1961

# General Information Manual
# An Introduction to
# Engineering Analysis for Computers

IBM

IBM

# General Information Manual
# An Introduction to
# Engineering Analysis for Computers

Address comments regarding this publication to:
IBM Technical Publications Department
112 East Post Road, White Plains, New York

CONTENTS

# SECTION I: INTRODUCTION

## Historical Notes

To most engineers practicing in industry the high speed digital computer is an unknown quantity. Just five years ago an engineer's chances of having an introduction to computing while in college were slim indeed.

Today virtually all the major engineering schools in this country have access to digital computers. Within the next five years the engineer who graduates without an introductive course in the use of computers will be the exception.

The growth in the use of the digital computer has been tremendous. The first large scale digital computers appeared in the early 1950s as an outgrowth of interest generated through the use of punched card calculators such as the IBM Card Programmed Calculator. Over 4,000 digital computers were installed in 1960, varying from small engineering machines to very large commercial and scientific data processing systems.

Like all technological developments, the digital computer has its historical antecedents. Computing itself is one of the oldest human activities, required by all civilizations for the conduct of business and the development of sciences. One of the most practical inventors of recent times, Charles Kettering, inventor of the auto self-starter, made the statement, "You cannot understand a thing unless you can give it a number." In many ways civilization has always been tied to this need for quantitative or numerical description.

In fact, the oldest surviving written document is a set of business records kept by a Sumerian of Mesopotamia 5,000 years ago.

The development of computing methods was painfully slow. Perhaps this was due in part to the unfortunate choice of number symbols (e.g., Roman numerals) and of number bases (e.g., the Babylonian choice of the base 60). Gradually the Arabic numeral system and the base 10 won out as modes of computation, although vestiges of other systems remain important (e.g., Roman numerals for date inscription, and base 12 for the inches in a foot measure).

Perhaps, too, this slow march forward in computing methods was due to the fact that they were not greatly needed, for certainly the first big breakthrough occurred when the science of astronomy depended on tremendous computing efforts for its advance. In the 16th century these efforts were greatly eased by the development of logarithms by Napier and Briggs. Considering the size of numbers used in astronomy, one can easily see why the logarithmic transformation which allows multiplication and division to be handled as addition or subtraction was a significant contribution.

In 1642 Blaise Pascal, a French mathematician, built the first successful digital computing machine. While this device demonstrated the feasibility of mechanical computation, the use of such devices remained very limited.

A most interesting personality in the science of machine computation was the misunderstood genius Charles Babbage of England, whose major work was done in the middle 1800s. During his lifetime, Babbage conceived and designed on paper what he termed the analytical machine. In virtually all respects this machine was to perform like the modern digital computer, except that it was to be mechanical rather than electronic. The fascination which Babbage's work holds for present-day computer people can be explained when some of his ideas are discussed.

For example, his analytical machine was to have the ability to "remember" 1,000 numbers up to 50 digits in length. This capacity is greater than that of many electronic computers now in existence. His machine was to be controlled by punched cards, an idea which he borrowed from the use of cards in controlling looms. In the light of the present day role of the punched card in data processing, Charles Babbage seems a prophet.

It is a certainty that if Babbage had been content to build on a smaller scale rather than design on such a grandiose level, the science of computing would have profited greatly in the 100 years between his analytical machine and the present day. As it was, however, Babbage never completed his machine, and in general we are aware of his genius primarily through the writings of an admirer, one Lady Lovelace, who wrote most descriptively of the design and philosophy of the analytical machine.

In truth one of her statements might stand as a preface to the study of the modern electronic computer: "The Analytical Engine has no pretensions whatever to originate anything; it can do whatever we know how to order it to perform."

Charles Babbage did successfully build what is termed a difference machine for the computation of tables. Many of the early efforts in machine computation took this line of construction of machines to do particular computing jobs. A simple example will illustrate a principle made use of in such machines:

To construct a table of values for

$$F = M^3 + M^2$$

where M is to take on the values 0, 1, 2, 3, -------, set down the first few entries and take successive differences in F:

| | | Differences | | |
|---|---|---|---|---|
| M | F | 1st | 2nd | 3rd |
| 0 | 0 | | | |
| | | 2 | | |
| 1 | 2 | | 8 | |
| | | 10 | | 6 |
| 2 | 12 | | 14 | |
| | | 24 | | 6 |
| 3 | 36 | | 20 | |
| | | 44 | | 6 |
| 4 | 80 | | 26 | |
| | | 70 | | 6 |
| 5 | 150 | | 32 | |
| | | 102 | | |
| 6 | 252 | | | |

The third difference is a constant. The table can be completed by working back from the third difference to compute further entries of F by simple additions. It is relatively simple to build a machine to perform these simple additions to successive columns.

The principle used here has been to take successive derivatives of the function F.

$$\frac{dF}{dM} = 3M^2 + 2M$$

$$\frac{d^2F}{dM^2} = 6M + 2$$

$$\frac{d^3F}{dM^3} = 6$$

The third derivative is the constant 6.

A great many of the functions for which tables are desired in the fields of navigation, astronomy and surveying can be handled by such techniques. The literature of computing abounds with the application of difference procedures.

However, modern computers are a far cry from the difference machines of Babbage's day.

At about the same time Babbage was laboring with his "Analytical Engine," George Boole, an English mathematician, was laying the foundation for later developments with his study of the algebra of logic. Boolean algebra is the foundation of work in machine logic (so necessary in the design of digital computers) and has contributed in the logic of business contracts and law (equally necessary in many areas of data processing).

In the 1930s the work of modern mathematicians, in particular the late John von Neumann, spurred interest in the possibilities of machine computation. Various computers — both special-purpose analog devices and true general-purpose electromechanical digital computers — were built

at universities. Harvard and MIT were the centers of most of this activity. World War II was a catalyst to machine computing in that the military technological advancements demanded tremendous computational facilities. By the end of the war the logic for construction of a general-purpose digital computer was fairly well specified.

As makers of computing devices for accounting and business, manufacturers of business machines became involved in this effort of machine computation. The first large scale general-purpose digital computer was built by IBM for use in scientific research. Known as the IBM Automatic Sequence Controlled Calculator, it was presented to Harvard in August 1944. Its capacities were not surpassed until the IBM Selective Sequence Electronic Calculator was dedicated in January 1948.

The SSEC was dismantled to make room for its successor, the IBM 701, announced in March 1953. At this time the general belief was that a few large scale machines could handle all the computing needs of the country. The demand for more and more machines when the first few became available soon made apparent the vast needs for computation in both technology and business. Today these needs have been multiplied by further recognition of existing needs and the discovery of new approaches to old problems.

How does this affect the engineer?

The field of engineering analysis for computers has experienced tremendous growth in the past five years. The advent of the electronic computer has created a new approach to engineering problems. The costly "build and try" method is now often replaced by "construct mathematical model and simulate."

In some respects this represents a return to the textbook for the engineer. He must describe the physical problem by means of a mathematical model. Then the computer can operate the model and describe operating results. The real power of the computer is that it will quickly describe operating results for many sets of operating conditions.

Many articles have been printed in professional publications on the solution of difficult engineering problems by the use of electronic computers. These applications have only scratched the surface of possible engineering use, and a lack of knowledge and experience prevents the full use of the computer as an engineering tool. The engineer is simply not aware of what a computer can do for him. It is highly desirable, therefore, that every engineer understand how problems can be described for handling by electronic computers.

It is convenient to think of computer problems as falling into one of two classes:

1. Obvious computations

2. Iterative problems

Obvious computations are those which have in the past been handled by slide rule or paper and pencil. They also include those for which the technique of solution, while known, has required too much computation to tackle.

Iterative problems comprise a large area, mostly unexplored. To explain this, the question should be asked, "What is the function of the engineer today?"

The engineer is a person who builds physical systems to do particular jobs. If a particular construction does not work, modifications are made until it does work. These modifications are often the result of vague ideas. Engineering is largely by trial and error.

A very obvious conclusion which one reaches in talking with engineers is that they employ a minimum of their textbook analysis techniques. Mathematical approaches soon pick up dust on the engineer's worktable. Why? One reason is that real life simply does not fit the pat equations of the textbooks. Often the basic difference is that in a textbook it always seems possible to apply mathematical techniques and arrive at solutions of the kind

$$x = f(y, z)$$

where the value of the variable of interest can be determined simply by plugging in known values for y and z on the right-hand side. However, in real-life engineering, often the problem becomes stated as

$$x = f(x, y, z)$$

where x is on both sides of the equation. x cannot be solved for explicitly; that is, there is not enough information to remove x from the right-hand side. The problem is too complex.

This problem can usually be handled mathematically by estimating the value of x, testing the estimate, and, if it is wrong, making a better estimate. Essentially, numerical analysis (in computer mathematics) is the science of making progressively better estimates. To estimate repeatedly is to "iterate." This iterative idea is the heart of the computer approach, whether for the simplest cam problem requiring algebra, or for the most complex nuclear problem involving sets of differential equations.

This manual describes typical problems which have been handled by engineers making use of computers. No attempt is made here to give all the mathematical techniques required in computer analysis, nor to explain fully the programming techniques required for control of electronic computers. It is felt that an introduction to actual computer problems, with an attempt toward classification as to type, will give the practicing engineers insight into what computing can do and perhaps suggest possible uses they can personally make of engineering computing analysis.

## The Digital Computer

The layman's image of the electronic computer contains a pinch of mystery, a dash of magic and a number of false analogies to human beings. These misconceptions have in some cases been nurtured by computer manufacturers through their use of such terminology as

1. Memory — for that part of the computer which stores information.

2. Nerve Center — for that part of the computer which controls its operation.

3. Brain — for that part of the computer which accomplishes arithmetic and logical operations.

While these analogies do help in understanding the functions of some of the components of a computer, they also tend to develop a camouflage of complexity.

The computer, like the automobile, airplane, nuclear reactor or any other engineering achievement, is the product of a group of engineers. (It should not be surprising that in developing this product, engineers have used other computers to solve some of their design problems.) As in the case of the automobile and airplane, a person need not be capable of designing and building a computer in order to use one. Few persons who drive an automobile are capable of describing the complex of gears, rods and shafts that allow a rotation of the steering wheel to change the direction of travel. This, however, does not prevent them from becoming excellent drivers. Similarly, a general knowledge of the functional components of a computer is all that is needed to use one effectively. Once the basic concepts are understood, greater familiarity is easily developed through experience.

What are these basic concepts? To answer this, consider the basic components of a computer.

### STORAGE

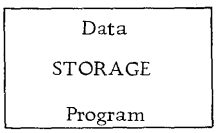
That part of a computer which most differentiates it from a calculator (desk-type, slide rule, adding machine) is its storage — or, to use the unapproved analogy, its "memory."


A computer can store numbers, alphabetic characters and some special symbols. In any calculation, it is necessary to store the numbers which begin the calculations, intermediate results and — at least temporarily — final results. A desk calculator has a very limited ability to store data. The operator usually must enter each number as it is needed. On the other hand, a small computer (such as the IBM 1620) can store 20,000

digits and call them out of storage whenever they are needed in a calculation.

THE STORED PROGRAM

The ability to store large amounts of data that is to be used in or has been developed by calculations is only part of the function of storage. Equally significant is the fact that the computer also contains within storage the program for the calculations to be performed.




A program is made up of instructions which, when acted upon by the computer, cause it to go through the sequence of operations necessary to arrive at a meaningful result. A single instruction may:

1. Cause data to be brought into storage from some external source, such as a card reader.

2. Cause a specified arithmetic operation to be performed on selected numbers.

3. Constitute a logical test to determine what part of the program should be performed next.

4. Cause results to be sent from storage to a recording device, such as a typewriter.

After both the data to be operated upon and the program which describes the operations are in storage, the computer is free to proceed with a series of calculations at its own natural speed. For present-day machines, this may be from 50 to 500,000 additions per second.

This leads to another salient fact: A computer can change or modify its own program! Since the program is stored in much the same form as data, one portion of a program may be a sequence of operations which will examine another portion of the same program and change it by arithmetic or logical manipulations. What advantage is to be gained from this?

First, it means that one set of instructions can be used to operate on a number of sets of data stored in different locations in storage. As each set of data is processed, the program is changed to refer to the next set of data.

Second, it allows the program to be changed on the basis of conditions which arise during the calculations. For example, suppose a calculation involves the evaluation of

$y = x^2$ , for $x \leq 1$

$y = -x^2+4x-2$, for $x > 1$.

At the point in the calculations where the value of x becomes greater than 1, that portion of the program involved in the evaluation of y can be changed to use the second of the two equations. This change might be accomplished by replacing one set of instructions with a new set stored elsewhere for that purpose, or it could also be done by causing the computer to select one of two instruction sequences on the basis of whether x is greater than 1 or not greater than 1.

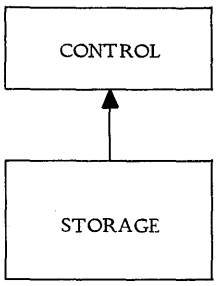
## ADDRESSES

To be able to select the item of data or the instruction to be operated upon next, the computer must be provided with a means of locating the desired information in storage. For this reason, storage is divided into units and each unit is identified by an address. Different computers are designed with different-size units of storage. The basic unit may vary from one decimal digit (as in the IBM 1620) to 36 binary digits (as in the IBM 7090). In other computers, units of one alphameric character (that is, a decimal digit, letter of the alphabet, or special character) or ten decimal digits are used.

Whatever the size of the basic unit, each is assigned a numerical address which identifies the location. Manipulation of the data stored in a location is accomplished through the use of the address corresponding to the location.

## CONTROL UNIT

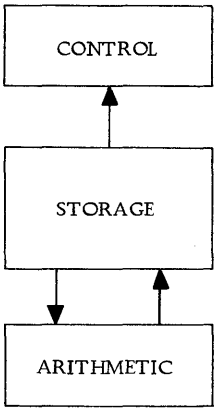
The control unit of a computer provides the means whereby the stored program causes the desired operations to be performed in the sequence specified. The control unit reads an instruction from storage, examines it, sets up the circuit conditions to perform the operation, and, when the operation is completed, reads the next instruction from storage and repeats these steps.




Generally, the first operation to be performed is manually entered into the control unit by the machine operator. Thereafter the action of the computer is completely controlled by the program in storage as interpreted by the control unit. An instruction which tells the computer to stop can be the last one in the program.

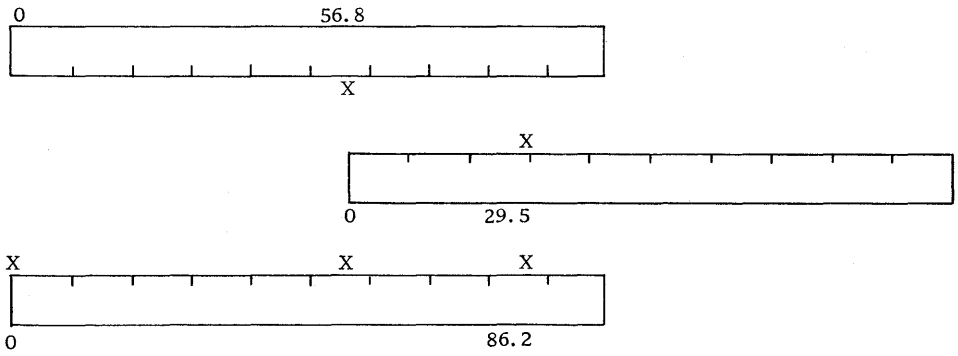
ARITHMETIC UNIT

This component contains the circuitry which performs arithmetic on numbers taken from storage. It usually includes a limited amount of storage in which to hold the operands involved in the arithmetic.




At this point it is wise to pin down the kind of arithmetic accomplished in a computer. The two basic types of computers — analog and digital — have different ways of accomplishing arithmetic. The analog computer performs arithmetic by measuring, the digital computer by applying a set of rules to the numbers involved.

To illustrate, suppose that two numbers to be added are 56.8 and 29.5. A simple analog device for obtaining the sum would be a ruler divided into 100 units. The ruler is laid on a piece of paper and marks are made on the paper next to the 0 and 56.8 points on the rule. Then the ruler is picked up and the zero point of the ruler is placed next to the mark on the paper representing the 56.8 point. Now a third mark is made on the paper opposite the 29.5 point on the rule. All three marks, of course, are made to fall on a straight line. Finally, the distance between the first mark and the third is measured with the rule. Because of the possibility of measurement errors inherent in such a procedure, the result obtained might perhaps be 86.2.



A schoolboy will add the numbers by writing one under the other and applying a set of rules involving such things as the sum of two digits and how to handle the carry when the sum is greater than 9. Thus,

```
 1 1
 5 6.8
 2 9.5
 8 6.3
```

This is the technique used by the digital computer — that is, the digital computer performs arithmetic by the application of a set of rules. There is no error inherent in such a technique.

## BINARY ARITHMETIC

While on the subject of arithmetic, it is well to dispense with another question which is bound to arise in the reader's mind: What is binary arithmetic?

A problem in computer design that the engineer may have recognized by now is that to store and manipulate decimal digits requires some device capable of assuming ten stable and unique states — one state to represent each of the digits 0 to 9. While such devices are available (the notched wheels in a desk calculator are an example), they lack the speed of operation and small size necessary for a computer. Furthermore, they don't fit into an electronic system.

Many electronic devices are available, however, which can assume two stable and unique states. A tube, for example, can be conducting or nonconducting; a magnetic field can have clockwise or counterclockwise rotation; a pulse can be transmitted or its transmission can be blocked.

The computer designer has at his disposal, therefore, a number of devices and techniques for operating in a number system with the base 2 — the binary number system. The rules for arithmetic are also much simpler in the binary number system than in the decimal system (base 10) which we are accustomed to using.

ADDITION TABLES

| | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 10 |

MULTIPLICATION TABLES

| | 0 | 1 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |

As a result, there are two basic types of digital computers: one which operates entirely in the binary number system and another which codes decimal digits as binary numbers. For the latter, the decimal digits appear as their binary equivalents:

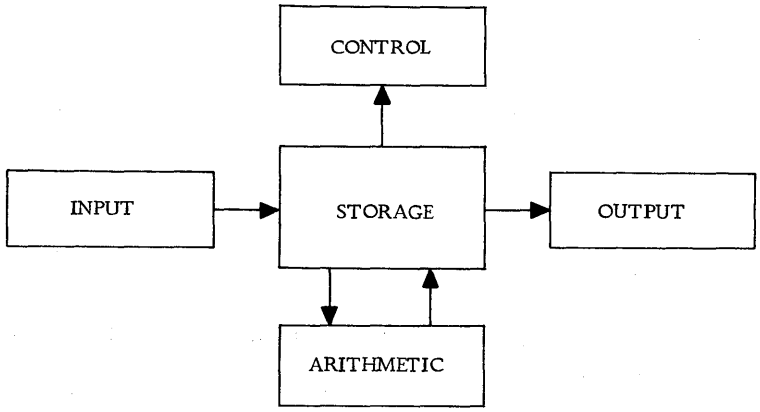
| DECIMAL | BINARY | DECIMAL | BINARY |
|---|---|---|---|
| 0 | 0 | 5 | 101 |
| 1 | 1 | 6 | 110 |
| 2 | 10 | 7 | 111 |
| 3 | 11 | 8 | 1000 |
| 4 | 100 | 9 | 1001 |

As far as the user is concerned, it is not necessary to be familiar with the binary nature of the computer. The computer is capable of translating the engineer's language to its own before starting a program and performs the reverse operation when communicating its results to the engineer.

INPUT AND OUTPUT

So far, the methods for getting data and programs into the computer and for getting the results out have been ignored. Every computer must have a means for communicating with the outside world.

Typical input devices are punched card readers, paper tape readers and manual keyboards. Card and paper tape punches, typewriters and printers are typical output devices. Magnetic tapes and magnetic disks provide a means of storing data




externally from the computer in a form which allows rapid re-entry.

The use of input-output devices is under control of the stored program. For example, an instruction to read a card will cause the card reader to start up, feed and read one card and transmit what has been read into storage.

SUMMARY

Thus there are five basic components of a computer:

1. A place to store data and instructions in such a way that each item can be located when needed.

2. A means of controlling the overall action of the system.

3. Devices for performing arithmetic and other operations required in manipulating the data to obtain desired results.

4. A way of getting information into the computer.

5. A way of getting results out of the computer.

## Exercises

1. The decimal system has as its base the number _____

2. The numeral system commonly used is the _____ numeral system.

3. Logarithms aid greatly in doing the arithmetic operations of _____ and _____ when working with many significant figures.

4. Difference principles were used in the construction of _____

5. The first large scale computers were electromechanical in design; the middle 1950s saw the common use of the _____ design, and the computers being designed and built now have solid-state (e.g., transistorized) components.

6. The general iterative problems may be stated by a formula of the type _____

7. Three computing devices with which most engineers are familiar are _____, _____, and _____

8. One of the difficulties in predicting the behavior of complex systems is that engineers have trouble maintaining the _____

9. Two economic aspects of a problem which most engineers are concerned with are _____ and _____

10. Two types of information stored in the memory of a computer are _____ and _____

11. Modification of the _____ by another set of operations is important for repetitive and logical abilities of a computer.

12. The _____ unit interrogates the memory for information as to what to do next.

13. The _____ gives the physical location of information in memory.

14. An analog device computes by _____, a digital computer by applying a _____

15. The binary system is a number system of base _____

16. Because the information in memory may be changed at any time, the digital computer is called a _____

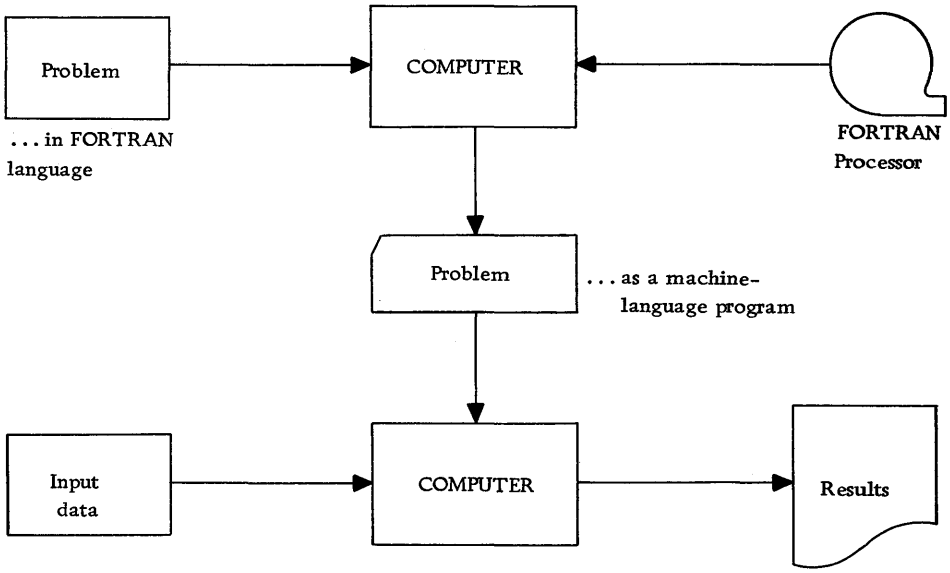
17. The physical property of _____ is what makes a device an acceptable memory component.

## SECTION II: THE FORTRAN LANGUAGE

FORTRAN (FORmula TRANslation) is a language which allows the engineer to communicate a problem to a computer for solution.

FORTRAN is not the natural language of a computer, nor is it the natural language of the engineer. Rather, it is a compromise between the two. To satisfy the computer, it uses symbols that the computer can understand and requires that the rules for their use be closely followed. To satisfy the engineer, it eliminates as many of the detailed computer control operations as possible from the job of writing programs and uses a problem statement format close to that of the mathematical equation.

The engineer describes his problem in the FORTRAN language; what he writes is translated into the natural machine language of the computer to be used in obtaining the solution. The translation is accomplished by the computer itself with the aid of a program called the FORTRAN Processor. The resulting machine-language program is then ready to be used to obtain the solution. This diagram illustrates the procedure:




Statements are the sentences of the FORTRAN language. They may:

1. Define the arithmetic steps which are to be accomplished by the computer.

2. Provide information for control of the computer during the execution of the program.

3. Describe input and output operations necessary to bring in data and write the results.

4. Specify certain additional facts such as the dimensions of a variable which appears in the program with subscripts.

Because the printer and typewriter on a computer print only in upper-case letters, have a limited number of different characters, and are incapable of properly showing exponents and subscripts, FORTRAN statements at first appear somewhat confusing.

An example of an arithmetic statement as it would appear on a FORTRAN coding sheet is:

```
ROOT = ( -B+SQRTF(B**2-4.*A*C))/(2.*A)
```

Translated, this says:

> The quantity to be known as root is equal to — that is, can be determined by — evaluating
>
> $$\frac{-B + \sqrt{B^2 - 4AC}}{2A}$$
>
> where A, B, C are given values stored within the computer.

Arithmetic statements look like simple statements of equality. The right side of all arithmetic statements is an expression which may involve parentheses, operation symbols, constants, variables, and functions, combined in accordance with a set of rules much like that of ordinary algebra. The symbols + and - are employed in the usual way for addition and subtraction. The symbol * is used for multiplication, and the symbol / is used for division. The fifth basic operation, exponentiation, is represented by the symbol **. A**B is used to represent A to the exponent B (that is, $A^B$).

The FORTRAN arithmetic expression

```
A**B*C + D**E/F - G
```

will be interpreted to mean

$$A^B C + \frac{D^E}{F} - G$$

That is, if parentheses are not used to specify the order of operations, the order is assumed to be:

1. exponentiation

2. multiplication and division

3. addition and subtraction

Parentheses are employed in the usual way to specify order. For example

$$(A(B + C))^{D}$$

is written in FORTRAN as

```
(A*(B + C)) **D
```

There are just three exceptions to the ordinary rules of mathematical notation. These are:

1. In ordinary notation AB means A•B or A times B. However, AB never means A*B in FORTRAN. The multiplication symbol cannot be omitted.

2. In ordinary usage, expressions like A/B•C and A/B/C are considered ambiguous. However, such expressions are allowed in FORTRAN and are interpreted as follows:

   A/B*C means (A/B)*C
   A*B/C means (A*B)/C
   A/B/C means (A/B)/C

Thus, for example, A/B/C*D*E/F means ((((A/B)/C)*D)*E)/F. That is, the order of operations is simply taken from left to right, in the same way that

```
A + B - C + D - E
```

means

```
(((A + B) - C) + D) - E
```

3. The expression $A^{B^C}$ is often considered meaningful. However, the corresponding expression using FORTRAN notation, A**B**C, is not allowed in the FORTRAN language. It should be written as (A**B)**C if $(A^B)^C$ is meant, or as A**(B**C) if $A^{(B^C)}$ is meant.

Besides the ability to indicate constants (like 3.57 and 2.), simple variables (like A and ROOT), and operations (like - and *), it is also possible to use functions. In the previous example, SQRTF( ) indicates the square root of the expression in parentheses.

Since the number of possible functions is very large, each computing center will have its own list of available functions, with information about their use. Functions given in this list must be referred to exactly as indicated.

Some functions which might appear in a typical list are:

FORTRAN Symbol

ABSF(X) . . . . . | X | (absolute value of X)
SQRTF(X) . . . . . $\sqrt{X}$
SINF(X) . . . . . sin X
COSF(X) . . . . . cos X
ATANF(X) . . . . . arctan X
EXPF(X) . . . . . $e^X$
LOGEF(X) . . . . . $\log_e X$

Input output statements are those used to bring data into the computer to be stored for processing and to send results out. Typical examples are:

| | |
|---|---|
| READ 1, A, B, C | This statement would cause the next card in the card reader to be read and the three numbers on it stored in locations assigned to the values A, B and C. |
| PRINT 2, ROOT | This statement would cause the number in storage identified as the variable ROOT to be printed. |
| PUNCH 4, SUMA, SUMB | This statement would cause the two values SUM A and SUM B to be punched on a single card. |

Similar input/output statements are included for reading and writing on magnetic tapes and magnetic drums and for such operations as rewinding or backspacing tapes.

The numbers which follow the statements in the examples above specify the format in which the input or output should appear. Usually, a few standard formats will be specified for use in an installation. If a programmer wishes to use something other than a standard format (for example, five numbers per card, in specified columns and in a specified notation), a new format can be introduced by means of a format statement. The FORTRAN Processor interprets the format statement and causes the machine-language program being generated to conform to the desired format in handling input or output statements which refer to it.

Control statements allow the programmer to state the flow of his program through use of the logical powers of the computer. In writing a FORTRAN program, any statement in the program which is referred to by another statement must be given an identifying number (every statement can be given a number if desired but this is not necessary). The control statements refer to these identifying numbers for the purpose of branching from one part of the program to another.

Important statements in this category are illustrated by the following examples:

| | |
|---|---|
| GO TO 4 | This statement indicates that the next statement to be executed (after having been converted to a machine-language program, of course) is the one numbered 4. |
| GO TO (4, 18, 20, 40), K | This statement is referred to as a computed GO TO since the value of K is computed in a previous statement. If, at the time this GO TO were executed, K = 3, then the third alternate (statement 20) would be the one chosen. |
| IF (A**2 - B) 10, 20, 30 | This statement allows one of three alternate next instructions to be chosen according to whether the expression in parentheses is less than, equal to, or greater than zero (in that order). |
| DO 25 I = 1, M, 2 | This statement is a command to execute the sequence of statements which follow, up to and including 25 (or any specified statement number), repeatedly. The first time, let the subscript I on the variables appearing in the statements equal 1 (the first number or variable following the = sign). The second and each subsequent time the sequence is executed, increase I by 2 (the third number or variable following the equal sign — this can be left blank if it is to be 1). When I becomes greater than M (the second number, or variable as in this case) do not repeat, but continue by executing the statement after 25. |
| END | This is the last statement in any FORTRAN program. |

In the DO example, reference was made to the changing of subscripts on variables. For example, A(I, J) would refer to a two-dimensional array of quantities $a_{ij}$. More will be said about this in the next section.

What has been said about the FORTRAN language so far will be more meaningful by examining a complete FORTRAN program. The complete FORTRAN program to find the roots of the quadratic equation

$$ax^2 + bx + c = 0$$

might be written in the following logical steps:

Arithmetic Statements

The two roots are evaluated as arithmetic statements. A separate statement evaluating the discriminant is used to avoid the extra computation

that could be involved if this expression were written in both root statements and to permit later testing of the discriminant.

```
2   D = B**2 - 4.0*A*C

4   ROOT 1 = (-B+SQRTF (D) )/ (2.0*A)

5   ROOT 2 = (-B-SQRTF (D) )/ (2.0*A)
```

All the parentheses in statements 4 and 5 are necessary. D is enclosed in parentheses to define the expression to be operated on by the square root function. -B + SQRTF (D) is in parentheses so the additions and subtractions will be performed prior to the division. 2.0*A is in parentheses so the multiplication will be performed prior to the division.

Input/Output Statements

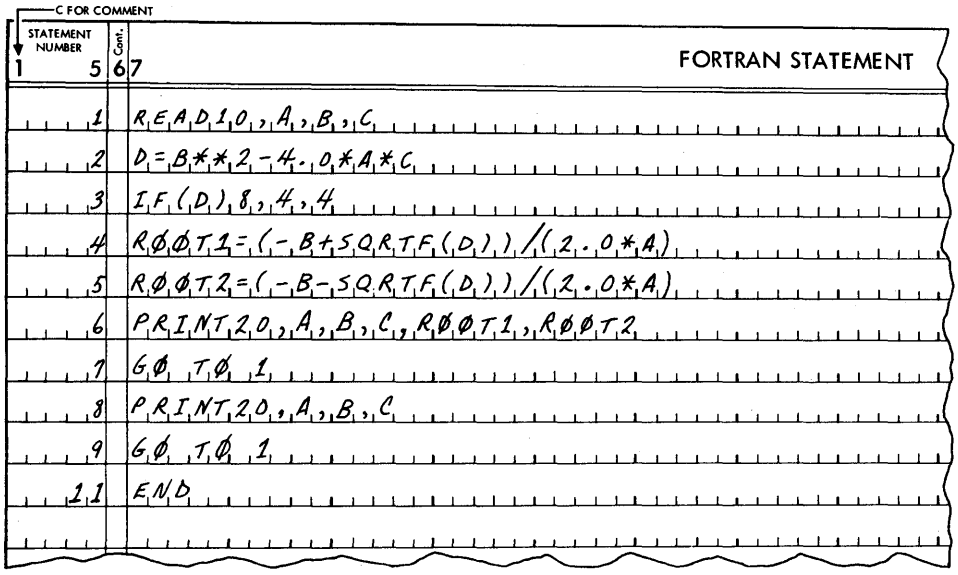
Statements are added to enter values for the constants A, B and C and to print the results of computation. Logically the read statement must precede the arithmetic statements so that values will be available for computation in the arithmetic expressions. The print statement must be after the arithmetic statements, so that values will be available for printing.

```
1   READ 10, A, B, C

2   D = B**2 - 4.0*A*C

4   ROOT 1 = (-B + SQRTF (D) ) / (2.0*A)

5   ROOT 2 = (-B - SQRTF (D) )/ (2.0*A)

6   PRINT 20, A, B, C, ROOT 1, ROOT 2
```

In addition to the roots, the values of the constants A, B and C are printed to identify the results with the input.

Control Statements

Statements are now added for decision making and flow of the program:




Statement 3 is added to test for the mathematical possibility of a negative discriminant (complex roots). If the value of D is zero or positive, the flow of the program will be the normal flow to statement 4. If the value of D is negative, the flow will be to statement 8. Several alternatives are possible at this point depending upon the desired results:

1. If complex roots are desired, they may be calculated and printed in their real and complex parts.

2. This particular set of data could be ignored completely by transferring control to statement 1, where the next set of values would be read.

3. Statement 8 could be a STOP statement which would cause the computer to stop execution of the program and permit manual intervention. This is not a recommended procedure.

4. Statement 8 could be a PRINT statement to signal an error condition. In this program values A, B and C will be printed without the root values and can later be analyzed to determine what was wrong with this set of data.

Statements 7 and 9 are added to complete the flow of the program by returning to statement 1 to read the next set of values.

Statement 11 is the END statement to indicate the end of this program.

This program will continue to be repeated as long as there are cards in the hopper of the card reader. When an attempt is made to read a card after the hopper has emptied, the computer will stop.

Additional examples of FORTRAN programs will be found in the sections that follow. The use of additional statements will be explained at that time.

