would generate (or call from other subroutines) data on physical properties of all the fluids and solids concerned with that process step (4). In addition, it might be necessary in some cases, such as a multi-component distillation, for further design calculations to be made within the subroutine.

Since capital and operating costs affect the process design, it might be necessary also to include further subroutines to handle cost calculations (4).

As the assembly of the flowsheet continues, the designer may wish to incorporate corrections and improvements; he may wish also from time to time to observe the effects of variation in process parameters. The program should include means to enable this to be done; it should not be necessary to start again from the beginning each time a modification has to be incorporated.

Obviously, a program to carry out all the functions mentioned above has become indeed a "software package". It will have developed gradually by the accumulation of new subroutines and by the extension of existing ones. Reference to advertisements in the technical press shows how software companies are constantly updating their material.

Typing in of the data and interpretation of the output will be difficult and confusing for a flowsheet of any complexity. Consequently graphical representations of the flowsheet and display of values would be extremely advantageous. However, it is not an essential part of the program, and the example in this chapter will show how a flowsheet program can be written in BASIC without the use of graphic output.

The program to be described involves no mathematics other than algebra; only simple process steps are included, but a more complex form of the program might incorporate material from the other chapters of this book. Writing of the program depends of course on the programmers ability to model the process (5).

The growth of the program will be described step by step so that the reader can follow the development of the logic.

STEP ONE

The beginning program must contain the minimum of necessary ingredients to operate in the intended manner, namely:

- the control program;
- a subroutine to handle data inputs and commands from the designer;
- a subroutine to simulate a simple process;
- a subroutine to handle the output of calculated values.

The Process Subroutine (MIX)

The operation of mixing of several process streams will be modelled. Given the flowrates, compositions and enthalpies of n entering streams, the flowrate, enthalpy and composition of the mixed exit stream should be calculated (see Figure 1.1). For the sake of simplicity the algorithms written here will be limited to the calculation of flowrate and composition (not enthalpy).

If we have J entering streams, then the flowrate leaving

$$F_{J+1} = F_1 + F_2 + \ldots . F_J \qquad (1.1)$$

If there are K components present in each stream then the composition of the exit stream,

$$x_{J+1,K} = \frac{(x_{1,K} * F_1 + x_{2,K} * F_2 + \ldots . \; x_{J,K} * F_J)}{F_{J+1}} \qquad (1.2)$$

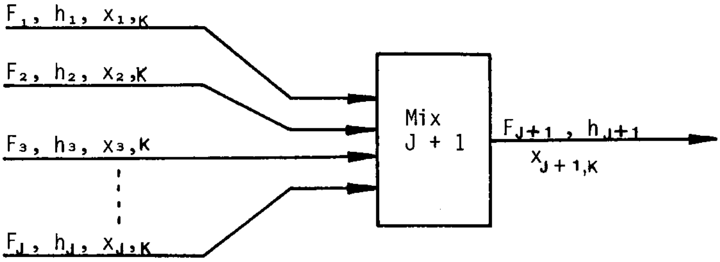
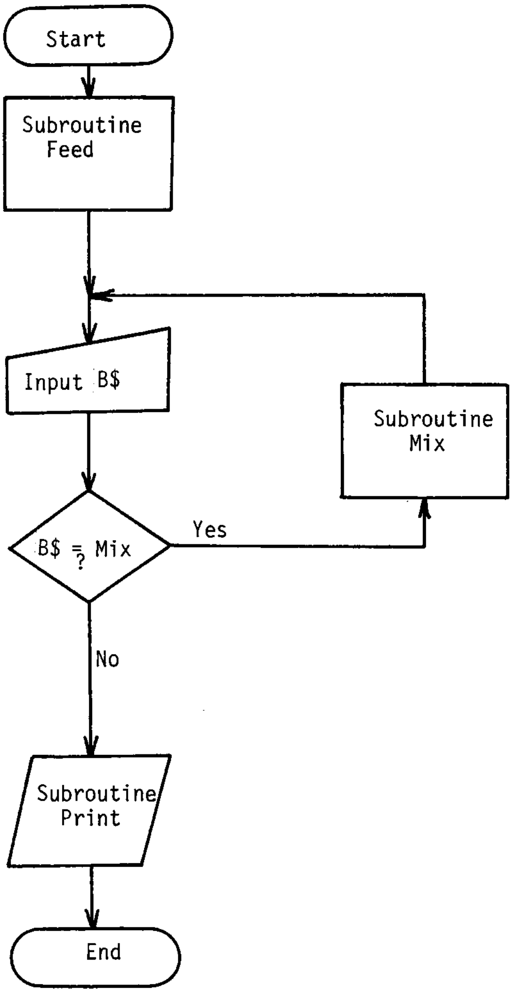


Fig. 1.1. Mixing of Streams.

This calculation should be performed for each component, i.e. for each value of the subscript K. In terms of the program, we can store all these values of composition in a two dimensional matrix X(J,K) where the first dimension refers to the number of the entering or leaving stream, and the second dimension refers to the components. We can then proceed to write a simple subroutine embodying the above two equations.

The program FSHT1 which follows, embodies this subroutine and the steps outlined above.

FSHT1. BAS


Start
Subroutine Feed
Input B$
B$ =? Mix
Yes
Subroutine Mix
No
Subroutine Print
End

```
10  REM ****************************************************
20  REM - PROGRAM FSHT1.BAS   FIRST STEP IN
30  REM - DEVELOPMENT OF A FLOWSHEET PROGRAM
40  REM - PROGRAM NOMENCLATURE:
50  REM - A$(J)    -    Names of subroutines and box numbers
60  REM - B$       -    Names of subroutines
70  REM - B(J)     -    Numbers of the streams entering
80  REM                 a subroutine box
90  REM - C$(J)    -    Names of components
100 REM - E1       -    Sum of values of F(J)
110 REM - E2       -    Sum of values of the product F*x
120 REM - E3       -    Sum of values of weight fractions
130 REM - F(J)     -    Flowrates
140 REM - N1       -    Number of components
150 REM - N2       -    Number of "boxes"
160 REM - N3       -    Number of flows to be mixed
170 REM - X(J<K)   -    Composition as weight fraction of
180 REM                 component K in stream from Box J
190 REM - Y(J<K)   -    Composition as weight fraction of
200 REM                 component K in stream from Box J
210 REM - PROGRAM DESCRIPTION
220 REM - LINES 1000,1010 Alphanumeric arrays are declared,
230 REM   and the arbitrary value of 20 allocated for the
240 REM - largest likely number of operations or "boxes"
250 REM - on the flowsheet.
260 REM - LINES 1020-1110 The control program;subroutine
270 REM - "Feed" is called first;then the designer is
280 REM - enabled to call subroutine "Mix" as required;
290 REM - finally subroutine "Print" is called.
300 REM - LINES 1130-1390 FEED subroutine;
310 REM - first, the number of components present is
320 REM - is established and these are named
330 REM - (lines 1180-1220).   Next, each entering stream
340 REM - is given a number or identifier referred to as
350 REM - a "Box Number", which the designer must keep a
360 REM - record of on his flow diagram.   As each stream is
370 REM - numbered it is also given the name "Feed", and
380 REM - values of flowrate and weight fraction of each
390 REM - component are entered (lines 1230-1380)
400 REM - LINES 1420-1650 MIX subroutine;each time this
410 REM - subroutine is called a box number is allocated
420 REM - and tbe box labelled "Mix" (lines 1430-1450).  The
430 REM - designer then enters the number of streams to be
440 REM - mixed and the boxes from which they arise(lines
450 REM - (1460-1490).   The algorithms previously written
460 REM - (equations 1.1 and 1.2) are then employed to
470 REM - calculate flowrate and composition of the mixed
480 REM - stream leaving(lines 1500-1640).
490 REM - LINES 1670-1780 PRINT subroutine; the name and
500 REM - number of each box is printed out with values of
510 REM - flowrate and weight fraction of each component.
520 REM - **************************************************
1000 DIM A$(20),C$(10)
1010 DIM B(20),F(20),X(20,10),Y(20,10)
1020 GOSUB 1130
1030 PRINT "ENTER MIX,OR END"
1040 INPUT B$
1050 IF B$="MIX" THEN 1080
1060 IF B$="END" THEN 1100
1070 GOTO 1030
1080 GOSUB 1400
```

```
1090 GOTO 1030
1100 GOSUB 1670
1110 GOTO 1790
1120 REM - *************************************************
1130 REM - FEED SUBROUTINE FOR FLOWRATES &
1140 REM - COMPOSITIONS OF ENTERING STREAMS
1150 PRINT "FLOWRATES & COMPOSITIONS FOR FLOWS"
1160 PRINT "ENTERING THE SYSTEM;ENTER THE NUMBER"
1170 PRINT "OF COMPONENTS & STICK TO THIS NUMBER"
1180 INPUT "FOR ALL STREAMS";N1
1190 FOR K=1 TO N1
1200 PRINT "NAME OF COMPONENT";K;
1210 INPUT C$(K)
1220 NEXT K
1230 PRINT "NUMBER OF STREAMS"
1240 INPUT N2
1250 FOR J=1 TO N2
1260 A$(J)="FEED"
1270 PRINT "FLOW";J;":"
1280 PRINT "FLOWRATE";
1290 INPUT F(J)
1300 IF F(J)=0 THEN 1390
1310 E3=0
1320 FOR K=1 TO N1-1
1330 PRINT "WT. FRACTION OF ";C$(K);
1340 INPUT X(J,K)
1350 E3=E3+X(J,K)
1360 NEXT K
1370 X(J,N1)=1-E3
1380 NEXT J
1390 RETURN
1400 REM - *************************************************
1410 REM - MIX SUBROUTINE, SIMULATES MIXING OF
1420 REM - STREAMS CONTAINING UP TO 10 COMPONENTS
1430 N2=N2+1
1440 PRINT "THIS IS MIX SUBROUTINE,BOX NUMBER";N2
1450 A$(N2)="MIX"
1460 INPUT "NUMBER OF STREAMS TO BE MIXED";N3
1470 FOR J=1 TO N3
1480 INPUT "BOX NUMBER FROM WHICH STREAM COMES";B(J)
1490 NEXT J
1500 E1=0
1510 FOR J=1 TO N3
1520 E1=E1+F(B(J))
1530 NEXT J
1540 FOR K=1 TO N1
1550 E2=0
1560 FOR J=1 TO N3
1570 E2=E2+F(B(J))*X(B(J),K)
1580 NEXT J
1590 Y(N2,K)=E2/E1
1600 NEXT K
1610 F(N2)=E1
1620 FOR K=1 TO N1
1630 X(N2,K)=Y(N2,K)
1640 NEXT K
1650 RETURN
1660 REM - *************************************************
1670 REM - PRINT SUBROUTINE
1680 FOR J=1 TO N2
1690 PRINT A$(J)
```

```
1700 PRINT "BOX NUMBER";J
1710 PRINT "FLOWRATE=";F(J)
1720 FOR K=1 TO N1
1730 PRINT "WEIGHT FRACTION OF ";C$(K);"=";
1740 PRINT USING "#.####";X(J,K)
1750 NEXT K
1760 PRINT
1770 NEXT J
1780 RETURN
1790 END
```

Example 1.1

A typical recipe for a barrier cream is as follows:

A:	stearic acid	14.0 kg
	zinc stearate	4.0 kg
	solubilising agent	2.5 kg
	water repellent	2.0 kg
B:	sorbitol (70%)	5.0 kg
	water	42.5 kg
C:	4% mucilage of methyl cellulose	23.0 kg

Mixes A and B are prepared separately, emulsified together and then Mix C is added.

Calculate the flowrates and compositions of all streams using the program FSHT1.

This is a trivial problem, but it serves to demonstrate the flexible nature of the program even at this beginning stage. A simple block diagram is first drawn, which indicates the numbering of the boxes (Fig. 1.2). Next the program is run and values entered, as follows:

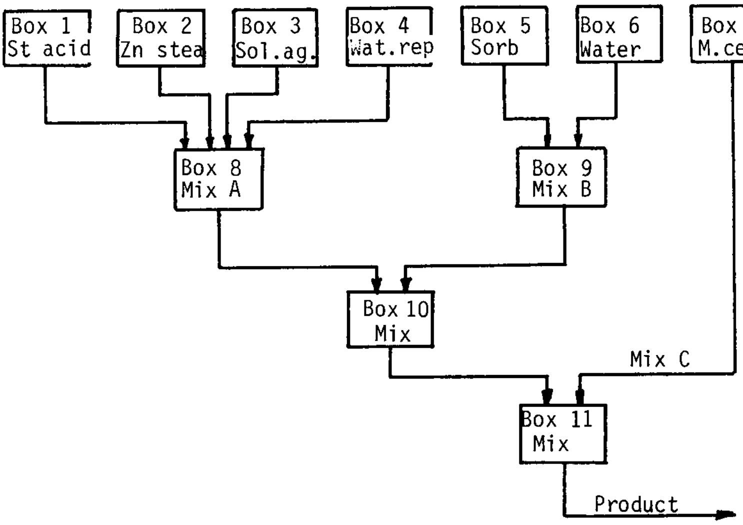


Fig. 1.2. Mixing of Barrier Cream.

```
RUN
FLOWRATES & COMPOSITIONS FOR FLOWS
ENTERING THE SYSTEM;ENTER THE NUMBER
OF COMPONENTS & STICK TO THIS NUMBER
FOR ALL STREAMS? 7
NAME OF COMPONENT 1 ? ST ACID
NAME OF COMPONENT 2 ? ZN STEA
NAME OF COMPONENT 3 ? SOL AGT
NAME OF COMPONENT 4 ? WATER REP
NAME OF COMPONENT 5 ? SORBITOL
NAME OF COMPONENT 6 ? METH CELL
NAME OF COMPONENT 7 ? WATER
NUMBER OF STREAMS
? 7
FLOW 1 :
FLOWRATE? 14
WT. FRACTION OF ST ACID? 1
WT. FRACTION OF ZN STEA? 0
WT. FRACTION OF SOL AGT? 0
WT. FRACTION OF WATER REP? 0
WT. FRACTION OF SORBITOL? 0
WT. FRACTION OF METH CELL? 0
FLOW 2 :
FLOWRATE? 4
WT. FRACTION OF ST ACID? 0
WT. FRACTION OF ZN STEA? 1
WT. FRACTION OF SOL AGT? 0
WT. FRACTION OF WATER REP? 0
WT. FRACTION OF SORBITOL? 0
WT. FRACTION OF METH CELL? 0
FLOW 3 :
FLOWRATE? 2.5
WT. FRACTION OF ST ACID? 0
WT. FRACTION OF ZN STEA? 0
WT. FRACTION OF SOL AGT? 1
WT. FRACTION OF WATER REP? 0
WT. FRACTION OF SORBITOL? 0
WT. FRACTION OF METH CELL? 0
FLOW 4 :
FLOWRATE? 2
WT. FRACTION OF ST ACID? 0
WT. FRACTION OF ZN STEA? 0
WT. FRACTION OF SOL AGT? 0
WT. FRACTION OF WATER REP? 1
WT. FRACTION OF SORBITOL? 0
WT. FRACTION OF METH CELL? 0
FLOW 5 :
FLOWRATE? 5
WT. FRACTION OF ST ACID? 0
WT. FRACTION OF ZN STEA? 0
WT. FRACTION OF SOL AGT? 0
WT. FRACTION OF WATER REP? 0
WT. FRACTION OF SORBITOL? .7
WT. FRACTION OF METH CELL? 0
FLOW 6 :
FLOWRATE? 42.5
WT. FRACTION OF ST ACID? 0
WT. FRACTION OF ZN STEA? 0
WT. FRACTION OF SOL AGT? 0
WT. FRACTION OF WATER REP? 0
WT. FRACTION OF SORBITOL? 0
```

```
WT. FRACTION OF METH CELL? 0
FLOW 7 :
FLOWRATE? 23
WT. FRACTION OF ST ACID? 0
WT. FRACTION OF ZN STEA? 0
WT. FRACTION OF SOL AGT? 0
WT. FRACTION OF WATER REP? 0
WT. FRACTION OF SORBITOL? 0
WT. FRACTION OF METH CELL? .04
ENTER MIX,OR END
? MIX
THIS IS MIX SUBROUTINE,BOX NUMBER 8
NUMBER OF STREAMS TO BE MIXED? 4
BOX NUMBER FROM WHICH STREAM COMES? 1
BOX NUMBER FROM WHICH STREAM COMES? 2
BOX NUMBER FROM WHICH STREAM COMES? 3
BOX NUMBER FROM WHICH STREAM COMES? 4
ENTER MIX,OR END
? MIX
THIS IS MIX SUBROUTINE,BOX NUMBER 9
NUMBER OF STREAMS TO BE MIXED? 2
BOX NUMBER FROM WHICH STREAM COMES? 5
BOX NUMBER FROM WHICH STREAM COMES? 6
ENTER MIX,OR END
? MIX
THIS IS MIX SUBROUTINE,BOX NUMBER 10
NUMBER OF STREAMS TO BE MIXED? 2
BOX NUMBER FROM WHICH STREAM COMES? 8
BOX NUMBER FROM WHICH STREAM COMES? 9
ENTER MIX,OR END
? MIX
THIS IS MIX SUBROUTINE,BOX NUMBER 11
NUMBER OF STREAMS TO BE MIXED? 2
BOX NUMBER FROM WHICH STREAM COMES? 7
BOX NUMBER FROM WHICH STREAM COMES? 10
ENTER MIX,OR END
? END
FEED
BOX NUMBER 1
FLOWRATE= 14
WEIGHT FRACTION OF ST ACID=1.0000
WEIGHT FRACTION OF ZN STEA=0.0000
WEIGHT FRACTION OF SOL AGT=0.0000
WEIGHT FRACTION OF WATER REP=0.0000
WEIGHT FRACTION OF SORBITOL=0.0000
WEIGHT FRACTION OF METH CELL=0.0000
WEIGHT FRACTION OF WATER=0.0000

FEED
BOX NUMBER 2
FLOWRATE= 4
WEIGHT FRACTION OF ST ACID=0.0000
WEIGHT FRACTION OF ZN STEA=1.0000
WEIGHT FRACTION OF SOL AGT=0.0000
WEIGHT FRACTION OF WATER REP=0.0000
WEIGHT FRACTION OF SORBITOL=0.0000
WEIGHT FRACTION OF METH CELL=0.0000
WEIGHT FRACTION OF WATER=0.0000

FEED
BOX NUMBER 3
```

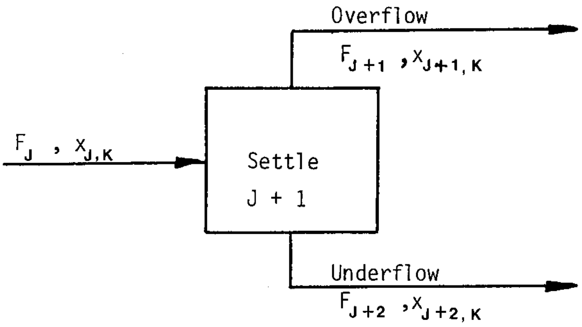
```
FLOWRATE= 2.5
WEIGHT FRACTION OF ST ACID=0.0000
WEIGHT FRACTION OF ZN STEA=0.0000
WEIGHT FRACTION OF SOL AGT=1.0000
WEIGHT FRACTION OF WATER REP=0.0000
WEIGHT FRACTION OF SORBITOL=0.0000
WEIGHT FRACTION OF METH CELL=0.0000
WEIGHT FRACTION OF WATER=0.0000

FEED
BOX NUMBER 4
FLOWRATE= 2
WEIGHT FRACTION OF ST ACID=0.0000
WEIGHT FRACTION OF ZN STEA=0.0000
WEIGHT FRACTION OF SOL AGT=0.0000
WEIGHT FRACTION OF WATER REP=1.0000
WEIGHT FRACTION OF SORBITOL=0.0000
WEIGHT FRACTION OF METH CELL=0.0000
WEIGHT FRACTION OF WATER=0.0000

FEED
BOX NUMBER 5
FLOWRATE= 5
WEIGHT FRACTION OF ST ACID=0.0000
WEIGHT FRACTION OF ZN STEA=0.0000
WEIGHT FRACTION OF SOL AGT=0.0000
WEIGHT FRACTION OF WATER REP=0.0000
WEIGHT FRACTION OF SORBITOL=0.7000
WEIGHT FRACTION OF METH CELL=0.0000
WEIGHT FRACTION OF WATER=0.3000

FEED
BOX NUMBER 6
FLOWRATE= 42.5
WEIGHT FRACTION OF ST ACID=0.0000
WEIGHT FRACTION OF ZN STEA=0.0000
WEIGHT FRACTION OF SOL AGT=0.0000
WEIGHT FRACTION OF WATER REP=0.0000
WEIGHT FRACTION OF SORBITOL=0.0000
WEIGHT FRACTION OF METH CELL=0.0000
WEIGHT FRACTION OF WATER=1.0000

FEED                           Mix C
BOX NUMBER 7
FLOWRATE= 23
WEIGHT FRACTION OF ST ACID=0.0000
WEIGHT FRACTION OF ZN STEA=0.0000
WEIGHT FRACTION OF SOL AGT=0.0000
WEIGHT FRACTION OF WATER REP=0.0000
WEIGHT FRACTION OF SORBITOL=0.0000
WEIGHT FRACTION OF METH CELL=0.0400
WEIGHT FRACTION OF WATER=0.9600

MIX                            Mix A
BOX NUMBER 8
FLOWRATE= 22.5
WEIGHT FRACTION OF ST ACID=0.6222
WEIGHT FRACTION OF ZN STEA=0.1778
WEIGHT FRACTION OF SOL AGT=0.1111
WEIGHT FRACTION OF WATER REP=0.0889
WEIGHT FRACTION OF SORBITOL=0.0000
WEIGHT FRACTION OF METH CELL=0.0000
WEIGHT FRACTION OF WATER=0.0000

MIX                      Mix B
BOX NUMBER 9
FLOWRATE= 47.5
WEIGHT FRACTION OF ST ACID=0.0000
WEIGHT FRACTION OF ZN STEA=0.0000
WEIGHT FRACTION OF SOL AGT=0.0000
WEIGHT FRACTION OF WATER REP=0.0000
WEIGHT FRACTION OF SORBITOL=0.0737
WEIGHT FRACTION OF METH CELL=0.0000
WEIGHT FRACTION OF WATER=0.9263

MIX                      A & B
BOX NUMBER 10
FLOWRATE= 70
WEIGHT FRACTION OF ST ACID=0.2000
WEIGHT FRACTION OF ZN STEA=0.0571
WEIGHT FRACTION OF SOL AGT=0.0357
WEIGHT FRACTION OF WATER REP=0.0286
WEIGHT FRACTION OF SORBITOL=0.0500
WEIGHT FRACTION OF METH CELL=0.0000
WEIGHT FRACTION OF WATER=0.6286

MIX                      Product
BOX NUMBER 11
FLOWRATE= 93
WEIGHT FRACTION OF ST ACID=0.1505
WEIGHT FRACTION OF ZN STEA=0.0430
WEIGHT FRACTION OF SOL AGT=0.0269
WEIGHT FRACTION OF WATER REP=0.0215
WEIGHT FRACTION OF SORBITOL=0.0376
WEIGHT FRACTION OF METH CELL=0.0099
WEIGHT FRACTION OF WATER=0.7105

Ok
```

STEP TWO

Having demonstrated the workability of the program, the next step is to add further subroutines representing process steps. Two simple examples have been selected.

Process Subroutine (SETTLE)

It is assumed that the feed to the settler is a two phase mixture of solid particles suspended in a solution of solute in solvent. In the settler, separation of the phases occurs, a solids-free solution leaving as overflow, the solids with the remaining solution leaving as underflow. (See Fig. 1.3).



K = 1 for solvent, 2 for solute, 3 for solid.
R1 = solids/solution ratio.

Fig. 1.3. Settling

It is further assumed that:

- the ratio of solids to solution in the underflow, is known;
- the concentration of solute in solvent is the same for both overflow and underflow liquors;
- all the solute present is in solution;
- the solid phase is insoluble in the solvent.

These assumptions are quite usual for leaching problems, as can be seen by reference to undergraduate chemical engineering texts (6) & (7).

Using the terminology of Figure 1.3 and that previously referred to, the following equations can be written:

Underflow

Weight of solids leaving = weight of solids entering = $F_J x_{J,3}$

Weight of solution leaving = $F_J x_{J,3}/R1$

Hence the total underflow rate,

$$F_{J+2} = F_J x_{J,3} * (1 + 1/R1) \tag{1.3}$$

Weight fraction of solid in the underflow,

$$x_{J+2,3} = \frac{F_J x_{J,3}}{F_{J+2}} = \frac{R1}{1+R1} \tag{1.4}$$

Weight of solute leaving in the underflow =

Wt. of solute entering $* \dfrac{\text{wt. solution in underflow}}{\text{total wt. of solution}} =$

$$F_J x_{J,2} * \frac{F_J x_{J,3}/R1}{F_J (x_{J,1} + x_{J,2})}$$

Hence the weight fraction of solute in the underflow =

$$x_{J+2,2} = \frac{F_J x_{J,2} x_{J,3}}{R1(x_{J,1} + x_{J,2})} * \frac{1}{F_J x_{J,3}(1 + 1/R1)}$$

$$\therefore x_{J+2,2} = \frac{x_{J,2}}{(x_{J,1} + x_{J,2})(R1 + 1)} \tag{1.5}$$

By difference, the weight fraction of solvent in the underflow,

$$x_{J+2,1} = 1 - x_{J+2,2} - x_{J+2,3} \tag{1.6}$$

Overflow

Weight of solution leaving = $F_{J+1} = F_J - F_{J+2}$ (1.7)

Based on the assumption of a solids-free overflow,

$$x_{J+1,3} = 0 \tag{1.8}$$

Weight of solute leaving in the overflow =

Wt. of solute entering $* \dfrac{\text{wt. solution in overflow}}{\text{total wt. of solution}} =$

$$F_J x_{J,2} * \frac{F_{J+1}}{F_J (x_{J,1} + x_{J,2})}$$

Hence the weight fraction of solute in the overflow,

$$x_{J+1,2} = \frac{x_{J,2}\,F_{J+1}}{x_{J,1} + x_{J,2}} * \frac{1}{F_{J+1}}$$

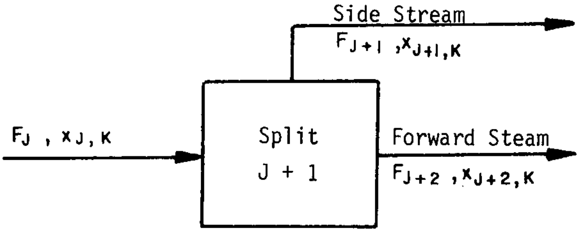
$$\therefore\ x_{J+1,2} = \frac{x_{J,2}}{x_{J,1} + x_{J,2}} \qquad (1.9)$$

By difference, the weight fraction of solvent in the underflow,

$$x_{J+1,1} = 1 - x_{J+1,2} - x_{J+1,3} \qquad (1.10)$$

Process Subroutine (SPLIT)

It is assumed that a process stream has to be subdivided, for example by removal of a purge stream, because of a leak, or by a by-pass arrangement. The situation is shown in Fig. 1.4.



R2 = sidestream/entering stream ratio.

Fig. 1.4. Splitting of a Stream.

Composition of both outlet streams will be the same, and equal to the inlet composition. Flowrate of the outlet streams are determined by the designated value of the ratio of sidestream flow, R2.

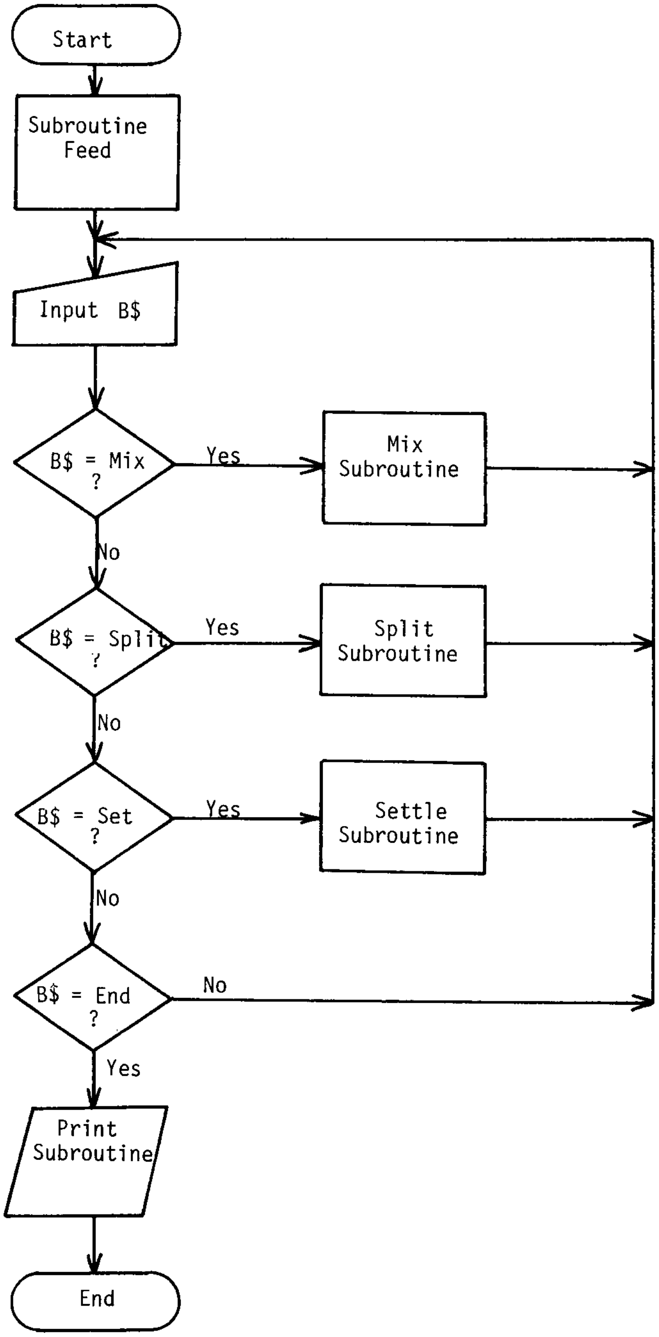
$$F_{J+1} = F_J * R2 \qquad (1.11)$$

$$F_{J+2} = F_J - F_{J+1} \qquad (1.12)$$

COMPUTER SOLUTION - STEP TWO

The program previously given has been modified by the insertion of the above two subroutines. Their operation is self explanatory. Of course the subroutines are added, tested and if necessary, debugged one at a time. This modified program is named FSHT2. The listing follows. No example of its use will be given until further modifications have been made.

FSHT2.BAS


Start
Subroutine Feed
Input B$
B$ = Mix ?
Yes
Mix Subroutine
No
B$ = Split ?
Yes
Split Subroutine
No
B$ = Set ?
Yes
Settle Subroutine
No
B$ = End ?
No
Yes
Print Subroutine
End

```
10 REM - ***************************************************
20 REM - PROGRAM FSHT2.BAS    SECOND STEP IN
30 REM - DEVELOPMENT OF A FLOWSHEET PROGRAM
40 REM - PROGRAM NOMENCLATURE   To the nomenclature given
50 REM - for program FSHT1.BAS has been added the following:
60 REM - R1          -   Solids/Solution ratio
70 REM - R2          -   Sidestream/entering stream ratio
80 REM - ***************************************************
1000 DIM A$(20),C$(10)
1010 DIM B(20),F(20),X(20,10),Y(20,10)
1020 GOSUB 1210
1030 PRINT "ENTER MIX,SPLIT,SETTLE,OR END"
1040 INPUT B$
1050 IF B$="MIX"  THEN 1120
1060 IF B$="SPLIT" THEN 1140
1070 IF B$="SETTLE" THEN  1160
1080 IF B$="SET"  THEN 1160
1090 IF B$="END" THEN 1180
1100 PRINT "INPUT NOT RECOGNISED"
1110 GOTO 1030
1120 GOSUB 1510
1130 GOTO 1030
1140 GOSUB 1790
1150 GOTO 1030
1160 GOSUB 1990
1170 GOTO 1030
1180 GOSUB 2240
1190 GOTO 2360
1200 REM - *************************************************
1210 REM - FEED SUBROUTINE FOR FLOWRATES &
1220 REM - COMPOSITIONS OF ENTERING STREAMS
1230 PRINT "FLOWRATES & COMPOSITIONS FOR FLOWS"
1240 PRINT "ENTERING THE SYSTEM;ENTER THE NUMBER"
1250 PRINT "OF COMPONENTS & STICK TO THIS NUMBER"
1260 PRINT "FOR ALL STREAMS (FOR SETTLING PROBLEMS USE:"
1270 PRINT "K=1 FOR SOLVENT,2 FOR SOLUTE,3 FOR SOLID)"
1280 INPUT N1
1290 FOR K=1 TO N1
1300 PRINT "NAME OF COMPONENT";K;
1310 INPUT C$(K)
1320 NEXT K
1330 PRINT "NUMBER OF STREAMS"
1340 INPUT N2
1350 FOR J=1 TO N2
1360 A$(J)="FEED"
1370 PRINT "FLOW";J;":"
1380 PRINT "FLOWRATE";
1390 INPUT F(J)
1400 IF F(J)=0 THEN 1490
1410 E3=0
1420 FOR K=1 TO N1-1
1430 PRINT "WT. FRACTION OF ";C$(K);
1440 INPUT X(J,K)
1450 E3=E3+X(J,K)
1460 NEXT K
1470 X(J,N1)=1-E3
1480 NEXT J
1490 RETURN
1500 REM - *************************************************
1510 REM - MIX SUBROUTINE, SIMULATES MIXING OF
1520 REM - STREAMS CONTAINING UP TO 10 COMPONENTS
```

```
1530 N2=N2+1
1540 PRINT "THIS IS MIX SUBROUTINE,BOX NUMBER";N2
1550 A$(N2)="MIX"
1560 PRINT "NUMBER OF STREAMS TO BE MIXED";
1570 INPUT N3
1580 FOR J=1 TO N3
1590 PRINT "BOX NUMBER FROM WHICH STREAM COMES";
1600 INPUT B(J)
1610 NEXT J
1620 E1=0
1630 FOR J=1 TO N3
1640 E1=E1+F(B(J))
1650 NEXT J
1660 FOR K=1 TO N1
1670 E2=0
1680 FOR J=1 TO N3
1690 E2=E2+F(B(J))*X(B(J),K)
1700 NEXT J
1710 Y(N2,K)=E2/E1
1720 NEXT K
1730 F(N2)=E1
1740 FOR K=1 TO N1
1750 X(N2,K)=Y(N2,K)
1760 NEXT K
1770 RETURN
1780 REM - *************************************************
1790 REM - SPLIT SUBROUTINE, SIMULATES SPLITTING OF
1800 REM - ONE STREAM INTO TWO IN A DESIGNATED RATIO
1810 N2=N2+2
1820 PRINT "THIS IS SPLIT SUBROUTINE,BOX NUMBER";N2-1
1830 PRINT "THIS NUMBER ALSO DESIGNATES SIDE STREAM."
1840 PRINT "FORWARD STREAM DESIGNATED BOX NUMBER";N2
1850 A$(N2-1)="SPLIT,SIDE STREAM"
1860 A$(N2)="SPLIT,FORWARD STREAM"
1870 PRINT "BOX NUMBER FROM WHICH STREAM COMES";
1880 INPUT B1
1890 PRINT "SIDE STREAM/TOTAL FLOW RATIO";
1900 INPUT R2
1910 F(N2-1)=F(B1)*R2
1920 F(N2)=F(B1)-F(N2-1)
1930 FOR K=1 TO N1
1940 X(N2-1,K)=X(B1,K)
1950 X(N2,K)=X(B1,K)
1960 NEXT K
1970 RETURN
1980 REM - *************************************************
1990 REM - SETTLE SUBROUTINE,SIMULATES OPERATION OF A
2000 REM - SETTLER.   1 ENTERING STREAM IS ASSUMED,
2010 REM - CONSISTING OF 3 COMPONENTS ONLY.   THESE ARE
2020 REM - NUMBERED 1 FOR SOLVENT, 2 FOR SOLUTE, 3 FOR
2030 REM - SOLID.   ZERO SOLIDS IN THE OVERFLOW IS ASSUMED
2040 N2=N2+2
2050 PRINT "THIS IS SETTLE SUBROUTINE,BOX NUMBER";N2-1
2060 PRINT "THIS NUMBER ALSO DESIGNATES OVERFLOW STREAM."
2070 PRINT "UNDERFLOW STREAM IS DESIGNATED BOX NUMBER";N2
2080 A$(N2-1)="SETTLE,OVERFLOW"
2090 A$(N2)="SETTLE,UNDERFLOW"
2100 PRINT "BOX NUMBER FROM WHICH STREAM COMES";
2110 INPUT B1
2120 PRINT "SOLIDS/SOLUTION RATIO FOR UNDERFLOW";
2130 INPUT R1
```

```
2140 F(N2)=F(B1)*X(B1,3)*(1+1/R1)
2150 F(N2-1)=F(B1)-F(N2)
2160 X(N2,3)=R1/(1+R1)
2170 X(N2,2)=X(B1,2)/((X(B1,1)+X(B1,2))*(R1+1))
2180 X(N2,1)=1-X(N2,2)-X(N2,3)
2190 X(N2-1,3)=0
2200 X(N2-1,2)=X(B1,2)/(X(B1,1)+X(B1,2))
2210 X(N2-1,1)=1-X(N2-1,2)
2220 RETURN
2230 REM - *************************************************
2240 REM - PRINT SUBROUTINE
2250 FOR J=1 TO N2
2260 PRINT A$(J)
2270 PRINT "BOX NUMBER";J
2280 PRINT "FLOWRATE=";F(J)
2290 FOR K=1 TO N1
2300 PRINT "WEIGHT FRACTION OF ";C$(K);"=";
2310 PRINT USING "#.####";X(J,K)
2320 NEXT K
2330 PRINT
2340 NEXT J
2350 RETURN
2360 END
```

STEP THREE

Many processes involve recycle loops, and in the development of the flowsheet the values of flowrate, composition, etc within the loop are usually unknown. The program so far will not handle this situation.

Where unknowns arise in a calculation in this way, it is frequently possible to solve for these by generating a sufficient number of simultaneous equations. This method is not applicable in the present case because in order to write the program it would be necessary to know the flowsheet beforehand.

The difficulty can be resolved by using an iterative method. Values are assumed for the composition and flowrate, etc of the unknown stream and these values are employed in the calculation. Entering values in this way is known as 'tearing' the stream (1). The calculation proceeds from box to box as before until the variables of the torn stream are recalculated. The assumed and recalculated values are compared; if they agree within specified limits then the calculation is complete, otherwise new values for the torn stream variables must be assumed (4).

The easiest way to assume new values is to employ the arithmetic mean between the previously assumed values and the recalculated ones. These new values provide the starting point for another calculation. Iteration proceeds in this manner until the values of all variables before and after the 'tear' are in agreement within the specified limits.

To modify the program in this way, an additional subroutine TEAR has been written.

Process Subroutine (TEAR)

The box numbers which each torn stream leaves and enters are entered by the designer. Flowrates and compositions of these streams are averaged (see Figure 1.5).

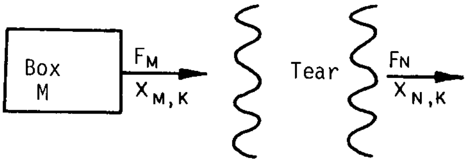


Fig. 1.5. The Torn Stream.

Thus the new value of F_N to be used at the next iteration, which we may call F_N^*, is:

$$F_N^* = \frac{F_M + F_N}{2}$$

Similarly,

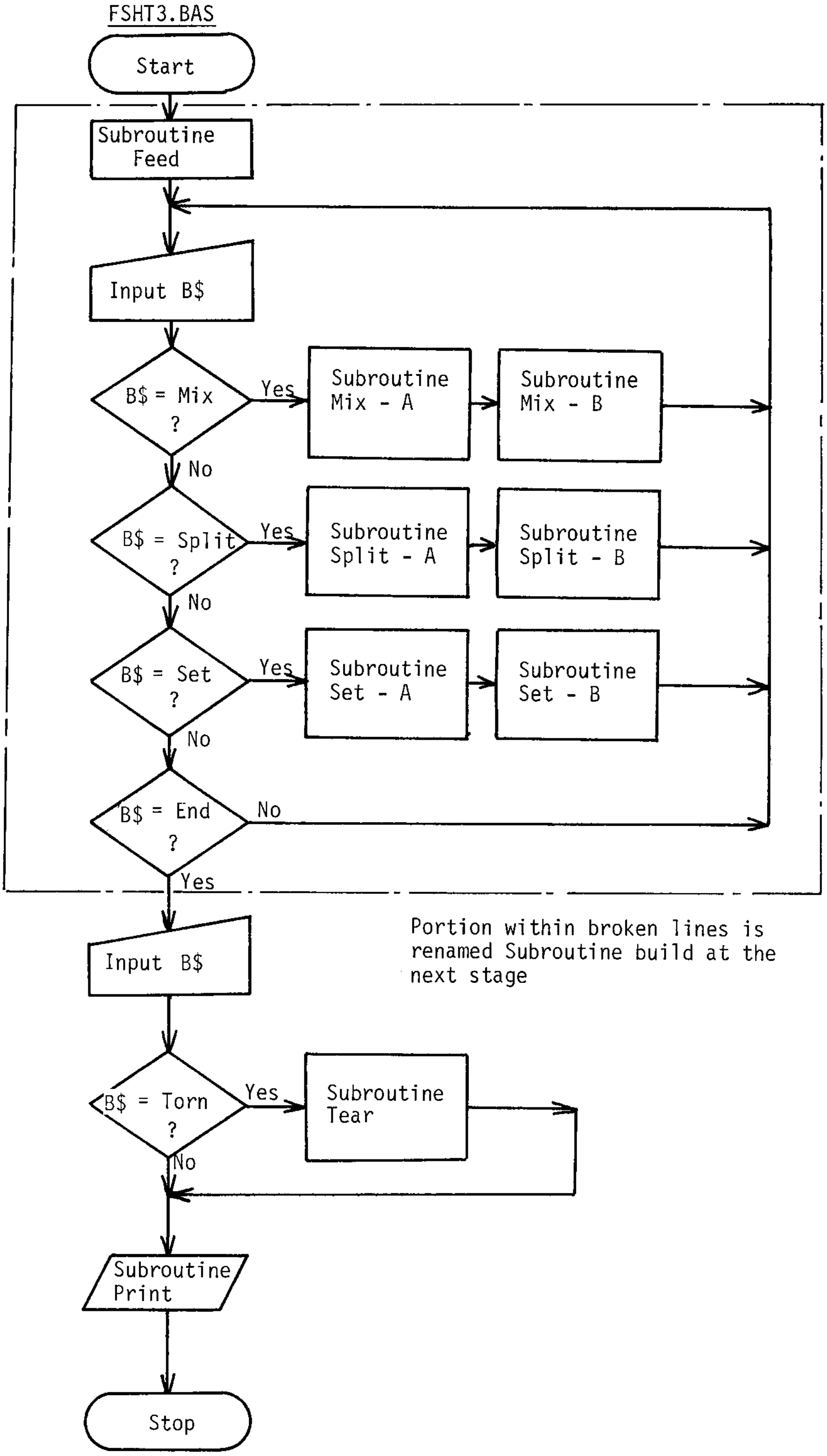
$$X_{N,K}^* = \frac{X_{M,K} + X_{N,K}}{2}$$

In the remainder of the subroutine new values for all variables in the flowsheet are recalculated.

Control is transferred to this subroutine when one or more torn streams occur. The program has been modified so as to incorporate these torn streams. This has necessitated certain other modifications namely:

> The first parts of subroutines 'Mix', 'Split' and 'Settle' were concerned with data acquisition. These parts have been made into separate subroutines (labelled Part A in each case), which are accessed only once. each time these operations are input by the designer. Part B of each subroutine is accessed as required at each iteration.

Program FSHT3 which follows, incorporates these modifications, and an example of its use is given.



Portion within broken lines is renamed Subroutine build at the next stage

```
10  REM ****************************************************
20  REM - PROGRAM FSHT3.BAS    THIRD STEP IN
30  REM - DEVELOPMENT OF A FLOWSHEET PROGRAM
40  REM - PROGRAM NOMENCLATURE  The following variables are
50  REM - used,in addition to those already in use
60  REM - (see FSHT1 and FSHT2):
70  REM - C1    -   Number of iterations to achieve the
80  REM             specified agreement in torn stream
90  REM -           values
100 REM - N4    -   Number of torn streams
110 REM - N5    -   Duplicate value of N2 used in certain
120 REM -           algorithms
130 REM - N(J)      Number of streams to be mixed at box J
140 REM - T(J,1)    Box number which torn stream J leaves
150 REM - T(J,2)    Box number which torn stream enters
160 REM - The following variables replace others previously
170 REM - used:
180 REM - B(J,K)    Box number of stream K entering box J
190 REM             replaces B(J) used in FSHT1 & FSHT2
200 REM - R(J)      Side stream/total flowratio, or solids/
210 REM             solution ratio, at box J.  Replaces R1
220 REM             & R2 used in FSHT2
230 REM - DETAILS OF SUBROUTINE TEAR
240 REM - LINES 2500 - 2570 The value of N4 is entered, that
250 REM - is the number of streams which are torn; for each
260 REM - of these the box numbers before and after the tear
270 REM - are entered (values of T(J,1) and T(J,2)).
280 REM - LINES 2580 - 2710  The box numbers corresponding
290 REM - to the values entered above are found (line 2620);
300 REM - flows are averaged (line 2640); compositions are
310 REM - averaged (line 2670); this is done for each torn
320 REM - stream.  These new values will be used as inputs
330 REM - to the boxes which the torn streams enter.
340 REM - LINES 2720 - 2820  The new values obtained above
350 REM - are compared with those calculated at the last
360 REM - iteration.  If agreement is within 1% the
370 REM - remainder of the subroutine is bypassed and the
380 REM - number of iterations is printed (line 2990).  If
390 REM - all values are not within 1%, another iteration
400 REM - is carried out.
410 REM - LINES 2830 - 2960  Box numbers are stepped
420 REM - through consecutively, the value of A$(L) being
430 REM - used to identify the nature of each box.  Part B
440 REM - of the appropriate subroutine is called.
450 REM - LINES 2970 - 3000  If the specified agreement
460 REM - has not been met, then the program terminates
470 REM - after 50 iterations.
480 REM ****************************************************
1000 DIM A$(20),C$(10)
1010 DIM B(20,10),F(20),N(20),R(20),X(20,10),Y(20,10)
1020 GOSUB 1290
1030 PRINT "ENTER MIX,SPLIT,SETTLE,OR END"
1040 INPUT B$
1050 IF B$="MIX" THEN 1120
1060 IF B$="SPLIT" THEN 1150
1070 IF B$="SETTLE" THEN 1180
1080 IF B$="SET" THEN 1180
1090 IF B$="END" THEN 1210
1100 PRINT "INPUT NOT RECOGNISED"
1110 GOTO 1030
1120 GOSUB 1590
```

```
1130 GOSUB 1750
1140 GOTO 1030
1150 GOSUB 1950
1160 GOSUB 2080
1170 GOTO 1030
1180 GOSUB 2220
1190 GOSUB 2360
1200 GOTO 1030
1210 PRINT "IF ONE OR MORE STREAMS ARE TORN,"
1220 INPUT "TYPE Y ELSE N";B$
1230 IF B$="Y" THEN 1260
1240 GOSUB 3030
1250 GOTO 3150
1260 GOSUB 2490
1270 GOTO 1240
1280 REM ***************************************************
1290 REM - FEED SUBROUTINE FOR FLOWRATES &
1300 REM - COMPOSITIONS OF ENTERING STREAMS
1310 PRINT "FLOWRATES & COMPOSITIONS FOR STREAMS"
1320 PRINT "ENTERING THE SYSTEM;ENTER THE NUMBER"
1330 PRINT "OF COMPONENTS & STICK TO THIS NUMBER"
1340 PRINT "FOR ALL STREAMS (FOR SETTLING PROBLEMS USE:"
1350 PRINT "K=1 FOR SOLVENT,2 FOR SOLUTE,3 FOR SOLID)"
1360 INPUT N1
1370 FOR K=1 TO N1
1380 PRINT "NAME OF COMPONENT";K;
1390 INPUT C$(K)
1400 NEXT K
1410 PRINT "NUMBER OF STREAMS"
1420 INPUT N2
1430 FOR J=1 TO N2
1440 A$(J)="FEED"
1450 PRINT "FLOW";J;":"
1460 PRINT "FLOWRATE";
1470 INPUT F(J)
1480 IF F(J)=0 THEN 1570
1490 E3=0
1500 FOR K=1 TO N1-1
1510 PRINT "WT. FRACTION OF ";C$(K);
1520 INPUT X(J,K)
1530 E3=E3+X(J,K)
1540 NEXT K
1550 X(J,N1)=1-E3
1560 NEXT J
1570 RETURN
1580 REM ***************************************************
1590 REM - MIX SUBROUTINE, SIMULATES MIXING OF
1600 REM - UP TO 10 STREAMS CONTAINING
1610 REM - UP TO 10 COMPONENTS - MIX, PART A ***************
1620 N2=N2+1
1630 N5=N2
1640 PRINT "THIS IS MIX SUBROUTINE,BOX NUMBER";N2
1650 A$(N2)="MIX"
1660 PRINT "NUMBER OF STREAMS TO BE MIXED";
1670 INPUT N(N5)
1680 FOR J=1 TO N(N5)
1690 PRINT "BOX NUMBER FROM WHICH STREAM COMES";
1700 INPUT B(N5,J)
1710 NEXT J
1720 RETURN
1730 REM ***************************************************
```

```
1740 REM - MIX, PART B ***************************************
1750 E1=0
1760 FOR J=1 TO N(N5)
1770 E1=E1+F(B(N5,J))
1780 NEXT J
1790 FOR K=1 TO N1
1800 E2=0
1810 FOR J=1 TO N(N5)
1820 E2=E2+F(B(N5,J))*X(B(N5,J),K)
1830 NEXT J
1840 Y(N5,K)=E2/E1
1850 NEXT K
1860 F(N5)=E1
1870 FOR K=1 TO N1
1880 X(N5,K)=Y(N5,K)
1890 NEXT K
1900 RETURN
1910 REM *****************************************************
1920 REM - SPLIT SUBROUTINE, SIMULATES SPLITTING OF
1930 REM - ONE STREAM INTO TWO IN A DESIGNATED RATIO
1940 REM - SPLIT,PART A **************************************
1950 N2=N2+2
1960 N5=N2
1970 PRINT "THIS IS SPLIT SUBROUTINE, BOX NUMBER";N2-1
1980 PRINT "THIS NUMBER ALSO DESIGNATES SIDE STREAM.
1990 PRINT "FORWARD STREAM DESIGNATED BOX NUMBER";N2
2000 A$(N2-1)="SPLIT,SIDE STREAM"
2010 A$(N2)="SPLIT,FORWARD STREAM"
2020 PRINT "BOX NUMBER FROM WHICH STREAM COMES";
2030 INPUT B(N5,1)
2040 PRINT "SIDE STREAM/TOTAL FLOW RATIO";
2050 INPUT R(N5)
2060 RETURN
2070 REM - SPLIT,PART B
2080 F(N5-1)=F(B(N5,1))*R(N5)
2090 F(N5)=F(B(N5,1))-F(N5-1)
2100 FOR K=1 TO N1
2110 X(N5-1,K)=X(B(N5,1),K)
2120 X(N5,K)=X(B(N5,1),K)
2130 NEXT K
2140 RETURN
2150 REM *****************************************************
2160 REM - SETTLE SUBROUTINE,SIMULATES OPERATION
2170 REM - OF A SETTLER.   1 ENTERING STREAM
2180 REM - IS ASSUMED, CONSISTING OF 3 COMPONENTS ONLY
2190 REM - THESE ARE NUMBERED 1 FOR SOLVENT,2 FOR SOLUTE,
2200 REM - 3 FOR SOLID.   ZERO SOLIDS IN THE OVERFLOW
2210 REM - IS ASSUMED - SETTLE, PART A ***********************
2220 N2=N2+2
2230 N5=N2
2240 PRINT "THIS IS SETTLE SUBROUTINE,BOX NUMBER";N2-1
2250 PRINT "THIS NUMBER ALSO DESIGNATES OVERFLOW STREAM."
2260 PRINT "UNDERFLOW STREAM IS DESIGNATED BOX NUMBER";N2
2270 A$(N2-1)="SETTLE,OVERFLOW"
2280 A$(N2)="SETTLE,UNDERFLOW"
2290 PRINT "BOX NUMBER FROM WHICH STREAM COMES";
2300 INPUT B(N5,1)
2310 PRINT "SOLIDS/SOLUTION RATIO FOR UNDERFLOW";
2320 INPUT R(N5)
2330 RETURN
2340 REM *****************************************************
```

```
2350 REM - SETTLE, PART B **********************************
2360 F(N5)=F(B(N5,1))*X(B(N5,1),3)*(1+1/R(N5))
2370 F(N5-1)=F(B(N5,1))-F(N5)
2380 X(N5,3)=R(N5)/(1+R(N5))
2390 X2=((X(B(N5,1),1)+X(B(N5,1),2))*(R(N5)+1))
2400 X(N5,2)=X(B(N5,1),2)/X2
2410 X(N5,1)=1-X(N5,2)-X(N5,3)
2420 X(N5-1,3)=0
2430 X(N5-1,2)=X(B(N5,1),2)/(X(B(N5,1),1)+X(B(N5,1),2))
2440 X(N5-1,1)=1-X(N5-1,2)
2450 RETURN
2460 REM ***************************************************
2470 REM - TEAR SUBROUTINE, USED WHEN STREAM DATA
2480 REM - HAS HAD TO BE ESTIMATED
2490 PRINT "THIS IS TEAR SUBROUTINE"
2500 PRINT "HOW MANY TORN STREAMS";
2510 INPUT N4
2520 FOR J=1 TO N4
2530 PRINT "BOX NUMBER WHICH TORN STREAM";J;" LEAVES";
2540 INPUT T(J,1)
2550 PRINT "BOX NUMBER WHICH TORN STREAM";J;" ENTERS";
2560 INPUT T(J,2)
2570 NEXT J
2580 FOR L=1 TO N2
2590 N5=L
2600 FOR M=1 TO N4
2610 FOR N=1 TO N(N5)
2620 IF B(N5,N)=T(M,2) THEN 2640
2630 GOTO 2690
2640 F(B(N5,N))=(F(T(M,1))+F(T(M,2)))/2
2650 C1=C1+1
2660 FOR P=1 TO N1
2670 X(B(N5,N),P)=(X(T(M,1),P)+X(T(M,2),P))/2
2680 NEXT P
2690 NEXT N
2700 NEXT M
2710 NEXT L
2720 FOR M=1 TO N4
2730 FOR P=1 TO N1
2740 IF X(T(M,1),P)=0 THEN 2800
2750 IF X(T(M,2),P)=0 THEN 2800
2760 IF X(T(M,2),P)/X(T(M,1),P)>1.01 THEN 2830
2770 IF X(T(M,2),P)/X(T(M,1),P)<.99 THEN 2830
2780 IF F(T(M,2))/F(T(M,1))>1.01 THEN 2830
2790 IF F(T(M,2))/F(T(M,1))<.99 THEN 2830
2800 NEXT P
2810 NEXT M
2820 GOTO 2990
2830 FOR L=1 TO N2
2840 N5=L
2850 IF A$(L)="FEED" THEN 2960
2860 IF A$(L)="MIX" THEN 2910
2870 IF A$(L)="SPLIT,SIDE STREAM" THEN 2960
2880 IF A$(L)="SPLIT,FORWARD STREAM" THEN 2930
2890 IF A$(L)="SETTLE,OVERFLOW" THEN 2960
2900 IF A$(L)="SETTLE,UNDERFLOW" THEN 2950
2910 GOSUB 1750
2920 GOTO 2960
2930 GOSUB 2070
2940 GOTO 2960
2950 GOSUB 2360
```

```
2960 NEXT L
2970 IF C1>50 THEN 2990
2980 GOTO 2580
2990 PRINT "C1=";C1
3000 RETURN
3010 REM ***************************************************
3020 REM - PRINT SUBROUTINE
3030 PRINT
3040 FOR J=1 TO N2
3050 PRINT A$(J)
3060 PRINT "BOX NUMBER";J
3070 PRINT "FLOWRATE=";F(J)
3080 FOR K=1 TO N1
3090 PRINT "WEIGHT FRACTION OF ";C$(K);"=";
3100 PRINT USING "#.####";X(J,K)
3110 NEXT K
3120 PRINT
3130 NEXT J
3140 RETURN
3150 PRINT "END"
3160 END
```

Example 1.2

A plastics manufacturer sells his product as a moulding compound consisting of a blend of raw polymer, with carbon black, antioxidant and other chemicals. The compound is prepared in a two stage process.

In the first step, all of the carbon black and antioxidant is mixed with polymer to produce a concentrated mix referred to as masterbatch. This operation is carried out in a batch mixer, the polymer being heated above its softening point in the process. The blended product is cooled and diced into granules.

In the second stage, the granular masterbatch and further raw polymer are mixed in a dry blender and then fed to a twin screw compounding extruder. Once again the polymer is heated above its softening point, is cooled and diced into granules.

At each stage of mixing a certain amount of leakage occurs. Some of this material is contaminated and must be discarded; the remainder is recycled to the first stage mixer. The flow diagram of the process is shown in simplified form in Figure 1.6.

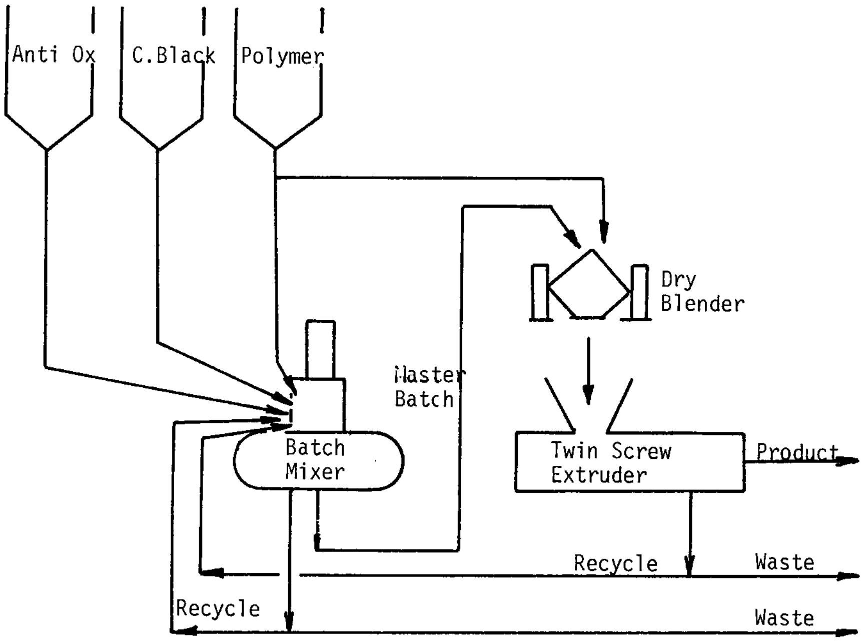


Figure 1.6 Polymer Compounding

Using the data below, use the program described above to calculate the flowrates and compositions of all streams.

Data

1.	Raw polymer to 1st stage mixer:	70 kg
2.	Raw polymer to 2nd stage mixer:	950 kg
3.	Carbon black:	30 kg
4.	Anti oxidant:	2 kg

5% of the 1st stage product appears as leakage; 25% of this has to be discarded, the remainder being recycled.

3% of the 2nd stage product appears as leakage; 30% of this has to be discarded, the remainder being recycled.

The first step in solving this problem is to draw a block diagram. Here the individual steps 'mix' and 'split' are shown, boxes and streams being numbered in accordance with the convention employed in the program (see Figure 1.7).

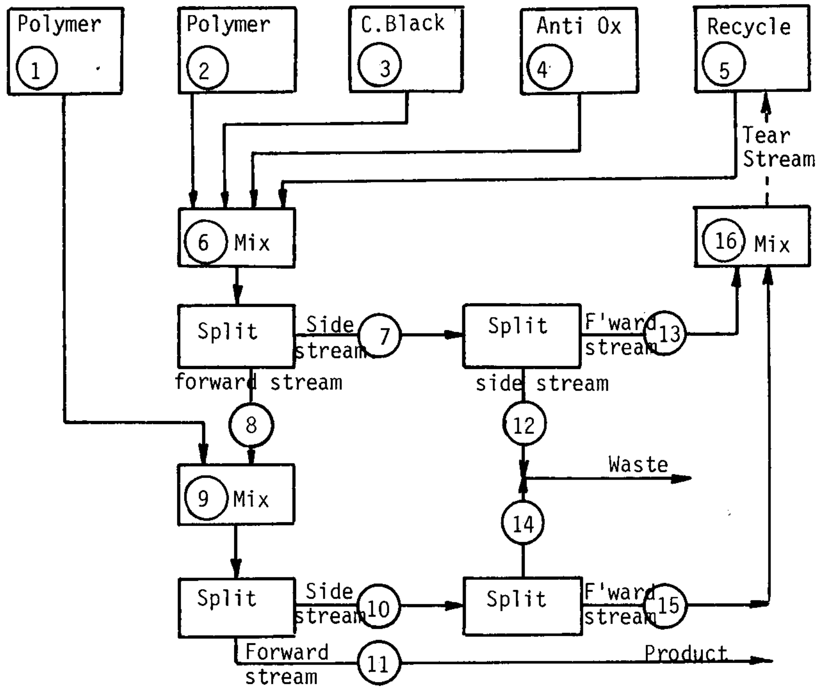


Figure 1.7 Block diagram for Polymer Compounding Process

Next the program FSHT3 is run and the values entered, as follows:

```
RUN
FLOWRATES & COMPOSITIONS FOR STREAMS
ENTERING THE SYSTEM;ENTER THE NUMBER
OF COMPONENTS & STICK TO THIS NUMBER
FOR ALL STREAMS (FOR SETTLING PROBLEMS USE:
K=1 FOR SOLVENT,2 FOR SOLUTE,3 FOR SOLID)
? 3
NAME OF COMPONENT 1 ? C BLACK
NAME OF COMPONENT 2 ? ANTI OX
NAME OF COMPONENT 3 ? POLYMER
NUMBER OF STREAMS
? 5
FLOW 1 :
FLOWRATE? 950
WT. FRACTION OF C BLACK? 0
WT. FRACTION OF ANTI OX? 0
FLOW 2 :
FLOWRATE? 70
WT. FRACTION OF C BLACK? 0
WT. FRACTION OF ANTI OX? 0
FLOW 3 :
FLOWRATE? 30
WT. FRACTION OF C BLACK? 1
WT. FRACTION OF ANTI OX? 0
FLOW 4 :
FLOWRATE? 2
WT. FRACTION OF C BLACK? 0
WT. FRACTION OF ANTI OX? 1
FLOW 5 :
FLOWRATE? 40
WT. FRACTION OF C BLACK? .03
WT. FRACTION OF ANTI OX? .002
```

```
ENTER MIX,SPLIT,SETTLE,OR END
? MIX
THIS IS MIX SUBROUTINE,BOX NUMBER 6
NUMBER OF STREAMS TO BE MIXED? 4
BOX NUMBER FROM WHICH STREAM COMES? 2
BOX NUMBER FROM WHICH STREAM COMES? 3
BOX NUMBER FROM WHICH STREAM COMES? 4
BOX NUMBER FROM WHICH STREAM COMES? 5
ENTER MIX,SPLIT,SETTLE,OR END
? SPLIT
THIS IS SPLIT SUBROUTINE, BOX NUMBER 7
THIS NUMBER ALSO DESIGNATES SIDE STREAM.
FORWARD STREAM DESIGNATED BOX NUMBER 8
BOX NUMBER FROM WHICH STREAM COMES? 6
SIDE STREAM/TOTAL FLOW RATIO? .05
ENTER MIX,SPLIT,SETTLE,OR END
? MIX
THIS IS MIX SUBROUTINE,BOX NUMBER 9
NUMBER OF STREAMS TO BE MIXED? 2
BOX NUMBER FROM WHICH STREAM COMES? 1
BOX NUMBER FROM WHICH STREAM COMES? 8
ENTER MIX,SPLIT,SETTLE,OR END
? SPLIT
THIS IS SPLIT SUBROUTINE, BOX NUMBER 10
THIS NUMBER ALSO DESIGNATES SIDE STREAM.
FORWARD STREAM DESIGNATED BOX NUMBER 11
BOX NUMBER FROM WHICH STREAM COMES? 9
SIDE STREAM/TOTAL FLOW RATIO? .03
ENTER MIX,SPLIT,SETTLE,OR END
? SPLIT
THIS IS SPLIT SUBROUTINE, BOX NUMBER 12
THIS NUMBER ALSO DESIGNATES SIDE STREAM.
FORWARD STREAM DESIGNATED BOX NUMBER 13
BOX NUMBER FROM WHICH STREAM COMES? 7
SIDE STREAM/TOTAL FLOW RATIO? .25
ENTER MIX,SPLIT,SETTLE,OR END
? SPLIT
THIS IS SPLIT SUBROUTINE, BOX NUMBER 14
THIS NUMBER ALSO DESIGNATES SIDE STREAM.
FORWARD STREAM DESIGNATED BOX NUMBER 15
BOX NUMBER FROM WHICH STREAM COMES? 10
SIDE STREAM/TOTAL FLOW RATIO? .3
ENTER MIX,SPLIT,SETTLE,OR END
? MIX
THIS IS MIX SUBROUTINE,BOX NUMBER 16
NUMBER OF STREAMS TO BE MIXED? 2
BOX NUMBER FROM WHICH STREAM COMES? 13
BOX NUMBER FROM WHICH STREAM COMES? 15
ENTER MIX,SPLIT,SETTLE,OR END
? END
IF ONE OR MORE STREAMS ARE TORN,
TYPE Y ELSE N? Y
THIS IS TEAR SUBROUTINE
HOW MANY TORN STREAMS? 1
BOX NUMBER WHICH TORN STREAM 1  LEAVES? 16
BOX NUMBER WHICH TORN STREAM 1  ENTERS? 5
C1= 7
```

```
FEED
BOX NUMBER 1
FLOWRATE= 950
WEIGHT FRACTION OF C BLACK=0.0000
WEIGHT FRACTION OF ANTI OX=0.0000
WEIGHT FRACTION OF POLYMER=1.0000

FEED
BOX NUMBER 2
FLOWRATE= 70
WEIGHT FRACTION OF C BLACK=0.0000
WEIGHT FRACTION OF ANTI OX=0.0000
WEIGHT FRACTION OF POLYMER=1.0000

FEED
BOX NUMBER 3
FLOWRATE= 30
WEIGHT FRACTION OF C BLACK=1.0000
WEIGHT FRACTION OF ANTI OX=0.0000
WEIGHT FRACTION OF POLYMER=0.0000

FEED
BOX NUMBER 4
FLOWRATE= 2
WEIGHT FRACTION OF C BLACK=0.0000
WEIGHT FRACTION OF ANTI OX=1.0000
WEIGHT FRACTION OF POLYMER=0.0000

                                 RECYCLE
FEED                             (after tear)
BOX NUMBER 5
FLOWRATE= 27.52879
WEIGHT FRACTION OF C BLACK=0.0663
WEIGHT FRACTION OF ANTI OX=0.0044
WEIGHT FRACTION OF POLYMER=0.9292

MIX                              MASTERBATCH
BOX NUMBER 6
FLOWRATE= 129.6587
WEIGHT FRACTION OF C BLACK=0.2454
WEIGHT FRACTION OF ANTI OX=0.0164
WEIGHT FRACTION OF POLYMER=0.7382

SPLIT,SIDE STREAM
BOX NUMBER 7
FLOWRATE= 6.482935
WEIGHT FRACTION OF C BLACK=0.2454
WEIGHT FRACTION OF ANTI OX=0.0164
WEIGHT FRACTION OF POLYMER=0.7382

SPLIT,FORWARD STREAM
BOX NUMBER 8
FLOWRATE= 123.1758
WEIGHT FRACTION OF C BLACK=0.2454
WEIGHT FRACTION OF ANTI OX=0.0164
WEIGHT FRACTION OF POLYMER=0.7382
```

```
MIX
BOX NUMBER 9
FLOWRATE= 1073.176
WEIGHT FRACTION OF C BLACK=0.0282
WEIGHT FRACTION OF ANTI OX=0.0019
WEIGHT FRACTION OF POLYMER=0.9700

SPLIT,SIDE STREAM
BOX NUMBER 10
FLOWRATE= 32.19528
WEIGHT FRACTION OF C BLACK=0.0282
WEIGHT FRACTION OF ANTI OX=0.0019
WEIGHT FRACTION OF POLYMER=0.9700

SPLIT,FORWARD STREAM   PRODUCT
BOX NUMBER 11
FLOWRATE= 1040.981
WEIGHT FRACTION OF C BLACK=0.0282
WEIGHT FRACTION OF ANTI OX=0.0019
WEIGHT FRACTION OF POLYMER=0.9700

SPLIT,SIDE STREAM        WASTE
BOX NUMBER 12
FLOWRATE= 1.620734
WEIGHT FRACTION OF C BLACK=0.2454
WEIGHT FRACTION OF ANTI OX=0.0164
WEIGHT FRACTION OF POLYMER=0.7382

SPLIT,FORWARD STREAM
BOX NUMBER 13
FLOWRATE= 4.862201
WEIGHT FRACTION OF C BLACK=0.2454
WEIGHT FRACTION OF ANTI OX=0.0164
WEIGHT FRACTION OF POLYMER=0.7382

SPLIT,SIDE STREAM        WASTE
BOX NUMBER 14
FLOWRATE= 9.658582
WEIGHT FRACTION OF C BLACK=0.0282
WEIGHT FRACTION OF ANTI OX=0.0019
WEIGHT FRACTION OF POLYMER=0.9700

SPLIT,FORWARD STREAM
BOX NUMBER 15
FLOWRATE= 22.53669
WEIGHT FRACTION OF C BLACK=0.0282
WEIGHT FRACTION OF ANTI OX=0.0019
WEIGHT FRACTION OF POLYMER=0.9700

                        RECYCLE
MIX
BOX NUMBER 16           (before tear)
FLOWRATE= 27.39889
WEIGHT FRACTION OF C BLACK=0.0667
WEIGHT FRACTION OF ANTI OX=0.0044
WEIGHT FRACTION OF POLYMER=0.9288

END
Ok
```

STEP FOUR

A further step in the evolution of the program is the incorporation of subroutines which will permit flowsheet modification, and also changing of the values of the process variables. To do this, three additional subroutines have been written.

These were incorporated into the program and debugged one at a time. For the sake of brevity however, all three are described below, and the program incorporating them all is given.

Process Subroutine (BUILD)

On inspection of the program so far, but having these new developments in mind, it was obvious that subroutine 'Feed' should not have been treated differently from the process subroutines 'Mix', 'Split', etc. At the same time it was decided to simplify the control program by making the segment in which the flowsheet building steps occur, into a new subroutine called 'Build'; this also made easier the writing of subroutine 'Modify'.

Process Subroutine (CHANGE)

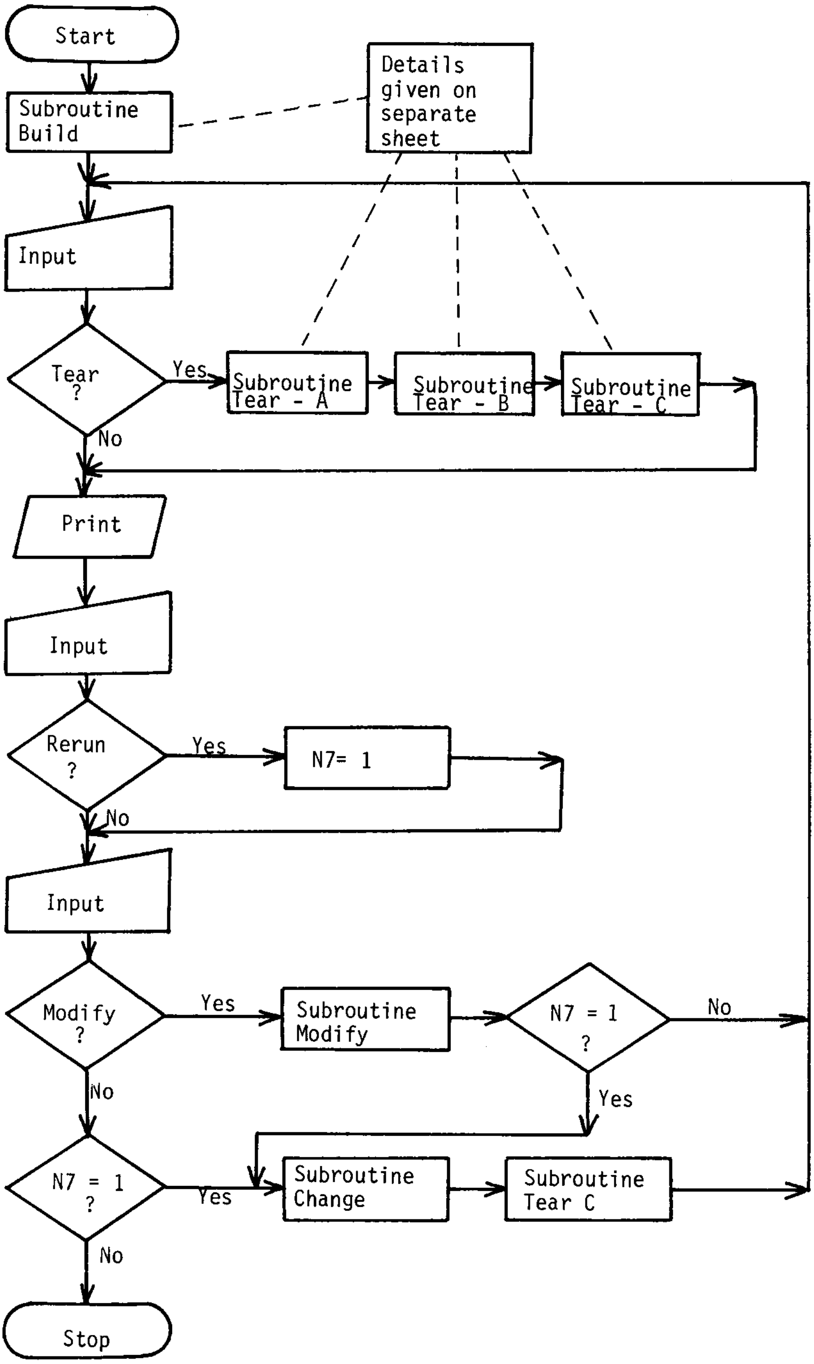
The purpose of this subroutine is to change the values of process variables on the flowsheet existing at the time the subroutine is called. The values involved are flowrates, compositions, splitter ratios and solids/solution ratios.

Process Subroutine (MODIFY)

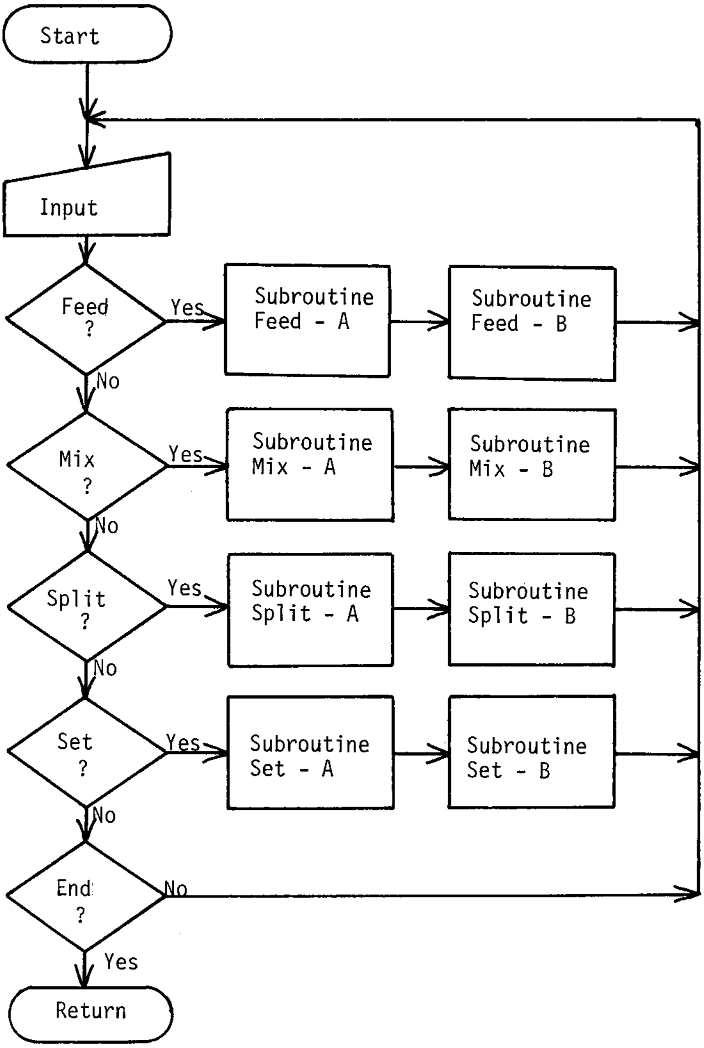
This subroutine is used to modify the flowsheet existing at the time the subroutine is called. It is in two parts. In the first part the functions of existing boxes and the interconnections between them, are altered; this is done by calling up the process subroutines as required. In the second part, control is transferred to 'Build' subroutine, which carries on with the assembly of the flowsheet from the latest value of N2.

Program FSHT4 which follows includes these further steps.