The FORTRAN language is designed to simplify engineering use of IBM computers. The use of the language with a particular IBM computer requires the use of a FORTRAN Processor for that machine.

The power of IBM computers increases as the size and speed, in terms of machine components, increase. For the more powerful machines the FORTRAN language is expanded to take advantage of the additional types of components. For example, computers with magnetic tape input/output units allow use of FORTRAN statements to control the use of tape.

Should the reader wish to become more expert on the FORTRAN language in general, or as it applies to a particular IBM machine, he may contact the local IBM representative for information on available IBM manuals and education courses.

## Exercises

Write FORTRAN-language programs to:

1. Put the numbers 1, 2, .0396, 30,000.0 and -6.5 into the storage of a computer.

*2. Compute C=A+B and print the resultant value for C, where A=3.5 and B=2.694.

3. Compute C=A+B, where A and B are to be read into the computer and C is to be printed.

*4. Compute C=A+B if $B > 0$
C=A-B if $B \leq 0$

where A and B are read into the computer, and C is to be printed.

*5. Generate the integers 1 to 100 and print.

6. Generate the value of sin x for x = .0000, .0001, .0002, - - - - - .5. Print the value of x and sin x at each interval.

*7. Change the program for solution of a quadratic equation to include solution for complex roots.

* Answers to these exercises are found in Appendix I.

## SECTION III: SOME NEW AND OLD CONCEPTS

Each new area of technology seems to construct a particular vocabulary for the necessary descriptions of efforts and communication of ideas. Computing is no exception to this rule; new terms such as "stored program" and "bit" have been added and new meaning given to old terms such as "iteration," "algorithm" and "memory."

As far as use of computers is concerned, this growth in vocabulary is brought about by the new man-machine relationship. Control of a computing machine requires complete accuracy. This demands conciseness in statement of what is to be done, and preconception of what alternatives the computer takes as the result of all possible logical decisions it is asked to make. Of course, the usual give and take is allowed in regard to the testing of ideas.

This section is designed to explain some of the procedures or techniques used in problem definition to simplify communication with the computer. In some cases this amounts to a re-emphasis of standard mathematical techniques.

### Block Diagramming

The old cliché — a picture is worth a thousand words — has particular truth for the engineer in preparing a program for a computer. Since the engineer normally finds many uses for pictorial representations in his work, the block diagram is readily accepted as a powerful tool in computer programming.

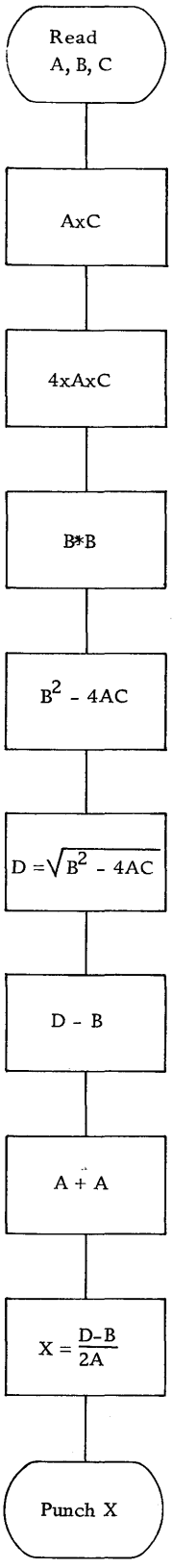
A block diagram is a picture of the steps which must be performed to accomplish a particular job. The major function of the diagram is to clarify what must be done as a result of each decision. All the alternatives are accounted for in each case. In programming a computer it must be remembered that the computer has no way of anticipating the requirements of a problem and must be provided with all the information needed to reach a solution.

The amount of information put into the block diagram, however, will depend upon personal preference, the programming techniques used, and the type of computer to be used. For example, Figure 1 illustrates two ways of block-diagramming the solution of a quadratic equation. The detailed diagram shows every step in the computation and is the type that might be used if the programmer were forced to work in machine language. The second diagram shows only the three main steps in the program but is quite sufficient for a programmer using the FORTRAN language.

To further illustrate, consider a problem which involves the repetition of the same operation over and over. Let the problem be to create the sum

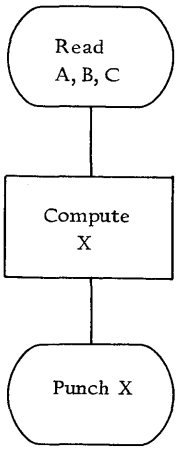
$$Z = \sum_{i=1}^{5} (X_i - Y_i)^2$$

This might be block-diagrammed in either of the two ways illustrated in Figure 2.




A.

DETAILED




B.

GENERAL

Figure 1.

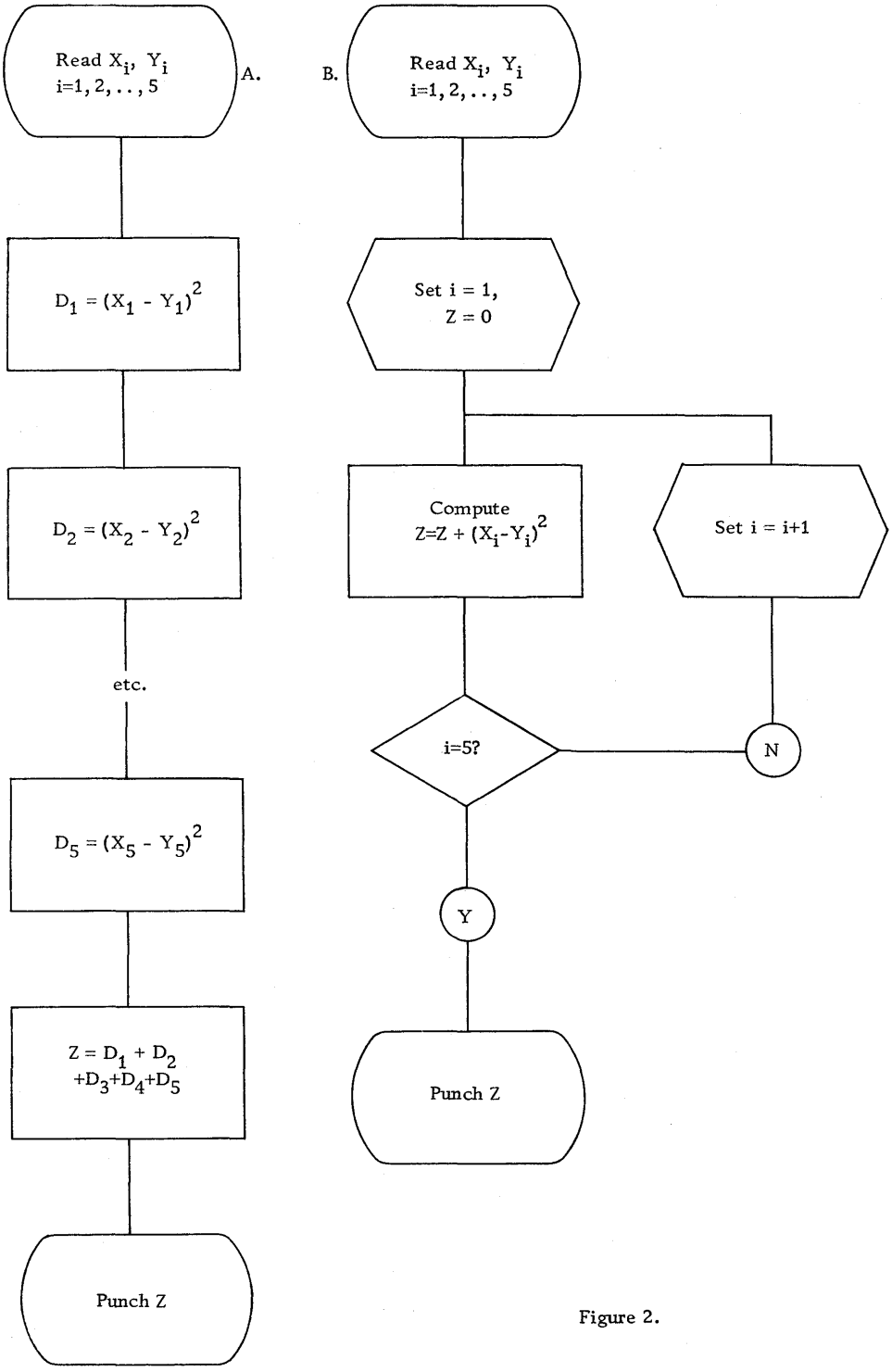

A.
Read $X_i$, $Y_i$
i=1, 2, . . , 5
$D_1 = (X_1 - Y_1)^2$
$D_2 = (X_2 - Y_2)^2$
etc.
$D_5 = (X_5 - Y_5)^2$
$Z = D_1 + D_2 + D_3 + D_4 + D_5$
Punch Z
B.
Read $X_i$, $Y_i$
i=1, 2, . . , 5
Set i = 1,
Z = 0
Compute
$Z=Z + (X_i-Y_i)^2$
Set i = i+1
i=5?
N
Y
Punch Z

Figure 2.

The two diagrams illustrate an important programming concept — that of looping. Diagram A corresponds to a program in which each $(X_i - Y_i)^2$ is computed separately and then all are summed to obtain Z. With only five values of i, this is not too unlikely a method of approaching the problem. But, consider a situation in which i = 100 or 1000 or may vary according to some other characteristic of the problem of which this computation is a part. In such a case not only would the diagram be large and time-consuming to draw, but the program would also be large and time-consuming to write.

In diagram B, advantage has been taken of the computer's ability to make logical decisions and to modify its own program. Since the number of times the computation is repeated depends only on the value of i (and the presence of successive values $X_i$ and $Y_i$), this one diagram — and the program which would be based on it — applies for any value of i.

The algebraically incorrect statement i = i + 1 means replace the current value of i with i + 1. Similarly, $Z = Z + (X_i - Y_i)^2$ means replace the partial sum, $Z_{i-1}$, with the next value $Z_i$. This use of subscripts brings us to our next topic.

## Subscripting

The purpose of subscripting in mathematics is to simplify notation and at the same time to remove ambiguity of meaning. In computer work the subscript is used in two ways:

1. Spacewise — to specify elements of arrays such as:

$$
\begin{array}{lcl}
A_1 & & A_{11}\ A_{12} \cdots A_{1N} \\
A_2 & & A_{21}\ A_{22} \cdots A_{2N} \\
A_3 & \text{or} & A_{31}\ A_{32} \cdots A_{3N} \\
\vdots & & \vdots \quad \vdots \quad \vdots \\
A_M & & A_{M1}\ A_{M2} \cdots A_{MN}
\end{array}
$$

This allows reference to elements of an array through simple manipulation of the subscripts — just as in normal mathematical thought.

2. Timewise — to specify the chronological order in which a procedure occurs. For example, in Figure 2, diagram B, the subscript i is used not only to denote which member of the X and Y array is being operated on, but also to indicate how many <u>times</u> the computational step $Z = Z + (X_i - Y_i)^2$ has been performed. This connotation and use is not so familiar to the engineer. The importance of the timewise aspect will become clear when iteration is discussed.

## Absolute Value

The absolute value becomes important because of the way in which a computer makes logical decisions. Computer decisions are made on the basis of the size of numbers — or, more specifically, upon whether a number is positive, zero or negative. For example, in order to do one of two things dependent upon whether A is (1) less than 500 or (2) equal to or greater than 500, the first thing to do is to subtract 500 from A:

$$A - 500 = E$$

Then the decision is based upon the size of E. Specifically, if E is negative, procedure 1 is done, and if E is positive or zero, procedure 2 is done.

If the difference represents the error in a procedure (that is, 500 is the true value and A is the estimate), the absolute value of the error must be less than a prescribed amount e and the test is:

$$\text{if} \quad |E| - e \geq 0 \qquad \text{do procedure 1}$$

$$\text{or if} \quad |E| - e < 0 \qquad \text{do procedure 2}$$

The absolute value must be used, for in general there is no prior knowledge of whether the difference A-500 will be positive or negative. All the alternatives must be spelled out to the computer in its program.

In many practical cases the "relative" error is a better measure of the error than the absolute error of a result. The relative error test for the above situation would be stated:

$$\text{if} \quad \left|\frac{E}{500}\right| - e \geq 0 \qquad \text{do procedure 1}$$

$$\text{if} \quad \left|\frac{E}{500}\right| - e < 0 \qquad \text{do procedure 2}$$

## Floating Point Arithmetic

Engineering calculations often involve handling of very large numbers such as Young's modulus, which is 30,000,000 lbs. per sq. in. The common procedure is to write such a number as $3.0 \times 10^7$ to facilitate computation.

A similar situation exists in handling very large or very small numbers in a computer. To get away from carrying a great many digits and to eliminate the effort of keeping track of the location of the decimal point for these numbers, a floating point notation is used. A common procedure is to maintain seven or eight most significant digits of a number plus a two-digit "characteristic" to indicate the proper position of the decimal point. The characteristic is developed from the exponent of 10 (assuming use of the decimal system).

For example, the number

$$-.00000061957533$$

can be represented as

$$-.61957533 \times 10^{-6}$$

However, this representation requires carrying two signs: one for the exponent and one for the fraction. This is inconvenient for computers which have only one sign associated with a storage location. Therefore, a common practice is to add the exponent to some positive base.

Using a base of 50, the characteristic becomes 50 + (-06) = 44 and the internal computer representation of the number becomes:

4461957533-

A FORTRAN programmer need not be concerned with this internal representation. The number might appear in a FORTRAN program as

-.61957533E-6

## Errors

There are several sources of errors in computation which deserve consideration in any problem.

Initial error: If x is the true value of a data reading and x* is the reading used in computation (reflecting an error in measurement, perhaps), the initial error is x - x*.

Rounding error: This type of error results when the less significant digits of a quantity are deleted and a rule of correction is applied to the remaining part. For example, pi, 3.14159265..., rounded to four decimals, is 3.1416.

Truncation error: To simply chop off at four decimal places for pi, giving 3.1415, would result in a truncation error. Another common source of truncation error is in chopping off all terms in an infinite series expansion after a particular term. For example, cutting the series for $e^x$ at

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!}$$

gives truncation error — sometimes called residual error for series approximations.

Propagated error: If x is the true value of a variable and x* the value used in computation, then f(x) - f(x*) is the propagated error. Errors may build up in computation!

## Recursion Formulas

Very often computation of a set of numbers can be simplified by formulas which give one member of the set in terms of other members of the set. An example of this is finding the new coefficients of a polynomial when the original polynomial has been reduced by the extraction of a root. If r is

one root of the polynomial equation $a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + \ldots a_o = 0$, then the original polynomial can be divided by $(x - r)$ to obtain the reduced polynomial:

$$x-r\overline{)\,a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + \ldots}\;\;\overline{a_n x^{n-1} + (a_{n-1} + ra_n) x^{n-2} + \left[a_{n-2} + r\,(a_{n-1} + ra_n)\right] x^{n-3} + \ldots}$$

$$\underline{a_n x^n - ra_n x^{n-1}}$$

$$(a_{n-1} + ra_n) x^{n-1} + a_{n-2} x^{n-2} + \ldots$$

$$\underline{(a_{n-1} + ra_n) x^{n-1} - r\,(a_{n-1} + ra_n) x^{n-2}}$$

$$\left[a_{n-2} + r\,(a_{n-1} + ra_n)\right] x^{n-2} + \ldots$$

Let: $c_k x^k + c_{k-1} x^{k-1} + \ldots\ldots + c_1 x + c_0 = 0$

be the resulting polynomial (where $k = n-1$).
Then, equating coefficients gives:

$$c_k = a_n$$

$$c_{k-1} = (a_{n-1} + ra_n) = a_{n-1} + rc_k$$

$$c_{k-2} = \left[a_{n-2} + r\,(a_{n-1} + ra_n)\right] = a_{n-2} + rc_{k-1}$$

or in general

$$c_i = a_{i+1} + rc_{i+1}$$

This is a recursion formula for $c_i$. This formula has obvious advantages for programming the calculation of the new coefficients.

## Algorithms

An algorithm is a theorem which may state that a solution to a problem exists, and/or a procedure for obtaining the solution. The term is frequently encountered in computer literature because the form of an equation is often all-important in programming efficiently.

Two examples will be given here:

1. To compute the value of the polynomial

   $$f(x) = a_0 x^5 + a_1 x^4 + a_2 x^3 + a_3 x^2 + a_4 x + a_5$$

   computation is simplified if it is written as

   $$f(x) = (((((a_0) x + a_1) x + a_2) x + a_3) x + a_4) x + a_5$$

   which indicates that the polynomial may be computed by repeating the steps "multiply and add" five times.

2. The square root of a positive real number, A, can be computed by using the algorithm

$$x_{i+1} = \frac{1}{2}(x_i + \frac{A}{x_i})$$

This example also introduces the idea of iteration. What is meant by this equation is that to obtain an estimate of $A^{\cdot 5}$, start with a first guess $x_0$. Substitute it in the right side of the equation to obtain the next estimate of $A^{\cdot 5}$. A few of the steps are:

$$x_1 = \frac{1}{2}(x_0 + \frac{A}{x_0})$$

$$x_2 = \frac{1}{2}(x_1 + \frac{A}{x_1})$$

$$x_3 = \frac{1}{2}(x_2 + \frac{A}{x_2})$$

$$\vdots$$

$$x_{i+1} = \frac{1}{2}(x_i + \frac{A}{x_i})$$

Note that the algorithm states a procedure for solution of the problem, not just one formula evaluation.

Further, note the timewise use of the subscript i; i + 1 indicates a result dependent upon the previous result subscripted by i.

Taking a value of A (say 25) for which the square root is known, and performing the indicated operations will make this algorithm clear. If 2 is used as a starting estimate, then;

$$x_o = 2$$

$$x_1 = \frac{1}{2}\left(2 + \frac{25}{2}\right) = 7.25$$

$$x_2 = \frac{1}{2}\left(7.25 + \frac{25}{7.25}\right) \cong 5.35$$

$$x_3 = \frac{1}{2}\left(5.35 + \frac{25}{5.35}\right) \cong 5.01$$

There are several ways of deciding the point at which to discontinue the iteration:

a. When differences between successive estimates are less than some $\epsilon$ of accuracy $|x_{i+1} - x_i| < \epsilon$

b. When relative differences between successive estimates are less than some $\epsilon$ of accuracy $\left|\frac{x_{i+1} - x_i}{x_i}\right| < \epsilon$

This makes the choice of $\epsilon$ independent of the size of the original function value A.

c. When relative differences between the function of the estimate and the original function value A are less than some $\epsilon$ of accuracy

$$\left| \frac{(x_{i+1})^2 - A}{A} \right| < \epsilon$$

This eliminates the possibility of an erroneous solution in a situation where convergence is slow and is the preferred type of test.

The general statement for this test in finding a value of x which satisfies

$$f(x) = A$$

is

$$\left| \frac{f(x_{i+1}) - A}{A} \right| < \epsilon$$

In succeeding chapters the problem of solving for values of x which satisfy equations of the form $x = f(x)$ are discussed. Methods a and b remain the same but the test equation corresponding to c above is

$$\left| \frac{x_{i+1} - f(x_{i+1})}{f(x_{i+1})} \right| < \epsilon$$

## Complex Number Calculations

Complex numbers can create some questions of handling and yet are of great importance to electrical and electronic engineers. The following rules handle complex arithmetic:

A. Addition

$$(a+bi) + (c+di) = (a+c) + (b+d)\,i$$

B. Multiplication

$$(a+bi)\,(c+di) = (ac-bd) + (ad+bc)\,i$$

C. Division

$$\frac{a+bi}{c+di} = \frac{(a+bi)}{(c+di)}\frac{(c-di)}{(c-di)} = \frac{(ac+bd)}{(c^2+d^2)} + \frac{(bc-ad)}{(c^2+d^2)}\,i$$

D. Square Root

If we let $u+vi = (a+bi)^{\frac{1}{2}}$

squaring both sides gives $u^2 + 2uvi - v^2 = a+bi$

Equating the real and imaginary parts

(1) $u^2 - v^2 = a$

(2) $2uv = b$ or $v = \frac{b}{2u}$

then substituting from (2) into (1)

$$u^2 - \frac{b^2}{4u^2} - a = 0$$

and multiplying through by $u^2$ gives

$$u^4 - au^2 - \frac{b^2}{4} = 0$$

and solving the quadratic gives

$$u^2 = \frac{a}{2} \pm \left[ \left(\frac{a}{2}\right)^2 + \left(\frac{b}{2}\right)^2 \right]^{\frac{1}{2}}$$

Only the positive sign is considered since u is real; thus

(3) $$u = \left[\frac{a}{2} + \left(\left(\frac{a}{2}\right)^2 + \left(\frac{b}{2}\right)^2\right)^{\frac{1}{2}}\right]^{\frac{1}{2}}$$

and (4) $$v = \left[\frac{-a}{2} + \left(\left(\frac{a}{2}\right)^2 + \left(\frac{b}{2}\right)^2\right)^{\frac{1}{2}}\right]^{\frac{1}{2}}$$

If a is positive, compute u by (3) and use (2) to find v.

If a is negative, compute v by (4) and use (2) to find u.

E. For a general power of a complex number use

$(a+bi)^n = r^n e^{in\theta}$ where $r = (a^2+b^2)^{\frac{1}{2}}$

$$\theta = \tan^{-1}\frac{b}{a}$$

and $e^{in\theta} = (\cos\theta + i\sin\theta)^n = \cos n\theta + i \sin n\theta$

then $(a+bi)^n = r^n (\cos n\theta + i \sin n\theta)$

## Transformations and Identities

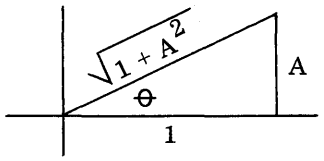
Recognition of mathematical transformations and identities which are possible may simplify computation.

For example, assume in a particular problem that the value for the tangent of an angle is given:

$$\tan \Theta = A$$

and that $\cos \Theta$ is to be computed. One approach would be to find $\Theta = \tan^{-1} A$ and then compute the $\cos \Theta$. However, the cosine may immediately be computed as

$$\cos \Theta = \frac{1}{\sqrt{1 + A^2}}$$




saving the evaluation of two trigonometric functions.

A useful transformation in computing is the logarithmic transformation. Suppose we have an exponential function $y = k x^a$. If we take the natural log of both sides, we have

$$\log y = a \log x + \log k$$

thus reducing an exponential relationship between x and y to a linear relationship between the transformed variables log x and log y.

Most of the mathematical techniques which have been outlined here are quite familiar to the average engineer. However, they are given this emphasis because recognition of their use in particular situations can greatly simplify solution of computer problems.

## Exercises

1. For the set of tabular data shown, construct the difference table to determine to what degree of x is y dependent.

| x | y |
|---|---|
| 0 | 0 |
| 1 | 3 |
| 2 | 18 |
| 3 | 57 |
| 4 | 132 |
| 5 | 255 |
| 6 | 438 |

2. Divide 8 + 12i by 3 + 2i.

3. Using the iteration formula for finding the square root of a number, find the square root of 12 accurate to 7 places (the absolute difference between two successive estimates of $\sqrt{12}$ is less than .00000005).

*4. Using only the functions SINF(X), CΦSF(X), ATANF(X), EXPF(X), LΦGF(X) and SQRTF(X), write the FORTRAN statement for computing all trigonometric, inverse trigonometric and hyperbolic functions.

5. Construct a block diagram chart to show the logic of computing the value of the polynomial

$$f(x) = a_5x^5 + a_4x^4 + a_3x^3 + a_4x^2 + a_1x + a_0$$

where the values of the coefficients $a_1$, $a_2$, _ _ _ _ _ $a_6$ are to be read into the computer and the value of f (x) is to be punched out.

*6. Repeat 5 and generalize the block diagram to show the logic of handling any degree polynomial. That is, block-diagram the computation of f (x), where

$$f(x) = a_1x^n + a_2x^{n-1} \text{ _ _ _ _ _ } + a_{n+1}$$

7. Write a FORTRAN program for exercise 6.

8. The block diagram in Figure 2 is incomplete. Why?

## SECTION IV: PRINCIPLES OF ITERATION

### Arithmetic, Mathematics and the Computer

Although arithmetic is one class of mathematical operation, for purposes of clarity it is best to differentiate arithmetic operations from the other mathematical operations in discussing computers. Arithmetic is performed when any of the operations of addition, subtraction, multiplication and division are applied to a pair of numbers. Other mathematical operations are performed when one of a variety of operations such as algebraic substitution, integration, differentiation, expansion to series representation, etc., is applied to one or more symbols. In essence arithmetic deals with numbers while mathematics is the manipulation of symbols.

Computers are designed to efficiently perform all the arithmetic operations. Most computers can also handle in a limited way the alphabetic and some special characters which might be used as symbols (as is obvious from the use of the computer to translate FORTRAN programs).

The computer, as designed, cannot do original mathematics. However, once the mathematics of a problem are accomplished, the computer is most useful in obtaining the required numerical values by performing the necessary arithmetic operations.

In addition, once a particular mathematical problem has been properly solved and the program for arithmetic evaluation on a computer has been completed, all similar problems, differing only in data values, can be henceforth evaluated by the one program. Two examples will illustrate these facts:

A. Given the two simultaneous equations

$$a_{11}x_1 + a_{12}x_2 = b_1 \qquad (1)$$

$$a_{21}x_1 + a_{22}x_2 = b_2 \qquad (2)$$

find values of $x_1$ and $x_2$ which satisfy these equations when

$$a_{11}=2,\ a_{12}=2,\ a_{21}=3,\ a_{22}=4,\ b_1=5,\ b_2=6$$

To solve this problem we have distinct mathematical and arithmetic steps.

Mathematically:

Algebraically using (1) write $x_1$ as a function of $x_2$

$$x_1 = f_1(x_2) = \frac{b_1 - a_{12}x_2}{a_{11}} \qquad (3)$$

Substitute this function $f_1(x_2)$ in the second equation to obtain an equation in $x_2$

$$\frac{-a_{21}\,a_{12}}{a_{11}}\ x_2\ \frac{+\,a_{21}\,b_1}{a_{11}}\ + a_{22}\,x_2 = b_2 \qquad (4)$$

This equation may then be treated algebraically to give $x_2$ as a function of the coefficients only

$$x_2 = \left(b_2 - \frac{a_{21}\,b_1}{a_{11}}\right) \Big/ \left(a_{22} - \frac{a_{21}\,a_{12}}{a_{11}}\right)$$

Arithmetically:

Substitute the values given for the coefficients in equation (5) to determine $x_2$

$$x_2 = \left(6 - \frac{3.5}{2}\right) \Big/ \left(4 - \frac{3.2}{2}\right) = -1.5$$

Use this result in equation (3) to find a value for $x_1$

$$x_1 = \frac{5 - 2(-1.5)}{2} = 4$$

The computer solution to this problem considers only the arithmetic operations implied by equations (3) and (5). A FORTRAN program to evaluate these two equations is shown in Figure 3.

C FOR COMMENT

| STATEMENT NUMBER 1 5 | Cont. 6 | FORTRAN STATEMENT 7 72 |
|---|---|---|
| 1 | | DIMENSION A(2,2),B(2),X(2) |
| 4 | | READ 10, K, A, B |
| 5 | | X(2) = (B(2)-A(2,1)*B(1)/A(1,1))/(A(2,2)-A(2,1)*A(1,2)/A(1,1)) |
| 6 | | X(1) = (B(1)-X(2)*A(1,2))/A(1,1) |
| 7 | | PRINT 20, K, X |
| 8 | | 60 TO 4 |
| 9 | | END |

Figure 3.

NOTE the following items of interest concerning this program:

1. Only the solution equations have been programmed. The original equations are not of interest to the computer since the mathematics must be performed before using the computer.

2. Equations (3) and (5) are programmed in their symbolic form and actual values for the coefficients are read into the computer before computation takes place. By programming in this way the computer can now evaluate the solution of any set of two linear equations in two variables,

in addition to the particular example discussed. In a sense, by our programming, we give the computer the capability of doing nonoriginal mathematics.

B. Solve the following differential equation with stated initial conditions for y when x is equal to 1:

$$\frac{dy}{dx} = xy \qquad (1)$$

$$y = 1 \text{ when } x = 0$$

Mathematically:

Integrate (1) for y as a function of x

$$\frac{dy}{dx} = x\,y$$

$$\frac{dy}{y} = x\,dx$$

$$\ln y = \frac{x^2}{2} + k$$

$$y = e^{\left(\frac{x^2}{2} + k\right)}$$

Considering the initial condition y = 1 when x = 0, then

$$k = 0$$

Arithmetically:

Since the analytic solution is $y = e^{\frac{x^2}{2}}$ it is now necessary only to substitute the value of x for which y is to be determined and perform the indicated arithmetic.

$$y = e^{\frac{1}{2}} = 1.64872$$

A FORTRAN program to accomplish the required calculations is shown in Figure 4.

C FOR COMMENT

| STATEMENT NUMBER 1–5 | Cont. 6 | FORTRAN STATEMENT 7–72 |
|---|---|---|
| | | READ 1,0, XMIN, DELX, XMAX |
| | | X = XMIN |
| 3 | | Y = EXPF(X**2/2.) |
| | | PRINT 20, X, Y |
| | | IF(X-XMAX)5,6,6 |
| 5 | | X = X + DELX |
| | | GO TO 3 |
| 6 | | STOP |
| | | END |

Figure 4.

NOTE the following:

1. Again only the analytic solution is programmed.

2. The program is written so as to allow many values of y to be computed over any range in x.

## Numerical Analysis

The two problems chosen happen to be of types for which the mathematical analysis is well defined. The variables in question may be equated to functions of the known quantities — analogous to the formula $y = f(x, z)$ as discussed in the first chapter.

Various analytic techniques successfully handle a large class of the mathematical problems which arise in the description of the physical world. However, a vastly larger number of mathematical problems are not subject to the normal analytic procedures of algebra, analytic geometry, or the calculus.

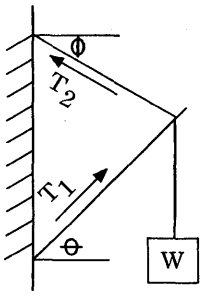
For the theoretical mathematician or physicist it may be sufficient, when meeting these problems, to be able to show that one or more approximations will allow an analytic solution of the problem to any desired degree of accuracy.

The engineer, on the other hand, must have a solution from which accurate numerical results can be obtained if the theory is to be of value. These results must be obtained at a reasonable expense of time and effort. A mathematician may be content to know that a series will converge. An engineer must be concerned with the number of terms in the series required for a given accuracy.

Numerical analysis provides techniques for finding concrete numerical results for mathematical problems of the type under discussion. This manual describes many physical problems which require numerical analysis. Some of the simpler analysis techniques are discussed. The reader is referred to the many excellent texts available on numerical analysis for an adequate discussion of particular analysis theory.

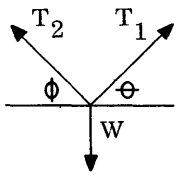
## Algebraic Equation

Problem A belongs to a very special class of algebraic problems, a set of n nonhomogeneous linear equations in n variables. Fortunately, a large class of engineering problems such as the force and moment problems of stress analysis give rise to this simply handled type of equation. The diagram below shows a simple force equation example giving rise to a set of two linear equations.




$$\sum F_x = 0$$

$$\sum F_y = 0$$




$$T_1 \cos\theta - T_2 \cos\phi = 0$$

$$T_1 \sin\theta + T_2 \sin\phi = W$$

$T_1$ and $T_2$ are readily solved for specified values of $\theta$, $\phi$ and W.

Consider a set of equations similar to A but containing a simple nonlinearity:

$$a_{11} x_1 + a_{12} x_2 = b_1 \qquad (1)$$

$$a_{21} x_1 + e^{a_{22}x_2} = b_2 \qquad (2)$$

The previous algebraic steps would give:

$$x_1 = -\frac{a_{12}}{a_{11}} x_2 + \frac{b_1}{a_{11}} \qquad (3)$$

and:

$$x_2 = \left(b_2 - \frac{a_{21}b_1}{a_{11}} - a_{22}e^{x_2}\right) \Big/ \left(-\frac{a_{21}a_{12}}{a_{11}}\right) \qquad (4)$$

or with appropriate substitutions for (4)

$$x_2 = ae^{bx_2} + c \qquad (5)$$

This gives an equation in $x_2$ only, but from the equation it is impossible to directly evaluate $x_2$. One approach is to expand $e^{bx2}$ to several terms of its series:

$$e^{bx_2} = 1 + \frac{bx_2}{1} + \frac{(bx_2)^2}{2!} + \frac{(bx_2)^3}{3!} + \ldots.$$

However, for sufficient accuracy it may be necessary to carry a large number of terms in the series resulting in a polynomial equation being of degree greater than 4. In such a case, the equation is again impossible to evaluate directly. This chapter will discuss handling equation (5) and polynomials by means of iteration.

## Equations of Calculus

Similarly the differential equation of example B has an analytic solution. However, the differential equation

$$\frac{dy}{dx} = x^2 + y^2$$

has no corresponding analytic solution. To find values of y for corresponding values of x, a numerical approach is necessary. Likewise, many integrals may be obtained only by numerical approximations. Again, some simple techniques will be given for handling this type of problem.

## Iteration Type Problems

There are three principal reasons for the use of iteration with computer handling of engineering problems.

1. The mathematical statement of the physical problem requires an iteration approach for evaluating one or more of the variables.

The chart below shows examples of four types of mathematical problem statements.

| Single Equations | Multiple Equations | |
|---|---|---|
| $x = \frac{1}{5} e^{x/2}$ | $M_H = A_H - m$<br>$M_{He} = A_{He} - 2m$<br>$m = \frac{R_{He} - R_H}{R_{H/M_H} - R_{He/M_{He}}}$ | Direct Iteration Sufficient |
| $\phi = \tan\phi - K$ | $B = 90^\circ - \gamma - \sin^{-1}\left(\frac{L + R + E}{A}\right)$<br>$L = A \sin(90^\circ - B - \gamma - \alpha) - (R-E)$ | Numerical Techniques Required |

2. Many designs are to be evaluated in a search for the best design, that is, optimization of design. Often this problem can be reduced to the previous problem by selection of an appropriately expressed mathematical criterion of what is optional.

3. The mathematical expression for the physical problem is to be evaluated for many values of one or more known parameters — such as time, in a problem of motion, or degree of rotation, in a geometry problem.