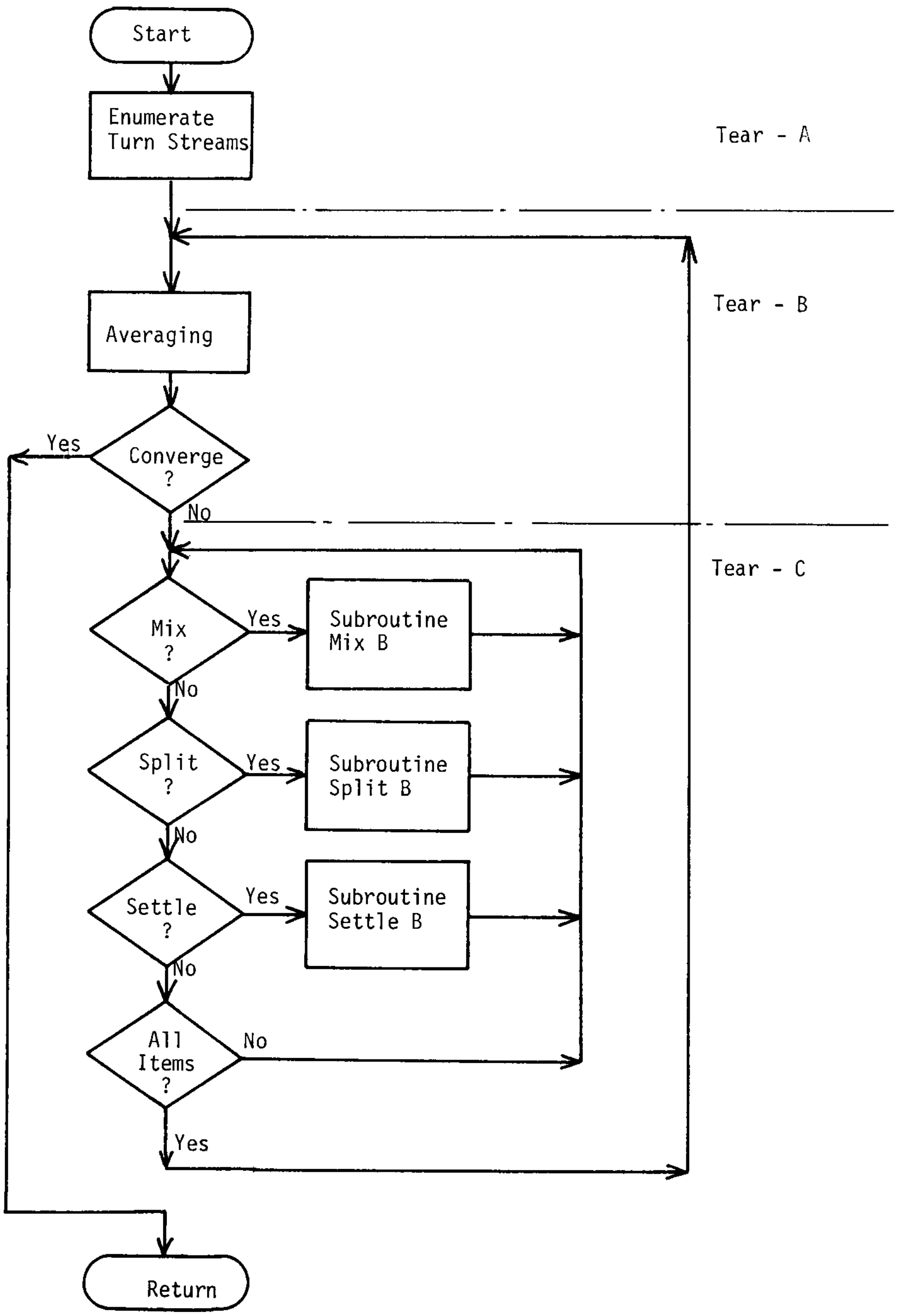
FSHT4.BAS


Start
Subroutine Build
Details given on separate sheet
Input
Tear ?
Yes
No
Subroutine Tear - A
Subroutine Tear - B
Subroutine Tear - C
Print
Input
Rerun ?
Yes
No
N7= 1
Input
Modify ?
Yes
No
Subroutine Modify
N7 = 1 ?
No
Yes
N7 = 1 ?
Yes
No
Subroutine Change
Subroutine Tear C
Stop

Subroutine Build
Used in FSHT4.BAS



Subroutine TEAR
Used in FSHT4. BAS



```
10  REM ****************************************************
20  REM - PROGRAM FSHT4.BAS LAST STEP IN
30  REM - DEVELOPMENT OF A FLOWSHEET PROGRAM
40  REM - PROGRAM NOMENCLATURE  The following additional
50  REM - variables are used:
60  REM - B1   -   Number of entering streams, splitters,
70  REM            settlers, or number of boxes,to be
80  REM            changed.
90  REM - B2   -   Box number which has to be changed or
100 REM            modified.
110 REM - N6   -   Duplicate of N2, used in "Modify"
120 REM            subroutine
130 REM - N7   -   This variable has two values; value 0
140 REM            indicates no changes to process
150 REM            variables; value 1 indicates changes
160 REM - DETAILS OF SUBROUTINE BUILD
170 REM - LINES 1430-1680   The required functions ("Feed",
180 REM - "Mix",etc) input from the keyboard,(lines 1430,
190 REM - 1440); the appropriate subroutines are then
200 REM - called (lines 1450-1670).
205 REM - DETAILS OF SUBROUTINE CHANGE
210 REM - LINES 3500-3710   The number of existing feed
220 REM - streams whose values will be changed is entered
230 REM - from the keyboard; for each of these streams the
240 REM - designer then enters the box number and values of
250 REM - flowrate and composition.
260 REM - LINES 3720-3850   New values of ratios for
270 REM - splitters are similarly entered.  Note that lines
280 REM - 3780 and 3790 correct the box number should the
290 REM - designer have entered that for the side stream,
300 REM - in error.
310 REM - LINES 3860-4000   In a similar manner, new values
320 REM - of ratios for settlers are entered.
330 REM - DETAILS OF SUBROUTINE MODIFY
340 REM - LINES 4020-4260   The number of existing boxes
350 REM - whose function will be changed is entered from the
360 REM - keyboard,and for each of these the designer then
370 REM - enters the box number and the required function
380 REM - ("Feed","Mix",etc - lines 4040 to 4110).  The
390 REM - appropriate subroutines are then called (lines
400 REM - 4120 to 4260).
410 REM - LINES 4270-4330   If new boxes are to be added to
420 REM - the flowsheet then the subroutine "Build"is called
430 REM ****************************************************
1000 DIM A$(20),C$(10)
1010 DIM B(20,10),X(20,10),Y(20,10)
1020 DIM F(20),N(20),R(20)
1030 GOSUB 1390
1040 N6=N2
1050 IF N4>0 THEN 1070
1060 GOTO 1090
1070 GOSUB 3020
1080 GOTO 1130
1090 PRINT
1100 PRINT "IF ONE OR MORE STREAMS ARE TORN,";
1110 INPUT "TYPE Y ELSE N";B$
1120 IF B$="Y" THEN 1150
1130 GOSUB 4350
1140 GOTO 1180
1150 GOSUB 2890
1160 GOSUB 3020
```

```
1170 GOTO 1130
1180 N7=0
1190 PRINT
1200 PRINT "TO RERUN WITH DIFFERENT VALUES ";
1210 INPUT "TYPE Y ELSE N";B$
1220 IF B$="Y" THEN 1240
1230 GOTO 1250
1240 N7=1
1250 PRINT
1260 INPUT "TO MODIFY FLOWSHEET TYPE Y, ELSE N";B$
1270 IF B$="Y" THEN 1300
1280 IF N7=1 THEN 1360
1290 GOTO 4540
1300 GOSUB 4020
1310 N4=0
1320 IF N7<>1 THEN 1350
1330 GOSUB 3500
1340 GOSUB 3350
1350 GOTO 1040
1360 GOSUB 3500
1370 GOSUB 3350
1380 GOTO 1050
1390 PRINT
1400 REM ****************************************************
1410 REM - BUILD SUBROUTINE
1420 REM - USED WHEN ASSEMBLING THE FLOWSHEET
1430 PRINT "ENTER FEED,MIX,SPLIT,SETTLE,OR END";
1440 INPUT B$
1450 IF B$="FEED" THEN 1530
1460 IF B$="MIX" THEN 1590
1470 IF B$="SPLIT" THEN 1620
1480 IF B$="SETTLE" THEN 1650
1490 IF B$="SET" THEN 1650
1500 IF B$="END" THEN 1680
1510 PRINT "INPUT NOT RECOGNISED"
1520 GOTO 1390
1530 IF N1>0 THEN 1570
1540 GOSUB 1700
1550 GOSUB 1860
1560 GOTO 1390
1570 GOSUB 1860
1580 GOTO 1390
1590 GOSUB 2020
1600 GOSUB 2160
1610 GOTO 1390
1620 GOSUB 2340
1630 GOSUB 2490
1640 GOTO 1390
1650 GOSUB 2580
1660 GOSUB 2770
1670 GOTO 1390
1680 RETURN
1690 REM ****************************************************
1700 REM - FEED SUBROUTINE FOR FLOWRATES &
1710 REM - COMPOSITIONS OF ENTERING STREAMS
1720 REM - FEED, PART A *************************************
1730 PRINT "FLOWRATES & COMPOSITIONS FOR STREAMS"
1740 PRINT "ENTERING THE SYSTEM;ENTER THE NUMBER"
1750 PRINT "OF COMPONENTS & STICK TO THIS NUMBER"
1760 PRINT "FOR ALL STREAMS (FOR SETTLING PROBLEMS USE:"
1770 PRINT "K=1 FOR SOLVENT,2 FOR SOLUTE,3 FOR SOLID)"
```

```
1780 INPUT N1
1790 FOR K=1 TO N1
1800 PRINT "NAME OF COMPONENT";K;
1810 INPUT C$(K)
1820 NEXT K
1830 RETURN
1840 REM *****************************************************
1850 REM - FEED, PART B *************************************
1860 N2=N2+1
1870 N5=N2
1880 PRINT "THIS IS FEED SUBROUTINE, BOX NUMBER";N2
1890 A$(N2)="FEED"
1900 PRINT "FLOWRATE";
1910 INPUT F(N2)
1920 IF F(N2)=0 THEN 2000
1930 E3=0
1940 FOR K=1 TO N1-1
1950 PRINT "WT FRACTION OF ";C$(K);
1960 INPUT X(N2,K)
1970 E3=E3+X(N2,K)
1980 NEXT K
1990 X(N2,N1)=1-E3
2000 RETURN
2010 REM *****************************************************
2020 REM - MIX SUBROUTINE, SIMULATES MIXING OF
2030 REM - UP TO 10 STREAMS CONTAINING
2040 REM - UP TO 10 COMPONENTS - MIX, PART A ***************
2050 N2=N2+1
2060 N5=N2
2070 PRINT "THIS IS MIX SUBROUTINE,BOX NUMBER";N2
2080 A$(N2)="MIX"
2090 PRINT "NUMBER OF STREAMS TO BE MIXED";
2100 INPUT N(N5)
2110 FOR J=1 TO N(N5)
2120 PRINT "BOX NUMBER FROM WHICH STREAM COMES";
2130 INPUT B(N5,J)
2140 NEXT J
2150 RETURN
2160 REM - MIX, PART B **************************************
2170 E1=0
2180 FOR J=1 TO N(N5)
2190 E1=E1+F(B(N5,J))
2200 NEXT J
2210 FOR K=1 TO N1
2220 E2=0
2230 FOR J=1 TO N(N5)
2240 E2=E2+F(B(N5,J))*X(B(N5,J),K)
2250 NEXT J
2260 Y(N5,K)=E2/E1
2270 NEXT K
2280 F(N5)=E1
2290 FOR K=1 TO N1
2300 X(N5,K)=Y(N5,K)
2310 NEXT K
2320 RETURN
2330 REM *****************************************************
2340 REM - SPLIT SUBROUTINE, SIMULATES SPLITTING OF
2350 REM - ONE STREAM INTO TWO IN A DESIGNATED RATIO
2360 REM - SPLIT,PART A *************************************
2370 N2=N2+2
2380 N5=N2
```

```
2390 PRINT "THIS IS SPLIT SUBROUTINE, BOX NUMBER";N2-1
2400 PRINT "THIS NUMBER ALSO DESIGNATES SIDE STREAM."
2410 PRINT "FORWARD STREAM DESIGNATED BOX NUMBER";N2
2420 A$(N2-1)="SPLIT,SIDE STREAM"
2430 A$(N2)="SPLIT,FORWARD STREAM"
2440 PRINT "BOX NUMBER FROM WHICH STREAM COMES";
2450 INPUT B(N5,1)
2460 PRINT "SIDE STREAM/TOTAL FLOW RATIO";
2470 INPUT R(N5)
2480 RETURN
2490 REM - SPLIT,PART B ************************************
2500 F(N5-1)=F(B(N5,1))*R(N5)
2510 F(N5)=F(B(N5,1))-F(N5-1)
2520 FOR K=1 TO N1
2530 X(N5-1,K)=X(B(N5,1),K)
2540 X(N5,K)=X(B(N5,1),K)
2550 NEXT K
2560 RETURN
2570 REM ***************************************************
2580 REM - SETTLE SUBROUTINE,SIMULATES OPERATION
2590 REM - OF A SETTLER.   1 ENTERING FLOW
2600 REM - IS ASSUMED, CONSISTING OF 3 COMPONENTS ONLY
2610 REM - THESE ARE NUMBERED 1 FOR SOLVENT, 2 FOR SOLUTE,
2620 REM - 3 FOR SOLID.   ZERO SOLIDS IN THE OVERFLOW
2630 REM - IS ASSUMED - SETTLE, PART A *********************
2640 N2=N2+2
2650 N5=N2
2660 PRINT "THIS IS SETTLE SUBROUTINE,BOX NUMBER";N2-1
2670 PRINT "THIS NUMBER ALSO DESIGNATES OVERFLOW STREAM."
2680 PRINT "UNDERFLOW STREAM IS DESIGNATED BOX NUMBER";N2
2690 A$(N2-1)="SETTLE,OVERFLOW"
2700 A$(N2)="SETTLE,UNDERFLOW"
2710 PRINT "BOX NUMBER FROM WHICH FLOW COMES";
2720 INPUT B(N5,1)
2730 PRINT "SOLIDS/SOLUTION RATIO FOR UNDERFLOW";
2740 INPUT R(N5)
2750 RETURN
2760 REM ***************************************************
2770 REM - SETTLE, PART B **********************************
2780 F(N5)=F(B(N5,1))*X(B(N5,1),3)*(1+1/R(N5))
2790 F(N5-1)=F(B(N5,1))-F(N5)
2800 X(N5,3)=R(N5)/(1+R(N5))
2810 X2=((X(B(N5,1),1)+X(B(N5,1),2))*(R(N5)+1))
2820 X(N5,2)=X(B(N5,1),2)/X2
2830 X(N5,1)=1-X(N5,2)-X(N5,3)
2840 X(N5-1,3)=0
2850 X(N5-1,2)=X(B(N5,1),2)/(X(B(N5,1),1)+X(B(N5,1),2))
2860 X(N5-1,1)=1-X(N5-1,2)
2870 RETURN
2880 REM ***************************************************
2890 REM - TEAR SUBROUTINE, USED WHEN STREAM DATA
2900 REM - HAS HAD TO BE ESTIMATED - TEAR, PART A **********
2910 PRINT "THIS IS TEAR SUBROUTINE"
2920 PRINT "HOW MANY TORN STREAMS";
2930 INPUT N4
2940 FOR J=1 TO N4
2950 PRINT "BOX NUMBER WHICH TORN STREAM";J;" LEAVES";
2960 INPUT T(J,1)
2970 PRINT "BOX NUMBER WHICH TORN STREAM";J;" ENTERS";
2980 INPUT T(J,2)
2990 NEXT J
```

```
3000 RETURN
3010 REM *****************************************************
3020 REM - TEAR, PART B **************************************
3030 C1=0
3040 FOR L=1 TO N2
3050 N5=L
3060 FOR M=1 TO N4
3070 FOR N=1 TO N(N5)
3080 IF B(N5,N)=T(M,2) THEN 3100
3090 GOTO 3150
3100 F(B(N5,N))=(F(T(M,1))+F(T(M,2)))/2
3110 C1=C1+1
3120 FOR P=1 TO N1
3130 X(B(N5,N),P)=(X(T(M,1),P)+X(T(M,2),P))/2
3140 NEXT P
3150 NEXT N
3160 NEXT M
3170 NEXT L
3180 FOR M=1 TO N4
3190 FOR P=1 TO N1
3200 IF X(T(M,1),P)=0 THEN 3260
3210 IF X(T(M,2),P)=0 THEN 3260
3220 IF X(T(M,2),P)/X(T(M,1),P)>1.01 THEN 3290
3230 IF X(T(M,2),P)/X(T(M,1),P)<.99 THEN 3290
3240 IF F(T(M,2))/F(T(M,1))>1.01 THEN 3290
3250 IF F(T(M,2))/F(T(M,1))<.99 THEN 3290
3260 NEXT P
3270 NEXT M
3280 GOTO 3320
3290 GOSUB 3360
3300 IF C1>50 THEN 3320
3310 GOTO 3040
3320 PRINT "C1=";C1
3330 RETURN
3340 REM *****************************************************
3350 REM - TEAR, PART C **************************************
3360 FOR L=1 TO N2
3370 N5=L
3380 IF A$(L)="MIX" THEN 3420
3390 IF A$(L)="SPLIT,FORWARD STREAM" THEN 3440
3400 IF A$(L)="SETTLE,UNDERFLOW" THEN 3460
3410 GOTO 3470
3420 GOSUB 2170
3430 GOTO 3470
3440 GOSUB 2490
3450 GOTO 3470
3460 GOSUB 2780
3470 NEXT L
3480 RETURN
3490 REM *****************************************************
3500 REM - CHANGE SUBROUTINE, USED WHEN INPUT STREAMS,
3510 REM - SPLITTING OR SETTLING RATIOS ARE TO BE CHANGED
3520 PRINT "THIS IS CHANGE SUBROUTINE.  INPUT NUMBER OF"
3530 PRINT "ENTERING STREAMS TO BE CHANGED";
3540 INPUT B1
3550 IF B1=0 THEN 3720
3560 FOR J=1 TO B1
3570 PRINT "BOX NUMBER OF ENTERING STREAM";
3580 INPUT B2
3590 IF A$(B2)="FEED" THEN 3620
3600 PRINT "INCORRECT BOX NUMBER HAS BEEN ENTERED"
```

```
3610 GOTO 3570
3620 PRINT "NEW FLOWRATE";
3630 INPUT F(B2)
3640 E3=0
3650 FOR K=1 TO N1-1
3660 PRINT "NEW WEIGHT FRACTION OF ";C$(K);
3670 INPUT X(B2,K)
3680 E3=E3+X(B2,K)
3690 NEXT K
3700 X(B2,N1)=1-E3
3710 NEXT J
3720 PRINT "NUMBER OF SPLITTERS TO BE CHANGED";
3730 INPUT B1
3740 IF B1=0 THEN 3860
3750 FOR J=1 TO B1
3760 PRINT "BOX NUMBER OF SPLITTER";
3770 INPUT B2
3780 IF A$(B2)="SPLIT,SIDE STREAM" THEN 3820
3790 IF A$(B2)="SPLIT,FORWARD STREAM" THEN 3830
3800 PRINT "INCORRECT BOX NUMBER HAS BEEN ENTERED"
3810 GOTO 3760
3820 B2=B2+1
3830 PRINT "NEW VALUE OF SIDESTREAM/TOTAL FLOW RATIO";
3840 INPUT R(B2)
3850 NEXT J
3860 PRINT "NUMBER OF SETTLERS TO BE CHANGED";
3870 INPUT B1
3880 IF B1=0 THEN 4000
3890 FOR J=1 TO B1
3900 PRINT "BOX NUMBER OF SETTLER";
3910 INPUT B2
3920 IF A$(B2)="SETTLE,OVERFLOW" THEN 3960
3930 IF A$(B2)="SETTLE,UNDERFLOW" THEN 3970
3940 PRINT "INCORRECT BOX NUMBER HAS BEEN ENTERED"
3950 GOTO 3900
3960 B2=B2+1
3970 PRINT "NEW VALUE OF SOLIDS/SOLUTION RATIO";
3980 INPUT R(B2)
3990 NEXT J
4000 RETURN
4010 REM ****************************************************
4020 REM - MODIFY SUBROUTINE, USED WHEN
4030 REM - MODIFYING THE FLOWSHEET
4040 PRINT "THIS IS MOD SUBROUTINE.   HOW MANY"
4050 INPUT "EXISTING BOXES TO BE ALTERED";B1
4060 FOR J=1 TO B1
4070 PRINT "BOX NUMBER";
4080 INPUT B2
4090 N2=B2-1
4100 PRINT "FEED,MIX,SPLIT,OR SETTLE";
4110 INPUT B$
4120 IF B$="FEED" THEN 4190
4130 IF B$="MIX" THEN 4210"
4140 IF B$="SPLIT" THEN 4230
4150 IF B$="SET" THEN 4250"
4160 IF B$="SETTLE" THEN 4250
4170 PRINT "INPUT NOT RECOGNISED"
4180 GOTO 4100
4190 GOSUB 1860
4200 GOTO 4260
4210 GOSUB 2020
```

```
4220 GOTO 4260
4230 GOSUB 2490
4240 GOTO 4260
4250 GOSUB 2580
4260 NEXT J
4270 PRINT "IF NEW BOXES TO BE ADDED INPUT Y ELSE N";
4280 INPUT B$
4290 IF B$="Y" THEN 4310
4300 GOTO 4330
4310 N2=N6
4320 GOSUB 1390
4330 RETURN
4340 REM ***************************************************
4350 REM - PRINT SUBROUTINE
4360 PRINT
4370 FOR J=1 TO N2
4380 PRINT A$(J)
4390 PRINT "BOX NUMBER";J
4400 IF A$(J)="SETTLE,UNDERFLOW" THEN 4430
4410 IF A$(J)="SPLIT,SIDE STREAM" THEN 4450
4420 GOTO 4460
4430 PRINT "SOLIDS/SOLUTION RATIO=";R(J)
4440 GOTO 4460
4450 PRINT "SIDE STREAM/TOTAL FLOW RATIO=";R(J)
4460 PRINT "FLOWRATE=";F(J)
4470 FOR K=1 TO N1
4480 PRINT "WEIGHT FRACTION OF ";C$(K);"=";
4490 PRINT USING "#.####";X(J,K)
4500 NEXT K
4510 PRINT
4520 NEXT J
4530 RETURN
4540 PRINT "END"
4550 END
```

Example 1.3

(a) A sand intended for concrete manufacture is washed to reduce its salt content in a 2-stage, cross-current operation as shown in Figure 1.8.

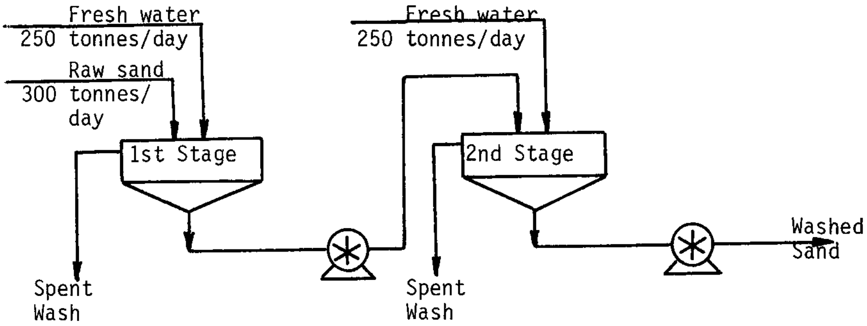


Figure 1.8. Two-stage cross-current sand washing operation.

300 tonnes per day of sand are washed in this way, using 250 tonnes per day of water at each stage. The composition of the sand is given below. The underflow leaving each stage is a slurry consisting of 66.6 wt% sand, the remainder being dilute salt solution.

Calculate the flowrates and compositions of all streams and determine the salt content of the washed sand on a dry basis.

(b) An improved slurry pump is available capable of handling a slurry containing 70 wt% sand. It has also been observed that by careful operation the water flowrate to a stage can be increased to 275 tonnes per day without carryover of sand.

It is proposed to replace the existing pumps with the improved model, and simultaneously to convert to countercurrent operation as shown in Figure 1.9.

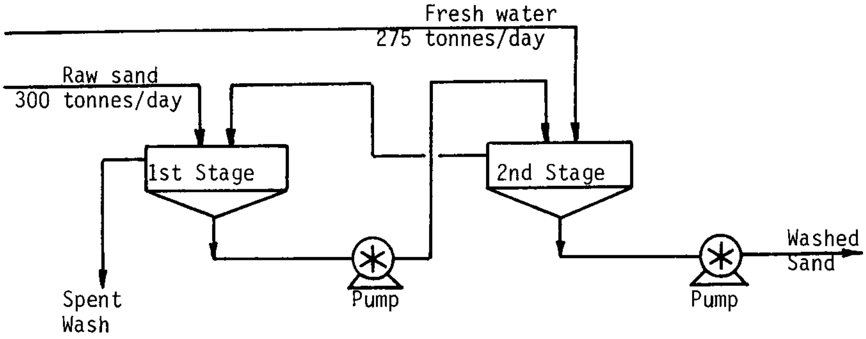


Figure 1.9 Two-stage counter-current sand washing operation.

In this way water usage will be reduced to 275 tonnes per day total. Can the same purity be attained for the washed sand?

Composition of Raw Sand	
Water	5.0%
Salt	1.5%
Sand	93.5%

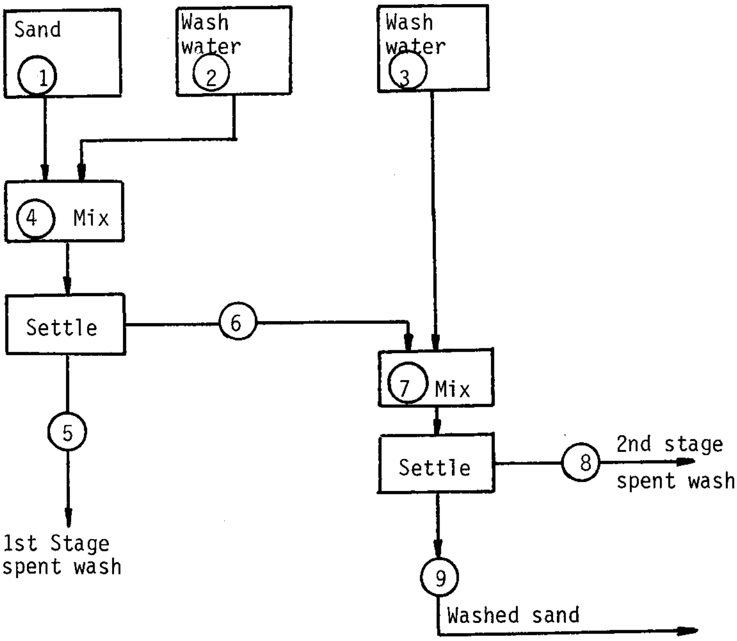


Figure 1.10 Block diagram for 2-stage cross-current washing operation.

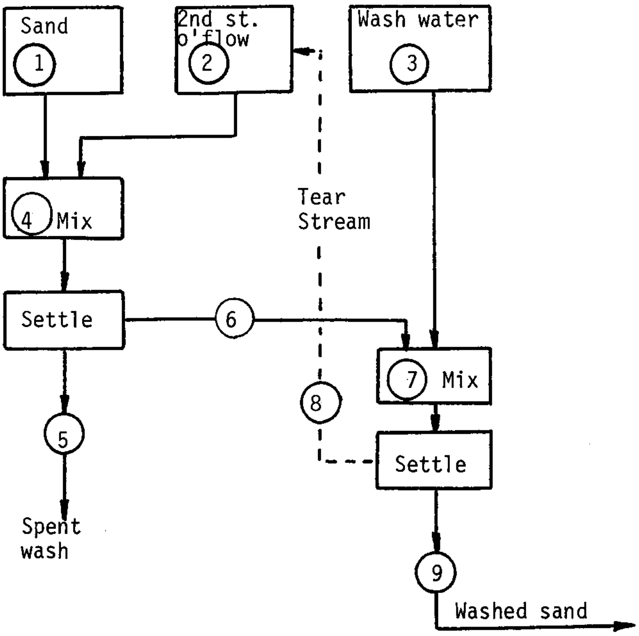


Figure 1.11 Block diagram for 2-stage counter-current washing operation.

The question gives the slurry composition on a percentage basis; before running the program it is necessary to convert the values given to a ratio basis.

Underflow composition: 66.6 wt% sand
33.4 wt% solution

Ratio sand/solution = $\frac{66.6}{33.4} \simeq 2.0$

Underflow composition: 70.0 wt% sand
30.0 wt% solution

Ratio sand/solution = $\frac{70}{30} = 2.33$

Next block diagrams are drawn for the two cases (Figures 1.10 & 1.11). The program FSHT4 is then run and the values entered, as follows:

```
ENTER FEED,MIX,SPLIT,SETTLE,OR END? FEED
FLOWRATES & COMPOSITIONS FOR STREAMS
ENTERING THE SYSTEM;ENTER THE NUMBER
OF COMPONENTS & STICK TO THIS NUMBER
FOR ALL STREAMS (FOR SETTLING PROBLEMS USE:
K=1 FOR SOLVENT,2 FOR SOLUTE,3 FOR SOLID)
? 3
NAME OF COMPONENT 1 ? WATER
NAME OF COMPONENT 2 ? SALT
NAME OF COMPONENT 3 ? SAND
THIS IS FEED SUBROUTINE, BOX NUMBER 1
FLOWRATE? 300
WT FRACTION OF WATER? .05
WT FRACTION OF SALT? .015

ENTER FEED,MIX,SPLIT,SETTLE,OR END? FEED
THIS IS FEED SUBROUTINE, BOX NUMBER 2
FLOWRATE? 250
WT FRACTION OF WATER? 1
WT FRACTION OF SALT? 0

ENTER FEED,MIX,SPLIT,SETTLE,OR END? FEED
THIS IS FEED SUBROUTINE, BOX NUMBER 3
FLOWRATE? 250
WT FRACTION OF WATER? 1
WT FRACTION OF SALT? 0

ENTER FEED,MIX,SPLIT,SETTLE,OR END? MIX
THIS IS MIX SUBROUTINE,BOX NUMBER 4
NUMBER OF STREAMS TO BE MIXED? 2
BOX NUMBER FROM WHICH STREAM COMES? 1
BOX NUMBER FROM WHICH STREAM COMES? 2

ENTER FEED,MIX,SPLIT,SETTLE,OR END? SET
THIS IS SETTLE SUBROUTINE,BOX NUMBER 5
THIS NUMBER ALSO DESIGNATES OVERFLOW STREAM.
UNDERFLOW STREAM IS DESIGNATED BOX NUMBER 6
BOX NUMBER FROM WHICH FLOW COMES? 4
SOLIDS/SOLUTION RATIO FOR UNDERFLOW? 2
```

```
ENTER FEED,MIX,SPLIT,SETTLE,OR END? MIX
THIS IS MIX SUBROUTINE,BOX NUMBER 7
NUMBER OF STREAMS TO BE MIXED? 2
BOX NUMBER FROM WHICH STREAM COMES? 3
BOX NUMBER FROM WHICH STREAM COMES? 6

ENTER FEED,MIX,SPLIT,SETTLE,OR END? SET
THIS IS SETTLE SUBROUTINE,BOX NUMBER 8
THIS NUMBER ALSO DESIGNATES OVERFLOW STREAM.
UNDERFLOW STREAM IS DESIGNATED BOX NUMBER 9
BOX NUMBER FROM WHICH FLOW COMES? 7
SOLIDS/SOLUTION RATIO FOR UNDERFLOW? 2

ENTER FEED,MIX,SPLIT,SETTLE,OR END? END

IF ONE OR MORE STREAMS ARE TORN,TYPE Y ELSE N? N
```

```
FEED
BOX NUMBER 1
FLOWRATE= 300
WEIGHT FRACTION OF WATER=0.0500
WEIGHT FRACTION OF SALT=0.0150
WEIGHT FRACTION OF SAND=0.9350

FEED
BOX NUMBER 2
FLOWRATE= 250
WEIGHT FRACTION OF WATER=1.0000
WEIGHT FRACTION OF SALT=0.0000
WEIGHT FRACTION OF SAND=0.0000

FEED
BOX NUMBER 3
FLOWRATE= 250
WEIGHT FRACTION OF WATER=1.0000
WEIGHT FRACTION OF SALT=0.0000
WEIGHT FRACTION OF SAND=0.0000

MIX
BOX NUMBER 4
FLOWRATE= 550
WEIGHT FRACTION OF WATER=0.4818
WEIGHT FRACTION OF SALT=0.0082
WEIGHT FRACTION OF SAND=0.5100

SETTLE,OVERFLOW      1st Stage
BOX NUMBER 5         Spent Wash
FLOWRATE= 129.25
WEIGHT FRACTION OF WATER=0.9833
WEIGHT FRACTION OF SALT=0.0167
WEIGHT FRACTION OF SAND=0.0000

SETTLE,UNDERFLOW
BOX NUMBER 6
SOLIDS/SOLUTION RATIO= 2
FLOWRATE= 420.75
WEIGHT FRACTION OF WATER=0.3278
WEIGHT FRACTION OF SALT=0.0056
WEIGHT FRACTION OF SAND=0.6667
```

```
MIX
BOX NUMBER 7
FLOWRATE= 670.75
WEIGHT FRACTION OF WATER=0.5783
WEIGHT FRACTION OF SALT=0.0035
WEIGHT FRACTION OF SAND=0.4182

SETTLE,OVERFLOW           2nd Stage
BOX NUMBER 8              Spent Wash
FLOWRATE= 250
WEIGHT FRACTION OF WATER=0.9940
WEIGHT FRACTION OF SALT=0.0060
WEIGHT FRACTION OF SAND=0.0000

SETTLE,UNDERFLOW          Washed Sand
BOX NUMBER 9
SOLIDS/SOLUTION RATIO= 2
FLOWRATE= 420.75
WEIGHT FRACTION OF WATER=0.3313
WEIGHT FRACTION OF SALT=0.0020
WEIGHT FRACTION OF SAND=0.6667
```

COMPUTER SOLUTION, PART B:

```
TO RERUN WITH DIFFERENT VALUES TYPE Y ELSE N? Y

TO MODIFY FLOWSHEET TYPE Y, ELSE N? Y
THIS IS MOD SUBROUTINE.   HOW MANY
EXISTING BOXES TO BE ALTERED? 0
IF NEW BOXES TO BE ADDED INPUT Y ELSE N? N
THIS IS CHANGE SUBROUTINE.   INPUT NUMBER OF
ENTERING STREAMS TO BE CHANGED? 1
BOX NUMBER OF ENTERING STREAM? 3
NEW FLOWRATE? 275
NEW WEIGHT FRACTION OF WATER? 1
NEW WEIGHT FRACTION OF SALT? 0
NUMBER OF SPLITTERS TO BE CHANGED? 0
NUMBER OF SETTLERS TO BE CHANGED? 2
BOX NUMBER OF SETTLER? 8
NEW VALUE OF SOLIDS/SOLUTION RATIO? 2.33
BOX NUMBER OF SETTLER? 6
NEW VALUE OF SOLIDS/SOLUTION RATIO? 2.33
```

```
IF ONE OR MORE STREAMS ARE TORN,TYPE Y ELSE N? Y
THIS IS TEAR SUBROUTINE
HOW MANY TORN STREAMS? 1
BOX NUMBER WHICH TORN STREAM 1  LEAVES? 8
BOX NUMBER WHICH TORN STREAM 1  ENTERS? 2
C1= 9

FEED
BOX NUMBER 1
FLOWRATE= 300
WEIGHT FRACTION OF WATER=0.0500
WEIGHT FRACTION OF SALT=0.0150
WEIGHT FRACTION OF SAND=0.9350

FEED                        Wash from Settler 2
BOX NUMBER 2                (after Tear)
FLOWRATE= 274.9512
WEIGHT FRACTION OF WATER=0.9936
WEIGHT FRACTION OF SALT=0.0064
WEIGHT FRACTION OF SAND=0.0000

FEED
BOX NUMBER 3
FLOWRATE= 275
WEIGHT FRACTION OF WATER=1.0000
WEIGHT FRACTION OF SALT=0.0000
WEIGHT FRACTION OF SAND=0.0000

MIX
BOX NUMBER 4
FLOWRATE= 574.9024
WEIGHT FRACTION OF WATER=0.5012
WEIGHT FRACTION OF SALT=0.0109
WEIGHT FRACTION OF SAND=0.4879

SETTLE,OVERFLOW
BOX NUMBER 5
FLOWRATE= 174.0161
WEIGHT FRACTION OF WATER=0.9788
WEIGHT FRACTION OF SALT=0.0212
WEIGHT FRACTION OF SAND=0.0000

SETTLE,UNDERFLOW
BOX NUMBER 6
SOLIDS/SOLUTION RATIO= 2.33
FLOWRATE= 400.8863
WEIGHT FRACTION OF WATER=0.2939
WEIGHT FRACTION OF SALT=0.0064
WEIGHT FRACTION OF SAND=0.6997

MIX
BOX NUMBER 7
FLOWRATE= 675.8863
WEIGHT FRACTION OF WATER=0.5812
WEIGHT FRACTION OF SALT=0.0038
WEIGHT FRACTION OF SAND=0.4150
```

```
SETTLE,OVERFLOW           Wash from Settler 2
BOX NUMBER 8              (before tear)
FLOWRATE= 275
WEIGHT FRACTION OF WATER=0.9935
WEIGHT FRACTION OF SALT=0.0065
WEIGHT FRACTION OF SAND=0.0000

SETTLE,UNDERFLOW          Washed Sand
BOX NUMBER 9
SOLIDS/SOLUTION RATIO= 2.33
FLOWRATE= 400.8863
WEIGHT FRACTION OF WATER=0.2984
WEIGHT FRACTION OF SALT=0.0019
WEIGHT FRACTION OF SAND=0.6997


TO RERUN WITH DIFFERENT VALUES TYPE Y ELSE N? V

TO MODIFY FLOWSHEET TYPE Y, ELSE N? N
END
Ok
```

Inspection of the printout shows that for the cross-current operation the underflow composition from the second stage is:

Water	0.3313
Salt	0.0020
Sand	0.6667
	1.0000

Hence for 1 kg of slurry, dry weight = 0.6687 kg and composition is:

Salt	0.0030
Sand	0.9970
	1.0000

For countercurrent operation the printout shows the underflow composition as:

Water	0.2984
Salt	0.0019
Sand	0.6997
	1.0000

Hence for 1 kg of slurry, dry weight = 0.7016 kg and composition is:

Salt	0.0027
Sand	0.0073
	1.0000

Countercurrent washing under the new conditions will give a slightly improved product.

By comparison, the composition of the raw sand on a dry basis is:

Salt	0.0158
Sand	0.9842
	1.0000

STEP FIVE

A further modification to the program would be the inclusion of means to generate a 'control block' when required. Suppose for instance, in example 1.3 above that outlet composition of the sand slurry had been specified, and it was required to determine the wash water rate. The execution of the program would then proceed inside another iterative loop; an initial guess for flowrate would be read in, and the calculated value of outlet compositions compared with that specified. The value of flowrate would then be adjusted, and another iteration performed, and so on until convergence had been achieved (1).

These and many other options are available in commercial flowsheeting software.

PROBLEMS - CHAPTER 1

1. Use the program to modify the solution given in example 1.3 by the addition of a third settler.

2. A vegetable oil is extracted from seeds by leaching with a hydrocarbon solvent. The process is carried out batchwise using 500 kg. batches of seed and 1000 kg. batches of solvent.

The fresh seeds contain 20 wt% oil, and 96% of their oil content is to be removed in the process. How may washes with fresh solvent will be required in order to achieve this if the seed retains 0.38 kg. solvent per kg. oil-free seed? If a countercurrent washing process is used instead, what quantity of solvent, and what number of stages, would you recommend?

3. A baking process involves the blending of the following ingredients to form Dry Blend A:

flour	160 kg
sugar	80 kg
salt	0.6 kg
raising agent	1.2 kg

Dried milk, eggs and fat are blended in the following quantities to form Blend B:

dried milk	10 kg
dried eggs	10 kg
fat	40 kg

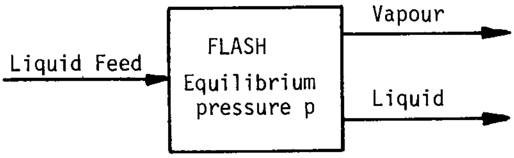
Dry Blend A is mixed with 70 kg water.

Dry Blend B is mixed with 20 kg water.

The two batches are then combined together for final mixing, and charged into moulds for baking.

Calculate the compositions and quantities of all streams, assuming spillage losses of 0.2% occur at each dry blending stage.

4. Write a subroutine to model a single stage flash evaporator.



5. Modify the program to include the control block described above under 'Step Five'.

6. Write subroutines appropriate to your own line of work and incorporate them in the program.

REFERENCES

1 R.M. Felder & R.W. Rousseau, Elementary Principles of Chemical Processes, John Wiley & Sons, New York, U.S.A., 1978.

2 P. Benedek, Editor, Steady State Flowsheeting of Chemical Plants, Elsevier Scientific Pub. Co. Amsterdam, Netherlands, 1980.

3 D.F. Rudd & C.C. Watson, Strategy of Process Engineering, John Wiley & Sons, New York, U.S.A., 1968.

4 M.E. Leesley, Editor, Computer-aided Process Plant Design, Gulf Publishing Co., Houston, Texas, U.S.A., 1982.

5 T.G. Poole, J.Z. Szymankiewicz, Using Simulation to Solve Problems, McGraw Hill, Maidenhead, U.K. 1977.

6 C.J. Geankopolis, Transport Processes & Unit Operations, Allyn & Bacon, New Jersey, U.S.A. 1978.

7 J.M. Coulson, J.F. Richardson and Others, Chemical Engineering, Vol. II, 3rd Edition, Pergamon Press, Oxford, U.K. 1978.

Chapter 2

INTERPRETATION AND ACCESSING OF RESULTS AND PHYSICAL DATA

Mathematical methods of handling these tasks are covered in standard mathematical and statistical text books. Some books in this area are of particular interest to chemical engineers (1), (2), (3), (4). Computer routines for many statistical methods are also available in the literature (5), (6). Two examples are given below, with possible applications. In addition, the applicability of thermodynamic methods and their use in computer programs should not be overlooked (7); see also references with Chapter 4. The chapter is concluded with a section on the solution of simultaneous linear equations.

Regression Analysis

Suppose a number of sets of observational or experimental data to exist. Each data set consists of values of a number of independent variables x; and the corresponding value of the dependent variable y. The purpose of regression analysis is to seek out a mathematical relationship between the dependent variable and the independent variables. The relationship selected should be that which predicts the value of the dependent variable with the least error.

Least Squares Polynomial Regression

The simplest form of this analysis method occurs when we consider data in which the dependent variable y is a function of only one independent variable x. Such data may be represented by a polynomial relationship of the form:

$$Y = a + bx + cx^2 + dx^3 + \dots.$$

For each data point exists a value of the independent variable x; Y is the value of the dependent variable predicted by the relationship; a, b, c, etc. are parameters whose values have to be determined by analysis of the data.

Thus if there are n data points, a polynomial relationship may be written for each one:

$$Y_1 = a + bx_1 + cx_1^2 + dx_1^3 + \dots. \qquad (2.1)$$

$$\vdots$$

$$Y_i = a + bx_i + cx_i^2 + dx_i^3 + \dots.$$

$$\vdots$$

$$Y_n = a + bx_n + cx_n^2 + dx_n^3 + \dots.$$

We assume that the uncertainty in x is insignificant compared with the uncertainty in Y. There is then, a difference or error between the predicted value Y, and the actual or measured value y, i.e. error =

$$Y_i - y_i = (a + bx_i + cx_i^2 + dx_i^3 \ldots.) - y_i \qquad (2.2)$$

For a polynomial of any given order, the best fitting curve through the data will be that which makes the sum of the squares of the errors a minimum. The sum of the squares is:

$$\sum_{i=1}^{n} (Y_i - y_i)^2 = (Y_1 - y_1)^2 + (Y_2 - y_2)^2 + \ldots. (Y_n - y_n)^2 \qquad (2.3)$$

Before going further it is necessary to look at the order of the polynomial employed. The simplest case would be of the employment of a polynomial of order 1, that is, the equation of a straight line:

$Y = a + bx$

In some cases this may fit the data adequately, in other cases, a polynomial of order 2, that is, the equation of a parabola may fit the data better, i.e.

$Y = a + bx + cx^2$.

In many cases, a polynomial of still higher order may be required. To carry out a full regression analysis, a series of polynomials of order 1, 2, 3 ...m would each be fitted to the data. The polynomial yielding the lowest value for the sum of the squares of the errors would be the optimum one to employ. Theoretically the sum of the squares of the errors should be less for successively higher order polynomials, but in practice this is not the case due to round off errors in the computations (8).

In order to explain the method further, a second order polynomial will be considered. In this case we require to find values of a, b and c only (equation 2.1). These values will be the ones which make the sum of the squares of the errors a minimum, and this will occur when:

$$\frac{\partial}{\partial a}\left[\sum (Y_i - y_{i.})^2\right] = 0 \qquad (2.4a)$$

$$\frac{\partial}{\partial b}\left[\sum (Y_i - y_i)^2\right] = 0 \qquad (2.4b)$$

$$\frac{\partial}{\partial c}\left[\sum (Y_i - y_i)^2\right] = 0 \qquad (2.4c)$$

The results of differentiation are:

$$\frac{\partial F}{\partial a} = na + b\ \Sigma x_i + c\ \Sigma x_i^2 = \Sigma y_i \qquad (2.5a)$$

$$\frac{\partial F}{\partial b} = a\Sigma x_i + b\Sigma x_i^2 + c\Sigma x_i^3 = \Sigma x_i y_i \qquad (2.5b)$$

$$\frac{\partial F}{\partial c} = a\Sigma x_i^2 + b\Sigma x_i^3 + c\Sigma x_i^4 = \Sigma x_i^2 y_i \qquad (2.5c)$$

If we consider the application of the method to the equation of a straight line (2-term polynominal) then we need only include the first two of the above equations.

Rearranging the first equation gives:

$$a = \frac{\Sigma y_i}{n} - \frac{b\Sigma x_i}{n} \qquad (2.6)$$

Substituting this value of a into the second equation and rearranging gives:

$$b = \frac{\Sigma x_i y_i - \frac{\Sigma x_i \Sigma y_i}{n}}{\Sigma x_i^2 - \frac{(\Sigma x_i)^2}{n}} \qquad (2.7)$$

Computer Solution of Linear Regression

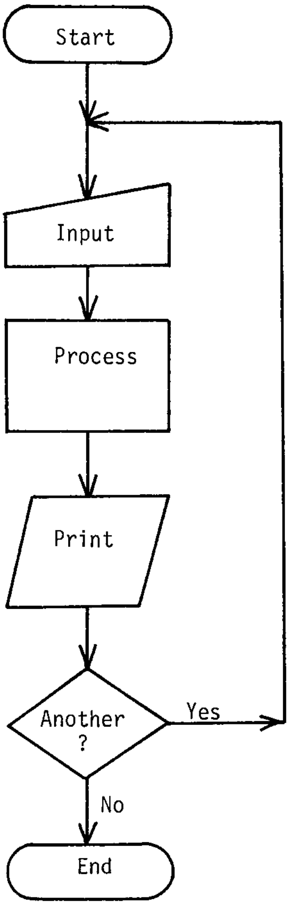
It is required to write a program which, given pairs of values $x_i\, y_i$ (where i can be any number greater than 2) will determine the values of the coefficients a and b in the equation:

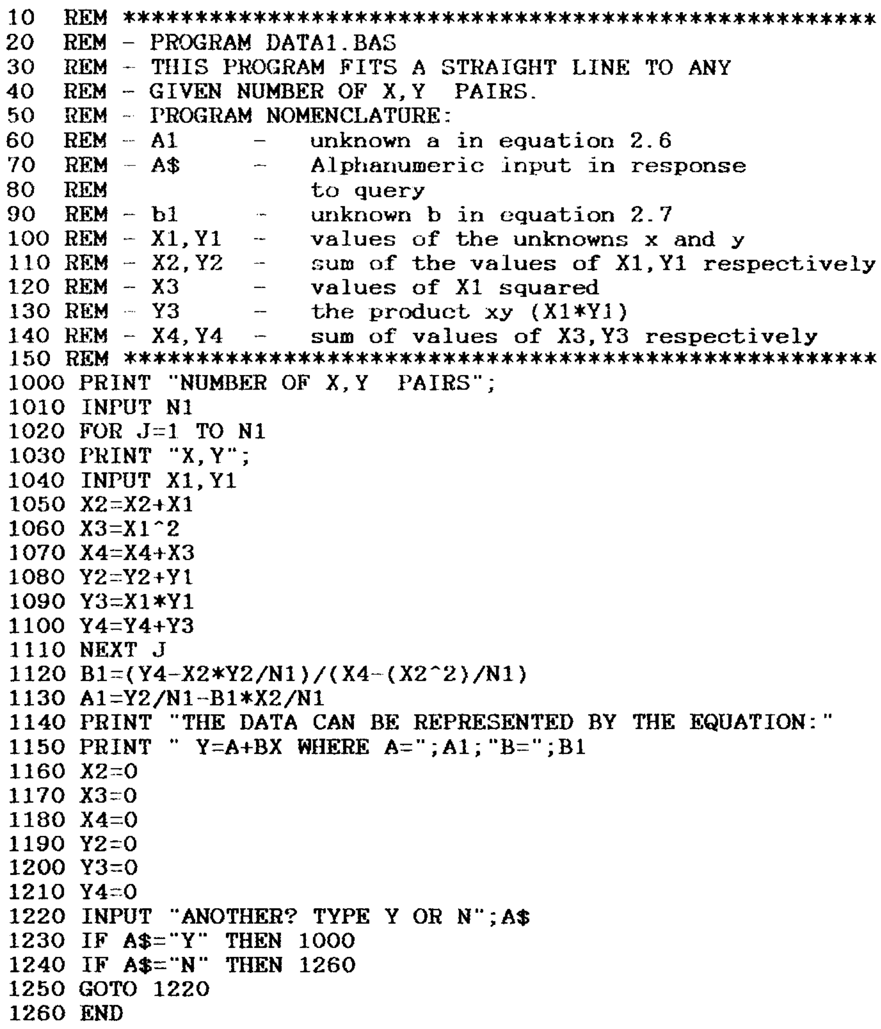
$y = a + bx$

Program DATA1 presented below does this, using equations 2.6 and 2.7.

The use of a straight line relationship may well be adequate to represent physical data over a limited range; experimental data is often represented in this way, by the choice of suitable ordinates:

DATA1.BAS





```
10   REM ************************************************
20   REM - PROGRAM DATA1.BAS
30   REM - THIS PROGRAM FITS A STRAIGHT LINE TO ANY
40   REM - GIVEN NUMBER OF X,Y  PAIRS.
50   REM - PROGRAM NOMENCLATURE:
60   REM - A1      -   unknown a in equation 2.6
70   REM - A$      -   Alphanumeric input in response
80   REM               to query
90   REM - b1      -   unknown b in equation 2.7
100  REM - X1,Y1   -   values of the unknowns x and y
110  REM - X2,Y2   -   sum of the values of X1,Y1 respectively
120  REM - X3      -   values of X1 squared
130  REM - Y3      -   the product xy (X1*Y1)
140  REM - X4,Y4   -   sum of values of X3,Y3 respectively
150  REM ************************************************
1000 PRINT "NUMBER OF X,Y  PAIRS";
1010 INPUT N1
1020 FOR J=1 TO N1
1030 PRINT "X,Y";
1040 INPUT X1,Y1
1050 X2=X2+X1
1060 X3=X1^2
1070 X4=X4+X3
1080 Y2=Y2+Y1
1090 Y3=X1*Y1
1100 Y4=Y4+Y3
1110 NEXT J
1120 B1=(Y4-X2*Y2/N1)/(X4-(X2^2)/N1)
1130 A1=Y2/N1-B1*X2/N1
1140 PRINT "THE DATA CAN BE REPRESENTED BY THE EQUATION:"
1150 PRINT " Y=A+BX WHERE A=";A1;"B=";B1
1160 X2=0
1170 X3=0
1180 X4=0
1190 Y2=0
1200 Y3=0
1210 Y4=0
1220 INPUT "ANOTHER? TYPE Y OR N";A$
1230 IF A$="Y" THEN 1000
1240 IF A$="N" THEN 1260
1250 GOTO 1220
1260 END
```

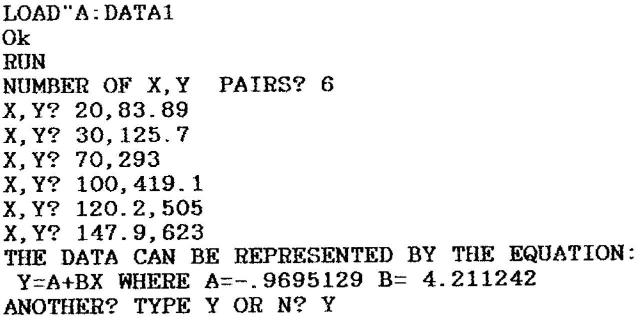
EXAMPLE 2.1

The data of Table 2-1 has been abstracted from Steam Tables. It is required to represent it in simple mathematical form for incorporation into a computer program.

TABLE 2.1

Temperature	Enthalpy of Saturated Liquid	Enthalpy of Saturated Vapour
^{o}C	KJ/kg	KJ/kg
20	83.9	2537.6
30	125.7	2555.7
70	293	2626.3
100	419.1	2675.8
120.2	505	2707
147.9	623	2744

Source: Reprinted with permission from Y.R. Mayhew and G.F.C. Rogers, Thermodynamic and Transport Properties of Fluids, Sl Units, 2nd Edition, Basil Blackwell Oxford, U.K. 1969.



```
LOAD"A:DATA1
Ok
RUN
NUMBER OF X,Y  PAIRS? 6
X,Y? 20,83.89
X,Y? 30,125.7
X,Y? 70,293
X,Y? 100,419.1
X,Y? 120.2,505
X,Y? 147.9,623
THE DATA CAN BE REPRESENTED BY THE EQUATION:
 Y=A+BX WHERE A=-.9695129 B= 4.211242
ANOTHER? TYPE Y OR N? Y
```

Lagrangian Interpolation Method

Much of the data used by engineers has traditionally been presented in tabular form, and consists of a series of values of y for corresponding values of x. Examples include tables of logarithmic and trigonometric functions, steam tables, and tables of physical chemical data such as vapour pressure, solubility, etc.

The use of interpolation formulae has an advantage in accuracy when using such tables in performing manual calculations. Their use is invaluable when that same data is to be incorporated in a computer program.

The Lagrangian method is one of several mathematical interpolation techniques (1). The technique is employed later, in the chapter on distillation.

It is assumed that we have a number of data pairs, x_1 y_1 , x_2 y_2 , ...x_n y_n. The intervals between these data points need not be regular.

We require to calculate the value of y for any given value of x between the limits x_1 and x_n. Then according to the Lagrangian interpolation method, this value of y will be:

$$\begin{aligned} y = {} & y_1 * \frac{(x - x_2)\ (x - x_3)\ (x - x_4)\ \ldots.\ (x - x_n)}{(x_1 - x_2)\ (x_1 - x_3)\ (x_1 - x_4)\ \ldots.\ (x_1 - x_n)} \\ + {} & y_2 * \frac{(x - x_1)\ (x - x_3)\ (x - x_4)\ \ldots.\ (x - x_n)}{(x_2 - x_1)\ (x_2 - x_3)\ (x_2 - x_4)\ \ldots.\ (x_2 - x_n)} \\ + {} & \ldots. \\ + {} & y_n * \frac{(x - x_1)\ (x - x_2)\ (x - x_3)\ \ldots.\ (x - x_{n-1})}{(x_n - x_1)\ (x_n - x_2)\ (x_n - x_3)\ \ldots.\ (x_n - x_{n-1})} \end{aligned} \qquad (2.8)$$

Program DATA2 has been written to employ this method.

The program stores values x_1, x_2, etc. in Matrix $A_1$and values of y_1, y_2, etc. in Matrix B.

Values $(x_1 - x_2)$ etc. are evaluated and stored in Matrix M. The appropriate values are then multiplied together to form the denominators of the above expression, and these are stored in Matrix D. These calculations are done in the first subroutine. See Table 2.2.

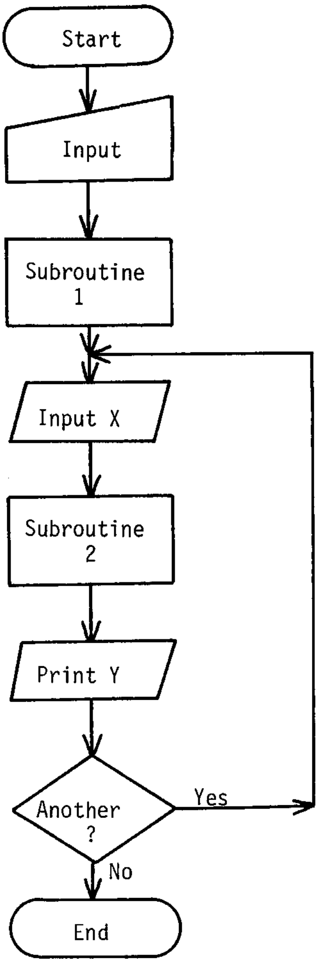
Values of $(x - x_1)$ etc. are then calculated for a given value of x for which the value of y is required. These values of $(x - x_1)$ etc. are again stored in Matrix M. Appropriate values are then multiplied together to give the numerator terms of the above expression, and these are stored in Matrix N. Values of each term in the expression are then evaluated and summed to give the required value of y. These calculations are performed in the second subroutine.

TABLE 2.2

Values stored in Matrix M. The product of each row gives the denominator terms in Equation 2.8. The matrix is used again in the 2nd subroutine to obtain the numerator terms of Equation 2.8.

$x_1 - x_2$	$x_1 - x_3$	$x_1 - x_4$	$x_1 - x_5$	$x_1 - x_6$	J = 1
$x_2 - x_1$	$x_2 - x_3$	$x_2 - x_4$	$x_2 - x_5$	$x_2 - x_6$	J = 2
$x_3 - x_1$	$x_3 - x_2$	$x_3 - x_4$	$x_3 - x_5$	$x_3 - x_6$	J = 3
$x_4 - x_1$	$x_4 - x_2$	$x_4 - x_3$	$x_4 - x_5$	$x_4 - x_6$	J = 4
$x_5 - x_1$	$x_5 - x_2$	$x_5 - x_3$	$x_5 - x_4$	$x_5 - x_6$	J = 5
K = 1	K = 2	K = 3	K = 4	K = 5	

DATA2.BAS


Start
Input
Subroutine 1
Input X
Subroutine 2
Print Y
Another ?
Yes
No
End

```
10   REM ****************************************************
20   REM - PROGRAM DATA2.BAS
30   REM - THIS PROGRAM PERFORMS INTERPOLATION
40   REM - BY THE METHOD OF LAGRANGE
50   REM - PROGRAM NOMENCLATURE:
60   REM - A(J),B(J)    -  Values of the variables x and y
70   REM -                 respectively
80   REM - A$           -  Alphanumeric input in response to
90   REM                   query
100  REM - A1           -  Value of x for which the
110  REM                   corresponding value of y is sought
120  REM - B1           -  Value of y corresponding to the
130  REM                   value of x input as A1
140  REM - D(J)         -  Denominator terms in the Lagrange
150  REM                   expression
160  REM - M(J,K)       -  Factors of the numerator terms,
170  REM -                 x-x, etc
180  REM - N(J)         -  Numerator terms in the Lagrange
190  REM                   expression
200  REM - N1           -  Number of data pairs
210  REM - PROGRAM DESCRIPTION
220  REM - LINES 1000-1060   Matrices are dimensioned, values
230  REM - of x are stored in matrix A, and values of y are
240  REM - stored in matrix B
250  REM - LINES 1070-1140   Using the subroutines contained
260  REM - in statement 1150 onwards, the value of y is
270  REM - calculated corresponding to the given value of x
280  REM - LINES 1160-1420   Values are calculated for the
290  REM - groups occurring in the denominators of the terms
300  REM - of equation 2.8; these are then stored in matrix M
310  REM - These groups are then multiplied together to form
320  REM - the denominators of the terms of equation 2.8,
330  REM - and these are stored in matrix D
340  REM - LINES 1440-1760   Values are calculated for the
350  REM - groups occurring in the numerators of the terms in
360  REM - equation 2.8; these are then stored in matrix M.
370  REM - These groups are then multiplied together to form
380  REM - the numerators of the terms in equation 2.8, and
390  REM - these are stored in matrix N
400  REM - The individual terms of equation 2.8 are then
410  REM - evaluated and summed, giving the required
420  REM - value of y
430  REM ****************************************************
1000 DIM A(20),B(20),D(20),N(20)
1010 DIM M(20,20)
1020 INPUT "NUMBER OF DATA PAIRS";N1
1030 PRINT "INPUT  X,Y"
1040 FOR J=1 TO N1
1050 INPUT A(J),B(J)
1060 NEXT J
1070 GOSUB 1160
1080 INPUT "INPUT VALUE OF X";A1
1090 GOSUB 1460
1100 PRINT "Y=";B1
1110 INPUT "ANOTHER VALUE? TYPE Y OR N";A$
1120 IF A$="Y" THEN 1080
1130 IF A$="N" THEN 1770
1140 GOTO 1110
1150 REM ****************************************************
1160 REM - FIRST SUBROUTINE OF LAGRANGIAN INTERPOLATION
1170 FOR J=1 TO 20
```

```
1180 FOR K=1 TO 20
1190 M(J,K)=0
1200 NEXT K
1210 NEXT J
1220 REM - CALCULATES VALUES OF X1-X2 ETC & ***************
1230 REM - STORES THEM IN MATRIX M
1240 FOR J=1 TO N1-1
1250 FOR K=J TO N1-1
1260 M(J,K)=A(J)-A(K+1)
1270 NEXT K
1280 NEXT J
1290 FOR K=1 TO N1-1
1300 FOR J=K+1 TO N1
1310 M(J,K)=A(J)-A(K)
1320 NEXT J
1330 NEXT K
1340 REM - CALCULATES DENOMINATOR TERMS & ******************
1350 REM - STORES THEM IN MATRIX D
1360 FOR J=1 TO N1
1370 D(J)=1
1380 FOR K=1 TO N1-1
1390 D(J)=D(J)*M(J,K)
1400 NEXT K
1410 NEXT J
1420 RETURN
1430 REM ***************************************************
1440 REM - SECOND SUBROUTINE OF
1450 REM - LAGRANGIAN INTERPOLATION
1460 FOR J=1 TO 20
1470 FOR K=1 TO 20
1480 M(J,K)=0
1490 NEXT K
1500 NEXT J
1510 REM - CALCULATES VALUES OF X-X1 ETC & *****************
1520 REM - STORES THEM IN MATRIX M
1530 FOR J=1 TO N1-1
1540 FOR K=J TO N1-1
1550 M(J,K)=A1-A(K+1)
1560 NEXT K
1570 NEXT J
1580 FOR K=1 TO N1-1
1590 FOR J=K+1 TO N1
1600 M(J,K)=A1-A(K)
1610 NEXT J
1620 NEXT K
1630 REM - CALCULATES NUMERATOR TERMS & ********************
1640 REM - STORES THEM IN MATRIX N
1650 FOR J=1 TO N1
1660 N(J)=1
1670 FOR K=1 TO N1-1
1680 N(J)=N(J)*M(J,K)
1690 NEXT K
1700 NEXT J
1710 B1=0
1720 REM - EVALUATES EACH TERM & ADDS THEM *****************
1730 FOR J=1 TO N1
1740 B1=B1+B(J)*N(J)/D(J)
1750 NEXT J
1760 RETURN
1770 END
```

EXAMPLE 2.2

Store the vapour/liquid equilibrium data of Table 2-3 using program DATA2.

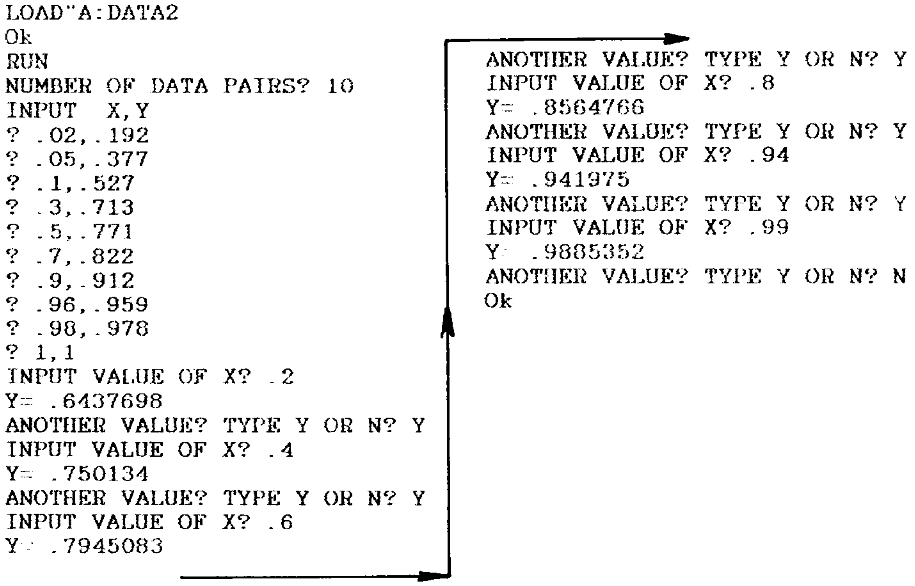
TABLE 2-3

Equilibrium Data for Ethanol-Water System
at 101.325kPa (1 Atm)*

Temperature		Vapour-Liquid Equilibria Mass Fraction Ethanol		Temperature		Vapour-Liquid Equilibria Mass Fraction Ethanol	
°C	°F	x	y	°C	°F	x	y
100.00	212	0	0	81.0	177.8	0.600	0.794
98.1	208.5	0.020	0.192	80.1	176.2	0.700	0.822
95.2	203.4	0.050	0.377	79.1	174.3	0.800	0.858
91.8	197.2	0.100	0.527	78.3	173.0	0.900	0.912
87.3	189.2	0.200	0.656	78.2	172.8	0.940	0.942
84.7	184.5	0.300	0.713	78.1	172.7	0.960	0.959
83.2	181.7	0.400	0.746	78.2	172.8	0.980	0.978
82.0	179.6	0.500	0.771	78.3	173.0	1.00	1.00

Source: Reprinted with permission from G.G. Brown & Others, Unit Operations, John Wiley & Sons Inc., New York, U.S.A. Copyright 1950 ©

Note that instead of inputting all the data to the program, (which would of course be done if it were to be used for further computations), only 10 data pairs have been selected. A comparison can thus be made between the given values for the remaining data pairs, and those calculated by interpolation.



```
LOAD"A:DATA2
Ok
RUN
NUMBER OF DATA PAIRS? 10
INPUT  X,Y
? .02,.192
? .05,.377
? .1,.527
? .3,.713
? .5,.771
? .7,.822
? .9,.912
? .96,.959
? .98,.978
? 1,1
INPUT VALUE OF X? .2
Y= .6437698
ANOTHER VALUE? TYPE Y OR N? Y
INPUT VALUE OF X? .4
Y= .750134
ANOTHER VALUE? TYPE Y OR N? Y
INPUT VALUE OF X? .6
Y= .7945083
ANOTHER VALUE? TYPE Y OR N? Y
INPUT VALUE OF X? .8
Y= .8564766
ANOTHER VALUE? TYPE Y OR N? Y
INPUT VALUE OF X? .94
Y= .941975
ANOTHER VALUE? TYPE Y OR N? Y
INPUT VALUE OF X? .99
Y= .9885352
ANOTHER VALUE? TYPE Y OR N? N
Ok
```

Functions of Two Variables

Such often required information as values of enthalpy or density, is frequently a function of two variables, pressure and temperature, or concentration and temperature.

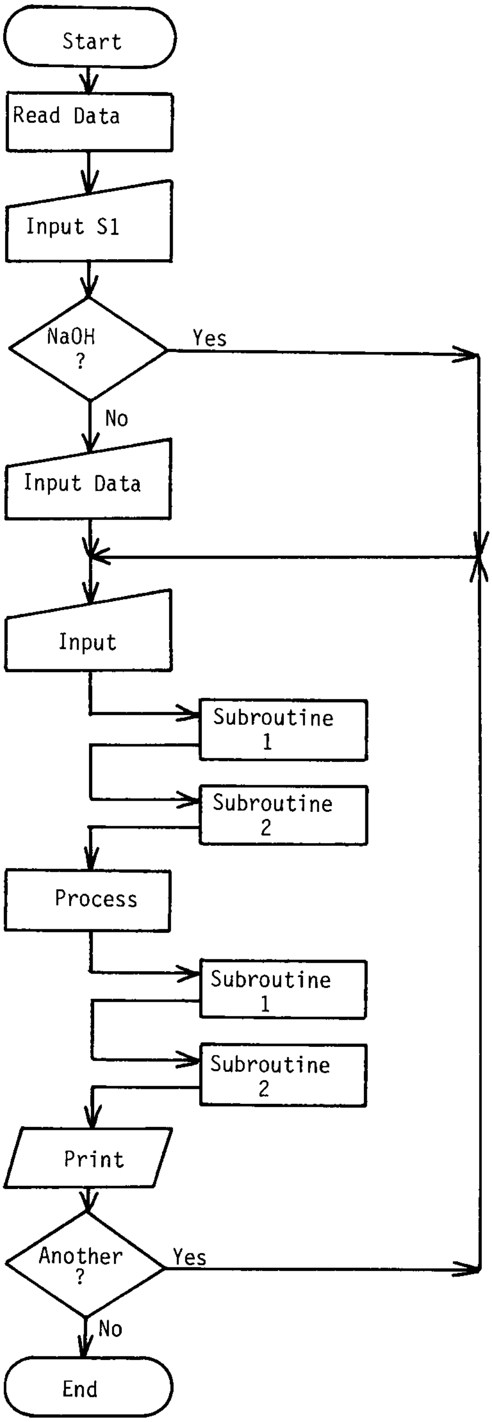
Modification of the previous program provides a method to store and represent such data.

Let the dependent variable be x and the independent variables y and z. By use of the subroutines already developed, we find by Lagrangian interpolation, the value of x, at the designated value of y for every value of z for which we have data. We thus have a number of values of x as a function of z only. The Lagrangian subroutines are used again to obtain the value of x at the designated value of z. This is the required value.

A(J) and B(J) are used in the subroutines as before. However, since these are now used twice, values of X(J,K) and Y(J,K) and later Z(J) and E(J) have to be exchanged with A(J) and B(J).

These procedures are demonstrated in program DATA3. Enthalpy data for NaOH solutions is included with the program; this was employed during program development.

DATA3.BAS


Start
Read Data
Input S1
NaOH
?
Yes
No
Input Data
Input
Subroutine
1
Subroutine
2
Process
Subroutine
1
Subroutine
2
Print
Another
?
Yes
No
End

```
10  REM ****************************************************
20  REM - PROGRAM DATA3.BAS
30  REM - THIS PROGRAM PERFORMS INTERPOLATION
40  REM - BY THE METHOD OF LAGRANGE APPLIED
50  REM - TO DATA WHICH IS A FUNCTION OF 2 VARIABLES
60  REM - PROGRAM NOMENCLATURE
70  REM - A(J),B(J) -  Sets of values of y and x
80  REM                respectively for a particular
90  REM                value of z
100 REM - A$        -  Alphanumeric input in response
110 REM                to query
120 REM - A1        -  Used to duplicate the value of Y1 or
130 REM                Z1 for use in the second Lagrange
140 REM                subroutine
150 REM - D(J)      -  Denominator terms, as in DATA2
160 REM - E(J)      -  Set of values of y, one for each
170 REM                value of z for which we have data
180 REM - M(J,K)    -  Factors of the numerator terms,as
190 REM                in DATA2
200 REM - N(J)      -  Numerator terms as in DATA2
210 REM - N1        -  Number of x,y pairs for each z value
220 REM - N2        -  Number of values of variable x
230 REM - X(J,K)    -  Values of the dependent variable x
240 REM - Y(J,K),Z(J,K) - Values of the independent
250 REM -              variables y and z
260 REM - Y1,Z1     -  Values of y and z for which the
270 REM                corresponding value of x is sought
280 REM - PROGRAM DESCRIPTION
290 REM - LINES 1340-1430   For each value of z, all the
300 REM - values of x and y are used to generate the
310 REM - denominator terms (equation 2.8).   The chosen
320 REM - value of y (Y1) is then ascribed to A1 and a
330 REM - value of x (B1) is obtained from the second
340 REM - subroutine at line 1410, for each value of z.
350 REM - These values are stored in matrix E (line 1420)
360 REM - LINES 1440-1580   The values of E(J) are taken
370 REM - with the values Z(J) for another run through
380 REM - the Lagrange subroutines, to generate the
390 REM - required value of x
400 REM ****************************************************
1000 DIM A(10),B(10),D(10),E(10),N(10),Z(10)
1010 DIM M(10,10),X(10,10),Y(10,10)
1020 N1=5
1030 N2=4
1040 FOR J=1 TO N2
1050 FOR K=1 TO N1
1060 READ Y(J,K),X(J,K)
1070 NEXT K
1080 NEXT J
1090 FOR J=1 TO N2
1100 READ Z(J)
1110 NEXT J
1120 REM - ENTHALPY OF NAOH IN BTU/LB,0-30%,40-400 DEG F
1130 DATA 40,8,100,68,200,167,300,268,400,375
1140 DATA 40,6,100,60,200,155,300,244,400,335
1150 DATA 40,5,100,57,200,144,300,232,400,322
1160 DATA 40,13,100,64,200,151,300,237,400,324
1170 DATA 0,10,20,30
1180 INPUT "TYPE 1 FOR NAOH DATA";S1
1190 IF S1=1 THEN 1320
1200 PRINT "INPUT YOUR OWN DATA"
```

```
1210 INPUT "HOW MANY Z VALUES";N2
1220 PRINT "HOW MANY Y/X PAIRS FOR EACH Z VALUE";
1230 INPUT N1
1240 FOR J=1 TO N2
1250 PRINT "Z(";J;")";
1260 INPUT Z(J)
1270 PRINT "Y/X VALUES?"
1280 FOR K=1 TO N1
1290 INPUT Y(J,K),X(J,K)
1300 NEXT K
1310 NEXT J
1320 PRINT "INPUT VALUES OF Z&Y FOR WHICH X VALUE REQUIRED"
1330 INPUT Z1,Y1
1340 FOR G=1 TO N2
1350 FOR H=1 TO N1
1360 A(H)=Y(G,H)
1370 B(H)=X(G,H)
1380 NEXT H
1390 GOSUB 1600
1400 A1=Y1
1410 GOSUB 1830
1420 E(G)=B1
1430 NEXT G
1440 FOR J=1 TO 10
1450 A(J)=Z(J)
1460 B(J)=E(J)
1470 NEXT J
1480 X1=N1
1490 N1=N2
1500 GOSUB 1600
1510 A1=Z1
1520 GOSUB 1830
1530 PRINT "X=";B1
1540 N1=X1
1550 INPUT "ANOTHER X VALUE?   TYPE Y OR N";A$
1560 IF A$="Y" THEN 1320
1570 IF A$="N" THEN 2100
1580 GOTO 1550
1590 REM ***************************************************
1600 REM - FIRST SUBROUTINE OF LAGRANGIAN INTERPOLATION
1610 REM - CALCULATES VALUES OF X1-X2 ETC &
1620 REM - STORES THEM IN MATRIX M
1630 FOR J=1 TO N1-1
1640 FOR K=J TO N1-1
1650 M(J,K)=A(J)-A(K+1)
1660 NEXT K
1670 NEXT J
1680 FOR K=1 TO N1-1
1690 FOR J=K+1 TO N1
1700 M(J,K)=A(J)-A(K)
1710 NEXT J
1720 NEXT K
1730 REM - CALCULATES DENOMINATOR TERMS & ******************
1740 REM - STORES THEM IN MATRIX D
1750 FOR J=1 TO N1
1760 D(J)=1
1770 FOR K=1 TO N1-1
1780 D(J)=D(J)*M(J,K)
1790 NEXT K
1800 NEXT J
1810 RETURN
```

```
1820 REM ***************************************************
1830 REM - SECOND SUBROUTINE OF LAGRANGIAN INTERPOLATION
1840 REM - CALCULATES VALUES OF X-X1 ETC &
1850 REM - STORES THEM IN MATRIX M
1860 FOR J=1  TO N1-1
1870 FOR K=J TO N1-1
1880 M(J,K)=A1-A(K+1)
1890 NEXT K
1900 NEXT J
1910 FOR K=1 TO N1-1
1920 FOR J=K+1 TO N1
1930 M(J,K)=A1-A(K)
1940 NEXT J
1950 NEXT K
1960 REM - CALCULATES NUMERATOR TERMS & *******************
1970 REM - STORES THEM IN MATRIX N
1980 FOR J=1 TO N1
1990 N(J)=1
2000 FOR K=1 TO N1-1
2010 N(J)=N(J)*M(J,K)
2020 NEXT K
2030 NEXT J
2040 B1=0
2050 REM - EVALUATES EACH TERM & ADDS THEM *****************
2060 FOR J=1 TO N1
2070 B1=B1+B(J)*N(J)/D(J)
2080 NEXT J
2090 RETURN
2100 END
```

EXAMPLE 2.3

Use program DATA3 to store typical density data such as that of Table 2.4, as a function of both temperature and concentration.

TABLE 2.4

Density of NaCl solutions

Sodium Chloride (NaCl)							
%	0°C.	10°C.	25°C.	40°C.	60°C.	80°C.	100°C.
1	1.00747	1.00707	1.00409	0.99908	0.9900	0.9785	0.9651
2	1.01509	1.01442	1.01112	1.00593	.9967	.9852	.9719
4	1.03038	1.02920	1.02530	1.01977	1.0103	.9988	.9855
8	1.06121	1.05907	1.05412	1.04798	1.0381	1.0264	1.0134
12	1.09244	1.08946	1.08365	1.07699	1.0667	1.0549	1.0420
16	1.12419	1.12056	1.11401	1.10688	1.0962	1.0842	1.0713
20	1.15663	1.15254	1.14533	1.13774	1.1268	1.1146	1.1017
24	1.18999	1.18557	1.17776	1.16971	1.1584	1.1463	1.1331
26	1.20709	1.20254	1.19443	1.18614	1.1747	1.1626	1.1492

Source: National Research Council, International Critical Tables, McGraw Hill Book Co, 1933.

A portion only of the above data has been used below. A comparison can thus be made between the remaining density values on the table, and those calculated by the interpolation program.

```
LOAD"A:DATA3
Ok
RUN
TYPE 1 FOR NAOH DATA? 2
INPUT YOUR OWN DATA
HOW MANY Z VALUES? 5
HOW MANY Y/X PAIRS FOR EACH Z VALUE? 4
Z( 1 )? 1
Y/X VALUES?
? 0,1.00747
? 25,1.00409
? 60,.99
? 100,.9651
Z( 2 )? 4
Y/X VALUES?
? 0,1.03038
? 25,1.0253
? 60,1.0103
? 100,.9855
Z( 3 )? 12
Y/X VALUES?
? 0,1.09244
? 25,1.08365
? 60,1.0667
? 100,1.042
Z( 4 )? 20
Y/X VALUES?
? 0,1.15663
? 25,1.14533
? 60,1.1268
? 100,1.1017
Z( 5 )? 26
Y/X VALUES?
? 0,1.20709
? 25,1.19443
? 60,1.1747
? 100,1.1492
INPUT VALUES OF Z&Y FOR WHICH X VALUE REQUIRED
? 2,10
X= 1.014241
ANOTHER X VALUE?   TYPE Y OR N? 2,80
?Redo from start
ANOTHER X VALUE?   TYPE Y OR N? Y
INPUT VALUES OF Z&Y FOR WHICH X VALUE REQUIRED
? 2,80
X= .9850922
ANOTHER X VALUE?   TYPE Y OR N? Y
INPUT VALUES OF Z&Y FOR WHICH X VALUE REQUIRED
? 8,25
X= 1.054119
ANOTHER X VALUE?   TYPE Y OR N? Y
INPUT VALUES OF Z&Y FOR WHICH X VALUE REQUIRED
? 16,60
X= 1.09625
ANOTHER X VALUE?   TYPE Y OR N? N
Ok
```

Solution of Simultaneous Linear Equations

This is a frequently required procedure; examples in this text include:

solution of multi-effect evaporators (Chapter 3);

solution of finite difference equations for heat transfer (Chapters 6, 7 and 8).

The choice of solution method lies between the use of matrix algebra and numerical methods involving iteration.

Solution by Matrix Algebra

This method is fully explained in numerous texts and will be only briefly discussed here (2), (3).

Suppose that we have a set of linear algebraic equations such as the following:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \dots a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \dots a_{nn}x_n &= b_n \end{aligned} \qquad (2.9)$$

where x_1, x_2 etc are unknown variables;

a_{11}, a_{12} etc are known constant coefficients;

b_1 , b_2 etc are known constants.

This set can be written in matrix notation as:

$$\begin{bmatrix} a_{11} & \dots\dots & a_{1n} \\ a_{21} & & a_{2n} \\ \vdots & & \vdots \\ a_{n1} & & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

In more precise matrix notation this can be expressed as:

$$\mathbf{A}\mathbf{X} = \mathbf{B}$$

where $\mathbf{A}$ is a square matrix of order n and $\mathbf{B}$ and $\mathbf{X}$ are column vectors of order n.

The solution of this equation is:

$$\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$$

where $\mathbf{A}^{-1}$ is the reciprocal of matrix $\mathbf{A}$.

The procedures for evaluating the inverse of a matrix are explained in mathematical textbooks, and are easily applied providing that the order of the matrix is not greater than 3 or 4. With higher orders, the difficulties in evaluation become formidable, and it then becomes necessary to employ Gaussian elimination (as used in Chapter 5 with the Simplex method). Nevertheless, a program to handle matrices of high order is complex and requires considerable storage capacity.

Consequently, micro and desk top computers do not include matrix manipulations among their normal range of mathematical functions.

Where matrix algebra is available to the programmer, then the solution of simultaneous equations is easy, as shown by the following simple program, DATA4.BAS.

```
10 REM ******************************************************
20 REM - PROGRAM DATA4.BAS
30 REM - THIS PROGRAM SOLVES SIMULTANEOUS LINEAR EQUATIONS
40 REM - BY MATRIX ALGEBRA
50 REM - UNSUITABLE FOR USE UNLESS YOU HAVE A MATRIX
60 REM - PROCESSOR
70 REM - LINE 1000   An arbitrary value of 20 has been set
80 REM - for the maximum number of unknowns to be handled
90 REM - LINES 1020 - 1050   Matrices are dimensioned for
100 REM - the inversion and multiplication steps
110 REM *****************************************************
1000 DIM A(20,20),B(20),C(20,20),T(20)
1010 INPUT "NUMBER OF UNKNOWNS";N
1020 MAT A=ZER(N,N)
1030 MAT B=ZER(N)
1040 MAT C=ZER(N,N)
1050 MAT T=ZER(N)
1060 PRINT "INPUT MATRIX"
1070 MAT INPUT A(N,N)
1080 PRINT "INPUT VECTOR"
1090 MAT INPUT B(N)
1100 MAT C=INV(A)
1110 MAT T=C*B
1120 MAT PRINT T
1130 END
```

EXAMPLE 2.4

Use the above program to solve the following three simultaneous equations:

$3x_1 + 8x_2 = 71$

$5x_2 + 3x_3 = 41$

$x_1 \ - 5x_3 = -40$

The equations are first rewritten in matrix form:

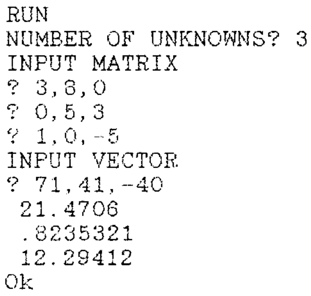
$$\begin{bmatrix} 3 & 8 & 0 \\ 0 & 5 & 3 \\ 1 & 0 & -5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 71 \\ 41 \\ -40 \end{bmatrix}$$

These values are then input to the program as below, the values obtained being:

$x_1 = \ 21.4706$

$x_2 = \ \ 0.8235$

$x_3 = \ 12.2941$



```
RUN
NUMBER OF UNKNOWNS? 3
INPUT MATRIX
? 3,8,0
? 0,5,3
? 1,0,-5
INPUT VECTOR
? 71,41,-40
 21.4706
 .8235321
 12.29412
Ok
```

Solution by Numerical Methods

Several variations of the iterative method exist (3).

The Jacobi Method

Suppose we have n linear equations, as before. Each equation in turn is rewritten so as to express each unknown in terms of the others.

Thus the equations of the set 2.9 are rewritten as:

$$x_1 = \frac{b_1 - (a_{12}\ x_2 + \ \ldots .\ a_{1n}x_n)}{a_{11}}$$

$$x_2 = \frac{b_2 - (a_{21}\ x_1 + a_{23}x_3 + \ \ldots .\ a_{2n}x_n)}{a_{22}} \qquad (2.10)$$

$$\vdots$$

$$x_n = \frac{b_n - (a_{11}\ x_1 + \ \ldots .\ a_{nn-1}\ x_{n-1})}{a_{nn}}$$

The process is begun by assuming values for each of the unknowns on the RHS of the above equations. The value of zero is convenient, in which case the first iteration yields the values of the unknowns as:

$x_1 = b_1/a_{11}$; $x_2 = b_2/a_{22}$; $x_n = b_n/a_{nn}$

These values are then inserted on the RHS of the equations, giving a further set of values of x. The process is conninued until the values obtained at successive iterations are considered to be sufficiently close.

EXAMPLE 2.5

Solve the equations of Example 2.4 by the Jacobi method.

First we rewrite the equations as:

$$x_1 = \frac{71 - 8x_2}{3}$$

$$x_2 = \frac{41 - 3x_3}{5}$$

$$x_3 = \frac{40 + x_1}{5}$$

Inserting the value $x_1 = x_2 = x_3 = 0$ into the RHS of each of the above equations, we obtain for the first iteration:

$x_1 = 23.667$

$x_2 = 8.2$

$x_3 = 8$

The second iteration yields:

$$x_1 = \frac{71 - 8 \ast 8.2}{3} = 1.8$$

$$x_2 = \frac{41 - 24}{5} = 3.4$$

$$x_3 = \frac{40 + 23.667}{5} = 12.73$$

The procedure is continued until successive values of the variables agree sufficiently closely. This may take a large number of iterations. Rather than proceed with the manual calculation, a computer calculation will be carried out later. It may be noted here however that 25 iterations were carried out to obtain a satisfactory result namely:

$x_1 = 21.468$

$x_2 = 0.823$

$x_3 = 12.293$

The Gauss-Seidel Method

This is similar to the Jacobi method except the new values of the unknowns as they are generated, are inserted into the following equation.

EXAMPLE 2.6

Solve the equations of Example 2.4 by the Gauss-Seidel method.

We use the equations as rewritten in Example 3.2 and commence with the assumption that $x_1 = x_2 = x_3 = 0$.

The first iteration then yields the following:

$$x_1 = \frac{71 - 8x_2}{3} = \frac{71}{3} = 23.667$$

$$x_2 = \frac{41 - 3x_3}{5} = \frac{41}{5} = 8.2$$

$$x_3 = \frac{40 + x_1}{5} = \frac{40 + 23.667}{5} = 12.73$$

The secod iteration yields:

$$x_1 = \frac{71 - 8 * 8.2}{3} = 1.8$$

$$x_2 = \frac{41 - 3 * 12.73}{5} = 0.562$$

$$x_3 = \frac{40 + 1.8}{5} = 8.36$$

Once again, rather than proceed with the manual calculation, a computer calculation will be carried out. This required 18 iterations to obtain a result closely similar to that obtained by the Jacobi method in 25 iterations.

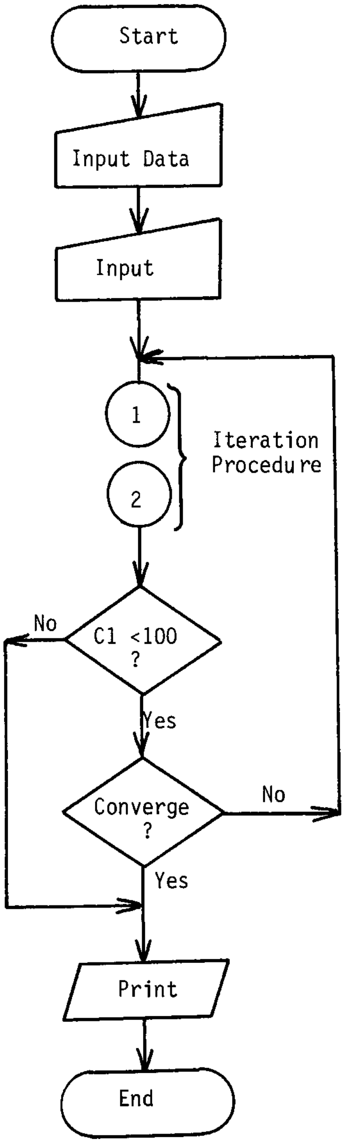
Convergence

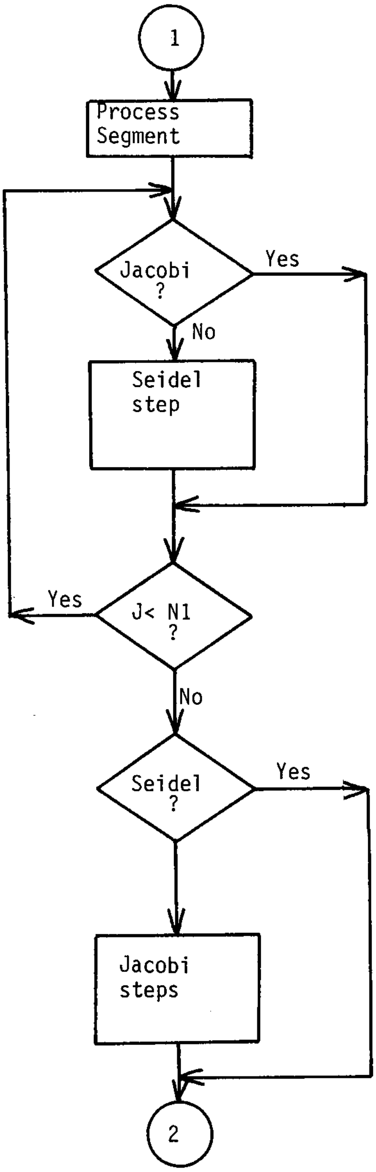
These iterative procedures will not converge in some cases. However, convergence is more likely if the equations to be solved are rearranged so as to present a strong leading diagonal (3). That is, when the equations are written as in equation 2.9, the largest coefficients in successive equations occur in the a_{11}, a_{22}, a_{33},... a_{nn} positions.

This can be done by rearranging the sequence of the equations and the sequence of unknowns within them.

Program DATA5 which follows solves simultaneous equations by these iterative methods.

DATA5.BAS


Start
Input Data
Input
1
2
Iteration Procedure
C1 <100 ?
No
Yes
Converge ?
No
Yes
Print
End


1
Process Segment
Jacobi ?
Yes
No
Seidel step
J< N1 ?
Yes
No
Seidel ?
Yes
Jacobi steps
2

```
10   REM ****************************************************
20   REM - PROGRAM DATA5.BAS
30   REM - SOLUTION OF UP TO 20 SIMULTANEOUS EQUATIONS
40   REM - BY THE ITERATIVE METHODS OF JACOBI & GAUSS-SEIDEL
50   REM - PROGRAM NOMENCLATURE:
60   REM - C(J)        -   Values of the vector (i.e. RHS of
70   REM                   equations 2.9)
80   REM - C1          -   Number of iterations
90   REM - E(J)        -   Values of the unknowns, evaluated
100  REM                   at each iteration
110  REM - I1          -   Skip convergence test on first
120  REM                   iteration
130  REM - M(J,K)      -   Coefficients of the equations
140  REM - N1              Number of unknowns
150  REM - V(J)        -   Values of the unknowns at the
160  REM                   previous iteration (Jacobi), or
170  REM                   Values of the unknowns as modified
180  REM                   at each step of the iteration
190  REM                   (Gauss Seidel)
200  REM - W(J)        -   Values of the unknowns at the
210  REM                   previous iteration (Gauss-Seidel)
220  REM - PROGRAM DESCRIPTION
230  REM - LINE 1000   The arbitrary number of 20 has been
240  REM - assumed for the maximum number of unknowns to
250  REM - be handled
260  REM - LINES 1010 - 1180   Choice of solution method is
270  REM - made, and values of the constants in the equations
280  REM - are entered
290  REM - LINES 1190 - 1470   The iteration proceeds using
300  REM - the form given by equation 2.10, the first assumed
310  REM - value for each unknown being zero.   Values of
320  REM - unknowns calculated at the previous iteration are
330  REM - multiplied by the appropriate coefficient to give
340  REM - values of the products ax etc..   These values are
350  REM - summed and entered into Matrix E (lines 1240,
360  REM - 1260, etc); the sum of these values is then
370  REM - subtracted from the vector value and divided by
380  REM - the appropriate coefficient (line 1380).   This
390  REM - gives the new value of each unknown.   If the
400  REM - Seidel method is being employed, this new value is
410  REM - immediately brought into use (line 1400).   If the
420  REM - Jacobi method is employed, the new values are
430  REM - brought into use only after the iteration is
440  REM - complete (lines 1430 - 1450)
450  REM - LINES 1480 - 1660   A count is kept of the running
460  REM - total of iterations.   If this number reaches
470  REM - 100, the program is terminated, the assumption
480  REM - being that convergence towards the correct
490  REM - solution is not occurring (lines 1480 - 1510).
500  REM - At each iteration new values of the unknowns are
510  REM - compared with previous values; the program
520  REM - terminates if these differ only within the set
530  REM - limits (lines 1520 - 1590).   Finally, the
540  REM - solution and the number of iterations taken, is
550  REM - printed (lines 1620 - 1660).
560  REM ****************************************************
1000 DIM M(20,20),C(20),E(20),V(20),W(20)
1010 PRINT "INPUT J FOR JACOBI METHOD, GS FOR GAUSS-SEIDEL"
1020 INPUT M$
1030 IF M$="J" THEN 1060
1040 IF M$="GS" THEN 1060
```

```
1050 GOTO 1010
1060 INPUT "NUMBER OF UNKNOWNS";N1
1070 FOR J=1 TO N1
1080 PRINT "COEFFICIENTS OF EQUATION ";J
1090 FOR K=1 TO N1-1
1100 INPUT;M(J,K)
1110 NEXT K
1120 INPUT M(J,N1)
1130 NEXT J
1140 PRINT "INPUT VECTOR TERMS"
1150 FOR J=1 TO N1-1
1160 INPUT;C(J)
1170 NEXT J
1180 INPUT C(N1)
1190 REM - ITERATION SEGMENT *******************************
1200 FOR J=1 TO N1
1210 IF M(J,J)=0 THEN 1410
1220 IF J=1 THEN 1290
1230 IF J=2 THEN 1340
1240 E(J)=M(J,1)*V(1)
1250 FOR K=2 TO J-1
1260 E(J)=E(J)+M(J,K)*V(K)
1270 NEXT K
1280 GOTO 1350
1290 E(1)=M(1,2)*V(2)
1300 FOR K=3 TO N1
1310 E(1)=E(1)+M(1,K)*V(K)
1320 NEXT K
1330 GOTO 1380
1340 E(2)=M(2,1)*V(1)
1350 FOR K=J+1 TO N1
1360 E(J)=E(J)+M(J,K)*V(K)
1370 NEXT K
1380 E(J)=(C(J)-E(J))/M(J,J)
1390 IF M$="J" THEN 1410
1400 V(J)=E(J)
1410 NEXT J
1420 IF M$="GS" THEN 1480
1430 FOR J=1 TO N1
1440 V(J)=E(J)
1450 NEXT J
1460 REM - NUMBER OF ITERATIONS AND ************************
1470 REM - TEST FOR CONVERGENCE ****************************
1480 C1=C1+1
1490 IF C1<100 THEN 1520
1500 PRINT "NO CONVERGENCE"
1510 GOTO 1620
1520 IF I1=0 THEN 1600
1530 FOR J=1 TO N1
1540 IF W(J)=0 THEN 1580
1550 IF V(J)/W(J)<1.0001 THEN 1570
1560 GOTO 1580
1570 IF V(J)/W(J)>.9999 THEN 1620
1580 W(J)=V(J)
1590 NEXT J
1600 I1=I1+1
1610 GOTO 1200
1620 PRINT "NUMBER OF ITERATIONS=";C1
1630 FOR J=1 TO N1
1640 PRINT "V(";J;")=";V(J)
1650 NEXT J
```

```
1660 END
```

```
LOAD"A:DATA5
Ok
RUN
INPUT J FOR JACOBI METHOD, GS FOR GAUSS-SEIDEL
? J
NUMBER OF UNKNOWNS? 3
COEFFICIENTS OF EQUATION  1
? 3? 8? 0
COEFFICIENTS OF EQUATION  2
? 0? 5? 3
COEFFICIENTS OF EQUATION  3
? 1? 0? -5
INPUT VECTOR TERMS
? 71? 41? -40
NUMBER OF ITERATIONS= 25
V( 1 )= 21.47083
V( 2 )= .8243408
V( 3 )= 12.29365
Ok
```

```
RUN
INPUT J FOR JACOBI METHOD, GS FOR GAUSS-SEIDEL
? GS
NUMBER OF UNKNOWNS? 3
COEFFICIENTS OF EQUATION  1
? 3? 8? 0
COEFFICIENTS OF EQUATION  2
? 0? 5? 3
COEFFICIENTS OF EQUATION  3
? 1? 0? -5
INPUT VECTOR TERMS
? 71? 41? -40
NUMBER OF ITERATIONS= 18
V( 1 )= 21.46843
V( 2 )= .8235001
V( 3 )= 12.29368
Ok
```

The Method of Residuals

This method was developed and widely used before the advent of computers, as a manual solution method. The technique was known as 'Relaxation' and is more fully described, with an example, in Chapter 7(9).

The equations are arranged with all terms on the L.H.S. Then if correct values of the unknowns are substituted, the sum of these terms will be zero for each equation. Should incorrect values have been chosen, then the terms will not add to zero; this non-zero sum is referred to as a residual.

Thus equations 3.1 are rewritten as:

$$a_{11}x_1 + a_{12}x_2 + \ldots.\ a_{1n}x_n - b_1 = R_1$$

$$a_{21}x_1 + a_{22}x_2 + \ldots.\ a_{2n}x_n - b_2 = R_2$$

$$\vdots$$

$$a_{n1}x_1 + a_{n2}x_2 + \ldots.\ a_{nn}x_n - b_n = R_n$$

where R_1, R_2 etc are the residuals.

The procedure involves progressive alteration of the variables so as to reduce all the residuals to acceptably small values.

PROBLEMS - CHAPTER 2

1. The Wilson procedure for determination of film heat transfer coefficients for fluid flow inside a pipe is based upon two assumptions (10), (11).

a. The outside film coefficient is assumed to be constant, (as for instance if the coefficient is large and relatively invariant, as with condensing steam).

b. Physical properties of the fluid in the tube do not vary appreciably at the test conditions.

In these circumstances, for a series of tests in a given apparatus, the relationships involved can be reduced to the following simple form:

$$\frac{1}{U} = K + \frac{1}{h_1V^{0.8}}$$

where U = overall coefficient of heat transfer;

K = constant;

h_1 = value of inside film heat transfer coefficient at unit fluid velocity;

V = fluid velocity.

Use the linear regression program to deduce h_1 and K from the following data:

U	fluid velocity
W/m²K	m/s
1845	0.899
2073	1.256
2271	2.060
1692	0.872
2725	2.774

2. Rearrange the following set of simultaneous equations so as to provide a strong leading diagonal, then solve them using the program, by the Gauss-Seidel method:

$$\begin{aligned} 2a + b + c + 8d + 9e &= 25 \\ a - 5e &= 8 \\ 4b - 3c + 3.1d &= 0 \\ 2.5b + d + 4e &= -18 \\ a - b + 4c &= 4.6 \end{aligned}$$

Can you modify the program to carry out this rearrangement for you?

3. Below is a table of distribution ratio or K values. Use the interpolation program DATA3.

a. to store K values for CH_4over the entire range of temperatures and pressures listed;

b. to store K values for all 9 paraffins listed, over the entire range of temperatures, at a pressure of 14.7 psia.

VOLATILITY EQUILIBRIUM DISTRIBUTION RATIOS, K = y/x FOR IDEAL SOLUTIONS						
	4.4	37.8	93.3	148.9	204.4	260 °C
	40°	100°	200°	300°	400°	500° °F
Pressure = 14.7 psia (1.01 x 10^5 N/m²)						
CH_4	214	252	276	286	291	296
C_2H_6	22.5	38.5	69.0	94.0	110	124
C_3H_8	4.95	10.0	25.0	41.0	56.0	71.0
i-C_4H_{10}	1.83	4.6	13.7	25.0	38.5	52.0
n-C_4H_{10}	1.19	3.27	10.4	20.5	32.5	46.0
i-C_5H_{12}	0.41	1.40	5.60	12.2	21.8	32.0
n-C_5H_{12}	0.30	1.02	4.26	7.15	14.2	28.3
n-C_6H_{14}	0.091	0.34	1.89	6.00	12.0	19.6
n-C_7H_{16}	0.028	0.112	0.82	3.23	7.5	13.4

Pressure = 50 psia (3.45×10^5 N/m^2)						
CH_4	64.3	76.4	83.8	87.0	89.0	89.5
C_2H_6	6.3	11.4	21.0	28.5	34.4	39.0
C_3H_8	1.5	3.15	7.85	13.1	17.8	22.3
i-C_4H_{10}	0.59	1.45	4.25	8.1	12.0	16.0
n-C_4H_{10}	0.40	1.03	3.24	6.7	10.4	13.9
i-C_5H_{12}	0.14	0.43	1.70	4.05	6.75	9.8
n-C_5H_{12}	0.104	0.322	1.39	3.44	5.85	8.8
n-C_6H_{14}	0.034	0.113	0.605	1.93	3.8	6.0
n-C_7H_{16}	0.011	0.0395	0.27	1.04	2.46	4.3

Pressure = 100 psia (6.9×10^5N/m^2)						
CH_4	32.0	37.8	41.8	43.9	45.0	45.5
C_2H_6	3.4	5.8	10.6	14.8	18.0	20.4
C_3H_8	0.795	1.70	4.17	6.87	9.4	11.8
i-C_4H_{10}	0.31	0.76	2.30	4.35	6.50	8.6
n-C_4H_{10}	0.217	0.545	1.77	3.5	5.55	7.4
i-C_5H_{12}	0.08	0.235	0.94	2.24	3.74	5.4
n-C_5H_{12}	0.058	0.172	0.745	1.89	3.23	4.8
n-C_6H_{14}	0.0197	0.062	0.327	1.05	2.08	3.3
n-C_7H_{16}	0.0067	0.022	0.143	0.575	1.38	2.25

Source: Reprinted with permission from G.G. Brown & Others, Unit Operations, John Wiley & Sons Inc., Copyright 1950 ©

4. Write a program which will store the temperature data of Table 2.2, in addition to the equilibrium values.

5. Write a program applying the least squares method to a polynomial of three terms.

REFERENCES

1 H.S. Mickley, T.K. Sherwood, C.E. Reed, Applied Mathematics in Chemical Engineering, McGraw Hill Book Co., New York, 1957

2 V.G. Jenson, G.V. Jeffreys, Mathematical Methods in Chemical Engineering, Academic Press, London, 1963

3 S.F. Hancock, Mathematics for Engineers, Macdonald & Evans, Plymouth, U.K., 1979

4 B.H. Chirgwin & C. Plumpton, A Course of Mathematics for Engineers and Scientists, Vol. 2., Pergamon Press, Oxford, 1978

5 J.R.F. Alonso, SIMPLE, Basic Programs for Business Applications, Prentice Hall Inc., New Jersey, 1981

6 L. Poole & Others, Some Common BASIC Programs, Osborne/McGraw Hill, 1982

7 M.E. Leesley, Editor, Computer-aided Process Plant Design, Gulf Publishing Co., Houston, Texas, 1982.

8 B.E. Gillett, Introduction to Operations Research; a computer oriented Algorithmic approach, McGraw Hill, New York, 1976

9 G.M. Dusinberre, Heat Transfer Calculations by Finite Differences, International Textbook Co., Scranton Pa, 1961

10 Transactions of the American Society of Mechanical Engineers, 37,47 1915

11 J.M. Coulson, J.F. Richardson & Others, Chemical Engineering, Vol. 1, 3rd Edition, Pergamon Press, Oxford, 1978

Chapter 3

SOLUTION OF MULTIPLE EFFECT EVAPORATOR PROBLEMS

The computer solution to such problems can follow exactly the usual manual method, and involves solution of simultaneous equations, coupled with an iterative procedure (1), (2), (3), (4).

Assumptions: The method to be described rests upon the following assumptions:

all effects are of the same area;
heat losses are negligible;
no carry over of liquid into the vapour phase occurs.

In the example which follows, further simplifications are made:

boiling point elevations are ignored;
only forward feed systems are considered;
thermal recompression is not included.

Obviously, none of these assumptions are fundamental, and they can be omitted if desired. Figure 3.1 shows the arrangement and nomenclature for a forward feed system. Details of specific evaporation problems can be found in specialist texts (5), (6) (7).

Procedure: The following values must be known or calculable:

overall heat transfer coefficient in each effect;
temperature of the heating medium;
pressure and hence temperature of saturated vapour from the last effect (presumed to be governed by conditions at the condenser);
feed flowrate, temperature and concentration;
desired product concentration.

Obviously, enthalpy data and boiling point elevation data (if applicable) must also be employed. The calculation procedure then involves the following steps:

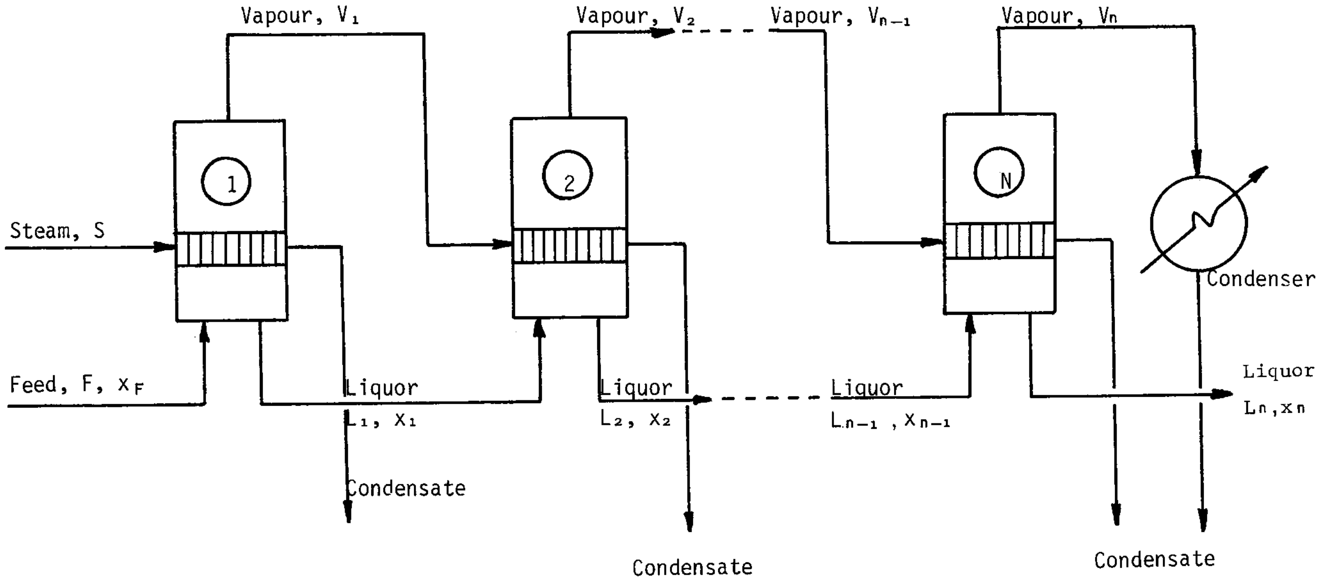
a. For n effects, product flowrate and total evaporation are calculated from a mass balance:

$$Fx_F = L_n x_n \tag{3.1}$$

Knowing x_n , product flowrate L_n can be calculated.

$$F = L_n + V_1 + V_2 \ldots . V_n \tag{3.2}$$

Figure 3.1. Flow diagram for Forward Feed Multiple Effect Evaporator System



Nomenclature: S,F - flowrates of steam, feed
V_1, V_2, ... V_n - vapour flowrates from 1st, 2nd, ... nth effect
L_1, L_2, ... L_n - liquor flowrates from 1st, 2nd, ... nth effect
x_1, x_2, ... x_n - wt. fraction of solute in liquor from 1st, 2nd, ... nth effect
x_F - wt. fraction of solute in feed

Where boiling point rises and enthalpies, dependent on concentration as well as temperature, have to be employed, it is necessary to determine the concentration in each effect. This can be done approximately at this stage, by making the further assumption of equal evaporation in each effect.

In this case

$$V_1 = V_2 = \ldots . V_n$$

$$= \frac{V_1 + V_2 + \ldots . V_n}{n} \qquad (3.3)$$

$F = V_1 + L_1$ Hence L_1 is obtained

$Fx_F = L_1x_1$ Hence x_1 is obtained, and so on for each effect.

b. Initially, the temperature driving force in each effect is evaluated by sharing the overall difference available, amongst the effects in proportion to the individual thermal resistances. Overall available temperature difference,

$$\Delta T_{oa} = T_S - T_n - \Sigma(\Delta T)_R \qquad (3.4)$$

where $(\Delta T)_R$ = boiling point rise in effect

T_S = temperature of the heating medium

T_n = temperature in the final effect. Then temperature driving force in the first effect,

$$\Delta T_1 = \Delta T_{oa} * \frac{1/U_1}{1/U_1 + 1/U_2 + \ldots 1/U_n} \qquad (3.5)$$

where U_1, U_2 , etc. are overall heat transfer coefficients in effects 1, 2, etc.

The values of ΔT_1 etc. and the consequent effect temperatures are adjusted at the second and subsequent iterations.

c. Evaporation rate from each effect is obtained from heat and material balances. An equation can be written for each effect and these simultaneous equations can be solved for the unknowns.

d. Knowing the vapourisation rates, heat loads can now be calculated for each effect, and the required heat transfer areas A_1, A_2, etc, calculated from the rate equation $q = UA\Delta T$.

e. Assuming that effects of equal area are required, (as mentioned before), then the mean area is calculated, and values of temperature driving force in each effect are adjusted:

$$A_m = \frac{A_1 + A_2 + \ldots . A_n}{n} \qquad (3.6)$$

$$\Delta T_1' = \Delta T_1 * A_1/A_m \qquad (3.7)$$

$$\Delta T_2' = \text{etc.}$$

f. The new ΔT values are again adjusted so that their sum =

$T_S - T_n - \Sigma(\Delta T)_R$ i.e.

$$\Delta T_1'' = \Delta T_1' * \frac{(T_S - T_n - \Sigma(\Delta T)_R)}{\Sigma\Delta T'} \qquad (3.8)$$

g. From the new ΔT values new effect temperatures are obtained and fresh calculation of evaporation from each effect is made (step c).

If necessary, new boiling point rise values can be incorporated. Steps c. to f. are repeated until the effect areas are the same within suitable limits.

MANUAL SOLUTION OF A MULTI EFFECT EVAPORATION PROBLEM

EXAMPLE 3.1

Use the method outlined above to solve the following problem:

A solution with a negligible boiling-point rise is evaporated in a triple-effect evaporator, which it enters at 30°C. Saturated steam at 121.8°C (395K) enters the calandria of the first effect. The pressure of the vapor in the last effect is 26 Kpa. 6 Kg/s of solution containing 10% solids enter the system. The product leaves the last effect containing 30% solids. The heat transfer coefficients are $U_1 = 3000$, $U_2 = 2000$, $U_3 = 1500\text{W/m}^2\text{K}$.

Assuming a forward feed system, with effects of equal area, calculate the following:

the temperature in each effect;

the vapourisation from each effect;

the area required in each effect.

Solution

From steam tables, the temperature in the last effect is found to be 65.9°C. The temperature driving force available therefore =

$T_S - T_n$ (since there are no BPt rises), = $121.8 - 65.9 = 55.9^{\circ}\text{C} = \Delta T_{oa}$

This is apportioned between the effects as follows:

$$\Delta T_1 = \Delta T_{oa} * \frac{1/U_1}{1/U_1 + 1/U_2 + 1/U_3} \qquad \text{(equation 3.5)}$$

$$\therefore \Delta T_1 = 55.9 * \frac{1/3000}{1/3000 + 1/2000 + 1/5000} = 12.4^{\circ}\text{C}$$

$$\Delta T_2 = 55.9 * \frac{1/2000}{1/3000 + 1/2000 + 1/1500} = 18.6^{0}C$$

Similarly:

$\Delta T_3 = 55.9 - 12.4 - 18.6 = 24.9^{0}C.$

Hence, the effect temperatures are:

1st effect temperature = 121.8 - 12.4 = $109.4^{0}C$.

2nd effect temperature = 109.4 - 18.6 = $90.8^{0}C$.

3rd effect temperature = 65.9^{0}as already determined.

By an overall material balance, the weight of product and the evaporation effected are calculated:

$Fx_F = L_n x_n$

$6 * 0.1 = L_n * 0.3$. Hence weight of product = $\frac{0.6}{0.3}$ = 2.0 Kg/s.

Total evaporation = 6.0 - 2.0 = 4.0 Kg/s.

Evaporation and heat load in each effect are now calculated using heat and material balances as follows:

1st effect

$S\lambda_s + Fh_F = L_1h_1 + V_1H_1$ but (3.9)

$F = V_1 + L_1 \therefore L_1 = F - V_1$

values of enthalpies obtained from steam tables are:

feed liquid at $30^{0}C$,	h_F	=	125.7KJ/kg;
liquid at $109.4^{0}C$,	h_1	=	458.4KJ/kg;
vapour at $109.4^{0}C$,	H_1	=	2690KJ/kg;
latent heat heating steam,	λ_s	=	2198KJ/kg;

$\therefore 2198S + 6 * 125.7 = (6-V_1) * 458.4 + V_1 * 2690$

$2198S + 6 * (125.7-454.3) = V_1(2690 - 458.4)$

$2198S - 1996 = 2232\ V_1$ (3.9a)

2nd effect

$V_1\ (H_1 - h_1) + L_1h_1 = L_2h_2 + V_2H_2$ but (3.10)

$L_1 = V_2 + L_2 \therefore L_2 = L_1 - V_2 = F - V_1 - V_2$

values of enthalpies obtained from steam tables are:

liquid at 90.8°C, $h_2 = 380.0\text{KJ/kg}$;
vapour at 90.8°C, $H_2 = 2661\text{KJ/kg}$.

$\therefore\ V_1(H_1 - h_1) + (F - V_1)h_1 = (F - V_1 - V_2)h_2 + V_2H_2$

$V_1(H_1 - 2h_1 + h_2) + F\,(h_1 - h_2) = V_2(H_2 - h_2)$

$V_1(2690 - 2 * 458.4 + 380) + 6\,(458.4 - 380) =$

$V_2(2661 - 380)$

$\therefore\ 2153V_1 + 470.4 = 2281\,V_2$ (3.10a)

<u>3rd effect</u>

$V_2(H_2 - h_2) + L_2h_2 = L_3h_3 + V_3H_3$ (3.11)

$L_2 = V_3 + L_3\ \therefore\ L_3 = L_2 - V_3 =$

$F - V_1 - V_2 - V_3$ but

$V_1 + V_2 + V_3 = 4$kg/s as calculated above and
$L_3 = 2$kg/s as calculated above.

values of enthalpies obtained from steam tables are:

liquid at 65°C, $h_3 = 276\text{KJ/kg}$;
vapour at 65°C, $H_3 = 2619\text{KJ/kg}$.

$\therefore\ V_2(H_2 - h_2) + (6 - V_2 - V_2)h_2 = 2h_3 + (4 - V_1 - V_2)H_3$

$V_1(H_3 - h_2) + V_2\,(H_2 - 2h_2 + H_3) =$

$2h_3 + 4H_3 - 6h_2$

$V_1(2619 - 280) + V_2\,(2661 - 2*380 + 2619) =$

$2 * 276 + 4*2619 - 6*380$

$2239V_1 + 4520V_2 = 8748$ (3.11a)

$\therefore\ 2153V_1 = 8412 - 4346V_2$

Substituting into equations 3.10a, hence

$V_2 = \dfrac{8412 + 470.4}{4346 + 2281} = 1.34$ kg/s hence

$V_1 = \dfrac{8412 - 4346 * 1.34}{2153} = 1.202$ kg/s and by difference

$V_3 = 4 - 2.542 = 1.458$ kg/s. From equation 3.9a

$S = \dfrac{2232 * 1.202 + 1996}{2198} = 2.129$ kg/s

Heat loads and required areas in each effect are now readily calculable:

1st effect heat load $= S\lambda_s =$ (3.12)

$2.129 * 2198 = 4.68 * 10^3$ KJ/s

required area =

$$\frac{4.68 * 10^{6}}{3000 * 12.4} = 126m^{2}$$

2nd effect head load = $V_1 (H_1 - h_1)$ =

$1.202 (2690 - 458.4) = 2.682 \times 10^{3}$ KJ/s

required area =

$$\frac{2.682 * 10^{6}}{2000 * 18.6} = 72.1m^{2}$$

3rd effect head load = $V_2 (H_2 - h_2)$ =

$1.34 \ (2661 - 380) = 3.05 \times 10^{3}$ KJ/s

required area =

$$\frac{3.05 * 10^{3}}{1500 * 24.9} = 81.6m^{2}$$

$$A_m = \frac{126 + 72.1 + 81.6}{3} = 93.2m^{2}$$

Corrected values of driving forces and temperatures in the affects are:

$$\Delta T_1 = 12.4 * \frac{126}{93.2} = 16.7^{0}$$

$$\Delta T_2 = 18.6 * \frac{72.4}{93.2} = 14.4^{0}$$

$$\Delta T_3 = 24.9 * \frac{81.6}{93.2} = \underline{21.8^{0}}$$

$$\qquad 52.9^{0}$$

The calculation is now repeated using the following values adjusted to give $\Delta T_{oa} = 55.9^{0}$.

$\Delta T_1 = 17.6^{0}$ $\qquad T_1 = 104.2^{0}$

$\Delta T_2 = 15.2^{0}$ $\qquad T_2 = 89.0^{0}$

$\Delta T_3 = 23.7^{0}$ $\qquad T_3 = 65.9^{0}$

The remainder of the calculation is left as an exercise for the reader; the result may be checked by comparison with the computer calculation described next.

COMPUTER SOLUTION OF MULTI-EFFECT EVAPORATOR PROBLEMS

Program EVAP1 which follows, is based on the assumptions which have already been stated. It is capable of handling any number of effects, the simultaneous equations generated being solved by either matrix algebra, or Seidel iteration.

The equations for the effects are dealt with below:

First effect

$$Fh_F + S\lambda_s = V_1H_1 + L_1h_1$$

Since $L_1 = F - V_1$ this equation can be rearranged as:

$$F_F(h \ - h_1) = - S\lambda_s + V_1 (H_1 - h_1) \qquad (3.15)$$

Intermediate effect

$$L_1h_1 + V_1 (H_1 - h_1) = V_2H_2 + L_2h_2$$

Since $L_2 = L_1 - V_2 = F - V_1 - V_2$ the equation can be rearranged as:

$$F (h_1 - h_2) = V_1 (2h_1 - H_1 - h_2) + V_2 (H_2 - h_2) \qquad (3.16a)$$

Similar heat and mass balances lead to further equations for additional intermediate effects:

Third effect

$$F (h_2 - h_3) = V_1 (h_2 - h_3) + V_2 (2h_2 - H_2 - h_3) + V_3 (H_3 - h_3) \qquad (3.16b)$$

Fourth effect

$$F(h_3 - h_4) = V_1 (h_3 - h_4) + V_2 (h_3 - h_4)$$

$$+ V_3 (2h_3 - H_3 - h_4) + V_4 (H_4 - h_4) \qquad (3.16c)$$

Since enthalpy values are known, the only unknowns in these equations are S; V_1 ; V_2 ; etc.

Table 3.1 shows these equations arranged in matrix form.

Final effect

Suppose that Effect 2 is the last; then using x_P to denote product concentration,

$Fx_F = L_2x_P$. By a heat balance, as before

$L_1h_1 + V_1 (H_1 - h_1) = V_2H_2 + L_2h_2$. But

$L_2 = Fx_F/x_P$ and

$V_2 = L_1 - L_2 = F - V_1 - Fx_F/x_P$

$$\therefore \ (F-V_1)h_1 + V_1 (H_1 - h_1) = (F - V_1 - Fx_F/x_P) H_2 + F \frac{x_F}{x_P} . h_2$$

Rearranging:

$$F \left[h_1 - H_2 + \frac{x_F}{x_P} (H_2 - h_2)\right] = V_1(2h_1 - H_1 - H_2) \qquad (3.17a)$$

Suppose that Effect 3 is the last; then

$Fx_F = L_3x_P$

Similar arguments to those above then lead to the equation:

$$F\left[h_2 - H_3 + \frac{x_F}{x_P} (H_3 - h_3)\right] = V_1(h_2 - H_3) + V_2(2h_2 - H_2 - H_3) \qquad (3.17b)$$

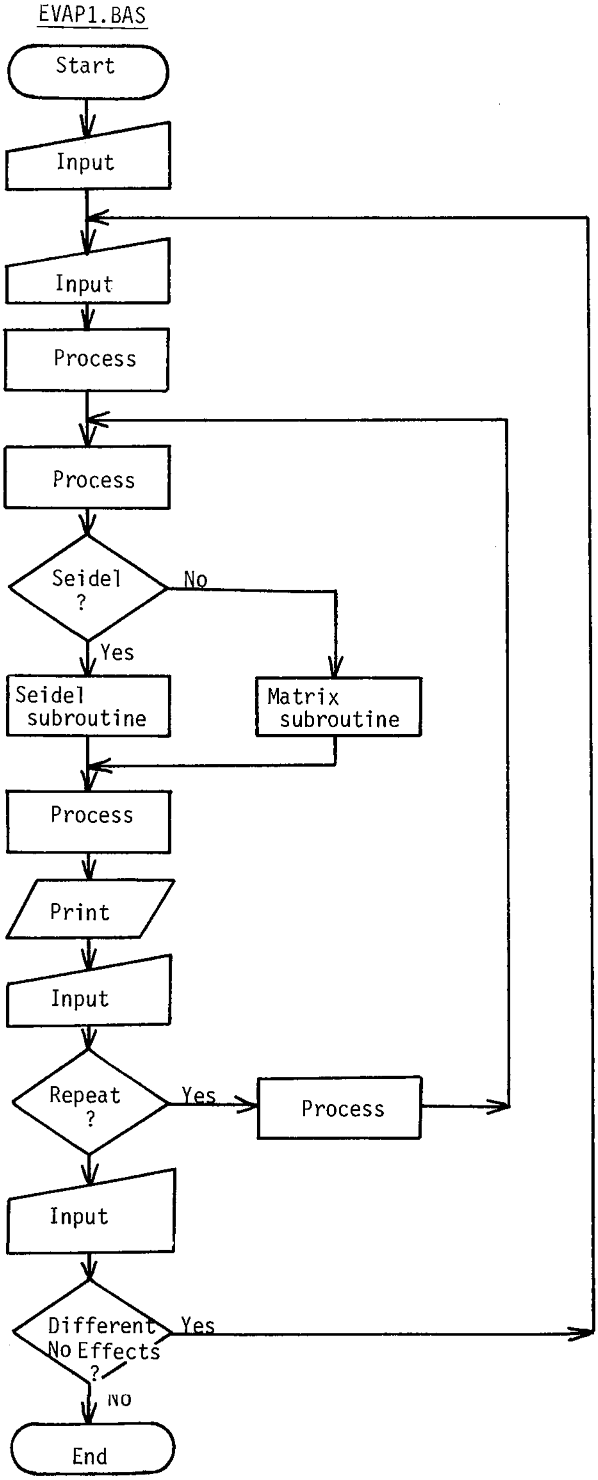
Table 3.2 tabulates in matrix form the various equations that may arise from the heat balance around the final effect. The appropriate values must be incorporated in the n^{th} row of Table 3.1.

TABLE 3.1 Vector and Matrix Values for First & Subsequent Effects, but excluding the Final Effect

	Matrix										Vector
	K = 1	2	3	4	5	...n					
J = 1	$-\lambda s$	$H_1 - h_1$	0	0	0	0		S			$F(h_F - h_1)$
2	0	$2h_1 - H_1 - h_2$	$H_2 - h_2$	0	0	0		V_1			$F(h_1 - h_2)$
3	0	$h_2 - h_3$	$2h_2 - H_2 - h_3$	$H_3 - h_3$	0	0	$*$	V_2	=		$F(h_2 - h_3)$
4	0	$h_3 - h_4$	$h_3 - h_4$	$2h_3 - H_3 - h_4$	$H_4 - h_4$	0		V_3			$F(h_3 - h_4)$
5	0	$h_4 - h_5$	$h_4 - h_5$	$h_4 - h_5$	$2h_4 - H_4 - h_5$	$H_5 - h_5$		V_4			$F(h_4 - h_4)$
⋮											
n-1	0	$h_{n-2}-h_{n-1}$	$h_{n-2}-h_{n-1}$	$h_{n-2}-h_{n-1}$	. . . . . .	$H_{n-1}-h_{n-1}$		V_{n-2}			$F(h_{n-2}-h_{n-1})$

TABLE 3.2 Vector and Matrix Values for the Final Effect. The Values listed are alternatives for the n^{th} row of Table 3.1

	Matrix									Vector
J =	K = 1	2	3	4	5	...n				
n = 2	0	$2h_1 - H_1 - H_2$	0	0	0	0				$F\left[h_1 - H_2 + \frac{X_F}{X_P}(H_2 - h_2)\right]$
n = 3	0	$h_2 - H_3$	$2h_2 - H_2 - H_3$	0	0	0				$F\left[h_2 - H_3 + \frac{X_F}{X_P}(H_3 - h_3)\right]$
n = 4	0	$h_3 - H_4$	$h_3 - H_4$	$2h_3 - H_3 - H_4$	0	0	$*$	V_{n-1}	=	$F\left[h_3 - H_4 + \frac{X_F}{X_P}(H_4 - h_4)\right]$
n = 5	0	$h_4 - H_5$	$h_4 - H_5$	$h_4 - H_5$	$2h_4 - H_4 - H_5$	0				$F\left[h_4 - H_5 + \frac{X_F}{X_P}(H_5 - h_5)\right]$
⋮										
n = n	0	$h_{n-1}-H_n$	$h_{n-1}-H_n$	$h_{n-1}-H_n$	. . . . . .	$2h_{n-1}-H_{n-1}-H_n$				$F\left[h_{n-1}-H_n + \frac{X_F}{X_P}(H_n-h_n)\right]$


EVAP1.BAS
Start
Input
Input
Process
Process
Seidel ?
No
Yes
Seidel subroutine
Matrix subroutine
Process
Print
Input
Repeat ?
Yes
Process
Input
Different No Effects ?
Yes
No
End

```
10  REM ****************************************************
20  REM - PROGRAM EVAP1.BAS  THIS PROGRAM SOLVES
30  REM - FORWARD FEED MULTI-EFFECT EVAPORATOR PROBLEMS
40  REM - UP TO 10 EFFECTS
50  REM - PROGRAM NOMENCLATURE
60  REM - A(J)     -   Heat transfer area, Jth effect, sq m
70  REM - B(J)     -   Intermediate values used in
80  REM                correction of temperature differences
90  REM - D(J)     -   Temperature difference across effect,
100 REM                Deg C
110 REM - H(J)     -   Heat load across effect, KJ/s
120 REM - L(J)     -   Liquid flowrate from effect, kg/s
130 REM - P(J)     -   Vectors for matrix calculations
140 REM - R(J)     -   Thermal resistance, used in
150 REM                calculation of D(J) values
160 REM - T(J)     -   Temperature in effect, DEG C
170 REM - U(J)     -   Overall heat transfer coefficient
180 REM                in effect, W/sq m,K
190 REM - V(J)     -   Vapour rate from effect, kg/s
200 REM - W(J)     -   Previously calculated values of V(J)
210 REM - X(J)     -   Vapour enthalpy. KJ/kg
220 REM - Y(J)     -   Liquid enthalpy, KJ/kg
230 REM - M(J,J)   -   Matrix of enthalpy values
240 REM - N(J,J)   -   Inverse matrix of enthalpy values
250 REM - PROGRAM DESCRIPTION
260 REM - LINES 1000 - 1320   Array dimensions are declared,
270 REM - and choice of solution method is made.   The
280 REM - relevant data concerning flows, temperatures, etc,
290 REM - are then input to the program
300 REM - LINES 1330 - 1450   Driving forces and
310 REM - temperatures in the effects are calculated
320 REM - using equations 3.4 and 3.5
330 REM - LINES 1220, 1460 - 1500   The properties of pure
340 REM - water have been assumed for the process fluid.
350 REM - Enthalpies of saturated liquid and vapour, taken
360 REM - from steam tables over a limited temperature range
370 REM - have been fitted to linear equations using
380 REM - regression analysis (see example 2.1)
390 REM - LINES 1510 - 1560   Calculations are performed for
400 REM - a single effect system, using equations 3.1,3.2,
410 REM - 3.8, and 3.11
420 REM - LINES 1570 - 1600, 2140 - 2560, and 2580 - 2960
430 REM - For two or more effects it is necessary to write
440 REM - general equations for the first, intermediate, and
450 REM - final effects.   The simultaneous linear equations
460 REM - generated are then solved by either matrix algebra
470 REM - or by Seidel iteration.   Subroutines to do this
480 REM - are accessed at lines 1570 - 1600
490 REM - SUBROUTINE FOR SOLUTION OF VAPOUR QUANTITIES BY
500 REM - MATRIX ALGEBRA   LINES 2140 - 2560
510 REM - Lines 2160 to 2190 establish the required matrix
520 REM - dimensions; lines 2200 to 2410 ascribe the matrix
530 REM - values from Tables 3.1 and 3.2 to the matrix M;
540 REM - lines 2420 to 2470 ascribe the vector values from
550 REM - Tables 3.1 and 3.2 to the matrix P.   Solution of
560 REM - the unknowns (steam supply rate S3, vapourisation
570 REM - from each effect V(J)) is obtained by
580 REM - multiplication of the inverse matrix by the vector
590 REM - at lines 2480 and 2490
600 REM - SUBROUTINE FOR SOLUTION OF VAPOUR QUANTITIES BY
610 REM - SEIDEL ITERATION   LINES 2580 - 2960
```

```
620 REM - First a set of duplicate values of V(J) are
630 REM - established as W(J) (lines 2600 to 2660).   If a
640 REM - double effect solution is required, then direct
650 REM - solution is possible, without iteration.   This
660 REM - is done using equation 3.17a (lines 2840 and 2850)
670 REM - and equation 3.15 (line 2860).   Determination of
680 REM - vapour from the last effect is not carried out
690 REM - within the subroutine.
700 REM - For cases involving more than two effects, first
710 REM - effect vapour is calculated using equation 3.16a
720 REM - (lines 2680 and 2690).   Vapour rates from
730 REM - intermediate effects are calculated using
740 REM - equations 3.16b,c etc (lines 2710 to 2770).
750 REM - Vapour from the last but one effect is calculated
760 REM - using equations 3.17b,c etc (lines 2780 to 2830).
770 REM - Steam rate is then calculated, as for the double
780 REM - effect case, at line 2860.   A test is then made
790 REM - to determine whether the values of V(J) and
800 REM - W(J) agree within preset limits.   If they do
810 REM - not, another iteration is carried out (lines
820 REM - 2920 to 2950)
830 REM - LINES 1610 - 1710    Vapour flowrate from the last
840 REM - effect is calculated, then for each effect, liquor
850 REM - flowrate, heat load, and required area for heat
860 REM - transfer, are calculated.
870 REM - LINES 1720 - 1870   Values calculated for each
880 REM - effect are printed.
890 REM - LINES 1880 - 2080   If inspection of the areas
900 REM - calculated for the effects shows these to differ
910 REM - beyond acceptable limits, then the calculation
920 REM - is repeated.   Lines 1930 to 2040 recalculate
930 REM - temperature driving forces using equations 3.6
940 REM - and 3.7;lines 2050 to 2070 correct these new
950 REM - values using equation 3.8
960 REM - LINES 2090 - 2120   Provide an opportunity to
970 REM - rerun using a different number of effects and/or
980 REM - different heat transfer coefficients.
990 REM ****************************************************
1000 DIM  A(10),B(10),D(10),H(10),L(10),M(10,10),N(10,10)
1010 DIM P(10),R(10),T(10),U(10),V(10),X(10),Y(10)
1020 PRINT "SIMULTANEOUS EQUATIONS GENERATED WITHIN THE"
1030 PRINT "PROGRAM ARE SOLVED BY A-MATRIX ALGEBRA,"
1040 INPUT "B-SEIDEL ITERATION.  INPUT A OR B";C$
1050 IF C$="A" THEN 1090
1060 IF C$="B" THEN 1090
1070 GOTO 1020
1080 PRINT
1090 PRINT "FORWARD FEED UP TO 10  EFFECTS"
1100 PRINT "STEAM  TEMP,DEG C";
1110 INPUT S1
1120 PRINT "INLET  LIQUOR TEMP,DEG C";
1130 INPUT T1
1140 PRINT "INLET  LIQUOR CONC,WT FRACN";
1150 INPUT C1
1160 PRINT "FEEDFLOWRATE,KG/S";
1170 INPUT F1
1180 PRINT "OUTLET LIQUOR TEMP,DEG C";
1190 INPUT T2
1200 PRINT "OUTLET LIQUOR CONC,WT FRACN";
1210 INPUT C2
1220 REM-LATENT HEAT OF STEAM ******************************
```

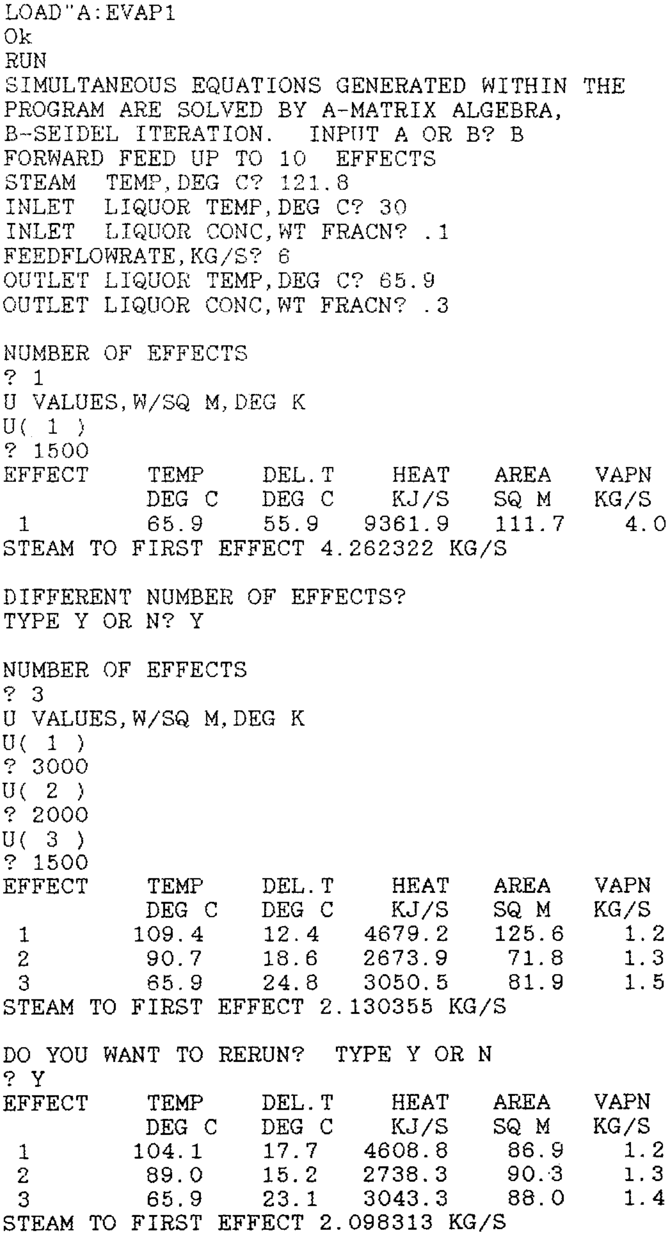
```
1230 S2=2581.74-3.16338*S1
1240 PRINT
1250 PRINT "NUMBER OF EFFECTS"
1260 INPUT N1
1270 IF N1>10 THEN 1240
1280 PRINT "U VALUES,W/SQ M,DEG K"
1290 FOR J=1 TO N1
1300 PRINT "U(";J;")"
1310 INPUT U(J)
1320 NEXT J
1330 REM - WORK OUT TEMPRE DIFFERENCES *********************
1340 REM - IN INVERSE PROPORTION TO U VALUES
1350 R1=0
1360 FOR J=1 TO N1
1370 R1=R1+1/U(J)
1380 NEXT J
1390 FOR J=1 TO N1
1400 D(J)=(S1-T2)/(U(J)*R1)
1410 NEXT J
1420 T(1)=S1-D(1)
1430 FOR J=2 TO N1
1440 T(J)=T(J-1)-D(J)
1450 NEXT J
1460 REM-SATURATED VAPOUR AND LIQUID ENTHALPIES ************
1470 FOR J=1 TO N1
1480 X(J)=2508.3+1.63539*T(J)
1490 Y(J)=4.21119*T(J)-.96387
1500 NEXT J
1510 IF N1>1 THEN 1570
1520 V(1)=F1*(1-C1/C2)
1530 H(1)=V(1)*X(1)+F1*(Y(1)*C1/C2-4.1868*T(1))
1540 A(1)=H(1)*1000/(U(1)*D(1))
1550 S3=H(1)/S2
1560 GOTO 1720
1570 IF C$="B" THEN 1600
1580 GOSUB 2140
1590 GOTO 1610
1600 GOSUB 2580
1610 J=N1
1620 V(J)=F1-V1-F1*C1/C2
1630 L(1)=F1-V(1)
1640 H(1)=L(1)*Y(1)+V(1)*X(1)-F1*4.1868*T1
1650 FOR J=2 TO N1
1660 L(J)=L(J-1)-V(J)
1670 H(J)=L(J)*Y(J)+V(J)*X(J)-L(J-1)*Y(J-1)
1680 NEXT J
1690 FOR J=1 TO N1
1700 A(J)=H(J)*1000/(U(J)*D(J))
1710 NEXT J
1720 PRINT "EFFECT    TEMP    DEL.T    HEAT   AREA   VAPN"
1730 PRINT "          DEG C   DEG C    KJ/S   SQ M   KG/S"
1740 FOR J=1 TO N1
1750 PRINT J;"      ";
1760 PRINT USING "###.#";T(J);
1770 PRINT "   ";
1780 PRINT USING "###.#";D(J);
1790 PRINT "   ";
1800 PRINT USING "####.#";H(J);
1810 PRINT "   ";
1820 PRINT USING "###.#";A(J);
1830 PRINT "  ";
```

```
1840 PRINT USING "###.#";V(J)
1850 NEXT J
1860 PRINT "STEAM TO FIRST EFFECT";S3;"KG/S"
1870 PRINT
1880 IF N1=1 THEN 2090
1890 PRINT "DO YOU WANT TO RERUN?  TYPE Y OR N"
1900 INPUT A$
1910 IF A$="Y" THEN 1930
1920 GOTO 2090
1930 A1=0
1940 FOR J=1 TO N1
1950 A1=A1+A(J)
1960 NEXT J
1970 FOR J=1 TO N1
1980 R(J)=A(J)*N1/A1
1990 NEXT J
2000 D1=0
2010 FOR J=1 TO N1
2020 B(J)=R(J)*D(J)
2030 D1=D1+B(J)
2040 NEXT J
2050 FOR J=1 TO N1
2060 D(J)=B(J)*(S1-T2)/D1
2070 NEXT J
2080 GOTO 1420
2090 PRINT "DIFFERENT NUMBER OF EFFECTS?"
2100 INPUT "TYPE Y OR N";B$
2110 IF B$="Y" THEN 1240
2120 GOTO 2970
2130 REM ****************************************************
2140 REM - SOLUTION OF VAPOUR TO EACH EFFECT
2150 REM - BY MATRIX ALGEBRA
2160 MAT M=ZER(N1,N1)
2170 MAT N=ZER(N1,N1)
2180 MAT P=ZER(N1)
2190 MAT V=ZER(N1)
2200 M(1,1)=-S2
2210 FOR J=1 TO (N1-1)
2220 K=J+1
2230 M(J,K)=X(J)-Y(J)
2240 NEXT J
2250 IF N1=2 THEN 2390
2260 FOR J=2 TO (N1-1)
2270 K=J
2280 M(J,K)=2*Y(J-1)-X(J-1)-Y(J)
2290 NEXT J
2300 FOR J=3 TO (N1-1)
2310 FOR K=2 TO (J-1)
2320 M(J,K)=Y(J-1)-Y(J)
2330 NEXT K
2340 NEXT J
2350 J=N1
2360 FOR K=2 TO J-1
2370 M(J,K)=Y(J-1)-X(J)
2380 NEXT K
2390 J=N1
2400 K=J
2410 M(J,K)=2*Y(J-1)-X(J-1)-X(J)
2420 P(1)=F1*(4.1868*T1-Y(1))
2430 FOR J=2 TO (N1-1)
2440 P(J)=F1*(Y(J-1)-Y(J))
```

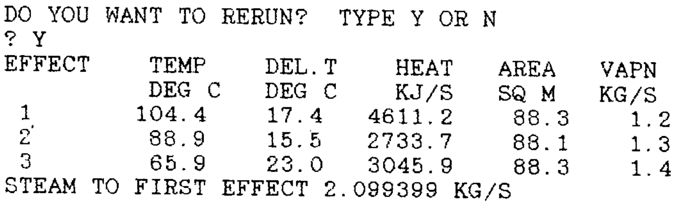
```
2450 NEXT J
2460 J=N1
2470 P(J)=F1*(Y(J-1)-X(J)+(C1/C2)*(X(J)-Y(J)))
2480 MAT N=INV(M)
2490 MAT V=N*P
2500 S3=V(1)
2510 V1=0
2520 FOR J=2 TO N1
2530 V(J-1)=V(J)
2540 V1=V1+V(J-1)
2550 NEXT J
2560 RETURN
2570 REM ***************************************************
2580 REM - SOLUTION OF VAPOUR TO EACH EFFECT
2590 REM - BY SEIDEL ITERATION
2600 FOR J=1 TO N1-1
2610 IF V(J)=0 THEN 2640
2620 W(J)=V(J)
2630 GOTO 2660
2640 V(J)=F1*(1-C1/C2)/N1
2650 W(J)=V(J)
2660 NEXT J
2670 IF N1=2 THEN 2840
2680 V(1)=F1*(Y(1)-Y(2))-V(2)*(X(2)-Y(2))
2690 V(1)=V(1)/(2*Y(1)-X(1)-Y(2))
2700 IF N1=3 THEN 2780
2710 FOR J=2 TO N1-2
2720 V(J)=F1*(Y(J)-Y(J+1))-V(J+1)*(X(J+1)-Y(J+1))
2730 FOR K=1 TO J-1
2740 V(J)=V(J)-V(K)*(Y(J)-Y(J+1))
2750 NEXT K
2760 V(J)=V(J)/(2*Y(J)-X(J)-Y(J+1))
2770 NEXT J
2780 V(N1-1)=F1*(Y(N1-1)-X(N1)+(C1/C2)*(X(N1)-Y(N1)))
2790 FOR K=1 TO N1-2
2800 V(N1-1)=V(N1-1)-V(K)*(Y(N1-1)-X(N1))
2810 NEXT K
2820 V(N1-1)=V(N1-1)/(2*Y(N1-1)-X(N1-1)-X(N1))
2830 GOTO 2860
2840 V(1)=F1*(Y(1)-X(2)+(C1/C2)*(X(2)-Y(2)))
2850 V(1)=V(1)/(2*Y(1)-X(1)-X(2))
2860 S3=(V(1)*(X(1)-Y(1))-F1*(4.1868*T1-Y(1)))/S2
2870 V1=0
2880 FOR J=1 TO N1-1
2890 V1=V1+V(J)
2900 NEXT J
2910 IF N1=2 THEN 2960
2920 FOR J=1 TO N1-1
2930 IF V(J)/W(J)>1.001 THEN 2600
2940 IF W(J)/V(J)>1.001 THEN 2600
2950 NEXT J
2960 RETURN
2970 END
```

EXAMPLE 3.2

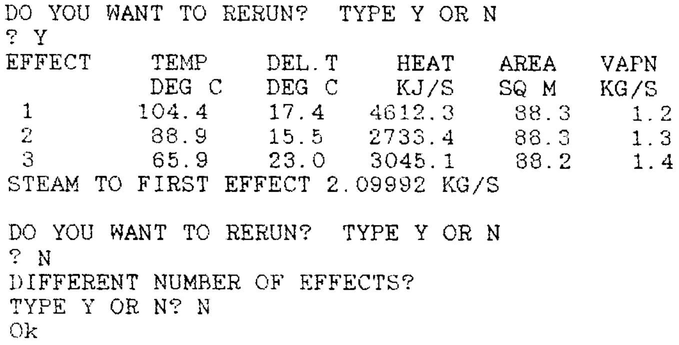
Use the computer program to repeat Example 3.1 It will be seen that the first computer iteration gives results closely similar to those obtained by the manual calculation. After a further two iterations, closely similar values are obtained for the required heat transfer areas.



```
LOAD"A:EVAP1
Ok
RUN
SIMULTANEOUS EQUATIONS GENERATED WITHIN THE
PROGRAM ARE SOLVED BY A-MATRIX ALGEBRA,
B-SEIDEL ITERATION.  INPUT A OR B? B
FORWARD FEED UP TO 10  EFFECTS
STEAM  TEMP,DEG C? 121.8
INLET  LIQUOR TEMP,DEG C? 30
INLET  LIQUOR CONC,WT FRACN? .1
FEEDFLOWRATE,KG/S? 6
OUTLET LIQUOR TEMP,DEG C? 65.9
OUTLET LIQUOR CONC,WT FRACN? .3

NUMBER OF EFFECTS
? 1
U VALUES,W/SQ M,DEG K
U( 1 )
? 1500
EFFECT     TEMP     DEL.T     HEAT    AREA    VAPN
           DEG C    DEG C     KJ/S    SQ M    KG/S
 1         65.9     55.9    9361.9    111.7     4.0
STEAM TO FIRST EFFECT 4.262322 KG/S

DIFFERENT NUMBER OF EFFECTS?
TYPE Y OR N? Y

NUMBER OF EFFECTS
? 3
U VALUES,W/SQ M,DEG K
U( 1 )
? 3000
U( 2 )
? 2000
U( 3 )
? 1500
EFFECT     TEMP     DEL.T     HEAT    AREA    VAPN
           DEG C    DEG C     KJ/S    SQ M    KG/S
 1        109.4     12.4    4679.2    125.6     1.2
 2         90.7     18.6    2673.9     71.8     1.3
 3         65.9     24.8    3050.5     81.9     1.5
STEAM TO FIRST EFFECT 2.130355 KG/S

DO YOU WANT TO RERUN?  TYPE Y OR N
? Y
EFFECT     TEMP     DEL.T     HEAT    AREA    VAPN
           DEG C    DEG C     KJ/S    SQ M    KG/S
 1        104.1     17.7    4608.8     86.9     1.2
 2         89.0     15.2    2738.3     90.3     1.3
 3         65.9     23.1    3043.3     88.0     1.4
STEAM TO FIRST EFFECT 2.098313 KG/S
```



```
DO YOU WANT TO RERUN?  TYPE Y OR N
? Y
EFFECT    TEMP     DEL.T     HEAT    AREA    VAPN
          DEG C    DEG C     KJ/S    SQ M    KG/S
 1        104.4    17.4    4611.2     88.3    1.2
 2         88.9    15.5    2733.7     88.1    1.3
 3         65.9    23.0    3045.9     88.3    1.4
STEAM TO FIRST EFFECT 2.099399 KG/S
```



```
DO YOU WANT TO RERUN?  TYPE Y OR N
? Y
EFFECT     TEMP     DEL.T     HEAT    AREA    VAPN
           DEG C    DEG C     KJ/S    SQ M    KG/S
 1         104.4    17.4    4612.3     88.3    1.2
 2          88.9    15.5    2733.4     88.3    1.3
 3          65.9    23.0    3045.1     88.2    1.4
STEAM TO FIRST EFFECT 2.09992 KG/S

DO YOU WANT TO RERUN?  TYPE Y OR N
? N
DIFFERENT NUMBER OF EFFECTS?
TYPE Y OR N? N
Ok
```

PROBLEMS - CHAPTER 3

1. A forward feed evaporator system has three effects each of 80m^2 heat transfer surface. Heat transfer coefficients in successive effects are 3000, 2500 and 2000 W/m^{2}K. Steam enters the calandria of the first effect at a presure of 320kPa; the pressure above the liquor in the final effect is 18kPa. The feed enters effect one at a concentration of 3 wt% solids and a temperature of 40^{0}C; product leaves the final effect at a concentration of 32 wt% solids.

Use the program to determine the feed and final product rates, assuming the solution to have the properties of water.

Hint: Assume a value for feedrate and use this to run the program; scale this figure up or down as required to satisfy the area requirement.

2. An evaporator system is to be installed for the concentration of an aqueous solution of an organic material. It is desired to make an estimate of the number of effects which should be employed in order to give the lowest annual total costs (operating plus fixed costs).

Pilot plant work has established that the overall coefficient of heat transfer in the evaporator, using low pressure steam for heating, is correlated approximately by the following simple relationship:

U = 2500 - 2000C W/m^{2}K where C = solution concentration, wt fraction.

Assuming a forward feed system is to be employed, use the program, and the data below, to estimate the optimum number of effects to be used:

Feed rate of solution	: 5.5 kg/s
Feed concentration	: 0.05 wt fraction
Feed temperature	: 18.0^{o}C
Product concentration	: 0.5 wt fraction
Physical properties of solution	: As for water
Steam temperature	: 140^{o}C
Final effect temperature	: 60^{o}C
Heating costs	: 5c/KWh
Capital cost for equipment	: \$5,000 per m^{2} of heat transfer surface installed
Interest payable on capital	: 15% p.a.
Maintenance costs and overheads	: 20% p.a. on installed capital
Additional operating costs incurred	: \$25,000 p.a.

What would be the effect on your appraisal if energy costs were to be doubled?

Note: The figures quoted above are conjectural only; they should not be taken as in any way typifying the industry.

3. Modify the program described in this chapter to handle:

   a. Backward feed systems.

   b. Crossflow systems.

   c. Process fluids exhibiting boiling point rise.

   d. Enthalpy data such as that discussed in Example 2.3.

   e. Computation of film coefficients as functions of physical properties.

4. Write an evaporator program using the Newton Raphson method (8).

REFERENCES

1 C.J. Geankoplis, Transport Processes and Unit Operations, Allyn and Bacon, Boston, U.S.A. 1978.

2 C.O. Bennett, J.E. Myers, Momentum Heat and Mass Transfer, McGraw-Hill International, Tokyo, Japan, 1983.

3 D. Azbel, Heat Transfer Applications in Process Engineering, Noyes Publications, New Jersey, U.S.A., 1984.

4 J.M. Coulson, J.F. Richardson, Chemical Engineering, Volume II, 3rd Edition, Pergamon Press, Oxford, 1978.

5 A. Nisenfeld, Industrial Evaporators, Principles of Operation & Control, Instrument Society of America, 1985.

6 F. Bosnjakovic, P.L. Blackshear, Technical Thermodynamics, Holt, Rinehart and Winston, Eastbourne, U.K., 1965.

7 G.P. Meade, Cane Sugar Handbook, 10th Edition, John Wiley & Sons, New York, U.S.A., 1977.

8 C.D. Holland, Fundamentals and Modelling of Separation Processes, Prentice Hall International, New Jersey, U.S.A., 1975.

Chapter 4

SOLUTION OF DISTILLATION PROBLEMS

The example given here, is based upon the well-known McCabe Thiele Graphical Solution method for binary separations (1), (2), (3). Sophisticated computer methods of solution have of course been developed (4), (5), (6), (7).

McCabe Thiele Graphical Method

Assumption: The method to be described rests upon the following assumptions:

constant molal overflow;
equilibrium is attained at each stage.

In the example which follows it is further assumed that:

the column consists of both stripping and enriching sections;
the feed enters between these sections;
all overhead vapour from the column is condensed and either withdrawn as product (distillate) or returned to the column (reflux).

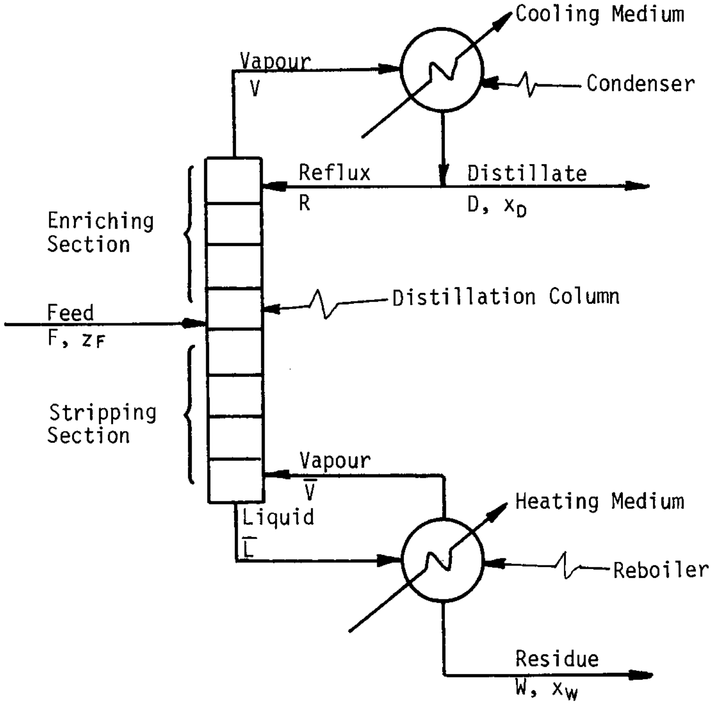
For complete information on the method the reader should refer to the basic texts listed above (1), (2), (3). However, an outline is given below, using the nomenclature of Figure 4.1.

Procedure

The following data must be available:

Feed flowrate, enthalpy and composition;
desired product concentrations and/or flowrates;
vapour/liquid equilibrium data at the chosen operating pressure.

Various methods for the calculation of equilibrium data are available (4), (8), (9), (10). The equilibrium data is first plotted as a graph of mol fraction of more volatile component in the vapour phase (values of y, plotted on the ordinate) versus mol fraction of more volatile component in the liquid phase (values of x, plotted on the abscissa). A graphical construction is then carried out. The procedure involves the construction on the diagram of the operating lines, the equations for which are derived as follows:



Nomenclature:

D,F,R - flowrates of distillate, feed, residue
L, V - flowrates of liquid, vapour from plate to plate, above the feed plate
$\bar{L},\bar{V}$ - flowrates of liquid and vapour below the feed plate
x_D, x_W, z_F - mol. fraction of more volatile component in distillate, residue, feed.

Figure 4.1. Distillation Column Flow Diagram.

Equation of the Upper Operating Line (Enriching Line)

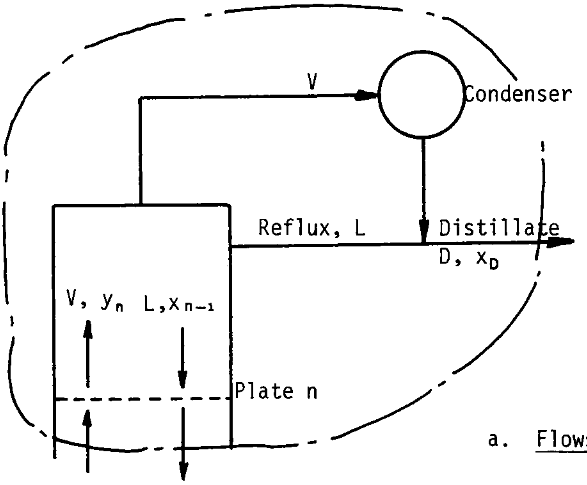
See Figure 4.2a. The plates or equilibrium stages are assumed to be numbered counting from the top downwards. A mass balance on a molar basis, over the top section of the column down to a typical stage n (above the feedplate) yields the equations:

$$V = L + D \quad \text{(by an overall balance)} \tag{4.1}$$

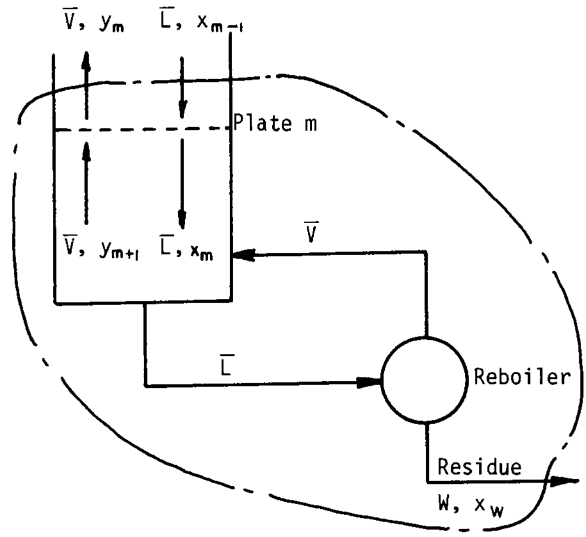
$$Vy_{n+1} = Lx_n + Dx_D \quad \text{(by a balance for the more volatile component)} \tag{4.2}$$

Calling the ratio L/D the reflux ratio R,

$$\therefore\ y_{n+1} = \frac{R}{R+1} \cdot x_n + \frac{x_D}{R+1} \tag{4.3}$$



a. Flows in the Enriching Section



b. Flows in the Stripping Section

Figure 4.2.

The assumption of constant molal overflow permits this equation to represent any equilibrium stage above the feedplate. The equation is that of a straight line on the graph of x, y values, known as the Upper Operating Line or Enriching Line.

Equation of the Lower Operating Line (Stripping Line)

See Figure 4.2b, then

$$\bar{V} = \bar{L} - W \quad \text{(by an overall balance)} \tag{4.4}$$

$$\bar{V}y_{m+1} + Wx_w = \bar{L}x_m \quad \text{(by a balance for the more volatile component)} \tag{4.5}$$

$$\therefore \quad y_{m+1} = \frac{\bar{L}}{\bar{V}} \cdot x_m - \frac{W}{\bar{V}} \cdot x_w \tag{4.6}$$

This equation is that of a straight line, known as the Lower Operating Line or Stripping Line.

Equation of the q - line

The locus of intersection of the Upper and Lower Operating Lines is required; it can be shown that this locus is also a straight line, the equation for which is:

$$y = \frac{q}{q-1} \cdot x - \frac{z_F}{q-1} \quad \text{where } q = \tag{4.7}$$

$$\frac{H_V - H_F}{H_V - H_L} \quad \text{and} \tag{4.8}$$

H_V ,H_L = enthalpies of vapour, liquid leaving any plate;

H_F = enthalpy of the feed.

For the derivation of the q-line, the reader should refer to a standard text (1), (2), (3).

Manual Solution of Binary Distillation Problem

EXAMPLE 4.1

Use the McCabe Thiele method to solve the following problem:

A mixture containing 35 wt% heptane and 65 wt% octane is to be fractionated so as to produce a distillate containing 97 wt% heptane, and a residue containing not more than 4 wt% heptane. The operating pressure is 1.3 x 10^5 N/m^2 and 1.5 kg/s of feed enters as a liquid at 40°C.

(i) Calculate the production rates of distillate and residue.

(ii) Determine the number of plates required at total reflux and the minimum reflux ratio required.

(iii) Specify the appropriate reflux ratio and number of ideal stages which should be provided.

The enthalpy and equilibrium data of Table 4.1 should be used.

TABLE 4.1

Properties of Heptane/Octane Mixtures
Equilibrium Data at 1.3 x 10^5 N/m^2

Mol fraction of heptane	
Liquid Phase	Vapour Phase
x	y
1.00	1.00
0.95	0.98
0.89	0.96
0.81	0.90
0.61	0.77
0.44	0.63
0.33	0.51
0.28	0.46
0.23	0.39
0.11	0.20
0.097	0.18
0.067	0.13
0.039	0.078
0.012	0.025

Enthalpy data:

Feed liquid at 40°C, H_F	=	87.1 KJ/kg
Feed liquid at bubble point, H_L	=	280.3 KJ/kg
Feed vapour at dew point, H_V	=	581.8 KJ/kg

Solution

(a) First the equilibrium data is plotted (Figure 4.3).

(b) Next the given compositions of feed, distillate and residue are converted to a mol fraction basis:

Molecular weight of heptane, C_7H_{16}= 100

Molecular weight of octane, $C_8 H_{18}$= 114

Using a basic weight of 100 kg for each of the three streams:

Stream	kg	kg mols	mol fraction	Mean mol wt
Feed	C_7 = 35	0.35	0.38	
(x_F)	C_8 = 65	0.57	0.62	
		$\sum$0.92	1.00	108.7
Distillate	C_7 = 97	0.97	0.973	
(x_D)	C_8 = 3	0.026	0.027	
		$\sum$0.996	1.000	100.4
Residue	C_7 = 4	0.04	0.045	
(x_W)	C_8 = 96	0.842	0.955	
		$\sum$0.882	1.000	113.4

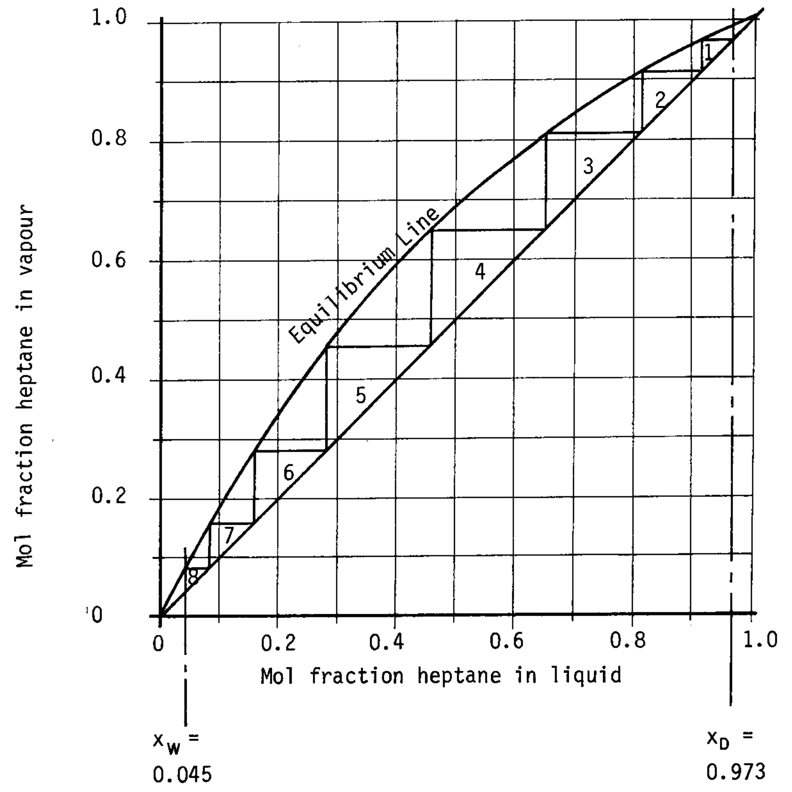


Figure 4.3. Distillation of Heptane/Octane Mixture at Total Reflux.

(c) Quantities of distillate and residue are calculated from a mass balance:

$F = 1.5$ kg/s $= D + W$ (from an overall balance) (4.9)

$F z_F = Dx_D + Wx_W$ (component balance)

$1.5 * 0.35 = 0.97D + 0.04W$

$\therefore\ 0.541 = D + 0.0412$ (4.10)

Subtracting 4.10 from 4.9 and solving for W

W = 1.0 kg/s (.00882 mol/s)
D = 0.5 kg/s (.00498 mol/s)
F = 1.5 kg/s (.1380 mol/s)

(d) The values of x_D and x_W in mol fractions are entered on the graph, and by a stepwise construction the number of plates required at total reflux is found to be 8 (Figure 4.3).

(e) $$q = \frac{H_V - H_F}{H_V - H_L} = \frac{581.8 - 87.1}{581.8 - 280.3} = 1.64$$

The slope of the q-line =

$$\frac{q}{q-1} = \frac{1.64}{0.64} = 2.56$$

A line is drawn through $z_F = 0.38$ to intersect the diagonal. Through this point of intersection the q-line is drawn at a slope of 2.56 (see Figure 4.4).

A line is drawn through $x_D = 0.973$ to intersect the diagonal. Through this point of intersection a line is drawn to meet the q-line where the latter cuts the equilibrium line. This line is the upper operating line, or enriching line, and in this position it corresponds to the condition of minimum reflux.

The enriching line cuts the y-axis at the point

$$y = \frac{x_D}{R+1} = 0.39$$ (Equation 4.3) Hence

$$R_{min} = \frac{0.973}{0.39} - 1 = 1.49$$

The value of R can also be determined from the slope of the operating line. This is more convenient for use with the program which follows.

By geometry, the slope

$$\frac{R}{R+1} = \frac{x_D - y_q}{x_D - x_q}$$

where x_q and y_q are the coordinates of the point of intersection of the q-line and the equilibrium curve.

These points are shown on Figure 4.4, and more clearly on Figure 4.7.

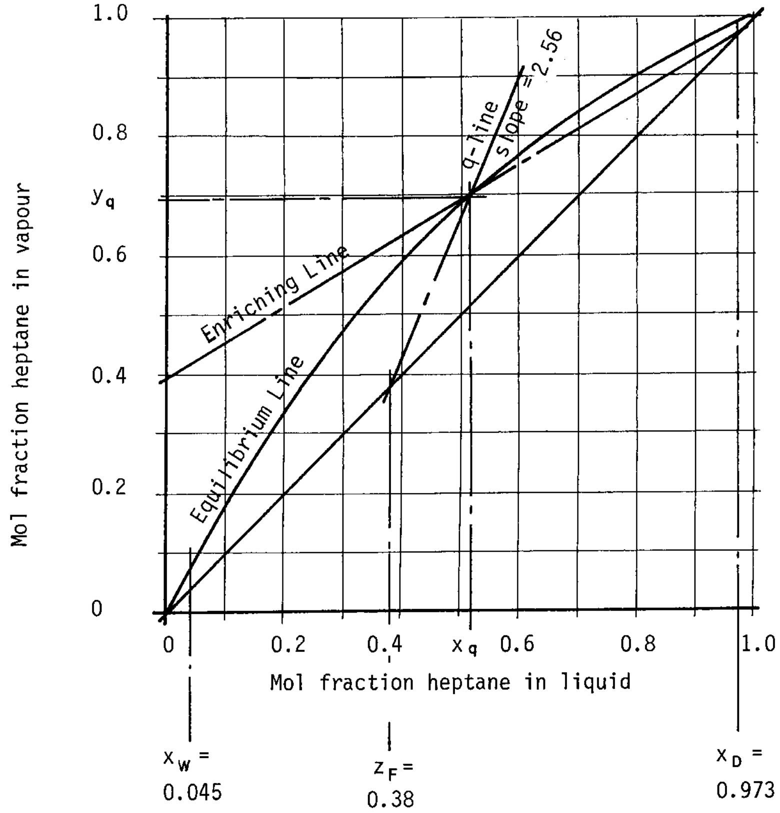


Figure 4.4. Distillation of Heptane/Octane Mixture
Minimum Reflux Condition with Cold (40°C) Feed

(f) A reflux ratio of between 1.2 and 1.5 times the minimum is usually recommended (1), (2), (3). Choosing a value of 1.4 for this case, with a cold feed, a reflux ratio of 1.4 ✶ 1.49 ≃ 2.1 will be chosen for the calculation.

It is usually recommended that an economic optimum number of stages is twice that required at total reflux, i.e. we anticipate that the calculated number of stages should be approximately 16.

The construction is shown in Figure 4.5. Verticals are drawn through the values of x_W, z_F and x_D, to intersect the diagonal. The q-line is drawn in at the slope of 2.56. The upper operating line is drawn, intercept being

$$\frac{x_D}{R+1} = \frac{0.973}{2.1+1} = 0.31$$

The lower operating line is then drawn, as shown.

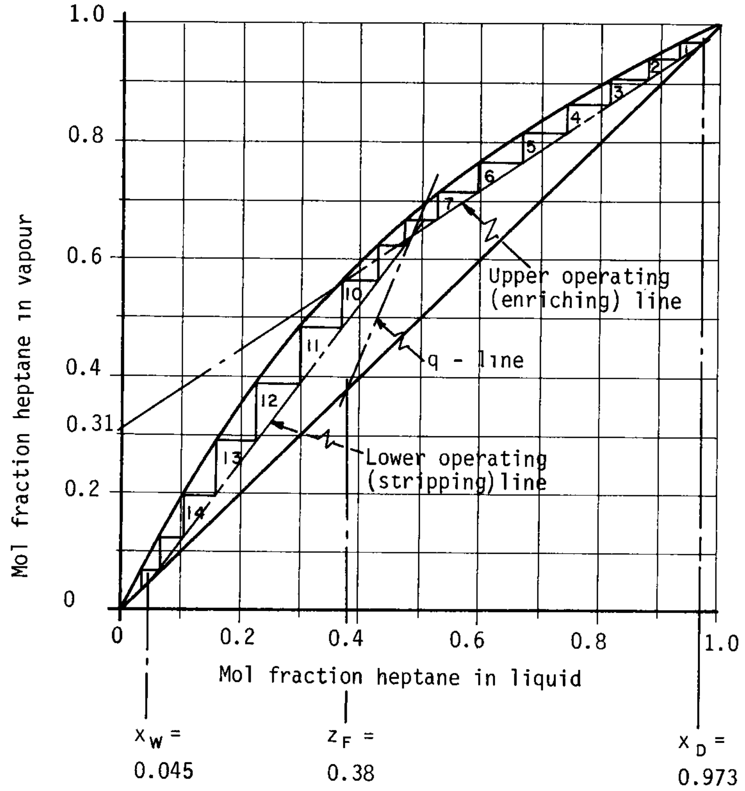


Figure 4.5. Distillation of Heptane/Octane Mixture at a Reflux Ration of 2.1.

The number of ideal stages computed by this technique is seen to be about 15½, the last stage being the reboiler. Plate Number 8 should be the feed plate.

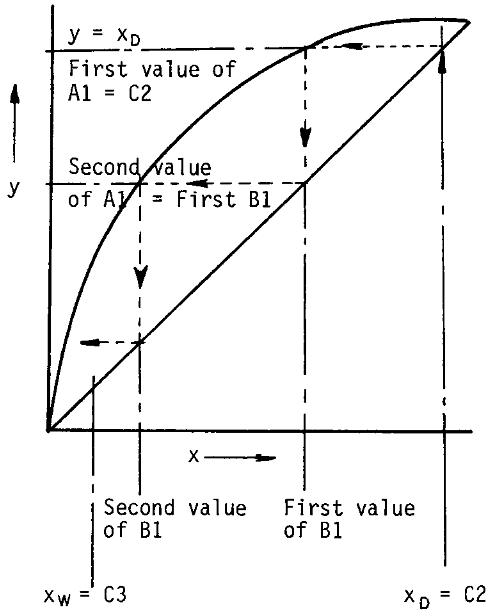


Figure 4.6. Sequence used in computing number of plates required at Total Reflux.

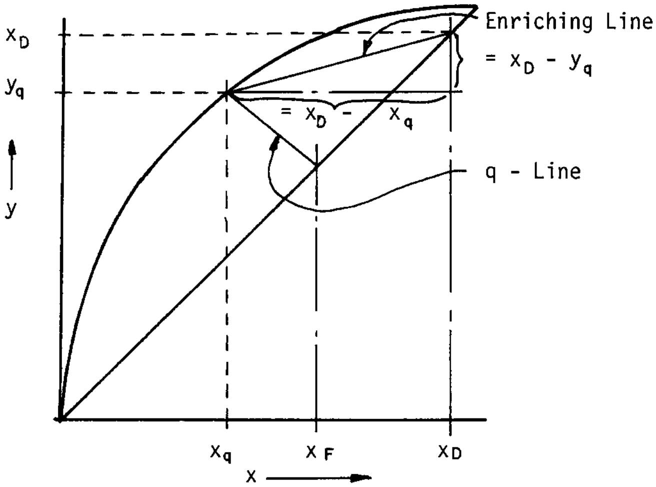


Figure 4.7. Determination of minimum reflux ratio from the slope of the Enriching Line.

COMPUTER SOLUTION OF A BINARY DISTILLATION PROBLEM BY THE McCABE-THIELE METHOD

This method utilises the equations of Lewis and Sorel which when drawn upon the y - x diagram of vapour-liquid concentrations, are the operating lines of the McCabe-Thiele method described above.

The equilibrium data required is input as a series of paired values of x and y. These data are then interpolated using the method of Lagrange detailed in Chapter 2. This technique is unreliable if the number of data pairs is inadequate, and it is advantageous to have other methods available, such as those given in references already quoted (4), (8), (9), (10). The following simple method which assumes ideal behaviour has been included in the program:

The Relative Volatility

$$\alpha = \frac{y_1/y_2}{x_1/x_2} \qquad (\ 411)$$

where y_1, y_2 etc. are mol fractions of component 1, 2, etc. in the vapour phase; x_1, x_2 etc. are mol fractions of component 1, 2, etc. in the liquid phase and equilibrium is assumed.

For a binary mixture, this relationship may be written as α

$$= \frac{y_1/(1 - y_1)}{x_1/(1 - x_1)} \qquad (4.12)$$

where x_1, y_1 are the mol fractions of the more volatile component in the liquid and vapour phases respectively.

In other words, each pair of values of x and y permits a value of α to be calculated. Over limited temperature ranges these values may not vary widely, and a geometric mean may be taken.

From a mean value of α and a given value of y, the corresponding value of x may be calculated by rearrangement of the above equation: Since

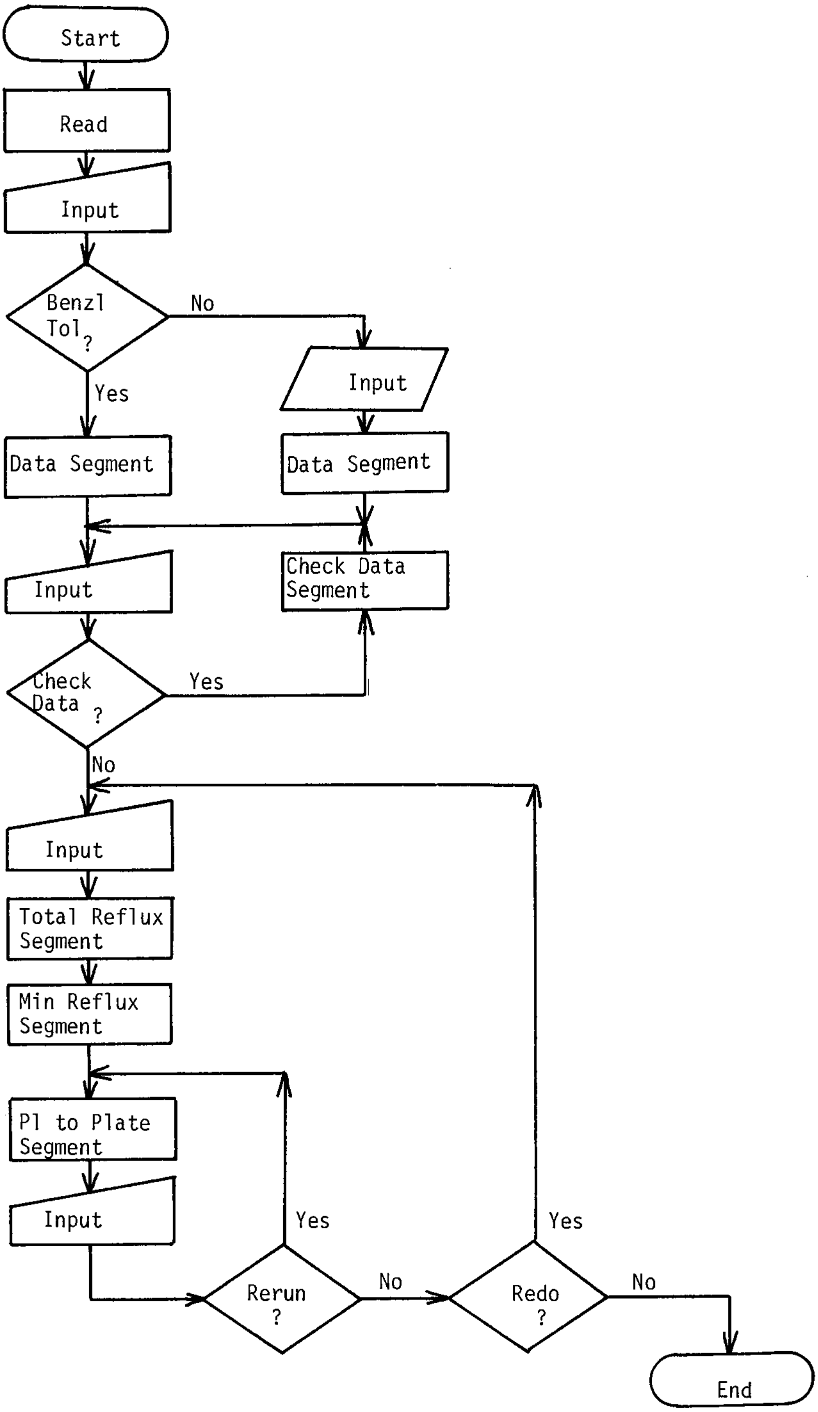
$$\alpha = \frac{y_1(1 - x_1)}{x_1(1 - y_1)}$$

$$\therefore\ \alpha x_1 - \alpha x_1 y_1 = y_1 - y_1 x_1$$

$$\therefore\ x_1 = \frac{y_1}{\alpha(1 - y_1) + y_1} \qquad (4.13)$$

Equilibrium data for the system Benzene/Toluene is included with the program; this was employed during program development.