A. One problem which illustrates iteration is the solution of an equation which arises frequently in absorption problems of optics, electric fields and nuclear engineering:

$$x = ae^{bx} \qquad (1)$$

where a and b are constants.

This cannot be solved explicitly for x, so the following iteration procedure is used:

1. Make a guess at x.
2. Use this in right-hand side of equation (1) to give a new value for x.
3. Call this new value of x the next guess.
4. Repeat steps 2 and 3 until two successive guesses either agree or differ by an amount less than the allowable error.

A FORTRAN program to solve such a problem, where

$$x = 0.2e^{0.5x}$$

is as follows:

| | |
|---|---|
| READ 10, A, B, X | Read values for A, B and initial estimate of x (1.0). |
| 1 PRINT,X | Print estimate of x. |
| 2 XNEW=0.2*EXPF(0.5*X) | FORTRAN arithmetic statement of equation to be solved. |
| TEST=ABSF(X-XNEW) | Find absolute value of difference in last two estimates. |
| IF(TEST-.00005)4,4,3 | If difference is less than or equal to .00005, go to step 4. Otherwise, go to step 3. |
| 3 X=XNEW | Store new estimate of x. |
| GO TO 1 | Return to step 1 and repeat. |
| 4 PRINT, XNEW | Print last estimate of x. |
| END | End of program. |

If this were translated into machine language and the resulting program run on a computer, the successive estimates of x would be:

| X |
|---|
| 1.000000 |
| .329744 |
| .235848 |
| .225032 |
| .223818 |
| .223682 |
| .223667 |

The last value in the list satisfies the requirement that the difference between it and the previous estimate be less than 0.00005 in absolute value. Hence it is the solution.

B. The simple iteration procedure in problem A seems like a nice way to solve problems — and in many cases it is. However, on application to problem B — much like problem A except that a constant has been changed — it fails:

$$f(x) = x - Ae^{Bx} = 0 \qquad A = .2,\ b=3 \qquad (1)$$

The reason for the failure of convergence of the direct iteration technique should be explained. Direct iteration has the formula

$$x_{n+1} = f(x_n)$$

This indicated procedure will converge only for

$$-1 < f'(x) < 1$$

where $f(x) = A\,e^{Bx}$ for this problem, and the prime indicates the first derivative. The choice of a larger B makes $f'(x) > 1$ and therefore makes convergence by the direct procedure impossible.

Since simple iteration fails to produce convergence, a different method to obtain an estimate must be found. The Newton-Raphson method answers this need. In this instance, successive estimates are obtained from

$$x_{m+1} = x_m - \frac{F(x_m)}{F'(x_m)}$$

From equation (1):

$$F(x_m) = x_m - Ae^{Bx_m}$$

and

$$F'(x_m) = 1 - BAe^{Bx_m}$$

Applying the Newton-Raphson formula:

$$x_{m+1} = x_m - \frac{x_m - Ae^{Bx_m}}{1 - BAe^{Bx_m}}$$

To solve this problem in FORTRAN, replace statement 2 in the previous FORTRAN program with the arithmetic statement for (3):

```
2  XNEW=X- (X-A*EXPF(B*X)) /(1-A*B*EXPF(B*X))
```

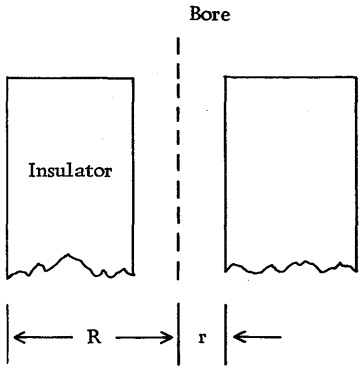
If this FORTRAN program were translated into a machine-language program and run on a computer, the successive values of x would be:

1.00000
.19742
.22364
.22366

Note the increased rate of convergence obtained using this new method; only three new estimates are computed, compared with seven in the previous example.

C. The following engineering problem will illustrate one origin of the type of equation just discussed. The analysis makes use of the classical optimization principle of equating the derivative of a function of one variable to zero. The values of the variable which satisfy the resulting equation are those for which the original function is either a maximum or a minimum. (Since engineers are often interested in optimization of design, it is unfortunate that so few design problems can be expressed in a form suitable for the use of this principle.)

For high-potential conducting through walls, tubular insulators are used which are covered with metal on the inner and outer surfaces.




What must be the ratio of the external diameter, 2R, to the bore width, 2r, for the cross section Q to be a minimum?

The ratio q of the line voltage to the maximum admissible field strength is given by

$$q = r \ln R/r \qquad (1)$$

The cross section Q is given by

$$Q = \pi (R^2 - r^2) \qquad (2)$$

According to (1), letting $x = R/r$, then

$$r = q/\ln x$$

or

$$Q = \pi q^2 (x^2 - 1)/(\ln x)^2 \qquad (3)$$

Since Q is to be a minimum, it is necessary to find a point on the curve defined by (3) where the slope is zero. To do this we take the first derivative and set it equal to zero:

$$\frac{dQ}{dx} = \pi q^2 \left[\frac{2x}{(\ln x)^2} - \frac{2(x^2 - 1)}{x(\ln x)^3}\right] = 0 \qquad (4)$$

From the bracketed expression:

$$x - (x^2 - 1)/x \ln x = 0$$

or

$$\ln x = 1 - 1/x^2$$

This in turn gives

$$x - e^{1-1/x^2} = 0 \qquad (6)$$

The solution for x is readily recognizable as the type of problem just discussed.

D. Another example can be taken from an equation which occurs when working with helical gears. This is

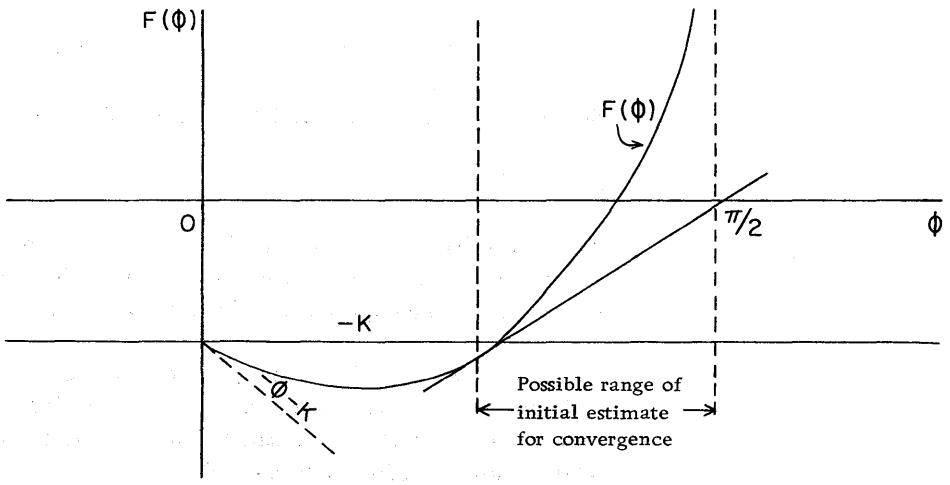
$$K = \text{involute of } \phi = \tan \phi - \phi \qquad (1)$$

where K is found in a table and it is necessary to find an angle $\phi$ satisfying the equation.

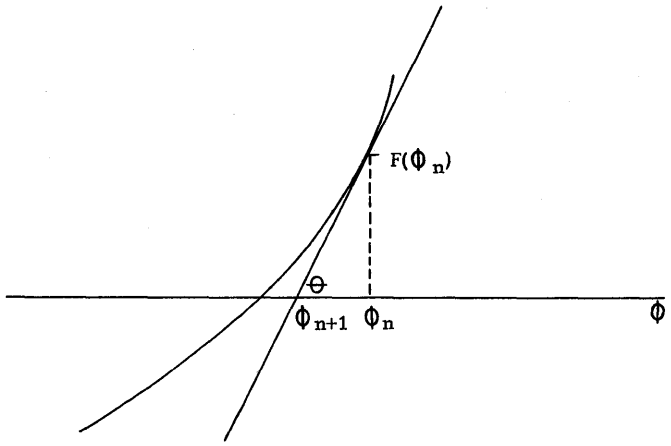
The Newton-Raphson method can again be used but it is worthwhile to stop and consider a good way to get a first estimate of $\phi$ . Since the rate of convergence — and in some cases, the possibility of converging — will depend on the initial guess, this is a subject worthy of some thought.

The following graphic presentation of this problem will clarify the situation.

Plotting Newton's function: $F(\phi) = \tan \phi - \phi - K$ gives the curve as shown:



Enlarging the area of intersection of F ($\phi$) with the $\phi$ axis shows the basis for Newton's procedure.




If $\phi_n$ is the first estimate of the root, the figure indicates that the next good estimate would be the intersection of the projected slope of F ($\phi$) evaluated for $\phi_n$. The following use of the construction angle $\theta$ shows this to be precisely Newton's procedure:

$$\tan\theta = \frac{F(\phi_n)}{\phi_n - \phi_{n+1}} = F'(\phi_n)$$

or

$$\phi_{n+1} = \phi_n - \frac{F(\phi_n)}{F'(\phi_n)}$$

The dotted lines in the figure show the limits on the range of the initial estimate for which Newton's technique will converge to the proper value.

A good first estimate can often be determined by a careful examination of the problem to be solved. In this case the first two terms of the trigonometric series for tan $\phi$ give:

$$\tan\phi \cong \phi + \frac{\phi^3}{3} \qquad (2)$$

Substituting this in (1) and solving for $\phi$ gives

$$\phi \cong (3K)^{1/3}$$

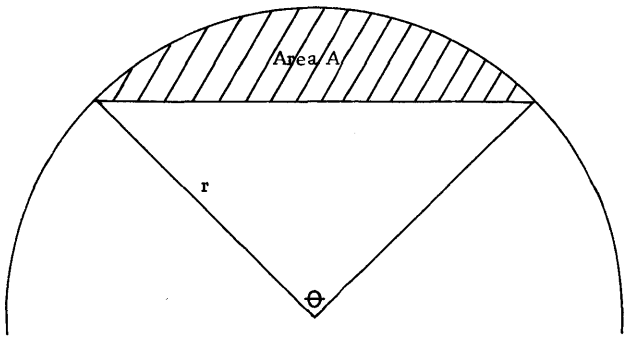
Thus an initial guess which should be reasonably close to the solution can be computed from the given value of K. This can be an important advantage where convergence is likely to be slow.

To illustrate the effect, the problem has been solved for two values of K (0.001 and 0.01). In each case, the initial estimate of $\phi$ = 0.52359 radians (30°) was used rather than a value computed as described above. The results obtained are as follows:

| $\phi$ (K = .001) | $\phi$ (K = .01) |
|---|---|
| .52359 | .52359 |
| .36535 | .39235 |
| .25482 | .32546 |
| .18624 | .30818 |
| .15296 | .30702 |
| .14440 | .30701 |
| .14385 | |
| .14372 | |

Note that for K = .01 fewer iterations were required because the initial estimate was closer to the correct result. Had the approximation formula developed above been used, the initial estimates would have been 0.14423 and 0.31072 respectively, and no more than two iterations would have been required in either case.

E. A simple geometry problem which lends itself to iterative solution is that of finding the angle for which the arc and chord in a circle of given radius will enclose a stated area.




Area A is given by

$$A = \pi \frac{r^2 \theta}{360} - \frac{r^2 \sin\theta}{2} \qquad (1)$$

Since $\theta$ cannot be solved for directly, the Newton-Raphson technique is used to find a solution. Converting $\theta$ to radians and applying the N-R formula gives the iteration equation:

$$\theta_{n+1} = \theta_n - \frac{F(\theta_n)}{F'(\theta_n)}$$

where

$$F(\theta) = \frac{r^2 \theta_n}{2} - \frac{r^2 \sin\theta_n}{2} - A$$

and

$$F'(\theta) = \frac{r^2}{2} - \frac{r^2 \cos\theta_n}{2}$$

It is interesting to note how frequently an iterative problem is given rise to by asking an unfamiliar question to a familiar relationship. Generally equation (1) is used to determine the area A. In a sense, iteration arises when the "answer" is given and the "question" is to be determined.

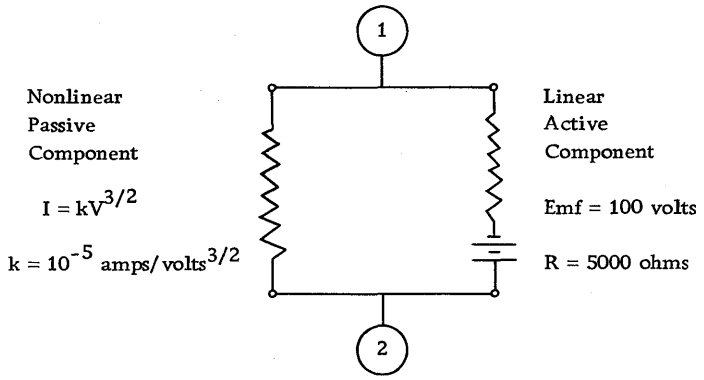
F. Very often iteration can be the most convenient procedure in finding a root of a cubic such as

$$ax^3 + bx^2 + cx + d = 0$$

Newton's technique will generally suffice here.

For polynomials of degree greater than 4, some type of iterative procedure must be used. There are many techniques available to handle the higher-degree polynomials, but they will not be discussed here. One of the important advantages of the FORTRAN language is that it makes it more readily possible for computing organizations to establish standard programs for such problems and exchange them.

G. This problem is an example of how a simple circuit can give rise to a cubic equation. Generally it is simpler to make successive approximations than to solve the cubic.




Problem: Given the circuit shown, find the potential difference V from (1) to (2) and the current I flowing in the circuit.

Using the basic relationship for the potential drop,

$$V = E - IR$$

and considering the active component, gives

$$V = 100 - 5000kV^{3/2}$$

or

$$V = 100 - .05V^{3/2}$$

a cubic equation easily solved using Newton's technique.

H. An interesting example of the use of iteration can be taken from the computations employed in spectroscopy. Spectroscopy and astronomy are two areas of science in which it is meaningful to speak of numbers accurate to seven or eight significant figures. The success of iterative approximations in the field of spectroscopy demonstrates beyond question its possibilities for accuracy.

Given experimentally determined values of the Rydberg constants for hydrogen ($R_H$) and helium ($R_{He}$) and the atomic mass ($A_H$, $A_{He}$) of the elements, it is possible to compute the mass of the electron.

Assume the following data is given

$$R_H = 109,677.68 \text{ cm}^{-1} \qquad A_H = 1.00812$$

$$R_{He} = 109,722.34 \text{ cm}^{-1} \qquad A_{He} = 4.00388$$

if $R_\infty$ is the Rydberg constant assuming the mass of the nucleus to be infinite, the following relationships exist where m is the mass of the electron:

$$R_\infty = \left(1+\frac{m}{M_H}\right) R_H \qquad (1)$$

$$R_\infty = \left(1+\frac{m}{M_{He}}\right) R_{He} \qquad (2)$$

The atomic mass includes mass of the electron:

$$A_H = M_H + m \qquad (3)$$

$$A_{He} = M_{He} + 2m$$

This gives a useful first approximation

$$M_H \cong A_H \qquad (4)$$

$$M_{He} \cong A_{He}$$

Equating (1) and (2) gives a first evaluation of m:

$$m = \frac{R_{He} - R_H}{\frac{R_H}{M_H} - \frac{R_{He}}{M_{He}}} \qquad (5)$$

To obtain subsequent approximations, equation (3) is rewritten as:

$$M_H = A_H - m \qquad (6a)$$

and

$$M_{He} = A_{He} - 2m \qquad (6b)$$

and, using the new values m is re-evaluated by (5).

A FORTRAN program for this procedure is:

```
 20  0 EMASS=0.0
 30  0 RH=109677.68
 40  0 RHE=109722.34
 50  0 AH=1.00812
 60  0 AHE=4.00388
 70  0 EPSLN=.00000005
  1  0 EMH=AH-EMASS
 80  0 EMHE=AHE-2.0*EMASS
 90  0 EMASP=(RHE-RH)/(RH/EMH-RHE/EMHE)
100  0 PUNCH 20,EMASP,EMH,EMHE
110  0 IF(ABSF(EMASS-EMASP)-EPSLN)3,3,2
  2  0 EMASS=EMASP
120  0 GO TO 1
  3  0 END
```

Running this program on a computer produces the following results:

| m | $M_H$ | $M_{He}$ |
|---|---|---|
| .00054871426 | 1.0081200 | 4.0038800 |
| .00054836568 | 1.0075713 | 4.0027826 |
| .00054836592 | 1.0075716 | 4.0027833 |

It is significant to note that the simplest iterative technique produces the necessary accuracy for this application in only three iterations.

I. Earlier it was stated that iteration could be used in obtaining numerical solutions to differential equations. To illustrate, consider a problem for which the exact integral is known:

$$\frac{dy}{dx} = xy \qquad (1)$$

and find y as x varies from 0 to 1.0. The initial conditions are that for $x = 0$, $y = 1.0$.

Since discrete values of dx and dy must be used, it is necessary that these be replaced with the approximations:

$$\Delta x \cong dx$$

and

$$\Delta y \cong dy$$

If a fixed value is assigned to $\Delta x$ in the range 0 to 1.0, then:

$$\begin{aligned} x_0 &= 0 \\ x_1 &= x_0 + \Delta x \\ x_2 &= x_1 + \Delta x \\ &\vdots \\ x_{n+1} &= x_n + \Delta x \\ &\vdots \end{aligned} \qquad (2)$$

To find the values of $\Delta y$, equation (1) is changed to the approximation:

$$\Delta y_n = x_{n+1} y_n \Delta x \qquad (3)$$

Then,

$$y_{n+1} = y_n + \Delta y_n \qquad (4)$$

Repeated evaluation of equations (2), (3) and (4) will give the solution:

```
3 DELX=.05
4 X=0.0
5 Y=1.0
1 PRINT 20,X,Y
6 X=X+DELX
7 DELY=X*Y*DELX
8 Y=Y+DELY
9 IF(X-1.0)1,1,2
2 END
```

Taking $\Delta x = .05$, the following solution was obtained

| x | y |
|---|---|
| .00000 | 1.00000 |
| .05000 | 1.00250 |
| .10000 | 1.00751 |
| .15000 | 1.01507 |
| .20000 | 1.02522 |
| .25000 | 1.03803 |
| .30000 | 1.05361 |
| .35000 | 1.07204 |
| .40000 | 1.09348 |
| .45000 | 1.11809 |
| .50000 | 1.14604 |
| .55000 | 1.17756 |
| .60000 | 1.21288 |
| .65000 | 1.25230 |
| .70000 | 1.29613 |
| .75000 | 1.34474 |
| .80000 | 1.39853 |
| .85000 | 1.45796 |
| .90000 | 1.52357 |
| .95000 | 1.59594 |
| 1.00000 | 1.67574 |

The analytic solution to (1) for the stated boundary conditions is

$$y = e^{\frac{x^2}{2}}$$

and for $x = 1.0$, $y = 1.64872$. The error in the numerical solution is $1.67574 - 1.64872 = 0.02702$, or less than 2%. Considering the size of the interval, this is not out of line. Increased accuracy could be obtained

by decreasing $\Delta x$ or by using a more sophisticated technique — and there are many.

Consideration of the first alternative should take into account that decreasing the value of $\Delta x$ may increase the rounding error and hence make it necessary to carry along a greater number of significant digits. The second alternative must be viewed in the light of the accuracy required. The more sophisticated techniques will require greater programming effort and more computer time. It is not reasonable to spend this time and effort to obtain accuracy of .01% if the required accuracy is only 2% — particularly in cases where the input data is accurate to, say, 5%. These points should be considered in programming any problem for a computer if the most economical use is to be realized.

J. It is easy to extend the simple numerical integration procedure just discussed to higher-order differential equations. For example, suppose the following is to be solved over the interval 0 to 1.0:

$$\frac{d^2y}{dx^2} + A\frac{dy}{dx} + Bxy = 0$$

where $x_0 = 0$, $y_0 = 1.0$, $(dy/dx)_0 = 1$, $A = 2$ and $B = 3$.

The analytic solution must be in terms of an infinite series. Therefore, whether a series is determined or numerical integration is performed, the solution involves considerable computation. The series representation allows control of the error while the stepwise integration procedure to be described may not.

An approach to this problem is to start with

$$\left(\frac{d^2y}{dx^2}\right)_0 = -A(dy/dx)_0 - Bx_0y_0$$

and then obtain successive values of each of the variables from

$$y_{n+1} = y_n + (dy/dx)_n \Delta x$$

$$x_{n+1} = x_n + \Delta x$$

$$(dy/dx)_{n+1} = (dy/dx)_n + (d^2y/dx^2)_n \Delta x$$

$$(d^2y/dx^2)_{n+1} = -A(dy/dx)_{n+1} - Bx_{n+1}y_{n+1}$$

A computer solution to this problem is shown in Figure 5.

* * * *

In the first few problems just presented, iteration was used to improve the estimate of a single solution. In the case of the differential equations, iteration provided successive values of y for finite changes in x. These are two entirely different concepts of iteration.

```
 1 0 A=2.0
 2 0 B=3.0
 3 0 XN=0.0
 4 0 YN=1.0
 5 0 DYDX=1.0
 6 0 D2YDX=-A*DYDX-B*XN*YN
 7 0 PRINT 20, XN,YN,DYDX,D2YDX
 8 0 IF(XN-1.0)9,14,14
 9 0 YN=YN+DYDX*.05
10 0 XN=XN+.05
11 0 DYDX=DYDX+D2YDX*.05
12 0 D2YDX=-A*DYDX-B*XN*YN
13 0 GOTO 7
14 0 END
```

| XN | YN | DYDX | D2YDX |
|---|---|---|---|
| .00000000 | 1.0000000 | 1.00000000 | -2.0000000 |
| .05000000 | 1.0500000 | .90000000 | -1.9575000 |
| .10000000 | 1.0950000 | .80212500 | -1.9327500 |
| .15000000 | 1.1351063 | .70548750 | -1.9217729 |
| .20000000 | 1.1703807 | .60939886 | -1.9210261 |
| .25000000 | 1.2008506 | .51334756 | -1.9273331 |
| .30000000 | 1.2265180 | .41698091 | -1.9378280 |
| .35000000 | 1.2473670 | .32008951 | -1.9499144 |
| .40000000 | 1.2633715 | .22259379 | -1.9612334 |
| .45000000 | 1.2745012 | .12453212 | -1.9696408 |
| .50000000 | 1.2807278 | .02605008 | -1.9731919 |
| .55000000 | 1.2820303 | -.07260951 | -1.9701310 |
| .60000000 | 1.2783998 | -.17111607 | -1.9588875 |
| .65000000 | 1.2698440 | -.26906045 | -1.9380749 |
| .70000000 | 1.2563910 | -.36596420 | -1.9064927 |
| .75000000 | 1.2380928 | -.46128884 | -1.8631311 |
| .80000000 | 1.2150284 | -.55444540 | -1.8071774 |
| .85000000 | 1.1873061 | -.64480427 | -1.7380221 |
| .90000000 | 1.1550659 | -.73170538 | -1.6552671 |
| .95000000 | 1.1184806 | -.81446874 | -1.5587323 |
| 1.00000000 | 1.0777572 | -.89240536 | -1.4484609 |

Figure 5

Engineering computational problems may be classified in the following way — the first category being of trivial interest here, although certainly not trivial as far as computing is concerned:

1. Parameters specified, explicit equations.
2. Parameters specified, iteration required.
3. Parameters variable, explicit equations.
4. Parameters variable, iteration required.

Problems A through H are typical of the second category. Differential equations may fit in either the third or fourth category — examples I and J falling in category 3. The next subject to be discussed — mathematical models — also comes under either category 3 or 4.

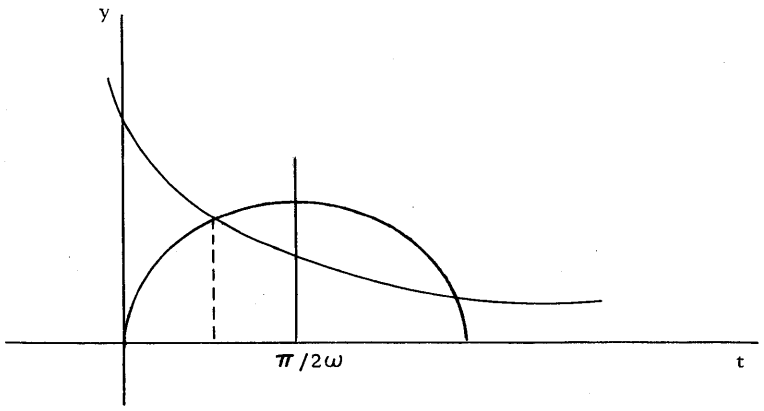
## Exercises

1. For problem C, find Newton's iterative equation and write the FORTRAN program to solve the problem.

2. Write Newton's equation to find a root of

$$5x^3 + 2x^2 + 3x + 1 = 0$$

3. The following is a problem which occurs in the planetary gear systems currently used in automatic transmissions on automobiles. The curves $y = \sin \omega t$ and $y = e^{-at}$ are plotted on the same set of y, t axes. Determine the smallest positive value of t at which they intersect.




a. Set up the equation for which a root is to be determined.

b. Write Newton's iterative equation to find the root.

*4. A common numerical procedure for calculating integrals is the use of Simpson's rule which may be stated: Given y as some function of x, the integral of the function y over the range $x = x_1$, to $x = x_n$ may be approximated by:

$$\int_{x = x_1}^{x = x_n} y\, dx \simeq \frac{\Delta x}{3} (y_1 + 4_{y2} + 2_{y3} + 4_{y4} + \cdots\cdots 2_{yn-2} + 4_{yn-1} + y_n)$$

where $x_i = x_{i-1} + \Delta x$

and $y_i$ is evaluated for the corresponding $x_i$.

Write a FORTRAN-language program to evaluate

$$\int_{x = 0}^{x = 1.0} y\, dx \qquad \text{where } y = \frac{1}{1 + x^2}$$

using a $\Delta x = .05$.

## SECTION V: MATHEMATICAL MODELS

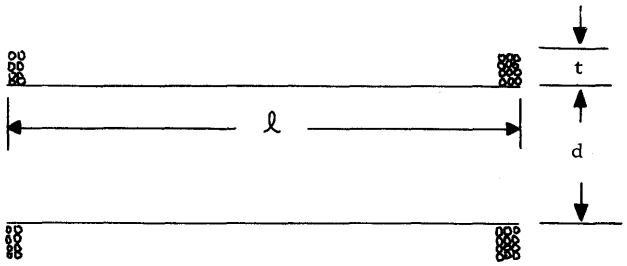
It is just as well not to belabor the question of what makes a mathematical model. However, it might be defined as a set of relationships which define a physical system. Different sets of operating conditions or input parameters, when handled by the model, give correspondingly different sets of operating results — problem category 3. If iteration is required for a set of input parameters, the problem is in category 4.

### Solenoid Design

This problem is most typical of design problems where several variables are involved and there is no obvious procedure for arriving at the best design (category 4 — parameters variable, iteration required).

Problem Statement:

Choose a wire for a solenoid such that the power consumed will be less than some fixed amount and the solenoid will give a fixed pull.




Engineering handbooks can supply the following set of relationships:

| | | |
|---|---|---|
| Pull | $F = \frac{IN}{\ell}$ | (1) where N=total number of turns |
| Current | $I = \frac{V}{R}$ | (2) where V=voltage applied |
| Power | $P = \frac{V^2}{R}$ | (3) |
| Resistance | $R = \frac{4\alpha^2 \rho L}{\pi}$ | (4) where $\rho$= resistivity of wire |
| Wire Length | $L = \pi D N$ | (5) |
| Solenoid Diameter | $D = d+t$ | (6) |
| Coil Thickness | $t = \frac{n}{\alpha}$ | (7) where $\alpha$=reciprocal of wire diameter (wire size) |
| Number of Layers | $n = \frac{N}{\alpha \ell}$ | (8) |

This is often the way a problem is presented to an engineer. If use of a computer is contemplated, the usual rules of analysis apply. That is, it

is best to do a little algebra before plugging in numbers. Too often the tendency is to choose wire size$\alpha$, start at equation 8 and work back, equation by equation, to arrive at power finally by equation 3. Application of algebra, however, gives:

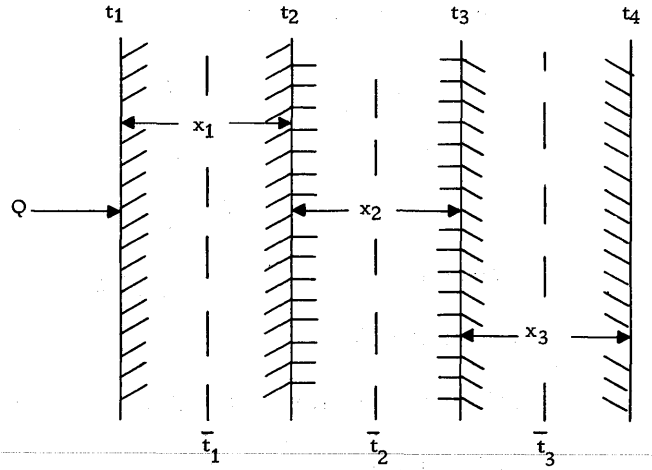
$$P = \frac{FV}{\left(\frac{V}{4F\rho\ell} - \alpha^2 d\right)} \qquad (9)$$

or power directly in terms of wire size. Savings in computer time can be just as important as savings in engineering analysis time. So it always pays to do a little analysis before muddying up the water with numbers.

The important aspect of this problem is not that one can now compute the power output for a given size and thus choose an acceptable wire size, but rather that a computer program can be generalized such that, given any requirements for a solenoid, many different combinations may be tried quickly to find the suitable combination for the job at hand. Thus, to design a solenoid for a given pull F, it might be desirable to vary $\ell$, d, and V or, perhaps, to limit t. Using equations 1 to 8, a variety of optimizing programs can be written for solenoid design. The question which would determine the general form of each program is: What particular specifications are to be met?

## Heat Transfer Problem

An insulating wall is made of three parallel layers of different insulating materials. The outside temperatures, $t_1$ and $t_4$, are known. The conductivity $k_i$ of each layer, a straight line function of the mean temperature $\bar{t}_i$, is known. The problem is to find the quantity of heat Q passing through a unit area of the wall per unit of time.




Q is given by

$$Q = \frac{t_1 - t_4}{\frac{x_1}{k_1} + \frac{x_2}{k_2} + \frac{x_3}{k_3}} \qquad (1)$$

However, the $k_i$ are dependent on the $\bar{t}_i$ in the following manner:

$$\begin{aligned} k_1 &= a_1\bar{t}_1 + b_1 \\ k_2 &= a_2\bar{t}_2 + b_2 \\ k_3 &= a_3\bar{t}_3 + b_3 \end{aligned} \qquad (2)$$

where the $a_i$ and $b_i$ are given constants for the particular intervals. At the same time, the $\bar{t}_i$ are given by

$$\begin{aligned} \bar{t}_1 &= t_1 - \frac{(t_1 - t_2)}{2} = \frac{1}{2}(t_1 + t_2) \\ \bar{t}_2 &= t_2 - \frac{(t_2 - t_3)}{2} = \frac{1}{2}(t_2 + t_3) \\ \bar{t}_3 &= t_3 - \frac{(t_3 - t_4)}{2} = \frac{1}{2}(t_3 + t_4) \end{aligned} \qquad (3)$$

but $t_2$ and $t_3$ are not known.

If the $k_i$ are known, $t_2$ and $t_3$ may be determined from the steady-state requirement that the quantity of heat passing through each layer per unit of time must be the same.

Or

$$\frac{k_1}{x_1}(t_1 - t_2) = \frac{k_2}{x_2}(t_2 - t_3) = \frac{k_3}{x_3}(t_3 - t_4)$$

which gives

$$t_3 = t_4 + \frac{\frac{x_3}{k_3}(t_1 - t_4)}{\frac{x_1}{k_1} + \frac{x_2}{k_2} + \frac{x_3}{k_3}} \qquad (4)$$

$$t_2 = t_1 - \frac{\frac{x_1}{k_1}(t_1 - t_4)}{\frac{x_1}{k_1} + \frac{x_2}{k_2} + \frac{x_3}{k_3}}$$

But this produces a vicious circle; the $k_i$ are needed to determine $t_2$ and $t_3$ and vice versa. Thus, iteration is required.

The procedure for this is as follows:

1. Make a reasonable guess at $t_2$ and $t_3$.
2. Use (3) to determine the $\bar{t}_i$.
3. Use (2) to determine the $k_i$.

4. Use (1) to determine Q.
5. Since values are known for $k_i$, recompute $t_2$ and $t_3$ using equations (4).
6. Repeat steps 2 through 5 until two successive values for Q agree.

This problem converges to a solution nicely using the simplest iterative scheme, $x = f(x,y,z)$. Experience has shown that the initial estimates of the internal temperatures are in no way critical except that $t_1$ must be greater than $t_2$, which must be greater than $t_3$, which must be greater than $t_4$.