DIST1.BAS


Start
Read
Input
Benzl Tol ?
No
Yes
Input
Data Segment
Data Segment
Input
Check Data Segment
Check Data ?
Yes
No
Input
Total Reflux Segment
Min Reflux Segment
Pl to Plate Segment
Input
Yes
Yes
Rerun ?
No
Redo ?
No
End

```
10  REM ****************************************************
20  REM - PROGRAM DIST1.BAS
30  REM - MC CABE THIELE DISTILLATION
40  REM - PROGRAM NOMENCLATURE
50  REM - A(J),B(J)  -   Values of vapour, liquid
60  REM                  compositions at equilibrium, input
70  REM                  from the terminal
80  REM - A1         -   Value input to either the inter-
90  REM                  polation or the relative
100 REM                  volatility subroutine
110 REM - B1         -   Value obtained from either of
120 REM                  the above subroutines
130 REM - C1,C2,C3   -   Feed, overhead and bottom
140 REM                  compositions in mol fraction
150 REM - D(J)       -   Used in the Lagrange interpolation
160 REM              -   subroutine described in Chapter 2
170 REM - F1,D1,W1   -   Feed, overhead and bottom
180 REM                  flowrates
190 REM - J1,J2      -   Tag the data pairs inappropriate
200 REM                  for calculation of rel. volatility
210 REM - K1         -   Increment in iteration procedure
220 REM - M(J,K)     -   Used in the Lagrange interpolation
230 REM - N(J)       -   subroutine described in Chapter 2
240 REM - N1         -   Number of data pairs
250 REM - P1         -   Number of plates at total reflux
260 REM - Q1         -   Value of the enthalpy ratio q
270 REM - Q2         -   Value of liquid composition at the
280 REM                  intersection of the q-line with the
290 REM                  enriching line
300 REM - R1         -   Reflux ratio
310 REM - R2         -   Slope of the enriching line
320 REM - V(J)       -   Values of relative volatility
330 REM - V1         -   Used in calculation of mean value
340 REM                  of relative volatility
350 REM - V2         -   Mean value of relative volatility
360 REM - X(G),Y(G)  -   Values of liquid,vapour,
370 REM                  compositions on the Gth plate
380 REM - Y2         -   Vapour composition
390 REM - Z1         -   Number of data pairs inappropriate
400 REM                  for calculation of rel. volatility
410 REM - PROGRAM DESCRIPTION
420 REM - LINES 2000 - 2010   Maximum array dimensions are
430 REM - declared;the arbitrary value of 20 has been
440 REM - selected for the number of data pairs, and 100
450 REM - for the maximum number of plates
460 REM - LINES 2020 - 2190   Equilibrium data for benzene/
470 REM - toluene is read from the data statements 2070
480 REM - & 2110, into matrices A & B.  Line 2170 calls the
490 REM - subroutine which uses these values to calculate
500 REM - denominator terms of the Lagrange interpolation
510 REM - described in Chapter 2.  Line 2180 calls for the
520 REM - subroutine which calculates relative volatilities
530 REM - LINES 2200 - 2310   As an alternative to the data
540 REM - for benzene/toluene, equilibrium data values are
550 REM - input to matrices A & B from the keyboard.
560 REM - LINES 2320 - 2450   Intermediate equilibrium
570 REM - values are checked if desired.
580 REM - LINES 2460 - 2470   Choice is made between the
590 REM - interpolation and relative volatility methods
600 REM - LINES 2480 - 2500   Composition values in mol
610 REM - fractions are entered
```

```
620 REM - LINES 2510 - 2700  The number of plates required
630 REM - at total reflux is computed in a manner analogous
640 REM - to that described for the manual calculation, and
650 REM - shown in Figure 4.6  For this case the operating
660 REM - lines lie on the diagonal.  Distillate
670 REM - composition (C2) is taken as a starting point, its
680 REM - value being equal to the vapour composition from
690 REM - the top plate (A1).  Using the x,y data input to
700 REM - the program, the value of A1 is used to evaluate
710 REM - the corresponding value of x (B1).  Since this
720 REM - value lies on the diagonal, it is equal to the
730 REM - value of vapour composition y from the next plate
740 REM - (A1).  This new value of A1 is used to compute
750 REM - its corresponding value (B1), and so on until
760 REM - the value of residue composition (C3)has been
770 REM - reached.  See Figure 4.6
780 REM - The minimum reflux ratio is
790 REM - now to be computed:
800 REM - LINES 2720 - 2750  A value of q (Q1)
810 REM - is input from the terminal, and depending on
820 REM - its value, various alternatives are followed.
830 REM - LINES 2760 - 2930  For values of Q1 other than
840 REM - zero or unity, values of vapour composition (Y2)
850 REM - are taken successively, starting from distillate
860 REM - composition (C2), stepwise in decrements of 0.1
870 REM - mol fraction (K1).  Each time this is done the
880 REM - value is ascribed to A1 and the corresponding
890 REM - value of liquid composition x (B1) is calculated
900 REM - using either the interpolation or relative
910 REM - volatility method (lines 2800 - 2840).
920 REM - A value of x (B2) is also
930 REM - calculated from the equation of the q-line
940 REM - (Equation 4.7), rearranged as line 2850.  If
950 REM - values B1 and B2 are the same, the point of
960 REM - intersection of the q-line and the equilibrium
970 REM - curve has been obtained.  This is tested at lines
980 REM - 2860 and 2900.  If B2 exceeds B1 but the values
990 REM - do not agree within 1% then the decrement (K1) is
1000 REM - reduced to one tenth of its previous value and
1010 REM - the search is continued (lines 2910 - 2930).
1020 REM - When agreement between values B1 and B2 has been
1030 REM - obtained within 1%, the arithmetic mean of these
1040 REM - values is ascribed to variable Q2 (line 2940)
1050 REM - LINES 2960 - 3020  For the case of a saturated
1060 REM - vapour, q=0 and the slope of the q-line=0.  In
1070 REM - this case, the q-line is horizontal through the
1080 REM - intersection of feed composition (C1) with the
1090 REM - diagonal.  The value is thus also the value of
1100 REM - y at the intersection of the q-line and the
1110 REM - equilibrium lines.  This value of y is ascribed
1120 REM - to A1, and the interpolation routine used to
1130 REM - obtain the corresponding value of xq (Q2)
1140 REM - LINES 3030 - 3170  For the case of a saturated
1150 REM - liquid, q=1 and the slope of the q-line =
1160 REM - infinity.  In this case, the q-line is a
1170 REM - vertical through the intersection of feed
1180 REM - composition with the diagonal.  This value is
1190 REM - thus also the required value of xq (Q2)
1200 REM - LINES 3180 - 3250  The minimum reflux ratio is
1210 REM - calculated from the slope of the enriching line
1220 REM - (Equation 4.3) as discussed in the text.
```

```
1230 REM - LINES 3260 - 3940   The calculation of number
1240 REM - of plates required at given reflux ratio and q
1250 REM - value is undertaken.   The given value of
1260 REM - distillate composition (C2) is the same as the
1270 REM - vapour composition from the top plate.   This
1280 REM - value is ascribed to A1 and the corresponding
1290 REM - equilibrium value in the liquid phase is obtained
1300 REM - from the subroutine (B1).   At line 3350 these
1310 REM - values (C2 and B1) corresponding to the vapour
1320 REM - and liquid equilibrium compositions on plate 1
1330 REM - are printed.   The value of B1 above is that of
1340 REM - X(1) (line 3330) and this value is used
1350 REM - at line 3390 to calculate from the
1360 REM - equation of the enriching line (Equation 4.3)
1370 REM - the composition of vapour from plate 2.
1380 REM - The corresponding equilibrium composition
1390 REM - of liquid on plate 2 is then calculated
1400 REM - (lines 3410 - 3440).   This procedure is
1410 REM - repeated until the q-line is reached.   At each
1420 REM - iteration the value of vapour composition  Y(G)
1430 REM - calculated above is also used to calculate a
1440 REM - value of liquid composition (Q2) from the
1450 REM - q-line equation (lines 3480 - 3610).   If
1460 REM - this value is greater than that calculated
1470 REM - from the operating line equation, then the
1480 REM - feedplate has been reached.   The program then
1490 REM - moves to the next routine, and continues, this
1500 REM - time using the equation for the stripping line
1510 REM - (Equation 4.6) at line 3710.   Iteration
1520 REM - continues until the value of residue
1530 REM - composition (C3) has been reached.
1540 REM - LINES 3960 - 4560   The Lagrange interpolation
1550 REM - routine described in Chapter 2
1560 REM - LINES 4580 - 4990   Calculation using the
1570 REM - relative volatility method.   A value of relative
1580 REM - volatility is calculated for each equilibrium
1590 REM - data pair, and the geometric mean value is then
1600 REM - calculated (lines 4600 - 4730).   The highest
1610 REM - and lowest values are then selected and printed
1620 REM - out (lines 4730 - 4970).   Values of liquid
1630 REM - composition (B1) are calculated as required at
1640 REM - line 5030.
1650 REM *************************************************
2000 DIM A(20),B(20),D(20),N(20),V(20),X(100),Y(100)
2010 DIM M(20,20)
2020 REM-DATA FOR BENZENE/TOLUENE
2030 N1=7
2040 FOR J=1 TO N1
2050 READ B(J)
2060 NEXT J
2070 DATA 1.0,0.78,0.581,0.411,0.258,0.130,0.04
2080 FOR J=1 TO N1
2090 READ A(J)
2100 NEXT J
2110 DATA 1.0,0.9,0.777,0.632,0.456,0.261,0.1
2120 PRINT "TO USE DATA FOR BENZENE/TOLUENE 101.325KPA"
2130 INPUT "INPUT 1";D
2140 PRINT
2150 IF D<>1 THEN 2200
2160 PRINT "USING DATA FOR BENZENE/TOLUENE"
2170 GOSUB 3960
```

```
2180 GOSUB 4590
2190 GOTO 2320
2200 PRINT "NUMBER OF DATA PAIRS";
2210 INPUT N1
2220 FOR J=1 TO N1
2230 A(J)=0
2240 B(J)=0
2250 NEXT J
2260 PRINT "INPUT Y,X"
2270 FOR J=1 TO N1
2280 INPUT A(J),B(J)
2290 NEXT J
2300 GOSUB 3960
2310 GOSUB 4590
2320 PRINT "DO YOU WANT TO CHECK DATA? TYPE Y OR N";
2330 INPUT A$
2340 PRINT
2350 IF A$="N" THEN 2450
2360 PRINT "INPUT VALUE OF VAP COMP Y";
2370 INPUT A1
2380 GOSUB 5010
2390 PRINT "BY RELATIVE VOLATILITY METHOD X=";
2400 PRINT USING "##.##";B1
2410 GOSUB 4260
2420 PRINT "BY LAGRANGE METHOD X=";
2430 PRINT USING "##.##";B1
2440 GOTO 2320
2450 PRINT
2460 PRINT "LAGRANGIAN INTERPOLATION OR RELATIVE VOLATILITY"
2470 INPUT "METHOD, TYPE LI OR RV";B$
2480 PRINT "FEED,OVERHEAD AND BOTTOM COMPOSITIONS AS"
2490 PRINT "MOL FRACT OF MORE VOLATILE COMPONENT";
2500 INPUT C1,C2,C3
2510 REM - TOTAL REFLUX CALCULATION ************************
2520 B1=C2
2530 PRINT " X=         Y=       PLATE NO"
2540 FOR G=1 TO 100
2550 A1=B1
2560 IF B$="LI" THEN 2590
2570 GOSUB 5010
2580 GOTO 2600
2590 GOSUB 4260
2600 PRINT USING "#.###      ";B1;A1;
2610 PRINT G
2620 IF B1<C3 THEN 2660
2630 NEXT G
2640 PRINT "ERROR IN DATA"
2650 GOTO 5010
2660 P1=G-(C3-B1)/(A1-B1)
2670 PRINT "AT TOTAL REFLUX "
2680 PRINT "NUMBER OF PLATES REQUIRED=";
2690 PRINT USING "#.##";P1
2700 PRINT
2710 REM - MINIMUM REFLUX CALCULATION **********************
2720 PRINT "Q VALUE";
2730 INPUT Q1
2740 IF Q1=0 THEN 2960
2750 IF Q1=1 THEN 3030
2760 K1=.1
2770 Y2=C2+K1
2780 FOR J=1 TO 100
```

```
2790 Y2=Y2-K1
2800 A1=Y2
2810 IF B$="LI" THEN 2840
2820 GOSUB 5010
2830 GOTO 2850
2840 GOSUB 4260
2850 B2=(Y2*(Q1-1)+C1)/Q1
2860 IF B2>B1 THEN 2900
2870 NEXT J
2880 PRINT "J=100"
2890 GOTO 5010
2900 IF B2/B1<1.01 THEN 2940
2910 Y2=Y2+K1
2920 K1=K1*.1
2930 GOTO 2780
2940 Q2=(B1+B2)/2
2950 GOTO 3180
2960 A1=C1
2970 IF B$="LI" THEN 3000
2980 GOSUB 5010
2990 GOTO 3010
3000 GOSUB 4260
3010 Q2=B1
3020 GOTO 3180
3030 Q2=C1
3040 K1=.1
3050 Y2=C1
3060 Y2=Y2+K1
3070 A1=Y2
3080 IF B$="LI" THEN 3110
3090 GOSUB 5010
3100 GOTO 3120
3110 GOSUB 4260
3120 IF B1<C1 THEN 3060
3130 IF B1=C1 THEN 3180
3140 IF B1/C1<1.01 THEN 3180
3150 Y2=Y2-K1
3160 K1=K1/10
3170 GOTO 3060
3180 R2=(C2-Y2)/(C2-Q2)
3190 R1=R2/(1-R2)
3200 PRINT "AT Q VALUE OF";Q1;" MINIMUM REFLUX RATIO=";
3210 PRINT USING "##.##";R1
3220 PRINT "FOR MINIMUM REFLUX RATIO AT ANOTHER Q VALUE"
3230 INPUT "TYPE MR";A$
3240 PRINT
3250 IF A$="MR" THEN 2720
3260 PRINT "REFLUX RATIO,Q VALUE,FEEDRATE";
3270 INPUT R1,Q1,F1
3280 A1=C2
3290 IF B$="LI" THEN 3320
3300 GOSUB 5010
3310 GOTO 3330
3320 GOSUB 4260
3330 X(1)=B1
3340 PRINT" X=         Y=       PLATE NO"
3350 PRINT USING "#.###      ";B1;C2;
3360 PRINT " 1"
3370 REM - ENRICHING LINE **********************************
3380 FOR G=2 TO 100
3390 Y(G)=X(G-1)*R1/(R1+1)+C2/(R1+1)
```

```
3400 A1=Y(G)
3410 IF B$="LI" THEN 3440
3420 GOSUB 5010
3430 GOTO 3450
3440 GOSUB 4260
3450 PRINT USING "#.###      ";B1;Y(G);
3460 PRINT G
3470 X(G)=B1
3480 REM - Q LINE ******************************************
3490 IF Q1=0 THEN 3530
3500 IF Q1=1 THEN 3600
3510 Q2=(Y(G)*(Q1-1)+C1)/Q1
3520 GOTO 3610
3530 A1=C1
3540 IF B$="LI" THEN 3570
3550 GOSUB 5010
3560 GOTO 3580
3570 GOSUB 4240
3580 Q2=B1
3590 GOTO 3610
3600 Q2=C1
3610 IF Q2>X(G) THEN 3650
3620 NEXT G
3630 PRINT "OVER 100 PLATES"
3640 GOTO 3900
3650 PRINT "FEEDPLATE"
3660 D1=F1*(C1-C3)/(C2-C3)
3670 W1=F1-D1
3680 L1=Q1*F1+R1*D1
3690 REM - STRIPPING LINE **********************************
3700 FOR H=G+1 TO 100
3710 Y(H)=L1*X(H-1)/(L1-W1)-C3*W1/(L1-W1)
3720 A1=Y(H)
3730 IF B$="LI" THEN 3760
3740 GOSUB 5010
3750 GOTO 3770
3760 GOSUB 4240
3770 PRINT USING "#.###      ";B1;Y(H);
3780 PRINT H
3790 IF B1<C3 THEN 3840
3800 X(H)=B1
3810 NEXT H
3820 PRINT "OVER 100 PLATES"
3830 GOTO 3900
3840 PRINT "FEEDRATE=";F1
3850 PRINT "DISTILLATE=";D1
3860 PRINT "RESIDUE=";W1
3870 PRINT "NO. OF PLATES=";
3880 P1=H-1+(X(H-1)-C3)/(X(H-1)-B1)
3890 PRINT USING "##.##";P1
3900 INPUT "TYPE RR FOR REPEAT RUN BY SAME METHOD";A$
3910 IF A$="RR" THEN 3260
3920 INPUT "TYPE RC FOR REPEAT CALCULATION FROM START";A$
3930 IF A$="RC" THEN 2450
3940 GOTO 5050
3950 REM ***************************************************
3960 REM - FIRST SUBROUTINE OF LAGRANGIAN INTERPOLATION
3970 FOR J=1 TO N1
3980 FOR K=1 TO N1
3990 M(J,K)=0
4000 NEXT K
```

```
4010 NEXT J
4020 REM - CALCULATES VALUES OF X1-X2 ETC ******************
4030 REM - & STORES THEM IN MATRIX M
4040 FOR J=1 TO N1-1
4050 FOR K=J TO N1-1
4060 M(J,K)=A(J)-A(K+1)
4070 NEXT K
4080 NEXT J
4090 FOR K=1 TO N1-1
4100 FOR J=K+1 TO N1
4110 M(J,K)=A(J)-A(K)
4120 NEXT J
4130 NEXT K
4140 REM - CALCULATES DENOMINATOR TERMS ********************
4150 REM - & STORES THEM IN MATRIX D
4160 FOR J=1 TO N1
4170 D(J)=1
4180 FOR K=1 TO N1-1
4190 D(J)=D(J)*M(J,K)
4200 NEXT K
4210 NEXT J
4220 RETURN
4230 REM ***************************************************
4240 REM - SECOND SUBROUTINE OF
4250 REM - LAGRANGIAN INTERPOLATION ************************
4260 FOR J=1 TO N1
4270 FOR K=1 TO N1
4280 M(J,K)=0
4290 NEXT K
4300 NEXT J
4310 REM - CALCULATES VALUES OF X-X1 ETC *******************
4320 REM - & STORES THEM IN MATRIX M
4330 FOR J=1 TO N1-1
4340 FOR K=J TO N1-1
4350 M(J,K)=A1-A(K+1)
4360 NEXT K
4370 NEXT J
4380 FOR K=1 TO N1-1
4390 FOR J=K+1 TO N1
4400 M(J,K)=A1-A(K)
4410 NEXT J
4420 NEXT K
4430 REM - CALCULATES NUMERATOR TERMS **********************
4440 REM - & STORES THEM IN MATRIX N
4450 FOR J=1 TO N1
4460 N(J)=1
4470 FOR K=1 TO N1-1
4480 N(J)=N(J)*M(J,K)
4490 NEXT K
4500 NEXT J
4510 B1=0
4520 REM - EVALUATES EACH TERM & ADDS THEM *****************
4530 FOR J=1 TO N1
4540 B1=B1+B(J)*N(J)/D(J)
4550 NEXT J
4560 RETURN
4570 REM ***************************************************
4580 REM - FIRST SUBROUTINE OF RELATIVE VOLATILITY METHOD
4590 REM - RELATIVE VOLATILITY VALUES *********************
4600 V1=1
4610 Z1=0
```

```
4620 FOR J=1 TO N1
4630 IF A(J)=1! THEN 4680
4640 IF A(J)=0 THEN 4700
4650 V(J)=A(J)*(1-B(J))/(B(J)*(1-A(J)))
4660 V1=V1*V(J)
4670 GOTO 4720
4680 J1=J
4690 GOTO 4710
4700 J2=J
4710 Z1=Z1+1
4720 NEXT J
4730 V2=V1^(1/(N1-Z1))
4740 FOR J=1 TO N1
4750 FOR K=J TO N1-1
4760 IF J=J1 THEN 4820
4770 IF J=J2 THEN 4820
4780 IF V(J)>V(K+1) THEN 4800
4790 GOTO 4820
4800 NEXT K
4810 GOTO 4830
4820 NEXT J
4830 PRINT "HIGHEST VALUE OF REL VOL=";
4840 PRINT USING "##.##";V(J)
4850 FOR J=1 TO N1
4860 FOR K=J TO N1-1
4870 IF J=J1 THEN 4940
4880 IF J=J2 THEN 4940
4890 IF V(K+1)=0 THEN 4920
4900 IF V(J)<V(K+1) THEN 4920
4910 GOTO 4940
4920 NEXT K
4930 GOTO 4950
4940 NEXT J
4950 PRINT "LOWEST VALUE OF REL VOL=";
4960 PRINT USING "##.##";V(J)
4970 PRINT "MEAN VALUE OF REL VOL=";
4980 PRINT USING "##.##";V2
4990 RETURN
5000 REM ****************************************************
5010 REM - SECOND SUBROUTINE OF RELATIVE VOLATILITY METHOD
5020 REM - COMPOSITION VALUES *******************************
5030 B1=A1/(V2*(1-A1)+A1)
5040 RETURN
5050 END
```

EXAMPLE 4.2

Use the above computer program to repeat Example 4.1. It will be seen that answers are not precisely the same. This may be attributed to inaccuracies in the graphical construction and in the interpolation procedure.

```
LOAD"A:DIST1
Ok
RUN
TO USE DATA FOR BENZENE/TOLUENE 101.325KPA
INPUT 1? 2

NUMBER OF DATA PAIRS? 14
INPUT Y,X
? 1,1
? .98,.95
? .96,.89
? .9,.81
? .77,.61
? .63,.44
? .51,.33
? .46,.28
? .39,.23
? .2,.11
? .18,.097
? .13,.067
? .078,.039
? .025,.012
HIGHEST VALUE OF REL VOL= 2.97
LOWEST VALUE OF REL VOL= 2.02
MEAN VALUE OF REL VOL= 2.20
DO YOU WANT TO CHECK DATA? TYPE Y OR N? Y

INPUT VALUE OF VAP COMP Y? .97
BY RELATIVE VOLATILITY METHOD X= 0.94
BY LAGRANGE METHOD X= 0.92
DO YOU WANT TO CHECK DATA? TYPE Y OR N? N


LAGRANGIAN INTERPOLATION OR RELATIVE VOLATILITY
METHOD, TYPE LI OR RV? LI
FEED,OVERHEAD AND BOTTOM COMPOSITIONS AS
MOL FRACT OF MORE VOLATILE COMPONENT? .38,.973,.045
 X=          Y=        PLATE NO
0.928       0.973        1
0.831       0.928        2
0.736       0.831        3
0.546       0.736        4
0.369       0.546        5
0.218       0.369        6
0.123       0.218        7
0.063       0.123        8
0.032       0.063        9
AT TOTAL REFLUX
NUMBER OF PLATES REQUIRED=8.58

Q VALUE? 1.64
AT Q VALUE OF 1.64  MINIMUM REFLUX RATIO= 1.31
FOR MINIMUM REFLUX RATIO AT ANOTHER Q VALUE
TYPE MR? M
```

```
REFLUX RATIO,Q VALUE,FEEDRATE? 2.1,1.64,.138
 X=          Y=         PLATE NO
0.928       0.973         1
0.852       0.942         2
0.804       0.891         3
0.777       0.859         4
0.751       0.840         5
0.721       0.823         6
0.678       0.802         7
0.617       0.773         8
0.539       0.732         9
0.477       0.679         10
FEEDPLATE
0.442       0.633         11
0.407       0.587         12
0.361       0.538         13
0.294       0.475         14
0.227       0.385         15
0.175       0.293         16
0.125       0.222         17
0.081       0.154         18
0.047       0.094         19
0.026       0.048         20
FEEDRATE= .138
DISTILLATE= 4.981681E-02
RESIDUE= 8.818319E-02
NO. OF PLATES=19.10
TYPE RR FOR REPEAT RUN BY SAME METHOD? N
TYPE RC FOR REPEAT CALCULATION FROM START? RC

LAGRANGIAN INTERPOLATION OR RELATIVE VOLATILITY
METHOD, TYPE LI OR RV? RV
FEED,OVERHEAD AND BOTTOM COMPOSITIONS AS
MOL FRACT OF MORE VOLATILE COMPONENT? .38,.973,.045
 X=          Y=         PLATE NO
0.942       0.973         1
0.882       0.942         2
0.772       0.882         3
0.607       0.772         4
0.412       0.607         5
0.242       0.412         6
0.127       0.242         7
0.062       0.127         8
0.029       0.062         9
AT TOTAL REFLUX
NUMBER OF PLATES REQUIRED=8.51

Q VALUE? 1.64
AT Q VALUE OF 1.64  MINIMUM REFLUX RATIO= 1.56
FOR MINIMUM REFLUX RATIO AT ANOTHER Q VALUE
TYPE MR? M
```

```
REFLUX RATIO,Q VALUE,FEEDRATE? 2.1,1.64,.138
 X=           Y=          PLATE NO
0.942        0.973          1
0.901        0.952          2
0.847        0.924          3
0.782        0.888          4
0.711        0.844          5
0.639        0.795          6
0.573        0.747          7
0.517        0.702          8
0.473        0.664          9
FEEDPLATE
0.435        0.629          10
0.383        0.577          11
0.318        0.506          12
0.245        0.417          13
0.175        0.318          14
0.115        0.222          15
0.069        0.140          16
0.037        0.078          17
FEEDRATE= .138
DISTILLATE= 4.981681E-02
RESIDUE= 8.818319E-02
NO. OF PLATES=16.75
TYPE RR FOR REPEAT RUN BY SAME METHOD? N
TYPE RC FOR REPEAT CALCULATION FROM START? N
Ok
```

PROBLEMS - CHAPTER 4

1. Repeat example 4.2 using program DIST1, and assuming that the feed enters the tower as a vapour/liquid mixture containing 80 wt% liquid and 20 wt% vapour.

2. Use the data for acetone/acetic acid, and program DIST1, to solve the following problems:

(a) Determine the number of plates required at total reflux, and the minimum reflux ratio required, if a distillation column operates with feed, distillate and residue compositions of 40 mol%, 97.5 mol%, and 5 mol% acetone respectively.

(b) A still column for the separation of an acetone/acetic acid mixture is operated as an enriching column only. The feed contains 30 mol% acetone, and enters the bottom of the column as a saturated vapour. If a reflux ratio of 4 is used, determine the number of ideal plates required in order to produce a distillate containing 96 mol% acetone. What is the composition of the liquid leaving the bottom plate?

Verify your answer by carrying out the McCabe Thiele construction.

Equilibrium Data for Acetone/Acetic Acid at 1 atm

Mol% Acetone		Temperature
In liquid	In Vapour	^{0}C
0	0	118.1
5	16.2	110.0
10	30.6	103.8
20	55.7	93.1
30	72.5	85.8
40	84.0	79.7
50	91.2	74.6
60	94.7	70.2
70	96.9	66.1
80	98.4	62.6
90	99.3	59.2
100	100	56.1

Source: Reprinted with permission from G.G. Brown & Others, Unit Operations, John Wiley & Sons Inc., copyright 1950 ©

3. Write a program to calculate bubble and dew points of multicomponent mixtures from:

(a) Equilibrium constants;

(b) Vapour pressures.

4. Write a program for a single stage flash of a multicomponent mixture:

(a) Isothermal

(b) Adiabatic

5. Write a program for a separation of multicomponent mixtures in a multi-stage column (5), (6), (11):

(a) With total condenser

(b) With partial condenser

6. Modify the McCabe-Thiele program given to handle stripping, with and without live steam.

REFERENCES

1 R.E. Treybal, Mass Transfer Operations, 2nd Edition, McGraw Hill Book Co., New York, U.S.A., 1968

2 J.M. Coulson, J.F. Richardson and Others, Chemical Engineering, Volume II, 3rd Edition, Pergamon Press, Oxford, U.K., 1978

3 C.J. Geankoplis, Transport Processes and Unit Operations, Allyn & Bacon, New Jersey, U.S.A., 1978

4 M.E. Leesley, Editor, Computer-aided Process Plant Design, Gulf Publishing Co., Houston, Texas, U.S.A., 1982

5 C.D. Holland, Fundamentals of Multi-component Distillation, McGraw Hill Book Co., New York, U.S.A., 1981

6 C.D. Holland, A.I. Liapis, Computer Methods for Solving Dynamic Separation Problems, McGraw Hill Book Co., New York, U.S.A., 1983

7 R.S. Mah, W.D. Seider, Editors, Foundations of Computer-Aided Chemical Process Design, Engineering Foundation, New York, U.S.A. 1981

8 S.M. Walas, Phase Equilibria in Chemical Engineering, Butterworth, London, U.K. 1985

9 M. Hirata, S. Ohe, K. Nagahama, Kodansha Ltd., Computer-aided Data Book of Vapour-Liquid Equilibria, Elsevier Scientific, Amsterdam, Netherlands, 1975

10 J. Prausnitz and Others, Computer Calculations for Multi-component Vapour-Liquid and Liquid-Liquid Equilibria, Prentice Hall, New Jersey, U.S.A., 1980

11 D.N. Hanson, D.H. Duffin, G.F. Somerville, Computation of Multi-stage Separation Processes, Reinhold, New York, U.S.A. 1962

Chapter 5

LINEAR PROGRAMMING
(A) THE SIMPLEX METHOD

Linear programming is concerned with finding solutions to problems about optimum allocation of resources, or production facilities, blending of products, etc. Generally these are looked at from the view point of finding the solution of lowest cost or greatest profitability. The mathematics of these methods were worked out by American economists between about 1945 and 1955 (1).

The term linear programming refers not to computer programming (although of course this is widely used in connection with the technique), but to the preparation of programs or plans concerning the future distribution of resources. The mathematical expressions employed can be expressed in linear form.

The following simple example demonstrates the linear nature of the relationships and also gives the basis of the solution method.

EXAMPLE 5.1

A plastics moulder produces two different qualities of moulded article:

Standard Quality - made from a mix consisting of
30% copolymer and
70% homopolymer

Super Quality - made from a mix consisting of
60% copolymer and
40% homopolymer

Supplies of the two constituent materials are available as follows:

Copolymer - 12 tonnes per week at $1900 per tonne
Homopolymer - 15 tonnes per week at $1450 per tonne

Maximum production capacity of the plant is 25 tonnes per week.

If the profit on each tonne of standard quality product is $700 and on each tonne of super quality is $1000, calculate how much of each quality should be manufactured in order to maximise the profit.

The above conditions and constraints can be represented mathemetically as follows:

Let x_1 and x_2 represent the tonnes of standard and super grades respectively to be produced each week.

Then the profit $z = 700x_1 + 1000x_2$

The total production cannot be greater than 25 tonnes per week, hence $x_1 + x_2 \leq 25$

The quantity of copolymer to be used per week is

$0.3x_1 + 0.6x_2 \leq 12$

The quantity of homopolymer to be used per week is

$0.7x_1 + 0.4x_2 \leq 15$

Since only two variables are involved in each of the above constraints, they can all be represented on the one graph. If we replace the inequalities by equalities, we see that they are all the equations of straight lines. These are shown on Figure 5.1. To comply with all the constraints operating conditions must lie on, or to the left of the lines denoted by the points B, C, D, E. These points are known as extreme points (3). The other extreme point which lies on the boundary of feasible solutions is the origin,Point A.

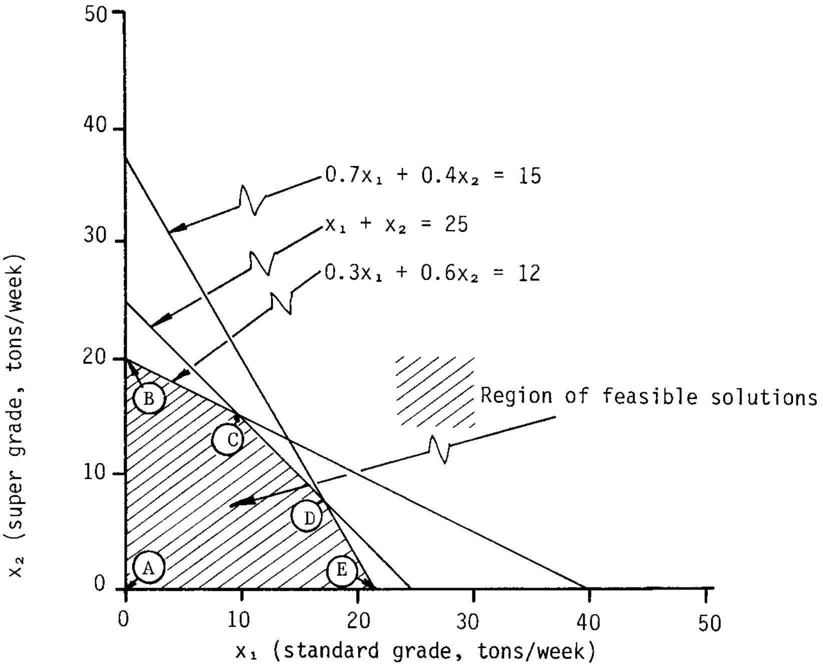


Figure 5.1. Polymer Blending

If we read the co-ordinates of the points of intersection B, C, D, E, we can evaluate the profit obtainable for each case, as follows:

Operation at point B: production of 20 tonnes per week of super grade only

Profit = 20 $*$ 1000 = $20,000

Operation at point C: production of 15 tonnes per week of super grade and 10 tonnes per week of standard

Profit = 15 $\ast$ 1000 + 10 $\ast$ 700 = \$22,000

Operation at point D: production of 8 tonnes per week of super grade and 17 tonnes per week of standard

Profit = 8 $\ast$ 1000 + 17 $\ast$ 700 = \$19,900

Operation at point E: production of 21.4 tonnes per week of standard grade only

Profit = 21.4 $\ast$ 700 = \$14,980

Operation along the CD line is obviously best, because maximum production is possible. Is the profitability at C the best that can be obtained? It is easily demonstrated by evaluating profits for other points along CD, that operation at point C is the most profitable.

The graphical technique can of course deal only with constraints involving two variables. Analogously, problems involving three variables could be handled by a three dimensional technique involving the intersection of plane surfaces. Larger numbers of variables can only be handled by mathematical methods involving the use of matrices.

THE SIMPLEX METHOD

The best known technique of linear programming is the Simplex method. It employs the variables in a matrix form and is based upon the observation that the optimum solution occurs at one of the 'extreme points' mentioned in Example 5.1. Thus referring to Figure 5.1, the Simplex method would commence at the origin (the first extreme point of the region of feasible solutions). It would then evaluate points B, C, D, etc. in turn until the optimum had been reached.

No attempt will be made here to explain or justify the method; many excellent tests are available which do this (1), (2), (3), (4). The mechanics of use of the method only will be given, as follows:

The basic form of the Simplex method involves the maximisation of a linear algebraic equation such as

$$z = c_1x_1 + c_2x_2 + \dots c_nx_n$$

z is called the objective function; the values of c are called the cost vector, and the above equation can be written in matrix notation as

$$\mathbf{z} = \mathbf{c}^T\mathbf{x}$$

There can be any number i of constraints, and these can be equality constraints or inequality constraints. The Simplex method only considers cases where the constraint is bounded by a maximum value, i.e.

$a_{i1}x_1 + a_{i2}x_2 + \ldots . a_{in}x_n \leq b_i$

(cases involving minimisation of the objective function, negative values of b and minimum value constraints will be briefly discussed later). This equation can be written in matrix notation as

$\mathbf{AX} \leq \mathbf{B}$ where

$\mathbf{A}$ is the constraint matrix;

$\mathbf{B}$ is the resource vector;

$\mathbf{X}$ is the variable vector.

Furthermore, we only consider positive values of the x variables, i.e. in matrix notation

$\mathbf{X} \geq 0$

The problem written in the above way is said to be 'standard form'.

The next step is to rewrite the problem in 'canonical form' which involves rewriting the constraints as equalities. This is done by introducing into each equation a 'slack variable', thus

$a_{i1}x_1 + a_{i2}x_2 + \ldots . a_{in}x_n + x_{n+m} = b_i$

Here the subsript n indicates the number of variables present in the constraints, and the subscript m indicates the number of constraints.

The various equations are then arranged in a table, also referred to as a 'tableau'. Suppose for instance that the problem to be maximised is as follows:

Maximise $z = c_1x_1 + c_2x_2 + c_3x_3$

subject to $a_{11}x_1 + a_{12}x_2 + a_{13}x_3 \leq b_1$

$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 \leq b_2$

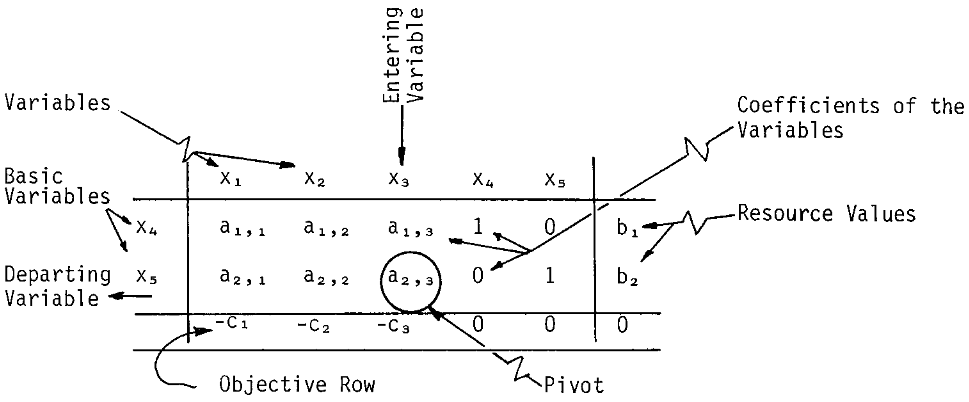
$x_1 \geq 0 \quad x_2 \geq 0 \quad x_3 \geq 0$

Slack variables are introduced into the constraint inequalities so that they become

$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + x_4 = b_1$

$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + x_5 = b_2$

The tableau is then drawn up as shown in Figure 5.2.



Basic Variables	x_1	x_2	x_3	x_4	x_5	
x_4	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	1	0	b_1
x_5	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	0	1	b_2
	$-c_1$	$-c_2$	$-c_3$	0	0	0

Figure 5.2. General form of the Simplex Tableau.

The value 1 is ascribed to each of the slack variables; each slack variable is also listed in the left hand column in which position it is referred to as a 'basic variable'. For its entry into the tableau the equation of the objective function is rewritten as:

$$-c_1x_1 - c_2x_2 - c_3x_3 + z = 0$$

In this form the coefficients appear in the bottom row, the value of z appearing in the right hand column. The initial value of z is zero, corresponding to the initial feasible solution (for example as seen to occur at the origin in Figure 5.1).

A new tableau is then constructed by the manipulation known as pivoting. To do this the 'pivotal column' and 'pivotal row' must first be established. The pivotal column is that above the most negative element in the objective row. We will suppose this to be $-c_3$. Then the pivotal column is that below variable x_3.

To find the pivotal row, 'θ ratios' are determined for each coefficient of x in the pivotal column. In this case, the θ ratios are:

$$\theta_1 = b_1/a_{1,3}$$

$$\theta_2 = b_2/a_{2,3}$$

The smallest positive value of θ establishes the location of the pivotal row. Thus if θ_2 is the smallest value then the pivotal row will be that containing b_2.

The intersection of the pivotal row and the pivotal column is known as the 'pivot'. Should there be no positive elements in the pivotal column, then no optimum solution to the problem is possible.

The variable listed at the top of the pivotal column is known as the 'entering'variable'; the basic variable in the pivotal row is known as the 'departing variable'. In this case, x_3 is the entering variable and x_5 is the departing variable.

The first step in drawing up the new tableau is to write the name of the entering variable in the space previously occupied by the departing variable in the column of basic variables.

Next, new values are entered in the pivotal row, by dividing the old value by the value of the pivot($a_{2,3}$ in this case). The element in the pivot position now has value 1 in the new tableau. Suitable values of the new pivotal row are then added to or subtracted from each other row, so that all the other elements in the pivotal column will have value zero. The new tableau is shown in Figure 5.3.

	x_1	x_2	x_3	x_4	x_5	
x_4	$a_{1,1} - a_{1,3} * \frac{a_{2,1}}{a_{2,3}}$	$a_{1,2} - a_{1,3} * \frac{a_{2,2}}{a_{2,3}}$	0	1	$\frac{-a_{1,3}}{a_{2,3}}$	$b_1 - b_2 * \frac{a_{1,3}}{a_{2,3}}$
x_3	$\frac{a_{2,1}}{a_{2,3}}$	$\frac{a_{2,2}}{a_{2,3}}$	1	0	$\frac{1}{a_{2,3}}$	$\frac{b_2}{a_{2,3}}$
	$-c_1 + c_3 * \frac{a_{2,1}}{a_{2,3}}$	$-c_2 + c_3 * \frac{a_{2,2}}{a_{2,3}}$	0	0	$\frac{c_3}{a_{2,3}}$	$c_3 * \frac{b_2}{a_{2,3}}$

Figure 5.3. The new form of the tableau shown in Figure 5.2, after pivoting about the element $a_{2,3}$

This tableau provides another feasible solution to the problem, which will have a larger value than the preceding solution. In this case, the solution would be:

$z = c_3 * \frac{b_2}{a_{2,3}}$ the values of

x_1 and x_2 being zero, and that of x_3 being $b_2/a_{2,3}$

The new tableau in its turn is subjected to the above procedure. The most negative element in the objective row is identified and the pivoting procedure is carried out to create another tableau. Where two elements in the objective row have the same negative value, it is unimportant which is chosen for the pivoting step. If no element in the objective row has a negative value, then the solution given by the tableau is the optimum one and the procedure is completed. An example of the technique follows:

EXAMPLE 5.2

Carry out the previous problem by the Simplex method. First we express the problem in standard form: Maximise $z = 700x_1 + 1000x_2$ (the objective function).

Subject to the restrictions:

$x_1 + x_2 \leq 25$

$0.3x_1 + 0.6x_2 \leq 12$

$0.7x_1 + 0.4x_2 \leq 15$

$x_1 \geq 0 \qquad x_2 \geq 0$

Next we rewrite the restrictions as equalities, introducing slack variables; they now are in canonical form:

$x_1 + x_2 + x_3 = 25$

$0.3x_1 + 0.6x_2 + x_4 = 12$

$0.7x_1 + 0.4x_2 + x_5 = 15$

We are now in a position to draw up the first tableau:

	x_1	x_2	x_3	x_4	x_5	
x_3	1	1	1	0	0	25
x_4	0.3	$\boxed{0.6}$	0	1	0	12
x_5	0.7	0.4	0	0	1	15
	-700	-1000	0	0	0	0

This is our first feasible solution, corresponding to the origin on Figure 7.1 (point A), i.e.

$x_1 = 0$ tonnes/week
$x_2 = 0$ tonnes/week,

Profit, $z = 0$

By inspection, the entering variable is x_2 since the value below it in the objective row (-1000), is the most negative. The column below x_2 is the pivotal column.

Next we identify the departing variable by determining the θ ratios for the pivotal column. The values are:

25/1 = 25
12/0.6 = 20
15/0.4 = 37.5

12/0.6 is the lowest value and hence x_2 is the departing variable. The row in which x_4 is seen as a basic variable is now the pivotal row. The intersection of the pivotal column and row is circled to identify it. This is called the Pivot.

Values in the pivotal row are now multiplied by 1/0.6, so as to bring the value of the pivot to 1.

Suitable multiples of the new pivotal row are now added to/subtracted from all other rows so that the element in the pivotal column, in each of these rows is zero.

Thus we obtain the next tableau:

	x_1	x_2	x_3	x_4	x_5	
x_3	0.5	0	1	-1.66	0	5
x_2	0.5	1	0	1.66	0	20
x_5	0.5	0	1	-0.66	1	7
	-200	0	0	1660	0	20,000

This is our next feasible solution, corresponding to point B on Figure 5.1, i.e.

x_1 = 0
x_2 = 20 tonnes/week,
Profit, z = $20,000

Again we identify the pivotal column, which this time is that under x_1 , the value in the objective row being the most negative (-200). Calculations of θ values for this column give

5/0.5 = 10
20/0.5 = 40
7/0.5 = 14

10 is the smallest value and so the departing variable is x_3. The intersection of the pivotal column and pivotal row is circled. The pivoting step is again carried out yielding the next tableau:

	x_1	x_2	x_3	x_4	x_5	
x_1	1	0	2	-3.32	0	10
x_2	0	1	-1	3.32	0	15
x_5	0	0	-1	1.0	1	2
	0	0	400	664	0	22,000

This is our optimum solution, since there are no negative values in the objective row. It corresponds to point C on Figure 5.1, i.e.

x_1 = 10 tonnes/week
x_2 = 15 tonnes/week,
Profit, z = $22,000

Computer Solution of Linear Programming Problems by the Simplex Method

The program which follows reproduces exactly the steps of the Simplex method just described. It is based upon the already stated assumptions namely; there are some negative elements in the objective row; the constraints are all of the type $\leq$; the right hand side of each is positive.

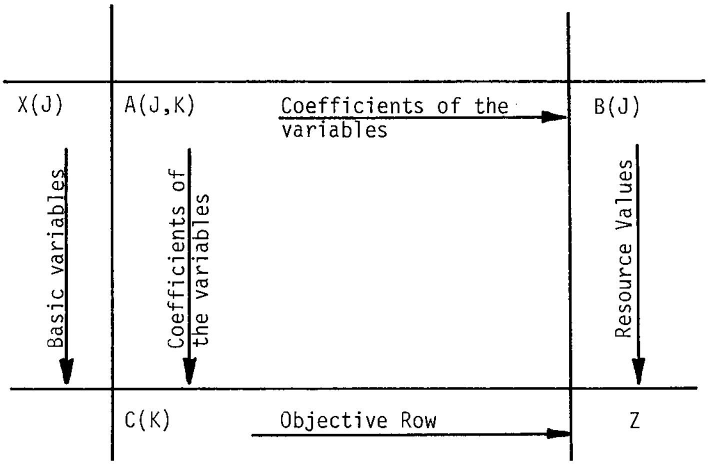
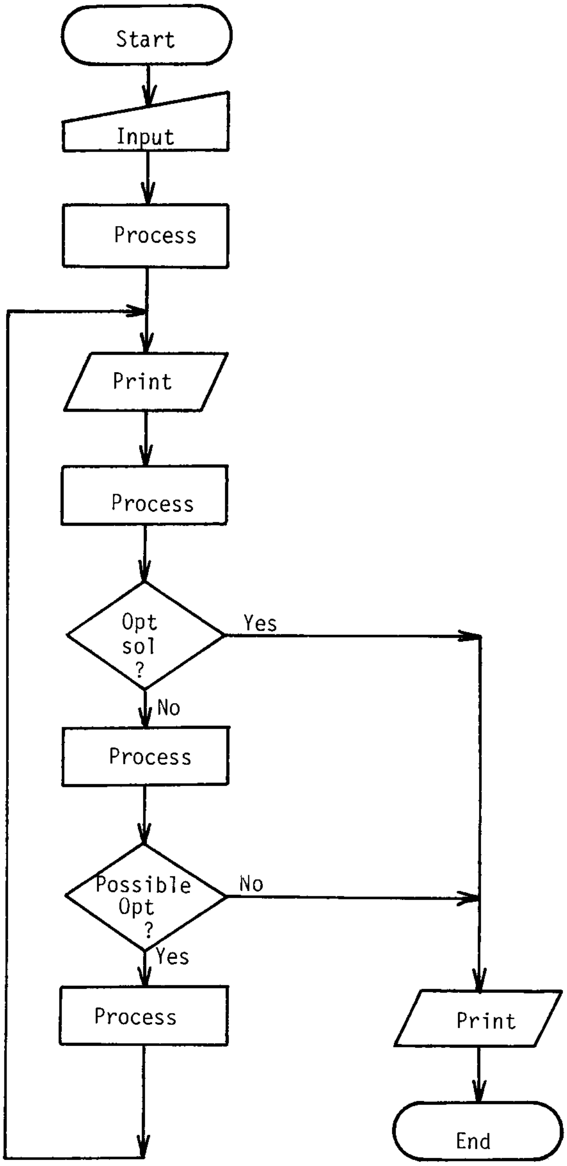


Figure 5.4. The relationship between program nomenclature and the tableau.

LNPRG1.BAS


Start
Input
Process
Print
Process
Opt sol ?
Yes
No
Process
Possible Opt ?
No
Yes
Process
Print
End

```
10  REM ****************************************************
20  REM - PROGRAM LNPRG1.BAS
30  REM - LINEAR PROGRAMMING SIMPLEX METHOD
40  REM - NOTE: RHS OF CONSTRAINTS MUST BE POSITIVE
50  REM - OBJECTIVE EQUATION MUST CONTAIN SOME POSITIVE
60  REM - COEFFICIENTS.  ALL CONSTRAINTS MUST BE OF TYPE<=
70  REM - PROGRAM NOMENCLATURE
80  REM - A(J,K)  -  Coefficients of the K variables in the
90  REM              J constraints (the body of the tableau)
100 REM - B(J)    -  The resource values, the right hand
110 REM              side of each constraint (elements of
120 REM              the right hand column of the tableau)
130 REM - C(J)    -  The elements of the objective row
140 REM - C1      -  The subscript number denoting the
150 REM              pivotal column
160 REM - N1      -  Number of variables in the objective
170 REM              function
180 REM - N2      -  Number of restrictions
190 REM - O(J)    -  Coefficients of the variables in the
200 REM              objective function
210 REM - P1      -  Reciprocal of the value of the pivot
220 REM - P2,P3   -  Used in bringing values in the pivotal
230 REM              column to zero
240 REM - R1      -  The subscript number denoting the
250 REM              pivotal row
260 REM - X(J)    -  The basic variables (elements of the
270 REM              left hand column of the tableau)
280 REM - PROGRAM DESCRIPTION
290 REM - LINES 1000 - 1300  Matrices likely to be greater
300 REM - in size than the default value of 10 are declared
310 REM - at line 1000;coefficients of the objective
320 REM - function are entered (lines 1010 - 1060) and
330 REM - subsequently the signs are changed (lines 1270 -
340 REM - 1300).
350 REM - The number of restrictions, the values of the
360 REM - coefficients and the resource values are then
370 REM - entered (lines 1070 - 1230).  Note that for each
380 REM - restriction, a number is allocated to a slack
390 REM - variable (lines 1240 - 1260), and a value of 1
400 REM - entered into the appropriate part of the table at
410 REM - line 1150
420 REM - LINES 1330 - 1460  The first tableau is printed.
430 REM - Figure 5.5 displays the relationship between
440 REM - the nomenclature of the program and the tableau
450 REM - While debugging the program, line 2240 read "GOTO
460 REM - 1330" so that the tableau was printed out at each
470 REM - iteration
480 REM - LINES 1500 - 1930  The test for an optimum
490 REM - solution, namely the absence of negative values in
500 REM - the objective row, is first made (lines 1500 -
510 REM - 1560).  Next the pivotal column is located by
520 REM - discovering the most negative value in the
530 REM - objective row (lines 1590 - 1680).  The test for
540 REM - a finite optimum, namely the presence of some
550 REM - positive entries in the pivotal column, is next
560 REM - applied (lines 1720 - 1770).  Finally the
570 REM - pivotal row is located by finding the smallest
580 REM - value of theta ratio in the pivotal column
590 REM - (lines 1800 - 1880); the coordinates of the pivot
600 REM - are ascribed to the variables C1 and R1, and the
610 REM - values are printed out (lines 1900 - 1930).
```

```
620 REM - LINES 1960 - 2210   In this segment the pivoting
630 REM - procedure is carried out.   First, the entries in
640 REM - the pivotal row are divided by the value of the
650 REM - pivot (lines 1960 - 1990); next the new values of
660 REM - the pivotal row are subtracted from the values of
670 REM - each of the other rows in turn, an appropriate
680 REM - number of times, so that all other elements in the
690 REM - pivotal column have value zero
700 REM - This is done in two stages, the first being for
710 REM - rows above the pivotal row (lines 2000 - 2080),
720 REM - the next for rows below the pivotal row (lines
730 REM - 2090 - 2160).   Finally the same operation is
740 REM - carried out on the objective row (lines 2170 -
750 REM - 2210)
760 REM - LINES 2240 - 2300   The program then returns
770 REM - to line 1460 for a further iteration (line 2240)
780 REM - As the tests at lines 1500 - 1560 and 1720 to
790 REM - 1770 determine, the program terminates, printing
800 REM - the appropriate message (lines 1550 & 1760)
810 REM ****************************************************
1000 DIM A(40,40),B(40),C(40),O(40),X(40)
1010 PRINT "NUMBER OF TERMS IN OBJECTIVE  FUNCTION"
1020 INPUT N1
1030 PRINT "COEFFICIENTS  OF X1 TO X";N1
1040 FOR J=1 TO N1
1050 INPUT O(J)
1060 NEXT J
1070 PRINT "NUMBER OF RESTRICTIONS";
1080 INPUT N2
1090 FOR J=1 TO N2
1100 PRINT "RESTRICTION";J
1110 PRINT "COEFFICIENTS OF X1 TO X";N1
1120 FOR K=1 TO N1
1130 INPUT A(J,K)
1140 NEXT K
1150 A(J,J+K-1)=1
1160 PRINT "RESOURCE VALUE";
1170 INPUT B(J)
1180 PRINT "INPUT Y TO CONTINUE ELSE N";
1190 INPUT A$
1200 IF A$="Y" THEN 1230
1210 PRINT "INPUT DATA AGAIN"
1220 GOTO 1100
1230 NEXT J
1240 FOR J=1 TO N2
1250 X(J)=N1+J
1260 NEXT J
1270 K=-1
1280 FOR J=1 TO N1+N2
1290 C(J)=K*O(J)
1300 NEXT J
1310 GOTO 1460
1320 REM ***************************************************
1330 REM - PRINT THE TABLEAU *******************************
1340 FOR J=1 TO N2
1350 PRINT X(J);"¦";
1360 FOR K=1 TO N1+N2
1370 PRINT A(J,K);'
1380 NEXT K
1390 PRINT "¦";B(J)
1400 NEXT J
```

```
1410 PRINT "    ";
1420 FOR K=1 TO N1+N2
1430 PRINT C(K);
1440 NEXT K
1450 PRINT "¦";C(N1+N2+1)
1460 PRINT "Z=";C(N1+N2+1)
1470 REM ***************************************************
1480 REM - TEST FOR OPTIMUM SOLUTION (NO NEGATIVE VALUES
1490 REM - IN OBJECTIVE ROW) *******************************
1500 FOR K=1 TO N1+N2
1510 IF C(K)>=0 THEN 1540
1520 IF ABS(C(K))<.0000001 THEN 1540
1530 GOTO 1590
1540 NEXT K
1550 PRINT "OPTIMAL SOLUTION"
1560 GOTO 2250
1570 REM ***************************************************
1580 REM - FIND PIVOTAL COLUMN *****************************
1590 N4=N4+1
1600 IF N4>50 THEN 2250
1610 FOR K=1 TO N1+N2-1
1620 IF C(K)>=0 THEN 1680
1630 FOR N=1 TO N1+N2-K
1640 IF C(K)<=C(K+N) THEN 1660
1650 GOTO 1680
1660 NEXT N
1670 GOTO 1700
1680 NEXT K
1690 REM ***************************************************
1700 REM - TEST FOR FINITE OPTIMUM (SOME POSITIVE ENTRIES IN
1710 REM - PIVOTAL COLUMN **********************************
1720 FOR J=1 TO N2
1730 IF A(J,K)<=0 THEN 1750
1740 GOTO 1800
1750 NEXT J
1760 PRINT "NO FINITE OPTIMAL SOLUTION"
1770 GOTO 2250
1780 REM ***************************************************
1790 REM - FIND PIVOTAL ROW ********************************
1800 FOR J=1 TO N2-1
1810 IF A(J,K)<=0 THEN 1880
1820 FOR N=1 TO N2-J
1830 IF A(J+N,K)<=0 THEN 1860
1840 IF B(J)/A(J,K)<=B(J+N)/A(J+N,K) THEN 1860
1850 GOTO 1880
1860 NEXT N
1870 GOTO 1890
1880 NEXT J
1890 X(J)=K
1900 C1=K
1910 R1=J
1920 PRINT "PIVOTAL ROW=";;J
1930 PRINT "PIVOTAL COLUMN=";;K
1940 REM ***************************************************
1950 REM - PIVOTING SEGMENT ********************************
1960 P1=1/A(R1,C1)
1970 FOR K=1 TO N1+N2
1980 A(R1,K)=A(R1,K)*P1
1990 NEXT K
2000 B(R1)=B(R1)*P1
2010 FOR J=1 TO R1-1
```

```
2020 IF A(J,C1)=0 THEN 2080
2030 P2=A(J,C1)/A(R1,C1)
2040 FOR K=1 TO N1+N2
2050 A(J,K)=A(J,K)-A(R1,K)*P2
2060 NEXT K
2070 B(J)=B(J)-B(R1)*P2
2080 NEXT J
2090 FOR J=R1+1 TO N2
2100 IF A(J,C1)=0 THEN 2160
2110 P2=A(J,C1)/A(R1,C1)
2120 FOR K=1 TO N1+N2
2130 A(J,K)=A(J,K)-A(R1,K)*P2
2140 NEXT K
2150 B(J)=B(J)-B(R1)*P2
2160 NEXT J
2170 P3=C(C1)/A(R1,C1)
2180 FOR K=1 TO N1+N2
2190 C(K)=C(K)-A(R1,K)*P3
2200 NEXT K
2210 C(N1+N2+1)=C(N1+N2+1)-B(R1)*P3
2220 REM ***************************************************
2230 REM - END OF PIVOTING SEGMENT *************************
2240 GOTO 1460
2250 FOR J=1 TO N2
2260 IF X(J)>N1 THEN 2280
2270 PRINT "COEFFICIENT OF X";X(J);"=";B(J)
2280 NEXT J
2290 PRINT "VALUE OF OBJECTIVE IS";C(N1+N2+1)
2300 END
```

EXAMPLE 5.3

Solve the following problem using the above computer program:

A fertiliser manufacturer markets three grades of fertiliser made from mixtures of potassium nitrate, calcium phosphate and ammonium sulphate. The mixture ratios and profit figures are tabulated below. For next weeks operations the stocks of components held are: nitrate 70 tonnes; phosphate 60 tonnes; sulphate 30 tonnes. Determine how the stocks should be used so as to maximise the profit.

	Nitrate	Phosphate	Sulphate	Profit
Grade A	57%	43%	-	\$350/ton
Grade B	57%	29%	14%	\$300/ton
Grade C	29%	29%	42%	\$250/ton

Let the weights of grades A, B & C produced be x_1, x_2, and x_3 respectively. Then profit $z = 350x_1 + 300x_2 + 250x_3$

The constraints are:

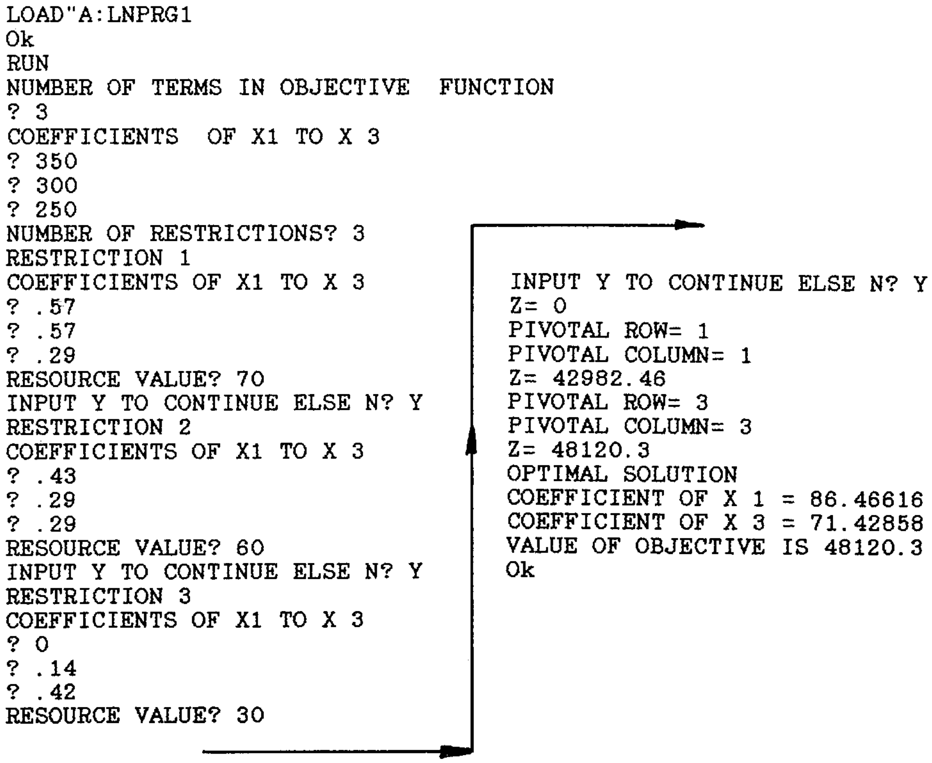
Nitrate - $0.57x_1 + 0.57_2 + 0.29_3 \leq 70$
Phosphate - $0.43x_1 + 0.29x_2 + 0.29x_3 \leq 60$
Sulphate - $0.14x_2 + 0.42x_3 \leq 30$

These figures are entered into the program; the values of x_1 and x_3 obtained are easily shown to meet the constraints, i.e.

wt nitrate used $= 0.57x_1 + 0.29x_3$
$= 0.57 \times 86.466 + 0.29 \times 71.428$
$= 70.00$ tonnes

wt phosphate used $= 0.43 \times 86.466 + 0.29 \times 71.428$
$= 57.89$ tonnes

wt sulphate used $= 0.42 \times 71.428 = 30.00$ tonnes
$= 30.00$ tonnes



```
LOAD"A:LNPRG1
Ok
RUN
NUMBER OF TERMS IN OBJECTIVE  FUNCTION
? 3
COEFFICIENTS  OF X1 TO X 3
? 350
? 300
? 250
NUMBER OF RESTRICTIONS? 3
RESTRICTION 1
COEFFICIENTS OF X1 TO X 3
? .57
? .57
? .29
RESOURCE VALUE? 70
INPUT Y TO CONTINUE ELSE N? Y
RESTRICTION 2
COEFFICIENTS OF X1 TO X 3
? .43
? .29
? .29
RESOURCE VALUE? 60
INPUT Y TO CONTINUE ELSE N? Y
RESTRICTION 3
COEFFICIENTS OF X1 TO X 3
? 0
? .14
? .42
RESOURCE VALUE? 30
```

```
INPUT Y TO CONTINUE ELSE N? Y
Z= 0
PIVOTAL ROW= 1
PIVOTAL COLUMN= 1
Z= 42982.46
PIVOTAL ROW= 3
PIVOTAL COLUMN= 3
Z= 48120.3
OPTIMAL SOLUTION
COEFFICIENT OF X 1 = 86.46616
COEFFICIENT OF X 3 = 71.42858
VALUE OF OBJECTIVE IS 48120.3
Ok
```

The preceeding simple examples do not do full justice to the power of the Simplex technique. However, in order to encompass more difficult problems, modifications to the technique are required.

For instance, it might be a requirement that the objective function be minimised rather than maximised. This might occur for example in a problem involving the usage of some constructional material. To do this we instead maximise the negative of the function. Thus if the objective function is:

$z = c_1x_1 + c_2x_2 + \ldots. c_nx_n$ then we write this problem as: maximise

$z =-c_1x_1 - c_2x_2 - \ldots. c_nx_n$

The Simplex method as given above will not work in this case because there will be no negative values in the bottom row of the tableau.

Or suppose that we need to meet an exact specification. For example, in the blending of gasolines, close specifications have to be met for both octane number and vapour pressure. The constraints in such cases could be expressed as statements of equality. Suppose the physical property of interest P, to be a simple additive function proportional to the weight fraction of each component in the mixture. The specified value of the property for the mixture is P_s
Then

$$\frac{P_1x_1 + P_2x_2 + \ldots. P_nx_n}{x_1 + x_2 + \ldots. x_n} = P_s$$

$$\therefore \quad (P_1 - P_s)x_1 + (P_2 - P_s)x_2 + \ldots. (P_n - P_s)x_n = o$$

This equality can be represented by the pair of inequalities:

$$(P_1 - P_s)x_1 + (P_2 - P_s)x_2 + \ldots. (P_n - P_s)x_n \geq 0$$

$$(P_1 - P_s)x_1 + (P_2 - P_s)x_2 + \ldots. (P_n - P_s)x_n \leq 0$$

To conform to the correct canonical form, the first inequality must be rewritten as:

$$-(P_1 - P_s)x_n - (P_2 - P_s)x_2 - \ldots. (P_n - P_s)x_n \leq 0$$

We now have a pair of constraints in suitable form for entry into the tableau. Unfortunately, difficulty will be experienced with selection of the pivot, as pivotal rows containing a zero in the right hand column will always be selected.

If a statement of equality contains a positive value on the right hand side, then when rewritten as a pair of inequalities, one of these will have a negative value on the right hand side. This is also inadmissible. For the resolution of these and other difficulties, more advanced methods such as the revised

Simplex method must be used (3).

EXAMPLE 5.4

The same manufacturer receives an order for 25 tonnes of a mixture which is to contain 30% nitrate, 30% phosphate and 40% sulphate. He decides to produce this by mixing appropriate quantities of A,B and C. How much of each should he use if the profit on the new mixture is $500 per tonne?

Let the weights of A, B and C to be used be x_1, x_2 and x_3 respectively. Then profit z =

$(x_1 + x_2 + x_3) * 500 - (350x_1 + 200x_2 + 250x_3)$

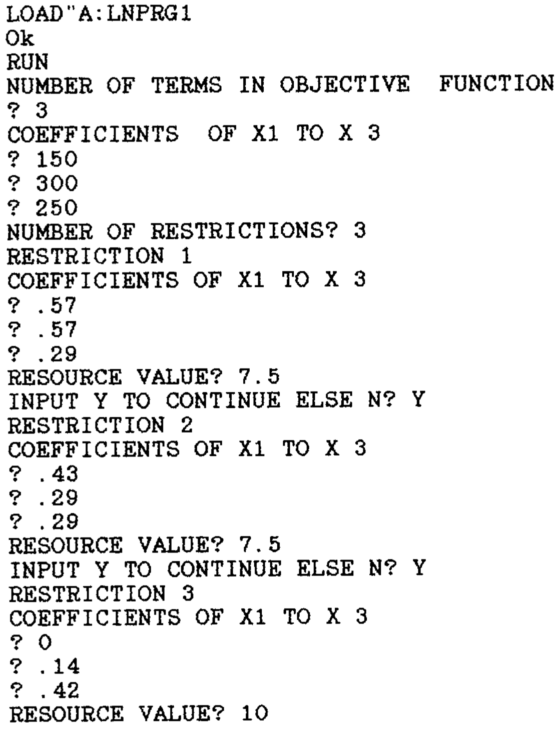
$\therefore$ $z = 150x_1 + 300x_2 + 250x_3$. The constraints are:

Nitrate $0.57x_1 + 0.57x_2 + 0.29x_3 = 0.3 * 25$

Phosphate $0.43x_1 + 0.29x_2 + 0.29x_3 = 0.3 * 25$

Sulphate $0.14x_2 + 0.42x_3 = 0.4 * 25$

The simplex method and the program above, assume not equalities but inequalities of the form $\leq$. Enter the above values and inspect the result.



```
LOAD"A:LNPRG1
Ok
RUN
NUMBER OF TERMS IN OBJECTIVE  FUNCTION
? 3
COEFFICIENTS  OF X1 TO X 3
? 150
? 300
? 250
NUMBER OF RESTRICTIONS? 3
RESTRICTION 1
COEFFICIENTS OF X1 TO X 3
? .57
? .57
? .29
RESOURCE VALUE? 7.5
INPUT Y TO CONTINUE ELSE N? Y
RESTRICTION 2
COEFFICIENTS OF X1 TO X 3
? .43
? .29
? .29
RESOURCE VALUE? 7.5
INPUT Y TO CONTINUE ELSE N? Y
RESTRICTION 3
COEFFICIENTS OF X1 TO X 3
? 0
? .14
? .42
RESOURCE VALUE? 10
```

```
INPUT Y TO CONTINUE ELSE N? Y
Z= 0
PIVOTAL ROW= 1
PIVOTAL COLUMN= 2
Z= 3947.368
PIVOTAL ROW= 3
PIVOTAL COLUMN= 3
Z= 6224.85
OPTIMAL SOLUTION
COEFFICIENT OF X 2 = 1.257546
COEFFICIENT OF X 3 = 23.39034
VALUE OF OBJECTIVE IS 6224.85
Ok
```

It will be seen that the quantities of grades A and C to be used add to 24.65 tonnes, and not 25.

Concentration of nitrate in the mix =

$$\frac{0.57 \ast 1.258 + 0.29 \ast 23.390}{24.65} =$$

$$\frac{7.50}{24.65} = 0.304$$

Concentration of phosphate in the mix =

$$\frac{0.43 \ast 1.258 + 0.29 \ast 23.390}{24.65} =$$

$$\frac{7.32}{24.65} = 0.297$$

Concentration of sulphate in the mix =

$$\frac{0.42 \ast 23.390}{24.65} =$$

$$\frac{9.82}{24.65} = 0.399$$

In order to meet the specification exactly, a more advanced calculation method must be used.

(B) THE TRANSPORTATION PROBLEM

This is one of several special classes of linear programming problem. It is concerned with the transportation of goods between a number of supply and delivery points. Having specified the journey costs for each possible route, the problem is to determine the allocation of routes which satisfies the demand at minimum cost of transportation.

The solution method is based upon the assumption that supply and demand are equal. Where demand exceeds supply, no solution is possible and the problem must be reformulated. The situation in which supply exceeds demand can be handled by the allocation of dummy delivery points having zero transportation costs.

The problem can be formulated for solution by the Simplex method (3). However, it is more usual to employ a special method which should take less computing time since it employs a much smaller matrix. This method will be briefly described below, further information being available in books already cited (1), (2), (3), (4).

The easiest way to explain the problem and the method of solution, is by an example.

EXAMPLE 5.5

Three factories (supply points) A, B and C produce respectively 500, 500 and 1000 tonnes per week of product. This has to be transported according to market demand to four warehouses (delivery points) U, V, W, and X. Next week their requirements are 300, 400, 500 and 800 tonnes respectively. We may represent these quantities as the supply vector,

$$s = \begin{bmatrix} 500 \\ 500 \\ 1000 \end{bmatrix}$$

and the demand vector

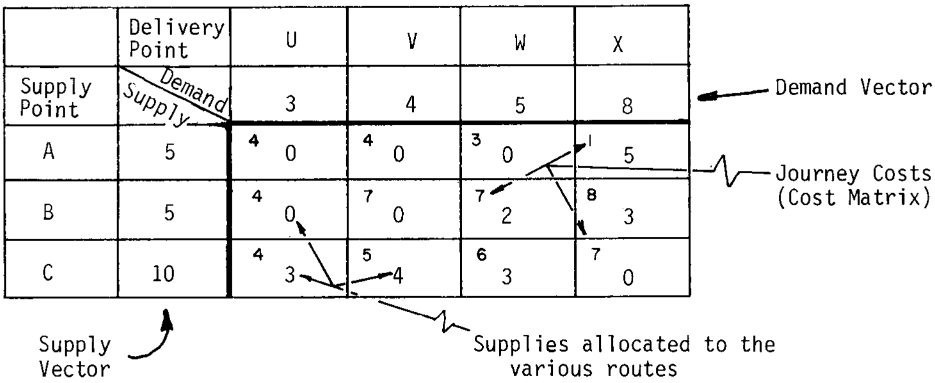
$$d = \begin{bmatrix} 300 \\ 400 \\ 500 \\ 800 \end{bmatrix}$$

The journey costs in \$/tonne between the various supply and delivery points are given in the following matrix (the cost matrix):

$$c = \begin{bmatrix} 4 & 4 & 3 & 1 \\ 4 & 7 & 7 & 8 \\ 4 & 5 & 6 & 7 \end{bmatrix}$$

How should the supply be allocated so as to satisfy the demand at the lowest possible cost for transportation?

The first step in the solution procedure is to present the above data in a tableau (Figure 5.5); units of 100 tonnes are shown.



	Delivery Point	U	V	W	X
Supply Point	Demand Supply	3	4	5	8
A	5	4 0	4 0	3 0	1 5
B	5	4 0	7 0	7 2	8 3
C	10	4 3	5 4	6 3	7 0

Demand Vector
Journey Costs (Cost Matrix)
Supply Vector
Supplies allocated to the various routes

Figure 5.5. Tableau for Example 5.5 - the basic solution.

The next step is to allocate as much of the supply as possible to the cheapest routes taken in turn, in order of increasing costs.

The lowest journey cost is that of route AX ($1/tonne); we allocate all the supply from point A by this route. The value 5 is entered in the appropriate square in the tableau. Supply from points B and C only remains to be considered.

The lowest journey costs from these supply points are for routes BU and CU ($4/tonne). Either might be chosen. We will suppose that route CU is the one selected; the value 3 is entered in the appropriate square or cell in the tableau. The remaining supplies are allocated in a similar manner and the result is also shown in Figure 5.5.

These figures are repeated in Figure 5.6A. The transportation cost for this allocation =

500*1 + 200*7 + 300*8 + 300*4 + 400*5 + 300*6 = $9,300.

Is this the cheapest possible allocation? Some sort of test must be devised which determines whether the lowest cost solution to the problem can be found. If the lowest cost solution has not been found, then a procedure must also be devised by which to modify the original solution.

The overall cost is lowered by altering the tableau by one journey at a time. This alteration is made for one unit of cargo at a time ,though in many cases involving a manual solution it may be appropriate to make the alteration for many units of cargo simultaneously.

Each alteration that we make to the tableau must be balanced so that the supply and demand restrictions are met.

Figure 5.6. Tableaux for Example 5.5

	U 3	V 4	W 5	X 8	C_i =
A 5	4 0 -2	4 0 -1	3 0 0	1 5 1	0
B 5	4 0 5	7 0 6	7 2 7	8 3 8	7
C 10	4 3 4	5 4 5	6 3 6	7 0 7	6
d_j =	-2	-1	0	1	

A - The Basic Solution after allocation of supply to the cheapest routes in turn.

Transportation Cost =
$9,300

				C_i =
4 0 -3	4 0 -2	3 0 -1	1 5 1	0
4 2 4	7 0 5	7 0 6	8 3 8	7
4 1 4	5 4 5	6 5 6	7 0 8	7
-3	-2	-1	1	

d_j = -3 -2 -1 1

B - The first transposition

Transportation Cost =
$9,100

				C_i =
4 0 -3	4 0 -1	3 0 0	1 5 1	0
4 3 4	7 0 6	7 0 7	8 2 8	7
4 0 3	5 4 5	6 5 6	7 1 7	6

d_j = -3 -1 0 1

C - The next and final transposition

Transportation Cost =
$9,000

Real costs are shown in the top left hand corner of each cell. Fictitious costs are shown in the bottom right hand corner of each cell. Supply allocated to that route is shown in the middle of each cell.

These alterations are made by considering the unused routes. The steps of the procedure are as follows (see Figure 5.7):

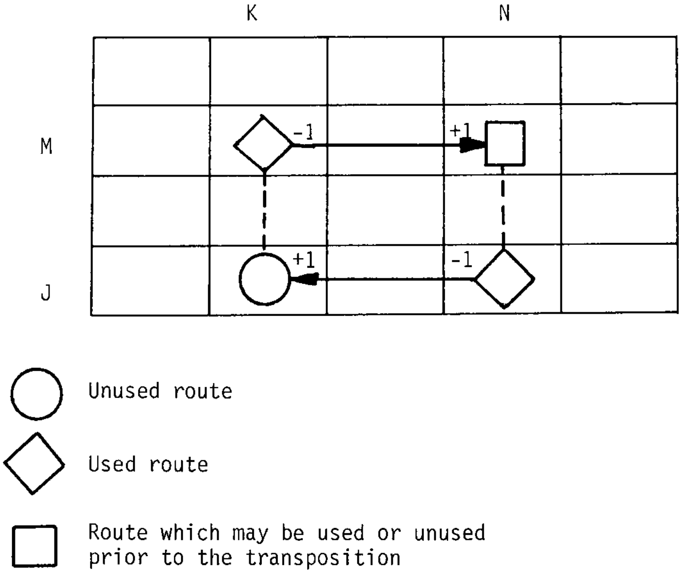


Figure 5.7. The Transportation Loop

1. An unused route is selected (co-ordinates J, K).

2. A used route is found in the column containing the unused route (co-ordinates M, K).

3. A used route is found in the row containing the unused route (co-ordinates J, N).

4. A unit of cargo allocated to route J, N is instead allocated to route J, K.

5. A unit of cargo allocated to route M, K is instead allocated to route M, N.

These transpositions, shown in Figure 5.7 are known as a 'transportation loop'.

How do we select the unused route for the next transposition of the tableau? Here are three methods:

Route Selection - Method 1, Fictitious Costs (5)

This method works well on small matrices, but contradictory values of fictitious costs can be obtained in some cases. It is therefore, unsuited to a computer solution. However, it will be briefly explained here, and then employed to complete Example 5.5.

Fictitious cost components c_i and d_j are defined for each route used in a particular tableau. The sum of these components must equal the actual costs of each route used in the tableau. Costs for the unused routes are then calculated using the same cost components. In this case, values of cost will be obtained which may or may not be the same as the actual cost for that particular route.

If the value of fictitious cost obtained for an unused route is greater than the real cost for that route, then the tableau is not optimal. The unused route which has been found should be used for the next transposition of the tableau.

Referring again to Figure 5.6A, the journey costs are shown in the top left hand corner of each cell. Fictitious costs are computed as follows: Assume

$d_X = 1$, then c_A must have value 0 since cost of route

AX is 1 $(d_X + c_A = 1 + 0 = 1)$. Consequently

$c_B = 7$ since $d_X = 1$ and cost of route BX is 8

$\therefore$ $d_W = 0$ since cost of route BW is 7 $(7 + 0 = 7)$

$c_C = 6$ since cost of route CW is 6.

Similarly values d_U and d_V are found; their values are marked on the diagram.

The addition of the cost components gives the cost figures shown in the bottom right hand corner of each cell. It will be seen that for cell BU, the fictitious cost exceeds the real cost.

We will employ BU as the unused route in the transposition of the tableau. 200 tonnes will be consigned along this route instead of along route BW. To balance this, 200 additional tonnes will be consigned along route CW instead of along route CU. This gives us the tableau shown in Figure 5.6B.

Transportation cost for this allocation of routes = \$9,100, a saving of \$200 compared with the previous tableau.

Cost components and fictitious costs are again computed and it is seen that for cell CX the fictitious cost again exceeds the real cost. A transposition about this cell yields the tableau of Figure 5.6C.

Transportation cost for this arrangement = \$9,000, a further saving of \$100.

Cost components and fictitious costs are again computed and this time in no case is the fictitious cost greater than the real cost. This tableau therefore represents an optimal solution.

We can express the solution verbally as:

500 tons from point A to point X
300 tons from point B to Point U
200 tons from point B to point X
400 tons from point C to point V
500 tons from point C to point W
100 tons from point C to point X

It should be noted that if route BU had been selected instead of route CU, at the initial allocation step, then an optimal solution would have been obtained directly.

Route Selection, Method 2 - Dual Problem

This method involves another technique of the Simplex method known as the Dual Problem. It will not be discussed here but is described by Kolman and Beck (3).

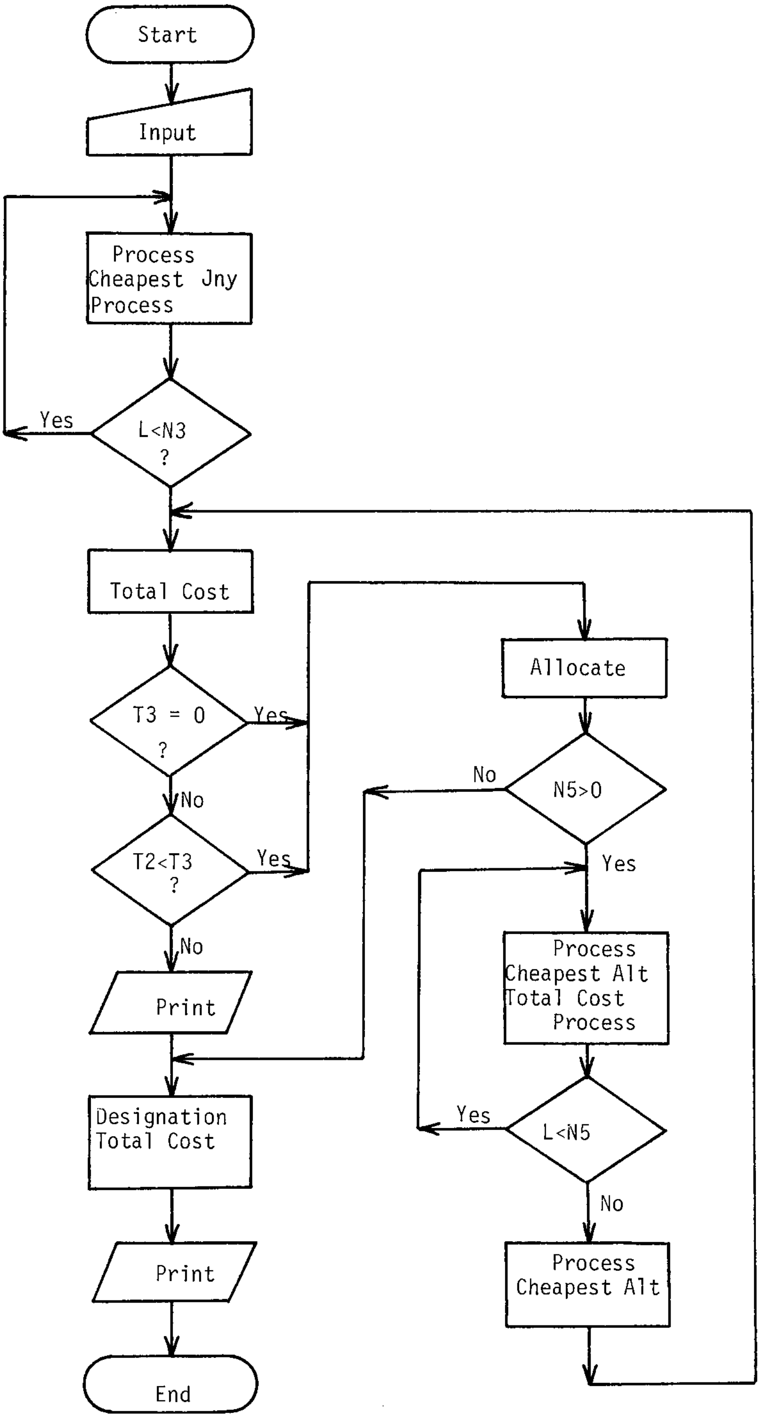
Route Selection, Method 3 - Trial and Error

All the possible transpositions of a given tableau, about all the unused routes, are evaluated and the one giving the lowest cost is selected. This procedure is followed with successive tableaux until a situation is reached wherein no further reduction in cost is achieved. This is then, the optimum solution.

Computer Solution of the Transportation Problem

Route selection by Method 3 above, although more wasteful of computer time than Method 2, is easier to program requiring no further knowledge of the Simplex method. It is employed in the program which follows.