The data used in this sample computation is as follows:

| | | $b_i$ | $a_i$ |
|---|---|---|---|
| Layer 1 | Kaolin Insulating Brick | .0623 | .00010 |
| Layer 2 | Kaolin Insulating Firebrick | .0255 | .00005 |
| Layer 3 | Magnesite | 2.4395 | .00060 |

$t_1 = 1400^o$

$t_4 = 200^o$

$x_1 = 2.15''$

$x_2 = 2.0''$

$x_3 = 6.0''$

The first set of results in Figure 6 shows the iterated solution to this problem giving the final value for the quantity of heat transmitted in a unit time as $\bar{Q} = 25.38$ Btu/min with the corresponding mean temperature $\bar{t}_1$, $\bar{t}_2$, and $\bar{t}_3$. An absurd case is also shown by the second set of results in Figure 6, with the input data changed in order to illustrate that convergence is not always possible. The data changed is

| $a_i$ |
|---|
| .02 |
| .01 |
| -.05 |

Actually it can be seen that the reason for the divergence of the second set was that the large negative slope $a_3$ caused $k_3$ to become negative (an impossible situation physically, since the coefficients of conductivity may not be negative). The value of -.05 for $a_3$ is not a reasonable value. It was chosen here to indicate that a mathematical procedure of this type is valid only within some range of selection of data.

```
      PQ=0.0
 1  0 READ 10,B1,B2,B3,T1,T4,X1,X2,X3
 2  0 READ 10,A1,A2,A3
 3  0 T2=2.*(T1-T4)/3.+T4
 4  0 T3=(T1-T4)/3.+T4
 5  0 T1B=.5*(T1+T2)
 6  0 T2B=.5*(T2+T3)
 7  0 T3B=.5*(T3+T4)
 8  0 Z1=A1*T1B+B1
 9  0 Z2=A2*T2B+B2
19  0 Z3=A3*T3B+B3
11  0 Q=(T1-T4)/(X1/Z1+X2/Z2+X3/Z3)
12  0 PRINT 20,Q,T1B,T2B,T3B
13  0 IF(ABSF(PQ-Q)-.00005)2,2,14
14  0 PQ = Q
15  0 T3=T4+X3/Z3*Q
16  0 T2=T1-X1/Z1*Q
17  0 GO TO 5
18  0 END
```

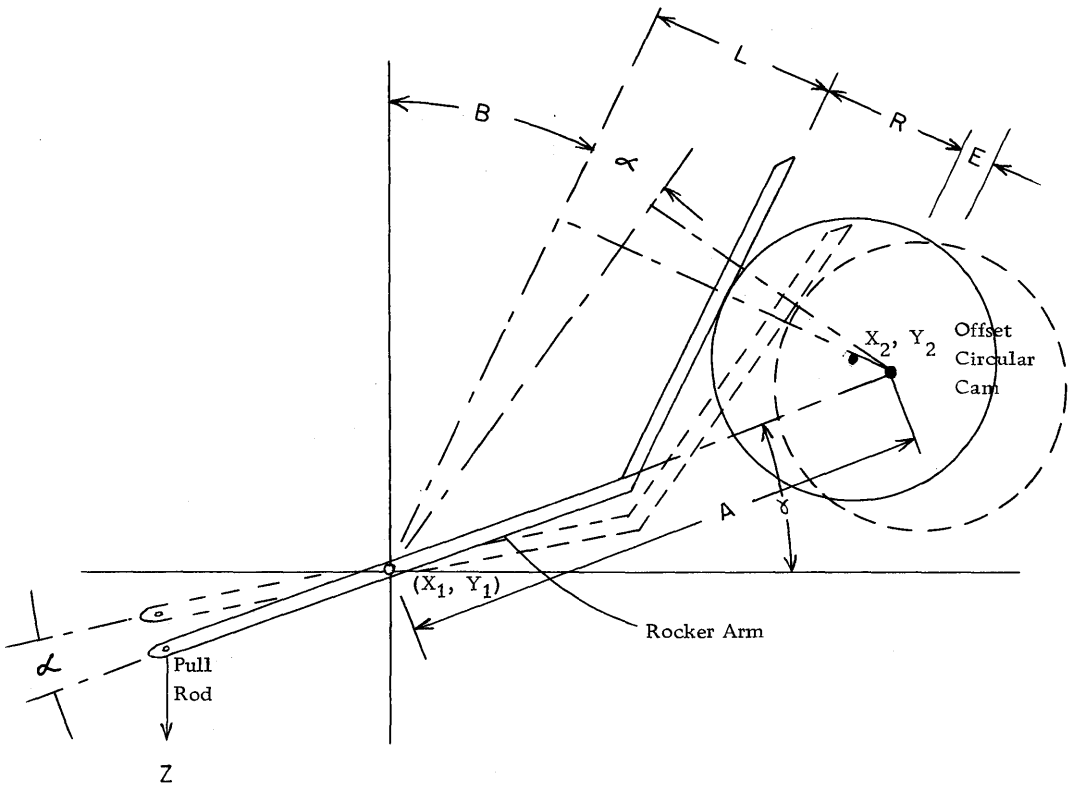
| Iteration | Q | $\bar{t}_1$ | $\bar{t}_2$ | $\bar{t}_3$ |
|---|---|---|---|---|
| 1 | 26.925558 | 1200.0000 | 800.00000 | 400.00000 |
| 2 | 25.146838 | 1241.2234 | 671.36960 | 230.14618 |
| 3 | 25.385972 | 1254.9914 | 684.25925 | 229.26788 |
| 4 | 25.384732 | 1254.6856 | 684.23785 | 229.55224 |
| 5 | 25.384376 | 1254.6691 | 684.21790 | 229.54884 |
| 6 | 25.384383 | 1254.6698 | 684.21825 | 229.54845 |

| Iteration | Q | $\bar{t}_1$ | $\bar{t}_2$ | $\bar{t}_3$ |
|---|---|---|---|---|
| 1 | -384754.98 | 1200.0000 | 800.00000 | 400.00000 |
| 2 | 188995.29 | 18589.197 | 83719.950 | 65930.755 |
| 3 | 145.56898 | 853.61840 | 81.496675 | 27.878275 |
| 4 | 38318.358 | 1390.8673 | 1208.5344 | 617.66710 |
| 5 | -823.77876 | -77.502100 | -4718.9752 | -3841.4731 |
| 6 | 4076.2550 | 804.76095 | 192.05571 | 187.29476 |

**Figure 6.**

## Rocker Arm Cam Problem

The accompanying figure shows a rocker arm and cam to be used in a fuel pump.




Specified are:

1. Axis of rotation of the off-center circular cam $(X_2, Y_2)$.
2. Eccentricity, E, and radius, R, of cam.
3. Axis of rotation of rocker arm $(X_1, Y_1)$.
4. Required angular rotation, $\alpha$ , of rocker arm to give desired travel of pull rod.

To be determined are:

1. Necessary angle, B, between rocker arm and vertical.
2. Necessary offset, L, for rocker arm.

To obtain a solution, "boundary conditions" are first considered to obtain expressions for B and L. For the "high" position of the rocker arm

$$B = 90^{o} - \gamma - \arcsin\left(\frac{L+R+E}{A}\right) \qquad (1)$$

For the "low" position of the rocker arm

$$L = A \sin(90^{o} - B - \gamma - \alpha) - (R-E) \qquad (2)$$

Note that because of the transcendental functions these equations may not be solved explicitly for either L or B.

The direct iterative procedure for solution is:

1. Make a reasonable guess at L.
2. Determine B from equation (1).
3. Then use this value of B in (2) to obtain a value for L.
4. Repeat steps 2 and 3 until two successive values for L agree.

This model requires iteration for the basic variables B and L. In actual practice, many other parameters are to be specified. These can be computed directly once B and L have been determined.

The solution here is interesting because the straightforward iteration procedure diverges. If equations (1) and (2) are stated as:

$$B = f_1(L)$$

$$L = f_2(B)$$

then a successful iterative equation to replace equation (2) in step 3 above is:

$$L_{i+1} = L_i - \frac{f_2(B_{i+1}) - L_i}{2} \qquad (3)$$

where

$$B_{i+1} = f_1(L_i)$$

Using this iterative procedure, convergence is relatively slow. An attempt to improve the convergence using the Newton-Raphson technique gives a divergent iterative scheme. However, a modification of the procedure shown does improve convergence.

The improvement was made simply by changing the equation to read:

$$L_{i+1} = L_i - (f_2(B_{i+1}) - L_i) \qquad (4)$$

This is treading on obviously dangerous ground, since arbitrary changes in the iterative equation are not easily justified. For this problem, however, it does work. This is mentioned here to show that experimentation with techniques is necessary in some instances when other approaches fail. But keep in mind that it is important to have some way of judging the correctness of the results.

Consideration of the following graph may assist in understanding the iterative process. Let

$$x_{n+1} = f(x_n)$$

be the direct iteration equation, and represent all other schemes as

$$x_{n+1} = W\,(f\,(x_n),\ x_n)$$

where the W represents the weighting function used in making the next estimate.

In either case the evaluation of the function giving the next estimate might be plotted as in the graph.

For a solution, the next estimate of x and the input value of x must be equal ( $\bar{f}\,(x)$ and $\bar{x}$ as shown in the graph). Assume an error, $\epsilon$ , is made in the first estimate, $x_n$. The graph indicates the next estimate to be in error by an amount $\epsilon_2$ where $\epsilon_2 > \epsilon_1$ (the technique is diverging and $\epsilon_3$ is even greater than $\epsilon_2$).

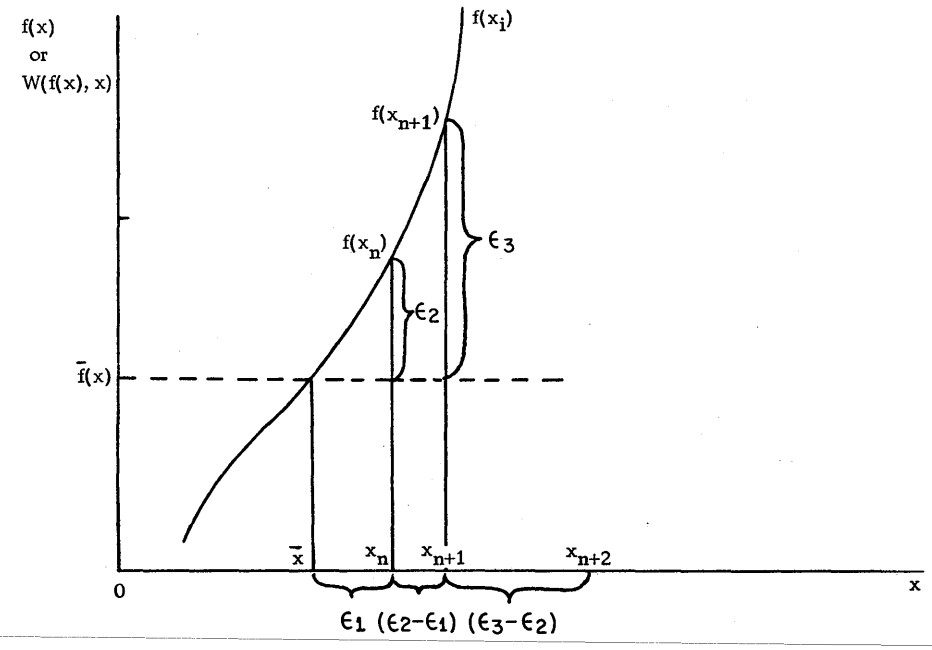
Given the initial estimate $x_n$ and the graph as shown, an obvious procedure for preventing the divergence is to make use of the difference $(\epsilon_2 - \epsilon_1)$ in writing a new iteration equation — that is:

$$x_{n+1} = x_n - (\epsilon_2 - \epsilon_1)$$

and since $(\epsilon_2 - \epsilon_1) = f\,(x_n) - x_n$

then

$$x_{n+1} = x_n - (f\,(x_n) - x_n)$$




This is recognized as equation (4) above for the cam problem. If $(\epsilon_2 - \epsilon_1) > \epsilon_1$ this procedure overshoots the correct result and divergence may again result. Equation (3) in the cam problem used a weighting factor of 1/2 to prevent the overshoot. Choice of the "best"

weighting factor would require complete knowledge of a similar graph — which of course is not available without a solution.

The intent here is to indicate the reasoning which may go into the selection of iteration equations.

The direct and Newton-Raphson iterative techniques are not the only procedures available. For example, wide use has been made of a procedure credited to Mr. J. H. Wegstein of the National Bureau of Standards. The Wegstein method does not require evaluation of the first derivative and therefore is a powerful tool in cases where the first derivative is impossible or difficult to evaluate.

The rocker arm cam problem illustrates the use of experimentation to find a successful iterative scheme. No mention has been made of the use of an iterative technique to handle simultaneous algebraic equations.

The problems discussed in this manual have been treated in the form

$$x = f(x, y, z)$$

Physical problems may frequently give rise to forms similar to:

$$x = f_1(x, y, z)$$

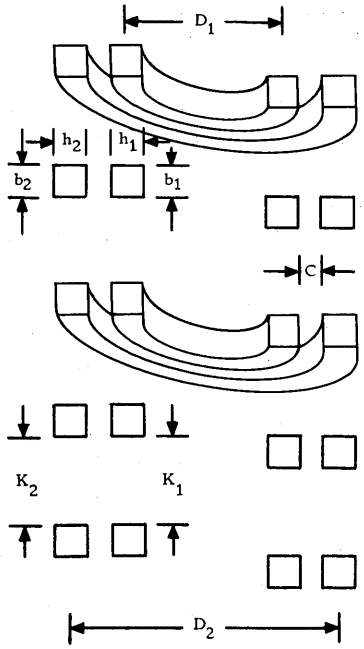
$$y = f_2(x, y, z)$$

As a matter of fact the rocker arm cam problem can be treated as a problem giving rise to simultaneous algebraic equations.

The Newton-Raphson and Wegstein methods may both be extended to handle simultaneous equations.

## Spring Problem

The engineering design of springs makes an excellent computer application, especially from the economic viewpoint. It takes a great many springs to keep this mechanical world operating, and a less expensive or longer-lasting spring design can mean large returns dollar-wise. This example is concerned with the design of a particular type of spring, but the general method of computer solution would apply to almost any spring problem.




Cross Section of a Double Helical Spring

A double helical spring is to be designed to support a given torque winding, such that the mean diameter and length of the spring will be within set limits — the shorter the length the better. The variables specified (subscript 1 refers to inner spring, subscript 2 to outer spring) are:

$M_1$ , $M_2$ = Winding torque

$S_1$ , $S_2$ = Maximum selected stress, or ultimate stress, of material

$T_1$ , $T_2$ = Maximum torsion angle

$K_1$ , $K_2$ = Required spacing between turns

C = Required spacing between coils

E = Young's Modulus

The following are to be computed:

$h_1$ , $h_2$ = Dimensions shown in the diagram

$b_1$ , $b_2$ = Dimensions shown in the diagram

$D_1$ , $D_2$ = Diameter of springs

$L_1$ , $L_2$ = Length of springs

$N_1$ , $N_2$ = Number of turns in the springs

Known relationships are:

(1) $M = \frac{E\, bh^3 T}{6.6DN}$ Gives winding torque for a given torsion angle.

(2) $M = \frac{Sbh^2}{6}$ Limits winding torque in terms of ultimate stress of the material.

(3) $L_2 = (N_2 + 1)(b_2 + K_2)$ Outer length by geometry.

(4) $D_1 = D_2 - h_1 - h_2 - 2C$ Inner dimension by geometry.

Note that for the outer spring five variables ($h_2$, $b_2$, $D_2$, $L_2$, $N_2$) must be determined but there are only three equations which apply. One approach to obtaining a solution is:

(a) Set $D_2$ (because limit exists)

$N_2$ (because this must be either an integer or an integer plus .5 — successive guesses will be simpler)

(b) Equate (1) and (2) and solve for h.

$$h_2 = \frac{1.1 D_2 S_2 N_2}{ET_2} \qquad (5)$$

(c) Then from (2) obtain

$$b_2 = \frac{6M_2}{S_2 h_2^2} \qquad (6)$$

(d) and $L_2 = (N_2 + 1)(b_2 + K_2)$ (7)

This provides the parameters for the outer spring. At this point it is necessary to make an additional guess to handle the inner spring.

(e) Set $N_1$

(f) Then solve

$$D_1 = D_2 - h_1 - h_2 - 2C$$

and $$h_1 = \frac{1.1\, D_1\, S_1\, N_1}{ET_1}$$

simultaneously for $h_1$ obtaining

$$h_1 = \frac{1.1\, S_1 N_1 (D_2 - h_2 - 2C)}{ET_1 + 1.1\, S_1 N_1} \qquad (8)$$

(g) From (2):

$$b_1 = \frac{6M_1}{S_1 h_1^2} \qquad (9)$$

(h) and finally:

$$L_1 = (N_2 + 1)(b_2 + K_2)$$

Now testing is performed to see whether the springs are suitable; $L_2$ must be equal to $L_1$.

According to the design criterion — that $L_2$ approximately equal $L_1$ — a decision can be made to accept or reject this design. In case of rejection the selection for $D_2$, $N_2$ and $N_1$ can be modified and the indicated calculations repeated.

A logic diagram of the procedure is shown in Figure 7.

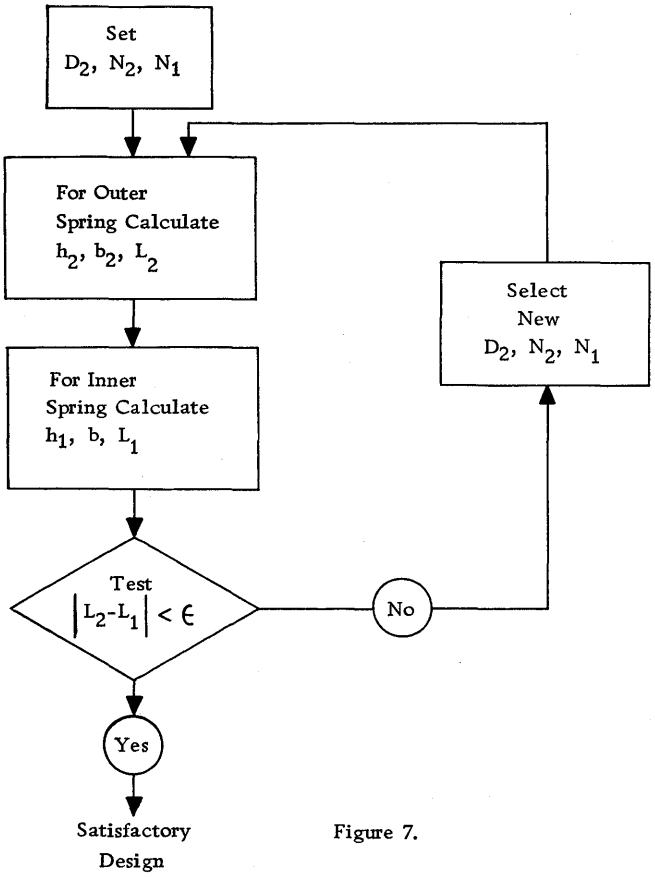



Figure 7.

The question of how to modify the selection of $D_2$, $N_2$ and $N_1$ must be considered. The mathematical nature of the problem (nonlinear equations with an infinite number of solutions) does not permit use of the iterative techniques mentioned thus far. Two possible procedures for varying the selection may be called scattering and search. Both of these procedures assume the existence of a maximum or minimum for the function G which is used to denote "goodness" or "acceptability." In the spring problem the function is

$$G - \left| L_2 - L_1 \right| = G\,(D_2, N_2, N_1)$$

Considering G as a function of only two variables

$$G = G\ (D_2,\ N_2)$$

allows visualization of the graph of G shown here. A corresponding three-dimensional picture for

$$G\ (D_2,\ N_2,\ N_1)$$

could also be visualized.

The continuous curves represent contours on which G is everywhere equal. There may exist one or more minimums as shown.

Scattering

A set of selected values for $D_2$ and $N_2$ (in three dimensions, $D_2$, $N_2$, and $N_1$) which create a grid covering the possible range in both $D_2$ and $N_2$ is

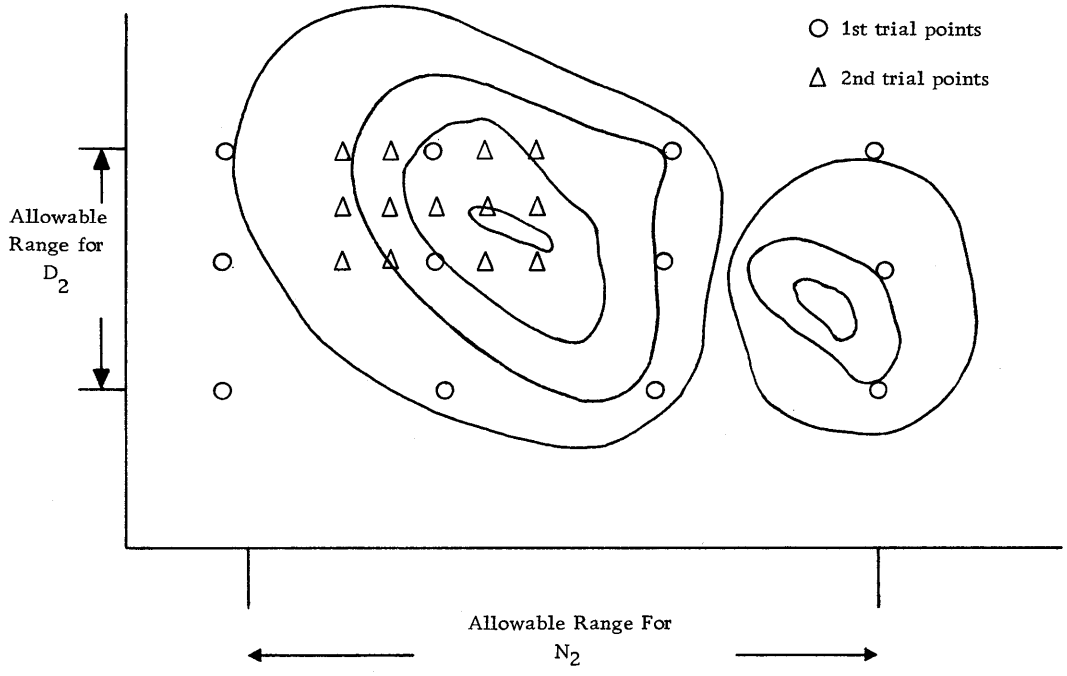



Figure 8. Scattering

used to compute corresponding values for G. The grid is shown in Figure 8 as a set of O's. A smaller area is selected around the minimum G and the process is repeated, as indicated by the Δ's in the figure.

This corresponds to the familiar trial-and-error approach except that many tries may be run at one time on a computer. Each solution may be made available allowing consideration of secondary design criteria. In the spring problem, a minimum $|L_2 - L_1|$ is not sufficient for a good design; the ratios $\frac{h_1}{b_1}$, $\frac{h_2}{b_2}$ and $\frac{h_1}{h_2}$ are also important. It is quite probable

that the finally selected spring would have a small but not the smallest $|L_2 - L_1|$ and other desirable features.

Search

One starting point in n dimensions is selected (shown in Figure 9 for the two-dimensional case).

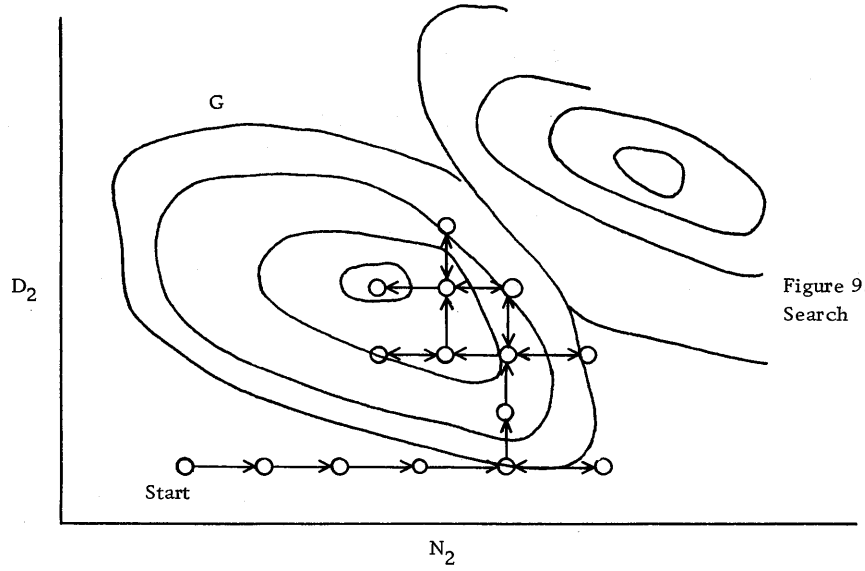



Figure 9. Search

For each variable in turn:

(a) The variable is changed by some small amount and G is recomputed.

(b) If G decreases, (a) is repeated until such time as G begins to increase and then the variable is set to correspond to the smallest G.

(c) If G increases for (a), an equal change is made to move in the opposite direction in a manner similar to (b).

Steps (a), (b) and (c) are repeated for each variable in turn, until such time as new changes within the practical bounds fail to decrease G significantly.

This method moves more quickly to the solution and represents a more fully automated procedure. However, it demands that the behavior of the function (in this case G) be well understood. Discontinuities or lesser "minimums" can destroy the effectiveness of this method.

The scattering approach can often be used to define the behavior of a criterion function such as G ($D_2$, $N_2$, $N_1$) above, to the extent that the search technique may be used in succeeding computations.

It is possible to add other selection criteria in the search technique. For example, the material cost in designing the above spring might be used such that the selected spring has the lowest material cost for those designs with a value of G within a specified range.

## Pitman Arm Stress Analysis

A problem often encountered in mechanical engineering is that of designing a member to withstand given tensile and shear stresses. This particular example concerns a pitman arm (see Figure 10) as a part of an auto steering mechanism. The drawing shows a pitman arm with the applied

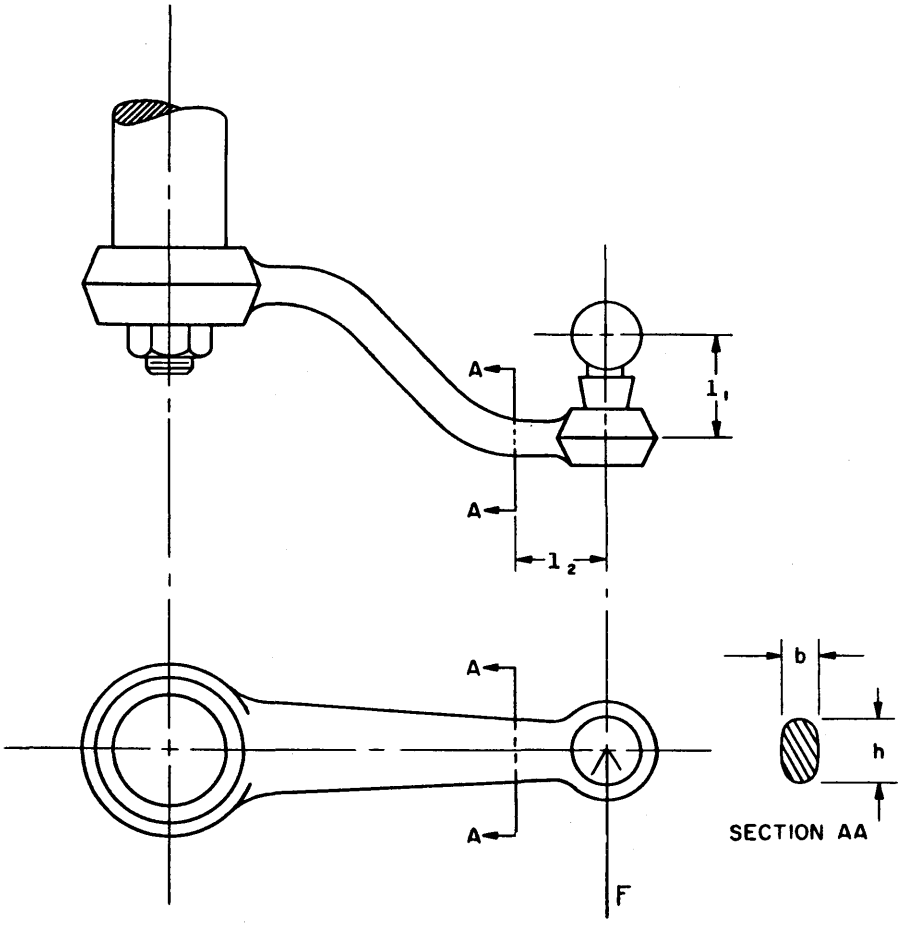



Figure 10. Pitman Arm

force F which gives rise to the various stresses in the arm.

The usual approach to this problem is for the engineer to specify dimensions which seem appropriate and then to check the design with calculations. This is done by choosing cross sections of the design at intervals along the arm — computing the stresses at each cross section and checking that these stresses are less than the failure stress for the material used.

For Section AA shown, the computations are:

(a) For bending stress

$$S_b = \frac{Fl_2}{Z_x} \qquad (1)$$

For torsional shear stress

$$S_t = \frac{Fl_1}{Z_p} \qquad (2)$$

where $Z_x$ and $Z_p$ are called section moduli and depend upon the moments of inertia for the member. For the elliptical case the moduli are

$$Z_x = \frac{\pi}{32} bh^2 \qquad (3)$$

$$Z_p = \frac{\pi}{16} bh^2 \qquad (4)$$

(b) The total normal tensile stress and the total shear stress can be computed in terms of the bending stress and the torsional shear stress.

For normal tensile stress

$$S_n = \frac{1}{2}\left[S_b \pm (S_b^2 + 4S_t^2)^{1/2}\right] \qquad (5)$$

For shear stress

$$S_s = \frac{1}{2}(S_b^2 + 4S_t^2)^{1/2} \qquad (6)$$

Once these total or maximum stresses have been calculated, they are compared with acceptable values for the material to check whether a feasible design has been arrived at.

An approach which may be of more use to the engineer — and which is facilitated by the use of computers — is to start with the maximum allowable stresses and from this to obtain the dimensions required for the member to support these stresses.

For example, assume that the pitman arm shown is to be elliptical in cross section, with h=2b, and that values have been assigned to $S_n$ and $S_s$. The problem now consists of determining values for b and h in terms of these values.

Equations (5) and (6) can be manipulated to give $S_b$ and $S_t$ in terms of $S_n$ and $S_s$:

$$S_b = 2(S_n - S_s) \qquad (7)$$

$$S_t = \left[S_s^2 - (S_n - S_s)^2\right]^{1/2} \qquad (8)$$

Equations (1) to (4) may be used finally to restrict b and h in terms of $S_b$ and $S_t$:

$$\frac{\pi}{32} bh^2 = \frac{Fl_2}{S_b}$$

and

$$\frac{\pi}{16} bh^2 = \frac{Fl_1}{S_t} \qquad \text{or, since } h = 2b,$$

$$b^3 - \frac{8}{\pi} \frac{Fl_2}{S_b} = 0 \qquad (9)$$

and $$b^3 - \frac{4}{\pi} \frac{Fl_1}{S_t} = 0 \qquad (10)$$

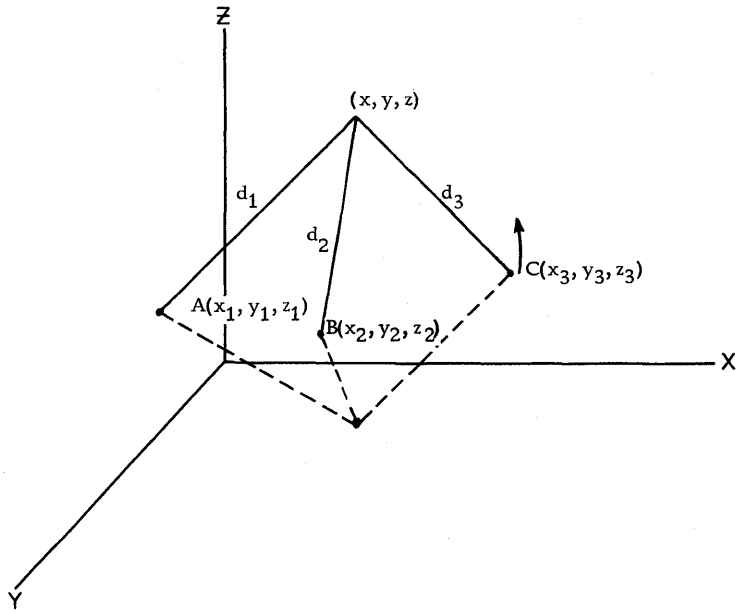
This gives two cubics which restrict b. The proper design must have a value of b which is at least as large as the larger of the two roots obtained from equations (9) and (10).

Now the proper dimensions can be computed for the arm as we move along the arm (that is, change $l_1$ and $l_2$). Furthermore, if the curve used for the arm can be expressed mathematically, then $l_1$ and $l_2$ can be automatically computed as the arm is traversed toward or away from the axis of rotation. If this is impossible, then $l_1$ and $l_2$ must be measured for various cross sections of the arm and the corresponding dimensions computed for the cross sections.

The point of this example is that very often a computer approach should start with a reanalysis of what is to be done, considering the possibilities of improving the usefulness of the results.

## Automobile Suspension

The following three-dimensional problem occurs in the engineering of auto suspension systems; arms of length $d_1$, $d_2$ and $d_3$ are fixed at points in space A, B and C respectively.



The point of juncture of the three rigid arms is found to be the intersection of three spheres with centers at A, B and C and radii $d_1$, $d_2$ and $d_3$ respectively.

Expressing this in equation form gives the following three equations:

$$(x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 = d_1^2 \qquad (1)$$

$$(x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 = d_2^2 \qquad (2)$$

$$(x - x_3)^2 + (y - y_3)^2 + (z - z_3)^2 = d_3^2 \qquad (3)$$

This set of equations appears difficult to solve for x, y and z, and iteration has been used in many instances to find a solution. Iteration in this problem is time-consuming and unreliable (convergence is not always possible).

If the following algebraic steps are performed, an explicit relationship for x, y and z is found:

(a) Perform indicated squaring in equations (1), (2) and (3).

(b) Subtract equation (2) from (1) and similarly equation (3) from (2). This will eliminate terms in $x^2$, $y^2$, and $z^2$ giving two linear equations in x, y and z.

(c) Solve these two linear equations algebraically to obtain x and y as linear functions of z.

(d) Substitute these functions for x and y into equation (1) and solve the resulting quadratic in z.

This will then give equations of the form:

$$z = \frac{-b + (b^2 - 4ac)^{\frac{1}{2}}}{2a}$$

$$x = e + fz$$

$$y = g + hz$$

where a, b, c, e, f, g and h are intermediate substitutions for terms made up of the known quantities $x_i$, $y_i$, $z_i$ and $d_i$.

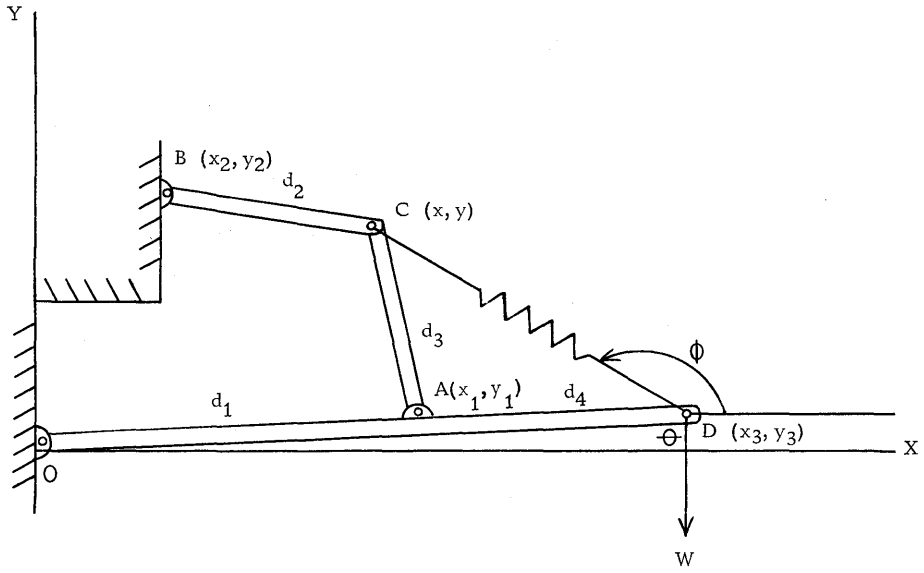
This gives two complete sets of answers: the correct physical answer and its mirror image as illustrated by the dotted lines. The selection of the proper set of results can be accomplished by programming the logical considerations of the physical system.

A computer is convenient for the evaluation of the lengthy equations involved. If the problem is to accomplish this evaluation for many points in the motion of one of the points (say point C, as indicated by the arrow), the use of a computer becomes mandatory. The orientation of a suspension system must be known for the many possibilities in the auto's motion.

This problem illustrates an important class of mechanical engineering problem — that of determining the orientation of physical bodies by specifying the coordinates of points in the body.

## Spring Stress Analysis Problem

The following illustrates the use of the same algebraic approach in another problem which also involves the use of the summation of forces so familiar in stress analysis. Find the necessary tension, S, in the spring for the support of W at various values of the angle $\theta$.




The analysis might proceed as follows:

To find point A

$$x_1 = d_1 \cos \theta \qquad (1)$$

$$y_1 = d_1 \sin \theta \qquad (2)$$

To find point C

$$(x-x_1)^2 + (y-y_1)^2 = d_3^2 \qquad (3)$$

$$(x-x_2)^2 + (y-y_2)^2 = d_2^2 \qquad (4)$$

Following the procedure as outlined in the previous problem, expand (3) and (4) to give:

$$x^2 - 2xx_1 + x_1^2 + y^2 - 2yy_1 + y_1^2 = d_3^2 \qquad (5)$$

$$x^2 - 2xx_2 + x_2^2 + y^2 - 2yy_2 + y_2^2 = d_2^2 \qquad (6)$$

Subtract (6) from (5) to give:

$$2x\,(x_2-x_1) + (x_1^2 - x_2^2) + 2y\,(y_2-y_1) + (y_1^2-y_2^2) = (d_3^2-d_2^2)$$

$$\text{or } x = Ey+F \qquad (7)$$

Where

$$E = \frac{(y_1-y_2)}{(x_2-x_1)}$$

and $$F = \left[(d_3^2-d_2^2) - (y_1^2-y_2^2) - (x_1^2-x_2^2)\right] / 2\,(x_2-x_1)$$

Substituting from (7) into (5) gives

$$E^2y^2 + 2EFy + F^2 - 2x_1Ey - 2x_1F + x_1^2 + y^2 - 2yy_1 - y_1^2 = d_3^2 \quad \text{or}$$

$$(1+E^2)y^2 + (2EF - 2x_1E - 2y_1)y + (F^2 - 2x_1F + x_1^2 - y_1^2 - d_3^2) = 0$$

$$\text{or finally } y = \frac{-Q + \sqrt{Q^2-4PR}}{2P}$$

where

$$P = (1+E^2)$$

$$Q = (2EF - 2x_1E - 2y_1)$$

$$R = (F^2 - 2x_1F + x_1^2 - y_1^2 - d_3^2)$$

and $x = Ey+F$

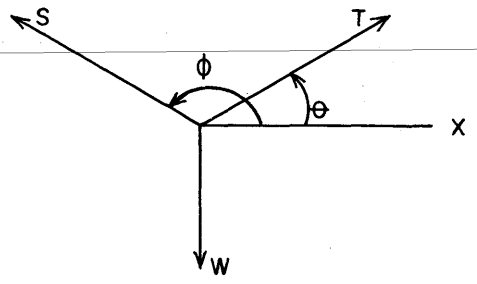
To find angle $\phi$

$$x_3 = (d_1 + d_4)\cos\theta$$

$$y_3 = (d_1 + d_4)\sin\theta$$

$$\text{and } \phi = \tan^{-1}\frac{(y-y_3)}{(x-x_3)}$$

The determination of $\phi$ allows the use of the force diagram and the equilibrium principle, which states that the sum of forces acting in both the x and y directions must be equal to zero, to determine the necessary tension, S, in the spring.



$$\sum F_x = T \cos\theta - S \sin(\phi - 90^o) = 0$$

$$\sum F_y = T \sin\theta + S \cos(\phi - 90^o) - W = 0$$

or

$$T = \frac{S \sin(\phi - 90^o)}{\cos\theta}$$

$$S = W / \left[\cos(\phi - 90^o) + \tan\theta \sin(\phi - 90^o)\right]$$

The usefulness in the design of a program to evaluate the tension in the spring for any configuration, motion of $\theta$, or load W is obvious. In this case the evaluation depends upon locating points in a plane. Traditionally this type of determination has been made on a drafting board. As in this case, where motion or many possible designs are to be evaluated, calculations on a computer can greatly reduce the time invested. The improvement in accuracy can be even more important. For example, a one-degree error in the steering geometry of an auto can completely change its steering characteristics.

## Vehicle Simulation Model

It would be unfair to close this brief discussion of a few engineering mathematical models without mention of at least one of the larger models which have been constructed.

Several auto manufacturers have succeeded in representing the operation of cars and trucks by a computer model. By operating the model in the computer, such items of performance as acceleration, fuel economy, and riding characteristics can be determined for a proposed model before the model is built.

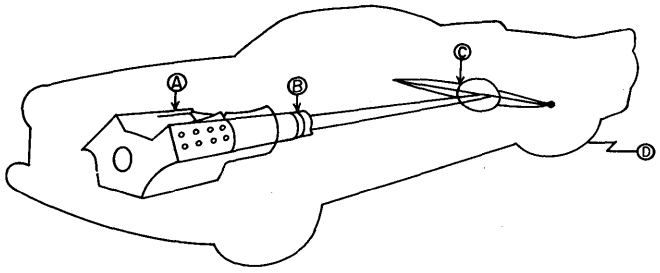
The procedure for accomplishing this introduces a very interesting concept. Normally the solution to a set of equations is obtained by solving for one point in time — a static situation. The performance of a car, however, is a dynamic situation. Beyond this, equations to describe all the myriad operations in moving a car simply do not exist. Rather, there exist combinations of theoretical equations, empirical equations and tabular data to describe the functioning of various parts of a car. The heart of a car simulation model is the logic which ties these pieces of theory and data together under one unifying variable. Different approaches choose different unifying variables; for illustration, however, suppose the variable chosen is time.

Typically, the motion of the car might be studied time-wise for a set level of fuel input. For the first unit of time the information representing the relationships between parts of the car is interrogated for the motion to be expected, working from the engine back through the transmission and finally to the wheels. Once the motion for the first unit of time is determined, the time is incremented and the operation repeated.

Another approach might be essentially the reverse of this: Starting with the desired velocity of the vehicle, work back through the wheels, differential and transmission to determine what the engine speed must be to give this velocity. However, time is again the unifying factor in that conditions for the previous time interval help determine the required motor speed for a particular velocity.

A look at the basic relationships in this example may be helpful.

Assume we want to know the linear acceleration of the vehicle at full throttle as a function of time.




Starting with a given engine speed $N_1$ and linear velocity V, the first set of relationships is concerned with the engine, A.

(a) The first step is to use a table to determine the output torque of the engine.

$$T_1 = f_1 (N_1)$$

The torque delivered to the transmission may likewise be a tabular function of $N_1$:

$$T_3 = f_2 (N_1)$$

as is the torque consumed in friction of the engine parts:

$$T_2 = f_3 (N_1)$$

At this point the acceleration of the engine for this time interval may be determined as

$$\alpha_m = \frac{T_1 - T_2 - T_3}{I}$$

where I = total engine inertia

(b) Drive shaft speed, $N_2$, is computed by

$$N_2 = \frac{V}{L} G$$

where G = axle ratio
and L = rolling radius.

The transmission ratio, TR, is a function of $N_2/N_1$

$$TR = f(N_2/N_1)$$

This relationship is best represented by a table of TR vs $N_2/N_1$ which has been determined experimentally for the particular transmission to be used.

(c) Moving on to the drive shaft, the torque delivered to the drive shaft may be computed as

$$T_4 = T_3 \cdot TR$$

(d) Finally, arriving at the rear wheels, the force exerted on the wheels due to the delivered torque may be given as

$$F = \frac{G}{L} (T_4 - T_5)$$

where $T_5$ = torque loss due to friction and acceleration of drive line parts.

Then the acceleration of the car, a, is given by Newton's second law of motion

$$a = \frac{F - R}{M}$$

where M = mass of the car and
R = rolling resistance.

The rolling resistance is generally given as an empirical function such as

$$R = aM + b\,AV^2$$

where A = frontal area of vehicle, and a and b are experimentally determined constants (see section on empirical relationships).

At this point the time is incremented and a new engine speed $N_1$ is computed as

$$t_m = t_m + \Delta t \qquad m+1$$

$$(N_1)_{m+1} = (N_1)_m + \int_m \propto_m \Delta t$$

The above procedure A, B, C and D may be repeated for as many time increments as desired.

This logical procedure may be represented by the flow chart in Figure 11.

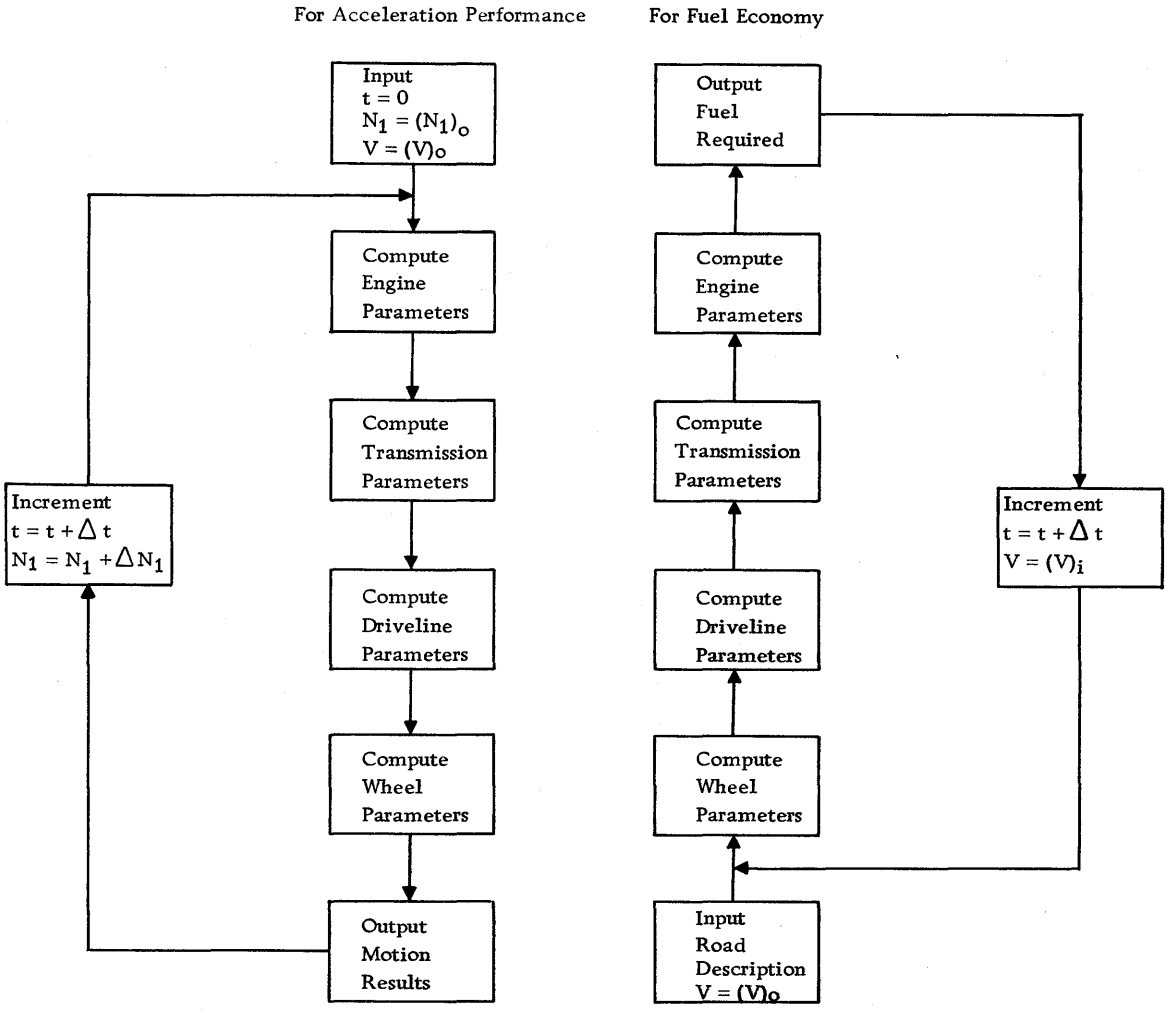



Figure 11.




Figure 12.

The flow chart in Figure 12 is included to illustrate how essentially the same mathematical model may be used in reverse to compute fuel economy. In this case the desired motion as a function of time is specified. Calculations determine the engine speed required to give the desired motion for each time interval. Finally the fuel consumed at a given engine speed is known and therefore the sum of the fuel to be consumed for all time intervals may be calculated.

Many simplifications of vehicle simulation have been made in this description.

The major objective in presenting a description of vehicle performance in this chapter is to illustrate the use of the three types of information (theoretical equations, empirical equations, and tabular test data) under a logic format to obtain a computer simulation of the physical object.

The next chapter describes the use of the computer as a tool in finding empirical relationships. Actual procedures for handling of tabular data in a computer program are also discussed.

## Exercises

1. Perform the algebra required to obtain equation 9 in the solenoid design problem, using equations 1 to 8.

2. Perform the algebra indicated in the suspension analysis problem and either show the complete equations for x, y and z or show the equations which define the intermediate variables a, b, c, r, s, t and u.

*3. Numerical techniques are often used to solve simultaneous differential equations with nonlinear coefficients. Prepare a logic flow chart to be used to program the solution of:

$$\frac{d^2y}{dt^2} + \alpha\frac{dx}{dt} + xy = A$$

$$\frac{dx}{dt} + \frac{dy}{dt} + \alpha = B$$

where A and B are known constants and $\alpha = t + \sin \alpha\, t$, given $t_o$, $x_o$, $y_o$ and $\left(\frac{dy}{dt}\right)_o$.

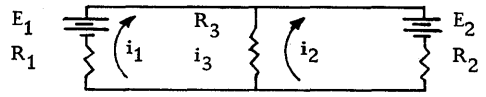
Hint: Use Euler's approximations as described in problems I and J of Section IV for the differential equations and Newton's procedure for evaluating $\alpha$ at each interval in t.

## SECTION VI: SELECTED MATHEMATICAL TOPICS

The problems discussed in this section have been chosen to answer some of the legitimate questions that many practicing engineers might well ask about terms often heard in computer groups. Such things as matrix inversion, relaxation techniques and eigenvalues are out of the realm of experience of many engineers. It is felt that some simple examples illustrating these terms may lead to eventual recognition of real problems which can be tackled using the techniques described.

### Matrix Inversion

A major application of computers is in handling the solution of large sets of simultaneous equations which may occur in such engineering areas as stress analysis, statistical least squares and circuit analysis.




For the simple circuit shown, applying Kirchhoff's law gives a set of linear equations in $i_1$ and $i_2$:

$$(R_1 + R_3)\, i_1 - R_3 i_2 = E_1$$

$$-R_3 i_1 + (R_2 + R_3)\, i_2 = -E_2$$

This set of two equations in two unknowns does not require sophisticated handling. However, consider a slightly more complex circuit in which the values of the resistances are known and the currents are to be determined. Again, Kirchhoff's law may be used to establish a set of linear equations:

$$I_1 + I_2 + I_3 = A_1$$

$$I_8 + I_9 + I_{10} = A_2$$

$$-I_1 + I_4 - I_6 = 0$$

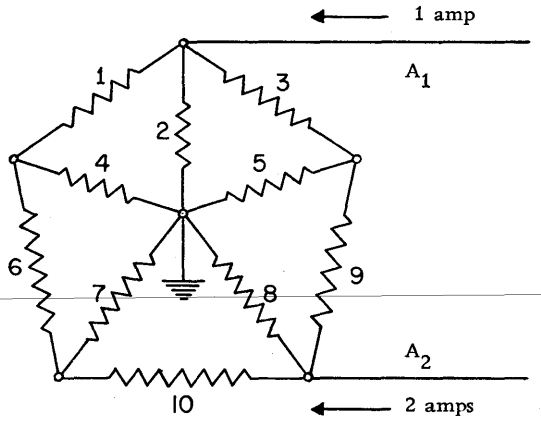
$$-I_3 + I_5 - I_9 = 0$$

$$I_6 + I_7 - I_{10} = 0$$

$$-R_7 I_7 + R_8 I_8 - R_{10} I_{10} = 0$$

$$-R_5 I_5 + R_8 I_8 - R_9 I_9 = 0$$

$$R_2 I_2 - R_3 I_3 - R_5 I_5 = 0$$



$$-R_1I_1 + R_2I_2 - R_4I_4 = 0$$

$$-R_4I_4 - R_6I_6 + R_7I_7 = 0$$

Since the solution of this set would require a considerable expense of time and effort, the problem is well suited for a computer.

The general procedure for solving a set of simultaneous equations on a computer is to make use of matrix inversion techniques. A standard elimination technique for matrix inversion is performed in the following manner:

Given the coefficient matrix:

$$\begin{array}{|cccc|c|} a_{11} & a_{12} & a_{13} \cdots & a_{1n} & 1 \\ a_{21} & a_{22} & a_{23} \cdots & a_{2n} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} \cdots & a_{nn} & 0 \\ \hline \end{array}$$

$$\square\square\square\ldots\square$$

add a column — a unit vector — which contains a 1 in the first row and zeros elsewhere. At the same time, add a row — called the pivot row — denoted by $\square$'s. Then perform the following computations to arrive at a new array:

1. For the pivot row elements:

$$a_{p,j} = \frac{a_{1,\,j+1}}{a_{1,1}} \qquad j = 1, 2, \ldots, n$$

2. For all other elements, compute a new value:

$$a'_{i,j} = a_{i,j+1} - (a_{i,1})(a_{p,j}) \qquad \begin{array}{l} i = 1, 2, \ldots, n-1 \\ j = 1, 2, \ldots, n \end{array}$$

3. As a result of step 2, all the new elements of row 1 are zero. This row is dropped, and the remaining n rows renumbered 1 through n. Thus, for the last row:

$$a'_{n,j} = a_{p,j} \qquad j = 1, 2, \ldots, n$$

4. Add a new unit vector and pivot row and repeat steps 1, 2 and 3 a total of n times. The resulting array is the inverse of the original matrix.

Given a set of simultaneous equations:

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\
\vdots \quad\quad \vdots \quad\quad\quad\quad \vdots \quad\quad \vdots \\
a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n &= b_n
\end{aligned}
$$

then the solution can be obtained directly by starting with an n+1 by n array in which the original matrix is augmented by the b vector. If values of x are required for more than one set of b values, the additional b vectors can be incorporated in the original array as shown:

$$
\begin{matrix}
a_{11} & a_{12} & \ldots & a_{1n} & b_{11} & b_{12} \\
a_{21} & a_{22} & \ldots & a_{2n} & b_{21} & b_{22} \\
\vdots & \vdots & & \vdots & \vdots & \vdots \\
a_{n1} & a_{n2} & \ldots & a_{nn} & b_{n1} & b_{n2}
\end{matrix}
$$

Subjecting this m by n matrix to the four-step procedure above a total of m times gives the array:

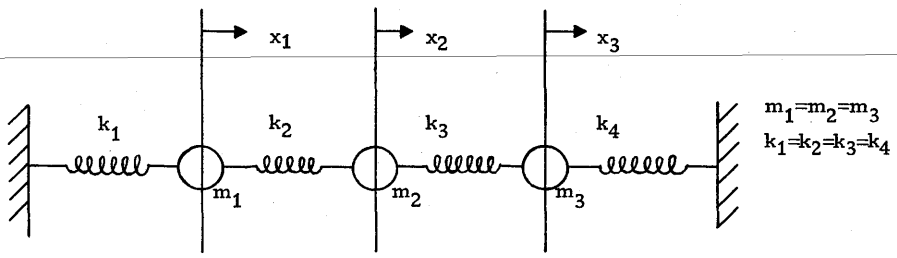
$$
\begin{pmatrix}
x_{11} & x_{12} & a'_{11} & a'_{12} & \ldots & a'_{1n} \\
x_{21} & x_{22} & a'_{21} & a'_{22} & \ldots & a'_{2n} \\
\vdots & \vdots & \vdots & & & \vdots \\
x_{n1} & x_{n2} & a'_{n1} & a'_{n2} & \ldots & a'_{nn}
\end{pmatrix}
$$

where the $a'_{i,j}$ are the elements of the inverse of the original coefficient matrix, and the $x_{i,j}$ are the solutions for each of the two b vectors of the matrix equation

$$AX = B$$

## Eigenvalue Problem

This problem is discussed to show, with a simple example, an application of computers in handling sets of differential equations. It illustrates the meaning of eigenvalues for a set of differential equations which describe the motion of a mechanical system.



Given the spring and mass system shown, assume that the normal modes of oscillation of the system are to be determined.

The differential equations of motion can be written:

$$m_1\ddot{x}_1 = -kx_1 + k(x_2 - x_1) = k(x_2 - 2x_1)$$

$$m_2\ddot{x}_2 = -k(x_2 - x_1) - k(x_2 - x_3) = -k(2x_2 - x_1 - x_3)$$

$$m_3\ddot{x}_3 = -kx_3 + k(x_2 - x_3) = k(x_2 - 2x_3) \qquad (1)$$

A procedure for solving these differential equations is to assume solutions of the form

$$x_1 = Ae^{i\omega t}\ ,\ x_2 = Be^{i\omega t}\ ,\ x_3 = Ce^{i\omega t} \qquad (2)$$

Substituting these into equations (1), performing the indicated differentiations and rearranging terms gives:

$$\begin{aligned} (2\tfrac{k}{m} - \omega^2)\ A - \tfrac{k}{m}\ B &= 0 \\ -\tfrac{k}{m}\ A + (2\tfrac{k}{m} - \omega^2)\ B - \tfrac{k}{m}C &= 0 \qquad (3) \\ -\tfrac{k}{m}\ B + (\tfrac{2k}{m} - \omega^2)C &= 0 \end{aligned}$$

which can be written in matrix form as

$$\left(\begin{pmatrix} 2\frac{k}{m} - \frac{k}{m} & 0 \\ \frac{-k}{m} + \frac{2k}{m} - \frac{k}{m} & \\ 0 - \frac{k}{m} + \frac{2k}{m} & \end{pmatrix} - \begin{pmatrix} \omega^2 & 0 & 0 \\ 0 & \omega^2 & 0 \\ 0 & 0 & \omega^2 \end{pmatrix}\right)\begin{pmatrix} A \\ B \\ C \end{pmatrix} = 0 \qquad (4)$$

or more simply,

$$(D - \lambda I)\ X = 0 \qquad (5)$$

For A, B, C to satisfy equations (3), the determinant of the coefficient matrix must vanish. This is equivalent to the statement

$$(D - \lambda I) = 0 \qquad (6)$$

Evaluating the determinant for equations (3) and equating it to zero gives a polynomial — called the characteristic equation — in $\omega^2$:

$$(\tfrac{2k}{m} - \omega^2)^3 - 2\tfrac{k^2}{m}\ (2\tfrac{k}{m} - \omega^2) = 0 \qquad (7)$$

with roots

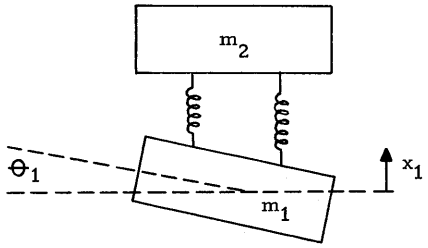
$$\begin{aligned} \omega_1^2 &= \frac{2k}{m} \\ \omega_2^2 &= \frac{2k}{m} \pm \sqrt{2}\ \frac{k}{m} \end{aligned} \qquad (8)$$

The values of $\omega$ which satisfy these equations are called eigenvalues. In general, values of $\lambda$ which satisfy (5) are eigenvalues. The vector X is called the eigenvector.

It can be seen that for larger sets of differential equations the solution of the characteristic equation requires some means for computing the roots of a polynomial.

For this problem the eigenvalues give the natural modes of vibration for the mass and spring system. This calculation of eigenvalues for differential equations is termed frequency analysis. In an earlier section attention was given to the solution of differential equations. In motion problems this amounts to determining the displacements as a function of time and is termed an amplitude analysis. Both frequency analysis and amplitude analysis are important computer applications.

To illustrate a computer solution to a frequency analysis, consider the spring and mass system shown in the diagram.




The differential equations which describe this system are as follows:

$$\frac{d^2x_1}{dt^2} + a_{11}x_1 + a_{12}x_2 + a_{13}\theta_1 + a_{14}\theta_2 = 0$$

$$\frac{d^2x_2}{dt^2} + a_{21}x_1 + a_{22}x_2 + a_{23}\theta_1 + a_{24}\theta_2 = 0$$

$$\frac{d^2\theta_1}{dt^2} + a_{31}x_1 + a_{32}x_2 + a_{33}\theta_1 + a_{34}\theta_2 = 0$$

$$\frac{d^2\theta_2}{dt^2} + a_{41}x_1 + a_{42}x_2 + a_{43}\theta_1 + a_{44}\theta_2 = 0$$

where the $a_{i,j}$ are dependent on the spring constants and the masses.

Assume solutions of the form:

$$x_1 = x_{10}\cos\omega t \qquad \theta_1 = \theta_{10}\cos\omega t$$

$$x_2 = x_{20} \cos \omega t \qquad \theta_2 = \theta_{20} \cos \omega t$$

where $x_{10}$, $x_{20}$, $\theta_{10}$ and $\theta_{20}$ are the initial displacements. The appropriate differentiations and substitutions give a homogeneous set of linear equations of the form:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_{10} \\ x_{20} \\ \theta_{10} \\ \theta_{20} \end{bmatrix} = \omega^2 \begin{bmatrix} x_{10} \\ x_{20} \\ \theta_{10} \\ \theta_{20} \end{bmatrix}$$

The iterative procedure for the solution of these equations involves the following steps:

1. Initially, let $x_{10} = x_{20} = \theta_{10} = \theta_{20} = 1.0$ and evaluate the left-hand side for new values of $\omega^2 x_{10}$, $\omega^2 x_{20}$, $\omega^2 \theta_{10}$ and $\omega^2 \theta_{20}$.

   That is:

   $$\omega^2 x_{10} = a_{11} x_{10} + a_{12} x_{20} + a_{13} \theta_{10} + a_{14} \theta_{20}$$

   and so on.

2. "Normalize" for new guesses at $x_{10}$, $x_{20}$, $\theta_{10}$, and $\theta_{20}$ by setting

   $$x_{10} = 1 \qquad \theta_{10} = \frac{\omega^2 \theta_{10}}{\omega^2 x_{10}}$$

   $$x_{20} = \frac{\omega^2 x_{20}}{\omega^2 x_{10}} \qquad \theta_{x_{20}} = \frac{\omega^2 \theta_{20}}{\omega^2 x_{10}}$$

3. Repeat steps 1 and 2 until such time as successive values of $x_{10}$, $x_{20}$, $\theta_{10}$, and $\theta_{20}$ are very close. At this time, convergence has occurred and $\omega^2$ can be computed.

4. Upon convergence $\omega^2$ can be evaluated from the given value of $\omega^2 x_{10}$; say

   $$\omega^2 x_{10} = K$$

The following is the iterative solution for $\omega^2$ of a sample set of coefficients:

Particular equation set

$$\begin{bmatrix} +929 & -332 & +3,590 & -3590 \\ -748 & +748 & -8,090 & +8090 \\ +4.17 & -4.17 & +35,400 & -11,760 \\ -14.31 & +14.31 & -40,300 & +40,300 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \theta_1 \\ \theta_2 \end{bmatrix} = \omega^2 \begin{bmatrix} x_1 \\ x_2 \\ \theta_1 \\ \theta_2 \end{bmatrix}$$

Iterative solution

| Iteration | $x_1$ | $x_2$ | $\theta_1$ | $\theta_2$ |
|---|---|---|---|---|
| 1 | 1.0 | 1.0 | 1.0 | 1.0 |
| 2 | 597.0 | 0.0 | 23640.0 | 0.0 |
| 3 | 143085.0 | -321095.0 | 1401773.0 | -1595813.0 |
| 4 | 76883.0 | -171908.0 | 477975.0 | -844314.0 |
| 5 | 63414.0 | -141558.0 | 349238.0 | -693154.0 |
| 6 | 60681.0 | -135398.0 | 323511.0 | -662485.0 |
| 7 | 60002.0 | -133869.0 | 317130.0 | -654870.0 |
| 8 | 59825.0 | -133468.0 | 315460.0 | -652876.0 |
| 9 | 59777.0 | -133362.0 | 315017.0 | -652347.0 |

$$\omega^2 = 59777.0$$

Knowing that $x_{10}$ is to be set to 1, then

$$\omega^2 = K$$

To clarify what has been done here, remember that the set of equations has no constant term — it is a homogeneous set of equations. Essentially this means that there are an infinite number of solutions which satisfy the equations. This is certainly reasonable when the physical system under consideration is examined. In a vibration problem of this kind the initial displacements $x_{10}$, $x_{20}$, $\theta_{20}$, and $\theta_{20}$ must be expected to take on different values.

In the above procedure a value of $x_{10}$ was selected which then fixed the values of the other variables $x_{20}$, $\theta_{10}$, $\theta_{20}$ and allowed $\omega^2$ to be determined.

A very similar iterative scheme (without the normalization step) can be used to solve sets of nonhomogeneous linear equations.

## Partial Differential Equations

Many engineering problems involve the handling of partial differential equations. Three classes have been distinguished as:

1. Elliptical equations (describing potential fields)

$$\nabla^2 \phi = g(x, y, z)$$

2. Parabolic equations (describing heat flow and diffusion)

$$\nabla^2 \phi = k \frac{\partial \phi}{\partial t}$$

3. Hyperbolic equations (describing wave action)

$$\nabla^2 \phi = \frac{1}{C^2} \frac{\partial^2 \phi}{\partial t^2}$$

where $\nabla^2$ is the Laplacian operator in rectilinear coordinates:

$$\nabla^2 \phi = \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} + \frac{\partial^2 \phi}{\partial z^2}$$

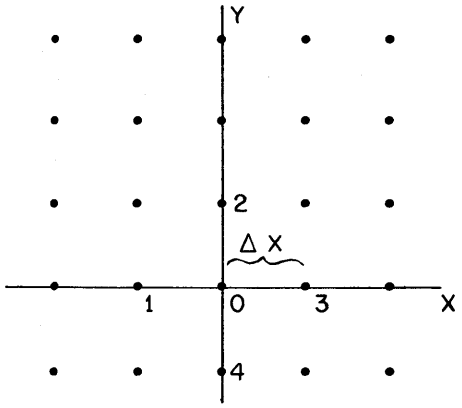
A basic approach to handling partial differential equations when describing a particular material or space is to create a grid of points covering the space as shown for a two-dimensional space.

Then, at any point 0 the first derivative with respect to x can be approximated in two ways:

$$\left(\frac{\partial \phi}{\partial x}\right)_1 \cong \frac{\phi_3 - \phi_o}{\Delta x}$$

or (1)

$$\left(\frac{\partial \phi}{\partial x}\right)_2 \cong \frac{\phi_o - \phi_1}{\Delta x}$$




The second derivative can be approximated as

$$\frac{\partial^2 \phi}{\partial x^2} \cong \frac{\left(\frac{\partial \phi}{\partial x}\right)_1 - \left(\frac{\partial \phi}{\partial x}\right)_2}{\Delta x} = \frac{\phi_3 + \phi_1 - 2\phi_o}{(\Delta x)^2} \qquad (2)$$

Derivatives in the y direction can be obtained in the same way. Using this procedure, any partial differential equation may be reduced to difference equations amenable to computer handling.