LNPRG2.BAS


Start
Input
Process
Cheapest Jny
Process
L<N3
?
Yes
Total Cost
T3 = 0
?
Yes
No
T2<T3
?
Yes
No
Print
Designation
Total Cost
Print
End
Allocate
N5>0
No
Yes
Process
Cheapest Alt
Total Cost
Process
L<N5
Yes
No
Process
Cheapest Alt

```
10  REM ****************************************************
20  REM - PROGRAM LNPRG2.BAS
30  REM - THE TRANSPORTATION PROBLEM
40  REM - PROGRAM NOMENCLATURE
50  REM - A(J)  -  Number of cargoes available at supply
60  REM            point J
70  REM - B(K)  -  Number of cargoes required at
80  REM            destination K
90  REM - C(J,K)   Cost of journey from supply point J to
100 REM            destination K
110 REM - D(J)  -  Number of the destination with cheapest
120 REM            journey cost from origin J
130 REM - E(J,K)   Number of cargoes despatched from supply
140 REM            point J to destination K
150 REM - F(J)  -  Number of cargoes despatched from supply
160 REM            point J
170 REM - G(K)  -  Number of cargoes received at
180 REM            destination K
190 REM - H(J,K)   Duplicate of Matrix C
200 REM - L(J)  -  Cost of cheapest journey from supply
210 REM            point J
220 REM - M(O)  -  Value of J from which a cargo is
230 REM -          despatched to destination K
240 REM - N(P)  -  Value of K to which a cargo is despatched
250 REM            from supply point J
260 REM - N1    -  Number of supply points
270 REM - N2    -  Number of destinations
280 REM - N3    -  Total number of cargoes residing at the
290 REM            supply points
300 REM - N4    -  Total number of cargoes required at the
310 REM            destinations
320 REM - N5    -  Number of starting points for a search
330 REM - S1    -  Duplicate of value N1
340 REM - S2    -  Duplicate of value N2
350 REM - S3    -  Value used to bypass print statements
360 REM - S4    -  Selected value used to bypass print
370 REM            statements
380 REM - T2    -  Total cost for any given designation
390 REM            of cargoes
400 REM - T3    -  Lowest value of total cost found in the
410 REM            previous iteration
420 REM - V(L)  -  Total cost if alternative rearrangement
430 REM            L is used in the allocation of a starting
440 REM            point
450 REM - W(J,K)   Duplicate of Matrix E
460 REM - X(L)  -  Value of J used with rearrangement L
470 REM - Y(L)  -  Value of K used with rearrangement L
480 REM - PROGRAM DESCRIPTION
490 REM - This program involves a control segment (lines
500 REM - 3010 - 4310) in conjunction with several
510 REM - subroutines
520 REM - CONTROL SEGMENT:
530 REM - LINE 3010   Arrays of more than 10 elements
540 REM - are dimensioned
550 REM - LINES 3020 - 3270  Data for the problem is entered
560 REM - If the total number of cargoes at the supply
570 REM - points does not equal the total number required at
580 REM - the destinations, a message is printed and the
590 REM - program execution is returned for re-entry of data
600 REM - (lines 3160 - 3200).   This reminder is useful
610 REM - when a dummy destination has to be allocated
```

```
620 REM - LINES 3280 - 3300   In developing the program,
630 REM - of intermediate values was required.   The
640 REM - facility to do this is retained so that values
650 REM - may be manually checked
660 REM - LINES 3310 - 3470   Since the subsequent
670 REM - manipulations will employ the variables N1 and N2
680 REM - and the contents of Matrix C, these values are
690 REM - duplicated.   The matrix of journey costs is
700 REM - printed out if desired
710 REM - LINES 3480 - 3740   This segment allocates cargoes
720 REM - to destinations in  the same way as described for
730 REM - the first part of the manual calculation.
740 REM - Cargoes are allotted one at a time up to the
750 REM - total number required (N3 at line 3480).   At
760 REM - each successive iteration within loop L, the
770 REM - route having the lowest cost is selected.   This
780 REM - selection is carried out in the subroutine
790 REM - Cheapest Journey (lines 4350 - 4620).
800 REM - Having selected the cheapest route, the record of
810 REM - journeys held in Mat E is increased by one (line
820 REM - 3520); the tally of cargoes despatched from the
830 REM - appropriate supply point is increased by one, and
840 REM - the value is compared with the number originally
850 REM - available (lines 3530 - 3540)
860 REM - When all cargoes residing at a given supply point
870 REM - have been despatched, the journey costs from that
880 REM - supply point are increased to a very large value
890 REM - so that no more journeys from that point will be
900 REM - selected (lines 3550 - 3570).   When the required
910 REM - number of cargoes has been allocated to a
920 REM - destination, the journey costs to that destination
930 REM - are increased to a very large value also (lines
940 REM - 3580 - 3630).   The matrix of selected journeys is
950 REM - then printed out (lines 3680 - 3740) if required.
960 REM - LINES 3750 - 3900   The total cost for the
970 REM - allocation of journeys made at the previous step
980 REM - is evaluated by subroutine Total Cost (lines 4650
990 REM - to 4750) and the value printed out if required.
1000 REM - If this allocation of cargoes is the first (line
1010 REM - 3780) or if it is cheaper than the previous
1020 REM - allocation (line 3790), then a search is made for
1030 REM - a cheaper allocation.   This search commences at
1040 REM - line 3930.
1050 REM - If the latest allocation is the cheapest then
1060 REM - this is indicated (line 3810), the complete
1070 REM - recommended designation of cargoes is printed out
1080 REM - at line 3870 utilising subroutine Designation
1090 REM - (lines 5620 - 5710), the total cost is evaluated
1100 REM - at line 3880 using subroutine Total Cost (lines
1110 REM - 4650 - 4750) and printed out, and execution is
1120 REM - terminated (line 3920)
1130 REM - LINES 3930 - 4310 Modifications are made
1140 REM - to the previous allocation of journeys,
1150 REM - in a search for a lower total cost.   First,
1160 REM - starting points for the search are allocated.
1170 REM - This is done at line 3940 using subroutine
1180 REM - Allocate (lines 4790 - 4970).   The starting
1190 REM - points found are possible journeys for which no
1200 REM - cargoes have been allocated.   The coordinates
1210 REM - of these possible journeys are stored in Mat X &
1220 REM - Mat Y.
```

```
1230 REM - These starting points are then employed to
1240 REM - determine an alternate allocation of cargoes at
1250 REM - lines 3970 - 4200.  Each starting point is taken
1260 REM - in turn (lines 4020 - 4040) and all the possible
1270 REM - rearrangements (transportation loops) using that
1280 REM - point are evaluated.  This is done at line 4050
1290 REM - using subroutine Cheapest Alternative (lines
1300 REM - 5020 - 5580), and subroutine Total Cost at line
1310 REM - 4060. At line 4070 the cost of the cheapest
1320 REM - rearrangement based on a particular starting
1330 REM - point is stored as V(L), L being the
1340 REM - designation of a particular rearrangement.
1350 REM - We now have a number of possible rearrangements,
1360 REM - one for each starting point.  Each of these
1370 REM - rearrangements is known to be the cheapest for
1380 REM - its particular starting point.  At lines 4140 -
1390 REM - 4200 the cheapest of these rearrangements is
1400 REM - selected.  The value of L found at this step is
1410 REM - used in statements 4210 - 4310 to reallocate
1420 REM - cargoes and then return program executionto line
1430 REM - 3650 for another iteration.
1440 REM - SUBROUTINE CHEAPEST JOURNEY (LINES 4350 - 4620)
1450 REM - This makes the preliminary allocation of cargoes
1460 REM - by selecting routes one at a time on the basis of
1470 REM - cheapest first.  Values of journey costs C(J,K),
1480 REM - are compared in order to find the lowest in each
1490 REM - row of the matrix (tableau) (lines 4370 - 4470).
1500 REM - This value is ascribed to L(S) and the column
1510 REM - number (value of K) is ascribed to D(S), S being
1520 REM - the row number (value of J).  This allocation
1530 REM - is made for values of S from 1 to S1 (the number
1540 REM - of supply points) within the S loop (lines 4350 -
1550 REM - 4510).
1560 REM - The members of the set L(S) are next compared, in
1570 REM - order to find the smallest value (lines 4520 -
1580 REM - 4610).  The value of N generated within the
1590 REM - first loop, and the value of S generated within
1600 REM - the second give the required coordinates of the
1610 REM - cheapest journey.
1620 REM - SUBROUTINE TOTAL COST (LINES 4650 - 4750)
1630 REM - This subroutine calculates the cost for a given
1640 REM - allocation of routes and cargoes.  The number of
1650 REM - carriers allotted to a given route, E(S,T), is
1660 REM - multiplied by the cost of that particular journey
1670 REM - H(S,T), the product being added to the running
1680 REM - total of costs, T2.
1690 REM - SUBROUTINE ALLOCATE (LINES 4790 - 4970)
1700 REM - Possible routes which are not in use are located.
1710 REM - The number of these,N5, is counted, and the
1720 REM - coordinates of each are stored in Matrices X & Y
1730 REM - SUBROUTINE CHEAPEST ALTERNATIVE (LINES 5020 -
1740 REM - 5580)  The subroutine starts off with values of
1750 REM - coordinates J & K corresponding to one of the
1760 REM - starting points for a search.  These values have
1770 REM - previously been retrieved from Matrices X and Y
1780 REM - at lines 4030 & 4040 and 4210 & 4220.
1790 REM - Coordinates J,K represent an unused route.
1800 REM - Taking destination K first, cargoes allocated to
1810 REM - it from other supply points are found and stored
1820 REM - in Matrix M (lines 5160 - 5200).  Taking supply
1830 REM - point J next, journeys from it allocated to
```

```
1840 REM - destinations other than K are stored in Matrix N
1850 REM - (lines 5210 - 5250).  Values of M and N are now
1860 REM - taken in turn, and every possible rearrangement
1870 REM - of journeys about the given starting point (J,K)
1880 REM - is evaluated (lines 5260 - 5460).
1890 REM - One cargo is allocated to route J,K; to
1900 REM - maintain the correct distribution, cargo
1910 REM - allocations to three other routes also have to be
1920 REM - altered, namely locations J,N; M,K; and M,N
1930 REM - (lines 5340 - 5370).  See also Figure 5.8.  The
1940 REM - total cost for the new allocations is now
1950 REM - calculated at line 5380 and subroutine Total Cost
1960 REM - The above sequence is carried out for every
1970 REM - possible rearrangement about a given starting
1980 REM - point.  This gives a number of values of total
1990 REM - journey costs.  Interim use is made of Matrix C
2000 REM - for storage of these (line 5390).  Subroutine
2010 REM - Cheapest Journey is now used again to select the
2020 REM - lowest total cost (line 5470)
2030 REM - The allocation of cargoes corresponding to this
2040 REM - lowest cost rearrangement is made and stored in
2050 REM - Matrix E (lines 5480 - 5580)
2060 REM - SUBROUTINE DESIGNATION (LINES 5620 - 5700)
2070 REM - The final step in the execution of the program is
2080 REM - the printing out of the optimum allocation of
2090 REM - cargoes.  If a dummy destination has been
2100 REM - employed because supply exceeds demand, then
2110 REM - allocations along the fictitious routes are not
2120 REM - printed (line 5650).
2130 REM ***************************************************
3000 REM - CONTROL SEGMENT *********************************
3010 DIM  V(100),X(100),Y(100)
3020 PRINT "NUMBER OF SUPPLY POINTS (MAX 10)";
3030 INPUT N1
3040 FOR J=1 TO N1
3050 PRINT "NUMBER OF CARGOES AT SUPPLY POINT";J;
3060 INPUT A(J)
3070 N3=N3+A(J)
3080 NEXT J
3090 PRINT "NUMBER OF DESTINATIONS (MAX 10)";
3100 INPUT N2
3110 FOR K=1 TO N2
3120 PRINT "NUMBER OF CARGOES REQUIRED AT DESTINATION";K;
3130 INPUT B(K)
3140 N4=N4+B(K)
3150 NEXT K
3160 IF N3=N4 THEN 3210
3170 PRINT "NO. OF CARGOES AVAILABLE NOT EQUAL TO"
3180 PRINT "CARGOES REQUIRED"
3190 N4=0
3200 GOTO 3090
3210 FOR J=1 TO N1
3220 PRINT "TYPE COST OF JOURNEYS FROM SUPPLY POINT";J
3230 PRINT "TO EACH DESTINATION"
3240 FOR K=1 TO N2
3250 INPUT C(J,K)
3260 NEXT K
3270 NEXT J
3280 PRINT "INPUT 1 FOR MIN,2 FOR PARTIAL,3 FOR FULL"
3290 PRINT "INFO ON INTERMEDIATE STEPS"
3300 INPUT S4
```

```
3310 FOR J=1 TO N1
3320 FOR K=1 TO N2
3330 H(J,K)=C(J,K)
3340 E(J,K)=0
3350 NEXT K
3360 NEXT J
3370 S1=N1
3380 S2=N2
3390 IF S4=1 THEN 3480
3400 PRINT
3410 PRINT "MATRIX OF JOURNEY COSTS"
3420 FOR J=1 TO N1
3430 FOR K=1 TO N2-1
3440 PRINT C(J,K);
3450 NEXT K
3460 PRINT C(J,N2)
3470 NEXT J
3480 FOR L=1 TO N3
3490 GOSUB 4340
3500 J=S
3510 K=N
3520 E(J,N)=E(J,N)+1
3530 F(J)=F(J)+1
3540 IF F(J)<A(J) THEN 3580
3550 FOR K=1 TO N2
3560 C(J,K)=1E+12
3570 NEXT K
3580 N=D(J)
3590 G(N)=G(N)+1
3600 IF G(N)<B(N) THEN 3640
3610 FOR J=1 TO N1
3620 C(J,N)=1E+12
3630 NEXT J
3640 NEXT L
3650 IF S4=1 THEN 3750
3660 PRINT
3670 PRINT
3680 PRINT "MATRIX OF JOURNEYS"
3690 FOR J=1 TO N1
3700 FOR K=1 TO N2-1
3710 PRINT E(J,K);
3720 NEXT K
3730 PRINT E(J,N2)
3740 NEXT J
3750 GOSUB 4640
3760 IF S4=1 THEN 3780
3770 PRINT "TOTAL COST";T2
3780 IF T3=0 THEN 3930
3790 IF T2<T3 THEN 3930
3800 PRINT
3810 PRINT "LOWEST TOTAL COST"
3820 FOR J=1 TO N1
3830 FOR K=1 TO N2
3840 E(J,K)=W(J,K)
3850 NEXT K
3860 NEXT J
3870 GOSUB 5610
3880 GOSUB 4640
3890 IF S4>1 THEN 3910
3900 PRINT "TOTAL COST=";T2
3910 PRINT
```

```
3920 GOTO 5720
3930 T3=T2
3940 GOSUB 4780
3950 IF N5>0 THEN 3970
3960 GOTO 3870
3970 FOR J=1 TO N1
3980 FOR K=1 TO N2
3990 W(J,K)=E(J,K)
4000 NEXT K
4010 NEXT J
4020 FOR L=1 TO N5
4030 J=X(L)
4040 K=Y(L)
4050 GOSUB 5000
4060 GOSUB 4640
4070 V(L)=T2
4080 FOR J=1 TO N1
4090 FOR K=1 TO N2
4100 E(J,K)=W(J,K)
4110 NEXT K
4120 NEXT J
4130 NEXT L
4140 FOR L=1 TO N5-1
4150 FOR M=L+1 TO N5
4160 IF V(L)<V(M) THEN 4180
4170 GOTO 4200
4180 NEXT M
4190 GOTO 4210
4200 NEXT L
4210 J=X(L)
4220 K=Y(L)
4230 FOR G=1 TO N1
4240 FOR H=1 TO N2
4250 E(G,H)=W(G,H)
4260 NEXT H
4270 NEXT G
4280 S3=1
4290 GOSUB 5000
4300 S3=0
4310 GOTO 3650
4320 REM ****************************************************
4330 REM - SUBROUTINE CHEAPEST JOURNEY **********************
4340 REM - FIND CHEAPEST JOURNEY FROM EACH SUPPLY POINT
4350 FOR S=1 TO S1
4360 IF S2=1 THEN 4490
4370 FOR N=1 TO S2-1
4380 FOR T=N+1 TO S2
4390 IF C(S,N)<=C(S,T) THEN 4430
4400 IF T<S2 THEN 4470
4410 N=S2
4420 GOTO 4440
4430 NEXT T
4440 L(S)=C(S,N)
4450 D(S)=N
4460 GOTO 4510
4470 NEXT N
4480 GOTO 4510
4490 L(S)=C(S,1)
4500 D(S)=1
4510 NEXT S
4520 FOR S=1 TO S1-1
```

```
4530 FOR N=S+1 TO S1
4540 IF L(S)<=L(N) THEN 4580
4550 IF N<S1 THEN 4600
4560 S=N
4570 GOTO 4610
4580 NEXT N
4590 GOTO 4610
4600 NEXT S
4610 N=D(S)
4620 RETURN
4630 REM ***************************************************
4640 REM - SUBROUTINE TOTAL COST ***************************
4650 T2=0
4660 FOR S=1 TO N1
4670 FOR T=1 TO N2
4680 IF E(S,T)=0 THEN 4700
4690 T2=T2+H(S,T)*E(S,T)
4700 NEXT T
4710 NEXT S
4720 IF S3>0 THEN 4750
4730 IF S4<3 THEN 4750
4740 PRINT "TOTAL COST=";T2
4750 RETURN
4760 REM ***************************************************
4770 REM - SUBROUTINE ALLOCATE *****************************
4780 REM - ALLOCATE STARTING POINTS FOR SEARCH *************
4790 N5=0
4800 FOR J=1 TO N1
4810 FOR K=1 TO N2
4820 C(J,K)=0
4830 NEXT K
4840 NEXT J
4850 FOR J=1 TO 100
4860 X(J)=0
4870 Y(J)=0
4880 NEXT J
4890 FOR J=1 TO N1
4900 FOR K=1 TO N2
4910 IF E(J,K)>0 THEN 4950
4920 N5=N5+1
4930 X(N5)=J
4940 Y(N5)=K
4950 NEXT K
4960 NEXT J
4970 RETURN
4980 REM ***************************************************
4990 REM - SUBROUTINE CHEAPEST ALTERNATIVE *****************
5000 REM - FIND THE CHEAPEST ALTERNATIVE REARRANGEMENT TO
5010 REM - REDUCE TOTAL COST *******************************
5020 FOR G=1 TO N1
5030 M(G)=0
5040 NEXT G
5050 FOR H=1 TO N2
5060 N(H)=0
5070 NEXT H
5080 FOR G=1 TO N1
5090 FOR H=1 TO N2
5100 C(G,H)=0
5110 E(G,H)=W(G,H)
5120 NEXT H
5130 NEXT G
```

```
5140 S1=0
5150 S2=0
5160 FOR M=1 TO N1
5170 IF E(M,K)=0 THEN 5200
5180 S1=S1+1
5190 M(S1)=M
5200 NEXT M
5210 FOR N=1 TO N2
5220 IF E(J,N)=0 THEN 5250
5230 S2=S2+1
5240 N(S2)=N
5250 NEXT N
5260 FOR O=1 TO S1
5270 M=M(O)
5280 FOR P=1 TO S2
5290 N=N(P)
5300 IF S3>0 THEN 5340
5310 IF S4=1 THEN 5340
5320 IF S4=2 THEN 5340
5330 PRINT "J=";J;"K=";K;"M=";M;"N=";N
5340 E(J,K)=E(J,K)+1
5350 E(J,N)=E(J,N)-1
5360 E(M,N)=E(M,N)+1
5370 E(M,K)=E(M,K)-1
5380 GOSUB 4640
5390 C(O,P)=T2
5400 FOR G=1 TO N1
5410 FOR H=1 TO N2
5420 E(G,H)=W(G,H)
5430 NEXT H
5440 NEXT G
5450 NEXT P
5460 NEXT O
5470 GOSUB 4340
5480 M=M(S)
5490 N=N(N)
5500 IF S3>0 THEN 5540
5510 IF S4<3 THEN 5540
5520 PRINT "LOWEST VALUE"
5530 PRINT "J=";J;"K=";K;"M=";M;"N=";N
5540 E(J,K)=E(J,K)+1
5550 E(J,N)=E(J,N)-1
5560 E(M,N)=E(M,N)+1
5570 E(M,K)=E(M,K)-1
5580 RETURN
5590 REM ***************************************************
5600 REM - SUBROUTINE DESIGNATION **************************
5610 REM - PRINT DESIGNATION OF CARGOES ********************
5620 PRINT
5630 FOR J=1 TO N1
5640 FOR K=1 TO N2
5650 IF H(J,K)=0 THEN 5690
5660 IF E(J,K)=0 THEN 5690
5670 PRINT "SEND";E(J,K);"CARGO(ES) FROM SUPPLY POINT";J
5680 PRINT "TO DESTINATION";K
5690 NEXT K
5700 NEXT J
5710 RETURN
5720 END
```

EXAMPLE 5.6

Repeat Example 5.5 using the above program. It will be observed that the program has produced two alternative allocations of minimum cost.

```
RUN
NUMBER OF SUPPLY POINTS (MAX 10)? 3
NUMBER OF CARGOES AT SUPPLY POINT 1 ? 5
NUMBER OF CARGOES AT SUPPLY POINT 2 ? 5
NUMBER OF CARGOES AT SUPPLY POINT 3 ? 10
NUMBER OF DESTINATIONS (MAX 10)? 4
NUMBER OF CARGOES REQUIRED AT DESTINATION 1 ? 3
NUMBER OF CARGOES REQUIRED AT DESTINATION 2 ? 4
NUMBER OF CARGOES REQUIRED AT DESTINATION 3 ? 5
NUMBER OF CARGOES REQUIRED AT DESTINATION 4 ? 8
TYPE COST OF JOURNEYS FROM SUPPLY POINT 1
TO EACH DESTINATION
? 4
? 4
? 3
? 1
TYPE COST OF JOURNEYS FROM SUPPLY POINT 2
TO EACH DESTINATION
? 4
? 7
? 7
? 8
TYPE COST OF JOURNEYS FROM SUPPLY POINT 3
TO EACH DESTINATION
? 4
? 5
? 6
? 7
INPUT 1 FOR MIN,2 FOR PARTIAL,3 FOR FULL
INFO ON INTERMEDIATE STEPS
? 2

MATRIX OF JOURNEY COSTS
 4  4  3  1
 4  7  7  8
 4  5  6  7


MATRIX OF JOURNEYS
 0  0  0  5
 3  0  0  2
 0  4  5  1
TOTAL COST 90


MATRIX OF JOURNEYS
 0  0  0  5
 3  0  1  1
 0  4  4  2
TOTAL COST 90

LOWEST TOTAL COST
```

```
SEND 5 CARGO(ES) FROM SUPPLY POINT 1
TO DESTINATION 4
SEND 3 CARGO(ES) FROM SUPPLY POINT 2
TO DESTINATION 1
SEND 2 CARGO(ES) FROM SUPPLY POINT 2
TO DESTINATION 4
SEND 4 CARGO(ES) FROM SUPPLY POINT 3
TO DESTINATION 2
SEND 5 CARGO(ES) FROM SUPPLY POINT 3
TO DESTINATION 3
SEND 1 CARGO(ES) FROM SUPPLY POINT 3
TO DESTINATION 4

Ok
```

EXAMPLE 5.7

Repeat the above program, but this time assume there is an excess of supply available, the quantities being 600, 600 and 1200 tons at supply points A, B and C respectively.

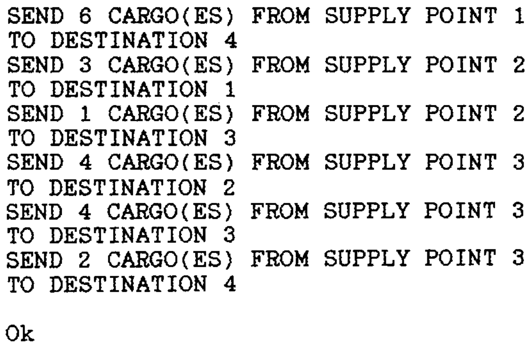
It will be observed that the increase in available supplies has permitted a lower cost allocation to be found. Note also that the progrmm has again produced two alternative allocations of minimum cost.

If Option 1 had been selected at the beginning, only the minimum cost allocation would have been printed out.

```
RUN
NUMBER OF SUPPLY POINTS (MAX 10)? 3
NUMBER OF CARGOES AT SUPPLY POINT 1 ? 6
NUMBER OF CARGOES AT SUPPLY POINT 2 ? 6
NUMBER OF CARGOES AT SUPPLY POINT 3 ? 12
NUMBER OF DESTINATIONS (MAX 10)? 4
NUMBER OF CARGOES REQUIRED AT DESTINATION 1 ? 3
NUMBER OF CARGOES REQUIRED AT DESTINATION 2 ? 4
NUMBER OF CARGOES REQUIRED AT DESTINATION 3 ? 5
NUMBER OF CARGOES REQUIRED AT DESTINATION 4 ? 8
NO. OF CARGOES AVAILABLE NOT EQUAL TO
CARGOES REQUIRED
NUMBER OF DESTINATIONS (MAX 10)? 5
NUMBER OF CARGOES REQUIRED AT DESTINATION 1 ? 3
NUMBER OF CARGOES REQUIRED AT DESTINATION 2 ? 4
NUMBER OF CARGOES REQUIRED AT DESTINATION 3 ? 5
NUMBER OF CARGOES REQUIRED AT DESTINATION 4 ? 8
NUMBER OF CARGOES REQUIRED AT DESTINATION 5 ? 4
TYPE COST OF JOURNEYS FROM SUPPLY POINT 1
TO EACH DESTINATION
? 4
? 4
? 3
? 1
? 0
```

```
TYPE COST OF JOURNEYS FROM SUPPLY POINT 2
TO EACH DESTINATION
? 4
? 7
? 7
? 8
? 0
TYPE COST OF JOURNEYS FROM SUPPLY POINT 3
TO EACH DESTINATION
? 4
? 5
? 6
? 7
? 0
INPUT 1 FOR MIN,2 FOR PARTIAL,3 FOR FULL
INFO ON INTERMEDIATE STEPS
? 2

MATRIX OF JOURNEY COSTS
 4  4  3  1  0
 4  7  7  8  0
 4  5  6  7  0


MATRIX OF JOURNEYS
 0  0  0  2  4
 3  0  0  3  0
 0  4  5  3  0
TOTAL COST 109


MATRIX OF JOURNEYS
 0  0  0  3  3
 3  0  0  2  1
 0  4  5  3  0
TOTAL COST 102


MATRIX OF JOURNEYS
 0  0  0  4  2
 3  0  0  2  1
 0  4  5  2  1
TOTAL COST 96


MATRIX OF JOURNEYS
 0  0  1  4  1
 3  0  0  2  1
 0  4  4  2  2
TOTAL COST 93


MATRIX OF JOURNEYS
 0  0  0  5  1
 3  0  1  1  1
 0  4  4  2  2
TOTAL COST 90
```

```
MATRIX OF JOURNEYS
 0  0  1  5  0
 3  0  0  1  2
 0  4  4  2  2
TOTAL COST 86


MATRIX OF JOURNEYS
 0  0  0  6  0
 3  0  1  0  2
 0  4  4  2  2
TOTAL COST 83


MATRIX OF JOURNEYS
 0  0  0  6  0
 2  0  1  0  3
 1  4  4  2  1
TOTAL COST 83

LOWEST TOTAL COST
```



```
SEND 6 CARGO(ES) FROM SUPPLY POINT 1
TO DESTINATION 4
SEND 3 CARGO(ES) FROM SUPPLY POINT 2
TO DESTINATION 1
SEND 1 CARGO(ES) FROM SUPPLY POINT 2
TO DESTINATION 3
SEND 4 CARGO(ES) FROM SUPPLY POINT 3
TO DESTINATION 2
SEND 4 CARGO(ES) FROM SUPPLY POINT 3
TO DESTINATION 3
SEND 2 CARGO(ES) FROM SUPPLY POINT 3
TO DESTINATION 4

Ok
```

Texts dealing with more advanced linear programming concepts, and their solution by computer, are available (6), (7).

PROBLEMS - CHAPTER 5

1. A manufacturer of rubber gloves produces these in three grades. The manufacturing step is the same for each grade, the differences being in testing and packaging.

Gloves for surgical use are stringently tested, elaborately packed and then sterilised by a radioactive source.

Gloves for clean non-sterile use undergo simple testing and packing procedures and no sterilisation step,

Gloves for household use are packed similarly to those for clean non-sterile use, but both the testing and sterilisation steps are omitted.

Maximise the net daily profit for the factory using the following figures:

Manufacturing capacity: 2,500 pairs/day of all types.

Testing capacity: 2 non-sterile gloves can be tested in the time taken for one surgical glove. Testing equipment can handle both types at a rate equivalent to 1,200 pairs/day of surgical gloves.

Packaging capacity: 1,000 pairs/day of surgical gloves and 2,000 pairs/day of other types.

Profit: On surgical gloves $1 per pair, on non-sterile gloves $0.75 per pair, on household gloves $0.50 per pair.

2. A manufacturer makes a single product, his maximum production rate being 5000 tons per month. Profit after deduction of raw material and production costs is $25/ton. Unsold material is stored in the warehouse at a cost of $1.50/ton per month, the cost being computed on the quantity in storage at the end of each month.

Sales expectations for the next four months are:

Month 1	4800 tons
Month 2	6000 "
Month 3	3000 "
Month 4	4000 "

There are 1000 tons in storage at the beginning of this four-month period. Determine the monthly production rates which will maximise the company profits and leave no unsold material in store at the end of this four-month period.

3. A company is receiving bids for the supply of centrifugal pumps in four capacities. Five pump manufactuters submit bids, which are tabulated in hundreds of dollars, in the table below.

The purchasing company decides to purchase no more than one pump from each supplier. How should the purchases be made in order to minimise the total cost of purchase?

Table of Bids for Pumps, $100s

	Pump Specification	U	V	W	X
Supplier	No Req'd	1	1	1	1
A		60	70	80	90
B		70	75	75	80
C		65	65	70	80
D		no bid	no bid	70	75
E		50	60	no bid	no bid

4. A manufacturer operates 8 batch reactors. Due to differences in age, design and location, production costs vary from reactor to reactor. Each can produce up to 4 batches per week. Five products are manufactured, production costs for each being tabulated below. What assignment of jobs will minimise the total cost of producing the batches enumerated in the table, during one week of operation?

Batch production cost, $1000s						
	Product	A	B	C	D	E
Reactor	No required No available	6	4	5	3	8
1	4	1.0	1.5	1.8	0.9	2.0
2	4	1.2	1.8	2.0	1.1	2.3
3	4	0.9	1.4	1.6	0.9	1.8
4	4	1.0	1.2	1.7	1.0	2.2
5	4	0.9	1.5	1.6	1.1	2.0
6	4	1.1	1.3	1.5	0.9	2.1
7	4	1.2	1.5	1.8	1.1	2.3
8	4	1.1	1.4	1.5	1.0	1.9

5. A chemicals manufacturer operates a batch reactor facility in which he produces three chemical intermediates on a week to week basis. Figures for each batch of these products are as follows:

Product	Production time per batch	Raw Material usage	Profit per batch
A	23 hours	500kg X 200kg Y	$2600
B	8 hours	400kg X	$ 950
C	31 hours	400kg X 400kg Y	$3200

The plant is normally operated for 100 hours per week; operation beyond this time incurs a cost penalty of $200 per hour. Raw material stocks available for next weeks operation consist of:

Material X	2000kg
Material Y	1000kg

Assuming all product can be sold, how should the manufacturer schedule his production for next week?

REFERENCES

1 L.W. Swanson, Linear Programming Basic Theory and Applications, McGraw Hill Book Co., New York, U.S.A., 1980

2 Numerical Computation Unit 5 Linear Programming 1, The Open Unversity Press Milton Keynes, U.K.,2nd Edition, 1979

3 B. Kolman, R.E. Beck, Elementary Linear Programming with Applications, Academic Press,London, U.K., 1980

4 W.A. Spivey, Linear Programming, an Introduction, Collier Macmillan Ltd., Basingstoke, U.K. 1963

5 D.F. Rudd and C.C. Watson, The Strategy of·Process Engineering, John Wiley and Sons, New York, U.S.A., 1968

6 B.E. Gillett, Introduction to Operations Research: a Computer Oriented Algorithmic Approach, McGraw Hill Book Co, New York, U.S.A., 1976

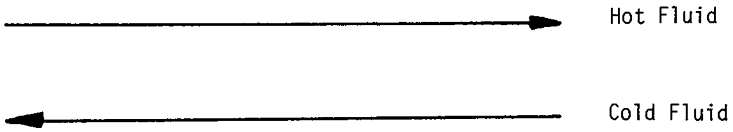
7 B.D. Bunday, Basic Linear Programming, Edward Arnold, London, U.K., 1984

Chapter 6

SOLUTION OF A COUNTERCURRENT STEADY STATE HEAT EXCHANGER

The Problem

We have a truly countercurrent heat exchanger of area A and overall heat transfer coefficient U. The inlet temperatures, mass flowrates and specific heats of each fluid are known. What are the outlet temperatures?



The conventional way to solve this problem is by the use of an effectiveness - number of transfer units (ε - NTU) chart. Such a chart is a representation of relationships derived analytically from first principles. For the countercurrent case the relationship used is:

$$\mathcal{E} = \frac{1 - \exp\left[- \frac{UA}{C_{min}} \cdot (1 - \frac{C_{min}}{C_{max}})\right]}{1 - \frac{C_{min}}{C_{max}} \; . \; \exp\left[- \frac{UA}{C_{min}} \cdot (1 - \frac{C_{min}}{C_{max}})\right]}$$ where:

C=m*c (i.e. the product of mass flowrate and specific heat for one fluid. The fluid having the smaller value of this product is designated by the subscript 'min' the other fluid by the subscript 'max').

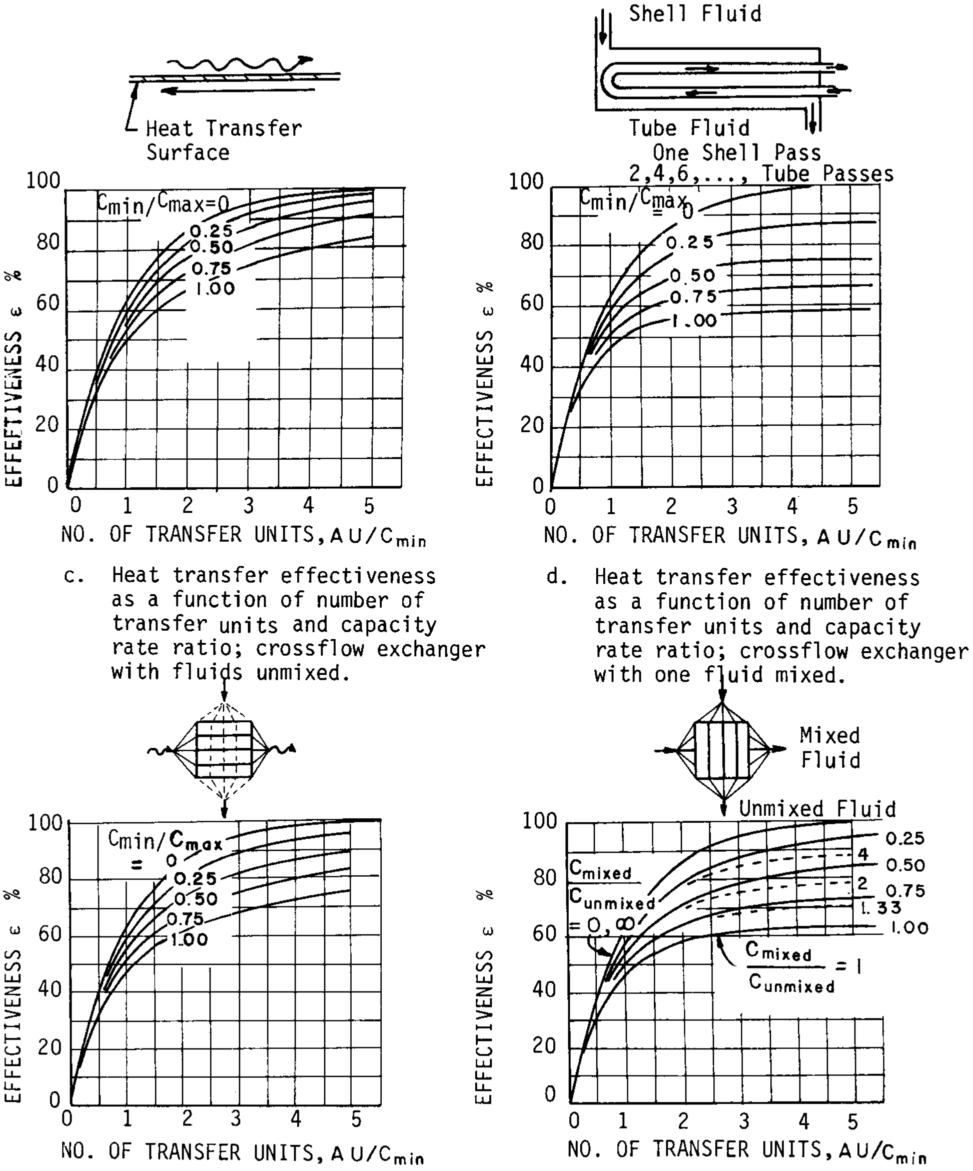
ε, the effectiveness = $\frac{\text{actual temperature change}}{\text{maximum possible temperature change}}$

(where the actual temperature change is evaluated for the fluid with minimum value of mc, and maximum possible change is equal to the difference between entering temperatures). The method is described in textbooks on heat transfer (1), (2), (3).

The relationship is usually expressed graphically, the heat transfer effectiveness ε, being plotted against the group $\frac{UA}{C_{min}}$ (referred to as the number of transfer units NTU).

The ratio $\frac{C_{min}}{C_{max}}$ is employed as a parameter.

Figure 6.1 shows charts for four different surface geometries.



a. Heat transfer effectivenesss as a function of number of transfer units and capacity rate ratio; counterflow exchanger.

b. Heat transfer effectiveness as a function of number of transfer units and capacity rate ratio; 1-2 parallel-counterflow exchanger.

c. Heat transfer effectiveness as a function of number of transfer units and capacity rate ratio; crossflow exchanger with fluids unmixed.

d. Heat transfer effectiveness as a function of number of transfer units and capacity rate ratio; crossflow exchanger with one fluid mixed.

Figure 6.1. Thermal effectiveness of various heat exchanger configurations. Source: Reprinted with permission from W.M. Kays & A.L. London, Compact Heat Exchangers, 3rd Ed., McGraw-Hill Book Co., 1984.

EXAMPLE 6.1

A heat exchanger has an area of $2m^2$, and an overall coefficient of 2500 w/m^2k. Fluids enter the exchanger as follows:

Hot fluid - 3 kg/s entering at 330 K
Cold fluid - 2.2 kg/s entering at 290 K
Specific heat of hot fluid - 1100J/kg,K
Specific heat of cold fluid - 3000 J/kg,K

mc for hot fluid = 1100 $*$ 3 = 3300 J/s,K
mc for cold fluid = 3000 $*$ 2.2 = 6600 J/s,K

$$\frac{C_{min}}{C_{max}} = \frac{3300}{6600} = 0.5$$

$$\frac{UA}{C_{min}} = \text{NTU} = \frac{2 * 2500}{3300} = 1.52.$$ From the chart,

$$\varepsilon = 0.69 = \frac{\text{temperature change of (mc) min fluid (hot fluid)}}{330 - 290}$$

∴ temperature change of hot fluid =

0.69 $*$ 40 = 27.6K. Hot fluid leaves at 339-27.6 = 302.4K.

By a heat balance, cold fluid temperature change =

$$27.6 * \frac{3300}{6600} = 13.8\text{K}.$$ Hence the cold fluid leaves at 290 + 13.8 = 303.8K.

We will now look at a computer solution to this problem, and use the computer to check the above answer. The solution is prepared from basic principles, using the method of finite differences.

Obviously this is a comparatively trivial problem. However, it shows how this method involves the solution of a number of simultaneous equations, and how these may be solved either by iteration or by matrix inversion. Further more, the method may be extended to cover other cases such as crossflow, and the one shell pass multi-tube pass exchanger etc.

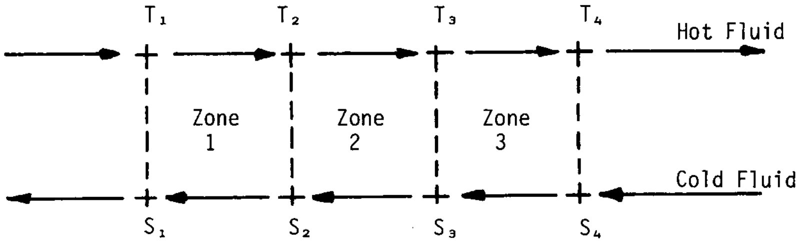
Solution of this problem by finite differences, using a manual solution by relaxation, is given by Dusinberre (4).

The finite difference solution starts by subdividing the exchanger into a number of zones. For example, 3 zones are shown below.

We use the relationships:

$q = UA\Delta T$ and

$q = mcdT = CdT$



We assume U is constant, no heat losses, steady state and constant values of C_H and C_C. Area of complete exchanger is A. We assume that Temperatures T_1 and S_4 are known. The area available for heat transfer = A/N_1, where N_1 = number of zones employed.

The driving force in any zone is assumed to be the difference between the fluid temperatures averaged between inlet and outlet.

Thus, for Zone 1:

$$q_1 = \left(\frac{T_1 + T_2}{2} - \frac{S_1 + S_2}{2}\right) * \frac{UA}{3} = (T_1 + T_2 - S_1 - S_2) * UA/6 \qquad (6.1)$$

Also $q_1 = C_H \; (T_1 - T_2)$ where $C_H = m * c$ for the hot fluid and (6.2)

$$q_1 = C_C \; (S_1 - S_1) \qquad (6.3)$$

Equating 6.1 and 6.2:

$$C_H.(T_1 - T_2) = (T_1 + T_2 - S_1 - S_2) * UA/6$$

$$\therefore \; T_1 - T_2 = \frac{UA.T_1}{6C_H} + \frac{UA.T_2}{6C_H} - \frac{UA.S_1}{6C_H} - \frac{UA.S_2}{6C_H}$$

$$\therefore \; T_2 \, . \left(1 + \frac{UA}{6C_H}\right) = T_1.\left(1 - \frac{UA}{6C_H}\right) + \frac{UA}{6C_H}.\; (S_1 + S_2)$$

$$T_2 = \frac{T_1 \left(1 - \frac{UA}{6C_H}\right) + \frac{UA}{6C_H}.\; (S_1 + S_2)}{(1 + UA/6C_H)}$$

Next, equating 6.1 and 6.3:

$$C_C \; (S_1 - S_2) = (T_1 + T_2 - S_1 - S_2) * UA/6$$

$$S_1 - S_2 = \frac{UA}{6C_C} \, . \, T_1 + \frac{UA.T_2}{6C_c} - \frac{UA.S_1}{6C_c} - \frac{UA.S_2}{6C_c}$$

$$\text{whence, } S_1 = \frac{S_2 \left(1 - \frac{UA}{6C_C}\right) + \frac{UA}{6C_C} \, . \, (T_1 + T_2)}{(1 + UA/6C_c)}$$

Zones 2 and 3 by analogy, result in similar equations, giving the following 6 equations in 6 unknowns (T_1 and S_4 being known):

$$T_2 = \frac{T_1\,(1 - UA/6C_H) + \frac{UA}{6C_H}\,(S_1 + S_2)}{\left(1 + \frac{UA}{6C_H}\right)} \qquad (6.4a)$$

$$T_3 = \frac{T_2\,(1 - UA/6C_H) + \frac{UA}{6C_H}\,(S_2 + S_3)}{\left(1 + \frac{UA}{6C_H}\right)} \qquad (6.4b)$$

$$T_4 = \frac{T_3\,\left(1 - \frac{UA}{6C_H}\right) + \frac{UA}{6C_H}\,(S_3 + S_4)}{\left(1 + \frac{UA}{6C_H}\right)} \qquad (6.4c)$$

$$S_1 = \frac{S_2\,\left(1 - \frac{UA}{6C_C}\right) + \frac{UA}{6C_C}\,(T_1 + T_2)}{\left(1 + \frac{UA}{6C_C}\right)} \qquad (6.4d)$$

$$S_2 = \frac{S_3\,\left(1 - \frac{UA}{6C_C}\right) + \frac{UA}{6C_C}\,(T_2 + T_3)}{\left(1 + \frac{UA}{6C_C}\right)} \qquad (6.4e)$$

$$S_3 = \frac{S_4\,\left(1 - \frac{UA}{6C_C}\right) + \frac{UA}{6C_C}\,(T_4 + T_4)}{\left(1 + \frac{UA}{6C_C}\right)} \qquad (6.4f)$$

There are two methods of solving the above simultaneous equations:-

1) By Matrix Inversion. It is too complex to solve the equations by direct substitution, but it can be done by matrix inversion.

The equations are rearranged with unknowns on the RHS and knowns on the LHS. T_1 and S_4 are known, also the groups UA/6 C_H and UA/6C_C (see Table 6.1). We arrange these into the rows and columns of the Vector and Matrix (see Table 6.2). Solution of the unknowns is then obtained by inversion of the matrix, followed by multiplication of the inverse matrix by the vector.

2) By Iteration. The equations developed above (equations 6.4a to f), are employed in a Seidel iteration, as described in Chapter 2.

Zone	Equation
1	$T_2\ (1 + \frac{UA}{6C_H}) - S_1 .\ \frac{UA}{6C_H} - S_2 .\ \frac{UA}{6C_H} = T_1(1 - \frac{UA}{6C_H})$
2	$T_3\ (1 + \frac{UA}{6C_H}) - T_2(1 - \frac{UA}{6C_H}) - S_2 .\ \frac{UA}{6C_H} - S_3 .\ \frac{UA}{6C_H} = 0$
3	$T_4\ (1 + \frac{UA}{6C_H}) - T_3(1 - \frac{UA}{6C_H}) - S_3 .\ \frac{UA}{6C_H} = S_4 .\ \frac{UA}{6C_H}$
1	$S_1\ (1 + \frac{UA}{6C_C}) - S_2(1 - \frac{UA}{6C_C}) - T_2 .\ \frac{UA}{6C_C} = T_1 .\ \frac{UA}{6C_C}$
2	$S_2\ (1 + \frac{UA}{6C_C}) - S_3(1 - \frac{UA}{6C_C}) - T_2 .\ \frac{UA}{6C_C} - T_3 .\ \frac{UA}{6C_C} = 0$
3	$S_3\ (1 + \frac{UA}{6C_C}) - T_3 .\ \frac{UA}{6C_C} - T_4 .\ \frac{UA}{6C_C} = S_4\ (1 - \frac{UA}{6C_C})$

Table 6.1. Countercurrent Heat Exchange using Three Zones
- the Node Equations

Matrix

Vector

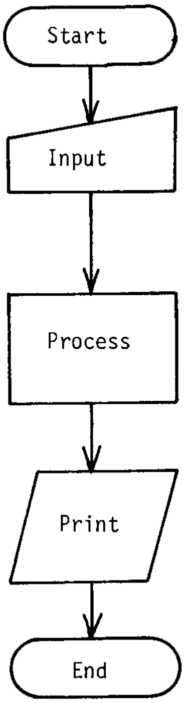
	K = 1	2	3	4	5	6			
J = 1	$(1 + \frac{UA}{6C_H})$	0	0	$- \frac{UA}{6C_H}$	$- \frac{UA}{6C_H}$	0	T_2		$T_1(1 - \frac{UA}{6C_H})$
2	$-(1 - \frac{UA}{6C_H})$	$(1 + \frac{UA}{6C_H})$	0	0	$- \frac{UA}{6C_H}$	$- \frac{UA}{6C_H}$	T_3		0
3	0	$-1(1 - \frac{UA}{6C_H})$	$(1 + \frac{UA}{6C_H})$	0	0	$- \frac{UA}{6C_H}$	T_4	=	$S_4 . \frac{UA}{6C_H}$
4	$- \frac{UA}{6C_C}$	0	0	$(1 + \frac{UA}{6C_C})$	$-(1 - \frac{UA}{6C_C})$	0	S_1		$T_1 . \frac{IA}{6C_C}$
5	$- \frac{UA}{6C_C}$	$- \frac{UA}{6C_C}$	0	0	$(1+ \frac{UA}{6C_C})$	$-(1 - \frac{UA}{6C_C})$	S_2		0
6	0	$- \frac{UA}{6C_C}$	$- \frac{UA}{6C_C}$	0	0	$(1 + \frac{UA}{6C_C})$	S_3		$S_4 . (1 - \frac{UA}{6C_C})$

Table 6.2. Countercurrent Heat Exchange using Three Zones - the Node Equations rearranged for solution by Matrix Algebra

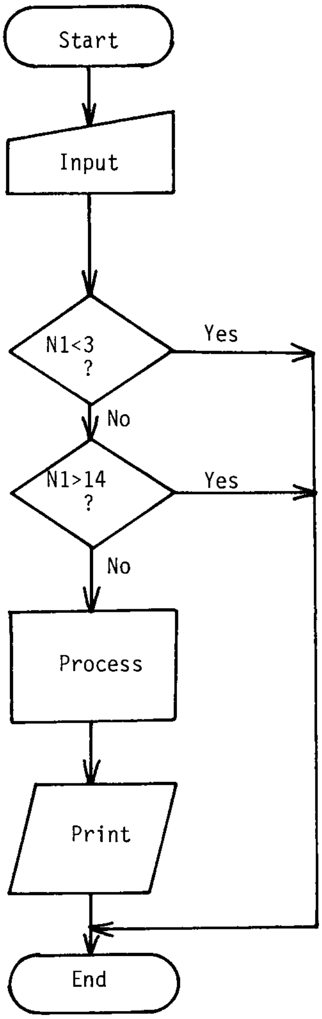
COMPUTER SOLUTION

Two programs are described below. The first employs matrix inversion to solve the simultaneous equations; the second employs iteration. It should be noted that the first program (CONV 1) is unsuitable for use on a desktop computer since it employs matrix algebra.

CONV1.BAS



CONV2.BAS



```
10   REM ****************************************************
20   REM - PROGRAM CONV1.BAS
30   REM - COUNTERCURRENT STEADY STATE HEAT EXCHANGE
40   REM - SOLUTION OF OUTLET TEMPERATURES BY
50   REM - MATRIX INVERSION
60   REM - PROGRAM NOMENCLATURE
70   REM - F1        -   Value of the product UA
80   REM - F2,F3     -   Values of the product mc for the
90   REM                 cold and hot fluids respectively
100  REM - M(J,K)    -   Matrix of coefficients of the unknowns
110  REM - N(J,K)    -   Inverse matrix of M
120  REM - P(J)      -   Vector values from Table 6.2
130  REM - Q2        -   The equivalent to the term UA/6Cc in
140  REM                 equation 6.4a to f etc
150  REM - Q3        -   The equivalent to the term UA/6Ch in
160  REM                 equation 6.4 etc
170  REM - N1        -   The number of zones
180  REM - T1,T2     -   Inlet temperatures of cold & hot
190  REM                 fluids respectively
200  REM - V(J)      -   Values obtained by multiplying the
210  REM                 inverse matrix N by vector P
220  REM - PROGRAM DESCRIPTION
230  REM - LINES 1000 - 1140   Maximum array sizes are
240  REM - declared (line 1000) and values of the variables
250  REM - are entered (lines 1010 - 1080).   Next the two
260  REM - groups which occur so frquently in the equations
270  REM - are evaluated (lines 1090 & 1100).   Then the
280  REM - matrices are dimensioned (lines 1110 - 1140).
290  REM - This is necessary to avoid errors in
300  REM - matrix inversion and multiplication as
310  REM - these operations would otherwise involve matrices
320  REM - of the sizes declared in the DIM statement
330  REM - LINES 1150 - 1510   Values of the coefficients of
340  REM - the unknowns in the equations (Equations 6.4a
350  REM - to f) are entered into Matrix M at the locations
360  REM - indicated by Table 6.2.   The vector values from
370  REM - equation 6.4 and Table 6.2 are, similarly, entered
380  REM - into Matrix P (lines 1470 - 1510).   Note that for
390  REM - all zones other than those at the ends of the
400  REM - exchanger, the vector value is zero.
410  REM - LINES 1520 - 1590   Finally, the matrix inversion
420  REM - and multiplication steps are carried out (lines
430  REM - 1520 & 1530), and the values obtained are printed
440  REM - out (lines 1540 - 1590).   Note that the values of
450  REM - the unknowns, as they occur in Matrix V, are in
460  REM - the sequence T2,T3,T4,etc, S1,S2,etc.
470  REM ****************************************************
1000 DIM  N(20,20),M(20,20),V(20),P(20)
1010 INPUT "VALUE  OF UA";F1
1020 PRINT "VALUE  OF PRODUCT OF MASS FLOWRATE &"
1030 INPUT "SPECIFIC HEAT FOR THE COLD FLUID.  M*C=";F2
1040 PRINT "VALUE  OF PRODUCT M*C FOR THE HOT FLUID "
1050 INPUT "M*C=";F3
1060 PRINT "INLET TEMPERATURES OF COLD & HOT FLUIDS "
1070 INPUT "RESPECTIVELY";T2,T1
1080 INPUT "NUMBER OF ZONES";N1
1090 Q2=F1/(2*N1*F2)
1100 Q3=F1/(2*N1*F3)
1110 MAT N=ZER(2*N1,2*N1)
1120 MAT M=ZER(2*N1,2*N1)
1130 MAT V=ZER(2*N1)
```

```
1140 MAT P=ZER(2*N1)
1150 FOR J=1 TO N1
1160 K=J
1170 M(J,K)=1+Q3
1180 NEXT J
1190 FOR J=N1+1 TO 2*N1
1200 K=J
1210 M(J,K)=1+Q2
1220 NEXT J
1230 FOR J=2 TO N1
1240 K=J-1
1250 M(J,K)=Q3-1
1260 NEXT J
1270 FOR J=N1+1 TO 2*N1-1
1280 K=J+1
1290 M(J,K)=Q2-1
1300 NEXT J
1310 FOR J=N1+1 TO 2*N1
1320 K=J-N1
1330 M(J,K)=-Q2
1340 NEXT J
1350 FOR J=N1+2 TO 2*N1
1360 K=J-1-N1
1370 M(J,K)=-Q2
1380 NEXT J
1390 FOR J=1 TO N1
1400 K=J+N1
1410 M(J,K)=-Q3
1420 NEXT J
1430 FOR J=1 TO N1-1
1440 K=J+1+N1
1450 M(J,K)=-Q3
1460 NEXT J
1470 P(1)=T1*(1-Q3)
1480 P(N1)=T2*Q3
1490 J=N1+1
1500 P(J)=T1*Q2
1510 P(2*N1)=T2*(1-Q2)
1520 MAT N=INV(M)
1530 MAT V=N*P
1540 PRINT "VALUES OF HOT & COLD FLUIDS"
1550 PRINT T1,V(N1+1)
1560 FOR J=1 TO N1-1
1570 PRINT V(J),V(J+N1+1)
1580 NEXT J
1590 PRINT V(N1),T2
1600 END
```

```
10  REM ****************************************************
20  REM - PROGRAM CONV2.BAS
30  REM - COUNTERCURRENT STEADY STATE HEAT EXCHANGE
40  REM - SOLUTION OF OUTLET TEMPERATURES
50  REM - BY SEIDEL ITERATION
60  REM - PROGRAM NOMENCLATURE
70  REM - C1       -   Number of iterations
80  REM - F1       -   Value of the product UA
90  REM - F2,F3    -   Values of the product mc for the cold
100 REM                and hot fluids respectively
110 REM - F4       -   Ratio of approach of successive
120 REM                iterations
130 REM - N1       -   The number of zones
140 REM - Q2       -   The equivalent to the term UA/6Cc in
150 REM                Equation 6.4a to f etc
160 REM - Q3       -   The equivalent to the term UA/6Ch in
170 REM                Equation 6.4 etc
180 REM - S(J)     -   The unknowns S in equations 6.4a to f
190 REM - T(J)     -   The unknowns T in equations 6.4a to f
200 REM - T1,T2    -   Inlet temparatures of cold and hot
210 REM                fluids respectively
220 REM - V(J),W(J)  Values of T(J),S(J) respectively, at
230 REM                the previous iteration
240 REM - PROGRAM DESCRIPTION
250 REM - LINES 1000 - 1150   Array values are declared
260 REM - (line 1000) and values of the variables are
270 REM - entered (lines 1020 - 1110).  A value is then
280 REM - chosen for the desired approach for successive
290 REM - iterations (lines 1140 & 1150); next the two
300 REM - groups UA/NC are evaluated (lines 1160 & 1170)
310 REM - LINES 1180 - 1250   Temperature values are
320 REM - assigned appropriately to the nodes.  Inlet
330 REM - values are assigned first (lines 1180 & 1190),
340 REM - and the arithmetic mean of these values to all
350 REM - other nodes (lines 1200 - 1250)
360 REM - LINES 1260 - 1320   Seidel iteration is performed
370 REM - using the form developed in Equations 6.4a to f; a
380 REM - count is kept of the running total of iterations
390 REM - (line 1320)
400 REM - LINES 1330 - 1440   At each iteration, values of
410 REM - the unknowns, generated at the step above, are
420 REM - compared with the previous values.  If new values
430 REM - (T(J) and S(J)) are within the ratio (F4) of
440 REM - previous values (V(J) and W(J)), then the program
450 REM - terminates (lines 1330 - 1370).  Otherwise, the
460 REM - new values of the unknowns become previous values
470 REM - by assignment to matrices V and W and a further
480 REM - iteration is performed (lines 1400 - 1440)
490 REM - LINES 1450 - 1520   Having satisfied the test at
500 REM - lines 1330 - 1380 above, the number of iterations
510 REM - taken, and the results, are printed out (lines
520 REM - 1460 - 1500).  Finally, the number of iterations
530 REM - (C1) is set to zero, and the user is invited to
540 REM - rerun the data (lines 1510,1520,1100).
550 REM ****************************************************
1000 DIM  T(15),S(15),V(15),W(15)
1010 C1=0
1020 INPUT "VALUE  OF UA";F1
1030 PRINT "VALUE  OF PRODUCT OF MASS FLOWRATE &"
1040 INPUT "SPECIFIC HEAT FOR THE COLD FLUID. M*C=";F2
1050 PRINT "VALUE OF PRODUCT M*C FOR THE HOT FLUID. ";
```

```
1060 INPUT "M*C=";F3
1070 PRINT "INLET TEMPERATURES OF COLD & "
1080 INPUT "HOT FLUIDS RESPECTIVELY";T2,T1
1090 PRINT
1100 PRINT "HOW MANY ZONES BETWEEN 3 & 14 INCLUSIVE";
1110 INPUT N1
1120 IF N1<3 THEN 1530
1130 IF N1>14 THEN 1530
1140 PRINT "RATIO OF APPROACH FOR SUCCESSIVE ";
1150 INPUT "ITERATIONS";F4
1160 Q2=F1/(2*N1*F2)
1170 Q3=F1/(2*N1*F3)
1180 T(1)=T1
1190 S(N1+1)=T2
1200 FOR J=2 TO N1+1
1210 T(J)=(T1+T2)/2
1220 NEXT J
1230 FOR J=1 TO N1
1240 S(J)=(T1+T2)/2
1250 NEXT J
1260 FOR J=2 TO N1+1
1270 T(J)=(T(J-1)*(1-Q3)+Q3*(S(J-1)+S(J)))/(1+Q3)
1280 NEXT J
1290 FOR J=1 TO N1
1300 S(J)=(S(J+1)*(1-Q2)+Q2*(T(J)+T(J+1)))/(1+Q2)
1310 NEXT J
1320 C1=C1+1
1330 FOR J=1 TO N1+1
1340 IF V(J)/T(J)>F4 THEN 1400
1350 IF W(J)/S(J)>F4 THEN 1400
1360 IF V(J)/T(J)<1/F4 THEN 1400
1370 IF W(J)/S(J)<1/F4 THEN 1400
1380 NEXT J
1390 GOTO 1450
1400 FOR J=1 TO N1+1
1410 V(J)=T(J)
1420 W(J)=S(J)
1430 NEXT J
1440 GOTO 1260
1450 PRINT
1460 PRINT "NUMBER OF ITERATIONS=";C1
1470 PRINT "VALUES OF HOT & COLD FLUIDS"
1480 FOR J=1 TO N1+1
1490 PRINT J,T(J),S(J)
1500 NEXT J
1510 C1=0
1520 GOTO 1100
1530 END
```

EXAMPLE 6.2

Solve the problem presented in Example 6.1 using program CONV2.

```
LOAD"A:CONV2
Ok
RUN
VALUE  OF UA? 5000
VALUE  OF PRODUCT OF MASS FLOWRATE &
SPECIFIC HEAT FOR THE COLD FLUID. M*C=? 6600
VALUE OF PRODUCT M*C FOR THE HOT FLUID. M*C=? 3300
INLET TEMPERATURES OF COLD &
HOT FLUIDS RESPECTIVELY? 290,330

HOW MANY ZONES BETWEEN 3 & 14 INCLUSIVE? 3
RATIO OF APPROACH FOR SUCCESSIVE ITERATIONS? 1.0001

NUMBER OF ITERATIONS= 9
VALUES OF HOT & COLD FLUIDS
 1             330           303.9195
 2             318.308       298.0639
 3             309.2314      293.5223
 4             302.1875      290
HOW MANY ZONES BETWEEN 3 & 14 INCLUSIVE? 9
RATIO OF APPROACH FOR SUCCESSIVE ITERATIONS? 1.0001

NUMBER OF ITERATIONS= 20
VALUES OF HOT & COLD FLUIDS
 1             330           303.9355
 2             325.7913      301.8108
 3             321.9187      299.8597
 4             318.3561      298.0681
 5             315.0793      296.4228
 6             312.0658      294.9117
 7             309.2948      293.5235
 8             306.7471      292.2483
 9             304.4049      291.0766
 10            302.2518      290
HOW MANY ZONES BETWEEN 3 & 14 INCLUSIVE? 2
Ok
```

PROBLEMS - CHAPTER 6

1. A thin copper pipe 1cm in diameter is used to convey water through an air space having a temperature of 40^{o}C. The pipe is 100m long, and water flows through it at a rate of $3 * 10^{-4} m^3/s$ If water enters the pipe at a temperature of 10^{o}C, determine its temperature at 10m intervals along the length of the pipe, using the data given below:

Mean Value of Overall Heat Transfer Coefficient	: $25W/m^2K$
Density of Water	: $999\ kg/m^3$
Specific heat of water	: 4800 J/kg,K

Hint: We assume no change in the air temperature and simulate this when using the program, byascribing a very large value to mc for the hot fluid. You can of course check your outlet temperature using the charts given earlier in the chapter.

2. A bio-organic nutrient medium is subjected to a sterilisation cycle before being admitted to a fermenter. The medium leaves the steriliser at 100^{o}C and is rapidly cooled to limit degradation; further cooling lowers the temperature to a value appropriate for fermentation. The cooler consists of two pipes arranged concentrically, medium flowing through the inner pipe, and cooling water passing countercurrently through the annulus.

Use the program CONV2 and the data below to determine the time taken to cool the medium from 100^{o}C to 70^{o}C:

Inlet temperature of water	: 10^{o}C
Flowrate of water	: 10.0kg/s
Specific heat of water	: 4800 J/kg,K
Flowrate of nutrient medium	: 2.5 kg/s
Specific heat of medium	: 4000 J/kg,K
Overall heat transfer coefficient based on onside diameter of inner pipe	: $500\ w/m^2K$
Length of inner pipe	: 150m
Diameter of inner pipe	: 5 cm

3. Using the technique described in this chapter, write programs to solve for the outlet temperatures of:

a. A one shell pass, two tube pass exchanger.

b. A crossflow exchanger, both fluids unmixed.

Use Fig. 6.1 to check your results.

4. Write a program which will determine the heat exchanger area required for a given duty if three of the four terminal temperatures, and film coefficients are known.

REFERENCES

1 J.P. Holman, 3rd Edition, Heat Transfer, McGraw Hill Book Co., New York, U.S.A., 1972

2 F. Kreith, 3rd Edition, Principles of Heat Transfer, International Textbook Co., Scranton Pa, U.S.A., 1973

3 S. Kakac, A.E. Bergles, F. Mayinger, Heat Exchangers, Hemisphere Publishing, New York, U.S.A., 1981

4 G.M. Dusinberre, Heat Transfer Calculations by Finite Differences, International Textbook Co., Scranton Pa, U.S.A. 1961

5 M.E. Leesley, Editor, Computer-aided Process Plant Design, Gulf Publishing Co., Houston, Texas, U.S.A., 1982

Further references to the finite difference technique are listed at the end of Chapter 7.

Chapter 7

SOLUTION OF PROBLEMS IN CONDUCTION HEAT TRANSFER AT STEADY STATE BY THE METHOD OF FINITE DIFFERENCES

This mode of heat transfer is said to take place by the transfer of vibrational energy between molecules. It is the mechanism which accounts for the transfer of heat within solid bodies. In the case of fluids, conduction heat transfer is generally much less significant than heat transfer by other modes such as convection, condensation or boiling.

The application of conduction theory is useful in the calculation of heat flows through insulating layers, and in the design of furnace enclosures. In the latter case, it is also important to be able to determine temperature distribution since this governs the phenomena of spalling and creep.

In this chapter, methods will be detailed for dealing with conduction heat transfer under steady state conditions (i.e. invariant with respect to time). The next chapter will deal with the extension of the arguments to include transient behaviour.

When heat flows through a body by the mechanism known as conduction, the quantity of heat flowing per unit of time is related to the other variables by the general expression:

$$\text{Flowrate} \;\propto\; \frac{\text{Driving Force}}{\text{Resistance}}$$

In this case the driving force is difference in temperature and the resistance results from the shape and physical properties of the conducting body. Thus in the case of a flat slab of material such as that shown in Figure 7.1, this relationship is written for conduction between the vertical faces shown as:

$$\frac{dQ}{dt} = -kA\,\frac{dT}{dx} \tag{7.1}$$

This equation is known as the Fourier equation (1), and is directly useful in many cases where the system can be described in one dimension and easily integrated. These cases are thoroughly treated in many textbooks and some examples are given later.

Where A = area across which heat transfer occurs ;
k = thermal conductivity of the conducting material (e.g. in W/m K);
Q = amount of heat transferred;
t = time;
x = distance;
T = temperature

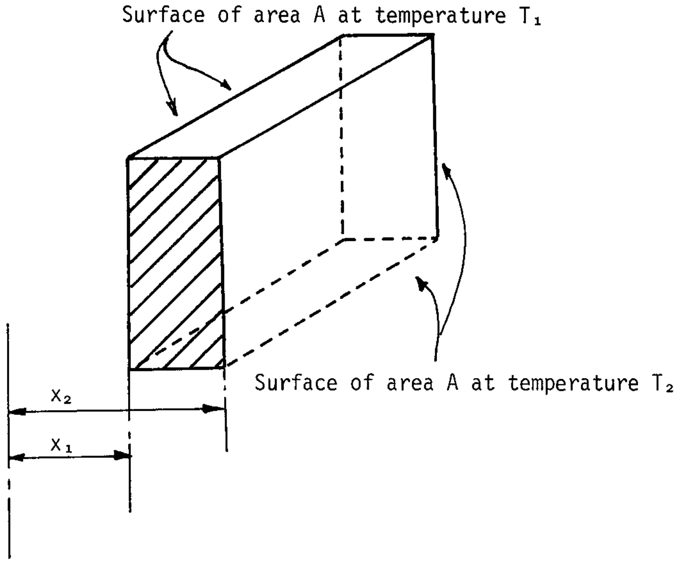

Surface of area A at temperature T_1
Surface of area A at temperature T_2
X_2
X_1

Figure 7.1. Heat conduction through a flat slab.

The more general case is arrived at by applying the above relationship to conduction in three dimensions. Thus consider a homogeneous solid body, having uniform values throughout of the properties ρ, c and k (density, specific heat, thermal conductivity).

Due to various heat transfer mechanisms operating at the surfaces of this body, there may be a wide range of temperatures within it. An element within the body is shown in Figure 7.2. This can exchange heat by conduction with six other contiguous cubical elements. It is assumed that the behaviour of each element can be represented by a point or node at its centre. Using the nomenclature of finite differences, a heat balance around the central element (element 1)) can be written. The subscripts refer to conditions within a given element of the same number:

$$k\,\frac{\Delta y\Delta z}{\Delta x}\,.\,(T_3 - T_1) + k\,\frac{\Delta y\Delta z}{\Delta x}\,.\,(T_4 - T_1) + k\,\frac{\Delta y\Delta x}{\Delta z}\,(T_6 - T_1) +$$

$$k\,\frac{\Delta y\Delta x}{\Delta z}\,.\,(T_7 - T_1) + k\,\frac{\Delta x\Delta z}{\Delta y}\,.\,(T_5 - T_1) + k\,\frac{\Delta x\Delta z}{\Delta y}\,.\,(T_2 - T_1) =$$

$$\frac{\Delta x\ \Delta y\ \Delta z\ \rho c\ (T_1^* - T_1)}{\Delta\theta} \tag{7.2}$$

where T_1^* = new value of temperature of node 1 after exchange of heat for time Δθ.

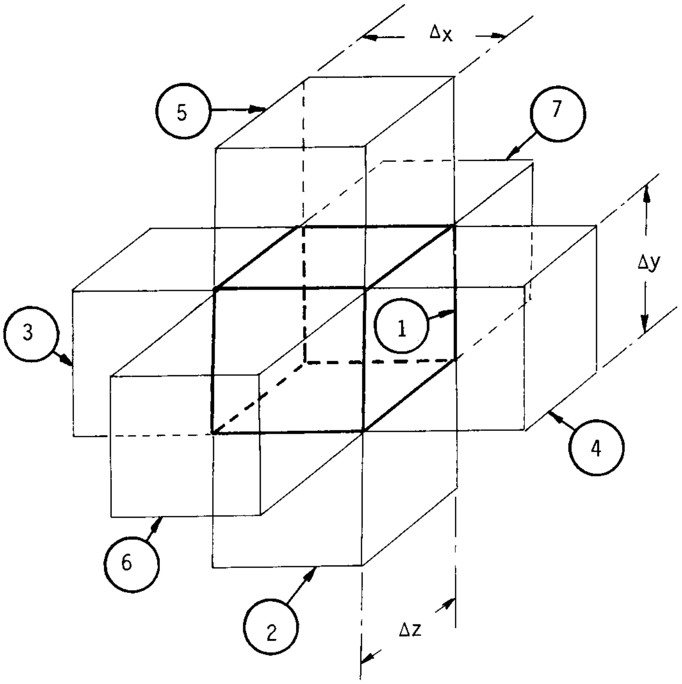


Figure 7.2. Conduction in three dimensions.

Dividing through by Δx . Δy . Δz and rearranging, this can be rewritten as:

$$\frac{\frac{k\,(T_3 - T_1)}{\Delta x} - \frac{k\,(T_1 - T_4)}{\Delta x}}{\Delta x} + \frac{\frac{k\,(T_6 - T_1)}{\Delta z} - \frac{k\,(T_1 - T_7)}{\Delta z}}{\Delta z} +$$

$$\frac{\frac{k\,(T_2 - t_1)}{\Delta y} - \frac{k\,(T_1 - T_5)}{\Delta y}}{\Delta y} = \frac{\rho c\,(T_1^{*} - T_1)}{\Delta\theta} \tag{7.3}$$

In the limit, when Δx, Δy, Δz, and Δθ → 0, then

$$\frac{\delta}{\delta x}\left(\frac{k\delta T}{\delta x}\right) + \frac{\delta}{\delta y}\left(\frac{k\delta T}{\delta y}\right) + \frac{\delta}{\delta z}\left(\frac{k\delta T}{\delta z}\right) = \rho c\,\frac{\delta T}{\delta\theta} \quad \text{or:} \tag{7.4}$$

$$\frac{\delta^2 T}{\delta x^2} + \frac{\delta^2 T}{\delta y^2} + \frac{\delta^2 T}{\delta z^2} = \frac{1}{a} \cdot \frac{\delta T}{\delta\theta} \quad \text{where } a = \frac{k}{\rho c} \tag{7.5}$$

This equation is a form of the Fourier equation expressing conduction in three dimensions. Obvious simplifications can be made if less than three dimensions need be considered. When the body is at steady state conditions, then the right hand side of equations 7.2 to 7.5 = 0. We thus get an equation known as the Laplace equation (1), (2):

$$\frac{\delta^2 T}{\delta x^2} + \frac{\delta^2 T}{\delta y^2} + \frac{\delta^2 T}{\delta z^2} = 0 \qquad (7.6)$$

For other applications, heat generation terms can also be included.

Although partial differential equations usefully describe what is occurring, they are often of little practical application, due to the complex geometry of the solid body under consideration. Furthermore, it is usually necessary to consider heat transfer occurring at the surfaces of the body by convection and radiation. In such cases, finite difference methods of solution are much more useful and applicable (1), (2), (3), (4).

In the derivation of the Fourier equation, the elements considered were assumed to be differentially small. By solution of the equation, is obtained the 'correct' solution to a problem. If elements finite in size are considered, the solution obtained will be less accurate. On the other hand, more complex situations can be dealt with. The procedure will now be described in more detail.

Bodies Having Surfaces Formed of Planes Arranged at Right Angles

In such cases, the body is subdivided into a convenient number of regular elements having plane boundaries. (Where the body has a uniform cross-section and relatively great length, a two dimensional system need only be considered). Each subdivision is considered to be a lumped system, that is, it can be represented by a point or node at its centre. An equation such as equation 7.2 is then written around each node in turn.

In the case of steady state problems, certain boundary conditions are known, or must be assumed, but otherwise nodal temperatures are unknown. Where the equations generated are linear, a variety of methods may be used to solve for the unknown values, namely trial and error, determinants, relaxation and iteration. Where the equations are non-linear (for example where radiation is involved) then some of these methods may not be available.

Examples of Finite Difference Solutions to Steady State Conduction Problems, Using Linear or Rectilinear Nodal Networks

EXAMPLE 7.1

(a) Estimate the temperature distribution within a duct having the typical cross section shown in Figure 7.3a. Hence calculate the heat loss from the duct per unit of length, assuming the thermal conductivity of the material to be 1 W/mK. The inner face of the duct is constant at 500 K and the outer face at 320 K.

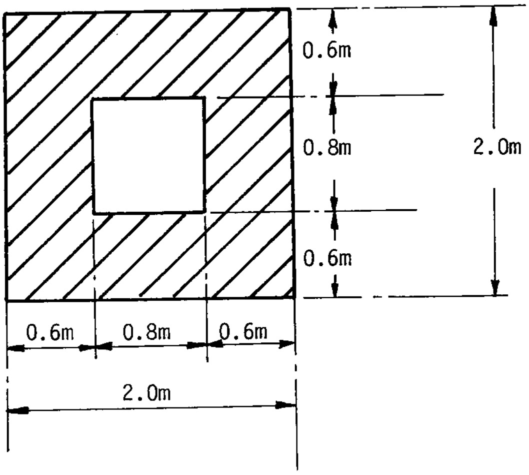


Figure 7.3a.

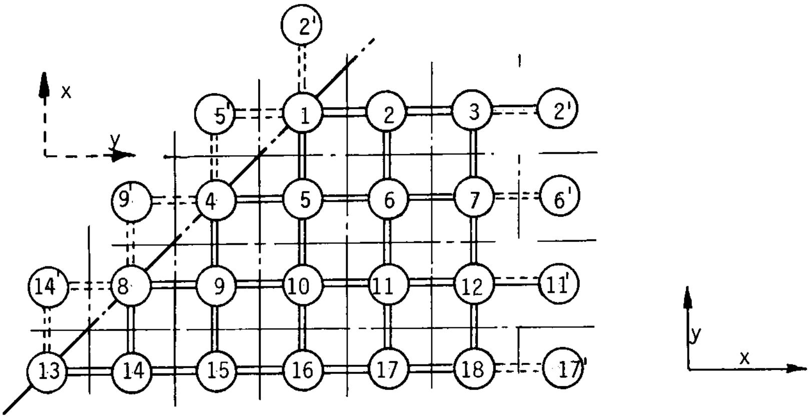


Figure 7.3b.

This is a well known example which has been repeated in a number of references. It is nevertheless included here as it provides an excellent example of the technique (1), (2).

On arguments of symmetry, only one eighth of the figure need be considered. On this a rectilinear network of nodes can be arranged. Such a network is shown in Figure 7.3b. Should a more accurate representation be required, then a finer network could be drawn. The network shown will not give an accurate result, but serves to illustrate the method.

First write the equations representing each node. The subscripts indicate the value of the property at the node under consideration.

$T_1 = T_2 = T_3 = T_2' = 500$ K

$T_{13} = T_{14} = T_{15}$etc. $= 320$ K

It should be noted that nodes such as 9'. 6', 11' and 14' are also mirror images of the unsuperscripted node of the same number.

For the remaining nodes, heat balance equations must be written

Node 4:

$$\frac{kA_C}{\Delta x}(T_5' - T_4) + \frac{kA_C}{\Delta x}(T_5 - T_4) + \frac{kA_C}{\Delta y}(T_9' - T_4) + \frac{kA_C}{\Delta y}(T_9 - T_4)$$

= sum of heat flows to and from node 4

= zero at steady state.

Since $\Delta x = \Delta y$, this equation simplifies to:

$$\frac{kAc}{\Delta x}(2T_5 + 2T_9 - 4T_4) = 0$$

The area for conduction between nodes, for unit depth normal to the cross section, $A_C = 0.2m^2$. Distance between nodes, $\Delta x = 0.2m$. Thermal conductivity, $k = 1$ W/mK. Therefore the above equation, without cancellation reduces to:

$$2T_5 + 2T_9 - 4T_4 = 0.$$

Equations for the remaining nodes are similarly written, and are tabulated in Table 7.1.

There are a number of ways by which the above equations may be solved. These will be discussed in turn in this and the examples which follow.

1. Direct Solution by Matrix Algebra.

In the example above there are as many equations as there are unknowns, consequently a direct analytical solution is possible. This can be done using determinants. The first step is to rewrite the above equations, substituting the given boundary conditions. Thus we obtain Table 7.2.

Node	Equation	Equation
4	$-4T_4 + 2T_5 + 2T_9 = 0$	$-4T_4 + 2T_5 + 2T_9 = 0$
5	$T_1 + T_4 - 4T_5 + T_6 + T_{10} = 0$	$T_4 - 4T_5 + T_6 + T_{10} = -500$
6	$T_2 + T_5 - 4T_6 + T_7 + T_{11} = 0$	$T_5 - 4T_6 + T_7 + T_{11} = -500$
7	$T_3 + 2T_6 - 4T_7 + T_{12} = 0$	$2T_6 - 4T_7 + T_{12} = -500$
8	$-4T_8 + 2T_9 + 2T_{14} = 0$	$-4T_8 + 2T_9 = -640$
9	$T_4 + T_8 - 4T_9 + T_{10} + T_{15} = 0$	$T_4 + T_8 - 4T_9 + T_{10} = -320$
10	$T_5 + T_9 - 4T_{10} + T_{11} + T_{16} = 0$	$T_5 + T_9 - 4T_{10} + T_{11} = -320$
11	$T_6 + T_{10} - 4T_{11} + T_{12} + T_{17} = 0$	$T_6 + T_{10} - 4T_{11} + T_{12} = -320$
12	$T_7 + 2T_{11} - 4T_{12} + T_{18} = 0$	$T_7 + 2T_{11} - 4T_{12} = -320$

Table 7.1.   Table 7.2.

These equations are written in matrix notation in Table 7.3 or, in general terms, $\mathbf{A}\,\mathbf{T} = \mathbf{B}$ where **A** represents a square matrix of the ninth order, and **T** and **B** are column vectors also of ninth order. The solution of this equation is

$\mathbf{T} = \mathbf{A}^{-1}\mathbf{B}$ where $\mathbf{A}^{-1}$ represents the inverse or reciprocal of the above matrix.

Rules for obtaining the inverse of a matrix have been briefly discussed in Chapter 2.

$$\begin{array}{cccc}
\begin{array}{ccccccccc} 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \end{array} & & & \\
\left[\begin{array}{ccccccccc}
-4 & +2 & 0 & 0 & 0 & +2 & 0 & 0 & 0 \\
+1 & -4 & +1 & 0 & 0 & 0 & +1 & 0 & 0 \\
0 & +1 & -4 & +1 & 0 & 0 & 0 & +1 & 0 \\
0 & 0 & +2 & -4 & 0 & 0 & 0 & 0 & +1 \\
0 & 0 & 0 & 0 & -4 & +2 & 0 & 0 & 0 \\
+1 & 0 & 0 & 0 & +1 & -4 & +1 & 0 & 0 \\
0 & +1 & 0 & 0 & 0 & +1 & -4 & +1 & 0 \\
0 & 0 & +1 & 0 & 0 & 0 & +1 & -4 & +1 \\
0 & 0 & 0 & +1 & 0 & 0 & 0 & +2 & -4
\end{array}\right]
&
\left[\begin{array}{c} T_4 \\ T_5 \\ T_6 \\ T_7 \\ T_8 \\ T_9 \\ T_{10} \\ T_{11} \\ T_{12} \end{array}\right]
& = &
\left[\begin{array}{c} 0 \\ -500 \\ -500 \\ -500 \\ -640 \\ -320 \\ -320 \\ -320 \\ -320 \end{array}\right]
\end{array}$$

Table 7.3.

Assuming matrix algebra is available, the simple Basic program DATA4, given in Chapter 2 may be employed:

```
RUN
NUMBER OF UNKNOWNS? 9
INPUT MATRIX
? -4,2,0,0,0,2,0,0,0
? 1,-4,1,0,0,0,1,0,0
? 0,1,-4,1,0,0,0,1,0
? 0,0,2,-4,0,0,0,0,1
? 0,0,0,0,-4,2,0,0,0
? 1,0,0,0,1,-4,1,0,0
? 0,1,0,0,0,1,-4,1,0
? 0,0,1,0,0,0,1,-4,1
? 0,0,0,1,0,0,0,2,-4
INPUT VECTOR
? 0,-500,-500,-500,-640,-320,-320,-320,-320
 387.204
 421.756
 432.743
 435.316
 336.326
 352.652
 367.077
 373.9
 375.779
Ok
```

Alternatively, the iterative procedure described in Chapter 2 may be employed, using Program Data 5. This yields a closely similar solution:

```
LOAD"A:DATA5
Ok
RUN
INPUT J FOR JACOBI METHOD, GS FOR GAUSS-SEIDEL
? GS
NUMBER OF UNKNOWNS? 9
COEFFICIENTS OF EQUATION  1
? -4? 2? 0? 0? 0? 2? 0? 0? 0
COEFFICIENTS OF EQUATION  2
? 1? -4? 1? 0? 0? 0? 1? 0? 0
COEFFICIENTS OF EQUATION  3
? 0? 1? -4? 1? 0? 0? 0? 1? 0
COEFFICIENTS OF EQUATION  4
? 0? 0? 2? -4? 0? 0? 0? 0? 1
COEFFICIENTS OF EQUATION  5
? 0? 0? 0? 0? -4? 2? 0? 0? 0
COEFFICIENTS OF EQUATION  6
? 1? 0? 0? 0? 1? -4? 1? 0? 0
COEFFICIENTS OF EQUATION  7
? 0? 1? 0? 0? 0? 1? -4? 1? 0
COEFFICIENTS OF EQUATION  8
? 0? 0? 1? 0? 0? 0? 1? -4? 1
COEFFICIENTS OF EQUATION  9
? 0? 0? 0? 1? 0? 0? 0? 2? -4
INPUT VECTOR TERMS
? 0? -500? -500? -500? -640? -320? -320? -320? -320
NUMBER OF ITERATIONS= 16
V( 1 )= 387.0281
V( 2 )= 421.6482
V( 3 )= 432.6691
V( 4 )= 435.2617
V( 5 )= 336.2427
V( 6 )= 352.5548
V( 7 )= 367.0021
V( 8 )= 373.8449
V( 9 )= 375.7379
Ok
```

The temperatures at the various nodes are listed in Table 7.4. These values are shown superimposed upon the node diagram in Figure 7.4.

Node	Temperature
4	387.2
5	421.8
6	432.7
7	435.3
8	336.3
9	352.7
10	367.1
11	373.9
12	375.8

Table 7.4. Values of Node Temperatures.

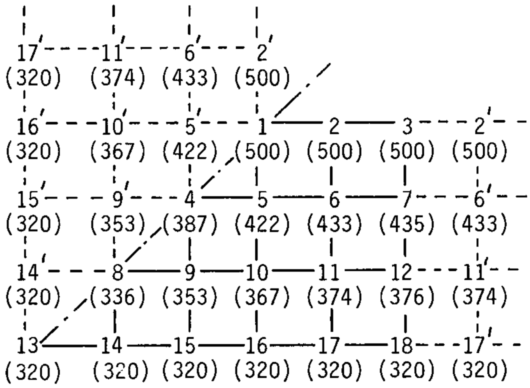


Figure 7.4. Values of temperature at the nodes.

2. Solution by Manual Iteration

The equation can also be solved by iteration, proceeding from guessed values of the temperatures. This can be done manually by the method of 'Relaxation', which was much favoured before the advent of computers (4).

If incorrect values of the temperatures are chosen, then the RHS of each equation listed in Table 1 will not be zero. It will have a value, called its residual value, R.

The method proceeds by adjusting the guessed temperature values so as to reduce the residuals until they are acceptably small.

Substituting known boundary values, the equations of Table 7.1 are rewritten as in Table 7.5.

The effect of unit changes in node temperature upon the values of the residuals is next tabulated (Table 7.6).

Node	Equation
4	$-4T_4 + 2T_5 + 2T_9 = R_4$
5	$500 + T_4 - 4T_5 + T_6 + T_{10} = R_5$
6	$500 + T_5 - 4T_6 + T_7 + T_{11} = R_6$
7	$500 + 2T_6 - 4T_7 + T_{12} = R_7$
8	$640 - 4T_8 + 2T_9 = R_8$
9	$320 + T_4 + T_8 - 4T_9 + T_{10} = R_9$
10	$320 + T_5 + T_9 - 4T_{10} + T_{11} = R_{10}$
11	$320 + T_6 + T_{10} - 4T_{11} + T_{12} = R_{11}$
12	$320 + T_7 + 2T_{11} - 4T_{12} + R_{12}$

Table 7.5. Equations arranged for solution by Relaxation.

	ΔR_4	ΔR_5	ΔR_6	ΔR_7	ΔR_8	ΔR_9	ΔR_{10}	ΔR_{11}	ΔR_{12}
$\Delta T_4 = 1$	-4	+1	0	0	0	+1	0	0	0
$\Delta T_5 = 1$	+2	-4	+1	0	0	0	+1	0	0
$\Delta T_6 = 1$	0	+1	-4	+2	0	0	0	+1	0
$\Delta T_7 = 1$	0	0	+1	-4	0	0	0	0	+1
$\Delta T_8 = 1$	0	0	0	0	-4	+1	0	0	0
$\Delta T_9 = 1$	+2	0	0	0	+2	-4	+1	0	0
$\Delta T_{10} = 1$	0	+1	0	0	0	+1	-4	+1	0
$\Delta T_{11} = 1$	0	0	+1	0	0	0	+1	-4	+2
$\Delta T_{12} = 1$	0	0	0	+1	0	0	0	+1	-4

Table 7.6. Relaxation Pattern.

Temperatures for the nodes are now assumed, and corresponding values of the residuals are worked out. Values of these are now put at the head of the Relaxation Table (Table 7.7) and stepwise temperature adjustments are made. The largest residuals are adjusted first. In this case, the value of 400 has been ascribed to every unknown temperature. Then

$R_4 = -1600 + 800 + 800 = 0$

$R_5 = 500 + 400 - 1600 + 400 + 400 = +100$

The other residuals are similarly computed and entered into the table. The first steps in the table have been numbered, the actions occurring at each step being listed below. It should be noted that as each alteration to a

TABLE 7.7. Relaxation Table.

	T_4	R_4	T_5	R_5	T_6	R_6	T_7	R_7	T_8	R_8	T_9	R_9	T_{10}	R_{10}	T_{11}	R_{11}	T_{12}	R_{12}
1	400	0	400	+100	400	+100	400	+100	400	-160	400	-80	400	-80	400	-80	400	-80
2									-40									
3									$\boxed{360}$	0		-120						
4											-30							
5		-60								-60	$\boxed{370}$	0		-110				
6													-25					
7				+75								-25	$\boxed{375}$	-10		+105		
8															-20			
9						+80								-30	$\boxed{380}$	-25		-120
10	400	-60	400	+75	400	+80	400	+100	360	-60	370	-25	375	-30	380	-25	400	-120
11																	-30	
12								+70								-55	$\boxed{370}$	0
13					+20													
14				+95	$\boxed{420}$	0		+110								-35		
15							+25											
16						+25	$\boxed{425}$	+10										
17			+20															
18		-20	$\boxed{420}$	+15		+45								-10				
19	400	-20	420	+15	420	+45	425	+10	360	-60	370	-25	375	-10	380	-35	370	+25
20									-15									
21									$\boxed{345}$	0		-40						
22					+10													
23				+25	$\boxed{430}$	+5		+30										
24															-10			
25						-5								-20	$\boxed{370}$	+5		+5
26											-10							
27		-40								-20	$\boxed{360}$	0		-30				
28	-10																	
29	$\boxed{390}$	0		+15								-10						
30							+7											
						+2	$\boxed{432}$	+2										+12
									-5									
									$\boxed{340}$	0		-15						
34	390	0	420	+15	430	+2	432	+2	340	0	360	-15	375	-30	370	+15	370	+12
													-7					
				+8								-22	$\boxed{368}$	-2		+8		
											-5							
		-10								-10	$\boxed{355}$	-2		-7				
																	+3	
								+5								+11	$\boxed{373}$	0
															+3			
						+5								-4	$\boxed{373}$	-1		+6
	-2																	
	$\boxed{388}$	-2		+6								-4						
									-2									
									$\boxed{338}$	-2		-6						
	388	-2	420	+6	430	+5	432	+5	338	-2	355	-6	368	-4	373	-1	373	+6

temperature is made, its value has been entered thus: $\boxed{360}$

Step 1 : enter estimated temperatures, and corresponding values of residuals calculated from Table 7.5;

Step 2 : the largest residual (R_8) is selected and an alteration to T_8 made which will result in value zero for this residual;

Step 3 : the new values of T_8, R_8 and the other residuals affected, are entered;

Steps 4,5 : the same procedure is repeated with the next largest residual, R_9;

Steps 6-9 : the procedure is repeated with residuals R_{10} and R_{11};

Step 10 : values of temperatures are entered and residuals calculated using the equations of Table 7.5. This is done as a check on the accuracy of the work. In this case no errors have occurred.

The calculation proceeds in this fashion until the desired degree of accuracy has been attained. At line 34 a discrepancy in the value of R_{11} may be noted. It is unnecessary to work back to correct the error. The calculation proceeds using the newly calculated values of the residuals.

3. Calculation of Heat Loss

From the temperature distribution obtained, and shown in Figure 7.4 the heat conducted along each path through the network can be calculated, as shown below. The heat conducted through each row of nodes should be the same, and indeed the values calculated are closely similar.

Considering one-eighth of the figure only:

Heat conducted node 1 to 5 =

$\frac{kA}{x} (T_1 - T_5) = 1(500 - 422) \qquad = 78$

Heat conducted node 2 to 6 similarly=

$1(500 - 433) \qquad = 67$

Heat conducted node 3 to 7 =

$\frac{1}{2}(500 - 435) \qquad = \underline{32\frac{1}{2}}$

total $\qquad = 177\frac{1}{2}$W

Similarly for the second row of nodes:

Nodes 4 to 9	=	34
Nodes 5 to 10	=	55
Nodes 6 to 11	=	59
Nodes 7 to 12	=	29½
total	=	177½W

or for the third row:

Nodes 8 to 14	=	16
Nodes 9 to 15	=	33
Nodes 10 to 16	=	47
Nodes 11 to 17	=	54
Nodes 12 to 18	=	28
total	=	178W

Total heat loss per metre length of the section thus = 8 * 178 = 1424W.

EXAMPLE 7.2

Repeat calculation A, but assuming the shape is made up of two layers of material of different thermal conductivities as shown in Figure 7.5.

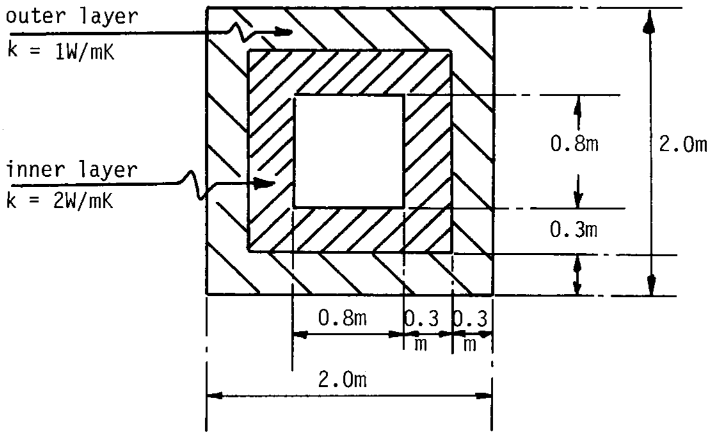


Figure 7.5.

Further assume that heat transfer between the duct walls and the surroundings are described by combined coefficients of heat transfer by radiation and convection of 30 W/m²K for the inner surface, and 7 W/m²K for the outer surface. Temperature of gas flowing in the duct is 500 K and ambient temperature is 290 K.

Since surface temperatures are unknown, a much greater number of nodes must be considered. Using the same network as before (Figure 7.3b), equations are written as follows:

Node 1 :

$$\left(\frac{\Delta x}{2} + \frac{\Delta x}{2}\right) h_1 .\ (T_F - T_1) + \frac{\Delta y}{2\Delta x} .\ k_1 .\ (T_2 - T_1) + \frac{\Delta x}{\Delta y} .\ k_1\ (T_5 - T_1) + T$$

$\frac{\Delta x}{\Delta y} .\ k_1\ (T_5' - T_1) + \frac{\Delta y}{2\Delta x} .\ k_1 .\ (T_2' - T_1) = 0$ where T_F = gas temperature within the duct.

Since $\Delta x = \Delta y = 0.2$; $T_F = 550$; $k_1 = 2$; and $h_1 = 30$, this simplifies to:

$$0.2 * 30\ (550 - T_1) + T_2 - T_1 + 2\ (T_5 - T_1) + 2\ (T_5' - T_1) + T_2' - T_1 = 0$$

$$\therefore\ 3300 - 12T_1 + 2T_2 + 4T_5 = 0$$

Node 2 :

$$\frac{\Delta y}{\Delta x} .\ k_1 .\ (T_6 - T_5) + \frac{\Delta x}{\Delta y} .2. \frac{1}{1/k_1 + 1/k_2} .\ (T_{10} - T_5) +$$

$$\frac{\Delta y}{\Delta x} .\ k_1 .\ (T_4 - T_5) + \frac{\Delta x}{\Delta y} .\ k_1 .\ (T_1 - T_5) = 0$$

$$\therefore\ 2(T_6 - T_5) + 2 * 0.666\ (T_{10} - T_5) + 2(T_4 + T_1 - 2T_5) = 0$$

$$\therefore\ 2T_1 + 2T_4 - 7.333\ T_5 + 2T_6 + 1.333\ T_{10} = 0$$

Node 10 :

$$\frac{\Delta y}{\Delta x} .\ k_2 .\ (T_{11} - T_{10}) + \frac{\Delta x}{\Delta y} .\ k_2 .\ (T_{16} - T_{10}) + \frac{\Delta y}{\Delta x} .\ k_2 .\ (T_9 - T_{10}) +$$

$$\frac{\Delta x}{\Delta y} .\ 2 .\ \frac{1}{1/k_1 + 1/k_2} .(T_5 - T_{10}) = 0$$

$$\therefore\ T_{11} - T_{10} + T_{16} - T_{10} + T_9 - T_{10} + 1.333\ (T_5 - T_{10}) = 0$$

$$\therefore\ 1.333\ T_5 + T_9 - 4.333\ T_{10} + T_{11} + T_{16} = 0$$

Node 16 :

$$\frac{\Delta y}{2\Delta x} \cdot k_1 \, (T_{17} - T_{16}) + \frac{\Delta y}{2\Delta x} \cdot k_1 . \, (T_{15} - T_{16}) + \frac{\Delta x}{\Delta y} \cdot k_1 . \, (T_{10} - T_{16}) +$$

$$\Delta x . h_2 . \, (T_a - T_{16}) = 0$$

where T_a = ambient temperature.

Since $h_2 = 7$; $k_1 = 1$; $T_a = 290$ this simplifies to :-

$$\tfrac{1}{2} (T_{17} - T_{16}) + \tfrac{1}{2} (T_{15} - T_{16}) + T_{10} - T_{16} + 1.4 \, (290 - T_{16}) = 0$$

$$\therefore \; 406 + T_{10} + 0.5 \, T_{15} - 3.4 \, T_{16} + 0.5 \, T_{17} = 0$$

The other nodes are similarly handled, leading to the equations tabulated in Table 7.8.

Node	Equation
1	$-12 \, T_1 + 2T_2 + 4T_5 = -3300$
2	$T_1 - 10T_2 + T_3 + 2T_6 = -3300$
3	$2T_2 - 10T_3 + 2T_7 = -3300$
4	$-6.666 \, T_4 + 4T_5 + 2.666 \, T_9 = 0$
5	$2T_1 + 2T_4 - 7.333 \, T_5 + 2 \, T_6 + 1.333T_{10} = 0$
6	$2T_2 + 2T_5 - 7.333 \, T_6 + 2 \, T_7 + 1.333T_{11} = 0$
7	$2T_3 + 4T_6 - 7.333 \, T_7 + 1.333T_{12} = 0$
8	$-4T_8 + 2T_9 + 2T_{14} = 0$
0	$1.333T_4 + T_8 - 4.333 \, T_9 + T_{10} + T_{15} = 0$
10	$1.333T_5 + T_9 - 4.333 \, T_{10} + T_{11} + T_{16} = 0$
11	$1.333T_6 + T_{10} - 4.333 \, T_{11} + T_{12} + T_{17} = 0$
12	$1.333T_7 + 2T_{11} - 4.333T_{12} + T_{18} = 0$
13	$-2.4T_{13} + T_{14} = 0$
14	$T_8 + 0.5T_{13} - 3.4T_{14} + 0.5 \, T_{15} = -406$
15	$T_9 + 0.5T_{14} - 3.4T_{15} + 0.5 \, T_{16} = -406$
16	$T_{10} + 0.5T_{15} - 3.4T_{16} + 0.5 \, T_{17} = -406$
17	$T_{11} + 0.5T_{16} - 3.4T_{17} + 0.5 \, T_{18} = -406$
18	$T_{12} + T_{17} - 3.4 \, T_{18} = -406$

Table 7.8.

These equations may be solved either by matrix inversion or by iteration. Using program DATA5 the following temperature values are obtained. These are shown in Figure 7.6.

```
RUN
INPUT J FOR JACOBI METHOD, GS FOR GAUSS-SEIDEL
? GS
NUMBER OF UNKNOWNS? 18
COEFFICIENTS OF EQUATION  1
? -12? 2? 0? 0? 4? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0
COEFFICIENTS OF EQUATION  2
? 1? -10? 1? 0? 0? 2? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0
COEFFICIENTS OF EQUATION  3
? 0? 2? -10? 0? 0? 0? 2? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0
COEFFICIENTS OF EQUATION  4
? 0? 0? 0? -6.666? 4? 0? 0? 0? 2.666? 0? 0? 0? 0? 0? 0? 0? 0? 0
COEFFICIENTS OF EQUATION  5
? 2? 0? 0? 2? -7.333? 2? 0? 0? 0? 1.333? 0? 0? 0? 0? 0? 0? 0? 0
COEFFICIENTS OF EQUATION  6
? 0? 2? 0? 0? 2? -7.333? 2? 0? 0? 0? 1.333? 0? 0? 0? 0? 0? 0? 0
COEFFICIENTS OF EQUATION  7
? 0? 0? 2? 0? 0? 4? -7.333? 0? 0? 0? 0? 1.333? 0? 0? 0? 0? 0? 0
COEFFICIENTS OF EQUATION  8
? 0? 0? 0? 0? 0? 0? 0? -4? 2? 0? 0? 0? 0? 2? 0? 0? 0? 0
COEFFICIENTS OF EQUATION  9
? 0? 0? 0? 1.333? 0? 0? 0? 1? -4.333? 1? 0? 0? 0? 0? 1? 0? 0? 0
COEFFICIENTS OF EQUATION  10
? 0? 0? 0? 0? 1.333? 0? 0? 0? 1? -4.333? 1? 0? 0? 0? 0? 1? 0? 0
COEFFICIENTS OF EQUATION  11
? 0? 0? 0? 0? 0? 1.333? 0? 0? 0? 1? -4.333? 1? 0? 0? 0? 0? 1? 0
COEFFICIENTS OF EQUATION  12
? 0? 0? 0? 0? 0? 0? 1.333? 0? 0? 0? 2? -4.333? 0? 0? 0? 0? 0? 1
COEFFICIENTS OF EQUATION  13
? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? -2.4? 1? 0? 0? 0? 0
COEFFICIENTS OF EQUATION  14
? 0? 0? 0? 0? 0? 0? 0? 1? 0? 0? 0? 0? .5? -3.4? .5? 0? 0? 0
COEFFICIENTS OF EQUATION  15
? 0? 0? 0? 0? 0? 0? 0? 0? 1? 0? 0? 0? 0? .5? -3.4? .5? 0? 0
COEFFICIENTS OF EQUATION  16
? 0? 0? 0? 0? 0? 0? 0? 0? 0? 1? 0? 0? 0? 0? .5? -3.4? .5? 0
COEFFICIENTS OF EQUATION  17
? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? 1? 0? 0? 0? 0? .5? -3.4? .5
COEFFICIENTS OF EQUATION  18
? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? 0? 1? 0? 0? 0? 0? 1? -3.4
INPUT VECTOR TERMS
? -3300? -3300? -3300? 0? 0? 0? 0? 0? 0? 0? 0? 0? -406? -406?
 -406? -406? -406?  -406
NUMBER OF ITERATIONS= 24
V( 1 )= 517.105
V( 2 )= 530.3788
V( 3 )= 532.3973
V( 4 )= 428.7415
V( 5 )= 461.3569
V( 6 )= 477.3167
V( 7 )= 481.7324
V( 8 )= 346.7833
V( 9 )= 380.326
V( 10 )= 403.1131
```

```
V( 11 )= 415.3587
V( 12 )= 419.0901
V( 13 )= 299.7733
V( 14 )= 313.539
V( 15 )= 326.8167
V( 16 )= 336.2425
V( 17 )= 341.4722
V( 18 )= 343.1065
Ok
```

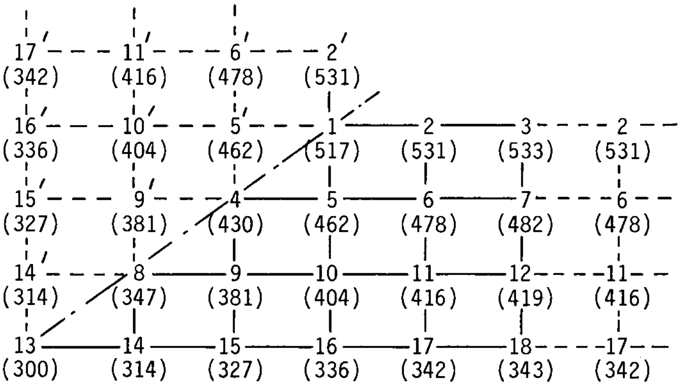


Figure 7.6. Values of temperature at the nodes.

NON-RECTILINEAR AND IRREGULAR BOUNDARIES

A. Modified Rectilinear Network

Suppose a portion of the network is intersected by a bounding surface of the solid, as shown for example in the two dimensional network of Figure 7.7. The heat transfer terms in the paths affected are rewritten thus:

Heat transfer between node 1 and the surface at 2' =

$$\frac{kA_{12}}{\Delta y_{12'}}(T_2' - T_1) = \frac{k(\Delta x + \Delta x_{15'})}{\Delta y * 2}(T_2' - T_1) \qquad (7.7)$$

Heat transfer between node 1 and the surface at 5' =

$$\frac{kA_{15}}{\Delta x_{15}}(T_5' - T_1) = \frac{k(\Delta y + \Delta y_{12'})}{\Delta x_{15'} * 2}(T_5' - T_1) \qquad (7.8)$$

Heat transfer between nodes 1 and 3, would =

$$\frac{kA_{13}}{\Delta x}(T_3 - T_1) = \frac{k(\Delta y)}{\Delta x}(T_3 - T_1) \qquad (7.9)$$

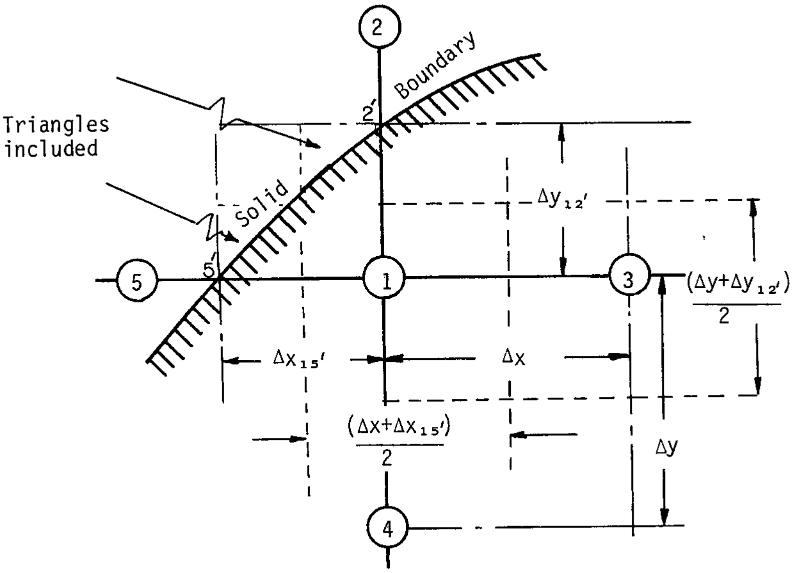


Figure 7.7. Rectilinear net, modified at an irregular boundary.

The method is of course, in error by the triangles which are included. Consequently, accuracy depends upon the fineness of the net used.

B. Triangular Network

The use of a triangular network in the vicinity of an irregular boundary has been proposed (4). The triangular network must be drawn so that no angle is greater than 90^0. Then supposing A, B and C to be three nodes at the corners of the triangle, the conductances between them are:

$$\text{conductance A to B} = \frac{k \cot \angle BCA}{2} \tag{7.10}$$

$$\text{conductance A to C} = \frac{k \cot \angle CBA}{2} \tag{7.11}$$

$$\text{conductance B to C} = \frac{k \cot \angle BAC}{2} \tag{7.12}$$

If there is more conducting material outside the triangle, then this must be included as additional conductance terms. It can easily be shown that use of the triangular network leads to the same result as the modified rectilinear net.

However, for purposes of visualisation, the triangular net may be easier to use, and can reduce the number of nodes required.

For a triangular network, the situation of Figure 7.7 would be redrawn as in Figure 7.8.

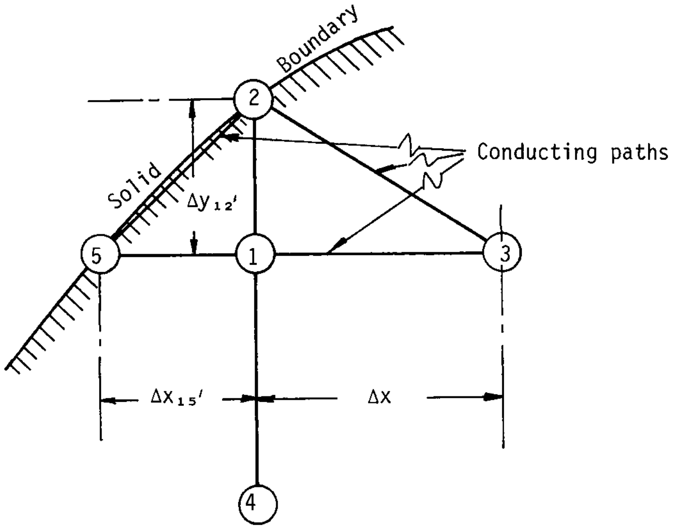


Figure 7.8. Triangular network alternative to that of Figure 1.10.

The heat transfer terms would then be as follows:

Heat transfer between nodes 1 and 2' =

$$\left[\frac{k\cot\angle 132}{2} + \frac{k\cot\angle 152}{2}\right](T_2' - T_1) =$$

$$\left[\frac{k}{2}\cdot\frac{\Delta x}{\Delta y_{12}} + \frac{k}{2}\cdot\frac{\Delta x_{15}}{\Delta y_{12'}}\right](T_2' - T_1) \tag{7.13}$$

which is the same result as before. Methods of minimising the number of nodes required, at a given mesh size, are available (5), (6).

C. Polar Co-ordinates

Where the boundary, or a portion of it has a cross section which is an arc of a circle, a polar co-ordinate system may be more convenient. Figure 7.9 shows a typical arrangement of nodes in an annular segment.

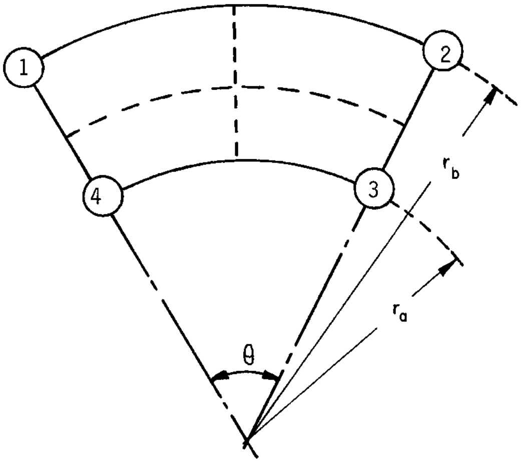


Figure 7.9. Nodes in an annular segment.

This arrangement has been made on the basis of equal increments of radius being assigned to each node. Other arrangements are of course possible, for example on the basis of equal increments of cross sectional area being assigned.

Taking the depth in the axial direction as unity, and using the arrangement of Figure 7.9, heat transfer by conduction between nodes can be written as:

Heat transfer between Nodes 1 and 2 =

$$\frac{Ak}{\Delta x}(T_2 - T_1) = \frac{r_b - r_a}{2} \cdot \frac{k\,(T_2 - T_1)}{\pi r_b\,(\theta/180)} \tag{7.14}$$

Heat transfer between Nodes 1 and 4 =

$$\pi\left[\frac{r_a + r_b)}{2}\right] \cdot \frac{\theta}{180} \cdot \frac{k\,(T_4 - T_1)}{(r_b - r_a)} \tag{7.15}$$

Heat transfer between Nodes 3 and 4 =

$$\frac{r_b - r_a}{2} \cdot \frac{k\,(T_4 - T_3)}{\pi r_a\,(\theta/180)} \tag{7.16}$$

Conduction through other paths outisde the annular segment should be included as additional terms.

EXAMPLE 7.3

Example of the solution of a problem involving non-rectilinear boundaries

Figure 7.10 shows the cross section proposed for a water cooled skid pipe for use in a heating furnace (7). The shape is to be constructed by welding strips made from rectangular bars, onto plain steel pipe. Assuming the thermal conductivity of the metal to be 48 W/mK, determine the temperature distribution over the cross section when the pipe is used in a furnace operating at 1300 K. The temperature of the water inside the pipe is 373 K.

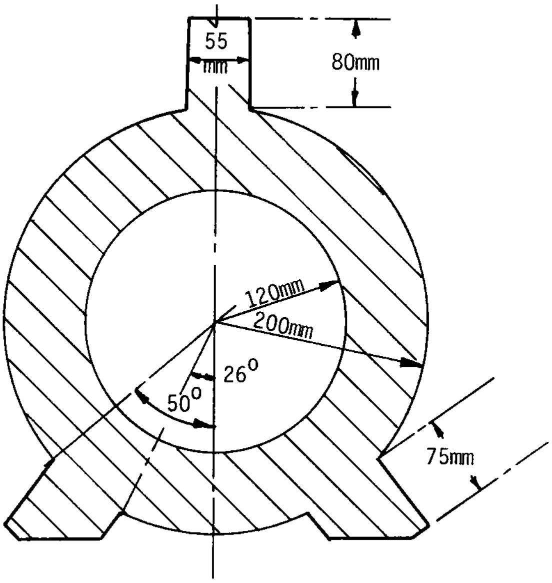


Figure 7.10. Cross-section of water-cooled skid pipe.

Solution

A suggested arrangement of nodes is shown in Figure 7.11. In order to simplify the calculations, a number of assumptions have been made:

i. The resistance to heat transfer offered by the water within the pipe is small, and will be ignored. Consequently nodes 11 to 16 inclusive are assumed to be at 373K.

ii. The furnace is assumed to behave as a black body enclosure, i.e. particular effects such as those due to an adjacent flame, are ignored.

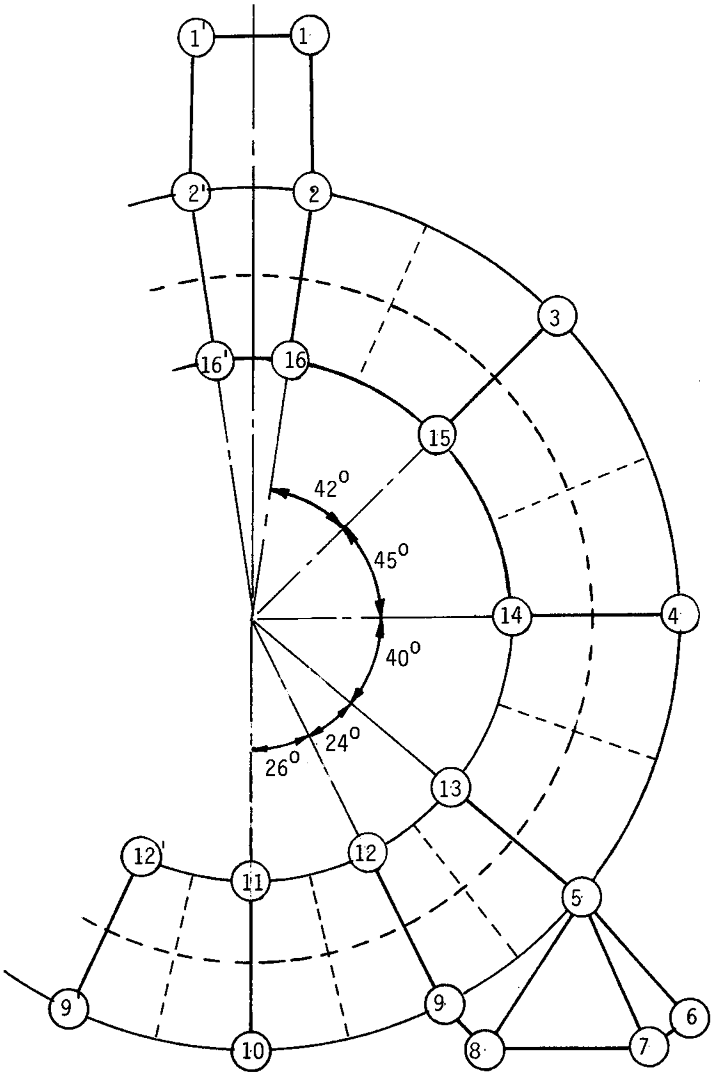


Figure 7.11. Suggested arrangement of nodes-water cooled skid pipe.

iii. No allowance is made for the effect of the adjacent floor etc. on the heat transfer by convection from the gas.

iv. No allowance is made for heat transfer by conduction between the furnace floor and nodes 7 and 8.

Node 1

$$A_{s1}\left[h_1(T_F - T_1) + \varepsilon\sigma(T_F - T_1^4)\right] + \frac{A_c k}{\Delta x}(T_2 - T_1) = 0$$

The coefficient of heat transfer by convection is assumed to be:

$$h_1 = 1.24\,(T_F - T_1)^{1/3}$$

This expression is based upon relationships for natural convection in air (1), (8).

Heat transfer by radiation is given by the expression

$\varepsilon\sigma\,(T_F^4 - T_1^4)$ where

ε = emissivity of the surface, (assumed to be in a heavily oxidised condition) = 0.95

σ = Stefan Bolzman constant = $5.668 * 10^{-8}\ \mathrm{Wm^{-2}\ K^{-4}}$

A_{s1} = area for heat loss by these mechanisms obtained from Fig. 7.10 = $\frac{0.0055 + 0.08}{2}\ \mathrm{m^2}$.

The equation for Node 1 can thus be written:

$$0.0837\,(T_F - T_1)^{4/3} + \frac{3.826}{10^9}\varepsilon(T_F^4 - T_1^4) +$$

$$0.3438k\,(T_2 - T_1) = 0$$

Node 2

$$\left(\frac{0.08}{2} + 0.2\pi * \frac{21}{180}\right)\left[1.24\,(T_F - T_2)^{4/3} + \frac{5.668\varepsilon}{10^8}(T_F^4 - T_2^4)\right] +$$

$$\frac{0.055k}{2*0.08}(T_1 - T_2) + 0.16\pi\,\frac{(21+3)}{180} * \frac{k(T_{16} - T_2)}{0.08} + \frac{0.04k\,(T_3 - T_2)}{0.2\pi\ *42/180} = 0$$

$$\therefore\ 0.1405\,(T_F - T_2)^{4/3} + \frac{6.422}{10^9}(T_F^4 - T_2^4) +$$

$$k\left[0.3438\,(T_1 - T_2) + 0.8378\,(T_{16} - T_2) + 0.2728\,(T_3 - T_2)\right] = 0$$

Node 5

$$\left(0.2\pi \ast \frac{20}{180} + \frac{0.075}{2}\right) \left[1.24\,(T_F - T_5)^{4/3} + \frac{5.668\varepsilon}{10^8}\,(T_F{}^4 - T_5{}^4)\right] +$$

$$\frac{0.04k\,(T_4 - T_5)}{0.2\pi \ast 40/180} + \frac{0.16\pi\,(20+12)k\,(T_{13} - T_5)}{180 \ast 0.08} +$$

$$\left(\frac{0.4}{0.2\pi \ast 24/180} + \frac{\cot\angle 589}{2}\right) \ast k\,(T_9 - T_5) +$$

$$\left(\frac{\cot\angle 598 + \cot\angle 578}{2}\right) \ast k\,(T_8 - T_5) + \left(\frac{\cot\angle 587 + \cot\angle 567}{2}\right) \ast$$

$$k\,(T_7 - T_5) + \frac{\cot\angle 576}{2} \ast k\,(T_6 - T_5) = 0$$

Equations for nodes 1 to 10 are written in this fashion and evaluated for angles and lengths measured from Figure 7.11.

It will be appreciated that direct solution of these simultaneous equations is impracticable since they are non-linear. Manual iteration by the method of relaxation is impossible for the same reason. In such a situation, computer iteration is the only way to obtain a solution.

Computer Solution of Temperature Distribution in a Water-Cooled Skid Pipe

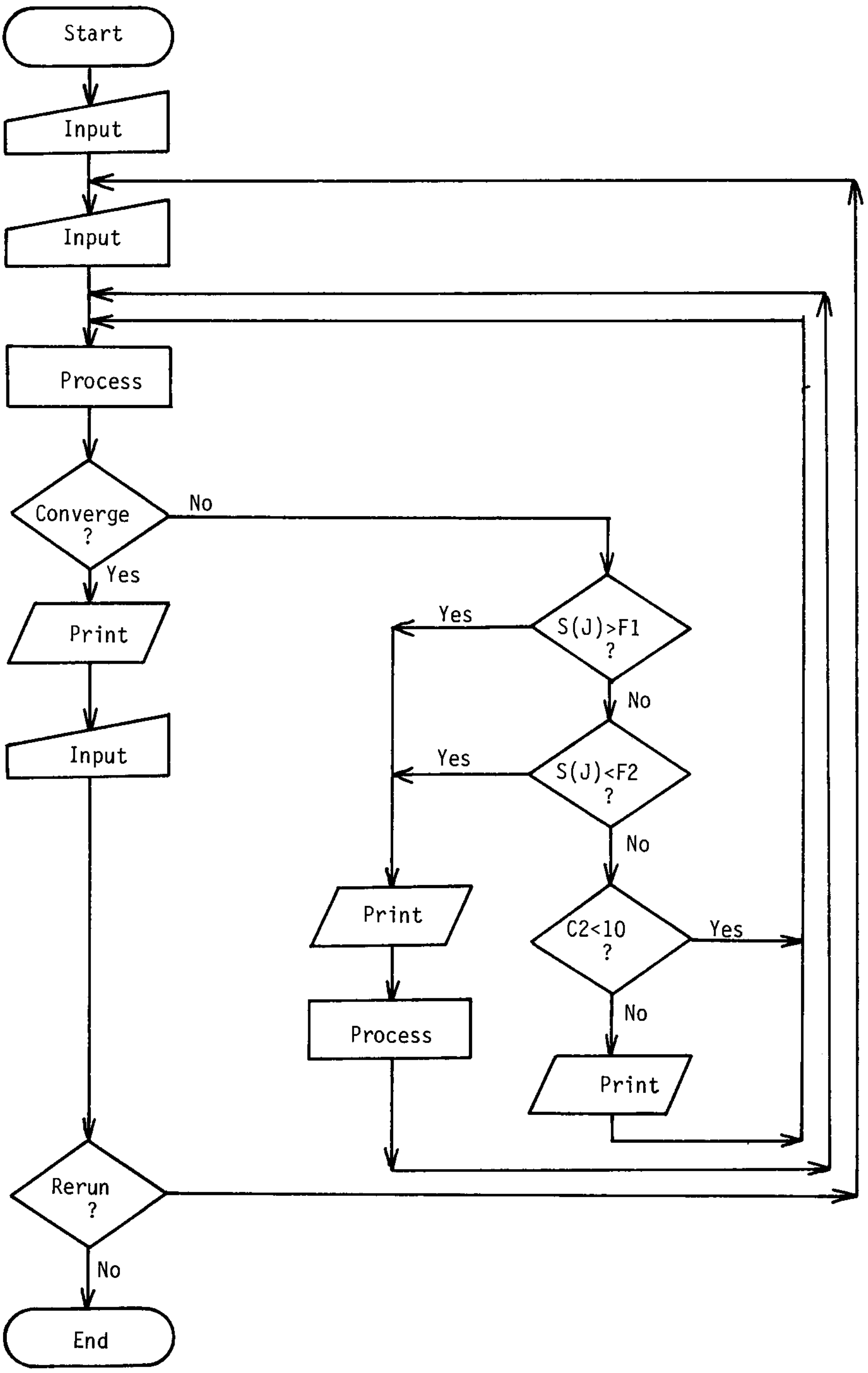
The equations for the nodes developed above, are used in the Basic program which follows. The equations are written in terms of finite remainders.

Starting from assumed temperature values, the program is executed by adjusting each temperature by an amount proportional to its remainder value. Iteration is continued until successive values of each temperature are substantially constant.

A difficulty with such a computer program is in deciding the size of adjustment to be made at each iteration. A large adjustment should reduce the number of iterations required, but may cause instability in the calculation. A small adjustment will necessitate a large number of iterations being made. A simple search routine has been included in order to select a suitable increment of the remainders.

The computer values of temperatures at the nodes are shown on Figure 7.12.

SSCOND1.BAS


Start
Input
Input
Process
Converge ?
No
Yes
Print
Input
S(J)>F1 ?
Yes
No
S(J)<F2 ?
Yes
No
Print
Process
C2<10 ?
Yes
No
Print
Rerun ?
No
End

```
10  REM ****************************************************
20  REM - THIS IS SSCOND1.BAS
30  REM - THIS PROGRAM EVALUATES TEMPERATURE DISTRIBUTION IN
40  REM - A WATER COOLED SKID PIPE
50  REM - PROGRAM NOMENCLATURE
60  REM - C1,C2    -   Number of iterations, and number
70  REM                of iterations between printouts,
80  REM                respectively
90  REM - E1       -   Emissivity
100 REM - F1,F2    -   Furnace and wall temperatures,
110 REM                respectively
120 REM - F4       -   Acceptable difference in nodal
130 REM                temperatures at successive iterations
140 REM - K        -   Thermal conductivity
150 REM - R(J)     -   Remainders calculated from the node
160 REM                equations
170 REM - R1       -   Factor used in incrementing nodal
180 REM                temperatures
190 REM - S(J)     -   New values of nodal temperatures
200 REM - T(J)     -   Values of nodal temperatures used in
210 REM                calculation of remainders
220 REM - PROGRAM DESCRIPTION
230 REM - LINES 1000 - 1070   Matrix T is dimensioned and
240 REM - physical properties and temperatures entered
250 REM - LINES 1080 - 1150   The count of iterations is set
260 REM - to zero (lines 1080 & 1090).   A mean between
270 REM - furnace and pipe wall temperatures is ascribed
280 REM - to the unknown temperatures at the nodes.
290 REM - Values of the wall temperature
300 REM - are ascribed to all nodes on the
310 REM - interior surface of the pipe (lines 1130 - 1150)
320 REM - LINES 1170 - 1460   Remainders are evaluated using
330 REM - the node equations, examples of which have been
340 REM - given earlier
350 REM - LINES 1470 - 1700   Values of temperatures at the
360 REM - nodes are adjusted, the adjustment being
370 REM - proportional to the size of the remainder
380 REM - (calculated at statements 1170 to 1460 above)
390 REM - The adjusted values are entered into Matrix S
400 REM - (lines 1470 - 1490)
410 REM - Next'a test is performed to check whether values
420 REM - at the latest iteration differ by more than a
430 REM - designated amount (F4) from those at the previous
440 REM - iteration (lines 1510 - 1530).   If they do not,
450 REM - the program terminates with a printout of the
460 REM - computed values (line 1540)
470 REM - If any temperature values differ by more than the
480 REM - designated amount, further iteration is necessary
490 REM - First however a check for stability is made (lines
500 REM - 1610 - 1640).   If all temperature values lie
510 REM - between those of furnace and pipe wall, a further
520 REM - iteration is carried out (lines 1650 & 1710 to
530 REM - 1780).   If instability is detected, a message is
540 REM - printed (line 1660), the value of increment R1 is
550 REM - halved (line 1680), and the calculation is
560 REM - restarted from the beginning (line 1700)
570 REM - LINES 1710 - 1780   Values are printed out at
580 REM - every tenth iteration
590 REM - LINES 1790 - 1870   Final values are printed out
600 REM - and the user is invited to rerun the program
610 REM ****************************************************
```

```
1000 DIM T(20)
1010 PRINT "PROPERTIES OF SKID PIPE"
1020 INPUT "THERMAL CONDUCTIVITY";K
1030 INPUT "EMISSIVITY";E1
1040 INPUT "FURNACE TEMPERATURE,ABSOLUTE";F1
1050 INPUT "WALL TEMPERATURE,ABSOLUTE";F2
1060 INPUT "DELTA R";R1
1070 INPUT "APPROACH, DEGREES";F4
1080 C1=0
1090 C2=0
1100 FOR J=1 TO 10
1110 T(J)=(F1+F2)/2
1120 NEXT J
1130 FOR J=11 TO 16
1140 T(J)=F2
1150 NEXT J
1160 REM - NODE EQUATIONS **********************************
1170 R(1)=.0837*(F1-T(1))^1.333+3.826E-09*E1*(F1^4-T(1)^4)
1180 R(1)=R(1)+.3438*K*(T(2)-T(1))
1190 R(2)=.1405*(F1-T(2))^1.333+6.422E-09*E1*(F1^4-T(2)^4)
1200 R(2)=R(2)+K*(.3438*(T(1)-T(2))+.8378*(T(16)-T(2)))
1210 R(2)=R(2)+K*.2723*(T(3)-T(2))
1220 R(3)=.1883*(F1-T(3))^1.333
1230 R(3)=R(3)+8.606001E-09*E1*(F1^4-T(3)^4)
1240 R(3)=R(3)+K*(.2728*(T(2)-T(3))+.2546*(T(4)-T(3)))
1250 R(3)=R(3)+K*1.518*(T(15)-T(3))
1260 R(4)=.184*(F1-T(4))^1.333+8.409E-09*E1*(F1^4-T(4)^4)
1270 R(4)=R(4)+K*(.2546*(T(3)-T(4))+.2865*(T(5)-T(4)))
1280 R(4)=R(4)+K*1.484*(T(14)-T(4))
1290 R(5)=.1331*(F1-T(5))^1.333+6.083E-09*E1*(F1^4-T(5)^4)
1300 R(5)=R(5)+K*(.2865*(T(4)-T(5))+.1529*(T(6)-T(5)))
1310 R(5)=R(5)+K*(.3373*(T(7)-T(5))+.182*(T(8)-T(5)))
1320 R(5)=R(5)+K*(.6497*(T(9)-T(5))+1.117*(T(13)-T(5)))
1330 R(6)=.06138*(F1-T(6))^1.333+2.806E-09*E1*(F1^4-T(6)^4)
1340 R(6)=R(6)+K*(.1625*(T(5)-T(6))+1.539*(T(7)-T(6)))
1350 R(7)=.06138*(F1-T(7))^1.333+2.806E-09*E1*(F1^4-T(7)^4)
1360 R(7)=R(7)+K*(.3373*(T(5)-T(7))+1.539*(T(6)-T(7)))
1370 R(7)=R(7)+K*.3768*(T(8)-T(7))
1380 R(8)=.0632*(F1-T(8))^1.333+2.891E-09*E1*(F1^4-T(8)^4)
1390 R(8)=R(8)+K*(.182*(T(5)-T(8))+.3768*(T(7)-T(8)))
1400 R(8)=R(8)+K*1.539*(T(9)-T(8))
1410 R(9)=.073*(F1-T(9))^1.333+3.337E-09*E1*(F1^4-T(9)^4)
1420 R(9)=R(9)+K*(.6497*(T(5)-T(9))+1.539*(T(8)-T(9)))
1430 R(9)=R(9)+K*(.4407*(T(10)-T(9))+.8727*(T(12)-T(9)))
1440 R(10)=.1125*(F1-T(10))^1.333
1450 R(10)=R(10)+5.144E-09*E1*(F1^4-T(10)^4)
1460 R(10)=R(10)+K*(.8815*(T(9)-T(10))+.9076*(T(11)-T(10)))
1470 FOR J=1 TO 10
1480 S(J)=T(J)+R(J)*R1
1490 NEXT J
1500 REM - CONVERGENCE TEST ********************************
1510 FOR J=1 TO 10
1520 IF ABS(S(J)-T(J))>F4 THEN 1550
1530 NEXT J
1540 GOTO 1790
1550 FOR J=1 TO 10
1560 T(J)=S(J)
1570 NEXT J
1580 C1=C1+1
1590 C2=C2+1
1600 REM - STABILITY TEST **********************************
```

```
1610 FOR J=1 TO 10
1620 IF S(J)>F1 THEN 1660
1630 IF S(J)<F2 THEN 1660
1640 NEXT J
1650 GOTO 1710
1660 PRINT "UNSTABLE BEHAVIOUR"
1670 PRINT
1680 R1=R1/2
1690 PRINT "NEW VALUE OF R1=";R1
1700 GOTO 1080
1710 IF C2<10 THEN 1170
1720 C2=0
1730 PRINT "ITERATION";C1
1740 FOR J=1 TO 10
1750 PRINT J,T(J)
1760 NEXT J
1770 PRINT
1780 GOTO 1170
1790 PRINT "NUMBER OF ITERATIONS=";C1
1800 PRINT "DELTA R VALUE USED =";R1
1810 PRINT "APPROACH USED =";F4;"DEGREES"
1820 FOR J=1 TO 16
1830 PRINT J,T(J)
1840 NEXT J
1850 INPUT "INPUT 1 TO RERUN";F4
1860 IF F4=1 THEN 1060
1870 END
```

```
LOAD"A:SSCOND1
Ok
RUN
PROPERTIES OF SKID PIPE
THERMAL CONDUCTIVITY? 48
EMISSIVITY? .95
FURNACE TEMPERATURE,ABSOLUTE? 1300
WALL TEMPERATURE,ABSOLUTE? 373
DELTA R? .05
APPROACH, DEGREES? .1
UNSTABLE BEHAVIOUR

NEW VALUE OF R1= .025
UNSTABLE BEHAVIOUR

NEW VALUE OF R1= .0125
UNSTABLE BEHAVIOUR

NEW VALUE OF R1= .00625
ITERATION 10
 1             1084.684
 2             816.3517
 3             699.1298
 4             693.2986
 5             756.1986
 6             1021.031
 7             994.5483
 8             855.6071
 9             747.7815
 10            711.468
```

```
ITERATION 20
 1             1113.917
 2             825.5598
 3             700.7467
 4             694.643
 5             763.6646
 6             1047.246
 7             1018.25
 8             866.6467
 9             754.7773
 10            714.8123

ITERATION 30
 1             1117.136
 2             826.6108
 3             700.9605
 4             694.9393
 5             765.3291
 6             1052.33
 7             1022.943
 8             869.3926
 9             756.6096
 10            715.8909

NUMBER OF ITERATIONS= 34
DELTA R VALUE USED = .00625
APPROACH USED = .1 DEGREES
 1             1117.366
 2             826.6862
 3             700.9777
 4               694.974
 5               765.5301
 6               1052.936
 7               1023.504
 8               869.7278
 9               756.8342
 10              716.025
 11              373
 12              373
 13              373
 14              373
 15              373
 16              373
INPUT 1 TO RERUN? 2
Ok
```

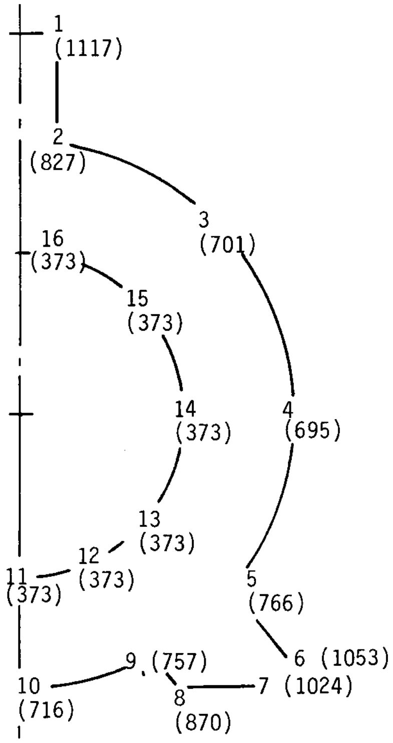


Figure 7.12. Temperatures at the nodes, using the computed values.

Three-Dimensional Networks

In the preceding examples, three-dimensional bodies have been represented mathematically by 1 - or 2 - dimensional systems. This very considerably reduces the complexity of the problem and the amount of work required to solve it. Writing finite difference equations is a laborious process.

Two devices are frequently employed either alone or together to reduce the complexity of the problem:

(a) the body is assumed to be of constant cross section, and end effects are ignored;

(b) the cross section of the body is represented uni-dimensionally by straight lines and circular ars.

The complexity of the shape which can be described by a 3-dimensional network is limited only by the powers of visualisation of the worker. However, where it is felt that a 3-dimensional system must be employed, extreme care should be exercised in setting up the equations. Drawings and sketches should be carefully inspected for any anomalies, which then help to pin-point errors.

Numerous references exist which can supply further information in more advanced areas (6), (8), (10), (11).

To conclude this chapter, here is an example of a problem in 3-dimensions:

EXAMPLE 7.4

Figure 7.13 shows a shape which is to be cast. It is required to determine the temperature distribution throughout the mould into which the casting material is poured. The following assumptions are made:

(a) The mould is contained within a cube of dimensions 11 units x 11 units x 11 units.

(b) The casting material and hence the faces of the casting are at a uniform temperature of 1200K.

(c) The outer faces of the moulding box are at a uniform temperature of 400K.

(d) Transient effects are ignored.

A program has been written which accepts the co-ordinates of the shape within a 3-dimensional matrix of a specified size. Figure 7.13 indicates the chosen co-ordinate system, coinciding with the edges of the moulding box.

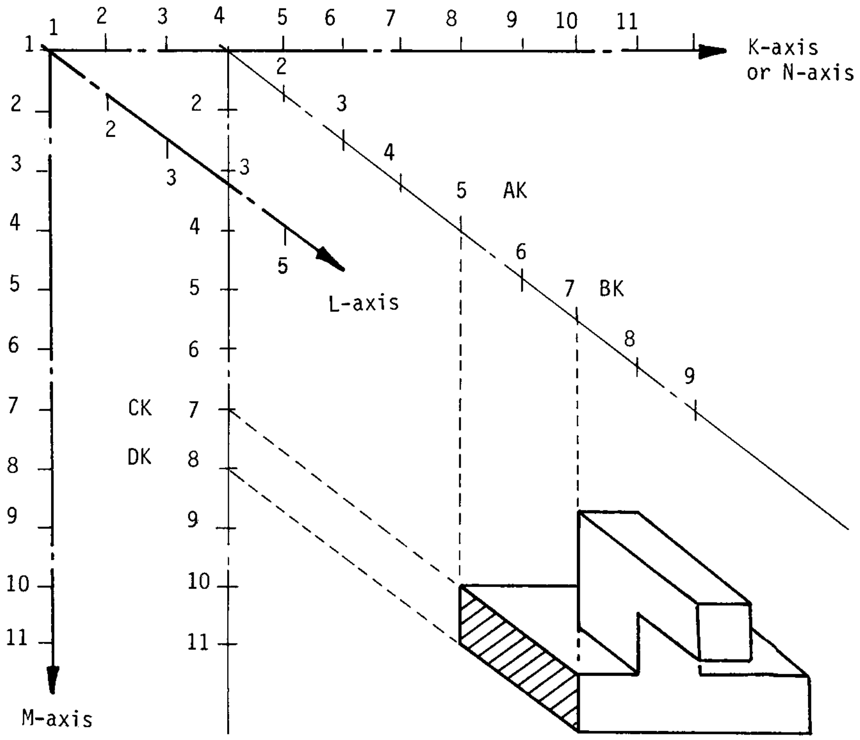


Figure 7.13. Co-ordinates for a 3-dimensional figure.

Heavy broken lines indicate the axes, which are coincident with the edges of the casting box.
Light broken lines indicate the coordinates of the L-M plane, at the fourth position along the N-axis.
Cross hatching indicates the intersection of this plane with the 3-dimensional figure.

This problem is one for which Basic is not well suited. Memory allocation for Basic is only 64K bytes, and this is inadequate for the storage of large arrays such as those required in this case. The only way out of this difficulty is by the use of files. Extra programming is required to handle the files which are stored on disc or cassette. Shuffling numbers back and forth between disc and compùter takes time, much of which is associated with the speed of response of the hardware. Consequently the program is excessively slow. There are however a number of ways in which computing time can be reduced:

1. Improve the hardware. This might be done by using a machine with a faster processor, and by using a hard disc for storage instead of a floppy disc.

2. Improve the program. In this instance this can be done by writing the program to use random files, rather than sequential files. Programs are given for both these cases.

3. Run the program on a ramdisc (virtual disc). This circumvents the storage limitations of the Basic compiler by taking the files out of the Basic area into the random access memory of the computer.

4. Improve or change the language. Basic normally operates through an interpreter, that is, each program line is compiled and checked for errors each time it is run through. A software option is available which produces a compiled version of a Basic program. Running such a compiled Basic program effects a considerable time saving. Another alternative is to use a language such as Pascal; since this does not have the memory limitations of Basic, the use of files is unnecessary, since large arrays can be generated. A consequent saving in processing time results.

Table 7.9 gives the times for only one iteration of the programs, using some of these various alternatives.

Computer	Drive	Program	Language	Time for one iteration of the program
PC	Floppy disc Ramdisc	Sequential files Sequential files	Basic Basic	15 minutes 7 min. 35 sec.
PC/XT	Hard disc Hard disc Ramdisc	Sequential files Random files Ramdom files	Basic Basic Basic	7 min. 56 sec. 5 min. 7 sec. 4 min. 24 sec.

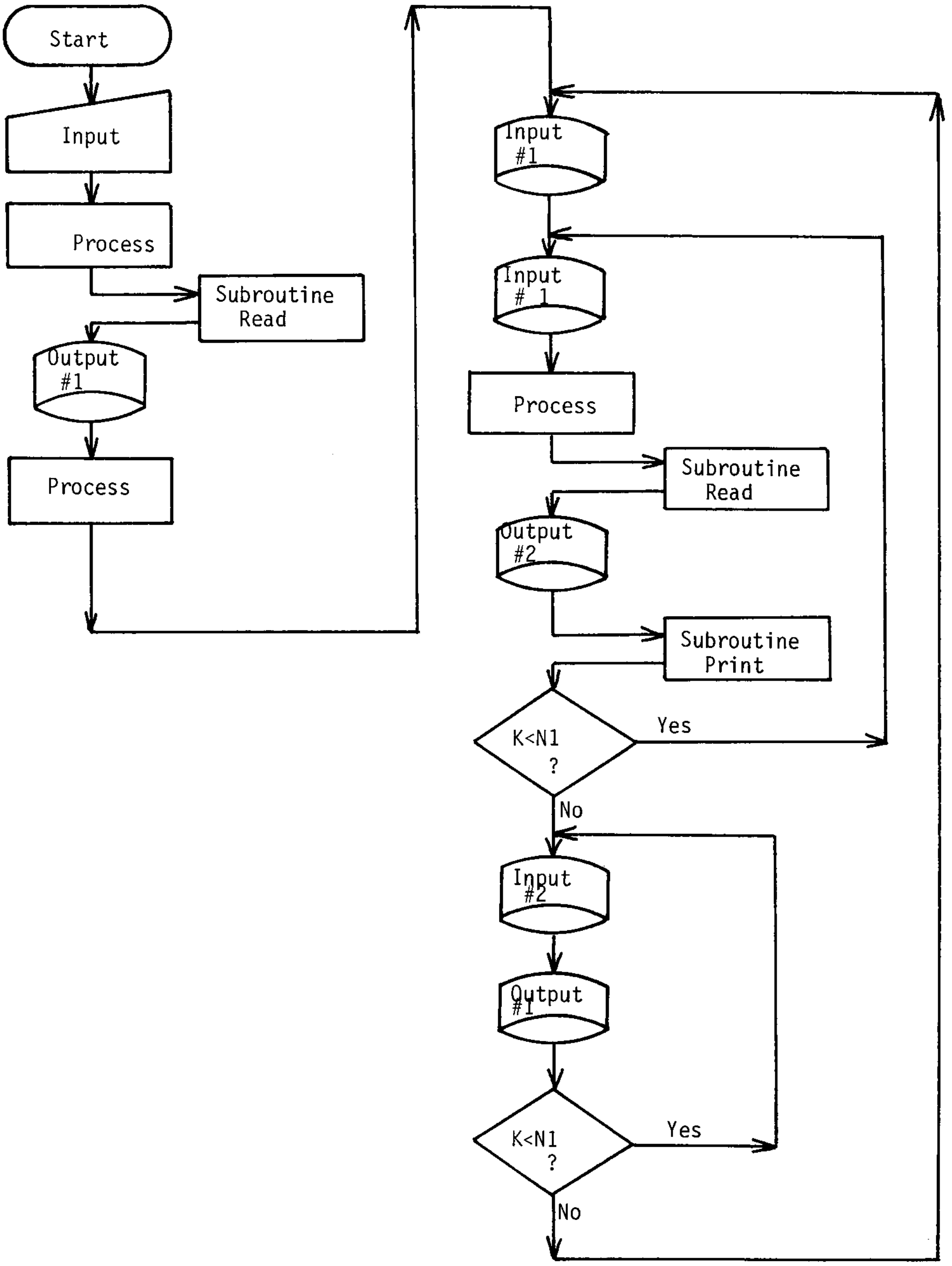
Table 7.9. Iteration times for solution of Example 7.4.

Since forty of fifty iterations are required in order to obtain a reasonable result, these programs take a long time to run.

There being no intersection of the shape with the L-M plane at values of N = 1,2,3, zeros are entered to the computer for these locations. At N = 4, values of L = 5,7 and M = 6,7 are entered. (This is also shown on Figure 7.13).

The computer programs, the input data values, and the temperatures calculated are given below. Table 7.10 gives the temperatures obtained at the 50th iteration and Figure 7.14 shows these values graphically.

SSCOND2.BAS AND
SSCOND3.BAS



```
10  REM ****************************************************
20  REM - PROGRAM SSCOND2.BAS   THIS PROGRAM
30  REM - CALCULATES TEMPERATURE DISTRIBUTION FOR
40  REM - SOLIDS IN THREE DIMENSIONS AT STEADY STATE
50  REM - USING SEQUENTIAL FILES
60  REM - PROGRAM NOMENCLATURE
70  REM - AK,CK   -   L,M coordinates of top left hand
80  REM               corner of a cross section of the
90  REM               shape (see Figure 7.13)
100 REM - BK,DK   -   L,M coordinates of bottom right hand
110 REM               corner of a cross section of the shape
120 REM               (see Figure 7.13)
130 REM - C1      -   Number of iterations
140 REM - N1      -   Number of subdivisions along an axis
150 REM - PK      -   Number of rectangular segments of the
160 REM               shape for which values of coordinates
170 REM               aAK,BK etc. have to be read
180 REM - T(L,M,N)    Values of temperatures at the nodes
190 REM - T1      -   Temperature of the surfaces of the box
200 REM - T2      -   Temperature of shape
210 REM - Z(L,M)  -   Values of temperatures at the nodes at
220 REM               one cross section (i.e. one particular
230 REM               value of ordinate N)
240 REM - PROGRAM DESCRIPTION
250 REM - LINES 2000 - 2020   Arrays are dimensioned and
260 REM - temperature values are entered from the keyboard
270 REM - LINES 2040 - 2110   These data statements contain
280 REM - data pertaining to Example 7.4
290 REM - LINES 2150 - 2320   File DATA1 is opened, ready
300 REM - for the transfer to it of values of temperatures
310 REM - at the nodes (line 2150).   Because of limitations
320 REM - in BASIC with regard to memory, this transfer must
330 REM - be done by instalments.   What we have to be able
340 REM - to do is to store all the elements of a 20,20,20
350 REM - matrix;since BASIC cannot handle this, we have to
360 REM - do this by taking the elements from successive
370 REM - 20,20 matrices.   First the value T1 input at line
380 REM - 2010, is ascribed to all the elements of Matrix Z
390 REM - (lines 2170 - 2210).   Next, some of these
400 REM - values are changed by entering values of T2 in
410 REM - accordance with the information contained in the
420 REM - data statements (line 2240 and subroutine Read)
430 REM - These values are then written to the file (lines
440 REM - 2250 - 2290).   This is done for each value of K
450 REM - (lines 2160 - 2300), so that all the initial
460 REM - values of temperatures at the nodes have been
470 REM - transferred to the file.   The file is then
480 REM - closed (line 2310);line 2320 allows the data
490 REM - statements to be read again.
500 REM - LINES 2330 - 2340   The iteration count is
510 REM - increased by 1, and the value is printed
520 REM - LINES 2420 - 3050   File Data1 is again opened,
530 REM - this time so that values may be read from the file
540 REM - In order to perform the finite difference
550 REM - calculation for each node, values for the nodes on
560 REM - either side must be available.   Referring to
570 REM - Figure 7.13, temperature values are required for
580 REM - all values of L & M, and for 3 values of N (or K)
590 REM - For this purpose the three dimensional matrix T
600 REM - is employed.   Temperatures at the surfaces of the
610 REM - box are assumed to be invariant at value T1.   The
```

```
620 REM - first two sets are read directly from the file
630 REM - (lines 2420 - 2490).  In order to get further
640 REM - values from the file in the correct order, it is
650 REM - necessary to close the file (line 2500), and then
660 REM - to reopen it (line 2520) for each successive step
670 REM - across the matrix (value of K).  Values are then
680 REM - read from the file, starting at the beginning of
690 REM - the record each time (lines 2540 - 2600).  Values
700 REM - preceding those required, are successively over-
710 REM - written at each step of the J loop. leaving the
720 REM - required values in Matrix T when J=K.  The file
730 REM - is then closed (line 2610).  Another file is
740 REM - employed to store the new temperature values
750 REM - generated at each iteration, and Data2 is opened
760 REM - for this purpose (line 2630).  Since boundary
770 REM - values = T1, these are first written to the file
780 REM - (lines 2640 - 2690) and these are printed out
790 REM - (line 2700 & subroutine Print).  Next the finite
800 REM - difference equations are evaluated (lines 2730 -
810 REM - 2780).  It was decided to apply these algorithms
820 REM - to all elements in the net, but this necessitates
830 REM - the reinstatement of the constant value T2 at the
840 REM - nodes corresponding to the location of the shape.
850 REM - This is done at line 2790 using subroutine Read
860 REM - (lines 3320 - 3420)  This completes one "cross
870 REM - section" of the solid, i.e. a unit step along the
880 REM - N (or K) axis.  The values are therefore written
890 REM - to the file (lines 2800 - 2840); the file is
900 REM - closed (line 2850) and the values are printed out
910 REM - (line 2860 & subroutine Print)
920 REM - The "cross section" is then moved one step along
930 REM - the N axis at lines 2890 - 2950.  The sequence
940 REM - described above is one traverse of the K loop;
950 REM - the program then moves from line 2960 back to
960 REM - line 2510, for the next value of K.  This
970 REM - procedure is continued until temperatures at all
980 REM - nodes have been computed.  Finally, temperatures
990 REM - at the end wall of the box (i.e. K=N1) are
1000 REM - reassigned the value T1; these values are
1010 REM - written to Data2 (lines 2980 - 3030); printed
1020 REM - (line 3040); and the file is closed (line 3050)
1030 REM - This completes one iteration.
1040 REM - LINES 3080 - 3290  Before another iteration can
1050 REM - be undertaken, the values calculated above for
1060 REM - nodal temperatures must be transferred from file
1070 REM - Data2 to file Data1.  This procedure also must
1080 REM - be carried out stepwise.  In order to obtain
1090 REM - values in their correct sequences, the J loop is
1100 REM - employed (lines 3100 - 3160) to remove values
1110 REM - from Data2 and successively overwrite them to
1120 REM - Matrix Z, until the correct ones appear at the
1130 REM - last step of the J loop.  These values are then
1140 REM - written to file Data1 (lines 3190 - 3270).  The
1150 REM - program then returns to line 2320 and another
1160 REM - iteration is commenced.
1170 REM - Due to the lengthy nature of this iterative
1180 REM - procedure, no means to test for convergence has
1190 REM - been included in the program.  Iteration will
1200 REM - continue until interrupted by the programmer.
1210 REM ***************************************************
2000 DIM Z(20,20),T(20,20,3)
```

```
2010 INPUT "BASE TEMPERATURE";T1
2020 INPUT "SHAPE TEMPERATURE";T2
2030 REM - DATA FOR EXAMPLE 7.4 ****************************
2040 N1=12
2050 DATA 0,0
2060 DATA 1,5,7,7,8
2070 DATA 2,6,8,5,6,5,7,7,8
2080 DATA 2,6,8,5,6,5,7,7,8
2090 DATA 1,5,7,7,8
2100 DATA 1,5,7,7,8
2110 DATA 0,0,0
2120 REM ***************************************************
2130 REM - TEMPERATURE VALUES T1 & T2 ARE ASCRIBED TO ALL
2140 REM - NODAL POINTS AND STORED IN FILE DATA1 ***********
2150 OPEN "DATA1" FOR OUTPUT AS #1
2160 FOR K=1 TO N1
2170 FOR L=1 TO N1
2180 FOR M=1 TO N1
2190 Z(L,M)=T1
2200 NEXT M
2210 NEXT L
2220 IF K=1 THEN 2250
2230 IF K=N1 THEN 2250
2240 GOSUB 3320
2250 FOR L=1 TO N1
2260 FOR M=1 TO N1
2270 PRINT #1,Z(L,M)
2280 NEXT M
2290 NEXT L
2300 NEXT K
2310 CLOSE #1
2320 RESTORE
2330 C1=C1+1
2340 PRINT
2350 BEEP
2360 BEEP
2370 PRINT "ITERATION";C1
2380 REM ***************************************************
2390 REM - INPUT TEMPERATURE VALUES FROM DATA1,EVALUATE
2400 REM - FINITE DIFFERENCE EQUATIONS & STORE THESE NEW
2410 REM - VALUES IN FILE DATA2 ****************************
2420 OPEN "DATA1" FOR INPUT AS #1
2430 FOR N=1 TO 2
2440 FOR L=1 TO N1
2450 FOR M=1 TO N1
2460 INPUT #1,T(L,M,N)
2470 NEXT M
2480 NEXT L
2490 NEXT N
2500 CLOSE #1
2510 FOR K=3 TO N1
2520 OPEN "DATA1" FOR INPUT AS #1
2530 IF EOF(1) THEN 2610
2540 FOR J=1 TO K
2550 FOR L=1 TO N1
2560 FOR M=1 TO N1
2570 INPUT #1,T(L,M,3)
2580 NEXT M
2590 NEXT L
2600 NEXT J
2610 CLOSE #1
```

```
2620 IF K>3 THEN 2720
2630 OPEN "DATA2" FOR OUTPUT AS #2
2640 FOR L=1 TO N1
2650 FOR M=1 TO N1
2660 Z(L,M)=T1
2670 PRINT #2,Z(L,M)
2680 NEXT M
2690 NEXT L
2700 GOSUB 3450
2710 GOTO 2730
2720 OPEN "DATA2" FOR APPEND AS #2
2730 FOR L=2 TO N1-1
2740 FOR M=2 TO N1-1
2750 Z(L,M)=T(L+1,M,2)+T(L-1,M,2)+T(L,M+1,2)+T(L,M-1,2)
2760 Z(L,M)=(Z(L,M)+T(L,M,3)+T(L,M,1))/6
2770 NEXT M
2780 NEXT L
2790 GOSUB 3320
2800 FOR L=1 TO N1
2810 FOR M=1 TO N1
2820 PRINT #2,Z(L,M)
2830 NEXT M
2840 NEXT L
2850 CLOSE #2
2860 GOSUB 3450
2870 REM ***************************************************
2880 REM - MOVE ONE STEP ALONG THE N AXIS ******************
2890 FOR N=1 TO 2
2900 FOR L=1 TO N1
2910 FOR M=1 TO N1
2920 T(L,M,N)=T(L,M,N+1)
2930 NEXT M
2940 NEXT L
2950 NEXT N
2960 NEXT K
2970 OPEN "DATA2" FOR APPEND AS #2
2980 FOR L=1 TO N1
2990 FOR M=1 TO N1
3000 Z(L,M)=T1
3010 PRINT #2,Z(L,M)
3020 NEXT M
3030 NEXT L
3040 GOSUB 3450
3050 CLOSE #2
3060 REM ***************************************************
3070 REM - TRANSFER VALUES FROM DATA2 TO DATA1 *************
3080 FOR K=1 TO N1
3090 OPEN "DATA2" FOR INPUT AS #2
3100 FOR J=1 TO K
3110 FOR L=1 TO N1
3120 FOR M=1 TO N1
3130 INPUT #2,Z(L,M)
3140 NEXT M
3150 NEXT L
3160 NEXT J
3170 CLOSE #2
3180 IF K>1 THEN 3210
3190 OPEN "DATA1" FOR OUTPUT AS #1
3200 GOTO 3220
3210 OPEN "DATA1" FOR APPEND AS #1
3220 FOR L=1 TO N1
```

```
3230 FOR M=1 TO N1
3240 PRINT #1,Z(L,M)
3250 NEXT M
3260 NEXT L
3270 CLOSE #1
3280 NEXT K
3290 GOTO 2320
3300 REM **************************************************
3310 REM - SUBROUTINE READ ********************************
3320 READ PK
3330 IF PK<1 THEN 3420
3340 FOR J=1 TO PK
3350 READ AK,BK,CK,DK
3360 FOR L=AK TO BK
3370 FOR M=CK TO DK
3380 Z(L,M)=T2
3390 NEXT M
3400 NEXT L
3410 NEXT J
3420 RETURN
3430 REM **************************************************
3440 REM - SUBROUTINE PRINT *******************************
3450 FOR M=1 TO N1
3460 FOR L=1 TO N1-1
3470 PRINT USING "#####";Z(L,M);
3480 NEXT L
3490 PRINT USING "#####";Z(N1,M)
3500 NEXT M
3510 PRINT
3520 RETURN
3530 END
```

```
10   REM ***************************************************
20   REM - PROGRAM SSCOND3.BAS
30   REM - THIS PROGRAM CALCULATES TEMPERATURE
40   REM - DISTRIBUTION FOR SOLIDS IN THREE DIMENSIONS
50   REM - AT STEADY STATE USING RANDOM FILES
60   REM - PROGRAM NOMENCLATURE
70   REM - This is as already given for SSCOND2, but the
80   REM - following additional variables are used:
90   REM - M$,N$  -  String variables used for random access
100  REM             to files Data2, Data1 respectively
110  REM - M6,N6  -  The number of the file record to be read
120  REM             (GET statements), or written (PUT
130  REM             statements).  Used for files Data2, &
140  REM             Data1 respectively
150  REM - PROGRAM DESCRIPTION
160  REM - The logic flowchart is identical with that for
170  REM - SSCOND2.  Changes made in the program are:
180  REM -    Use of different statements to address the
190  REM -    files (i.e. GET, PUT etc.)
200  REM -    Opening and closing of files during the running
210  REM -    of the program is eliminated
220  REM -    Since N6 and M6 keep a record of the positions
230  REM -    of the file pointers, the "J loops" used in
240  REM -    SSCOND2 are not required
250  REM - The latter two changes account for the reduced run
260  REM - time of this program when compared with SSCOND2
270  REM ***************************************************
2000 DIM Z(20,20),T(20,20,3)
2010 INPUT "BASE TEMPERATURE";T1
2020 INPUT "SHAPE TEMPERATURE";T2
2030 REM - DATA FOR EXAMPLE 7.4 ****************************
2040 N1=12
2050 DATA 0,0
2060 DATA 1,5,7,7,8
2070 DATA 2,6,8,5,6,5,7,7,8
2080 DATA 2,6,8,5,6,5,7,7,8
2090 DATA 1,5,7,7,8
2100 DATA 1,5,7,7,8
2110 DATA 0,0,0
2120 REM ***************************************************
2130 REM - TEMPERATURE VALUES T1 & T2 ARE ASCRIBED TO ALL
2140 REM - NODAL POINTS AND STORED IN FILE DATA1 ***********
2150 OPEN "DATA1" AS #1 LEN=12
2160 FIELD #1,N1 AS N$
2170 OPEN "DATA2" AS #2 LEN=12
2180 FIELD #2,N1 AS M$
2190 FOR K=1 TO N1
2200 FOR L=1 TO N1
2210 FOR M=1 TO N1
2220 Z(L,M)=T1
2230 NEXT M
2240 NEXT L
2250 IF K=1 THEN 2280
2260 IF K=N1 THEN 2280
2270 GOSUB 3310
2280 FOR L=1 TO N1
2290 FOR M=1 TO N1
2300 LSET N$=MKS$(Z(L,M))
2310 PUT #1
2320 NEXT M
2330 NEXT L
```

```
2340 NEXT K
2350 RESTORE
2360 C1=C1+1
2370 PRINT
2380 BEEP
2390 BEEP
2400 PRINT "ITERATION";C1
2410 REM ***************************************************
2420 REM - INPUT TEMPERATURE VALUES FROM DATA1,EVALUATE
2430 REM - FINITE DIFFERENCE EQUATIONS & STORE THESE NEW
2440 REM - VALUES IN FILE DATA2 ****************************
2450 M6=0
2460 N6=0
2470 FOR N=1 TO 2
2480 FOR L=1 TO N1
2490 FOR M=1 TO N1
2500 N6=N6+1
2510 GET #1,N6
2520 T(L,M,N)=CVS(N$)
2530 NEXT M
2540 NEXT L
2550 NEXT N
2560 FOR K=3 TO N1
2570 FOR L=1 TO N1
2580 FOR M=1 TO N1
2590 N6=N6+1
2600 GET #1,N6
2610 T(L,M,3)=CVS(N$)
2620 NEXT M
2630 NEXT L
2640 IF K>3 THEN 2740
2650 FOR L=1 TO N1
2660 FOR M=1 TO N1
2670 Z(L,M)=T1
2680 M6=M6+1
2690 LSET M$=MKS$(Z(L,M))
2700 PUT #2,M6
2710 NEXT M
2720 NEXT L
2730 GOSUB 3440
2740 FOR L=2 TO N1-1
2750 FOR M=2 TO N1-1
2760 Z(L,M)=T(L+1,M,2)+T(L-1,M,2)+T(L,M+1,2)+T(L,M-1,2)
2770 Z(L,M)=(Z(L,M)+T(L,M,3)+T(L,M,1))/6
2780 NEXT M
2790 NEXT L
2800 GOSUB 3310
2810 FOR L=1 TO N1
2820 FOR M=1 TO N1
2830 M6=M6+1
2840 LSET M$=MKS$(Z(L,M))
2850 PUT #2,M6
2860 NEXT M
2870 NEXT L
2880 GOSUB 3440
2890 REM ***************************************************
2900 REM - MOVE ONE STEP ALONG THE N AXIS ******************
2910 FOR N=1 TO 2
2920 FOR L=1 TO N1
2930 FOR M=1 TO N1
2940 T(L,M,N)=T(L,M,N+1)
```

```
2950 NEXT M
2960 NEXT L
2970 NEXT N
2980 NEXT K
2990 FOR L=1 TO N1
3000 FOR M=1 TO N1
3010 Z(L,M)=T1
3020 M6=M6+1
3030 LSET M$=MKS$(Z(L,M))
3040 PUT #2,M6
3050 NEXT M
3060 NEXT L
3070 GOSUB 3440
3080 REM ***************************************************
3090 REM - TRANSFER VALUES FROM DATA2 TO DATA1 *************
3100 M6=0
3110 N6=0
3120 FOR K=1 TO N1
3130 FOR L=1 TO N1
3140 FOR M=1 TO N1
3150 M6=M6+1
3160 GET #2,M6
3170 Z(L,M)=CVS(M$)
3180 NEXT M
3190 NEXT L
3200 FOR L=1 TO N1
3210 FOR M=1 TO N1
3220 N6=N6+1
3230 LSET N$=MKS$(Z(L,M))
3240 PUT #1,N6
3250 NEXT M
3260 NEXT L
3270 NEXT K
3280 GOTO 2350
3290 REM ***************************************************
3300 REM - SUBROUTINE READ *********************************
3310 READ PK
3320 IF PK<1 THEN 3410
3330 FOR J=1 TO PK
3340 READ AK,BK,CK,DK
3350 FOR L=AK TO BK
3360 FOR M=CK TO DK
3370 Z(L,M)=T2
3380 NEXT M
3390 NEXT L
3400 NEXT J
3410 RETURN
3420 REM ***************************************************
3430 REM SUBROUTINE PRINT **********************************
3440 FOR M=1 TO N1
3450 FOR L=1 TO N1-1
3460 PRINT USING "#####";Z(L,M);
3470 NEXT L
3480 PRINT USING "#####";Z(N1,M)
3490 NEXT M
3500 PRINT
3510 RETURN
3520 CLOSE #1
3530 CLOSE #2
3540 END
```

L - axis

M - axis

400	400	400	400	400	400	400	400	400	400	400	400
400	411	421	433	443	450	452	447	436	424	412	400
400	422	445	469	492	509	512	500	477	450	424	400
400	434	471	511	551	581	585	564	522	477	436	400
400	447	498	558	620	665	668	632	565	501	447	400
400	458	524	606	695	740	734	674	589	513	452	400
400	464	542	647	781	816	798	689	590	512	452	400
400	461	535	637	769	799	778	662	569	499	446	400
400	449	505	572	643	667	650	588	527	476	436	400
400	432	468	506	541	555	545	515	481	450	424	400
400	416	433	450	464	470	466	454	439	425	412	400
400	400	400	400	400	400	400	400	400	400	400	400

400	400	400	400	400	400	400	400	400	400	400	400
400	415	430	447	463	474	477	469	453	435	417	400
400	431	463	499	536	565	572	554	515	473	435	400
400	448	501	563	630	690	702	670	590	515	453	400
400	466	542	636	749	858	870	819	669	554	469	400
400	483	582	715	887	977	976	875	701	571	477	400
400	492	611	803	1200	1200	1200	878	691	566	475	400
400	488	602	790	1200	1200	1200	828	653	544	465	400
400	469	552	662	799	831	806	684	584	509	451	400
400	446	497	555	610	630	616	568	516	471	434	400
400	422	446	471	493	501	495	477	455	435	417	400
400	400	400	400	400	400	400	400	400	400	400	400

400	400	400	400	400	400	400	400	400	400	400	400
400	418	436	456	476	492	496	486	466	442	420	400
400	436	476	520	569	611	621	600	545	490	442	400
400	457	522	599	695	805	825	790	653	545	466	400
400	479	571	691	867	1200	1200	1200	788	599	486	400
400	498	616	775	988	1200	1200	1200	815	617	495	400
400	509	648	855	1200	1200	1200	988	762	603	491	400
400	504	636	840	1200	1200	1200	892	701	572	478	400
400	482	580	705	849	884	857	732	619	530	460	400
400	454	515	583	646	668	652	598	538	485	440	400
400	427	455	484	509	519	513	491	466	441	420	400
400	400	400	400	400	400	400	400	400	400	400	400

400	400	400	400	400	400	400	400	400	400	400	400
400	418	438	458	479	495	498	489	468	444	421	400
400	438	479	525	574	617	627	605	549	492	444	400
400	460	527	607	704	813	832	796	658	549	468	400
400	483	579	702	877	1200	1200	1200	795	604	489	400
400	503	626	788	999	1200	1200	1200	823	624	498	400
400	515	658	868	1200	1200	1200	999	773	610	494	400
400	509	647	852	1200	1200	1200	905	713	580	482	400
400	487	589	717	862	897	870	745	629	536	463	400
400	457	521	592	657	680	664	608	545	489	442	400
400	428	458	489	515	526	519	496	469	444	421	400
400	400	400	400	400	400	400	400	400	400	400	400

Table 7.10. Values of temperature across the L-M plane at values of N=3,4,5 & 6.

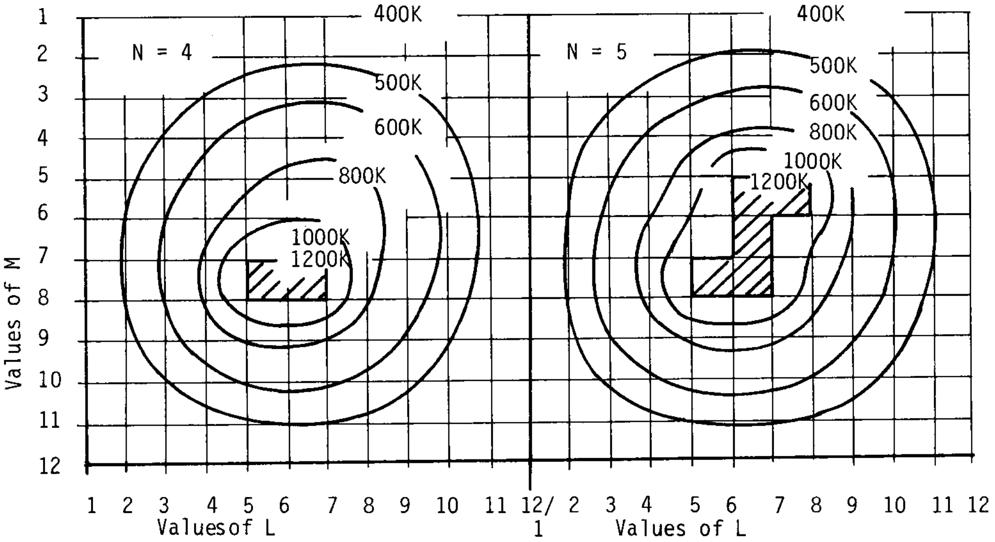


Figure 7.14. Isotherms across the L-M plate at values of N = 4 and 5.

PROBLEMS - CHAPTER 7

1. Use program SSCOND1 to determine the temperature distribution within the skid pipe of Example 7.3 assuming it to be constructed of alumiunium, of thermal conductivity 234W/mK.

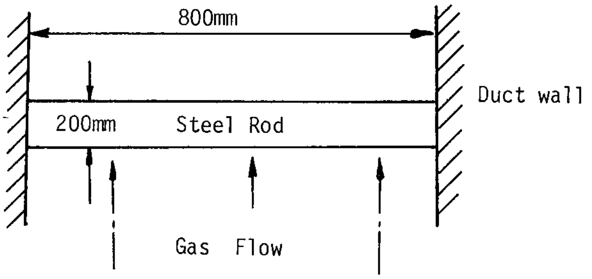
2. Use program SSCOND3 to determine steady state temperature distribution within a moulding box containing a vertical cylinder, height 6 units, diameter 8 units, located centrally within the box. Assume the face temperatures of box and cylinder to be 400K and 1200K respectively.

Note: Rewrite the Data statements at the beginning of the program so as to describe the new configuration. It will be represented as a number of contiguous rectangular slabs.

3. A solid steel rod 2 cm in diameter, spans a duct 800 mm wide, through which gas passes at 400K. The ends of the rod in contact with the duct walls, may be assumed to be at a constant temperature of 900K.

Thermal conductivity of steel = 50 W/mK

Heat transfer coefft, rod to gas = 25 W/mK



Determine the temperature distribution in the rod, and the heat loss from it, under steady state conditions, by the following methods:

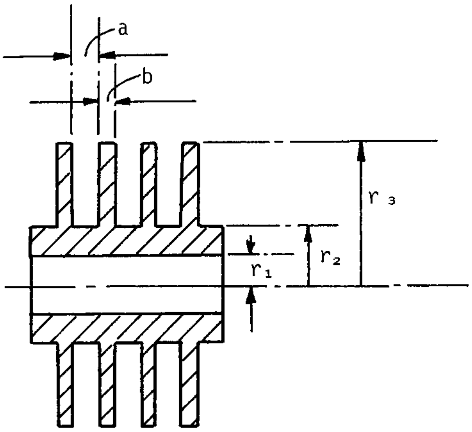
a. By formulation and solution of the differential equation describing the system.

b. By the finite difference method, employing manual solution of the equations.

c. By writing a computer program based on the finite difference method, to handle the general case, using n nodes

4. The sketch shows a portion of a pipe provided with external annular fins. Fluid flowing within the pipe is cooled by atmospheric air flowing transversely across the pipe.

Write a program to deal with this configuration, which will determine the temperature distribution within a fin, and the heat loss from the assembly.



REFERENCES

1. J.P. Holman, Heat Transfer, 3rd Edition, McGraw Hill Book Co., New York, U.S.A., 1972
2. F. Kreith, Principles of Heat Transfer, 3rd Edition, International Textbook Co., Scranton Pa., U.S.A., 1973
3. Heat Transfer 1970, Fourth International Heat Transfer Conference, Vol. I., Elsevier, Amsterdam, Netherlands, 1970
4. G.M. Dusinberre, Heat Transfer Calculations by Finite Differences, International Textbook Co., Scranton Pa., U.S.A., 1961
5. Heat Transfer 1974, Fifth International Heat Transfer Conference, Vol. I., Japan Society of Mechanical Engineering and Society of Chemical Engineering
6. J.F. Thompson, Editor, Numerical Grid Generation, Elsevier, Amsterdam, Netherlands, 1982
7. W. Trinks and M.H. Mawhinne, Industrial Furnaces, 5th Edition, McGraw Hill Book Co., New York, U.S.A., 1961
8. W.H. McAdams, Heat Transmission, McGraw Hill Book Co., New York, U.S.A., 3rd Edition, 1954
9. T.M. Shih, Numerical Heat Transfer, Hemisphere/Springer, Berlin, 1984
10. G.D. Smith, Numerical Solution of Partial Differential Equations: Finite Difference Methods, Clarendon Press, Oxford, U.K., 1978
11. L. Lapidus, G.F. Pinder, Numerical Solution of Partial Differential Equations in Science and Engineering, Wiley, New York, U.S.A., 1982

Chapter 8

UNSTEADY STATE CONDUCTION OF HEAT

Transient situations occur for instance in casting, ingot heating, and annealing; and in furnace start up and shut-down. Problems encountered include the prediction of heating or cooling times; and the effect which particular rates of heating or cooling will have on the sizes of the physical forces generated by differential thermal expansion.

This chapter is a continuation of the methods discussed in the previous chapter. Transient conduction in three dimensions is described by the equation previously derived:

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2} = \frac{\rho c}{k} \cdot \frac{\partial T}{\partial \theta} \qquad (7.5)$$

Analytical solution is possible if suitable simplification is made:

(a) Assuming Negligible Thermal Resistance

The assumption can be made that the thermal resistance within the circuit is negligible (the lumped parameter model). This assumption is considered to be satisfactory if the Biot Modulus (hL/k) is less than 0.1 (1). In this case, heat transfer between the body and its surroundings only need be considered, and is described by the equation:

$$-c\rho V dT = hA(T_\theta - T_a)d\theta \qquad (8.1)$$

where A = surface area of body;
V = volume of body.

This equation is easily solved by separation of the variables and integration, yielding:

$$\frac{T_\theta - T_a}{T_o - T_a} = e^{-(hA/c\rho V)\theta} \qquad (8.2)$$

where subscript 0 indicates time θ = zero
a indicates ambient

Where the temperature of the environment is also varying the solution is more difficult, but can be found in text books dealing with conduction heat transfer (1), (2).

(b) Assuming Finite Thermal Resistance, but simple Geometry (i.e. one dimensional system)

In this case, equation 7.5 is rewritten as:

$$\frac{\partial^2 T}{\partial x^2} = \frac{\rho c}{k} \cdot \frac{\partial T}{\partial \theta} \qquad (8.3)$$

The mathematics of this solution can also be found.

The results are available in graphical form, for a number of bodies which can be treated one dimensionally (infinite plane slab, infinite cylinder, sphere). The method can also be extended to deal with brick-shaped solids and finite cylinders. The range of practical applications of these graphical methods is severely limited, particularly when one bears in mind the present day availability of computers.

(c) Finite Difference Methods

As an alternative to analytical methods, Finite Difference methods can be employed. These methods can handle finite thermal resistances coupled with complex geometries. However this is done by assuming temperatures to remain constant during finite time increments. This of course leads to inaccuracy in the results obtained, accuracy depending on the size of the time increments selected and approaching the analytical solution (if one were obtainable) as the time increments become very small. These methods are nowadays preferable to analytical methods, due to the availability of computers.

Graphical Solution of the One Dimensional Case:

This is an application of the finite difference equation and is known as a Schmidt plot (1), (3).

For this case, equation 8.3 is written in finite difference notation as:

$$\frac{\Delta^2 T}{\Delta x^2} = \frac{\rho c}{k} \cdot \frac{\Delta T}{\Delta \theta} \qquad (8.4)$$

Rewriting this in terms of temperatures at the beginning and end of one time interval, we obtain this equation in a usable form:

$$\frac{\frac{k(T_{n-1}^{t} - T_{n}^{t})}{\Delta x} - \frac{k(T_{n}^{t} - T_{n+1}^{t})}{\Delta x}}{\Delta x} = \rho c \, \frac{(T_{n}^{t+1} - T_{n}^{t})}{\Delta \theta} \qquad (8.5)$$

Hence, $T_n^{t+1} = T_n^t + \frac{k}{\rho c} \cdot \frac{\Delta\theta}{\Delta x^2} (T_{n+1}^t - 2T_n^t + T_{n-1}^t)$ (8.6)

where T_n^t refers to the temperature of node n at time t

and T_n^{t+1} refers to the temperature of that node at time $t + \Delta\theta$

In solving a problem algebraically, the system would be split into nodes, and an equation such as this would be written for each.

In order to employ the method graphically, select values of Δx and $\Delta\theta$ so that

$$\frac{k}{\rho c} \cdot \frac{\Delta\theta}{\Delta x^2} = \frac{1}{2} \qquad (8.7)$$

Then equation 8.6 can be written as:

$$T_n^{t+1} = T_n^t + \frac{1}{2} (T_{n+1}^t - 2T_n^t + T_{n-1}^t) = (T_{n+1}^t + T_{n-1}^t)/2 \qquad (8.8)$$

In other words, the new value of temperature at node n is simply the arithmetic mean of the old values of temperature at the nodes on either side of it. If we represent nodes by equidistant lines, the length of line being proportional to temperature, then a straight line drawn from T_{n+1}^{t} to T_{n-1}^{t} will intersect the line for node n at value T_n^{t+1} .

EXAMPLE 8.1

A flat slab of metal 0.3m thick and at a temperature of 300K is suddently inserted into a furnace having a temperature of 1300K. The combined coefficient of heat transfer by convection and radiation from the furnace to the slab, (h_F) may be assumed constant, with the value of 124 W/m²K.

Determine the time required for the centre of the slab to reach 600K. Assume the following values of metal properties:

Thermal conductivity, k = 17.3 W/m ,K
Specific Heat, c = 232.5 J/kg,K
Density, ρ= 6000 kg/m³

SOLUTION A - GRAPHICAL METHOD

The slab is divided into an arbitrary number of layers. Assuming a layer thickness of 0.03m, only 5 layers need be considered, from arguments of symmetry. These are shown on Fig. 8.1. An additional layer is shown on this figure, representing the resistance at the surface.