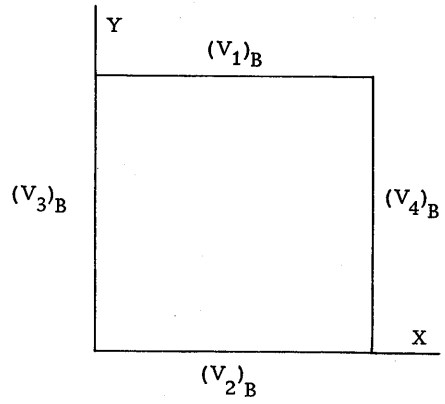
## Potential Problem

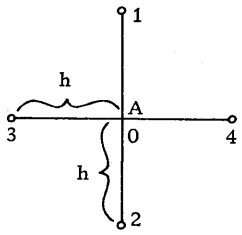
Apply this to the two-dimensional problem of finding the potential distribution in a square whose sides are maintained at voltages

$$(V_1)_B, (V_2)_B, (V_3)_B, \text{ and } (V_4)_B$$

If there is no charge within the square, the potential distribution is defined by the Laplace equation:

$$\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} = 0 \qquad (1)$$




Setting up a square grid system to cover the square, for the general point A, the approximations for the partial derivatives become:

$$\left(\frac{\partial V}{\partial x}\right)_1 = \frac{V_4 - V_o}{h} \quad , \quad \left(\frac{\partial V}{\partial x}\right)_2 = \frac{V_o - V_3}{h}$$

$$\frac{\partial^2 V}{\partial x^2} \cong \frac{V_4 + V_3 - 2V_o}{h^2}$$

Similarly, for the y dimension:

$$\frac{\partial^2 V}{\partial y^2} \cong \frac{V_2 + V_1 - 2V_o}{h^2}$$

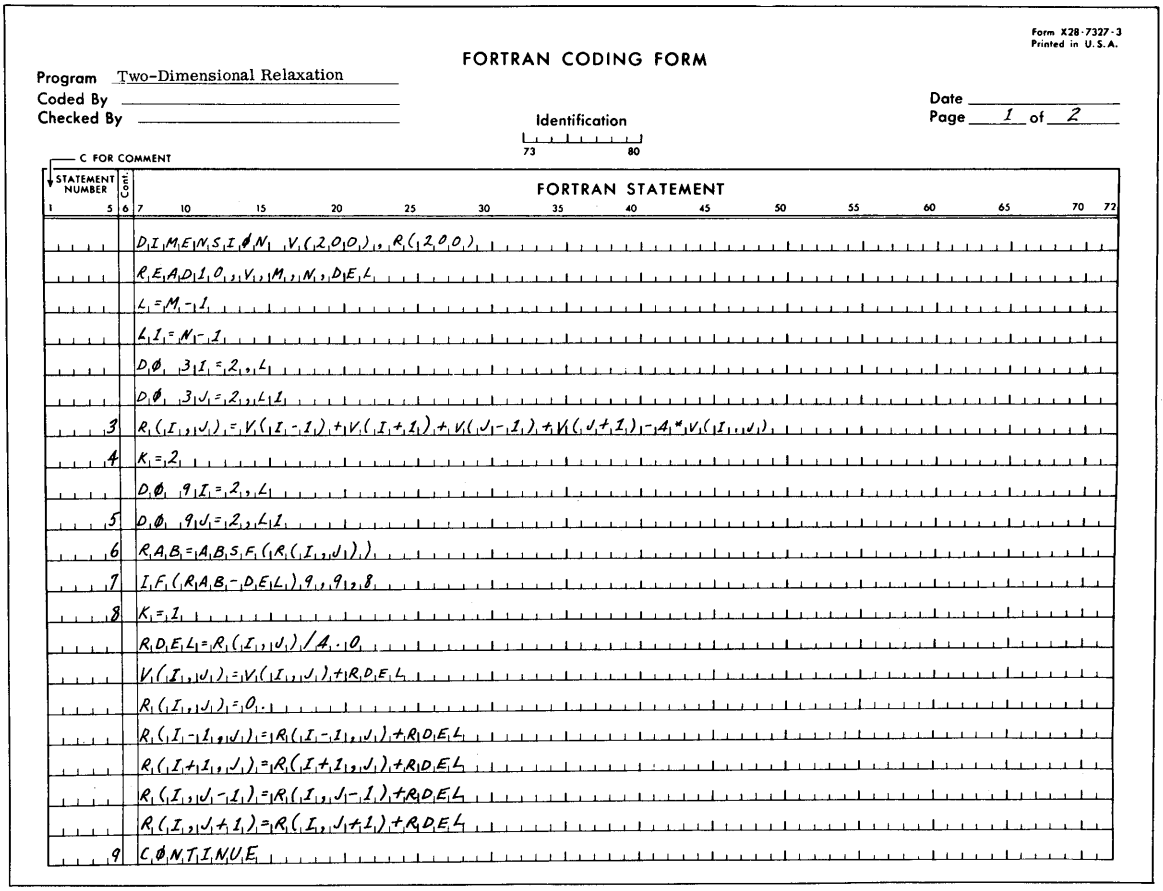
Then (1) becomes:

$$V_1 + V_2 + V_3 + V_4 - 4\,V_o = 0 \qquad (2)$$

This is the basic relaxation equation. It is applied in the following way:

1. A first guess at the potential of each point on the grid is made on the basis of the known boundary conditions.

2. Moving systematically through the points of the grid, the quantity called the residual is computed for each point and stored. The residual is given by:



Form X28-7327-3
Printed in U.S.A.

FORTRAN CODING FORM

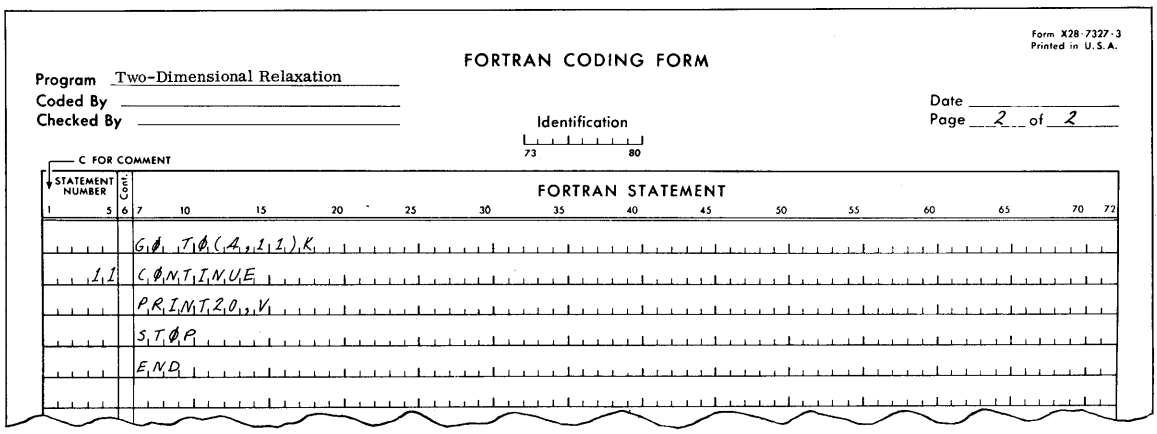
Program Two-Dimensional Relaxation
Coded By
Checked By

Identification (73–80)

Date
Page 1 of 2

C FOR COMMENT

| STATEMENT NUMBER (1–5) | Cont. (6) | FORTRAN STATEMENT (7–72) |
|---|---|---|
| | | DIMENSION V(20,20), R(20,20) |
| | | READ 10, V, M, N, DEL |
| | | L = M - 1 |
| | | L1 = N - 1 |
| | | DO 3 I = 2, L |
| | | DO 3 J = 2, L1 |
| 3 | | R(I,J) = V(I-1) + V(I+1) + V(J-1) + V(J+1) - 4.*V(I,J) |
| 4 | | K = 2 |
| | | DO 9 I = 2, L |
| 5 | | DO 9 J = 2, L1 |
| 6 | | RAB = ABS(R(I,J)) |
| 7 | | IF (RAB - DEL) 9, 9, 8 |
| 8 | | K = 1 |
| | | RDEL = R(I,J)/A |
| | | V(I,J) = V(I,J) + RDEL |
| | | R(I,J) = 0. |
| | | R(I-1,J) = R(I-1,J) + RDEL |
| | | R(I+1,J) = R(I+1,J) + RDEL |
| | | R(I,J-1) = R(I,J-1) + RDEL |
| | | R(I,J+1) = R(I,J+1) + RDEL |
| 9 | | CONTINUE |



Form X28-7327-3
Printed in U.S.A.

FORTRAN CODING FORM

Program Two-Dimensional Relaxation
Coded By
Checked By

Identification (73–80)

Date
Page 2 of 2

C FOR COMMENT

| STATEMENT NUMBER (1–5) | Cont. (6) | FORTRAN STATEMENT (7–72) |
|---|---|---|
| | | GO TO (4, 11), K |
| 11 | | CONTINUE |
| | | PRINT 20, V |
| | | STOP |
| | | END |

Figure 13.

$$R_o = V_1 + V_2 + V_3 + V_4 - 4V_o$$

Initially, equation (2) will not be satisfied since the potentials are guesses.

3. Again moving systematically and considering each point not on the boundary, the potential is adjusted to make the residual for the point equal to zero by applying the following equation:

$$\text{new } V_o = V_o + R_o/4$$

4. This affects the residuals of the surrounding points, so they are adjusted by:

$$\text{new } R_i = R_i + R_o/4$$

5. Steps 3 and 4 are repeated until no residual is found whose absolute value is greater than some predetermined limit of accuracy. At this time the relaxation equation is satisfied and the potential distribution is known.

It is possible to quickly write a FORTRAN program to do the necessary computation. For this problem let

M = Number of points in grid on x axis (200 max.)

N = Number of points in grid on y axis (200 max.)

V (I, J) = Potential at points on grid (initial guesses) plus boundary values

R (I, J) = Associated residual

DEL = Limit of accuracy desired.

The resulting FORTRAN program is shown in Figure 13.

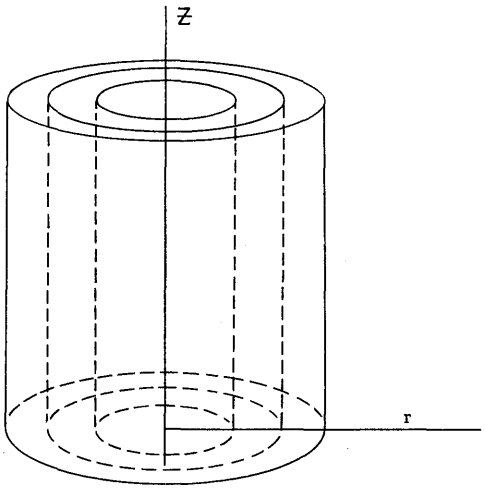
With this example in mind, it is of value to consider how a more complex problem of this type might be handled.

## Temperature Distribution Problem

For a cylinder made up of layers of material of different conductivities, find the steady-state temperature distribution. The outer surface temperature (boundary temperature) is known. Here the use of cylindrical coordinates facilitates analysis. In cylindrical coordinates the heat flow equation is:

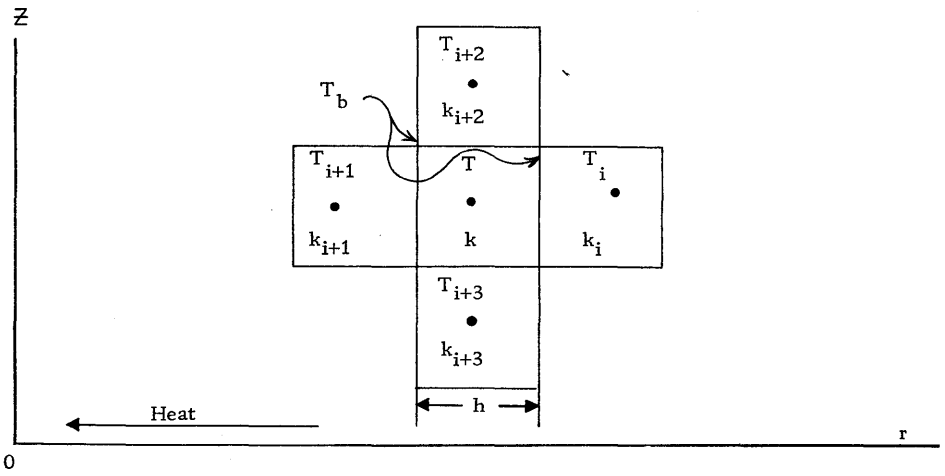
$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) + \frac{1}{r^2}\frac{\partial T}{\partial \phi^2} + \frac{\partial^2 T}{\partial z^2} = \frac{k}{\partial t}\frac{\partial T}{\partial t} \quad (1)$$

For the steady state, where $\frac{\partial T}{\partial t} = 0$, then




$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right)+\frac{1}{r^2}\frac{\partial^2 T}{\partial \phi^2}+\frac{\partial^2 T}{\partial z^2}=0 \qquad (2)$$

Since this deals with a nonhomogeneous cylinder, any approximation to the partial differential equations must show a dependence on the conductivity of the material. A suitable approximation may be derived by reference to the accompanying diagram.




First assume the boundary between materials to be midway between two points on the grid and introduce a boundary temperature $T_b$. Then the quantity of heat which would pass from one square and (assuming incorrect initial assignment of temperatures) the quantity received by the second must be equal; or

$$k_i\,(T_i - T_b) = k\,(T_b - T)$$

or

$$T_b = \frac{k_i\,T_i + kT}{k + k_i}$$

Then approximate

$$\left(\frac{\partial T}{\partial r}\right)_1 = \frac{T_b - T}{h/2}$$

or

$$\left(\frac{\partial T}{\partial r}\right)_1 = \left(\frac{k_i}{k_i + k}\right) \left(\frac{T_i - T}{h/2}\right)$$

An equivalent approximation redefining $T_b$ may be made from the opposite side to give:

$$\left(\frac{\partial T}{\partial r}\right)_2 = \left(\frac{k_i + 1}{k_{i+1}+k}\right) \left(\frac{-T_{i+1} + T}{h/2}\right)$$

The final approximation to the first derivative may be taken as:

$$\frac{\partial T}{\partial r} \cong \frac{\left(\frac{\partial T}{\partial r}\right)_1 + \left(\frac{\partial T}{\partial r}\right)_2}{2} = \left(\frac{k_i}{k_i+k}\right)\left(\frac{T_i-T}{h}\right) + \left(\frac{k_{i+1}}{k_{i+1}+k}\right)\left(\frac{-T_{i+1}+T}{h}\right)$$

The second partial may be taken as:

$$\frac{\partial^2 T}{\partial r^2} \cong -\frac{\left(\frac{\partial T}{\partial r}\right)_2 + \left(\frac{\partial T}{\partial r}\right)_1}{h} = \left(\frac{k_{i+1}}{k_{i+1}+k}\right)\left(\frac{T_{i+1}-T}{h^2/2}\right) + \left(\frac{ki}{k_i+k}\right)\left(\frac{T_i-T}{h^2/2}\right)$$

For uniform boundary temperature:

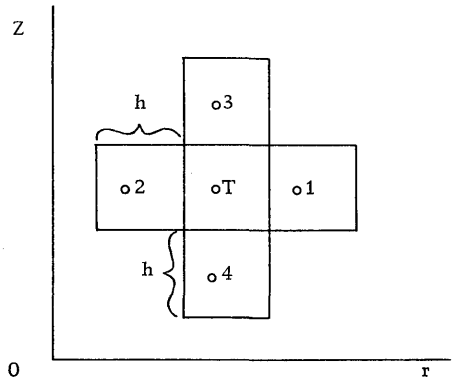
$$\frac{\partial^2 T}{\partial \phi^2} = 0$$

and $$\frac{\partial^2 T}{\partial z^2} \cong \left(\frac{k_{i+3}}{k_{i+3}+k}\right)\left(\frac{T_{i+3}-T}{h^2/2}\right) + \left(\frac{k_{i+2}}{k_{i+2}+k}\right)\left(\frac{T_{i+2}-T}{h^2/2}\right)$$

Making use of these approximations in equation (2) gives the steady-state heat flow difference equation:

$$\frac{2}{h^2}\left\{\left(\frac{k_1}{k_1+k}\right)\left(1 + \frac{h}{2r}\right)\left[T_1-T\right] + \left(\frac{k_2}{k_2+k}\right)\left(1 - \frac{h}{2r}\right)\left[T_2-T\right] + \left(\frac{k3}{k_3+k}\right)\left[T_3-T\right] + \left(\frac{k4}{k_4+k}\right)\left[T_4-T\right]\right\} = 0$$

referring to the general cell illustrated.




With this equation the standard technique of relaxation by residuals may be used to find the steady-state temperature distribution.

This problem has been discussed in order to illustrate how the relaxation technique may be applied in cases of two added types of complexities:

1. Cylindrical coordinate system

2. Nonhomogeneous material

## Exercises

1. Write a FORTRAN program to perform a matrix multiplication. The maximum size of the input matrices is 20 by 20. The actual size of the matrix is L, m, n, and is to be read from data cards.

   Note:

$$C_{ij} = \sum_{k=1}^{L} A_{ik} \cdot B_{kj}$$

*2. Using the method outlined in the matrix inversion topic, find the coefficient inverse and solution for:

$$x_1 + 2x_2 + 3x_3 = 3$$

$$2x_1 + 3x_2 + 4x_3 = 5$$

$$3x_1 + 4x_2 + 4x_3 = 5$$

*3. The discussion of the problem on the iterative solution for homogeneous equations mentions the use of the iterative process for nonhomogeneous equations.

   Given the set of simultaneous equations

$$AX = B$$

one such iterative procedure is to rewrite the equations in the form

$$X = CX + F$$

and use

$$X_{n+1} = CX_n + F$$

as the iteration equation.

For the set of equations

$$25x_1 + 2x_2 + x_3 = 69$$

$$2x_1 + 10x_2 + x_3 = 63$$

$$x_1 + x_2 + 4x_3 = 43$$

the iterative form is

$$x_1 = -24x_1 - 2x_2 - x_3 + 69$$

$$x_2 = -2x_1 - 9x_2 - x_3 + 63$$

$$x_3 = -x_1 - x_2 - 3x_3 + 43$$

Iterate for the values of $x_1$, $x_2$ and $x_3$ which satisfy the equations.

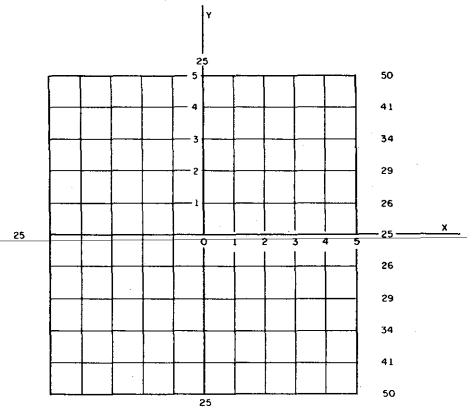
*4. Perform relaxation on the below grid which has boundary conditions:

$$V_{boundary} = x^2 + y^2$$

for the potential described by Laplace's equation

$$\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} = 0$$

Make use of symmetry wherever possible.



# SECTION VII: EMPIRICAL RELATIONSHIPS

The previous chapters have discussed various ways of handling problems adequately described by theoretical equations. Some attention has been given to each of the general steps in computer solution of the problems, namely:

1. Selection of the appropriate relationships.

2. Mathematical manipulations involving algebra or the calculus to arrive at a suitable evaluation equation.

3. Choice of numerical technique when necessary.

4. FORTRAN programming and actual computer evaluation.

It must be recognized that this approach does not suffice in situations where step 1 is impossible or where the known relationships do not adequately describe the problem.

The lack of theoretical knowledge may often be adjusted for by the use of empirical data taken by means of experiments and test runs.

The discussion of the vehicle simulation in a previous section introduced the use of tabular data and an empirically derived equation together with theoretical equations in the description of the vehicle operation.

The engineer is no stranger to the use of tables of data. Page upon page of collected data arises in almost every test of an engineered part or system. Conclusions are as often drawn from the consideration of tables of data as from theoretical data. It is important that this source of information be available, as is, for use with computer analysis. At the same time the possibility often exists for derivation of an empirical relationship to fit the data. This section will consider some of the standard procedures for:

1. Use of tables of data as they exist.

2. Derivation of empirical relationships from the data itself.

The discussion will be broken into four areas, as shown in the chart below:

| | FUNCTIONS OF A SINGLE VARIABLE | FUNCTIONS OF MULTIPLE VARIABLES |
|---|---|---|
| TABLE LOOKUP | a. Table storage<br>b. Table interpolation | a. Table storage<br>b. Interpolation logic |
| DATA FITTING | a. Fitting Criteria<br>1. Selected points<br>2. Least squares | a. Linear regression<br>b. Multiple linear regression<br>c. Nonlinear Estimation<br>d. Dimensional Analysis |

Given the following table of data, how might the proper value of Y for a given value of X be obtained during a computer run?

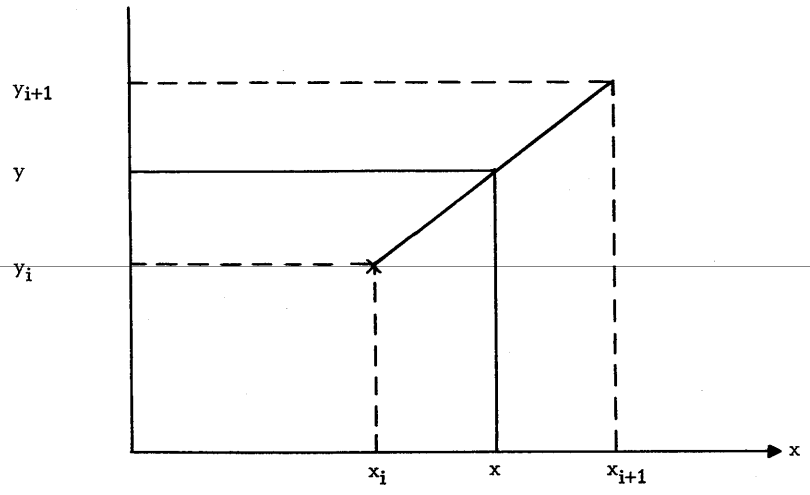
| X | X(I) | Y | Y(I) |
|---|---|---|---|
| 0.0 | X(1) | .913 | Y(1) |
| 4.0 | X(2) | .930 | Y(2) |
| 8.0 | X(3) | .941 | Y(3) |
| 12.0 | X(4) | .946 | Y(4) |
| 16.0 | X(5) | .948 | Y(5) |
| 20.0 | X(6) | .950 | Y(6) |
| 24.0 | X(7) | .951 | Y(7) |
| 28.0 | X(8) | .948 | Y(8) |
| 32.0 | X(9) | .944 | Y(9) |
| 36.0 | X(10) | .938 | Y(10) |
| 40.0 | X(11) | .928 | Y(11) |
| 44.0 | X(12) | .914 | Y(12) |

TABLE LOOKUP

One procedure is to load the entire table into the storage of the computer, and then for a given value of X search the table for the corresponding value of Y.

A FORTRAN program to accomplish the loading of the above table into storage and table lookup for several values of X is shown in Figure 14. Note that the search is accomplished with the use of an IF statement within a DO loop. For the case of the argument X being exactly equal to a table entry value of X, the corresponding value of Y is simply selected and printed. When the argument X falls between two table entries linear interpolation is performed. The graphic derivation of the linear interpolation equation is:

$$y = y_i + \frac{(y_{i+1} - y_i)\,(x - x_i)}{x_{i+1} - x_i}$$



This is the equation of the straight line joining points $(x_i, y_i)$ and $(x_{i+1}, y_{i+1})$. Evaluation of the right-hand side for a particular value of x gives the corresponding value of y. This equation is used in programming statement 4.

C FOR COMMENT

| STATEMENT NUMBER | Cont. | FORTRAN STATEMENT |
|---|---|---|
| C | | TABLE LOOKUP PROGRAM |
| | | DIMENSION X(12), Y(12) |
| | | READ 10, X, Y |
| 1 | | READ 10, TX |
| | | DO 6 I = 1, 12 |
| | | IF( TX-X(I) ) 4, 5, 6 |
| 6 | | CONTINUE |
| | | GO TO 1 |
| 4 | | CORY = Y(I-1) + (Y(I)-Y(I-1))*(TX-X(I-1)/(X(I)-X(I-1)) |
| | | GO TO 8 |
| 5 | | CORY = Y(I) |
| 8 | | PRINT 20, TX, CORY |
| | | GO TO 1 |
| | | END |

Figure 14.

If the approximation of the function by a straight line in the interval $(y_{i+1} - y_i)$ is not of sufficient accuracy, the function may then be approximated by a parabola or higher-degree polynomial by such methods as the Lagrange interpolation formula.

## DATA FITTING

If the problem involving use of this table were to be run many times on a computer, consideration may be given to finding an equation which will pass either through or within tolerance of all the points in the table. In most engineering problems it is sufficient to find an equation which passes within a specified tolerance of all the points in a table of data.

### Model Selection

First a selection of the equation form to be fitted to the data must be made. A few of the possible selections are listed below.

| TYPE | EQUATION |
|---|---|
| 1. Polynomial | From $y = a_0 + a_1x$<br>To $y = a_0 + a_1x_1 + a_2x_2 + a_3x_3 + a_4x_4 + a_5x_5$<br>Generally restricted to this range. |
| 2. Logarithmic | $y = a_0 + a_1 \log x$ |
| 3. Exponential | $y = a_0a_1^x$ |
| 4. Power | $y = a_0x^{a_1}$ |
| 5. Fourier Series | $y = a_0 + \sum_{n=1}^{m} (a_n \cos nx + b_n \sin nx)$ |

The actual decision may be made on the basis of theory, some preliminary plotting, past experience, or by trial and error.

Model Fitting Criteria

Next a criterion for actual fitting of the equation form (finding values for the $a_i$) must be selected. Some of the possible methods are listed below:

Selected Points. As many sets of observation data as there are $a_i$ to be determined are substituted into the selected equation, and the resulting system of equations is solved for the $a_i$. While this method is very crude, it may be of value in situations in which available data is limited (see conical filter shell stress problem later in this section).

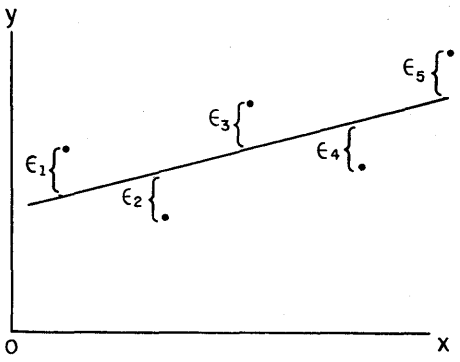
Harmonic Analysis. This widely used computer application is useful in fitting a Fourier series to a set of periodic data.

Least Squares. This is the most widely used procedure for calculating the parameters $a_i$ for the selected model.

All of the models mentioned above (Fourier series excepted) may be fitted to a set of data by least squares. The principle will be explained with respect to the simple linear model

$$y = a_0 + a_1 x$$

Given a table of n sets of data, where y is assumed to be the above linear function of x, determine $a_0$ and $a_1$.




The least squares criterion is to determine $a_0$ and $a_1$ such that the sum of the squares of the vertical distance between the data points and the straight line is a minimum. Referring to the graph, this may be stated mathematically as:

$$\sum_{i=1}^{n} \epsilon_i^2 = \text{minimum}$$

This is true only if

$$\frac{\partial \sum_{i=1}^{n} \epsilon_i^2}{\partial a_0} = 0$$

and

$$\frac{\partial \sum_{i=1}^{n} \epsilon_i^2}{\partial a_1} = 0$$

The sum may be expressed in terms of the equation to be fitted and the original data points.

$$\epsilon_i = y_i - (a_0 + a_1 x_i)$$

Then:

$$\frac{\partial \sum_{i=1}^{n} \epsilon_i^2}{\partial a_0} = \frac{\partial \sum_{i=1}^{n} \left[ y_i - (a_0 + a_1 x_i) \right]^2}{\partial a_0} = 0$$

$$\frac{\partial \sum_{i=1}^{n} \epsilon_i^2}{\partial a_1} = \frac{\partial \sum_{i=1}^{n} \left[ y_i - (a_0 + a_1 x_i) \right]^2}{\partial a_1} = 0$$

Performing the differentiation and simplifying gives:

$$\sum_{i=1}^{n} y_i - \sum_{i=1}^{n} a_1 x_i - n a_0 = 0$$

$$\sum_{i=1}^{n} x_i y_i - a_1 \left( \sum_{i=1}^{n} x_i \right)^2 - a_0 \sum_{i=1}^{n} x_i = 0$$

These two linear, nonhomogeneous equations may be solved for $a_0$ and $a_1$. The parameters $a_0$ and $a_1$, therefore, are computed in terms of sums and sums of cross products of the raw data.

## Functions of Multiple Variables

TABLE LOOKUP

Given a set of engineering data of the following form, from which x is to be computed for various sets of values of y and z:

| x | y | z |
|---|---|---|
| $x_1$ | $y_1$ | |
| $x_2$ | $y_2$ | $z_1$ |
| $x_3$ | $y_3$ | |
| $x_4$ | $y_4$ | |
| $x_5$ | $y_5$ | |
| $x_6$ | $y_6$ | $z_2$ |
| $x_7$ | $y_7$ | |
| $x_8$ | $y_8$ | |
| $x_9$ | $y_9$ | |
| $x_{10}$ | $y_{10}$ | |
| $x_{11}$ | $y_{11}$ | $z_3$ |

A table lookup with interpolation procedure, more complex logically to be sure, may again be used. This procedure may be stated as:

(a) For $z_1$ interpolate for x as a function of y alone.

(b) Store the resultant value of x along with $z_1$ in another table.

(c) Repeat steps (a) and (b) for all values of z giving a complete table of x as a function of z alone.

(d) Interpolate in this resultant table for final value of x to be used.

DATA FITTING

Quite often, particularly in such fields as chemical engineering, the table lookup possibility is impractical for one of two reasons:

1. The problem demands a fitting equation for a meaningful solution.

2. The data cannot be obtained in a form similar to that shown in the previous table.

When this situation is true, curve fitting may again be used.

For example, an equation in the description of the vehicle simulation model was presented as an empirical relationship. In this case R is assumed to be a function of three variables M, A and V.

$$R = aM + bAV^2$$

The least squares criteria may again be used in conjunction with experimental data to determine values for a and b which best fit the equation to the data.

To apply least squares relationships in this instance would require calling $AV^2$ a new variable $x_2 = AV^2$, giving a linear relationship of the form:

$$Y = ax_1 + bx_2$$

In general, data fitting of linear equations is called linear regression. If, as in this case, there are more than one independent variables, the procedure is called multiple linear regression.

This chain of first assuming the form of a relationship and then using mathematical criteria to fit the relationship to experimental data can be thought of as a search for a useful "prediction equation."

A typical application of statistical techniques to arrive at experimental relationships may be represented by the problem of fuel oil smoking. Smoking is an undesirable trait of fuel oils. If the effect of smoking can be related quantitatively to properties of fuel oil, then smoking may be controlled.

The method of attack is first to isolate the variables which affect smoking. In this case, they are aromatic content, olefin content, sulfur content, and boiling point.

An arbitrary scale for measuring smoking experimentally must be agreed upon. Then, a large number of fuel oil samples are analyzed for the four characteristics above and burned to determine smoking. This gives rise to the following table of data:

| Sample No. | Smoking Y | %Aromatics $X_1$ | %Olefins $X_2$ | %Sulfur $X_3$ | Boiling Point $X_4$ |
|---|---|---|---|---|---|
| 1 | 4.3 | 1.37 | .07 | .00 | 42.9 |
| 2 | 3.7 | 1.29 | .06 | .01 | 41.8 |
| 3 | 4.9 | 1.41 | .13 | .02 | 37.0 |
| 4 | etc. | . | . | . | . |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| . | . | . | . | . | . |

This, in turn, is subject to the following statistical analysis:

1. Regression Equation: If smoking is assumed to be a linear function of the variables, the analysis will determine the best fitting equation:

   $$Y = b_1x_1 + b_2x_2 + b_3x_3 + b_4x_4 + b_5$$

   where $b_1$, $b_2$, ..., $b_5$ are the fitting constants.

2. Partial Correlation Coefficients: The square of one of these coefficients expresses the percentage of change in smoking which is due to a corresponding change in the respective variable. In this sense it is a measure of the dependence of smoking on the variable.

3. Multiple Correlation Coefficient: This gives a measure by which it may be determined whether all significant variables have been selected.

4. Standard Error of Estimate: This gives the accuracy of prediction by the regression equation.

If the analysis indicates that the proper variables have been chosen and that the accuracy of the regression equation is sufficient, it should be possible to determine means to control smoking of the fuel oil. If not, other possible variables must be selected and new data obtained from which to determine other contributing factors.

The preceding discussion has stated that the models chosen for multiple regression must be linear models. More explicitly, this means that the partial derivative of the model function with respect to one of the parameters must be independent of that parameter. This mathematical restriction is severe for some desired applications.

The Mathematics and Applications Department of IBM has developed an approach to fitting nonlinear models. The mathematical technique is interesting in that it consists of iterating through the two steps of (1) making a linear approximation to the nonlinear model and (2) fitting the linear model by multiple regression. The result is the possibility of fitting nonlinear models, an example of which might be

$$y = \frac{a_1}{a_1 + a_2}\left(e^{a_1 x_1} + e^{a_2 x_2}\right)$$

This nonlinear estimation has been used further in fitting sets of equations to data.

## Dimensional Analysis

This technique for model selection of relationships between physical variables has wide application to engineering problems. With electronic computation the application of dimensional analysis can be greatly facilitated.

As usually written, each physical variable of an engineering equation is expressed in some combination of fundamental units. These units — time, length, mass, temperature, etc. — are the dimensions.

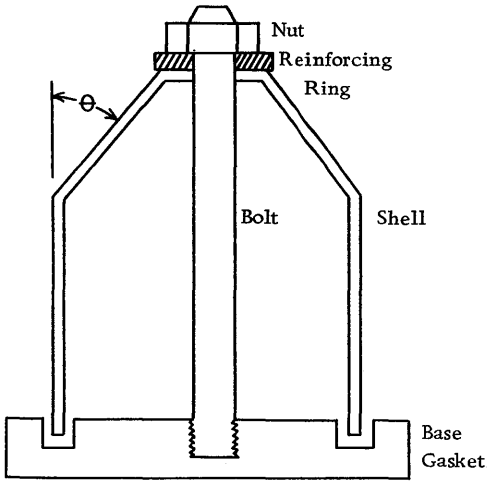
Dimensional analysis is a systematic study of the dimensions of an equation containing several variables in order to:

1. Discover any variables whose dimensions are incompatible.

2. Ascertain the nature (but not the true value) of the relationship between the variables.

3. Assemble the variables in dimensionless groups so that the relationship between these groups can be expressed algebraically.

As an engineering application of dimensional analysis which will make the procedure clear, consider a problem of conical filter shell stress. A filter shell must be designed to withstand large stresses applied through a tightened nut at the top. This force is needed to overcome the force due to oil under pressure inside the shell and maintain a firm seal at the base gasket.




An equation is desired which will predict the applied stress, W, for which the shell will fail. For this structure a complete theoretical analysis is too involved. Therefore, an empirical relationship will be determined. It is assumed that variables involved are:

S = Ultimate shear stress of the material

t = Thickness of the shell

D = Mean diameter of shell

$\theta$ = Cone angle with the vertical

d = Diameter of reinforcing ring

W = Applied stress

Dimensional analysis now consists of finding dimensionless groups of variables so that a dimensionally correct equation may be written. An approach here is to let

$$\pi_1 = S^a t^b D^c d$$

$$\pi_2 = S^e t^f D^g \theta \qquad (1)$$

$$\pi_3 = S^h t^i D^j W$$

The dimensions of each of the variables to be inserted are:

$S = MT^{-2} L^{-1}$
$t = L$
$D = L$
$d = L$
$\theta = 1$
$W = MT^{-2} L^{-1}$

M for mass
L for length
T for time

For $\pi_1$ to be dimensionless, the sum of the exponents for each fundamental dimension must equal zero.