The width of this layer on the diagram is evaluated so as to have an equivalent conductance to that of the surface coefficient.

i.e. h, the surface coefficient of heat transfer

$= \frac{k}{n\Delta x}$, the equivalent conductance of a solid layer of slab material;

$n\Delta x$ = the thickness of this simulated extra layer

$$= \frac{17.3}{124} = 0.14\text{m}$$

Lines representing the nodes, and the additional layer representing the surface conductance , are shown on Fig. 8.1. Initially temperatures at the various stations are indicated by the horizontal line at 300K. The construction is then commenced. Thus T_1 after one time interval $\Delta\theta$ is obtained by constructing a line joining the values of 1300 at T_0 with 300 at T_2^0. The intersection of this line with node 1, gives the value T_1^1 , and completes the first iteration. At the second iteration T_1^1 is joined to T_3^0 the intersection with node 2 giving the value T_2^2 . This completes the second iteration. T_0 is then joined to $T_2^{2'}$ and T_2^2 is joined to T_4^0 giving T_1^3 and T_3^3 . This completes the third iteration. At iteration 5, a line is drawn from T_4^4 to T_6^0 ,

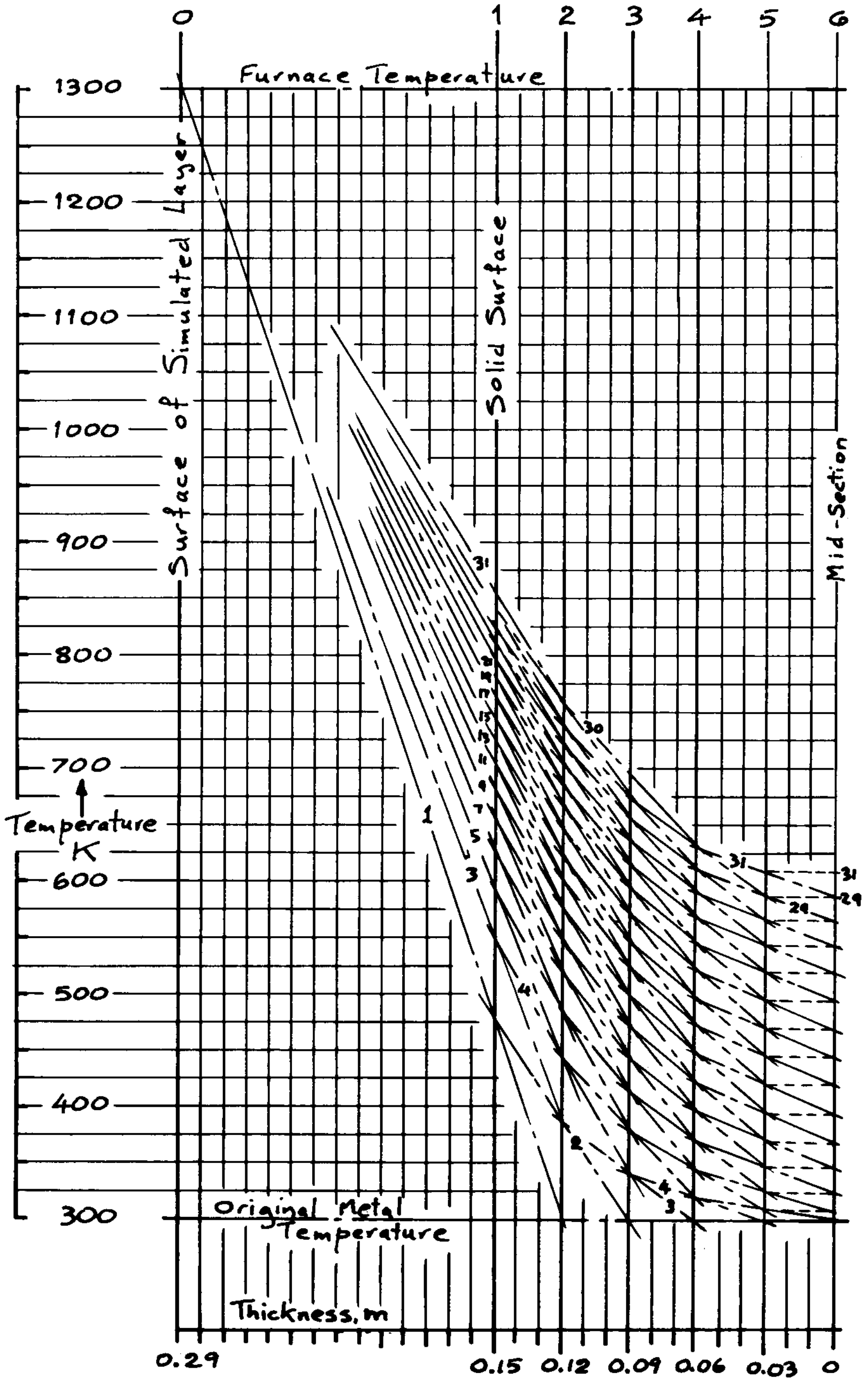

0
1
2
3
4
5
6
Furnace Temperature
1300
1200
1100
1000
900
800
700
600
500
400
300
Temperature
K
Surface of Simulated Layer
Solid Surface
Mid-Section
31
21
19
17
15
13
11
9
7
5
3
1
30
41
2
4
3
31
29
31
29
Original Metal Temperature
Thickness, m
0.29
0.15
0.12
0.09
0.06
0.03
0

Figure 8.1.

giving T_5^5 . Assuming there is no temperature gradient at the centre of the slab (by reasons of symmetry), then this value of T_5^5 will also be the value of T_6^5 . This is indicated on the diagram by a dotted line. The construction is continued until one of these dotted lines is drawn at or above the stipulated value of 600K. Note that in the preceding section, superscripts refer to the number of the iteration (not an exponent), and subscripts refer to location, as on Figure 8.1.

To count the iterations required, it should be noted that construction lines are drawn from nodes 1 to 0 at alternate iterations only. It will be seen that about 31 iterations are required to reach 600K. From equation 8.7,

$$\Delta\theta = \frac{\Delta x^2}{2} \cdot \frac{\rho c}{k} = \frac{0.03^2}{2} * \frac{10^5}{1.24} = 36.3 \text{ seconds}$$

Hence the time to reach 600K $\simeq$ 1125 seconds.

The temperatures at the various nodes will be approximately:

T_1 = 855K; T_2 = 757K; T_3 = 695K; T_4 = 631K; T_5 = 610K; T_6 = 610K.

SOLUTION B - COMPUTER ITERATION

An alternative computer solution to this problem, requires that equations be generated describing a heat balance around each node. The system of nodes used in this calculation is the same as that used in Solution A, and is shown in Fig. 8.2. The heat balances are written considering unit area of surface.

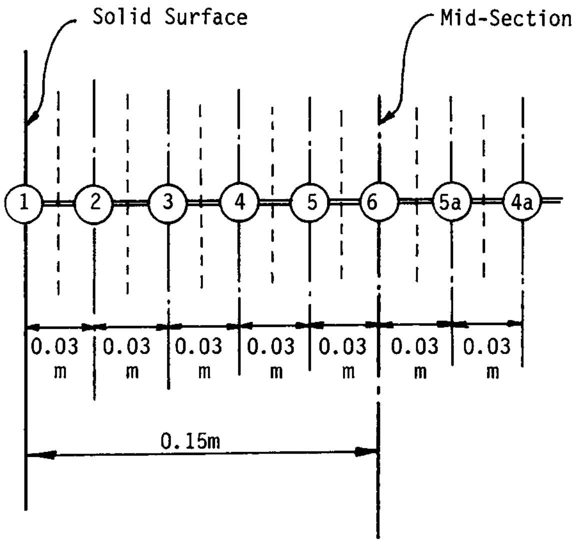


Figure 8.2. System of Nodes for Example 8.1.

Node 1

$$(T_F - T_1^t)\,h_F + \frac{k}{\Delta x}\,.\,(T_2^t - T_1^t) = \frac{\rho c\Delta x/2}{\Delta\theta}\,.\,(T_2^{t+1} - T_2^t) \qquad (8.9)$$

Rearranging this equation to solve for the new value of T_1, we obtain:

$$T_1^{t+1} = T_1^t + \frac{2\Delta\theta k}{\rho c(\Delta x)^2}\,.\,(T_2^t - T_1^t) + \frac{2\Delta\theta h_F}{\rho c\Delta x}\,.\,(T_F - T_1^t) \qquad (8.10)$$

Similarly we obtain the other equations:

$$T_2^{t+1} = T_2^t + \frac{\Delta\theta k}{\rho c(\Delta x)^2}\,.\,(T_1^t + T_3^t - 2T_2^t) \qquad (8.11)$$

$$T_3^{t+1} = T_3^t + \frac{\Delta\theta k}{\rho c(\Delta x)^2}\,.\,(T_2^t + T_4^t - 2T_3^t) \qquad (8.12)$$

etc.

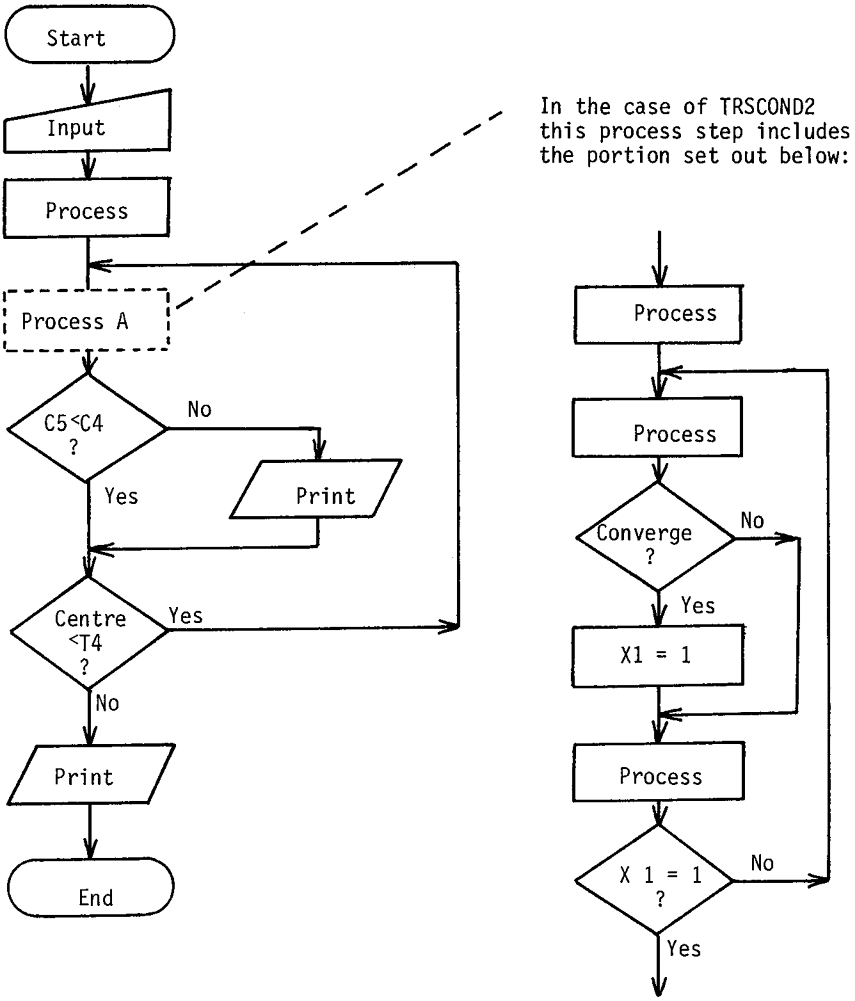
$$T_6^{t+1} = T_6^t + \frac{\Delta\theta k}{\rho c(\Delta x)^2}\,.\,(T_5^t + T_{5a}^t - 2T_6^t) \qquad (8.13)$$

$$= T_6^t + \frac{2\Delta\theta k}{\rho c(\Delta x)^2}\,.\,(T_5^t - T_6^t)$$

These equations are incorporated in the Basic program below. The results obtained are closely similar to those already given.

For a calculation which has to be performed once only the graphical technique may well be preferred, since the construction time is probably less than that required to generate the node equations and write the program. It is probably also sufficiently accurate. The computer solution of course has the advantage where a large number of similar calculations have to be performed.

TRSCOND1 and TRSCOND2


In the case of TRSCOND2 this process step includes the portion set out below:

```
10  REM ****************************************************
20  REM - PROGRAM TRSCOND1.BAS CALCULATES TEMPERATURE
30  REM - DISTRIBUTION WITHIN A PLANE SLAB SUDDENLY SUBJECT
40  REM - TO A STEP CHANGE IN ENVIRONMENTAL TEMPERATURE
50  REM - USING EXPLICIT OR FORWARD DIFFERENCE ITERATION
60  REM - PROGRAM NOMENCLATURE
70  REM - C1   -  Time interval, delta t
80  REM - C2   -  Time to latest iteration
90  REM - C3   -  Adjusts C2 at the last iteration
100 REM - C4   -  Time since last printout
110 REM - C5   -  Interval between printouts
120 REM - D1   -  Density
130 REM - D2   -  C1/(D1*S1*T3)
140 REM - H1   -  Heat transfer coefficient
150 REM - K1   -  Thermal conductivity
160 REM - K2   -  K1/T3
170 REM - N1   -  Number of nodes
180 REM - N2   -  Slab thickness
190 REM - S(J) -  Temperature values at the nodes at the
200 REM           latest iteration
210 REM - S1   -  Specific heat
220 REM - T(J) -  Temperature values at the nodes at the
230 REM           previous iteration
240 REM - T1   -  Slab temperature at start
250 REM - T2   -  Environment temperature
260 REM - T3   -  Distance between nodes
270 REM - T4   -  Temperature required at centre of slab
280 REM - PROGRAM DESCRIPTION
290 REM - LINES 1010 - 1130   Values of temperature and
300 REM - physical properties are entered
310 REM - LINES 1170 - 1210   Intermediate values required
320 REM - in the node equations are calculated
330 REM - LINES 1220 - 1270   Value T1 is ascribed to all
340 REM - nodes
350 REM - LINES 1290 - 1330   Temperatures at the nodes are
360 REM - calculated from previous values
370 REM - LINES 1340 - 1430   The count of iteration times
380 REM - is increased (lines 1340,1350); if interval since
390 REM - last printout equals C5, then the latest values
400 REM - are printed
410 REM - LINES 1440 - 1560   If temperature at centre of
420 REM - slab is less than the required value, then program
430 REM - execution performs another iteration (line 1440)
440 REM - Otherwise the total time is adjusted in proportion
450 REM - to the amount the central node temperature has
460 REM - exceeded T4 at the last iteration, and values are
470 REM - printed out
480 REM ****************************************************
1000 REM - PHYSICAL PROPERTIES *****************************
1010 INPUT "DELTA T, SECONDS";C1
1020 INPUT "INTERVAL BETWEEN PRINTOUTS, SECS";C4
1030 INPUT "NUMBER OF NODES-UP TO 10";N1
1040 PRINT "PHYSICAL PROPERTIES OF THE SLAB, SI UNITS:"
1050 INPUT "SLAB THICKNESS";N2
1060 INPUT "THERMAL CONDUCTIVITY";K1
1070 INPUT "DENSITY";D1
1080 INPUT "SPECIFIC HEAT";S1
1090 INPUT "SLAB TEMPERATURE";T1
1100 INPUT "ENVIRONMENT TEMPERATURE";T2
1110 INPUT "REQUIRED TEMPERATURE AT CENTRE OF SLAB";T4
1120 PRINT "COMBINED COEFFICIENT OF CONVECTIVE"
```

```
1130 INPUT "AND RADIATIVE TRANSFER";H1
1140 PRINT
1150 REM - CALCULATE INTERMEDIATE VALUES & ASCRIBE INITIAL
1160 REM - VALUES TO THE NODES *****************************
1170 T3=N2/(2*(N1-1))
1180 C2=0
1190 K2=K1/T3
1200 D2=D1*S1*T3/C1
1210 D2=1/D2
1220 FOR J=1 TO N1
1230 S(J)=T1
1240 NEXT J
1250 FOR J=1 TO N1
1260 T(J)=S(J)
1270 NEXT J
1280 REM - CALCULATE TEMPERATURES AT THE NODES**************
1290 S(1)=T(1)+2*D2*((T2-T(1))*H1+(T(2)-T(1))*K2)
1300 FOR J=2 TO N1-1
1310 S(J)=T(J)+D2*(T(J-1)+T(J+1)-2*T(J))*K2
1320 NEXT J
1330 S(N1)=T(N1)+2*D2*(T(N1-1)-T(N1))*K2
1340 C2=C2+C1
1350 C5=C5+C1
1360 IF C5<C4 THEN 1440
1370 REM - PRINT VALUES ************************************
1380 FOR J=1 TO N1
1390 PRINT USING "####";S(J);
1400 NEXT J
1410 PRINT "     ";C2;"SECS"
1420 C5=0
1430 PRINT
1440 IF S(N1)<T4 THEN 1250
1450 C3=(S(N1)-T4)/(S(N1)-T(N1))
1460 C2=C2-C1*C3
1470 FOR J=1 TO N1
1480 T(J)=S(J)-(S(J)-T(J))*C3
1490 NEXT J
1500 PRINT
1510 PRINT "TIME TO REACH";T4;"=";C2;"SECS"
1520 PRINT
1530 FOR J=1 TO N1
1540 PRINT USING "####.#";T(J);
1550 NEXT J
1560 END
```

EXAMPLE 8.2

Use the program TRSCOND1 to solve Example 8.1

```
LOAD"A:TRSCOND1
Ok
RUN
DELTA T, SECONDS? 30
INTERVAL BETWEEN PRINTOUTS, SECS? 180
NUMBER OF NODES-UP TO 10? 6
PHYSICAL PROPERTIES OF THE SLAB, SI UNITS:
SLAB THICKNESS? .3
THERMAL CONDUCTIVITY? 17.3
DENSITY? 6000
SPECIFIC HEAT? 232.5
SLAB TEMPERATURE? 300
ENVIRONMENT TEMPERATURE? 1300
REQUIRED TEMPERATURE AT CENTRE OF SLAB? 600
COMBINED COEFFICIENT OF CONVECTIVE
AND RADIATIVE TRANSFER? 124

 593 467 377 332 309 304      180 SECS

 677 556 462 398 361 349      360 SECS

 732 619 529 464 425 412      540 SECS

 776 672 588 526 489 476      720 SECS

 816 720 642 584 549 537      900 SECS

 853 764 691 638 605 594      1080 SECS


TIME TO REACH 600 = 1098.862 SECS

 856.5 768.0 696.2 643.3 610.9 600.0
Ok
```

METHODS OF ITERATION

The method of solution just described is referred to as an explicit or forward difference method. Values of temperature known at the beginning of a time interval, are used to solve the node equations and hence determine values of temperature at the end of each time interval. The disadvantage of the method is that if a sufficiently large value of the time interval $\Delta\theta$ is chosen, absurd answers are obtained. As a general guide, the numerical solution for interior points in a network will converge (and not diverge) (3) if

$M_1 \geq 2$ for one dimensional cases

$M_2 \geq 4$ for two dimensional cases

$M_3 \geq 6$ for three dimensional cases

where $M_1 = \frac{\Delta x^2}{\Delta\theta} \cdot \frac{\rho c}{k}$ etc.

Nevertheless, situations do sometimes arise in which the value of $\Delta\theta$ required by the above considerations is excessively small, resulting in excessively long computation time. This is exemplified in problems such as ablation, where decomposition of the conducting material is occurring, leading to smaller values of Δx at each iteration.

In such situations, an implicit method of calculation may be employed. The implicit method uses temperatures at the middle or end of the time interval. Since these temperatures are unknown, the node equations must be solved simultaneously at each iteration. This is a more laborious job than the explicit method. Only experience can decide whether this will involve less computing time than the normal method. However, it is certain that the implicit method will considerably reduce the fluctuations encountered otherwise (4), (5), (6).

As an example, the previous problem will be reworked using the implicit technique.

SOLUTION C - COMPUTER ITERATION (IMPLICIT TECHNIQUE)

The node equations are rewritten based upon the temperature driving forces obtaining at the end of a time interval instead of at the beginning:

Node 1

$$(T_F - T_1^{t+1})h + \frac{k}{\Delta x} . (T_2^{t+1} - T_1^{t+1}) = \frac{\rho c\Delta x}{2\Delta\theta} . (T_1^{t+1} - T_1^{t})$$

$$\therefore \quad T_1^{t+1} = \frac{T_F h + \frac{k}{\Delta x} . T_2^{t+1} + \frac{\rho c\Delta x}{2\Delta\theta} . T_1^{t}}{h + \frac{k}{\Delta x} + \frac{\rho c\Delta x}{2\Delta\theta}}$$

Node 2

$$\left[(T_1^{t+1} - T_2^{t+1}) + (T_3^{t+1} - T_2^{t+1})\right]\frac{k}{\Delta x} = \frac{\rho c\Delta x}{\Delta\theta} . (T_2^{t+1} - T_2^{t})$$

$$\therefore \quad T_2^{t+1} = \frac{(T_1^{t+1} + T_3^{t+1})\frac{k}{\Delta x} + \frac{\rho c\Delta x}{\Delta\theta} . T_2^{t}}{\frac{2k}{\Delta x} + \frac{\rho c\Delta x}{2\Delta\theta}}$$

Similarly we obtain the other equations. The central node (node 6 for this example) yields the equation:

$$T_6^{t+1} = \frac{T_5^{t+1}\cdot\frac{2k}{\Delta x} + \frac{\rho c\Delta x}{\Delta\theta}\cdot T_6^{t}}{\frac{2k}{\Delta x} + \frac{\rho c\Delta x}{\Delta\theta}}$$

We now have a set of simultaneous equations, and also a set of unknowns, since values T_1^{t+1} , T_2^{t+1} etc. are unknown. A solution can be obtained by matrix algebra or by iteration. In the Basic program given below, Seidel iteration is employed within the main iterative loop.

```
10   REM ***************************************************
20   REM - PROGRAM TRSCOND2.BAS CALCULATES TEMPERATURE
30   REM - DISTRIBUTION WITHIN A PLANE SLAB SUDDENLY SUBJECT
40   REM - TO A STEP CHANGE IN ENVIRONMENTAL TEMPERATURE
50   REM - USING IMPLICIT OR BACKWARD DIFFERENCE ITERATION
60   REM - PROGRAM NOMENCLATURE: As for TRSCOND1 except for
70   REM - the following
80   REM - D2    -  D1*S1*T3/C1
90   REM - R(J)     Temperature values at the previous
100  REM            iteration within the Seidel loop for
110  REM            solution of simultaneous equations
120  REM - S(J)     Temperature values at the nodes during
130  REM            the Seidel iteration
140  REM - X1    -  Value 1 when simultaneous equations have
150  REM            converged, otherwise zero
160  REM - S(J)     Temperature values at the nodes during
170  REM            the Seidel iteration
180  REM - PROGRAM DESCRIPTION: As for TRSCOND1 except for
190  REM - the portion dealing with solution of simultaneous
200  REM - equations as follows -
210  REM - LINES 1300 - 1460   Node temperatures are
220  REM - calculated (lines 1300 - 1340), using the Seidel
230  REM - iteration method.   These values are stored as
240  REM - values of S(J).   They are compared with
250  REM - values of R(J) (lines 1370,1380), and the values
260  REM - are then transferred from matrix S to matrix R
270  REM - Unless agreement is within specified limits,
280  REM - another Seidel iteration is performed (lines 1440
290  REM - to 1460)
300  REM ****************************************************
1000 REM - PHYSICAL PROPERTIES ****************************
1010 INPUT "DELTA T, SECONDS";C1
1020 INPUT "INTERVAL BETWEEN PRINTOUTS, SECS";C4
1030 INPUT "NUMBER OF NODES-UP TO 10";N1
1040 PRINT "PHYSICAL PROPERTIES OF THE SLAB, SI UNITS:"
1050 INPUT "SLAB THICKNESS";N2
1060 INPUT "THERMAL CONDUCTIVITY";K1
1070 INPUT "DENSITY";D1
1080 INPUT "SPECIFIC HEAT";S1
1090 INPUT "SLAB TEMPERATURE";T1
1100 INPUT "ENVIRONMENT TEMPERATURE";T2
1110 INPUT "REQUIRED TEMPERATURE AT CENTRE OF SLAB";T4
1120 PRINT "COMBINED COEFFICIENT OF CONVECTIVE"
1130 INPUT "AND RADIATIVE TRANSFER";H1
1140 PRINT
```

```
1150 REM - CALCULATE INTERMEDIATE VALUES & ASCRIBE INITIAL
1160 REM - VALUES TO THE NODES *****************************
1170 T3=N2/(2*(N1-1))
1180 C2=0
1190 K2=K1/T3
1200 D2=D1*S1*T3/C1
1210 FOR J=1 TO N1
1220 S(J)=T1
1230 R(J)=S(J)
1240 NEXT J
1250 FOR J=1 TO N1
1260 T(J)=S(J)
1270 NEXT J
1280 REM - CALCULATE TEMPERATURES AT THE NODES SOLVING FOR
1290 REM - THE UNKNOWNS BY SEIDEL ITERATION ****************
1300 S(1)=(T(1)*D2/2+S(2)*K2+T2*H1)/(D2/2+K2+H1)
1310 FOR J=2 TO N1-1
1320 S(J)=(T(J)*D2+(S(J-1)+S(J+1))*K2)/(2*K2+D2)
1330 NEXT J
1340 S(N1)=(T(N1)*D2+S(N1-1)*2*K2)/(2*K2+D2)
1350 REM - CHECK CONVERGENCE OF SEIDEL ITERATIONS **********
1360 FOR J=1 TO N1
1370 IF R(J)/S(J)>1.0001 THEN 1410
1380 IF S(J)/R(J)>1.0001 THEN 1410
1390 NEXT J
1400 X1=1
1410 FOR J=1 TO N1
1420 R(J)=S(J)
1430 NEXT J
1440 IF X1=1 THEN 1460
1450 GOTO 1300
1460 X1=0
1470 C2=C2+C1
1480 C5=C5+C1
1490 IF C5<C4 THEN 1570
1500 REM - PRINT VALUES ************************************
1510 FOR J=1 TO N1
1520 PRINT USING "####";S(J);
1530 NEXT J
1540 PRINT "     ";C2;"SECS"
1550 C5=0
1560 PRINT
1570 IF S(N1)<T4 THEN 1250
1580 C3=(S(N1)-T4)/(S(N1)-T(N1))
1590 C2=C2-C1*C3
1600 FOR J=1 TO N1
1610 T(J)=S(J)-(S(J)-T(J))*C3
1620 NEXT J
1630 PRINT
1640 PRINT "TIME TO REACH";T4;"=";C2;"SECS"
1650 PRINT
1660 FOR J=1 TO N1
1670 PRINT USING "####.#";T(J);
1680 NEXT J
1690 END
```

If sufficiently small values of $\Delta\theta$ are employed, both the explicit and implicit methods give closely similar solutions. With regard to Example 8.2 , according to the stability criteria already given, the maximum value of $\Delta\theta$ to be used with the explicit method

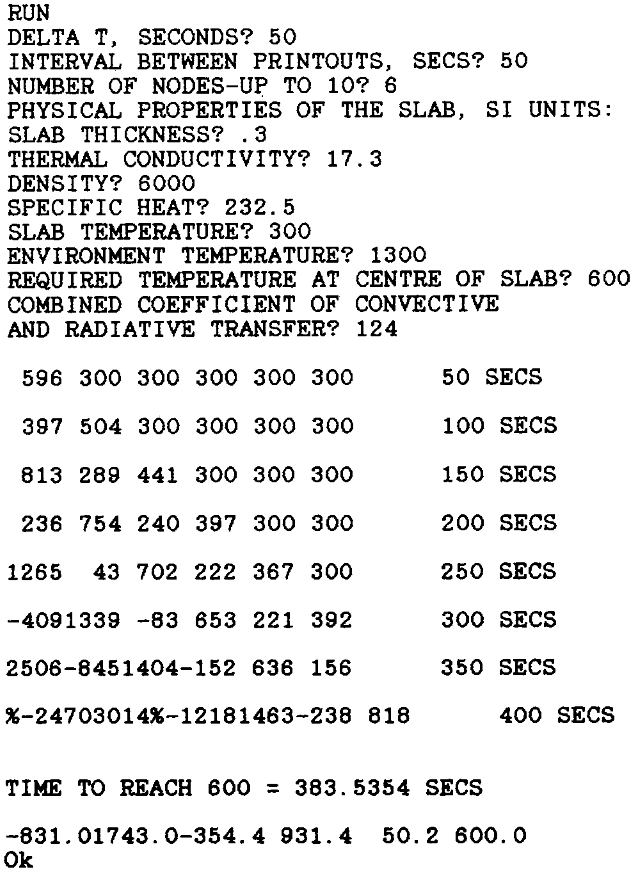
$$= \frac{(\Delta x)^2}{M} \cdot \frac{\rho c}{k} = \frac{0.03^2}{2} \cdot \frac{10^5}{1.24}$$

= 36.3 seconds

A value of $\Delta\theta$ = 30 seconds was used in the explicit solution already given. What happens if a value greater than 36.3 is employed? The following printouts show results for both methods, using values of $\Delta\theta$ greater than the critical value.

EXAMPLE 8.3

Use the program TRSCOND1 to solve Example 8.1, but use a value of $\Delta\theta$ greater than the critical value of 36.3 seconds.

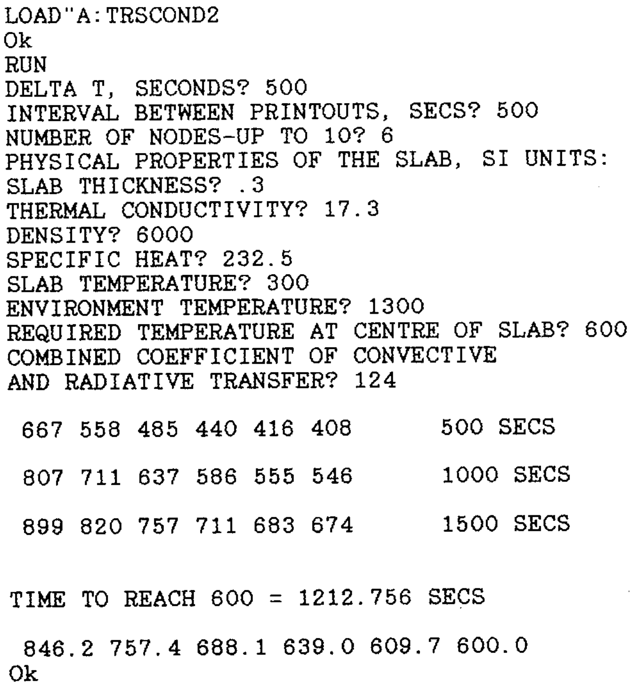


```
RUN
DELTA T, SECONDS? 50
INTERVAL BETWEEN PRINTOUTS, SECS? 50
NUMBER OF NODES-UP TO 10? 6
PHYSICAL PROPERTIES OF THE SLAB, SI UNITS:
SLAB THICKNESS? .3
THERMAL CONDUCTIVITY? 17.3
DENSITY? 6000
SPECIFIC HEAT? 232.5
SLAB TEMPERATURE? 300
ENVIRONMENT TEMPERATURE? 1300
REQUIRED TEMPERATURE AT CENTRE OF SLAB? 600
COMBINED COEFFICIENT OF CONVECTIVE
AND RADIATIVE TRANSFER? 124

 596 300 300 300 300 300      50 SECS

 397 504 300 300 300 300      100 SECS

 813 289 441 300 300 300      150 SECS

 236 754 240 397 300 300      200 SECS

1265  43 702 222 367 300      250 SECS

-4091339 -83 653 221 392     300 SECS

2506-8451404-152 636 156     350 SECS

%-24703014%-12181463-238 818      400 SECS


TIME TO REACH 600 = 383.5354 SECS

-831.01743.0-354.4 931.4  50.2 600.0
Ok
```

It will be seen that a value of $\Delta\Theta$ = 50 seconds, used with the explicit method, shows instability which has increased sufficiently to yield negative values for node temperatures.

EXAMPLE 8.4

Repeat the above exercise using the program TRSCOND2.



```
LOAD"A:TRSCOND2
Ok
RUN
DELTA T, SECONDS? 500
INTERVAL BETWEEN PRINTOUTS, SECS? 500
NUMBER OF NODES-UP TO 10? 6
PHYSICAL PROPERTIES OF THE SLAB, SI UNITS:
SLAB THICKNESS? .3
THERMAL CONDUCTIVITY? 17.3
DENSITY? 6000
SPECIFIC HEAT? 232.5
SLAB TEMPERATURE? 300
ENVIRONMENT TEMPERATURE? 1300
REQUIRED TEMPERATURE AT CENTRE OF SLAB? 600
COMBINED COEFFICIENT OF CONVECTIVE
AND RADIATIVE TRANSFER? 124

 667 558 485 440 416 408      500 SECS

 807 711 637 586 555 546      1000 SECS

 899 820 757 711 683 674      1500 SECS


TIME TO REACH 600 = 1212.756 SECS

 846.2 757.4 688.1 639.0 609.7 600.0
Ok
```

This time the very large value of 500 seconds has been used for $\Delta\Theta$. Nevertheless, the implicit method gave acceptable answers and required only three iterations.

To conclude this chapter, here is a problem involving transient heat transfer within a three-dimensional solid form. The physical situation is more complicated, but the program itself is quite simple.

EXAMPLE 8.5.

A furnace is employed for heating long steel billets which cannot be fully contained by it. The arrangement is shown in Figure 8.3. Write a program to calculate the heating time and heat losses for such an arrangement.

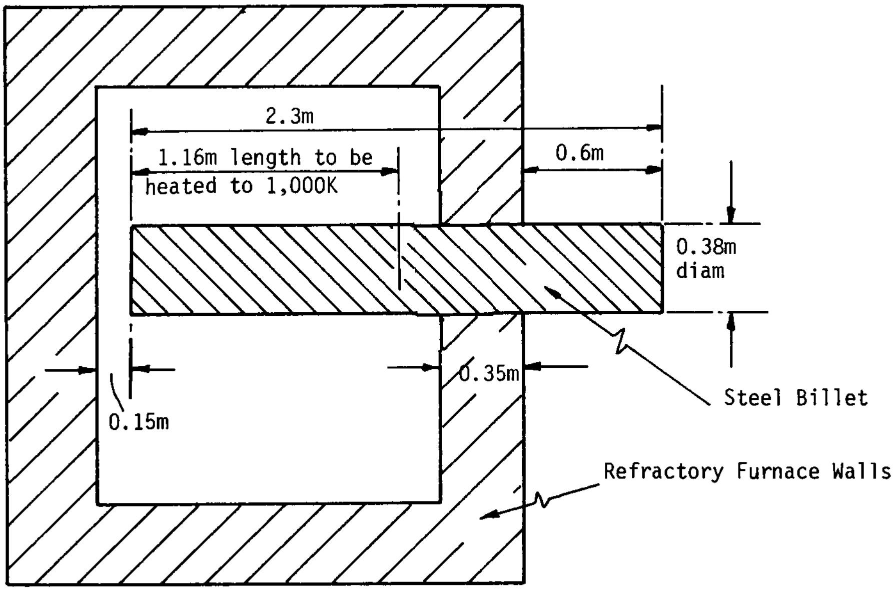


Figure 8.3. Problem in Billet Heating.

A steel billet 0.38m diameter by 2.3m long has to be heated so that 1.16m at one end is above 900 K. Calculate heating time and heat requirement assuming furnace temperature to be 1200K, and ambient temperature 300 K. Values of physical properties required are given in Table 8.1 below.

Emissivity of mild steel = 0.94
Stefan radiation constant = 5.668 x 10^{-8} W/m^2K^4

	Density ρ	Specific Heat c	Thermal Conductivity k	Viscosity μ	Volume coefficient of Expansion β
	kg/m^3	J/kgK	W/mK	Ns/m^2	K^{-1}
Mild Steel (subscript m)	7850	460	45	-	-
Refractory (subscript r)	2300	837	1.23	-	-
Air at 925K	0.382	1125	0.0639	$4 * 10^{-5}$	$1.22 * 10^{-3}$
Air at 325K	1.09	1006	0.0275	$1.97 * 10^{-5}$	$3.1 * 10^{-3}$

Table 8.1. Physical Properties and Constants.

The solution will assume instantaneous insertion of the billet, and that gaps between billet and furnace walls are sealed with refractory cement.

For natural convection from horizontal cylinders, McAdams recommends the following equation:

$$\frac{hD}{k} = 0.53\left[\frac{D^3\rho^2 g\,\beta\,\Delta T}{\mu^2}\cdot\left(\frac{c\mu}{k}\right)\right]^{0.25}$$

where ΔT = temperature difference between surface and fluid, and
h = film heat transfer coefficient.

This equation can be used for values of the Rayleigh number (within the square brackets) between 10^3 and 10^9. Physical properties are evaluated at a film temperature halfway between that of the surface of the cylinder, and that of the ambient fluid.

Inside the Furnace

The circulation of gas within the furnace does not conform to the patterns of natural circulation appropriate to the above correlation; in particular it has been developed for cases where the cylinder is hotter than its surroundings. Nevertheless it will be sufficiently accurate in the present case, since the greater part of the heat transfer will occur by radiation. This equation will also be applied to the ends.

At the commencement of heating, average film temperature = $\frac{300 + 1200}{2}$ = 750 K; and at the end of heating, average film temperature = $\frac{1000 + 1200}{2}$ = 1100 K.

To simplify the calculation, convection within the furnace will be evaluated using fluid properties at 925 K, as given in Table 8.1.

$$\text{Then } h = \frac{0.0639 * 0.53}{0.38} \cdot \left[\frac{0.38^3 * 0.382^2 * 9.81 * 1.22 * 10^{-3} * 1125}{4 * 10^{-5} * 0.0639}\right]^{0.25} \Delta T_F^{\frac{1}{4}}$$

$$= 1.277\ \Delta T_F^{\frac{1}{4}} = h_f$$

Outside the Furnace

At the commencement of heating, film temperature = 300 K. A preliminary rough calculation indicates the mean surface temperature of the billet protruding from the furnace will be 400 K at the end of the heating period, i.e. an average film temperature of 350 K.

Convection from the exposed portion will be evaluated using film properties at 325 K, as tabulated in Table 8.1.

$$\text{Then } h = \frac{0.0275 * 0.53}{0.38} \cdot \left[\frac{0.38^3 * 1.09^2 * 9.81 * 3.1 * 10^{-3} * 1006}{1.97 * 10^{-5} * 0.0275}\right]. \quad \Delta T_a^{\frac{1}{4}}$$

$$= 1.68\ \Delta T_a^{\frac{1}{4}} = h_a$$

The proposed arrangement of nodes is shown in Figure 8.4. A concentric subdivision of the billet into two zones, has been made. For greater accuracy further subdivision would be employed.

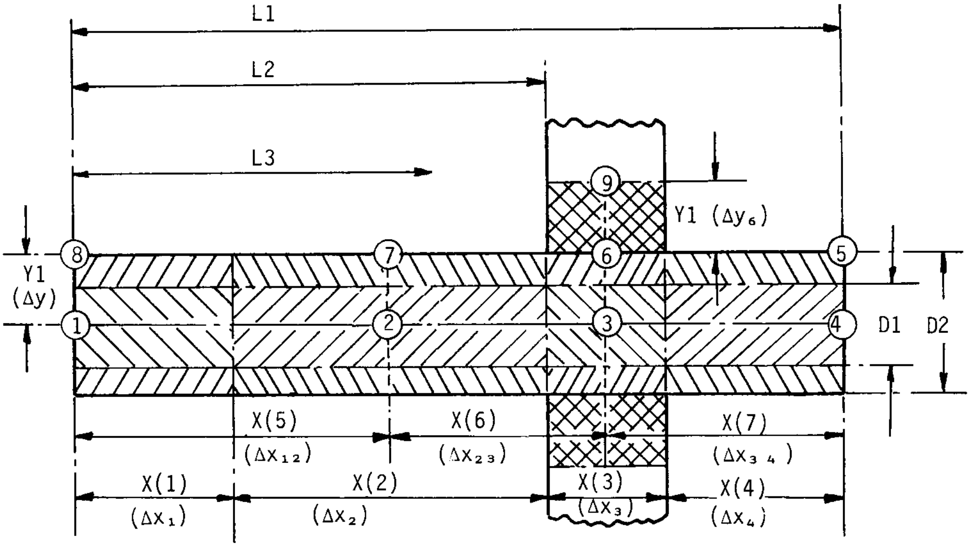


Figure 8.4. System of Nodes for Example 8.2.

The equations for the nodes can now be written, and these are tabulated in Table 8.2. The equations can be simplified and rearranged for use in the computer program.

These modifed equations are listed in Tables 8.3a and 8.3b. Note that D(1) etc. of Table 8.3 = $\frac{4}{\pi D_i^2}$ $*$ R.H.S. of the corresponding equation of Table 8.2.

When evaluating heatloads, the increment of sensible heat added to the billet = D(1) etc. $* \frac{\pi D_i^2}{4} * \Delta\Theta$.

A value for the mean refractory temperature at node 9 is required. This has been left as an input obtained by a simple preliminary calculation.

The computer program which has been developed from these equations is given below. A sample calculation based on the specified temperatures and dimensions is also given.

TABLE 8.2 - Node equations for Problem 8.2.

Node	Equation
1	$\rho c\Delta X_1 . \frac{\Pi D_1^2}{4} . \frac{(T_1^{t+1} - T_1^t)}{\Delta\Theta} = \frac{\Pi D_1^2}{4}\left[\varepsilon\sigma(T_F{}^4 - T_1^{t^4}) + h_f (T_F - T_1^t) + \frac{k}{\Delta X_{12}} . (T_2^t - T_1^t)\right] + \frac{\Pi D_1 \Delta X_1 k}{\Delta y} . (T_8^t - T_1^t)$
2	$\rho c\Delta X_2 \frac{\Pi D_1^2}{4} \frac{(T_2^{t+1} - T_2^t)}{\Delta\Theta} = \frac{\Pi D_1^2}{4}\left[\frac{k}{\Delta X_{12}} . (T_1^t - T_2^t) + \frac{k}{\Delta X_{23}} (T_3^t - T_2^t)\right] + \frac{\Pi D_1 \Delta X_2 k}{\Delta y} . (T_7^t - T_2^t)$
3	$\rho c\Delta X_3 \frac{\Pi D_1^2}{4} \frac{(T_3^{t+1} - T_3^t)}{\Delta\Theta} = \frac{\Pi D_1^2}{4}\left[\frac{k}{\Delta X_{23}} (T_2^t - T_3^t) + \frac{k}{\Delta X_{34}} (T_4^t - T_3^t)\right] + \frac{\Pi D_1 \Delta X_3 k}{\Delta y} . (T_6^t - T_3^t)$
4	$\rho c\Delta X_4 \frac{\Pi D_1^2}{4} \frac{(T_4^{t+1} - T_4^t)}{\Delta\Theta} = \frac{\Pi D_1^2}{4}\left[\varepsilon\sigma(T_a{}^4 - T_4^{t^4}) + h_a (T_a - T_4^t) + \frac{k}{\Delta X_{34}} (T_3^t - T_4^t)\right] + \frac{\Pi D_1 \Delta X_4 k}{\Delta y} . (T_5^t - T_4^t)$
5	$\rho c\Delta X_4 \frac{\Pi}{4} (D_2^2 - D_1^2) . \frac{(T_5^{t+1} - T_5^t)}{\Delta\Theta} = \left[\frac{\Pi}{4}(D_2^2 - D_1^2) + \Pi D_2 \Delta X_4\right] * \left[\varepsilon\sigma (T_a{}^4 - T_5^{t^4}) + h_a (T_a - T_5^t)\right]$ $+ \frac{\Pi}{4} (D_2^2 - D_1^2) . \frac{k}{\Delta X_{34}} (T_6^t - T_5^t) + \frac{\Pi D_1 \Delta X_4 k}{\Delta y} . (T_4^t - T_5^t)$
6	$\rho c\Delta X_3 \frac{\Pi}{4} (D_2^2 - D_1^2) . \frac{(T_6^{t+1} - T_6^t)}{\Delta\Theta} = \frac{\Pi}{4} (D_2^2 - D_1^2)\left[\frac{k(T_5^t - T_6^t)}{\Delta X_{34}} + \frac{k(T_7^t - T_6^t)}{\Delta X_{23}}\right] + \frac{\Pi D_1 \Delta X_3}{\Delta y} . k (T_3^t - T_6^t) + \frac{\Pi D_2 \Delta X_3 k_r}{\Delta y_6} (T_a - T_6^t)$
7	$\rho c\Delta X_2 \frac{\Pi}{4} (D_2^2 - D_1^2) . \frac{(T_7^{t+1} - T_7^t)}{\Delta\Theta} = \Pi D_2 \Delta X_2 \left[\varepsilon\sigma (T_F{}^4 - T_7^{t^4}) + h_f (T_F - T_7^t)\right] + \frac{\Pi}{4} (D_2^2 - D_1^2)\left[\frac{k}{\Delta X_{12}} (T_8^t - T_7^t) + \frac{k}{\Delta X_{23}} *\right.$ $\left.(T_6^t - T_7^t)\right] + \frac{\Pi D_1 \Delta X_2 k}{\Delta y} . (T_2^t - T_7^t)$
8	$\rho c\Delta X_1 \frac{\Pi}{4} (D_2^2 - D_1^2) \frac{(T_8^{t+1} - T_8^t)}{\Delta\Theta} = \left[\frac{\Pi}{4} (D_2^2 - D_1^2) + \Pi D_2 \Delta X_1\right] * \left[\varepsilon\sigma (T_F{}^4 - T_8^{t^4}) + h_f (T_F - T_8^t)\right]$ $+ \frac{\Pi}{4} (D_2^2 - D_1^2) . \frac{k}{\Delta X_{12}} . (T_7^t - T_8^t) + \frac{\Pi D_1 \Delta X_1 k}{\Delta y} . (T_1^t - T_8^t)$

TABLE 8.3a - Equations for inner nodes rearranged for use in computer program (Problem 8.2)

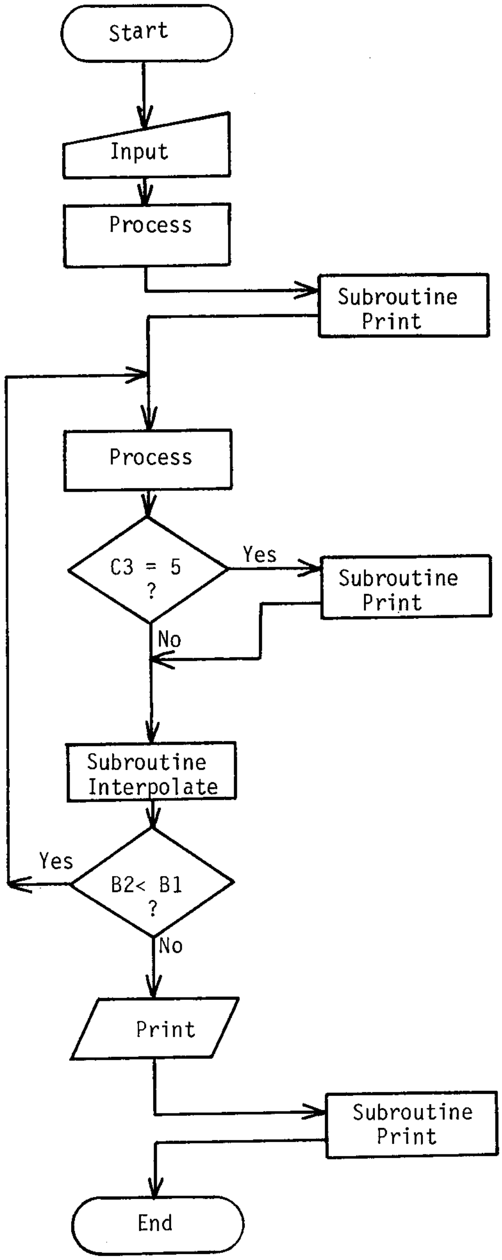
Note: In this and preceding table, the use of a double superscript. Thus $T_1^{t\,4}$ indicates temperature of node 1 at time t, raised to the fourth power.

Node	Equation
1	$D(1) = \varepsilon\sigma(T_F^4 - T_1^{t\,4}) + 1.277\,(T_F - T_1^t)^{1.25} + \frac{k}{\Delta X_{12}} \cdot (T_2^t - T_1^t) + \frac{4k\Delta x_1(T_8^t - T_1^t)}{D_1\Delta y}$ $T_1^{t+1} = T_1^t + D(1) * \frac{\Delta\Theta}{\rho C\Delta X_1}$
2	$D(2) = k\left[\frac{(T_1^t - T_2^t)}{\Delta X_{12}} + \frac{(T_3^t - T_2^t)}{\Delta X_{23}} + \frac{4\Delta x_2(T_7^t - T_2^t)}{D_1\Delta y}\right]$ $T_2^{t+1} = T_2^t + D(2) * \frac{\Delta\Theta}{\rho C\Delta X_2}$
3	$D(3) = k\left[\frac{(T_2^t - T_3^t)}{\Delta X_{23}} + \frac{(T_4^t - T_3^t)}{\Delta X_{34}} + \frac{4\Delta x_3(T_6^t - T_3^t)}{D_1\Delta y}\right]$ $T_3^{t+1} = T_3^t + D(3) * \frac{\Delta\Theta}{\rho C\Delta X_3}$
4	$D(4) = \varepsilon\sigma\,(T_a^4 - T_4^{t\,4}) + 1.68\,(T_a - T_4^t)^{1.25} + k\left[\frac{(T_3^t - T_4^t)}{\Delta X_{34}} + \frac{4\Delta x_4\,(T_5^t - T_4^t)}{D_1\Delta y}\right]$ $T_4 = T_4 + D(4) * \frac{\Delta\Theta}{\rho C\Delta X_3}$

TABLE 8.3b - Equations for outer nodes rearranged for use in computer program (Problem 8.2)

Node	Equation
5	$D(5) = \left[\varepsilon\sigma\,(T_a^4 - T_5^{t^4}) + 1.68\,(T_a - T_5^t)^{1.25}\right] * \left[(D_2^2 - D_1^2) + D_2\Delta X_4\right] + k\left[\frac{(D_2^2 - D_1^2)\,(T_6^t - T_5^t)}{\Delta X_{34}}\right.$ $\left. + \frac{4D_1\,\Delta X_4\,(T_4^t - T_5^t)}{\Delta y}\right]$ $T_5^{t+1} = T_5^t + D(5) * \frac{\Delta\Theta}{\rho C\,(D_2^2 - D_1^2)\,\Delta X_4}$
6	$D(6) = k\,(D_2^2 - D_1^2)\,.\left[\frac{(T_5^t - T_6^t)}{\Delta X_{34}} + \frac{(T_7^t - T_6^t)}{\Delta X_{23}}\right] + \frac{4kD_1\,\Delta X_3(T_3^t - T_6^t)}{\Delta y}$ $T_6^{t+1} = T_6^t + D(6) * \frac{\Delta\Theta}{\rho c(D_2^2 - D_1^2)\Delta X_3} + 4k_r D_2\Delta X_3(T_a^t - T_6^t)/\Delta y$
7	$D(7) = \left[\varepsilon\sigma\,(T_F^4 - T_7^{t^4}) + 1.277\,(T_F - T_7^t)^{1.25}\right] * 4D_2\Delta X_2 + k\,.\,\frac{4D_1\Delta X_2\,(T_2^t - T_7^t)}{\Delta y} + k\,.(D_2^2 - D_1^2) *$ $\left[\frac{(T_8^t - T_7^t)}{\Delta X_{12}} + \frac{(T_6^t - T_7^t)}{\Delta X_{23}}\right]$ $T_7^{t+1} = T_7^t + D(7) * \frac{\Delta\Theta}{\rho c(D_2^2 - D_1^2)\Delta X_2}$
8	$D(8) = \left[\varepsilon\sigma\,(T_F^4 - T_8^{t^4}) + 1.277\,(T_F - T_8^t)^{1.25}\right] * \left[4D_2\Delta X_1 + (D_2^2 - D_1^2)\right] + k\left[\frac{4D_1\Delta X_1(T_1^t - T_8^t)}{\Delta y}\right.$ $\left. + \frac{(D_2 - D_1)\,(T_7^t - T_8^t)}{\Delta X_{12}}\right]$ $T_8^{t+1} = T_8^t + D(8) * \frac{\Delta\Theta}{\rho c(D_2^2 - D_1^2)\Delta X_1}$

TRSCOND3.BAS



```
10  REM ****************************************************
20  REM - PROGRAM TRSCOND3.BAS   CALCULATES TEMPERATURE
30  REM - DISTRIBUTION AND HEATING TIME FOR A BILLET
40  REM - PARTIALLY INSERTED IN A FURNACE
50  REM - PROGRAM NOMENCLATURE
60  REM - A1    -  Ambient temperature
70  REM - A2    -  Intermediate value using A1
80  REM - B1    -  Required billet temperature
90  REM - B2    -  Billet temperature at distance L3
100 REM - C1    -  Time interval between iterations
110 REM - C2    -  Total time taken to latest iteration
120 REM - C3    -  Number of iterations since last printout
130 REM - D(J) -  Right hand sides of node equations listed
140 REM           in Table 8.3
150 REM - D1    -  Inner element diameter
160 REM - D2    -  Billet diameter
170 REM - D3    -  Intermediate value using D1 & D2
180 REM - D4,D5    Intermediate forms of D(4),D(5)
190 REM           respectively
200 REM - E1    -  Emissivity
210 REM - E2    -  E1*Stefan constant
220 REM - F1    -  Furnace temperature
230 REM - F2    -  Intermediate value using F1
240 REM - K1,K2    Thermal conductivities of billet,
250 REM           refractory respectively
260 REM - L1    -  Billet overall length
270 REM - L2    -  Billet length within furnace
280 REM - L3    -  Billet length to be heated above
290 REM           designated temperature
300 REM - M2    -  Multiplier used in evaluating Q2
310 REM - Q1    -  Sensible heat in billet
320 REM - Q2    -  Related to rate of heat loss
330 REM           to atmosphere
340 REM - Q3    -  Heat to atmosphere
350 REM - R1    -  Density of billet
360 REM - S(J)     Temperatures at the nodes at the latest
370 REM           iteration
380 REM - S1    -  Specific heat of billet
390 REM - S2    -  Product R1*S1
400 REM - T(J)     Temperatures at the nodes at the previous
410 REM           iteration
420 REM - T9    -  Mean refractory temperature
430 REM - X(J)     Length of billet segments as shown on
440 REM           Figure 8.4
450 REM - XJ    -  Distance to node measured from furnace end
460 REM           of billet
470 REM - Y1    -  Radial distance between nodes
480 REM - ZJ    -  Used to locate end of heated section
490 REM - PROGRAM DESCRIPTION
500 REM - LINES 1010 - 1240   Values of dimensions and
510 REM - physical properties are entered
520 REM - LINES 1340 - 1390   Distances between nodes
530 REM - are evaluated
540 REM - LINES 1400 - 1460   Initially, temperatures at all
550 REM - nodes are set to ambient temperature value A1
560 REM - LINES 1490 - 1750   Values of D(J) as tabulated in
570 REM - Table 8.3 are evaluated
580 REM - LINES 1770 - 1830   Heat loads are evaluated
590 REM - LINES 1840 - 1890   New values of node
600 REM - temperatures are calculated and stored in matrix S
610 REM - LINES 1900 - 2040   Time and iteration counts are
```

```
620 REM - updated and values at each 5th iteration are
630 REM - printed (lines 1920,1940).   The temperature at
640 REM - the end of the billet section (L3) required to be
650 REM - above the predetermined value B1, is calculated
660 REM - using subroutine "Interpolate" (line 1960); if
670 REM - this value is less than B1, then the program
680 REM - executes another iteration (line 1970).
690 REM - Otherwise, final values are printed out &
700 REM - execution ceases (lines 1990 - 2040)
710 REM - SUBROUTINE PRINT (LINES 2060 - 2230)   First the
720 REM - distances of the nodes,measured from the hottest
730 REM - end of the billet are printed.  Below them are
740 REM - printed temperature values at nodes 8,7,6 & 5.
750 REM - Below these are printed temperature values at
760 REM - nodes 1,2,3 & 4
770 REM - SUBROUTINE INTERPOLATE (LINES 2270 - 2350)   The
780 REM - end of the billet section required to be above
790 REM - temperature B1, does not necessarily coincide with
800 REM - a node.  The number of the node on each side of
810 REM - this end is found (lines 2290,2300), then the
820 REM - value of temperature at the end of the section
830 REM - (i.e. distance L3 from the hottest end of the
840 REM - billet & denoted B2) is found by interpolating
850 REM - linearly between the values of temperature of the
860 REM - adjoining nodes (line 2310).
870 REM ************************************************
1000 REM - DIMENSIONS AND PHYSICAL PROPERTIES ************
1010 PRINT "GIVE ALL DIMENSIONS IN METRES"
1020 INPUT "BILLET DIAMETER";D2
1030 INPUT "INNER ELEMENT DIAMETER";D1
1040 INPUT "THICKNESS OF FURNACE WALL";X(3)
1050 INPUT "LENGTH OF BILLET";L1
1060 INPUT "BILLET LENGTH WITHIN FURNACE";L2
1070 IF L2+X(3)<L1 THEN 1110
1080 PRINT "PROGRAM DOES NOT HANDLE THIS - OUTER END"
1090 PRINT "OF BILLET WITHIN FURNACE WALL"
1100 GOTO 2360
1110 PRINT "LENGTH OF BILLET TO BE HEATED ABOVE"
1120 INPUT "DESIGNATED TEMPERATURE";L3
1130 PRINT "PHYSICAL PROPERTIES OF BILLET,SI UNITS:"
1140 INPUT "EMISSIVITY";E1
1150 INPUT "THERMAL CONDUCTIVITY";K1
1160 INPUT "DENSITY";R1
1170 INPUT "SPECIFIC HEAT";S1
1180 INPUT "THERMAL CONDUCTIVITY OF REFRACTORY";K2
1190 PRINT "GIVE ALL TEMPERATURES IN DEG KELVIN"
1200 INPUT "FURNACE TEMP";F1
1210 INPUT "REQUIRED BILLET TEMP";B1
1220 INPUT "AMBIENT TEMP";A1
1230 INPUT "MEAN REFRACTORY TEMP";T9
1240 INPUT "TIME INTERVAL IN SECONDS";C1
1250 S2=S1*R1
1260 REM - EMISSIVITY MULTIPLIED BY ***********************
1270 REM - STEFAN BOLZMAN CONSTANT,SI UNITS ***************
1280 E2=E1*5.6688E-08
1290 Y1=D2/2
1300 D3=D2^2-D1^2
1310 PI=3.14159
1320 F2=F1^4
1330 A2=A1^4
1340 X(7)=L1-L2-X(3)/2
```

```
1350 X(4)=L1-L2-X(3)
1360 X(5)=(L2+X(3)/2)/2
1370 X(6)=X(5)
1380 X(2)=2*(X(6)-X(3)/2)
1390 X(1)=L2-X(2)
1400 FOR J=1 TO 8
1410 S(J)=A1
1420 NEXT
1430 GOSUB 2050
1440 FOR J=1 TO 8
1450 T(J)=S(J)
1460 NEXT
1470 REM - EVALUATE RIGHT HAND SIDES *********************
1480 REM - OF NODE EQUATIONS *****************************
1490 D(1)=E2*(F2-T(1)^4)+1.277*(F1-T(1))^1.25
1500 D(1)=D(1)+K1*(T(2)-T(1))/X(5)
1510 D(1)=D(1)+4*K1*X(1)*(T(8)-T(1))/(D1*Y1)
1520 D(2)=(T(1)-T(2))/X(5)+(T(3)-T(2))/X(6)
1530 D(2)=K1*(D(2)+4*X(2)*(T(7)-T(2))/(Y1*D1))
1540 D(3)=(T(2)-T(3))/X(6)+(T(4)-T(3))/X(7)
1550 D(3)=K1*(D(3)+4*X(3)*(T(6)-T(3))/(Y1*D1))
1560 IF A1>=T(4) THEN 1580
1570 D4=-E2*(T(4)^4-A2)-1.68*(T(4)-A1)^1.25
1580 D(4)=K1*((T(3)-T(4))/X(7)+4*X(4)*(T(5)-T(4))/(Y1*D1))
1590 IF A1>=T(4) THEN 1610
1600 D(4)=D4+D(4)
1610 IF A1>=T(5) THEN 1630
1620 D5=(-E2*(T(5)^4-A2)-1.68*(T(5)-A1)^1.25)*(D3+4*D2*X(4))
1630 D(5)=K1*(D3*(T(6)-T(5))/X(7)+4*D1*X(4)*(T(4)-T(5))/Y1)
1640 IF A1>=T(5) THEN 1660
1650 D(5)=D5+D(5)
1660 D(6)=K1*D3*((T(5)-T(6))/X(7)+(T(7)-T(6))/X(6))
1670 D(6)=D(6)+4*K1*D1*X(3)*(T(3)-T(6))/Y1
1680 D(6)=D(6)+4*K2*D2*X(3)*(T9-T(6))/Y1
1690 D(7)=(E2*(F2-T(7)^4)+1.277*(F1-T(7))^1.25)*D2*X(2)*4
1700 D(7)=D(7)+4*K1*D1*X(2)*(T(2)-T(7))/Y1
1710 D(7)=D(7)+K1*D3*((T(8)-T(7))/X(5)+(T(6)-T(7))/X(6))
1720 D(8)=E2*(F2-T(8)^4)+1.277*(F1-T(8))^1.25
1730 D(8)=D(8)*(D2*X(1)*4+D3)
1740 D(8)=D(8)+K1*D1*X(1)*4*(T(1)-T(8))/Y1
1750 D(8)=D(8)+K1*D3*(T(7)-T(8))/X(5)
1760 REM - EVALUATE HEAT LOADS *****************************
1770 Q1=Q1+C1*PI*D1^2*(D(1)+D(2)+D(3)+D(4))/4
1780 Q1=Q1+C1*PI*(D(5)+D(6)+D(7)+D(8))/4
1790 Q2=D1^2*(E2*(T(4)^4-A2)+1.68*(T(4)-A1)^1.25)/4
1800 M2=D3/4+D2*X(4)
1810 Q2=Q2+M2*(E2*(T(5)^4-A2)+1.68*(T(5)-A1)^1.25)
1820 Q2=Q2+D2*X(8)*(E2*(T(6)^4-A2)+1.68*(T(6)-A1)^1.25)
1830 Q3=Q3+Q2*C1*PI
1840 FOR J=1 TO 4
1850 S(J)=T(J)+C1*D(J)/(S2*X(J))
1860 NEXT
1870 FOR J=5 TO 8
1880 S(J)=T(J)+C1*D(J)/(S2*X(9-J)*D3)
1890 NEXT
1900 C2=C2+C1
1910 C3=C3+1
1920 IF C3=5 GOTO 1940
1930 GOTO 1960
1940 GOSUB 2050
1950 C3=0
```

```
1960 GOSUB 2250
1970 IF B2<B1 THEN 1440
1980 PRINT
1990 PRINT "TIME TO REACH";B1;"DEGREES K=";C2;"SECONDS"
2000 PRINT "HEAT TO BILLET=";Q1;"JOULES"
2010 PRINT "HEAT TO ATMOSPHERE=";Q3;"JOULES"
2020 PRINT "TOTAL HEAT REQUIREMENTS=";Q1+Q3;"JOULES"
2030 GOSUB 2050
2040 GOTO 2360
2050 REM - PRINT SUBROUTINE ********************************
2060 PRINT
2070 PRINT;C2;"SECONDS"
2080 XJ=0
2090 PRINT USING "#.##   ";0;
2100 FOR J=5 TO 7
2110 XJ=XJ+X(J)
2120 PRINT USING "#.##   ";XJ;
2130 NEXT
2140 PRINT " METRES"
2150 FOR J=8 TO 5 STEP -1
2160 PRINT USING "####   ";S(J);
2170 NEXT
2180 PRINT " DEGREES"
2190 FOR J=1 TO 4
2200 PRINT USING "####   ";S(J);
2210 NEXT
2220 PRINT " DEGREES"
2230 RETURN
2240 REM ***************************************************
2250 REM - SUBROUTINE INTERPOLATE   CALCULATES TEMPERATURE
2260 REM - AT THE END OF LENGTH L3 *************************
2270 XJ=0
2280 FOR J=5 TO 7
2290 ZJ=XJ+X(J)
2300 IF L3>ZJ THEN 2330
2310 B2=S(J-4)-(S(J-4)-S(J-3))*(L3-XJ)/X(J)
2320 GOTO 2350
2330 XJ=ZJ
2340 NEXT
2350 RETURN
2360 END

RUN
GIVE ALL DIMENSIONS IN METRES
BILLET DIAMETER? .38
INNER ELEMENT DIAMETER? .25
THICKNESS OF FURNACE WALL? .35
LENGTH OF BILLET? 2.3
BILLET LENGTH WITHIN FURNACE? 1.35
LENGTH OF BILLET TO BE HEATED ABOVE
DESIGNATED TEMPERATURE? 1.16
PHYSICAL PROPERTIES OF BILLET,SI UNITS:
EMISSIVITY? .94
THERMAL CONDUCTIVITY? 45
DENSITY? 7850
SPECIFIC HEAT? 460
THERMAL CONDUCTIVITY OF REFRACTORY? 1.23
GIVE ALL TEMPERATURES IN DEG KELVIN
FURNACE TEMP? 1200
REQUIRED BILLET TEMP? 900
```

```
AMBIENT TEMP? 300
MEAN REFRACTORY TEMP? 870
TIME INTERVAL IN SECONDS? 400

 0 SECONDS
0.00    0.76    1.53    2.30     METRES
 300     300     300     300     DEGREES
 300     300     300     300     DEGREES

 2000 SECONDS
0.00    0.76    1.53    2.30     METRES
1114     972     349     301     DEGREES
 997     748     333     301     DEGREES

 4000 SECONDS
0.00    0.76    1.53    2.30     METRES
1191    1143     417     304     DEGREES
1178    1068     407     304     DEGREES

 6000 SECONDS
0.00    0.76    1.53    2.30     METRES
1198    1180     485     311     DEGREES
1195    1156     477     311     DEGREES

 8000 SECONDS
0.00    0.76    1.53    2.30     METRES
1199    1189     543     320     DEGREES
1198    1177     537     321     DEGREES

 10000 SECONDS
0.00    0.76    1.53    2.30     METRES
1199    1191     591     331     DEGREES
1199    1182     586     332     DEGREES

 12000 SECONDS
0.00    0.76    1.53    2.30     METRES
1200    1192     630     343     DEGREES
1199    1185     626     344     DEGREES

TIME TO REACH 900 DEGREES K= 12800 SECONDS
HEAT TO BILLET= 5.532156E+08 JOULES
HEAT TO ATMOSPHERE= 1752116 JOULES
TOTAL HEAT REQUIREMENTS= 5.549678E+08 JOULES
```

PROBLEMS - CHAPTER 8

1. The metal slab of Example 8.1 is subjected to a furnace environment at 1300 K on one side only, the environment on the other side remaining constant at 300 K. The combined coefficient of heat transfer by convection and radiation from the slab to ambient air (h_a) may be taken as constant at 20 W/m^{2}K. Using the physical properties of Example 8.1, determine the time required to reach steady state and the temperature distribution within the slab at that time:

   a. By a Schmidt plot;
   b. By modifying program TRSCOND1. This might be done by modifying equation 8.13 for node 6, and assuming this node to be on the

cold face of the slab. By analogy with equation 8.9, the heat balance equation would be:

$$(T_a - T_6^t)h_a + \frac{k}{\Delta x}(T_5^t - T_6^t)$$

$$= \frac{\rho c \Delta x/2}{\Delta \Theta}(T_6^{t+1} - T_6^t)$$

2. In the moulding of rubber tyres, we might approximate the tyre to a hollow torus, the internal and external faces of which are raised to a constant elevated temperature. Write a program for this configuration which will compute the time required to reach any chosen vulcanisation temperature within the tyre.

Volume of a torus = $2\pi^2 Rr^2$
Surface area " = $4\pi^2 Rr$

where r = radius of the cross section;
R = mean radius of the ring.

Physical data on rubber is available (7).

3. Suppose that the steel billet of Example 8.5 is withdrawn from the furnace after heating to the required temperature. Determine the temperatures at the nodes, one hour after withdrawal:

a. Approximately, by a manual calculation, working from the temperatures which have been generated by the progrma TRSCOND3.

b. By modifying program TRSCOND3. This might be done by ascribing to furnace temperature F1 and refractory temperature T9 the value of the ambient air temperature A1. What further modifications to the program would yield a better answer?

4. Modify program TRSCOND3 so that the number of nodes employed can be selected at the keyboard.

REFERENCES

1. F. Kreith, Principles of Heat Transfer, 3rd Edition, International Textbook Co., Scranton Pa U.S.A., 1973.
2. H.S. Carslaw and J.C. Jager, Conduction of Heat in Solids, Clarendon Press, Oxford, U.K., 1947.
3. J.P. Holman, Heat Transfer, 3rd Edition, McGraw Hill Book Co., New York, U.S.A., 1972.
4. G.R. Gaumer, Stability of Three Finite Difference Methods of Solving for Transient Temperatures, Journal of the American Rocket Society, p. 159S, 1962.
5. T.M. Shih, Numerical Heat Transfer, Hemisphere/Springer, Berlin, 1984.
6. G.D. Smith, Numerical Solution of Partial Differential Equations: Finite Difference Methods, Clarendon Press, Oxford, 1978.
7. K.- Othmer, Rubber, Natural, Encyclopedia of Chemical Technology, 3rd Edition, John Wiley & Sons, New York, U.S.A.

Chapter 9

AUTOMATIC CONTROL

Problems in control are mostly concerned with transient behaviour (1), (2), (3), (4). Even quite simple systems when tackled from a theoretical standpoint involve the solution of complicated differential equations. The use of Laplace transforms was applied to this type of problem in the 1940's (1).

The concept of the transfer function was defined as the ratio of the Laplace transforms of the responding variable to that of the disturbing variable,

$$G(s) = \frac{Y(s)}{X(s)}$$

where $G(s)$ = transfer function for process element
$Y(s)$ = Laplace transform of responding variable
$X(s)$ = Laplace transform of the disturbing variable.

The procedure involves the drawing up of a Block Diagram, each block on the diagram representing a process step, or control element. A transfer function based on a linear differential equation is then written for each element. These elements are then combined according to the rules of algebra, and finally the solution is obtained by inversion back into the time domain, of the equation representing the system and the disturbing variable. The use of Laplace transforms has been likened to the use of logarithms (1). By converting numbers to their logarithms, the processes of multiplication and division are replaced by the simpler processes of addition and subtraction; on completion of the manipulations the transformation back to the real number system is made using anti-logarithms. By the use of Laplace transforms in this way, we obtain the response, with respect to time of the responding, that is the output, variable:

$$L\,\{Y(t)\} = G(s)\,.\,X(s)$$

Even this procedure becomes unwieldy when the system is greater than third order. Graphical methods were therefore developed which allow systems of any order to be represented, namely the Root Locus and Frequency Response techniques. The Root Locus technique, first published in 1948 (1), although a laborious procedure, does permit a full evaluation of the system response to be made. Unfortunately this method is unsuitable where the system involves a transportation lag, so common in chemical engineering situations. The Frequency Response technique does permit the inclusion of the transportation lag, but on the other hand it does not provide a complete prediction of the system response.

Both these procedures can of course be simulated on the digital computer, but this is largely unnecessary, as a program can be written to determine the time response of a system directly, using finite differences. As with the older methods, it is convenient to start with a block diagram. A differential equation is then written for each component of the diagram, but instead of transforming these by the Laplace method, each is rewritten as a finite difference equation (5), (6), (7), (8).

In the remainder of this chapter, both approaches will be demonstrated for some simple open and closed loop systems.

The First Order Function

Many simple lumped parameter systems may be represented by a first order differential equation. Examples include the thermometer, liquid level systems, and mixing, heat transfer, mass transfer and chemical reaction when these are carried out in a stirred tank. Distributed parameter systems may often be represented by a series of such first order elements. For example, the hydraulic behaviour of a distillation column containing N trays, might be represented as a series of N first order elements.

The transfer function and corresponding finite difference equation will be derived for the heated and stirred tank shown in Figure 9.1.

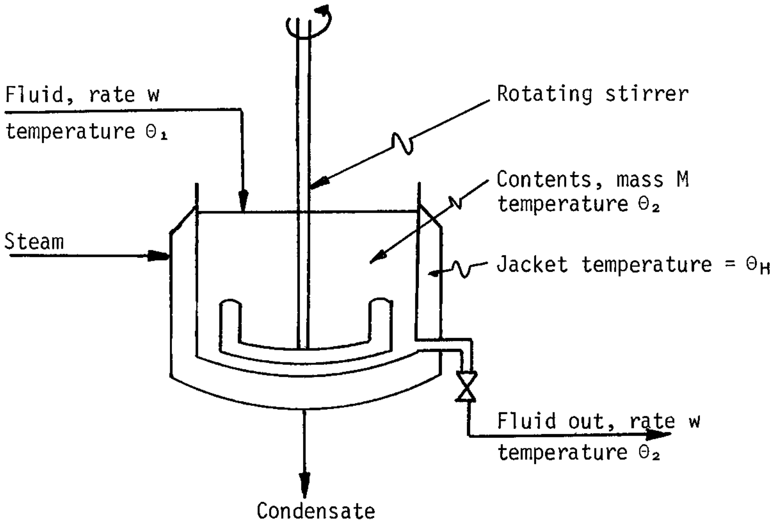


Figure 9.1. Heat transfer to stirred tank.

A heat balance over the tank can be written as:

$$wc\theta_1 - wc\theta_2 + UA\,(\theta_H - \theta_2) = mc\,\frac{d\theta_2}{dt} \qquad (9.1)$$

where w = mass flowrate of liquid entering and leaving the tank;
c = specific heat of the liquid;
m = mass of liquid contained in the tank, assumed to be constant (including if necessary a mass to represent the tank walls and stirrer);
θ_1 = temperature of entering liquid;
θ_2 = temperature of tank contents, assumed to be uniform, and therefore equal to the temperature of the liquid leaving;
θ_H = jacket temperature; assumed to be constant;
U = overall coefficient of heat transfer, jacket to tank contents;
A = heat transfer area.

At steady state, this differential equation becomes:

$$wc\theta_{1s} - wc\theta_{2s} + UA\,(\theta_H - \theta_{2s}) = 0 \qquad (9.2)$$

where θ_{1s}= steady state value of entering liquid temperature;
θ_{2s}= steady state value of leaving liquid temperature.

Subtracting 9.2 from 9.1 and rewriting in terms of Deviation Variables, we obtain:

$$wcT_1 - wcT_2 + UAT_2 = mc\,\frac{dT_2}{dt}$$

where $T_1 = \theta_1 - \theta_{1s}$ = the deviation of $\theta_1$from its steady state value
and $T_2 = \theta_2 - \theta_{2s}$ similarly.

Rearranging this equation:

$$wcT_1 = (wc + UA)\,T_2 + mc\frac{dT_2}{dt}$$

$$\text{or } KT_1 = T_2 + \tau\,\frac{dT_2}{dt} \qquad (9.3)$$

where K, the gain constant = $\dfrac{wc}{wc + UA}$, dimensionless

τ, the time constant = $\dfrac{mc}{wc + UA}$, dimension of time.

We can rewrite this equation in more general terms as:

$$KX = Y + \tau\,\frac{dY}{dt} \qquad (9.4)$$

where X = the deviation of the disturbing variable from its steady state value;
Y = the deviation of the responding variable from its steady state value.

a. We can transform equation 9.4 into the standard form of a first order transfer function. Before doing so, it should be noted that the variables X, Y etc. above are all time dependent and more correctly should be written $X(t)$, $Y(t)$, etc. indicating that the value is that which obtains at time t.

By taking Laplace transforms of equation 9.4, we obtain:

$$KX(s) = Y(s) + \tau s\, Y(s);$$

where $X(s)$ = Laplace transform of the function represented by X above;
$Y(s)$ = Laplace transform of the function represented by Y above;
$sY(s)$ = Laplace transform of $\frac{dY}{dt}$

This equation rearranged as:

$$\frac{Y(s)}{X(s)} = \frac{K}{Ts+1} \tag{9.5}$$

is now in the form of a first order transfer function.

b. Alternatively, equation 9.4 can be written in finite difference notation as:

$$KX = Y_2 + \frac{T\,(Y_2 - Y_1)}{\Delta t} \tag{9.6}$$

where Y_1 = deviation value of response at time t
Y_2 = deviation value of response at time t + Δt
and Δt = time interval.

Rearranging the equation,

$$KX = Y_2(1 + \frac{\tau}{\Delta t}) - Y_1 \frac{\tau}{\Delta t}$$

$$\therefore \quad Y_2 = \frac{KX + Y_1\tau/\Delta t}{1 + \tau/\Delta t} \tag{9.7}$$

This equation can be used in a computer program to represent a first order system component.

The Second Order Function

The large class of mechanical devices involving both a spring and a damping resistance are described by this function. This includes vehicle suspension systems, hydraulic servomechanisms, and the pneumatic control valve. Everyday examples include the hydraulic door closer and the liquid - filled manometer.

As an example, the transfer function and corresponding finite difference equation will be derived for the pneumatic control valve shown in Figure 9.2.

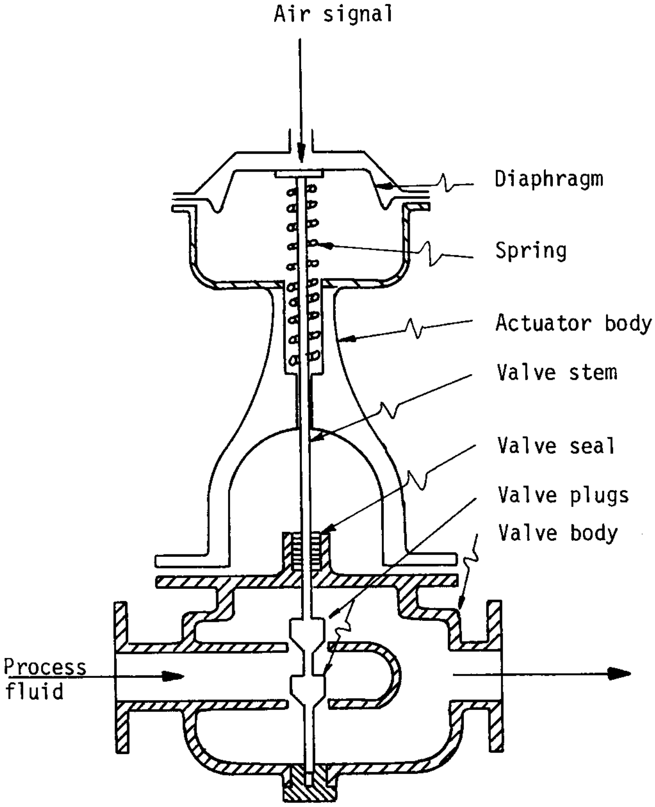


Figure 9.2. Direct-acting pneumatically operated control valve.

Forces involved in the operation of this mechanism are:

1. The force exerted by air pressure applied to the top of the diaphragm, acting downwards. This force F(t) = PA
where P = air pressure (gauge)
A = diaphragm area

2. The force exerted by the return spring, acting upwards. This force = -HL
where H = Hooks constant (force/length)
L = distance travelled by the end of the spring at the underside of the diaphragm.

the negative sign indicates that this force operates in the opposite direction to that of the diaphragm.

3. The force to overcome viscous friction acting at the valve guides and the valve seal. This force = -CdL/dt
where C = a coefficient of friction under laminar (viscous) flow conditions.

4. The force to overcome the inertia of the moving parts. This force = $-Md^2L/dt^2$
where M = mass of the moving parts (valve stem and diaphragm).

5. Pressure drop forces acting across the valve plugs. With a double-seated valve such as that shown, the forces across the seats act in opposition to one another and may be assumed to have a resultant of zero.

The sum of forces 1 to 3 will result in acceleration of the Mass M;

$$\frac{Md^2L}{dt^2} = F(t) - HL - \frac{CdL}{dt}$$

which can be rearranged as:

$$\frac{M}{H} \cdot \frac{d^2L}{dt^2} + \frac{C}{H} \cdot \frac{dL}{dt} + L = \frac{F(t)}{H} \qquad (9.8)$$

Rewriting in standard nomenclature we obtain:

$$\tau^2\frac{d^2Y}{dt^2} + 2\zeta\tau\frac{dY}{dt} + Y = X \qquad (9.9)$$

where X and Y are the disturbing and responding variables as before.

τ, (tau) the time constant = $\left[\frac{M}{H}\right]^{\frac{1}{2}}$, dimensions of time

ζ, (zeta) the damping coefficient = $\frac{1}{2}\left[\frac{C^2}{MH}\right]^{\frac{1}{2}}$, dimensionless

a) If we assume the moving parts to be at rest before the disturbance occurs, then the Laplace transform of equation 9.9 is

$$\tau^2s^2Y(s) + 2\zeta\tau Y(s) + Y(s) = X(s)$$

This equation rearranged as:

$$\frac{Y(s)}{X(s)} = \frac{1}{\tau^2s^2 + 2\zeta\tau s + 1} \qquad (9.10)$$

is now in the form of a second order transfer function.

b) Alternatively, the differential equation of 9.9 can be rewritten in finite difference notation as:

$$\frac{\tau^2\Delta^2 Y}{\Delta t^2} + \frac{2\zeta\tau\Delta Y}{\Delta t} + Y = X \tag{9.11}$$

It should be noted in what follows, the interval Δt is assumed constant in size; subscripts 1, 2, 3, etc. indicate values at successive time intervals.

Hence $\frac{\Delta Y}{\Delta t} = \frac{Y_2 - Y_1}{\Delta t}$

and $\frac{\Delta^2 Y}{\Delta t^2} = \frac{\frac{Y_2 - Y_1}{\Delta t} - \frac{Y_1 - Y_0}{\Delta t}}{\Delta t}$

where Y_2, Y_1, Y_0, indicate values of Y at times t $+\Delta t$, t, and t-Δt respectively.

Hence the equation is:

$$\tau^2 \frac{\left[\frac{Y_2 - Y_1}{\Delta t} - \frac{Y_1 - Y_0}{\Delta t}\right]}{\Delta t} + \frac{2\zeta\tau(Y_2 - Y_1)}{\Delta t} + Y_2 = X \tag{9.12}$$

Rearrange the equation to solve for Y_2:

$$\frac{\tau^2}{\Delta t^2}(Y_2 - 2Y_1 + Y_0) + \frac{2\zeta\tau(Y_2 - Y_1)}{\Delta t} + Y_2 = X$$

$$\frac{\tau^2}{\Delta t^2} \cdot Y_2 + \frac{\tau^2}{\Delta t^2}(Y_0 - 2Y_1) + \frac{2\zeta\tau Y_2}{\Delta t} - \frac{2\zeta\tau Y_1}{\Delta t} + Y_2 = X$$

$$Y_2\left(\frac{\tau^2}{\Delta t^2} + \frac{2\zeta\tau}{\Delta t} + 1\right) = X - \frac{\tau^2}{\Delta t^2}(Y_0 - 2Y_1) + \frac{2\zeta\tau Y_1}{\Delta t}$$

$$\therefore \quad Y_2 = \frac{X + \frac{\tau^2}{\Delta t^2}(2Y_1 - Y_0) + \frac{2\zeta\tau Y_1}{\Delta t}}{\frac{\tau^2}{\Delta t^2} + \frac{2\zeta\tau}{\Delta t} + 1} \tag{9.13}$$

This equation can be used in a computer program to represent a second order system component.

THE CONTROLLER MECHANISM

This consists of two parts:

1. The Comparator

This performs the arithmetic operation of subtraction. The measured value of the controlled variable is subtracted from the desired value or set point. The difference between these values is called the error.

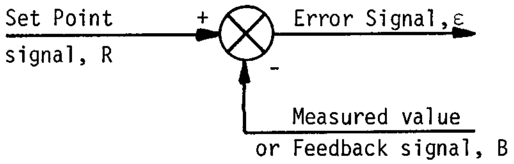
$$\varepsilon(t) = R(t) - B(t) \tag{9.14}$$

where ε = the error signal

R = deviation from its steady state value, of the desired value or set point signal

B = deviation from its steady state value, of the measured value or feedback signal.

Figure 9.3a shows the conventional representation of the comparator.

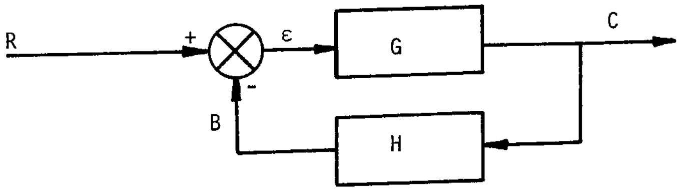


a. - The Comparator

In many cases, disturbance of the system does not arise by alteration of the set point (the servo problem). Instead it arises in some other way (the regulator problem); in such cases the value of R = 0 and equation 9.14 becomes:

$\varepsilon(t) = - B(t)$

a) In the Laplace transform method, the comparator does not appear as a separate function; it is incorporated within the algebra of the feedback loop, shown for example in Figure 9.3b.



b. The Negative Feedback Loop

Figure 9.3. Elements of the Block Diagram.

The Laplace transform for such a loop is written:

$$\frac{C(S)}{R(S)} = \frac{G}{1 + GH} \qquad (9.15)$$

Where G = product of transfer functions in the forward path;
H = product of transfer functions in the backward path.

b) Using the finite difference method, the comparator will be regarded as a separate part of the system, represented by equation 9.14.

2. The Controller

This acts upon the error signal so as to produce the control signal which goes to the final control element (e.g. the control valve, or other device).

The proportional - integral - derivative modes of the controller are represented by the following equation:

$$P = K_C\varepsilon + \frac{K_C}{\tau_I}\int \varepsilon\, dt + K_C\tau_D \frac{d\varepsilon}{dt} \qquad (9.16)$$

where P = deviation from its steady state value, of the output (control) signal;
K_C= controller gain constant (dimensionless);
τ_I= controller integral time constant (dimension of time);
τ_D= controller derivative time constant (dimension of time).

The values of K, τ_I and τ_D are selected and set by the operator. This may be done using the adjustment knobs within the controller, or alternatively by inputting values at the keyboard of the computer which replaces it (9).

a) The Laplace transform of equation 9.16 is:

$$P(s) = K_C\varepsilon(s) + \frac{K_C}{T} \cdot \frac{\varepsilon(s)}{s} + K_C\tau_D s\varepsilon(s)$$

which rearranged as a transfer function is:

$$\frac{P(s)}{\varepsilon(s)} = K_C \left(1 + \frac{1}{\tau_I s} + \tau_D s\right) \qquad (9.17)$$

b) Alternatively, equation 9.16 can be rewritten in finite difference notation as:

$$P = K_C\varepsilon + \frac{K_C}{\tau_I} \sum \varepsilon\Delta t + K_C\tau_D \frac{\Delta\varepsilon}{\Delta t}$$

$$= K_C\varepsilon_2 + \frac{K_C}{\tau_I} (\varepsilon_2 \Delta t + \sum\varepsilon\Delta t)$$

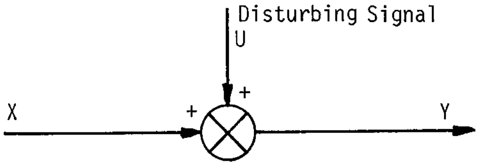
$$+ K_C\tau_D \frac{(\varepsilon_2 - \varepsilon_1)}{\Delta t}$$

Where $\varepsilon_2, \varepsilon_1$ represent values of ε at times t+Δt and t respectively.

OTHER ELEMENTS OF THE BLOCK DIAGRAM

The Summing Junction

This performs the arithmetic operation of addition, and is shown diagrammatically in Figure 9.4a.



a. The Summing Junction

The summing junction is conventionally used to represent the point of entry of a disturbance to the system.

In the example shown,

$$Y(t) = X(t) + U(t) \tag{9.17}$$

where U = the disturbing signal.

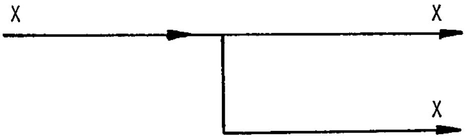
a) In the Laplace transform method, equation 9.17 is replaced by:

$$Y(s) = X(s) + U(s) \tag{9.18}$$

b) With the finite difference method, the summing junction will be represented by equation 9.17.

The Takeoff Point

This indicates a branch on the block diagram, the same signal now appearing at two points as shown in Figure 9.4b.



b. The Takeoff Point

Figure 9.4. Elements of the Block Diagram

REPRESENTATION OF THE DISTURBING FUNCTION

The four functions most commonly used are as follows:

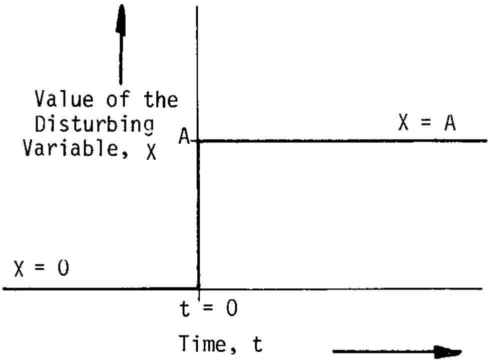
The Step Function

An instantaneous change in the value of the variable takes place. We normally assume this change to occur at time t = 0, in which case the function can be represented as:

$$\left.\begin{array}{ll} X = 0 & t \ <0 \\ X = A & + \ \geq 0 \end{array}\right\} \qquad (9.19)$$

where A = magnitude of the deviation in the value of X.

This function is shown graphically in Figure 9.5a.



a. The Step Function
Figure 9.5. Disturbing Functions.

a) The Laplace transform of the Step Function is

$$X(s) = \frac{A}{s} \qquad (9.20)$$

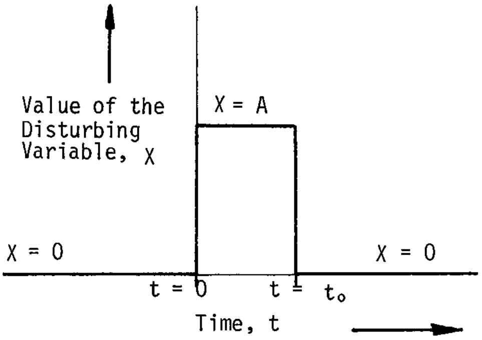
b) Using finite differences, the step function will be represented by equation 9.19.

The Pulse Function

An instantaneous change in the value of the variable occurs, followed later by another instantaneous change of equal magnitude, but opposite sign. If we assume the first change to occur at time t = 0, then the function can be represented as:

$$\left.\begin{array}{ll} X = 0 & t \ <0 \\ X = A & t \ \leq t_o \\ X = 0 & t \ >t_o \end{array}\right\} \qquad (9.21)$$

This function is shown graphically in Figure 9.5b.



b. The Pulse Function

Figure 9.5. Disturbing Functions.

A special case of the pulse function occurs if t_o approaches zero. If $t_o = 1/A$, then at large values of A, t_o approaches zero, but the area of the pulse on the diagram is equal to 1. This is referred to as a unit impulse.

a) The Laplace transform of the Pulse Function is:

$$X(s) = \frac{A}{s}(1 - e^{-t_o s}) \tag{9.22}$$

The Laplace transform of the Impulse Function is:

$$X(s) = k \tag{9.23}$$

where k = area of the function;
= $A*t_o$

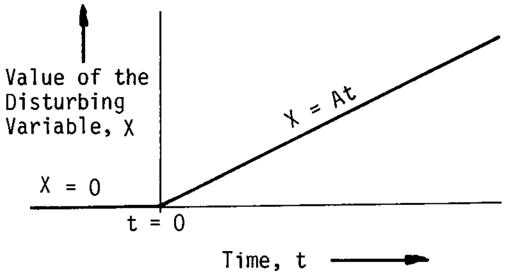
b) For the purpose of the finite difference method, equation 9.21 will be used; the value of t_o must be exactly divisible by Δt, and for an impulse, the value of $t_o = \Delta t$ should be chosen.

The Ramp Function

In this case the variable varies linearly with time. As before, this change is usually assumed to occur at time t = 0, in which case it can be represented as:

$$\left.\begin{array}{ll} X = 0 & t < 0 \\ X = At & t \geq 0 \end{array}\right\} \tag{9.24}$$

This function is shown graphically in Figure 9.5c.



c. The Ramp Function.

Figure 9.5. Disturbing Functions.

a) The Laplace transform of the Ramp Function is:

$$X(s) = \frac{A}{s^2} \tag{9.25}$$

b) The finite difference method will use equation 9.24.

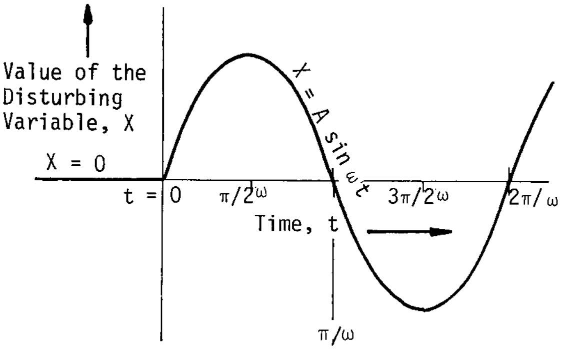
The Sine Function

This function, of very great importance because of its use with the Frequency Response method, can be represented as:

$$\left.\begin{array}{ll} X = 0 & \cdot\, t < 0 \\ X = A\sin\omega t & t \geq 0 \end{array}\right\} \tag{9.26}$$

where ω(omega) = radian frequency (radians/time).

This is shown graphically in Figure 9.5d.



d. The Sine Function.

Figure 9.5. Disturbing Functions.

a) The Laplace transform of the Sine Function is:

$$X(s) = \frac{\omega^2}{s^2 + \omega^2} \qquad (9.27)$$

b) The finite difference method will use equation 9.26.

EXAMPLE 9.1

A control valve is operated by an air-actuated diaphragm motor having an effective area of 6.5 x 10^{-2}m^2. The weight of the moving parts is 140kg; the stiffness of the spring is 10^5 N m^{-1}; the damping constant is 3 x 10^3 N sm^{-1}.

Determine the response of the valve to the following disturbances:

a) A step change in air pressure from 2.1 x 10^5 to 2.7 x 10^5 Nm^{-2};

b) A change in air pressure from 2.1 x 10^5 to 3.2 x 10^5 Nm^{-2}lasting for 0.2 seconds.

The differential equation for this case has already been given:

$$\frac{M}{H} \cdot \frac{d^2L}{dt^2} + \frac{C}{H} \cdot \frac{dL}{dt} + L = \frac{F(t)}{H} \qquad (9.8)$$

For this case, M = 140kg
H = 10^5Nm^{-1}
C = 3 $*$10 ^{3}NsM^{-1}

$$\therefore \frac{M}{H} = \frac{140}{10^5} = 1.4 * 10^{-3}s^2$$

$$\frac{C}{H} = \frac{3 * 10^3}{10^5} = 3 * 10^{-2}s$$

$$\frac{F(t)}{H} = \frac{6.5 * 10^{-2} * (2.7 - 2.1)10^5}{10^5} = 0.039\text{m}$$

Using the identity

$$\frac{\tau^2 d^2Y}{dt^2} + \frac{2\zeta\tau dY}{dt} + Y = X \qquad (9.9)$$

$$\tau^2 = \frac{M}{H} = 1.4 * 10^{-3} \text{ whence } \tau = 0.037S$$

$$2\zeta\tau = \frac{C}{H} = 3 * 10^{-2} \text{ whence } \zeta = \frac{3 \times 10^{-2}}{2 \times 0.037}$$

$$= 0.405$$

a) For the case of a step change,

$X = 0 \qquad t < 0$

$X = 0.039 \quad t \geq 0$

$\therefore X(s) = \frac{0.039}{s}$

The transfer function for the 2nd order case is:

$$\frac{Y(s)}{X(s)} = \frac{1}{\tau^2 s^2 + 2\zeta\tau s + 1} \tag{9.10}$$

$$\therefore Y(s) = \frac{0.039}{s} \cdot \frac{1}{1.4 * 10^{-3}s^2 + 3 * 10^{-2}s^2 + 1}$$

The transient response of the system may be obtained by transformation of this equation back into the time domain. Details of this procedure will not be given here, but may be obtained from specialist texts (1), (2), (3), (4), (8).

The result may be expressed in general terms as:

$$\text{If } Y(s) = \frac{A}{s(\tau^2 s^2 + 2\zeta\tau s + 1)}$$

then, for the case where $\zeta < 1$, the solution is

$$Y(t) = A\left[1 - \frac{1}{\sqrt{1-\zeta^2}} \cdot e^{-\zeta t/\tau} \cdot \sin\left(\sqrt{1-\zeta^2} \cdot \frac{t}{\tau} + \tan^{-1}\frac{\sqrt{1-\zeta^2}}{\zeta}\right)\right] \tag{9.28}$$

Using the values found above, that

$A = 0.039$

$\tau = 0.037$

$\zeta = 0.405$

we obtain the values listed in Table 9.1a and shown graphically on Figure 9.6 which demonstrate the oscillatory nature of the response.

b) As an approximation, we may regard this change in air pressure as an impulse of size:

$$\frac{6.5 * 10^{-2} * (3.2 - 2.1) * 10^5 \text{ x } 0.2}{10^5} = 1.43 * 10^{-2} \text{ ms}$$

The Laplace transform of this impulse is

$X(S) = 1.43 * 10^{-2}$

Consequently we require the solution to the equation

$Y(s) = \frac{1.43 * 10^{-2}}{\tau^2 s^2 + 2\zeta\tau s + 1}$ using the values of τ and ζ found above.

$$\text{If } \quad Y(s) = \frac{A}{\tau^2 s^2 + 2\zeta\tau s + 1}$$

$$\text{then } Y(t) = A\left[\frac{e^{-\zeta t/\tau}}{\tau\sqrt{1-\zeta^2}} \cdot \sin\left(\sqrt{1-\zeta^2} \cdot \frac{t}{\tau}\right)\right] \tag{9.29}$$

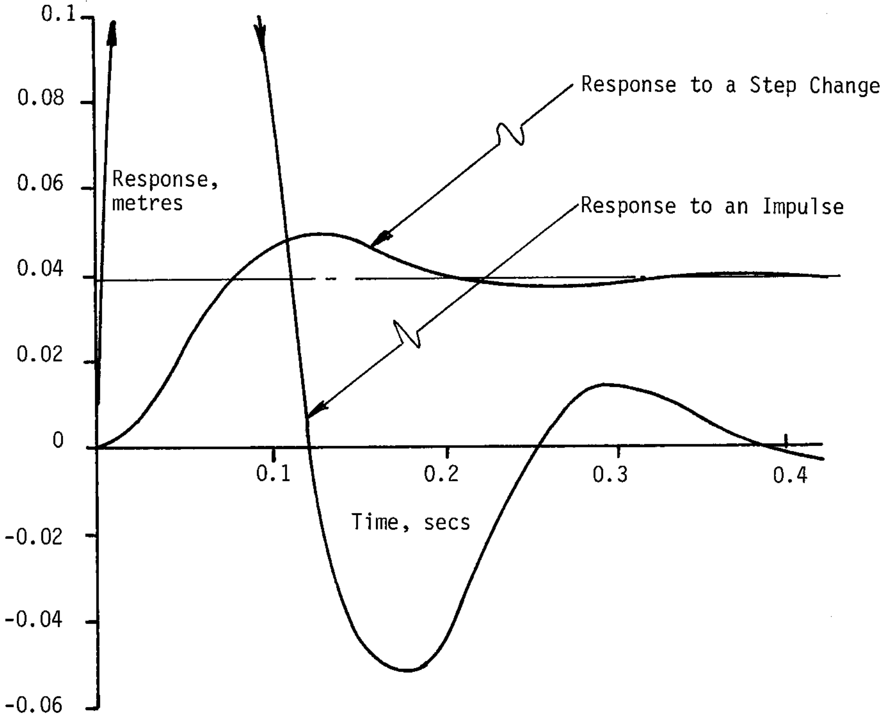


Figure 9.6. Response of a Pneumatic Control Valve to a change in Air Pressure.

Using the values found above, with A = 1.43 x 10^{-2} we obtain the values listed in Table 9.1b and shown in Figure 9.6.

It will be be seen that there is an error in the result. The response is shown to be decaying after 0.05 seconds although the duration of the pulse is 0.2 seconds. The error has of course arisen because, for simplicity, the pulse has been treated as an impulse.

TABLE 9.1

Response of Pneumatic Control Valve to Changes in Air Pressure

a) Step Change from 2.1 x 10^5 to 2.7 x 10^5N/m^2

Time Secs	Displacement metres
0	0
0.01	0.0013
0.05	0.0221
0.10	0.0456
0.15	0.0472
0.20	0.0399
0.25	0.0366
0.30	0.0378
0.35	0.0393

b) Impulse occasioned by a change in air pressure from 2.1×10^5 to 3.2×10^5 N/m^2 lasting for 0.2 seconds

Time secs	Displacement metres
0	0
0.01	+0.0927
0.05	+0.2309
0.10	+0.0879
C.15	-0.0438
0.20	-0.0461
0.25	-0.0029
0.30	+0.0143
0.35	+0.0064
0.40	-0.0024

EXAMPLE 9.2

The temperature of the contents of a jacketted kettle such as that shown in Figure 9.1 is measured by means of a thermocouple. The thermocouple signal is received by a controller, which sends a pneumatic signal to an air-actuated control valve. This valve controls the steam supply to the jacket. The transfer functions for the various elements are as follows:

G_V (valve) $\dfrac{6.4}{1 + 12s}$ $\dfrac{\text{kW}}{\text{kPa}}$

G (kettle) $\dfrac{0.085}{1 + 300s}$ $\dfrac{^\circ\text{C}}{\text{kW}}$

H (thermocouple) 0.038 $\dfrac{\text{mV}}{^\circ\text{C}}$

Whilst the plant is operating at steady state, the line carrying air to the pneumatic actuator is damaged, resulting in a sudden drop in signal pressure. This may be modelled as a step change of magnitude - 6kPa.

What will be the effect on the temperature of the vessel contents? Solve this problem by the Laplace transform technique, assuming a proportional controller G_C, having a gain constant of 350 kPa/mV.

The Block Diagram for this system is shown in Figure 9.7.

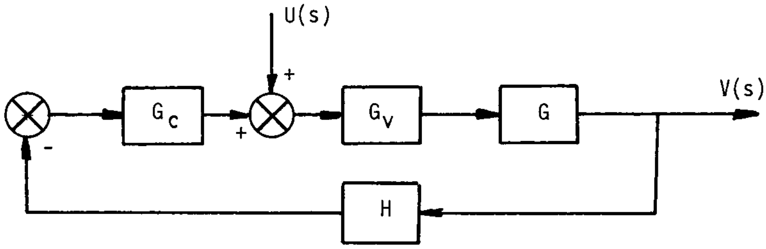


Figure 9.7. Block diagram for Example 9.2.

By the rules of block diagram reduction, the transfer function for the entire process is:

$$\frac{V(s)}{U(s)} = \frac{G_V G}{1 + G_V G H G_C}$$

The Laplace transform of the step change U(t) is:

$$U(s) = -\frac{6}{s}$$

Hence, $V(s) = \dfrac{-6\ G_V G}{s(1 + G_V G H G_C)} =$

$$-\frac{6}{s} \cdot \frac{\dfrac{6.4}{(1+12s)} * \dfrac{0.085}{(1+300s)}}{1 + \dfrac{6.4}{1+12s} * \dfrac{0.085}{1+300s} * 0.038 * 350} =$$

$$\frac{-3.264}{s\left[(1+12s)(1+300s) + 7.235\right]} = \frac{-3.264}{s\left[1 + 312s + 3600s^2 + 7.235\right]}$$

Or, rewritten in standard quadratic form:

$$V(s) = \frac{-0.396}{s(437.1s^2 + 37.89s + 1)}$$

The ultimate time response of the system is easily determined from this equation by applying the Final Value Theorem. This theorem may be stated mathematically as:

$$\lim_{S \to 0} sV(s) = \lim_{t \to \infty} V(t)$$

Multiplying the expression for V(s) by s, and then setting $s=0$, we obtain:

$$\underset{t \to \infty}{V(t)} = -0.396$$

The transient response is obtained using Equation 9.28 and results are tabulated in Table 9.2 and shown graphically in Figure 9.8. We see that there is an oscillatory response, and that after about 300 seconds the final response is obtained, the temperature of the kettle being low by -0.396°C.

TABLE 9.2

Deviation of kettle temperature due to a step change in signal pressure of -6kPa

t, Time from commencement of disturbance Seconds	V(t) deviation of bath temperature ^{o}C
0	0
50	-0.26690
100	-0.37945
150	-0.39487
200	-0.39603
250	-0.39602
300	-0.39600

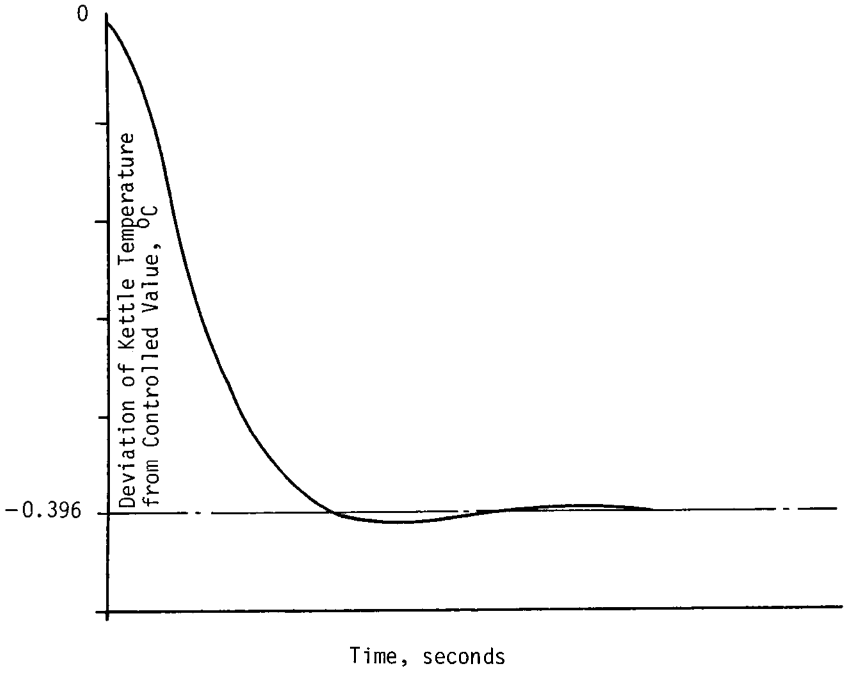


Figure 9.8. Response of kettle temperature to an Air Leak.

EXAMPLE 9.3

Using the Laplace transform method, rework Example 9.2 this time assuming the controller also has integral and derivative functions.

a) If the values of derivative and integral time constants are 10 seconds and 25 seconds respectively, what is the effect of the disturbance?

b) At the given values of K_C and τ_D, can an incorrect choice of τ_I cause instability?

a) Referring to the solution of Example 9.2,

$$V(S) = -\frac{6\ G_VG}{s(1 + G_V\ HG_C)}$$

In this case $G_C = K_C + K_C\tau_D s + K_C/\ s\tau_I \quad =$

$$350 + 3500s + 350/25s$$

Hence,

$$V(s) = -\frac{6 * \frac{6.4}{1 + 12s} * \frac{0.085}{1 + 300s}}{S\left[1 + \frac{6.4}{1 + 12s} * \frac{0.085}{1 + 300s} * 0.038\ (350 + 3500s + \frac{350}{25s})\right]} =$$

$$\frac{-3.264}{3600s^3 + 384.4s^2 + 8.235s + 0.289}$$

Solution of this expression involves factorisation of the denominator, separation into partial fractions, and transformation of the individual terms back into the time domain.

This arduous procedure will not be given here; however the ultimate response of the system is easily determined by application of the Final Value Theorem. In this case,

$$\lim_{s \to 0} sV(s) \quad = \quad \underset{t \to \infty}{V(t)}$$

$$= \quad \underset{s \to 0}{\frac{-3.264}{3600s^2 + 384.4s + 8.235 + \frac{0.289}{s}}} = 0$$

b) A complete exploration of the effects of varying the controller function is best carried out by the Root Locus method. This involves factorisation of the numerator and denominator of the transform, plotting the poles and zeros thus obtained, and investigating graphically, the behaviour of the roots as a chosen parameter is varied.

The procedure is lengthy, but one portion of the method may be quickly applied to the present problem. This is the Routh Test (1), (2), (3), (8).

Routh Test

This is carried out upon the denominator of the system transfer function. The denominator equated to zero, is called the Characteristic Equation. In this case,

$$1 + G_V G H G_C = 0$$

The controller function we wish to explore is the value of integral time, τ_I. The characteristic equation may therefore be written:

$$(1 + 12s)(1 + 300s) + \frac{2.067}{10^2}\left(350 + 3500s + \frac{350}{\tau_I s}\right) = 0$$

Expanding the terms and multiplying throughout by s to eliminate the term 1/s, we obtain:

$$3600s^3 + 384.4s^2 + 8.235s + \frac{7.235}{\tau_I} = 0$$

The coefficients of s are then arranged to form the first two rows of the Routh Array. The lower rows are then formed by multiplying the appropriate terms together as shown in Table 9.3.

TABLE 9.3

The Routh Array - Example 9.3

Row			
s^3	3600	8.235	0
s^2	384.4	$\frac{7.235}{\tau}$	0
s^1	$8.235 - \frac{67.76}{\tau}$	0	0
s^0	$7.235/\tau$	0	0

$$s^1 = \frac{384.4 * 8.235 - 3600 * 7.235/\tau_I}{384.4}$$

$$= 8.235 - \frac{67.76}{\tau_I}$$

$$s^0 = \frac{\left(8.235 - \frac{67.76}{\tau_I}\right) * \frac{7.235}{\tau_I} - 0}{\left(8.235 - \frac{67.76}{\tau_I}\right)}$$

$$= \frac{7.235}{\tau_I}$$

The system is unstable if a negative value appears. In this case, the first term in the s' row will be negative if

$$8.235 - \frac{67.76}{\tau_1} < 0 \text{ i.e. if } \tau_1 < \frac{67.76}{8.235} = 8.23$$

τ_1 = 8.23 is a critical value for marginal stability of the system.

COMPUTER SOLUTION OF CONTROL PROBLEMS BY FINITE DIFFERENCES

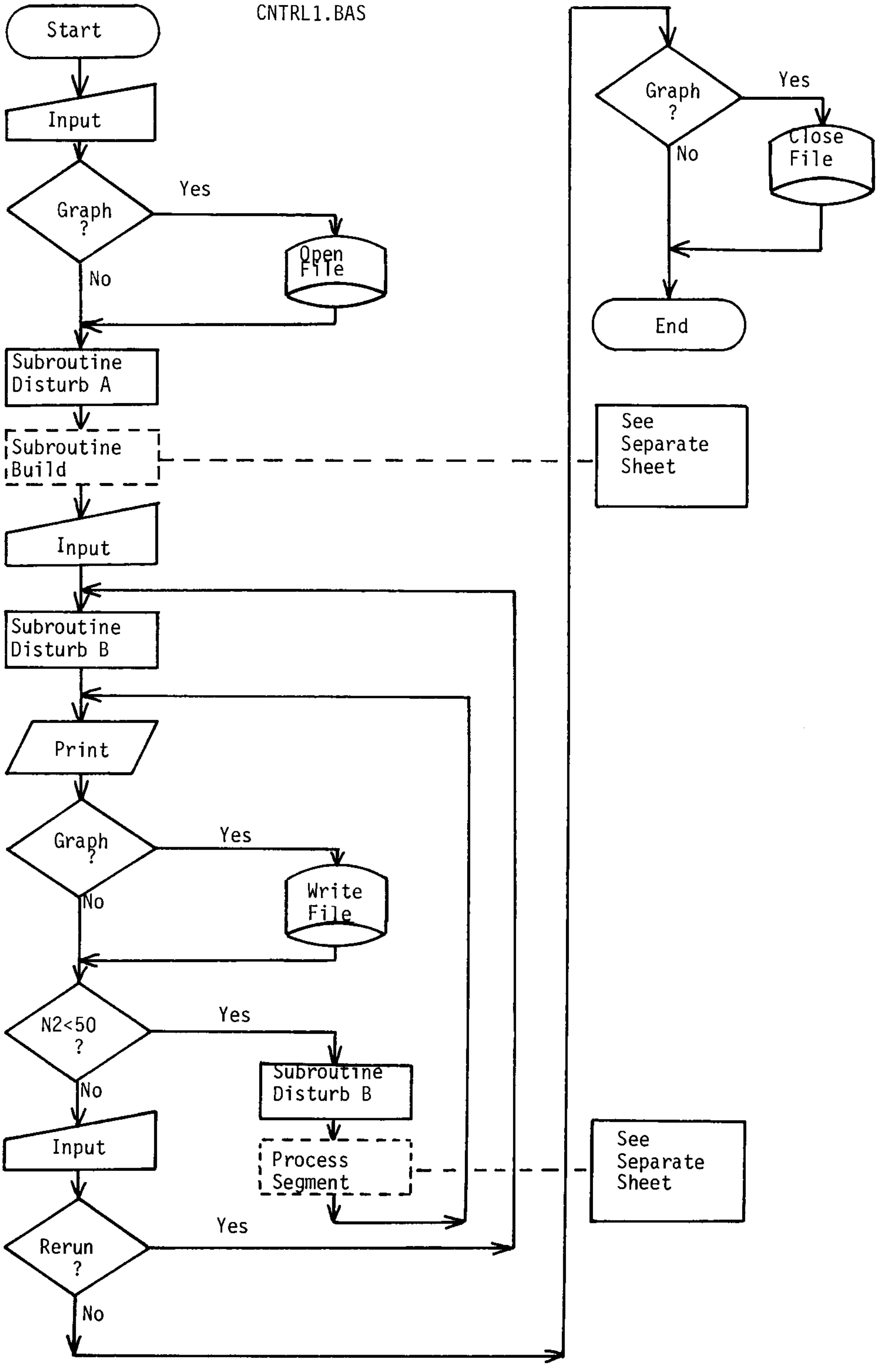
Program CNTRL1 utilises the finite difference equations developed earlier in the chapter. The structure of the program is generally similar to that developed for Flowsheeting (chapter 1).

Since it is convenient to have the response data in graphical form, the program provides means for this. Output data are stored on a disc file; the program Graph 1 which follows can be used subsequently to retrieve this data and produce a graphical plot.

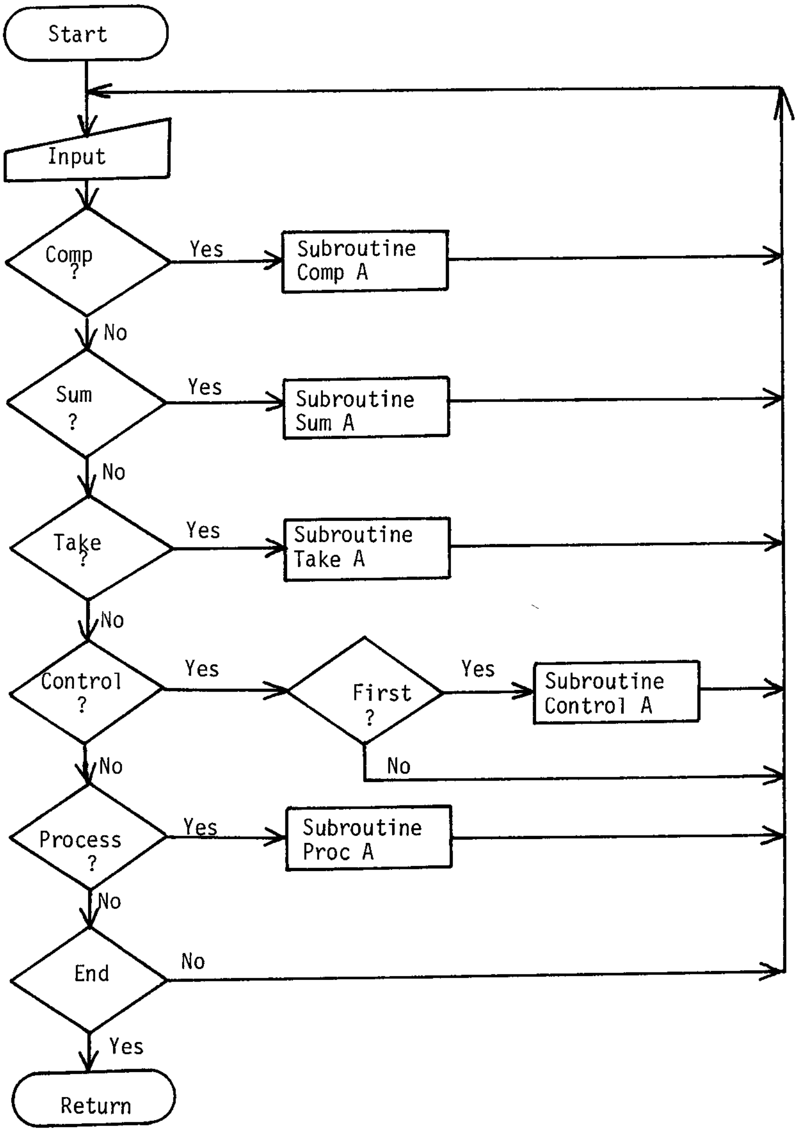
To use the latter program it is necessary for your computer to be provided with a graphics card, and of course the supporting software.

The program is short and simple but employs functions not seen in the preceeding programs.

CNTRL1.BAS

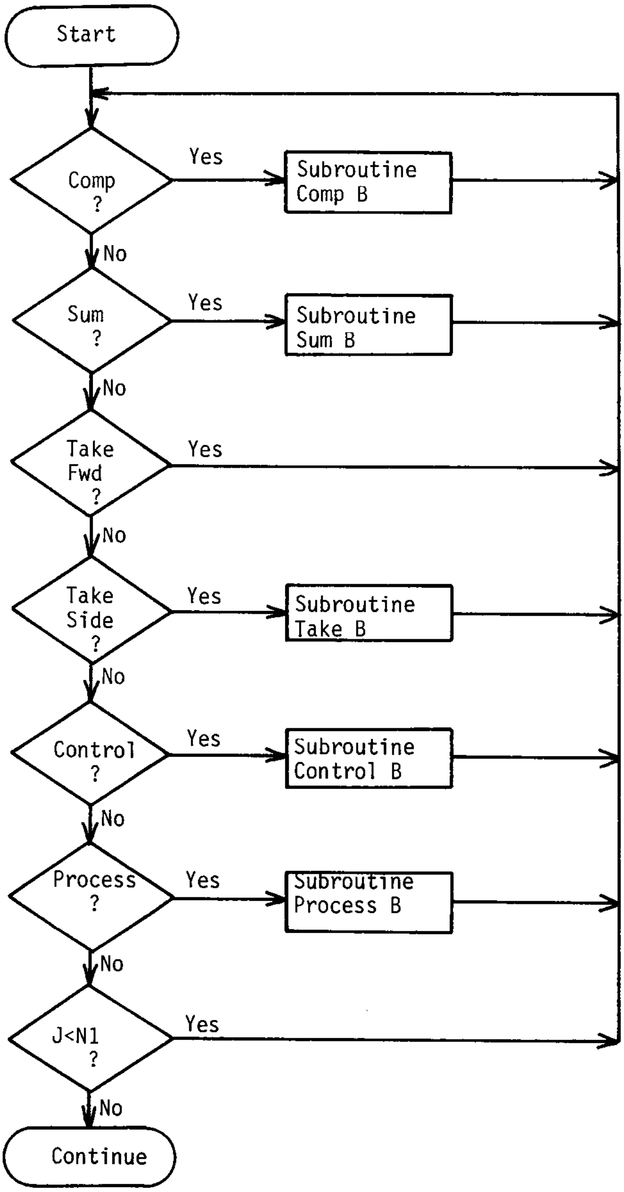

Start
Input
Graph ?
Yes
No
Open File
Subroutine Disturb A
Subroutine Build
See Separate Sheet
Input
Subroutine Disturb B
Print
Graph ?
Yes
No
Write File
N2<50 ?
Yes
No
Subroutine Disturb B
Process Segment
See Separate Sheet
Input
Rerun ?
Yes
No
Graph ?
Yes
No
Close File
End

Subroutine BUILD - CNTRL1.BAS



Process Segment

CNTRL1.BAS



```
10  REM ***************************************************
20  REM - PROGRAM CNTRL1.BAS THIS PROGRAM SIMULATES
30  REM - OPEN AND CLOSED LOOP SYSTEMS USING
40  REM - A FINITE DIFFERENCE METHOD
50  REM - PROGRAM NOMENCLATURE
60  REM - A$,B$ -  Alphanumeric inputs in response to
70  REM            computer queries
80  REM - A$(J) -  Names of the items on the Block Diagram
90  REM - B(J,K)-  Where K=1, this is the item from which
100 REM            comes the forward signal entering item J
110 REM            Where K=2, this is the item number from
120 REM            which comes the backward or side signal
130 REM - C(J) -   Damping coefficient, 2nd order function
140 REM            item J
150 REM - C1   -   When value =1, prevents programming of a
160 REM            second controller
170 REM - E1   -   Value of integral of controller
180 REM            error signal
190 REM - K(J) -   Value of gain constant, item J
200 REM - M1   -   Magnitude of step or impulse, slope of
210 REM            ramp, amplitude of sine wave
220 REM - M2   -   Duration of impulse, radian frequency of
230 REM            sine wave
240 REM - N1   -   Number of items on the block diagram
250 REM            (where an item can be a block, summing
260 REM            junction, comparator or takeoff point)
270 REM - N2   -   Number of data printouts; program
280 REM            terminates at N2=50
290 REM - N3   -   Item number from which response signal
300 REM            comes
310 REM - N6   -   Duplicates final value of N1
320 REM - O(J) -   Indicator of first or second order for
330 REM            process at item J
340 REM - R(J) -   Value of process time constant, item J
350 REM - S(J) -   Value of the signal leaving item J at
360 REM            time t+dt
370 REM - T(J) -   Value of the signal leaving item J
380 REM            at time t
390 REM - T1   -   Value of the time interval dt,secs
400 REM - T2   -   Controller derivative time, secs
410 REM - T3   -   Controller integral time, secs
420 REM - T4   -   Total elapsed time, secs
430 REM - T5   -   Time interval required between
440 REM            printouts, secs
450 REM - T6   -   Time interval since last printout
460 REM - T7,T8    Intermediate values in calculation
470 REM            of response of second order function
480 REM - U(J) -   Value of the signal leaving item J
490 REM            at time t-dt
500 REM - PROGRAM DESCRIPTION:
510 REM - THE CONTROL PROGRAM:   LINES 2000 - 3030
520 REM - LINES 2000 - 2210   Necessary array declarations
530 REM - are made (line 2000) then the opportunity is
540 REM - given to file the output, and to select the disc
550 REM - to be used for storage.  Next the subroutines
560 REM - "Disturb" and "Build" are called (lines 2200 &
570 REM - 2210).   These enable the designer to enter from
580 REM - the keyboard those parameters concerned with the
590 REM - disturbing signal and the block diagram
600 REM - respectively
610 REM - LINES 2220 - 2290   The finite difference time
```

```
620 REM - interval (T1), and printout details are entered
630 REM - LINES 2300 - 2680  By calling up appropriate
640 REM - subroutines as needed, the required disturbing
650 REM - signal is generated (line 2340).  The response
660 REM - is then calculated for each element of the block
670 REM - diagram in turn (lines 2440 - 2610)  Values of
680 REM - these responses are stored in Matrix S.  Before
690 REM - undertaking another iteration, the set of previous
700 REM - responses stored in Matrix T, is transferred to
710 REM - Matrix U and the set of values in Matrix S is now
720 REM - transferred to Matrix T (lines 2620 - 2650).  In
730 REM - this way, three successive sets of values of
740 REM - system element responses, are available.  These
750 REM - are required for example when dealing with a 2nd
760 REM - order function.  A check is then made (lines 2660
770 REM - to 2680) as to whether or not a data printout is
780 REM - required, and the program then returns either to
790 REM - line 2350 (printout), or to line 2410 (no
800 REM - printout).  A running check is also kept on the
810 REM - number of data printouts; when
820 REM - this reaches the arbitrary total of 50,
830 REM - iteration ceases (lines 2380 & 2390)
840 REM - LINES 2690 - 3030  In this segment are contained
850 REM - the options to change parameters.  Variables are
860 REM - first set to zero (lines 2750 - 2790)
870 REM - Parameters are then varied as follows:
880 REM -  Finite difference time, & time
890 REM -  between printouts (lines 2810 - 2850)
900 REM -  Disturbing signal (lines 2860 - 2900)
910 REM -  Controller settings (lines 2910 - 2940)
920 REM - DETAILS OF DISTURB SUBROUTINE
930 REM - LINES 3050 - 3390 (Disturb Part A)  This part of
940 REM - the subroutine is used to establish the desired
950 REM - characteristics of the disturbing signal
960 REM - LINES 3410 - 3560 (Disturb Part B)  The required
970 REM - disturbing signal is simulated using the
980 REM - appropriate equation from among those quoted
990 REM - earlier in Chapter 9.  This part is called at
1000 REM - each iteration of the control program
1010 REM - DETAILS OF SUBROUTINE BUILD
1020 REM - LINES 3590 - 3920  Each numbered element of the
1030 REM - block diagram is taken up in turn during the
1040 REM - operation of this subroutine.  According to the
1050 REM - nature of the element (comparator,summing
1060 REM - junction, etc), an appropriate subroutine
1070 REM - is then called.  Build subroutine is called
1080 REM - only once during the running of the program
1090 REM - DETAILS OF SUBROUTINES COMP, SUM, TAKE, CONTROL
1100 REM - AND PROCESS.
1110 REM - LINES 3940 - 4980  Each of these subroutines is
1120 REM - similarly constructed.  Part A of each
1130 REM - subroutine is used to ascribe the appropriate
1140 REM - name to the corresponding item
1150 REM - on the block diagram; and to
1160 REM - establish the number of adjacent items,
1170 REM - from which signals come.  Additionally, with
1180 REM - subroutines Control and Process, appropriate
1190 REM - parameters are entered from the keyboard.
1200 REM - With the exception of Control subroutine, Part A
1210 REM - of each of the subroutines may be called
1220 REM - only once during the running of the program
```

```
1230 REM - In part B of each subroutine is carried out the
1240 REM - calculation of the response appropriate to the
1250 REM - item, using the equations developed earlier in
1260 REM - Chapter 9.   These parts of the subroutines are
1270 REM - called when necessary at each iteration of the
1280 REM - control program.
1290 REM ****************************************************
2000 DIM A$(20),S(20),T(20),U(20)
2010 PRINT "DO YOU WANT TO FILE OUTPUT FOR GRAPHS?  "
2020 INPUT "TYPE Y OR N";B$
2030 IF B$="Y" THEN 2060
2040 IF B$="N" THEN 2200
2050 GOTO 2010
2060 PRINT "ON WHICH DISC DO YOU WANT FILES TO RESIDE"
2070 INPUT "TYPE A,B,C,OR D";A$
2080 IF A$="A" THEN 2130
2090 IF A$="B" THEN 2150
2100 IF A$="C" THEN 2170
2110 IF A$="D" THEN 2190
2120 GOTO 2060
2130 OPEN "A:DATA" FOR OUTPUT AS #1
2140 GOTO 2200
2150 OPEN "B:DATA" FOR OUTPUT AS #1
2160 GOTO 2200
2170 OPEN "C:DATA" FOR OUTPUT AS #1
2180 GOTO 2200
2190 OPEN "D:DATA" FOR OUTPUT AS #1
2200 GOSUB 3050
2210 GOSUB 3580
2220 PRINT
2230 PRINT "DELTA T, SECONDS";
2240 INPUT T1
2250 PRINT "ITEM  NUMBER FROM WHICH THE REQUIRED OUTPUT"
2260 INPUT "SIGNAL COMES";N3
2270 PRINT "REQUIRED TIME INTERVAL BETWEEN PRINTOUTS,"
2280 INPUT "SECONDS";T5
2290 T5=T5/T1
2300 E1=0
2310 PRINT
2320 PRINT "   TIME     DISTURBING   RESPONSE"
2330 PRINT "   SECS      SIGNAL"
2340 GOSUB 3410
2350 PRINT USING "####.###   ";T4,S(1),S(N3)
2360 IF B$="N" THEN 2380
2370 WRITE #1,T4,S(1),S(N3)
2380 N2=N2+1
2390 IF N2=50 THEN 2690
2400 T6=1
2410 T4=T4+T1
2420 T6=T6+1
2430 GOSUB 3410
2440 FOR J=2 TO N1
2450 IF A$(J)="COMP" THEN 2520
2460 IF A$(J)="SUM" THEN 2540
2470 IF A$(J)="TAKE,FORWARD" THEN 2610
2480 IF A$(J)="TAKE,SIDE" THEN 2560
2490 IF A$(J)="CON" THEN 2580
2500 IF A$(J)="PROC" THEN 2600
2510 GOTO 2610
2520 GOSUB 4060
2530 GOTO 2610
```

```
2540 GOSUB 4250
2550 GOTO 2610
2560 GOSUB 4400
2570 GOTO 2610
2580 GOSUB 4620
2590 GOTO 2610
2600 GOSUB 4890
2610 NEXT J
2620 FOR I=1 TO N1
2630 U(I)=T(I)
2640 T(I)=S(I)
2650 NEXT I
2660 IF T6<T5 THEN 2410
2670 IF T6=T5 THEN 2410
2680 GOTO 2350
2690 PRINT "INPUT Y TO RERUN";
2700 INPUT A$
2710 IF A$="Y" THEN 2730
2720 GOTO 5010
2730 N2=0
2740 E1=0
2750 FOR I=1 TO N1
2760 S(I)=0
2770 T(I)=0
2780 U(I)=0
2790 NEXT I
2800 T4=0
2810 PRINT "DELTA T,SECS";
2820 INPUT T1
2830 PRINT "REQUIRED TIME INTERVAL BETWEEN PRINTOUTS,SECS";
2840 INPUT T5
2850 T5=T5/T1
2860 PRINT "INPUT Y TO CHANGE DISTURBING SIGNAL";
2870 INPUT A$
2880 IF A$="Y" THEN 2900
2890 GOTO 2910
2900 GOSUB 3090
2910 PRINT "INPUT Y TO CHANGE CONTROLLER SETTINGS";
2920 INPUT A$
2930 IF A$="Y" THEN 2950
2940 GOTO 2320
2950 PRINT "INPUT ITEM NUMBER OF CONTROLLER";
2960 N6=N1
2970 INPUT N1
2980 IF A$(N1)="CON" THEN 3010
2990 PRINT "INCORRECT CONTROLLER NUMBER"
3000 GOTO 2910
3010 GOSUB 4530
3020 N1=N6
3030 GOTO 2320
3040 REM ***************************************************
3050 REM - DISTURB SUBROUTINE USED TO GENERATE
3060 REM - THE DISTURBING SIGNAL
3070 REM - DISTURB PART A **********************************
3080 N1=1
3090 PRINT "THIS IS DISTURB SUBROUTINE ITEM";N1
3100 PRINT "ENTER STEP,RAMP,PULSE OR SINE";
3110 INPUT A$
3120 IF A$<>"STEP" THEN 3170
3130 A$(1)="STEP"
3140 PRINT "MAGNITUDE OF STEP";
```

```
3150 INPUT M1
3160 GOTO 3390
3170 IF A$<>"RAMP" THEN 3220
3180 A$(1)="RAMP"
3190 PRINT "SLOPE OF RAMP";
3200 INPUT M1
3210 GOTO 3390
3220 IF A$<>"PULSE" THEN 3240
3230 GOTO 3250
3240 IF A$<>"PUL" THEN 3310
3250 A$(1)="PULSE"
3260 PRINT "MAGNITUDE OF PULSE";
3270 INPUT M1
3280 PRINT "DURATION OF PULSE, SECONDS";
3290 INPUT M2
3300 GOTO 3390
3310 IF A$="SINUSOID" THEN 3350
3320 IF A$="SIN" THEN 3350
3330 PRINT "INPUT NOT RECOGNISED"
3340 GOTO 3100
3350 PRINT "AMPLITUDE OF SINE WAVE";
3360 INPUT M1
3370 PRINT "RADIAN FREQUENCY";
3380 INPUT M2
3390 RETURN
3400 REM ***************************************************
3410 REM - DISTURB PART B **********************************
3420 IF A$(1)="STEP" THEN 3460
3430 IF A$(1)="RAMP" THEN 3480
3440 IF A$(1)="PULSE" THEN 3500
3450 GOTO 3550
3460 S(1)=M1
3470 GOTO 3560
3480 S(1)=M1*T4
3490 GOTO 3560
3500 IF (T4-T1)>M2 THEN 3530
3510 S(1)=M1
3520 GOTO 3560
3530 S(1)=0
3540 GOTO 3560
3550 S(1)=M1*SIN(M2*T4)
3560 RETURN
3570 REM ***************************************************
3580 REM - BUILD SUBROUTINE USED WHEN ASSEMBLING THE
3590 REM - BLOCK DIAGRAM - BUILD PART A ********************
3600 PRINT
3610 PRINT "BUILD SUBROUTINE.   ENTER COMPARATOR,"
3620 PRINT "SUMMING JUNCTION,TAKEOFF POINT, CONTROLLER,"
3630 INPUT "PROCESS OR END";A$
3640 IF A$="COMPARATOR" THEN 3790
3650 IF A$="COMP" THEN 3790
3660 IF A$="SUMMING JUNCTION" THEN 3810
3670 IF A$="SUM" THEN 3810
3680 IF A$="TAKEOFF POINT" THEN 3830
3690 IF A$="TAKE" THEN 3830
3700 IF A$="TAKEOFF" THEN 3830
3710 IF A$="CONTROLLER" THEN 3850
3720 IF A$="CONTROL" THEN 3850
3730 IF A$="CON" THEN 3850
3740 IF A$="PROCESS" THEN 3900
3750 IF A$="PROC" THEN 3900
```

```
3760 IF A$="END" THEN 3920
3770 PRINT "INPUT NOT RECOGNISED"
3780 GOTO 3600
3790 GOSUB 3940
3800 GOTO 3600
3810 GOSUB 4130
3820 GOTO 3600
3830 GOSUB 4290
3840 GOTO 3600
3850 IF C1=0 THEN 3880
3860 PRINT "PROGRAM CAN ONLY HANDLE ONE CONTROLLER"
3870 GOTO 3600
3880 GOSUB 4450
3890 GOTO 3600
3900 GOSUB 4680
3910 GOTO 3600
3920 RETURN
3930 REM ***************************************************
3940 REM - COMP SUBROUTINE, SIMULATES THE COMPARATOR
3950 REM - COMP PART A *************************************
3960 N1=N1+1
3970 PRINT "COMP SUBROUTINE, ITEM NUMBER";N1;" THIS NUMBER"
3980 PRINT "ALSO DENOTES THE OUTPUT SIGNAL"
3990 A$(N1)="COMP"
4000 PRINT "ITEM NUMBER FROM WHICH FEED FORWARD "
4010 INPUT "SIGNAL COMES, IF NO SIGNAL INPUT ZERO";B(N1,1)
4020 PRINT "ITEM NUMBER FROM WHICH FEED BACK SIGNAL COMES";
4030 INPUT B(N1,2)
4040 RETURN
4050 REM ***************************************************
4060 REM - COMP PART B *************************************
4070 IF B(J,1)=0 THEN 4100
4080 S(J)=S(B(J,1))-S(B(J,2))
4090 GOTO 4110
4100 S(J)=-S(B(J,2))
4110 RETURN
4120 REM ***************************************************
4130 REM - SUM SUBROUTINE, SIMULATES A SUMMING JUNCTION
4140 REM - SUM PART A **************************************
4150 N1=N1+1
4160 PRINT "SUM SUBROUTINE, ITEM NUMBER";N1;"THIS NUMBER"
4170 PRINT "ALSO DENOTES THE OUTPUT SIGNAL"
4180 A$(N1)="SUM"
4190 PRINT "ITEM NUMBER FROM WHICH FORWARD SIGNAL COMES";
4200 INPUT B(N1,1)
4210 PRINT "ITEM NUMBER FROM WHICH SIDE SIGNAL COMES";
4220 INPUT B(N1,2)
4230 RETURN
4240 REM ***************************************************
4250 REM - SUM PART B **************************************
4260 S(J)=S(B(J,1))+S(B(J,2))
4270 RETURN
4280 REM ***************************************************
4290 REM - TAKE SUBROUTINE, SIMULATES A TAKEOFF POINT'
4300 REM - TAKE PART A *************************************
4310 N1=N1+2
4320 PRINT "TAKE SUBROUTINE ITEM NUMBER";N1-1;"THIS NUMBER"
4330 PRINT "ALSO DESIGNATES THE FORWARD SIGNAL."
4340 PRINT "SIDE SIGNAL IS DENOTED BY NUMBER";N1
4350 A$(N1-1)="TAKE,FORWARD"
4360 A$(N1)="TAKE,SIDE"
```

```
4370 PRINT "ITEM NUMBER FROM WHICH SIGNAL COMES";
4380 INPUT B(N1,1)
4390 RETURN
4400 REM - TAKE PART B
4410 S(J-1)=S(B(J,1))
4420 S(J)=S(B(J,1))
4430 RETURN
4440 REM ***************************************************
4450 REM - CONTROL SUBROUTINE, SIMULATES THE CONTROLLER
4460 REM - CONTROL PART A **********************************
4470 N1=N1+1
4480 PRINT "CONTROL SUBROUTINE ITEM NUMBER";N1;"THIS NUMBER"
4490 PRINT "ALSO DESIGNATES THE OUTPUT SIGNAL"
4500 A$(N1)="CON"
4510 PRINT "ITEM NUMBER FROM WHICH SIGNAL COMES";
4520 INPUT B(N1,1)
4530 PRINT "VALUE OF GAIN CONSTANT";
4540 INPUT K(N1)
4550 PRINT "VALUE OF DERIVATIVE TIME, SECONDS";
4560 INPUT T2
4570 PRINT "VALUE OF INTEGRAL TIME, SECONDS";
4580 INPUT T3
4590 C1=1
4600 RETURN
4610 REM ***************************************************
4620 REM - CONTROL PART B **********************************
4630 E1=E1+T1*S(B(J,1))
4640 S(J)=K(J)*S(B(J,1))
4650 S(J)=S(J)+K(J)*T2*(S(B(J,1))-T(B(J,1)))/T1+K(J)*E1/T3
4660 RETURN
4670 REM ***************************************************
4680 REM - PROCESS SUBROUTINE, SIMULATES ANY FIRST OR SECOND
4690 REM - ORDER FUNCTION - PROCESS PART A *****************
4700 N1=N1+1
4710 PRINT "PROCESS SUBROUTINE ITEM NUMBER";N1
4720 A$(N1)="PROC"
4730 PRINT "ITEM NUMBER FROM WHICH SIGNAL COMES";
4740 INPUT B(N1,1)
4750 PRINT "VALUE OF TIME CONSTANT";
4760 INPUT R(N1)
4770 PRINT "INPUT 1 OR 2 FOR FIRST OR SECOND ORDER";
4780 INPUT O(N1)
4790 IF O(N1)=1 THEN 4820
4800 IF O(N1)=2 THEN 4850
4810 GOTO 4770
4820 PRINT "VALUE OF GAIN CONSTANT";
4830 INPUT K(N1)
4840 GOTO 4870
4850 PRINT "VALUE OF DAMPING COEFFICIENT";
4860 INPUT C(N1)
4870 RETURN
4880 REM ***************************************************
4890 REM - PROCESS PART B **********************************
4900 IF O(J)=2 THEN 4950
4910 REM - FIRST ORDER FUNCTION ****************************
4920 S(J)=(S(B(J,1))*K(J)+T(J)*R(J)/T1)/(1+R(J)/T1)
4930 GOTO 4980
4940 REM - SECOND ORDER FUNCTION ***************************
4950 T7=R(J)/T1
4960 T8=2*T7*C(J)
4970 S(J)=(S(B(J,1))+(2*T(J)-U(J))*T7^2+T(J)*T8)/(T7^2+T8+1)
```

```
4980 RETURN
4990 REM**************************************************
5000 IF B$="N" THEN 5020
5010 CLOSE #1
5020 END
```

EXAMPLE 9.4 COMPUTER SOLUTION OF EXAMPLE 9.1

```
LOAD"A:CNTRL1
Ok
RUN
DO YOU WANT TO FILE OUTPUT FOR GRAPHS?
TYPE Y OR N? Y
ON WHICH DISC DO YOU WANT FILES TO RESIDE
TYPE A,B,C,OR D? A
THIS IS DISTURB SUBROUTINE ITEM 1
ENTER STEP,RAMP,PULSE OR SINE? STEP
MAGNITUDE OF STEP? .039

BUILD SUBROUTINE.  ENTER COMPARATOR,
SUMMING JUNCTION,TAKEOFF POINT, CONTROLLER,
PROCESS OR END? PROC
PROCESS SUBROUTINE ITEM NUMBER 2
ITEM NUMBER FROM WHICH SIGNAL COMES? 1
VALUE OF TIME CONSTANT? .037
INPUT 1 OR 2 FOR FIRST OR SECOND ORDER? 2
VALUE OF DAMPING COEFFICIENT? .405

BUILD SUBROUTINE.  ENTER COMPARATOR,
SUMMING JUNCTION,TAKEOFF POINT, CONTROLLER,
PROCESS OR END? END

DELTA T, SECONDS? .01
ITEM  NUMBER FROM WHICH THE REQUIRED OUTPUT
SIGNAL COMES? 2
REQUIRED TIME INTERVAL BETWEEN PRINTOUTS,
SECONDS? .01

   TIME     DISTURBING   RESPONSE
   SECS       SIGNAL
   0.000      0.039      0.000
   0.010      0.039      0.002
   0.020      0.039      0.006        0.170      0.039      0.044
   0.030      0.039      0.011        0.180      0.039      0.043
   0.040      0.039      0.016        0.190      0.039      0.042
   0.050      0.039      0.022        0.200      0.039      0.041
   0.060      0.039      0.027        0.210      0.039      0.040
   0.070      0.039      0.031        0.220      0.039      0.040
   0.080      0.039      0.035        0.230      0.039      0.039
   0.090      0.039      0.039        0.240      0.039      0.039
   0.100      0.039      0.041        0.250      0.039      0.038
   0.110      0.039      0.043        0.260      0.039      0.038
   0.120      0.039      0.044        0.270      0.039      0.038
   0.130      0.039      0.045        0.280      0.039      0.038
   0.140      0.039      0.045
   0.150      0.039      0.045
   0.160      0.039      0.044
   0.290        0.039        0.038
   0.300        0.039        0.038
   0.310        0.039        0.038
   0.320        0.039        0.038
   0.330        0.039        0.038
   0.340        0.039        0.039
   0.350        0.039        0.039
   0.360        0.039        0.039
   0.370        0.039        0.039
   0.380        0.039        0.039
   0.390        0.039        0.039
   0.400        0.039        0.039
   0.410        0.039        0.039
   0.420        0.039        0.039
   0.430        0.039        0.039
   0.440        0.039        0.039
   0.450        0.039        0.039
   0.460        0.039        0.039
   0.470        0.039        0.039
   0.480        0.039        0.039
   0.490        0.039        0.039
```

(a) Response to a Step Change

```
INPUT Y TO RERUN? Y
DELTA T,SECS? .01
REQUIRED TIME INTERVAL BETWEEN PRINTOUTS,SECS? .01
INPUT Y TO CHANGE DISTURBING SIGNAL? Y
THIS IS DISTURB SUBROUTINE ITEM 2
ENTER STEP,RAMP,PULSE OR SINE? PUL
MAGNITUDE OF PULSE? .0715
DURATION OF PULSE, SECONDS? .2
INPUT Y TO CHANGE CONTROLLER SETTINGS? N
   TIME      DISTURBING    RESPONSE
   SECS        SIGNAL
   0.000        0.072        0.000
   0.010        0.072        0.004
   0.020        0.072        0.011
   0.030        0.072        0.020
   0.040        0.072        0.030
   0.050        0.072        0.039
   0.060        0.072        0.049
   0.070        0.072        0.058
   0.080        0.072        0.065
   0.090        0.072        0.071
   0.100        0.072        0.076
   0.110        0.072        0.079
   0.120        0.072        0.082
   0.130        0.072        0.083
   0.140        0.072        0.083
   0.150        0.072        0.083
   0.160        0.072        0.082
   0.170        0.072        0.080
   0.180        0.072        0.079
   0.190        0.072        0.077
   0.200        0.072        0.076
   0.210        0.000        0.070
   0.220        0.000        0.062
   0.230        0.000        0.052
   0.240        0.000        0.042
   0.250        0.000        0.031
   0.260        0.000        0.021
   0.270        0.000        0.012
   0.280        0.000        0.005
```

(b) Response to a Pulse

```
   0.290        0.000       -0.001
   0.300        0.000       -0.006
   0.310        0.000       -0.009
   0.320        0.000       -0.011
   0.330        0.000       -0.012
   0.340        0.000       -0.012
   0.350        0.000       -0.012
   0.360        0.000       -0.010
   0.370        0.000       -0.009
   0.380        0.000       -0.007
   0.390        0.000       -0.005
   0.400        0.000       -0.004
   0.410        0.000       -0.002
   0.420        0.000       -0.001
   0.430        0.000       -0.000
   0.440        0.000        0.001
   0.450        0.000        0.001
   0.460        0.000        0.002
   0.470        0.000        0.002
   0.480        0.000        0.002
   0.490        0.000        0.002
INPUT Y TO RERUN? N
Ok
```

```
10  REM ****************************************************
20  REM - PROGRAM GRAPH1.BAS THIS PROGRAM IS USED
30  REM - TO GRAPH THE OUTPUT FROM CNTRL1.BAS
40  REM - PROGRAM DESCRIPTION
50  REM - LINES 1000,1690  Used to erase the soft key
60  REM - display on commencement and to reinstate the
70  REM - display on completion, of a program run
80  REM - LINE 1190  Used to avoid the "Input past
90  REM - End" error which would otherwise appear if one
100 REM - inadvertently attempted to access more data than
110 REM - that actually held in the file
120 REM - LINES 1390 -  The screen is cleared (line 1390);
130 REM - the scale of the graph is set (line 1400); the
140 REM - graph of the disturbance (lines 1420 & 1440) and
150 REM - of the response (lines 1430 & 1450) are drawn
160 REM - LINE 1520  Stops the program until any key
170 REM - is pressed
180 REM ****************************************************
1000 KEY OFF
1010 DIM M(100,100)
1020 REM - FILES *******************************************
1030 PRINT "ON WHICH DISC DO THE FILES RESIDE"
1040 INPUT "TYPE A,B,C,OR D";A$
1050 IF A$="A" THEN 1100
1060 IF A$="B" THEN 1120
1070 IF A$="C" THEN 1140
1080 IF A$="D" THEN 1160
1090 GOTO 1030
1100 OPEN "A:DATA" FOR INPUT AS #1
1110 GOTO 1170
1120 OPEN "B:DATA" FOR INPUT AS #1
1130 GOTO 1170
1140 OPEN "C:DATA" FOR INPUT AS #1
1150 GOTO 1170
1160 OPEN "D:DATA" FOR INPUT AS #1
1170 B1=0
1180 FOR J=1 TO 50
1190 IF EOF(1) THEN 1610
1200 INPUT #1,M(J,1),M(J,2),M(J,3)
1210 NEXT J
1220 IF A1>0 THEN 1410
1230 REM - GRAPH SIZE **************************************
1240 PRINT "INPUT COORDINATES FOR BTM LH CORNER"
1250 INPUT "OF GRAPH X,Y";X1,Y1
1260 PRINT "INPUT COORDINATES FOR TOP RH CORNER"
1270 INPUT "OF GRAPH X,Y";X2,Y2
1280 PRINT "INPUT Y TO SUPERIMPOSE GRAPHS ELSE N"
1290 PRINT "AFTER THEY APPEAR PRESS ANY KEY TO CONTINUE"
1300 INPUT B$
1310 IF B$="Y" THEN 1360
1320 IF B$="N" THEN 1340
1330 GOTO 1280
1340 A1=1
1350 GOTO 1390
1360 PRINT "HOW MANY GRAPHS";
1370 INPUT A1
1380 REM - PLOT THE GRAPHS *********************************
1390 SCREEN 2:CLS
1400 WINDOW (X1,Y1)-(X2,Y2)
1410 FOR J=1 TO 50
1420 PSET (M(J,1),M(J,2))
```

```
1430 PSET (M(J,1),M(J,3))
1440 LINE (M(J-1,1),M(J-1,2))-(M(J,1),M(J,2))
1450 LINE (M(J-1,1),M(J-1,3))-(M(J,1),M(J,3))
1460 NEXT J
1470 B1=B1+1
1480 IF B1=A1 THEN 1500
1490 GOTO 1180
1500 A1=0
1510 B1=0
1520 C$=INKEY$: IF C$="" THEN 1520
1530 CLS
1540 PRINT "INPUT S TO RUN SAME DATA,N TO RUN
1550 INPUT "NEXT SET,ELSE E";B$
1560 IF B$="E" THEN 1590
1570 IF B$="S" THEN 1240
1580 IF B$="N" THEN 1180
1590 CLOSE #1
1600 GOTO 1630
1610 PRINT "END OF FILE"
1620 CLOSE #1
1630 PRINT "INPUT A TO RERUN FROM THE BEGINNING,"
1640 INPUT "ELSE E";B$
1650 IF B$="E" THEN 1690
1660 IF B$<>"A" THEN 1630
1670 A1=0
1680 GOTO 1050
1690 KEY ON
1700 END
```

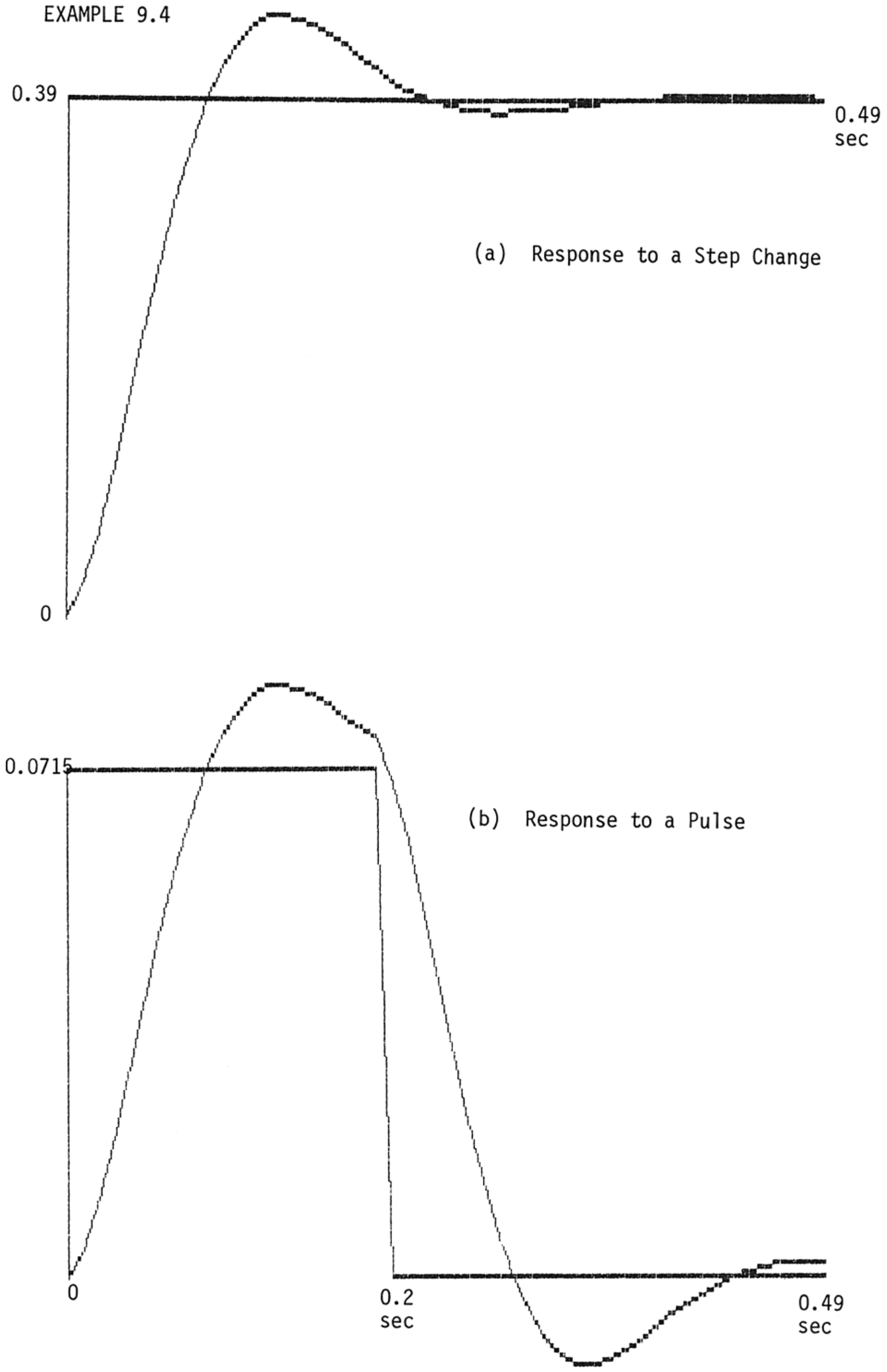
```
LOAD"A:GRAPH1
Ok
RUN
ON WHICH DISC DO THE FILES RESIDE
TYPE A,B,C,OR D? A
INPUT COORDINATES FOR BTM LH CORNER
OF GRAPH X,Y? -.1,-.01
INPUT COORDINATES FOR TOP RH CORNER
OF GRAPH X,Y? .7,.06
INPUT Y TO SUPERIMPOSE GRAPHS ELSE N
AFTER THEY APPEAR PRESS ANY KEY TO CONTINUE
? N
```

Draw graph (a),
response to a step
change.

```
INPUT S TO RUN SAME DATA,N TO RUN
NEXT SET,ELSE E? N
INPUT COORDINATES FOR BTM LH CORNER
OF GRAPH X,Y? -.1,-.03
INPUT COORDINATES FOR TOP RH CORNER
OF GRAPH X,Y? .7,.1
INPUT Y TO SUPERIMPOSE GRAPHS ELSE N
AFTER THEY APPEAR PRESS ANY KEY TO CONTINUE
? N
```

Draw graph (b),
response to a pulse.

COMPUTER - GENERATED PLOTS OF DISTURBANCE AND RESPONSE



(a) Response to a Step Change

(b) Response to a Pulse

Example 9.5

Repeat Examples 9.2 and 9.3 using the above programs.

First the Block Diagram is redrawn, showing the item numbers to be used when inputting data (Figure 9.9).

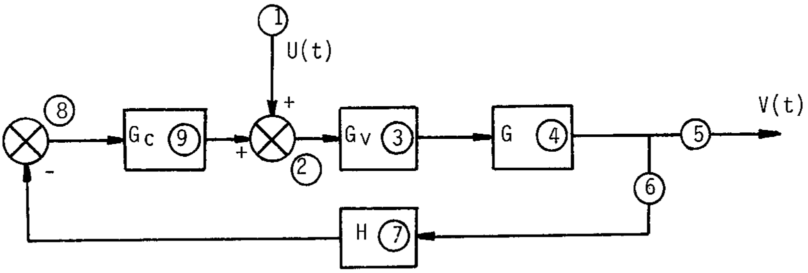


Figure 9.9. Block diagram for Example 9.5.

In repeating Example 9.2 the following values are used:

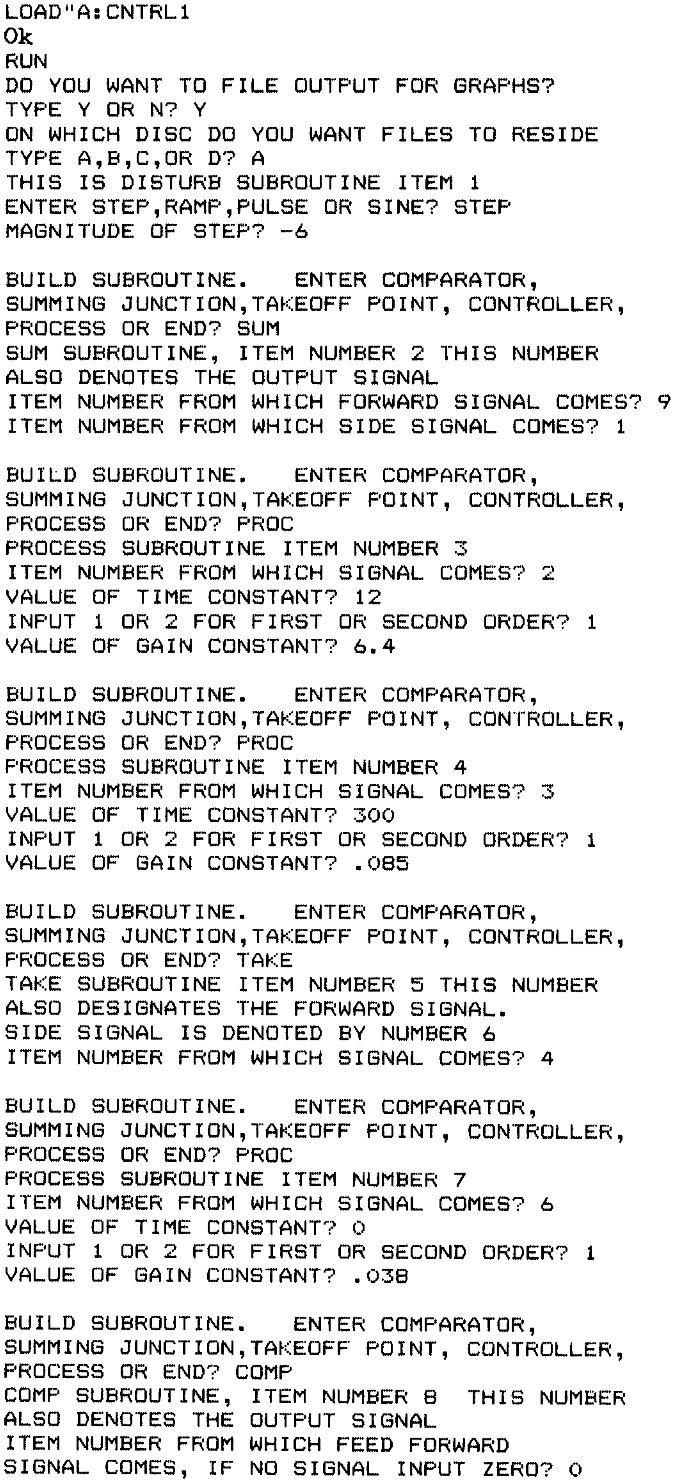
Item 1 - Step, magnitude = -6
" 2 - Sum
" 3 - Process, 1st order, τ = 12, K = 6.4
" 4 - Process, 1st order, τ = 300, K = 0.085
" 5 - Takeoff
" 6 - "
" 7 - Process, 1st order, τ = 0, K = 0.038
" 8 - Comparator, no feedforward signal
" 9 - Controller, K = 350
τ_D = 0
$\tau_I = \infty$

In repeating Example 9.3a the following controller settings are used:

K = 350
τ_D = 10
τ_I = 25

In repeating Example 9.3b the following controller settings are used:

K = 50
τ_D = 10
τ_I = 8.23



```
LOAD"A:CNTRL1
Ok
RUN
DO YOU WANT TO FILE OUTPUT FOR GRAPHS?
TYPE Y OR N? Y
ON WHICH DISC DO YOU WANT FILES TO RESIDE
TYPE A,B,C,OR D? A
THIS IS DISTURB SUBROUTINE ITEM 1
ENTER STEP,RAMP,PULSE OR SINE? STEP
MAGNITUDE OF STEP? -6

BUILD SUBROUTINE.   ENTER COMPARATOR,
SUMMING JUNCTION,TAKEOFF POINT, CONTROLLER,
PROCESS OR END? SUM
SUM SUBROUTINE, ITEM NUMBER 2 THIS NUMBER
ALSO DENOTES THE OUTPUT SIGNAL
ITEM NUMBER FROM WHICH FORWARD SIGNAL COMES? 9
ITEM NUMBER FROM WHICH SIDE SIGNAL COMES? 1

BUILD SUBROUTINE.   ENTER COMPARATOR,
SUMMING JUNCTION,TAKEOFF POINT, CONTROLLER,
PROCESS OR END? PROC
PROCESS SUBROUTINE ITEM NUMBER 3
ITEM NUMBER FROM WHICH SIGNAL COMES? 2
VALUE OF TIME CONSTANT? 12
INPUT 1 OR 2 FOR FIRST OR SECOND ORDER? 1
VALUE OF GAIN CONSTANT? 6.4

BUILD SUBROUTINE.   ENTER COMPARATOR,
SUMMING JUNCTION,TAKEOFF POINT, CONTROLLER,
PROCESS OR END? PROC
PROCESS SUBROUTINE ITEM NUMBER 4
ITEM NUMBER FROM WHICH SIGNAL COMES? 3
VALUE OF TIME CONSTANT? 300
INPUT 1 OR 2 FOR FIRST OR SECOND ORDER? 1
VALUE OF GAIN CONSTANT? .085

BUILD SUBROUTINE.   ENTER COMPARATOR,
SUMMING JUNCTION,TAKEOFF POINT, CONTROLLER,
PROCESS OR END? TAKE
TAKE SUBROUTINE ITEM NUMBER 5 THIS NUMBER
ALSO DESIGNATES THE FORWARD SIGNAL.
SIDE SIGNAL IS DENOTED BY NUMBER 6
ITEM NUMBER FROM WHICH SIGNAL COMES? 4

BUILD SUBROUTINE.   ENTER COMPARATOR,
SUMMING JUNCTION,TAKEOFF POINT, CONTROLLER,
PROCESS OR END? PROC
PROCESS SUBROUTINE ITEM NUMBER 7
ITEM NUMBER FROM WHICH SIGNAL COMES? 6
VALUE OF TIME CONSTANT? 0
INPUT 1 OR 2 FOR FIRST OR SECOND ORDER? 1
VALUE OF GAIN CONSTANT? .038

BUILD SUBROUTINE.   ENTER COMPARATOR,
SUMMING JUNCTION,TAKEOFF POINT, CONTROLLER,
PROCESS OR END? COMP
COMP SUBROUTINE, ITEM NUMBER 8  THIS NUMBER
ALSO DENOTES THE OUTPUT SIGNAL
ITEM NUMBER FROM WHICH FEED FORWARD
SIGNAL COMES, IF NO SIGNAL INPUT ZERO? 0
```

```
ITEM NUMBER FROM WHICH FEED BACK SIGNAL COMES? 7

BUILD SUBROUTINE.   ENTER COMPARATOR,
SUMMING JUNCTION,TAKEOFF POINT, CONTROLLER,
PROCESS OR END? CONTROL
CONTROL SUBROUTINE ITEM NUMBER 9 THIS NUMBER
ALSO DESIGNATES THE OUTPUT SIGNAL
ITEM NUMBER FROM WHICH SIGNAL COMES? 8
VALUE OF GAIN CONSTANT? 350
VALUE OF DERIVATIVE TIME, SECONDS? 0
VALUE OF INTEGRAL TIME, SECONDS? 9E19

BUILD SUBROUTINE.   ENTER COMPARATOR,
SUMMING JUNCTION,TAKEOFF POINT, CONTROLLER,
PROCESS OR END? END

DELTA T, SECONDS? 1
ITEM  NUMBER FROM WHICH THE REQUIRED OUTPUT
SIGNAL COMES? 5
REQUIRED TIME INTERVAL BETWEEN PRINTOUTS,
SECONDS? 10

   TIME       DISTURBING   RESPONSE
   SECS        SIGNAL
   0.000      -6.000        0.000
  10.000      -6.000       -0.036
  20.000      -6.000       -0.104
  30.000      -6.000       -0.176
```

Solution to Example 9.2
Response data output by the
computer has been omitted

```
 460.000      -6.000       -0.396
 470.000      -6.000       -0.396
 480.000      -6.000       -0.396
 490.000      -6.000       -0.396
INPUT Y TO RERUN? Y
DELTA T,SECS? 1
REQUIRED TIME INTERVAL BETWEEN PRINTOUTS,SECS? 10
INPUT Y TO CHANGE DISTURBING SIGNAL? N
INPUT Y TO CHANGE CONTROLLER SETTINGS? Y
INPUT ITEM NUMBER OF CONTROLLER? 9
VALUE OF GAIN CONSTANT? 350
VALUE OF DERIVATIVE TIME, SECONDS? 10
VALUE OF INTEGRAL TIME, SECONDS? 25
   TIME       DISTURBING   RESPONSE
   SECS        SIGNAL
   0.000      -6.000        0.000
  10.000      -6.000       -0.034
  20.000      -6.000       -0.094
  30.000      -6.000       -0.152
```

Solution to Example 9.3a
Response data output by the
computer has been omitted

```
 460.000      -6.000      -0.003
 470.000      -6.000      -0.005
 480.000      -6.000      -0.007
 490.000      -6.000      -0.007
INPUT Y TO RERUN? Y
DELTA T,SECS? 1
REQUIRED TIME INTERVAL BETWEEN PRINTOUTS,SECS? 10
INPUT Y TO CHANGE DISTURBING SIGNAL? N
INPUT Y TO CHANGE CONTROLLER SETTINGS? Y
INPUT ITEM NUMBER OF CONTROLLER? 9
VALUE OF GAIN CONSTANT? 350
VALUE OF DERIVATIVE TIME, SECONDS? 10
VALUE OF INTEGRAL TIME, SECONDS? 8.23
   TIME      DISTURBING   RESPONSE
   SECS        SIGNAL
   0.000      -6.000       0.000
  10.000      -6.000      -0.034
  20.000      -6.000      -0.092
  30.000      -6.000      -0.140
                 ↓
```

Solution to Example 9.3b
Response data output by the
computer has been omitted

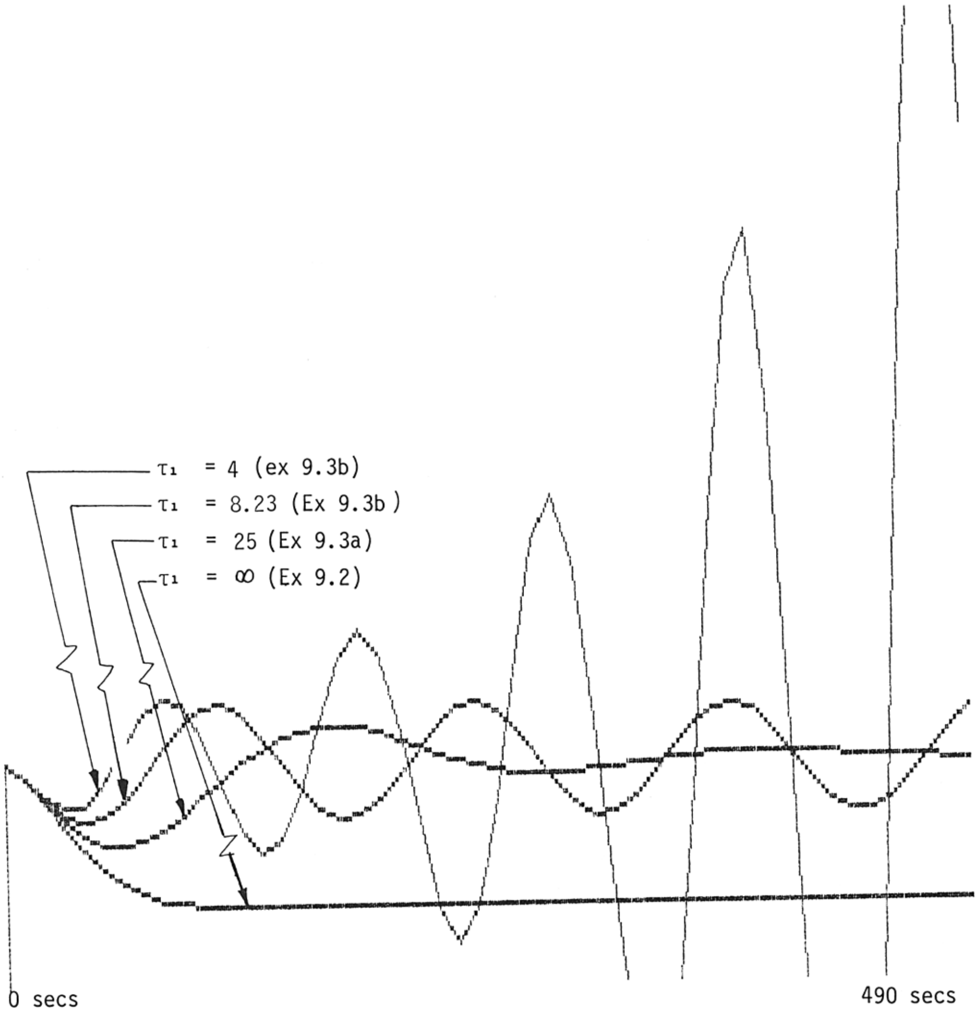
```
 460.000      -6.000      -0.055
 470.000      -6.000       0.014
 480.000      -6.000       0.079
 490.000      -6.000       0.127
INPUT Y TO RERUN? Y
DELTA T,SECS? 1
REQUIRED TIME INTERVAL BETWEEN PRINTOUTS,SECS? 10
INPUT Y TO CHANGE DISTURBING SIGNAL? N
INPUT Y TO CHANGE CONTROLLER SETTINGS? Y
INPUT ITEM NUMBER OF CONTROLLER? 9
VALUE OF GAIN CONSTANT? 350
VALUE OF DERIVATIVE TIME, SECONDS? 10
VALUE OF INTEGRAL TIME, SECONDS? 4
   TIME      DISTURBING   RESPONSE
   SECS        SIGNAL
   0.000      -6.000       0.000
  10.000      -6.000      -0.034
  20.000      -6.000      -0.088
  30.000      -6.000      -0.122
                 ↓
```

Solution to Example 9.3b continued.
Response data omitted

```
 460.000      -6.000       1.587
 470.000      -6.000       2.669
 480.000      -6.000       2.773
 490.000      -6.000       1.704
INPUT Y TO RERUN? N
```

```
LOAD"A:GRAPH1
Ok
RUN
ON WHICH DISC DO THE FILES RESIDE
TYPE A,B,C,OR D? A
INPUT COORDINATES FOR BTM LH CORNER
OF GRAPH X,Y? -.1,-.6
INPUT COORDINATES FOR TOP RH CORNER
OF GRAPH X,Y? 650,2
INPUT Y TO SUPERIMPOSE GRAPHS ELSE N
AFTER THEY APPEAR PRESS ANY KEY TO CONTINUE
? Y
HOW MANY GRAPHS? 4
```

The four plots generated above are shown superimposed below. Note that the scale chosen does not permit inclusion of the disturbance.



The results obtained show the value of T = 8.23 to be close to critical; the response is decaying slowly. Finally, using the value of T = 4, an unstable response is demonstrated.

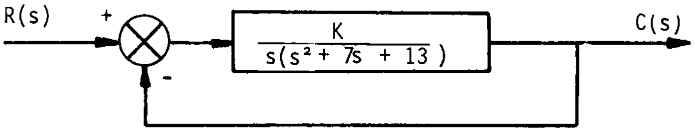
PROBLEMS - CHAPTER NINE

1. The response of the tube side effluent temperature of a shell and tube heat exchanger to changes in shell side flowrate is described by the following transfer function:

$$G = \frac{3.5}{(1+28s)(1+4s)} \quad \frac{^{0}C}{\text{litre/s}}$$

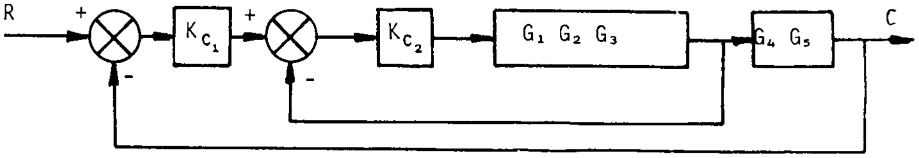
Plot the response of the effluent temperature to a step disturbance in the shell side flowrate of 1 litre/sec.

2. The Block Diagram for a control system is given below. Select a value of K which will provide an acceptable response:



3. Water flows through two tanks in series, with hold up capacities of 1 and 2m^3 respectively, at the rate of 1.5 kg/s. 1 litre of N H_2SO_4 is accidentally dropped into the first tank. Assume it is instantly dispersed into the rest of the contents of the tank, and that perfect mixing occurs in each tank. Hence plot the concentration of acid in the effluent from the second tank as a function of time.

4. Modify the program to include for the presence in the circuit of more than one controller. Use the modified program to determine optimum settings for the two proportional controllers in the cascade system shown below:



G_1, G_2, G_3, G_4 and G_5 are each first order functions in series, with time constants of 10, 20, 30, 40, and 50 seconds respectively. Gain constants are 2, 1, 1, 2, 1 respectively.

5. Write additional subroutines to represent the following functions:

   Capacitance;
   transportation lag;
   exo/endo thermic reactions;
   flow resistance

6. The program assumes that responses from controller, valve, transmitter, etc. are all unbounded. In fact such devices have a limited range of response. For example, pneumatic systems frequently operate within the range of 0 to 15 psig, and electrical systems within the range 0 to 20m A.

   Modify the program to account for such limitations.

REFERENCES

1. F.H. Raven, Automatic Control Engineering, McGraw Hill Book Co., New York, U.S.A., 1968
2. D.R. Coughanowr, L.B. Koppel, Process Systems Analysis and Control, McGraw Hill Book Co., New York, U.S.A., 1965
3. A. Pollard, Process Control, Heinemann Educational Books, London, U.K. 1971
4. D.D. Perlmutter, Introduction to Chemical Process Control, John Wiley & Sons, New York, U.S.A., 1965
5. G.A. Korn, J.V. Wait, Digital Continuous - System Simulation, Prentice Hall Inc. 1978
6. Z. Bulnicki, Identification of Control Plants, Elsevier, Amsterdam, 1980
7. P.L. Ginn, An Introduction to Process Control and Digital Microcomputers, Gulf Publishing, Houston, Texas, U.S.A., 1982
8. D.K. Anand, Introduction to Control Systems, Pergamon, 1984
9. T.H. Tsai, J.W. Lane, C.S. Lin, Modern Control Techniques for the Processing Industries, Marcel Dekker Inc., New York, U.S.A., 1986.

SUBJECT INDEX