For M, $a = 0$
For T, $a = 0$
For L, $-1a + 1b + 1c + 1 = 0$ or $b = -(1+c)$

For this particular case the resulting three equations have an infinite number of solutions. For simplicity consider:

$$c = 0$$

Then

$$b = -1$$

and $\pi_1 = S^o t^{-1} D^o d$

or $\pi_1 = (\frac{d}{t})$

Similarly, $\pi_2 = (\frac{D}{t})\ \Theta$

(Since $\Theta$ is dimensionless, this $\pi$ factor may be rewritten as two dimensionless groups.) And

$$\pi_3 = (\frac{St^2}{W})$$

At this point the variables are arranged in dimensionless groups and a relationship between the groups may be assumed.

A reasonable form to assume is:

$$\pi_3 = K\,(\pi_2)^a\ (\pi_1)^b$$

or

$$\frac{St^2}{W} = K\left(\frac{D}{t}\right)^a \left(\frac{d}{t}\right)^b \Theta^c \qquad (2)$$

where $\Theta$ has been used as a separate $\pi$ factor because it is dimensionless.

Rewriting (2) gives:

$$S = \frac{KW}{t^2}\left(\frac{D}{t}\right)^a \left(\frac{d}{t}\right)^b \Theta^c \qquad (3)$$

where K, a, b and c are constants to be determined.

One method for calculating suitable values for the constants K, a, b and c is that of selected points covered earlier in this section.

(a) Take the log of each side of (1)

$$\log S = \log (\frac{KW}{t^2}) + a \log (\frac{D}{t}) + b \log (\frac{d}{t}) + c \log \Theta \qquad (4)$$

(b) Make four shells of different dimensions with known ultimate stress, S.

(c) Crush shells to obtain W.

(d) Substitute all the values into (4) to obtain four equations in four unknowns (K, a, b, c) and solve these equations simultaneously.

(e) Apply constants to (3) and verify by testing other shells.

For improved accuracy at the expense of shells, the following procedure may be used (this is equivalent to the least squares done graphically):

(a) Holding S, t, D and d constant, crush many shells of varying $\Theta$.

For this case (4) becomes

$$- \log W = c \log \Theta + B$$

where B is a constant.

(b) Plot W versus $\Theta$ on log-log paper. The slope of the straight line is then c.

(c) Repeat a and b for other variables.

Note the possibilities for computer usage on larger problems of this kind. Simultaneous equations (to be discussed later) are handled twice. Furthermore, the procedure for improving the accuracy is a simple curve fitting problem. In fact, the whole procedure for dimensional analysis, since it is well defined, can be programmed for a computer.

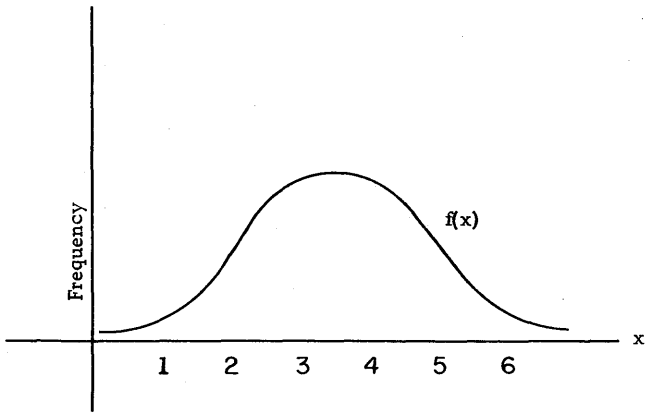
## Probabilities

The probability of a variable having a certain value is a question which often arises in engineering. If historical data on the variable is available, it is often possible to do the necessary predicting. This problem illustrates a method for handling normally distributed variables.

Given a table of observed values for a particular variable whose frequency distribution is assumed to be normal:

(a) Find the probability of a reading being less than any given value of x, and

(b) Find the probability of a particular reading lying between any two values of x.




The general distribution function is:

$$f(x) = \frac{e^{-\frac{1}{2}\left(\frac{x-m}{\sigma}\right)^2}}{\sqrt{2\pi}}$$

where m = mean
$\sigma$ = standard deviation

The standard normal curve is given by:

$$f(t) = \frac{e^{-t^2/2}}{\sqrt{2\pi}}$$

Therefore a transformation of variables will allow us to make use of standard normal distribution function. This transformation is

$$t = \frac{x-m}{\sigma}$$

The probability of a reading falling below a given value of x is given by:

$$\int_{-\infty}^{x} f(x)dx = \int_{-\infty}^{t} f(t)dt$$

where $t = \frac{x-m}{\sigma}$

The probability of a reading lying between $x_1$ and $x_2$ is:

$$\int_{x_1}^{x_2} f(x)dx = \int_{t_1}^{t_2} f(t)dt$$

where $t_1 = \frac{x_1-m}{\sigma}$

$$t_2 = \frac{x_2-m}{\sigma}$$

The problem, then, is one of evaluating the integral

$$\frac{1}{\sqrt{2\pi}} \int_a^b e^{\frac{-t^2}{2}} dt$$

A table of values of this integral for various values of t (normal areas and ordinates) is available in any statistics text. In computing, problems of this type might be tackled in two ways:

1. Store the table in memory.

2. Use numerical approximation for evaluating the integral.

An approximation by Hastings* might be used with the latter approach. It is

$$\phi(x) \cong \frac{2}{\sqrt{\pi}} \int_o^x e^{-t^2} dt \text{ for } o \leq x < \infty$$

where $$\phi(x) = 1 - \frac{1}{\left[1+a_1x+a_2x^2+a_3x^3+a_4x^4\right]^4}$$

and

$a_1 = .278393$ $\qquad$ $a_3 = .000972$

$a_2 = .230389$ $\qquad$ $a_4 = .078108$

## Exercises

1. Write a FORTRAN-language program to perform table lookup with linear interpolation for a tabulated function of two variables as described under "Functions of Multiple Variables."

2. Evaluate the coefficients for the linear least squares fit

   $y = a_0 + a_1x$

   for the following table of data

---

*Approximations for Digital Computers by Cecil Hastings, Jr., Princeton University Press, 1955.

| y | x |
|---|---|
| 14.0 | 64. |
| 13.9 | 54. |
| 13.5 | 55. |
| 13.4 | 56. |
| 12.6 | 46. |
| 12.6 | 51. |
| 11.8 | 42. |
| 11.4 | 47. |
| 11.3 | 48. |
| 10.5 | 35. |
| 10.3 | 44. |
| 10.3 | 29. |
| 9.3 | 31. |
| 7.6 | 14. |

*3. The distribution in measured diameter of the dirt particles normally deposited in a carburetor sediment bowl is a normal one.

For a mean of .0025 centimeters and a standard deviation of .0002, find the probability of a dirt particle being greater than .0027 centimeters in diameter.

Use any statistics text for necessary tables.

# SECTION VIII: OPERATIONS RESEARCH

The reason for including this as a topic of engineering analysis for computers is that often the technical personnel of a company will be turned to for answers concerning the mathematical aspects of Operations Research. Books have been written on each of the subjects to be mentioned. The intention here is merely to present examples in such a way as to make possible the recognition of problem types as they occur.

Operations Research can be thought of as the application of scientific methods, techniques or tools in the management decision area. It attempts to present a clear picture of the system under study and at the same time predict the consequences of alternate sets of actions — in other words, show the best decision. Methods for attaining either or both of these goals require knowledge of the system. Managed systems are, in general, complex, which means that vast amounts of data are required to describe these systems. Manipulation of this data to attain the stated goals generally requires a computer.

OR seems to be at the "clear picture" stage at the present time. The more complex problem of "prediction" has been hindered by the fact that

1. The mathematical and logical methods employed by OR are unfamiliar to those most intimately concerned with the problems (the executives).

2. The historical data describing the system does not exist in proper or even intelligible form.

However, consideration of the prediction level of OR while working on the clear-picture level has the following benefits:

1. Creating a clear picture of a system results in the organization of data into intelligible form. This data can later be used at the prediction level of analysis.

2. Consideration, even without use, of more advanced techniques often gives a better understanding of, or "feeling for," a system.

Almost any procedure for presenting data to an executive can be thought of as part of the clear-picture operation. However, the computer allows concise data handling in many new areas at reasonable costs. Some of these are:

1. Graphic presentation of parts failures in the field, showing effect of engineering changes on failures, failures per unit of operation, failures as a function of production group, etc., on a day-to-day basis.

2. Presentation of historical data concerning use of spare parts in the past so as to gain some insight into what an "all-time production" of a spare part should be.

3. Creation of a weekly report to a supervisor, showing jobs assigned to his shop, estimated time for each job, group within shop to which it was assigned, actual time for completed jobs, and year-to-date figures for number of jobs assigned to each group with totals for estimated and actual job times. This would allow analysis of each group's performance.

What are some of the prediction techniques presently being employed?

1. Linear Programming: This technique may be used on any system which may be represented by a set of linear equations.

   There may be more variables than equations and therefore an infinite number of solutions to the set of equations. The object of linear programming is to find a particular solution which will optimize (make maximum or minimum) another linear function (usually a profit or cost function). This technique has been very successful in the blending of petroleum products and the mixing of grain feeds. It is not unusual for the mathematical model of a gasoline refinery to be as large as 100 equations in 500 variables. The actual mathematical techniques employed for linear programming will not be discussed here. However, it is worth noting that the techniques have been refined to the point where their application can be rather mechanical. That is to say, a knowledge of the mathematics is not required for successful use of linear programming.

   An important special case of linear programming is the transportation problem. It may be used to optimize any system for which a commodity is available at several sources and is required at several destinations, and for which the transportation rate for each possible source-to-destination combination is known. The most economical shipping assignment will be determined.

2. Forecasting: Historical data is analyzed statistically to enable simultaneous forecasting of many interrelated factors of management interest in terms of other known variables that affect them. Although, in general, the statistical techniques are not new, the computer allows consideration of many more variables and, therefore, the handling of complex management systems.

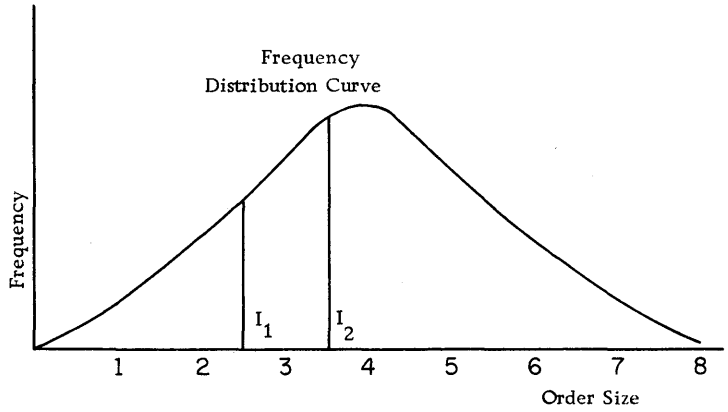
3. Simulation: Here a mathematical and logical model of the system in question is created. This model is operated according to the normal rules of the system operation for a specified period of time. Concise data on the results is made available. This gives a picture of how the system will operate — quickly, and in clear, concise form. More important, the system may be altered or the rules changed and the model run again; thus, the effect of alternative decisions can be tested on paper.

   The term "simulation" was introduced earlier in the discussion of vehicle simulation. In the present context, however, the system simulated can be a company organization and/or its manufacturing tools — a much larger concept.

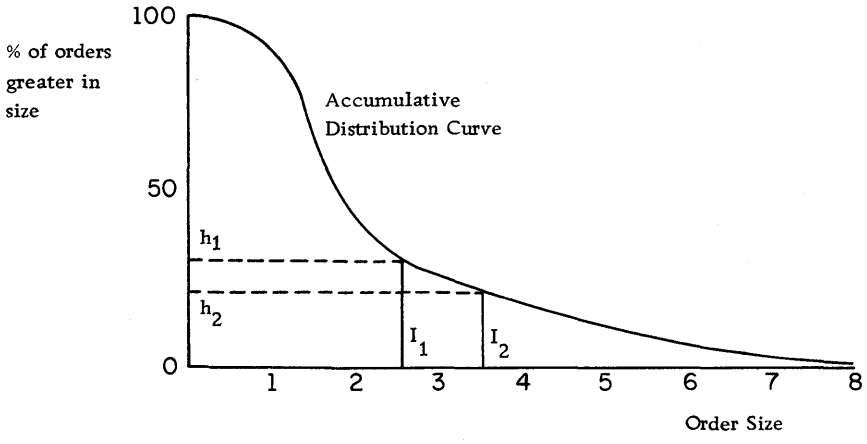
Some discussion of the techniques of simulation is in order here, since, while the problems tackled via simulation may be extremely complex, the basic mathematical elements are quite simple.

There are two basic elements of any simulation, the first being the reproduction of actual (or perhaps expected) events. This can be done either by using actual historical data (for example, past sales record by month) or by using statistical parameters which describe the desired behavior.

To reproduce arrival of orders by size when the past




frequency distribution of order sizes is known, an accumulative distribution curve is constructed as the first step.




Using random numbers (generated or taken from a table) to represent the "% of orders greater in size," the corresponding order size can be read from the curve or obtained from a table or equation representing the curve.

This use of random numbers and the accumulative distribution curve will reproduce the behavior pattern from which the curve was constructed. Further, each order size will occur at random within the overall pattern. This is exactly what is desired in simulation.

The second element in simulation of a system is time. Two distinct approaches have been used here:

1. The system is reviewed at definite intervals of time and all activities are updated at each review. A disadvantage of this approach is that for intervals of review, no activity may have occurred. However, many systems with natural periods can be simulated nicely using this procedure. Simulation of inventory and truck fleet operations fits this pattern.

2. The system is reviewed only when something important happens. These important points in time may be recorded in a time status record (TSR). Initially a distribution-of-events curve is consulted for the entry to the TSR. Subsequently each time an entry to the TSR is handled by updating the activity, the first step is to sample the events distribution for the next TSR entry.

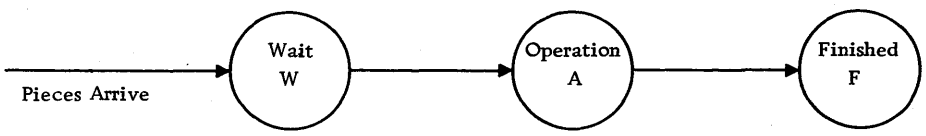
## Job Shop Simulation

A job shop is distinguished from an assembly-line shop by its flexibility in processing articles of manufacture. The amount of time required for a given operation varies from article to article and the route of operations varies from article to article.

Important advantages may often be realized through simulation of a job shop on a computer, considering such things as machines available, labor available, operating costs, fluctuation in orders, priority of orders, operating rules, etc. Records of order completions, in-process inventory, and waiting lines in front of machines can be maintained. The effect of proposed changes in the physical plant or operating rules may be pretested using such a simulator.

A short description of the logic which goes into the simulation of a single operation job shop will illustrate the basics of this type of simulator. It illustrates the use of probability distributions and the time status record approach of handling the time element.

The logic diagram for a single operation job shop might be as follows:




The information required to simulate this system would be:

1. The probability distribution of time between the arrival of pieces.

2. The probability distribution of operation time per piece.

The results desired would be:

1. The number of pieces which can be completed per unit time.

2. The amount of idle time experienced per unit time.

3. The average waiting time per piece.

On the basis of the two probability distributions, a time status record is generated which guides the simulation program. The procedure for manipulating the model is shown in Figure 15.

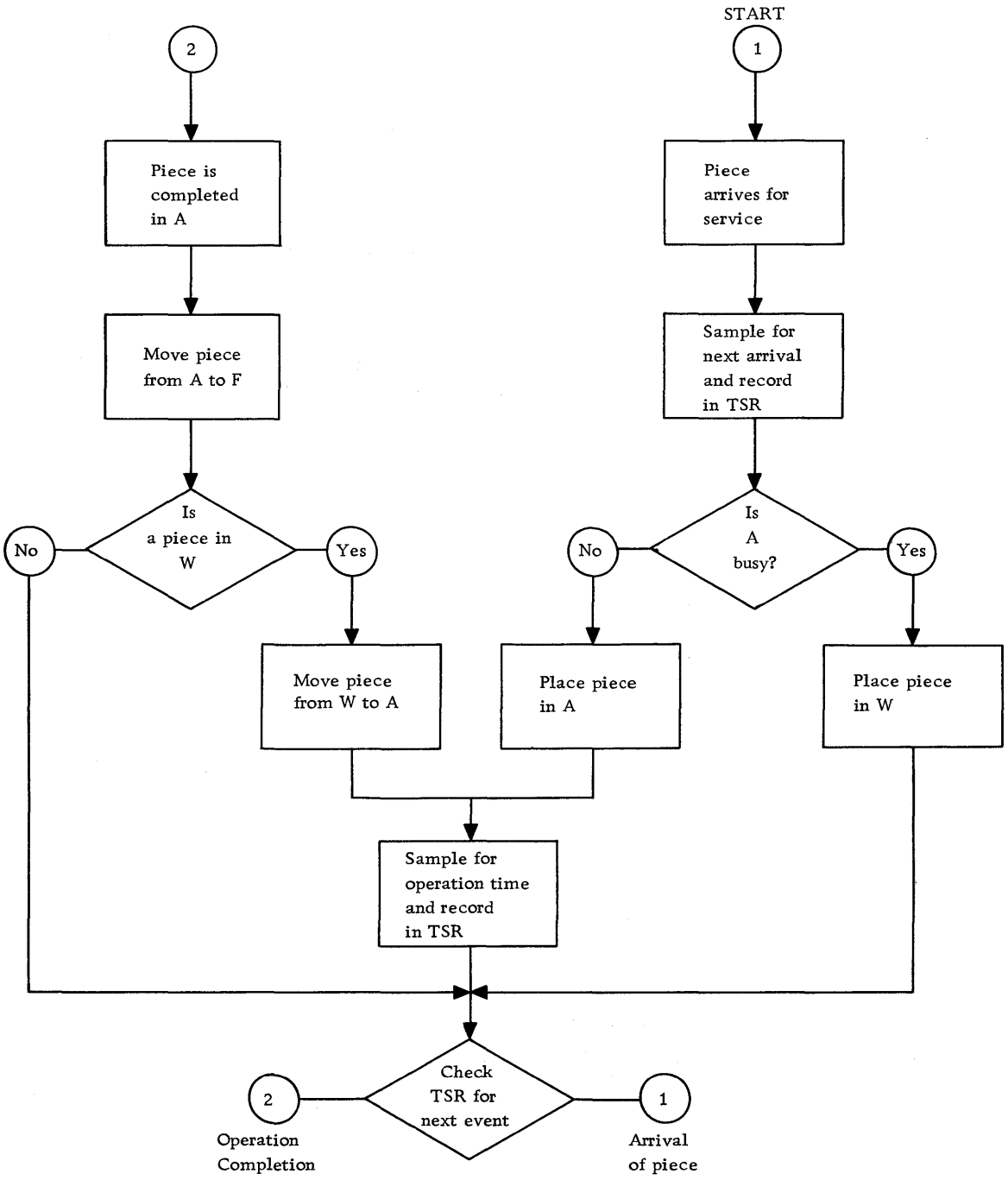



Figure 15. Job Shop Simulation

Two curves which might represent the required distributions are shown in Figure 16 along with a short, random sequence of numbers. Figure 17 shows the time status record, and the associated recordkeeping which might be performed by the computer in simulating the operation.

The conditions for beginning this simulation are:

(a) Line 1 represents the status of the one operation shop at the beginning of the day.

(b) Lines 2 and 3 are entries to the TSR made during the previous day.

(c) Lines 4 and beyond represent entries generated during this day's simulation.

(d) The simulation of this day's events begins at 9:00 a.m. (the first event) and can be followed in the logic chart beginning at the point marked START.

This simple statistical technique for accomplishing simulation can be used for a wide variety of operations research problems.

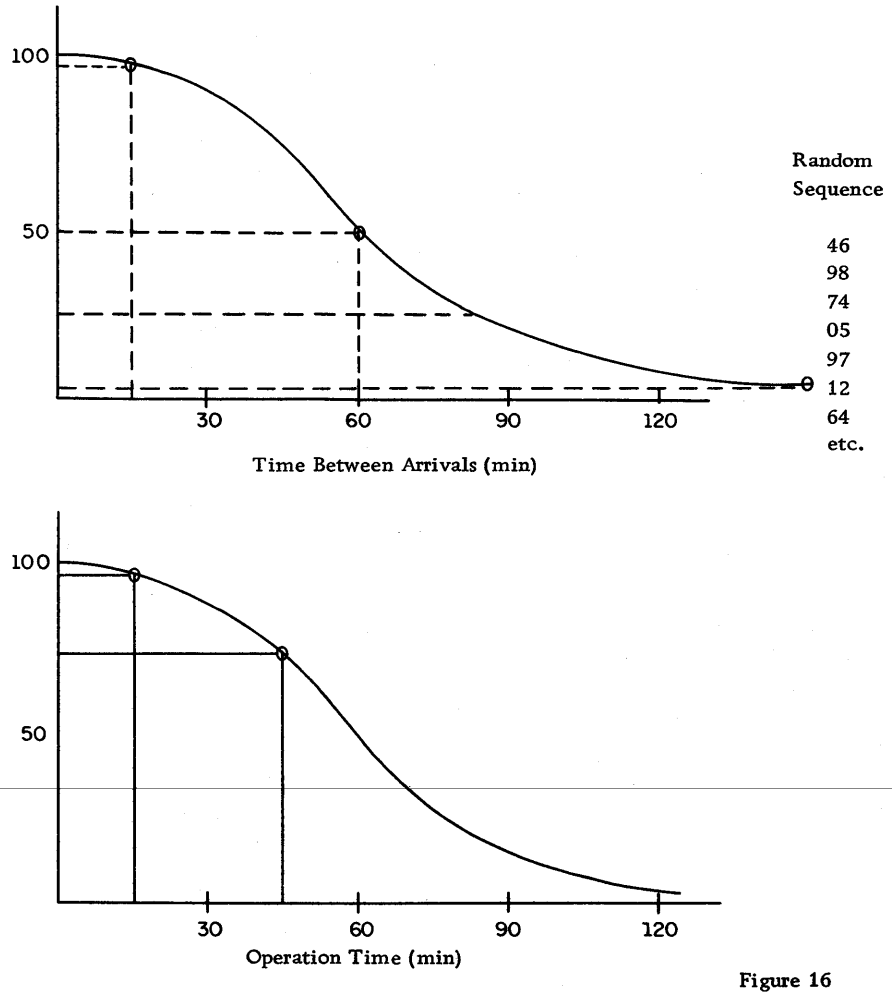



Figure 16

TIME STATUS RECORD

| | TIME | EVENT | W | A | F | $\sum W$ | Random table number used for next event's time |
|---|---|---|---|---|---|---|---|
| 1 | 8:00 | | 1 | 1 | | | |
| 2 | 9:00 | Arrival | 2 | 1 | | 1.0 | 46 |
| 3 | 9:30 | Completion | 1 | 1 | 1 | 2.0 | 98 |
| 4 | 9:45 | Completion | 0 | 1 | 2 | 2.5 | 74 |
| 5 | 10:00 | Arrival | 1 | 1 | 2 | 2.5 | 05 |
| 6 | 10:30 | Completion | 0 | 1 | 3 | 2.75 | 97 |
| 7 | 12:30 | Arrival | 1 | 1 | 3 | 2.75 | 12 |
| 8 | 12:45 | Arrival | 2 | 1 | 3 | 3.00 | 64 |

Figure 17

## Inventory Control

Inventory control is important to industry for sizable dollar reasons. It is an important area of effort computer-wise because (1) much of the recordkeeping required for inventory control is presently being done by data processing systems, and (2) many problems of inventory control can be tackled by those mathematical treatments which are greatly facilitated by use of computers.

Inventory control, from a data processing viewpoint, is best broken into two distinct procedures:

1. Forecasting the future sale of items from inventory.

2. Use of the resulting forecast figures in a control procedure which will maintain that level of inventory which best meets the requirements laid down by management operating decisions.

To take a hypothetical company in order to illustrate a possible study of these two areas, assume that the Best Parts Company has a central warehouse which supplies retail dealers. The situation contains the following factors:

1. The inventory status must be reviewed each week. Total sales for each item are recorded and needed replenishments of stock are ordered at each review.

2. In practice, all replenishment orders to the plant can be filled within a lead time of 1 1/2 weeks.

3. The activity of different parts varies greatly.

4. There are no largely seasonal parts.

5. Two years of sales records are available. This record gives the total sales of each part for each one-week period.

Best Parts Company management believes that it can improve the control of the inventory in one or more of three ways: (1) reducing inventory, (2) reducing out-of-stock conditions, and (3) giving simpler processing of control information.

A study of the control procedure is initiated with the understanding that the computer facility, with whatever new techniques available, may be used. The study takes the following form:

Forecasting

Four approaches to arriving at the forecast sales to retailers for each part are considered:

1. The use of linear regression in finding a prediction relationship of the form

$$\text{Forecast Sale} = a_o + a_1x_1 + a_2x_2 + \text{----} a_nx_n$$

where the x's may be factors felt to correlate with or influence the sales, such as gross national product, number of people in $5,000 - $10,000 income bracket, etc. This approach requires extensive analytic effort, much diverse data, and large amounts of computing effort.

2. Classical time series analysis gives a forecast based only on analysis of previous sales data. The linear trend line of the form

$$\text{Forecast Sale} = a + bT$$

where T is time in sales periods, requires only past sales data, but considerable computing for fitting of a and b.

3. A common practice in industry is to use a moving average for sales forecasting. The formula for a four-week moving average is

$$F_{r+1} = \frac{S_r + S_{r-1} + S_{r-2} + S_{r-3}}{4}$$

where $F_{r+1}$ is the forecast sales for week r+1 and the S's are the actual sales for the previous four weeks. This method is simple, data-wise and computation-wise. A major disadvantage is that it does not generate statistics which are found important in the control procedure — in particular, the statistic called the standard deviation of actual sales from the forecast. Approaches 1 and 2 may generate this statistic.

4. A weighting technique credited to Robert K. Brown and called exponential smoothing has gained attention and application in the past few years.

Forecasting is based, as in the moving average, on past sales data and requires even less computing. A major advantage is that a simply computed estimate of the required standard deviation is available and corrected to the most recent sales review figures.

The exponential smoothing procedure is chosen for study on the basis of limited available data and limited computing time. The forecast equations for exponential smoothing may be given as

$$P_i = P_{i-1} + \alpha\ (S_{i-1} - P_i^*{}_{-1}) \qquad \text{(Demand)}$$

$$T_i = \alpha\ (P_i - P_i^*{}_{-1}) + (1 - \alpha\ )\ T_{i-1} \qquad \text{(Trend)}$$

$$P_i^* = P_i + \frac{(1-\alpha)}{\alpha}\ (T_i) \qquad \text{(Forecast)}$$

$$D_i = \alpha\ (S_{i-1} - P_i^*{}_{-1}) + (1 - \alpha\ )\ D_{i-1} \qquad \text{(Deviation)}$$

where

$S_{i-1}$ = sales figure for period just ending.

Then

$P_i$ = demand expected for next period

$T_i$ = trend measure of whether last several periods have seen a rise or fall

and

$P_i^*$ = trend adjusted, final forecast, for next period sales

with

$1.25\ D_i$ = estimate of standard deviation of actual sales from forecast.

The constant $\alpha$ represents the weighting choice. Without the trend adjustment, the demand ($P_i$) computed will closely approximate the moving average based on N periods if $\alpha$ is chosen as

$$\alpha = \frac{2}{N+1}$$

With the use of the trend adjustment, experience has shown that a choice of $\alpha = .1$ is most satisfactory.

Control Procedure

The key to inventory control is in the way one parameter is computed. This parameter is the requirements figure. The requirements figure must be arrived at in such a way that it relects (1) the knowledge of actual sales, so that stock is available to meet the demand, and (2) management's ideas of what is adequate by computing the proper stock level, within a

defined approximation, so that out-of-stock conditions are kept below a set level.

This implies that management must set the level for out-of-stock conditions — and this is true. The familiar quality decisions such as "we must have few out-of-stock conditions" or "we must reduce inventory" must be replaced with decisions such as "an average of 5% out-of-stock is to be our goal."

Literature is available on methods of arriving at a proper out-of-stock percentage figure which tends to minimize inventory operating costs for particular inventory situations.

In this example, the management percentage level figure is assumed available.

One approach to defining the necessary requirements equation is to construct an inventory model similar to that shown below. This model assumes the following "ideal" conditions:

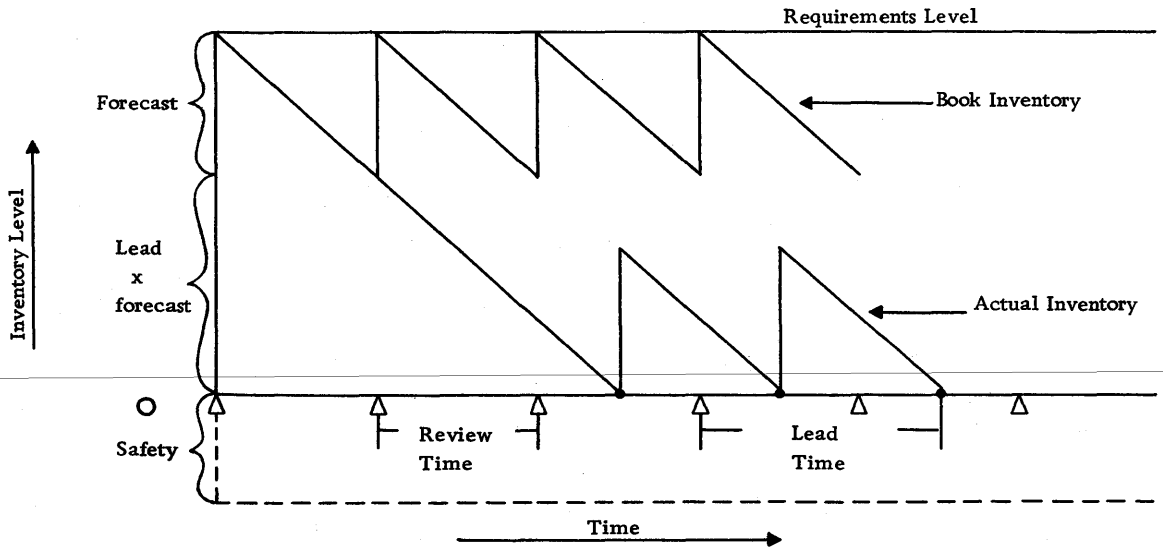
1. It represents a possible inventory level as a function of time for one item.

2. The sales are linear — the same sales figure for each period.

3. The sales in each period are exactly the amount forecast for that period.

4. The lead time is 1 1/2 review periods (lead time being the time between generation of an order and actual receipt of replenishment stock).

The graph begins at a point in time where the actual inventory is exactly at the requirements level.



The upper curve represents the inventory actually in the warehouse at any given time plus the inventory on order.

The graph shows that for a requirements figure computed strictly in terms of the forecast and the lead time, zero stock conditions are to be expected. Since the Best Parts Company knows that it is not operating under "ideal" conditions (sales fluctuate from period to period), a safety stock is needed. The final requirements equation then is:

$$R_i = P_i^* + \text{Lead} \cdot P_i^* + \text{Safety}$$

The Safety Stock Calculation

Reference must be made here to the discussion of probabilities in the chapter on empirical relationships. If a frequency distribution of errors in the forecast over a period of time is plotted, it might look like Figure 18 (assuming a normal distribution).

The accumulative area under the curve, as we move from left to right along the error axis, represents the probability of the error in forecast being equal to or less than the corresponding value of the error.

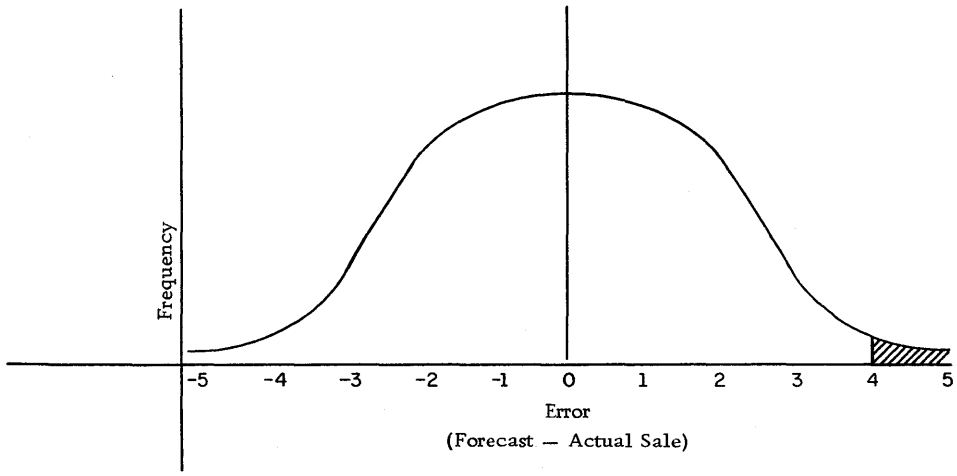



Figure 18

If the shaded area in Figure 18 represents 5% of the total area, then only 5% of the time is the error in forecast greater than four units.

Therefore, if four units are always added to the inventory stock requirements, the out-of-stock conditions should occur only for errors larger than four units — or only 5% of the time.

For computational purposes, the same proper error value for a stated probability may be obtained by using the transformation equation:

$$t = \frac{x-m}{\sigma}$$

where for this use

t = value corresponding to required probability level and is taken from the tabulated values available for the normal distribution

x = desired error value

$\sigma$ = standard deviation of error

m = mean of the errors (in this case m = o)

Then

$$x = \sigma t$$

and since $1.25\ D_{i-1}$ is an estimate of $\sigma$, the safety stock required may be computed as

$$x = 1.25\ D_i\ *t$$

The value of t corresponding to a probability level allowing 5% out-of-stocks is 1.65. The final requirements equation to be used by Best Parts Company is

$$R_i = P_i* + \text{Lead} \cdot P_i* + 1.65 \cdot 1.25 \cdot D_i$$

Note that the statistic D, like all the other quantities here, is computed for each item in the inventory. The requirements figure for an item level is thus dependent on that item's own past sales behavior. Items with steady sales will have small safety stocks. Items with widely fluctuating sales will have large safety stocks. This may often make possible the seeming paradox of reducing overall inventory and at the same time reducing out-of-stock situations.

This completes the two basic selections needed in establishing an inventory control procedure — a forecasting technique and a requirements computing technique. All other quantities relating to inventory control can be computed each period in terms of the forecast and the requirements. For example, the replenishments equation for the inventory might be given as:

ORDER = REQUIREMENTS - ON HAND - ON ORDER

This procedure requires extensive testing before it may be successfully initiated. One way of accomplishing this testing is to simulate on a computer what would have happened if the procedure had been used over some time interval in the past. A simulation of inventory control has two basic elements: actual events and time. Actual sales data (quantity of sales of each item or each period in the time interval) is all that is needed to satisfy the reproduction of actual events elements. The normal use of a review period in inventory control makes it a natural for the use of review at intervals in handling the time element.

A logic flow chart for a computer program to simulate the inventory model for a single inventory item is shown in Figure 19.

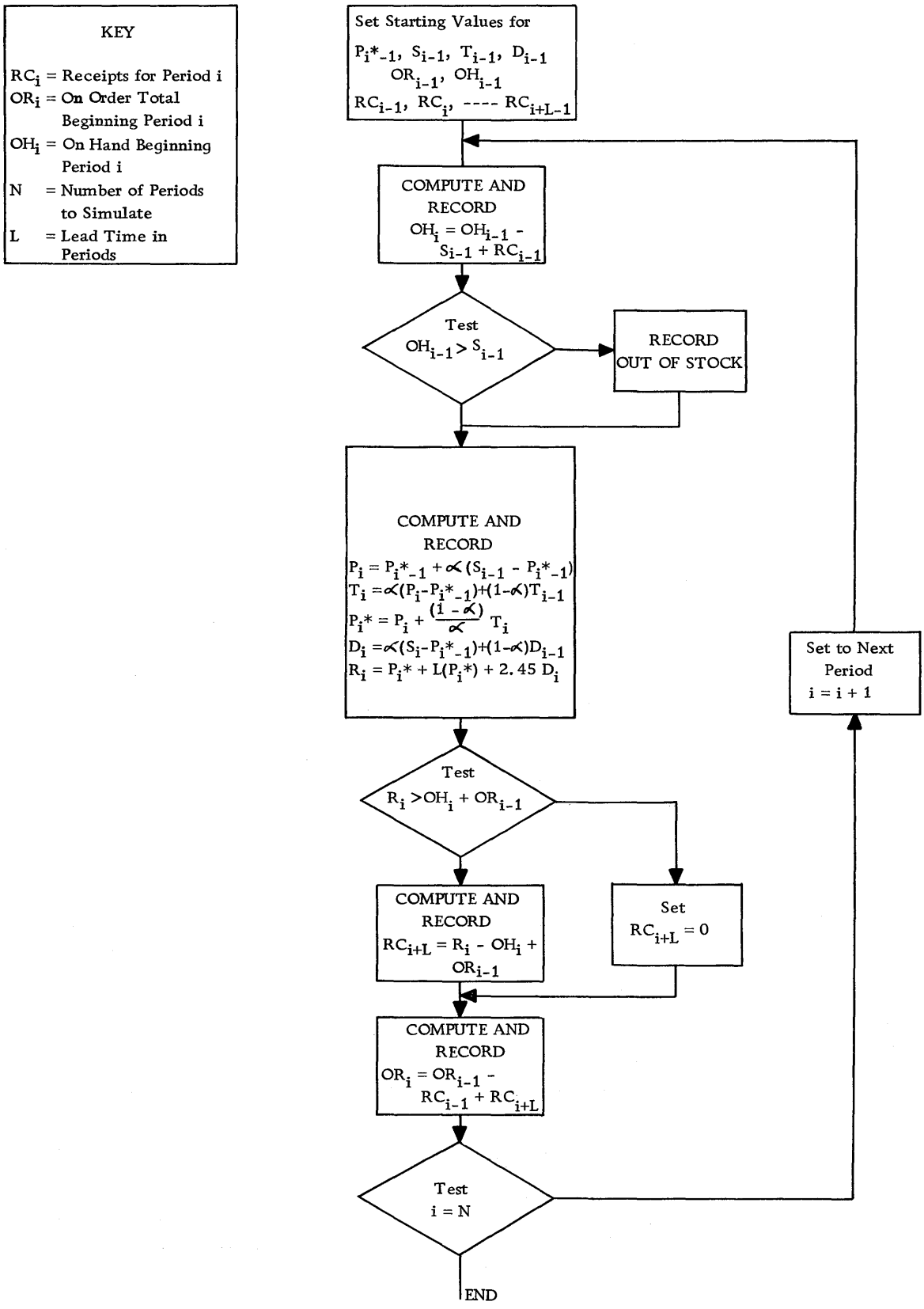

KEY
$RC_i$ = Receipts for Period i
$OR_i$ = On Order Total Beginning Period i
$OH_i$ = On Hand Beginning Period i
N = Number of Periods to Simulate
L = Lead Time in Periods
Set Starting Values for
$P_i^*{}_{-1}$, $S_{i-1}$, $T_{i-1}$, $D_{i-1}$
$OR_{i-1}$, $OH_{i-1}$
$RC_{i-1}$, $RC_i$, ---- $RC_{i+L-1}$
COMPUTE AND RECORD
$OH_i = OH_{i-1} - S_{i-1} + RC_{i-1}$
Test
$OH_{i-1} > S_{i-1}$
RECORD OUT OF STOCK
COMPUTE AND RECORD
$P_i = P_i^*{}_{-1} + \alpha(S_{i-1} - P_i^*{}_{-1})$
$T_i = \alpha(P_i - P_i^*{}_{-1}) + (1-\alpha)T_{i-1}$
$P_i^* = P_i + \frac{(1-\alpha)}{\alpha} T_i$
$D_i = \alpha(S_i - P_i^*{}_{-1}) + (1-\alpha)D_{i-1}$
$R_i = P_i^* + L(P_i^*) + 2.45\ D_i$
Test
$R_i > OH_i + OR_{i-1}$
COMPUTE AND RECORD
$RC_{i+L} = R_i - OH_i + OR_{i-1}$
Set
$RC_{i+L} = 0$
COMPUTE AND RECORD
$OR_i = OR_{i-1} - RC_{i-1} + RC_{i+L}$
Test
i = N
Set to Next Period
i = i + 1
END

Figure 19

An important result of such a simulation is the data needed to prepare charts similar to that shown in Figure 20.

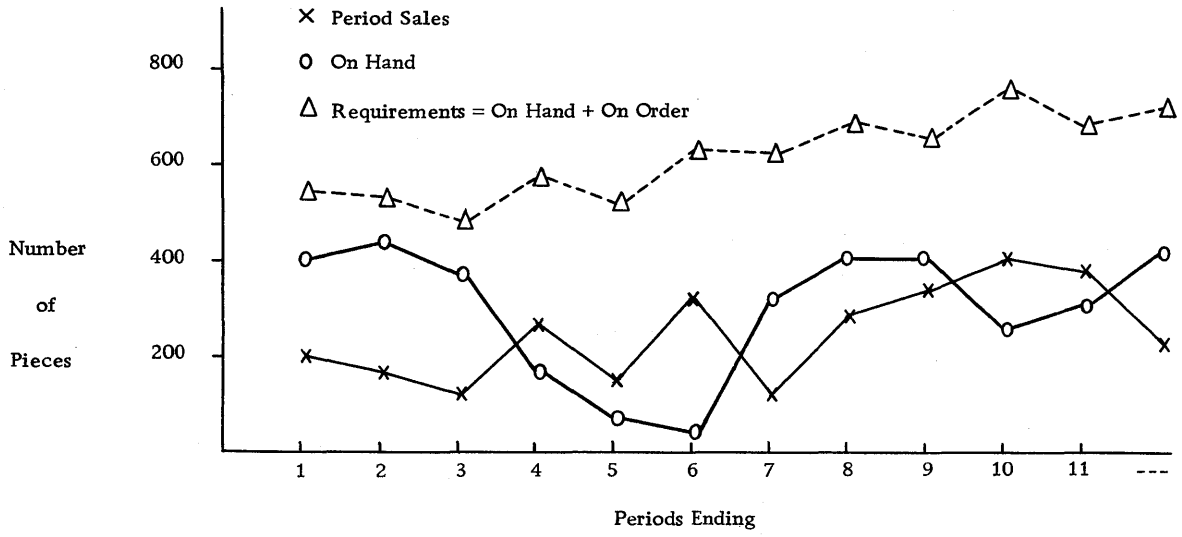



Figure 20

A graphic display of this kind, showing what will happen if a procedure is adopted, represents an excellent pretesting of the practicality of the procedure.

## Linear Programming

Mathematically, linear programming is finding a solution of a set of equations similar to:

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 + a_{15}x_5 = b_1$$

$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 + a_{25}x_5 = b_2$$

$$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 + a_{35}x_5 = b_3$$

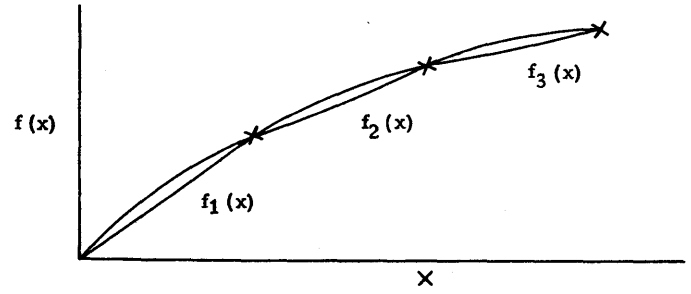
in which we have more variables than equations, and therefore an infinite number of solutions.

The particular solution chosen is the one which maximizes or minimizes an additional equation (generally a cost equation) which might look like:

$$\text{cost} = c_1x_1 + c_2x_2 + c_3x_3 + c_4x_4 + c_5x_5$$

Any system (either physical, such as a petroleum refinery, or operational, such as work assignments, or a combination of both) which may be described in terms of linear relationships may be studied by linear programming.

Many relationships which are not linear may be approximated by combinations of linear segments as illustrated graphically in the accompanying figure.




Much of the present-day power of linear programming is due to its having been taken out of the hands of the mathematicians and reduced to a fairly common engineering procedure. Rather than working with sets of equations, the engineer works with an array or matrix which is the format required for entering data to the computer. In place of mathematical operations in creating this matrix, the engineer employs straightforward rules.

| | Variables / Restrictions | Straight Run Gasoline | Alkylate | Base Stock | Butane | Reformate One | Reformate Two | Crude One | Crude Two | Lead One | Lead Two | Lead Three | Excess | Requirements |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Profit Coeff. | 5.0 | 5.0 | 5.0 | 5.0 | 4.80 | 4.60 | -2.80 | -3.10 | -.092 | -.092 | -.092 | -10.0 | |
| 1 | Octane | .3 | -.7 | -.1 | -.09 | -.1 | -.3 | | | -.1 | -.3 | -.7 | | 0 |
| 2 | Boiling Point | .8 | .3 | | .248 | -.7 | -.7 | | | | | | | 0 |
| 3 | Vapor Pressure | 1.0 | -.5 | -1.0 | 2.6 | -2.0 | -2.0 | | | | | | | 0 |
| 4 | Quantity | 1.0 | 1.0 | 1.0 | .1 | 1.0 | 1.0 | | | | | | -1.0 | 1.0 |
| 5 | Lead 1 | -1.5 | -1.5 | -1.5 | -.15 | -1.5 | -1.5 | | | 1.0 | | | | 0 |
| 6 | Lead 2 | -1.0 | -1.0 | -1.0 | -.1 | -1.0 | -1.0 | | | | 1.0 | | | 0 |
| 7 | Lead 3 | -.5 | -.5 | -.5 | -.05 | -.5 | -.5 | | | | | 1.0 | | 0 |
| 8 | Reformer Charge | | | | | 1.11 | 1.25 | -.15 | -.2 | | | | | .5 |
| 9 | Reformer Capacity | | | | | 1.11 | 1.25 | .05 | | | | | | .6 |
| 10 | Butane | | | | .1 | -.03 | -.04 | | -.01 | | | | | .05 |
| 11 | Straight Run Gasoline | 1.0 | | | | | | -.1 | -.15 | | | | | .15 |

Figure 21.

A look at the array in Figure 21, which represents a complex problem to be optimized, will illustrate the use of such rules.

This array represents the statement of the problem to be optimized. The objective will be to determine a recipe for blending available petroleum components to obtain a motor fuel meeting several specifications — and return maximum profit to the refinery.

Each horizontal row in the array represents a restriction (equation) — and each vertical column a variable. The actual numbers in the array are

coefficients which would appear in the corresponding set of linear equations which are required to describe the problem mathematically. The variables are the components to be blended. The restrictions may be of two types:

1. Quantity restrictions — for example, where the amount of motor fuel to be produced is restricted.

2. Qualitative restrictions — for example, where the Reid vapor pressure of the resulting fuel is restricted.

Line 4 in the body of the array is read: 1.0 times amount of straight run gasoline to be used in the blend, plus 1.0 times amount of alkylate to be used in the blend, plus----------, minus 1.0 times excess motor fuel, must be equal to the 1.0 barrels of motor fuel which is required. The coefficients can be entered to the array without ever putting down the equation stated above for this quantitative restriction.

Row 1 represents the representation of the octane number restriction. The refinery sets a minimum level for the resultant octane number. In this case, let the minimum specification be S and let the actual octane ratings of the variables $X_i$ to be blended be $I_i$ for i = 1, 2 ----------n where n is the number of variables.

The assumption is that the resultant blend octane rating will be a weighted average by volume of the individual specification or:

$$S + P = \frac{\sum_{i=1}^{n} I_i X_i}{\sum_{i=1}^{n} X_i}$$

where the P allows for the weighted average to be greater than the minimum specifications. A suitable form of the above equation is:

$$\sum_{i=1}^{n} (S - I_i) X_i + P \sum_{i=1}^{n} X_i = 0$$

The term $P \sum_{i=1}^{n} X_i$ is called the slack.

Note that the coefficients for each of the variables is to be $S-I_i$ or the minimum specification minus the individual specification.

This characteristic allows two rules for automatically entering coefficients into an array to handle most quality restrictions:

1. The magnitude of the coefficient is to be the absolute value of the difference between the required specification and the individual specifications.

2. The sign of the coefficient is chosen positive if the individual specification exceeds the required specification, negative if the individual specification is below the required specification.

Linear programming codes exist for most of the digital computers in commercial use. Once the array has been determined to adequately describe the system under study, the data represented by the array is presented to a computer along with the linear program.

The output to be expected from the optimization computation is:

1. Variable activities. This is a list of those variables chosen to satisfy the restrictions. With each variable is given the amount of the variable to be used in the resultant product.

   Many linear programs also indicate the range, if any, through which this amount may be varied, leaving the solution at a maximum or minimum as the case may be.

2. Shadow prices. For those variables not selected as part of the solution, an indication of the reason is given by the shadow prices. The shadow price for an unselected variable represents the cost of forcing one unit of the variable into the solution, considering the adjustments that would be necessary on other variables to again satisfy the stated restrictions. Again there is often indicated a range through which the cost or profit coefficient for the unselected variable may be moved without removing the optimization.

   Results of linear programming computation are used as a recipe for day-to-day selection of alternatives in such diverse areas as gasoline blending of available stocks, mixture of grain for feed, blending of meats for sausage, and slitting of paper rolls and fabric stock.

   The technique is also used on a study basis to aid long-range planning in the selection of refinery crude stocks, the selection of machine components in process industries, the scheduling of machine loads, and other areas.

SUMMARY ON OPERATIONS RESEARCH

As a brief summary of Operations Research, there are three techniques which have gained considerable acceptance and use in the study of man/machine systems: linear programming, forecasting, and simulation.

Linear programming computes the optimum choice of decision alternatives for a linear system within the available definition of that system.

Forecasting by statistical means has important uses in long-range planning and in day-to-day control of such things as inventory.

Simulation may be used to study systems thus far undescribed mathematically, by reproducing the logical operation of the system in testing alternate decisions.

## Exercise

1. A company would like to simulate the operation of a fleet of trucks which deliver from one factory to many parts of the country. The delivery is to be made for one product. The company has the following data which is important to the simulation:

| Day | Number of Orders (Units of 1 Truckload) |
|---|---|
| 1 | 100 |
| 2 | 97 |
| 3 | 115 |
| 4 | 40 |
| 5 | 80 |
| . | . |
| . | . |
| . | . |
| . | . |
| . | . |
| . | . |
| 150 | 102 |

| Number of Days Required to Deliver and Return | Number of Times This Period Was Required |
|---|---|
| 1 | 200 |
| 2 | 500 |
| 3 | 700 |
| 4 | 2000 |
| 5 | 4000 |
| 6 | 5500 |
| 7 | 3500 |
| 8 | 3000 |
| 9 | 500 |
| 10 | 300 |
| 11 | 40 |
| 12 | 0 |

   a. Sketch the distribution curve which might be used in simulation of the receipt of orders.

   b. Sketch the distribution curve which might be used to determine the length of time required to deliver each order (turnaround time) during the simulation.

The company would like to determine the best number of trucks to use to satisfy the delivery requirements while keeping the truck fleet cost as low as possible.

The company estimates that it costs D dollars to maintain and operate one truck each year, and that the cost of back orders in terms of future

sales lost is:

$$C = AN_1 + BN_2$$

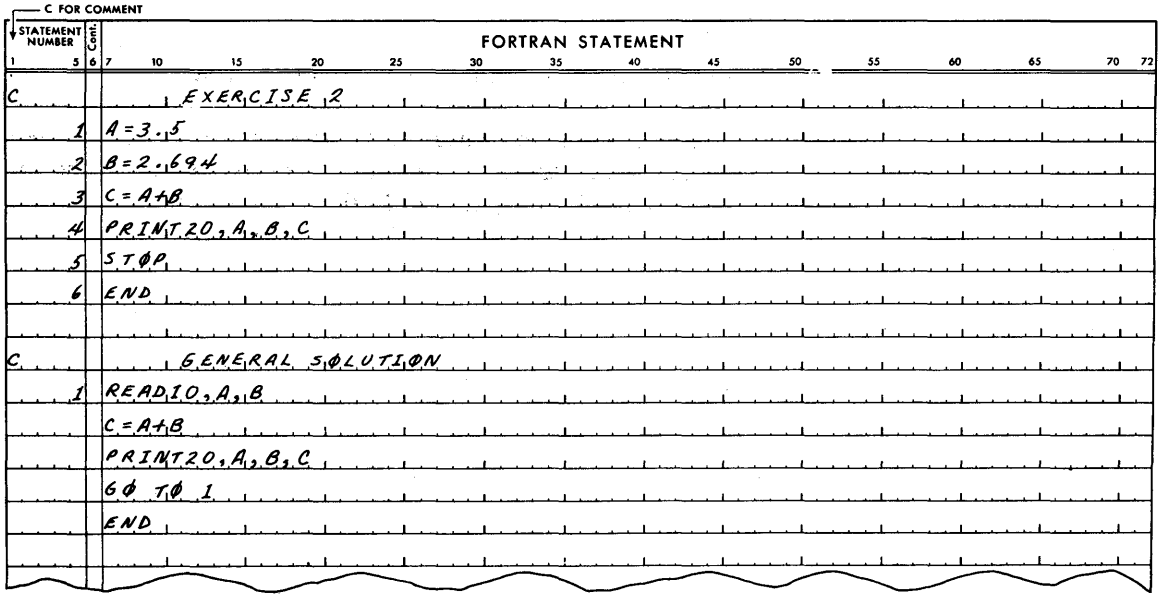
where A and B are constants, $N_1$ is the number of back orders not more than seven days old, and $N_2$ is the number of back orders over seven days.

Deliveries may be made every day of the year.

c. Draw a logic flow chart of a simulation program to operate for one year, which the company might use to determine the optimum number of trucks to maintain in the truck fleet.

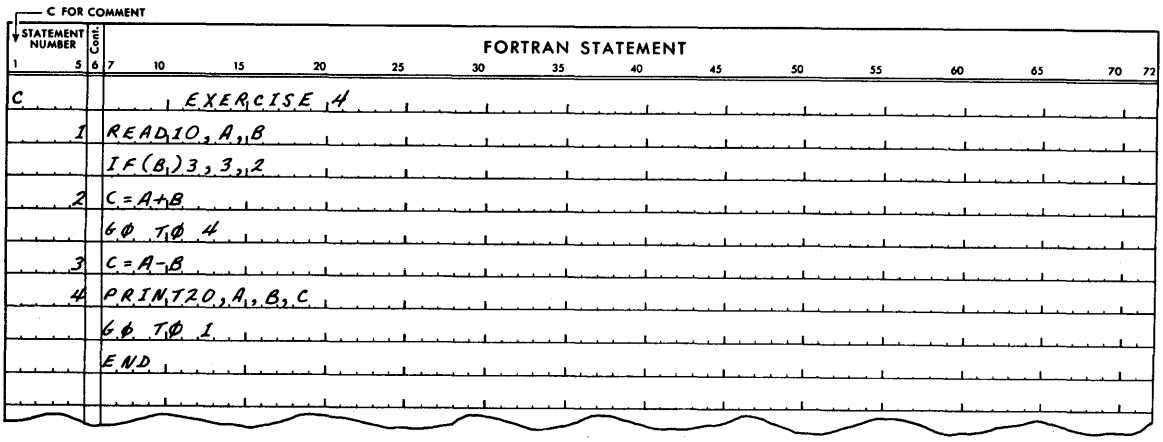
# APPENDIX I: ANSWERS TO SELECTED EXERCISES

## Section II



```
C FOR COMMENT
STATEMENT NUMBER | Cont. | FORTRAN STATEMENT
1 5 | 6 | 7 10 15 20 25 30 35 40 45 50 55 60 65 70 72
C       | EXERCISE 2
    1   | A=3.5
    2   | B=2.694
    3   | C=A+B
    4   | PRINT20,A,B,C
    5   | STOP
    6   | END
C       | GENERAL SOLUTION
    1   | READ10,A,B
        | C=A+B
        | PRINT20,A,B,C
        | GO TO 1
        | END
```

Section 2. Exercise 2



```
C FOR COMMENT
STATEMENT NUMBER | Cont. | FORTRAN STATEMENT
1 5 | 6 | 7 10 15 20 25 30 35 40 45 50 55 60 65 70 72
C       | EXERCISE 4
    1   | READ10,A,B
        | IF(B)3,3,2
    2   | C=A+B
        | GO TO 4
    3   | C=A-B
    4   | PRINT20,A,B,C
        | GO TO 1
        | END
```

Section 2. Exercise 4

C FOR COMMENT

| STATEMENT NUMBER (1–5) | Cont. (6) | FORTRAN STATEMENT (7–72) |
|---|---|---|
| C | | EXERCISE 5 |
| | | I = 1 |
| 3 | | PRINT 20, I |
| | | I = I + 1 |
| | | IF (I - 100) 3, 3, 4 |
| 4 | | STOP |
| | | END |
| | | |
| C | | ANOTHER WAY |
| | | |
| | | DO 2 I = 1, 100 |
| 2 | | PRINT 20, I |
| | | STOP |
| | | END |

Section 2. Exercise 5

C FOR COMMENT

| STATEMENT NUMBER (1–5) | Cont. (6) | FORTRAN STATEMENT (7–72) |
|---|---|---|
| C | | EXERCISE 7 |
| 1 | | READ 10, A, B, C |
| 2 | | D = B*B - 4.0*A*C |
| 3 | | IF (D) 8, 4, 4 |
| 4 | | ROOT1 = (-B + SQRTF(D))/(2.0*A) |
| 5 | | ROOT2 = (-B - SQRTF(D))/(2.0*A) |
| 6 | | PRINT 20, A, B, C, ROOT1, ROOT2 |
| 7 | | GO TO 1 |
| 8 | | REAL = -B/(2.0*A) |
| 9 | | CMPLX = SQRTF(-D)/(2.0*A) |
| 15 | | PRINT 20, A, B, C, REAL, CMPLX, D |
| 11 | | GO TO 1 |
| 12 | | END |

Section 2. Exercise 7

## Section III

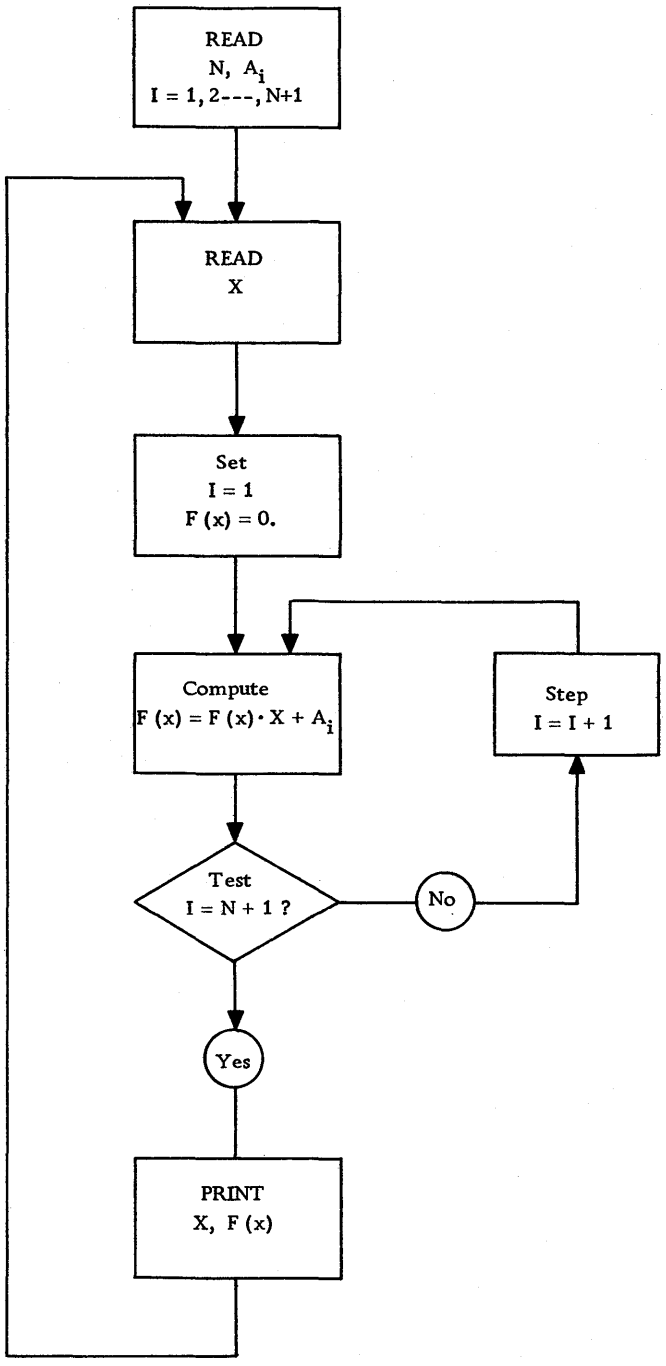
C FOR COMMENT

| STATEMENT NUMBER (1–5) | Cont. (6) | FORTRAN STATEMENT (7–72) |
|---|---|---|
| C | | EXERCISE 4 |
| C | | TRIGONOMETRIC FUNCTIONS |
| 1 | | TANX=SINF(X)/COSF(X) |
| 2 | | COTX=COSF(X)/SINF(X) |
| 3 | | SECX=1.0/COSF(X) |
| 4 | | CSCX=1.0/SINF(X) |
| C | | INVERSE TRIGONOMETRIC FUNCTIONS |
| 5 | | ASINX=ATANF(X/SQRTF(1.0-X*X)) |
| 6 | | ACOSX=ATANF(SQRTF(1.0-X**2)/X) |
| 7 | | ACTNX=ATANF(1.0/X) |
| 8 | | ASECX=ATANF(SQRTF(1.0-X*X)) |
| 9 | | ACSCX=ATANF(1.0/SQRTF(1.0-X*X)) |
| | | HYPERBOLIC FUNCTIONS |
| 10 | | SINHX=0.5(EXPF(X)-EXPF(-X)) |
| 11 | | COSHX=0.5(EXPF(X)+EXPF(-X)) |
| 12 | | TANHX=(EXPF(X)-EXPF(-X))/(EXPF(X)+EXPF(-X)) |
| C | | OR |
| | | E1=EXPF(X) |
| | | E2=EXPF(-X) |
| | | TANHX=(E1-E2)/(E1+E2) |
| | | |

Section 3. Exercise 4

EXERCISE 6

Generalized flow chart to compute $F(x) = a_1 x^n + a_2 x^{n-1} ---- + a_{n+1}$



## Section IV

C FOR COMMENT

| STATEMENT NUMBER (1–5) | Cont. (6) | FORTRAN STATEMENT (7–72) |
|---|---|---|
| C | | EXERCISE 4 |
| | | I=1 |
| | | X=.05 |
| | | SUM2=0.0 |
| | | SUM4=0.0 |
| 1 | | Y=1.0/(1.0+X*X) |
| | | GO TO (4,2),I |
| 4 | | SUM4=SUM4+Y |
| | | I=2 |
| | | GO TO 3 |
| 2 | | SUM2=SUM2+Y |
| | | I=1 |
| 3 | | X=X+.05 |
| | | IF(X-1.0)1,5,5 |
| 5 | | SUM=0.05/3.0*(1.5+4.0*SUM4+2.0*SUM2) |
| | | PRINT20,SUM |
| | | END |

Section 4. Exercise 4

EXERCISE 3

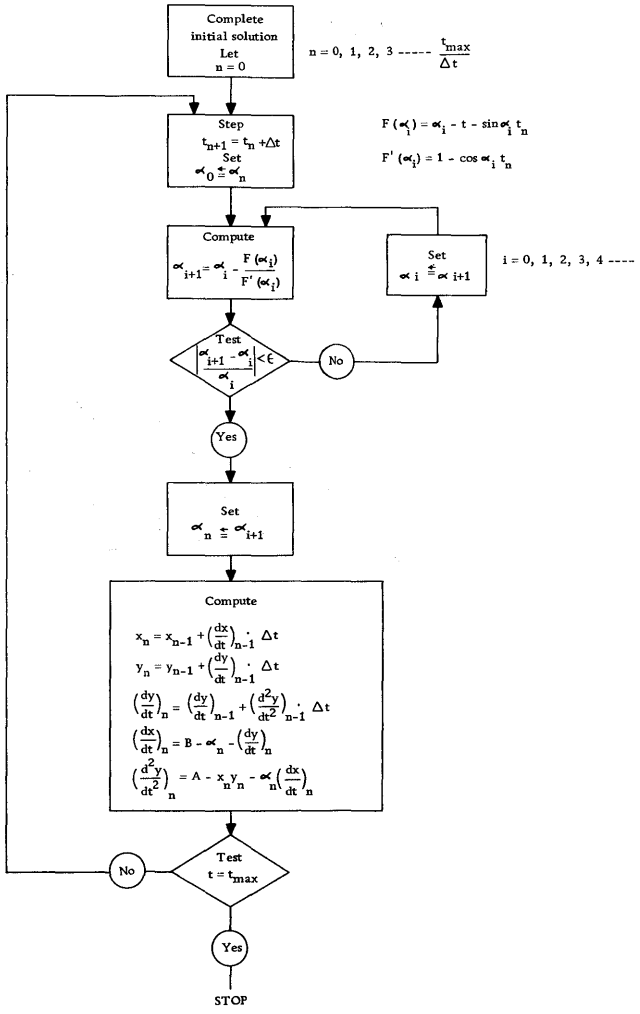
The initial conditions may be used to complete the solution at t = 0.

$$\alpha_0 = 0$$

$$\left(\frac{dx}{dt}\right)_0 = B - \alpha_0 - \left(\frac{dy}{dt}\right)_0$$

$$\left(\frac{d^2y}{dt^2}\right)_0 = A - x_0 y_0 - \alpha_0 \left(\frac{dx}{dt}\right)_0$$

Succeeding values for the variables and their derivatives may be obtained by two-level iteration. For each value of t, iteration is used to find a solution to $\alpha = t + \sin \alpha\, t$. The following logic diagram makes this clear:



## Section VI

### EXERCISE 2

Inverse is

$$\begin{matrix} -4 & 4 & -1 \\ 4 & -5 & 2 \\ -1 & 2 & -1 \end{matrix}$$

Solution is

$x_1 = 3$

$x_2 = -3$

$x_3 = 2$

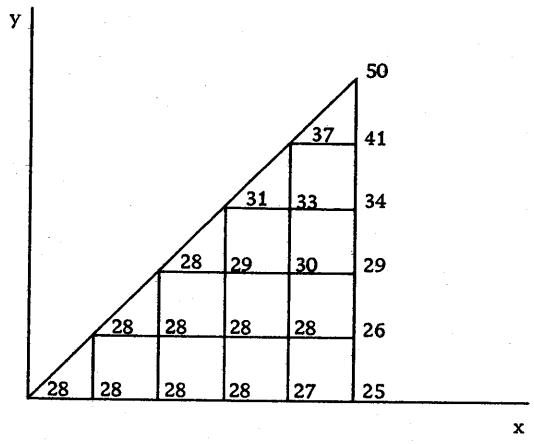
### EXERCISE 3

$x_1 = 2$

$x_2 = 5$

$x_3 = 9$

### EXERCISE 4



## Section VII

EXERCISE 3

$$t = \frac{x - m}{\sigma} = \frac{.0027 - .0025}{.0002} = 1.0$$

for t = 1.0, from table of probabilities

$$\frac{2}{\sqrt{2\pi}} \int_{2.5}^{\infty} e^{-t^2/2} \, dt \cong .1587$$

## APPENDIX II : BIBLIOGRAPHY

The Data Processing Bibliography (form J20-8014) provides abstracts of several excellent texts on numerical techniques, scientific applications and related subjects in the field of data processing. IBM publications of interest to the engineer include:

Programmer's Primer for FORTRAN Automatic Coding System for the IBM 704 Data Processing System (F28-6019)

Random Number Generation and Testing (C20-8011)

Flow Charting and Block Diagramming Techniques (C20-8008)

Introduction to IBM Data Processing Systems (F22-6517)

Inventory Management Simulation (E20-8063)

AUTOPROMT — Automatic Programming of Machine Tools (E20-6104)

Production Line Balancing at Westinghouse (E20-4037)

PERT — A Dynamic Project Planning and Control Method (E20-8067)

Optimum Shop Scheduling and Machine Loading (E20-4044)

Quality Control (320-0955)

Improved Job Shop Management through Data Processing (E20-6071)

F20-8077

IBM

International Business Machines Corporation

Data Processing Division

112 East Post Road, White Plains, New York